User Interface Design using CGI Programming and Boa Web Server on M5249C3 Board

by: H.K. Au
MCD Applications

1 Introduction

This application note describes the implementation of a user interface on the M5249C3 board. The user interface is developed using CGI programming with Boa server running on uClinux at M5249C3 board side as shown below:

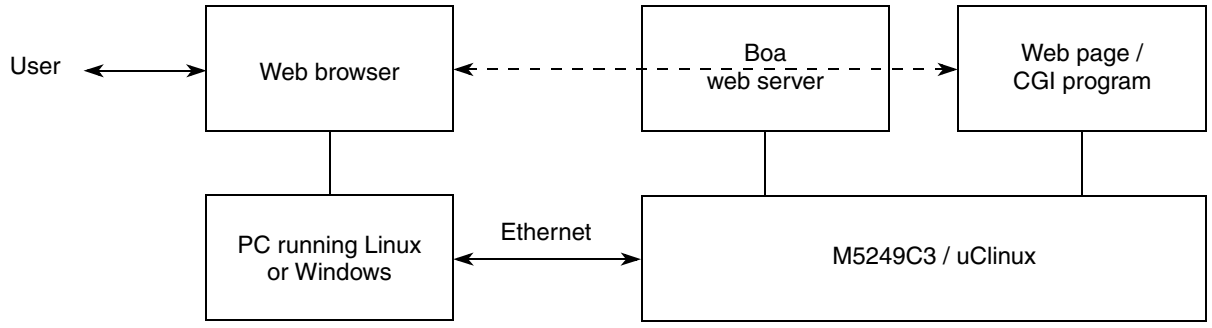

Figure 1. System Setup

The user can get the web page or invoke an executable program stored at the M5249C3 board side through the web browser and the Boa web server. A simple example is used in this document to illustrate how to implement this kind of user interface.

This example allows the user to input three parameters at the parameter setting web page, and then display them at the result display page. This application note assumes a familiarity of running uClinux on the ColdFire EVB. Therefore, the procedure of compiling uClinux and download the uClinux image to the

EVB are not included. The EVB used in this application note is the M5249C3. However, the procedure described in this application note can be applied to other ColdFire EVBs without a big difference.

2 Boa Setup

Boa web server can be setup in uClinux by enabling the boa in '`make xconfig`' ('uClinux Application Configuration => Network Application').

The default setting of Boa is:

The directory index file is 'index.html' in '/home/httpd/'

The CGI binary files are in '/home/httpd/cgi-bin/'

These settings can be changed by modifying the file 'home//httpd/boa.conf'. No need to modify boa.conf if the default setting is used. The example in the document used the default setting.

In uClinux source code, the index.html file is located in 'vendor/Generic/httpd'. After compilation, the index.html will be copied to 'home/httpd' in romfs.

So, the first step is to create the index.html file as shown at Appendix A and then copy the index.html to 'vendor/Generic/httpd and replace the original index.html from the uClinux distribution.

When the user accesses the Boa web server using 'http:// <IP address of the M5249C3 board>', the index.html will be shown at the user's web browser as below:

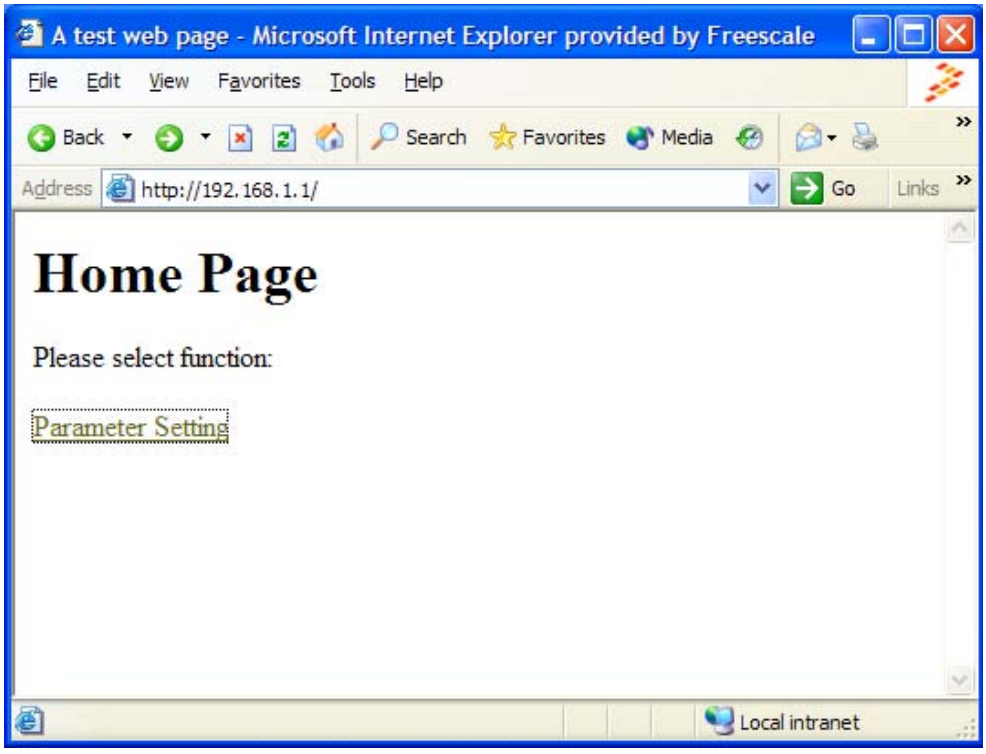

Figure 2. Home Page

3 CGI Programming Example

The common gateway interface (CGI) is a protocol that transmits data between the user web browser and the web server. In the example used in this document, the data are the parameters that user input to the M5249C3 board running uClinux. The CGI programs used in the example are all written in C language.

There are two CGI programs used in this example and they are modified from the demo programs in the uClinux distribution. The first is the ‘cgi_demo’ in /user/cgi_generic/. This program was modified to get the parameters from the user.

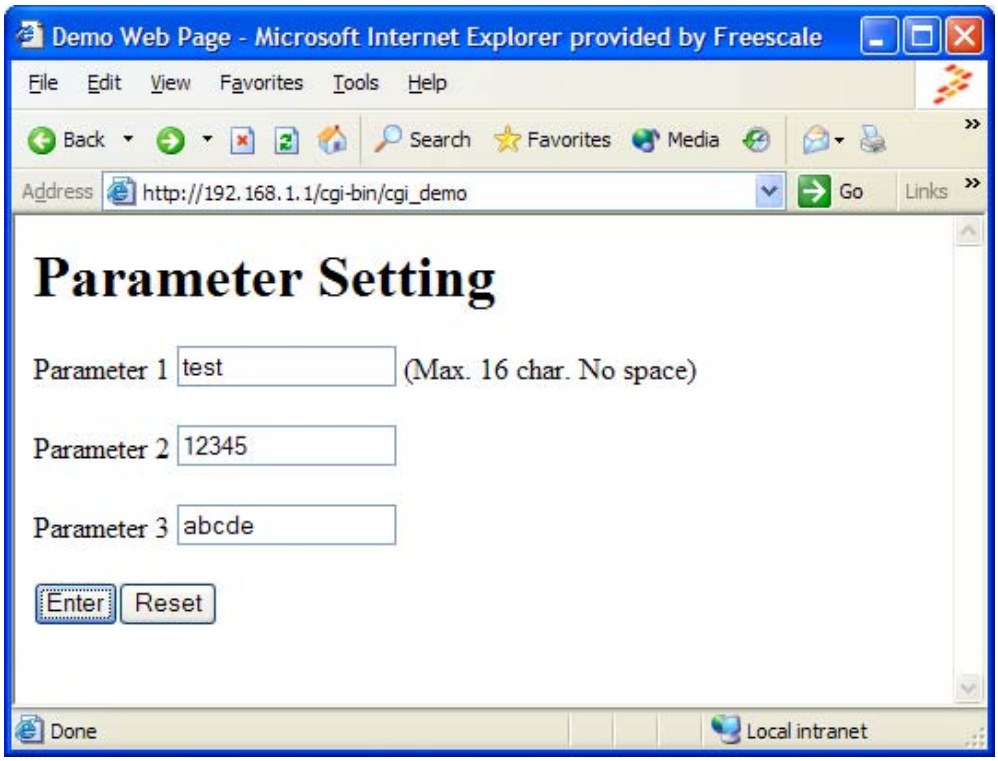

Figure 3. Parameter Setting Page

When the ‘Enter’ button is clicked, all three parameters shown will be shown at the resulting web page. When the ‘Reset’ button is clicked, the parameters will be reset to null and any modification done on the browser will be lost. For each input parameter in the example, the maximum length is 16 characters and no space character is allowed.

The second CGI program is the ‘query-results’ file in /user/cgihtml/examples/. This program was modified to get the parameters input by user then display the parameters on the web browser as shown:

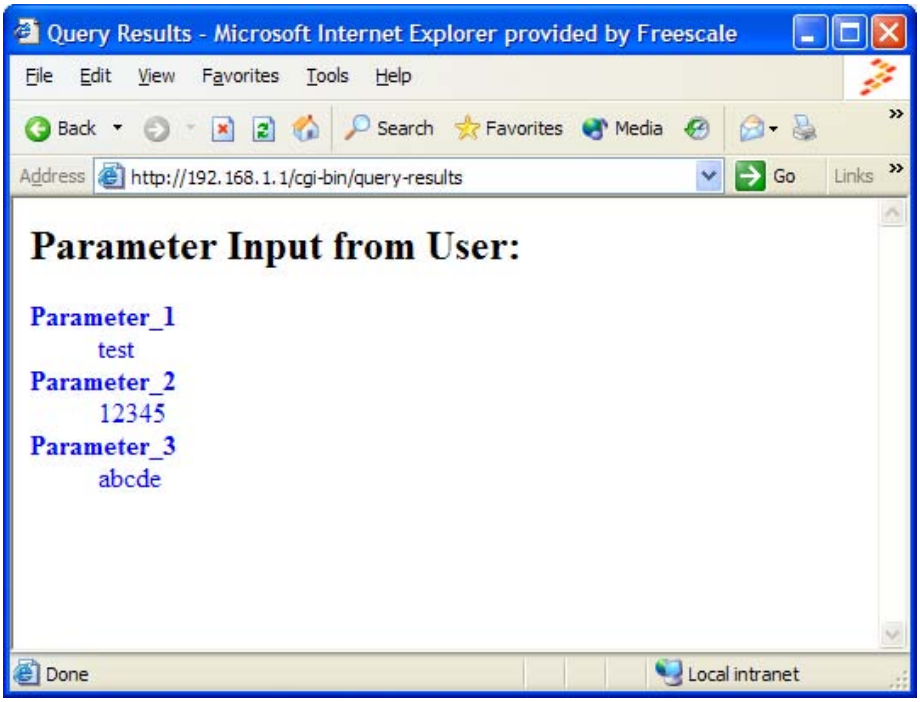

Figure 4. Parameter Display Page

4 Design of the CGI Program

The design of the example shown in previous section will be described in this section.

4.1 Home Page

The home page is the ‘index.html’ file in directory ‘home/httpd/index.html’. In the example, there is a link to parameter setting page in the home page. This is done by the following line of code in index.html:

<A HREF=/cgi-bin/cgi_demo>Parameter Setting</A>

When the ‘Parameter Setting’ icon is clicked on the home page, it will execute the ‘cgi_demo’ file in /cgi-bin/.

4.2 Parameter Setting Page

The ‘Parameter Setting Page’ is created by the ‘cgi_demo’ file which is a CGI program in C language. This C program is modified from the demo program in uClinux (/user/cgi_generic/). If this demo program is not used in real software development, a new application under uClinux can be created.

The subroutine template_page() is responsible for displaying the parameter setting page. The source code is in template.c. The template_page() subroutine uses the HTML form attribute to display the user input parameters and allow the parameter modification. In the ‘form’ action attribute, it specifies the URL of ‘/cgi-bin/query-results’.

The template_page() subroutine is invoked by main() in /user/cgi_generic/cgi.c as shown in Appendix A.

4.3 Parameter Display Page

The ‘query-results’ file is a CGI program which displays the parameter input at ‘Parameter Display Page’. The ‘query-results’ is modified from the example of cgihtml library. cgihtml is a collection of CGI parsing and HTML output functions written in C. These routines simplify the task of writing CGI programs in C.

The source code of ‘query-results’ is in ‘/user/cgihtml/examples/query-results.c’. ‘query-results’ uses the subroutines read_cgi_input() and print_entries() in cgihtml library for getting and display parameter respectively.

There are a lot of functions in the cgihtml library for writing a CGI program in C. Please refer to the documents in ‘/user/cgihtml/docs/’ for more information.

After displaying the input parameters, the program terminates for this example. However, if further processing on the input parameter is required, the inputs are available to the C program for processing. For example, use the function cgi_val() as shown in the query-result.c source code in Appendix D. In summary, the design of the CGI program example in this documents looks like this:

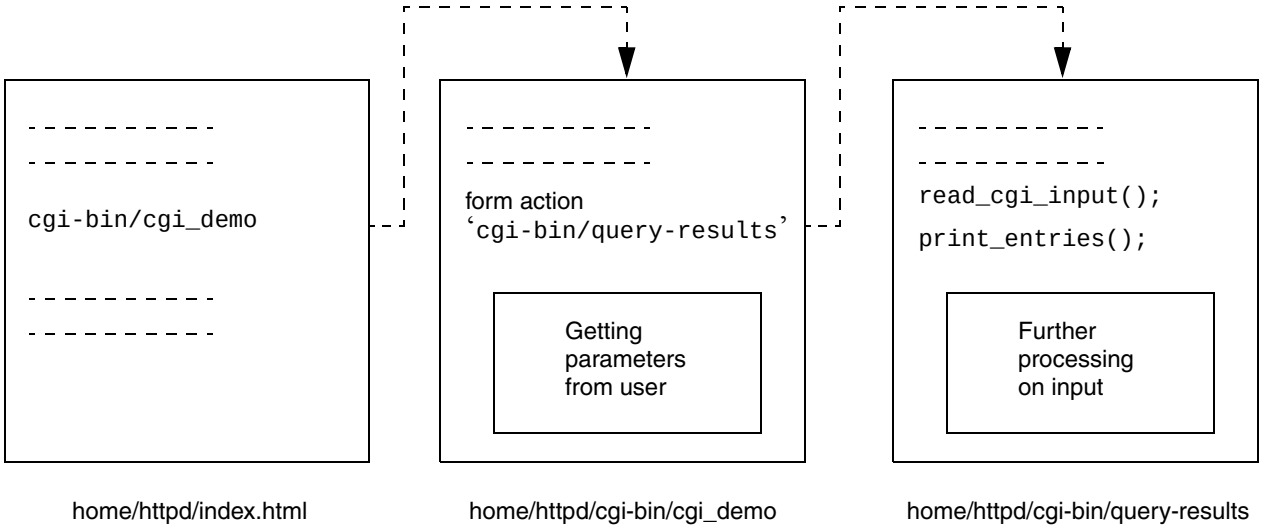

Figure 5. CGI Program Flow

5 Summary of Procedure of Preparing the CGI Example Source Code and Testing

Procedure:

1. Prepare index.html as shown in Appendix A. Copy to uClinux source code ‘vendor/Generic/httpd/’ and replace the original index.html.
2. Prepare cgi.c as shown in Appendix B. Copy to uClinux source code ‘/user/cgi_generic/’
3. Prepare template.c file as shown in Appendix C. Copy to uClinux source code ‘/user/cgi_generic/’

4. Prepare query-results.c file as shown in Appendix D. Copy to uClinux source code ‘user/cgihtml/examples/’
5. Enable boa in make xconfig (in Network Application)
6. Enable ‘cgi generic’ and ‘cgihtml’ in make xconfig (in uClinux Application Configuration =>Miscellaneous Configuration).
7. Execute `make dep` and `make` to generate the uClinux image.bin
8. Download the image.bin file to the M5249C3 board and run uClinux (g 20000)
9. Configure the IP address of uClinux and the host computer of the web browser on the same IP address domain. Test the CGI demo as described in Section 2, “Boa Setup,” and Section 3, “CGI Programming Example.”

6 Conclusion

An example of how to design a user interface for a M5249C3 board/uClinux embedded system using CGI/Boa is shown in this application note. Through good web page design using HTML, it is possible to develop a simple, user-friendly, and interactive method of getting input from the user. Software development effort can also be reduced if the existing CGI library in the uClinux distribution is used, as shown in this application note.

Appendix A
index.html

```
<HTML>
<HEAD>
<TITLE>A test web page</TITLE>
</HEAD>
<BODY>
<H1>Home Page</H1>
<P>
Please select function:
<P>
<A HREF=/cgi-bin/cgi_demo>Parameter Setting</A>
</BODY>
</HTML>
```

Appendix B
cgi.c

```
/* cgi.c */
#include <stdio.h>
#include <string.h>
#include "cgivars.h"
#include "htmllib.h"
#include "template.h"
int main() {
        char **postvars = NULL; /* POST request data repository */
        char **getvars = NULL; /* GET request data repository */
        int form_method; /* POST = 1, GET = 0 */
        form_method = GET;
        htmlHeader("Demo Web Page");
        htmlBody();
        template_page(postvars, form_method);
        htmlFooter();
        cleanUp(form_method, getvars, postvars);
                fflush(stdout);
        exit(0);
}
```

Appendix C
template.c

```
/* template.c */

#include <stdio.h>

#include "cgivars.h"

#include "htmllib.h"

int template_page(char **postvars, int form_method) {

        int i;

        char *a=NULL;

        addTitleElement("Parameter Setting");


        /* GET */

        printf("<FORM ACTION=\"%s\" METHOD=POST>", "/cgi-bin/query-results");


        printf("<P>Parameter 1 ");

        printf("<INPUT NAME=\"Parameter_1\" VALUE=%s SIZE=\"16\"
MAXLENGTH=\"16\">(Max. 16 char. No space) </BR>",a);


        printf("<P>Parameter 2 ");

        printf("<INPUT NAME=\"Parameter_2\" VALUE=%s SIZE=\"16\"
MAXLENGTH=\"16\"></BR>", a);


        printf("<P>Parameter 3 ");

        printf("<INPUT NAME=\"Parameter_3\" VALUE=%s SIZE=\"16\"
MAXLENGTH=\"16\"></BR>", a);


        printf("<BR><INPUT TYPE=submit VALUE=\"Enter\">");

        printf("<INPUT TYPE=reset VALUE=\"Reset\">");

        printf("</FORM>");

        return 0;

}
```

Appendix D
query-results.c

```
/* query-results.c */
#include <stdio.h>
#include "cgi-lib.h"
#include "html-lib.h"
int main() {
        llist entries;
        int status;
        char *p1="Parameter_1";
        char *p2="Parameter_2";
        char *p3="Parameter_3";
        char *first_input;
        char *second_input;
        char *third_input;
        html_header();
        html_begin("Query Results");
        status = read_cgi_input(&entries);
        h1("<small><font color=red>Parameter Input from User:</small><font
color=blue>");
        print_entries(entries);
        html_end();
        /*************************************************************
        Futher processing on the input parameters can be done here.
        *************************************************************/
        first_input  = cgi_val(entries, p1); // Example of getting the input to
        second_input = cgi_val(entries, p2); // the C program for processing
        third_input  = cgi_val(entries, p3); // using function cgi_val()
        list_clear(&entries);
        return 0;
}
```

This page intentionally left blank.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

AN3238
Rev. 0, 02/2006

freescale™ semiconductor