ST-061793-09-R1
Sega Enterprises, Ltd.

# THE C LANGUAGE CODING STANDARD

PRELIMINARY DRAFT

REV. 0.11

JUNE 1 1993

HARDWARE R & D GROUP
SYSTEM SOFTWARE DIVISION



Translated by J.T. /August 11, 1993

(4) xxx.h header file

The header file corresponds to xxx.c code file and defines public routine prototype within xxx.c or data type.

(5) The use of space

- Do not insert space around binary operators such as "." and "->".

```
abc.xyz      abcPtr->xyz
```

- Insert one space after the reserved words.

```
for (;;)     switch (iSw)
```

- Do not insert space between () after the function call.

```
subFunction();
```

- Insert one space before the "," or ";".

```
function(para1, para2, para3);
for (n = 1; n < SIZE; n++)
```

- Insert one space before and after assignment operators and comparison operators.

```
a = b + c;
```

- Insert no space between unary operators "++", "--", and variables.

```
i++;      j--;
```

- Insert no space before ")]" and after "([" brackets.

```
x[a + c] = 3 + (y + z);
```

(6) Precedence and order of evaluation

Add () consciously when the precedence in the expression is not clear.

(7) Using sizeof operator

Set the size of the struct, union, and array with the sizeof operator.

(8) Control character

When using control characters in storing, etc., do not input the code directly. Use \t for tab and \33 for escape.

(9) Number of statements per line

Each line includes only one statement.
Change to a new line for the next statement, such as an if statement—and a null statement

```
if  (a  ==  b)
a  =  c;

for  (n  =0;  array[n]  !=  0;  n++)
     ;
```

(10) Do not evaluate numeric operations as logical operations

```
if  (a  +  b)          if  (! (a  +  b))          Incorrect
if  (a  +  b  != 0)    if  (a  +  b  ==  0)       Correct
```

(11) Tab size

Each tab size consists of 8 spaces.

(12) Control sentence format

Ranking by {} starts from the upper right and ends at the lower left. 4 spaces are set as 1 indent unit.

• if statement

```
If   (conditional statement)  {conditional statement)  { if   (conditional statement 1){
  statement;                    statement 1;                  statement 1;
}                               } else  {                } else  if  (conditional statement 2){
  statement;                    statement 2;                  statement 2;
                                }                               } else  {
                                                              statement 3;
                                                            }
```

• for statement

```
for  (statement 1;  statement 2;  statement 3) {
   statement;
}
```

• while statement

```
while (conditional statement) {
    statement;
}
```

• do statement

```
do  {
    statement;
} while  (conditional statement);
```

• switch statement

```
switch (statement)  {

case 0:   /*  comment   */
    statement 1;
    break;
default:  /*  comment   */
    statement 2;
    break;
}
```

```
switch (statement)  {

case 0:   /*  comment   */
      statement 1;
      break;
default:  /*  comment   */
      statement 2;
      break;
 }
```

(13) Comment

To exchange the source code internationally, comments are inserted in English. In principle, when using Japanese, EUC code is used with UNIX, and Shift-JIS is used with MS-DOS. However, this may change depending on the environment.

• Block comment

```
/*
*       Comment
*/
Code
```

align /* column position and */ column position.

• Statement comment

```
Statement      /*  Comment  */
```

- Comment to ignore part of the code

```
#if 0
    Code set
#endif
```

(14) Identifier

- Specify identifiers in 32 characters or less and describe them fully or near-fully spelled. For functions, use names in the form of verb+name which invoke the action. Markoff and punctuation using upper case letters and "-" should be consistent within the same project.

- Local variable name

  Upper case letters are used in variable names to separate single words.

```
localVariableName
local_variable_name
local_Variable_Name
```

- Global variable name or data type name

  Upper case letters are used in variable names to separate single words and as leading characters.

```
Sint16
GlobalVariableName
Global_variable_name
Global_Variable_Name
```

- Macro name

  Macro names are all in upper case letters and separated by "-".

```
MY_MACRO_NAME
```

(15) Restrictions

- Direct specification with binary operators such as (".", "," ,and "->") shall not exceed 4 steps.

```
val1Ptr->val2Ptr->val3Ptr->valName      is good!
```

- Do not write to constant data area.

```
str1Ptr = "abcdefg" ;
strcpy(str1Ptr, str2Ptr);
```

- When dividing a signed variable by a power of 2, the operation is not carried through the right shift. The MSB state of a signed variable during the right shift is not defined by ANSI-C.

- Do not nest the header file (include header within header).

## 3.2. Declaration

(1) Declaration of data type

- The leading character of the struct name shall be in the upper case, and when declaring the struct as typedef, the struct name shall be identical to the typedef name.

```
typedef struct StructName   {
    Sint16    var1;     /*  comment     */
    Sint8     var2;     /*  comment     */
}   StructName;
```

- Do not use the basic data type (char, int, short, etc.) prepared by the C language; use virtual basic data type (such as Sint32, Float 64) declared by xport.h.

(2) Constant data

- Main constant data is declared by define statement.

- Constant data to secure variable area and parameters for referencing only, are const declared.

```
const    Sint16    iConstData = 10;
char     *strcpy(char *,  const  char  *);
```

(3) Restrictions

- Do not use register variables.
- Do not use bit field definition.

## 4. Coding Example

The following keywords embedded within the source are to be used with the UNIX text revision control tools (Revision Control System or RCS) and data corresponding to each keyword within the text will be automatically embedded when checked out from the RCS.

| | | |
|---|---|---|
| $Author$ | = | Log in name of the user who stored revision |
| $Date$ | = | Date when revision was stored |
| $Header$ | = | RCS file name, revision number, date, author, standard |
| header | | which includes the status |
| $Locker$ | = | Log in name of the user who locked the revision |
| $Log$ | = | This is a log message during storage and is inserted in |
| sequence | | when updating revision |
| $Revision$ | = | Revision number assigned to revision |
| $Source$ | = | Full bus name for RCS file |
| $State$ | = | Status assigned to revision |

## 4.1 template.c code file

```
/*----------------------------------------------------------------------------
 * FILE: template.h
 *      $Author$
 *      $Date$
 *      $Locker$
 *      $Revision$
 *      $Source$
 *      $State$
 *
 *      Copyright (c) by SEGA Enterprises, Ltd. 1993. All right reserved.
 *
 * PURPOSE:
 *
 *          General description of program file
 *
 * DESCRIPTION:
 *
 *          How to use the module and detailed description of the data format
 *
 * INTERFACE:
 *
 *          The list and description of global variables and global functions
 *          defined in this file as an external interface
 *
 *      < GLOBAL VARIABLES LIST >
 *          PublicData1             -  description of PublicData1
```

```
*
*      < FUNCTIONS LIST >
*           main                    - main routine
*
*  CAVEATS:
*
*    For the description of remarks
*
*  AUTHOR(S):
*
*      List of author(s), modifier(s), etc.
*
*  MOD HISTORY:
*       $Log$
*
*-----------------------------------------------------------------------
*/

/*
 * C STANDARD LIBRARY FUNCTIONS/MACROS DEFINES
 */
#define  XPRINTF          -- Definitions of standard library usage routine names
#include "xstdlib.h"      -- Definitions of standard library

/*
 * C VIRTUAL TYPES DEFINITIONS
 */
#include "xport.h"        --  Definitions of virtual basic data types

/*
 * USER SUPPLIED INCLUDE FILES
 */
#include "template.h"          --  The definition of user include files

/*
 * GLOBAL DECLARATIONS
 */
Uint8 PublicData1 [5];                  /* Comment */

/*
 * LOCAL DEFINES/MACROS
 */
#define  MAX_NO   10                    /* Comment */
```

```
/*
 * STATIC DECLARATIONS
 */
static Uint8 *rcsid = "$Header: /home/ssdg/masat/psat/doc/RCS/cstand.tex,v 1.1 1993/05/29
```

Program header data from the Control System

```
Uint16 StaticData;                    /* Comment */

/*
 * STATIC FUNCTION PROTOTYPE DECLARATIONS
 */
static PROTOTYPE(Sint16, plus,  (Sint16, Sint16));
static PROTOTYPE(Sint16, minus, (Sint16, Sint16));

^L
/*****************************************************************************
 *
 * NAME:  main()  - main routine —  The routine names and overview
 *
 * PARAMETERS :                  ←  The description of parameters
 *     (1) Sint16    argc  - <i>   number of input command's argument.
 *     (2) Char     **argv  - <i>   argument string array.
 *
 * DESCRIPTION:
 *   The description of routine functions
 *
 * PRECONDITIONS:
 *   The description of any preconditions such as changes in the intput parameters,
 *   global variables, etc.. before the execution
 * POSTCONDITIONS:
 *   The description of any changes in the  return values, output parameters, global
 *   variables, etc..after the execution
 * CAVEATS:
 *   The description of remarks
 *
 *****************************************************************************
 */

Sint16
main(Sint16 argc, Char **argv)
{
    Sint16  i;      /* Comment */     —  Local variables
    Sint16  j;      /* Comment */        definition
```



Translated by J.T. /August 11, 1993

```
Sint16  k;      /* COMMENT */

/** BEGIN ***************************************************************/
printf("\nstart template\n");

/*
 *  process 1
 */
for (i = 0, j = MAX_NO; i < MAX_NO; i++, j--) {
    if ((i % 3) == 0) {
        k = plus(i, j);
        printf("i(%d) + j(%d) = %d\n", i, j, k);
    } else if ((i % 3) == 1) {
        k = minus(i, j);
        printf("i(%d) - j(%d) = %d\n", i, j, k);
    } else {
        k = minus(j, i);
        printf("j(%d) - i(%d) = %d\n", j, i, k);
    }
}

/*
 *  process 2
 */
i = 0;
j = MAX_NO;
while (i < MAX_NO) {
    switch (i % 3) {
        case 0:  /* COMMENT */
            k = plus(i, j);
            printf("i(%d) + j(%d) = %d\n", i, j, k);
            break;
        case 1:  /* COMMENT */
            k = minus(i, j);
            printf("i(%d) - j(%d) = %d\n", i, j, k);
            break;
        default: /* COMMENT */
            k = minus(j, i);
            printf("j(%d) - i(%d) = %d\n", j, i, k);
            break;
    }
    i++;
    j--;
```

```
    }

    printf("end template\n");
    return OK;
}

^L
/*****************************************************************************
 *
 * NAME:  plus()  - plus prosess
 *
 * PARAMETERS :
 *     (1) Sint16  a       - <i>   parameter 1.
 *     (2) Sint16  b       - <i>   parameter 2.
 *
 * DESCRIPTION:
 *     compute a + b and return result value.
 *
 * POSTCONDITIONS:
 *     (1) Sint16  result - return value.
 *
 *****************************************************************************
 */

static Sint16
plus(Sint16 a, Sint16 b)
{
    /** BEGIN ********************************************************************/
    return a + b;
}

^L
/*****************************************************************************
 *
 * NAME:  minus()  - minus prosess
 *
 * PARAMETERS :
 *     (1) Sint16  a       - <i>   parameter 1.
 *     (2) Sint16  b       - <i>   parameter 2.
 *
 * DESCRIPTION:
 *     compute a - b and return result value.
 *
```

```
 * POSTCONDITIONS:
 *     (1) Sint16  result - return value.
 *
 ********************************************************************************
 */

static Sint16
minus(Sint16 a, Sint16 b)
{
    /** BEGIN ***************************************************************/
    return a - b;
}
```

## 4.2 template.h Include file

```
/*-------------------------------------------------------------------------
 *  FILE: template.h
 *      $Author$
 *      $Date$
 *      $Locker$
 *      $Revision$
 *      $Source$
 *      $State$
 *
 *      Copyright (c) by SEGA Enterprises, Ltd. 1993. All right reserved.
 *
 *  PURPOSE:
 *
 *        General description of "include file".
 *
 *  DESCRIPTION:
 *
 *        How to use "Include file" and description of data format.
 *
 *
 *  AUTHOR(S):
 *
 *        List of authors, modifiers, etc.
 *
 *  MOD HISTORY:
 *      $Log$
 *
 *-------------------------------------------------------------------------
 */

#ifndef TEMPLATE_H
#define TEMPLATE_H          ← Setting include flags

/*
 * GLOBAL DEFINES/MACROS DEFINES          Setting a flag for the included file
 */
#define  TEMPLATE_DEF              1

/*
 * STRUCTURE DECLARATIONS
 */
struct NewStruct1 {  /* comment */
```

```
    Sint16  var1;     /* comment */
    Sint16  var2;     /* comment */
};

struct NewStruct2 {                     /* comment */
    struct  NewStruct1 *newStruct1;     /* comment */
    Sint16  var1;                       /* comment */
    Sint16  var2;                       /* comment */
};

/*
 * TYPEDEFS
 */
typedef struct NewStruct2 NewStruct2;

/*
 * EXTERNAL VARIABLE DECLARATIONS
 */

/*
 * EXTERNAL FUNCTION PROTOTYPE DECLARATIONS
 */
 extern PROTOTYPE(Sint16, main, (Sint16, Char**));

#endif /* ifndef TEMPLATE_H */
```

## 4.3 xstdlib.h standard library include file

```
/*-----------------------------------------------------------------------------
 *  FILE: xstdlig.h
 *      $Author$
 *      $Date$
 *      $Locker$
 *      $Revision$
 *      $Source$
 *      $State$
 *
 *      Copyright (c) by SEGA Enterprises, Ltd. 1993. All right reserved.
 *
 *  PURPOSE:
 *
 *      standard liblary header files for different C environment.
 *
 *  DESCRIPTION:
 *
 *      <Usage>   #define XPRINTF    <-- use function name is "printf".
 *                #define XSTRCPY
 *                    ...
 *                #include "xstdlib.h"
 *
 *  AUTHOR(S):
 *
 *      H.Endo   Hardware R&D Dept. System Software Section
 *
 *  MOD HISTORY:
 *      $Log$
 *
 *-----------------------------------------------------------------------------
 */

#ifdef  XPRINTF
    #define L_STDIO_H
#endif

#ifdef  XSTRCPY
    #define L_STRING_H
#endif

#ifdef L_STDIO_H
```

```
    #ifndef XSTDIO_H
        #define  XSTDIO_H
        #include <stdio.h>
    #endif
#endif

#ifdef L_STRING_H
    #ifndef XSTRING_H
        #define  XSTRING_H
        #include <string.h>
    #endif
#endif
```

## 4.4 xport.h virtual basic data type include file

```
/*----------------------------------------------------------------------
 *  FILE: xport.h
 *      $Author$
 *      $Date$
 *      $Locker$
 *      $Revision$
 *      $Source$
 *      $State$
 *
 *      Copyright (c) by SEGA Enterprises, Ltd. 1993. All right reserved.
 *
 *  PURPOSE:
 *
 *      C virtual types, grobal constant data and macros definitions.
 *
 *  AUTHOR(S):
 *
 *      H.Endo   Hardware R&D Dept. System Software Section
 *
 *  MOD HISTORY:
 *      $Log$
 *
 *----------------------------------------------------------------------
 */

#ifndef XPORT_H
#define XPORT_H

/*
 * C VIRTUAL TYPES DECLARATIONS
 */
typedef unsigned char       Uint8;
typedef char                Sint8;
typedef unsigned short      Uint16;
typedef short               Sint16;
typedef unsigned long       Uint32;
typedef long                Sint32;
typedef float               Float32;
typedef double              Float64;

typedef unsigned char       Byte;
```

```
typedef char                Char;
typedef int                 Int;
typedef long                Long;
typedef float               Float;
typedef double              Double;
typedef Sint32              Fixed32; /* Fixed-point representation        */
                                     /* Radix point is between bit 15 & 16 */


/*
 * CONSTANT DATA DECLARATIONS
 */
#define BOOLEAN             Uint8

#ifndef FALSE
#define FALSE               0
#endif

#ifndef TRUE
#define TRUE                1
#endif

#ifndef NULL
#define NULL                0
#endif

#ifndef NIL
#define NIL                 0
#endif

#ifndef VOID
#define VOID                void
#endif

#ifndef OK
#define OK                  0
#endif

#ifndef NG
#define NG                  -1
#endif

/*
 * MACROS DECLARATIONS
```

```
 */
#define PROTOTYPE(returnType, fName, pList) returnType fName pList

#define FIXED(x)                ((Fixed32)((x) * 65536.0))
#define INT_TO_FIXED(x)         ((Fixed32)((x) * 65536))
#define FIXED_TO_INT(x)         ((Sint32)((x) / 65536))
#define FIXED_TO_FLOAT(x)       ((Float32)((long)(x) / 65536.0))
#define MUL_FIXED(a, b)         (((a) / 256) * ((b) / 256))
#define DIV_FIXED(a, b)         ((a / ((b) / 256)) * 256)

#define ABS(x)                  (((x) < 0) ? -(x) : (x))
#define MAX(x, y)               (((x) < (y)) ? (y) : (x))
#define MIN(x, y)               (((x) < (y)) ? (x) : (y))

#endif /* ifndef XPORT_H */
```