ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 183-188 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

Implementation of Light Weight Internet Controlled Web Server in Embedded Systems

[1]Prof.P.Rama Bayapa Reddy **[2]Dr.K.Soundararajan** **[3]Dr.M.H.M.Krishna Prasad**

[1]Professor, Dept. of CSE, Dhruva Institute of Engineering and Technology, Hyderabad, India,
[2] Professor, Department of ECE, JNTUACE, Anantapur, A.P, India
[3]Associate Professor of CSE, Head, Dept. of IT, JNTUK Vizianagaram Campus, A.P, India

Abstract*:* This paper presents an implementation of a plat form independent embedded web server and its integration into a network of wireless sensor nodes. This allows the user to monitor the operation of the WSN remotely, to periodically download the sensed data, and to change the operation mode of the network. In addition to providing monitoring and data collection services, the embedded web server can generate email alerts about critical issues in the WSN, provide secure access to modules that change the operation of the WSN, shut down sensor nodes, and log data from the network into an on-board flash memory. This paper describes a development of a (LWWS) lightweight web server that runs on limited hardware resources of an embedded system. The web server hosts static web pages, enables HTTP file transfer and supports input/output communication. The embedded system is based on the Microblaze softcore processor implemented on a FPGA platform. The web server application runs on the uClinux operating system. The embedded web server is designed and built as an expansion module for one of the nodes in the wireless sensor network (WSN). It allows authorized Internet users to establish two-way communication with the sensor network. The server uses limited available hardware resources to implement an interface to the WSN node and to serve dynamic HTML pages to the remote user.

Key words: HTML, WSN; Embedded internet server; remote monitoring and control; HTTP

I. INTRODUCTION

Embedded systems are specialized computer systems designed and optimized to perform a particular task. Usually they are a part of a larger system or a machine [1]. The Embedded Web Server Technology is most evolving technology for Internet Devices. There are many application areas including internet devices, telecommunication devices, measuring instruments and lots of consumer electronics. WEB-enabled systems have offered great promise to education and science, businesses, and consumers. Traditional Web servers are designed to serve static Web pages from high-end workstations with plentiful CPU and memory resources. Embedded Web servers have different requirements for which traditional technologies are unsuitable. The server is the repository for the Web pages, and it handles requests and passes data back to the browser. In fig. 1 The browser does the more difficult work of presenting the text, displaying graphics, generating sound or video and running Java applets. In today's world, embedded systems are everywhere, homes, offices, cars, factories, hospitals, plains and consumer electronics. They span all aspects of modern life and examples of their use are numerous. Modern embedded systems are able to connect to the internet and can be remotely maintained and diagnosed [2]. M2M (Machine to Machine) communication is growing with a considerable rate.The possibility to connect two or more embedded systems enables developers to build more powerful distributed systems such as networked embedded systems. Remote maintenance is performed by different communication protocols. The most common communication protocol is HTTP which enables remote system control and monitoring. A web server is a computer program that implements HTTP protocol. It accepts HTTP requests from clients like web browsers and serves HTTP responses which are usually HTML pages with linked objects. There are many web servers available, and a number of them are free, like Apache [3], AOL [4], Roxen5 [5]. There are a multitude of standards that address mid to high data rates for voice, PC LANs, video, etc. However, up till now there hasn't been a wireless network standard that meets the unique needs of sensors and control devices. Sensors and controls don't need high bandwidth but they do need low latency and very low energy consumption for long battery lives and for large device arrays. There are a multitude of proprietary wireless systems manufactured today to solve a multitude of problems that also don't require high data rates but do require low cost and very low current drain. These proprietary systems were designed because there were no standards that met their requirements. These legacy systems are creating significant interoperability problems with each other and with newer technologies. The ZigBee Alliance is not pushing a technology; rather it is providing a standardized base set of solutions for sensor and control systems.

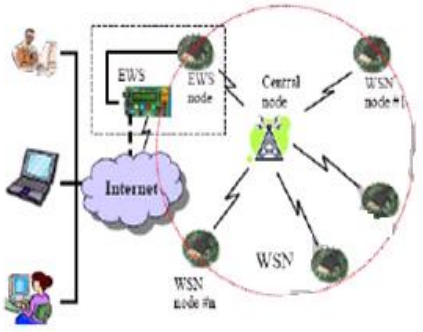

Fig. 1: Internet based enabled network

Internet Information Services [6], Sun Java System web server [7] are some of the most common commercial web servers. Some web servers can run on almost any operating

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 183-188 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

system while others are platform specific. The general purpose web servers are intended to run on powerful server computers, workstations or personal computers and support a number of advanced features. On the other hand, web servers for embedded systems have limited resources and offer only a set of required features. Requirements of an embedded system web server are:

• Small RAM and ROM footprint,
• Low CPU consumption,
• Easy integration with existing application, including static link with the operating system and application,
• serving pages from the RAM if there is no hard drive.

Embedded system web servers are used in different applications. In our case, we developed a generic embedded system web server for remote monitoring in integrated ICT solutions for aging society. In the following, we describe our approach and the main features of the developed system.

II. IMPLEMENTATION

Because we are working on several different wireless sensor platforms we needed the EWS to be platform independent. Therefore, we prefer the server to be a separate device that can be used as an expansion board for the existing sensor nodes. One of the sensor nodes of a currently monitored network is attached (using a standard serial connection) to the web server. In our terminology this node is called a EWS node. The EWS node is running the same software and implementing the same communication protocol as the rest of the sensor network nodes. The only difference is that, in addition to its normal functions, it can also forward messages from the EWS to the central node of the network and vice versa. Various infrastructure scenarios called for the web server capable of working with both wired (Ethernet) and wireless (802.11b) interfaces. Also, we needed flexibility in obtaining the IP address for the EWS. To allow access from off-campus locations we needed the server to be able to use a fixed, pre-assigned IP address that is visible from the other side of the University firewall.

For on-campus wired or wireless applications we needed the web server to be able to obtain an IP address from the DHCP server. Finally, for periodic visits to the WSNs employed in environments without Internet infrastructure, we needed the server to be able to operate in Adhoc mode by establishing point to point connection with a Wi-Fi enabled Personal Digital Assistant (PDA) or a notebook. From the hardware point of view the embedded system web server must be comprised of an embedded processor, embedded memory and required peripherals like Ethernet controller and input/output interface. In order to assure flexibility for different possible applications the embedded system was implemented on a FPGA platform. Soft-core processors like Xilinx Microblaze, Altera Nios as well as any processor written in HDL languages can be implemented in FPGA devices. This avoids the need of a separate processor chip. The soft-core processor configuration can be additionally tailored for specific application. Required peripheral devices are connected via general purpose bus. The embedded system must have sufficient memory to contain an operating system with a server application. We decided to use the open source operating system uClinux.

III. HARDWARE PLATFORM

The designed embedded system is based on Xilinx ML401 development board with Virtex 4 [8] shown in Fig. 2. A Microblaze embedded processor is used [9]. The Microblaze soft processor core is a reduced instruction set computer (RISC) optimized for implementation in Xilinx FPGAs. The core is highly configurable, allowing one to select a specific set of features required by your design. Fixed processor features are:

• Thirty-two 32-bit general purpose registers,
• 32-bit instruction word with three operands and two addressing modes,
• 32-bit address bus,
• Single issue pipeline.

Fig. 2: Soft cosole Eclipse based IDEIncludes a tiny embedded web server with CGI scripting

The processor is adjusted and implemented in the FPGA. The peripherals are connected to the processor via IBM OPB (Open Peripheral Bus) [10]. The Embedded system architecture is depicted in Fig. 2. An external dynamic RAM of the size 64 MB is used as main memory. The internet connection is established by an on board Ethernet controller. Platform flash is used as a nonvolatile memory to program the FPGA and boot the server on power up. For the purpose of outside interface the server uses two digital expansion connectors with 64 digital input/output signals. It also has some human interface devices such as LCD display, buttons, LEDs, VGA output and a sound card.

3.1 Web server configuration

The demo application includes an embedded web server. The IP address used by the web server is fixed, and set set by the constants configIP_ADDR0 to configIP_ADDR3. The definition of these constants can be found near the bottom of the FreeRTOSConfig.h header file, which is itself located in the same directory as the SoftConsole project file. Constants that define the MAC address, and the NET mask, are located in the same header file. The IP addresses used by the web server running on the SmartFusion device, and the web browser used to connect to the web server, must be compatible with each other. To ensure this is the case, make the first three octets of both IP addresses the same. For example, if the web browser computer uses IP address 192.168.0.1, then the SmartFusion device can be given any IP address in the range 192.168.0.2 to 192.168.0.254 - other than any IP addresses that is already present on the same network. Fig. 3 contains MAC address assigned to the Smart Fusion device must be unique on the network to which the device is being connected.

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 183-188 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

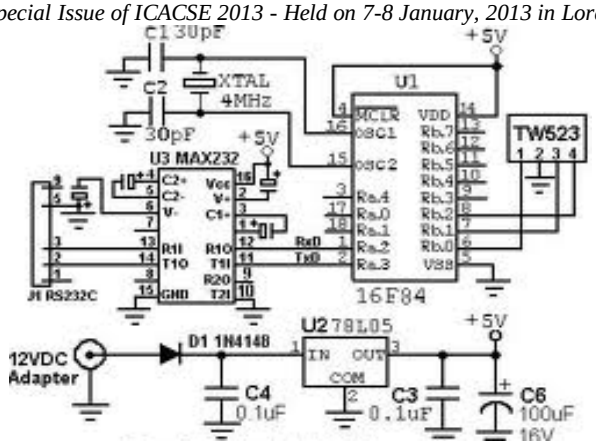

Fig. 3: Hardware connection inside embedded web server development board

IV. SOFTWARE DESIGN

A. Interrupt service routines

Unlike most ports, interrupt service routines that cause a context switch have no special requirements, and can be written as per the compiler documentation. The macro portEND_SWITCHING_ISR() can be used to request a context switch from within an interrupt service routine.

Note that portEND_SWITCHING_ISR() will leave interrupts enabled.

This demo project provides examples of FreeRTOS interrupt service routines - namely GPIO8_IRQHandler() defined in main-full.c and main-blinky.c, and the Ethernet MAC interrupt handler.

Only FreeRTOS API functions that end in "FromISR" can be called from an interrupt service routine - and then only if the priority of the interrupt is less than or equal to that set by the configMAX_SYSCALL_INTERRUPT_PRIORITY configuration constant.

The web server runs on uClinux (microcontroller Linux version) operating system [11]. The uClinux is a port of Linux to systems without a Memory Management Unit (MMU) and therefore designed for dedicated small computer systems like microcontrollers and small microprocessors which are suitable for implementation in FPGA. Today's uClinux releases are based on current versions of linux kernel (2.6, 2.4 as well as 2.2) and include matching tool-chains and collection of different libraries and applications. uClinux kernel has to be adjusted to the target system and compiled. Drivers are selected and some custom made drivers are added as shown in Fig. 4.

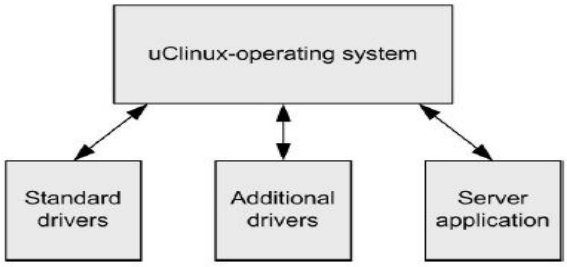

Fig. 4: Software parts

Embedded web server application was programmed in Linux environment. The differences between our embedded web server and web servers like Apache are limited configuration settings and limited number of simultaneous connections. Major general purpose servers immediately spawn a dedicated process for every incoming request of a client. The process then continues the communication with the client. This requires a lot of memory resources during run time and of course a MMU. Many web browsers can access such a web server simultaneously. On the other hand, embedded web server works only with one single process. If two users need to get access to an embedded web server simultaneously, one of them has to wait until the other finishes. This may be justifiable in specific applications of embedded systems (i.e., remote maintenance, remote configuration, etc.) where not many simultaneous requests are expected.

V. WEB SERVER PROGRAM

This simple web server provides access to an embedded system via web browser [12], [13]. The server delivers the desired HTML pages and pictures over the internet or a local intranet network to the web browser. The web content is built by individual files. The base is built by static files with HTML pages. Within such HTML files there are references to other embedded files. These files are typically pictures in GIF or JPEG format and are also transmitted to the web clients. The communication protocol is HTTP described in RFC 1945 [14] and RFC 2616 [15]. A web server in a simplest form can be viewed as a special kind of a file server. Its operation can be summarized as follows:

- A web browser requests a file from a web server by issuing the HTTP GET request.
- The web server receives the HTTP GET request and accesses its file system for the requested file.
- The web server generates the HTTP response which is comprised of an http header and the requested file and transmits it to the browser.
- If the requested file is not accessible a special error response is generated and transmitted to the browser. The process of client-server file transfer is illustrated in Fig. 5.

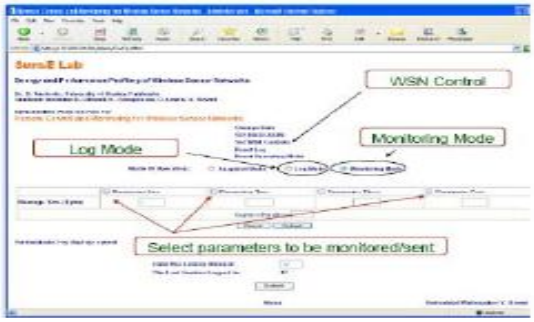

Figure 5: Client-server file transfer

Apart from transferring HTML and other files, this server application can communicate with the input/output devices like sensors and actuators, LCD display, buttons, LEDs, VGA display of the embedded system:

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 183-188 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

• The web browser request comprises of a command and parameters.

• The server recognizes the command and writes on the requested input/output device or reads from it and transmits the data in a form of HTML page to the web browser.

The process of client-server communication and server access to the file system and input/output devices is illustrated in Fig. 6.

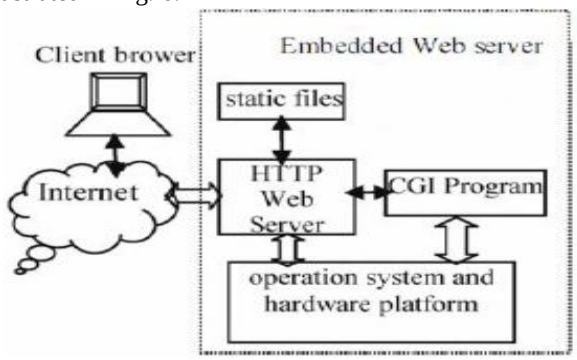

Fig. 6: Embedded web Client-server communication

VI. IMPLEMENTATION ISSUES

Web Server Operating Modes: The web server can operate in one of the following three modes:

a. **Snapshot mode,**
b. **Monitoring mode,**
c. **Log mode**.

Snapshot mode: In the Snapshot mode the server passively listens to the requests for network parameters (data) from the clients. When the request is received, the server initiates communication with the EWS node. The embedded web server node forwards the request to the central node. The central node retrieves the most recently collected network data and responds to the EWS node which in turn forwards it to the server. The server processes the received data and presents [12]-[13] the graphical representation of the parameter value to the client. Snapshot mode is used to display the current network data in near-real-time.

Monitoring mode: In the Monitoring mode the server continuously monitors the network and actively collects the data from the network, as configured. When the server is operating in monitoring mode, the result shows a continuous live parameter values using a graphical representation of the data. The time between the requests depends on the frame period of the WSN. In the Log mode [20] the server collects the request subset of network data for a given period of time. The data is stored in the EWS' serial flash. The data can be remotely read from the flash and decoded and displayed using an application developed in MATLAB®. Depending on availability, the flash contents can be read either through the web browser or directly from the serial port of the web server. The RCM3750 has an Atmel Flash AT45DB081B, a serial interface 1 MB flash memory designed specifically for code and data storage applications.

Log mode: In log mode, the server collects data from the network and stores it on the serial flash .The web server was implemented on Virtex 4 FPGA board. A Xilinx Embedded Development Kit (EDK) was used to select and construct the required hardware. The Microblaze processor was adjusted to minimize FPGA resources and peripherals were connected via OPB bus. The hardware drivers of peripherals were selected from Xilinx IP cores. The uClinux libraries were added to the EDK to include the uClinux operating system in the software platform settings. Then the hardware was compiled and downloaded into the FPGA device. Occupied space and FPGA resources are shown in Tab.1.

Table 1: Implementation results

	Used capacity	Whole capacity	Percent of capacity
Slices	7,112	10,752	66 %
Lookup tables	10,014	21,504	46 %
Flip Flops	6,382	21,504	29 %
I/O signals	176	448	39 %
BRAM	54	72	75 %
DRAM	7MB	64 MB	11%

The software development of the embedded system is divided into two parts:

• Configuration and compilation of uClinux operating system.

• Programming of server application.

As regards operation system, the latest distribution of uClinux was downloaded. The integrated web server will serve the web pages described below, to [14] a standard web browser.

To connect to the target: Open a web browser on the connected computer. As shown in Fig. 7, type "HTTP://" followed by the target IP address into the browsers address bar.

Fig. 7: Entering the IP Address Into the web browser.

The uClinux kernel was adjusted to the given embedded system. The appropriate drivers were selected and custom drivers for LCD display and general purpose input/output devices were developed. The server program was first programmed and debugged on a personal computer and then [15]-[16]-[17] ported to uClinux. The uClinux operating system and developed web server were packed to a software image. This image is downloaded to an embedded system memory during the boot-up process. The image was built on a personal computer in a Linux environment with a Micro blaze tool chain. To initialize the server at power-up a platform flash was used. On the first location of the platform flash, a hardware configuration was stored along with a small boot loader program. On the second location of the flash, uClinux operating system with the web server were stored. The boot loader program is used to transfer the software from

second part of the flash into DRAMs and to boot the uClinux operating system.

VII. FUNCTIONALITY

The LEDs marked D1, D2 and D3 are under the control of the standard demo 'flash' tasks. Each will toggle at a fixed frequency, fig. 8 with LED D1 using the fastest frequency, and LED D3 using the slowest frequency. The LEDs marked D4 and D5 can be turned on an off interactively using a web browser. The facility to do this is provided on the "IO" page that is served by the embedded web server. The "IO" page also displays the raw reading [18]-[19]-[20] of the analogue input port connected to the potentiometer marked 50K_POT (mounted next to the OLED on the development board). The LED marked D6 is under the control of queue receive task. It will toggle each time the queue receive task receives a value, which will be every 200ms. The LED marked D7 will turn on when button SW1 is pressed (assuming it was not already on). Five seconds after SW1 was last pressed, the LED software timer will turn LED D7 off again. The LED marked D8 is under the control of the 'check' software timer. D8 will toggle every 3 seconds if no errors have been reported, or every 500ms if a standard demo task has ever reported an error.

The developed embedded system has the following hardware specifications:

- Xilinx Virtex 4 XC4VLX25 FPGA.
- Xilinx Microblaze soft core embedded processor.
- 100 MHz computer clock.
- 16 kB data cache and 16 kB instruction cache.
- 64kB block RAM inside FPGA.
- 64MB DDR SDRAM as main memory.
- 10/100/1000 tri-speed Ethernet PHY transceiver
- Xilinx XCF32P Platform Flash configuration storage device.
- Input/output devices.

The web server program has the following features:

- HTTP file transfer,
- Static HTML page hosting,
- Up to 100 simultaneous connections,
- custom input/output communication support. Various input/output devices are connected to the embedded system, such as:
- A/D converter,
- Temperature and pressure sensors,
- LCD display,
- AC97 audio sound card,
- Push buttons and position switches,
- LEDs.

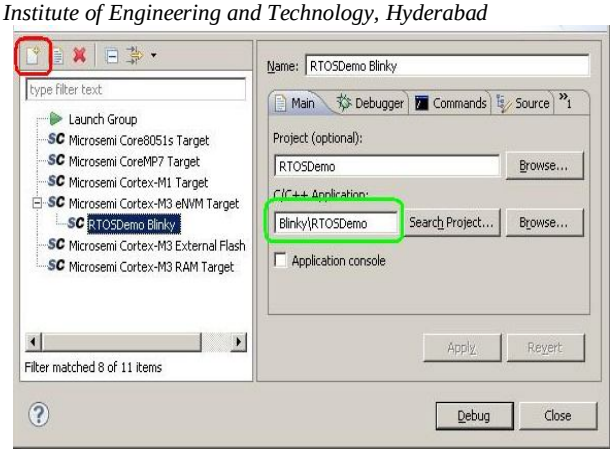

Fig. 8: Creating a debug configuration in embedded server

VIII. PRACTICAL EXPERIENCE

For initial system test in practice, we developed a simple embedded application. A web page was made and selected input/output devices were remotely accessed. The application includes:

- Parallel input/output testing using push buttons and LEDs,
- LCD display to show messages from web,
- A/D converter to interface the analog sensors,
- Temperature sensor to measure room temperature,
- Pressure sensor to measure air pressure,
- AC97 audio sound card to play and record sound.

IX. CONCLUSION

With an HTTP over IP based application. The application allows a user to access WSN data directly from a Web browser. The paper described the main building blocks of the gateway connecting the Web client with the WSN. The gateway is still in prototype phase and it requires the development of proxy and observation functionalities. The database performance needs to be tested for scalability purpose. The application will be deployed and tested in a greenhouse monitoring testbed. We developed an embedded system web server that provides a simple network interface for custom FPGA embedded designs. In the development of the embedded web server we dealt with the definition of user requirement specification and selection of software environment dilemma, selection of supporting hardware platform and availability of associated cores, and provision of development tools, and programming of target application. Limited space does not allow us to describe individual issues in details. Nevertheless, the described solution may be helpful to potential designers facing similar problems in practice. In addition, given references may broaden the perspective and suggest possible alternatives. The system is based on the Microblaze soft-core processor and the application runs on the uClinux operating system.

REFERENCES

[1] A. Sanz, J. I. Garcia-Nicolas, and P. Estopinan, ―A complete node for power line communications in a single chip,‖ in Proc. 2005 Int. Symp. Power Line Communications and its Applications, Apr. 6–8, 2005, pp. 285–289.

[2] K. B. Lee and R. D. Schneeman, ―Distributed measurement and control based on the IEEE 1451 smart transducer interface standards,‖ IEEE Trans. Instrum. Meas., vol. 49, pp. 621–627, Jun. 2000.

[3] S. Mylvaganam, H.Waerstad, and L. Cortvriendt, ―From sensor to web using PLC with embedded web server for remote monitoring of processes,‖ in Proc. 2003 IEEE Sensors, Oct. 22–24, 2003, vol. 2, pp. 966–969.
[4] Tammy Noergaard, Embedded Systems Architecture, Elsevier inc., 2005.
[5] James Kurien,Xsenofon Koutsoukos, Feng Zhao, Distributed Diagnosis of Networked, Embedded Systems, proceedings of the 13th International Workshop on Principles of Diagnosis, 2002.
[6] http://www.apache.org/ (access date 10.1.2008).
[7] http://www.aolserver.com/ (access date 10.1.2008).
[8] http://www.roxen.com/products/cms/webserver/ (access date 10.1.2008).
[9] http://www.microsoft.com/windowsserver2003/ iis/default.mspx/ (access date 10.1.2008).
[10] http://www.sun.com/software/products/web_srvr/ home_web_srvr.xml/ (access date 10.1.2008).
[11] Xilinx, ML401/ML402/ML403 Evaluation Platform User Guide, version 2.5, 2006.
[12] Xilinx, MicroBlaze Processor Reference Guide UG081, version 8.0, 2007.
[13] IBM, On-Chip Peripheral Bus, Architecture Specifications, version 2.1, 2001.
[14] Xilinx, Getting Started with uClinux on MicroBlaze Processor, version 1.2, 2007.
[15] Xilinx, Embedded System Example, XAPP433, version 2.2, 2006.
[16] W. Richard Stevens, UNIX Network Programming, Prentice Hall, London, 1990.
[17] Berners-Lee, T., Fielding, R. and H. Frystyk, Hypertext Transfer Protocol-HTTP/1.0, RFC 1945, 1996.
[18] Berners-Lee, T., Fielding, R. and H. Frystyk, Hypertext Transfer Protocol-HTTP/1.1, RFC 2616, 1999.
[19] D. Stipanicev and J. Marasovic, ―Networked embedded greenhouse monitoring and control, in Proc. 2003 IEEE Conf. Control Applications, Jun. 23–25, 2003, vol. 2, pp. 1350–1355.
[20] M. Bertocco and M. Parvis, ―Platform independent architecture for distributed measurement systems, in Proc. 17th IEEE Instrumentation Measurement Technology Conf., Baltimore, MD, May 1–4, 2000, vol. 2, pp. 648–651.

AUTHORS' PROFILE

1]. Prof. P. Rama Bayapa Reddy completed BE in computers from Marathwada University, Aurangabad and M.Tech from JNTU, Anantapur. Now presently pursuing Ph.D from JNTUA, Anantapur. My interested Research area is internet controlled embedded systems. I have published twelve papers on Embedded systems in International Journals. I conducted two national level seminars. I also conducted two workshops on Real Time Embedded Systems. Also I attended workshops conducted by JNTUK. I have total of 15 years of teaching experience. Right now I am working as Professor in Computer Science and Engineering in Dhruva institute of Engineering and Technology, Hyderabad. putluru.ramreddy@gmail.com

2]. Dr.K.Soundararajan received the B.E Degree in Electronics and communications from S.V.U, Tirupati and M.Tech Degree in Instrumentation and control Engg. from J.N.T.U, Kakinada. He received Ph.D from University of Roorkee, Roorkee. He is having 32 years of teaching experience. He got the Best teacher award for the year 2006 from Andhra Pradesh Government and from Lead India2020 in 2005, President of India Award in Bharat Scouts & Guides in 1968, Best Paper Award in 1990–91 and 2000 from Institution of Engineers (India) for best technical paper published in Journal. He is an Expert Committee member, for different Central Government organizations, affiliation committee member for several colleges and different academic boards. He published 21International Journals/Conferences and 27 national journals/conferences and he participated in 12 National seminars. He guided 9 Ph.Ds s and 10 research scholars are working to obtain their Ph.D. He worked as principal of JNT university College of Engineering Anantapur, A.P and Rector of JNTU Anantapur, Anantapur. Presently, he is working as Professor, Department of ECE, JNTUACE, of Jawaharlal Nehru Technological University College of Engineering, Anantapur, A.P, India.

soundararajan_jntucea@yahoo.com

3]. Dr M.H.M Krishna Prasad, has completed B.Tech, M.Tech (CSE), JNTU, Hyderabad Ph.D., in Computer Science & Engineering from JNTU Hyderabad with specialization as Data Mining and successfully completed a two year MIUR fellowship at University of Udine, Italy. He has published no of papers in International and National Journals. Under his guidance multiple research scholars are doing research. Presently he is working as Associate Professor of CSE & Head, Dept. of Information Technology University College of Engineering, JNTUK-Vizianagaram Campus, Vizianagaram, A.P, India**.** krishnaprasad.mhm@gmail.com.