Getting Started with Ethernet on the STM32 Nucleo

Using STM32CubeMX with Light-Weight IP (LwIP) and System Workbench for STM32 (Eclipse)

Daniel W Rickey CancerCare Manitoba drickey@cancercare.mb.ca 2018-03-07

Introduction

The Nucleo boards produced by ST Microelectronics are wonderfully powerful and cheap. It is fantastic that ST doesn't treat developers as just another revenue stream. On a whim I bought a Nucleo-F767ZI, which has a built-in ethernet connector (not Wifi). I found that setting up LwIP was a fairly involved process, thus I made these notes. I am assuming the reader, like myself, is new to these environments and will find these useful. To avoid complexities, I have not made use of RTOS.

CubeMX

Fortunately CubeMX has improved a great deal since it was first introduced. It is used here to configure the peripherals and LwIP. The first step is to select the board from the list.

oard Filters	-	S -NUC	CLEO-F767ZI						
	©	м		STMicroelectro	onics NUCLE	O-F767ZI Board	Support and E	xamples	í
Part Number Search	8			ACTIVE		Unit Price	(US\$):23.0		
Q	-		SIM32 F/	Product is in mass	production	Mounted d	evice: STM32F767ZI	Tx	
Check/Uncheck All	×	coodi (tran mp)		The STM: to try out combinati STM32 m reduces p The ST Z	32 Nucleo-144 t new concepts a ons of performa icrocontroller. F power consumpt io connector, wh	oards provide an aff nd build prototypes b nce and power consu or the compatible bo ion in Run mode. nich extends the Ardu	ordable and flexibl y choosing from tr umption features, j ards, the SMPS sig ino™ Uno V3 com	le way for users he various provided by the gnificantly nectivity, and	
Check/Uncheck All		000000000000000000000000000000000000000		of the Nuc shields. The STM	cleo open devel	opment platform with	a wide choice of s	obe as it	
Evaluation Board		000000000000000000000000000000000000000		integrates The STM software I	s the ST-LINK/V 32 Nucleo-144 b Ibraries and exa	2-1 debugger/program board comes with the imples available with	sTM32 comprehe the STM32Cube N	nsive free I/CU Package.	
Evaluation Board Nucleo 144 MCU Series	8	0000000	Features	Large Picture	the ST-LINK/V2 32 Nucleo-144 b Ibraries and exa	2-1 debugger/program oard comes with the imples available with Docs & Resource	sTM32 comprehe the STM32Cube N	ensive free ACU Package.	ct
Evaluation Board Nucleo 144 MCU Series Check/Uncheck Al STM32F2	8	Boards List:	Features 3 litems	integrates The STM: software I	the ST-LINK/V/ 32 Nucleo-144 b Ibraries and exa	2-1 debugger/program oard comes with the imples available with Docs & Resource	s any soperatory promer. STM32 comprehe the STM32Cube N ss Buy	Insive free MCU Package.	sct
Evaluation Board Viudeo 144 HCU Series Check/Uncheck Al STM32F2 STM32F3 STM32F4	۲	Boards List:	Features 3 Items Overview	Integrates The STM: software I	the ST-LINK/V 32 Nucleo-144 b Ibraries and exa Datasheet	2-1 debugger/program oard comes with the imples available with Docs & Resource Marketing Status	Unit Price (US\$)	ACU Package.	ect X
Evaluation Board Vitudeo 344 MCU Series Check/Induck Al STM32F2 STM32F3 STM32F7 STM32F7 STM32F7	8	Boards List:	Features	Patt No NUCLEO-F7222E	the ST-LINK/V3 32 Nucleo-144 b Ibraries and exa Datasheet Type Nucleo144	2-1 debugger/program ovard comes with the imples available with Docs & Resource Marketing Status Active	Unit Price (US\$)	Mounted Device	ect
 ■valuation Board ✓ Nucles 144 Mucles 144 MUS2retes MU32r2 MU32r3 MU32r3 MU32r4 ✓ STM32r7 ✓ STM32r4 ✓ STM32r4 ✓ STM32r4 ✓ STM32r4 ✓ STM32r4 ✓ STM32r4 	0	Boards List:	Features	Part No Part No NuCLEO+F74226	s the ST-LINK/V2 32 Nucleo-144 t Datasheet Datasheet Type Nucleo144	1-1 debugger/program 1 debugger/prog	Inmer: STM32 comprehe It STM32Cube k Its Unit Price (USS) 19.0 23.0	Noive free ACU Package. Start Proje Mounted Device STM32F7422Tx STM32F7462GTx	x

CubeMX will ask if all peripherals should be initialised to their default mode. For the Nucleo boards, I'm not sure this makes much difference as there are not very many peripherals. I selected "yes".



As shown below, enable LWIP, which is listed under "MiddleWares". Check that the ethernet "Eth" is enabled.



The Nucleo boards contains LEDs that the user can turn on and off. These are used in this example. Note that ST failed here because different Nucleo boards use different names for the same LEDs. So much for code portability. This is a good time to note the names used by your board: here they are "LD2" and "LD3".



The next step is to configure the various options for LwIP. These are accessed through the "LWIP" button on the "Configuration" tab.

Configuration	Power Consumption Calculator					
			Middle	ewares		
5. B	Multimedia	Connectivity ETH USART3 CONSTRUCTION	Analog	System	Control	Security

At this point development is a whole lot easier if you hardwire the IP address of the board. That is, disable DHCP (LWIP_DHCP). I plugged my board into the same switch as my development computer so it was easy to reach. Pick an address that's reachable but not already used by another device on your network: you could ping the proposed address to make sure it's free.

Also note that RTOS is disabled. A simple poling loop will be used instead.

	Statistics Checksum Debug User Constants
General Settings	PPP PIPv6 PITPD
nfigure the below parameters :	
arch : Search (Crtl+F)	
LwIP Version	
LwIP Version (Version of LwIP supported by CubeMX ** CubeMX specific	. 2.0.3
IPv4 - DHCP Options	
LWIP_DHCP (DHCP Module)	Disabled
IP Address Settings	
IP_ADDRESS (IP Address)	192.168.000.055
NETMASK_ADDRESS (Netmask Address)	255.255.255.000 IP_ADDRESS (IP Address)
GATEWAY_ADDRESS (Gateway Address)	192.168.000.002
RTOS Dependency	
WITH_RTOS (Use FREERTOS ** CubeMX specific **)	Disabled
Protocols Options	
LWIP_ICMP (ICMP Module Activation)	Enabled
LWIP_IGMP (IGMP Module)	Disabled
LWIP_DNS (DNS Module)	Disabled
LWIP_UDP (UDP Module)	Enabled
MEMP_NUM_UDP_PCB (Number of UDP Connections)	4
LWIP_TCP (TCP Module)	Enabled
MEMP NUM TCP PCP (Number of TCP Connections)	5

Most of the settings were left at their defaults. However, to enable the webserver, the following options were enabled: LWIP_HTTPD, LWIP_HTTPD_CGI and LWIP_HTTPD_SSI. Because the Nucleo board has bucket loads of memory, the maximum tag length (LWIP_HTTPD_MAX_TAG_NAME_LEN) was increased from 8 to 16 characters.

I found out that it is important to set these options before importing the project into Eclipse. I encountered bizarre behaviour when I used CubeMX to enable an option after the project had been imported.

Note that I didn't bother with "HTTP CGI NEW STYLE" as I could not find much in the way of documentation or examples.

SNTP	MDNS/T	FTP	Perf/Checks	d	Statistics	Y	Checksum	Debug	User Constants
ď	General Settings	2	Key Options		PPP		V IPv6	- WHIPD	SNMP
nfigure the l	pelow parameters :								
arch : Sea	rch (Crtl+F)		•					V Show	Advanced Parameters
HTTPD Op	tions								
LWIF	P_HTTPD (LWIP HTT	PD Support **	CubeMX specific **)		Ena	bled			
LWIF	-HTTPD_CGI (HTTP	CGI Old Style	e)		Ena	bled			
LWIF	HTTPD_CGI_SSI (HTTP CGI Nev	/ Style)		Dis	abled			
LWIF	HTTPD_SSI (HTTP	Server Side I	ncludes)		Ena	bled			
LWIF	HTTPD_SSI_RAW	(HTTP SSI Ta	g Handler Callback)		Dis	abled			
LWIF	HTTPD_SUPPORT	POST (HTTP	POST)		Dis	abled			
LWIF	HTTPD_MAX_CGI	PARAMETERS	(Max Sent Parameters	Numbe	er for CGI) 16				
LWIF	HTTPD_SSI_MULT	IPART (Serve	-Side-Includes Multipart	.)	Dis	abled			
LWIF	HTTPD_MAX_TAG	_NAME_LEN (I	Max Tag Name String Le	ngth)	16				
LWIF	HTTPD_MAX_TAG	INSERT_LEN	(Max Tag Inserted Strin	g Leng	th) 193	192			
LWIF	HTTPD_POST_MA	NUAL_WND (H	TTP POST Manual WND)	Dis	abled			
HTTP	D_SERVER_AGENT	(HTTP Server)		Tw	"lwIP/2.0.0 (http://savannah.nongnu.org/projects/lwip)"			
LWIF	HTTPD_DYNAMIC	HEADERS (H	TTP Dynamic Headers Cr	eation) Dis	Disabled			
HTTP	D_USE_MEM_POOL	(HTTP Use M	emory Pool)		Dis	abled			
HTTP	D_SERVER_PORT (HTTP Server F	ort)		80				
HTTP	D_MAX_RETRIES (HTTP Max Ret	ries)		4				
HTTP	D_POLL_INTERVAL	(HTTP Pool de	elay)		4				
HTTP	D_TCP_PRIO (HTTP	P TCP Priority)			1				
LWIF	-HTTPD_TIMING (H	TTP Timing)			Dis	abled			
LWIF	HTTPD_SUPPORT	EXTSTATUS	Error Pages Display)		Dis	abled			
LWIF	HTTPD_SUPPORT	_V09 (Drop HT	TP 0.9 Support)		Ena	bled			
LWIF	-HTTPD_SUPPORT	_11_KEEPALI	E (HTTP 1.1 Persistent	Connec	tions) Dis	abled			
LWIF	HTTPD_SUPPORT	REQUESTLIS	T (HTTP Request from M	ultiple	Packets) Ena) Enabled			
LWIF	HTTPD_REQ_QUE	UELEN (Numb	er of Incoming Rx Pbufs)	5				
LWIF	HTTPD_REQ_BUR	SIZE (Number	of TCP Payload in Pbufs)	103	1023			
LWIF	_HTTPD_MAX_REQ	LENGTH (Ma	x Number of HTTP Requ	est)	103	23			
VIP_HTTP /IP_HTTPD trameter I arning: Th served for a	D (LwIP HTTPD S Description: is parameter introdu advanced users.	upport ** Cu	IbeMX specific **)	ack. En	able HTTPD requ	ires use	er specific code n	ot yet generated by	STM32Cube MX. This is

Under "Advanced Settings", it is possible to disable initialisation of the peripherals. Here, I disabled the UART and USB initialisation as they are not needed for this example.

roject Code Ge	nerator Advanced Settings				
Driver Selector –					
Search : Search	(Crtl+F)	-		2↓ 💷	
USART			HAL		
LWIP			HAL		
RCC			HAL		
USB_OTG_FS			HAL		
CORTEX_M7			HAL		
			HAL		
GPIO			HAL		
GPIO Generated Funct	on Calls	TD Instance Mar	HAL	Tuskilin, (Cauca)	
GPIO Generated Funct Rank	on Calls Function Name	IP Instance Nam	e Not Generate Function Call	Visibility (Static)	
GPIO Generated Funct Rank	on Calls Function Name MX_GPIO_Init	IP Instance Nam GPIO	e Not Generate Function Call	Visibility (Static)	
GPIO Generated Funct Rank	on Calls Function Name MX_GPIO_Init SystemClock_Config	IP Instance Nam GPIO RCC	e Not Generate Function Call	Visibility (Static)	
GPIO Generated Funct Rank	on Calls Function Name MX_GPI0_Init SystemClock_Config MX_CORTEX_M7_Init	IP Instance Nam GPIO RCC CORTEX_M7	HAL e Not Generate Function Call	Visibility (Static)	
GPIO Generated Funct Rank 1 2 3 4	Function Name MX_GPIO_Init SystemClock_Config MX_CORTEX_M7_Init MX_USART3_UART_Init	IP Instance Nam GPIO RCC CORTEX_M7 t USART3	HAL	Visibility (Static)	
GPIO Generated Funct Rank 1 2 3 4	IN Calls Function Name MX_GPIO_Init SystemClodc_Config MX_CORTEX_M7_Init MX_USART3_UART_Init MX_USART3_UART_Init MX_USA_TG_FS_PC.	IP Instance Nam GPIO RCC CORTEX_M7 t USART3 . USB_OTG_FS	HAL	Visibility (Static)	

Export the project using the toolchain "SW4STM32" (System Workbench for STM32). I think you want "Generate Under Root" enabled.

oject Code Generator Adv	anced Settings			
Project Settings				
Project Name				
myServer1				
Project Location				
C: \Users \cameras \Documen	its\STM32 Stuff			
Toolchain Folder Location				
C: \Users\cameras\Documen	its\STM32 Stuff\myServer1\			
Toolchain / IDE				
SW4STM32		▼ Generate Under Rom	ot	
Minimum Stack Size	0x400			
Mcu Reference				
STM32F767ZITx				
Firmware Package Name and	Version			
STM32Cube FW_F7 V1.9.0				
	ocation			
🔽 Use Default Firmware Lo				Browse
Use Default Firmware Lo	ube/Repository/STM32Cube_f	FW_F7_V1.9.0		Brombe
Use Default Firmware Lo	ibe/Repository/STM32Cube_F	FW_F7_V1.9.0		biolise

The following dialogue box is presented. I just selected "Close".

Code	le Generation
1	The Code is successfully generated under C:/Users/cameras/Documents/STM32 Stuff/myServer
	Open Folder Open Project Close

ST-Link

It is a good idea to confirm that the ST-Link debugger has the latest firmware on it. Use the ST-Link utility to do this. This also ensures that the required device drivers have been installed.

Memory display Address: 0x0	18000000 🕶 Si	ze: 0x0FF) Data W	idth: 32 bits	Devic Devic Revis	e ID ion ID	STM32F76x 0x451 Rev Z	
Device Memory (⊉ 0x08000000 :	Binary File			Flash	size	2MBytes	Update
arget memory, A	ddress range: [0x	08000000 0x080	00FF0]			_		
Address	0	4	8	С	ASCII			^
0x08000000	20080000	080097F1	08009841	08009841	ñ — A `	<mark>A</mark> ~.		
0x08000010	08009841	08009841	08009841	00000000	A~A~A~			
0x08000020	00000000	00000000	00000000	08009841		۹~		
0x08000030	08009841	00000000	08009841	08009793	A~A~.	."—.		
0x08000040	08009841	08009841	08009841	08009841	A~A~A~	A~.		
0x08000050	08009841	08009841	08009841	08009841	A~A~A~	A~.		-
•	III							•
21:53:03 : ST-LIN 21:53:03 : ST-LIN 21:53:03 : Conne 21:53:03 : SWD F 21:53:03 : Conne 21:53:03 : Debug 21:53:03 : Device	K SN : 0672FF525 K Firmware versic cted via SWD. irequency = 4,0 N ction mode : Norm in Low Power mo : ID:0x451 - Stab Stab : 2MPbb	57508772670720 n : V2J29M18 1Hz. nal. de enabled.	32					

Importing into Eclipse

Importing into Eclipse requires a couple of steps but works well. Import through the dropdown menu file:import as "Existing Projects into Workspace".

Select	~
Create new projects from an archive file or directory.	
Select an import wizard:	
type filter text	
🔺 🗁 General	
🚇 Archive File	
Existing Projects into Workspace	
😂 File System	
Preferences	
Projects from Folder or Archive	E
▷ 🗁 C/C++	
D Git	
De Install	
Demok	
Bemote Systems	
Rendle Systems	
Run/Debug	
N PASTIN	-

Browse to the folder holding the project. There should only be one project visible, and tell it to finish. If the project imports properly, you should be able to few the various files.

Import		
Import Projects Select a directory to sear	ch for existing Eclipse projects.	
 Select root directory: Select archive file: 	\Users\cameras\Documents\STM32 Stuff\myServer1	Browse Browse
✓ myServer1 (C:\U	sers\cameras\Documents\STM32 Stuff\myServer1)	Select All
Options Options Search for nested pro Copy projects into w Hide projects that all	ojects orkspace ready exist in the workspace	
Working sets Add projec <u>t</u> to work Working sets:	cing sets	Ne <u>w</u>
?	< Back Next > Finish	Cancel

Missing bits

If you try to build the project it will fail miserably as shown in the following figure. The problem is that there is a very important file missing: fsdata.c. This is the file that holds the content of your website and is needed by HTTPD.

50	cription	
6	8 Errors (3 items)	
	🔕 fatal error: fsdata.c: No such file or directory	
	📀 make: *** [Middlewares/Third_Party/LwIP/src/apps/httpd/fs.o] Error 1	
	😣 recipe for target 'Middlewares/Third_Party/LwIP/src/apps/httpd/fs.o' failed	
	🚯 Warnings (4 items)	
	MX_USART3_UART_Init' defined but not used [-Wunused-function]	
	MX_USB_OTG_FS_PCD_Init' defined but not used [-Wunused-function]	
	Unused static function 'MX_USART3_UART_Init'	
	Unused static function 'MX_USB_OTG_FS_PCD_Init'	
	III.	

To quickly try out the HTTP server, you can make use of an existing fsdata.c that is included as an example with CubeMX. You may need to search your drive for it although one is located at:

...\STM32Cube\Repository\STM32Cube_FW_F7_V1.9.0\Middlewares\Third_Party\LwIP\src\apps\htt pd

Annoyingly LwIP has hard coded the path so you will need to copy fsdata.c to the following folder:

...\myServer1\Middlewares\Third_Party\LwIP\src\apps\httpd

Note that although fsdata.c appears to be a C source file, it must not be compiled because truly bizarre error messages will result. Select fsdata.c in the project explorer and exclude it from the build as shown in the following two figures.



Exclude from b	ouild	
Exclude object(s)) from build in the follo	wing configurations
	Select All	Deselect All
?	ОК	Cancel

Ping

At this point the code should build but it won't work properly. The function call *MX_LWIP_Process()* needs to be added to the while loop in main.

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
   //Read a received packet from the Ethernet buffers and Send it to the lwIP stack for
handling
   MX_LWIP_Process();
   /* USER CODE END WHILE */
   /* USER CODE BEGIN 3 */
   }
   /* USER CODE END 3 */
```

The code should be able to build. Then you should be able to ping your board as shown in the following figure.

Viewing a webpage

The next step is to start the webserver by adding *httpd_init()* to main. Be sure to add the corresponding include statement for httpd.h.

```
/* USER CODE BEGIN Includes */
#include "lwip/apps/httpd.h"
/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_LWIP_Init();
/* USER CODE BEGIN 2 */
//start the web server
httpd_init();
```

/* USER CODE END 2 */

When the code is running on the Nucleo board, you should be able to access a webpage by typing http://192.168.0.55 into a web browser. At this point you should be able to view the webpage in a browser.

Making a Webpage

You will need to make your own webpage. Because the microcontroller doesn't (usually) have a file system, the webpages are converted to a single file fsdata.c that is included at compile time. To do this use the command line utility htmlgen.exe in the DOS command line or makefsdata in unix. Rather annoyingly a binary for DOS of this utility is not included with LWIP but there is one on the web ...somewhere.

For example, if the html files are contained in the folder "leds", the command is *htmlgen leds* - *f:fsdata.c.* The folder should contain an index.html file and a 404.html file at a minimum. The 404.html file is helpful if you make a mistake and tell the server to load a non-existent webpage. Otherwise, it would just sit there like a bump and you won't know what is wrong.

Note that you should do a clean rebuild after changing fsdata.c.

Simple CGI handler

Implementing a CGI handler isn't too difficult. Start with a simple web page that has two checkboxes on it. The web page used here is really short.

```
<!DOCTYPE html>
<html><head><title>LED Test</title>
```

This allows you to control the LEDs: LED1 and LED2. You have to click on "Send" button to change the LEDs

```
<form method="get" action="/leds.cgi">
<input value="1" name="led" type="checkbox">LED1<br>
<input value="2" name="led" type="checkbox">LED2<br>
<br>
<input value="Send" type="submit"> </form>
</html>
```

The idea behind this webpage is it allows the user to turn on the LEDs on the Nucleo board. So you will need to enable these LEDs with code similar to the following

```
/* USER CODE BEGIN PV */
/* Private variables ------*/
static GPIO_InitTypeDef GPIO_InitStruct;
/* USER CODE BEGIN 2 */
//setup the blue LED
//note that different Nucleo boards use different names for the same LEDs
GPIO_InitStruct.Pin = LD2_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_PULLUP;
GPIO_InitStruct.Speed = GPIO_SPEED_FAST;
HAL_GPIO_Init(LD2_GPIO_Port, &GPIO_InitStruct);
//setup the red LED
GPIO_InitStruct.Pin = LD3_Pin;
```

```
GPI0_InitStruct.Mode = GPI0_MODE_OUTPUT_PP;
GPI0_InitStruct.Pull = GPI0_PULLUP;
GPI0_InitStruct.Speed = GPI0_SPEED_FAST;
HAL_GPI0_Init(LD3_GPI0_Port, &GPI0_InitStruct);
```

The CGI handler for turning on the LEDs is the following. The function header is defined in httpd.h as type tCGIHandler.

/* USER CODE BEGIN 0 */

```
/**** CGI handler for controlling the LEDs ****/
// the function pointer for a CGI script handler is defined in httpd.h as tCGIHandler
const char * LedCGIhandler(int iIndex, int iNumParams, char *pcParam[], char *pcValue[])
{
    uint32_t i=0;

    // index of the CGI within the theCGItable array passed to http_set_cgi_handlers
    // Given how this example is structured, this may be a redundant check.

    // Here there is only one handler iIndex == 0
    if (iIndex == 0)
    {
        // turn off the LEDs
    }
}
```

```
HAL_GPI0_WritePin(LD3_GPI0_Port, LD2_Pin, GPI0_PIN_RESET);
 HAL_GPI0_WritePin(LD3_GPI0_Port, LD3_Pin, GPI0_PIN_RESET);
 // Check the cgi parameters, e.g., GET /leds.cgi?led=1&led=2
  for (i=0; i<iNumParams; i++)</pre>
   ł
   //if pcParmeter contains "led", then one of the LED check boxes has been set on
   if (strcmp(pcParam[i], "led") == 0)
     ł
     //see if checkbox for LED 1 has been set
     if(strcmp(pcValue[i], "1") == 0)
      {
      // switch led 1 ON if 1
      HAL_GPI0_WritePin(LD3_GPI0_Port, LD2_Pin, GPI0_PIN_SET);
      }
      //see if checkbox for LED 2 has been set
     else if(strcmp(pcValue[i], "2") == 0)
      {
      // switch led 2 ON if 2
      HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_SET);
      }
     }//if
   } //for
 }//if
//uniform resource identifier to send after CGI call, i.e., path and filename of the
response
return "/index.html";
```

} //LedCGIhandler

We need to tell the HTTPD code to use this handler. To do this, the LedCGIhandler is added to a list. First a structure is created that shows that leds.cgi corresponds to the LedCGIhandler.

```
/* USER CODE BEGIN PV */
/* Private variables ------*/
// prototype CGI handler for the LED control
const char * LedCGIhandler(int iIndex, int iNumParams, char *pcParam[], char *pcValue[]);
// this structure contains the name of the LED CGI and corresponding handler for the LEDs
const tCGI LedCGI={"/leds.cgi", LedCGIhandler};
//table of the CGI names and handlers
tCGI theCGItable[1];
```

Also add the following function to /* USER CODE BEGIN 0 */

```
// Initialize the CGI handlers
void myCGIinit(void)
{
   //add LED control CGI to the table
   theCGItable[0] = LedCGI;
   //give the table to the HTTP server
   http_set_cgi_handlers(theCGItable, 1);
} //myCGIinit
```

Then in the main code, call myCGIinit.

```
//start the web server
httpd_init();
//initialise the CGI handlers
myCGIinit();
```

The code should compile and run. With some luck the you will be able to set the LEDs from the webpage.

🚖 🥝 Remote Access Gateway
This allows you to control the LEDs: LED1 and LED2. You have to click on "Send" button to change the LEDs
□LED1 □LED2
Send

Server side includes (SSI)

Server side includes let the microcontroller generate text that is displayed on the webpage. For example, this could be used to display the output of an analogue-to-digital converter . To do this, tags located in the HTML code are replaced by text when the webpage is served to the client. Note that the SSI function is called each time the HTTPD server detects a tag of the form <!--#name--> in a .shtml, .ssi or .shtm file. It won't work if the file has a .html extension. Start by adding a couple of lines to the previous webpage as shown below. Note that the filename must be index.shtml, and you will need to adjust this file name in the LedCGIhandler function. Use the command line utility to convert this to fsdata.c

```
<!DOCTYPE html>
<html><html><html><
title>LED Test</title>
<body>
This allows you to control the LEDs: LED1 and LED2. You have to click on "Send" button
to change the LEDs
<form method="get" action="/leds.cgi">
<input value="1" name="led" type="checkbox">LED1<br>
<input value="1" name="led" type="checkbox">LED1<br>
<input value="2" name="led" type="checkbox">LED1<br>
<br/><br>
```

```
text for tag1: <!--#tag1-->
text for tag2: <!--#tag2-->
<input value="Send" type="submit"> </form>
</body></html>
```

The HTTPD functions need a list of the tags contained in the HTML code.

```
//array of tags for the SSI handler
//these are the tags <!--#tag1--> contained in the shtml file
#define numSSItags 2
char const *theSSItags[numSSItags] = {"tag1","tag2"};
```

The handler function is relatively easy to write. The iIndex tells you which tag in the array to take care of. Place the text to be displayed into pcInsert and return the number of characters inserted.

```
/**** SSI handler ****/
// This function is called each time the HTTPD server detects a tag of the form
// <!--#name--> in a .shtml, .ssi or .shtm file
// It won't work if the file has a .html extension.
u16_t mySSIHandler(int iIndex, char *pcInsert, int iInsertLen)
{
// see which tag in the array theSSItags to handle
if (iIndex == 0) //is "tag1"
  {
  char myStr1[] = "Hello from Tag #1!"; //string to be displayed on web page
  //copy the string to be displayed to pcInsert
  strcpy(pcInsert, myStr1);
  //return number of characters that need to be inserted in html
  return strlen(myStr1);
  }
  else if (iIndex == 1) //is "taq2"
  {
  char myStr2[] = "Hello from Tag #2!"; //string to be displayed on web page
  //copy string to be displayed
  strcpy(pcInsert, myStr2);
  //return number of characters that need to be inserted in html
  return strlen(myStr2);
  }
  return 0;
} //mySSIHandler
```

Call the following function from main to initialise the SSI handler.

```
/**** Initialize SSI handlers ****/
void mySSIinit(void)
{
    //configure SSI handler function
    //theSSItags is an array of SSI tag strings to search for in SSI-enabled files
    http_set_ssi_handler(mySSIHandler, (char const **)theSSItags, numSSItags);
} //mySSIinit
/* USER CODE END 0 */
```

Then you should be able to compile and run the code. With some luck the text will be displayed on the webpage as shown below.

Sending email

The LWIP code does not include an email client. However, there is an SMTP client available as a separate download. See contrib at download.savannah.nongnu.org/releases/lwip/. There are two files smtp.c and smtp.h. Place these in the Src and Inc folders in your project, respectively.

Using the email routines is really easy. Simply set up the server address and authentication. Then sending an email is a single function call.

```
//this function is called when SMTP wants to tell us something
void mySMTPresult(void *arg, u8_t smtp_result, u16_t srv_err, err_t err)
{
    printf("mail (%p) sent with results: 0x%02x, 0x%04x, 0x%08x\n", arg, smtp_result, srv_err,
    err);
} //mySMTPresult
/**** send an email using SMTP ****/
static void sendAnEmail(void)
```

```
{
#define emailFrom
                        "yourEmail@yourISP.ca"
#define emailTo
                        "drickey@cancercare.mb.ca"
                        "annoying"
#define emailSubject
#define emailMessage
                        "this is an annoying message"
int * some_argument = 0;
//IP address or DNS name for your SMTP connection
//smtp_set_server_addr("mail.yourISP.ca"); //if using DNS
smtp_set_server_addr("xxx.xxx.128.128");
                                           //if using IP address
// set both username and password as NULL if no authentication needed
smtp_set_auth("yourUserID", "yourPassowrd");
smtp_send_mail(emailFrom,
                             textTo,
                                         emailSubject,
                                                           emailMessage,
                                                                            mySMTPresult,
some_argument);
} //sendAnEmail
```

I believe the password is sent unencrypted, so be careful with your selection of email host. You can use this to send a text message to a cell phone via email. This could be used to let you know that your garage door is open or for annoying your friends.

Problems

1) If you encounter bizarre behaviour where a #define seems to change values, i.e., is "1" in one location and "0" in another, then try exporting the project from CubeMX under a new name. This will require you to reimport it into Eclipse and add the example code to main.c.

2) When using server side includes, the file must have a .shtml extension otherwise the tags won't work.

3) Be sure to include a 404.html file in case you screw up and ask the server to show a non-existent webpage.

4) If the webpage displays but is really slow, try rebooting the whole works (computer and microcontroller).

5) Make sure fsdata.c is in the correct folder but is excluded from the build. Trying to compile fsdata.c will generate very odd errors.

6) Always do a clean rebuild after changing fsdata.c.

-- end --