Application Note

eZ80® Remote Access

AN013401-0303

ZiLOG Worldwide Headquarters • 532 Race Street • San Jose, CA 95126
Telephone: 408.558.8500 • Fax: 408.558.8300 • www.ZiLOG.com

ZiLOG Worldwide Headquarters
532 Race Street
San Jose, CA 95126
Telephone: 408.558.8500
Fax: 408.558.8300
www.zilog.com

Information Integrity

Document Disclaimer

Table of Contents

List of Figures

List of Tables

eZ80® Remote Access

Introduction

ZiLOG offers an eZ80® Remote Access software package that provides online configuration and firmware updates for servers based on the eZ80® product line, which features the eZ80190 and eZ80L92 microprocessors. This eZ80® Remote Access software package includes the following:

- eZ80® Remote Access Server software in the form of libraries with defined APIs to allow integration with user software
- A remote access Java client PC application, offering client interface features and a TCP/IP connectivity paradigm
- Examples of packet protocol exchange between a client and a server, such as
  - UDP control protocol
  - TCP data transfer channel establishment and usage
- Flash programming capabilities
- This Application Note document

These updates are available on ZiLOG's Software Tools/Downloads page.

The eZ80® Remote Access software package consists of a server application and a client application. eZ80® Remote Access Server software provides the following functionality:

- Online configuration capability
- Online remote access server firmware updates
- Web page updates and downloads over an Ethernet or PPP network

eZ80® Remote Access Client software provides platform-independent user access to the server in two versions:

- A Java VM version from Sun Microsystems (JClient)
- A Windows® JVM MFC classes version (WClient)

eZ80® Remote Access Server functionality is provided via the following two eZ80® applications:

- A NetBooter application that updates server firmware online

- An eZConfig application that performs online configuration for concurrent evaluation of the entire library of IPWorks™ protocols

General Overview

This software has a flexible structure of a set of libraries available to the user.

The current implementation offers the following benefits to the user:

- A convenient method for configuring an eZ80® Remote Access server, allowing every parameter of the server to be accessed and controlled
- Fast, easy firmware updates over Ethernet without requiring additional hardware or connections
- On-the-fly web page and software script updates, which reduce application development time and offer debugging in a real-time network environment

eZ80® Remote Access software is extensible on both server and client sides. New features required by the user application can be added by means of some additional reconfiguration and user callback implementations.

A functional diagram of the interaction between client and server is illustrated in Figure 1.

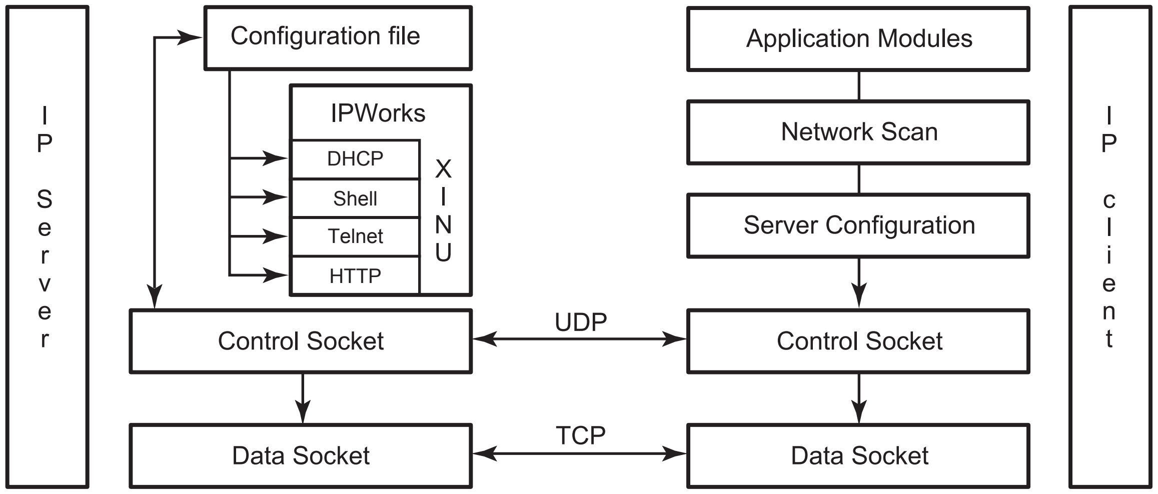


Figure 1. Client-Server Interaction

The client modules, such as Network Scan or Server Configuration, use control and data sockets to transfer commands and data to and from the server. The server uses those sockets, receives command requests, and performs accordingly. The XINU multiprocess environment makes it possible to run multiple network services at the same time. Therefore the client allows to enable several network services in parallel. The services being started up by the Server depending on the current configuration.

The structure, called a configuration file, is stored within Flash memory on the eZ80® Remote Access Server. It controls which modules are loaded upon starting the Server. Figure 1 displays examples of modules available in the IPWorks™ stack at startup. IPWorks™ runs on top of the XINU system. A control UDP socket is opened to allow clients to connect to the server and to control the configuration.

Theory of Operation

The eZ80® Remote Access Server is a powerful connectivity engine and a complex system. This section describes how eZ80® Remote Access software can simplify the user's interactions with the eZ80® Remote Access Server to achieve the advantages featured in the preceding section.

To enable flexible server configuration, configuration data is organized as a special file. This file is stored in Flash memory on the eZ80® Development Platform. After a reset during the boot phase, server initialization occurs according to the date stored in Flash memory. The server starts the most recent configuration.

The Metro IPWorks™ TCP/IP stack runs on the XINU operating system, a real-time multitasking OS. The eZ80® Remote Access Server software is implemented as a process of XINU. To initiate this process, the firmware calls the **ra_init()** function to hide most of the stack-level APIs from the user and to enable connectivity, as described below.

- A datagram command server is started. This server accepts UDP connections on a Port 3000 control port. It is used by client for passing commands in ASCII format to the server and receiving literal responses consisting of a code number and a verbal response from the server. The server parses these commands and performs accordingly. For example, when the Server receives a **Scan** command, it reports its IP and EMAC addresses to the client.
- The server supports configuration file management. A client can command the server to read and store a configuration to Flash memory. Executing this command requires a binary data transfer to be performed. For this purpose, a TCP connection on Port 3001 is used.
- For different application purposes, such as a web file transfer or firmware download, other commands are used, and the callback functions for those commands are implemented on the server side to make the commands work.

Consequently, the server application is flexible and the user can add custom commands to manage the eZ80® Remote Access Server.

The client part of the Server Update software implements the user interface and supports the interface protocol to the server. The client accesses the server by its IP address via port 3000. Each transaction between client and server begins with the client sending a control command. For a description of the control command protocol, see the Control Socket Functions section on page 6.

Server Structure

eZ80® Remote Access Software is built upon two structural organization principles. From an application point of view, it is an integrated system of applications. From an internal structure point of view, it is a set of libraries bringing the application to life in a heterogeneous interoperational fashion. Figure 1 illustrates the structure of the eZ80® Remote Access Software.

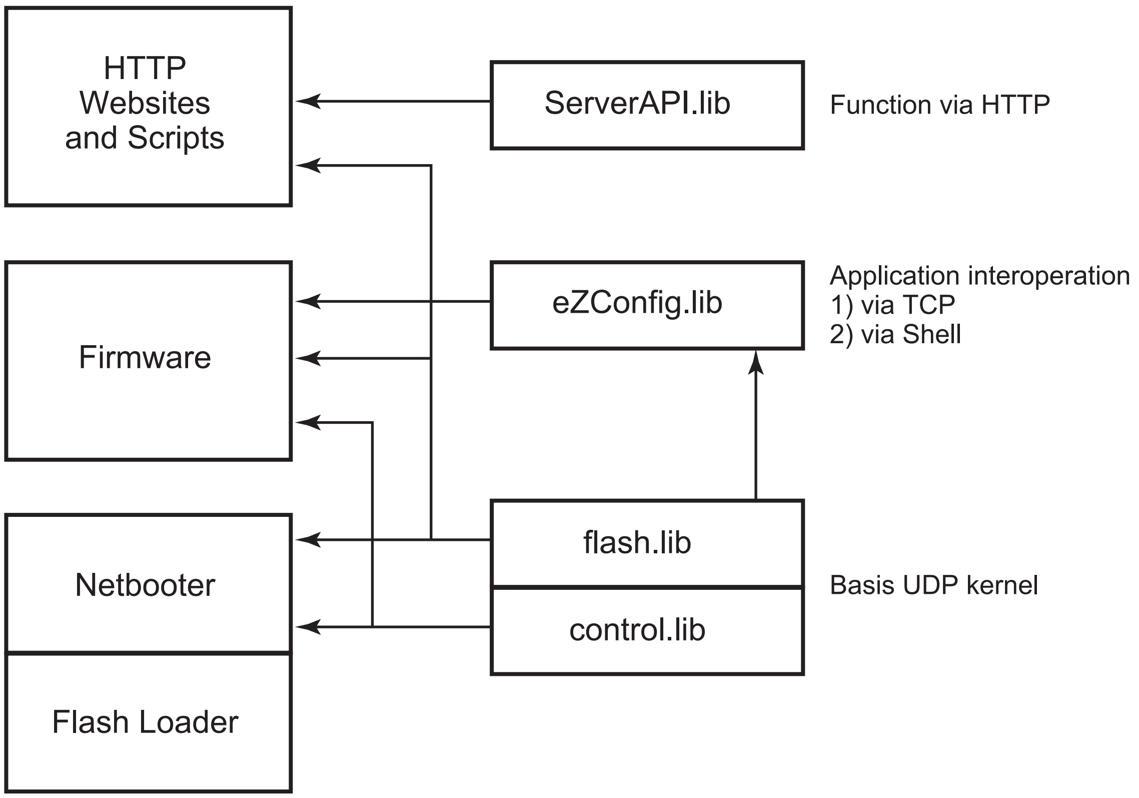


Figure 2. The Server Structure

The complex structure of the eZ80® Remote Access Software is organized on the server side as a set of libraries. These libraries are used to support the following utilities and software:

- Flash Loader
- NetBooter
- User firmware

The libraries encapsulate the features and functions required for server functionality. This functionality is distributed as shown in the following table:

control.lib	Kernel IP transport utilization and remote command processing.
flash.lib	Configuration, firmware, and web page updating.
ezconfig.lib	A configuration utility.
serverAPI.lib	A remote function invocation method.

Although the Flash Loader utility is not a part of eZ80® Remote Access, it is considered to be a part of the software system, and is integrated into the server structure. There are commands that can provide control to the Flash Loader utility when accessed from the command prompt.

The goal of this extensible modular structure is to provide the user with an integrated application development system.

Configuration Files

Configuration files are data structures that are stored in the parameter block of eZ80® Flash memory. During Server startup, these files are read during a bootstrap procedure that launches the services enabled in the structure. The parameter contained in these structure entries is taken as the default setting for the Server.

The configuration files consist of a personality configuration file and a website configuration file. The personality configuration file contains the logical name of the server. The website configuration file contains the location of web pages and a table of contents of the website.

Every time the client application sends a command to update the settings the structure is filled with new data and written to Flash memory. This configuration remains non-volatile until a new change request. The server operates under the XINU core. User processes are determined by the standard code added to main() function. These services must first be initialized. The configuration file is read from Flash memory, and subsequent services, such as DHCP, HTTP, SNMP, TELNET, shell, and TIMEP, are started according to the data from the server configuration file. Only if a corresponding enable flag for a service is set, the service is started.

The server project includes a definable option for enabling LCD support. With LCD functions enabled, the eZ80190 Evaluation Board, when supplied with the Thermostat Application Module and LCD display, shows its IP address in the LCD display. Additionally, the functions from the library of LCD files, `lcd.lib`, are available.

The server project also includes personality and configuration settings for a higher level of communication service. In its current implementation, the personality is known as the name.

Control Socket Functions

The Control protocol is used to establish an exchange of commands between the client and the Server. When the user clicks a button in a client application, this causes the client to send a certain commands to the server over a UDP control channel.

To further develop and customize the control protocol, the user must understand command implementation and format. This section explains the commands implemented by the control protocol. These commands are listed in Table 1, along with their formats and comments on their usage.

Table 1. Control Protocol Commands

Command	Server Routine	Command Description
helo	app_hello	**General Handshake** The server returns an answer string containing its name as an acknowledge.
scan	pscan	**Scanner Request** The answer features the format: 100 <name>, <IP address>, <EMAC address>
rset	reset	Remotely reset the server.
flsh	flash	**Start the netbooter** If the NetBooter application is present in the server's Flash memory, it is given control to enable the firmware update.
name	name	**Ask/set the name of the server** The **name** command without a parameter returns the name of the server. The **name** command followed by a string parameter assigns the server a new name.

Table 1. Control Protocol Commands (Continued)

Command	Server Routine	Command Description
gcon	getcon	**Get the server configuration** The complete configuration structure is returned following file size information. This size data is written into the control socket as Information Code 100. A data socket is immediately created, and the configuration file data is transferred via this socket.
scon	setcon	**Set the new configuration** The configuration structure is populated with data submitted via the data socket. This socket is immediately created and waits for a binary transfer.
gweb	getweb	Get the link list.
sweb	setweb	Download the link list.
page	page	Submit web pages.
test	test	Test function.
help	help	**Help information** A Help display. Example: help `<command_name>`, where `<command_name>` can be any of the control socket functions (names).

For any binary transfers, and transfers of large amounts of data, a data socket is used. The data socket utilizes the connection-oriented TCP protocol to ensure data integrity. The governing control socket protocol supervises the data socket connection.

The Netbooter Utility

This section describes how the user can adjust the NetBooter project to best suit application requirements. The user can set target Server parameters to achieve maximum flexibility when using the NetBooter for remote access purposes. The structure of the project corresponds to the eZ80® Remote Access server structure described previously.

Common Netbooter Files

The following files for the Netbooter utility are common to the eZ80190 and eZ80L92 microprocessors.

netbooter.c. This main() implementation for the Netbooter utility performs the hardware initialization, then launches the ra_init() to activate the eZ80® Remote Access Server. The source file also contains the implementation of the write_data() function that defines the method of data writing to Flash memory for the firmware update.

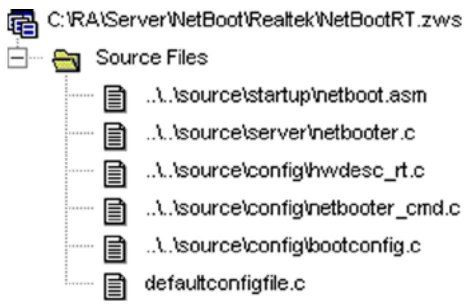
netbooter_cmd.c. This command configuration structure for the Netbooter utility advises the eZ80® Remote Access Server to respond to the following commands:

- rset
- helo
- scan
- fwup
- fwud

The first three bullet items address server control and handshake functions. The remaining two are firmware update protocol commands.

eZ80190 Netbooter Files

The following files for the Netbooter utility are common only to the eZ80190 microprocessor. Figure 3 displays a typical listing of these files.



```
C:\RA\Server\NetBoot\Realtek\NetBootRT.zws
  Source Files
    ..\..\source\startup\netboot.asm
    ..\..\source\server\netbooter.c
    ..\..\source\config\hwdesc_rt.c
    ..\..\source\config\netbooter_cmd.c
    ..\..\source\config\bootconfig.c
    defaultconfigfile.c
```

Figure 3. Netbooter Utility Files for the eZ80190 Microprocessor

netboot_asm. This eZ80190 start-up file defines the bootstrap segment that contains the application header, application name, and the firmware validation check code.

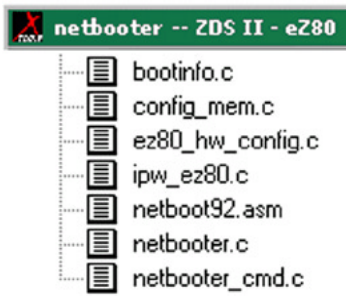
bootconfig.c. This file is the netconfig() function implementation for the Netbooter utility.

defaultconfigfile.c. This file provides the default values for the configuration file.

hwdesc_rt.c. This eZ80190 Realtek memory configuration file is a Flash descriptor of the structure type *TypeFlashDescriptor* defined in the `flashld.h` file. It contains start address and size information about every Flash chip available to the system. The hardware structure **hwdesc** is also determined in this file.

eZ80L92 Netbooter Files

The following files for the Netbooter utility are common only to the eZ80L92 microprocessor. Figure 4 displays a typical listing of these files.


netbooter -- ZDS II - eZ80

bootinfo.c
config_mem.c
ez80_hw_config.c
ipw_ez80.c
netboot92.asm
netbooter.c
netbooter_cmd.c

Figure 4. Netbooter Utility Files for the eZ80L92 Microprocessor

netboot92_asm. This eZ80L92 start-up file defines the bootstrap segment that contains the application header, application name, and the firmware validation check code.

bootinfo.c. This file contains the IPWorks™ bootinfo structure as a simple means of configuration.

config_mem.c. This eZ80L92 memory configuration file is a Flash descriptor of the structure type *TypeFlashDescriptor* defined in the `flashld.h` file. It contains start address and size information about every Flash chip available to the system. The hardware structure **hwdesc** is also determined in this file.

IPWorks™ Netbooter Files

The Netbooter utility uses the following files for version 1.3.3.7 of IPWorks™:

ez80_hw_config.c. This file is the standard IPWorks™ file for eZ80 hardware configuration.

ipw_ez80.c. This file is the standard IPWorks™ file for system configuration.

The eZConfig Server Project

This section describes how the user can adjust the eZConfig project to best suit application requirements. Command shell extensions and configuration files contained in the project are explained. The structure of the project corresponds to the Remote Access server structure described previously.

Common eZConfig Server Files

The following files for the eZConfig Server utility are common to the eZ80190 and eZ80L92 microprocessors.

source\server\data.c. The file governs the implementation of TCP data transfer threads and how received data is written to Flash memory.

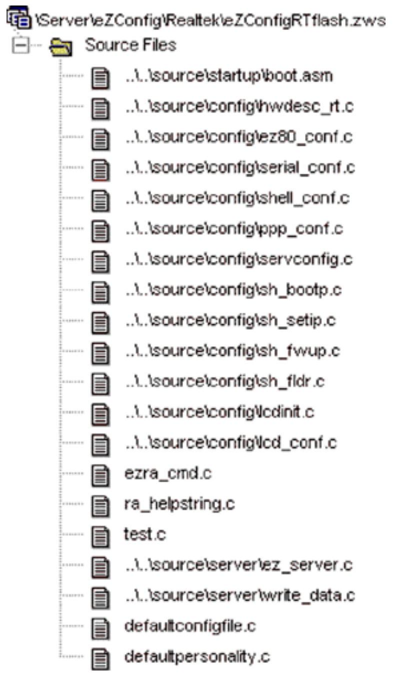
source\server\ez_server.c. This main() implementation for the eZServer performs the hardware initialization, then launches ra_init() to activate the eZ80® Remote Access Server. The source file also contains the implementation of the netconfig() function that starts the services setup in the configuration file.

ezra_cmd.c. This file contains the eZConfig Server control command set.

source\server\write_data.c. This source file contains the implementation of the write_data() function that defines the method of writing data to Flash memory during a web page update.

eZ80190 eZConfig Server Files

The following files for the eZConfig Server utility are common only to the eZ80190 microprocessor. Figure 5 displays a typical listing of these files.



```
\Server\eZConfig\Realtek\eZConfigRTflash.zws
Source Files
    ..\..\source\startup\boot.asm
    ..\..\source\config\hwdesc_rt.c
    ..\..\source\config\ez80_conf.c
    ..\..\source\config\serial_conf.c
    ..\..\source\config\shell_conf.c
    ..\..\source\config\ppp_conf.c
    ..\..\source\config\servconfig.c
    ..\..\source\config\sh_bootp.c
    ..\..\source\config\sh_setip.c
    ..\..\source\config\sh_fwup.c
    ..\..\source\config\sh_fldr.c
    ..\..\source\config\lcdinit.c
    ..\..\source\config\lcd_conf.c
    ezra_cmd.c
    ra_helpstring.c
    test.c
    ..\..\source\server\ez_server.c
    ..\..\source\server\write_data.c
    defaultconfigfile.c
    defaultpersonality.c
```

Figure 5. eZConfig Server Project Files for the eZ80L92 Microprocessor

boot.asm. This eZ80190 start-up file defines the bootstrap segment that contains the application header and application name, along with initialization code.

hwdesc_rt.c. This eZ80190 Realtek memory configuration file is a Flash descriptor of the structure type *TypeFlashDescriptor* defined in the `flashld.h` file. It contains start address and size information about every Flash chip available to the system. The hardware structure **hwdesc** is also determined in this file.

ez80_conf.c, serial_conf.c, shell_conf.c, and ppp_conf.c. The IPWorks™ configuration files required to set up the eZConfig server.

servconfig.c. This module contains the implementation to read and write a boot information block into Flash memory. The boot information block contains the Boot Record field, application-related information, and the network configuration. This boot information block must be read after power-on, and can be modified by the client software.

sh_bootp.c. The BOOTP command shell extensions for the eZConfig server OS shell allow the user to switch the BOOTP client on and off, as follows:

- BOOTP ON—switches the BOOTP on
- BOOTP OFF—switches the BOOTP off

sh_setip.c. The SETIP command shell extensions for the eZConfig server OS shell set IP addressing, as follows:

- SETIP nnn.nnn.nnn.nnn—the default IP address is set to nnn.nnn.nnn.nnn

sh_fwup.c. The FWUP command shell extensions for the eZConfig server OS shell start the Netbooter utility.

sh_fldr.c. The FLDR command shell extensions for the eZConfig server OS shell start the Flash Loader utility.

lcdinit.c. This file is an optional add-in for initializing the LCD display panel on the ZiLOG Thermostat Demo board.

lcd_conf.c. This file is an optional add-in for configuring the LCD display panel on the ZiLOG Thermostat Demo board.

ra_helpstring.c. This file contains control command help functions.

test.c. The test control command callback can be used as a template example of user control command implementation.

source\server\write_data.c. This source file contains the implementation of the write_data() function, which defines the method of data writing to Flash memory during a web page update.

source\server\wplinklist.c: A web page link list structure for the contents of the Server.

defaultconfigfile.c. This file contains the default values for the configuration file.

defaultpersonality.c. This file contains the default values for the personality file.

eZ80L92 eZConfig Server Files

The following files for the eZConfig Server utility are common only to the eZ80L92 microprocessor. Figure 6 displays a typical listing of these files.

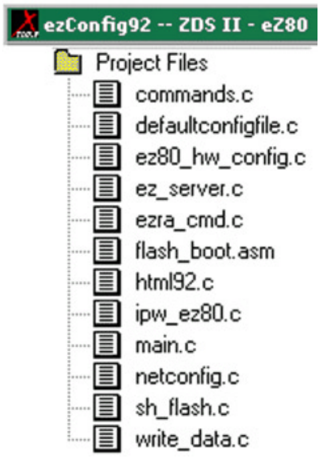

ezConfig92 -- ZDS II - eZ80
Project Files
commands.c
defaultconfigfile.c
ez80_hw_config.c
ez_server.c
ezra_cmd.c
flash_boot.asm
html92.c
ipw_ez80.c
main.c
netconfig.c
sh_flash.c
write_data.c

Figure 6. eZConfig Server Project Files for the eZ80L92 Microprocessor

defaultconfigfile.c. This file contains the default values for the configuration file.

netconfig.c. This file contains the netconfig() implementation.

html92.c. This file represents the default website for the project.

IPWorks™ eZConfig Server Files

The eZConfig Server utility uses the following files for version 1.3.3.7 of IPWorks™:

ez80_hw_config.c. This file is the standard IPWorks™ file for eZ80® hardware configuration.

ipw_ez80.c. This file is the standard IPWorks™ file for system configuration.

The Client Reference

The eZ80® Remote Access Client includes two implementations of the Java client (JClient) and the Windows client (WClient) to provide flexibility to developers using different OS platforms. The typical client software contains four server access modules:

- Network Scan
- Server Configuration
- Firmware Update
- Website Manager

The Network Scan module allows the user to find all remote access servers connected to the local network. It broadcasts the Scan command through the network to all active servers. The client gathers these server responses, which contain IP and EMAC address information and server names. This information is then displayed in a table.

The Server Configuration module manages the configuration of a selected server. It features an interface for editing the default network parameters of the server. The user can enable and disable network services running on this server.

The Firmware Update module performs the download of the firmware to a server via the network. The user selects a firmware `.hex` file. The client processes this `.hex` and transfers the binary image via Ethernet to the server. The server reprograms Flash memory with the supplied data.

The Website Manager module makes it possible to visually edit the contents of the server's website and then update this website on the server side. The website structure is represented in the form of a table or a file list which can easily be edited. After the structure is complete, the client transfers this website information—its structure, web pages, images, and scripts—to the server, replacing prior contents.

The Client Structure

The structure of the eZ80® Remote Access Client is shown in Figure 7.

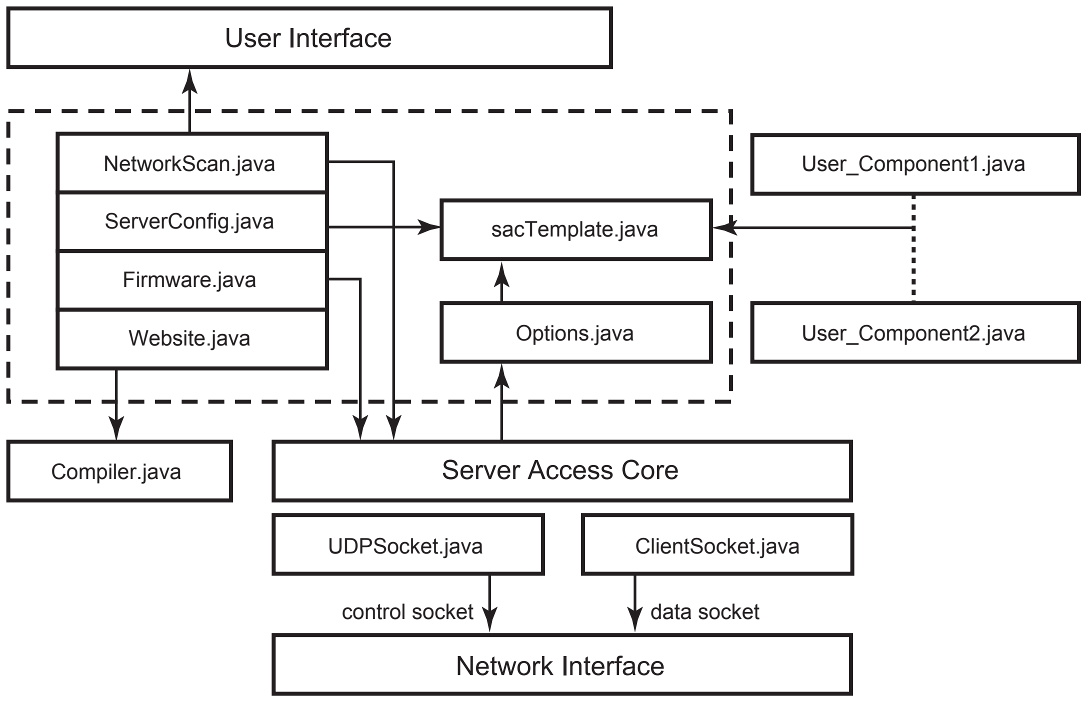


Figure 7. The Remote Access Client

The core mechanism of the client provides for modular application management and supports a networking mechanism. The user interface classes transfer user commands to the core to access the services provided and to display the results in their interface panels.

The advantage of this modular structure allows the user to create custom Java application classes. This advantage allows the user to extend the standard functionality of the JClient. The *sacTemplate.java* class is provided as a template for building the user interface modules that can be easily integrated into the eZ80® Remote Access Client software. As a result, server functionality can be added to the standard server configuration, and the user's project can be tested and controlled from the Java application interface.

eZ80® Remote Access Software Configuration and Installation

This section describes the installation options for the eZ80® Remote Access software, including the required third party software. A step-by-step installation procedure is provided.

Minimum Configuration

A minimum configuration requires that a Java Runtime Environment (JRE) be installed on the client PC. The user must own either the eZ80190 Development Kit or the eZ80L92 Development Kit, which include an eZ80® Development Platform and an eZ80190 Module or an eZ80L92 Module. The eZ80® Development Platform and the client PC must be connected via a LAN network. This configuration allows complete access to the functionality of the software and its use as described in this Application Note.

Complete Configuration

A complete configuration requires the minimum configuration noted above plus the installation of ZiLOG Developer Studio II (ZDS II). Installation of the Java Software Development Kit might be considered for creating custom application modules for the client software in the Java language, available free from Sun Microsystem's Java website.

ZiLOG Developer Studio II

The eZ80® Remote Access Server software and firmware projects can be built using ZiLOG Developer Studio II (ZDS II). Currently, eZ80190 projects require version 3.68 of ZDS II, which can be downloaded from the ZiLOG website. eZ80L92 projects require version 4.10 of ZDS II, which is included in the eZ80L92 Development Kit and is also available on the ZiLOG website. Be sure to check the ZiLOG website for later versions of ZDS II that support eZ80® microprocessors.

Java Virtual Machine

eZ80® Remote Access Client software is a Java application that requires a corresponding Java Machine to be installed on the client computer. This JClient requires the Sun Java Runtime Environment JRE v1.3, available free from Sun Microsystem's Java website. WClient requires the latest version of Microsoft Java VM, which is available free from Microsoft.

Remote Access Software Installation

eZ80® Remote Access software is supplied as a .zip archive. Extract the files contained within this .zip file to a directory on the client PC. The subdirectory structure of the eZ80® Remote Access software is further described in the Remote Access Directory Structure section on page 27.

The eZ80® Remote Access files can be found in the `..\bin` subdirectory. Executables for the client PC are found in the `..\JClient` and `..\WClient` subdirectories.

Getting Started

eZ80® Remote Access software is offered as a stand-alone program and as firmware. The firmware is supported by online updates, with which a new firmware version can be downloaded into Flash memory on the eZ80190 Evaluation Board or the eZ80® Development Platform (for the eZ80L92 device) using JClient. The stand-alone version is loaded into Flash using the External Flash Loader utility via a serial port connection. For more information about the External Flash Loader utility, please refer to the External Flash Loader Product User Guide for eZ80190 (PUG0012), available on zilog.com, for details.

Note: A separate External Flash Loader utility exists for the eZ80L92 microcontroller. It is designed for use with the eZ80® Development Platform and the eZ80L92 Module.

Stand-Alone Version

Load the `eZConfigRTflash.hex` file to the eZ80190 Evaluation Board featuring the Realtek EMAC chip with the aid of the External Flash Loader utility. For the eZ80190 Evaluation Board featuring the Crystal EMAC, use the `eZConfigCS-flash.hex` file. Reset the eZ80190 Evaluation Board. It is now possible to access the eZ80190 Evaluation Board via the client PC.

Note: In the stand-alone version of eZ80® Remote Access, the NetBooter is not required and the firmware update cannot be used.

Firmware Version

The NetBooter is required to support the firmware updates online. NetBooter is first loaded into Flash using the External Flash Loader. The `NetBootRTflash.hex` file is the NetBooter executable for the eZ80® Development Platform with the Realtek EMAC chip. After the eZ80® Development Platform is reset, the NetBooter application launches. NetBooter can be accessed form a Remote Access Client, such as JClient or WClient. JClient is described in this section.

For the Crystal EMAC board, the `NetBootCSflash.hex` version of Netbooter should be used. Use the corresponding `eZConfigRT(CS)Firmware.hex` files as NetBooter firmware. These files can be downloaded using the Firmware Update module of the client.

The eZ80190 Evaluation Board and the eZ80® Development Platform for the eZ80L92 device each contain a serial port labeled *Console*. At the beginning of the remote access evaluation, ZiLOG recommends that both versions of the NetBooter utility use the serial cable connection between this Console port and the COM port of the PC. Use HyperTerminal to observe the activity of the eZ80® Remote Access software during startup after a reset.

Note: A DHCP server, when enabled and running, can provide easy control over the general LAN network setup. See your network administrator for assistance, or refer to one of the many DHCP server software applications available for free trial on the Internet, such as WinRoutePro from Kerio (www.kerio.com).

The following section offers detailed instructions for setting up the stand-alone and firmware versions of the eZ80® Remote Access Server software with the eZ80190 Evaluation Board with a Realtek EMAC. When developing for the eZ80190 device, it is assumed that the eZ80190 Evaluation Board with a Realtek EMAC chip is connected to the network and to the client PC via a serial cable. For projects using the eZ80L92 device, use eZ80® Development Platform and the corresponding files: `netbooterl92.hex`, `ezConfigL92.hex`, and `ezconfig92fw.hex`.

Note: The following instruction applies only to the eZ80190 device.

Procedure for the Stand-Alone Version

1. Program the `eZConfigRTFlash.hex` file into Flash memory using the External Flash Loader utility. Reset the eZ80® Development Platform. Using HyperTerminal, the user can observe the progress of the server software load.
2. Start the JClient application. In the main window of the Remote Access interface, click the **Network Scan** tab. In the **Network Scan** panel, click the **Scan** button to scan the network. The names of the eZ80190 Evaluation Board, IP address, and EMAC address are displayed in the **Network Scan** panel.

Note: The default name for the software that appears in the list is *eZ80® Remote Access Server*.

3. Select all **Network Scan** entries and click on the Folder button to store the IP addresses in the **Network Scan** interface's **Server Address** pull-down menu.
4. From the Remote Access main window, click the **Server Configuration** tab. Using the pull-down menu from the **Server Address** list, select the IP address of a server of choice.
5. Click on the **Get Configuration** button. The fields of the **Server Configuration** panel should populate with the selected server's configuration.
6. Make changes to the server configuration, as appropriate.
7. Click the **Put configuration to the server** button. The status bar should display a message stating that the configuration has successfully updated.

8. Click the **Reset** button to command the server to reboot. After reboot, the server functions using the new configuration parameters.
9. Click on the **Get Configuration** button a second time to observe the changes.

Caution: Changing the server configuration causes the server to accept the new settings after Reset. Caution should be exercised when setting the DHCP, IP address, and subnet mask parameters of the server. Be sure to determine that the server's new parameters are appropriate to the network that supports the client PC.

10. In the **Website Manager** panel, select **File** → **New** to clear the website table.
11. Click the **Add files** button. Navigate to the demo web directory, and select all files in the project's root directory. Click **OK**.
12. Add the six files in the root directory to the project.
13. Click the **Add** button a second time to add the files from the `\feat` directory. These files are added, but the **Web address** column still contains no path information.
14. Select the files that were just added from `\feat` directory. To edit the relative paths, click the **path edit** button. The path selections appear in the drop-down menu.
15. Select the entry `\feat\` and click **OK**. Add the files from the `\image` directory and adjust their paths in the same manner.
16. To access a Server that is listed only in the **Web address** column, the IPWorks™ HTTP implementation requires an entry that contains a slash only, e.g.: "/". Select the `index.html` entry and double-click its name in the **Web address** column. The name can now be edited.
17. Edit the entry to leave only the slash character.
18. The website is now ready for upload. Click the **Generate Website** button, reset the Server, and check the results.

Procedure for the Firmware Version

1. Program the `NetBooterRTflash.hex` file into Flash memory using the External Flash Loader utility. Reset the eZ80® Development Platform.
2. Start the client application. Go to the **Network Scan** tab and click the **Scan** button to scan the network.. The names of the eZ80® Development Platform, IP address, and EMAC address are displayed in the **Network Scan** panel.

Note: The default name for the software that appears in the list is *eZ80®* Remote Access *Server*.

3. Select all **Network Scan** entries and click on the Folder button to store the IP addresses in the **Server Address** pull-down menu.
4. In the main Remote Access window, click the **Firmware Update** tab. The **Firmware Update** panel appears.
5. In the **Firmware Update** panel, click the **Load HEX** button to launch the **Open** dialog box. Using a web browser, find the `eZConfigRTFirmware.hex` file in the `/bin` directory. Click the **Open** button.
6. Wait until the status bar completes and the hex file is ready for download.
7. Click the **Firmware Update** button and wait until the file is transferred to the server. The server resets automatically and starts the newly-downloaded firmware.
8. Repeat Steps 2 through 3 to perform a network scan. Note the new name in the entry.
9. Configure the server using Steps 4 through 8 above.
10. Perform a website update using Steps 10 through 18 in the previous section.

Note: For further details and instructions, please refer to the The JClient User Guide section of this document on page 20. Additionally, the `notes.html` file provided with the eZ80® Remote Access application lists the latest software updates.

JClient

The JClient application is a Java implementation that can run on any platform equipped with a Java Runtime Environment. The JClient consists of a set of application modules. The advantage of the modular approach for the client application is that the interface can be easily changed or adjusted for new tasks and new components can be flexibly added, be set up, and run together. As a consequence, the user can create and install his own Java modules that can inherit and extend the functionality of the eZ80® Remote Access software for the purpose of solving specific tasks.

Each of the application modules use the ClientSocket and UDPSocket APIs to gain access to the server. Every user application module must be implemented within the Java application. The Java implementation for eZ80® Remote Access is named *com.zilog.components*. This name corresponds to the location of the user files, and is based on the *sacTemplate* template class. The Server Access core reads all modules built on the *sacTemplate* from the *com.zilog.components* package and displays them as windows within the Java application.

The JClient User Guide

This section describes all of the components and modules of the eZ80® Remote Access software. It is assumed that the network includes an eZ80® Development Platform with eZ80® Remote Access Server software running on it and a local PC with an eZ80® Remote Access Client installed and running.

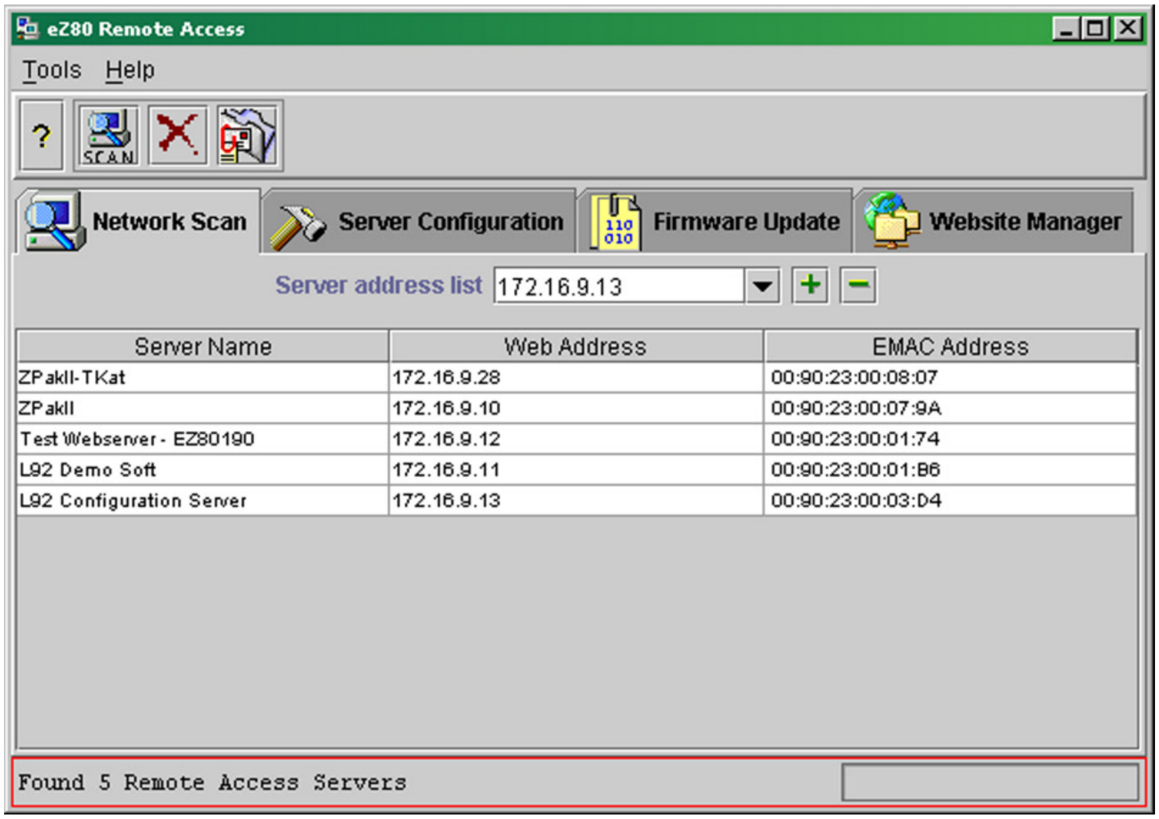
The pages that follow describe the JClient modules in the list below.

- Network Scan
- Server Configuration
- Firmware Update
- Website Manager

Network Scan

Each eZ80® Remote Access Server is detected on the network by the **Scan** command. This Network Scan client module is the user's first client software selection.

Click the **Network Scan** tab in the eZ80® Remote Access Server interface to display the **Network Scan** panel, as shown in Figure 8. The central component of this pane is the server table. The server table is populated with data upon scanning the network. The table contains the names, IP, and EMAC addresses of the servers found on the network. A list of server addresses is available in a pull-down menu for easy selection.



Server Name	Web Address	EMAC Address
ZPakII-TKat	172.16.9.28	00:90:23:00:08:07
ZPakII	172.16.9.10	00:90:23:00:07:9A
Test Webserver - EZ80190	172.16.9.12	00:90:23:00:01:74
L92 Demo Soft	172.16.9.11	00:90:23:00:01:B6
L92 Configuration Server	172.16.9.13	00:90:23:00:03:D4

```
Found 5 Remote Access Servers
```

Figure 8. Network Scan Panel of the eZ80® Remote Access Server Interface

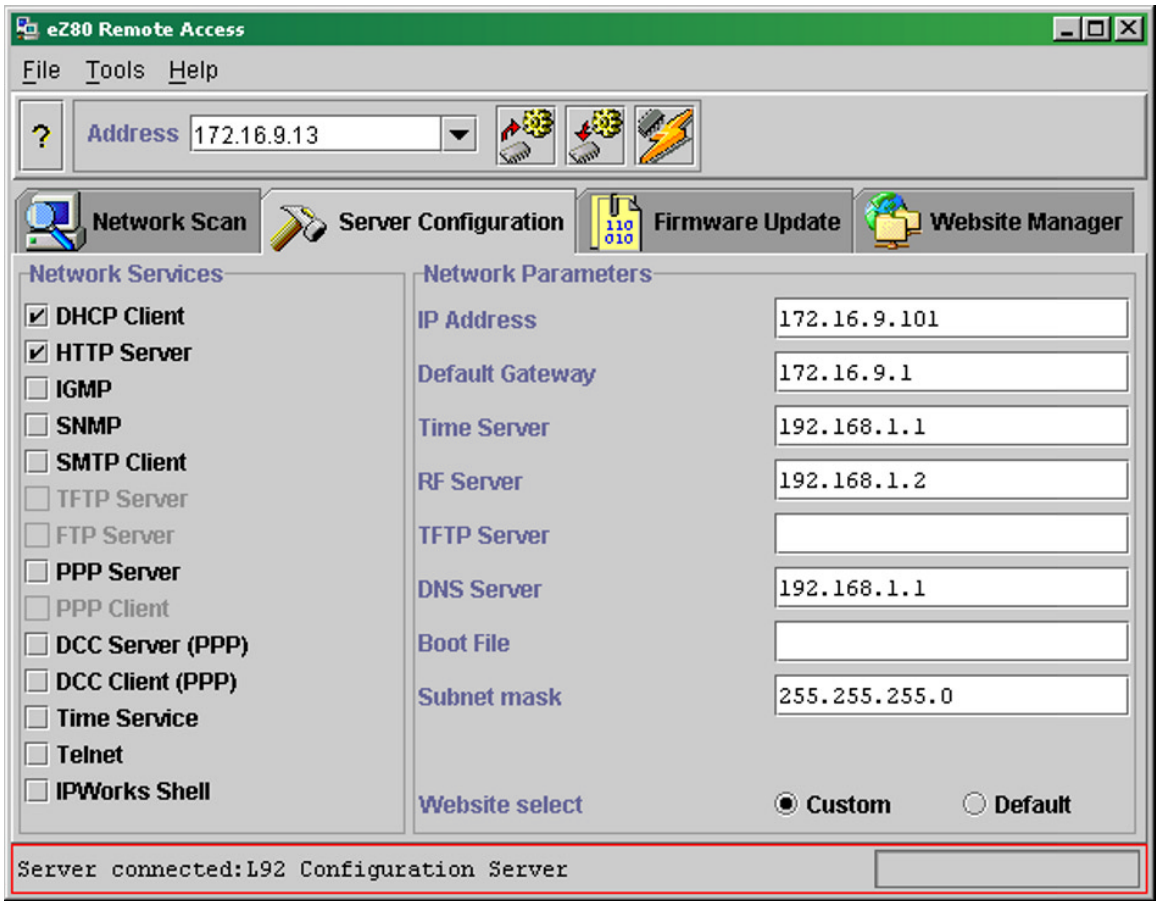
Click the **Scan** button in the upper left of this window to issue the **Multicast Scan** command and to display a list of available servers. The scan is only possible if the network address is set in the **Address** field.

The **Folder** button writes the selected server entries from the scan table to the list of active server IP addresses. This list is available in the form of a pull-down menu that exists in all tab windows of the Remote Access interface for easy user access.

Work Principle. The client broadcasts a **Scan** command to the network. Each Remote Access Server replies to that command with the response `220 Name, address`. The client gathers the data and places it into the server list.

Server Configuration

Click the **Server Configuration** tab **in** the eZ80® Remote Access Server interface to display the **Server Configuration** panel, as shown in Figure 9.



Address 172.16.9.13

Network Services

- [✓] DHCP Client
- [✓] HTTP Server
- [ ] IGMP
- [ ] SNMP
- [ ] SMTP Client
- [ ] TFTP Server
- [ ] FTP Server
- [ ] PPP Server
- [ ] PPP Client
- [ ] DCC Server (PPP)
- [ ] DCC Client (PPP)
- [ ] Time Service
- [ ] Telnet
- [ ] IPWorks Shell

Network Parameters

IP Address	172.16.9.101
Default Gateway	172.16.9.1
Time Server	192.168.1.1
RF Server	192.168.1.2
TFTP Server	
DNS Server	192.168.1.1
Boot File	
Subnet mask	255.255.255.0

Website select [✓ Custom] Default

```
Server connected:L92 Configuration Server
```

Figure 9. Server Configuration Panel of the eZ80® Remote Access Server Interface

As shown in Figure 9, the **Network Services** pane contains a list of TCP/IP services and the controls that are used to enable or disable each particular service.

The **Network Parameters** pane contains a list of TCP/IP BootInfo entries, such as the IP address of the server, the network gateway, the subnet mask, etc. The user can modify values using editing controls.

The **Website select** control determines which website is displayed—one supplied with the Firmware `.hex` file, or one updated during a website update session. The website contained in the `.hex` file is called the *default* website for that firmware. The web page downloaded using the website manager is termed a *custom page*.

The following passages describe the three icons at the top of the interface depicted in Figure 9, reading from left to right.

The **Get Configuration** button requests the current configuration from the selected server. When this configuration is received by the server, the network services and parameters are displayed in the **Server Configuration** view panel. In the **Network Services** pane, a checked checkbox indicates that a network service is enabled. Those services in bold type but without a checked checkbox are available but not enabled. Those greyed out are not available.

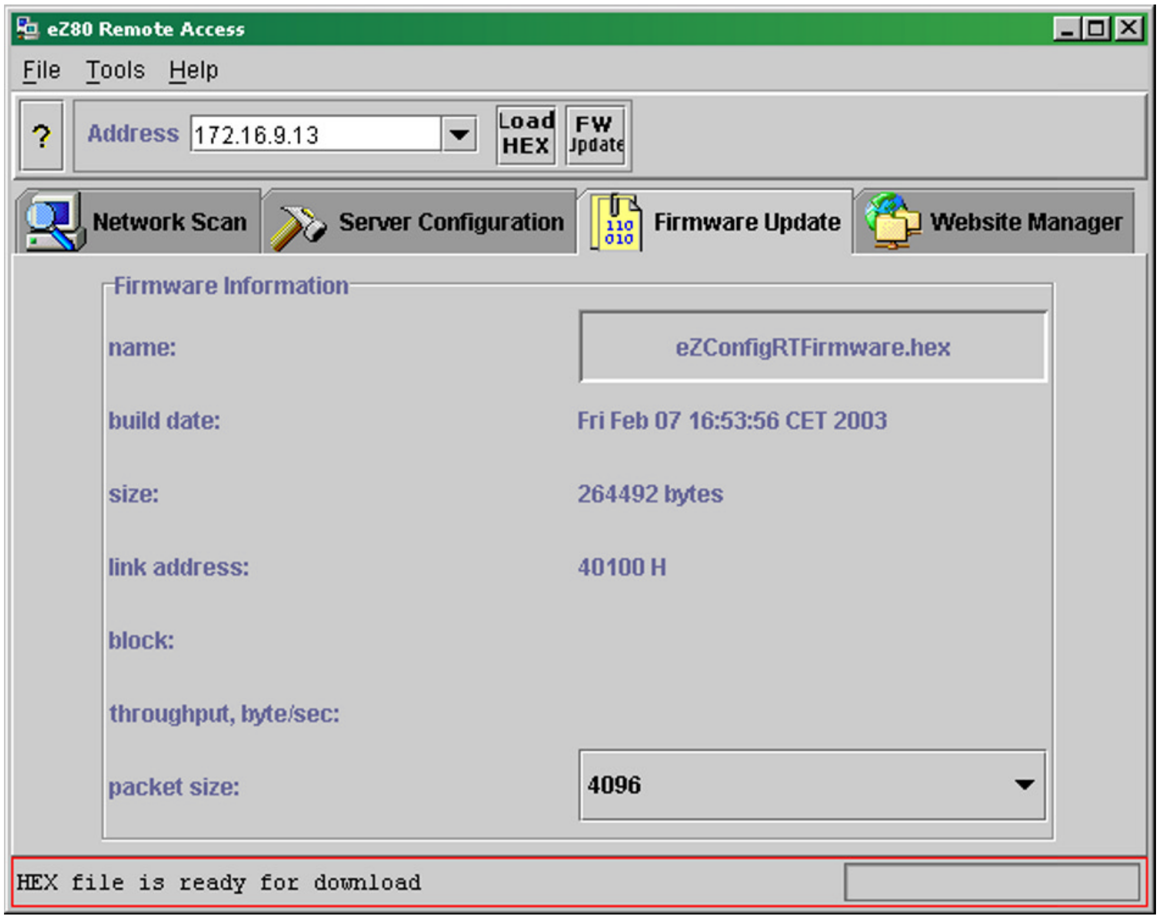
The **Set Configuration** button submits modified parameters to the server. These modified parameters compose a new configuration file on the server side and store this file in Flash memory.

The **Reset** button sends a command to reset the server. This command is necessary to remotely restart the server with a new set of parameters.

Work Principle. A server configuration is written into Flash memory at a certain address as a configuration file. This file is formatted with a validity flag and a checksum. When started, the server attempts to read the information contained in the configuration file. If this information is not valid, the default configuration stored in program code is used. When the client issues a **gcon** command (see Table 1) via UDP socket 3000, the server transfers the contents of the file using TCP socket 3001. The client must access this socket to read the information requested. When the client issues a **scon** command (see Table 1), the server prepares a TCP socket for accepting the new configuration file data. The client must connect to the TCP socket and write the new information to the server. The server reads this information, generates a configuration file, and stores this file into Flash memory.

Firmware Update

Click the **Firmware Update** tab in the eZ80® Remote Access Server interface to display the **Firmware Update** panel, as shown in Figure 6. The panel displays the statistics of the currently-selected firmware: the filename, the start-up address, the link address, and the size in bytes. During an update, the average throughput rate is displayed. The user can set the packet size of the transferred data to the server.



Address 172.16.9.13

Firmware Information

name:	eZConfigRTFirmware.hex
build date:	Fri Feb 07 16:53:56 CET 2003
size:	264492 bytes
link address:	40100 H
block:	
throughput, byte/sec:	
packet size:	4096

```
HEX file is ready for download
```

Figure 10. Firmware Update Panel of the eZ80® Remote Access Server Interface

The **Load HEX** button opens a file dialog, within which the user can select a file containing the appropriate version of the firmware. When selected and subsequently read, the firmware file creates a binary image to produce pure binary data, thereby accelerating the transfer.

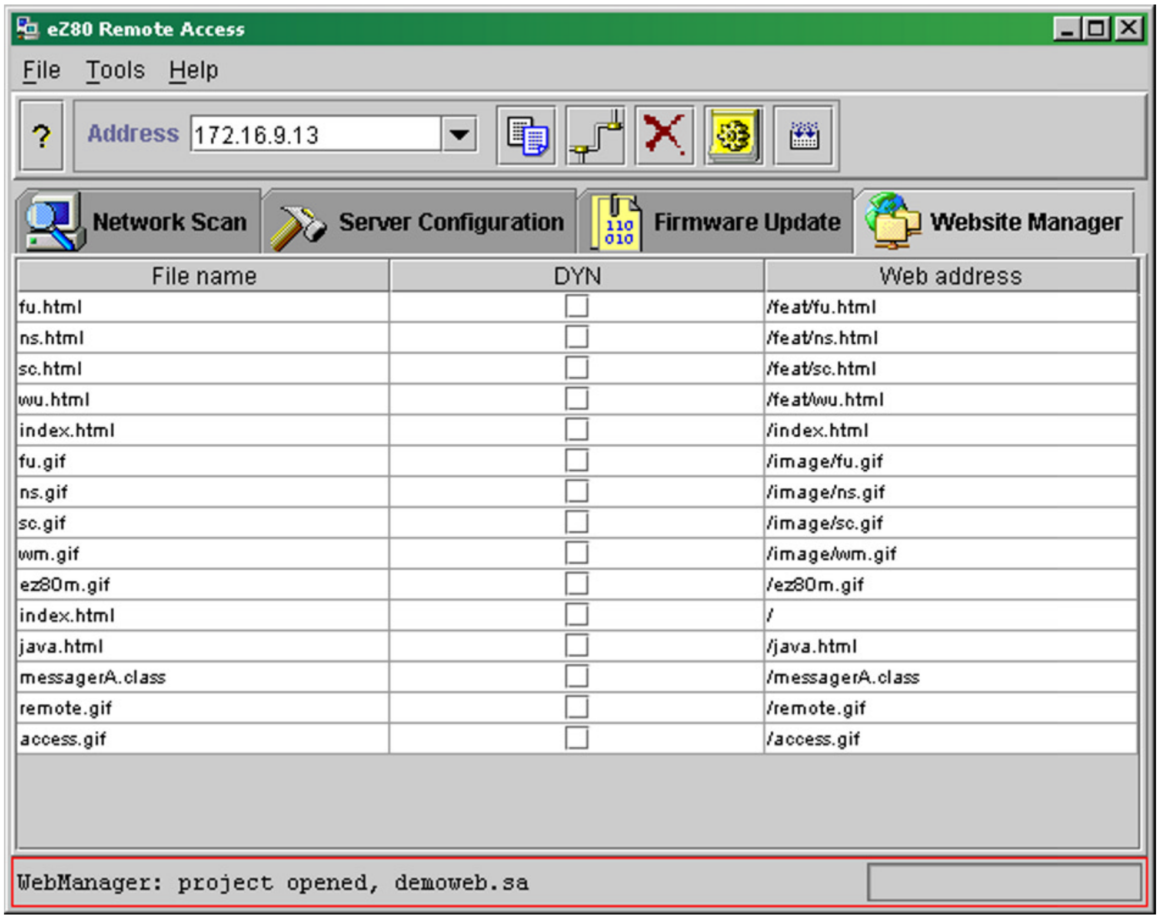
The **Firmware Update** button initiates the firmware update.

Work Principle. The write sequence starts with the **Firmware Update** command. On receipt of this command, the firmware application cedes control to the NetBooter. After the NetBooter starts, the client senses its presence by using scan commands. The client then issues a *page* command without parameters. On receipt of this command, the server returns firmware linking and download buffer size information. The contents of application memory are erased and this memory is prepared for the download of the firmware.

Note: The NetBooter application must be present in the program memory of the Server to enable firmware updates. It implements the same method of communication to the client as other Remote Access Server applications. The NetBooter provides a secure way of remote firmware updates.

Website Manager

Click the **Website Manager** tab in the eZ80® Remote Access Server interface to display the **Website Manager** panel, as shown in Figure 11. This panel lists a table of the files that comprise the website. The left column contains the original file names, the center column allows the user to check whether the file is a script or a static page, and the right column displays how these names are represented on the website. This column can be edited.



File name	DYN	Web address
fu.html	☐	/feat/fu.html
ns.html	☐	/feat/ns.html
sc.html	☐	/feat/sc.html
wu.html	☐	/feat/wu.html
index.html	☐	/index.html
fu.gif	☐	/image/fu.gif
ns.gif	☐	/image/ns.gif
sc.gif	☐	/image/sc.gif
wm.gif	☐	/image/wm.gif
ez80m.gif	☐	/ez80m.gif
index.html	☐	/
java.html	☐	/java.html
messagerA.class	☐	/messagerA.class
remote.gif	☐	/remote.gif
access.gif	☐	/access.gif

```
WebManager: project opened, demoweb.sa
```

Figure 11. Website Manager Panel of the eZ80® Remote Access Server Interface

The following passages describe the first, second, and fourth icons at the top of the interface depicted in Figure 11, reading from left to right.

The **Add** button adds files to the website table. When clicked, the **Open File** dialog appears, allowing selection of various types of web files.

The **Path Edit** button is used to adjust the pathname of a selected file for the relative pathname of that file in the website structure. For best results, the original files on the disk should be placed in the same relationship to the root or to the index file as is assumed by the website structure. Proper file placement guarantees that the final web file placement on the eZ80® embedded HTTP server corresponds to the structure.

The **Generate Website** button finalizes website composition, makes a connection to the Server, and, if successful, transfers website data to the server, where it is written to Flash memory.

To use the website manager requires updating the website contents with the website structure of web files and path names so that it can be displayed. A project can be created on the local hard drive, launched, and viewed locally using an Internet browser application. The files are stored in subdirectories for the user's convenience. The start page of the project is named `index.html`. This file is located in the demo web subdirectory of the `\doc` directory. The `\feat` subdirectory contains demo pages and the `\image` subdirectory contains images.

Note: To start working with the website update function, select the **Website Manager** tab and navigate to **Tools** → **Options** (or press Ctrl–O) to open the **Options** panel for the module. In the **Options** panel, enter the path names for the folders where IPWorks™, the eZ80® C Compiler, ZDS II, and the server software are installed. These path names are required for the functionality of the website update component.

To prepare a website for uploading files, the following instruction offers guidance.

Work Principle. When provided with the web files and their relative path names, all of the information necessary to compose the website structure is obtained. The structure is written into the Server using the *set web* UDP command. Binary data is transferred using the TCP connection in the same manner as configuration updates. After transfer, the contents of all files are written to the Server's Flash memory. The *page* command is used for this file transfer.

Demonstration Applications

ZiLOG offers three demonstration applications to allow the user to become accustomed to the eZ80® Remote Access software on a step-by-step basis. These executables are described below.

The Remote Control Demo

This firmware uses the `control.lib` library file to start the Remote Access (RA) control thread by calling the **ra_init()** function. The Control Demo firmware can process the **Scan** command, the **Name Change** command, and the **Firmware Update** commands.

The Dynamic IP Demo

A number of users wanted to be able to change IP addresses at their convenience. The Dynamic IP Demo enables this capability. The demo uses the `ezconfig.lib` library file supported by the `flash.lib` library file to provide configuration read, modify, and store features.

The OS Demo

This simplest of the three IPWorks™ demos is converted for eZ80® Remote Access to be used in firmware downloads. The project is set up to be linked at ROM address `40100h` to allow the NetBooter to write the application into Flash memory. As the reader may recall, the NetBooter resides in the address range `8000h–30000h`. In essence, the first free Micron Flash block is selected for the firmware start at address `40000h`. `100h` bytes are occupied by the application header.

The OS Demo features a small file size making download time using the NetBooter very short. After the NetBooter cedes control to the firmware, a HyperTerminal window can be used to enter shell commands.

Two of these shell commands, *fwup* and *fldr*, are implemented in this demo, and used to cede control back to the NetBooter or to the External Flash Loader, respectively. The sources of these two commands are supplied with the project. These sources help the user to understand the structure of the application header and the meanings of entries.

Note: TCP/IP layer functionality is not enabled in the OS Demo application. The serial connection to the Console port of the eZ80® Development Platform is required when using HyperTerminal.

Each of these executable files can be found in the `/bin` directory and on ZiLOG's Software Tools/Downloads page.

Remote Access Directory Structure

The eZ80® Remote Access directory structure is shown in Figure 12.

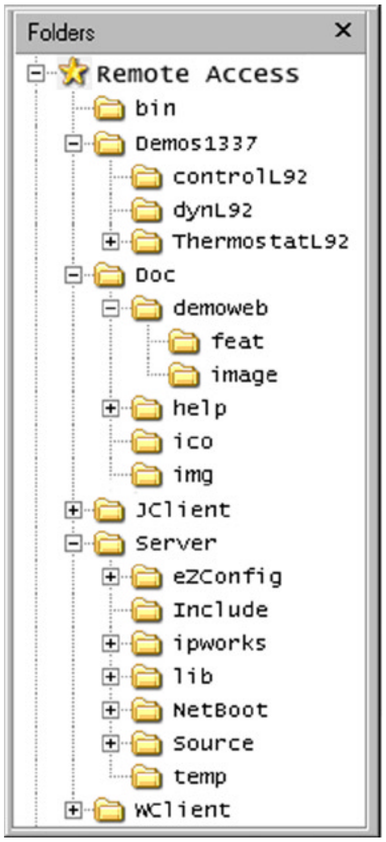


Figure 12. [The eZ80® Remote Access Directory Structure

File Locations

The project and source files for the demonstration projects are placed into the `/demos` directory. Documentation and description files are located in the `/doc` directory. The client Java classes hierarchy and the JClient executable are located in the `/JClient` folder. The `/server` directory contains source files and projects for the different components of the eZ80® Remote Access Server. The include files for most of the projects are located in the `/Server/Include` directory.

Other include files are located in the IPWorks™ and eZ80® C Compiler directories, respectively. Settings for these include files must be adjusted by the user accordingly for each of the projects. The remote server libraries are located in the `/Server/lib` directory.

The `/bin` directory contains eZ80® Remote Access executables in the hexadecimal file format. These files can be used as described in the Getting Started section on page 16. The code size of the eZ80® Remote Access executable, not including the standard IPWorks™ libraries, is 22 KB for the eZConfig project and 7 KB for the NetBooter project.