# Development of a reconfigurable Embedded Web Server.

Krerk Piromsopa, Boonchai Sowanwanichakul.
Department of Computer Engineering,
Faculty of Engineering, Chulalongkorn University,
254 Phayathai Road, Patumwan,
Bangkok, Thailand 10330.
Tel. (662)-218-6956, Fax. (662)-218-6955
g41kpr@cp.eng.chula.ac.th

**Abstract**

As the Internet continue to grow, the number of devices and appliances connect to Internet are increasing. Consequently, the web server is developed to embedded with these devices for access, monitor and control. It's essential that this web server be Reconfigurable to make it function with any devices or appliances. Scripting Language such as javascript is used to customize, manage and reconfigure the system in order to integrate with different environment. This paper provides the issue of how to create an embedded web server box, the design requirements by using Intel MCS-51 the 8 bit microcontroller as a main processor with the power of sever-side script and Password Authentication.

## 1. Introduction

To obtain data from an embedded devices or appliances can be difficult. Traditionally, the data has been transferred through dedicate serial port. That means the numbers of devices connected is the more serial I/O interfaces require. If the terminal supported graphics, it might also be necessary to write a graphical interface; otherwise the data would dump out as straight text. By using an embedded Web server, developers can format and display the same data with HTML though any standard browser. Moreover, communication can be use Ethernet and HTTP can handle the transfer of larger amounts of data to any device on the network.

The design of embedded systems is the state of art computing system. It's the meeting of Qualities of service and Pricing. In the other word, the computer can control the cooking program in your microwave oven but it's not reasonable to use the high performance mainframe computer as the small and easy using microcontroller one. As a result this work will use the MCS-51 family from Intel which is widely use in the industry and appliances. In contract, the small 8bit microcontroller may be powerful enough for control simple device but it might be too slow to directly connect to the Ethernet network. This way, a network-interfacing controller is required. (As every systems need a dedicate NIC Adapter to connect to the network)

While each vendor try to develop their own web server for embedded into their device, the idea of how to create an Open Source Embedded Web servers is presented by GoAhead Software. Once the Open Source Embedded Web server is used, the developer have to modify and recompile the source code to make it function with their desire hardware. This is a strong drive for create the Reconfigurable embedded web server that is working as a component. This way, if the developer want to connect their device to the web, they can place this component as plug-in module and write some server-side script to satisfy their work.

In order to let the embedded web server gain an ability of reconfigured for functioning in any environment, the scripting language is the must. Scripting languages have proven particularly adept at integration applications, where new functionality is layered on top of existing components and resources rather than built from scratch. Like this, if the simple embedded web server is being built, it can be configured to working with any device as it'll connected through the I/O port of the microcontroller. Moreover, once useful resources were made available, Thus a strong security system was needed. That's the means of security authentication that should be also implemented to the systems. The very simple but powerful is the using of what you're known. (the password) as an authentication system.

This paper will take a look at the role and implementation for create and use a Reconfigurable embedded web server. The standard and protocol that must be meet and how to keep the system more secure with basic web password authentication.

## 2. Hardware Design Considerations.

The embedded web server is named as a thin server. With all the function that should be support in order to services, the server requires a communication channels with the client, and must provide enough memory for storing the web pages. Consequently, the network interface and Memory will be considerate in this section.

Network the embedded devices, cannot be easily done as the Personal Computer which require only a network adapter to plug in to the mother's board. Since the network interfaces adapter for PC usually comes in the standard bus such as ISA or PCI bus. Which is not support in the small microcontroller. The Alternative way is to build the LAN adapter myself. First step in creating a LAN adapter is selecting the Network Interface Controller. There are several Network Interface Controllers (NIC) from various vendors. Some of them support 100Mbs Ethernet. After take a look at each NIC, they can be classified by the bus width. In order to functioning with MCS-51 microcontroller, the 8bit bus width is chosen. The NSC DP83902A is selected since it's the very common compatible with NE2000 and the packet driver is easily ported to the MCS-51.

RAM is not only functioning as a data memory for program, but also must be acting as File Systems for the web server. As the common web server usually store information on the base of file and directory structure, It should be easy if the thin server handle it in the same manner. However, it's not quite a good idea to add a disk controller to the systems. The better solution is emulating the file systems using memory. (As if using in Window CE) The battery packed RAM is seeing more and use as an alternative to the more traditional non-volatile memory devices (As describe in Memory Interfacing and Architectures for Embedded Systems. [1]). Anyway, the MCS-51 address is limited to 64Kbyte. Most of them are using as variable and buffer. Consequently, there is not enough address for storing the big file system. This problem is also found on the old Apple computer that is limitation in memory address but still function with CPM by sliding the memory in to small bank. Each time using the memory, the preferred memory bank must be selected first.

Due to the only support of memory mapped I/O, both Memory and I/O must be connected to the same system bus with the different address mapped. The overview of how each component is connected is show in figure 1.

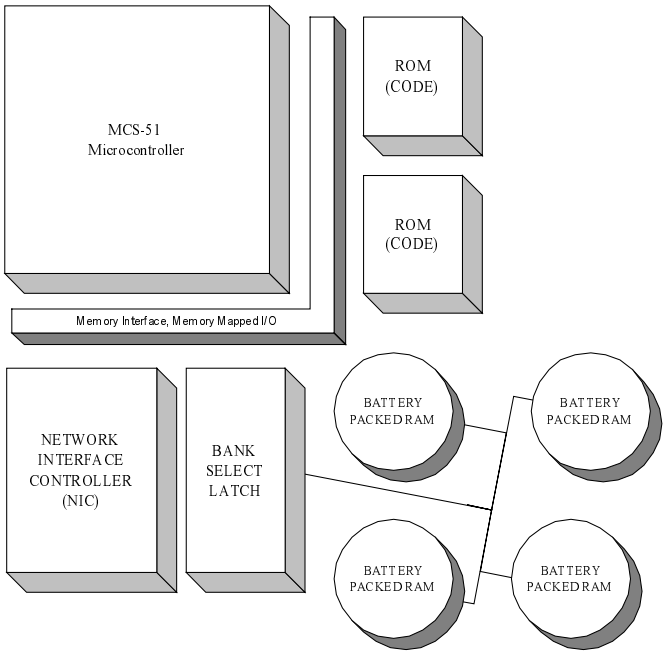


Figure 1 Hardware Block Diagram

## 3. Protocol and Standard Related

To construct web server, There are many protocol and standard that must be meet to make it functioning. Firstly the network standard will be considerate. Next, we will take a look at how to make web server functioning on the top of network protocol.

***IP***

Internet Protocol is the heart on the Internet. The current version of IP is 4.0. How to make the IP function is very simple, Since the IP is require only to identifying the network address. Which require only a little checksum function.

***ARP***

Address Resolution Protocol is the mechanism that make the IP protocol function on the top of Data link Layer, especially on Ethernet network. The goal of ARP protocol is to map the IP Address with the Ethernet MAC Address. ARP helps the client to find the true MAC address of the server. For example, when the client wants to send a Datagram to the server. They will first check their own ARP table if the MAC address of destination IP is known. If do so the out going Datagram is sent. In the other hand, the ARP request packet will be broadcast to ask for the MAC address of the destination IP. The server has to send the ARP response if the incoming ARP request's IP is match with the server IP address.

***ICMP***

Internet Control Message Protocol usually comes with the kernel of the Operating System. There are many functions that served by ICMP. The Purpose is for the host to exchange the error messages. This function is required for the embedded web server as a result of the way that each node use to check if the other node address is alive (called PING). Also, a number of ICMP message type is shown in the standard. But the only one that must be

implemented is the PING response. The others can be ignored due to the space limitation of embedded systems.

***UDP***

User Datagram Protocol is a simple, Datagram-oriented, transport layer protocol. UDP provides no reliability. It sends the Datagram that is requested application to the IP layer, but no guarantee for reaching the destination. The UDP is useful to send the emergency from the web server to the other host when there are any error on the embedded system. The implementation is not necessary but since you try to build TCP, the UDP is on the way ready to service for you.

***TCP***

Transmission Control Protocol is an connection-oriented protocol. Before either end can exchange data, a connection must be established between them. Like the UDP and IP, TCP also have a checksum property. The different between the checksum from UDP and TCP is that the one from TCP is mandatory. TCP is used to handle the data between client and server. The point that TCP is being used by various applications is the abilities to guarantee the correct and successful data transfer. Moreover, TCP can be fragmented with the powerful sliding window. But to implement the fully support for the TCP Datagram. It's might be too complex and some function is useless for embedded system. In this prototype some part of TCP is being simplified to gain a benefit of easily implementation.

***HTTP***

HyperText Transfer protocol is the engine of the Web server. The main function of HTTP protocol is to provide the data transfer between the client (Browser) and the server. The HTTP can be divided into 2 parts. First is the Header that will carry the request and the response that description the content of the data in the following part. The Second part carry the data, which can be any type of documents. Ex. Images, plain text or binary stream that can be seen in MIME. But most of the document type is the HTML. However, HTTP is running as the application service on the top of TCP/IP network. On the server based web server HTTP will work as an application that transfer the data located in the file system to the network. In the embedded web is view is different. The requirement of the embedded web server is to acquire data from devices as well. We will describe later on how to acquire and setup the device using the scripting language.

***4. Web Authentication***

Authentication is one from the three methods in create security systems. The others are the Authorization and Accounting, which are differences in each application and should be binding at the runtime (Defined in Project Athena). As describe, the only thing that web server can be done to give the security is the Authentication.

To authenticate, the HTTP is being extended using methods called basic and digest. The basic authentication provides the simple authentication with base-64encode. While the digest authentication is useful for a complex security system. But so far, difficult to be implement on the small embedded system. As a result the basic authentication system is being implemented. And give the scripting language ability to handle the Authorization and Accounting by passing the parameter as a variable to the script process.

**5. Embedded server-side scripting language.**

Scripting language have proven particularly adept at integration applications, where new functionality is layered on top of exist components and resources rather that built from scratch (John Ousterhout says in the Embedded System Conference on "How Scripting Adds Value to Embedded Systems"). In the server based web server, the server side script is using to bind WebPages to the application in order to reduce the overhead of CGI method. Embedded Web server requires the same theme. Various scripting languages are being reviewed to meet the requirement of web server.

Active Server Pages (ASP) is a Microsoft developed approach to allow the easy creation of dynamic Web pages. The script language can be selected at runtime by specifying the desired language. However, multiple-scripting languages would be rare suitable for embedded web servers. (Michael O'Brien, 1999)

JavaScript has gained widespread attention as a leading scripting language, but its large memory footprint (200-400K) prevented its use in embedded applications. The idea of creating a strict subset of JavaScript called "Embedded JavaScript" is introduce by Michael O'Brien from GoAhead Software. The resulting implementation is a 15K embedded JavaScript interpreter that is enough for embedded application

The Embedded JavaScript consist of global function, global variable, conditional control, loop, simple operation and comments. This feature provides a powerful to create the dynamic web application for embedded system.

**6. Software Implementation**

The system software part of the Embedded web server can be section into 2 parts. First is the, Network Connection. The other is the Device and I/O Handling. Both of them can be called as "Embedded Kernel". The applications that service on the top of embedded will ask the kernel to create, connect, store and retrieved information through the HTTP protocol. (As shown in figure 2)

The network function is consisting of ARP that is working in the same level as IP. The incoming packet from the Media can be only IP or ARP. Other packets that the encapsulation data is not ARP or IP Datagram will be reject. In case of IP, the MAC address of incoming packet must match with one of the Network Interface Adapter. Otherwise it'll be ignored too.

After passing the IP checking, packet can be identified as ICMP (Ping request), TCP, and UDP by checking at the IP Header. If the incoming packet is not in any of these three types, It's recommend to invoke no process, as they are not necessary to be implemented. In the case that incoming packet is classified as ICMP Ping request, the process only just answer by sending the ping reply immediately. Unlike ICMP, UDP packet will be pass to the application with out checking anything. And the answering Datagram is depending on the application. If the packet is identified as TCP, there must be method of managing and maintaining each session. Since the connection must be synchronization and acknowledge.

The File system management driver must manage the file systems that are based on the Battery packed RAM. As the memory is being sliding to small bank, It's necessary to have a file allocation table as if there are in the floppy disk. The FAT style file system is being implemented. When the Application request for file access the driver then look at the allocation table to checking for the filename and where they are exactly store. Then provide a cursor pointing at a file for read and write.

The web server is working on the top of file system handler and the TCP network. They receive data from network then checking and involve a data from file system that must be process through the Server Side Script Interpreter then send back to the Network. As there are an Interpreter, this means the small lexical analyzer and parser is implemented as a module of web server function.

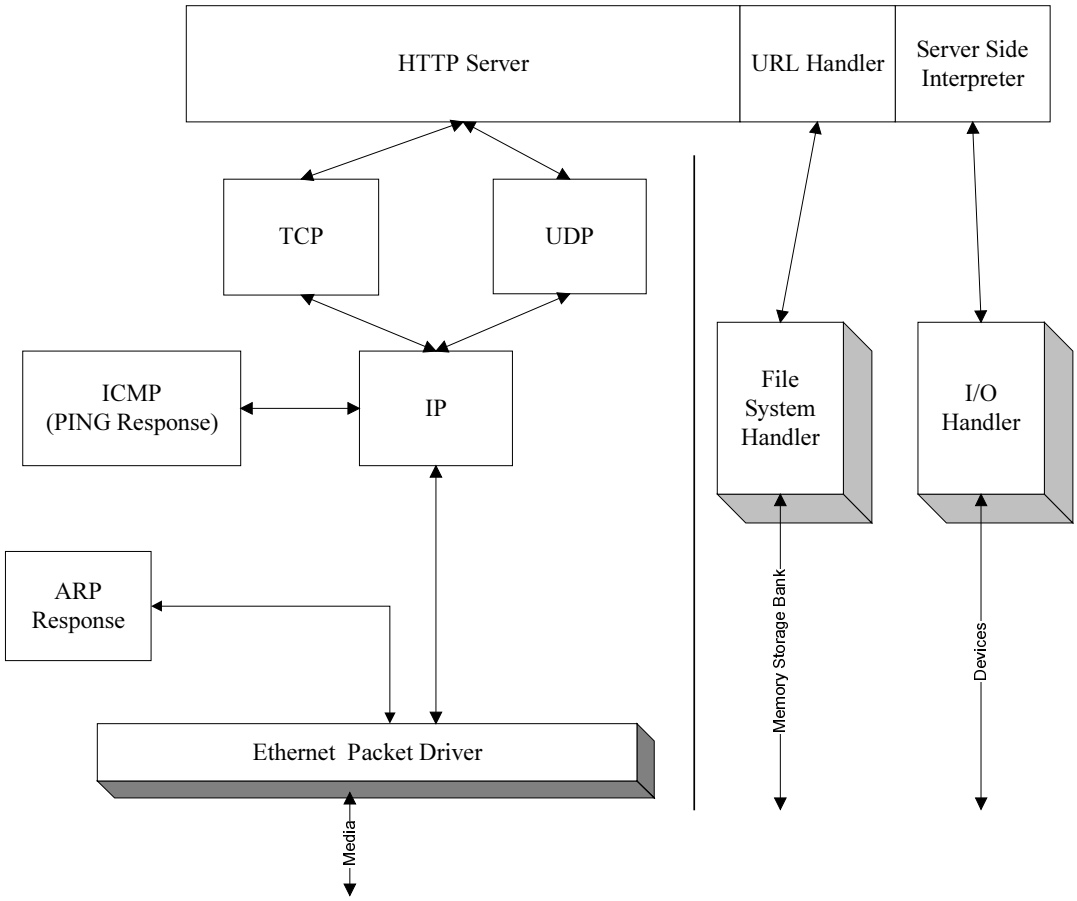


Figure 2 The software component in embedded web server.

**7. Web Programming.**

To program or reconfigure the web pages to dynamically function with any application, the server-side script is involved. As the server-side script usually content the standard HTML mixed with scripting language. There must be a tag to identify as if the preferred text is script or HTML tag. The ASP style of script is implemented. When the user want to write the script they must put it in the "<%" and "%>" tags.

The server will provide the standard function of JavaScript function such as read or write. With the special function to interface to the device (I/O). More over the user may create their own variable, which will always be the global variable. Or even create their own defined function with a few parameters.

Example. The Server side javascript on how to loop and output the value of I/O port.

<B> The Port Value of Server is </B>
<%

```
        for (I=0;I<10;I++)
                write("The value of port[" + I + "] is" + getport(I) + "<BR>");
%>
```

**8. Application, Future Vision and Development**

Reconfigurable embedded web server can be useful in many situations. Since the power of web protocol is an easy way to create the user interface and remote data retrieval. Every devices and appliances should be embedded with the web server. As it'll provide the Open system and information sharing. For example, if the fax machine has a web capability. The whole office can use only one fax machine. The users in the difference part of the office can see the incoming document through the web. The other instance is the telephone answering machine. Suppose that you are away from you home and want to check if there is anybody making a call to you. Then you just dial your notebook to your home and see the answering machine status or even listen to the message.

It's been predicted that every device and appliance should be running over IP protocol. To archive this goal, an embedded web server and TCP/IP should have an open standard as they were in the Server based web server.

**9. Conclusions**

As the number of devices grow up everyday. It strongly forces to network the devices as the benefit of distributes the information and management. To add the web interfacing to device or appliance, people can utilize the information easier.

As the embedded web server can be reconfigure to function in any environment, The developer can easily connect their exist devices to the web with a little modification of server side scripting language without creating the whole system of their own.

**References**


[1] Joseph L.Long, "Memory Interfacing and Architectures for Embedded Systems," Embedded System Conference, Class 405

[2] D. C. Plummer, 1992. "An Ethernet Address Resolution Protocol," RFC 826, 10 pages

[3] H. Frystyk, R. Fielding, and T. Berners-Lee, 1993. "Hypertext Transfer Protocol – HTTP/1.0," RFC 1945, 60 pages

[4] J.B Postel, 1980. "User Datagram Protocol," RFC 768, 3 pages

[5] J.B Postel, 1981a. "Internet Protocol," RFC 791, 45 pages

[6] J.B Postel, 1981b. "Internet Control Message Protocol," RFC 792, 21 pages

[7] J.B Postel, 1981c. "Transmission Control Protocol," RFC 793, 85 pages

[8] H. Frystyk, R. Fielding, and T. Berners-Lee, 1993. "Hypertext Transfer Protocol – HTTP/1.0," RFC 1945, 60 pages

[9] H.Frystyk, J. Gettys, J. Mogul, R. Fielding, and T. Berners-Lee, 1997. "Hypertext Transfer Protocol – HTTP/1.1," RFC 2068,162 pages

[10] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A.Luotonen, and L.Stewart, 1999. "HTTP Authentication: Basic and Digest Access Authentication," RFC 2069, 34 pages

[11] Steve Wingard, Spyglass, Inc. "Embedding HTTP Functionality for Web-Based Configutaion and Management of Devices," Embedded System Conference, Class 460

[12] Michael O'Brien, GoAhead Software, "Open Source Embedded Web Servers," Embedded System Conference, Class 407