A Roadmap to Z-World's Rabbit-Based TCP/IP Sample Programs

This document contains a summary of some of the TCP/UDP/IP sample programs provided for the Rabbit. It can help you find a sample that might be a close match to an application you want to develop. Each sample program is briefly described, along with keywords for the topic of each program. Some run on all network boards (either Ethernet or PPP), and some are designed for specific boards.

Some of the application listed in this document call the TCP functions directly, and some use application libraries for protocols such as FTP and HTTP. The "generic" samples are not board-specific; the PPP examples require a serial port. There are samples for the uC/OS II multi-tasking environment ported by Z-World.

The sample programs are divided into several broad categories.

- Generic TCP Samples
- Generic UDP Samples
- Board-Specific TCP Samples
- Board-Specific UDP Samples
- µC/OS II Samples
- Point-to-Point Protocol (PPP) Samples
- Generic Internet/ICMP Samples
- Board-Specific Miscellaneous Internet/ICMP Samples

If you know a protocol, you can just search this text file. Each entry starts with the filename of the sample program, followed on the next line by keywords relevant to the sample. After that is a summary of the program, highlighting its functions and/or features.

Click here for a roadmap to board-specific sample programs.

1.1 Generic TCP Samples

This group of programs contain TCP samples that work on any Ethernet board. They can be modified to use PPP instead of Ethernet. These samples are generally the higher protocols, such as HTTP, telnet, and SMTP.

`Samples\Tcpip\active.c`

Keywords: TCP, http

Retrieves a web page from an HTTP server. All addresses are compiled in. Demonstrates using **`sock_established()`** and **`sock_bytesready()`** to make sure the socket is connected.

`Samples\Tcpip\echo.c`

Keywords: TCP, listen

This program demonstrates the **`tcp_listen()`** call. A basic server, that when a client connects, echoes back to them any data that they send. Data also displayed on Stdout. Can be configured to listen on different TCP ports.

`Samples\Tcpip\serialexa.c`

Keywords: TCP, telnet, serial

This program directs all of the data from a serial port to a TCP port and vice versa. Demonstrates buffering and timeouts. Can handle multiple connections through a state machine it uses.

`Samples\Tcpip\state.c`

Keywords: TCP, HTTP server

This program demonstrates building a state machine-based Internet server. The example is a simple web server (it doesn't use HTTP.LIB). Study the usage of **`sock_dataready()`** and **`sock_fastwrite()`** here when implementing your own server.

`Samples\Tcpip\tcp_time.c`

Keywords: TCP, UDP, time, RFC868

Demonstrate how to talk to one or more "Time Protocol" servers (see RFC868 at **`http://www.faqs.org/rfcs/rfc868.html`**). Code includes both client and server. Permits Rabbit to get a reliable wall-clock time from the network. Uses TCP/UDP to connect to a sequence of Time servers. The list of servers is specified in an initialized array. You might like to use this as a basis for adding a reliable wall-clock time. Note that UDP is used to query the servers, but the target allows both TCP and UDP queries to its server sockets. This sample also demonstrates the extended UDP functions and waiting for DNS name-to-IP resolving (some of the finer details or DCRTCP).

Samples\Rtclock\nist_time.c

Keywords: TCP daytime, RFC867, RFC868

Accesses the current time from a well-known "daytime server." Decodes the time and updates the Rabbit's RTC with the time.

Samples\Tcpip\Bsd\bsd.c

Keywords: resolve, TCP, connecting, name service, domain name

Opens a port and waits for connections. Demonstrates some of the informational routines in the TCP stack, including **set/gethostname()**, **set/getdomainname()**, **getsockname()**, and **getpeername()**. Also how to format for printing the IP address.

Samples\Tcpip\Ftp\ftp_client.c

Keywords: FTP client

Demonstrates the Rabbit FTP client library. Pass all your parameters to **ftp_client_setup()** open a connection and the library will download a file. If successful it displays the file, otherwise the error code.

Samples\Tcpip\Ftp\ftp_server.c

Keywords: FTP server, sspec

Demonstrates the Rabbit FTP server library. Two files are given to user "anonymous" and three files for user "foo" who can see the first two files, but not the other way around. One file is created from **xstring** text. Uses **sspec_addxmemfile()** to add compiled in files, and **sauth_adduser()** to create authorized users, joined together with **sspec_setuser()**.

Samples\Tcpip\Http\cgi.c

Keywords: HTTP, CGI

Demonstrates a CGI function with the HTTP server library. **HTTPSPEC_FUNCTION** means a function will be creating the whole page dynamically. This program outputs a simple counter, incremented each time the CGI page is accessed. The function **http_date_str()** is used in the header, and **cgi_sendstring()** sends the whole response to the browser. The counter is not referenced using SSI.

Samples\Tcpip\Http\flashlog.c

Keywords: HTTP, Forms, SSPEC, CGI, Flash memory

The program logs users that request a controller's page. The log can be viewed online. Uses CGI pages to make entries, clear and display the access log. A form to alter the date uses **sspec_addform()** to construct it dynamically. Form inputs are programmed with a maximum width and a specific valid numeric range for each field.

Samples\Tcpip\Http\form1.c

Keywords: HTTP, Forms, SSPEC

Demonstrates dynamically building adding pages to the HTTP server. These pages have forms with two dependent variables. Software validates this, and rejects incorrect values.

Samples\Tcpip\Http\getfile.c

Keywords: FTP server, HTTP server, SSPEC, SAUTH, file system

Using the combination of FTP and HTTP server, a user can upload to a specific file that is visible through the HTTP server. A writable FTP file is created, whose contents can be updated (available in DC 7.25 and later). (This sample does not support uploading and creating new files.). Uses file system to store uploaded file.

Samples\Tcpip\Http\post.c

Keywords: HTTP server, CGI, form post

Maintain two string fields posted from a form. Includes logic to decode the form URL sent back on submit.

Samples\Tcpip\Http\post2.c

Keywords: HTTP server, CGI, form post, SSPEC

Extends the **http\post.c** example to also control LEDs on the prototype board. Once the user registers themselves, a cookie is sent to the browser. Whenever the user changes an LED, a user-specific log entry is created. This tracks (records) the user's actions based on their cookie. Uses the **http_flashspec** (and related) routines to manually maintain the state.

Samples\Tcpip\Http\post2a.c

Keywords: HTTP server, CGI, form post, ZSERVER

This is an alternate version of **http\post2.c** that uses the **ZSERVER.LIB** functionality to build the server table. Comparison with **post2.c** shows how a program can be converted from using **http_flashspec** to **ZSERVER.LIB** functionality. This uses the **http\post.c** style form submission, and the **ssi.c** style dynamic pages to build a fully functional and audited controller. At the user's first access the page, they enter their name into a form, which is then stored in a HTTP cookie. This is used to later build an audit trail of what changes each user makes.

Samples\Tcpip\Http\refresh.c

Keywords: HTTP server, SSI variable, auto-refresh

Sets up Java Script to keep on refreshing a web page. The page has an SSI variable that increments on each refresh. Refresh done with:

```
<BODY onLoad=window.setTimeout("location.href='index.shtml'",1000)>
```

Samples\Tcpip\Http\ssi.c

Keywords: HTTP server, dynamic web page, SSI

Uses SSI (server-side includes) to display the state of four virtual lights. After the user hits a (web page) button, the form submits to update the LED states. This program simulates the LED's. See **Samples\BL2000\TcpIP\ssi.c** for a program to change real LED's.

Samples\Tcpip\Http\ssi2.c

Keywords: HTTP server, dynamic web page, SSI, peer name

The same lights and buttons from http\ssi.c, with an audit log of who made what changes, encoded as a hash of the remote user's IP address using getpeername(). The audit log is a circular queue.

Samples\Tcpip\Http\static.c

Keywords: HTTP server, static web page

A very basic web server example with static pages. This program completely initializes the library, outputting a basic static web page. Contains lots of comments explaining the defines and settings.

Samples\Tcpip\Http\static2.c

Keywords: HTTP server, static web page, xmemory file, FTP server

Basic web server example that serves its "index.html" page from xmemory. It can be uploaded with FTP.

Samples\Tcpip\Pop\parse_extra.c

Keywords: POP client, read E-mail

Reads and downloads E-mail from a POP3 server. Defines **POP_PARSE_EXTRA** to displays messages in a nice format, separating header fields from the message body.

Samples\Tcpip\Pop\pop.c

Keywords: POP client, read E-mail

Reads and downloads E-mail from a POP3 server. Displays messages.

Samples\Tcpip\Smtp\smtp.c

Keywords: send E-mail

Sends a E-mail using a root memory string.

Samples\Tcpip\Smtp\smtpxmem.c

Keywords: send E-mail

Sends a E-mail, taking the message body from xmemory.

Samples\Tcpip\Telnet\rxsample.c

Keywords: telnet server, serial

Starts up a listen on TCP port 23. When a connection is established, all writes from the remote host are sent to Stdout (displayed in the Dynamic C IDE). This sample doesn't provide a way to send data to the remote host.

Samples\Tcpip\Telnet\vserial.c

Keywords: telnet server, virtual serial driver

Uses **VSERIAL.LIB** to create a bidirectional stream between a telnet port and a serial port on the Rabbit. It extends the application, **TcpIp\telnet\rxsample.c**, from just output to a serial port to bidirectional communication. Connection will be re-established after it's closed.

1.2 Generic UDP Samples

This group of programs contain UDP samples that work on any Ethernet board.

Samples\Tcpip\Tftp\tftp.c

Keywords: TFTP server (Trivial FTP)

Implements a TFTP server that can send one file and receive into another area. (The file system is not used here.) Eliminates all TCP buffers (to reduce RAM footprint). Although the code stores the file in a fixed-sized block of root memory, commented code shows how to store it in xmemory.

Samples\Tcpip\Tftp\tftpclnt.c

Keywords: TFTP client (Trivial FTP)

Uses TFTP to download one file from the server, then upload the same file. The file is stored into xmemory. Requires a remote host to act as the TFTP server.

Samples\Tcpip\Udp\udp_cli.c

Keywords: UDP send

Uses **udp_send()** to transmit a “heart beat” datagram once a second. Can be sent to a specific host, or broadcast to the whole local network.

Samples\Tcpip\Udp\udp_srv.c

Keywords: UDP receive

Uses **udp_recv()** to collect heartbeat datagrams. Since it doesn’t use the extended UDP receive function, the first client to send a packet “sets” which host subsequent datagrams will be received from.

1.3 Board-Specific TCP Samples

These samples demonstrate networking in conjunction with specific hardware features, such as digital-analog converters. Some of the board-specific examples have been ported to run on different boards.

Samples\BL2000\Applet\dac-ctrl.c

Keywords: Java, applet, DAC

Provides a Java applet to control DAC's on BL2000. Includes Java source and byte-code files.

Samples\BL2000\TcpIP\smtp.c
Samples\BL2100\TcpIP\smtp.c

Keywords: SMTP, E-mail

Send a canned E-mail when a button is pressed. Uses **SMTP.LIB** library.

Samples\BL2000\TcpIP\ssi.c
Samples\BL2100\TcpIP\ssi.c

Keywords: dynamic web page, SSI

Uses SSI (server-side includes) to display the state of four lights. After the user hits a (web page) button, the form submits to update the LED states.

Samples\BL2000\TcpIP\telnet.c
Samples\BL2100\TcpIP\telnet.c

Keywords: serial, telnet client, TCP, digital output

Sets up TCP listen and waits for a connection. Reads a serial port and writes the socket (a telnet client). Has a state-machine to track the state of the socket. Toggles a digital output when connection is made, and clears it when the connection is closed.

Samples\dmtarget\pas_open.c

Keywords: telnet server, echo

Shows use of passive open (listen) by implementing a server which returns "Hello world," then echoing whatever is received. The **\Samples\dmunit\tcp.c** program must be running on the DeviceMate processor.

Samples\dmtarget\smtp.c

Keywords: SMTP, DeviceMate, TCP

Demonstrates the use of the **DM_SMTP.LIB** library for sending e-mail messages from a target processor with a DeviceMate. The program will send out a single test message through an attached DeviceMate that is providing the basic TCP/IP service.

Samples\ICOM\tcp_respond.c
Samples\ICOM\tcp_send.c

Keywords: TCP, LCD display, client, server

A program pair for sending a message back and forth. **tcp_send.c** sends it first and awaits a reply (client). **tcp_respond.c** does a "listen" for connections. The query is shown on the Intellicom display. The user presses a button that determines which reply to send back. The device running **tcp_send.c** then shows the response on its display. If you want to test either of these programs with a single Rabbit device, substitute programs are provided. In the "windows\" and "UNIX\" sub-directories are programs to emulate both sides of this conversion.

(There are similar programs in **Samples\LCD_Keypad\122x32_1x7\TCPIP**.)

Samples\MiniWeb\Console.c

Keywords: TCP, ZConsole, SMTP, E-mail

It might not look like a network application but it is! Uses the ZConsole on the serial port to manage pages offered by the web server. From ZConsole, an operator (or remote program) can also send E-mail to a predefined address.

Samples\RCM2100\EthCore2.c

Keywords: TCP states, telnet server

Listens for a connection. Once connected, all serial input goes out the socket, and all socket data is sent out the serial port. An LED indicates a connection has been made. If a button is pressed, the socket is closed.

Samples\RCM2100\EthCore1.c
Samples\RCM2200\EthCore1.c
Samples\RTDK\ssi3.c

Keywords: TCP, web server, SSI, digital outputs, ximport

Runs a web server to control the four LEDs on the RCM2100 protoboard. The web user "clicks" on a button graphic to toggle the LED on and off. One web page is the program's source code!

Samples\ZConsole\Fs2Console.c

Keywords: TCP, ZConsole, SMTP, E-mail, SSPEC, FS2

Uses many of the ZConsole routines to support connections from either TCP or serial. Clients see a command prompt from which they can display and set variables (which can be used by the web server), change various IP parameters, list named files, and send E-mail. This sample is small and relies heavily on library function.

1.4 Board-Specific UDP Samples

These samples demonstrate networking in conjunction with specific hardware features.

`Samples\dmtarget\udp_echo.c`

Keywords: UDP, echo

Opens a UDP connection to another computer and sends UDP datagrams to it. A UDP echo server must be running on some other machine at ports 7656 and 7657.

`Samples\dmtarget\tcp_time.c`

Keywords: UDP, NTP, time, RFC868, Internet

Demonstrate use of DeviceMate to talk to one or more “Time Protocol” servers (see RFC868 at **`http://www.faqs.org/rfcs/rfc868.html`**). You must run **`\Samples\dmunit\tcp.c`** on the DeviceMate for this sample to work (on the target processor). You might like to use this as a basis for adding a reliable wall-clock time to an appliance which only has a crystal oscillator and no other means for setting the actual date and time.

1.5 µC/OS II Samples

This group of programs are µC/OS-II samples. Be aware that a 2K byte stack is required for the single thread that makes calls to the network libraries.

Samples\Tcpip\Ucos\ucos2.c

Keywords: TCP, uCOS, telnet, serial

A sample data-gathering application that combines a task reading from the serial port with another task to report this data to the user over the network, with associated local messaging and processing. Using µC/OS tasks, creates a reader and a writer task. One side controls a serial port, and the other works with a TCP socket. Data is forwarded and read from the other side. A semaphore regulates access to shared data (i.e. the buffers between the two tasks). Currently, at most one task can call any socket functions, and that task's stack must be at least 2K in size.

Samples\RDTK\ucos.c

Keywords: TCP, uCOS web server, HTML form

Sets up a web server and create two µC/OS threads. One thread blinks LEDs and another runs the web server. The browser user can change the LED state by pressing a button, which actually submits a small form.

Samples\RDTK\ucos3.c

Keywords: TCP, uCOS, SMTP, E-mail, web server, HTML form

Sets up a web server and create three µC/OS threads. One thread blinks LEDs and another runs the web server. The browser user can submit an E-mail address and message body to the server. Once posted, a third thread will send the E-mail message to the specified client. Each socket must be used by only a single thread. One web page is the source code.

1.6 Point-to-Point Protocol (PPP) Samples

These applications demonstrate PPP (and PPP over Ethernet) features.

Samples\Ppp\pppoe_test.c

Keywords: Internet, PPP, PPPoE, PAP, DHCP, LCP, dynamic IP, ISP, ximport

Connects using Point-to-point over Ethernet protocol on a serial port. Rabbit will obtain its IP address from the server. Requires setting the PAP username and password.

Samples\Ppp\ppp_answer.c

Keywords: Internet, PPP, dynamic IP, LCP, PAP, AT commands

Waits for a phone call and explicitly answers it. then authenticates to the caller (i.e. the ISP calls the Rabbit, great for remote sensor stations). It obtains an IP address from the caller (great for remote sensor stations). While connected, the web server is available. Loops around to wait for another call.

Samples\Ppp\modem_test.c

Keywords: Internet, PPP, dynamic IP, SMTP, PAP

Dials an ISP and authenticates. Once connected, an E-mail is sent to the defined SMTP server. The “From,” “To” and “Body” portions are hard-coded (compiled in). Then it hangs up. This test is repeated three times.

Samples\Ppp\modem_server.c

Keywords: Internet, PPP, LCP, PAP, web server

Dials an ISP and authenticates to it. The Rabbit obtains an IP address. While connected, the web server is available.

Samples\Ppp\co_modem_test.c

Keywords: costatements, costate, Internet, PPP, SMTP

Similar to **modem_test.c** but uses co-statements in the main loop. Uses non-blocking PPP co-functions. All the while, another costatement blinks an LED.

1.7 Generic Internet/ICMP Samples

This groups of programs include examples of printing the Ethernet MAC address, starting up a DHCP client, resolving a domain name using the DNS library, and the ICMP ping feature. They don't use application-level protocols such as HTTP or FTP.

Samples\Tcpip\dhcp.c

Keywords: DHCP, TFTP

Goes through the paces of obtaining a DHCP lease, and then 90 seconds later releases it. After a short pause, it obtains another lease. If there was a download file (and download succeeded), then it uses TFTP to upload it as **/tftpboot/echo** to the server.

Samples\Tcpip\bsd\bsd.c

Keywords: getpeername(), sethostname(), setdomainname(), getsockname()

The Rabbit TCP stack does not use the UNIX sockets API. However, it does support some of the naming function calls. This program opens a TCP port and provides information about which hosts connect to it.

Samples\Tcpip\DNS\dns2.c

Keywords: DNS, resolve

Manually goes through of process of resolving a network name into an IP address. Uses **resolve_name_check()** for non-blocking operation. Uses **MY_DOMAIN** for the default domain (for not fully qualified domain names). This function looks up multiple host names simultaneously. It is organized around a state machine, with separate state information maintained for each request.

Samples\dmtarget\dns_look.c

Keywords: DNS, Internet

Uses the DeviceMate to lookup the IP address for a hostname (DNS).

Samples\Tcpip\display_mac.c

Keywords: MAC, Ethernet

Displays the board's Ethernet MAC (Media Access Connection) "station" address, which is unique to the board. Uses a packet driver function to return the MAC. Z-World uses a range of addresses; reseller's might want to change it to a value inside their own range. Not the same as the IP, which can be bound to different network hardware addresses. The program then loops calling **tcp_tick()**, so you can ping it from another computer.

Samples\Tcpip\ping.c

Keywords: ICMP, ping, resolve

Uses compiled in addresses for the host and the target computer to send an ICMP ping request to. Does this once a second using **_ping()** to handle the ICMP transmission.

Samples\Tcpip\icmp\pingme.c
Samples\Tcpip\icmp\pingyou.c

Keywords: ICMP, ping, resolve

The program **pingyou.c** will send out a series of ICMP ping requests. The program **pingme.c** just sets up the TCP stack and then calls **tcp_tick()**, which gives time to the TCP stack to handle the ping requests.

1.8 Board-Specific Miscellaneous Internet/ICMP Samples

Although these are board-specific, a similar application has been adapted for use on several different boards.

Samples\RCM2100\ping.c

Keywords: ICMP, ping, resolve

This sample is for the RCM2100. It sends out an ICMP ping and flashes one LED. When the ICMP response arrives, flashes the other LED.

Samples\RCM2200\pingled.c

Keywords: ICMP, ping,

This sample is for the RCM2200. It sends out an ICMP ping and flashes one LED. When the ICMP response arrives, flashes the other LED.

INDEX

B

board-specific TCP/IP
- CONSOLE.C 9
- DAC-CTRL.C 8
- ETHCORE1.C 9
- ETHCORE2.C 9
- FS2CONSOLE.C 9
- PAS_OPEN.C 8
- RESPOND.C 9
- SMTP.C 8
- SSI.C 8
- TELNET.C 8

board-specific UDP
- ECHO.C 10
- TCP_TIME.C 10

G

generic Internet/ICMP
- BSD.C 13
- DHCP.C 13
- DISPLAY_MAC.C 13
- DNS_LOOK.C 13
- DNS2.C 13
- PING.C 13
- PINGME.C 14
- PINGYOU.C 14

generic TCP/IP
- ACTIVE.C 2
- BSD.C 3
- CGI.C 3
- ECHO.C 2
- FLASHLOG.C 3
- FORM1.C 4
- FTP_CLIENT.C 3
- FTP_SERVER.C 3
- GETFILE.C 4
- NIST_TIME.C 3
- PARSE_EXTRA.C 5
- POP.C 5
- POST.C 4
- POST2.C 4
- POST2A.C 4
- REFRESH.C 4
- RXSAMPLE.C 6
- SERIALEXA.C 2
- SMTP.C 5
- SMTPXMEM.C 5
- SSI.C 5
- SSI2.C 5
- STATE.C 2
- STATIC.C 5
- STATIC2.C 5
- TCP_TIME.C 2
- VSERIAL.C 6

generic UDP
- SRV.C 7
- TFTP.C 7
- TFTPCLNT.C 7
- UDP_CLI.C 7

M

miscellaneous Internet/ICMP
- PING.C 15
- PINGLED.C 15

µ

µC/OS II
- UCOS.C 11
- UCOS2.C 11
- UCOS3.C 11

P

point-to-point (PPP) protocol
- CO_MODEM_TEST.C 12
- MODEM_SERVER.C 12
- MODEM_TEST.C 12
- PPP_ANSWER.C 12
- PPPOE_TEST.C 12