NEC

User's Manual

TK-850/SG2+UZ

ZigBee™-ready Wireless Networks to Ethernet Gateway

V850ES 32-bit Microcontroller Evaluation Board

Application Development Manual
C-NET API for HTTP, SMTP, POP3, and DHCP

Document No. U19108EE1V0UM00

Date published January 2008

Printed in Germany

Legal Notes

- **The information in this document is current as of November, 2007. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**

- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.

- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.

- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.

- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.

- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

- The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime

systems, safety equipment and medical equipment (not specifically designed for life support).
"Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)
(1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives anddistributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and otherlegal issues may also vary from country to country.

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668, Japan
Tel: 044 4355111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554,
U.S.A.
Tel: 408 5886000
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211 65030
http://www.eu.necel.com/

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908 691133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01 30675800

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 6387200

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02 667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven,
The Netherlands
Tel: 040 2654010

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27
ZhiChunLu Haidian District,
Beijing 100083, P.R.China
Tel: 010 82351155
http://www.cn.necel.com/

NEC Electronics Shanghai Ltd.
Room 2511-2512, Bank of China
Tower,
200 Yincheng Road Central,
Pudong New Area,
Shanghai 200120, P.R. China
Tel: 021 58885400
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
12/F., Cityplaza 4,
12 Taikoo Wan Road, Hong Kong
Tel: 2886 9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R.O.C.
Tel: 02 27192377

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253 8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku, Seoul,
135-080, Korea Tel: 02-558-3737
http://www.kr.necel.com/

Preface

Copyright

C-NET is based on lwIP (Light Weight TCP/IP).
lwIP can be distributed with Modified BSD License.

This file is part of the lwIP TCP/IP stack.
Author: Adam Dunkels <adam@sics.se>

http://savannah.nongnu.org/projects/lwip/

Table of Contents

Chapter 1 Introduction

C-NET enables the TK-850/SG2+UZ board to offer TCP/IP, SMTP, POP3, and HTTP functions. This manual introduces you how to utilize these functions to develop your application software. This manual does not provide with any information on IEEE 802.15.4 MAC stack and ZigBee, because the wireless protocol is independent from C-NET.

Chapter 2 Preparations

First of all, the development environment is prepared in your PC.

2.1 Installation of software

When you installed a sample program for the TK-850/SG2+UZ board at C:/TK850/ZBV850. In this directory, you will find a sub-directory, named /CNET. If you do not like to destroy the sample program, you may copy the sub-directory of /CNET to your favorite location. Now, the /CNET directory in the new location is the folder you are going to create your own application from now on.

```
/cnet..............User program folder
     /bin..........Tools to registor web files
     /fs...........User html files
          /cgi.....User cgi files
          /gif.....User gif files
```

Chapter 3 How to create your application

3.1 Overview

Application Programming Interface, API, is available for your development.

We have two kinds of API:

A. Network API
B. Application Layer API

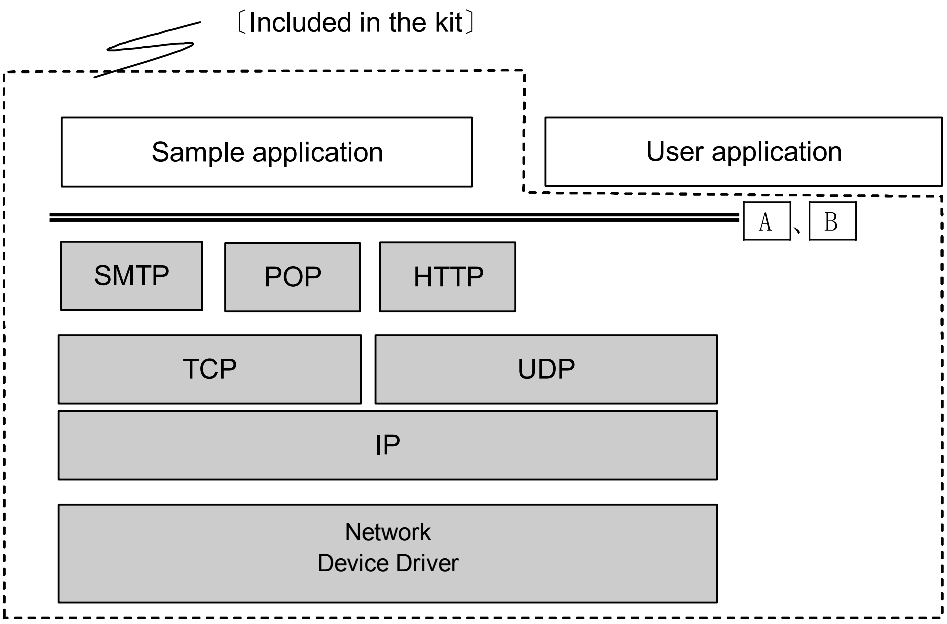

Figure 3-1 Layer structure

3.1.1 Network API

Network API supports network configuration.

3.1.2 Application layer API

Application Layer API supports HTTP, POP3, SMTP, and DHCP.

- HTTP

C-NET supports Hyper text transfer between Web server and Web client. Therefore, C-NET enables for Web server to recognize and execute requests generated in Web clients

- POP3 and SMTP

C-NET supports transmission and reception of e-mail

- DHCP

C-NET can negotiate with Dynamic Host Configuration Protocol, DHCP, to acquire/extend IP address, subnet mask, and gateway address.

3.2 Steps to create your applications

A flow chart to create user applications:

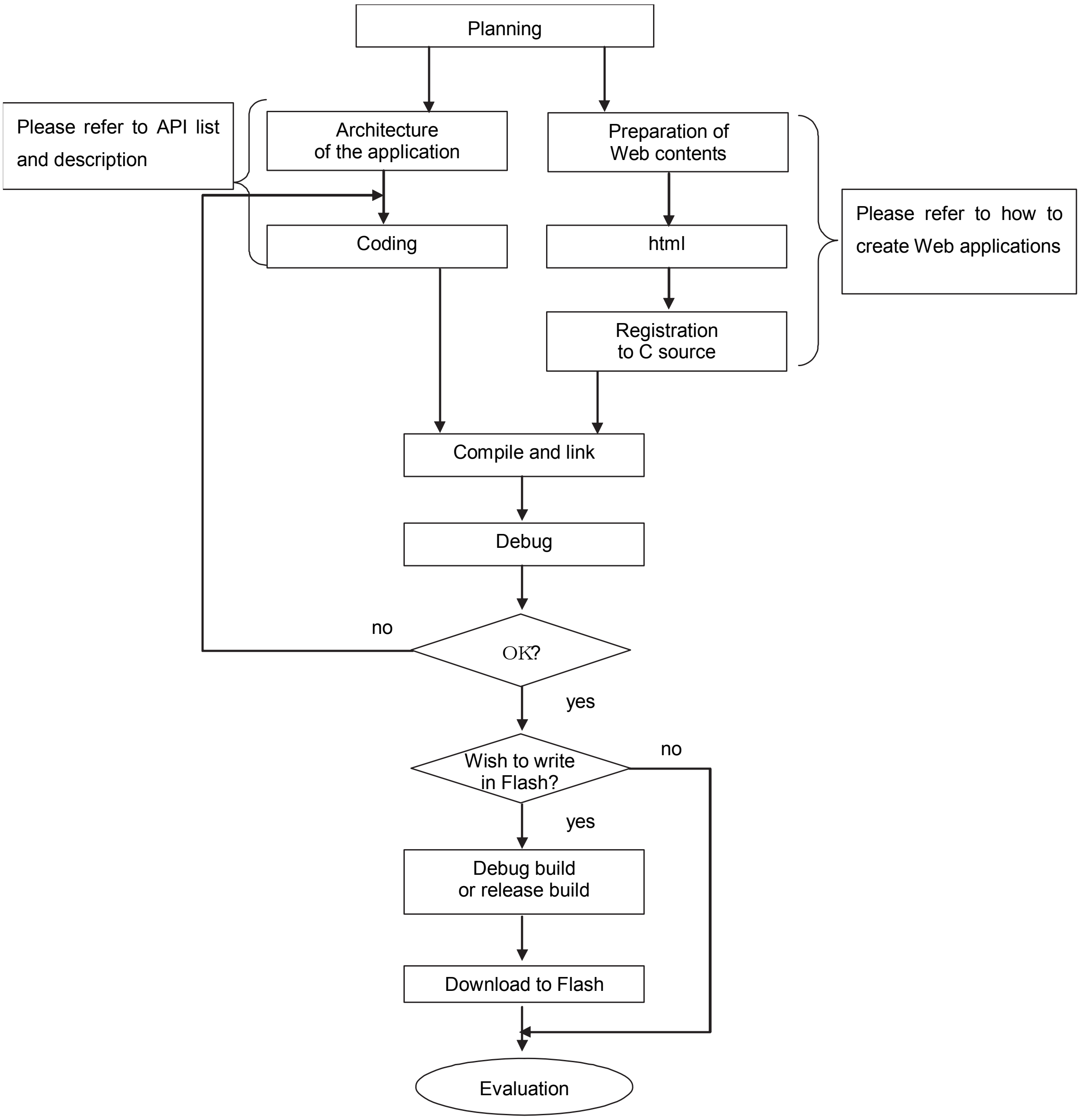

Figure 3-2 Creating user applications

3.2.1 API list

Table 3-1 Network API

Name of API	Function
cnet_init	Initialization
netif_add	Add Network Information
netif_remove	Remove Network Information
netif_set_default	Set Default Network Information
netif_set_addr	Modify Network Information
cnet_getipaddr	Get IP address
cnet_getnetmask	Get subnet mask
cnet_getgwaddr	Get gateway address
inet_addr	Convert IP address from dot format to byte order format
inet_ntoa	Convert IP address to dot format from byte order format
htonl	Convert 32-bit data from host byte order to network byte order format
htons	Convert 16-bit data from host byte order to network byte order format
ntohl	Convert 32-bit data to host byte order from network byte order format
ntohs	Convert 16-bit data to host byte order from network byte order format

Table 3-2 Application Layer API (SMTP, POP3, and HTTP)

Name of API	Function
SMTP	
cnet_smtpinit	Initiate session with SMTP server
cnet_smtpend	Terminate session with SMTP server
cnet_smtpdatafirst	Initiate transmission of mail data
cnet_smtpdatanext	Transmission of mail text
cnet_smtpdataend	Terminate transmission of mail data
POP3	
cnet_pop3init	Initiate session with POP3 server where mail is spooled
cnet_pop3end	Terminate session with POP3 server where mail is spooled
cnet_pop3auth	Send data for authentication to POP3 server
cnet_pop3stat	Ask POP3 server to check mail status for the user
cnet_pop3getsize	Ask POP3 server to check the mail size
cnet_pop3getmsgfirst	Initiate reception of mail data
cnet_pop3getmsgnext	Continue reception of mail data
cnet_pop3getmsgend	Declare termination of mail data reception
cnet_pop3delete	Ask POP3 server to put delete mark on a mail
cnet_pop3reset	Ask POP3 server to clear the delete mark
cnet_pop3getuid	Ask POP3 server to check Unique-ID of the mail
HTTP	
cnet_httpdinit	Initialize transfer function of Hyper text

Name of API	Function
cnet_httpdsend	Send data to Web client
DHCP Client	
cnet_dhcpstart	Ask DHCP server to assign network information automatically
cnet_dhcpstop	Clear the assigned network information

3.3 How to program user application

3.3.1 How to use network API

Structure is used to call network API

in_addr Structure(cnet/ipv4/lwip/ip_addr.h)

```
struct in_addr {
  u32_t s_addr;   /* Network address */
};
```

Netif Structure(cnet/lwip/netif.h)

```
struct netif {
  struct netif *next;      /* Pointer for the next netifã*/
  struct ip_addr ip_addr;  /* Host IP address: Byte order format */
  struct ip_addr netmask;  /* Netmask: Byte order format */
  struct ip_addr gw;       /* Gateway address: Byte order format */
  err_t (* input)(struct pbuf *p, struct netif *inp);
                           /* TCP/IP function calld by network driver function */
  err_t (* output)(struct netif *netif, struct pbuf *p, struct ip_addr *ipaddr);
                           /*Transmission driver function on the network layer */
  err_t (* linkoutput)(struct netif *netif, struct pbuf *p);
                           /* Transmission driver function on the link layer */
  void (* linkproc)(struct netif *netif); /* Event execution function of network driver */
  void *state; /* Status */
#if LWIP_DHCP
    struct dhcp *dhcp; /* dhcp info is stored */
#endif
  unsigned char hwaddr_len; /* hardware address length */
  unsigned char hwaddr[NETIF_MAX_HWADDR_LEN]ï¼ /* Hardware address */
  u16_t mtu;    /* MTU */
  char name[2];
  u8_t num;
  u8_t flags;
};
```

Network interface information, netif, consists of the following members,

- next

  Pointer to connect network interface information, netif, is stored

- ip_addr

  IP address, network byte order format, which is specified by parameter when “netif_add”, or “netif_set_addr” is issued, is stored

- netmask

  Subnet mask, network byte order format, which is specified by parameter when “netif_add”, or “netif_set_addr” is issued, is stored

- gw

  Gateway address, network byte order format, which is specified by parameter when “netif_add”, or “netif_set_addr” is issued, is stored

- input (struct pbuf *p , struct netif *inp)

  This is pre-occupied by system. Fixed value of tcpip_input is stored

- output (struct netif *netif , struct pbuf *p , struct ip_addr *ipaddr)

  Interface function, to transmit data by TCP/IP protocol stack, is stored

- linkoutput (struct netif *netif , struct pbuf *p)

  Interface function, to transmit ARP data by TCP/IP protocol stack, is stored

- linkproc (struct netif *netif)

  Pre-occupied by system

- state

  Status information, specified parameter when netif_add is issued, is stored

- dhcp

  DHCP information, acquired when cnet_dhcpstart is issued, is stored

- hwaddr_len

  Size of Mac address stored in hwaddr is stored in byte

- hwaddr

  Mac address of the target system is stored

- mtu

  Pre-occupied by system

- name

  Two bytes of ID of network interface is stored
  Network interface name in C-NET is a concatenation of the two strings stored at NAME and NUM

- num

  Network interface number is stored

- flags

  Status flags of network interface are stored

3.3.2 How to send/receive e-mail

You can find an example at /cnet/src/userap.c of the sample program

3.3.3 Caution

Please do not touch or modify any file in /cnet/lib and /cnet/include directories.

If it is modified, C-NET will not work.

3.4 How to create Web applications

3.4.1 Overview

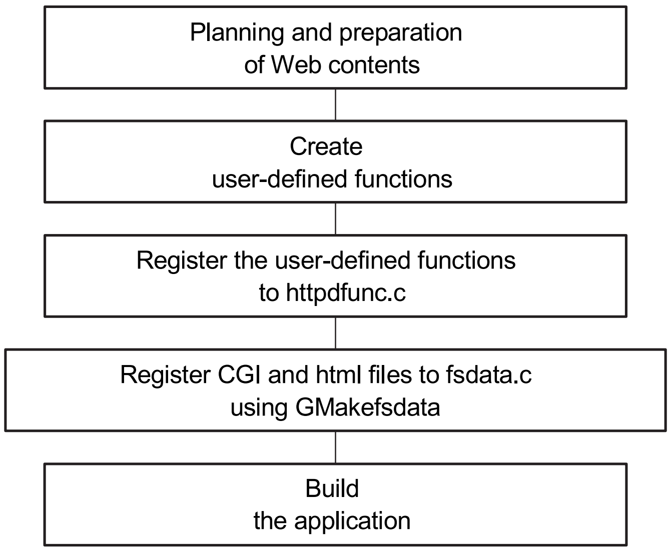

Figure 3-3 Creating web applications

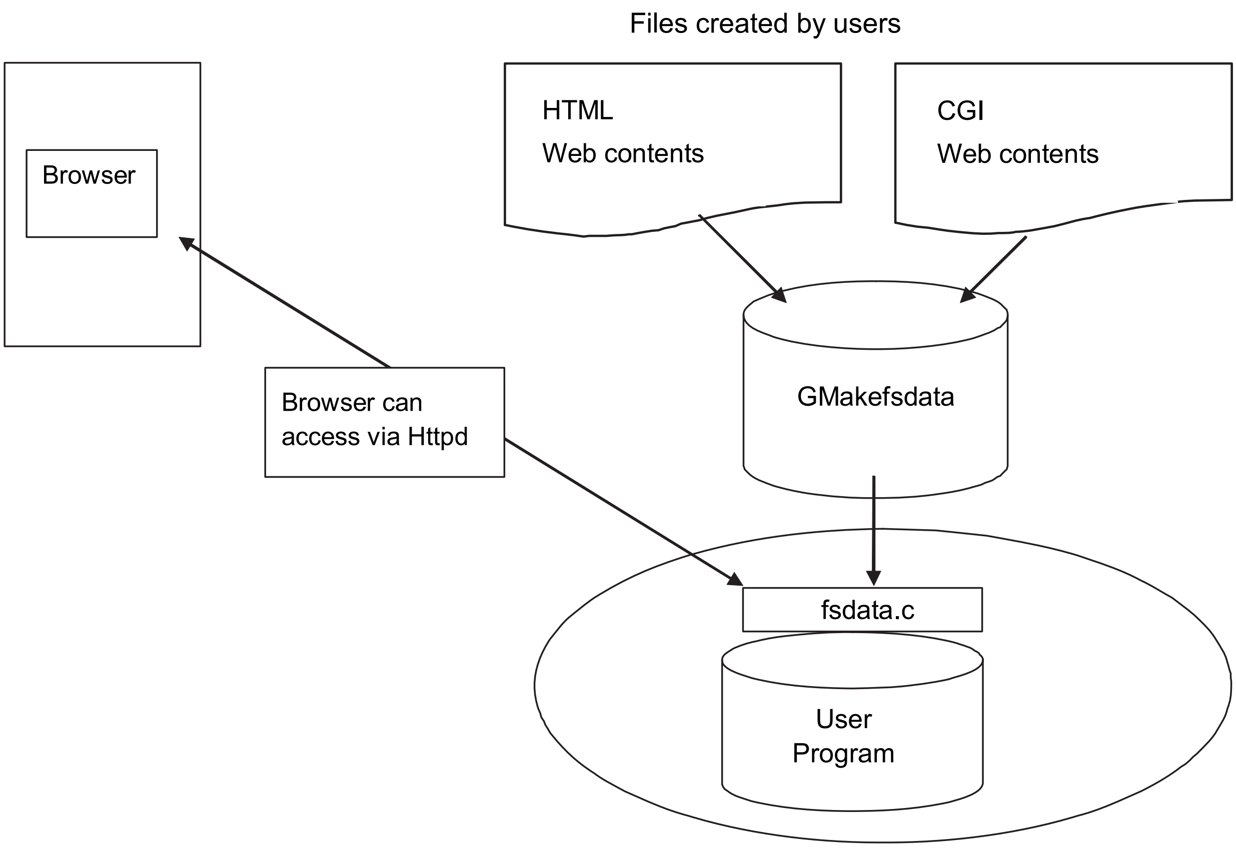

Figure 3-4 File structure of Web application

3.4.2 Note for CGI

CGI format

One line for one execution unit
The first character specifies an action

- i Output contents in the file specified after i
- t Output data specified after t
- c Execute the function, ID or parameters of which is specified after c
- * Send out 204 (No Contents)
- # Comment

.(Period)

End of HTML description (</BODY></HTML>\r\n)

Others - Nothing happens

3.4.3 Definition of method

(1) GET Method

[Function]

Request display of HTML, CGI, or image

[Rule]

If only “/” is specified, it displays index.html.
If the specified html or cgi is not found, it displays 404.html.
If the file under /cgi/ is specified, it assumes the specified file to be CGI.

[Example]

GET /index.html HTTP/1.0
GET /test.html?func1=param1 HTTP/1.0

(2) POST Method

[Function]

It passes parameters to CGI.

[Rule]

If the file under /cgi/ is not specified, it returns error 405.
If the specified cgi is not found, it displays 404.html.

3.4.4 How to define user functions

If you like to create your own function to execute tasks in CGI, you can define it by writing a definition of the function in httpd.h, and the mapping of function names and functions in the httpd_first structure. You can call up the function, if the function name is specified in CGI or data passed in the POST method.

(1) User-defined functions

[Function]
A user-defined function will be called up by specifying its function name in any of the following: a parameter of the GET method, data passed by the POST method, or the “c” (function execution) command of CGI. You can create user-defined functions in httpdsub.c or in any other file.

(2) Format of user-defined functions

User-defined functions have 3 parameters, and the number and content of parameters are fixed.

First parameter	int	ID for HTTP server to manage
Second parameter	const char*	Function ID
Third parameter	const char*	Specified parameter for CGI, or the received data for GET and POST method

Format of cnet_httpd_datacmn()

cnet_httpd_datacmn() is a function which is called up each time after the processing of data received in GET or POST method, and has one parameter. However, it is not called up, if no parameter is specified in GET method.

First parameter	int	ID for HTTP server to manage

You can find some examples of implementation at /cnet/src/httpdfunc.c.

(3) Registration of user-defined functions

This is an example to register a user-function in httpdfunc.c

```
    /* Declare the structure defined in cnet_httpd.h */
        typedef struct {
          const char  *fn;                                                /* Function ID */
          void       (*fi)(int id, const char *name, const char *value);    /* Function Pointer */
        } cnet_finfo;

=== httpdfunc.c ==========================================================

    /* Definition of the user function in httpdfunc.c and an example of httpd_flist */
        static void a(int id, const char *name, const char *value)
        {
            ~
        }
        static void b(int id, const char *name, const char *value)
        {
            ~
        }
        static void c(int id, const char *name, const char *value)
        {
            ~
        }
        static void func1(int id, const char *name, const char *value)
        {
            ~
        }

        const cnet_finfo  httpd_flist[] = {
            {"a", a},
            {"b", b},
            {"c", c},
            {"func1", func1},
            {NULL, NULL}
        };
======================================================================
```

User-defined function names and function pointers are specified in the httpd_first structure. The end of registration is recognized by specifying a NULL pair as the function name and function pointer. Therefore, please write a NULL,NULL at the end.

[Example]

1. Example of POST method:
   POST /cgi/test.cgi HTTP/1.0
   Content-Length: 12
   func1=param1
2. Example for CGI:
   c func1 param1
3. Function call up:
   func1(pcd,hs,“func1”, “param1”);

3.4.5 How to use the Web file registration tool (makefsdata command)

Files for http server need to be registered in fsdata.c using the following makefsdata command.
The file registration procedure is as follows:

1. Create the fs (fixed name) directory under the \cnet\src directory of the project.
   -> In the operation environment of the preloaded sample, \fs and \cgi are already created.

```
\cnet
   \fs
      (html file)
       \cgi
          (cgi file)
       \gif
          (gif file)
   \src
       fsdata.c           This is the data file of your new contents
```

2. Create html, cgi, and image files, which make up web pages, under the created fs directory. The directory tree can be created arbitrarily, as loon as the cgi file is created under fs\cgi.
3. Once you have created all files under the fs directory, open a command prompt and run Gmakefsdata. You can find Gmakefsdata.exe at /cnet/bin/.
4. You will see the following window. Now, you specify the source of the web data and the destination of the fs directory.

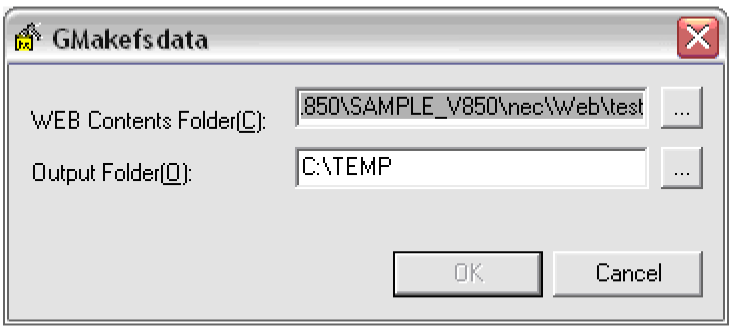

Figure 3-5 Window of GMakefsdata.exe

5. You will find GMakefsdata at the destination.

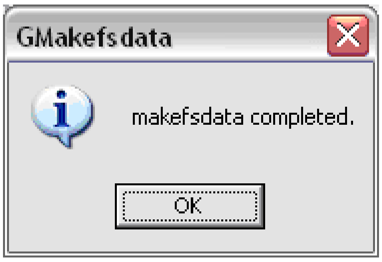

Figure 3-6 Gmakefsdata has been completed

6. Now you can make rebuild.

Chapter 4 API Specifications

4.1 Network API

4.1.1 cnet_init function

[Scope]	Initialization of the network
[Format in C]	void cnet_init (void);
[Parameters]	None
[Functions]	The network information is initialized with the default value stored in Flash EEPROM
[Return values]	None

4.1.2 cnet_add function

[Scope]	Add network information
[Format in C]	struct netif *netif_add (struct ip_addr *ip_addr , struct ip_addr *netmask , struct ip_addr *gw , void *state , err_t (*init) (struct netif *netif) , err_t (*input) (struct pbuf , struct netif *netif)) ;

[Parameters]

I/O	Parameters	Functions
I	struct ip_addr *ip_addr ;	Pointer to the assigned IP adress
I	struct ip_addr *netmask ;	Pointer to the netmask
I	struct ip_addr *gw ;	Pointer to the gateway address
I	Void *state ;	Pointer to the status info
I	err_t (*init) (struct netif *netif) ;	Pointer to the physical network driver
I	err_t (*input) (struct pbuf , struct netif *netif) ;	Pre-occupied by system tcpip_input is fixed

[Functions]

IP address, netmask, gateway address, and status are registered on C-NET.
At the same time, initiation of the physical network driver is completed.
Notes:
(1) After this API is executed, if you like to utilize cnet_dhcpstart, please set 0x0 in ip_addr, netmask, and gw.
(2) Do not set any conflicting parameter in the physical network driver.
(3) This API is valid, only after cnet_init.

[Return values]

Value	Meaning
Any value	Normally completed with the pointer to the network interface information
NULL	Abnormal

4.1.3 netif_remove

[Scope] Remove network information
[Format in C] void netif_remove (struct netif *netif) ;

[Parameters]

I/O	Parameters	Functions
I	struct netif *netif ;	Pointer to network interface information: netif

[Functions] This API erases the network information, such as IP address, subnet mask, and gateway address.
Notes:
(1) The network interface information shall match the return value of the previous netif_add.
(2) Prior to this API, netif_add has to be executed.
[Returns] None

4.1.4 netif_set_default

[Scope] Registration of default network information
[Format in C] void netif_set_default (struct netif *netif) ;

[Parameters]

I/O	Parameter	Function
I	struct netif *netif ;	Pointer to network information: netif

[Functions] Network information sefined in netif is registered as default
Notes:
(1) The network information registered shall be compatible with the return values in netif_add
(2) Prior to this API, “netif_add” has to be ececuted.
[Return values] None

4.1.5 netif_set_addr

[Scope] Modification of network information
[Format in C] void netif_set_addr (struct netif *netif , struct ip_addr *ipaddr , struct ip_addr *netmask, struct ip_addr *gw);

[Parameters]

I/O	Parameter	Function
I	struct netif *netif ;	Pointer to network information: netif
I	struct ip_addr *ipaddr ;	Pointer to ip_addr, where the modified IP address is stored
I	struct ip_addr *netmask ;	Pointer to ip_addr, where the modified subnet mask is stored
I	struct ip_addr *gw ;	Pointer to ip_address, where the modified gateway address is stored

[Functions] Modify the network information registered by netif_add to the IP address specified by ipaddr, the subnet mask specified by netmask, and the gateway address by gw.
Notes:
(1) The network interface information shall match the return value of the previous netif_add.
(2) Prior to this API, netif_add has to be executed.

[Return values] None

4.1.6 cnet_getipaddr

[Scope] Get IP address
[Format in C] int cnet_getipaddr (struct netif *netif) ;

[Parameters]

I/O	Parameter	Function
I	struct netif *netif ;	Pointer to network information: netif

[Functions] Read out IP address, which has been defined by netif_add, or assigned after cnet_dhcpstart. The returned IP address is in network byte order format.
Notes:
(1) The network interface information shall match the return value of the previous netif_add.
(2) Prior to this API, netif_add, or assignment of network information initiated by cnet_dhcpstart has to be executed.

[Returns]

Value	Meaning
Value	Normally completed with IP address in network byte order format

4.1.7 cnet_getnetmask

[Scope] Get subnet mask
[Format in C] int cnet_getnetmask (struct netif *netif) ;

[Parameters]

I/O	Parameter	Function
I	struct netif *netif ;	Pointer to network information: netif

[Functions]

Read out subnet mask, which has been defined by netif_add, or assigned after cnet_dhcpstart. The returned subnet mask is in network byte order format.
Notes:
(1) The network interface information shall match the return value of the previous netif_add.
(2) Prior to this API, netif_add, or assignment of network information initiated by cnet_dhcpstart has to be executed.

[Returns]

Value	Meaning
Value	Normally completed with subnet mask in network byte order format

4.1.8 cnet_getgwaddr

[Scope] Get gateway address
[Format in C] int cnet_getgwaddr (struct netif *netif) ;

[Parameters]

I/O	Parameter	Function
I	struct netif *netif ;	Pointer to network information: netif

[Functions]

Read out gateway address, which has been defined by netif_add, or assigned after cnet_dhcpstart. The returned gateway address is in network byte order format.
Notes:
(1) The network interface information shall match the return value of the previous netif_add.
(2) Prior to this API, netif_add, or assignment of network information initiated by cnet_dhcpstart has to be executed.

[Returns]

Value	Meaning
Value	Normally completed with gateway address in network byte order format

4.1.9 inet_addr

[Scope] Convert IP address from dot format to byte order format
[Format in C] u32_t inet_addr (const char *cp) ;

[Parameters]

I/O	Parameter	Function
I	Const char *cp ;	Pointer to the memory where IP address with dot format is stored

[Functions] Convert IP address from dot format to byte order format
Return IP address in byte order format.
(dot format is something like 202.247.5.136)

[Return values]

Value	Meaning
Value	IP address with network byte order format

4.1.10 inet_ntoa

[Scope] Convert IP address to dot order format from byte format

[Format in C] u8_t *inet_ntoa (u32_t addr) ;

[Parameters]

I/O	Parameter	Function
I	U32_t addr ;	IP address with network byte order format

[Functions] Convert IP address from byte format to dot order format
Return IP address in dot order format.
dot format is something like 202.247.5.136.

[Return values]

Value	Meaning
Value	IP address with dot order format

4.1.11 htonl

[Scope] Convert 32-bit data from host byte order to network byte order format

[Format in C] u32_t htonl (u32_t n) ;

[Parameters]

I/O	Parameter	Function
I	u32_t n ;	32-bit data in host byte order format

[Functions] Convert 32-bit data in n from host byte order to network byte order format
32-bit data are returned in network byte order format

[Return values]

Value	Meaning
Value	32-bit data are returned in network byte order format

4.1.12 htons

[Scope] Convert 16-bit data from host byte order to network byte order format

[Format in C] u16_t htons (u16_t n) ;

[Parameters]

I/O	Parameter	Function
I	u16_t n ;	16-bit data in host byte order format

[Functions] Convert 16-bit data in n from host byte order to network byte order format
16-bit data are returned in network byte order format

[Return values]

Value	Meaning
Value	16-bit data are returned in network byte order format

4.1.13 ntohl

[Scope] Convert 32-bit data from network byte order to host byte order format

[Format in C] u32_t ntohl (u32_t n) ;

[Parameters]

I/O	Parameter	Function
I	u32_t n ;	32 -bit data in network byte order format

[Functions] Convert 32-bit data in n from network byte order to host byte order format
32-bit data are returned in host byte order format

[Return values]

Value	Meaning
Value	32-bit data in host byte order format

4.1.14 ntohs

[Scope] Convert 16-bit data from network byte order to host byte order format

[Format in C] u16_t ntohs (u16_t n) ;

[Parameters]

I/O	Parameter	Function
I	u16_t n ;	16-bit data in network byte order format

[Functions] Convert 16-bit data in n from network byte order to host byte order format
16-bit data are returned in host byte order format

[Return values]

Value	Meaning
Value	16-bit data are returned in host byte order format

4.2 Application layer API: HTTP, SMTP, POP3 and DHCP

4.2.1 cnet_smtpinit

[Scope] Initiate session with SMTP server

[Format in C] int cnet_smtpinit (unsigned int addr , unsigned short port , const char *domain) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int addr ;	IP address in network byte order format
I	unsigned short port ;	Port number in host byte order format
I	Const char *domain ;	Pointer to the memory domain name is stored

[Functions] Initiate session with SMTP server by sending HELO command and domain name to the port, in the SMTP mail server, specified by addr
Notes:
(1) If 0x0 is found in the port number, it is interpreted that the port number 25 is specified.
(2) This API is valid, only if netif_set_default has assigned the default value, or automatic assignment has been completed based on cnet_dhcpstart.

[Return values]

Macro	Value	Meanings
SMTP_ERROR_NORMAL	0x0	No error
SMTP_ERROR_NOBUF	0xffffff80	Not enough working buffer
SMTP_ERROR_CONNECT	0xfffffff8	Failed to network connection
SMTP_ERROR_STATE	0xfffffffe	Wrong sequence of API
SMTP_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.2 cnet_smtpend

[Scope] Terminate session with SMTP server
[Format in C] int cnet_smtpend (unsigned char mode) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned char mode ;	Mode 0: Normal termination Mode 1: Forced termination

[Functions]

Terminate session with SMTP mail server, which is specified by cnet_smtpinit.
There are two modes to terminate.
[Mode 0]
Terminate session after receiving response by SMTP server to QUIT command
[Mode 1]
Terminate session immediately without receiving response by SMTP server to QUIT command
Note:
Prior to this API, session with the SMTP server has to be initiated with cnet_smtpinit.

[Return values]

Macro	Value	Meanings
SMTP_ERROR_NORMAL	0x0	No error
SMTP_ERROR_READ	0xffffffe0	Timeout to wait for a response by SMTP server
SMTP_ERROR_DISCONNECT	0xfffffff0	Disconnected with SMTP server
SMTP_ERROR_REPLY	0xfffffffc	Response error by SMTP server to QUIT command
SMTP_ERROR_STATE	0xfffffffe	Wrong sequence of API

4.2.3 cnet_smtpdatafirst

[Scope] Initiate transmission of mail data
[Format in C] int cnet_smtpdatafirst (const char *sendaddr , const char *recvaddr , const char *databuf , unsigned short datalen) ;

[Parameters]

I/O	Parameters	Functions
I	Const char *sendaddr ;	Pointer to the memory where sender's mail address is stored
I	Const char *recvaddr ;	Pointer to the memory where receiver's mail address is stored
I	Const char *databuf ;	Pointer to the memory where mail text is stored
I	unsigned short datalen ;	Data size of the mail text specified by databuf in bytes

[Functions]

Send out MAIL command, RCPT command, DATA command, and mail data to SMTP server, which is specified in cnet_smtpinit. Mail data comprises of sender's mail address, which is specified in sendaddr, receiver's mail address, which is specified in receiver's mail address, and mail text, which is specified in databuf and datalen.
Notes:
(1) C-NET recognizes those information available between this API and net_smtpdataend as one e-mail.
(2) This API is valid, only after cnet_smtpinit has initiated a session with SMTP server.

[Return values]

Macro	Value	Meanings
SMTP_ERROR_NORMAL	0x0	No error
SMTP_ERROR_READ	0xffffffe0	Timeout to wait for a response by SMTP server
SMTP_ERROR_DISCONNECT	0xfffffff0	Disconnected by SMTP server
SMTP_ERROR_REPLY	0xfffffffc	Response error by SMTP server to MAIL, RCPT, or DATA commands
SMTP_ERROR_STATE	0xfffffffe	Wrong sequence of API
SMTP_ERROR_PARAMETER	0xffffffff	Parameter error

4.2.4 cnet_smtpdatanext

[Scope] Transmission of mail text

[Format in C] int cnet_smtpdatanext (const char *databuf , unsigned short datalen) ;

[Parameters]

I/O	Parameters	Functions
I	Const char *databuf ;	Pointer to the mail text
I	unsigned short datalen ;	Size of the mail text specified in databuf in bytes

[Functions]

Transmit the mail text which is specified in databuf and datalen.
This API follows cnet_smtpdatafirst.

[Return values]

Macro	Value	Meanings
SMTP_ERROR_NORMAL	0x0	No error
SMTP_ERROR_READ	0xffffffe0	Timeout of SMTP server response
SMTP_ERROR_DISCONNECT	0xfffffff0	Disconnected network by SMTP server
SMTP_ERROR_STATE	0xfffffffe	Wrong API sequence
SMTP_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.5 cnet_smtpdataend

[Scope] Terminate transmission of mail data
[Format in C] int cnet_smtpdataend (void) ;
[Parameters] None
[Functions] Terminate transmission of mail data by sending out <CR> and <LF>.
cnet_smtpdatafirst is mandatory, prior to this API.

[Return values]

Macro	Value	Meanings
SMTP_ERROR_NORMAL	0x0	No error
SMTP_ERROR_READ	0xffffffe0	Timeout of SMTP server response
SMTP_ERROR_DISCONNECT	0xfffffff0	Disconnected network by SMTP server
SMTP_ERROR_REPLY	0xfffffffc	Response error in SMTP server to <CR> and <LF>
SMTP_ERROR_STATE	0xfffffffe	Wrong API sequence

4.2.6 cnet_pop3init

[Scope] Initiate session with POP3 server where mail is spooled
[Format in C] int cnet_pop3init (unsigned int addr , unsigned short port) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int addr ;	IP address in network byte order format
I	unsigned short port ;	Port number in host byte order format

[Functions] Initiate session with the port of POP3 server where mail is spooled
Notes:
(1) If the port number is 0x0, the port number is assumed as 110.
(2) This API shall be issued after netif_set_default, or cnet_dhcpstart

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_NOBUF	0xffffff80	Not enough buffer space
POP3_ERROR_CONNECT	0xfffffff8	Failed to establish connection
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.7 cnet_pop3end

[Scope] Terminate session with POP3 server where mail is spooled

[Format in C] int cnet_pop3end (unsigned char mode) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned char mode ;	Mode 0: Normal termination Mode 1: Forced termination

[Functions] Terminate session with POP3 server where mail is spooled
[Mode 0]
Terminate after receiving a POP3 server response to QUIT command
[Mode 1]
Terminate immediately without receiving any response from POP3 server
Note:
cnet_pop3init is assumed, prior to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	Response timeout of POP3 server
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to QUIT command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence

4.2.8 cnet_pop3auth

[Scope] Send data for authentication to POP3 server

[Format in C] int cnet_pop3auth (const char *user , const char *pass) ;

[Parameters]

I/O	Parameters	Functions
I	Const char *user ;	Pointer to user ID
I	Const char *pass ;	Pointer to password

[Functions]

Send USER command, PASS command, and authentication data to the POP3 server.
Authentication data comprises of user name, and password.
Authentication is executed by the POP3 server
Note:
cnet_pop3init has to be completed prior to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	Response timeout of POP3 server
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to USER or PASS command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.9 cnet_pop3stat

[Scope] Ask POP3 server to check mail status for the user

[Format in C] int cnet_pop3stat (unsigned int *number , unsigned int *size) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int *number ;	Pointer to number of spooled mails
I	unsigned int *size ;	Pointer to the size of the spooled mails in byte

[Functions]

Send STAT command to the POP3 server.
This API is applied to the mails for the user specified in cnet_pop3auth.
Number of the spooled mail for the user is stored in number, and the total size of the mails for the user is stored in size.
Notes:
(1) Spooled mails mean those mails undelivered to the user, tentatively stored in the POP3 server.
(2) cnet_pop3auth has to be precedent to this API

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to STAT command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.10 cnet_pop3getsize

[Scope] Ask POP3 server to check the mail size.

[Format in C] int cnet_pop3getsize (unsigned int msgno , unsigned int *size) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int msgno ;	NUMBER of a mail
O	unsigned int *size ;	Pointer to the mail size of the mail specified by msgno

[Functions] Send LIST command and NUMBER of a mail to the POP3 server.
If the mail is being spooled in the POP3 server, the size of the mail is stored at size in bytes.
Note:
cnet_pop3auth has to be precedent to this API

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to LIST command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.11 cnet_pop3getmsgfirst

[Scope] Initiate reception of mail data.

[Format in C] int cnet_pop3getmsgfirst (unsigned int msgno , unsigned int maxline , char *msgbuf , unsigned short msglen , unsigned char mode) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int msgno ;	NUMBER of a mail
I	unsigned int maxline ;	Acceptable maximum lines of the mail. If zero is specified, all lines will be subject to processing.
O	char *msgbuf ;	Pointer to the memory where the mail is stored
I	unsigned short msglen ;	Size of the mail in bytes
I	unsigned char mode ;	Selection of receive mode POP3_MODE_TEXT: Received line by line POP3_MODE_TRUNC: Received line by line POP3_MODE_BINARY: Received upto the size

[Functions]

Send TOP command or RETR command to the POP3 server.
The mail specified by msgno is retrieved.
There are three operation modes.

[POP3_MODE_TEXT]
The mail is retreived till LF, or end code is detected.
Otherwise, msglen is the maximum size to retrieve.
The mail is stored in the memory specified by msgbuf.
If the return value is 0x0, the rest of the mail text can be retrieved by issuing cnet_pop3getmsgnext

[POP3_MODE_TRUNC]
The mail is retreived till LF, or end code is detected.
Otherwise, msglen is the maximum size to retrieve.
The mail is stored in the memory specified by msgbuf.
If the return value is 0x0, the rest of the mail text can be retrieved from the next line by issuing cnet_pop3getmsgnext

[POP3_MODE_BINARY]
The mail is retreived till end code is detected.
Otherwise, msglen is the maximum size to retrieve.
The mail is stored in the memory specified by msgbuf.
If the return value is 0x0, the rest of the mail text can be retrieved by issuing cnet_pop3getmsgnext

Notes:
(1) The size of memory specified by msgbuf shall be (msglen + 1) bytes
(2) If maxline is 0, all lines of the mail will be retrieved.
(3) If maxline is 0, RETR command will be issued. Otherwise, TOP command will be issued
(4) Information retrieved between this API and cnet_pop3getmsgend is interpreted as one mail.
(5) cnet_pop3auth has to be precedent to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_GETEND	0x1	No error. End code was detected, or lines specified by “maxline” were retrieved
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout

Macro	Value	Meanings
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to TOP or RETR command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.12 cnet_pop3getmsgnext

[Scope] Continue reception of mail data.

[Format in C] int cnet_pop3getmsgnext (char *msgbuf , unsigned short msglen , unsigned char mode) ;

[Parameters]

I/O	Parameters	Functions
O	char *msgbuf ;	Pointer for the memory to store mail data
I	unsigned short msglen ;	Mail data size in bytes
I	unsigned char mode ;	Selection of receive mode POP3_MODE_TEXT: Received line by line POP3_MODE_TRUNC: Received line by line POP3_MODE_BINARY: Received up to the size

[Functions]

Continue receiving the mail data after it has been initiated by cnet_pop3getmsgfirst.
The mail data is stored in the memory specified by msgbuf.
There are three operation modes.

[POP3_MODE_TEXT]
The mail is retreived till LF, or end code is detected.
Otherwise, msglen is the maximum size to retrieve.
The mail is stored in the memory specified by msgbuf.
If the return value is 0x0, the rest of the mail text can be retrieved by issuing another cnet_pop3getmsgnext

[POP3_MODE_TRUNC]
The mail is retreived till LF, or end code is detected.
Otherwise, msglen is the maximum size to retrieve.
The mail is stored in the memory specified by msgbuf.
If the return value is 0x0, the rest of the mail text can be retrieved by issuing another cnet_pop3getmsgnext

[POP3_MODE_BINARY]
The mail is retreived till end code is detected.
Otherwise, msglen is the maximum size to retrieve.
The mail is stored in the memory specified by msgbuf.
If the return value is 0x0, the rest of the mail text can be retrieved by issuing another cnet_pop3getmsgnext

Notes:
(1) The size of memory specified by msgbuf shall be (msglen + 1) bytes
(2) cnet_pop3getmsgfirst has to be precedent to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error, but mail data is partially retrieved normally
POP3_ERROR_GETEND	0x1	No error. End code was detected, or lines specified by “maxline” were retrieved
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.13 cnet_pop3getmsgend

[Scope] Declare termination of mail data reception.
[Format in C] int cnet_pop3getmsgend (void) ;
[Parameters] None
[Functions] Declare termination of mail data reception to C-NET
Notes:
(1) Information retrieved between cnet_pop3getmsgfirst and cnet_pop3getmsgend is interpreted as one mail.
(2) cnet_pop3getmsgfirst has to be precedent to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence

4.2.14 cnet_pop3delete

[Scope] Assignment of delete mark
[Format in C] int cnet_pop3delete (unsigned int msgno) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int msgno ;	NUMBER of a mail

[Functions] Send DELE command and the NUMBER of a mail, which is to be deleted, to the POP3 server.
The POP3 server assigns a “delete mark” on the mail, which is being spooled.
Notes:

(1) The mail with the “delete mark” is actually deleted in the POP3 server based on the issue of cnet_pop3end.
(2) cnet_pop3auth has to be precedent to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to DELE command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.15 cnet_pop3reset

[Scope] Clear “delete mark”
[Format in C] int cnet_pop3reset (void) ;
[Parameters] None
[Functions] Send RESET command to the POP3 server.
“Delete mark”, assaigned by cnet_pop3delete is cleared.
Cnet_pop3auth has to be precedent to this API.

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to RSET command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence

4.2.16 cnet_pop3getuid

[Scope] Get Unique-ID
[Format in C] int cnet_pop3getuid (unsigned int msgno , char *uidbuf , unsigned short uidbuflen) ;

[Parameters]

I/O	Parameters	Functions
I	unsigned int msgno ;	Mail number
O	Char *uidbuf ;	Pointer to store Unique-ID
I	unsigned short uidbuflen ;	Size of memory, which is pointed by uidbuf, to store Unique-ID in bytes

[Functions] Send UIDL command and Mail Number to the POP3 server.
Unique ID of the mail is stored in memory where uidbuf specified.
Cnet_pop3auth has to be precedent to this API

[Return values]

Macro	Value	Meanings
POP3_ERROR_NORMAL	0x0	No error
POP3_ERROR_NOBUF	0xffffff80	Not enough memory to store Unique-ID
POP3_ERROR_READ	0xffffffe0	POP3 server response timeout
POP3_ERROR_DISCONNECT	0xfffffff0	Disconnected by POP3 server
POP3_ERROR_REPLY	0xfffffffc	POP3 server response error to UIDL command
POP3_ERROR_STATE	0xfffffffe	Wrong API sequence
POP3_ERROR_PARAMETER	0xffffffff	Illegal parameter

4.2.17 cnet_dhcpstart

[Scope] Ask DHCP server to assign network information automatically

[Format in C] int cnet_dhcpstart (struct netif *netif) ;

[Parameters]

I/O	Parameters	Functions
I	struct netif *netif ;	Pointer to network information, netif

[Functions] Ask DHCP server to assign network information automatically.
Network information comprises of IP address, subnet mask, and gateway address.
The network information is stored in the memory, which is pointed by netif.
Notes:
(1) The network information shall match the return value of the previous netif_add.
(2) Valid period of the network information depends on a DHCP server. Therefore, it is suggested to update it in every 100 msec by cnet_dhcptmr.
(3) Once the automatic assignment has been completed, the network information can be read by issuing cnet_getipaddr, cnet_getnetmask, or cnet_getgwaddr.
(4) netif_set_default has to be precedent to this API.

[Return values]

Value	Meanings
0	No error
-1	Not enough memory
-8	Illegal parameter
-9	DHCP server response timeout

4.2.18 cnet_dhcpstop

[Scope] Clear the assigned network information
[Format in C] void cnet_dhcpstop (struct netif *netif) ;

[Parameters]

I/O	Parameters	Functions
I	struct netif *netif ;	Pointer to network information, netif

[Functions] Clear the assigned network information
Notes
(1) The network information shall match the information assigned by cnet_dhcpstart
(2) cnet_dhcpstart has to be precedent to this API.
[Return values] None

4.2.19 cnet_httpdinit

[Scope] Initialize transfer function of Hyper text
[Format in C] void cnet_httpdinit (void) ;
[Parameters] None
[Functions] Initialize transfer function of Hyper text
Initialization is executed.
Note:
netif_set_default and cnet_dhcpstart have to be precedent to this API.
[Return values] None

4.2.20 cnet_httpdsend

[Scope] Send data to Web client
[Format in C] int cnet_httpdsend (int id , const char *msg , int msglen , unsigned char copy) ;

[Parameters]

I/O	Parameters	Functions
I	int id ;	ID assigned to User defined function
I	const char *msg ;	Pointer to the content area
I	int msglen ;	Data size assigned by msg in byte
I	unsigned char copy ;	Requirement of copy 0: if volatile content 1: If fixed content

[Functions]

Destination is the Web client specified by msg.
Note:
This API is valid in user defined function.

[Return values]

Value	Meanings
0	No error
-1	Not enough work space
-8	Illegal parameter