

**COMMUNICATION BLADES: MODULAR COMMUNICATIONS FOR  
TANGIBLE AND EMBEDDED INTERFACES**

A Thesis

Submitted to the Graduate Faculty of the  
Louisiana State University and  
Agricultural and Mechanical College  
in partial fulfillments of the  
requirements for the degree of  
Master of Science in Electrical Engineering

In

The Department of Electrical and Computer Engineering

By

Karun K Kallakuri

Bachelor of Engineering (Electronics and Communications)

Madras University, India, 2002

December 2006

## **Acknowledgments**

I am thankful to all the people for their invaluable support and encouragement, which made this work possible.

I would like to specially thank

Dr.Brygg Ullmer, for introducing me to this research, for numerous hours of discussion, for the guidance and for the support.

Dr.Ramanujam, for his valuable suggestions, for all the timely help offered and for being a co-chair in my thesis committee.

Dr.Jerry Trahan, for being a very supportive committee member and proof reading my thesis.

My parents Mr.Prasad and Mrs.Vara Lakshmi and my brother Chaitanya, for pretty much everything in my life.

Srikanth, Rajesh, Cornelius and Blake my fellow students, for their help and time in reviewing my work.

Silpa, Hari, and Srikanth for patiently proof reading my thesis.

CCT and Dept. of Physics and Astronomy, for supporting my master's program financially.

And lastly, to all my friends in America and in India for the wonderful time we had.

# Table of Contents

<b>Acknowledgments</b> .....	<b>ii</b>
<b>List of Tables</b> .....	<b>v</b>
<b>List of Figures</b> .....	<b>vi</b>
<b>Abstract</b> .....	<b>viii</b>
<b>Chapter 1. Introduction</b> .....	<b>1</b>
1.1 Motivation.....	3
1.2 Thesis Statement.....	7
1.3 Thesis Overview.....	8
<b>Chapter 2. Related Research Areas</b> .....	<b>9</b>
2.1 Overview.....	9
2.2 Ubiquitous Computing .....	11
2.3 Tangible Interfaces.....	12
2.4 Wearable Computing .....	13
<b>Chapter 3. Related Hardware Platforms</b> .....	<b>16</b>
3.1 Berkeley Motes.....	19
3.1.1 Hardware.....	19
3.1.2 Software .....	20
3.1.3 Applications and Discussion .....	21
3.2 iStuff (Interactive Stuff).....	22
3.2.1 Hardware.....	22
3.2.2 Software .....	23
3.2.3 Applications and Discussion .....	23
3.3 Smart-its.....	24
3.3.1 TecO PArticle Smart-its.....	25
3.3.2 BTNode .....	27
3.3.3 Lancaster DIY Smart-its .....	28
3.3.4 Discussion.....	29
3.4 Phidgets.....	30
3.4.1 Hardware.....	31
3.4.2 Software .....	31
3.4.3 Applications and Discussion .....	32
3.5 Teleo .....	33
3.5.1 Hardware.....	33
3.5.2 Software .....	35
3.5.3 Applications and Discussion .....	35
3.6 Arduino.....	35
3.6.1 Hardware .....	36
3.6.2 Software .....	36

3.6.3 Applications and Discussion	36
<b>Chapter 4. Thesis Approach</b>	<b>40</b>
4.1 Bladed Tiles Toolkit	40
4.1.1 Hardware	40
4.1.2 Software	42
4.2 Communication Blades	43
4.2.1 Concept	43
4.2.1.1 Flexibility	43
4.2.1.2 Connectivity	45
4.3 Implementation	46
4.3.1 Architecture	47
4.3.1.1 Model	47
4.3.1.2 I2C	48
4.3.2 Hardware	50
4.3.2.1 Serial Blade	51
4.3.2.2 USB Blade	52
4.3.2.3 Bluetooth Blade	53
4.3.2.4 Gumstix Blade	55
4.3.2.5 Electrical Properties	58
4.4 Applications	58
4.4.1 eNote	58
4.4.2 Core Tiles	59
<b>Chapter 5. Discussion and Conclusion</b>	<b>61</b>
5.1 Discussion	61
5.1.1 Discussion about API	66
5.1.2 Integrated vs Modular Approach	68
5.1.3 Limitations	69
5.1.4 Technological Implications	70
5.1.5 Design Issues	71
5.2 Conclusion	71
<b>References</b>	<b>74</b>
<b>Appendix- A: Specifications of Microchip PIC 16F876A</b>	<b>78</b>
<b>Appendix- B: Specifications of Motes USB Module</b>	<b>79</b>
<b>Appendix- C: Specifications of BlueSMiRF Bluetooth Modem</b>	<b>81</b>
<b>Appendix- D: Specifications of Gumstix</b>	<b>82</b>
<b>Vita</b>	<b>83</b>

## List of Tables

3.1 Currently available Berkeley Motes modules.....	21
3.2 Comparison of different platforms.....	38
3.3 Characteristics of different platforms .....	39
4.1 Communication blades power ratings.....	45
4.2 Electrical characteristics of Communication Blades .....	57

## List of Figures

1.1 Marble Answering Machine, Concept sketch and Prototype.....	2
1.2 Different types of PCI cards .....	4
1.3 Usage scenario VS kinds of users.....	6
2.1 Illustration of related user interface fields (from [Svendson 2001, p.13]).....	10
2.2 Illustration of major user interface fields (from [Svendson, Ullmer]).....	10
2.3 Major trends in computing (from [5]).....	11
2.4 Illustration explaining tangible user interface; from [8].....	13
2.5 Steve Mann and his portable device (in 1998).....	14
3.1 MICA mote from Crossbow Technology.....	19
3.2 iStuff architecture diagram.....	22
3.3 Particle in <1cm <sup>3</sup> includes Battery, Sensors, CPU & RF communications.....	25
3.4 Sensor Board.....	26
3.5 BTnode Smart-its.....	27
3.6 Lancaster Mini Core, Add-on board with acceleration, light, temperature and touch sensor, Actuator Add-on board.....	29
3.7 Output and Input Phidgets.....	31
3.8 Multi I/O module board and a concept sketch.....	33
3.9 Modules in linear and multiple arm configurations .....	34
3.10 Arduino Board (left) and Arduino with a shield (Right) .....	36
4.1 Examples of function blades.....	41
4.2 Common pin description of a blade .....	41
4.3 Tiles with underlying function blades.....	42
4.4 Architecture of communication blades.....	47

4.5 Master and Slave devices connected to I2C bus.....	49
4.6 PCB layout of serial blade .....	52
4.7 PCB layout of USB blade and implemented USB blade .....	53
4.8 PCB layouts of Bluetooth blade and implemented Bluetooth communication blade.	55
4.9 PCB layout Gumstix communication blade and Gumstix .....	56
4.10 eNote individual components and when functional.....	59
4.11 Core tiles .....	60
5.1 Water Lamp.....	64
5.2 General API function calls.....	67
5.3 Abstracted and actual data communication routes.....	68

## **Abstract**

Bladed Tiles is a modular hardware toolkit for building tangible and embedded interface devices. It includes “function blades” and “interaction tiles,” which can provide a flexible, inexpensive, open-ended platform for constructing a wide variety of tangible and embedded interfaces.

In this paper, we propose Communication Blades. These are a class of electronic modules with varied computational capabilities for interfacing devices built using bladed tiles toolkit and also for interfacing embedded devices as adapters with external communication networks.

These blades provide flexibility by offering the ability to select between different communication technologies and connectivity by providing devices with interoperability over different communication mediums.

Furthermore, the modular blade architecture allows different types of communication blades to be plugged in on demand. This reduces the need for development and knowledge of communication protocols by the developers, thus abstracting the underlying complexity.

My research work includes studying and designing various communication blades i.e. Serial, USB, Bluetooth and Gumstix. It also includes prototyping, testing and implementing the communication blades.

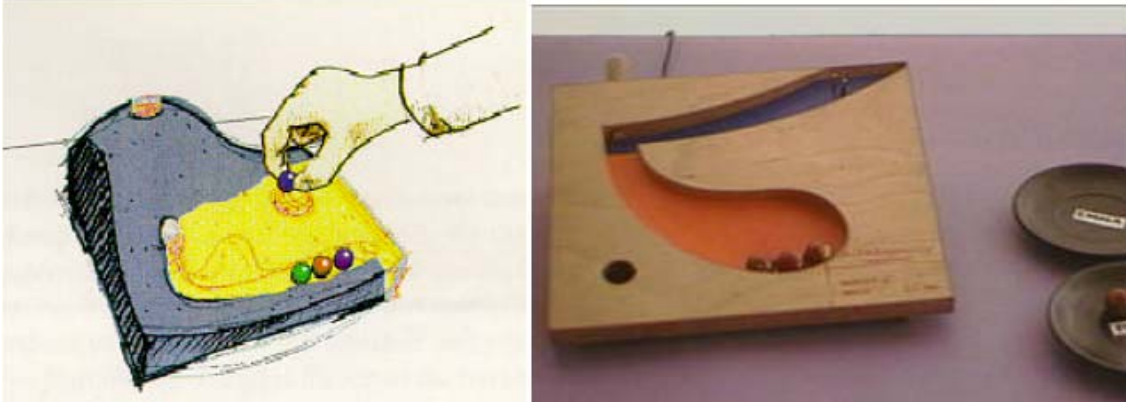
# Chapter 1

## Introduction

How might small, special purpose devices be used across diverse communication infrastructures? How can the transition from one infrastructure to the others be eased both for developers and potentially end users?

Numerous wired and wireless communication infrastructures exist, each differing in protocols, serving diverse purposes, and supported by certain types of devices. For example, USB connectivity is used where physical connection is acceptable, and Bluetooth connectivity is used for wireless connectivity. The special purpose devices referred here are portable, mobile, tangible and are used for Human Computer Interaction (HCI). HCI also includes various fields such as ubiquitous computing, tangible interfaces and wearable computing.

When using such special purpose devices, users might switch between different communication technologies over time according to evolving requirements. Consider Bishop's Marble Answering Machine [1] concept sketch which illustrates the use of physical data containers and controls for manipulating voice messages. The marbles are moved between active surfaces to replay marble contents, redial a marble message's caller, or store the message for future reference. It is shown in Figure [1.1]. Consider it has a POTS (plain old telephone service) wired connection, what if the user wants to have a USB for VoIP and WiFi or Bluetooth for wireless to move the answering machine around the house.



**Figure [1.1] Marble Answering Machine, Concept sketch and Prototype**

This project started when building such devices using a hardware/software technology, “Bladed Tiles”, which is currently under development in our group for implementing tangible interfaces. Support of transition for these devices from one communication network to other is required as users move across organizational borders and through heterogeneous communication infrastructures.

With many available and emerging communication technologies; and noting that these technologies are not entirely dependent on the device’s functionality, determining the appropriate kind of technology to a particular device is a difficult task. The choice also depends upon the available communication networks accessible to users over time. The answer has to be adaptable. When users build devices they should be allowed to add and remove different interconnection technologies.

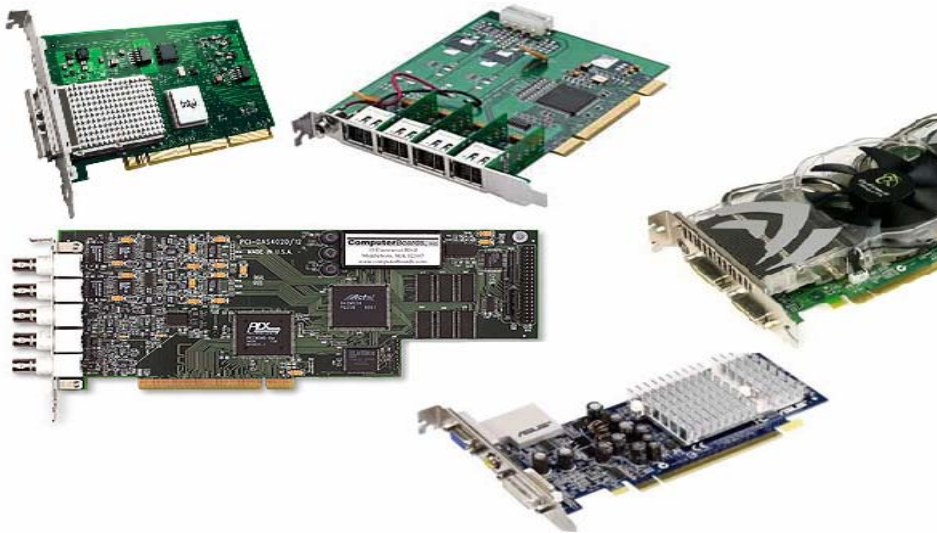
Throughout this project, the work has been influenced by the concept of plugging different communication technologies by the users with ease. This concept was raised through evolution of several generations of blades by Ullmer [4] and is implemented with his expertise in the current generation of blades. The implementation and advantages of these communication blades are discussed in detail within this thesis.

## 1.1 Motivation

About eighty percent of electronic products are discarded while they are still functional. This is not only because consumers desire products with newer features, but also because of the current marketing practice convincing consumers that buying the latest model is a must. Conceivably the most visible example of this practice is the discarded cell phones. Every year, they comprise an estimated 65,000 tons of waste as they are abandoned to obtain the latest models with higher mega pixels camera, more storage, music and video playback capabilities [36]. Flexible products are potentially enduring ones. A product that is adaptable and customizable is likely to be used for longer periods of time. Perhaps the best example of this practice is televisions where new devices such as the latest media players or sound systems can be added to old TVs, without actually discarding them.

Communication technologies are rapidly growing with increase in processing capabilities of embedded systems and innovations in fabrication technology. Each communication technology is different from others in its application and usage scenario. For example, RS232 is used for point to point data transmissions whereas Bluetooth is used for ad hoc wireless data transmission. These differences are obvious. Also, different applications require different communication technologies. The best way to facilitate various communication technologies is to employ modular architecture, where external communications are separated from other core operations, thereby, allowing communication types to be interchanged without disturbing internal operations. When new communication connectivity is desired it is simply added, thus extending the life of an older system.

Communication blade modules facilitate these various communication technologies to the developers on demand. However, for successful applications to emerge, setting up these communications should not require much expertise, and they should be easy to install. A proven example of this practice is the existing different types of PCI cards used in desktop computers. There are many types of cards currently available in the market such as network, graphics, DSP, video capture and so on each differing in their functions and purpose. These PCI cards can be installed with minimum guidance by the consumers with basic setup knowledge.



**Figure [1.2] Different types of PCI cards**

In a modular system, we do not have to predict exactly what functions every type of user might require. Every function has its own designated modules and new modules can always be added. Because the user is able to customize as desired, the applications chosen can be changed to meet an individual's evolving technological and physical needs. A modular system has many added advantages. The decentralized sensing of the system allows components to be added or removed at will, and ensures that if any

modules of the system break down, the object will remain functional. New modules can be exchanged for those that are no longer working, inhibiting the entire system from being discarded.

Modularity also assists the designers and engineers in focusing on what they are best at rather than worrying about implementing or learning about communication protocols. The designers can finish their end product by simply abstracting the communication media without learning the know how of each medium. They can add the required communications at the deployment stage to complete the device. Consequently, this potentially reduces the product complexity, development effort and time.

In modular architecture, it is increasingly possible to design devices that have the ability to be configured to meet the preferences of individual users. In software engineering, reusability is the likelihood that a segment of source code can be used again to add new functionalities with slight or no modification. Similarly, hardware modular components can be reused within the same architecture. Modularity allows the locus of product definition to be available to designers.





This is one of the main goals of the Bladed-Tiles toolkit. It is a modular hardware toolkit for building tangibles and interaction devices currently under development within our group. It comprises functional Blades and Interaction tiles, which can provide a flexible, inexpensive, open-ended platform for constructing a wide variety of tangible interfaces. More details about Bladed Tiles are presented in Chapter 4.

Users can easily customize blades to meet changing requirements. Using modular communication blades has the following advantages.

- Flexibility to choose from different communications based on various constraints.

- Reduce product complexity and development effort.
- Provide connectivity to different infrastructures and interoperability.
- Product reusability.
- Simplicity in usage.

This thesis work is specifically interested in providing users with various communication technologies for the devices developed using Bladed Tiles. Users referred here are primarily designers who develop tangible and embedded interfaces and end-users who use the products. Usage scenario with respect to kinds of users is given in Figure [1.3]. Two usage scenarios are mentioned here, firstly, ‘Industrial usage’ where applications are developed for mass production and distribution and secondly, ‘Development usage’ where applications are prototyped and implemented. Two kinds of users are mentioned here, the Designers who are the developers and the End-users who are the final users of the product.

usage kinds of users	Designers	End users
Industrial	 eg: companies like sony	 eg: Common man
Developing/ Protoyping	 eg: Researchers	 eg: Electronic Hobbyists

**Figure [1.3] Usage scenario VS kinds of users**

## 1.2 Thesis Statement

**Communication blades can facilitate communication infrastructure flexibility and connectivity for embedded and tangible interfaces.**

Here, we consider flexibility as the ability to choose from multiple types of communications, depending upon various constraints. We consider connectivity as seamless transition from one communication system to another with minor changes.

The embedded and tangible interface devices developed using Bladed Tiles are built for specific functionality and are for variety of purposes. Each might require different interconnection technologies depending upon their available constraints. As an example, some devices might need to have low power; wireless communication interfaces to gain more freedom of movement, while other devices might be adequate with a wired communications interface. Also, the communication technologies are rapidly changing over a period of time and new types are also emerging with advances in semi-conductor fabrication.

Consequently, the Bladed Tiles toolkit should not only provide the current communication technologies but also adapt to the emerging ones. Communication blades provide the above required flexibility of choice. They also provide interoperability with the current and new technologies without changing underlying hardware. These communication blades are plug-in modules that users can add a new communication blade or replace the older one with newer communication technology altogether.

These communication blades can also be used as interconnection adapters for interfacing I2C based embedded devices for prototyping and development. More about flexibility and connectivity are discussed in Section 4.

### **1.3 Thesis Overview**

Chapter 2 considers relevant research areas for this thesis. The chapter provides various fields in Human Computer Interaction based on Svendsen illustration [2]. This chapter also briefly describes the included research areas which includes ubiquitous computing, tangible interaction and wearable computing with a few examples within the context.

Chapter 3 presents related hardware platforms that are helpful in building special purpose devices. Detailed review of each platform includes their hardware, software and applications. This is followed by a discussion of each platform based on its characteristics.

Chapter 4 presents the thesis approach, which includes detailed explanation of the modular Bladed Tiles toolkit. It is followed by the motivation behind this work and the concept of communication blades. Implementation of communication blades is presented in detail along with schematics and PCB layouts. Finally, the chapter ends with two implemented application devices, eNote and DataTiles.

Chapter 5 presents discussions about API, integrated vs. modular communications and future work along with two future scenarios and also conclusion of this work in brief.

## Chapter 2

### Related Research Areas

#### 2.1 Overview

Human-computer interaction (HCI) is defined as

**Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them – ACM [2].**

This thesis has developed within the context of designing human-computer interaction systems that are driven by embedded technologies which support construction of various kinds of input, output and interaction devices. This chapter briefly provides context for related subdisciplines of human-computer interaction, including ubiquitous computing, tangible interfaces and wearable computing. The chapter begins with the illustration of various fields in HCI by Svendsen [3] and continues with brief introduction of the related research fields.

Svendsen, in his Danish language thesis, presents several illustrations of the relationships and interplay between the above mentioned research fields. The Svendsen figure provides perspectives of various subgroups in HCI.

This thesis is limited to the usage of communication blades, primarily within the context of the Bladed Tiles toolkit<sup>1</sup>. The Bladed Tiles toolkit will be predominantly used in developing devices for the following research areas:

---

<sup>1</sup> The toolkit is not limited to these research areas. The author chooses to mention most likely fields for blades depending upon their usage, to maintain the brevity of this thesis.



## 2.2 Ubiquitous Computing

Ubiquitous computing is the method of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user - Weiser [39].

Ubiquitous computing emerged over a decade ago, and has recently gained momentum with the development and diffusion of wireless networks and mobile services. It is often presented as a third wave of computing, a departure from its predecessors - the mainframe and personal computers. Figure [2.3] illustrates major trends in computing over the last sixty years. The mainframe can be represented as the era of "many people, one computer"; the PC, "one person, one computer"; and ubiquitous computing, "one person, many computers".

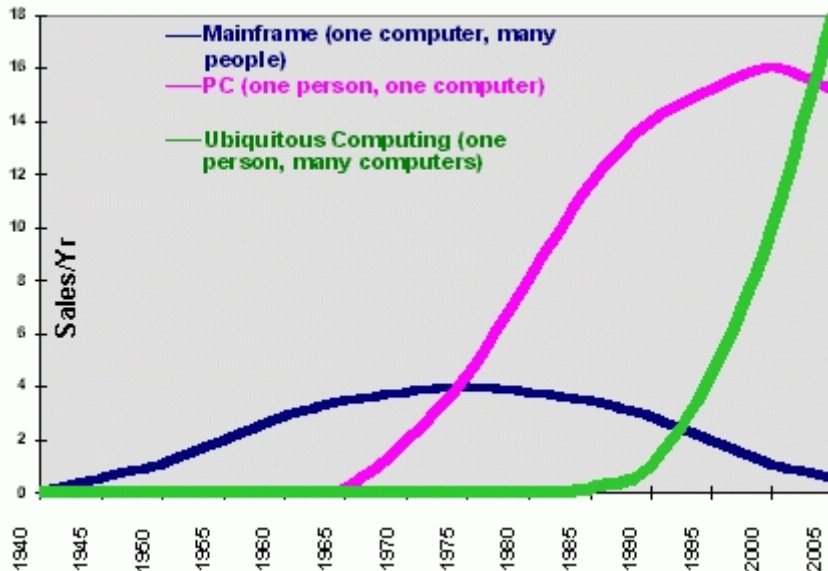


Figure [2.3] Major trends in computing (from [5])

Ubiquitous computing is an emerging branch of computing where networked devices are seamlessly embedded in the background of the user environment to serve preconfigured as well as emergent purposes. These devices are designed to blend into people's physical surroundings, and engineered to support work practices and routine activities within and across boundaries. From a technical perspective, ubiquitous computing refers to a coordinated array of task-oriented computing devices that operate semi-independently in net-centric environments enabled by wireless and mobile technologies. This new breed of computing is based on architectures that are not tied to personal devices but instead moves into the fabric of life. In recent years the terms "pervasive computing" and "things that think" are being used, to describe similar concepts [6] [7].

### **2.3 Tangible Interfaces**

**Tangible Bits allows users to "grasp & manipulate" bits in the center of users' attention by coupling the bits with everyday physical objects and architectural surfaces - Ishii and Ullmer [8]**

Tangible or graspable interfaces refer to interfaces that use physical objects to manipulate the digital world. The vision is that they provide a more natural way of interacting with the digital world as they can be more integrated into the physical environment of the user than traditional interfaces. For example, the vision of "tangible bits" by Ishii and Ullmer explores interactive surfaces and physical objects to bridge the gap between the physical and virtual world [8]. It is also described as 'giving physical form to digital information' by Ullmer [10].



**Figure [2.4] Illustration explaining tangible user interface; from [8]**

Examples of the rapidly emerging tangible interfaces field include ToolStone, mediaBlocks, etc. The ToolStone is a tangible interface in form of a wireless, cube-shaped object with multiple degrees of freedom. It has to be used together with a table to act as an interface to manipulate the digital world [9].

mediaBlocks develops a related approach for introducing new objects to the physical world. mediaBlocks are rectangular physical blocks that are used for transporting digital content obtained from a media input device, such as a digital whiteboard or a video camera. Applications include the transportation of data between input and output devices. Examples would include a display, a printer or to new kinds of specialized devices such as a media sequencer for generating new content.

This field of research encourages interaction with objects of our physical world that have been augmented with digital information and act as manipulable interfaces beyond the traditional graphical user interface, keyboard, mouse and rectangular screen.

## **2.4 Wearable Computing**

**Wearable computing facilitates a new form of human-computer interaction comprising a small body-worn computer (e.g. user-**

**programmable device) that is always on and always ready and accessible. In this regard, the new computational framework differs from that of hand held devices, laptop computers and personal digital assistants (PDAs). The ``always ready'' capability leads to a new form of synergy between human and computer, characterized by long-term adaptation through constancy of user-interface - Steve Mann[11]**

Wearable computers are fully functional, self-powered, self-contained computers that allow the user to access information anywhere, at any time and have both operational and interactional constancy. Notably, these are devices that are always with the user, and can interpret user commands and execute the same commands, while the user can do so walking around or doing other activities. Figure [2.5] shows Steve Mann with a bulky, cumbersome, unconcealed system, yet a very functional and capable personal computer.



**Figure [2.5] Steve Mann and his portable device (in 1998)**

Today, designers and engineers more often want their technology to assimilate with current fashion trends. Users generally do not want to lug around intrusive devices

with visible wiring that distract from form and comfort. They prefer wearables that emphasize flexibility, ease, comfort, and other attributes that people expect from their daily attire [12]. For this reason, companies like Technology Enabled Clothing, Eleksen, *et al.*, conceal electronics within fabrics [13] [14].

## Chapter 3

### Related Hardware Platforms

This section briefly describes several different hardware platforms that can be considered as enabling technologies for prototyping devices considered in this thesis. Here, the term hardware platform is used for a family of embedded hardware devices and includes software such as operating systems and application programming interfaces (APIs). The following hardware platforms are discussed in detail, where each platform is targeted to a specific field.

- The Berkeley Motes design is based on the Smart Dust vision and self-monitoring networks. These Motes are tiny, self-contained, battery-powered computers with radio links, which enable them to communicate and exchange data with one another, and to self-organize into ad hoc networks.
- iStuff are intended to simplify the novel interaction techniques of multiple users, systems and devices. iStuff devices are small, untethered, and remotely accessible through the room software infrastructure (the iROS). They can be combined on-the-fly to build or extend physical post-desktop user interfaces.
- SmartIts are developed to address deployment issues and do-it-yourself custom interactive devices. SmartIts are embedded device modules integrating a microprocessor and wireless networking that can be customized with add-on boards for physical interaction.
- Phidgets are designed for rapid prototyping of plug and play USB custom devices. Phidgets are an easy to use set of building blocks for low cost sensing and control from a PC. Using the Universal Serial Bus (USB) as the basis for all Phidgets, the

complexity is managed behind a simple and robust Application Programming Interface (API).

- Teleo is developed for scalability and for building digital and complex devices. It consists of a line of modular and networkable hardware components that can easily be connected to a computer via USB. Components range from a variety of input and output modules, motor controller modules and accessories.
- Arduino is developed to implement Processing and Wiring languages [34]. Arduino can be used to develop stand-alone interactive objects or can be connected to software running on a computer.

For each platform the description includes a summary of their hardware properties the development process on each platform and major applications based on them. Discussions are also made based on the following three distinguishing characteristics.

#### **a) Distribution of Intelligence**

Here intelligence refers to the processing capabilities of the individual modules which depend upon the CPU power, memory, storage capacity and so on. Some modules are sufficiently capable of doing the required low level functions such as reading data from a single or few sensors, while other modules are capable of having their own operating system that can control other modules. For example, RTOS operating system can be run in PIC microcontrollers, in future fully functional Linux operating systems might be able to run on microcontrollers as small as PIC's. These high level modules can be dispersed over the network for distributing the control. These can also work stand alone without any control from external systems. There are several advantages of

distributed systems over a centralized system. One of the main advantages is scalability of the system with out disturbing the existing system.

### **b) Communications Integration**

Rapid developments in various technologies lead to integration of communications onto systems. This integration is sometimes necessary and offers advantages for particular devices. For example, consider a mobile phone that has integrated GSM/GPRS/EDGE communication technologies or a laptop with integrated Ethernet, WiFi, USB and Bluetooth. Even though more communication technologies are being invented over time and need to be updated in these devices, this integration is a must in a mobile phone to stick to its constraints such as form factor, power consumption, weight, etc. But for some devices such as tangible devices, a modular approach in communication has added advantages as discussed in Section 4.

### **c) Flexibility in Switching Between Different Communications Infrastructures**

This refers to the various available communication technologies in each platform. Some platforms might support more than one communication technology as an interface to the external communications while other platforms stick to a single communication technology. The platforms having only one type of communication are obviously not flexible in switching among various types of communication technologies. We also discuss about how easily users can change these technologies to migrate from one infrastructure to the other. Are there any additional steps in changing them or is the platform API clever enough to recognize and accept the change without breaking the regular operation?

### 3.1 Berkeley Motes

Berkeley motes are a commercially available, open embedded hardware platform designed on account of wireless sensing networks. Berkeley motes are based on the Smart Dust vision that seeks to pack autonomous sensing, computing and communication system into the size of a cubic-millimeter mote [15]. Several prototypes on millimeter level have been built. Additionally, inch-sized prototypes have been built that are also commercially available from Crossbow Technology, Inc. [16].

#### 3.1.1 Hardware

The hardware platform consists of two modules.

Firstly, the processor and radio platform (MPR) and secondly, the mote sensor and data acquisition boards (MTS). The MTS can be stacked on top of the MPR. The latest generation of the MPR includes the MICA2 and MICA2DOT mote. Figure [3.1]



**Figure [3.1] MICA mote from Crossbow Technology**

shows a second generation MICA mote. All the MICA motes are based on an Atmel ATmega 128L microprocessor offering 128Kbytes of Flash program memory, 4Kbytes SRAM and 4Kbytes EEPROM running at 4 MHz. All MICA motes use a multi-channel radio transceiver operating on 868/916 MHz or 433 MHz for wireless communication. The MICA2DOT (25mm x 6 mm) mote is a smaller, circular shaped version of the MICA2 mote (58mm x 32 mm x 7 mm). It has a reduced set of input/output channels and an on-board temperature sensor. The MICA2 motes can be plugged into a Mote Interface Board (MIB) providing serial and parallel connectivity to the sensor network, e.g. in form of a web gateway which is available from Crossbow Technology, Inc.

Three versions of the MTS sensor boards are available with a combination of various sensors. These sensors include light, temperature, acoustic, 2-axis accelerometer and 2-axis magnetometer sensors. Two of the boards include a sounder and one includes a wired breadboard area for prototyping. An additional prototype board for the MICA2DOT mote is also available. Apart from the sounder and three LEDs on the MPR, no actuators are available.

### **3.1.2 Software**

Application development on the motes is based on the TinyOS [17] operating system. TinyOS is an event driven operating system supporting the requirement of concurrent applications in sensor networks. TinyOS uses a component based programming model. Each component defines an upper interface and a lower interface. The upper interface defines synchronous commands implemented by the component and asynchronous events signaled to other components. The lower interface defines the commands it uses and the events it handles.

Development support on the Berkeley motes also includes TinyDB implementing a SQL-like query language for sensor networks [18]. Queries can be directed to the network as a whole. The individual sensor nodes cooperate in order to deliver the results of the query back to the source. The largest deployment of Berkeley motes included 800 nodes in a demonstration given to the audience of the Intel Developer Forum Conference [19]. However, the largest continuous setup of Berkeley motes includes 32 motes that have been deployed on an island for habitat monitoring of petrels. Further applications include the usage of motes for robotic applications [20].

### 3.1.3 Applications and Discussion

Given the commercial availability of the hardware and the support by the large developer community for the open source operating system TinyOS and the query language TinyDB, the Berkeley motes are a promising candidate for an enabling technology for dispersed Systems and Sensor Networks. The operating system itself focuses on dealing with the resource constraints inherent to the motes. Further abstractions that are based on the TinyOS provide solutions for a specific application – in the case of TinyDB the application domain is that of large scale sensor networks.

Intelligence can be distributed over a network with the help of the modular nature of the platform. Communication with an external system is handled by a gateway mote. Transition between different communication architectures can be made by changing different gateway motes. Changing different gateway Motes does not require any additional software steps and it is automatic. Motes can be reused for various applications as they are physically isolated and can be reprogrammed for any number of times.

**Table [3.1]: Currently available Berkeley Motes modules**

Different Modules currently available in Berkely Motes			
Processor/Radio Modules	Data Acquisition Modules	Sensor Modules	Gateway Modules
MPR4x0 - MICA2 Series	MDA100	MTS300/310 - Multi-Sensor	MIB600 - Ethernet
MPR2400 - MICAz ZigBee	MDA300	MTS400/420 - Environmental/GPS Sensor	MIB510 - Serial
MPR600 - MICA2 OEM	MDA320	MTS510 - Multi-Sensor	MIB520 - USB
MPR2600 - MICAz OEM	MDA500		SPB400 - Stargate
MPR5x0 - MICA2DOT			
MCS410 - MCS Cricket			

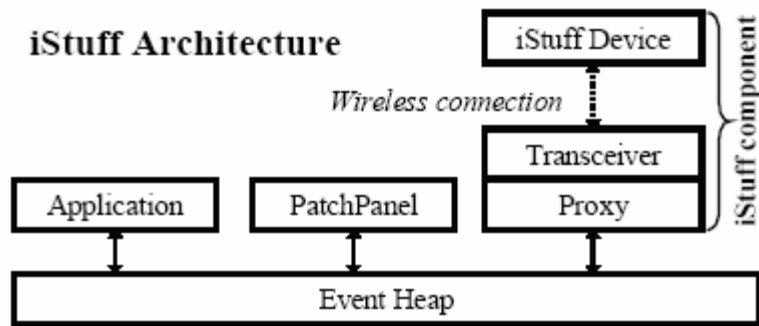
However, TinyOS does not support wired communications such as I2C, USB for communicating within the devices. Table [1] shows the currently available Berkeley Motes. They are developed for deploying wireless sensor networks off the shelf.

### 3.2 iStuff (Interactive Stuff)

iStuff [21] is a toolkit comprising physical devices and flexible software infrastructure (iROS). It was designed to simplify the exploration of interaction techniques of multiple users, devices, systems and applications collaborating in an interactive environment. To simplify the mapping of devices to applications, the supporting software framework provides a dynamically configurable intermediary agent.

#### 3.2.1 Hardware

iStuff architecture shown in Figure [3.2] is divided into three modules. iStuff components consist of wireless devices along with a machine connected to the



**Figure [3.2]: iStuff architecture diagram**

Event Heap that has a transceiver and related software. The device and proxy are necessary for an iStuff component, although multiple iStuff devices can share a single proxy. This design separates most of the devices allowing them to be very simple and light weight. All that is required for a physical device to become an iStuff component is a proxy that encapsulates data into an event, making it independent of any technology or protocol. Examples include iButton, iSwitch, wireless mouse, etc.

### **3.2.2 Software**

Event Communication is handled as supported by the Stanford Interactive Room Operating System (iROS) [22] infrastructure. In this architecture, producers post events to the Event Heap and consumers register to receive desired events, specifying the event type and optionally other criteria based on the requirements.

It is designed specifically to be robust against failure and to support easy restarting of different parts of the system. iROS is implemented primarily in Java and is available in open source distribution. Lately, Patch Panel was introduced as an abstraction layer for flexibility and reusability of events. Patch Panel provides dynamic manipulation of events. It consists of an intermediary application that implements event mapping and one or more GUIs that provide a user accessible way to configure events.

### **3.2.3 Applications and Discussion**

A broad variety of applications have been designed using iStuff. The majority of them were developed as a part of iRoom (Interactive Room project in Stanford University). iWall is implemented using iStuff, which is a distributed whiteboard application that allows multiple people using different cursors to interact with different images and various other graphical objects on multiple machines and displays. It supports multiple users and cursors by associating each cursor with a unique ID. Users may use any input device like mouse, iWand, or a prototype device that includes two knobs.

iStuff gains its power from its usage along with the iROS, which creates a flexible and robust software framework. This allows custom and legacy applications to communicate with each other and with user interface devices in a dynamically configurable way. Even though iStuff follows a modular architecture, its intelligence is

concentrated in the host system containing iROS software. Any physical device can be integrated with iStuff if it can communicate with event heap, irrespective of the interface. It is more of a software toolkit for having interoperability among various devices than a hardware toolkit to prototype devices. Also, it does not aid in changing various communication technologies of a device, if devices need to be used outside iROS.

### **3.3 Smart-its**

Smart-its are a family of hardware platforms specifically designed for the post-hoc augmentation of everyday objects. Smart-its are designed for unobtrusive augmentation of objects with sensing, computing and wireless communication capabilities.

The objectives of the Smart-its project are to investigate perceptual and collective computing methods for Smart Objects, to develop software architecture for collective context-awareness and to explore novel applications based on Smart Objects [23]. Three hardware platforms have been built, each with different optimization goals.

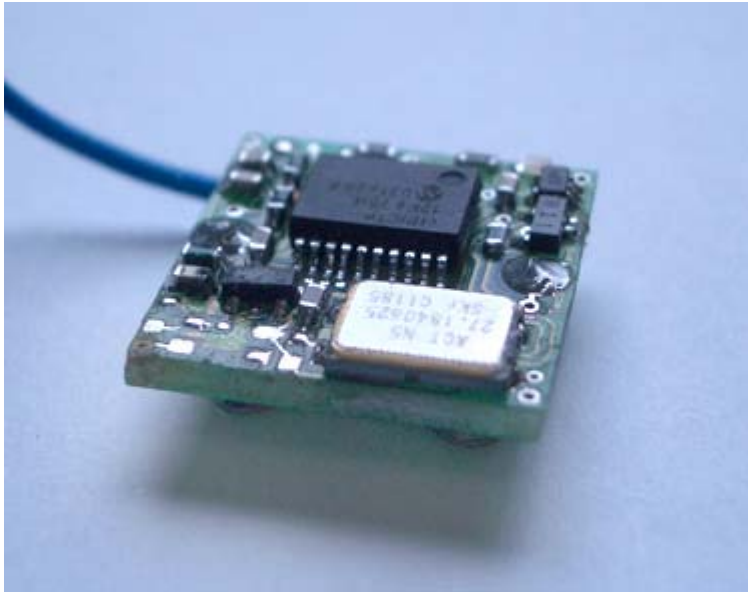
- The TecO platform was designed for size, power consumption and reliable broadcast communication.
- The BTnodes have been optimized for interoperability with existing wireless consumer devices using Bluetooth for communication.
- Finally, the Lancaster DIY prototyping platform has been optimized for simplicity in assembly and extensibility.

All the above three platforms will be discussed in this chapter briefly which includes hardware, software and their applications.

### 3.3.1 TecO Particle Smart-its

Similar to the aforementioned hardware platforms, TecO Smart-its follows a modular design separating the RF communication from sensing and processing. The core board is based on a PIC12F675 measuring  $<1\text{cm}^3$  in the latest version (Figure [3.3]).

For communication a transceiver operating on 868MHz is used. Two LEDs provide simple debugging. There is also a ball switch for waking-up the board in stand



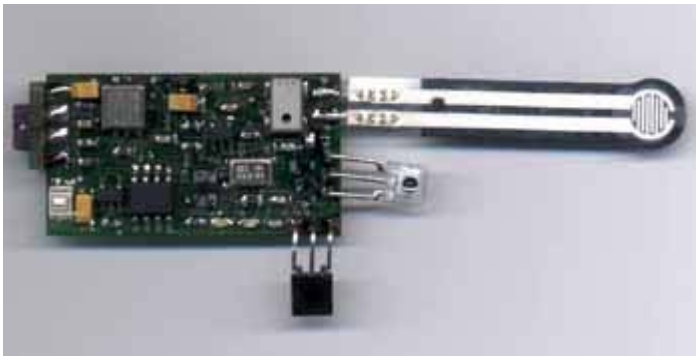
**Figure [3.3] Particle in  $<1\text{cm}^3$  includes battery, Sensors, CPU & Memory, RF communication**

alone applications. The core board has two connectors, one for connecting a power source providing voltages between 1V and 3.3V and stack connector for connecting the sensor board. An earlier version of the core board measured 46x42 mm and was produced in larger quantities making it available to the project partners. Instead of a stack connector, a molex connector was used as a connecting extension like the sensor boards via cable. The sensor board is based on a PIC12F675.

Sensors include a pressure sensor, a two-dimensional acceleration sensor, light sensors, a temperature sensor, a magnetic field sensor and a microphone. Actuators include LEDs and a buzzer. A power connector for input voltages between 1V and 5V

and connectors for extensions like additional sensor boards and connection to the core board are provided as well.

Figure [3.4] shows the latest version of the TecO sensor board with onboard force and pressure sensors, acceleration sensor which can be used for waking up the microprocessor based on simple movement detection. It also includes switchable daylight and IR light sensor, temperature sensor and a microphone. However, several versions of the sensor boards are available each implementing a different set of sensors. Further information about the TecO Smart-its can be found in [24].



**Figure [3.4] Sensor Board**

A wide range of various applications have been developed for the TecO Smart-its. One of the first applications was Smart-its friends, an interactive technique for linking personal objects based on similar context [25]. The technique has been implemented on the TecO Smart-its platform for demonstrating an application of the technique for linking Smart-its augmented objects by shaking them together. Both Smart-its compare the trajectory recorded by their acceleration sensors. Application scenarios for this technique include linking a credit card to a wallet so that it could only be used in close proximity of the wallet.

### 3.3.2 BTNode

The BTNode smart-its have been designed at ETH Zurich as an alternative smart-its augmentation platform for exploring interoperability with existing consumer Bluetooth devices. The ATmega 128L based BTNode measures approximately 58x33 mm (Figure [3.5]). The Zeevo ZV4002, supporting AFH/SFH, Bluetooth module implements the lower layers of the Bluetooth stack offering its functionality to the ATmega 128L via the host controller interface (HCI) through a serial port. Several connectors provide different communication interfaces splitting the core board from other components such as sensor, actuator or an alternative radio board.

The BTnodes support two programming models requiring different software. The sequential model allows tight control of all resources (wireless and wired communication, real-time clock, digital and analogue I/O), but leaves more tasks to the application developer. The event-driven model with



**Figure [3.5]: BTnode Smart-its**

cooperative multitasking provides management of the resources and is more suitable for sensing applications. However, it is more restrictive in accessing the resources.

BTnodes are mainly used for research in small-scale, Bluetooth-based mobile ad hoc networks (MANET), and for ubiquitous computing applications using BTnodes for tagging objects. BTnodes have also been used in sample applications demonstrating how users can interact with Smart Objects by the help of a Bluetooth enabled mobile phone,

BTnodes and RFID tags [26]. Three sample applications demonstrated explicit and implicit interaction initiated by users, by Smart Objects or both. For example, an egg carton was tagged with a BTnode and an attached sensor board in order to monitor its condition. Alerts were forwarded to a contact person's Bluetooth mobile phone, when the product has been damaged. The detection of a contact person's phone was implemented by detecting the proximity to the product using RFID tags.

### **3.3.3 Lancaster DIY Smart-its**

The Lancaster Smart-its [27] have been designed for simplicity in assembly and extensibility as an experimental platform for interactive applications. Its modular design is based on a core board that provides processing and communication capabilities. These core boards can be extended with add-on boards. The core boards are available in two size variations. The standard core boards measures approximately 70mm by 53mm. It provides an RS232 connector that can be used for debugging or connecting the Smart-its to other devices such as a PC. It can be powered by AAA batteries, rechargeable or an external power supply. The mini Smart-its measures 44mm by 44mm and is powered by a 3V lithium coin cell. This version sacrifices the RS232 connector in favor of a smaller form factor more suitable for mobile applications. Add-on boards extend the core boards with sensors, actuators or additional computing power. A wide range of existing add-on boards is available including the following,

- Generic Sensor Board. Features a passive infrared sensor, temperature sensor, 3D accelerometer, touch sensor, two light sensors and two LEDs
- Load Sensing Board. Interfaces industrial load cells
- Actuator: A generic actuator board, e.g. for switching halogen lights

- RFID reader. A RFID reader for the DIY Smart-its
- Wearable 3D Accelerometer Network, an add-on board that interfaces wearable 3D accelerometer boards.



**Figure [3.6]: Lancaster Mini Core, Add-on board with acceleration, light, temperature and touch sensor, Actuator Add-on board**

As Lancaster Smart-its are not commercially available, application development involves both, assembling hardware and writing software. However, they were specifically designed for easy assembly requiring little to no knowledge in electrical engineering. All necessary information for producing and assembling the Smart-its is available online [27]. Among these applications was a smart ball, able to report when it is thrown or caught, a wireless gesture remote control for MP3 players, and a RFID player. For some of the applications, even additional hardware design and implementation was realized in short time by the use of hardware libraries requiring little understanding of the circuits used.

### 3.3.4 Discussion

Smart-its platform was introduced as a valuable tool for researchers in need of a hardware platform for wireless sensing applications. The DIY Smart-its platform has been

presented as a tool for rapid prototyping. The Particle Smart-its have been introduced as a platform to address deployment issues. The BTnodes were introduced as an alternative Smart-its augmentation platform for exploring the interoperability with existing consumer Bluetooth devices.

All three types of particles are modular in architecture. All types of particles can be reused as they are plug-in modules. The ability of distributing intelligence over the system is possible as processor particles are capable of working stand alone. Also, each has different communication gateways tailored for specific communication interfaces. The transition from one communication infrastructure to another can be done by using the required Smart-its platforms. The transition can be done simply by exchanging the corresponding modules and it is automatic. Communication modules are integrated into processor modules. If the communication interface needed to be changed, then the entire processor module has to be changed, which might require re-programming of the module. Communication blades can be integrated into Smart-its system readily as Smart-its use I2C protocol for internal communications, which is also used by the communication blades. This integration might provide more flexibility and adaptability to the smart-its systems.

### **3.4 Phidgets**

Phidgets are a prototyping platform for interactive applications. Phidgets use the concepts of physical widgets to provide hardware and software building blocks for physical input and output [28].

### 3.4.1 Hardware

The actual hardware consists of several kits that are connected via USB to a PC. Output Phidgets include servos, LEDs, and software-controllable power bars. Input Phidgets include light, force, heat and capacitive sensors, various buttons and sliders and an RFID reader (Figure [3.7]). Phidgets were designed to be a customizable and easy to use hardware platform targeted at application designers not familiar with hardware design. As a consequence Phidgets are simplest hardware platform presented in this section, omitting example features like wireless communication or size and power optimizations. Furthermore, they were not designed for technical extensibility as the rest of the hardware platforms presented here. The idea is rather to change the physical appearance around a basic set of hardware, e.g., for building ambient displays, than the hardware platform itself.

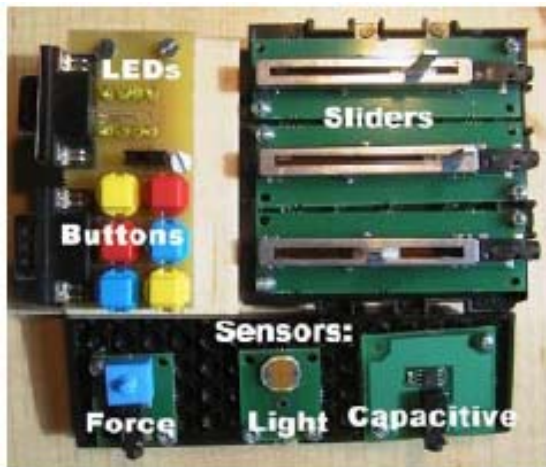


Figure [3.7] Output and Input Phidgets

### 3.4.2 Software

Phidgets are tightly coupled with a software toolkit that provides an API to the actuators or sensors of a phidget. Furthermore, programmer objects called *widget taps* have been

implemented. They expose an application's functionality as controlled by its graphical user interface (GUI). This allows writing small software programs that allow using phidgets as physical input or output to existing applications such as an email client without changing the application code [29]. Phidget hardware and software is commercially available at [30].

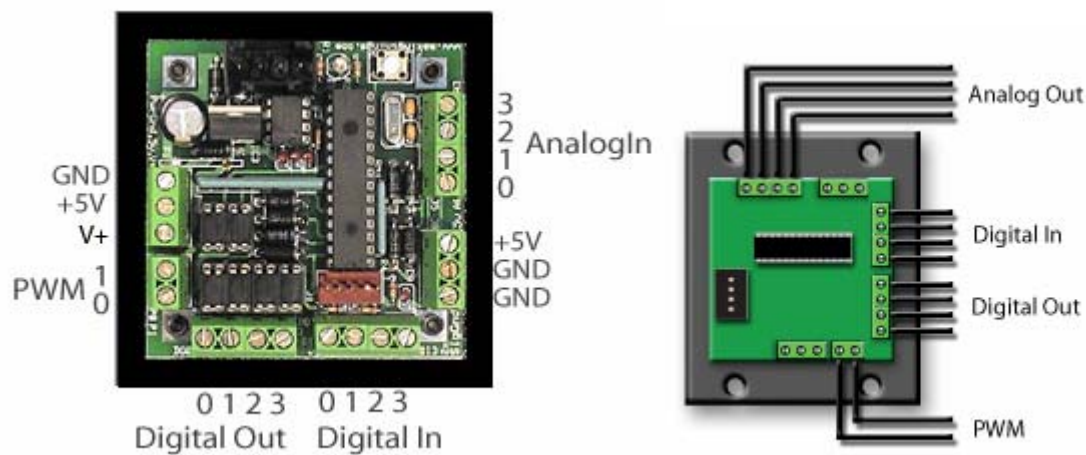
### **3.4.3 Applications and Discussion**

The software support provides good abstractions from the hardware and consequently easy application development. However, there is a lack of an overall framework providing useful guidelines and concepts for application development. As a consequence most of the implemented applications have the character of interesting toy applications such as the Buddy Bug, a physical interface for Windows Instant Messenger [31].

Phidgets are among the most simple to use hardware platforms of those presented. Thus, they provide an interesting hardware platform for simple interfaces, especially as they were designed for physical interaction. Wireline communication, relatively larger size and non-programmability of the Phidgets make them less suitable for building a chain of devices as a basis for dispersed systems. External communication is integrated onto each module. Also, they use only USB connection for communication with the external host system and changing to another communication technology is not possible with the current available phidgets. However, if phidgets can provide various communication interfacing technologies, they can be very flexible for building larger systems.

### 3.5 Teleo

Teleo [32] is a modular set of components used for building kinetic and interactive systems, devices or machines. It consists of a line of modular hardware components that can easily be connected to a computer via USB. These modular components can also be networked together to form a complex sensing and interactive platforms. Components can be programmed and controlled using any of the provided programming languages. These modules range from a variety of input and output modules, motor controller modules and accessories.

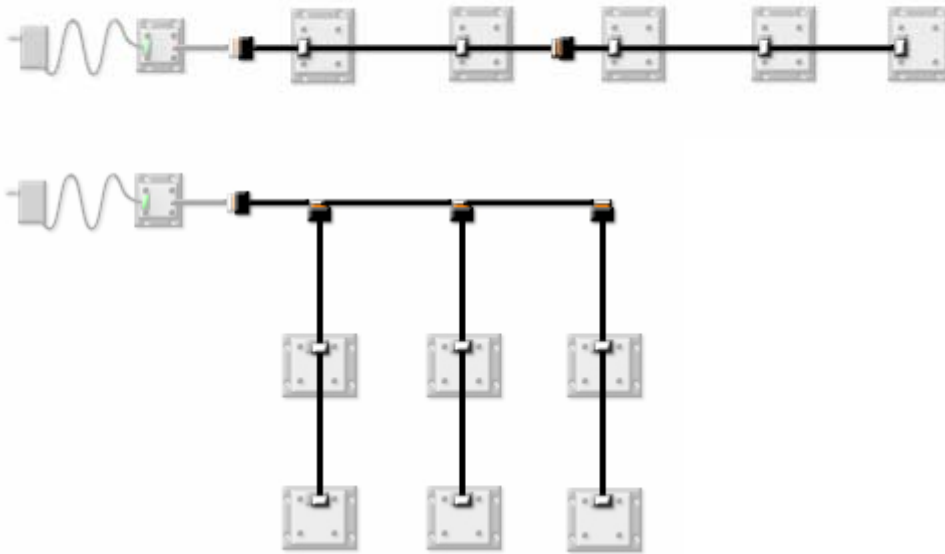


**Figure [3.8] Multi I/O module board and a concept sketch**

#### 3.5.1 Hardware

Teleo components are classified as modules and blocks. The heart of a Teleo system is its modules. Each module has approximately the same design. Modules measure around 2.5” x 2.5” in dimension. Electronically, the main feature of the board is the microchip 18F series microprocessor, which performs and controls all the computational work of the board, running at 40MHz. It also includes a power regulator for maintaining constant 5v

supply required by the processor and an RS-485 driver chip to permit communication on the network [33]. Different modules are available for different applications, for example a Multi IO Module is a multi-purpose module designed to provide a Teleo network with analog inputs, digital outputs and PWM outputs. Modules can form a network of devices as shown in Figure [3.9].



**Figure [3.9] Modules in linear and multiple arm configurations**

Blocks are a family of compatible electronic building blocks that provide an easier and cheaper way to build interactive devices without having much electrical knowledge. There are several different kinds of blocks available; some blocks have preprogrammed functions to process the information from inputs and outputs. All blocks rely on simple screw terminals to make wire connections with the outside world. This enables users to buy and use their own sensors and actuators. Different kinds of blocks include Poly Block, Servo Block, Stepper Block, Line Block, etc.

### **3.5.2 Software**

Teleo's have good support in different languages including C++, Macromedia's Flash and more. Their support and abstraction of programming languages and hardware allows end users to rapidly prototype and need not be experts or low-level programmers. Teleo's wide range of customers is also testament to this fact.

### **3.5.3 Applications and Discussion**

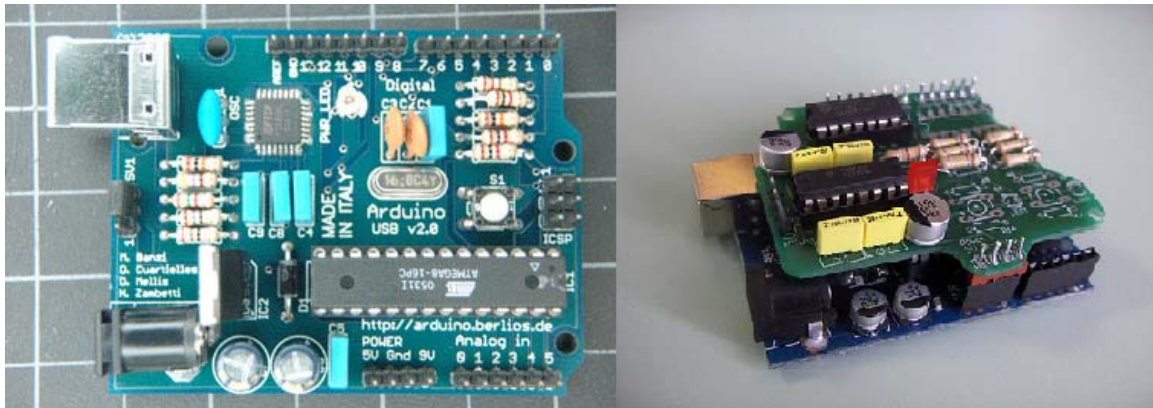
Teleo is flexible, expandable and relatively inexpensive. Most of the applications implemented are at the University level where it is used as an ideal teaching and learning tool [34]. Intelligence can be distributed over the network because of their modular architecture and processing power of the I/O boards which can work stand alone. However, their communication is restricted to only one type of interface, i.e., USB interface. Also changing from one communication interface to another is not currently possible. Teleos are more flexible and scalable hardware platforms than those mentioned above. Moreover their simplicity, reusability and relatively easy to build complex networked devices would be an ideal platform for building interactive and tangible devices.

## **3.6 Arduino**

Arduino is a physical computing platform based on a simple I/O board and a development environment to develop stand-alone interactive objects or can be interfaced to a computer. Arduino [35] is an open source platform and is similar to Teleo's platform in its modular architecture.

### 3.6.1 Hardware

Arduino hardware consists of I/O boards and shields. There are three versions of the Arduino boards and all of them use the Atmel ATmega8 [36] running at 16MHz. Serial I/O board is the basic board that uses RS232 as an interface to a computer for programming and communications as shown in Figure [3.10]. USB and Bluetooth I/O boards use USB and Bluetooth interfaces, respectively.



**Figure [3.10] Arduino Board (left) and Arduino with a shield (Right)**

Shields are boards to be mounted on top of the Arduino board and that extend the functionality of Arduino boards for adding new sensors or devices to acquire or display data. Current shields include Motor control, Biosensors board, RFID shield, Prototyping shields and more. Shields are shown in Figure [3.10].

### 3.6.2 Software

In Arduino platform, devices are programmed using Arduino language, which is simple to use and resembles C programming language.

### 3.6.3 Applications and Discussion

Arduino platform is the latest of all the presented platforms in this section. Most of the work is currently under progress and a lot more devices are expected in their roadmap

timeline. It mainly concentrates on building devices by the end users beginning from PCB layout drawing to soldering all the required electronics. Their documentation provides instructions on how to build even to individual components with schematics of each board and system. Most of their applications involve building interactive devices.

Arduino's are modular and flexible to build interactive devices. These modules can be reused as they are plug-in modules and requires no soldering for building a new device. These modules can work stand alone, and intelligence can be distributed over the network. They currently have two types of communication technologies USB and Bluetooth. This would allow devices to change from USB to Bluetooth connectivity and vice versa by exchanging the corresponding I/O boards. Developing more types of different communication I/O boards can be helpful in supporting different interfacing technologies. Given the above advantages, Arduino platform can be an interesting platform for constructing interactive and tangible devices.

The following tables show the comparative study of the above platforms based on their physical and electrical properties and also based on the discussed characteristics.

**Table [3.2]: Comparison of different platforms**

Platform	Platform Architecture	Electrical Features	Physical Dimensions	Interface connectivity		Applications	Philosophy
				External	Internal		
Berkeley Motes	wireless dispersed modules	Atmel Atmega 128L, 128Kb Flash, 4Kb EEPROM at 4 Mhz	Cubic millimeters, Mica2 mote 58x32x7 mm	Mote Interface Board, RS-232, USB, Ethernet	315, 433, 868/900MHz, IEEE802.15.4 and ZigBee	Self forming, Low power, Ad-Hoc, mesh sensor networks	Smart Dust vision, monitoring networks
iStuff	iStuff components, Event Heap, Patch Panel	Any physical device having a Proxy that encapsulates data into an event to Event Heap	Any	Any	Any	iRoom, heterogeneous environments with run-time retargetable device data flow	Simplify the novel interaction techniques of multiple users, devices, systems and applications
Smart-its	Teco Particles	PIC12F675, 16F87, at 4MHz, RF communication	<1cm <sup>3</sup> / 45x18 mm / 17x35 mm	RS-232, USB, Ethernet	RFcomm 868MHz, I2C	Interactive devices	Wireless sensing applications, to address deployment issues and do it your self custom devices.
	Btnodes	ATmega 128L, Zeevo ZV4002 Bluetooth	58.15x33 mm	Bluetooth	Bluetooth	Mobile Ad-Hoc networks prototype interactive devices	
	Lanaster DIY	PIC 16F876, 16F87	70x53 mm/ 44x44 mm	RS-232, USB	I2C	rapid prototyping of complex and simple devices	
Phidgets	Integrated components	Cypress CY7C63000	varied	USB	---	Rapid prototyping	Plug and play custom devices
Teleo	Modular, classified as Modules and Blocks	PIC 18F series at 40MHz, Maxim-485	2.5"x2.5"	RS-232, USB	RS-485	rapid prototyping of complex and simple devices	Scalability, development of digital and complex devices
Arduino	Modular, classified as I/O boards and Shields	Atmel ATmega8 at 16Mhz	varied	RS-232, USB	I2C	Stand alone interactive objects or computer connected	Implement Processing/ wiring language
Bladed - Tiles	Modular, classified as Blades and Tiles	PIC 16F876, 16F87 at 20MHz, USB - FTDI, Blue-SMiRf, gumstix	Blade=2x10 cm Tile= 10x10cm	RS-232, USB, Bluetooth	I2C	rapid prototyping of complex and scalable interactive devices	Easy prototyping and deployment of interactive and tangible devices

**Table [3.3]: Characteristics of different platforms**

Platforms	Properties		
	Distribution of Intelligence	communications Integration	Flexibility in switching communications
Berkeley Motes	dispersed	modular	easy and various
iStuff	concentrated	specific	heterogeneous
Smart its	dispersed	varied	limited
Phidgets	concentrated	fully	only USB
Teleo	dispersed	specific	limited
Arduino	dispersed	specific	limited
Bladed-Tiles	dispersed	modular	plugin and various

Communications approach in different platforms
Modular --> Separated from other functions and can be interchanged.
Any --> Any type of communications is supported.
Varied --> Different communications available integrated with processor boards.
Fully integrated --> Communications are integrated within different modules.

## **Chapter 4**

### **Thesis Approach**

The Bladed Tiles toolkit is introduced briefly in this chapter. It is followed by the motivation behind this work and the concept of communication blades. The architecture of communication blades and I2C protocol is provided in the implementation section. More technical details of communication blades are also provided in this section. This chapter ends with the discussion of two example applications that are implemented using these blades.

#### **4.1 Bladed Tiles Toolkit**

Bladed Tiles is a modular hardware toolkit for building tangibles and interaction devices. It comprises of function Blades and Interaction tiles, which can provide a flexible, inexpensive, open-ended platform for constructing a wide variety of tangible and embedded interfaces.

##### **4.1.1 Hardware**

It consists of “function blades,” and “Tiles,” which expose sensing, display, communications, and computation capabilities in a consistent, modular, extensible, easily employed fashion. These blades are each 10x2x1cm in size, corresponding to the 10cm modal length defined by human’s palmar hand grasp posture. The tiles are 10x10x1 cm and can accommodate 5-10 underlying function blades. Each blade has two physically identical 14-pin (7x2) connectors on either end of the blade. One connector is functionally identical across all blades and provides power and communications (RS232,

USB and I2C). The second connector's function is entirely specific to the individual blade.

Tiles and Blades together allow very flexible construction of custom tangible devices that can be used without any knowledge of hardware or low-level software. Because of modular approach, it enables designers to prototype and develop, by plugging in the necessary blades; these tiles can be reused as they are just plug-in devices.

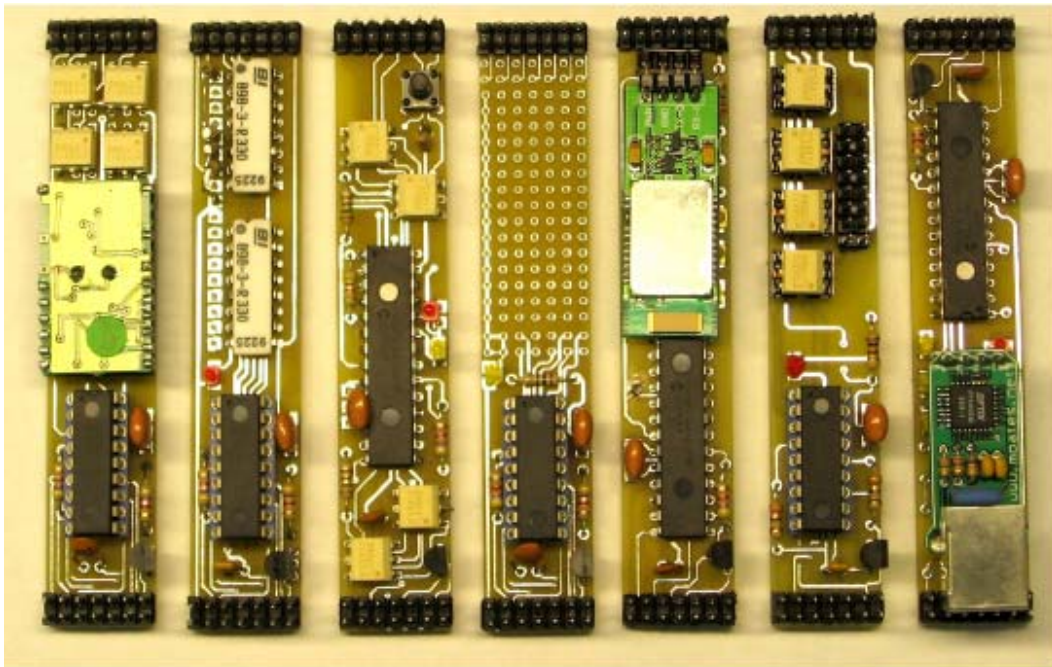


Figure [4.1]: Examples of function blades

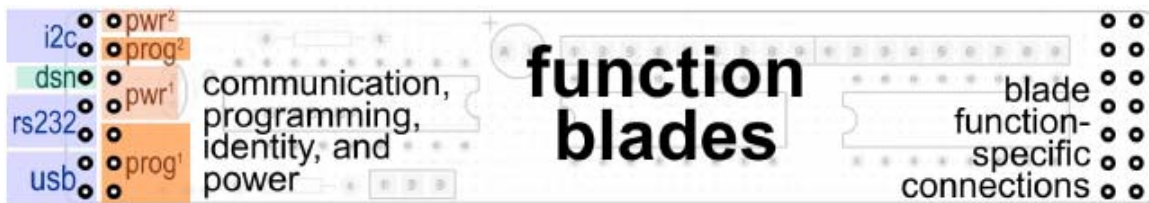


Figure [4.2]: Common pin description of a blade

These blades are extensively used in both the tiled forms of core tangibles, and within our tangible visualizations research group. Nonblade electronics (e.g., Phidgets and Particles) can also be embedded within tiles using their standard USB and I2C buses.

In order to support the evolution of tangible interfaces, Ullmer's group is working to release a suite of blades, core tangibles tiles, and software as open source hardware, firmware, and software. Moreover, this system is designed to be open-ended, allowing a broad variety of special purpose tiles and devices to be designed and integrated at very low costs.

#### 4.1.2 Software APIs

While we believe blades and bladed tiles could become an important technology tool for the development of tangible interfaces, diversity is a virtue, and tangibles using widely varying hardware/software toolkits should be expected. As we feel interoperability should remain a major tangible user interface (TUI) goal, resolving some form of hardware, software, and toolkit independent device description seems highly important. Toward this, we have begun working on several forms of XML-based device descriptions. One of these (in collaboration with Dr. Albrecht Schmidt) is at the hardware protocol level. This will allow diverse software to interact with underlying functions implemented by specific blades, Phidgets, Smart-Its, Particles, etc.



**Figure [4.3]: Tiles with underlying function blades.**

A second level of XML description relates to the composite functional roles of individual interaction devices. Here, we intend to expose an API which is largely independent of the underlying technical implementation. Thus, ideally an interaction device could be reimplemented with blades, Phidgets, etc., while leaving its external functional API unmodified. We are also exploring different kinds of registries and matching software for pairing interacting interaction devices together, including (e.g.) Prolog interactions with Grid-enabled SQL databases. Finally, while we hope end-users will wish to interact with our tangible interfaces, we believe graphical interfaces and other interaction techniques will sometimes remain preferable. We also wish to map the same set of tangibles across various kinds of software implementations (e.g., visualizations using Amira, VTK, OpenDX, AVS, TechPlot, etc.).

## **4.2 Communication Blades**

### **4.2.1 Concept**

Restating thesis statement from Chapter 1

**Communication blades can facilitate communication infrastructure flexibility and connectivity for embedded and tangible interfaces.**

Both the concepts flexibility and connectivity provided by communication blades are discussed briefly in this section.

#### **4.2.1.1 Flexibility**

Communication blades provide flexibility in the following issues while choosing communications for a particular application. Users can choose communication blades for applications built using Bladed-Tiles toolkit depending upon these varying factors.

- a) Complexity
- b) Communication types
- c) Power constraints
- d) Development time and cost
- e) Form factors

a) Complexity:

Devices are expected to be either simple or complex depending upon their processing capabilities and application. Communications on these devices might be simple or complex. Simple devices having lighter communications act like data stream converters or protocol converters such as serial communication blade, converting USB protocol signals to serial signals. Further, complex devices are capable to handle data routing and traffic, such as gumstix communication blade having 400 MHz ARM processor.

b) Communication types:

Communication blades can be selected from variety of communication types such as USB, Bluetooth, serial etc., depending upon the available architectures. Applications requiring wireless connectivity can choose Bluetooth or Gumstix communication blade, whereas the applications requiring wired connectivity can choose USB or serial communication blades.

c) Power constraints:

Power stringent applications require communications that consume less number of watts, whereas power lenient applications can try efficient communication models irrespective of power consumption. Many applications in wearable computing are

power conscious because of power storage restrictions. Hence communications with less power consumption are preferred for these applications. Table [4] shows the corresponding power consumption of each communication blade.

**Table [4.1]: Communication blades power ratings**

Communication Blades	Power Consumption
Serial	~15mA (Processor) + ~10mA (led)
USB	~500mA (usb)+ ~15mA (Processor) + ~10mA (led)
Bluetooth	~25mA + ~15mA (Processor) + ~10mA (led)
Gumstix	<250mA at 400Mhz without Bluetooth

d) Development time and Cost:

With the available development time and costs involved, users can choose from various communication blades. An ambitious project which requires high accuracy and reliability would obviously choose an intelligent/complex communication module. This can be later molded to their necessities, while a hobbyist would choose readily available communication blade, which definitely serve his purpose and also fits in his budget. Using communication blades, users can dramatically reduce the development time and cost as no knowledge of communication protocols is required.

**4.2.1.2 Connectivity**

Communication blades provide connectivity by supporting interoperability and adaptability over different infrastructures. Seamless connectivity over different networks is achieved by interchanging corresponding communication blades. For example, a device containing, USB communication blade can be replaced with a Bluetooth communication blade to adapt to the wireless network. As a result, the same device can be used over different communication architectures by simply interchanging various communication blades. Hence communication blades provide

- a) Interoperability,
- b) Product development, independent of communication infrastructure, and
- c) Adaptability to future connection technologies.

Communication blades support different connection media. More types of communication blades can be developed over time along with the increase in number of communication technologies and necessity of a particular communication protocol. This larger support allows smoother transition over different architectures, there by allowing a device to be interoperable between different systems, without any significant change and additional development effort.

Communication infrastructures vary largely from place to place and are dispersed. This diversity makes it difficult for designers and engineers to develop a product that can be used in various locations. With the help of a modular communications approach, products can be developed independently of various infrastructures. New communication blades can be developed to support the latest communication technologies. These new communication blades can be plugged into older devices, there by adapting to newer communication technologies.

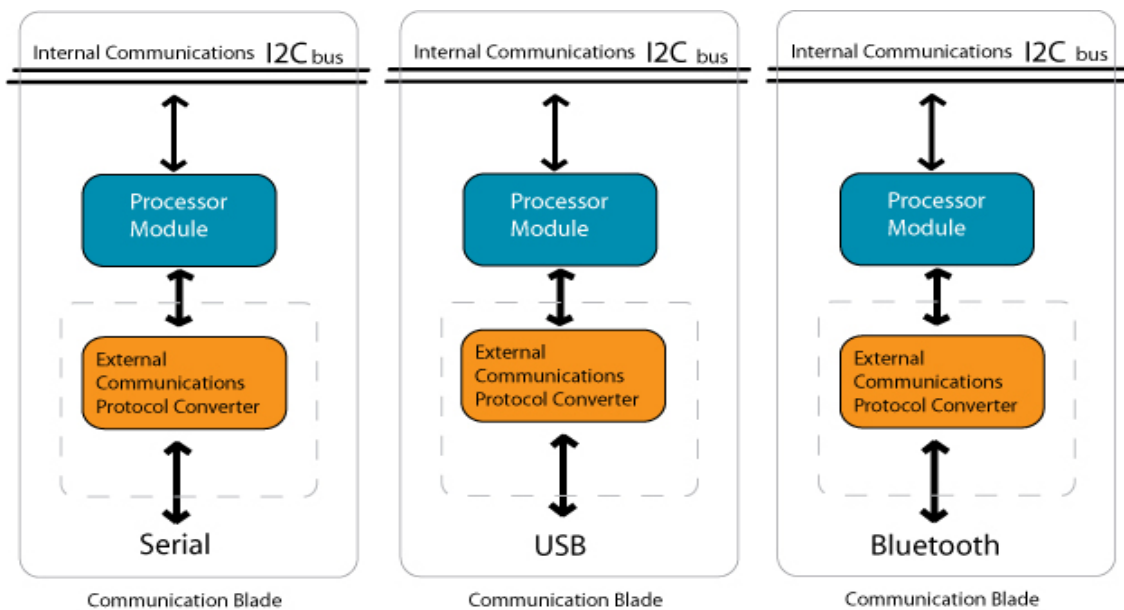
### **4.3 Implementation**

This section briefly describes the architecture and the various communication blades implemented, based on the Bladed-tiles hardware platform. Communication blades are implemented using commercially available off-the-shelf components. These prototypes have been rigorously tested, revised and some of them are currently being used in practical applications.

### 4.3.1 Architecture

#### 4.3.1.1 Model

A communication blade is itself implemented using a modular approach. As a modular approach supports faster integration of latest communication protocols on to communication blades, variants of communication Blades can be implemented on demand. The basic function of a communication blade is to convert different communication protocols to the I2C protocol, thereby acting as a gateway between the internal and external communications. A communication blade consist of two modules as shown in Figure [4.4], the processor module and the communication module.



**Figure [4.4] Architecture of communication blades**

The processor modules are based on microchip PIC16F876 microcontroller offering 8Kx14 words of Flash program memory, 368 x 8 bytes of data memory (RAM), and 256 x 8 bytes of EEPROM data memory running at 20 MHz, and also support serial and I2C communications. It can function both as I2C master and slave during the operation. The main function of the processor module is to interface the communication

module and the functional blades, by converting signals to the I2C protocol, there by making blades independent of any technology or protocol. The processor modules can also incorporate intelligence to handle address control, data control and data processing.

The communication module consists of commercially available protocol converters, which helps in converting a high level protocol to a lower level protocol. Higher level is the interface to the external communications and lower level is the interface to the processor module. This version is implemented in the Serial, USB and Bluetooth communication Blades. Further, the communication module can have multiple communication protocols simultaneously. This version is implemented in Gumstix communication blade. All the communication blades follow the same pin out structure as followed by function blades. Their common pin out layout is shown in Figure [4.2].

#### **4.3.1.2 I2C**

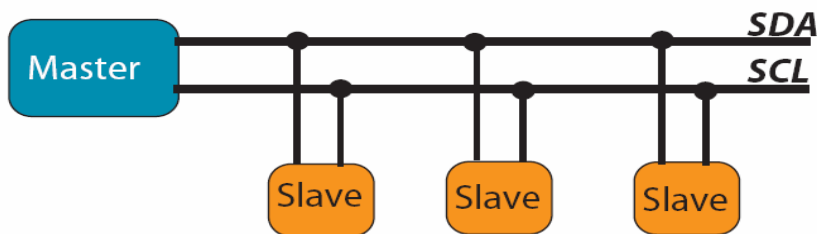
The I2C bus [38] is a two-line communication protocol developed in the early 1980's by Philips Semiconductors. Its original purpose was to provide an easy way to connect a CPU (Central Processing Unit) to peripheral chips in a television set to provide a communication link between integrated circuits. I2C is a low-bandwidth, short distance protocol for on-board communications. I2C is an acronym for Inter-IC bus.

##### **I2C Bus Protocol:**

The I2C bus physically consists of 2 active wires and a ground connection. The active wires, called SDA and SCL, are both bi-directional. SDA is the Serial DATA line, and SCL is the Serial CLOCK line. Every device connected to the bus has its own unique address, no matter whether it is an MCU, LCD driver, memory, or ASIC. Each of these chips can act as a receiver and/or transmitter, depending on the functionality. Moreover,

an LCD driver is only a receiver, while a memory or I/O chip can be both transmitter and receiver.

The I2C bus is a multi-master bus. This means that more than one IC capable of initiating a data transfer can be connected to it. The I2C protocol specification states that the IC that initiates a data transfer on the bus is considered the Bus Master. Consequently, at that time, all the other ICs are regarded to be Bus Slaves. This is shown in Figure [4.5].



**Figure [4.5] Master and Slave devices connected to I2C bus**

**Features:**

I2C has many important features worth mentioning. It supports multiple data speeds: standard (100 kbps), fast (400 kbps) and high speed (3.4 Mbps) communications.

Other features include:

- Built-in collision detection,
- 10-bit addressing,
- Multi-master support,
- Data broadcast (general call).

More features can be found at [38].

**Benefits and Drawbacks:**

Since only two wires are required, I2C is well suited for boards with many devices connected on the bus. This helps reduce the cost and complexity of the circuit as additional devices are added to the system.

Due to the presence of only two wires, there is additional complexity in handling the overhead of addressing and acknowledgments. This can be inefficient in simple configurations and a direct-link interface such as SPI might be preferred.

### **Why I2C?**

Communication blades use the I2C protocol to communicate with functional blades. Other protocols are also considered for inter-embedded device communications such as USB, RS232, and SPI, which are popular protocols, but we chose I2C because of its scalability and simplicity. USB provides scalability, but requires a hub to do so. The USB would require more complexity in microcode of each device and is actually overkill for the simple communications between devices. RS232 is simple but does not scale because device communication is only one-to-one. The implementation of SPI requires  $n+2$  lines,  $n$  being the number of devices. This limits flexibility in the number of devices that can be added once the hardware is finalized.

I2C scales in the number of devices without any additional hardware, provides a flexible addressing scheme, supports plug and play and is widely available on a number of variant devices. I2C's speed is sufficient because communications between devices are simple and small in size because only processed data is exchanged.

### **4.3.2 Hardware**

This section briefly describes about electrical features and characteristics of four implemented communication blades.

The following four communication blades are discussed further in detail.

- a) Serial Blade
- b) USB Blade
- c) Bluetooth Blade
- d) Gumstix Blade

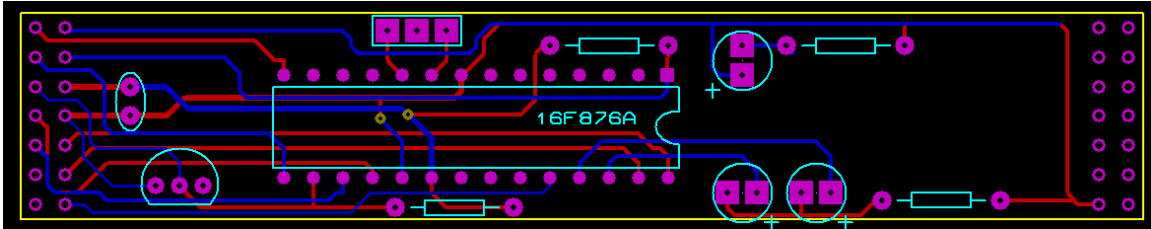
#### **4.3.2.1 Serial Blade**

The basic function of the serial communication blade is to provide serial connectivity to the function blades. It provides standard RS-232 serial interface to the computers at a maximum baud rate of 57.6kbytes/sec. The hardware module consists of Microchip's PIC 16F786A microcontroller running at 20 MHz. It supports both I2C master and slave functions. This module implements I2C master functions. Tx and Rx of the microcontroller are connected to the corresponding pins of the DB9 serial connector. The microcontroller converts the serial signals to I2C signals and distributes the data to the connected function blades.

The implemented serial blade communicates at 9600 baud with the external serial link and at 100 KHz with internal I2C bus. Two LED's provide simple debugging. It is the simplest of all the communication blades described. Serial blade simply converts serial protocol to I2C protocol, without much data processing or manipulation, there by acting as a plain protocol translator. This serial blade is transparent to the users, since no data handling is involved. This communication blade is shown in Figure [4.6] along with its PCB schematics.

Serial ports are widely used in the past decade for common device interface connectivity. Although its usage diminished after the advent of the USB interface, a serial

port still exists in most computers. Moreover, it is easier to implement than any other interfacing technologies both in hardware and software. Hence, these blades can be useful in prototyping embedded and interactive tangible devices with the Bladed Tiles toolkit.



**Figure [4.6] PCB layout of serial blade**

#### **4.3.2.2 USB Blade**

A USB blade provides plug-and-play interface connectivity to the functional blades. A USB blade consists of two modules as explained in the architecture, the processor and the communication module. The processor module consists of Microchip's PIC 16F786A microcontroller running at 20 MHz and implements I2C master functions. The communications module consists of off-the-shelf readily available Moates USB to serial converter Figure [4.7], which converts external USB signals to serial signals. Tx and Rx of the microcontroller are connected to the corresponding pins of the USB to serial converter. The processor converts the serial signals from the communications module to I2C signals, there by bridging I2C bus to the external communications.

The implemented USB blade communicates at 9600 baud rate with the USB host and at 100 KHz with internal I2C bus. Two LEDs provide simple debugging. USB blade interfaces the function blades as USB serial client device to the host using FTDI B-232 chipset having internal EEPROM, I/O's at TTL levels and supports baud rate up to 3Mbit/second. FTDI chip indirectly induces plug-and-play capability to the connected function blades. Moates USB serial converter can be replaced by any other USB serial

converters commercially available with identical hardware pin outs. USB blade converts serial protocol to I2C protocol, without much data manipulation, there by acting as a protocol converter. This USB blade is transparent to the users, since no data handling is involved. This communication blade is shown in Figure [4.7] along with its schematics. More specifications of the Motes USB module can be found in the Appendix.

With more than 2 billion legacy wired USB connections in the world, USB has become the de facto standard in the personal computing industry. Its success is mainly because of its simplicity and plug-and-play capability. However, developing a USB client device require hardware skills, time consuming and complex. A USB blade can be helpful in developing or prototyping devices with function blades requiring USB connectivity and without any USB knowledge.

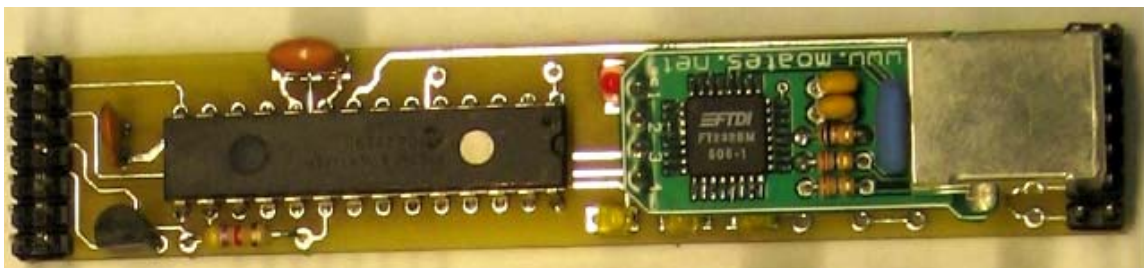
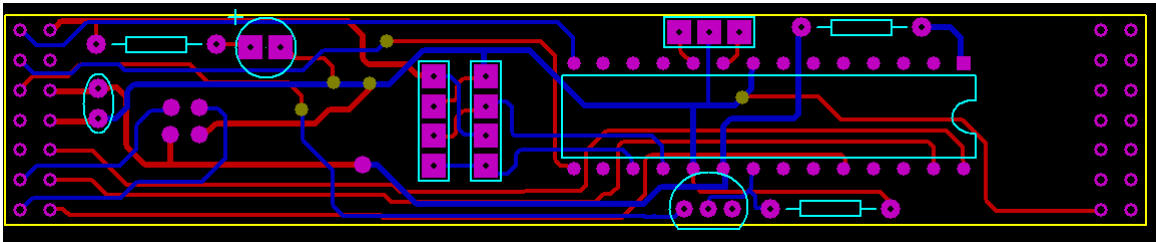


Figure [4.7] PCB layout of USB blade and implemented USB blade

#### 4.3.2.3 Bluetooth Blade

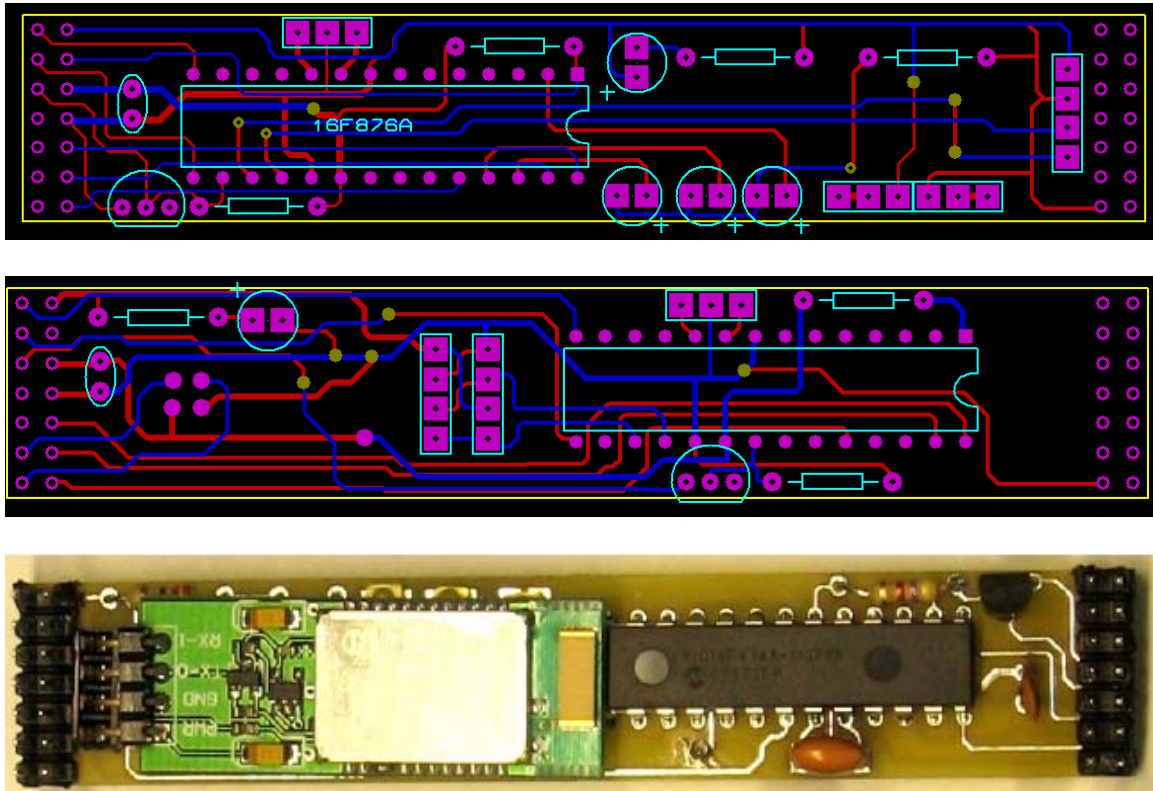
Bluetooth is a low speed, short-range wireless communication standard using Frequency Hop Spread Spectrum (FHSS) technology at 2.4 GHz. It facilitates interconnection between devices within a specified range to exchange data and also to maintain voice

connectivity. The Bluetooth blade consists of two modules and it is almost similar to the USB blade. The processor module is replica of the one used in the USB blade, it has a PIC 16F876A microcontroller running at 20 MHz and implements I2C master functions. The communication module consists of Bluetooth modem BlueSMiRF from Sparkfun electronics instead of the USB to serial converter as used in the USB blade. BlueSMiRF modem is shown in Figure [4.8] converts external Bluetooth signals to serial signals and vice versa. The processor converts the serial signals from the Bluetooth module to I2C signals. Hence, the Bluetooth blade provides wireless connectivity to the function blades. BlueSMiRF is a class 1 product and has a maximum range of 100 meters.

The developed Bluetooth blade communicates at 9600 baud rate with the paired Bluetooth device and at 100 KHz with internal I2C bus. These modems work as a serial (RX/TX) pipe. Any serial stream from 9600 to 115200bps can be passed seamlessly from the paired unit. It needs 3 to 5v to function under normal conditions. Power distributed to the Bluetooth module is separated from the power distributed to the processor module, to avoid power glitches generated by the Bluetooth module during the operation. More technical specifications of BlueSMiRF can be found in the Appendix. The Bluetooth modem converts external wireless signals to serial signals and vice versa. The processor module in turn converts serial to I2C signals and vice versa as implemented in the USB blade. The schematics and developed model can be seen in Figure [4.8].

Bluetooth is popularly used for short range communications and device interconnectivity. It has gained popularity because it could remove user intervention for near-by communications, wires and the entire process of pairing to neighboring device, authentication and exchanging data is automatic. Bluetooth is widely used in ubiquitous,

augmentative, pervasive computing and tangible interfaces. This blade can provide wireless connectivity while developing devices requiring Bluetooth communications with the Bladed Tiles toolkit.



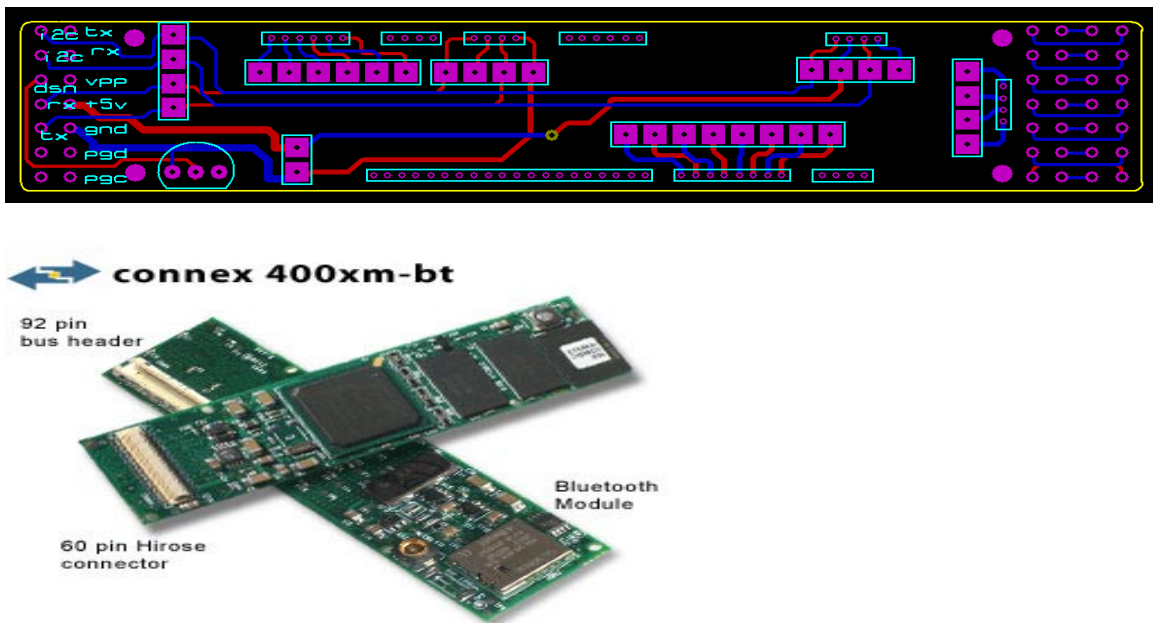
**Figure [4.8] PCB layouts of Bluetooth blade and implemented Bluetooth communication blade**

#### **4.3.2.4 Gumstix Blade**

Gumstix are Full Function Miniature Computers (FFMC) running Linux 2.6 operating system at 200/400 MHz with Intel X-Scale PXA255 processors having 64MB flash memory. They measure about 80mm x 20mm x 6.3mm in physical dimension. The Gumstix line features three platforms: connex, connex-xm, and basix, shown in Figure [4.9]. Gumstix feature an Infineon Bluetooth Module as an option. All the three platforms draw less than 250mA at 400MHz without Bluetooth while running and less than 50mA

while waiting for input. Gumstix can be interfaced using any one of the Bluetooth, USB, serial and Ethernet protocols. On all Gumstix platforms, expansion cards can be connected via an onboard 60-pin Hirose I/O header. Additional specifications about Gumstix can be found in the Appendix. These Gumstix platforms offer high performance, low power consumption and allow greater design flexibility for wireless and embedded products.

A developed Gumstix blade fits like an expansion board to the original Gumstix. This blade integrates Gumstix into the Bladed Tile toolkit as a compute and communication blade. Gumstix has onboard serial and I2C master interface that can be controlled using Linux shell programming. Schematics and developed Gumstix blade are shown in Figure [4.9]. Gumstix is programmed to communicate with the external interface using Bluetooth or USB interface at a baud rate of 9600 bits. Xterasys Bluetooth dongle is used as a pairing device, which is connected to the host computer. Gumstix communicates at a rate of 100 KHz with the internal I2C bus.



**Figure [4.9] PCB layout Gumstix communication blade and Gumstix**

Data received over Bluetooth or USB is processed and routed to the corresponding function blades. Gumstix is capable of routing, data processing and addressing with the help of its dominant 400MHz processor and flexible Linux OS. Gumstix incorporates intelligence onto the blade level, thereby distributing intelligence over the infrastructure as desired. Gumstix acts a mediator between the external Bluetooth and USB interfaces and the internal I2C bus. It can also act as a stand-alone module if sufficient information is provided.

The current decade will probably be known as the dawn of pervasive computing, when PCs were dethroned by technology to embed computers in almost everything. The trend in adding features such as artificial intelligence and wireless connectivity to attain ubiquity is growing enormously. When necessary hardware become small, smart and possibly even invisible, the things in the environment will communicate with one other and without human interference and knowledge. Gumstix can facilitate in transforming bladed-tiles toolkit device applications to the pervasive and ubiquitous applications.

**Table [4.2]: Electrical characteristics of Communication Blades**

Communication Blades	Data Baud rates		Operating Voltages	Operating Frequency	Components	Power Consumption
	Max	Imple- mented				
Serial	57.6 Kb	9.6Kb	2.0 - 5.5 V	DC - 20 MHz	PIC 16F876A	~15mA (Processor) + ~10mA (led)
USB	1 Mb	9.6Kb	4.35 - 5.25V	6 - 48 MHz	PIC 16F876A, FTDI FT232BM	~500mA (USB)+ ~15mA (Processor) + ~10mA (led)
Bluetooth	115.2 Kb	9.6Kb	3 - 6V	2.4 - 2.524 GHz (ISM)	PIC 16F876A, BlueSMiRF	~25mA + ~15mA (Processor) + ~10mA (led)
Gumstix	---		3.4 - 5.2 V	200 MHz and 400 MHz	Intel XScale® PXA255, Infineon Bluetooth	<250 mA at 400Mhz without Bluetooth

#### **4.3.2.5 Electrical Properties**

The electrical specifications of all the communications blades are summarized in table [4.2].

### **4.4 Applications**

This section presents two applications that are implemented using the Bladed Tiles toolkit, namely the Core tiles and eNote. Core tiles application is used in building various tangible devices while eNote is an end product which notifies user emails on the go. Both of these applications are explained in brief.

#### **4.4.1 eNote**

eNote is a simple device built using the Bladed Tiles toolkit. It is used for notifying users whenever they receive an email. It is constructed using switch-led blade and a communication blade. It has four LEDs, a vibrator and a mono buzzer. Users are alerted whenever there is a new email to their account by turning on the appropriate led along with buzzing sound and vibration. eNote is shown in Figure [4.10] .

eNote is controlled by a simple java application. In this java program, users aggregate their preferred email ID's into four categories and label them appropriately. In this example they are grouped as work, family, friends and others. When the user receives new mail, the Java program identifies the class to which it belongs and sends corresponding signals to the eNote device. This java program resides on a remote machine to which eNote can be connected by serial communication blade or by Bluetooth communication blade.



Figure [4.10] eNote individual components and when functional

Users can choose the preferred communication blade when required. eNote is customizable and personalized. For example, the same device can be utilized for notifying job status on a supercomputer. It can be personalized by changing the faceplate on the device.

#### 4.4.2 Core Tiles

Communication blades can be used in interfacing various devices for them to talk to the external world. One of the main areas they were used was in core tiles/tangibles. Core tangibles/tiles are a convergent technology that can do core operations. Core operations include loading and saving data on various applications and with different kinds of interfaces (GUI, virtual reality). More about core tiles/tangibles can be found at [40].

These tangibles are built using bladed-tiles modular toolkit. Communication blades serve in terms of internal communication between blades housed on a tile and with much bigger and complex systems outside the tile. Communication blades act as gateways between the communication network and the core tangibles.

Here USB and Bluetooth Communication blades can be used interchangeably. USB connectivity is used when the core tiles are stationary and Bluetooth communication blade is used when the tiles need mobility over a distance.



**Figure [4.11] Core tiles**

## **Chapter 5**

### **Discussion and Conclusion**

#### **5.1 Discussion**

Recently, there is a significant increase in the number of communication technologies. New technologies enabled by Moore's Law and the advances in wired and wireless communications are the primary forces responsible for the explosion of these communication technologies. This trend is evident. Keeping up with the latest communication technologies and knowing which ones have technical merit for a particular application is an increasingly overwhelming task. These technologies are not only different from one another but are also widely dispersed.

For a device to endure for a longer period it should be adaptable and customizable. Consider the first generation digital cameras; all of them have inbuilt memory to store the digital pictures. In the second generation users were given the option to plug-in different capacities of memory depending upon their usage. Currently, there are numerous types of memory cards in different sizes available in the market for the users to choose. Over time, improvements in VLSI and packaging technologies can reduce the physical size and cost of communication technologies. If a common interface architecture is followed even in embedded devices as the PCI architecture in laptop computers, then there can be various plug-in communication modules. This trend of modular approach in embedded systems is evident from the existing communication modules based on RS232 communications such as serial-based Bluetooth, USB etc.

It is not only adaptability and customizability of a product that are needed to survive in the market but also how easily they can be customized and used by the users.

For example, interfacing devices such as joystick, keyboard, etc., with RS232 protocol requires users to configure few settings such as the port number, baud rate, IRQ. This might be difficult to some users. On the other hand, interfacing devices using USB connection is a single step process, simply plug it and start using the device. No configuration is needed by the users, and this effect is phenomenal for USB's dominance in the current market. The future communication modules have to be plug-and-play and should not require much user attention for configuring the modules.

Devices built using Bladed-Tiles are customizable because of modular architecture. Each blade can be developed or improvised individually irrespective of the other blades. Communication blades provide adaptability and extensibility over different communication infrastructures. These blades are also easy to use requiring no further setup once installed. More types of communication blades can be developed in the future to support new growing technologies. For example developing a WiFi communication blade would help in using the existing device to communicate with the WiFi network.

In the future there might be a library of hardware blades from which the users can choose to develop a new device. This can be compared to the current practice of including different modules of code from a package or library during software development. Users can simply select various blades from the library to build a device without bothering about the working of each blade. Let us consider the following two future scenarios, where communication blades can be used.

Consider a device developed using bladed-tiles, which displays the current temperature and weather. These types of devices are placed in all the shuttles in the United States so that the commuters can know the temperature and weather of the day.

These devices are updated regularly by connecting to the local cellular mobile network using GSM communication blade on road. What if a similar type of device has to be used in Europe or India for their transportation system? GSM (Global System for Mobile communications) specifications in Europe and India are different from United States. The communication module has to be replaced with European standard GSM module to work in Europe.

One might argue that integrating different GSM modules in the devices during development and using those in the corresponding zones should solve the problem. But cellular specifications in India alone are varied from region to region. Some regions might support only wiLL (Wireless in Local Loop) rural technology and few other regions might support only GSM technology. Will integrating different communication modules solve the problem? Switching a corresponding communication blade module may be a better option. While the above mentioned differences are only in cellular communication technologies, consider the wide variants of available connection technologies.

Consider a tangible device, that has a flat surface embedded with RFID sensors underneath it. It contains physical blocks in shapes of numerical numbers. These physical numbers are tagged with RFID coils and can be detected when placed on the sense table. It also has physical blocks in the form of a telephone, radio and a T.V. This device is connected to the network server which controls T.V., radio and telephone in a room. Using this device, users can control T.V., channels by placing physical numbers tagged with RFID's on the table. The sense table identifies the numbers placed on the table and changes the T.V., channel accordingly. The same technique can be used to tune

the radio to a desired frequency or to call a person using customized blocks that store telephone numbers. Interaction with T.V, radio and telephone can be changed by simply placing the corresponding blocks on the table.

Consider an existing tangible device such as a water lamp as shown in Figure [5.1]. In this fixture water ripples are created by raindrops on the surface of still water. These physical raindrops are envisioned as “bits” (digital information) falling from cyberspace creating physical water ripples. The raindrops of “bits” are realized with computer-controlled solenoids tapping the water.



**Figure [5.1] Water Lamp**

These types of tangible devices are not rare in future rooms [41]. This device can be placed in a waiting room or a hall where the visitors can use it as desired. This device needs to be constantly in touch with the concerned devices to control them. How can we

make similar domain specific devices work in different regions with varying network infrastructures? It can be made possible by following a modular approach in communications.

Bladed-Tiles can be used to build domain specific tangible interfaces, such as devices which can depict culture, religion and so on. India is one of the most religiously diverse nations in the world, with some of the most deeply religious societies and cultures. Religion plays a central and definitive role in the life of the country and most of its people. Yet, people from one religion might not know much about other religions.

Consider a tangible device, which is built by using Bladed-Tiles. This is an artistic device built by a designer and the outer skeleton of this device depicts an Indian map with all the states represented, each having an RFID reader underneath. This device also has eight inch touch screen to display the information. Various blocks tagged with RFID's are available to place them on the map. These blocks are of various shapes and are in the forms of different categories, such as Church to represent Christianity, Temple to represent Hinduism, Mosque to represent Islam and so on. There would also be blocks named rituals, locations, etc.

Users can place the desired blocks on a state and know the information. For example, if the user wants to know about Christianity in a particular state, he can place the church block on the desired state. The information about Christianity corresponding to that state is displayed on the screen. If he also wants to know about the various churches available or the rituals practiced in that state he can simply place the locations or rituals block on that state.

This device needs to be constantly updated with data over the Internet, but the communication networks in India are widely varied and changing rapidly. Some users will have only dial up connection whereas few might have fast WiFi network. How can we make similar devices work in different regions with varying network infrastructures? It can be done by simply plugging in the corresponding communication blades.

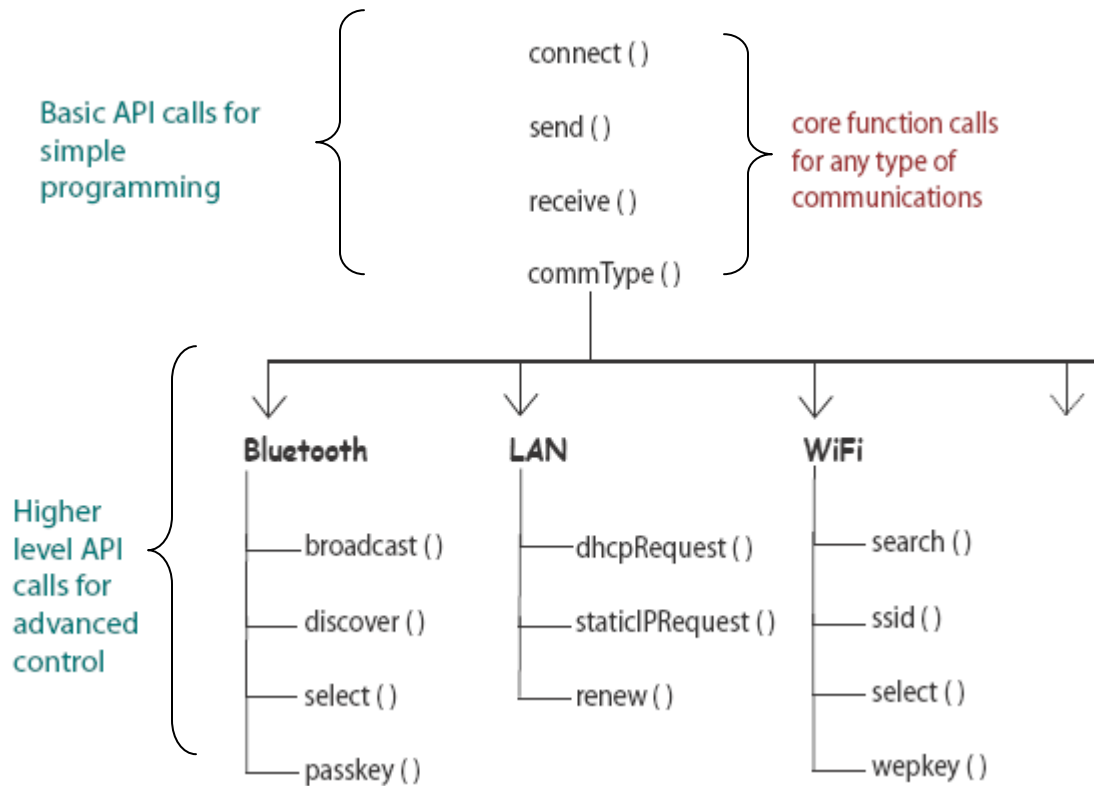
### **5.1.1 Discussion about API**

An application programming interface (API) is the interface that a computer system, library or application provides in order to allow requests for services to be made of it by other computer programs, and/or to allow data to be exchanged between them. A good API makes it easier to develop a program by providing all the building blocks. Programmers put the required blocks together to complete a task.

Here we will discuss about the general API function calls that are expected to ease the programming of various communication types. Using these function calls programmers can automate the process of detecting the type of communication module plugged in and use it accordingly. Programmers can program it in such a way that the automation process can be completely hidden from the end users or give some control to them. For example, if a user switches a USB module with a Bluetooth module, it can start working without any further confirmation from the user or the system can ask the user to choose the desired external Bluetooth link it has to connect to, passkey and so on.

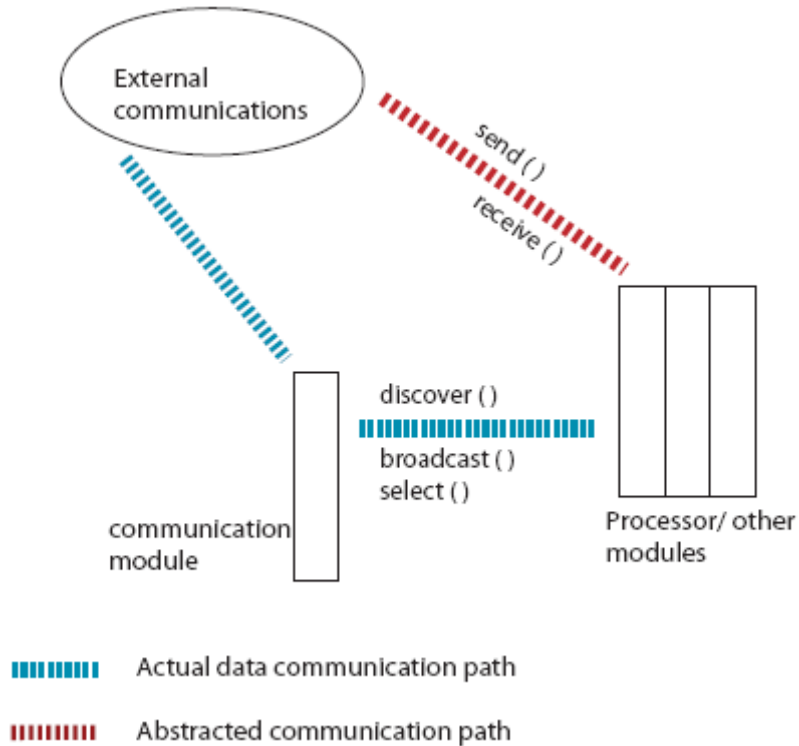
Figure [5.2] presents most of the API calls for some of the communication types. These API calls are to be made by other modules to the communication modules. The communication modules respond to the API calls and initiate connection with the appropriate external communications link. Once the connection has been made, the data

can be transferred by encapsulating the communication module. Figure [5.3] shows the actual communication path and the abstracted communication path.



**Figure [5.2]: General API function calls**

Send, receive and commType are basic function calls for any type of communication modules. Send and receive function calls are used for sending and receiving data correspondingly. CommType returns the type of communication module currently plugged in. Remaining function calls are specific to each communication type. For example a Bluetooth module has broadcast, discover, select and passkey function calls. New communication modules can be added by simply defining the necessary API function calls.



**Figure [5.3]: Abstracted and actual data communication routes.**

### 5.1.2 Integrated vs Modular Approach

There is always a question of whether to have communications in a device using a modular approach along with one or more communications integrated into the device or to only have communications using a modular approach. That is, one might argue that having minimum of one communication type integrated into a device is better than a having no integrated communications at all. On the other hand, one might also argue that having communications using only modular approach can be advantageous.

The solution can be dependent on the device type and also production costs. Consider a device like a laptop. Even though it allows new communications by using PCI cards, it has integrated WiFi, LAN and modem. Laptops are for mobility and they need to provide connectivity when required. Here integration of the most common and used

communication types seems to be advantageous as they are used most and frequently. Moreover, integration of these technologies is not a big cost factor since they are mass produced.

Consider another device like the marble answering machine as proposed by Bishop. These kind of devices need to provide connectivity with the available networks. These are not mass produced as much as laptops are produced and adding multiple communication types can be a costly affair. Communications using a modular approach seems to be better than integrating, as needed communications can be added when required.

One can argue and take a stand on any side of this question, but this discussion is to tell that it depends upon the type of the device and production costs.

### **5.1.3 Limitations**

This section briefly describes the limitations of communication blades.

- 1) Developers have to use the breakout board if the communications blades are to be used without the tiles and they can only be used to interface I2C based devices.
- 2) Modularity has its own advantages but may lead to longer design times due to more complexity introduced because of the modularity. Communication blades are already designed for the end users.
- 3) Developers need to know the communications architecture and protocols used in the bladed-tiles toolkit, if they have to develop a new communication blade. As most of the communications are controlled by the communication blade the other blades can be developed easily and at a faster rate. This limitation is an advantage and leverages the usage of the Bladed Tiles toolkit by the end users.

- 4) Communication blades are relatively more costly than other blades. They are more intelligent and have more processing power than other blades and are necessary to control the communications.

#### **5.1.4 Technological Implications**

This section briefly presents the various technological implications of the Bladed-Tiles toolkit. Currently there are two types of communications to be implemented in this toolkit. One is I2C which is bus type communications and is fully implemented. The second one is USB based which is a hierarchical type of communications and is partially implemented. There is no assurance that we would use the same I2C and USB communication protocols in the future. Over the evolution there might be new technologies invented and can be dominant. The Bladed Tiles toolkit is designed in such a way that they can readily integrate the new communication protocols depending upon whether it is bus or hierarchical type.

Hazardous wastes are generated by nearly every industry. The earth's environmental problems, primarily the problem of global warming and industrial waste management, are a serious issue common to all mankind. In fact, it is clear that in a socioeconomic system characterized by mass production, mass consumption and mass generation of waste, the sustainability of the earth is in danger. Most of the current electronic devices are discarded if they malfunction since they are either hard to repair because of dense integration or repair costs are higher than production costs. The Bladed tiles toolkit is designed for reusability keeping environmental consequences in mind. Systems developed using the Bladed Tiles toolkit can be repaired easily by simply replacing the damaged module with a newer or working module.

### **5.1.5 Design Issues**

Various issues are considered while building communication blades. Mainly,

- a) Form factor: All the blades in the bladed tiles toolkit should have the same physical form factor and connectors. Therefore they can be used interchangeably. Communication blades should also have the same form factor as other blades.
- b) Design evolutions: Different forms of communication blades were built and changed over the usage. Such as having USB physical connector within the blade over a period time and removing it off the blade.
- c) Security: It's always been a question of concern during the development of these devices. As we always be aiming at usage of these devices in accessing important and at times sensitive data. Although most of the communication protocols provide some form of encryption, it is a still a question as to whether that is sufficient or not.

## **5.2 Conclusion**

How might small, special purpose devices be used across diverse communication infrastructures? How can the transition from one infrastructure to the others be eased both for developers and potential end users? These above questions that are mentioned in the introduction section of this thesis can be answered by using modular based communications. Modularity allows flexibility and extensibility. Communication blades are developed to be modular in architecture.

The following communication blades are developed in this work for using them in Bladed-Tiles toolkit. This toolkit is used for rapid prototyping and for deployment of

tangible and embedded devices. These blades are used for switching between different communication technologies.

- a) Serial communication blade
- b) USB communication blade
- c) Bluetooth communication blade
- d) Gumstix communication blade

Both serial and USB communication blades are wired interface technologies and are used with devices where wired connection is available and suitable. The Bluetooth communication blade implements Bluetooth wireless communication protocols on the blade. This blade is used on devices where wireless connection is preferred. The Gumstix communication blade is a readily available small Linux computer running on an ARM processor. It is used for communications and also for onboard computations as it is more powerful than the other implemented blades.

Communication blades provide flexibility and connectivity over different communication infrastructures. They provide flexibility in choosing from different

- Complexity of devices ,
- Communication types,
- Power constraints,
- Development time and cost, and
- Form factors.

They provide extensibility while switching from one communication interface to another by providing:

- Interoperability,

- Product development and usage, independent of communication infrastructure,
- Adaptability to future connection technologies.

The goal is to provide connectivity that is effortless and reliable in all kinds of communication infrastructures. Using modular communication blades in Bladed-Tiles toolkit has the following advantages:

- Flexibility to choose from different communications based on various constraints,
- Reduce product complexity and development effort,
- Provide connectivity to different infrastructures and interoperability,
- Product reusability, and
- Simplicity in usage.

Communication blades can also be used as adapters to interface I2C based embedded devices.

The above communication blades have been designed, prototyped, tested and redesigned through the work. Some of them are currently being used in the real time deployment of tangible devices [40]. Building devices using hardware toolkits having modular communications has potential for interactive products that are flexible and customizable.

My research work includes studying and designing various communication blades i.e. Serial, USB, Bluetooth and Gumstix. It also includes prototyping, testing and implementing the communication blades.

## References

- [1] Crampton Smith, G., "*The Hand That Rocks the Cradle*". I.D., May/June 1995, pp. 60-65.
- [2] Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong and Verplank "*Curricula for Human-Computer Interaction*", ACM SIGCHI
- [3] Ernest Holm-Svendsen, Master Thesis project on the "*theory of Augmented Reality*"
- [4] Brygg Ullmer, "*Tangible Interfaces for Manipulating Aggregates of Digital Information*"
- [5] [http://www-static.cc.gatech.edu/classes/cs6751\\_97\\_fall/projects/gacha/daniels\\_essay.html](http://www-static.cc.gatech.edu/classes/cs6751_97_fall/projects/gacha/daniels_essay.html)
- [6] Satyanarayana, M. "*Pervasive Computing: Vision and Challenges*". IEEE Personal Communications, 2001.
- [7] "*Things that think*", <http://ttd.media.mit.edu/>
- [8] H. Ishii and B. Ullmer. "*Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms*". In Proceedings of Conference on Human Factors in Computing Systems (CHI '97), ACM, Atlanta, March 1997, pp. 234-241.
- [9] J. Rekimoto and E. Sciammarella. "*ToolStone: Effective Use of the Physical Manipulation Vocabularies of Input Devices*". In Proceedings of UIST 2000.
- [10] <http://www.zib.de/ullmer/tutorials/swiss03/s1/html/Folie06.html>
- [11] Steve Mann's Keynote Address entitled "*WEARABLE COMPUTING as means for PERSONAL EMPOWERMENT*" presented at the 1998 International Conference on Wearable Computing ICWC-98, Fairfax VA, May 1998

- [12] Mann, Steve. “*Smart Clothing: The Shift to Wearable Computing*”. Communication of the ACM, Vol 39, No. 8. August 1996.
- [13] *Technology Enabled Clothing*. <http://www.technologyenabledclothing.com>
- [14] <http://www.eleksen.com/>
- [15] B. Warneke, L. M. Last, B. Liebowitz and K.S.J. Pister. “*Smart Dust: Communicating with a Cubic-Millimeter Computer*”. In IEEE Computer. January 2001, pp. 44-51
- [16] *Crossbow Technology, Inc.*, <http://www.xbow.com>
- [17] D. E. Culler, J. Hill, P. Buonadonna, R. Szewczyk, and A. Woo. “*A Network-Centric Approach to Embedded Software for Tiny Devices*”. EMSOFT 2001, Oct 2001.
- [18] S. Madden, W. Hong, J. M. Hellerstein, and M. Franklin. TinyDB web page. <http://telegraph.cs.berkeley.edu/tinydb>.
- [19] Deployment of 800 Berkeley motes. Live demonstration during the keynote of the Intel Developer Forum in Fall 2001. <http://today.cs.berkeley.edu/800demo/>
- [20] P. Levis and D. Culler. “*Maté: A Tiny Virtual Machine for Sensor Networks*”. In Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS X), October 2002.
- [21] Rafael Ballagas, Meredith Ringel, Maureen Stone<sup>1</sup>, Jan Borchers “*iStuff: A Physical User Interface Toolkit for Ubiquitous Computing Environments*”
- [22] <http://iros.sourceforge.net/>
- [23] Smart-its Project Webpage, <http://www.smart-its.org>

- [24] M. Beigl, T. Zimmer, A. Krohn, C. Decker and P. Robinson. “*Smart-Its - Communication and Sensing Technology for UbiComp Environments*”. Technical Report ISSN 1432-7864 2003/2
- [25] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, H.-W. Gellersen. “*Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts*”. In Proceedings of Ubicomp 2001, Springer-Verlag LNCS 2201, pp. 116-122, 2001.
- [26] A. Schmidt, M. Strohbach, K. Van Laerhoven and H.-W. Gellersen. “*Ubiquitous interaction - Using surfaces in everyday environments as pointing devices*”. In Lecture Notes in Computer Science (LNCS), Volume 2615, N. Carbonell & C. Stephanidis (Eds.). Springer Verlag, 2002, pp. 263-279.
- [27] Lancaster Smart-its Documentation Homepage. <http://ubicomp.lancs.ac.uk/twiki>
- [28] S. Greenberg, and C. Fitchett. “*Phidgets: Incorporating Physical Devices into the Interface*”. In Proceedings of the Workshop on Building the Ubiquitous Computing User Experience. M. Newman, K. Edwards and J. Sedivy (Eds). Held at ACM CHI'01, Seattle, 2001.
- [29] S. Greenberg and M. Boyle. “*Customizable physical interfaces for interacting with conventional applications*”. In Proceedings of the UIST 2002 15th Annual ACM Symposium on User Interface Software and Technology, ACM Press. April, 2002
- [30] Phidgets, Inc. Homepage. <http://www.phidgets.com>
- [31] S. McPhail. “*Buddy Bugs: A Physical User Interface for Windows® Instant Messenger*”. Western Computer Graphics Symposium (Skigraph'02), March 2002.
- [32] <http://www.makingthings.com/teleo.htm>
- [33] [http://www.makingthings.com/products/documentation/teleo\\_multi\\_io\\_user\\_guide/index.html](http://www.makingthings.com/products/documentation/teleo_multi_io_user_guide/index.html)

- [34] <http://www.makingthings.com/teleo/customers.htm>
- [35] <http://www.arduino.cc/en/Reference/HomePage>
- [36] <http://www.arduino.cc/en/Main/Hardware>
- [37] <http://www.epa.gov/epaoswer/education/pdfs/life-cell.pdf>
- [38] <http://www.standardics.philips.com/support/i2c/usage/>
- [39] M. Weiser. “*The Computer for the 21st Century*. In *Scientific American*”, pp. 94  
September 1991
- [40] Brygg Ullmer et al, “*Core Tangibles: Modular Interaction Devices for Common Operations across Diverse Applications*”
- [41] <http://research.microsoft.com/sds/>

## Appendix- A: **Specifications of Microchip PIC 16F876A**

- Analog Features: 10-bit, up to 8 channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with: - Two analog comparators - Programmable on-chip voltage reference (VREF) module
- 28/40-Pin Enhanced FLASH Microcontrollers
- High Performance RISC CPU
- Operating speed: DC - 20 MHz clock input DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory, Up to 368 x 8 bytes of Data Memory (RAM)
- Up to 256 x 8 bytes of EEPROM Data Memory ·· 100,000 erase/write cycle
- Enhanced FLASH program memory
- In-Circuit Serial Programming TM (ICSPTM) via two pins
- Single supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Power saving SLEEP mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins
- Low power consumption

2001 Microchip Technology Inc. [www.microchip.com](http://www.microchip.com)

## Appendix- B: Specifications of Motes USB Module

Uses FTDI 232BM chip, key features are

- Single Chip USB <=> parallel FIFO bi-directional data transfer
- Data transfer rate of up to 1M Byte/second (D2XX drivers)
- Data transfer rate of up to 300K Byte/second (VCP drivers)
- Simple to interface to MCU/PLD/FPGA logic with a 4-wire handshaking interface
- Entire USB protocol handled on-chip: no USB-specific firmware programming required
- FTDI's royalty-free VCP and D2XX drivers eliminate the requirement for USB driver development in most cases
- 384 Byte FIFO Tx buffer/128 Byte FIFO Rx buffer for high data throughput
- New send immediate support via SI pin for optimised data throughput
- Support for USB suspend/resume through PWREN# and WAKEUP pins
- Support for high power USB bus powered devices through PWREN# pin
- Adjustable Rx buffer timeout
- Built-in support for event characters
- Integrated level converter on FIFO and control signals for interfacing to 5V and 3.3V logic
- Integrated 3.3V regulator for USB IO
- Integrated Power-On-Reset circuit
- Integrated 6MHz - 48Mhz clock multiplier PLL
- USB Bulk or Isochronous data transfer modes

- New Bit-Bang Mode allows the data bus to be used as an 8 bit general purpose IO port without the need for MCU or other support logic
- 4.35V to 5.25V single supply operation
- UHC/OHCI/EHCI host controller compatible
- USB 1.1 and USB 2.0 compatible
- USB VID, PID , serial number and product description strings in external EEPROM
- EEPROM programmable on-board via USB
- Compact 32-LD LQFP or QFN-32 package
- Available as a Lead-free device (FT245BL and FT245BQ) compliant to EU directive 2002/95/EG RoHS

[www.ftdichip.org](http://www.ftdichip.org)

## Appendix- C: Specifications of BlueSMiRF Bluetooth Modem

- FCC Approved Class 1 Bluetooth Radio Modem
- Extremely small radio - 0.15x0.6x1.9"
- Very robust link both in integrity and transmission distance (100m) - no more buffer overruns!
- Low power consumption : 25mA avg
- Hardy frequency hopping scheme - operates in harsh RF environments like WiFi, 802.11g, and Zigbee
- Encrypted connection
- Frequency: 2.4~2.524 GHz
- Operating Voltage: 3V-6V
- Serial communications: 2400-115200bps
- Operating Temperature: -40 ~ +70C
- Built-in antenna

[www.sparkfun.com](http://www.sparkfun.com)

## Appendix- D: Specifications of Gumstix

- Processor: Intel XScale® PXA255 200MHz and 400 MHz
- Memory: 64MB SDRAM 16MB Strataflash - xtended memory
- Storage: Via Type II compact flash on cfstix
- Operating system: u-boot - bootloader , Linux kernel - 2.6,
- Userspace includes: sshd, apache bluetooth utilities and more.
- Connector: 60-pin Hirose I/O header and 92-pin bus header
- Wireless: WiFi available through expansion card cfstix, Infineon Bluetooth™  
Module
- Networking: Bluetooth, usbnet, PPP
- Power management: Draws <250 mA at 400MHz without Bluetooth
- Sleep mode: Draws < 50 mA while waiting for input (not even sleep mode)
- Power requirements: 3.4V - 5.2V takes Li-Ion, Li-Polymer, 3-NiMH, standard  
4.5V or 5.0V inputs
- Operating temperatures: Commercial temperatures (est'd: 5C to 30C )
- Dimensions: 80mm x 20mm x 6.3mm

[www.gumstix.org](http://www.gumstix.org)

## **Vita**

The author, Karun Kumar Kallakuri was born in August, 1983, in Andhra Pradesh, India. He completed his high school from SBOA, Chennai, India, in 2000. In the same year he attended SMIT College, Chennai, India, with a major in electronics and communications engineering. He received his Bachelor of Engineering degree from Madras University, Chennai, India, in 2004. He joined for the master's program at Louisiana State University, Baton Rouge in fall 2004. He is expected to graduate with the degree Master of Science degree in Electrical Engineering in fall 2006.