

# HITACHI

Software Manual

Operation

RPDP for Windows®

---

***SIOVE***

SEE-3-133 (A)

Software Manual

Operation

RPDP for Windows®

---

***SIOVE***

First Edition, May 2020, SEE-3-133(A)

All Rights Reserved, Copyright © 2020, Hitachi, Ltd.

The contents of this publication may be revised without prior notice.

No part of this publication may be reproduced in any form or by any means without permission in writing from the publisher.

Printed in Japan.

TP<IC> (FL-MW2007)

## SAFETY PRECAUTIONS

- Before building a system, creating or program, or otherwise using this product, carefully read the contents of this manual to ensure that you have a sufficient understanding of the instructions and warnings herein. Incorrect operations might result in system failure.
- Keep this manual in a readily accessible place so that users can refer to it quickly as needed.
- If you are uncertain about any of the information contained in this manual or if any information is unclear, contact our nearest company location or your SE.
- Hitachi is not responsible for accidents or damage caused by incorrect customer use of the product.
- Hitachi is not responsible for accidents or damage caused by use of the product after the customer has modified the Hitachi-provided software.
- Hitachi is not responsible for product reliability in cases where software other than that provided by Hitachi is used.
- Include file backups as part of your daily operations. The contents of files might be lost because of a failure in a file device, a power failure that occurs while files are being accessed, incorrect operations, or other reasons. Back up files regularly as a safeguard against such situations.
- To ensure sufficient safety with respect to the system being used even in cases of product failure, incorrect operations, or program defects, design your system so that the protective and safety circuits are located externally, and implement sufficient safety measures with respect to injuries and major disasters.
- Configure the system so that the emergency stop circuit, interlock circuit, and other such circuits are outside the PLC. PLC failure might result in product failure or accidents.
- Sufficiently verify safety before performing such actions as modifications to a program that is operating, forced output, or a RUN or STOP instruction. Incorrect operations might result in product failure or accidents.
- Safety precautions in this manual are classified into four levels according to the severity of potential hazards: DANGER, WARNING, CAUTION, and Notice.



: Indicates a hazardous situation which, if not avoided, will result in death or serious injury.





: Indicates a hazardous situation which, if not avoided, could result in death or serious injury.



: Indicates a hazardous situation which, if not avoided, could result in minor or moderate injury.



: Indicates a danger (resulting from incorrect use of the product) that can cause property damage or loss not related to personal injury if the safety precautions are not observed.

Failure to observe precautions marked with  or  could also lead to a serious consequence depending on the situation in which the product is used. Therefore, you must observe all of those precautions without fail.

The following are definitions of *serious injury*, *minor or moderate injury*, and *property damage or loss not related to personal injury* used in the safety labels.

**Serious injury:** Is an injury that has aftereffects and requires hospitalization for medical treatment or long-term follow-up care. Examples of serious injuries are as follows: vision loss, burns (caused by dry heat), low-temperature burns, electric-shock injuries, broken bones, and poisoning.

**Minor or moderate injury:** Is an injury that does not require either hospitalization for medical treatment or long-term follow-up care. Examples of minor or moderate injuries include burns and electric-shock injuries.

**Property damage or loss not related to personal injury:** Is damage other than personal injury. Examples of property damage or loss not related to personal injury are as follows: damage or loss of personal property, failure or damage of the main unit of the product, and loss of data.

The safety precautions stated in this manual are based on the general rules of safety applicable to this product. These safety precautions are a necessary complement to the various safety measures included in this product. Although they have been considered carefully, the safety precautions posted on this product and in the manual do not cover every possible hazard. Common sense and caution must be used when operating this product. For safe operation and maintenance of this product, establish your own safety rules and regulations according to your specific needs. A variety of industry standards are available to aid in establishing such safety rules and regulations.

### Revision History

Revision No.	History (revision details)	Issue date	Remarks
A	First edition	May 2020	

**This Page Intentionally Left Blank**

## PREFACE

This manual describes how to create real-time programs that run on the CPMS of the S10VE.

- The following table shows the related software manuals.

Manual number	Manual name
SEE-3-201	S10VE Software Manual CPMS General Description and Macro Specifications
SEE-1-001	S10VE User's Manual General Description

- Note that the following terms have special meanings in this manual:

Abbreviation	Official name
RPDP	Realtime Program Developing Package for S10VE
CPMS	Compact Process Monitor System
PCs	Programmable Controllers
PLC	Programmable Logic Controller

- This manual consists of PART 1 GENERAL DESCRIPTION, PART 2 COMMAND REFERENCE, AND APPENDIXES.

### PART 1 GENERAL DESCRIPTION

This part describes how to develop real-time programs that run on the SV10E and provides an outline of the commands used for development.

### PART 2 COMMAND REFERENCE

This part is a reference for the commands used to develop real-time programs that run on the S10VE. This part describes the functions of each command, as well as optional functions.

### APPENDIXES

The appendixes provide points to note when developing real-time programs that run on the S10VE, error messages, and the formats in which the results of executed commands are displayed.



<Trademarks>

- Microsoft® and Windows® are either a registered trademark or trademarks of Microsoft Corporation in the United States and/or other countries.

<Note for storage capacity calculations>

- Memory capacities and requirements, file sizes and storage requirements must be calculated according to the formula  $2^n$ . The following examples show the results of such calculations by  $2^n$  (to the right of the equal signs).

1 KB (kilobyte) = 1,024 bytes

1 MB (megabyte) = 1,048,576 bytes

1 GB (gigabyte) = 1,073,741,824 bytes

1 TB (terabyte) = 1,099,511,627,776 bytes

- As for disk capacities, they must be calculated using the formula  $10^n$ . Listed below are the results of calculating the above example capacities using  $10^n$  in place of  $2^n$ .

1 KB (kilobyte) = 1,000 bytes

1 MB (megabyte) =  $1,000^2$  bytes

1 GB (gigabyte) =  $1,000^3$  bytes

1 TB (terabyte) =  $1,000^4$  bytes

# CONTENTS

## PART 1 GENERAL DESCRIPTION

CHAPTER 1 OVERVIEW .....	1-2
1.1 About RPDP .....	1-2
1.2 Commands .....	1-4
1.3 Using the Different Processors (CP and HP) .....	1-7
1.3.1 Configuration and roles .....	1-7
1.3.2 Programming environment .....	1-9
1.3.3 RPDP functions and specifications of the CP and HP sites .....	1-10
CHAPTER 2 PROCEDURES FOR PROGRAM DEVELOPMENT .....	1-12
2.1 Overall Flow .....	1-12
2.2 Site Environment .....	1-15
2.2.1 Connection to S10VE by specifying site .....	1-16
2.3 Area Management and Area Divisions in the Main Memory .....	1-17
2.4 Area Allocation for Tasks .....	1-22
2.5 Area Allocation for IRSUBs .....	1-23
2.6 Loading Programs and Creating Tasks .....	1-24
2.7 Indirect Link Resident Subprograms .....	1-24
2.8 Global (GLB) .....	1-24
2.9 Inter-PU Shared Memory (CM) .....	1-24
2.10 Value (VAL) .....	1-25
2.11 Indirect Link Global Data .....	1-25
2.12 Programming Guide for GLB, VAL, and IRSUB .....	1-25
2.13 Constraints on CPMS Program Creation .....	1-34
CHAPTER 3 INSTALLATION AND EXECUTION ENVIRONMENT .....	1-40
3.1 Installation .....	1-40
3.2 Prerequisite Software Products .....	1-40
3.3 Notes on Installation .....	1-40
3.3.1 Notes on installing RPDP .....	1-40
3.3.2 Notes on installing the SHC compiler .....	1-40
3.4 RPDP Execution Environment .....	1-41
3.5 Registering an RPDP User Account .....	1-42
3.5.1 Registering a new account .....	1-42
3.5.2 Adding RPDPusers as a group to which an existing account belongs .....	1-43
CHAPTER 4 COMPILER .....	1-44
4.1 Details of C Compiler Options .....	1-44
4.2 Notes on Compiling .....	1-46
4.2.1 Compiling by using shc .....	1-46
4.3 shc Version Comparisons .....	1-48
4.3.1 Command line options .....	1-48
4.4 Data Generator .....	1-49
CHAPTER 5 PROGRAMMING COMMANDS .....	1-58
5.1 Notes on Programming Commands .....	1-58

CHAPTER 6	ALLOCATOR	1-59
6.1	Allocating and Deallocating Split Areas	1-59
6.1.1	Necessity for split areas	1-59
6.1.2	Allocating split areas	1-60
6.1.3	Deallocating split areas	1-64
6.1.4	Assigning names to GLB and VAL	1-64
6.1.5	Allocating split areas for the CM	1-65
6.2	Value (VAL) Registration and Deletion	1-67
CHAPTER 7	LOADER	1-68
7.1	Linking and Loading	1-68
7.2	Loader Operating Environment	1-69
7.3	Library Search Paths	1-72
7.4	Notes on Linking and Loading	1-72
CHAPTER 8	BUILDER	1-73
8.1	Registering and Deleting Tasks	1-73
8.1.1	About tasks	1-73
8.1.2	Registering a task	1-73
8.1.3	Deleting a task	1-74
8.2	Registering and Deleting a Resident Subprogram	1-75
8.2.1	About indirect link subprograms (IRSUBs)	1-75
8.2.2	Registering an indirect link subprogram (IRSUB)	1-75
8.2.3	Deleting an indirect link subprogram (IRSUB)	1-76
8.3	Registering and Deleting a Built-in Subroutine	1-77
8.3.1	About built-in routines	1-77
8.3.2	Registering a built-in subroutine	1-78
8.3.3	Deleting a built-in subroutine	1-78
CHAPTER 9	MAP	1-79
9.1	Purpose of Displaying Allocator Management Table Information	1-79
9.2	svmap Command Options and Displayed Information	1-80
9.2.1	Map information that is output	1-80
9.2.2	Description of output map information	1-80
9.2.3	Map information output format	1-80
9.3	Logical Address Specification and Information Displayed by the svadm Command	1-82
CHAPTER 10	STARTUP AND PU CONTROL	1-83
10.1	Overview	1-83
10.2	Basic Concept of Startup and PU Control	1-84
10.3	Startup and PU Control Procedure	1-85
10.4	Startup and Stop Types	1-86
10.5	PU State Transitions	1-89
10.5.1	Startup procedure	1-90
10.5.2	PU control procedure	1-92
CHAPTER 11	svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT FUNCTIONS	1-94
11.1	Overview	1-94

11.2	S10VE Status and Subcommand Availability .....	1-95
11.3	Basic Functions .....	1-97
11.4	Other Functions .....	1-101
11.5	Debug Support Commands .....	1-102
11.5.1	svelog command .....	1-102
11.5.2	svdhp command .....	1-103
11.5.3	svcpunow command .....	1-104
11.5.4	svtimex command .....	1-105

## PART 2 COMMAND REFERENCE

CHAPTER 1	COMPILER .....	2-2
svdatagen	.....	2-2
CHAPTER 2	PROGRAMMING COMMAND .....	2-4
makehce	.....	2-4
CHAPTER 3	ALLOCATOR .....	2-16
svdfa	.....	2-16
svdla	.....	2-18
svdfs	.....	2-19
svdls	.....	2-22
svdfv	.....	2-23
svdlv	.....	2-24
CHAPTER 4	LOADER .....	2-25
svload	.....	2-25
svdload	.....	2-39
svcomp	.....	2-40
CHAPTER 5	BUILDER .....	2-45
svctask	.....	2-45
svdtask	.....	2-47
svbuild	.....	2-48
svdbuild	.....	2-51
svirglb	.....	2-54
CHAPTER 6	MANAGEMENT TOOL .....	2-56
svmap	.....	2-56
svadm	.....	2-59
svsitecntl	.....	2-62
CHAPTER 7	STARTUP AND PU CONTROL .....	2-63
svrpl	.....	2-63
svcpuctl	.....	2-66
CHAPTER 8	svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS .....	2-68
svdebug	.....	2-68
qu	.....	2-71

ab	2-72
re	2-73
ta	2-74
su	2-77
rs	2-78
tm	2-79
ct	2-81
sht	2-82
md	2-83
sd	2-87
bs	2-90
bg	2-92
mcp	2-94
mmv	2-96
mf	2-98
el	2-100
ss	2-100
st	2-101
gt	2-102
br/stickybr	2-103
rb	2-109
rd	2-110
rr	2-113
go	2-114
ld	2-115
sv	2-123
cm	2-125
dr	2-127
ds	2-127
svdhp	2-128
svadm	2-128
si	2-129
sp	2-130
ps	2-132
pe	2-132
ver	2-133
lbr	2-134
lrb	2-135
lrd	2-136
lrr	2-137
lgo	2-138
s	2-138
help	2-139
q	2-141
!	2-141
svelog	2-142
svdhp	2-144
svcpunow	2-147
svtimex	2-148

## **APPENDIXES**

APPENDIX A	NAMES USABLE IN PROGRAMS .....	A-2
APPENDIX B	LIBRARIES .....	A-6
APPENDIX C	SITE MANAGEMENT FILES .....	A-9
APPENDIX D	ERROR MESSAGES .....	A-15
APPENDIX E	NOTES ON USING RPDP .....	A-47
APPENDIX F	MAP DISPLAY FORMAT .....	A-49
APPENDIX G	DISPLAY FORMATS OF md AND sd OF svdebug (ONLINE DEBUGGER) .....	A-64
APPENDIX H	LIST OF STACK SIZES FOR LIBRARY USE .....	A-68

## FIGURES

Figure 1-1	Configuration of a System in which the Development Tool is Used .....	1-2
Figure 1-2	S10VE Hardware Configuration .....	1-7
Figure 1-3	HP and CP Site Environment and Hardware .....	1-8
Figure 1-4	Program Development Flowchart (from Site Construction to Program Development) .....	1-13
Figure 1-5	S10VE Site Directory Structure .....	1-15
Figure 1-6	Logical Space Managed by the CPMS .....	1-17
Figure 1-7	S10VE Physical Memory Map .....	1-20
Figure 1-8	Arrangement of Tasks in Logical Space .....	1-22
Figure 1-9	Arrangement of Tasks in Logical Space (Multitasks) .....	1-22
Figure 1-10	IRSUB Arrangement in Logical Space .....	1-23
Figure 1-11	IRSUB Arrangement (Multi-entry) in Logical Space .....	1-23
Figure 1-12	Enabling or Disabling Writing .....	1-35
Figure 1-13	Comparison of Data Sizes .....	1-38
Figure 1-14	Sample Declaration of a Structure with Data Relocation Taken into Account ....	1-39
Figure 1-15	Example of Explicitly Declared Free Areas .....	1-39
Figure 1-16	Sample Declaration with Structure Size Taken into Account .....	1-39
Figure 1-17	defines.h .....	1-53
Figure 1-18	Sample Layout of Split Areas .....	1-61
Figure 1-19	Correspondence between CM Spaces in the Logical Space of the CPMS and CM Spaces in the S10VE Main Memory .....	1-65
Figure 1-20	Creating a Load Module and Backup File .....	1-68
Figure 1-21	Load Module Structure .....	1-69
Figure 1-22	Loading Processing .....	1-70
Figure 1-23	S10VE Startup from a Development Machine .....	1-83
Figure 1-24	Concept of Overall Control of the S10VE .....	1-84
Figure 1-25	PU (OS Startup/Stop) State Transitions .....	1-89
Figure 2-1	Function Call Relationship and Stack Usage Amount .....	2-31
Figure 2-2	Format of the Display Produced by svcomp for a Program or Subprogram .....	2-43
Figure 2-3	Format of the Display Produced by svcomp for GLB or CM .....	2-44
Figure 2-4	Memory Access Range .....	2-86
Figure 2-5	Operation Procedure for Dynamic Display .....	2-86
Figure 2-6	Memory Access Range .....	2-95

## TABLES

Table 1-1	RPDP Commands .....	1-4
Table 1-2	Access Targets of the HP and the CP .....	1-8
Table 1-3	Programming Resources and their Availability at the CP Site .....	1-9
Table 1-4	RPDP Functions and Sites to be Processed .....	1-10
Table 1-5	RPDP Commands and Sites that are Processed .....	1-11
Table 1-6	Usage of the Logical Spaces .....	1-18
Table 1-7	Logical Space Addresses and Sizes .....	1-19
Table 1-8	Assigning Names to GLB and VAL .....	1-25
Table 1-9	Using GLB and VAL Names .....	1-26
Table 1-10	IRSUB Usage .....	1-32
Table 1-11	Prerequisite Software Products of RPDP .....	1-40
Table 1-12	Values Specified for the S10VE RPDP Execution Environment .....	1-41
Table 1-13	Environment Variables Required for shc Compiler Operation .....	1-45
Table 1-14	Floating-point Number Control Options .....	1-46
Table 1-15	Handling of Floating-point Numbers and Applicable Standard Libraries .....	1-46
Table 1-16	Comparison of Versions of the shc Command Line Options .....	1-48
Table 1-17	Initial Value Type Conversion Specifications .....	1-57
Table 1-18	Relationship between Split Area Use and GAREA Selection .....	1-60
Table 1-19	Conditions for Load Modules .....	1-69
Table 1-20	External Reference Combinations .....	1-71
Table 1-21	Combinations of Output Information and Selectable Output Formats .....	1-81
Table 1-22	Startup and Stop Types .....	1-86
Table 1-23	Download Options .....	1-90
Table 2-1	Binary Data Layout .....	2-3
Table 2-2	Combinations of svdfa Options .....	2-17
Table 2-3	Relationship between the Value Specified for svtype and the Alignment Count ...	2-20
Table 2-4	Combinations of svdfs Options .....	2-20
Table 2-5	Stack Size Calculation Examples .....	2-31
Table 2-6	Available Combinations of Output Resources and Output Sequence, and Default Output Sequences .....	2-58
Table 2-7	Functions of svdebug .....	2-70
Table 2-8	Task States .....	2-75
Table 2-9	Status Bit Strings .....	2-75
Table 2-10	Explanation of the id, t, and cyct Parameters .....	2-80
Table 2-11	Relationships between Specifiable Values and Options .....	2-85
Table 2-12	Display Formats Depending on the Combination of Options .....	2-85
Table 2-13	Management States of Resources .....	2-119
Table A-1	Conditions for Specifying Library Names .....	A-6
Table A-2	Error Messages .....	A-16
Table A-3	Real-Time Source Management Status .....	A-51



**This Page Intentionally Left Blank**

## **PART 1 GENERAL DESCRIPTION**

## CHAPTER 1 OVERVIEW

### 1.1 About RPDP

The Realtime Program Developing Package (RPDP/S10VE) is a tool that is used to develop programs that are run on the real-time operating system (CPMS) of the S10VE. This tool operates on a development machine on which the 64-bit version of Windows® 7 or Windows® 10 is installed. The following figure shows the configuration of a system in which the development tool is used.

- RPDP/S10VE: Realtime Program Developing Package for S10VE
- CPMS: Compact Process Monitor System

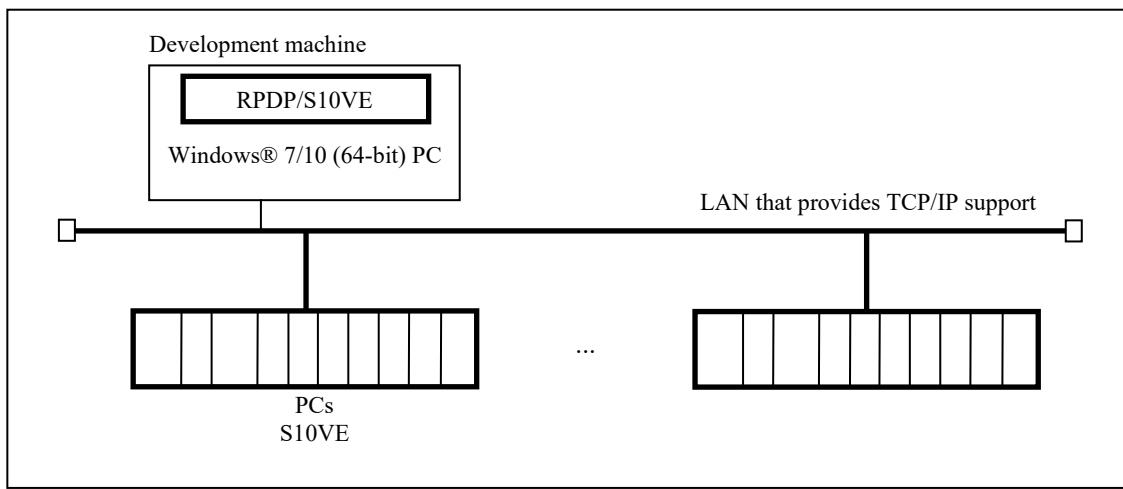
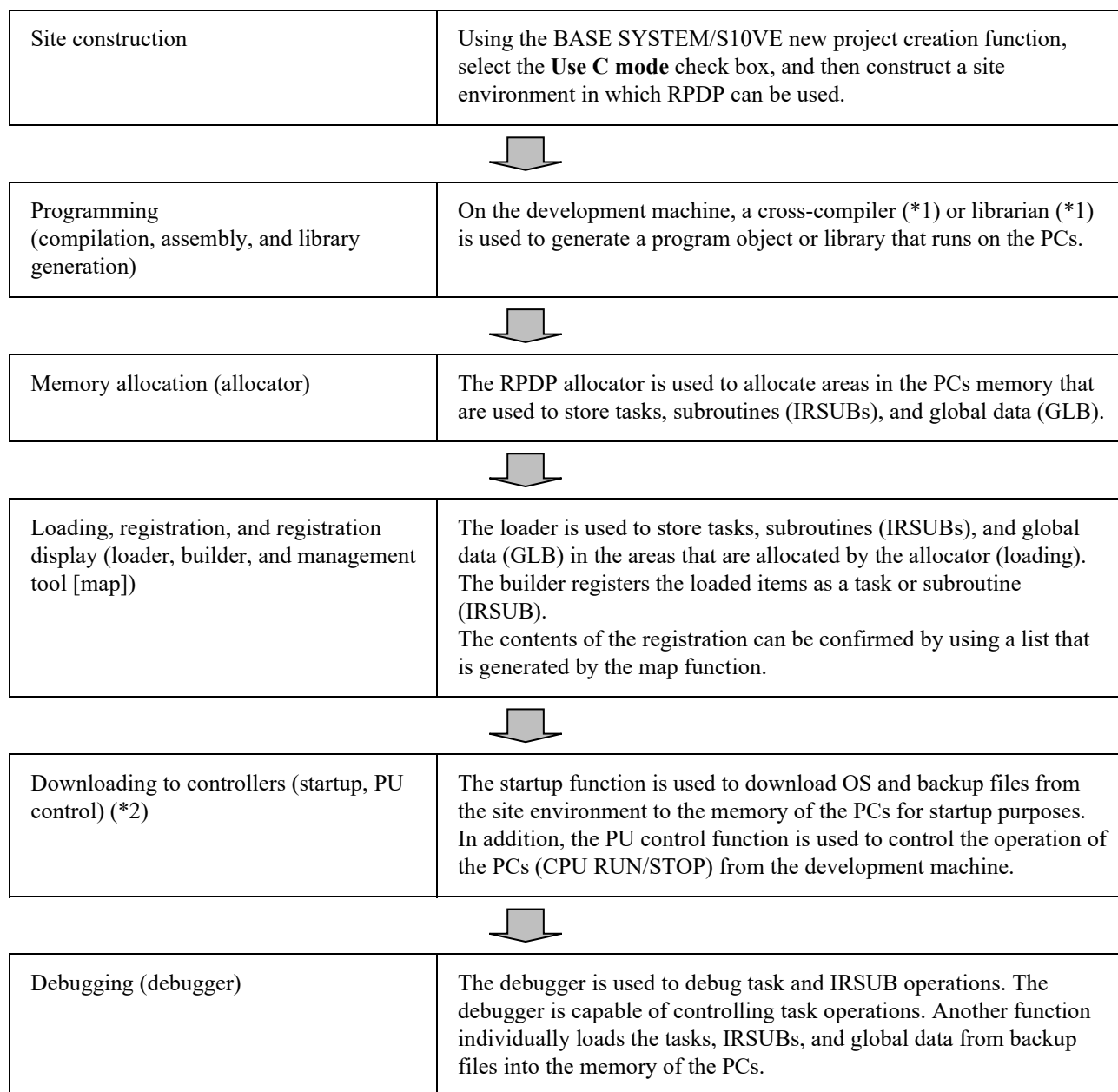


Figure 1-1 Configuration of a System in which the Development Tool is Used

When developing real-time programs that run on the CPMS, use the dedicated development system RPDP/S10VE (hereinafter abbreviated to *RPDP*). Doing so enables programs running on the CPMS to use attributes and functions provided for high-speed, real-time processing.

The following describes the RPDP program development sequence and supported functions:



(\*1) Use the “Renesas Microcomputer Development Environment System SuperH RISC engine C/C++ Compiler Package Ver. 9.04 Release 00” as a cross-compiler, assembler, and librarian.

(\*2) The PCs subject to downloading are the S10VE PCs.

## 1. OVERVIEW

### 1.2 Commands

Table 1-1 lists the RPDP commands.

Table 1-1 RPDP Commands (1/3)

Classification	Command name	Function	Reference page
Compiler	svdatagen	Creates a loadable binary file of initial-value data.	2-2
Programming commands	optlnk	Librarian (part of the compiler package)	(*)
	optlnk	Linker (part of the compiler package)	(*)
	makehce	make command	2-4
Allocator	svdfa	Allocates split areas and generates backup files.	2-16
	svdla	Releases split areas and deletes backup files.	2-18
	svdfs	Allocates secondary partition areas.	2-19
	svdls	Releases secondary partition areas.	2-22
	svdfv	Registers a VAL.	2-23
	svdlv	Releases a VAL.	2-24
Loader	svload	Stores a resource in a backup file and registers it as management information.	2-25
	svdload	Deletes a resource from the management information.	2-39
	svcomp	Provides a comparison with a registered resource.	2-40
Builder	svctask	Creates a task.	2-45
	svdtask	Deletes a task.	2-47
	svbuild	Creates an indirect link subprogram.	2-48
		Creates a built-in subroutine.	2-49
	svdbuild	Deletes an indirect link subprogram.	2-51
		Deletes a built-in subroutine.	2-52
svirglb	Registers or deletes an IRGLB.	2-54	

(\*) See the manual for the “Renesas Microcomputer Development Environment System SuperH RISC engine C/C++ Compiler Package Ver. 9.04 Release 00”.

Table 1-1 RPD P Commands (2/3)

Classification	Command name	Function		Reference page	
Online debugger	svdebug	Starting or stopping tasks	qu	Requests the start of a task.	2-71
			ab	Prohibits task startup.	2-72
			re	Cancels prohibition of task startup.	2-73
			ta	Displays the status of a task.	2-74
			su	Suppresses task execution.	2-77
			rs	Cancels suppression of task execution.	2-78
			tm	Starts a task cyclically.	2-79
			ct	Cancels cyclical task startup.	2-81
			sht	Displays cyclical task startup.	2-82
			si	Initializes the stack.	2-129
			sp	Displays the amount of the stack in use.	2-130
		Memory printing or patching	md	Displays or changes the contents of memory according to a specified address.	2-83
			sd	Displays or changes the contents of memory according to a specified name.	2-87
			bs	Sets data in a specified bit position.	2-90
			bg	Displays the data that exists in a specified bit position.	2-92
			mcp	Copies the contents of memory.	2-94
			mmv	Moves the contents of memory.	2-96
			mf	Sets a pattern value in memory.	2-98
		Breakpoints	br	Sets or displays breakpoints.	2-103
			stickybr	Sets or displays breakpoints that are not deleted by a reset start.	2-103
			rb	Deletes breakpoints.	2-109
			rd	Displays registers.	2-110
			rr	Changes the contents of registers.	2-113
			go	Resumes execution from a breakpoint.	2-114
		System error display	el	Displays the error log.	2-100
			ss	Displays the system status.	2-100
		Current time setting or display	st	Sets the current time.	2-101
			gt	Displays the current time.	2-102
		Uploading/downloading/comp aring	ld	Downloads resources individually.	2-115
			sv	Backs up resources individually.	2-123
			cm	Compares the contents of a backup file and the S10VE memory.	2-125
		Enabling or disabling the DHP recording function	dr	Enables the DHP recording function.	2-127
			ds	Disables the DHP recording function.	2-127
		Ladder debugging functions	lbr	Sets or displays breakpoints.	2-134
			lrb	Deletes breakpoints.	2-135
			lrd	Displays registers.	2-136
			lrr	Rewrites registers.	2-137
			lgo	Resumes execution from a breakpoint.	2-138
			s	Executes steps.	2-138

## 1. OVERVIEW

Table 1-1 RPD P Commands (3/3)

Category	Command name	Function		Reference page	
Online debugger	svdebug	Other	svdhp	Displays the DHP.	2-128
			svadm	Displays the resource name for an address.	2-128
			ps	Starts displaying a debug statement.	2-132
			pe	Finishes displaying a debug statement.	2-132
			ver	Displays the version of the CPMS.	2-133
			help	Displays the subcommands.	2-139
			q	Terminates the debugger.	2-141
			!	Runs a command on the development machine at the time that svdebug is run.	2-141
Management tools	svmap	Displays map information.		2-56	
	svadm	Displays information corresponding to an address.		2-59	
	svsitecntl	Controls and displays the site status.		2-62	
Startup/PU control	svrpl	Performs remote loading.		2-63	
	svcpuctl	Controls remote statuses.		2-66	
Activity management	svcpunow	Displays the CPU load (ratio).		2-147	
	svtimex	Displays the task activity ratio.		2-148	
Display of error logs and DHP traces	svelog	Outputs error log information.		2-142	
	svdhp	Displays DHP trace information.		2-128	

### 1.3 Using the Different Processors (CP and HP)

The SH4A dual processor is used as the CPU in the S10VE. Core 0 of the dual processor is used as the CP (communication processor), and Core 1 is used as the HP (high-speed processor). This section describes the roles of and programming methods for these processors.

#### 1.3.1 Configuration and roles

The SH4A dual processor is used as the CPU in the S10VE. Core 0 of the dual processor is used as the CP (communication processor) for control and communications, and Core 1 is used as the HP (high-speed processor) for control. These two cores are therefore used for control and communications processing and for control processing, respectively. Core 0 (the CP) controls communications to distribute load in order to improve the performance of the control programs, such as PI/O accesses, of Core 1 (the HP).

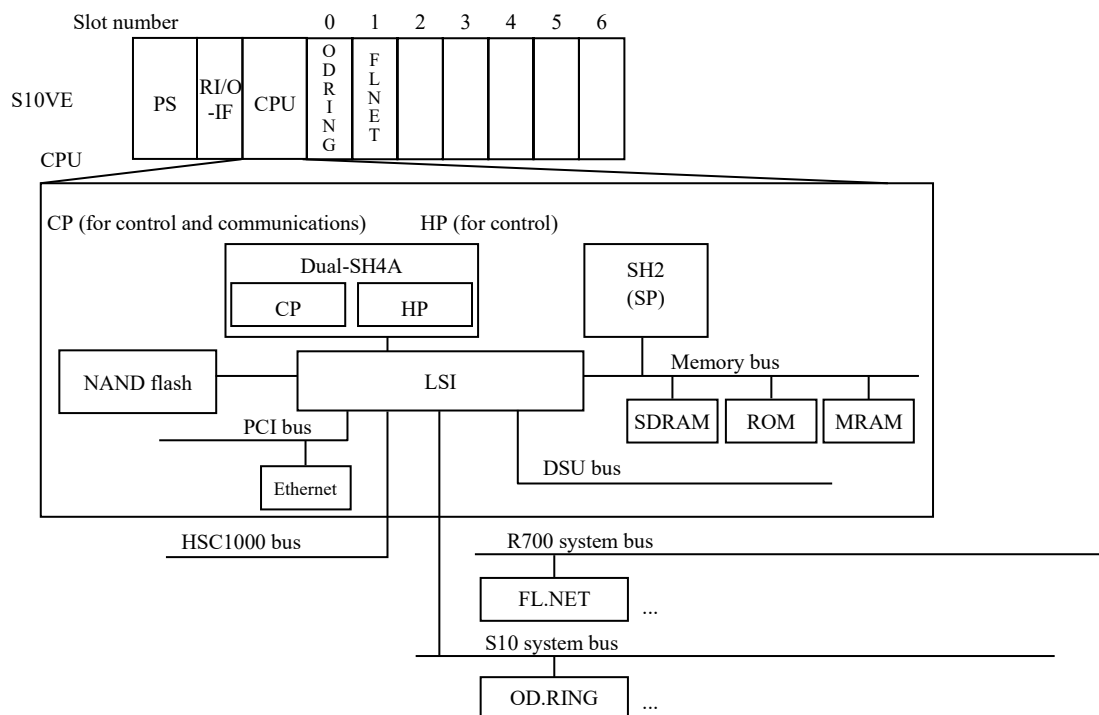


Figure 1-2 S10VE Hardware Configuration

The CP runs communication-processing programs and control programs. Some examples of communication-processing programs include the following: programs that perform communications by using a network upon a request from the HP, communication-processing system tasks provided by subsystems such as RCTLNET, servers for connecting to tools, and system tasks for running the Ethernet communication instructions of ladder programs. Control programs regularly access PI/O units through the memory interface to perform calculations and other operations.



## 1. OVERVIEW

Control programs mainly run in the HP. Control programs regularly access PI/O units by using ladder programs and HI-FLOW programs, to perform calculations and other operations. Control programs on the HP side are capable of data transmission and reception by using the Ethernet communication instructions of ladder programs.

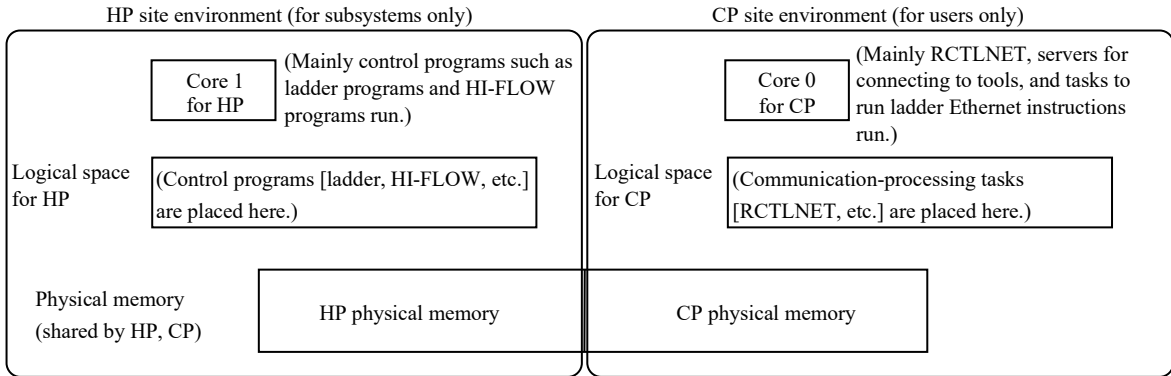


Figure 1-3 HP and CP Site Environment and Hardware

The following table shows the access targets of the CP and the HP. Communication tasks run on the CP, and high-speed sequence tasks run on the HP. Tasks on the HP cannot communicate via the built-in Ethernet by using the I/O interface. Similarly, tasks on the CP cannot run ladder programs. The usable networks and I/O interfaces therefore differ depending on the core in which the task operations are performed. The following table shows the classification of the access targets of the CP and HP according to the use of the CP and HP cores.

Table 1-2 Access Targets of the HP and the CP

Access target	CP	HP	Notes
NAND flash	Accessible	Inaccessible	
Built-in Ethernet (socket)	Accessible	Inaccessible	Socket: Error return
HSC-1000	Accessible	Accessible	
HSC-2100	Accessible	Accessible	
S10 module (S10 bus space)	Accessible	Accessible	

## 1.3.2 Programming environment

The RPDP of the S10VE allocates site names for the cores (the CP and the HP) of PCs to manage programming resources (including tasks, subprograms, and global data).

When creating a new BASE SYSTEM/S10VE project, selecting the **Use C mode** check box creates CP and HP sites with PCs numbers and unique names for each core.

In the RPDP, programming resources are provided for the CP site and the HP site. The CP site is used to run communication-processing system tasks provided by subsystems (such as RCTLNET), servers for connecting to tools, and system tasks to run the Ethernet communication instructions of ladder programs. The HP site is used to run control programs by regularly accessing PI/O units by using ladder programs and HI-FLOW programs, as well as to perform calculations, so do not register user tasks for the HP site.

The following table shows programming resources and their availability at the CP site.

Table 1-3 Programming Resources and their Availability at the CP Site

Programming resource	CP site (for communication tasks)
Task	Available (User task: TN = 1 to 224)
IRSUB	Available (Quantity: 8191, but with 256 reserved by the OS)
Built-in subroutine	Available (point(14)* entry(4), but with entry1 reserved by the OS)
GLB (including definition in the CM)	Available (Quantity: 8192, but with 256 reserved by OS)
VAL	Available (Quantity: 4096, but with 10 reserved by the OS)

(\*) When initial values are registered in (loaded to) the GLB (with initial values) of the CM, register initial values at the CP site. Because the CM is a memory space to be shared by both the CP and HP sites, the site where initial values can be registered is limited to the CP site.

## 1. OVERVIEW

### 1.3.3 RPDP functions and specifications of the CP and HP sites

Table 1-4 shows the relationship between the sites to be processed and the RPDP functions that process sites.

Table 1-4 RPDP Functions and Sites to be Processed

RPDP function	Site to be processed
Compiler	The CP site and HP site do not need to be specified.
Programming command	The CP site and HP site do not need to be specified.
Allocator, loader, builder, and management tools (such as svmap)	The provided functions (including program registration and area allocation) are run individually for each site by specifying either the CP or HP site.
Online debugger	The debug function is run individually for each site by specifying either the CP or HP site.
Startup and PU control	These are not run individually for the CP and HP sites, but rather are run simultaneously for these sites. However, the CP site is specified as the target of the operation.
Activity management	This is run individually for each site by specifying the CP or HP site.
Maintenance commands	These are run individually for each site by specifying the CP or HP site.

Table 1-5 shows the sites that are processed for each RPDP command.

Table 1-5 RPDP Commands and Sites that are Processed

RPDP function	Command name	Site to be processed	
		Processing CP and HP sites individually	Processing CP and HP sites simultaneously
Compiler	svdatagen	Applicable	
Programming command	optlnk	No site specified	No site specified
	makehce	No site specified	No site specified
<ul style="list-style-type: none"> <li>• Allocator</li> <li>• Loader</li> <li>• Builder</li> </ul>	svdfa	Applicable	
	svdla	Applicable	
	svdfs	Applicable	
	svdls	Applicable	
	svdfv	Applicable	
	svdlv	Applicable	
	svload	Applicable	
	svdload	Applicable	
	svcomp	Applicable	
	svctask	Applicable	
	svdtask	Applicable	
	svbuild	Applicable	
	svdbuild	Applicable	
svirglb	Applicable		
Online debugger	svdebug	Applicable	
Management tools	svmap	Applicable	
	svadm	Applicable	
	svsitecntl	Applicable	
<ul style="list-style-type: none"> <li>• Startup</li> <li>• PU control</li> </ul>	svrpl		Applicable
	svcpuctl		Applicable
Activity management	svcpunow	Applicable	
	svtimex	Applicable	
Maintenance commands	svdhp	Applicable	
	svelog	Applicable	

## **CHAPTER 2 PROCEDURES FOR PROGRAM DEVELOPMENT**

### **2.1 Overall Flow**

Figure 1-4 shows the overall flow of the procedures for program development.

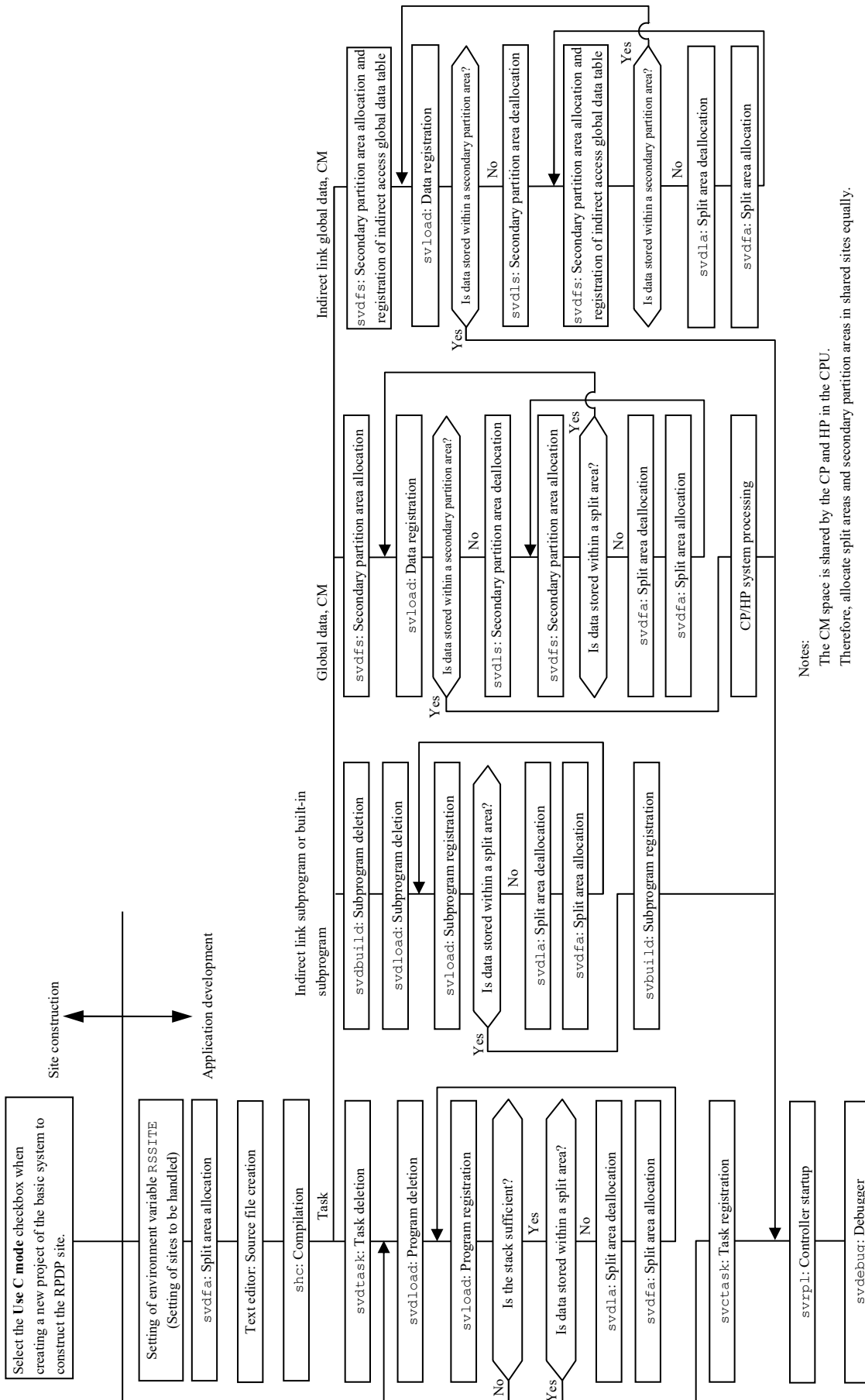


Figure 1-4 Program Development Flowchart (from Site Construction to Program Development) (1/2)

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

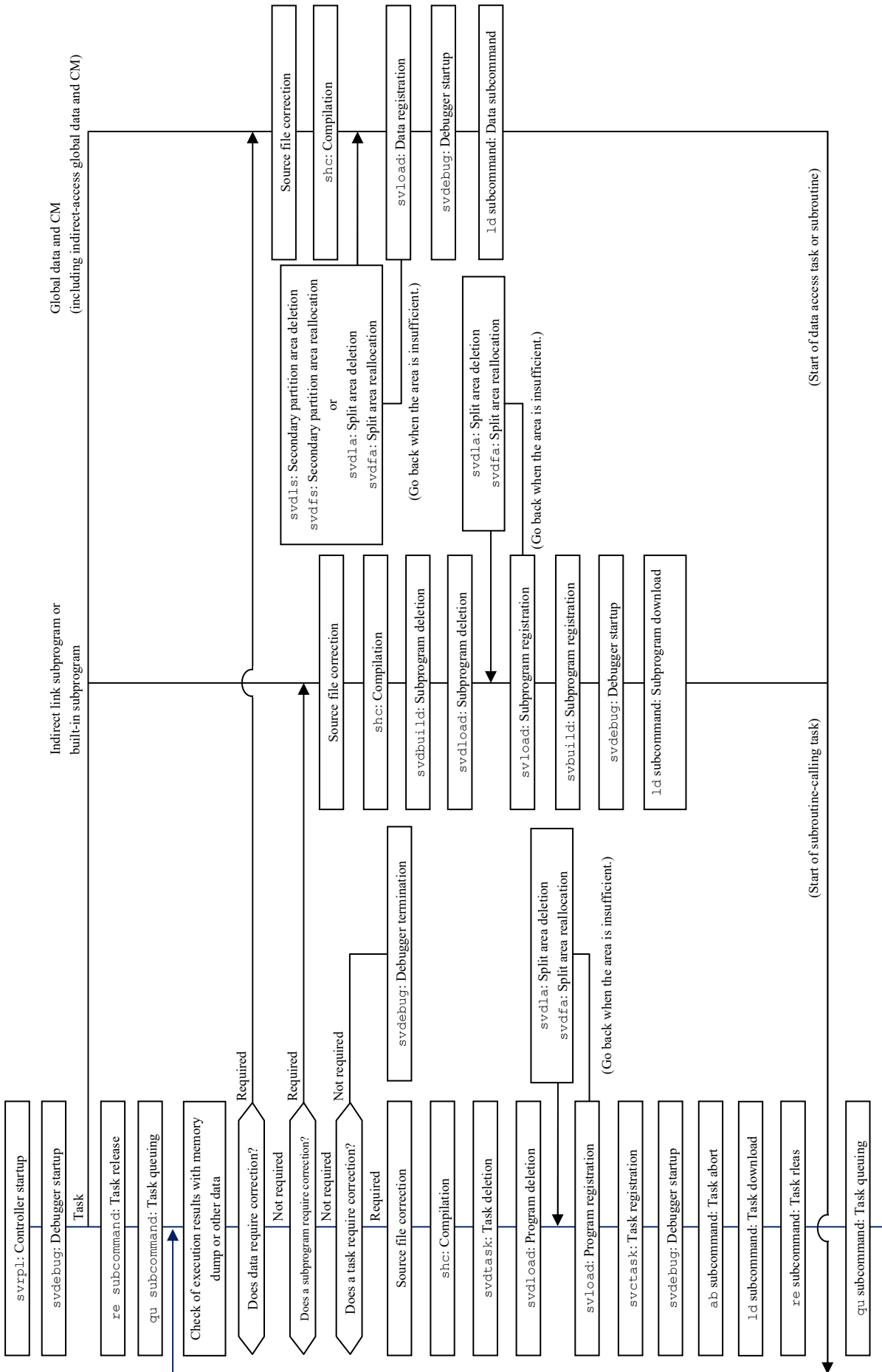


Figure 1-4 Program Development Flowchart (from Site Construction to Program Development) (2/2)

## 2.2 Site Environment

The RPDP allocates site names to each CPU core of the S10VE (CP, the communication processor, and HP, the high-speed processor) and manages tasks that run on the CPMS, subprograms, and global data for each core.

Site names are unique to each core. Site names are determined by the BASE SYSTEM/S10VE when the **Use C mode** project is created in the BASE SYSTEM/S10VE. Site names are created as *PCs-number<sub>cp</sub>* name or *PCs-number<sub>hp</sub>* name for each PCs number of the project. The BASE SYSTEM/S10VE creates a site name directory (called the site directory) for each site, and then copies the base site to the site directory. Management files owned by each site are placed into sites. Management files include the backup files, which contain the initial-value data files of the S10VE memory, and the files that manage tasks, subroutines, and global data that are stored in the backup files.

The site directory is created in a fixed directory for each PCs number (*C:\S10VE\PCs-number\PCs-number\_unit\PCs-number<sub>cp</sub>*) for each PCs number.

The RPDP also manages a CPU site, which refers collectively to the CP and HP sites. The CPU name is the same as the site name allocated to the CP.

When the CP and HP must be operated simultaneously, such as during remote loading, specify the CPU name.

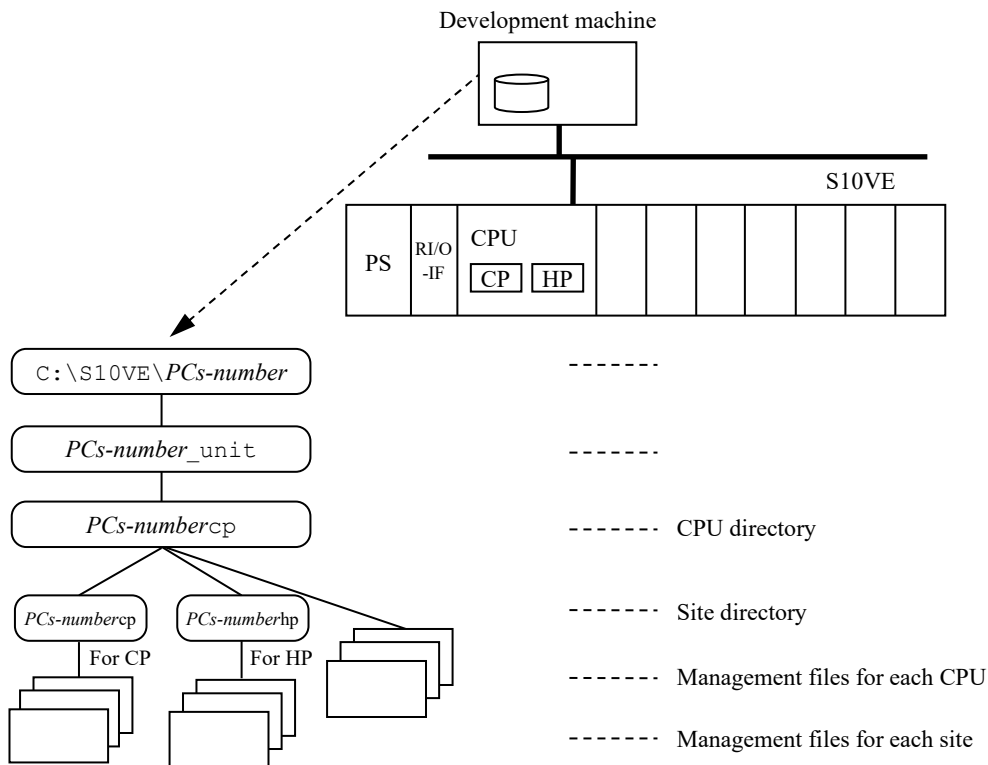


Figure 1-5 S10VE Site Directory Structure



## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

### 2.2.1 Connection to S10VE by specifying site

When establishing a connection to the S10VE for remote loading or error log collection by specifying the site created in the BASE SYSTEM/S10VE, open the applicable project of the PCs number in the BASE SYSTEM/S10VE, and then change the PCs to be connected.

Connect the PCs to the S10VE that has the IP address for which the connected PCs were changed.

After the connected PCs have been changed, the project does not need to be opened, but it must be reconnected by changing the connected PCs when connecting it to another S10VE. Because RPDP does not support connections to the ET.NET module, always connect to the CPU module.

### 2.3 Area Management and Area Divisions in the Main Memory

On the development machine, RPDP performs area management for the main memory of the S10VE. The purpose of area management is to allocate programs, subprograms, and data in the main memory in an efficient manner, without duplicate allocations. The memory space subject to area management by RPDP is the logical space managed by CPMS on the physical memory and S10VE. The physical memory is mapped from the beginning of the logical space by using the GAREA-defined size for each use. (Each CPMS manages the logical space for the CP and HP of the CPU.)

Figure 1-6 shows the logical space managed by the CPMS. Table 1-6 shows the usage of the logical space.

Logical space

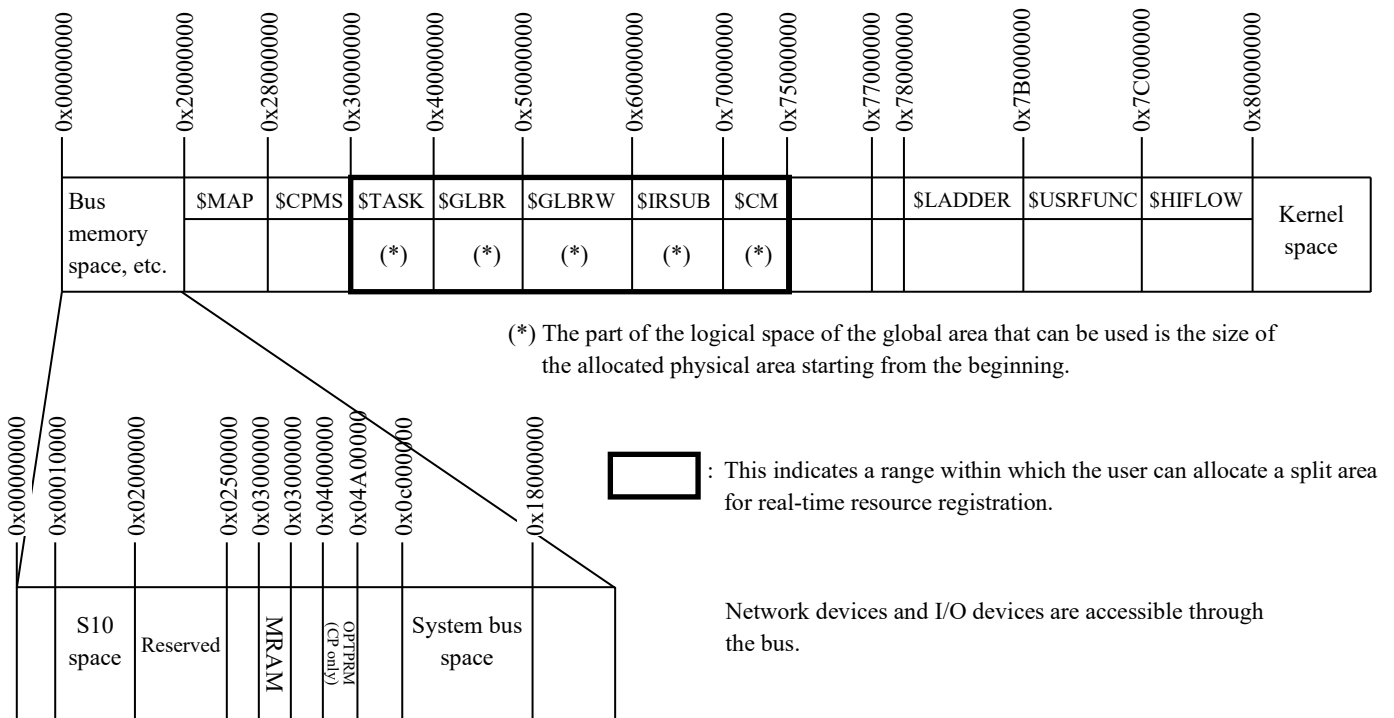


Figure 1-6 Logical Space Managed by the CPMS

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

Table 1-6 Usage of the Logical Spaces

Global area name of the logical space	Use
\$TASK	This area stores tasks (programs).
\$GLBR	This area stores read-only GLB.
\$GLBRW	This area stores read/write GLB.
\$IRSUB	This area stores subprograms.
\$CM	This space can be shared by PUs.
\$LADDER	This area stores ladder programs. (This area is available only in the HP.)
\$USRFUNC	This area stores the user-calculation function of ladder programs. (This area is available only in the HP.)
\$HIFLOW	This area stores the HI-FLOW program. (This area is available only in the HP.)
\$MAP	This stores the following tables, which are managed by RPDP: <ul style="list-style-type: none"> <li>• IRSUB indirect link table (IRSUBT)</li> <li>• Task control block (TCB)</li> <li>• IRGLB indirect link table (IRGLBT)</li> <li>• Built-in subroutine table (USLCB)</li> </ul>
\$CPMS	This space is used by the CPMS.

Table 1-7 shows the addresses and sizes of the logical spaces managed by the CPMS that are shown in Figure 1-6.

Table 1-7 Logical Space Addresses and Sizes

S10VE			
Global area name of the logical space (use)	Address	Size	Area name of the SH4 virtual address space
Reserved	0x00000000-0x0000ffff	64 KB	P0 area (U0 area)
S10 space	0x00010000-0x01ffffff	32 MB - 64 KB	
Reserved	0x02000000-0x02ffffff	16 MB	
MRAM	0x03000000-0x032fffff	3 MB	
Reserved	0x03300000-0x03ffffff	13 MB	
OPTPRM	0x04000000-0x049fffff	10 MB	
Reserved	0x04a00000-0x0bffffff	118 MB	
Bus memory space	0x0c000000-0x17ffffff	192 MB	
Reserved	0x18000000-0x1ffffff	128 MB	
\$MAP	0x20000000-0x27ffffff	128 MB (1.5 MB)	
\$CPMS	0x28000000-0x2ffffff	128 MB	
\$TASK	0x30000000-0x3ffffff	256 MB (8 MB)	
\$GLBR	0x40000000-0x4ffffff	256 MB (4 MB)	
\$GLBRW	0x50000000-0x5ffffff	256 MB (4 MB)	
\$IRSUB	0x60000000-0x6ffffff	256 MB (4 MB)	
\$CM	0x70000000-0x74ffffff	80 MB (2 MB)	
Reserved	0x77000000-0x77ffff	16 MB	
\$LADDER	0x78000000-0x7affffff	48 MB (8 MB)	
\$USRFUNC	0x7b000000-0x7bffffff	16 MB (2 MB)	
\$HIFLOW	0x7c000000-0x7ffffff	64 MB (8 MB)	
Kernel space	0x80000000-0x9ffffff	512 MB	P1 area
	0xa0000000-0xbffffff	512 MB	P2 area
Reserved	0xc0000000-0xdffffff	512 MB	P3 area
	0xe0000000-0xfffffff	512 MB	P4 area

The value in parentheses given for the size of each logical space is the size for which the physical memory is allocated.

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

Figure 1-7 shows the details of the physical memory map.

0x04000000	0x04080000	0x04100000	0x04360000	0x04380000	0x04D80000	0x04F80000	0x05580000	0x06780000	0x05F80000	0x06780000	0x07D00000	0x09280000	0x0935C000	0x0BEFC000	0x0C0000000	
OS								User space for HP				User space for CP				
SPM, HKP	CPMS for HP	CPMS for CP	OPT PRM	CM	F.U.	LAD DER	USRF UNC	HIFLOW	Tasks, subprograms, and GLB for HP				Tasks, subprograms, and GLB for CP			
512 KB	1472 KB	1472 KB	128 KB													

Figure 1-7 S10VE Physical Memory Map

SPM: This is the CPMS edition data.

HKP: This is a space used to run the Hardware KROM Program that is copied from the ROM.

CPMS: The CPMS is the main program of the OS. The CPMS is located and runs on both the HP (the high-speed processor) and the CP (communication processor) of the CPU.

The 128 KB space at the end of the map is the memory space shared by the CPMS for the HP and CPMS for the CP.

OPTPRM: Parameters for optional modules are set in this area.

CM: The CM is memory that is shared between processors that can be accessed from the CM space in the logical spaces of both processors (CP and HP) of the CPU.

F.U.: This is reserved for future use.

LADDER: Ladder programs are downloaded to this area.

USRFUNC: The user calculation function of ladder programs is downloaded to this area.

HIFLOW: HI-FLOW programs are downloaded to this area.

Area for tasks, subprograms, and GLB:

Tasks, subprograms, and GLB data are downloaded to this area.

Map information, TCB, IRSUBT, IRGLBT, and USLCB are also downloaded to this area.

OS work for HP, OS work for CP, and network buffer:

These are buffer areas used by the OS, consisting mainly of a DHP area and network buffer.

System reserved area:

This area is reserved by the hardware.

● Details of the task, subprogram, and GLB area

The details for the task, subprogram, and GLB area are as follows:

Task, subprogram, and GLB area				
\$MAP	\$TASK	\$GLBR	\$GLBRW	\$IRSUB

\$MAP: This area is for storing management information in the S10VE memory.

\$TASK: This area is for storing tasks (programs).

\$GLBR: This area is for storing read-only GLB.

\$GLBRW: This area is for storing read/write GLB.

\$IRSUB: This area is for storing subprograms.

When generating a task or subprogram that runs on the CPMS or data to be used by the task or subprogram, first allocate a split area (area) within a global area (GAREA) for task, subprogram, and data storage. For GLB (global data) and CM, further divide the split area into secondary partition areas (sarea). Tasks and subprograms specify secondary partition area names to access data.

(1) Split area

Split areas (area) are allocated by using `svdfa` and deallocated by using `svdla`. Multiple split areas can be allocated in a global area (GAREA).

When a split area is allocated, a backup file of sufficient size for the allocated area is generated.

(2) Secondary partition area

Multiple resources can be arranged within a split area allocated by `svdfa`. Tasks and subprograms are placed in split areas by using `svload` and deallocated by using `svdload`. A GLB and CM secondary partition area (sarea) is allocated by using `svdfs` and deallocated by using `svdls`.

To deallocate a split area (area), use `svdla`.

**2.4 Area Allocation for Tasks**

All tasks that run under the CPMS are allocated in a single logical space. The tasks are stored in a split area allocated to \$TASK. Multiple tasks can be stored in a single split area. Place text on a page boundary (4 KB boundary), and data/bss and stack on an 8-byte boundary.

In addition, ensure that text/data and stack/bss for the same task and the OS work are located on different pages.

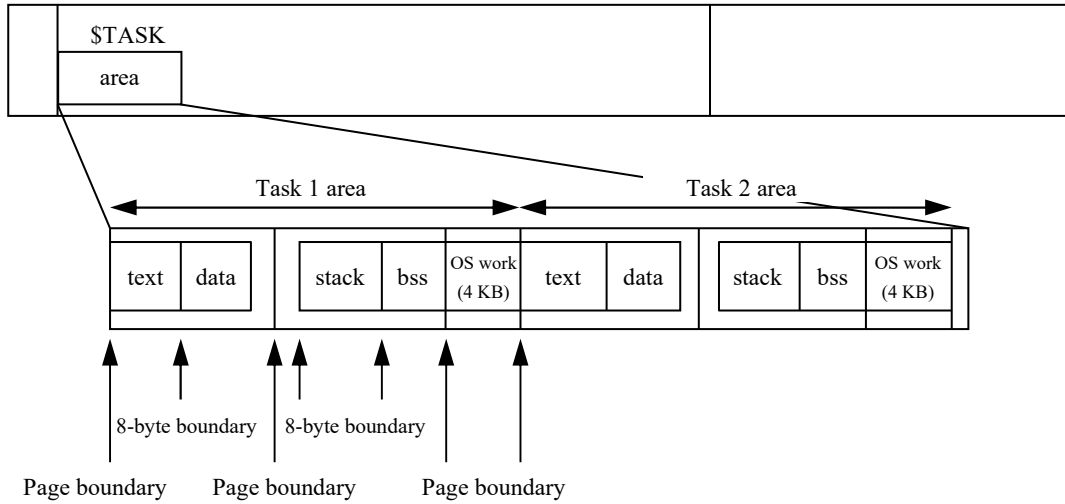


Figure 1-8 Arrangement of Tasks in Logical Space

● Stack arrangement for a multitask

For multitasks, place a stack and bss on different pages.

The OS work and stack are to be allocated as needed for all multitasks at the time of loading. The portion to be used is specified by the user at the time that the task is generated. (In the figure below, task 1 and task 2 are multitasks.)

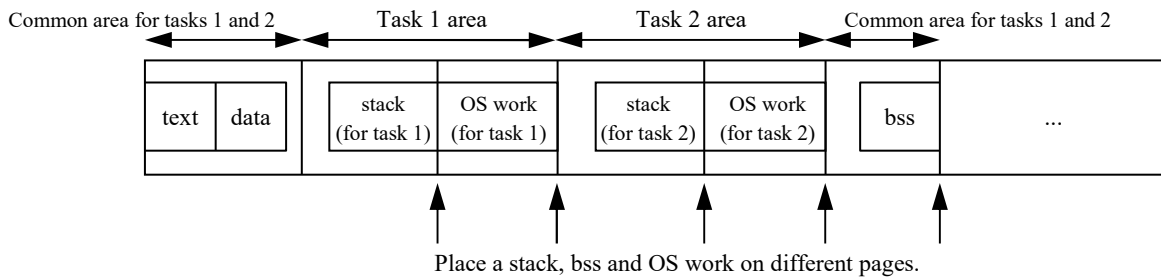


Figure 1-9 Arrangement of Tasks in Logical Space (Multitasks)

A multitask is a set of  $n$  tasks that are generated for one program to reduce the memory required for the program. Multitasks are implemented by sharing the text, data, and bss parts of the loaded programs, and by placing the stack parts in a different area for each task. Note that when a task within a multitask writes data to the bss part, the written data is applied to the other tasks within the multitask. As a result, tasks that operate with the expectation of the initial bss state might not operate correctly. For this reason, do not write data to the bss part of a multitask.

**2.5 Area Allocation for IRSUBs**

All IRSUBs running on the CPMS are placed in a single logical space. The IRSUBs are stored in a split area allocated to \$IRSUB. Multiple IRSUBs can be stored in a single split area. Place text on a 32-byte boundary and data on an 8-byte boundary.

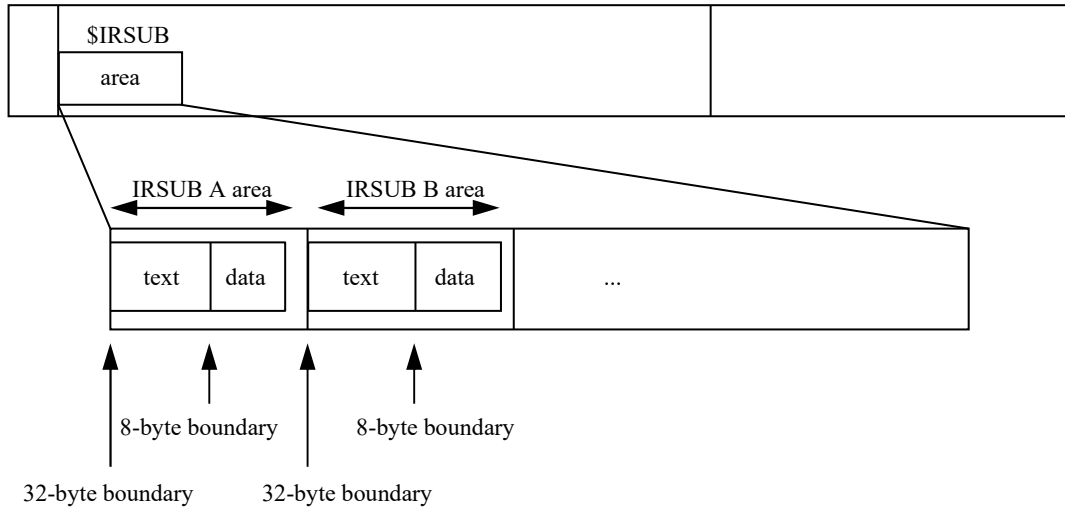


Figure 1-10 IRSUB Arrangement in Logical Space

● Multi-entry arrangement

The same text/data arrangement applies to multi-entry IRSUBs.

Multi-entry names and relative entry addresses are managed by retaining them in a management file of RPDP. Figure 1-11 shows how multi-entry IRSUBs (with entry names "A", "B", and "C".)

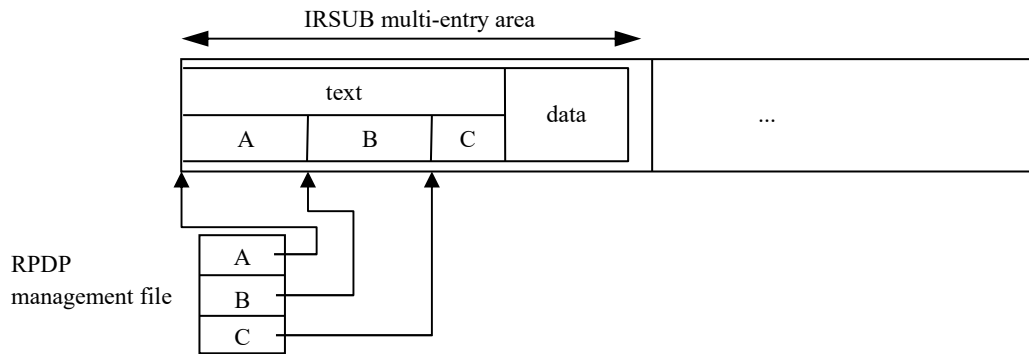


Figure 1-11 IRSUB Arrangement (Multi-entry) in Logical Space



## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

### 2.6 Loading Programs and Creating Tasks

The loader (`svload`) loads programs and data in areas or sareas based on management information determined by the allocator.

The loader creates executable modules while retrieving information about CPMS resources, such as global data, from area management information, and specifying the referenced information. The created executable modules are stored in backup files on the disk in the development machine.

Executable modules are loaded as programs. Register them as tasks by using the builder (`svctask`). `svctask` sets task attributes in a CPMS-managed table called the task control block (TCB).

### 2.7 Indirect Link Resident Subprograms

Suppose that a task consists of several subprograms. Of these, subprograms that are part of the task itself are called internal subprograms (ISUBs). In contrast, resident subprograms (RSUBs) are subprograms that are outside the task and that reside in the main memory so that they are shared by other tasks.

RPDP supports indirect link RSUBs (IRSUBs). IRSUBs are provided with a management table linked to tasks. The IRSUB itself can easily be changed without changing the tasks linked to the management table.

The loader (`svload`) updates IRSUBs, while the builder (`svbuild`) updates the indirect link management table.

### 2.8 Global (GLB)

The CPMS supports GLB so that the main memory can be shared by tasks. An area for GLB is allocated by the allocator in advance in the GLB space in the logical space dedicated to CPMS tasks. A name is assigned to the area so that multiple tasks and subroutines can share it. The area is divided into areas by `svdfa`, each of which is further divided into sareas by `svdfs`.

### 2.9 Inter-PU Shared Memory (CM)

The CM is provided in the CPMS for sharing the main memory between PUs in the same unit. In the same way as global (GLB), an area is allocated in the CM space in the logical space of CPMS tasks, and this area is named by using the allocator in advance so that it can be shared by tasks and subroutines in the same unit. The area is divided into areas by `svdfa`, each of which is further divided into sareas by `svdfs`.

To reference the common area by using the same sarea name between PUs that share the CM, split area names and secondary partition area names allocated to the CM and addresses must be defined identically in the site corresponding to each PU. To ensure that the addresses of the split areas to be allocated in the CM space in each site are the same, allocate the split areas by using the `-f` option of `svdfs`.

Allocation of split areas in each site can be made identical by specifying the relative address from the beginning of the global area `$CM` by using the `-f` option.

For details about using the CM area, see 6.1.5 Allocating split areas for the CM.

**2.10 Value (VAL)**

The user can register constants to be shared by programs as external names. These external names are called VAL data. VAL data can be registered by using `svdfv` and deleted by `svdlv`. VAL data is specified by the loader when load modules are loaded in backup files. For this reason, VAL data must be specified before tasks and subprograms that reference the VAL data are loaded.

**2.11 Indirect Link Global Data**

RPDP supports indirect link global data. A management table linked to global data is specified for indirect link global data. Updating this linked management table makes it easy to change global data.

The global data itself is updated by using the allocator (`svdfs`) and the loader (`svload`). Areas are defined by using `svdfs`, and initial values are loaded by using `svload`. The builder (`svirglb`) updates the management table for indirect linking.

**2.12 Programming Guide for GLB, VAL, and IRSUB**

This section describes the coding and linking methods for GLB, VAL, and IRSUB that are used by programs and subprograms. (CM can be used in the same way as for GLB.)

(1) Assigning names to GLB and VAL

Table 1-8 Assigning Names to GLB and VAL

Item	Specifications
Maximum number of characters	14 characters (not counting <code>_g</code> and <code>_v</code> )
Naming rules	Single-byte alphanumeric characters and underscores ( <code>_</code> ). However, the first character must be an alphabetic character. The last character, which indicates the attribute, must be in the following form: GLB: <code>_g</code> VAL: <code>_v</code>
Uniqueness	Identical names cannot be used.

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

### (2) Using GLB and VAL names

Table 1-9 shows how to use GLB and VAL names.

Table 1-9 Using GLB and VAL Names

No.	Item	C language
1	To declare GLB (on the referencing side)	extern long name_g[size]; Explanation: name: Global data name size: Global data size
2	To reference GLB (specification of a secondary partition area name)	extern long name_g[size]; main() { long i; i = name_g[index]; } Explanation: name: Global data name size: Global data size
3	To declare GLB (on the referenced side)	There is no need to declare GLB. Set initial values as shown in 4.
4	To set initial values in GLB	long name_g[size] = {1,2,3,...}; Explanation: name: Global data name size: Global data size
5	To reference VAL values	extern long name_v; long y = (long) &name_v; main() { long x; x = y; } Explanation: name: VAL name

## (3) Notes on GLB data reference

When GLB data is referenced at program creation, data handling varies depending on whether the GLB to be referenced has a defined initial value in the same program. For this reason, create a program to reference the GLB data, taking into consideration the following points:

## 1. If the GLB to be referenced has not been defined in the same program

The above condition is met if the GLB to be referenced in the source program that becomes an object file to be linked by the source program or `svload` is not defined as shown in number 3 or 4 in Table 1-9.

In this case, take (a) to (b) into consideration.

## (a) GLB declaration

For GLB declaration, a capacity declaration for each name can be performed as shown in number 1 in Table 1-9. The compiler and assembler do not perform a rationality check of this capacity with respect to the size of the area reserved by the `svdfs` command. For this reason, if the program references an address that exceeds the actual area, no error is generated.

Example: Referencing an address that exceeds the declared area

Allocator

```
svdfs usrresp0 glb2 100
C
extern long glb2_g[100];
  ⋮
  ⋮
  glb2_g[100]= .....; } No error is detected.
```

## (b) Reference of a relative address

A GLB reference can be performed in the form of name  $\pm\alpha$  (where  $\alpha$  is a relative byte address). In this case, the range is  $-2^{31} \leq \alpha \leq 2^{31} - 1$ .

## 2. If the GLB to be referenced is defined in the same program

The above condition is met if the GLB to be referenced in the source program that becomes an object file to be linked by the source program or `svload` is defined as shown in number 3 or 4 of Table 1-9.

In this case, take (a) to (c) into consideration.

## (a) Reference of the GLB name only

When referencing only a GLB name that has the initial value defined in the program, there are no particular constraints.

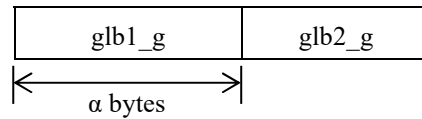
Example: Referencing a name only

```
C
extern long glb2_g ;
long glb1_g[3] = { (long)&glb2_g , 0 , 100 } ;
  ⋮
glb2_g = glb1_g[0] ;
```

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

### (b) Referencing relative addresses from the beginning of GLB

When referencing in the form of name +  $\alpha$ , the  $\alpha$  value cannot exceed the defined range. Note that out-of-range check errors are not detected.



Namely, when  $\text{glb1\_g} + \beta$  is indicated,  $0 \leq \beta < \alpha$  is required.

Example: Referencing a relative address

C

```
int glb1_g[3] = { 1, 2, 3 } ;
int glb2_g[2] = { (long)&glb1_g[0], 0 } ; ... The relative address is
                                                within the range.
```

```
int glb1_g[3] = { (long)&glb2_g[4], 0 } ; ... The relative address is
                                                outside the range.
```

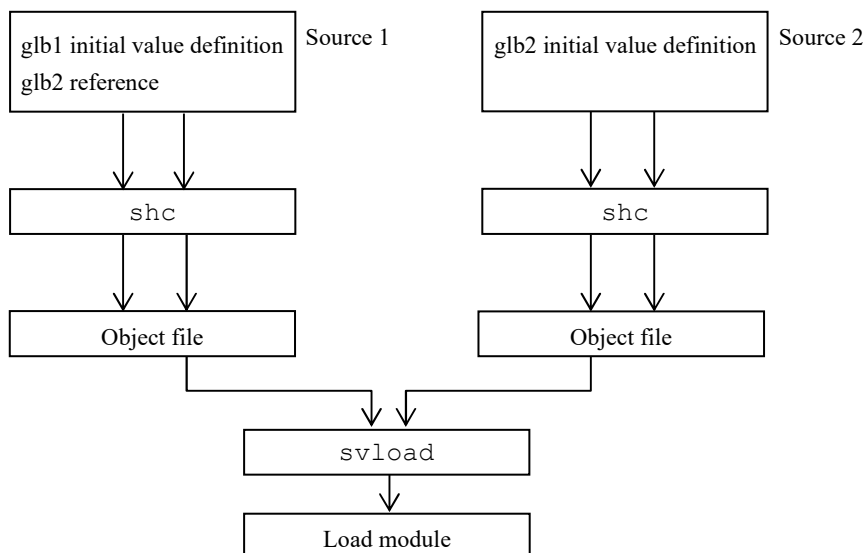
### (c) Notes on operating svload

Suppose that a load module was created from a source program that has a text part and GLB initial data. If the load module is then loaded by `svload` simply as a program or subprogram, no initial values are loaded. To load the initial-value data, run `svload` again by using the `+D` option. That is, with respect to a single load module, `svload` must be run twice by using different options. We recommend that you create a file for initial-value data only, and perform execution separately. You can define multiple GLB initial values in a source program.

### 3. Notes on linking

As already described in 1 and 2, when multiple object files are linked by using `svload`, they are regarded as linked files even if they are different source files.

Example: Linking two object files



In this example, source 1 and source 2 are regarded as a source program.

4. When the GLB initial value to be referenced is defined within the same program  
 When GLB data is to be referenced at the time of program creation, the GLB initial value to be referenced must not be defined within the same program.

If a GLB initial value definition and its reference are in the same program, the reference is handled as a reference to local data.

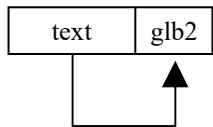
Example 1: When the same program contains both a program and GLB data

```
a.c
long glb2_g = 8;

main(){
long x;
    x = glb2_g;
}
```

When the program is as shown to the left, the glb2 reference from the program is handled as a reference to the data part.

svload +P -o pname a.obj



glb2 reference

GLB initial-value data is loaded as another program.

Example 2: When a program and GLB data are separated from each other

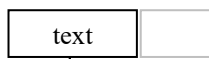
```
a.c
extern long glb2_g ;

main(){
long x;
    x = glb2_g;
}
```

```
b.c
long glb2_g = 8;
```

svload +P -o pname a.obj

svload +D b.obj



GLB area

glb2

glb2 reference

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

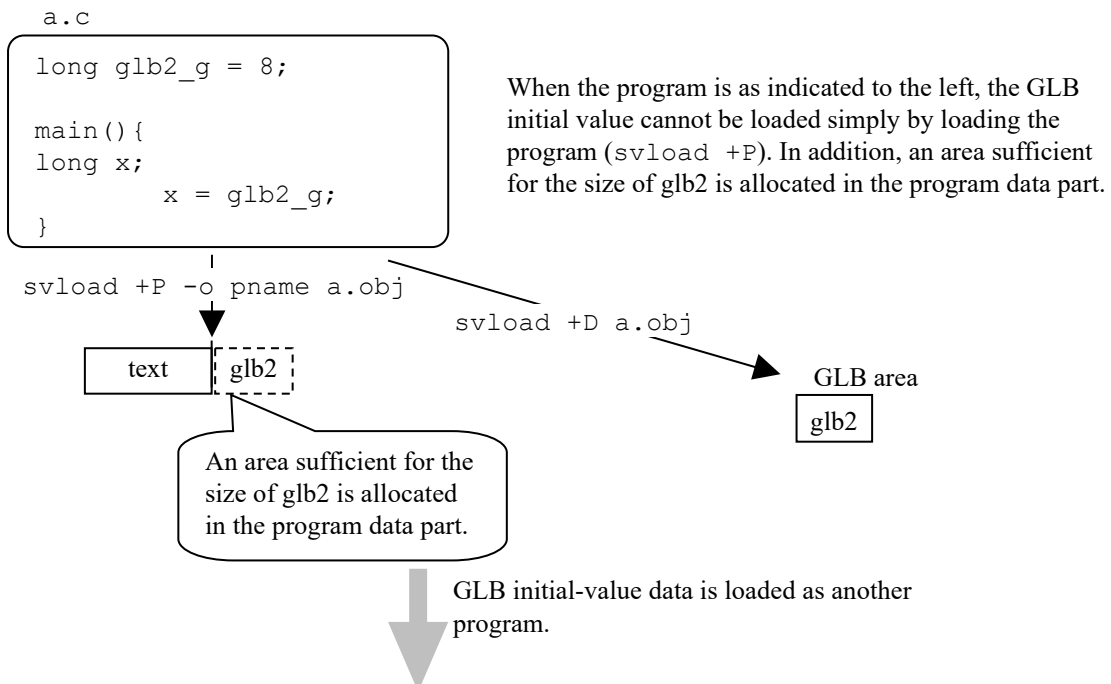
### (4) Notes on GLB data loading

#### 1. Coexistence of a program and GLB data

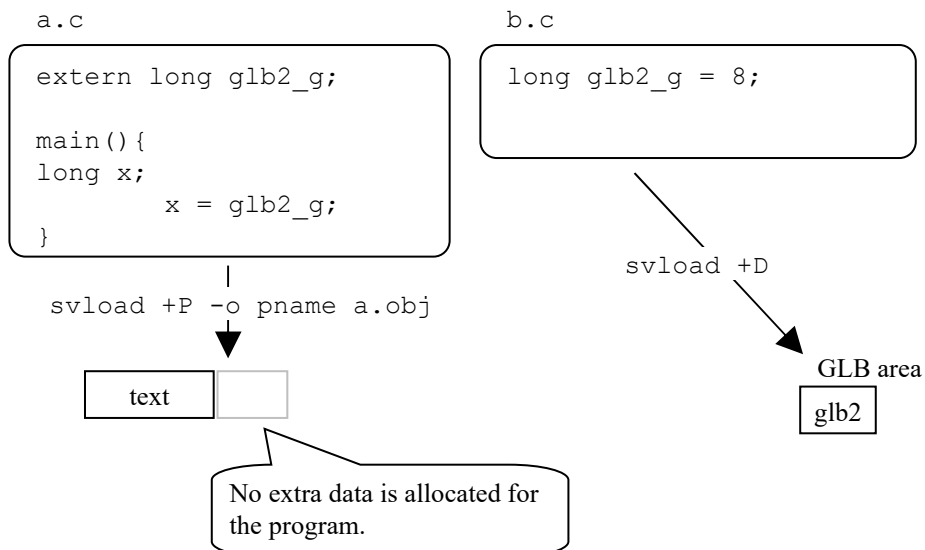
In situations where a program and subprogram that have a text part are included with GLB initial-value data in the same program, GLB initial-value data is not loaded even if the loader (svload) loads the program and subprogram as a program (+P) and subprogram (+I). To load the GLB initial-value data, load again by specifying the +D option. That is, the program is loaded twice by specifying different options.

In addition, an area sufficient for the size of the GLB initial-value data is allocated in the program and subprogram data part. For this reason, the program for defining the GLB initial-value data only needs to provide an initial value definition for GLB.

Example 1: When a program and GLB data coexist (are contained in the same program)



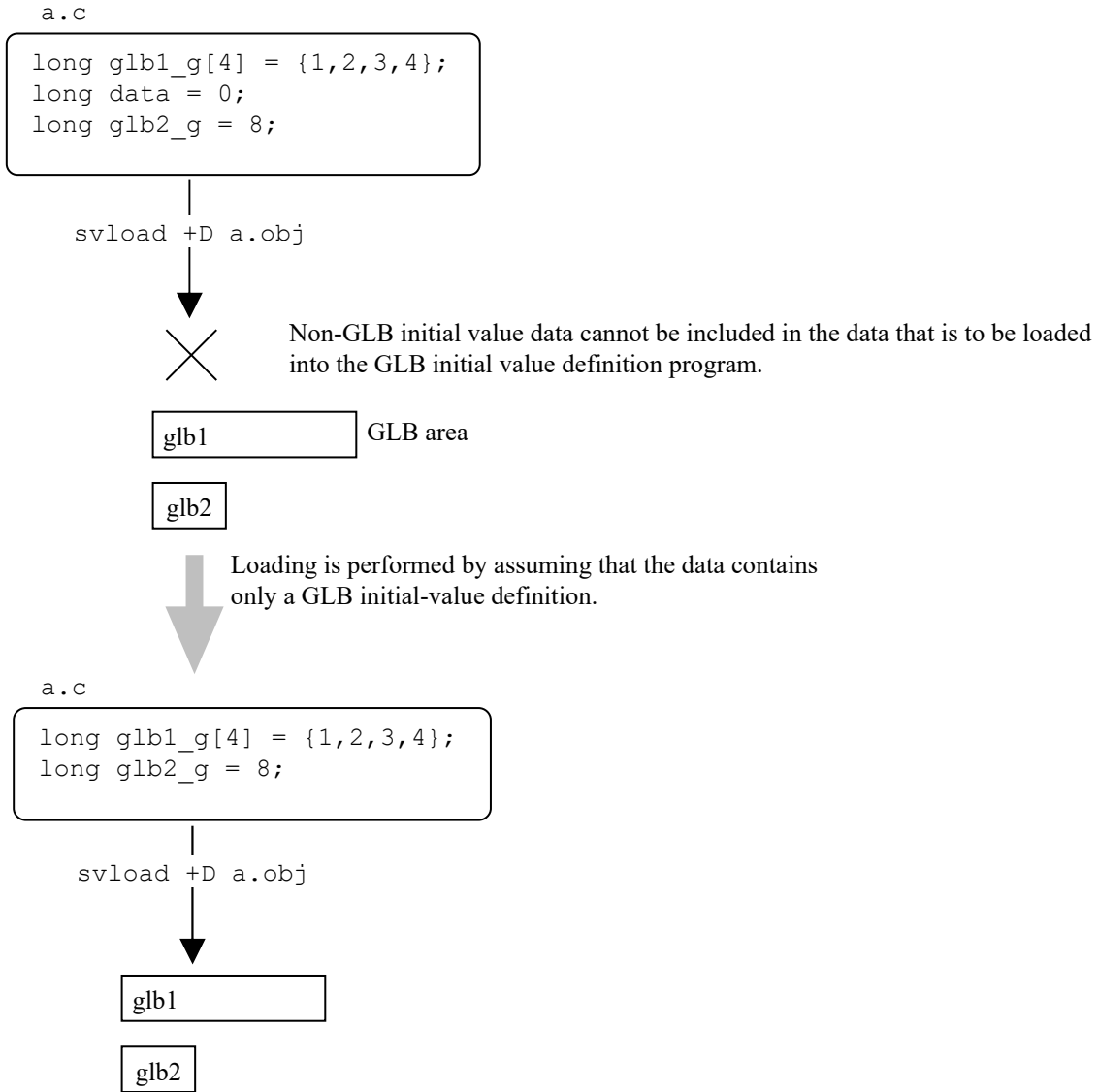
Example 2: When a program and GLB data are separated from each other



2. Coexistence of GLB initial values and non-GLB data

Multiple sets of GLB initial-value data can be defined in a single program. However, no data other than a GLB initial value can be defined in a GLB initial-value definition program.

Example: GLB initial value definition and non-GLB data coexist in the same program





## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

### (5) IRSUB usage

Table 1-10 IRSUB Usage

No.	Item	C language
1	IRSUB reference (IRSUB name selection)	<pre>main() {     name (); }</pre> <p>Explanation: name: IRSUB name</p>
2	IRSUB reference (irsubad function selection)	<pre>void *irsubad(); main() {     long no;     long (*adr) ();     no = xxx;     adr = irsubad(no);     if(adr != 0) {         (*adr) ();     }else{         /* Unregistered IRSUB processing */     } }</pre> <p>Explanation: xxx: IRSUB number The irsubad function is used to reference an address in an indirect link table.</p>

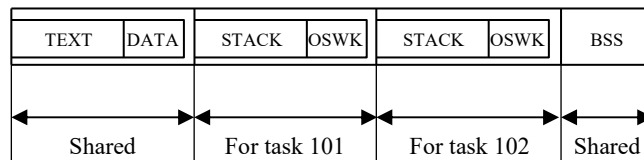
(6) Generating a multitask

To generate a program as a multitask consisting of two tasks, run the `svload` and `svctask` commands as follows:

```
a.c
extern long glb2_g;

main() {
long x;
    x = glb2_g;
}
```

```
svload +P -o pname -w 4096 M 2 a.obj
svctask pname task101 101 -r 1
svctask pname task102 102 -r 2
```

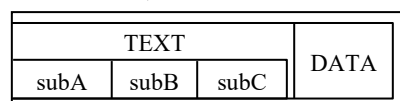


(7) Generating a multi-entry IRSUB

To generate a subprogram as a multi-entry IRSUB containing three entries, run the `svload` and `svbuild` commands as follows. The following example assumes that a single-source file contains three modules.

```
subA.c
subA() {
long X;
    X = 10;
}
subB() {
long X;
    X = 20;
}
subC() {
long X;
    X = 30;
}
```

```
svload +I -o subA -m subB subC subA.obj
svbuild subA -ir -e 101
svbuild subB -ir -e 102
svbuild subC -ir -e 103
```



### 2.13 Constraints on CPMS Program Creation

The following constraints apply to the creation of CPMS programs.

(1) Overlay structures are not allowed.

For the CPMS, an overlay structure for tasks or resident subprograms is not allowed. Accordingly, when creating a task or resident subprogram, be careful that the program does not become large.

(2) Bulk subroutines are not supported.

The CPMS does not support bulk subroutines that are operated by storing a subprogram in the auxiliary memory and loading it in the main program as required. Use an indirect resident subprogram (IRSUB) or internal subprogram (ISUB) built into a task.

(3) Notes on indirect resident subprogram (IRSUB) creation

The IRSUB resides in the main memory unit and is shared by multiple main programs. Accordingly, the IRSUB occupies a main memory area independent of the main programs that use it. Because the IRSUB is used by multiple main programs at the same time, make it reentrant.

A non-reentrant program cannot be an IRSUB. The term *reentrant* means that when the IRSUB is used by a main program, the same IRSUB can be used by another main program.

The following describes the correct procedure for creating an IRSUB.

A reentrant IRSUB is divided into an invariable part, which consists of a text part and a data part, and a variable part, which consists of work areas. The invariable part is shared among multiple main programs.

Each main program reserves its variable part, and the IRSUB uses the variable part prepared for the main program. Accordingly, program the variable part to be used by the IRSUB so that the stack area can be referenced. The IRSUB cannot use a work area (bss part) without an initial value.

When creating a reentrant IRSUB, consider the following three points:

- (a) Make all work areas stack areas.
- (b) When the IRSUB consists of multiple programs, do not use an area shared by different programs.
- (c) When the defined initial value is a static variable, do not change the value.

The conditions indicated by (a) and (b) above can be verified by checking whether the section information in the compilation list or linkage map list indicates a B section size of 0.

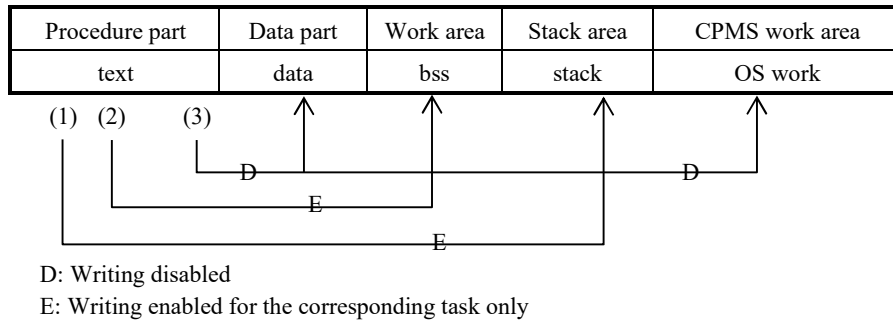


Figure 1-12 Enabling or Disabling Writing

The number (1) indicates writing to the stack area. The corresponding task can be written to the stack area.

The number (2) indicates writing to the work area. Under normal circumstances, in the IRSUB, do not reserve a work area or write data. The corresponding task can be written to the work area.

The number (3) indicates writing to the data part. Tasks cannot be written to the data part.

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

The following are notes on reentrant IRSUB creation in each language.

Example C program:

```
int b1 ; ..... (1)
int d1 = 10 ; ..... (2)
static int b2 ; ..... (3)
static int d2 = 100 ; ..... (4)
ex() {
static int b3 ; ..... (5)
static int d3 = 1000 ; ..... (6)
int s1 ; ..... (7)
int s2 = 20 ; ..... (8)

    ⋮

}
```

The program that performs writing for b1 declared in (1) becomes non-reentrant. The program that performs writing for d1 declared in (2) becomes non-reentrant. The program that performs writing for b2 declared in (3) becomes non-reentrant. The program that performs writing for d2 declared in (4) becomes non-reentrant. The program that performs writing for b3 declared in (5) becomes non-reentrant. The program that performs writing for d3 declared in (6) becomes non-reentrant. If writing is performed for s1 or s2 declared in (7) or (8), the reentrant characteristics of the program are not lost. To use as an IRSUB, create by using only the variables shown in (7) or (8).

The following describes the area to which each variable is assigned.

Usually, b1 is assigned to the bss area. (\*)

b2 is assigned to the bss area.

b3 is assigned to the bss area.

d1 is assigned to the data area.

d2 is assigned to the data area.

d3 is assigned to the data area.

s1 is assigned to the stack area.

s2 is assigned to the stack area.

(\*) In other programs, when the initial value is set in b1, it is assigned to the data area.

- (4) Relocation of programs is not possible.  
 Program and subprograms are not relocated. A program or subprogram for which the run area is specified cannot be run in another area. To run a program or subprogram in another area, delete the program or subprogram, and then re-register the program or subprogram.
- (5) Names can include a maximum of 14 characters.  
 For program and subprogram names, use a maximum of 14 single-byte alphanumeric characters and underscores (\_). GLB names and VAL names can also include a maximum of 14 characters. For the representation of GLB or VAL in C, use a maximum of 16 characters for the name, including the `_g` and `_v` parts attached to the end.
- (6) Names for GLB or VAL  
 When a name that ends with `_g` or `_v` is declared as an external name, it is interpreted as a GLB or VAL name. Accordingly, ensure that programs that do not use GLB or VAL data do not have names that end with `_g` or `_v`. In addition, do not use names that end with `_b`, because such names are reserved for a future extension.
- (7) Make external names unique.  
 Make external names unique among all GLB names, program names, subprogram names, and VAL names in the system.
- (8) Some names cannot be used.  
 There are some names that cannot be used and some names that required extra caution when used to create programs. For details, see APPENDIX A NAMES USABLE IN PROGRAMS.
- (9) Program structure  
 The structure of programs that run on the CPMS is as follows:

text	Area for program procedures
data	Area for initial program data
stack	Dynamic work area used by tasks
bss	Static work area used by programs
OS work	Dynamic work area used by the OS

The sizes of these areas are corrected to a 4-byte integer. Furthermore, each of these areas is arranged so that its start address is a multiple of 8 or 4096. For details on memory allocation, see section 2.5.

- (10) Constraints on start addresses  
 By default, the allocator corrects the GLB area so that its start address is a multiple of 4.

## 2. PROCEDURES FOR PROGRAM DEVELOPMENT

### (11) Handling of initial values

Note the following with respect to the handling of initial values.

Area	CPMS
data	Programmed value
bss	Unfixed
stack	Unfixed

### (12) Size of the initial data of GLB and CM

Note that, after the compiler completes alignment, the data in the object file might become larger than the size defined in the source program. The following is a specific example.

To speed data access, CPMS uses natural alignment, in which fixed addresses are used according to the data type. Because the compiler or linker automatically places data, alignment does not need to be accounted for in user code. Note, however, that the actual size of the initial data in GLB and CM might exceed the size of the coded structure, as shown in the following example.

Size of code: 16 bytes

Size of initial data in memory: 24 bytes

```

struct{
    char    a;
    short   b;
    long    c;
    char    d;
    double  e;
}ABC;
    
```

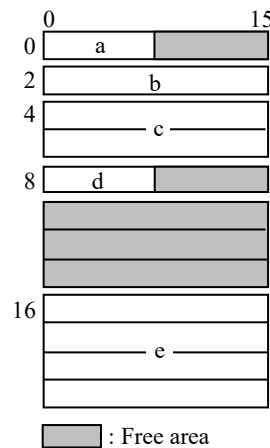


Figure 1-13 Comparison of Data Sizes

Action

- (1) When configuring the structure, consider the order in which data is placed so no free areas remain.

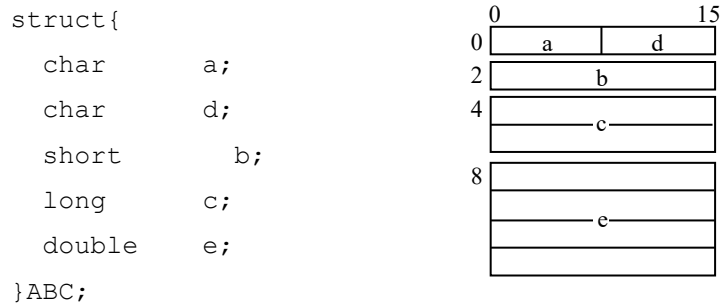


Figure 1-14 Sample Declaration of a Structure with Data Relocation Taken into Account

- (2) If free areas must remain, explicitly declare that the structure contains one or more free areas. (If you do not do so, free areas might or might not be allocated depending on the machine.)

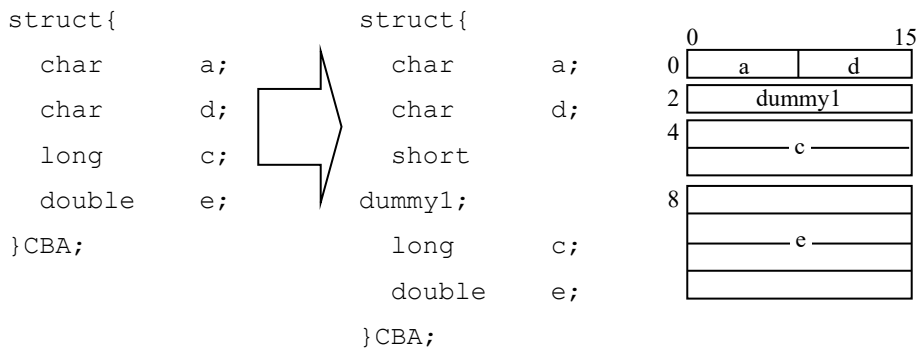


Figure 1-15 Example of Explicitly Declared Free Areas

- (3) The structure size in bytes must be a multiple of the maximum type in the structure. When allocating structure arrays, make sure that the first address of the second or subsequent structure is an accessible address.

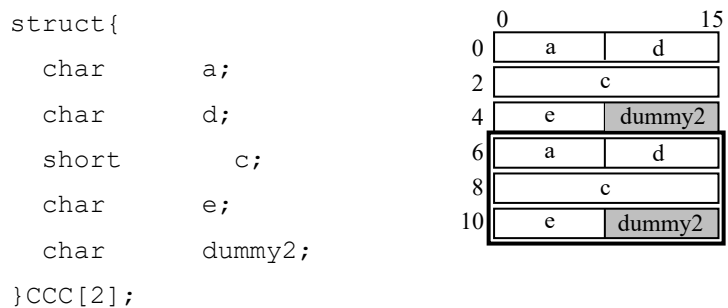


Figure 1-16 Sample Declaration with Structure Size Taken into Account



## CHAPTER 3 INSTALLATION AND EXECUTION ENVIRONMENT

### 3.1 Installation

Install the RPDP/S10VE software product that includes RPDP by using the installer. Insert the RPDP/S10VE disc into the CD drive, and then from the Explorer, run `setup.exe` in the S789810 folder of the CD drive. Log in to the system as an administrator, and then start installation.

### 3.2 Prerequisite Software Products

The prerequisite software products listed in Table 1-11 are required to use RPDP. Furthermore, the SHC compiler (Ver. 9.04 Release 00) is required to compile and load programs by using RPDP.

Table 1-11 Prerequisite Software Products of RPDP

Name	Model
CPMS/S10VE	S-7898-05
RCTLNET/S10VE	S-7898-60
RPDP/S10VE	S-7898-10
BASE SYSTEM/S10VE	S-7898-38

### 3.3 Notes on Installation

#### 3.3.1 Notes on installing RPDP

After RPDP is installed, log in to the system again.

To reinstall RPDP, open the **Control Panel**, select **Uninstall a program**, uninstall the software, and then reinstall RPDP/S10VE.

#### 3.3.2 Notes on installing the SHC compiler

To install the SHC compiler (Ver. 9.04 Release 00), install it in the default installation destination (C:\Program Files (x86)).

### 3.4 RPDP Execution Environment

#### (1) Execution environment setup file

Environment settings are required to use RPDP. The RPDP execution environment of S10VE is specified by the setup file

(%SystemRoot%\renix\usr\rpdp\_hce\etc\RPDP.ini). When RPDP/S10VE is installed, a default setup file is created. To use values that differ from the default values, modify the setup file data after installing RPDP.

Table 1-12 Values Specified for the S10VE RPDP Execution Environment

No.	Environment variable	Setting description	Default value	Remarks
1	SHCPU	Sets the CPU type. You must specify SH4.	SH4	
2	SHC_INC	Sets the include file storage directory of the compiler.	%ProgramFiles(x86)%\Renesas\Hew\Tools\Renesas\Sh\9_4_0\include	
3	SHC_LIB	Sets the installation directory of the compiler.	%ProgramFiles(x86)%\Renesas\Hew\Tools\Renesas\Sh\9_4_0\bin	
4	SHC_TMP	Sets the directory in which the compiler creates temporary files.	%SystemRoot%\renix\tmp	
5	HLNK_DIR	Sets the library search path of the svload command of RPDP for S10VE.	%SystemRoot%\renix\S10VE\lib	
6	HLNK_TMP	Sets the directory in which the linkage editor creates temporary files.	%SystemRoot%\renix\tmp	

#### (2) Setting the environment variable PATH

For the environment variable PATH, set the command path of RPDP. When RPDP/S10VE is installed, the default value %SystemRoot%\renix\S10VE\bin;

%ProgramFiles(x86)%\Renesas\Hew\Tools\Renesas\Sh\9\_4\_0\bin is set at the beginning of PATH.

### 3. INSTALLATION AND EXECUTION ENVIRONMENT

#### 3.5 Registering an RPDP User Account

To use RPDP, log on to the system by using the account belonging to the dedicated group (the RPDPUsers group). You can create a new account that belongs to the RPDPUsers group or make an existing account a member of the RPDPUsers group.

##### 3.5.1 Registering a new account

- (1) Log in as the Administrator.
- (2) From the **Control Panel**, select **Administrative Tools** and then **Computer Management**.
- (3) Double-click **System Tools, Local Users and Groups**, and then **Users** in the console tree (the left pane) to display the users list.
- (4) Register the dedicated account as follows:
  1. From the **Action** menu, select **New User**. The New User dialog box appears.
  2. In the New User dialog box, enter the user name and other necessary items, and then register the new account. You can enter a user name and password of your choice.
  3. Double-click the created user to display the user properties dialog box.
  4. Select the **Member Of** tab and click the [Add] button to display the Select Groups dialog box.
  5. Click the [Advanced] button, and then click the [Search] button to display a list of groups. In the **Name (RDN)** column, select RPDPUsers from the list, and then click the [OK] button to add RPDPUsers.  
The RPDPUsers group is automatically registered when RPDP/S10VE is installed.
  6. Click the [OK] button to close the Select Groups dialog box.
  7. Click the [OK] button to close the user properties dialog box.

## 3.5.2 Adding RPDUsers as a group to which an existing account belongs

- (1) Log in as the Administrator.
- (2) From the **Control Panel**, select **Administrative Tools** and then **Computer Management**.
- (3) Double-click **System Tools, Local Users and Groups**, and then **Users** in the console tree (the left pane) to display the users list.
- (4) Add RPDUsers as a group to which the account belongs as follows:
  1. Double-click the user who you want to belong to the RPDUsers group. The user properties dialog box appears.
  2. Select the **Member Of** tab and click the [Add] button to display the Select Groups dialog box.
  3. Click the [Advanced] button, and then click the [Search] button to display a list of groups. In the **Name (RDN)** column, select RPDUsers from the list, and then click the [OK] button to add RPDUsers.  
The RPDUsers group is automatically registered when RPD/S10VE is installed.
  4. Click the [OK] button to close the Select Groups dialog box.
  5. Click the [OK] button to close the user properties dialog box.

## CHAPTER 4 COMPILER

This chapter describes the S10VE compiler and assembler in detail. For the command reference, see PART 2 COMMAND REFERENCE.

With respect to compiler and assembler use, RPDP assumes the use of Renesas Microcomputer Development Environment System SuperH RISC engine C/C++ Compiler Package Ver. 9.04 Release 00 (hereinafter abbreviated to *the shc compiler*).

### 4.1 Details of C Compiler Options

This section describes how to compile with `shc`, as well as points to note when compiling. For detailed specifications for `shc`, see the documentation included with the `shc` compiler.

- Command syntax

```
shc [ $\Delta$ option...][ $\Delta$ file-name[ $\Delta$ option...]...]
```

Example: `shc $\Delta$ test1.c $\Delta$ test2.c`

- Setting up the RPDP execution environment

For details on the execution environment settings, see 3.4 RPDP Execution Environment.

- Setting up the `shc` execution environment

To directly use the `shc` compiler, the environment variables necessary for the `shc` compiler operations described in Table 1-13 must be set.

To directly use the Ver. 9.04 `shc` compiler, environment variables settings must be specified for Ver. 9.04.

- Setting up the interface used with the CPMS

To use the user interface with the CPMS, you need to specify the include file storage directory to be used with S10VE (`%windir%\renix\s10ve\include`), which is shown in Appendix C, in the search directory of the include file.

The search directory can be specified by using the `SHC_INC` environment variable or the `include` option of the `shc` compiler.

Table 1-13 Environment Variables Required for shc Compiler Operation

No.	Environment variable	Setting description
1	path	<p>Add the executable file storage directory of the installed compiler package to the path environment variable.</p> <p>You need to set the paths of both the compiler (shc) and optimization linkage editor (optlnk).</p> <p>This environment variable must always be specified.</p> <p>Specification format:</p> <pre>path= executable-file-path [ ; existing-path ; ...]</pre>
2	SHC_LIB	<p>Specify the directory in which the load module and system include file of the compiler are stored.</p> <p>This environment variable must always be specified.</p> <p>Specification format:</p> <pre>set SHC_LIB= executable-file-path</pre>
3	SHCPU	<p>Specify the target CPU type. For this system, set SHCPU to SH4.</p> <p>When this environment variable is not specified, the system assumes that the CPU type is SH1. The CPU type can also be specified using the <code>-cpu</code> option.</p> <p>Specification format:</p> <pre>set SHCPU= CPU</pre>
4	SHC_INC	<p>Specify the include file storage directory of the compiler. The system include file search will be conducted in the following order of locations: the directory specified by the <code>include</code> option, the directory specified by SHC_INC, and the system directory (SHC_LIB).</p> <p>The user include file search will be conducted in the following order of locations: the current directory, the directory specified by the <code>include</code> option, and the directory specified by SHC_INC.</p> <p>Specification format:</p> <pre>set SHC_INC= include-path [ ; include-path ; ...]</pre>
5	HLNK_DIR	<p>Specify the input file storage directory for the optimization linkage editor.</p> <p>Files specified by the <code>input</code> option and <code>library</code> option will be searched for in the following order of locations: the current directory and the directory specified by HLNK_DIR.</p> <p>The library search path of the loader also follows the HLNK_DIR setting.</p> <p>Specification format:</p> <pre>set HLNK_DIR= input-file-name [ ; input-file-name ; ...]</pre>
6	SHC_TMP	<p>Specify the directory in which the compiler creates temporary files.</p> <p>Specification format:</p> <pre>set SHC_TMP= directory</pre>
7	HLNK_TMP	<p>Specify the directory in which the linkage editor creates temporary files.</p> <p>Specification format:</p> <pre>Set HLNK_TMP= directory</pre>

## 4. COMPILER

### 4.2 Notes on Compiling

#### 4.2.1 Compiling by using shc

- Handling of floating-point numbers

The shc compiler can control denormalized numbers and the rounding of floating-point numbers by using compilation options.

However, note that the standard libraries to be linked during loading vary depending on their handling. The following shows the options that control handling of denormalized numbers and rounding, as well as the applicable standard libraries. (When no library is specified during loading, the loader links `libsh4nbmdn.lib`.)

Table 1-14 Floating-point Number Control Options

	Specification	Option (*2)	Default	Handling in cchr
Handling of a denormalized number	Handled as 0	<code>-denormalization=off</code>	Handled as 0	Handled as a denormalized number
	Handled as a denormalized number (*1)	<code>-denormalization=on</code>		
Rounding of the result value	The part beyond the significant figures is rounded down.	<code>-round=zero</code>	Rounded down	Rounded off
	The part beyond the significant figures is rounded off.	<code>-round=nearest</code>		

(\*1) If the S10VE CPU SH4A (SH7786) is running in a mode where denormalized numbers are handled as denormalized numbers, an FPU error occurs when denormalized numbers are entered. Therefore, the SH4A is run in a mode where denormalized numbers are handled as 0.

(\*2) Specify `-denormalization=on` and `-round=nearest` to handle floating-point numbers in the same way as cchr.

Table 1-15 Handling of Floating-point Numbers and Applicable Standard Libraries

	<code>-denormalization</code>	<code>-round</code>	Standard library
Specified option	<code>off</code>	<code>zero</code>	<code>libsh4nbmzz.lib</code>
	<code>on</code>	<code>zero</code>	-
	<code>off</code>	<code>nearest</code>	-
	<code>on</code>	<code>nearest</code>	<code>libsh4nbmdn.lib</code>

- Two standard libraries are provided: one is the shc default `-denormalization=off` and `-round=zero`. The other is the cchr-compatible `-denormalization=on`, and `-round=nearest`.

- Generating and saving a compilation list (shc)

Generate and save a compilation list that is required to calculate the stack sizes to be used by tasks. To generate a compilation list, specify the following option.

Specify the `-listfile` option prior to the C source file to be compiled.

If this option is specified after the C source file, a compilation list of only the last file is generated.

- Specifying the generation of a compilation list

```
-listfile [= list-file-name] -show=source,object
```

If the list file name is not specified, a file with the same file name as the source file name (with the extension `lst` added) is created.

**Example:**

- ◆ `shc Δ-listfile Δtest1.c Δtest2.c`  
The `listfile` option is enabled for `test1.c` and `test2.c`.
- ◆ `shc Δtest1.c Δtest2.c Δ-listfile`  
The `listfile` option is enabled only for `test2.c`.

- Specifying `-fpscr=safe` when compiling a built-in subroutine

The `shc` compiler has a function to specify whether to ensure the precision mode of the FPSCR register around the time that a function is called (`-fpscr`). The default specification is such that precision is not ensured around the time that a function is called (`-fpscr=aggressive`). Therefore, a code to return the FPSCR precision mode to single-precision after a return from the function call is generated.

In the S10VE, floating-point calculations cannot be performed in the built-in subroutine. (An “FPU Unavailable” exception occurs during execution and the CPU stops.)

When the built-in subroutine is compiled with `-fpscr=aggressive` specified, an FPSCR access is made and an “FPU Unavailable” exception occurs when a function is called, even if floating-point calculations are not performed.

For this reason, when compiling the built-in subroutine (including an `IRSUB` called from the built-in subroutine), specify the `-fpscr=safe` option.



## 4. COMPILER

### 4.3 shc Version Comparisons

#### 4.3.1 Command line options

Table 1-16 shows a comparison of the command line options of shc.

Table 1-16 Comparison of Versions of the shc Command Line Options

shc	Version		Meaning
	V7	V9	
<u>-code</u> = <u>machinecode</u>	○	○	Does not provide linking. Generates an object module.
<u>-define</u> =name <u>-define</u> =name=def	○	○	Defines the name. Defines the name in the def position.
<u>-debug</u>	○	○	Generates debug information.
<u>-listfile</u> <u>-show</u> = <u>source</u> , <u>object</u>	○	○	Inserts a source file line into an assembler source.
ANSI compliance by default	○	○	Compiles only programs that are ANSI-compliant.
<u>-endian</u> = <u>big</u>	○	○	Performs compilation in big-endian mode (the default value is big).
<u>-endian</u> = <u>little</u>	○	○	Performs compilation in little-endian mode.
<u>-sjis</u> (default)	○	○	Supports kanji (Shift JIS). This option can be specified only for the K%R specification.
<u>-show</u> =length= <u>n</u>	○	○	Specifies the number of lines per source list page.
<u>-listfile</u> <u>-listfile</u> =filename	○	○	Displays a source list. However, the list contents for cchr and shc differ.
<u>-include</u> =dir	○	○	Adds an include file search directory.
<u>-optimize</u> =0 <u>-optimize</u> =1	○	●	Sets the optimization level. For shc V7 and V9: optimize=0: provides no optimization; optimize=1: provides optimization. You can select an optimization method with <u>-speed</u> , <u>-nospeed</u> , or <u>-size</u> . For shc V9: Optimization related to the deletion of individual statements is fully suppressed, and local variable information can be referenced at all times.
<u>-speed</u> <u>-nospeed</u> <u>-size</u>	○	○	
<u>-preprocessor</u> [=file]	○	○	For shc: Performs preprocessor execution only, and stores the result in the .p file.
<u>-code</u> = <u>asmcode</u>	○	○	Generates an assembler source. Does not start the assembler or linker.

The underlined portions of the shc options are the abbreviated forms that can be used when specifying the options. The italicized options are the default options that take effect when the relevant option is omitted.

#### V7 and V9 legend

- : A corresponding option exists.
- : Changes are present in V9.

#### 4.4 Data Generator

Using the data generator makes it possible to load initial values to the GLB and CM areas in an environment in which the shc compiler is not installed.

The following describes the procedure to load initial values to the GLB and CM areas without using the compiler.

1. Create a text file that contains edition data to be loaded.
  2. Convert the text file to binary data by using the data generator command (svdatagen).
  3. Load initial values to the backup file by using the loader (svload).
- For the specifications of the text files that contain data, see (a) svd atagen input specifications, (b) Differences between C language declaration statements and restrictions, (c) Example of a valid input file, (d) Examples of invalid input files, (e) Restrictions on preprocessor functions, and (f) Initial value type conversion specifications.

The following describes loader (svload) operations to load the binary data generated by the data generator (\*.bin) to the backup file.

```
svload +B xx.bin
  +B: Loads the initial value data generated by svdatagen.
  xx.bin: Specifies the binary file generated by svdatagen.
```

The following describes svcomp operations to compare the binary data generated by the data generator (\*.bin) with the GLB data loaded in the backup file.

```
svcomp +B xx.bin
  +B: Compares the initial value data generated by svdatagen with the initial GLB
  value.
  xx.bin: Specifies the binary file generated by svdatagen.
```

## 4. COMPILER

### (a) svdatagen input specifications

#### Input file specifications:

The input file specifications are a subset of the declaration statements of the C language.

The following describes the syntax of the data that can be included in the input file.

The `#include` and `#define` preprocessor reference terms can also be included. For the specifications and restrictions of preprocessor reference terms, see (e) Restrictions on preprocessor functions.

#### Declaration:

```
extern type specifier array[] ;  
Type specifier declarator = initial-value ;  
Structure specifier ;
```

#### Type specifiers:

```
void  
char  
short  
int  
long  
float  
double  
signed  
unsigned  
Structure specifier
```

#### Structure specifiers:

```
struct { structure-member-declaration-list }  
struct tag { structure-member-declaration-list }  
struct tag
```

#### Structure member declaration list:

```
Type specifier declarator ;
```

#### Declarator:

```
Variable  
Array [size]  
*Variable  
*Array [size]
```

#### Initial values:

```
Assignment expression  
{ initial-value-list }  
{ initial-value-list, }
```

#### Initial value list:

```
Initial-value  
Initial-value-list, initial-value
```

#### Assignment expression:

```
Constant
```

#### Constants:

```
Integer constant  
Character constant  
Floating-point constant  
GLB name  
VAL name
```

## (b) Differences between C language declaration statements and restrictions

The following describes the differences between the C language declaration statements, as well as restrictions.

- (1) Only the preprocessor functions `#define` and `#include` can be used with restrictions.  
For details about restrictions, see (e) Restrictions on preprocessor functions.
- (2) Only declaration statements can be included in the input file.
- (3) Only the storage class specifier `extern` can be included.  
(See Example 2 in (d) Examples of invalid input files.)
- (4) Type names from `union`, `enum`, or `typedef` cannot be used for the type specifier.  
(See Examples 3 and 4 in (d) Examples of invalid input files.)
- (5) The bit field cannot be used.  
(See Example 5 in (d) Examples of invalid input files.)
- (6) Type modifiers (`const` and `volatile`) cannot be used.  
(See Example 6 in (d) Examples of invalid input files.)
- (7) Only variables, arrays, pointers, and pointer arrays can be included in a declaration. Pointers to functions and pointers to arrays cannot be included.  
(See Example 7 in (d) Examples of invalid input files.)
- (8) The array size cannot be omitted. Declare the required size.  
(See Example 8 in (d) Examples of invalid input files.)
- (9) Only constants and GLB or VAL external names can be provided as initial values. No expressions can be included.  
However, an expression consisting of the unary operators `+` (plus) and `-` (minus), as well as constants (decimal numbers or floating-point constants), can be included.  
(See Example 9 in (d) Examples of invalid input files.)
- (10) No octal values, hexadecimal values, or character constants can be included after a `+` (plus) or `-` (minus) sign.  
(See Example 10 in (d) Examples of invalid input files.)
- (11) Enumeration constants and character string constants cannot be provided as constants. Initialize a character string sequentially at the individual character level by using character constants.  
(See Example 8 in (d) Examples of invalid input files.)
- (12) Codes that are two or more bytes (including kanji and abcd) cannot be provided as character constants.  
(See Example 11 in (d) Examples of invalid input files.)
- (13) Escape sequences (except for `\0`) in the `\ooo` or `\xhh` format cannot be provided as character constant.  
(See Example 12 in (d) Examples of invalid input files.)
- (14) Wide-character constants (including `L 'x'`) cannot be used.  
(See Example 13 in (d) Examples of invalid input files.)
- (15) The suffixes of integer constants (`u`, `U`, `l`, `L`, `ul`, and `UL`) cannot be used. Match the constant type with the declarator type.  
(See Example 14 in (d) Examples of invalid input files.)
- (16) The suffixes of floating-point constants (`f`, `F`, `l`, and `L`) cannot be used. Match the constant type with the declarator type.  
(See Example 14 in (d) Examples of invalid input files.)

#### 4. COMPILER

- (17) When an aggregate is included in the member of an aggregate (structure, array), the initial value of the subaggregate (the aggregate of the member) cannot be specified by enclosing it with curly brackets ({}). In other words, curly brackets ({} ) cannot be included in the initial value list ({} ).  
Specify all of the required arrays and initial values of the structure. However, when the number of initial values is less than the number of array and structure elements, the rest are initialized as 0.  
(See Examples 15 and 16 in (d) Examples of invalid input files.)
- (18) Enclose a comment by using /\* \*/. A comment can span multiple lines.  
Comments of the // format cannot be used.  
(See Example 17 in (d) Examples of invalid input files.)
- (19) When an error or unsupported syntax is detected in the input file, processing terminates immediately, and subsequent inputs are not interpreted.
- (20) The type conversion when the declaration type does not match the format for specifying the initial values differs depending on the compiler.  
For details, see (f) Initial value type conversion specifications.

**(c) Example of a valid input file**

The following is an example of a valid input file.

```

/* Include file */
#include "defines.h"
/* GLB/VAL external name declaration */
extern int tbl1_g[] ;
extern int tbl20_g[] ;
extern int tbl21_g[] ;
extern int tbl22_g[] ;
extern int tbl23_g[] ;
extern int vall_v[] ;

/* Variable */
int int_var_g = 1 ;

/* Array */
int int_array[10] = {0,1,2,3,4,5,6,7,8,9} ;

/* Structure */
struct {
    int a ;
    int b ;
    int c[10] ;
} struct_var_g = {1,2,10,11,12,13,14,15,16,17,18,19} ;

/* Pointer */
int *tbl1p_g = tbl1_g ;
int *vallp_g = vall_v ;

/* Pointer array */
int *tbl2xp_g[4] = {tbl20_g,tbl21_g,tbl22_g,tbl23_g} ;

/* Initialization of character array */
char str_g[12] = {'I',' ','a','m',' ','s','t','r','i','n','g','\0'} ;

/* Initialization of an aggregate including a subaggregate */
struct Y {
    int a ;
    struct X {
        int a ;
        float b ;
        float c ;
    } x ;
} y_g = {-1,+2,3.0,4e1} ;

float f_g[4][3] = {1,3,5,2,4,6,3,5,7} ;

/* Reference of define value */
int defines[4] = {VAL1, VAL2, GLB1ADDR, GLB2ADDR} ;

```

Contents of defines.h

```

#define VAL1      100
#define VAL2      200
#define GLB1ADDR  0x50020000
#define GLB2ADDR  0x50021000

```

Figure 1-17 defines.h

## 4. COMPILER

### (d) Examples of invalid input files

The following are some examples of invalid input files.

```
extern int tbl1_g[] ;
extern int tbl20_g[] ;
extern int tbl21_g[] ;
extern int tbl22_g[] ;
extern int tbl23_g[] ;
extern int vall_v[] ;

/* Example 1: Preprocessor reference term (out of specification range) */
#define XMAX 10.0e100

/* Example 2: static storage class specifier */
static int int_array[10] = {0,1,2,3,4,5,6,7,8,9} ;

/* Example 3: union */
union {
    long a ;
    float b ;
} struct_var_g = {1} ;

/* Example 4: typedef */
typedef int* tblp ;
tblp tbl1p_g = tbl1_g ;

/* Example 5: Bit field */
struct bit{
    int a:8 ;
    int b:8 ;
    int c:8 ;
    int d:8 ;
}bit_g = {1,2,3,4} ;

/* Example 6: Type modifier */
const int ro_g = 1 ;

/* Example 7: Complex declaration */
int *(tbl2xp_g[])[4] = {tbl20_g,tbl21_g,tbl22_g,tbl23_g} ;

/* Example 8: Omission of array size and character string */
char str_g[] = "I am string" ;

/* Example 9: Expression */
int *vallp_g = vall_v+4 ;
int *tbl1p_g = &tbl1_g[0] ;
int var3_g = XMAX+1 ;
int var4_g = (int)1 ;
```

```
/* Example 10: Signed octal number, hexadecimal number, and character
constant */
int var5_g[3] = {+01, -0x1, -'x'} ;

/* Example 11: Code consisting of two or more bytes */
int a_g = 'abcd' ;
int b_g = 'Kanji' ;

/* Example 12: Escape sequence */
char bell_g = '\007' ;

/* Example 13: Wide-character constant */
wchar_t wc_g = L'x' ;

/* Example 14: Suffix of constant */
unsigned int ui_g = 1U ;
long l_g = 1L ;
unsigned long ul_g = 1UL ;
float f_g = 1.0F ;
double d_g = 1.0L ;

/* Example 15: Initialization in a subaggregate { } */
struct Y {
    int a ;
    struct X {
        int a ;
        float b ;
        float c ;
    } x ;
} y_g = {-1, {+2, 3.0, 4e1}} ;

/* Example 16: Omission of the initial values of a subaggregate */
float f_g[4][3] = {{1},{2},{3},{4}} ;

/* Example 17: Comment format */
// Comments cannot be used in this format.
```



## 4. COMPILER

### (e) Restrictions on preprocessor functions

- (1) Only `#include` and `#define` can be specified.
- (2) Only `#define` can be specified in files included by `#include`.  
`#include` cannot be nested.
- (3) Specify `#include` in the following format.  
Enclose file names by using double quotation marks (“ ”). File names enclosed by angle brackets (<>) are not interpreted as include files.  
When the file name is a relative path, it is interpreted as relative from the directory that contains the input file.

```
#include "file-name"
```

- (4) The `#define` function supports only the following format. Only the replacement of simple names and integer constants is possible.  
Macros for values other than floating-point constants, character constants, and constants cannot be included.  
Octal, decimal, and hexadecimal constants can be provided as integer constants.  
Signed decimal values can also be included.

```
#define name integer-constant
```

- (5) Reference of names defined by `#define` can be included only in the list of initial values. If such references appear in anything other than the initial values list, they are not expanded.
- (6) Values defined by `#define` are valid throughout the entire input file, but cannot be redefined.

## (f) Initial value type conversion specifications

The following table shows variable declaration types, availability of type conversion according to the format for specifying initial values, and differences from the compiler (shc compiler).

Table 1-17 Initial Value Type Conversion Specifications

		Format for specifying initial values											
		Octal number		Decimal number		Hexadecimal number		Floating-point number		Character constant		External name	
		shc	svdatagen	shc	svdatagen	shc	svdatagen	shc	svdatagen	shc	svdatagen	shc	svdatagen
Type of variable	char	char	char	char	char	char	char	char	–	char	char	–	char
	short	short	short	short	short	short	short	short	–	short	short	–	short
	long	long	long	long	long	long	long	long	–	long	long	–	long
	float	float	–	float	float	float	–	float	float	float	–	–	–
	double	double	–	double	double	double	–	double	double	double	–	–	–
	Pointer	–	addr	–	addr	–	addr	–	–	–	–	–	addr

char, short, long, float, double: Generates initial values in accordance with the respective types.

addr: Generates a 4-byte address value.

–: Indicates that no initial value can be generated.

□: Indicates differences between the shc compiler and the svdatagen command.

## CHAPTER 5 PROGRAMMING COMMANDS

### 5.1 Notes on Programming Commands

As S10VE programming commands, the librarian provides `optlnk` and the linker provides `svload`. The `makehce` command is supported as a `make` command.

For details about `optlnk`, see Section 4 Optimizing Linkage Editor Options in the `shc` manual (*SuperH RISC engine C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual*).

## CHAPTER 6 ALLOCATOR

### 6.1 Allocating and Deallocating Split Areas

#### 6.1.1 Necessity for split areas

Storage areas must be allocated for the tasks, subprograms, GLB, and other shared resources to be used by a real-time program before its development.

To speed up processing, the real-time system allows access to various resources through the addresses where they are stored. To achieve this, the storage addresses of resources must be consistent throughout program execution. The allocator divides areas in the computer into user-specified areas. The system keeps track of these areas so that they are retained at the specified addresses. In the system design stage, check how much data is required against the target to which the system is applied, and determine the size and position of GLB. In most real-time systems, the design of the data is more important than the creation of individual programs, and significantly affects the overall performance of the system.

Allocate areas at two stages:

- (1) Define split areas (AREAs) for tasks, subprograms, and GLB.
- (2) Use `svdfs` further to divide AREAs for GLB into secondary partition areas (SAREAs) as necessary.

When split areas and secondary partition areas are defined by the allocator, their names, attributes, positions, sizes, and other information are recorded as area management information. The real-time program can use the names defined in the information to reference or call shared resources such as GLB.

Since areas for shared resources are allocated in a fragmentary manner, they can be allocated separately from those for ordinary resources. This is especially useful when redefining allocated split areas, because the need to redefine other split areas is minimized.

## 6. ALLOCATOR

### 6.1.2 Allocating split areas

Split areas are arranged within a predefined global area (GAREA) in accordance with their use. Table 1-18 shows the relationship between split area use and GAREA selection.

The use of the split area to be defined must be specified by using an option with the `svdfa` command at the time of split area allocation.

Table 1-18 Relationship between Split Area Use and GAREA Selection

Split area use	GAREA selection	svdfa option
Task (program)	\$TASK	-p
Data for CM	\$CM	-cmi, -cmw
Read-only GLB data	\$GLBR	-gr
Read/write GLB data	\$GLBRW	-gi, -gw
Subprogram	\$IRSUB	-s

When a split area is allocated, an area sufficient for the specified size is acquired within the global area. However, it is not reflected in the S10VE memory until the allocated split area is downloaded.

Furthermore, when a split area is allocated, an initial value setup file (backup file) for the real-time resource to be placed within the split area is generated.

With respect to a split area for GLB or CM, without an initial value, however, no backup file will be generated.

The contents of the backup file are initialized to zero (0).

The following commands are supported to allocate split areas:

`svdfa`: Allocates a split area (AREA).

`svdfs`: Allocates a secondary partition area (SAREA).

Figure 1-18 shows a sample layout of split areas.

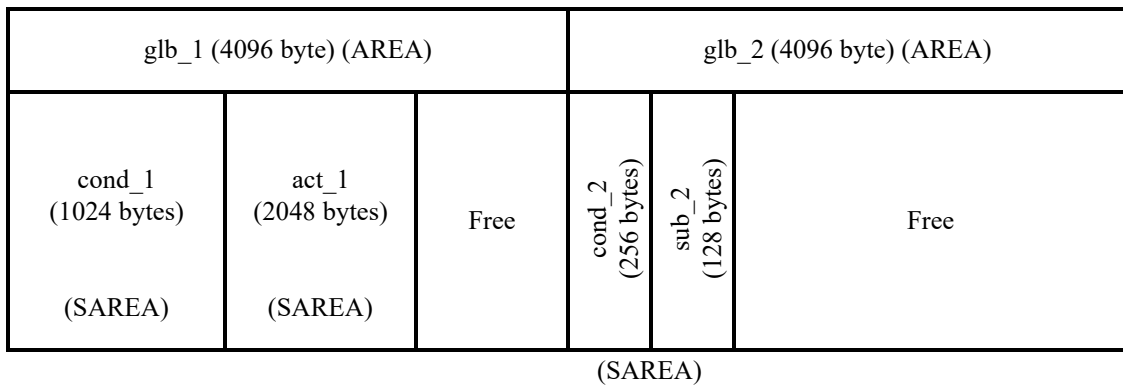


Figure 1-18 Sample Layout of Split Areas

Allocation example for the preceding layout

```
#svdfa glb_1 4096 -gw
#svdfs glb_1 cond_1 1024
#svdfs glb_1 act_1 2048

#svdfa glb_2 4096 -gw
#svdfs glb_2 cond_2 256
#svdfs glb_2 sub_2 128
```

## 6. ALLOCATOR

Allocate the split areas according to the layout in Figure 1-18.

```
$svdfa glb_1 4096 -gi
$svdfa glb_2 4096 -gi
$svdfs glb_1 cond_1 1024
$svdfs glb_1 act_1 2048
$svdfs glb_2 cond_2 256
$svdfs glb_2 sub_2 128
$svmap -g -a -e
** allocator map **                                     2018/02/07 15:19:13

site name = 0001cp

< area >
garea/aname                raddr    size     laddr    kind  bkupfile
:
:
$GLBRW/glb_1                + u 00002000 00001000 50002000 glbi  glb_1.bkf
$GLBRW/glb_2                + u 00003000 00001000 50003000 glbi  glb_2.bkf
$GLBRW/                      00004000 00020000 50004000
:
:

< sarea >
garea/aname/sname          raddr    size     laddr
:
:
$GLBRW/glb_1/cond_1        + u 00000000 00000400 50002000
$GLBRW/glb_1/act_1        + u 00000400 00000800 50002400
$GLBRW/glb_1/              00000c00 00000400 50002c00
$GLBRW/glb_2/cond_2        + u 00000000 00000100 50003000
$GLBRW/glb_2/sub_2        + u 00000100 00000080 50003100
$GLBRW/glb_2/              00000180 00000e80 50003180
:
:
** map output end **
$
```

In the real-time program, the names `cond_1`, `act_1`, `cond_2`, and `sub_2` are assigned to the defined split areas so that these shared resources can be used.

```
$notepad sample.c
```

```
extern int cond_1_g[256];
extern int act_1_g[512];
extern char cond_2_g[256];
extern short sub_2_g[64];
main()
{
  short abc;
  cond_1_g[10] = 0;
  act_1_g[20] = 30;
  cond_2_g[255] = 'A';
  abc = sub_2_g[0];
}
```

When changing the size or attribute of an allocated split area, be careful not to make the new size smaller than the total size of the defined secondary partition areas.

If the new size is smaller than the total size of the secondary partition areas, secondary partition areas cannot be allocated in the specified split areas.



## 6. ALLOCATOR

### 6.1.3 Deallocating split areas

The following commands are supported to deallocate split areas:

`svdla`: Deallocates a split area (AREA).

`svdla`: Deallocates a secondary partition area (SAREA).

For example, `glb_1` and `glb_2`, which were allocated in section 6.1.2, are to be deleted from the GLB as follows. Note that these deletion operations delete the desired split area along with any secondary partition areas inside it. For example, if `glb_2` is deleted, the secondary partition areas `cond_2` and `sub_2` defined within it are also deleted at the same time.

```
$svdla cond_1
$svdla act_1
$svdla glb_1
$svdla glb_2
$svmap -g
** allocator map **                               2018/02/07 15:19:13

site name = 0001cp
:
< global,CM,DCM,irglb >
:

** map output end **
$
```

### 6.1.4 Assigning names to GLB and VAL

The names of GLB and VAL must be unique in the system.

- The maximum number of characters in a name (the number of bytes) is limited to 14.
- Each name must begin with a letter or an underscore (`_`).
- The second and subsequent characters can be a combination of letters, numbers, and underscores (`_`).

Note, however, that real-time programs using names under these rules are also restricted by the following naming rules in C.

Restrictions in C:

- The maximum number of characters in a name (the number of bytes) is limited to 14.
- Each name must begin with a letter.
- The second and subsequent characters are a combination of letters, numbers, and underscores (`_`).
- Each name is suffixed with one of the following character strings. Characters in the suffix are not counted as part of the name.  
For GLB: `_g`  
For VAL: `_v`
- When declaring GLB or VAL, be sure to use an external variable with `extern` specified.

6.1.5 Allocating split areas for the CM

The CPMS provides CM as a means to share main memory between PUs in the same unit. This method allocates an area in the CM space in the logical space of CPMS tasks, uses the allocator to name the area, and then shares the area among tasks and subroutines in the same unit. The aforementioned area is divided into areas by using *svdfa*, and each divided area is further divided into areas by using *svdfs*.

The CM space is used as communication memory between the HP and the CP of the CPU. The following figure illustrates the CM space in the logical space of the CPMS.

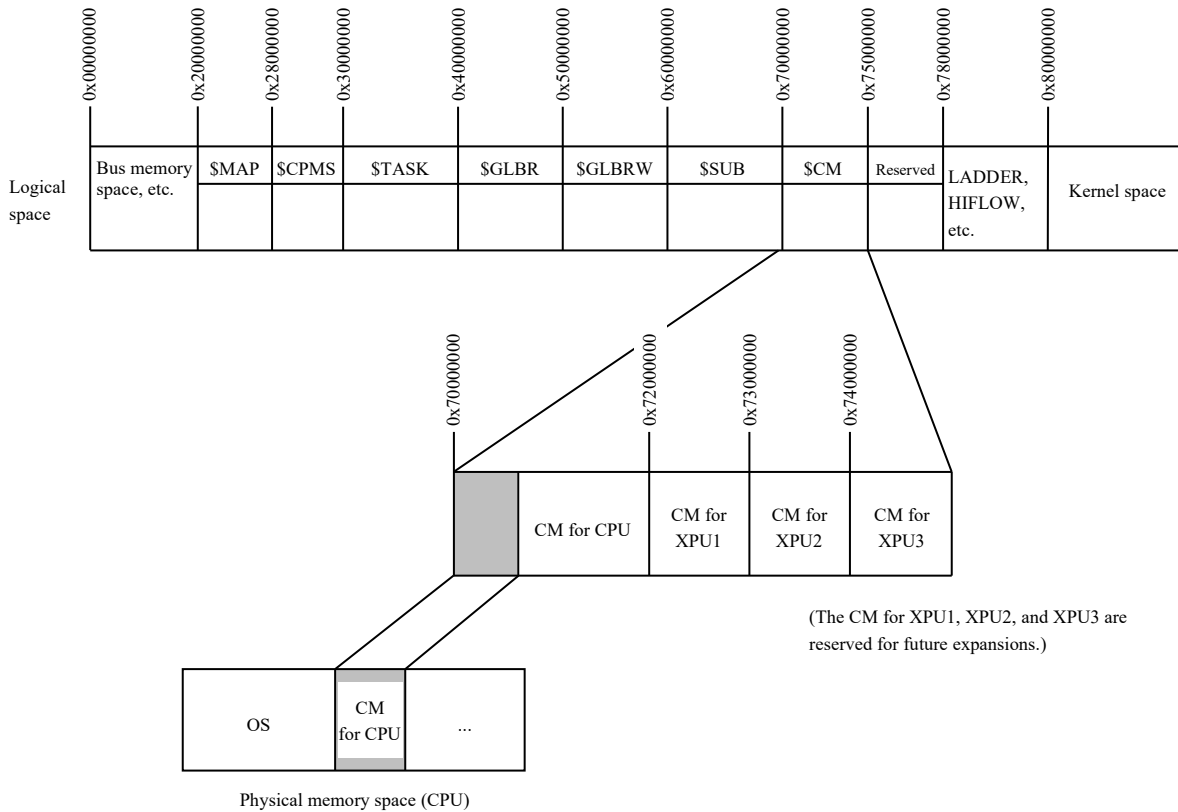


Figure 1-19 Correspondence between CM Spaces in the Logical Space of the CPMS and CM Spaces in the S10VE Main Memory

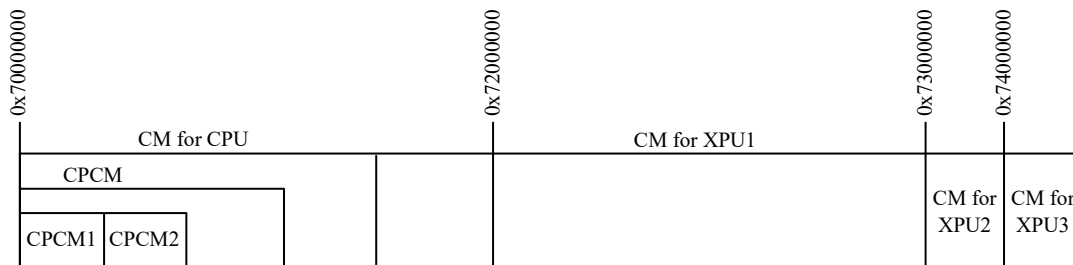
Using CM

- Allocating split areas in CM areas

In the site for each PU (HP and CP) of the RPDP, the names of any split areas and secondary partition areas (defined in the CM space corresponding to a site) are valid only in the relevant site. Therefore, when sharing these split areas and secondary partition areas between the site for the HP and the site for the CP, split areas and secondary partition areas must be identically defined in both sites.

To allocate split areas defined in the CM area to the same addresses in both sites, specify the `-f` option when split areas are defined by the `svdfa` command. Doing so equalizes the relative address from the beginning of the global area `$CM` between both sites. Failure to specify the `-f` option when defining split areas in the CM area results in an error. The backup file for the split areas defined in the CM area is created in the site for CP.

The following is an example of sharing CM between the CP and the HP.



CP-side area definition operation

```
svdfa CPCM 4096 -cmi -f 0x80000
svdfs CPCM CPCM1 128
svdfs CPCM CPCM2 128
```

HP-side area definition operation

```
svdfa CPCM 4096 -cmi -f 0x80000
svdfs CPCM CPCM1 128
svdfs CPCM CPCM2 128
```

As shown in the preceding operation example, define areas so that identical names and identical addresses are obtained, including the areas on the CP side and HP side.

- Loading data to CM areas

Split areas and secondary partition areas in CM areas must be defined for allocation at the same addresses in the site for the HP and the site for the CP. Of these areas, only those that allow the loading of initial value data are allocated in the CP site.

## 6.2 Value (VAL) Registration and Deletion

The allocator registers or deletes a constant called a value (VAL), which is common to all programs. The `svdfv` and `svdlv` commands are used for VAL registration and deletion, respectively.

## CHAPTER 7 LOADER

### 7.1 Linking and Loading

Object modules (.obj files) created by the `shc` command are linked by using the `svload` command to integrate them into a single program, which is then loaded by using `GLB`, `IRSUB`, and other items of allocator management information. The resulting load module is then written to a backup file (see Figure 1-20).

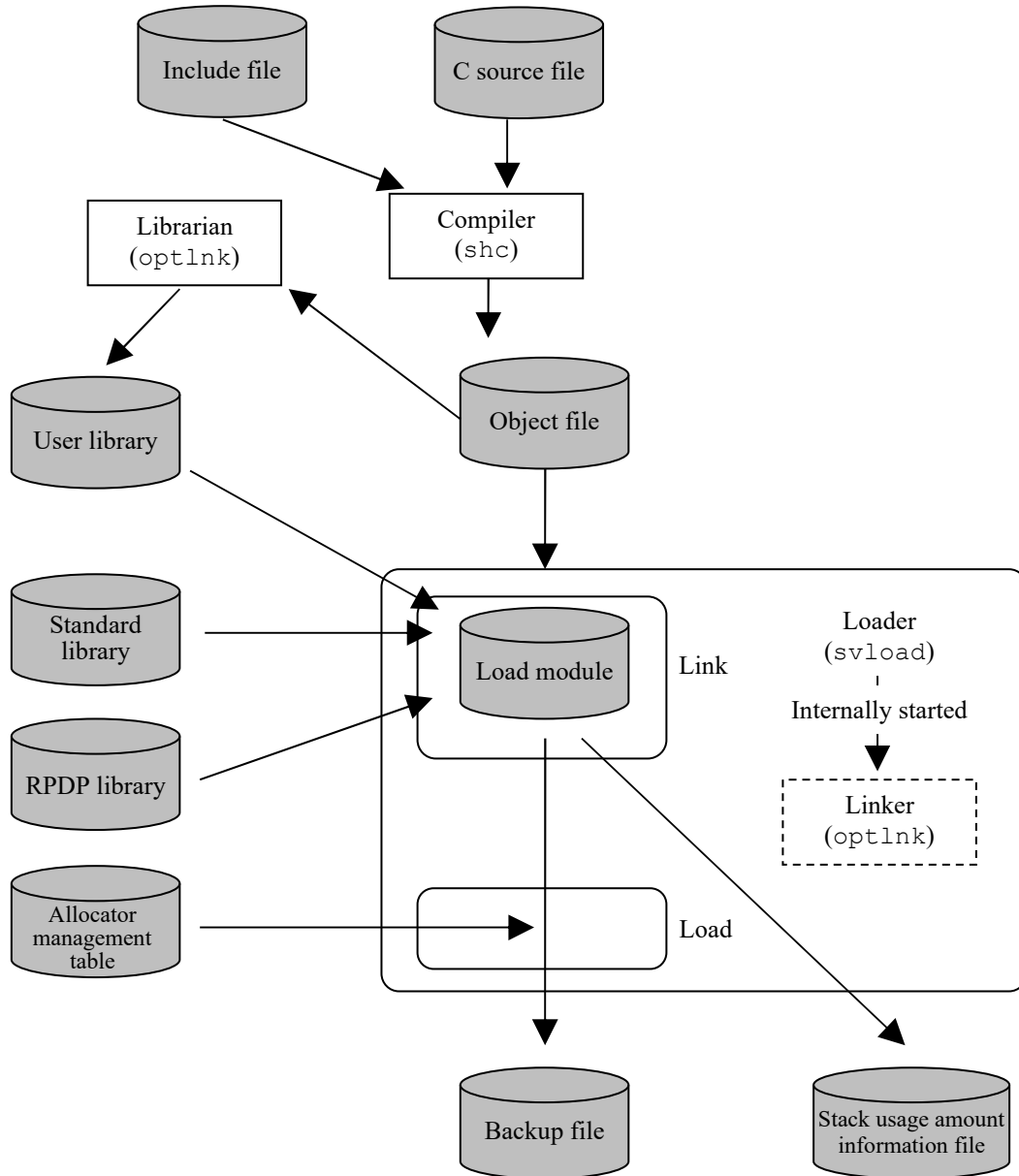


Figure 1-20 Creating a Load Module and Backup File

## 7.2 Loader Operating Environment

With respect to the program to be entered into the loader, ensure that the load module resulting from linkage in the loader satisfies the conditions described in Table 1-19.

Table 1-19 Conditions for Load Modules

Option	Load module		
	TEXT	DATA	BSS
Program registration	> 0	–	–
Subprogram/built-in subroutine registration	> 0	–	– (*)
Data registration	–	> 0	–

TEXT: Executable portion DATA: Data with initial value BSS: Area without initial value

–: Can be processed when size = 0 or > 0

> 0: Error except when size > 0

(\*) The BSS parts of subprograms are write-protected. Ensure that subprograms do not have BSS parts.

Figure 1-21 shows the structure of a load module that is generated by the loader.

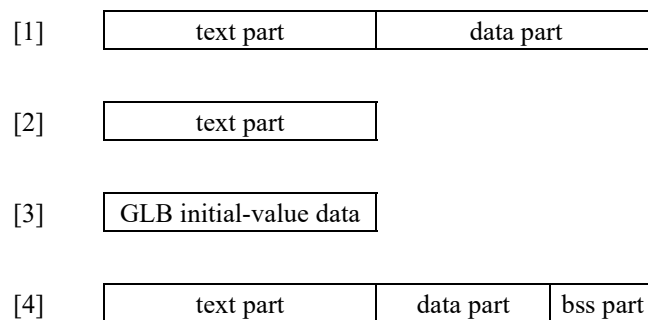


Figure 1-21 Load Module Structure

Explanation of Figure 1-21:

[1] is a load module of a program or subprogram that has a text part and a data part.

This load module can be loaded as a program or subprogram.

[2] is a load module of a program or subprogram that has only a text part.

It can be loaded in the same way as [1].

[3] is a load module of a GLB initial-value settings program.

It can be loaded as data.

[4] is a load module of a program that has a text part, a data part, and a bss part.

It can be loaded as a program.

Ensure that subprograms do not have bss parts.

(1) Loader processing

The following explains the loading process of the loader with reference to load module structures [3] and [4] in Figure 1-21.

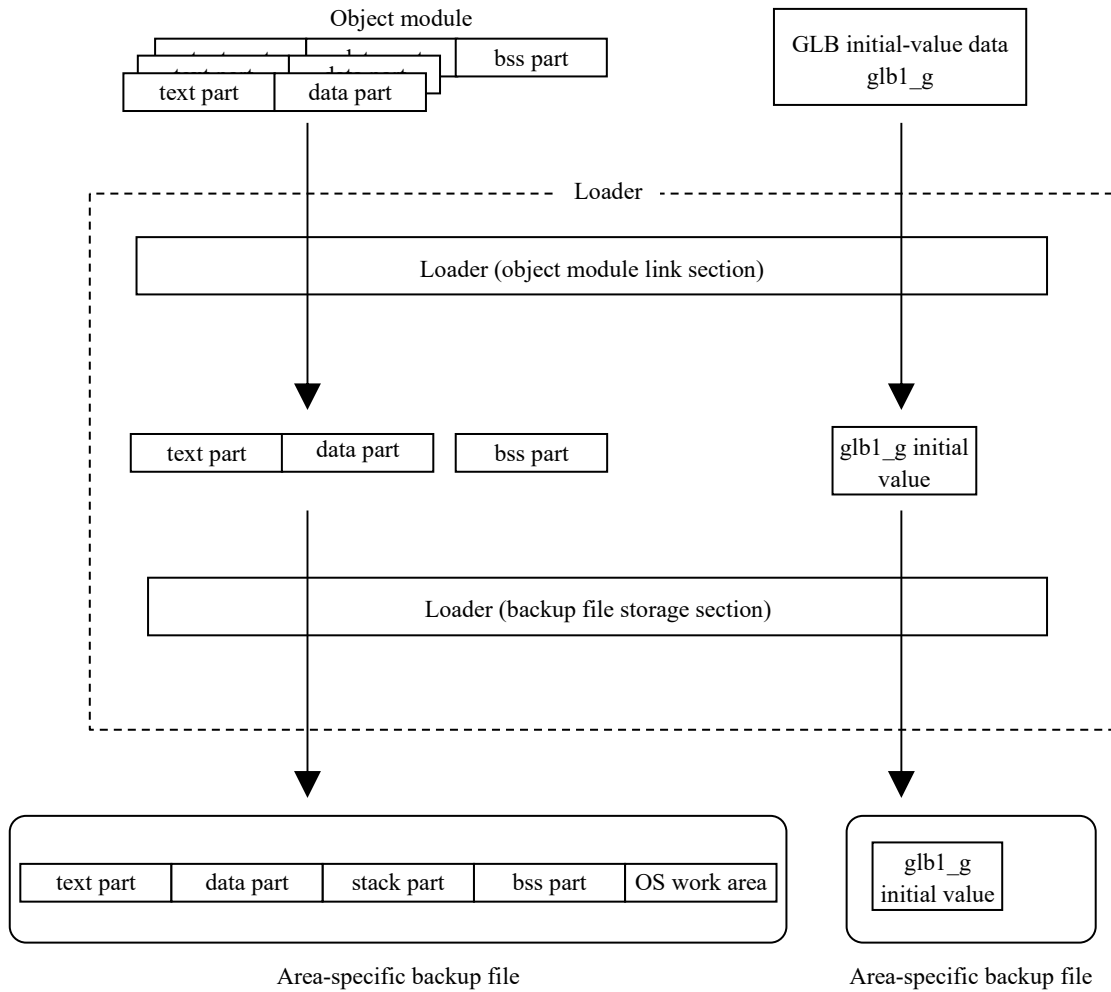


Figure 1-22 Loading Processing

Explanation of Figure 1-22:

- [1] The global initial-value data section is loaded into a location corresponding to a secondary partition area that is registered by *svdfs* in the allocator-managed table.
- [2] The program is loaded, as a load module, into an area that is specified by the loader command. The loader is stored in a backup file with a stack area specified by the load module and the OS work area attached.

## (2) Uniqueness of names

Between the system and user, the same name must not appear as a program name, subprogram name, built-in subroutine name, global name, or value name.

## (3) External reference checks for the system and users

User information cannot be referenced from the system.

System subprograms can only be referenced by users. Table 1-20 shows the possible reference combinations.

Table 1-20 External Reference Combinations

Referenced \ Referencing		Subprogram		Global		Value	
		S	U	S	U	S	U
Program	S	R	–	R	–	R	–
	U	R	R	–	R	–	R
Subprogram	S	R	–	R	–	R	–
	U	R	R	–	R	–	R
Global	S	(*1)	–	(*2)	–	(*3)	–
	U	R	(*1)	–	(*2)	–	(*3)

R: Can be referenced S: System –: Cannot be referenced U: User

(\*1) An IRSUB number is embedded in the global area.

The subprogram must therefore be a built IRSUB.

(\*2) The referenced global address is embedded in the global area.

(\*3) The VAL value is embedded in the global area.



### 7.3 Library Search Paths

The library search path of the loader (the library search sequence specified by the `-l` option) follows with the sequence in which the system searches for the input file of the optimization linkage editor in the shc compiler package.

The input file search sequence for the optimization linkage editor is as follows:

- (1) Current directory
- (2) Directory specified by `HLNK_DIR` in the RPDP operating environment setup file

Multiple paths can be set for `HLNK_DIR` in the RPDP operating environment setup file.

To specify multiple paths, separate them with semicolons.

### 7.4 Notes on Linking and Loading

When linking and loading real-time programs, make sure that the `GLB`, `CM`, and `VAL` used in them have been allocated. When using `IRSUBs`, make sure that they have been built.

## CHAPTER 8 BUILDER

### 8.1 Registering and Deleting Tasks

#### 8.1.1 About tasks

A load module that has been loaded by the loader (`svload`) can be used to create a task via the `svctask` command of the builder to prepare for operations. `svctask` creates task information that specifies a task name, task number, task execution level, and other parameters. `svctask` then combines this task information with the task information managed by the operating system.

#### 8.1.2 Registering a task

The following is an example of registering a program as a task.

```

$ svdfa area1 0x3000 -p
$ shc sample.c
$ svload +P -w 1024 1024 -a area1 -o sample sample.obj
$ svmap -p -t
** allocator map **                               2018/02/07 15:20:38

site name = 0001cp

< task-program >
tn  tname                tnox rmtn lvl  sp    pname                st mtn  texttop
lastaddr tsize  dsize  ssize (part ) bsize  extra  oswork
:
                                sample                + u ls 0001 30034000
30037000 0000f4 000022 000400(000400) 000000 000000 001000
:
** map output end **
$ svctask sample sample_1 10 -l 25
$ svmap sample_1 -t
** allocator map **                               2018/02/07 15:21:38

site name = 0001cp

< task-program >
tn  tname                tnox rmtn lvl  sp    pname                st mtn  texttop
lastaddr tsize  dsize  ssize (part ) bsize  extra  oswork
:
10 sample_1                + u 000a 0001 19 30036000 sample                . u cs 0001 30034000
30037000 0000f4 000022 000400(000400) 000000 000000 001000
:
** map output end **
$

```

The string `sample` specified in `svctask` is the name of the load module to be used as a resource for the task. When the task is started, the specified load module is run as the main program of the task.

In this example, the priority of the task is 25, and the task number is 10.

After a task is created, use the task number or task name to identify the task.

## 8. BUILDER

### 8.1.3 Deleting a task

To delete a registered task, use `svdtask`. Specify the name of the task to be deleted for `svdtask`. The following is an example of deleting a registered task.

```
$ svdtask sample_1
$ svmap sample_1 -t
** allocator map **                               2018/02/07 15:21:09

site name = 0001cp

< task-program >

** map output end **
svmap : Specified name is undefined ( sample_1 )
$ svdload sample +P
$ svdla areal
$
```

## 8.2 Registering and Deleting a Resident Subprogram

### 8.2.1 About indirect link subprograms (IRSUBs)

An indirect link subprogram does not require task re-registration, even if it is replaced after task registration.

An indirect link subprogram is created in the same way as a subroutine of a program. Unlike a subroutine, an indirect link subprogram is shared by multiple tasks, so it must be re-entrant. For this reason, do not declare static variables in indirect link subprograms.

When an indirect link subprogram is registered again, only the address corresponding to the registered number changes. Tasks need not be registered again, even if they were registered before the re-registration of indirect link subprograms.

### 8.2.2 Registering an indirect link subprogram (IRSUB)

After using the loader (`svload`) to load an executable module, register the executable module by using the builder (`svbuild`).

The following is an example of registering a program as an indirect link subprogram (IRSUB).

Create an indirect link subprogram. Use Notepad or another text editor.

```
$notepad sub_a.c
```

```
sub_a()
{
    return;
}
```

Compile the indirect link subprogram.

```
$shc sub_a.c
```

Register the indirect link subprogram in the split area named `areal` using the registration number 10.

```
$ svdfa areal 4096 -s
$ svload +I -a areal -o sub_a sub_a.obj -w 0
$ svbuild sub_a -ir -e 10
$ svmap sub_a -s -ir
```

```
** allocator map **
```

```
2018/02/07 15:21:09
```

```
site name = 0001cp
```

```
< IRSUB >
```

irno	entname	st	laddr	subname	offset	texttop	bsslast	tsi
ze	dsize	bsize	extra	ssize (part)				
10	sub_a		+ u	ib 60010000	sub_a	000000	+ u 60010000	60010004 000
004	000000	000000	000000	000000	000000	(000000)		

```
** map output end **
```

## 8. BUILDER

### 8.2.3 Deleting an indirect link subprogram (IRSUB)

To delete a registered indirect link subprogram (IRSUB), use `svdbuild`. Specify the name of the subprogram to be deleted and the `-ir` option for `svdbuild`.

The following is an example of deleting a registered indirect link subprogram.

```
$ svdbuild sub_a -ir
$ svdload sub_a +I
$ svmap -s -ir
** allocator map **                                     2018/02/07 15:21:09

site name = 0001cp

< IRSUB > [max_entry= 8191, use_entry= 256]

** map output end **
$
```

## 8.3 Registering and Deleting a Built-in Subroutine

### 8.3.1 About built-in routines

This section describes built-in subroutines. Built-in subroutines can be incorporated into the system to run user-specified processing in place of the standard error handling of the OS whenever a hardware-detected exception or software-detected event occurs.

Built-in subroutines are loaded in by the loader (`svload`) and incorporated by the builder (`svbuild`) as part of the event-handling programs supported by the operating system.

In this system, some entry points for incorporating built-in subroutines are provided. Different points are associated with different events. When incorporating built-in subroutines, select entry points according to the events by which their processing is initiated.

Up to four built-in subroutines can be registered for incorporation at each entry point. When an event occurs, the registered built-in subroutines are called and run sequentially in ascending order of their entry numbers. In this way, entry numbers specify the order in which registered built-in subroutines are run. However, entry numbers 1 and 2 are reserved for the OS and NXACP. Use entry numbers 3 and 4 for user programs.

To use the structure defined by the CPMS within a built-in subroutine, specify a CPMS include file at the time of compiling. For details, see [Setting up the interface used with the CPMS in 4.1 Details of C Compiler Options](#).

## 8. BUILDER

### 8.3.2 Registering a built-in subroutine

The following is an example of registering a built-in subroutine. In this example, the program named `uabs_usr` is a subroutine that is run when a task is aborted. Register the subroutine at the entry point that has the point name `ABS`. To link and load the built-in subroutine, use the loader (`svload`) with the `+U` option specified.

```
$ shc uabs_usr.c

Register a built-in subroutine.

$ svdfa area1 4096 -s
$ svload +U -a area1 -o uabs_usr uabs_usr.obj -w 512
$ svbuild uabs_usr ABS 3 -ul
$ svmap -s -ul
** allocator map **                                     2018/02/07 15:21:09

site name = 0001cp

< ULSUB >
pnt typ ent subname          texttop  bsslast  tsize  dsize  bsize  extra
ssize (part )
abs os 1 . ulsubabs          . s 60000000 60000499 0003cc 0000c9 000000 000000
000080(000080)
abs user 1 + uabs_usr        + u 60100000 60100004 000004 000000 000000 000000
000200(000200)

** map output end **
$
```

### 8.3.3 Deleting a built-in subroutine

To delete a registered built-in subroutine, use `svdbuild`. Specify the name of the subroutine to be deleted, the entry point name, and the `-ul` option for `svdbuild`. The following is an example of deleting a registered built-in subroutine.

```
$ svdbuild uabs_usr ABS -ul
$ svdload uabs_usr +U
$ svmap -s -ul
*** allocator map **                                     2018/02/07 15:21:09

site name = 0001cp

< ULSUB >
pnt typ ent subname          texttop  bsslast  tsize  dsize  bsize  extra
ssize (part )
abs os 1 . ulsubabs          . s 60000000 60000499 0003cc 0000c9 000000 000000
000080(000080)

** map output end **
$
```

## CHAPTER 9 MAP

### 9.1 Purpose of Displaying Allocator Management Table Information

Allocator management table information is displayed to help users develop real-time programs smoothly.

- Information on the storage areas for tasks, subprograms, GLB, and other shared resources is displayed to support the creation of individual programs and the development of system design.
- Allocator management table information in the system that has been loaded in the S10VE is displayed to support debugging.



## 9.2 **svmap Command Options and Displayed Information**

The display format for each option of the `svmap` command is shown in APPENDIX F MAP DISPLAY FORMAT. The underlined portions within the display formats are the data displayed by the `svmap` command.

### 9.2.1 Map information that is output

The following map information is output:

- (1) Map information about resources managed by the development machine
- (2) Map information about resources downloaded to the S10VE

### 9.2.2 Description of output map information

The following information is output as map information:

- (1) Header and footer
- (2) Global area information
- (3) Split area information
- (4) Secondary partition area information
- (5) Program information
- (6) Subprogram information
- (7) Task information
- (8) Global information
- (9) VAL information
- (10) IRSUB entry information
- (11) IRGLB entry information
- (12) ULSUB entry information
- (13) Information about how much physical memory is available

### 9.2.3 Map information output format

Map information can be output in the following formats:

- (1) Hierarchical map output
- (2) Address-order list output
- (3) Name-order list output
- (4) Numerical-order list output
- (5) Specified name output

The hierarchical map output format is used to hierarchically output map information about resources arranged in a logical space for individual global or split areas.

The listing output formats are used to output specified information in address order, name order, or numerical order.

The name of a resource can also be specified to output information about that name.

Table 1-21 shows the combinations of output information and selectable output formats.

Table 1-21 Combinations of Output Information and Selectable Output Formats

Output format Output content	Hierarchical output	Address-order output	List-order output	Numerical- order output	Specified name output
Global area information	Y	Y	–	–	Y
Split area information	Y	Y	Y	–	Y
Secondary partition area information	Y	Y	Y	–	Y
Program information	–	–	Y	–	Y
Subprogram information	–	–	Y	Y	Y
Task information	–	–	Y	Y	Y
Global information	–	–	Y	Y	Y
VAL information	–	–	Y	–	Y
IRSUB entry information	–	–	Y	Y	Y
IRGLB entry information	–	–	Y	Y	Y
ULSUB entry information	–	–	Y	Y	Y

Y: Can be specified    –: Cannot be specified

### 9.3 Logical Address Specification and Information Displayed by the svadm Command

The `svadm` command displays a name and other information for the specified logical address. An address can be specified by using a command or by using an interactive interface. The information that is displayed is useful when debugging.

#### (1) Specifying an address by using a command

`svadm logical address`

When a logical address is specified in the parameter, the name and other information are displayed in the following format, where the uppercase Xs represent the data displayed by the `svadm` command.

```
name = XXXXXXXX type = XXXXXXXXXXXX raddr = XXXXXXXX
```

#### (2) Specifying an address by using the interactive interface

When a logical address is not specified in the parameter, the user is prompted to enter the logical address. The name and other information are displayed in the same format as when using a command.

Specifying an address by using the interactive interface

```
#svadm
```

```
++ address information display start --> site(XXXXX) ++
```

```
addr : addr
```

```
Displayed information
```

```
addr : q
```

```
++ address information display end ++
```

```
#
```

## CHAPTER 10 STARTUP AND PU CONTROL

### 10.1 Overview

To enable S10VE startup by using the RPDP, use the `svrpl` command to download the initial data file (backup file) stored in the S10VE main memory (SDRAM) on the development machine to the S10VE main memory.

As shown here, the backup file created by the OS and the allocator is downloaded to the main memory of the specified site (PU) of the S10VE to start the specified site (PU).

It is not possible, however, to start an S10VE by using the `svrpl` command if the OS has never been downloaded via the CPMS download of the BASE SYSTEM/S10VE to that S10VE. Load the OS via the CPMS download of the BASE SYSTEM/S10VE first.

PU control of the S10VE controls the status of the specified site (PU) after startup.

When the CPU name (the same name as for the CP site name) is specified as a site name, the CPU is processed in the CP and HP sites. If the HP site name is specified as a site name, an error occurs.

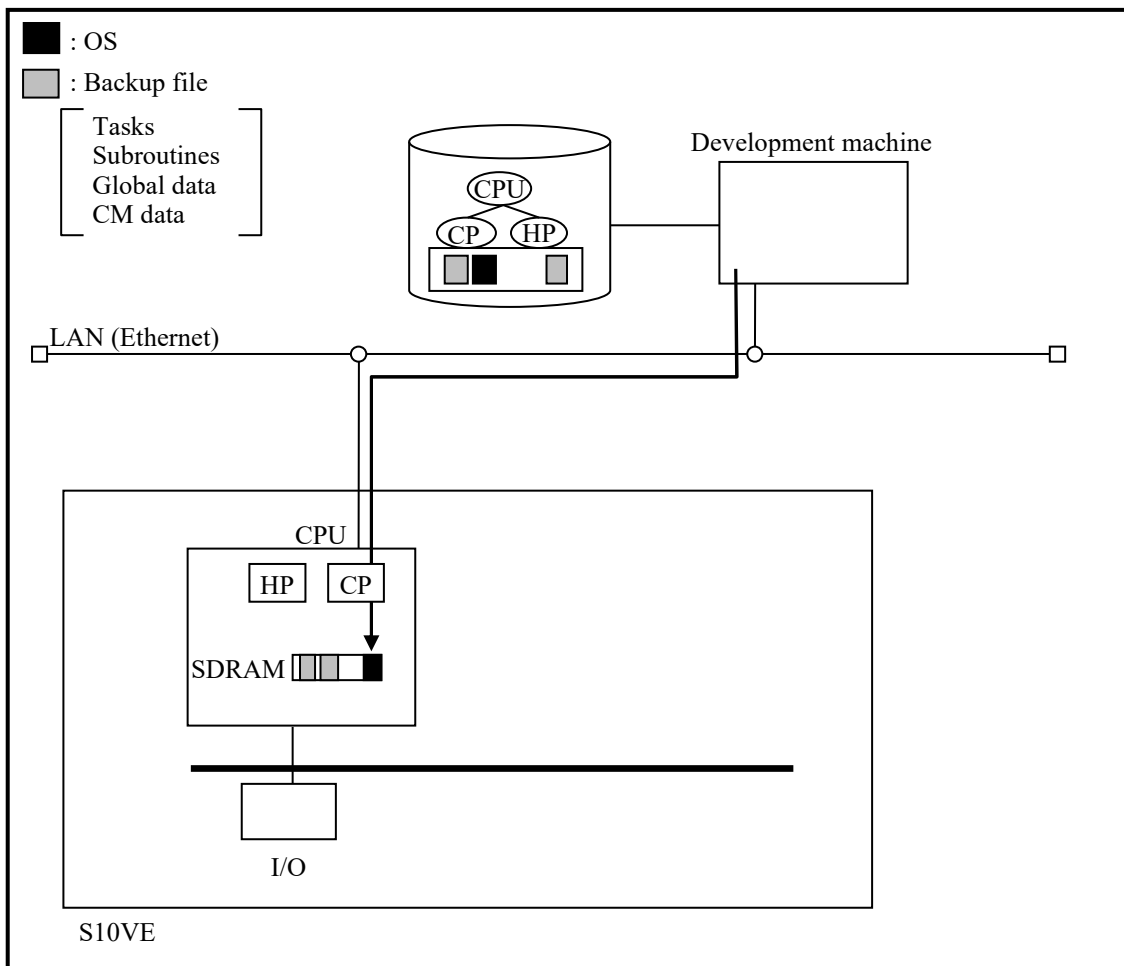


Figure 1-23 S10VE Startup from a Development Machine

**10.2 Basic Concept of Startup and PU Control**

The following is a basic overview of startup and PU control processing.

Overall control of the S10VE is shown in the following figure. There are two cores in the CPU (CP and HP), and these cores control sites independently. However, because startup and PU control are performed by the CPU, the CP site is specified to perform processing.

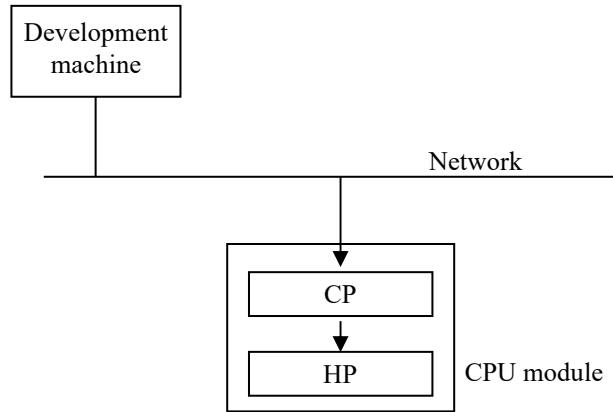
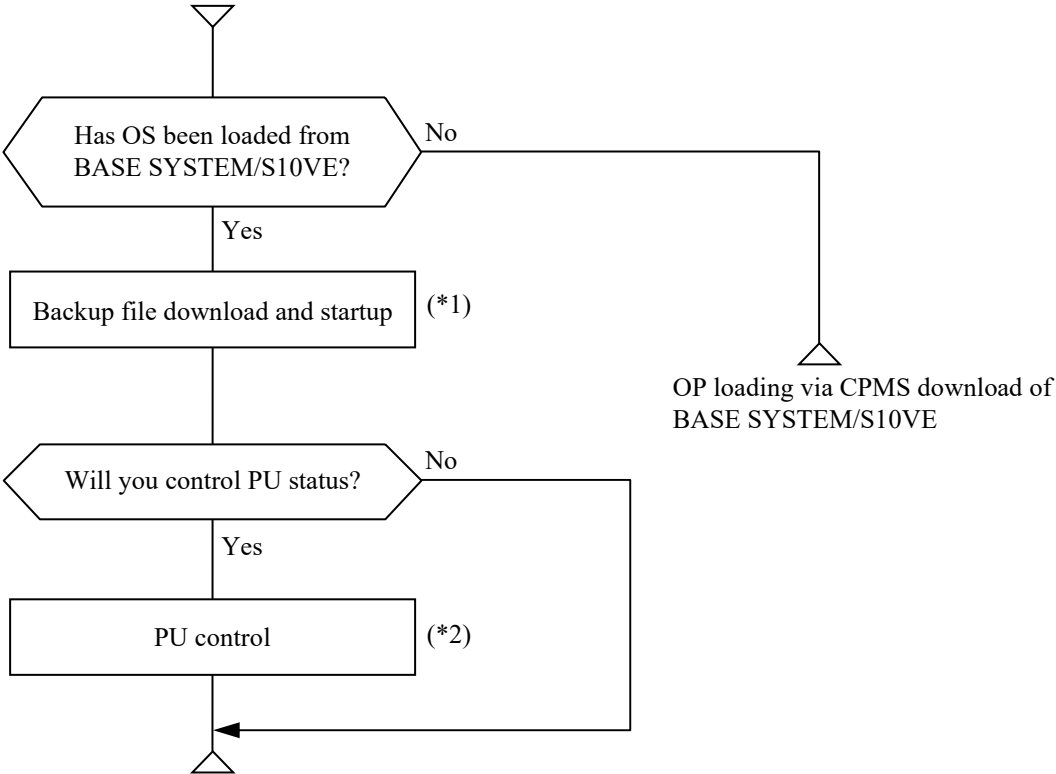


Figure 1-24 Concept of Overall Control of the S10VE

**10.3 Startup and PU Control Procedure**

The `svrpl` command and the `svcpuctl` command are used for startup and PU control of the S10VE. The following describes the startup and PU control procedure.



(\*1) Startup when the `svrpl` command is used  
(\*2) PU control when the `svcpuctl` command is used

**10.4 Startup and Stop Types**

Table 1-22 lists the eight startup and stop types of the S10VE.

Table 1-22 Startup and Stop Types

No.	Type	Start/stop category
1	IPL start	Start
2	Online restart	
3	Reset start	
4	Power-recovery start	
5	ROM start	
6	Power-off stop	Stop
7	STOP request stop	
8	ERROR STOP	

The details of 1 through 8 are described in the following (1) to (8).

The following describes the startup types of the S10VE.

(1) IPL start (including GLB-save IPL start)

The `svrpl` command is used for an IPL start.

Download the OS and the backup files (tasks, subroutines (IRSUBs and built-in subroutines), and GLB and CM data), and then start the S10VE. The period from the end of the download until the activation of the initial start task is called an IPL start.

There are four download types with respect to the OS and backup files (see the following). LADDER and HI-FLOW programs are not downloaded by the `svrpl` command.

- Downloads of the OS, tasks, subroutines (IRSUBs and built-in subroutines), and GLB and CM data

(2) Online restart (power-recovery fast restart, reset fast restart)

Online restart starts when either of the following events occurs:

- When power is recovered after a power failure stop (power-recovery fast restart)
- When a CPU RUN request is generated after stopping upon a CPU STOP request (reset fast restart)

At this time, S10VE processing restarts from the processing at the time when the power failure or the CPU STOP request occurred. Online restart processing covers the period after a CPU RUN request is generated until activation of the initial start task. (At this time, downloading to the S10VE from the development machine is not performed.) I/O units are initialized by the time the initial start task is activated, changing all tasks to the DORMANT state.

(3) Reset start

Reset start begins when a CPU RUN request is generated after an error stop due to a CPU shutdown. The OS data with initial values is returned to the post-IPL state, and processing begins from post-IPL processing. Reset start processing covers the period after a CPU RUN request is generated until activation of the initial start task. (At this time, downloading to the S10VE from the development machine is not performed.)

**(4) Power-recovery start**

A power-recovery start commences when a power-on reset occurs after an error stop due to a CPU shutdown. OS data with initial values is returned to the post-IPL state, and processing begins from post-IPL processing. Power-recovery start processing covers the period from power recovery until activation of the initial start task. (At this time, downloading to the S10VE from the development machine is not performed.)

**(5) ROM start**

The OS, programs, and data are loaded from the ROM, and then the S10VE is started. LADDER and HI-FLOW programs are loaded in addition to tasks, subroutines (IRSUBs and built-in subroutines), and GLB and CM data.

The following describes the stop types of the S10VE.

**(6) Power-off stop**

The S10VE stops due to the power turning off (including power failures). Active tasks stop without being aborted. The CPU stops without closing open I/O units. During a power-off stop, when the S10VE is turned on, a ROM start is performed if data is saved in the ROM.

**(7) STOP request stop**

The S10VE stops upon receiving a CPU STOP request. Active tasks stop without being aborted. The CPU stops without closing open I/O units.

The CPU and I/O units remain on.

An online restart (reset fast restart) is performed when a CPU RUN request is received.

After a CPU STOP request stop, if power is restored, a ROM start is performed.

**(8) ERROR STOP**

The S10VE stops when a fatal hardware or software error occurs. The CPU and I/O units remain on.

To restart the S10VE, turn off the power and then turn it on again (power-recovery start), issue a CPU STOP request and then issue a CPU RUN request for a reset start, or perform an IPL start (OS download). When memory data is cleared at power-recovery start, a ROM start is performed.



## 10. STARTUP AND PU CONTROL

### CPU STOP request

The following events occur with a CPU STOP request:

- Setting the CPU RUN/STOP switch to STOP
- Receiving a STOP interrupt upon a CPU STOP request from the `svrpl` or `svcpuctl` command

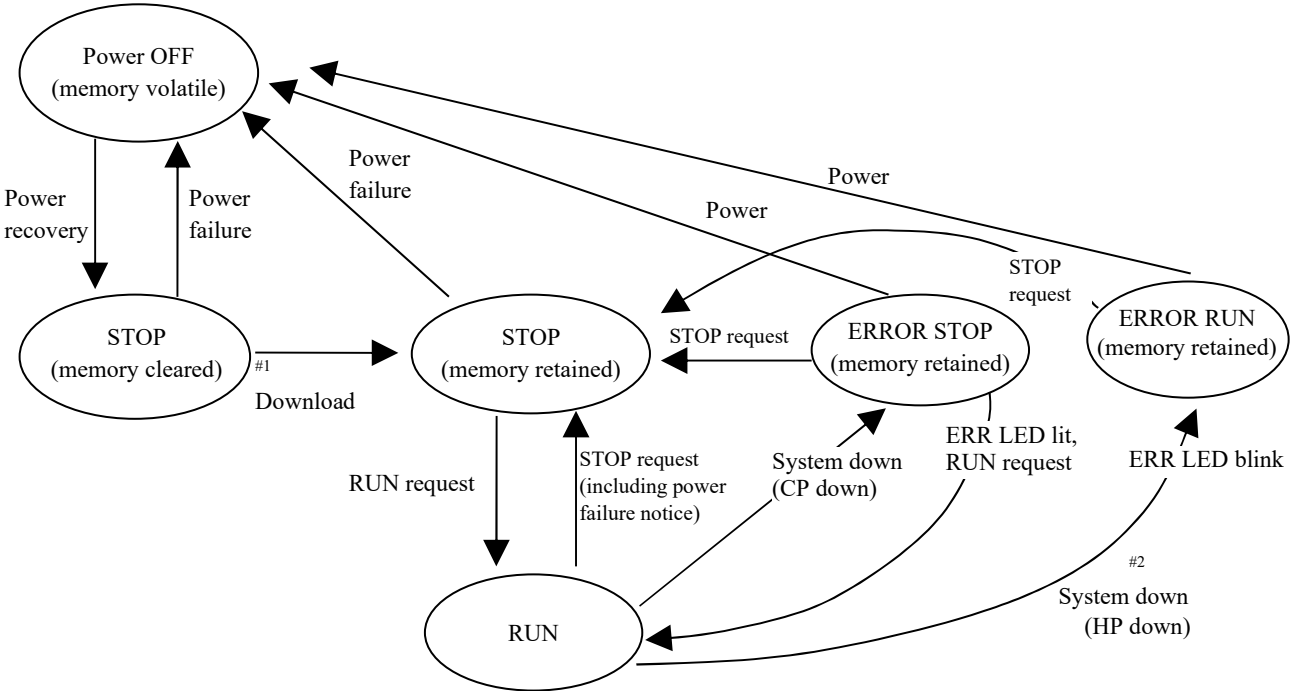
### CPU RUN request

The following events occur with a CPU RUN request:

- Setting the CPU RUN/STOP switch to RUN
- Transmitting a CPU RUN request after a CPU RUN request from the `svcpuctl` command or a download by the `svrpl` command

10.5 PU State Transitions

Figure 1-25 shows the PU state transitions.



(\*1) When the memory is backed up in the ROM, data is downloaded from the ROM to the memory. If the memory is not backed up in the ROM, data is downloaded from the development machine to the memory over the network.

(\*2) The CP continues operating even during an HP stop. In the ERROR RUN state, only the CPMS of the HP stops. In this case, after both processors transition to the CPU STOP state, a CPU RUN request can be received.

Figure 1-25 PU (OS Startup/Stop) State Transitions

## 10. STARTUP AND PU CONTROL

### 10.5.1 Startup procedure

As described in section 10.3, startup is performed by running the `svrpl` command from the development machine.

For the procedure to be performed on the development machine, see CHAPTER 7 STARTUP AND PU CONTROL in PART 2 COMMAND REFERENCE. The following describes the main functions and uses.

#### Functions

The main functions of the `svrpl` command are as follows:

- The command starts the specified site or the specified unit (all sites in the S10VE unit).
- The command downloads the OS and backup files.

You can specify an option for the `svrpl` command to separate the files to be downloaded. Table 1-23 lists the files that are downloaded according to the specified options.

Table 1-23 Download Options

File \ Option	-all	No option specified
OS	Y	Y
• Task • Subroutine • GLB	Y	Y
CM	Y	Y

Y: Downloaded    -: Not downloaded

- The command specifies whether to set the CPU time at startup.
- The command specifies whether to start the PU after the OS and backup files have been downloaded.
- The command stops the PU without receiving a PU startup or stop confirmation response (as to whether the PU can be stopped) during execution of the `svrpl` command.

The following are some startup examples in which the various command functions are applied.

- Downloading the OS and all backup files to the CPU, setting the time, and then starting the CPU

Mounted PU

CPU CP (site name = 0001cp)  
CPU HP (site name = 0001hp)

```
>svrpl -u 0001cp
**** svrpl start (site = 0001cp) ****
All PU status CPU(0001cp)=RUN
Do you stop CPU(0001cp) ? (yes/no)= yes
Remote loading start(site = 0001cp)
address : 0X0400d100-0X0400d8ff
.
address : 0X0400d000-0X0400d0ff
.
address : 0X041f0000-0X0435ffff
....
address : 0X0437f000-0X0437fe57
.
address : 0X04d80000-0X0557ffff,0X07500000-0X0927ffff
.....
address : 0X0400c100-0X0400c8ff
.
address : 0X04080000-0X041effff
....
address : 0X0437e000-0X0437ee57
.
address : 0X06780000-0X074fffff
.....
start to modify allocator management tables

finished to modify allocator management tables
Remote loading end
Please wait 84 seconds for ROM-SAVE
Reset start CPU(0001cp)
**** svrpl end ****

>
```

- Downloading the OS and all backup files to the CPU, setting the time, and then starting the CPU without receiving a PU stop confirmation response (No message appears when the `-s` option is specified and no confirmation response is received.)

```
>svrpl -u 0001cp -s
```

## 10. STARTUP AND PU CONTROL

### 10.5.2 PU control procedure

PU control is performed by running the `svcpuctl` command from the development machine. For the procedure to be performed on the development machine, see CHAPTER 7 STARTUP AND PU CONTROL in PART 2 COMMAND REFERENCE.

The following describes the main functions and uses:

#### Functions

The main functions of the `svcpuctl` command are as follows:

- The command controls the PU state (running or stopped).
- The command sets the CPU time when a run request is received.
- The `svcpuctl` command runs without acknowledging whether the PU is to be started or stopped (whether it is all right to begin execution).
- The command displays the site status.

The following are some startup examples in which the various command functions are applied.

- Setting the CPU time and issuing a run request to the CPU simultaneously

Mounted PU

```
CPU CP (site name = 0001cp)
CPU HP (site name = 0001hp)
```

```
>svcpuctl -u 0001cp -time

All PU status CPU(0001cp)=STOP
Input CPU(0001cp) status ? (stop/run)= run
Do you really request OK ? (yes/no)= yes
CPU(0001cp) = RUN
```

- Issuing a stop request to the CPU

Mounted PU

```
CPU CP (site name = 0001cp)
CPU HP (site name = 0001hp)
```

```
>svcpuctl -u 0001cp

All PU status CPU(0001cp)=RUN
Input CPU(0001cp) status ? (stop/run)= stop
Do you really request OK ? (yes/no)= yes
CPU(0001cp) = STOP
```

- Issuing a run request to the CPU without acknowledging whether to stop a PU, and setting the CPU time  
(When the `-s` option is specified to skip the acknowledgment sequence, no message is output to the display.)

```
#svcpuctl -u 0001cp -s -run -time
```

- Displaying the CPU status

```
#svcpuctl -u 0001cp -ss

PU status CPU(0001cp)=STOP
```

## CHAPTER 11 svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT FUNCTIONS

### 11.1 Overview

svdebug is a command that enables the debugging of programs running on the S10VE from the development machine. The command has the following basic functions:

Classification	Subcommand	Function
Task starting or stopping	qu ab re ta su rs tm ct sht	Requests that a task be started. Prevents tasks from being started. Cancels the prevention of task startup. Displays task status information. Suppresses task execution. Cancels suppression of task execution. Starts a task cyclically. Cancels cyclic task startup. Displays cyclic task startup.
Memory printing or patching	md sd bs bg mcp mmv mf	Displays or modifies the contents of memory in accordance with a specified address. Displays or modifies the contents of memory in accordance with a specified name. Sets data in specified bit positions. Displays the data that exists in specified bit positions. Copies the contents of memory. Moves the contents of memory. Sets pattern data in the memory.
System error display	el ss	Displays error logs (runs the svelog command). Displays system status information (runs the svcpuctl command).
Current time setting or display	st gt	Sets the current time. Displays the current time.
Breakpoint setup and resetting	br or stickybr rb rd rr go	Sets or displays breakpoints. Resets breakpoints. Displays registers. Changes the contents of registers. Resumes execution from a breakpoint.
Uploading, downloading, or comparisons	ld sv cm	Transfers backup file data to the controller memory. Transfers S10VE memory data to a backup file. Compares the contents of a backup file with those of the memory in the S10VE.
Enabling or disabling DHP logging	dr ds	Enables DHP logging (runs the svdhp command). Disables DHP logging (runs the svdhp command).
Ladder debug functionality	lbr lrb lrd lrr lgo s	Sets and displays breakpoints. Resets breakpoints. Displays registers. Rewrites the contents of registers. Resumes execution from a breakpoint. Runs steps.
Other	si sp ps pe ver svadm svdhp help q !	Initializes the stack. Displays the amount of stack use. Starts displaying debug statements. Stops displaying debug statements. Displays the version of the CPMS. Displays information about addresses (runs the svadm command). Displays the DHP (runs the svdhp command). Displays a list of subcommands. Terminates the debugger. Runs a command on the development machine.

### 11.2 S10VE Status and Subcommand Availability

Whether subcommands can be run depends on the state of the S10VE. The following table lists the S10VE states and whether subcommands can be run in each state.

Classification	Subcommand	POFF	STOP	RUN	ERR RUN (HP stop)	ERR STOP
Task starting or stopping	qu, ab, re, ta, su, rs, tm, ct, sht	-	-	Y	Y (*6)	-
Memory printing or patching	md, sd, bs, bg, mcp, mmv, mf	-(*1)	Y	Y	Y	Y
System error display	e1, ss	-	Y	Y	Y	Y
Current time setting or display	st, gt	-	-	Y	Y (*6)	-
Breakpoint setup and resetting	br, rb, rd, rr, go	-	-(*5)	Y	Y (*6)	-
Uploading, downloading, or comparisons	ld, sv, cm	-	C (*7)	C (*2)	Y (*2)(*6)	C (*7)
Enabling or disabling DHP logging	dr, ds	-	-	Y	Y (*6)	-
Ladder debug functionality	lbr, lrb, lrd, lrr, lgo, s	-	-	Y	-	-
Other	si, sp, ps, pe	-	-	Y	Y (*6)	-
	svdhp	-	-(*3)	Y	Y (*6)	-(*3)
	ver	-(*4)	Y	Y	Y(*6)	Y
	svadm, help, q, !	Y	Y	Y	Y	Y

Y: Available - : Unavailable C: Available when communication is enabled

(\*1) Backup files can be printed and patched.

(\*2) The following option functions are available only when the S10VE is in the STOP state:

ld: -cm

(\*3) When the S10VE is not in the CPU RUN state, DHP can be collected by using svhidas.

(\*4) Information about the development machine can also be shown in the CPU STOP state.

(\*5) Settings can also be shown in the CPU STOP state.

(\*6) These subcommands cannot be run in the HP site.

(\*7) The ld -g subcommand can also be run in the CPU STOP state.

State	Description
POFF	S10VE power-off state
STOP	OS, middleware, and application stop state
RUN	OS, middleware, and application run state
ERR RUN	Stop state due to an HP error
ERR STOP	Stop state due to a CP error



## 11. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT FUNCTIONS

svdebug provides the aforementioned functions in the form of subcommands. When svdebug is started, a prompt is displayed together with the target site name. You can check the current site to be debugged from the site name. At the prompt, enter an appropriate subcommand to begin debugging. To stop debugging, enter the q subcommand. svdebug will then terminate. An example is as follows:

```
$ svdebug
++ debugger start ++
0001cp> q
++ debugger end ++
```

Access to real-time resources is restricted by the owner type and user type. Set the appropriate user type before starting svdebug.

Most functions are available while the OS of the S10VE is running, but some functions are available even when the OS of the S10VE is not running.

You can suspend the processing of the svdebug subcommands by pressing **Ctrl+C**.

### 11.3 Basic Functions

#### (1) Subcommands to download and upload programs and data

The allocator, loader, and builder (ALB) incorporate programs and data into backup files. The contents of backup files need to be reflected in the S10VE memory in some way. The `ld` subcommand can be used to load the contents of backup files into memory in the S10VE, and the `sv` subcommand to upload the contents of the S10VE main memory to backup files. The `cm` subcommand can also be used to compare the contents of the backup file and the corresponding main memory.

In the following example, the status of a task that has task number 5 (`tn=5`) is checked and the contents of the main memory of the controller are updated. GLB (with the name `indata`) is also updated.

```
$ svdebug
++ debugger start ++
0001cp > ta 5
tn=5   (0x05)   tname=ta5           task state=DORMANT       (0x00000000)
tcb top=0x841e1aac   fact=0x00000000   level=27 (27)
task top=0x3006b000   stack=0x3006c000-0x3006cfff

0001cp > ld -t ta5
address: 0x20000140-0x2000017f
address: 0x3006b000-0x3006b1f7
address: 0x20000140-0x2000017f
0001cp > ld -g indata
address: 0x50050000-0x50050fa0
0001cp >
```

As described here, ALB operations are not reflected directly in the S10VE, which means that there might be a mismatch in the backup files between the development machine and the S10VE.

- Main memory image files are present in the backup file on the development machine but not on the S10VE.  
This occurs when the main memory image files were registered with ALB, but were not incorporated by the `svrpl` command or by the `ld` subcommand of the `svdebug` command.
- Main memory image files are present both in the backup files on the development machine and on the S10VE.  
There are two possible states: (1) identical main memory image files are present on the development machine and S10VE, and (2) different main memory image files are present on the development machine and S10VE. In state (1), the main memory image files that were registered with ALB were only incorporated by the `svrpl` command or by the `ld` subcommand of the `svdebug` command. In state (2), main memory image files are registered and deleted repeatedly with ALB.
- Main memory image files are not present in the backup files on the development machine but are present on the S10VE.  
This occurs when the main memory image files were registered with ALB and incorporated by the `svrpl` command or by the `ld` subcommand of the `svdebug` command, and the relevant files were then deleted with ALB.

## 11. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT FUNCTIONS

- Main memory image files are not present either in the backup file on the development machine or on the S10VE.

This occurs when the main memory image files that were registered with ALB and incorporated by the `svrpl` command or by the `ld` subcommand of the `svdebug` command were then deleted with ALB, and the main memory image files were deleted from the S10VE.

The states of the development machine and S10VE can be checked by using the `svmap` command or the `ld` subcommand of the `svdebug` command. Match the states as needed by using the `svrpl` command or the `ld` subcommand of the `svdebug` command. In particular, when resources are deleted from the development machine, make sure that they are also deleted from the S10VE.

### (2) Task control subcommands

In general, a program downloaded to the S10VE starts when it receives a start request from another task. For convenience during debugging, however, tasks can be started or stopped according to the operator's instructions. `svdebug` supports subcommands that start or stop tasks.

In the following example, a task that has task number 5 (`tn=5`) starts while its status is being checked.

```
$ svdebug
++ debugger start ++
0001cp > ta 5
tn=5   (0x05)   tname=ta5           task state=DORMANT       (0x00000000)
tcb top=0x841e1aac fact=0x00000000 level=27 (27)
task top=0x3006b000 stack=0x3006c000-0x3006cfff

0001cp > re 5
OK (0)
0001cp > qu 5
OK (0)
0001cp > ta 5
tn=5   (0x05)   tname=ta5           task state=IDLE         (0x00000000)
tcb top=0x841e1aac fact=0x00000000 level=27 (27)
task top=0x3006b000 stack=0x3006c000-0x3006cfff

0001cp >
```

**(3) Memory printing and patching subcommands**

svdebug supports subcommands that display or change the contents of memory. These subcommands can be used to perform testing while changing the contents of memory such as GLB to desired values.

Main memory, the backup files, or both can be specified for patching or display purposes. In the following example, a task is started that sets data in GLB containing input data (named `indata`), processes the data, and writes the resulting data to GLB (named `outdata`). Finally, the task checks the processing results.

```
$ svdebug
++ debugger start ++
0001cp > sd
1 name : indata -g
2 storage (s,m,*) : m
3 baddr : 0
4 raddr : 0
0x50050000(0x000000) 00000000 : 0x1000
0x50050004(0x0000004) 00000000 : e
4 raddr : *1
1 name : indata -g
2 storage (s,m,*) : m
3 baddr : 0
4 raddr : 0
0x50050000(0x000000) 00001000 :
0x50050004(0x0000004) 00000000 : e
4 raddr : e
0001cp > qu test
OK(0)
0001cp > sd
1 name : outdata -g
2 storage (s,m,*) : m
3 baddr : 0
4 raddr : 0
0x50051000(0x000000) 00002000 : e
4 raddr : e
0001cp >
```

**(4) Time setting and time display subcommands**

The S10VE has a built-in clock. svdebug can also be used to set a time in the built-in clock. In the following example, the time currently set in the S10VE is checked and then changed.

```
$ svdebug
++ debugger start ++
0001cp > gt
2018.02.07.20:27:32
0001cp > st 2018.02.07.20:30:00
OK(0)
0001cp >
```

(5) System error display and system status display subcommands

During testing or debugging, you might need to reference the system status or a system error. RPDP supports commands that display system errors or the system status (`svelog` and `svcpuctl`). `svdebug` can start these commands as subcommands (`el` and `ss`).

(6) Subcommands for enabling, disabling, and displaying DHP logging

The S10VE provides a debugging helper (DHP) function to support debugging. The DHP function can be used to analyze the operation of the operating system and user tasks. RPDP supports a command that displays DHP traces, enables DHP logging, and disables DHP logging (`svdhp`). `svdebug` can start these commands as subcommands (`svdhp`).

(7) Other subcommands

In addition to the preceding subcommands, other subcommands are also supported.

The `si` and `sp` subcommands simplify the determination of the capacity of the stack used during task execution.

The `svadm` subcommand finds the SAREA name and IRSUB name corresponding to an address.

The `!` subcommand can be used to run commands on the development machine without having to terminate `svdebug`.

The `help` subcommand displays simple explanations.

## 11.4 Other Functions

svdebug supports command line options for users' convenience. The following options can be used to tailor the operation of svdebug.

- i: Outputs keyed-in data to the output file. The output results of subcommands are not logged.
- o: Outputs the date and results of operations to the specified file.  
This option is useful for logging the operations performed by a user.
- r: Runs subcommand lines in the specified command file.  
This option is useful for performing a sequence of operations by using a single operation.  
The file created with the -i option specified can be used as an input.
- s: Directly runs the subcommand specified in this option.  
This option is useful for creating command procedures.
- u *site*: Specifies the name of the site to be processed by the debugger.

## 11.5 Debug Support Commands

### 11.5.1 svelog command

The `svelog` command fetches the error log from the error log buffer in the S10VE over the network, and displays error log entries sequentially from the most recent error. For each error, the error log shows (1) the time that the error occurred; (2) the error code indicating the cause of the error (EC); (3) the task number of the task that caused the error (TN); (4) the contents of the program counter that indicates the location of the error (PC); (5) `iarv0` to `iarv8` and `iarvn1` to `iarvn8` (contents of the program around the location indicated by the PC); which indicates processing that would have been performed; and (6) other information. This command also displays register information and DHP traces to allow for more detailed analysis.

The following are some examples of typical operations.

#### Example 1:

Displaying the error log at the site named `0001cp` in simplified format

```
svelog -u 0001cp -f s
```

#### Example 2:

Displaying all error log entries at the site named `0001cp`

```
svelog -u 0001cp -f m
```

Even when the `-f m` options are omitted, the same result is obtained.

#### Example 3:

Displaying the error log and DHP traces at the site named `0001cp`

```
svelog -u 0001cp -f l
```

#### Example 4:

Displaying the error log and DHP traces at the site named `0001cp` and storing the information displayed on the screen in the file named "abc"

```
svelog -u 0001cp -f l -o abc
```

### 11.5.2 svdhp command

The `svdhp` command fetches the current DHP trace from the DHP trace buffer in the S10VE over the network and displays the trace.

The displayed DHP trace includes the DHP logging time, DHP trace points, and data required for analysis. The command also stops or restarts DHP logging.

The following are some examples of typical operations.

#### Example 1:

Displaying DHP traces at the site named “0001cp” for the specified number of counts (10)

```
svdhp -u 0001cp +10
```

If a count (+10 in this example) is omitted, all DHP traces are displayed.

#### Example 2:

Stopping DHP logging at the site named “0001cp”

```
svdhp -u 0001cp -off
```

#### Example 3:

Restarting DHP logging at the site named “0001cp”

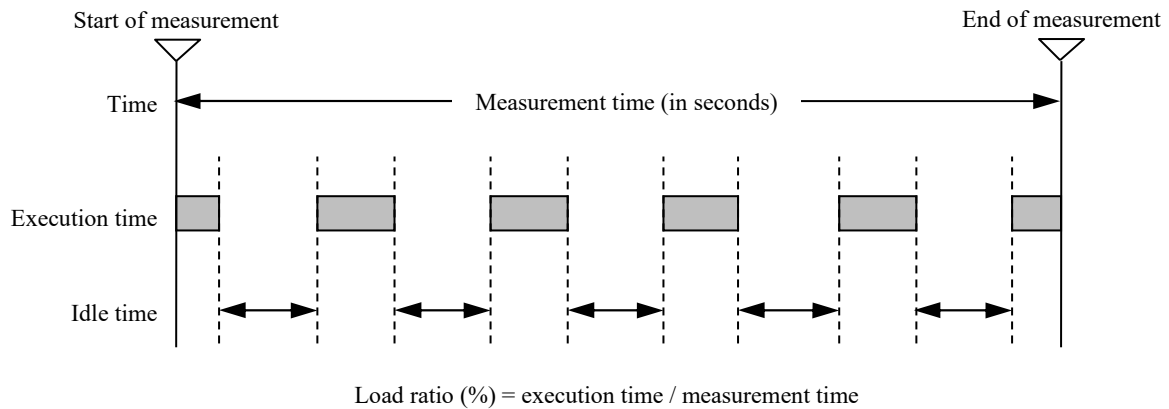
```
svdhp -u 0001cp -on
```



## 11. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT FUNCTIONS

### 11.5.3 svcpunow command

The `svcpunow` command displays the load ratio of the S10VE at the specified site (PU). The load ratio is the percentage of the time from the start to the end of measurement during which the PU is used for program executions (total execution time).



The following is an example of a typical operation.

Example: Displaying the load ratio measured over 10 seconds at the site named “0001cp”

```
#svcpunow -u 0001cp -t 10
2018/02/07 09:56:00 SITE=0001cp ** 10 second wait **
CPU(0001cp) load ratio = 12.34%
```

(\*1) (\*2) (\*3)

Note: The underlined text indicates the portion that is entered by the user.

(\*1) PU name

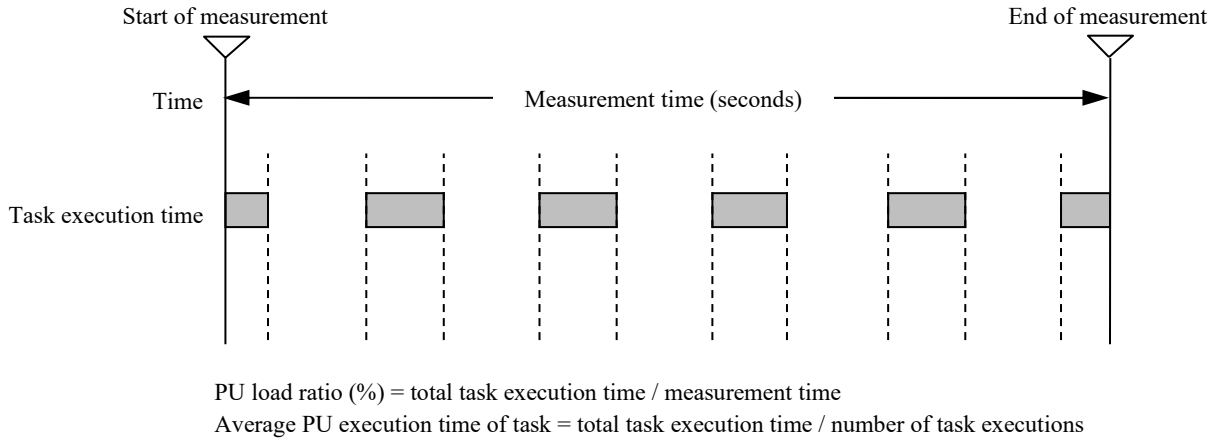
(\*2) Site name

(\*3) PU load ratio

If the measurement time (specified by `-t 10` in this example) is omitted, the load ratio measured over one second is displayed.

11.5.4 svtimex command

The `svtimex` command displays activity information for a task registered in the S10VE. The activity information consists of the PU load ratio of the task, the number of executions, execution times, and the average execution time.



The following are some examples of typical operations.

Example 1: Measuring the activity of the following task for 10 seconds:

Task name = `taska`, task number = 123

```
#svtimex -u 0001cp taska -t 10
2018/02/07 09:56:00 SITE=0001cp ** 10 second wait **
taska(123) load ratio=3.00% execute count=10 total time=0.030sec average time=0.003sec
```

(\*1) (\*2) (\*3) (\*4) (\*5)

Note: The underlined text indicates the portion that is entered by the user.

- (\*1) Task name (task number)
- (\*2) PU load ratio of the task
- (\*3) Number of task executions
- (\*4) Total task execution time
- (\*5) Average PU execution time of the task

If a measurement time (such as `-t 10`) is not specified, the load ratio is measured over one second.

## 11. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT FUNCTIONS

Example 2: Measuring the activities of the following tasks over 3,600 seconds by using the interactive interface:

Task name = `taska`, task number = 123

Task name = `taskb`, task number = 124

Task name = `task22c`, task number = 111

```
#svtimex -u 0001cp
SITE=0001cp Task measuring period[sec] = 3600
Task name or number = taska
Task name or number = taskb
Task name or number = 111
Task name or number = Enter (specify nothing)
2018/02/07 13:30:24 SITE=0001cp ** 3600 second wait **
taska(123) load ratio=3.00% execute count=36000 total time=108.000sec average time=0.003sec
taskb(124) load ratio=2.50% execute count=18000 total time=90.000sec average time=0.005sec
task22c(111) load ratio=0.01% execute count=360 total time=0.360sec average time=0.001sec
```

Note: The underlined text indicates the portion that is entered by the user.

## **PART 2 COMMAND REFERENCE**

## CHAPTER 1 COMPILER

### Name

svdatagen

### Syntax

svdatagen [-u site] *file*

### Description

The `svdatagen` command generates a binary file of loadable initial-value data in a GLB backup file by using the loader (`svload`).

The name of the generated binary file is output by using the input file name and changing its suffix to `.bin`. If the input file name has no suffix, a binary file is generated by using the name of the input file and the suffix `.bin`.

### Arguments

`-u site`: Specifies the site name to reference when address resolution for the GLB/VAL name is required to generate the initial values.

If this option is omitted, processing is performed for the site set by the environment variable `RSSITE`.

The site name does not need to be specified if address resolution for the GLB/VAL name is not required.

*file*: Specifies a text file that includes the initial-value data.

Only one input file can be specified.

This file can contain multiple initial GLB values, including GLB data for multiple areas.

### Termination codes

0: Normal termination

Other than 0: Abnormal termination

## Data layout

The following table shows the layout of the binary data generated by this command. If the initial-value data is structured, assuming that the structure begins with a 4-byte boundary, the alignment of the members of the structure are as shown in the following table.

Table 2-1 Binary Data Layout

Type	Alignment (byte)
char	1
short	2
int	4
long	4
float	4
double	4
Array	Dependent on element type
Structure	Same as the member of the maximum alignment number

## CHAPTER 2 PROGRAMMING COMMAND

### Name

`makehce` - Maintains, updates, and regenerates program groups

### Syntax

`makehce [-Seiknpqrst] [-f makefile] [variable=value...] [target...]`

### Description

- Structure of makefile  
Target lines, shell command lines, macro definitions, and include lines can be included in makefile.
- Target lines  
A target line consists of a non-null list of targets delimited by blanks, colons (:) or double colons (::), as well as a list of prerequisite files called dependents. When creating a file name as a dependent, the pattern matching notation is supported.
- Shell command lines  
The text following a semicolon (;) in the target line and all subsequent lines beginning with a tab are shell commands to be run to update the target. The first line that does not begin with a tab or hash mark (#) begins with a new target definition, macro definition, or include line. Using the sequence of a backslash (\) followed by a newline character makes it possible to continue the shell command across multiple lines.  
A target line associated with the command line is called a rule.

- Macro

A line in the format of *character-string-1* = *character-string-2* is a macro definition. Macros can be defined at any position of *makefile*, but are usually defined together at the beginning. The first character string is the macro name, and the second is the macro value. The second character string is defined as the entire string of characters up to a comment character or non-extended newline character. A null or tab on either side of the equal sign (=) are ignored (except those in comments). The representation  $\$character-string-1$  at another position in *makefile* is replaced with the second character string. When a single-character macro name is used and no character string is defined as its replacement, parentheses can be used to enclose an option. A replacement character string can be specified as an option by using  $\$(character-string-1 [ : subset-1 = [subset-2]])$ . When such a character string is specified, all instances of subset 1 that do not overlap at the end of the subcharacter string within the *character-string-1* value are replaced with subset 2. A subcharacter string in a macro value is delimited by a blank, a tab, a newline character, or the beginning of a line.

Example:

```
OBJS = file1.obj file2.obj file3.obj
```

Suppose that the preceding character string is entered.

```
$(OBJS:.obj=.c)
```

The preceding character string would then evaluate as follows:

```
file1.c file2.c file3.c
```

A macro value can include a reference to another macro:

```
ONE = 1
TWELVE = $(ONE) 2
```

The value for  $\$(TWELVE)$  is set to  $\$(ONE) 2$ , but this value expands to 12 when it is used in a target line, command line, or include line. After that, if the value of ONE is further changed by *makefile* or another definition of the command line, the reference to  $\$(TWELVE)$  reflects this change.

Specifying a macro definition by using the command line disables the definition in *makefile*. Specific macros are automatically defined for *makehce*.



## 2. PROGRAMMING COMMAND

- Include line

When the first seven characters of *makefile* are “include” and one or more null or tab characters follow, the subsequent part of the line is interpreted as a file name. This part is read and processed as a separate *makefile* after any macro in the file is extended by the current `makehce` call.

- General description

The `makehce` command runs the commands entered in *makefile* to update one or more target names. A target name becomes a program name. When the `-f` option is not specified, *makefile* is processed first, and then *Makefile* is processed. When `-` is specified in the `-f` option, the standard input is used. Multiple `-f` options can be specified. The arguments of *makefile* are processed in the specified sequence. To specify multiple *makefile* names, the `-f` option must be attached to the beginning of each *makefile* name. If a built-in rule or a macro is present, the contents of *makefile* take priority over the built-in rule or macro.

If no target name is specified in the command line, the `makehce` command updates the first target of *makefile*. Only targets subordinated to a newer file are updated.

If no file is specified, an old one file is assumed. If necessary, all dependents of the target (those subordinate to the target) are recursively updated before the target is updated. This affects the depth priority (vertical) update of the subordinate tree related to targets.

If a target does not have an explicit dependent specified after the delimiting character in the target line, the shell command associated with the target is run when the target becomes old.

The target line can include a single colon or double colon between one or more target names and explicit dependent names. A target name can be specified across multiple target lines, but all target lines must be the same type (either single colon or double colon). In the case of a single colon, only one target line can be associated with an explicit command. When an explicit command is specified, if the target for which a dependent is specified in any line becomes old, the explicit command is run. However, when no explicit command is specified, the default rule can be run. In the case of a double colon, an explicit command can be associated with multiple target lines that include the target name. When the target for which a dependent is specified in a specific line becomes old, the command related to the line is run.

Target lines and their related shell command lines are also called the rule. Hash marks (`#`) and newline characters enclose comments in *makefile*, with the exception of those in the rule.

Comments in the rule are determined by the SHELL macro settings.

The following *makefile* shows that `pgm` is subordinate to two files (`a.obj` and `b.obj`) and that these files are subordinate to the corresponding source files (`a.c` and `b.c`) and the common file `incl.h`.

```

OBJS = a.obj b.obj
pgm:$(OBJS)
    svload +P -o pgm -a tsk00 -w 128 4096 $(OBJS)
a.obj:incl.h a.c
    shc a.c
b.obj:incl.h b.c
    shc b.c

```

Command lines are run sequentially by the shell. Either or both of the prefixes `-` and `+` can be specified for each command line. These two prefixes are described later.

The `makehce` command terminates when it returns a command termination code other than 0. When the `-i` option is specified, or the special target `.IGNORE` is present in *makefile*, an error message is printed to the standard output. However, the `makehce` command continues to run *makefile* even if many command lines cause errors. When a hyphen (`-`) is present at the beginning of a command line, the error returned by the line is printed to the standard output, but the `makehce` command does not terminate. Using the prefix `-` selects an error in *makefile* and ignores it.

When a command line returns an error while the `-k` option is specified, work is abandoned in the current target, but work continues in other branches that are not subordinate to the target. When the `-k` option is present in the `MAKEFLAGS` environment variable, the default processing can be restored by specifying the `-S` option.

The `-n` option specifies command line print without execution. However, when the prefix `+` is appended, the line is always run. The `-t` (`touch`) option updates the file modification date without running a command.

## 2. PROGRAMMING COMMAND

Command lines are printed before they are run, but printing is suppressed when the character @ is present at the beginning of a command line. When the `-s` option is specified, or the special target `.SILENT` is present in *makefile*, the printing of all command lines is suppressed. All information to be printed by the `makehce` command, with the exception of the starting tag, is directly transferred to the shell without being modified.

Therefore:

```
echo a\  
b
```

This command generates the following character string in the same way that the shell does.

```
ab
```

- Options

Options can be specified in any order. All options except for the `-f` option can be specified independently, or collectively by using hyphens (-).

`-e`: Environment variables take priority over the contents of *makefile*.

`-f`: *makefile*

The specified file name becomes *makefile*. The file name given as `-` displays the standard input. If a built-in rule or a macro is present, the contents of *makefile* take priority over the built-in rule or macro. This option can be specified multiple times, and the options are processed in the order that they are specified.

`-p`: Writes a complete set of macro definitions and the target content to the standard output.

`-i`: Ignores the error code returned from the command. This mode is also valid when the special target name `.IGNORE` is included in *makefile*.

`-k`: When the command returns a state value other than 0, work for the current entry is abandoned, but work for other branches that are not subordinate to the target continues, as opposed to the `-S` option.

If the `-k` and `-S` options are specified simultaneously, the option specified later is used.

`-n`: Indicates non-execution mode. The command is printed but is not run. Lines beginning with @ are also printed. However, lines with the prefix `+` attached to the command are run.

`-q`: Indicates a question. The `makehce` command returns a value of 0 or another value depending on whether the target file is the latest file or not. Even if this option is specified, the target file is not updated.

`-r`: Clears the suffix list without using the built-in rule.

`-s`: Indicates silent mode. Command lines are run without being printed to the standard output. This mode is also valid when the special target name `.SILENT` is included in *makefile*.

`-S`: When an error occurs during execution of the target updating command, this option terminates the command. This option is the default value, as opposed to the `-k` option. If the `-k` and `-S` options are specified simultaneously, the option specified last is used.

Specifying this option can disable the `k` flag in the `MAKEFLAGS` environment variable.

`-t`: Processes (updates) the target file without issuing a regular command.

*macro-name* = *value*

Zero or more command line macro definitions can be specified.

*target-name*

Zero or more target names can be specified in *makefile*. The specified targets are updated by the `makehce` command. If no file name is specified, the `makehce` command updates the first target other than the reasoning rule in *makefile*.

- Environment

Except for the `SHELL` environment variable, which is always ignored, all variables defined for the environment are read by the `makehce` command and are processed as a macro definition. Undefined variables and variables provided with a definition consisting of null characters are included in the `makehce` command.

The macro definition has four sources that are read in the following order: internal (default), current environment, *makefile*, and command line. Due to this processing order, the macro assignment in *makefile* takes priority over environment variables. When the `-e` option is specified, environment variables take priority over the macro assignment in *makefile*. Macro definitions in command lines always take priority over any other definitions.

The `MAKEFLAGS` environment variable processed by the `makehce` command is processed assuming that it includes valid input options (other than `-f`, `-p`, and `-d`) defined in command lines. The `MAKEFLAGS` variable can also be specified in *makefile*.

If `MAKEFLAGS` is not specified in a command line or *makefile*, the `makehce` command uniquely composes the variable and includes the options specified in command lines and the default option in the variable. It then transfers the variable to commands. Therefore, the current input options are always included in `MAKEFLAGS`. This is very useful when the `makehce` command is run recursively. For this reason, running `makehce -n` recursively throughout the entire software system can be used to examine what has been run. This is a way to debug *makefile* related to the software project without running any commands.

## 2. PROGRAMMING COMMAND

- Suffix

In many cases, a suffix is attached to targets and dependents. Because information on specific suffixes is included in the `makehce` command, the reasoning rule that is suitable for updating targets can be identified by referring to the information. (See the following *Reasoning rule* section.) A list of the current default suffixes is as follows.

```
.obj .c .c~ .src
```

These suffixes are defined as dependents of the special built-in target `.SUFFIXES`. This is defined automatically by the `makehce` command.

Additional suffixes can be specified in *makefile* as the dependent list of `.SUFFIXES`. These additional values are added to the default values. Thus, multiple suffixes are accumulated. The suffix list sequence has a meaning. (See the following *Reasoning rule* section.) To change the sequence of suffixes, define `.SUFFIXES` that made the dependent list NULL, clear the current `.SUFFIXES` value, and then define `.SUFFIXES` with suffixes that are specified in the desired order.

- Reasoning rule

A specific target or dependent name (such as a name ending with `.obj`) is provided with dependents that can be inferred (such as `.c` and `.src`). When *makefile* has no command to update these names, or when a dependent file that can be inferred exists, the dependent file is compiled to update targets. In this case, the reasoning rule of the `makehce` command examines suffixes and determines an appropriate reasoning rule to create a file from other files. The following default reasoning rules are defined at present.

### Single-suffix rule

```
.c~
```

### Double-suffix rule

```
.c.obj .c~.obj  
.s.obj
```

The double-suffix reasoning rule (`.c.obj`) defines the method to create `x.obj` from `x.c`. The rule that creates a file that has suffix `.obj` from a file that has suffix `.c` is specified as an entry that has `.c.obj` as a target and that has no dependent. The shell command defined for the target defines the rule to create an `.obj` file from a `.c` file. A target name beginning with a period (`.`) that does not include a slash (`/`) is not identified as an explicit target rule, but is rather identified as an implicit reasoning rule. A target provided with a period is the single-suffix reasoning rule. A target provided with two periods is the double-suffix reasoning rule. Users can define additional reasoning rules in *makefile*, and also can re-define or cancel the default reasoning rule.

The default reasoning rule that changes a `.c` file to an `.obj` file is as follows.

```
.c.obj:
    $(CC) $(CFLAGS) -c $<
```

The specific macro is used in the default reasoning rule so that options can be inserted into the result command. For example, `CFLAGS` is used in the compiler options for `shc`.

This macro is defined automatically by the `makehce` command, but can be redefined in *makefile*.

There are some special built-in macros used in the reasoning rule (`<`). (See the *Built-in macros section*.)

When the target has no explicit dependent, or when the dependent has no target that matches the defined explicit rule, the first reasoning rule that the `makehce` command searches meets both files that match the dependent suffix of the target (`NULL` in some cases) and other suffixes of the rule. Because the `makehce` command performs this search from the front to the back of the `.SUFFIXES` value list, the `.SUFFIXES` definition sequence has an important meaning. The example shown in *General description* can be rewritten more simply because the `makehce` command defines the reasoning rule `.c.obj`.

```
OBJS = a.obj b.obj
pgm: $(OBJS)
    svload +P -o pgm -a tsk00 -w 128 4096
$(OBJS)
$(OBJS): incl.h
```

## 2. PROGRAMMING COMMAND

- Built-in targets

The `makehce` command has the following pieces of information on special targets. Specify the pieces of information to be issued by *makefile*. (Except for `.SUFFIXES`, these pieces of information are automatically specified by the `makehce` command, but can be modified.)

- `.DEFAULT`: When a file must be created but there is no explicit command in the file or related built-in rule, a command with purpose name `.DEFAULT` is used if `.DEFAULT` is defined in *makefile*. `.DEFAULT` has no explicit dependent.
- `.PRECIOUS`: Even if the `QUIT`, `INTERRUPT`, `TERMINATE`, or `HANGUP` command is entered, the dependent of this target is not deleted.
- `.SILENT`: This information has the same function as the `-s` option. There is no need to specify a dependent or explicit command.
- `.IGNORE`: This information has the same function as the `-i` option. There is no need to specify a dependent or explicit command.
- `.SUFFIXES`: Explicit dependents of `.SUFFIXES` are added to the built-in list of known suffixes and are used together with the reasoning rule. If `.SUFFIXES` has no dependent, the list of known suffixes is cleared. No commands are associated with `.SUFFIXES`.

- Built-in macros

Five macros that are useful for creating a target creation rule are included. To clearly define these macros, the terms of *target* and *dependent* need to be explained. When the `makehce` command updates targets, it actually generates all targets to be updated. Before a rule (explicit or implicit) is applied to a target, recursion is performed for each dependent of the target. After recursion, each dependent becomes a target, and a unique dependent is generated. In such a dependent, recursion continues until a target that has no dependent is found. Not all targets processed by the `makehce` command are specified by *makefile* as an explicit target. Some targets are made explicit dependents by *makefile* and some targets become implicit dependents that are generated when the `makehce` command recursively updates targets. For example, consider the execution of the following *makefile*:

```
pgm: a.obj b.obj
    svload +P -o pgm -a tsk00 -w 128 4096 a.obj b.obj
```

The following targets to be created are generated:

`pgm`: Two dependents and a single explicit rule are present.

`a.obj`: An implicit dependent `a.c` that meets the implicit rule `.c.obj` is present.

`a.c`: There is no implicit dependent or implicit rule. This stops recursion and returns the date and time of the last correction of the `a.c` file.

`b.obj`: An implicit dependent `b.c` that meets the implicit rule `.c.obj` is present.

`b.c`: There is no implicit dependent or implicit rule. This stops recursion and returns the date and time of the last correction of the `b.c` file.

In these definitions (`$$`, `?`, `<`, `*`, and `$$`), the word *target* indicates the following:

- A target specified by *makefile*
- An explicit dependent (specified in *makefile*) that is to be a target when the `makehce` command performs recursion
- An implicit dependent that is to be a target when the `makehce` command performs recursion (generated as a result of identification of the reasoning rule and file)

The word *dependent* indicates the following:

- An explicit dependent (specified in *makefile*) on specific targets
- An explicit dependent generated as a result of identification of the appropriate reasoning rule that meets target suffix and corresponding file

It can be useful to consider the target rule to be a user-specified rule for a specific target name, and to consider the reasoning rule to be a user-specified or `makehce`-specified rule for a specific target name class. Furthermore, it can be useful to memorize the fact that, when the `makehce` command performs recursion with explicit and implicit dependents, the target name and the corresponding dependent name value vary, and that the reasoning rule is applied only to implicit dependents for which the target rule is not defined in *makefile* or explicit dependents.



## 2. PROGRAMMING COMMAND

- `$$`: The `$$` macro becomes the full target name of the current target. This macro is evaluated as both the target and the reasoning rule.
- `$$?`: The `$$?` macro is a list of old dependents on the current target, which is practically a recreated module. This macro is evaluated as both the target and the reasoning rule, but is usually used only for the target rule. Usually, the `$$?` macro is evaluated as a single name in the reasoning rule, but is evaluated as multiple names in the target rule in some cases.
- `$$<`: The `$$<` macro is evaluated as the source file name corresponding to the implicit rule that meets the suffix of the created target in the reasoning rule. In other words, this macro is an old file related to the target. In the `.DEFAULT` rule, the `$$<` macro is evaluated as the current target name. The `$$<` macro is evaluated only as the reasoning rule. Therefore, the `$$<` macro is evaluated as a `.c` file in the `.c.obj` rule. The following shows an example of creating an optimized `.obj` file from the `.c` file.

```
.c.obj:
    shc -c -O $$*.c
```

or

```
.c.obj:
    shc -c -O $$<
```

- `$$*`: The `$$*` macro is the current target name from which the suffix is deleted. This macro is evaluated only for the reasoning rule.

In addition to the built-in macros ( $\$@$ ,  $\$?$ ,  $\$<$ , and  $\$*$ ) listed here, other macros for general use are defined by the `makehce` command. These macros are available in the target rule in *makefile*, and can also be redefined by *makefile*.

$\$\$@$ : The  $\$\$@$  macro has a meaning only in subordinate lines. Macros in this format are called dynamic dependents, because they are evaluated when dependents are actually processed. The  $\$\$@$  macro is evaluated in the same way as the  $\$@$  macro in the command line. That is, the  $\$\$@$  macro is evaluated as the current target name. This macro can be useful when creating many executable files, when each of which has only one source file. For example, the following command can be created by using only this rule:

```

CMDS = cat echo cmp chown
$(CMDS) : $$@.c
        $(CC) -O $?
        svload +P -o $@ -a tsk00 -w 128 4096 $*.obj

```

When this *makefile* is called by `makehce cat echo cmp chown`, the `makehce` command creates each target by using the general rule. The  $\$\$@$  macro is evaluated as `cat` when the target is `cat`, and is evaluated as `echo` when the target is `echo`.

- Return value

The `makehce` command returns 0 when it terminates successfully, and returns a value larger than 0 when an error occurs.

## CHAPTER 3 ALLOCATOR

### Name

svdfa

### Syntax

svdfa *aname* *size* [*option*]

### Description

The `svdfa` command allocates a specified split area within a specified global area and generates a backup file.

### Arguments

*aname*: Name of the split area to be allocated.

*size*: Size of the split area to be allocated. Specify a multiple of 4,096 bytes. If you do not do so, a warning message is displayed and the specified size is rounded up to the nearest multiple of 4,096 bytes.

### Options

- p: Allocates an area to store a task.
- s: Allocates an area to store a subprogram.
- gi: Allocates a GLB area with an initial value in a read/write global area.
- gw: Allocates a GLB area without an initial value in a read/write global area.
- gr: Allocates a GLB area with an initial value in a read-only global area.
- cmi: Allocates a CM area with an initial value in the memory area shared by PUs.
- cmw: Allocates a CM area without an initial value in the memory area shared by PUs.
- dcmi: Allocates a DCM area with an initial value in the duplexed shared memory area. Because this is an extended option, it cannot be used with the S10VE.
- dcmw: Allocates a DCM area without an initial value in the duplexed shared memory area. Because this is an extended option, it cannot be used with the S10VE.
- S: Specifies the use of the system access rights. If this option is omitted, the environment variable `RSUTYP` that has been specified previously takes effect. By default, this variable is set to the user's access rights (`RSUTYP=u`).
- u *site*: Specifies the site name (*site*) to be processed by the allocator. If this option is omitted, the allocator processes the site that is set in the environment variable `RSSITE`.
- f *adr*: For *adr*, specify the address of the split area to be allocated, relative to the beginning of the global area. The address must be a multiple of 4,096. If it is not, a warning message is displayed and the specified address is rounded up to the nearest multiple of 4,096. If this option is omitted, an area is automatically allocated; that is, the first available area that is found is allocated.

## Notes

If none of the following options is specified, `-p` is assumed: `-p`, `-s`, `-gi`, `-gw`, `-gr`, `-cmi`, `-cmw`, `-dcmi`, or `-dcmw`.

When a secondary partition area is allocated by the `svdfa` command in the split area allocated with `-gi`, `-gr`, or `-cmi` specified, the secondary partition area is zero-cleared.

When allocating a CM area, be sure to specify the address of the CM area to be allocated by using the `-f` option. If the `-f` option is omitted, an error occurs.

The address and size of the CM area to be allocated must be the same at all sites in the unit. If it is not, data might be lost.

Table 2-2 shows the option combinations that can be specified.

Table 2-2 Combinations of `svdfa` Options

Parameter	Area type							
	Task	Sub-program	Read/write GLB with an initial value	Read/write GLB without an initial value	Read-only GLB with an initial value	CM with an initial value	CM without an initial value	
<i>aname</i>	R	R	R	R	R	R	R	
<i>size</i>	R	R	R	R	R	R	R	
Options	<code>-p</code>	Y (default)	–	–	–	–	–	
	<code>-s</code>	–	R	–	–	–	–	
	<code>-gi</code>	–	–	R	–	–	–	
	<code>-gw</code>	–	–	–	R	–	–	
	<code>-gr</code>	–	–	–	–	R	–	
	<code>-cmi</code>	–	–	–	–	–	R	
	<code>-cmw</code>	–	–	–	–	–	–	R
	<code>-dcmi</code>	–	–	–	–	–	–	–
	<code>-dcmw</code>	–	–	–	–	–	–	–
	<code>-s</code>	Y	Y	Y	Y	Y	Y	Y
	<code>-u site</code>	Y	Y	Y	Y	Y	Y	Y
<code>-f adr</code>	Y	Y	Y	Y	Y	R	R	

R: Required Y: Can be specified –: Cannot be specified

The following table shows the relationships between the user types and the owner types for areas to be allocated:

User type	Owner type for area to be allocated	
	System	User
System	Y	–
User	–	Y

Y: Can be generated –: Cannot be generated

## Termination codes

The `svdfa` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

### 3. ALLOCATOR

#### Name

svdla

#### Syntax

svdla *aname* [*option*]

#### Description

The `svdla` command deletes split areas allocated by `svdfa` and backup files.

#### Arguments

*aname*: Name of a split area to be deleted.

#### Options

- S: Specifies the use of the system access rights. If this option is omitted, the default (set in the `RSUTYP` environment variable in advance) is used.
- u *site*: For *site*, specify the name of the site to be handled by the allocator. If this option is omitted, the allocator processes the site that is set in the environment variable `RSSITE`.

#### Notes

When secondary partition areas of a readable/writable global area, a read-only global area, inter-PU shared memory, or duplexed shared memory exist in the specified split area, secondary partition areas in the specified split area are also deleted at the same time.

The following table shows the relationships between user types and owner types for the areas to be deleted:

User type	Owner type for area to be deleted	
	System	User
System	Y	Y
User	-	Y

Y: Can be deleted    -: Cannot be deleted

#### Termination codes

The `svdla` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

**Name**

svdfs

**Syntax**svdfs *aname sname size [option]***Description**

The `svdfs` command allocates a global secondary partition area in a split area allocated by `svdfa`.

The command initializes the allocated area to zero (0).

**Arguments**

*aname*: Specifies the name of the split area to be divided into secondary partition areas.

*sname*: Specifies the name of the global secondary partition area to be allocated.

*size*: Size of the secondary partition area to be allocated (in bytes).

**Options**

- S: Specifies the use of the system access rights. If this option is omitted, the default (set in the `RSUTYP` environment variable in advance) is used.
- u *site*: For *site*, specify the name of the site to be handled by the allocator. If this option is omitted, the allocator processes the site that is set in the environment variable `RSSITE`.
- l *adr*: For *adr*, specify the address of the secondary partition area to be allocated, relative to the beginning of the split area. The address must be a multiple of 4. If it is not, a warning message is displayed and the specified address is rounded up to the nearest multiple of 4. If this option is omitted, an area is automatically allocated; that is, the first available area that is found is allocated.
- a *align*: For *align*, specify the number of alignments with a value of 2 raised to the *n*th power (where  $0 \leq n \leq 12$ ). The secondary partition area is allocated based on this number. The default is 2.
- t *svtype*: Affects alignment in accordance with the data type specified by *svtype*. Table 2-3 shows the relationship between the value specified for *svtype* and the alignment count.
- e *idxnum*: When using the allocated secondary partition area as an indirect link global area, specify the entry number to be assigned to the secondary partition area for *idxnum*. If this option is omitted, no entry number is assigned. For *idxnum*, specify a value in the range from 1 to 7,934.

### 3. ALLOCATOR

#### Notes

- When secondary partition areas allocated without specifying the `-e` option are used as indirect link global areas, the `svirglb` command can be used to assign an entry number to the desired secondary partition area.
- Determine the number of alignments according to the size declared for the data.
- The `-l`, `-a`, `-t` options are mutually exclusive. Do not specify them together.
- When allocating secondary partition areas in a split area allocated in the CM area, the address and size of the CP site and HP site must be the same.  
If they are not, data might be lost.

Table 2-4 shows the option combinations that can be specified.

Table 2-3 Relationship between the Value Specified for `svtype` and the Alignment Count

svtype	Data type	Alignment count
1	char	0 (1 byte)
2	short	1 (2 bytes)
3	long	2 (4 bytes)
4	struct	3 (8 bytes)
5	float	2 (4 bytes)
6	double	3 (8 bytes)
7	long double	4 (16 bytes)

When specifying the data type, the alignment specifications must be the same as those of the R700.

Table 2-4 Combinations of `svdfs` Options

Parameter	Area type							
	Task	Sub-program	Read/write GLB with an initial value	Read/write GLB without an initial value	Read-only GLB with an initial value	CM with an initial value	CM without an initial value	
<i>aname</i>	-	-	R	R	R	R	R	
<i>sname</i>	-	-	R	R	R	R	R	
<i>size</i>	-	-	R	R	R	R	R	
Options	<code>-S</code>	-	-	Y	Y	Y	Y	Y
	<code>-u site</code>	-	-	Y	Y	Y	Y	Y
	<code>-l adr</code>	-	-	Y	Y	Y	Y	Y
	<code>-a align</code>	-	-	Y	Y	Y	Y	Y
	<code>-t svtype</code>	-	-	Y	Y	Y	Y	Y
	<code>-e index</code>	-	-	Y	Y	Y	Y	Y

R: Required Y: Can be specified -: Cannot be specified

The following table shows the relationships between the user types and the owner types for secondary partition areas to be allocated:

User type	Owner type for secondary partition area	
	System	User
System	Y (System)	Y (User)
User	–	Y (User)

Y: Can be allocated –: Cannot be allocated

The type in parentheses is the owner type of the allocated secondary partition area.

### Termination codes

The `svdfs` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination



### 3. ALLOCATOR

#### Name

`svdls`

#### Syntax

`svdls sname [option]`

#### Description

The `svdls` command deletes secondary partition areas allocated by `svdfs`.

#### Arguments

*sname*: External name of the secondary partition area to be deleted.

#### Options

- S: Specifies the use of the system access rights. If this option is omitted, the default (set in the `RSUTYP` environment variable in advance) is used.
- u *site*: For *site*, specify the name of the site to be handled by the allocator. If this option is omitted, the allocator processes the site that is set in the environment variable `RSSITE`.

#### Notes

The following table shows the relationships between the user types and the owner types for areas to be deleted:

User type	Owner type for area to be deleted	
	System	User
System	Y	Y
User	-	Y

Y: Can be deleted    -: Cannot be deleted

#### Termination codes

The `svdls` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

**Name**

svdfv

**Syntax**svdfv *ename value [option]***Description**

The `svdfv` command registers information about external references of value information.

**Arguments**

*ename*: External name to be registered.

*value*: Value of the external name (where  $-2^{31} \leq value \leq 2^{31} - 1$ ).

**Options**

-S: Specifies the use of the system access rights. If this option is omitted, the default (set in the `RSUTYP` environment variable in advance) is used.

-u *site*: For *site*, specify the name of the site to be handled by the allocator. If this option is omitted, the allocator processes the site that is set in the environment variable `RSSITE`.

**Notes**

The value (*value*) is handled as an integer. If a value that is outside the valid range ( $-2^{31} \leq value \leq 2^{31} - 1$ ) is specified, an error occurs.

The value used during execution depends on the type of the value name in the language used (specified by *ename*).

The following table shows the relationships between the user types and the owner types for value areas to be allocated:

User type	Owner type	
	System	User
System	Y	-
User	-	Y

Y: Can be generated    -: Cannot be generated

**Termination codes**

The `svdfv` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

### 3. ALLOCATOR

#### Name

svdlv

#### Syntax

svdlv *ename* [*option*]

#### Description

The `svdlv` command deletes external reference information registered by `svdfv`.

#### Arguments

*ename*: Specifies the external name to be deleted.

#### Options

- S: Specifies the use of the system access rights. If this option is omitted, the default (set in the `RSUTYP` environment variable in advance) is used.
- u *site*: For *site*, specify the name of the site to be handled by the allocator. If this option is omitted, the allocator processes the site that is set in the environment variable `RSSITE`.

#### Notes

The following shows the relationship between the user type and deletion value owner type.

User type	Deletion value owner type	
	System	User
System	Y	Y
User	-	Y

Y: Can be deleted    -: Cannot be deleted

#### Termination codes

The `svdlv` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

## CHAPTER 4 LOADER

### Name

svload

### Syntax

svload [*option*] *file*...

### Description

The `svload` command links the specified object file or library; registers it in the development environment by using a name that represents a program, a subprogram, or data; and stores the file or library in a backup file.

### Arguments

*file*: Specify one or more object files or libraries to be linked.

### Options

- S: This option specifies the system processing mode. When this option is omitted, the predefined default access rights (defined by the environment variable `RSUTYP`) are selected.
- u *site*: For *site*, specify the name of the site to be handled by the loader. When this option is omitted, the loader processes the site that is set in the environment variable `RSSITE`.
- C *n*: For *n*, specify the first address of the area for storing the program or subprogram. When specifying the address, ensure that it is a multiple of 4096 for a program or a multiple of 32 for a subprogram. When the specified address is not a multiple of 4096 or 32, the system displays a warning message and raises the specified value to a multiple of 4096 or 32.
- p *n*: For *n*, specify the relative address in the area at which to start loading. This option is valid for a program or subprogram. When this option is omitted, the file is automatically registered. This option and the `-C` option are mutually exclusive. Do not specify them together. When specifying the address, ensure that it is a multiple of 4096 for a program or a multiple of 32 for a subprogram. When the specified address is not a multiple of 4096 or 32, the system displays a warning message and raises the specified value to a multiple of 4096 or 32.
- a *area*: For *area*, specify an area to which to load the program or subprogram. When the file is to be loaded is a program or subprogram, this option must be specified.
- +P: This option specifies that the file is to be loaded as a program (task).
- +I: This option specifies that the file is to be loaded as an indirect link subroutine (IRSUB).
- +U: This option specifies that the file is to be loaded as a built-in subroutine (ULSUB).
- +D: This option specifies that the file is to be loaded as global or CM data. The data type depends on the attribute of the split area to which the data belongs.
- +B: This option specifies that the binary data generated by the data generator (`svdatagen`) is to be loaded as global or CM data. The data type depends on the attribute of the split area to which the data belongs. This option specifies the binary data (`*.bin`) generated by `svdatagen` for the file argument.

## 4. LOADER

- M *n*: A multitask program is created. The value *n* represents the number of tasks in a multitask program. Specify a value for *n* that is in the range from 2 to 128.
- m *n*[ *n...*]: This option specifies multi-entry loading of an IRSUB. For *n*, specify an entry name as an entry point. This option is valid only when +I is specified.
- Z: The sizes of the text, data, bss, and stack sections of the load module are output. This option does not register a program.  
If this option is specified together with the -P option, the command generates a linkage map list without loading.  
If this option is specified together with the -w *n* [ , *m*] option, the command makes it possible to determine the necessity for caller-side reloading (stack size enlargement) when the stack size used by the called IRSUB increases. In such cases, specify the values *n* and *m* so that they are equal to the values specified for the last registration.
- d: This option does not delete the load module file after storage in the backup file. The load module file is generated in PGM, SUB, or GLB within the site directory.
- s: Generates a stack use amount information file. The stack use amount information file is generated in the PGM/SUB directory within the site directory.
- l *lib*: For *lib*, specify the library to be linked. libcpms.lib and libsh4nbmdn.lib are automatically linked.
- P [*file*]: Outputs the linkage map list of the program. If *file* is not specified, the command generates a file in PGM, SUB, or GLB within the site directory with the resulting file name formed by adding .map to the end of the name of the program or subprogram to be loaded. For initial value data loading, the file name is formed by adding .map to the end of the sarea name of the beginning of the loaded data.  
When the environment variable LOADHR\_FORCE\_MAP=YES is set, a linkage map list is generated even when this option is not specified. If both the environment variable LOADHR\_FORCE\_MAP and this option are specified, both specifications are considered valid.
- o *obj*: For *obj*, specify the name of the program to be created. When the specified program name represents a subprogram, the specified name is used as the subprogram name. Make sure that the subprogram name is identical to the function name in the program.
- E *n*: This option achieves storage with a redundant byte count *n* taken into account at the time of program or subprogram linkage. This option is useful when the program or subprogram is likely to be replaced with a larger one in the future.
- r: This option checks whether a program or subprogram can replace the contents of a specified area. When specifying the storage address, ensure that it is identical to the program or subprogram address before replacement.

- `-w n [m]`: This option specifies the stack area size in bytes.  
 This option is mandatory for a program or subprogram.  
 For  $n$ , specify the size of the stack area to be used locally.  
 For  $m$ , specify the stack size to be allocated. If  $m$  is not specified, the resulting stack size will be determined by adding the local stack size  $n$  to the maximum stack size for use by the called IRSUB.  
 If the stack size  $m$  to be allocated is smaller than  $n$  plus the called maximum stack size of the called IRSUB, the command generates a warning message.  
 Ensure that  $n$  and  $m$  are in the range from 0 to 8388608 (0x800000) and are multiples of 8. If neither of the specified values is a multiple of 8, the command displays a warning message and raises the value to a multiple of 8 for processing.
- `-Xref`: Outputs the cross-reference information into the linkage map list of the program. This option is valid only when `-P` is specified.

### Library search path

The library search path of the loader (the library search sequence specified by the `-l` option) conforms to the input file search sequence for the optimization linkage editor in the shc compiler package.

The input file search sequence for the optimization linkage editor is as follows:

- (1) Current directory
  - (2) Directory specified by `HLNK_DIR` in the RPDP operating environment setup file
- Multiple paths can be specified for `HLNK_DIR` in the RPDP operating environment setup file. To specify two or more paths, separate them with semicolons.

### Notes

The system area is used as the stack area of the built-in subroutine. The stack size of the built-in subroutine must be 512 bytes or less.

Because IRSUBs and multitasks are re-entrant programs, they cannot have a BSS area. If an IRSUB or a multitask that has a BSS area is loaded, a warning message appears.

Task execution starts at the beginning of the program. It does not always start from main.

When loading the program, specify the object file of the main routine first.

Do not load an object compiled by `cchc` of the R700 to the S10VE.

### Stack size

When the program uses a stack area, specify its size.

## 4. LOADER

### System/user external reference check

The system cannot reference user information. The user can reference system subprograms only. The following table shows the combinations that can be referenced.

Referencing \ Referenced		Subprogram		Global (including CM)		Value	
		S	U	S	U	S	U
Program	S	Yes	–	Yes	–	Yes	–
	U	Yes	Yes	–	Yes	–	Yes
Subprogram	S	Yes	–	Yes	–	Yes	–
	U	Yes	Yes	–	Yes	–	Yes
Global (including CM)	S	Yes	–	Yes	–	Yes	–
	U	Yes	Yes	–	Yes	–	Yes

S: System U: User

Yes: Can be referenced –: Cannot be referenced

Note: When global data references a subprogram, an indirect link table number corresponding to its name is embedded in the global data. When global data references global data, an absolute address is embedded in the global data. When global data references a value, the value is embedded in the global data.

### Notes

Because IRSUBs and multitasks are re-entrant programs, they cannot have a BSS area. If they do, a warning message appears.

- If multiple global data items exist within the program to be loaded, the local label address solution will not be implemented. In this case, divide the global data into multiple files for loading.
- Task execution starts at the beginning of the program. It does not always start with main.
- The option combinations are as follows:

		-o obj	-a area	-w n	m	-S	-u site	-C n	-p n	-M n	-d	-Z	-P file	-E n	-r	-m n	-l	-Xref
Program	+P	R	R	R	O	O	O	O	O	–	O	O	O	O	O	–	O	O
IRSUB	+I	R	R	R	O	O	O	O	O	–	O	O	O	O	O	O	O	O
ULSUB	+U	R	R	R	O	O	O	O	O	–	O	O	O	O	O	–	O	O
Data	+D	–	–	–	–	O	O	–	–	–	–	O	O	–	–	–	O	–
Binary data	+B	–	–	–	–	O	O	–	–	–	–	–	–	–	–	–	–	–

R: Required O: Optional –: Cannot be specified

### Termination codes

The `svload` command returns the following termination codes:

0: Normal termination

Non-0: Abnormal termination

### Cross-reference information

Cross-reference information of symbols is output. An example of the cross-reference information output is as follows:

```

** Cross Reference List **

No      Unit Name  Global. Symbol  Location  External Information
(1)     (2)         (3)           (4)       (5)
0001    a
        SECTION=P
            _func                00000100
            _func1               00000116
            _main                 0000012c
            _g                     00000136
        SECTION=B
            _a                    00000190  0001(00000140:P)
                                   0002(00000178:P)
                                   0003(0000018C:P)
0002    b
        SECTION=P
            _func01              00000154  0001(00000148:P)
            _func02              00000166  0001(00000150:P)
0003    c
        SECTION=P
            _func03              00000184

```

(1) A unit number that identifies an object.

(2) The object name shown in the input specification order at the time of the link.

(3) The symbol name, which is output in ascending order for each section.

(4) The symbol location address.

(5) The address of the external symbol reference location. This address is output in the following format:

*unit-number(address-or-offset-in-section:section-name)*



## 4. LOADER

### Procedure for calculating stack size

The overall amount used for the stack area can be determined by calculating the stack usage amounts of the functions included in the program and considering function call relationships.

#### (1) Calculating the stack area used by a function

The size of the stack area used by a function can be determined from `frame size` in the object list generated by the compiler.

The following is a specific example.

#### ■ Source code

```
extern int h(char , int *, double);
int h(char a, register int *b, double c)
{
    char *d;

    d = &a;
    h(*d,b,c);
    {
        register int i;

        i = *d;
        return i;
    }
}
```

#### ■ Object list

SCT	OFFSET	CODE	C LABEL	INSTRUCTION	OPERAND	COMMENT
P	00000000		_h:			; function: h
						; <b><u>frame size=12</u></b>
	00000000	2FE6		MOV.L	R14,@-R15	
	00000002	4F22		STS.L	PR,@-R15	

In this example, the size of the stack area used by function `h` is 12 bytes, which is the value of `frame size` given under `COMMENT` in the object list.

(2) Calculating the overall stack size from function call relationships

The size of the stack area to be used can be calculated from function call relationships.

Figure 2-1 shows how to calculate the stack usage amount from a function call relationship.

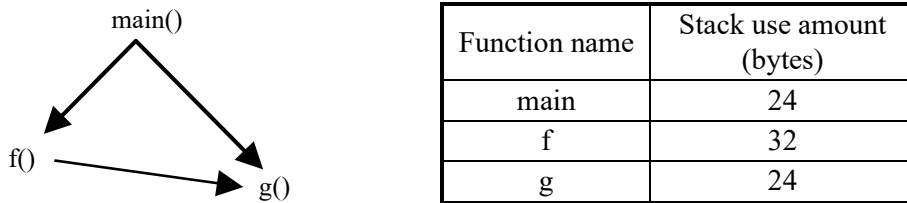


Figure 2-1 Function Call Relationship and Stack Usage Amount

When function g is called via function f as shown in the preceding figure, the stack area size is as indicated in Table 2-5.

Table 2-5 Stack Size Calculation Examples

Call route	Stack size (bytes)
main(24) -> f(32) -> g(24)	80
main(24) -> g(24)	48

As shown here, it is necessary to calculate the stack size for a function at the deepest call level and allocate at a minimum the stack area that is required for its maximum size.

When using a standard library function, it is also necessary to consider the stack size used by the library function. For details about the stack sizes used by standard library functions, see APPENDIX H LIST OF STACK SIZES FOR LIBRARY USE.

The stack size for a recursively called function must be calculated by multiplying the function stack size by the maximum recursive call count.

Even when no library function is used with a source program, an execution routine required to run the program might be linked. In this case, the stack size used by the execution routine must be considered. The stack size used by the execution routine can be confirmed by using the stack analysis tool described on the next page, which shows the procedure for determining the stack usage amount.

## 4. LOADER

### Procedure for determining the stack usage amount

When the `-s` option is specified when a program or subprogram is loaded, a stack usage amount information file can be generated.

The amount of overall stack usage by a program or subprogram can be determined by analyzing the stack usage amount information file generated by the loader by using the stack analysis tool supplied with the compiler package.

- Generating the stack usage amount information file

When the `-s` option is specified when a program or subprogram is loaded, the system generates the stack usage amount information file.

The stack usage amount information file is generated in the PGM/SUB directory within the site directory. The name of the file is formed by attaching `_.sni` to the program or subprogram name.

Example:

```
svload +P -o pgm01 -a tskarea -w 4096 pgm01.obj -s
```

In this example, the file `site-directory\PGM\pgm01_.sni` is generated.

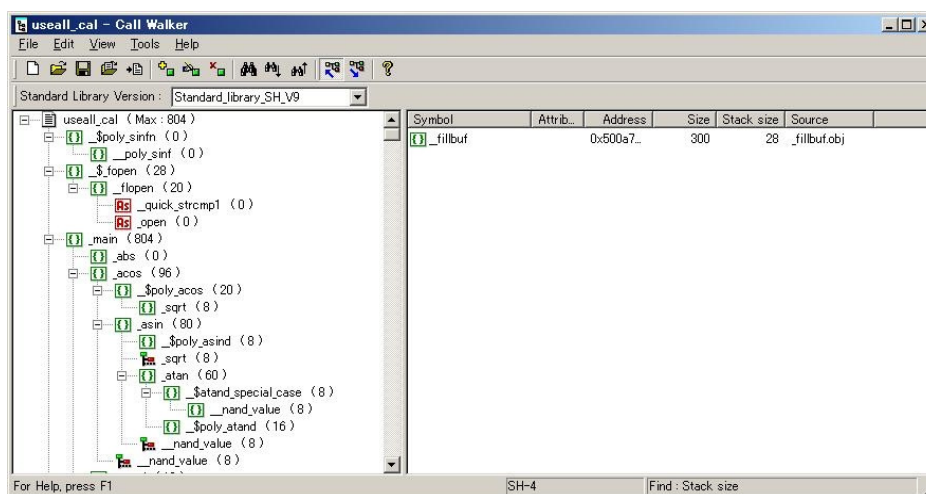
- Using the stack analysis tool

You can start the stack analysis tool to display the amount of stack usage by a program or subprogram by performing the following procedure.

For a detailed description of how to use the stack analysis tool, see the documentation supplied with the compiler package and the Help for the stack analysis tool.

- (1) To start the stack analysis tool: In Windows® 7, from the Windows® **Start** menu, select **Renesas, High-performance Embedded Workshop**, and then **Call Walker**. In Windows® 10, from the Windows® **Start** menu, select **Renesas** and then **Call Walker**.
- (2) From the **File** menu of the stack analysis tool, select **Import Stack file**. A dialog box appears. In the **File** field of the dialog box, specify the stack usage amount information file that was generated by the loader, and then click **Open**.

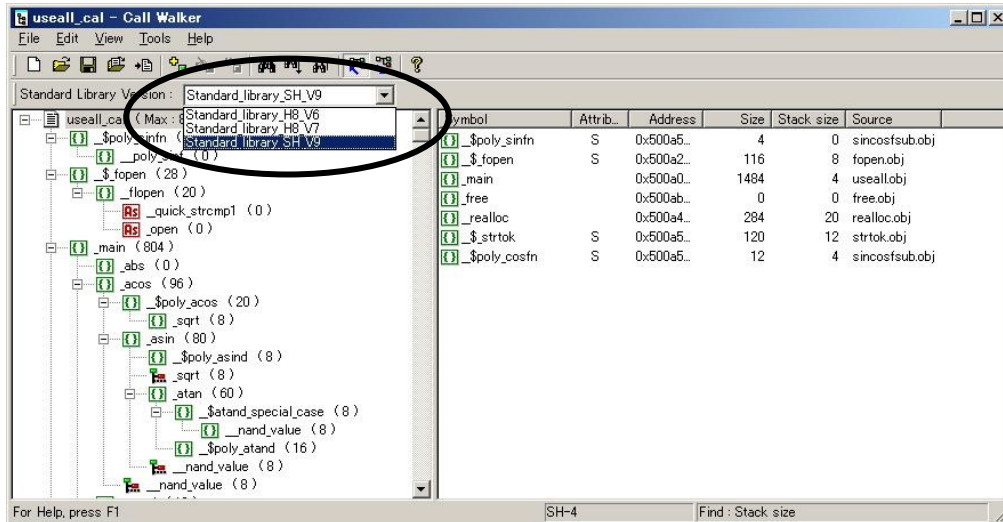
- Display example of the stack analysis tool



## Notes

From the **Standard Library Version** menu in the stack analysis tool, select **Standard\_Library\_SH\_V9**.

Do not use **Standard\_Library\_H8\_V6** or **Standard\_Library\_H8\_V7**, because these options are for different CPU types.



- Notes on analyzing the amount of stack usage by a loaded program or subprogram  
When used to calculate the amount of stack usage, the stack analysis tool indicates that the stack size of a program or subprogram written by the assembler is 0 bytes. Therefore, when you use the stack analysis tool to determine the amount of stack usage by a program or subprogram loaded by RPDP, note the following:

- memcpy() stack size

In a loaded program or subprogram, the CPMS library memcpy() function is linked instead of the C standard library memcpy() function. The stack analysis tool indicates that the stack size used by the CPMS library memcpy() function is 0. In reality, however, the CPMS library memcpy() function uses 28 bytes of the stack.

From the **Edit** menu of the stack analysis tool, use the **Modify** command to change the memcpy() function stack size to 28 bytes, and then recalculate the amount of stack usage. To determine whether the loaded program or subprogram uses memcpy(), search for “memcpy” by using the search function of the stack analysis tool.

## 4. LOADER

### ■ IRSUB stack size

When an IRSUB is called from a loaded program or subprogram, the stack analysis tool calculates that the amount of stack used by the IRSUB is 0 bytes. To calculate the stack usage amount including the stack size used by the called IRSUB, use the **Modify** command in the **Edit** menu of the stack analysis tool to change the stack size and recalculate the amount of stack use, in the same way as with memcopy().

The stack size used by an IRSUB can be determined by generating a stack usage amount information file when the IRSUB is loaded and by using the stack analysis tool to perform an analysis.

When determining the stack size used locally as specified for the loader (excluding the called IRSUB), you do not have to recalculate the IRSUB stack size.

### ■ Stack size of a program or subprogram written by the assembler

To determine the stack size used by a program or subprogram written by the assembler, use the preceding procedure. More specifically, use the **Modify** command in the **Edit** menu of the stack analysis tool to change the stack size and recalculate the amount of stack usage.

(3) Specifying the stack size by using `svload`

Specify the stack size by using `svload` as follows:

`-w n m`

*n*: Stack size to be used by the program or subprogram to be loaded.

*m*: Size including the stack size to be used by an IRSUB called from the program or subprogram to be loaded.

When this value is specified when a program is loaded, stack allocation takes place according to the specified size. If this value is not specified, stack allocation takes place according to the value *n* plus the value of the stack size used by the called IRSUB.

If the value specified for *m* is smaller than *n* plus the value of the stack size used by the IRSUB, the loader outputs the following warning message:

```
Warning: Stack size (name) = xxxx (zzzz) byte [Max refered (sname) size = yyyy byte] Err
```

*name*: Name of the program or subprogram to be loaded

*xxxx*: Size including the stack size to be used by the called IRSUB

*zzzz*: Stack size specified for *m*

*sname*: Name of the called IRSUB (having the maximum stack size)

*yyyy*: Stack size used by *sname*

To calculate the stack size used by a program or subprogram, follow the procedure described in (2) Calculating the overall stack size from function call relationships.

The following is an example specification of the stack size by using `svload` with reference to a call-related program, as shown in Figure 2-1 Function Call Relationship and Stack Usage Amount.

- When no IRSUB is used

main: Program main

f: ISUB

g: ISUB

In this case, specify the following:

```
svload +P -o main ..... -w 80 4096 .....
```

Because the maximum stack size used by the program to be loaded is 80 bytes, specify 80 for *n*.

For *m*, specify the size of the stack to be allocated. If *m* is not specified, the stack size is set to 80 bytes. To allow for an increase in the stack size for program modification or for the necessity of IRSUB calls, specify a stack size that is larger than the minimum required.

The stack/BSS of the program is located on a page that is separate from the text/data page.

Therefore, even if the specified stack size is larger than the minimum required, the program size does not increase unless the 4096-byte limit is exceeded by the BSS size plus the stack size.

## 4. LOADER

- When an IRSUB is used

main: Program main

f: ISUB

g: IRSUB

In this case, specify the following:

```
svload +I -o g ..... -w 24 .....  
svload +P -o main ..... -w 56 4096 .....
```

The IRSUB and program must be loaded separately. The stack size must be specified for both the IRSUB and the program.

- For IRSUB loading

Because the stack size used by the IRSUB ( $g$ ) is 24 bytes, specify 24 bytes as the stack size. We recommend not specifying a value for  $m$ . If a value is specified for  $m$ , it is used as the IRSUB stack size that is calculated by the IRSUB caller side.

- For program loading

Because the stack size used by the program is 56 bytes, specify 56 as the value of  $n$ .

For  $m$ , specify a value that is at least 80 bytes, the value that is obtained by adding the value 56 to the stack size used by the IRSUB ( $g$ ), which is 24. As in the case where no IRSUB is used, specify a value for  $m$  that is greater than the minimum required to allow for program modifications.

If the value specified for  $m$  is smaller than 80 bytes, the loader outputs the following warning message:

```
Warning: Stack size (main) = 80 (m) byte [Max refered (g) size = 24 byte] Err
```

If no value is specified for  $m$ , the loader allocates 80 bytes of stack by adding the stack size used by the IRSUB ( $g$ ), which is 24.

- When an IRSUB is used from another IRSUB

main: Program main

f: IRSUB

g: IRSUB

In this case, specify the following:

```
svload +I -o g ..... -w 24 .....
svload +I -o f ..... -w 32 .....
svload +P -o main ..... -w 24 4096 .....
```

- For IRSUB (g) loading

Because the stack size used by the IRSUB (g) is 24 bytes, specify 24 bytes as the stack size. We recommend not specifying a value for *m*. If a value is specified for *m*, it is used as the IRSUB stack size that is calculated by the IRSUB caller side.

- For IRSUB (f) loading

Because the stack size used by the IRSUB (f) is 32 bytes, specify 32 bytes as the stack size. If no value is specified for *m*, the IRSUB (f) stack size calculated by the IRSUB caller side is 56 bytes, which is determined by adding the stack size of the IRSUB (g) that is called by the IRSUB (f). If a value is specified for *m*, it is used as the IRSUB (f) stack size that is calculated by the IRSUB caller side. We recommend not specifying a value for *m*.

If the value specified for *m* is smaller than 56 bytes, the loader outputs the following warning message:

```
Warning: Stack size (f) = 56 (m) byte [Max refered (g) size = 24 byte] Err
```

- For program loading

Because the stack size used by the program is 24 bytes, specify 24 as the value *n*.

For *m*, specify a value that is at least 80 bytes, a value that is obtained by adding the value 24 to the stack size used by the called IRSUB (f) having the maximum stack size, which is 56. As in the case where no IRSUB is used, specify a value for *m* that is greater than the minimum required to allow for program modifications.

If the value specified for *m* is smaller than 80 bytes, the loader outputs the following warning message:

```
Warning: Stack size (main) = 80 (m) byte [Max refered (f) size = 56 byte] Err
```

If no value is specified for *m*, the loader allocates 80 bytes of stack by adding the stack size used by the IRSUB (f), which is 56.



## 4. LOADER

### Library integrity check

When compilation is performed with the `shc -denormalization=off` and `-round=zero` options specified, you must specify the library `libsh4nbmzz.lib` (`-lsh4nbmzz`) at the time of loading. If `libsh4nbmzz.lib` is not specified, the loader outputs the following error message:

```
svload : Error : Undefined symbols
svload :          _use_libsh4nbmzz
```

Similarly, if `libsh4nbmzz.lib` is specified in situations where the `shc -denormalization off` and `-round=zero` options are not specified for compilation, the loader outputs the following error message:

```
svload : Error : Undefined symbols
svload :          _use_libsh4nbmdn
```

If both objects are mixed to specify `-lsh4nbmzz` as well as `-lsh4nbmdn`, the loader outputs the following error message:

```
svload : rpdpload: Inconsistent object was mixed (NO:2004-25)
```

**Name**

svdload

**Syntax**

svdload *pname* [*option*]

**Description**

The `svdload` command deletes the program or subprogram registered by the `svload` command from the development environment. However, the backup file is not zero-cleared.

**Arguments**

*pname*: Specify the name of the program or subprogram to be deleted.

**Options**

- S: This option specifies the system processing mode. When this option is omitted, the predefined default access rights (defined by the environment variable `RSUTYP`) are selected.
- u *site*: For *site*, specify the name of the site to be handled by the loader. When this option is omitted, the loader processes the site that is set in the environment variable `RSSITE`.
- +P: The program (task) is deleted.
- +I: The indirect link subroutine (IRSUB) is deleted.
- +U: The built-in subroutine (ULSUB) is deleted.

**Termination codes**

The `svdload` command returns the following termination codes:

0: Normal termination

Non-0: Abnormal termination

## 4. LOADER

### Name

svcomp

### Syntax

svcomp [*option*] *file*...

### Description

The `svcomp` command compares the contents of the backup files for programs, subprograms, and data registered by the loader with the contents of the load module. The command then displays the results.

### Arguments

`file`: Specify one or more object files or libraries to be combined.

### Options

- S: This option specifies the system processing mode. When this option is omitted, the predefined default access rights (defined by the environment variable `RSUTYP`) are selected.
- u *site*: For *site*, specify the name of the site to be handled by the loader. When this option is omitted, the loader processes the site that is set in the environment variable `RSSITE`.
- C *n*: This option has no meaning when used with `svcomp`.
- p *n*: This option has no meaning when used with `svcomp`.
- a *area*: This option has no meaning when used with `svcomp`.
- +P: Specifies that the item loaded as a program (task) is to be used for comparison.
- +I: Specifies that the item loaded as an indirect link subroutine (IRSUB) is to be used for comparison.
- +U: Specifies that the item loaded as a built-in subroutine (ULSUB) is to be used for comparison.
- +D: Specifies that the item loaded as global or CM data is to be used for comparison. The data type depends on the attribute of the split area to which the data belongs.
- +B: Compares loaded global or CM data with binary data generated by the data generator (`svdatagen`). This option specifies the binary data (`*.bin`) generated by `svdatagen` for the file argument.
- M *n*: This option has no meaning when used with `svcomp`.
- m *n*[ *n...*]: This option has no meaning when used with `svcomp`.
- Z: This option has no meaning when used with `svcomp`.
- d: This option has no meaning when used with `svcomp`.
- l *lib*: Specifies the library (*lib*) to be linked. Note that `libcpms.lib` and `libsh4nbmdn.lib` are automatically linked.
- P [*file*]: This option has no meaning when used with `svcomp`.
- o *obj*: For *obj*, specify the name of the load module to be compared with.
- E *n*: This option has no meaning when used with `svcomp`.
- r: This option has no meaning when used with `svcomp`.
- w *n* [*m*]: This option has no meaning when used with `svcomp`.
- Xref: This option has no meaning when used with `svcomp`.

### Library search path

The library search path of the loader (the library search sequence specified by the `-l` option) conforms to the input file search sequence for the optimization linkage editor in the `shc` compiler package.

The input file search sequence for the optimization linkage editor is as follows:

(1) Current directory

(2) Directory specified by `HLNK_DIR` in the RPDP operating environment setup file

Multiple paths can be specified for `HLNK_DIR` in the RPDP operating environment setup file.

To specify two or more paths, separate them with semicolons.

### Termination codes

The `svcomp` command returns the following termination codes:

0: Not different.

1: Different.

101: A command option was specified.

Other codes: The comparison failed.

## 4. LOADER

### svcomp results display

The `svcomp` command outputs the following messages depending on whether differences were found during the comparison.

(1) If no differences were found during the comparison:

If no differences were found in the comparison, the `svcomp` command outputs the following message:

```
compare OK (type = X name = xxx)
```

*X*: Indicates the type of the resource to be compared.

pgm: Program.

irsub: IRSUB

ulsub: Built-in program

data: Data (GLB or CM)

*xxx*: Indicates the name of the resource to be compared.

This reads `data` when data is to be used for comparison.

(2) If differences were found during the comparison:

If differences were found during the comparison, the `svcomp` command outputs the following message:

### Display format

The display produced by the `svcomp` command consists of the following fields:

(1) Header

(2) Detailed information

(3) Footer

Figure 2-2 shows the format for a program or subprogram, while Figure 2-3 shows the format for data. The italicized character strings vary depending on the execution environment.

```

** compare list **                                     date

site name = site-name          (1)

type = type name = name

< text >          (2)
  text size      new = xxxxxxx    old = xxxxxxx    (a)
  loc = xxxxxxx  new = xxxxxxx    old = xxxxxxx    (b)
                                     :
                                     :

< data >
  data size      new = xxxxxxx    old = xxxxxxx    (c)
  loc = xxxxxxx  new = xxxxxxx    old = xxxxxxx    (d)
                                     :
                                     :

< bss >
  bss size       new = xxxxxxx    old = xxxxxxx    (e)

** compare list output end ** (3)

```

Figure 2-2 Format of the Display Produced by svcomp for a Program or Subprogram

The following is an explanation of the fields in the figure.

(1) Header field

*date*: Time at which the `svcomp` command started  
*site-name*: Name of the processed site  
*type*: Type of the program compared  
 pgm: Program used as a task  
 irsub: Indirect link subroutine  
 ulsub: Built-in subprogram  
*name*: Name of the program compared

(2) Detailed information field

This field displays the results of comparison between the old sizes of the text, data, and bss sections and their new sizes and between the old contents of the text and data sections and their new contents. When the new size differs from the old size, the comparison result is displayed relative to the smaller size.

(a), (c), and (e): The results of comparison between the old sizes of the text, data, and bss sections and their new sizes are displayed in hexadecimal. If no differences exist between them, nothing is displayed.

(b), (d): The results of comparison between the old contents of the text and data sections and their new contents are displayed in hexadecimal. If no differences exist between them, nothing is displayed. The hexadecimal data displayed immediately after `loc=` is the address where a difference was found. The address is relative to the beginning of the text or data.

(3) Footer field

This field indicates that the comparison is complete.

## 4. LOADER

```
** compare list ** date

site name = site-name (1)

type = data name = data

<data-name> (2)
  data size new = xxxxxxxx old = xxxxxxxx (a)
  loc = xxxxxxxx new = xxxxxxxx old = xxxxxxxx (b)
                :
                :

** compare list output end ** (3)
```

Figure 2-3 Format of the Display Produced by svcomp for GLB or CM

The following is an explanation of the fields in the figure.

(1) Header field

*date*: Time at which the svcomp command started

*site-name*: Name of the processed site

(2) Detailed information field

This field displays the results of comparison between the old size of GLB or CM and its new size and between the old contents of the text and data sections and their new contents. When the new size differs from the old size, the comparison result is displayed relative to the smaller size.

(a): The results of comparison between the old size of GLB and CM and its new size are displayed in hexadecimal. If no differences exist between them, nothing is displayed.

(b): The results of comparison between the old contents of GLB and CM and its new contents are displayed in hexadecimal. If no differences exist between them, nothing is displayed.

The hexadecimal data displayed immediately after loc= is the address where a difference was found. The address is relative to the beginning of the text or data.

(3) Footer field

This field indicates that the comparison is complete.

## CHAPTER 5 BUILDER

### Name

svctask

### Syntax

svctask *pname tname tn* [*option*]

### Description

The `svctask` command creates a task by using the load module loaded by the loader as a resource.

### Arguments

*pname*: Program name of the load module to be used as a resource for task creation.

*tname*: Name of the task to be created.

*tn*: Task number from 1 to 224 for a user task or from 225 to 300 for a system task. If the specified task number is already in use, an error occurs.

### Options

- S: Specifies that the system processing mode is to be used. If this option is omitted, the predefined default access rights (defined by the environment variable `RSUTYP`) are selected.
- u *site*: For *site*, specify the name of the site to be handled by the builder. If this option is omitted, the builder processes the site that is set in the environment variable `RSSITE`.
- l *lvl*: For *lvl*, specify the execution level at the initial startup of the task. The level must be from 4 to 27 for a user task or from 0 to 31 for a system task. If this option is omitted, *lvl* is assumed to be 27 for a user task or 0 for a system task.
- r *n*: Specifies the stack area number to use when generating multiple tasks from a program that serves as a resource. This value cannot exceed the number of multitasks (`-M n`), which is specified by the load command. If this option is omitted, the system assumes that the minimum unused stack area number is specified.

### Notes

- User tasks are tasks for which a TN value from 1 to 224 is registered. System tasks are tasks for which a TN value from 225 to 300 is registered.
- For registrations in which `RSUTYP=s` is specified, a system task can be created.
- For registrations in which `RSUTYP=u` is specified, specifying the `-S` option allows the creation of a system task.
- If `RSUTYP=u` is specified and the `-S` option is omitted, no system task can be created. The table on the following page shows the available combinations of task types and options.



## 5. BUILDER

Parameter		Task type	
		Single task	Multi-task
<i>pname</i>		R	R
<i>tname</i>		R	R
<i>tn</i>		R	R
Options	<i>-u site</i>	Y	Y
	<i>-l lvl</i>	Y	Y
	<i>-S</i>	Y	Y
	<i>-r n</i>	-	R

R: Required Y: Can be specified -: Cannot be specified

The following table shows the relationships between user types and program owner types:

User type	Program owner type	
	System	User
System	Y	Y
User	-	Y

Y: Task can be created -: Task cannot be created

### Termination codes

The `svctask` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

**Name**

svdtask

**Syntax**svdtask *tname* [*option*]**Description**

The `svdtask` command deletes task that were created by `svctask`.

**Arguments**

*tname*: Name of the task to be deleted

**Options**

- S: Specifies that the task to be deleted is a system task. If this option is omitted, the default (set in the `RSUTYP` environment variable in advance) is used.
- u *site*: Specifies the name of the site to be processed by the builder. If this option is omitted, the builder processes the site that is set in the environment variable `RSSITE`.

**Notes**

- The following table shows the relationships between user types and owner types for tasks to be deleted.

User type	Deletion task owner type	
	System	User
System	Y	Y
User	-	Y

Y: Task can be deleted    -: Task cannot be deleted

**Termination codes**

The `svdtask` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

## 5. BUILDER

### Name

`svbuild` (registration of indirect link resident subprograms)

### Syntax

`svbuild name -ir -e irno [option]`

### Description

The `svbuild` command registers indirect link resident subprograms.

### Arguments

*name*: Indirect link resident subprogram name (entry name when multiple entries are loaded).

`-ir -e irno`: For *irno*, specify the registration number of an indirect link resident subprogram.

A value in the range from 1 to 7,935 can be specified for *irno*.

### Options

`-u site`: For *site*, specify the name of a site to be handled by the builder. If this option is omitted, the builder processes the site that is set in the environment variable `RSSITE`.

Parameter		Type	Indirect link resident subprogram
<i>name -ir -e irno</i>			R
Option	<code>-u site</code>		Y

R: Required Y: Can be specified

### Termination codes

The `svbuild` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

**Name**

`svbuild` (registration of built-in subroutines)

**Syntax**

`svbuild name point en -ul [option]`

**Description**

The `svbuild` command registers built-in subroutines.

**Arguments**

*name*: Built-in subroutine name.

*point en -ul*: For *point*, specify the entry point of the built-in subroutine by using one of the following character strings. For *en*, specify the entry number in the range from 1 to 4. Note that entry number 1 is reserved for the OS, and entry number 2 is reserved for the NXACP.

CPES: CPES built-in subroutine

IES: IES built-in subroutine

EAS: EAS built-in subroutine

INS: INS built-in subroutine

EXS: EXS built-in subroutine

ABS: ABS built-in subroutine

PCKS: PCKS built-in subroutine

MODES: MODES built-in subroutine

WDTES: WDTES built-in subroutine

XEAS: Subroutine that links at an XPU error

**Options**

`-u site`: For *site*, specify the name of the site to be handled by the builder. If this option is omitted, the builder processes the site that is set in the environment variable `RSSITE`.

## 5. BUILDER

### Notes

Although the same subprogram can be incorporated at multiple locations (specified by *point*), it cannot be incorporated for multiple entry numbers (*en*).

The following table shows the available option combinations.

Parameter		Type	Built-in subroutine
<i>name point en -u1</i>			R
Option	<i>-u site</i>		Y

R: Required Y: Can be specified

### Termination codes

The `svbuild` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

**Name**

svdbuild (deletion of indirect link resident subprograms)

**Syntax**

svdbuild *name* -ir [*option*]

**Description**

The svdbuild command deletes indirect link resident subprograms.

**Arguments**

*name*: Indirect link resident subprogram name (if multiple entries are loaded, the entry name)

-ir: The specified indirect link resident subprogram is deleted.

**Options**

-u *site*: For *site*, specify the name of the site to be handled by the builder. If this option is omitted, the builder processes the site that is set in the environment variable RSSITE.

Parameter \ Type		Indirect link resident subprogram
<i>name</i> -ir		R
Option	-u <i>site</i>	Y

R: Required Y: Can be specified

**Termination codes**

The svdbuild command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

## 5. BUILDER

### Name

`svdbuild` (deletion of built-in subroutines)

### Syntax

`svdbuild name point -ul [option]`

### Description

The `svdbuild` command deletes built-in subroutines.

### Arguments

*name*: Built-in subroutine name.

*point* `-ul`: The specified built-in subroutine is deleted.

For this argument, specify one of the following character strings as the entry point of the built-in subroutine:

CPES: CPES built-in subroutine

IES: IES built-in subroutine

EAS: EAS built-in subroutine

INS: INS built-in subroutine

EXS: EXS built-in subroutine

ABS: ABS built-in subroutine

PCKS: PCKS built-in subroutine

MODES: MODES built-in subroutine

WDTES: WDTES built-in subroutine

XEAS: Subroutine that links at an XPU error

### Options

`-u site`: For *site*, specify the name of the site to be handled by the builder. If this option is omitted, the builder processes the site that is set in the environment variable `RSITE`.

**Notes**

The following table shows the available option combinations.

Parameter \ Type		Built-in subroutine
<i>name point -ul</i>		R
Option	<i>-u site</i>	Y

R: Required Y: Can be specified

**Termination codes**

The `svdbuild` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination



## 5. BUILDER

### Name

svirglb

### Syntax

svirglb *idxnum name* [*option*]

### Description

The `svirglb` command registers or deletes secondary partition areas allocated by `svdfs` as indirect link global areas.

### Arguments

- idxnum*: Specify the registration number of the indirect link global table in the range from 1 to 7,935.
- name*: Specify the indirect link global area name. (When the `-s` or `-a` option is omitted, specify the name of the secondary partition area already allocated by `svdfs` as the indirect link global area name.)

### Options

- `-u site`: For *site*, specify the name of the site to be handled by the builder. If this option is omitted, the builder processes the site that is set in the environment variable `RSSITE`.
- `-s name`: When using the address of the indirect link global area plus an offset as the address to be stored in the indirect link global table, specify the secondary partition area name for *name*.
- `-o offset`: When using the address of the indirect link global area plus an offset as the address to be stored in the indirect link global table, specify the offset for *offset* as a decimal or hexadecimal number. A number starting with `0x` is handled as a hexadecimal number.
- `-a adr`: Specify the address to be stored in the indirect global table as an absolute address in decimal or hexadecimal. A number starting with `0x` is handled as a hexadecimal number.
- `-d`: The specified registration number is deleted from the indirect link global table.

## Notes

- The `-s` and `-o` options must be specified together if either option is specified.
- When the `-s`, `-o`, and `-a` options are specified together, an error occurs.
- The following table shows the available option combinations:

Parameter \ Type		Indirect link global	
		Registration	Deletion
<i>idxnum</i>		R	R
<i>name</i>		R	R
Option	<code>-u site</code>	Y	Y
	<code>-s name</code>	Y	Y
	<code>-o offset</code>	Y	Y
	<code>-a</code>	Y	Y
	<code>-d</code>	-	R

R: Required Y: Can be specified -: Cannot be specified

- Only addresses in the CM and GLB spaces can be specified in the `-a` option.
- If the specified offset is outside the split area that includes the secondary partition area specified by the `-s` option, an error occurs.

## Termination codes

The `svirglb` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

**CHAPTER 6 MANAGEMENT TOOL****Name**

svmap

**Syntax**• **Name**

```
svmap name [name...] [-site sitename] -a [-e] [-f] [-CON] [-osrsv]
      [-site sitename] -e      [-f] [-CON] [-osrsv]
      [-site sitename] -p      [-f] [-CON] [-osrsv]
      [-site sitename] -s      [-f] [-CON] [-osrsv]
      [-site sitename] -t      [-f] [-CON] [-osrsv]
      [-site sitename] -g      [-f] [-CON] [-osrsv]
      [-site sitename] -v      [-f]      [-osrsv]
```

• **Number**

```
svmap num [num...] [-site sitename] -irs [-f] [-CON] [-osrsv]
      [-site sitename] -irg [-f] [-CON] [-osrsv]
      [-site sitename] -uls [-f] [-CON] [-osrsv]
```

• **Entire display**

```
svmap [-site sitename] [-G] [-a] [-e] [+n] [+a]      [-f] [-CON] [+gn gname] [-osrsv]
      [-site sitename] [-a] [-e]      [+n] [+a]      [-f] [-CON]      [-osrsv]
      [-site sitename] [-e]      [+n] [+a]      [-f] [-CON]      [-osrsv]
      [-site sitename] [-p]      [+n]      [-f] [-CON]      [-osrsv]
      [-site sitename] [-s]      [+n]      [+e] [-f] [-CON] [-en] [-osrsv]
      [-site sitename] [-t]      [+n]      [+e] [-f] [-CON]      [-osrsv]
      [-site sitename] [-g]      [+n]      [+e] [-f] [-CON] [-en] [-osrsv]
      [-site sitename] [-v]      [+n]      [-f]      [-en] [-osrsv]
      [-site sitename] [-irs]      [+n]      [+e] [-f] [-CON] [-en] [-osrsv]
      [-site sitename] [-irg]      [+n]      [+e] [-f] [-CON] [-en] [-osrsv]
      [-site sitename] [-uls]      [+n]      [+e] [-f] [-CON]      [-osrsv]
      [-site sitename] [-en]      [-f] [-CON]      [-osrsv]
      [-site sitename] [-fm]      [-f] [-CON]      [-osrsv]
```

**Description**

The `svmap` command outputs information about resources managed by RPDP. The information about resources displays system information and user information regardless of the `RSUTYP` settings.

## Arguments

*name*: Specifies the name of a resource to be displayed. At this time, the option that indicates the name type cannot be omitted. You can specify multiple names.

*num*: Specifies the number of a resource to be displayed. You can specify multiple numbers. At this time, the option that indicates the number type cannot be omitted.

When `-irs` or `-irg` is specified, specify the number of the IRSUB or IRGLB that you want to display for *num*.

When `-uls` is specified, specify the built-in point and entry number for *num*. Specify the built-in point and entry number in *pnt*, *typ*, and *ent* format (example: `eas, os, 1`).

## Options

- `-site sitename`: Specifies the name of the site to which the map information is to be output. If this option is omitted, `svmap` processes the site that is set in the environment variable `RSSITE`.
- `-G`: Outputs information about a global area. Combining this option with the `-a` and `-e` options makes it possible to output the hierarchical map of global areas, split areas, and secondary partition areas.
- `-a`: Outputs information about a split area. Combining this option with the `-e` option makes it possible to output the hierarchical map of split areas and secondary partition areas.
- `-e`: Outputs information about a secondary partition area (an sarea of a GLB or CM, program, or subprogram).
- `-p`: Outputs information about programs.
- `-s`: Outputs information about subprograms (IRSUBs and ULSUBs).
- `-t`: Outputs information about tasks.
- `-g`: Outputs global (GLB and CM) information.
- `-v`: Outputs VAL information.
- `-irs`: Outputs IRSUB entry information.
- `-irg`: Outputs IRGLB entry information.
- `-uls`: Outputs ULSUB entry information.
- `-en`: Outputs information about the number of IRSUB, GLB (including CM), and VAL registrations.  
Specifying this option together with `-s`, `-g`, `-v`, `-irs`, and `-irg` displays only information about the number of registrations of the specified items.
- `-f`: Displays detailed information.
- `-fm`: Outputs information about the available space in the physical memory.
- `-CON`: Outputs the S10VE memory map. If this option is not specified, the map information about resources managed by the development machine will be output.
- `-osrsv`: Displays resources reserved by the OS. If this option is omitted, OS-reserved resources are not displayed. Names beginning with `osreserve` are handled as OS-reserved resources. Do not use such names.
- `-help`: Displays a list that describes the command startup format.
- `+a`: Outputs the results after sorting them by address.

## 6. MANAGEMENT TOOL

+n: Outputs the results after sorting them by name.

+e: Outputs the results after sorting them by entry number.

+gn *gname*: Specifies the global area name when only information about a specific global area is output during output of a hierarchical map for a garea, an area, or an sarea. Specify a name without a dollar sign (\$) (such as TASK and IRSUB) as the global area name.

Table 2-6 shows the available option combinations that specify the type of resources to be displayed (-G, -a, -e, -p, -s, -t, -g, -v, -irs, -irg, and -uls) and options that specify the result output sequence (+a, +n, and +e), as well as the default output sequence when the +a, +n, and +e options are omitted.

If all of the options that show the type of resources to be displayed (-G, -a, -e, -p, -s, -t, -g, -v, -irs, -irg, and -uls) and -en, -fm, and -help are omitted, the command assumes that the following options are specified:

```
-G -a -e -t -v -irs -irg -uls
```

### Termination codes

0: Normal termination

Other than 0: Abnormal termination

Table 2-6 Available Combinations of Output Resources and Output Sequence, and Default Output Sequences

Output resource Output sequence	-G	-a	-e	-p	-s	-t	-g	-v	-irs	-irg	-uls
+a	Y	Y	Y	-	-	-	-	-	-	-	-
+n	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
+e	-	-	-	-	Y	Y	Y	-	Y	Y	Y
None (default)	+a	+a	+a	+n	+n	+e	+n	+n	+e	+e	+e

Y: Can be combined -: Cannot be combined

+a: Address order (default) +n: Name order (default) +e: Numerical order (default)

**Name**

svadm

**Syntax**

svadm [*addr*] [*option*]

**Description**

The `svadm` command outputs the name and other information about the resource (global or IRSUB) registered at a specified logical address (for details, see *Display formats*).

When no logical address is specified, the command prompts the user to enter it interactively, and displays names and other information.

**Arguments**

*addr*: For *addr*, specify a logical address in the range from 0x3000000 to 0x7bffffff (for the task space, GLB space, subprogram space, CM space, or LADDER, USRFUNC, or HI-FLOW space). If this argument is omitted, the user is prompted to enter a logical address interactively.

**Options**

- u *site*: For *site*, specify the name of the site for which information on the area indicated by the specified logical address is to be displayed. If this option is omitted, `svadm` processes the site that is set in the environment variable `RSSITE`.
- o *file*: For *file*, specify the name of the file to which the operation result is to be output.

**Termination codes**

The `svadm` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

## Operation

When an address is specified in the \_\_\_\_\_

```
#svadm addr
```

```
Information is displayed.
```

```
#
```

When no address is specified in the \_\_\_\_\_

```
#svadm
```

```
++ address information display start --> site(XXXXX) ++
```

```
addr : addr
```

```
Information is displayed.
```

```
addr : q
```

```
++ address information display end ++
```

```
#
```

## Explanation

The underlined text indicates data to be entered by the user.

XXXXX: Site name

addr: Specify the address that indicates the area from which the user wants to acquire information.

q: The command is terminated.

## Display formats

The following three display formats are supported.

- Format when a resource is registered at the specified address:

```
name = NAME type = TYPE raddr = XXXXXXXX
```

*NAME*: Resource name (sarea name, program name, or subprogram name)

*TYPE*: Resource type

task (TEXT) : Text part of a task

task (DATA) : Data part of a task

task (BSS) : BSS part of a task

task (STACK) : Stack of a task

sub: Subprogram

data: sarea of GLB

CM: sarea of CM area

*XXXXXXX*: Offset from the beginning of the resource. For tasks, the offset is the relative position from the beginning of the TEXT, DATA, BSS, or STACK area. For subprograms, the offset is the relative position from the beginning of each subprogram. For GLB and CM, the offset is the relative position from the beginning of each sarea.

- Format when no resource is registered at the specified address:

```
lspace = SPACE external name is not defined
```

*SPACE*: GAREA name of the specified address

- Format when the specified address is out of the GAREA range

```
address error (0xXXXXXXXX)
```

*XXXXXXXX*: Specified address

This format is also shown when the address of the LADDER, USRFUNC, or HIFLOW space is specified, because no area is defined in these cases.



## 6. MANAGEMENT TOOL

### Name

svsitecntl

### Syntax

svsitecntl [*option*]

### Description

The `svsitecntl` command controls and displays the status of a specified site. When the `-rsrcv` option is specified, recovery of the specified site is inhibited.

### Options

`-query [-sort]`: Lists the sites registered with the development machine.

- **Displaying by `-query`**

When the `-query` option is specified, S10VE sites are displayed.

The following shows the output format.

When `-sort` is specified, sites are sorted and shown in alphabetical order for each model.

```
+++++++ S10VE site +++++++
0001cp is active
0001hp is active
0002cp is active
0002hp is active
1000cp is active
1000hp is active
```

`-rsrcv site`: Makes commands available without using RPDV recovery processing during suspension by the `ld` subcommand of the `svdebug` command.

However, the `ld` subcommand of `svdebug` cannot be used.

(For details, see APPENDIX E NOTES ON USING RPDV.)

### Termination codes

The `svsitecntl` command returns the following termination codes:

0: Normal termination

Other than 0: Abnormal termination

## CHAPTER 7 STARTUP AND PU CONTROL

### Name

`svrpl` - Performs remote loading.

### Syntax

```
svrpl [{-u site | -U unit}{-s}] [-all] [-r] [{-time|-notime}] [-ROMSV | -NOROMSV]
[-setpcsno]
```

### Description

The `svrpl` command stops the specified site (PU) and transfers backup file data to the main memory of the specified site (PU) on the S10VE to start the specified site. The following options can be specified.

### Options

- `-u site`: Specifies the site to be loaded. If this option is omitted, `svrpl` processes the site that is set in the environment variable `RSSITE`.  
Specify the CPU name (the same name as the CP site name).
- `-U unit`: Specifies the unit when all sites are to be loaded.  
The `-u` and `-U` options are mutually exclusive.
- `-all`: Downloads all backup files.

The following table lists the files that are downloaded depending on the specified options.

Option \ File	OS	TASK, SUB, GLB	CM
<code>-all</code>	Y	Y	Y
None	Y	Y	Y

Y: Downloaded    -: Not downloaded

- `-r`: Does not start a specified site (PU) after loading is complete.
- `-s`: Stops a specified site without prompting the user whether to stop the specified site (PU).  
Specify this option together with the `-u` option.  
If the `-s` option is omitted, `svrpl` prompts the operator to confirm whether to stop the site at the time of a download.
- `-time`: Sets the time when the specified site is CPU.
- `-notime`: Does not set the time when the specified site is CPU.  
Specify this option if you do not want to set the time.

## 7. STARTUP AND PU CONTROL

- ROMSV: Saves remotely loaded SDRAM data in the ROM (NAND flash).
- NOROMSV: Does not save remotely loaded SDRAM data in the ROM (NAND flash).  
If neither -ROMSV nor -NOROMSV is specified, -ROMSV is assumed.
- setpcsno: Enables the function for setting the site name and PC number.

The `svrpl` command first displays the states of all PUs mounted in the unit to which the specified site belongs. The command then checks whether the site (PU) to be downloaded can be stopped. Typing `yes` when prompted here stops the specified site (PU) and starts downloading. Typing `no` terminates the command with no action.

### Notes

- The `svrpl` command does not download LADDER or HI-FLOW programs.
- Only the S10VE connected by the connection PCs change function of the BASE SYSTEM/S10VE can be loaded by the `svrpl` command.
- The `svrpl` command cannot start an instance of S10VE for which the OS has never been downloaded via the CPMS download of the BASE SYSTEM/S10VE. Be sure to load the OS first via the CPMS download of the BASE SYSTEM/S10VE.
- If an HP site name is specified for the site name, an error message appears, and the command terminates.
- If an error occurs during loading, the specified site (PU) is stopped and the command terminates.
- If the specified site (PU) is not to be started after a download (the `-r` option is specified), `svrpl` ignores the `-time` option if it is specified, and does not set the time.
- When the state control (`svcpuctl`) command sends a RUN request to the CPU, the time can be set if the `-time` option is specified.
- When the `-time` option is not specified for downloading to the CPU, the time is not set.

### Site name and PC number setting function

When the SYSCB table of CPMS is loaded with the `-setpcsno` option specified, the site name (`sy_site`) is rewritten with the loaded site name. This makes it possible to rewrite the site name of the real machine with the PC number if the BASE SYSTEM/S10VE loads the site of each generated PC number, by copying the base site to the real machine.

When the SYSCB table of CPMS is loaded with the `-setpcsno` option omitted, the site name (`sy_site`) is not loaded. In this case, the site name of the real machine remains the site name that was loaded (by the CPMS download of the BASE SYSTEM/S10VE) before it is loaded by the `svrpl` command.

When the `-setpcsno` option is specified and the site for a PC number is loaded, the PC number is written to the MRAM. The site name is recognized as a site for a PC number when the CP site name is a 4-digit decimal number (0000 to 9998) + `cp`. When the CP site name is a number other than a 4-digit decimal number (0000 to 9998) + `cp` or 9999`cp`, it is not recognized as a site for a PC number, and no PC number is written to the MRAM.

When the `-setpcsno` option is omitted, no PC number is written to the MRAM.

When the `-setpcsno` option is specified, the `-NOROMSV` option cannot be specified.

### Termination codes

The `svrpl` command returns the following termination codes:

- 0: Normal termination
- 1: Abnormal termination
- 2: Communication error
- 3: Suspension by **Ctrl+C**

## 7. STARTUP AND PU CONTROL

### Name

`svcpuctl` - Controls the remote status.

### Syntax

```
svcpuctl [{-u site} {-s{-stop|-run}}][-time] (status control)
svcpuctl [-u site] -ss (status display)
```

### Description

The `svcpuctl` command controls the specified site (PU) or displays the status of the specified site (PU). The following options can be specified.

#### Options for status control

- `-u site`: Specify the name of the site to be handled. When this option is omitted, the site set in the `RSSITE` environment variable is used.  
Specify the CPU name (the same name as the CP site name).
- `-s`: Omits the confirmation (a prompt that asks whether the user really wants to run the command). Specify the `-s` option together with the `-u` option as necessary. If the `-s` option is omitted, operation by an operator is assumed, and the user is prompted to specify the status (STOP or RUN) interactively.
- `-stop`: The specified site (PU) is placed in the CPU STOP status.
- `-run`: The specified site (PU) is placed in the CPU RUN status. Specify the `-run` option together with the `-s` option as necessary. The `-run` and `-stop` options are mutually exclusive. Do not specify them together.
- `-time`: For a CPU RUN request, sets the time when the specified site is CPU.  
The `svcpuctl` command first displays the states of all PUs mounted in the unit to which the specified site belongs. When users indicate the RUN or STOP status, `svcpuctl` prompts them to specify whether they really want to change the specified site to that status. To change the status of the specified site, enter `yes`. If you enter `no`, `svcpuctl` does nothing and terminates.

#### Options for status display

- `-ss`: The PU status (CPU RUN or STOP) of the specified site is displayed.  
When the CP is in the CPU RUN state and the HP is in the CPU STOP state, RUN (HP STOP) appears.

### Notes on use

- The `svcpuctl` command can control and display the state of only instances of S10VE connected by the connection PC change function of the BASE SYSTEM/S10VE.
- If the HP site name is specified for the site name, an error message appears, and the command terminates.
- The options for status control and for status display are mutually exclusive. Do not specify these two types of options together.

### Termination codes

The `svcpuctl` command returns the following termination codes:

- 0: Normal termination
- 1: Abnormal termination
- 2: Communication error
- 3: Suspension by **Ctrl+C**

## CHAPTER 8 svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

svdebug - Online debugger for S10VE

### Syntax

svdebug [*option*]

### Description

The svdebug command provides online debugging functionality for the S10VE.

When the svdebug command is started and terminated, it outputs the messages described hereinafter.

When the site name prompt appears upon debugger startup, the command is ready to accept the various subcommands.

If breakpoints are set in the S10VE when svdebug is run, the svdebug command displays the setting of each breakpoint.

### Message displayed when the debugger starts:

```
++ debugger start ++
```

```
break point
```

```
name = program-name raddr = program-internal-relative-address object = machine-language-instruction-pattern
```

```
.
```

```
.
```

```
name = program-name raddr = program-internal-relative-address object = machine-language-instruction-pattern  
site-name>
```

### Message displayed when the debugger terminates:

```
++ debugger end ++
```

Note: If the debugger is started with the `-s` option specified, the above messages are not displayed.

### Options

- i *fname*: Key-input results are output to the file specified by *fname*.
- o *fname*: The date and results of operation are output to the file specified by *fname*.
- r *fname*: Subcommands in the command file specified by *fname* are run. After all subcommands are run, the svdebug command automatically terminates. The file created by the `-i` option is used as a command file.
- s *sub command*: The subcommands specified by this option are run directly. After running the subcommands, the svdebug command automatically terminates.
- u *site*: Specify the name of the site to be handled by the debugger. When this option is omitted, the site set in the `RSSITE` environment variable is used.
- debug: Specifies a debug mode. Extended subcommands are usable.

### Termination codes

When it terminates normally, the `svdebug` command returns the value 0. When it terminates abnormally, the command returns the value 1. However, if a subcommand specified by the `-s` option terminates abnormally, the command returns the value 255.

### Notes

- When multiple options are specified, the options specified after the `-s` option are interpreted as subcommands and ignored.

#### Example

`svdebug -i fname -s sub command`: `-i` is interpreted as an option.

`svdebug -s sub command -i fname`: `-i` is interpreted as part of the subcommand specified by the `-s` option.

- When the option specified in the `svdebug` command is not one of the options described in the preceding *Description* section, the following usage information is displayed:

```
Usage: svdebug [options]
Options:
    -i fname          specify a "key-input result file"
    -o fname          specify a "operation result file"
    -r fname          specify a "command file"
    -s sub command   "sub command" direct run
    -u site           specify a "site name"
    -debug           specify debug mode
```

- When creating a key-input results file as the command file used by the `-r` option rather than using the file specified by the `-i` option, note the following points:
  - When data that does not conform to the subcommand specifications is set, the line that contains the data is run without being checked.
  - For subcommands that have interactive interfaces, enter data corresponding to the prompt numbers for each operational step on a per-line basis.
  - Blank lines in the command file are ignored, assuming that a subcommand line is not encountered.
- When the `-r` option or `-s` option is used, breakpoint subcommands (`br`, `rb`, `rr`, `rd`, `go`, and `stickybr`) cannot be used.



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

- List of subcommands

Table 2-7 shows the functions of svdebug.

Table 2-7 Functions of svdebug

Classification	Subcommand	Function
Task starting or stopping	qu	Requests that a task be started.
	ab	Prevents tasks from being started.
	re	Cancels the prevention of task startup.
	ta	Displays task status information.
	su	Suppresses task execution.
	rs	Cancels suppression of task execution.
	tm	Starts a task cyclically.
	ct	Cancels cyclic task startup.
	sht	Displays cyclic task startup.
Memory printing or patching	md	Displays or modifies the contents of memory in accordance with a specified address.
	sd	Displays or modifies the contents of memory in accordance with a specified name.
	bs	Sets data in specified bit positions.
	bg	Displays the data that exists in specified bit positions.
	mcp	Copies the contents of memory.
	mmv	Moves the contents of memory.
	mf	Sets pattern data in the memory.
Breakpoints	br/stickybr	Sets or displays breakpoints.
	rb	Resets breakpoints.
	rd	Displays registers.
	rr	Changes the contents of registers.
	go	Resumes execution from a breakpoint.
System error display	el	Displays error logs.
	ss	Displays system status information.
Current time setting or display	st	Sets the current time.
	gt	Displays the current time.
Uploading, downloading, or comparisons	ld	Transfers backup file data to the controller memory.
	sv	Transfers S10VE memory data to a backup file.
	cm	Compares the contents of a backup file with those of the memory in the S10VE.
Enabling or disabling DHP logging	dr	Enables DHP logging.
	ds	Disables DHP logging.
Ladder debug functionality	lbr	Sets and displays breakpoints.
	lrb	Resets breakpoints.
	lrd	Displays registers.
	lrr	Rewrites the contents of registers.
	lgo	Resumes execution from a breakpoint.
	s	Runs steps.
Other	si	Initializes the stack.
	sp	Displays the amount of stack use.
	svdhp	Displays the DHP.
	svadm	Displays a resource name for an address.
	ps	Starts displaying debug statements.
	pe	Stops displaying debug statements.
	ver	Displays the version of the CPMS.
	help	Displays a list of subcommands.
	q	Terminates the debugger.
!	Runs a command on the development machine at the time that svdebug is run.	

**Name**

qu - Requests that a task be started.

**Syntax**

```
qu tn[,fact]
qu tname[,fact]
```

**Description**

The qu subcommand starts the specified task. Specify the following parameters:

*tn*: Task number (1 to the maximum task number)

*fact*: Start factor (1 to 32)

*tname*: Task name

**Result**

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

The macro return code is shown in ( $\neq 0$ ).

**Notes**

- If *fact* is omitted, 0 is assumed.
- When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter e or press the **Enter** key in this state, the subcommand process terminates.

```
input tn[,fact] or tname[,fact]
:
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

ab - Prevents tasks from being started.

### Syntax

ab *tn1*[-*tn2*]  
ab *tname*

### Description

The ab subcommand prevents the specified tasks from being started. Specify the following parameters:

*tn1*: First task number (1 to the maximum task number)  
*tn2*: Last task number (*tn1* to the maximum task number)  
*tname*: Task name

### Result

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

When *tn1-tn2* is specified, however, the ab subcommand always terminates normally.

The macro return code is shown in ( $\neq 0$ ).

### Notes

When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter e or press the **Enter** key in this state, the subcommand process terminates.

```
input tn1[-tn2] or tname  
:
```

**Name**

`re` - Cancels the prevention of task startup.

**Syntax**

```
re tn1[-tn2]  
re tname
```

**Description**

The `re` subcommand releases the specified tasks from the status in which task startup is prevented. Specify the following parameters:

- tn1*: First task number (1 to the maximum task number)
- tn2*: Last task number (*tn1* to the maximum task number)
- tname*: Task name

**Result**

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

When *tn1*-*tn2* is specified, however, the `re` subcommand always terminates normally.

The macro return code is shown in ( $\neq 0$ ).

**Notes**

When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn1[-tn2] or tname  
:
```

**Name**

`ta` - Displays task status information.

**Syntax**

```
ta tn1[-tn2 [-s|-r]] [-susp]
ta tname [-susp]
```

**Description**

The `ta` subcommand displays status information about the specified tasks. When a specified task is in the SUSPENDED state, in which task execution is suppressed, the contents of the registers involved are also displayed. Specify the following parameters:

*tn1*: First task number (1 to the maximum task number)

*tn2*: Last task number (*tn1* to the maximum task number)

*tname*: Task name

-s: Only task numbers, task names, and task status information are displayed.

-r: Status information about tasks that are not in the NON-EXISTENT, DORMANT, or IDLE state is displayed.

-susp: Forcibly transitions the specified task that is not in the WAIT state or SUSPENDED state to the SUSPENDED state and displays register information.

This option is available when the `-debug` option is specified upon `svdebug` startup. Using this option in other cases results in an error.

The `ta` subcommand displays status information in the following format:

```
tn=*** (0x**)   tname=*****   task state=*****   (0x*****)
tcb top=0x***** fact=0x***** level=** (**)
task top=0x***** stack=0x*****-0x*****
(contents-of-registers)
```

- Explanation of displayed information

`tn`: Task number (decimal or hexadecimal)

`tname`: Task name

`task state`: State of the task (The settings of status bits are displayed in hexadecimal.)

`fact`: Start factor of the task

`level`: Current task level (The value in parentheses is the initial task level.)

`tcb top`: First address of the TCB

`task top`: First address of the task

`stack`: Task stack range

*contents-of-registers*: When the task is in the SUSPENDED state, the contents of the registers involved are displayed.

The contents of registers are displayed in the following format:

```

SR  =0x*****   PC  =0x*****   GBR =0x*****   PR  =0x*****
MACH=0x*****   MACL=0x*****
R0  =0x*****   R1  =0x*****   R2  =0x*****   R3  =0x*****
R4  =0x*****   R5  =0x*****   R6  =0x*****   R7  =0x*****
R8  =0x*****   R9  =0x*****   R10 =0x*****   R11 =0x*****
R12 =0x*****   R13 =0x*****   R14 =0x*****   R15 =0x*****
    
```

Table 2-8 lists the task states, one of which is displayed for a task at a time.

Table 2-8 Task States

State	Meaning
NON EXISTENT	The task is not registered.
DORMANT	Startup of the task is being prevented.
IDLE	The task is waiting to be started.
READY	The task is being run or is ready to be run.
WAIT	The task is waiting for an event.
SUSPENDED	Execution of the task is suppressed.

Table 2-9 shows the status bit strings and their meanings.

Table 2-9 Status Bit Strings

Status bit string	Meaning
0x1	Multiple starts were requested.
0x10	Execution is being suppressed by DELAY.
0x20	Execution is being suppressed by SUSP.
0x40	The unlocking of resources that are being locked by RSERV or PRSRV is being awaited.
0x80	Execution is being suppressed by a breakpoint.
0x1000	Processing by EXIT is in progress.
0x2000	Processing by RLEAS is pending.
0x4000	Processing by ABORT is in progress.
0x8000	Processing by QUEUE is pending.

Note: Multiple bits in each status bit string might be turned on at the same time.

### Notes

- When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn1[-tn2 [-s|-r]] or tname
:
```

- When the `-susp` option is specified, the task range cannot be specified.
- If the task is in the DORMANT or NON-EXISTENT state when the `-susp` option is specified, the system generates an error message and terminates the subcommand.
- When the `-susp` option is specified, the task is temporarily placed in the SUSPENDED state to acquire register information, even if the task was not in the SUSPENDED state.

**Name**

`su` - Suppresses task execution.

**Syntax**

```
su tn1[-tn2]  
su tname
```

**Description**

The `su` subcommand suppresses execution of the specified tasks. Specify the following parameters:

- tn1*: First task number (1 to the maximum task number)
- tn2*: Last task number (*tn1* to the maximum task number)
- tname*: Task name

**Result**

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

When *tn1*-*tn2* is specified, however, the `su` subcommand always terminates normally.

The macro return code is shown in ( $\neq 0$ ).

**Notes**

When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn1[-tn2] or tname  
:
```



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`rs` - Cancels suppression of task execution.

### Syntax

```
rs tn1[-tn2]  
rs tname
```

### Description

The `rs` subcommand releases the specified tasks from the state in which task execution is suppressed. Specify the following parameters:

*tn1*: First task number (1 to the maximum task number)  
*tn2*: Last task number (*tn1* to the maximum task number)  
*tname*: Task name

### Result

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

When *tn1-tn2* is specified, however, the `rs` subcommand always terminates normally.

The macro return code is shown in ( $\neq 0$ ).

### Notes

When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn1[-tn2] or tname  
:
```

**Name**

`tm` - Starts a task cyclically.

**Syntax**

`tm`

*id*:

*tn[,fact]*: (or *tname[,fact]*)

*t, cyct*:

(or)

`tm`

*id*:

*tn[,fact]*: (or *tname[,fact]*)

*t*:

**Description**

The `tm` subcommand starts the specified task cyclically. When `tm` is started, subprompts are displayed. Specify the following parameters:

*id*: Type of the timer task to be started (1 to 4)

*tn*: Task number (1 to the maximum task number)

*tname*: Task name

*fact*: Start factor (1 to 32)

*t*: Time of day or length of time relative to the current time when the timer event is started for the first time. The relative time must be specified in milliseconds within the following range:

When *id* is 1 or 3: 1 to 86,400,000

When *id* is 2 or 4: 0 to 86,399,999

*cyct*: Interval at which events are generated cyclically. The interval must be specified in milliseconds within the range of 1 to 86,400,000.

For more information about *id*, *t*, and *cyct*, see Table 2-10.

**Result**

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

The macro return code is shown in ( $\neq 0$ ).

## Notes

- If the *fact* parameter is omitted, 0 is assumed.
- If you enter `e` or press the **Enter** key while the system is waiting for parameter input, the subcommand terminates.

Table 2-10 Explanation of the *id*, *t*, and *cyct* Parameters

Timer event	id	t	cyct	Explanation
Length-of-time basis	1	Relative time up to the start time, measured from the current time of day		After the length of time specified by the <i>t</i> parameter elapses, the task specified by the <i>tn</i> or <i>tname</i> parameter is started.
Time-of-day basis	2	Time of day at which the task is started, measured from 00:00		The task specified by the <i>tn</i> or <i>tname</i> parameter is started at the time specified by the <i>t</i> parameter.
Length-of-time and cycle basis	3	Relative time up to the start time, measured from the current time of day (relative time up to the first start)	Interval at which the task is started cyclically after the first start	After the length of time specified by the <i>t</i> parameter elapses, the task specified by the <i>tn</i> or <i>tname</i> parameter is started. The task is then started cyclically at the interval specified by the <i>cyct</i> parameter.
Time-of-day and cycle basis	4	Time of day at which the task is started, measured from 00:00 (time of day at the first start)	Interval at which the task is started cyclically after the first start	The task specified by the <i>tn</i> or <i>tname</i> parameter is started at the time specified by the <i>t</i> parameter. The task is then started cyclically at the interval specified by the <i>cyct</i> parameter.

**Name**

`ct` - Cancels cyclic task startup.

**Syntax**

```
ct tn[,fact]
ct tname[,fact]
```

**Description**

The `ct` subcommand cancels the cyclic startup of the specified task. Specify the following parameters:

*tn*: Task number (1 to the maximum task number)

*fact*: Start factor (1 to 32)

*tname*: Task name

**Result**

OK (0) : Normal termination

NG ( $\neq 0$ ) : Macro error

The macro return code is shown in ( $\neq 0$ ).

**Notes**

- When the *fact* parameter is omitted, 0 is assumed.
- When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:). If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn[,fact] or tname[,fact]
:
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`sht` - Displays cyclic task startup.

### Syntax

`sht`

### Description

The `sht` subcommand displays the cyclic startup of all currently selected tasks.

### Result

Cyclic task startup is displayed in the following form:

ID	TN	FACT	TIME	CYT
*	***	**	*****/**/** **:*:*:*.***	*****
			.	
			.	
			.	

ID: Timer type

TN: Task number

FACT: Start factor

TIME: Start time (*year . month . day hour : minute : second . millisecond*)

CYT: Cycle (in milliseconds)

### Notes

Multiple instances of the `sht` subcommand cannot be started simultaneously for the same site.

**Name**

md - Displays or modifies the contents of memory in accordance with a specified address.

**Syntax**

```
md
1 storage (s,m,*): {s}
                  {m}
                  {*}
                  {e}
                  {Enter}
2 addr : {addr1 [-addr2] [-h |-d |-f] [-l |-w |-b] [-all |-omit] }
         {addr1 [-addr2] [-fd] [-all |-omit] }
         {addr1 [,len] [-h |-d |-f] [-l |-w |-b] [-all |-omit] }
         {addr1 [,len] [-fd] [-all |-omit] }
         {*n}
         {e}
0xaaaaaaaa 0xdddddddd : {data}
                        {Enter}
                        {e}
```

Note: The underlined portions are to be entered by the user.

**Description**

The md subcommand displays or modifies the contents of memory specified by a logical address. When this subcommand is started, subprompts are displayed. Respond to each subprompt as follows:

- ```
1 storage (s,m,*)
```
- s: The backup file is modified or displayed.
  - m: Memory in the controller is modified or displayed.
  - Enter**: Memory in the controller is modified or displayed.
  - \*: Both the backup file and memory in the real machine are modified.
  - e: The subcommand is instructed to terminate.

Note: When storage=\* is specified, data in the backup file is displayed.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

2 addr

*addr1-addr2*: Data from the first display address (*addr1*) to the last display address (*addr2*) is displayed.

*addr1, len*: Data is *displayed* for the number of bytes specified by *len*, starting from the first display address (*addr1*).

-h: Data is displayed in hexadecimal.

-d: Data is displayed in decimal.

-f: Data is displayed in single-precision floating-point format.

-fd: Data is displayed in double-precision floating-point format.

-l: The data length is set to four bytes.

-w: The data length is set to two bytes.

-b: The data length is set to one byte.

-all: Specifies that an abbreviated display mode is not to be used when there are two or more consecutive lines of the same data.

-omit: Specifies that an abbreviated display mode is to be used when there are two or more consecutive lines of the same data.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1).

e: The subcommand is instructed to terminate.

0xaaaaaaaa 0xdddddddd

*data*: Specify a new value as the data to be modified.

**Enter**: No data is changed.

e: Control returns to 2 addr.

### Notes

- When the *addr2* or *len* parameter is not specified in 2 addr, data change (patch) mode is enabled. When either parameter is specified, data display (print) mode is enabled. Upon completion of data display, the user is prompted for input again in 2 addr.
- When neither a data output format option nor a data length option is specified, the last options specified in this subcommand are used. By default, the -h and -l options are used, that is, four-byte data is displayed in hexadecimal.
- The values specifiable in data change (patch) mode depend on the combination of the specified data output format option and data length option. Table 2-11 shows the combinations of specifiable values and options.

Table 2-11 Relationships between Specifiable Values and Options

| fmt \ size       | -l | -w | -b | No specification |
|------------------|----|----|----|------------------|
| -h               | ○  | ○  | ○  | -                |
| -d               | ○  | ○  | ○  | -                |
| -f               | ⊙  | ○  | ○  | -                |
| -fd              | ×  | ×  | ×  | ⊙                |
| No specification | -  | -  | -  | -                |

○: Octal, decimal, or hexadecimal numbers can be specified.  
 ⊙: Real numbers can be specified.  
 -: Specifiable values depend on the values of the previously specified data output format and size.  
 ×: Invalid combination

fmt: Data output format size: Data size

- After data is displayed or changed by specifying -fd (double-precision floating-point format) as the data output format, data can be displayed or changed, with either the data output format or data size option omitted. When the data output format option is omitted, -h (hexadecimal format) is assumed. When the data size option is omitted, -l (four bytes) is assumed.
- If s is specified for 1 storage and an address at which no backup file exists is specified for 2 addr, the system displays an error message and waits for 2 addr input.
- In data display (print) mode, the display format depends on the combination of the specified data output format option and data length options, as shown in Table 2-12.

Table 2-12 Display Formats Depending on the Combination of Options

| fmt \ size       | -l  | -w  | -b  | No specification |
|------------------|-----|-----|-----|------------------|
| -h               | h/4 | h/2 | h/1 | -                |
| -d               | d/4 | d/2 | d/1 | -                |
| -f               | f/4 | h/2 | h/1 | -                |
| -fd              | ×   | ×   | ×   | f/8              |
| No specification | -   | -   | -   | -                |

fmt/size: fmt: h (hexadecimal)  
 d (decimal)  
 f (floating-point)  
 size: byte size  
 -: Specifiable values depend on the values of the previously specified data output format and size.  
 ×: Invalid combination

fmt: Data output format size: Data size



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

- Figure 2-4 shows the memory access range allowed for the `md` subcommand that displays (reads) or modifies (writes) the contents of memory. However, spaces are inaccessible if they are not mapped in the physical memory.

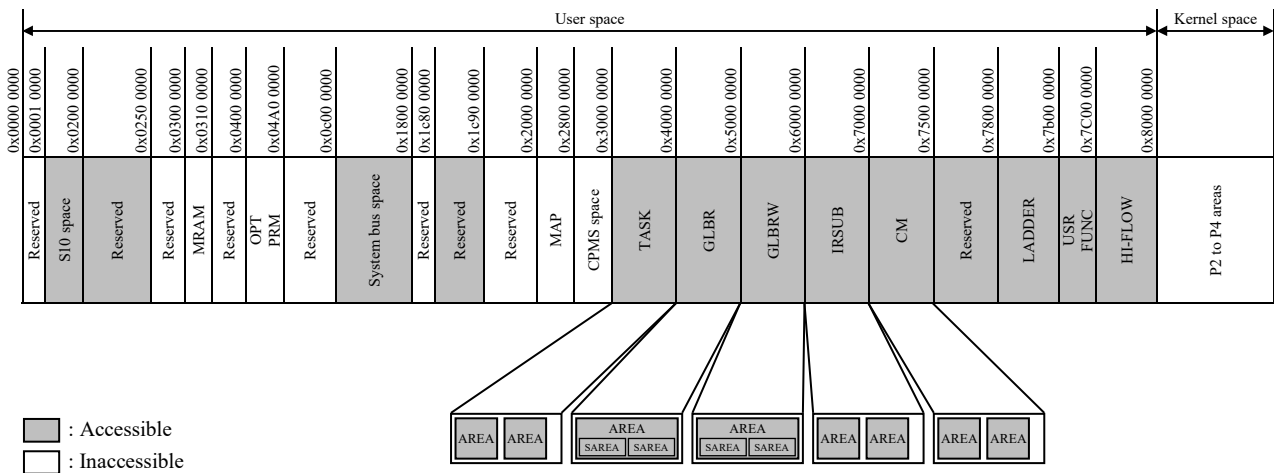
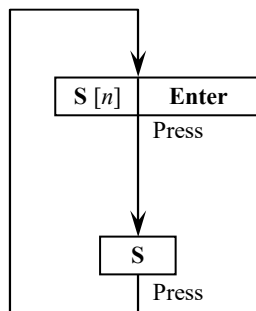


Figure 2-4 Memory Access Range

- `md` supports the dynamic display function in the data display (print) mode. Figure 2-5 shows the operation procedure for dynamic display.



Pressing the **S**, *n*, and **Enter** keys starts dynamic display.

For *n*, specify the interval time (in ms) for dynamic display. If *n* is omitted, the interval time becomes 0 ms. A value in the range from 0 to 60000 can be specified for *n*.

During monitoring, only the **S** key can be used.

Dynamic display is terminated. (Key operation becomes possible.)

Figure 2-5 Operation Procedure for Dynamic Display

**Name**

sd - Displays or modifies the contents of memory in accordance with a specified task name, subprogram name, program name, or global name.

**Syntax**

sd

1 name : {name [-t |-s |-g |-b |-w]}

{e}

2 storage (s,m,\*) : {s}

{m}

{\*}

{\*n}

{e}

**Enter**

3 baddr : {addr [-h |-d |-f] [-l |-w |-b] [-all |-omit]}

{addr [-fd] [-all |-omit]}

{\*n}

{e}

4 raddr : {addr1 [-addr2 |-\*]}

{addr1 [, len |, \*]}

{\*}

{\*n}

{e}

0xaaaaaaaa (0x1111111) 0xdddddddd : {data}

**Enter**

{e}

Note: The underlined portions are to be entered by the user.

## Description

The `sd` subcommand displays or modifies the contents of memory specified by a task name, subprogram name, program name, or global name. When this subcommand is started, subprompts are displayed. Respond to each subprompt as follows:

1 `name`

*name*: Name for which memory is displayed or modified

`-t`: The name specified by *name* is handled as a program name. The relative address is handled as the address counted from the beginning of the text.

`-s`: The name specified by *name* is handled as a subprogram name.

`-g`: The name specified by *name* position is handled as the secondary partition area (SAREA) name of the global data.

`-b`: Specifies that the BSS area of a program corresponding to the program name specified by *name* is to be handled. The relative address is the address counted from the beginning of the BSS area for a specified program.

`-w`: The stack area of the task identified by the name specified by *name* is handled. The relative address is the address counted from the beginning of the stack area for the specified task.

`e`: The subcommand is instructed to terminate.

Note: When none of the `-t`, `-s`, `-g`, `-b`, and `-w` options are specified, `-g` is assumed.

2 `storage (s, m, *)`

`s`: The backup file is modified or displayed.

`m`: Memory in the real machine is modified or displayed.

**Enter**: Memory in the real machine is modified or displayed.

`*`: Both the backup file and memory in the real machine are modified.

`*n`: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1).

`e`: The subcommand is instructed to terminate.

Note: When `storage=*` is specified, data in the backup file is displayed.

3 `baddr`

*addr*: Specify an address relative to the beginning of the area to be modified or displayed.

`*n`: Control is returned to the preprocessing indicated by the prompt number specified in *n* (where *n* = 1 or 2).

`-h`: Data is displayed in hexadecimal.

`-d`: Data is displayed in decimal.

`-f`: Data is displayed in floating-point format.

`-fd`: Data is displayed in double-precision floating-point format.

`-l`: The data length is set to four bytes.

`-w`: The data length is set to two bytes.

`-b`: The data length is set to one byte.

`-all`: Specifies that an abbreviated display mode is not to be used even when there are two or more consecutive lines of the same data.

`-omit`: Specifies that an abbreviated display mode is to be used when there are two or more consecutive lines of the same data.

`e`: The subcommand is instructed to terminate.

4 raddr

*addr1-addr2*: Data from the first display address (*addr1*) to the last display address (*addr2*) is displayed. (These addresses are counted starting from *addr* of `baddr`.)

*addr1, len*: Data is displayed for the number of bytes specified by *len*, starting from the first display address (*addr1*). (The address is counted starting from *addr* of `baddr`.)

*addr1, \**: The area indicated by the specified symbol is displayed, starting from the first display address (*addr1*) and continuing up to the end of the area. (The address is counted starting from *addr* of `baddr`.)

*addr1-\**: The area indicated by the specified symbol is displayed, starting from the first display address (*addr1*) and continuing up to the end of the area. (The address is counted, starting from *addr* of `baddr`.)

\*: The entire area indicated by the specified symbol is displayed.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1, 2, or 3).

e: The subcommand is instructed to terminate.

0xaaaaaaaa (0x1111111) 0xdddddddd

*data*: Specify a new value as the data to be modified.

**Enter**: No data is changed.

e: Control is returned to 4 raddr.

## Notes

- When only the *addr1* parameter is specified in 4 raddr, data change (patch) mode is enabled. Other combinations of specified parameters enable data display (print) mode. Upon completion of data display, the user is prompted for input again in 4 raddr.
- When neither a data output format option nor a data length option is specified, the last options specified in this subcommand are used. By default, the `-h` and `-l` options are used, that is, four-byte data is displayed in hexadecimal.
- The areas that can be displayed or modified by `sd` are secondary partition areas (SAREAs) allocated in the global area when a global name is specified. When a program or subprogram name is specified, the displayable or modifiable area is the text and data sections, the bss section, or the stack section.
- The values specifiable in data change (patch) mode depend on the combination of the specified data output format option and data length option. Table 2-11 shows the relationships between specifiable values and options.
- For the display format of the data display (print), the combinations of the data output format and data length options are the same as that for the `md` subcommand.
- As with the `md` subcommand, the `sd` subcommand supports the dynamic display function in the data display (print) mode. For more information about dynamic display operation, see the section on the `md` subcommand.
- If an area containing no backup file is specified for 1 name and `s` is specified for 2 storage, the system displays an error message and waits for 2 storage input.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

bs - Sets data in specified bit positions.

### Syntax

```
bs
1 storage (s, m, *) : {s}
                    {m}
                    {*}
                    {e}
                    {Enter}
2 addr : {addr}
        {*n}
        {e}
3 bit :  {bit1, len}
        {bit1-bit2}
        {*n}
        {e}
4 data : {data}
        {*n}
        {e}
```

Note: The underlined portions are to be entered by the user.

### Description

The `bs` subcommand sets data at the bit positions that have the specified addresses. When this subcommand is started, subprompts are displayed. Respond to each subprompt as follows:

1 storage (s, m, \*)

s: Data is set in bits in the backup file.

m: Data is set in bits in memory in the real machine.

**Enter:** Data is set in bits in memory in the real machine.

\*: Data is set in bits in both the backup file and memory in the real machine.

e: The subcommand is instructed to terminate.

2 addr

addr: Specify the address of memory in which to set data.

\*n: Control is returned to the preprocessing indicated by the prompt number  $n$  (where  $n = 1$ ).

e: The subcommand is instructed to terminate.

## 3 bit

*bit1, len*: Data is set in bits whose length is specified by *len*, starting from the first bit number (*bit1*).

*bit1-bit2*: Data is set in bits from the first bit number (*bit1*) to the last bit number (*bit2*).

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1, 2, or 3).

e: The subcommand is instructed to terminate.

## 4 data

*data*: Specify data to be set. When the data starts with 0x or 0X, it is handled as hexadecimal data. Otherwise, the data is handled as binary data.

Specify as many patterns as the number of specified bits.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1, 2, or 3).

e: The subcommand is instructed to terminate.

## Notes

- The user can modify the contents of memory within the same range as that for the `md` subcommand.
- If `s` is specified for 1 `storage` and an address at which no backup file exists is specified for 2 `addr`, the system displays an error message and waits for 2 `addr` input.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

bg - Displays the data that exists in specified bit positions.

### Syntax

```
bg
1 storage (s, m, *) : {s}
                    {m}
                    {*}
                    {e}
                    {Enter}
2 addr : {addr}
        {*n}
        {e}
3 bit : {bit1, len}
        {bit1-bit2}
        {*n}
        {e}
```

Note: The underlined portions are to be entered by the user.

### Description

The bg subcommand displays data at the specified bit positions. When this subcommand is started, subprompts are displayed. Respond to each subprompt as follows:

1 storage (s, m, \*)

s: Bit data is read from the backup file.

m: Bit data is read from memory in the real machine.

**Enter:** Bit data is read from memory in the real machine.

\*: Bit data is read from both the backup file and memory in the real machine.

e: The subcommand is instructed to terminate.

Note: When storage=\* is specified, backup file data is displayed.

2 addr

addr: Specify the address of memory from which to read bit data.

\*n: Control is returned to the preprocessing indicated by the prompt number n (where n = 1).

e: The subcommand is instructed to terminate.

3 bit

bit1, len: Data in bits whose length is specified by len is displayed, starting from the first bit number (bit1).

bit1-bit2: Data in bits from the first bit number (bit1) to the last bit number (bit2) is displayed.

\*n: Control is returned to the preprocessing indicated by the prompt number n (where n = 1 or 2).

e: The subcommand is instructed to terminate.

**Result**

The contents of memory are displayed in the following format:

|             |                                 |
|-------------|---------------------------------|
| a d d r     | 0 1 2 3 4 5 6 7 8 9 a b c d e f |
| 0xNNNNNNNNN | c c c c c c c c c c c c c c c c |

0xNNNNNNNNN: Address

c: 1, 0, or \*. When the specified bit position is out of range, \* is displayed.

After the contents of memory are displayed, the user is prompted for input again in 2 addr.

**Notes**

- The user can display the contents of memory within the same range as that for the md subcommand.
- If s is specified for 1 storage and an address at which no backup file exists is specified for 2 addr, the system displays an error message and waits for 2 addr input.



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

mcp - Copies the contents of memory.

### Syntax

```
mcp
1 storage (s,m,*): {s}
                  {m}
                  {*}
                  {e}
                  Enter
2 s_addr : {addr1, len}
           {addr1-addr2}
           {*n}
           {e}
3 d_addr : {addr}
           {*n}
           {e}
```

Note: The underlined portions are to be entered by the user.

### Description

The mcp subcommand copies the contents of memory or the backup file to a specified address.

1 storage (s,m,\*)

s: The backup file is copied.

m: Memory in the object machine is copied.

**Enter:** Memory in the object machine is copied.

\*: Both the backup file and memory in the object machine are copied.

e: The subcommand is instructed to terminate.

2 s\_addr

*addr1, len*: A copy is made starting from address *addr1*. *len* specifies the number of bytes to be copied.

*addr1-addr2*: A copy is made starting from address *addr1* and ending at address *addr2*.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1).

e: The subcommand is instructed to terminate.

3 d\_addr

*addr*: Specify the first address of the destination area to which a copy is sent.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1 or 2).

e: The subcommand is instructed to terminate.

Notes

- The mcp subcommand displays the following confirmation message after the destination address is entered in 3 d\_addr but before a copy starts.

```

Range of source addresses      Range of destination addresses
copy : 0x*****-0x***** -> 0x*****-0x*****
memory data copy ok ? (y/n) :
    
```

Message corresponding to the storage type

storage type = s: backup file  
 m: memory  
 \*: memory and backup file

Enter a y or Y to start a copy. The following messages appear when a copy starts and terminates. If another character is entered, control is returned to the prompt in 2 s\_addr without a copy being made.

```

At the start of copying      ***** memory copy start *****
At the end of copying        ***** memory copy end *****
    
```

Message corresponding to the storage type

- When an odd-numbered address is entered, 1 is subtracted from the address to make it an even-numbered address.
- At the end of memory copying, control is returned to the prompt in 2 s\_addr.
- When s or \* is specified for 1 storage and an address at which no backup file exists is specified, the system displays an error message and waits for 2 s\_addr input.
- The memory access range allowed for mcp is indicated by the shaded areas in the following figure. Unlike md, bs, and bg, mcp does not access the R700 system bus space or S10 space. Furthermore, it cannot access spaces that are not mapped in the physical memory.

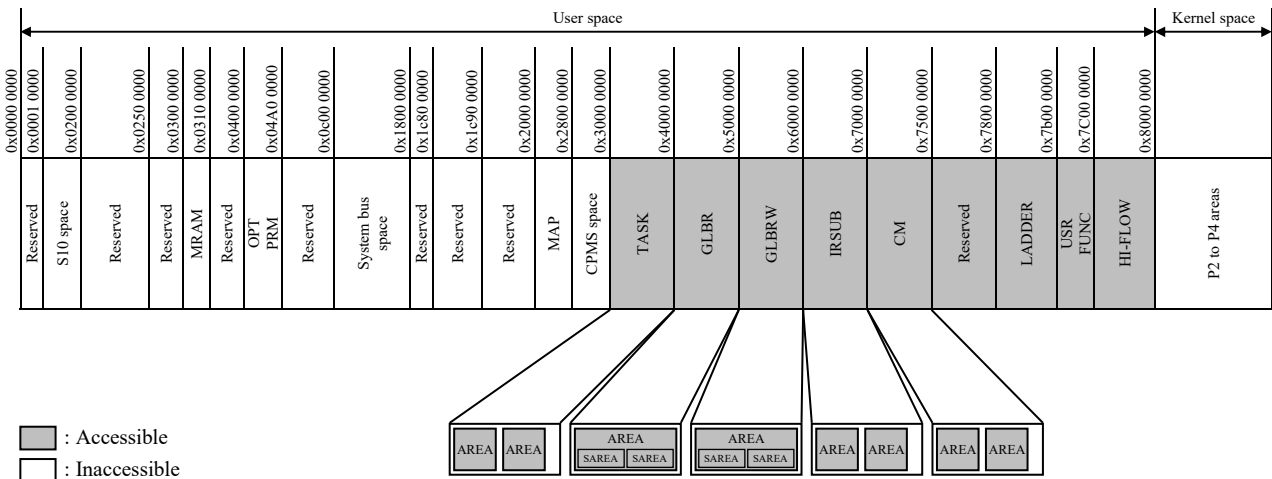


Figure 2-6 Memory Access Range

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

mmv - Moves the contents of memory.

### Syntax

```
mmv
1 storage (s, m, *) : {s}
                       {m}
                       {*}
                       {e}
                       {Enter}
2 s_addr : {addr1, len}
           {addr1-addr2}
           {*n}
           {e}
3 d_addr : {addr}
           {*n}
           {e}
```

Note: The underlined portions are to be entered by the user.

### Description

The mmv subcommand moves the contents of memory and clears the source to zero.

1 storage (s, m, \*)

s: The contents of the backup file are moved.

m: Contents of memory in the object machine are moved.

**Enter:** Contents of memory in the object machine are moved.

\*: The contents of both the backup file and memory in the object machine are moved.

e: The subcommand is instructed to terminate.

2 s\_addr

*addr1, len*: A move is made starting from address *addr1*. *len* specifies the number of bytes to be moved.

*addr1-addr2*: A move is made starting from address *addr1* and ending at address *addr2*.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1).

e: The subcommand is instructed to terminate.

3 d\_addr

*addr*: Specify the first address of the destination area to which to move contents.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1 or 2).

e: The subcommand is instructed to terminate.

Notes

- The `mmv` subcommand displays the following confirmation message after the destination address is entered in 3 `d_addr` but before a move starts.

```

Range of source addresses      Range of destination addresses
move : 0x*****-0x***** -> 0x*****-0x*****
memory data move ok ? (y/n) :
    
```

Message corresponding to the storage type

storage type = s: backup file  
 m: memory  
 \*: memory and backup file

Enter a `y` or `Y` to start a move. The following messages appear when a move starts and terminates. If another character is entered, control is returned to the prompt in 2 `s_addr` without a move being made.

```

At the start of a move      ***** memory move start *****
At the end of a move       ***** memory move end *****
    
```

↑  
 Message corresponding to the storage type

- When an odd-numbered address is entered, one is subtracted to make it an even-numbered address.
- The memory access range allowed for `mmv` is the same as that for `mcp`.
- At the end of moving contents of memory, control is returned to the prompt in 2 `s_addr`.
- If `s` or `*` is specified for 1 `storage` and an address at which no backup file exists is specified, the system displays an error message and waits for 2 `s_addr` input.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

mf - Sets pattern data in the specified address range.

### Syntax

```
mf
1 storage (s,m,*): {s}
                  {m}
                  {*}
                  {e}
                  Enter
2 addr : {addr1, len [-l/-w/-b]}
         {addr1-addr2 [-l/-w/-b]}
         {*n}
         {e}
3 data : {data}
         {*n}
         {e}
```

Note: The underlined portions are to be entered by the user.

### Description

The mf subcommand sets pattern data within a given address range.

1 storage (s,m,\*)

s: Pattern data is set in the backup file.

m: Pattern data is set in memory in the object machine.

**Enter:** Pattern data is set in memory in the object machine.

\*: Pattern data is set in both the backup file and memory in the object machine.

e: The subcommand is instructed to terminate.

2 addr

*addr1, len*: Pattern data is set for the number of cases specified by *len*, starting from address *addr1*.

*addr1-addr2*: Pattern data is set, starting from address *addr1* and ending at address *addr2*.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1).

e: The subcommand is instructed to terminate.

-l: The data length is set to 4 (bytes).

-w: The data length is set to 2 (bytes).

-b: The data length is set to 1 (byte).

3 data

*data*: Specify data to be set.

\**n*: Control is returned to the preprocessing indicated by the prompt number *n* (where *n* = 1 or 2).

e: The subcommand is instructed to terminate.

**Notes**

- If no data length is specified, the data length in the mf command specified last is used. The default is -1 (four bytes).
- The mf subcommand displays the following confirmation message after pattern data is set in 3 data but before the setting of the pattern data starts.

| Range of addresses to set pattern data | Pattern data (hexadecimal) |
|----------------------------------------|----------------------------|
| write : 0x*****-0x*****                | pattern data = 0x*****     |
| memory data write ok ? (y/n) :         |                            |

↑  
Message corresponding to the storage type

storage type = s: backup file  
 m: memory  
 \*: memory and backup file

Enter a y or Y to start the setting of pattern data. If another character is entered, control is returned to the prompt in 2 addr without pattern data being set.

- If the specified address is not on a data-length boundary, the value of the address is decremented accordingly.
- The specifiable address range allowed for mf is the same as the memory access range for mcp.
- After pattern data is set in the memory, control is returned to the prompt in 2 addr.
- If s or \* is specified for 1 storage and an address at which no backup file exists is specified for 2 addr, the system displays an error message and waits for 2 addr input.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`e1` - Displays error logs.

### Syntax

`e1 [-u site] [-f {s|m|l}] [-logno] [+count] [-o fname]`

### Description

The `e1` subcommand starts the `svelog` command to display error logs. For more information about the `e1` subcommand, see the command specifications of the `svelog` command.

### Name

`ss` - Display system status information.

### Syntax

`ss`

### Description

The `ss` subcommand starts the `svcpuctl` command to display system status information. For more information about the `ss` subcommand, see the command specifications of the `svcpuctl` command.

**Name**

`st` - Sets the current time.

**Syntax**

`st [yyyy.mm.dd.hh:mt:ss]`

**Description**

The `st` subcommand sets the time as managed by the S10VE as the current time. Specify the following parameters:

*yyyy*: Year (four digits)

*mm*: Month

*dd*: Day

*hh*: Hours

*mt*: Minutes

*ss*: Seconds

Note: Enter each part of the time data as decimal values.

**Result**

OK (0) : Normal termination

**Notes**

- When the time setting parameters (*yyyy.mm.dd.hh:mt:ss*) are not specified in this subcommand, the following message is displayed to prompt the user for them. If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.  
YYYY.MM.DD.HH:MT:SS :
- When this subcommand is run directly with the `-s` option and an error occurs, no error message is displayed.



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`gt` - Displays the current time.

### Syntax

`gt`

### Description

The `gt` subcommand displays the current time as managed by the S10VE.

### Result

The current time is displayed in the following format:

*yyyy.mm.dd.hh:mt:ss*

*yyyy*: Year (four digits)

*mm*: Month

*dd*: Day

*hh*: Hours

*mt*: Minutes

*ss*: Seconds

**Name**

br - Sets or displays breakpoints.  
 stickybr

**Syntax**

```
br [pname break1 ... break5 [-t|-s]]
stickybr [pname break1 ... break5 [-t|-s]]
```

**Description**

The `br` and `stikybr` subcommands set breakpoints or display breakpoints that are set. Breakpoints can only be set in a task program (TEXT space) and indirect link subprogram (TEXT space).

When a break occurs in a task, the task goes into the WAIT state. In this case, the task cyclic start timer continues running so that timer startup takes place. However, the task stays in the WAIT state unless execution is resumed from a breakpoint.

While a task is stopped due to a breakpoint, no break occurs at the other breakpoints. The other breakpoints become valid when the task stopped due to a break is resumed.

The following parameters are specified:

*pname*: Specifies the name of the program for which breakpoints are to be set.

*break1* to *break5*: Specifies breakpoints (relative addresses in a program).

-t: Indicates that the specified program name is a program name for a task.

-s: Indicates that the specified program name is a subprogram name.

Note: When the -t and -s options are not specified, a search will be conducted in the task and subprogram order.

**Result**

The following message appears when breakpoint setup is completed normally:

```
break point set
name = program-name raddr = program-internal-relative-address object = machine-language-instruction-pattern
```

When *pname* and *break* are not specified, br displays the currently set breakpoints as follows:

```
break point
name = program-name raddr = program-internal-relative-address object = machine-language-instruction-pattern
name = program-name raddr = program-internal-relative-address object = machine-language-instruction-pattern *BREAK*
```

·  
·

↑  
This mark appears for a breakpoint at which execution is currently halted.

An error message appears if breakpoint setup is being performed by another terminal.

## Notes

- Up to five breakpoints can be set for each site.
- When a breakpoint is reached, the following message appears:

```
break!!
```

```
tn = task-number pname = program-name raddr = program-internal-relative-address
```

- If `rb`, `rd`, `rr`, `go`, or a similar subcommand terminates abnormally, you must issue only `br` to check the breakpoint status. This makes it possible to align the breakpoint information of the development machine with the information of the S10VE if the two sets of information do not match.
- After the S10VE is restarted, the breakpoint information that was set by the `br` subcommand before the S10VE stopped is not stored. In this case, you must issue only `br` to adjust the breakpoint information of the development machine for the S10VE. The breakpoint that was set by the `stickybr` subcommand is not reset even by restarting the S10VE with a reset start.
- No breakpoint can be set for the RPC server task, system initial start task (SIST), and built-in subroutines.
- This subcommand can be used when the `-debug` option is specified to start `svdebug`. When the subcommand is used in other situations, an error occurs.
- If the execution machine differs from the development machine in breakpoint setup due to a S10VE restart or abnormal debugger termination, proceed as follows:  
Run the `br` subcommand with no parameter specified. When any breakpoint setup remains after execution, clear it with the `rb` subcommand.

**Usage example**

The following describes the breakpoint setup procedure.

- The following example explains the procedure for setting a breakpoint at point (1) in situations where a task program named prog is generated from prog1.c and prog2.c.

**prog1.c**

```
main()
{
    int    a, b, c;
    int    ret;

    a = 10;
    b = 20;

    c = add( a , b );
    ans_print( c );

    exit();
}

int add( int a , int b )
{
    int ans;

    ans = a + b; ← (1)

    return(ans);
}
```

**prog2.c**

```
extern char  glb01_g[1024];      /* 1024byte */

void ans_print( int ans )
{
    int    ret;

    ret = rs_printf( &glb01_g[0] , "anser = %d\n" , ans );

    return;
}
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

- A C-language source file can be inserted into an assembler source when source compilation is performed with the `-SH=SO -1` option specified. `prog1.lst` is referenced in order to set a breakpoint in the `prog1.c` source.

Because the C-language source for breakpoint setup is (1), the assembler instruction (2), which immediately follows (1), corresponds to the C-language source.

The offset in the corresponding instruction source (`prog1.c`) is `0x00000026`.

### `prog1.lst`

```
***** OBJECT LISTING *****
FILE NAME: prog1.c
SCT OFFSET  CODE      C LABEL      INSTRUCTION OPERAND      COMMENT
          prog1.c 1  main()
P 00000000          _main:                ; function: main
                                ; frame size=8
          00000000 2FE6          MOV.L        R14,@-R15
          00000002 4F22          STS.L        PR,@-R15
          prog1.c 2  {
          prog1.c 3          int      a, b, c;
          prog1.c 4          int      ret;
          prog1.c 5
          prog1.c 6          a = 10;
          prog1.c 7          b = 20;
          prog1.c 8
          prog1.c 9          c = add( a , b );
          00000004 E514          MOV #20,R5 ; H'00000014
          00000006 DE09          MOV.L        L12,R14 ; H'FFE7FFFF
          00000008 B00D          BSR _add
          0000000A E40A          MOV #10,R4 ; H'0000000A
          0000000C 016A          STS FPSCR,R1
          prog1.c 10         ans_print( c );
          0000000E D208          MOV.L        L12+4,R2 ; _ans_print
          00000010 6403          MOV R0,R4
          00000012 21E9          AND R14,R1
          00000014 420B          JSR @R2
          00000016 416A          LDS R1,FPSCR
          prog1.c 11
          prog1.c 12         exit();
          00000018 D606          MOV.L        L12+8,R6 ; _exit
          0000001A 076A          STS FPSCR,R7
          0000001C 27E9          AND R14,R7
          0000001E 476A          LDS R7,FPSCR
          00000020 4F26          LDS.L        @R15+,PR
          00000022 462B          JMP @R6
          00000024 6EF6          MOV.L        @R15+,R14
          prog1.c 13 }
          prog1.c 14
          prog1.c 15 int add( int a , int b )
```

(The figure continues on the next page.)

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

```
00000026      _add:                                ; function: add
  ; frame size=0
    prog1.c 16  {
    prog1.c 17      int  ans;
    prog1.c 18
    prog1.c 19      ans = a + b; (1)
    prog1.c 20
    prog1.c 21      return(ans);
    00000026 345C      ADD      R5,R4 (2)
    prog1.c 22  }
00000028 000B      RTS
0000002A 6043      MOV      R4,R0
0000002C      L12:
0000002C FFE7FFFF      .DATA.L  H'FFE7FFFF
00000030 <00000000>    .DATA.L  _ans_print
00000034 <00000000>    .DATA.L  _exit
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

- When the `svload` command is run with the `-P` option specified to generate an executable program, a map file will be generated.

If you check `SECTION=P` in the map, you will see that `prog1.obj` begins with `0x3003c014`. Because the internal relative address for breakpoint setup of the source is `0x00000014`, the breakpoint setup address is `0x3003c028` ( $= 0x3003c014 + 0x00000014$ ).

The internal relative address of the program can be determined from the breakpoint setup address and the start address of the program, and is therefore `0x00000028` ( $= 0x3003c028 - 0x3003c000$ ).

### prog.map

```
SECTION=P
FILE=C:\WINDOWS\renix\tmp\str2545.obj
    3003c000 3003c013      14
__start
    3003c000      0 none ,g      *
L237
    3003c00c      0 none ,l      *
FILE=prog1.obj
    3003c014 3003c037      24
_main
    3003c014      14 func ,g      *
_add
    3003c028      10 func ,g      *
FILE=prog2.obj
    3003c038 3003c04f      18
_ans_print
    3003c038      18 func ,g      *
FILE=cpms_exit
    3003c050 3003c09f      50
_exit
    3003c098      0 none ,g      *
FILE=rpdp_rsprintf
    3003c0a0 3003c113      74
_rs_printf
    3003c0a0      74 func ,g      *
FILE=rpdp_rssetmsg
    3003c114 3003c16b      58
_rssetmsg
    3003c160      0 none ,g      *
FILE=sprintf
    3003c16c 3003c19b      30
_sprintf
    3003c16c      30 func ,g      *
```

**Name**

`rb` - Resets breakpoints.

**Syntax**

`rb [pname break1 ... break5 [-t|-s]]`

**Description**

The `rb` subcommand resets the currently set breakpoints. If no parameter is specified, `rb` resets all the currently set breakpoints. The following parameters can be specified:

*pname*: Specifies the name of the program for which breakpoints are to be set.

*break1* to *break5*: Specifies breakpoints (relative address in a program).

`-t`: Indicates that the specified program name is a program name of a task.

`-s`: Indicates that the specified program name is a subprogram name.

Note: When the `-t` and `-s` options are not specified, a search will be conducted in the task and subprogram order.

**Result**

The following message appears when a breakpoint reset is completed normally:

```
break point reset
```

```
name = program-name raddr = program-internal-relative-address object = machine-language-instruction-pattern
```

An error message appears if breakpoint setup is being performed by another terminal.

**Notes**

- This subcommand can be used when the `-debug` option is specified to start `svdebug`. If the subcommand is used in other situations, an error occurs.
- If any task is halted at a breakpoint, that breakpoint cannot be reset.



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

rd - Display registers.

### Syntax

rd [-f | -h]

### Description

The `rd` subcommand displays the contents of registers prevailing at a breakpoint. If no parameter is specified, the floating-point registers are excluded from the resulting on-screen information.

The following parameters can be specified:

- f: Displays the contents of floating-point registers as real numbers.
- h: Displays the contents of floating-point registers in hexadecimal format.

### Result

OK (0) : Normal termination.

`rd` displays the contents of the registers as follows.

When abnormally terminated, `rd` displays an error message.

```
SR =0x***** PC =0x***** GBR =0x***** PR =0x*****
MACH=0x***** MACL=0x***** FPUL=0x***** FPSCR=0x*****
R0 =0x***** R1 =0x***** R2 =0x***** R3 =0x*****
R4 =0x***** R5 =0x***** R6 =0x***** R7 =0x*****
R8 =0x***** R9 =0x***** R10 =0x***** R11 =0x*****
R12 =0x***** R13 =0x***** R14 =0x***** R15 =0x*****
```

When the `-f` option is specified, `rd` displays the contents of floating-point registers in the form of a real number as indicated below:

```
FR0 =** .*****E*** FR1 =** .*****E*** FR2 =** .*****E*** FR3 =** .*****E***
FR4 =** .*****E*** FR5 =** .*****E*** FR6 =** .*****E*** FR7 =** .*****E***
FR8 =** .*****E*** FR9 =** .*****E*** FR10=** .*****E*** FR11=** .*****E***
FR12=** .*****E*** FR13=** .*****E*** FR14=** .*****E*** FR15=** .*****E***
XF0 =** .*****E*** XF1 =** .*****E*** XF2 =** .*****E*** XF5 =** .*****E***
XF4 =** .*****E*** XF5 =** .*****E*** XF6 =** .*****E*** XF7 =** .*****E***
XF8 =** .*****E*** XF9 =** .*****E*** XF10=** .*****E*** XF11=** .*****E***
XF12=** .*****E*** XF13=** .*****E*** XF14=** .*****E*** XF15=** .*****E***
DR0 =** .*****E*** DR2 =** .*****E***
DR4 =** .*****E*** DR6 =** .*****E***
DR8 =** .*****E*** DR10=** .*****E***
DR12=** .*****E*** DR14=** .*****E***
XD0 =** .*****E*** XD2 =** .*****E***
XD4 =** .*****E*** XD6 =** .*****E***
XD8 =** .*****E*** XD10=** .*****E***
XD12=** .*****E*** XD14=** .*****E***
```

When the `-h` option is specified, `rd` displays the contents of floating-point registers in hexadecimal notation.

```
FR0 =0x***** FR1 =0x***** FR2 =0x***** FR3 =0x*****
FR4 =0x***** FR5 =0x***** FR6 =0x***** FR7 =0x*****
FR8 =0x***** FR9 =0x***** FR10=0x***** FR11=0x*****
FR12=0x***** FR13=0x***** FR14=0x***** FR15=0x*****
XF0 =0x***** XF1 =0x***** XF2 =0x***** XF5 =0x*****
XF4 =0x***** XF5 =0x***** XF6 =0x***** XF7 =0x*****
XF8 =0x***** XF9 =0x***** XF10=0x***** XF11=0x*****
XF12=0x***** XF13=0x***** XF14=0x***** XF15=0x*****
DR0 =0x***** 0x***** DR2 =0x***** 0x*****
DR4 =0x***** 0x***** DR6 =0x***** 0x*****
DR8 =0x***** 0x***** DR10=0x***** 0x*****
DR12=0x***** 0x***** DR14=0x***** 0x*****
XD0 =0x***** 0x***** XD2 =0x***** 0x*****
XD4 =0x***** 0x***** XD6 =0x***** 0x*****
XD8 =0x***** 0x***** XD10=0x***** 0x*****
XD12=0x***** 0x***** XD14=0x***** 0x*****
```

### Register descriptions

**SR:** Status register.

**PC:** Program counter.

**GBR:** Global base register. This register is for storing the base address of a GBR indirect with displacement or GBR indirect with index for use in addressing.

**PR:** Procedure register used to call a subroutine. If the executed program is at the end of subroutine calling, this register stores a return address.

**MACH:** System register (sum-of-products upper-level register) used to store the addition value of the MAC instruction (sum-of-products operation) and the results of the execution of MAC and MUL instructions. When the operation result is a 64-bit value, this register stores the most significant 32 bits. When the operation result is a 32-bit value, this register stores 32 bits.

**MACL:** System register (sum-of-products lower-level register). When the operation result is a 64-bit value, this register stores the least significant 32 bits.

**FPUL:** Floating-point communication register.

**FPSCR:** Floating-point status and control register.

**R0-R15:** General-purpose registers (the R15 is used as a stack pointer).

**FR0-FR15:** Single-precision floating-point registers. This register represents the `FPRxx_BANK0` value when `FPSCR.PR` (the 19th bit value of bits 31-0) = 0 or the `FPRxx_BANK1` value when `FPSCR.PR` = 1.

**XF0-XF15:** Single-precision floating-point extended registers. This register represents the `FPRxx_BANK1` value when `FPSCR.PR` (the 19th bit value of bits 31-0) = 0 or the `FPRxx_BANK0` value when `FPSCR.PR` = 1.

**DR0-DR15:** Double-precision floating-point registers. This register represents the `FPRxx_BANK0` value when `FPSCR.PR` (the 19th bit value of bits 31-0) = 0 or the `FPRxx_BANK1` value when `FPSCR.PR` = 1.

**XD0-XD15:** Double-precision floating-point extended registers. This register represents the `FPRxx_BANK1` value when `FPSCR.PR` (the 19th bit value of bits 31-0) = 0 or the `FPRxx_BANK0` value when `FPSCR.PR` = 1.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Notes

- This subcommand can be used when the `-debug` option is specified to start `svdebug`. If the subcommand is used in other situations, an error occurs.
- Even when a real-number display mode is selected, this subcommand displays the contents of registers in hexadecimal format if the floating-point data is as indicated below. The subcommand also displays the associated character string after the hexadecimal number display.

| Floating-point data       | Character string | Display example       |                                 |
|---------------------------|------------------|-----------------------|---------------------------------|
|                           |                  | Single precision      | Double precision                |
| Non-numeric               | Na               | FR0 = 0x7fffffff:Na   | DR0 = 0xfff 00000 0x00000001:Na |
| Infinite                  | In               | FR0 = 0x7f800000:In   | DR0 = 0xfff 00000 0x00000000:In |
| Maximum expressible value | Ma               | FR0 = 0x7f7fffffff:Ma | DR0 = 0x7fefffff 0xffffffff:Ma  |
| Minimum expressible value | Mi               | FR0 = 0xff7fffffff:Mi | DR0 = 0xffefffff 0xffffffff:Mi  |

**Name**

`rr` - Changes the contents of registers.

**Syntax**

```
rr  
register : rx  
data : datax
```

Note: The underlined portions are to be entered by the user.

**Description**

The `rr` subcommand changes the contents of a register when it is halted at a breakpoint.

*rx*: Specifies the abbreviated name of the register whose contents are to be changed. Use the abbreviated register name that appears upon `rd` subcommand execution.

*datax*: Specifies the data to be changed. (Octal, decimal, hexadecimal, and real-number data can be specified.) For double-precision floating-point registers, only real-number data input is possible.

**Result**

OK (0) : Normal termination.

When abnormally terminated, `rr` displays an error message.

**Notes**

- The `rr` subcommand is valid only when a task is halted at a breakpoint.
- This subcommand can be used when the `-debug` option is specified to start `svdebug`. If the subcommand is used in other situations, an error occurs.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

go - Resumes execution from a breakpoint.

### Syntax

go

### Description

The go subcommand resumes a program at an address at which the program was halted due to a breakpoint. The breakpoint resets after the program is resumed.

### Result

OK (0) : Normal termination.

When abnormally terminated, the go subcommand displays an error message.

### Notes

- The go subcommand is valid only when a task is halted at a breakpoint.
- This subcommand can be used when the -debug option is specified to start svdebug. If the subcommand is used in other situations, an error occurs.

**Name**

ld - Transfers backup file data to the S10VE memory.

**Syntax**

```
ld {-t tname}
ld {-s sname}
ld {-g gname}
ld {-a aname}
ld {-m addr, len}
ld {-T [tno]}
```

```
ld {-U [point, ent]}
```

```
ld {-S [sno]}
```

```
ld {-G [gno]}
```

```
ld {-dcm}
```

```
ld {-cm}
```

```
ld {-f fname}
```

**Description**

The ld subcommand transfers the contents of a backup file to memory in the S10VE, the file that is specified by one of the options listed below.

- t *tname*: The program specified by *tname* is downloaded. A task is created for *tname* on the S10VE. When the program specified by *tname* is a multitask program, as many tasks as are required are also created on the S10VE.
- s *sname*: The subprogram specified by *sname* is loaded.  
This option downloads the subprogram itself, and also downloads entries of the indirect link resident subprogram address table or entries of the built-in subroutine management table. When the subprogram specified by *sname* is registered as a multi-entry IRSUB, all the entries in the indirect link resident subprogram address table corresponding to the IRSUB entry points are downloaded.
- g *gname*: The global data specified by *gname* is downloaded. When the global data is indirect global data, the associated entry for the indirect link global address table is also downloaded.
- a *aname*: The contents of the split area specified by *aname* are downloaded. The split area specified by *aname* must have been defined in the GLB area and CM area.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

- m *addr, len*: Data of *len* bytes is downloaded, starting from the first address specified by *addr*.  
When the specified address range includes a non-allocated space, downloading does not take place.
- T [*tno*]: The task that has the task number specified by *tno* is deleted from or created on the S10VE. If *tno* identifies a task that has been created on the development machine, this option creates a task for *tno*. If *tno* identifies a task that has not been created on the development machine, the option deletes the task. When *tno* is omitted, a list is displayed that shows the tasks for which creation or deletion on the development machine is not reflected on the S10VE.
- U [*point, ent*]: The entry for the built-in subroutine management table specified by *point* and *ent* is downloaded. Specify *point* and *ent* as follows:
  - point*: This character string indicates the location where the built-in subroutine is incorporated. Specify CPES, IES, EAS, INS, EXS, ABS, PCKS, MODES, WDTES, or XEAS.
  - ent*: This is an entry number from 1 to 4.  
When both *point* and *ent* are omitted, a list is displayed that shows the built-in subroutine management table entries for which registration or deletion on the development machine is not reflected on the S10VE.
- S [*sno*]: The indirect link resident subprogram address table entry specified by *sno* is downloaded. When *sno* is omitted, a list is displayed that shows the indirect link resident subprogram address tables for which registration or deletion on the development machine is not reflected on the S10VE.
- G [*gno*]: The indirect link global address table specified by *gno* is downloaded. When *gno* is omitted, a list is displayed that shows the indirect link global address tables for which registration or deletion on the development machine is not reflected on the S10VE.
- dcm: Only areas allocated in the DCM are downloaded. Because this is an extension option, it cannot be used with the S10VE.
- cm: Only areas allocated in the CM are downloaded.
- f *fname*: The contents of the file specified by *fname* are downloaded. Only the name of the file output by the *sv* subcommand can be specified by *fname*.

**Result**

When resources are downloaded by `ld`, the result is displayed upon completion of downloading. The following paragraphs explain the results of `ld` with regard to each option.

**(1) -t *tname* (downloading an individual program)**

The first line shows the main unit address. The second and subsequent lines show the TCB addresses at which a download was performed due to task generation or deletion. (When the old task exists, the first line shows the old TCB address, the second line shows the main unit address, and the third line shows the new TCB address.)

For a multitask, the display shows all the TCB addresses for downloading.

```
address : 0x*****-0x*****
address : 0x*****-0x*****
```

**(2) -s *sname* (downloading an individual subprogram)**

The result is displayed in a different way according to the type of the downloaded subprogram.

**● IRSUB**

The range of addresses where the IRSUB was downloaded is displayed in the format below. The first line shows the address of the subprogram, while the second line shows the address of the indirect link resident subprogram address table.

When the IRSUB has multiple entries, the addresses of all indirect link resident subprogram address tables to be downloaded are displayed.

```
address : 0x*****-0x*****
address : 0x*****-0x*****
```

**● Built-in subroutine**

The range of addresses where the subroutine was downloaded is displayed together with the entry for the built-in subroutine management table, in the following format.

`address` is the address of the downloaded built-in subroutine, while `point,ent` is the include point representing the downloaded entry for the built-in subroutine management table and the entry number.

```
address : 0x*****-0x*****
point,ent : POINT,N
```

For *POINT*, one of the following character strings, each of which represents an include point, is displayed: CPES, IES, EAS, INS, EXS, ABS, PCKS, MODES, WDTES, and XEAS. For *N*, an entry number from 1 to 4 is displayed.



(3) `-g gname` (downloading individual global data)

The range of addresses where global data was downloaded is displayed in the format below. When the global data is indirect link global data, the address of the indirect link global address table is also displayed.

When the global data specified by *gname* is registered as multiple indirect link global data items, the addresses of all indirect link global address tables are displayed.

```
address : 0x*****-0x*****
```

(4) `-a aname` (downloading an individual split area)

The range of addresses where the split area was downloaded is displayed in the following format:

```
address : 0x*****-0x*****
```

(5) `-m addr, len` (address-based downloading)

The range of addresses where downloading was performed is displayed in the following format:

```
address : 0x*****-0x*****
```

(6) `-T tno` (reflecting task creation or deletion in memory in the controller)

- When task generation or deletion is reflected in the controller memory

The display shows the TCB addresses at which a download was performed due to task generation/deletion.

```
address : 0x*****-0x*****
```

- When a list of the tasks requiring reflection of their memory images in memory in the controller whenever necessary is requested for display:

When only the `-T` option is specified, a list is displayed in the format below. This list shows the tasks that need to be reflected in memory in the controller.

| TN        | TNAME        | PNAME        | STATUS      |
|-----------|--------------|--------------|-------------|
| <i>tn</i> | <i>tname</i> | <i>pname</i> | <i>stat</i> |

*tn*, *tname*, *pname*, and *stat* in the above list represent the task number, task name, program name, and management state of the task, respectively. Table 2-13 explains the management states. When different task numbers are used for a task in the unmatch management state on the development machine and on the S10VE, the task number on the development machine is displayed.

Table 2-13 Management States of Resources

| No. | Management state | Description                                                                                                                                             |
|-----|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | non-exist        | Registered in neither the development machine nor the S10VE.                                                                                            |
| 2   | not-build        | Loaded by a subprogram but not built.                                                                                                                   |
| 3   | defined-POC      | Registered in the development machine only.                                                                                                             |
| 4   | defined          | Registered in both the development machine and S10VE.                                                                                                   |
| 5   | defined-CON      | Registered in the S10VE only.<br>Deleted from the development machine after downloading into the S10VE.                                                 |
| 6   | unmatch          | Registered in both the development machine and S10VE but mismatched.<br>Deleted or reregistered at the development machine but not downloaded.          |
| 7   | non-exist2       | Build information in the S10VE memory was deleted by the subprogram in the defined-CON state, but has not been downloaded from the development machine. |
| 8   | defined-CON2     | Build information was downloaded by the subprogram in the defined-CON state from the development machine, but remains in the S10VE memory.              |

(7) `-U point, ent` (downloading a built-in subroutine management table)

- When a built-in subroutine management table was downloaded:

The entry of the updated built-in subroutine management table is displayed in the following format:

```
address : 0x*****-0x*****
```

- When a list of the built-in subroutine management tables requiring reflection in memory in the S10VE is requested for display:

When only the `-U` option is specified, a list is displayed in the format below, the list that shows the built-in subroutine management tables that need to be reflected in memory in the controller.

```
POINT    ENT      SUBNAME      STATUS
pnt      eno      subname      stat
```

*pnt*, *eno*, *subname*, and *stat* in the above list represent an include point, entry number, subprogram name, and management state of the built-in subroutine management table entry, respectively. Table 2-13 explains the management states.

As the management state of the built-in subroutine management table entry, defined-POC, defined-CON, not-build, non-exist2, or unmatch is displayed.

When different include points are used for a built-in subroutine management table entry in the unmatch state on the development machine and that on the controller, the include point on the development machine is displayed.

(8) -S *sno* (downloading an indirect link resident subprogram address table)

- When an indirect link resident subprogram address table was downloaded:  
The address of the downloaded indirect link resident subprogram address table is displayed in the following format:

```
address : 0x*****-0x*****
```

- When a list of the indirect link resident subprogram address tables requiring reflection in memory in the S10VE is requested for display:  
When only the -S option is specified, a list is displayed in the format below, the list that shows the indirect link resident subprogram address tables that need to be reflected in memory in the S10VE.

| IRSBNO     | ENTNAME        | SUBNAME        | STATUS      |
|------------|----------------|----------------|-------------|
| <i>sno</i> | <i>entname</i> | <i>subname</i> | <i>stat</i> |

*sno*, *entname*, *subname*, and *stat* in the above list represent an IRSUB number, entry name, subprogram name, and management state of the IRSUBT entry. Table 2-13 explains the management states.

When different IRSUB numbers are used for an IRSUBT entry in the unmatched state on the development machine and that on the S10VE, the IRSUB number on the development machine is displayed. When different subprogram names are used, the subprogram name on the development machine is also used.

(9) -G *gno* (downloading an indirect link global address table)

- When an indirect link global address table was downloaded:  
The address of the downloaded indirect link global address table (IRGLBT) is displayed in the following format:

```
address : 0x*****-0x*****
```

- When a list of the indirect link global address tables requiring reflection in memory in the S10VE is requested for display:

When only the `-G` option is specified, a list is displayed in the format below, the list that shows the indirect link global address tables that need to be reflected in memory in the S10VE.

| IRGLBNO    | ENTNAME        | SNAME        | STATUS      |
|------------|----------------|--------------|-------------|
| <i>gno</i> | <i>entname</i> | <i>sname</i> | <i>stat</i> |

*gno*, *entname*, *sname*, and *stat* in the above list represent an indirect link global number, entry name, secondary partition area, and management state of the indirect link global address table entry, respectively. Table 2-13 explains the management states.

When different indirect link global numbers are used for an indirect link global address table entry in the unmatched state on the development machine and that on the S10VE, the indirect link global number on the development machine is displayed.

When different secondary partition area names are used, the secondary partition area name on the secondary partition area is also used.

- (10) `-cm` (downloading CM)

The range of addresses where downloading was performed is displayed in the following format:

```
address : 0x*****-0x*****
```

Downloading to the CM is enabled only when the S10VE is in the STOP state.

Only the CM backup data of the execution site is downloaded to the CM.

- (11) `-f fname` (downloading from the specified file)

The range of addresses where downloading was performed is displayed in the following format:

```
address : 0x*****-0x*****
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Notes

- Before replacing a resource, abort all tasks that reference the resource or take a similar action, so that the replacement will not cause a problem.
- When a resource is downloaded with the `-a` or `-m` option, be sure to download the management table with the `-T`, `-U`, `-S`, or `-G` option and also create or delete tasks.
- If a resource or area containing no backup file is specified without specifying the file name with the `-f` option, the system displays an error message and terminates the subcommand.
- If the failure recovery process is skipped, the individual loading function of the `ld` subcommand cannot be exercised until batch loading is performed by the `svrpl` command.
- Programs and subprograms for which a breakpoint is set cannot be downloaded.

**Name**

sv - Transfers S10VE memory data to a backup file.

**Syntax**

```
sv {-t tname [-f fname]}
sv {-s sname [-f fname]}
sv {-g gname [-f fname]}
sv {-a aname [-f fname]}
sv {-m addr, len [-f fname]}
```

**Description**

The sv subcommand transfers the contents of memory in the S10VE to a backup file as specified by the following options.

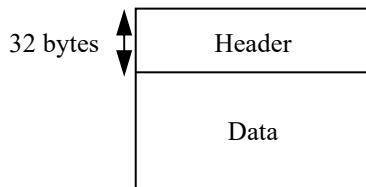
- t *tname*: The text and data sections of the program specified by *tname* are transferred.
- s *sname*: The text and data sections of the subprogram specified by *sname* are transferred.
- g *gname*: The contents of the secondary partition area specified by *gname* are transferred.
- a *aname*: The contents of the split area specified by *aname* are transferred. The split area specified by *aname* must have been defined in the GLB area and CM area.
- m *addr, len*: Data is transferred for the number of bytes specified by *len*, starting from the first address specified by *addr*.  
When the specified address range includes a non-mapped space, a transfer does not take place.
- f *fname*: Data is transferred to the file specified by *fname*.  
When this option is omitted, data is transferred to the backup file.  
When an error is detected during saving, the specified file is deleted and this subcommand is terminated.

**Result**

The range of addresses where data was transferred is displayed in the following format:

```
address : 0x*****-0x*****
```

When transferring to a file other than the backup file, make sure that the file is in the following format:



32-byte header is followed by binary data.

The header consists of the following character strings:

```
svdebugΔ*****Δ*****\0\0...\0\0
           First address      Size
```

\*\*\*\*\* is an eight-digit hexadecimal number.

\0 is NULL (0).

Δ is a blank character (0X20).

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Notes

- When transferring CM data to the backup file, only the data of the CM allocated to the local site can be transferred.
- If an area that does not contain the backup file is specified without specifying a file name by the `-f` option, an error message is output and the subcommand is terminated.

**Name**

cm - Compares the contents of a backup file with those of the memory in the S10VE.

**Syntax**

```
cm {-t tname}
cm {-s sname}
cm {-g gname}
cm {-a aname}
cm {-m addr, len}
cm {-f fname}
```

**Description**

The `cm` subcommand compares the contents of a backup file with those of memory in the S10VE. Use the following options to specify the comparison target.

- t *tname*: The text and data sections of the program specified by *tname* are compared.
- s *sname*: The text and data sections of the subprogram specified by *sname* are compared.
- g *gname*: The contents of the secondary partition area specified by *gname* are compared.
- a *aname*: The contents of the split area specified by *aname* are compared. The split area specified by *aname* must have been defined in the GLB area and CM area.
- m *addr, len*: Data is compared for the number of bytes specified by *len*, starting from the first address specified by *addr*.  
When the specified address range includes a non-allocated space, a comparison does not take place.
- f *fname*: The file specified by *fname* is compared with the contents of memory. (Only the file to which an output is made by the `sv` subcommand can be specified.)

**Result**

When the comparison result is normal, the address range is displayed in the following format:

```
address : 0x*****-0x*****
++ compare OK ++
```

When a mismatch is found during comparison, the result is displayed one word (two bytes) at a time, as follows:

```
address : 0x*****-0x*****
address = 0x*****  memory data = 0x****  backup data = 0x****
```



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Notes

- CM data and backup file data can be compared only in the CM allocated to the local site.
- If an area containing no backup file is specified without specifying the file name with the `-f` option, the system displays an error message and terminates the subcommand.

**Name**

`dr` - Enables DHP logging.

**Syntax**

`dr`

**Description**

The `dr` subcommand starts the `svdhp` command to enable DHP logging. For details on the `dr` subcommand, see the specifications for the `-on` option of the `svdhp` command.

**Name**

`ds` - Disables DHP logging.

**Syntax**

`ds`

**Description**

The `ds` subcommand starts the `svdhp` command to disable DHP logging. For details on the `ds` subcommand, see the specifications for the `-off` option of the `svdhp` command.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

svdhp - Displays the DHP.

### Syntax

svdhp [-u *site*] [+*count*] [-on|-off] [-o *fname*] [-f *fname*] [-all [*fname*]]

### Description

The `svdhp` subcommand starts the `svdhp` command to display the DHP.

For details on the `svdhp` subcommand, see the section on the `svdhp` command.

### Name

svadm - Displays a resource name for an address.

### Syntax

svadm [*addr*] [-u *site*]

### Description

The `svadm` subcommand starts the `svadm` command to display the name and other information about a specified logical address.

For details on the `svadm` subcommand, see the section on the `svadm` command.

**Name**

`si` - Initializes the stack.

**Syntax**

```
si tn1[-tn2][, data]
si tname[, data]
```

**Description**

The `si` subcommand initializes the stack of the specified task with a fixed pattern. Specify the following parameters:

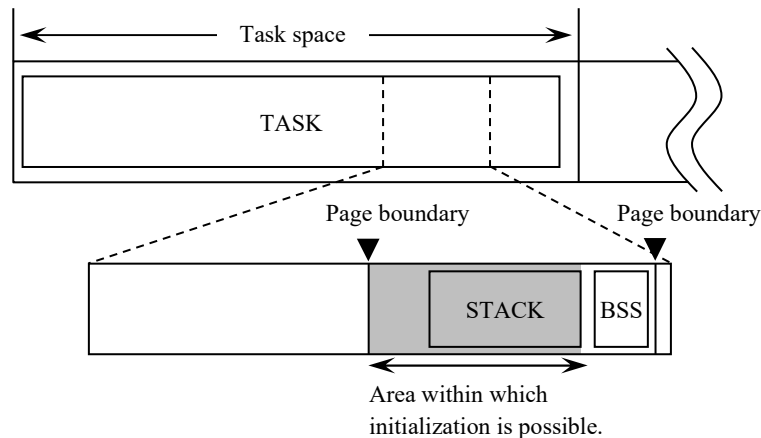
*tn1*: First task number (1 to the maximum task number)  
*tn2*: Last task number (*tn1* to the maximum task number)  
*tname*: Task name  
*data*: Initialization data (0 to 9, a to f)

**Result**

OK (0) : Normal termination

**Notes**

- When nothing is specified in *data*, the stack is initialized with all *f* values.
- The task for which initialization is to be performed must be in the DORMANT state.
- The stack of the specified task can be initialized only within the range of the first address of the page on which the stack is present to the last address of the stack. (See the following figure.)



- When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:).  
 If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn1[-tn2] [,data] or tname [,data]
:
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`sp` - Displays the amount of stack use.

### Syntax

```
sp tn1[-tn2][, data]
sp tname[, data]
```

### Description

The `sp` subcommand displays the size of the stack used by the specified task. Specify the following parameters:

- tn1*: First task number (1 to the maximum task number)
- tn2*: Last task number (*tn1* to the maximum task number)
- tname*: Task name
- data*: Check pattern (0 to 9, a to f)

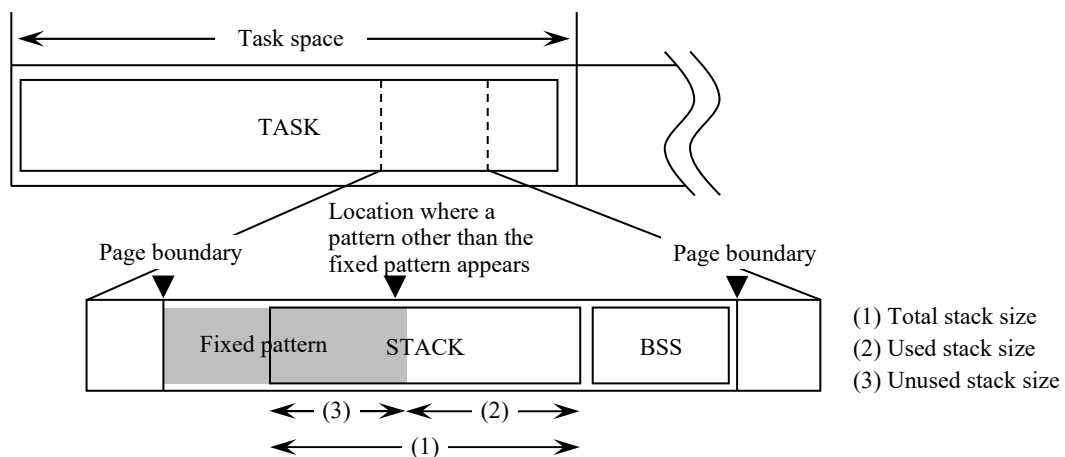
### Result

The size of the stack used by the task is displayed in the following format:

| <code>tn=***</code> | <code>total:*****bytes</code> | <code>use:*****bytes</code> | <code>rest:*****bytes</code> |
|---------------------|-------------------------------|-----------------------------|------------------------------|
| Task number         | Total stack size              | Used stack size             | Unused stack size            |

### Notes

- The check pattern specified in *data* must be the same as the initialization data specified in the `si` subcommand. When the check pattern parameter (*data*) is omitted, `0xf` is assumed.
- The used stack size is calculated from the address at which a pattern other than the check pattern specified in the *data* parameter appears in the page area in the stack used by the specified task. For this reason, if the pattern at the beginning of the stack is the same as the check pattern, the used stack size cannot be displayed correctly.
- The following figure shows the relationship between the information displayed by `sp` and the task operation space.



- When this subcommand is started with no parameters specified or with invalid arguments specified, the following message is displayed to prompt for parameters. Enter correct parameters after the colon (:).  
If you enter `e` or press the **Enter** key in this state, the subcommand process terminates.

```
input tn[-tn2] [-data] or tname [-data]
:
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`ps` - Starts displaying debug statements.

### Syntax

`ps`

### Description

The `ps` subcommand instructs that the messages output by `rs_printf()` within a program should start appearing on the terminal display.

Debug statements output prior to the execution of the `ps` subcommand do not appear on the display.

(For more information about `rs_printf()`, see APPENDIX B LIBRARIES.)

### Result

The debug statement display sequence starts after normal process termination.

### Notes

If the buffer for debug statement storage is not sufficient, the subcommand might fail to output debug statements.

### Name

`pe` - Stops displaying debug statements.

### Syntax

`pe`

### Description

The `pe` subcommand stops the messages output by `rs_printf()` within a program from appearing on the terminal display. (For more information about `rs_printf()`, see APPENDIX B LIBRARIES.)

### Result

The debug statement display sequence stops after normal process termination.

**Name**

`ver` - Displays the version of the CPMS.

**Syntax**

`ver [-m|-s]`

**Description**

The `ver` subcommand displays the version of the CPMS and KROM. The following options are provided.

- m: Displays the version information in the S10VE memory.
- s: Displays the version information in the development machine.

When these options are omitted, both of these version information items are displayed.

**Result**

The version of the CPMS and MP farm appears in the following format upon normal process termination.

```
Secondary version:
CPMS      Ver.** Rev.** XXXX
Memory version:
CPMS      Ver.** Rev.** XXXX
KROM                      Rev.**
```

**Notes**

- When the version information in the development machine is displayed, the MP firmware version is not displayed.
- Only the character string indicating the revision is displayed for the MP firmware version.
- An optional character string *XXXX* (such as *-A*) is displayed following the CPMS version. If no character string is set, null is displayed.
- When the CPMS has not been downloaded to the S10VE, `ver.-- Rev.--` is displayed.



## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`lbr` - Sets and displays breakpoints in ladder programs.

### Syntax

`lbr addr [tn]`

### Description

The `lbr` subcommand sets a breakpoint in a ladder program. Only a single breakpoint can be set. For `addr`, specify the address of the breakpoint of the LADDER space.

Specify `tn` (a task number) when setting a breakpoint only for the ladder program run from a specific task. If `tn` is omitted or is set to 0, a breakpoint is set for ladder programs run from all tasks.

When the breakpoint is reached, the ladder program suspends the processing immediately before running the breakpoint instruction. While the ladder program is breaking, execution of all ladder programs is inhibited.

If `addr` and `tn` are omitted, the currently set breakpoint and break status are displayed.

### Result

When a breakpoint has been successfully set, the following message appears:

```
break point set
addr = 0xXXXXXXXX tn = xxx
```

`0xXXXXXXXX`: Address at which breakpoint is set

`xxx`: Task number of the task that detects that the breakpoint has been reached

The currently set breakpoint is displayed as follows:

```
break point
addr = 0xXXXXXXXX tn = xxx
```

`0xXXXXXXXX`: Address at which breakpoint is set

`xxx`: Task number of the task that detects that the breakpoint has been reached

When a breaking ladder program is present, the following message is displayed:

```
break on 0xXXXXXXXX (tn = xxx)
0xXXXXXXXX 0xYYYYYYYY
```

`0xXXXXXXXX`: Address at which break is present

`xxx`: Task number of the task that detects that the breakpoint has been reached

`0xYYYYYYYY`: Next instruction to be run

**Name**

`lrb` - Resets breakpoints of ladder programs.

**Syntax**

`lrb`

**Description**

The `lrb` subcommand resets the breakpoint setting in the ladder programs.

**Result**

The following message appears when a breakpoint reset is completed normally:

```
break point reset  
addr = 0xXXXXXXXX tn = xxx
```

`0xXXXXXXXX`: Address at which breakpoint is set

`xxx`: Task number of the task that detects that the breakpoint has been reached

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`lrd` - Displays the contents of the registers of ladder processors.

### Syntax

`lrd [-h]`

### Description

The `lrd` subcommand displays the contents of the registers of breaking ladder programs. When the `-h` option is specified, floating-point registers are shown as hexadecimal numbers.

### Result

```
R0 =0x***** R1 =0x***** R2 =0x***** R3 =0x*****
R4 =0x***** R5 =0x***** R6 =0x***** R7 =0x*****
R8 =0x***** R9 =0x***** R10 =0x***** R11 =0x*****
R12 =0x***** R13 =0x***** R14 =0x***** R15 =0x*****
PC =0x***** SP =0x*****
FR0 =**.*E*** FR1 =**.*E*** FR2 =**.*E*** FR3 =**.*E***
FR4 =**.*E*** FR5 =**.*E*** FR6 =**.*E*** FR7 =**.*E***
FR8 =**.*E*** FR9 =**.*E*** FR10=**.*E*** FR11=**.*E***
FR12=**.*E*** FR13=**.*E*** FR14=**.*E*** FR15=**.*E***
FPUL=0x***** FPSCR=0x*****
DSEG0 =0x***** DSEG1 =0x***** DSEG2 =0x***** DSEG3 =0x*****
DSEG4 =0x***** DSEG5 =0x***** DSEG6 =0x***** DSEG7 =0x*****
```

The following floating-point data is converted to hexadecimal numbers, and the converted data is then displayed. Furthermore, a corresponding character string is shown following each hexadecimal number.

| Floating-point data       | Character string | Display example       |
|---------------------------|------------------|-----------------------|
| Non-numeric               | Na               | FR0 = 0x7fffffff:Na   |
| Infinite                  | In               | FR0 = 0x7f800000:In   |
| Maximum expressible value | Ma               | FR0 = 0x7f7fffffff:Ma |
| Minimum expressible value | Mi               | FR0 = 0xff7fffffff:Mi |

When the `-h` option is specified, `rd` displays the contents of floating-point registers in hexadecimal notation.

```
FR0 =0x***** FR1 =0x***** FR2 =0x***** FR3 =0x*****
FR4 =0x***** FR5 =0x***** FR6 =0x***** FR7 =0x*****
FR8 =0x***** FR9 =0x***** FR10=0x***** FR11=0x*****
FR12=0x***** FR13=0x***** FR14=0x***** FR15=0x*****
```

**Name**

`lrr` - Changes the registers of the ladder processor.

**Syntax**

```
lrr  
register : reg  
data : datax
```

Note: The underlined portions are to be entered by the user.

**Description**

The `lrr` subcommand rewrites the contents of the registers of breaking ladder programs.

*reg*: Specifies the name of the target register.

Specify the name shown by the `lrd` subcommand for the register name.

*datax*: Specifies the data to be changed. Octal, decimal, or hexadecimal data can be specified.

Specify a real number for floating-point registers.

**Result**

OK (0) : Normal termination.

NG ( $\neq 0$ ) : Macro error

The macro return code is shown in ( $\neq 0$ ).

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

lgo - Resumes execution of ladder programs.

### Syntax

lgo

### Description

The lgo subcommand resumes execution of breaking ladder programs. Breakpoints are not reset even after execution resumes.

### Result

OK (0) : Normal termination.

no break: There is no breaking ladder program.

### Name

s - Runs steps of ladder programs.

### Syntax

s

### Description

The s subcommand executes steps of breaking ladder programs. When a ladder program is terminated by the execution of steps, ladder programs that have been inhibited by breaking resume execution.

### Result

Upon successful completion of step execution, the following message appears:

0xXXXXXXXX 0xYYYYYYYY

0xXXXXXXXX: Address of the instruction to be executed next

0xYYYYYYYY: Instruction to be executed next

In the case of abnormal termination, the following message appears:

no break: There is no breaking ladder program.

**Name**

`help` - Displays a list of subcommands.

**Syntax**

`help`

**Description**

The `help` subcommand displays a list of `svdebug` subcommands.

The `help` subcommands list the subcommand names together with an outline of their functions, in the following format:

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

| <Sub Command> | <Function>                              |
|---------------|-----------------------------------------|
| qu            | ...task queue                           |
| ab            | ...task abort                           |
| re            | ...task release                         |
| ta            | ...task status                          |
| su            | ...task suspend                         |
| rs            | ...task resume                          |
| tm            | ...task timer request                   |
| ct            | ...task cancel timer                    |
| sht           | ...task show timer                      |
| md            | ...memory print/patch appointed address |
| sd            | ...memory print/patch appointed name    |
| mcp           | ...memory copy                          |
| mmv           | ...memory move                          |
| mf            | ...memory fill                          |
| bs            | ...bit data set                         |
| bg            | ...bit data get                         |
| el            | ...system error display                 |
| ss            | ...system status display                |
| st            | ...set timer                            |
| gt            | ...get timer                            |
| br            | ...break point set                      |
| stickybr      | ...sticky break point set               |
| rb            | ...break point reset                    |
| rd            | ...register print                       |
| rr            | ...register set                         |
| go            | ...break point restart                  |
| ld            | ...memory load                          |
| sv            | ...memory save                          |
| cm            | ...memory compare                       |
| dr            | ...DHP regist start                     |
| ds            | ...DHP regist stop                      |
| svdhp         | ...DHP data display                     |
| svadm         | ...address -> sarea name                |
| si            | ...stack initial                        |
| sp            | ...stack print                          |
| ps            | ...debug message print start            |
| pe            | ...debug message print end              |
| ver           | ...CPMS version print                   |
| q             | ...svdebug end                          |
| !             | ...execute external command             |
| help          | ...command menu display                 |
| lbr           | ...ladder break point set               |
| lrb           | ...ladder break point reset             |
| lrd           | ...ladder register print                |
| lrr           | ...ladder register set                  |
| lgo           | ...ladder break point restart           |
| s             | ...ladder step execution                |

### Name

q - Terminates the debugger.

### Syntax

q

### Description

The q subcommand terminates the debugger.

However, if a breakpoint is set, the subcommand displays it and waits for key input.

### Notes

If the message `Breakpoint is used` appears, reset it by using the `go` subcommand if execution is halted at a breakpoint or by using the `rb` subcommand if execution is not halted at a breakpoint. Then, reissue the `q` subcommand.

### Name

! - Executes a command on the development machine at the time of `svdebug` execution.

### Syntax

! *command-on-development-machine*

### Description

The ! subcommand runs the character strings after the exclamation point (!) as commands on the development machine.



**Name**

`svelog` - Outputs error log information.

**Syntax**

```
svelog [-u site] [-f {s|m|l}] [-logno] [+case] [-d fname] [-o fname]
```

Error log for one screen is displayed.

```
{ p }
{ - }
{ ±nl }
{ n }
{ (blank) }
{ q }
```

**Description**

The `svelog` command reads error logs from the error log buffer in the S10VE, and displays the error logs. The following options can be specified:

- u *site*: Specify the name of the site to be handled. When this option is omitted, the site set in the `RSSITE` environment variable is used.
- f {s|m|l}: Specify one of the following output formats for error log display.  
The following formats are available. The default format is m.
  - s: Error logs are displayed in a simplified format.
  - m: The complete error log is displayed.
  - l: Error logs are displayed together with DHP traces.
- logno: The error log that has the error log number specified by *logno* is displayed.
- +case: Specify the number of log cases to be displayed. When this option is omitted, the complete error log is displayed, starting from the most recent case.
- d *fname*: Specify the name of the file in which to store the screen operation history (operation results).  
When a file name already in use is specified, the screen operation history is added to the specified file.
- o *fname*: Specify the name of the file in which to store the error log.  
When a file name already in use is specified, the file is deleted and a new file is created.

### Notes on use

- `svelog` is operable when the CPU is in the RUN or STOP state.
- When the log number specified by `-logno` is smaller than the log number of the most recent log, the most recent log is displayed.
- When both `-logno` and `+case` are specified, error logs are displayed by the number of cases specified by `+case`, starting from the case that has the log number specified by `logno`.
- The default format is `-f m` in the S10VE.

### Termination codes

The `svelog` command returns one of the following termination codes:

- 0: Normal termination
- 1: Abnormal termination
- 2: Communication error
- 3: Suspension by pressing **Ctrl+C**

**Name**

svdhp - Displays DHP traces.

**Syntax**

```
svdhp [-u site] [+count] [-on|-off|-stat] [-d fname] [-o fname] [-all [fname] | -f fname] [-freeze]
```

One screen of DHP trace information is displayed.

```
{ p }
{ - }
{ ±nl }
{ n }
{ (blank) }
{ q }
```

**Description**

The `svdhp` command displays DHP traces stored in the DHP trace buffer in the S10VE sequentially from the most recent DHP trace. The following options can be specified:

- u *site*: Specify the name of the site to be handled. When this option is omitted, the site set in the `RSSITE` environment variable is used.
- +*count*: The trace specified by count is displayed.  
When this option is omitted, all traces are displayed.
- on: DHP logging is enabled.
- off: DHP logging is disabled.
- stat: Displays the DHP recording mode. If recording is permitted, DHP ON is displayed. If recording is prohibited, DHP OFF is displayed.
- all [*fname*]: Simultaneously fetches and displays the DHP log of the CP and HP.  
When this option is specified, CP data and HP data are not displayed but are output to different files. File names (the file name specified by `fname_cp.txt` and `fname_hp.txt`) are output. File names can be omitted. When file names are omitted, `CP-site-name.txt` and `HP-site-name.txt` are used respectively as file names.  
This option cannot be specified together with the `-f` option or `-freeze` option.
- d *fname*: Specify the name of the file in which to store the screen operation history (operation results). When a file name already in use is specified, the screen display is added to the file.
- o *fname*: Specify the name of the file in which to store the displayed DHP trace information.  
When a file name already in use is specified, the file is deleted and a new file is created.
- f *fname*: Specifies the DHP log input file name.  
The input file is acceptable only when it stores a DHP log in a GLB with the `dhpread` macro and stores the GLB DHP log by using the `sv` subcommand of the debugger. This option cannot be specified together with the `-all` option or `-freeze` option.

`-freeze`: Collects DHP trace information without occupying CPU time. Because the recording mode becomes the prohibition mode while DHP trace information is being collected by this option, DHP trace information that is being collected is not recorded. After the DHP trace information has been collected, DHP recording is resumed when the recording mode before collection was the permission mode. If the recording mode was the prohibition mode, DHP recording does not start. Use the `-stat` option to check the DHP recording mode after the DHP trace information has been collected. This option cannot be specified together with the `-all` option or the `-f` option.

The following display commands are supported to control DHP trace display.

`p`, (blank): The next page is displayed.

`-`: The previous page is displayed.

`±nl`: DHP trace display starts from the line specified by `nl` before (when a `-` is specified) or after (when a `+` is specified) the current line.

`n`: DHP trace display starts from the *n*th line.

`q`: DHP trace display is terminated.

`svdhp` displays traces in the following format:

```

Debugging helper trace list      [XXXX]   Tue Feb 13 15:37:05 2018
                                (7)

Processor type = CP
                                (8)

  DHP TIME      EVENT          TN  LV  DATA1    DATA2    DATA3    DATA4    DATA5
  nnnn tt.tttttt ssssssssssss xxx xx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx
  (1)    (2)          (3)          (5) (6)                                (4)

```

(1) DHP trace display number

(2) Time of tracing

tt.tttttt

| |

Seconds Output to the nearest microsecond

(3) Trace point type

(4) Trace data (hexadecimal)

(5) Task number

(6) Priority level

(7) Site name or the file name specified by the `-f` option

(8) Processor type

#### Notes on use

- The `svdhp` command works only while the CPU is in the RUN state.
- When `-on` or `-off` is specified, DHP traces are not displayed.

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Termination codes

The `svdhp` command returns one of the following termination codes:

- 0: Normal termination
- 1: Abnormal termination
- 2: Communication error
- 3: Suspension by pressing **Ctrl+C**

**Name**

svcpunow - Displays the PU load ratio.

**Syntax**

svcpunow [-u *site*] [-t *seconds*]

**Description**

The `svcpunow` command fetches the accumulated idle time and the time of the day from the specified site (PU), and displays the PU load ratio.

Calculation expression

PU load ratio = (measurement time - idle time) / measurement time

The following options can be specified:

- u *site*: Specify the name of the site to be handled. When this option is omitted, the site set in the `RSSITE` environment variable is used.
- t *seconds*: Specify the length of time in seconds during which the PU load ratio is measured, within the range of 1 to 3,600. The default is 1.

**Notes on use**

Even if this command is re-executed when the `svcpunow` command has already been executed, no request is accepted.

**Termination codes**

The `svcpunow` command returns one of the following termination codes:

- 0: Normal termination
- 1: Abnormal termination
- 2: Communication error
- 3: Suspension by pressing **Ctrl+C**

**Output format**

The results are output as follows:

```
2018/02/07 17:57:33 SITE=0001cp ** 1 second wait **
CPU(0001cp) load ratio = 0.06%
```

## 8. svdebug (ONLINE DEBUGGER) AND DEBUG SUPPORT COMMANDS

### Name

`svtimex` - Displays the task activity ratio.

### Syntax

```
svtimex [-u site] [tname] [-t second]  
        [tn]
```

### Description

The `svtimex` command fetches the number of task executions and the accumulated execution time in the measurement time as well as the time of day.

According to them, `timex` displays the task activity ratio.

The following options can be specified:

- u *site*: Specify the name of the site to be handled. When this option is omitted, the site set in the `RSSITE` environment variable is used.
- tn*: Specify the task number from 1 to 300 as a decimal or hexadecimal value. (To specify a hexadecimal number, prefix it with 0x.)
- tname*: Specify the task name.  
When neither *tn* nor *tname* is specified, the user is prompted to enter a measurement time. Enter a measurement time within the range of 1 to 86,400. The user is then prompted to enter task names or task numbers. Settings for up to 10 tasks are possible. To run the `svtimex` command after entry of a task name or number, press the **Enter** key only, without entering task names or task numbers.
- t *seconds*: Specify the length of time in seconds during which the task activity ratio is measured, within the range of 1 to 86,400. The default is 1 (second).

### Notes on use

- When the -t option is used to specify a measurement time, be sure to specify the task number (*tn*) or task name (*tname*) together.
- Even if the `svtimex` command is re-run after it has already been executed, no request is accepted.
- The *tn* and *tname* options are mutually exclusive. They cannot be specified together. Up to a maximum of 10 task names or task numbers can be specified interactively.

### Termination codes

The `svtimex` command returns one of the following termination codes:

- 0: Normal termination
- 1: Abnormal termination
- 2: Communication error
- 3: Suspension by pressing **Ctrl+C**

### Output format

The results are output as follows:

```
2018/02/07 18:02:18 SITE=0001cp ** 1 second wait **  
sist(255) load ratio=0.00% execute count=0 total time=0.000sec average time=0.000sec
```



**This Page Intentionally Left Blank**

## **APPENDIXES**

## APPENDIX A NAMES USABLE IN PROGRAMS

Use caution when using a program that includes the same name as a subroutine provided by the system. All subroutines provided by the system are contained in the library files. These subroutines can be linked simply by running `svload` with the `-l` option specified. When linking a program that has the same name as a system subroutine, be sure to specify the object file in which the subroutine is defined as an argument of `svload`. If you do not do so, the subroutine that has the same name will be linked from the library file.

The library files provided for each system and the names defined there are described here. when programming, be careful to avoid duplicating names.

If using the same name is unavoidable, specify the library file after the object file to be linked. A subroutine that has the same name will not be linked from the library file.

In the lists that follow, the subroutines provided by the system are grouped by library.

Names that begin with an underscore (`_`) are reserved by the system. Do not use these names.

The following table shows the supported libraries.

| Library name                 | Library description                                                                                                                 | Remarks                                                                  |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| <code>libsh4nbmzz.lib</code> | Subroutines for the C language<br>Compiler: Version 9<br>Denormalized numbers: Denormalized numbers<br>Value rounding: Rounded down | For details, refer to the <code>shc</code> compiler manual of version 9. |
| <code>libsh4nbmdn.lib</code> | Subroutines for the C language<br>Compiler: Version 9<br>Denormalized numbers: 0<br>Value rounding: Rounded down                    |                                                                          |
| <code>libfirad.lib</code>    | Indirect link address reference subroutines                                                                                         |                                                                          |
| <code>libcpms.lib</code>     | CPMS macro linkage subroutines                                                                                                      | Refer to the relevant documentation.                                     |
| <code>libsysctl.lib</code>   | Subroutines for system control                                                                                                      |                                                                          |
| <code>libcycm.lib (*)</code> | Cyclic communication subroutines                                                                                                    |                                                                          |
| <code>libnet.lib</code>      | Socket communication subroutines                                                                                                    |                                                                          |
| <code>libcrs.lib</code>      | IEEE floating-point processing environment control subroutines                                                                      |                                                                          |

(\*) Provided by an IRSUB in the S10VE.

libsh4nbmzz.lib

libsh4nbmdn.lib

|           |         |                 |                 |         |
|-----------|---------|-----------------|-----------------|---------|
| abs       | acos    | acosf           | asin            | asinf   |
| atan      | atan2   | atan2f          | atanf           | atof    |
| atoi      | atol    | atoll           | bsearch         | calloc  |
| ceil      | ceilf   | clearerr        | cos             | cosf    |
| cosh      | coshf   | div             | exp             | expf    |
| fabs      | fabsf   | fclose          | feof            | ferror  |
| fflush    | fgetc   | fgets           | floor           | floorf  |
| fmod      | fmodf   | fopen           | fprintf         | fputc   |
| fputs     | fread   | free            | freopen         | frexp   |
| frexpf    | fscanf  | fseek           | ftell           | fwrite  |
| getc      | getchar | gets            | isalnum         | isalpha |
| iscntrl   | isdigit | isgraph         | islower         | isprint |
| ispunct   | isspace | isupper         | isxdigit        | labs    |
| ldexp     | ldexpf  | ldiv            | llabs           | lldiv   |
| log       | log10   | log10f          | logf            | longjmp |
| longjmp_a | malloc  | memchr          | memcmp          | memcpy  |
| memmove   | memset  | modf            | modff           | perror  |
| pow       | powf    | printf          | putc            | putchar |
| puts      | qsort   | quick_strerror1 | quick_strerror1 | rand    |
| realloc   | rewind  | scanf           | setbuf          | setjmp  |
| setjmp_a  | setvbuf | SFTRL1          | sin             | sinf    |
| sinh      | sinhf   | sml_buf         | sprintf         | sqrt    |
| sqrtf     | srand   | sscanf          | strcat          | strchr  |
| strcmp    | strcpy  | strcspn         | strerror        | strlen  |
| strncat   | strncmp | strncpy         | strpbrk         | strrchr |
| strspn    | strstr  | strtod          | strtok          | strtol  |
| strtoll   | strtoul | strtoull        | tan             | tanf    |
| tanh      | tanhf   | tolower         | toupper         | ungetc  |
| vfprintf  | vprintf | vsprintf        |                 |         |

libfirad.lib

|         |         |
|---------|---------|
| irglbad | irsubad |
|---------|---------|

## APPENDIX A NAMES USABLE IN PROGRAMS

### libcpms.lib

|             |             |             |           |            |
|-------------|-------------|-------------|-----------|------------|
| abort       | arsum       | asusp       | atmadd    | atmand     |
| atmcas      | atmor       | atmswap     | atmtas    | atmxor     |
| cfread      | cfwrite     | chap        | chkbmem   | chktaer    |
| ctime       | delay       | dhpctl      | dhpread   | elctl      |
| exit        | free        | geterrno    | getputype | getsysinfo |
| gettaskinfo | gettimebase | gfact       | gtime     | gkmem      |
| latoma      | memcpy      | pfree       | post      | prog_call  |
| prog_exit   | prog_start  | prog_switch | prsrv     | queue      |
| resume_env  | rleas       | romread     | romwrite  | rs_printf  |
| rserv       | rsum        | save_env    | sfact     | stime      |
| susp        | timer       | usrdhp      | usrel     | wait       |
| wrtmem      |             |             |           |            |

### libsysctl.lib

|           |          |            |         |                |
|-----------|----------|------------|---------|----------------|
| cardstat  |          |            |         |                |
| cpustpctl | dcmcheck | dcmctl     | dcmstat | dsuctl         |
| dsustat   | ledctl   | progwdtset | sysdo   | TimebaseToSecs |
| wdtset    |          |            |         |                |

## APPENDIX A NAMES USABLE IN PROGRAMS

### libcycm.lib

|            |             |           |              |            |
|------------|-------------|-----------|--------------|------------|
| ctlcyc_ghr | getcycm_ghr | rcycm_ghr | restcycm_ghr | stcycm_ghr |
| wcycm_ghr  |             |           |              |            |

### libnet.lib

|          |          |         |            |            |
|----------|----------|---------|------------|------------|
| accept   | bind     | connect | getsockopt | listen     |
| recv     | recvfrom | send    | sendto     | setsockopt |
| shutdown | socket   |         |            |            |

### libcrs.lib

|            |             |             |            |           |
|------------|-------------|-------------|------------|-----------|
| fpcheck    | fpchecko    | fpgetmask   | fpgetround | fpsetmask |
| fpsetround | fpsetsticky | fpgetsticky |            |           |

**APPENDIX B LIBRARIES**

## (1) Conditions for specifying library files

When specifying library files with `svload`, specify the library names shown in Table A-1.

Table A-1 Conditions for Specifying Library Names

| Condition                                                                                      | Library name                                           | Specification in <code>svload</code>                             | Notes                                                                                                             |
|------------------------------------------------------------------------------------------------|--------------------------------------------------------|------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| Programs are coded in C. (The functions in the relevant library shown in Appendix A are used.) | <code>libcrs.lib</code><br><code>libnet.lib</code>     | <code>-lcrs</code><br><code>-lnet</code>                         |                                                                                                                   |
| Created programs use CPMS macros.                                                              | <code>libcpms.lib</code><br><code>libsysctl.lib</code> | <code>-lcpms</code><br><code>-lsysctl</code>                     | See the <i>SIOVE Software Manual CPMS General Description and Macro Specifications</i> (manual number SEE-3-201). |
| Indirect link addresses or indirect link subroutines are referenced.                           | <code>libfirad.lib</code>                              | <code>-lfirad</code>                                             | See (3) Subroutines that reference indirect link addresses.                                                       |
| User-specific libraries are used.                                                              | <code>user_library</code>                              | <code>-l character-string</code><br>or <code>library-name</code> |                                                                                                                   |

## (2) Library specification order

When specifying libraries in `svload`, note the following:

- Specify libraries that contain common subroutines as close to the end as possible.
- When the same name appears in more than one specified library, specify the library holding the object file to be linked first.

**(3) Subroutines that reference indirect link addresses**

When using the following module, link with `libfirad.lib`.

**Name**

`irglbad`

**Syntax**

```
int *irglbad (no)
int no;
```

**Description**

When an indirect link global number (in the range from 1 to the maximum global number) is specified for *no*, the `irglbad` subroutine returns the corresponding global address.

**Return codes**

When a registered indirect link global number is specified for *no*, `irglbad` returns the corresponding global address.

When an unregistered indirect link global number is specified for *no*, `irglbad` returns a value of 0.

**Name**

`irsubad`

**Syntax**

```
int *irsubad (no)
int no;
```

**Description**

When an IRSUB number (in the range from 1 to the maximum global number) is specified for *no*, the `irsubad` subroutine returns the corresponding IRSUB address.

**Return codes**

When a registered IRSUB number is specified for *no*, `irsubad` returns the corresponding IRSUB address.

When an unregistered IRSUB number is specified for *no*, `irsubad` returns a value of 0.



#### (4) Message output routine

When using the following module, link with `libcpms.lib`.

##### Name

`rs_printf`

##### Syntax

```
int rs_printf (buf, fmt, p1, p2, ..., p10)
char *buf;
char *fmt;
long p1, p2, ..., p10;
```

##### Description

`rs_printf` converts data to a message in a specified format, and then writes the message to the message buffer area of the OS.

This message can be read by using the `ps` subcommand of the `svdebug` command.

Ensure that the character string size of the converted message is in the range from 1 and 1,024 bytes. If the character string size of the converted message is outside the range from 1 to 1,024 bytes, a parameter error occurs, and `rs_printf` becomes unable to write data.

##### Parameters

*buf*: Specifies the start address of a memory area into which the message to be written is to be temporarily stored.

*fmt*: Specifies the start address of a memory area that stores a format-indicating character string.

*p1* to *p10*: Used to specify data.

##### Return codes

When terminated normally, `rs_printf` returns the character string size (in bytes) of the converted message.

When terminated abnormally, `rs_printf` returns one of the following return codes:

0: The message buffer area of the OS was full and could not be written to.

-1: A write operation could not be performed due to a parameter error.

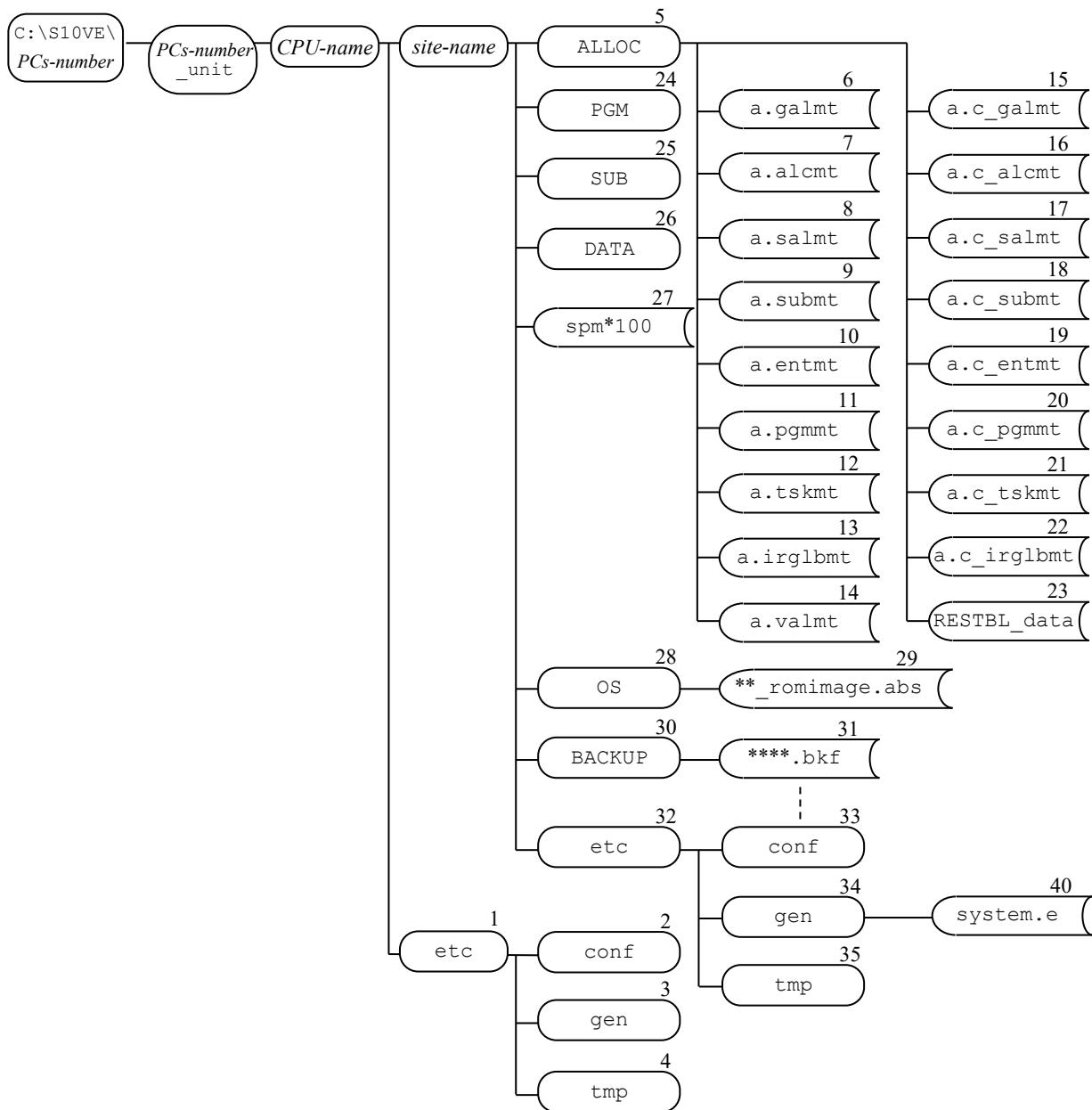
-2: A write operation could not be performed due to a message write error.

##### Notes

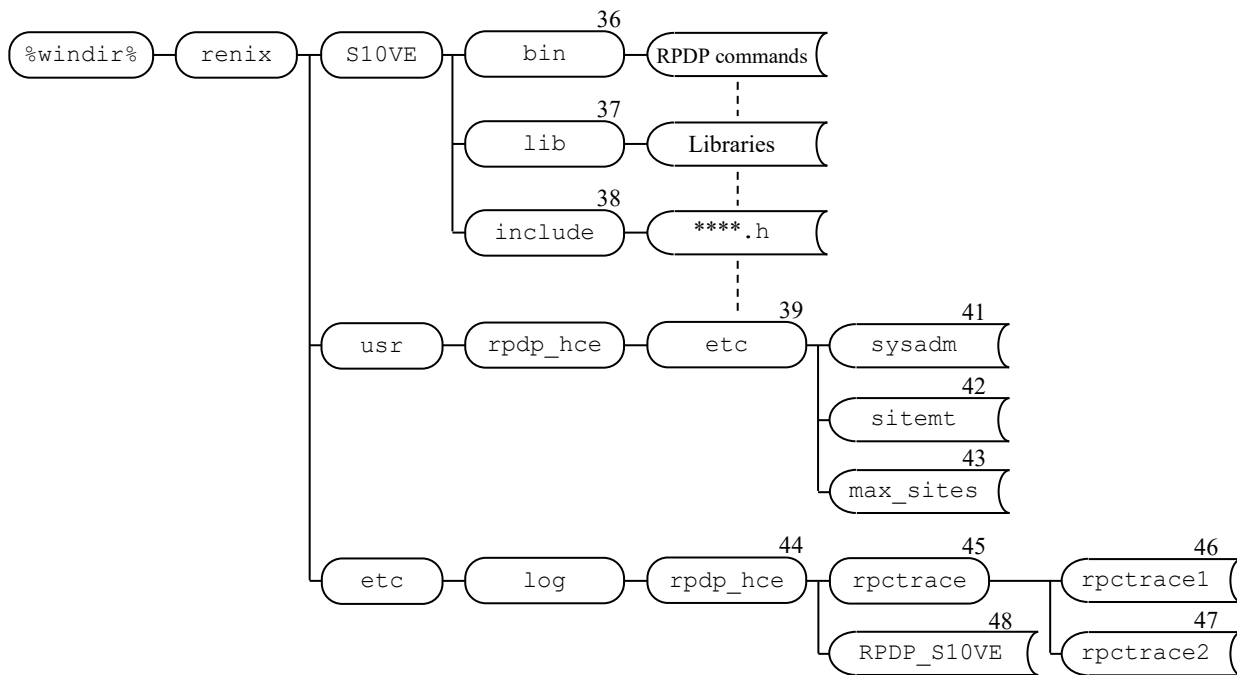
Use this routine for debugging purposes only.

## APPENDIX C SITE MANAGEMENT FILES

This appendix shows the configuration of the directory that contains the files used for site management, and also explains each file.



APPENDIX C SITE MANAGEMENT FILES



| No. | File or directory name | Name                                                                               | Description                                                                                                                               | Notes                                                                                                                                                           |
|-----|------------------------|------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | etc                    | Storage directory for the CPU system generation files                              | This directory stores the system generation files of the CPU (for both CP and HP).                                                        |                                                                                                                                                                 |
| 2   | conf                   | Storage directory for the CPU definition information files                         | This directory contains the user-defined files when a CPU (both CP and HP) system is generated.                                           | ***.u files are defined by users.<br>***.s files are definition files for systems.<br>The svgen command creates a template file.                                |
| 3   | gen                    | Storage directory for the CPU actual reference definition information output files | This directory stores the configuration definition information files of the CPU (both CP and HP), which are output by the svconf command. |                                                                                                                                                                 |
| 4   | tmp                    | Storage directory for the CPU definition information output files                  | This directory stores the definition information output files created based on the information in conf.                                   |                                                                                                                                                                 |
| 5   | ALLOC                  | Storage directory for the allocator management table                               | This directory stores allocator management tables.                                                                                        |                                                                                                                                                                 |
| 6   | a.galmt                | garea management file                                                              | Manages the names, sizes, and other attributes of areas within a logical space.                                                           |                                                                                                                                                                 |
| 7   | a.alcmt                | area management file                                                               | Manages task text and data, subprogram text and data, and AREAs (split areas) allocated within a global (including CM) area.              | Includes entries of the number defined by MAXAREA in the system.u file.                                                                                         |
| 8   | a.salmt                | sarea management file                                                              | Manages the SAREAs (secondary partition areas) allocated within an AREA (split area) in a global (including CM) area.                     | Includes entries to which the number of entries defined by MAXSAREA in the system.u file are added.                                                             |
| 9   | a.submt                | Subprogram management file                                                         | Manages subprograms (IRSUBs and built-in subroutines).                                                                                    | Includes entries of the number equivalent to the sum of the maximum number of built-in subroutines and the number defined by ENTMT_MAXENT in the system.u file. |
| 10  | a.entmt                | IRSUB management file                                                              | Manages indirect link subprograms (IRSUBs).                                                                                               | Includes entries of the number defined by ENTMT_MAXENT in the system.u file.                                                                                    |

APPENDIX C SITE MANAGEMENT FILES

| No. | File or directory name | Name                                       | Description                                                                                                                                    | Notes                                                                                                  |
|-----|------------------------|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| 11  | a.pgmmt                | Program management file                    | Manages programs registered as tasks.                                                                                                          | Includes entries of the number defined by PGM_MAXNUM in the system.u file.                             |
| 12  | a.tskmt                | Task management file                       | Manages tasks.                                                                                                                                 |                                                                                                        |
| 13  | a.irglbmt              | Indirect global management file            | Manages the registration of indirect link global data.                                                                                         | Includes entries of the number defined by IRG_MAXENT in the system.u file.                             |
| 14  | a.valmt                | Value management file                      | Manages the registration of values.                                                                                                            | Includes entries to which the number of entries defined by MVAL_MAXNUM in the system.u file are added. |
| 15  | a.c_galmt              | S10VE garea management file                | Manages the names, sizes, and other attributes of areas within a logical space on the S10VE side.                                              | Downloaded into the S10VE with the svrpl command or the ld subcommand of the svdebug command.          |
| 16  | a.c_alcmt              | S10VE area management file                 | Manages task text and data, subprogram text and data, and AREAs (split areas) allocated within a global (including CM) area on the S10VE side. |                                                                                                        |
| 17  | a.c_salmt              | S10VE sarea management file                | Manages the SAREAs (secondary partition areas) allocated within an AREA (split area) in a global (including CM) area on the S10VE side.        |                                                                                                        |
| 18  | a.c_submt              | S10VE subprogram management file           | Manages the subprograms (IRSUBs and built-in subroutines) on the S10VE side.                                                                   |                                                                                                        |
| 19  | a.c_entmt              | S10VE IRSUB management file                | Manages indirect link subprograms (IRSUBs) on the S10VE.                                                                                       |                                                                                                        |
| 20  | a.c_pgmmt              | S10VE program management file              | Manages programs registered as S10VE tasks.                                                                                                    |                                                                                                        |
| 21  | a.c_tskmt              | S10VE task management file                 | Manages S10VE tasks.                                                                                                                           |                                                                                                        |
| 22  | a.c_irglbmt            | S10VE indirect link global management file | Manages indirect link global registration on the S10VE side.                                                                                   |                                                                                                        |
| 23  | RESTBL_data            | Resource management table data file        | This table data file manages site resources.                                                                                                   |                                                                                                        |
| 24  | PGM                    | Program storage directory                  | This directory stores program load modules.                                                                                                    | Stored when svload is run with the -d option specified.                                                |

| No. | File or directory name                                    | Name                                                                        | Description                                                                                                                    | Notes                                                                                                                                         |
|-----|-----------------------------------------------------------|-----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| 25  | SUB                                                       | Subprogram storage directory                                                | This directory stores subprogram load modules.                                                                                 |                                                                                                                                               |
| 26  | DATA                                                      | Storage directory for global initial-value data                             | This directory stores the initial-value data of a global area.                                                                 |                                                                                                                                               |
| 27  | spm*100<br>(CP side:<br>spmd100)<br>(NP side:<br>spmcl00) | SPM file                                                                    | This is a data file for the OS and drivers in the S10VE main memory (SPM area). File names differ for CP and HP.               | This file is loaded into the S10VE main memory by the <code>svrpl</code> command.                                                             |
| 28  | OS                                                        | OS storage directory                                                        | This directory stores the OS base portion of S10VE.                                                                            |                                                                                                                                               |
| 29  | **_romimage.a<br>bs                                       | OS files                                                                    | The OS files of the S10VE, including I/O and network driver functions.                                                         | This file is loaded into the S10VE main memory by the <code>svrpl</code> command.                                                             |
| 30  | BACKUP                                                    | Backup file storage directory                                               | This directory stores backup files.                                                                                            | Backup files created by <code>svdfa</code> are stored.                                                                                        |
| 31  | ****.bkf                                                  | Backup files                                                                | S10VE memory initial-value files for each split area.                                                                          | These files are loaded into the S10VE main memory by the <code>svrpl</code> command.                                                          |
| 32  | etc                                                       | Storage directory for the system generation files                           | This directory stores system generation files.                                                                                 |                                                                                                                                               |
| 33  | conf                                                      | Storage directory for the site definition information files                 | This directory contains files that are user-defined at the time of system generation.                                          | ***.u files are defined by users.<br>***.s files are definition files for systems.<br>The <code>svgen</code> command creates a template file. |
| 34  | gen                                                       | Storage directory for the configuration definition files                    | This directory stores configuration definition information files that are output by the <code>svconf</code> command.           |                                                                                                                                               |
| 35  | tmp                                                       | Storage directory for the configuration definition information output files | This directory stores definition information output files that are created according to the information in <code>conf</code> . |                                                                                                                                               |
| 36  | bin                                                       | RPDP command storage directory                                              | This directory stores RPDP commands.                                                                                           |                                                                                                                                               |
| 37  | lib                                                       | S10VE library storage directory                                             | This directory stores S10VE libraries.                                                                                         |                                                                                                                                               |
| 38  | include                                                   | Include file storage directory for S10VE                                    | This directory stores S10VE include files.                                                                                     |                                                                                                                                               |

APPENDIX C SITE MANAGEMENT FILES

| No. | File or directory name | Name                                              | Description                                                     | Notes                                                                                                      |
|-----|------------------------|---------------------------------------------------|-----------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| 39  | etc                    | Storage directory for the system management files | This directory stores system management files.                  |                                                                                                            |
| 40  | system.e               | Site constant management file                     | This file stores site constant information.                     |                                                                                                            |
| 41  | sysadm                 | System management files                           | This file stores site information.                              |                                                                                                            |
| 42  | sitent                 | System management files                           | This file stores site information.                              |                                                                                                            |
| 43  | max_sites              | Maximum site count management file                | This file stores information about the maximum number of sites. |                                                                                                            |
| 44  | rpdp_hce               | Log file storage directory                        | This directory stores log files.                                |                                                                                                            |
| 45  | rpctrace               | Storage directory for the RPC library log files   | This directory stores log files for RPC libraries.              |                                                                                                            |
| 46  | rpctrace1              | RPC library log file                              | Log file for the RPC library                                    | When the stored log count reaches the maximum number, the log location changes to <code>rpctrace2</code> . |
| 47  | rpctrace2              | RPC library log file                              | Log file for the RPC library                                    | When the stored log count reaches the maximum number, the log location changes to <code>rpctrace1</code> . |
| 48  | RPDP_S10VE             | RPDP log file                                     | Log file for RPDP                                               |                                                                                                            |

## APPENDIX D ERROR MESSAGES

This appendix describes the error messages that are output by commands, as well as the actions to be taken by the user in response to the error messages.

Values and character strings related to errors are represented in the error messages as *%s*, *%d*, and *%x*.

| No. | Item                                                          | Command   | Corresponding page |
|-----|---------------------------------------------------------------|-----------|--------------------|
| (1) | Error messages of the allocator, loader, and builder commands | svdfa     | A-16               |
|     |                                                               | svdla     |                    |
|     |                                                               | svdfs     |                    |
|     |                                                               | svdls     |                    |
|     |                                                               | svdfv     |                    |
|     |                                                               | svdlv     |                    |
|     |                                                               | svload    |                    |
|     |                                                               | svdload   |                    |
|     |                                                               | svcomp    |                    |
|     |                                                               | svctask   |                    |
|     |                                                               | svdtask   |                    |
|     |                                                               | svbuild   |                    |
|     |                                                               | svdbuild  |                    |
|     |                                                               | svirglb   |                    |
| (2) | svdebug error messages                                        | svdebug   | A-24               |
| (3) | svrpl command error messages                                  | svrpl     | A-32               |
| (4) | svcpuctl command error messages                               | svcpuctl  | A-34               |
| (5) | svelog command error messages                                 | svelog    | A-35               |
| (6) | svdhp command error messages                                  | svdhp     | A-36               |
| (7) | svcpunow command error messages                               | svcpunow  | A-39               |
| (8) | svtimex command error messages                                | svtimex   | A-42               |
| (9) | svdatagen command error messages                              | svdatagen | A-45               |



(1) Error messages displayed by the allocator, loader, and builder commands

- Error messages

Table A-2 lists the error messages displayed by the allocator, loader, builder, and map display commands. When these commands detect an error, they display the relevant error message and then terminate.

Table A-2 Error Messages

(1/8)

| Error code                                                     | Message                                  | Explanation                                                      | User action                                                                  |
|----------------------------------------------------------------|------------------------------------------|------------------------------------------------------------------|------------------------------------------------------------------------------|
| Internal errors requiring recovery by the system administrator |                                          |                                                                  |                                                                              |
| 1001-1                                                         | Abnormal allocator management table (%s) | An allocator management table contains an error.                 | Collect data for examination, and then restart the development machine. (*1) |
| Errors related to insufficient resources                       |                                          |                                                                  |                                                                              |
| 1002-4                                                         | Not enough physical memory allocated     | The physical memory is insufficient.                             | Delete unnecessary resources, and then try again.                            |
| 6                                                              | Not enough area allocated (%s)           | The GLB area is insufficient.                                    |                                                                              |
| 8                                                              | No task number available                 | No more task numbers are available.                              |                                                                              |
| 10                                                             | No free table to make new entry (%s)     | No more allocator management tables are available.               |                                                                              |
| 13                                                             | Cannot get RSSITE (%s)                   | The RSSITE environment variable could not be fetched.            | Set the environment variable, and then try again.                            |
| 15                                                             | Please set environment variable (%s)     | The environment variable was not set.                            |                                                                              |
| Errors related to insufficient system resources                |                                          |                                                                  |                                                                              |
| 1003-1                                                         | Memory allocation error (malloc, %s)     | Memory could not be allocated by using malloc.                   | Check the amount of available memory, and then try again.                    |
| 3                                                              | Cannot create temporaries: %s            | A temporary file could not be created.                           | Check the amount of available disk space, and then try again.                |
| Errors unrecoverable by retries                                |                                          |                                                                  |                                                                              |
| 1004-1                                                         | Allocator management table is busy       | The allocator management table is being used by another command. | Re-run the command.                                                          |
| System call errors due to identifiable causes                  |                                          |                                                                  |                                                                              |
| 1005-1                                                         | Cannot open %s (%s)                      | The file could not be opened.                                    | Review the file access rights, and then try again.                           |
| 2                                                              | Cannot read %s (%s)                      | Data could not be read from the file.                            |                                                                              |
| 3                                                              | Cannot write %s (%s)                     | Date could not be written to the file.                           |                                                                              |
| 4                                                              | Cannot stat %s (%s)                      | Status information could not be read from the file.              | Collect data for examination, and then restart the development machine. (*1) |
| 5                                                              | Cannot lseek %s (%s)                     | A file pointer could not be sought.                              |                                                                              |

| Error code                                      | Message                                                      | Explanation                                                                           | User action                                                                  |
|-------------------------------------------------|--------------------------------------------------------------|---------------------------------------------------------------------------------------|------------------------------------------------------------------------------|
| System call errors due to unidentifiable causes |                                                              |                                                                                       |                                                                              |
| 1006-1                                          | syscall error (%s, errno=%d) (*2)                            | A system call error occurred.                                                         | Collect data for examination, and then restart the development machine. (*1) |
| 2                                               | Command name:WIN32API error (API-name, EC= error code) (*3)  | A function that has API-name caused the error indicated by the error code.            |                                                                              |
| Parameter insufficiency errors                  |                                                              |                                                                                       |                                                                              |
| 1007-3                                          | Not enough parameter                                         | The arguments are insufficient.                                                       | Review the arguments.                                                        |
| Out-of-range input parameters                   |                                                              |                                                                                       |                                                                              |
| 2001-2                                          | Align number is out of range                                 | The alignment number is invalid.                                                      | Verify the data that can be entered, and then re-run the command.            |
| 3                                               | Task number is out of range (1 to %d)                        | The task number is invalid.                                                           |                                                                              |
| 5                                               | Priority level is out of range (%d to %d)                    | The user task execution level is invalid.                                             |                                                                              |
| 6                                               | Priority level for system is out of range (%d to %d)         | The system task execution level is invalid.                                           |                                                                              |
| 9                                               | Bad align type                                               | The alignment type is invalid.                                                        |                                                                              |
| 10                                              | Illegal point number (%d)                                    | The point number for a built-in subroutine is invalid.                                |                                                                              |
| 11                                              | Entry number is out of range (1 to %d)                       | The entry number for a built-in subroutine is invalid.                                |                                                                              |
| 12                                              | Specified index number is out of range (1 to %d)             | The specified index number is invalid.                                                |                                                                              |
| 13                                              | Invalid name (%s)                                            | The specified name contains an error.                                                 |                                                                              |
| 18                                              | Illegal point name (%s)                                      | The point name specified for a built-in subroutine contains an error.                 |                                                                              |
| 19                                              | Numeric value is out of range                                | The specified numeric value is invalid.                                               |                                                                              |
| 23                                              | Limit size for stack is out of range (0 to 2097152)          | The specified stack size is invalid.                                                  |                                                                              |
| 24                                              | User task number is out of range (1 to 224)                  | The specified user task number is invalid.                                            |                                                                              |
| 25                                              | Specified system index number is out of range (1 to %d)      | The specified system index number is invalid.                                         |                                                                              |
| 26                                              | Specified number with -r is out of range                     | The number specified by the -r option is invalid.                                     |                                                                              |
| 27                                              | Loading data is empty                                        | No loading data exists.                                                               |                                                                              |
| 28                                              | Number of user task is over (number-of-tasks)                | The number of user tasks registered is greater than the value of the system constant. | Delete unnecessary resources, and then try again.                            |
| 33                                              | ULSUB stack size (%d) is out of range (0 to 512)             | The stack size of a built-in subprogram is larger than 512 bytes.                     | Review the stack size.                                                       |
| 35                                              | Program/IRSUB stack size (%d) is out of range (0 to 8388608) | The stack size of a task is larger than 8 MB.                                         |                                                                              |

| Error code                        | Message                                             | Explanation                                                                             | User action                                                       |
|-----------------------------------|-----------------------------------------------------|-----------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| Undefined input parameters        |                                                     |                                                                                         |                                                                   |
| 2002-1                            | Specified name is undefined (%s)                    | An undefined name is specified.                                                         | Verify the data that can be entered, and then re-run the command. |
| 4                                 | Specified point number in the entry number is empty | An undefined point number is specified for a built-in subroutine.                       |                                                                   |
| 5                                 | Specified IRSUB is not built (%s)                   | An IRSUB that has not yet been built is specified.                                      |                                                                   |
| 7                                 | Specified IRSUB is already built (%s)               | An already-built IRSUB is specified.                                                    |                                                                   |
| 8                                 | Specified number is undefined                       | An undefined number is specified.                                                       |                                                                   |
| 10                                | %s is undefined                                     | An undefined name was detected in an object.                                            |                                                                   |
| 11                                | Loading data is empty                               | No data to be loaded exists.                                                            |                                                                   |
| 13                                | Area (%s) kind is wrong                             | The area type is incorrect.                                                             | Review the specified area.                                        |
| 14                                | Can not load data in GLBW, CMW, and DCMW (%s)       | No initial value can be loaded into a GLB or CM area without an initial value.          |                                                                   |
| 15                                | Can not load CM, DCM data from NP or HP site        | Loading from the HP site to the CM is disabled.                                         |                                                                   |
| Multiply defined input parameters |                                                     |                                                                                         |                                                                   |
| 2003-1                            | Specified name is already defined (%s)              | A name that has already been registered is specified.                                   | Verify the data that can be entered, and then re-run the command. |
| 5                                 | Task number is already defined                      | A task number that has already been registered is specified.                            |                                                                   |
| 7                                 | Point number is already defined                     | A point number that has already been registered is specified for a built-in subroutine. |                                                                   |
| 8                                 | Specified IRSUB number is already defined           | An IRSUB number that has already been registered is specified.                          | Change the IRSUB number, and then re-run the command.             |
| 13                                | Unmatched reserved index number                     | The specified index number does not match.                                              | Verify the data that can be entered, and then re-run the command. |
| 15                                | PN=%s is already defined                            | A duplicate program management number is specified.                                     |                                                                   |
| 19                                | Specified number is already defined                 | A number that has already been registered is specified.                                 |                                                                   |
| 21                                | Can not specify -s or -a with SAREA (%s)            | The -s or -a option specified for a secondary partition area cannot be specified.       |                                                                   |
| 22                                | Specified pgmname is already defined as TASK (%s)   | A task name that has already been registered is specified as a program name.            |                                                                   |
| 23                                | PN=%d is already loaded for single task             | A program management number that has already been registered is specified.              |                                                                   |

| Error code                                     | Message                                           | Explanation                                                                                            | User action                                                                          |
|------------------------------------------------|---------------------------------------------------|--------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| <b>Non-matching input parameter attributes</b> |                                                   |                                                                                                        |                                                                                      |
| 2004-1                                         | Unmatched owner type                              | The owner type does not match.                                                                         | Verify the data that can be entered, and then re-run the command.                    |
| 2                                              | Illegal user type (%s)                            | The user type does not match.                                                                          |                                                                                      |
| 3                                              | Specified area is not global (%s)                 | The area type was not GLB.                                                                             |                                                                                      |
| 5                                              | Illegal program type                              | The program type does not match.                                                                       |                                                                                      |
| 8                                              | Unmatched entry number                            | A non-matching entry number is specified for a built-in subroutine.                                    |                                                                                      |
| 10                                             | Area type is not GLBI                             | The area type is not GLB with initial data.                                                            |                                                                                      |
| 11                                             | Multi task attribute error                        | The multitask attribute does not match.                                                                |                                                                                      |
| 16                                             | Unmatched entry type                              | A non-matching entry set number is specified for a built-in subroutine.                                |                                                                                      |
| 17                                             | Specified name is defined as GLB, CM or DCM (%s)  | A GLB or CM name that has already been defined is specified.                                           |                                                                                      |
| 19                                             | Specified name is defined as VAL (%s)             | A VAL name that has already been defined is specified.                                                 |                                                                                      |
| 21                                             | 4096 aline error (%s)                             | The specified address is not on a 4096-byte boundary.                                                  |                                                                                      |
| 22                                             | Physical address error (%s)                       | An invalid logical address is specified.                                                               |                                                                                      |
| 23                                             | Area (%s) kind is wrong                           | An invalid area attribute is specified.                                                                |                                                                                      |
| 24                                             | Loading data is too large (sname=%s)              | The specified data size is greater than the area size.                                                 |                                                                                      |
| 25                                             | Inconsistent object was mixed                     | Both -lsh4nbmzz and -lsh4nbmdn are specified for library.                                              |                                                                                      |
| <b>Incorrect operations</b>                    |                                                   |                                                                                                        |                                                                                      |
| 2005-1                                         | Cannot delete area which is already used          | The area could not be deleted because tasks or subprograms are registered.                             | Execute svdload before deleting the area.                                            |
| 2                                              | Cannot delete program which is registered as task | The program could not be deleted because it is registered as a task.                                   | Execute svdtask before deleting the program.                                         |
| 3                                              | Cannot delete built subprogram (%s)               | The subprogram could not be deleted because it has already been built.                                 | Execute svdbuild before deleting a subprogram that has already been built.           |
| 6                                              | Cannot delete defined %s (%s)                     | The area could not be deleted because it is registered as a GLB or VAL.                                | Execute svdls or svdlv before deleting the area.                                     |
| 7                                              | Specified name (%s) is referenced by PROG or SUB  | The specified resource could not be deleted because it is being referenced by a program or subprogram. | Delete the referencing program or subprogram before deleting the specified resource. |

| Error code                                                     | Message                                                  | Explanation                                                                                                                                                                                                                                                                                                                                   | User action                                                                                                         |
|----------------------------------------------------------------|----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| Invalid parameters detected in the <code>svload</code> command |                                                          |                                                                                                                                                                                                                                                                                                                                               |                                                                                                                     |
| 2006-1                                                         | Invalid subargument: <code>-W%s</code>                   | An unusable sub-argument is specified.                                                                                                                                                                                                                                                                                                        | Verify the data that can be entered, and then re-run the command.                                                   |
| 5                                                              | Too few arguments                                        | The number of specified arguments is insufficient.                                                                                                                                                                                                                                                                                            |                                                                                                                     |
| 8                                                              | Missing operand ( <code>%s</code> )                      | Some operands are missing.                                                                                                                                                                                                                                                                                                                    |                                                                                                                     |
| 9                                                              | Bad option ( <code>%s</code> )                           | An unusable option is specified.                                                                                                                                                                                                                                                                                                              |                                                                                                                     |
| 10                                                             | Invalid name ( <code>%s</code> )                         | A name is specified incorrectly.                                                                                                                                                                                                                                                                                                              |                                                                                                                     |
| Invalid parameters detected in the <code>svload</code> command |                                                          |                                                                                                                                                                                                                                                                                                                                               |                                                                                                                     |
| 2008-1                                                         | Error in <code>%s</code> ; Status termination code       | An error occurred in an internal command ( <code>%s</code> ).<br>For the internal command error cause, see the internal command error message that was output at the same time. For error messages of internal commands (linker), refer to the compiler package manual (PDF file). (Termination codes do not have a meaning in this context.) | Check whether the specified library is correct, review the object file and its source, and then re-run the command. |
| 2008-2                                                         | Fatal error in <code>%s</code> ; Status termination code | An internal command ( <code>%s</code> ) caused a fatal error.<br>For the internal command error cause, see the internal command error message that was output at the same time. For error messages of internal commands (linker), refer to the compiler package manual (PDF file). (Termination codes do not have a meaning in this context.) | Collect data for examination, and then restart the development machine. (*4)                                        |
| Allocator management table errors                              |                                                          |                                                                                                                                                                                                                                                                                                                                               |                                                                                                                     |
| 200-1                                                          | <code>RMphase, (0x%02x) (*)</code>                       | The processing phase is invalid.                                                                                                                                                                                                                                                                                                              | Restart the development machine.                                                                                    |
| Invalid input data                                             |                                                          |                                                                                                                                                                                                                                                                                                                                               |                                                                                                                     |
| -                                                              | Argument list too long                                   | Too many arguments are specified.                                                                                                                                                                                                                                                                                                             | Verify the data that can be entered, and then re-run the command.                                                   |
| -                                                              | Argument data too large                                  | Too much data is specified in an argument.                                                                                                                                                                                                                                                                                                    |                                                                                                                     |
| -                                                              | File open error                                          | The specified file could not be opened.                                                                                                                                                                                                                                                                                                       |                                                                                                                     |
| -                                                              | Illegal format in <i>file-name</i> , <i>line-number</i>  | <i>Line-number</i> in the specified file is in an invalid format.                                                                                                                                                                                                                                                                             |                                                                                                                     |
| -                                                              | Illegal format of name                                   | A name is specified in an invalid format.                                                                                                                                                                                                                                                                                                     |                                                                                                                     |
| -                                                              | Illegal format of numeric value                          | Numerical data is specified incorrectly.                                                                                                                                                                                                                                                                                                      |                                                                                                                     |
| -                                                              | Illegal format of task name                              | A task name is specified in an invalid format.                                                                                                                                                                                                                                                                                                |                                                                                                                     |

| Error code                                            | Message                                                    | Explanation                                                                                                        | User action                                                       |
|-------------------------------------------------------|------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| Invalid input data (continued from the previous page) |                                                            |                                                                                                                    |                                                                   |
| -                                                     | Illegal operand                                            | An operand is specified incorrectly.                                                                               | Check the data that can be entered, and then re-run the command.  |
| -                                                     | Program text is empty                                      | The specified program text size is 0.                                                                              |                                                                   |
| -                                                     | %s is different from subprogram top name                   | A subprogram name is specified incorrectly.                                                                        |                                                                   |
| -                                                     | %s is referred from system type                            | A system task is referencing the user task.                                                                        |                                                                   |
| -                                                     | %s is referred from user type                              | A user task is referencing the system task.                                                                        | Verify the data that can be entered, and then re-run the command. |
| -                                                     | Illegal option                                             | An invalid option is specified.                                                                                    |                                                                   |
| -                                                     | Illegal option combination                                 | Options were combined incorrectly.                                                                                 |                                                                   |
| -                                                     | Missing option parameter                                   | An option parameter is specified incorrectly.                                                                      |                                                                   |
| -                                                     | Numeric value is out of range                              | The specified numeric value is invalid.                                                                            |                                                                   |
| -                                                     | Parameter error                                            | A parameter is specified incorrectly.                                                                              |                                                                   |
| -                                                     | Specified number is undefined<br>( <i>specified-name</i> ) | The name and -n options are specified in the svmap command, but the entry number specified by name is not defined. |                                                                   |
| -                                                     | Specified number is illegal<br>( <i>specified-name</i> )   | The name and -n options are specified in the svmap command, but the entry number specified by name is invalid.     |                                                                   |
| -                                                     | Specified name is undefined<br>( <i>specified-name</i> )   | The name option is specified in the svmap command, but the specified name is undefined.                            |                                                                   |
| -                                                     | Task number error                                          | A task number is specified incorrectly.                                                                            |                                                                   |
| -                                                     | Bad file name                                              | The file name, specified as the operation result output destination in the svadm command, exceeds 255 characters.  |                                                                   |
| -                                                     | Bad site name                                              | The site name specified in the svadm command exceeds 14 characters.                                                |                                                                   |
| -                                                     | Parameter is too long                                      | Too many arguments are specified.                                                                                  |                                                                   |
| -                                                     | Illegal parameter                                          | The argument parameters contain an error.                                                                          |                                                                   |
| -                                                     | Sitename max length is 14 character (-u)                   | The site name exceeds 14 characters.                                                                               |                                                                   |
| -                                                     | Multi task count is out of range<br>(2 to 128)             | The number of multitasks contains an error.                                                                        | Specify a value in the range from 2 to 128.                       |
| -                                                     | Can not get RSSITE                                         | The environment variable could not be fetched.                                                                     | Specify the RSSITE environment variable.                          |
| -                                                     | No such site (%s)                                          | The specified site was not found.                                                                                  | Review the specified site.                                        |

| Error code                                            | Message                                                               | Explanation                                                                              | User action                                                                    |
|-------------------------------------------------------|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| Invalid input data (continued from the previous page) |                                                                       |                                                                                          |                                                                                |
| -                                                     | Some system constants are not defined                                 | The specified constant was not found in the system constants.                            | Review the specified constant.                                                 |
| -                                                     | -w option argument is not 8 byte align                                | A value that is not a multiple of 8 is specified for the stack size.                     | Review the specified size.                                                     |
| -                                                     | Total stack size (%d) is too small                                    | The stack size to be allocated is smaller than the stack size to be used by the program. | Review the stack size.                                                         |
| -                                                     | -C option argument is not %d byte align                               | An invalid value is specified by the -C option.                                          | Specify a multiple of 4096 for programs, and a multiple of 32 for subprograms. |
| -                                                     | Bad realtime environment                                              | An invalid environment variable contains an error.                                       | Review the environment variable.                                               |
| -                                                     | Program text is empty                                                 | The text size of the program is 0.                                                       | Review the specified object file.                                              |
| -                                                     | %s is different from subprogram top name                              | The subprogram name is incorrect.                                                        |                                                                                |
| -                                                     | .rodata cannot locate GLB area                                        | The data declared by const cannot be loaded.                                             | Review the const declaration of the initial value data program.                |
| -                                                     | %s is not defined (Sarea)                                             | Initial value data of an undefined GLB or CM is included in the object file.             | Review the initial value data.                                                 |
| -                                                     | Program has BSS area                                                  | A multitask or IRSUB has a BSS area.                                                     | Check the program.                                                             |
| -                                                     | Stack size (%s) = %d (%d) byte<br>[MAX refered (%s) size %d byte] Err | The accumulation stack size exceeded the specified stack size.                           | Review the specified stack size.                                               |
| -                                                     | Can not get site information                                          | Site information cannot be fetched.                                                      | Check the site definition, and then try again.                                 |
| -                                                     | CM area address or size is different form another CPU                 | The address or size differs from another CPU (CP, HP).                                   | Match the CM address and size.                                                 |
| Execution environment errors                          |                                                                       |                                                                                          |                                                                                |
| -                                                     | Please set RSSITE                                                     | The RSSITE environment variable is not set.                                              | Set the RSSITE environment variable, and then re-run the command.              |
| -                                                     | Unknown RSUTYP                                                        | An invalid parameter is set in the RSUTYP environment variable.                          | Set an s or u in the RSUTYP environment variable.                              |

(8/8)

| Error code     | Message                           | Explanation                                                           | User action                                                                  |
|----------------|-----------------------------------|-----------------------------------------------------------------------|------------------------------------------------------------------------------|
| Command errors |                                   |                                                                       |                                                                              |
| -              | cannot perform malloc             | A work area could not be allocated by using malloc or realloc.        | Re-run the command.                                                          |
| -              | Internal error (timeout detected) | A communication timeout was detected.                                 |                                                                              |
| -              | Internal error (no valid data)    | An error was found in the communication data.                         | Collect data for examination, and then restart the development machine. (*1) |
| -              | cannot open %s                    | The svmap command could not open the allocator management table file. |                                                                              |
| -              | Specified site is undefined       | The specified site could not be found.                                | Review the site name.                                                        |

(\*1) Data for examination is stored in the following files:

- %windir%\renix\etc\log\rpdp\_hce\RPDP\_S10VE
- %windir%\renix\etc\log\rpdp\_hce\1\_ RPDP\_S10VE

(\*2) The meaning of errno, as indicated in error code 1006-1:

| errno | Meaning                                   |
|-------|-------------------------------------------|
| 2     | A file is missing.                        |
| 3     | A path is invalid.                        |
| 9     | A file is invalid.                        |
| 12    | The available memory is insufficient.     |
| 13    | Access rights have not been provided.     |
| 24    | Too many files are open.                  |
| 28    | The available disk space is insufficient. |

(\*3) The meaning of the error codes indicated in error code 1006-2:

| Error code | Meaning                               |
|------------|---------------------------------------|
| 2          | A file is missing.                    |
| 3          | A path is invalid.                    |
| 4          | Too many files are open.              |
| 5          | Access rights have not been provided. |
| 6          | The handle is invalid.                |
| 8, 14      | The available memory is insufficient. |

(\*4) Meaning of the processing phases:

- 0x01: The management table in the main memory is being updated.
- 0x02: The management table file is being updated.
- 0x03: The hash table is being updated.
- 0xff: A system status error occurred.



## (2) Error messages displayed by the svdebug command

(1/7)

| No. | Error message                         | Explanation                                                   | User action                                            |
|-----|---------------------------------------|---------------------------------------------------------------|--------------------------------------------------------|
| 1   | File <i>file-name</i> already exists  | The specified file already exists.                            | Specify the correct file name.                         |
| 2   | Site <i>site-name</i> not found       | The specified site could not be found.                        | Check the site name.                                   |
| 3   | Cannot open <i>file-name</i>          | The specified file could not be opened.                       | Specify the correct file name.                         |
| 4   | No filename given for -i/-o/-r option | A file name is missing.                                       | Specify a file name.                                   |
| 5   | No sitename given for -u option       | A site name is missing.                                       | Specify a site name.                                   |
| 6   | Task No error                         | A task number is specified incorrectly.                       | Check the task number.                                 |
| 7   | Task name error                       | A task name is specified incorrectly.                         | Check the task name.                                   |
| 8   | Factor error                          | A start factor is specified incorrectly.                      | Check the start factor.                                |
| 9   | Cannot Specify RPC-server task        | An RPC server task cannot be specified.                       | Check the task number and task name.                   |
| 10  | Unknown sub command                   | An uninterpretable subcommand name is specified.              | Check the subcommand name.                             |
| 11  | Name error                            | An uninterpretable name is specified.                         | Check the name.                                        |
| 12  | Option error                          | An uninterpretable option is specified.                       | Check the options.                                     |
| 13  | Storage error                         | An uninterpretable storage device is specified.               | Check the storage device.                              |
| 14  | Invalid address set                   | An inaccessible address is specified.                         | Check the address.                                     |
| 15  | Misformed patch data                  | New data used for modification is set incorrectly.            | Enter a real number in octal, decimal, or hexadecimal. |
| 16  | Unknown RSSITE                        | The RSSITE environment variable is not yet set.               | Set the RSSITE environment variable.                   |
| 17  | Unknown RSUTYP                        | An attempt was made to access a system resource in user mode. | Set an s or u in the RSUTYP environment variable.      |

| No. | Error message                                             | Explanation                                                                            | User action                                                                          |
|-----|-----------------------------------------------------------|----------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| 18  | Break point already used by another process               | The breakpoint is being used by another debugger process.                              | Use the breakpoint after the other user finishes.                                    |
| 19  | Input error                                               | The input format is incorrect.                                                         | Check the input format.                                                              |
| 20  | Type or length error                                      | The md or sd option is specified incorrectly.                                          | Check the options.                                                                   |
| 21  | RPDP library error<br>( <i>library-name: error-code</i> ) | An error occurred in the RPDP library.                                                 | Collect data for examination, and then restart the development machine.<br>(*1) (*3) |
| 22  | Allocator management table busy                           | The allocator management table is busy.                                                | Re-run the command.                                                                  |
| 23  | Unmatch resource status                                   | A disagreement exists in the resource state between the development machine and S10VE. | Check the specified name.                                                            |
| 24  | Id NO error                                               | The value id is incorrectly specified for the tm subcommand.                           | Review the value id.                                                                 |
| 25  | Time error                                                | The value t is incorrectly specified for the tm subcommand.                            | Review the value t.                                                                  |
| 26  | Cycle time error                                          | The value cyct part is incorrectly specified for the tm subcommand.                    | Review the value cyct.                                                               |
| 27  | Initial/check data error                                  | An initialization check pattern is specified incorrectly in the si or sp subcommand.   | Check the specified initialization check pattern.                                    |
| 28  | Addr error                                                | An address is specified incorrectly in the br, rb, or as subcommand.                   | Check the specified address.                                                         |
| 29  | Break point is used                                       | A breakpoint is being used.                                                            | Reset the breakpoint, and then exit the debugger.                                    |
| 30  | Bit data error                                            | Bit data is specified incorrectly in the bs or bg subcommand.                          | Check the specified bit data.                                                        |
| 31  | Sht sub command is already executed by another process    | The sht subcommand is being run by another debugger process.                           | Use the subcommand after the other user finishes.                                    |
| 32  | Task no error (NO.2100-01)                                | The task number is incorrectly specified.                                              | Review the task number.                                                              |
| 33  | Task name error (NO.2100-02)                              | The task name is incorrectly specified.                                                | Review the task name.                                                                |

| No. | Error message                                  | Explanation                                                                     | User action                                                                  |
|-----|------------------------------------------------|---------------------------------------------------------------------------------|------------------------------------------------------------------------------|
| 34  | Factor error (NO.2100-03)                      | The start factor is incorrectly specified.                                      | Review the start factor.                                                     |
| 35  | Cannot Specify RPC-server task (NO.2100-04)    | No RPC server task can be specified.                                            | Review the task number and task name.                                        |
| 36  | Specified task is dormant (NO.2100-05)         | The specified task is dormant.                                                  | Confirm the task status, and then try again.                                 |
| 37  | Specified task is not dormant (NO.2100-06)     | The specified task is not dormant.                                              |                                                                              |
| 38  | Specified task is already suspend (NO.2100-07) | The specified task has already been suspended.                                  |                                                                              |
| 39  | Specified task is not suspend (NO.2100-08)     | The specified task is no longer suspended.                                      |                                                                              |
| 40  | Specified task is not registered (NO.2100-09)  | The specified task is not registered.                                           | Review the task number and task name.                                        |
| 41  | Backup file access error (NO.2100-10)          | The backup file could not be accessed.                                          | Collect data for examination, and then restart the development machine. (*1) |
| 42  | Unmatched RSUTYP (NO.2100-11)                  | The system resources cannot be accessed in the user mode.                       | Gain access in the system mode.                                              |
| 43  | Unmatch resource status (NO.2100-12)           | A resource status mismatch was found between the development machine and S10VE. | Review the specified name.                                                   |
| 44  | Specified task is undefined (NO.2100-13)       | The specified task name is not defined.                                         | Review the specified task.                                                   |
| 45  | CPMS not running (NO.2100-14)                  | The CPMS is not running.                                                        | Ensure that the CPMS is running.                                             |
| 46  | Task is not dormant (tn=%d) (NO.2100-15)       | The task is not dormant.                                                        | Render the task dormant, and then try again.                                 |
| 47  | Invalid address error (NO.2100-16)             | An attempt was made to access an inaccessible address.                          | Review the address.                                                          |
| 48  | Invalid address set (NO.2100-17)               | An inaccessible address is specified.                                           | Review the address.                                                          |
| 49  | Cannot open <i>file-name</i> (NO.2100-18)      | The file could not be opened.                                                   | Specify a correct file name.                                                 |

| No. | Error message                                               | Explanation                                                                      | User action                                                                  |
|-----|-------------------------------------------------------------|----------------------------------------------------------------------------------|------------------------------------------------------------------------------|
| 50  | Processor connection table is full<br>(NO.2100-19)          | The inter-process connection table is full.                                      | Wait until another user finishes, and then try again.                        |
| 51  | Specified task is not idle<br>(NO.2100-20)                  | The specified task is not idle.                                                  | Confirm the task status, and then try again.                                 |
| 52  | Timer event is not registered<br>(NO.2100-21)               | No timer event is registered.                                                    | Review the specified task.                                                   |
| 53  | Time error (NO.2100-22)                                     | The value <i>t</i> is incorrectly specified for the <i>tm</i> subcommand.        | Review the value <i>t</i> .                                                  |
| 54  | Cycle time error (NO.2100-23)                               | The value <i>cyct</i> is incorrectly specified for the <i>tm</i> subcommand.     | Review the value <i>cyct</i> .                                               |
| 55  | Cannot get system constant<br>(NO.2100-24)                  | The system constant could not be acquired.                                       | Collect data for examination, and then restart the development machine. (*1) |
| 56  | Ent error (NO.2100-25)                                      | The value <i>ent</i> is incorrectly specified for the <i>ld</i> subcommand.      | Review the specified <i>ent</i> value.                                       |
| 57  | Irsb No error (NO.2100-26)                                  | The indirect link subroutine number is incorrect.                                | Review the indirect link subroutine number.                                  |
| 58  | Irglobal No error<br>(NO.2100-27)                           | The indirect link global number is incorrectly specified.                        | Review the indirect link global number.                                      |
| 59  | Task suspend failed<br>(NO.2100-28)                         | The task was not successfully suspended by the <i>ta</i> subcommand.             | Confirm the task status, and then try again.                                 |
| 60  | Point error (NO.2100-29)                                    | An incorrect point is specified for the <i>ld</i> subcommand.                    | Review the specified point.                                                  |
| 61  | Cannot get register information<br>(NO.2100-30)             | The contents of a register could not be acquired by the <i>ta/rr</i> subcommand. | Collect data for examination, and then restart the development machine. (*1) |
| 62  | Specified name ( <i>name</i> ) is undefined<br>(NO.2100-31) | The specified name is not defined.                                               | Review the specified name.                                                   |
| 63  | Cannot register timer event in TRB<br>(NO.2100-32)          | Timer event registration failed.                                                 | Collect data for examination, and then restart the development machine. (*1) |
| 64  | File <i>file-name</i> already exists<br>(NO.2100-33)        | An existing file is specified.                                                   | Check the specified file name.                                               |
| 65  | File <i>file-name</i> create error<br>(NO.2100-34)          | A file could not be created.                                                     |                                                                              |
| 66  | Cannot save <i>file-name</i><br>(NO.2100-35)                | A file could not be saved.                                                       |                                                                              |

| No. | Error message                                                   | Explanation                                                            | User action                                                                  |
|-----|-----------------------------------------------------------------|------------------------------------------------------------------------|------------------------------------------------------------------------------|
| 67  | File <i>file-name</i> read error<br>(NO.2100-36)                | A file could not be read.                                              | Check the specified file name.                                               |
| 68  | File <i>file-name</i> format error<br>(NO.2100-37)              | A file is specified in an invalid format in the ld or cm subcommand.   |                                                                              |
| 69  | Pname <i>file-name</i> not found<br>(NO.2100-38)                | The program name was not found.                                        | Review the program name.                                                     |
| 70  | Must specify address in text space<br>(NO.2100-39)              | Specify an address within a text space.                                | Review the specified address.                                                |
| 71  | Specified address is already set<br>(NO.2100-40)                | The specified address was already set.                                 |                                                                              |
| 72  | Must specify break point address<br>(NO.2100-41)                | Specify a breakpoint address.                                          | Review the specified address.                                                |
| 73  | Cannot get TCB (NO.2100-42)                                     | The ta subcommand could not fetch a TCB.                               | Collect data for examination, and then restart the development machine. (*1) |
| 74  | Cannot set break point beyond the max<br>(NO.2100-43)           | The maximum selectable number of breakpoints (5) is already specified. | Reset the breakpoints, and then try again.                                   |
| 75  | Cannot use ld sub command after RSSRCV set (NO.2100-44)         | An attempt was made to use the ld subcommand after RSSRCV was set.     | Perform batch loading by using the svrpl command.                            |
| 76  | Inconsistency detected %s<br>(NO.2100-46)                       | An inconsistency was found in the allocator management table.          | Collect data for examination, and then restart the development machine. (*1) |
| 77  | CM is not defined<br>(NO.2100-48)                               | The CM area is not defined.                                            | Review the specified options.                                                |
| 78  | DCM is not defined<br>(NO.2100-49)                              | The DCM area is not defined.                                           |                                                                              |
| 79  | Specified area is not defined for glb<br>(NO.2100-50)           | The specified split area is not a global area.                         | Review the specified split area name.                                        |
| 80  | Cannot specify another CM space<br>(NO.2100-51)                 | No other CM space can be specified.                                    | Review the specified parameters.                                             |
| 81  | Specified pgm number is out of range<br>(1 to 255) (NO.2100-52) | The specified program number is out of range.                          |                                                                              |
| 82  | Cannot load CM/DCM when PU is running (NO.2100-53)              | Loading to the CM is disabled when S10VE is in the RUN state.          | Check the S10VE status, and then try again.                                  |

| No. | Error message                                                                     | Explanation                                                                                   | User action                                                                                                    |
|-----|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| 83  | ADT channel is already set (NO.2100-56)                                           | The ADT is already set.                                                                       | Reset the ADT, and then run the command.                                                                       |
| 84  | Illegal break point address (laddr= logical address) (NO.2100-58)                 | The logical address set for a breakpoint is not registered as a program address.              | Restart the CPU, and then reset the breakpoint setting.                                                        |
| 85  | Specified raddr is not break point address (raddr= relative address) (NO.2100-59) | No breakpoint is set at the specified relative address (raddr).                               | Review the specified address.                                                                                  |
| 86  | Break task is not found (NO.2100-60)                                              | No task was found to be halted at a breakpoint.                                               | Confirm the breakpoint setup.                                                                                  |
| 87  | Specified name ( <i>program-name</i> ) is used break point (NO.2100-61)           | A breakpoint is set for the program name that is specified by the <code>ld</code> subcommand. | Reset the breakpoint, and then try again.                                                                      |
| 88  | Specified area is not initialize data area (NO.2100-62)                           | The specified address points to an area that contains no backup file.                         | Review the specified address.                                                                                  |
| 89  | Specified address is not initialize data area (NO.2100-63)                        | No backup file was found in the specified area.                                               | Review the specified area.                                                                                     |
| 90  | Specified name (%s) is not GLB/CM/DCM area (NO.2100-66)                           | The specified name does not represent a GLB or CM area.                                       | Specify a GLB or CM area.                                                                                      |
| 91  | Cannot access CM/DCM backupfile from NP or HP site (NO.2100-67)                   | The CM backup file cannot be accessed from the HP site.                                       | Access the CM backup file from the CP site.                                                                    |
| 92  | Specified sub command can use in HP site only (NO.2100-73)                        | The specified subcommand cannot be used in a site other than the HP site.                     | Specify the HP site.                                                                                           |
| 93  | Communication error (catch signal) (NO.2101-01)                                   | A signal was received.                                                                        | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 94  | Communication error (connection timeout) (NO.2101-02)                             | A timeout was generated.                                                                      |                                                                                                                |
| 95  | Communication error (connection refused) (NO.2101-03)                             | No RPC server exists.                                                                         |                                                                                                                |
| 96  | Communication error (connection cut) (NO.2101-04)                                 | The RPC server is disconnected.                                                               |                                                                                                                |
| 97  | Communication error (connection reset) (NO.2101-05)                               | A connection was reset.                                                                       |                                                                                                                |

APPENDIX D ERROR MESSAGES

(7/7)

| No. | Error message                                                               | Explanation                                       | User action                                                                                                    |
|-----|-----------------------------------------------------------------------------|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| 98  | Communication error (server closed) (NO.2101-06)                            | The RPC server is closed.                         | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 99  | Communication error (port busy) (NO.2101-07)                                | The line port is busy.                            | Wait until another user terminates communication, and then re-run the command.                                 |
| 100 | Communication error (bad socket specified) (NO.2101-08)                     | The specified socket is invalid.                  | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 101 | Communication error (socket creat error) (NO.2101-09)                       | A socket could not be created.                    |                                                                                                                |
| 102 | Communication error (no buffer) (NO.2101-10)                                | Memory could not be allocated.                    |                                                                                                                |
| 103 | Communication error (network not reached) (NO.2101-11)                      | The network is not connected.                     |                                                                                                                |
| 104 | Communication error (network down) (NO.2101-12)                             | The interface connected to the network is down.   |                                                                                                                |
| 105 | Communication error (port No error) (NO.2101-13)                            | A port number could not be fetched.               |                                                                                                                |
| 106 | Communication error (IP address error) (NO.2101-14)                         | An IP address could not be fetched.               |                                                                                                                |
| 107 | Communication error (memory attach failed) (NO.2101-15)                     | Shared memory could not be attached.              |                                                                                                                |
| 108 | Communication error (trace file cannot open) (NO.2101-16)                   | A trace file could not be opened.                 |                                                                                                                |
| 109 | Communication error (trace file cannot copy) (NO.2101-17)                   | A trace file could not be copied.                 |                                                                                                                |
| 110 | Communication error (fatal error) (NO.2101-18)                              | A fatal error was detected.                       |                                                                                                                |
| 111 | Communication error ( <i>library-name: error-number</i> ) (NO.2101-19) (*2) | An error occurred in an RPL or RRB library.       |                                                                                                                |
| 112 | Communication error (inter PU communication time out) (NO.2101-20)          | A timeout occurred during inter-PU communication. |                                                                                                                |
| 113 | Communication error ( <i>rc=%d</i> ) (NO.2101-21)                           | An RPC library error occurred.                    | Check the versions of the CPMS and RPDP. (*4)                                                                  |

(\*1) Data for examination is stored in the following files:

- %windir%\renix\etc\log\rpdp\_hce\RPDP\_S10VE
- %windir%\renix\etc\log\rpdp\_hce\1\_RPDP\_S10VE

(\*2) In the event of a communication error, see the following:

Meanings of communication error codes:

- |                                                                                                                                         |                                                                                       |
|-----------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| 0x11: The socket is invalid.                                                                                                            | 0x04: The frame creation memory could not be allocated.                               |
| 0x12: The IP address is invalid.                                                                                                        | 0x05: Data transmission failed.                                                       |
| 0x14: The storage area address is invalid<br>(0 specified, dta).                                                                        | 0x06: An error occurred during a wait for response<br>reception.                      |
| 0x15: The storage area address is invalid<br>(0 specified, wka).                                                                        | 0x07: The maximum retry count was exceeded, because the<br>response was not received. |
| 0x16: The size is invalid (smaller than<br>0 KB or larger than 16 KB).                                                                  | 0x08: Data reception failed.                                                          |
| 0x17: The size is invalid<br>(non-longword size).                                                                                       | 0x18: The storage area address is invalid<br>(0 specified, dmaia).                    |
| 0x03: The remote adapter type is invalid.                                                                                               | 0x19: The storage area address is invalid<br>(0 specified, reta).                     |
| 0x8000000X: Error report with a response (status code in the CPU control header)<br>X: Status code, 4: $\mu\Sigma 1000$ network not set |                                                                                       |
| 0xFFFFFFFF: The environment file settings contain an error.                                                                             |                                                                                       |

(\*3) Meaning of RPDP library error codes:

- 1: The system file could not be opened.
- 2: The system file could not be loaded.
- 3: The site directory path is too long.
- 4: The system route could not be fetched.

(\*4) Meaning of rc in RPC library errors:

- 22: An illegal request was generated.
- 23: No function corresponding to the CPMS exists.



(3) List of svrpl command error messages

(1/2)

| No. | Error message                                      | Explanation                                                    | User action                                                                                                                                                                                                          |
|-----|----------------------------------------------------|----------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | No sitename given for -u option                    | The site name is not specified.                                | Specify the site name.                                                                                                                                                                                               |
| 2   | No unitname given for -U option                    | The unit name is not specified.                                | Specify the unit name.                                                                                                                                                                                               |
| 3   | unknown RSSITE                                     | The specified site name was not found.                         | Confirm the specified site.                                                                                                                                                                                          |
| 4   | Site=%s not found                                  |                                                                |                                                                                                                                                                                                                      |
| 5   | Unit=%s not found                                  | The specified unit name was not found.                         | Confirm the specified unit.                                                                                                                                                                                          |
| 6   | %s cannot open                                     | The file could not be opened.                                  | Check whether the file is normal.                                                                                                                                                                                    |
| 7   | %s file access error                               | The file could not be accessed.                                |                                                                                                                                                                                                                      |
| 8   | Internal error (%s)                                | An internal error occurred.                                    | Try again.                                                                                                                                                                                                           |
| 9   | download file (%s) not found                       | The backup file to be downloaded was not found.                | Confirm the environment.                                                                                                                                                                                             |
| 10  | site (%s) lock busy                                | The site is being used by another process.                     | Try again.                                                                                                                                                                                                           |
| 11  | site (%s) lock error                               |                                                                |                                                                                                                                                                                                                      |
| 12  | communication error (%s, RC=0x%x, error address)   | A communication error occurred.                                | Identify the cause of the error in accordance with the RC (*1).<br>An ST# setting is incorrect (%s:rrw_rpl_p) or an Ethernet cable is not connected.<br>Check and correct the settings.                              |
| 13  | communication error (%s, RC=0x%x)                  |                                                                |                                                                                                                                                                                                                      |
| 14  | site (%s) allocator management tables modify error | An error occurred during an allocator management table update. | Use the svmkrestbl command to repair the allocator management table.                                                                                                                                                 |
| 15  | IP ADDRESS SET ERROR (RC=0x%x)                     | An error occurred during IP address setup.                     | Identify the cause of the error in accordance with the RC (*1).<br>After a change in the connected PCs, an IP address setting is incorrect or an Ethernet cable is not connected.<br>Check and correct the settings. |
| 16  | %s (slot=%d) NON EXIST                             | The slot was not found.                                        | Confirm the system generation information.                                                                                                                                                                           |
| 17  | File mapping error (%s)                            | The RPDP resource table could not be set up.                   | Try again.                                                                                                                                                                                                           |
| 18  | site (%s) unlock error                             | An error occurred when an attempt was made to unlock the site. |                                                                                                                                                                                                                      |

| No. | Error message                                                                                      | Explanation                                                             | User action                                                                                                                                                                              |
|-----|----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 19  | Can not specified NP or HP site (%s)                                                               | The specified site was an HP site.                                      | Specify a CP site.                                                                                                                                                                       |
| 20  | Usage:svrpl [{-u site -U unit} {-s}] [-all] [-r] [{-time -notime}] [-ROMSV   -NOROMSV] [-setpcsno] | -                                                                       | -                                                                                                                                                                                        |
| 21  | Can not get site information                                                                       | Site information could not be acquired.                                 | Check the status of the specified site. (*2)                                                                                                                                             |
| 22  | Can not load data until finish initializing module-hardware                                        | Data cannot be downloaded due to initialization of the module hardware. | After the hardware has been initialized, try again. (The STBY LED on the module blinks during hardware initialization. When hardware initialization is complete, the STBY LED turns on.) |
| 23  | Command I/F (command code=0x%x) time out                                                           | A command interface timeout error occurred. (*3)                        | Check the network connection status, and then try again.                                                                                                                                 |
| 24  | Not RPDUsers                                                                                       | This user does not have the RPDUsers authority.                         | Execute the command with a user who has the RPDUsers authority.                                                                                                                          |
| 25  | -NOROMSV cannot be specified with -setpcsno                                                        | -NOROMSV and -setpcsno cannot be specified together.                    | Review the specified options.                                                                                                                                                            |
| 26  | CPMS has not been downloaded. CPMS must be downloaded from BASE SYSTEM in advance.                 | The CPMS has not been downloaded.                                       | Download the CPMS from BASE SYSTEM/S10VE.                                                                                                                                                |

(\*1) In the event of a communication error, see the following:

Meanings of communication RCs:

- 0x11: The socket is invalid.
- 0x12: The IP address is invalid.
- 0x14: The storage area address is invalid (0 specified, dta).
- 0x15: The storage area address is invalid (0 specified, wka).
- 0x16: The size is invalid (smaller than 0 KB or larger than 16 KB).
- 0x17: The size is invalid (non-longword size).
- 0x03: The remote adapter type is invalid.
- 0x8000000X: Error report with a response (status code in the CPU control header)  
X: Status code, 4: μΣ1000 network not set
- 0xFFFFFFFF: The environment file settings contain an error.
- 0x04: The frame creation memory could not be allocated.
- 0x05: Data transmission failed.
- 0x06: An error occurred during a wait for response reception.
- 0x07: The maximum retry count was exceeded because the response was not received.
- 0x08: Data reception failed.
- 0x18: The storage area address is invalid (0 specified, dmaia).
- 0x19: The storage area address is invalid (0 specified, reta).

(\*2) Confirm the No. 41 file on page A-9.

(\*3) The following shows the meaning of command codes that are shown in the event of a command interface timeout error:

- 0x01C00000: Physical address access
- 0x11C00000: Time setting
- 0x21C00000: Batch ROM saving

(4) List of svcpuctl command error messages

| No. | Error message                                                                          | Explanation                                                     | User action                                                                                                                                                                                  |
|-----|----------------------------------------------------------------------------------------|-----------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | No sitename given for -u option                                                        | The unit name is not specified.                                 | Specify the unit name.                                                                                                                                                                       |
| 2   | unknown RSSITE                                                                         | The specified site name was not found.                          | Confirm the specified site.                                                                                                                                                                  |
| 3   | Site=%s not found                                                                      |                                                                 |                                                                                                                                                                                              |
| 4   | Internal error (%s)                                                                    | An internal error occurred.                                     | Try again.                                                                                                                                                                                   |
| 5   | site (%s) lock busy                                                                    | The site is being used by another process.                      |                                                                                                                                                                                              |
| 6   | site (%s) lock error                                                                   |                                                                 |                                                                                                                                                                                              |
| 7   | communication error (%s, RC=0x%x, error-address)                                       | A communication error occurred.                                 | Identify the cause of the error in accordance with the RC. (*) An ST# setting is incorrect (%s:rrw_rpl_p) or an Ethernet cable is not connected (%s:set_ip). Check and correct the settings. |
| 8   | communication error (%s, RC=0x%x)                                                      |                                                                 |                                                                                                                                                                                              |
| 9   | %s (slot=%d) NON EXIST                                                                 | The slot was not found.                                         | Confirm the system generation information.                                                                                                                                                   |
| 10  | site (%s) unlock error                                                                 | An error occurred when an attempt was made to unlock the site.  | Try again.                                                                                                                                                                                   |
| 11  | Site=%s is NP or HP site                                                               | The HP site is specified.                                       | Specify the CP site.                                                                                                                                                                         |
| 12  | Command I/F (command code=0x11C00000) time out                                         | A timeout error occurred in the command interface time setting. | Check the network connection status, and then try again.                                                                                                                                     |
| 13  | Not RPDUsers                                                                           | This user does not have the RPDUsers authority.                 | Execute the command with a user who has the RPDUsers authority.                                                                                                                              |
| 14  | Usage: svcpuctl [{-u site} {-s {-stop -run}}] [-time]<br>Usage: svcpuctl [-u site] -ss | -                                                               | -                                                                                                                                                                                            |

(\*) In the event of a communication error, see the following:

Meanings of communication RCs:

- |                                                                                                                               |                                                                                   |
|-------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| 0x11: The socket is invalid.                                                                                                  | 0x04: The frame creation memory could not be allocated.                           |
| 0x12: The IP address is invalid.                                                                                              | 0x05: Data transmission failed.                                                   |
| 0x14: The storage area address is invalid (0 specified, dta).                                                                 | 0x06: An error occurred during a wait for response reception.                     |
| 0x15: The storage area address is invalid (0 specified, wka).                                                                 | 0x07: The maximum retry count was exceeded because the response was not received. |
| 0x16: The size is invalid (smaller than 0 KB or larger than 16 KB).                                                           | 0x08: Data reception failed.                                                      |
| 0x17: The size is invalid (non-longword size).                                                                                | 0x18: The storage area address is invalid (0 specified, dmaia).                   |
| 0x03: The remote adapter type is invalid.                                                                                     | 0x19: The storage area address is invalid (0 specified, reta).                    |
| 0x8000000X: Error report with a response (status code in the CPU control header)<br>X: Status code, 4: μΣ1000 network not set |                                                                                   |
| 0xFFFFFFFF: The environment file settings contain an error.                                                                   |                                                                                   |

(5) Error messages displayed by the svelog command

| No. | Error message                                                               | Explanation                                                                 | User action                                                                  |
|-----|-----------------------------------------------------------------------------|-----------------------------------------------------------------------------|------------------------------------------------------------------------------|
| 1   | Usage: svelog [-u site] [-f {s m l}] [-logno] [+case] [-d fname] [-o fname] | An option is specified incorrectly.                                         | Specify the options correctly.                                               |
| 2   | Unknown RSSITE                                                              | The RSSITE environment variable is not set.                                 | Set the RSSITE environment variable.                                         |
| 3   | logno error. logno is 1-999                                                 | The specified log number is invalid.                                        | Check the log number.                                                        |
| 4   | unknown site ( <i>site-name</i> )                                           | The specified site could not be found.                                      | Check the site name.                                                         |
| 5   | communication error ( <i>error-code</i> )                                   | Transmission or reception failed between the development machine and S10VE. | Check the cause based on the error code. (*2)                                |
| 6   | memory allocate error                                                       | Memory could not be allocated.                                              | Try again.                                                                   |
| 7   | logno <i>log-number</i> : not found                                         | The error log that has the specified log number could not be found.         | Check the log number.                                                        |
| 8   | no error log.                                                               | No error log exists.                                                        | No error has occurred.                                                       |
| 9   | cannot open <i>file-name</i>                                                | A file could not be opened.                                                 | Collect data for examination, and then restart the development machine. (*1) |
| 10  | cannot read <i>file-name</i>                                                | An error was detected while a file was being read.                          |                                                                              |
| 11  | specified logno is not found                                                | The error log that has the specified log number could not be found.         | Check the log number.                                                        |
| 12  | eloghr : Invalid file name ( <i>XXXX</i> )                                  | File name <i>XXXX</i> is invalid.                                           | Specify the correct file name, and then re-run the command.                  |

(\*1) Data for examination is stored in the following files:

- %windir%\renix\etc\log\rpdp\_hce\RPDP\_S10VE
- %windir%\renix\etc\log\rpdp\_hce\1\_RPDP\_S10VE

(\*2) In the event of a communication error, see the following:

Meanings of communication error codes:

- |                                                                                                                               |                                                                                   |
|-------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| 0x11: The socket is invalid.                                                                                                  | 0x04: The frame creation memory could not be allocated.                           |
| 0x12: The IP address is invalid.                                                                                              | 0x05: Data transmission failed.                                                   |
| 0x14: The storage area address is invalid (0 specified, dta).                                                                 | 0x06: An error occurred during a wait for response reception.                     |
| 0x15: The storage area address is invalid (0 specified, wka).                                                                 | 0x07: The maximum retry count was exceeded because the response was not received. |
| 0x16: The size is invalid (smaller than 0 KB or larger than 16 KB).                                                           | 0x08: Data reception failed.                                                      |
| 0x17: The size is invalid (non-longword size).                                                                                | 0x18: The storage area address is invalid (0 specified, dmaia).                   |
| 0x03: The remote adapter type is invalid.                                                                                     | 0x19: The storage area address is invalid (0 specified, reta).                    |
| 0x8000000X: Error report with a response (status code in the CPU control header)<br>X: Status code, 4: μΣ1000 network not set |                                                                                   |
| 0xFFFFFFFF: The environment file settings contain an error.                                                                   |                                                                                   |

## (6) Error messages displayed by the svdhp command

(1/3)

| No. | Error message                                                                                                         | Explanation                                                       | User action                                                                                                                                                           |
|-----|-----------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | usage: svdhp [-u site] [+count]<br>[-on -off -stat]<br>[-d fname] [-o fname] [-all<br>[fname]] -f fname]<br>[-freeze] | An option is specified incorrectly.                               | Specify the options correctly.                                                                                                                                        |
| 2   | Unknown RSSITE                                                                                                        | The RSSITE environment variable is not set.                       | Set the RSSITE environment variable.                                                                                                                                  |
| 3   | No such site ( <i>site-name</i> )                                                                                     | The specified site was not found.                                 | Review the site name.                                                                                                                                                 |
| 4   | No such PUname ( <i>CPU-name</i> )                                                                                    | The specified CPU was not found.                                  | Review the CPU name.                                                                                                                                                  |
| 5   | specified CPU ( <i>CPU-name</i> ) is CP only                                                                          | Only CP is specified as a CPU.                                    |                                                                                                                                                                       |
| 6   | Some system constants are not defined                                                                                 | An undefined system constant exists.                              | Review the system constant definitions.                                                                                                                               |
| 7   | Bad realtime environment                                                                                              | The system environment contains an error.                         | Collect data for examination, and then restart the development machine. (*)                                                                                           |
| 8   | memory allocate error                                                                                                 | Memory could not be allocated.                                    | The error might have occurred because the available memory was temporarily insufficient. Verify that the available memory is sufficient, and then re-run the command. |
| 9   | cannot open <i>file-name</i>                                                                                          | A file could not be opened.                                       | Confirm the security and other conditions of files and directories.                                                                                                   |
| 10  | cannot read <i>file-name</i>                                                                                          | An error was detected while a file was being read.                |                                                                                                                                                                       |
| 11  | cannot write <i>file-name</i>                                                                                         | An error was detected while a file was being written.             |                                                                                                                                                                       |
| 12  | No such AREA (DHP_RD) in salmt                                                                                        | The DHP read area (DHP_RD) could not be found in the global area. | Collect data for examination, and then restart the development machine. (*)                                                                                           |
| 13  | Memory access error                                                                                                   | The S10VE memory could not be accessed.                           |                                                                                                                                                                       |

(2/3)

| No. | Error message                                   | Explanation                                     | User action                                                                                                                                                           |
|-----|-------------------------------------------------|-------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 14  | Memory allocation error (malloc, dhp read area) | A DHP read area could not be allocated.         | The error might have occurred because the available memory was temporarily insufficient. Verify that the available memory is sufficient, and then re-run the command. |
| 15  | Communication error (catch signal)              | A signal was received.                          | Check the network connection status, S10VE, and the IP address of the development machine, and then try again.                                                        |
| 16  | Communication error (connection timeout)        | A connection timeout was generated.             |                                                                                                                                                                       |
| 17  | Communication error (connection refused)        | No RPC server exists.                           |                                                                                                                                                                       |
| 18  | Communication error (connection cut)            | A connection was discontinued.                  |                                                                                                                                                                       |
| 19  | Communication error (connection reset)          | A connection was reset.                         |                                                                                                                                                                       |
| 20  | Communication error (server closed)             | The RPC server is closed.                       | Collect data for examination, and then restart the development machine. (*)                                                                                           |
| 21  | Communication error (port busy)                 | The line port is busy.                          | Wait until another user terminates communication, and then re-run the command.                                                                                        |
| 22  | Communication error (socket create error)       | A socket could not be created.                  | Check the network connection status, S10VE, and the IP address of the development machine, and then try again.                                                        |
| 23  | Communication error (no buffer)                 | Memory could not be allocated.                  |                                                                                                                                                                       |
| 24  | Communication error (network not reached)       | The network is not connected.                   |                                                                                                                                                                       |
| 25  | Communication error (network down)              | The interface connected to the network is down. |                                                                                                                                                                       |
| 26  | Communication error (port No error)             | A port number could not be fetched.             |                                                                                                                                                                       |
| 27  | Communication error (IP address error)          | An IP address could not be fetched.             |                                                                                                                                                                       |
| 28  | Communication error (memory attach failed)      | Shared memory could not be attached.            |                                                                                                                                                                       |
| 29  | Communication error (trace file cannot open)    | A trace file could not be opened.               |                                                                                                                                                                       |
| 30  | Communication error (trace file cannot copy)    | A trace file could not be copied.               |                                                                                                                                                                       |

| No. | Error message                     | Explanation                                                                   | User action                                                                                                    |
|-----|-----------------------------------|-------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| 31  | Communication error (fatal error) | A fatal error was detected.                                                   | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 32  | dhp data read error               | An error was detected when an attempt was made to read dhp trace data.        | Collect data for examination, and then restart the development machine. (*)                                    |
| 33  | Cannot dhp trace ON/OFF           | An error was detected when an attempt was made to exercise dhp trace control. |                                                                                                                |
| 34  | svdhp : Invalid file name (XXXX)  | File name XXXX is invalid.                                                    | Specify the correct file name, and then re-run the command.                                                    |
| 35  | svdhp : DHP data illegal          | The file data is abnormal.                                                    | Specify the correct file.                                                                                      |
| 36  | Not RPDUsers                      | This user does not have the RPDUsers authority.                               | Execute the command with a user who has the RPDUsers authority.                                                |

(\*) Data for examination is stored in the following files:

- %windir%\renix\etc\log\rpdp\_hce\RPDP\_S10VE
- %windir%\renix\etc\log\rpdp\_hce\1\_RPDP\_S10VE

(7) Error messages displayed by the svcpunow command

(1/3)

| No. | Error message                                     | Explanation                                                                  | User action                                                                                                                                                           |
|-----|---------------------------------------------------|------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | Usage: svcpunow [-u site] [-t second]             | An option is specified incorrectly.                                          | Specify the options correctly.                                                                                                                                        |
| 2   | Unknown RSSITE                                    | The RSSITE environment variable is not set.                                  | Set the RSSITE environment variable, and then try again.                                                                                                              |
| 3   | memory allocation error (malloc, puloadinfo area) | An area from which to read PU load ratio information could not be allocated. | The error might have occurred because the available memory was temporarily insufficient. Verify that the available memory is sufficient, and then re-run the command. |
| 4   | cannot get PU load information                    | A PU load ratio could not be fetched.                                        | Collect data for examination, and then restart the development machine. (*1)                                                                                          |
| 5   | Not available parameter                           | An unusable parameter was detected.                                          |                                                                                                                                                                       |
| 6   | No such AREA (puloadinfo) in salmt                | An area from which to read PU load ratio information could not be searched.  |                                                                                                                                                                       |
| 7   | Communication error (catch signal)                | A signal was received.                                                       | Check the network connection status, S10VE, and the IP address of the development machine, and then try again.                                                        |
| 8   | Communication error (connection timeout)          | A timeout was generated.                                                     |                                                                                                                                                                       |
| 9   | Communication error (connection refused)          | No RPC server exists.                                                        |                                                                                                                                                                       |
| 10  | Communication error (connection cut)              | The RPC server was disconnected.                                             |                                                                                                                                                                       |
| 11  | Communication error (connection reset)            | A connection was reset.                                                      |                                                                                                                                                                       |
| 12  | Communication error (server closed)               | The RPC server was closed.                                                   | Wait until other communication finishes, and then try again.                                                                                                          |
| 13  | Communication error (port busy)                   | The line port is busy.                                                       |                                                                                                                                                                       |
| 14  | Communication error (bad socket specified)        | A socket descriptor is specified incorrectly.                                |                                                                                                                                                                       |
| 15  | Communication error (socket create error)         | A socket could not be created.                                               | Check the network connection status, S10VE, and the IP address of the development machine, and then try again.                                                        |



APPENDIX D ERROR MESSAGES

(2/3)

| No. | Error message                                           | Explanation                                                                      | User action                                                                                                                                                           |
|-----|---------------------------------------------------------|----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16  | Communication error (no buffer)                         | Memory could not be allocated.                                                   | The error might have occurred because the available memory was temporarily insufficient. Verify that the available memory is sufficient, and then re-run the command. |
| 17  | Communication error (network not reached)               | The network is not connected.                                                    | Collect data for examination, and then restart the development machine. (*1) (*2)                                                                                     |
| 18  | Communication error (network down)                      | The interface connected to the network is down.                                  | Check the network connection status, S10VE, and the IP address of the development machine, and then try again.                                                        |
| 19  | Communication error (port No error)                     | A port number could not be fetched.                                              |                                                                                                                                                                       |
| 20  | Communication error (IP address error)                  | An IP address could not be fetched.                                              |                                                                                                                                                                       |
| 21  | Communication error (memory attach failed)              | Shared memory could not be attached.                                             |                                                                                                                                                                       |
| 22  | Communication error (trace file cannot open)            | A trace file could not be opened.                                                |                                                                                                                                                                       |
| 23  | Communication error (trace file cannot copy)            | A trace file could not be copied.                                                |                                                                                                                                                                       |
| 24  | Communication error (fatal error)                       | A fatal error was detected.                                                      |                                                                                                                                                                       |
| 25  | Communication error (cannot connection errno = %x) (*2) | A communication line could not be established.                                   |                                                                                                                                                                       |
| 26  | Communication error (RRB errno = %x) (*2)               | Memory could not be read.                                                        |                                                                                                                                                                       |
| 27  | Memory access error                                     | Memory in the S10VE could not be read or written.                                |                                                                                                                                                                       |
| 28  | target status error                                     | An S10VE command support task could not be started.                              | Collect data for examination, and then restart the development machine. (*1)                                                                                          |
| 29  | Cannot get TCB                                          | TCB information could not be read.                                               | Re-run the command.                                                                                                                                                   |
| 30  | command is already execution                            | The command could not be run because another user was measuring a PU load ratio. |                                                                                                                                                                       |

| No. | Error message                                    | Explanation                                     | User action                                                       |
|-----|--------------------------------------------------|-------------------------------------------------|-------------------------------------------------------------------|
| 31  | No sitename given for -u option                  | A site name is missing.                         | Verify the data that can be entered, and then re-run the command. |
| 32  | Site=%s not found                                | No such site could be found.                    |                                                                   |
| 33  | PU load measuring period error [second = 1-3600] | A measurement time is specified incorrectly.    |                                                                   |
| 34  | Not RPDUsers                                     | This user does not have the RPDUsers authority. | Execute the command with a user who has the RPDUsers authority.   |

(\*1) Data for examination is stored in the following files:

- %windir%\renix\etc\log\rpdp\_hce\RPDP\_S10VE
- %windir%\renix\etc\log\rpdp\_hce\1\_RPDP\_S10VE

(\*2) In the event of a communication error, see the following:

- |                                                                                  |                                                                                   |
|----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| 0x11: The socket is invalid.                                                     | 0x04: The frame creation memory could not be allocated.                           |
| 0x12: The IP address is invalid.                                                 | 0x05: Data transmission failed.                                                   |
| 0x14: The storage area address is invalid (0 specified, dta).                    | 0x06: An error occurred during a wait for response reception.                     |
| 0x15: The storage area address is invalid (0 specified, wka).                    | 0x07: The maximum retry count was exceeded because the response was not received. |
| 0x16: The size is invalid (smaller than 0 KB or larger than 16 KB).              | 0x08: Data reception failed.                                                      |
| 0x17: The size is invalid (non-longword size).                                   | 0x18: The storage area address is invalid (0 specified, dmaia).                   |
| 0x03: The remote adapter type is invalid.                                        | 0x19: The storage area address is invalid (0 specified, reta).                    |
| 0x8000000X: Error report with a response (status code in the CPU control header) |                                                                                   |
| X: Status code, 4: μΣ1000 network not set                                        |                                                                                   |
| 0xFFFFFFFF: The environment file settings contain an error.                      |                                                                                   |

(8) Error messages displayed by the svtimex command

(1/3)

| No. | Error message                                        | Explanation                                                                  | User action                                                                                                                                                           |
|-----|------------------------------------------------------|------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | Usage: svtimex [-u site] [tname] [-t second] [tn]    | An option is specified incorrectly.                                          | Specify the options correctly.                                                                                                                                        |
| 2   | Unknown RSSITE                                       | The RSSITE environment variable is not set.                                  | Set the RSSITE environment variable, and then try again.                                                                                                              |
| 3   | memory allocation error (malloc, taskloadinfo area)  | An area from which to read PU load ratio information could not be allocated. | The error might have occurred because the available memory was temporarily insufficient. Verify that the available memory is sufficient, and then re-run the command. |
| 4   | cannot get task load information                     | A task load ratio could not be fetched.                                      | Collect data for examination, and then restart the development machine. (*1)                                                                                          |
| 5   | cannot get task load information<br>Taskname=%s (%s) | A task load ratio could not be fetched. (The task is identifiable.)          |                                                                                                                                                                       |
| 6   | Not available parameter                              | An unusable parameter was detected.                                          |                                                                                                                                                                       |
| 7   | No such AREA (puploadinfo) in salmt                  | An area from which to read task load ratios could not be allocated.          |                                                                                                                                                                       |
| 8   | Communication error (catch signal)                   | A signal was received.                                                       | Check the network connection status, S10VE, and the IP address of the development machine, and then try again.                                                        |
| 9   | Communication error (connection timeout)             | A timeout was generated.                                                     |                                                                                                                                                                       |
| 10  | Communication error (connection refused)             | No RPC server exists.                                                        |                                                                                                                                                                       |
| 11  | Communication error (connection cut)                 | The RPC server was disconnected.                                             |                                                                                                                                                                       |
| 12  | Communication error (connection reset)               | A connection was reset.                                                      |                                                                                                                                                                       |
| 13  | Communication error (server closed)                  | The RPC server is closed.                                                    |                                                                                                                                                                       |
| 14  | Communication error (port busy)                      | The line port is busy.                                                       | Wait until other communication finishes, and then try again.                                                                                                          |

| No. | Error message                                           | Explanation                                     | User action                                                                                                    |
|-----|---------------------------------------------------------|-------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| 15  | Communication error (bad socket specified)              | A socket descriptor is specified incorrectly.   | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 16  | Communication error (socket create error)               | A socket could not be created.                  |                                                                                                                |
| 17  | Communication error (no buffer)                         | Memory could not be allocated.                  |                                                                                                                |
| 18  | Communication error (network not reached)               | The network is not connected.                   | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 19  | Communication error (network down)                      | The interface connected to the network is down. |                                                                                                                |
| 20  | Communication error (port No error)                     | A port number could not be fetched.             |                                                                                                                |
| 21  | Communication error (IP address error)                  | An IP address could not be fetched.             |                                                                                                                |
| 22  | Communication error (memory attach failed)              | Shared memory could not be attached.            |                                                                                                                |
| 23  | Communication error (trace file cannot open)            | A trace file could not be opened.               | Check the network connection status, S10VE, and the IP address of the development machine, and then try again. |
| 24  | Communication error (trace file cannot copy)            | A trace file could not be copied.               |                                                                                                                |
| 25  | Communication error (fatal error)                       | A fatal error was detected.                     |                                                                                                                |
| 26  | Communication error (cannot connection errno = %x) (*2) | A communication line could not be established.  |                                                                                                                |
| 27  | Communication error (RRB errno = %x) (*2)               | Memory could not be read.                       |                                                                                                                |
| 28  | Memory access error                                     | Failed to read and write memory data.           | Collect data for examination, and then restart the development machine. (*1)                                   |
| 29  | target status error                                     | A command support task could not be started.    |                                                                                                                |
| 30  | Cannot get TCB                                          | TCB information could not be read.              |                                                                                                                |

| No. | Error message                                  | Explanation                                                                       | User action                                                       |
|-----|------------------------------------------------|-----------------------------------------------------------------------------------|-------------------------------------------------------------------|
| 31  | command is already execution                   | The command could not be run because another user was measuring task load ratios. | Re-run the command.                                               |
| 32  | No sitename given for -u option                | A site name is missing.                                                           | Verify the data that can be entered, and then re-run the command. |
| 33  | Site=%s not found                              | The specified site could not be found.                                            |                                                                   |
| 34  | Task measuring period error [second = 1-86400] | A measurement time is specified incorrectly.                                      |                                                                   |
| 35  | taskname or number set data over (max=10)!!    | More than 10 task names or task numbers are specified.                            |                                                                   |
| 36  | taskname or number error (%s)                  | A task name or task number is specified incorrectly.                              |                                                                   |
| 37  | %s (%s) task non exist or unmatched            | The specified task was not registered in both the development machine and S10VE.  | Register the task in both the development machine and S10VE.      |
| 38  | Not RPDUsers                                   | This user does not have the RPDUsers authority.                                   | Execute the command with a user who has the RPDUsers authority.   |

(\*1) Data for examination is stored in the following files:

- %windir%\renix\etc\log\rpdp\_hce\RPDP\_S10VE
- %windir%\renix\etc\log\rpdp\_hce\1\_RPDP\_S10VE

(\*2) In the event of a communication error, see the following:

- |                                                                                  |                                                                                   |
|----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| 0x11: The socket is invalid.                                                     | 0x04: The frame creation memory could not be allocated.                           |
| 0x12: The IP address is invalid.                                                 | 0x05: Data transmission failed.                                                   |
| 0x14: The storage area address is invalid (0 specified, dta).                    | 0x06: An error occurred during a wait for response reception.                     |
| 0x15: The storage area address is invalid (0 specified, wka).                    | 0x07: The maximum retry count was exceeded because the response was not received. |
| 0x16: The size is invalid (smaller than 0 KB or larger than 16 KB).              | 0x08: Data reception failed.                                                      |
| 0x17: The size is invalid (non-longword size).                                   | 0x18: The storage area address is invalid (0 specified, dmaia).                   |
| 0x03: The remote adapter type is invalid.                                        | 0x19: The storage area address is invalid (0 specified, reta).                    |
| 0x8000000X: Error report with a response (status code in the CPU control header) |                                                                                   |
| X: Status code, 4: μΣ1000 network not set                                        |                                                                                   |
| 0xFFFFFFFF: The environment file settings contain an error.                      |                                                                                   |

(9) svdatagen command error messages

(1/2)

| No. | Error message                           | Explanation                                                                             | User action                                                                     | Restrictions                                                        |
|-----|-----------------------------------------|-----------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|---------------------------------------------------------------------|
| -   | Usage: svdatagen [-u site] file         | An error was found in the command startup format.                                       | Review the format of the specified options.                                     |                                                                     |
| -   | Cannot open input file (%s)             | The input file cannot be opened.                                                        | Review the specified file path and access rights.                               |                                                                     |
| -   | Cannot open include file (%s)           | The include file cannot be opened.                                                      | Review the specified include file.                                              | Preprocessor                                                        |
| -   | sitename is too long                    | A site name of more than 14 characters is specified.                                    | Review the specified site name.                                                 |                                                                     |
| -   | No such site (%s)                       | The specified site does not exist.                                                      |                                                                                 |                                                                     |
| -   | No type nor storage class               | The declaration statement does not include a type specifier or storage class specifier. | Review the declaration statement.                                               | Restriction (14)                                                    |
| -   | Illegal token (%s)                      | A syntax error was found or an unsupported syntax is being used.                        | Review the input file data.                                                     | Restrictions (7), (9), (10), (11), (12), (13), (15), (16), and (17) |
| -   | Multiple storage classes                | An error was found in the specified storage class specifier.                            | Review the storage class specifier.                                             | Restriction (3)                                                     |
| -   | Unexpected token appeared (%s)          | An unsupported syntax was detected.                                                     | Review the input file data.                                                     | Restrictions (2), (3), (4), (5), (6), and (10)                      |
| -   | Illegal array size (%s)                 | The array element count cannot be omitted.                                              | Review the array.                                                               | Restriction (8)                                                     |
| -   | Invalid initializer (%s)                | A non-convertible initial value appeared in the variable type.                          | Review the initial value.                                                       | Initial value                                                       |
| -   | Illegal type combination                | An incorrect combination of type specifiers was made.                                   | Review the type specifier.                                                      | Restriction (4)                                                     |
| -   | #define expected. (%s)                  | A specification other than #define appeared in the include file.                        | Review the include file data.                                                   | Preprocessor                                                        |
| -   | Invalid define value (%s)               | The specification of the define value is incorrect.                                     | Review #define. Check whether a signed octal or hexadecimal number is included. | Preprocessor                                                        |
| -   | EOF encountered in a comment            | The end of the comment was not found.                                                   | Review the comment.                                                             | Restriction (18)                                                    |
| -   | Too few initializer (%s)                | Initial values of array and structure are not enclosed by curly brackets ({}).          | Review the initial values of the array and structure.                           | Restriction (17)                                                    |
| -   | Too many initializer (%s)               | Too many initial values exist.                                                          | Review the initial values.                                                      |                                                                     |
| -   | Incomplete tag used in declaration (%s) | An undefined structure tag is being used.                                               | Review the declaration of the structure.                                        | Restriction (4)                                                     |

The *Restrictions* column shows the correspondence with 4.4 Data Generator in PART 1.

For details, see the applicable restrictions.

Restriction: (b) Differences from C language declaration statements and restrictions

Preprocessor: (e) Restrictions on preprocessor functions

Initial value: (f) Specifications of initial value type conversion

| No. | Error message                      | Explanation                                                                                   | User action                                                                                                                                                                             | Restrictions |
|-----|------------------------------------|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| -   | Illegal void type                  | An initial value is set for void-type variables (not a pointer).                              | Review the specification of the void type.                                                                                                                                              |              |
| -   | Undeclared name (%s)               | The specified define value or GLB or VAL name was not found.                                  | Review the input file data. Check the GLB and VAL registration of the site.                                                                                                             |              |
| -   | Site is not specified              | An attempt was made to resolve addresses of GLB and VAL names, but the site is not specified. | Specify the site, and then try again.                                                                                                                                                   |              |
| -   | Cannot alloc memory                | Memory cannot be allocated by using malloc.                                                   | Check the amount of available memory, and then try again.                                                                                                                               |              |
| -   | Allocator management table is busy | The allocator management table is being used by another command.                              | Wait until the other command finishes, and then try again.                                                                                                                              |              |
| -   | Cannot make temporary file (%s)    | Failed to generate a temporary file name.                                                     | Check the output file creation directory, the access rights of the directory indicated by the environment variable (%TMP%), and the amount of available disk space, and then try again. |              |
| -   | Cannot open temporary file (%s)    | Failed to generate a temporary file.                                                          |                                                                                                                                                                                         |              |
| -   | Cannot write output file           | Failed to write the output file.                                                              |                                                                                                                                                                                         |              |
| -   | Cannot rename (%s)                 | Failed to rename the output file.                                                             |                                                                                                                                                                                         |              |

The *Restrictions* column shows the correspondence with 4.4 Data Generator in PART 1.

For details, see the applicable restrictions.

Restriction: (b) Differences from C language declaration statements and restrictions

Preprocessor: (e) Restrictions on preprocessor functions

Initial value: (f) Specifications of initial value type conversion

## APPENDIX E NOTES ON USING RPDP

### (1) Recovery from suspension of processing by the `ld` subcommand of the `svdebug` command

When processing by the `ld` subcommand of the `svdebug` command is suspended due to, for example, a communication error, an inconsistency might arise between the memory in the S10VE and the backup files for the site on the development machine. To prevent this, RPDP recovers the `ld` subcommand. During recovery processing, the suspended `ld` subcommand is re-run to place the subcommand in the state after execution. Recovery processing is performed when the RPDP command is run or the development machine starts up. If the communication error persists, however, an error message such as `Communication error (connection timeout)` or `Communication error (inter-PU communication timeout)` appears, preventing the use of RPDP commands. To resolve this problem, perform the following operation:

#### Operation

```
svsitecntl -rssrcv site-name
```

After performing this operation, commands can be run without the need for RPDP to perform recovery processing. Note, however, that because the `ld` subcommand has not yet been recovered, no other `ld` subcommand can be run for the site after the operation.

The error message is as follows:

```
Cannot use ld sub comandnd after RSSRCV set (NO.2100-44)
```

This restriction continues until the `svrpl` command is run for downloading to the site.



(2) Notes on use

- When transferring a text file to be entered into the debugger (`svdebug`) from another machine to the current machine via FTP, be sure to enable ASCII mode.

Restrictions:

- The following names are reserved. These names, including those followed by extensions such as `.c`, `.obj`, and `.txt`, cannot be used as site names, unit names, directory names, or file names.

- |        |        |       |
|--------|--------|-------|
| • AUX  | • CON  | • NUL |
| • COM1 | • LPT1 | • PRN |
| • COM2 | • LPT2 |       |
| • COM3 | • LPT3 |       |
| • COM4 | • LPT4 |       |
| • COM5 | • LPT5 |       |
| • COM6 | • LPT6 |       |
| • COM7 | • LPT7 |       |
| • COM8 | • LPT8 |       |
| • COM9 | • LPT9 |       |

- A file on another disk cannot be specified by the `ld`, `cm`, or `sv` subcommand of the `svdebug` command with the `-f` option specified.

## APPENDIX F MAP DISPLAY FORMAT

The following map information is output:

- (1) Header and footer
- (2) Global area information
- (3) Split area information
- (4) Secondary partition area information
- (5) Program information
- (6) Subprogram information
- (7) Task information
- (8) Global information
- (9) VAL information
- (10) IRSUB entry information
- (11) IRGLB entry information
- (12) ULSUB entry information
- (13) Information about how much physical memory is available

Map information output format

Map information can be output in the following formats:

- (1) Hierarchical map output
- (2) Address-order list output
- (3) Name-order list output
- (4) Numerical-order list output
- (5) Specified name output

The hierarchical map output format is used to hierarchically output map information about resources arranged in a logical space for individual global or split areas.

The listing output formats are used to output specified information in address order, name order, or numerical order.

The name of a resource can also be specified to output information about that name.

The following describes the map information output formats.

The underlined portions in the following display formats are the output map information, which varies with the map output target.

(1) Header and footer field

The map information is to be output with a header attached to the beginning and a footer attached to the end.

The header and footer formats are as follows.

(a) Header

```

** allocator map **                               YYYY/MM/DD hh:mm:ss
site name = site
    
```

**\*\* allocator map \*\*:**  
 Displays a header string.

**\*\* allocator map \*\*:** This header is used for normal map output.  
**\*\* allocator map (CON) \*\*:** This header is used for controller logical spacemap output (when -CON is specified).

**YYYY/MM/DD hh:mm:ss:**  
 Displays the time at which the map output command (svmap) was started.  
**YYYY:** Year (4-digit)  
**MM:** Month  
**DD:** Day  
**hh:mm:ss:** Hour, minute, and second

**site:** Displays the name of a site that is targeted for map information display.

(b) Footer

```

** map output end **
    
```

(2) Global area information

The map of global areas defined at the time of system generation is displayed.  
 The start address of the logical space of a global area is fixed.

| < garea >    |              |                     |                    |
|--------------|--------------|---------------------|--------------------|
| <i>gname</i> | <i>laddr</i> | <i>paddr</i>        | <i>size</i>        |
| \$MAP        | 20000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$CPMS       | 28000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$TASK       | 30000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$GLBR       | 40000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$GLBRW      | 50000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$IRSUB      | 60000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$CM         | 70000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$DCM        | 75000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$LADDER     | 78000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$USRFUNC    | 7B000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |
| \$HIFLOW     | 7C000000     | <u><i>paddr</i></u> | <u><i>size</i></u> |

- gname*: Name of a global area.
- laddr*: Logical address of the beginning of a global area.
- paddr*: Physical address of the beginning of a global area.
- size*: Size of a global area.

Table A-3 Real-Time Source Management Status

| Symbol | Status      | Meaning                                                                                                                          |
|--------|-------------|----------------------------------------------------------------------------------------------------------------------------------|
| @      | not-build   | Loaded into a backup file only.                                                                                                  |
| +      | defined-POC |                                                                                                                                  |
| .      | defined     | Loaded into a backup file as well as the memory of a real machine.                                                               |
| -      | defined-CON | Loaded into the memory of a real machine only.<br>After a download, deleted only from the development machine.                   |
| *      | unmatch     | Loaded into a backup file and the memory of a real machine, but mismatched.                                                      |
| _      | non_exist2  | Downloaded without running <code>dload</code> for an IRSUB and a built-in subprogram for which <code>dbuild</code> has been run. |

When -CON is specified, the display does not show `s`, `date`, `lddate`, or `svdate`.



(4) Secondary partition area information  
Information about secondary partition areas is displayed.

```

< sarea >
garea/aname/sname          raddr  size  laddr  date  lddate  svdate
garea/aname/sname         s k raddr size laddr YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
garea/aname/_____         raddr size laddr

```

*garea*: Shows the GAREA name of the parent global area.

The meanings of the displayed GAREA names are the same as those described in (3) *Split area information*.

*aname*: Shows the name of a split area.

*sname*: Shows the name of a secondary partition area.

A blank secondary partition area name field represents an unoccupied area.

*s*: Indicates the resource status.

For details on the resource status, see Table A-3.

*k*: Indicates the owner type (s: system; u: user).

*raddr*: Shows the relative byte address of the beginning of a secondary partition area in relation to the beginning of a split area in 8-digit fixed hexadecimal notation.

*size*: Shows the size of a secondary partition area in 8-digit fixed hexadecimal notation.

*laddr*: Shows the start logical address of a secondary partition area in 8-digit fixed hexadecimal notation.

*date*: Shows the time at which a secondary partition area was allocated by `svdfs` or the time at which a program or subprogram was loaded into a backup file by `svload`.

*lddate*: Shows the time of downloading to the S10VE memory.

When no such download is complete, this field is blank.

*svdate*: Shows the time at which the `sv` subcommand of the debugger was used for storage to a backup file.

When no such storage operation is complete, this field is blank.

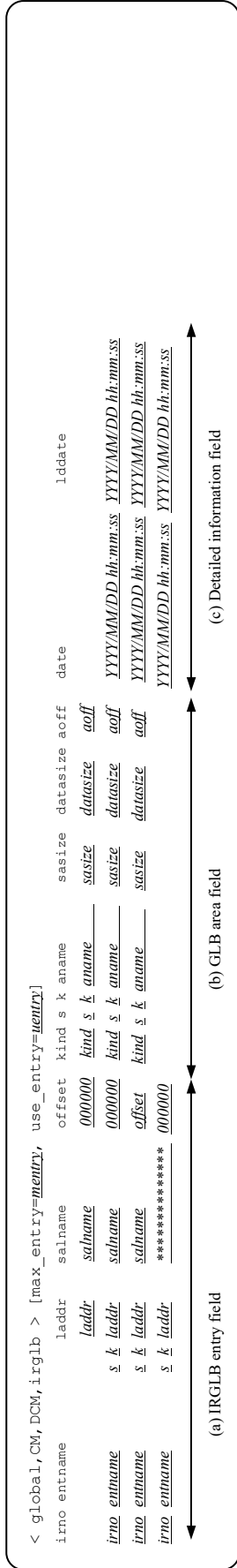
The display shows `date`, `lddate`, and `svdate` only when the `-f` option is specified (`s`, `date`, `lddate`, and `svdate` do not appear on the display when `-CON` is specified).







(8) Global information  
Information about global data (GLB, CM, and DCM) is displayed.

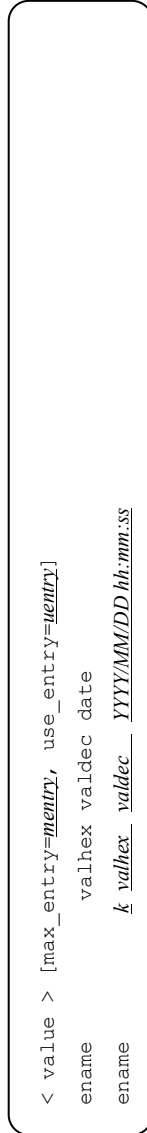


The global information display format is the same as for (11) IRGLB entry information. For the meaning of each field, see (11) IRGLB entry information.

Global information and IRGLB entry information have the following two differences:

- By default, sorting is performed by global name when global data is specified (-g) or by IRGLB number when IRGLB entry information is specified (-irg).
- When information is displayed with a name specified, the specified name is handled as a global name if global data is specified (-g). For an IRGLB entry (-irs), the specified number is handled as an IRGLB number.

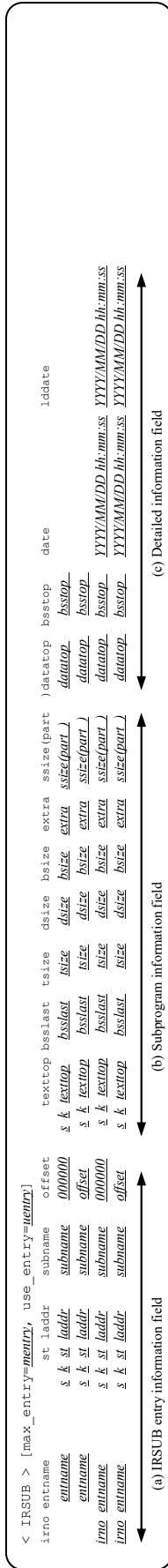
(9) VAL information  
Information related to values (VAL) is displayed.



- ename*: Shows the name of a value.
- k*: Indicates the owner type (*s*: system; *u*: user).
- valhex*: Shows a value in 8-digit, fixed-length, fixed hexadecimal notation.
- valdec*: Shows a value in 10-digit, variable-length, fixed decimal notation.
- date*: Shows the registration time.
- entry*: Shows the number of registrable VAL entries in 6-digit fixed decimal notation.
- entry*: Shows the number of currently used VAL entries in 6-digit fixed decimal notation.

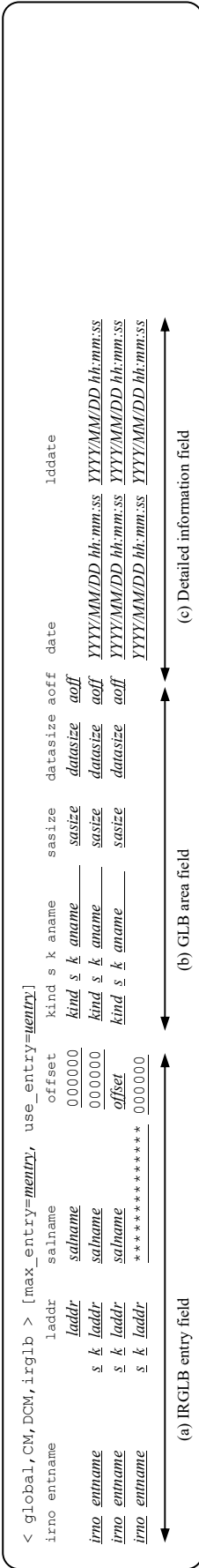
The display shows date only when the -f option is specified.

(10) IRSUB entry information  
Information about IRSUB entries is displayed.



- mentry:** Shows the number of registrable IRSUB entries in 6-digit fixed decimal notation.
- umentry:** Shows the number of currently used IRSUB entries in 6-digit fixed decimal notation.
- (a) IRSUB entry information field
- irno:** Indirect link table entry number.  
If the IRSUB is not built, the `irno` field is blank.
  - entname:** IRSUB entry name.
  - s:** Shows the status of an IRSUB entry.  
For details on the resource status, see Table A-3.
  - k:** Indicates the owner type (`s`: system; `u`: user).  
(If the IRSUB is not built, this field is blank.)
  - st:** Shows the IRSUB entry type and assignment.
  - il:** Indicates the top entry function of an IRSUB that is not built.
  - ml:** Indicates the multi-entry function of an IRSUB that is not built.
  - ib:** Indicates the top entry function of an IRSUB that is built.
  - mb:** Indicates the multi-entry function of an IRSUB that is built.
  - laddr:** Shows the logical space address of an entry point in 8-digit fixed hexadecimal notation.
  - subname:** Shows the name of a subprogram that contains an entry.
  - offset:** Shows the relative offset between the beginning of a subprogram and an entry point in 6-digit fixed hexadecimal notation.
- (b) Subprogram information field
- s:** Indicates the status of a subprogram.  
For details on the resource status, see Table A-3.
  - k:** Indicates the owner type (`s`: system; `u`: user).
- (c) Detailed information field
- texttop:** Shows the start logical address of a text part in 8-digit fixed hexadecimal notation.
  - bsplast:** Shows the end logical address of a BSS part in 8-digit fixed hexadecimal notation.  
The end logical address is a value that includes the redundancy byte size (*extra*).
  - tsize:** Shows the size of a text part in 6-digit fixed hexadecimal notation.
  - dsize:** Shows the size of a data part in 6-digit fixed hexadecimal notation.
  - bsize:** Shows the size of a BSS part in 6-digit fixed hexadecimal notation.
  - extra:** Shows the redundancy byte size specified for a load operation in 6-digit fixed hexadecimal notation.
  - ssize (part):** Shows the size of a stack in 6-digit fixed hexadecimal notation.  
The (*part*) section shows the stack size for use by the subprogram specified for the loader.
- (c) Detailed information field
- datatop:** Shows the start logical address of a data part in 8-digit fixed hexadecimal notation.
  - bstopt:** Shows the start logical address of a BSS part in 8-digit fixed hexadecimal notation.
  - date:** Shows the time at which a build was performed with `svbuild`. If no such build was performed, this field is blank.
  - lddate:** Shows the time of downloading to the S10VE memory.  
If no such download is complete, this field is blank.
- The detailed information field appears only when the `-f` option is specified.

(11) IRGLB entry information



*mentry*: Shows the number of registrable GLB (including CM and DCM) entries in 6-digit fixed decimal notation.

*umentry*: Shows the number of currently used GLB (including CM and DCM) entries in 6-digit fixed decimal notation (including the 14 GLB entries for use by the OS).

- (a) IRGLB entry field
- irno*: Indirect link table entry number.  
(If the entry is not built as an IRGLB, this field is blank.)
  - entname*: IRGLB entry name.  
(If the entry is not built as an IRGLB, this field is blank.)
  - s*: Indicates the IRGLB entry status.  
For details on the resource status, see Table A-3.
  - k*: (If the entry is not built as an IRGLB, this field is blank.) Indicates the owner type (*s*: system; *u*: user).
  - laddr*: Shows the logical space address of an entry point in 8-digit fixed hexadecimal notation.  
(If the entry is not built as an IRGLB, this field shows the GLB (sarea) address.)
  - salname*: Shows the name of a secondary partition area containing an entry.  
If the entry address is given as an absolute address, this field displays a string of asterisks (\*).
  - offset*: Shows the relative offset between the beginning of a secondary partition area and an entry point in 6-digit fixed hexadecimal notation.
- (b) GLB area field
- kind*: Secondary partition area type.
  - glbi*: Represents a read/write global area with an initial value.

- glbw*: Represents a read/write global area without an initial value.
- glbr*: Represents a read-only global area with an initial value.
- cmi*: Represents an inter-PU (processor) shared memory area with an initial value.
- cmw*: Represents an inter-PU (processor) shared memory area without an initial value.
- dcmi*: Represents a duplexed (inter-controller) shared memory area with an initial value.
- dcmw*: Represents a duplexed (inter-controller) shared memory area without an initial value.
- s*: Indicates the GLB status.  
For details on the resource status, see Table A-3.
- k*: Indicates the owner type (*s*: system; *u*: user).
- aname*: Shows the name of a parent split area.
- sasize*: Indicates the area size in 8-digit fixed hexadecimal notation.
- datasize*: Indicates the data size in 8-digit fixed hexadecimal notation.
- aoff*: Shows the relative byte address in relation to the beginning of a split area in 8-digit fixed hexadecimal notation.

- (c) Detailed information field
- date*: Shows the time at which a build was performed to create an IRGLB with `svirglb` or `svdfs -e`.  
If an IRGLG is not built, this field is blank.
  - lddate*: Shows the time of downloading to the SIOVE memory.  
If no such download is complete, this field is blank.
- The detailed information field appears only when the `-f` option is specified.

## (12) ULSUB entry information

```

< ULSUB >
pnt typ ent  subtype      texttop bsslast tsize dsize bsize extra ssize(part ) datatop  bssstop_  YYYY/MM/DD hh:mm:ss  YYYY/MM/DD hh:mm:ss
pnt typ ent  b  subtype      s  k  texttop  bsslast  tsize  dsize  bsize  extra  ssize(part )  datatop_  bssstop_  YYYY/MM/DD hh:mm:ss  YYYY/MM/DD hh:mm:ss

```

- pnt:** Indicates the incorporation point of a built-in subroutine.
- typ:** Indicates the type (OS or user) of a built-in subroutine.
- ent:** Indicates the entry number of a built-in subroutine.
- b:** Indicates the build status of a built-in subroutine.  
For details on the resource status, see Table A-3.
- subtype:** Shows the name of a built-in subroutine.
- s:** Indicates the status of a built-in subroutine.
- k:** For details on the resource status, see Table A-3.
- texttop:** Indicates the owner type (*s*: system; *u*: user).
- bsslast:** Shows the start logical address of a text part in 8-digit fixed hexadecimal notation.  
Shows the end logical address of a BSS part in 8-digit fixed hexadecimal notation.  
The end logical address is a value including the redundancy byte size (*extra*).
- tsize:** Shows the size of a text part in 6-digit fixed hexadecimal notation.
- dsize:** Shows the size of a data part in 6-digit fixed hexadecimal notation.
- bsize:** Shows the size of a BSS part in 6-digit fixed hexadecimal notation.
- extra:** Shows the redundancy byte size specified for a load operation in 6-digit fixed hexadecimal notation.
- ssize(part):** Shows the size of a stack in 6-digit fixed hexadecimal notation.  
The (*part*) section shows the stack size for use by the subprogram specified for the loader.
- datatop:** Shows the start logical address of a data part in 8-digit fixed hexadecimal notation.
- bssstop:** Shows the start logical address of a BSS part in 8-digit fixed hexadecimal notation.
- date:** Shows the time at which a build was performed to create a built-in subroutine with `svbuild`.  
If no such build was performed, this field indicates the time at which a load operation was performed.
- lddate:** Shows the time of downloading to the S10VE memory.  
If no such download is complete, this field is blank.

The detailed information field appears only when the `-f` option is specified.

(13) Information about how much physical memory is available

```
< physical memory >
garea      use      free      total
$TASK     xxx Kbyte  yyy Kbyte  zzz Kbyte
$GLBR     xxx Kbyte  yyy Kbyte  zzz Kbyte
$GLBRW    xxx Kbyte  yyy Kbyte  zzz Kbyte
$IRSUB    xxx Kbyte  yyy Kbyte  zzz Kbyte
$CM       xxx Kbyte  yyy Kbyte  zzz Kbyte
$DCM     xxx Kbyte  yyy Kbyte  zzz Kbyte
```

use: Size of the physical memory used by a GAREA.  
 free: Size of the physical memory that is free for a GAREA.  
 total: Physical memory size that was allocated to a GAREA at the time of site construction.

(14) Hierarchical map output  
 The hierarchical map output is generated in the following format.  
 (a) garea/area hierarchical map (with +gn, gname, -G, and -a specified)

```
** allocator map **
site name = site
< garea >
gname      laddr      paddr      size
gname      laddr      paddr      size
< area >
garea/aname      s k      raddr      size      laddr      bkupfile      svdate
garea/aname      s k      raddr      size      laddr      bkupfile      YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
garea/_____      raddr      size      laddr
** map output end **
```

Note: The display shows date, lddate, and svdate only when the -f option is specified.

## (b) garea/area/sarea hierarchical map (with +gn, gname, -G, -a, and -e specified)

```

** allocator map **
site name = site
< garea >
gname      laddr  paddr  size
gname      laddr  paddr  size
< area >
garea/aname  raddr  size  bkupfile  date  lddate  svdate
garea/aname  s k raddr size bkupfile YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
garea/ _____
< sarea >
garea/aname/sname  raddr  size  bkupfile  date  lddate  svdate
garea/aname/sname  s k raddr size bkupfile YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
garea/aname/ _____
** map output end **

```

Note: The display shows date, lddate, and svdate only when the -f option is specified.

(c) area/sarea hierarchical map (with +gn, aname, -a, and -e specified)

```

** allocator map **
site name = site
< area >
garea/aname  raddr  size  bkupfile  date  lddate  svdate
garea/aname  s k raddr size bkupfile YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
< sarea >
garea/aname/sname  raddr  size  bkupfile  date  lddate  svdate
garea/aname/sname  s k raddr size bkupfile YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
garea/aname/ _____
** map output end **

```

Note: The display shows date, lddate, and svdate only when the -f option is specified.

(15) Default display format

(a) Default display format (no detailed information)

If you specify the -u option and omit all other options, the system outputs split area and secondary partition area listings sorted by address; task, program, IRSUB, built-in subroutine, and IRGLB listings sorted by number; and VAL listings sorted by name with no detailed information attached. The display format is as follows:

```

** allocator map **
site name = site
< garea >
gname laddr paddr size
gname laddr paddr size
< area >
garea/area raddr size kind bkupfile
garea/area raddr laddr kind bkupfile
garea/area raddr laddr
< sarea >
garea/area raddr size laddr
garea/area/ raddr size laddr
garea/area/ raddr size laddr
< task-program >
tn tname tnox rmtm lvl sp
tn tname s k max rmtm lvl sp
tn tname st mtm texttop lastaddr tsize dsiz ssize (part ) bsize extra oswork
tn tname pname s k st mtm texttop lastaddr bsize dsiz ssize (part ) bsize extra oswork
tn tname s k st mtm texttop lastaddr bsize dsiz ssize (part ) bsize extra oswork
< IRSUB > [max_entry=entry, use_entry=entry.]
irno entname st laddr subname offset
irno entname s k st laddr subname offset
irno entname s k st laddr subname offset
irno entname s k st laddr subname offset
< ULISUB >
pnt typ ent subname texttop bsslast tsize dsiz bsize extra ssize (part )
pnt typ ent b subname s k texttop bsslast dsiz bsize extra ssize (part )
< global,CM,DCM,irglb > [max_entry=entry, use_entry=entry.]
irno entname laddr salname
irno entname s k laddr salname offset kind s k aname ssize datasize aoff
irno entname s k laddr salname offset kind s k aname ssize datasize aoff
irno entname s k laddr salname offset kind s k aname ssize datasize aoff
*****
< value > [max_entry=entry, use_entry=entry.]
ename
ename k valhex valdec
** map output end **

```

(b) Default display format (detail display)

If you specify the `-u` and `-f` options and omit all other options, the system outputs split area and secondary partition area listings sorted by address; task, program, IRSUB, built-in subroutine, and IRGLB listings sorted by number; and VAL listings sorted by name with detailed information attached. The display format is as follows:

```

** allocator map **
site name = site
< gareal >
gname laddr paddr size
gareal/aname
gareal/____ raddr size
< area >
gareal/aname raddr size laddr kind bkupfile date lddate svdate
gareal/aname s k raddr size laddr YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
gareal/aname/____ raddr size laddr
< sareal >
gareal/aname raddr size laddr raddr size lddate svdate
gareal/aname/____ raddr size laddr YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
< task-program >
tn tname tnox rmtm lv1 sp pname pname st mtn texttop lastaddr tsize dsize ssize (part ) bsize extra oswork ddatop bstop date
lv tname s k lv min texttop lastaddr lastaddr bsize dsize ssize (part ) bsize extra oswork ddatop bstop YYYY/MM/DD hh:mm:ss YYYY/MM/DD
hh:mm:ss
< IRSUB > [max_entry=entry, use_entry=entry]
irno entname laddr subname offset texttop bsslast tsize dsize bsize extra ssize (part ) ddatop bstop lddate
entname s k laddr subname 000000 s k texttop bsslast bsize dsize bsize extra ssize (part ) ddatop bstop
entname s k laddr subname offset s k texttop bsslast bsize dsize bsize extra ssize (part ) ddatop bstop
irno entname s k laddr subname 000000 s k texttop bsslast bsize dsize bsize extra ssize (part ) ddatop bstop
irno entname s k laddr subname offset s k texttop bsslast bsize dsize bsize extra ssize (part ) ddatop bstop
< ULSUB >
pnt typ ent subname texttop bsslast tsize dsize bsize extra ssize (part ) ddatop bstop date lddate
pnt typ ent b subname s k texttop bsslast bsize dsize bsize extra ssize (part ) ddatop bstop YYYY/MM/DD hh:mm:ss
< IRGLB GLB > [max_entry=entry, use_entry=entry]
irno entname laddr salname offset kind s k aname sasive datasize aoff date
irno entname s k laddr salname 000000 kind s k aname sasive datasize aoff YYYY/MM/DD hh:mm:ss
irno entname s k laddr salname 000000 kind s k aname sasive datasize aoff YYYY/MM/DD hh:mm:ss
irno entname s k laddr salname ***** kind s k aname sasive datasize aoff YYYY/MM/DD hh:mm:ss
< value > [max_entry=entry, use_entry=entry]
ename valhex valdec valdec_ YYYY/MM/DD hh:mm:ss
** map output end **

```



## APPENDIX G DISPLAY FORMATS OF md AND sd OF svdebug (ONLINE DEBUGGER)

### (1) Display format of the md subcommand

- Format of display (print) data

```

0xaaaaaaaa dddddddd dddddddd dddddddd dddddddd '.....'
  Address           Displayed data           Character code
    
```

**Address:** First address of the displayed data, in hexadecimal.

**Displayed data:** The content at the address is displayed by using the specified data length in the specified output format.

Up to 16 bytes of data can be displayed on a single line.

When the floating-point data output format is used (`-f -l`, `-fd`) and the following data is at the address, the data is displayed in hexadecimal. The corresponding character string is displayed after the hexadecimal display.

| Floating-point data       | Character string | Display example  |                             |
|---------------------------|------------------|------------------|-----------------------------|
|                           |                  | Single precision | Double precision            |
| Non-numeric               | Na               | 0x7fffffff: Na   | 0xfff00000 0x00000001: Na   |
| Infinite                  | In               | 0x7f800000: In   | 0xfff00000 0x00000000: In   |
| Maximum expressible value | Ma               | 0x7f7fffffff: Ma | 0x7fefffffff 0xffffffff: Ma |
| Minimum expressible value | Mi               | 0xff7fffffff: Mi | 0xffefffffff 0xffffffff: Mi |

**Character code:** For the hexadecimal data output format (`-h`) only, data is converted into character codes and is displayed on the right side of the window. Data that cannot be converted is shown as periods ( `.` ).

When a line has the same data as the previous line, the following message is displayed (when the `-all` option is specified, the display shows all consecutive data):

```

0xaaaaaaaa-0xaaaaaaaa as previous
  |
  |-----> First address of identical data
  |
  |-----> Last address of identical data
    
```

- Format of modification (patch) data

```

0xaaaaaaaa dddddddd :
  Address   Display data
    
```

● Examples

The following are examples displayed by md with various combinations of data output format options and data length options:

```

*****|*****|*****|*****|*****|*****|*****|*****|
Four-byte display in 0x00ec0000 0000000a 00000064 000003e8 00002710 '.....d.....'
hexadecimal (-h, -l) 0x00ec0010 000186a0 '....'

Two-byte display in 0x00ec0000 0000 000a 0000 0064 0000 03e8 0000 2710 '.....d.....'
hexadecimal (-h, -w) 0x00ec0010 0001 86a0 '....'

One-byte display in 0x00ec0000 00 00 00 0a 00 00 00 64 00 00 03 e8 00 00 27 10 '.....d.....'
hexadecimal (-h, -b) 0x00ec0010 00 01 86 a0 '....'

Four-byte display in 0x00ec0000 10 100 1000 10000
decimal (-d, -l) 0x00ec0010 100000

Two-byte display in 0x00ec0000 0 10 0 100 0 1000 0 10000
decimal (-d, -w) 0x00ec0010 1 -31072

One-byte display in 0x00ec0000 0 0 0 10 0 0 0 100 0 0 3 -24 0 0
decimal (-d, -b) 39 16
0x00ec0010 0 1 -122 -96

Single-precision real 0x00ec0020 1.1200000 2.1229999 10.1230001 20.1233997
number display (-f, -l) 0x00ec0030 100.123451

Double-precision real 0x96000000 1.0000000000000000E+00 2.0000000000000000E+00
number display (-fd) 0x96000010 -1.0000000000000000E+00 1.0000000000000000E+100
0x96000020 0x7fefffff 0xffffffff:Ma 0xfefeffff 0xffffffff:Mi
    
```

(2) Display format of sd

● Format of display (print) data

0xaaaaaaaa (0xllllll) ddddddd ddddddd ddddddd ddddddd '.....'  
 Address      Offset      Displayed data      Character code

Address: First address of the displayed data, in hexadecimal.

Offset: Offset relative to the beginning of the data.

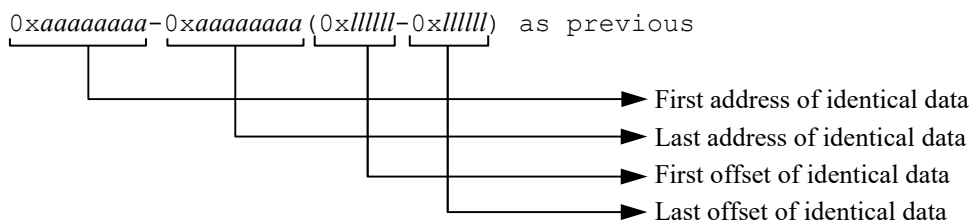
Displayed data: The content at the address is displayed by using the specified data length in the specified output format. Up to 16 bytes of data can be displayed on a single line.

When the floating-point data output format is used (-f -l, -fd) and the following data is at the address, the data is displayed in hexadecimal. The corresponding character string is displayed after the hexadecimal display.

| Floating-point data       | Character string | Display example  |                             |
|---------------------------|------------------|------------------|-----------------------------|
|                           |                  | Single precision | Double precision            |
| Non-numeric               | Na               | 0x7fffffff: Na   | 0xfff00000 0x00000001: Na   |
| Infinite                  | In               | 0x7f800000: In   | 0xfff00000 0x00000000: In   |
| Maximum expressible value | Ma               | 0x7fefffffff: Ma | 0x7fefffffff 0xffffffff: Ma |
| Minimum expressible value | Mi               | 0xff7fffffff: Mi | 0xffefffffff 0xffffffff: Mi |

Character code: For the hexadecimal data output format (-h) only, data is converted into character codes and is displayed on the right side of the window. Data that cannot be converted is shown as periods (.).

When a line has the same data as the previous line, the following message is displayed (when the -all option is specified, the display shows all consecutive data):



● Format of modification (patch) data

0xaaaaaaaa (0xllllll) ddddddd :  
 Address      Offset      Display data

● Examples

The following are examples displayed by sd with various combinations of data output format options and data length options:

```

*****|*****|*****|*****|*****|*****|*****|*****|
Four-byte display in 0x00ec0000 (0x000000) 0000000a 00000064 000003e8 00002710 '.....d.....'
hexadecimal (-h, -l) 0x00ec0010 (0x000010) 000186a0 '....'

Two-byte display in 0x00ec0000 (0x000000) 0000 000a 0000 0064 0000 03e8 0000 2710 '.....d.....'
hexadecimal (-h, -w) 0x00ec0010 (0x000010) 0001 86a0 '....'

One-byte display in 0x00ec0000 (0x000010) 00 00 00 0a 00 00 00 64 00 00 03 e8 00 00 27 10 '.....d.
hexadecimal (-h, -b) .....'.
0x00ec0010 (0x000010) 00 01 86 a0 '....'
,

Four-byte display in 0x00ec0000 (0x000000) 10 100 1000 10000
decimal (-d, -l) 0x00ec0010 (0x000010) 100000

Two-byte display in 0x00ec0000 (0x000000) 0 10 0 100 0 1000 0 10000
decimal (-d, -w) 0x00ec0010 (0x000010) 1 -31072

One-byte display in 0x00ec0000 (0x000000) 0 0 0 10 0 0 0 100 0 0 3 -24
decimal (-d, -b) 0 0 39 16
0x00ec0010 (0x000010) 0 1 -122 -96

Single-precision real 0x00ec0020 (0x000000) 1.1200000 2.1229999 10.1230001 20.1233997
number display (-f, -l) 0x00ec0030 (0x000010) 100.123451

Double-precision real 0x96000000 (0x000000) 1.0000000000000000E+00 2.0000000000000000E+00
number display (-fd) 0x96000010 (0x000010) -1.0000000000000000E+00 1.0000000000000000E+100
0x96000020 (0x000020) 0x7fefefffff 0xffffffffff:Ma 0xffefefffff 0xffffffffff:Mi
    
```

**APPENDIX H LIST OF STACK SIZES FOR LIBRARY USE**

The following are lists of the stack sizes used by the libraries.

## (1) List of stack sizes of the C standard library

| No. | Function name | Stack size      |                 |
|-----|---------------|-----------------|-----------------|
|     |               | libsh4nbmdn.lib | libsh4nbmzz.lib |
| 1   | atof          | 440             | 436             |
| 2   | frexp         | 8               |                 |
| 3   | ldexp         | 20              |                 |
| 4   | memchr        | 0               |                 |
| 5   | memset        | 4               |                 |
| 6   | modf          | 40              |                 |
| 7   | sscanf        | 532             |                 |
| 8   | sprintf       | 804             |                 |
| 9   | strcat        | 0               |                 |
| 10  | strchr        | 0               |                 |
| 11  | strcmp        | 0               |                 |
| 12  | strcpy        | 24              |                 |
| 13  | strcspn       | 0               |                 |
| 14  | strlen        | 0               |                 |
| 15  | strncat       | 4               |                 |
| 16  | strncmp       | 4               |                 |
| 17  | strncpy       | 0               |                 |
| 18  | strpbrk       | 0               |                 |
| 19  | strchr        | 12              |                 |
| 20  | strspn        | 0               |                 |
| 21  | strtod        | 440             | 436             |
| 22  | strtoll       | 68              |                 |
| 23  | vsprintf      | 804             |                 |
| 24  | acos          | 96              |                 |
| 25  | asin          | 80              |                 |
| 26  | atan          | 60              |                 |
| 27  | atan2         | 124             |                 |
| 28  | ceil          | 28              |                 |
| 29  | exp           | 48              |                 |
| 30  | fabs          | 0               |                 |
| 31  | floor         | 28              |                 |
| 32  | fmod          | 36              |                 |
| 33  | log           | 48              |                 |
| 34  | log10         | 48              |                 |
| 35  | pow           | 96              |                 |
| 36  | cos           | 60              |                 |
| 37  | sin           | 60              |                 |
| 38  | cosh          | 68              |                 |
| 39  | sinh          | 60              |                 |
| 40  | sqrt          | 8               |                 |
| 41  | tan           | 32              |                 |
| 42  | tanh          | 60              |                 |

## (2) List of stack sizes of libfirad.lib

| No. | Function name | Stack size |
|-----|---------------|------------|
| 1   | irglbad       | 0          |
| 2   | irsubad       | 0          |

## (3) List of stack sizes of libers.lib

| No. | Function name | Stack size |
|-----|---------------|------------|
| 1   | fpgetmask     | 0          |
| 2   | fpgetround    | 0          |
| 3   | fpgetsticky   | 0          |
| 4   | fpsetmask     | 0          |
| 5   | fpsetround    | 0          |
| 6   | fpsetsticky   | 0          |
| 7   | fpcheck       | 0          |
| 8   | fpchecko      | 0          |

## (4) List of stack sizes of libcpms.lib

| No. | Function name | Stack size |
|-----|---------------|------------|
| 1   | memcpy (*)    | 28         |

(\*) In a program or subprogram loaded by the loader, the memcpy () function of the CPMS library is used instead of the memcpy () function of the C standard library.

**This Page Intentionally Left Blank**