Freescale Semiconductor
Application Note

Document Number: AN3930
Rev. 0, 09/2009

Email Client Using MCF51CN Family and FreeRTOS

by: Paolo Alcantara
Applications Engineering
RTAC Americas

1 Introduction

This document describes an email client using the MCF51CN128, the open source RTOS FreeRTOS® v5.3.0, and the TCP/IP stack lwIP v1.3.0.

This document presents how an embedded device can send emails using the Internet infrastructure. It gives a brief explanation about the simple mail transport protocol (SMTP) and how to use it with the MCF51CN128.

This document is intended to be used by all software development engineers, test engineers, and anyone else who has to send an email from an MCU.

Contents

2 Email Client

An email is a digital message sent from one computer to another. The protocol that deals with sending an email is basically the Simple Mail Transport Protocol (SMTP). The protocol that deals with email retrieval is basically the post office protocol (POP3) or Internet Message Access Protocol (IMAP). This document deals with sending only emails or SMTP communication.

A list of possible uses of email for embedded devices is as follows:

- Send a condition that needs to be notified to a PC. This information can be analyzed later on by a person receiving the email.
- Send log files of a sensor at regular times
- Send status information when requested by another TCP/IP application

Figure 1 shows an email client single scenario.

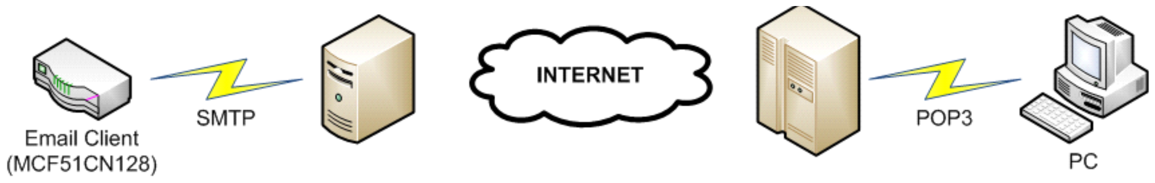


Figure 1. Email client connecting to an SMTP server

2.1 Hardware Implementation

This application note works with the MCF51CN128 reference design and the Tower System. For more information about the MCF51CN128 reference design, go to the MCF51CN128 Product Summary Page. For Tower System information, visit www.Freescale.com/tower.

A hardware block diagram about the MCF51CN128 reference design is presented for clarity.

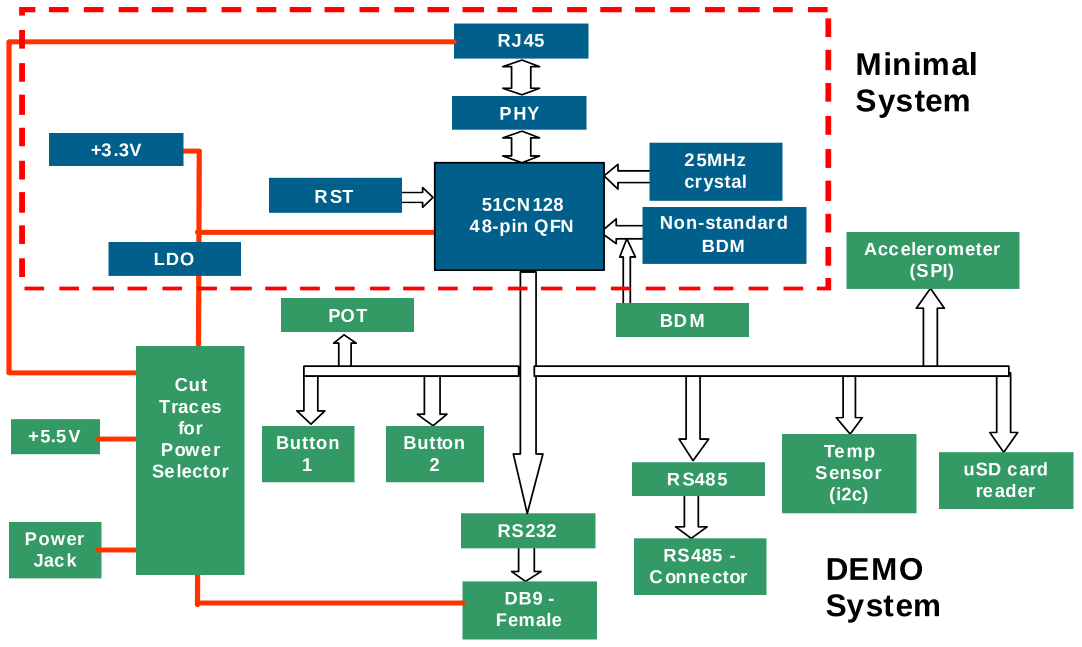


Figure 2. Hardware block diagram of MCF51CN128 reference design board

For the MCF51CN128 reference design hardware, jumpers must remain in the same position. The board is then ready to use as it is. Board schematics, layout, and gerber files are provided in case a customization is required in the hardware for specific use of the email client, like removing additional components besides Ethernet or sensors.

For the TWR-MCF51CN Tower Rev C, default jumper configuration must be used.

2.2 Principle of Operation

The email client is a useful tool that sends information through the Internet to another computer in an asynchronous scheme. This means sent information can be analyzed by a person or by a computer as soon as it is received, or at any another moment. All of this relies on the use of the SMTP server that completes the process of sending an email. The following application deals only with server-client communication to an SMTP server. The rest of the process after the email is stored in the email receiver server goes beyond the scope of this document.

As already mentioned, email works with the client-server model spread on most of the TCP applications. In the process of sending an email, application software acts as a client that sends commands to the server, and receives a message as a result of each request. This communication is completed until the email is successfully sent or communication is aborted.

The software that comes with this application note tests the email client functionality in two ways:

- An internal email SMTP server at Freescale Semiconductor LAN—Internal test
- An external email SMTP server outside of Freescale using a yahoo.com email account—external test

For this application, as soon as the DHCP gets a valid IP address, it sends its IP address by email to an email account stored in the MCU memory.

If the email is not correctly sent, the following is a list of possible causes.

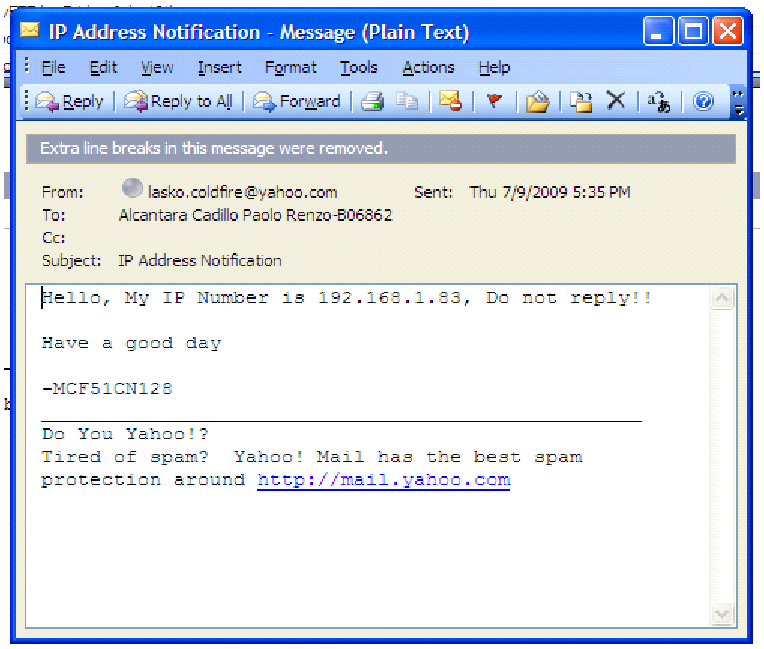
- LAN DHCP server cannot provide a valid IP address
- LAN does not have access to the Internet
- LAN does have SMTP port 25 blocked
- LAN DNS server does not recognize the SMTP server address to connect to it
- SMTP server cannot validate username or password provided
- SMTP server is not responding to requests, then a time-out occurred
- SMTP server requires an SSL session to validate the user

Approximated time to receive an email can vary from 1 minute to 10 minutes or more, even if the SMTP server showed a successful communication occurred. This indicates email is not a real time communicator.

A way to debug problems during communication with the SMTP server is to use a network analyzer like Wireshark. This can be downloaded from the Internet under the GNU General Public License. Wireshark shows all the packets in the sub-LAN by using a network hub. A router or a network switch does not work because they filter by using a MAC address and requests will not be read by the Wireshark network analyzer.

The application can send an email with whatever text needed. However, the text memory space can be overwritten only after the email is sent and connection is closed. The text has this restriction because the text is passed to the email application as a reference and not as a copy. A software set of functions are provided. See Section 5, "Email Client API."

If the application uses a static IP address scheme, then the email will not be sent. This restriction comes from the application project because the email is sent only if the dynamic host configuration protocol (DHCP) service is started. For details on how to change media access controller (MAC) parameters, refer to application note *Serial-to-Ethernet Bridge Using MCF51CN Family and FreeRTOS* (document AN3906).


IP Address Notification - Message (Plain Text)
File Edit View Insert Format Tools Actions Help
Reply Reply to All Forward

Extra line breaks in this message were removed.

From: lasko.coldfire@yahoo.com Sent: Thu 7/9/2009 5:35 PM
To: Alcantara Cadillo Paolo Renzo-B06862
Cc:
Subject: IP Address Notification

```
Hello, My IP Number is 192.168.1.83, Do not reply!!

Have a good day

-MCF51CN128
____________________________________________________
Do You Yahoo!?
Tired of spam?  Yahoo! Mail has the best spam
protection around http://mail.yahoo.com
```

Figure 3. Email received from MCF51CN128

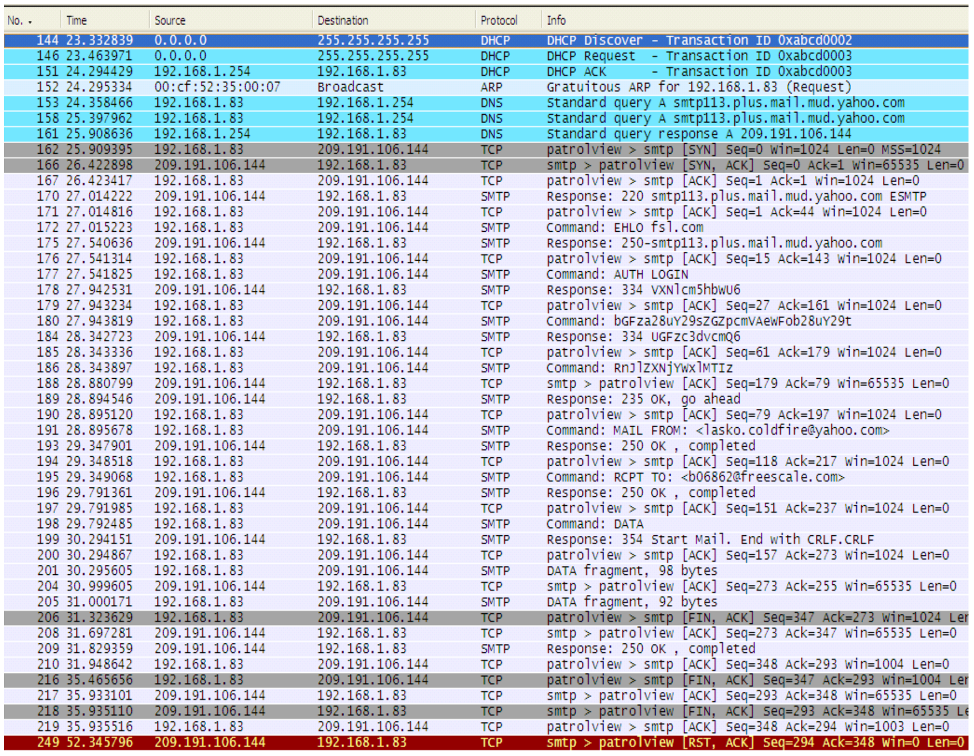
The email client application starts with the following configuration, but can be changed at runtime using the configuration web page, viewable through a web browser.

Table 1. Default MAC Parameters

MAC Parameters	
MAC Address	00:CF:52:35:00:07
IP Address	192.168.1.3 for static implementation
Mask Address	255.255.255.0
Gateway Address	192.168.1.1
Server Address to Connect to an Address	192.168.1.3
Static or Dynamic Address	Static

3 Introduction to the Email Client Software

LwIP provides three levels of APIs for sockets. The email client uses the netconn level. For more details about socket's level with lwIP, refer to application note *Serial-to-Ethernet Bridge Using MCF51CN Family and FreeRTOS* (document AN3906).



No.	Time	Source	Destination	Protocol	Info
144	23.332839	0.0.0.0	255.255.255.255	DHCP	DHCP Discover - Transaction ID 0xabcd0002
146	23.463971	0.0.0.0	255.255.255.255	DHCP	DHCP Request - Transaction ID 0xabcd0003
151	24.294429	192.168.1.254	192.168.1.83	DHCP	DHCP ACK - Transaction ID 0xabcd0003
152	24.295334	00:cf:52:35:00:07	Broadcast	ARP	Gratuitous ARP for 192.168.1.83 (Request)
153	24.358466	192.168.1.83	192.168.1.254	DNS	Standard query A smtp113.plus.mail.mud.yahoo.com
158	25.397962	192.168.1.83	192.168.1.254	DNS	Standard query A smtp113.plus.mail.mud.yahoo.com
161	25.908636	192.168.1.254	192.168.1.83	DNS	Standard query response A 209.191.106.144
162	25.909395	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [SYN] Seq=0 Win=1024 Len=0 MSS=1024
166	26.422898	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
167	26.423417	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=1 Ack=1 Win=1024 Len=0
170	27.014222	209.191.106.144	192.168.1.83	SMTP	Response: 220 smtp113.plus.mail.mud.yahoo.com ESMTP
171	27.014816	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=1 Ack=44 Win=1024 Len=0
172	27.015223	192.168.1.83	209.191.106.144	SMTP	Command: EHLO fsl.com
175	27.540636	209.191.106.144	192.168.1.83	SMTP	Response: 250-smtp113.plus.mail.mud.yahoo.com
176	27.541314	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=15 Ack=143 Win=1024 Len=0
177	27.541825	192.168.1.83	209.191.106.144	SMTP	Command: AUTH LOGIN
178	27.942531	209.191.106.144	192.168.1.83	SMTP	Response: 334 VXNlcm5hbWU6
179	27.943234	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=27 Ack=161 Win=1024 Len=0
180	27.943819	192.168.1.83	209.191.106.144	SMTP	Command: bGFza28uY29sZGZpcmVAeWFob28uY29t
184	28.342723	209.191.106.144	192.168.1.83	SMTP	Response: 334 UGFzc3dvcmQ6
185	28.343336	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=61 Ack=179 Win=1024 Len=0
186	28.343897	192.168.1.83	209.191.106.144	SMTP	Command: RnJlZXNjYWxlMTIz
188	28.880799	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [ACK] Seq=179 Ack=79 Win=65535 Len=0
189	28.894546	209.191.106.144	192.168.1.83	SMTP	Response: 235 OK, go ahead
190	28.895120	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=79 Ack=197 Win=1024 Len=0
191	28.895678	192.168.1.83	209.191.106.144	SMTP	Command: MAIL FROM: <lasko.coldfire@yahoo.com>
193	29.347901	209.191.106.144	192.168.1.83	SMTP	Response: 250 OK , completed
194	29.348518	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=118 Ack=217 Win=1024 Len=0
195	29.349068	192.168.1.83	209.191.106.144	SMTP	Command: RCPT TO: <b06862@freescale.com>
196	29.791361	209.191.106.144	192.168.1.83	SMTP	Response: 250 OK , completed
197	29.791985	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=151 Ack=237 Win=1024 Len=0
198	29.792485	192.168.1.83	209.191.106.144	SMTP	Command: DATA
199	30.294151	209.191.106.144	192.168.1.83	SMTP	Response: 354 Start Mail. End with CRLF.CRLF
200	30.294867	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=157 Ack=273 Win=1024 Len=0
201	30.295605	192.168.1.83	209.191.106.144	SMTP	DATA fragment, 98 bytes
204	30.999605	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [ACK] Seq=273 Ack=255 Win=65535 Len=0
205	31.000171	192.168.1.83	209.191.106.144	SMTP	DATA fragment, 92 bytes
206	31.323629	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [FIN, ACK] Seq=347 Ack=273 Win=1024 Ler
208	31.697281	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [ACK] Seq=273 Ack=347 Win=65535 Len=0
209	31.829359	209.191.106.144	192.168.1.83	SMTP	Response: 250 OK , completed
210	31.948642	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=348 Ack=293 Win=1004 Len=0
216	35.465656	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [FIN, ACK] Seq=347 Ack=293 Win=1004 Ler
217	35.933101	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [ACK] Seq=293 Ack=348 Win=65535 Len=0
218	35.935110	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [FIN, ACK] Seq=293 Ack=348 Win=65535 Le
219	35.935516	192.168.1.83	209.191.106.144	TCP	patrolview > smtp [ACK] Seq=348 Ack=294 Win=1003 Len=0
249	52.345796	209.191.106.144	192.168.1.83	TCP	smtp > patrolview [RST, ACK] Seq=294 Ack=348 Win=0 Len=0

Figure 4. Wireshark window showing SMTP requests and responses

The following email client implementation initiates a TCP connection to port 25 this the designated port the SMTP to communicate with the SMTP server. Some firewalls block TCP port 25 as a security police to avoid email spam.

The email function that sends emails is a function that invokes the Email RTOS task. This task processes the SMTP connection and is deleted as soon as the email is sent or an error occurs.

The email client is able to support authentication if a password for a username is required by the SMTP server. The username and password are ciphered to Base64, which is a multipurpose Internet mail extension (MIME) encoded transfer. Base64 has its own alphabet derived with all the printable ASCII characters like: A-Z, a-z, 0-9, + and /.

An email sent to the Internet has its content viewable by a third party. The application must make sure no sensitive information is sent using the email client.

While sending an email, the email socket task gives control back to the CPU while waiting for a response from the SMTP server. This avoid blocks the execution of critical actions by FreeRTOS.

3.1 Limitations

The following email client implementation does not deal with encryption. Almost all the actual SMTP servers use secure socket layer (SSL) to correctly sign-in and to exchange information over a secure communication protocol. Popular email service providers like Hotmail, Yahoo, or Google expect SSL. Files attachments to emails is work that must be considered for future developement.

At the time of testing this application note software, the following SMTP server is the only server that accepts regular requests without the SSL:

smtp113.plus.mail.mud.yahoo.com

Username and password used with the Yahoo email account are embedded in the source code and can not be valid during source code testing.

3.2 Principle of Operation

The software was developed for the MCF51CN128 reference design hardware to demonstrate low-cost and small board size. But, it can also be used in the Tower board.

This is a selection between either the M51CN128RD or V1TOWER C-macros inside m51cn128evb.h file.

```
/******************************************************************************/

/*Warning: only define one of them*/
#define M51CN128RD            /*pins moved to reference design hardware*/
//#define V1_TOWER              /*pins moved to reference design hardware*/
```

Figure 5. Code snippet for hardware change

4 Email Client Software

4.1 Software Architecture

Figure 6 shows how the email client is divided and what software blocks are used for its implementation.

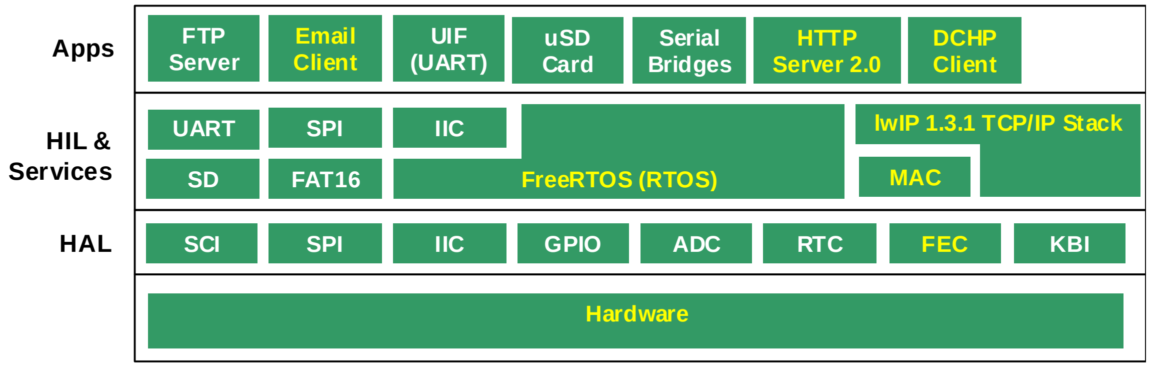


Figure 6. Software segmentation

4.2 Software Hierarchy

Figure 7 shows the files hierarchy.

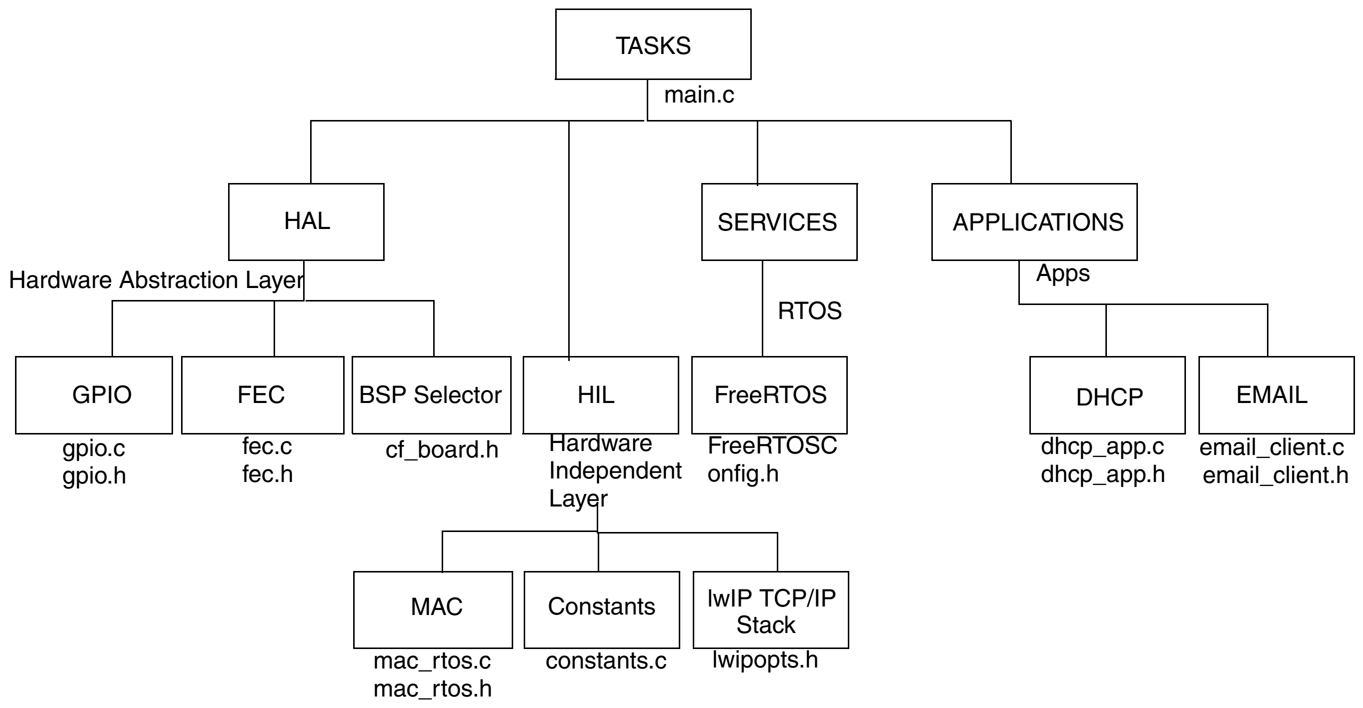


Figure 7. File implementation

Table 2 provides an explanation of each hierarchical file distribution.

Table 2. Software file description

Layer	File Name	Description
Main	dhcp_app.c	File that calls the email client function to send requested IP
HAL	gpio.c	Routines that use directly pins for the selected MCU
	gpio.h	Points all the modules to a specific pin for the selected MCU
	fec.c	Low level Init for FEC driver
	fec.h	Number and length of RX/TX buffers
	cf_board.h	HAL layer to use with this Serial Bridge
HIL	mac_rtos.c	MAC driver used by lwIP TCP/IP stack
	mac_rtos.h	MAC driver header
	constants.c	Structure containing all the default parameters after reset
	lwipopts.h	lwIP options to enable/disable services
	FreeRTOSConfig.h	FreeRTOS options to enable/disable services
Applications	email_client.c	Requests and error messages from SMTP client/server communication
	email_client.h	SMTP request strings

4.2.1 Hardware Abstraction Layer (HAL) Implementation

The HAL is defined as the collection of software components that gives direct access to the hardware resources, such as peripherals, configuration registers, optimized assembler routines (with their appropriate prototypes), pre-compiled object code libraries, or any other hardware-dependent resource, through the HAL-HW interface.

4.2.2 Fast Ethernet Controller (FEC) Handling

Due to a reduced memory footprint, a single Tx buffer is used to transmit data and two Rx buffers are used to receive information. For details on Fast Ethernet Controller (FEC) handling, refer to application note *Serial-to-Ethernet Bridge Using MCF51CN Family and FreeRTOS* (document AN3906).

4.2.3 Hardware Independent Layer (HIL) Implementation

To maintain hardware independence, software components that belong to this layer can access the controller's resources only by means of HIL components. Therefore, they refrain from directly accessing the resources of the controller on which they are running. This feature allows for components from this and the above sitting layers to run on different controllers without further change.

For more details about HIL blocks, refer to application note *Serial-to-Ethernet Bridge Using MCF51CN Family and FreeRTOS* (document AN3906).

4.3 Email Application Implementation

Figure 8 represents the communication between the client to the server to send an email using SMTP without authentication.

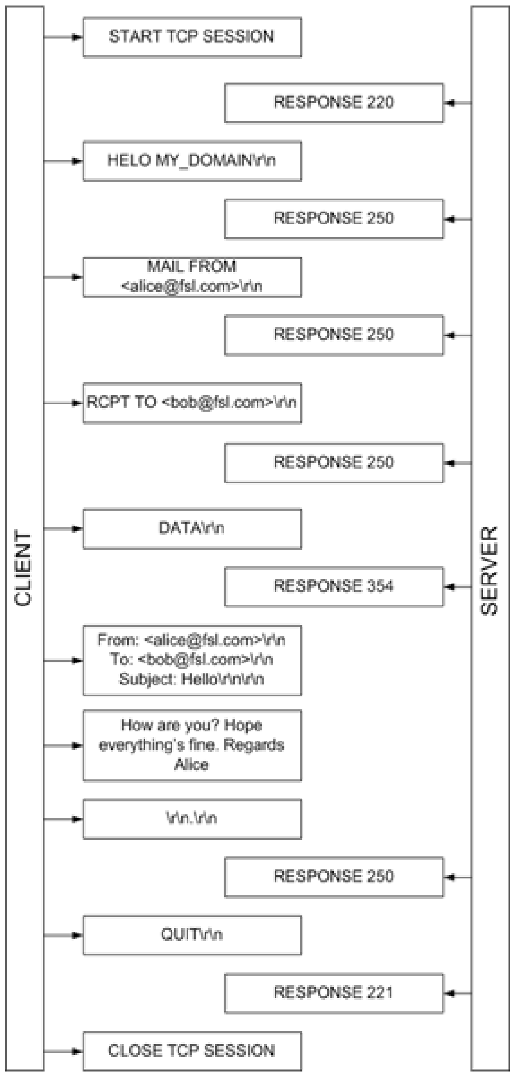

CLIENT
SERVER
START TCP SESSION
RESPONSE 220
HELO MY_DOMAIN\r\n
RESPONSE 250
MAIL FROM <alice@fsl.com>\r\n
RESPONSE 250
RCPT TO <bob@fsl.com>\r\n
RESPONSE 250
DATA\r\n
RESPONSE 354
From: <alice@fsl.com>\r\n To: <bob@fsl.com>\r\n Subject: Hello\r\n\r\n
How are you? Hope everything's fine. Regards Alice
\r\n.\r\n
RESPONSE 250
QUIT\r\n
RESPONSE 221
CLOSE TCP SESSION

Figure 8. SMTP without authentication required

Figure 9 represents the communication between the client to the server, to send an email using SMTP with authentication.

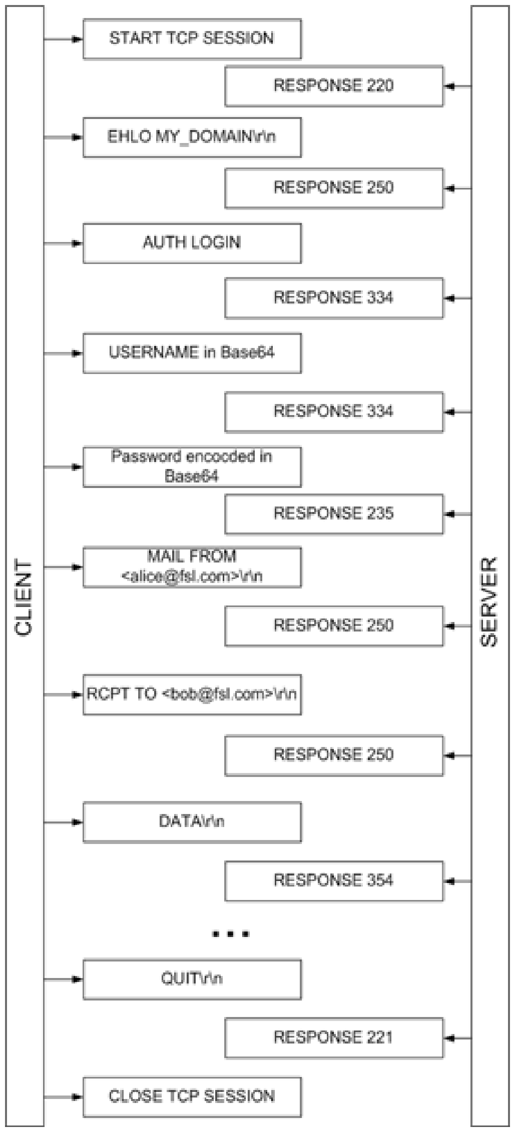


Figure 9. SMTP with authentication required

5 Email Client API

The following steps must be performed to send an email:

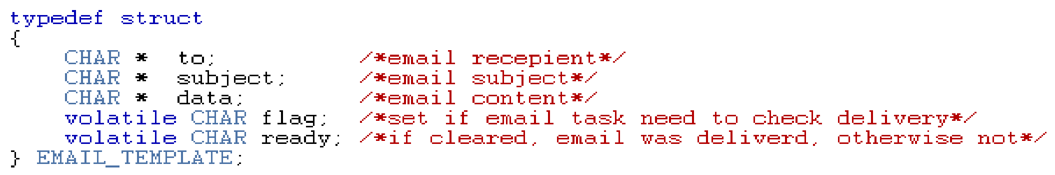
1. Add the following line of software to the C-file which needs email services:

```
#include "email_client.h"
```

2. Create an email C-structure type, EMAIL_TEMPLATE.
3. Fill the TO, SUBJECT, and DATA structure elements with desired information.
4. Call `Email_init()` function once.
5. Call `Email_send()` function using the created variable of type EMAIL_TEMPLATE.
6. Next two steps are optional. Number seven in case a new email needs to be sent. Number eight to check if it was correctly sent.

7. Wait until C-structure element, named flag, is equal to zero. If not, application must wait 100 RTOS ticks to allow other tasks to run.
8. Optional — If it was successfully sent, check C-structure element named ready. If the variable ready is equal to zero, then email was correctly sent, otherwise not.
9. For every new email repeat from step number 5

EMAIL_TEMPLATE C-structure is defined as follows. Only first three struct's elements must be filled.



```
typedef struct
{
    CHAR *  to;             /*email recepient*/
    CHAR *  subject;        /*email subject*/
    CHAR *  data;           /*email content*/
    volatile CHAR flag;     /*set if email task need to check delivery*/
    volatile CHAR ready;    /*if cleared, email was deliverd, otherwise not*/
} EMAIL_TEMPLATE;
```

Figure 10. Email C-structure

Details about the email public functions are provided:

Syntax

```
void
Email_init ( void )
```

Description — Starts email service (call it once)

```
/**
 * Start email service by creating mutex
* @param none
* @return none
*/
```

Syntax

```
void *
Email_send ( EMAIL_TEMPLATE *info )
```

Description — Sends email using parameter C-structure information

```
/**
* Send an email using SMTP protocol to specified variables.
* flag element cleared means attempt to send is over
* ready element cleared means successful attempt, otherwise error
* @param info email info to be used for SMTP communication
* @return pointer to the created task
* NULL if no task could be created
*/
```

6 Customization

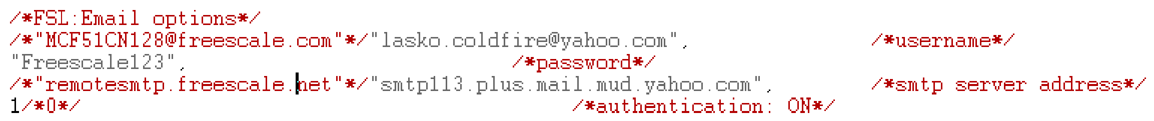
For customization, the following files must be modified for a change in software or hardware:

File Name	Description
cf_board.h	Used to point to a new BSP, new HAL software drivers
lwipopts.h	lwip configuration file. Enable/disable TCP/IP options
gpio.c/gpio.h	Change GPIO used for all modules in MCU
FreeRTOSConfig.h	FreeRTOS user configuration file. Enable and disable features
static_web_pages.c	Contains web pages
http_server.h	Present error messages
email_client.c	Basic SMTP communication to send an email
email_client.h	Present email client messages to the email server

For a standard email client, the following files are required to be changed:

- constants.c — If a different username, password, SMTP server address, or use of authentication is required, change it on the C-struct ROM_OPTIONS called “params”.

  C-struct related to email details are as follows:



```
/*FSL:Email options*/
/*"MCF51CN128@freescale.com"*/"lasko.coldfire@yahoo.com",            /*username*/
"Freescale123",                            /*password*/
/*"remotesmtp.freescale.net"*/"smtp113.plus.mail.mud.yahoo.com",     /*smtp server address*/
1/*0*/                                          /*authentication: ON*/
```

Figure 11. constants.c with email settings

- email_client.c — If more SMTP header options are needed to implement, add them to the vEmailClient() task function. Options like—carbon copy (CC), blind carbon copy (BCC), and Content-Type (MIME type) can be added using the same template for the other implemented parameters like—“MAIL from” and “RECIPIENT from”.
- email_client.h — If further requests are needed besides: HELO, EHLO or AUTH, add them to this file and then call them from email_client.c.

7 Conclusion

This document described how an email client can be implemented in the MCF51CN128 to send e-mails. The application note document and software demonstrate how to send an email to the Internet using an external email server (Yahoo email account) and basic authentication. The email API presented is able to send emails using lwIP and FreeRTOS, and hide SMTP details, therefore integration to existing lwIP application is clearer and faster than starting from scratch.

8 Considerations and References

Find the newest software updates and configuration files for the MCF51CN128 on the Freescale Semiconductor home page, www.freescale.com.

- MCF51CN128 Reference Design and Tower System were the hardware used to test the AN3930SW.
- For more information on FEC, refer to *MCF51CN128 Reference Manual* (document MCF51CN128RM) at www.freescale.com.
- To learn more about the Tower System, refer to www.freescale.com/tower.
- To learn more about the MCF51CN128 Reference Design details, refer to MCF51CN128 Product Summary Page.
- The BridgeSoftwareDemo software was developed and tested with CodeWarrior for ColdFire V6.2.1.
- Download the source files for AN3930SW.zip from www.freescale.com.
- For more information regarding software or hardware, refer to www.freescale.com/support.

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

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)
www.freescale.com/support

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3930
Rev. 0
09/2009