Hitachi America, Ltd.

TN-0093

Application Engineering
TechNote

Jennifer Ediyanto

HAL/GNU Linker Command File

Hitachi America provides the HAL/GNU H8/300 Software Tools to program the Hitachi H8/300 series microcontrollers. These software tools consist of C compiler, assembler, linker, librarian, and debugger. This paper will provide some explanations about the linker command file.

In order to download a program to the H8/300 Series Evaluation Boards without using the HAL/GNU GDB Debugger, we need to compile the program into the Motorola S-record format absolute file. The following is the common-used linker command file to produce an S-record format file:

```
OUTPUT_ARCH(h8300)
OUTPUT_FORMAT(srec)

SECTIONS
{
     .text    0x8000: { *(.text) _etext = . ; }
     .data    .     : { *(.data) _edata = . ; }
     .bss     .     : { *(.bss)  *(COMMON) _end = . ; }
     .stack   0xf800: { *(.stack) }
}
```

We can name this linker command file as **Hitachi.cmd**. In Hitachi.cmd file, we use several linker commands, which are:

OUTPUT_ARCH	specify the target architecture, i.e., H8/300
OUTPUT_FORMAT	specify the output file format, i.e., srec
SECTIONS	control where input sections are placed into output sections

The most frequently used statement in the SECTIONS command is the section definition, which we can use to specify the properties of an output section: its location, alignment, contents, fill pattern, and target memory region. The above section definition uses the following syntax:

output_section start_address : {input_section}

where:

output_section	The output section names, i.e., text, data, bss, stack.
start_address	The location where we want the section to start. The dot (.) means to continue after the above section address ends.
input_section	The input section names, i.e., text, data, bss, stack. We can also specify the symbol names, which are the compiler generated symbols. The symbol name is always preceded by an underscore.

The following table shows the sections that are used by the compiler:

Name	Contents
text	program code
data	explicitly initialized variables in the program
bss	local common variable storage (variables that are used commonly by any functions) and uninitialized variables (variables that are not initialized in the program)
stack	stack

The following is a simple C program that prints 'Hitachi America #' message ten times to the screen:

```
#include <stdio.h>

main()
{
        int i;
        for (i = 0; i < 10; i++)
                iprintf("Hitachi America %d\n",i);
}
```

We can name this C program as **Hitachi.c**. Let us compile this Hitachi.c program into an S-record absolute file with the following command:

GCC -o Hitachi.x -O -Xlinker -Thitachi.cmd Hitachi.c

where

GCC	name of the compiler
-o	compiler switch that specifies output file name
-O	compiler switch that optimizes the source code
-Xlinker	compiler switch that passes option to the linker
-T	linker switch that directs the linker to read commands from Hitachi.cmd file

We can download the **Hitachi.x** file to the H8/300 series Evaluation Board by using any Terminal Emulation Program, i.e., Procomm, Crosstalk, Mirror.

HITACHI
Hitachi America, Ltd. • San Francisco Center • 2000 Sierra Point Parkway • Brisbane, CA 94005-1819 • (415) 589-8300