

ROM FILE MAKER

MANUAL

V2.5



Wiznet, Inc

Copyright 2002 WIZnet, Inc. All Rights Reserved.

Technical Support: support@wiznet.co.kr

Sales & Distribution: sales@wiznet.co.kr

General Information: info@wiznet.co.kr

For more information,

visit our website at <http://www.iinchip.com> or <http://www.wiznet.co.kr>

CONTENTS

1	What is 'ROM File System'?	3
2	About 'ROM File Maker'	4
3	Installation	4
4	How to use	4

FIGURES

<Fig 1.1>	Structure of iinChip™ EVB B/D's ROM File System	3
<Fig 4.1>	ROM File Maker execution	5
<Fig 4.2>	EVB-A1's "types.h"	6
<Fig 4.3>	'ROM File' additional files list	7
<Fig 4.4>	'ROM File System' creating completion	7
<Fig 4.5>	"romfs.h"	8
<Fig 4.6>	"romfs.c"	9

TABLES

<Table 2-1>	search_file()	4
-------------	---------------	---

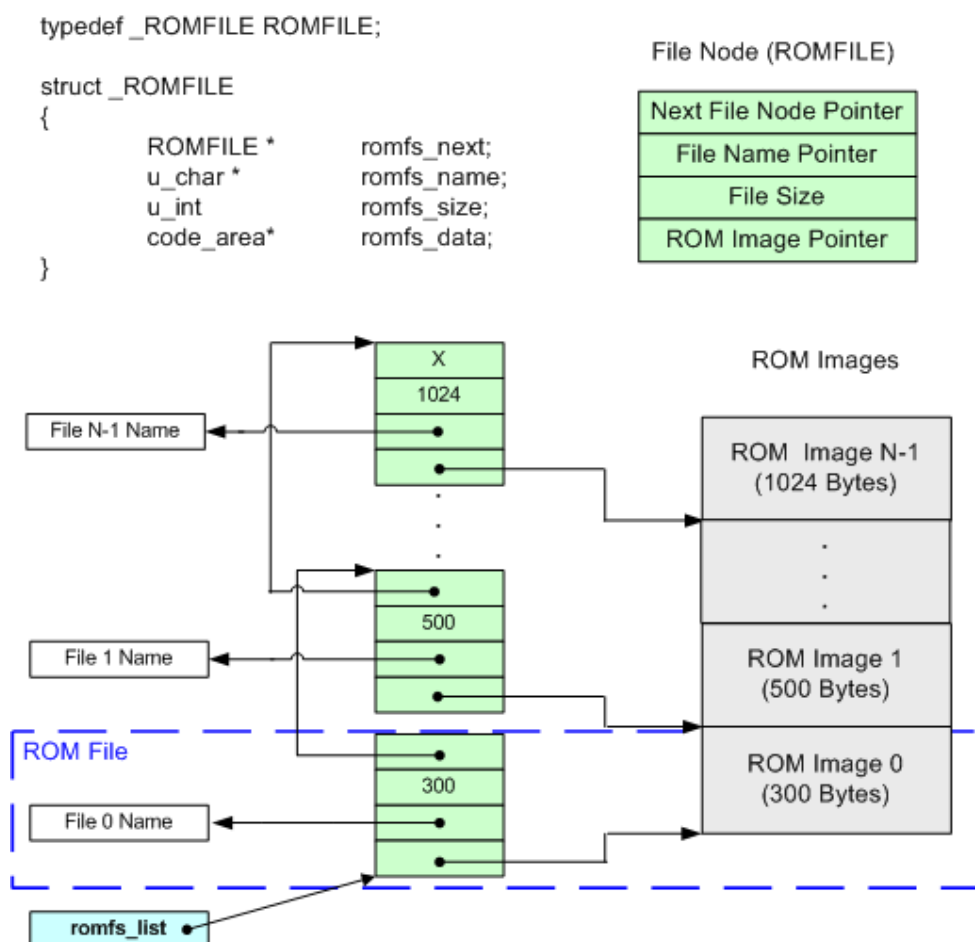
1 What is 'ROM File System'?

'ROM Files' of iinChip™ EVB B/D means loaded files in Code Memory Area(i.e. ROM,Flash, etc). Each ROM File is made up of File Name, File Size and File Contents. Each contents of ROM File are converted into Binary Code and are loaded in ROM(Hence, we mention the loaded contents to 'ROM Image".)

Originally, 'ROM File System' is designed for easier management of 'ROM File'. This system defines each 'ROM File' as **ROMFILE** which is File Node Structure and manages them through **romfs_list** which makes you possible to refer ROMFILES orderly.

When you handle various files (i.e. web page files (HTML, JPEG, GIF)) for development iinChip™ EVB,'ROM File System' helps you develop efficiently and manage easily.

Following figure introduces the structure of EVB B/D's ROM File System. You can see the correlation between ROMFILES as well as ROMFILE and ROM Image as shown in <Fig 1.1>.



<Fig 1.1> Structure of iinChip™ EVB B/D's ROM File System

As shown in <Fig 1.1>, ‘ROM Image’ is located in iinChip™ EVB B/D’s ROM and **romfs_list**, which is the Linked List of File Nodes is located in EVB B/D’s RAM.

2 About ‘ROM File Maker’

‘ROM File Maker’ is the composition tool for ‘ROM File System’, Window-based PC program. It is required when you develop iinChip™ EVB F/W.

‘ROM File Maker’ executes **ROMFILE** Structure definition, ‘ROM Image’ creation, ‘ROM File’ search function, and Source Files(romfs.h & romfs.c) written in C Language.

You can search each ‘ROM File’ in ‘ROM File System’ using ‘**search_file()**’.

Following Table is the explanation about **search_file()**.

<Table 2-1> search_file()

Prototype	u_char search_file(u_char * name, code_area ** buf, u_int * len)	
Parameter	name	searching ROM File Name
	buf	searched ROM File Location Return
	len	searched ROM File Size Return
Result	> 0 : Success, 0 : Fail	

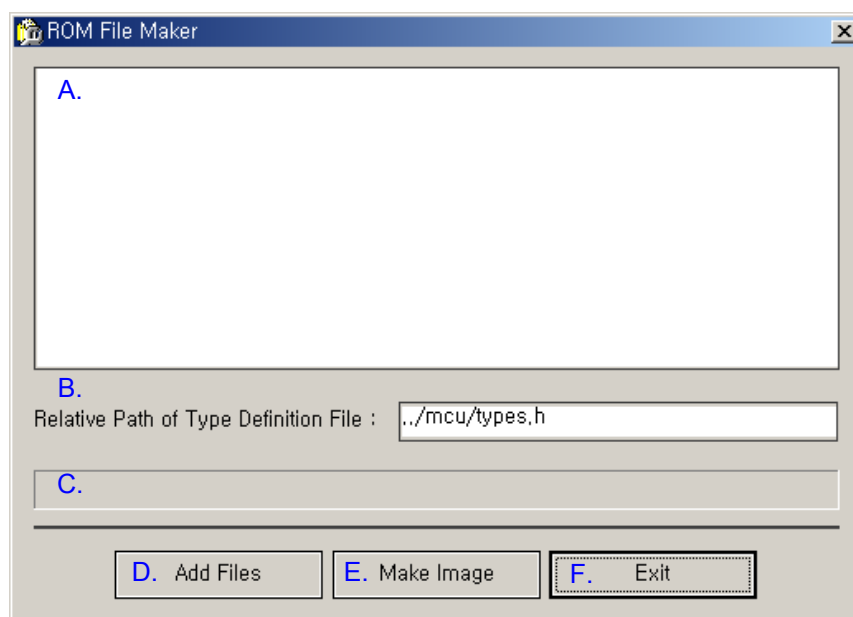
3 Installation

You can install ‘ROM File Maker’ through executing “RFMaker VX.X.exe”

iinChip™ EVB B/D’s Software CD provides this install file. Moreover, you can download the latest version of install file from the manufacturer ‘WIZnet’ homepage(www.wiznet.co.kr)

4 How to use

- ① Please execute “RFMaker.exe”



<Fig 4.1> ROM File Maker execution

- A. File List
- B. Relative Path Name that is defined as Data Type used in iinChip™ EVB B/D
- C. ‘ROM File System’ creation process.
- D. Add the file that would be formed into ‘ROM File’
- E. ‘ROM File System’ creation
- F. Program shut down.

② Please appoint Relative Path about the file defined “code_area”Type.

<Attention!> You should define “code_area” Type in the appointed file necessarily.
The reason why “code_area” Type is using CODE Memory Type that is referred in “romfs.h” and “romfs.c”.

iinChip™ EVB F/W defines “code_area” Type of “types.h” as shown in <fig 4.2>

```

#ifndef _TYPE_H_
#define _TYPE_H_

#include <avr/pgmspace.h>

#ifndef NULL
#define NULL ((void *) 0)
#endif

typedef enum { false, true } bool;

#ifndef _SIZE_T
#define _SIZE_T
typedef unsigned int size_t;
#endif

typedef unsigned char    BYTE;        /* 8-bit value */
typedef unsigned char    UCHAR;       /* 8-bit value */
typedef unsigned int     INT;         /* 16-bit value */
typedef unsigned int     UINT;        /* 16-bit value */
typedef unsigned short   USHORT;     /* 16-bit value */
typedef unsigned short   WORD;       /* 16-bit value */
typedef unsigned long    ULONG;      /* 32-bit value */
typedef unsigned long    DWORD;      /* 32-bit value */

/* bsd */
typedef unsigned char    u_char;     /* 8-bit value */
typedef unsigned short   u_short;    /* 16-bit value */
typedef unsigned int     u_int;      /* 16-bit value */
typedef unsigned long    u_long;     /* 32-bit value */

typedef UCHAR    SOCKET;

typedef union _un_l2cval {
    u_long    lVal;
    u_char    cVal[4];
}un_l2cval;

typedef union _un_i2cval {
    u_int     iVal;
    u_char    cVal[2];
}un_i2cval;

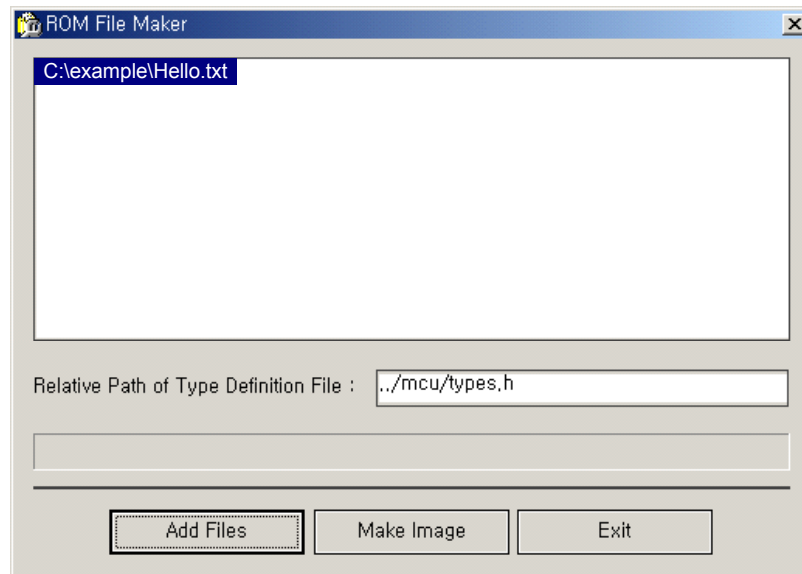
typedef prog_char    code_area; ←
#endif          /* _TYPE_H_ */

```

<Fig 4.2> EVB-A1's "types.h"

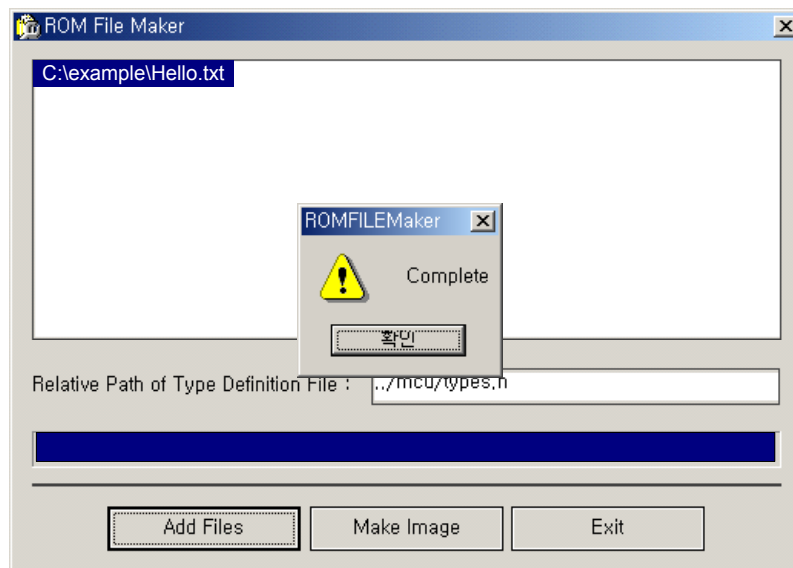
③ Please click 'Add File' button.

In "File Open Dialog", select files that you are willing to add into 'ROM File System'.



<Fig 4.3> 'ROM File' additional files list

- ④ Please create 'ROM File System' through the click of 'Make Image' button.



<Fig 4.4> 'ROM File System' creating completion

'ROM File System' creating process is formed out of 'C Language Source File' of 'romfs.h' and 'forfs.c'.

If you add "romfs.h" and "romfs.c" into iinChip™ EVB F/Ws Project and compile, the embodiment of 'ROM File System' easily is completed.

```
#ifndef _ROMFS_H_
#define _ROMFS_H_

#include "../mcu/types.h"

typedef struct _ROMFILE ROMFILE;

struct _ROMFILE
{
    ROMFILE *romfs_next;    /* Link to next ROMFILE structure. */
    u_char  *romfs_name;    /* Filename */
    u_int    romfs_size;    /* File size. */
    code_area *romfs_data; /* File contents. */
};

extern ROMFILE* romfs_list;

u_char search_file(u_char * name, code_area ** buf, u_int * len); // Search a file from ROM FILE

#endif /* ROMFS H */
```

<Fig 4.5> “romfs.h”


```

/*
 * This file is automatically created by ROM File Maker
 */
#include <string.h>
#include "romfs.h"

/*
 * File 0 : D:\WIZCHIP_TEST_APP\ROMFileMaker\example\Hello.txt
 */
code_area file0data[] = {
ROM IMAGE → 0x48, 0x65, 0x6c, 0x6c, 0x6f, 0x2c, 0x20, 0x57, 0x49, 0x5a, 0x6e, 0x65, 0x74, 0x2e,
};
static ROMFILE file0entry = { 0, "Hello.txt", 14, (code_area *)file0data };

Linked List
of → ROMFILE* romf1_list = &file0entry;
ROMFILE

/*
Description : Search a file from ROM FILE
Argument    : name - file name
              buf - file contents to be return
              len - file length to be return
Return Value :
Note        :
*/
Search API → u_char search_file(u_char * name, code_area ** buf, u_int * len)
{
    int i;
    ROMFILE *romfs;

    i = 0;

    for (romfs = romfs_list; romfs; romfs = romfs->romfs_next)
    {
        if (!strcmp(name, romfs->romfs_name))
        {
            *len = romfs->romfs_size;
            *buf = romfs->romfs_data;
            return ++i;
        }
    }

    return 0;
}

```

<Fig 4.6> “romfs.c”