Table of Contents

[1. Introduction 4](#_Toc530588314)

[1.1 Notation of This Document 4](#_Toc530588315)

[1.1.1 Notational convention 4](#_Toc530588316)

[1.1.1.1 Notational convention of specifications not open to users 5](#_Toc530588317)

[2. About the Product 6](#_Toc530588318)

[2.1 Position of Product 6](#_Toc530588319)

[2.2 Outline 6](#_Toc530588320)

[2.3 Supported Environment 6](#_Toc530588321)

[3. Installation 7](#_Toc530588322)

[3.1 Installation 7](#_Toc530588323)

[3.2 Uninstallation 7](#_Toc530588324)

[4. Manipulation Method 8](#_Toc530588325)

[4.1 Command Input Format 8](#_Toc530588326)

[4.2 Output of Result 8](#_Toc530588327)

[4.3 Type and Function of Options 8](#_Toc530588328)

[4.3.1 List of Options 9](#_Toc530588329)

[4.3.2 Priority of Options 9](#_Toc530588330)

[4.3.3 Default Setting of Options for Checking the Area Size and Specifying the Start Address 10](#_Toc530588331)

[4.3.4 Details of Options 10](#_Toc530588332)

[4.3.4.1 Display of a Version 10](#_Toc530588333)

[4.3.4.2 Display of a Help 11](#_Toc530588334)

[4.3.4.3 Specification of a Library 12](#_Toc530588335)

[4.3.4.4 Specification of an Output File Name 15](#_Toc530588336)

[4.3.4.5 Specification of an Output File Format 16](#_Toc530588337)

[4.3.4.6 Check of the Size of a Program Area 17](#_Toc530588338)

[4.3.4.7 Check of the Size of a Uniform Area 18](#_Toc530588339)

[4.3.4.8 Specification of the Start Address of uniform 19](#_Toc530588340)

[4.3.4.9 Check of the Size of an lwm Area 20](#_Toc530588341)

[4.3.4.10 Specification of the Start Address of lwm (Non-common) 21](#_Toc530588342)

[4.3.4.11 Specification of the Start Address of lwm (Common) 22](#_Toc530588343)

[4.3.4.12 Check of the Size of a gwm Area 23](#_Toc530588344)

[4.3.4.13 Specification of the Start Address of gwm (Non-common) 24](#_Toc530588345)

[4.3.4.14 Specification of the Start Address of gwm (Common) 25](#_Toc530588346)

[4.3.4.15 Check of the Size of an impc Area 26](#_Toc530588347)

[4.3.4.16 Specification of the Start Address of impc (Non-common) 27](#_Toc530588348)

[4.3.4.17 Specification of the Start Address of impc (Common) 28](#_Toc530588349)

[4.3.4.18 Specification of the Output Destination Section of a Stack Area 29](#_Toc530588350)

[4.3.4.19 Specification of the Size of a Stack Area 30](#_Toc530588351)

[4.3.4.20 Specification of the Output of Debugging Information 31](#_Toc530588352)

[4.3.4.21 Create library file from object file 32](#_Toc530588353)

[4.3.4.22 Add Object File to existing Library File 33](#_Toc530588354)

[4.3.4.23 Delete Module from existing Library File 34](#_Toc530588355)

[4.3.4.24 Extract Module from existing Library File 35](#_Toc530588356)

[4.3.4.25 Replace Module to existing Library File 36](#_Toc530588357)

[5. Linking Sections 37](#_Toc530588358)

[5.1 Section Allocation Using the Option for Specifying the Start Address of Memory 37](#_Toc530588359)

[5.1.1 Sections Having the Same Name 37](#_Toc530588360)

[5.1.2 Sections of the Same Memory Type 39](#_Toc530588361)

[5.2 Supported Sections 41](#_Toc530588362)

[6. Special Symbols of Linker 42](#_Toc530588363)

[7. Generation of a Stack Area 43](#_Toc530588364)

[8. Linker Messages 48](#_Toc530588365)

[8.1 Message Format 48](#_Toc530588366)

[8.2 Messages 49](#_Toc530588367)

# Introduction

This document describes the linker system of ELF object format files for the CVengine compiler (CCIMP).

## Notation of This Document

### Notational convention

This document is written according to the following notational convention.

... The same format will continue.

[ ] The item in square brackets can be omitted.

{ } One of the items in curly brackets should be selected.

<Item> The item in angle brackets is replaced with a specific string (not enclosed in double quotation marks (" ")) corresponding to that item.1

Δ One or more spaces or tabs

" " Characters enclosed in double quotation marks (" ")

CR Carriage return

LF Line feed

HT Horizontal tab

Messages and strings should be enclosed with double quotation marks (" "). (Note that escape characters, such as \" or \\, should be added to characters that have a special meaning in a string, e.g. double quotation mark (") or backslash (\).)

The following mark is used to indicate undetermined items.

TBD

#### Notational convention of specifications not open to users

Specifications that are not disclosed to users are indicated with the following notation in this document. Note that the color and color name to be used must be among the colors and color names included in the color map of the drop-down list that appears when clicking the relevant toolbar button (Font Color button) of Microsoft Word 2007.

The notation is as follows:

Not open to users

For example, this line is a description for a specification not open to users.

The contents are written in a tabular form of only a single cell (without ruled lines). Not open to users is explicitly shown in red on the first line, and the reason for not being open to users (can be omitted) is enclosed in parentheses and in the color of 50% gray on the second line. Details of the specification not open to users are written from the third line. The table cell is shaded in 50% gray.

# About the Product

## Position of Product

This product is a tool for linking ELF object format files for the CVengine compiler (CCIMP).

## Outline

This product uses relocatable files in the ELF object format and library files as input, and outputs them in the load module file format and the library file format.

## Supported Environment

This tool operates in the following operating environment.

[Hardware environment]

Conforms to the operating environment of CS+.

[OS environment]

Conforms to the operating environment of CS+.

Not open to users

Wide characters

Operation in cases where wide characters are included in path and file names is supported in only a Windows environment.

Such operation is not supported in a Linux environment, which is not formally supported.

# Installation

## Installation

1. Place the execution module in a desired folder.
2. Set the path for that folder to environment variable PATH.

## Uninstallation

1. Delete the execution module.
2. Delete the path name set in environment variable PATH.

# Manipulation Method

## Command Input Format

The following should be entered on the command line.

command-name [Δinput-file-name or option]...

command-name: slink

option: Character string starting with -.

When more than one library specification option is specified, all the specified libraries are effective.

If any other option is specified more than once, the last specified option is effective. No message is output to indicate duplicate specification.

input-file-name: Character string not starting with a hyphen (-)\*.

\* Only relocatable files in the ELF object format created with sasm are supported as input files.

## Output of Result

The link result is output to the current folder.

The output file name has a format of "first-input-file-name.default-extension" where the extension of the first input file name is replaced with the default extension\* of the output file format.

However, it is possible to assign any file name by specifying a relevant option\*.

\* For details on the default extension and the relevant option specification, refer to section 4.3.4.4.

## Type and Function of Options

A list of options is shown in this section. The letters in options are case-sensitive. When an option not included in the list of options is specified, an error will occur.

<size> and <address> of an option must be specified in the format and range shown below:

[+] [0 [{ x | X }]] digits

* "digits" must be one or more numeric characters and alphabets ('a' to 'f' or 'A' to 'F').
* When the first character is '0' and the second character is other than 'x' or 'X', the character string is interpreted as an octal numeral.
* When the first character is '0' and the second character is 'x' or 'X', the character string is interpreted as a hexadecimal numeral.
* When the first character is in the range of 1 to 9, the character string is interpreted as a decimal numeral.
* A prefix of a positive sign (+) can be used.

Range of "digits" values: 0 to 0xfffffffe

### List of Options

Table 4.3.1 shows a list of options.

Table 4.3.1 List of Options

|  |  |  |
| --- | --- | --- |
| Classification | Option | Specification Format |
| Display of a version | -V | -V |
| Display of a help | -h | -h |
| Specification of a library | -l | -lΔfile01.a |
| Specification of an output file name | -o | -oΔresult.out |
| Specification of an output file format | -O | -OΔABS |
| Check of the size of a program area | -P | -PΔ65536 |
| Check of the size of a uniform area | -U | -UΔ64 |
| Specification of the start address of uniform | -D | -DΔ0xf0018000 |
| Check of the size of an lwm area | -W | -WΔ1024 |
| Specification of the start address of lwm (non-common) | -L | -LΔ0xf8000000 |
| Specification of the start address of lwm (common) | -X | -XΔ0xf0000000 |
| Check of the size of a gwm area | -G | -GΔ2048 |
| Specification of the start address of gwm (non-common) | -B | -BΔ0xf8008000 |
| Specification of the start address of gwm (common) | -Y | -YΔ0xf0008000 |
| Check of the size of an impc area | -S | -SΔ8192 |
| Specification of the start address of impc (non-common) | -C | -CΔ0xed800000 |
| Specification of the start address of impc (common) | -Z | -ZΔ0xed000000 |
| Specification of the output destination section of a stack area | -zΔstack-section | -zΔstack-section=lwm |
| Specification of the size of a stack area | -zΔstack-size | -zΔstack-size=32 |
| Specification of the output of debugging information | -d | -d |
| Create library file | -c create | -O LIB -c create sinf.o -oΔlib.a |
| Add object file to existing library file | -c add | -O LIB -c add tanf.o -oΔlib.a |
| Delete module from existing library file | -c delete | -O LIB -c delete sinf -oΔlib.a |
| Extract module from existing library file | -c extract | -O LIB -c extract sinf -oΔlib.a |
| Replace module in existing library file | -c replace | -O LIB -c replace sinf.o -oΔlib.a |

### Priority of Options

There is no priority for specifying options.

All options are logically ORed.

### Default Setting of Options for Checking the Area Size and Specifying the Start Address

Table 4.3.2 Settings when Options are omitted

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Memory Area | Option | Set Value for Checking the Size | | | |
| IMP-X5-V3M | IMP-X5-V3M2 | IMP-X5-V3H | IMP-X6-V3U |
| Program | -P | 65536 | 65536 | 65536 | 65536 |
| uniform | -U | 256 | 256 | 256 | 256 |
| lwm | -W | 16384 | 16384 | 16384 | 32768 |
| gwm | -G | 4096 | 4096 | 4096 | 8192 |
| impc | -S | 917504 | 917504 | 2097152 | 16777216 |
| Memory Area | Option | Set Value for Specifying the Start Address | | | |
| IMP-X5-V3M | IMP-X5-V3M2 | IMP-X5-V3H | IMP-X6-V3U |
| uniform | -D | 0xf0018000 | 0xf0018000 | 0xf0018000 | 0xf0018000 |
| lwm (non-common) | -L | 0xf8000000 | 0xf8000000 | 0xf8000000 | 0xf8000000 |
| lwm (common) | -X | 0xf0000000 | 0xf0000000 | 0xf0000000 | 0xf0000000 |
| gwm (non-common) | -B | 0xf8008000 | 0xf8008000 | 0xf8008000 | 0xf8008000 |
| gwm (common) | -Y | 0xf0008000 | 0xf0008000 | 0xf0008000 | 0xf0008000 |
| impc (non-common) | -C | 0xed800000 | 0xed800000 | 0xed800000 | 0xed800000 |
| impc (common) | -Z | 0xed000000 | 0xed000000 | 0xed000000 | 0xee000000 |

### Details of Options

#### Display of a Version

[Option]

-V

[Function]

Outputs the version information of slink to the standard error output and terminates processing.

[Format]

-V

[Interpretation when omitted]

The version information of slink is not displayed.

[Description]

The version information of slink is output to the standard error output.

An output example is shown below:

Shader Linker E0.00.00a [dd MMM yyyy]

Copyright (C) 2016 Renesas Electronics Corporation

[Relationship with other options]

When the -h option is specified earlier than this option, this option becomes invalid.

#### Display of a Help

[Option]

-h

[Function]

Outputs descriptions of the slink option (including version information) to the standard error output and terminates processing.

[Format]

-h

[Interpretation when omitted]

Descriptions of the slink option are not displayed.

However, if no option and no input file are specified, the operation is the same as that when the -h option is specified.

[Description]

Descriptions of the slink option are output to the standard error output.

An output example is shown below:

Shader Linker E0.00.00a [dd MMM yyyy]

Copyright (C) 2016 Renesas Electronics Corporation

-------------------------------------------------------------------------------

Command Line Format

slink [option] object-filename

Option Function

-V Display slink version number.

-h Display this message.

-l <filename> Specify input library file name.

-o <filename> Specify output file name.

-O {ABS|LIB} Specify output file format.

-P <size> Specify value of program check size.

-U <size> Specify value of uniform check size.

-D <address> Specify uniform address.

-W <size> Specify value of lwm check size.

-L <address> Specify lwm address (non-common).

-X <address> Specify lwm address (common).

-G <size> Specify value of gwm check size.

-B <address> Specify gwm address (non-common).

-Y <address> Specify gwm address (common).

-S <size> Specify value of impc check size.

-C <address> Specify impc address (non-common).

-Z <address> Specify impc address (common).

-z stack-section={impc|lwm} Specify stack area section.

-z stack-size=<size> Specify stack area size.

-d Outputs debugging information.

-c create Create library file from object file.

-c add Add object file to existing library file.

-c delete Delete module from existing library file.

-c extract Extract module from existing library file.

-c replace Replace module to existing library file.

-------------------------------------------------------------------------------

[Relationship with other options]

When the -V option is specified earlier than this option, this option becomes invalid.

#### Specification of a Library

[Option]

-l

[Function]

Specifies an input library file.

[Format]

-lΔfilename

[Interpretation when omitted]

None

[Description]

* Specify an input library file.
* After linkage processing among files specified as input files, this option searches the library file for undefined symbols.
* Search for symbols in the library file is performed in the order specified in this option.
* A module having a symbol which was firstly found by symbol search in a library file is loaded as an input module.

An example of symbol search is provided below:

|  |  |  |  |
| --- | --- | --- | --- |
| # sasm -c V3M2 -T REL main.s  .EXTERN \_a  .EXTERN \_b  .SECTION .text  .DCW \_a  .DCW \_b | # sasm -c V3M2 -T REL lwm.s  # slink -O LIB lwm.o  .PUBLIC \_a  .PUBLIC \_b  .SECTION lwm  \_a:  \_b: | # sasm -c V3M2 -T REL gwm.s  # slink -O LIB gwm.o  .PUBLIC \_a  .SECTION gwm  \_a: | # sasm -c V3M2 -T REL impc.s  .PUBLIC \_a  .PUBLIC \_b  .SECTION impc  \_a:  \_b: |

Example 1: A case where a module in a library file is loaded (1)

slink main.o -l lwm.a -l gwm.a -o result01.out

The lwm module in the lwm.a library is loaded and definition reference relationship of \_a and \_b is solved.

helfdmp -T -s result01.out

------------------------------------------------------------------------------

[Symbol Table] ".symtab"

Index Info Value Size Section/Name

0. l- 0x00000000 0x00000000 <UNDEF>

1. l-file 0x00000000 0x00000000 <ABS>/lwm.s

2. l-file 0x00000000 0x00000000 <ABS>/main.s

3. l-sect 0x00000000 0x00000000 .text/.text

4. l-sect 0xf8000000 0x00000000 lwm/lwm

5. g- 0xf8000000 0x00000000 lwm/\_a

6. g- 0xf8000000 0x00000000 lwm/\_b

------------------------------------------------------------------------------

Example 2: A case where a module in a library file is loaded (2)

slink main.o -l gwm.a -l lwm.a -o result02.out

The gwm module in the gwm.a library is loaded and definition reference relationship of \_a is solved.

The lwm module in the lwm.a library is loaded and definition reference relationship of \_b is solved.

Since the lwm module is loaded and the definition of \_a is overlapped with that of a global symbol, \_a of the lwm module loaded later becomes a local symbol.

helfdmp -T -s result02.out

------------------------------------------------------------------------------

[Symbol Table] ".symtab"

Index Info Value Size Section/Name

0. l- 0x00000000 0x00000000 <UNDEF>

1. l-file 0x00000000 0x00000000 <ABS>/gwm.s

2. l-file 0x00000000 0x00000000 <ABS>/lwm.s

3. l-file 0x00000000 0x00000000 <ABS>/main.s

4. l-sect 0x00000000 0x00000000 .text/.text

5. l-sect 0xf8000000 0x00000000 lwm/lwm

6. l-sect 0xf8008000 0x00000000 gwm/gwm

7. l- 0xf8000000 0x00000000 lwm/\_a

8. g- 0xf8008000 0x00000000 gwm/\_a

9. g- 0xf8000000 0x00000000 lwm/\_b

------------------------------------------------------------------------------

Example 3: Another case where no module in a library file is loaded

slink main.o -l gwm.a -l lwm.a impc.o -o result03.out

Since the definition reference relationship of \_a and \_b is solved between input files of main.o and impc.o, no module in the library file is loaded.

helfdmp -T -s result03.out

------------------------------------------------------------------------------

[Symbol Table] ".symtab"

Index Info Value Size Section/Name

0. l- 0x00000000 0x00000000 <UNDEF>

1. l-file 0x00000000 0x00000000 <ABS>/impc.s

2. l-file 0x00000000 0x00000000 <ABS>/main.s

3. l-sect 0x00000000 0x00000000 .text/.text

4. l-sect 0xed800000 0x00000000 impc/impc

5. g- 0xed800000 0x00000000 impc/\_a

6. g- 0xed800000 0x00000000 impc/\_b

------------------------------------------------------------------------------

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of an Output File Name

[Option]

-o

[Function]

Specifies an output file name for the link result.

[Format]

-oΔfilename

[Interpretation when omitted]

The file is output to the current folder.

The output file is assigned a name in the "first-input-file-name.default-extension" format.

The default extensions are as shown below:

* When the -O ABS option is specified: out
* When the -O LIB option is specified: a

[Description]

An output destination file for the link result is specified.

[Relationship with other options]

When the -V or -h option is specified, this option becomes invalid.

#### Specification of an Output File Format

[Option]

-O

[Function]

Specifies an output file format.

[Format]

-OΔ{format}

format: ABS|LIB

[Interpretation when omitted]

The operation is the same as that when the -O ABS option is specified.

[Description]

The output file format "format" is specified.

Table 4.3.3 shows available formats.

Table 4.3.3 Output File Format

|  |  |
| --- | --- |
| format | Description |
| ABS | A load module file is output. |
| LIB | A library file is output. |

[Relationship with other options]

When the -V or -h option is specified, this option becomes invalid.

#### Check of the Size of a Program Area

[Option]

-P

[Function]

Specifies the size of a program memory area to be checked.

[Format]

-PΔ<size>

Unit: 32 bits (word size)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

When the -O ABS option is specified, the following process takes place regardless of specification of this option.

* The linker checks the memory size to see whether the program exceeds the specified size of the memory area.

If it exceeds the specified size, the linker outputs an error and terminates processing.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Check of the Size of a Uniform Area

[Option]

-U

[Function]

Specifies the size of a uniform memory area to be checked.

[Format]

-UΔ<size>

Unit: 32 bits (word size)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

When the -O ABS option is specified, the following process takes place regardless of specification of this option.

* The linker checks the memory size to see whether uniform exceeds the specified size of the memory area.

If it exceeds the specified size, the linker outputs an error and terminates processing.

* The linker checks to see whether the result exceeds the 32-bit memory address range when the size of uniform is "start address + memory area ". If it exceeds the specified address range, the linker outputs an error and terminates processing.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of uniform

[Option]

-D

[Function]

Specifies the start address of the uniform memory area.

[Format]

-DΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the uniform memory area is specified.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Check of the Size of an lwm Area

[Option]

-W

[Function]

Specifies the size of an lwm memory area to be checked.

[Format]

-WΔ<size>

Unit: 8 bits (byte size)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

When the -O ABS option is specified, the following process takes place regardless of specification of this option.

* The linker checks the memory size to see whether lwm exceeds the specified size of the memory area.

If it exceeds the specified size, the linker outputs an error and terminates processing.

* The linker checks whether the result exceeds the 32-bit memory address range when the size of lwm is "start address + memory area". If it exceeds the specified address range, the linker outputs an error and terminates processing.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of lwm (Non-common)

[Option]

-L

[Function]

Specifies the start address of the lwm (non-common) memory area.

[Format]

-LΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the lwm (non-common) memory area is specified.

If the lwm\_comm section exists in the input file, an address which has been automatically adjusted according to this address or the lwm\_comm section size is specified for the start address of the lwm section.

Note: For details on automatic adjustment of an address, refer to section 5.1.2, Sections of the Same Memory Type.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of lwm (Common)

[Option]

-X

[Function]

Specifies the start address of the lwm (common) memory area.

[Format]

-XΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the lwm (common) memory area is specified.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Check of the Size of a gwm Area

[Option]

-G

[Function]

Specifies the size of a gwm memory area to be checked.

[Format]

-GΔ<size>

Unit: 8 bits (byte size)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

When the -O ABS option is specified, the following process takes place regardless of specification of this option.

* The linker checks the memory size to see whether gwm exceeds the specified size of the memory area.

If it exceeds the specified size, the linker outputs an error and terminates processing.

* The linker checks whether the result exceeds the 32-bit memory address range when the size of gwm is "start address + memory area". If it exceeds the specified address range, the linker outputs an error and terminates processing.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of gwm (Non-common)

[Option]

-B

[Function]

Specifies the start address of the gwm (non-common) memory area.

[Format]

-BΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the gwm (non-common) memory area is specified.

If the gwm\_comm section exists in the input file, an address which has been automatically adjusted according to this address or the gwm\_comm section size is specified for the start address of the gwm section.

Note: For details on automatic adjustment of an address, refer to section 5.1.2, Sections of the Same Memory Type.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of gwm (Common)

[Option]

-Y

[Function]

Specifies the start address of the gwm (common) memory area.

[Format]

-YΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the gwm (common) memory area is specified.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Check of the Size of an impc Area

[Option]

-S

[Function]

Specifies the size of an impc memory area to be checked.

[Format]

-SΔ<size>

Unit: 8 bits (byte size)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

When the -O ABS option is specified, the following process takes place regardless of specification of this option.

* The linker checks the memory size to see whether impc exceeds the specified size of the memory area.

If it exceeds the specified size, the linker outputs an error and terminates processing.

* The linker checks whether the result exceeds the 32-bit memory address range when the impc is "start address + memory area". If it exceeds the specified address range, the linker outputs an error and terminates processing.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of impc (Non-common)

[Option]

-C

[Function]

Specifies the start address of the impc (non-common) memory area.

[Format]

-CΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the impc (non-common) memory area is specified.

If the impc\_comm section exists in the input file, an address which has been automatically adjusted according to this address or the impc\_comm section size is specified for the start address of the impc section.

Note: For details on automatic adjustment of an address, refer to section 5.1.2, Sections of the Same Memory Type.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Start Address of impc (Common)

[Option]

-Z

[Function]

Specifies the start address of the impc (common) memory area.

[Format]

-ZΔ<address >

Unit: 8 bits (byte address)

[Interpretation when omitted]

The setting provided in Table 4.3.2 is used.

[Description]

The start address of the impc (common) memory area is specified.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Output Destination Section of a Stack Area

[Option]

-zΔstack-section

[Function]

Specifies the output destination section of a stack area.

[Format]

-zΔstack-section={impc|lwm}

[Interpretation when omitted]

The operation is the same as that when the -z stack-section=impc option is specified.

[Description]

* "impc" indicates impc (non-common), and "lwm" indicates lwm (non-common).
* The output destination section of the stack area is generated as a section belonging to "slink\_generates".
* For details on the stack area, refer to chapter 7, Generation of a Stack Area.

Note: "slink\_generates" is a dummy file name used within the linker.

* The alignment value of the section to be created for a stack differs depending on the core that is used. If the core is IMP-X5-V3M, the alignment value is fixed to 4\*. If the core is IMP-X5-V3M2, IMP-X5-V3H, V3U or later, the alignment value is fixed to 8\*.

However, if a section having the same name as the stack section exists in the input file (object and library), the maximum alignment value of these sections is used as the alignment value of the output file section.

\* The alignment value of the stack area in the section is the fixed value shown above.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.
* The -S and -C options enable the impc section created by this option to be aligned at the specified address and be subject to size check.
* The -W and -L options enable the lwm section created by this option to be aligned at the specified address and be subject to size check.

#### Specification of the Size of a Stack Area

[Option]

-zΔstack-size

[Function]

Specifies the size of a stack area.

[Format]

-zΔstack-size=<size>

Unit: 8 bits (byte size)

[Interpretation when omitted]

* If the .STACKSIZE local label exists in the input file, the maximum value is obtained from the .STACKSIZE symbol sizes of all files and the stack area is generated using the maximum value as "stack area size".
* If the .STACKSIZE local label does not exist in the input file, the stack area is not generated.

[Description]

* When both the .STACKSIZE local label and the -z stack-size option exist, the value specified by the option takes precedence.
* <size> must be 0 or a multiple of 4.
* When 0 is specified for <size>, a stack area is not generated.
* When other than 0 is specified for <size>, the stack area is generated using the specified value as "stack area size".
* For details on the stack area, refer to chapter 7, Generation of a Stack Area.

[Relationship with other options]

* When the -O LIB option is specified, this option outputs an error and terminates processing.
* When the -V or -h option is specified, this option becomes invalid.

#### Specification of the Output of Debugging Information

[Option]

-d

[Function]

Enables the output of debugging information.

[Format]

-d

[Interpretation when omitted]

The output of debugging information is disabled.

[Description]

If the input file contains debugging information, the linker links debugging information of the input file and outputs debugging information to the output file.

The linker does not edit (generate, delete, change, etc.) debugging information.

Even if the input file does not contain debugging information, no message is output.

[Relationship with other options]

* When the -O LIB option is specified, the specification of this option is ignored.
* When the -V or -h option is specified, this option becomes invalid.

#### Create library file from object file

[Option]

-cΔcreate

[Function]

Create a library file from object file.

[Format]

-cΔcreateΔ<filename>

[Interpretation when omitted]

* When LIB is not specified at -O, ignore this option.
* When LIB is specified at -O, it is assumed that -cΔcreate is specified.

[Description]

A new library file is output from linked object files.

[Relationship with other options]

* When other than LIB is specified at -O, the error F0561009 will be output.
* When the option -o is specified, specifies an output file name for the linked result.
* When the other option is specified, the error F0561009 will be output.

An example of creating library is provided below:

|  |  |
| --- | --- |
| Object file sinf.o and cosf.o are available.  # slink -O LIB -c create sinf.o cosf.o  → sinf.a will be created. | Object file sinf.o and cosf.o are available.  # slink -O LIB -o lib.a -c create sinf.o cosf.o  → lib.a will be created. |

#### Add Object File to existing Library File

[Option]

-cΔadd

[Function]

Add object file to existing library file.

[Format]

-cΔaddΔ<filename>

[Interpretation when omitted]

None.

[Description]

A new library file is created from adding object file to existing library file.

[Relationship with other options]

* When other than LIB is specified at -O, the error F0561009 will be output.
* When the option -o is not specified, the error F0561021 will be output.
* When the other option is specified, this option outputs an error F0561009 and terminates processing.

An example of adding object file to existing library file is provided below:

|  |  |
| --- | --- |
| Object file sinf.o and library file lib.a are available.  # slink -O LIB -c add sinf.o -o lib.a  → sinf.o will be added to lib.a | Object file tanf.o, cosf.o and library file lib.a are available.  # slink -O LIB -c add tanf.o -c add cosf.o -o lib.a  → cosf.o and tanf.o will be added to lib.a |

#### Delete Module from existing Library File

[Option]

-cΔdelete

[Function]

Delete module from existing library file.

[Format]

-cΔdeleteΔ<modulename>

[Interpretation when omitted]

None.

[Description]

A new library file is created from deleting object file from existing library file.

[Relationship with other options]

* When the <module name> is not specified, the error F0561009 will be output.
* When the option -o is not specified, the error F0561021 will be output.
* When the other option is specified, this option outputs an error F0561009 and terminates processing.

An example of deleting object file from existing library file is provided below:

|  |  |
| --- | --- |
| Library file lib.a that containing module “sinf” is available.  # slink -O LIB -c delete sinf -o lib.a  → Module “sinf” will be deleted from lib.a. | Library file lib.a that containing module “sinf” and “cosf” is available.  # slink -O LIB -c delete sinf -c delete cosf -o lib.a  → Module “sinf” and “cosf” will be deleted from lib.a. |

#### Extract Module from existing Library File

[Option]

-cΔextract

[Function]

Extract module from existing library file.

[Format]

-cΔextractΔ<modulename>

[Interpretation when omitted]

None.

[Description]

A new object file is created from extracting module from existing library file.

[Relationship with other options]

* When other than LIB is specified at -O, the error F0561009 will be output.
* When the option -o is not specified, the error F0561021 will be output.
* When the other option is specified, this option outputs an error F0561009 and terminates processing.

An example of extracting object file from existing library file is provided below:

|  |  |
| --- | --- |
| Library file lib.a that containing module “sinf” is available.  # slink -O LIB -c extract sinf -o lib.a  → sinf.o that contain “sinf” module will be extracted from lib.a and lib.a does not change. | Library file lib.a that containing module “sinf” and “cosf” is available.  # slink -O LIB –c extract sinf -c extract cosf -o lib.a  → sinf.o that contain “sinf” module will be extracted from lib.a and lib.a does not change.  → cosf.o that contain “cosf” module will be extracted from lib.a and lib.a does not change. |

#### Replace Module to existing Library File

[Option]

-cΔreplace

[Function]

Replace module to existing library file.

[Format]

-cΔreplaceΔ<modulename>

[Interpretation when omitted]

None.

[Description]

A new library file is created from replacing module to existing library file.

[Relationship with other options]

* When other than LIB is specified at -O, the error F0561009 will be output.
* When the option -o is not specified, the error F0561021 will be output.
* When the other option is specified, this option outputs an error F0561009 and terminates processing.

An example of replacing object file to existing library file is provided below:

|  |  |
| --- | --- |
| Library file lib.a that containing module “sinf” is available.  Object file sinf.o that containing module “sinf” is available.  # slink -O LIB -c replace sinf.o -o lib.a  → sinf.o will be replaced module “sinf” in lib.a. | Library file lib.a that containing module “sinf” module “cosf” is available.  Object file sinf.o that containing module “sinf” is available  Object file cosf.o that containing module “cosf” is available.  # slink -O LIB -c replace sinf.o -c replace cosf.o -o lib.a  → sinf.o will be replaced module “sinf” in lib.a.  → cosf.o will be replaced module “cosf” in lib.a. |

# Linking Sections

This product links the same sections in both the input relocatable file and the library file, and allocates the linked result at the address specified with the option for specifying the start address of memory.

At the time of allocation, alignment is adjusted for each section according to the alignment value specified in the section header of the ELF file.

However, since common and non-common sections are allocated together with each other ultimately, the larger one of their alignment values is set for both sections.

In this product, even if the section size is 0, alignment is adjusted for each section.

## Section Allocation Using the Option for Specifying the Start Address of Memory

### Sections Having the Same Name

Files having the same name are allocated to memory sequentially in the order of input.

The input orders of files are (1) specified input files and (2) modules to be loaded in the input library file.

For details on loading modules in the library file, refer to section 4.3.4.3, Specification of a Library.

Note: In the example shown below, all modules are loaded in the order of specified input library files.

If load modules of the sections with the same name have different alignment values respectively, sections for respective modules are allocated with the alignment value of each load module. The alignment value of the sections on the output file (ABS file), however, is set to the maximum alignment value among the sections with the same name.

Example:

"lib2.a"

"file1.o"

|  |
| --- |
| Section lwm\_comm |
| Section gwm\_comm |
| Section impc\_comm |

"file2.o"

|  |
| --- |
| Section uniform |
| Section gwm\_comm |

"lib1.a"

Module 1

|  |
| --- |
| Section lwm\_comm |
| Section gwm\_comm |

Module 2

|  |
| --- |
| Section uniform |

Options specified at linkage

|  |
| --- |
| slink file1.o -l lib1.a file2.o -l lib2.a -D 0xf0018000 -X 0xf0000000 -Y 0xf0008000 -Z 0xed000000 |

Module 3

|  |
| --- |
| Section lwm\_comm |

"file1.out"

|  |  |  |
| --- | --- | --- |
| 0xf0000000 | file1 | Section  lwm\_comm |
|  | lib1 (module 1) |
|  | lib2 (module 3) |

|  |  |  |
| --- | --- | --- |
| 0xf0008000 | file1 | Section  gwm\_comm |
|  | file2 |
|  | lib1 (module 1) |

|  |  |  |
| --- | --- | --- |
| 0xf0018000 | file2 | Section  uniform |
|  | lib1 (module 2) |

|  |  |  |
| --- | --- | --- |
| 0xed000000 | file1 | Section  uniform |

### Sections of the Same Memory Type

Common and non-common sections are available for the lwm, gwm, and impc memories.

When allocating the common and non-common sections, the former takes precedence and they are allocated on the memory sequentially. At that time, the memory start address of the non-common section is adjusted automatically. A gap\* generated between common and non-common sections is padded with 0.

Table 5.1.1 shows the memory types.

Table 5.1.1 Memory Type

|  |  |  |
| --- | --- | --- |
| Memory Type | Common Section Name | Non-common Section Name |
| lwm | lwm\_comm | lwm |
| gwm | gwm\_comm | gwm |
| impc | impc\_comm | impc |

Example:

"fileA.o"

|  |  |
| --- | --- |
| Section name | Section size |
| lwm\_comm | 0x00000004 |
| lwm | 0x00000020 |
| gwm | 0x00000080 |
| impc\_comm | 0x00000004 |

"fileB.o"

|  |  |
| --- | --- |
| Section name | Section size |
| lwm\_comm | 0x00000004 |
| lwm | 0x00000020 |
| gwm\_comm | 0x00000004 |
| impc\_comm | 0x00000080 |
| impc | 0x00000080 |

"fileA.out"

|  |  |  |  |
| --- | --- | --- | --- |
|  | File | Section | Size |

|  |  |  |  |
| --- | --- | --- | --- |
| 0xf0000000 | fileA | lwm\_comm | 0x00000008 |
|  | fileB |
|  | Gap\* | 0x00000018 |

|  |  |  |  |
| --- | --- | --- | --- |
| 0xf8000004 | fileA | lwm | 0x00000040 |
|  | fileB |

|  |  |  |  |
| --- | --- | --- | --- |
| 0xf0008000 | fileB | gwm\_comm | 0x00000004 |
|  | Gap\* | 0x0000007c |

|  |  |  |  |
| --- | --- | --- | --- |
| 0xf8008004 | fileA | gwm | 0x00000080 |

|  |  |  |  |
| --- | --- | --- | --- |
| 0xed000000 | fileA | impc\_comm | 0x00000084 |
|  | fileB |
|  | Gap\* | 0x0000007c |

|  |  |  |  |
| --- | --- | --- | --- |
| 0xed800008 | fileB | impc | 0x00000080 |

Options specified at linkage

|  |
| --- |
| slink fileA.o fileB.o -L 0xf8000000 -B 0xf8008000 -C 0xed800000 -X 0xf0000000 -Y 0xf0008000 -Z 0xed000000 |

Automatic adjustment of the memory start address of the non-common section aligns the common section size by 32-byte alignment for lwm and by 128-byte alignment for gwm and impc, divides the number of threads with 8 threads for lwm or 32 threads for gwm and impc, and obtains the address by adding the resulting value to the memory start address specified by the option.

In case of impc:

(1) Adjusts the alignment of the impc\_comm section size 0x00000084 to 128-byte alignment: 0x00000100

(2) Divides the aligned size 0x00000100 by 32 threads: 0x00000008

(3) Adds 0x00000008 resulted in (2) to address 0xed800000 specified by the -C option: 0xed800008

Note: For details on common and non-common sections and threads, refer to "Assembler System Specifications for the CVengine Compiler (CCIMP)".

## Supported Sections

Table 5.2.1 shows the supported sections.

If any other section than those listed in the table is input, errors are output and processing is terminated.

Note that the section alignment value is not checked.

Table 5.2.1 Supported Sections

|  |  |  |  |
| --- | --- | --- | --- |
| Section Name | Section Type  (sh\_type) | Section Attribute  (sh\_flags) | Section Alignment  (sh\_addralign) |
| "" | SHT\_NULL | 0 | 0 |
| ".symtab" | SHT\_SYMTAB | 0 | 4 |
| ".strtab" | SHT\_STRTAB | 0 | 1 |
| ".shstrtab" | SHT\_STRTAB | 0 | 1 |
| ".rela"\* | SHT\_RELA | 0 | 4 |
| "uniform" | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| ".text" | SHT\_PROGBITS | SHF\_ALLOC | SHF\_EXECINSTR | Variable |
| lwm\_comm | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| "lwm" | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| gwm\_comm | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| "gwm" | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| "impc\_comm" | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| "impc" | SHT\_PROGBITS | SHF\_ALLOC | Variable |
| ".debug\_info" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_abbrev" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_line" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_loc" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_pubnames" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_aranges" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_pubtypes" | SHT\_PROGBITS | 0 | 1 |
| ".debug\_frame" | SHT\_PROGBITS | 0 | 4 |

\* When the section name starts with ".rela".

# Special Symbols of Linker

Table 6.1 and Table 6.2 show the special symbols and their details.

Table 6.1 List of Special Symbols

|  |  |  |
| --- | --- | --- |
| Symbol Name | Related Function | Description |
| .STACKSIZE | Generation of a stack area | Internal symbol (local undefined label) between the assembler and the linker created by the assembler to inform the linker of the value specified by #pragma stacksize. |
| \_\_\_stack | Generation of a stack area | Global definition label that indicates the start address of the stack area created by the linker at generation of the stack area. |
| \_\_\_stackEnd | Generation of a stack area | Global definition label that indicates the last address of the stack area created by the linker at generation of the stack area. |

Table 6.2 Details of Special Symbols

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Symbol Name | Associated Section Name | Symbol Attribute  (st\_info) | Symbol Address  (st\_value) | Symbol Size  (st\_size) |
| .STACKSIZE | None | STB\_LOCAL | STT\_NOTYPE | 0 | Value specified by #pragma stacksize\* |
| \_\_\_stack | lwm or impc | STB\_GLOBAL | STT\_NOTYPE | Start address of the stack area | 0 |
| \_\_\_stackEnd | lwm or impc | STB\_GLOBAL | STT\_NOTYPE | Last address of the stack area + 1 | 0 |

\* As a value of #pragma stacksize, specify 4 or a multiple of 4.

# Generation of a Stack Area

When the .STACKSIZE local label exists in the input file, or when the -z stack-size option specifies other than 0, the linker generates a stack area of the specified size in the output destination section specified by the -z stack-section option.

When both the .STACKSIZE local label and the -z stack-size option exist, the value specified by the option takes precedence.

The generated stack area is padded with 0s. At the start and last addresses of the stack area, the global definition label \_\_\_stack and \_\_\_stackEnd are created, respectively.

Since .STACKSIZE is an internal symbol, it is not output to the load module file.

If the input file contains a section (impc (non-common) or lwm (non-common)) having the same name as the output destination section of the stack area, the section is combined at the end of the section with the same name in the input file.

Example 1: When .STACKSIZE exists in the input file with the -z stack-size option not specified:

* The maximum value is obtained from the .STACKSIZE symbol sizes of all files and the stack area is generated using the maximum value as "stack area size".

<Example of operation>

// testA.c

extern int func(int x, int y);

extern int c;

#pragma stacksize 64

#pragma entry main master

const int a=1,b=2;

void main(void)

{

c = func(a, b);

}

// testB.c

#pragma stacksize 256

int func(int x, int y)

{

return x + y;

}

// testC.c

#pragma section bss lwm

int c;

ccimp -S testA.c testB.c testC.c

Options specified at compilation

# testA.s

.section .text

.public \_main

\_main:

.entry \_main, master

movi r2, \_\_\_stackEnd - 0x00000000

sethi r2, \_\_\_stackEnd - 0x00000000

movu r17, \_b

movu r16, \_a

call r31, \_func

movu.i r17, \_c

sti r16, r17, 0x00000000

trap

.section uniform

.public \_a

\_a:

.dcw 0x00000001

.public \_b

\_b:

.dcw 0x00000002

.section impc

\_\_\_stack:

.ds 16

\_\_\_stackEnd:

# testB.s

.section .text

.public \_func

\_func:

add r16, r16, r17

brar r31

.section impc

\_\_\_stack:

.ds 64

\_\_\_stackEnd:

# testC.s

.section lwm

.align 4

.public \_c

\_c:

.dsb 4

.section impc

\_\_\_stack:

.ds 32

\_\_\_stackEnd:

Options specified at assembly

sasm -c V3M2 -T REL testA.s

sasm -c V3M2 -T REL testB.s

sasm -c V3M2 -T REL testC.s

\* To be continued on the next page.

# testB.o Dump image

Symbol Table

Info Value Size Section/Name

l- 0x0000 0x0000 <UNDEF>

l-file 0x0000 0x0000 <ABS>/testB.s

l-sect 0x0000 0x0000 .text/.text

l- 0x0000 0x0100 <UNDEF>/.STACKSIZE

g- 0x0000 0x0000 .text/\_func

-------------------------------------------

.section .text

\_func:

ADD R16,R16,R17

BRAR R31

# The assembler deletes a stack area.

# testC.o Dump image

Symbol Table

Info Value Size Section/Name

l- 0x0000 0x0000 <UNDEF>

l-file 0x0000 0x0000 <ABS>/testC.s

l-sect 0x0000 0x0000 lwm/lwm

l- 0x0000 0x0080 <UNDEF>/.STACKSIZE

g- 0x0000 0x0000 lwm/\_c

-------------------------------------------

.section lwm

\_c:

.DCW 0x00000000

# The assembler deletes a stack area.

# testA.o Dump image

Symbol Table

Info Value Size Section/Name

l- 0x0000 0x0000 <UNDEF>

l-file 0x0000 0x0000 <ABS>/testA.s

l-sect 0x0000 0x0000 uniform/uniform

l-sect 0x0000 0x0000 .text/.text

l- 0x0000 0x0040 <UNDEF>/.STACKSIZE

g- 0x0000 0x0000 <UNDEF>/\_\_\_stackEnd

g- 0x0000 0x0000 uniform/\_a

g- 0x0004 0x0000 uniform/\_b

g- 0x0000 0x0000 <UNDEF>/\_c

g- 0x0000 0x0000 <UNDEF>/\_func

g-entry 0x0000 0x0000 .text/\_main

-------------------------------------------

.section uniform

\_a:

.DCW 0x00000001

\_b:

.DCW 0x00000002

uniform\_00000008:

.DCW 0x00000000

.section .text

\_main:

MOVI R2,\_\_\_stackEnd

SETHI R2,\_\_\_stackEnd

MOVU R17,\_b

MOVU R16,\_a

CALL R31,\_func

MOVU R17,uniform\_00000008

STI R16,R17,0

TRAP

# The assembler deletes a stack area.

* The assembler deletes a stack area.

The \_\_\_stack definition label, the .DS pseudo-instruction, and the \_\_\_stackEnd definition label are deleted by the assembler.

In place of them, the local non-definition label .STACKSIZE is created with the stack area size specified as the symbol size.

Options specified at linkage

slink testA.o testB.o testC.o -o result.out

# result.out Dump image

f0018000 .section uniform

f0018000 \_a:

f0018000 .DCW 0x00000001

f0018004 \_b:

f0018004 .DCW 0x00000002

f0018008 .DCW 0xf8000000

00000000 .section .text

00000000 \_main:

00000000 MOVI R2,256

00000004 SETHI R2,-310378496

00000008 MOVU R17,1

0000000c MOVU R16,0

00000010 CALL R31,\_func

00000014 MOVU R17,2

00000018 STI R16,R17,0

0000001c TRAP

00000020 \_func:

00000020 ADD R16,R16,R17

00000024 BRAR R31

f8000000 .section lwm

f8000000 \_c:

f8000000 .DCW 0x00000000

ed800000 .section impc

ed800000 \_\_\_stack: # The linker generates a stack area.

ed800000 .DCW 0x00000000

: (Omitted)

ed8000fc .DCW 0x00000000

ed800100 \_\_\_stackEnd:

* The linker generates a stack area.

Since the -z stack-size option is not specified, the maximum value is obtained from the .STACKSIZE symbol sizes of all files and the stack area is generated in the output destination section using the maximum value (0x100) as "stack area size".

Example 2: When other than 0 is specified by the -z stack-size option:

* The stack area is generated using the value specified by the option as "stack area size".

<Example of operation>

The procedure before "Options specified at linkage" is the same as that in Example 1.

Options specified at linkage

slink testA.o testB.o testC.o -o result.out -z stack-size=16 –z stack-section=lwm

# result.out Dump image

f0018000 .section uniform

f0018000 \_a:

f0018000 .DCW 0x00000001

f0018004 \_b:

f0018004 .DCW 0x00000002

f0018008 .DCW 0xf8000000

00000000 .section .text

00000000 \_main:

00000000 MOVI R2,20

00000004 SETHI R2,-134217728

00000008 MOVU R17,1

0000000c MOVU R16,0

00000010 CALL R31,\_func

00000014 MOVU R17,2

00000018 STI R16,R17,0

0000001c TRAP

00000020 \_func:

00000020 ADD R16,R16,R17

00000024 BRAR R31

f8000000 .section lwm

f8000000 \_c:

f8000000 .DCW 0x00000000

f8000004 \_\_\_stack: # The linker generates a stack area.

f8000004 .DCW 0x00000000

f8000008 .DCW 0x00000000

f800000c .DCW 0x00000000

f8000010 .DCW 0x00000000

f8000014 \_\_\_stackEnd:

* The linker generates a stack area.

The stack area is generated in the output destination section using the value specified by the -z stack-size option (0x10) as "stack area size".

The option specification takes precedence over the .STACKSIZE symbol.

Example 3: When the -z stack-size option is not specified and .STACKSIZE does not exist in the input file:

* No stack area is generated.

Example 4: When 0 is specified by the -z stack-size option:

* No stack area is generated.

# Linker Messages

## Message Format

message-type component-number phase-number-of-occurrence message-number: message

* The following contents are output as the continued character string.

Message type: One alphabetic character

Component number: 05

Phase number of occurrence: 6

Message number: Four digits

* Message types (one alphabetic character) are as follows:

|  |  |  |
| --- | --- | --- |
| Message Type | Level | Remarks |
| C | Internal error | Processing is terminated (aborted).  An output file for the link result is not generated. |
| F | Fatal error | Processing is terminated (aborted).  An output file for the link result is not generated. |
| E | Error | Link processing is continued partially and terminated (aborted).  An output file for the link result is not generated. |
| W | Warning | Processing is continued.  An output file for the link result is generated.  (It may not be what the user intended; the operation of the link result cannot be guaranteed.) |

Example: When an invalid "slink -@" option is specified:

F0561009:Invalid option "-@".

## Messages

|  |  |  |
| --- | --- | --- |
| C0564001 | Message | Internal Error. |
| Description | An internal error occurred. Please contact your vendor or your Renesas Electronics overseas representative. |
| F0561001 | Message | No object files specified. |
| Description | An object file is not specified. |
| F0561002 | Message | Cannot open file "*file*". |
| Description | The input file cannot be opened. |
| F0561003 | Message | Cannot read file "*file*". |
| Description | The file cannot be read. |
| F0561004 | Message | Cannot create file "*file*". |
| Description | The output file cannot be created. |
| F0561005 | Message | Cannot write file "*file*". |
| Description | The output file cannot be written. |
| F0561006 | Message | Illegal file format "*file*". |
| Description | The file format is incorrect. |
| F0561007 | Message | Illegal cpu type "*CPU type*" in "*file*". |
| Description | A file of a different CPU type was input. |
| F0561008 | Message | Illegal encode type "*endian type*" in "*file*". |
| Description | A file of a different endian type was input. |
| F0561009 | Message | Invalid option "*option*". |
| Description | An invalid option was specified. |
| F0561010 | Message | Duplicate file "*file*". |
| Description | The same file is specified more than once. |
| F0561011 | Message | Invalid section "*section*" in "*file*". |
| Description | An unsupported section was found in the file. |
| F0561012 | Message | Invalid relocation type in "*file*". |
| Description | An unsupported relocation type was found in the file. |
| F0561013 | Message | Too many symbols. |
| Description | The number of symbols exceeded the translation limit. |
| F0561014 | Message | Too many modules. |
| Description | The number of modules exceeded the translation limit. |
| F0561015 | Message | Section range is out of address space : "*section (Start address:xxxxxxxx,Size:xxxxxxxx)*" |
| Description | Section address exceeded the upper limit of the available area. |
| F0561016 | Message | Section size overflow : "*section (Max size:xxxxxxxx,Section size:xxxxxxxx)*" |
| Description | The section size exceeded the memory size. |
| F0561017 | Message | Memory overflow. |
| Description | There is no more space in the usable memory within the linker. |
| F0561018 | Message | Cannot find “*module name*” in target library. |
| Description | Cannot find specified module in option -c extract / -c delete/ -c replace. |
| F0561019 | Message | Duplicate module “*module name*”. |
| Description | The specified module name is duplicated with the module name in library file. |
| F0561020 | Message | No module in library file. |
| Description | There are no modules in the library. |
| F0561021 | Message | Missing target library file (-o) for library manipulation. |
| Description | Target library file is not specified for library manipulation. |

|  |  |  |
| --- | --- | --- |
| E0562001 | Message | Undefined external symbol "symbol" referenced in "*file-name*". |
| Description | An undefined symbol was referenced in the file. |
| E0562002 | Message | Duplicate symbol "*symbol*" in "*file1 - file2*". |
| Description | The symbol named *symbol* is duplicated. |
| E0562004 | Message | Relocation size overflow : "*file*"-"*section*"-"*offset*" |
| Description | The result of the relocation operation exceeded the relocation size. |
| W0563001 | Message | Duplicate symbol "*symbol*" in "*file1(module) - file2*". |
| Description | The symbol named *symbol* is duplicated in library manipulation.  The symbol in the first file input is given priority. |
| W0563002 | Message | Section address is incompatible with section alignment "*section (address, alignment)*". |
| Description | The address of the section conflicts with the alignment number of the section.  The address is modified to conform to the alignment number. |