Title	pushPin : programming stimulus-response coordination among everyday objects using paired physical tags
Sub Title	
Author	Fernando, Charith Lasantha(Inami, Masahiko) 稲見, 昌彦
Publisher	慶應義塾大学大学院メディアデザイン研究科
Publication year	2010
Jtitle	
JaLC DOI	
Abstract	
Notes	修士学位論文. 2010年度メディアデザイン学 第55号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40001001-00002010-0055

Master's Thesis

pushPin

Programming Stimulus-response Coordination Among Everyday Objects Using Paired Physical Tags

by

Charith Lasantha Fernando

Submitted to the Graduate School of Media Design
in partial fulfillment of the requirements for the degree of

MASTER OF MEDIA DESIGN

at the

KEIO UNIVERSITY

Academic Year 2010

pushPin

Programming Stimulus-response Coordination Among Everyday Objects Using Paired Physical Tags

by

Charith Lasantha Fernando

B.Sc., The University of Moratuwa. Sri Lanka, 2007

Submitted to the Graduate School of Media Design
in partial fulfillment of the requirements for the degree of

MASTER OF MEDIA DESIGN

at the

KEIO UNIVERSITY

Academic Year 2010

Certified by .....
Professor. Masahiko Inami
Professor, Graduate School of Media Design
Thesis Supervisor

Certified by .....
Asst.Professor Maki Sugimoto
Asst.Professor, Graduate School of Media Design
Thesis Co-Supervisor

Accepted by .....
Professor Masa Inakage
Professor and Dean, Graduate School of Media Design

pushPin

Programming Stimulus-response Coordination Among Everyday Objects Using Paired Physical Tags

by

Charith Lasantha Fernando

Submitted to the Graduate School of Media Design
on August 26, 2010, in partial fulfillment of the
requirements for the degree of
Master of Media Design

Abstract

With the development of inexpensive embedded processing technologies, consumer electronics are getting vivid and smarter than before. Consumers mind sometimes lead to think about connecting devices together to perform a continuous task in their lifestyle. A programmable alarm clock and a human detection module can not only trigger some of your daily routines such as lights, water heater, Washing machine, but also can save energy by turning OFF the devices when you are not present. This smartness can be a benefit to ordinary users only if they can connect, configure and reconfigure everyday objects according to their lifestyle needs with minimum effort. Commercially available systems such as *Insteon, X.10* have step in the direction of "smart homes" provides integration of several sensors, actuators and customize the device pairing at the installation stage, but not allowing rapid changes to configuration. Furthermore a re-configuration of existing system will require a significant amount of time. With busy lifestyle, we sometimes see existing home automation systems are sometimes not utilized well in consumers everyday life. Graphical, direct manipulation user interfaces are available for making devices program rapidly, but it is still easier to manipulate physical objects in the real world than it is to interact with virtual objects inside a computer screen. Tangible or graspable user interfaces help bridge the gap between the virtual world and the physical world by allowing us to manipulate digital information directly with our hands. "pushPin Programming" metaphor, an extension of Tangible user interface which resembles the traditional method of connecting devices using wires. We have made wires wireless to reduce the tangling and complexity of having meshes, pairing them using coloured/iconed pins just representing the end of the wires. We provide easy to understand real-time program/reprogram, debugging features built onto the system to help users program their everyday objects without any prior programming knowledge. In this thesis I want to show that pushPin programming is easy to learn, understand, remember, with minimum instructions so that everyday users can easily integrate objects in to their schedule.

Keywords: tangible user interfaces, ubiquitous computing, everyday objects networking, end-user programming

Thesis Advisor: Professor. Masahiko Inami
Thesis Co-Advisor: Asst.Professor Maki Sugimoto

Table of Contents

Appendices

List of Tables

List of Figures

Acknowledgements

It's really hard to mention names when it comes to Acknowledgements. I would like to convey my sincere gratitude for all who helped me throughout the project and make it a success. This project could not have been successful without the support of my friends and colleagues at the Keio Media Design, the insightful suggestions from my advisors Professor. Masahiko Inami and Asst. Professor. Maki Sugimoto for their continuous support.

Also i would like to thank Professor. Takeo Igarashi, **JST ERATO** design Interface group leader, who was guiding me with valuable comments.

Last-but-not-least, i would like to thank my colleague in KMD, Hirokazu Kawana, for spending his valuable time with me to make the Pins more beautiful.

This project is Funded by JST ERATO Grant. *http://www.designinterface.jp*

Part I

Thesis

Chapter 1

Introduction

1.1 Programming in Living Space

Mark Weiser's vision about ubiquitous computing [39] explains computers seamlessly integrated into our everyday environment. In terms of ubiquitous living space, it's common that objects having small invisible computers to provide various tasks and supports our daily activities. If a machine is to serve one distinct role, it can be hidden and of no concerns. With the technological advancement of 21st century, these invisible computers are growing in everyday space and offer more customizable options to users, with the aim of providing better services. In users perspective, having more options forces them to manipulate directly or indirectly, which sometimes leads to confusion.

Consumers sometimes try to sequence some events such as air conditioner to OFF at mid sleep and turn ON the air conditioner before wake up. This is currently being done by setting the timer feature in air conditioner. In the case of more than one device in sequence it is hard to schedule. Having lot of small objects in the living space results in configuring all of them separately one by one. If the user has to do the individual settings only one time, it's acceptable. But in the case of most common tasks performed in everyday life (ex: boil water, make coffee, wake up alarm) performs as a routine tasks has to perform everyday, leads for a repeated programming on a everyday basis. For most of the users this is not acceptable due to their busy schedules and getting bored doing the same thing over and over again. With the today's computing environment, if consumers can make connections between devices and perform programming, it will be the ultimate solution for the living space scheduling and automation.

Programming can be applied indirectly to living space and provide real-time customization. Traditional text based programming meant for professional programmers is not suitable for a living space with ordinary users. Graphical User Interfaces (GUI) based programming languages such as Matlab [9] is ideal for Academic purposes and LEGO Mindstorm [8] is for kids, but not suited for everyday users. Direct manipulation of everyday objects using tangible techniques is the best way to grasp the controllability easily compared to virtual objects being controlled inside a computer screen. Tangible programming allows users to feel digital information on hand and manipulate them to access features of the everyday objects.

1.1.1 Programming accessibility to everyone

Programming in the broad sense is no longer for professional programmers. Cashiers use cash registers to perform calculations and issue an invoice. Accountants use electronic spreadsheets to build business documents. Musicians program their MIDI synthesizers to custom adjust their sound in performances. Kids use LEGO Blocks to make models and give dynamics by downloading programs from the LEGO Mindstorm [8] software platform. In a global scale, every consumer electronics can be considered as everyday objects. To make connections between these objects we should have a common language. In this research i propose an easy to remember, understand tangible programming language for everyday objects. My goal is to find out an easy to understand common physical programming method which can be benefited to both busy consumers and ordinary everyday users.

1.1.2 Automation and Scheduling

Automation is the use of control systems and information technologies reducing the need for human intervention. In the scope of industrialization, automation is a step beyond mechanization, whereas mechanization provided human operators with machinery to assist them with the muscular requirements of work. Production lines (as Shown in Figure 1.1) uses heavy automation of machinery to custom cater their product line. [1] These systems are mainly pre programmed using PLC's[2] which runs Ladder Programs. Such kind of control systems and its' programming can only be performed by professional programmers and requires much time to adopt a new change into the system.

Scheduling is the process of deciding how to commit resources between a variety of possible tasks. Time can be specified (scheduling the coffee machine to run at 7:00a.m) or floating (if wake up, run the coffee machine) as part of a sequence of events.

Scheduling and Automation both can be seen in everyday life. Thermostats in Air Conditioners can be used to cut the power off when it reach the desired cooling temperature. A microwave oven will beep when the food is ready to eat. By introducing programming in to scheduling and automation, it greatly reduces the need for human sensory and mental requirements to perform tasks and plays an increasingly important role in daily experience.

[1] $http://www.flickr.com/photos/harry_nl/3030267182/$ Under Creative Commons License
[2] Programmable Logic Controllers

Figure 1.1: A fully automated automobile Bodyshop

Figure 1.2: A modern air conditioner remote controller

1.1.3 User interfaces of everyday objects

With the inexpensive technologies available, digital controls can be widely seen on consumer electronics front panel which maps the user input to the analog functions. The embedded microprocessors allows designers to decouple the functions from the controls, and even worse, allowed them to have fewer controls than functions. For example a modern air conditioner will only have a button for ON/OFF at the front panel. Setting the timer, temperature, louver adjustment, fan speed etc... which does not require regularly are accessed through it's remote controller (Figure 1.2. Typically home air conditioner is set for a comfortable temperature (ex: 20°C) at first time, and it will be saved internally, and there after user will be just turning ON and OFF. Thus the designers have reduced the amount of time and complexity taken to use their products in a more convenient way.

Figure 1.3: Professional DJ console

In contrast a Professional DJ console (Figure 1.3 cannot made to be less complex, because the professionals need all the virtual controls mapped into the physical controller. But controlling these DJ consoles requires special training and prior knowledge on how to use them.

These two extremes shows that for ordinary users, not requires much controls, but for professionals doing specialized tasks the full control is essential. A typical DVD Player might has many buttons to control it's various functions. Imagine a situation of schedule a favorite TV program recording to a DVD and it might require to press many buttons before it starts recording. Even though these systems are smart, still it becomes difficult when it comes to just perform a simple task. My research is to introduce an easy to understand programming language for everyday objects which can perform scheduling

and automation in a networked environment.

1.2 Overview

pushPin system is a common physical programming tool for connecting and programming everyday objects. The inter-connections uses the pushPin metaphor which explains by virtual wires to interconnect many objects. The virtual wires are made of colored Pin Pairs acting as different subnets in the entire network. It not only establishes a connection directly but also indicates the connection directly, so that user can easily tell which devices are connected. Closing or changing a connection is also done by an intuitive physical action, the result of which is clearly evident. In addition it provides an easy to understand multi-colored indicator based debugging features as well.

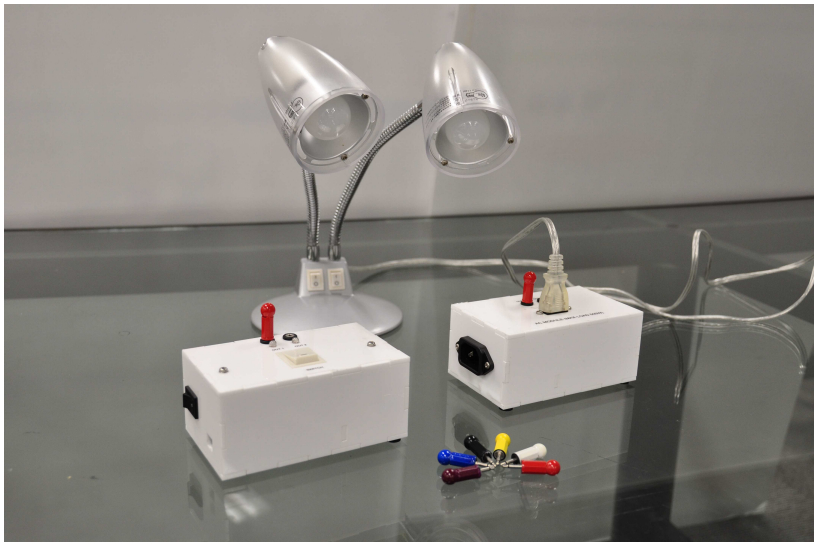

Figure 1.4: Connecting a Switch with AC light source using pushPin

To the end user, it offers very simple instructions to pair everyday objects and program sequential or concurrent tasks seamlessly. All the objects will have either inputs, outputs or inputs and outputs. End user can use same colored pins to network devices by connecting an output of one device to input of another. To connect a Switch Module with a AC light source (As shown in Figure 1.4), user can choose 2 pins of same color and put one on the output of Switch Module, and other on the input of AC light source. The inputs can be analog, digital, sensors or even computer nodes. Outputs can be a series of everyday objects. Absolute timing triggers can be generated using clock modules and relative timing triggers are generated using timers which will be described later in the document.

1.3 Organization of the thesis

This thesis is divided into 7 Chapters. Chapter 1 has presented a set of problems that have motivated my research and basic overview of the system. Chapter 2 gives the context and scope of the project, introduction to virtual wires, related works and the technologies that i have used in my design in development of the pushPin programming system. Chapter 3 provides the detailed description of the system design describing how the initial prototype helped to modify the pin identification, debugging features, and reliable networking. Chapter 4 describes the detailed Implementation plan of the final pushPin system clearly stating the changes from the prototype version, overall system block diagram, firmware model, and software applications that I made for evaluating the system. Chapter 5 offers the evaluation of the system including technical evaluation results followed by a user study. Chapter 6 is a in-depth discussion on the overall system based on the user study results, advantages and disadvantages of the system, possible improvements to the system and future plans. Chapter 7, concludes the thesis. Appendix A includes schematics, drawings and artworks of the final design.

Chapter 2

Motivation and Context

2.1 Scope: What is programming?

In my thesis I have define programming as a very broad topic. It ranges from complex to simple and everyday. Commercial softwares written by Professional programmers can take years to write in span thousands of pages. Programs written by kids are typically less than a page, and even those few codes can produce their creativity and what they want to achieve.

To consumers, the term "programming" often means something more basic like configuring a device to work according to their needs. In fact programming in small scale really means configuring, tailoring or personalizing. These activities often have to do little with time. For example, A modern wrist watch will require the user to set the time on the first use by pressing few buttons. But this activity is only done once every few years or never, so it is typically performed using hand manually. Another such example is setting the Air conditioner's temperature profile. This is done typically once a month or a week when the owner feels the temperature is not so comfortable, so it is typically performed by manually using the remote controller buttons. A housewife cooking everyday at home might need to use the microwave several times per day. Every time she uses it, has to program the Power or time setting according to what is being cooked. It is programs of this scale that this thesis focuses upon. I make no claims that my techniques are ready to compete with, for example complex programs written in "Microsoft C++" [15] or "Xcode" [18], but do not consider this as a shortcoming, as even simple programs are useful in everyday context to make life much easier.

2.2 What is programming with virtual wires?

The term "Virtual wires" was first introduced by Ayatsuka et. al. to describe their tranSticks [20] approach of connecting computers together. The novelty of tranSticks sets apart from other connection techniques where it propose this labeled ends to be thought of as wires physically separated, but virtually connected to each other. Virtually connected wires can not only replace a physical cable but also provide a connection between remote places and eliminates the unwanted, long, tangled, cables which can be seen typically when lot of wires in one place.

2.3 Why virtual wires?

There has been work on connection establishment using intuitive methods [19, 22, 28, 30, 31, 36]. However, from a connection establishment point of view physical cables has several advantages. When two devices are connected wired, it's not only the connection establishment, but also the connection path is directly indicated. This helps sometimes in debugging connection errors. For example, to connect an iPod in to stereo Hi-Fi system[1], a cable with ϕ3.5mm stereo Jack on one side and RCA[2] connectors on other side would be sufficient. Most consumers, by common sense knows how to wire this scale of simple circuits.

When connecting using physical wires, in fact we do mostly care about the start and end points of a wire and we do not pay attention on the intermediate wires. As shown in Figure 2.1 by labeling or coloring the end of the cables, it is much easier to understand the end points of a cable.

We propose such labeled ends virtually connected, physically separated to each other not only allows the data to transfer between the links bi-directionally, but also limitless links unbounded by the length. This virtual wires metaphor is named as "pushPin".

2.3.1 Reduce complexity

Having a bunch of physical wires to establish several connections sometimes leads to confusion due to tangling of cables and difficulty in seeing the end of the cables. With pushPins connected physically apart, carries no physical wires, helps to zero the tangling and provide better visualization of the connected pairs through colors.

2.3.2 Different types of data

For different purposes we use different types of wires. For example to connect a PC to a projector we use a D-Sub15 component video (RGB) cable. But to connect a Digital camera to a PC we will have to use a USB[3] A to mini-B type connector. In general having different types of input/output interfaces force you to use these specialized cables, and makes the ordinary user more confused. With pushPin, we do only care about the start and destination. More technically it's the container to transport data. With respect to container, the data can be of any means. It can be analog, digital, ip, etc... This provides an easy to understand connection establishment mechanism compared to physical wires.

[1] High Fidelity
[2] Audio/Video Connection Cable which initially invented by Recording company of America
[3] Universal Serial bus

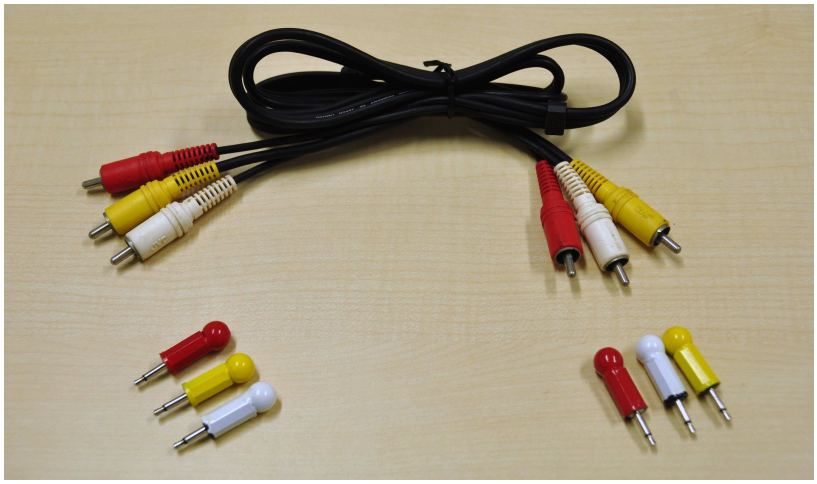

Figure 2.1: RCA cables alternative virtual representation

2.3.3 Debugging

Establishing connections not always works on first try. Sometimes the users have to debug for any connection errors. The simplest debugging we might consider is swapping the error containing wire pair with a working pair. Again this operation requires finding the ends of the cable. With pushPin system, broken wires scenario is inherently removed as the real connection is based on a reliable wireless channel. But in case of a advanced debugging scenario which can think of, dealt with easily swapping the pins representing the ends of the virtual cables.

2.3.4 Reversibility

pushPin system operations are fully reversible. A connection established by a pair of pushPin on two different devices can be break apart by removing either one end pin. A connection from device A to B can be easily re-programmed to device C by just replacing the pin on B, putting on device C. Similarly if another pin on the same color placed on device B works as a one-to-many network, controlling multiple connections from a single entry point.

2.4 Related Work

My work is mostly related to direct manipulation techniques [29, 32], tangible user interfaces [21, 23, 35, 37], and end-user programming [33, 34]. The focus on thesis is about end user programming method for everyday users. There has been quite a few commercially available end user automation gadgets for smart homes. [6, 7, 16]. Within the same context I also looked into networking objects using pins and found Pin&Play [25], lately continued as VoodooIO [38] system. My work follows mainly the

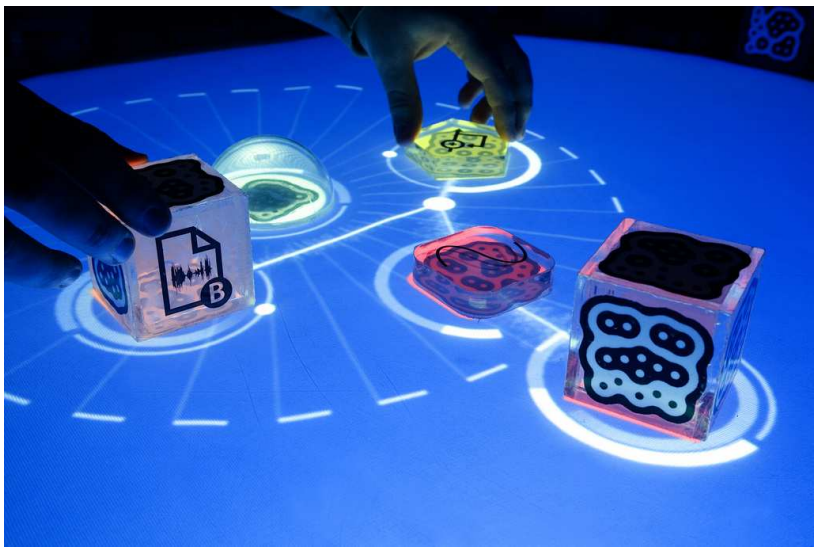

Figure 2.2: Reactable: live music performance and tabletop tangible interfaces

research based on Sony CSL labs by Ayatsuka et. al [20] which shares the same concept of virtual wires being used to associate objects.

2.4.1 Tangible user interfaces

When GUI systems first introduced, the amount of interaction to physical objects reduced considerately. Then, researches found a way to combine the virtual objects inside a computer screen to physical objects which now we call it as tangible or graspable interfaces. Ishii et. al. defines "tangible bits" as "Tangible bits allows users to grasp and manipulate bits in the center of users' attention by coupling the bits with everyday physical objects and architectural surfaces" [23]. In the context of my research, objects can be considered as the bits introduced by Ishii. The metaphor "tangible bits" has been used by Sergi et. al. in "Reactable" a musical instrument based on a tabletop interface [24] that exemplifies the tangibility (Figure 2.2 [4]) in real life together with music.

There has been several research related to tangible programming which encouraged me to focus on the path of tangible programming in order to provide a more easy to understandable method. Algoblocks [35], system which invented by Suzuki et. al. has been used to study collaborative learning of programming concepts among children. The system consist of hand sized blocks which equivalent to one simple action, for example "go forward", "turn left". In "Tangible Programming Bricks" [26] Timothy McNerney makes a tangible physical programming interface specially for kids at a size of a LEGO block.

[4] *http://www.flickr.com/photos/arselectronica/4361758807/* Under Creative Commons License

2.4.2 Direct manipulation

Similar to tangible interfaces, direct manipulation of objects also very important fact when considering the 'programming for everyone' metaphor. Even though if you can touch and control virtual objects, if there is no direct manipulation from the input, user sometimes gets confused about what and how to control the specific object. With a direct manipulation technique, the advantage is that we map the physics to the action required for programming. For example, if you need to rotate a knob, you can just do the same to the tangible interface. The output will reflect in GUI or Physical device. Rekimoto et. al. demonstrated a system called, Pick-and-drop [29] used in a networked computer environment to pick up an object on a display and drop it on another display as if she were manipulating a physical object. pushPin also inspired by this idea because this shows the connection path is really not useful, but what matters is the start point, end point and natural gestures on how to connect devices.
A more recent work from MIT media lab, David Merrill made a small ubiquitous widget called "Siftables" [27]. Siftables aims to enable people to interact with information and media in physical, natural ways that approach interactions with physical objects in our everyday lives.

2.4.3 Networking objects through Pins

The most relevant related research topic is networking objects through pins as pushPin metaphor shares the virtual wires concept. tranSticks system [20] by Ayatsuka used the virtual wires concept of linking two or more computers in a network using a common digital ID. They used the media as a SD card, where the ID is written on to the card and labeled in colors to recognize the pairs. My research is a continuation of this former research, focusing mainly on how virtual wires can be used in everyday space to program objects. Even though the system looks quite similar, there are several difference. tranSticks is a passive system based on computers and physical networks such as LAN. Conversely pushPin is a base platform providing a active service with its own network. When designing programming for everyday users, hiding the computers in to background helps them to grasp the services more easily. Also tranSticks does not provide advanced programming techniques such as loops, timers, logic or even cascade networking, thus it is more a connection establishment mechanism. Since it is using computers, It is difficult to combine with everyday objects, rather it can only provide virtual connection between computer peripherals. Last but least since it's active mechanism and small, easy to integrate into the living space.

Pin&Play [25] by Laernhoven et. al. consist of small objects which can be tide with sensors or actuators and connected to a special made prototype board surface. The special surface is made such a way that it consists 3 layers of conductive yarn loomed fabrics

wafered together which gives access to 3 different signalling layers when connected. The novelty stands from the point that these layers are used to provide Power, Ground and Signal bus to objects using a special pin design. The bus for signal transmission, receiving is a Dallas 1-wire technology. In this research it explains how flexible users can perform programming because when connecting to the prototype boards, users do not have to think about the orientation or the absolute position of the blocks. The next version of this research has been carried out as VoodooIO [38], not limiting the surface as table tops, but to any vertical surfaces (ex: walls). The authors have made several applications such as gaming, hardware prototyping, cockpit and even flight simulators etc... to test the freedom index when users are customizing their control space.

This inspired me to use a wireless, small packaging for the pushPin so that consumers can experience the same freedom of putting these blocks in anywhere they wanted and perform the programming.

2.4.4 Commercially available home automation systems

There are several companies providing home automation technologies for smart homes. Namely KNX [7], BACnet [2], and LonWorks [4], provide their custom protocols, for industrial use with higher bandwidths, but still INSTEON [6] and X10 [16] remains popular in home environment with millions of units sold worldwide, and inexpensive availability of new components provided by several leading companies in the world. X10 is an international and open industry standard for communication among electronic devices used for home automation. It primarily uses power line wiring for signaling and control, where the signals involve brief radio frequency bursts representing digital information. //
INSTEON [6] initially used a RF based connection establishment, but because of the large boom in X.10, INSTEON started to give backward compatibility for X.10 in their products. These commercially available home automation systems can be connected to home servers, giving access to your home control from anywhere in the world. As shown in Figure 2.3 they are available in packages to fit your home needs. But one of the problems in these systems are, it requires some professional help in configuring, wiring for the first time. Figure 2.4 shows the bare wires of the device which needs to connect to wall outlet at the beginning. Another major disadvantage is these networks depends on the utility power and it's specifications. For example, neither INSTEON nor X10 does not work in Japan domestic electrical standards (i.e 100v, 50Hz) and it only works with the US domestic electrical standards. (i.e 120V, 60Hz). The company has no near future plans to provide these units for outside US market.

After the installation, it's flexible to use either in physical space or web based control. If the user plans to re-configure some connection pair, he first need to detach the input

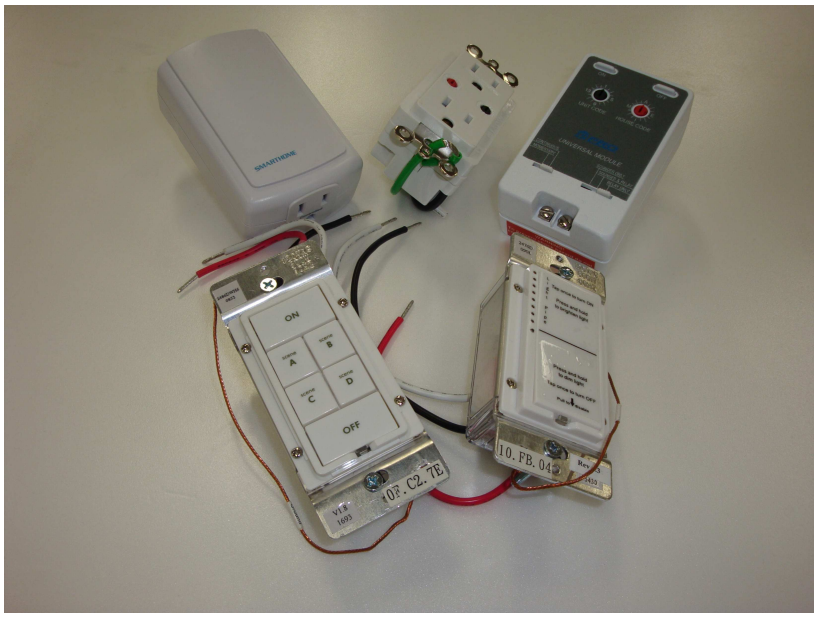

Figure 2.3: Insteon: Typical kit for light dimming

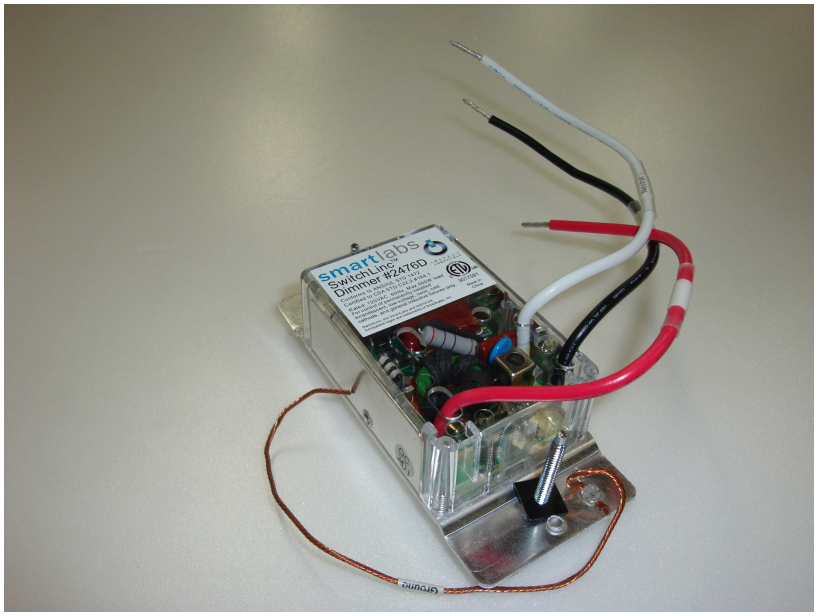

Figure 2.4: Insteon: back side complex wires

and outputs from it's previous assignment. Visualizing the earlier configuration in these systems are very difficult, in fact user needs to do trial and error by manually switching. Of course the web interface allows users to virtually visualize the configurations, but not all users can do this and even so it's more time consuming compared to a physical visualization interface.

2.5 Novelty of pushPin

Considering the above related works, pushPin system can raise in to the novelty of direct manipulation of everyday objects through virtual connections to provide simple network programming features for everyone. There virtual connection networking works exist, but not providing programming. Also direct manipulation techniques for physical programming exist, but the scope is for either kids programming or for academic related people who wants to learn programming. It is not in a reach for everyday users. Furthermore, pushPin is re configurable, not requiring special skills for installation and safety are additional benefits for the consumers.

Chapter 3

Design

3.1 Functionality design

The functionality design of pusnPin was carefully thought to provide all requirements for consumers and thinking about user skills. By networking devices, the primary goal was to access any node from any other node, service discovery and broadcasting. These functions were already inherited and defined in RFC0791, "Internet Protocol" [5] which is widely used in all computer networks. But this protocol is very heavy to use with embedded platforms due processing power limitations. Thus we decided to scale down the design to fit our requirements. pushPin network uses broadcast packets with designated receiver id as a group to reach all the devices connected to the network. It also provides individual addressing and it is being used in connection established notifications.

Below is the detailed design of each section.

3.1.1 Digital pins

Figure 3.1: ϕ2.5mm Mono Audio Jack

There were number of considerations when designing pins. The pins should be small enough to easily grab from fingers. Then the pins would be passive in order to reduce the size, complexity and price so that in case of a misplacement it can be easily replaced by another. Finally the design would not include complex wiring or connectors. Considering the above facts, the design included a ϕ2.5mm mono audio jack (Figure 3.1). The jack only has 2 connections to the internal circuit.

First prototype was done using a resistor divider used as the pin id. One resistor R_1) inside the circuit connecting the other resistor at pin (R_2). As shown in Figure 3.2, equation 3.1 calculates V_{out} for different R_2's

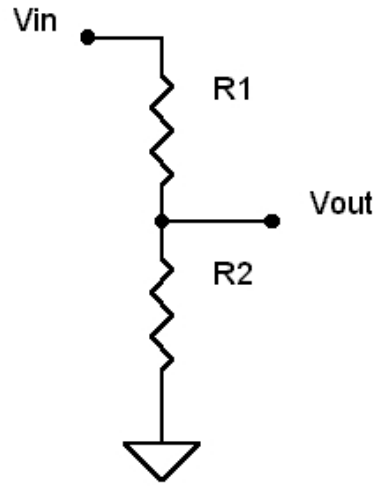

Figure 3.2: Resistor divider schematic

$$V_{out} = V_{in} \times (R_2)/(R_1 + R_2) \tag{3.1}$$

Different ID's can be generated from getting different voltage levels and converted to digital representation using an analog to digital conversion process.

3.1.2 Analog and Digital data

The type of data which carries by the wireless network is important when considering about how the devices being associated. For example, just binary information would not sufficient to dim brightness of a light, changing the volume knob of a radio set etc... So we considered having 1-byte digital data (0 to 255) so that it can handle analog and digital data with 8-bit resolution.

3.1.3 Wireless mesh networking

The system should be able to handle hot swap connections. When a device appear online it should join the network, and should leave the network when it appears offline. We decided to use Zigbee mesh networking which already provide the necessary framework for hot swap connections.

3.1.4 Program debugging

Since it is a physical programming language, it is necessary to have some debug features to easily troubleshoot the system. We decided to provide 4 levels of debugging using RGB LED indicators.

1. Successful network join attempt on module power up (success: Green)

2. Successful Pin insertion on each pin slot (success: Blue)
3. Successful Pairing of two objects (success: Blue)
4. Successful Data transfer (success: Red)

3.1.5 Power supply /Batteries

For the devices connecting directly to utility power, they are powered by the same AC connection. Other modules which is mobile, are powered by LiPo 3.7v batteries which can run up to 10 hours at full charge. A USB mini-B to A cable can be used for charging.

3.1.6 Safety

When designing the modules, safety has been considered as a secondary requirement because everyday users should be able to program safely. Previous commercially available systems such as X.10 [16] does not provide any safety features, in fact the initial wiring has to be done with the help of a professional. In pushPin no external wiring required for setting up. Thus it eliminates directly contacting with utility power.

3.2 Exterior design

3.2.1 Colored pin pairs/groups

Figure 3.3: pushPin, different colors of pin groups

The color pins (Figure 3.3) were made out of ϕ2.5mm mono audio jack connectors. By modifying the original case, adding a ball and colored using 6 different paints it looks much cuter for everyday space.

3.2.2 Phicons

With the color pin design, user cannot understand the function of the task that is being performed by the pin specially the end device is not visible at the time of programming. For these types of cases, we created different shapes of "phicons" [1] matching the shapes of common devices such as floor lamp, ceiling lamp, circulator, TV set, audio player can be represent using pin icons. (Figure 3.4).

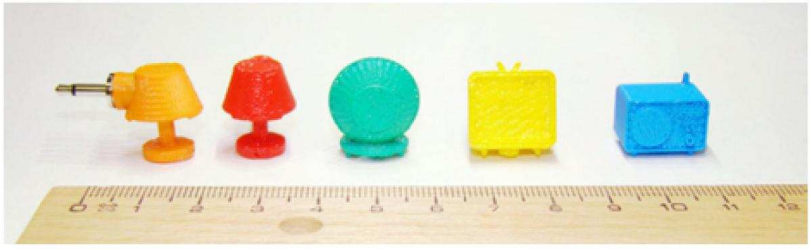

Figure 3.4: pushPin Phicons

These Phicons can be easily attach/detachable for the existing color pins so that it

[1] physical icons

provides visual bookmarks like option. For example user wants to put the PIR sensor out of the living room to detect if anyone comes closer. In this case it is not visible for the user when programming. It is obvious that if he is doing the programming on the same day. But in a weeks time if he wants to change the configuration, he might not remember which color was associated with PIR modules. Thus he can use a PIR shaped Phicon in the pin pair used to connect the module.
We envision that the size of the pin will be similar to that of real push pins. They will be neither too large nor too small but just perfect to manipulate by a human hand. Each pin has it's own ID shared with color groups and are passive electronics which does not require any onboard battery.

3.2.3 outer design

Maintaining a common standard is important for making the system easy to adoptable by any user. All the modules in the network has been designed to use one design. Different faceplates will give a different function for each modules.

Chapter 4

Implementation

4.1 Hardware Implementation

Since pushPin system (Figure 4.1 is a programming language for end users, the design adhere to the structure of a common programming language. In order to provide that, pushPin system contains a modular based approach. With a modular based approach, not only the design is clear to the end users, but also easy to adopt any future modifications. The modular based design contains 3 main types of blocks. i.e

- Input modules (Switches, Sliders, Sensors etc...)
- Intermediate modules (Timers, Delays, Logics etc...)
- Output modules (AC loads, DC loads, IR remote control bridge etc...)

By examining the system requirement iterations, final design was came up with a main controller board providing native platform features, plus additional daughter controller boards for special purpose modules.

Figure 4.1: pushPin system, final implementation

4.1.1 Digital Pins

First prototype of pushPin was designed according to resistor divider theory which explained in Chapter 3. But we experienced that voltage output of the resistor divider is not constant over time and temperature. Thus long time operations gave us undetectable pin or wrong pin detection. Also with resistor divider, having only a small voltage full span (3.3v) it was difficult to select different resistor values.

We decided to use Silicon Serial Number (SSN) technology where it can read a 48-bit factory-lasered unique ID for pin identification. The chip (DS2401) uses Maxim's One Wire technology [10] to communicate as well as to supply burst power so that it can work as a passive device requiring no power supply. The main disadvantage of the SSN technology used in our design was that, there was no way to interpret the ID's to common groups. We implemented and tested a lookup table mechanism on each module, where it search for the matching ROM ID and allocate the group ID. This requires us to run a search ROM function every time a pin is detected, and it was a big overhead to CPU specially when the lookup table was long. (Ex: more than 100) Thus, in later design it was migrated to 1-wire based EEPROM[1] technology (DS2431) where it can store, erase any ID via the 1-wire protocol. The technology used to access the SSN and 1-Wire EEPROM was same, but with different ROM command set, so we didn't have to change any hardware to adopt this change.

Moreover, with EEPROM based Digital pin it allows users to store data at the operation time. For example, a value related to dimming can be stored in the EEPROM and later being used to trigger an AC light only at that stored dimming level. In application, pins connected to a switch and a AC module will only light up in full brightness. With this concept of "physical macro", the users can store their favorites and personalize the event triggers.

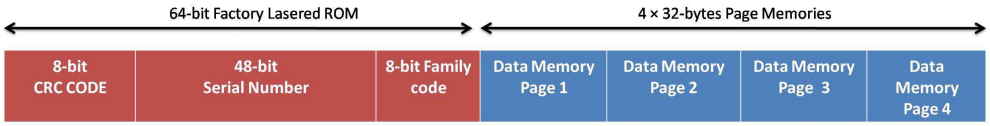

Figure 4.2: Digital Pin, Memory Map

1-Wire EEPROM based digital PIN also has factory lasered 64-bit unique identification. EEPROM memory consists of 4 memory pages, each having 32-bytes of storage capacity limiting the total for 128-bytes. In partitioning the memory map for pushPins, we allocated 8-bytes for unique identification. This space is equivalent for a 64-bit address space which gives almost unlimited programmable ID's. The rest of the 120-bytes space

[1] Electrically Erasable Programmable Read-Only Memory

is allocated for the physical macro feature. Memory map of Digital Pin is shown in Figure 4.2

4.1.2 Main controller board

Main controller board (Figure 4.3) includes a powerful Microchip PIC18F6622 [11] micro-controller which controls all the peripherals in the board.

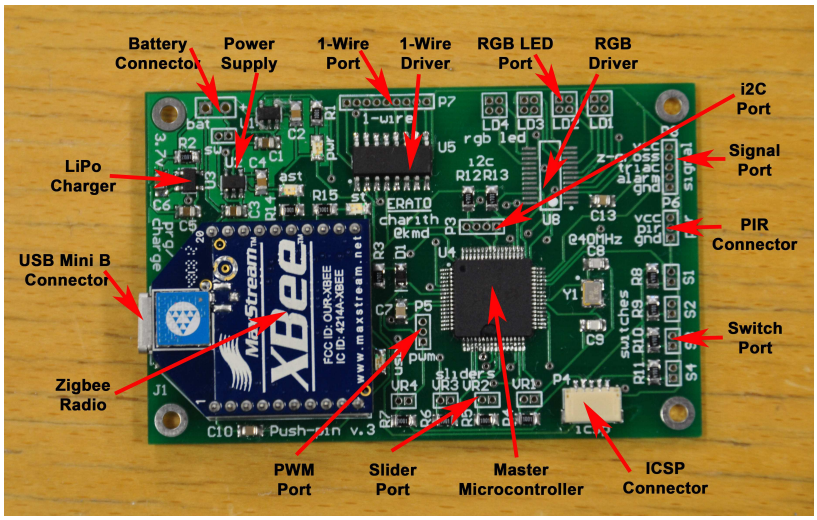

Figure 4.3: pushPin: Main Controller PCB

Main controller board is powered by a LiPo[2] 3.7v with a capacity of 1000mAh. Onboard low dropout linear voltage regulators (Texas Instruments TPS73633) [14] supply 3.3v to the main controllers and peripherals. A small LiPo charger is included in the main board to support recharging over USB cable. The Pins are connected using a 1-wire driver to minimize the timing constraints in 1-wire protocol. 4 × RGB LED's are controlled using a special RGB driver connected to micro-controller by i2c communication. Wireless communication is obtained by implementing Xbee wireless radio module [17] connected to primary RS-232 port. A secondary RS-232 connection is connected to a USB to UART bridge and its being used for debugging, firmware upgrading and for emulation. The same USB connection is used to Charge the LiPo batteries as well.

Additionally 2 × PWM ports and secondary i2c port have been included in order to provide flexible integration of other daughter cards at a future revision (currently not in use).

[2]Lithium Polymer

1-wire pin driver

First implementation of pin identification was based on explanation of Maxim's application note[3] about the serial interfacing 1-wire devices using precise timing. This was working quite well, except that the timing is critical for smooth operation. in the case of, master controller utilizing its processing power to some high priority parallel thread, the timing becomes little unstable and results in error reading the 1-wire device. This is not so critical for less priority threads, but pushPin system's AC light dimming requires a high priority thread for calculating the AC line zero crossing (executes at every 39.20us). the other disadvantage was, when multiple devices plugged on to the same 1-wire bus, at first master controller has to perform a search rom, which gives all factory lasored unique identification numbers, and then address different devices using unique ID. This methord was okay, if the system has static device connections. (i.e not plug and play insertion). But pushPin needs to use plug and play pin insertion to provide the flexibility of programming to end users.

Due to the above reasons, the design was changed to use a standalone 1-wire driver, DS2482-800 [3] which has 8 individual channels which can be switched internally and read contents directly. Each pin slot connects to different channels so that there will be no bus conflicts, having multiple 1-wire devices in same bus. This eliminates the reading of ROM's unique ID at first and allows directly reading the EEPROM contents. Writing a Pin's identification also shares the same steps.

RGB LED driver

RGB LED's were added to the design for the purpose of error reporting and visualizing. Number of LED's were depend on the maximum number of pin slots per module. In current design maximum slots were 4, thus we designed the system so that 4 RGB LED's can be controlled concurrently with custom brightness. This requires 3 Channels(R,G,B) $\times$ 4 LED's = 12 individually controllable PWM [4] channels. The current micro controller, PIC18F6622, nor any other micro controllers does not support 12 native PWM channels. One option was to time division the existing PWM channels and assign time slots, This implementation was carried out in a different project and the result was sometimes flickering LED's due to concurrent thread processing. So the final design included a standalone 17-channel PWM dimming driver, MAX6964 [1] which can control individual channel dimming by changing an internal register by an I2C interface.

pushPin design used a common Anode ϕ5mm type LED (OSTA71A1D-A), which of the forward voltage characteristics were as in Table 4.1. Furthermore, each channel maximum current was calculated as 30mA and limited using a series resistor at each

[3] http://www.maxim-ic.com/app-notes/index.mvp/id/74
[4] Pulse Width Modulation

LED	Forward Voltage(V_f)
Red	2.0v
Green	3.2v
Blue	3.2v

Table 4.1: RGB LED, forward voltages @I_f held constant 20mA

output channel.

Zigbee Wireless Mesh Networking

For the wireless mesh networking in pushPin, XB24-ACI-001 module [17] from Digi International is used. This Xbee module is equipped with a 1mW RF transiver and it's detectable range is up to 30m indoor(100m outdoor). Each Zibgee node was given a unique MYID and since pushPin system uses broadcast addressing, the Destination address LS-Byte[5] is set to 0xFFFF. As shown in Figure 4.4 there are some other custom settings have been used as well.

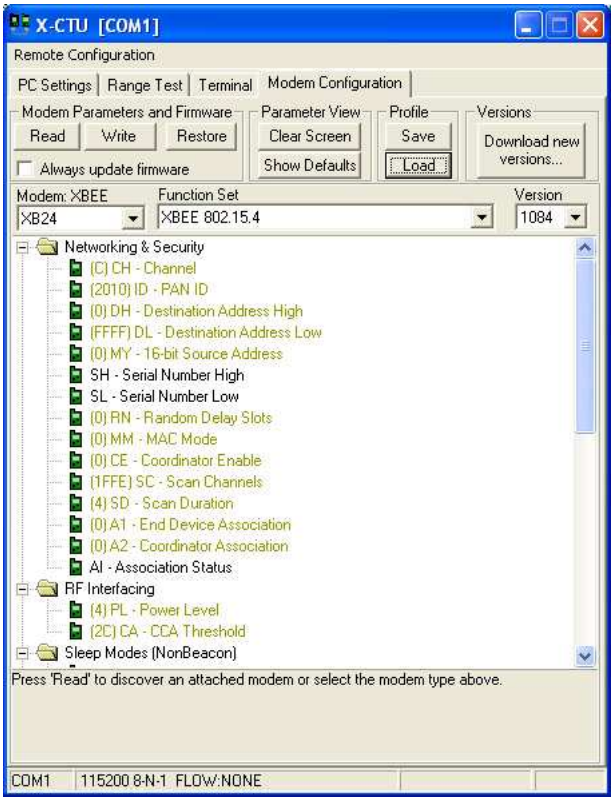

Figure 4.4: Xbee custom configuration for pushPin

[5]Least Significant Byte

USB UART connection

A secondary USB connection was implemented using FT232R [6], USB to UART Bridge IC for debugging, firmware upgrading and emulation purposes. The firmware includes a emulation menu which can be accessed via the USB serial connection at a rate of 115000bps. In this menu all the features of pushPin system can be emulated including AC light ON/OFF or dimming, RGB LED blinking Pin ID reading, writing etc...This emulation menu is not only useful for debugging and emulation, but also it is used for firmware upgrading.

Signal port

A dedicated signal port is included in designing the main board to connect to daughter boards such as AC dimmers, RTC[7] modules. It acts as a bi-directional bus port providing output ports for Triac Firing pulses, input ports for zero-cross detection pulses, alarm trigger pulses and a V_cc, GND bus for supplying power to the daughter boards in case of if it is not self powered.

Switch port

Switch port is connected to Port B from B4 to B7 utilizing the KeyBoard (KB) interrupts of the master PIC. All ports are externally pulled up to V_cc and weak internal pull ups have disabled.

Slider port

Slider inputs are connected to Port A from A0 to A3 utilizing the internal 10-bit A/D converter. Even though the design includes a 1k pull up resistor to V_cc, in actual dimmer module implementation we have eliminated the pull up and used the full scale of 10k potentiometer to get a more desirable linear scale.

PIR port

A special PIR port has been implemented to connect the Panasonic MP motion sensor [12]. Being trigger pulse less than 100uA, it is not possible to directly connect to a micro controller. Thus, we have designed a small external circuit acting as a current amplifier with a general purpose NPN Transistor (2SC1015). Amplified signal is directly fed in to the PIR port of the master board.

4.1.3 AC dimming

Dimming a Load is simple when the controls are analog. for example, in commercial dimmers the dimming is archived using a TRIAC and a DIAC providing the synchronous

[6] http://www.ftdichip.com/Products/FT232R.htm
[7] Real Time Clock

firing angle pulsating to every cycle in AC wave. In digital control, the problem is how to do the synchronization because the controller is DC and TRIAC uses AC wave. In pushPin design the AC wave is continuously monitored for zero-crossing and the interrupts were used to synchronize the TRAIC's triggering pulses.

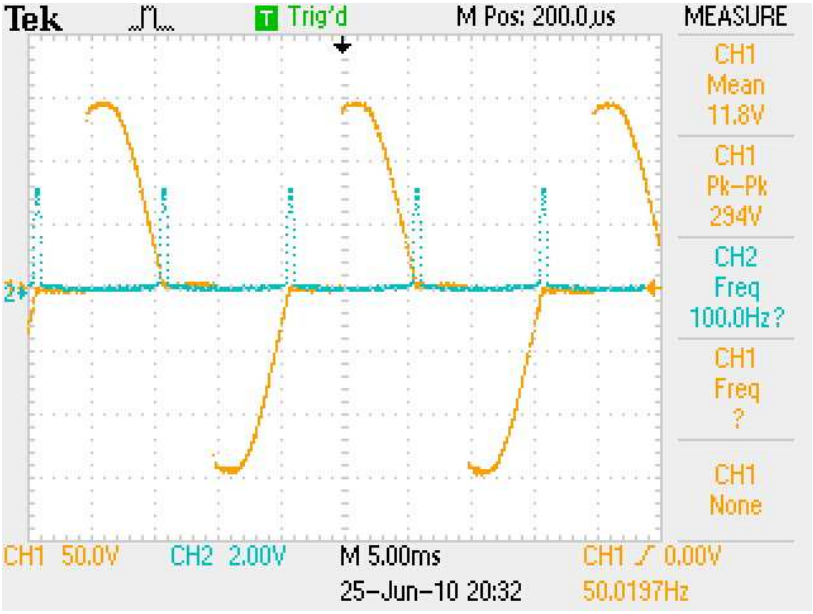

Figure 4.5: Zero Cross triggers

In Figure 4.5, blue trace indicates the zero cross and orange trace indicates the AC wave. As can be seen, at every zero crossing of the AC waveform, the trigger pulse is generated and sent to the master controller.

pushPin system provides 255 levels of dimming by dividing a 50Hz (in case of Tokyo, Japan) half cycle (i.e 10ms) into 255 time divisions (each 39.2us) and firing on specific divisions. For example firing the TRIAC at division 1 provides full brightness, where was firing at 255 provides null brightness (OFF). Figure 4.6 shows the synchronized TRIAC pulses firing at every half cycle to obtain a 75% (i.e 192/255) brightness level.

4.1.4 Real time clock

A real time clock is useful when the absolute time requires in a embedded platform. In pushPin the timer/delay module requires a time reference. If the time is calculated by counting cycles it can be a overhead to the micro controller. At the same time this relative time can be un available if the module restarts or in case of a low battery shutdown. To address these problems, we implemented a Real Time Clock with battery

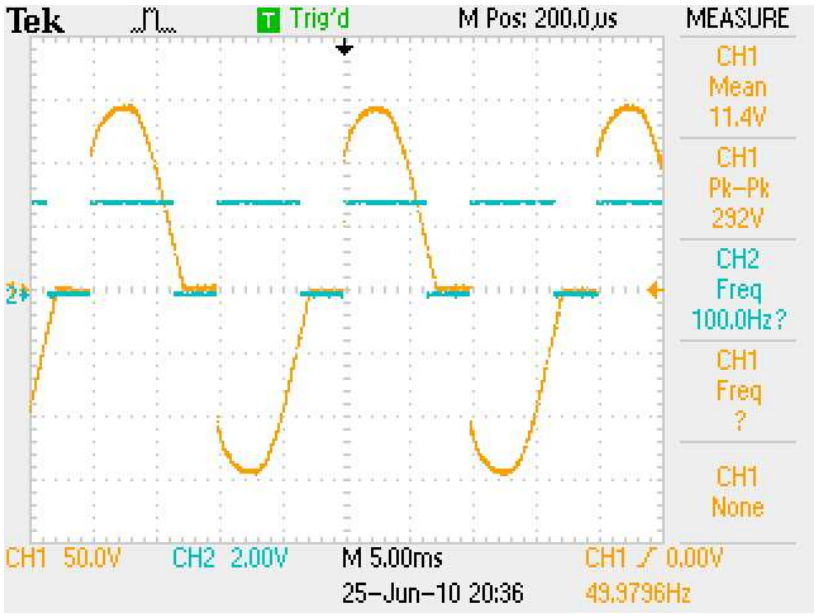

Figure 4.6: Syncronized TRIAC pulses for a 75% brightness

backup to ensure the accuracy of the time and allow to read on demand time values from the RTC. We have used DS1340 RTC from Maxim in our design. Additionally the RTC feature can be used in many future applications where it needs absolute timing such as scheduling some task in a specific date and time.

4.1.5 Firmware upgrading

A custom written bootloader code was initially programmed into the micro-controller using a PicKit 3 Programmer[8] at ICSP[9] port. With this bootloader code, the board is searching for SW1: ON state at the boot up to skip the program load and waits for a valid hex file to be loaded via the USB port. This bootloader was very useful when implementing the firmware codes for the first time as it does not require a standard PIC programmer for upgrading. I have used SIOW (Serial Input/Output Monitor) program provided with CCS IDE[10] to load the hex files via USB cable.

4.1.6 Xbee-USB Dongle

Since almost all the Xbee USB Dongles commercially available only supports the modem feature or Programmer only. Jig Boards are bigger in size and attached using wires, i decided to make a custom made Xbee dongle for pushPin having both functions of programming and acting as modem. As shown in Figure 4.7 the design is small enough to

[8] www.microchip.com/pickit3/
[9] In Circuit Serial Programming
[10] www.ccsinfo.com

directly plug into USB connector of a PC or a Laptop.

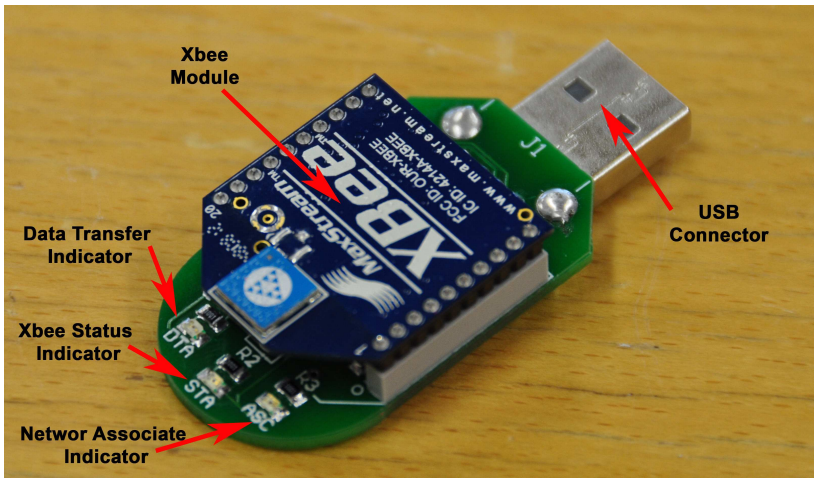

Figure 4.7: Xbee USB Dongle for pushPin

This dongle was used for programming the network parameters and node identifiers in xbee modules before inserted into pushPin modules. At the same time it was used for analyzing network traffic using a custom application acting as a PC node.

4.1.7 Module outer design

pushPin outer design is common for all modules. Different faceplates on each modules will give a different function for each modules.

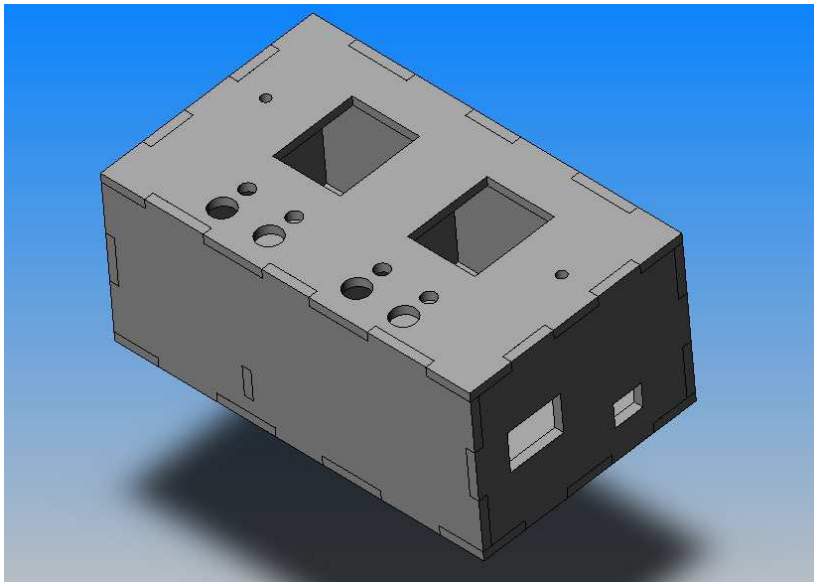

Figure 4.8: Double switch design

These outer designs have modeled in 2D data sets using SolidWorks [13] software. Each module has been simulated in a 3D construction view (Figure 4.8, SolidWorks assembly view) to find out the internal and external structure problems. After examining and iterating the design until satisfactory, it was broken down into 2D pieces and with Acrylic sheets in a Laser Cutter, models were made to real.

4.1.8 Module implementation

Using the master board and two daughter boards (RTC, AC dimming) we have implemented 8 different modules as the first prototype. One more module is at the final stage (InfraRED bridge) when i'm writing this thesis draft. Table 4.2 shows the implemented modules and it's feature description, Figure 4.9 shows the overall pushPin module implementation.

Module name	**Functional Description**
Single Switch	2 × output pin slots
Double Switch	4 × output pin slots (2 per switch)
Slider	2 × output pin slots
Clock	2 × output pin slots
Motion detection (PIR)	1 × input (trigger) pin slots, 1 × output pin slot
Timer	1 × input (trigger) pin slots, 1 × output pin slot
AND Logic	2 × input (logic) pin slots, 1 × output (logic) pin slot
AC Load	2 × input (logic) pin slots
InfraRED bridge	2 × input (channel) pin slots

Table 4.2: Functional description of pushPin modules

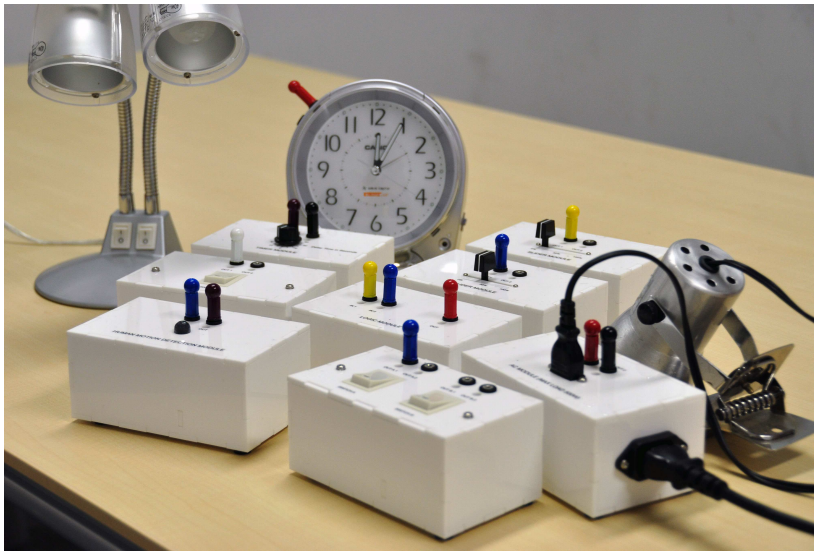

Figure 4.9: pushPin complete module implementation

4.2 Software Implementation

4.2.1 Firmware

All firmware codes on pushPin is written using CCS C[11] firmware programming language. The main concern when writing the firmware code was to make a unique code for all the different modules and minimize the polling of functions repeatedly. Fortunately the firmware code only contains one polling thread, i.e for analog to digital conversion process required for reading the slider values. Other than that, all firmware codes are interrupt driven so that it minimizes the cpu overhead. Figure 4.10 shows the firmware block diagram of pushPin. A unique code for all the modules were not really successful as for different modules, it required to share a same resources which being used by another module. The current implementation is partially unique to all modules except for the clock and motion sensing module. The rest shares a common firmware.

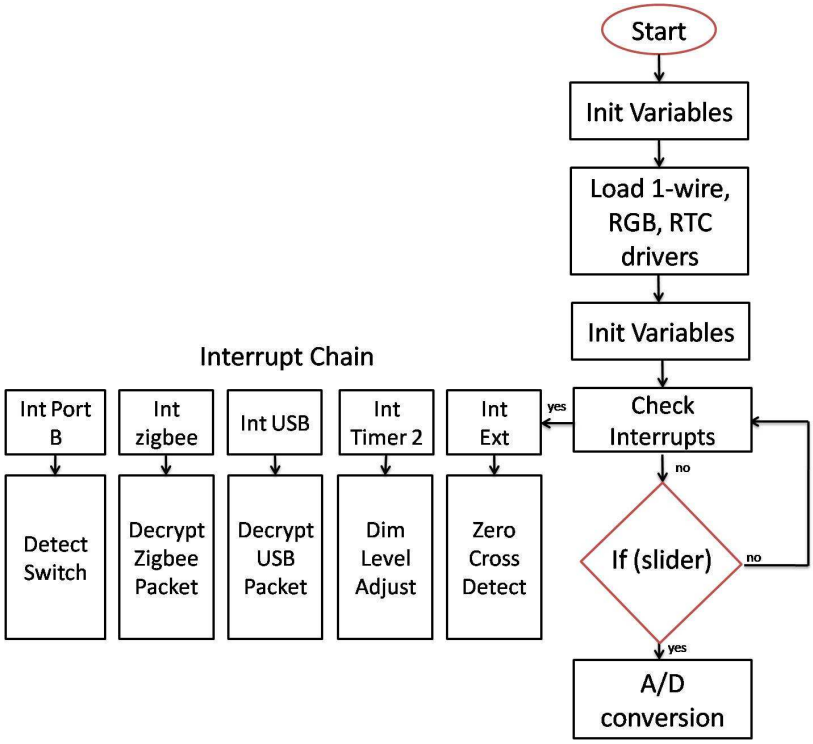

Figure 4.10: Firmware block diagram of pushPin

Major implementations of the firmware includes 3 low level drivers written in order to communicate with RTC, RGB Driver and 1-wire master driver IC's [12]. Even though these IC's are connected through i2c interface to master controller, the functionality cannot be directly accessed. Thus 3 low level drivers has been written and the functions

[11] http://www.ccsinfo.com/
[12] Integrated Circuits

of these drivers are being used in the master controller firmware. Note that code related details are attached in the Appendix B section for easy referencing.

pushPin protocol structure

Communication protocol structure of pushPin is simple. Entire protocol is 6 bytes and it is similar to the IP protocol used in computer networks. As shown in Figure 4.11 first byte is the preamble header which is always set to 's'. This is used for identify the start of the packet. Next is the destination address which can be designated address of a device or a broadcast. Broadcast address is 'x' in pushPin network. Next is the sender address which is defined in the device_id's header file in firmware. Apart from pushPin devices, a PC is denoted by address 'p'. This is used for pinging the devices from a PC or to dump the network traffic for debugging. Next is the sender Pin ID. Pin slot associated for the specific packet is analyzed depends on the operation performed (if switch 1 pressed, pin slots associated to switch 1 is scanned) and pin ID is read from and send as the sender pin ID.

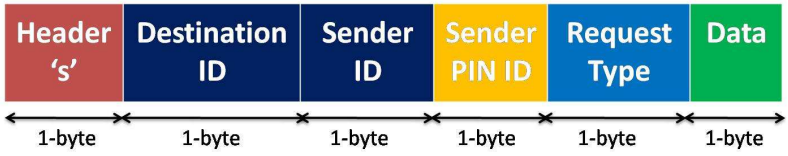

Figure 4.11: pushPin communication protocol structure

The last two bytes are selecting the data and the request type. 5th Byte denotes the request type which can be '0' for normal operation, '1' for ping request, and '2' for synchronous LED blink packet. For future revisions there is a possibility of adding more request types such as read switch status, read slider status, firmware revision etc.... Last byte is the data which is carried in the packet. As common term for activation we use decimal 255 and for deactivation we use decimal 0 as the value. Intermediate levels represent the analog data which is applicable for AC dimming using the Slider module.

4.2.2 Application software

pushPin system does not require of any application software neither a computer to run. It has it's own network and communication protocol in the base level for sending and receiving messages across the Zigbee network. Even though, for technical evaluation I have made 2 applications to measure the network performance of pushPin network and packet sniffer for sniffing the network traffic in case of a troubleshoot. These softwares are written using C#.NET Language in Microsoft Visual Studio 2008 suite and requires .NET Framework to be installed before running on a PC.

pushPin network packet sniffer/logger

Network packet analyzer captures all the traffic in the network, no matter where the packets are addressed to. Moreover, for easy understanding there are 3 display modes, user can choose either decimal, hex or text based output.

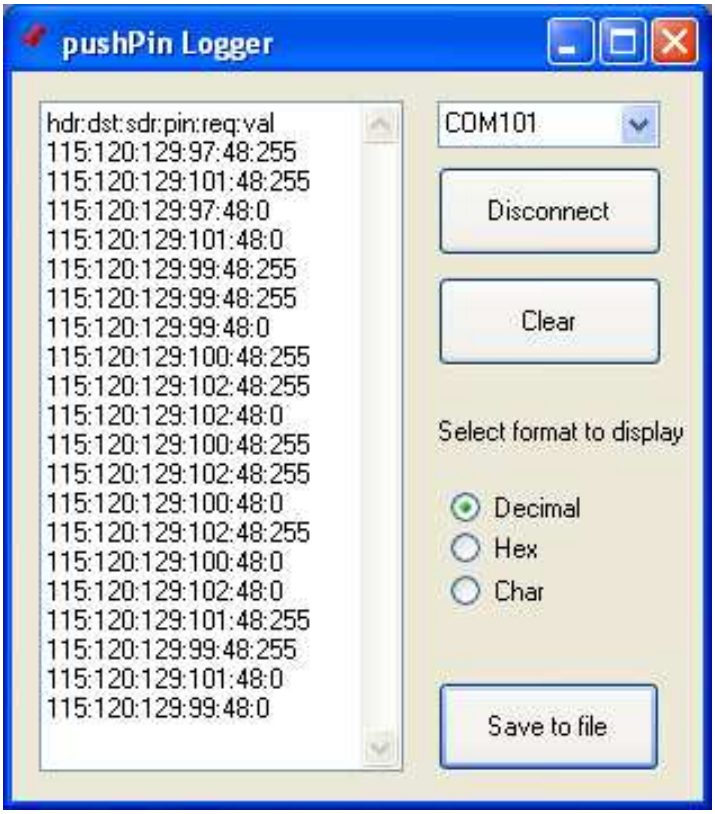

Figure 4.12: pushPin Logger: network packet sniffer/logger

To use this software, connect the pushPin Xbee Dongle to a PC, and run the pushPin logger application. After choosing the corresponding COM port from the drop down menu above the connect button, press connect. Software will connect to the dongle and display the response when connected. After that it is automatically enabled for capturing the packets. If the user needs to clear the response display, simply click the clear button. Figure 4.12 shows a sample packet sniff during a network test. Once finished capturing, user can choose to save it to a text file by pressing the 'Save to file' button. A text file is generated with the "log-yyyy-mm-dd_hh-mm-ss-am/pm" format in the same directory that you have executed the application.

Round Trip calculator

Round trip calculator calculates the Round Trip time taken for a ping packet to travel from PC to a device and come back to it's start node. This software can do a discovery search on pushPin network and find the available devices within the wireless detectable area. The devices dropdown list will be filled with the available devices and user can choose one of the devices and ping.

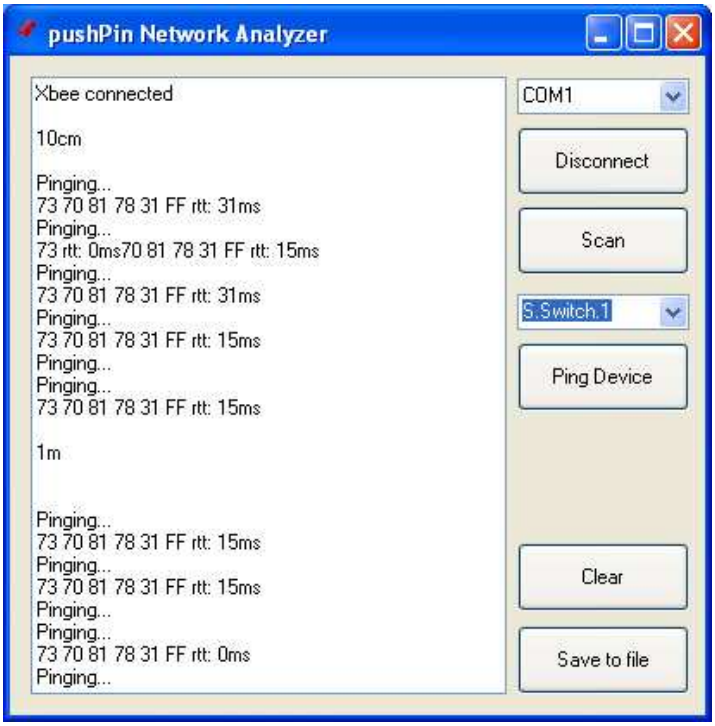

Figure 4.13: pushPin Network Analyzer: Round Trip Calculator

To use this software, as explained in the previous pushPin logger, connect the dongle, choose the correct COM port and press Connect. Do a Discovery scan by pressing the Scan button and select the appropriate device from the device dropdown list. User can send a ping request by pressing the Ping button and the Round trip time is displayed as rtt: ??ms in milli seconds. (Figure 4.13 As previous, Clear button for clearing the response display and Save to file button for save a file can be used. A text file with the "network_dump-yyyy-mm-dd_hh-mm-ss-am/pm" format in the same directory that you are have executed the application.

4.3 Programming with pushPin

In this section I will indicate few typical programming scenarios with the implemented modules. Like an ordinary programming language, the advanced usage is dependable

with the users creativity.

4.3.1 Data types: Digital, Analog and Sensor data

AC load with a Toggle switch

Figure 4.14: An AC lamp controlled by a toggle switch

As shown in Figure 4.14, Output Port-1 of Switch module is connected with Input Port-1 of AC load module. Therefore AC load can be turned ON/OFF by toggling the Switch module.

AC load with a slider

Not only digital data can be handled by pushPin, the communication protocols were designed in capable of handling analog data. As shown in Figure 4.15, slider module output is connected with AC load input. In this case, AC load not only turn ON/OFF, but also it can be dimmed by adjusting the slider input.

AC load with trigger of a Human Motion sensor

Analog data, Digital data provided directly using a switch or slider is not the only case. Sometimes we might monitor some external parameter such as room temperature, humidity, motion sensor, sunlight etc... and use their trigger for actuating another object. In Figure 4.16, Output of a Human motion sensor is connected with AC load module. In this case, by turning the switch ON, motion sensor starts detecting the environment and looks for a passive IR emitting object. In this case, once a human is detected, connected output (in this case AC load) is activated.

Figure 4.15: An AC lamp dimmed by a slider input

Figure 4.16: An AC lamp triggered by a Human Motion sensor

AC load with trigger of a Alarm Clock

Figure 4.17: An AC lamp triggered by an Alarm Clock

Similar to the previous case, a different sensor, an alarm clock is being used. In Figure 4.17, Output of clock module is connected with AC load module. In this case, user can set the desired time for the alarm to ring by dialing the alarm tick. When the alarm rings, connected output (in this case AC load) is activated.

IR controller TV with a Toggle switch

IR command based pushPin module is an output module which allows users to teach any IR command to the device and manipulate them later with any combination of pushPin activities. In Figure 4.18 user programs the Turn ON command to the module by pressing the small switch on top of the module. Once in the learning mode the small RED LED next to the small switch will keep blink for 10s until it receives a valid IR command. The IR command is stored in the EEPROM and later user can associate the learnt command by inserting a PIN to the IR commander pin slot.

In this module, the advantage is that, user can program/re=program any command by simply pointing at the module and these commands can be used with any combinations of pushPin such as Motion detection trigger, switch trigger, leveling the volume of TV using slider and many more appliances.

Figure 4.18: Programming ON command of a TV remote to use with Switch module

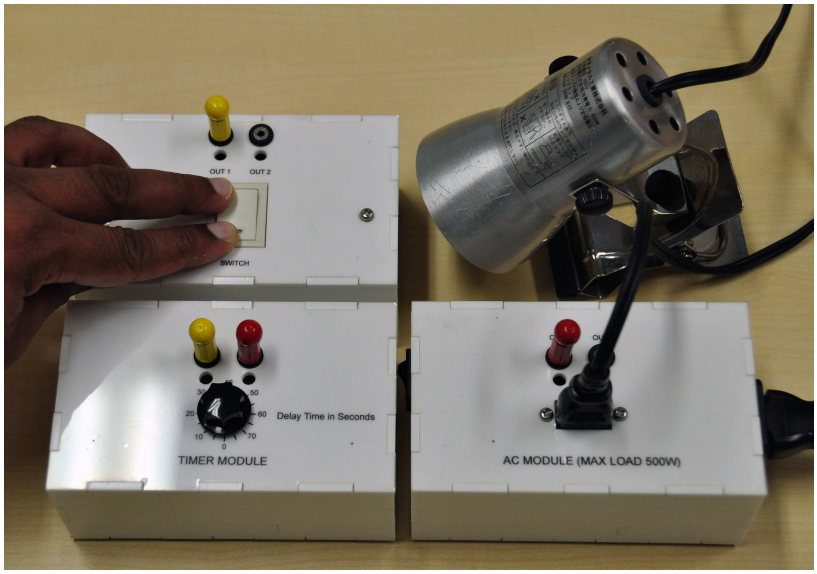

Figure 4.19: An AC lamp connected to a delayed switch input

4.3.2 Sequential Operations

AC load with delayed trigger

In this example, (Figure 4.19 an Intermediate module (Timer Module) cascaded with the input and output. Output of switch module is connected to input of Timer and output of Timer is connected with input of the AC load. Switch activates the timer count down. User can set the desired delay time by rotating the knob (5s, 10s, 15s, 20s, ... 80s) and at end of countdown, AC load will activate.

4.3.3 Parallel Operations

AC load with combinational logic

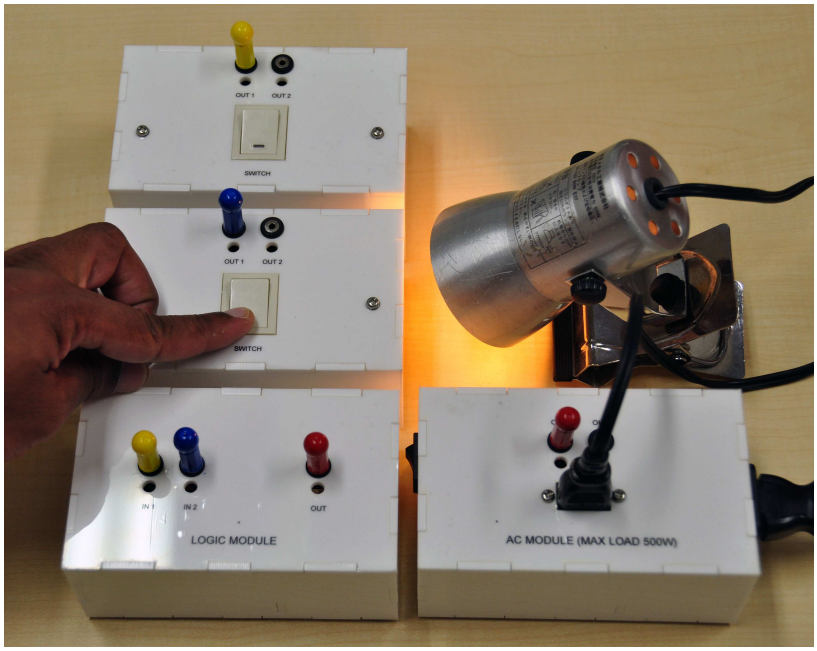

Figure 4.20: Two switches performing combinational logic

pushPin can perform logic operations. Figure 4.20 shows two switches connected logically to perform a combinational task (only if both inputs present) and up on success to activate some object. Output of switch modules are connected with inputs of logic module and output of logic module is connected to a AC load. When both switches are active the output will activate, by disabling either one will deactivate the load.

AC load with threshold

In this example, intermediate module (Logic) is cascaded with the input and output. The difference is that the logic unit compares the two analog inputs, This method can set a threshold point by moving the slider. It can be explained better using a real world

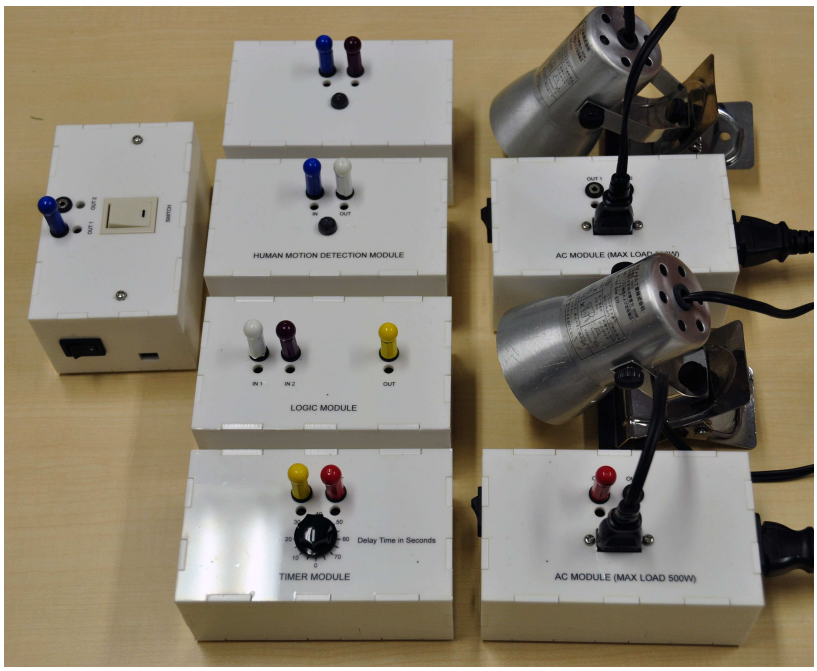

Figure 4.21: Setting the Threshold for output activation

application. If user wants to set the temperature (threshold) of a Air conditioner to 24 and if exceeds, to cut off power. This example assumes the connected air conditioner sends the temperature data in decimal over the linked channel. According to Figure 4.21 one slider can be an analog input, and the second slider can be the threshold set. When the analog input exceeds the set threshold, output logic becomes true and therefore AC load activates.

4.3.4 Mix operations

Coffee machine and Lights On with Alarm Clock trigger

In this example, (Figure 4.22 shows the experimental setup for a alarm clock driven Coffee machine. When alarm struck, the Coffee machine will turn ON and start making coffee. After 5 minutes Coffee machine finishes and ringing a buzzer for wake up and turn all lamps in the room.

(please note that in Figure 4.22 the coffee machine and lights are both represented by Lamps)

Intruder system alarm with 2 × Human motion sensors

In this scenario (Figure 4.23), one AC load has been used as the indicator for motion sensor 'in active' and other as motion sensor trigger detection. Motion sensors are turned

Figure 4.22: Connecting Coffee machine, Alarm Clock and Lights together

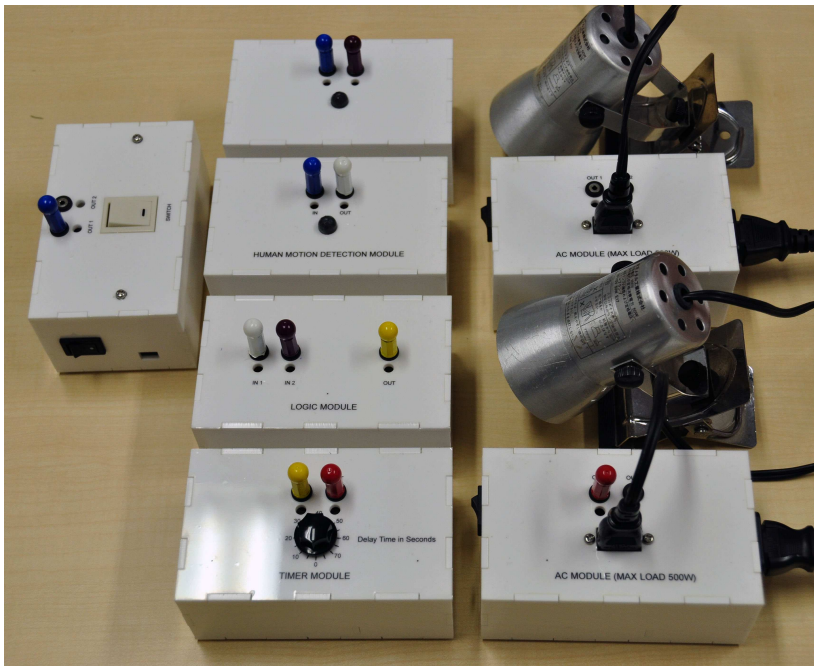

Figure 4.23: Advanced Intruder Alarm system

ON using the switch module connected and it is indicated by the AC load 1. After activating the Motion sensors, it waits for a human motion to be detected by both sensors. On trigger, AC load 2 will be activated.

With these examples it showed pushPin system supports sequential programming, concurrent events using the parallel networking and the output modules can be any directly controllable AC device or any InfraRED command based appliances.

Chapter 5

Evaluation

5.1 Technical Evaluation

In technical evaluation, pushPin system has been tested for network redundancy over adding new modules (plug-n-play) to an existing network and also a typical round trip response time over physical distance.

5.1.1 Zigbee Network redundancy

In this experiment, pushPin devices were placed in same physical space and starting from 2, the devices were added to the network and perform controlling devices. All the data traveling through the Zigbee network was monitored using pushPin "network packet sniffer/logger" software explained in Chapter 4.

The results were logged into a text file for understanding where the problem occurs and observation was done in real time. We added 20 modules one-by-one to the network and performed discovery scan and repeated the same steps for 20 times. The results were compelling that pushPin network showed very robust and accurate data traffic with no failures.

Even though 20 modules (nodes) are enough for this thesis evaluation, Understanding that this will be very useful for the commercialization and any future implementations, we are planning to do more in depth scalability study by increasing the number of nodes. The scope of this project was only to create a everyday object programming environment for everyone and in any case the large scale scalability did not stop me from evaluating the current design with ordinary users.

5.1.2 Typical Node Response time

The response time was calculated based on a echo reply of a ping request. The environment was set as indoor and receiver (end device) is placed in distances and repeatedly ping at each distance for more than 5 times. Average response time was taken for a given distance and the process was repeated until 15m at 1m pitch. [1]

[1] Xbee indoor LOS is less than 30m

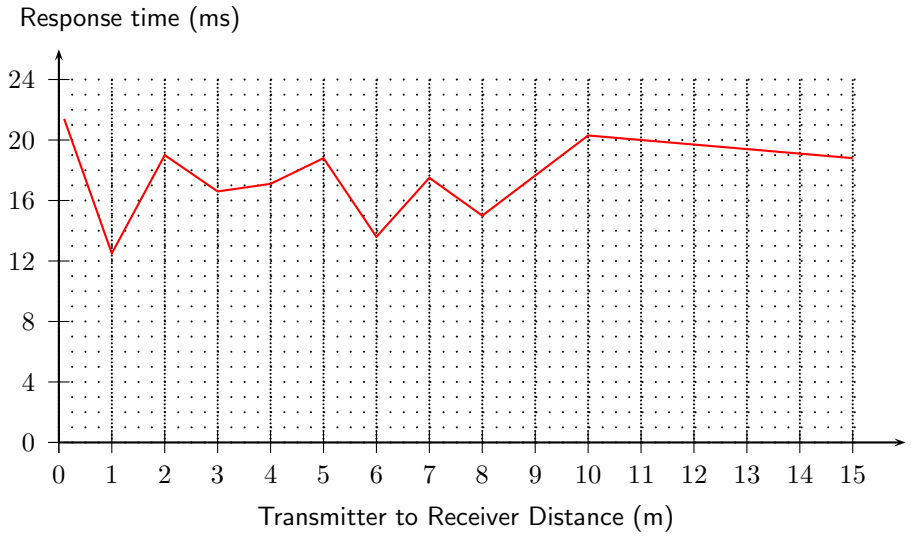

Figure 5.1: Response time for ping packets

Figure 5.1 shows the response time curve. It shows the response time is in between 12.5ms to 21.4ms and stays within an acceptable range for a home environment.

5.2 User Study

5.2.1 Early stage pilot evaluation

After the very first prototype is finished, we tested the system with five participants (four males and one female age ranging from 23 to 31, with a mean of 27.8. Two of them were living with family and 3 of them were living by their own). These preliminary interviews taught us about their current living conditions and habits, and how they wanted to use our proposed system.

This early evaluation helped us in many ways. We observed and understood what people need and how our system can fit into their daily lifestyle. In this pilot study we basically wanted to monitor how participants react to the system at first sight, how they figure out the different modules and how to connect each other and what are the most promising features to users. Also we looked into the reliability of the system by noting down the bugs, deadlocks in the system for correcting them before the formal user study. Because of the above reasons our user selection was a variety of people including technically sound, networking skilled, wacky test users, and with no knowledge about any of the above. Having known these criteria are not suitable for a official user study, what we basically wanted was to study the system in extreme conditions and novice conditions.

After explaining the system to them with the wired connections being wireless, they figured out how to connect the basic modules (one-to-one) by their own. Interestingly ordinary users did not pay attention to link more than 3 devices together and their example scenarios showed that what ordinary users need is very basic linking mechanism with existing devices to get their work done easily. For example, many users said that they want to wake up with the sunlight by opening the curtain automatically on a set time.

During technical and wacky users' testing it was discovered that slider with switch combination caused the output module to hang at a certain time. Also Having 2 switched and 2 output modules caused the same problem. After analyzing the network traffic of those users, it was found that actually it's caused by the self re-transmission occurred by the switch, slider modules. (In the implementation, to make sure one device gets the stimulus, we had implemented a self re-transmission mechanism at all switch, slider modules. This mechanism applies when the device receives any packet, if it's not designated for itself, it will retransmit.) There was an ambiguity in this re-transmission where all modules started to retransmit at the same time and the USART IN buffer was not sufficient to handle that incoming data in the AC module. This was solved in the later experiment by canceling the unnecessary retransmissions.

Also by observing the users behavior it was found that Logic module was the hardest to understand by ordinary users. For 3 participants the understanding of Logic module at first sight was okay. However by explaining the usage everyone understood how it can be used. Due to this in formal evaluation Logic module was not presented.

Please note that the time different between Pilot study and Working prototype evaluation is just one day.

5.2.2 Evaluation of working prototype

Participants

8 More participants (4 males and 4 females with age ranging from 22 to 41, with a mean of 26.7) were invited to take part in the user study. 5 of them were staying with family and 3 of them were staying alone in apartments. Further more, 4 of them stated that their programming or networking skills are very weak and 2 stated that their programming and networking knowledge is sound. The rest 2 participants stated their knowledge is fair.

Setup and procedure

The study involves three steps. 1) Questionnaire 2) Understanding by seeing 3) Using pushPin to connect devices. Hardware setup is as shown on Figure 5.2 and a Web camera is used to record all user behaviors. Questionnaire collects background information such as demographic information, living habits, usage of Alarm Clock, usage of Timer feature in everyday devices and what devices would they like to connect together in their home in future to make their daily life easier. The questionnaire was conducted with no information provided about the pushPin system.

Understanding by seeing step aims to further test the intuitiveness of the proposed colored pins, intuitiveness of device pin slot arrangement, usage of each device with regards to pushPin metaphor and the proposed method of cascading devices. First the users were given a brief introduction of connecting devices based on input and output port relationship by giving an example of connecting the iPhone's charger to iPhone. Next they were given a Switch module and AC load module and a pair of pins (not describing it's a pair) and asked them to connect each other. Next they were shown the entire set of colored pins and asked to guess what the color means to the pins. Further they were asked what means if there are more than two pins on same color. Next they were asked to discover the 6 different modules on the table and identify how to connect each device and it's purpose. Next the system and how it can be connected with pins were explained (using RCA cable) to the participants and asked them to write down, was their previous guess was correct or not.

Figure 5.2: User Evaluation Setup

Once they finished step 2, they proceed to step 3 and were instructed verbally in everyday language, and asked them to perform various actions in pushPin system using pins. (Ex: Connect the Switch module to AC module so that when you switch ON light should turn ON and vise versa). Furthermore, direct connection relations (Ex: Connect Switch input to AC module output) were carefully avoided in the verbal instructions. They started from simple tasks having input and output relationship with switch + AC module, slider + AC module and Alarm clock + AC module. Advanced tasks includes cascading 3 devices (Switch + Timer + AC module, Switch + Motion detection + AC module) and parallel threads. Time consuming tasks such as Alarm Clock was accelerated by turning the alarm dial counter clockwise to the current time where it activates the Alarm. After going through the planned tasks users were asked to perform their own connections with the system. Finally a small interview was carried out to ask the experience they had in the system. All three steps with break times and questionnaire time lasted about one hour for each participant.

Results and observations

All users understood the first connection in step 2, without any explanation. Overall, users did a very well understanding in understanding colors to pairs and groups in step 2. 7 out of 8 users correctly interpreted that the colors represent a group. One user misinterpreted the meaning of the colors, but it was not far from the correct meaning. She interpreted colors represent a ranking and for example RED could be first priority

and BLUE could be 2nd priority and so on.

All participants also did very well in understanding the devices and it's purpose with regards to pushPin system in step 2. 6 out of 8 participants interpreted the correct use of all devices. Two users understood what the Alarm clock's purpose but, they could not understand how it can be connected to pushPin system and how it relates. Out of the same two users, one user could not understand how Human Motion detection modules can be connected to pushPin system.

All participants also did a very good job in making connections in step 3. All participants correctly interpreted the simple tasks where only two modules involve in the connection. Also in advanced tasks where they need more than one connection, 4 participants interpreted the correct use of different colored pairs being used for different signal connections. Rest of the participants understood the concept clearly when explained.

While being aware of the systems' complexity when using intermediate modules by ordinary users, We performed a test understanding how far they can figure out without any assistance. Almost no participant could not understand the meaning of input/output relationship in the intermediate modules by their own. By explaining the concept, soon they understood the concept clearly. Overall all users mentioned that it was very straight forward to connect using the colored pins.

In step 3, users own connection implementations, users were vigorous to find out the advance features and different connections using the system. In fact one user asked if there is a way to perform when two Switches were ON to turn ON a light, and was given the Logic module to perform the task. Later it was explained why the logic module was not included in this experiment.

In the post experiment interview users response for the pushPin system was very positive and 6 participants stated that the reconfiguration from one step to the other was very easy and 2 participants were marginal. Many participants said that they would like to purchase if it's available. In terms of most liked Modules, 5 participants out of 8 were really liked the Alarm Clock module and 2 participants liked the Motion sensor and one was interested in Timer. She further stated that Timer can be used to turn ON the lights at 6:00p.m pretending that they are inside the house, when they going trips. Many suggested that the system is easy to understand by ordinary users and they can use it in many activities. They also praised the concept of connecting everyday objects together is very useful. In fact one participant stated that he cannot usually get up by the iPhone alarm sound since it is not too loud for him and with pushPin system he can connect his iPhone to another sound source like Home Theater system or

if possible to ring his phone when the alarm rings. He further stated that when he hears an incoming call, the probability of waking up is higher than the ordinary Alarm sound. Few users stated that they want the sunlight to come into the room when they wake up in the morning and pushPin can be used to control automatic curtain systems with the Alarm Clock. One participant thought beyond the existing modules and said that she really worries when the Dog barks thinking that it might be a sign of a problem. But specially when watching an interesting TV program, it's hard to go and see outside. With pushPin system, she can install a camera on the Dog's cage and remotely switch the TV program or Camera feed by push of a button.

Compared with the positive responses, perhaps for the research community any concerns or problems found in when doing the user study is more useful. When asked to connect the pins to the Switch module and AC module, almost all participants further thought about the IN1, IN2, OUT1, OUT2 in the labels and connected as IN1 to OUT1. This was not a problem for the operations in our system, but may be the time taken for them to figure out IN1 or IN2 would be unnecessary load to the users. Even though they figured out that way, everyone connected the devices within 3 seconds time. Another concern noticed that when asked to connect the Switch Module + Timer + AC load and the same Switch + AC module, almost everyone selected a different pair color for the second connection. Although this was fully supported by our system and did not run into any problems during the user study. But it was interesting to find that, ordinary users didn't think of the tapped wire concept provided in pushPin.

Two participants suggested that when using the Slider and Switch simultaneously, if it can turn ON to the previous states Brightness it would be better. Current pushPin system goes to full bright when using the switch no matter what the last brightness value. Actually this can be implemented easily by modifying the firmware. 3 participants stated that if input/output pin slots are different shape or colors, they can understand the relationship very quickly without any explanations. During the user study, there were times that it needed restarting the modules in the middle of the experiment. Even though restarting was easy using the main power switch, this was due to a network overload caused by in-proper programming. After the user evaluation, we addressed this issue by updating a new firmware version. 5 Participants out of 8 stated that the connection troubleshooting was not annoying and 3 stated that marginally true. With regards to connection troubleshooting many participants said that a LED indicator will be more than enough to understand where the connection problem and one stated that it can possibly alert me if I forgot to put the output pin or input pin in a connection.

Overall this evaluation's expected results were very good and got many points to re-think when designing the next system. The RGB LED indicator was already implemented

in the current system and a delay in getting the LED's caused the indicator to be missing in this experiment. We believe that this caused the user to get annoyed when troubleshooting the connections and we plan to do another user study with all features included.

Chapter 6

Discussion

6.1 Limitations

As the first step of this research, there are many limitations of the system. Availability of Input/Output modules also limits the number of different devices to be connected to the system. During the user study some users were interested in connecting the pushPin system to a web based email service so that on trigger it can generate an email. At the same time some users were interested in connecting different types of media together such as connecting a TV to a large screen or connecting a camera to the TV for a slide show. These technologies have not been thought into the pushPin system, but surely it would be valuable for the ordinary users and pushPin's ultimate goal is to provide connectivity between any object, any media seamlessly.

The mental limitation of number of pin pairs and different colors also initially thought when designing the system. The user study showed that most users do not want to connect several devices together, but by connecting just one or two devices together they can make a big difference in their living space. Even so this is a limitation, pushPin system's scope is not to wire a whole home like in home automation or smart homes. It is to provide easy to understand programming technique for everyday objects in everyday space. With this concept we are not considering to re-configure every object in users living space on a daily basis, but to connect some of the devices that users frequently uses. However to overcome this situation we have been thinking to have a mixture of physical icons (phicons) and colors so that users can select not only the favorite color but also the phicon.

The AC module dimming is tuned to 100v, 50Hz (Japan) utility power and it will not work in voltages or frequencies other than specified. Even so adopting the system to other voltages does not require any hardware changes and its just an firmware upgrade.

6.2 Future work

6.2.1 the near term

As near future implementations we would like to expand the system to non-pushPin based appliances by introducing Bluetooth, Zigbee and even Ethernet based bridge modules. These module can be integrate by linking them with a input source such as a switch, slider, clock or even a sensor.

Battery charging of the modules requires USB power and currently its being provided by connecting a USB HUB having 4 outlets and connecting 4 modules at a time. This drains lot of current from the PC and in near future I would like to make an external charger which can handle more than four connections and higher current charging.

LED indicators are good for troubleshooting, but only it can be visible to the users. We thought of adding a small beep to indicate a successful pairing in the network and this will also helpful to find the other end (specially if not visible) by disconnecting and connecting from the accessible end.

6.2.2 possible future implementations at thought

During the user study, some users were little confused with having two pin slot names IN1, IN2 or OUT1, OUT2. Furthermore some were puzzled when connecting the intermediate modules having IN and OUT slots. In next stage we plan to come up with a more sophisticated design with one pin slot per device and depends on the operation performed, it can act as either input or output port. This way the users will not have to consider complex wiring patterns, but only to consider which device connects to what device. This somehow limits the flexibility in using the devices for advanced users, but will serve the majority of ordinary users. However this design is not yet finalized and if the system cannot simplify as a one pin slot design for all modules, the next approach is to clearly show the difference of input/output pin slots using a different color or a shape.

pushPin system is currently installable at only one physical location and does not support long distance operations. We plan to implement extender modules over the Internet so that users can connect two different physical locations over the Internet and take control of the devices over the other side. This allows users to remotely configure your space, and having collaborative programming environments. Further more we plan to extend the reachability from 3G and GSM networks so that consumers can get in touch with the system wherever they go. This benefits users programming/scheduling their home appliances while they are out of the house. Heating the water and fill the bath before

getting into the house, turn ON the roof lights and activate the security system when you leave your premises, escort a delivery person to the garage with limited security and under video surveillance when user is not at home at the time of delivery, can be some practical examples at thought.

Although pushPin system is a novel concept, some users might misinterpret that by using commercially available systems such as Insteon or X10 they can perform the same operations. It might be true that it is possible to implement the same configuration, but they become easy to control once it is setup by professionals. Re-configuration after the initial setup is not that easy for an ordinary user and practically it takes more time compared to pushPin system. So in near future I would like perform another user evaluation comparing the pushPin system with existing commercial systems such as Insteon and X10 to show that pushPin is better for fast re-configuration device coordinations.

6.3 Applications

pushPin system in general context can be used in many day-to-day scenarios. I would like to pay attention about few of the scenarios here which will be conceptual and made with some imaginary modules which is not yet available in the system.

When you are watching the favorite TV show, you get a call from your boss. Obviously you cannot answer the call without lowering the Televisions volume. With pushPin this operation can be automated. It senses the phone ringing and if you happen to link the phone and TV set, it will automatically reduce the volume of the TV set. Once the call has been answered the volume goes back to the original state. If you are listening to music by linking the HiFi setup and phone, volume of HiFi system will be controlled automatically.

Waking up in the morning by setting the Alarm is not new. In fact sometimes you would like to oversleep some more time and time you overslept will vary depend on your mood. If you plan to run the water kettle using a Timer at the same time, you might have experienced when you get up after overslept, the water is already cold. With pushPin, once the alarm struck, you can use a Human motion sensor to detect whether you have actually got up and then to turn the Kettle ON.

Another scenario is that sometimes you are expecting deliveries to your doorstep, but without waiting the all day for delivery, you want to watch a movie or listen to some music with headphones. In this case you might not hear the delivery person pressing the door bell and might miss the delivery. What we need is a small extension to the

existing door bell to convert the bell to a Light indicator. With pushPins by attaching one pin to the door bell and another pin to an Indicator will provide user with an visual alert when the delivery arrives.

Another possible application scenario is to connect the virtual tasks to physical indicators. Have you ever missed important chat messages from online buddies when you are away from computer. With pushPin you can simply link the desired software application to a everyday object such as a bell, light indicator or to ring the mobile phone. In this way you can re-direct the virtual alerts to the physical world and do other physical activities without starring at the computer screen at all time.

These are only a few example scenarios where pushPin concept can be used by everyday users. but in reality the pushPin concept can be applied in a limitless manner.

Chapter 7

Conclusion

In today's world, we can clearly see there are lot of devices around us. But there's neither a coordination mechanism among the devices nor with humans. We envision the future of pushPin technology will be the next generation device coordination mechanism to help everyone embed their everyday objects into their lifestyle.

In this thesis I have presented the technical details of the pushPin system, a virtual connection based stimulus-response coordination among everyday objects that I designed to conduct research in constructive to tangible user interfaces in general and virtual connections being manipulated physically for device coordination in particular. A pilot study and a formal user study was sufficient to prove that this system is more intuitive for ordinary users to connect, re-configure devices together and combine their everyday tasks to implement some service which cannot be archived from the current commercially available systems. First time users were able to figure out how to connect simple one-to-one device relations where as complex task after being thought. The flexibility of changing the system connections at any given time was a plus feature for everyone.

pushPin provides an easy to understand, fast reconfiguration mechanism for the everyday device coordination. Even though current implementation is probably too expensive for a commercialization, we have been thinking about a scale down, low cost version in the future. The proposed method, 'pushPin' can be easily implemented as a generic module distributed to popular consumer electronic companies to embed at their production stage. With a wide expansion of pushPin in future, consumers will gain the freedom to program their home appliances according to their needs.

Bibliography

[1] 17-output led driver/gpo with intensity control and hot-insertion protection. http://www.maxim-ic.com/datasheet/index.mvp/id/4206/t/al.

[2] Bacnet official website. http://www.bacnet.org/.

[3] Ds2482-800, 8-channel 1-wire master. http://www.maxim-ic.com/datasheet/index.mvp/id/4338.

[4] Echelon coorporation, embedded control networking technology. http://www.echelon.com/.

[5] Ietf-rfc0791, internet protocol. http://www.ietf.org/rfc/rfc0791.txt.

[6] Insteon, wireless home control solutions for lighting, security, hvac, and a/v systems. http://www.insteon.net/.

[7] Knx association offcial website. http://www.knx.org/.

[8] Lego.com mindstorms offcial website. http://mindstorms.lego.com/en-us/Default.aspx.

[9] Mathworks - matlab and simulink for technical computing. http://www.mathworks.com/.

[10] Maxim 1-wire technology. http://www.maxim-ic.com/products/1-wire/.

[11] Microchip pic 18f6622, data sheet. http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en021969.

[12] Panasonic mp motion sensors. pewa.panasonic.com/pcsd/product/sens/pdf/amn.pdf.

[13] Solidworks 3d cad design software, offcial home page. http://www.solidworks.com/.

[14] Texas instruments tps73633, data sheet. http://focus.ti.com/docs/prod/folders/print/tps73633.html.

[15] Visual studion 2010, visual c++. http://msdn.microsoft.com/en-us/visualc/default.aspx.

[16] X10.com, security cameras, x10 home security, wireless camera, home automation. `http://www.x10.com/`.

[17] Xbee and xbee pro 802.15.4 oem rf modules, offcial home page. `http://www.digi.com/products/wireless/point-multipoint/xbee-series1-module.jsp#overview`.

[18] Xcode developer tools technology overview. `http://developer.apple.com/technologies/tools/xcode.html`.

[19] Matsushita N. Ayatsuka, Y. and J. Rekimoto. Gaze-link: A new metaphor of real-world oriented user interface. In *IPSJ Journal*, volume 42, pages 1330–1337, Japan, 2001. IPSJ.

[20] Yuji Ayatsuka and Jun Rekimoto. transticks: physically manipulatable virtual connections. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 251–260, New York, NY, USA, 2005. ACM.

[21] Matthew G. Gorbet, Maggie Orth, and Hiroshi Ishii. Triangles: tangible interface for manipulation and exploration of digital information topography. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 49–56, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

[22] Ken Hinckley. Synchronous gestures for multiple persons and computers. In *Proc. of UIST 2003*, volume 2003, pages 149–158, 2003.

[23] Hiroshi Ishii and Brygg Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 234–241, New York, NY, USA, 1997. ACM.

[24] Sergi Jordà, Günter Geiger, Marcos Alonso, and Martin Kaltenbrunner. The reactable: exploring the synergy between live music performance and tabletop tangible interfaces. In *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 139–146, New York, NY, USA, 2007. ACM.

[25] Kristof Van Laerhoven, Albrecht Schmidt, and Hans-Werner Gellersen. Pin&play: Networking objects through pins. In *UbiComp '02: Proceedings of the 4th international conference on Ubiquitous Computing*, pages 219–228, London, UK, 2002. Springer-Verlag.

[26] Timothy S. McNerney. From turtles to tangible programming bricks: explorations in physical language design. *Personal Ubiquitous Comput.*, 8(5):326–337, 2004.

[27] David Merrill, Jeevan Kalanithi, and Pattie Maes. Siftables: towards sensor network user interfaces. In *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 75–78, New York, NY, USA, 2007. ACM.

[28] And The Pervasive, Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki. Touch-and-connect: A connection request framework for ad-hoc networks. In *First IEEE Annual Conference on Pervasive Computing and Communications (PerCom 2003*, pages 20–29. IEEE, 2003.

[29] Jun Rekimoto. Pick-and-drop: a direct manipulation technique for multiple computer environments. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 31–39, New York, NY, USA, 1997. ACM.

[30] Jun Rekimoto, Yuji Ayatsuka, and Michimune Kohno. Synctap: An interaction technique for mobile networking. In *In Proc. of Mobile HCI 2003*, pages 104–115. Springer, 2003.

[31] Jun Rekimoto, Yuji Ayatsuka, Michimune Kohno, and Hauro Oba. Proximal interactions: A direct manipulation technique for wireless networking. In *In Proceedings of INTERACT2003, Sep.-Oct*, pages 511–518. Press, 2003.

[32] B. Shneiderman. Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69, 1983.

[33] E. Soloway and James C. Spohrer. *Studying the Novice Programmer*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1988.

[34] Supporting. Designing for serendipity.

[35] Hideyuki Suzuki and Hiroshi Kato. Interaction-level support for collaborative learning: Algoblock—an open programming language. In *CSCL '95: The first international conference on Computer support for collaborative learning*, pages 349–355, Hillsdale, NJ, USA, 1995. L. Erlbaum Associates Inc.

[36] Colin Swindells, Kori M. Inkpen, John C. Dill, and Melanie Tory. That one there! pointing to establish device identity. In *UIST '02: Proceedings of the 15th annual ACM symposium on User interface software and technology*, pages 151–160, New York, NY, USA, 2002. ACM.

[37] Brygg Ullmer, Hiroshi Ishii, and Dylan Glas. mediablocks: physical containers, transports, and controls for online media. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 379–386, New York, NY, USA, 1998. ACM.

[38] Nicolas Villar, Kiel Mark Gilleade, Devina Ramduny-Ellis, and Hans Gellersen. The voodooio gaming kit: a real-time adaptable gaming controller. In *ACE '06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, page 1, New York, NY, USA, 2006. ACM.

[39] Mark Weiser. The computer for the twenty-first century. *Scientific American*, 265(3):94–104, 1991.

Appendix A

Schematics, PCB designs, CAD drawings

A.1 Schematics

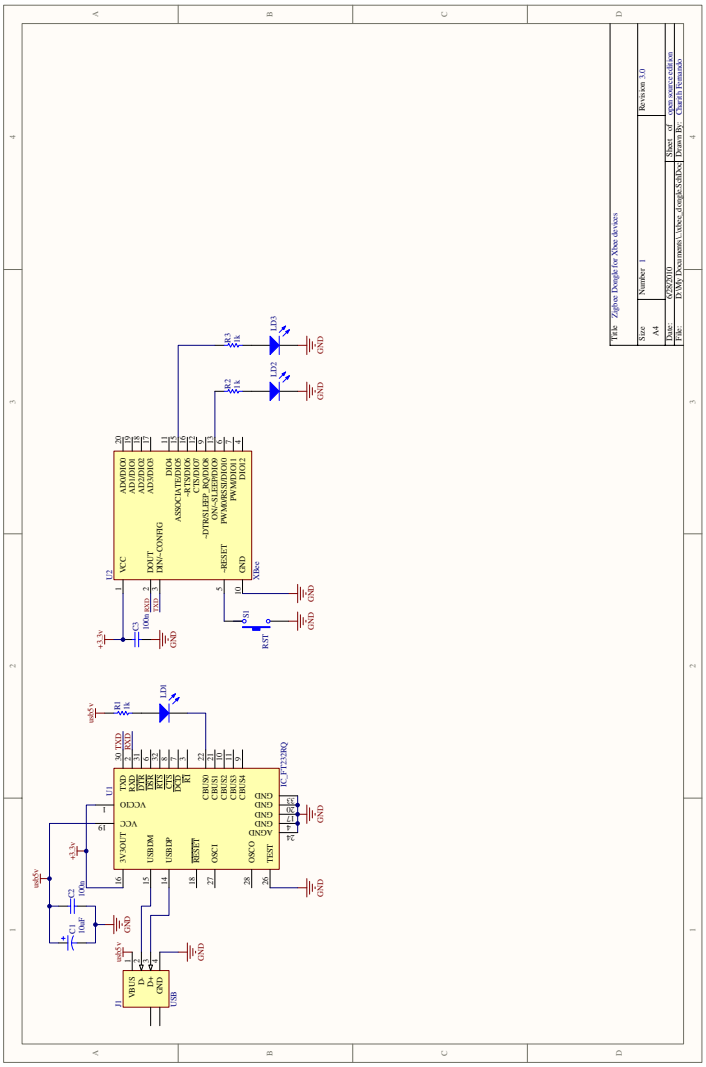

Figure A.1: pushPin xbee dongle, schematic diagram

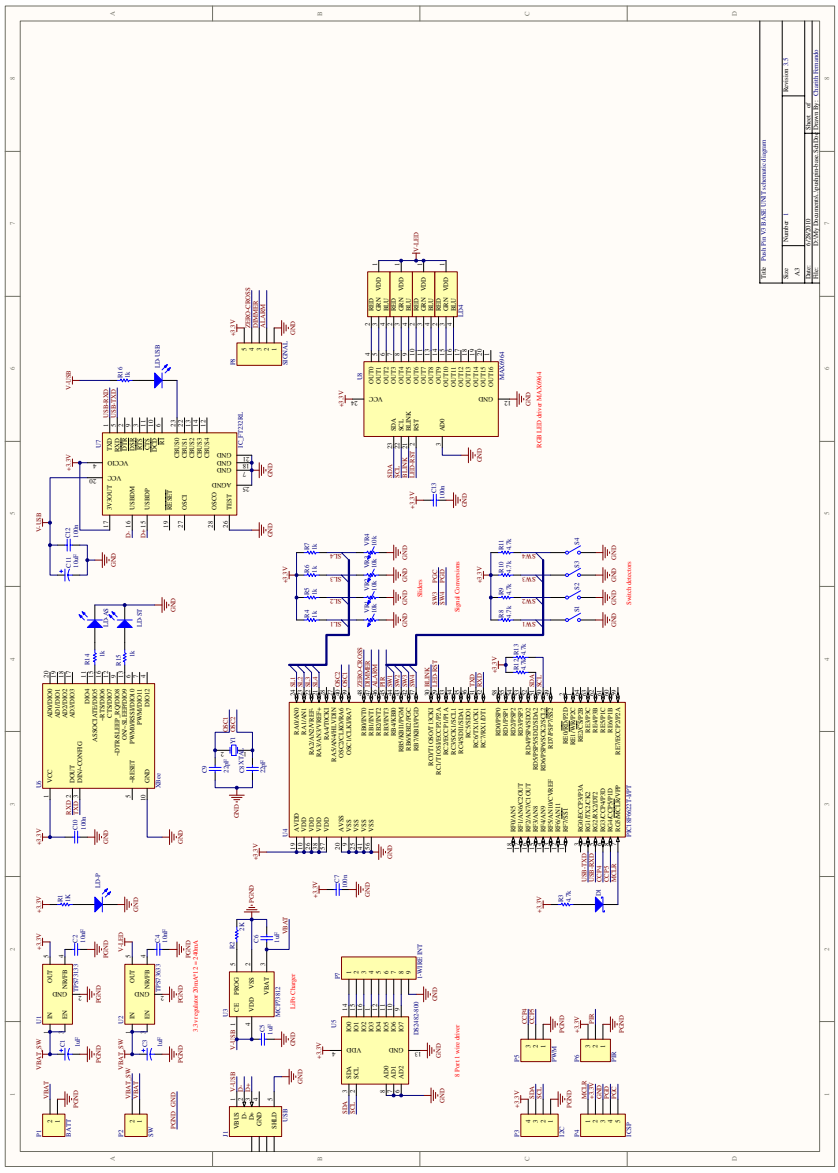

Figure A.2: pushPin base unit, schematic diagram

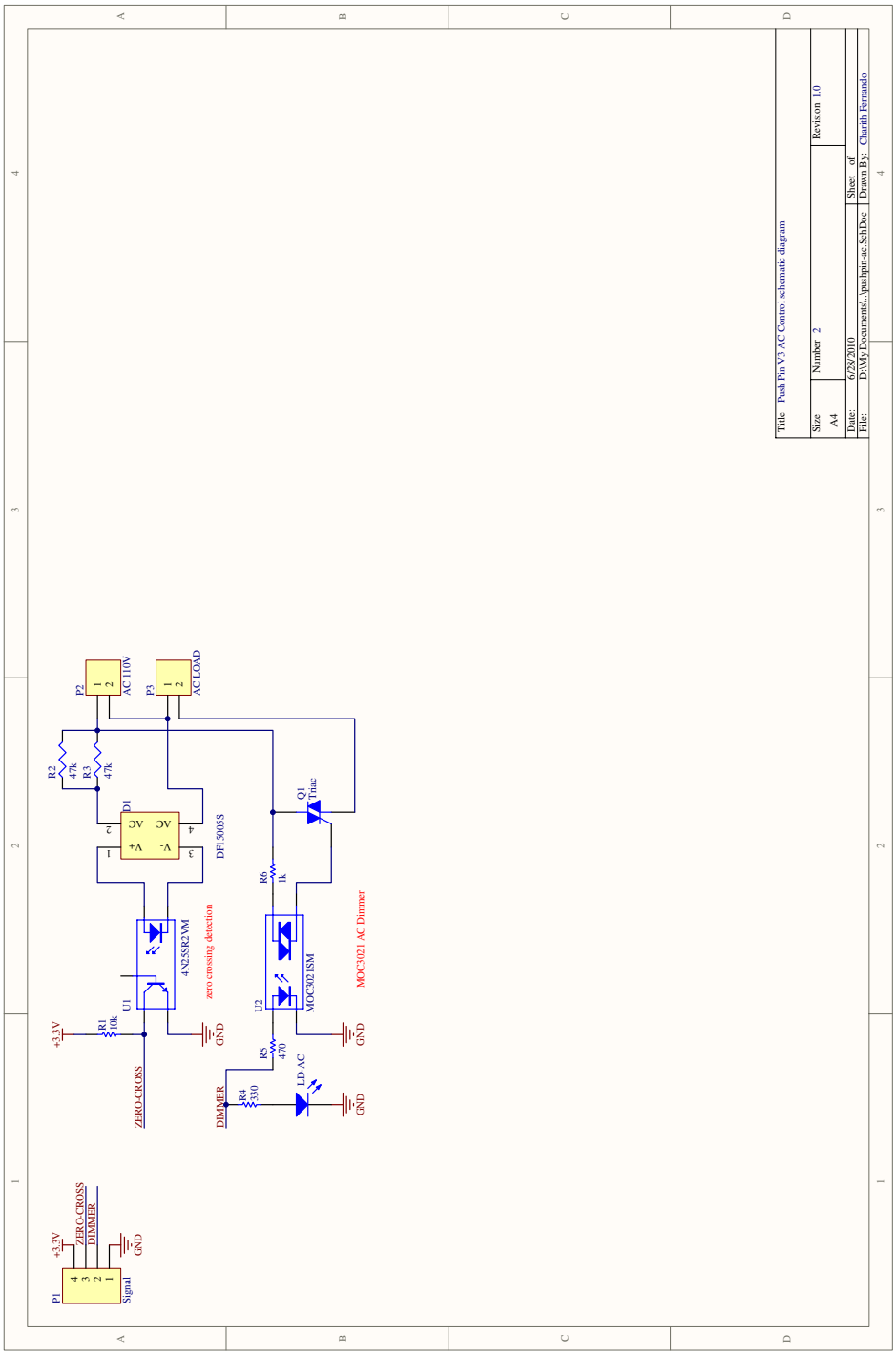

Figure A.3: pushPin ac controller, schematic diagram

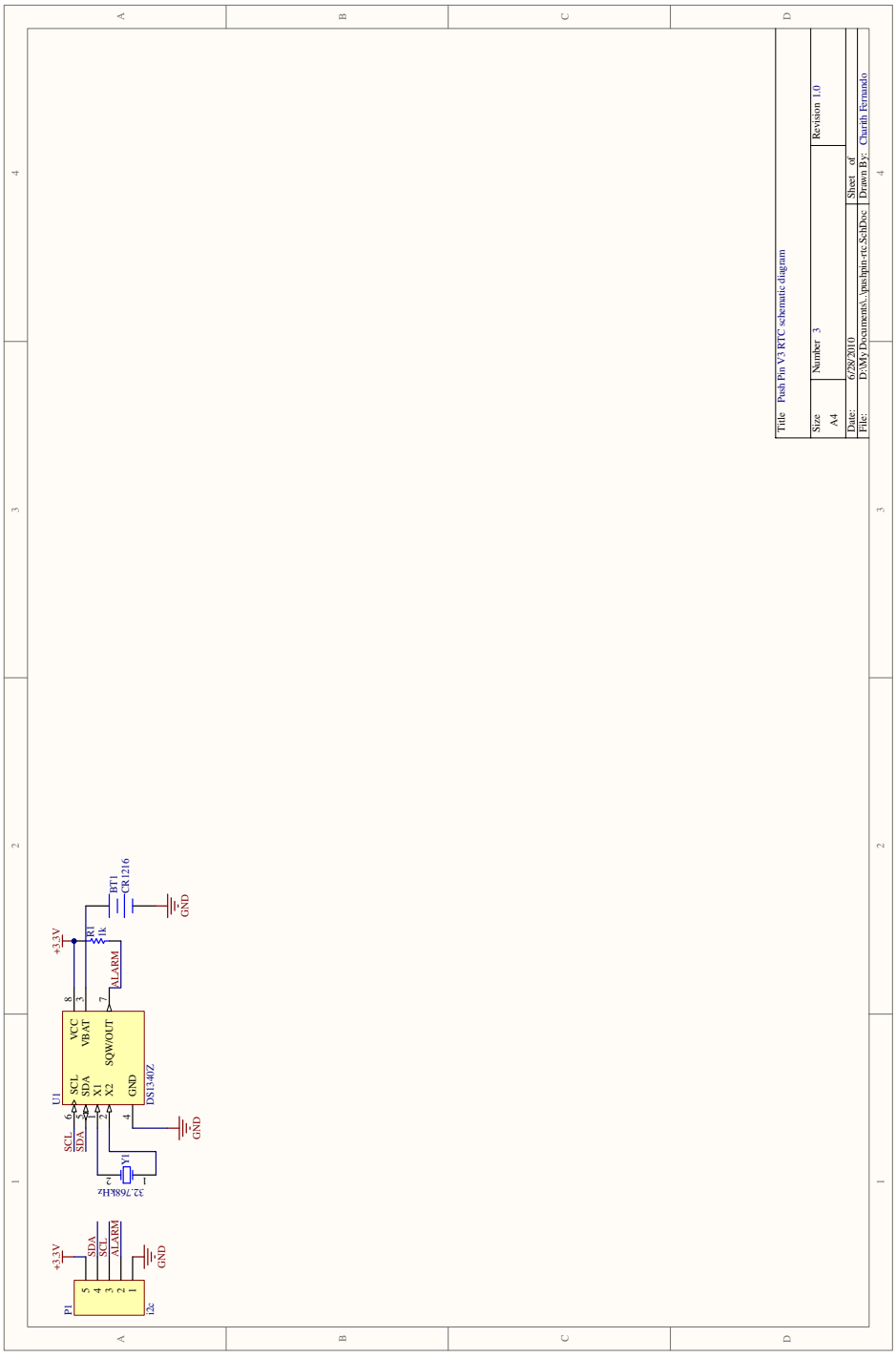

Figure A.4: pushPin real time clock, schematic diagram

A.2 PCB designs

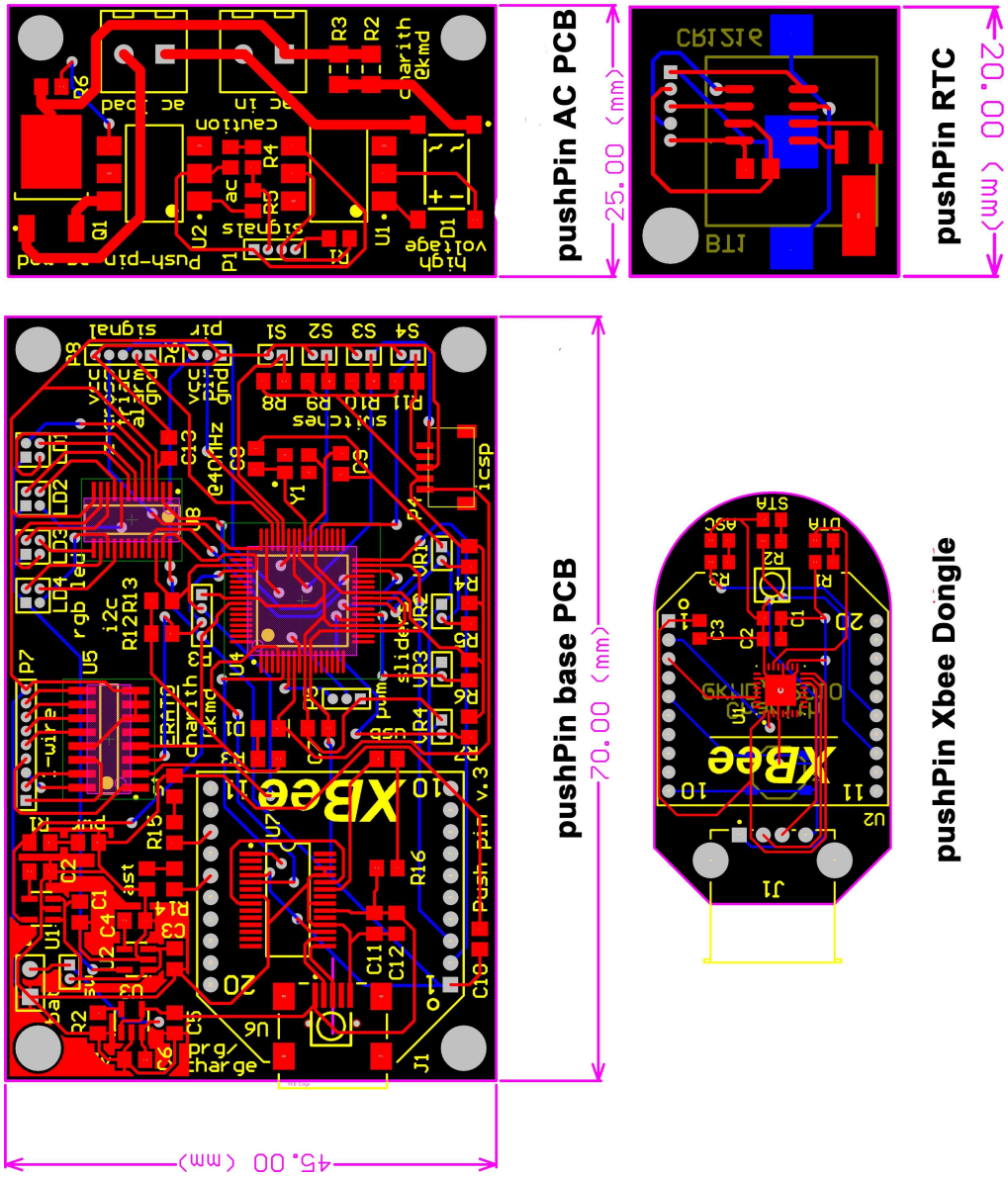

Figure A.5: pushPin PCB designs

A.3 CAD drawings

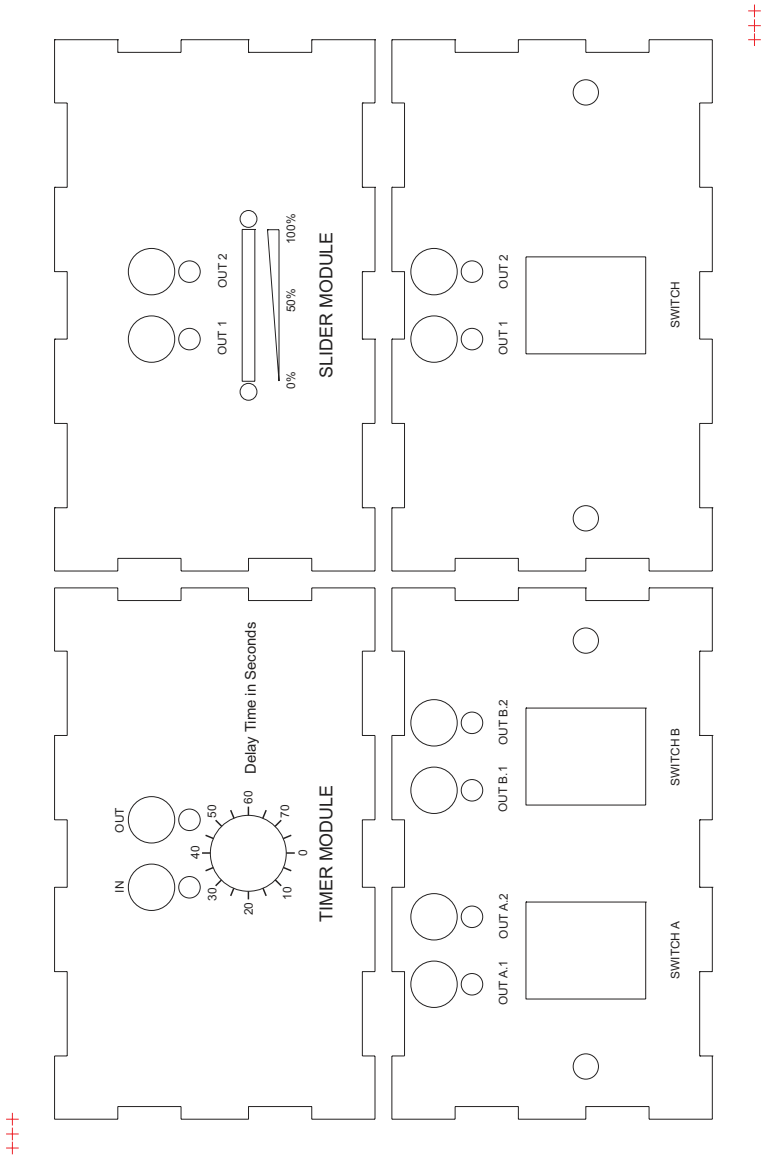

Figure A.6: pushPin CAD faceplate design 1

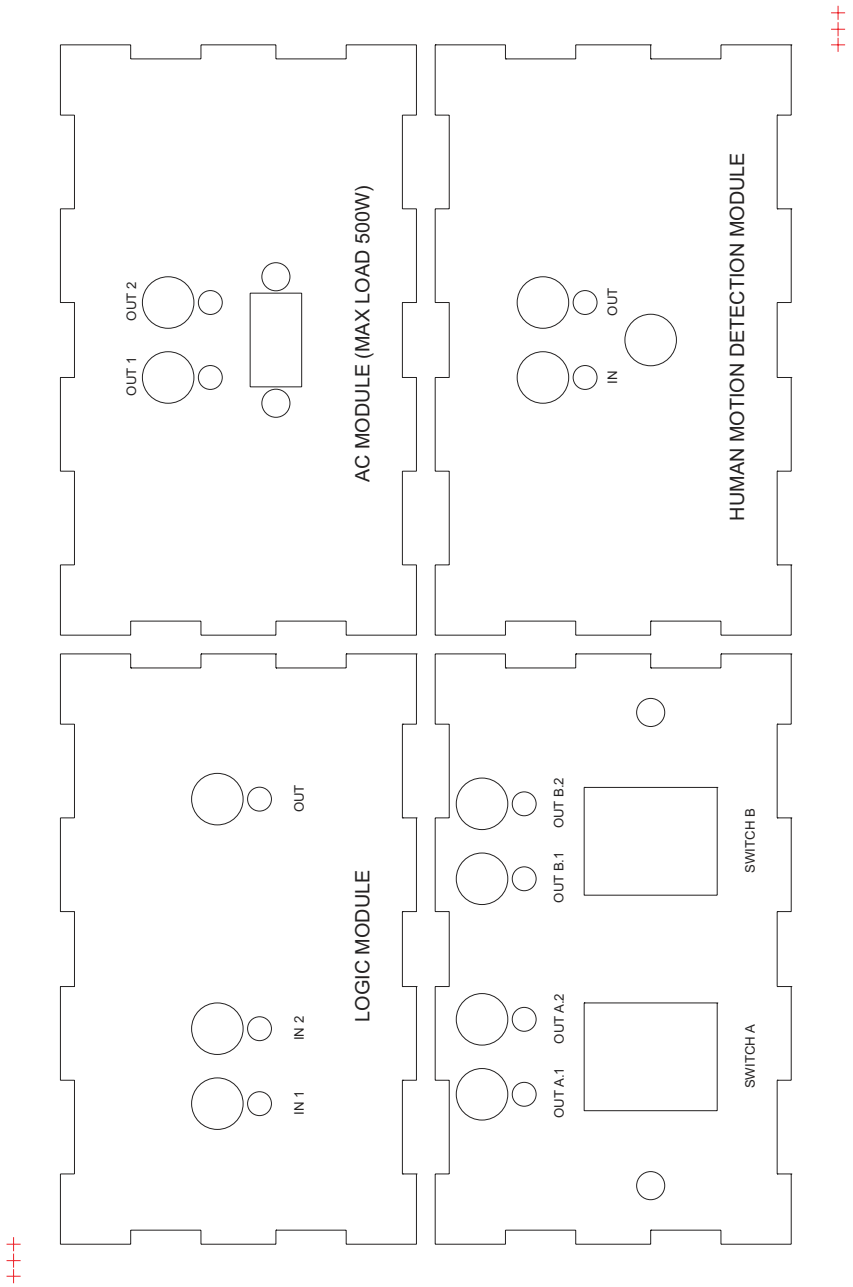

Figure A.7: pushPin CAD faceplate design 2