HUMAN PACMAN

GOH KOK HWEE

(B.Eng.(Hons.), NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2004

Abstract

Human Pacman is a novel interactive entertainment system that ventures to embed the natural physical world seamlessly with a fantasy virtual playground by capitalizing on mobile computing, wireless LAN, ubiquitous computing, and motion tracking technologies. Our Human Pacman research is a physical role-playing augmented-reality computer fantasy together with real human-social and mobile-gaming. It emphasizes collaboration and competition between players in a wide outdoor physical area which allows natural wide-area human-physical movements. Pacmen and Ghosts are now real human players in the real world experiencing mixed computer graphics fantasy-reality provided by using the wearable computers on them. Virtual cookies and actual tangible physical objects are incorporated into the game play to provide novel experiences of seamless transitions between real and virtual worlds. We believe Human Pacman is pioneering a new form of gaming that anchors on physicality, mobility, social interaction, and ubiquitous computing.

Acknowledgement

I would like to express my heartfelt thanks to the following people for their invaluable guidance and assistance during the course of my work.

- Dr Adrian David Cheok
- Ms Veron Ng
- Ms Liu Wei
- Mr Teo Sze Lee
- Dr Farzam Farbiz
- Mr Lee Shangping
- Ms Li Yu
- All others from Mixed Reality Laboratory (Singapore) who have helped me in one way or another.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Permeation of technology into everyday life is made easier when the human experience it creates is made associable with day-to-day encounters. The research work, Human Pacman, is based on the popular arcade Pacman from the 1980s, is a novel and entertaining game which seeks to bring about such association through stimulating multiple human senses and perception. It is a real-world-physical, social, and wide area mobile entertainment system that is built upon the concepts of ubiquitous computing, tangible human-computer interaction, and wide-area entertainment networks. Human Pacman is pioneering a new form of gaming that anchors on physicality, mobility, social interaction, and ubiquitous computing.

In this chapter, we will first look at the novel aspects of this work. Then some background information follows, including the discussion of some human-computer interaction (HCI) issues and how Human Pacman relates to it. Finally a list of publications of the work will be given.

1.1 Novelty

The research work encompasses several novel aspects of human computer interactive entertainment.

Firstly, the players immerse in role-playing of the characters Pacmen and Ghosts by physically enacting the roles. Players physically move around in a wide-area setting, performing tasks to reach their goals. Utilizing the high computing power of wearable computers and the underlying network support, Human Pacman takes mobile gaming to a new level of sophistication by incorporating virtual fantasy and imaginative play activity elements, factors which propelled the popularity of computer game [3] , with the implementation of Mixed Reality on the Head Mounted Displays (HMD).

Secondly, Human Pacman also explores novel tangible aspects of human physical movement, senses and perception, both on the player's environment and on the interaction with the digital world. For example to devour the virtual "enemy", the player has to tap on the real physical enemy's backpack, which is an instinctive action to "catch" the "enemy". By employing the philosophy of ubiquitous computing [4], we have implemented a system that embeds everyday physical objects with digital fantasy meanings. For example, players have to collect virtual special cookies by intuitively picking up physical treasure boxes laid across the game area. These Bluetooth embedded boxes when picked up will automatically communicate with the wearable computer by adding the special cookie to the inventory list of

the player.

Thirdly, users enjoy unrestricted movement outdoor and indoor while maintaining social contact with each other. Players interact both face-to-face with other players when in proximity or indirectly via the wireless local area network (WLAN).

Human Pacman ventures to elevate the sense of thrill and suspended disbelief of the players in this atypical computer game. Each of the novel interactions mentioned is summarized in Table 1.1.

1.2 Background

Human Pacman has its roots in embodied interaction [5] in the computer gaming arena. "Embodiment" is the way physical and social phenomena unfold in real time and real space as a part of the world in which we are situated, right alongside and around us. It is at the center of phenomenology which explores our experiences as embodied actors interacting in the world, participating in it and acting through it, in the absorbed and unreflective manner of normal experience. The paradigm of embodied computing explored in Human Pacman incorporates elements of ubiquitous, physical, tangible, and social computing.

Tangible computing

Even though Graphical User Interface (GUI) has been and still is the dominant paradigm for interactions with computers, we are increasingly encountering com-

Table 1.1: Detail descriptions of each novel features of Human Pacman.

Feature	**Details**
Physical Gaming	Players are physically role-playing the characters of Pacmen and Ghost; with wearable computers donned, they use free bodily movements as part of interaction between each person, and among objects in the real wide area landscapes and virtual environments.
Social Gaming	Players interact both directly with other players when they are in physical proximity, or indirectly via the Wireless LAN network by instant messaging. All Internet users can participate in the game by viewing and collaborating with real Human Pacmen and Ghosts.
Mobile Gaming	Players are free to move about in the indoor\outdoor space without being constrained to the 2D\3D screen of desktop computers.
Ubiquitous Computing	Everyday objects throughout the environment seamlessly have a real-time fantasy digital world link and meaning. There is automatic communication between wearable computers and Bluetooth devices embedded in certain physical objects used in game play.
Tangible Interaction	Throughout the game people interact in a touch and tangible manner. For example, Players need to physically pick up objects and tap on the shoulder of other players to devour them.
Outdoor Wide-Area Gaming Arena	Large outdoor areas can be set up for the game whereby players carry out their respective missions for the role they play.

putation that moves beyond the traditional confines of the desk and attempts to incorporate itself more richly into our daily experience of the physical and social world. Work on physical interaction started to appear in literatures in the early 90s

with the introduction of Computer-Augmented Environments [6] that have visioned the merging of electronic systems into the physical world instead of attempting to replace them as in virtual reality environments.

Over the years, a number of projects have explored this new paradigm of interaction termed tangible computing. Early attempts include Bishop's Marble Answering Machine [7] that has made a compelling demonstration of passive marbles as "containers" for voice messages; "Brick" by Fitzmaurice [8] that are essentially new input devices that can be tightly coupled to virtual objects for manipulation or for expressing action (e.g., to set parameters or for initiating processes). Ishii's "Tangible Bits" [9] coupled the idea of tangible computing with ubiquitous computing. In his work, users "grasp & manipulate" bits in the center of their attention by coupling the bits with everyday physical objects and architectural surfaces using small tangible objects that serve as physical icons ("phicons").

In Human Pacman the use of Bluetooth embedded objects explores tangible interaction in a ubiquitous computing setting. To register collection of a special cookie in the system, a player has to physically hold a wooden treasure box. Such graspable interaction uses physical and tactile skills we are intimately familiar with. Unlike in Tangible Bits, active communication is explored in Human Pacman. This is instantiated with the Bluetooth embedded device alerting the player of its physical proximity. Another important aspect of the design is the clever exploitation of the affordances of the object's physical properties whereby without prior training, players can intuitively associate the action of picking up the "special cookie" ob-

ject with the collection of it in their virtual inventory as well as having the action simultaneously occur in the virtual world.

Physical computing

People in the world today enjoy entertainment experience that they are in control and are involved physically [10]. In pre-computer age, games were designed and played out in the physical world with the use of real world properties, such as physical objects, our sense of space, and spatial relations. Nowadays computer games focus the user's attention mainly on the computer screen or 2D/3D virtual environments, thereby constraining physical interactions. However, there seems to be a growing interest in physical gaming and entertainment, even in industry. Commercial arcade games have recently seen a growing trend of games that require human physical movement as part of interaction. For example, dancing games such as Dance Dance Revolution and ParaParaParadise [11] are based on players dancing in time with a musical dance tune and moving graphical objects (see Fig. 1.1). However these systems still force the person to stand in more or less the same spot, and focus on a computer screen in front of them.

Players physically enact the roles of Pacman and Ghost in Human Pacman. Donning on a set of wearable computer whilst connected wirelessly to a network, a player interacts with the game world using well-acquainted methods such as capturing an enemy by physical touch. Such naturalistic activity greatly reduces the distinction between "interface" and "action" Mobility and natural interaction in

Figure 1.1: A player of the ParaParaParadise arcade game. © Konami.

the dualistic real-fantasy world gives a sense of immersion and physical involvement in the game.

Social computing

Research described by the Entertainment Software Association [12] shows that one of the top reasons why people like to play games is that it is usually a social activity people can enjoy with family and friends. With advancement in networking technology, social gaming has gained popularity since the introduction of networked games [12]. Networked games overcame the barrier of distance, enabling real people to play against each other over large areas. After all there is no opponent like a live opponent, since no current computer model can rival the richness of human interaction [13]. Nevertheless, even in networked computer games, social interaction

between players is limited since natural interactions such as behavioral engagement, and cognitive states are lost. Thus, by bringing players in physical proximity for interaction, Human Pacman brings networked social computer gaming to a new ground because humans enjoy being physically together, and socially interacting with each other [14]. Essentially, Human Pacman brings the exciting interactive aspects of networked gaming, and merges it with the real physical world, to allow a seamless real-time networked social contact between humans in both the real and virtual worlds simultaneously. Social mingling in the "catching" action brings along human perceptions, instincts, and reactions not possible without physical proximity.

Similar work

Human Pacman has also aspects derived from pioneering work that has been developed on ubiquitous gaming. Multi-player mobile gaming is demonstrated in "Pirates!" [15]. "Pirates!" implements the game on PDAs with proximity sensing technology to incorporate a player's contextual information (such as physical co-location of players and objects in the world) into the game context as important elements of the game mechanics. However, visual and sound effects of game play are limited by the relatively low computing power of PDAs. Augmented Reality (AR) and Virtual Reality (VR) cannot be implemented; therefore immersive experience is rather limited due to the flat 2-D display used on PDAs. The E3 project [16] examines the essential elements of free play, and multi-user social

interaction. It focuses on human-to-physical interaction and human-to-human interaction. However it does not explore large-scale configuration where users walk around.

Pac Manhattan [17] is a project similar to Human Pacman. It is a large-scale urban game that utilizes the New York City grid to re-create arcade Pacman. In this analog version of Pacman a player dressed as Pacman will run around the Washington square park area of Manhattan (New York, US) while attempting to collect all of the virtual "dots" that run the length of the streets. Four players dressed as the ghosts will attempt to catch Pacman before all of the dots are collected. Using cell-phone contact, each player on the street is teamed with a Controller who updates the position of street player as she runs through the streets. At every intersection, the player on the street updates the Controller with her position which is then updated via software and shared with the other Controllers as well as being broadcast over the Internet for viewers. Fig. 1.2 shows a Pacman in Pac Manhattan communicating with Controller through cell-phone.

Like Pac Manhattan, Human Pacman also explores new synergy in game play when games are removed from their "little world" of table-tops, televisions and computers, and placed in the larger "real world" of street corners and cities. Progress of the game is also shared via the Internet to Helpers from all around the world. However Human Pacman employs technology that couples GPS tracking and step counting to automatically track mobile players in the game area, rather than having the players report their positions periodically through cell-phones. Hence Human

Figure 1.2: Pacman in Pac Manhattan communicating with Controller through cell-phone. © PacManhattan.

Pacman, unlike Pac Manhattan, is able to provide visualization and connection between the online players and the actual real-time positions of the physical players. Furthermore, in Pac Manhattan, Pacman does not know which "dots" have already been eaten in the game area as there is no visualization of the virtual game world by the physical players. The only contact with the game is through mobile phone communication. This is in contrast to Pacman in Human Pacman where the player, connected to the game engine primarily through wireless LAN, has constant AR visual display of virtual cookies that lies in her path. By looking at a virtual map in her cockpit within her HMD display, she is able to know the cookies layout in the game area.

"Can You See Me Now?" (CYSMN) [18], [19], [20], [21] is a chase game where online players are chased across a virtual city by three performers who are running through the actual city streets. These runners, who are professional performers, run

through actual city streets equipped with handheld computers, wireless networking (using 802.11b), and GPS receivers as shown in Fig. 1.3. They chase up to 15 online players through a virtual model of a city. Online players are dropped into a 3D model of the hosting city. They can move through this model with a fixed maximum speed, access a city map view, see themselves represented as running avatars, see other players' and runners' positions, and exchange text messages with them. They also hear the runners' walkie-talkie communication as a live audio stream. They must avoid the runners; if a runner gets within five virtual meters of an online player, the player is seen and out of the game.

Figure 1.3: A runner in CYSMN with a handheld computer. © Equator Project.

Human Pacman is very similar to CYSMN with the positions of mobile players being tracked by GPS and updated online in real-time. Both games allow online players to participate and interact with mobile players through the Internet. However in CYSMN visuals are limited to a non-immersive 2-D display on the handheld computer as shown in Fig. 1.4. Runners interact with the game engine only through

their 2-dimensional positions (x,y) in the game area. In Human Pacman the head orientation of the mobile players, in addition to their physical positions, are closely and accurately tracked. An AR view, rendered based on these data, is projected in the HMD the player dons. This helps to create an immersive experience for the mobile player. In addition mobile players in Human Pacman interact with the game engine through physical interaction with their environment through finding and touching physical "treasures" in the game area. Such tangible interaction with real physical object in the game area is absent from CYSMN.

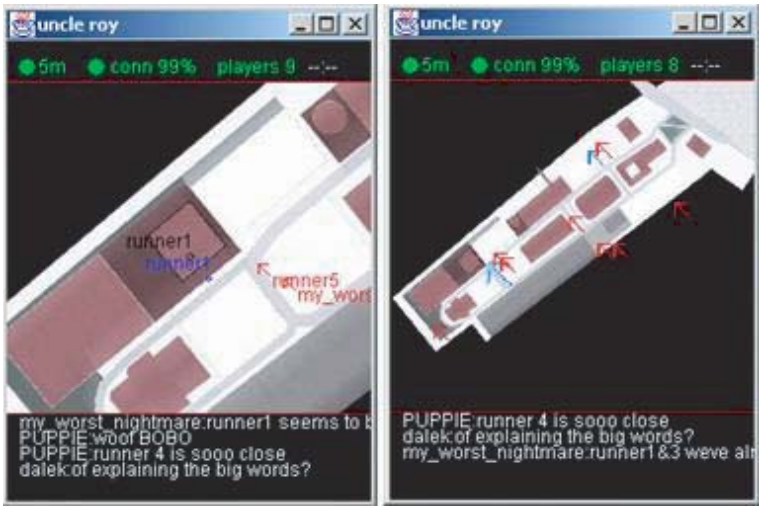


Figure 1.4: Runner's visual interface in CYSMN. © Equator Project.

Even though Human Pacman uses AR techniques as part of its interface, it is only for providing a comprehensive user interface for the players.

There were some previous works done which focus on using AR in entertainment. AR2 Hockey [22] , seen in Fig. 1.5, is a system that allows two users to hit a virtual puck on a real table, as seen through a HMD. AquaGaunlet [23] is a

multi-player game where players fight with strange invaders coming from the virtual world through some egg-shape objects into the physical space. These games are played in a small and restricted area, with limited movement, and little interaction with physical space. There is no exploration on the physical environment the player is in.

Figure 1.5: AR2 Hockey. © Mixed Reality Systems Laboratory Inc.

Touch-Space [24] is an embodied computing based mixed reality game space. Players move within a predefined area collecting treasures and avoiding danger spots. Wearing a HMD they immerse in an AR world fighting witches and finding treasures. Unlike in Human Pacman where mobile players are tracked with an integrated GPS and step-counting device, tracking device in Touch-Space is mounted on top of the game space and the extent of coverage of this tracking device constrains the size of the game space that can be defined to that of a small room. Users experience tangible interaction in Touch-Space through finding treasures by opening boxes with visual markers in it. This is shown in Fig. 1.6. Visual tracking

rather than the tangible interaction triggers the effect of finding the treasure. In Human Pacman finding and touching of Bluetooth embedded objects triggers an immediate effect of collection of the "treasure", thus creating a different experience for the user.

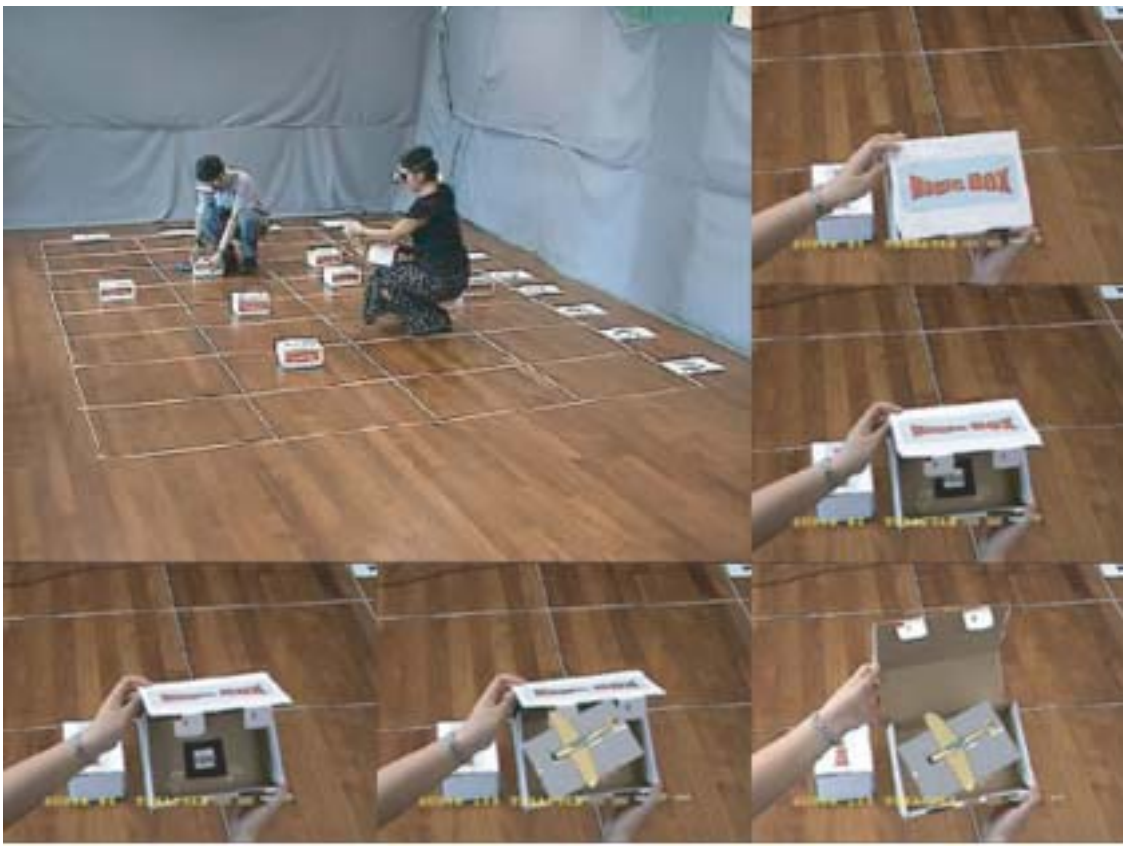


Figure 1.6: Collection of treasure in Touch-Space.

In [25] Starner created a new mobile experience by introducing AR to wearable computing. An important mobile AR game is ARQuake [26], which is an AR extension of the popular computer game Quake. Using wearable computer equipped with GPS, ARQuake can be played indoor and outdoor using GPS tracking and fiducial marker tracking. However it is a single player game with practically no

social interaction. Apart from the use of a haptic gun that player carries around for shooting at enemies, ARQuake does not explore other tangible interaction such as with the surroundings. In Human Pacman, Pacman has to explore her surroundings to find hidden Bluetooth embedded object she is alerted to.

1.3 History of Project

The basis of the Human Pacman project came from an exploratory research work on wearable computer by Ms Fong Siew Wan. She was working on an augmented reality navigation system, which uses data from an inertia sensor and a GPS unit to determine position and orientation of the user. The embedded computer used was of low processing power and poor graphics capability, hence limiting the complexity of the application developed.

Human Pacman project was conceptualised by a group of researchers, including the author, in November 2002. As the Human Pacman is a large and complex system, it had to be built in stages. As the project progresses, new researchers came on board to work on different parts. Development of the system is currently still ongoing.

This thesis encapsulates the contribution by the author from 2002 to 2004.

1.4 Publications

Human Pacman has been published and presented in a number of papers and conferences. Demos has been shown in CHI 2004, Vienna (see Appendix A, Sec D.1), and in ACE2004, Singapore (see Appendix A, Sec D.2). The following is a list of conferences and publications it has been featured in.

- *Personal and Ubiquitous Computing Journal, Volume 8, Issue 2 (May 2004) (pg 71-81)*

  "Human Pacman: A Mobile, Wide-Area Entertainment System Based on Physical, Social, and Ubiquitous Computing."

- *5th International Symposium on Human-Computer Interaction with Mobile Devices and Services, Mobile HCI 2003, Udine, Italy (Pg 209-223)*

  "Human Pacman: A Mobile Entertainment System with Ubiquitous Computing and Tangible Interaction over a Wide Outdoor Area."

- *Proceedings of the 2nd Workshop on Network and System Support for Games, NetGames 2003, California, USA (pg 106-117)*

  "Human Pacman: A Sensing-Based Mobile Entertainment System With Ubiquitous Computing and Tangible Interaction."

- *Extended abstracts of the 2004 Conference on Human Factors in Computing Systems, CHI 2004, Vienna, Austria (pg 779-780)*

  "Human Pacman: A Wide Area Socio-Physical Interactive Entertainment

System in Mixed Reality."

- *Proceedings of International Conference on Advances in Computer Entertainment Technology 2004, Singapore (pg 360-361)*

  "Human Pacman: A Mobile Wide-Area Entertainment System based on Physical, Social and Ubiquitous Computing."

- *Proceedings of SICE2004 Annual Conference 2004, Sapporo, Japan (pg 1662-1667)*

  "Ubiquitous Human Media for Social and Physical Interaction."

- *International Workshop of Interaction Design and Children 2004, Maryland, USA*

  "Human Pacman: A Mobile Outdoor Entertainment System for Children."

- *IFIP 18th World Computer Congress, Toulouse, France, August 2004*

  "Connecting the Real World and Virtual World Through Gaming."

- TODAY (23rd June 2004) – English newspaper in Singapore

- Lian He Zao Bao (14th June 2004) – Chinese newspaper in Singapore

- The New Paper (9th August 2004) – English newspaper in Singapore

1.5 Prelude to Subsequent Chapters

Chapter 2 will look at how Human Pacman game is played. The main concepts involved in playing the game will first be introduced. This is followed by an introduction to the individual player role (Pacman, Ghost, and Helper) in the game. With a better understanding of the game, a run through of the actual game play will be given to see how the individual roles inter-mingle with each other.

Chapter 3 will look at the overall system design, including the hardware and software aspects of the system. In the hardware aspect, focus will be placed on make-up of the wearable computer and the Bluetooth embedded object. The software aspect will start off with an overall system data flow overview. This is followed by a description of the software flow in the server, Helper and wearable computer systems. The design of the backpack used for housing the wearable computer will be described. The chapter ends with the highlighting of some issues encountered during the research work.

Chapter 4 is a discussion of the Human Pacman system based on a user study that was done. The questions used will be detailed in the chapter and the results will be discussed.

Chapter 5 looks at an indoor maze version of the Human Pacman system. The chapter will start by looking at how the game has been implemented in the maze version. This is followed by a description of the location tracking system employed.

Chapter 6 summarizes the thesis, highlighting areas of possible improvements

in the future. The chapter ends with a conclusion to future possibilities that the Human Pacman entertainment system entails.

Appendix A is a tutorial for setting up and using the wearable computer. A detailed description of the connections between the individual modules within the wearable computer will be given. Settings for configuring the system follows and the chapter ends off with a step-by-step guide to starting the Human Pacman program.

Appendix B will contain listings of the source codes for the various firmware used in the lab-built hardware.

Appendix C gives the various PCB layout designs used for fabricating the lab-built hardware.

Appendix D will briefly outline the Human Pacman demos done at CHI 2004 and ACE 2004.

Appendix E shows newspaper articles featuring Human Pacman.

Chapter 2

Game Play

Pacman was introduction by Namco to Japanese arcade fans in 1979. Originally inspired by "paku", a Japanese folk hero known for his appetite, it is drawn as a simple, solid yellow circle with a missing wedge for a mouth [27]. The game was first christened as "Puckman", from the Japanese phrase pakupaku, meaning to flap one's mouth open and close. It was later renamed to Pacman. After the distinctive theme music plays, players find themselves guiding Pacman around a single maze eating dots, while avoiding the four ghosts who escape from a cage in the middle of the screen and will end Pacman's life if they touch him. The maze is shown in Fig. 2.1. In each corner of the square playfield is a large dot that when eaten will turn the ghosts blue for a brief period, during which time the table turns and Pacman can eat the ghosts, leaving only the apparently indigestible eyes which make their way back to the cage for reincarnation into another ghost. The game is deceptively simple, with only a four-position joystick needed to guide Pacman

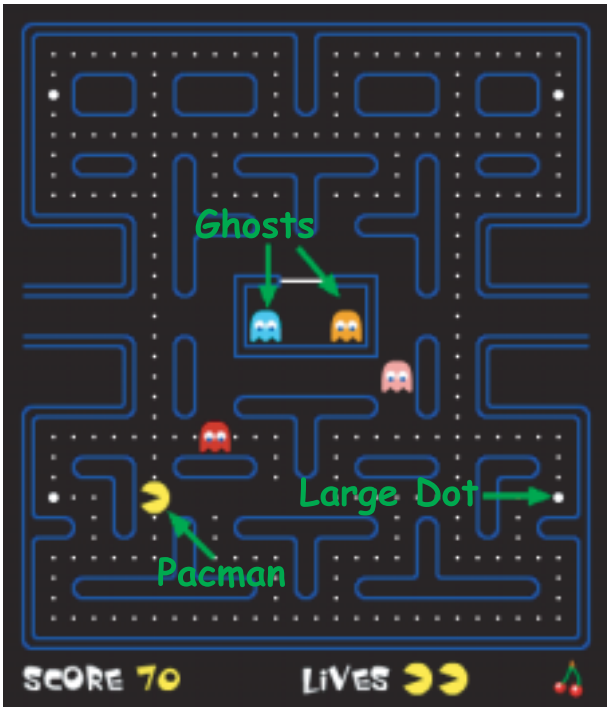


Figure 2.1: Original arcade Pacman maze. © Namco Limited.

around the maze, but with each successive screen the ghosts get faster and their time of blue-invulnerability less.

Pacman has gone through numerous stages of development including Ms. Pacman (1982), Pacman Plus (1982), Super Pacman (1982), Baby Pacman (1982), Jr. Pacman (1983), Professor Pacman (1983), Pac & Pal (1983), Pac-Land (1984), Pac-Mania (1986), Pac-Attack (1993). A VR version of the game, Pacman VR, was introduced in 1996, and a 3D version of the game, Pacman World, in 1999. Fig. 2.2 shows screen shots of Pacman VR and Pacman World.

Despite the numerous variation of the game, the ultimate goal of the game remains fundamentally unchanged. Human Pacman has been designed to be in close resemblance to the original Pacman in terms of game objectives so that the

Figure 2.2: Screen shots of Pacman VR [1] (© KLOV.com) and Pacman World [2] (© Sony Computer Entertainment Inc.)

players' learning curves are very much levelled to the point that they can pick up the game in very little time and enjoy the associated familiarity.

In this chapter, we shall examine the main game concepts and roles in Human Pacman. This will be followed by details of the actual game play and elaboration on the gaming experience of the players. Having better understood how the game is to be played, the subsequent chapter will examine how Human Pacman system has been implemented to achieve the mentioned game play and experience for the players.

2.1 Main concepts

The system aims to provide users with a new human-computer social interaction and entertainment experience. Users get a unique experience, moving around in

physical space and interacting with others as if they have become part of a fantasy world where the virtual Pac-World and the real-world has become one. Before looking at the actual game play, a few main game concepts have to be introduced:

a. Team Collaboration - The players are assigned to two opposing teams, namely the Pacman team and the Ghost team. The former consists of two Pacmen and two Helpers; correspondingly, the latter consists of two Ghosts and two Helpers. Each Pacman\Ghost is in coalition with one Helper, promoting collaboration and interaction between the users. Pacmen and Ghosts are mobile players in the game, moving in the game area physically. Helper players are essentially participating in the game play remotely via a computer terminal connected to the game engine through LAN. Hence Human Pacman can effectively be expanded to include online players from any part of the world, who can view and collaborate, in real-time over the Internet, with real human Pacmen and Ghosts who are immersed in the physical playground.

b. Ultimate Game Objectives - Basically the goal of the Pacman team is to collect all virtual plain cookies and hidden special cookies in Pac-World. The Ghost team on the other hand attempts to stop the Pacman team from reaching their goal by devouring all the Pacmen in Pac-World. To add to the excitement of the game play, after a Pacman "consumes" a special cookie, she transforms into a Super-Pacman for a short period of one minute, giving her the immunity against Ghosts' "attacks"

and gaining Ghost-devouring capability.

c. The Nature of Pac-World - Pac-World is a fantasy world existing simultaneously in physical reality, augmented reality (AR) and virtual reality (VR) mode. Pacmen and Ghosts, who are walking around in the real world with their networked wearable computers and head mounted displays (HMD), view the world in AR mode. Helpers, on the other hand, can view it in VR mode since they are stationed in front of networked computers. More importantly there is a direct and real time link between the wide-area physical world and the virtual Pac-World at all times, thus providing the users with a ubiquitous and seamless merging of the fantasy digital world and the realistic physical world. In Fig. 2.3 the 2D map of the selected game play area in our university campus and the 3D map of Pac-World are shown side-by-side. The central library in the center of the former corresponds to a virtual building in the center of the latter. Similarly, as shown in the figure, roads in the real physical space has a corresponding path routed in the virtual space. We have converted the real world to a fantasy virtual playground by ingraining the latter with direct physical correspondences.

Virtual cookies are scattered in a maze like manner (for example on real footpath) over the real physical game area, awaiting collection by the Pacmen. In the AR mode, they are displayed in the first-person perspective of the mobile player, which is dependent on her physical position and head motion. This is achieved using two sensor modules, namely the Dead Reckoning Module (DRM-III) and In-

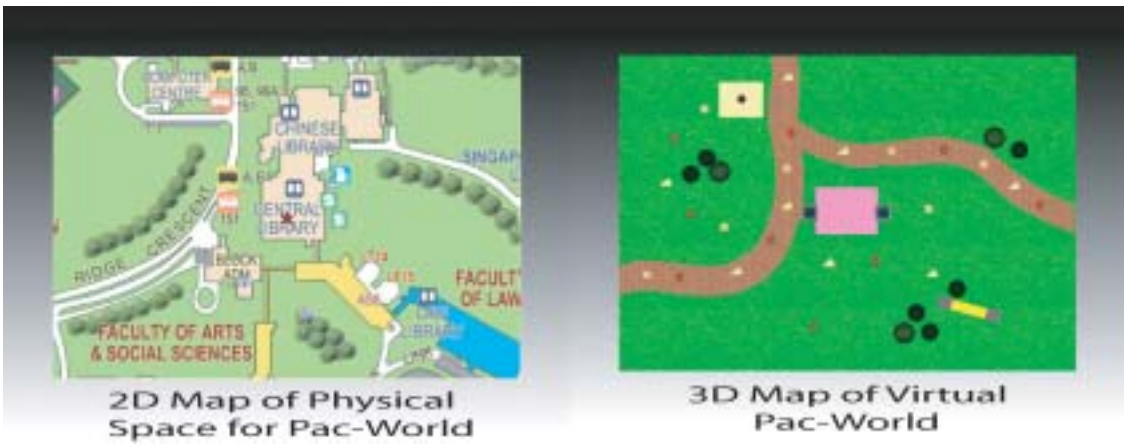


Figure 2.3: 2D map of game play area and its corresponding 3D map of Pac-World

ertiaCube2 which tracks the position and head orientation of the user respectively.

The real-time position and head orientation of each mobile user is sent periodically (between 10 ms to 21 ms) to the server through wireless LAN. Upon receiving the position data, the server sends an update to each wearable computer detailing the position of other mobile players, as well as the positions of all "non-eaten" plain cookies. This essentially maintains a tight link between the physical world, AR world, and the VR world. Fig. 2.4 shows a correspondence between the physical real world and its virtual Pac-World representation. Each physical location in the physical world has a corresponding position in the virtual world. The 3-dimensional physical position (x, y, z) of each player is being tracked real-time and her avatar is placed in the virtual world accordingly as seen in the figure. Similarly the head orientation (row, pitch, yaw) of each player is also being tracked real-time and her avatar in the virtual world is rotated according to it such that the avatar faces where she is physically facing. Hence the actual physical location and orientation

Figure 2.4: Correspondence between the physical world and virtual Pac-World of each player is reflected closely in the virtual Pac-World.

2.2 Pacman, Ghost, and Helper

Pacman has to physically move within the game area to collect all virtual plain cookies overlaid in the real world, as seen through her HMD (see Fig. 2.5). In addition she has to find and collect special cookies in virtual Pac-World. These are directly linked and represented by Bluetooth embedded objects shown in Fig. 2.6. This creates a sense of presence and immersion within the virtual Pac-World, as well as a feeling of active participation in the real world.

The Ghost can devour a Pacman by tapping on a capacitive sensor attached to Pacman's shoulder. Likewise, a Ghost can be devoured by Pacmen endowed with Ghost-devouring power. She should avoid so when playing this role. Such tangible

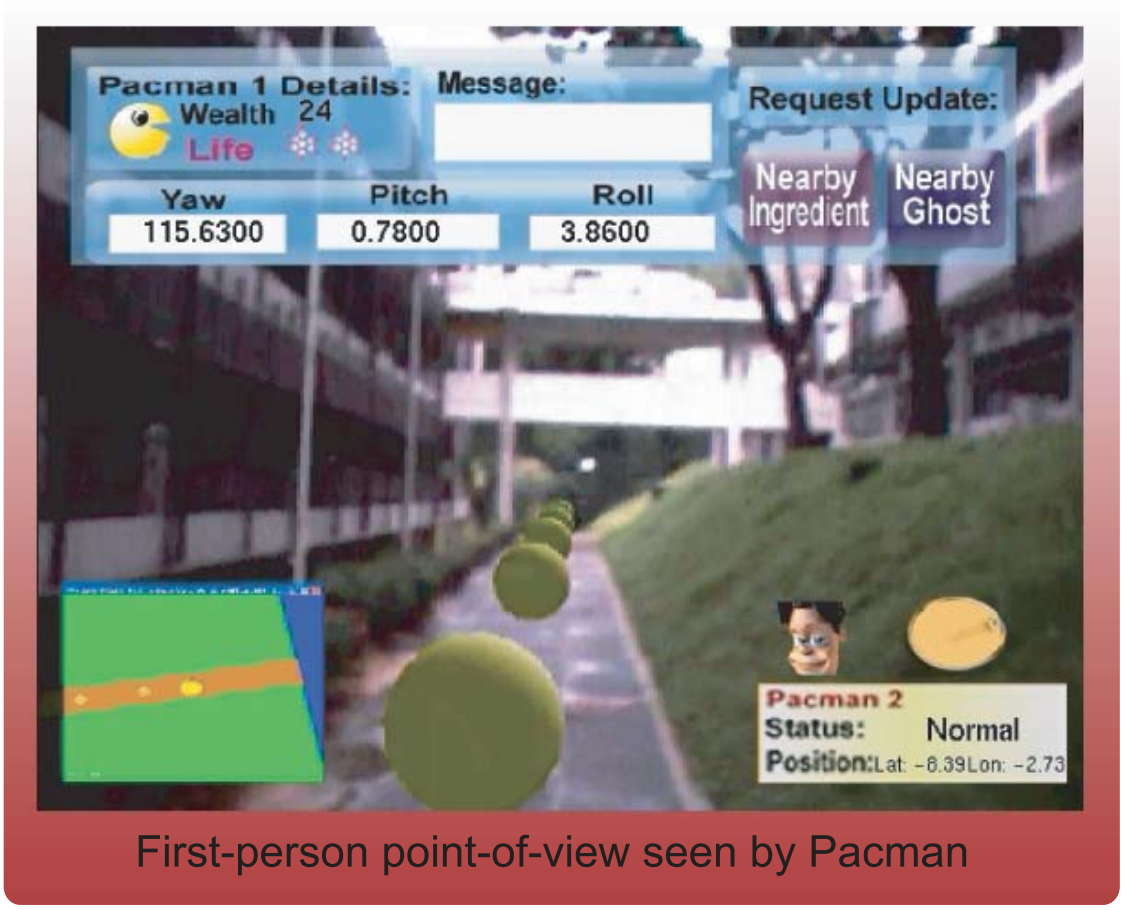


Figure 2.5: First person view of Pacman.

physical interaction between human, commonly found in traditional games such as hide-and-seek and the local classic "catching" game, is now revived in this computer gaming arena.

Each Pacman and Ghost will be assigned a partner Helper who acts as an intelligence, advisor, and coordinator in her quest to achieve her goal. To enhance the gaming experience for both Pacman and Ghost, these players in the physical world with wearable computer are not able to see enemy mobile units (positions of the enemies are not shown in the virtual map, and there is no augmented reality

Figure 2.6: Bluetooth embedded object.

labelling on them) and hidden special cookies. As shown in Fig. 2.7 the Helper, who is in VR mode and sees all, guides her partner by messaging her with important information. Her partner uses the Twiddler2, a handheld keyboard and mouse hybrid, to send predefined messages to her. This is shown as text overlay on the VR world display that Helper sees. In response the Helper types text messages using the keyboard and sends this to her partner. This information is displayed visually on the top control panel within her partner's HMD view. Such two directional communication promotes collaboration and interaction between players through the internet.

Figure 2.7: Close collaboration between Pacman and her Helper.

2.3 Actual Game Play

Collection of plain cookies: Pacman collects a cookie by walking through it. Such physical action is reflected visually in Pac-World through the disappearing of the cookie in both the AR and VR mode. In Fig. 2.8, the top images show the HMD view of the Pacman player as she collects a cookie. When she walks through the cookie, the cookie disappears from her AR view. This collection is also reflected real time in the virtual Pac-World (seen by Helpers) and Pac-World map (seen by both Pacmen and Ghosts) through the disappearing of the cookie in the corresponding location. This is done by real-time physical location tracking of the player. When the location of the Pacman is within a predefined radius of the virtual cookie, the

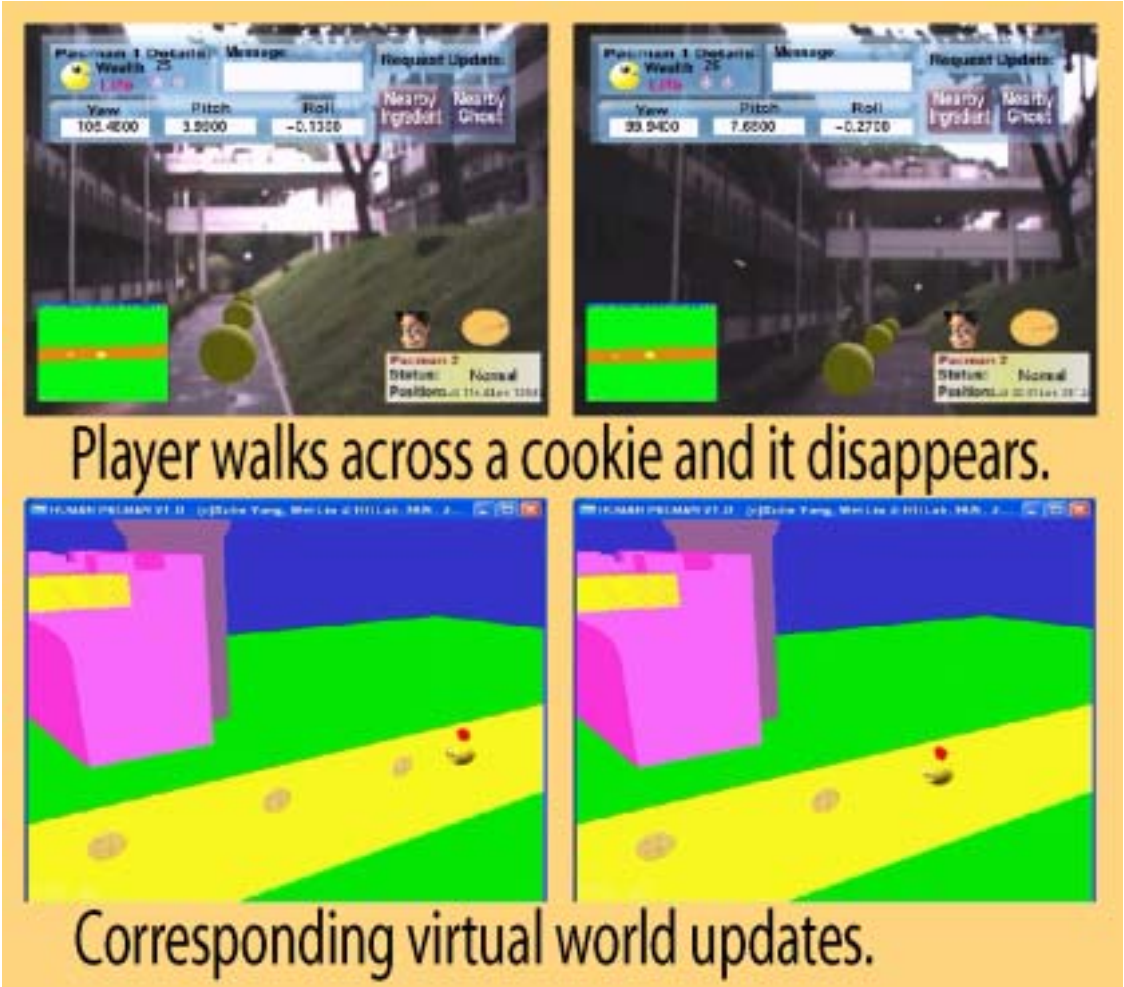


Figure 2.8: Pacman collecting cookies.

cookie is deemed to have been collected and this is reflected on the respective views. From the figure it should also be noted that the Pacman avatar in the virtual world has the same orientation as the visual direction the player is looking at (i.e. along the path). This is accomplished by real-time tracking of the head orientation of the player which, apart from giving the orientation of the avatar, is also needed for the rendering of the virtual cookie in the correct position within the HMD view (i.e. along the path as correspond to what is seen in the virtual world).

Ghosts are not able to collect cookies. Although a Ghost is not able to see enemy Pacman on the map, the disappearing of cookies in her map can give her a hint as to where to find a Pacman. Therefore a Pacman has to be careful as her physical interaction with the real world (i.e. movement) can be digitally reflected in the virtual world, and be made used of by a Ghost. Novelty is again seen in such intimate relationship between interaction in the physical world and its effect in the fantasy virtual world. Neither physical distance nor mobility could restrict each player from seeing this effect real-time as all players, including the Ghosts, can see an update of the virtual map in real-time.

Collection of special cookies: Pacman collects a special cookie by touching real Bluetooth-embedded objects placed in different parts of the game area. In Fig. 2.9, a sequence of pictures shows a Pacman collecting a special cookie. When the Pacman is within range of the Bluetooth object (about a distance of 10 meters), communication takes place between the wearable computer and the Bluetooth device. An alert message will be shown in the player's HMD display.

The player has to hunt for the Bluetooth embedded object in the surrounding physical area upon receiving the alert message, thus adding elements of fun and adventure to the game play. Having found the object, collection is done simply by physically holding the object in her hands. Once haptic data is collected by the touch sensor, the Bluetooth device embedded in the object will send an alert message to the wearable computer, which will in turn be relayed to the server.

Figure 2.9: Sequence of pictures showing the collection of a special cookie.

The server then proceeds to update its database as well as to inform the Helper systems of the collection. The collection of the special cookie exemplifies a natural tangible interaction involving physically interacting with this object through human touch. Pacman is able to hold a real object naturally in her hands as should be in real-life treasure finding. Such tangible action provides the player a sense of touch in the fantasy domain of the game play. The collected special cookie will be kept in a virtual inventory list. As seen in the figure, collection is shown by an addition of an icon to the inventory list after the special cookie has been collected. Pacman need not lug the physical object with her as she has collected the special

cookie virtually.

Collaboration between players: There is an essential element of collaboration in the game play between a Pacman\Ghost with her Helper. The Helper is in a good position to assist her partner as she has a complete view of Pac-world all the time, including the positions of all players and special cookies. Furthermore as Helpers within the same team could be physically close or could communicate with each other readily (e.g. through phone or online chat), they are able to collaborate between themselves and work out a strategy to achieve the team's goal. For example Helpers from the Ghost team could coordinate the movement of the Ghosts so as to corner one of the Pacman. The advantage of this setup is that social interaction and collaboration is significant between Helpers, as well as between Helpers and her partner.

A Pacman\Ghost can request for information, such as the update of the position of the enemy or the location of special cookies, by sending predefined messages to her Helper. This is done with a handheld Twiddler2 inputting device. In response, the Helper sends text messages to her partner with the requested information. This information is displayed visually on the top control panel within the HMD of the mobile unit. With the use of wireless LAN, mobility does not affect the ability of the Pacman\Ghost to constantly communicate with her Helper. It is also noted that since Helpers can be connected from anywhere in the world, this configuration allows for a global social collaboration with the physical players immersed in the

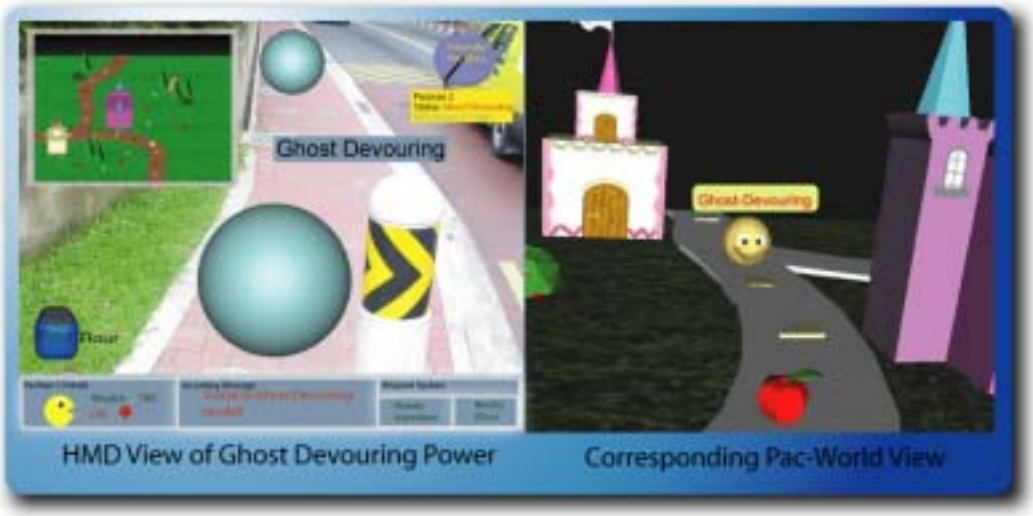


Figure 2.10: HMD display and the corresponding VR mode view.

Pac-World.

Use of special cookie: All special cookies can only be used once. When a Pacman consumes a special cookie, she transforms into a Super-Pacman. She will see an alert message in her HMD, informing her of the 1 minute Ghost-devouring power she has acquired. In real-time a label describing her acquired-power will also be placed on top of her Pacman avatar in the VR mode. This serves to inform all Helpers, including those from the Ghost-team, of her ability. This is illustrated in Fig. 2.10.

Devouring enemy player: To devour a Pacman, a Ghost must physically touch Pacman's backpack (see Fig. 2.11) to trigger the touch sensor located there. The same applies when a Pacman with Ghost-devouring capability devours a Ghost.

Figure 2.11: Ghost catching a Pacman.

Devouring involves tangible physical touch contact between two players. As close proximity is involved, other forms of human interaction come into play. The act of devouring makes the game more tangible and fun by involving more types of natural physical movement. When a Pacman player is the prey, her agility determines the "life-and-death" of her virtual Pacman role. Hence, not only tangibility is brought to play in this fantasy world, but also other human perceptions and instincts. Thus, this computer game provides the benefits of natural wide area free bodily movements as part of humanistic interaction between each person.

Ending the game: The game ends when either team meets their goal or when a time limit of fifteen minutes (chosen arbitrarily) has been reached.

2.4 Key Motivation for Players

In the process of achieving the respective goals of the Pacman team and Ghost team as described earlier, players encounter several novel gaming experiences.

They are able to physically immerse in the role play of Pacman or Ghost. Unlike conventional games, where players are deskbound, Pacman and Ghost players get to move around in a wide outdoor gaming area. As the game progress, players come across familiar physical objects that are associated digitally to a virtual object. Interaction with such physical objects brings about a "digital reaction" in the fantasy virtual world. This brings out a unique quality of tangible interaction in the world of ubiquitous computing.

All players are able to interact with each other socially, either digitally online between Helpers and their partners, or in physical proximity through interpreting nuances of body language between Pacman and/or Ghost.

These novel features, as summarized in Table 1.1, promise to give players a unique and entertaining gaming experience.

Chapter 3

Human Pacman System Design

Human Pacman features a centralized client-server architecture that is made up of four main entities, namely a central server, and client wearable computers, helper laptops and Bluetooth embedded objects. An overview of the system is shown in Fig. 3.1. Wireless and wired LAN serves as a communication highway between the wearable computers, the helpers' computers (laptops),and the server desktop computer. Bluetooth embedded objects communicate with the wearable computer through Bluetooth communication. The server is kept updated on the status of the object through its network link with the wearable computer.

In this chapter, we will first look at the hardware aspect of the whole Human Pacman system, in particular that of the wearable computer and the Bluetooth embedded object. Next we will examine the software components of the system. Then we will look at the design of the backpack. Finally we will discuss about the issues encountered during the course of development.

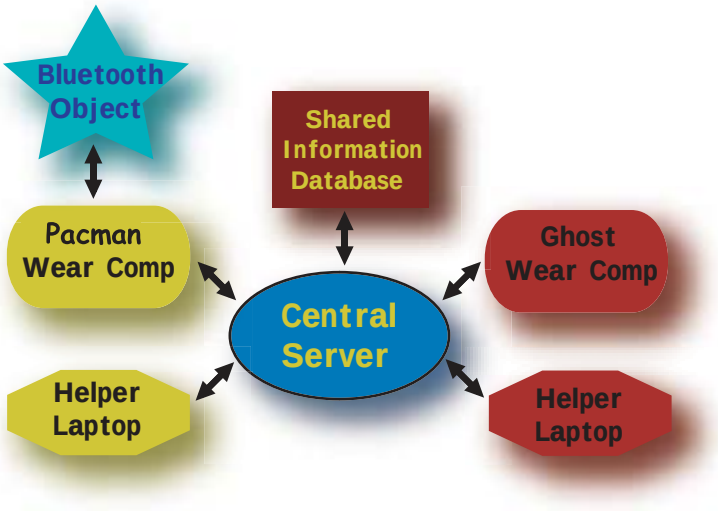


Figure 3.1: Architecture of the Human Pacman System

3.1 Hardware

In the Human Pacman system, the four main hardware components are the wearable computer, Bluetooth embedded object, server system, and the Helper system.

Some of the components used in the wearable computer and Bluetooth object were built in the laboratory by the author. The other parts were bought off the shelf and integrated into the system. This is listed in Table 3.1 and Table 3.2. The use of components-off-the-shelf (COTS) is preferred as extensive testing has already been done by the manufacturers.

3.1.1 The Wearable Computer

Fig. 3.2 shows the different components of the wearable computer that was built for Human Pacman. A description of each component is as follows.

Table 3.1: List of hardware components used in the wearable computer.

Component	**COTS**	**Lab-built**
Desknote A980	X	
Cyvisor DH-4400 VP	X	
Fire-i Digital Camera	X	
Twiddler2	X	
DLink DWL120+	X	
Bluetooth USB Dongle	X	
InertiaCube2	X	
DRM-III	X	
Touch Sensor		X
Power Supply and Controller unit		X
Keyspan USB Serial Adaptor	X	
Backpack		X

Table 3.2: List of hardware components used in the Bluetooth embedded object.

Component	**COTS**	**Lab-built**
Transmeta System	X	
TIQIT System	X	
Bluetooth module	X	
Touch Sensor		X
Power Supply and Controller unit		X
Casing		X

Figure 3.2: Components of the wearable computer built for Human Pacman

Desknote A980

At the heart of the whole system is a Desknote A980 [28] system, with an Intel Pentium 4 3.06GHz processor and an NVIDIA GeForce4 420 GO-32 [29] video card. The A980 is a hybrid between a laptop and a desktop. Externally it looks like a laptop without a monitor. However it is able to take high speed desktop Pentium 4 processors.

The following factors were taken into consideration when choosing A980 to be the core of the wearable computer.

- It runs on an external battery which means it need not be plugged to a wall socket.

- The high processing speed makes it advantages to use the A980 as compared to conventional single board computers (SBC).
- A good graphics processing unit is required to render 3D graphics smoothly. An NVIDIA Geforce4 420 Go can be fixed to the A980 for this purpose.
- A980 compares favorably with laptops of the same processing power given its low price.
- The A980 has a 6-pin IEEE1394 port as compared to a 4-pin IEEE1394 port in conventional laptops. This makes it simpler for Firewire camera connection since the camera can obtain power directly from the port without the need for an additional power supply.
- A980 does not have any RS-232 serial ports for the sensor devices. However this can be easily overcome by the use of a USB to serial adapter.
- The A980 is bulky and weighs almost 2.5kg with battery. It should be placed in a backpack to make carrying more comfortable for the user.

Cyvisor DH-4400VP

Cy-Visor DH-4400VP [30] is a video see-through Head Mounted Display (HMD). It has a resolution of 800 x 600 pixel SVGA display with a refresh rate of 60Hz. Operating at 9V, it consumes about 6 watts of power from either a battery or an external DC supply. As the battery is bulky and heavy, a regulated 9V DC supply

is connected to the external power jack of the HMD in the wearable computer so as to eliminate the need for an additional battery.

Fire-i Digital Camera

Unibrain's Fire-i Digital Camera [31] is a IIDC-1394 (V1.04) compliant Firewire camera. It supports a 640 x 480 VGA resolution with a frame rate of up to 30Hz. Given the high frame rate and good resolution, it is suited for use in the wearable computer system as a video input of the surroundings for display in the HMD. The camera is mounted in front of the HMD as seen in Fig. 3.3. Under normal operation, the camera requires 0.9 watts of power obtained from the Firewire port on the PC.

Figure 3.3: HMD with Unibrain camera and InertiaCube2 mounted.

Twiddler2

Twiddler2 [32] acts as a handheld keyboard and mouse inputting device for the system. Weighing only 165 grams and pocket-sized, it is a mouse pointer plus a full-function keyboard in a single unit that fits neatly in either right or left hand. As it uses the PS/2 interface, a USB to PS/2 adapter is required for connection to the Desknote.

DLink DWL120+

DLink DWL120+ [33] is a wireless LAN USB adapter that supports IEEE 802.11b standard. It allows a network transfer data rate of up to 11MBps and has a range of up to 300m outdoor. Such specification is sufficient for the proper operation of the Human Pacman system.

Bluetooth USB Dongle

Bluetooth communication is made with TDK's Bluetooth USB Adaptor [34]. The adapter is a Bluetooth 1.1 compliant device with a communication range of 10m. This makes it ideal for Human Pacman as its short range makes it essential for the user to be physically near the Bluetooth embedded device before she is being alerted to the latter. Added to that the device does not burden the wearable computer as it has a low power consumption of only 0.175 watts under normal operation.

InertiaCube2

InertiaCube2 [35] is a 3-DOF (Degree of Freedom) orientation tracking system. It is used to track the user's head orientation (yaw, pitch, roll) with a dynamic accuracy of 3° RMS. With a low latency of 8ms and high update rate of 180Hz, it is possible to have a real-time tracking of the user's head motion. This is important because the virtual cookies are kept in absolute real space positions in the user's AR view as she moves her head. A high latency and low update rate will affect the realism of the experience of having virtual cookies in real space.

InertiaCube2 operates at 6V and has a power consumption of 0.6 watts. Weighing only 28g, it is attached to the HMD as shown in Fig. 3.3. It uses a RS-232 serial interface and has to be connected to the Desknote via a USB to serial adapter.

DRM-III

The DRM-III [36] is an electronic module comprising of a 12-channel Global Positioning System (GPS) receiver, digital compass, pedometer, and altimeter. The DRM measures the displacement of the user from an initialization point by measuring the direction (with data obtained from the compass), and distance travelled (using accelerometer data) with each footstep taken. Although the DRM is a self-contained navigation unit, when GPS position data is available, it can be used to correct both the distance and direction calculations with the help of a Kalman filter algorithm. Besides, in conjunction with the step detection logic (pedometer), the module can detect running, sideways, and backwards walking which is necessary

for our application. The GPS has a 5 meters accuracy, whereas typical dead reckoning accuracy of 2% to 5% of distance travelled, entirely without GPS, is possible with DRM-III. The DRM-III is a self-contained module with a rechargeable Li-ion battery built in.

Touch Sensor

Touch-sensor is placed on the backpack of a player and is triggered when touched by an enemy. The circuit was designed and built using the capacitive touch sensor chip QT160 [37].

QT160 is a 6 channel burst mode digital charge-transfer sensor designed specifically for touch controls. It is a self-contained RISC core digital controller designed specifically for human interfaces. It first measures the background amount of capacitance on an object, and treats that like a "tare". It then looks at very small changes in the measured signal from that point on, caused by nearby objects, which in this case is the touch by a hand. QT160 employs basic capacitance measurement principle as follows. With reference to Fig. 3.4 the capacitance is ascertained by first closing switch S1 and charging a sense electrode Cx (which is an aluminium foil on the backpack of the wearable computer) to a fixed potential V_R, then, by reopening switch S1 and closing switch S2, transferring that charge to a charge detector comprising another known capacitor Cs [38]. The voltage V_S is determined through means of analog to digital conversion. Charge transference equation is given in Eq. 3.1. For Cx is much smaller than reference capacitance Cs the equa-

tion simplifies to Eq. 3.2, thus obtaining the value of Cx. Note that switch S3 is closed to reset the circuit for next cycle of measurement.

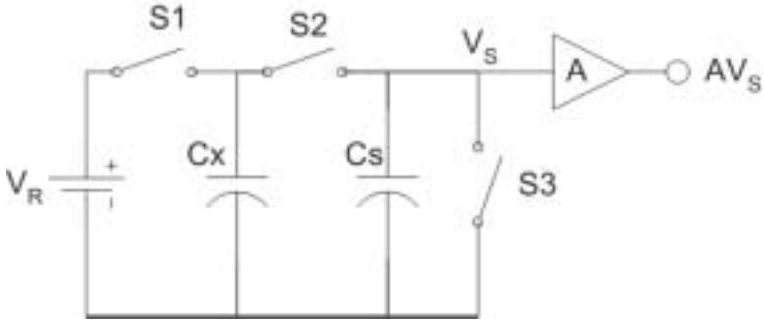


Figure 3.4: Circuit for measurement of capacitance of electrode Cx

$$V_S = V_R \frac{Cx}{Cx + Cs} \tag{3.1}$$

$$Cx = Cs \frac{V_S}{V_R} \tag{3.2}$$

Apart from basic capacitance measurement, QT160 has drift compensation algorithm to compensate for signal drift due to changes in external electrical characteristics over time. It also features Max On-Duration to prevent "stuck key" condition. If an object or material contacts a sense pad the signal may rise enough to trigger an output preventing further normal operation. Max On-Duration allows the chip to do a full recalibration after a predefined time.

Fig. 3.5 shows the circuit diagram of the touch-sensor circuit designed for the purpose of Human Pacman. Vdd is set to 5V; Vss to ground. An external oscillator circuit sets the frequency at 10MHz. The option pins OPT1 and OPT2 are connected to ground and Vdd respectively. This sets the Max On-Duration at 10s

(at oscillator frequency of 10MHz) in DC mode. In DC mode output will remain active for the duration of the detection. This is as opposed to the Toggle mode where the previous state of the output affects the current state of output during detection.

Only channel 1 on the QT160 is used. A value of 10nF was obtained by trial and error for the reference capacitance Cs of channel 1 and this is connected to SNS1B pin on QT160. A nominal value of 1nF is used as Cs for the other channels to allow the internal circuit of the QT160 to continue to function properly without interference from noise. SNS1A is connected to a sense electrode, made of aluminium foil, on the backpack. Digital output (OUT1), indicating if touch has been sensed, is connected to a microcontroller. The microcontroller will relay the triggered event to the PC via the serial connection. This will be described in detail in the next section. The touch sensor circuit is integrated with the controller circuit and is indicated in the PCB layout and actual circuit board in Fig. 3.7 and 3.8 respectively.

Power Supply and Controller Unit

The Cyvisor HMD, InertiaCube2, Touch Sensor, and controller unit are powered by a 12V rechargeable lead acid battery. The breakdown of the power specification of individual module is as follows.

- Cyvisor DH-4400VP – 9V, 0.667A
- InertiaCube2 – 6V, 0.100A

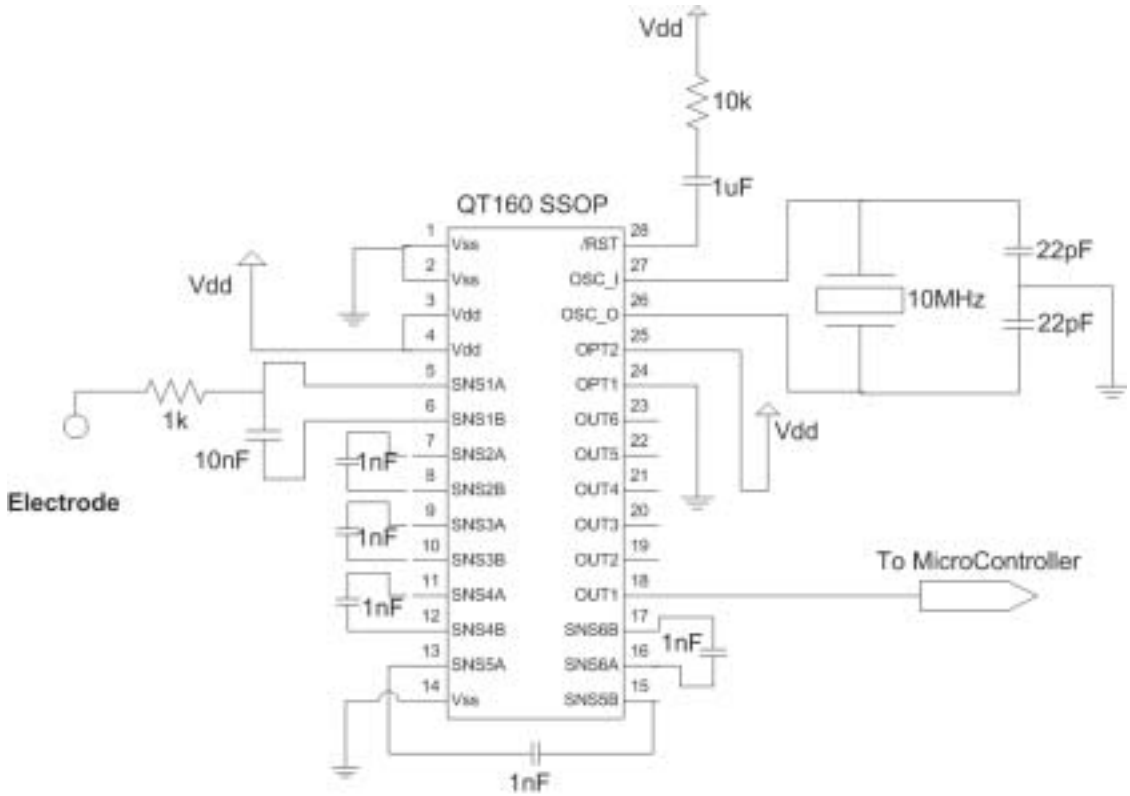


Figure 3.5: Touch Sensor Circuit

- Touch Sensor – 5V, 0.005A
- Controller Unit – 5V, 0.080A

A controller unit is implemented to keep track of the triggering of the touch sensor, the battery level, and the serial interface to the Desknote. It also gives a visual indication on the status of the touch sensor and battery level by means of LEDs. In addition it was found that when the InertiaCube2 is subjected to voltage less than 5.4V, the sensor gets damaged. Hence the controller unit switches off the sensor, by cutting off the power supply of the InertiaCube2, when voltage falls below a predefined level.

The circuit diagram of the power supply and controller unit is given in Fig. 3.6. As seen in the figure, the circuit, powered by linearly regulated voltages, comprises of a few main modules including the microcontroller (the PIC16F76 chip), voltage divider module, InertiaCube2 power control module, touch sensor module, LED display module, and RS232 interface module. Regulated voltages from the 6V and 9V linear regulators are pulled down to below 5V by the voltage divider module so as to allow the microcontroller to monitor the battery power. The InertiaCube2 power control module switches on/off the supply to the InertiaCube2 as dictated by the microcontroller. The circuit diagram of the touch sensor module is shown in Fig. 3.5. It informs the microcontroller of any "touch" event it senses. A detailed description of the touch sensor module was given in the previous section. The LED display module provides a visual affordance to information such as touch sensor and battery level status. Finally the RS232 interface module acts as an interfacing bridge to facilitate communication between the microcontroller and the serial port of the PC. These modules will be described in more details later in this section. Fig. 3.7 and 3.8 shows the PCB layout and actual circuit board of the controller board respectively. The main modules are indicated on these figures.

The linear voltage regulators LM7805, LM7806 and LM7809 [39] are used to provide +5V, +6V and +9V DC voltage respectively. These regulators has a maximum output current of 1A each, and provide voltages of $4.75V \leq V5 \leq 5.25$, $5.75V \leq V6 \leq 6.25$, and $8.65V \leq V9 \leq 9.35$ respectively. This is within the limits required for proper functioning of the individual module listed above. As

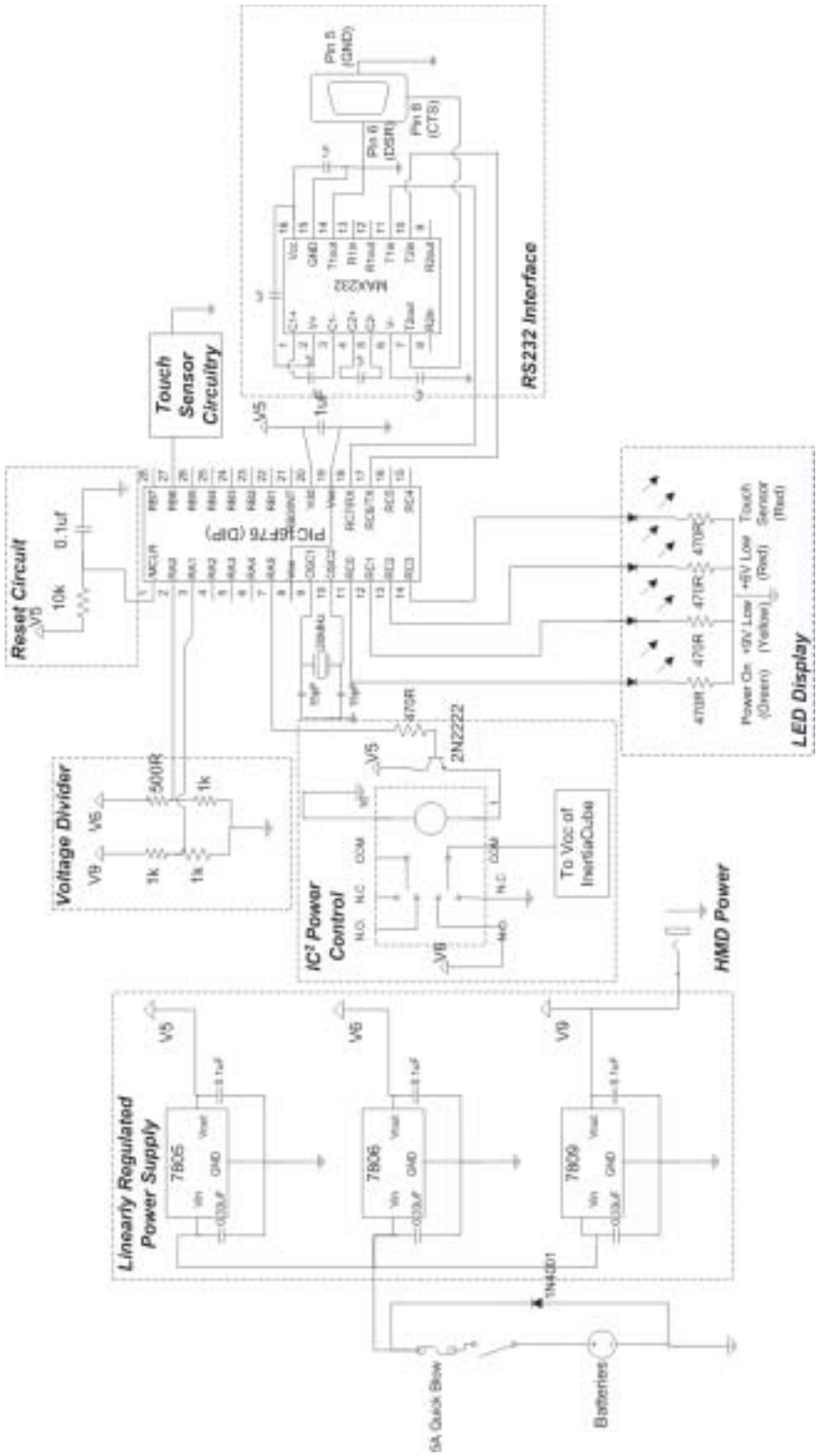


Figure 3.6: Circuit diagram of controller unit and power supply in the wearable computer

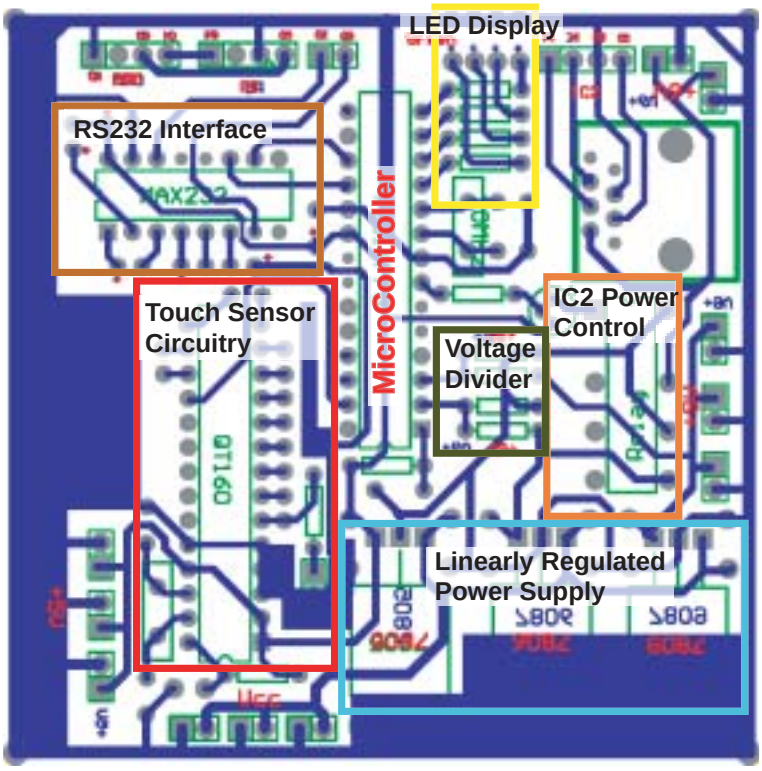


Figure 3.7: PCB layout of controller unit and power supply in the wearable computer with the main modules labelled

shown in the circuit diagram the HMD power input is obtained directly from the 9V regulated supply.

At the heart of the controller unit is a PIC16F76 [40]. The PIC16F76 is a low power 8-bit CMOS RISC microcontroller with 8k X 14 words of Flash program memory and 368 bytes of RAM. It has a 5 channel 8-bit Analog-to-Digital converter (ADC), supports 1 UART connection, and has 3 I/O ports. A 20MHz crystal oscillator circuit is connected to the PIC16F76. The frequency is chosen as it allows a convenient UART speed (9600bps) to be set accurately for communication to the PC. To reset the microcontroller whenever the power supply is turned on, a simple

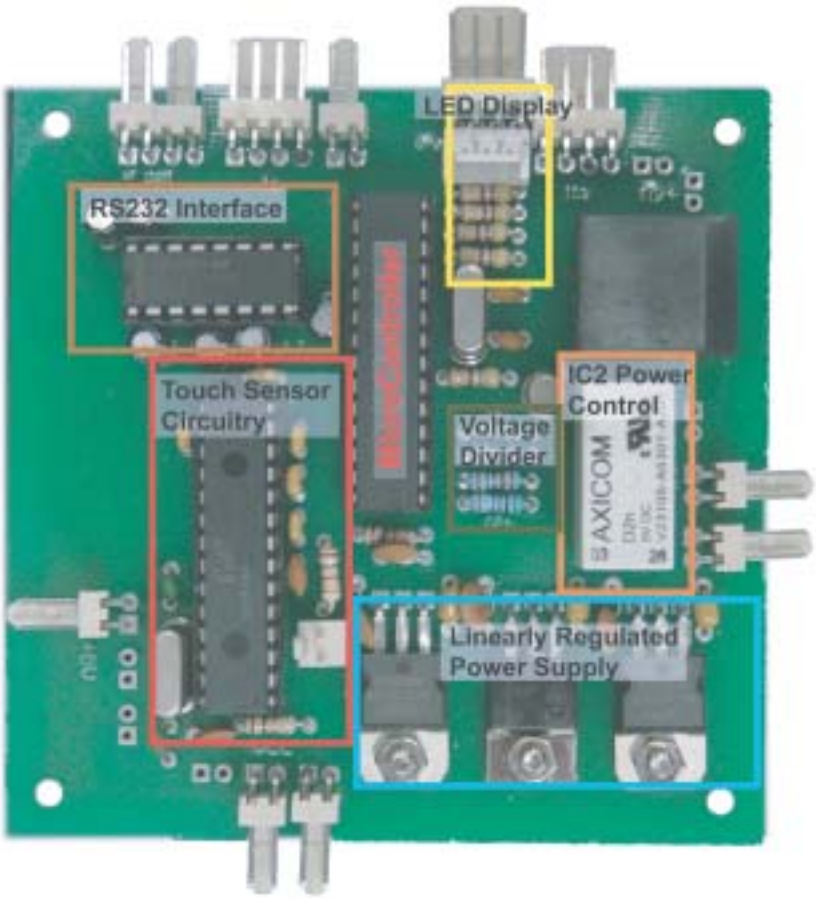


Figure 3.8: Actual circuit board of controller unit and power supply in the wearable computer with the main modules labelled

RC-circuit is connected to the MCLR pin. This is indicated in Fig. 3.6 as the reset circuit module.

There are 5 main functions performed by the PIC16F76 and it includes the following.

- Determination of battery level
- Cutting off of power supply to InertiaCube2 so as not to allow its power

supply to fall below 5.4V.

- Visual indication of touch sensor and battery level status using LEDs.
- Detection of the triggering of touch sensor.
- Conveying of touch sensor and battery level status to the PC.

Battery Level Determination

Typically when the input supply to the linear regulator falls below 2V above its rated output voltage, the output voltage starts to decline. Based on this characteristic, the power capacity left in the battery is divided into three level, namely high, mid and low power. High power indicates that the battery supply is sufficient to keep the 9V regulated supply to above 8.60V. Mid power indicates that the 9V supply has fallen below 8.60V but the 6V regulated supply is still above 5.70V. Low power indicates that the 6V regulated supply has fallen below 5.70V.

The outputs from the 6V and 9V regulators are connected to the RA0 and RA1 pins of port A respectively through a potential-divider circuit as shown in the diagram. Using potential-divider rule, the voltage input at RA0 and RA1 are 0.67 x V6 and 0.5 x V9 respectively, which under high power operation is below the 5V reference used for the ADC operation. Note that the upper limit of voltage at RA0/RA1 is chosen near the reference voltage so as to allow for a higher fraction of the original voltage (V6/V9). As the resolution of the ADC is fixed for a given reference voltage, a large fraction is favored as any change in V6/V9 will translate

to a large change at RA0/RA1 as compared to when a small fraction is used. This improving the sensitivity towards changes in V6 and V9.

When the battery capacity reaches the low power level, power supply to the InertiaCube2 is cut off. Removal of the load causes the voltage of the battery to increase, leading to an increase in V6 and V9. A hysteresis loop has to be in place to prevent the oscillatory switching between the mid and low power levels under such situation. When power capacity left is in the low power level, V9 has to be raised above 7V (found by trial and error) to return to the mid level. Similarly V9 has to be raised above 8.75V in order for the controller unit to register that power capacity has changed from the mid level to the high level. Fig. 3.9 shows the timing diagram at the microcontroller as the battery level changes. Three LEDs attached to the microcontroller indicate the state of the battery supply. As shown in the diagram, a signal is sent to the PC's serial port (CTS pin) when the battery level is out of the high power level. This change in power level will be indicated in the HMD display so that the user can take the necessary action (eg. charge the battery before the HMD cuts off).

In order to determine the binary ADC reading for each cutoff voltage, a preliminary program (see Sec. B.2) is written to calibrate the readings at the voltages. These values are then hard-coded in the final firmware of the controller unit. As a small note, 1% accuracy resistors are used in the potential-divider circuit so that cutoff voltages for replicas of the controller circuit for each wearable computer does not vary too much.

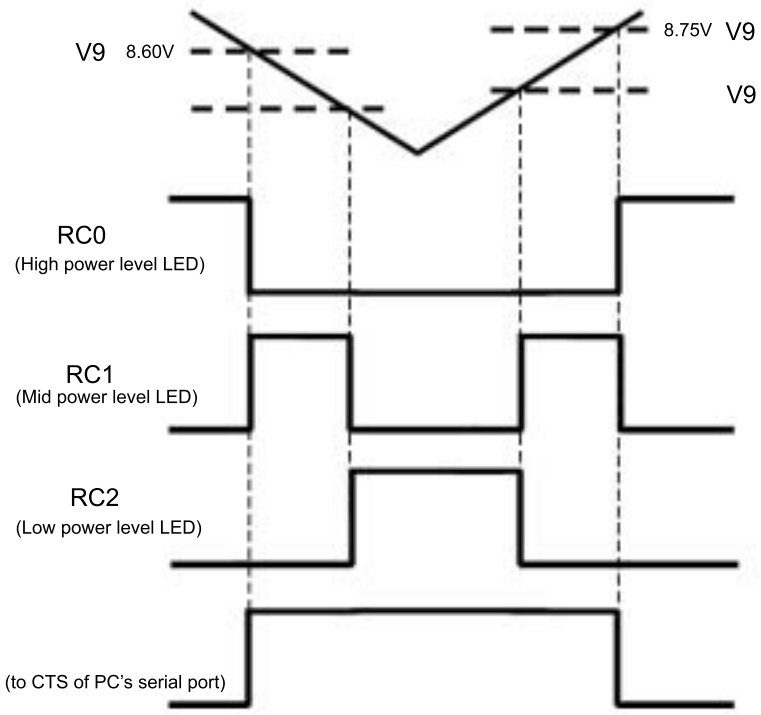


Figure 3.9: Timing diagram at the microcontroller as the battery level changes

InertiaCube2 Power Control

A 5V relay is used to control the power supply to the InertiaCube2. The power supply is connected to the common (COM) pin of the relay. The normally close (NC) pin of the relay is connected to the circuit ground and the normally open (NO) pin to V6. RA5 pin controls the relay via a 2N2222 NPN transistor. RA5 is high under normal operation. When the battery goes low, RA5 goes low and this switches off the relay, effectively cutting off the power supply to the InertiaCube2.

Triggering of Touch Sensor

The touch sensor was described in the previous section. Fig. 3.5 gives the circuit diagram of the touch sensor. The output from this module is connected to RB6 of Port B on the microcontroller as indicated in Fig. 3.6. Fig. 3.10 shows the timing diagram at the microcontroller when the touch sensor is activated. When the low-to-high signal from the touch sensor is seen at RB6 of the microcontroller, RC3 is set to high to light up the red LED. A signal is sent to the DSR pin of the PC's serial port from the RC6 pin of the microcontroller. When the signal from the touch sensor goes to low again, RC3 and RC6 are set to low.

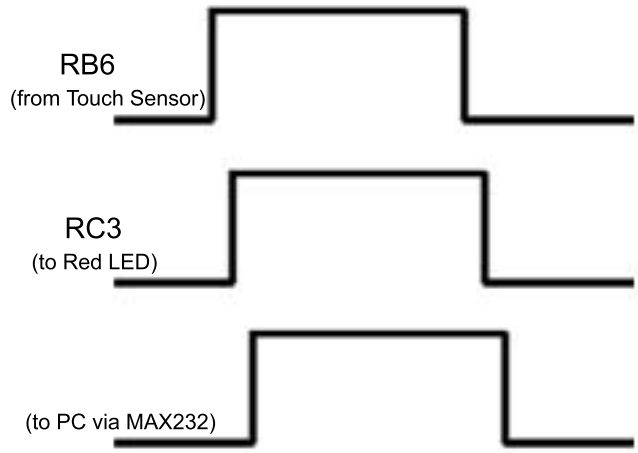


Figure 3.10: Timing diagram at the microcontroller when the touch sensor changes between the activated and inactivated states

LED Display

Green, yellow, and red LEDs are used to indicate the high, mid, and low power

level states. At any one time only one of the three will be on. A fourth LED is turned on when touch is sensed by the sensor. The LEDs are connected to RC0 - RC3 of port C as indicated in Fig. 3.6.

RS232 Interface

RS232 interface is still very much used in embedded computer system as it is a simple, robust and low-powered solution. In the case of the controller unit, only two bits of data are used for communication with the wearable computer. During preliminary calibration of the controller unit (for the battery-level determination function), the data communicated to the PC requires only low data rates. A RS232 serial interface suffice for these purposes and is hence used.

In order to connect the PIC16F76 to PC, RS232 physical layer standard must be met at the interfacing connection on the side of the controller. This is done using MAX232 [41] chip. RC6 is set to high when the touch sensor is triggered and it returns to low when it is not. RC7 is set to low when the battery is in high power state and high otherwise. These outputs are connected, through MAX232, to pin 8 (CTS) and pin 6 (DSR) of the PC serial port respectively.

RC6 also serves as the TX pin for UART communication to the PC during the calibration of ADC readings mentioned earlier. Note that the corresponding connection to the PC in this case is pin 2 (RX). In this preliminary program, UART is configured as a 9600bps 8-bit communication, with 1 stop bit and no parity bit.

Fig. 3.11 summarises the connection between the PC serial port and the microcontroller (via the MAX232 chip) during calibration and normal operation.

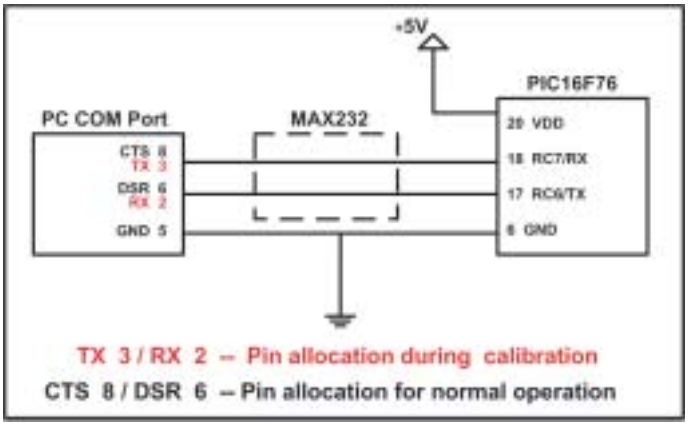


Figure 3.11: Connection between PC serial port and PIC16F76 of the controller during calibration and normal operation

Fig. 3.12 shows the software flowchart of the firmware in the controller unit. The source code is given in Appendix A (see Sec. B.1). A polling mechanism has been used to cycle through the different functions performed by the controller. The flow of the program is as follows.

1. The program starts by initializing the I/O ports on the microcontroller and setting the parameters.

2. Analog-to-digital conversion are done for voltages at pins RA0 and RA1 and the results are stored.

3. With these readings the power level is determined and the LEDs are turned on or off accordingly.

4. If the power level is high, RC7 is set low so as to send a '0' to pin 6 (DSR) of the PC serial port; otherwise it is set high.

5. RB6 is then checked to see if the touch sensor has been activated. If RB6 is high, indicating that the sensor has been activated, RC6 is set high so as to send a '1' to pin 8 (CTS) of the PC serial port; otherwise it is set low.

6. The program then loops back to (2).

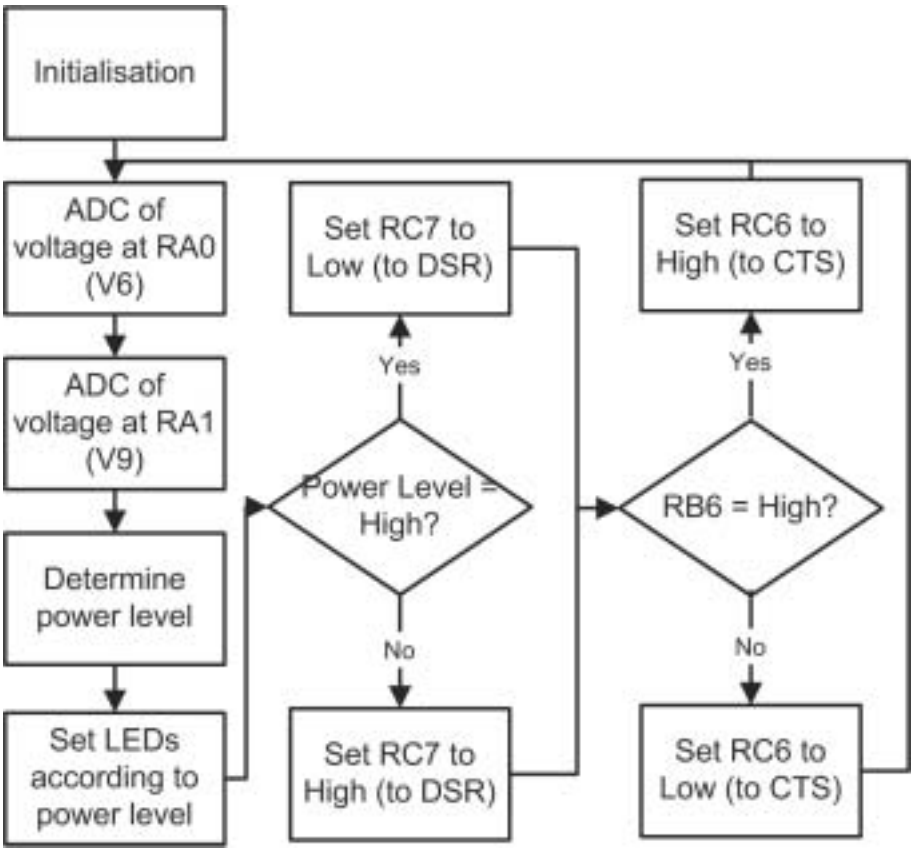


Figure 3.12: Firmware flowchart of the controller unit in the wearable computer

Keyspan USB Serial Adapter

Keyspan USB Serial Adapter [42] is used for adding RS232 connectivity to the Desknote. It acts an adapter for connection of InertiaCube2, DRM-III, and touch sensor and low power data to the Desknote. The adapter was chosen as the InertiaCube2 was incompatible with the other brands. In addition it comes with a Keyspan Serial Assistant software which allows mapping of a COM port number to the hardware. This makes it convenient to assign a fix COM port number to each device. Furthermore as the InertiaCube2 could only use COM 1 to 4, using the software solves the problem when the operating system assigns a COM port number higher than 4 to the InertiaCube2.

3.1.2 Bluetooth Embedded Object

Two types of Bluetooth embedded objects were built, namely the Transmeta and the Tiqit systems shown on the left and right in Fig. 3.13 respectively. Both types of system has a similar general makeup, mainly a single board computer (SBC), a serial Bluetooth dongle, touch sensor circuitry, controller module, and power supply. The components in each type of Bluetooth embedded object are indicated in the figure. Note that the touch sensor circuitry and controller module has been integrated into the controller unit in the figure. Both SBC runs Linux kernel 2.4.22 and uses the BlueZ [43] Bluetooth stack.

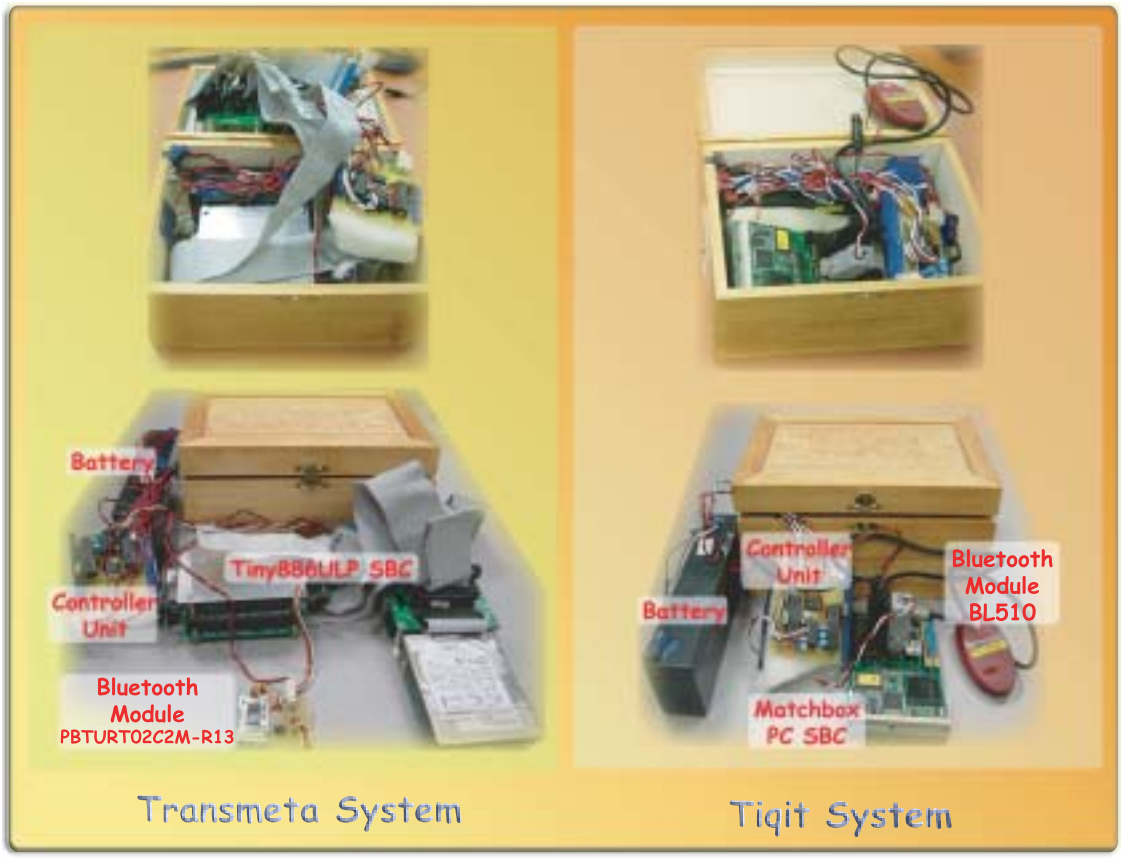


Figure 3.13: Two types of Bluetooth embedded objects

Transmeta System

Tiny886ULP [44] is the SBC used in the Transmeta system. Tiny886ULP is an ultra low power PC-AT compatible embedded computer in a PC/104 form factor. It features a Pentium-class Crusoe TM5600 [45] processor running at up to 733MHz. The SBC uses standard 2.5 inch hard disk and provides two RS232 serial ports.

The Bluetooth module used in tandem with Tiny886ULP is PBTURT02C2M-R13. It is a Class 2 Bluetooth V1.1 compliant module with a UART interface running at 57.6kbps. The module is chosen for its small form factor so as to

maintain an overall small size for the Bluetooth embedded object. As it uses a normal UART interface, a surface mount MAX233 [41] chip is used as an interface between the Bluetooth module and the serial port of the SBC. Fig. 3.14 illustrates the connections between PBTURT02C2M-R13 and the serial port on the SBC. Fig. 3.15 and Fig. 3.16 gives the PCB layout of the interface and the actual boards respectively.

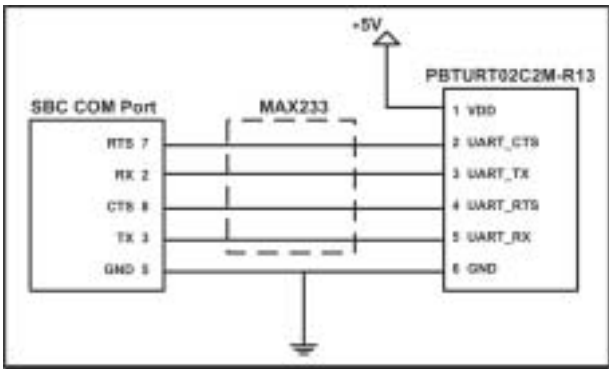


Figure 3.14: Connection between PBTURT02C2M-R13 and the serial port from the SBC

Tiqit System

The SBC used in Tiqit System is a Matchbox PC [46] (MPC). The MPC has a 66MHz 486-SX based CPU with 16MB SDRAM. It has two RS232 serial ports and has an IDE controller controlling a fixed 16MB Flash and a 1GB Microdrive. It requires 3 to 7.5 watts of power and is smaller than the Tiny886ULP.

The Bluetooth module used in tandem with the MPC is the BL510 [46] from Brain Boxes. It is a RS232 Class 1 Bluetooth module compliant with Bluetooth

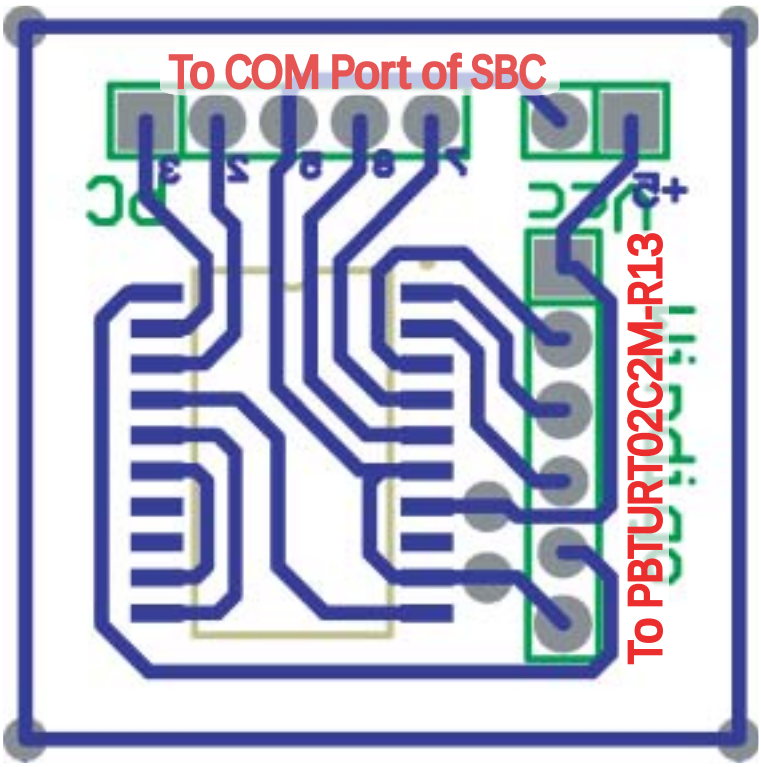


Figure 3.15: PCB layout of RS232 interface circuit for PBTURT02C2M-R13

V1.1. The module requires a 5V DC supply and is comparatively larger than the PBTURT02C2M-R13.

Touch Sensor

A similar design for the touch sensor is implemented in the Bluetooth embedded object as in the wearable computer given in Sec. 3.1.1 under the "Touch Sensor" section. Two aluminium discs at the side of the wooden box serves as the electrode for the touch sensor. The aluminium disc electrode is shown in Fig. 3.17.

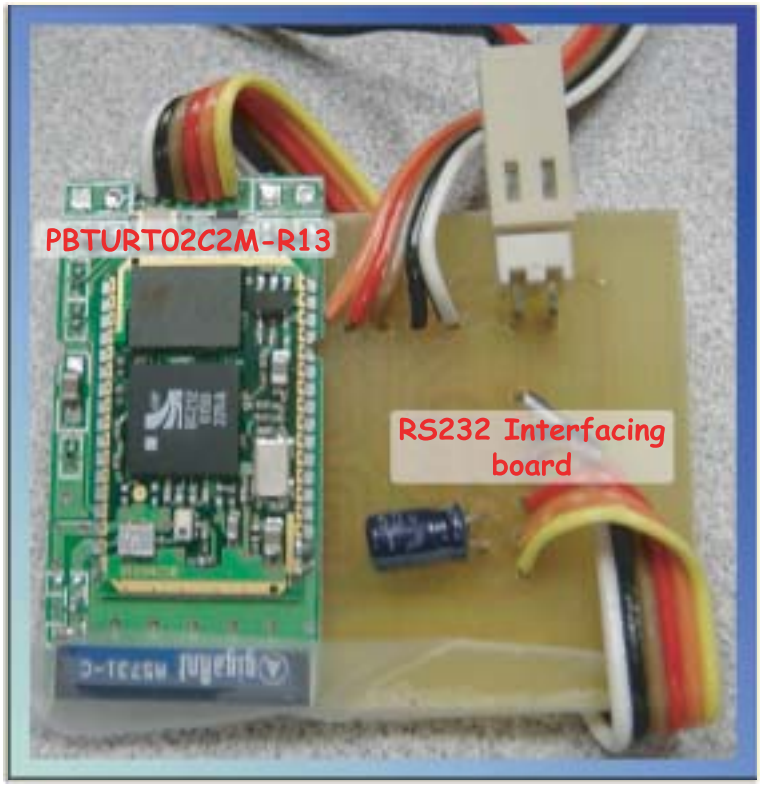


Figure 3.16: The PBTURT02C2M-R13 with the RS232 interfacing board

Power Supply and Controller Unit

The Bluetooth embedded object runs on a 12V rechargeable lead acid battery. Non-regulated power is supplied to the SBC. A 5V regulated supply from LM7805 is connected to the Bluetooth module, touch sensor, and controller circuit.

The main purposes of the controller circuit include detection of touch sensor events, and the control of external interfaces for a display-less and keyboard-less operation. Similar to the controller unit in the wearable computer, the PIC16F76 acts as the brain to the controller unit in the Bluetooth embedded object.

Fig. 3.18 shows the circuit diagram. The basic configuration of the circuit is

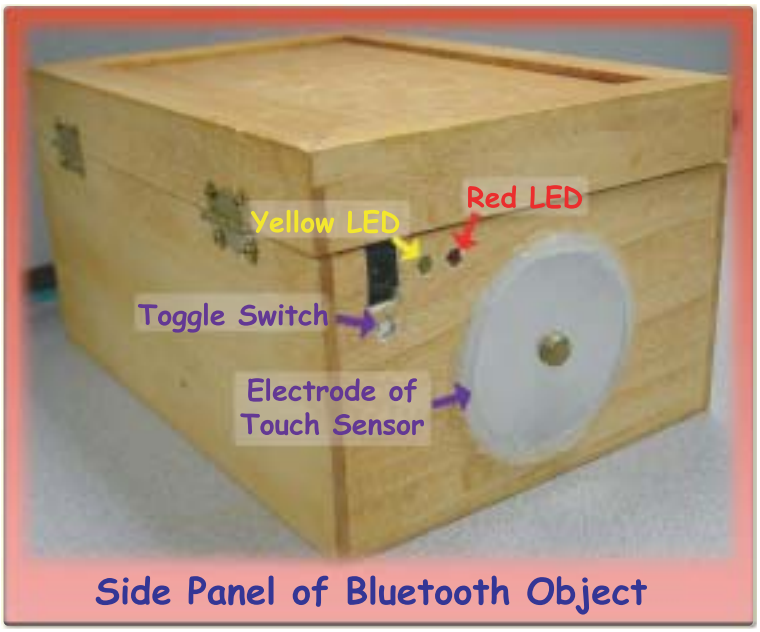


Figure 3.17: Side panel of Bluetooth embedded object

similar to that used in the wearable computer. As seen in the figure, the circuit, powered by linearly regulated voltages, comprises of a few main modules including the microcontroller (the PIC16F76 chip), touch sensor module, LED display module, toggle switch module and RS232 interface module. The circuit diagram of the touch sensor module is shown in Fig. 3.5. It informs the microcontroller of any "touch" event it senses on its electrode. A detailed description of the touch sensor module was given previously in Sec. 3.1.1 under the "Touch Sensor" section. The LED display module provides a visual affordance to information such as touch sensor and SBC boot-up status. The toggle switch module allows the user to send commands to the SBC, such as to shutdown the operating system, by toggling it. Finally the RS232 interface module acts as an interfacing bridge to facilitate communication between the microcontroller and the serial port of the

SBC. The use of these modules will be highlighted in more details later in this section. Fig. 3.19 and 3.20 shows the PCB layout and actual circuit board of the controller board respectively. The main modules are indicated on these figures.

As the Bluetooth embedded object has to be a solitary device with no external display or keyboard inputs, visual indications and external inputting methods has to be implemented for the user to easily operate the device. For this purpose two LEDs (red and yellow) and a toggle switch are added on to the device. They are on the side panel as shown in Fig. 3.17. These additional components are controlled by the PIC16F76. Information is relayed between the PIC16F76 and the SBC through UART communication. The UART is configured as a 9600bps 8-bit communication, with 1 stop bit and no parity bit. Fig. 3.21 shows the pin-to-pin connection between the serial port of the SBC and the PIC16F76.

Device startup

During the powering up of the device, the operating system (Linux) can be set to auto-start the client program after it finishes its booting up process. However the process takes a long time, and since no external display device is connected to the SBC, it is difficult to estimate when the device is ready. To overcome this, red and yellow LEDs are turn on and off respectively by the controller to indicate the booting process. On starting, the client program sends a PGM_START character serially to the controller to indicate that the system is ready for normal mode operation. The controller toggles the LEDs, leaving the red and yellow LEDs off and

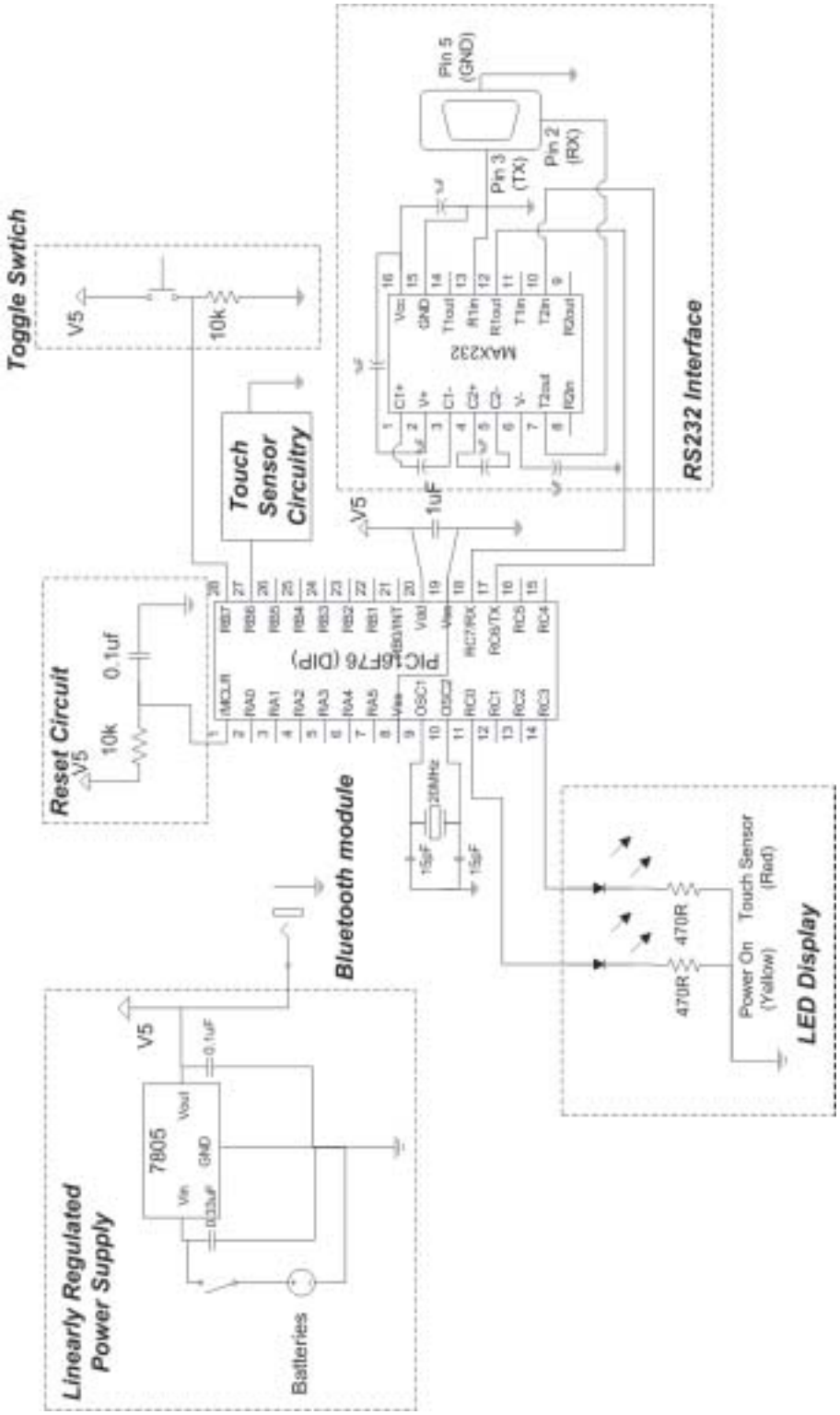


Figure 3.18: Circuit diagram of controller unit in the Bluetooth embedded device

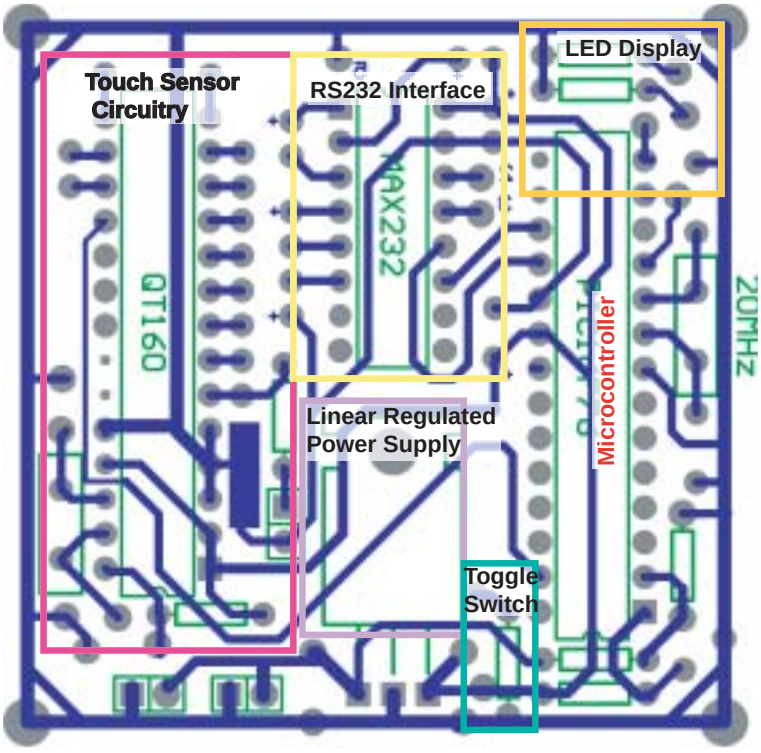


Figure 3.19: PCB layout of controller unit in the Bluetooth embedded device with main modules labelled

on respectively.

Triggering of Touch Sensor

In normal mode operation, the red LED is turned on when the touch sensor is activated. A TS_ON character is sent to the SBC. When touch is removed from the sensor, the red LED is turned off and a TS_OFF character is sent.

Shutting Down

Proper shutdown of Linux in the SBC is necessary to prevent corruption of data

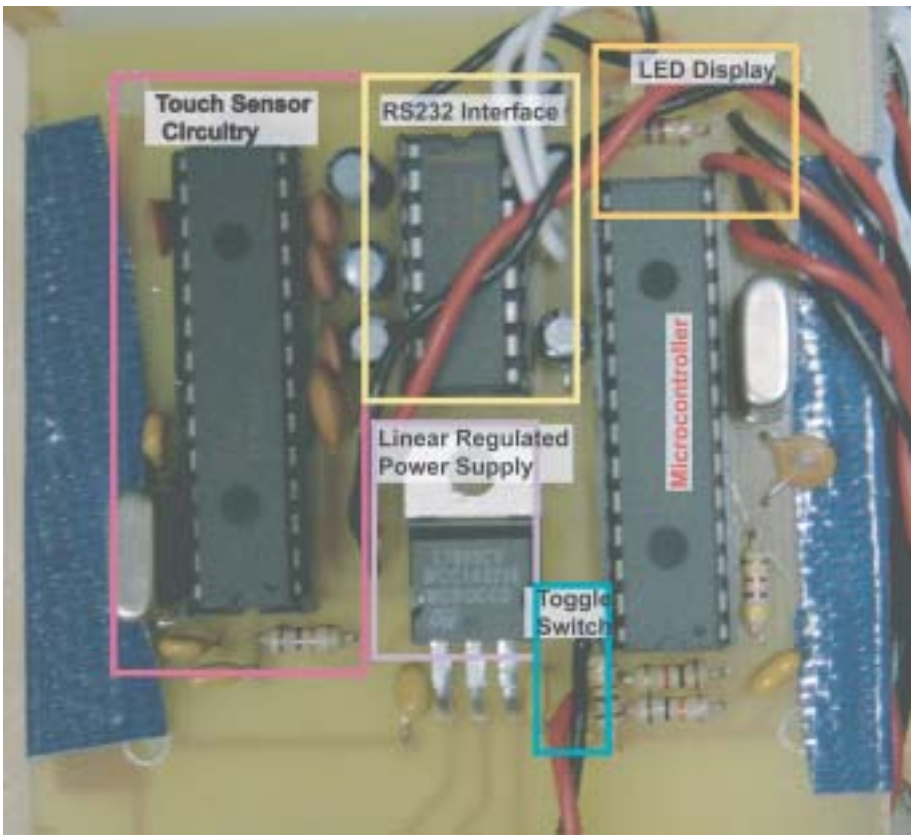


Figure 3.20: Actual circuit board of controller unit in the Bluetooth embedded device with main modules labelled

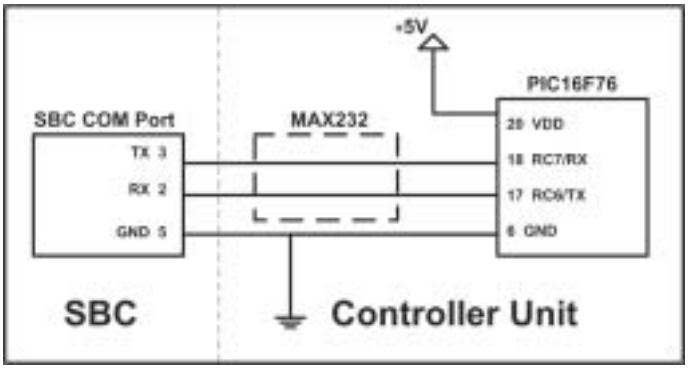


Figure 3.21: Connection between SBC serial port and PIC16F76 of the controller in the Bluetooth embedded object

in the system. To shutdown the system, the toggle switch has to be pressed and held on for 5 seconds. The controller will then issue a SHUT_CHAR character to the SBC as an indication to shutdown the system. The LEDs will then blink alternately at a rate of 2Hz.

Fig. 3.22 shows the software flowchart of the firmware in the controller unit. The source code is given in Appendix A (see Sec. B.3). An interrupt mechanism has been used to perform the functions. The flow of the program is as follows.

1. Initialization of microcontroller, including configuring the UART communication, I/O ports, turning on the Port B and UART interrupt, and turning off the timer 1 interrupt.

2. The system is set to stall mode. To indicate stall mode, the red LED is turned on and yellow LED turned off.

3. Wait for interrupt. Jump to (4) if there is an interrupt

4. If it is not timer 1 interrupt, jump to (5)

   (a) Check if the toggle switch has been depressed for 5 seconds.

      i. If yes, then send a SHUT_CHAR character through the UART to the SBC. Then blink the LEDs infinitely.

      ii. If not, then return from interrupt and jump to (3).

5. If Port B has not been changed, jump to (6)

(a) Check if system is in stall mode.

i. If yes, then return from interrupt and jump to (3).

ii. If not, then check if RB6, the input to touch sensor, has changed.

A. If yes, then check if touch sensor is activated.

- If yes, then turn on both red and yellow LEDs and send a TS_ON character through the UART to the SBC. Return from interrupt and jump to (3).
- It not, then turn red LED off and yellow LED on and send a TS_OFF character through the UART to the SBC. Return from interrupt and jump to (3).

B. If not, then check if RB7, the input to toggle switch, has changed.

- If yes, then check if the toggle switch has been pressed. If the switch is closed then start timer 1 interrupt; else stop timer 1 interrupt. Return from interrupt and jump to (3).
- If not, then return from interrupt and jump to (3).

6. Data has been received through the UART of the controller from the SBC. Check if PGM_START character has been received.

(a) If yes, then set the system to normal mode Turn the red LED off, and yellow LED on. Return from interrupt and jump to (3).

(b) If not, then return from interrupt and jump to (3).

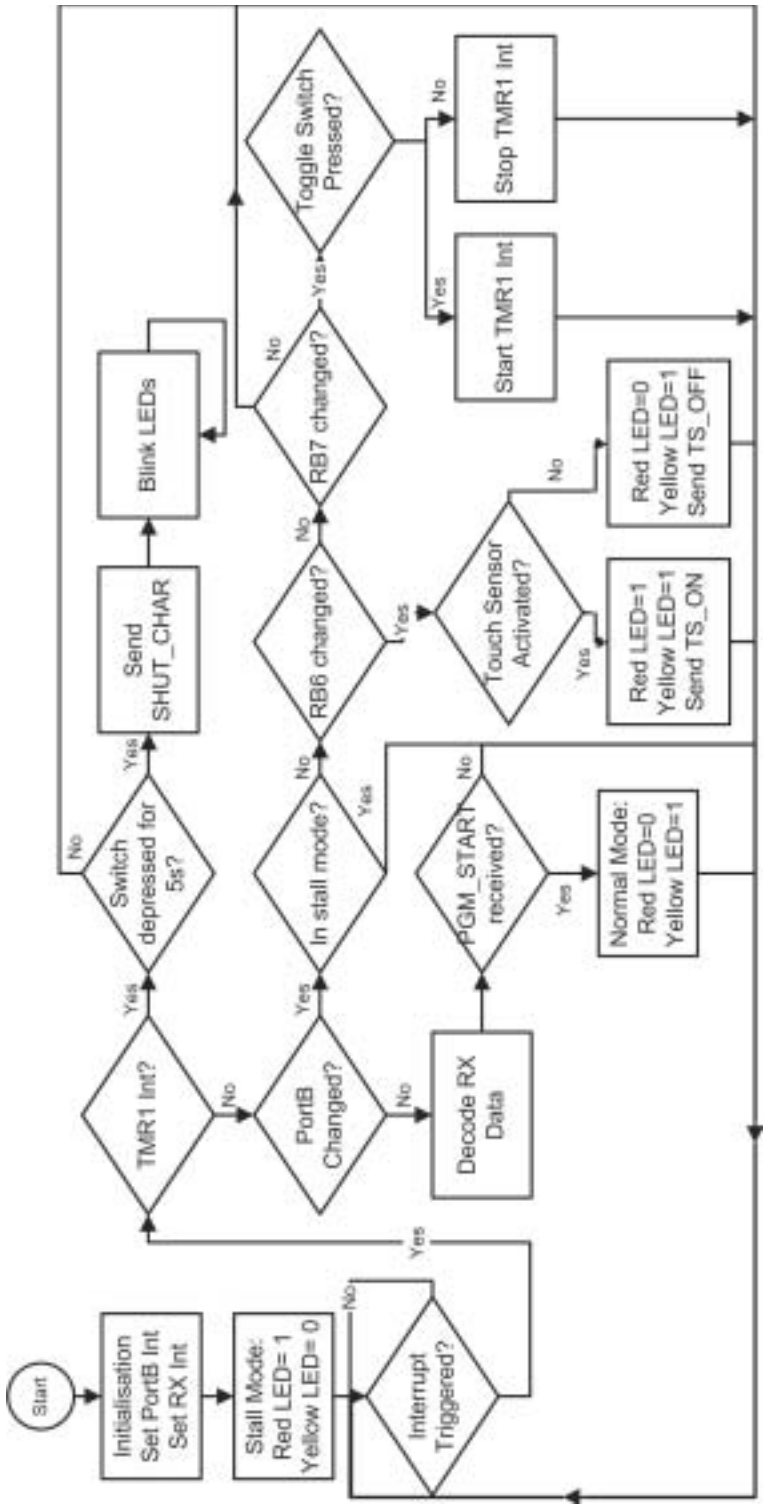


Figure 3.22: Firmware flowchart of the controller unit in the Bluetooth embedded object

3.1.3 Server and Helper Systems

The system requirements of both the server and Helper systems are less stringent as compared to the wearable computer. Both systems should have a 100Mbps LAN adapter and be able to run Microsoft Windows XP [47]. In addition, the Helper systems should have a graphics card which supports OpenGL 1.1 [48].

3.1.4 Miscellaneous Hardware

802.11b wireless LAN router is used to connect all the server and the clients in the network.

3.2 Software

Except for the software in the SBC of the Bluetooth embedded object, which is coded in GNU C, the other programs are coded in Microsoft Visual C++ 6.0. OpenGL 1.1 is used for rendering of graphics in the Pacman\Ghost and Helper systems. MXRToolkit [49], an open source development kit developed in our lab, is used in the Pacman\Ghost systems for AR calculations.

3.2.1 Overall System Data Flow

Fig. 3.23 gives a summary of the overall system data flow between the wearable computer, server, Helper system and the Bluetooth embedded object.

At the heart of the Human Pacman system is the server which acts to relay and

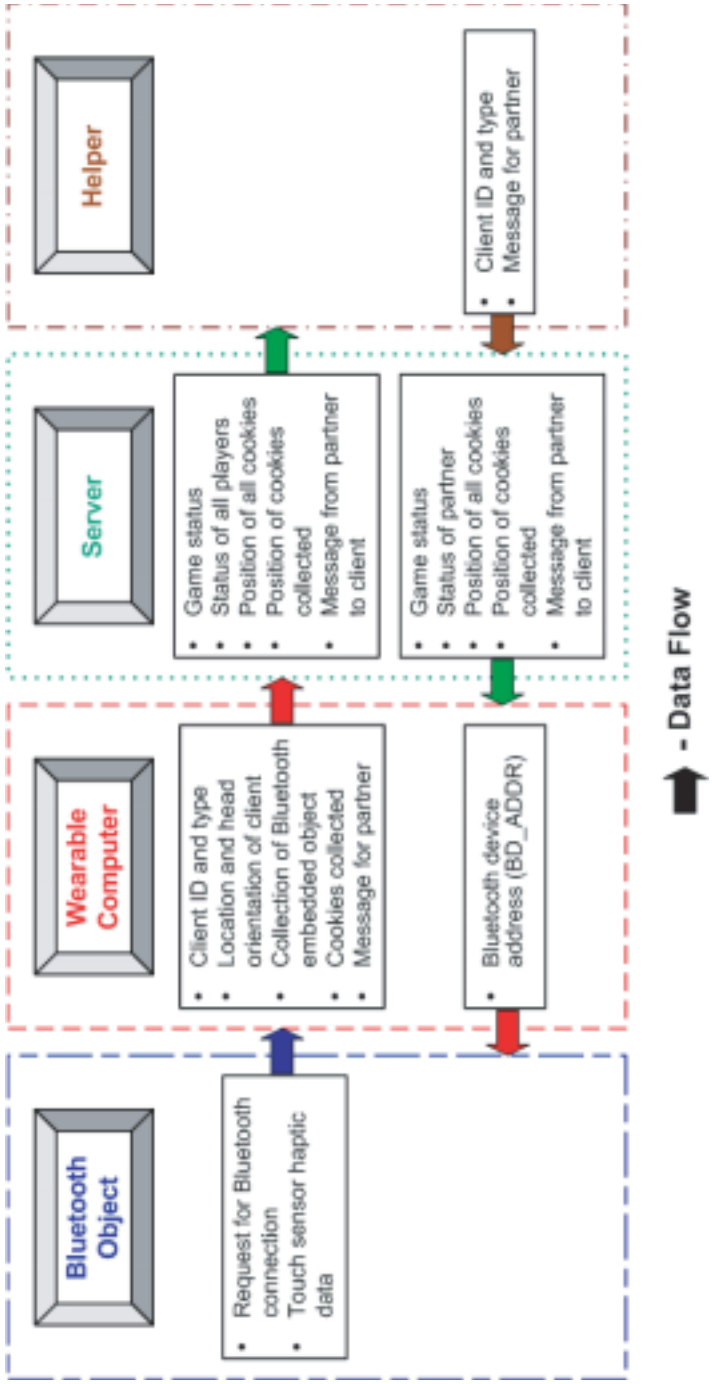


Figure 3.23: Overall system data flow between the wearable computer, server, Helper system and the Bluetooth embedded object

broadcast information between the systems of Pacmen, Ghosts and Helpers. The server sends the following data to Pacman\Ghost wearable computer system.

- The game status such as whether Pacman or Ghost has won.
- The client ID of user's partner if she has been assigned one.
- Initial position of all virtual cookies in the game area.
- Positions of all the virtual cookies that has already been collected.
- Any message that has been sent by the user's partner.

The Helper system receives the same kind of data from the server as the wearable computer. In addition she also receives status of all Pacman and Ghost in the game. This includes the position and orientation of these players, the event of a Pacman turning into a Super-Pacman, and the devouring of any Pacman\Ghost. She also receives the position of the Bluetooth embedded object in the game area. In turn the Helper system sends to the server her client ID and type (i.e. Helper), and any message for her partner player.

Wearable computers system has data exchange with the server and Bluetooth embedded object. The wearable computer system sends the following data to the server.

- The client ID and type (i.e. Pacman or Ghost) of the player.
- The player's location and her head orientation data.

- If it is a Pacman system, whether the player has become a Super-Pacman.
- If it is a Pacman system, the virtual cookies collected by the Pacman.
- Any message for player's partner Helper.

The Bluetooth embedded object sends the request for Bluetooth connection to the Pacman wearable computer system. In turn it receives the unique Bluetooth device address (BD_ADDR) for it to ascertain that the Bluetooth server is a Pacman wearable computer system. The Bluetooth embedded object also sends any touch sensor haptic data, indicating that the touch sensor has been activated, to the Pacman system, if one is connected.

3.2.2 Server System

The server maintains up-to-the-minute players' information (location, status etc.) and sends this information to the relevant clients. Physical location and players' status updates are sent from the client wearable computers to the server every 10 ms to 21 ms, depending on the processing load on the client. The software flow of the server system is given in Fig. 3.24. The program flow is as follows.

1. The program parameters are first initialized.
2. A TCP socket at port 9090 is created in a listening mode.
3. Server waits for client request. Program jumps to (4) if there is a request, else it loops here.

4. The server gets the client ID.

5. If client is new, then create a new member in the server's client map.

6. Data is received from the client and this is stored in a database.

7. The client is updated with cookies and other information.

8. Program jumps to (3).

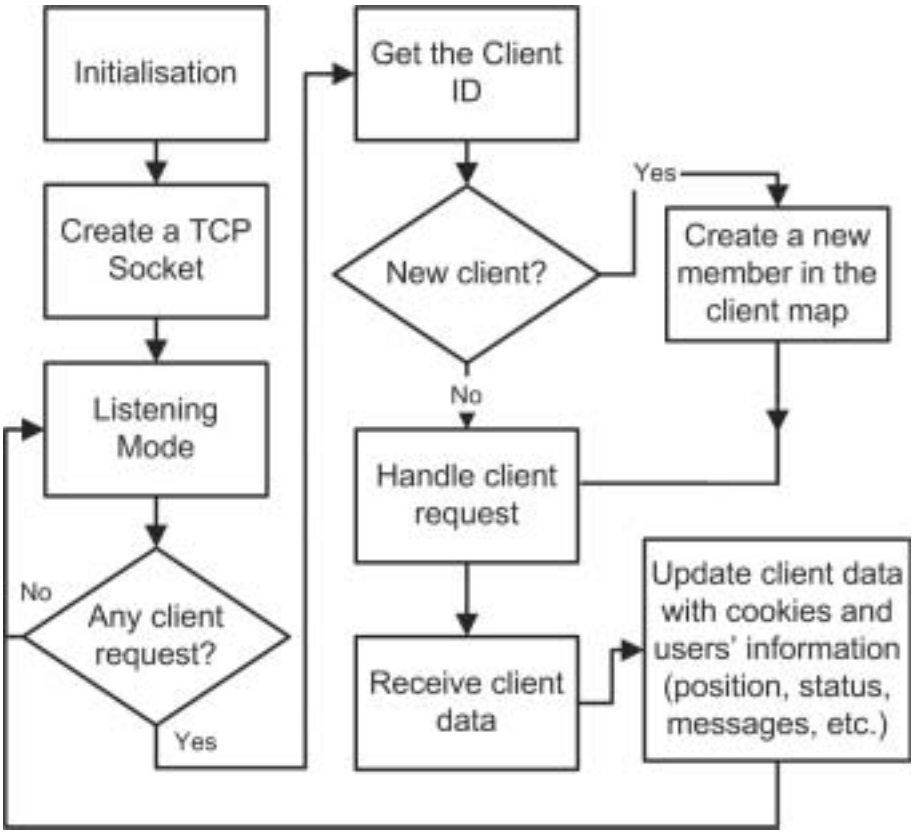


Figure 3.24: Flowchart of server

3.2.3 Helper System

Multiple threads run in the Helper system. The main thread initializes the system, renders the VR world using VRML, and updates the position and orientation of the VRML models of the Pacman, Ghost and cookies. A network thread exchanges data with the server. Data obtained from the server includes positions of the mobile players and cookies, and data sent includes messages sent to the Helper's partner. The software flow of the Helper system is given in Fig. 3.25. The program starts by initializing parameters and starting a new network thread. Data flow between the threads is done through setting a shared semaphore.

The flow of the network thread is as follows.

1. The position of all the cookies in the game are obtained from the server and stored as shared data.

2. The status of all the mobile players, such as if there is a Super-Pacman, is obtained from the server.

3. A list of cookies eaten is obtained from server.

4. The server is updated if Helper has any message to send to her partner.

5. The server is checked to see if the Helper's partner has any message for her.

6. The program jumps to (2).

The flow of the main thread is as follows.

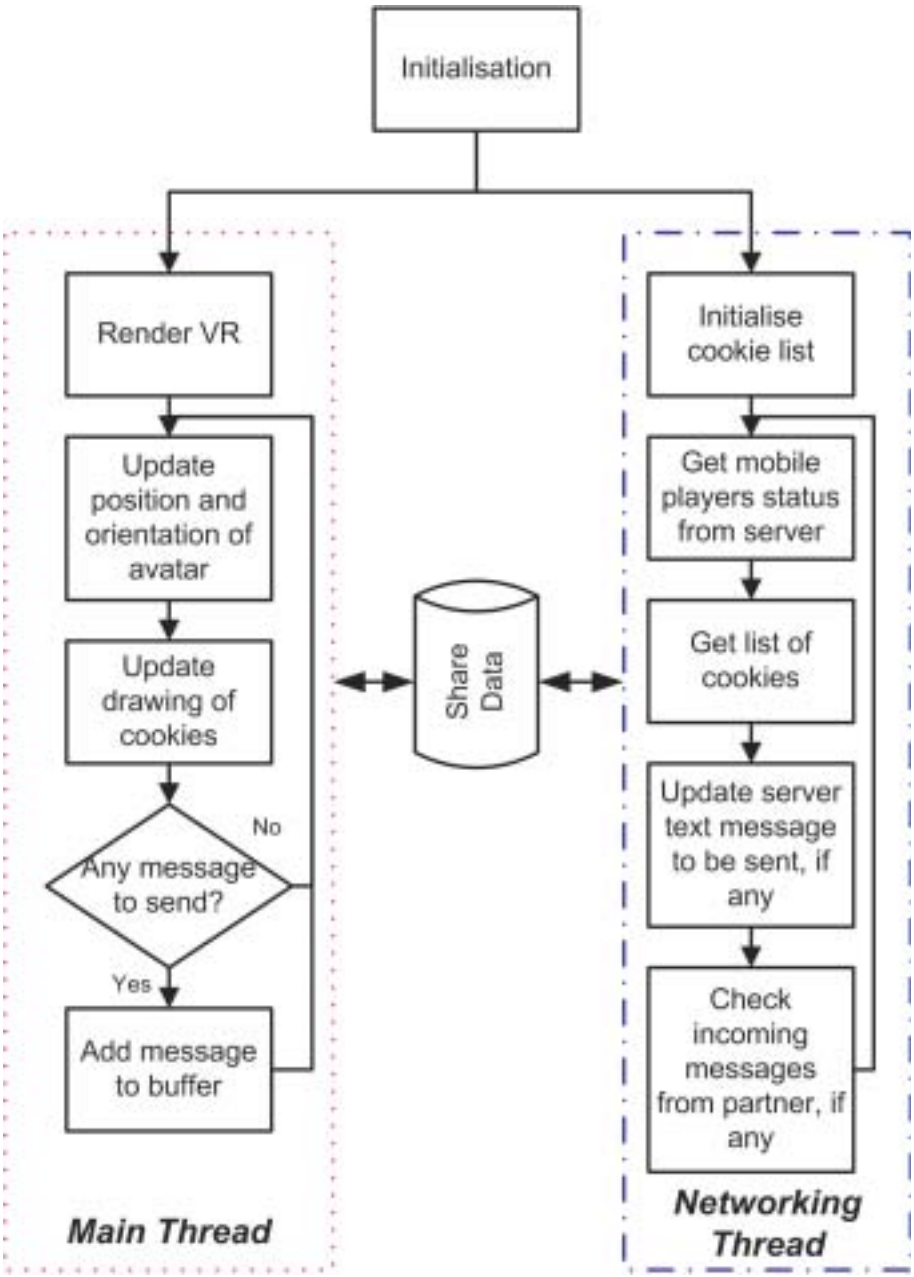


Figure 3.25: Flowchart of Helper system

1. Rendering of 3D VR view of the virtual Pac-World is done.

2. Avatar of each Pacman and Ghost are drawn based on the location and orientation of the respective player as broadcasted by the server.

3. Cookies that are not eaten are drawn.

4. If the user (i.e. the Helper) has typed in a message for sending, it will be stored in the buffer for the network thread to send.

5. The program jumps to (2).

3.2.4 Wearable Computer System

Similar to the Helper system, the Pacman\Ghost system runs multiple threads. The most important being the main thread and the network thread. In the Pacman system, the purpose of the main thread includes initializing of the hardware modules, updating of the visual display, collection and processing of sensors data, deciding if a cookie has been collected, and updating of shared data between the threads. In addition events from the keyboard (Twiddler2) inputs and from the Bluetooth are also processed and the corresponding shared data updated. The network thread exchanges this shared data with data from the remote server. Fig. 3.26 gives the flow of the main and network threads of the Pacman system. The program starts by initializing parameters and starting a new network thread and Bluetooth thread. Data flow between the threads is done through setting a shared semaphore. The Ghost system is similar to the Pacman system, except that it does not have cookies collection and Bluetooth events.

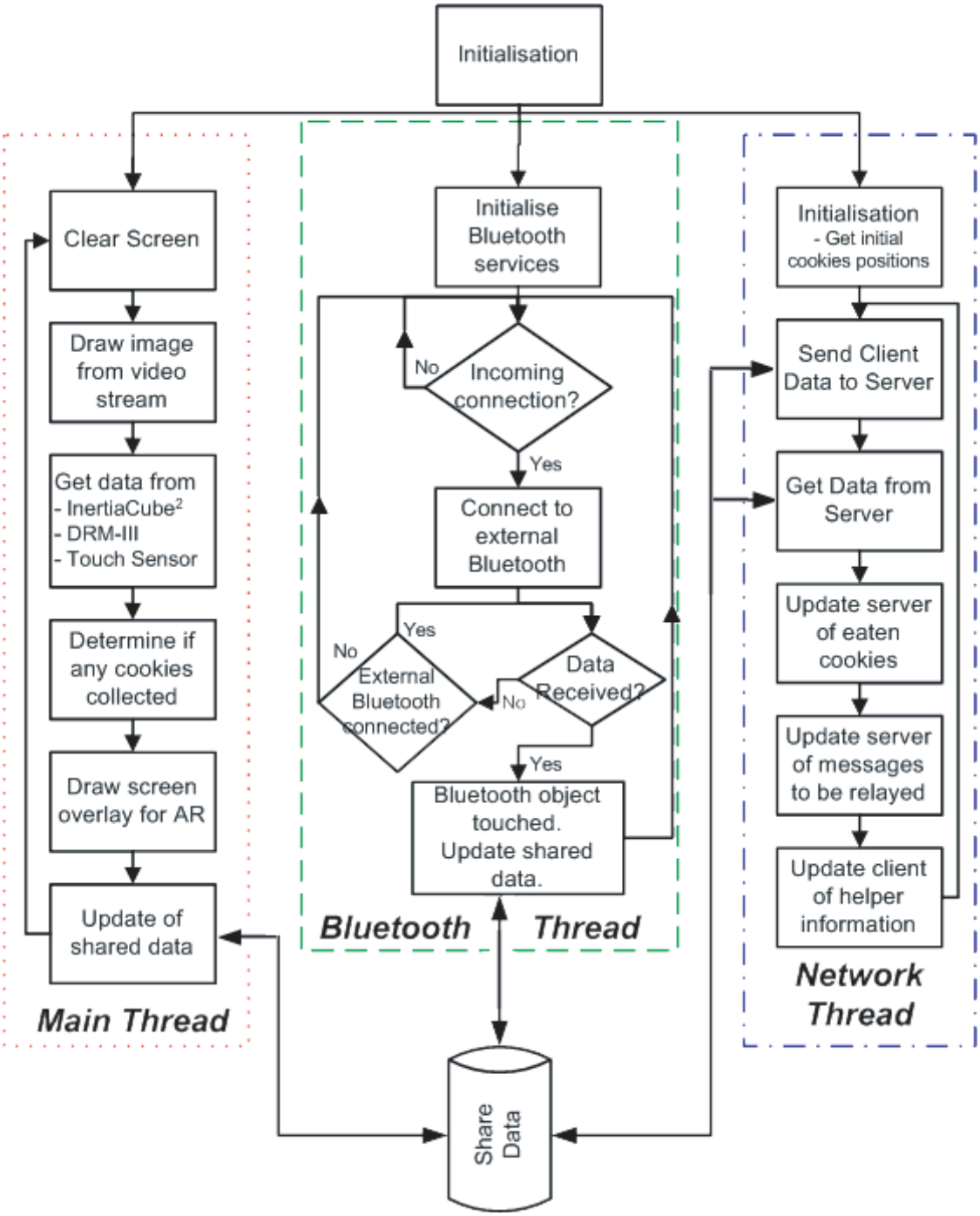


Figure 3.26: Flowchart of Pacman system

The flow of the network thread is as follows.

1. The position of all the cookies in the game are obtained from the server and stored as shared data.

2. Client status data is sent to server such as whether if Bluetooth embedded object has been collected.

3. A list of cookies eaten and the status of the game is obtained from the server.

4. The server is updated of cookies eaten by Pacman.

5. The server is updated if the Pacman has any message for her partner.

6. The server is checked to see if the Pacman's partner has any message for her.

7. The program jumps to (2).

The flow of the Bluetooth thread is as follows.

1. Bluetooth services are initialized.

2. Check is done for any incoming connection request. If not then loop (2), else program jumps to (3).

3. Incoming connection request is accepted.

4. A check is done for any data received at the L2CAP layer of the Bluetooth communication.

(a) If data is received, then program jumps to (5)

(b) If no data is received, than check if system is still connected to external Bluetooth device. If still connected, then program jumps to (4), else program jumps to (2).

5. Bluetooth embedded object has been touched. Shared data is updated.

6. The program jumps to (2).

The flow of the main thread is as follows.

1. The screen is clear and image from video stream obtained from camera is drawn.

2. Head orientation information is obtained from InertiaCube2. Positional data is obtained from DRM-III.

3. Check is made to see if touch sensor has been activated. If it has been activated and player is not Super-Pacman, the player is informed that she has been devoured. The shared data with regards to this is updated for sending to server by the network thread. The game ends. If it is not activated then program jumps to (4).

4. Base on the position of the Pacman, a check is done to see if any cookie is within a predefined radius of the player. If there is then the cookie is collected.

5. The screen overlay for the AR part of the display is drawn. This includes the rendering of virtual cookies in the HMD view of the player.

6. The shared data is updated with regards to the status of the Pacman and if any cookie has been collected.

7. The program jumps to (2).

Collection of Virtual Cookie

A Pacman collects a virtual cookie when her computed shortest distance from the cookie is within a predefined threshold. As the (x, y) positions of both the player and the cookies are with reference to an origin, the shortest distance could be calculated using Pythagoras theorem. This calculation, performed in the main thread, is done for each uncollected cookie in the game area. The network thread sends information of the cookies collected to the server.

Collection of Special Cookie

In the Bluetooth network between the Pacman system and the Bluetooth embedded object, the former acts as the server and the latter as the client. After initialization, the Bluetooth device on the wearable computer listens for incoming connection from the Bluetooth embedded object. After a link has been established, the player is informed of the Bluetooth embedded object if she has yet to collect it. When the player touches the Bluetooth embedded object the haptic information is sent via

the L2CAP layer of the Bluetooth communication to the wearable computer. The network thread updates the server of this collection.

Fig. 3.27 gives the flow of program in the SBC inside the Bluetooth embedded object. The flow of the program is as follows.

1. On power up, the Linux operating system in the SBC boots up.
2. When the boot-up completes, the Bluetooth program runs.
3. The SBC sends a PGM_START character to the controller unit serially to indicate boot-up has completed.
4. The Bluetooth services are initialized.
5. The program checks if a Pacman's Bluetooth device is found in the vicinity.
   (a) If yes, then connection to the Pacman's Bluetooth device is made. Program jumps to (6).
   (b) If not, then check if the controller unit has sent a SHUT_CHAR character.
      i. If yes, then jump to (7).
      ii. If not, then jump to (5).
6. Check if TS_ON character has been received from controller unit.
   (a) If yes, then touch sensor has been activated. Send data through the L2CAP layer of Bluetooth communication. Program jumps to (7).

(b) If not, then check if the controller unit has sent a SHUT_CHAR character.

i. If yes, then jump to (7).

ii. If not, then check if the system is still connected to Pacman's Bluetooth. If it is still connected, then program jumps to (6), else program jumps to (5).

7. Close connections and exit program.

8. Linux shutdown.

Sending of Predefined Messages

When the mobile player presses one of the selected keys on the Twiddler2, a keyboard event is triggered and a predefined message for the key is stored and relayed to the server via the network thread. Table 3.3 gives the list of the predefined messages that a Pacman and Ghost could send using the Twiddler2.

Table 3.3: List of predefined messages that Pacman and Ghost could send and the corresponding key on the Twiddler2

Key	Pacman's Messages	Ghost's Messages
'a'	Where is the Ghost?	Where is the Pacman?
'b'	Where is the treasure?	Has the Pacman become Super-Pacman?
'c'	Where are the remaining cookies?	Where should I move now?

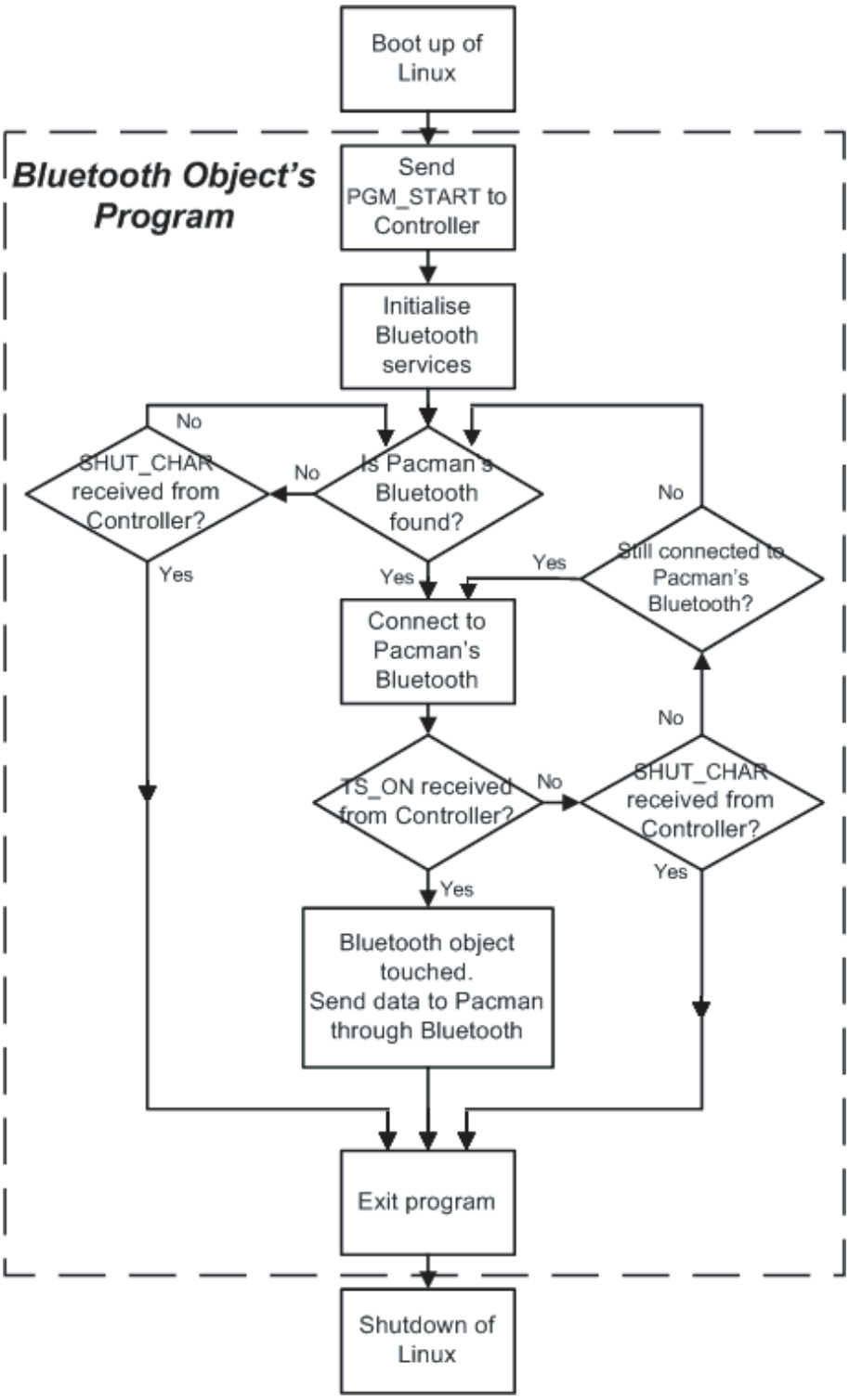


Figure 3.27: Flowchart of program in SBC inside Bluetooth embedded object

Getting Devoured

In the main thread when the touch sensor is activated, the system determines if the player has been devoured by looking at a few criteria. Firstly, there must be an enemy player within a predefined radius. This is to prevent false activation by a non-player. For the Pacman she must not be a Super-Pacman, and for the Ghost her enemy must be a Super-Pacman. Table 3.4 gives a summary of the conditions under which the Pacman or Ghost gets devoured. The server is updated if the player has been devoured.

Table 3.4: Conditions for devouring a Pacman or Ghost

Devouring Pacman	**Devouring Ghost**
Pacman must not be Super-Pacman.	Pacman must be Super-Pacman.
Ghost must be within a predefined radius.	Pacman must be within a predefined radius.
Ghost must have activated the touch sensor on the backpack.	Pacman must have activated the touch sensor on the backpack.

3.3 Design of Backpack

A backpack is used to house all the wearable computer hardware in a portable and comfortable manner for the user. The hardware used in the prototype system is heavy and bulky, with the A980 and its battery being the largest and heaviest (weighing almost 2.5kg). Hence it is essential that the backpack used is

large and sturdy. An initial backpack design was made by modifying commercially available hard-shell backpacks. Its use met with several problems, with the problem of overheating being the most serious. A new aluminium backpack was custom-made in our workshop specifically tailored to package the wearable computer. Fig. 3.28 shows the previous and current backpack designs. Fig. 3.29 shows the same user carrying each of the backpack design.

Figure 3.28: Two different designs of backpack

3.3.1 Previous Backpack Design

Fig. 3.30 shows the previous backpack design. Modifications has been done on the hard-shell backpack for the purpose of use as a wearable computer. Ventilation holes, made on the front of the backpack as seen in the figure, serves to provide ventilation to the A980 system running inside. A fan inside the backpack is attached

Figure 3.29: User carrying the previous and current backpacks

as seen inside the backpack to blow hot air out from within the backpack during operation. More ventilation holes are made at the side of the backpack. A panel on the side of the backpack features an on/off switch for the lead acid battery, and LEDs for visual affordance controlled by the controller unit.

Despite the considerations taken into improving the ventilation to the A980 system, the system still overheats and stalls after about an hour use. In addition there are other problems in using the backpack. The following is a list of areas that the previous backpack design is lacking in.

- The system overheats quickly causing the system to stall.
- It is not possible to debug program codes on the fly as the A980 is inside the backpack. Debugging would require the programmer to remove the system

Figure 3.30: Previous backpack design

from the backpack.

- As the A980's battery is inside the backpack, charging of the battery is inconvenient as it requires removal of the battery from the backpack.
- Switching on the HMD requires direct handling of the HMD control box inside the backpack and this is difficult in the already cramp backpack.
- The IEEE1394 cable (connecting the Fire-i digital camera to A980) disconnects easily from the IEEE1394 port on the A980. This is as the backpack is too cramp and adjustment of any component in the backpack may cause accidental disconnection of the cable.
- It is difficult to fix the individual components inside the backpack as it is not rigid inside the backpack. This adds extra difficulty in protecting circuit boards made in the lab.

Hence a new backpack had to be designed to overcome the above problems. The backpack should be comfortable to carry and look neat and compact.

3.3.2 New Backpack Design

Fig. 3.31 shows the six sides of the current backpack designed. The backpacks were made in our workshop using aluminium plates 1mm and 2mm thick. Considerations were taken in the design to overcome the limitations of the previous backpack design.

Figure 3.31: The six sides of the current backpack design

Fig. 3.32 shows the backpack with all the components housed in it. As seen from the figure, the backpack overcomes the shortfall of the previous backpack design with the following features.

- The A980 is minimally covered to allow proper ventilation.
- Debugging the program on the fly is made easy with the keyboard and mouse pad on the A980 exposed and easily accessible.
- Battery for the A980 could be easily charged as it is accessible from outside the backpack. A separate socket is made for charging of lead acid battery.
- HMD control box is placed close to an opening for easy reach to its control buttons.
- The IEEE1394 is not cramped into the backpack and so does not have the problem of accidental disconnection.
- The individual components can be fixed to the rigid aluminium frame.
- Most of the components of the wearable computer is hidden from view, making it neat looking.
- The backpack is designed to have minimal empty space internally to give it a compact look.

Figure 3.32: The current backpack with all the components housed in it

3.4 Issues Encountered

Several issues came up during the development of the system. The following are some of the more important ones.

3.4.1 Multiple USB Connection

For a Pacman system, 6 USB ports were previously required for connection to external devices, including direct/indirect connections to the wireless LAN card, Bluetooth USB dongle, DRM-III, InertiaCube2, controller unit and the Twiddler2. Desknote has 4 USB ports, making a USB hub necessary to extend the number of USB connections. However it is noticed that inputs from the Twiddler2 are sometimes not detected, which was later found to be attributed to the many USB connections. As the DRM-III utilizes only the RX and TX pins of the serial port (on the USB to serial adapter), the controller unit, previously using the RX and TX pins, was redesigned to use the DSR and CTS pins, making it possible for it to share the port with DRM-III. This reduces the number of components in the wearable computer as well as improves the detection of Twiddler2 inputs. A USB hub is not needed in the Ghost system as without the need for a Bluetooth USB dongle, only 4 USB ports are required.

3.4.2 Unstable Graphics Card Drivers

The graphics card drivers sometimes become corrupted after reboot of the computer. This could possibly be as the NVidia GeForce4 card is not generic but has been modified by the manufacturer to suit the Desknote system. As Microsoft Windows XP was used as the operating system, system restore points could be set to backup the driver for easy restoring of an uncorrupted driver.

Chapter 4

User Study on Human Pacman System

To gain useful feedback from the end user, an experimental user-study survey on the Human Pacman system was conducted. The aim is to find out from actual users their experience of the positive and negative aspects, interaction, and level of enjoyment in using the Human Pacman system. In these tests, the focus had been placed on the different novel experiences offered by the game.

Due to the limited budget, subjects are obtained from a pool of volunteers. A large number of subjects is not required. Nielsen [50] found that usability studies involving human computer interface requires only 15 subjects to discover all the usability problems. The cost, in terms of time and money, associated with employing additional subjects do not qualify the additional trivial amount of information obtained. The study involved 23 subjects between the age of 21 and 33, of which

eight were females and fifteen were males. Amongst these people, 39% indicated their level of expertise in computers as advanced, 43% as intermediate, and the rest as beginners. These subjects are either students or working adults. As it is not the aim of this study to examine how people of different age group respond to the use of the system, the subjects are chosen such that their age are not too vastly diverse. Young adults are chosen as their interaction with the system is less likely to be constrained by their physical incapability to carry the heavy prototype.

The experiment setup consisted of four parts. First the subject was asked to play traditional arcade Pacman games on a desktop computer for 5 minutes. Then a 3 minutes Human Pacman video was shown to give him or her a better understanding of the game. This was followed by a 15 minutes trial where the subject tried the roles of Pacman, Ghost and Pacman's Helper for 5 minutes each, along with other subjects taking a different role. An expert user acted as the Ghost's Helper. The trial is conducted in the university campus. Buildings are reflected in the virtual Pac-World as obstacles to make the association of the virtual and real world more intuitive to the subjects. Finally the subject had to fill up a questionnaire and to provide comments on the system. The experiment is conducted by an expert in the system to assist the subjects if they have any question.

In this chapter, we shall first look at the questions asked in the user study and the aim of each question used. This is followed by a discussion of the results obtained. The chapter ends off with a summary of the findings and motivation for future work.

4.1 Questions and aims

Table 4.1, 4.2 and 4.3 show the list of questions that were asked in the survey. Following each question are the reason for asking this question in the user study. Except for questions (i), (iv), (xi) and (xv), multiple choices are give for all other questions.

Fig. 4.1, 4.2, 4.3, 4.4, and 4.5 gives the user study results of all the multiple-choice questions listed earlier. The options for each question and the percentage of users who chose each option are given in the figures.

4.2 Discussion

In this section, the response to the questions will be discussed. All the data has been analyzed using statistical methods. Insights provided by users' comments

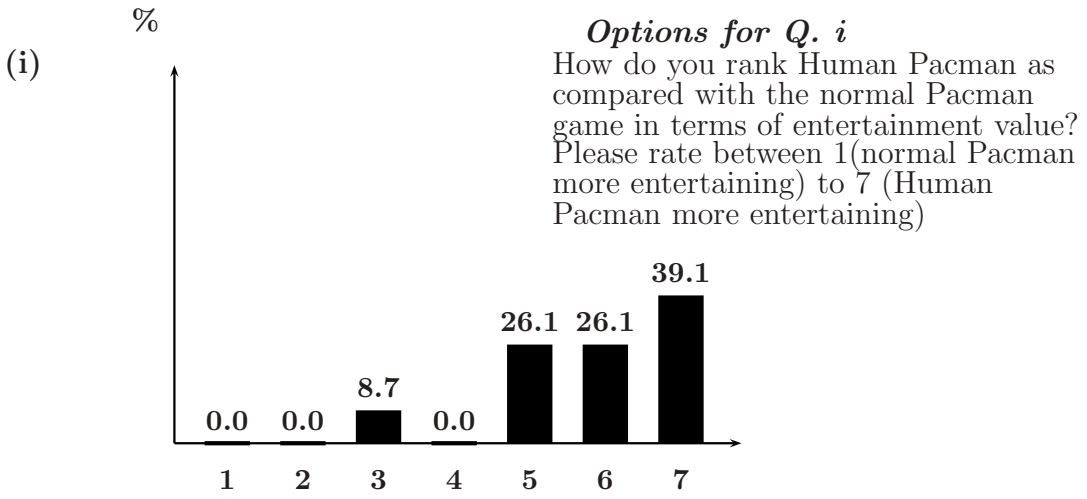


Figure 4.1: Graph results for question (i)

i. How do you rank Human Pacman as compared with the normal Pacman game in terms of entertainment value? Please rate between 1 (normal Pacman more entertaining) to 7 (Human Pacman more entertaining).	*Reason for asking: As the idea of Human Pacman originates from the previous arcade Pacman, the fundamental concepts of game play is similar. The question aims to find out if any value has been added to the old Pacman game in the new system.*
ii. How comfortable do you feel when you are using Human Pacman system?	*Reason for asking: The wearable computer system is still rather bulky and heavy compared to mobile devices such as phones, Game Boys, and PDA's. We want to find out if and how much it affects the level of comfort of the user when user dons it.*
iii. How intuitive do you think it is to collect cookies by physically walking through them?	*Reason for asking: In everyday life, collection of an item is seldom, if ever, made by walking through it. We seek to understand if user finds it intuitive to collect virtual cookies by walking through them just as is done in the original Pacman game.*
iv. Please rate, from 1 (lowest) to 7 (highest), the level of excitement of playing as a Pacman in Pac-World (first person experience), in comparison with the arcade Pacman that you can play using joystick/keyboard (third person experience)?	*Reason for asking: We want to find out if the immersive experience of Human Pacman makes the game more exciting. Arcade Pacman is used as the baseline for comparison as it is fundamentally similar in game-play.*
v. What do you think of displaying "cookies" as virtual objects augmented inside the real world?	*Reason for asking: From this question, we want to find out how realistic is the experience of collecting virtual cookies using AR.*

Table 4.1: Questions (i) – (v) of user study

vi. Does the physical collection of real object (special cookies) enhance the gaming experiences?	*Reason for asking: The collection of special cookies is a tangible interaction with a physical object that translates into a digital meaning (i.e. update of Pacman's inventory list). We want to find out if such graspable interaction enhances the game for the user.*
vii. How do you feel about the collaboration of the Pacman and the Helper in Human Pacman game?	*Reason for asking: The Helper role is added to bring in an element of collaborative experience between the human players. However it makes the game more sophisticated. The question aims to find out if such collaboration makes the game more exciting, or does it makes the game overly complex.*
viii. Does the VR mode give you a good idea about the game environment?	*Reason for asking: It is important for the Helper to understand the situation her partner is in so as to dispense the appropriate advice. The ability to comprehend the game environment from the VR mode "Godsview" is essential in helping her access the situation. Thus we wanted to see if this is confirmed in the user's opinion*
ix. What do you think of the "capturing" event implemented in our system (touching the Pacman by Ghost)?	*Reason for asking: The "capturing" event is a reflection of the naturalistic and physical approach Human Pacman took towards tangible interaction. We seek to find out if user enjoyed this feature.*
x. Do you like to play the Human Pacman game?	*Reason for asking: Having reflected on the game by answering the previous questions, the user is quizzed on the overall level of interest she has on Human Pacman.*
xi. Please rate, from 1 (lowest) to 7 (highest), the feeling of "social interaction" in being Ghost, Pacman, and Helper.	*Reason for asking: We want to find out the level of social interaction experienced by the user in each role.*

Table 4.2: Questions (vi) – (xi) of user study

xii. How do you compare this game with other computer games?	*Reason for asking: As Human Pacman aims to extend and differ itself from conventional human-computer-interface used in normal computer games, this question investigates how well has Human Pacman achieved it's aim.*
xiii. How do you compare Human Pacman with the traditional "Catch Me" game?	*Reason for asking: The question looks at how Human Pacman compares with simple, non-computer based games. Traditional "Catch Me" game is used as a basis for comparison due to its similarity with the "capturing" event in Human Pacman. We want to see if adding the fantasy element has any benefit in the user's enjoyment over a normal catch game.*
xiv. If there is such a game in an amusement park, how much are you willing to pay to play the game?	*Reason for asking: Currently acquiring the whole Human Pacman system requires high overhead. So a commercially viable version of the system could only be sustained based on a pay-per-use basis, which is a norm in amusement parks. The question investigates the amount of revenue per person the system could bring in if implemented commercially. The question is asked to see if there is a link between such research systems, and potential commercial use.*
xv. How often do you play computer games?	*Reason for asking: The question finds out how frequent does the user play computer games. This enable us to find out if her love for conventional computer games would influence her desire to play Human Pacman.*
xvi. Please give us some comments on how we can improve the system and what the current drawbacks of the system are.	*Reason for asking: We seek user's comments on our system to further improve the system in areas we may have neglected.*

Table 4.3: Questions (xii) – (xvi) of user study

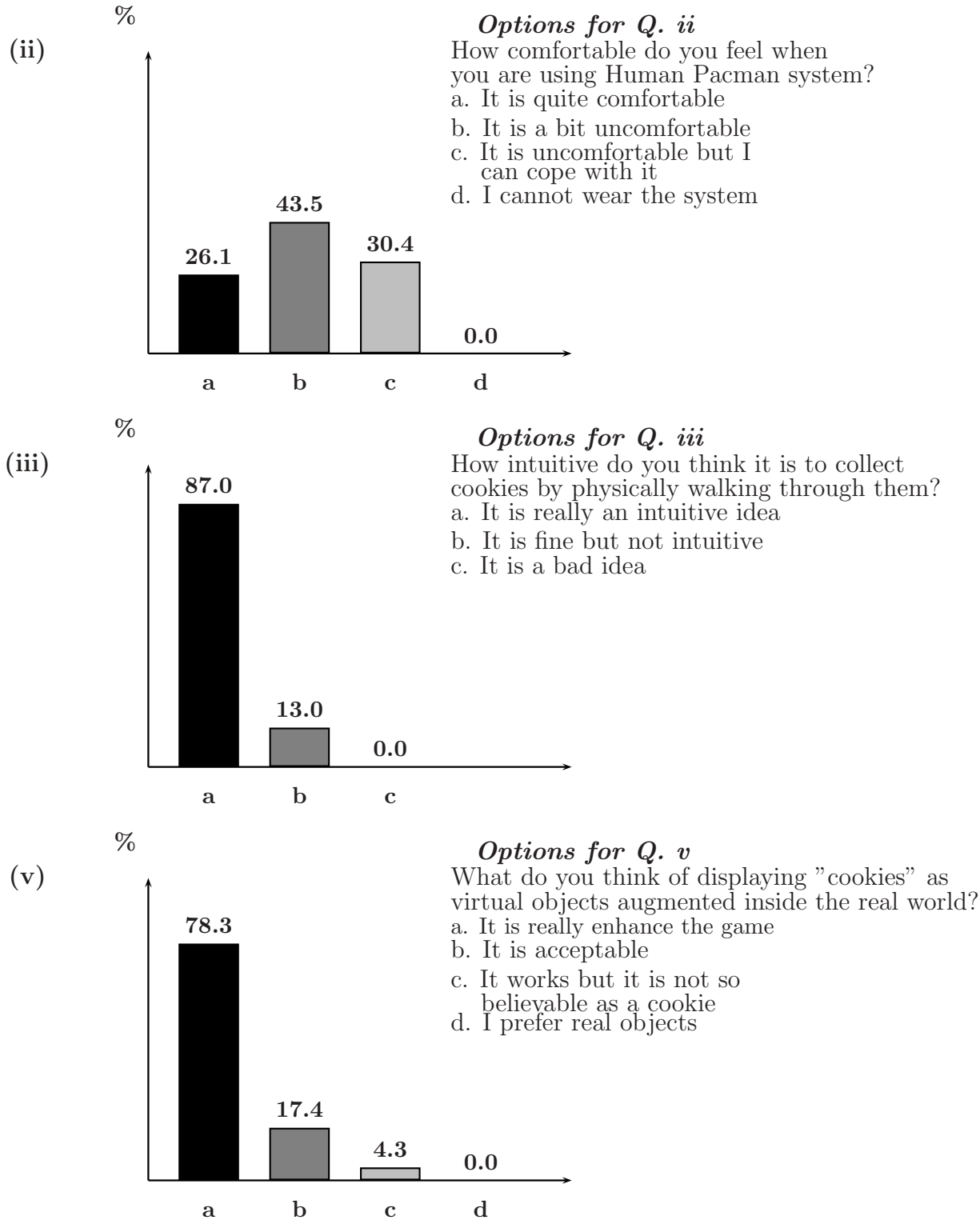


Figure 4.2: Graph results for multiple choice questions (ii), (iii), and (v)

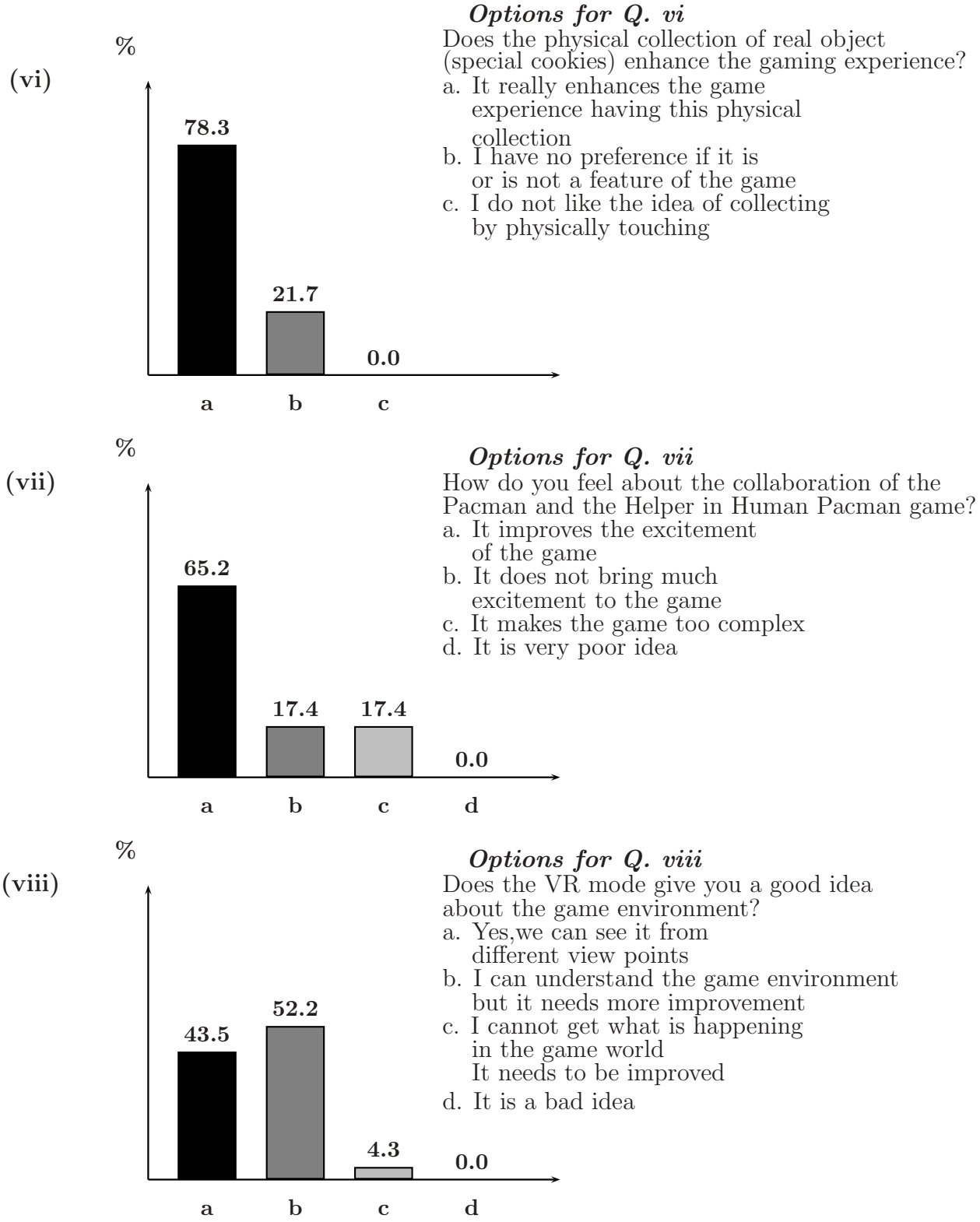


Figure 4.3: Graph results for multiple choice questions (vi), (vii), and (viii)

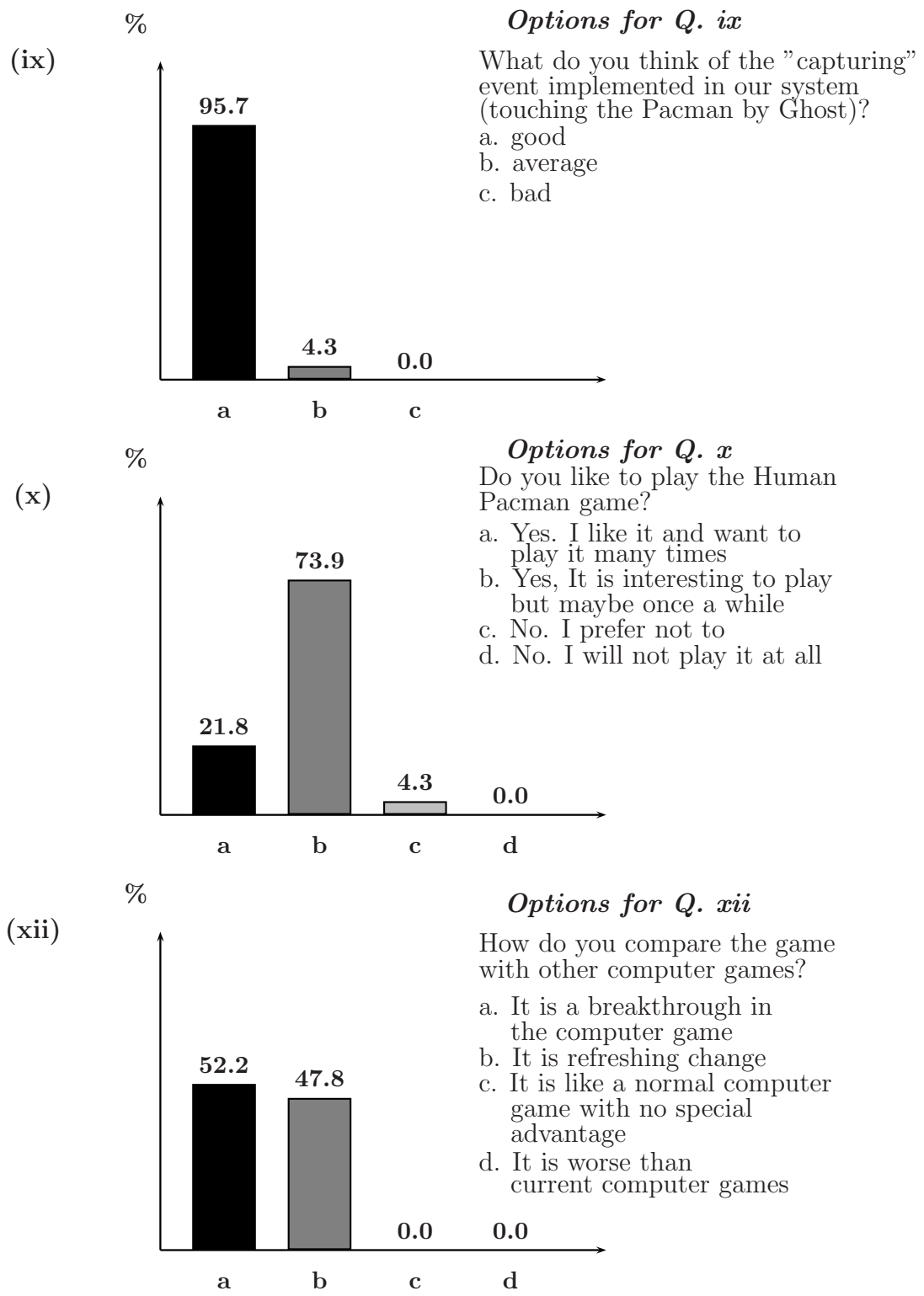


Figure 4.4: Graph results for multiple choice questions (ix), (x), and (xii)

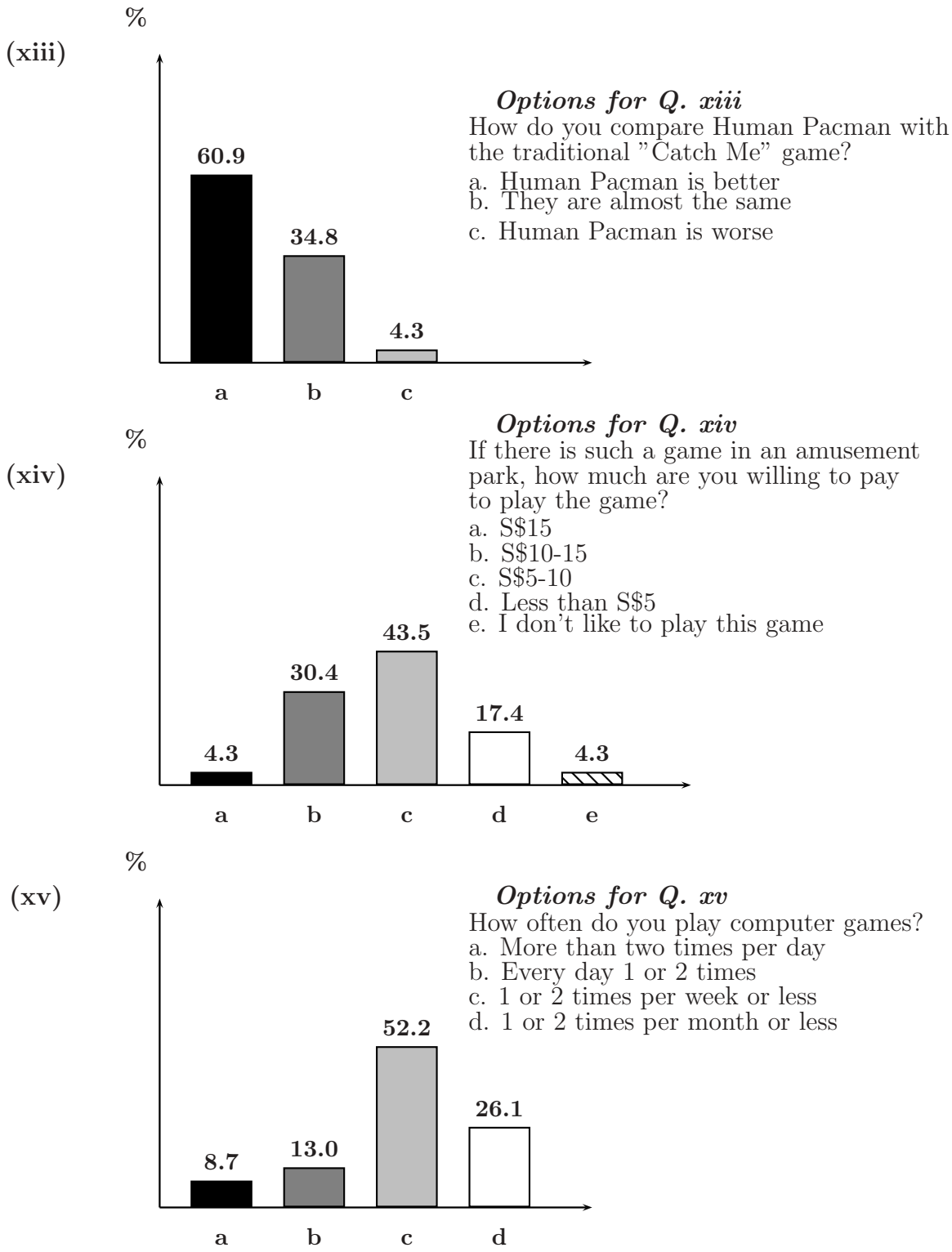


Figure 4.5: Graph results for multiple choice questions (xiii), (xiv), and (xv)

from question (xvi) will be used in the analysis too.

Questions (i), (x), (xii), and (xiii) examine how well received is Human Pacman by users; and how it compares with respect to other types of game in terms of user preference. As seen from the respective findings given in Fig. 4.4 and 4.5, most of the users are enthusiastic about Human Pacman. However it is noted that when compared with traditional "Catch Me" game, 34.8% of the users gave a neutral stand in their preference. Results from (i) gives an average rating of 5.85 (variance of 1.46) indicating that Human Pacman is much more favoured than normal Pacman in term of entertainment value. Statistical analysis with T-test confirms the significance of our inference ($p = 6.72 \times 10^{-7}$).

The element of physicality may have been the pushing factor for the preference shown towards Human Pacman over arcade Pacman and conventional computer games. However this is not so much of a benefit over the traditional "Catch Me" game (which by itself is a game that involves a high level of physical participation). A number of users commented that they like the idea of "physical involvement" and "physical movement" in Human Pacman. Some said that such movement is a good form of exercise. Note that 60.9% of the users still prefer the Human Pacman over the "Catch Me" game. This indicates that the element of physical involvement in Human Pacman is not its sole attraction. The immersive experience in the role playing of Pacman could be another element that users enjoyed over arcade Pacman and conventional computer games. Findings from question (iv) gives the average level of excitement rated by the subjects to be 6.0 (variance of 0.182) for the first

person experience in Pac-world and 3.33 (variance of 0.97) for the third person experience in arcade Pacman, indicating a higher level of excitement in the former. Non-parametric two conditions Wilcoxon statistical test confirms the significant difference in the feeling of excitement by the players in level of $p = 4.88 \times 10^{-4}$.

As reflected by many users, the backpack holding the wearable computer system is bulky and heavy, and the HMD is cumbersome to wear. As seen from results obtained for question (ii), 73.9% of the users found the wearable computer to be uncomfortable. This could be the deterring factor for 77% of those who indicated that they like to play Human Pacman but would refrain from playing it frequently. The absence of equipment weighing down the user could also make traditional "Catch Me" game more attractive as one is unencumbered physically.

Despite not being a normal day-to-day experience, collection of virtual cookies by walking through them is deemed to be intuitive by 87% of the users (as seen in Fig. 4.2(iii)). The rest found the experience to be acceptable, though not intuitive. Findings for question (v) shows that 78.3% found that the use of virtual cookies enhances the game, whereas 17.4% feels that the cookies are acceptable as virtual objects but it fails to enhance the game.

A shift in alignment of virtual cookies from its supposed absolute position in real space (caused by sensing drifts), mentioned by some users, could be the reason why virtual cookies seems lacking to some users in being realistic and thus its inability to enhance the game. As the view of virtual cookies are calculated with respect to the user's location, any discrepancies of her exact position may lead to

the "shifting" of the absolute position of the virtual cookies in real space. Since the system uses DRM-III and dead-reckoning method to estimate displacement of the user, an inaccurate estimation of her stride (which varies with individual) or the wrong count for the number of steps taken will introduce error in the estimate. The location of the user thus computed is an estimate and may not reflect her true position in the physical space. Some users have reflected that the DRM-III module failed to sense all the footsteps taken by the user.

Users also found the visual cue of the cookies collection (i.e. cookies disappearing from AR world when collected) to be weak and insufficient in providing a "better feel" of collection. A number of users suggested using sound, for example a "beep", to indicate collection of each virtual cookie. A lack of realistic affordance of virtual cookies makes reliance on other cues, to indicate collection, more important.

On the issue of tangible interaction element in Human Pacman, 78.3% (as seen in Fig. 4.3(vi)) found that the graspable interaction offered by the collection of real objects enhances the game. The other 21.7% gave a neutral response towards having this collection as part of the game. Almost all the users indicated in question (ix) that they like the "capturing" event. Despite being both naturalistic interaction with the physical world, users seems to like the "capturing" event more. This suggests the physical human-to-human interaction in the process of "capturing" makes the event more enjoyable.

In response to question (vii), 34.8% of the users did not find that the Helper

role makes the game more exciting. In fact 17.4% found that it makes the game too complex. Results obtained for question (xi) shows that the feeling of having social interaction has rated mean of 5.67, 5.41 and 4.17 with variances of 0.97, 1.17 and 0.88 in playing as Pacman, Ghost and Helper respectively. The non-parametric Wilcoxon test shows insignificant difference in level of social interaction between Pacman and Ghost ($p = 0.5$), while the difference between being Ghost and Helper is significant with $p = 0.0039$. Helper role is perceived to have lesser social interaction as compared to the other roles. As the study does not involve a trial of the role of Helper in a full-scale long game, the users may have not been able to fully comprehend the essence of the Helper role.

As reflected by the response for question (viii) almost all of the users are able to comprehend the game environment from the VR mode. It is noted that more than half of the users feels that more improvement still needs to be made for the VR mode. Users would like to see better 3D graphics and more variety of virtual objects in the VR world. Perhaps the lack of visually appealing 3d interface as compared to that found in commercial computer games makes the Helper role less attractive (our 3d models are non-commercial and designed in the laboratory by students).

Based on the results from question (xiv), the average amount the users are willing to spend to play Human Pacman in amusement parks is SGD$8.15 with a variance of 16.5. This is in the typical price range of amusement rides available locally. It is noted that the 17.4% who would pay less than S$5 to play Human

Pacman also indicated in question (xv) that they seldom play computer games. Perhaps they do not enjoy playing games and are therefore less willing to spend on it.

4.3 Summary of Findings and Future Work

From the findings we can say that users like the idea of Human Pacman as a whole. This is seen from their attitude towards playing the game, their willingness to pay to play the game, and their preference of Human Pacman over other types of games. It is promising that users were positive about the physical interaction aspects of the game such as the first person point of view and the tangible interactive elements. However it is clear that improvements can be made to reduce the size and weight of the wearable computer.

On the individual elements of Human Pacman, the collection of virtual cookies is well accepted though improvements can be made to make the whole experience of moving towards virtual cookies and collecting it more realistic. Sound could be added as a cue to the collection. More accurate and precise tracking device could also be developed to minimize error in location tracking, which was a factor of disapproval from the users.

Though both the "capturing" event and collection of physical objects in the game adds value to the game, it is found that the former is better liked. This could be because during the "capturing" of Pacman, physical human-to-human

interaction is involved, along with other forms of human interaction that comes into play (e.g. shrieking). The study also shows that immersive experience is valued by users. Users like to be "physically involved" in this first-person gaming experience. This is positive reception is important to note, as the tangible and physical aspects introduced into Human Pacman are one of its major thesis.

As for the issue of having the Helper role in Human Pacman, a less than enthusiastic response is obtained from a large portion of users, with half of these users feeling that it makes the game too complex. A more in-depth study will in future be made on why is such sentiment prevalent. A preliminary suggestion is the lack of a full long trial of the game in the experiment for the user to fully appreciate the important role played by the Helper. Improvement in the aesthetical value of the VR mode may also make the Helper role more enticing.

Despite encouraging responses from the users, the Human Pacman system is still unsuitable for commercial production for the following reasons.

- The cost of each set of wearable computer system is very high as they are lab prototypes.
- Bulkiness and heaviness of the prototype wearable computer and HMD restricts the user to only walking, though the short response time of the system software does totally allow for quick movement like running.
- Accurate and sensitive positioning system is needed to enhance the realism of the game.

- Robustness of the system needs improvement.

Chapter 5

Indoor Maze Version

A maze version of the Human Pacman system was developed for indoor, small area game play suited for demonstration and exhibition purposes. The indoor maze version was shown during CHI2004 in Austria, and ACE2004 in Singapore. The maze, measuring 2.4m x 2.4m, is divided into a 4 x 4 squares matrix. Fig. 5.1 shows the layout, the physical maze, and its virtual representation. The maze layout indicates the way the paths of the maze in both the physical and virtual worlds are routed. "©" in the maze layout indicate the locations of virtual cookies in Pac-World. Correspondingly virtual cookies will be seen at these locations in the VR and AR world. The position of special cookie is indicated as "OBJ" in the figure. The special cookie in the VR world is also labelled in the figure. A Bluetooth embedded object is placed at the corresponding location in the physical maze. As the maze is small, there are only one Pacman and one Ghost in the game. The starting positions for the Pacman and Ghost are indicated in the maze layout.

Figure 5.1: The maze layout, the physical maze, and its virtual representation

Fig. 5.2 shows the Pacman and Ghost at their starting positions in both the physical world and the corresponding VR world.

Fig. 5.3 shows the physical view, Pacman's AR view, Ghost's AR view, and Helper's VR view when Pacman collects a Bluetooth embedded object and becomes Super-Pacman. As seen in the figure, Pacman becomes a Super-Pacman when she touches the Bluetooth embedded object. She is alerted visually of her Super-Pacman status in the lower left corner of her AR view as shown. A bar counting down in her AR view indicates to her the amount of time left before she becomes a normal Pacman again. An enlarged red 3D Pacman avatar instead of a normal

Figure 5.2: Pacman and Ghost at their starting positions in both the physical world and the corresponding VR world

yellow one in the VR view indicates to the Helpers that the Pacman is now a Super-Pacman. The Ghost is not alerted and remains oblivious to the Super-Pacman until her Helper informs her or she is devoured.

Fig. 5.4 shows the physical view, Pacman's AR view, Ghost's AR view, and Helper's VR view when Pacman devours Ghost by touching her backpack. Pacman can only do so when she is a Super-Pacman as seen in the figure. When the Ghost

Figure 5.3: The real, AR, and VR views of Pacman becoming a Super-Pacman in the maze

gets devoured, she is alerted through her AR view that the game is over for her. Pacman has to continue to collect all the virtual cookies in the game area before she wins the game. After the Ghost gets devoured, her avatar disappears from the

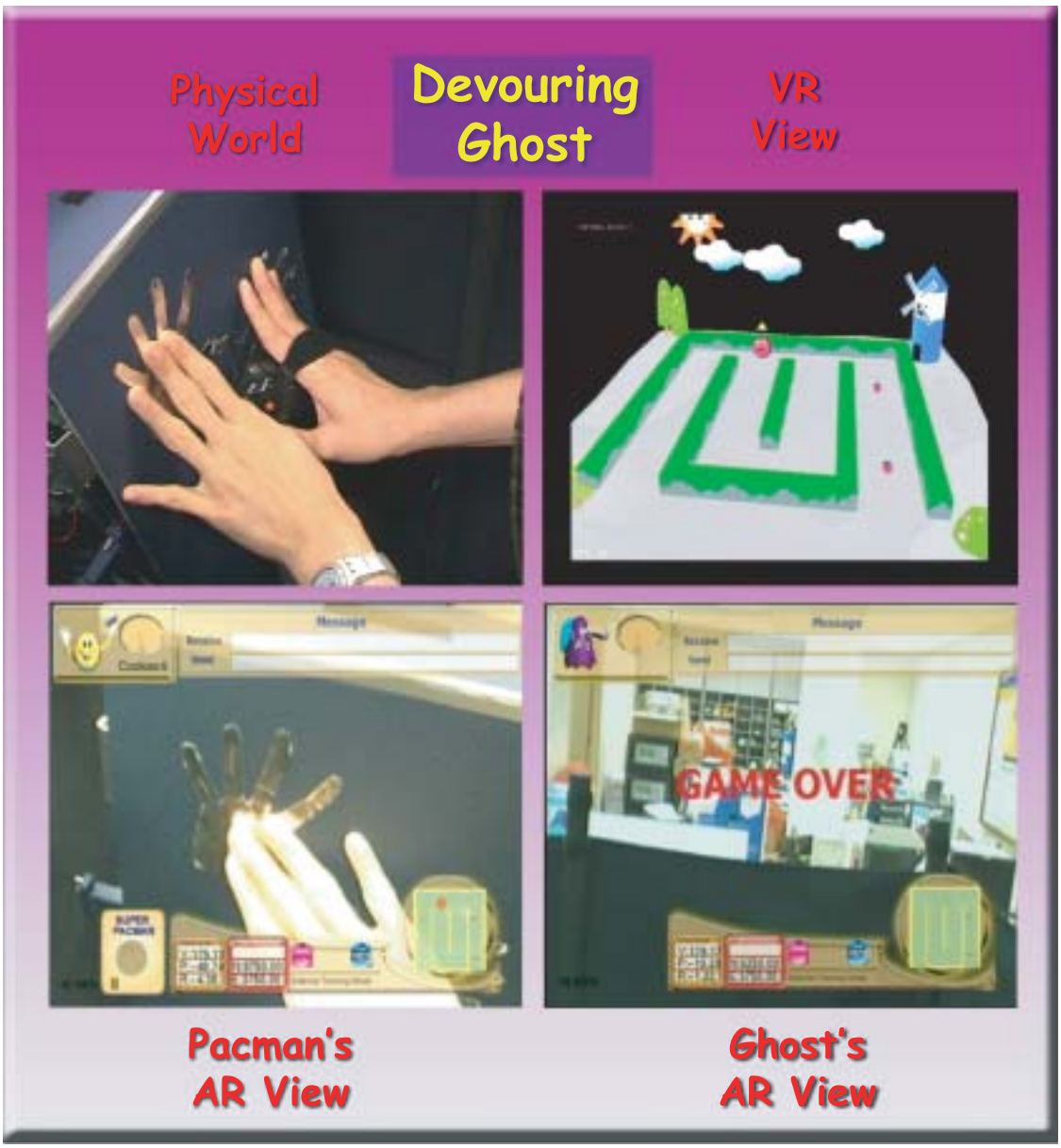


Figure 5.4: The real, AR, and VR views of devouring a Ghost in the maze

VR view as shown.

Fig. 5.5 shows the physical view, Pacman's AR view, Ghost's AR view, and Helper's VR view when Ghost devours Pacman by touching her backpack. Ghost can only do so when Pacman is not a Super-Pacman as seen in the figure. When

the Pacman gets devoured, she is alerted through her AR view that the game is over for her. As there is only one Pacman in the maze version of Human Pacman, Ghost is alerted through her AR view that she has won the game. As shown, the Pacman avatar disappears from the VR view after she gets devoured.

Figure 5.5: The real, AR, and VR views of devouring a Pacman in the maze

Fig. 5.6 shows a Helper messaging her partner Pacman. As seen in the figure, the message the Helper sends to her partner Pacman is displayed in the message box at the top of Pacman's AR view. Pacman can message the Helper using her Twiddler2 device as seen in the figure. Fig. 5.7 shows the exchange of messages between Pacman and her Helper.

Figure 5.6: Helper messaging her partner

For the Pacman to win the game, she has to collect all the virtual cookies in the maze. This is illustrated in Fig. 5.8 where the Pacman collects the last cookie and wins. A visual alert in her AR view informs her that she has won the game. Ghost

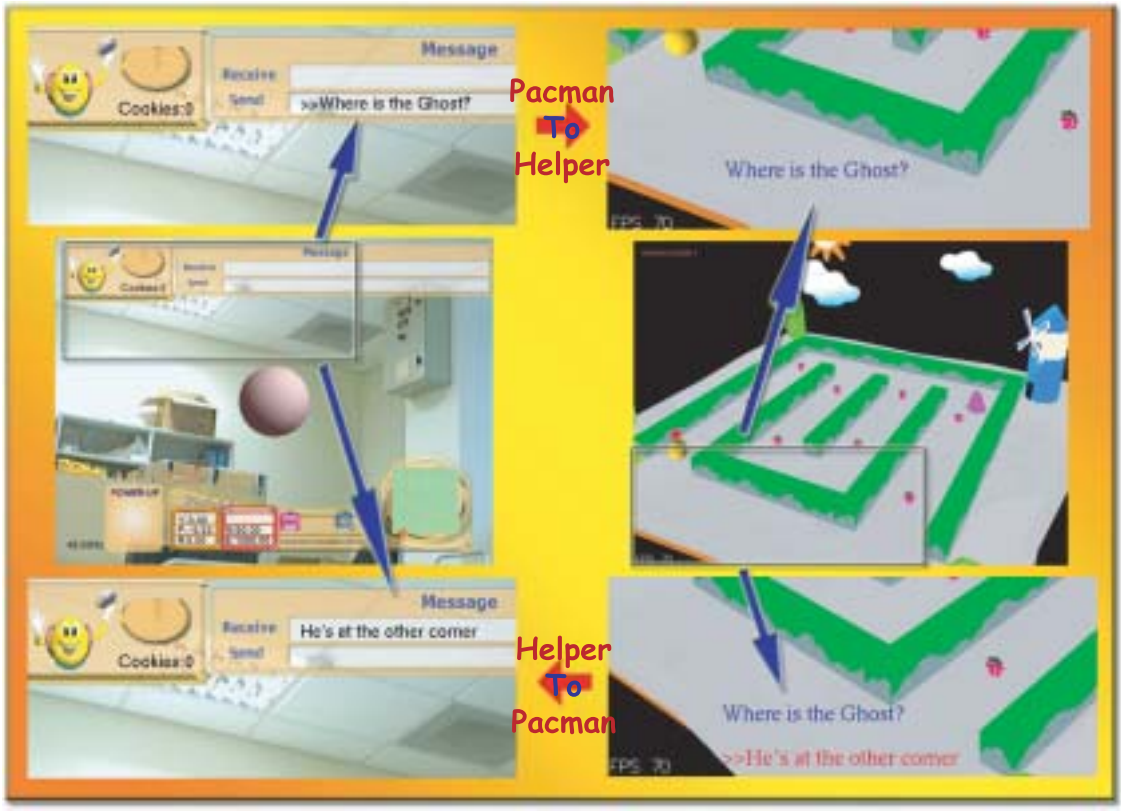


Figure 5.7: Exchange of messages between Pacman and her Helper

on the other hand loses the game and disappears from the VR view as shown.

5.1 Location Tracking of Players

Initial trials show that the DRM-III is not accurate enough for our purpose under such small and built up environment. DRM-III is based on a pre-defined stride length estimation of 60cm for all players. As stride length varies from person-to-person and there is no means of recalibration of position, e.g. using GPS indoors, any small error in estimation will accumulate as players move around. The error becomes largely noticeable through the drift in absolute positions of virtual cookies,

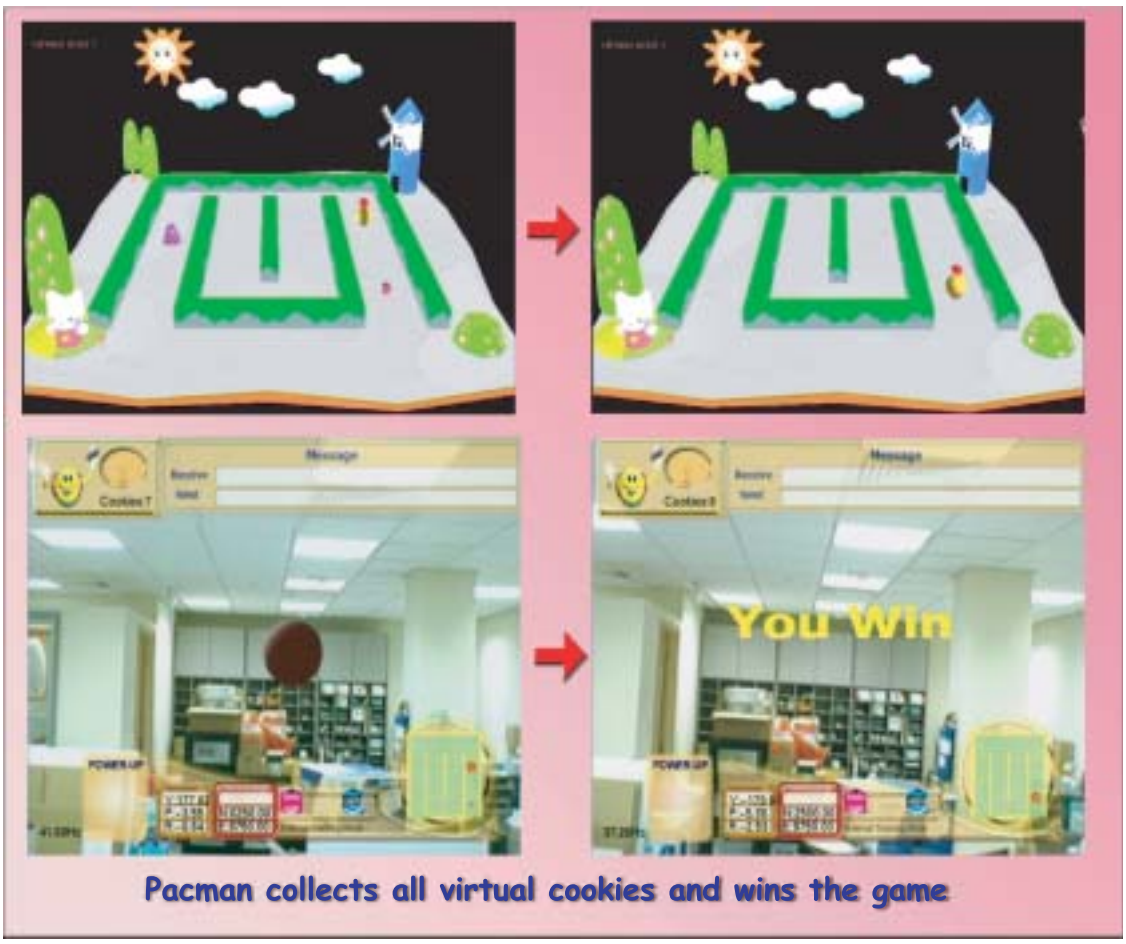


Figure 5.8: Pacman collects all virtual cookies in the maze and wins the game

and this is especially so in the small maze where the walls are only 60cm apart. Hence a modified system needs to be implemented to prevent such errors.

In this chapter, we shall examine the infrared (IR) tracking system employed. We will also look at the shortcomings of the design.

5.1.1 Infrared Tracking

In the IR tracking system, IR emitters are attached to the underside of the wearable computer. Each IR emitter, controlled by a microcontroller, transmits a uniquely

coded signal. As the player walks within the maze, IR receivers mounted at regular intervals in the floor board pick up this signal. The signal is decoded to give the identity of the user and hence her position. This data is sent to a server system for processing and broadcasting.

Fig. 5.9 shows a maze block used in the construction of the maze. Each block measures about 60cm x 60cm and represents a single matrix in the 4 x 4 square matrix. Hence 16 of such maze blocks are used in the maze. At the center of the maze block is an IR receiver. The IR circuit board is hidden under the panel on the maze block as shown. To connect the IR circuit boards together, cables are ran through the sides of the maze blocks and are interconnected in a manner shown in Fig. 5.15. This will be described later in Sec. 5.1.1. Fig. 5.10 shows the maze blocks fitting into each other like a jigsaw puzzle. Cables that run through the side of the blocks are connected together.

The main hardware components of this system include the IR emitters, IR receivers, and a receiver-to-server interface. As the maze is too small for more than a pair of Pacman and Ghost, only two unique codes are required – one for Pacman, one for Ghost.

IR Emitter

The emitter circuit is shown in Fig. 5.11. The PCB design and actual circuit board made are shown in Fig. 5.12 and Fig. 5.13 respectively. The main modules in this circuit are highlighted on these boards, namely the linearly regulated power supply,

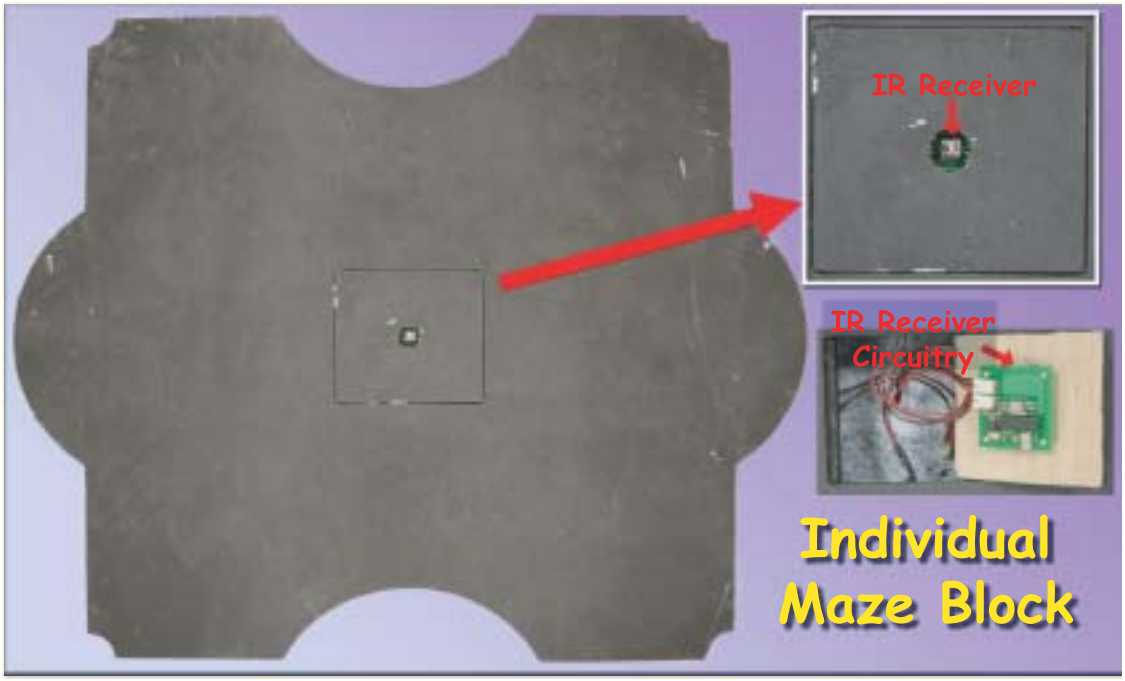


Figure 5.9: A maze block used in the construction of the maze

microcontroller, IR emitter module, and the selector switch module.

A 7805 linear voltage regulator is used for regulating the power supply at 5V. In the IR emitter module, the IR emitter used is the TSUS5400 [51] . TSUS5400 is a low cost emitter with peak wavelength of 950nm and angle of half intensity at $\pm 22°$. It is connected to the microcontroller PIC16F76 (see Sec. 3.1.1) through pin RB7. A square wave of frequency $\frac{1}{T}$, modulated with a 38kHz square wave, is generated by the controller at the IR emitter (see Fig. 5.14). This is later demodulated at the receiver end. The frequency $\frac{1}{T} = 700Hz$ is used to denote Pacman and $\frac{1}{T} = 1000Hz$ for Ghost. The 2N2222 NPN transistor acts as a current buffer for the emitter. A 100Ω resistor is connected in parallel to the emitter to tie the voltage across it to zero during the 'off' phase of the transistor. A selector switch is added for selection

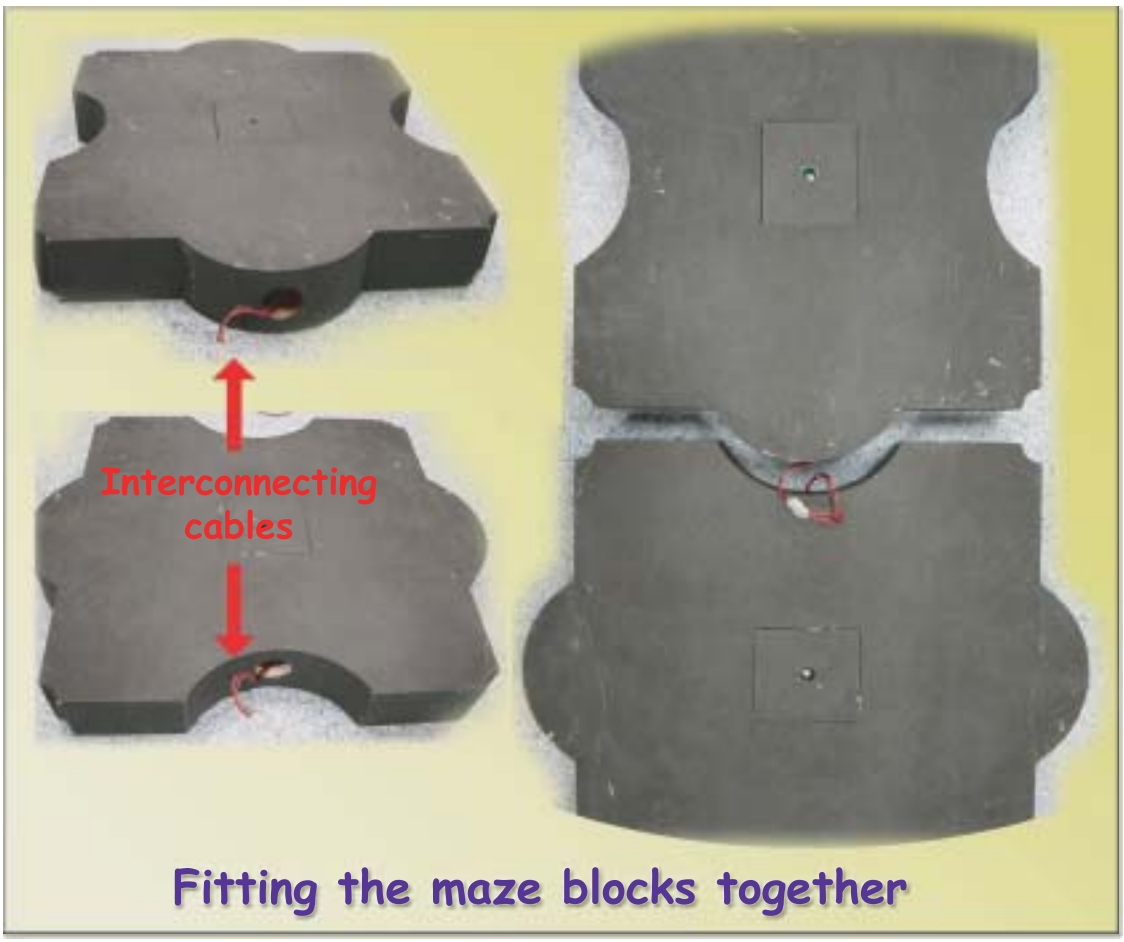


Figure 5.10: The maze blocks fitting into each other like a jigsaw puzzle

of either the Pacman or Ghost frequency.

IR Receiver

An IR receiver is attached to the center of each maze block, as shown in Fig. 5.9, in the 4 x 4 square matrix of the maze to track the players. When a player stands in a square the receiver should identify her and sends her ID and its location to a server. A daisy chain topology as shown in Fig. 5.15 is used to connect the IR receivers to the server. Starting from node 1, a data packet is relayed from node

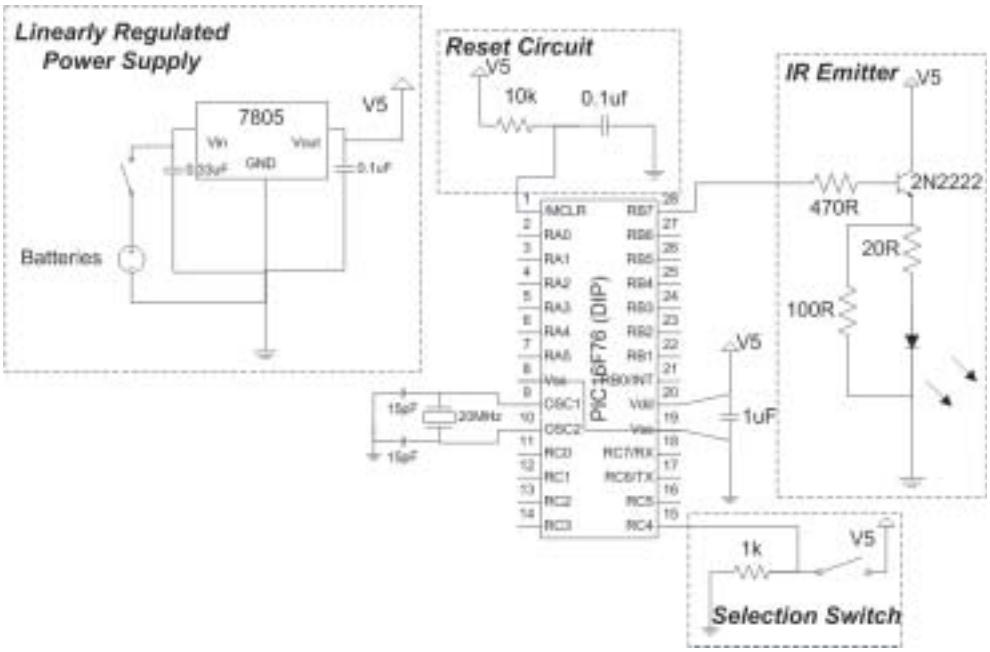


Figure 5.11: Circuit diagram of IR Emitter

to node, with each subsequent node adding its own data byte just before the end byte in the packet. An example packet received by the server is shown in Fig. 5.16. The data byte identifies the node and the ID of the player, if any, it detects. The assumption is that only one player will stand in the square, which is valid given the size of the square block. Fig. 5.17 illustrates the packet sent by node n. The data byte gives the node ID and one of three possible states of the receiver. The codes for detection of Pacman, Ghost and neither are 011, 110, and 101 respectively. Note that the 'distance' between any two codes is 2. Hence it requires 2 error bits to give a wrong code. As the data bytes are in sequence starting from node 1 the server could identify the origin of the byte without looking at the node ID in the data byte.

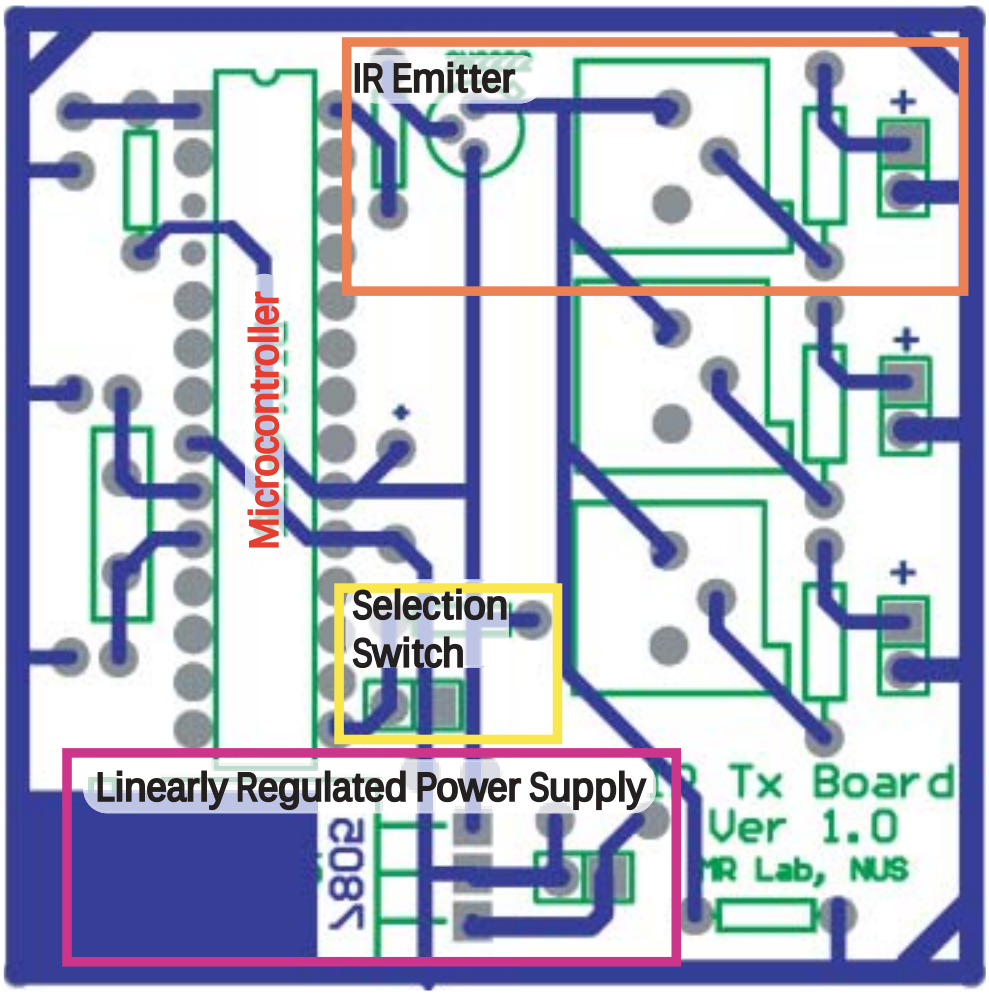


Figure 5.12: PCB layout of IR Emitter with main modules labelled

The receiver circuit is shown in Fig. 5.18. The PCB design and actual circuit board made are shown in Fig. 5.19 and Fig. 5.20 respectively. The main modules in this circuit are highlighted on these boards, namely the linearly regulated power supply, microcontroller, IR receiver module, external interfacing module and the pull-up module.

A 7805 linear voltage regulator is used for regulating the power supply at 5V. In the IR receiver module, the IR receiver used is HIM602H [52]. Its peak wavelength

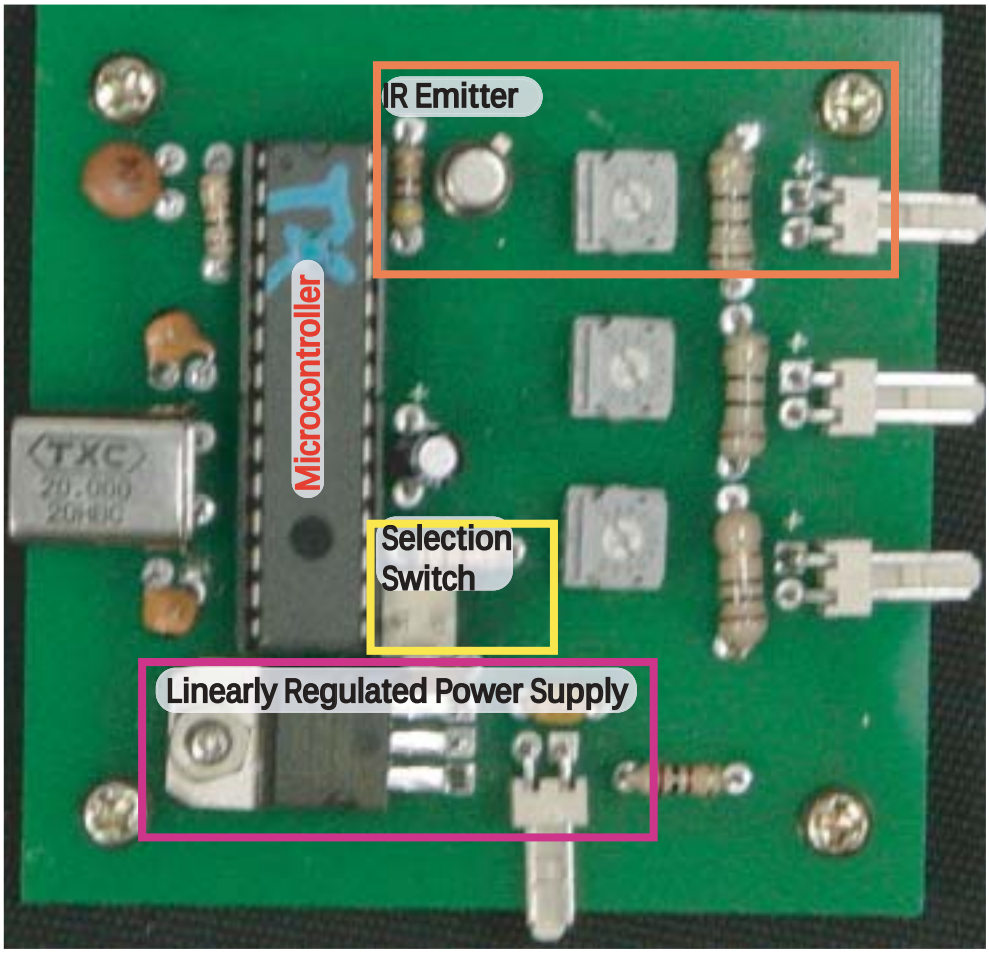


Figure 5.13: Actual circuit board of IR Emitter with main modules labelled

of 940nm is close to that of the TSUS5400. It has a half intensity angle of $\pm 50°$. The HIM602H demodulates the signal it receives and its output is connected to RB0 of the controller.

Each circuit is made modular such that its ID depends on its position in the chain, rather than a predefined node number. The 'previous node' connection of the current node connects to the 'next node' connection of the node before through the external interfacing module. RC4, which is tied to high through a $1k\Omega$ resistor

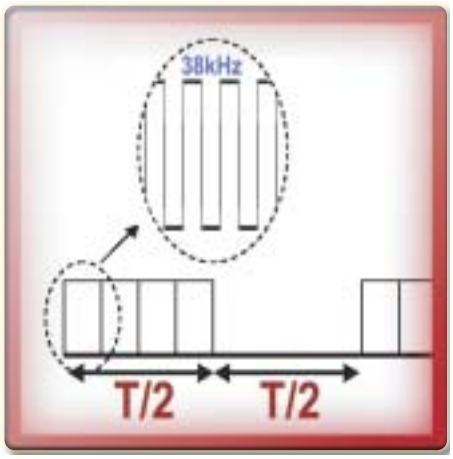


Figure 5.14: Signal generated at the IR emitter

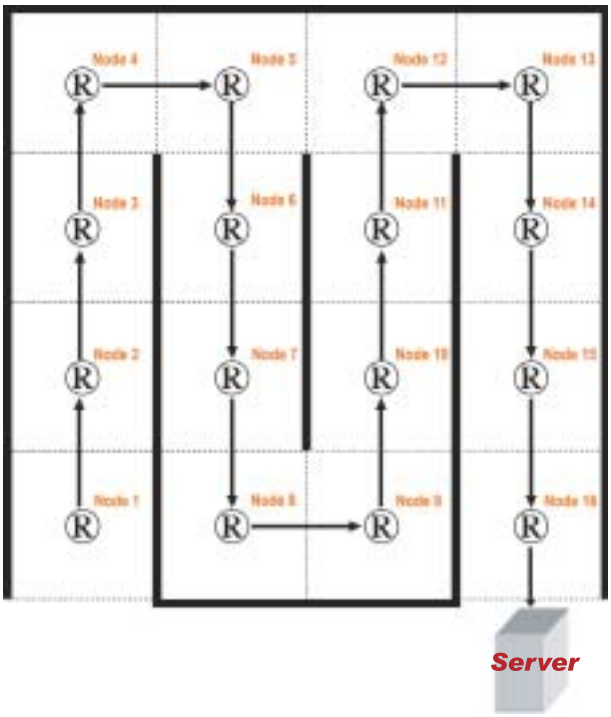
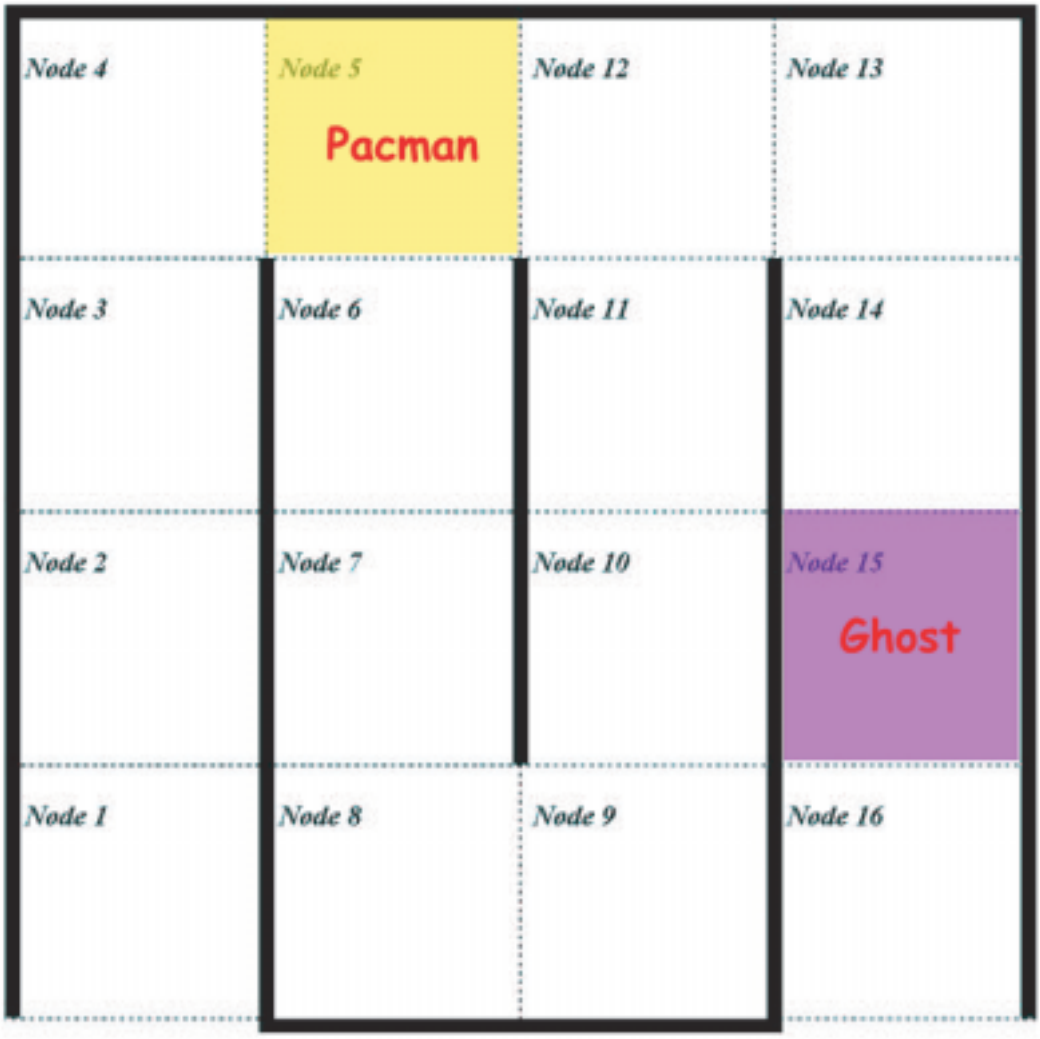


Figure 5.15: Topology used to connect the IR receivers

in the pull-up module, connects to DP pin of the previous node, where it is tied directly to ground. Hence the node where RC4 is read as high is identified as node 1 since no node precedes it.



Data no.	Data Byte	
0	0000 0000	*Start Byte*
1	0000 1101	*From Node 1*
2	0001 0101	*From Node 2*
3	0001 1101	*From Node 3*
4	0010 0101	*From Node 4*
5	0010 1011	*From Node 5*
6	0011 0101	*From Node 6*
7	0011 1101	*From Node 7*
8	0100 0101	*From Node 8*

Data no.	Data Byte	
9	0100 1101	*From Node 9*
10	0101 0101	*From Node 10*
11	0101 1101	*From Node 11*
12	0110 0101	*From Node 12*
13	0110 1011	*From Node 13*
14	0111 0101	*From Node 14*
15	0111 1110	*From Node 15*
16	1000 0101	*From Node 16*
17	1111 1111	*End Byte*

Figure 5.16: An example packet received by the server when Pacman is in node 5 and Ghost is in node 15

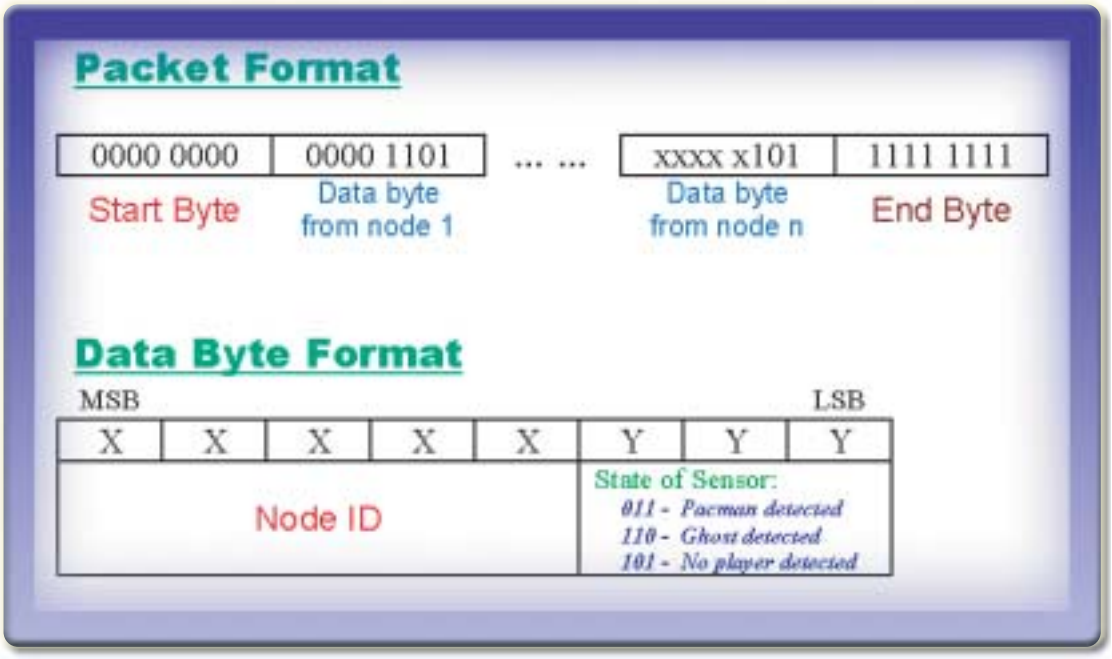


Figure 5.17: Packet and data byte format used in IR tracking

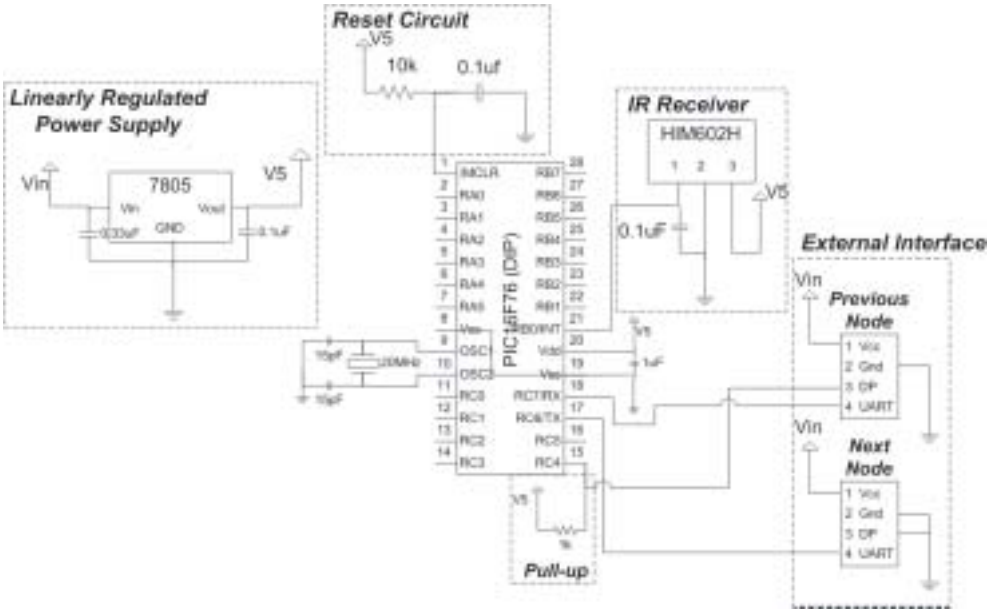


Figure 5.18: Circuit diagram of IR receiver

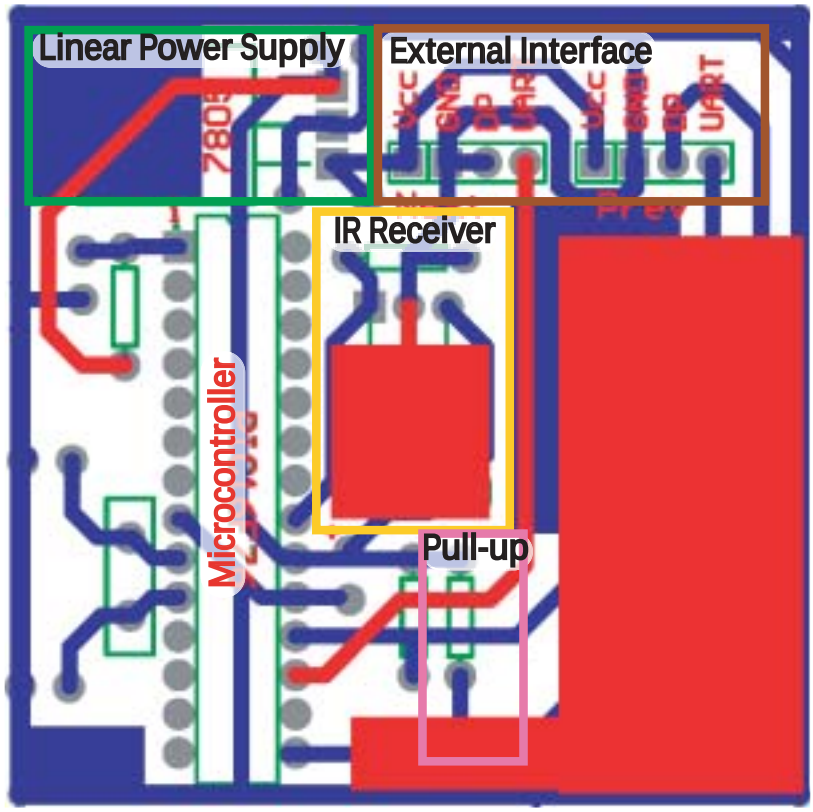


Figure 5.19: PCB layout of IR receiver with main modules labelled

Node 1 is configured to send a packet every 500ms. Subsequent nodes upon receiving the packet will add its data byte and relay it immediately. The server will extract relevant data from the final packet. It is noted that in a daisy chain topology a node that is not working will result in the breaking of data flow. To help identify which node is not functioning, a time-out is added to each node if no data is received from its previous node after 1s. Upon activation of this timeout, the node will send a dummy packet to the next node. If no packet is received after

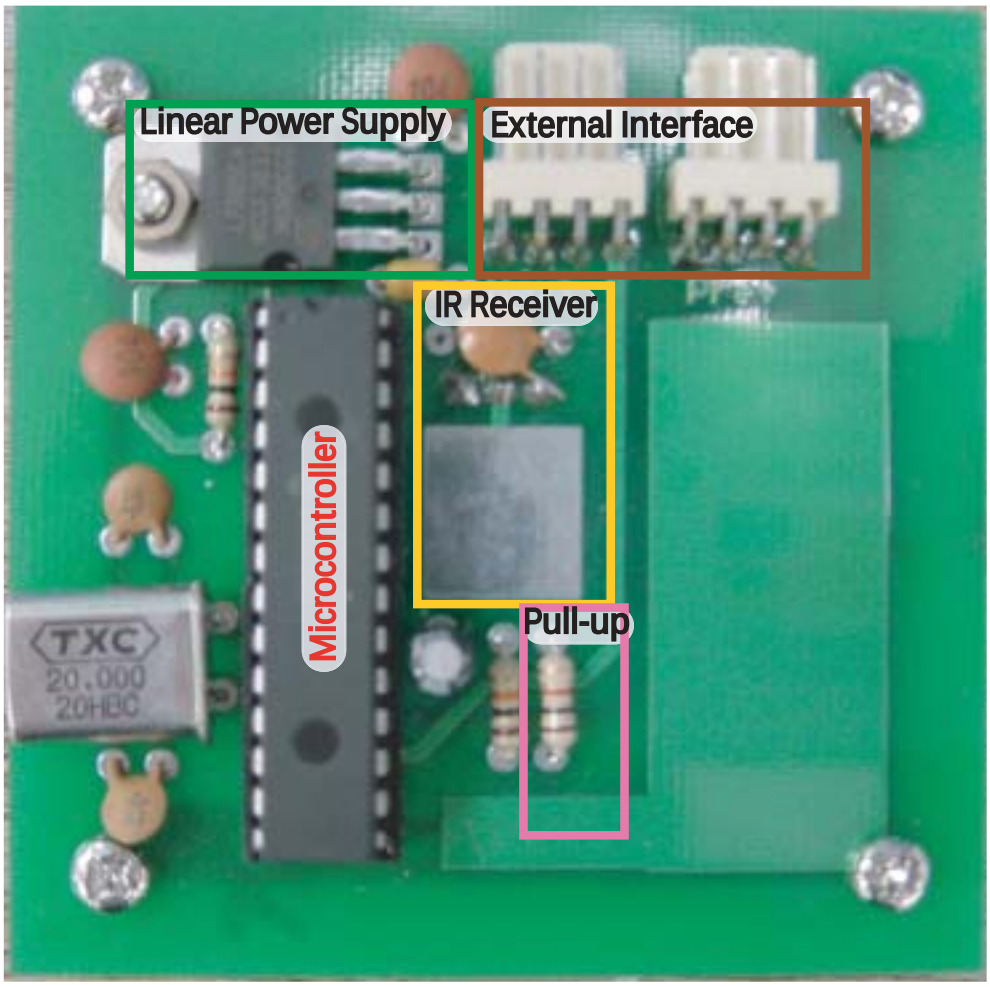


Figure 5.20: Actual circuit board of IR receiver with main modules labelled

5 timeouts a node will assume the role of node 1. The dummy packet serves to prevent subsequent nodes from assuming the position of node 1. At the server side the receiving of $n < 16$ data bytes indicates that node (16 - n) is not functioning.

The program flow of the IR receiver firmware is given in Fig. 5.21. The source code is given in Appendix A (see Appendix A: IR Receiver)). The program flow is as follows.

1. The program starts by initializing the parameters and settings to the ports

and interrupts.

2. RC4 is checked if it is high.

   (a) If it is not then there is a previous node to this node. The program jumps to (3).

   (b) If it is then program is node 1. The timer is set to trigger at 500ms interval. The state of the IR sensor is determined. The node will initiate the sending of a packet every 500ms. The program loops here.

3. Timer_Triggered set to 0.

4. Timer interrupt is set to trigger at 1 sec intervals.

5. Determine state of sensor.

6. Check if UART has received any data.

   (a) If yes, check the packet. If it is a dummy packet, then relay the packet. Else add the data byte of the node to the packet and relay the packet. Program jumps to (3).

   (b) If not, check if the timer interrupt has triggered. If not program jumps to (5), else Timer_Triggered is incremented by 1 and a dummy packet is sent through the UART.

   (c) Check if Timer_Triggered is 5. If it is then node assumes the position of node 1 and program jumps to (2b), else program jumps to (4).

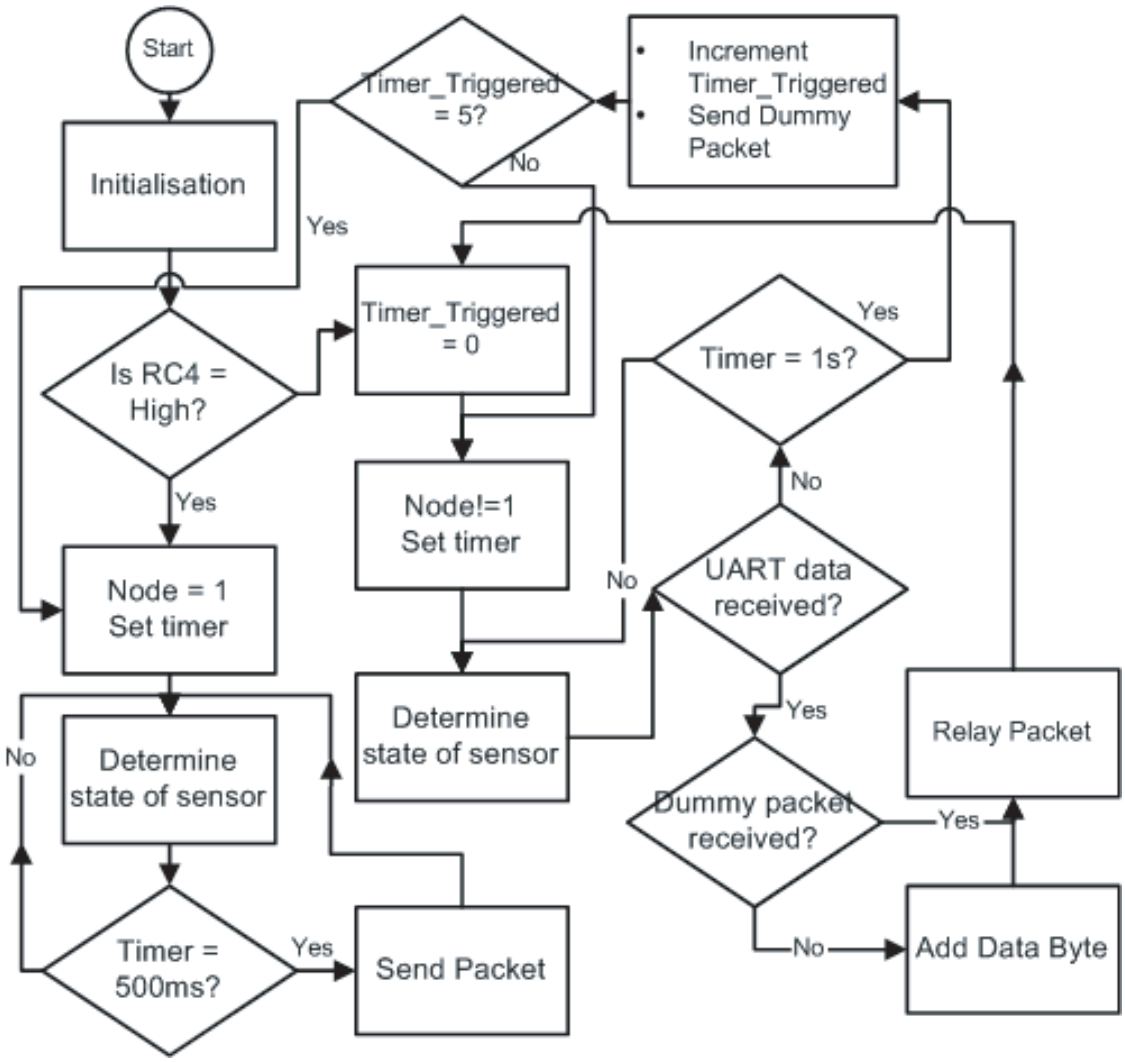


Figure 5.21: Flow of the IR receiver firmware

Receiver-to-Server Interface

The UART pin from node 16 is connected to the serial port of the server through a MAX232 chip. As communication is half duplex only pin 2 (RX) of the server's serial port needs to be connected. Fig. C.6 in Appendix A shows the PCB design.

Modification to Program Flow

With the use of IR tracking in the maze, the tracking device is connected directly to the server, rather than through the wearable computer of each player as in the case of DRM-III. Slight modifications has to be done in the program flow where each wearable computer is updated on its position by the server.

5.1.2 Shortcomings

As only one IR receiver is attached to each square in the matrix, the resolution of tracking is limited to only one square. More IR receivers could be added to each square to improve the precision offered by the current setup. Alternatively other forms of tracking such as video tracking could be used.

It was found that the spread of the IR emitter and receiver is wide enough for multiple receivers to sense an emitter. To limit the spread a short black tube shields the IR emitter from an overly large spread. A balance, through trial and error, has to be made to ensure that players walking over an IR receiver would be detected despite the smaller spread used. The effect of limiting the spread of the IR emitter is good. Actual testing shows that players are tracked by the IR receiver in the block they are standing on and wrong readings by neighboring blocks are quite rare.

Chapter 6

Conclusion

This thesis introduces Human Pacman, a sensing based mobile entertainment system that gives an embodied gaming experience in a real-virtual world. Using the ideas of ubiquitous, physical, and tangible computing the player is brought into a believable make-believe fantasy world weaved into the real world. Tangible interface is being explored with tangible physical objects incorporated into the game play to provide the novel experience of seamless juxtaposition of real and virtual worlds. These objects, with real world affordances we are intimately familiar, are embedded with a real-time digital link and meaning with objects in the virtual world. Social interaction in ways of collaboration and direct mingling enriches the experience of the game. Intriguing and unparalleled attributes of direct physical human-to-human interaction is recognized and explored through the "devouring" action where the digital meaning of apprehending the enemy is embellished by vivacious human responses.

In Chapter 1 the six novel aspects of Human Pacman are described. These aspects include physical gaming, social gaming, mobile gaming, ubiquitous computing, tangible interaction, and outdoor wide-area gaming. Table 1.1 summarises these novel features. As the chapter progresses, similar works are introduced and the novelties of Human Pacman over these works are described. "Pac Manhattan" and "Can You See Me Now" are works that are very similar to Human Pacman. Users move around in a wide outdoor setting as they participate in the respective games by interacting with their environment and other participants. The chapter ends with a brief history of Human Pacman, and a list of publications featuring the project.

Chapter 2 started off with a short introduction to the history of Pacman. The main concepts of the game and the roles of each player are detailed in this chapter. The actual game play is elaborated to bring out the requirements of the system to achieve the said gaming experience for the players. Key motivation for the players, described at the end of the chapter, links to the novelties mentioned in the previous chapter.

Chapter 3 goes into the details of how the requirements of the system, as listed out in Chapter 2, are realised. The hardware aspect of the whole Human Pacman system is first looked at. This includes the wearable computer, Bluetooth embedded object, Helper computer, and server computer. Most of the components are commercially acquired and integrated to the system. Components off-the-shelf are preferred as they are well tested by the manufacturer. Design and construction de-

tails of circuits custom built by the author are elaborated on. The software aspect of the Human Pacman system is also delved into. An overall data flow amongst the major components of the system is given and described. This is followed by a more detailed description of the program flow of the server, Helper, wearable computer, and Bluetooth-object subsystems. The design of the backpack is detailed near the end of the chapter. This aspect is crucial in enhancing the overall experience of the user, and for protecting the wearable computer hardware. Criterias for the backpack design are set based on previous experiences with earlier designs. At the end of the chapter, the author gives snippets of some issues that came up during the course of developing the system.

A user study for the Human Pacman system was conducted on the university campus. The study and its results are elaborated in Chapter 4. The aim of this study is to find out actual experiences of non-expert users with the different aspects of the system, as well as their level of enjoyment in playing the game. In general users are enthusiastic about Human Pacman. The study suggests that the element of physical involvement is a key attraction of the game. Users enjoy the immersive experience of role-playing. On the aspect of tangible interaction, most of the users found that the tangible collection of objects enhances their gaming experience. As for the social interaction aspect of the game, the Helper role is perceived to have lesser social interaction as compared to the Pacman and Ghost roles. Users brought up several usability problems during the course of the study. Many reflect that the wearable computer is bulky and heavy, and the HMD is cumbersome

to wear. Despite that the collection of virtual cookies by walking through them is deemed to be an intuitive action, users commented that shifts in alignment of virtual cookies from it supposed absolute position in real space causes its lack in realistic feel. Some users reflected that they want to see more visually appealing 3D graphics and more variety of virtual objects in the VR world. It is concluded in the study that users like the idea of Human Pacman as a whole. However there are still room for improvement. More in-depth studies should be carried out on the role of Helper and how it could be made less complex and more appealing.

Chapter 5 gives a description of a modified version of Human Pacman game developed specially for indoor areas. This small scale indoor maze version uses infrared transceivers to detect the movement of players within the maze area. The design and construction of the hardware, as well as the design of the coding for the infrared position determination, is given in this chapter. Its shortcoming is examined at the end of the chapter.

Human Pacman explores several novel experiences in the area of human-computer interface. The thesis examined the innovative work done by other researchers in this arena and differentiated Human Pacman from the other works. An elaboration of the game play and experience for the users is given, and this is followed by a description of the design and construction of the system to fulfill the mentioned functionalities of the game. A user study on the Human Pacman system was done and the results analysed. It was found that Human Pacman achieved its objectives in providing the novel experiences to the users. However there is still room for

improvement. As an extension of Human Pacman indoors, an indoor maze version of the game was developed.

In conclusion, Human Pacman is a novel system in the new hybrid field of physical, social, and mobile gaming that is built on ubiquitous computing and networking technology. The players are able to experience seamless links between the real and virtual world and therefore a higher than ever level of sensory gratification are obtained.

6.1 Future Works

Further enhancement on the Human Pacman system can be made in terms of miniaturization and improvement in robustness of the wearable computers. Currently the visual display used for the Pacman and Ghost is two-dimensional. This lacks the capability in providing the user with a stereo visual affordance of the gaming environment. The implementation of stereo visual display, probably using two cameras and a stereo HMD, could enhance the immersive experience of the user.

Accurate, sensitive and precise positioning methodologies and systems could be explored to enhance the realism of the game. A differential GPS tracking system could be implemented to improve the precision and accuracy of position tracking outdoor. The use of wireless LAN network for position tracking indoors should be explored to make the game transit seamlessly between indoor and outdoor. Further

work could also involve enhancing the accuracy of position tracking through the use of visual tracking on fixed-positioned markers. Given the small network data exchange of the Human Pacman system, existing mobile GPRS network infrastructure could be tapped into to create a truly on-the-move gaming system happening anytime, anywhere.

The use of haptic data gloves could add on a new intuitive reactive dimension to the collection of virtual cookies, bringing a tangible feel to an intangible object. More work could be done on the audio aspect of human senses. Perhaps a 3-dimensional audio cue could indicate to the player the approaching of an enemy in a certain direction.

The proliferation of innovative digital communication and entertainment influences the way we think, work and play. With physical and mobile gaming gaining popularity, traditional paradigms of entertainment will irrevocably shake from the stale television-set inertia. We believe that Human Pacman heralds the conjuration and growth of a new genre of computer game that is built on mobility, physical actions and the real world as a playground. Reality, in this case, is becoming more exotic than fantasy because of the mixed reality element in the game play. On the other hand, emphasis on physical actions might even bring forth the evolvement of professional physical gaming as competitive sport of the future, for example "Pacman International League".

The element of social gaming in Human Pacman symbolizes the nascence of humanity in future digital entertainment. People are looking forward to widening

their circle of friends and colleagues through social collaboration in game play. A new form of interactive entertainment is evolved.

Another important area of impact is the field of education. The technology presented in Human Pacman can be exported to applications in educational training that stresses on "learn by experience". Students are immersed in real site of action, and are given instructions visually through head mounted displays or verbally through speaker\earphone. This technology serves as a powerful instrument of cognition since it can enhance both experimenting and reflective thoughts through mixed reality and interactive experience.

Appendix A

Wearable Computer: A Tutorial

A.1 Connections Within the Wearable Computer

The wearable computer is made up of several components. It is essential that these individual modules are connected correctly for the proper functioning of the system.

Fig. A.1 shows the video connection from the A980. To connect to both the HMD and wireless video transmitter, a PC to TV converter was used to convert the RGB signal form A980 to NTSC signal sent via S-video and video connection respectively. A wireless video transceiver is used so that spectators could look at the HMD view seen by the user.

The I/O connections from the A980 is shown in Fig. A.2. As there are only 4 USB ports on the A980, a USB hub is required for the 5 USB connections. The configuration, as shown in the figure, is such that little amount of power is sourced

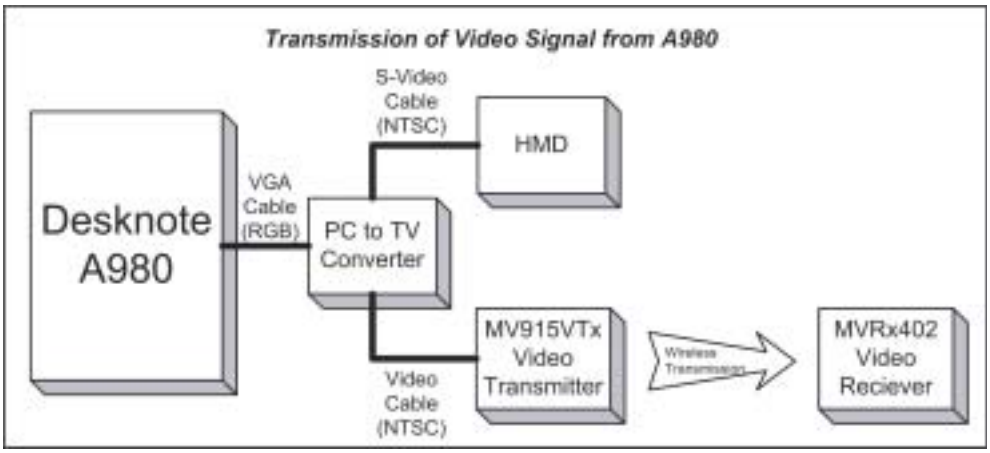


Figure A.1: Transmission of video signal from A980

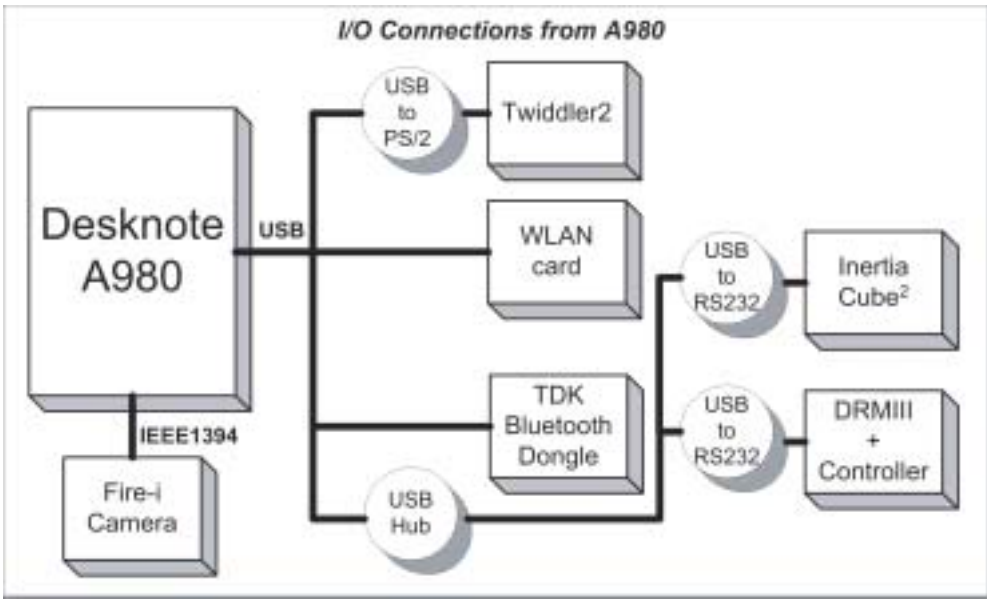


Figure A.2: I/O connections from A980

from the ports on the hub, making it unnecessary to add an external power supply to it.

Apart from the A980 and DRM-III, power for the other modules come from the 12V lead acid battery as seen in Fig. A.3. The voltage each module is connected to

is shown in the figure. Fig. A.4, Fig. A.6 and Fig. A.8 show the connectors on the actual controller circuit board, transmitter circuit board, and receiver circuit board respectively. The corresponding circuit diagrams are given in Fig. A.5, Fig. A.7 and Fig. A.9 respectively. The pins out of the connections are indicated in the respective figure on the actual PCB boards.

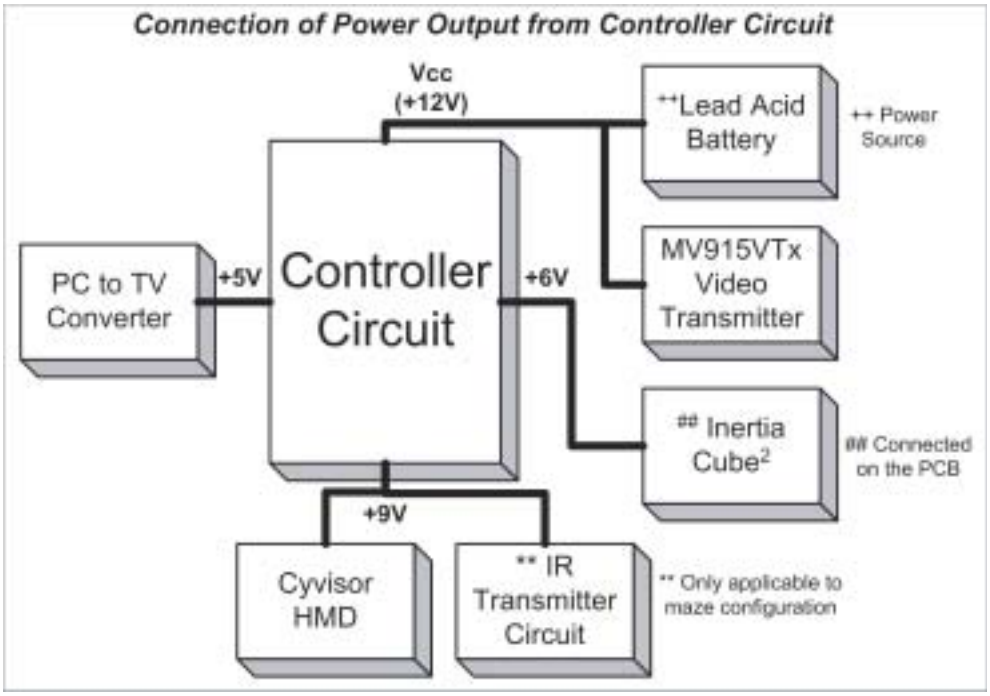


Figure A.3: Connection of power outputs from controller

A.2 Tuning the System

Prior to running the Human Pacman software, the system has to be configured. The COM Ports assigned to the individual USB to Serial adapters should be configured to COM2 and COM7 for InertiaCube2 and Controller/DRM-III respectively.

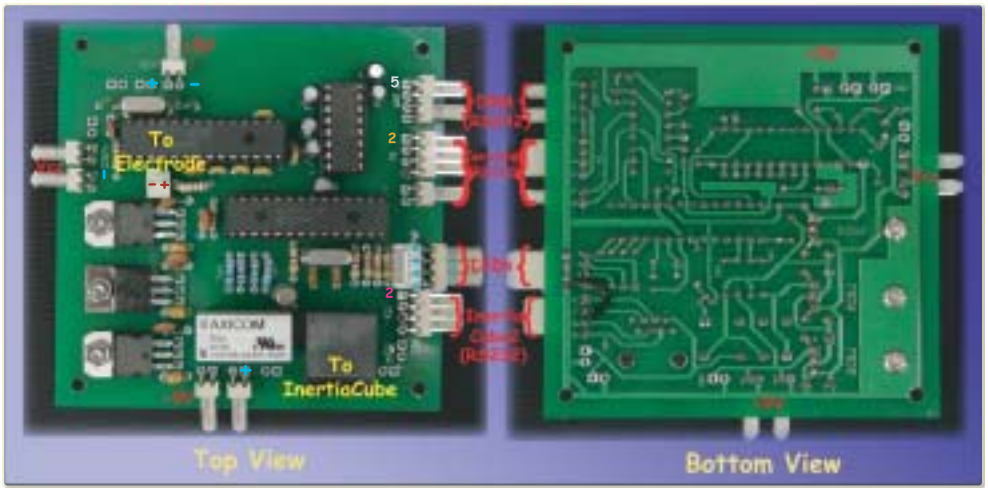


Figure A.4: Connections on the actual controller circuit board

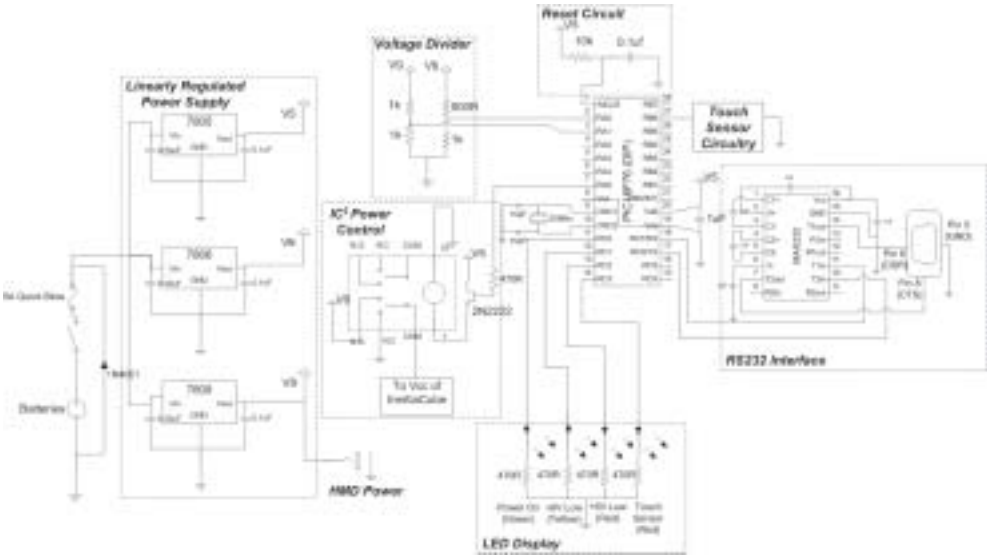


Figure A.5: Circuit diagram of the controller circuit in wearable computer

Manual configuration is done using Keyspan Serial Assistant. Fig. A.10 shows the dialog box of Keyspan Serial Assistant. The COM port is changed by selecting the "Port Mapping" tab and the appropriate COM port number as circled in the figure.

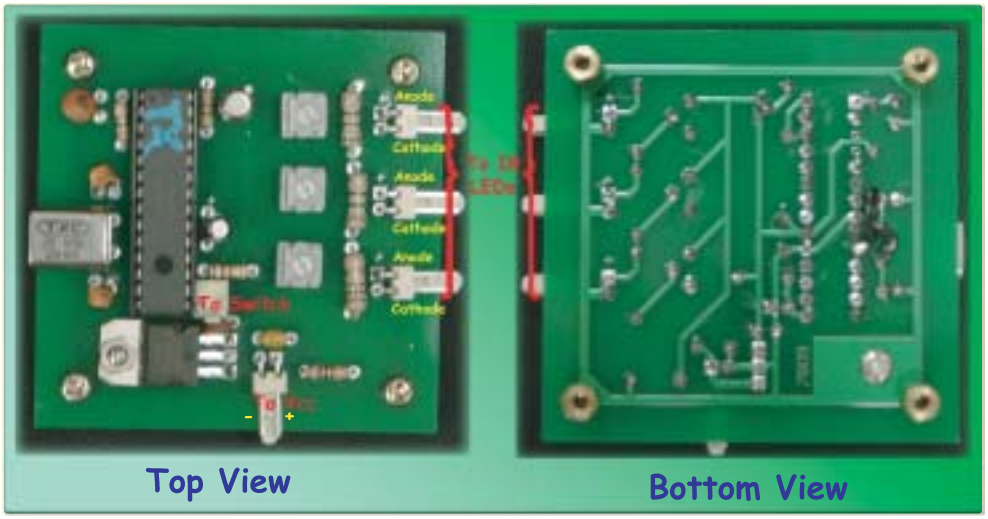


Figure A.6: Connections on the actual IR transmitter circuit board

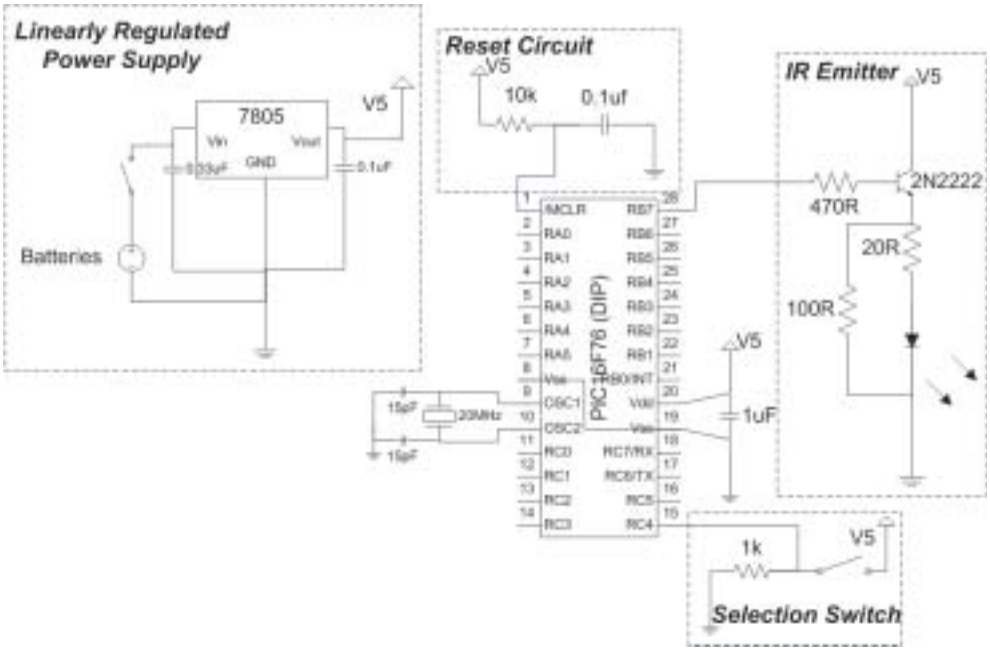


Figure A.7: Circuit diagram of the IR transmitter circuit

In addition, the wireless LAN network of each system has to be configured to work with each other. For easy configuration, the following settings are used.

- SSID – HuPacNet

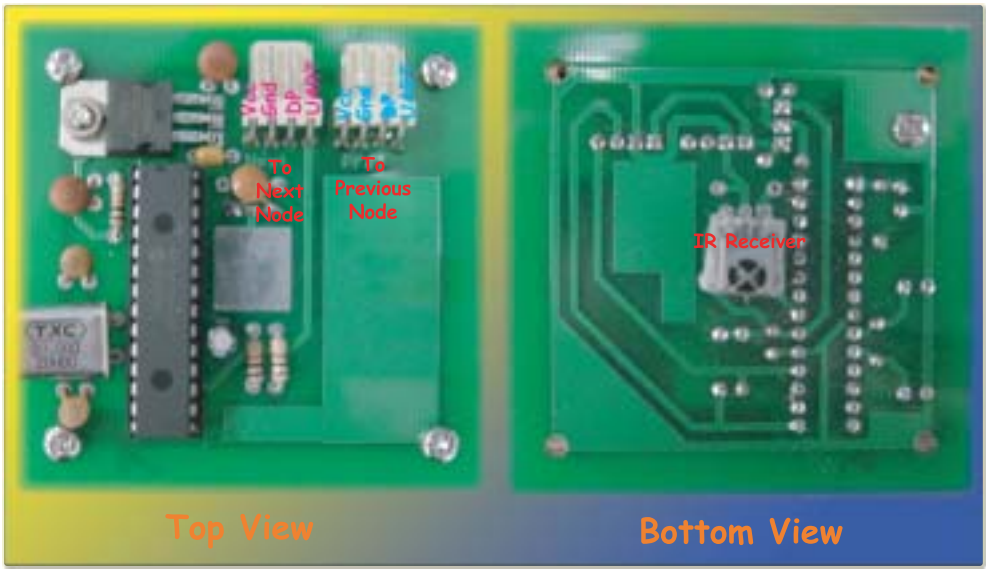


Figure A.8: Connections on the actual IR receiver circuit board

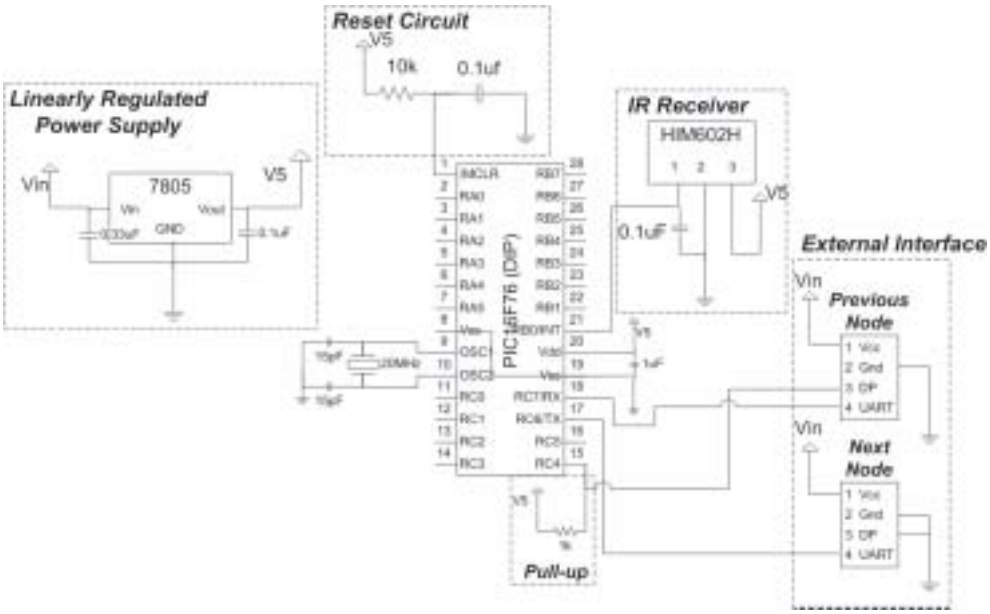


Figure A.9: Circuit diagram of the IR receiver circuit

- Channel – 10
- WEP – Off

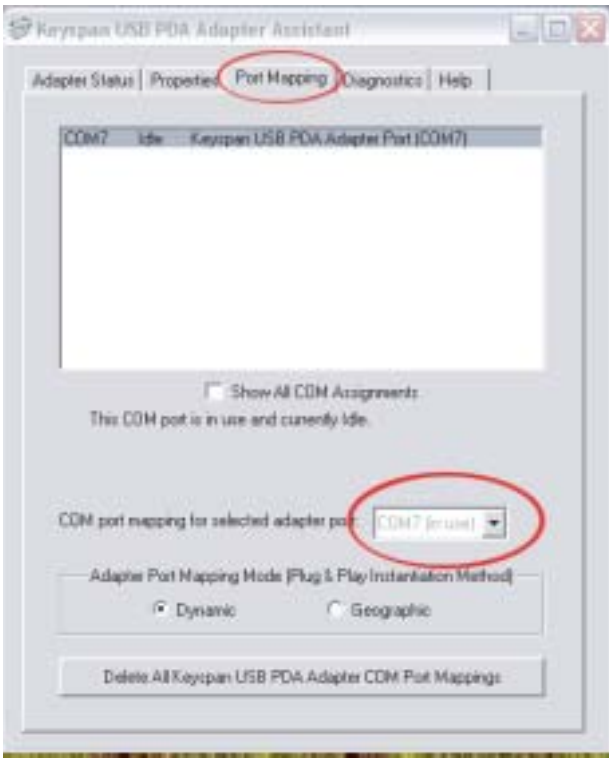


Figure A.10: Keyspan Serial Assistant

- DHCP – Enabled

A.3 Starting the System

The sequence of starting the wearable computer system should be as follows.

1. Ensure that all connections are properly connected.

2. Boot up the Desknote A980.

3. Turn on the power for lead acid battery. As shown in Fig. A.11, the green LED should light up to indicate fully charged battery. Yellow LED indicates that the power in the battery is too low to power up the HMD. Red LED

indicates that the power in the battery is too low to power up the HMD and InertiaCube2.

4. Turn on the HMD and select the S-VHS mode.

5. Select either the Pacman or Ghost IR selector switch if the system is to be used in the maze.

6. Turn on the DRM-III.

7. Run the program with the correct server IP address parameters as shown in Fig. A.12. You should see the DOS window screen as seen in Fig. A.13. The program will boot up to the AR view of the player.

Figure A.11: Side panel of the backpack

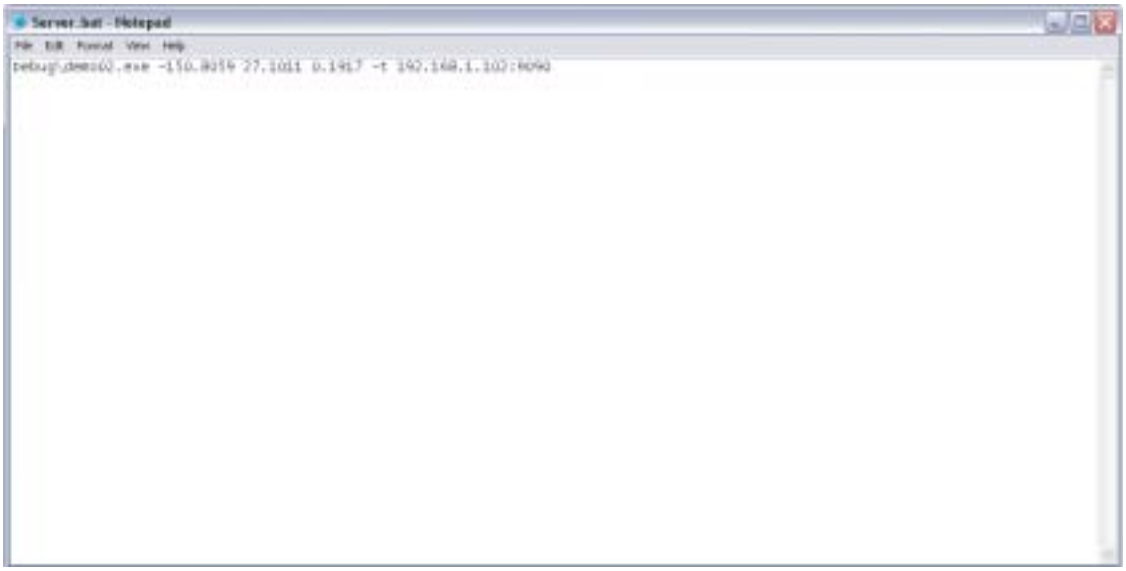


Figure A.12: Batch file used for running the program

Figure A.13: Dos window when program is running

Appendix B

Source Codes

B.1 Controller Firmware (Wearable Computer)

```
; ********************************
; Human Pacman Controller Program
;
; Filename:  humpac.asm
; Date:  14/03/04
; File Version:  4.2
;
; Author:  Goh Kok Hwee
;
; Files required:  p16f76.inc
; ********************************


#include <p16f76.inc>                                ; processor specific variable definitions

; *** Constants ***
CUTOFF6A                    EQU B'11000010'          ; cutoff voltage at V6 - 5.70V
CUTOFF9A                    EQU B'11011011'          ; cutoff voltage at V9 - 8.60V
CUTOFF9B                    EQU B'11100000'          ; hysteresis loops for V9 - 8.75V at V9
CUTOFF9C                    EQU B'11011000'          ; hysteresis loops for V6 - 7V at V9

; LED Pin allocation
LED_POW                     EQU D'0'
LED_POW9                    EQU D'1'
LED_POW6                    EQU D'2'
LED_TOUCHSENSOR             EQU D'3'

; Low Voltages
LV9_OUT                     EQU D'7'                 ; RC7 -> DSR
LV6_OUT                     EQU D'5'                 ; RC5 -> unconnected

; Touch Sensor
TS_PIN                      EQU D'6'                 ; RB6
TS_BIT                      EQU B'01000000'
TS_OUT                      EQU D'6'                 ; RC6 -> CTS

; Misc
RB_MASK                     EQU B'01000000'
```

```
; *** Variables ***
; Bank 0 register
; for ADC
AN_CHANNEL                  EQU 0x07F                 ; to specify which channel being read
ACQ_COUNTER                 EQU 0x07E
A2D_READING                 EQU 0x07D
A2D_9V                      EQU 0x07C                 ; A2D reading for V9
A2D_6V                      EQU 0x07B                 ; A2D reading for V6
HIGH_POWER                  EQU 0x07A                 ; power level in

; for Touch Sensor
RB_STORED                   EQU 0x07B
RB_CURRENT                  EQU 0x07A

; Misc
W_TEMP                      EQU 0x079
STATUS_TEMP                 EQU 0x078


; ******************************************

ORG          0x000
GOTO         main

; ********** Interrupt Vector **********
ORG          0x004
RETFIE

; ********** Main Program **********
main:

CALL         INIT

main1:
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVLW        B'10000001'                ; turn on ADON for Channel AN0
MOVWF        AN_CHANNEL

CALL         A2D
CALL         POLL_A2D

BCF          STATUS, RP0                ; Bank0
MOVF         A2D_READING, 0
MOVWF        A2D_6V

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVLW        B'10001001'                ; turn on ADON for Channel AN1
MOVWF        AN_CHANNEL

CALL         A2D
CALL         POLL_A2D

BCF          STATUS, RP0                ; Bank0
MOVF         A2D_READING, 0
MOVWF        A2D_9V

CALL         POW_MONITOR
CALL         TS_CHECK

GOTO         main1
; **********
```

```
; Main Init
; **********
INIT

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0
MOVLW        0x000
MOVWF        INTCON                     ; disable interrupt

CLRF         RCSTA                      ; Disable serial comms

; Configure PORTx
CLRF         PORTA
CLRF         PORTC

CLRF         HIGH_POWER

BSF          STATUS, RP0                ; Bank1
MOVLW        0x00
MOVWF        TRISC                      ; all outputs

BCF          STATUS, RP0                ; Bank0
BSF          PORTC, 0                   ; Switch on LED at RC0

; set ADCON1
BSF          STATUS, RP0                ; Bank1
MOVLW        B'00000100'                ; only AN0 and AN1 used for analog with no Vref
MOVWF        ADCON1

MOVLW        B'00000011'                ; AN0, AN1 are inputs
MOVWF        TRISA

; * setup for touch sensors *
BSF          STATUS, RP0                ; Bank1
MOVLW        RB_MASK                    ; Set RB6 as input
MOVWF        TRISB

RETURN

; ***************
; Configure A/D
; ***************
A2D
BCF          STATUS, RP1

; Configure ADCON0
BCF          STATUS, RP0                ; Bank0
MOVF         AN_CHANNEL, 0              ; move to W
MOVWF        ADCON0

CALL         WAIT_ACQUISITION

BCF          STATUS, RP0                ; Bank0
BSF          ADCON0, GO_DONE            ; Start the conversion

RETURN

; ******************************
; Wait for Acquistion to be done
; ******************************
WAIT_ACQUISITION

; 11.974 uS = 60 machine cycles
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0
```

```
MOVLW        D'18'
MOVWF        ACQ_COUNTER                ; 5 machine cycles up to here

acq_decre:
DECFSZ       ACQ_COUNTER                ; 1
GOTO         acq_decre                  ; 2

RETURN                                  ; 2 machine cycles

; ****************
; Polling for A2D
; ****************
POLL_A2D

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

poll:
BTFSC        ADCON0, GO_DONE            ; keep polling until cleared
GOTO         poll

MOVF         ADRES, 0
MOVWF        A2D_READING                ; save the reading

RETURN

; *****************
; Power Monitoring
; *****************
POW_MONITOR
; This function loops and monitors the voltage at AN0
; and AN1 to make sure sufficient voltage for 6V and 9V

; To find which stage (high power, mid power, low power) is the battery in

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

; ** 1 **
MOVLW        CUTOFF9B                   ; compare with the upper limit
SUBWF        A2D_9V, 0

; A2D_READING > CUTOFF9B for normal operation i.e no borrow
BTFSC        STATUS, C                  ; take next step if no borrow
GOTO         MaxPower

; ** 2 **
; power is below CUTOFF9B
MOVLW        CUTOFF9A                   ; compare with the upper limit
SUBWF        A2D_9V, 0

; A2D_READING > CUTOFF9B for normal operation i.e no borrow
BTFSC        STATUS, C                  ; take next step if no borrow
GOTO         HighPower

; ** 3 **
; power is below CUTOFF9A
MOVLW        CUTOFF6A                   ; compare with the upper limit
SUBWF        A2D_6V, 0

; A2D_READING > CUTOFF6A for normal operation i.e no borrow
BTFSC        STATUS, C                  ; take next step if no borrow
GOTO         PreventOsc
GOTO         LowPower

PreventOsc:
```

```
; To make sure that there is no oscillatory switching of relay

MOVF        HIGH_POWER, 0
XORLW       D'2'
BTFSS       STATUS, Z                   ; skip if in stage 2 (to prevent oscillation)
GOTO        MidPower

; power is above CUTOFF6A
MOVLW       CUTOFF9C
SUBWF       A2D_9V, 0

; A2D_READING > CUTOFF9C for normal operation i.e no borrow
BTFSC       STATUS, C                   ; take next step if no borrow
GOTO        MidPower                    ; mid power if previously low power
RETURN

LowPower:
; Voltage lower than CUTOFF6A
MOVLW       D'2'
MOVWF       HIGH_POWER
BCF         PORTA, 5                    ; switch off relay
BCF         PORTC, LED_POW
BCF         PORTC, LED_POW9
BSF         PORTC, LED_POW6

BSF         PORTC, LV9_OUT
BSF         PORTC, LV6_OUT

RETURN

MaxPower:
; Voltage higher than CUTOFF9B
; High Power Stage
CLRF        HIGH_POWER
BSF         PORTA, 5                    ; switch on relay
BSF         PORTC, LED_POW
BCF         PORTC, LED_POW9
BCF         PORTC, LED_POW6

BCF         PORTC, LV9_OUT
BCF         PORTC, LV6_OUT

RETURN

HighPower:
; Voltage higher than CUTOFF9A
; to check if previously voltage lower than CUTOFF9A
MOVF        HIGH_POWER, 1               ; check if high power previously
                                        ; (HIGH_POWER == 0 if previously high power)
BTFSC       STATUS, Z
GOTO        MaxPower                    ; goto MaxPower if previously was HighPower,
                                        ; else treat as MidPower

MidPower:
; Voltage higher than CUTOFF6A
MOVLW       D'1'
MOVWF       HIGH_POWER
BSF         PORTA, 5                    ; switch on relay
BCF         PORTC, LED_POW
BSF         PORTC, LED_POW9
BCF         PORTC, LED_POW6

BSF         PORTC, LV9_OUT
BCF         PORTC, LV6_OUT

RETURN
```

```
; *******************
; Check Touch Sensor
; *******************
TS_CHECK

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVF         PORTB, 0
MOVWF        RB_STORED
BTFSC        RB_STORED, TS_PIN          ; if TS_PIN is high, then touch sensor ON
GOTO         TS1

BCF          PORTC, LED_TOUCHSENSOR
BCF          PORTC, TS_OUT

RETURN

TS1:
BSF          PORTC, LED_TOUCHSENSOR
BSF          PORTC, TS_OUT
RETURN

END
```

B.2 Calibration of ADC Readings

```
; **********************************
; Bluetooth Controller Program
;
; Filename:  calibr.asm
; Date:  11/03/04
; File Version:  1.1
;
; Author:  Goh Kok Hwee
;
; Files required:  p16f76.inc
; **********************************


#include <p16f76.inc>                                ; processor specific variable definitions

AN_CHANNEL                  EQU 0x07F                ; Bank 0 register for specification
                                                     ; of which channel being read
ACQ_COUNTER                 EQU 0x07E
A2D_READING                 EQU 0x07D
TX_BUFFER                   EQU 0x07C
COUNTER                     EQU 0x07B


ORG          0x000
GOTO         main

;interrupt
ORG          0x004

;main program
main:

CALL         INIT

main1:
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVLW        B'10000001'                ; turn on ADON for Channel AN0
MOVWF        AN_CHANNEL

CALL         A2D
CALL         POLL_A2D
CALL         SEND_READING

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVLW        B'10001001'                ; turn on ADON for Channel AN1
MOVWF        AN_CHANNEL

CALL         A2D
CALL         POLL_A2D
CALL         SEND_READING

CALL         SEND_CR
GOTO         main1

;***************
; Main Init
;***************
INIT

BCF          STATUS, RP1
```

```
;setup serial comms to async, 9600, no parity, 8bits
BSF          STATUS, RP0                ;Bank1
MOVLW        D'129'
MOVWF        SPBRG

MOVLW        B'00100100'                ;8-bit, async, high baud
MOVWF        TXSTA

BCF          STATUS, RP0                ; Bank0
MOVLW        B'10000000'                ; only send no recv
MOVWF        RCSTA

;Configure PORTA
BCF          STATUS, RP0                ; Bank0
CLRF         PORTA

;setup port C for LED
BCF          STATUS, RP0                ;Bank0
CLRF         PORTC

BSF          STATUS, RP0                ;Bank1
MOVLW        0x00
MOVWF        TRISC                      ;all outputs

BCF          STATUS, RP0                ;Bank0
BSF          PORTC, 0                   ; Switch on LED at RC0

;set ADCON1
BSF          STATUS, RP0                ; Bank1
MOVLW        B'00000100'                ; only AN0 and AN1 used for analog with no Vref
MOVWF        ADCON1

MOVLW        B'00000011'                ; AN0, AN1 are inputs
MOVWF        TRISA

RETURN

;***************
; Configure A/D
;***************
A2D
BCF          STATUS, RP1

;Configure ADCON0
BCF          STATUS, RP0                ; Bank0
MOVF         AN_CHANNEL, 0              ; move to W
MOVWF        ADCON0

CAL          WAIT_ACQUISITION

BCF          STATUS, RP0                ; Bank0
BSF          ADCON0, GO_DONE            ; Start the conversion

RETURN

;******************************
; Wait for Acquistion to be done
;******************************
WAIT_ACQUISITION

;11.974 uS = 60 machine cycles
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVLW        D'18'
```

```
MOVWF        ACQ_COUNTER                ; 5 machine cycles up to here

acq_decre:
DECFSZ       ACQ_COUNTER                ; 1
GOTO         acq_decre                  ; 2

RETURN                                  ; 2 machine cycles

;****************
; Polling for A2D
;****************
POLL_A2D

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

poll:
BTFSC        ADCON0, GO_DONE            ; keep polling until cleared
GOTO         poll

MOVF         ADRES, 0
MOVWF        A2D_READING                ; save the reading

RETURN

;****************
; Polling for A2D
;****************
SEND_READING

BCF          STATUS, RP1

BCF          STATUS, RP0                ; Bank0
MOVLW        D'8'
MOVWF        COUNTER

rotate:
RLF          A2D_READING, 1

BTFSS        STATUS, C
GOTO         send0
GOTO         send1

send0:
MOVLW        A'0'
MOVWF        TX_BUFFER
CALL         SEND_CHAR

BCF          STATUS, RP0                ; Bank0
DECFSZ       COUNTER
GOTO         rotate

GOTO         Send_Space

send1:
MOVLW        A'1'
MOVWF        TX_BUFFER
CALL         SEND_CHAR

BCF          STATUS, RP0                ; Bank0
DECFSZ       COUNTER
GOTO         rotate

GOTO         Send_Space

Send_Space:
```

```
BCF          STATUS, RP0                ; Bank0
MOVLW        A' '
MOVWF        TX_BUFFER
CALL         SEND_CHAR

RETURN

;****************
; Carriage Return
;****************
SEND_CR
BCF          STATUS, RP0                ; Bank0
MOVLW        0x0D                       ; CR
MOVWF        TX_BUFFER
CALL         SEND_CHAR

RETURN

;********************
; Serial Transmission
;********************
SEND_CHAR
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0
MOVF         TX_BUFFER, 0
MOVWF        TXREG

BSF          STATUS, RP0                ; Bank1

WaitChar:
BTFSS        TXSTA, TRMT                ; wait for transmit buffer to clear (TRMT set)
GOTO         WaitChar

RETURN

END
```

B.3 Controller Firmware (Bluetooth Embedded Object)

```
; ********************************
; Bluetooth Controller Program
;
; Filename:  BTObj.asm
; Date:  01/03/04
; File Version:  5.0
;
; Author:  Goh Kok Hwee
;
; Files required:  p16f76.inc
; ********************************


#include <p16f76.inc>                                ; processor specific variable definitions

CUTOFF7                     EQU B'00111101'          ;cutoff voltage to low power at 7V
AN_CHANNEL7                 EQU B'10000001'

LED_POW                     EQU D'0'
LED_TOUCHSENSOR             EQU D'3'

START_CHAR                  EQU A'S'
SHUT_CHAR                   EQU A's'
PGM_START                   EQU A'G'
PGM_STALL                   EQU A'g'

RT_PIN                      EQU D'7'
RT_BIT                      EQU B'10000000'
TS_PIN                      EQU D'6'
TS_BIT                      EQU B'01000000'
TS_OFF                      EQU A't'
TS_ON                       EQU A'T'

RB_MASK                     EQU B'11000000'

SERIAL_TX                   EQU 0x07C                ; do not delete

RB_STORED                   EQU 0x07B
RB_CURRENT                  EQU 0x07A

RCV_STATUS                  EQU 0x079
RCV_BUFFER                  EQU 0x078

SHUT_COUNT                  EQU 0x077
SHUT_COUNT1                 EQU 0x076



ORG          0x000
GOTO         main

;interrupt
ORG          0x004

BCF          STATUS, RP0
BTFSC        PIR1, TMR1IF
CALL         TIMER1_INT

BCF          STATUS, RP0
BTFSS        PIR1, RCIF
GOTO         RBPort
```

```
CALL          RCV_DECODE
RETFIE                                  ; don't process change in port B if something recv

RBPort:
BTFSC         INTCON, RBIF
CALL          RB_VECTOR

RETFIE


;main program
main:
CALL          INIT

loop:
GOTO          loop

; CALL        POWER_MONITOR


;****************
; Initialisation
;****************
INIT

BCF           STATUS, RP1

;* setup serial comms to async, 9600, no parity, 8bits *
BSF           STATUS, RP0               ;Bank1
MOVLW         D'129'
MOVWF         SPBRG

MOVLW         B'00100100'               ;8-bit, async, high baud
MOVWF         TXSTA

BCF           STATUS, RP0               ;Bank0
MOVLW         B'10010000'
MOVWF         RCSTA


;* setup port C for LED *
BCF           STATUS, RP0               ; Bank0
CLRF          PORTC

BSF           STATUS, RP0               ; Bank1
MOVLW         0x080
MOVWF         TRISC                     ; all outputs except RC7

BCF           STATUS, RP0               ; Bank0

BCF           PORTC, LED_POW            ; Unlit "Power-On" LED
BSF           PORTC, LED_TOUCHSENSOR            ; Lit Touch Sensor LED

MOVLW         0x01
MOVWF         RCV_STATUS                ; Non-zero to indicate stall_mode

BSF           STATUS, RP0               ; Bank1
BSF           PIE1, RCIE

BSF           STATUS, RP0               ; Bank1
MOVLW         RB_MASK                   ; Set RB6 as input
MOVWF         TRISB

BCF           STATUS, RP0               ; Bank0
MOVF          PORTB, 0
```

```
ANDLW        RB_MASK                    ; only store the relevant bits
MOVWF        RB_STORED

MOVLW        B'11001000'                ; enable RB Port Change Interrupt
MOVWF        INTCON                     ; start interrupt

RETURN

;***************
; Recv Data decoding
;***************
RCV_DECODE

BCF          STATUS, RP0                ; Bank0

MOVF         RCREG, 0
MOVWF        RCV_BUFFER                 ; Save rcv in buffer

XORLW        PGM_START
BTFSC        STATUS, Z                  ;skip next sentence if don't match
GOTO         Start_PGM

MOVF         RCV_BUFFER, 0
XORLW        PGM_STALL
BTFSS        STATUS, Z                  ;skip next sentence if match
RETURN

;Stall mode
BCF          PORTC, LED_POW             ; Unlit "Power-On" LED
BSF          PORTC, LED_TOUCHSENSOR             ;Lit Touch Sensor LED

MOVLW        0x01
MOVWF        RCV_STATUS                 ; Non-Zero to indicate stall mode

MOVLW        0x00
MOVWF        RB_STORED                  ; Reset RB_Stored

MOVLW        PGM_STALL
MOVWF        SERIAL_TX
CALL         SEND_CHAR

RETURN

; Normal mode
Start_PGM:

BSF          PORTC, LED_POW             ; Lit "Power-On" LED
BCF          PORTC, LED_TOUCHSENSOR             ; Unlit Touch Sensor LED

MOVLW        0x00
MOVWF        RCV_STATUS                 ; Zero to indicate normal mode

MOVLW        PGM_START
MOVWF        SERIAL_TX
CALL         SEND_CHAR

RETURN

;***************************
; Send character in SERIAL_TX
;***************************

SEND_CHAR
BCF          STATUS, RP0                ; Bank0
MOVF         SERIAL_TX, 0
MOVWF        TXREG
```

```
BSF          STATUS, RP0                ; Bank1

WaitChar:
BTFSS        TXSTA, TRMT                ; wait for transmit buffer to clear (TRMT set)
GOTO         WaitChar

RETURN

;**********************
; Vectoring out from RB
;**********************
RB_VECTOR
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVF         PORTB, 0
ANDLW        RB_MASK                    ; only store the relevant bits
MOVWF        RB_CURRENT

ANDLW        RT_BIT                     ;check if it RT_PIN is set
BTFSC        STATUS, Z
GOTO         RB_VecCont

XORWF        RB_STORED, 0               ; to check if already sent Start Character
BTFSC        STATUS, Z                  ; zero if same as previously stored
RETURN

MOVLW        RT_BIT
MOVWF        RB_STORED                  ;to indicate already sent start character
                                        ;(this is necessary as TS may trigger this int)
                                        ; so if push button activated, will ignore touch sensor
; to start timer

BCF          STATUS, RP0                ; Bank0
MOVLW        D'191'                     ; About 5 secs
MOVWF        SHUT_COUNT

MOVLW        D'2'                       ; About 5 secs
MOVWF        SHUT_COUNT1

MOVLW        0x000
MOVWF        TMR1L
MOVWF        TMR1H

MOVLW        0x001
MOVWF        T1CON

BSF          STATUS, RP0                ; Bank0
BSF          PIE1, TMR1IE
BCF          STATUS, RP0                ; Bank0

;send start character
MOVLW        START_CHAR
MOVWF        SERIAL_TX
CALL         SEND_CHAR
RETURN


RB_VecCont:
MOVF         RB_CURRENT, 0
XORWF        RB_STORED, 0
BTFSC        STATUS, Z                  ; zero if same as previously stored
RETURN

ANDLW        RT_BIT                     ; check if the RT pin has just been released
```

```
BTFSC        STATUS, Z
GOTO         RB_VecCont2

MOVLW        00H
MOVWF        RB_STORED                  ; clear whatever has been stored
             ; end timer
BCF          T1CON, TMR1ON
BSF          STATUS, RP0                ; Bank0
BCF          PIE1, TMR1IE

RETURN

RB_VecCont2:
MOVF         RCV_STATUS
BTFSC        STATUS, Z                  ; Go next step if RCV_STATUS is zero
GOTO         ContinueVec

RETURN

ContinueVec:
MOVF         RB_CURRENT, 0
XORWF        RB_STORED, 0
BTFSC        STATUS, Z                  ; zero if same as previously stored
RETURN

ANDLW        TS_BIT                     ; check if it is TS_PIN
BTFSC        STATUS, Z
RETURN       ;@@@ to be changed for step counter in future

CALL         TS_CHECK                   ; change in Touch Sensor

BCF          STATUS, RP0                ; Bank0
MOVF         RB_CURRENT, 0
MOVWF        RB_STORED

RETURN

;*******************
; Check Touch Sensor
;*******************
TS_CHECK

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

             ; to check if touch sensor has been resetted

BTFSC        RB_STORED, TS_PIN          ; if TS_PIN is high, then touch sensor ON
GOTO         TS1

MOVLW        TS_ON
BSF          PORTC, LED_TOUCHSENSOR
MOVWF        SERIAL_TX
CALL         SEND_CHAR

RETURN
TS1:
MOVLW        TS_OFF
BCF          PORTC, LED_TOUCHSENSOR
MOVWF        SERIAL_TX
CALL         SEND_CHAR

RETURN

;*******************
; Timer Interrupt
```

```
;*******************
TIMER1_INT

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

BCF          PIR1, TMR1IF
DECFSZ       SHUT_COUNT, 1
RETURN

MOVLW        D'191'
MOVWF        SHUT_COUNT

DECFSZ       SHUT_COUNT1, 1
RETURN

MOVLW        SHUT_CHAR
MOVWF        SERIAL_TX
CALL         SEND_CHAR

;LED Flashing

FlashLED:
BCF          STATUS, RP0                ; Bank0

BSF          PORTC, LED_TOUCHSENSOR
BSF          PORTC, LED_POW

MOVLW        0x000
MOVWF        T1CON

MOVLW        D'38'                      ; About 500ms
MOVWF        SHUT_COUNT

MOVLW        0x000
MOVWF        TMR1L
MOVWF        TMR1H

MOVLW        0x001
MOVWF        T1CON

WaitOverFlow:
BTFSS        PIR1, TMR1IF
GOTO         WaitOverFlow

BCF          PIR1, TMR1IF

DECFSZ       SHUT_COUNT, 1
GOTO         WaitOverFlow

FlashLED1:
BCF          STATUS, RP0                ; Bank0

BCF          PORTC, LED_TOUCHSENSOR
BCF          PORTC, LED_POW

MOVLW        0x000
MOVWF        T1CON

MOVLW        D'38'                      ; About 500ms
MOVWF        SHUT_COUNT

MOVLW        0x000
MOVWF        TMR1L
MOVWF        TMR1H
```

```
MOVLW         0x001
MOVWF         T1CON

WaitOverFlow1:
BTFSS         PIR1, TMR1IF
GOTO          WaitOverFlow1

BCF           PIR1, TMR1IF

DECFSZ        SHUT_COUNT, 1
GOTO          WaitOverFlow1

GOTO          FlashLED

RETURN


END
```

B.4 IR Receiver

```
; ********************************
; IR Receiver Program
;
; Filename:  irrcv.asm
; Date:  14/03/04
; File Version:  1.5
;
; Author:  Goh Kok Hwee
;
; Files required:  p16f76.inc
; ********************************


#include <p16f76.inc>      ; processor specific variable definitions

; ********* Constants *********
DP_PIN                      EQU D'4'                  ; RC4
IR_PIN                      EQU D'0'                  ; RB0

MODE_NOR                    EQU 0x02                  ; Normal mode
MODEBIT_NOR                 EQU 0x01

MODE_NODE1                  EQU 0x04                  ; Node 1
MODEBIT_NODE1               EQU 0x02

PKT_DP                      EQU 0x0FF
PKT_STARTFLAG               EQU 0x000
PKT_ENDFLAG                 EQU 0x0FF
IR_PACMAN                   EQU B'01100000'
IR_GHOST                    EQU B'11000000'
IR_NIL                      EQU B'10100000'
IR_NILMASK                  EQU B'00000101'
IR_ERROR                    EQU B'00000000'

TMR1_NODE1                  EQU D'51'                 ; 500ms
TMR1_NOR                    EQU D'201'                ; 1000ms

DP_SEND                     EQU D'5'

; ********* Temp Data *********
MODE                        EQU 0x07F                 ; mode uC is in
RCV_BUFFER                  EQU 0x07E                 ; Receiver Buffer
TMR1_COUNT                  EQU 0x07D                 ; Multiple of 5ms
TX_BUFFER                   EQU 0x07C                 ; Transmit Buffer
DELAY                       EQU 0x07B                 ; Delay
RB_BUFFER                   EQU 0x07A
NODE_ID                     EQU 0x079
DATA_BYTE                   EQU 0x078
DP_TOSEND                   EQU 0x077
TEMP                        EQU 0x076
W_TEMP                      EQU 0x075
STATUS_TEMP                 EQU 0x074
IR_STATUS                   EQU 0x073
IR_LOW                      EQU 0x072

TEMPL                       EQU 20h                   ; Temporary storage location
TEMPH                       EQU 21h                   ; Character storage location
COUNTL                      EQU 24h

OUTPUT                      EQU 2AH
COUNTH                      EQU 2BH
PAC_FLAG                    EQU 2CH
GHO_FLAG                    EQU 2DH
TOTAL_TIME                  EQU 2EH
```

```
PAC_PERIOD                  EQU D'100'
GHO_PERIOD                  EQU D'50'
PAC_PERIOD_BOUND            EQU D'10'
GHO_PERIOD_BOUND            EQU D'10'



ORG          0x000
GOTO         main

             ; ******************************
             ; ******** Interrupt Vector *********
             ; ******************************
ORG          0x004

             ; ******************************
             ; ******** main program *********
             ; ******************************
main:

CALL         INIT_PORT
CALL         INIT_SERIAL
CALL         INIT_INT                  ; to check

BCF          STATUS, RP1
BCF          STATUS, RP0               ; Bank 0
BTFSC PORTC, DP_PIN
GOTO         main_node1                ; if set mean node 1

CALL         SET_NOR

loop:
GOTO         START                     ; written by SP
GOTO         loop

main_node1:
CALL         SET_NODE1
GOTO         loop

; ***************
; Init PORTs
; ***************
INIT_PORT
BCF          STATUS, RP1
BSF          STATUS, RP0               ; Bank 1

; Init Port B
MOVLW        0x000
MOVLW        TRISB                     ; make all output pins

BSF          TRISB, IR_PIN             ; set to input

; Init Port C
MOVLW        0x000
MOVLW        TRISC                     ; make all output pins

BSF          TRISC, 7                  ; set to serial receive to input
BSF          TRISC, DP_PIN

BCF          STATUS, RP0               ; BANK 0
MOVLW        B'00000000'               ; Clear all PORTx

MOVF         PORTC
MOVLW        PAC_FLAG
MOVLW        GHO_FLAG
```

```
RETURN

; *****************
; Init Serial Port
; *****************
INIT_SERIAL

BCF          STATUS, RP1

; *setup serial comms to async, 9600, no parity, high baud
BSF          STATUS, RP0                ; Bank 1
MOVLW        D'129'
MOVLW        SPBRG

MOVLW        B'00100100'
MOVLW        TXSTA

BCF          STATUS, RP0                ; Bank 0
MOVLW        B'10010000'
MOVLW        RCSTA

RETURN


; ***************
; Init Interrupt
; ***************

INIT_INT
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank 1

MOVLW        0x000
MOVLW        INTCON                     ; Disable global interrupt
MOVLW        PIR1
MOVLW        PIR2

RETURN

; ***************
; Set Node1 Mode
; ***************
SET_NODE1

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank 0
MOVLW        MODE_NODE1                 ; Set mode as Node1
MOVLW        MODE

MOVLW        IR_NIL
MOVLW        IR_STATUS                  ; load with default

MOVLW        0x01
MOVLW        NODE_ID

MOVLW        TMR1_NODE1
MOVLW        TMR1_COUNT
CALL         RESET_TMR1

RETURN

; ***************
; Set Normal Mode
; ***************
SET_NOR
```

```
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank 0
MOVLW        MODE_NOR                   ; Set mode as Normal
MOVLW        MODE

MOVLW        0x000
MOVLW        NODE_ID

MOVLW        IR_NIL
MOVLW        IR_STATUS                  ; load with default

MOVLW        DP_SEND
MOVLW        DP_TOSEND

MOVLW        TMR1_NOR
MOVLW        TMR1_COUNT
CALL         RESET_TMR1

RETURN


; **********************
; Rcv Interrupt Routine
; **********************
RCV_INT
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank 0

; read in serial port data and store
MOVF         RCREG, 0
MOVLW        RCV_BUFFER

BTFSC        MODE, MODEBIT_NODE1
                                        ; take next step if in NODE1 mode
GOTO         Rcv_Node1

; Normal mode
BCF          STATUS, RP0                ; Bank 0
MOVF         NODE_ID, 0
BTFSC        STATUS, Z
GOTO         Normal_Start

MOVF         RCV_BUFFER, 0
XORLW        PKT_ENDFLAG
BTFSC        STATUS, Z
GOTO         Normal_End

; forward data
INCF         NODE_ID, 1
MOVF         RCV_BUFFER, 0

; check for error from transmission
ANDLW        B'00000111'
MOVLW        TEMP
SWAPF        TEMP, 1                    ; now is 0XXX0000
BCF          STATUS, C                  ; clear carry flag
RLF          TEMP, 1                    ; rotate left

MOVLW        IR_NIL
XORWF        TEMP, 0
BTFSC        STATUS, Z
GOTO         NoErrorRecv

MOVLW        IR_PACMAN
XORWF        TEMP, 0
```

```
BTFSC        STATUS, Z
GOTO         NoErrorRecv

MOVLW        IR_GHOST
XORWF        TEMP, 0
BTFSC        STATUS, Z
GOTO         NoErrorRecv

; error.  default to IR_NIL
MOVLW        B'11111000'
ANDWF        RCV_BUFFER, 1              ; remove bit 0-2 in RCV_BUFFER
MOVLW        IR_NILMASK
IORWF        RCV_BUFFER, 1              ; add default values give in IR_NILMASK

NoErrorRecv:
MOVF         RCV_BUFFER, 0              ; load RCV_BUFFER into W

MOVLW        TX_BUFFER
CALL         SEND_CHAR

GOTO         Normal_Timeout

Normal_Start:

BCF          STATUS, RP0                ; Bank 0

MOVF         RCV_BUFFER, 0
XORLW        PKT_STARTFLAG
BTFSS STATUS, Z
GOTO         Normal_Timeout             ; Error start flag

INCF         NODE_ID, 1

MOVLW        PKT_STARTFLAG
MOVLW        TX_BUFFER
CALL         SEND_CHAR

GOTO         Normal_Timeout

Normal_End:
CALL         SEND_DATA                  ; Add in own data

BCF          STATUS, RP0                ; Bank 0

MOVLW        0x000
MOVLW        NODE_ID

MOVLW        PKT_ENDFLAG
MOVLW        TX_BUFFER
CALL         SEND_CHAR

Normal_Timeout:
BCF          STATUS, RP0                ; Bank 0
MOVLW        DP_SEND
MOVLW        DP_TOSEND

MOVLW        TMR1_NOR
MOVLW        TMR1_COUNT
CALL         RESET_TMR1

RETURN

Rcv_Node1:
; Node1 mode
CALL         SET_NOR
```

```
RETURN

; **********************
; Tmr1 Interrupt Routine
; **********************
TMR1_INT
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank 0

DECFSZ TMR1_COUNT, 1
GOTO         Tmr1_IntA

BCF          T1CON, TMR1ON              ; stop timer 1
BCF          PIR1, TMR1IF               ; clear timer 1 overflow flag

; TMR1_COUNT zero
BTFSC        MODE, MODEBIT_NODE1
                                        ; take next step if in NODE1 mode
GOTO         Tmr1_IntNode1

; Mode Normal Timeout
DECFSZ DP_TOSEND, 1
GOTO         SendDPAgain

CALL         SET_NODE1
RETURN

SendDPAgain:

BCF          STATUS, RP0                ; Bank 0
MOVLW        PKT_DP
MOVLW        TX_BUFFER
CALL         SEND_CHAR

MOVLW        TMR1_NOR
MOVLW        TMR1_COUNT
CALL         RESET_TMR1

RETURN

Tmr1_IntA:
; TMR1_COUNT not zero
CALL         RESET_TMR1
RETURN

Tmr1_IntNode1:
; Mode Node 1 Timeout
BCF          STATUS, RP0                ; Bank 0
MOVLW        PKT_STARTFLAG
MOVLW        TX_BUFFER
CALL         SEND_CHAR

CALL         SEND_DATA

BCF          STATUS, RP0                ; Bank 0
MOVLW        PKT_ENDFLAG
MOVLW        TX_BUFFER
CALL         SEND_CHAR

BCF          STATUS, RP0                ; Bank 0
MOVLW        TMR1_NODE1
MOVLW        TMR1_COUNT
CALL         RESET_TMR1

RETURN
```

```
; ***************
; Reset Timer1
; ***************
RESET_TMR1
; each interrupt will be 0.005 sec

BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank 0

BCF          T1CON, TMR1ON              ; Stop Timer 1

MOVLW        0x057                      ; to load 5 ms
MOVLW        TMR1L
MOVLW        0x09E
MOVLW        TMR1H

MOVLW        0x000                      ; Clear Timer 1
MOVLW        T1CON                      ; Timer1 control, 1:1 prescalar, internal clock
                                        ;
stop timer1
BCF          PIR1, TMR1IF               ; clear the overflow flag

BCF          STATUS, RP0                ; Bank 0
BSF          T1CON, TMR1ON              ; Start Time 1

RETURN

; ***************************
; Serial Transmission
; ***************************

SEND_CHAR
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0
MOVF         TX_BUFFER, 0
MOVLW        TXREG

BSF          STATUS, RP0                ; Bank1

WaitChar:
BTFSS        TXSTA, TRMT                ; wait for transmit buffer to clear (TRMT set)
GOTO         WaitChar

RETURN

; **********
; Data Byte
; **********
SEND_DATA
BCF          STATUS, RP1
BCF          STATUS, RP0                ; Bank0

MOVF         NODE_ID, 0
MOVLW        DATA_BYTE

RLF          IR_STATUS, 1               ; to load into carry
RLF          DATA_BYTE, 1               ; to load from carry

RLF          IR_STATUS, 1
RLF          DATA_BYTE, 1

RLF          IR_STATUS, 1
RLF          DATA_BYTE, 1

MOVLW        IR_NIL
MOVLW        IR_STATUS
```

```
MOVF        DATA_BYTE, 0

MOVLW       TX_BUFFER
CALL        SEND_CHAR

RETURN

START
BCF         STATUS, RP1
BCF         STATUS, RP0             ; Bank 0

; Settings:
CLRF        OUTPUT
BCF         PORTC,1
BCF         PORTC,0
; MAIN CODE:
LOOP
CLRF        COUNTL                  ; clear the counter
MOVF        OUTPUT,0                ; W <- OUTPUT

WAIT0
; Serial Rcv Interrupt
BCF         STATUS, RP0             ; Bank 0
BTFSC       PIR1, RCIF
CALL        RCV_INT

            ; Timer1 Interrupt
BCF         STATUS, RP0             ; Bank 0
BTFSC       PIR1, TMR1IF
CALL        TMR1_INT

BCF         STATUS, RP0             ; Bank 0
BTFSC       PORTB,IR_PIN            ; check IR receiver
GOTO        WAIT0                   ; wait till the receiver becomes 0

BSF         PORTC,2

; come here if RB0 = 0
COUNTING    ; COUNTL
INCF        COUNTL                  ; This loop must be 25 instruction cycles on 20MHz crystal
MOVF        COUNTL,1                ; check whether count =0
BTFSC       STATUS, Z               ; check the zero flag
GOTO        WAIT1                   ; if COUNTL = 0 then goto WAIT1
NOP                                 ;
NOP                                 ; For 1KHz input the count should be 100 (ghost)
NOP                                 ;
NOP                                 ; For 500Hz input the count should be 200 (pacman)
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
NOP                                 ;
BTFSS       PORTB, IR_PIN           ;
GOTO        COUNTING                ; if still LOW then goto COUNTING

MOVF        COUNTL,0                ; W <- COUNTL
MOVLW       TEMPL                   ; TEMP <- W
```

```
BCF          STATUS,C
RRF          TEMPL,1                    ; rotate temp to right
BCF          STATUS,C                   ; clear carry bit
; now TEMPL= COUNTL/2

LOOPH
CLRF         COUNTH                     ; Now we check whether the high time is
                                        ; EQUal to the low time of IR_input

COUNTING_H
INCF         COUNTH                     ; This loop must be 25 instruction cycles on 20MHz crystal
MOVF         COUNTH,1                   ; Check whether count =0
BTFSC        STATUS, 2                  ; check the zero flag
GOTO         NO_VALID                   ;
NOP                                     ;
NOP                                     ; For 1KHz input the count should be 100 (ghost)
NOP                                     ;
NOP                                     ; For 500Hz input the count should be 200 (pacman)
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
NOP                                     ;
BTFSC        PORTB, IR_PIN              ;
GOTO         COUNTING_H                 ;

BSF          PORTC,3

MOVF         COUNTH,0                   ; W <- COUNTH
MOVLW        TEMPH                      ; TEMPH <- W
BCF          STATUS,C
RRF TEMPH,1  ; rotate temp to right
BCF          STATUS,C                   ; clear carry bit

; Now,
; TEMPL = COUNTL/2
; TEMPH = COUNTH/2
MOVF         TEMPL, 0                   ; W <- TEMP
ADDWF        TEMPH, 0                   ; W = W + TEMPH
MOVLW        TOTAL_TIME                 ; TOTAL_TIME <- W

; TOTAL_TIME=TOTALTIME/2:
BCF          STATUS,C
RRF          TOTAL_TIME,1
BCF          STATUS,C

; Now TOTAL_TIME is the sum of LOW and HIGH time divided by 4

MOVLW        PAC_PERIOD                 ; W <- PAC_PERIOD
SUBWF        TOTAL_TIME, 1              ; TOTAL_TIME <- TOTAL_TIME - PAC_PERIOD
BTFSS        TOTAL_TIME, 7              ; TEST MSB OF RESULT
GOTO         RESULT_IS_POSITIVE
COMF         TOTAL_TIME, 1              ; COMPLEMENT TOTAL_TIME
INCF         TOTAL_TIME, 1
; now TOTAL_TIME= ABS( PAC_PERIOD - (COUNTL + COUNTH)/4 )
RESULT_IS_POSITIVE                      ; ABS(TOTAL_TIME - PAC_PERIOD)
MOVLW        PAC_PERIOD_BOUND           ; W <- PAC_PERIOD_BOUND
```

```
SUBWF        TOTAL_TIME, 1              ; TOTAL_TIME <- TOTAL_TIME - PAC_PERIOD_BOUND
BTFSS        TOTAL_TIME, 7              ; TEST MSB OF RESULT. IT SHOULD BE NEGATIVE
GOTO         TEST_GHOST
             ; @@@@@ PACMAN detected @@@@@
MOVLW        IR_PACMAN
MOVLW        IR_STATUS
BCF          OUTPUT,1                   ; clear ghost
BSF          OUTPUT,0                   ; detect PACMAN
BCF          PORTC,1
BSF          PORTC,0
GOTO         LOOP                       ; Its PACMAN

TEST_GHOST
MOVF         TEMPL, 0                   ; W <- TEMP
ADDWF        TEMPH, 0                   ; W = W + TEMPH
MOVLW        TOTAL_TIME                 ; TOTAL_TIME <- W
; Now TOTAL_TIME is the sum of LOW and HIGH time

; TOTAL_TIME=TOTALTIME/2:
BCF          STATUS,C
RRF          TOTAL_TIME,1
BCF          STATUS,C

MOVLW        GHO_PERIOD                 ; W <- GHO_PERIOD
SUBWF        TOTAL_TIME, 1              ; TOTAL_TIME <- TOTAL_TIME - GHO_PERIOD
BTFSS        TOTAL_TIME, 7              ; TEST MSB OF RESULT
GOTO         RESULT_IS_POSITIVE2
COMF         TOTAL_TIME, 1              ; COMPLEMENT TOTAL_TIME
INCF         TOTAL_TIME, 1
RESULT_IS_POSITIVE2                     ; ABS(TOTAL_TIME - GHO_PERIOD)
MOVLW        GHO_PERIOD_BOUND           ; W <- GHO_PERIOD_BOUND
SUBWF        TOTAL_TIME, 1              ; TOTAL_TIME <- TOTAL_TIME - GHO_PERIOD_BOUND
BTFSS        TOTAL_TIME, 7              ; TEST MSB OF RESULT
GOTO         NO_VALID
; @@@@@ GHOST detected @@@@@
MOVLW        IR_GHOST
MOVLW        IR_STATUS
BSF          OUTPUT,1                   ; detect GHOST
BCF          OUTPUT,0                   ; clear PACMAN
BSF          PORTC,1
BCF          PORTC,0
GOTO         LOOP                       ; its GHOST

WAIT1
BTFSS        PORTB,IR_PIN               ; check IR receiver
GOTO         WAIT1                      ; wait till the receiver becomes 1

NO_VALID
CLRF         OUTPUT
BCF          PORTC,1
BCF          PORTC,0
GOTO         LOOP                       ; go to the first loop

END
```

Appendix C

PCB Designs

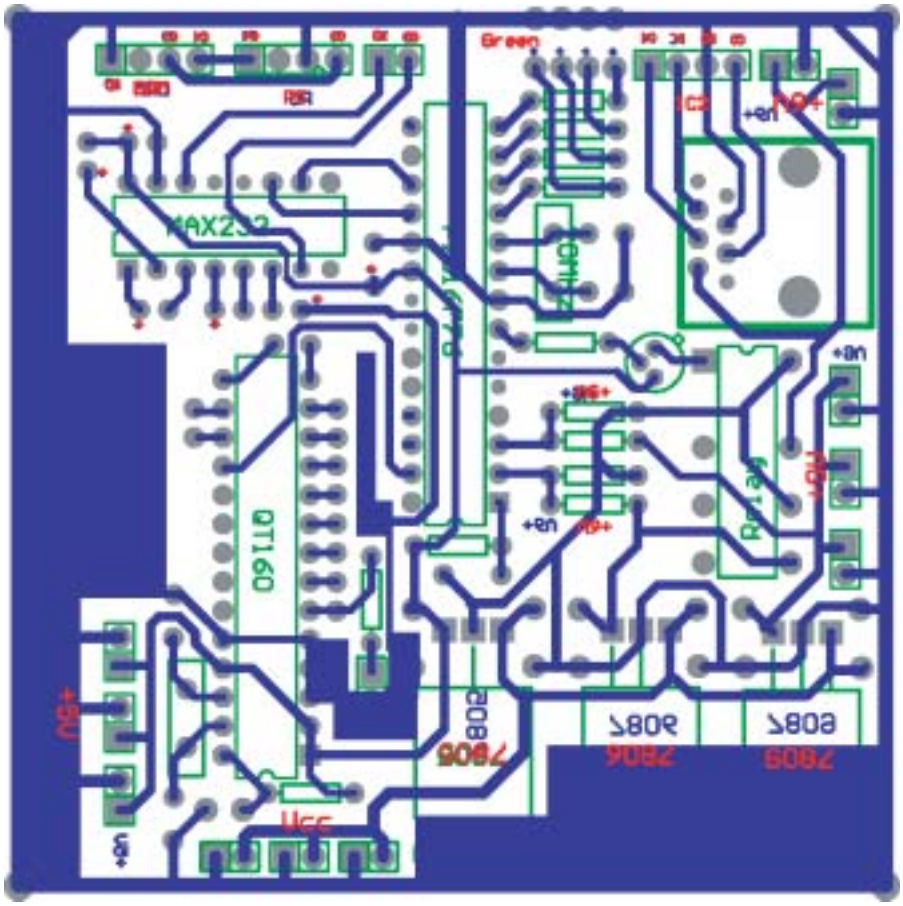


Figure C.1: PCB design of controller circuit in wearable computer.

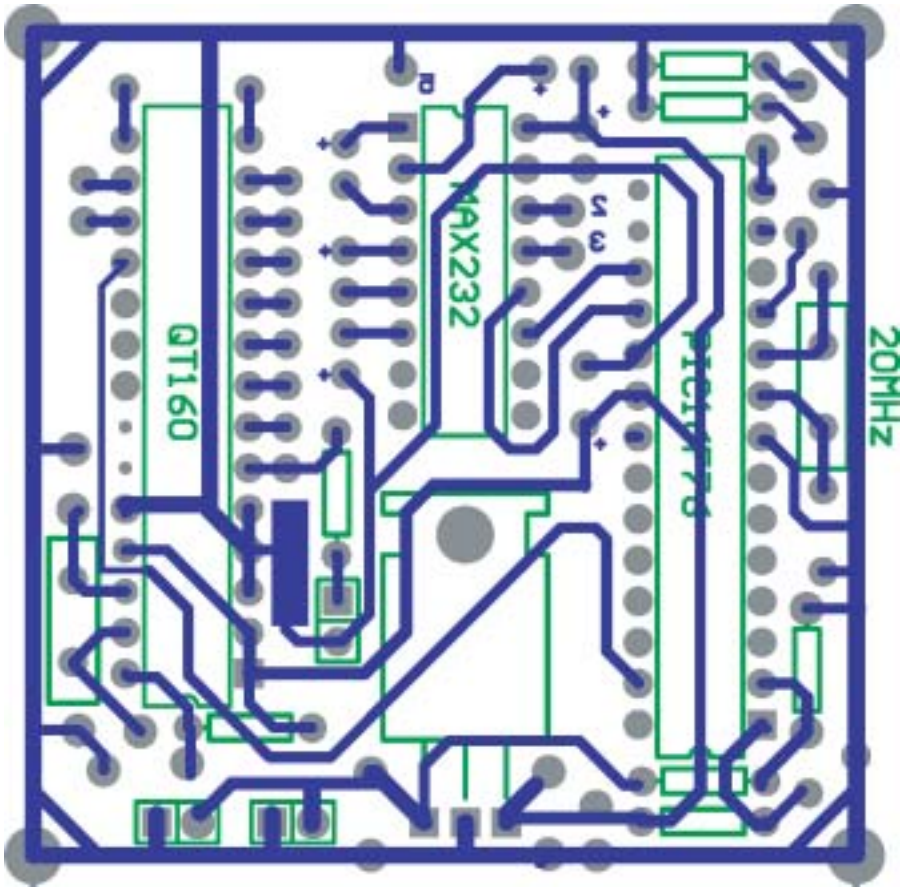


Figure C.2: PCB design of controller circuit in Bluetooth embedded object.

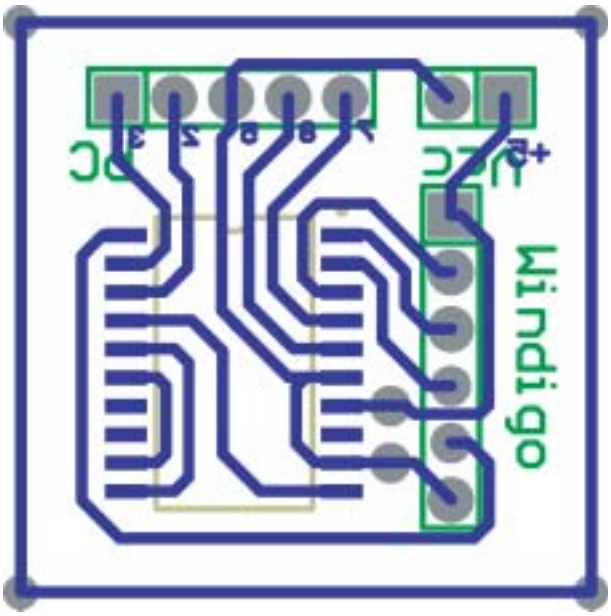


Figure C.3: PCB design of adding RS232 interface to Windigo Bluetooth module.

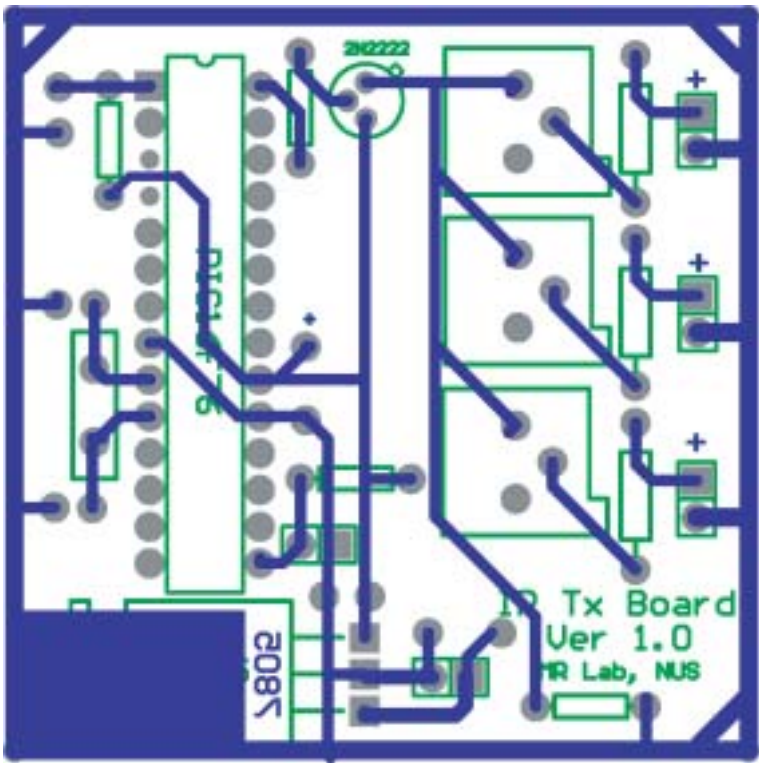


Figure C.4: PCB design of infra-red emitter circuit.

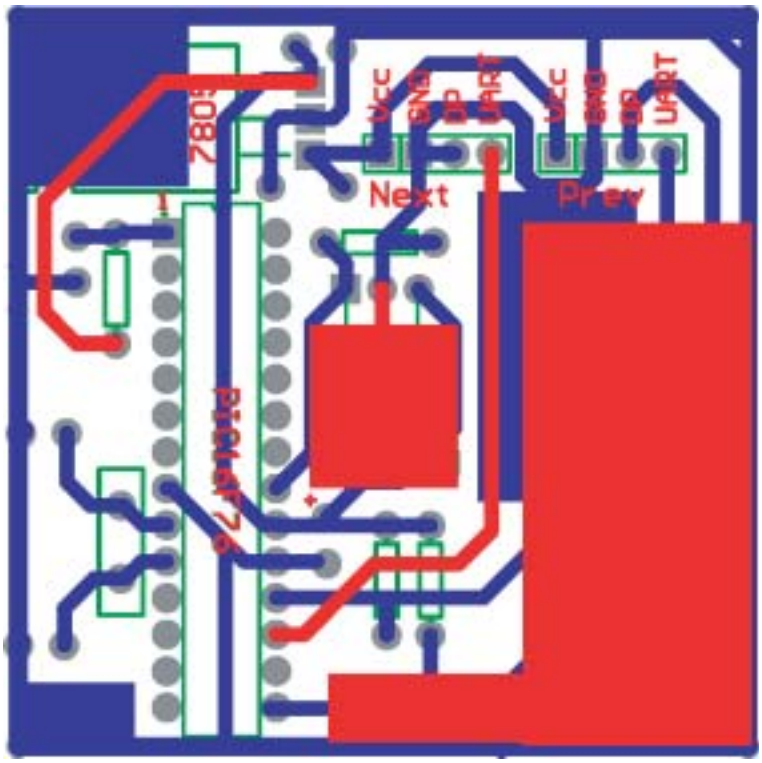


Figure C.5: PCB design of infra-red receiver circuit.

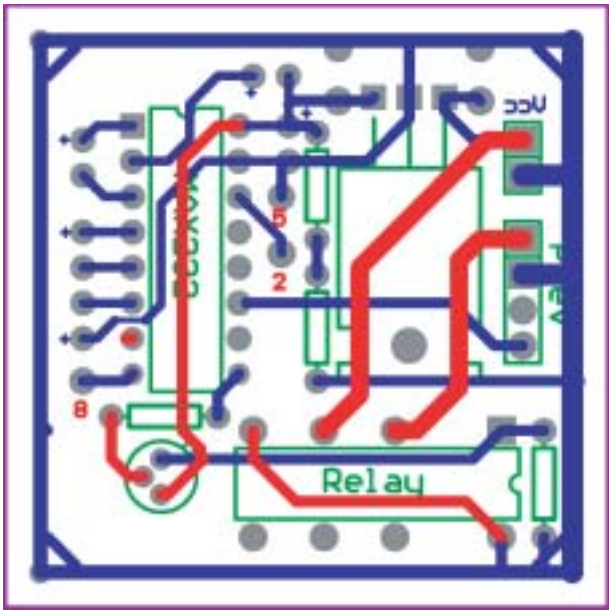


Figure C.6: PCB design of interfacing connection between PC and infra-red receiver circuit.

Appendix D

Demo Write-up

D.1 CHI 2004 - Connect

Date : **27th April(Tuesday) to 29th April 2004(Thursday)**

Venue : **Austria Centre Vienna (ACV) (Vienna, Austria)**

CHI 2004 was held in Vienna, Austria over a period of three days, from 27th to 29th April 2004. The theme for this year's conference is CONNECT. Our lab sent a team of 6 people for the demonstration of our Human Pacman project. We were also invited to host a station in the Games SIG: The Untapped World of Video Gaming during the conference. It took us 8 months to prepare for this big event held for an international audience from the HCI community.

The Game SIG took place on the 27th from 1430hrs to 1600hrs. The SIG is meant for researchers who use games as their research platform. Essentially this was meant to be an informal opportunity to allow other games researchers hands-on access to the game that we worked with. Session attendees would be able to move from station to station during the 90 minute session to play the games and

discuss games research with the station hosts. We showed our lab's work on Tilt Pad Pacman, which brought interest from several people. Fig. D.1 shows the demo being done.

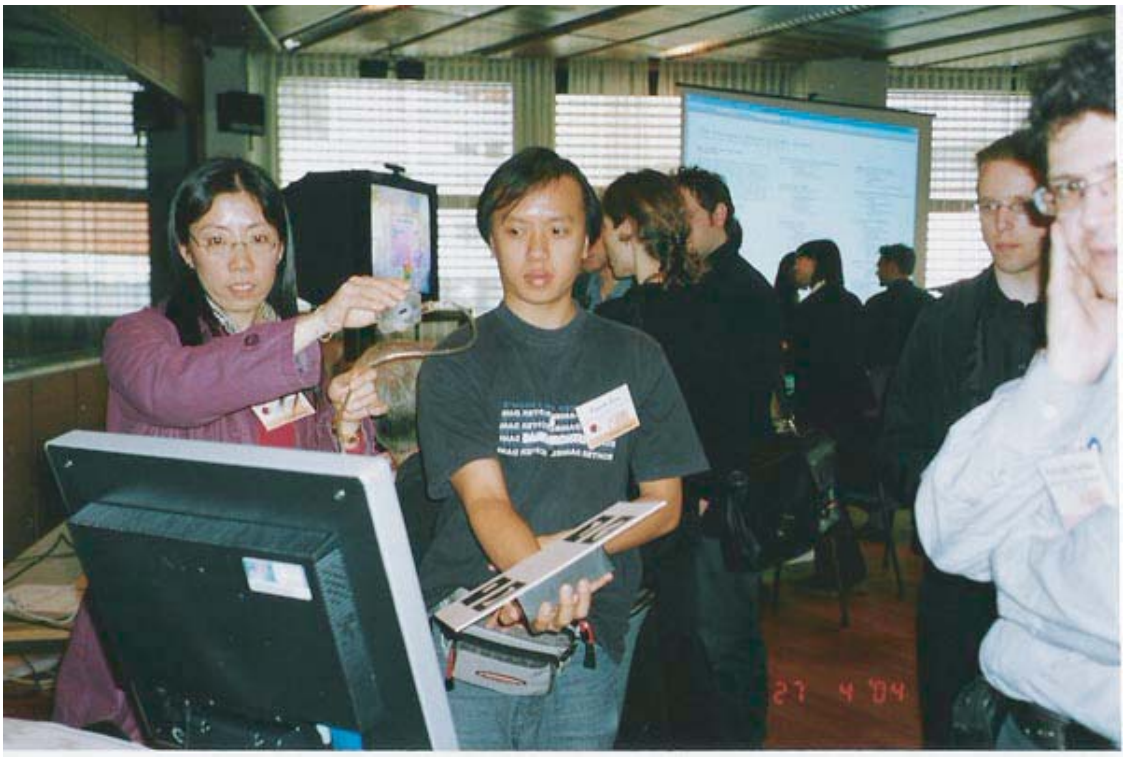


Figure D.1: Tilt-pad Pacman demo during the Game SIG

Our demo, under the session of Games and Virtual Environments, was shown on the 29th between 1130hrs - 1200hrs. Despite having some network glitches which caused one of our wearable computer to crash, a successful demo was still shown when the whole system was up and working finally. We successfully presented our work to the HCI community. During the Q & A, the demo chair stood up, didn't ask a question but just said "Thank you; you have raised the bar of demos in CHI", bringing on applause from the spectators in the Hall. After the session, several

enthusiastic members of the public tried the Human Pacman system, enacting as either the Pacman or the Ghost.

In all, CHI 2004 was a success for Mixed Reality Lab. We look forward in sharing more work with the HCI community in the near future.

Figure D.2: The Human Pacman team after the demo in CHI2004

D.2 ACE 2004

Date : **3rd June (Thursday) to 5th June 2004(Saturday)**

Venue : **National University of Singapore (Singapore)**

ACE 2004 was held in Singapore over a span of three days, from 3rd June to

5th June 2004. The purpose of this conference was to bring together academic and industry researchers, as well as computer entertainment developers and practitioners, to address and advance the research and development issues related to computer entertainment.

Over the course of the conference, researchers from all over the world came together to share their work on computer entertainment. Works range from system that detects anxiety to vary difficult of game, to physical cubes that are put together to form virtual objects. Our lab featured several research works during the poster/demo on the 4th from 1420hrs to 1620hrs. These include "Human Pacman", "AR Post-It System", "Jumanji Singapore", "Magic Story Cube", and "A Step Towards Anywhere Gaming". Fig. D.3 shows the Human Pacman demo during the demo session. The demo went smoothly for the whole session, with several interested parties enquiring about the game. Local reporters did an interview with us, and also took some materials and photos of the Human Pacman system.

In all, ACE 2004 was a meaningful exchange for the researchers, and a successful avenue for the Mixed Reality Lab to share our work with the international audience.

Figure D.3: Human Pacman demo in ACE 2004

Appendix E

News Articles

Bibliography

[1] The Killer List Of Videogames. Pac-man v r. [Online]. Available at: http://www.klov.com/game_detail.php?game_id=8960&letter=P.

[2] Sony Computer Entertainment Inc. Pacman world 2. [Online]. Available at: http://nz.playstation.com/ps2/games/p/pacmanworld2.jhtml.

[3] D. Myers. Computer game semiotics. *Play and Culture*, 4:334–345, 1991.

[4] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):94–100, 1991.

[5] Dourish. P. *Where The Action Is: The Foundations of Embodied Interaction.* MIT Press, Cambridge, Massachusetts, 2001.

[6] P. Wellner, W. Mackay, and R. Gold. Computer augmented environments: Back to the real world. *Communications of the ACM*, 36(7):24–26, July 1993.

[7] Smith, C.G. The hand that rocks the cradle. *I.D.*, pages 60–65, May/June 1995.

[8] Fitzmaurice, G., et al. Bricks: Laying the foundations for graspable user interfaces. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI '95), ACM*, pages 442–449, Denver, May 2001.

[9] Ishii, H. and Ullmer, B. Tangible bits:towards seamless interfaces between people, bits, and atoms. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI '97)*, pages 22–27, March 1997.

[10] Z. Zhang and Y. Shan. State of the industry report. [Online], 2001. Available at: http://www.theesa.com/pressroom.html.

[11] Konami Corporation. Paraparaparadise. [Online], 2001. Available at: http://www.konami.com.hk/kma_ac/ac_ppp/ppp.html.

[12] Entertainment Software Association. Essential facts about the computer and video game industry. [Online], 2002. Available at: http://www.theesa.com/pressroom.html.

[13] Crawford C. Live: What a concept! networked games. *Digital Illusion*, pages 241–248.

[14] J., Bowlby. *Attachment and Loss, Volume I: Attachment.* Basic Books, New York, USA, 1983.

[15] Björk, S., Falk, J., Hansson, R., K. Nakao and Ljungstrand, P. Pirates! - using the physical world as a game board. In *Interact 2001, IFIP TC. 13 Conference on Human-Computer Interaction*, Tokyo, Japan, July 2001.

Available at:

http://www.viktoria.se/fal/publications/play/2001/pirates.interact.pdf.

[16] Mandryk, R. L. and Inkpen, K. M. Supporting free play in ubiquitous computer games. In *Workshop on Designing Ubiquitous Computer Games, UbiComp 2001*, Atlanta, 2001. Available at:

http://www.playresearch.com/workshops/ubigame.ubicomp/.

[17] Lantz, L. et. al. Pac manhattan. [Online]. Available at:

http://pacmanhattan.com/.

[18] Anastasi R. Flintham M. Greenhalgh C. Tandavanitj N. Adams M. Row-Farr J. Benford, S. *IEEE Pervasive Computing*, 2(3):34–41, 2003.

[19] Benford, S., Fraser, M., Reynard, G., Koleva, B., Drozd A. Staging and evaluating public performances as an approach to cve research. In *Proceedings of the 4th international conference on Collaborative virtual environments*, pages 80–87, 2002.

[20] Flintham, M., Benford, S., Anastasi, R., Hemmings, T., Crabtree, A., Greenhalgh, C., Tandavanitj, N., Adams, M., Row-Farr, J. Where on-line meets on the streets: experiences with mobile mixed reality games. In *Proceedings of the 2003 Conference on Human Factors in Computing Systems, CHI 2003*, pages 569–576, April 2003.

[21] Crabtree, A., Benford, S., Rodden, T., Greenhalgh, C., Flintham, M., Anastasi, R., Drozd, A., Adams, M., Row-Farr, J., Tandavanitj, N., Steed, A. Orchestrating a mixed reality game 'on the ground'. In *Proceedings of the 2004 Conference on Human Factors in Computing Systems, CHI 2004*, pages 391–398, April 2004.

[22] Oshima, T., Satoh, K., Yamamoto, H. and Tamura, H. Ar2 hockey system: A collaboration mixed reality system. *Trans. of Virtual Reality Society of Japan*, 3(2):55–60, 1998.

[23] Tamura, H., Yamamoto, H. and Katayama, A. Mixed reality: Future dreams seen at the border between real and virtual worlds. *Computer Graphics and Applications*, 21(6):64–70, 2001.

[24] Cheok, A.D., Yang, X., Zhou, Z.Y., Billinghurst, M., Kato, H. Touch-space: Mixed reality game space based on ubiquitous, tangible, and social computing. *Personal and Ubiquitous Computing*, 6(5-6):430 – 442, 2002.

[25] Starner, T. et. al. Augmented reality through wearable computing. *Presence*, 6(4), Winter 1997.

[26] Thomas, B., Close, B., Donoghue, J., Squires, J., Bondi, P. D.,and Piekarski, W. First person indoor\outdoor augmented reality application: Arquake. *Personal and Ubiquitous Computing*, 6(1):75–86, 2002.

[27] W. Hunter. The dot eater. [Online]. Available at: http://www.emuunlim.com/doteaters/play2sta4.htm.

[28] Elitegroup Computer Systems. Desknote a980. [Online]. Available at: http://www.desknote.net/dn/spec.cfm?modID=117.

[29] NVIDIA Corporation. Nvidia geforce4 go. [Online]. Available at: http://www.nvidia.com/page/geforce4go.html.

[30] Daeyang E& C Co. Ltd. Cyvisor. [Online]. Available at: http://www.personaldisplay.com/english/default.html.

[31] Unibrain S.A. Fire-i digital camera. [Online]. Available at: http://www.unibrain.com/1394_products/fire-i_dig_cam/digital_camera_pc.htm.

[32] Handkey Corporation. Twiddler 2. [Online]. Available at: http://www.handykey.com/site/twiddler2.html.

[33] DLink Australia. Dwl-120+. [Online]. Available at: http://www.dlink.com.au/products/wireless/dwl120+/.

[34] TDK Systems. Bluetooth usb adapter. [Online]. Available at: http://www.tdksystems.com/products/intro.asp?id=4.

[35] InterSense Inc. *Inertia Cube2 Bringing 3D To Life - Manual for Serial Port Model*. InterSense Inc, 2001.

[36] Point Grey Research Corporation. *DRM III Dead Reckoning Module Engineering Evaluation Kit Programmer/User Guide*. Point Grey Research Corporation, 2003.

[37] Quantum Research Group Ltd. Qt160 datasheet. [Online]. Available at: http://www.qprox.com.

[38] Philipp H. Charge transfer sensing. [Online]. Available at: http://www.qprox.com/background/white_paper.php.

[39] FairChild Semiconductor. *MC78XX/LM78XX/LM78XXA Datasheet*. 2001.

[40] MicroChip. *PIC16F7X Datasheet*. 2002.

[41] Maxim. *+5V-Powered, Multichannel RS-232 Drivers/Receivers*. 2003.

[42] Keyspan. High speed usb serial adapter. [Online]. Available at: http://www.keyspan.com/products/usb/USA19HS/.

[43] BlueZ Project. Bluez. [Online]. Available at: http://www.bluez.org.

[44] Advanced Micro Peripherals Ltd. *Tiny 886ULP-X Reference Manual*. 2002.

[45] Transmeta Corporation. Crusoe tm5600 and tm5400 processors. [Online]. Available at: http://www.transmeta.com/crusoe/crusoe_tm5600_tm5400.html.

[46] TIQIT Computers. *Matchbox PC Manual*. Oct 2000.

[47] Microsoft Corporation. Windows xp. [Online]. Available at: http://www.microsoft.com/windowsxp/default.mspx.

[48] SGI. Open gl. [Online]. Available at: http://www.opengl.org/.

[49] Mixed Reality Lab. mxrtoolkit. [Online]. Available at: http://mixedreality.nus.edu.sg/.

[50] Nielsen, Jakob, and Landauer, Thomas K. A mathematical model of the finding of usability problems. In *Proceedings of ACM INTERCHI'93 Conference*, pages 206–213, Amsterdam, The Netherlands, 24-29 April 1993. ACM Press.

[51] Vishay Telefunken. *TSUS540*. 1999.

[52] Hero Electronics Limited. *HIM602 Infrared Remote Control Receiver*. Mar 2002.