Wireless Inspirational Bits for Facilitating Early Design

DENIZ AKKAYA

Degree project in
Communication Systems
Second level, 30.0 HEC
Stockholm, Sweden
Wireless Inspirational Bits for Facilitating Early Design

May 6, 2013

Deniz E. Akkaya

Degree Project in
Communication Systems
Second Level, 30.0 HEC
Stockholm, Sweden
## Contents

1 Introduction ................................................................. 1
   1.1 Low Power Technologies ........................................ 2
   1.2 Role of Wireless Technology .................................... 2
   1.3 Internet of Things .............................................. 3
   1.4 Internet Services with IoT Vision ............................. 4
   1.5 Earl Toolkit ..................................................... 4

2 Background Work .......................................................... 7
   2.1 Need of a toolkit ................................................ 7
      2.1.1 What is a Toolkit ......................................... 7
      2.1.2 Sketching and Prototyping ................................ 8
         2.1.2.1 Sketching in Hardware and Software ............ 8
         2.1.2.2 Do It Yourself Technology ....................... 9
   2.2 Current Toolkits ................................................ 9
      2.2.1 Arduino ................................................... 9
      2.2.2 mBed .................................................... 10
      2.2.3 Other Toolkits .......................................... 11

3 Earl: Toolkit for Sketching ............................................ 12
   3.1 System Description .............................................. 12
   3.2 Design Objectives ................................................ 12
   3.3 Requirements .................................................... 14
      3.3.1 Earl Toolkit Requirements ............................... 14
         3.3.1.1 Hardware Requirements ............................ 14
         3.3.1.2 Embedded Software Requirements ............... 16
      3.3.2 Mobile Application Requirements ..................... 18
      3.3.3 Web Service Requirements .............................. 18
   3.4 Technologies for Earl .......................................... 19
      3.4.1 Wireless Communication Technologies .................. 19
         3.4.1.1 Bluetooth ......................................... 19
         3.4.1.2 ZigBee ............................................ 20
         3.4.1.3 WiFi .............................................. 21
         3.4.1.4 ANT ............................................... 22
      3.4.2 Low Power Technologies ................................. 23
      3.4.3 Low Power Units of Earl ................................ 25

4 Implementation .......................................................... 26
   4.1 Earl Toolkit .................................................... 26
      4.1.1 Hardware Design ......................................... 26
         4.1.1.1 PCB Design ....................................... 26
         4.1.1.2 Modules and Components Used in Hardware Toolkit 27
            4.1.1.2.1 Battery ...................................... 27
            4.1.1.2.2 TPS77333 Voltage Regulator Module ....... 27
            4.1.1.2.3 MSP430G2553 Microcontroller Module ...... 29
4.1.1.2.4 ANT nRF24AP2 Module .................................................. 30
4.1.1.2.5 DRV8601 Haptic Driver Module ..................................... 31
4.1.1.2.6 Connectors and LEDs .................................................. 32
4.1.2 Embedded Software Design ............................................... 33
  4.1.2.1 Embedded Software Architecture ..................................... 33
    4.1.2.1.1 Constants Module ................................................. 33
    4.1.2.1.2 Base Module .................................................... 34
    4.1.2.1.3 Flash Module ................................................... 34
    4.1.2.1.4 UART Module .................................................... 34
    4.1.2.1.5 ADC Module ..................................................... 35
    4.1.2.1.6 PWM Module ..................................................... 35
    4.1.2.1.7 Scheduler Module ............................................... 35
    4.1.2.1.8 ANT Module ...................................................... 35
    4.1.2.1.9 Test Module .................................................... 36
    4.1.2.1.10 Main Module ................................................... 36
  4.1.2.2 Wireless Communication Configuration for Hardware Toolkit ....... 36
    4.1.2.2.1 Earl Toolkit Wireless Communication Channel Structure ............. 36
4.2 Android Mobile Application ............................................... 39
  4.2.1 Android Mobile Phone ................................................ 39
  4.2.2 Android Software Development Platform ................................ 40
  4.2.3 Android Mobile Application Implementation ............................ 40
4.3 Earl Web Services .............................................................. 42
  4.3.1 Web Server Details ..................................................... 42
  4.3.2 Server System Implementation ......................................... 43
    4.3.2.1 Information Pushing .............................................. 43
    4.3.2.2 Information Request from Web Server .............................. 43
    4.3.2.3 Monitoring and Controlling Hardware Toolkit from Web Server ...... 43
4.4 Working System ................................................................. 44
  4.4.1 Hardware Toolkit Power Consumption ................................... 44
    4.4.1.1 Power Consumption In Receive Mode ................................ 45
    4.4.1.2 Power Consumption In Transmit Mode ................................ 47
    4.4.1.3 Power Consumption In Sensing Mode ................................ 49
    4.4.1.4 Power Consumption In Actuating Mode ................................ 51
    4.4.1.5 Power Consumption In Microcontroller Sleep Mode ................. 53
    4.4.1.6 Power Consumption Results ........................................ 53
  4.4.2 Data Exchange Latency ................................................ 56
    4.4.2.1 Hardware Toolkit - Mobile Phone ................................... 56
    4.4.2.2 Mobile Phone - Web Services ...................................... 57

5 Conclusion and Future Work .................................................. 60
List of Figures

1.1 Earl System Layout ................................................. 5
3.1 Earl Hardware Toolkit Communication With Earl Web Services ................. 13

4.1 Earl Hardware Toolkit ............................................. 27
4.2 TPS77333 I/O Pins ................................................ 28
4.3 MSP430G2553 I/O Pins ............................................ 29
4.4 ANT nRF24AP2 I/O Pins ........................................... 31
4.5 DRV8601 Haptic Driver I/O Pins .................................. 31
4.6 Connectors and LEDs .............................................. 32
4.7 Communication Packet Flow Between Hardware toolkit and Mobile Phone ........ 37
4.8 Earl Broadcast Message Structure .................................. 38
4.9 Sony Ericsson Xperia Active ....................................... 39
4.10 Hardware Toolkits Connection Example ................................ 41
4.11 Hardware Toolkits Connection Example with Web Services Connection ........ 42
4.12 Information Pushing ............................................... 43
4.13 Information Request from Web Server ................................ 44
4.14 Monitoring Sensor Data from Web Server ................................ 44
4.15 Controlling Actuator Connected to Hardware Toolkit from Web Server ........ 44
4.16 Current Consumption Measurement Setup ........................... 45
4.17 Current Consumption Measurement for Receiving Mode .................... 46
4.18 Current Consumption Measurement for Transmit Mode ..................... 47
4.19 Current Consumption Measurement for Shared Channel Slave Mode .......... 48
4.20 Force Sensor ....................................................... 50
4.21 Current Consumption Measurement for Sensing Mode ..................... 50
4.22 Haptic Actuator .................................................... 52
4.23 Haptic Driver Test - Current Consumption ................................ 52
4.24 Average Power Consumption Results ................................ 56
4.25 Earl Hardware Toolkit Communication Latency ............................ 57
4.26 Experiment Results for Communication Between Mobile Phone and EWS ........ 59
# List of Tables

3.1 Hardware Toolkit Requirements ........................................ 14
3.2 Earl Metrics .............................................................. 16
3.3 Requirements for Software of Hardware Toolkit ....................... 16
3.4 Metrics for Software of Hardware Toolkit ............................ 17
3.5 Earl Mobile Platform Requirements .................................... 18
3.6 Mobile Platform Metrics .................................................. 18
3.7 Earl Web Platform Requirements ....................................... 18
3.8 Earl Web Services Metrics .............................................. 19
3.9 Bluetooth Classes with Maximum Permitted Power .................... 19
3.10 ANT AP2 Power States @3.3 Supply Volt when 32KHz External Oscillator Used ........ 23

4.1 Parts used in Earl hardware toolkit .................................... 28
4.2 Earl Hardware Toolkit Channel Configuration ......................... 37
4.3 Earl Hardware Toolkit Channel Configuration ........................ 40
4.4 Earl Hardware Toolkit Mobile Phone Connection Example Device Parameters .......... 41
4.5 EWS Web Server Specifications ...................................... 42
4.6 Force Sensor Specifications ............................................ 49
4.7 Haptic Vibrator Specifications ....................................... 51
4.8 Haptic Driver Test Inputs and Current Consumptions ................ 52
4.9 Hardware Toolkit Peak Power Consumption Results .................. 54
4.10 Hardware Toolkit Average Power Consumption Results ............. 55
4.11 Mobile Phone and Web Server Latency Test .......................... 58
Abstract

Proof of concept systems are the significant examples of technology designs to test and analyze technology ideas. Toolkits enable technology developers to decrease the complexity and time to create draft designs with many iterations. Toolkits with combinations of sensors and actuators may lead technology designers to new design spaces and opportunities by decreasing design difficulties.

This thesis proposes a new wireless toolkit, called Earl, for beginner level technology designers. Without being experienced on engineering skills Earl allows technology designers to create proof of concept systems in shorter time than implementing the system from scratch. For experiencing with sensors and actuators, this toolkit has a vision to design plug and experience systems. By using Earl, designers can connect analog actuators and sensors by plugging them to toolkit. Additionally, Earl is designed to increase the experience time by lowering the power consumption as much as possible. Therefore, application developers can use Earl portably in long term experiments. Furthermore, Earl is designed in Internet of Things concept. By using Earl web services, technology designer can access the data sent from the sensors and can control the actuators connected to hardware toolkit from a web browser.

Earl hardware toolkit aims to provide long experiment time as much as possible. This is why, Earl hardware toolkit is designed to lower the power consumption if hardware toolkit is not actively handling processes. In sensing only mode, technology developers can use Earl hardware toolkit more than 1 month without recharging the toolkit battery. Using actuators in application may change power consumption of the hardware toolkit. Therefore, applications with actuators may have lower experiment time.
Acknowledgements

First of all, I would like to thank to my supervisor, Prof. Mark Smith for his invaluable support in every stage of this thesis work. I learned a lot from his experiences, but, maybe the most important one is he showed me how to think analytically about real life problems.

Secondly, I also would like to thank to all Mobile Life Centre and Swedish Institute of Computer Science (SICS) researchers, especially Prof. Kristina Höök, Petra Sundström, and Jordi Solsona, for their contributions and providing me fantastic opportunities.

Finally, I am very greatful to my family and my love, Ece Mandiracioglu, for their endless support on any single moment of my life. It would not be possible without them.
Chapter 1

Introduction

Prototyping is a significant activity of technology development that helps developers shape the technology design. Technology developers can experience and get inspired by draft designs, discover innovative solutions and produce better technology by iterations. This is why, design concepts may assist technology designers to improve the technology implemented. Prototypes are working examples of a technology that enables developers to test and analyse ideas. Technology developers may develop or improve prototypes by using a system, which can facilitate designers’ effort as well as shortening design time for development and prototyping [1]. These kinds of systems are called toolkits. They are composed of both hardware and software modules allowing application developers to design, test and break down the implemented technology.

Technology developers use sensors to sense the environment where the sensors are placed and use actuators to have the ability to react to changes in that environment. Combinations of sensing and actuating systems enable technology developers to create variety of applications for technology designs. This is why, toolkits lead to new design spaces and opportunities for developers by enabling technology developers to test design ideas with sensors and actuators easily and without dealing with complexities.

Mobility is a feature which is getting popular in toolkit applications. In mobile systems, mobile devices can function in a certain distance without cable limitations. For some applications, mobility is a desired requirement. For instance, in human body related applications, position of a toolkit in 3D space may change time to time. In this case, cables may get broken or limit the movement of the toolkit. This is why, developers prefer to use wireless communication between the connection points to move freely without cable limitations. However, mobile devices have limitations with battery lifetime. Wireless communication is a more power consuming technology than wired communication. Moreover, wireless communication may affect the application quality in terms of communication speed and communication range.

Developers have been using mobile toolkits for development and testing especially for early design stages [2]. In the beginning of technology development stages, developers examine many ideas and implement draft solutions for existing problems by using toolkits. The goal of this thesis is to introduce a wireless toolkit that helps technology developers to experiment with most common analog sensors and actuators in a rapid way and without dealing with implementation complexities. In order to increase experiment time with this toolkit, it is intended to decrease power consumption of the toolkit as much as possible. To help to decrease the power consumption, low power technologies are used. By using low power and wireless features\(^1\), technology developers can create portable systems with longer battery life. In the coming two sections, I will talk about benefits of low power technologies and role of wireless technology. Then, I will explain concept of “always connected” devices and advantages of merging systems with Internet applications. Finally, I will introduce Earl Toolkit and explain how Earl is working.

\(^1\)In the Section 3.4.2 and Section 3.4, low power technologies are explained in details and compared with other technologies.
CHAPTER 1. INTRODUCTION

1.1 Low Power Technologies

Low power technologies have a variety of benefits for mobile toolkits. First of all, low power consumption supplies longer battery life. Especially mobile toolkit applications’ battery life is an important parameter for the experiment duration. For some mobile toolkit applications, testing may take a long time, thus application users can not test draft solutions properly without recharging the battery, which may conclude in wrong experiment results due to the interruption of the testing.

Secondly, low power technology decreases the application costs. Due to high heat generated in high power consumption systems, the system requires more complicated, thus costly, silicon structures. Furthermore, low power consumption decreases the cost since smaller battery supplies cost less.

Moreover, longer experience time with draft technology solutions, smaller size of toolkit, low cost of the proposed solution, and portability are desired requirements of mobile toolkits used in the early design processes. This is why, low power consumption is a targeted characteristic of mobile toolkits.

1.2 Role of Wireless Technology

Wireless communication has great opportunities. However, it has some drawbacks as well. In this section, first I will compare wired and wireless communication. Then, I will explain why I used wireless communication technology which is a desired featured of toolkits.

Firstly, wired connection has less power consumption in compare to wireless solutions. Most wireless technologies requires extra modules, components or peripherals such as antenna circuit and technology related Integrated Circuits (IC) instead of single wire. This is why, wired solutions are cheaper solutions.

Security is another significant challenge for wireless systems. Most wireless technologies have vulnerabilities, such as unauthorized access or damage to wireless devices using wireless network. If the exchanged information has high confidentiality, wireless communication should be secure enough to operate. On the other hand, wire communication is isolated from its ecosystem. Therefore, wired communication is much safer than wireless solutions.

Wired communications are more reliable communication compare to wireless connections in terms of communication interference. There are shielded cables available to prevent interference by noisy machineries. For instance electric motors may interfere wireless communication devices. This is why, wired communication is less problematic in terms of interference. However, wireless communications may interfered with other wireless networks which are operating in the same frequency range. Physical conditions such as temperature, environment permeability and acoustics also effects wireless communication quality. Because of these reasons, exchanged information can be corrupted during the wireless transmission. This may cause delay or miscommunication in wireless connection.

In computer networking, bandwidth term is used for consumed information resources in bits per second for communication. Wireless communication has limitations in terms of bandwidth whereas wired communication has much higher bandwidth limits. Wireless transmission speed depends on the technology implemented. Used modulation techniques and methods directly affect the communication quality[3]. Physical and environmental conditions, for example electromagnetic waves generated from other devices, interfere wireless communication. Microwave ovens, which use radio waves to heat food, interfere most commercial wireless communication since they are both working at commercial 2.4 GHz operational frequency range. On the other side, wired communication speed depends on quality and material, length and size, and shield of cable.

Mobile devices exchange information between two or more devices that are not connected by a wire. Portability and communication access in limited regions, enable greater flexibility and mobility for application users. Technology users can relocate the communication devices without the additional cost of rewiring and excessive downtime while moving. Toolkit users prefer to use wireless communication in applications where relocation of toolkit is possible.

Short distance wireless communication may be useful where physical connection is not available between the connection points. Wireless communication is mostly used in cases where the communication device has to be able to communicate with other devices without cable limitations. Therefore, portable devices can communicate with each other in a certain range where wireless communication technology operates.
Another benefit of wireless technology is having less cables around portable devices. Many technology developers don’t like junk of cables, because cables prevent the device to change design aesthetics. Moreover, in applications where a portable device is attached to human body, wires may prevent human body to move naturally. Thus, wires may change human behavior while application user is experimenting which may lead to false results. Since technology development market has need of portable devices, wireless communication technologies are available for near field applications.

Until recently, most sensor and actuator based applications have been using wires or logging systems to transfer information or obtain data from a sensor node to application users. In the past, wireless communication was also available to exchange information between mobile devices. However, amount of power consumption was limiting the battery life; therefore, application developers could not use wireless mobile systems in the long run.

The ultra low power technologies mainly serve the need of targeted device to operate for longer time durations. There are ultra low power wireless communication technologies available in the market for near field communication. In low power wireless communication technologies, communication range can be dependent on many factors; such as wireless communication operation frequency, antenna type, antenna power, antenna length, environment, communication protocol and many more. Currently in the market, most near field communication range for ultra low power technologies vary from couple of centimeters up to order of 10 meters. Technology designers should consider wireless communication range according to specifications of the application.

To sum up, wireless communication may be more advantageous than wired communication. In the early design phase, developers can create working proof of concepts of intended application by using toolkits. If this toolkit has a low power wireless communication module, toolkit users can create working proof of concepts rapidly and can exchange information with other devices in a close range without cable limitations.

1.3 Internet of Things

While technology is advancing in great number of fields, wired and wireless networks are enclosing around us. Innovations in technology create a new trend in Internet world that is called “always connected” [4] concept. New Internet technologies such as IPv6[5] and push notification[6] allow for instant communication and unique Internet addresses for technology devices.

A concept, in where physical objects becomes part of the Internet, is called Internet of Things (IoT)[7]. IoT concept enables physical objects to be accessible through the network, therefore information like status, position, and location of physical objects are part of Internet services and intelligence of physical things [4]. Linking physical objects to Internet Network enables experiencing with everyday objects in personal and social environments.

Toolkits for rapid developments empower technology experiences by adding short range mobile communication in IoT concept. Technology developers can explore new ideas by adding wide range of additional gadgets[8] and everyday items. Therefore, developers link people and digital design materials by using mobile communication in anytime and anyplace.

First concept in IoT is designed for object tagging [4] however, then the IoT concept became more colorful with sensors and actuators. After International Telecommunication Union’s first announced report in 2005 [4], the aim of IoT was taken the current shape. New dimensions with sensors and actuators will be approachable from anytime, any place with connectivity for anyone. This means that real objects could be connected to each other. Moreover, connections will multiply and create an entirely new dynamic network of networks.

Cellular phones became light-weighted, more skilled and part of the Internet network. Many companies focused on interactive products so that technology users started to open a space for these devices in their life. All these changes consequenced to boom in Internet of Things world. In the last years, there are important milestones in silicon based technologies. As a consequence of price fall of the transistors for the last ten years, silicon vendors have started to produce low cost and high performance new technologies; such as Bluetooth[9], ZigBee[10], and ANT[11]. With the help of enhanced technology, new wireless solutions and communication protocols are introduced. Internet can become a more social and commercial platform than before by connecting everyday objects with our lives. Wireless connection may be a more
CHAPTER 1. INTRODUCTION

proper solution to connecting all everyday objects in IoT concept by removing cable limitation. By using IPv6, connecting 50 billion different objects together is possible on the Internet network[12].

Accelerated developments in an interactive digitalized world have led to an increase in the interest of sensing and actuating experiences with the concept of IoT. Designing interactive platforms with the environments that are supported by hardware, software and applications are complicated processes. Difficulties in designing digital material, such as sensors and actuators, and material properties make harder to examine the interaction between the environment and the digitalized material. Rapid growth of technological opportunities may improve, limit or help design processes. Thus, technology developers can get experienced and inspired by digitalized material such as sensors and actuators by the help of toolkits.

1.4 Internet Services with IoT Vision

IoT concept systems have a vision to distribute work balance to all connection nodes in the network. IoT comprises of four major concepts [13]: distributed computing, distributed information processing, basic wireless networking and automatic identification. Every device in IoT vision has a network connection and a service provided by Internet. According to Kuniavsky [13], IoT’s vision is, instead of just connecting devices over a network, creating new experiences according to developers’ interests by using Internet services. Internet services add values to users by having more interactive experiences; such as well connected ecosystems and information exchange between connection points.

Technology designers can connect different Internet services and applications to explore different ideas. IoT concept enables technology developers to convert technology devices into tools that technology users get the Internet services. Therefore, technology devices can get a shape according to Internet services [13]. Using combination of Internet services provided by a back hand supporter such as a cloud system[14], enables developers to experience technology immediately with the possible combination of inexpensive things. Therefore, it is a profound change that physical actions can transform into knowledge rapidly on the cloud system by the help of IoT vision. Through this vision, deploying ubiquitous wireless systems and spreading information processing enable technology developers to create enormous number of applications.

1.5 Earl Toolkit

Power consumption is a limiting factor for mobile devices in terms of being portable. This is why, I took the approach of decreasing power consumption of the toolkit for the sake of increasing the time of use. Furthermore, I also aimed to exchange information via wireless technologies for portability. In this thesis, I will focus on short range communication with low power facilities. Therefore, I implemented a portable wireless toolkit that exchanges information with mobile devices in a short range. By the help of this toolkit, technology developers can experience with most common analog sensors and actuators for a longer time than other near field communication technologies without cable limitations. This system also enables developers to create quick implementation of small systems without dealing with embedded software development challenges.

In this thesis, I proposed a solution which is a hardware and software toolkit. Designed system can be divided into 3 main parts. First part is Earl, a wireless mobile device that technology developers can rapidly connect to and experience with analog sensors and actuators. Second part is a web platform. Earl can not only communicate with other Earls but also can exchange information with a web platform. Web platform is used both for storing the data received from Earl and for controlling Earl itself. Last part is Android [15] mobile phone which is used as a connection bridge between Earl mobile devices and Web Services. Figure 1.1 shows the general system layout for Earl toolkit.

Most early design toolkits help technology developers to create systems to prove the concept by programming microprocessors or micro controllers. However, designing an embedded system application may require embedded systems knowledge and software coding skills. Therefore, developers may have limitations to create a system. In most applications that sensors control actuators, developers should write applications for the embedded device. However, Earl helps developers to obtain analog sensor data and control actuators by a web interface or an Android mobile application without writing embedded software
code. Therefore, developers will not have to deal with hardware system design and low level embedded programming. Thanks to Earl, technology developers can implement small example applications rapidly without dealing with complexities. Earl not only decreases early design development process but also enables technology developers to focus on interaction with idea instead of technical implementation. For example, let’s consider a technology developer who wants to experience with a balloon. She intends to measure the pressure inside the balloon. According to how much pressure inside the balloon, vibration on a vibrator is adjusted. Normally, measuring the pressure inside a balloon without using a supportive system takes an effort. Even more, if the technology developer has no engineering background, it may take too much time and effort to design, test and experience with a system. Earl toolkit enable technology developers to implement small systems as like this experiment. In order to implement this experiment with Earl, application developer should connect an analog pressure sensor and a vibrator to Earl. Next, technology developer should set input pressure sensor to control the output vibrator on Android mobile application. Finally, application developer should connect a battery to Earl, place the Earl into balloon and start the experiment on Android application. In this way, application developer can experiment with a balloon and a vibrator rapidly without need of writing embedded software code.

Developers can create web applications that can exchange information with Earl system. In addition, developers can create systems that can communicate with multiple Earl toolkits through web platforms in IoT concept. Therefore, by controlling one or multiple sensors, developers can experience with one or multiple actuators through Internet communication. By the help of Internet communication, developers can collect information and control environment in different locations.

This thesis report is dedicated to proposing the Earl toolkit as an alternative system for technology developers to create rapid sensor and actuator applications that developers does not have to deal with low level engineering difficulties in the early design processes. The rest of this report is organised as follows: Chapter 2 explains what a toolkit is and why it is a useful device for early design processes. Moreover, Chapter 2 introduces and compares some toolkits that technology developers are using to create prototypes and proof of concepts with their benefits and drawbacks in terms of low power consumption, wireless communication and ease of use. Chapter 3 explain goals of design, requirements, main units
CHAPTER 1. INTRODUCTION

and technologies used in the development of Earl Toolkit. Chapter 4 explains implementation of Earl hardware toolkit. Chapter 4 explains Android mobile platform that supports the Earl toolkits to connect to Internet. Web services platform, which enables application developers to connect their applications to web applications, is also explained in Chapter 4. In the end, Chapter 5 draws the conclusion and suggests possible future works about Earl toolkit project.
Chapter 2

Background Work

2.1 Need of a toolkit

2.1.1 What is a Toolkit

Technology designers create draft designs in order to get experienced and inspired. The produced draft design in the early design stages is called prototype. Prototypes are the significant proofs of a design to test and analyze ideas. During the design process, technology designer discovers innovative solutions and produces better technology by many iterations. The repetitive process of refining the design may take quite a long time and much effort. That is why, technology developers get help from development aid systems. A system which helps to develop or improve prototypes with less effort is called a toolkit.

As part of a product development process, designers create examples of the early stage conceptual approaches in order to experience how a product might work, rather than creating fully functional systems. While technology developers are implementing complicated systems, first they design small sub systems and later they merge small systems together. In this way, technology developers first experience with small systems then add complicated functions and details later. That is why, technology draft designs are supposed to be lightweight, user-centered, agile and focused on users’ experiences initially in order to explore new ideas without technologic limitations and complexities[16]. In the next stage of early design, designers merge satisfied draft designs and design as a proof of concept, named prototype. During all these phases, toolkits play significant roles in shaping the design by simplifying complexities, supplying pre-designed modules, and showing example applications.

There are many toolkits available for different application purposes. Technology developers should select a toolkit based on the application requirements. In this thesis, I focused on toolkits which aid designers to monitor the changes in toolkit’s environment by sensors and react toolkit’s environment by actuators. Designers can use this kind of toolkits to create intelligent everyday objects by attaching toolkits to them. For example, designers can use toolkits to understand the change of temperature on the table surface or acceleration of a soccer ball. In an early design process, technology developers can create a rapid proof of concepts by using toolkits instead of implementing the whole design from scratch. Therefore, by using toolkits, technology developers can experience with draft designs without dealing with complicated details.

Toolkits are used to help developers to consume less time and effort during the design process. Many toolkits are combined with hardware and software for supporting designers to provide ease of use. Hardware toolkits may provide jacks or clips to connect sensors and actuators to the toolkit. Thus, designers can test and experience small systems without soldering peripherals. Some toolkits come with default modules to decrease implementation time. By using default modules, developers don’t need to make an effort for implementing all parts of a desired system. Software toolkits may provide easy code development, better code readability, code compatibility with multiple hardware devices, and rapid design implementation. By using software toolkits, developers can reduce the complexities during the design process and don’t have to deal with low level software design.
2.1.2 Sketching and Prototyping

Sketches and prototypes are both core components of design processes [16]. However, they are used in different phases of the design processes. Sketching is an activity to project abstract ideas into draft designs in order to experiment with the ideas. Designers create sketches to obtain better designs during early design processes. These sketches can be small systems that demonstrate how designers’ ideas work. Designers implement physical designs of their ideas. This physical designs are called sketches. These sketches can be get shaped after iterations to form a proposed solution. These proposed solutions are called prototypes.

Sketches are dominantly used in early design processes because of cost efficiency, timeliness, quantity, and disposability. Prototypes are formed by many iterations of sketching. This is why, sketches take less time and are cheaper. Sketches have less property than prototypes because sketches are subset formed ideas of prototypes. Furthermore, designers create sketches to be influenced from draft designs. Thus, designers use sketches to explore new concepts and form the prototypes.

Prototypes are more comprehensive instantiations than sketches since they contain many ideas and cost more than sketches. Sketches are faster ways to experience ideas. Technology developers spend less time to create a sketch. Moreover, designer can produce many samples of sketches with less cost and in more quantities. In addition to this, a technology developer can restore the sketch easily, sketches have lower complexity than prototypes. Prototypes may not exactly resemble the design ideas in the technology development process, but sketching is a significant process in the technology design procedure because of its contributions to development.

Basically, we can compare sketches with prototypes in the following terms [16]:

- Sketches are imitative instantiations of design ideas. However, prototypes are functional instantiations of design ideas.
- Sketches may suggest new arguments to technology designers, while prototypes describe functionality.
- Technology designers create sketches for exploring new ideas and concepts. However, prototypes are refined forms of ideas and concepts.
- Sketches propose a potential solution to a question. On the other hand, developers use prototypes to test a concept.
- Sketches are tentative instantiations of a concept. Therefore, they provoke designer to create different concepts. However, prototypes are more specific designs than sketches. Thus, technology developers create more detailed concepts by using prototypes.

2.1.2.1 Sketching in Hardware and Software

In the early stage of technology design process of ideation and iteration, technology designers explore different solutions after many iterations [17, 18]. Designers are looking for feasible and desired solution for the technology design, whereas developers use hardware and software as design materials to create sketches [19]. However, it may be difficult for designers to expose and experience with design material to understand the technology [19].

To take an advantage of a design material, technology developers might improve hardware and software in innovative ways [19]. In order to create a possible, desired and acceptable idea by using a digital design material, technology designers should form hardware and software sketches for imitative instantiations of the desired concept. That is why, technology developers use toolkits to create a small system rapidly in order to experience with design materials.

Using digital design material such as sensors and actuators supported with software may require deep knowledge of engineering and high programming abilities. Difficulties in technical skills may limit the technology developers. That is why, a toolkit for sketching in hardware and software should be user friendly to designers who does not have technical background on digital design materials.
2.1.2.2 Do It Yourself Technology

Most available toolkits are used by the technology developers who want to create the whole system from scratch. Application developers may consider building, modifying and fixing the developed draft systems without using predesigned systems. Developers, who are producing the technological designs without using almost any kind of pre-assembled parts, called Do It Yourself (DIY) developers [2]. Therefore, iterations for sketching and prototyping became slow and expensive. Instead of developing every small detail in a draft system for early design process, application developers should use pre-made parts, modules, and software libraries to experience the technology as soon as possible. Although, technology developers will be limited with non-customized systems, sketches will be working conceptually with low cost in a short time period.

Although, DIY designs may have higher performance since they are designed for a customised project instead of a general purpose, they take designs takes more effort and time; moreover, finding mediums cost as well [2]. This is why, application developers should select pre-made systems, modules and libraries which are compatible with the toolkits in their sketching.

Today, technology designers can use different toolkits to speed up early design processes. In the following chapter, most commonly used early design toolkits will be compared with Earl toolkit in terms of power consumption, mobility and ease of use.

2.2 Current Toolkits

Currently, there are many hardware and software toolkits that assist technology designers to explore new ideas and experience new concepts. In this section, most commonly used toolkits for sketching and prototyping will be discussed in terms of power consumption, mobility, and ease of use.

2.2.1 Arduino

Arduino [20] is one of the most popular sketching and prototyping toolkit for technology developers [21]. Arduino is an open-source platform that lets technology designer to build standalone controllers with or without a PC connection. Developers can select specific Arduino boards according to the application. For example, there are Arduino boards available for sensor applications, actuator applications, wearables and textile applications, and wireless communication related applications and more.

Arduino boards have input and output pins that enable developers to connect external peripherals and modules. Technology developers can find compatible sensors and actuators, modules and peripherals according to developer’s specific applications on commercial web sites such as SparkFun [22]. On web sites like Sparkfun, designers can find variety of sensors and actuators that are compatible with Arduino. Developers can connect analog and digital sensors to Arduino board and experience it by the example code supported by Arduino’s web site. Arduino also supports many actuators such as different kinds of motors, vibrators, and speakers.

Arduino is supported by multi-operating systems on PC and Unix computers. Technology developers can test their applications on most common operating systems such as Windows, Mac OS, and many Linux distributions. Arduino has pre-written libraries to support application developers. By the help of pre-defined functions and variables, technology designers may not have to use too much low level programming compared to PIC [23] programming.

Arduino boards have 16-bit and 32-bit Atmel [24] microcontrollers. More processing power needed applications can be implemented on 32-bit Atmel processors. For example, Arduino Due has 32-bit Atmel CPU based on ARM’s [25] Cortex M3 Architecture[26] which is operating at 84 MHz. On the other hand, some Arduino boards are designed for low power consuming applications such as battery critical applications.

Power consumption is a matter of design issue for Arduino application developers. For mobile prototyping and sketching, designers should consider power consumption in order to increase battery lifetime, thus application experience time. Most Arduino boards are designed for cable powered applications. This is why, for stable or power cable limited applications developers don’t have to consider low power consuming modules. On the other side, for mobile applications, developers should consider less power
CHAPTER 2. BACKGROUND WORK

10

consuming peripherals and boards, bigger battery size, and software oriented power optimisation modes such as sleep mode.

According to my experiments one of Arduino’s basic level boards, Arduino Uno consumes 47 mA current by applying 9 Volt input voltage to board. Arduino board programmed as doing nothing and having an empty while loop. My setup to measure power consumption of Arduino is as follows: I connected Sinometer MY-68 Multimeter [27] to 9 Volt DC supply and Arduino Uno board serially. On the current reading mode, I measured current consumption of Arduino Uno board. By using equations on 5, I calculated power consumption of Arduino board as 0.423 Watt. According to Arduino’s specifications, Arduino Uno board can consume current up to 500 mA operating on 9 Volt, thus 4.5 Watt power consumption[28].

Some Arduino kits have available pins to connect wireless modules such as Wi-Fi and Bluetooth modules to Arduino boards. Therefore, developers can exchange information wirelessly in their applications. In order to connect wireless module to an Arduino board, developers should connect module’s output and input pins to Arduino’s general purpose input and output pins. Then, application developer can download module related source code from Arduino’s or module’s web site and load the code to Arduino board. Developers can change the code behaviour by editing the downloaded code from Arduino’s web site. Furthermore, developers can connect multiple modules and create more complicated systems by using Arduino toolkits. However, developers have to know how to connect modules to Arduino boards and program Arduino boards by sample code provided on Arduino web site. Some technology developers may have difficulties to create wireless application because of lack of knowledge and practical experiences. In this case, developers can get help from Arduino’s forum web site[29].

Arduino is a toolkit for developers who has a background on electronics and programming. However, freshman developers also can implement small systems by doing tutorials on web site. Connecting peripherals and communicating with connected peripherals are much easier by using Arduino’s supported software framework. Developers who want to have long experiences with Arduino in mobile applications, should consider lower power consumption Arduino boards since power consumption of Arduino boards are high relatively to other toolkit solutions such as mBed[30].

2.2.2 mBed

mBed[31] is an online platform for rapid prototyping system with micro controller-based systems. It has a 32-bit ARM main processing unit, which is powerful enough for many applications such as sensor and actuator based applications, signal and sound processing applications, and motor control applications. mBed is a breadboard attachable board with many digital and analog inputs. Technology designers can attach analog and digital sensors to mBed boards. Additionally, analog pin outputs are also available to control most actuators.

mBed has a web site [30] that contains software development platform for prototyping applications. In this web site, developers can write software by using the supplied software framework and test with mBed’s prototyping boards. Currently mBed has 2 board selection with different ARM processors. mBed NXP LPC11U24 is for prototyping USB (Universal Serial Bus) Devices and battery powered mobile applications. mBed NXP LPC1768 is more convenient for prototyping Ethernet, USB Host Devices applications, and power needed applications. Both boards have different processor models and soldered components on the board therefore power consumption of boards are different.

LPC1768 board does not consuming less power than Arduino Uno board. According to my experiments I calculated LPC1768 board consumes 141 mA when it is working on 5 Volt which is supplied by a USB connector. In order to measure the current consumption I followed the same path as I did for Arduino Uno board. I programmed LPC1768 board for an empty while loop therefore board was not doing any process. According to my experiment I calculated power consumption as 0.705 Watt by using Equation 5.2. LPC1768 board supports Ethernet connections. However, Ethernet module on board consumes so much power. In this experiment, I turned off Ethernet features. On the other hand, when I did the same experiment with Ethernet features enabled, I measured 256 mA current consumption while LPC1768 is supplied with 5 Volts. Therefore, power consumption is calculated as 1.28 Watt according Equation 5.2 given in Section 5. On the other side, mBed has another low power prototyping board called LPC11U24. According to specifications given in mBed’s web site [32] LPC11U24 may consume 16 mA with 3.3 supply voltage.
LPC1768 is a board for power required applications such as motor control and ethernet communication applications. On the other hand, LPC11U24 is a less power consuming board than LPC1768 board such that application developers can use LPC11U24 board on mobile and low power required applications. There are many wireless applications examples available for LPC11U24 board on mBed’s web site. Technology developers can create small systems with low power wireless communication technologies by using this board.

As an example, technology designers can create small applications by obtaining a wireless communication module, sensors and actuators that are compatible with LPC11U24. Most compatible modules are listed on the board software library. Designers can use examples and sample codes on the web site to setup test environments for design concepts. However, in order to design complicated systems, mBed requires technical background for developers. To sum up, mBed is a toolkit for engineers to create small systems easily by using prewritten libraries and sample codes. LPC11U24 board enables designers to implement mobile and battery powered applications with long battery life compare to LPC1768 board.

2.2.3 Other Toolkits

Modkit is a toolkit for developers who have beginner level engineering skills [33]. It has visual block diagram interface for developers who program popular hardware toolkits such as Arduino. By using Modkit, designers can implement basic sensor and actuator applications without writing software code. Modkit is a web application that enables developers to draw graphical structures for basic coding functions such as loops, variable assignment, and fetching data from a peripheral. Technology designers can read sensor data from sensors and control actuators by designing boxes. Designers who has no coding background can create small applications without knowing software coding. Furthermore, technology designers can see designed code in code view perspective of Modkit. Developers who have technical background can edit the code and reshape designed structure for complex algorithms.

On the other side, Modkit does not help application developers to lower power consumption and mobility. Developers who has programming skills can implement mobile communication features on code view perspective. However, Modkit is not a powerful toolkit for designers who wants to create rapid mobile systems by lowering the power consumption.

Wiring is an open-source development platform for Wiring boards that developers can control attached peripherals to Wiring toolkit[34]. Wiring has special supportive libraries that makes easier to program Wiring microcontrollers. It helps developers to increase code readability and decrease code complexity. Developers can connect analog and digital sensors and actuators to Wiring toolkit. By the help of Wiring software development platform, developers can exchange information with sensors and actuators without writing low level software code. Designers can also connect Wiring to a computer by using a USB cable. Wiring software toolkit supports sending and receiving information through a computer. Thus, developers can merge small systems that they designed with computer applications.

Little Bits is a very simple but powerful toolkit for sketching [35]. Application developers can merge small magnet modules to create simple but effective systems. However, all used modules in Little Bits are analog and they are not re-configurable. In addition to this, application developers can only experience with sensors and actuators supported by Little Bits.

Makey Makey [36] is a toolkit that designers can connect daily object to Makey Makey toolkit and control computer applications. Furthermore, developers can also control popular toolkit platforms such as Arduino. Makey Makey has ability to control basic keyboard buttons such as arrows and control buttons. Developers can control a PC application by pressing pre-assigned buttons on Makey Makey toolkit. For example, developers can use a banana to control a key on computer keyboard by using Makey Makey. By using Makey Makey, technology developers can create rapid systems without dealing with technical complexities.

In this chapter, I introduced what a toolkit is and why it is playing a significant role in an early design stage of technology development. Then, I presented most commonly used toolkits that designers are using for creating small systems. In the next chapter, I will talk about objectives and requirements of Earl toolkit. Furthermore, I will also explain main units and technologies used for Earl toolkit.
Chapter 3

Earl: Toolkit for Sketching

In this chapter, first I will describe what the whole system ideally should be. Secondly, I will describe goals and vision of this toolkit. Next, I will explain requirements for the design and metrics of the toolkit. In the end, I will explain how I selected wireless communication technology and low power microcontroller unit for Earl toolkit.

3.1 System Description

Earl is a toolkit that application designers can create proof of concept systems faster than implementing the system from scratch. In these systems, Earl enables technology developers to use analog sensors and actuators. One of the significant benefit of Earl toolkit is developers can connect sensors and actuators to Earl toolkit without soldering or wiring. In Earl toolkit, developers can exchange information between Earl toolkit and Earl web services through Internet. In order to connect Earl to Earl web services over Internet, Earl connects to an Android mobile phone wirelessly and then the mobile phone takes care of the data exchange between mobile phone and web services.

Earl is a mobile toolkit and it is working with a battery. This is why, technology designer can move Earl freely without cable limitations. Earl is designed to take advantage of low power technologies. Therefore, application designers can use Earl toolkit in long term experiments without re-charging or changing the battery.

In this thesis, I proposed a solution which is hardware and software toolkit. Designed system can be divided into 3 main parts. First part is Earl, a wireless mobile device that technology developers can rapidly connect to and experience analog sensors and actuators. Second part is a web platform. Earl can not only communicate with other Earls but also can exchange information with a web platform. Web platform is used both for storing the data received from Earl and for controlling Earl itself. Last part is Android mobile phone which is used as a connection bridge between Earl mobile devices and Web Services. Following Figure 3.1 demonstrates the system Earl Toolkit and its supportive systems, which are Android mobile phone and web services.

3.2 Design Objectives

This section explains target goals of the toolkit design. In the beginning of the project, I investigated possible toolkits that designers can use in an early design stage of technology development. These toolkits are described in Chapter 2.2. Based on these toolkits, I collected the requirements of a toolkit using which designers can create proof of concept systems. Among all toolkit requirements, I focused on those of sketching toolkits that designers can implement sensor and actuator applications.

The project’s design objectives to decide the requirements are as below:

- (DO1) Toolkit for sketching in hardware and software (TSHS) enables technology designer to create sketches faster than implementing by designing from the scratch. Most designers use toolkits instead of implementing the whole design from scratch. However, developers who have enough knowledge can create DIY systems. These systems can be much expensive and take more time than
implementing the system with a toolkit. This is why, technology developers use toolkits and ready
made systems to experience with design implementation rapidly such as Arduino and mBed.

- (DO2) TSHS enables application developers to create sketches in order to experience with sensors
  and actuators. Most toolkits support digital and analog inputs and outputs. By using inputs
  and outputs developers can connect sensors and actuators. Some toolkit boards can be more
  customized according to the application type. For example, most Arduino boards have analog to
digital converters. Therefore, application developers can connect analog sensors to the application
board and process the sensor data. Furthermore, some toolkits such as mBed’s LPC1768 board has
analog output that developers can directly connect this output to control analog actuators.

- (DO3) TSHS can be used for small experiments in order to get influenced from the sketches. Most
toolkits that I have introduced are used for prototyping such as Arduino and mBed. These toolkits
are designed to build refined forms of ideas and concepts. However, sketching toolkits are more
focused to explore ideas and concepts by trying different design materials. For instance, Makey
Makey enable designers to use many daily objects as preassigned keyboard buttons. Designer can
try many idea and design concepts rapidly without writing software.

- (DO4) TSHS has enough battery power that enables application user to experience technology
for enough time. TSHS can be used for long term experiments that takes hours or couple of days
without needing to charge or change the battery. Most toolkits introduced in Chapter 2.2 are
designed for stationary applications. Therefore, power consumption is not a significant issue for
these toolkits. On the other hand, developers may need toolkits that function for longer experiences.
This is why, power consumption of a toolkit can be important parameter for application developers.

- (DO5) TSHS is a light device that it is effortless to carry and attach anywhere. Designers may need
small sized and lightweight toolkits especially if implemented design is a portable system. Some
toolkits, such as Arduino Nano, have small size and customized for portable applications.

- (DO6) TSHS is a wireless toolkit that without cable limitations it can communicate with other
devices. Developers prefer wireless communication and battery powered applications in order to
design portable applications. Main vision for portable application is that mobile devices don’t have
cable limitations. Thus, they can move much easier. In addition to this, they can communicate with other devices, which support same communication technology, in a certain range. Many toolkits such as mBed and Arduino support wireless communication technologies for mobile applications.

- (DO7) Technology designer can access the data sent from the sensor and can control the actuators from a web browser. The TSHS allows the technology designer to merge the proof of concept design with different Internet services. Most toolkits are designed to handle the sensor and actuator data locally. However, by adding internet connection functionality to the toolkit, developers can exchange information through Internet.

### 3.3 Requirements

In this section, demanded requirements and specifications are given for the toolkit design. These requirements introduced below are proposed in order to fit the functionality of the toolkit. I will split requirements section according to main components of the system. These requirements are toolkit requirements, mobile application requirements, and web service requirements. I will also divide toolkit requirements into hardware toolkit requirements and embedded software requirements.

#### 3.3.1 Earl Toolkit Requirements

Earl toolkit consist of two main units: Hardware toolkit and embedded software which is designed for hardware toolkit. I will explain requirements below.

##### 3.3.1.1 Hardware Requirements

Hardware toolkit is the device that designers can connect sensors and actuators, and can exchange information with this device wirelessly. To maximize the mobility, lower the power consumption, enable ease of use, I have defined requirements on Table 3.1 and according to defined requirements I have defined specifications on Table 3.2 for the design process.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Related Design Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Earl is a mobile toolkit that is independent from stable power connection or wired communication with web services.</td>
<td>DO4, DO6, DO7</td>
</tr>
<tr>
<td>• Earl has small physical dimensions that can fit into technology designs effortlessly.</td>
<td>DO5</td>
</tr>
<tr>
<td>• Earl has a system that application designers can try different technologies without soldering or wiring.</td>
<td>DO1</td>
</tr>
<tr>
<td>• Earl has low power components that enable target users to use it for long term experiences.</td>
<td>DO4</td>
</tr>
<tr>
<td>• Earl has a framework that its functionalities can be controllable from distance</td>
<td>DO6</td>
</tr>
<tr>
<td>• Earl is a system that technology developers can test most common analog sensors and actuators rapidly.</td>
<td>DO2, DO3</td>
</tr>
</tbody>
</table>

Table 3.1: Hardware Toolkit Requirements
### Metric Value Units

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum number of analog sensors can be attached</td>
<td>1</td>
<td>Peripheral</td>
</tr>
<tr>
<td>Minimum number of analog actuators can be attached</td>
<td>1</td>
<td>Peripheral</td>
</tr>
<tr>
<td>Maximum hardware toolkit device dimensions (without battery)</td>
<td>$3.5 \times 2.5 \times 0.5$</td>
<td>cm$^3$</td>
</tr>
<tr>
<td>Toolkit battery life lasts with 150 mAh battery (1 Sensor only connected)</td>
<td>1</td>
<td>day</td>
</tr>
<tr>
<td>Maximum Wireless communication range</td>
<td>10</td>
<td>meter</td>
</tr>
<tr>
<td>Wireless ISM (Industrial, Scientific and Medical radio bands) band operation frequency</td>
<td>2.4</td>
<td>GHz</td>
</tr>
<tr>
<td>Wireless Maximum Data Rate</td>
<td>2048</td>
<td>bps</td>
</tr>
<tr>
<td>Maximum current consumption of hardware on Tx mode at 0 dB antenna power (1 analog sensor connected, 3.3 Operation Voltage, 1MHz, under normal conditions)</td>
<td>20</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum current consumption of hardware on Rx mode at 0 dB antenna power (1 analog sensor connected, 3.3 Operation Voltage, 1MHz, under normal conditions)</td>
<td>20</td>
<td>mA</td>
</tr>
<tr>
<td>Battery type and number of cells</td>
<td>Polymer Lithium Ion Battery 1 Cell</td>
<td></td>
</tr>
<tr>
<td>Battery cells output voltage</td>
<td>3.7</td>
<td>V</td>
</tr>
<tr>
<td>Min. battery capacity</td>
<td>150</td>
<td>mAh</td>
</tr>
<tr>
<td>Hardware toolkit operation voltage</td>
<td>3.3</td>
<td>V</td>
</tr>
<tr>
<td>System Master Clock Frequency</td>
<td>1</td>
<td>MHz</td>
</tr>
<tr>
<td>Maximum current supply from power source under normal conditions</td>
<td>30</td>
<td>mA</td>
</tr>
<tr>
<td>Maximum voltage dropout on voltage regulator unit</td>
<td>200</td>
<td>mV</td>
</tr>
<tr>
<td>Supply Voltage interval for microcontroller</td>
<td>1.8-3.6</td>
<td>V</td>
</tr>
<tr>
<td>Current consumption of microcontroller on active mode at 1 MHz @ 2.2V</td>
<td>230</td>
<td>$\mu$A</td>
</tr>
<tr>
<td>Current consumption of microcontroller on standby mode at 1 MHz @ 2.2V</td>
<td>0.5</td>
<td>$\mu$A</td>
</tr>
<tr>
<td>Current consumption of microcontroller on Off Mode at 1 MHz @ 2.2V</td>
<td>0.5</td>
<td>$\mu$A</td>
</tr>
<tr>
<td>Number of pins of microcontroller</td>
<td>32</td>
<td>Pin</td>
</tr>
<tr>
<td>Communication channels supported</td>
<td>SPI,UART,I2C</td>
<td>Communication Channels</td>
</tr>
<tr>
<td>Number of minimum channel support for wireless communication</td>
<td>1</td>
<td>Channel</td>
</tr>
<tr>
<td>Configurable periodic message transmit/receive rate</td>
<td>5.2-2000</td>
<td>milli second</td>
</tr>
</tbody>
</table>
CHAPTER 3. EARL: TOOLKIT FOR SKETCHING

- Maximum Message Receive/Transmit Rate: 32 Message Per Second
- Wireless operational frequency interval: 2400 - 2500 MHz
- Wireless operational frequency: 2468 MHz
- Antenna output power: 0, -6, -12, -18 dBm
- Antenna output power: -85 dBm
- Analog to Digital Converter (ADC) Bit Resolution: 8, 10 Bit per sample
- Max ADC Sample rate: 32 Sample Per Second
- Maximum A/D Conversion Time under normal conditions: <100 µs
- Maximum supported analog input channel count: 2 Analog Channel
- Maximum supported analog output channel count: 1 Analog Channel

Table 3.2: Earl Metrics

### 3.3.1.2 Embedded Software Requirements

Embedded software makes Earl a functional device. In the beginning of embedded software design, specifications are defined to determine the functionalities for a sketching toolkit. When I defined the embedded software requirements, I used the hardware toolkit requirements as base requirements of embedded software requirements. This is why, capabilities of embedded software depends on embedded hardware. In embedded software design, I focused on functionalities such as mobility, low power consumption, and ease of use. Requirements for software design are listed below in the table 3.3.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Design Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earl is a toolkit that has mobile features to communicate with Android devices which are ANT enabled. There should not need a coding effort to establish a communication between hardware toolkit and mobile phone.</td>
<td>DO6</td>
</tr>
<tr>
<td>Earl has a software that can control assembled components on the hardware toolkit.</td>
<td>DO2, DO6</td>
</tr>
<tr>
<td>Embedded software of Earl has adaptable interface to exchange information with most common analog sensors and actuators at the same time. Users should not design an embedded software to control the sensors and actuators.</td>
<td>DO2</td>
</tr>
<tr>
<td>Earl software framework has low power modes in order to decrease power consumption.</td>
<td>DO4</td>
</tr>
</tbody>
</table>

Table 3.3: Requirements for Software of Hardware Toolkit

The metrics of software for embedded hardware are given below in Table 3.4.

Earl hardware toolkit has 2 main functionalities which are transmitting fetched data from sensors to Android mobile phone and controlling actuators via the data received from Android mobile phone. Whole system architecture provides functionalities to exchange information with end points. In the following list, principals of embedded software is given.

- Earl enables application developers to test and analyze ideas about sketching by using a toolkit. In some cases, one toolkit may not be enough to create proper design sketching. Designer might be able to connect multiple Earl toolkits to one network. This is why, every Earl toolkit should have a unique ID number which should be already predefined and a Class ID that shows the category of Earl toolkit belongs. Users might be able to transmit or receive information to one whole class. Device ID number and Class ID number together should represent a unique ID of the toolkit and Earl toolkit should use this unique ID to communicate with other devices in the same network.
<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming language used in embedded software</td>
<td>ANSI C&lt;sup&gt;1&lt;/sup&gt;</td>
<td>-</td>
</tr>
<tr>
<td>Programming development environment - IDE&lt;sup&gt;2&lt;/sup&gt;</td>
<td>CCS&lt;sup&gt;3&lt;/sup&gt; 4.2.4</td>
<td>-</td>
</tr>
<tr>
<td>IDE Compiler version</td>
<td>CCS Compiler 4.2</td>
<td>-</td>
</tr>
<tr>
<td>Operating system used during software development</td>
<td>Microsoft 7 32Bit Professional</td>
<td>-</td>
</tr>
<tr>
<td>Maximum software size allocated on embedded chip</td>
<td>16</td>
<td>KB</td>
</tr>
<tr>
<td>Maximum allocated memory on embedded chip</td>
<td>512</td>
<td>Byte</td>
</tr>
<tr>
<td>Max number of used Timers</td>
<td>2</td>
<td>Timer</td>
</tr>
<tr>
<td>Watchdog Timer Support (WDT)</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>Supported communication protocols</td>
<td>UART, SPI, I2C Communication Protocols</td>
<td>-</td>
</tr>
<tr>
<td>Object Oriented Programming Support</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>Logging Feature</td>
<td>YES</td>
<td>-</td>
</tr>
<tr>
<td>Low power modes</td>
<td>Standby Mode and Off Mode</td>
<td>-</td>
</tr>
<tr>
<td>Sensor and Actuator sampling algorithm</td>
<td>Round robin</td>
<td>-</td>
</tr>
<tr>
<td>Wireless communication working modes</td>
<td>Transmit/Receive</td>
<td>Mode</td>
</tr>
<tr>
<td>Wireless Data Packet Size for Information Exchange</td>
<td>5</td>
<td>Byte</td>
</tr>
<tr>
<td>Max Wireless communication information exchanged per second</td>
<td>2048</td>
<td>Byte per second</td>
</tr>
<tr>
<td>Network topology</td>
<td>Star Network Topology</td>
<td>-</td>
</tr>
<tr>
<td>Wireless Information Exchange channel period</td>
<td>50-2000 milli second</td>
<td>-</td>
</tr>
<tr>
<td>ADC bit resolution</td>
<td>8</td>
<td>Bit</td>
</tr>
<tr>
<td>Maximum A/D Conversion Time</td>
<td>20</td>
<td>µs</td>
</tr>
</tbody>
</table>

Table 3.4: Metrics for Software of Hardware Toolkit

- ANT communication channel has maximum information packet size which is 13 bytes. These bytes might be structured that every packet might have a standard format for wireless information exchange. Communication packet should contain sensor information with type of the sensor or actuator information with duty cycle and period information.

- Earl toolkit might have a communication channel that master node of the channel might be able to transmit information with all slaves connected to the same network. Moreover, whenever master node of the network requests information from a slave node, slave node should return the information.

- Earl toolkit microcontroller should monitor ANT module to obtain and send information as much as possible.

- Earl might be able to generate PWM signal with given period and duty cycle in order to control actuators. Signal generated by toolkit should be continuous and accurate for better experiences.
3.3.2 Mobile Application Requirements

Mobile application is one of the fundamental system for Earl sketching toolkit. Mobile platform is a tool that exchanges the information between hardware toolkit and web services. Android application is responsible to forward formatted sensor data to Internet server by using WiFi [37] or cellular data connection. Android application is also responsible to deliver actuator information to hardware toolkit. In the Table 3.5 requirements are given with design objectives of the toolkit.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Design Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earl hardware toolkit has low power wireless communication technology with a mobile phone that is running Android operating system</td>
<td>DO6</td>
</tr>
<tr>
<td>Earl mobile application is a system that technology developers can use it for communication between hardware toolkit and web services without writing code for mobile application</td>
<td>DO6</td>
</tr>
<tr>
<td>By using Earl, mobile application developers can control functionalities of hardware toolkit from the distance</td>
<td>DO6</td>
</tr>
</tbody>
</table>

Table 3.5: Earl Mobile Platform Requirements

According to requirements of android mobile application, metrics are stated as below.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>WiFi Support</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>3G Cellular Network Support</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>Object Oriented Programming Support</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>Communication Modes</td>
<td>Transmit/Receive</td>
<td>Modes</td>
</tr>
<tr>
<td>Maximum Time to Travel by using WiFi on Android mobile phone</td>
<td>500</td>
<td>milli seconds</td>
</tr>
<tr>
<td>Minimum Server Polling Time</td>
<td>125</td>
<td>milli seconds</td>
</tr>
<tr>
<td>Android Operating System Version</td>
<td>4.0.4</td>
<td></td>
</tr>
<tr>
<td>Android Phone ANT Chip Interface Library</td>
<td>ANTLib (Provided by <a href="http://www.thisisant.com">www.thisisant.com</a>)</td>
<td></td>
</tr>
<tr>
<td>Android Phone ANT Chip Interface Library Version</td>
<td>3.2.2</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.6: Mobile Platform Metrics

3.3.3 Web Service Requirements

In this thesis, web server is a tool of Earl that enables technology designers to control actuators and monitor sensor data. These two fundamental functionalities help technology developer to create systems. By using web services, application designers can merge their applications with other web services. By this way, designers can facilitate early design processes by having help from other web services.

In this section, I listed requirements of web services. These requirements are connected to design objectives as previous requirements.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Design Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earl is a toolkit that connects small systems to web services with wireless technology.</td>
<td>DO7</td>
</tr>
<tr>
<td>Technology developers can do complicated process on the Earl web server instead of Earl hardware toolkit.</td>
<td>DO4</td>
</tr>
</tbody>
</table>

Table 3.7: Earl Web Platform Requirements

Metrics of web services are listed below in Table 3.8.
CHAPTER 3. EARL: TOOLKIT FOR SKETCHING

3.4 Technologies for Earl

3.4.1 Wireless Communication Technologies

In section 1.2, need of wireless technology for toolkits is discussed. Earl is designed for near field wireless communication application with long battery life constraint. Most near field communication range for low power wireless technologies vary from couple of centimeters up to order of 10 meters. In this master thesis, I aimed to design a toolkit which is working in an average size of a room with normal conditions [43]. Therefore, Earl is designed to have a wireless communication between devices in 10 meter range.

Power consumption of wireless communication is a constraint for Earl toolkit. In the market, there are low power wireless communication technologies such as Bluetooth, ANT, ZigBee, and Wi-Fi. All these solutions have different functions according to application purposes. In this section, these communication technologies will be discussed with their benefits and drawbacks. Moreover, these communication technologies will be compared with Earl’s operating requirements in terms of power consumption, network topology, communication range, transmission efficiency, and robustness. In the end, I will explain why I chose ANT as wireless communication technology for Earl toolkit.

3.4.1.1 Bluetooth

Bluetooth is a wireless communication technology that intends to replace cables. Wide range of products in many fields, such as mobile devices, headsets, and medical equipments, use Bluetooth to communicate with each other. Bluetooth works on 2400 MHz commercial ISM band.

In Bluetooth network structure, maximum 8 devices can connect to a network at the same time[44]. A master bluetooth enabled device can create a small network connected with 7 slaves to one master. Devices not only can share information in the network in period of time, but also can stream data as well. In Bluetooth network, master and slaves can change the roles. To join a network and start the communication, master and slave should pair with each other. After pairing is successfully done, master and slave can exchange information with each other.

Bluetooth technology has 3 classes to support different range of distances[44]. These classes have limited maximum permitted antenna power. In Table 3.9, classes of Bluetooth are compared with range in terms of permitted antenna power. There are also other reasons for communication range such as environmental effects, antenna settings, material coverage, and battery condition.

<table>
<thead>
<tr>
<th>Bluetooth Class</th>
<th>Maximum Permitted Power (mW)</th>
<th>Maximum Permitted Power (dBm)</th>
<th>Range (meter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1</td>
<td>100</td>
<td>20</td>
<td>~100</td>
</tr>
<tr>
<td>Class 2</td>
<td>2.5</td>
<td>4</td>
<td>~10</td>
</tr>
<tr>
<td>Class 3</td>
<td>1</td>
<td>0</td>
<td>~1</td>
</tr>
</tbody>
</table>

Table 3.9: Bluetooth Classes with Maximum Permitted Power

Bluetooth has different versions. These versions are Version 1.2[45], Version 2.0 + EDR (Enhanced Data Rate)[46], Version 3.0 + HS (High Speed)[47] and Version 4.0 LE (Low Energy)[48]. These versions are improved in terms of security, communication data rate and power consumption.
Bluetooth is a robust communication technology. Any data packet can be acknowledged. Developer can handle the information lost during the transmission by checking acknowledgement information of every transferred packet. Bluetooth is also using channel hopping to find the most efficient communication channel. By using Adaptive Frequency Hopping (AFH) [49], Bluetooth module can discover the most reliable channel frequency out of its operational spectrum and avoid using less efficient channels for future transactions.

Bluetooth communication technology offers several security levels. Devices make handshake before they start to exchange information. Unauthorized devices are not able to exchange information. In order to protect the communication channel, transmitting packets are encrypted because of security reasons.

In terms of power consumption, Bluetooth 4.0 is a low energy solution for wireless communication. Versions before Bluetooth 4.0 have higher power consumption than previous versions[48]. Bluetooth Low Energy (BLE) is designed especially for mobile phones. Today, new mobile phones such as Apple iPhone 5[50] and Samsung Galaxy S3[51] have BLE support. BLE has Star Network Topology(SNT). In SNT, there is one master device in a network and other slave devices connect to the master device in the same network. In BLE technology, 3 slave BLE devices can connect to 1 BLE enabled master device at the same time. This is why, more than 4 device can not communicate in BLE network at the same time. On the other hand, according to Texas Instruments’ CC2540 BLE Stack[52], by changing connection interval and data throughput, BLE master can connect more than 200 slaves in a network. In this case, slave devices can not communicate with master at the same time.

Currently there are BLE modules available in the market. According to Nordic Semiconductors’ nRF8001 BLE specifications[53] power consumption per bit calculations are given below. How power consumption calculation made is explained in Appendix 5.

In BLE technology every packet which is transferred or received has application data and headers. In BLE, payload can be up to 31 byte over 47 byte, which is the full packet size. Therefore, a BLE packet may contain up to %66 application information.

\[
\text{BLE Payload Ratio to Transferred Packet} = \frac{31\text{byte}}{31\text{byte} + 16\text{byte}} = %66 \quad (3.1)
\]

According to the nRF8001’s specifications, operational voltage is taken as 3.0 Volt. This calculation is done when TX peak current at 0dBm output power for 1000 ms connection interval. BLE nRF8001 module has 1 Mbps on air data rate.

\[
\text{PowerConsumption} = 11\text{mA} \times 3V = 0.033W \quad \text{By using equation 5.2} \quad (3.2)
\]

\[
\text{BitsPerSecond} = \frac{1000000 \times 1}{1000000} = \frac{bits}{second} \quad \text{By using equation 5.4} \quad (3.3)
\]

\[
\text{PowerPerBit} = \frac{0.033}{1000000} = 33nW \text{ bit} \quad \text{By using equation 5.5} \quad (3.4)
\]

3.4.1.2 ZigBee

ZigBee is a low power wireless communication technology based on IEEE (Institution of Electrical and Electronics Engineering) [54] 802.15.4-2003 Standard[55]. ZigBee is designed for Mesh Networking Topology (MNT) which enables each node to receive, transmit or relay other nodes’ information in a network. Thus, network area can be extended since nodes can pass information from one to another till packet arrives its destination.

ZigBee is mostly used in sensor networks[56]. Range of wireless technology is proportional to the radio frequency and environmental conditions such as temperature and moisture. Additionally, antenna sensibility and power of transmission are other factors for communication range. ZigBee is working on 2.4 GHz. According to IEEE 802.15.4 technical attributes[57], ZigBee supports transmission distances range from 10 meter to 100 meters, depending on power output and environmental circumstances.

ZigBee technology supports multiple channels and every channel is 2MHz wide and they are separated by 5 MHz. Channel hopping or frequency hopping is a term used for changing operational channel from one to another. ZigBee does not support frequency hopping between channels. This is why, developers who use ZigBee for wireless communication should consider operational frequency in order to decrease
signal interfering. Before ZigBee starts a communication on a channel, network node scans for available spectrum to establish communication. However, if multiple nodes force to scan available channels then there may be interference between nodes. Thus, Mesh network nodes may have problems to start channel. This is why, ZigBee may not have robust communication under this condition.

In ZigBee technology every packet has data unit and headers. In ZigBee packet, data unit can be up to 127 byte whereas headers are 6 byte[55]. Therefore, a ZigBee packet may contain up to %95 application information.

\[
\text{ZigBee Payload Ratio to Transferred Packet} = \frac{127\text{byte}}{6\text{byte} + 127\text{byte}} = %95 \quad (3.5)
\]

As an example Silicon Lab’s EM35x[58] module is ZigBee compliant family. To transfer 250000 bit at 0dBm of data in 1 second required power consumption is shown below. In this calculation, operational supply voltage is set to 2.5 Volt.

\[
\text{PowerConsumption} = 31\text{mA} \times 2.5V = 0.0775W \quad \text{By using equation 5.2} \quad (3.6)
\]

\[
\text{BitsPerSecond} = 250000 \times \frac{1}{1} \times 1 = 250000 \frac{\text{bits}}{\text{second}} \quad \text{By using equation 5.4} \quad (3.7)
\]

\[
\text{PowerPerBit} = \frac{0.0775}{250000} = 310\frac{nW}{\text{bit}} \quad \text{By using equation 5.5} \quad (3.8)
\]

3.4.1.3 WiFi

WiFi is IEEE’s 802.11 Standard[59] for wireless communication. Today, almost every portable computer has WiFi module to access Local Area Networks (LAN) or Internet. WiFi technology has a very high speed communication. Moreover, WiFi is very optimised for large data transfers. Today, IEEE’s 802.11 standard support up to 300 Mbps.

In WiFi technology, point to point communication and star network[59] are the most commonly used network topologies in user applications. In point to point communication, communications connect two network nodes or end points. Two network nodes don’t forward packet to other nodes in the network. In star network topology, there is a central node in network where other nodes connect in the same network which reduces the chance of network failure. Therefore, other network nodes can communicate with each other through central node.

Wi-Fi is one of the most complicated wireless technology that works out among all wireless solutions explained in this chapter. Because of the high communication speed and radio specified standards [59], technology developer should design hardware with many constraints such as timing and power constraints. This is why, WiFi modules are mostly used when high data rate is a need in the application. In addition to this, power consumption of WiFi module and communication range should be considered according to application needs.

According to IEEE’s 802.11b and 802.11g standards[59], antenna structure, and environment conditions. Most commercial WiFi modules have range up to 30 meter in door and 100 meter in out door applications. This range is doubled on 802.11n standard[60]. On the other side, because of high communication range, WiFi has more power consumption than other wireless technologies.

As an example Roving Networks’ RN-XV WiFly Module[61] is a 802.15.4 wireless module enables devices to connect to WiFi networks. According to module specifications, to transfer 54000000 bit of data at 10dBm in 1 second required power consumption is shown below. In this calculation, operational power supply voltage is set to 3.3 Volt.

\[
\text{PowerConsumption} = 180\text{mA} \times 3.3V = 0.594W \quad \text{By using equation 5.2} \quad (3.9)
\]

\[
\text{BitsPerSecond} = 54000000 \times \frac{1}{1} \times 1 = 54000000 \frac{\text{bits}}{\text{second}} \quad \text{By using equation 5.4} \quad (3.10)
\]

\[
\text{PowerPerBit} = \frac{0.594}{54000000} = 11\frac{nW}{\text{bit}} \quad \text{By using equation 5.5} \quad (3.11)
\]
3.4.1.4 ANT

ANT is a wireless communication solution which is one of the lightest protocol available operating on the 2.4Ghz which is a commercial ISM band. ANT has good performance with its ultra low power and ultra-high efficiency features on application fields where developers are having sensor connectivity problems [62]. Thus, developers prefer ANT technology for developing applications related to digital sensing projects [62]. One of the biggest opportunity of using ANT is enabling developers to implement complex network topologies and communication methods rapidly. ANT technology is standing out with its competitive features against Bluetooth technology such as compact design, scalability and flexibility in wireless network [63].

ANT has supportive built-in features such as device search, pairing, timing, power management and interference handling. Therefore, application designers can establish and handle communication rapidly. Application developers can create different network topologies by using ANT such as simple to complex networks, peer-to-peer, star, tree and mesh [64]. Moreover, ANT has public and private network support for developers to customize the wireless network for more secure applications. Especially for sensor networks which are monitoring and controlling peripherals, ANT technology is supporting multiple network topologies by its enhanced ANT protocol stack to establish flexible networks.

In ANT network topology, every ANT enabled device can receive and transmit information. In ANT network topology, there are 3 types of data transmission modes: broadcast mode, acknowledged mode, and burst transmission mode. Sensor applications mostly use one way data transmission from sensor node to data collection node. This transmission is called broadcast mode in ANT systems. Transmitter nodes can also get acknowledgement if transmitting data is significant to deliver safely. In case packet does not deliver safely to target node, transmitter gets acknowledgement about packet transmission. In case of unsuccessful transmissions, ANT does not try to resend packets. Application developer should care about retransmission of packets in case of unsuccessful transmission. For high data rate applications, ANT can transfer information up to 20 kbps in burst mode. Burst mode is a multi-message transmission mode that is using the whole allocated bandwidth. However, transferring information in burst mode may change the network topology. In case of unsuccessful transmission, ANT cannot resend failed packets.

According to Nordic Semiconductors specifications[11], under normal conditions[43] ANT wireless technology can exchange information up to 30 meters. On the other hand, some factors such as environment conditions, antenna structure and interference with other wireless technologies which are working in the same band can affect to change communication distance.

In ANT networks, application developers can connect many ANT enabled devices to the same network with different network topologies. According to Nordic Semiconductors’ specifications[11], 65535 devices can connect to same ANT wireless network. In addition to this, some ANT chips support up to 8 channels. Application designer can set every channel to different networks. Therefore, developer can create complex network topologies by using ANT wireless technology.

ANT protocol has two type of packet type which are standard data packet and burst mode data packet. Standard data packet has 8 byte usable data from application. There are 5 more bytes for packet headers and ANT protocol related bytes. This means that every ANT packet contains 61.5 application information.

\[
\text{ANT Payload Ratio to Transferred Packet} = \frac{8\text{byte}}{8\text{byte} + 5\text{byte}} = \%61.5
\]

For low rate data transmission applications ANT is an efficient solution in terms of low power consumption. According to ANT product specification [65], to transfer 20000 bit of data at 0dBm in 1 second required power consumption is shown below. In this calculation, operational supply voltage is set to 2 Volts.

\[
\text{PowerConsumption} = 5.9\text{mA} \times 2.0V = 0.0118W \quad \text{By using equation 5.2}
\]

\[
\text{BitsPerSecond} = 20000 \times \frac{1}{1} \times 1 = 20000 \text{bits second} \quad \text{By using equation 5.4}
\]

\[
\text{PowerPerBit} = \frac{0.0118}{20000} = 590 \text{nW bit} \quad \text{By using equation 5.5}
\]
According to ANT’s official Power State document[66], an ANT AP2 module consume current as mentioned in Table 3.10.

<table>
<thead>
<tr>
<th>Power State</th>
<th>Current Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>~3mA</td>
</tr>
<tr>
<td>Suspend</td>
<td>2uA</td>
</tr>
<tr>
<td>Sleep</td>
<td>3uA</td>
</tr>
<tr>
<td>Idle</td>
<td>2uA</td>
</tr>
<tr>
<td>Deep Sleep</td>
<td>0.5uA</td>
</tr>
</tbody>
</table>

Table 3.10: ANT AP2 Power States @3.3 Supply Volt when 32KHz External Oscillator Used

According to ANT’s official Power State document, ANT module only consumes 5.9 mA when it is transmitting messages. If ANT module is not sending or receiving messages, it changes its power state from Active power state to suspend, sleep, or idle power states. In order to save power, ANT module can be changed to deep sleep mode. However, deep sleep mode kills channel synchronization. Therefore, ANT enabled device should reopen a channel to communicate with other ANT enabled devices.

ANT wireless technology is getting popular because of its used wireless network topologies, number of connected nodes in a network, and low power consumption. ANT has been used in many applications such as wellness, home health monitoring, home and industrial automation, environmental sensor networks, active RFID, logistics and goods tracking, audience response systems and many more [63]. Sony Mobile Phones[67] are the first mobile phones which are used embedded ANT modules, hence technology developers can exchange information with ANT enabled devices by using mobile phones.

Application developers can use ANT enabled mobile phones as a bridge to Internet. Today, mobile phones have high speed Internet connection. In this way, developers can connect applications to Internet and exchange information between ANT network and Internet. In IoT vision, every device is connected to Internet to exchange information. By using an ANT enabled mobile phone, developers can create systems in IoT vision. Any information coming from Internet and from ANT enabled devices can distribute through ANT enabled Internet connected mobile phone.

In this thesis, I used ANT wireless technology for Earl to exchange information between other Earl toolkits and ANT enabled devices such as ANT enabled mobile phones. ANT wireless technology enables technology developer to work with up to 65535 Earl toolkits which are connected to same network. Furthermore, ANT is supporting multiple network topologies such as peer-to-peer, star, tree and mesh networks. Earl hardware toolkit is designed for star network topology. Therefore, multiple slave devices can communicate through one master device. Earl is designed to have a wireless communication between devices in 10 meter range. According to Nordic Semiconductors specifications[65], under normal conditions [43] ANT wireless technology can exchange information up to 30 meter. As it was mentioned previously, ANT has low data transmission rate compared to its competitors such as Bluetooth, ZigBee and WiFi. However, in terms of power consumption, according to Calculation 3.2, 3.6, 3.9, and 3.13 ANT is a more efficient solution than other available wireless communication technologies.

Lastly, in Earl, I assumed designers don’t need high transmission rate such as WiFi has. Earl is designed for rapid implementation to create small systems. On the other hand, if desired application requires higher communication speed, developers can use more advanced technologies in terms of transmission rate for prototyping and product design phase. This is why, I used ANT wireless technology for Earl Toolkit as communication module between other Earl devices and ANT enabled devices.

### 3.4.2 Low Power Technologies

Most early design toolkits are designed for stationary experiments. However, significance of power consumption arises when the technology developer wants to use sensing and actuating environment while toolkit is moving. One crucial reason to decrease the power consumption in hardware can be mobility [68]. There are other reasons such as overheating and high design costs [68]. In this thesis, I will focus on low power consumption for longer mobile experiences. This section explains how designers can increase battery life by keeping power consumption low. After explaining how designers can achieve longer mobile experiences, I will talk about low power consuming components that I used in Earl toolkit.
Mobile devices have limitations in terms of power consumption since the battery energy is limited. In some experiments developers desire long battery life. However, interrupted experiments due to short battery life may cause wrong results. For example, battery time can be very critical such as an application that measures arousal level once every second through bio-sensors attached to human body during a month. If battery lasts once every two days, in the end of the month experiment results can be false. Developers can change or charge batteries of the device or increase battery capacity to have longer experiences. However, changing or charging batteries may cause pauses in the experiment. Increasing battery capacity may cause having bigger sized batteries in mobile devices. Big physical sized mobile devices can be difficult to attach to human body and carry the device. Therefore, developers can use smaller capacity batteries with low power devices for the same battery life as higher power consuming devices with bigger capacity batteries. This is why, low power consuming mobile devices can be useful in portable applications.

Application developers can calculate approximate operation time of a device by measuring current consumption. To calculate the current consumption of the device, a multimeter can be connected in between positive terminal of battery and positive terminal of device’s power connector. Then, the multimeter should be set in DC current function mode to measure the current passing through multimeter. The value on screen is the current consumption of the device according to functional state of the device. Therefore, designers can estimate approximate operational time of the device by using this method.

Approximate battery life of the device can be calculate as following [69]:

\[
\text{Approximate Operational Time of the Device} = \frac{\text{Battery Capacity in Amper Hours}}{\text{Device's DC Current Consumption in Amper}} \quad (3.16)
\]

According to the Equation 3.16, technology developers should consider battery capacity and device’s DC current consumption of its functional states in order to decide how long the battery will last. According to the equation, adding a new peripheral to the system that increases DC current consumption will increase the energy needed. That is why, application designers should prefer to use low power consuming components in order to have longer battery life. Another way to increase battery life can be removing unused resources from the system. For example, unused but connected sensors and actuators will draw extra current from battery. This is why, designers should remove extra peripherals from the device if they are not using them.

Software loops constitute the most used structures in a software program [70]. Redundant and inefficient loops may lead to unnecessary power consumption. Designing a low power system is not only a matter of using low power components but also a matter of designing hardware compatible software structures. Optimized software for microcontrollers would have longer battery life as well.

Application designers can get longer battery life by using system resources more efficiently. Some modules and microcontrollers have special hardware structures for some purposes such as hardware multipliers for more low power multiplications. Application developers should consider using custom structures placed on chips in order to implement the processes by software [68]. By this way, they may get lower power consumption and much faster solutions than software implementations.

One another way to decrease the power consumption can be using low power modes of modules on the toolkit. Some modules have low power modes such as ultra low power mode (sleep) and power down modes. Designers can switch these modules into low power modes and activate these modules when they need to be used. By using low power modes designers can get much longer battery life.

Low supply voltage also decreases power consumption. According to Equation 5.2, decreasing supply voltage affects power consumption with the second order of voltage. Furthermore, increasing operational frequency also effects power consumption. Formula 5.6 shows that there is a relationship between the change in frequency and power consumption[71]. This is why, low operational frequency effect power consumption of microcontrollers. On the other hand, there is a trade of between speed and power consumption. Low operational frequency slows down tasks which is handled by microcontrollers. This is why, designers should select optimised operational frequency and supply voltage for devices to achieve maximum battery life.
3.4.3 Low Power Units of Earl

In Earl toolkit, low power consuming modules are used to decrease the power consumption. These modules are microcontroller and wireless communication unit. As I explained the reasons in Section 3.4.1.4, I used Nordic Semiconductors nRF24AP2 Single Channel ANT module [72] for the low power wireless communication unit. For the control unit of the toolkit, I choose Texas Instrument’s MSP430G2553 [73] microcontroller. The reasons I chose MSP430G2553 microcontroller are low development costs and power consumption of the microcontroller. TI has a low power microcontroller development board called TI Launch Pad costs less than $5 (According to price at March 9, 2013). This microcontroller is compatible with TI LaunchPad and it has necessary resources such as enough memory and essential on board communication protocols. According to hardware toolkit system requirements given in Section 3.3.1.2, MSP430G2553 microcontroller fulfills system requirements in terms of low power consumption, memory size, on board communication protocols, and number of ADC modules. Therefore, I decided to use MSP430G2553 for Earl toolkit.

MSP430G2553 has supply voltage range between 1.8 Volt and 3.6 Volt [73]. It has internal oscillator working at 1 MHz and has support for external oscillator up to 16 MHz. According to its data sheet [73], under normal conditions [43] MSP430G2553 consumes 230 $\mu$A working at 1 MHz with 2.2 Volt voltage supply. MSP430G2553 has also low power modes that consumes 0.5 $\mu$A in standby mode and 0.1 $\mu$A in off mode under normal conditions [43] and works at 1 MHz frequency with 2.2 Volt voltage supply.

MSP430G2553 has 512 byte Ram and 16 KB Flash memory. In the class of MSP430G2 series of microcontrollers, MSP430G2553 has the maximum memory size. Furthermore, MSP430G2553 has 256 Byte program flash memory. Program flash memory is readable and writable memory located on the microcontroller. If microcontroller switched off or restarted, program flash memory can be accessible by microcontroller. In Earl toolkit, I used program flash memory to save embedded software constants and keep system statistics. For example, in program flash memory I keep software version information and hardware information. Moreover, I keep statistics such as system startup counter that counts number of turn on and restart.

MSP430G2553 has on chip ADC (Analog to Digital Converter) modules. By using these modules, Earl enables developers to convert analog sensor data to digital data. In Earl toolkit, I used MSP430G2553 QFN (Quad-Flat No-leads) package. Therefore, there are 8 ADC channels with 10-bit resolution available on this package. In an Earl application, developers can use 2 of these ADC channels to convert analog information into digital information. ADC module which is located on the microcontroller, can convert information with the bandwidth of 200-kmps (Kilo Sample per Second) from one single channel at a time. In Earl toolkit, developers can get samples in 8 Hz within 8 byte information.

MSP430G2553 has the most common communication protocols such as UART (Universal Asynchronous Receiver/Transmitter), Synchronous SPI (Serial Peripheral Interface Bus) and I$^2$C (read as “eye-squared cee”). In Earl toolkit, I used UART to communicate with ANT module. In implementation chapter, the design of UART channel will be explained in details.

MSP430G2553 has also 2 16-bit Timer for time dependent applications. Time based events on Earl, such as analog to digital data conversion, use timer to sample on the information in periods.
Chapter 4

Implementation

This chapter explains how Earl toolkit is implemented. As I mentioned in Section 3.1, I divide whole system into 3 main parts. These parts are Earl toolkit, Android mobile application, and Earl web services.

In this chapter, firstly, I will explain Earl hardware toolkit implementation. I will introduce the components and how toolkit hardware is designed. Next, I will explain the software structure that is running on Earl toolkit. Furthermore, I will explain the ANT messaging protocol and packet structure of wireless transmission between toolkit and Android mobile phone. Secondly, I will introduce Android mobile application and its functions. Next, I will explain how Earl web server works. In this section, I will also talk about how exchanged information between hardware toolkit and web services are stored in the Earl web services database. In the end of this chapter, I will talk about power consumption and data exchange performance of Earl hardware toolkit.

4.1 Earl Toolkit

4.1.1 Hardware Design

Fundamental components of Earl hardware toolkit can be listed as MSP430G2553 microcontroller, nRF24AP2 ANT Module and Antenna circuit, TI TPS77333 voltage regulator, TI DRV8601 Haptic driver, Input/Output pins (IO), debug and status LEDs (Light-Emitting Diode), and battery connector and external battery. According to given specifications in Table 3.2 and Table 3.3, every module is designed separately with fundamental functionalities and merged on main design. In the following sections, I will explain how sub components are designed and placed on Earl hardware toolkit PCB (Printed Circuit Board).

4.1.1.1 PCB Design

Earl PCB has been designed by using Altium Software [74]. Earl toolkit board is designed with given specifications in Table 3.2. In order to decrease the board dimensions, while designing the board, minimum length of wires on the board set to 0.30 inch. In some places on the board, maximum wires length set to 0.40 inch. Furthermore, 0402 sized components are used. In Figure 4.1, the picture of Earl board is given.
In Earl toolkit board design, I used sample schematics given on datasheets. Detailed schematics of components will be explained in following sections. According to ICs’ (nRF24AP2, MSP430G2553, TPS77333, and DRV8601) sample schematics given on their datasheets, low error tolerance (error on value of the component) resistors, capacitors and inductors are selected to increase the system performance and fine tuned circuits. List of the components used for Earl hardware toolkit will be given in following Section 4.1.1.2.

4.1.1.2 Modules and Components Used in Hardware Toolkit

In this chapter, sub modules of Earl toolkit are explained with schematics. Following chapters will explain how modules are designed. To demonstrate modules with components I will use schematics. In these schematics, components such as resistor (R), capacitor (C), inductor (I), LED (D), Cristals (X) are represented by their abbreviations and identification numbers (ID). To obtain component’s values, reader can check the Table 4.1. For example, on TPS77333’s schematic, there is a capacitor named C21. This capacitor’s value is given on Table 4.1 as 10 $\mu$F.

Furthermore, I also gave part numbers of the components from Farnell [75] which is an electronics parts distributor. However, there are some components such as nRF24AP2 can only be found from special distributors with a request.

4.1.1.2.1 Battery

Earl toolkit is designed as a mobile toolkit. This is why, it is working with a battery. According to the specifications given in Table 3.2, battery is selected as Polymer Lithium Ion Battery with 1 cell and battery supply voltage is chosen as 3.7 Volt. Under these specifications, batteries with different capacities can be used according to application purposes.

It is significant to mention battery size, therefore total hardware toolkit size, is directly related to battery capacity. In my applications and for test purposes, I used single cell polymer lithium ion battery with 850mAh capacity. This battery can be found at SparkFun [http://www.sparkfun.com]2012 with part number PRT-00341. I used this battery because I had this battery before and it fits to my specifications.

4.1.1.2.2 TPS77333 Voltage Regulator Module

TPS77333 is a DC linear voltage regulator that provide a constant voltage from battery to toolkit board. While toolkit is powered on, there may be small changes on supply voltage. Voltage regulator plays a role to fix the supplied voltage ($V_{dd}$) to components assembled on the board. It has a capability to supply 3.3 volt with maximum 250 mA current output to toolkit board. Figure 4.2 shows how TPS77333 module is designed.

TPS77333 datasheet suggests a sample schematic for 3.3 supply voltage applications. In Earl toolkit, this sample schematic [76] is used for implementation of TPS77333. TPS77333 has a feature to reset when the device is power-on. Therefore, all components, which are connected to reset pin of TPS77333, can be reset to their initial state. In Figure 4.2, pin RST_PROG controls microcontroller to reset and set microcontroller to its initial state. This pin is also used to hold the system in reset state. RST_PROG pin
CHAPTER 4. IMPLEMENTATION

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Distributor:PartID</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSP430G2553</td>
<td>-</td>
<td>Farnell:1903590</td>
<td>16BIT, 16K FLASH, Package: QFN</td>
</tr>
<tr>
<td>nRF24AP2</td>
<td>-</td>
<td>NordicSemi: nRF24AP2-1CH</td>
<td>AP2 1CH Package: QFN</td>
</tr>
<tr>
<td>TPS7733</td>
<td>-</td>
<td>Farnell:2118304</td>
<td>3.3V, 250mA, Package: MSOP-8, 400 mA</td>
</tr>
<tr>
<td>DRV8601</td>
<td>-</td>
<td>Texas Instruments:DRV8601</td>
<td></td>
</tr>
<tr>
<td>Cristal X1</td>
<td>16MHZ</td>
<td>Farnell:1712816</td>
<td>50PPM(Parts per million), 12.5PF</td>
</tr>
<tr>
<td>Cristal X2</td>
<td>32.768KHZ</td>
<td>Farnell:1278038</td>
<td></td>
</tr>
<tr>
<td>C1,C2,C17,C18</td>
<td>15 pF</td>
<td>Farnell:1327631RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C23</td>
<td>1 nF</td>
<td>Farnell:2280785</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C21,20</td>
<td>10 μF</td>
<td>Farnell:1886093</td>
<td>Package: 0603</td>
</tr>
<tr>
<td>C22,C11,C9,C7</td>
<td>0.1 μF</td>
<td>Farnell:1327657RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C24</td>
<td>10 nF</td>
<td>Farnell:1327624RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C5</td>
<td>1.5 pF</td>
<td>Farnell:1327633</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C6</td>
<td>1 pF</td>
<td>Farnell:1800794</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C10,C8</td>
<td>33 nF</td>
<td>Farnell:1865445RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C3</td>
<td>2.2 μF</td>
<td>Farnell:1740585RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>C4</td>
<td>Not Placed</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>R1</td>
<td>22 Kohm</td>
<td>Farnell:1358077RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>R2,R3</td>
<td>4.7 Kohm</td>
<td>Farnell:1358060</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>R4,R5</td>
<td>47 Kohm</td>
<td>Farnell:1803053RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>R6,R7,R8</td>
<td>47 ohm</td>
<td>Farnell:2078852</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>D1,D2,D3</td>
<td>LED Red</td>
<td>Farnell:2001787</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>L1,L2</td>
<td>6.8 nH</td>
<td>Farnell:1669642RL</td>
<td>Package: 0402</td>
</tr>
<tr>
<td>L3</td>
<td>4.7 nH</td>
<td>Farnell:HL024R7BTTR</td>
<td>Package: 0402</td>
</tr>
</tbody>
</table>

Table 4.1: Parts used in Earl hardware toolkit

![Figure 4.2: TPS7733 I/O Pins](image)

Note: Unnamed I/O pins are not connected. This figure is generated by using Altium Software.

becomes active and resets when it is set to logic low and until RST_PROG pin’s value set to logic high.\(^1\) Moreover, when too much current is flowing over voltage regulator (according to TPS7733’s datasheet it is given as 237.5 mA, where maximum current flow over this component is 250 mA), TPS7733 set RST_PROG pin to logic low. Thus, microcontroller and other ICs connected to this pin will be hold in reset state.

There are 2 main active DC to DC voltage regulator types. These are linear regulators and switching regulators [77]. Differences between linear DC voltage regulators and switching DC voltage regulators are explained in Texas Instruments’ article with Literature Number SNVA558. According to this article, linear voltage regulators consumes more power than switching voltage regulators. Especially, linear voltage

---

\(^{1}\)In the rest of the thesis, logic high means 3.3 Volt which is also called \(V_{dd}\) value. Opposite to this, logic low means 0 Volt which is also called \(V_{ss}\) or Ground (GND).
CHAPTER 4. IMPLEMENTATION

Regulators dissipate energy to supply constant output voltage. Emerging energy of voltage difference between battery output voltage (which is also voltage regulator input voltage) and voltage regulator output voltage is dissipated on voltage conversion in linear DC voltage regulator. On the other hand, switching DC voltage regulators can only dissipate energy when there is a voltage drop or jump occurred in the input of voltage regulator. This is why, linear DC voltage regulators have lower efficiency with respect to switching DC voltage regulators in low power applications.

In Earl toolkit, battery output is chosen as 3.7 Volt and linear DC voltage regulator output \( V_{cc} \) is selected 3.3 Volt. Energy emerged because of the voltage difference in input and output voltage of voltage regulator \( (0.4 \text{ Volt}^2) \) is dissipated by TPS77333. Therefore, according to current flowed from battery to voltage regulator and voltage difference between voltage regulator and battery changes the power consumption of the device. However, in the ideal case where battery voltage is 3.3 Volt and voltage regulator output voltage is 3.3 Volt as well, power consumption of TPS77333 approaches to 0.

4.1.1.2.3 MSP430G2553 Microcontroller Module

MSP430G2553 microcontroller is heart of Earl hardware toolkit. All peripherals connected to hardware toolkit are controlled by this microcontroller. Furthermore, microcontroller also provide needed commands to ANT module to transmit information to Android mobile phone. Figure 4.3 demonstrates schematic of the MSP430G2553.

![Figure 4.3: MSP430G2553 I/O Pins](image)

Note: Unnamed I/O pins are not connected. This figure is generated by using Altium Software.

Power pins of MSP430G2553 \( V_{cc} \) and \( V_{ss} \) (GND) are connected to voltage regulator in order to get a constant supply voltage. In Earl toolkit, \( V_{cc} \) is set to 3.3 Volt. Moreover, capacitors C19 and C20 (known as Coupling Capacitors) are designed to provide constant supply to microcontroller. These capacitors are used to remove the AC signal on the supply voltage. Especially, in voltage critical applications, such as ADC applications, change in power supply voltage of microcontroller may effect the microcontroller.

---

20.4 Volt difference is the case of battery output is 3.7 Volt. In some cases it can be more or less than 3.7 Volt according to charge in the battery.

3Calculation of power consumption is explained in Section 5
processes. In ADC applications, the result of ADC conversion may be wrong if there is a change in microcontroller power supply voltages. This is why, coupling capacitors (C19 and C20) are connected next to power supply pin of microcontroller.

According to Formula 5.6, clock frequency of a component effects the power consumption directly. In Earl application, clock speed of microcontroller is set to 1 MHz and an external oscillator is not used. MSP430G2553 microcontroller has a UART communication port. UART channel is configured for communication with ANT module. On Figure 4.3, pins assigned to UART communication labeled with “UART_TX” and “UART_RX”. UART_TX is connected to ANT module’s communication transmit pin. This pin on microcontroller obtain information from ANT module. UART_RX is connected to ANT module’s communication receive pin and by using this pin, microcontroller can send information to ANT module. “SLEEP” and “SUSPEND” pins of microcontroller controls ANT module’s SLEEP and SUSPEND pins to control ANT’s power states. For ANT’s power state please refer to ANT Power State documentation [66].

MSP430G2553 has one 8 channel 10 bit embedded ADC feature. In Earl toolkit, two sensor data are converted by using internal ADC of microcontroller. These ADC input pins are labeled as ADC1(ADC Channel 3) and ADC2(ADC Channel 4) on Figure 4.3.

In order to control haptic driver, PWM (Pulse-width Modulation) signal is generated from pin labeled as “PWM” on Figure 4.3. PWM is a periodic signal that is formed by logic low and logic high values. In one cycle of the period, the ratio of time for logic high signal over cycle time gives duty cycle of PWM signal. Frequency of repetitive cycles in one second give period of PWM signal. According to period and duty cycle changes, it is possible to control haptic driver. EN_OUTPUT pin shown on Figure 4.3 enable and disable haptic driver IC. In order to decrease the power consumption, enable pin should be set to logic low.

Pins RST_PROG and pin TEST_PROG are assigned to program microcontroller. There are male pin headers connected to these two pins to program microcontroller easily. Ground pin is also included to programming port of hardware toolkit to set ground voltage reference and protect the hardware circuit.

There are 2 LEDs located on the board to debug and test the application. DOUTPUT1 and DOUTPUT2 are allocated for LEDs. DOUTPUT1 is assigned for successfully transmit/receive data by ANT module. Whenever microcontroller transmits or receives a data this LED blinks. Other LED is for debugging purposes. During programming the board, this LED is used for tracing the code.

4.1.1.2.4 ANT nRF24AP2 Module

ANT Technology was introduced in Section 3.4.1.4. Earl toolkit is using ANT as a communication technology in order to exchange information with Android mobile phone. In this section, I will explain how ANT module is designed.

In Earl hardware toolkit, Nordic Semiconductor ANT AP2 single channel module is used. This module has 32 pin QFN package as shown in figure 4.4. It has 3.3 power supply on V_{dd} pins. ANT circuit which is implemented on Earl toolkit is referenced from its datasheet [65].

In Earl hardware toolkit, ANT circuit is designed based on suggested schematic on ANT datasheet. Some configurations such as communication protocol between microcontroller and communication speed is configured according to given specifications. According to the specifications, ANT module is set to UART communication protocol with 57600 kbps. To set the communication speed (baud rate) between microcontroller and ANT module to 57600 kbps, pin 15 (BR1), pin 18(BR2), and pin 14(BR3) are set logic high. Pin 5 (PORTSEL) is set to logic low since ANT module is designed for Asynchronous Serial Interface (UART). Pin 16 (Sleep pin) is set to logic low in order to decrease the design complexity. Therefore, ANT module can listen mobile phone anytime. UART RX and TX pins are connected to microcontroller’s UART Communication channel.

ANT module is designed as it is suggested in Nordic Semiconductor’s ANT datasheet[72]. In order to decrease the board size, in hardware toolkit Fractus FR05-S1-N-0-102 model omnidirectional chip antenna is used. This antenna is 50Ω impedance linear antenna with the dimensions 7mm × 2mm × 3mm sizes. It has frequency range between 2.4-2.5 GHz, therefore this antenna can be used in Bluetooth, WLAN 802.11 b/g and other applications which are supposed to work at 2.4 commercial ISM band. ANT is also specified to operate at 2.4 GHz. This is why, I used this antenna to decrease the space that antenna allocated on the PCB board. FR05-S1-N-0-102 has 1.7 dB Peak Gain between input and output of the antenna. Average gain on antenna is given as 1.32 dB between input and output according to FR05-S1-N-0-102 datasheet. In the antenna operational frequencies (frequencies between 2.4 GHz and
CHAPTER 4. IMPLEMENTATION

Figure 4.4: ANT nRF24AP2 I/O Pins
Note: Unnamed I/O pins are not connected. This figure is generated by using Altium Software.

2.5 GHz), FR05-S1-N-0-102 has efficiency between %69 - %75 across the band. Rest of efficiency can be assumed as loss in power at antenna. According to FR05-S1-N-0-102’s datasheet, average efficiency is given as %72 in the operational band.

4.1.1.2.5 DRV8601 Haptic Driver Module  Haptic Driver DRV8601 is a hardware component to drive actuators such as DC Motors and Linear Vibrators. DRV8601 provides differential output to connected actuators. By providing differential outputs, haptic driver allows the polarity of the voltage across the output to be reversed quickly. This functionality enable haptic driver to rotate connected motor both in clock wise and counter clock wise. Moreover, differential output also enable quick motor stop.

Haptic driver schematic is implemented according to example taken from DRV8601 datasheet [78]. Figure 4.5 shows how DRV8601 module is designed.

Figure 4.5: DRV8601 Haptic Driver I/O Pins
Note: Unnamed I/O pins are not connected. This figure is generated by using Altium Software.
Supply voltage of haptic driver is connected to voltage regulator’s output instead of battery output. Therefore, haptic driver does not effect instant voltage changes on battery. DRV8601 has a capacity to drive up to 400 mA from a 3.3 Volt supply. However, voltage regulator can provide 250 mA to all system including haptic driver.

Haptic driver is working with PWM signal which is generated on Pin 19 located on MSP430G2553 microcontroller (4.3 shows microcontroller input and output pins). Section 3.3.1.2 explains how PWM functionality is implemented.

4.1.1.2.6 Connectors and LEDs In this section, connectors and LEDs will be explained with their functionalities. Figure 4.6 shows used connectors and LEDs on the board.

![Figure 4.6: Connectors and LEDs](image)

Note: Unnamed I/O pins are not connected. This figure is generated by using Altium Software.

P1 and P2 are connectors that technology developers can connect analog sensors. Sensor inputs are designed for Phidgets [79] sensor connectors. Therefore, both of the inputs have 3 pins. 3.3 \( V_{cc} \) and Ground pins are already supported to the sensor to provide power to the sensor. 3rd pin is used to read analog sensor information from sensor. P3 is the battery connector. A lithium ion battery can be connected to here to provide the energy. P4 is the connector to program the microcontroller. By using these pins debugging the microcontroller is also possible. In order to debug or program the microcontroller, these pins can be connected to MSP430 LaunchPad development board’s RST and TEST pins. D1 is the LED to shows that the hardware toolkit is switched on. D2 is the LED that shows hardware toolkit is sending or receiving information through ANT module. D3 LED is used for debug purposes during the development phase.
CHAPTER 4. IMPLEMENTATION

4.1.2 Embedded Software Design

In this section, firstly I will give information about the software development environment that is used during implementation. Secondly, I will talk about embedded software architecture that is designed for hardware toolkit. Earl embedded software is designed in different modules. Each module contains sub-functionality of embedded software. Each file will be explained with its functionalities. Next, I will explain how wireless communication packets are designed. Finally, I will give information about how Earl toolkit is communicating with mobile phone.

In Embedded software design Code Composer Studio (CCS) 4.2.4 version is used. CCS IDE installed on a Microsoft Windows 7 32-Bit Professional operating system and CCS IDE is using CCS Compiler 4.2 for code compilation. Embedded software has been written in ANSI C language. During the embedded software design and debug process, TI’s MSP430 Launch Pad [80] is used to program the hardware toolkit.

4.1.2.1 Embedded Software Architecture

Embedded software contains few modules. Every module contains sub functionality and constant variables of hardware toolkit. These modules are explained in following sub sections.

- Constants Module
- Base Module
- Flash Module
- UART Module
- ADC Module
- PWM Module
- Scheduler Module
- ANT Software Module
- Test Module
- Main Module

4.1.2.1.1 Constants Module

In Constants module, constant definitions and constant variables are defined. These definitions enable embedded software code easily readable and easily configurable code. Moreover, some constant values such as baud rate and buffer sizes, program constants and register names are declared in this file.

Earl operational modes can be changed by using defined parameters in Constants module. There are 4 operational modes of Earl hardware toolkit. Normal mode is the mode that Earl supposed to do its defined functions. Debug mode enables the developer to test the embedded software. In some parts of the embedded software code, this mode prints information and messages to computer screen by using UART communication channel. More information will be explained in “UART Module” subsection. Thirdly, test mode enables embedded software developer to test hardware functionalities such as PWM, ADC and UART modules. Lastly, register mode resets all the values in flash memory of MSP430G2553 microcontroller and sets to its default values. Information about flash memory can be found in “Base Module” and “Flash Module” subsections.

Constant module contains MSP430G2553 register address information which is provided by CCS IDE, and used ANSI C “stdarg.h” library. MSP430G2553 header file enable application developer to access special registers in microcontroller by predefined register names instead of register addresses. By using stdarg ANSI library, functions can accept an indefinite number of arguments. This library is specifically used to implement “printf function” in UART module design. More detailed information can be found in sub section UART module.
CHAPTER 4. IMPLEMENTATION

4.1.2.1.2 Base Module  First initialized module of Earl hardware toolkit is Flash module. Flash memory is non-volatile storage that is readable, writable, and erasable. Flash module enables embedded software to access flash memory of MSP430G2553. In MSP430G2553’s flash memory, Earl embedded software stores system information. By the help of predefined constant address in Constant module, Earl can access to Flash memory. In Earl’s flash memory, software and hardware version number, runtime counter, device ID and device class ID\(^4\) are stored. Runtime counter is a counter that increases its value whenever system starts. This information helps to keep statistics about how many times system restarted.

Second initialized module is UART module. By using this module it is possible to print information stored in UART to computer screen in debug mode. After UART module, other modules such as ADC and Test modules initiates to prepare the system to start as it is supposed.

Base module also make Input Output (IO) pins ready according to Earl hardware toolkit configurations. For example, base module sets Debug and TX/RX microcontroller outputs as output pins.

4.1.2.1.3 Flash Module  In this module, functions to write, erase, and read flash memory are implemented. In MSP430G2553, flash memory can be read, wrote, and erased with blocks of flash memories. Every block has size of 64 byte. This is why, a buffer with the size of 64 byte is implemented. Embedded software uses the flash memory via buffer. In MSP430G2553, there is 256 byte flash memory available to store information. This means, 4 blocks of flash memory can be used to store application information. In Earl embedded software, only first block of flash memory is used to store application information.

4.1.2.1.4 UART Module  UART (Universal Asynchronous Receiver/Transmitter) is a communication protocol for serial information exchange between integrated circuits. UART communication protocol contains two channels, which called Transmit and Receive channels. In one UART channel, two devices can connect and talk. Two connected devices can send and receive information at the same time. This communication is called full duplex. UART communication may be one way, or send or receive at one time. In Earl toolkit, full duplex communication is used in UART communication.

In UART communication, information may be sent in 8 bit (1 Byte) packets. Every packet starts with a Start bit and ends with one or two Stop bits. In Earl UART configuration, every packet starts with a Start bit and ends with one Stop bit. According to communication options, UART communication packets may contain check sums to detect accidental errors in communication. In Earl toolkit, UART communication packets contain 8 bit information and does not include check sum bit. Communication between microcontroller and ANT module is set to 57600 bps. Moreover, in ANT UART communication, least significant bit of the transmitting byte is set to send first. Likewise, least significant bit of the received byte is the first bit received from microcontroller. More information about UART channel between microcontroller and ANT module is discussed in Section 4.1.1.2.4.

In MSP430G2553, there are two UART modules that is possible to use in Earl hardware toolkit. These modules are called Universal Serial Communication Interface (USCI) and Universal Serial Interface (USI). In Earl hardware toolkit, USCI UART module is used to provide communication between microcontroller and ANT module. More information about USI and USCI can be obtained from MSP430x2xx User’s Guide [81].

In Earl hardware toolkit, USCI UART module of microcontroller is implemented with interrupts. In embedded systems, interrupt is a signal that microcontroller obtained as an event by hardware or software that microcontroller should take action immediately. In Earl toolkit, received or transmitted data through UART communication is handled by interrupts. Any incoming bit on UART transmit buffer, when microcontroller finishes to send process.

When microcontroller receive a byte, it evaluates the received information according to ANT packet format. More information about ANT packet format can be obtained in ANT Message Protocol and Usage reference [82]. If complete information obtained from received series of byte, microcontroller takes action according to message content. For example, according to ANT Message Protocol and Usage reference, if any ANT packet does not start with 0xA4 (Hexadecimal A4, equals to 164 in decimal representation) microcontroller drop the packet and reject the information. More information about Earl’s ANT packet format will be explained in the end of Section 4.1.2.

\(^4\)In some part of the thesis, device class ID can be called as device set ID. More information about device class ID is explained in Section 3.3.1.2.
UART software module can send single byte and multiple bytes in an order. In embedded software, transmitting String text information through UART communication channel is possible. Therefore, in debug mode, printing text to computer screen via UART channel is possible. As it is mentioned in Section 4.1.2.1.1, stdarg library is used to implement custom printf function. Normally, printf function is implemented in stdio library of ANSI C. However, since memory of MSP430G2553 is not enough to include stdio library and other functionalities in the library is not used in embedded software, I implemented a custom printf function by using stdarg library. Therefore, as in ANSI C printf function, by giving a reference in the string, it is possible to print most commonly used types such as string, char, signed integer, unsigned integer, long, and hexadecimal.

4.1.2.1.5 ADC Module In Earl hardware toolkit, there are two ADC channels available to connect sensors. MSP430G2553 has 1 ADC with 8 channels. In Earl hardware toolkit 2 of these channels are used. In section 4.1.1.2.3, ADC hardware implementation is explained. This section, explains how microcontroller convert analog signal to digital signal and obtain the conversion result.

When ADC is initialized, ADC’s channel 3 and 4 get ready to 8 bit and 10 bit signal conversion. In ADC software module there are functions to get the conversion result in 8 bit and 10 bit data. According to required precision, conversion result can be obtained. ADC conversion is handled by interrupts. Whenever conversion is completed, ADC set its interrupt flag and inform the microcontroller. While conversion is on process, microcontroller waits the ADC module. Whenever ADC module finishes conversion, ADC module becomes ready for the next conversion. For instance, two channels, channel 3 and channel 4, can be sampled right after each other.

4.1.2.1.6 PWM Module PWM signal is discussed in Section 4.1.1.2.3. This section explains how PWM signal is implemented in embedded software. As mentioned in Section 3.3.1.2, Earl should be able to generate PWM signal with given period and duty cycle in order to control actuators. Signal generated by toolkit should be continuous and accurate for better experiences. This is why, one of the Timers of microcontroller is used to generate a continues PWM signal.

Timer in a microcontroller counts number of clock ticks of microcontroller and calculate the approximate time elapsed. Timers can count down to 0 or count up to a value in the range of its register size. MSP430G2553 has 2 16 bit Timer. Timer 1 of MSP430G2553 is assigned to PWM signal generation. Timer 1 is set to count up to a value according to PWM signal period and duty cycle.

In PWM signal, whenever timer reaches to its set value, it flips the value of the PWM signal. By flipping the PWM signal, duty cycle of the PWM signal can be obtained. According to the PWM function prototypes PWM Period can be set in milliseconds and PWM duty cycle can be set in percentage value. By setting these values, microcontroller sets the timer and create the PWM signal in terms of given duty cycle and frequency without being effected from processor running tasks. PWM signal can be switched off if by using Stop function which is declared in PWM Module.

4.1.2.1.7 Scheduler Module Scheduler module enable embedded software to do periodic tasks. Timer 0 of MSP430G2553 is assigned to create periodic events. Embedded software developer can start periodic events by setting scheduler task mode and scheduler period time in milliseconds. Moreover, embedded software designer can stop the scheduled task by using implemented Cancel Task functionality of Scheduler module.

4.1.2.1.8 ANT Module ANT software module, mostly forms an array and transmit the array to ANT hardware module by using UART software module functions and UART hardware module. Likewise, if microprocessor receives an information from ANT hardware module, received information is interpreted by ANT software module. In order to send an information to ANT hardware module, ANT software module uses SendCommand function. This function sends a packet to UART software module in a format.

In ANT software module, ANT hardware module is initialized by sending reset signal to ANT module from microcontroller. After waiting 000 ms, ANT hardware module sends a message to microcontroller that means ANT module is reset and ready for commands. In the following steps, microcontroller sets up the channel and opens the channel for communication. During the channel setup procedure,
microcontroller sends information to ANT module. These information also contain predefined channel settings. These settings were given in Section 3.3.1.2. In the order, messages which are sent to ANT hardware module from microcontroller contains information of Assign Channel, Channel ID, Channel RF Frequency, Channel Period, Channel Search Timeout and Open Channel functions. In order to get more information about these messages, please refer to ANT Message Protocol and Usage reference [82].

As explained in Section 3.3.1.2, every Earl hardware toolkit has a unique ID and class ID. As mentioned in section 4.1.2.1.2 and section 4.1.2.1.3 Earl stores these information in flash memory. These information are used for addressing purposes in broadcasting a packet. As it was be explained in Section 4.1.2, first two bytes of 8 byte information in ANT data packet are allocated by Earl unique ID and Earl class ID. This information is provided by stored information in flash memory.

Whenever a successful message sends or receives, TX/RX LED blinks. This LED is not only for debug purposes, but also gives information how fast data are exchanging between devices. More information about TX/RX LED and how it is implemented is explained in Section 4.1.1.2.6.

4.1.2.1.9 Test Module Test module is designed for two main purposes. Firstly to test software sub modules and secondly to register the device. When hardware toolkit starts, embedded hardware toolkit tests its peripherals. In order to test its peripherals, hardware toolkit uses functions declared in Test module. For example, when hardware toolkit powers up, it gives a PWM output signal for short amount of time and blink both of the LEDs. These test functions are also used for debug purposes as well. As it is discussed in Section 4.1.2.1.1, any of test modes can be set and test the hardware toolkit for a specific test mode.

Whenever a new Earl hardware toolkit module is designed, in order to function properly, Earl hardware toolkit should be registered. Registration mode can be set in Constant module with the name of Working Mode variable. In the end of registration process, hardware toolkit write necessary information to the flash memory.

4.1.2.1.10 Main Module Main module is the top design file that initializes all other software modules. In the beginning of the main module, Watch Dog Timer (WDT) is switched off. WDT is a timer that realizes system malfunctions and takes action. Mostly, WDT resets microcontroller if microcontroller gets stuck or spends more than expected time on one code line. This feature is switched off in microcontroller.

Secondly, microcontroller clock speed is set to 1 MHz as it is mentioned on Table 3.2. Afterwards, Base module is initialized to make the hardware toolkit ready to operate. Then test functions examine peripherals on the board. Finally, ANT software module sets up the communication channel.

In the main program, whenever ANT gets a message that requests to send information (i.e. Request Packet, This information will be explained in section 4.1.2.2.), ANT samples ADCs and creates a wireless communication information packet in a format (Section 4.1.2.2.2) and broadcast it. Whenever ANT software module gets a message from ANT hardware module, UART Receive Interrupt code takes action according to the message received. For example, according to a received message, microcontroller can create a PWM signal that controls an actuator.

4.1.2.2 Wireless Communication Configuration for Hardware Toolkit

Whenever Earl hardware toolkit is switched on, hardware toolkit follow processes stated in Section 4.1.2.1.10. Right after Earl hardware toolkit connected to a channel that mobile phone set already, mobile phone sends “Request Packets” to hardware toolkits which are connected to established channel. Request Packets may content sensor data request information or actuator data information. Whenever hardware toolkit obtains sensor data request packet, it fetches connected sensor data information and sends an “Acknowledgement Packet” to mobile phone. If mobile phone sends an actuator data information to the hardware toolkit, hardware toolkit controls connected actuator according to the information stated in packet. Then, hardware toolkit sends an acknowledgement packet to mobile phone that contains information mobile phone sent to hardware toolkit.

4.1.2.2.1 Earl Toolkit Wireless Communication Channel Structure In Earl toolkit network configuration, all ANT enabled devices are connected to the same channel. Earl hardware toolkit is using shared channel. In this structure, a single ANT node must receive message from many nodes. Multiple
Figure 4.7: Communication Packet Flow Between Hardware toolkit and Mobile Phone
Note: Plain lines represent transmitted information and dashed lines represent returned information.

nodes can share a single independent channel to communicate with the central node. Central node plays master node role. Therefore, all slave nodes are connected to master node. Message exchange is done with unique device ID number.

If any slave nodes leave the channel and return back to channel, slave node which left the channel can reconnect to channel directly. In order to provide this functionality channel search timeout is set to infinity. Therefore, any device can connect to channel whenever device is in the range of master node.

More information about network structure and shared channel configuration can be obtained on ANT Message Protocol and Usage datasheet [65].

Earl hardware toolkit channel configuration is done as in Table 4.3.

<table>
<thead>
<tr>
<th>Setting Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Channel Number</td>
<td>0</td>
</tr>
<tr>
<td>Channel Type</td>
<td>Shared Bidirectional Receive Channel</td>
</tr>
<tr>
<td>Network Number</td>
<td>0</td>
</tr>
<tr>
<td>Radio Frequency</td>
<td>2468 MHz</td>
</tr>
<tr>
<td>TX Type</td>
<td>3</td>
</tr>
<tr>
<td>Device Type</td>
<td>3</td>
</tr>
<tr>
<td>Device ID</td>
<td>XXX (Should be unique for each hardware toolkit)</td>
</tr>
<tr>
<td>ANT Messaging Period</td>
<td>32 Packet per second</td>
</tr>
</tbody>
</table>

Table 4.2: Earl Hardware Toolkit Channel Configuration

Some ANT module supports more than one channel. In this case another channel number can be used for this channel configuration. However, in this thesis, channel 0 is assigned for configurations given above. Device ID of each device is a unique parameter in Earl hardware toolkit configuration. Please refer to Section 4.1.2.1.2 for more information about how Device ID is set for unique devices in Earl toolkit.

Moreover, during the information exchange, some information packets may be lost in the air or may be skipped by hardware toolkit or mobile phone. It is significant to mention that there is not a system to provide an insurance to recover the lost data or an acknowledgement information about the lost data.

4.1.2.2.2 Structure of Wireless Packets ANT technology enables hardware toolkit to exchange information with other ANT enabled devices. Information transmission is done by packets. According to ANT Message Protocol and Usage reference [82], maximum information packet size is formed of 13 Byte. There is also burst mode of information exchange which is sending data in a stream, however it is not supported in Earl toolkit since main purpose of toolkit is not broadcasting data continuously.
As mentioned in Section 4.1.2.1.4, in Earl hardware toolkit, microcontroller and ANT module communicates through UART channel. Microcontroller commands ANT module with a command structure as explained in ANT Message Protocol and Usage reference [82]. By exchanging information between microcontroller and ANT module, hardware toolkit can transmit and receive wireless packets. Packets that provides the information exchange between ANT enabled devices are called broadcast messages. Earl has constant broadcast message structure for information exchange. In Earl, every broadcast message consists of 13 Byte information. The broadcast message structure is shown in the Figure 4.8.

<table>
<thead>
<tr>
<th>SYNC</th>
<th>MSG LENGTH</th>
<th>CHANNEL NO</th>
<th>MSG ID</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>D7</th>
<th>D8</th>
<th>CHECKSUM</th>
</tr>
</thead>
</table>

Figure 4.8: Earl Broadcast Message Structure

Note: Each box represents 1 Byte information.

First byte of data structure which is called SYNC byte, enables ANT communication to transmit or receive whole packet. This field also helps ANT module to form the rest of the packet by declaring first byte of the packet. Second byte of packet has information about number of data bytes in the message. This information helps ANT module to shape the packet size. Third byte shows channel number of the packet transmitted or received. According to ANT datasheet, it is possible to broadcast data channels between 0 and 7. Some ANT modules support single channel, therefore these ANT modules can only use channel 0. In Earl toolkit implementation, all Earl enabled toolkits operates on channel 0. Message ID represents type and functionality of packet. Detailed information and message id codes are explained in ANT data sheet [65]. The last byte of the packet is assigned to packet checksum. Therefore, ANT module or microcontroller can use this information in order to validate the packet received or transmitted without any corruption.

Earl has 8 bytes information data allocated out of 13 bytes standard packet. Last 2 bytes of 8 bit data package (D7 and D8) are unused and can be used for future purposes.

- **Earl Device ID (D1):** Earl toolkit can exchange information with 65535 Earl enabled device in the same channel [65]. In order to setup such a network, every ANT enabled device has a unique ID. This is why, Earl is using Shared ANT Channel [65]. More information about ANT shared channel can be found in ANT Message Protocol and Usage [65]. As explained in section 4.1.2.1.2, Earl has unique Device ID and Device Set ID. Device ID is stored in this byte of broadcast message.

- **Earl Device Set ID (D2):** Device Set ID is stored in this byte of broadcast message.

- **SensorID (D3):** Peripheral ID represents unique ID of sensors and actuators types used in the system. Therefore, Earl toolkit can differentiate the broadcast message content over the network. Any number between 0 to 255 is possible to represent sensors and actuators. Currently 1 is assigned to analog sensors and 2 is assigned to analog actuators.

- **DataHigh (D4):** Every broadcast message represent one sensor or actuator with its value. Value of represented peripheral contains 16 bit (2 byte) information. This byte is assigned to most significant byte value of peripheral.

- **DataHigh (D5):** This byte is assigned to least significant byte value of peripheral.

- **TimeDuration (D6):** This byte is reserved for active time duration of actuator in milli seconds. To active the actuator continuously it should set to 0. For sensors, this value is supposed to be 0.

As it explained in Figure 4.8, every wireless packet contains two byte address information and 6 Bytes payload in this configuration. However, 2 byte information is not used in the packet. Since channel frequency is set to 32 Hz, communication speed can be calculated as follows.

\[
\text{Communication Speed} = \frac{\text{Packet}}{\text{Second}} \times 8\text{Byte} \times \frac{\text{Bits}}{\text{Byte}} = 2048\text{bps} \tag{4.1}
\]
CHAPTER 4. IMPLEMENTATION

4.2 Android Mobile Application

Android mobile application is another main part of described system in Section 3.1. Main functionality of Android mobile application is to obtain and send information to Earl hardware toolkits and provide information exchange between Earl web services and hardware toolkit.

In this section, first I will talk about Android mobile device that enables hardware toolkits to exchange information with web services. Next, I will give information about Android software development platform. Thirdly, I will explain how Android mobile application is designed. Finally, I will give information about how mobile application is working.

4.2.1 Android Mobile Phone

In Earl toolkit system, Android mobile phone enables Earl hardware toolkit to exchange information with web services. In order to provide communication between Earl hardware toolkit and Android mobile phone, ANT wireless communication supported mobile phone should be used. Currently, there are some mobile devices that supports ANT wireless communication technology [83]. In this thesis, Sony Xperia Active [84] mobile phone is used for communication. Sony Xperia Active is an ANT wireless communication technology enabled Android mobile phone. Sony Xperia Active has Android 4.0.4 operating system\(^5\) installed and it supports WiFi and 3G cellular network support [85]. By using WiFi and 3G cellular network connection, Android mobile application can access to Earl web services.

\(^{5}\)Android OS 4.0.4 is also called as Android Ice Cream Sandwich.
In the following section, development environment of Android application is discussed.

### 4.2.2 Android Software Development Platform

In Android application software design Android Development Tools (ADT) Revision 21.1 [86] is used. ADT is a plugin for Integrated Development Environment (IDE) [87]. In this thesis, Eclipse 4.2.2 is used as software development environment.

During the software development process Microsoft Windows 7 32-Bit Professional operating system is used. Android mobile application is written in Java programming language [88].

### 4.2.3 Android Mobile Application Implementation

Earl mobile platform has been designed in order to enable hardware toolkit to communicate with Earl web services. In the implementation, in order to function ANT wireless communication module on ANT enabled Android mobile device, ANTLib library, which is provided by Official ANT Developers, is used [89]. This library is an interface between hardware ANT chipset and Android application. To use the application on an Android supported device, ANT module and Internet connection modules are needed for proper functionality of toolkit.

For Android application essential permissions are enabled to control internal needed modules. Therefore, ANT module permissions and Internet access permissions are enabled to enable communication with hardware toolkit. Moreover, during the Android application development process “Write External Storage” permissions are also allowed to write and read the logs on external storage unit such as SD card.

In Android application, there is one main program, which is called ANTActivity. Main functionalities of mobile application, which are opening and closing the ANT communication channel are functioning on this main program. Once mobile application user opens communication channel, mobile application configures ANT wireless communication channel settings as given values in the Table 4.3.

<table>
<thead>
<tr>
<th>Setting Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Channel Number</td>
<td>0</td>
</tr>
<tr>
<td>Channel Type</td>
<td>Shared Bidirectional Transmit Channel</td>
</tr>
<tr>
<td>Network Number</td>
<td>0</td>
</tr>
<tr>
<td>Radio Frequency</td>
<td>2468 MHz</td>
</tr>
<tr>
<td>TX Type</td>
<td>3</td>
</tr>
<tr>
<td>Device Type</td>
<td>3</td>
</tr>
<tr>
<td>Device ID</td>
<td>1</td>
</tr>
<tr>
<td>ANT Messaging Period</td>
<td>8 Packet per second</td>
</tr>
</tbody>
</table>

Table 4.3: Earl Hardware Toolkit Channel Configuration

Some ANT enabled Android phones supports more than one channel. In this case another channel number can be used for this channel configuration. However, in this thesis, channel 0 is assigned for configurations as given above. Device ID of mobile phone is configured as 1. Please refer to Section 4.1.2.1.2 for more information about how Device ID is set for unique devices in Earl toolkit.

In mobile application, there is a list of Device IDs and Device Set IDs. Mobile application sends request packets to hardware toolkits which are connected to communication channel. Mobile phone sends request packets to hardware toolkits according to the list order in a loop and hardware toolkits reply back with the same order. If the hardware toolkit gets involved in the list of Device IDs and does not connect to wireless communication channel, mobile phone still sends a request packet. However, in this case it would not reply back from any hardware toolkit which are connected to channel. Request packets and Acknowledgment packets are exchanged while the technology developer is having experience with the sensors and actuators.

To illustrate this, in the Figure 4.10 6 hardware toolkits are connected to mobile phone. Each hardware toolkit will have a unique Device ID. These devices may share the same Device Set ID. In this example, let’s say Device IDs and Device Set IDs are given as in the Table 4.4.

In this example, mobile phone send request packets in the same order as listed in Table 4.4. For example, Mobile Phone (MP) sends request packet to Hardware Toolkit (HT) 3. According to the
CHAPTER 4. IMPLEMENTATION

Figure 4.10: Hardware Toolkits Connection Example

Note: Plain lines represent transmitted information and dashed lines represent returned information.

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Device ID</th>
<th>Device Set ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile Phone</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Hardware Toolkit 1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Hardware Toolkit 2</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>Hardware Toolkit 3</td>
<td>41</td>
<td>1</td>
</tr>
<tr>
<td>Hardware Toolkit 4</td>
<td>21</td>
<td>1</td>
</tr>
<tr>
<td>Hardware Toolkit 5</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Hardware Toolkit 6</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Hardware Toolkit 7</td>
<td>71</td>
<td>1</td>
</tr>
<tr>
<td>Hardware Toolkit 8</td>
<td>81</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4.4: Earl Hardware Toolkit Mobile Phone Connection Example Device Parameters

information in the request packet, either HT controls connected actuators or measure sensor information that is connected to HT. In the end, HT replies back with Acknowledgment packet to MP. Next, MP applies the same procedure next device in the list which is HT 4 in this example. In this example Hardware Toolkit 7 and 8 are not connected to wireless channel. Therefore, MP sends request packet but can not receive back acknowledgement packet.

Next example, explains how MP exchanges information with Earl Web Services (EWS). While MP communicating with 3 HTs, it also communicate with EWS. There are two EWS communications which are Information Pushing and Information Polling. Any acknowledgement packet received from HTs is sent to EWS. In order to send information to EWS, MP uses Hypertext Transfer Protocol (HTTP) [90]. By using HTTP, MP calls a PHP script located on EWS Server called PostData.php. Therefore, MP can send all acknowledgement information coming from all connected HTs to MP. This communication is called Information Pushing to EWS.

In Earl toolkit, HT, MP and EWS exchange data in a proper format. This format contains sensor or actuator data information discussed in Section 4.1.2.2.2. While MP exchanges information with EWS, every packet is tagged with a unique ID by MP. Purpose of tagging the packets is differentiating proof of concept designs created by technology designers. Designers can store and reuse the information in the EWS database that the information sent or received from MP during the experience with created proof of concept systems. This is why, every experience with designed proof of concept system is labeled with “ExperimentID”. Therefore, MP sends Acknowledgement Packet information with ExperimentID information to EWS by calling the PostData PHP page.

Second communication way with EWS is called Information Polling. MP polls the EWS for any

\footnote{Please refer Earl Web Services Implementation section for more information.}
new information to be sent to HT. MP polls the EWS in periodic time intervals. Every 125 milliseconds, MP calls Request.php PHP file located on EWS to check new informations on EWS. In order to call Request.php, MP uses HTTP like Information Pushing function. Every call to EWS is done with ExperimentID. According to the reply received from EWS, MP sends incoming information to HT with its unique Device ID and Device Set ID.

This section explained how mobile application is implemented. In the next section how Earl Web Services implemented.

### 4.3 Earl Web Services

Web service is a system that enable technology developers to control and monitor hardware toolkit. By using Earl Web Services (EWS), technology developers can send information to or receive sensors information which are connected to hardware toolkit.

Earl web services are composed of a web server, web server application, and database that all information is stored. In this section, first, I will explain about server computer details. Secondly, I will explain how web services of Earl toolkit are implemented.

#### 4.3.1 Web Server Details

EWS is designed on a web server located in Mobile Life Centre, Stockholm [91]. This web server is a desktop PC with following properties.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mother Board</td>
<td>Winbond Electronics 0F8098 Motherboard</td>
</tr>
<tr>
<td>Processor (CPU)</td>
<td>Intel(R) Pentium(R) 4 CPU 3.00GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>1GB DIMM DDR Synchronous 533 MHz</td>
</tr>
<tr>
<td>Graphics</td>
<td>Intel Onboard 82945G/GZ Integrated Graphics</td>
</tr>
<tr>
<td>Ethernet</td>
<td>Broadcom NetXtreme BCM5751 Gigabit Ethernet PCI</td>
</tr>
</tbody>
</table>

Table 4.5: EWS Web Server Specifications

In EWS, Ubuntu 12.10 Server Edition is installed on web server. In order to provide web service, Apache Software Foundation 2.2 [92] is installed. Furthermore, for database support, MySQL 5.6 is installed [38]. EWS is implemented with PHP scripting language. This is why, PHP 5.4.12 is installed on web server.
CHAPTER 4. IMPLEMENTATION

43

EWS is designed to support multiple connection at the same time. According to web service requirements which is given in Section 3.3.3, EWS server is supporting 20 concurrent connections at the same time. In order to arrange this property, default Apache settings are used. According to “MaxClients Directive” setting in Apache 2.2, 256 concurrent settings can be provided at the same time to the server.

In EWS design, Vi text editor application [93] is used to write web applications. During the software development process Ubuntu 12.10 Server Edition operating system is used. During implementation, web user interfaces of EWS is designed using HTML and JavaScript scripting languages. Mobile phone communication interface is designed in PHP scripting language. For database storage MySQL service is used.

4.3.2 Server System Implementation

Web server has four main functionalities. These functionalities will be explained in following sections.

4.3.2.1 Information Pushing

Mobile phone collects information from hardware toolkits that are connected to itself. All collected information is pushed to Earl web server by using HTTP protocol. This process is explained in Section 4.2.3. This section explains how web server reacts to received information.

When mobile phone calls PostData.php with information obtained from hardware toolkit and ExperimentID, these information are stored to Earl web services database. In order to obtain posted information, HTTP Post request method is used [94]. In this method, sensor information is posted to server with ExperimentID via web page URL.

It is significant that whenever mobile phone tries to send information to web server and fails in the transmission, mobile phone drop the information and will not carry about the lost information. Therefore, there is no guarantee that mobile phone transmits all information to web server.

4.3.2.2 Information Request from Web Server

Application developers can send information to hardware toolkit by using web services. In order to send information to hardware toolkits from web services, application developers should add the information to Earl web services database with ExperimentID. Mobile phone requests PHP page periodically to obtain the information that application developers added to Earl web services database. Request page is called with ExperimentID and DeviceID information. Based on these information, requested page search stored information in the Earl web services database and returns an answer to mobile phone. If web server can not find any result for search process, return answer will be empty result. Therefore, mobile phone will not transfer any information to hardware toolkit for the request.

4.3.2.3 Monitoring and Controlling Hardware Toolkit from Web Server

Application developers can transmit and receive information with hardware toolkit. In order to receive information from hardware toolkit, application developers should request information from EWS web server via ExperimentID and DeviceID. Requested information will be searched on EWS Database and
CHAPTER 4. IMPLEMENTATION

4.4 Working System

In this section, I will give information about power consumption of Earl hardware toolkit. I will also give information about how all these data are measured. Moreover, I will present data about latency of data exchange between hardware toolkit, mobile phone and Earl web services.

4.4.1 Hardware Toolkit Power Consumption

In this section, power consumption of hardware toolkit will be shown in the sensing and actuator control mode. Furthermore, ANT Transmit and receive power consumption will also be shown in the following section. In the end of this section, all power consumption data will be summarized in a table.
Power consumption of hardware toolkit is measured in the following way. Earl board is a portable hardware toolkit. This is why, it uses a 3.7 Volt rechargeable single cell polymer Lithium Ion battery. This battery has 850mAh capacity. Voltage of the fully charged battery is measured with Sinometer MY68 Multimeter [27] and according to the measurement, battery has 3.76 V between anode and cathode terminals. In order to measure the flowing current to the hardware toolkit, a serial resistor is connected serially to the battery and hardware toolkit.

According to the Equation 5.1, current passing through a resistive component can be calculated with the ratio of voltage over resistance. In order to put low resistance to decrease the current consumption of experiment setup, lowest resistor value I could find is used. Thus, a small resistor is chosen with the value of 10 Ω. Furthermore, according to measurement with Sinometer MY68 multimeter, resistor’s resistance is measured as 10.3 Ω. This resistor will be called $R_{ex}$ in the rest of the section. An oscilloscope probe is connected to both sides of the resistor. Therefore, voltage passing across the resistor can be observed on the oscilloscope screen. According to Equation 5.1, current consumption can be calculated. Moreover, power consumption can be calculated via Equation 5.2. In the end, experiment setup is demonstrated in Figure 4.16.

![Oscilloscope Probe](image)

**Figure 4.16: Current Consumption Measurement Setup**

In this experiment setup, current consumption of whole Earl hardware toolkit is measured. In the following sub sections, this setup will be used to measure power consumption of toolkit in different modes.

### 4.4.1.1 Power Consumption In Receive Mode

Earl hardware toolkit is configured as receive only mode where mobile phone is configured as shared channel master node of established channel. In receive only mode, Earl hardware toolkit only receives messages coming from channel master. In this case, mobile phone sends request packets to Earl hardware toolkits which are connected to mobile phone in every channel period. In the following Figure 4.17, voltage across $R_{ex}$ will be shown. As it is explained in Section 4.4.1, power consumption of hardware toolkit will be calculated. It is significant that, actuating and sensing modes are switched off. Therefore, Earl hardware toolkit only processes receive related functionalities. Note that all LEDs are switched off in order to decrease power consumption since each LED consumes approximately 2 mA current consumption at 3.25 (Vcc) Volt.

In the Figure 4.17, mobile phone sends information to hardware toolkit in 32 Hz. In the figure two voltage peaks can be observed.

According to the measurements by oscilloscope (Figure 4.17), average current consumption of the
hardware toolkit is 1.13 mA at 3.25 V \( V_{cc} \) voltage when device is not receiving any message from mobile phone (Idle mode). Therefore, power consumption of the device can be calculated in following by using Equation 5.3 and 5.7.

\[
\text{Average Power consumption of Idle Mode in Watt} = 1.13 \times 10^{-3} \times 3.25V = 0.0036725\text{Watt} \quad (4.2)
\]

\[
\text{Average Power consumption of Idle Mode in Joule} = 0.0036725\text{Watt} \times 3600\text{sec} = 13.221\text{Joule} \quad (4.3)
\]

Moreover, when hardware toolkit receives information, peak current consumption of the hardware toolkit is measured as 14.20 mA at 3.25 V \( V_{cc} \) where average power consumption of receive event is measured 13.7 mA. A receive event takes 0.70 ms. In following power consumption is calculated when hardware toolkit receive information.

\[
\text{Peak Power consumption of Receive Event in Watt} = 14.20 \times 10^{-3} \times 3.25V = 0.04615\text{Watt} \quad (4.4)
\]

\[
\text{Peak Power consumption of Receive Event in Joule} = 0.04615\text{Watt} \times 3600\text{sec} = 166.14\text{Joule} \quad (4.5)
\]

\[
\text{Average Power consumption of Receive Event in Watt} = 13.70 \times 10^{-3} \times 3.25V = 0.044525\text{Watt} \quad (4.6)
\]

\[
\text{Average Power consumption of Receive Event in Joule} = 0.044525\text{Watt} \times 3600\text{sec} = 160.29\text{Joule} \quad (4.7)
\]

Note that all ANT configurations are set to its defaults as it is mentioned in ANT Message Protocol and Usage. Receive sensitivity is set to its default value which is \(-85\) dBm. Ant module is running on 3.25 Volt. In this mode, received only channel is configured as it is stated in ANT Message Protocol and Usage. Therefore, it is much clear to see only receive activities. For more information please refer to ANT Message Protocol and Usage [82].
4.4.1.2 Power Consumption In Transmit Mode

Earl hardware toolkit is configured as transmit only mode where mobile phone is configured as shared channel master node of established channel. In transmit only mode, hardware toolkit only transmit information to channel master which is mobile phone in this case. In the following figure, voltage across $R_{ex}$ will be shown. As it is explained in Section 4.4.1, power consumption of hardware toolkit will be calculated. It is significant that, actuating and sensing modes are switched off. Therefore, Earl hardware toolkit only process transmit related functionalities. Note that all LEDs are switched off in order to decrease power consumption since each LED consumes approximately 2 mA current consumption at 3.25 (V_{cc}) Volt.

![Figure 4.18: Current Consumption Measurement for Transmit Mode](image)

According to the measurements by oscilloscope (Figure 4.18), average current consumption of the hardware toolkit is 1.21 mA at 3.25 V $V_{cc}$ voltage when device is not transmitting anything (Idle mode). Therefore, power consumption of the device can be calculated as follows by using Equation 5.3 and 5.7.

\[
\text{Average Power consumption of Idle Mode in Watt} = 1.21 \times 10^{-3} \times 3.25V = 0.0039325 \text{Watt} \tag{4.8}
\]

\[
\text{Average Power consumption of Idle Mode in Joule} = 0.0039325 \text{Watt} \times 3600 \text{sec} = 14.157 \text{Joule} \tag{4.9}
\]

Moreover, when hardware toolkit started to transmit information, peak current consumption of the hardware toolkit is measured as 10.50 mA at 3.25 V $V_{cc}$ where average current consumption is 10.2 mA. A transmit event takes 0.35 ms. In following power consumption is calculated when hardware toolkit transmit information as follows:

\[
\text{Peak Power consumption of Transmit Mode in Watt} = 10.50 \times 10^{-3} \times 3.25V = 0.034125 \text{Watt} \tag{4.10}
\]

\[
\text{Peak Power consumption of Transmit Mode in Joule} = 0.034125 \text{Watt} \times 3600 \text{sec} = 122.85 \text{Joule} \tag{4.11}
\]
CHAPTER 4. IMPLEMENTATION

Average Power consumption of Transmit Mode in Watt = \(10.20 \times 10^{-3} \times 3.25\) \(\text{V} = 0.03315\) Watt (4.12)

Average Power consumption of Transmit Mode in Joule = \(0.03315\) Watt \(\times 3600\) sec = 119.34 Joule (4.13)

Note that all ANT configurations are set to its defaults as it is mentioned in ANT Message Protocol and Usage[82]. Transmission power is set to maximum which is 0 dBm. Furthermore, in order to get more information about channel configuration please refer to Section 4.1.2.2. ANT module is running on 3.25 Volt. In this mode, transmit only channel is configured as it is stated in ANT Message Protocol and Usage. Therefore, it is much clear to see only transmit activities. For more information please refer to ANT Message Protocol and Usage [82].

So far, individual power consumptions of hardware toolkit in receive only and transmit only modes. It is significant to mention that Earl hardware toolkit is configured as shared channel slave device. Therefore, whenever it receives information, it may reply back to channel master device. In following Figure 4.19, current consumptions are on both receive and transmit instants. In Figure 4.19, first peak on the left shows received message from master device and the peak on the right shows transmitted message to master device.

![Figure 4.19: Current Consumption Measurement for Shared Channel Slave Mode](image)

Moreover, in shared channel configurations, communication range between master node and slave node is tested on 10 meters distance. The environment where test is done is a daily office environment with normal conditions [43]. In the test room, many wireless devices were operating on 2.4 GHz Wireless Commercial Band.

In the following power consumption the hardware toolkit is calculated in different event instances. According to the measurements by oscilloscope (Figure 4.19), average current consumption of the hardware toolkit is 1.70 mA at 3.25 V \(V_{cc}\) voltage when device is not transmitting anything (Idle mode). Therefore, power consumption of the device can be calculated as follows by using Equation 5.3 and 5.7.
Average Power consumption of Idle Mode in Watt = $1.70 \times 10^{-3}A \times 3.25V = 0.005525$Watt \hspace{1cm} (4.14)

Average Power consumption of Idle Mode in Joule = $0.005525$Watt × 3600sec = $19.89$Joule \hspace{1cm} (4.15)

Moreover, when hardware toolkit receive information, peak current consumption of the hardware toolkit is measured as $15.00$ mA at $3.25$ V where average power consumption $14.4$ mA. A receive event takes $0.7$ ms. Power consumption is calculated when hardware toolkit receive information as follows:

Peak Power consumption of Receive Event in Watt = $15.00 \times 10^{-3}A \times 3.25V = 0.04875$Watt \hspace{1cm} (4.16)

Peak Power consumption of Receive Event in Joule = $0.04875$Watt × 3600sec = $175.89$Joule \hspace{1cm} (4.17)

Average Power consumption of Receive Event in Watt = $14.40 \times 10^{-3}A \times 3.25V = 0.0468$Watt \hspace{1cm} (4.18)

Average Power consumption of Receive Event in Joule = $0.0468$Watt × 3600sec = $168.48$Joule \hspace{1cm} (4.19)

Right after the receive event, hardware toolkit transmits information. In this case, current consumption of the hardware toolkit is measured as $11.00$ mA at $3.25$ V where average power consumption is $10.00$ mA. A transmit event takes $0.3$ ms. Power consumption is calculated when hardware toolkit transmit information as follows:

Peak Power consumption of Transmit Event in Watt = $11.00 \times 10^{-3}A \times 3.25V = 0.03575$Watt \hspace{1cm} (4.20)

Peak Power consumption of Transmit Event in Joule = $0.03575$Watt × 3600sec = $128.7$Joule \hspace{1cm} (4.21)

Average Power consumption of Transmit Event in Watt = $10.00 \times 10^{-3}A \times 3.25V = 0.0325$Watt \hspace{1cm} (4.22)

Average Power consumption of Transmit Event in Joule = $0.0325$Watt × 3600sec = $117$Joule \hspace{1cm} (4.23)

### 4.4.1.3 Power Consumption In Sensing Mode

Earl hardware toolkit is configured as only sensing mode. In the sensing mode, every incoming request that contains sensing information (such as sensor ID), triggers sensing unit of Earl hardware toolkit. In order to demonstrate this, in every request event time interval sensing unit is called. In the following figure, voltage across $R_{ex}$ will be shown. As it is explained in Section 4.4.1, power consumption of hardware toolkit will be calculated. It is significant that, transmit and receive modes are switched off. Therefore, Earl hardware toolkit only processes receive related functionalities. In this example, an analog force sensor is used [95]. In the Table 4.6, more information about the force sensor can be obtained.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Type</td>
<td>Force Sensor</td>
</tr>
<tr>
<td>Sensor Output Type</td>
<td>Ratiometric (Directly proportional to an input)</td>
</tr>
<tr>
<td>Min. Force</td>
<td>0 N</td>
</tr>
<tr>
<td>Max. Force</td>
<td>39.2 N</td>
</tr>
<tr>
<td>Current Consumption Max</td>
<td>500µA</td>
</tr>
<tr>
<td>Output Impedance</td>
<td>10 kΩ</td>
</tr>
<tr>
<td>Operating Temperature Min</td>
<td>10 Deg. C</td>
</tr>
<tr>
<td>Operating Temperature Max</td>
<td>40 Deg. C</td>
</tr>
<tr>
<td>Lifespan</td>
<td>100000 actuations</td>
</tr>
</tbody>
</table>

Table 4.6: Force Sensor Specifications
Note that all LEDs are switched off in order to decrease power consumption since each LED consumes approximately 2 mA current consumption at 3.25 $V_{cc}$ Volt.

According to the measurements by oscilloscope (Figure 4.21), current consumption of the hardware toolkit is 1.3 mA at 3.25 V $V_{cc}$ voltage when device is not sensing anything (Idle mode). Sampling one ADC channel takes 80 $\mu$s. Therefore, power consumption of the device can be calculated by using Equation 5.3 and 5.7.

$$\text{Power consumption in Watt} = 1.3 \times 10^{-3} \text{A} \times 3.25 \text{V} = 0.004225 \text{Watt}$$  \hspace{1cm} (4.24)
Power consumption in Joule = 0.004225\text{Watt} \times 3600\text{sec} = 15.21\text{Joule} \quad (4.25)

Moreover, when hardware toolkit starts sensing, peak current consumption of the hardware toolkit is measured as 6.20 mA at 3.25 V \(V_{cc}\) where average current consumption is 5.60 mA. Power consumption is calculated when hardware toolkit start sensing as bellow:

Peak Power consumption of Sensing Event in Watt = 6.20 \times 10^{-3}\text{A} \times 3.25\text{V} = 0.02015\text{Watt} \quad (4.26)

Peak Power consumption of Sensing Event in Joule = 0.02015\text{Watt} \times 3600\text{sec} = 72.54\text{Joule} \quad (4.27)

Average Power consumption of Sensing Event in Watt = 5.60 \times 10^{-3}\text{A} \times 3.25\text{V} = 0.0182\text{Watt} \quad (4.28)

Average Power consumption of Sensing Event in Joule = 0.0182\text{Watt} \times 3600\text{sec} = 65.52\text{Joule} \quad (4.29)

Note that there are two consecutive peaks in the wave shown. The reason of these peaks is that there are two ADC channels. Hardware toolkit samples two channels right after the first one finishes.

4.4.1.4 Power Consumption In Actuating Mode

Earl hardware toolkit is configured as only actuating mode. In the actuating mode, every incoming request that contains actuating information (such as actuator ID), triggers actuating unit of Earl hardware toolkit. In order to demonstrate this, in every request event time interval actuating unit is called. As it is explained in Section 4.4.1, power consumption of hardware toolkit will be calculated. It is significant that, transmit and receive modes are switched off. Therefore, Earl hardware toolkit only processes receive related functionalities. In this example, an analog haptic vibrator is used [96]. In the following Table 4.7, more information about the haptic vibrator can be obtained.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMS Voltage</td>
<td>2 Volt RMS</td>
</tr>
<tr>
<td>Rated Resonant Frequency</td>
<td>175 Hz</td>
</tr>
<tr>
<td>Max. Operating Voltage</td>
<td>2.05 Volt RMS</td>
</tr>
<tr>
<td>Certified Start Voltage</td>
<td>0.3 Volt RMS</td>
</tr>
<tr>
<td>Max. Operating Current</td>
<td>90 mA RMS</td>
</tr>
<tr>
<td>Typical Max. Terminal Resistance</td>
<td>28 Ω</td>
</tr>
<tr>
<td>Typical Max. Terminal Inductance</td>
<td>130 μH</td>
</tr>
<tr>
<td>Typical Start Voltage</td>
<td>0.15 Volt RMS</td>
</tr>
<tr>
<td>Typical Lag Time</td>
<td>5 ms</td>
</tr>
<tr>
<td>Typical Rise Time</td>
<td>19 ms</td>
</tr>
<tr>
<td>Typical Stop Time</td>
<td>275 ms</td>
</tr>
<tr>
<td>Min. Operating Temperature</td>
<td>−25 Deg. C</td>
</tr>
<tr>
<td>Max. Operating Temperature</td>
<td>70 Deg. C</td>
</tr>
</tbody>
</table>

Table 4.7: Haptic Vibrator Specifications

Note that all LEDs are switched off in order to decrease power consumption since each LED consumes approximately 2 mA current consumption at 3.25 (\(V_{cc}\)) Volt.

During the power consumption measurement test, 6 different PWM signal applied to motor driver component by microcontroller. In the Table 4.8, tested duty cycles and frequency values are given with current consumption.

According to information in Table 4.8, power consumption is calculated as follows:

**Test 1:**

Power consumption in Watt = \(12.42 \times 10^{-3}\text{A} \times 3.25\text{V} = 0.040365\text{Watt} \quad (4.30)

Power consumption in Joule = 0.040365\text{Watt} \times 3600\text{sec} = 145.314\text{Joule} \quad (4.31)

**Test 2:**

Power consumption in Watt = \(19.51 \times 10^{-3}\text{A} \times 3.25\text{V} = 0.0634075\text{Watt} \quad (4.32)
CHAPTER 4. IMPLEMENTATION

Figure 4.22: Haptic Actuator

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Frequency</th>
<th>Duty Cycle</th>
<th>Current Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1</td>
<td>175 Hz</td>
<td>5</td>
<td>12.42 mA</td>
</tr>
<tr>
<td>Test 2</td>
<td>175 Hz</td>
<td>10</td>
<td>19.51 mA</td>
</tr>
<tr>
<td>Test 3</td>
<td>175 Hz</td>
<td>20</td>
<td>22.73 mA</td>
</tr>
<tr>
<td>Test 4</td>
<td>175 Hz</td>
<td>30</td>
<td>26.25 mA</td>
</tr>
<tr>
<td>Test 5</td>
<td>175 Hz</td>
<td>40</td>
<td>36.35 mA</td>
</tr>
<tr>
<td>Test 6</td>
<td>175 Hz</td>
<td>50</td>
<td>47.29 mA</td>
</tr>
</tbody>
</table>

Table 4.8: Haptic Driver Test Inputs and Current Consumptions

![Figure 4.23: Haptic Driver Test - Current Consumption](image)

Test 3:
Power consumption in Joule $= 0.0634075 \text{Watt} \times 3600 \text{sec} = 228.267 \text{Joule}$  
(4.33)

Power consumption in Watt $= 22.73 \times 10^{-3} \text{A} \times 3.25 \text{V} = 0.0738725 \text{Watt}$  
(4.34)

Power consumption in Joule $= 0.0738725 \text{Watt} \times 3600 \text{sec} = 265.941 \text{Joule}$  
(4.35)

Test 4:
Power consumption in Watt $= 26.25 \times 10^{-3} \text{A} \times 3.25 \text{V} = 0.0853125 \text{Watt}$  
(4.36)
4.4.1.5 Power Consumption In Microcontroller Sleep Mode

In this section, minimum power consumption of hardware toolkit is measured. In order to decrease all the power consumptions:

- Microcontroller is set to lowest sleep mode (According to MSP430G2533 data sheet, LPM4)
- ANT module is switched off.
- All LEDs are switched off.
- Motor driver is disabled.

In the end, current consumption of the hardware toolkit is measured as 0.4 mA at 3.25 Volt. Power consumption is calculated as follows:

\[
\text{Power consumption in Watt} = 0.4 \times 10^{-3} \times 3.25 = 0.0013 \text{Watt} \quad (4.42)
\]

\[
\text{Power consumption in Joule} = 0.0013 \times 3600 = 4.68 \text{Joule} \quad (4.43)
\]

4.4.1.6 Power Consumption Results

This section discusses the power consumption results of the hardware toolkit. Earl hardware toolkit has different activities such as information receive, transmit, obtaining sensor value, controlling actuator and sleep mode. All these events are handled by embedded software. According to event handling on microcontroller, hardware toolkit consumes different power consumption. In previous sections, power consumption of hardware toolkit has demonstrated. In following table, all measured power consumptions are listed.

According to power consumption test results, any activity on hardware toolkit effect the power consumption. Therefore, battery life may change according to number of activities that hardware toolkit is doing. Test results demonstrate that most power consuming activities are wireless information change activities. In Earl hardware toolkit, 32 packets of information can be exchanged between mobile phone and hardware toolkit in a second. High message traffic between the mobile phone and hardware toolkit increases the power consumption, thus decreases the battery life.

On the other hand, actuators, for example haptic vibrator which is described in previous section, consumes more power compared to activities listed above. This is why, battery life of hardware toolkit will be less than non actuator used applications.

In order to increase the battery life, hardware toolkit switches to sleep mode when there is no activity to handle on hardware toolkit. Therefore, hardware toolkit decreases the power consumption compared to idle mode. However, hardware toolkit does not switch to sleep mode when it is controlling an actuator.

In this section, approximate power consumption of hardware toolkit will be discussed. In order to describe approximate power consumption, a sensing application will be given as an example. In this application, a force sensor used above connected to analog inputs of hardware toolkit. In this application, mobile phone request information from hardware toolkit in 32 Hz. According to Figure 4.25 hardware toolkit receives information every 32 millisecond. For every request, hardware toolkit obtains sensor information and transmit to mobile phone by using ANT wireless communication. Rest of the times when hardware toolkit is not handling an event, hardware toolkit switches to sleep mode to decrease the power consumption and wait the next event. This periodic event continues 32 times per second. Average
Table 4.9: Hardware Toolkit Peak Power Consumption Results

current consumption is explained in following. In following Equation 4.44, average current consumption and event duration time are used from Table 4.10.

\[
\text{Average Current Consumption} = \frac{\text{Receive Event Current Consumption} \times \text{Receive Event Duration}}{\text{Event Period Time}} + \frac{\text{Transmit Event Current Consumption} \times \text{Transmit Event Duration}}{\text{Event Period Time}} + \frac{\text{Sensing Event Current Consumption} \times \text{Sensing Event Duration}}{\text{Event Period Time}} + \frac{\text{Sleep Current Consumption} \times \text{Sleep Duration}}{\text{Event Period Time}}
\] (4.44)
### Activity Name

<table>
<thead>
<tr>
<th>Current Consumption in mA at 3.25V (in Activity)</th>
<th>Power Consumption in Idle Mode (Watt)</th>
<th>Power Consumption in Idle Mode (Joule)</th>
<th>Power Consumption in Activity (Watt)</th>
<th>Power Consumption in Activity (Joule)</th>
<th>Event Duration in millisecond (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Power Consumption in Receive Mode</td>
<td>13.70</td>
<td>13.221</td>
<td>0.044525</td>
<td>160.29</td>
<td>0.70</td>
</tr>
<tr>
<td>Average Power Consumption in Transmit Mode</td>
<td>10.20</td>
<td>14.157</td>
<td>0.03315</td>
<td>119.34</td>
<td>0.35</td>
</tr>
<tr>
<td>Average Power Consumption for Receive in Ear1 Applications</td>
<td>14.40</td>
<td>19.89</td>
<td>0.0468</td>
<td>168.48</td>
<td>0.70</td>
</tr>
<tr>
<td>Average Power Consumption for Transmit in Ear1 Applications</td>
<td>10.00</td>
<td>19.89</td>
<td>0.0325</td>
<td>117</td>
<td>0.30</td>
</tr>
<tr>
<td>Average Power Consumption in Sensing Mode</td>
<td>5.60</td>
<td>15.21</td>
<td>0.0182</td>
<td>65.52</td>
<td>0.08</td>
</tr>
<tr>
<td>Average Power Consumption in Sleep Mode</td>
<td>0.40</td>
<td>-</td>
<td>0.0013</td>
<td>4.68</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4.10: Hardware Toolkit Average Power Consumption Results

Average Current Consumption (mA)

\[
\begin{align*}
\text{Average Current Consumption (mA)} &= \frac{14.40\text{mA} \times 0.7\text{ms}}{32\text{ms}} + \\
&= \frac{10.00\text{mA} \times 0.3\text{ms}}{32\text{ms}} + \\
&= \frac{5.60\text{mA} \times 0.16\text{ms}}{32\text{ms}} + \\
&= \frac{0.4\text{mA} \times 30.84\text{ms}}{32\text{ms}} \\
&= 0.82225\text{mA}
\end{align*}
\]  

(4.45)

As shown in Calculation 4.45, average power consumption of the system is 0.82225 mA when hardware toolkit is sensing without actuating in 32 Hz. In Section 4.1.1.2.1, it is stated that 850 mAh Polymer
Lithium Ion battery is used in test and design applications. According to the average current consumption calculated above, under normal conditions, approximate battery life of hardware toolkit is calculated as follows.

\[
\text{Battery Capacity in mAh} = \frac{\text{Average Current Consumption}}{0.82225}
\]

\[
= 850 \text{ mAh} \\
= 1033.748 \text{ hours}
\]

\[
\approx 43 \text{ days}
\]

As calculated in Equation 4.46, under normal conditions and given specification in Section 3.3, technology developers can experience with hardware toolkit more than one month. However, it is important that in this case examined above, an actuator is not used. Actuators, mostly consume more power than sensing events. This is why, power consumption, thus battery life, may change if technology developer uses an actuator in an application or not.

### 4.4.2 Data Exchange Latency

In this section, latency between wireless connection points are measured and demonstrated. In the first section, latency between hardware toolkit and mobile phone is measured. In the second section, a test is done to measure the latency between mobile phone and web services.

#### 4.4.2.1 Hardware Toolkit - Mobile Phone

ANT wireless communication technology is a synchronous communication technology that payload can be transferred from master to slave in every channel period time. In Earl toolkit, channel period is set to 32 Hz. Therefore, every period there is a message transfer from master to one of the connected slaves and optionally in the same period there is a transfer between slave to master.

As depicted in Figure 4.25, in the following wave captured from oscilloscope, one of the LEDs on Earl hardware toolkit called TX/RX Led toggles whenever hardware toolkit receives a message from master. In this example, in order to demonstrate communication latency between mobile device and hardware
toolkit, there is one master and one slave device. In this case, mobile phone is master and Earl hardware toolkit is slave on a shared channel with 32 Hz channel period.

![Earl Hardware Toolkit Communication Latency](image)

**Figure 4.25: Earl Hardware Toolkit Communication Latency**

### 4.4.2.2 Mobile Phone - Web Services

Since data sent from mobile phone is not streamed to web services, latency highly dependent on internet connection quality. Therefore, web server may not receive the information in the same order as in mobile application. Information can go to the web services in different paths in the Internet. Therefore, latency may change without any control.

In order to measure the latency, a test mobile application software is designed. This test application sends information packets to web services every 32.25 millisecond which is channel period of Earl hardware toolkit and mobile phone. These packets are all labeled and timestamped on mobile phone. Whenever an information is sent to EWS, EWS returns an approval message. In this test procedure, time between this processes is measured. In this experiment, Sony Xperia Active mobile phone is used. It is connected to Internet through its WiFi communication. WiFi router to which the mobile phone is connected has 10 Mbps connection and is located in Nacka, Stockholm, Sweden. Earl Web Services web server has 100 Mbps internet connection and located in Kista, Stockholm, Sweden. The test is run for 1 hour and in this period data are posted from mobile phone to EWS. Based on this test, the latency between mobile phone and web server is calculated as follows. All time logs are stored in milliseconds in SD card on mobile phone. In the end, 115200 sensor information sent to server and 90127 (78.235%) of these sensor information successfully transmitted. The information that couldn’t be sent to server are failed because of various reasons such as timeout, bad HTTP request, busy server and unknown reasons. Based on the time logs of successful transmissions, mean latency is calculated using Matlab. Results are listed on Table 4.47. In Table 4.11 minimum, maximum, and mean latency values can be found. Furthermore, a plot is created of all information sent as shown in Figure 4.26.
Information Sent To Server In 1 Hour = \(32 \times \frac{\text{Information}}{\text{Second}} \times \frac{3600 \text{Second}}{\text{Hour}} = 115200 \frac{\text{Information}}{\text{Hour}} \) (4.47)

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min Latency</td>
<td>364 ms</td>
</tr>
<tr>
<td>Max Latency</td>
<td>1276 ms</td>
</tr>
<tr>
<td>Mean Latency</td>
<td>457.4 ms</td>
</tr>
<tr>
<td>Variance</td>
<td>25749.35 ((ms)^2)</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>160.46605 ms</td>
</tr>
<tr>
<td>Absolute Deviation</td>
<td>84.06105 ms</td>
</tr>
</tbody>
</table>

Table 4.11: Mobile Phone and Web Server Latency Test

As shown in the Table 4.11, mean latency of transferred information is 457.4 ms. The fastest transferred information is 364 ms where outlier of the latency for transferred information is 1276 ms. Mean value of the latencies is much close to minimum latency value which is obtained throughout the experiment. On the other hand, standard deviation, absolute deviation, and variance of latencies obtained from experiment shows that, latencies obtained from the experiment did not have close values. Moreover, 21.766 % of transferred information is failed, dropped or unsuccessfully transmitted. This is why, some of the information is lost and could not sent to EWS successfully.
Figure 4.26: Experiment Results for Communication Between Mobile Phone and EWS
Chapter 5

Conclusion and Future Work

In this thesis, we have introduced a low power wireless toolkit for sketching in hardware and software for early technology design stages. Technology developers can experience and get inspired by draft designs, discover innovative solutions and produce better technology by iterations. In order to create rapid proof of concept systems, developers get help from toolkits. This thesis describes a toolkit for technology developers that proposes low power consumption, portability, wireless communication, and ease of use.

As discussed in Background chapter, those toolkits that are already in the market have difficulties in terms of rapid development. In this thesis, it is visioned to design a toolkit that application designers can create proof of concept systems faster than implementing the system from scratch. Earl is a toolkit that technology developers can experience with most common analog sensors and actuators without engineering difficulties. Technology developers can experience different sensors and actuators by connecting peripherals to connectors on Earl hardware toolkit. Furthermore, they do not have to design a software to control connected sensors or actuators.

Earl toolkit exchanges information with Earl web services. Therefore, application developers can obtain information or control Earl hardware toolkits by web services without need of coding. Furthermore, Earl is a wireless portable toolkit. Therefore, application developers can exchange information between hardware toolkit and web services wirelessly. During the development process of Earl hardware toolkit, low power consumption is a desired requirement to increase the portability of hardware toolkit. Thus, application developers can use Earl for long time as a proof of concept systems.

Since power consumption directly effects the battery life of hardware toolkit, low power components and modules are used on hardware toolkit. According to activities during the application such as sensing, actuating, and exchanging information, power consumption of hardware toolkit may change. In Section 4.4, power consumption of Earl hardware toolkit is discussed. Especially, actuating consumes much higher power consumption than sensing and wireless information exchange. Furthermore, different information exchange period, speed, and information content size may cause different power consumption than given in Section 4.4.1. Likewise, different sensor information sampling rates may also cause unexpected power consumption. This is why, battery life should be taken into account according to power consumption of hardware toolkit. According to the test setup given in Section 4.4.1.6, technology developers can use Earl hardware toolkit more than a month without recharging the battery in sensing mode. Furthermore, by using higher battery capacities technology developers may have longer application experiences.

In Earl toolkit, hardware toolkits communicate with a mobile phone via ANT wireless communication technology because of the reasons explained in Section 3.4.1.4. One of the noticeable property of ANT communication is low power consumption. According to given specification on Table 3.2 power consumption is presented in Section 4.4.1. Under different conditions than Table 3.2, power consumption of ANT module, therefore hardware toolkit, may be different. Hardware toolkit battery capacity directly effects Earl hardware toolkit’s service time. This is why, larger batteries enable longer experiences with hardware toolkit. However, too big batteries may limit portability.

Earl toolkit is not designed for fast sensing and rapid actuating proof of concept systems. For example, some technology developers may want to sample data in much high sample rates. In this case, it is suggested to use other prototyping toolkits or do it yourself solutions. Earl toolkit is a system to create proof of concept applications under specifications defined in Table 3.2. Earl toolkit may disfunction under
constrains which are not fitting with the specifications.

This thesis covers design and implementation of first version of a hardware and software toolkit for early technology design processes. That means there are many possible ideas to add as new features and improvements on current toolkit system. In this section, these possible works are listed as below.

- **Using Lower Power Consuming ICs:** Earl hardware toolkit is a portable device. This is why, it is working with a battery and power consumption is significant parameter for battery life of hardware toolkit. In Earl hardware toolkit, currently a linear voltage regulator is used. However, as mentioned in Section 4.1.1.2.2, switching voltage regulator may decrease power consumption of the hardware toolkit.

- **Lower Power Consumption On Sleep Mode:** One improvement can be to decrease power consumption by putting all logic components on sleep mode in hardware toolkit. Currently, ANT wireless communication module does not deep sleep when module is not used. Switching off all unused components may help to decrease power consumption of hardware toolkit. Therefore, power consumption of hardware toolkit may decreased for longer battery life.

- **Different Motor Driver:** Another improvement to decrease toolkit cost can be changing haptic driver with a simple transistor. In order to fulfil the same requirements like haptic driver, two different transistor may be used for clock-wise and counter clock-wise rotation.

- **Better User Interface for Mobile Application and Web Services:** Currently, mobile application and web services are designed to demonstrate proof of concept of Earl toolkit. That means, mobile application and web services does not have a user friendly user interface. As a future work, a new user interface can help toolkit users better to control toolkit.

- **Data Streaming Instead of One by One Transfer:** In fast information exchange between mobile phone and web server, some of information get lost or time out. Instead of exchanging every packet of information one by one, socket streaming technologies may be used to transfer the information. Therefore, information will be transferred in an order and much faster.

- **More Mobile Phone Connectivity:** Only ANT enabled Android mobile phones can be used as an adapter to connect Earl hardware toolkits to Internet. Currently, only some of the mobile phones are ANT enabled. This is why, a small apparatus can be designed to convert ANT communication to another most common wireless communication technology such as BLE. Therefore, Earl hardware toolkits can exchange information with Earl web services with more Internet connected device via this converter.

- **Enriched Sensors and Actuators Can Be Used with Earl Toolkit:** Currently only most common analog sensors and actuators can be used with Earl hardware toolkit. It would be significant improvement for functionality of Earl toolkit to add more sensors and actuators. Therefore, digital sensors and actuators may enrich Earl toolkit. As a future work, an interface for digital sensors and actuators can be implemented.
Appendix

Power Consumption Calculation

Power consumption in a system is consumed electric energy in terms of Watts. In order to measure the power consumption following formulas can be used.

\[ \text{Voltage (Volts)} = \text{Resistance (Ohms)} \times \text{Current (Amps)} \]  \hfill (5.1)

\[ \text{Power Consumption (Watt)} = \text{Voltage (Volts)} \times \text{Current (Amps)} = \frac{V^2 (Volts^2)}{R \text{(ohm)}} \]  \hfill (5.2)

\[ \text{Bytes Per Second (} \frac{\text{Bytes}}{\text{Second}} \text{)} = \frac{\text{Number of Bytes in Payload} \times \frac{1 \text{ second}}{\text{Message Interval}} \times \text{Number of Channels}}{} \]  \hfill (5.3)

\[ \text{Bits Per Second (} \frac{\text{bits}}{\text{Second}} \text{)} = \text{Bytes Per Second} \times 8, \text{ where 1 Byte} = 8 \text{ Bits} \]  \hfill (5.4)

\[ \text{Power Consumption Per Bit} = \frac{\text{Power Consumption}}{\text{Bits Per Second}} \]  \hfill (5.5)

Power Consumption = \[ C \times V^2 \times f \], where \( C \) is Capacitance, \( V \) is Voltage and \( f \) is operational frequency \hfill (5.6)

Power Consumption in Joule = Power Consumption in Watt \times 3600\text{ second} \hfill (5.7)
References


[96] Precision Microdrives. Precision haptic 10mm linear resonant actuator - 3.6mm type, https://catalog.precisionmicrodrives.com/order-parts/product/100-10mm-linear-resonant-actuator-3-6mm-type accessed on, 05 2013.