Embedded web server remote control and communication

Lihua Liu & Weikun Huo
Embedded web server remote control and communication

Bachelor thesis
2012 June

Author: Lihua Liu & Weikun Huo
Supervisor: Tommy Salomonsson
Examiner: Kenneth Nilsson

School of Information Science, Computer and Electrical Engineering
Halmstad University
PO Box 823, SE-301 18 HALMSTAD, Sweden
Acknowledgement

It is a great pleasure for us to introduce our respected supervisor Tommy Salomonsson. We are all appreciate the support, encouragement and feedback he gave us during the whole time of the thesis work. To a large degree, these inspired us to finish this work.

Weikun Huo : Thanks my partner Lihua Liu, who performed as a good team player. She is a girl with patience, self-motivated, perseverance, high responsibility, thrive on problems. She also gave me a lot of encouragement and support. It’s a pleasure to work with her.

Lihua Liu : Thanks my partner Weikun Huo, who performed as a good team player. She is a girl with careful, high responsibility, aggressive, thrive on problems. It’s a pleasure to work with her.
Abstract

This is a embedded system that designed to achieve the goal of remote control and wireless communication with web page. The embedded web server using uIP which is TCP/IP stack. Web page are designed using HTML5, JavaScript and CSS language to format the layout. On the home page, you can see the index of different items.

On the page of remote Control LED. User can control the onboard LED with the buttons on the web page. The page is simple and operation procedure is friendly. Everyone can use it without any train.

Another page is for wireless communication which is Control robot-PIE implemented by radio frequency. One radio module connected to the server board which is the master part, and another radio module installed on the PIE as slave part. On the web page, there are four buttons which stand for the four directions that PIE can move. User can control PIE via press the corresponding button. This communication implemented over radio frequency. In this communication, there are no protocols like WIFI or even bluetooth, only the most basic protocol to send data is used to build the communication.

Over remote control, every domestic appliances can be connected to the Internet and control no matter where the location of users. Applying this system, users just need install common web browser to achieve a truly cross-platform system of great practical significance.

[Key words]Embedded web server; μ IP TCP/IP stack; Remote control; Wireless communication; Radio frequency
Contents

1 Introduction ............................................................................................................................................. 1
  1.1 Project proposal ................................................................................................................................. 1
  1.2 Meaning of the work. ......................................................................................................................... 1
  1.3 Goals .................................................................................................................................................. 2

2 Background ........................................................................................................................................... 3
  2.1 B/S structure ...................................................................................................................................... 3
  2.2 About ARM ....................................................................................................................................... 3
  2.3 History of Embedded System ............................................................................................................ 4
  2.4 Literature study .................................................................................................................................. 5
     2.4.1 Analysis of project Proposal ...................................................................................................... 5
     2.4.2 Device introduction .................................................................................................................... 6
     2.4.3 Web server ................................................................................................................................. 9
     2.4.4 Protothread ...............................................................................................................................11
     2.4.5 Embedded Ethernet ...................................................................................................................12
     2.4.6 HTTP Protocol ..........................................................................................................................12
     2.4.7 Html5. Shtml .............................................................................................................................13
     2.4.8 JavaScript ...................................................................................................................................15
     2.4.9 CSS ............................................................................................................................................16
     2.4.10 CGI ...........................................................................................................................................16
     2.4.11 Hardware Interrupt ..................................................................................................................17
     2.4.12 SSP ..........................................................................................................................................20
  2.5 Similar Works .................................................................................................................................... 21

3 Method ..................................................................................................................................................... 23
  3.1 Why uIP ............................................................................................................................................ 23
  3.2 Web browser ..................................................................................................................................... 24
  3.3 Web page ......................................................................................................................................... 24
  3.4 Hardware interrupt ........................................................................................................................... 26
  3.5 SPI .................................................................................................................................................... 27

4 Results ................................................................................................................................................... 29
  4.1 web server ....................................................................................................................................... 29
  4.2 web page ......................................................................................................................................... 29
  4.3 Led blinking ....................................................................................................................................... 32
  4.4 Hardware interrupt .......................................................................................................................... 32
  4.5 Wireless communication and LCD display ...................................................................................... 34
5 Conclusions..................................................................................................................39
  5.1 Result Demonstrate....................................................................................................39
  5.2 New..........................................................................................................................40
  5.3 Experience................................................................................................................41
  5.4 Future plan...............................................................................................................41

6 Discussion.....................................................................................................................43
  6.1 Statement..................................................................................................................43
  6.2 Weakness................................................................................................................43

7 Reference.......................................................................................................................45
Chapter 1 Introduction

1 Introduction

1.1 Project proposal

In IDE lab room, there are mobile robots which are called PIE. There is a video camera in the ceiling of the lab that can monitor the position of the PIE, if someone control the PIE, it will be recorded. This project provider wants to control the robot-PIE via an embedded remote control system while monitoring from the camera.

With a web server installed on an embedded system development board many things can be remote checked and/or controlled. Users can control equipments via web page which opened in a common web browser, like IE, Firefox, Chrome, etc. The aim of the work is to implement remote control PIE. The communication between web server and PIE is achieved by radio modules. The communication between client and server use the relative fast and reliable Ethernet to transmit data.

According to the requirements of the instructor, the development of the project is based on LPC-2378-STK from OLIMEX.

1.2 Meaning of the work

A monitoring control system based on embedded web technology for indoor devices (PIE) is presented to realize the on-line remote supervision, management and maintenance, using a standard Web browser over Internet without time-space constraints and some special software. The system has cross-platform compatibility which resolves network access problem effectively making use of finite resource. Practical results demonstrate the favorable stability and real time characteristic of the system which can meet the most of network application demands and is world popularizing.

1.3 Goals

In order to achieve the goal, there are some mandatory tasks lists below:
Embedded web server remote control and communication

Mandatory Tasks:

1. Build a web browser (PC) to show the web page from the server (remote board).
2. Implement TCP/IP stack on the development board.
3. Use the LEDs (on the board) to signal the process.
4. The client can read values of the server and show on the web page.
5. The boards which join the remote communication as a server could show part or all of the data received from the client on its LCD.
6. Implement drivers for the radio module.

Bonus Tasks:

7. Via setting accounts, defines different multiple role groups and makes a finite set of rules to assign user to roles. Login with different restrictions like administrator and student account.
8. Implement communication with the mobile robot --- “PIE”
Chapter 2 Background

2 Background

2.1 B/S structure

One of the biggest attraction of the project is use web page access web server and remote control device. This structure is the current popular B/S structure in stead of traditional C/S structure. B/S means Browser/Server, is a new network architecture model after the rising up of WEB technology. The web browser is the most important applications of the client. This model unifies the client, make the core part of implementation of systemic functions focus on the server side. Simplifies the system development, maintenance and use. With the only installed web browser on the client side, such as Nerscape Navigator or Internet Explorer, the server install the Oracle, the Sybase, Informix, or SQL server and other databases. The browser interacts with the database via server.

The biggest advantages of the B/S architecture is that it can be operated anywhere without having to install any special software, as long as there is a computer with Internet access. Achieving zero maintenance client. The expansion of the system is very easy.

B/S architecture is now widely used and promote the develop of AJAX[1], it can deal part of functions on the client side, thereby greatly reducing the burden on the server side and increasing interaction. AJAX can also refresh part of web page.

Advantages :

✓ Easy to maintenance and upgrade
✓ Cost reduction, more selection

2.2 About ARM

ARM[2] (Advanced RISC Machines), both considered to be the name of a company and a generic term for a class of microprocessors, but also can be considered to be the name of a technology.
Embedded web server remote control and communication

ARM was founded in 1991 in Cambridge, England, mainly work on the sale of the authorization of the chip design technology. At present, the use of ARM technology microprocessor intellectual property (IP) cores, which usually refer to ARM technology-based microprocessor applications, throughout the industrial control, consumer electronics, communication systems, network systems, wireless systems and other types of product market. ARM technology-based microprocessor applications, accounting for more than 75% of the 32-bit RISC microprocessor market share.

ARM company is specialized in the RISC-based chip design and development company. As a supplier of intellectual property rights, does not engaged in chip production directly, distinctive chips produced by the partner companies through the assignment of license, the world’s major semiconductor manufacturers buy the design of the ARM microprocessor core, according to different fields of application, add the appropriate external circuit to form their own ARM microprocessor chip to enter the market. World dozens of large semiconductor companies are using ARM’s authorization, not only makes the ARM technology supported by more third-party tools, manufacturing and software, but also reduce the cost of the whole system to make this products more accessible to the market and accepted by consumers with more competitive.

2.3 History of Embedded System

Embedded system has a history near to 30 years since the appearance of the first single chip in 70s of 20th century to the large-scale application of embedded processors and microcontrollers.

The developing of a system is always supported by the hardware and software alternate development. Embedded system is becoming steady and mature in this way.

Embedded system is based on single chip at first. The automobile, appliance, industry, communication equipments and thousands of products achieved better property like easy to use, faster, and cheaper because of the built-in electric chips. These Equipments basically be provided with the characters of embedded. The embedded systems technology has very broad application prospects, its applications could include:

1. Robot: the development of embedded chips will enable the robot get advantages in miniaturization, high intelligence, while a significant reduction in the price of the robot, that a wider range of applications in industrial areas and service areas.

2. Industrial control: currently, there is a large number of 8, 16, 32-bit embedded microcontroller applications, the network is the main way to improve production efficiency and product quality, reduce manpower, such as industrial process control,
Embedded web server remote control and communication

digital machine tools, power systems, grid security, grid equipment monitoring, petrochemical system.

3. Family intelligent management system: water, electricity, gas meters, remote automatic meter reading, fire safety, anti-theft system, which inserts some special control chip will replace the traditional manual inspection, and to achieve higher, more accurate and safer properties.

2.4 Literature study

2.4.1 Analysis of project Proposal

According to the thesis proposal, what need to be done is:

For user interface:

✓ Web page - interaction with users
✓ Common web browser

For server side:

✓ Web server
✓ Wireless Communication between web server and radio module

There project can be seen as two parts communication. The first one is "Ethernet communication", the other one is "Wireless communication".

Ethernet communication

Figure 2-1 shows the main structure of this part of communication. It is clearly show that if build a web page, HTML[3], CSS[4] is necessary, and according to this situation, there is another technology that is SHTML[5] need to be use. JavaScript[6] is used to make animation on web page. The web server is built use uIP. In this figure, shows access a web page of a server need Ethernet and hypertext transfer protocol. To make the web page dynamic that user can interact with it will need CGI function to work on the server side. CGI deals with the request sent in by server, process it then transfer back to web page over serve. Details about technology would be explained later.
Embedded web server remote control and communication

This part is one of the cores of the thesis. In order to achieve the communication, SSP and Hardware Interrupt are used to built, these two techniques will be introduced later.

The communication between figure 2-2(a) and figure 2-2(b) is using SSP. MCU send command to Radio module and get data back from figure2-3(c)radio module and show data on the screen. Communication between (b) and (c) is completed by trigger an interrupt. When radio modules (b) receive command from MCU, it will send them out as radio frequency, in this project, heartbeats have been sent. If the heartbeat received by PIE-radio, it will trigger interrupt then transfer data back.

Figure 2-2: wireless communication
2.4.2 Device introduction

LPC-2378STK

Development prototype board with LPC2378[7]Ethernet, USB, 2x CAN, 2x RS232, Ethernet, SD/MMC, Audio IN-OUT. Figure 3 give a whole view(Figure 2-3).

Features (what being used in project):

- MCU: LPC2378 16/32 bit ARM7TDMI-S™t with 512K Bytes Program Flash, 16K Bytes RAM, EXTERNAL MEMORY BUS, RTC, 4x 10 bit ADC 2.44 uS, 2x UARTs, 4x CAN, I2C, SPI, 2x 32bit TIMERS, 7x CCR, 6x PWM, WDT, 5V tolerant I/O, up to 60MHz operation.
- Standard JTAG connector with ARM 2x10 pin layout for programming/debugging with ARM-JTAG
- Two USER button
- 128x128 pixel 12 bit color TFT LCD with backlight
- Ethernet 100MBit

nRF24L01

MOD-NRF24Lx[8]: RF 2.4GHz Transciever module with nRF24L01(Figure 2-4)

Features :

- NRF24L01 2.4GHz low power transciever
- SPI 0-8Mhz interface
- 125 RF channels
- Up to 2Mbit data rate
- PCB: FR-4, 1.5 mm (0.062”), red soldermask, white silkscreen component print

SAM7P256

Development Board for AT91SAM7S256 [9]ARM7TDMI-S Microcontroller(Figure 2-6).

Features:
Embedded web server remote control and communication

☑ MCU: AT91SAM7S256 16/32 bit ARM7TDMI™ with 256K Bytes Program Flash, 64K Bytes RAM, USB 2.0, RTT, 10 bit ADC 384 ksp, 2x UARTs, TWI (I2C), SPI, 3x 32bit TIMERS, 4x PWM, SSC, WDT, PDC (DMA) for all peripherals, up to 60MHz operation

☑ standard JTAG connector with ARM 2x10 pin layout for programming/debugging with ARM-JTAG

☑ USB connector

☑ UEXT connector

☑ two buttons

☑ two status LEDs

PIE

Figure 2-5 shows appearance of PIE. This robot comes from IDE laboratory provided by the supervisor. A Sam7p256 board with a radio module on it used as slave board installed on the PIE as the brain and heart.

Figure 2-3 : LPC-2378STK-F development board

Figure 2-4 : mod-nRF24L01 radio module

Figure 2-5 : PIE-the robot

Figure 2-6 : sam7p256
2.4.3 Web server

Embedded web-servers are miniature specialized computers that contain a CPU (central processing unit), a TCP/IP stack and an Ethernet connection to provide a bridge to different devices with serial interfaces (RS-232, RS-422, or RS-485), CAN(Controller Area Network) interfaces, GPIB (General Purpose Interface Bus) interfaces etc. They are available as chips, boards, and boxes.

Embedded web server have some figures:

- The only software that remote monitoring and controlling terminal have to install is normal web browser. There is no need to develop specialized software. In that way lower expense of whole system.
- Monitoring terminal platform where web browser located have no independent with the platform of web server. The monitoring terminal platform implements a cross-platform application.
- A simple user interface is straightforward, everyone can access without any train.
- Easy to expand. When upgrade system, you can just add corresponding modules in server side.

Nowadays the common used web servers in embedded devices are: BOA, tHTTPd, lighttpd, apache, etc. There are some descriptions below here:

**uIP 1.0 web server — TCP/IP stack**

1. Introduction: The uIP[10] is an open source TCP/IP stack capable of being used with tiny 8- and 16-bit microcontrollers. It was initially developed by Adam Dunkels of the "Networked Embedded Systems" group at the Swedish Institute of Computer, licensed under a BSD style license, and further developed by a wide group of developers.

uIP is widely used in the embedded systems industry and has been ported to several platforms, including DSP platforms.

The uIP implementation is designed to have only the absolute minimal set of features needed for a full TCP/IP stack. It can only handle a single network interface and does not implement UDP, but focuses on the IP, ICMP and TCP protocols.

The uIP stack does not use explicit dynamic memory allocation. Instead, it uses a single global buffer for holding packets and has a fixed table for holding connection state. The global packet buffer is large enough to contain one packet of maximum size.

2. Features: uIP only have one buffer is the biggest feature. The benefits of uIP are
Embedded web server remote control and communication

saving memory and also support IPv4, IPv6. But there are some disadvantages. If the processing speed can not keep up a little may cause packet lose. Limited to the network environment.

With the success of the Internet, the TCP/IP protocol suite has become a global standard for communication. TCP/IP is the underlying protocol used for web page transfers, e-mail transmissions, file transfers, and peer-to-peer networking over the Internet. For embedded systems, being able to run native TCP/IP makes it possible to connect the system directly to an intranet or even the global Internet. Embedded devices with full TCP/IP support will be first-class network citizens, thus being able to fully communicate with other hosts in the network.

Traditional TCP/IP implementations have required far too much resource both in terms of code size and memory usage to be useful in small 8 or 16-bit systems. Code size of a few hundred kilobytes and RAM requirements of several hundreds of kilobytes have made it impossible to fit the full TCP/IP stack into systems with a few tens of kilobytes of RAM and room for less than 100 kilobytes of code.

Many other TCP/IP implementations for small systems assume that the embedded device always will communicate with a full-scale TCP/IP implementation running on a workstation-class machine. Under this assumption, it is possible to remove certain TCP/IP mechanisms that are very rarely used in such situations. Many of those mechanisms are essential, however, if the embedded device is to communicate with another equally limited device, e.g., when running distributed peer-to-peer services and protocols. uIP is designed to be RFC compliant in order to let the embedded devices to act as first-class network citizens. The uIP TCP/IP implementation that is not tailored for any specific application.

**BOA**

Boa[11] was born in 1991, the author is Paul Philips. It is open source, the application is widely used, especially suited for embedded devices. Boa is a single-task HTTP SERVER, unlike traditional web servers as open a process for each access connection and also it will not open more than their own copies for multiple connections. Boa deals http connections of all the activities in the internal processing, and only open new process for every CGI connection (separated process for each CGI).

The main difference between BOA and Apache[12](high performance web server) is that they are generally a single process server, only after the completion of a user request then respond to another user’s request, Boa is unable to make concurrent response, but in embedded devices of applications it is already sufficient.

Boa have minimal resource requirements. Very small memory requirements. Particularly suitable for the embedded market.
2.4.4 Protothread

ProtoThreads are an extremely lightweight, stackless type of threads that provides a blocking context on top of an event-driven system, without the overhead of per-thread stacks. The purpose of protothreads is to implement sequential flow of control without complex state machines or multi-threading. Protothreads provides conditional blocking inside C functions[13].

The advantage of protothreads over a purely event-driven approach is that protothreads provides a sequential code structure that allows for blocking functions. In purely event-driven systems, blocking must be implemented by manually breaking the function into two pieces - one for the piece of code before the blocking call and one for the code after the blocking call. This makes it hard to use control structures such as if() conditionals and while() loops.

Protothread is a thread model without use of stack and resource-intensive small which specially designed for resource limited system has the following advantages:

- Minimal resource requirements, each Protothread requires only two extra bytes
- Pure C language, no hardware dependence, so there is no transplant difficulties. Can be used with or without an OS
- Support blocking manipulation and no stack switch (protothread blocking mechanism, first the current address is stored into a variable, and then determine whether the conditions set up. If the condition is true, the program keep execute, if the condition does not hold, return.)
- Freely available under a BSD-like open source license

But everything has two sides, there is a list of disadvantages below:

- Function does not have a reentrant type, you can not use local variables, only use static variables.
- In order to determine the various task conditions are met, no preemptive priority.
- The conditions of task are determined by sequence, in this case, the requirements must show up in sequence.
2.4.5 Embedded Ethernet

Ethernet is a family of computer networking technologies for local area networks. It is a standard local area network (LAN) access method. Defined by the IEEE as the 802.3 standard, Ethernet is used to connect computers in a company or home network as well as to connect a single computer to a cable modem or DSL modem for Internet access.

A single-chip implementation of the Ethernet networking standard. It is used to attach devices such as environmental monitors, sensors and Webcams directly to an Ethernet without requiring a personal computer for connection.

Features:

✓ Ethernet is the most widely deployed network in offices and industrial buildings. Thus Ethernet is popular.

✓ Ethernet is based on standards (IEEE802.3) that ensure reliability of network connections and data transmission. This ensures interoperability

✓ Ethernet networks are scalable from the simplest to most complex networks or up to $2^{^48}$ network nodes

✓ Once equipment is connected to a Ethernet network, it can be monitored or controlled through the Internet removing any distance barrier that may have inhibited remote communication previously.

2.4.6 HTTP Protocol

The Hypertext Transfer Protocol (HTTP)[14] is an object-oriented application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of it's request methods, error codes and headers.

HTTP work on the TCP protocol in the TCP/IP protocol system. The client and server must support HTTP protocol to send and receive HTML file over WWW.

HTTP has been in use by the World-Wide Web global information initiative since 1990.

The features of HTTP protocol:

Support for B/S model. Simple and fast: the customer service request to the server, just send request method and the path. The common methods to send request are
Embedded web server remote control and communication

GET, HEAD, POST. Each method provides different types of connection between customers and server. HTTP is a simple protocol make HTTP server have a small scale of program and thus make communication faster.

Flexible: HTTP allows transmission of any type of data object. The current transmission type is marked by Content-Type.

No connection: no connection means each connection deal with one request. After the server finishes process the customer’s request, and receive the customer’s response, then disconnected. In this way, the transmission time will be less.

Stateless: the HTTP protocol is stateless protocol. Stateless protocol have no memory abilities for transaction processing. Stateless means that if the subsequent processing of information in need of front information, it must retransmit data which could lead to increased the amount of data each time customer connect. On the other hand, when the server does not require previous information the response time is less.

2.4.7 Html5. Shtml

HTML5

HTML is stand for Hyper Text Markup Language and it is a markup language. A markup language is a set of markup tags. Html uses markup tags to describe web pages. Html tags are keywords surrounded by angle brackets like <html>, and come in pairs like <html>and </html>. The first tag in a pair is the start tag, the second tag is the end tag which are also called opening tags and closing tags.

HTML5 is the new generation of HTML. The previous version of HTML is HTML 4.01, came in 1999. The web has changed a lot since then.

HTML5 is a cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG)[15].

WHATWG was working with web forms and applications, and W3C was working with XHTML 2.0[16]. In 2006, they decided to cooperate and create a new version of HTML.

Some rules for HTML5 were established:

- New features should be based on HTML, CSS, DOM, and JavaScript
- Reduce the need for external plug-ins
- Better error handling
Embedded web server remote control and communication

- More markup to replace scripting
- HTML5 should be device independent
- The development process should be visible to the public

In HTML5 there is only an <!doctype> declaration:<!DOCTYPE html>

Some of the new features in HTML5

- The <canvas> element for 2D drawing
- The <video> and <audio> elements for media playback
- Support for local storage
- New content-specific elements, like <article>, <footer>, <header>, <nav>, <section>
- New form controls, like calendar, date, time, email, url, search

SHTML

Shtml is the extension name of the html files that is SSI(Server Side Include). The SSI (Server Side Include), is similar to the ASP[17] server-based web publishing technology.

Web server processes the SSI command when process the html web page. When the Web server encountered an SSI directive, will contain the contents of the file directly into the HTML page. If the included file contains SSI commands, and then insert this file. In addition to the basic instructions for include files, you can also use the SSI directives insert information (such as file size) into the file or run an application or shell command.

At present, there are the following purposes:

- Display server-side environment variables <# echo>
- Call server side functions use mark: %!
- Call other html or shtml files mark: %!: /filename
- Display relevant information of Web documents <#flastmod # fsize> (such as file creation date / size, etc.)
- Direct implementation of the various procedures on the server <# exec> (such as CGI or other executable program)
Embedded web server remote control and communication

✔ Set the SSI information display format <# config> (such as file creation date / size)

2.4.8 JavaScript

JavaScript is the scripting language of the web. It is a client side scripting language. It was designed to add interactivity to HTML pages. JavaScript is usually embedded directly into HTML pages and an interpreted language (means the scripts execute without preliminary compilation). JavaScript is free and easy to use.

JavaScript is the most powerful tool in web technology.

JavaScript have a lot of functions, it's not only a programming tool for HTML designers have a great manipulation on HTML elements. JavaScript can be used to validate form input. JavaScript can load another page specifically designed for that browser depending on the browser. Create cookies, it can be used to store and retrieve information on the visitor’s computer.

A JavaScript location in an html file is related to execution time and method. A JavaScript inside the tag <body /> in a html file will be executed immediately when the browser encounter this code.

Eg: <html>
<body>
<script>document.write("Hello World")</script>
</body>
</html>

A JavaScript located inside the tag <head />. The script executed when its called.

Eg : <html>
<head>
<script type="text/JavaScript">
...
</script>
</head>
</html>
Embedded web server remote control and communication

JavaScript can be the same as CSS file stored as an external file with extension .js. In this way, the reusability of code increased.

Eg: <html>
<head>
<script src="scripts.js">
</script>
</head>
</html>

2.4.9 CSS

CSS stands for Cascading Style Sheets, define styles about how to display HTML elements. The style can be specified in a single HTML element, the first element of the HTML page, or in an external CSS file. Even within the same HTML document references multiple external style sheet.

2.4.10 CGI

Abbreviation of Common Gateway Interface, a specification for transferring information between a Web server and a CGI program. A CGI program is designed to accept and return data that conforms to the CGI specification. The program could be written in any programming language, including C, Perl[18], Java[19], or Visual Basic.

"CGI" is a standard protocol, not an actual implementation. However, it has become common to refer to a program which uses the CGI standard as "a CGI", and here will follow that custom.

The most common way for web server interacts dynamically with user. In order to make dynamic web page, HTML files contain forms, a form is the method submit data to server, CGI function is the method that server deals with the submitted data. Another increasingly popular way to provide dynamic feedback for web user is to include scripts or programs that run on the user's machine rather than the web server. These programs can be java applets, java scripts, or ActiveX controls. These technologies are known as client side solutions, while the use of CGI is a server-side solution.
2.4.11 Hardware Interrupt

Hardware interrupt is referred to as interrupt which indicates in the normal operation of the program, due to the arrangement of pre-selection or the occurrence of a variety of random internal or external events, causes the CPU to interrupt the running program, turn to the appropriate server program to deal with. This process called program interruption. Hardware interrupts are delivered directly to the CPU using a small network of interrupt management and routing devices. This is done to minimize CPU time, else the CPU would have to check all installed hardware for data in a big loop (this method is called 'Polling') and this would take much CPU time.

**Polling**

Polling is simply reading the button input regularly. There are a lot of other tasks running on the system. If the system keeps monitoring the port, it will cost a lot of CPU time. And also there will be delay between reads of the button. Only if the delays are small enough and the speed of the input change is slow, in that way no button presses will be missed.

If you have to do some long calculation then a button press will be missed while the processor is busy. In this case, the time is enough or not for each essential activity should be more aware of.

**Hardware Interrupt common terms**

Terms associated with hardware interrupts are ISR, interrupt mask, non maskable interrupt, an asynchronous event, interrupt vector and context switching.

<table>
<thead>
<tr>
<th>ISR</th>
<th>Interrupt Service Routine.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt vector</td>
<td>The address that holds the location of the ISR.</td>
</tr>
<tr>
<td>NMI</td>
<td>Non Maskable Interrupt - an interrupt that is always active.</td>
</tr>
<tr>
<td>Asynchronous event</td>
<td>An event that could happen at any time.</td>
</tr>
<tr>
<td>Context switching</td>
<td>Saving/restoring data before &amp; after the ISR.</td>
</tr>
</tbody>
</table>

*Table 2-3: interrupt items description*

**ISR**

ISR is a function does context switching (save/restore processor registers) before and after execution. In other aspects, it is the same as with other functions. At the end of ISR, the interrupt is reenabled.
Embedded web server remote control and communication

The program will load the original context after the ISR finishes program execution and continues from where it was interrupted. Other programs will have no idea that this has happened.

**Hardware Interrupt vector**

The Interrupt vector is the place store the fixed address of the ISR.

Program the hardware interrupt routine to the vector. Whenever the interrupt is triggered, program operation jumps to the location of the interrupt routine. After the ISR executed, the program will jump back.

**NMI**

None Masked Interrupt. The NMI is exactly the same as a normal interrupt except that you can not control whether or not it is active - it’s always active. It is more commonly found on larger (non-microcontroller) processors as a dedicated input pin.

**Asynchronous event**

This is an event that the processor can not predict, like button press. It is simply an event that is not synchronized to the system clock.

**Context switching**

This means that the register context is preserved at the start of an ISR and restored automatically at the end of an ISR. It ensures that the function that was interrupted is not affected by the ISR.

**Input sources**

In general, external hardware interrupts are mainly the following:

- I/O device: such as monitors, keyboards, printers, etc.
- Data channel: floppy disk, hard disk, CD-ROM, etc.
- Real-time clock: external timing circuits, etc.
- User false: such as power down, parity error, etc.

CPU Internal software interrupts:

- Execute the interrupt instruction INT.
- The illegal operation or instruction that caused the exception.
Embedded web server remote control and communication

**Interrupt flags**

There should be an associated interrupt flag to each hardware interrupt source. Whenever the interrupt is triggered the corresponding interrupt flag is set. In an interrupt register, these flags usually exist as bits.

The processor can interact with the interrupt register for read from and write to data. Reading from it to find out which interrupts occurred and writing to it to clear the interrupt flags.

**Interrupt mask**

There is a flow graph here to show you how Hardware Interrupt work.

```
Program -Process1
  Process2
    ......  
      process n

Interrupt happens here  process n

This contains the interrupt service routine(ISR)

Save the contents of the registers and the context of the current process

Service the interrupt task

Restore the contents of the registers and the context of the current process

Resume the original process

Process n+1
  Process n+2.....
```

*Figure 2-7: Hardware interrupt process*

**Edge trigger or Level trigger**

There is two ways to trigger an interrupt:

Level Trigger:
Embedded web server remote control and communication

✓ When the electrical signal is either HIGH or LOW, the input signal is sampled.

✓ Commonly use for latch.

Edge Trigger:

✓ At the RISING EDGE or FALLING EDGE of the electrical signal.

✓ Commonly use for Flipflop.

2.4.12 SSP

SSP

LPC2378 provide SSP to implement communication with other peripheral devices. SSP contains two operating modes: Serial Peripheral Interface (SPI) and I2C bus. 

"The SSP is a Synchronous Serial Port (SSP) controller capable of operation on a SPI, 4-wire SSI, or Microwire bus. It can interact with multiple masters and slaves on the bus. Only a single master and a single slave can communicate on the bus during a given data transfer. Data transfers are in principle full duplex, with frames of 4 to 16 bits of data flowing from the master to the slave and from the slave to the master. In practice it is often the case that only one of these data flows carries meaningful data. [20]

SPI

SPI, is the English abbreviation of the Serial Peripheral Interface, the name suggests is a serial peripheral interface. It was first defined by Motorola in its MC68HCxx series processors. The main application of the SPI interface are the EEPROM, FLASH, real-time clock, AD converter, as well as digital signal processors and digital signal decoder. The SPI is a high-speed, full-duplex, synchronous communication bus, and the pins of the chip only take up four lines, as well as to save space on the PCB layout, and provide convenient.

SPI contains four lines: SCK(serial clock), MISO(master in slave out), MOSI(master out slave in) and SS(low level active slave chip select). The principle of SPI communication is quite simple. It works in master-slave mode. Usually there is one master device and one or more slave devices, at least four lines needed to build communication, actually in one-way transmission, three lines is works.

In the peer-to-peer communication, the SPI does not require addressing operating, being full-duplex communication is simple and efficient. For the multi-slave devices system, every device need independent enable signal. On hardware level, SPI is slightly complicated than I2C.
Embedded web server remote control and communication

Finally, a disadvantage of the SPI: no specify flow control, no acknowledgment mechanism to confirm whether received data.

LPC23xx has two Synchronous Serial Port controllers --SSP0, SSP1.

SSP0, SSP1 these two cases are define their SPI Frame Format via setting the Clock Polarity (CPOL) and the Phase (CPHA) Control register. To the above two cases, set CPOL and CPHA all equal to zero which means the data will be transmit when the series clock turns to high level. See the graph below.

![Figure 2-8: SPI Frame Format with CPOL=0 and CPHA =0](image)

"For device configured as a master in this mode. CLK and FS are forced LOW, and the transmit data line DX is tristated whenever the SSP is idle. Once the bottom entry of the transmit FIFO contains data, FS is pulsed HIGH for one CLK period. The value to be transmit logic. On the next rising edge of CLK, the MSB of the 4 to 16 bit data frame is shifted out on the DX pin. Likewise, the MSB of the received data is shifted onto the DR pin by the off-chip serial slave device. Both the SSP and the off-chip serial slave device then clock each bit into their serial shifter on the falling edge of each CLK. The received data is transferred from the serial shifter to the receive FIFO on the first edge if CLK after the LSB has been latched." [22]

2.5 Similar Works

There is an embedded operating system which is called uTasker. uTasker is an OS with TCP/IP stack.

uTasker is an embedded operating system (with simulation environment) specifically targeting smaller single-chip Internet-enabled embedded processors (with internal Ethernet controller, internal SRAM and FLASH). It is small footprint
Embedded web server remote control and communication

but still offers comfortable development and powerful features as typically required in control type applications.

There is an application demo which ported uTasker into LPC-2378STK. Implement web server and control LED and LCD.
Chapter 3 Method

3 Method

3.1 Why uIP

At the background part there is a comparison between two different embedded web
below are details:

uIP has the following functions:

✓ Good documentation and source code comments, almost each line of code
  has a comment. Easy to understand and learn.

✓ The code is very small, easy port.

✓ Execute fast.

✓ Take up very little memory and can be set at compile time.

✓ Support ARP, IP, SLIP UDP, the ICMP(ping) and TCP protocols.

✓ Provide a set of sample programs: the web server, web client, e-mail
  program(SMTP client), Telnet Server, DNS host name resolution process.

✓ At the same time the number of active TCP connections is not limited and
  can be set at compile time.

✓ Free for commercial and non-commercial use.

✓ TCP and IP protocols follow the RFC standards, including flow control,
  fragment segmentation and retransmission time out estimate.

BOA got full functionality, but it is too much for this project, cost a lot resource.

So uIP was chose to be the web server of this project.
3.2 Web browser

According to the proposal, a web browser is need to be built and use C# language, so this web browser is a simple frame-work browser. At first, the reason to build a web browser is security about log in information and the right of remote control, but later found out security is not depend on the browser, it can be realized by username and compatible password. The existed web browser like IE, Firefox, Chrome are all far perfect than the one built here. Anyway this is not the important part of this project. After all, this part was deleted.

3.3 Web page

Developer kits : Notepad++

Language : HTML5, SHTML, JavaScript, CSS

One of the most important features in Embedded Web Server is Web page. Users open the web page that stored in the web server (embedded development board) through common web browser with the inputed server address and thus the user can interact with web server, like read data or send command to the web server.

**HTML**

How to build a web page? The answer must be as using the HTML language. HTML5 is the new generation of HTML language which meets this project requirement just in time. Due to the limitation of memory size, only simple html pages with basic html language were made.

There is one problem bother here for a long time. The web pages need to be interesting and pretty according to the plan, but bigger pictures can not be stored in the board. Finally, there is a way to link the pictures which published on the Internet in stead of store them on the board, in this way, there is only one line code need to be added in the HTML script.

After studied the uIP 1.0 TCP/IP stack and found out how Html works in the server side. There is a subfolder in uIP project which store the html files. But when server works they are not transmitted in html files format, there are written in the code as array.

Figure 3-1 is an example of Html script code, and Figure 3-2 is the Hex Array. This is only a quite simple example to show how it works. When the ethernet connected, log in the home page of the web server, the server will transfer this array back, according to ASCII table, change the Hex characters back to English letters. Web browsers will analysis this code and then output them as Html file.
Embedded web server remote control and communication

```c
176 static const unsigned char data_footer_html[] = {
177     /* /footer.html */
178     0x2f, 0x66, 0x6f, 0x74, 0x65, 0x72, 0x6f, 0x66, 0x69, 0x6e, 0x61, 0x6c, 0,
179     0x20, 0x20, 0x2c, 0x2f, 0x62, 0x6f, 0x66, 0x69, 0x6e, 0x61, 0x6c, 0x69, 0x6e, 0x67, 0x72, 0x69, 0x6e, 0x67, 0x2c, 0x30,
180     0x3c, 0x62, 0x6f, 0x66, 0x69, 0x6e, 0x61, 0x6c, 0x69, 0x6e, 0x67, 0x72, 0x69, 0x6e, 0x67, 0x2c, 0x30,
};
```

`Figure 3-2: footer.html file in Hex format`

**JavaScript**

In order to make the web page more vivid, there is some animation added to the webpage using JavaScript. JavaScript was written inside the tag `<script>` that embedded in HTML files. The JavaScript was used to implement the animate navigation of the web page.

```javascript
5 <script type="text/javascript">
6     var i=-335;
7     var intHide;
8     var speed=10;
9     function showsmenu()
10     {
11         clearInterval(intHide);
12         intShow=setInterval("show()",20);
13     }
14 </script>
```

`Figure 3-4: script function example`

**SHTML**

There is also have another file called Shtml. You can find the explanation at the Background area.

Website maintenance often encountered a problem that the structure of the site has been fixed, but when the developers want to update a little content they had to redo a large number of pages which cost a lot of manpower and resource. SSI provides a simple, effective way to solve this problem, it will place a basic site structure which do not need update often on a few simple HTML files (template), separated them from the frequency updated content. The site structure files can be called in other
Embedded web server remote control and communication

files. When update content, developers only need to update the specific part without changing whole web pages. In this condition, the reusability increased and make website maintenance more efficiency.

There is another reason. Because of the limitation of memory size of embedded system, if the less important code is smaller, it would be better.

The most important feature of SHTML to this project is that it can call Protothread format CGI function.

The first line and the last line of the Figure 3-5 shows the HTML file which called here. The last second line is showing function call. "tcp-connections"is a Protothread format CGI function.

```
%! : /header.html
<h1>Current connections</h1><br> <table width="100%">
<tr><th>Local</th><th>Remote</th><th>State</th>
<th>Retransmissions</th><th>Timer</th><th>Flags</th></tr>
<tr><th>tcp-connections</th></tr>
%
% ! : /footer.html
```

Figure 3-5: shtml file example

3.4 Hardware interrupt

In this thesis project, button and led control are necessary. There are two ways either by Polling or by using Hardware Interrupt to implement read a push button which are connected to one pin. In order to choose one way, the most important part is figure out exactly how it works and the advantages and disadvantages.

Hardware Interrupt or Polling

Polling:

- Takes CPU time even when no requests pending
- Overhead may be reduced at expense of response time
- Can be efficient if events arrive rapidly

Hardware Interrupt:

- Processor time is used efficiently and not wasted
- Fast response
Embedded web server remote control and communication

In a metaphor way: “Polling is like picking up your phone every few seconds to see if you have a call. Interrupts are like waiting for the phone to ring.”[23] Obviously, you will choose hardware interrupt.

Considered all the situations Hardware interrupt was chose, may be it is not easy to do, but when it worked out, there would be benefit to this project.

3.5 SPI

The microcontroller of the development board (LPC2378-STK) here has two Synchronous Serial Port controllers -- SSP0 and SSP1. There are two communications need to be built. One is communication between LCD and MCU, another is realize data transmission between nRF24L01 radio module and MCU.

Even these two communications are all built on the base of SSP interfaces, but there still exists some differences between them:

In the Radio-MCU data transmission, the connected Radio module is selected as the only slave and the microcontroller as master. With using 5 specific pins and use PINSEL set them in a appropriate way which shown below to realize the transmission.

In order to fulfill the requirements, SSP1 was chose to build the communication for Radio-MCU.

On the other hand, the communication between LCD and CPU is a little bit different from the case above.

Still use some specific pins to control and realize the transmission, but instead of 5 pins, here three pins- SCK0,MISO0,SSEL0 being used. This limits the master just can write the data into LCD but can not read data from its buffer which suitable for this situation. In this project, the LCD shows the data transfer back from Radio module to MCU. For these three pins, their use pattern is almost as same as the corresponding pins which used in SSP1 interface communication that already listed above. In this case, SSP0 is the best choice to build communication between LCD and MCU.

By the way, the SSP0 the first case does not have reset pin.
Chapter 4 Result

4 Results

4.1 Web server

uIP is selected as web server for this thesis project. With this web server, there are a lot of knowledge come clear to us like how to build web server in embedded system, how does the web server deal with web pages, how to write data into LCD, how to implement hardware interrupt, what does CGI function looks like, how to build wireless communication, etc.

uIP web server is a TCP/IP stack which only implement the basic communication part of TCP/IP protocol, it is the so-called "thin server", but this is enough for this project. In uIP, CGI function was written in Protothread. "The purpose of protothreads is to implement sequential flow of control without complex state machines or full multi-threading."[5]

uIP stores the HTML file into its code by translate them into Hexadecimal characters array according to ACSII. The server will translate these characters back to HTML file before transfer back to web browser.

4.2 Web page

Web page is the interface for user to access server. The first impression user get from web page is very important, if its not good, maybe the user will not continue visit the web site. On the homepage of the project, a cartoon background is applied. The Embedded technology may give a boring impression to normal people who have no idea about this project. So there must exist a fun and interesting way to attract users’ attention. The word on the picture is "Intelligent world" which associate with the theme of the project. The navigation is made into an animation. When mouse hover over the navigation bar, it will slide out and user can choose the items in the navigation bar. When the mouse moves to any other place, the navigation bar will slide back.

Because of the limited size of embedded system, the pictures on the web page are all coming from Internet.
Embedded web server remote control and communication

The basic knowledge of build web pages are HTML, CSS scripting language. Considering the actual situation of this project, the JavaScript and SHTML are applied. Figure 4-1 and Figure 4-2 is the welcome page for user.

Most of the web site is based on this background and navigation. So these content is not changing with the switch of different web pages, so they are stored in the file "header.html", "footer.html" files. "header.html" file contains the background and navigation bar. The "style.css" file is necessary for the format of the content.

The navigation bar can slide out(Figure 4-2) and in according to the mouse move. This animation is implemented by JavaScript. The JavaScript is written in functions base on Java language. Embedded these functions inside tag<head /> <script /> of HTML files. The script wrote in <head /> tag is executed when they were called. The source code are enclosed in Appendix item.

Figure 4-3 show the main application of this project---Mobile robot control. On the interface there are four buttons which indicate four directions. When the button pressed, it will send command to radio module and move the pie.

This application implemented in the file "ControlPIE. shtml". This is SHTML files. For the file contained "header.html" and "footer.html " to display the basic layout of web site. Syntax of include other files is "%!: /header.html". It is server side language, the server translates this language into HTML language and then transfer them back to web browser, web browser interpret them and show out. The white box part is new content here. Every button on the page call a CGI function which is written in protothread format. This protothread is single-threaded. If one protothread is executed others can not be processed until the current one stop. For example, the left button call function "%! go-right". The other three are implemented in the same way.
Embedded web server remote control and communication

*Figure 4-3*: ControlPIE.shtml and the screen show the server send command "L" to the slave and receive data from the slave Radio module.

This Figure4-4 and Figure4-5 show the procedure control LED. The background changed to make this process more vivid. The way to achieve this application is the same as ControlPIE.

*Figure 4-4*: ControlLED, green button turn on LED, red one turn off LED and onboard led is turn off now

*Figure 4-5*: ControlLED, when the led is lightened

Figure 4-6 and 4-7 shows the information about this project. Figure 4-6 is the description of the project and 4-7 is show the designer information.

*Figure 4-6*: description of our project

*Figure 4-7*: Designer information
4.3 Led blinking

This function is quite simple. In Figure 4-8 Line 33 code is used to configure the pin of led as output also means the led is off now, line 37 configure led at the state of on. Here the flag change to toggle led.

```c
28 void Toggle_LED(void) {
29  static int myflag = 0;
30  if(myflag){
31    USB_LINK_LED_POLA = USB_LINK_LED_MASK;
32    myflag = 0;
33  }
34  else{
35    USB_LINK_LED_PBRT = USB_LINK_LED_MASK;
36    myflag = 1;
37  }
38}
39}
```

*Figure 4-8: toggle-led function*

4.4 Hardware interrupt

![Diagram of VICADDRESS and Interrupt handler address]

When the interrupt happens, the address of the specific interrupt will be copied to the VICADDRESS, then the Interrupt handler(ISR) will be executed.

*Figure 4-9: interrupt*

In Figure 4-9 the VICADDRESS register contains the Interrupt Service Routine(ISR) address of the currently active interrupt. If no interrupt is currently active, the register holds the ISR address of the last active interrupt. This register can be accessed with zero wait states. A read of this register returns the address of the ISR and set the current interrupt as being serviced. A read must only be performed while there is an active interrupt. A write of any value to this register will clear the current interrupt. A write must only be performed at the end of an interrupt service routine.

Before this work is necessary to do some configuration and it is a little complicate.

In order to implement use button to control the led, hardware interrupt was chose. The pins which indicate buttons should be configured first. On the base of schematic,
P0.29 is button1, P0.18 is button2. According to Figure19, bits 26, 27 are used to set up button. Figure 4-10 and Figure 4-11 shows the button configuration.

![Figure 4-10: pin select for button 1][21]

According to background, EDGE trigger was chose here to trigger interrupt. Figure 4-12 show use EXTMODE to set up.

![Figure 4-11: pin select for button2][21]

![Figure 4-12: external interrupt mode register][21]

**EXTPOLAR configuration(Figure4-13):**

![Figure 4-13 : EXTPOLAR configuration][21]

According to the datasheet for LPC-2378, in order to achieve this goal, (External Interrupt) EINT3 is chose to be the interface. Figure 4-14 shows the detail about EINT3 definition.
Embedded web server remote control and communication

Each bit in these read-write registers enables the falling edge interrupt for the corresponding GPIO port pin. Enable P0.18, P0.29 for button as output. Show in Figure 4-15.

Until now, the prerequisites of button hardware interrupt are finished basically. Inside the ISR, there is a register to store the current status of interrupt, if the particular bits in the register are changed, it means interrupt is triggered.

4.5 Wireless communication and LCD display

Wireless communication is a series of dynamic process. It is hard to show the essence of the results in a few pictures.

Here are a few pictures to show the outline first, details would come later.
Embedded web server remote control and communication

Slave board
When powered up. The green led is blinking means the system is running now and ready to receive data.

Master board
Screen shows the current communication status. Right now no command sent from the web page, so it shows "waiting send data". The second box show receive status, cause slave is sending data, it shows received :nrf_type_HB. The left corner led is blinking

Figure 4-17: system start

Slave board
When received data, yellow led blinking. start send data- heartbeat. The green led is blinking.

Master board
Screen shows the current communication status. Right now it shows "Sending 'F' out". The second box show receive status, it shows "received :nrf_type_HB". The left corner led will blinking during system running

Figure 4-18: user send command
Embedded web server remote control and communication

PIE
Actually the PIE is moving now, unfortunately, it can not be seen from a picture.

*Figure 4-19: move the PIE*

Wireless communication implementation

The device LPC-2378STK has two interfaces to realize wireless communication with radio module. They are SSP1 and SSP0. After analysis, the details about comparison are placed at the Method part. Here the SSP1 is used to implement wireless communication. SSP1 works in SPI mode here.

The principle of SPI communication is quite simple. It works in master-slave mode. Usually there is one master and one or more slave device in this mode. The four lines to build communication is: SCK, MISO, MOSI, CS.

CS indicates if the chip is selected, that means only the chip select signal is the predetermined enable signal, the operation to the chip will be effective.

The next three lines is for communication. Communication is completed through data exchange. SPI is a serial communication protocol, which means the data transmitted one bit by one bit. This is the reason for the existence of the SCK clock line. The MISO, MOSI transmit data based on the clock pulse generated by SCK. Data output by the MOSI, the data changed in the clock rising edge or falling edge, followed read by falling or rising edge. In this way, at least eight times the change of the clock signal will complete the 8-bit data transmission.

It should be noted that the SCK signal line is controlled by the master device, slave device can not control the signal line. In a SPI-based device, there is at least one master. One advantage of such transmission, it is different from common serial communication, the common transmission transmit at least 8-bit data continuous in one time, while the SPI allows the transmission of data one bit by one bit, and even allows to pause. SCK clock line is controlled by the master device, when there is no
**Embedded web server remote control and communication**

clock transition, the slave device does not collect or transmit data. In other words, complete control of the communication through the control of the SCK clock line. SPI is also a data exchange protocol: SPI data input and output lines are independent, thus allowing the completion of data input and output simultaneously.

Figure 4-19 is come from oscilloscopes that show the wireless communication is succeeded. The underline is SCK- clock signal generated by Mater device. The upper line is MOSI, which is data signal.

![Oscilloscope Image]

*Figure 4-20: picture from oscilloscope, communication built*

**LCD display**

Write data into LCD is the communication between LCD and MCU which use SSP0 to complete data transmission. This communication is quite familiar with radio communication. The only difference between them is here only use 3 lines-SCK, MOSI, CS. Because only input data to LCD, no output data from LCD. Figure 4-20 show data on the screen.
Embedded web server remote control and communication
Chapter 5 Conclusion

5 Conclusions

The results of research project show that it is possible to use LPC-2378STK as a web server combined with radio module to realize remote control robot and wireless communication.

5.1 Result Demonstrate

It is time to make a conclusion. In order to make the structure more logical, here is a list to do a comparison between Specific goals and the achievements until now.

This project is aimed at realize remote control robot-PIE. All work before this is preparations. The mandatory tasks are all implemented and finally the robot can be control. There is only one bonus task to set different user accounts is not achieved. But this is not affect the final result and this is extra task when time available.

<table>
<thead>
<tr>
<th>Specific goals</th>
<th>Achievements</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Build a web browser (PC) to show the web page from the server (remote board).</td>
<td>1. After discussion with supervisor , then come to an agreement that this part is meaningless to this project. So it was cut off.</td>
</tr>
<tr>
<td>2. Implement TCP/IP stack on the development board.</td>
<td>2. Web server:ported uIP TCP/IP stack into LPC-2378STK as web server</td>
</tr>
<tr>
<td>3. Use the LEDs (on the board) to signal the process.</td>
<td>3. LED blinking: dynamically indicates system processing</td>
</tr>
<tr>
<td>4. The client can read values of the server and show on the web page.</td>
<td>4. Web page: interface between user and Embedded Web Server and other applications</td>
</tr>
</tbody>
</table>
5. The boards which join the remote communication as a server could show part or all of the data received from the client on its LCD.

5. LCD display: show sent data and related information about communication condition

6. Implement drivers for the radio module.

6. Hardware Interrupt: use the buttons on the board control led on the board which indicates the hardware control is achieved

   Wireless communication: communication between two radio

7. *(bonus part)* Via setting accounts, defines different multiple role groups and makes a finite set of rules to assign user to roles. Login with different restrictions like administrator and student account.

8. *(bonus part)* Implement communication with the mobile robot --- “PIE”

7. robot control: send command via web page to move PIE-the robot

<table>
<thead>
<tr>
<th>Table 5-1: comparison between specify goals and achievement</th>
</tr>
</thead>
</table>

5.2 New

In traditional embedded web server application. Limited by memory space, web pages always made in a simple way. Although it is not affect the functionality of the system, it is just not that attraction. Users may be complaining about the boring interface.

In the project work, the picture on the web page are all come from Internet which means they are not stored in the web server, but cloud storage. In this way, the resource of system is not occupied and the web page got more interesting.

There is another technology applied to the web page. Embedded Javascript in the html file to build animation on the web page.
5.3 Experience

The web server in the project is build on uIP, an open source TCP/IP stack. The implementation of the whole project proved that uIP is relative easy to use for newbie.

At first, it is hard to read datasheet to find useful information efficiency. But time proved, practicing is the way to solve this problem.

The SPI interface clock configuration experience:

When configure SPI clock at master side the requirements from the slave device must be figured out. Because the master side clock polarity and phase are based on the slave device. Therefore, the slave device receive data on clock rising or falling edge must be clear when configure master clock polarity. But this should be noted that the MOSI of master device connect with MISO of the slave device, slave device MISO connect with master device MOSI. The data received by MISO of slave device is sent by master MOSI, master device MOSI receive data from slave MISO. So the configuration of master clock polarity is opposite from the slave side. Only with the correct configuration of clock polarity and phase, data can be send and receive accurately. It is necessary to compare the slave SPI timing or follow the description of datasheet to correct configure the master clock.

5.4 Future plan

The whole achieved results are almost meet expectation. But to a future work, there is still a lot to optimize and develop. Security is always the big issue in Internet. There is one bonus task left behind because of time issue. The original plan is to setup administer account and client account to put limitations on user permission. Embedded remote control pays much attention on security.

At current level, it is monopoly to control the system in one time, multi-users is not supported. In future, this can be fixed.

There is a camera to monitor PIE condition in the IDE laboratory. So user can detect remote control via this video camera. The window of video camera can be embedded in the web page. And also take pictures from the camera, use MATLAB analysis picture to locate PIE.

Real-time is a feature of embedded system. Especially remote controls have high demand on instantaneity. There is a big room for improvement.
Embedded web server remote control and communication
Chapter 6 Discussion

6 Discussion

6.1 Statement

During the thesis time, we are trying our best to do a good work. The results are meeting our expectation except the User login part. "There is no best, only better." There are a lot can be improved, but until now the completed part is a good work.

6.2 Weakness

One time one user

At the same time, only one user can control the system.

Security

Without user limitation, anyone who knows the url of the system can control it.

Unreliable communication

The communication between server and PIE is based on Radio Frequency, basically no protocol ensure the communication.

Time latency

User start control PIE, when button clicked, server start send command to PIE. The data will be send bit by bit without stop to make sure PIE can get the message. PIE send HB to server indicates it is active instead of send acknowledge signal back.
Embedded web server remote control and communication
Chapter 7 Reference

7 Reference


[5] Shtml http://httpd.apache.org/docs/2.0/howto/ssi.html#basic


[10] uIP 1.0 web server --TCP/IP stack http://www.sics.se/~adam/


[17] ASP script language http://www.asp.net/

[18] Perl programming language http://www.perl.org/
Embedded web server remote control and communication

[19] JAVA http://home.java.net/

[20] Page 466 LPC23XX datasheet


[22] Page 468 LPC23XX datasheet