

Circuit Cellar / ZILOG Flash Nets Cash Design Contest
September 30, 2004

Entry #eZ2853

Voyeur:

A flexible, attractive medium to display dynamic text
content from the Internet.

Abstract

Overview

Several years ago, when broadband Internet connections were just becoming commonplace in the home, I discovered a search engine that captured my attention. MetaCrawler, a search engine which uses other search engines to provide its results, provides a section of their web site they call MetaSpy. MetaSpy lists the last 10 searches that people have entered on their site. The page reloads itself automatically after a period of time, displaying 10 new searches. This happens over, and over, and over again. “Ultrasonic liposuction”, “president bush resume”, “symptoms of depression”, “minesweeper”. Each search was more interesting than the previous one. Some were of a serious nature, and some were just plain silly. These MetaSpy results were a sort of “collective conscious” of the wired at any given moment. Each visit to the MetaSpy web site never ceased to amuse us. We felt that it deserved a different kind of attention other than a bunch of people at the office huddled around a computer screen.

The ZILOG eZ80Acclaim!, with its on-chip MAC hardware and many peripheral interfaces, seemed to provide the ideal solution to this problem. Our goal became to create an attractive, easy-to-configure, and easy-to-read Internet-connected display box that was capable of displaying alphanumeric data with a minimal amount of user interaction and setup complexity. We wanted something that could be hung on a wall, in an area of the home or office where people spent their time, in order to create discussion and provide amusement. We figured that, after seeing Voyeur, people would generate additional ideas of what different types of dynamic text data they would want displayed. With that in mind, it became an additional goal of ours to make Voyeur extensible such that the user would be able to customize the type of data they would like displayed. The photograph below shows the final product.



Figure 1: Voyeur unit mounted on wall

System Overview

The Voyeur hardware consists of an eZ80F91 Contest Kit Adapter Board, eZ80F91 Mini-Module, a 16x2 Futaba US162SD03CB Vacuum Fluorescent Display, and a push-button. All the components (with the exception of the power supply) are mounted within an IKEA 4" deep 5"x7" wooden box frame. The push-button is mounted on the back of the unit, and there are two ports on the bottom of the frame: one for the Ethernet connection and one for the 5VDC wall transformer (from the contest kit). Thus, in order to operate the Voyeur, the unit must be mounted close to an AC outlet and a 10/100 Ethernet connection which has access to the Internet (Voyeur works through most cable/DSL routers).

The Voyeur software consists of three components: the embedded software executing on the eZ80, the Voyeur Setup utility which executes on a PC, and the server software which will typically execute on any standard web hosting account. The server software is responsible for interpreting the data provided by MetaSpy and converting search phrases to a command language that the Voyeur unit can interpret. The embedded software is responsible for retrieving, via a TCP connection, the command sequence generated by the script on the web server and passing the text to the Futaba display in an attractive fashion. The embedded software is also responsible for loading the new command page specified by the URL in one of the commands. The Setup utility is needed to set non-volatile parameters in the Voyeur unit to the user's specific setup, such as the initial URL of the server that is being used and the network parameters for the network to which the Voyeur unit is connected.

Figure 2 below details the hardware and software elements that make up the entire Voyeur system.

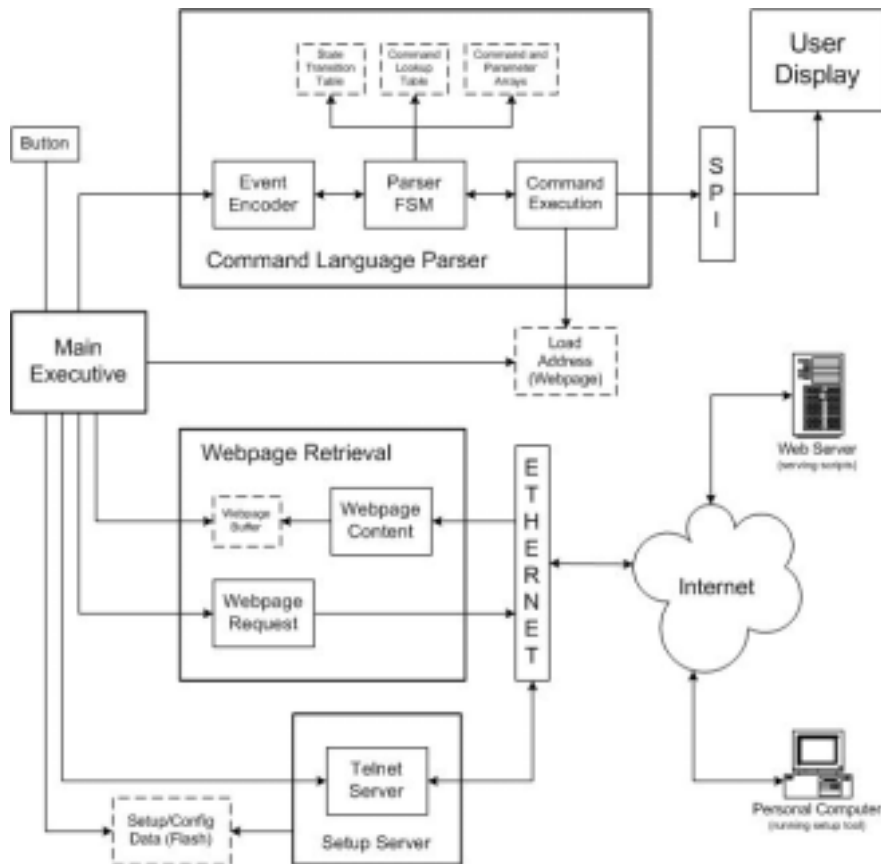


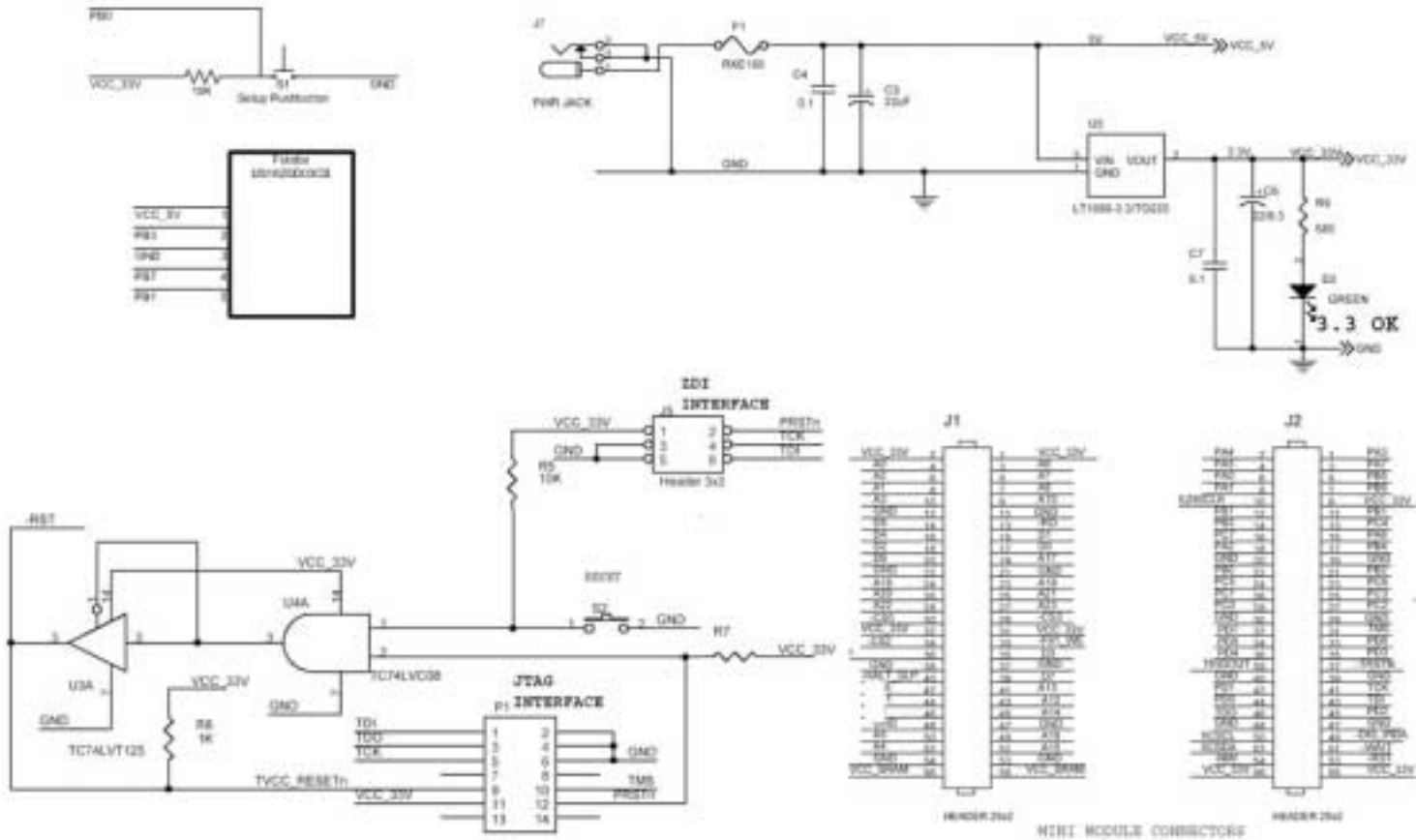
Figure 2: Voyeur system diagram

Schematics

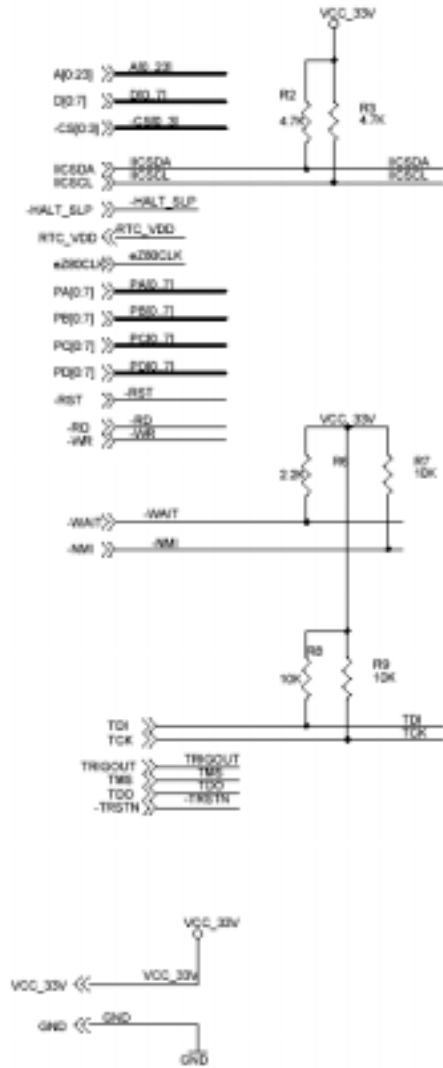
Voyeur Modified Adapter Board Schematic

Project #eZ2853

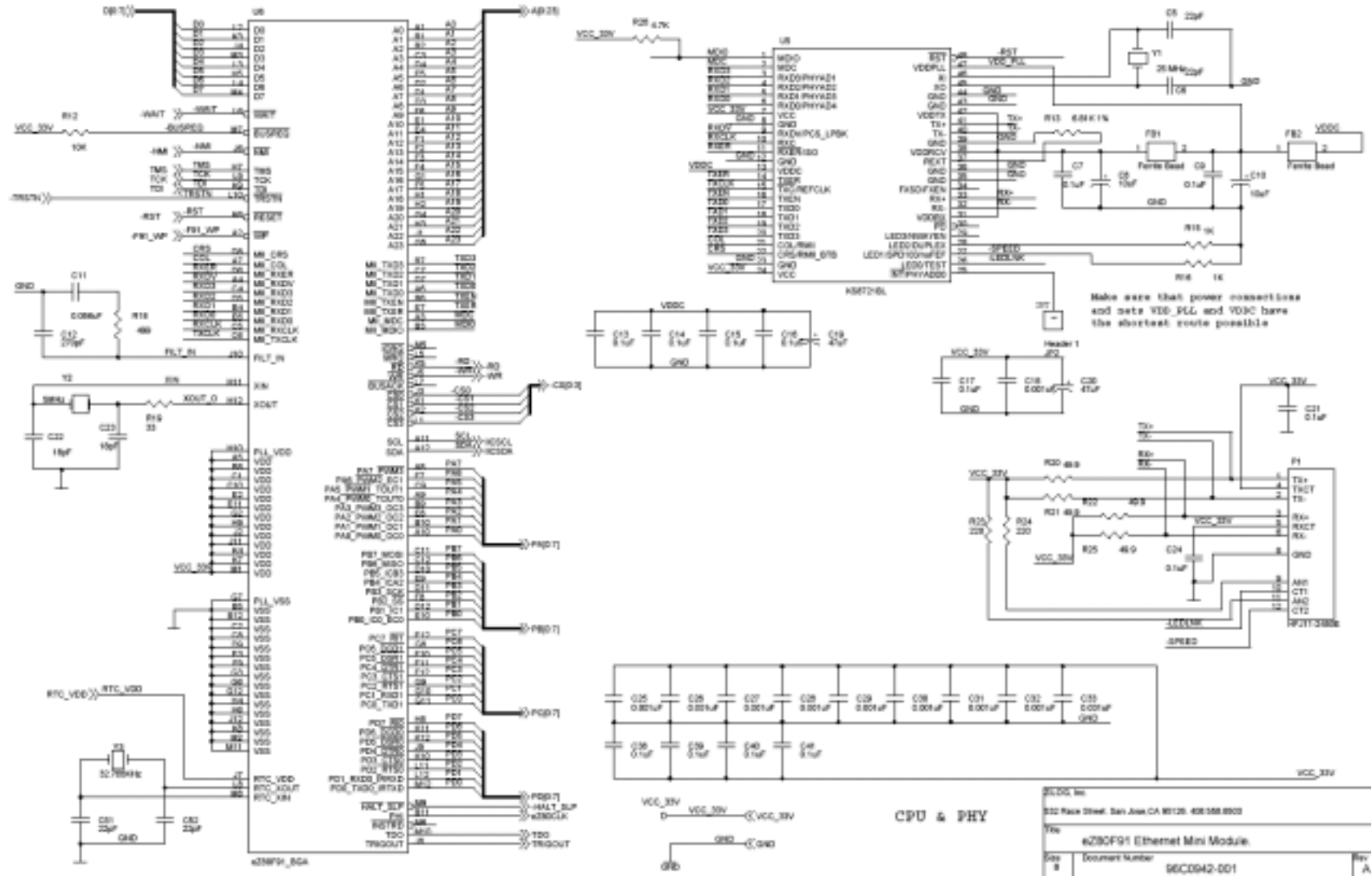
Part taken from ZILOG eZ2853 Adapter Pattern 98C2845-001 © ZILOG, Inc.



Mini-Module Schematic Sheet 1



Mini-Module Schematic Sheet 2



Code snippet: Main executive, main.c

```

/*
*****
*                               C M O D U L E F I L E
*
*****
*   FILE NAME:      main.c
*
*   ORIGINATOR:
*
*   DATE OF ORIGIN: 09/12/2004
*
*   FUNCTIONS:     main
*
*   DESCRIPTION:   Main executive for Voyeur
*
*   CAUTIONS:      None
*
*   MODIFICATION HISTORY:
*
*   Date      Name      Prob#  Description
*   -----
*   09/12/04
*
*****
*/

/*----- HEADER FILE INCLUDES -----*/
#include "micronet.h"
#include "FDisplay.h"
#include "lcd.h"
#include "WebPage.h"
#include "TelnetServer.h"
#include "Rtc.h"
#include "Timer.h"
#include "parser.h"
#include "misc.h"
#include "Setup.h"
#include <Stdio.h>

/*----- CONSTANTS -----*/
#define SECS_BEFORE_REBOOT 3500 /* Just under an hour */
#define PAGE_TEXT_LEN 1024

/*----- ENUMERATIONS -----*/
/*----- BASIC DATA TYPES -----*/
/*----- MACROS -----*/
#define reset_cpu() do { WDT_CTL = 0x83; while(1); } while (0)
#define pushbutton_is_pressed() ((PB_DR & 0x01) == 0x00)

/*----- FUNCTION PROTOTYPE(S) -----*/

/*----- GLOBAL DATA -----*/
char page_text[PAGE_TEXT_LEN];

/*
*****
*   FUNCTION NAME:      main
*
*   ORIGINATOR:
*
*   DATE OF ORIGIN:    09/12/04
*
*   INPUT PARAMETERS:
*
*   OUTPUT PARAMETERS:
*
*   VALUE RETURNED:
*
*   GLOBAL VARIABLES:  None
*
*   DESCRIPTION:
*
*   CAUTIONS:          None
*
*****
*/
int main(void)
{
    byte voyeur_configured_ok = FALSE;
    byte enter_setup_mode = FALSE;

    /* Initialize Futaba display over SPI */
    FDisplay_Init();

    /* Initialize Real Time Clock */
    Rtc_Init();

    /* Initialize command parser */
    parser_reset();

```

ZILOG Flash Nets Cash Entry eZ2853

```

/* Read setup data from flash */
Setup_Init();

/* Print welcome message and delay 5 seconds */
FDisplay_PutString("    voyeur    (c) 2004");
Timer_DelaySeconds(2);

/* Check button state to see if we need to go to setup mode */
if (pushbutton_is_pressed())
{
    enter_setup_mode = TRUE;
}

/* Initialize TCP/IP stack if it has not been initialized properly.
 * If no ethernet connection is present, stack will not init properly.
 */
/* Get IP addresses for tcp/ip stack from flash */
Setup_GetIpAddr(ip_src_addr);
Setup_GetGateway(gateway_ip_addr);
Setup_GetNetmask(subnet_mask);

/* Attempt to initialize stack, which also checks for ethernet connection */
if (mn_init() < 0)
{
    /* Print failure message */
    FDisplay_PutChar(CLEAR_CMD);
    FDisplay_PutString(" Check Internet    Connection    ");
    Timer_DelaySeconds(10);

    /* Reboot to trying again */
    reset_cpu();
}

/* Check if we need to enter setup mode */
if (enter_setup_mode == TRUE)
{
    int byte_count;
    byte j;
    char ipstr[18];
    byte setup_success = FALSE;

    FDisplay_PutChar(CLEAR_CMD);
    FDisplay_PutString("Hold button to  restore defaults");
    Timer_DelaySeconds(3);

    if (pushbutton_is_pressed())
    {
        Setup_RestoreDefaultSetup();
        FDisplay_PutChar(CLEAR_CMD);
        FDisplay_PutString(" Defaults Set,    Rebooting... ");
        Timer_DelaySeconds(5);
        reset_cpu();
    }

    /* Instruct user to telnet to setup IP address */
    FDisplay_PutChar(CLEAR_CMD);
    FDisplay_PutString("Current IP addr:");
    sprintf(ipstr, "%d.%d.%d.%d", ip_src_addr[0],
            ip_src_addr[1], ip_src_addr[2], ip_src_addr[3]);
    FDisplay_PutString(ipstr);

    /* Wait for user to send data */
    byte_count = TelnetServer_Get((byte *) page_text, PAGE_TEXT_LEN);

    if (byte_count > 0)
    {
        int i;

        /* Pass config data to parser. Results will be stored in globals:
         * g_loaddatavalid, g_idaddr, g_loadhost, and g_loadpath.
         */
        parser_feed(page_text, byte_count);

        /* Setup Complete */
        FDisplay_PutChar(CLEAR_CMD);
        FDisplay_PutString("Setup Complete,    Rebooting... ");

    } /* end if byte count > 0 */

    else
    {
        /* Something went wrong. Reboot. */
        FDisplay_PutChar(CLEAR_CMD);
        FDisplay_PutString(" Setup Failed,    Rebooting... ");
    }

    /* Wait to display message to user */
    Timer_DelaySeconds(5);

    /* Reset CPU */
    reset_cpu();
} /* end if (enter_setup_mode == TRUE) */

else

```

ZILOG Flash Nets Cash Entry eZ2853

```

{
    /* We are not in setup mode, tell user we are loading a page */
    FDisplay_PutChar(CLEAR_CMD);
    FDisplay_PutString("    voyeur    loading...  ");
}

/* Loop forever until we manually reboot (evaluation version of MicroNet
 * TCP/IP stack requires a reboot every hour or 5000 packets, whichever
 * comes first.)
 */
while (1)
{
    /* Check if we need to configure Voyeur IP addresses (e.g., if config
     * page has been loaded and parsed successfully.)
     */
    if (voyeur_configured_ok == FALSE)
    {
        byte config_addr[IP_ADDR_LEN];
        char config_hostname[MAX_HNAME_LEN];
        char config_request[MAX_REQ_LEN];

        /* Attempt Configure step */
        int byte_count;

        /* Retrieve Config from Setup data in flash */
        Setup_GetWebpageConfig(config_addr, config_hostname, config_request);

        /* Perform initial config and request */
        byte_count = WebPage_Get(config_addr, config_hostname, config_request,
                                (byte *) page_text, PAGE_TEXT_LEN);
        byte_count = WebPage_StripHeader(page_text, byte_count);

        if (byte_count > 0)
        {
            int i;

            /* Pass config data to parser. Results will be stored in globals:
             * g_loaddatavalid, g_idaddr, g_loadhost, and g_loadpath.
             */
            parser_feed(page_text, byte_count);

            /* If parser received a "load" command, then config happened
             * properly
             */
            if (g_loaddatavalid == TRUE)
            {
                voyeur_configured_ok = TRUE;
            }
        }

        /* end if byte count > 0 */

        if (voyeur_configured_ok == FALSE)
        {
            /* Print configuration failure message */
            FDisplay_PutChar(CLEAR_CMD);
            FDisplay_PutString("    Check Voyeur    Setup & Config ");

            /* Wait some time before trying again */
            Timer_DelaySeconds(10);
        }
    }

    /* end if (tcp_ip_configured == TRUE) && (voyeur_configured == FALSE) */

    /* Check if everything has been configured. Only then may we continue
     * loading and processing command pages.
     */
    if (voyeur_configured_ok == TRUE)
    {
        int i;
        int byte_count;

        /* Get page and strip HTTP header */
        byte_count = WebPage_Get(g_ipaddr, g_loadhost, g_loadpath,
                                (byte *) page_text, PAGE_TEXT_LEN);
        byte_count = WebPage_StripHeader(page_text, byte_count);

        /* Pass page data to parser. Results will be stored in globals:
         * g_loaddatavalid, g_ipaddr, g_loadhost, and g_loadpath.
         */
        parser_feed(page_text, byte_count);

        /* Check result of parsing to see if it received a valid load
         * command.
         */
        if (g_loaddatavalid == FALSE)
        {
            /* If there was no load command, something went wrong.
             * Force and attempt to re-configure.
             */
            voyeur_configured_ok = FALSE;
        }
    }

    /* end if (voyeur_configured == TRUE) */
}

```

ZILOG Flash Nets Cash Entry eZ2853

```
/* Reset CPU if we have been running for a certain duration. This is an
 * evaluation version of MicroNet TCP/IP stack which requires a reboot
 * every hour or 5000 packets, whichever comes first.)
 */
if (Rtc_GetElapsedSeconds() > SECS_BEFORE_REBOOT)
{
    reset_cpu();
}
} /* end while(1) */
return (0);
}
```