NETWORK MANIAS WHITE PAPER

Recommended C Style and Coding Standards version 3.0

by 유창모(cmyoo@medialincs.com)

Recommended C Style and Coding Standards version 3.0

2001년 4월 23일 유창모

1. Introduction

본 문서는 소프트웨어 개발에 있어서의 표준적인 방향을 제시함으로서 코딩을 함에 있어서의 비효율성 및 인력간의 편차를 줄이고 의사소통의 원활화를 통하여 개발 기간의 단축 및 소스 코드의 재활용성을 제공하는데 그 목적을 두고 있다. 본 문서에서 언급하는 내용은 단지 본인의 스타일을 정리한 것 뿐이며 어느 누구에게도 이 규칙을 강요 할 생각은 없다. 다만 소프트웨어 엔지니어라면 각자 자신의 스타일을 가지고 있어야 한다고 생각한다.

Symbol	Description
␣	space
↹	tab(/t)
[↹]	number of tabs
↵	new line(/n)

2. File Heading

소스 파일(*.c/*.s), 헤더 파일(*.h), makefile등의 파일 첫머리에 아래와 같은 prologue를 붙여준다. 박스 안에 있는 내용은 경우(copyright or copyleft)에 따라서 바뀔 수 있다. 프로그램 수정 시에는 반드시 Revision History에 그 내용을 적어 주도록 하며, 수정한 사람의 이름 전체와 그 약자를 함께 적어서 본 코드의 수정자를 다른 사람들이 쉽게 알아 볼 수 있도록 한다.

Copyright

```
/* set tabstop=4 */
/*****************************************************************************
 *                                                                           *
 * (c) COPYRIGHT 1999-2001␣␣Medialincs                                       *
 *                                                                           *
 * ALL RIGHTS RESERVED                                                       *
```

```
 *                                                                            *
 * This software is the property of Medialincs and is furnished under         *
 * license by Medialincs.  This software may be used only in accordance       *
 * with the terms of said license.  This copyright notice may not be          *
 * removed, modified or obliterated without the prior written permission      *
 * of Medialincs.                                                             *
 *                                                                            *
 * This software may not be copied, transmitted, provided to or otherwise     *
 * made available to any other person, company, corporation or other entity   *
 * except as specified in the terms of said license.                          *
 *                                                                            *
 * No right, title, ownership or other interest in the software is hereby     *
 * granted or transferred.                                                    *
 *                                                                            *
 * The information contained herein is subject to change without notice and   *
 * should not be construed as a commitment by Medialincs.                     *
 *                                                                            *
 ******************************************************************************
↲
␣␣MODULE NAME:␣␣SAMPLE.C
↲
  REVISION HISTORY:
↲

  Date␣␣␣␣␣␣␣␣Ver.␣Name␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Description
  ---------- --- ------------------ ----------------------------------------
  02/25/2000 1.0 ChangmoYoo(cmyoo)  Created

 ..............................................................................
↲
␣␣DESCRIPTION:
↲
␣␣This module contains ...
↲
..............................................................................
*/
↲
↲
```

Copyleft

```
/* set tabstop=4 */
/******************************************************************************
 *                                                                            *
 * 1999-2001  Changmo Yoo                                                     *
 *                                                                            *
 * This program may be distributed and used for any purpose.                  *
 * I ask only that you:                                                       *
 *      1. Leave this author information intact.                              *
 *      2. Document any changes you make.                                     *
 *                                                                            *
 * Changmo Yoo                                                                *
 * Medialincs                                                                 *
 *                                                                            *
 * cmyoo@medialincs.com                                                       *
 * cmyoo@lycos.co.kr                                                          *
 *                                                                            *
 ******************************************************************************
↲
␣␣MODULE NAME:␣␣SAMPLE.C
↲
  REVISION HISTORY:
↲
```

```
  Date␣␣␣␣␣␣␣␣Ver␣Name␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Description
  ---------- --- ------------------ ----------------------------------------
  02/25/2000 1.0 ChangmoYoo(cmyoo)  Created

 ..........................................................................
↲
␣␣DESCRIPTION:
↲
␣␣This module contains ...
↲
..........................................................................
*/
↲
↲
```

3. C Coding Conventions

3.1 Program File Layout

File heading에 이어서 소스 파일(C source/Assembler source)에 아래와 같이 module layout을 붙여 준다.

```
/** ************************************************************************ **
 ** includes
 ** ************************************************************************ **/
include되는 header file을 적어준다. 가급적이면 파일이 위치하는 전체 경로를 적지 않도록 한다. 이 경로는 makefile에 적어주도록 한다.
↲
↲
/** ************************************************************************ **
 ** defines
 ** ************************************************************************ **/
#define을 이용한 constant/function macro를 정의한다. constant macro 이후에 function macro를 위치시킨다. 만약 constant definition이
특정 전역 변수와 관련이 있을 경우, 본 정의는 그 전역 변수를 정의하기 바로 전에 하도록 하며, 특정 structure와 관련이 있을 경우에는 그
structure의 typedef 안에서 정의하도록 한다.
↲
↲
/** ************************************************************************ **
 ** typedefs
 ** ************************************************************************ **/
typedef를 이용한 데이터 타입 및 스트럭쳐를 정의한다.
↲
↲
/** ************************************************************************ **
 ** globals
 ** ************************************************************************ **/
둘 이상의 파일에서 공유되어 지는 전역 변수를 정의한다.
↲
↲
/** ************************************************************************ **
 ** locals
 ** ************************************************************************ **/
이 파일에서만 사용되는 전역 변수를 정의한다.
↲
↲
/** ************************************************************************ **
 ** forward declarations
 ** ************************************************************************ **/
```

함수 프로토타입을 정의한다. "depth-first" 보다는 "breadth-first" 방식으로 나열하며, 동일 레벨의 독립된 함수가 다수개 존재하게 될 경우, alphabetical ordering으로 한다.
↲
↲

3.2 Header File Layout

Header 파일에는 file heading에 이어 아래와 같이 module layout을 붙여준다. 한번 이상 헤더 파일이 include되지 않도록, 파일의 시작과 끝에 "trapdoor"를 포함시키며, 헤더 파일에서는 전역 변수를 선언하지 않도록 한다.

```
#ifdef↹ _FILENAME_H_
#define↹ _FILENAME_H_
↲
↲
/** ************************************************************************ **
 ** includes
 ** ************************************************************************ **/
include되는 header file을 적어준다.
↲
↲
/** ************************************************************************ **
 ** defines
 ** ************************************************************************ **/
#define을 이용한 constant 또는 macros를 정의한다.
↲
↲
/** ************************************************************************ **
 ** typedefs
 ** ************************************************************************ **/
typedef를 이용한 데이터 타입 및 스트럭쳐를 정의한다.
↲
↲
/** ************************************************************************ **
 ** function prototypes
 ** ************************************************************************ **/
함수 프로토타입을 정의한다.
↲
↲
#endif↹ _FILENAME_H_
↲
↲
```

3.3 C Subroutine Layout

각 함수의 처음과 끝에 다음과 같은 subroutine layout을 붙여준다.

```
↲
↲
/*****************************************************************************
 *
 *␣␣FUNCTIONS: 함수 이름을 적는다.
 *↲
 *  SYNOPSIS: 함수 기능을 설명한다.
 *↲
 *  EXTERNAL EFFECTS: 이 함수에서 참조(read/write)되는 전역변수를 나열한다.
```

```
 *↲
 *  PARAMETERS: 파라미터의 의미를 나열한다.
 *↲
 *  RETURNS: 리턴 값 및 그 의미를 적는다.
 *↲
 *  ERRNO: 에러 발생시 리턴되는 값의 의미를 적는다.
 *↲
 ******************************************************************************/
void
function_name(void)
{

}↹ /* end of function_name */
```

3.4 C Declaration Formats

3.4.1 Types

unsigned형 및 boolean형 변수 타입 및 리턴 값(OK, ERR등)에 대한 definition은, 사용하는 OS의 정의를 그대로 사용하도록 하며, 만약 OS에서 이를 정의하지 않았을 경우에는 다음과 같이 한다.

```
typedef unsigned char         uns8;     /*  8-bit */
typedef unsigned short        uns16;    /* 16-bit */
typedef unsigned int          uns32;    /* 32-bit */

#define YES                   1
#define NO                    0
#define TRUE                  1
#define FALSE                 0
typedef unsigned int          bool;
```

3.4.2 Variables

❶ 포인터 변수 선언 시, "*"는 변수이름 쪽에 붙이도록 한다.

```
ex) uns8   *p_ptr;
```

❷ 스트럭쳐 선언은 다음과 같은 모양으로 한다. 이 포맷은 *union*, *enum* 선언시에도 동일하게 적용된다.

```
ex)
struct app_control                          /* Application_Control           */
{
      struct native_ifcb *p_native_ifcb;    /* Native Interface Control Block */
      struct atm_conn_id conn_id;           /* ATM Connection id(vpi/vci)     */
};
struct app_control  s_app_ctl;
```

3.4.3 Subroutines

❶ 함수의 리턴타입과 아규먼트에 대한 설명은 앞서 보인 "C Subroutine Layout"에서 하도록 하며 함수의 정의는 다음 예와 같은 모양을 갖도록 한다. 함수의 리턴 타입이 포인터일 경우, "*"는 리턴 타입 쪽에 붙이도록 하며, 리턴 타입과 함수 이름 사이에는 return(CR/LF)을 넣어준다. 만약 함수의 아규먼트를

통해서 값이 리턴 될 경우, 이 아규먼트를 함수의 첫번째 파라미터로 한다.

```
ex)
void*
m860drv_init(uns32 debug_value)
{

}↹ /* end of m860drv_init */
```

❷ 함수 수행의 성공 여부를 리턴하는 경우에 일관성 있는 리턴값을 가지도록 한다. 예를 들어 성공적인 수행에 대해서는 SUCCESS(0)을 리턴하고 그렇지 않을 경우에는 음수값을 가지는 FAILURE(-1, -2, ...)를 리턴함으로서 그 결과와 실패 시에 이유를 동시에 알 수 있도록 한다.

3.5 C Code Layout

코드의 모든 라인은 최대 80 character를 넘지 않도록 한다.

3.5.1 Vertical Spacing

❶ 서로 관계가 없는 변수들의 타입이 같다고, 이 변수들의 선언을 한 줄에 하지 않도록 한다. 즉, 가급적 전역/지역 변수 선언은 한 줄에 하나씩만 하도록 한다.

❷ Braces('{' and '}')와 case 라벨, if, else if, else는 독립된 라인을 가지도록 한다.

```
ex)
if (condition)
{
      statements;
}
else
{
      statements;
}
```

```
switch (input)
{
     case 'a':
          statements;
          break;
     default:
          statements;
          break;
}
```

3.5.2 Horizontal Spacing

❶ *if*, *else if*, *switch*, *for*, *while*, *return*과 left paranthesis('(') 사이에는 한 칸 띄워주도록 하며, function, macro, *sizeof*는 바로 붙여주도록 한다.

```
ex)
if␣(ret_val != 0)
m_SAR_INIT(g_ifcb, NULL);
return␣(a);
```

❷ paranthesis('(' and ')') 안에 들어가는 숫자, 문자의 길이가 짧을 경우 paranthesis와 붙여 쓰고, 길 경우에는 한 칸 띄어주도록 한다.

```
ex)
for (i = 0; i < 10; i++)
if (␣(ret_val = m860_open_aal5_connection()) == ERROR␣)
```

❸ 하나의 문장이 2줄 이상 계속될 경우, 보기 좋도록 배열한다.

```
ex)
ret = m860_send_aal5_cpcs_sdu(struct app_control *p_sap, uns8* p_call_payload,
                              enum cell_type cell_type);
```

❹ 포인터 타입으로 캐스팅 할 경우, 다음 예와 같이 "*"와 캐스팅 되는 타입간에 공백을 준다.

```
ex) if (gp_ifcb == (struct native_ifcb␣*)NULL)
```

3.5.3 Intention

tab size는 4나 8로 설정하도록 한다. 원칙적으로는 4로 하며, out sourcing하는 코드가 이를 지키기 않을 경우에는 그 코드의 rule을 따르도록 한다.

3.5.4 Comments

"When the code and the comments disagree, both are probably wrong." - Norm Schryer

❶ 주석문 안에서는 tab을 쓰지 말고 space를 쓰도록 한다(File Heading, Program/Header File Layout, C Subroutine Layout 모두에 해당됨).

❷ 코드 수정 시에는 다음과 같이 일관된 포맷의 주석문을 달아 주도록 한다. 맨 앞의 이름은 File Heading의 Revision History에 명시한 수정자의 약자를 사용하고, 이후에 수정한 날짜를 적고 @ 표시 다음에는 코드 수정 내용(add, modify, delete등)을 적도록 한다(코드 수정 시에 이와 같이 하는 것이지 처음 코드를 작성(생성) 할 때는 본 rule을 따를 필요가 없음).

```
ex)
/*␣cmyoo-20010515@add: Initialize the underlying SAR driver */
m_SAR_INIT(g_ifcb, NULL);
```

❸ 한 줄로 끝나는 주석문의 경우, 다음 예와 같이 한다.

```
ex)
/*␣Initialize the underlying SAR driver */
m_SAR_INIT(g_ifcb, NULL);
```

❹ 두 줄 이상 되는 주석문의 경우, 다음 예와 같이 한다.

```
ex)
/*
 *␣Wait for SAR initialization to complete before opening
 * the given VPI/VCI channel
 */
while (m_SAR_GET_LOS_FLAG(((void *)g_ifcb), NULL) == SAR_LOS_PRESENT)
```

❺ 변수에 대한 주석문의 경우, 선언된 라인의 우측에 적당한 공백(tab)을 두도록 하며, 만약 두 줄 이상이 될 경우, 다음과 같은 모양을 가지도록 한다.

```
ex)
int   a;   /*␣This is the correct format for a multiline comment *
            * in a declaration                                    */
```

❻ for, while, switch, if가 nesting된 복잡한 구조를 가질 경우, braces의 쌍을 명시해주도록 한다.

```
ex)
for (...)
{
    while (...)
    {
        ...
        if(...)
            ...
    }↹/* end of while (...) */
}↹/* end of for (...) */
```

3.6 C Naming Conventions

프로그램 작성 시, 파일 이름, 함수 이름, 전역/지역 변수 등의 이름을 어떻게 명명하느냐는 프로그램 작업에서 매우 중요한 사항이다. 다음과 같은 규칙을 따른다.

❶ 함수 이름, 전역/지역 변수 이름은 모두 소문자로 하며, *typedef struct*, *typedef union*, *typedef enum*, *#define*에 의해 정의되는 것은 모두 대문자로 한다.

❷ 하나 이상의 파일에서 공유하는 함수 및 전역 변수 이름은 「**module_verb_noun**」 rule을 지키도록 한다. 여기서 맨 앞에 위치하는 module은 파일 이름이나 네트워크 프로토콜의 경우, 해당 레이어 이름으로 하며, 최대한 짧게 이름을 짓도록 한다.

```
ex)
m860drv.c                              /* file name */
m860drv_send_aal5_cpcs_sdu();          /* function name */
uns32     g_m860drv_conn_count;        /* global variable name */
```

❸ 하나 이상의 파일에서 공유되는 전역 변수 앞에는 「**g_**」 prefix를 붙인다. 이 변수 선언들은 "C Module Layout"의 "globals"에 놓이게 된다.

```
ex) uns32      g_m860drv_conn_count;
```

❹ 하나의 파일에서만 사용되는 전역 변수 또는 함수 내부에서 선언되는 static 변수는 앞에 「**s_**」 prefix를 붙인다. 이 변수 선언들은 "C Module Layout"의 "locals" 또는 함수 내부에 놓이게 된다.

```
ex) static uns32        s_conn_count;
```

❺ 매크로 이름 앞에는 「**m_**」 prefix를 붙인다.

```
ex) #define    m_INC_CONN_COUNT()      (g_conn_count++);
```

❻ 포인터 변수 앞에는 「**p_**」 prefix를 붙이도록 하며, 포인터 레벨에 따라서 p의 개수를 늘려준다.

```
ex) uns8          **pp_conn_count;
```

❼「g(s)_」prefix와「p_」prefix가 같이 사용될 경우의 예는 다음과 같다.

```
ex)
uns32          *gp_m860drv_conn_count;
static uns32   **spp_conn_count;
```

❽ 파일 이름은 「**module_faculty**」rule을 지키도록 한다. 여기서 faculty는 해당 파일에 정의된 함수들의 기능을 함축적으로 나타낸 수 있는 이름으로 짓는다.

```
ex)
sar_rx.c       sar_init.c    sar_isr.c     sar_def.h
```

4. C Style

- *#define*을 이용한 constant definition이 특정 전역 변수와 관련이 있을 경우, 그 정의는 전역 변수를 정의한 곳에서 가까운 위치에 하도록 하며, 특정 structure와 관련이 있을 경우에는 그 structure의 typedef 안에서 정의하도록 한다.

```
ex1)
#define MAX_CONNECTION        32

static CONN_CBF     g_conn_cbf[MAX_CONNECTION];
static int          g_conn_count;

ex2)
struct _            /* Connection_ControlBlock */
{
      #define VALID      1
      #define INVALID    0

      int       valid;
      int       vpi;
      int       vci;
      uns32     (*rcv_func)(uns8 *p_cpcs_pdu, int sdu_len);
} conn_cb;
```

- constant definition의 값에 큰 의미가 없을 경우, *#define*보다는 *enum*을 사용하도록 한다.

```
ex)
enum ea                             /* EntryAttribyte */
{
      INVALID, VALID
};

struct conn_cb                      /* Connection_ControlBlock */
{
      enum EA   valid;
      int       vpi;                /* 8 bit VPI  */
      int       vci;                /* 16 bit VCI */
      uns32     (*rcv_func)(uns8 *p_cpcs_pdu, int sdu_len);
};
```

- 둘 이상의 파일이 공유하는 전역 변수의 경우, 헤더 파일에 그 정의를 하지 않도록 한다. 대신 소스 파일에 정의를 하고, 그 변수를 사용하는 다른 파일(들)에서는, 그 변수를 사용하는 함수마다 그 안에서 *extern*으로 명시하고 사용하도록 한다. 또한 external variable이 배열일 경우, extern 선언 시에 배열 크

기를 명시하도록 한다.

- 하나의 파일에서만 사용되는 전역 변수 및 함수 앞에는, 반드시 *static*을 붙여주도록 한다. 만약 debugger에서 static 변수의 값을 watch할 수 없을 경우, 다음의 예와 같이 해 주도록 한다.

```
ex)
#ifdef     _DEBUG_SUPP
#    define STATIC
#else
#    define STATIC   static
#endif

STATIC int g_conn_count;
```

- 만약 함수 내에서 전역 변수를 수정하거나, 아규먼트로 넘어온 포인터를 수정할 경우, "C Subroutine layout"의 "EXTERNAL EFFECTS"에 그 리스트를 적도록 한다.

- 루프 문에서 사용되는 루프 카운터는 *int*를 사용한다. 이는 machine에서 가장 효율적으로 처리할 수 있는 단위이기 때문이다.

- boolean 테스트에 대해서 FALSE가 항상 0이라 가정하지 마라. 어떤 경우에는 -1이 될 수도 있으며, 좌측의 예의 경우에는 문제가 된다.

```
ex)
/* Bad */                          /* Good */
if (call_subroutine())             if (call_subroutine() != FALSE)
{                                  {
    printf("success\n");               printf("success\n");
}                                  }
else                               else
{                                  {
    printf("error\n");                 printf("error\n");
}                                  }
```

- *for*나 *while* 루프가 null body를 가질 경우, 다음과 같은 모양으로 한다.

```
ex)
while (*dest++ = *src++)
    ;␣/* nop */
```

- TRUE나 FALSE를 리턴 하는 함수의 경우, 그 이름의 의미를 분명히 하도록 한다.

```
ex)
if (check_valid() == TRUE)         /* Bad  */
if (is_valid() == TRUE)            /* Good */
```

- 함수 호출 시 아규먼트로 스트럭쳐를 패싱해야 될 경우, 스트럭쳐 포인터를 넘기도록 한다. (스트럭쳐 값을 패싱하더라도, 일반적으로 컴파일러에 의해서 포인터 패싱으로 변환된다.)

- 모든 지역 변수의 이름에도 전역 변수처럼 의미성을 두려고 노력하지 마라.

```
ex)
/* Bad */
for (element_index = 0; element_index < DIMENSION; elemnet_index++)
    printf("%d\n", element[element_index]);
```

```
/* Good */
for (i = 0; i < DIMENSION; i++)
    printf("%d\n", element[i]);
```

- CPU의 cache 성능을 최대화하기 위해서, *goto*, *continue* 문은 가급적 사용하지 않도록 한다. 다음의 예와 같이, for, while, if가 nesting되어 지는 경우에만 goto를 사용하도록 한다.

```
ex)
for (...)
{
    while (...)
    {
        ...
        if(disaster)
            goto ERROR;
    }
}
....

ERROR:
```

- 코드 내에서 상수를 바로 사용하지 말고, 대신 *#define*으로 의미있는 이름을 지어주도록 한다. 단, 1과 0의 경우에는 때에 따라서 그냥 그 값을 사용하도록 한다.

```
ex)
for (i = 0; i < 12; i++)                    /* Bad */

#define ARY_BOUND  12
for (i = 0; i < ARY_BOUND; i++)             /* Good */
```

- *if/else*와 function macro가 함께 사용될 경우에 다음 두가지 중 하나를 사용하도록 한다.

❶ if/else안이 한 문장이더라도 brace를 해 준다.

```
ex)
if (x == 3)
{
    m_SP3();
}
else
{
    m_BORK();
}
```

❷ 매크로 내에서 brace를 하도록 한다.

```
ex)
#define m_STMT(stuff)      { do { stuff } while(0) }
#define m_SP3()            m_STMT( if(b) { int x; ab = f(&x); bv += x; } )

if (x == 3)
    m_SP3();
else
    m_BORK();
```

- *#ifdef/#if*에서 그 값이 명시되지 않았을 경우, 컴파일 에러를 발생시키도록 한다.

```
ex)
#if (CPU == PPC860)
    ....
#elif (CPU == I960)
    ....
```

```
#else
   #error "CPU is not defined!"
```

- *#ifdef/#if* 사용은 가급적 헤더 파일에서만 하도록 한다. 다음 예와 같이 *#ifdef*과 *#define*을 사용하여, 소스 파일에서는 동일한 인터페이스를 가지도록 한다.

```
ex)
#ifdef DEBUG
   extern void* mm_malloc();
#   define MALLOC(size)  (mm_malloc(size))
#else
   extern void* malloc();
#   define MALLOC(size)  (malloc(size))
#endif
```

- 시스템에서 제공하는 함수나 본인이 작성한 함수 호출 시에, 리턴 되는 값에 대해서 에러가 발생하였는지를 일일이 체크하도록 한다. 이를 위해서 여러개의 함수를 한 줄에서 호출하는 경우와 return 문에서 함수를 호출하는 일은 삼가하도록 한다.

참고 문헌

1. Bell Labs, University of Toronto, “Indian Hill C Style and Coding Standards as amended for U of T Zoology UNIX”, April 1990
2. Bell Labs, University of Toronto, University of Washington, “Recommended C Style and Coding Standards”, July 1991
3. Rob Pike, “Notes on Programming in C”
4. “Standards and Style for Coding in ANSI C”
5. WindRiver, “Tornado User's Guide, Appendix F. Coding Conventions”

예제 코드

ipfe_arp.c

```
/* set tabstop=4 */
/*****************************************************************************
 *                                                                           *
 * 1999-2001  Changmo Yoo                                                    *
 *                                                                           *
 * This program may be distributed and used for any purpose.                 *
 * I ask only that you:                                                      *
 *     1. Leave this author information intact.                              *
 *     2. Document any changes you make.                                     *
 *                                                                           *
 * Changmo Yoo                                                               *
 * Medialincs                                                                *
 *                                                                           *
 * cmyoo@medialincs.com                                                      *
 * cmyoo@lycos.co.kr                                                         *
 *                                                                           *
 *****************************************************************************

  MODULE NAME:  IPFE_ARP.H

  REVISION HISTORY:

  Date        Version   Name             Description
  ----------  -------   ---------------- ---------------------------------------
  11/01/2000  1.0A      ChagmoYoo(cmyoo) Created
 ...........................................................................

  DESCRIPTION:

  This file contains ARP definitions

 ...........................................................................
*/

#ifndef _IPFE_ARP_H
#define _IPFE_ARP_H


/** ********************************************************************* **
 ** includes
 ** ********************************************************************* **/


/** ********************************************************************* **
 ** defines
 ** ********************************************************************* **/


/** ********************************************************************* **
 ** typedefs
 ** ********************************************************************* **/


/** ********************************************************************* **
 ** function prototypes
 ** ********************************************************************* **/
void    arp_request(int ifnum, uns32 dstip);
```

```
int     arp_input(uns8 *data, uns32 *ip, uns8 *hwaddr);

#endif _IPFE_ARP_H

/* end of ipfe_arp.h */
```

ipfe_arp.c

```
/* set tabstop=4 */
/*****************************************************************************
 *                                                                           *
 * 1999-2001  Changmo Yoo                                                    *
 *                                                                           *
 * This program may be distributed and used for any purpose.                 *
 * I ask only that you:                                                      *
 *     1. Leave this author information intact.                              *
 *     2. Document any changes you make.                                     *
 *                                                                           *
 * Changmo Yoo                                                               *
 * Medialincs                                                                *
 *                                                                           *
 * cmyoo@medialincs.com                                                      *
 * cmyoo@lycos.co.kr                                                         *
 *                                                                           *
 *****************************************************************************

  MODULE NAME:  IPFE_ARP.C

  REVISION HISTORY:

  Date        Version   Name             Description
  ----------  -------   ---------------- ---------------------------------------
  11/01/2000  1.0A      ChagmoYoo(cmyoo) Created
 ...........................................................................

  DESCRIPTION:

  This module contains ARP methods

 ...........................................................................
*/


/** ********************************************************************* **
 ** includes
 ** ********************************************************************* **/
#include <socket.h>
#include <mib.h>

#include "gl_def.h"
#include "ipfe_def.h"
#include "ipfe_rt.h"


/** ********************************************************************* **
 ** defines
 ** ********************************************************************* **/
#define DBG_LEVEL_7       (DebugMsgControl[FROM_IPFE2] & LEVEL_7)
#define DBG_LEVEL_8       (DebugMsgControl[FROM_IPFE2] & LEVEL_8)
#define DBG_LEVEL_10      (DebugMsgControl[FROM_IPFE2] & LEVEL_10)
```

```
/** ********************************************************************** **
 ** typedefs
 ** ********************************************************************** **/
/* if_arp.h, if_ether.h in BSD */
__packed__ struct arphdr
{
#   define  ARP_REQUEST    1
#   define  ARP_REPLY      2
    uns16   ar_hrd;         /* format of hardware address */
    uns16   ar_pro;         /* format of protocol address */
    uns8    ar_hln;         /* length of hardware address */
    uns8    ar_pln;         /* length of protocol address */
    uns16   ar_op;          /* ARP/RARP operation */
};

__packed__ struct ether_arp
{
    struct arphdr  ea_hdr;
    uns8           arp_sha[6];    /* sender hardware address */
    uns32          arp_spa;       /* sender protocol address */
    uns8           arp_tha[6];    /* target hardware address */
    uns32          arp_tpa;       /* target protocol address */
};
#define arp_hrd    ea_hdr.ar_hrd
#define arp_pro    ea_hdr.ar_pro
#define arp_hln    ea_hdr.ar_hln
#define arp_pln    ea_hdr.ar_pln
#define arp_op     ea_hdr.ar_op


/** ********************************************************************** **
 ** globals
 ** ********************************************************************** **/
extern struct ni_table g_ni_tbl[NC_NNI];


/** ********************************************************************** **
 ** locals
 ** ********************************************************************** **/
static const uns8 c_broadcastaddr[] = {0xFF,0xFF,0xFF,0xFF,0xFF,0xFF};


/** ********************************************************************** **
 ** forward declarations
 ** ********************************************************************** **/


/*****************************************************************************
 *
 *  FUNCTIONS: arp_request
 *
 *  SYNOPSIS: Send an ARP request packet to dstip host
 *
 *  EXTERNAL EFFECTS: g_ni_tbl[]
 *
 *  PARAMETERS: ifnum - network interface number
 *              dstip - destination ip address to
 *
 *  RETURNS: n/a
 *
 *  ERRNO: n/a
 *
 *****************************************************************************/
void
arp_request(int ifnum, uns32 dstip)
{
```

```
#ifdef  IPFE2_SUPP
    union ifentry      ni;
    struct ether_arp   *ea;
    uns8               *data;
    uns32              srcip;
    uns8               srcmac[6];
    char               buf[64];

    if (get_netinfo(ifnum, &srcip, srcmac) != 1)
    {
        if (DBG_LEVEL_10)
            Print("$arp_request: get_netinfo() failed\n");
        return;
    }
    if (DBG_LEVEL_7)
        Print("@arp_request: [if%d] ip=%s\n", ifnum, inet_ntoa(dstip, buf));

    /* get buffer from transmit interface */
    ni.getpkb.count = sizeof(struct ether_arp);
    ni.getpkb.hwa_ptr = (char *)c_broadcastaddr;
    ni.getpkb.if_num = ifnum;
    data = (uns8 *)g_ni_tbl[ifnum].entry(GETPKB, &ni);
    if (data == (uns8 *)-1)
    {
        if (DBG_LEVEL_10)
            Print("$arp_request: getpkb(ifnum=%d) failed\n", ifnum);
        return;
    }
    memset(data, 0, 46);   /* 46 = minimum pdu size */
    ea = (struct ether_arp *)data;
    ea->arp_hrd = htons(1);               /* type of hardware address is Ethernet */
    ea->arp_pro = htons(0x0800);          /* type of protocol address is IP Address */
    ea->arp_hln = sizeof(ea->arp_sha);    /* hardware address length */
    ea->arp_pln = sizeof(ea->arp_spa);    /* protocol address length */
    ea->arp_op = htons(ARP_REQUEST);
    memcpy((char *)ea->arp_sha, (char *)srcmac, sizeof(ea->arp_sha));
    ea->arp_spa = srcip;
    memset((char *)ea->arp_tha, 0, sizeof(ea->arp_tha));
    ea->arp_tpa = dstip;
    if (DBG_LEVEL_8)
    {
        int     i;
        Print("\n[arp request]\n");
        Print("sender: %s-%x:%x:%x:%x:%x:%x\n",
              psys_inet_ntoa(ea->arp_spa, buf),
              ea->arp_sha[0], ea->arp_sha[1], ea->arp_sha[2],
              ea->arp_sha[3], ea->arp_sha[4], ea->arp_sha[5]);
        Print("target: %s-%x:%x:%x:%x:%x:%x",
              psys_inet_ntoa(ea->arp_tpa, buf),
              ea->arp_tha[0], ea->arp_tha[1], ea->arp_tha[2],
              ea->arp_tha[3], ea->arp_tha[4], ea->arp_tha[5]);
        for (i = 0; i < sizeof(struct ether_arp); i++)
        {
            if (!(i % 16))  Print("\n%04d : ", i);
            Print("%02X ", data[i]);
        }
        Print("\n");
    }
    /* transmit */
    ni.send.hwa_ptr = (char *)c_broadcastaddr;
    ni.send.buff_addr = data;
    ni.send.count = sizeof(struct ether_arp);
    ni.send.type = 0x0806;
    ni.send.if_num = ifnum;
    if (g_ni_tbl[ifnum].entry(SEND, &ni) != 0)
    {
        if (DBG_LEVEL_10)
```

```
            Print("$arp_request: send(ifnum=%d) failed\n", ifnum);
        return;
    }
#endif  IPFE2_SUPP
}   /* end of arp_request */


/* end of ipfe_arp.c */
```