International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online)) Volume 3, Issue 2, August 2014)

Server for Remote Generation of Wake On LAN Packets in the LAN

Gencho Stoitsov[1], Zhelyan Guglev[2]

[1]*Plovdiv University "Paisii Hilendarski", 236 Bulgaria Blvd., 4003 Plovdiv, Bulgaria*
[2]*Plovdiv University "Paisii Hilendarski", 236 Bulgaria Blvd., 4003 Plovdiv, Bulgaria*

***Abstract*—The article offers a solution for using the technology Wake On Local Area Network (WOL), based on a hardware device with multiples low cost and power consumption compared to a standard computer system.**
This implementation was based to the simplicity of the WOL protocol and low requirements for computing power.

***Keywords*—EEPROM memory, ENC28J60, Local Area Network, MAC, Microprocessor, MPFS2, OSI model, PIC18F4620, Regular expression, Wake on LAN, WOL packets.**

I. INTRODUCTION

The technology Wake On Local Area Network or WOL (lit. translation Wake On LAN) is a communication standard that allows a remote device to be turned on or brought out of standby mode (stand by) by sending specially formatted packets on Ethernet (IEEE 802.3) network. The basis of WOL is the idea that after shutting down or putting the computer system in a standby mode, its network card continues to monitor the communication traffic for WOL packets addressed to it. If valid ones are detected, the computer turns on. The packets of the WOL type use the channel (the second) layer of the OSI model, and in some implementations they may also cover the network (the third) and the transport (the fourth) layer.

Although Wake On LAN technology has been designed to work at a local network, there is a possibility by conducting a number of additional settings, WOL packets sent from external networks including the Internet to reach its recipient. This variation of Wake On LAN technology is known as Wake On LAN over Internet. Most often, the settings that are made concern the routers and firewalls of a specific network. If the router supports a direct Wake On LAN over Internet setup - it is activated, and if this is not available - it is necessary to set up port forwarding. Network firewalls should also be set so as to allow the WOL packets [1][4][6][8].

One of the biggest problems for Wake On LAN technology is that there is a high probability a WOL packet sent from a source outside the local network where the recipient is, may not be able to reach its goal [1][2][3][4][6].

Routers that do not support direct Wake On LAN over Internet setup, even with a properly configured port forwarding, it is possible due to internal technical specifications not to be able to forward the WOL packets coming from external networks.

If the track, where there is network communication, has placed firewalls a common situation is by default that they filter all WOL traffic, and the end users do not always have authority to change settings or replace firewalls & routers.

II. ONE SOLUTION OF THE PROBLEM

The proposal that we do is linked to the realization of a hardware device and its software provisioning, allowing the access to the server application for generating WOL packets in LAN.

Since the WOL protocol is relatively simple and does not require much computing power, it can be integrated into a number of alternative devices that cost much less and consume less electricity compared to a typical computer system. The following configuration is an example of such a platform:

- Microprocessor: Microchip PIC18F4620:
  - Architecture: HARVARD, RISC
  - Maximum speed: 10 MIPS @ 40 MHz, target speed 8 MIPS @ 32 MHz
  - Program memory: 64 KB
  - Data memory:
    - SRAM: 3986 bytes
    - EEPROM: 1024 bytes
- Ethernet Interface: Microchip ENC28J60:
  - Communication Interfaces:
    - Ethernet 10Base-T
    - Serial Peripheral Interface (SPI)
  - MAC (OSI 2)
  - PHY (OSI 1)
  - RAM buffer: 8 KB

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online)) Volume 3, Issue 2, August 2014)

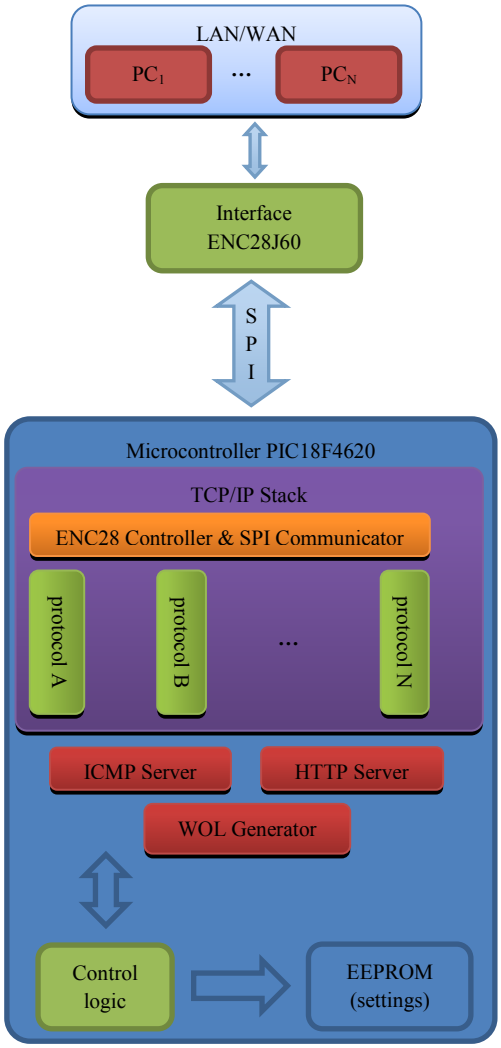

Figure 1. Project architecture

The end result is a combination of several types of applications with the following functionality:

- Web-based user interface that allows remote access via HTTP protocol which supports access control through authorization with a username and a password;
- Manual and automatic mode of generating WOL packets on the local network;
- User list of devices and target MAC addresses;
- Manual or automatic configuration of an individual network parameters plus support of DHCP protocol;
- Displaying statistical and diagnostic information - indicators of the server status, ICMP server module;

III. User Interface (Front-End)

Built on one of the most popular web technologies - HTML5, CSS3 and JavaScript, the user interface provides the link between the management and the user, but at the same time it serves as a preliminary validator of the data that will be transmitted to the control part.

- **Inspection and control of the input data**

Implementation of front-end validation is done by java script. A useful side effect of it is reducing unnecessary traffic to the server as potentially unfulfillable requests are "removed" beforehand. One of the main tools for performing validations are the regular expressions - special templates giving specific criteria with which bits of data can be analysed, substituted or "selected":

TABLE I
List of Used Regular Expressions

Expression	Purpose
/suffix$/	To check for a string ending with a suffix, where the suffix is a dynamic parameter.
/([0-9a-f]{2})[\-:]([0-9a-f]{2})[\-:]([0-9a-f]{2})[\-:]([0-9a-f]{2})[\-:]([0-9a-f]{2})[\-:]([0-9a-f]{2})/i	To recognize and select each of the six hexadecimal numbers of a MAC address.
/:/g	To find (and possibly substitute) all occurrences of the symbol ":".
/([0-9]{1,3})[\.]([0-9]{1,3})[\.]([0-9]{1,3})[\.]([0-9]{1,3})/	To recognize and select each of the four numbers of the IPv4 address. A further validation of the range of input numbers is necessary (using js).
/[^\w\-\.]/g	To finding (and possibly substitute) parts of the string in which there are symbols other than the permitted ones.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online)) Volume 3, Issue 2, August 2014)

- **Compression of the user interface**

The limited amount of memory available to the platform requires the use of different compression techniques on the files building the user interface that are applied once before the process of compilation of the final software product. Besides conventional techniques such as GZIP compression without loss of data and specialized in HTML, CSS, JS compression with loss of data, a technique is used for the fragmentation of common parts of HTML resources and separating them as internal files that remain invisible to the user. In requiring a HTML resource, the modules for content management dynamically "stitch" the fragments associated with the resource providing the user a complete result without generating unnecessary network traffic. The effect of the fragmentation technique is a compression that gains about 29% and the combination of all kinds of compression techniques gain- 68%.

IV. Management (back-end)

Control logic of the project is written in C language and uses extended from ANSI standard offered by the compiler MPLAB C18 [9], which is specially optimized for the platform on which the code is run. Control logic manages the hardware resources, manages the network interface, the data resources, supports and interacts with a number of communication protocols, controls the access and the dynamic content management.

A major role in achieving the objectives listed above performs the Microchip TCP/IP Stack, which can be defined as a basis and a framework, a set of libraries and modules around which the back-end logic is built. The stack provides the basis for the interaction between the physical communication environment and the application that works with it.

Modules are included that provide support for some of the most commonly used application layers such as HTTP, SMTP, SNMP, Telnet, TFTP, Serial-To-Ethernet, etc. as well as fast and lightweight implementations of TCP and UDP transport layers plus modules that support protocols such as IP, ICMP, DHCP, ARP and DNS.

The modules in the stack are built on the principle of cooperative multitasking - an approach where applications responsible for different types of functionalities are divided into modules that work "simultaneously" [8][5]. To be able to run "simultaneous" work with several modules, they must be made and invoked in a way that provides a "multitasking" performance - each module has its own mechanism by which to determine whether the current state of the system requires it to use its resources in order to perform the processing for which it is intended, or make the resources available for use to other modules.

Each module contains a code divided into blocks as the performance of the block is an atomic operation. Once the block is filled, the current state of the module is stored, and the processor resources are available to other modules.

V. Structures for processing and storage of data

- **Positioning in the memory**

The specific features of HARVARD architecture and compiler C18 permit management of data location in the memory in a manner similar to x86 and x86_64 architectures, but adding advanced control techniques.

Two types of memory models - close (near) and distant (far) and two types of memory - program (rom) and data memory (ram). Qualifiers used by default, and implicitly by the compiler are far and ram. The effects from the different combinations [16]:

TABLE II
Qualifiers for the Memory Type of the Compiler MPLAB C18

	rom	**ram**
far	Anywhere in the program memory.	Anywhere in the data memory.
near	In the program memory with address space no larger than 64 KB.	In the data memory - the bank with a direct access.

- **Storage and reading settings from EEPROM memory**

The electrically erasable program memory is a medium for a long-term data storage. In a three-tier Web architecture, the EEPROM memory together with the code that manages the storage and retrieval of data from it, form the data layer.

Different data categories of different types are stored in the EEPROM memory - Boolean, integer of between 1 and 4 bytes strings. To ensure correct recording and reading - address definitions (offsets) are used, which have the corresponding variable or with which its location can be calculated, knowing its size.

The project allows keeping a stored list of device names and MAC addresses, to which the user can manually send a WOL packet or set the server to automatically send such packets every 5 minutes.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online)) Volume 3, Issue 2, August 2014)

The list is kept in the EEPROM memory in a fixed tabular form, in which the size of the stored item for one device is 22 bytes - 16 for the device name and 6 for a MAC address, and the maximum number of entries is 42 - about 90% of the EEPROM memory (the remaining 10% are for storing control variables and network settings).Information of the current number of stored items is kept separately, outside the table to improve the performance. Using a table storage technique allows quickly and easily to determine the location in the table where to add a new entry and eliminates the problems of fragmentation that would occur in recording data of varying size in a random location. "Deleting" a recording is done as on the first byte, which stores the name of the device is assigned a zero terminator - i.e. it acts as an indicator noting that the relevant "row" of the table is free. This method simultaneously enhances the system's performance and reduces wearing out the EEPROM memory.

- **Storage of front-end components in the program memory**

Storing files that build user interface is done by the specially designated file system - Microchip File System 2, equal in functionality to other file systems, but providing increased efficiency at the cost of a read-only mode. Since the available internal program memory of the micro-controller is used, MPFS2 is compiled with the code of the entire application, becoming part of it and reducing the time to retrieve a requested web resource.

VI. Dynamic Content Management

In the HTTP server module of the project is an integrated logic that allows associating the content of a given HTML resource with compiled C functions. The result of their implementation dynamically replaces specialized syntactic structures that have been previously declared in the HTML resource. The association between a resource, a syntactic structure and a C function happens in the process of building MPFS2. An example of such association being used, is between the file "index.html", association "uptime" and its implementation in the file "CustomHTTPApp.c":

```
<!-- ... -->
<!-- File index.html -->
<!-- ... -->
<tr>
  <td class="contentTableRowCaption">Up Time in minutes:</td>
  <td class="contentTableRowValue">~uptime~</td>
<!-- ... -->
```

```
/**
 * ...
 * File CustomHTTPApp.c
 * ...
 */
void HTTPPrint_uptime(void)
{
  BYTE seconds[11];
  ultoa(WOLOI_UpTimeSeconds / 120, seconds);
  TCPPutString(sktHTTP, seconds);
}
/* ... */
```

VII. Some Peculiarities When Using Network Protocols

The protocol for a Dynamic Host Configuration provides methods for automated configuration of network parameters such as IP address, network mask, gateway and DNS servers [8][12].

Its use by default by the Wake On LAN over Internet server is activated as the user has the option to disable it and manually set network parameters.

If the use of the DHCP server is enabled, but for some reason it cannot be accessed, the Wake On LAN over Internet server uses factory set network parameters that the user can change:

- IP address: 192.168.1.254;
- Mask: 255.255.255.0;
- Gateway: 192.168.1.1;
- (Network class: C, 24 bits for the identifier of a network and 8 bits for identifying the host).

Hypertext Transfer Protocol and protection

A protocol running on the application layer of the OSI model. In most cases, accessing HTTP services is carried out at a transport layer using TCP protocol, although access is possible via UDP [8][13].

The module, implementing HTTP in the project, supports the latest updated version of the protocol that is HTTP/1.1.

The web access to Wake on LAN over Internet server is protected by default with a username ("admin") and a password ("default"). The protection itself is based on a built-in implementation in the HTTP module, which is standard for the protocol and works as follows:

- When a new client tries to access a resource, the server returns an error code "401 Unauthorized" in addition it requires a password for the relevant class of resources, which in this case is only one ("WWW-Authenticate: Basic realm =" Protected "", "401 Unauthorized: Password required").

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online)) Volume 3, Issue 2, August 2014)

- A standard window appears in the browser of the client, prompting to enter the required data. After entering the data, they are encoded to base64 string, which is sent to the server along with the requested resource ("Authorization: Basic YWRtaW46ZGVmYXVsdA ==").
- The server decodes the received data and decides whether to return the resource ("200 OK") or a message that authorization is required ("401 Unauthorized", as in step I).

Wake On LAN

There are two options for generating WOL packets - generating directly after the Ethernet portion of the corresponding frame or by using a transport protocol. The current implementation of the project uses the second option relying on the UDP protocol. Parameters are set as follows:

- Recipient: 255.255.255.255;
- Local port: random;
- Remote port: 9 (standard adopted for WOL over UDP).

The content of the WOL packet can be separated into three parts - a synchronization flow, a MAC address and a heterogeneous type of post-data.

A synchronization stream is defined as 6 followers bytes with a value FFh, the synchronization stream facilitates the process of scanning incoming data packets for the network adapter [1][4].

The defined hardware address of the network adapter (A MAC address) consists of 6 eight bit numbers with possible values in the range [0h; FFh] works on the channel (second) layer of the OSI model [17]. Unique to each network adapter and used in the process of commutation of network packets. If no power or after rebooting, it does not change. When building a WOL packet, 16 copies of it (for a total of 96 bytes) have to be available within the package.

Heterogeneous type of post-data (optional) - it is possible that at the end of the WOL packet, additional information to be attached. In rare cases, when the network adapter is set to receive WOL packets, authorized with a password, the password is placed at the end of the magic packet. Its size is mostly between 4 and 6 bytes.

The password is sent in an unencrypted form. In combination with the fact that in most cases WOL packets are sent to a multicast or to a broadcast address, this means that any person having access to the relevant network can read the password. Therefore, it is not a common practice WOL packets to be sent in combination with a password [1].

The current version of the software does not add post-data to the WOL packets.

VIII. Conclusion

A fully functional product is created, based on a non-standard platform. It successfully integrates a number of both old and established as well as latest up-to-date standards and technologies affecting the areas of network programming, web programming and marking, multitasking, compression, data fragmentation.

After several tests it was found that the product has stable performance and the speed with which it performs the assigned tasks is more than satisfactory. In spite of the modest hardware parameters of the platform, the time for which the system is ready for use after switching or rebooting is less than 1 second. Data transfer from the server to the client and vice versa is almost instantaneous in the local network, which fully justifies the choice of a target speed limit to 10 Mbit/s.

- **A brief analysis of the alternatives of economic importance**

The main parameters of alternative platforms are compared, on which a similar project could be built, as the cells with the most optimal parameters are marked in green:

TABLE III
Comparison of Important Project Parameters, Characteristic of the Different Platforms

Name	Cost BGN (Bulg. lev)	Built-in 802.3 interface	Average productivity MIPS	Electric energy consumption W	Prototype
Current platform	25	no	8	1,89	yes
Arduino Uno	40	no	16	2,16	no
Raspberry Pi	62	yes	640	2,5 – 3	no
PC - Laptop	800	yes	1178	65	no
PC - Desktop	400	yes	1200	100	no
Server	2500	yes	2412	160	no

The currently selected platform has two of the four optimal indicators that characterize it as the most economical one. Considering that it is a prototype, it means that its parameters are still far from being the most optimal.

References

[1] AMD, "Magic Packet Technology," November 1995. [Online]. Available: http://support.amd.com/TechDocs/20213.pdf. [Accessed 04 May 2014]

[2] Cisco, "Internetworking Basics," 1998. [Online]. Available: http://www.cisco.com/cpress/cc/td/cpress/fund/ith/ith01gb.htm#xtocid166844. [Accessed 04 May 2014].

[3] IEEE, "History of Ethernet," 22 May 2013. [Online]. Available: http://standards.ieee.org/events/ethernet/history.html. [Accessed 11 May 2014].

[4] Microchip, "AN1120 - Ethernet Theory of Operation," 02 January 2008. [Online]. Available: http://ww1.microchip.com/downloads/en/AppNotes/01120a.pdf. [Accessed 11 May 2014].

[5] Microchip, "AN1264 - Integrating Microchip Libraries with a Real-Time Operating System," 03 March 2009. [Online]. Available: http://ww1.microchip.com/downloads/en/AppNotes/01264a.pdf. [Accessed 29 May 2014].

[6] Microchip, "Microchip ENC28J60 Stand-Alone Ehternet Controller with SPI Interface," 26 October 2012. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/39662e.pdf. [Accessed 05 May 2014].

[7] Microchip, "Microchip PIC18F2525/2620/4525/4620 Data Sheet," 02 January 2008. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/39626e.pdf. [Accessed 05 May 2014].

[8] Microchip, "Microchip v2013-06-15 TCPIP Stack Help," 2013.

[9] Microchip, "MPLAB C18 Getting Started," Microchip, 2005.

[10] Microchip, "SPI Overview and Use of the PICmicro Serial Peripheral Interface," [Online]. Available: http://ww1.microchip.com/downloads/en/devicedoc/spi.pdf. [Accessed 17 May 2014].

[11] Mozilla, "JavaScript," 02 June 2014. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript. [Accessed 19 May 2014].

[12] Network Sorcery, Inc, "DHCP, Dynamic Host Configuration Protocol," [Online]. Available: http://www.networksorcery.com/enp/protocol/dhcp.htm. [Accessed 01 June 2014].

[13] Network Sorcery, Inc, "HTTP, HyperText Transfer Protocol," [Online]. Available: http://www.networksorcery.com/enp/protocol/http.htm. [Accessed 01 June 2014].

[14] Network Sorcery, Inc, "TCP, Transmission Control Protocol," [Online]. Available: http://www.networksorcery.com/enp/protocol/tcp.htm. [Accessed 01 June 2014].

[15] Network Sorcery, Inc, "UDP, User Datagram Protocol," [Online]. Available: http://www.networksorcery.com/enp/protocol/udp.htm. [Accessed 01 June 2014].

[16] Oracle, "Application Architecture," Oracle, 2002. [Online]. Available: http://docs.oracle.com/cd/B10501_01/server.920/a96524/c07dstpr.htm. [Accessed 12 June 2014].

[17] Stoitsov, G. (2010). Types of addresses and levels of use in the TCP/IP protocol stack, REMIA 2010, Plovdiv

[18] W3C, "Cascading Style Sheets (CSS) Snapshot 2010," 12 May 2011. [Online]. Available: http://www.w3.org/TR/CSS/. [Accessed 17 May 2014].

[19] W3C, "HTML5 differences from HTML4," 28 May 2013. [Online]. Available: http://www.w3.org/TR/html5-diff/. [Accessed 17 May 2014].

[20] W3C, „Scalable Vector Graphics (SVG),“ May 2014. [Online]. Available: http://www.w3.org/Graphics/SVG/. [Accessed 26 May 2014].

[21] WHATWG, „HTML Living Standard,“ 11 June 2014. [Online]. Available: http://www.whatwg.org/specs/web-apps/current-work/multipage/introduction.html#introduction. [Accessed 12 June 2014].