# lwHTTP

This documents presents information about how tou use lwHTTP. Examples are based on the SP605 board but this manual describes what is needed in order for it to work in other designs.

## About the Library

lwHTTP is a tool to implement an http server on a microncontroller. It is designed to work in "superloop" mode, where no OS or multithreading support are necesary. The following premises were considered in it's design:

- Static content(html pages, scripts, or images) must be delivered whith HTTP's GET mechanism. When you pack this resources you can specify the "cacheability" of these.
- Dynamic content should be delivered through WebSockets by your application code. Ajax is an old technology and WebSocket has more capabilities and is being supported but most of the user platforms already. Is even more optimal than HTTP for data transmission in low processing capacity systems.

The library has some other features that benefit performance:

- Purely C written
- To accelerate the match of static resources URLs, a hash-table is implemented.
- Resources are mem-cached and in a very optimal layout.
- Zero-copy must be a priority. It's done that way for the static resources.
- Can support many connections simultaneously, HTTP's GETs and multiple WebSockets protocols.
- Macros for defining the library's memory usage like buffers size and max number of simultaneous connections can be defined at compile time from a file in your application, no need to modify library's source code.
- It's not invasive with lwip, yo ustill can use separate servers.

To pack your static content you have to use the tools at https://github.com/RaulHuertas/rhpackageexporter.

## Using the example:

From the repositorie, download the file "lwHTTP.ace" and "site.rhd". The ace file contains both the fpga and software while the rhd file contains the packaged web site.

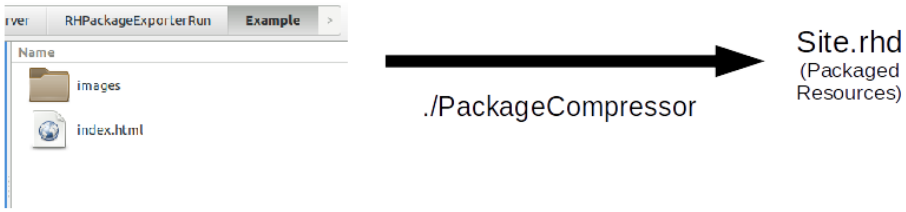
Connect the Compact Flash fro myour SP605 to your board. Assuming it gets mounted on "F:" drive, do the following

- Copy the ace file into one of the folders of your SystemACE setup (F:\XYLYNX\cf6\lwHTTP.ace)
- Create a folder called "site" and copy there the "site.rhd" file
- (F:\site\site.rhd)
- Eject the Compact flash and insert it into your board
- Connect the board to ethernet
- Use an mini-usb cable to connect the uart output to your PC and see the program output. Use a terminal program(putty,Hyperterminal,cutecom...) and set it up to 115200 bps.
- Configure the board to use load from the Compact Flash and boot.
- The board will use the static IP 192.168.1.10. Test it whith the ping utility
- Open a browser window and enter the address http://192.168.1.10.
- Follow the instructions of the page to control the LEDs of the board. Push the buttons on your board and see the changes reflected on the browser.

## How to use it

There is essentially three steps to use lwHTTP in your application:

1. Prepare your web contents, store it on your board:




2. Write your WebSockets protocol and register them with lwHTTP:

```
########YOUR SERVICE CODE
int basiWSApp_ConnStarted(struct lwHTTPConnection*); //Your code
int basiWSApp_ConnRcvd(struct lwHTTPConnection*);    //Your code
int basiWSApp_ConnFinished(struct lwHTTPConnection*);//Your code
int basiWSApp_Register(struct lwHTTPDispatcher*);    //Your code
int basiWSApp_ConnEvaluate();                        //your code
########main.c
struct lwHTTPDispatcher lwHTTPDispatcherObject;     //
...
basiWSApp_Register(&lwHTTPDispatcherObject); //register your ws protocol serv
while (...) {                               //Your app main loop
    ...
    lwHTTPDispatcher_Evaluate(&lwHTTPDispatcherObject);
    basiWSApp_ConnEvaluate();
    ...
}
```

3. Test

Apart from your server code, when linking your executable, lwHTTP expects to find the following symbols :

- lwHTTPDefaulIsClientValidQ(): So you can can accept or reject connections with your criteria. The default implementation simply returns '0'(accept all connections.
- lwHTTPDefaultGetTime(): So it can read tiem from your board. time out macros are specified in your timer units
- lwHTTPSite_LoadFromFile(): To load the packaged resources of your site.

Check an implementation of this methods in the file "lwhttp_apineeds.c" in the example app.

The library expects a file "lwhttp_apineeds.c" in the building dir to check the deesired values of some macros used to control the memory usage.

- LWHTTP_CONN_TX_BUFFER_SIZE.- Used to control the size of the transmission buffers.
- LWHTTP_CONN_RX_BUFFER_SIZE.- Used to control the size of the reception buffers. **MUST** be a power of 2.
- LWHTTP_MAX_CONNS.- How many connections can be opened at the same time.
- LWHTTP_MAX_WRITE_SIZE.- How mant bytes sent at most in each tcp package transmitted.
- LWHTTP_MAX_WS_PROTOCOL_NAME_LEN.- Max length accepted for the WebSockets protocols to accept.
- LWHTTP_MAX_WS_PROTOCOLS.- How many WebSockets protocol are going to be used at most in the application.