Application Note

A Serial-to-TCP Gateway for eZ80Acclaim*Plus!*™ Connectivity ASSP

AN016007-0708

Abstract

This application note describes Zilog's eZ80®-based serial-to-TCP gateway that transfers data between a UART interface and an Ethernet interface. A web-server powered by the eZ80F91 MCU and running the Zilog TCP/IP (ZTP) software suite can accept incoming data from a LAN and send it to a serial port to make the web-server function as a serial-to-TCP gateway. The data entering through a TCP/IP channel is converted into a series of bytes and then sent to a UART device.

Note: *The source code files associated with this application note,* `AN0160-SC01.zip` *and* `AN0160-SC08.zip` *(TCP client module), are available for download at* www.zilog.com.

Zilog® Product Overview

This section provides a brief overview of the Zilog products used in this application note, which include eZ80Acclaim*Plus!*™ Connectivity ASSP and the ZTP software suite.

eZ80Acclaim*Plus!*™ MCU Family Overview

The Flash-based eZ80Acclaim*Plus!* MCUs are suited for designing high-performance embedded applications. With execution speeds up to 50 MHz and an On-Chip Ethernet MAC, you can execute complex applications supporting networking functions quickly and efficiently. Combining On-Chip Flash and SRAM, eZ80Acclaim*Plus!* devices provide the memory required to implement communication protocol stacks and achieve flexibility when performing in-system updates of application firmware.

Zilog TCP/IP Software Suite Overview

The ZTP integrates a rich-set of networking services with an efficient real-time operating system (RTOS). The operating system is a compact preemptive multitasking, multithreaded kernel with inter-process communications (IPC) support and soft real-time attributes. Table 1 lists the standard network protocols implemented as part of the embedded TCP/IP protocol stack in ZTP.

Table 1. Standard Network Protocols in ZTP

HTTP	TFTP	SMTP	Telnet	IP	PPP
DHCP	DNS	TIMEP	SNMP	TCP	UDP
ICMP	IGMP	ARP	RARP		

Many TCP/IP application protocols are designed using the client-server model. The final stack size is link-time configurable and determined by the protocols included in the build.

Discussion

Web-enabling a device that serves as a source of data to an external processing device is convenient using the Internet. Often, the output of the device is through a serial-UART-compatible channel. This data output can be a continuous stream or a series of data packets offering greater reliability. However, a remote device featuring a UART communications port can contain information that can be processed by another external processing device, such as a CPU. Sending information to this external processing device can be accomplished by using a web-enabling device such as a serial-to-TCP gateway. This method of information transfer results in a throughput performance that offers a significant improvement over current communication methods, such as modems or low-end Integrated Services Digital Network (ISDN).

Theory of Operation

Any device with a serial UART channel can be connected to a gateway to copy data using a TCP/IP connection. However, to ensure proper reception, the serial data from the UART must be converted into TCP packets of a specific format. Figure 1 displays a functioning serial-to-TCP gateway system.

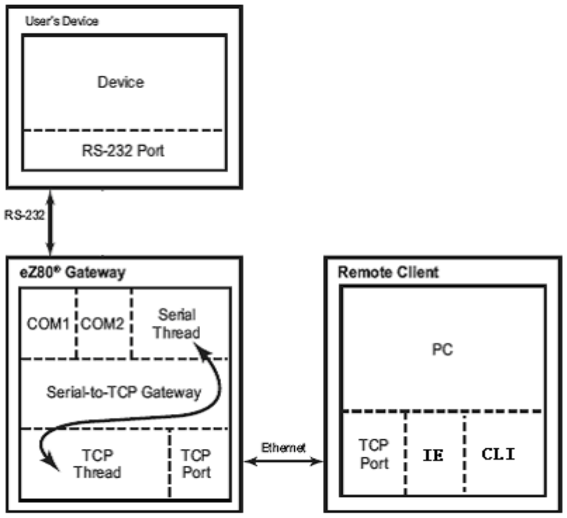


Figure 1. Block Diagram of Serial-to-TCP Gateway

The function of the eZ80 serial-to-TCP gateway is to transfer data from a client PC to a serial communication port on the user's device. One end of the eZ80 device is connected to an HTML (web) page using the CGI function interface, and the other end is connected to a serial device using the UART driver interface. One end of the eZ80F91 web server transmits/receives data from the HTML (web) page using the CGI function interface. The other end of this web-server is connected to the serial device (HyperTerminal) to transmit and receive data from this end, using the UART driver interface. The gate is transparent, and the client operates as if it is working directly with the user's device.

The client program is implemented in HTML.

Description of Components

This application note implements the following modules:

- Network scan module

  This module uses the IGMP protocol to join a multicast group 226.2.2.2. After that, it opens a UDP port 4111 and waits to receive a 'discover' command from the HOST. When it receives a 'discover' request, then it sends its IP address, EMAC address, and the name of the device to the system that sent the request. Any other messages on the port are ignored.

- Online configuration module

  The online configuration module requires data persistence to be enabled; the configuration settings are stored in the Flash Info page of the eZ80F91 device. At boot-up, these configurations are read, and correspondingly the values are set. The eZ80F91 opens a TCP socket on port 4000 and listens for a connection.

  After a connection is accepted on the listening socket, the client is authenticated by a common shared string (password). If the client is authenticated, the eZ80F91 checks for the requested command.

  If the command is 'QUERRY,' then eZ80F91 replies with the current configuration of the system. For the 'QUERRY' command, the configuration that is sent to the CLI (command line interface) utility is the one stored in the Flash Info page. If DHCP is enabled, then the IP parameters assigned are the ones provided by the DHCP Server.

  If the command is 'SETPAR,' then eZ80F91 reads the configuration parameters sent along with the command. These parameters are then stored in the Flash Info page, and the system is rebooted for the new configuration parameters to take effect.

- Serial interface module

  The serial interface module uses the HyperTerminal application as a serial input/output device. Whenever the user presses a key, the **Serial Read** thread continuously reads the data from the UART driver and stores this data in the **Current** buffer. If the **Current** buffer is full or if presses the **Enter** key, this data is transferred from the **Current** buffer to the **TCP** buffer. The **TCP** window reads the data from the **TCP** buffer and updates the **Serial Read** window. The **data-upload** CGI function uploads the data to the HTML page after receiving repeated requests from the browser.

- TCP interface module

  The TCP interface module uses the **TCP** window (HTML web page) that contains two separate windows for writing and reading data. After clicking the **Submit** button, the **TCP Write** window transfers the data to the buffer using the CGI function interface. The **TCP Read** window is updated automatically and continuously. The **TCP-to-serial** CGI function interface reads the data from the HTML page on receiving a request from the browser. This browser request is generated after the user clicks the **Submit** button in the **TCP Write** window. The browser request updates the **serial** buffer with the current received data and uploads the data to the serial driver for transmission over the serial link (HyperTerminal in this case).

Quick Start

The features of the serial-to-TCP application can be demonstrated using an eZ80 development platform and a PC. The PC's COM (serial) port represents the RS-232 compatible data port of the embedded device. In addition, a terminal program on the PC, such as the HyperTerminal, represents the embedded application requiring serial data to be exchanged with another device. In this demonstration, the serial-to-TCP gateway connects the HyperTerminal program to the peer-end HTML application. The serial-to-TCP application uses the same PC to host the HyperTerminal and web page applications; however, such usage is not a requirement. The web page can run on any networked PC that can connect to the serial-to-TCP gateway. This PC can be located anywhere—down the hall or on the other side of the planet. Figure 2 displays the hardware configuration of the laboratory demonstration.

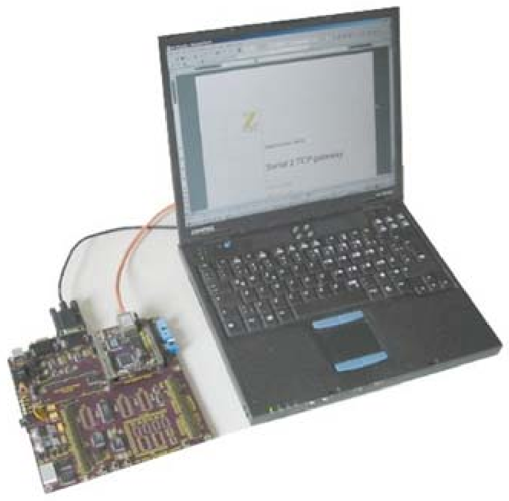


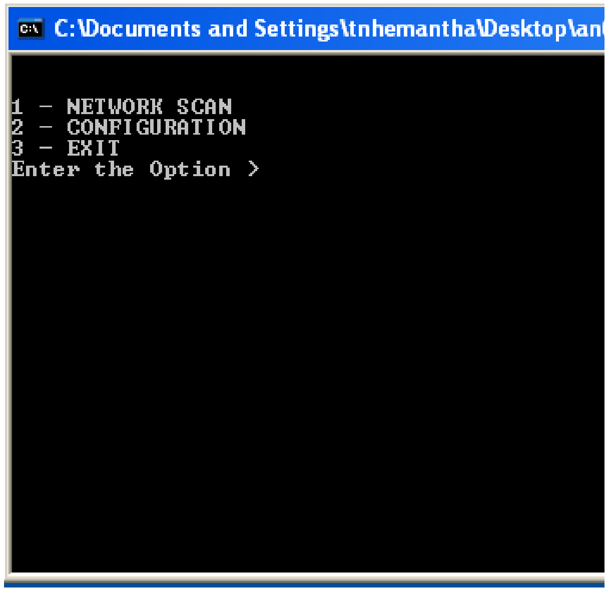
Figure 2. Serial-to-TCP Gateway Setup

PC Host

The PC host consists of two modules: a network scan application and an HTML webpage.

Network Scan

This module creates a UDP socket and sends a 'discover' command to a multicast group address 226.2.2.2 with port 4111 and waits to receive a reply. When eZ80 responds with the name, IP address, and MAC address, this module describes information on the terminal as in Figure 3 on page 4. It also allows to remotely configure the gateway. For more details, see Procedure on page 9.

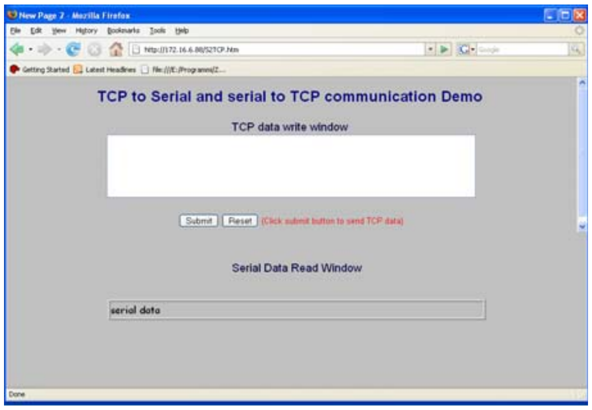


```
C:\Documents and Settings\tnhemantha\Desktop\an

1 - NETWORK SCAN
2 - CONFIGURATION
3 - EXIT
Enter the Option >
```

Figure 3. Network Scan Module

TCP Client

The web page interface is used as the TCP client for the serial data. The data coming from the serial device is displayed on the web page as in Figure 4. The data can be entered on the TCP data write window. When presses the **SUBMIT** button, the data is sent to the serial device.


New Page 2 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://172.16.6.88/S2TCP.htm
Getting Started Latest Headlines file:///E:/Programs/Z...
Submit
Reset
Done

TCP to Serial and serial to TCP communication Demo

TCP data write window

(Click submit button to send TCP data)

Serial Data Read Window

serial data

Figure 4. TCP Client

Developing the Serial-to-TCP Gateway Application

The serial-to-TCP gateway functions as a server in a client-server application. The server provides the serial connection, converts the data into TCP packets, and maintains the TCP connection. The client, which operates only with TCP, establishes the connection to the server and performs terminal emulation on a TCP stream. Configuring the hardware is described in Quick Start on page 3.

Except for the power requirement, the eZ80 development platform is ready to use in a serial-to-TCP application. For information on power requirements, refer to *eZ80F91 Module Product Specification (PS0193)*.

Adding and Integrating Serial-to-TCP Gateway Files to ZTP

The serial-to-TCP gateway described in this application note requires the eZ80Acclaim*Plus!* development board and the ZTP. For the serial-to-TCP gateway execution, the files specific for the demo must be added and integrated to the ZTP stack before it is downloaded onto the eZ80Acclaim*Plus!* development board. This section contains the details of adding the serial-to-TCP gateway files to the ZTP stack.

The serial-to-TCP gateway files that must be added to the ZTP project files are in the `AN0160-SC01.zip` file available for download at ww.zilog.com. The demo files are of the following types:

- C (`*.c`) files
- HTML (`*.htm`) files

The ZTP stack is available on ww.zilog.com and can be downloaded to a PC with a user registration key. or ZTP can be installed in the specified location; the default location is `C:\Program Files\ZiLOG`.

Follow the steps below to add and integrate the demo files to the ZTP stack:

1. Download ZTP to your PC, browse to the location where ZTP is downloaded, and open the `...\ZTP_2.1.0\ZTP\SamplePrograms\ZTPDemo` folder.
2. Download the `AN0160-SC01.zip` file and extract its contents to a folder on your PC. The following folders are extracted:

   ```
   \ serial2TCPDemo
   \ serial2TCPWebsite.Acclaim
   ```
3. Select and copy all of the `*.html` files and folder in the `\serial2TCPWebsite.Acclaim` folder and paste them into the `...\ZTP_2.1.0\ZTP\SamplePrograms\Website.Acclaim` folder.
4. Select and copy all of the `*.c and *.h` files located in the `\serial2TCPDemo` folder and paste them into the `...\ZTP_2.1.0\ZTP\SamplePrograms\ZTPDemo` directory.
5. Launch ZDS II and open the project file, `website.zdsproj`, located in the following path: `...\ZTP\SamplePrograms\Website.Acclaim`.
6. To add all the `*.htm` files located in the `...\ZTP_2.1.0\ZTP\SamplePrograms\Website.Acclaim` folder to the project, right-click the project and then select **Add files to project**. The following files need to be added:
   - `call_cgi.htm`
   - `S2TCP.htm`
   - `serial_to_tcp.htm`
7. Open the `website.c` file from within ZDS II and enter the following prototype declarations into it:

   ```
   // HTML pages
   extern const struct staticpage call_cgi_htm;
   extern const struct staticpage serial_to_tcp_htm;
   extern const struct staticpage S2TCP_htm;
   // CGIs
   extern INT16 S2TCP_cgi(struct http_request *request);
   extern INT16 SerialRead_cgi(struct http_request *request);
   ```
8. The `website.c` file contains an array, `Webpage website[]`, that provides information about the HTML pages. Replace the last line of the array, `{0,NULL, NULL, NULL }`, with the following lines of code:

   ```
   {HTTP_PAGE_STATIC, "/S2TCP.htm", "text/html", &S2TCP_htm },
   {HTTP_PAGE_STATIC, "/serial_to_tcp.htm", "text/html", &serial_to_tcp_htm },
   {HTTP_PAGE_DYNAMIC, "/cgi-bin/serial_to_tcp", "text/html",(struct
   staticpage*)S2TCP_cgi },
   {HTTP_PAGE_STATIC, "/call_cgi.htm", "text/html",&call_cgi_htm },
   {HTTP_PAGE_DYNAMIC, "/cgi-bin/call_cgi", "text/html",(struct staticpage *)
   SerialRead_cgi },
   {0, NULL, NULL, NULL }
   ```

9. From within ZDS II, open the `left.htm` file under `\Web Files`. Search for `CGI Calculator` and locate the following line:

```
&nbsp;&nbsp; <a href="cgif.htm" target="_top">CGI Calculator</a></
font><p><font face="Verdana" size="1"><b>Site Info<br>
```

10. Replace `</font>` with the following piece of HTML code to create a link from the default eZ80Acclaim*Plus!* web page to the serial-to-TCP gateway web page.

```
<br>Serial2TCP<br> &nbsp;&nbsp; <a href=" S2TCP.htm " target="_top">
S2TCP Client</a></font>
```

11. Build the `website.zdsproj` project to obtain the new library file `Acclaim_website.lib`.
12. Copy this `Acclaim_website.lib` to the `...\ZTP_2.1.0\ZTP\Lib` folder where it replaces the existing one.
13. Close the `website.zdsproj` project.
14. In ZDS II, open the `ZTPDemo_F91.zdsproj` file available in the following path:

    `...\ZTP_2.1.0\ZTP\SamplePrograms\ZTPDemo`

15. Add the `*.c` files located in the `...\ZTP_2.1.0\ZTP\SamplePrograms\ZTPDemo` folder to the `ZTPDemo_F91.zdsproj` project. To do so, select the Project menu and then select **Add Files**. The following `*.c` files need to be added to the project:
    - `Config_Flash_Server.c`
    - `Discovery.c`
    - `RemoteConfig.c`
    - `S2TCP_cgi.c`
16. Look for the **If** structure definition in the `ZTPConfig.c` file:

```
struct commonServers csTbl=
{
"172.16.6.28",        // Default Timer Server
"",                   // Default rfs server
"",                   // Default File Server - // Not currently Used
"172.16.6.194"        // Default Name Server
};


struct If ifTbl [MAX_NO_IF] = {// interface 0 -> Ethernet Configuration
{
&usrDevBlk [0],                      // Control block for this // device
ETH,                                 // interface type
ETH_MTU,                             // MTU
ETH_100,                             // Speed can be ETH_10 or AUTOSENCE
"172.16.6.198",                      // Default IP address
"172.16.6.1",                        // Default Gateway
0xffff0000UL                         // Default Subnet Mask
}
}
```

Note: *The `ifTbl` variable in the `ZTPConfig.c` file contains network parameters and settings (in the four-octet dotted decimal format) specific to the local area network at Zilog by default. Modify the structure definition in Step 16 with appropriate IP addresses within your local area network. For details about modifying the structure definition, refer to the ZTP v2.0.0 documents.*

17. Disable the DHCP (Dynamic Host Configuration Protocol) in the `ZTPConfig.c` file to assign the network parameters set in `ifTble[]` to the module.

    ```
    UINT8 b_use_dhcp = FALSE
    ```

18. Open the `emac_conf.c` file and change the default MAC address (provided by ZTP) such that each eZ80® development platform on the LAN contains a unique MAC address. For example:

    ```
    INT8 f91_mac_addr [ETHPKT_ALEN] = {0x00, 0x90, 0x23, 0x00, 0x01,
    0x01};
    ```

    In the 6 byte MAC address shown above, the first three bytes must not be modified; the last three bytes can be used to assign a unique MAC address to the eZ80Acclaim*Plus!* development platform.

19. Open the `main.c` file of the `ZTPDemo_F91.zdsproj` project and add the following function prototypes and global variables:

    ```
    extern void Flash_config_Discovery( void );
    extern UINT8  g_HTTP;
    extern UINT8  g_IGMP;
    extern UINT8  g_TELNET;
    extern UINT8  g_SNMPD;
    extern UINT8  g_TIMED;
    // Macros for threads.
    #define PRIORITY 20 // Thread priority.
    #define STACK_SIZE 1024 // Stack size for the thread.
    #define RR_TICK 5 // Round robin tick for the
    // scheduler.
    extern void SerialReadThread();
    // Global variables.
    // Thread handles to store.
    RZK_THREADHANDLE_t g_hthd1;
    // Stack for the thread.
    char g_thd1stack [ STACK_SIZE ];
    ```

20. Replace the complete `ZTPAppEntry()` function code with the following lines of code in the `main.c` file:

    ```
    {
    RZK_DEVICE_CB_t *TTYDevID;
          /** This function displays the IP Address and the related
                information of all the interfaces */
          nifDisplay(CONSOLE);
    /*this is for initializing the HTTP server without authentication. Use the
      Acclaim_Website.lib Website Library for this (Modify at Project->Set-
    tings->Linker->General->Object/library modules)*/
    ```

```
        http_init(http_defmethods,httpdefheaders,website,80);
        /** IGMP Initialization */
        hginit(0);


        /** Time Request */
        if( g_TIMED )
              time_rqest();


        /** Telnet Server Initialization **/
        if( g_TELNET )
              telnet_init();


        /** SNMP Initialization */
        if( g_SNMPD )
              snmp_init(snTrapNotifyCallback);
        /**************** Serial-to-TCP demo *****************/
        printf("\nSerial To TCP Ready" );
        printf("\n>" ) ;


        g_hthd1 = RZKCreateThread(
        ( RZK_NAME_t *) "Thread1",
        (RZK_PTR_t)SerialReadThread,
        NULL,
        (CADDR_t)( g_thd1stack + STACK_SIZE ),
        PRIORITY,
        RR_TICK,
        RZK_THREAD_PREEMPTION | RZK_THREAD_ROUNDROBIN, 0 ) ;
        // RZK_THREAD_AUTOSTART | RZK_THREAD_PREEMPTION |
        // RZK_THREAD_ROUNDROBIN ) ;
        // Validate the thread handle.
        if( g_hthd1 == NULL )
        {
        printf("\nUnable to create the thread #1, error description is");
        RZKFormatError(RZKGetErrorNum()) ;
        return -1;
        }
        RZKResumeThread(g_hthd1);


        Flash_config_Discovery();


  return 0;
  }
```

21. In the `ZTPInit_Conf.c` file, uncomment out the following line:

    ```
    Init_DataPersistence ();
    ```

22. In the `ZTPConfig.c` file, make `g_ShellLoginReqd = FALSE`.
23. Save the files, build and close the `ZTPDemo_F91.zdsproj` project.

Note: *The Network Scan client program is available in the* `AN0160-SC01.zip` *file. Extract it to get all the source files and the* `NetworkScan.exe file`.

Testing

This section discusses the basic setup, the equipment used, and the procedure followed to test the serial-to-TCP gateway application.

Setup

The basic setup for testing the serial-to-TCP gateway is described in the following sections. The serial-to-TCP gateway application emphasizes the On-Chip peripherals MAC and UART for the eZ80Acclaim*Plus!* Connectivity ASSP and the TCP/IP stack ZTP. Figure 5 displays the connections among a PC, LAN/WAN, and the eZ80Acclaim*Plus!* Evaluation Kit

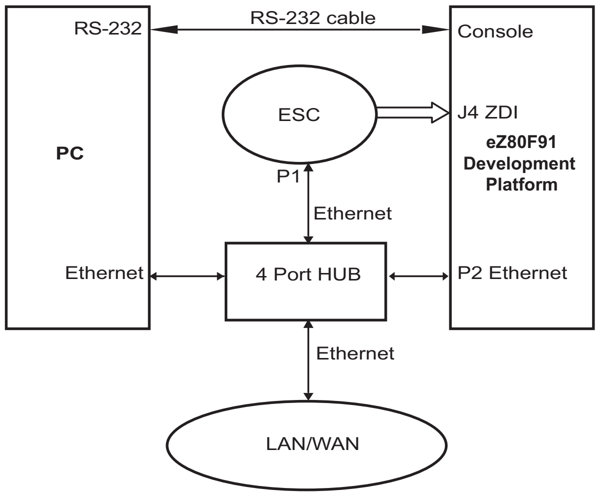


Figure 5. Test Setup for Serial-to-TCP Gateway

Equipment Used

The equipment used includes the following:

- eZ80F91 development kit
- eZ80Acclaim*Plus!*-based evaluation board

HyperTerminal Settings

Set the HyperTerminal to 57.6 kbps Baud, 8-N-1 protocols, and flow control none for COM port 1.

Procedure

Follow the steps below to test the serial-to-TCP gateway application:

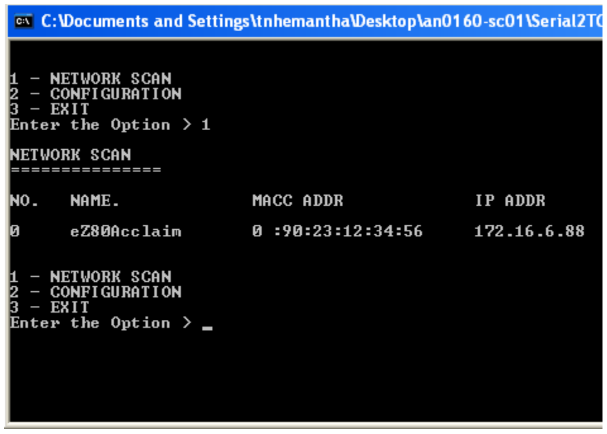
1. Connect the eZ80 development platform to the PC with a crossover Ethernet cable.

   This connection is made by means of the orange-colored cable. This Ethernet connection can also be made using a Hub or switch.

2. Program the `Serial2TCP.hex` file (located in the `\serial2TCPDemo` directory) into an eZ80F91 module that is attached to the eZ80 development platform. Push the **Reset** button on the platform. After reset, the eZ80 development platform is ready to work as a serial-to-TCP gateway. The programming is performed using ZDS II Flash Loader utility.

3. Connect the port marked *CONSOLE* with a straight-through cable to a PC port used by a terminal program such as HyperTerminal.

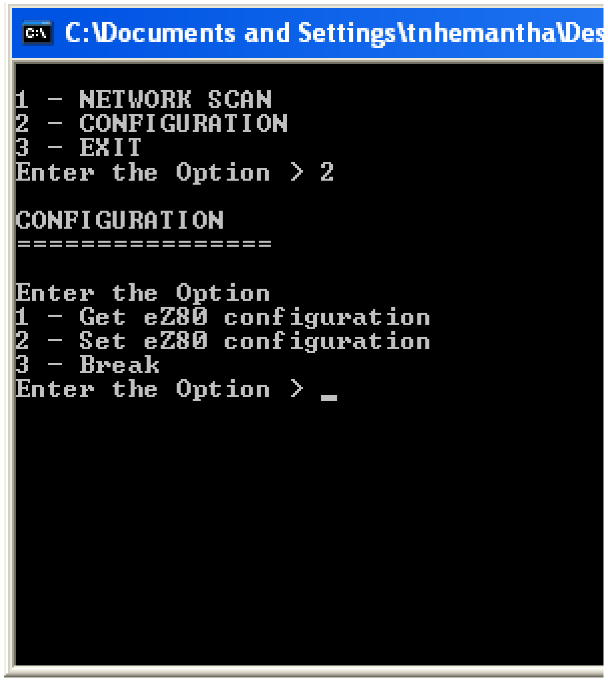
4. Start the Network Scan Client and choose option **1- NETWORK SCAN**, and then eZ80 responds with the 'gateway,' IP address, and MAC address.



```
C:\Documents and Settings\tnhemantha\Desktop\an0160-sc01\Serial2TC

1 - NETWORK SCAN
2 - CONFIGURATION
3 - EXIT
Enter the Option > 1

NETWORK SCAN
==============

NO.   NAME.              MACC ADDR             IP ADDR

0     eZ80Acclaim        0 :90:23:12:34:56     172.16.6.88

1 - NETWORK SCAN
2 - CONFIGURATION
3 - EXIT
Enter the Option > _
```

Figure 6. Network Scan

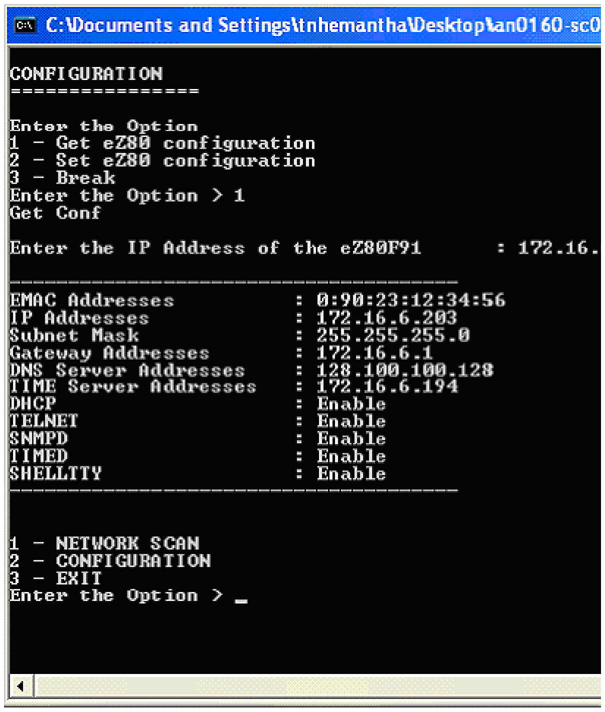
5. Now you can configure the gateway with the option **2 – CONFIGURATION** to set the IP address and chose the required protocols.



```
C:\Documents and Settings\tnhemantha\Des

1 - NETWORK SCAN
2 - CONFIGURATION
3 - EXIT
Enter the Option > 2

CONFIGURATION
================

Enter the Option
1 - Get eZ80 configuration
2 - Set eZ80 configuration
3 - Break
Enter the Option > _
```

Figure 7. Network Scan

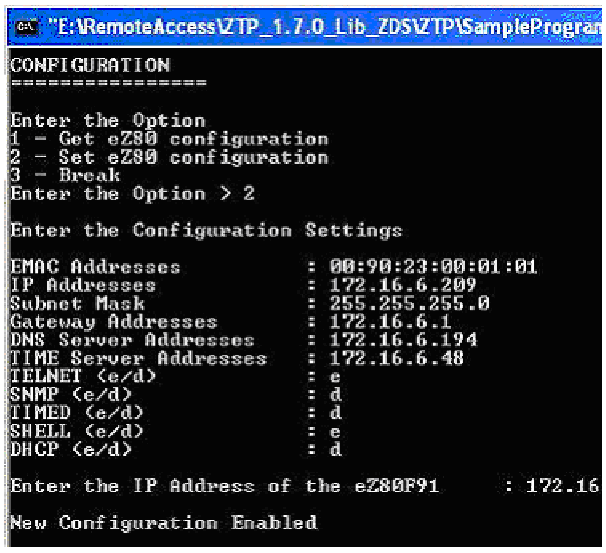
– Enter 1 to get the current configuration of eZ80.



```
C:\Documents and Settings\tnhemantha\Desktop\an0160-sc0

CONFIGURATION
================

Enter the Option
1 - Get eZ80 configuration
2 - Set eZ80 configuration
3 - Break
Enter the Option > 1
Get Conf

Enter the IP Address of the eZ80F91        : 172.16.

-----------------------------------------------------
EMAC Addresses               : 0:90:23:12:34:56
IP Addresses                 : 172.16.6.203
Subnet Mask                  : 255.255.255.0
Gateway Addresses            : 172.16.6.1
DNS Server Addresses         : 128.100.100.128
TIME Server Addresses        : 172.16.6.194
DHCP                         : Enable
TELNET                       : Enable
SNMPD                        : Enable
TIMED                        : Enable
SHELLTTY                     : Enable
-----------------------------------------------------

1 - NETWORK SCAN
2 - CONFIGURATION
3 - EXIT
Enter the Option > _
```

Figure 8. Current Configuration

– Enter 2 to set the eZ80 configuration.



```
"E:\RemoteAccess\ZTP_1.7.0_Lib_ZDS\ZTP\SampleProgran

CONFIGURATION
================

Enter the Option
1 - Get eZ80 configuration
2 - Set eZ80 configuration
3 - Break
Enter the Option > 2

Enter the Configuration Settings

EMAC Addresses               : 00:90:23:00:01:01
IP Addresses                 : 172.16.6.209
Subnet Mask                  : 255.255.255.0
Gateway Addresses            : 172.16.6.1
DNS Server Addresses         : 172.16.6.194
TIME Server Addresses        : 172.16.6.48
TELNET (e/d)                 : e
SNMP (e/d)                   : d
TIMED (e/d)                  : d
SHELL (e/d)                  : e
DHCP (e/d)                   : d

Enter the IP Address of the eZ80F91       : 172.16

New Configuration Enabled
```

Figure 9. Setting the Configuration

When the IP assigned to the gateway is known, you can use the IP address in Internet Explorer and enter data into the terminal program and observe the data arriving at the gateway using the TCP Connection.

Summary

This application note is provided so that you can use Ethernet-enable devices that communicate using a serial communication port. The implemented software, based on the ZTP for the eZ80 CPU, creates a bridge between the UART and Ethernet interfaces.

References

The documents associated with the eZ80®, eZ80Acclaim!®, and eZ80Acclaim*Plus!*™ family available at www.zilog.com are provided below:

- eZ80® Remote Access Application Note (AN0134)
- eZ80F91 Product Specification (PS0192)
- Zilog TCP/IP Software Suite Programmer's Guide Reference Manual (RM0008)
- Zilog Developer Studio II—eZ80Acclaim! User Manual (UM0144)
- eZ80Acclaim*Plus!*™ Connectivity ASSP eZ80F91 ASSP Product Specification (PS0270)
- eZ80Acclaim*Plus!*™ ASSP Product Brief (PB0220)
- eZ80F91 Mini Enet Module for eZ80Acclaim*Plus!*™ Product Specification (PS0271)
- Zdots SBC for eZ80Acclaim*Plus!*™ Connectivity ASSP Product Specification (PS0261)
- The eZ80F91 MCU as a Mail Server Application Note (AN0207)
- Interfacing an HTML Form to the eZ80F91 MCU (AN208)

Appendix A—Glossary

Table 2 lists the definitions for terms and abbreviations used in this application note.

Table 2. Glossary

Term/Abbreviation	Definition
Ethernet interface	A fast networking media interface that allows for computer communication using a LAN, WAN, or Internet; physical media type for TCP/IP networks.
Serial interface	An interface that allows for communication between computers using two wires, sending one bit of data at a time; a slow means of communication relative to Ethernet communication.
UART	Universal Asynchronous Receiver/Transmitter.
Client-server application	In general, all of the machines on the Internet can be categorized into two types: servers and clients. Those machines that provide services to other machines (such as Internet or FTP services) are servers. The machines that are used to connect to those services are clients. When you connect to Yahoo! at www.yahoo.com to read a page, Yahoo! is providing a machine (probably a cluster of very large machines) on the Internet to service your request. Yahoo! is providing a server. The typical PC most likely provides no services to the Internet. Therefore, the typical PC is most likely a client. It is possible for a machine to be both a server and a client. However, for purposes of this application note, you can think of most machines as one or the other.

Warning: DO NOT USE IN LIFE SUPPORT

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer