

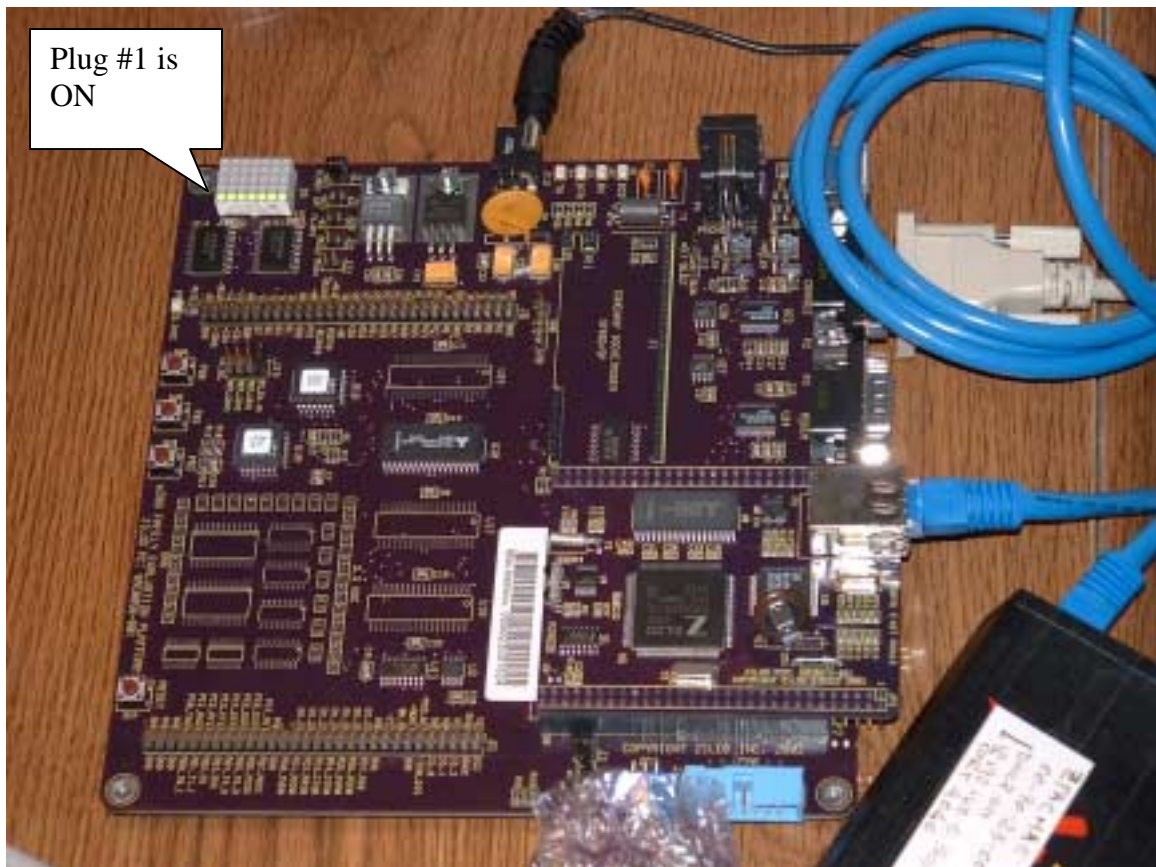
# **eZ2981 – Rhttpd Web Server**

---

**Description:** Small http server made for embedded applications - low memory footprint and low CPU resources.

**Features:** Basic Authentication for content protection, implementation for eZ80( ZDSII 4.8.0 and ZTP 1.4.0Beta2 TCP/IP stack), Windows (Cygwin/GCC) and Linux(GCC).

**Hardware:** eZ80F910200ZCO eZ80F91 Development kit. This project is software only. The EVB\_F91 development kit is used to demonstrate the functionality of the software. The LED matrix from the development board will show the three “plugs” on or off – a column of the matrix will be on for each plug. In the image one can see “Plug #1” on.

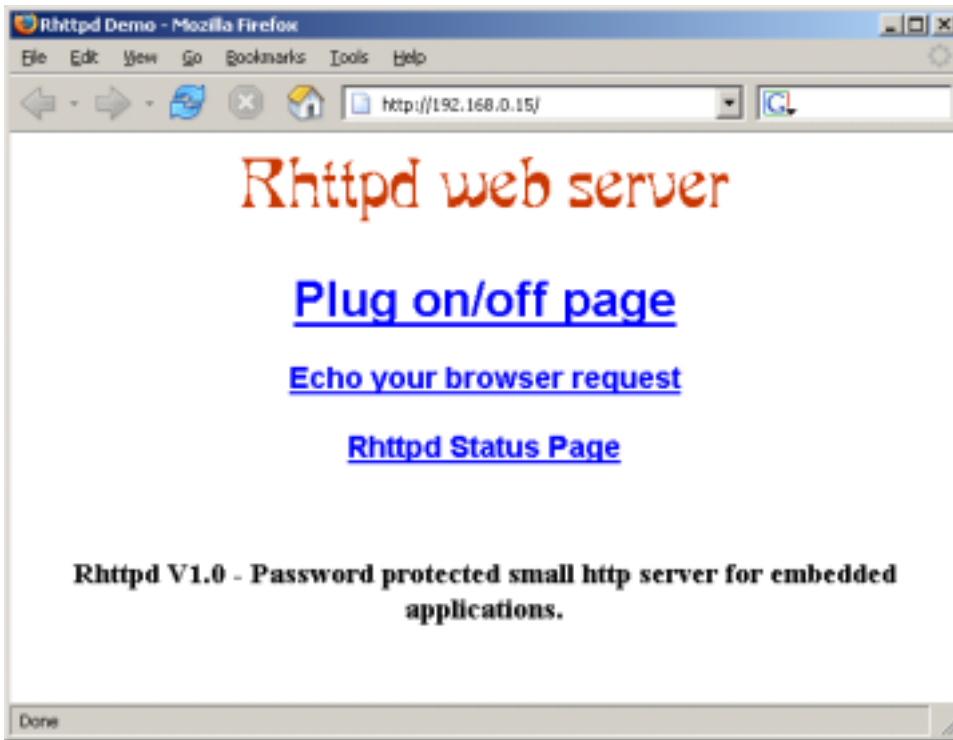


## **Application Example:**

1. Authentication screen – Please use User name: test, Password: test for this application example

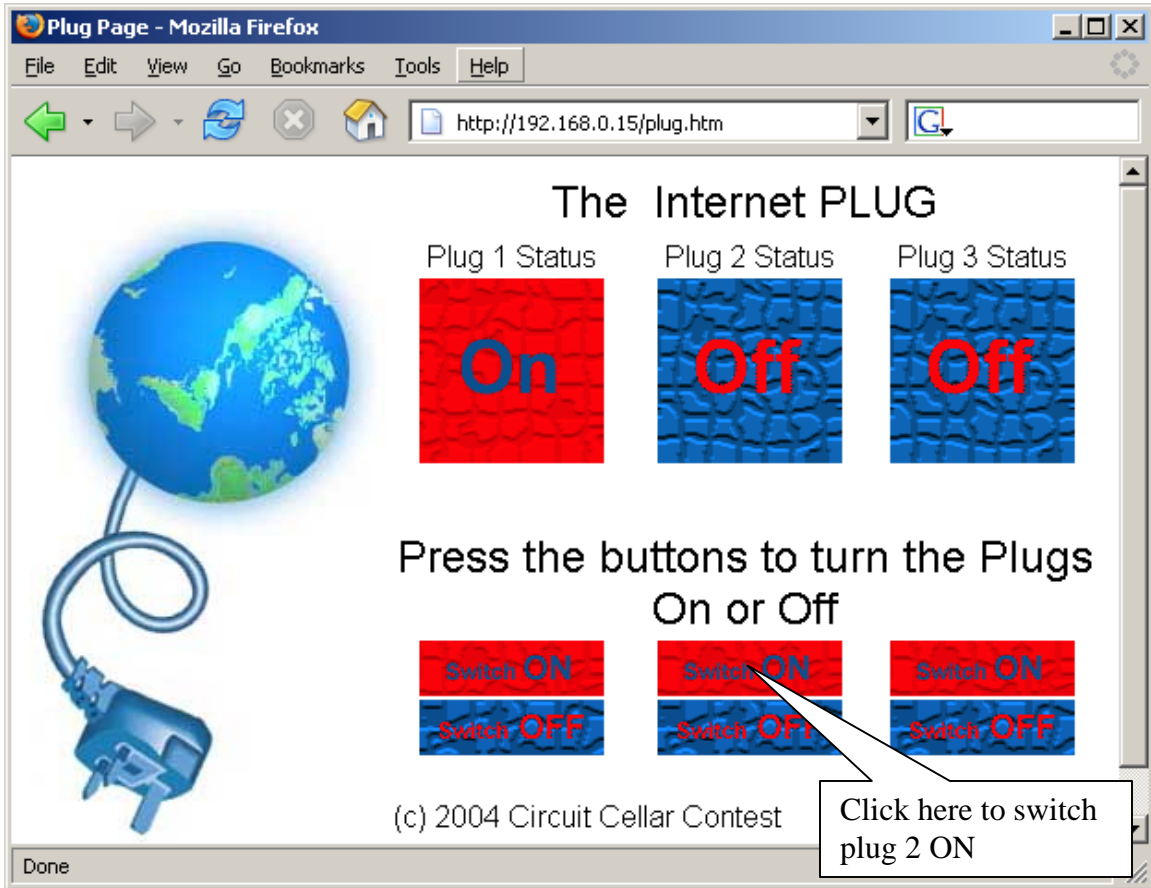


## 2. Main Page of the Application Example

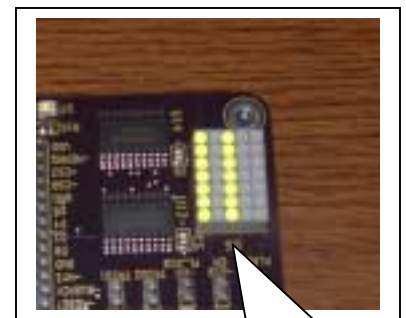
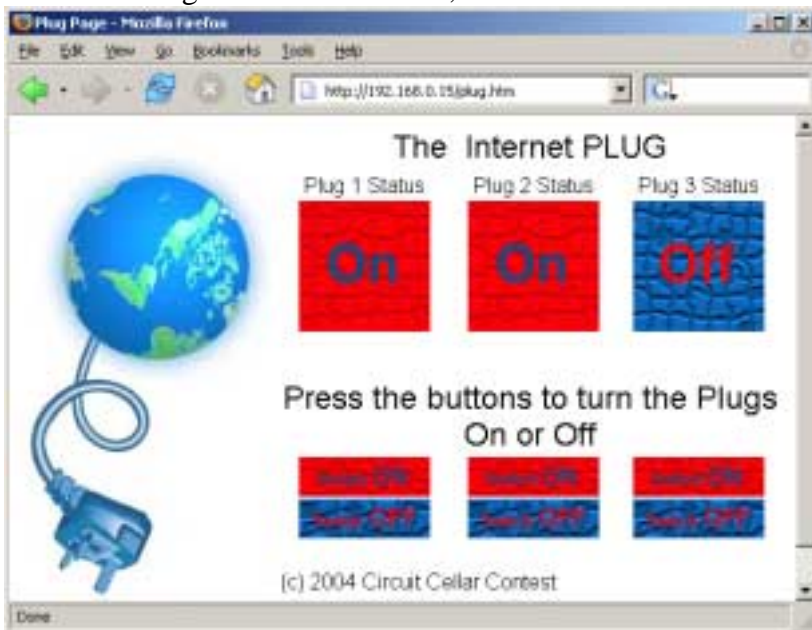


3. The Plug page – this page will show the three “Plugs” that can be switched on or off using the mouse. The page will change according to the user selections. Also the LED-s

from the LED matrix of the demo board will go on and off according to the user selections.



Now both Plugs 1 and 2 will be on, also columns 1 and 2 will be on the LED matrix:



## Anatomy of a Web Server

A Web Server is a program that listens for requests and answer them. Each request from a browser contains the name of the requested page and a number of other headers.

There are two main implementations for web servers – the one that generates a thread or a process for each page request and the “select model” HTTP server that will keep just a context record for each request and use non-blocking I/O and the select() system call to handle multiple requests in a single process, single thread.

Generating a new thread for each request is a bad idea on small CPU like the eZ80.

Another choice for really small servers is to have just one thread and keep the rest of the connections “on hold” until the current page is served – and this is the case for Rhttpd.

One connection will be accepted at a time and the request will be completed then the connection is closed. Meantime other connections can be in SIN/ACK state on hold.

### Resources:

Zilog documentation and application notes – [www.Zilog.com](http://www.Zilog.com)

thttpd - tiny/turbo/throttling HTTP server - <http://www.acme.com/software/thttpd/>

The uIP TCP/IP Stack for Embedded Microcontrollers - <http://www.sics.se/~adam/uip/>

### Source Code of the Main Server (server\_c.c)[FRAGMENT]

```
    if (fpath_length==0)
    {
        // this is not a GET request...so just respond with file
not found
        // file not found page 404
        //local_time=localtime(&calendar_time);

        printf("File not found 404 %s\r\n", fpath);

        nw = write_s(newsockfd,err4041,strlen(err4041),0);
        nw = write_s(newsockfd,fpath,fpath_length,0);
        nw = write_s(newsockfd," ",2,0);
        //nw =
write_s(newsockfd,asctime(local_time),strlen(asctime(local_time)));
        nw = write_s(newsockfd,err4042,strlen(err4042),0);
    }
    else if (auth_ok==0)
    {
```

```

// not authenticated, return 401 error message
//local_time=localtime(&calendar_time);

printf("%s", "Wrong User:Password\r\n");

nw = write_s(newsockfd, err4011, strlen(err4011), 0);
nw = write_s(newsockfd, fpath, fpath_length, 0);
nw = write_s(newsockfd, " ", 2, 0);
//nw =
write_s(newsockfd, asctime(local_time), strlen(asctime(local_time)));
nw = write_s(newsockfd, err4012, strlen(err4012), 0);
}
else if (!strcmp(fpath, "/")) // authenticated ok - continue to
parse the get string
    staticpage(&index_htm);
else if (!strcmp(fpath, "/index.htm"))
    staticpage(&index_htm);
else if (!strcmp(fpath, "/rhttpd_web.gif"))
    staticgif(&rhttpd_web_gif);
else if (!strcmp(fpath, "/echopage.cgi"))
    echopage();
else if (!strcmp(fpath, "/statuspage.cgi"))
    statuspage();
else if (!strcmp(fpath, "/plug.htm"))
    plugpage(&plug_htm); // this is a special one
else if (!strcmp(fpath, "/swoff.gif"))
    staticgif(&swoff_gif);
else if (!strcmp(fpath, "/swon.gif"))
    staticgif(&swon_gif);
else if (!strcmp(fpath, "/pict0.gif"))
    staticgif(&pict0_gif);
else if (!strcmp(fpath, "/pict1.gif"))
    staticgif(&pict1_gif);
else if (!strcmp(fpath, "/logovert30.jpg"))
    staticjpg(&logovert30_jpg);
// parse the plug on/off changes in here
else if (!strcmp(fpath, "/onoff.spi?io0=0")) // plug 1 off
{
    p1=(char)(0);
    nw =
write_s(newsockfd, plugchange, strlen(plugchange), 0);
    LEDMATRIX_COLUMN=(unsigned
char)(!p1+2+!p2*4+8+!p3*16); //Cathode
}
else if (!strcmp(fpath, "/onoff.spi?io0=1")) // plug 1 on
{
    p1=(char)(1);
    nw =
write_s(newsockfd, plugchange, strlen(plugchange), 0);
    LEDMATRIX_COLUMN=(unsigned
char)(!p1+2+!p2*4+8+!p3*16); //Cathode
}
else if (!strcmp(fpath, "/onoff.spi?io1=0")) // plug 2 off
{
    p2=(char)(0);
    nw =
write_s(newsockfd, plugchange, strlen(plugchange), 0);

```

```

        LEDMATRIX_COLUMN=(unsigned
char)(!p1+2+!p2*4+8+!p3*16); //Cathode
    }
    else if (!strcmp(fpath, "/onoff.spi?io1=1")) // plug 2 on
    {
        p2=(char)(1);
        nw =
write_s(newsockfd,plugchange,strlen(plugchange),0);
        LEDMATRIX_COLUMN=(unsigned
char)(!p1+2+!p2*4+8+!p3*16); //Cathode
    }
    else if (!strcmp(fpath, "/onoff.spi?io2=0")) // plug 3 off
    {
        p3=(char)(0);
        nw =
write_s(newsockfd,plugchange,strlen(plugchange),0);
        LEDMATRIX_COLUMN=(unsigned
char)(!p1+2+!p2*4+8+!p3*16); //Cathode
    }
    else if (!strcmp(fpath, "/onoff.spi?io2=1")) // plug 3 on
    {
        p3=(char)(1);
        nw =
write_s(newsockfd,plugchange,strlen(plugchange),0);
        LEDMATRIX_COLUMN=(unsigned
char)(!p1+2+!p2*4+8+!p3*16); //Cathode
    }
    else // finally give up!
    {
        // file not found page 404
        //local_time=localtime(&calendar_time);

        printf("File not found 404 %s\r\n", fpath);

        nw = write_s(newsockfd,err4041,strlen(err4041),0);
        nw = write_s(newsockfd,fpath,fpath_length,0);
        nw = write_s(newsockfd," ",2,0);
        //nw =
write_s(newsockfd,asctime(local_time),strlen(asctime(local_time)),0);
        nw = write_s(newsockfd,err4042,strlen(err4042),0);
    }

    if (nw < 0)
        printf("The browser closed the connection %d\r\n",nw);
        //error("ERROR writing to socket");

close_s(newsockfd);
} //while(1)

//return 0;
}

```