

HITACHI

ソフトウェアマニュアル

オペレーション

RPDP For Windows[®]

S10VE

SEJ-3-133(A)

ソフトウェアマニュアル

オペレーション

RPDP For Windows®

SIOVE

この製品を輸出される場合には、『外国為替及び外国貿易法』の規制ならびに『米国輸出管理規則』など外国の輸出関連法規をご確認のうえ、必要な手続きをお取りください。
なお、ご不明な点がございましたら、当社担当営業にお問い合わせください。

2018年 8月 (第1版) SEJ-3-133 (A)

- このマニュアルの一部または全部を無断で転写したり複製したりすることは、固くお断りいたします。
- このマニュアルの内容を、改良のため予告なしに変更することがあります。

安全上のご注意

- システムの構築やプログラムの作成などは、このマニュアルの記載内容をよく読み、書かれている指示や注意を十分理解してから行ってください。誤操作により、システムが故障することがあります。
- このマニュアルは、必要なときすぐに参照できるよう、手近なところに保管してください。
- このマニュアルの記載内容について疑問点または不明点がございましたら、最寄りの弊社営業またはSEまでお知らせください。
- お客様の誤操作に起因する事故発生や損害については、弊社は責任を負いかねますのでご了承ください。
- 弊社提供ソフトウェアを改変して使用した場合に発生した事故や損害については、弊社は責任を負いかねますのでご了承ください。
- 弊社提供以外のソフトウェアを使用した場合の信頼性については、弊社は責任を負いかねますのでご了承ください。
- ファイルのバックアップ作業を日常業務に組み入れてください。ファイル装置の障害、ファイルアクセス中の停電、誤操作、その他何らかの原因によりファイルの内容を消失することがあります。このような事態に備え、計画的にファイルのバックアップを取っておいてください。
- 弊社製品が故障や誤動作したりプログラムに欠陥があった場合でも、使用されるシステムの安全が十分に確保されるよう、保護・安全回路は外部に設け、人身事故や重大な災害に対する安全対策が十分確保できるようなシステム設計としてください。
- 非常停止回路、インターロック回路などはPLCの外部で構成してください。PLCの故障により、機械の破損や事故の恐れがあります。
- 運転中のプログラム変更、強制出力、RUN、STOPなどは十分安全を確認してから行ってください。誤操作により、機械の破損や事故の恐れがあります。
- このマニュアルでは、安全上の注意事項のランクを潜在危険の重大度によって、「危険」、「警告」、「注意」、「通知」と区分しています。

警告表示の定義



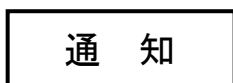
: この表示を無視して誤った取り扱いをすると、死亡または重大な傷害を引き起こす危険の存在を示す。



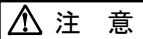

: この表示を無視して誤った取り扱いをすると、死亡または重大な傷害を引き起こすおそれのある危険の存在を示す。



: この表示を無視して誤った取り扱いをすると、軽度の傷害または中程度の傷害を引き起こすおそれのある危険の存在を示す。



: この表示を無視して誤った取り扱いをすると、人身傷害とは関係のない損害を引き起こすおそれのある危険の存在を示す。

なお、 **注意**、 **通知** に記載した事項でも、状況によっては重大な結果に結びつく可能性があります。どれも重要な内容を記載していますので必ず守ってください。

「重大な傷害」、「軽度の傷害または中程度の傷害」、「人身傷害とは関係のない損害」について、具体的な内容を以下に示します。

重大な傷害

失明、けが、やけど（高温、低温）、感電傷害、骨折、中毒などで、後遺症が残るものおよび治療のために入院、長期の通院を要するもの

軽度の傷害または中程度の傷害

治療のために入院や長期の通院を必要としないけが、やけど、感電傷害など

人身傷害とは関係のない損害

周囲の財物の損傷、弊社製品の故障や破損、データの損失など、人身傷害以外の損害

安全上の注意事項は、安全性を確保するための原則に基づいた、弊社製品における各種対策を補完する重要なものです。弊社製品やマニュアルに表示されている安全上の注意事項は、十分に検討されたものですが、それでも、予測を超えた事態が起こることが考えられます。操作するときは指示に従うだけでなく、常に自分自身でも注意するようにしてください。また、弊社製品の安全な運転および保守のために、各種規格、基準に従って安全施策を確立してください。

来歴一覧表

改訂No.	来歴（改訂内容）	発行年月	備考
A	新規作成	2018.8	

このページは白紙です。

はじめに

このマニュアルは、S10VEのCPMS下で動作するリアルタイムプログラムを作成する方法について述べたものです。

- 下表に関連マニュアル（ソフトウェアマニュアル）を示します。

マニュアル番号	マニュアル名称
SEJ-3-201	S10VE ソフトウェアマニュアル CPMS概説&マクロ仕様
SEJ-1-001	S10VE ユーザーズマニュアル 総合編

- 次の用語は、このマニュアルにおいて特殊な意味に用いますので注意してください。

略称	正式名称
RPDP	Realtime Program Developing Package for S10VE
CPMS	Compact Process Monitor System
PCs	Programmable Controllers
PLC	Programmable Logic Controller

- このマニュアルは「第1編 概説」と「第2編 コマンドリファレンス」の2つの編と付録から構成されます。

第1編 概説

S10VEで動作するリアルタイムプログラムの開発手順と開発に使用するコマンドの概要について説明しています。

第2編 コマンドリファレンス

S10VEで動作するリアルタイムプログラムの開発に使用するコマンドのリファレンスです。コマンドごとの機能やオプション機能について説明しています。

付録

S10VEで動作するリアルタイムプログラムを開発するうえでの注意事項やエラーメッセージ、さらにコマンド実行結果の表示フォーマットを示します。

<商標について>

- Microsoft®, Windows®は、米国Microsoft Corporationの米国およびその他の国における登録商標または商標です。

<記憶容量の計算値についての注意>

- 2ⁿ計算値の場合（メモリ容量・所要量、ファイル容量・所要量など）
 - 1KB（キロバイト）＝1,024バイトの計算値です。
 - 1MB（メガバイト）＝1,048,576バイトの計算値です。
 - 1GB（ギガバイト）＝1,073,741,824バイトの計算値です。
 - 1TB（テラバイト）＝1,099,511,627,776バイトの計算値です。
- 10ⁿ計算値の場合（ディスク容量など）
 - 1KB（キロバイト）＝1,000バイトの計算値です。
 - 1MB（メガバイト）＝1,000²バイトの計算値です。
 - 1GB（ギガバイト）＝1,000³バイトの計算値です。
 - 1TB（テラバイト）＝1,000⁴バイトの計算値です。

目次

第1編 概説

第1章 概要	1-2
1. 1 RPDПについて	1-2
1. 2 コマンド一覧	1-4
1. 3 プロセッサー (CP、HP) の使い分け	1-7
1. 3. 1 構成と役割	1-7
1. 3. 2 プログラミング環境	1-9
1. 3. 3 RPDП機能とCP、HPサイト指定	1-10
第2章 プログラム開発手順	1-12
2. 1 全体フロー	1-12
2. 2 サイト環境	1-15
2. 2. 1 サイト指定でのSIOVEとの接続	1-16
2. 3 主メモリのエリア管理と領域分割	1-17
2. 4 タスクのエリア配置	1-22
2. 5 IRSUBのエリア配置	1-23
2. 6 プログラムのロードとタスクの生成	1-24
2. 7 間接リンク常駐サブプログラム	1-24
2. 8 グローバル (GLB)	1-24
2. 9 PU間共有メモリ (CM)	1-24
2. 10 バリュ (VAL)	1-25
2. 11 間接リンクグローバル	1-25
2. 12 GLB、VAL、IRSUBプログラミングガイド	1-25
2. 13 CPMS上のプログラム作成の制約条件	1-34
第3章 インストールと実行環境	1-40
3. 1 インストール	1-40
3. 2 前提P.P.	1-40
3. 3 インストール時の注意事項	1-40
3. 3. 1 RPDПインストール時の注意事項	1-40
3. 3. 2 SHCコンパイラインストール時の注意事項	1-40
3. 4 RPDП実行環境	1-41
3. 5 RPDП使用者アカウントの登録	1-42
3. 5. 1 新規アカウントの登録	1-42
3. 5. 2 既存のアカウントの所属するグループにRPDPusersを追加	1-43

第4章 コンパイラ	1-44
4. 1 Cコンパイラオプション詳細	1-44
4. 2 コンパイル時の注意点	1-46
4. 2. 1 shcを使用してコンパイルする場合	1-46
4. 3 shcのバージョン比較	1-48
4. 3. 1 コマンド行オプション	1-48
4. 4 データジェネレータ	1-49
第5章 プログラミングコマンド	1-58
5. 1 プログラミングコマンドについて	1-58
第6章 アロケータ	1-59
6. 1 分割領域の確保と削除	1-59
6. 1. 1 領域の分割の必要性	1-59
6. 1. 2 分割領域の確保	1-60
6. 1. 3 分割領域の削除	1-64
6. 1. 4 GLB、VALの名前の付け方	1-64
6. 1. 5 CM用の分割領域の確保	1-65
6. 2 バリュ (VAL) の登録、削除	1-67
第7章 ローダ	1-68
7. 1 リンク・ロードとは	1-68
7. 2 ローダの動作環境	1-69
7. 3 ライブラリのサーチパス	1-72
7. 4 リンク・ロード時の注意事項	1-72
第8章 ビルダ	1-73
8. 1 タスクの登録と削除	1-73
8. 1. 1 タスクとは	1-73
8. 1. 2 タスクの登録	1-73
8. 1. 3 タスクの削除	1-74
8. 2 常駐サブプログラムの登録と削除	1-75
8. 2. 1 間接リンクサブプログラム (IRSUB) とは	1-75
8. 2. 2 間接リンクサブプログラム (IRSUB) の登録	1-75
8. 2. 3 間接リンクサブプログラム (IRSUB) の削除	1-76
8. 3 組み込みサブルーチンの登録と削除	1-77
8. 3. 1 組み込みサブルーチンとは	1-77

8. 3. 2	組み込みサブルーチンの登録	1-78
8. 3. 3	組み込みサブルーチンの削除	1-78
第9章 マップ 1-79		
9. 1	アロケータ管理テーブル情報表示の目的	1-79
9. 2	svmapコマンドのオプション指定と表示情報	1-80
9. 2. 1	マップ情報の出力対象	1-80
9. 2. 2	マップ情報の出力内容	1-80
9. 2. 3	マップ情報の出力形式	1-80
9. 3	svadmコマンドの論理アドレス指定と表示情報	1-82
第10章 立ち上げ/PU制御 1-83		
10. 1	概要	1-83
10. 2	立ち上げ/PU制御の基本的な考え方	1-84
10. 3	立ち上げ/PU制御手順	1-85
10. 4	立ち上げ/停止種別	1-86
10. 5	PUの状態遷移	1-89
10. 5. 1	立ち上げ操作	1-90
10. 5. 2	PU制御操作	1-92
第11章 svdebug (オンラインデバッガ) とデバッグ支援コマンド 1-94		
11. 1	概要	1-94
11. 2	S10VE状態とサブコマンド実行可否	1-95
11. 3	基本的な機能	1-97
11. 4	その他の機能	1-101
11. 5	デバッグ支援コマンド	1-102
11. 5. 1	svelogコマンド	1-102
11. 5. 2	svdhpコマンド	1-103
11. 5. 3	svcpunowコマンド	1-104
11. 5. 4	svtimexコマンド	1-105
第2編 コマンドリファレンス		
第1章 コンパイラ 2-2		
svdatagen		2-2
第2章 プログラミングコマンド 2-4		
makehce		2-4

第3章 アロケータ	2-16
svdfa	2-16
svdla	2-18
svdfs	2-19
svdls	2-22
svdfv	2-23
svdlv	2-24
第4章 ローダ	2-25
svload	2-25
svdload	2-39
svcomp	2-40
第5章 ビルダ	2-45
svctask	2-45
svdtask	2-47
svbuild	2-48
svdbuild	2-51
svirglb	2-54
第6章 管理ツール	2-56
svmap	2-56
svadm	2-59
svsitectl	2-62
第7章 立ち上げ/PU制御	2-63
svrpl	2-63
svcpuctl	2-66
第8章 svdebug (オンラインデバッガ) とデバッグ支援コマンド	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

付録

付録A プログラムで使用できる名称	A-2
付録B ライブラリ	A-6
付録C サイト管理ファイル	A-9
付録D エラーメッセージ	A-15
付録E RPDP使用上の注意事項	A-47
付録F マップの表示フォーマット	A-49
付録G svdebug (オンラインデバッガ) md、sdの表示フォーマット	A-64
付録H ライブラリの使用するスタックサイズ一覧	A-68

図目次

図 1-1	ツールの適用システム構成	1-2
図 1-2	S10VEのハードウェア構成	1-7
図 1-3	HP、CPサイト環境とハードウェア	1-8
図 1-4	プログラム開発手順全体フロー（サイト構築からプログラム開発までの流れ）	1-13
図 1-5	S10VEのサイトディレクトリ構成	1-15
図 1-6	CPMSが管理する論理空間	1-17
図 1-7	S10VEの物理メモリマップ	1-20
図 1-8	論理空間内のタスク配置	1-22
図 1-9	論理空間内のタスク配置（マルチタスク）	1-22
図 1-10	論理空間内のIRSUB配置	1-23
図 1-11	論理空間内のIRSUB配置（マルチエントリ）	1-23
図 1-12	書き込みの可否	1-35
図 1-13	データサイズ比較	1-38
図 1-14	データ配置順を考慮した構造体宣言例	1-39
図 1-15	空きエリアを明示的に宣言した例	1-39
図 1-16	構造体サイズを考慮した宣言例	1-39
図 1-17	defines.h	1-53
図 1-18	分割領域のレイアウト	1-61
図 1-19	CPMSの論理空間上のCM空間とS10VE主メモリのCM空間との対応	1-65
図 1-20	ロードモジュール、バックアップファイルの作成	1-68
図 1-21	ロードモジュールの構成	1-69
図 1-22	ローディング処理	1-70
図 1-23	開発系マシンからのS10VE立ち上げ概略	1-83
図 1-24	S10VE全体の制御の考え方	1-84
図 1-25	PU（OS立ち上げ・停止）の状態遷移	1-89
図 2-1	関数呼び出しの関係とスタック使用量	2-31
図 2-2	svcomp（プログラム、サブプログラム）のフォーマット	2-43
図 2-3	svcomp（GLB、CM）のフォーマット	2-44
図 2-4	メモリアクセス範囲	2-86
図 2-5	ダイナミック表示のオペレーション	2-86
図 2-6	メモリアクセス範囲	2-95

表目次

表 1-1	RPDPの提供コマンド一覧	1-4
表 1-2	HP、CPのアクセス対象可否	1-8
表 1-3	プログラミング資源とCPサイトでの利用可否	1-9
表 1-4	RPDP機能と処理対象サイト	1-10
表 1-5	RPDPコマンドと処理対象サイト	1-11
表 1-6	各論理空間の用途	1-18
表 1-7	各論理空間のアドレスとサイズ	1-19
表 1-8	GLB、VALの名称の付け方	1-25
表 1-9	GLBおよびVALの使い方	1-26
表 1-10	IRSUBの使い方	1-32
表 1-11	RPDPの前提P.P.	1-40
表 1-12	S10VE RPDP実行環境の設定値一覧	1-41
表 1-13	shcコンパイラの動作に必要な環境変数	1-45
表 1-14	浮動小数点数の扱い制御オプション	1-46
表 1-15	浮動小数点数の扱いと対応する標準ライブラリ	1-46
表 1-16	shcのコマンド行オプションのバージョン比較	1-48
表 1-17	初期値型変換仕様	1-57
表 1-18	分割領域の用途と配置するGAREAの対応	1-60
表 1-19	ロードモジュールの条件	1-69
表 1-20	外部参照の組み合わせ	1-71
表 1-21	出力内容と指定可能出力形式の組み合わせ	1-81
表 1-22	立ち上げ/停止種別	1-86
表 1-23	ダウンロードオプション	1-90
表 2-1	バイナリデータ配置	2-3
表 2-2	svdfaのオプションの組み合わせ	2-17
表 2-3	svtypeに指定する値とアライン数の関係	2-20
表 2-4	svdfsのオプションの組み合わせ	2-20
表 2-5	スタックサイズの計算例	2-31
表 2-6	出力リソース指定と出力順指定の組み合わせ可否とデフォルトの出力順	2-58
表 2-7	svdebug機能一覧	2-70
表 2-8	タスクの状態	2-75
表 2-9	ステータスビットの構成	2-75
表 2-10	id、t、cyctの説明	2-80
表 2-11	指定できる値とオプションの組み合わせ	2-85
表 2-12	オプションの組み合わせによる表示フォーマット	2-85
表 2-13	リソースの管理状態	2-119

表A-1	ライブラリの指定条件	A-6
表A-2	エラーメッセージ	A-16
表A-3	リアルタイムリソースの管理状態	A-51

このページは白紙です。

第1編 概説

第1章 概要

1.1 RPDPIについて

リアルタイムプログラム開発パッケージ（RPDP/S10VE）とは、S10VEのリアルタイムOS（CPMS）上で動作するプログラムを開発するツールです。このツールは、Windows® 7/10（x64）を搭載している開発系マシン上で動作します。以下にこのツールを使用するシステム構成を示します。

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

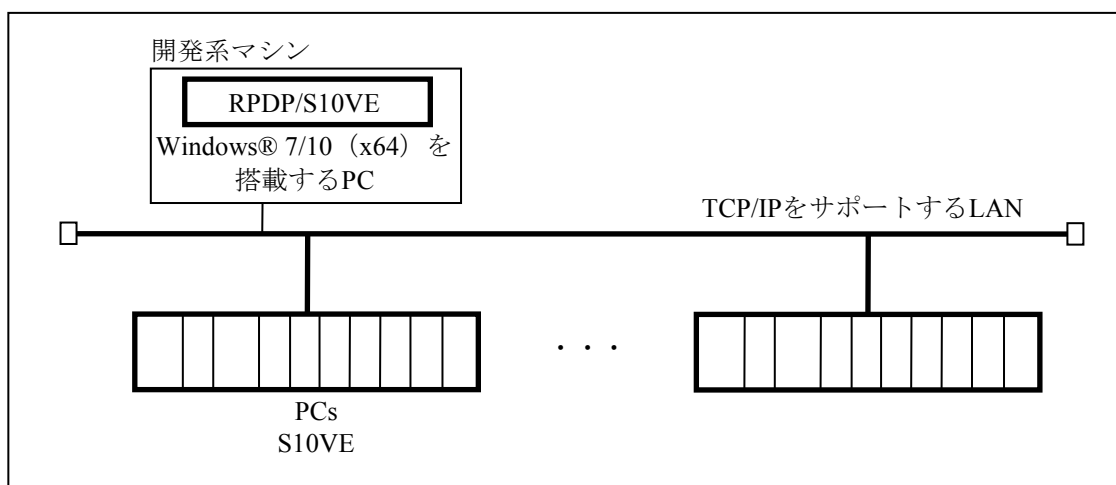
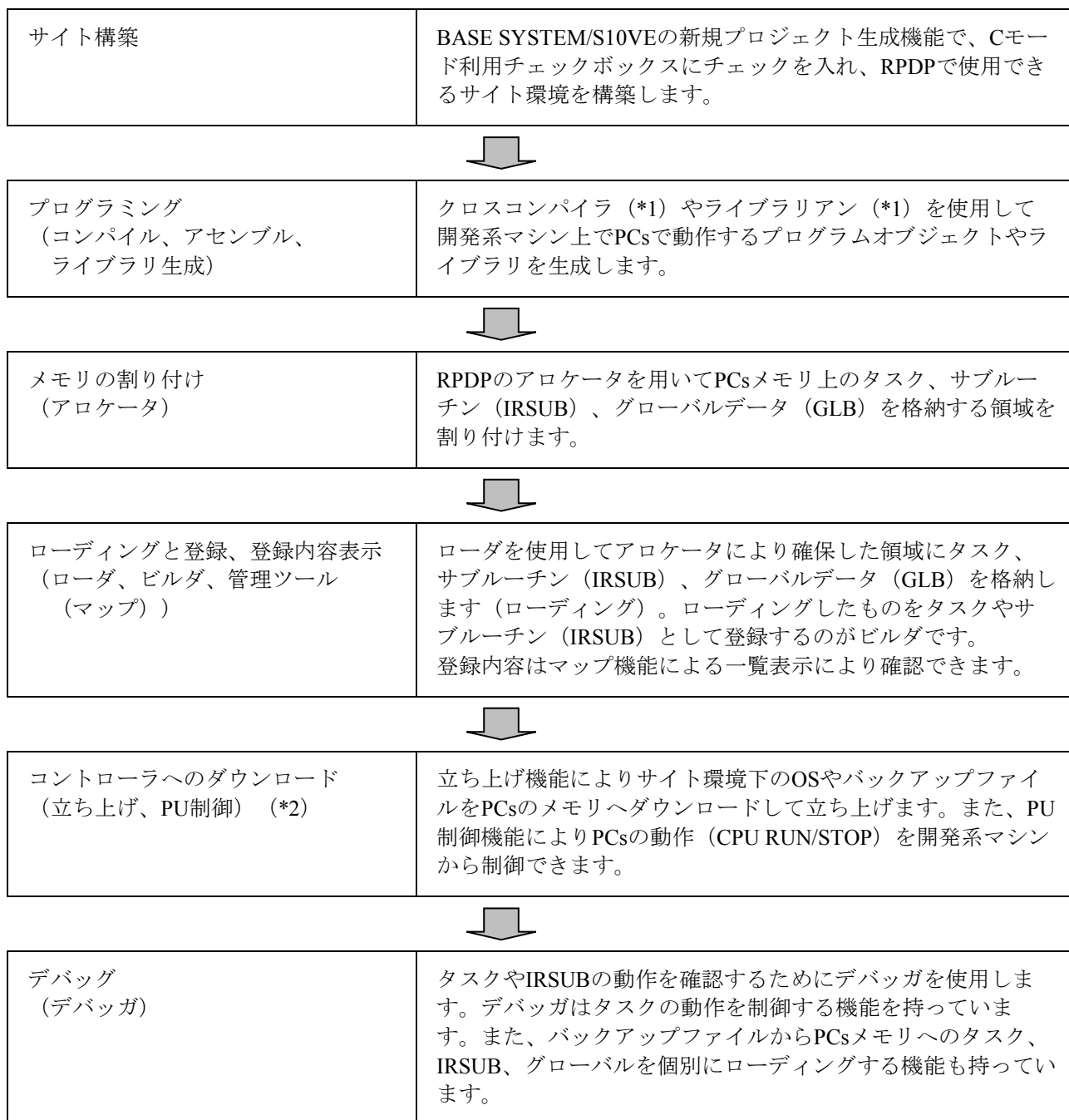


図1-1 ツールの適用システム構成

CPMS上で動作するリアルタイムプログラムの開発は、専用の開発システムRPDP/S10VEを用いて行います。このマニュアルではRPDP/S10VEをRPDPと呼びます。RPDPを用いてプログラムを開発することにより、CPMS上で動作するプログラムは、リアルタイム処理を高速で行えるようにするための属性や機能を使用できます。

以下にPCsのRPDPのプログラム開発手順とサポート機能について示します。



(*1) クロスコンパイラ、アセンブラ、ライブラリアンは、「ルネサスマイクロコンピュータ開発環境システムSuperH RISC engine C/C++コンパイラパッケージVer.9.04 Release 00」を使用します。

(*2) ダウンロード対象となるPCsは、S10VEです。

第1章 概要

1.2 コマンド一覧

RPDPが提供するコマンド一覧を表1-1に示します。

表1-1 RPDPの提供コマンド一覧 (1/3)

分類	コマンド	機能概要	参照ページ
コンパイラ	svdatagen	ロード可能な初期値データのバイナリファイルを生成	2-2
プログラミング コマンド	optlnk	ライブラリアン (コンパイラパッケージの一部)	(*)
	optlnk	リンカ (コンパイラパッケージの一部)	(*)
	makehce	makeコマンド	2-4
アロケータ	svdfa	分割領域の確保、バックアップファイルの生成	2-16
	svdla	分割領域の解放、バックアップファイルの削除	2-18
	svdfs	細分割領域の確保	2-19
	svdls	細分割領域の解放	2-22
	svdfv	VALの確保	2-23
	svdlv	VALの解放	2-24
ローダ	svload	リソースのバックアップファイルへの格納と管理情報への登録	2-25
	svdload	リソースの管理情報からの削除	2-39
	svcomp	格納済みリソースとの比較	2-40
ビルダ	svctask	タスクの生成	2-45
	svdtask	タスクの削除	2-47
	svbuild	間接リンクサブプログラムの登録	2-48
		組み込みサブルーチンの登録	2-49
	svdbuild	間接リンクサブプログラムの削除	2-51
		組み込みサブルーチンの削除	2-52
	svirglb	IRGLBの登録/削除	2-54

(*) 「ルネサスマイクロコンピュータ開発環境システムSuperH RISC engine C/C++コンパイラパッケージ Ver.9.04 Release 00」のマニュアルを参照してください。

表1-1 RPDPの提供コマンド一覧 (2/3)

分類	コマンド	機能概要		参照ページ	
オンライン デバッガ	svdebug	タスク起動/ 停止	qu	タスクの起動要求	2-71
			ab	タスクの起動禁止	2-72
			re	タスクの起動禁止解除	2-73
			ta	タスクの状態表示	2-74
			su	タスクの実行抑止	2-77
			rs	タスクの実行抑止解除	2-78
			tm	タスクの周期起動	2-79
			ct	タスクの周期起動解除	2-81
			sht	タスクの周期起動表示	2-82
			si	スタック初期化	2-129
			sp	スタック使用量の表示	2-130
		メモリプリント/ パッチ	md	アドレス指定によるメモリ内容の表示/ 変更	2-83
			sd	名称指定によるメモリ内容の表示/ 変更	2-87
			bs	指定ビットへのデータ設定	2-90
			bg	指定ビットのデータ表示	2-92
			mcp	メモリ内容のコピー	2-94
			mmv	メモリ内容の移動	2-96
		ブレークポイント	mf	メモリへのパターン値設定	2-98
			br	ブレークポイントの設定/表示	2-103
			stickybr	リセットスタートで解除されないブレーク ポイントの設定表示	2-103
			rb	ブレークポイントの解除	2-109
			rd	レジスタの表示	2-110
			rr	レジスタの内容変更	2-113
		システム エラー表示	go	ブレークポイントからの実行再開	2-114
			el	エラーログの表示	2-100
		現在時刻設定 /表示	ss	システムの状態表示	2-100
			st	現在時刻の設定	2-101
		アップ/ダウン ローディング、コンペア	gt	現在時刻の表示	2-102
			ld	リソースの個別ダウンロード	2-115
			sv	リソースの個別バックアップ	2-123
		DHP記録許可 /禁止	cm	バックアップファイルとS10VEメモリの 内容比較	2-125
			dr	DHP記録許可	2-127
		ラダーのデ バッグ機能	ds	DHP記録禁止	2-127
			lbr	ブレークポイントの設定/表示	2-134
			lrb	ブレークポイントの解除	2-135
			lrd	レジスタの表示	2-136
			lrr	レジスタの書き換え	2-137
			lgo	ブレークポイントからの実行再開	2-138
			s	ステップ実行	2-138

表 1-1 RPDの提供コマンド一覧 (3/3)

分類	コマンド	機能概要		参照ページ	
オンライン デバッガ	svdebug	その他	svdhp	DHPの表示	2-128
			svadm	アドレスに対するリソース名称の表示	2-128
			ps	デバッグ文の表示開始	2-132
			pe	デバッグ文の表示終了	2-132
			ver	CPMSのバージョン表示	2-133
			help	サブコマンド一覧表示	2-139
			q	デバッガの終了	2-141
			!	svdebug実行時の開発系マシン上のコマンド実行	2-141
管理ツール	svmap	マップ情報表示		2-56	
	svadm	アドレスに対する名称の表示		2-59	
	svsitectl	サイト状態の制御と状態の取得		2-62	
立ち上げ/PU制御	svrpl	リモートローディング		2-63	
	svcpuctl	リモート状態制御		2-66	
稼働管理	svcpunow	CPU負荷率の表示		2-147	
	svtimex	タスク稼働率表示		2-148	
エラーログ、DHP 表示	svelog	エラーログ情報出力		2-142	
	svdhp	DHPトレース情報の表示		2-128	

1. 3 プロセッサ（CP、HP）の使い分け

S10VEでは、CPUにSH4Aデュアルプロセッサを使用しています。デュアルプロセッサのコア0をCP（Communication Processor）、コア1をHP（High-speed Processor）として使用します。これらのプロセッサの役割とプログラミング方法について説明します。

1. 3. 1 構成と役割

S10VEではCPUに、SH4Aデュアルプロセッサを使用しています。デュアルプロセッサのコア0を、制御・通信用途のCP（Communication Processor）、コア1を制御用途のHP（High-speed Processor）として使用します。2つのコアを制御・通信処理と制御処理とで使い分け、コア0（CP）に通信制御を行わせることで負荷を分散させ、コア1（HP）のPI/Oアクセスなどの制御プログラムのパフォーマンスを向上させることを目的としています。

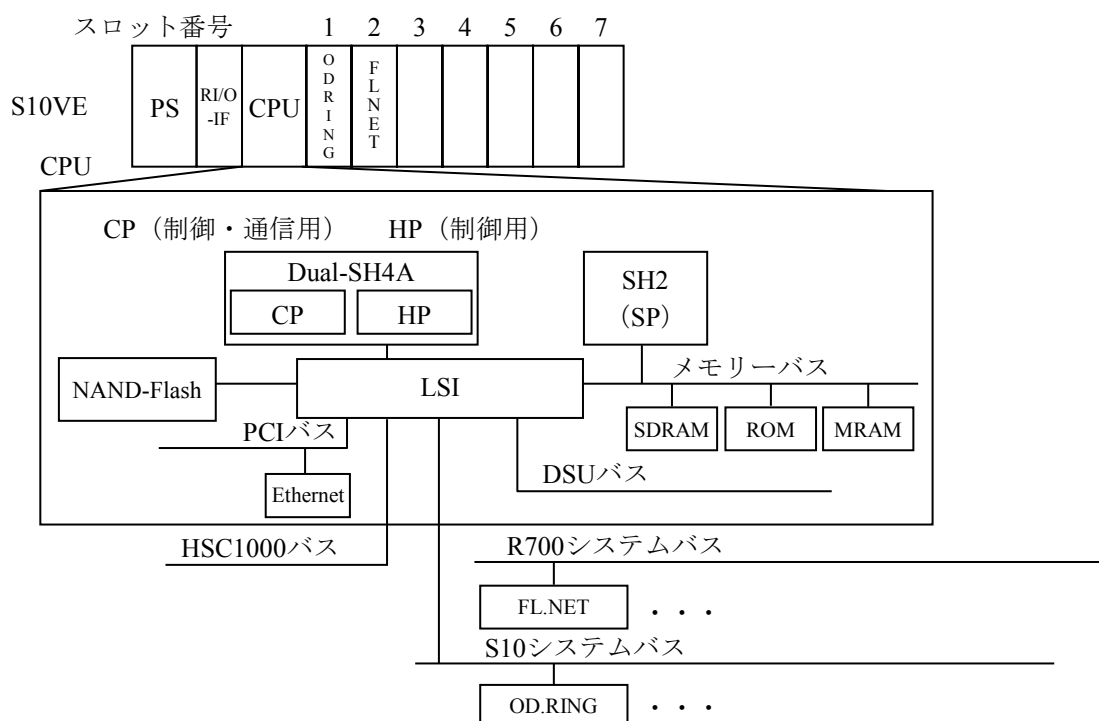


図 1-2 S10VEのハードウェア構成

CPでは、通信処理用のプログラムと制御用プログラムが動作します。通信処理用プログラムとは、HPからの依頼を受けて実際にネットワークを使用して通信を行うプログラムやRCTLNETなどのサブシステムが提供している通信処理用のシステムタスクおよびツール接続用のサーバー、ラダープログラムのイーサネット通信命令を実行するシステムタスクなどがこれに該当します。

制御用プログラムとは、メモリーインターフェイスを介して周期的にPI/Oをアクセスし、演算などを実行して制御を行うプログラムです。

HPでは、主に制御用プログラムが動作します。制御用プログラムとは、ラダープログラムやHI-FLOWプログラムを使用して周期的にPI/Oをアクセスし、演算などを実行して制御を行うプログラムです。HP側のプログラムでデータの送受信を行うこともできます。データの送受信を行う場合には、ラダープログラムのイーサネット通信命令を使用します。

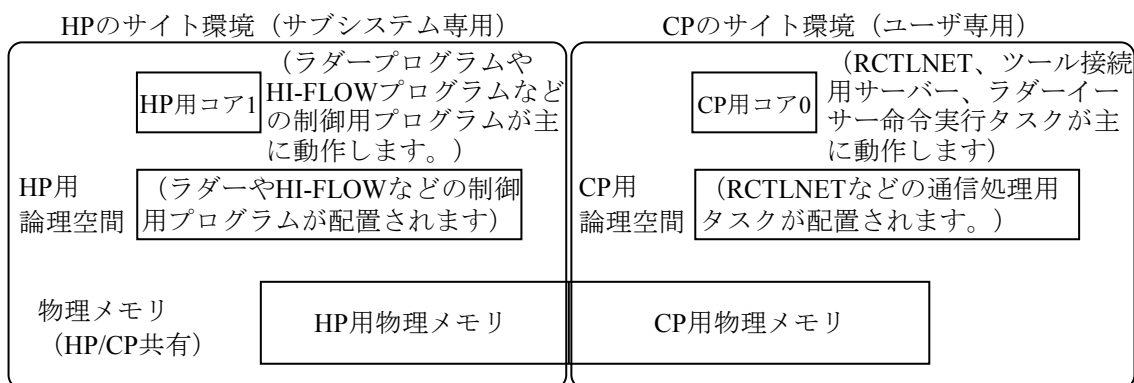


図1-3 HP、CPサイト環境とハードウェア

以下にCP、HPのアクセス対象を示します。

CP上で通信用タスク、HP上で高速シーケンス用タスクが動作します。HP上のタスクからは、入出力インターフェースを使用して内蔵Ethernetでの通信を行うことはできません。同様に、CP上のタスクからは、ラダープログラムを起動できません。このようにタスクが動作するコアによって使用できるネットワークやI/Oは異なります。

CP、HPそれぞれのコアの用途により、CP、HPのアクセス対象を下表のように分類します。

表1-2 HP、CPのアクセス対象可否

アクセス対象	CP	HP	備考
NAND-Flash	○	×	
内蔵Ether (socket)	○	×	socketはエラーリターン
HSC-1000	○	○	
HSC-2100	○	○	
S10モジュール (S10バス空間)	○	○	

○ : アクセス可能

× : アクセス不可

1. 3. 2 プログラミング環境

S10VEのRPDPは、PCsのコア単位（CP、HP）にサイト名称を割り当てて、コア単位にプログラミング資源（タスク、サブプログラム、グローバルなど）を管理します。

BASE SYSTEM/S10VEの新規プロジェクト作成時に「Cモード利用」チェックボックスをチェックすると、PCs番号とコア単位にユニークな名称を付けて、CP、HPサイトが作成されます。

RPDPでは、CP、HPサイトごとにプログラミング資源を用意しています。CPサイトはRCTLNETなどのサブシステムが提供している通信処理用のシステムタスクおよびツール接続用のサーバー、ラダープログラムのイーサネット通信命令を実行するシステムタスクなどを動作させるために使用します。また、HPサイトは、ラダープログラムやHI-FLOWプログラムを使用して周期的にPI/Oをアクセスし、演算などを実行して制御を行うプログラムを動作させるために使用しますので、ユーザータスクは登録しないでください。

以下に、プログラミング資源とCPサイトでの利用可否を示します。

表1-3 プログラミング資源とCPサイトでの利用可否

プログラミング資源	CPサイト (通信用タスク用)
タスク	○ (ユーザタスク : TN=1~224)
IRSUB	○ (8191個ただし256個はOSリザーブ)
組み込みサブルーチン	○ (point(14)*entry(4)ただしentry1はOSリザーブ)
GLB (CM内の定義含む)	○ (8192個ただし256個はOSリザーブ)
VAL	○ (4096個ただし10個はOSリザーブ)

(*) CMの初期値ありGLBに初期値を登録（ローディング）する場合は、CPサイトで登録してください。CMは、CPサイト、HPサイトで共有する空間（メモリ）のため、初期値を登録できるサイトをCPサイトに限定しています。

第1章 概要

1. 3. 3 RPDP機能とCP、HPサイト指定

どのサイトがRPDPのどの機能によって処理されるかについて、表1-4に示します。

表1-4 RPDP機能と処理対象サイト

RPDP機能	処理対象サイト
コンパイラ	CP、HPサイトを指定する必要はありません。
プログラミングコマンド	CP、HPサイトを指定する必要はありません。
アロケータ、ローダ、ビルダ、 管理ツール (svmapなど)	CP、HPサイトを指定して、サイト個別に（プログラム登録、エリアの確保などの提供機能を）実行します。
オンラインデバッガ	CP、HPを指定して、サイト個別にデバッグ機能を実行します。
立ち上げ、PU制御	CP、HPサイト個別ではなく、CP、HPを同時に実行します。ただし、操作上は、対象としてCPのサイトを指定します。
稼働管理	CP、HPを指定して、サイト個別に実行します。
保守コマンド	CP、HPを指定して、サイト個別に実行します。

RPDPコマンドごとの処理対象サイトを表1-5に示します。

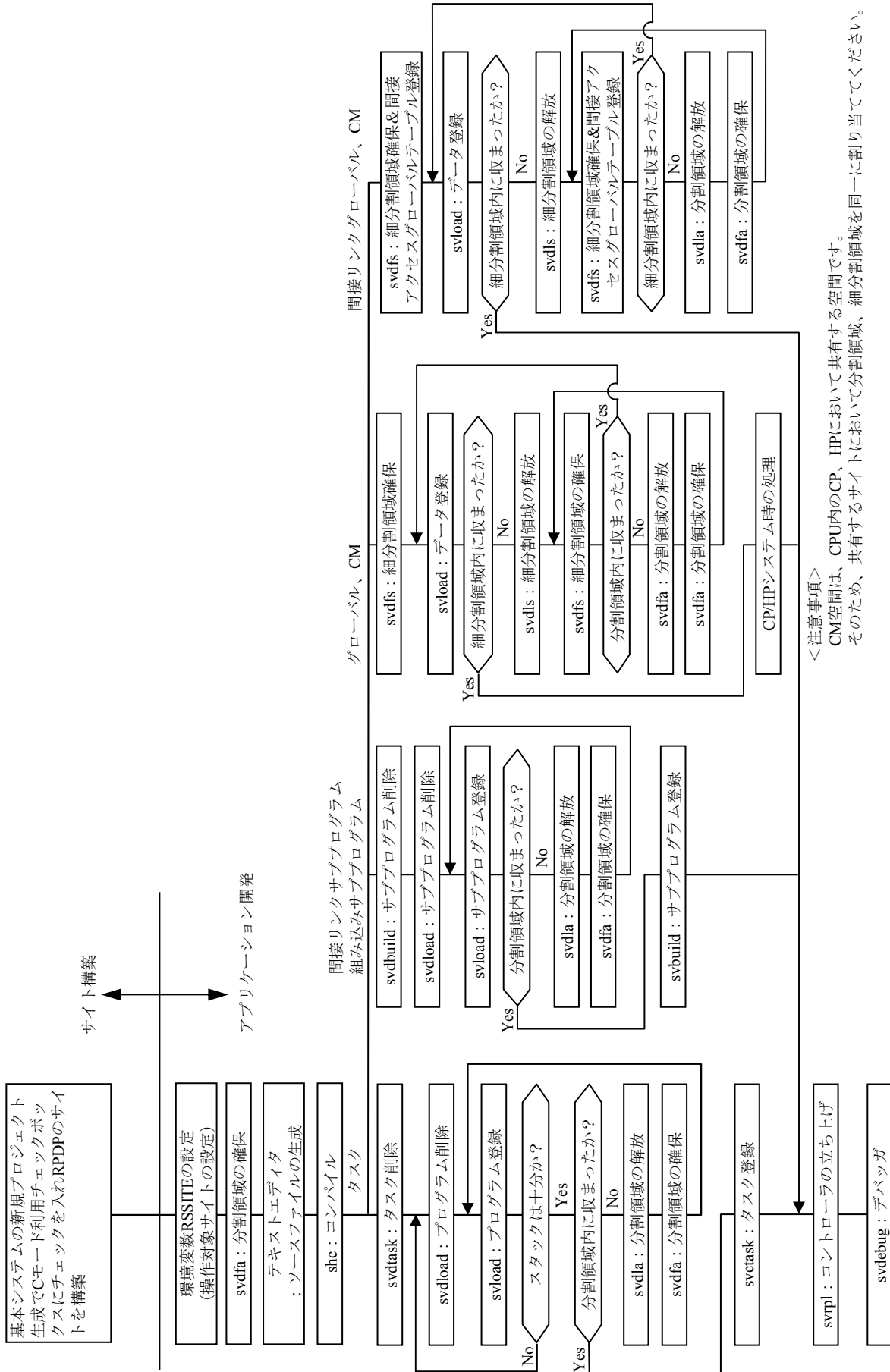
表1-5 RPDPコマンドと処理対象サイト

RPDP機能	コマンド名称	処理対象サイト	
		CP、HPサイトを個別に処理	CP、HPサイトを同時に処理
コンパイラ	svdatagen	○	
プログラミングコマンド	optlnk	サイト指定なし	サイト指定なし
	makehce	サイト指定なし	サイト指定なし
<ul style="list-style-type: none"> ・アロケータ ・ローダ ・ビルダ 	svdfa	○	
	svdla	○	
	svdfs	○	
	svdls	○	
	svdfv	○	
	svdlv	○	
	svload	○	
	svdload	○	
	svcomp	○	
	svctask	○	
	svdtask	○	
	svbuild	○	
	svdbuild	○	
	svirglb	○	
オンラインデバッガ	svdebug	○	
管理ツール	svmap	○	
	svadm	○	
	svsitectl	○	
<ul style="list-style-type: none"> ・立ち上げ ・PU制御 	svrpl		○
	svcpuctl		○
稼働管理	svcpunow	○	
	svtimex	○	
保守コマンド	svdhp	○	
	svelog	○	

第2章 プログラム開発手順

2. 1 全体フロー

プログラム開発手順の全体フローを図1-4に示します。



<注意事項>
 CM空間は、CPU内のCP、HPにおいて共有する空間です。
 そのため、共有するサイトにおいて分割領域、細分割領域を同一に割り当ててください。

図1-4 プログラム開発手順全体フロー (1/2) (サイト構築からプログラム開発までの流れ)

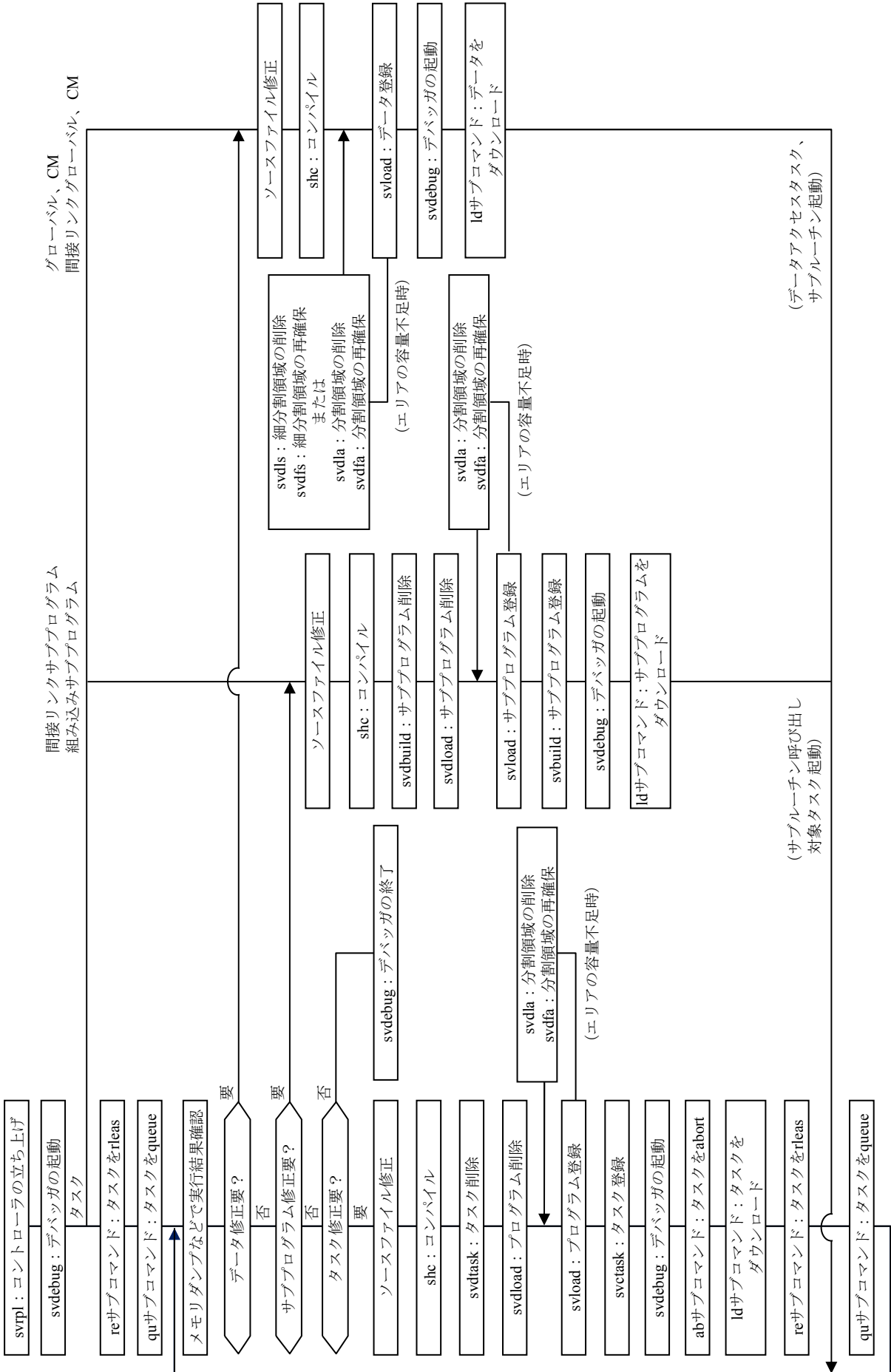


図1-4 プログラム開発手順全体フロー (2/2) (サイト構築からプログラム開発までの流れ)

2.2 サイト環境

RPDPはS10VEのCPUのコア（CP（Communication Processor）、HP（High-speed Processor））ごとにサイト名称を割り当て、コア単位にCPMS上で動作するタスク、サブプログラム、グローバルなどを管理します。

サイト名称はコア単位のユニークな名称です。これはBASE SYSTEM/S10VEで「Cモード利用」プロジェクトを生成したときにBASE SYSTEM/S10VEが決定します。プロジェクトのPCs番号ごとに、PCs番号cp, PCs番号hpの名称で作成されます。

BASE SYSTEM/S10VEはサイトごとにサイト名称のディレクトリ（これをサイトディレクトリと呼びます）を作成し、サイトディレクトリ下にベースサイトをコピーします。サイトにはサイト単位に所有する管理ファイルを配置します。ここの管理ファイルとは、S10VEメモリの初期値データファイルであるバックアップファイルや、バックアップファイル内に格納されているタスク、サブルーチン、グローバルなどを管理するファイルです。

サイトディレクトリはPCs番号単位に固定のディレクトリ（C:\\$S10VE¥PCs番号¥PCs番号_unit¥PCs番号cp）の下に作成されます。

また、RPDPではCPのサイト、HPのサイトを総称してCPUサイトとして管理します。CPU名称はCPに割り当てたサイト名と同一名称となります。

リモートローディングなど、CP、HPを同時に操作しなければならない場合にCPU名称を指定して操作します。

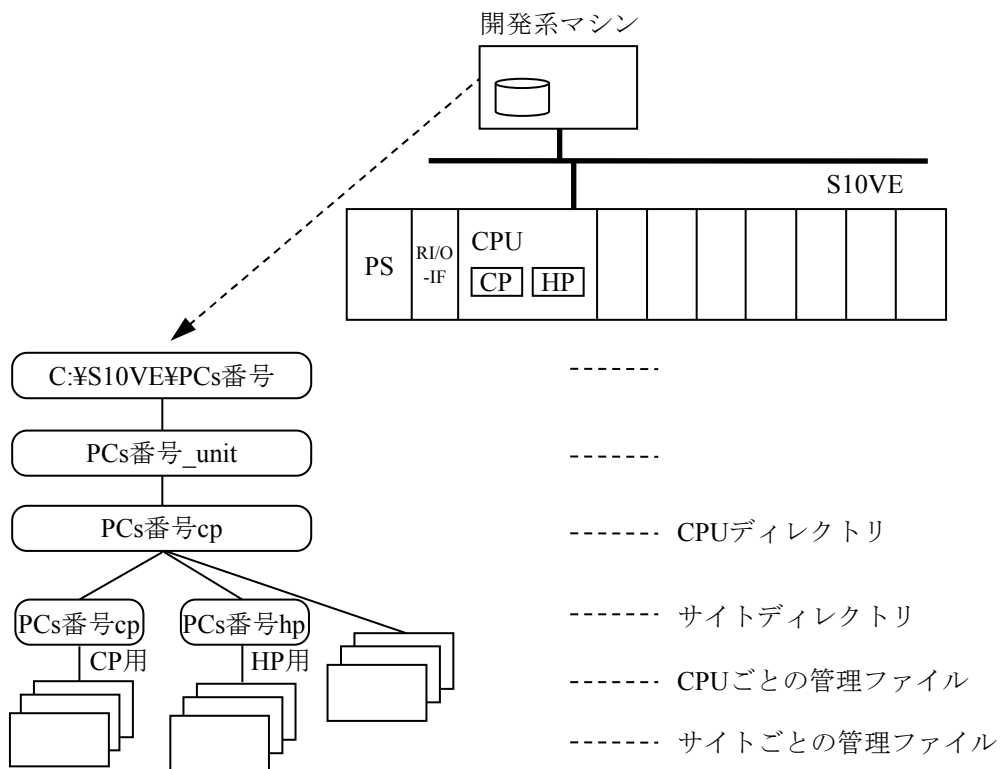


図1-5 S10VEのサイトディレクトリ構成

第2章 プログラム開発手順

2. 2. 1 サイト指定でのS10VEとの接続

BASE SYSTEM/S10VEで生成したサイトを指定して、リモートローディングやエラーログの収集など、S10VEに接続する操作を行うときは、BASE SYSTEM/S10VEで対象となるPCs番号のプロジェクトを開いて、接続PCs変更を行ってください。接続PCs変更を行ったIPアドレスのS10VEに接続します。

接続PCs変更を行ったあとはプロジェクトを開いている必要はありませんが、異なるS10VEに接続するときは、接続PCs変更で接続しなおす必要があります。

なお、RPDPはET.NETモジュールへの接続はサポートしていないため、必ずCPUモジュールに接続してください。

2. 3 主メモリのエリア管理と領域分割

RPDPは開発系マシン上でS10VE主メモリの管理を行います。エリア管理の目的は、プログラム、サブプログラム、データを主メモリ上に重複することなく効率よく配置することにあります。RPDPのエリア管理の対象となるメモリ空間は、物理メモリとS10VE上でCPMSが管理する論理空間です。物理メモリは用途ごとにGAREAを定義したサイズで論理空間の先頭からマッピングされます（CPUのCP、HPごとに各々のCPMSが論理空間を管理します）。

図1-6にCPMSが管理する論理空間を、表1-6に論理空間の用途を示します。

論理空間

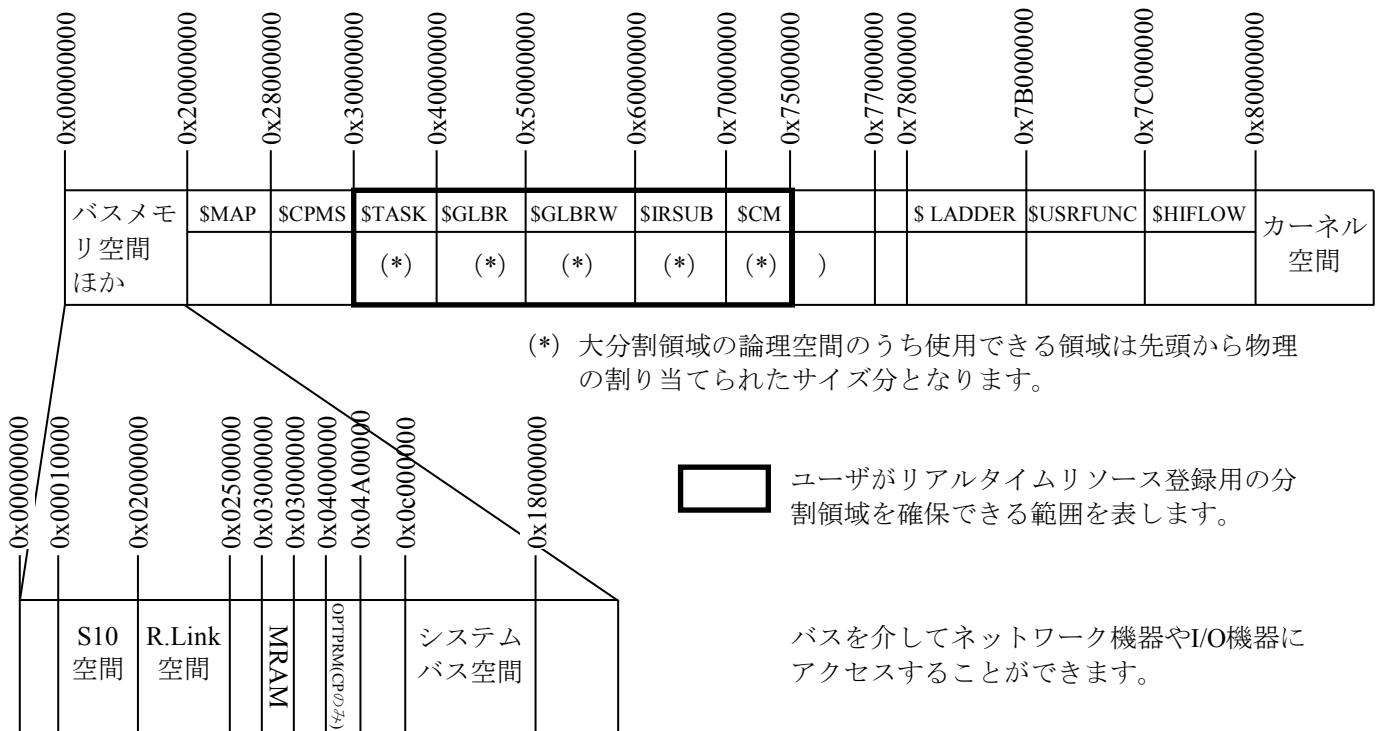


図1-6 CPMSが管理する論理空間

表1-6 各論理空間の用途

論理空間の大分割領域名	用途
\$TASK	タスク（プログラム）を格納するエリアです。
\$GLBR	読み出し専用GLBを格納するエリアです。
\$GLBRW	読み書き両用GLBを格納するエリアです。
\$IRSUB	サブプログラムを格納するエリアです。
\$SCM	PU間で共通に使用できる空間です。
\$LADDER	ラダープログラムを格納するエリアです（HPでのみ使用できる空間です）。
\$USRFUNC	ラダーのユーザ演算ファンクションを格納するエリアです（HPでのみ使用できる空間です）。
\$HIFLOW	HI-FLOWプログラムを格納するエリアです（HPでのみ使用できる空間です）。
\$MAP	RPDPが管理する下記のテーブルを格納します。 <ul style="list-style-type: none"> ・ IRSUBの間接リンクテーブル（IRSUBT） ・ タスクコントロールブロック（TCB） ・ IRGLBの間接リンクテーブル（IRGLBT） ・ 組み込みサブルーチンのテーブル（USLCB）
\$CPMS	CPMSが使用する空間です。

図1-6に示すCPMSが管理する論理空間のアドレスとサイズを表1-7に示します。

表1-7 各論理空間のアドレスとサイズ

S10VE			
論理空間の大分割領域名 (用途)	アドレス	サイズ	SH4の仮想アドレス 空間領域名
リザーブ	0x00000000-0x0000ffff	64KB	P0領域 (U0領域)
S10空間	0x00010000-0x01ffffff	32MB-64KB	
R.Link空間	0x02000000-0x02500000	5MB	
リザーブ	0x02500000-0x02ffffff	11MB	
MRAM	0x03000000-0x032fffff	3MB	
リザーブ	0x03300000-0x03ffffff	13MB	
OPTPRM	0x04000000-0x049fffff	10MB	
リザーブ	0x04a00000-0x0bffffff	118MB	
バスメモリ空間	0x0c000000-0x17ffffff	192MB	
リザーブ	0x18000000-0x1ffffff	128MB	
\$MAP	0x20000000-0x27fffff	128MB(1.5MB)	
\$CPMS	0x28000000-0x2ffffff	128MB	
\$TASK	0x30000000-0x3ffffff	256MB(8MB)	
\$GLBR	0x40000000-0x4ffffff	256MB(4MB)	
\$GLBRW	0x50000000-0x5ffffff	256MB(4MB)	
\$IRSUB	0x60000000-0x6ffffff	256MB(4MB)	
\$CM	0x70000000-0x74fffff	80MB(2MB)	
リザーブ	0x77000000-0x77ffff	16MB	
\$LADDER	0x78000000-0x7affffff	48MB(8MB)	
\$USRFUNC	0x7b000000-0x7bfffff	16MB(2MB)	
\$HIFLOW	0x7c000000-0x7ffffff	64MB(8MB)	
カーネル空間	0x80000000-0x9ffffff	512MB	P1領域
	0xa0000000-0xbffffff	512MB	P2領域
リザーブ	0xc0000000-0xdffffff	512MB	P3領域
	0xe0000000-0xfffffff	512MB	P4領域

各論理空間のサイズの()内の値は物理メモリの割り当てられているサイズ。

第2章 プログラム開発手順

図1-7に物理メモリマップの詳細を示します。

0x04000000	0x04080000	0x041f0000	0x04360000	0x04380000	0x04d80000	0x04f80000	0x05580000	0x06780000	0x05f80000	0x06780000	0x07d00000	0x09280000	0x0935c000	0x0befe000	0x0c000000
OS										HP用ユーザ空間	CP用ユーザ空間				
SPM, HKP	HP用 CPMS	CP用 CPMS	OPT PRM	CM	F.U.	LAD DER	USRF UNC	HIFL OW	HP用タスク、サブプログラム、GLB	CP用タスク、サブプログラム、GLB	HP用 OSワーク	CP用 OSワーク	空きメモリ	ネットワークバッファ	システムリザーブエリア (16MB+256KB)
512 KB	1472 KB	1472 KB	128 KB												

図1-7 S10VEの物理メモリマップ

SPM : CPMSのエディションデータです。

HKP : Hardware KROM ProgramがROMからコピーされて動作する空間です。

CPMS : OSのプログラム本体です。CPUのHP (High-speed Processor) 、CP (Communication Processor) 上で動作する各々のCPMSが配置されます。末尾の128KBの空間は、HP用CPMS、CP用CPMSが共通に使用するメモリです。

OPTPRM : オプションモジュールのパラメータが設定されるエリアです。

CM : CPUのCP、HPの両プロセッサの論理空間上のCM空間からアクセスすることができるプロセッサ間の共有メモリです。

F.U. : 将来用

LADDER : ラダーのプログラムがダウンロードされるエリアです。

USRFUNC : ラダーのユーザ演算ファンクションがダウンロードされるエリアです。

HIFLOW : HI-FLOWのプログラムがダウンロードされるエリアです。

タスク、サブプログラム、GLB用エリア :

タスク、サブプログラム、GLBがダウンロードされるエリアです。

マップ情報、TCB、IRSUBT、IRGLBT、USLCBもこのエリアにダウンロードされます。

HP用OSワーク、CP用OSワーク、ネットワークバッファ :

OSが使用するバッファエリアです。DHPエリア、ネットワークバッファなどから構成されます。

システムリザーブエリア :

ハード的にリザーブされているエリアです。

● タスク、サブプログラム、GLB用エリア詳細

以下にタスク、サブプログラム、GLB用エリアの詳細を示します。

タスク、サブプログラム、GLB用エリア				
\$MAP	\$TASK	\$GLBR	\$GLBRW	\$IRSUB

\$MAP : S10VEメモリ上の管理情報を格納するエリアです。

\$TASK : タスク (プログラム) を格納するエリアです。

\$GLBR : 読み出し専用GLBを格納するエリアです。

\$GLBRW : 読み書き両用GLBを格納するエリアです。

\$IRSUB : サブプログラムを格納するエリアです。

CPMS上で動作するタスクやサブプログラム、またはそれらが使用するデータを生成する場合には、初めにタスクやサブプログラム、データを格納する大分割領域 (GAREA) に分割領域 (area) を確保します。グローバル (GLB) およびCMの場合は、さらに分割領域を細分割領域 (sarea) に分けます。タスクやサブプログラムは細分割領域名称を指定してデータをアクセスします。

(1) 分割領域

分割領域 (area) はsvdfaで確保し、svdlaで解放します。大分割領域 (GAREA) 内に複数の分割領域を確保できます。

分割領域を確保すると、確保したサイズ分のバックアップファイルが生成されます。

(2) 細分割領域

svdfaで確保した1つの分割領域の中には複数のリソースが配置できます。タスクおよびサブプログラムはsvloadで分割領域内に配置し、svdloadで解放します。GLB、CMの細分割領域 (sarea) はsvdfsで確保し、svdlsで解放します。

分割領域 (area) を解放するときは、svdlaで解放してください。

2. 4 タスクのエリア配置

CPMS上で動作するすべてのタスクは1つの論理空間内に配置されます。タスクは、\$TASKに確保した分割領域内に格納します。1つの分割領域内には複数のタスクを格納することができます。

textはページ境界（4KB境界）に、data/bssとstackは8バイト境界に配置します。

また、同一タスクのtext/data、stack/bss、およびOSワークは別ページに配置します。

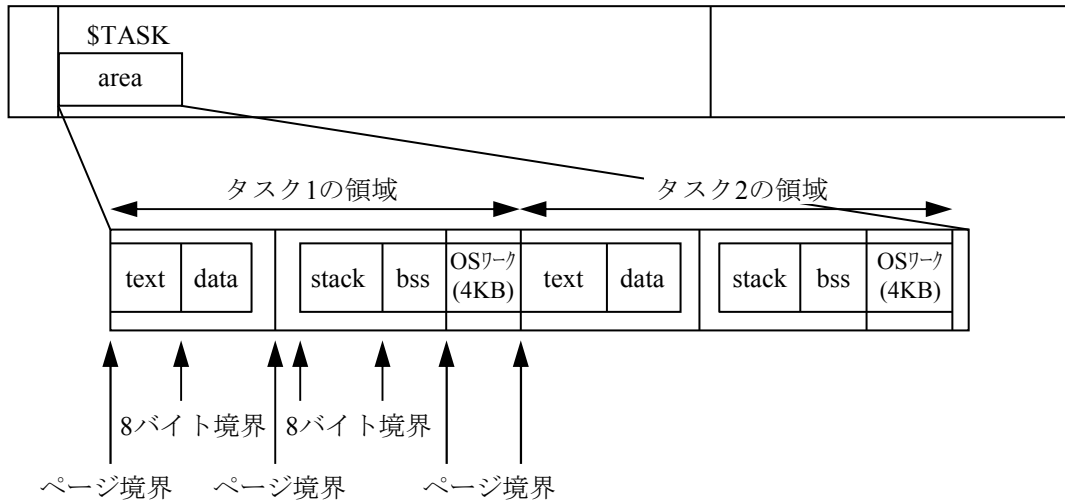


図1-8 論理空間内のタスク配置

● マルチタスクのスタック配置

マルチタスクのときはstackとbssそれぞれを別ページに配置します。

stackとOSワークはマルチタスクの個数分ロード時に確保し、どこを使用するかはタスク生成時にユーザが指定します（下図は、タスク1とタスク2がマルチタスクの場合を示します）。

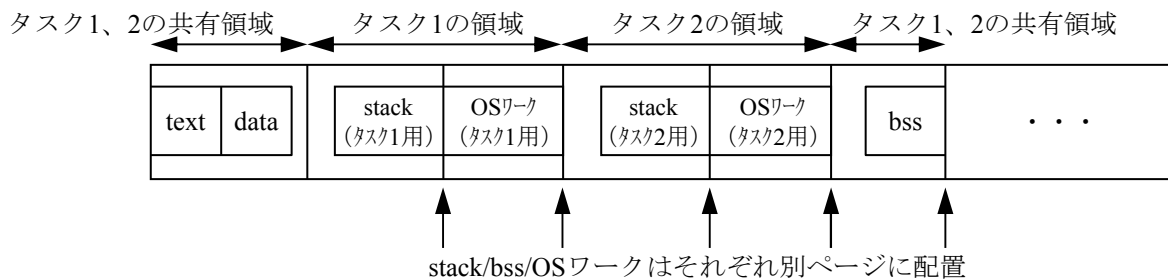


図1-9 論理空間内のタスク配置（マルチタスク）

マルチタスクとは、プログラムのメモリ容量削減を目的とし、1つのプログラムに対してn個のタスクとして生成したものです。マルチタスクは、ロードしたプログラムのtext/data/bss部を共有し、stack部を各タスクに別エリアに配備することで実現しています。

このため、マルチタスクを組む1つのタスクで、bss部にデータを書き込むとマルチタスクを組む他のタスクにもその情報が伝わるため、初期状態bssを期待し動作するタスクの動作保証ができませんので注意してください。したがって、マルチタスクの場合は、bssにデータを書き込まないでください。

2. 5 IRSUBのエリア配置

CPMS上で動作するすべてのIRSUBは1つの論理空間内に配置されます。IRSUBは、\$IRSUBに確保した分割領域内に格納します。1つの分割領域内には複数のIRSUBを格納することができます。textは32バイト境界に、dataは8バイト境界に配置します。

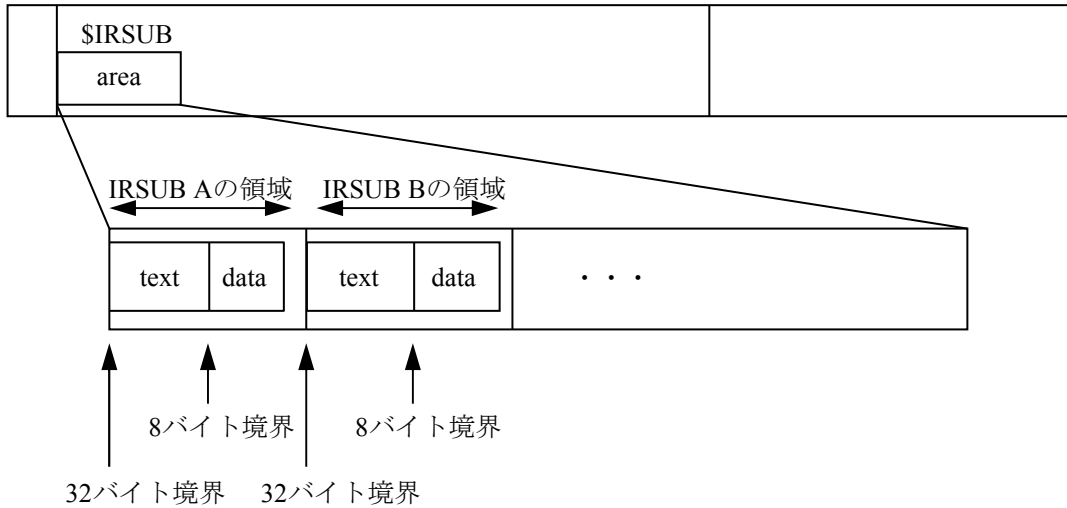


図1-10 論理空間内のIRSUB配置

● マルチエントリ配置

IRSUBマルチエントリでも、text/dataの配置は変わりません。

RPDPの管理ファイル上にマルチエントリされたエントリ名称と相対エントリアドレスを持ち管理します。図1-11にマルチエントリIRSUB（エントリ名称A、B、C）の配置を示します。

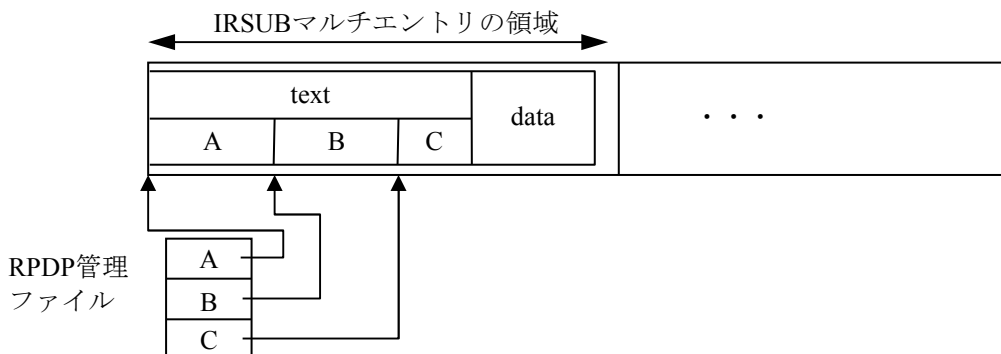


図1-11 論理空間内のIRSUB配置（マルチエントリ）

2. 6 プログラムのロードとタスクの生成

アロケータによって決定された管理情報に基づいて、ローダ (svload) はプログラムやデータを area、sareaにロードします。

ローダはグローバルなどのCPMSの資源情報をエリア管理情報から引き出し、それらを設定しながら実行可能モジュールを生成します。生成した実行可能モジュールは開発系マシンのバックアップファイルに格納されます。

プログラムとしてロードした実行可能モジュールは、ビルダ (svctask) を用いてタスクとして登録します。svctaskはCPMSが管理するタスク制御ブロック (TCB : Task Control Block) にそのタスクの属性を設定します。

2. 7 間接リンク常駐サブプログラム

タスクがいくつかのサブプログラムから構成されているとします。このサブプログラムのうち、タスク本体に組み込まれているサブプログラムを内部サブプログラム (ISUB) と呼びます。タスク本体とは別な場所にあり、他のタスクからも共有できるように主メモリ上に常に存在するサブプログラムを常駐サブプログラム (RSUB) と呼びます。

このRPDPでは間接リンクRSUB (IRSUB) をサポートしています。IRSUBはタスクからリンクする管理テーブルを設け、この管理テーブルをリンクするタスクを変更することなく、IRSUB本体の変更を容易に行えます。

IRSUBの本体を更新するのはローダ (svload) です。間接リンク用の管理テーブルの更新はビルダ (svbuild) が行います。

2. 8 グローバル (GLB)

CPMSではタスク間の主メモリの共有方法としてGLBを提供しています。これは、アロケータを使ってあらかじめCPMSタスクの論理空間内のGLB空間に領域を確保して名前を付けておき、その領域を複数のタスクやサブルーチンで共同に使用するものです。svdfaを用いて、この領域をareaに分割し、さらにsvdfsを用いてsareaに細分割して使用します。

2. 9 PU間共有メモリ (CM)

CPMSでは同一ユニット内のPU間の主メモリを共有する方法としてCMを提供しています。これは、グローバル (GLB) と同様に、アロケータを使ってあらかじめCPMSタスクの論理空間内のCM空間に領域を確保して名前を付けておき、その領域を同一ユニット内の複数のタスクやサブルーチンで共同に使用するものです。svdfaを用いて、この領域をareaに分割し、さらにsvdfsを用いてsareaに細分割して使用します。

CMを共有するPU間で同一のsarea名で共通の領域を参照するためには、各PUに対応するサイトでCMに配置する分割領域、細分割領域の名称とアドレスを同一に定義しなければなりません。各サイトでCM空間に配置する分割領域のアドレスを同一とするためには、svdfsの-fオプションを使用して分割領域を確保します。

-fオプションを使用して、大分割領域\$CMの先頭からの相対アドレスを指定することにより、各サイトでの分割領域の配置を同一にすることができます。

CM領域の使用の詳細は、「6. 1. 5 CM用の分割領域の確保」を参照してください。

2. 10 バリュ (VAL)

ユーザは、プログラム間で共通に使用する定数を外部名称として登録し使用できます。これをバリュ (VAL) と呼びます。VALの登録はsvdfv、削除はsvdlvを用いて行います。

VALはロードモジュールをバックアップファイルにロードするときにローダが設定します。したがって、VALを参照するタスク、サブプログラムをロードする前にVALを登録しておいてください。

2. 11 間接リンクグローバル

RPDPでは間接リンクグローバルをサポートしています。間接リンクグローバルは、グローバル本体にリンクする管理テーブルを設定し、このリンク管理テーブルの内容を更新することにより、グローバル本体の更新を容易にしています。

グローバルの本体を更新するのはアロケータ (svdfs)、ローダ (svload) です。svdfsでエリアを定義し、svloadで初期値をロードします。間接リンク用の管理テーブルの更新はビルダ (svirglb) が行います。

2. 12 GLB、VAL、IRSUBプログラミングガイド

プログラム、サブプログラムから使用するGLB、VAL、IRSUBのコーディング方法、リンク方法について示します (CMの使用方法はGLBと同様です)。

(1) GLB、VALの名称の付け方

表 1 - 8 GLB、VALの名称の付け方

項目	仕様
最大文字数	14文字 (_g、_vは除く)
名称規則	英字、数字、_ (アンダーライン) ただし、先頭は英字に限ります。 最終文字は属性を表し、以下の形式にしてください。 GLBのとき_g VALのとき_v
名称のユニーク性	同一名称は使えません。

(2) GLB、VALの使い方

表1-9にGLBおよびVALの使い方を示します。

表1-9 GLBおよびVALの使い方

No.	項目	C言語
1	GLBの宣言 (参照側)	extern long name_g[size]; [説明] name : GLB名称 size : GLBのサイズ
2	GLBの参照 (細分割領域指定)	extern long name_g[size]; main() { long i; i = name_g[index]; } [説明] name : GLB名称 size : GLBのサイズ
3	GLBの宣言 (被参照側)	宣言する必要はありません。 No.4に示すように初期値を設定してください。
4	GLBへの初期値設定	long name_g[size] = {1, 2, 3, ...}; [説明] name : GLB名称 size : GLBのサイズ
5	VALの参照	extern long name_v; long y = (long)&name_v; main() { long x; x = y; } [説明] name : VAL名称

(3) GLBデータ参照時の注意事項

プログラム作成時にGLBデータを参照する場合、参照するGLBが同一プログラム上で初期値を定義しているかどうかによってデータの取り扱いが異なります。そのため、以下に示す点に注意してGLBデータを参照するプログラムを作成してください。

① 参照するGLBが同一プログラム内で定義されていない場合

上記の条件は、ソースプログラムまたはsvloadで結合されるオブジェクトファイルの基となるソースプログラム内で参照するGLBが表1-9のNo.3、No.4に示すような定義をしていない場合です。

この場合、以下の(a)、(b)について注意してください。

(a) GLBの宣言

GLBの宣言は、表1-9のNo.1に示すように各名称に対する容量宣言が行えます。コンパイラやアセンブラは、この容量に対して、svdfsコマンドで確保した領域の大きさとの合理性チェックは行いません。したがって、プログラムが実際の領域を超えたアドレスを参照してもエラーとなりません。

(例) 宣言したエリアを超えたアドレスの参照

<アロケータ>

```
svdfs usrresp0 glb2 100
```

<c>

```
extern long glb2_g[100] ;
```

```
⋮
```

```
⋮
```

```
glb2_g[100] = ……;
```

} エラー検出されません。

(b) 相対アドレスの参照

GLBは、名称± α (α は相対バイトアドレス)の形式で参照できます。その場合の範囲は、 $-2^{31} \leq \alpha \leq 2^{31}-1$ です。

② 参照するGLBが同一プログラム内で定義されている場合

上記の条件は、ソースプログラムまたはsvloadで結合されるオブジェクトファイルの基となるソースプログラム内で参照するGLBが、表1-9のNo.3、No.4に示すような定義をしている場合です。

この場合、以下の(a)～(c)について注意してください。

(a) GLB名称だけの参照

プログラム中に初期値定義のあるGLBの名称だけを参照する場合は、特に制限事項はありません。

(例) 名称だけの参照

<c>

```
extern long glb2_g ;
```

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

```
⋮
```

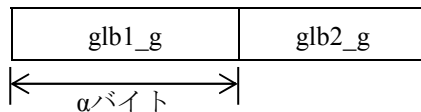
```
⋮
```

```
⋮
```

```
glb2_g = glb1_g[0] ;
```

(b) GLB先頭から相対アドレスを参照

名称+ α の形式で参照する場合、 α の値は定義した範囲を超えてはいけません。なお、範囲外チェックエラーは検出しませんので注意してください。



すなわち、 $glb1_g + \beta$ と記述する場合、 $0 \leq \beta < \alpha$ でなければなりません。

(例) 相対アドレス参照

<c>

```
int glb1_g[3] = { 1, 2, 3 } ;
```

```
int glb2_g[2] = { (long)&glb1_g[0], 0 } ; ... 相対アドレスは範囲内です。
```

```
a = glb1_g[3] + glb2_g[4] ; ..... 相対アドレスは範囲外です。
```

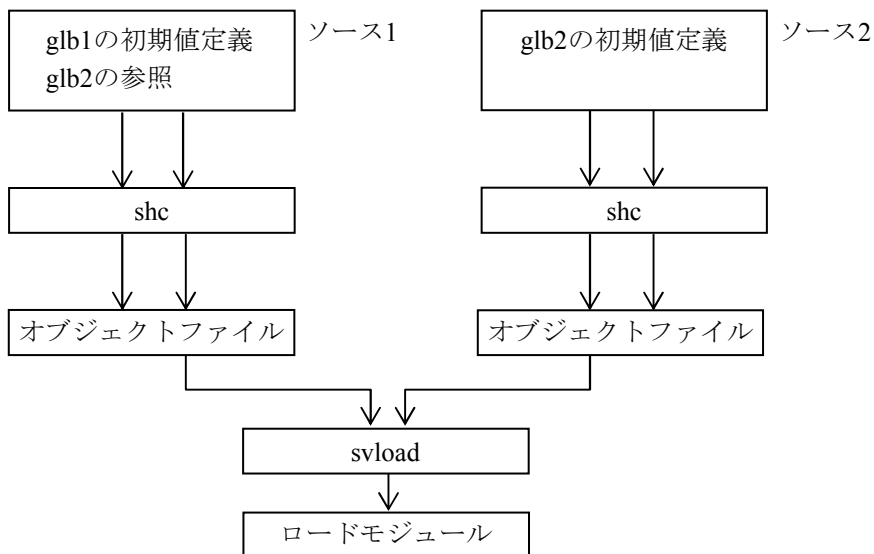
(c) svload操作上の注意事項

例えば、text部とGLB初期データを持つソースプログラムから作成されたロードモジュールをsvloadでロードするとき、単にプログラムまたはサブプログラムとしてロードしたのでは初期値データはロードされません。初期値データをロードするためには改めて+Dオプションを付けてsvloadを実行してください。つまり1つのロードモジュールに対してオプションを変えて2度svloadを行うことになります。初期値データだけのファイルを作成し、別々に作業してください。1つのソースプログラムで複数のGLB初期値を定義できます。

③ リンク上の注意事項

すでに①、②でも述べましたが、svloadでオブジェクトファイルを複数結合する場合、例えそれぞれが別のソースファイルであっても、結合した結果1個のファイルであるものとします。

(例) 2つのオブジェクトファイルの結合



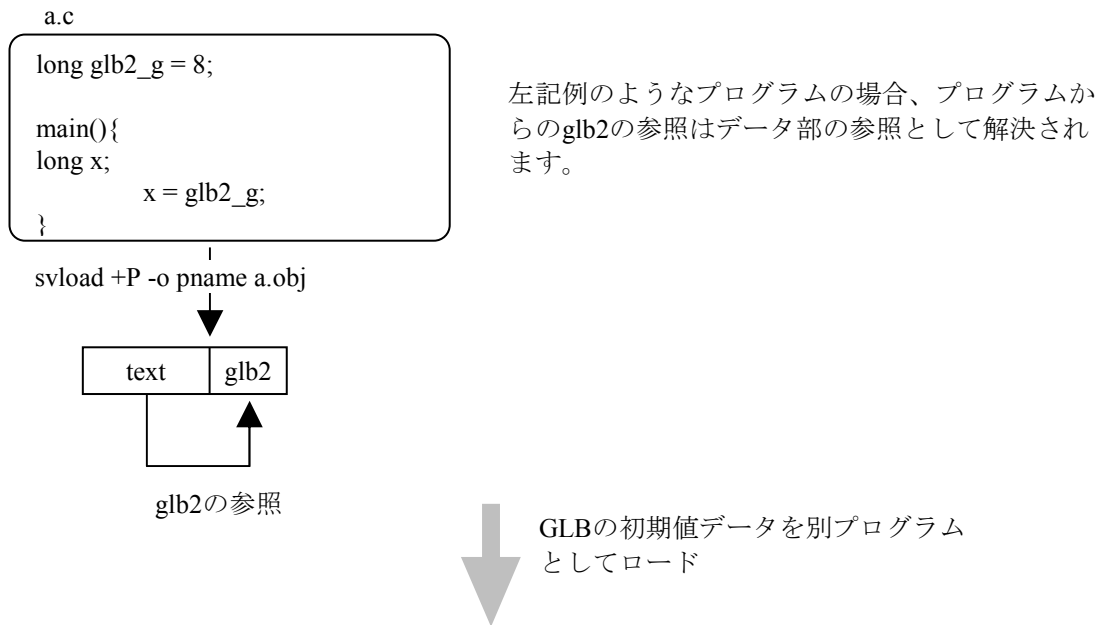
この例ではソース1とソース2は1つのソースプログラムとみなします。

④ 参照するGLBの初期値を同一プログラム上で定義している場合

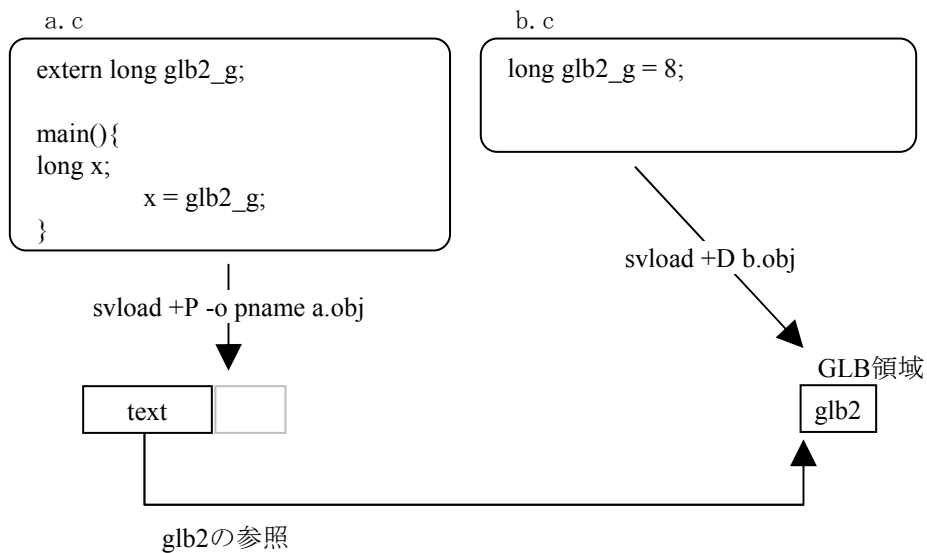
プログラム作成時にGLBデータを参照する場合、参照するGLBの初期値を同一プログラム上で定義しないでください。

GLBの初期値定義と参照が同一プログラム内にある場合、ローカルなデータの参照として解決されます。

(例1) プログラムとGLBデータを同一のプログラムに記述した場合



(例2) プログラムとGLBデータを分離した場合



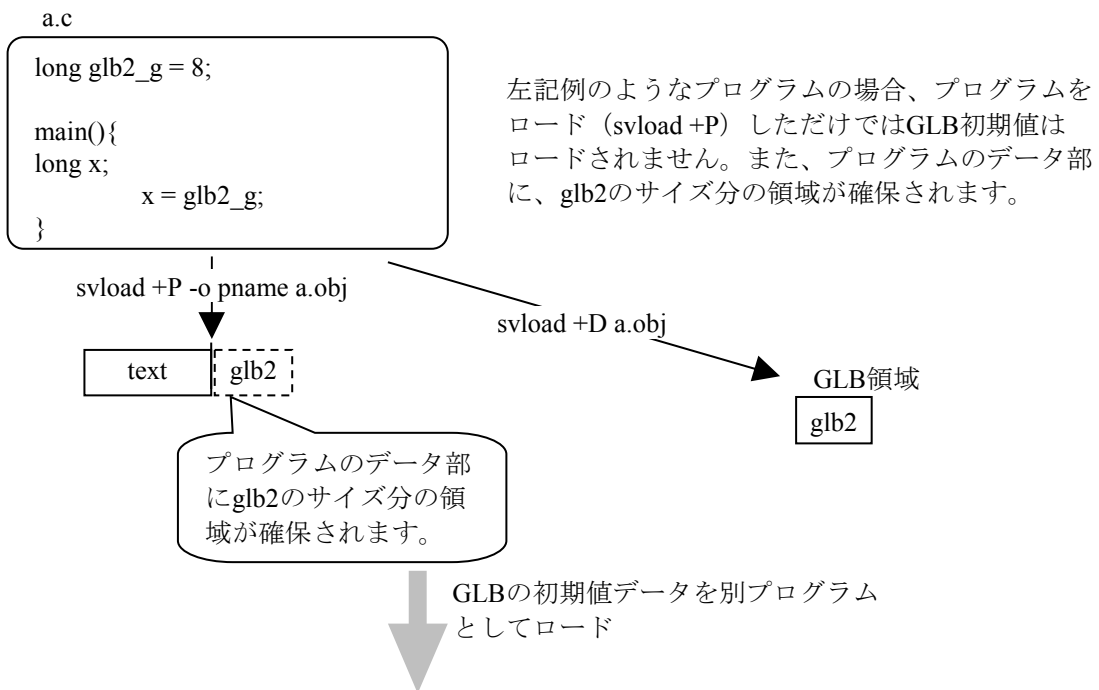
(4) GLBデータロード時の注意事項

① プログラムとGLBデータの混在

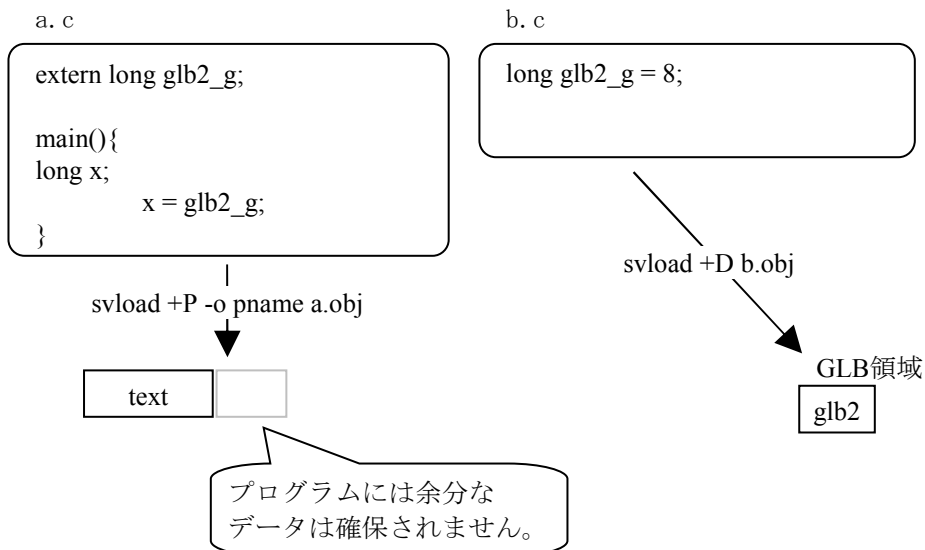
テキスト部を持つプログラム、サブプログラムと、GLBの初期値データを同一のプログラムに記述した場合、ローダ (svload) でプログラム (+P)、サブプログラム (+I) としてロードしても、GLBの初期値データはロードされません。GLBの初期値データをロードするためには、改めて+Dオプションを指定してロードしなければなりません。つまり1つのプログラムのオプションを変えて2回ロードすることになります。

また、プログラム、サブプログラムのデータ部にはGLB初期値データのサイズ分のエリアが確保されてしまいます。このため、GLBの初期値データを定義するプログラムは、GLBに初期値を定義するだけのプログラムとしてください。

(例1) プログラムとGLBデータを混在させた (同一のプログラムに記述した) 場合



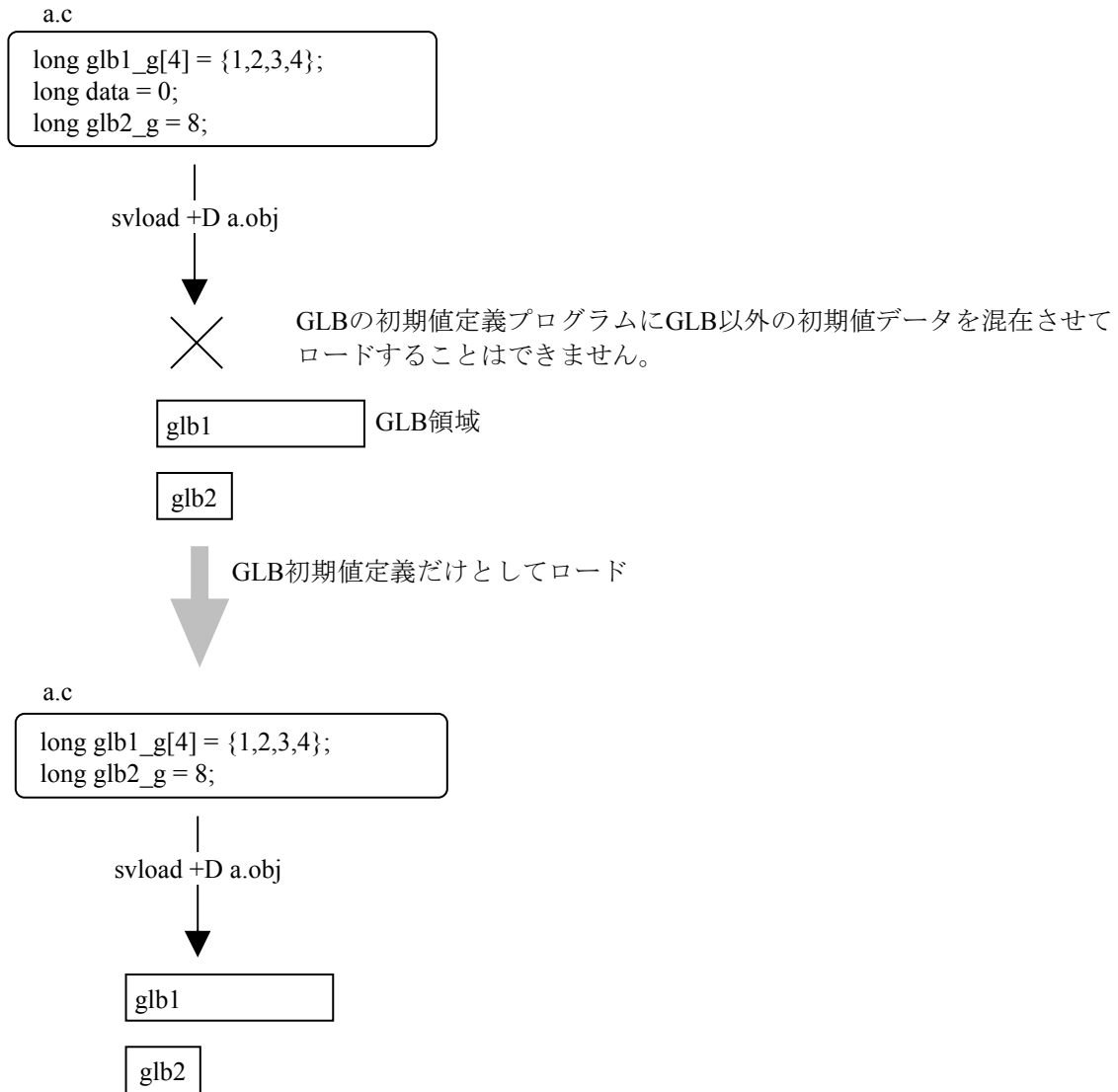
(例2) プログラムとGLBデータを分離した場合



② GLB初期値とGLB以外のデータの混在

1つのプログラムに複数のGLB初期値データを定義することができます。ただし、GLBの初期値定義プログラムにはGLB初期値以外のデータを定義することはできません。

(例) 同一のプログラムにGLB初期値定義とGLB以外のデータを混在させた場合



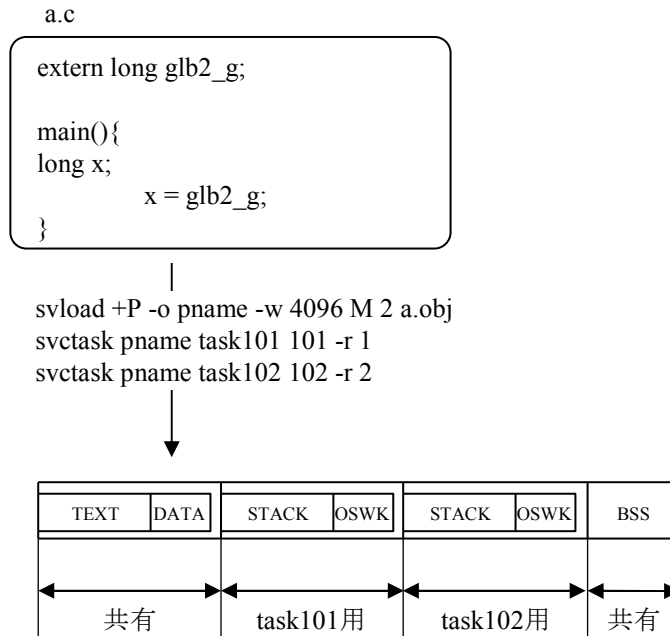
(5) IRSUBの使い方

表1-10 IRSUBの使い方

No.	項目	C言語
1	IRSUBの参照 (IRSUB名称指定)	<pre>main() { name(); }</pre> <p>[説明] name : IRSUB名称</p>
2	IRSUBの参照 (irsubad関数指定)	<pre>void *irsubad(); main() { long no; long (*adr)(); no = xxx; adr = irsubad(no); if(adr != 0) { (*adr)(); } else { /* IRSUB未登録処理 */ } }</pre> <p>[説明] xxx : IRSUB番号 irsubad関数を使用して間接リンクテーブルからアドレスを参照して使用します。</p>

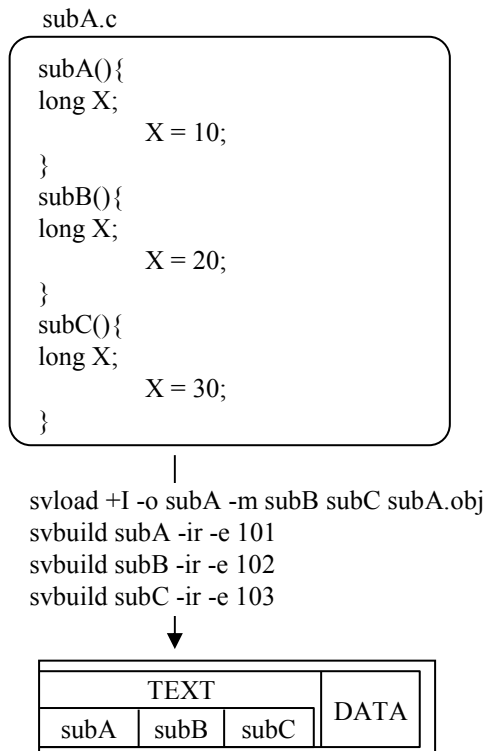
(6) マルチタスクの生成方法

1つのプログラムを2つのマルチタスクとして生成する場合、下記のようにsvload/svctaskコマンドを実行します。



(7) マルチエントリIRSUBの生成方法

サブプログラムを3つのマルチエントリIRSUBとして生成する場合、下記のようにsvload/svbuildコマンドを実行します。1つのソースファイル内に3つのモジュールがある場合を以下に示します。



2. 13 CPMS上のプログラム作成の制約条件

CPMS上で動作するリアルタイムプログラム作成に当たっては、以下に示す制約事項があります。

(1) オーバレイ構造不可

CPMSはタスク、常駐サブプログラムのオーバレイ構造は許していません。したがって、タスク、常駐サブプログラムを作成するときは、そのプログラムが大きくなるように注意してください。

(2) バルクサブルーチン未サポート

CPMSは補助メモリ上にサブルーチンを置いておき、必要に応じて主メモリに乗せて動作させるバルクサブルーチンはサポートしていません。間接リンク常駐サブプログラム (IRSUB) とするか、タスク内に組み込んだ内部サブプログラム (ISUB) としてください。

(3) 間接リンク常駐サブプログラム (IRSUB) 作成上の注意

IRSUBは主記憶装置上に常駐し、複数のメインプログラムから共通に使用されます。そのため、IRSUBはこれを使用するメインプログラムとは独立な主記憶装置の領域を占有します。また、同一時刻に複数のメインプログラムにより使用されるのでリエントラントにしてください。

リエントラント性のないプログラムはIRSUBにはできません。リエントラントとは、あるメインプログラムがそのIRSUBを使用中でも、他のメインプログラムがその同じIRSUBを使用できることを言います。

以下に正しいIRSUBの作成方法について説明します。

リエントラントなIRSUBは処理手続き部 (text部)、データ部 (data部) からなる不変部分と作業エリアからなる可変部分の2つに分離されます。不変部分は複数のメインプログラムが共有します。可変部分は各メインプログラムが各メインプログラムの可変部分に確保し、IRSUBはメインプログラムに確保された可変部分を使用します。したがって、IRSUBが使用する可変部分はスタックエリアを参照するようにプログラミングしてください。IRSUBは初期値なし作業エリア (bss部) を使用できません。

リエントラントなIRSUBを作成する場合、下記の3点に注意してください。

- (a) 作業エリアはすべてスタックとする。
- (b) IRSUBが複数のプログラムで構成されている場合、プログラム間共通エリアを使用しない。
- (c) 静的変数の初期値を定義している場合、その値を変更しない。

上記(a)、(b)は、コンパイルリストまたはリンケージマップリスト中のセクション情報で、Bセクションのサイズが0になっていることで確認できます。

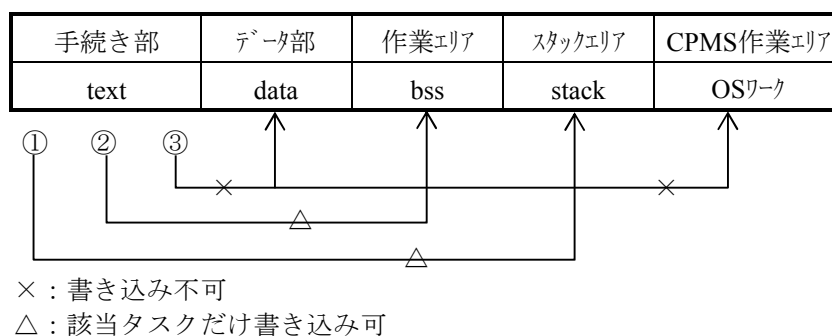


図1-12 書き込みの可否

- ① : スタックエリアへの書き込みを示します。該当タスクはスタックエリアへ書き込むことができます。
- ② : 作業エリアへの書き込みです。通常IRSUBでは作業エリアを確保しない、また書き込みもしないでください。該当タスクは作業エリアへ書き込むことができます。
- ③ : データ部への書き込みです。タスクはデータ部へ書き込めません。

以下に各言語でのリエントラントなIRSUBの作成時の注意点を示します。

<Cプログラム例>

```
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

:
:
:
}
```

- ①で宣言したb1に対して、書き込みを行うプログラムはノンリエントラントとなります。
- ②で宣言したd1に対して、書き込みを行うプログラムはノンリエントラントとなります。
- ③で宣言したb2に対して、書き込みを行うプログラムはノンリエントラントとなります。
- ④で宣言したd2に対して、書き込みを行うプログラムはノンリエントラントとなります。
- ⑤で宣言したb3に対して、書き込みを行うプログラムはノンリエントラントとなります。
- ⑥で宣言したd3に対して、書き込みを行うプログラムはノンリエントラントとなります。
- ⑦、⑧で宣言したs1、s2に対して、書き込みを行ってもプログラムのリエントラント性は損なわれません。IRSUBとして使用する場合には、⑦、⑧のような変数だけにしてください。

以下に各変数が割り当てられている領域について説明します。

b1は通常bss領域に割り当てられます。 (*)

b2はbss領域に割り当てられます。

b3はbss領域に割り当てられます。

d1はdata領域に割り当てられます。

d2はdata領域に割り当てられます。

d3はdata領域に割り当てられます。

s1はstack領域に割り当てられます。

s2はstack領域に割り当てられます。

(*) 他のプログラムでb1に初期値を設定している場合にはdata領域に割り当てられます。

(4) プログラムの再配置はできない。

プログラム、サブプログラムの再配置はありません。一旦動作エリアが確定したプログラム、サブプログラムを別のエリアでそのまま動かすことはできません。プログラム、サブプログラムを別のエリアに動かすにはプログラム、サブプログラムを削除し、再登録してください。

(5) 名称は14文字まで

プログラム、サブプログラムの名称の文字数は、半角英数字および_（アンダーライン）で14文字以内としてください。GLB、VALの名称も14文字以内としてください。C言語でのGLB、VALの表現は、これらの名称のあとに各々_g、_vを付けた16文字以内としてください。

(6) GLB、VALの名称

_g、_vで終わる名称は外部名として宣言すると、GLB、VALとして扱われます。したがって、GLB、VALを使わないプログラムでは、名称の末尾が_g、_vとならない名称を付けてください。また、_bで終わる名称も将来拡張用として予約されていますので使用しないでください。

(7) 外部名称はユニークにしてください。

外部名称はシステム内のGLB名、プログラム名、サブプログラム名、VAL名全体でユニークに付けてください。

(8) 使用してはいけない名称があります。

プログラム作成において使用できない名称や、使用上注意の必要な名称があります。詳細については、「付録A プログラムで使用できる名称」を参照してください。

(9) プログラムの構造

CPMSのもとで動くプログラムの構造は次のようになっています。

text	プログラムのプロシジャが入るエリア
data	プログラムの初期データが入るエリア
stack	タスクが使う動的なワークエリア
bss	プログラムが使う静的なワークエリア
OSワーク	OSが使う動的なワークエリア

これらのエリアの大きさは各々4バイトの整数に補正され、かつ各エリアの先頭アドレスも8または4096の倍数になるように配置されます。配置の詳細については2.5節を参照してください。

(10) 先頭アドレスの制約

GLBエリアはデフォルト値では4の倍数になるようにアロケータで補正します。

(11) 初期値の取り扱い

初期値の取り扱いは、下記のようになりますので注意してください。

領域	CPMS
data	プログラムされた値
bss	不定
stack	不定

(12) GLB、CM初期値設定データの大きさ

オブジェクトファイルのデータの大きさは、コンパイラのアラインメント処理によって、ソースプログラムで定義した以上の容量となる場合がありますので注意してください。以下に具体例を示します。

CPMSでは、データアクセスの高速化を目的とし、データ型などに合わせ配置（アドレス）を固定化するナチュラルアラインメント方式を採用しています。データの配置は、コンパイラやリンカが自動的に行うため、ユーザは、アラインメントを意識しないでコーディングすることができます。しかし、GLBやCMの初期値設定データの実際のサイズは、次に示すようにコーディングした構造体のサイズより大きくなる場合がありますので注意してください。

<コーディングの大きさ：16バイト> <初期値設定データのメモリ上の大きさ：24バイト>

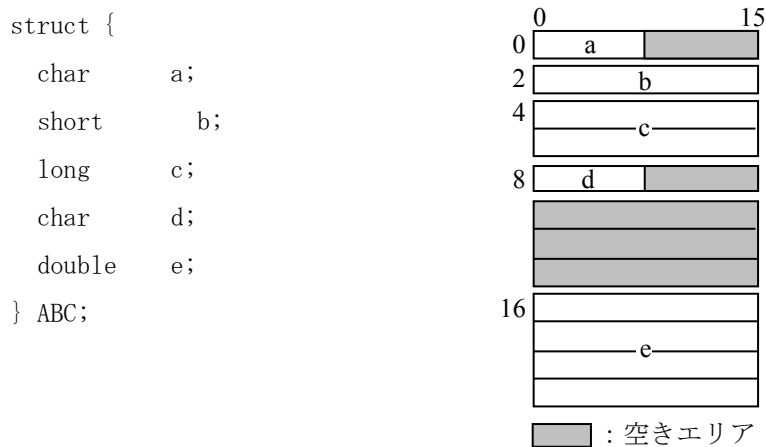


図1-13 データサイズ比較

<対処方法>

(1) 構造体内の空きエリアをなくすような構成（データ配置順）とすること。

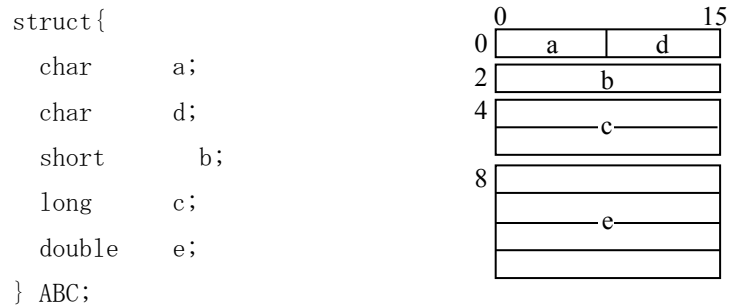


図1-14 データ配置順を考慮した構造体宣言例

(2) やむを得ず空きエリアが入る場合は、構造体内に空きエリアであることを明示的に宣言すること（明示的にしないとマシンによって空きエリアが確保される場合と確保されない場合がある）。

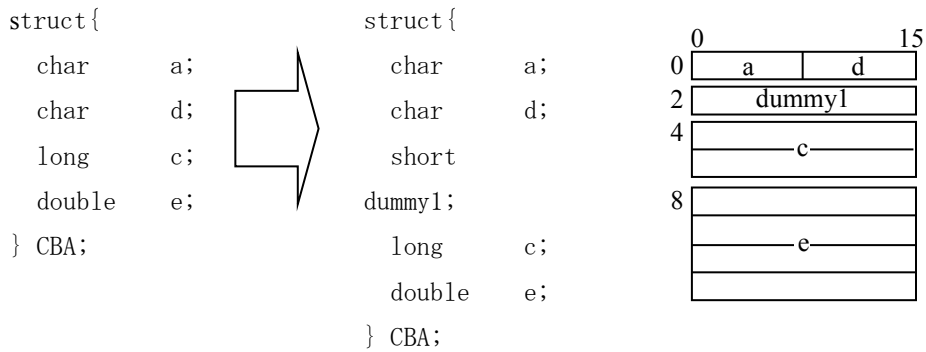


図1-15 空きエリアを明示的に宣言した例

(3) 構造体の大きさ（バイト数）は、構造体内の最大型の倍数となるようにすること。構造体の配列を確保するとき、2つ目以降の構造体先頭アドレスが、アクセス可能なアドレスに割り当てられるようにする。

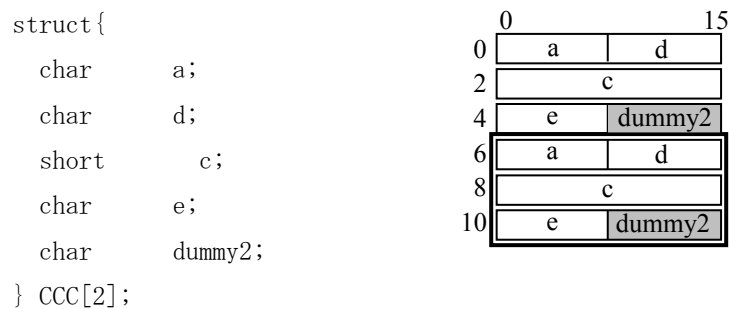


図1-16 構造体サイズを考慮した宣言例

第3章 インストールと実行環境

3. 1 インストール

RPDPを含むP.P. (プログラムプロダクト: P.P.名称 RPDP/S10VE) をインストーラでインストールします。RPDP/S10VEのディスクをCDドライブに挿入し、エクスプローラーから、CDデバイスのS789810フォルダ下にあるsetup.exeを実行してください。インストールは、Administratorでログインしてから実施してください。

3. 2 前提P.P.

RPDPを使用するためには表1-11に示す前提P.P.が必要です。また、RPDPを使用してプログラムのコンパイル・ロードを行うためには、SHCコンパイラ (Ver.9.04 Release 00) が必要です。

表1-11 RPDPの前提P.P.

名称	型式
CPMS/S10VE	S-7898-05
RCTLNET/S10VE	S-7898-60
RPDP/S10VE	S-7898-10
BASE SYSTEM/S10VE	S-7898-38

3. 3 インストール時の注意事項

3. 3. 1 RPDPインストール時の注意事項

インストール後はログインし直してください。

再インストールする場合は、[コントロールパネル] の [プログラムのアンインストール] からインストール済みのRPDP/S10VEをアンインストールしてから、インストールを実行してください。

3. 3. 2 SHCコンパイラインストール時の注意事項

SHCコンパイラをインストールするときは、SHCコンパイラ (Ver.9.04 Release 00) のインストール先は、デフォルトのインストール先 (C:\Program Files (x86)) としてください。

3.4 RPDP実行環境

(1) 実行環境設定ファイル

RPDPを使用するためには環境の設定が必要です。S10VEのRPDP実行環境は設定ファイル

(%SystemRoot%\¥renix¥usr¥rpdp_hce¥etc¥RPDP.ini) で設定します。RPDP/S10VEをインストールするとデフォルトの設定ファイルが作成されます。設定値をデフォルトの値と異なる値で運用する場合は、RPDPインストール後に設定ファイルの内容を変更してください。

表 1-12 S10VE RPDP実行環境の設定値一覧

No.	設定項目	設定内容	デフォルト値	備考
1	SHCPU	CPU種別を設定します。 “SH4”を設定してください。	SH4	
2	SHC_INC	コンパイラのインクルードファイル格納ディレクトリを設定します。	%ProgramFiles(x86)%¥Renesas¥Hew¥Tools¥Renesas¥Sh¥9_4_0¥include	
3	SHC_LIB	コンパイラのインストールディレクトリを設定します。	%ProgramFiles(x86)%¥Renesas¥Hew¥Tools¥Renesas¥Sh¥9_4_0¥bin	
4	SHC_TMP	コンパイラがテンポラリファイルを作成するディレクトリを設定します。	%SystemRoot%\¥renix¥tmp	
5	HLNK_DIR	S10VE用RPDPのsvloadコマンドのライブラリのサーチパスを設定します。	%SystemRoot%\¥renix¥S10VE¥lib	
6	HLNK_TMP	リンカージェディタがテンポラリファイルを作成するディレクトリを設定します。	%SystemRoot%\¥renix¥tmp	

(2) 環境変数PATHの設定

環境変数のPATHに、RPDPのコマンドのパスを設定します。RPDP/S10VEをインストールするとデフォルトのパス「%SystemRoot%\¥renix¥S10VE¥bin;%ProgramFiles(x86)%¥Renesas¥Hew¥Tools¥Renesas¥Sh¥9_4_0¥bin」が、PATHの先頭に設定されます。

3. 5 RPDP使用者アカウントの登録

RPDPを使用するためには、専用のグループ（RPDPusersグループ）に所属したアカウントでログインしなければなりません。新規にRPDPusersに所属するアカウントを作成することも、既存のアカウントをRPDPusersに所属させることもできます。

3. 5. 1 新規アカウントの登録

- (1) Administratorでログインします。
- (2) [コントロールパネル] の [管理ツール] から [コンピュータの管理] を起動します。
- (3) コンソールツリー（左のペイン）の [システム ツール] - [ローカル ユーザーとグループ] - [ユーザ] をダブルクリックし、[ユーザーの一覧] を表示させます。
- (4) 以下の手順で、専用のアカウントを登録します。
 - ① [操作(A)] メニューの [新しいユーザー(N)...] を選択し、[新しいユーザー] ダイアログボックスを表示します。
 - ② [新しいユーザー] ダイアログボックスが表示されたら、ユーザ名などの必要な項目を入力し新しいアカウントを登録してください。ユーザ名やパスワードは任意です。
 - ③ 作成したユーザをダブルクリックし、[ユーザのプロパティ] ダイアログボックスを表示させます。
 - ④ [所属するグループ] タブを選択し、[追加(D)] ボタンをクリックして、[グループの選択] ダイアログボックスを表示させます。
 - ⑤ [詳細設定(A)] ボタンをクリックし、次に [検索(N)] ボタンをクリックすると、グループの一覧が表示されますので、[名前(RDN)] 欄の「RPDPusers」をクリックして選択し、[OK] ボタンをクリックして追加します。

「RPDPusers」グループはRPDP/S10VEインストール時に自動的に登録されています。
 - ⑥ [OK] ボタンをクリックし、[グループの選択] ダイアログボックスを終了します。
 - ⑦ [OK] ボタンをクリックし、[ユーザのプロパティ] ダイアログボックスを終了します。

3. 5. 2 既存のアカウントの所属するグループにRPDPusersを追加

- (1) Administratorでログインします。
- (2) [コントロールパネル] の [管理ツール] から [コンピュータの管理] を起動します。
- (3) コンソールツリー (左のペイン) の [システム ツール] - [ローカル ユーザーとグループ] - [ユーザ] をダブルクリックし、[ユーザーの一覧] を表示させます。
- (4) 以下の手順で、アカウントの所属するグループに、「RPDPusers」を追加します。
 - ① 「RPDPusers」グループに所属させたいユーザをダブルクリックし、[ユーザのプロパティ] ダイアログボックスを表示させます。
 - ② [所属するグループ] タブを選択し、[追加(D)] ボタンをクリックして、[グループの選択] ダイアログボックスを表示させます。
 - ③ [詳細設定(A)] ボタンをクリックし、次に [検索(N)] ボタンをクリックするとグループの一覧が表示されますので、[名前(RDN)] 欄の「RPDPusers」をクリックして選択し、[OK] ボタンをクリックして追加します。
 「RPDPusers」グループはRPDP/SIOVEインストール時に自動的に登録されています。
 - ④ [OK] ボタンをクリックし、[グループの選択] ダイアログボックスを終了します。
 - ⑤ [OK] ボタンをクリックし、[ユーザのプロパティ] ダイアログボックスを終了します。

第4章 コンパイラ

この章では、S10VE用Cコンパイラおよびアセンブラの詳細について説明します。コマンドリファレンスについては、第2編を参照してください。

このRPDPではコンパイラおよびアセンブラを使用する場合には、「ルネサスマイクロコンピュータ開発環境システムSuperH RISC engine C/C++コンパイラパッケージVer.9.04 Release 00」（以下、shcコンパイラと表記します）を使用することを前提としています。

4. 1 Cコンパイラオプション詳細

以下にshcを使用したコンパイルの方法と、コンパイル時の注意点を示します。

shcの仕様の詳細は、shcコンパイラ付属のマニュアルを参照してください。

- コマンド形式

```
shc [ $\Delta$ <オプション>...][ $\Delta$ <ファイル名>[ $\Delta$ <オプション>...]...]
```

(例) shc Δ test1.c Δ test2.c [Enter]

- RPDP実行環境の設定

実行環境の設定方法の詳細は、「3. 4 RPDP実行環境」を参照してください。

- shc実行環境の設定

shcコンパイラを直接使用するためには表1-13に示すshcコンパイラの動作に必要な環境変数の設定が必要です。

Ver.9.04のshcコンパイラを直接使用するためには環境変数の設定をVer.9.04用に設定する必要があります。

- CPMSとのインターフェイス利用時の設定

CPMSとユーザーのインターフェイスを利用するためには、付録Cに示すS10VE用インクルードファイル格納ディレクトリ（%windir%\¥renix¥s10ve¥include）を、インクルードファイルのサーチディレクトリに指定する必要があります。

サーチディレクトリの指定は、環境変数のSHC_INCやshcコンパイラオプションのincludeで指定する方法があります。

表 1-13 shcコンパイラの動作に必要な環境変数

No.	環境変数	設定内容
1	path	<p>環境変数pathにインストールしたコンパイラパッケージの実行ファイルの格納ディレクトリを追加してください。</p> <p>コンパイラ (shc) 、最適化リンカージェディタ (optlnk) のパスの設定が必要です。この環境変数の指定は必須です。</p> <p>指定フォーマット : path=<実行ファイルパス名>[;<既存パス名>;...]</p>
2	SHC_LIB	<p>コンパイラのロードモジュールおよびシステムインクルードファイルを格納したディレクトリを指定してください。</p> <p>この環境変数の指定は必須です。</p> <p>指定フォーマット : set SHC_LIB=<実行ファイルパス名></p>
3	SHCPU	<p>対象とするCPU種別を指定します。</p> <p>このシステムでは、SHCPU=SH4を指定してください。</p> <p>この指定を省略した場合、CPU種別はSH1として扱われるので注意してください。</p> <p>CPU種別は-cpuオプションでも指定することができます。</p> <p>指定フォーマット : set SHCPU=<CPU></p>
4	SHC_INC	<p>コンパイラのインクルードファイル格納ディレクトリを指定してください。システムインクルードファイルの検索順序は、includeオプション指定ディレクトリ、SHC_INC指定ディレクトリ、システムディレクトリ (SHC_LIB) となります。</p> <p>ユーザインクルードファイルの検索順序はカレントディレクトリ、includeオプション指定ディレクトリ、SHC_INC指定ディレクトリとなります。</p> <p>指定フォーマット :</p> <p>set SHC_INC=<インクルードパス名> [;<インクルードパス名>;...]</p>
5	HLNK_DIR	<p>最適化リンカージェディタの入力ファイル格納ディレクトリを指定します。</p> <p>Inputオプション、libraryオプションで指定したファイルの検索順序は、カレントディレクトリ、HLNK_DIR指定ディレクトリになります。</p> <p>ローダのライブラリサーチパスもHLNK_DIRの設定に従います。</p> <p>指定フォーマット :</p> <p>set HLNK_DIR=<入力ファイルパス名>[;<入力ファイルパス名>;...]</p>
6	SHC_TMP	<p>コンパイラがテンポラリファイルを作成するディレクトリを設定します。</p> <p>指定フォーマット :</p> <p>set SHC_TMP=<ディレクトリ></p>
7	HLNK_TMP	<p>リンカージェディタがテンポラリファイルを作成するディレクトリを設定します。</p> <p>指定フォーマット :</p> <p>set HLNK_TMP=<ディレクトリ></p>

4. 2 コンパイル時の注意点

4. 2. 1 shcを使用してコンパイルする場合

● 浮動小数点数の扱い

shcでは浮動小数点数の非正規化数と丸めの扱いをコンパイルオプションで制御することができます。

ただし、それぞれの扱いによってロード時にリンクする標準ライブラリが異なりますので注意してください。以下に非正規化数の扱いと丸め方を制御するオプションと対応する標準ライブラリの対応を示します（ロード時にライブラリを指定しないとローダはlibsh4nbmdn.libをリンクします）。

表 1-14 浮動小数点数の扱い制御オプション

	仕様	オプション (*2)	デフォルト	cchrでの扱い
非正規化数の扱い	0として扱う	-denormalization=off	0として扱う	非正規化数として扱う
	非正規化数として扱う (*1)	-denormalization=on		
結果の値の丸め方	有効数字を超える部分を切り捨てる	-round=zero	切り捨て	四捨五入
	有効数字を超える部分を四捨五入する	-round=nearest		

(*1) S10VEのCPUであるSH4A (SH7786) は、非正規化数を非正規化数として扱うモードで動作すると、非正規化数入力時にFPUエラーが発生するため、実行時には0として扱うモードで実行されます。

(*2) 浮動小数点数の扱いをcchrと同様とするためには、-denormalization=on、-round=nearestを指定してください。

表 1-15 浮動小数点数の扱いと対応する標準ライブラリ

	-denormalization	-round	標準ライブラリ
指定オプション	off	zero	libsh4nbmzz.lib
	on	zero	—
	off	nearest	—
	on	nearest	libsh4nbmdn.lib

- 標準ライブラリは、shcのデフォルト-denormalization=off、-round=zero用およびcchr互換の-denormalization=on、-round=nearest用の2つを用意します。

- コンパイルリストの生成と保存 (shc)

タスクの使用するスタックサイズの算出などで必要となるため、コンパイルリストを生成して保存しておいてください。コンパイルリストを生成するためには、以下に示すオプションを指定します。

-listfileオプションはコンパイルするCソースファイルよりも前に指定してください。

Cソースファイルよりも後ろに指定した場合は、最後の1ファイルだけコンパイルリストが生成されます。

- コンパイルリスト生成指定

```
-listfile [= <リストファイル名>] -show=source, object
```

リストファイル名の指定を省略した場合、ソースファイル名と同じファイル名に拡張子“lst”を付加したファイルを生成します。

(例)

```
◆ shc Δ-listfile Δtest1.c Δtest2.c [Enter]
```

「test1.c」、「test2.c」ともlistfileオプションが有効となります。

```
◆ shc Δtest1.c Δtest2.c Δ-listfile [Enter]
```

listfileオプションは「test2.c」だけに対して有効になります。

- 組み込みサブルーチンコンパイル時の-fpscr=safe指定

shcコンパイラには関数呼び出しの前後でFPSCRレジスタの精度モードを保証するかどうかの指定 (-fpscr) があります。この指定は、デフォルトでは関数呼び出しの前後で精度を保証しない指定 (-fpscr=aggressive) になっています。このため、関数呼び出しから戻ったあとにFPSCRの精度モードを単精度に戻すコードが生成されます。

S10VEでは、組み込みサブルーチンにおいて浮動小数点演算は実行できません（実行時にはFPU Unavailable例外が発生し、CPUはSTOPします）。

組み込みサブルーチンを-fpscr=aggressive指定でコンパイルすると、浮動小数点演算を行っていないくても、関数呼び出しを行うとFPSCRのアクセスが発生し、FPU Unavailable例外が発生します。

したがって、組み込みサブルーチン（組み込みサブルーチンから呼び出されるIRSUBを含む）をコンパイルするときは、-fpscr=safeオプションを指定してコンパイルしてください。

4. 3 shcのバージョン比較

4. 3. 1 コマンド行オプション

表1-16にshcのコマンド行オプションの比較を示します。

表1-16 shcのコマンド行オプションのバージョン比較

shc	Version		意味
	V7	V9	
<u>-code</u> = <i>machinecode</i>	○	○	リンクをしない。オブジェクトモジュールを生成する。
<u>-define</u> =name <u>-define</u> =name=def	○	○	nameを定義する。 nameをdefに定義する。
<u>-debug</u>	○	○	デバッグ情報を生成する。
<u>-listfile</u> <u>-show</u> =source,object で代用可	○	○	アセンブラソースにソースファイルの行を入れる。
デフォルトでANSI準拠	○	○	ANSI Cに適合したプログラムだけをコンパイルする。
<u>-endian</u> =big	○	○	big-endianモードでコンパイルする (デフォルトはbig)。
<u>-endian</u> =little	○	○	little-endianモードでコンパイルする。
<u>-sjis</u> (デフォルト)	○	○	漢字 (シフトJIS) をサポートする。K%R仕様時だけ指定可。
<u>-show</u> =length=n	○	○	ソースリストの1ページの行数を指定する。
<u>-listfile</u> <u>-listfile</u> =filename	○	○	ソースリストを表示する。 ただし、cchrとshcのリストの内容は異なる。
<u>-include</u> =dir	○	○	インクルードファイルのサーチディレクトリを追加する。
<u>-optimize</u> =0 <u>-optimize</u> =1	○	●	最適化レベルを設定する。 <shc V7/V9> optimize=0 : 最適化なし、optimize=1 : 最適化あり。 -speed、-nospeed、-sizeで最適化の方法選択可。
<u>-speed</u> <u>-nospeed</u> <u>-size</u>	○	○	<shc V9> optimize=Debug_only: 文単位の削除に関する最適化も完全に抑止し、ローカル変数の情報を常に参照できるようになります。
<u>-preprocessor</u> [=file]	○	○	<shc> プリプロセッサだけを実行し、結果を.pファイルに格納する。
<u>-code</u> =asmcode	○	○	アセンブラソースを生成する。 アセンブラ、リンカを起動しない。

shcのオプションの表記で、下線部 () は短縮形指定時の文字を示します。また、斜体字は省略時解釈を示します。

<p>V7、V9凡例 ○ : 対応するオプションあり ● : V9で変更あり</p>
--

4. 4 データジェネレータ

データジェネレータを使用するとshcコンパイラがインストールされていない環境で、GLB、CMエリアに初期値をローディングすることができます。

コンパイラを使用しないでGLB、CMエリアに初期値をロードするための手順を以下に示します。

- ① ロードするエディションデータを記述したテキストファイルを生成します。
 - ② データジェネレータコマンド (svdatagen) を使用し、テキストファイルをバイナリデータに変換します。
 - ③ ローダ (svload) を使用して初期値をバックアップファイルにロードします。
- データを記述するテキストファイルの仕様については、(a) svdatagenの入力仕様、(b) C言語の宣言文との相違点と制限事項、(c) 記述可能な入力ファイルの例、(d) 記述不可入力ファイルの例、(e) プリプロセッサ機能の制限事項、(f) 初期値の型変換の仕様を参照してください。

データジェネレータで生成したバイナリデータ (*.bin) をバックアップファイルにロードするための、ローダ (svload) のオペレーションを以下に示します。

svload +B xx.bin

+B : svdatagenで生成した初期値データをロードします。

xx.bin : svdatagenで生成したバイナリファイルを指定します。

データジェネレータで生成したバイナリデータ (*.bin) をバックアップファイルにロードされているGLBデータとコンペアするための、svcompのオペレーションを以下に示します。

svcomp +B xx.bin

+B : svdatagenで生成した初期値データをGLB初期値とコンペアします。

xx.bin : svdatagenで生成したバイナリファイルを指定します。

(a) svdatagenの入力仕様

<入力ファイルの仕様>

入力ファイルの仕様はC言語の宣言文のサブセットです。

入力ファイルに記述可能な構文を以下に示します。

#include、#defineプリプロセッサ指示語も記述できます。プリプロセッサ指示語の仕様、制限事項は「(e) プリプロセッサ機能の制限事項」を参照してください。

宣言 :

```
extern 型指定子 配列[] ;  
型指定子 宣言子 = 初期値 ;  
構造体指定子 ;
```

型指定子 :

```
void  
char  
short  
int  
long  
float  
double  
signed  
unsigned  
構造体指定子
```

構造体指定子 :

```
struct { 構造体メンバ宣言リスト }  
struct tag { 構造体メンバ宣言リスト }  
struct tag
```

構造体メンバ宣言リスト :

```
型指定子 宣言子 ;
```

宣言子 :

```
変数  
配列[サイズ]  
*変数  
*配列[サイズ]
```

初期値 :

```
代入式  
{ 初期値リスト }  
{ 初期値リスト , }
```

初期値リスト :

```
初期値  
初期値リスト , 初期値
```

代入式 :

```
定数
```

定数 :

```
整定数  
文字定数  
浮動小数点定数  
GLB名称  
VAL名称
```

(b) C言語の宣言文との相違点と制限事項

C言語の宣言文との相違点と制限事項を以下に示します。

- (1) プリプロセッサの機能は`#define`、`#include`だけが制限付きで使用できます。
制限の内容は「(e) プリプロセッサ機能の制限事項」を参照してください。
- (2) 入力ファイルには宣言文だけが記述できます。
- (3) 記憶クラス指定子は`extern`だけが記述できます。
(「(d) 記述不可入力ファイルの例」例2参照)
- (4) 型指定子に`union`、`enum`、`typedef`による型名は使用できません。
(「(d) 記述不可入力ファイルの例」例3、4参照)
- (5) ビットフィールドは使用できません。
(「(d) 記述不可入力ファイルの例」例5参照)
- (6) 型修飾子 (`const`、`volatile`) は使用できません。
(「(d) 記述不可入力ファイルの例」例6参照)
- (7) 宣言は変数、配列、ポインタ、ポインタ配列だけが記述できます。
関数へのポインタ、配列へのポインタなどは記述できません。
(「(d) 記述不可入力ファイルの例」例7参照)
- (8) 配列のサイズは省略できません。必要なサイズを宣言してください。
(「(d) 記述不可入力ファイルの例」例8参照)
- (9) 初期値には定数と`GLB/VAL`の外部名だけが記述できます。式は記述できません。
ただし、単項演算子の+ (プラス) および- (マイナス) と定数 (10進数、浮動小数点定数) からなる式は記述できます。
(「(d) 記述不可入力ファイルの例」例9参照)
- (10) + (プラス) および- (マイナス) 符号の後ろに8進数、16進数、文字定数は記述できません。
(「(d) 記述不可入力ファイルの例」例10参照)
- (11) 定数に列挙型定数、文字列定数は記述できません。
文字列は文字定数で1文字ずつ初期化してください。
(「(d) 記述不可入力ファイルの例」例8参照)
- (12) 文字定数に2バイト以上のコード (漢字、`abcd`など) は記述できません。
(「(d) 記述不可入力ファイルの例」例11参照)
- (13) 文字定数に`¥ooo`、`¥xhh`の形式のエスケープ列は記述できません。ただし、`¥0`を除きます。
(「(d) 記述不可入力ファイルの例」例12参照)
- (14) 幅広文字定数 (L 'x' など) は使用できません。
(「(d) 記述不可入力ファイルの例」例13参照)
- (15) 整定数の接尾子 (`u`、`U`、`l`、`L`、`ul`、`UL`) は使用できません。定数の型は宣言子の型に合わせてます。
(「(d) 記述不可入力ファイルの例」例14参照)

- (16) 浮動小数点定数の接尾子 (f、F、l、L) は使用できません。定数の型は宣言子の型に合わせて
ます。

(「(d) 記述不可入力ファイルの例」例14参照)

- (17) 合成体 (構造体、配列) のメンバに合成体を含む場合、部分合成体 (メンバの合成体) の初期
値を {} で囲んで指定することはできません。つまり、初期値リスト {} の中に更に {} を書く
ことはできません。

配列、構造体の初期値は必要な数だけすべて指定してください。ただし、初期値が配列、構
造体の要素数より少ない場合、残りは0で初期化します。

(「(d) 記述不可入力ファイルの例」例15、16参照)

- (18) コメントは /* */ で囲んでください。コメントは複数行にまたがっていても構いません。
// の形式のコメントは使用できません。

(「(d) 記述不可入力ファイルの例」例17参照)

- (19) 入力ファイルに誤り、未サポート構文を検出した場合はその時点で処理を打ち切ります。
以降の入力の解釈は行いません。

- (20) 宣言の型と初期値の記述形式が合わない場合の型変換はコンパイラと異なります。
詳細は「(f) 初期値の型変換の仕様」を参照してください。

(c) 記述可能な入力ファイルの例

以下に記述可能な入力ファイルの例を示します。

```

/* インクルードファイル */
#include "defines.h"
/* GLB/VALの外部名宣言 */
extern int tbl1_g[] ;
extern int tbl20_g[] ;
extern int tbl21_g[] ;
extern int tbl22_g[] ;
extern int tbl23_g[] ;
extern int vall_v[] ;

/* 変数 */
int int_var_g = 1 ;

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

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

/* ポインタ */
int *tbl1p_g = tbl1_g ;
int *vallp_g = vall_v ;

/* ポインタ配列 */
int *tbl2xp_g[4] = {tbl20_g, tbl21_g, tbl22_g, tbl23_g} ;

/* 文字配列の初期化 */
char str_g[12] = {'I', ' ', 'a', 'm', ' ', 's', 't', 'r', 'i', 'n', 'g', '\0'} ;

/* 部分合成体を含む合成体の初期化 */
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} ;

/* define値の参照 */
int defines[4] = {VAL1, VAL2, GLB1ADDR, GLB2ADDR} ;

```

defines.hの内容

```

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

```

図 1-17 defines.h

第4章 コンパイラ

(d) 記述不可入力ファイルの例

以下に記述できない入力ファイルの例を示します。

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

/* 例1 プリプロセッサ指示語(仕様範囲外) */
#define XMAX 10.0e100

/* 例2 static記憶クラス指定子 */
static int int_array[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} ;

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

/* 例4 typedef */
typedef int* tblp ;
tblp tbl1p_g = tbl1_g ;

/* 例5 ビットフィールド */
struct bit{
    int a:8 ;
    int b:8 ;
    int c:8 ;
    int d:8 ;
}bit_g = {1, 2, 3, 4} ;

/* 例6 型修飾子 */
const int ro_g = 1 ;

/* 例7 複雑な宣言 */
int *(tbl2xp_g[])[4] = {tbl20_g, tbl21_g, tbl22_g, tbl23_g} ;

/* 例8 配列のサイズ省略、文字列 */
char str_g[] = "I am string" ;
```

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

/* 例10 符号付き8進数、16進数、文字定数 */
int var5_g[3] = {+01, -0x1, -'x'} ;

/* 例11 2バイト以上のコード */
int a_g = 'abcd' ;
int b_g = '漢字' ;

/* 例12 エスケープ列 */
char bell_g = '\007' ;

/* 例13 幅広文字定数 */
wchar_t wc_g = L'x' ;

/* 例14 定数の接尾子 */
unsigned int ui_g = 1U ;
long l_g = 1L ;
unsigned long ul_g = 1UL ;
float f_g = 1.0F ;
double d_g = 1.0L ;

/* 例15 部分合成体の{}での初期化 */
struct Y {
    int a ;
    struct X {
        int a ;
        float b ;
        float c ;
    } x ;
} y_g = {-1, {+2, 3.0, 4e1}} ;

/* 例16 部分合成体の初期値省略 */
float f_g[4][3] = {{1}, {2}, {3}, {4}} ;

/* 例17 コメントの形式 */
//この形式のコメントは使用できません
```

(e) プリプロセッサ機能の制限事項

- (1) `#include`、`#define`だけが記述できます。
- (2) `#include`でインクルードされるファイルには、`#define`だけが記述できます。
`#include`をネストすることはできません。
- (3) `#include`は下記の形式で記述してください。
ファイル名は“”で囲んでください。<>で囲まれている場合はインクルードファイルとはみなしません。
“ファイル名”が相対パスである場合は、入力ファイルのあるディレクトリからの相対と解釈します。

`#include “ファイル名”`

- (4) `#define`は下記の形式だけをサポートします。単純な名前と整定数の置換だけが可能です。
浮動小数点定数、文字定数、定数以外のマクロは記述できません。
整定数には8進、10進、16進の定数が記述できます。
10進数は符号付きの値も記述できます。

`#define 名前 整定数`

- (5) `#define`で定義した名前の参照は初期値にだけ記述できます。初期値リスト以外に記述しても展開されません。
- (6) `#define`で定義した値は入力ファイル内全体で有効です。値の再定義はできません。

(f) 初期値の型変換の仕様

以下に変数の宣言の型と初期値の記述形式による型変換の可否、およびコンパイラ（shcコンパイラ）との相違点を示します。

表 1-17 初期値型変換仕様

		初期値の記述形式											
		8進数		10進数		16進数		浮動小数点数		文字定数		外部名	
		shc	svdata gen	shc	svdata gen	shc	svdata gen	shc	svdata gen	shc	svdata gen	shc	svdata gen
変数 の型	char	char	←	char	←	char	←	char	×	char	←	×	char
	short	short	←	short	←	short	←	short	×	short	←	×	short
	long	long	←	long	←	long	←	long	×	long	←	×	long
	float	float	×	float	←	float	×	float	←	float	×	×	←
	double	double	×	double	←	double	×	double	←	double	×	×	←
	ポインタ	×	addr	×	addr	×	addr	×	←	×	←	addr	←

char、short、long、float、double：それぞれの型に応じた初期値を生成します。

addr：アドレスとして4バイトの値を生成します。

×：初期値生成不可を表します。

□：shcコンパイラとsvdatagenコマンドの相違点を表します。

←：同左（shcと同じ）

第5章 プログラミングコマンド

5. 1 プログラミングコマンドについて

S10VEのプログラミングコマンドでは、ライブラリアン、リンカは、それぞれoptlnkとsvloadを用意します。makeコマンドとして、makehceをサポートします。

Optlnkの詳細はshcのマニュアル（SuperH RISC engine C/C++コンパイラ、アセンブラ、最適化リンケージエディタ ユーザーズマニュアル）の「第4章 最適化リンケージエディタ操作方法」を参照してください。

第6章 アロケータ

6.1 分割領域の確保と削除

6.1.1 領域の分割の必要性

リアルタイムプログラムの開発に先立って、あらかじめリアルタイムプログラムが使用するタスク、サブプログラム、GLBなどの共有資源用の格納領域を確保（アロケート）しておく必要があります。

リアルタイムシステムでは処理の高速性を追求するため、さまざまな資源に対するアクセスをその資源の格納アドレスを使って行います。このため、プログラム実行時資源の格納アドレスが不定であってはならないため、アロケータで計算機内のエリアをユーザの指定領域に分割し、これが指定アドレスからずれないように管理しています。システム設計時にはこのシステムの適用対象から、どれくらいのデータが必要かを調べてGLBの大きさ、配置を決めてください。リアルタイムシステムでは多くの場合、個々のプログラム作りよりも、このデータに関する設計が重要でシステムの総合性能に大きく影響します。

割り当ては、次の2段階で行います。

- (1) タスク、サブプログラム、GLB用に分割領域（AREA）を定義します。
- (2) GLBについてはsvdfsを使用してAREAを分割し、細分割領域（SAREA）を定義できます。

アロケータで定義した分割領域、細分割領域については、その名前と属性、位置、サイズなどが領域管理情報に記録され、リアルタイムプログラム中からは、ここで定義した名前を用いてGLBなどの共有資源を参照したり、呼び出したりできます。

このように共有資源用の領域を階層的に確保するのは、共有資源用の領域が通常の資源とは別に確保されていて、いったん確保した分割領域を再定義した際に、なるべく他の分割領域を定義し直さなくても良いように配慮しているためです。

6. 1. 2 分割領域の確保

分割領域はその用途によって決められた大分割領域（GAREA）内に配置されます。

表 1-18に分割領域の用途と配置するGAREAの対応を示します。

定義する分割領域の用途は、分割領域の確保時にsvdfaコマンドのオプションで指定します。

表 1-18 分割領域の用途と配置するGAREAの対応

分割領域の用途	配置するGAREA	svdfaのオプション
タスク（プログラム）	\$TASK	-p
CM用のデータ	\$CM	-cmi、-cmw
読み出し専用GLB用のデータ	\$GLBR	-gr
読み書き両用GLB用のデータ	\$GLBRW	-gi、-gw
サブプログラム	\$IRSUB	-s

分割領域を確保すると、指定サイズ分のエリアが大分割領域内に確保されます。ただし、確保した分割領域をダウンロードするまでは、SIOVEメモリには反映されません。

また、分割領域を確保すると、分割領域内に配置するリアルタイムリソースの初期値設定用のファイル（バックアップファイル）を生成します。

ただし、初期値なしGLB、CMの分割領域はバックアップファイルを生成しません。

バックアップファイルの内容は0で初期化されています。

分割領域の確保に用いるコマンドを次に示します。

- svdfa 分割領域（AREA）の確保
- svdfs 細分割領域（SAREA）の確保

分割領域のレイアウトを図1-18に示します。

glb_1 (4096バイト) (AREA)			glb_2 (4096バイト) (AREA)		
cond_1 (1024バイト) (SAREA)	act_1 (2048バイト) (SAREA)	空き	cond_2 (256バイト)	sub_2 (128バイト)	空き
(SAREA)					

図1-18 分割領域のレイアウト

上記のレイアウトの確保実行例

```
#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
```


図1-18に基づいて分割領域を確保します。

```

$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 **
$

```

リアルタイムプログラムの中では、定義した細分割領域の名前、`cond_1`、`act_1`、`cond_2`、`sub_2`を使用することによって、これらの共有資源を利用できます。

```
$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];
}
```

なお、一度確保した分割領域のサイズや属性を変更する場合、細分化して定義した細分割領域の合計より小さくならないように注意してください。

細分割領域の合計より小さい場合、細分割領域を指定分割領域内に確保できなくなります。

6. 1. 3 分割領域の削除

分割領域の削除に用いるコマンドを次に示します。

```
svdla      分割領域 (AREA) の削除
svdls      細分割領域 (SAREA) の削除
```

6. 1. 2項で確保したGLB部のglb_1およびglb_2を削除する例を示します。削除しようとする分割領域にさらに細分割領域があれば、その細分割領域も同時に解放されます。glb_2の場合、細分割領域のcond_2、sub_2を定義してありますが、glb_2を削除すると、cond_2、sub_2も自動的に削除されます。

```
$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 GLB、VALの名前の付け方

GLB、VALはシステム内でユニークな名前にしてください。

- 名前の文字数 (バイト数) は、最大14文字
- 先頭文字は英字またはアンダーライン (_)
- 2文字目以降は、英字、数字、アンダーラインの組み合わせ

ただし、これらの名前を利用するリアルタイムプログラムでは、使用するC言語に次のような名前の制約がありますので注意してください。

<C言語の制約>

- 名前の文字数 (バイト数) は、最大14文字
- 先頭文字は英字
- 2文字目以降は、英字、数字、アンダーラインの組み合わせ
- 名前のあとに次のようなサフィックスを付加します。サフィックスの文字数は、名前の文字数に含みません。

GLBのとき _g

VALのとき _v

- GLB、VALの宣言は、必ずextern指定による外部変数の宣言としてください。

6. 1. 5 CM用の分割領域の確保

CPMSでは同一ユニット内PU間の主メモリを共有する方法としてCMを提供しています。

これはアロケータを使ってあらかじめCPMSタスクの論理空間内のCM空間に領域を確保して名前を付けておき、その領域を同一ユニット内の複数のタスクやサブルーチンで共同に使用するものです。svdfaを用いて、この領域を areaに分割し、さらにsvdfsを用いてarea内をsareaに細分割して使用します。

CM空間はCPUのHP、CP間の連絡用メモリとして使用されます。

以下にCPMSの論理空間におけるCM領域の詳細を示します。

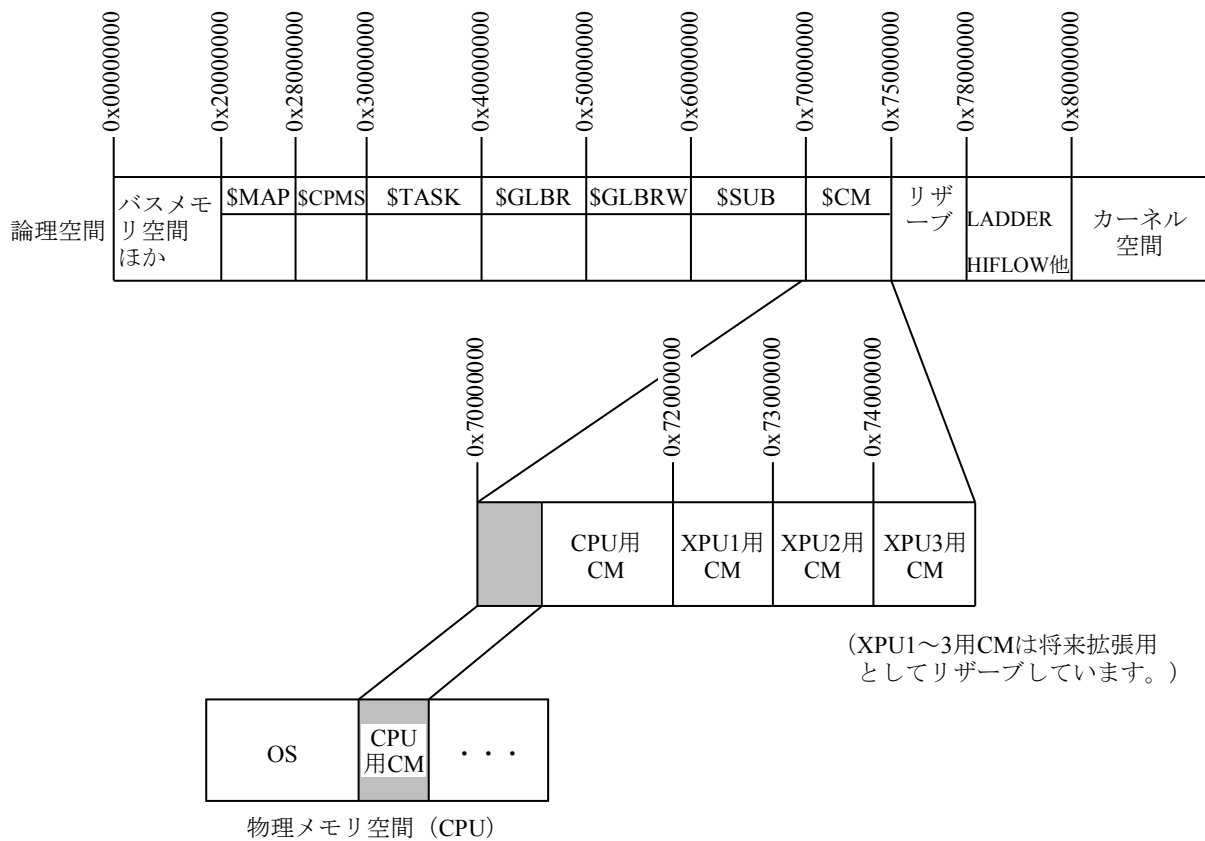


図 1 - 19 CPMSの論理空間上のCM空間とS10VE主メモリのCM空間との対応

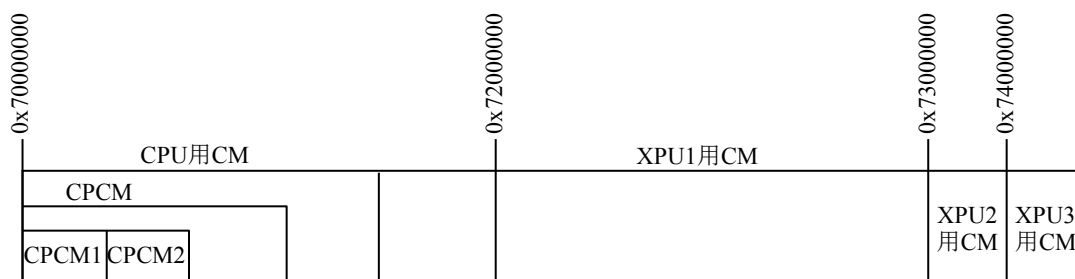
<CMの使用方法>

● CM領域の分割領域確保

RPDPでは各PU（HP、CP）に対応するサイトにおいて、サイトに対応するCM空間内に分割領域、細分割領域を定義した場合、その領域名は該当サイトにおいてのみ有効になります。そのため、CM領域内に定義した分割領域や細分割領域をHP用のサイトとCP用のサイトで共有する場合、HP用のサイトとCP用のサイトで同様に分割領域や細分割領域を定義する必要があります。

CM領域内に定義する分割領域をHP用のサイトとCP用のサイトで同一アドレスに配置するためには、svdfaコマンドで分割領域を定義するときに-fオプションを指定して、大分割領域\$CMの先頭からの相対アドレスがHP用のサイトとCP用のサイトで同じになるようにしてください。CM領域の分割領域の定義時に-fオプションを省略するとエラーとなります。CM領域に定義した分割領域に対応するバックアップファイルは、CP用のサイトに生成します。

CPとHP間でCMを共有する場合の例を以下に示します。



CP側エリア定義オペレーション

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

HP側エリア定義オペレーション

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

上記のオペレーション例のように、CP側とHP側ではお互いのエリアも含めて、同一名称、同一アドレスとなるようにエリアを定義してください。

● CM領域へのデータローディング

CM領域内の分割領域や細分割領域は、HP用のサイトとCP用のサイトで同一のアドレスに配置されるように定義する必要があります。このうち初期値データのローディングが可能である領域は、CPのサイトのみとなります。

6.2 バリュ (VAL) の登録、削除

アロケータは、バリュ (VAL) と呼ばれる全プログラム間共通の定数の登録、削除を行います。

VALの登録、削除はそれぞれsvdfv、svdlvコマンドで行います。

第7章 ローダ

7.1 リンク・ロードとは

shcコマンドを用いて作成したオブジェクトモジュール（.objファイル）を、svloadコマンドを用いてリンクし（1つのプログラムにまとめ）、GLB、IRSUBなどのアロケータ管理情報を用いてロードし、ロードモジュールをバックアップファイルに書き込みます（図1-20参照）。

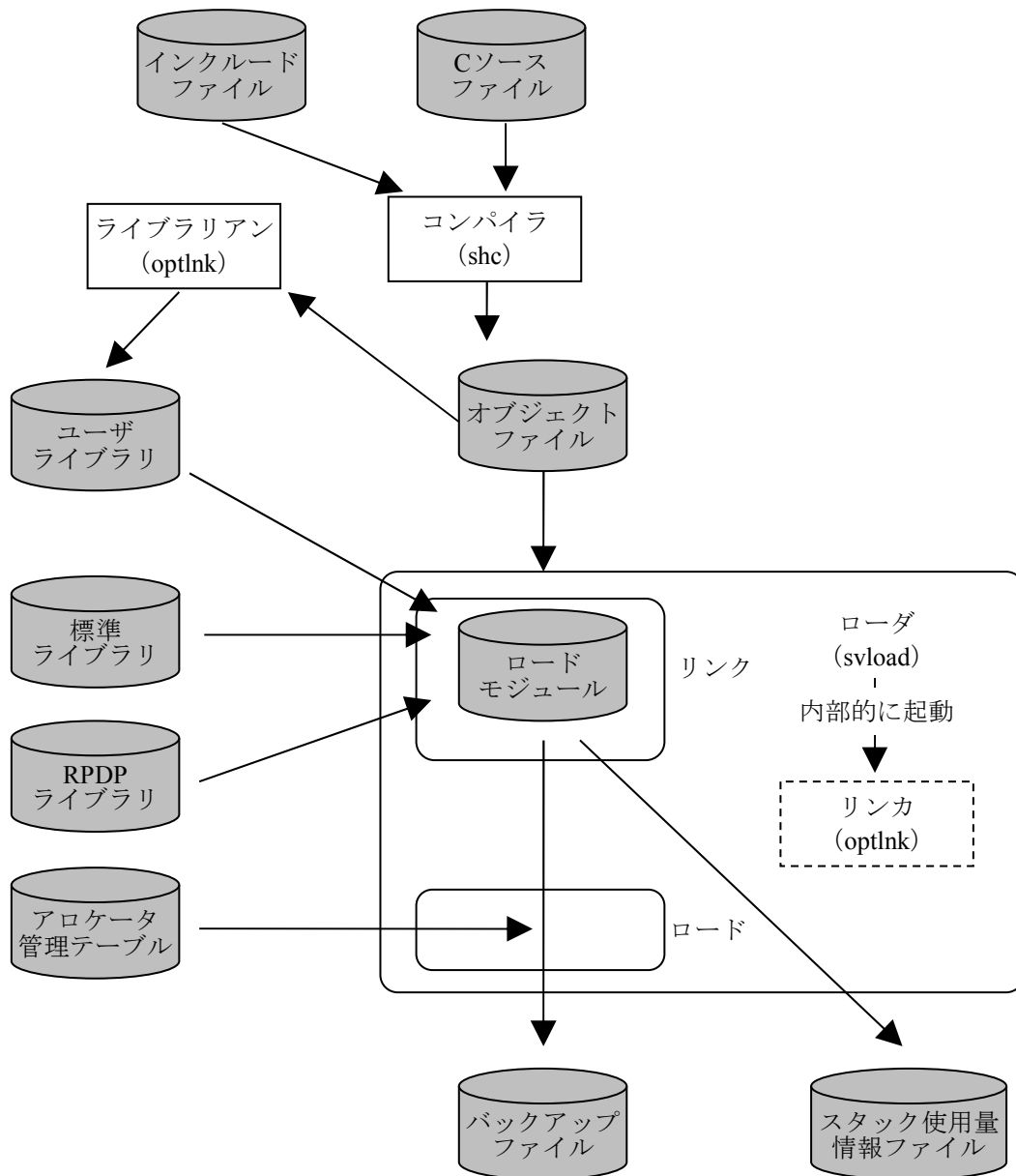


図1-20 ロードモジュール、バックアップファイルの作成

7. 2 ローダの動作環境

ローダに入力するプログラムは、ローダでの結合後のロードモジュールが、表1-19の条件を満たすようなプログラムとしてください。

表1-19 ロードモジュールの条件

オプション	ロードモジュール		
	TEXT	DATA	BSS
プログラム登録	>0	—	—
組み込みサブプログラムの登録	>0	—	— (*)
データの登録	—	>0	—

TEXT：実行できる部分 DATA：初期値ありデータ BSS：初期値なしエリア

—：サイズ=0、>0のどちらでも処理できます。

>0：サイズ>0以外はエラーとします。

(*) サブプログラムのBSS部は書き込み禁止です。

サブプログラムはBSSを持たないでください。

ローダが生成するロードモジュールの構成を図1-21に示します。

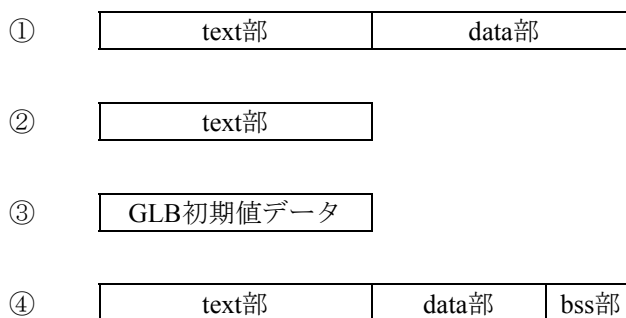


図1-21 ロードモジュールの構成

<図1-21の説明>

- ① text部とdata部を持つプログラムまたはサブプログラムのロードモジュールです。プログラムまたはサブプログラムとしてローディングできます。
- ② text部のみからなるプログラムまたはサブプログラムのロードモジュールです。①と同様にローディングします。
- ③ GLBの初期値設定プログラムのロードモジュールです。データとしてローディングできます。
- ④ text部、data部とbss部を持つプログラムのロードモジュールです。プログラムとしてローディングできます。サブプログラムには、bss部を持たせないでください。

(1) ローダの処理

図1-21に示したロードモジュールのうち③、④の構成を例に取り、ローダのローディング処理を説明します。

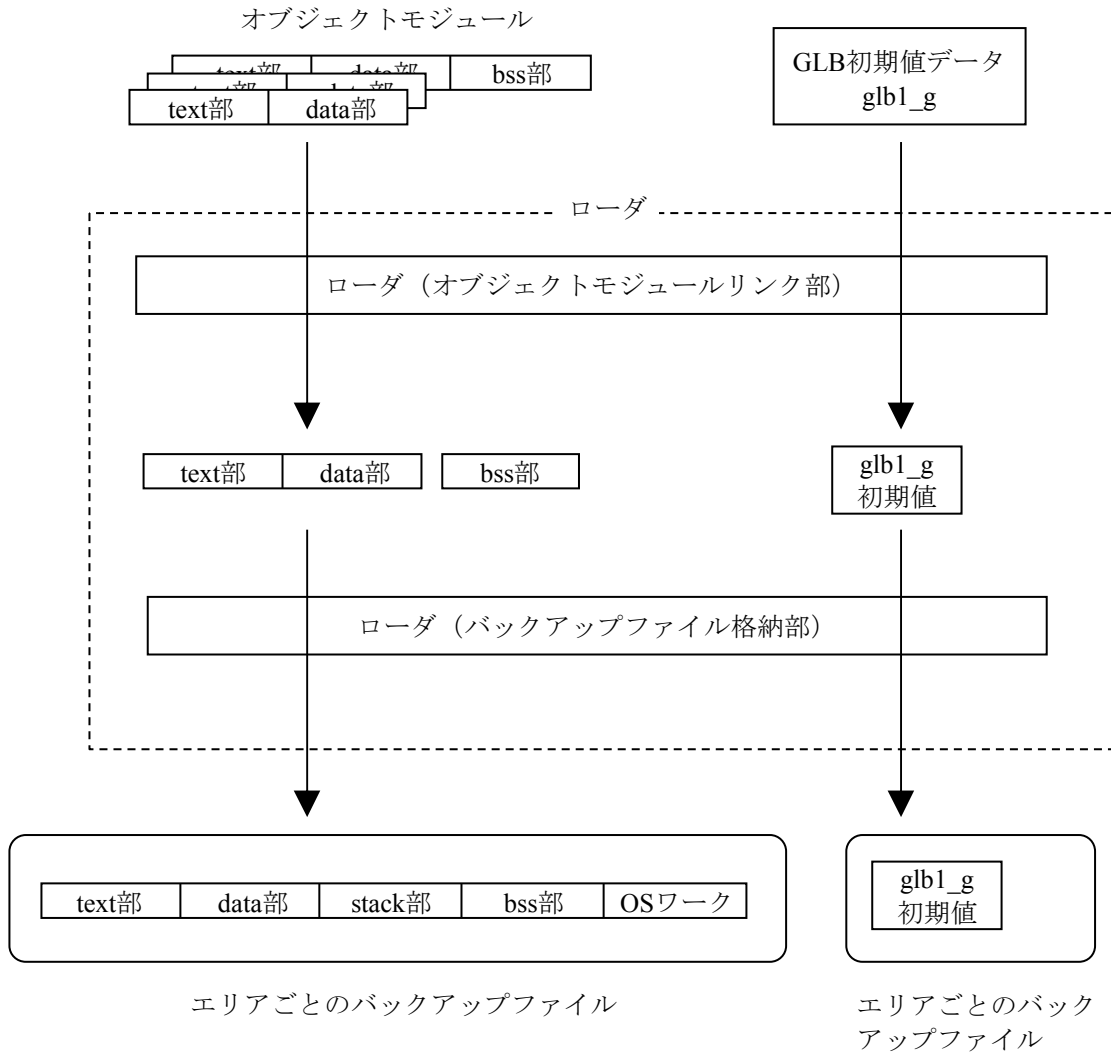


図1-22 ローディング処理

<図1-22の説明>

- ① グローバルの初期値データ部分はアロケータの管理するテーブルにsvdfsで登録した細分割領域に対応する箇所にもローディングされます。
- ② プログラムは実行モジュールとしてローダコマンドで指定した領域にローディングされます。ローダは実行モジュールに指定されたスタック領域のエリアとOSワークを付加してバックアップファイルに格納します。

(2) 名称のユニーク性

system/user間でプログラム名称、サブプログラム名称、組み込みサブルーチン名称、グローバル名称、バリュ名称はユニークな名称にしてください。

(3) system/user外部参照チェック

システムからユーザの参照はできません。

ユーザからシステムのサブプログラムのみ参照できます。表1-20に参照可能な組み合わせを示します。

表1-20 外部参照の組み合わせ

参照 \ 被参照		サブプログラム		グローバル		バリュ	
		S	U	S	U	S	U
プログラム	S	○	×	○	×	○	×
	U	○	○	×	○	×	○
サブプログラム	S	○	×	○	×	○	×
	U	○	○	×	○	×	○
グローバル	S	○ (*1)	×	○ (*2)	×	○ (*3)	×
	U	○	○ (*1)	×	○ (*2)	×	○ (*3)

○：参照可 S：システム ×：参照不可 U：ユーザ

(*1) グローバル内にIRSUB番号が埋め込まれます。

したがって、サブプログラムはbuildされているIRSUBでなければなりません。

(*2) グローバル内に被参照グローバルのアドレスが埋め込まれます。

(*3) グローバル内にVAL値が埋め込まれます。

7. 3 ライブラリのサーチパス

ローダのライブラリのサーチパス (-Iオプションで指定されたライブラリのサーチ順) は、shcコンパイラパッケージの最適化リンケージエディタの入力ファイルを検索する順序に従います。

最適化リンケージエディタの入力ファイルの検索順は下記となります。

(1) カレントディレクトリ

(2) RPD動作環境設定ファイルのHLNK_DIRで指定されたディレクトリ

RPDP動作環境設定ファイルのHLNK_DIRには複数のパスが設定できます。

複数のパスを設定する場合は、セミコロンで区切って指定してください。

7. 4 リンク・ロード時の注意事項

リアルタイムプログラムをリンク、ロードする際には、プログラム中で使用している、GLB、CM、VALの割り当てを済ませておいてください。また、IRSUBを使用しているときにはIRSUBのビルドまで済ませておいてください。

第8章 ビルダ

8. 1 タスクの登録と削除

8. 1. 1 タスクとは

ローダ (svload) で作成したロードモジュールは、ビルダのsvctaskによってタスクとして生成し、稼働の準備を整えます。svctaskは、ロードモジュールに対応するタスク名、タスク番号、タスクの実行レベルなどを作り、OSが管理しているタスク情報に登録します。

8. 1. 2 タスクの登録

プログラムをタスクとして登録する例を以下に示します。

```
$ svdfa areal 0x3000 -p
$ shc sample.c
$ svload +P -w 1024 1024 -a areal -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 **
$
```

svctaskで指定しているsampleは、タスクの資源となるロードモジュールの名前です。タスクの起動時には、ここで指定したロードモジュールがタスクのメインプログラムとして実行されます。

このタスクは、タスクの優先レベルを25に設定しています。また、タスク番号には10を設定しています。

タスクの生成後、タスクの識別にはタスク番号またはタスク名を用います。

8. 1. 3 タスクの削除

タスクを削除するには、`svdtask`を使用します。`svdtask`には、削除すべきタスクの名前を指定します。タスクを削除する例を以下に示します。

```
$ 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 常駐サブプログラムの登録と削除

8. 2. 1 間接リンクサブプログラム (IRSUB) とは

タスク登録後に、間接リンクサブプログラムを入れ替えてもタスクを再登録する必要のないサブプログラムのことです。

間接リンクサブプログラムは、プログラム内サブルーチンと同様に作成します。プログラム内サブルーチンと異なるのは、複数のタスクから共有されるため、リエントラントでなくてはならないことです。したがって、間接リンクサブプログラム内に静的変数を宣言して使用しないでください。

タスク登録後に間接リンクサブプログラムを再登録しても、登録番号は変わらず、対応するアドレスだけが変更されるため、タスクを再登録する必要はありません。

8. 2. 2 間接リンクサブプログラム (IRSUB) の登録

ローダ (svload) で実行可能モジュールを作成したあと、ビルダのsvbuildで登録します。

以下にサブプログラムを間接リンクサブプログラム (IRSUB) として登録する例を示します。

間接リンクサブプログラムを作成します。notepadなどのエディタを用います。

```
$notepad sub_a.c
```

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

間接リンクサブプログラムをコンパイルしてください。

```
$shc sub_a.c
```

間接リンクサブプログラムの登録番号を10として分割領域arealに登録します。

```
$ 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 = 000lcp

< 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)

** map output end **
```

8. 2. 3 間接リンクサブプログラム (IRSUB) の削除

登録済みの間接リンクサブプログラム (IRSUB) を削除するには、svdbuildを使用します。svdbuildには、削除すべきサブルーチンの名前と-irオプションを指定します。

以下に間接リンクサブプログラムを削除する例を示します。

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

site name = 000lcp

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

** map output end **
$
```

8. 3 組み込みサブルーチンの登録と削除

8. 3. 1 組み込みサブルーチンとは

組み込みサブルーチンについて説明します。組み込みサブルーチンでは、ハードウェアが検出した例外やソフトウェアで検出したイベントなどが発生したときに、OSで標準的に対処する異常処理の代わりにユーザのオリジナル処理をシステムに組み込むことができます。

組み込みサブルーチンは、svloadでロードしたものをビルダ (svbuild) によってOSのイベント処理プログラムの一部として組み込みます。

このシステムでは、いくつかの組み込み箇所 (エントリポイント) が用意してあり、それぞれがさまざまなイベントに対応しています。したがって、サブルーチンの組み込み時に、どのイベントに対応する処理かによって組み込み箇所を選びます。

各組み込み箇所はポイント名称で識別し、サブルーチンの組み込み時に、このポイント名称を指定します。

1つの組み込み箇所には4個のサブルーチンが登録できます。実際にイベントが発生すると、登録された組み込みサブルーチンがエントリ番号の小さい方から順々に呼び出され実行されます。エントリ番号とは、登録した組み込みサブルーチンの実行順を指定するものです。ただし、エントリ番号の1と2はOSとNXACP用に予約されています。ユーザプログラムはエントリ番号3と4を使用してください。

組み込みサブルーチン内でCPMSが定義する構造体を使用する場合は、コンパイル時にCPMSのインクルードファイルを指定してください。詳しくは、「4. 1 Cコンパイラオプション詳細」の「● CPMSとのインターフェイス利用時の設定」を参照してください。

8. 3. 2 組み込みサブルーチンの登録

プログラムを組み込みサブルーチンとして登録する例を以下に示します。例にあげたプログラムuabs_usrは、タスクがabortされたときに実行させるサブルーチンです。そこで、このサブルーチンをポイント名称ABSの組み込み箇所に登録します。なお、組み込みサブルーチンのリンク、ロードには、ローダ (svload) を使用し、かつオプション+Uを指定します。

```

$ shc uabs_usr.c

組み込みサブルーチンに登録します。

$ svdfa areal 4096 -s
$ svload +U -a areal -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 組み込みサブルーチンの削除

登録済みの組み込みサブルーチンを削除するには、svdbuildを使用します。svdbuildには、削除すべきサブルーチンの名前、組み込みポイント名称と-ulオプションを指定します。

登録してある組み込みサブルーチンを削除する例を以下に示します。

```

$ 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 **
$

```

第9章 マップ

9.1 アロケータ管理テーブル情報表示の目的

アロケータ管理テーブル情報表示の目的は、リアルタイムプログラムの開発を円滑に進めることです。

- タスク、サブプログラム、GLBなどの共有資源用の格納領域情報を表示し、個々のプログラム作成はもとより、システム設計の開発を支援します。
- S10VEにロードしたシステムのアロケータ管理テーブル情報を表示することで、デバッグ作業を支援します。

9. 2 svmapコマンドのオプション指定と表示情報

svmapコマンドのオプション指定ごとの表示フォーマットは、「付録F マップの表示フォーマット」に示します。表示フォーマット中の下線 () 部分は、svmapコマンドによる表示データです。

9. 2. 1 マップ情報の出力対象

マップ情報は、以下に示す対象の情報を出力します。

- (1) 開発系マシン側で管理するリソースのマップ情報
- (2) S10VEにダウンロードしたリソースのマップ情報

9. 2. 2 マップ情報の出力内容

マップ情報は、以下に示す情報を出力します。

- (1) ヘッダーとフッター
- (2) 大分割領域情報
- (3) 分割領域情報
- (4) 細分割領域情報
- (5) プログラム情報
- (6) サブプログラム情報
- (7) タスク情報
- (8) グローバル情報
- (9) VAL情報
- (10) IRSUBエントリ情報
- (11) IRGLBエントリ情報
- (12) ULSUBエントリ情報
- (13) 物理メモリの空き情報

9. 2. 3 マップ情報の出力形式

マップ情報は、以下に示す形式で出力することができます。

- (1) 階層マップ出力
- (2) アドレス順リスト出力
- (3) 名称順リスト出力
- (4) 番号順リスト出力
- (5) 名称指定出力

階層マップ出力は、指定大分割領域、分割領域単位に、論理空間上に配置されるリソースのマップ情報を階層的に出力します。

リスト出力は、指定情報を、アドレス順、名称順、番号順に並べて出力します。

また、リソースの名称を指定し、その名称単独の情報を出力することもできます。

表 1-21に出力内容と指定可能出力形式の組み合わせを示します。

表 1-21 出力内容と指定可能出力形式の組み合わせ

出力内容 \ 出力形式	階層出力	アドレス順出力	名称順出力	番号順出力	名称指定
大分割領域情報	○	○	×	×	○
分割領域情報	○	○	○	×	○
細分割領域情報	○	○	○	×	○
プログラム情報	×	×	○	×	○
サブプログラム情報	×	×	○	○	○
タスク情報	×	×	○	○	○
グローバル情報	×	×	○	○	○
VAL情報	×	×	○	×	○
IRSUBエントリ情報	×	×	○	○	○
IRGLBエントリ情報	×	×	○	○	○
ULSUBエントリ情報	×	×	○	○	○

○：指定可 ×：指定不可

9.3 svadmコマンドの論理アドレス指定と表示情報

svadmコマンドは、指定論理アドレスに対して名称などの情報を、コマンド、会話形式で表示し、デバッグ作業を支援します。

(1) コマンド形式

svadm 論理アドレス

パラメータに論理アドレスを指定した場合、以下のフォーマットで名称などの情報を表示します。

“XXX” は、svadmコマンドによる表示データを示します。

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

(2) 会話形式

パラメータに論理アドレスを指定しなかった場合、会話形式で論理アドレスを取り込み、コマンド形式と同一フォーマットで名称などの情報を表示します。

```
—会話形式の論理アドレス取り込み—
#svadm [Enter]

++ address information display start --> site(XXXXX) ++
addr : addr [Enter]



情報表示



addr : q [Enter]
++ address information display end ++
#
```

第10章 立ち上げ/PU制御

10. 1 概要

RPDPを使用したS10VEの立ち上げは、開発系マシン内のS10VE主メモリ（SDRAM）の初期データファイル（バックアップファイル）をsvrplコマンドでS10VE主メモリへダウンロードすることにより行います。

下図に示すように、OSとアロケータにより生成したバックアップファイルをS10VEの指定サイト（PU）の主メモリへダウンロードし、指定サイト（PU）を立ち上げます。

ただし、BASE SYSTEM/S10VEの「CPMSダウンロード」で一度もOSをダウンロードしたことのないS10VEをsvrplで立ち上げることはできません。最初はBASE SYSTEM/S10VEの「CPMSダウンロード」でOSをローディングしてください。

S10VEのPU制御は、立ち上げ後の指定サイト（PU）の状態を制御します。

サイト名称に、CPU名称（CPのサイト名称と同一名称）を指定した場合、CPUを処理対象とし、CP、HPの2つのサイトに対して処理を行います。サイト名称に、HPのサイト名称を指定した場合は、エラーとなります。

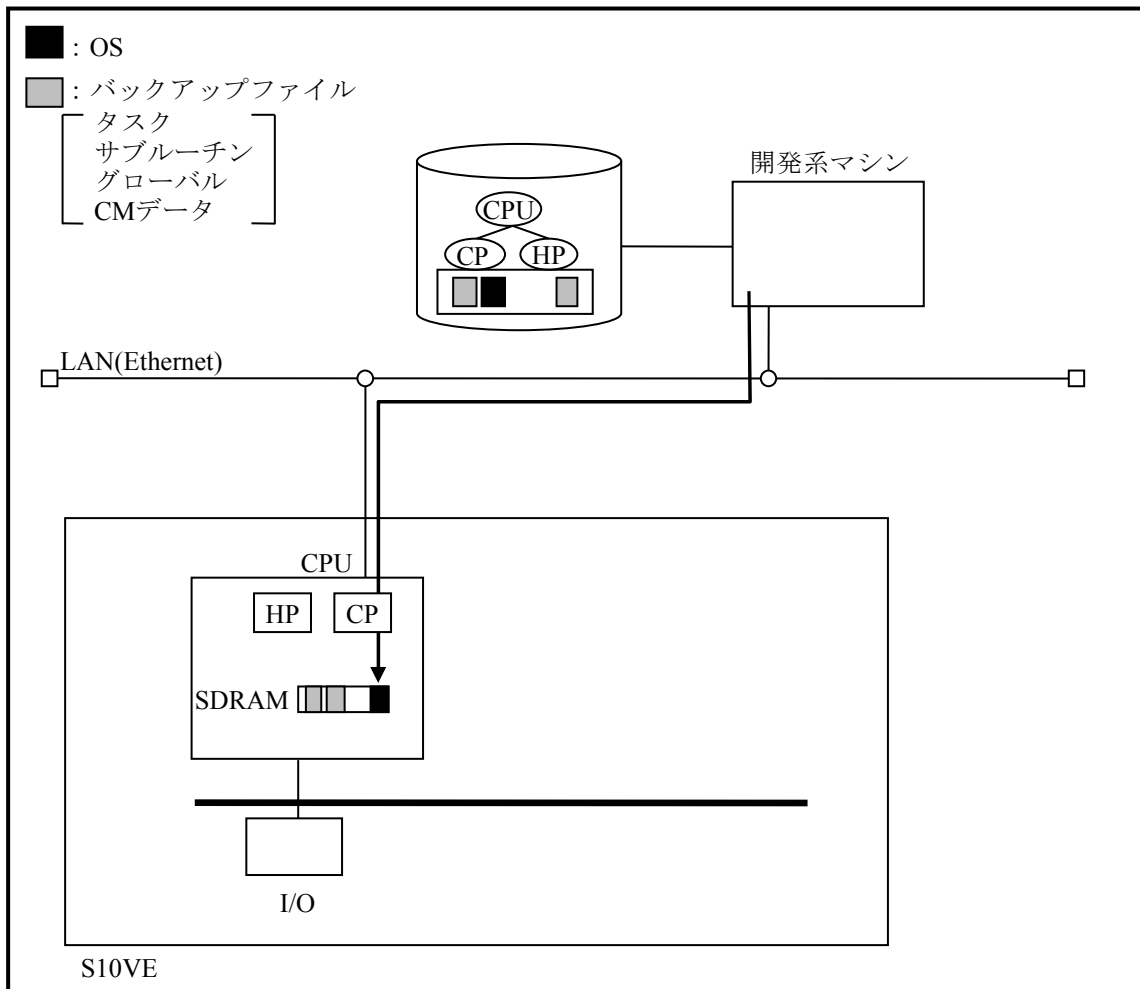


図 1 - 23 開発系マシンからのS10VE立ち上げ概略

10. 2 立ち上げ／PU制御の基本的な考え方

立ち上げ／PU制御処理における基本的な考え方を以下に示します。

S10VE全体の制御の考え方は下図に示すように、CPUにはCPとHPの2つのコアが存在し、サイト管理もそれぞれ別になっていますが、立ち上げ／PU制御はCPUとして行うため、CPのサイトを指定して処理を行います。

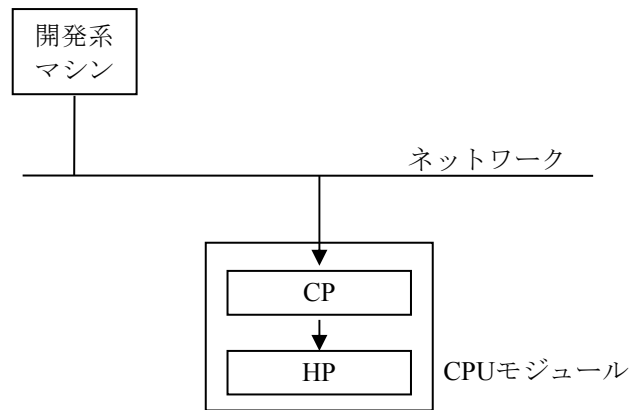
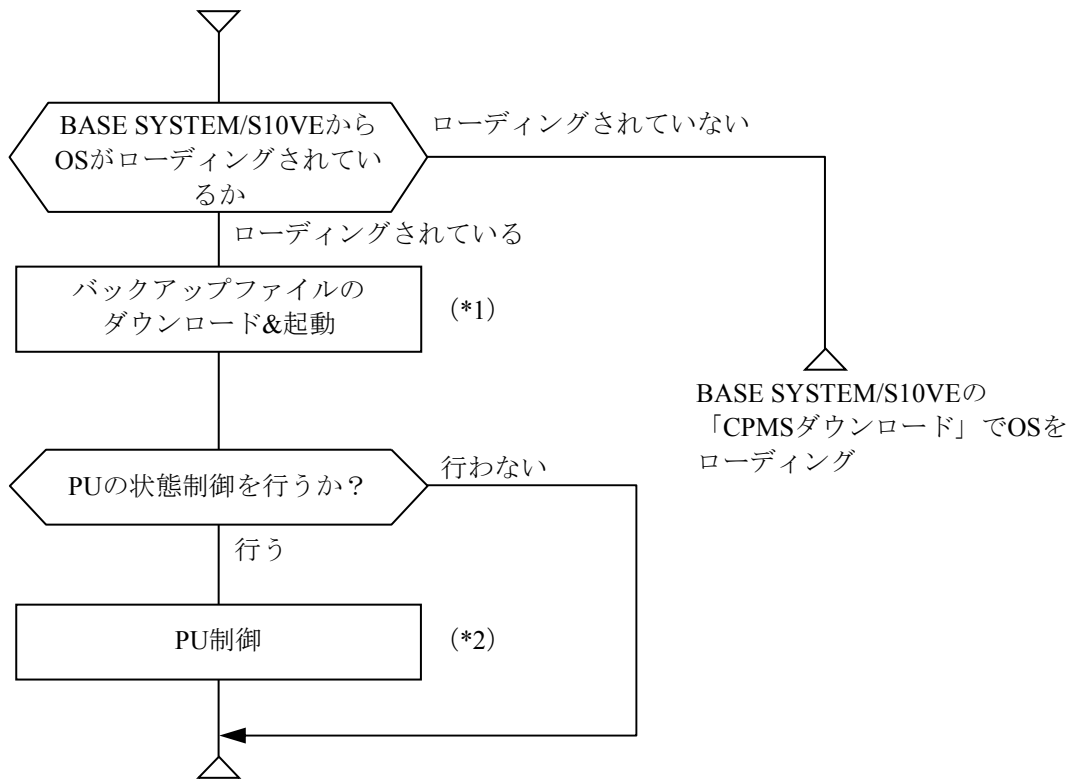


図 1-24 S10VE全体の制御の考え方

10. 3 立ち上げ/PU制御手順

S10VEの立ち上げおよびPU制御は、svrplコマンドおよびsvcpuctlコマンドを使用します。以下に手順を示します。



(*1) svrplコマンド使用：立ち上げ

(*2) svcpuctlコマンド使用：PU制御

10. 4 立ち上げ／停止種別

S10VEの立ち上げ／停止種別は表 1-22に示すように8種類に分かれます。

表 1-22 立ち上げ／停止種別

No.	種別	立ち上げ／停止区分
1	IPLスタート	立ち上げ
2	オンラインリスタート	
3	リセットスタート	
4	復電スタート	
5	ROMスタート	
6	電源断停止	停止
7	STOP要求停止	
8	ERROR STOP	

No.1～8の説明は（1）～（8）を参照してください。

S10VEの立ち上げ種別は、次のとおりです。

（1）IPLスタート（GLB保存IPLスタート含む）

svrplコマンドを使用してのスタートがIPLスタートです。

OS、バックアップファイル（タスク、サブルーチン（IRSUB、組み込みサブルーチン）、GLB、CMデータ）をダウンロードしてから、S10VEを立ち上げます。ダウンロード終了からイニシャルスタートタスクを起動するまでをIPLスタートと称します。

なお、OS、バックアップファイルのダウンロードには下記の4種類があります。svrplではLADDERとHI-FLOWのプログラムはダウンロードされません。

- ・OS、タスク、サブルーチン（IRSUB、組み込みサブルーチン）、GLB、CMデータのダウンロード

（2）オンラインリスタート（復電高速リスタート、リセット高速リスタート）

次のいずれかのイベントが発生したときに、スタートします。

- ・電源断停止のあとで復電したとき（復電高速リスタート）
- ・CPU STOP要求による停止のあとでCPU RUN要求が発生したとき（リセット高速リスタート）

このときS10VE側の処理は、電源断またはCPU STOP要求が発生したときの処理から再開します。

CPU RUN要求が発生してから、イニシャルスタートタスクを起動するまでがオンラインリスタートの処理です（このときには開発系マシンからS10VEへのダウンロードは行いません）。イニシャルスタートタスクを起動するまでに、I/Oの初期化が行われ、すべてのタスクがDORMANT状態となります。

(3) リセットスタート

CPUダウンにより異常停止したあとで、CPU RUN要求が発生したときにスタートします。OSの初期値ありデータは、IPL後の状態に戻されてIPL後の処理からスタートします。CPU RUN要求が発生してから、イニシャルスタートタスクを起動するまでがリセットスタートの処理です（このとき開発系マシンからS10VEへのダウンロードは行いません）。

(4) 復電スタート

CPUダウンにより異常停止したあとで、停復電が発生したときにスタートします。OSの初期値ありデータは、IPL後の状態に戻されてIPL後の処理からスタートします。復電からイニシャルスタートタスクを起動するまでが復電スタートの処理です（このとき開発系マシンからS10VEへのダウンロードは行いません）。

(5) ROMスタート

ROMからOSとプログラムやデータをローディングしてからS10VEを立ち上げます。タスク、サブルーチン（IRSUB、組み込みサブルーチン）、GLB、CMデータの他LADDERとHI-FLOWのプログラムもローディングされます。

S10VEの停止種別は、次のとおりです。

(6) 電源断停止

電源断（停電も含みます）することにより停止します。実行中のタスクは、アボートされることなく実行を停止します。また、オープン中のI/Oはクローズされることなく、CPUの実行を停止します。電源断停止中にS10VEの電源を入れると、ROMにデータが格納されている場合はROMスタートします。

(7) STOP要求停止

CPU STOP要求を受け付けることにより停止します。実行中のタスクはアボートされることなく実行を停止します。また、オープン中のI/Oはクローズされることなく、CPUの実行を停止します。CPU、I/Oともに電源は入れたままです。

CPU RUN要求を受け付けると、オンラインリスタート（リセット高速リスタート）します。

CPU STOP要求停止後、復電した場合にはROMスタートします。

(8) ERROR STOP

ハードウェアまたはソフトウェアの致命的なエラーによって停止します。CPU、I/Oともに電源は入ったままです。

再立ち上げるには、電源を切ってから再び入れる（復電スタート）か、CPU STOP要求を発行してからCPU RUN要求を発行して、リセットスタートするか、IPLスタート（OSをダウンロード）をしなければなりません。復電スタート時、メモリがクリアされていればROMスタートします。

<CPU STOP要求について>

CPU STOP要求として以下のイベントがあります。

- CPU RUN/STOPスイッチをSTOPにする。
- svrpl、svcpuctlコマンドからのCPU STOP要求によりSTOP割り込みを受信する。

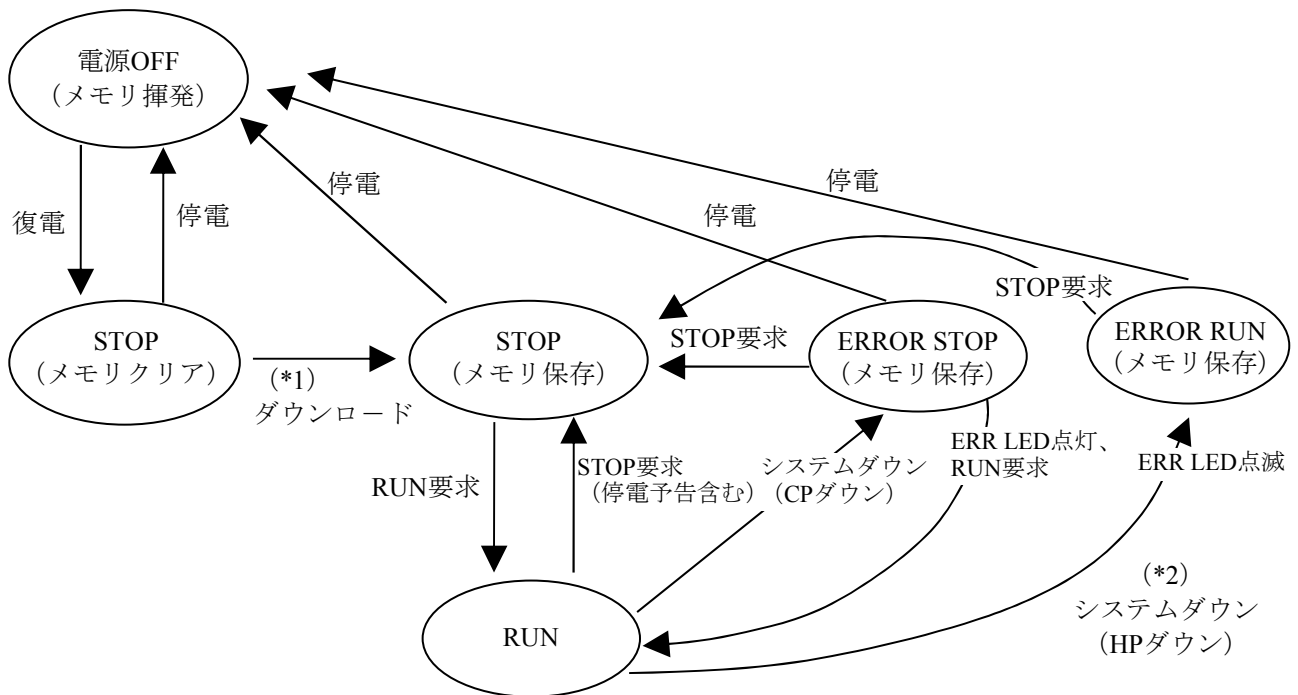
<CPU RUN要求について>

CPU RUN要求として以下のイベントがあります。

- CPU RUN/STOPスイッチをRUNにする。
- svcpuctlコマンドからのCPU RUN要求、svrplコマンドでダウンロード後にCPU RUN要求を送信する。

10. 5 PUの状態遷移

PUの状態遷移を図1-25に示します。



- (*1) ROM内にメモリがバックアップされている場合、ROMからメモリヘデータをダウンロードします。ROMにメモリがバックアップされていない場合には開発系マシンからネットワークを使用してメモリヘダウンロードします。
- (*2) HP停止時においてもCPは、動作を継続します。ERROR RUN状態は、HPのCPMSのみ停止している状態です。この場合には、両プロセッサの状態をCPU STOP状態としたあとでCPU RUN要求を受け付けます。

図1-25 PU (OS立ち上げ・停止) の状態遷移

10. 5. 1 立ち上げ操作

立ち上げは、10. 3節に示すように、開発系マシンからsvrplコマンドを実行することによって行います。開発系マシンでの操作については、「第2編 コマンドリファレンス」の「第7章 立ち上げ/PU制御」を参照してください。以下に主な機能と使用例を示します。

<機能>

svrplコマンドの主な機能を以下に示します。

- ・ 指定サイトまたは指定ユニット（S10VEユニット内全サイト）を立ち上げます。
- ・ OS、バックアップファイルをダウンロードします。

svrplコマンドは、オプション指定によってダウンロードするファイルを分けることができます。オプションによるダウンロードファイルを表1-23に示します。

表1-23 ダウンロードオプション

オプション ファイル	-all	指定なし
OS	○	○
・タスク ・サブルーチン ・GLB	○	○
CM	○	○

○：ダウンロードする。

×：ダウンロードしない。

- ・ 立ち上げ時にCPUの時刻を設定する/しないを指定します。
- ・ OS、バックアップファイルのダウンロード終了後にPUを起動する/しないを指定します。
- ・ svrplコマンド実行時にPUの起動/停止の確認応答（PUを停止させてもよいか）を取らないでPUを停止します。

各機能を使用した場合の立ち上げ例を以下に示します。

- CPUにOS、全バックアップファイルをダウンロードし、CPUへの時刻設定およびダウンロード後のCPU起動をする場合

実装されているPU

CPU CP (サイト名称=0001cp)
CPU HP (サイト名称=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 ****

>
```

- PUを停止させてもよいかどうかの確認応答を取らないで、CPUへOS、全バックアップファイルをダウンロードし、CPUへの時刻設定およびダウンロード後のCPU起動を行う場合
(-sオプションを指定し、確認応答を取らない場合はメッセージを表示しません。)

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

10. 5. 2 PU制御操作

PU制御は、開発系マシンからsvcpuctlコマンドを実行することにより行います。開発系マシンでの操作については、「第2編 コマンドリファレンス」の「第7章 立ち上げ／PU制御」を参照してください。以下に主な機能と使用例を示します。

<機能>

svcpuctlコマンドの主な機能を以下に示します。

- PUの状態（runまたはstop）を制御します。
- run要求時にCPUの時刻を設定します。
- svcpuctlコマンド実行時にPUの起動／停止の確認応答（本当に実行させてもよいか）を取らないで実行します。
- サイトの状態を表示します。

各機能を使用した場合の立ち上げ例を以下に示します。

- CPUにrun要求を行うと同時に、CPUへの時刻設定を行う場合

実装されているPU

```
CPU CP (サイト名称=0001cp)
CPU HP (サイト名称=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
```

- CPUにstop要求を行う場合

実装されているPU

```
CPU CP (サイト名称=0001cp)
CPU HP (サイト名称=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
```

- PUを停止させてもよいかの確認応答を取らないで、CPUにrun要求を行い、CPUへの時刻設定を行う場合

(-sオプションを指定し、確認応答を取らない場合はメッセージを表示しません。)

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

- CPUの状態表示を行う場合

```
#svcpuctl -u 0001cp -ss
```

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


第11章 svdebug (オンラインデバッガ) とデバッグ支援コマンド

11. 1 概要

svdebugは、S10VE上で動作するプログラムを開発系マシン上からデバッグするためのコマンドで、以下のような基本的なデバッグ機能を提供します。

分類	サブコマンド	機能
タスク起動/停止	qu ab re ta su rs tm ct sht	タスクの起動要求 タスクの起動禁止 タスクの起動禁止解除 タスクの状態表示 タスクの実行抑止 タスクの実行抑止解除 タスクの周期起動 タスクの周期起動解除 タスクの周期起動表示
メモリプリント/ パッチ	md sd bs bg mcp mmv mf	アドレス指定でのメモリプリント/パッチ 名称指定でのメモリプリント/パッチ ビットデータの設定 ビットデータの表示 メモリのコピー メモリの移動 メモリのパターンセット
システムエラー表示	el ss	システムエラーの表示 (svelogコマンドの実行) システム状態表示 (svcpuctlコマンドの実行)
現在時刻設定/表示	st gt	現在時刻設定 現在時刻表示
ブレークポイント 設定/解除	br/stickybr rb rd rr go	ブレークポイントの設定/表示 ブレークポイントの解除 レジスタの表示 レジスタの書き換え ブレークポイントからの実行再開
アップ/ダウン ローディング、 コンペア	ld sv cm	バックアップファイル → S10VEメモリ転送 S10VEメモリ → バックアップファイル転送 S10VEメモリ/バックアップファイル比較
DHP記録許可/禁止	dr ds	DHP記録許可 (svdhpコマンドの実行) DHP記録禁止 (svdhpコマンドの実行)
ラダーのデバッグ機 能	lbr lrb lrd lrr lgo s	ブレークポイントの設定/表示 ブレークポイントの解除 レジスタの表示 レジスタの書き換え ブレークポイントからの実行再開 ステップ実行
その他	si sp ps pe ver svadm svdhp help q !	スタック初期化 スタック使用量表示 デバッグ文の表示開始 デバッグ文の表示終了 CPMSのバージョン表示 アドレスに対する情報表示 (svadmコマンドの実行) DHPの表示 (svdhpコマンドの実行) サブコマンド一覧表示 デバッガの終了 開発系マシン上のコマンドの実行

11. 2 S10VE状態とサブコマンド実行可否

S10VEの状態により、サブコマンドを実行できるかどうか異なります。S10VE状態とサブコマンドの実行可否の一覧を以下に示します。

分類	サブコマンド	POFF	STOP	RUN	ERR RUN (HP停止)	ERR STOP
タスク起動/停止	qu, ab, re, ta, su, rs, tm, ct, sht	×	×	○	○ (*6)	×
メモリプリント/ パッチ	md, sd, bs, bg, mcp, mmv, mf	×	○	○	○	○
システムエラー表示	el, ss	×	○	○	○	○
現在時刻設定/表示	st, gt	×	×	○	○ (*6)	×
ブレークポイント 設定/解除	br, rb, rd, rr, go	×	×	○	○ (*6)	×
アップ/ダウン ローディング、 コンペア	ld, sv, cm	×	△ (*7)	△ (*2)	○ (*2) (*6)	△ (*7)
DHP記録許可/禁止	dr, ds	×	×	○	○ (*6)	×
ラダーのデバッグ機能	lbr, lrb, lrd, lrr, lgo, s	×	×	○	×	×
その他	si, sp, ps, pe	×	×	○	○ (*6)	×
	svdhp	×	×	○	○ (*6)	×
	ver	×	○	○	○ (*6)	○
	svadm, help, q, !	○	○	○	○	○

○：使用可、×：使用不可、△：通信可ならば使用可

(*1) バックアップファイルをプリント/パッチできます。

(*2) 下記オプションの機能はS10VEの状態がSTOP状態でのみ使用できます。

ld : -cm

(*3) S10VEがCPU RUN状態以外の場合は、svhidasでDHPを収集できます。

(*4) 開発系マシン側の情報はCPU STOP状態でも表示できます。

(*5) 設定の表示はCPU STOP状態でも表示できます。

(*6) HP側のサイトには実行できません。

(*7) ld -gはCPU STOP状態でも実行できます。

状態名	説明
POFF	S10VE電源断状態
STOP	OS/ミドルウェア/アプリケーション停止状態
RUN	OS/ミドルウェア/アプリケーション実行状態
ERR RUN	HPエラーによる停止状態
ERR STOP	CPエラーによる停止状態

第11章 svdebug（オンラインデバッガ）とデバッグ支援コマンド

svdebugは前述の機能をサブコマンドとして実現しています。svdebugを起動するとプロンプトが出力されます。プロンプトには、処理対象サイト名もあわせて表示されます。これにより、現在のデバッグ対象となっているサイトを確認できます。このプロンプトに対して適切なサブコマンドを入力し、デバッグを進めることができます。サブコマンド“q”を入力すると、svdebugは終了します。

以下に例を示します。

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

リアルタイムリソースは、所有者種別、使用者種別によりアクセスが制限されます。svdebugを起動する前に適切な使用者種別を設定しておいてください。

ほとんどの機能はS10VEのOSが立ち上がった状態で使用できます。また、一部の機能はS10VEのOSが立ち上がっていても使用できます。

svdebugのサブコマンド処理は、[Ctrl] + [C] によって中断できます。

11. 3 基本的な機能

(1) プログラム、データのダウンロード・アップロードサブコマンド

アロケータ・ローダ・ビルダ (ALB) はバックアップファイル中にプログラムやデータを組み込みます。何らかの方法でバックアップファイルの内容をS10VEのメモリに反映しなければなりません。ldサブコマンドは、バックアップファイルの内容をS10VEのメモリ上に反映します。また、その逆にS10VEのメモリ内容をバックアップファイルに反映することもできます。これにはsvコマンドを使用します。また、バックアップファイルと対応する主メモリの内容を比較するために、cmサブコマンドを用意しています。

以下の例では、TN=5のタスクの状態を確認後、ダウンロードを行い、さらにGLB (名称: indata) のダウンロードを行っています。

```
$ 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 >
```

上述したように、ALBでの操作はそのままではS10VEに反映されません。つまり、開発系マシン上のバックアップファイルとS10VE上の状態が不一致となる場合があります。

- 開発系マシン上のバックアップファイルには存在するがS10VE上には存在しない場合
ALBで登録したが、svrplやsvdebugのldサブコマンドによって反映されていない場合です。
- 開発系マシン上のバックアップファイルとS10VE上の両方に存在する場合
この場合、2つの状態が考えられます。
1つは開発系マシン上とS10VE上に全く同一のものが存在する場合と、もう1つは開発系マシン上とS10VE上に異なるものが存在する場合です。前者はALBで登録したものをsvrplやsvdebugのldサブコマンドによって反映した直後の状態です。後者はALBで登録や削除を繰り返した場合は。
- 開発系マシン上のバックアップファイルには存在しないが、S10VE上には存在する場合
ALBで登録し、さらにsvrplやsvdebugのldサブコマンドにより反映したあとに、ALBで該当のものを削除した場合です。

- 開発系マシン上のバックアップファイルとS10VE上の両方に存在しない場合

ALBで登録し、さらにsvrplやsvdebugのldサブコマンドにより反映したあとに、ALBで該当のものを削除し、S10VE上からも削除した場合は、

開発系マシンとS10VEの状態は、svmapコマンドやsvdebugのldサブコマンドにより調べることができます。必要に応じてsvrplまたはsvdebugのldサブコマンドにより状態を一致させてください。特に、リソースの削除時は、開発系マシン上だけで削除され、S10VEに反映されていない状態にならないようにしてください。

(2) タスク制御サブコマンド

コントローラ上にダウンロードしたプログラムは、通常他のタスクから起動要求を受けて起動されますが、デバッグ時にはオペレータの指示によりタスクを起動したり停止したりできると便利です。

svdebugではタスクの起動や停止を行うためのサブコマンドを用意しています。

状態を確認しながらTN=5のタスクを起動する例を以下に示します。

```

$ 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) メモリプリント・パッチサブコマンド

svdebugはメモリの内容を表示したり変更したりするサブコマンドを持ちます。これによってGLBなどのメモリ内容を任意の値に変更したり、参照したりしながらテストを行えます。

パッチまたは表示の対象は、主メモリ、バックアップファイル、またはその両方を指定できます。入力データを格納するGLB (名称: indata) にデータを設定したあと、該当データを処理GLB (名称: outdata) に書き込むタスクを起動し、処理結果を確認する例を以下に示します。

```
$ 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(0x000004) 00000000 : e
4 raddr : *1
1 name : indata -g
2 storage (s,m,*) : m
3 baddr : 0
4 raddr : 0
0x50050000(0x000000) 00001000 :
0x50050004(0x000004) 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) 時刻設定表示サブコマンド

S10VEは内蔵クロックを持ちます。この内蔵クロックへの時刻設定はsvdebugからも行えます。

S10VE上の現在時刻を確認したあと、時刻を再設定する例を以下に示します。

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

(5) システムエラー表示、システム状態表示

テストやデバッグ作業を進めているときには、システムの状態やシステムエラーの内容を参照してください。RPDPではシステムエラーを表示したり、システムの状態を表示したりするためのコマンド (svelog、svcpuctl) を用意していますが、svdebugでは、それらのコマンドをサブコマンド (el、ss) として起動できます。

(6) dhp記録許可・禁止、表示サブコマンド

S10VEはデバッグ支援機能としてDHP (Debugging Helper) 機能を提供します。DHPにより、OSやユーザタスクの動作を解析できます。RPDPではDHPの表示や、DHP記録許可、禁止を行うコマンド (svdhp) を用意していますが、svdebugでは、それらのコマンドをサブコマンド (svdhp) として起動できます。

(7) その他のサブコマンド

今まで説明したサブコマンドの他にもいくつかのサブコマンドがあります。

si、spサブコマンドタスク実行時のスタック使用量を求める作業がより容易になります。

svadmコマンドアドレスに対応するSAREA名やIRSUB名を求めることができます。

!サブコマンドsvdebugを終了することなく開発系マシン上のコマンドを実行できます。

helpサブコマンド簡潔な説明文が表示されます。

11. 4 その他の機能

svdebugではユーザの利便性を考慮し、いくつかのコマンド行オプションを設けています。以下のオプションを指定することで、svdebugの動作を変更し、用途に合わせて使用できます。

-i : キー入力結果をファイルに出力します。サブコマンドの出力結果は記録されません。

-o : 指定されたファイルに操作の日付および結果を出力します。

どのような操作を行ったのか記録したい場合に便利です。

-r : 指定されたコマンドファイル内のサブコマンド行を実行します。

一連の操作を1回で行いたいときに便利です。

-iオプションで作成したファイルを入力に使用できます。

-s : このオプションに指定したサブコマンドを直接実行します。

コマンドプロシージャなどを作成する場合に便利です。

-u site : デバッガの処理対象となるサイト名称を指定します。

11. 5 デバッグ支援コマンド

11. 5. 1 svelogコマンド

SIOVE内エラーログバッファからエラーログ情報をネットワークを介して取り込み、最新のエラーログ情報から順に出力します。

出力されたエラーログ情報からは、いつ (発生時刻)、何が原因で (EC: エラーコード)、だれが (TN: タスク番号)、どこで (PC: プログラムカウンタ)、何をしようとしたか (iarv0~iarv8, iarvn1~iarvn8: PC前後のプログラムの内容) などを知ることができます。さらに詳細に解析するための、レジスタ情報やDHPトレース情報などもあわせて出力します。代表的な操作例を以下に示します。

(例1)

サイト0001cpのエラーログ情報を、簡略化した短い形式で出力します。

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

(例2)

サイト0001cpのエラーログ情報を、すべて出力します。

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

-f mオプション省略時も同様です。

(例3)

サイト0001cpのエラーログ情報に加えてDHPトレース情報も出力します。

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

(例4)

サイト0001cpのエラーログ情報に加えてDHPトレース情報も出力すると共に、abcというファイルにエラーログ情報を格納します。

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

11. 5. 2 svdhcpコマンド

SIOVEのDHPトレースバッファに格納されている、現在のDHPトレース情報をネットワークを介して取り込んで出力します。

出力するDHPトレース情報には、DHP記録時刻、DHPトレースポイント、解析に必要なデータがあります。また、DHP記録の停止／再開を制御します。

代表的な操作の例を以下に示します。

(例1)

サイト0001cpのDHPトレース情報を指定されたカウント分 (10) 出力します。

```
svdhcp -u 0001cp +10
```

指定されたカウント (+10) が省略された場合は、すべてのDHPトレース情報を出力します。

(例2)

サイト0001cpのDHP記録を停止します。

```
svdhcp -u 0001cp -off
```

(例3)

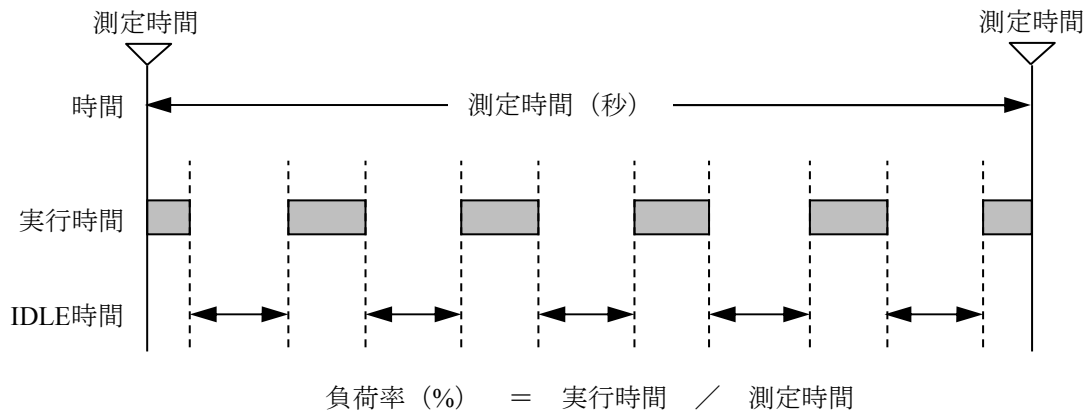
サイト0001cpのDHP記録を再開します。

```
svdhcp -u 0001cp -on
```

11. 5. 3 svcpunowコマンド

SIOVEの負荷率をサイト (PU) 単位に出力します。

負荷率は、測定開始から終了までの実行時間の合計によってPUをどの位占有したかを百分率 (%) で出力します。



代表的な操作例を以下に示します。

(例) サイト0001cpの負荷率を10秒測定します。

```
#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)

(注) アンダーライン部はユーザが入力してください。

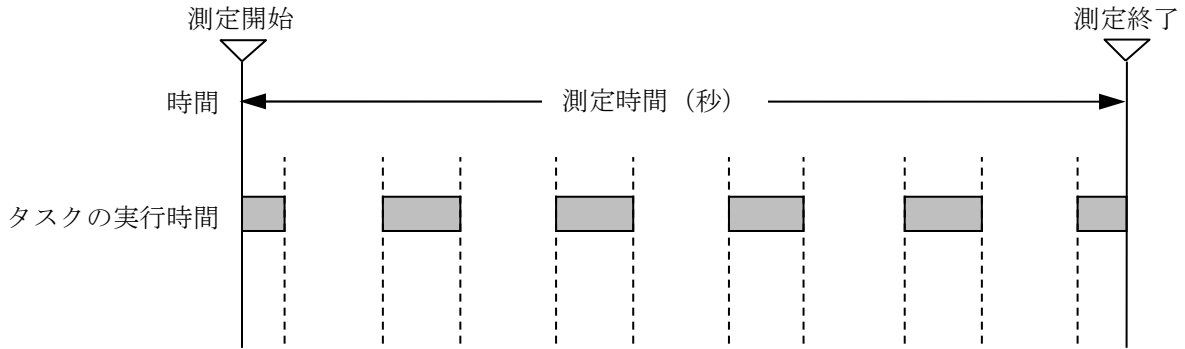
- (*1) PU名称
- (*2) サイト名称
- (*3) PU負荷率

測定時間の指定 (-t 10) が省略された場合は、1秒間の負荷率を測定します。

11. 5. 4 svtimexコマンド

コントローラに登録されているタスクの稼働情報を出力します。

出力するタスクの稼働情報には、タスクのPU負荷率、実行回数、実行時間、平均実行時間があります。



$$\text{タスクのPU負荷率 (\%)} = \text{タスクの実行時間の合計} / \text{測定時間}$$

$$\text{タスクの平均PU実行時間} = \text{タスクの実行時間の合計} / \text{タスクの実行回数}$$

代表的な操作例を以下に示します。

(例1) 下記タスクの稼働情報を10秒間測定します。

タスク名称=taska、タスク番号=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)

(注) アンダーライン部はユーザが入力してください。

- (*1) タスク名称 (番号)
- (*2) タスクのPU負荷率
- (*3) タスクの実行回数
- (*4) タスクの実行時間の合計
- (*5) タスクの平均PU実行時間

測定時間の指定 (-t 10) が省略された場合は、1秒間の負荷率を測定します。

(例2) 会話形式で下記タスクの稼働情報を3600秒間測定します。

タスク名称=taska、 タスク番号=123

タスク名称=taskb、 タスク番号=124

タスク名称=task22c、 タスク番号=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]
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
```

(注) アンダーライン部はユーザが入力してください。

第2編 コマンドリファレンス

第1章 コンパイラ

<名前>

svdatagen

<形式>

svdatagen [-u site] file

<機能説明>

svdatagenは、ローダ（svload）でGLB用のバックアップファイルにロード可能な初期値データのバイナリファイルを生成します。

生成するバイナリファイルの名称は、入力ファイルの名称のサフィックスを.binに変更した名称で出力します。入力ファイル名にサフィックスがない場合は、入力ファイル名の後ろに.binを付加した名称で生成します。

<引数説明>

-u site：初期値生成にGLB/VAL名のアドレス解決が必要な場合に参照するサイト名称を指定します。

このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

GLB/VAL名のアドレス解決が不要である場合は、サイト名を指定する必要はありません。

file：初期値データを記述したテキストファイルを指定します。

入力ファイルは1ファイルだけ指定できます。

1ファイルの中に複数のGLB初期値が記述できます。複数エリアのGLBが混在していても構いません。

<終了コード>

0：正常終了

0以外：異常終了

<データの配置>

以下に、このコマンドで生成するバイナリデータ内のデータの配置を示します。

初期値データが構造体型である場合は構造体内のメンバのアラインは、構造体の先頭を4バイトの境界と仮定しそれぞれ下記とします。

表 2-1 バイナリデータ配置

型	アライン (バイト)
char	1
short	2
int	4
long	4
float	4
double	4
配列	要素の型に従う
構造体	最大アライン数のメンバに同じ

第2章 プログラミングコマンド

<名前>

`makehce` - プログラムグループの保守、アップデート、および再生成

<形式>

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

<機能説明>

• `makefile`の構造

`makefile`にはターゲット行、シェルコマンド行、マクロ定義、およびインクルード行を含めることができます。

• ターゲット行

ターゲット行は、空白で区切られたターゲットの非ヌルリストと、コロン (:) またはダブルコロンの (::) と、ディペンデントと呼ばれる前提条件ファイルのリストから構成されます。ファイル名をディペンデントとして生成するときには、パターンマッチング表記がサポートされます。

• シェルコマンド行

ターゲット行のセミコロン (;) のあとに続くテキストと、タブで始まる後続のすべての行はターゲットを更新するために実行されるシェルコマンドです。タブや#で始まらない最初の行は、新しいターゲット定義、マクロ定義、またはインクルード行から始まります。<円記号><復帰改行>というシーケンスを使用すればシェルコマンドを複数の行にわたって続けることができます。コマンド行が関連付けられているターゲット行はルールと呼ばれます。

• マクロ

文字列1=文字列2という書式の行はマクロ定義です。マクロは、`makefile`のどの位置でも定義できますが、普通は、先頭にまとめて定義します。文字列1はマクロ名、文字列2はマクロ値です。文字列2は、コメント文字または非拡張復帰改行文字までの全文字数として定義されます。=の左右にある空白とタブはコメント内にあるものを除いて無視されます。`makefile`内の他の場所にある\$ (文字列1) は文字列2に置換されます。1文字のマクロ名を使用し、かつ、置換文字列がない場合、括弧はオプションになります。オプションの置換文字列、\$(文字列1[:サブセット1=[サブセット2]])を指定できます。この文字列を指定した場合、文字列1の値にあるサブ文字列の終わりでオーバーラップしていないすべてのサブセット1がサブセット2に置換されます。マクロ値内のサブ文字列は、空白、タブ、復帰改行文字、行の始めによって区切られます。

例えば、

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

という文字列を入力すると、

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

という文字列は、次のように評価されます。

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

マクロ値には、他のマクロへの参照を含めることができます。

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

\$(TWELVE)の値は\$(ONE)2に設定されますが、ターゲット行、コマンド行、またはインクルード行で使用するときには、12に拡張されます。このあと、`makefile`やコマンド行の別の定義によってONEの値がさらに変更された場合、\$(TWELVE)への参照はこの変更を反映します。

マクロ定義をコマンド行で指定すれば、`makefile`の定義を無効にできます。`makehce`には特定のマクロが自動的に定義されます。

- ・インクルード行

makefileの最初の7文字がincludeで、そのあとに1つまたは複数の空白またはタブ文字がある場合、その行の他の部分はファイル名とみなされます。この部分は、現在のmakehce呼び出しによってそのファイル内のいずれかのマクロが拡張されたあとに別のmakefileとして読み取り処理されます。

- ・一般的な説明

makehceは、makefileに入力されたコマンドを実行して、1つまたは複数のターゲット名を更新します。ターゲット名はプログラムの名前になります。-fオプションを指定しなかった場合、makefile、Makefileの順番で処理されます。-fに - を指定した場合、標準入力を使用されます。複数の-fオプションを指定できます。makefileの引数は、指定された順番で処理されます。複数のmakefile名を指定する場合、それぞれのmakefile名の前に-fオプションを付けなければなりません。組み込みルールやマクロがある場合、makefileの内容はそれらに優先します。

コマンド行でターゲット名を指定しなかった場合、makehceは、makefileの最初のターゲットを更新します。ターゲットは、ターゲットよりも新しいファイルに付属している場合にだけ更新します。

ファイルを指定しなかった場合、古いものと判断されます。必要な場合、ターゲットのすべてのディペンデント（ターゲットに付属なもの）は、ターゲットが更新される前に再帰的に更新されます。これは、ターゲットに関する従属ツリーの深さ優先（縦型）更新に影響を与えます。

ターゲットが、ターゲット行の区切り文字のあとに指定されたディペンデント（明示的なディペンデント）を持っていない場合、そのターゲットに関連付けられているシェルコマンドは、そのターゲットが古くなったときに実行されます。

ターゲット行では、1つまたは複数のターゲット名と明示的なディペンデント名の間、シングルまたはダブルコロンを入れることができます。1つのターゲット名を複数のターゲット行で指定できますが、すべてのターゲット行が同じ（シングルまたはダブルコロン）タイプでなければなりません。シングルコロンの場合、明示的コマンドに関連付けられるターゲット行の数は多くても1つです。明示的コマンドが指定されている場合、いずれかの行にディペンデントが指定されているターゲットが古くなると、明示的コマンドが実行されますが、明示的コマンドが指定されていない場合には、デフォルトルールを実行できます。ダブルコロンの場合、ターゲット名を含んでいる複数のターゲット行に明示的コマンドに関連付けることができます。特定の行にディペンデントが指定されているターゲットが古くなると、その行に関するコマンドが実行されます。

ターゲット行とそれらの関連シェルコマンド行も、ルールと呼ばれます。ハッシュマーク（#）と復帰改行文字は、ルール内にあるコメントを除いてmakefile内にあるコメントを囲みます。ルール内のコメントは、SHELLマクロの設定によって決定されます。

以下のmakefileは、pgmが2つのファイル（a.objおよびb.obj）に従属していることと、それらに対応するソースファイル（a.cおよびb.c）と共通ファイルincl.hに従属していることを示しています。

```

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

```

コマンド行は、そのシェルによって1つずつ実行されます。それぞれのコマンド行には、- または + のどちらか一方のプリフィックスまたは両方のプリフィックスを指定できます。この2つのプリフィックスについては以下で説明します。

makehceは、コマンドが0以外の状態（コマンド終了コード）を戻したときに終了します。

-iオプションが指定されているか、または**makefile**に**.IGNORE**という特別なターゲットがある場合、エラーメッセージが標準出力にプリントされることに変わりはありませんが、エラーを引き起こすコマンド行がいくつあっても、**makehce**は**makefile**の実行を継続します。コマンド行の先頭に - が存在する場合、その行によって戻されたエラーは標準出力にプリントされますが、**makehce**は終了しません。- というプリフィックスを使用すれば、**makefile**内でエラーを選択して無視できます。

-kオプションが指定されているときにコマンド行がエラー状態を戻した場合、現在のターゲットにおいては作業が放棄されますが、そのターゲットに従属していない他の分岐では作業が継続されます。

MAKEFLAGS環境変数に**-k**オプションがある場合、**-S**オプションを指定すれば、処理をデフォルトに戻すことができます。

-nオプションは、実行を伴わないコマンド行のプリントを指定します。ただし、+ というプリフィックスが付けられている場合、その行は必ず実行されます。**-t** (**touch**) オプションは、コマンドを実行しないでファイルの修正日を更新します。

第2章 プログラミングコマンド

コマンド行は実行される前にプリントされますが、その行の先頭に@という文字があるとプリントが抑制されます。-sオプションが指定されているかまたはmakefileに.SILENTという特別なターゲットがある場合、すべてのコマンド行のプリントが抑制されます。makehceによってプリントされる（開始タブを除いた）すべての情報は、修正を受けないでシェルに直接渡されます。

したがって、

```
echo a¥  
b
```

というコマンドは、シェルと同じように、

```
ab
```

という文字列を生成します。

• オプション

オプションは、任意の順序で指定できます。また、-fオプション以外は別々に指定することも、1つの - でまとめて指定することもできます。

-e 環境変数がmakefile内の割り当てに優先します。

-f makefile

指定した記述ファイル名がmakefileになります。-のファイル名は標準入力を表します。

makefileの内容は、組み込みの規則やマクロが存在する場合、それらに優先します。このオプションは複数指定でき、指定された順に処理されます。

-p マクロ定義およびターゲット記述の完全なセットを標準出力に書き込みます。

-i コマンドによって戻されたエラーコードを無視します。makefileに.IGNOREという特殊ターゲット名が含まれている場合にも、このモードになります。

-k コマンドが0以外の状態に戻すと、現在のエンタリに対する作業を放棄されますが、そのターゲットに従属しない他の分岐の作業は続きます。これは-Sオプションの反対です。

-kと-Sを同時に指定した場合、あとに指定したオプションが使用されます。

-n 非実行モード。コマンドをプリントしますが、実行はしません。@で始まる行もプリントされません。ただし、コマンドに+というプリフィックスが付いている行は実行されます。

-q 質問。makehceコマンドは、ターゲットファイルが最新状態であるかどうかに応じて、0または0以外の状態に戻します。このオプションを指定しても、ターゲットファイルは更新されません。

-r サフィックスリストをクリアし、組み込みの規則を使用しません。

-s サイレントモード。コマンド行は、標準出力にプリントされずに、実行されます。makefile内に.SILENTという特殊なターゲット名がある場合にも、このモードになります。

- S ターゲットを更新するコマンドを実行しているときにエラーが発生すると、そのコマンドを終了させます。これはデフォルト値であり、-kと反対のオプションです。-kと-Sを同時に指定した場合、最後に指定したオプションが使用されます。このオプションを指定すれば、MAKEFLAGS環境変数にあるkフラグを無効にできます。
- t 普通のコマンドを発行しないで、ターゲットファイル进行处理します（ターゲットファイルを更新します）。

[マクロ名 = 値]

ゼロまたはそれ以上のコマンド行マクロ定義を指定できます。

[ターゲット名]

makefile内に0またはそれ以上のターゲット名を指定できます。指定されたターゲットがmakehceによって更新されます。ファイル名を指定しなかった場合、makehceは、makefile内にある推論規則以外の最初のターゲットを更新します。

• 環境

常に無視されるSHELL環境変数を除いて、環境に定義されたすべての変数は、makehceによって読み取られ、マクロ定義として処理されます。定義のない変数や、空の文字定義を備えた変数は、makehceに含められます。

マクロ定義には、次の順序で読み取られる4つのソースがあります。すなわち、内部（デフォルト）、現在環境、makefile、コマンド行です。こうした処理順序があるためmakefile内のマクロ割り当ては環境変数に優先します。-eオプションを使用すれば、環境変数をmakefile内のマクロ割り当てに優先させることができます。コマンド行のマクロ定義は、常に他のすべての定義に優先します。

makehceが処理するMAKEFLAGS環境変数は、コマンド行に定義されている（-f、-p、-d以外の）正当な入力オプションを含んでいるものとします。MAKEFLAGS変数はmakefileにも指定できます。

MAKEFLAGSがコマンド行にもmakefileにも指定されていない場合、makehceは、その変数を独自に構成して、その中にコマンド行で指定されたオプションとデフォルトオプションを入れ、その変数をコマンドに渡します。したがって、MAKEFLAGSには、常に現在の入力オプションが含まれます。このことは、再帰的なmakehceにとって非常に便利なことです。このため、ソフトウェアシステム全体でmakehce -nを再帰的に実行すれば、何が実行されたかを調べることができます。これは、コマンドを一切実行しないでソフトウェアプロジェクトに関するすべてのmakefileをデバッグする1つの方法です。

第2章 プログラミングコマンド

・サフィックス

多くの場合、ターゲットやディペンデントにはサフィックスが付きます。`makehce`には特定のサフィックスに関する情報が組み込まれているため、それを参照すれば、ターゲットの更新に適切な推論ルールが特定できます（「推論ルール」の項を参照してください）。現在のデフォルトサフィックスのリストは次のとおりです。

```
.obj .c .c- .src
```

これらのサフィックスは、`.SUFFIXES`という特殊な組み込みターゲットのディペンデントとして定義されます。この定義は、`makehce`によって自動的に行われます。

`makefile`において`.SUFFIXES`のディペンデントリストとして追加サフィックスを指定できます。こうした追加値は、デフォルト値に追加され、複数のサフィックスが累積されます。サフィックスリストの順序には意味があります（「推論ルール」の項を参照してください）。サフィックスの順序を変更したいときには、まず、ディペンデントリストを`NULL`にした`.SUFFIXES`を定義し、`.SUFFIXES`の現在の値をクリアしてから、希望する順序でサフィックスを指定した`.SUFFIXES`を定義してください。

・推論ルール

特定のターゲットまたはディペンデント名（例：`.obj`で終わるもの）は、推論可能なディペンデント（例：`.c`や`.src`）を備えています。こうした名前に関する更新コマンドが`makefile`にない場合や、推論可能なディペンデントファイルが存在する場合には、ターゲットを更新するためにそのディペンデントファイルがコンパイルされます。この場合、`makehce`が備えている推論ルールは、サフィックスを調べ、適切な推論ルールを決定することによって他のファイルからファイルを作成します。現在、次のようなデフォルト推論ルールが定義されています。

シングルサフィックスルール

```
.c-
```

ダブルサフィックスルール

```
.c.obj .c-.obj  
.s.obj
```

ダブルサフィックス推論ルール (.c.obj) は、x.cからx.objを作成する方法を定義します。

サフィックス.objからサフィックス.objのファイルを作成するルールは、ターゲットとして.c.objを備え、ディペンデントを備えていないエントリとして指定されます。ターゲットに定義されたシェルコマンドは、.cファイルから.objファイルを作成するためのルールを定義します。スラッシュを含まずにドットで始まるターゲット名は、ターゲット（明示的な）ルールではなく、推論（暗黙的な）ルールとして識別されます。1つのドットを備えたターゲットはシングルサフィックス推論ルールです。また、2つのドットを備えたターゲットはダブルサフィックス推論ルールです。ユーザは、makefileにおいて追加推論ルールを定義でき、デフォルト推論ルールを再定義することや、キャンセルすることができます。

.cファイルを.objファイルに変更するデフォルトの推論ルールは、次のようになります。

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

結果コマンドにオプション事項を挿入できるように、デフォルトの推論ルールでは特定のマクロが使用されます。例えば、shcに対するコンパイラオプションには、CFLAGSが使用されます。

このマクロは、makehceによって自動的に定義されますが、makefileで定義し直すことができます。

推論ルール (<) で使用される特殊な組み込みマクロもいくつかあります（「組み込みマクロ」の項を参照してください）。

ターゲットが明示的なディペンデントを持っていない場合、またはディペンデントにも定義された明示的なルールで一致するターゲットがない場合、makehceが探す最初の推論ルールは、ターゲットの（ディペンデントの）サフィックス（NULLになる場合もある）と、そのルールの他のサフィックスに一致するファイルの両方に一致するものです。makehceは、.SUFFIXES値リストの前から後ろへ向かってこの探索を行うため、.SUFFIXESを定義する順序が重要な意味を持ちます。

makehceは推論ルール.c.objを定義するため、「一般的な説明」の項に示されている例を、もっと簡単に書き換えることができます。

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


第2章 プログラミングコマンド

・組み込みターゲット

`makehce`は、特殊なターゲットに関する情報を備えています。これらは、`makefile`で発行するように指定してください（`.SUFFIXES`を除いて、これらは`makehce`によって自動的に指定されますが、変更はできません）。

`.DEFAULT` : ファイルを作成しなければならないときに、そのファイルに関する明示的なコマンドや関連する組み込みルールがない場合、`.DEFAULT`が`makefile`に定義されていれば、目的名`.DEFAULT`を伴ったコマンドが使用されます。`.DEFAULT`は明示的なディペンデントを持っていません。

`.PRECIOUS` : `QUIT`、`INTERRUPT`、`TERMINATE`、`HANGUP`コマンドが入力されても、このターゲットのディペンデントは削除されません。

`.SILENT` : `-s`オプションと同じ働きをします。ディペンデントや明示的なコマンドの指定は不要です。

`.IGNORE` : `-i`オプションと同じ働きをします。ディペンデントや明示的なコマンドの指定は不要です。

`.SUFFIXES` : `.SUFFIXES`の明示的なディペンデントが、既知のサフィックスの組み込みリストに追加され、推論ルールと併用されます。`.SUFFIXES`がディペンデントを持たない場合、既知のサフィックスのリストはクリアされます。`.SUFFIXES`に関連付けられているコマンドはありません。

・組み込みマクロ

ターゲットの作成ルールを作る際に役立つ5つのマクロが組み込まれています。こうしたマクロの意味を明確に定義するためには、ターゲットおよびディペンデントという用語について説明する必要があります。`makehce`は、ターゲットを更新する場合、更新すべき一連のターゲットを実際に生成します。ターゲットに（明示的または暗黙的な）ルールが適用される前に、そのターゲットの各ディペンデントにおいて再帰が行われます。再帰が行われると、ディペンデントがターゲットになり、独自のディペンデントを生成し、今度はそうしたディペンデントにおいて、ディペンデントのないターゲットが発見されるまで再帰が行われます。`makehce`によって処理されるすべてのターゲットが`makefile`によって明示的なターゲットとして指定されるわけではありません。`makefile`によって明示的なディペンデントになるものもあり、`makehce`がターゲットを再帰的に更新するときに生成される暗黙的なディペンデントになるものもあります。例えば、次の`makefile`が実行されると、

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

次のような作成すべき一連のターゲットが生成されます。

pgm : 2つのディペンデントと1つの明示的なルールがある場合です。

a.obj : .c.objという暗黙的なルールに一致するa.cという暗黙的なディペンデントがある場合です。

a.c : 暗黙的なディペンデントも暗黙的なルールもない場合です。これは、再帰を停止し、a.cファイルの最後の修正日時を戻します。

b.obj : .c.objという暗黙的なルールに一致するb.cという暗黙的なディペンデントがある場合です。

b.c : 暗黙的なディペンデントも暗黙的なルールもない場合です。これは、再帰を停止し、b.cファイルの最後の修正日時を戻します。

これらの定義 (\$@, \$?, \$<, \$*, \$\$@) においてターゲットという単語は、

- makefileで指定されたターゲット
- makehceが再帰を行うときにターゲットになるmakefileに指定された明示的なディペンデント
- makehceが再帰を行うときにターゲットになる（推論ルールおよびファイルを特定した結果として生成される）暗黙的なディペンデント

を指します。

ディペンデントという単語は、

- 特定のターゲットに関するmakefileで指定された明示的なディペンデント
- ターゲットのサフィックスに一致する適切な推論ルールおよび対応するファイルを特定した結果として生成される明示的なディペンデント

を指します。

ターゲットルールは、特定のターゲット名についてユーザの指定するルールと考え、推論ルールは特定のターゲット名クラスについてユーザまたはmakehceの指定するルールと考えれば便利です。また、makehceが明示的および暗黙的なディペンデントで再帰を行うとターゲット名とそれに対応するディペンデント名の値が変わることや、推論ルールが適用されるのは、makefileにターゲットルールが定義されていない暗黙的なディペンデントまたは明示的なディペンデントだけであることを覚えておけば便利です。

第2章 プログラミングコマンド

- \$@** : **\$@**マクロは、現在のターゲットの完全なターゲット名になります。これは、ターゲットと推論ルールの両方として評価されます。
- \$?** : **\$?**マクロは、現在のターゲットに関する古くなったディペンデントのリストであり、実質的には、作成し直されたモジュールです。これは、ターゲットと推論ルールの両方として評価されますが、通常はターゲットルールでしか使用されません。**\$?**は、普通、推論ルールにある1つだけの名前と評価されますが、ターゲットルールにある複数の名前と評価されることもあります。
- \$<** : 推論ルールにおいて、**\$<**は、作成されているターゲットのサフィックスに一致する暗黙的なルールに対応したソースファイル名と評価されます。言い換えれば、これは、ターゲットに関する古くなったファイルです。**.DEFAULT**ルールでは、**\$<**マクロは、現在のターゲット名と評価されます。**\$<**は、推論ルールとしてだけ評価されます。したがって、**.c.obj**ルールでは、**\$<**というマクロは**.c**ファイルと評価されます。次に示すのは、**.c**ファイルから最適化した**.obj**ファイルを作成する場合の例です。

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

または

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

- \$*** : **\$***というマクロは、サフィックスの削除された現在のターゲット名です。これは、推論ルールに関してだけ評価されます。

先にリストした組み込みマクロ（`$@`、`$?`、`$<`、`$*`）に加えて、一般的に使用される他のマクロが `makehce`によって定義されています。こうしたマクロは、`makefile`内のターゲットルールにおいて使用できます。また、`makefile`で再定義することもできます。

`$$@` : `$$@`マクロは、従属行においてだけ意味を持ちます。この書式のマクロは、ディペンデントが実際に処理される時点で評価されるため、ダイナミックディペンデントと呼ばれます。`$$@`は、`$@`がコマンド行で行うことと全く同じものと評価されます。つまり、現在のターゲット名と評価されます。このマクロは、それぞれがソースファイル1つだけを持つ数多くの実行可能ファイルを作成する際に役立ちます。例えば、次のコマンドは、すべてこのルールで作成できます。

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

この`makefile`が`makehce cat echo cmp chown`によって呼び出されると、`makehce`は、一般ルールを使用して各ターゲットを作成し、ターゲットが`cat`である場合には、`$$@`は`cat`と評価され、ターゲットが`echo`である場合には、`echo`と評価されます。

・戻り値

`makehce`は、成功した場合には0を返し、エラーが発生した場合には0よりも大きな値を返します。

第3章 アロケータ

<名前>

svdfa

<形式>

svdfa aname size [オプション]

<機能説明>

svdfaは、指定した大分割領域内に指定した分割領域を確保し、バックアップファイルを生成します。

<引数説明>

aname : 確保する分割領域名称を指定します。

size : 確保する分割領域の大きさを指定します。4096の倍数のバイト数で指定してください。

4096の倍数以外を指定した場合は警告メッセージを表示したあと、4096の倍数に切り上げます。

<オプション>

-p : タスクの格納エリアを確保します。

-s : サブプログラムの格納エリアを確保します。

-gi : 読み書き両用グローバルエリアに初期値ありGLBのエリアを確保します。

-gw : 読み書き両用グローバルエリアに初期値なしGLBのエリアを確保します。

-gr : 読み出し専用グローバルエリアに初期値ありGLBのエリアを確保します。

-cmi : PU間共有メモリエリアに初期値ありCMのエリアを確保します。

-cmw : PU間共有メモリエリアに初期値なしCMのエリアを確保します。

-dcmi : 二重化共有メモリエリアに初期値ありDCMのエリアを確保します。拡張オプションのため、S10VEでは、利用できません。

-dcmw : 二重化共有メモリエリアに初期値なしDCMのエリアを確保します。拡張オプションのため、S10VEでは、利用できません。

-S : アクセス権がシステムであることを指定します。このオプション省略時は、あらかじめ設定された環境変数RSUTYPとします。デフォルトは、ユーザ (RSUTYP=u) です。

-u site : アロケータの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数 "RSSITE" に設定されたサイトに対して処理します。

-f adr : 確保する領域のアドレス (adr) を指定します。大分割領域の先頭からの相対バイトアドレスとして4096の倍数を指定してください。4096の倍数以外を指定した場合は、警告メッセージを表示したあと4096の倍数に切り上げます。なお、このオプションを省略した場合は自動割り付けとなり、最初に検出した空き領域に割り付けます。

<注意事項>

-p、-s、-gi、-gw、-gr、-cmi、-cmw、-dcmi、-dcmwオプションのうちどの指定もなかった場合、-pが指定されたものとします。

-gi、-gr、-cmiオプションのうちどれかを指定して確保した分割領域内に、svdfsコマンドで細分割領域を確保したとき、確保した細分割領域は0で初期化します。

CM領域を確保するときは、必ず-fオプションで確保するCM領域のアドレスを指定してください。-fオプションの指定がない場合はエラーになります。

なお、確保するCM領域のアドレスとサイズはユニット内すべてのサイトで同じにしてください。アドレスとサイズが異なる場合、データが破壊される可能性があります。

表2-2にオプションの組み合わせを示します。

表2-2 svdfaのオプションの組み合わせ

パラメータ	領域種別							
	タスク	サブプログラム	読み書き 両用初期値 ありGLB	読み書き 両用初期値 なしGLB	読み出し 専用初期値 ありGLB	初期値 ありCM	初期値 なしCM	
aname	◎	◎	◎	◎	◎	◎	◎	
size	◎	◎	◎	◎	◎	◎	◎	
オプション	-p (デフォルト)	×	×	×	×	×	×	
	-s	×	◎	×	×	×	×	
	-gi	×	×	◎	×	×	×	
	-gw	×	×	×	◎	×	×	
	-gr	×	×	×	×	◎	×	
	-cmi	×	×	×	×	×	◎	
	-cmw	×	×	×	×	×	×	◎
	-dcmi	×	×	×	×	×	×	×
	-dcmw	×	×	×	×	×	×	×
	-S	○	○	○	○	○	○	○
	-u site	○	○	○	○	○	○	○
-f adr	○	○	○	○	○	◎	◎	

◎：必須 ○：指定できます。×：指定できません。

使用者種別と確保領域の所有者種別の関係を以下に示します。

使用者種別	所有者種別	
	システムエリア	ユーザエリア
システム	○	×
ユーザ	×	○

○：生成できます。×：生成できません。

<終了コード>

svdfaコマンドは次の終了コードを返します。

0 : 正常終了

0以外：異常終了

第3章 アロケータ

<名前>

svdla

<形式>

svdla aname [オプション]

<機能説明>

svdlaは、svdfaで確保した分割領域を削除し、バックアップファイルを削除します。

<引数説明>

aname : 削除する分割領域名称を指定します。

<オプション>

- S : アクセス権がシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）にします。
- u site : アロケータの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

<注意事項>

指定した分割領域内に読み書き両用グローバル、読み出し専用グローバル、PU間共有メモリ、二重化共有メモリの細分割領域が存在する場合は、指定した分割領域内の細分割領域も同時に削除します。使用者種別と削除領域の所有者種別の関係を以下に示します。

使用者種別	削除領域の所有者種別	
	システム	ユーザ
システム	○	○
ユーザ	×	○

○ : 削除できます。× : 削除できません。

<終了コード>

svdlaコマンドは次の終了コードを返します。

- 0 : 正常終了
- 0以外 : 異常終了

<名前>

svdfs

<形式>

svdfs aname sname size [オプション]

<機能説明>

svdfsは、svdfaで確保した分割領域内にグローバルの細分割領域を確保します。
確保した領域は0で初期化します。

<引数説明>

- aname : 細分割する分割領域名称を指定します。
- sname : 確保するグローバルの細分割領域名称を指定します。
- size : 確保する細分割領域のサイズをバイト数で指定します。

<オプション>

- S : アクセス権がシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）にします。
- u site : アロケータの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
- l adr : 確保する細分割領域のアドレス（adr）を指定します。分割領域の先頭からの相対バイトアドレスとして4の倍数を指定してください。4の倍数以外を指定した場合は、警告メッセージを表示したあと、4の倍数に切り上げます。なお、このオプションを省略した場合は自動割り付けとなり、最初に検出された空き領域に割り付けます。
- a align : 細分割領域確保時のアライン数（align）を指定します。2のn乗（ $0 \leq n \leq 12$ ）を指定してください。このオプションを省略した場合のアライン数は2となります。
- t svtype : svtypeで指定するデータ型に従ったアラインとします。svtypeに指定する値とアライン数の関係を表2-3に示します。
- e idxnum : 細分割領域を間接リンクグローバルに割り当てる場合に、確保する分割領域に割り当てるエントリ番号（idxnum）を指定します。このオプションを省略した場合、エントリ番号は割り当てられません。idxnumには1～7934の値が指定できます。

<注意事項>

- -eオプションを指定しないで確保した細分割領域を間接リンクグローバルに割り当てる場合、svirglb コマンドで任意の細分割領域にエントリ番号を割り当てることができます。
- アライン数は使用するデータ型に合わせて決定してください。
- -l、-a、-tオプションを同時に指定することはできません。
- CM領域に確保した分割領域に細分割領域を確保する場合は、アドレスとサイズをCP、HPサイトで同じ値にしてください。

アドレスとサイズが異なる場合、データが破壊される可能性があります。

表2-4に指定可能なオプションの組み合わせを示します。

表2-3 svtypeに指定する値とアライン数の関係

svtype	データ型	アライン数
1	char	0 (1バイト)
2	short	1 (2バイト)
3	long	2 (4バイト)
4	struct	3 (8バイト)
5	float	2 (4バイト)
6	double	3 (8バイト)
7	long double	4 (16バイト)

データ型指定でのアラインは、R700と同様の仕様とします。

表2-4 svdfsのオプションの組み合わせ

パラメータ	領域種別							
	タスク	サブプログラム	読み書き 両用初期値 ありGLB	読み書き 両用初期値 なしGLB	読み出し 専用初期値 ありGLB	初期値 ありCM	初期値 なしCM	
aname	×	×	◎	◎	◎	◎	◎	
sname	×	×	◎	◎	◎	◎	◎	
size	×	×	◎	◎	◎	◎	◎	
オプション	-S	×	×	○	○	○	○	○
	-u site	×	×	○	○	○	○	○
	-l adr	×	×	○	○	○	○	○
	-a align	×	×	○	○	○	○	○
	-t svtype	×	×	○	○	○	○	○
-e index	×	×	○	○	○	○	○	

◎：必須 ○：指定できます。×：指定できません。

使用者種別と確保される細分割領域の所有者種別の関係を以下に示します。

使用者種別	細分割領域の所有者種別	
	システム	ユーザ
システム	○ (システム)	○ (ユーザ)
ユーザ	×	○ (ユーザ)

○：確保できます。×：確保できません。

() 内は確保される細分割領域の所有者種別を示します。

<終了コード>

svdfsコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

第3章 アロケータ

<名前>

svdls

<形式>

svdls sname [オプション]

<機能説明>

svdlsは、svdfsで確保した細分割領域を削除します。

<引数説明>

sname : 削除する細分割領域名称を指定します。

<オプション>

-S : アクセス権がシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）にします。

-u site : アロケータの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

<注意事項>

使用者種別と削除領域の所有者種別の関係を以下に示します。

使用者種別	削除領域の所有者種別	
	システム	ユーザ
システム	○	○
ユーザ	×	○

○ : 削除できます。× : 削除できません。

<終了コード>

svdlsコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

<名前>

svdfv

<形式>

svdfv ename value [オプション]

<機能説明>

svdfvは、バリュ情報の外部参照情報を登録します。

<引数説明>

ename : 登録する外部名称を指定します。

value : 外部名の取る値 ($-2^{31} \leq \text{value} \leq 2^{31}-1$) を指定します。

<オプション>

-S : アクセス権がシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）にします。

-u site : アロケータの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

<注意事項>

バリュ値 (value) は整数型データとして扱い、 $-2^{31} \leq \text{value} \leq 2^{31}-1$ の範囲外を指定した場合、エラーとなります。

なお、実行時のバリュ値は、使用言語のバリュ名称 (ename) のデータ型に依存します。

使用者種別と確保されるバリュの所有者種別の関係を以下に示します。

使用者種別	所有者種別	
	システムバリュ	ユーザバリュ
システム	○	×
ユーザ	×	○

○ : 生成できます。 × : 生成できません。

<終了コード>

svdfvコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

第3章 アロケータ

<名前>

svdlv

<形式>

svdlv ename [オプション]

<機能説明>

svdlvは、svdfvで登録された外部参照情報を削除します。

<引数説明>

ename : 削除する外部名称を指定します。

<オプション>

- S : アクセス権がシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）にします。
- u site : アロケータの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

<注意事項>

使用者種別と削除バリュの所有者種別の関係を以下に示します。

使用者種別	削除バリュの所有者種別	
	システム	ユーザ
システム	○	○
ユーザ	×	○

○ : 削除できます。 × : 削除できません。

<終了コード>

svdlvコマンドは次の終了コードを返します。

- 0 : 正常終了
- 0以外 : 異常終了

第4章 ローダ

<名前>

svload

<形式>

svload [オプション] file...

<機能説明>

svloadは、オブジェクトファイル、ライブラリを結合し、プログラム、サブプログラム、データを指定した名称で開発環境に登録し、バックアップファイルに格納します。

<引数説明>

file : 結合するオブジェクトファイル、ライブラリを指定します。ファイルは複数指定できます。

<オプション>

- S : 処理モードがシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）として扱います。
- u site : ローダの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE” に設定されたサイトに対して処理します。
- C n : プログラム、サブプログラムの格納先頭アドレス (n) を指定します。アドレスは、プログラムの場合4096の倍数を、サブプログラムの場合32の倍数を指定してください。4096または32の倍数以外を指定した場合、警告メッセージを表示したあと、4096または32の倍数に切り上げます。
- p n : エリア内のローディング位置を相対バイトアドレス (n) で指定します。プログラム、サブプログラムのときに有効です。このオプションを省略した場合は自動登録となります。-Cオプションと同時指定はできません。アドレスは、プログラムの場合4096の倍数を、サブプログラムの場合32の倍数を指定してください。4096または32の倍数以外を指定した場合、警告メッセージを表示したあと、4096または32の倍数に切り上げます。
- a area : プログラム、サブプログラムのローディングエリア (area) を指定します。プログラム、サブプログラムのときはこのオプションは省略できません。
- +P : プログラム (タスク) としてローディングすることを指定します。
- +I : 間接リンクサブルーチン (IRSUB) としてローディングすることを指定します。
- +U : 組み込みサブルーチン (ULSUB) としてローディングすることを指定します。
- +D : グローバル、CMデータとしてローディングすることを指定します。データ種別は属する分割領域の属性に従います。
- +B : データジェネレータ (svdatagen) で生成したバイナリデータを、グローバル、CMデータとしてローディングすることを指定します。データ種別は属する分割領域の属性に従います。fileにはsvdatagenで生成したバイナリデータ (*.bin) を指定します。

第4章 ローダ

- M n : マルチタスクを生成する場合に指定します。nはマルチタスクの個数を示します。nは2~128の範囲としてください。
- m n[n...]: IRSUBのマルチエン트리ローディングの指定であり、エン트리ポイントとしたいエン트리名(n)を指定します。このオプションは、+I指定時にだけ指定できます。
- Z : ロードモジュールのtext、data、bss、stackセクションのサイズを出力します。このオプションはプログラムを登録しません。
この指定と-Pオプションを組み合わせることで、実際にロードしなくてもリンケージマップリストを生成することができます。
また、-w n [,m]オプションと組み合わせることで、呼び出しているIRSUBが使用するスタックサイズが増加した場合に、呼び出し側の再ロード(スタックサイズの拡張) 可否をチェックすることができます。この場合、n、mには前回登録時に指定した値と同じ値を指定してください。
- d : バックアップファイルに格納後、ロードモジュールファイルを削除しません。
ロードモジュールファイルは、サイトディレクトリの下PGM、SUB、GLB下に生成されません。
- s : スタック使用量情報ファイルを生成します。スタック使用量情報ファイルは、サイトディレクトリ下にあるPGM/SUBディレクトリに生成されます。
- llib : 結合するライブラリ(lib)を指定します。なお、libcpms.libとlibsh4nbmdn.libは自動で結合しません。
- P [file] : プログラムのリンケージマップリストを出力します。fileを省略した場合は、サイトディレクトリの下PGM、SUB、GLB下に、ロードするプログラム名、サブプログラム名の後ろに、.mapを付加したファイル名で生成します。初期値データのロードの場合は、ロードした先頭のsarea名.mapの名称で生成します。
環境変数LOADHR_FORCE_MAP=YESを設定しておくこと、このオプションを指定しない場合でもリンケージマップリストを生成します。環境変数LOADHR_FORCE_MAPとこのオプションの両方を指定した場合は、両方の指定が有効となります。
- o obj : 作成するプログラム名(obj)を指定します。サブプログラムの場合、指定した名前がサブプログラム名となります。サブプログラム名とプログラム内の関数は一致させてください。
- E n : プログラム、サブプログラムを結合時、冗長なバイト数(n)を加味して格納します。
これは、将来、プログラム、サブプログラムを入れ替え、容量が増えた場合に有効です。
- r : 指定したエリアにプログラム、サブプログラムが入れ替え可能かどうかをチェックします。
格納アドレスは入れ替え前のプログラム、サブプログラムと同じアドレスを指定してください。

-w n [m] : スタックエリアの大きさをバイト長で指定します。

このオプションはプログラム、サブプログラムの場合、省略できません。

nには自分自身が使用するスタックエリアの大きさを指定します。

mには実際に確保するスタックサイズを指定します。**m**を省略するとスタックサイズは自分自身のスタックサイズ (**n**) に、呼び出しているIRSUBが使用する最大スタックサイズを加算した値で確保します。

確保するスタックサイズ (**m**) が、**n**+呼び出しているIRSUBの最大スタックサイズより小さい場合はWarningメッセージを出力します。

n、**m**は、0~8388608 (0x800000) かつ8の倍数を指定してください。8の倍数以外を指定した場合は、警告メッセージを表示したあと8の倍数に切り上げて処理します。

-Xref : プログラムのリンケージマップリスト内にクロスリファレンス情報を出力します。このオプションは、**-P**指定時にだけ指定できます。

<ライブラリのサーチパス>

ローダのライブラリのサーチパス (**-l**オプションで指定されたライブラリのサーチ順) は、shcコンパイラパッケージの最適化リンケージエディタの入力ファイルを検索する順序に従います。

最適化リンケージエディタの入力ファイルの検索順は下記となります。

(1) カレントディレクトリ

(2) RPD動作環境設定ファイルのHLNK_DIRで指定されたディレクトリ

RPD動作環境設定ファイルのHLNK_DIRには複数のパスが設定できます。

複数のパスを設定する場合は、セミコロンで区切って指定してください。

<注意事項>

組み込みサブルーチンのスタック領域はシステムの領域を使用します。組み込みサブルーチンのスタック容量は512バイト以内としてください。

IRSUB、マルチタスクはリエントラントなプログラムでなければならないため、BSSエリアを持つことはできません。BSSエリアを持つIRSUB、マルチタスクをロードした場合は、警告メッセージを表示します。

タスクはプログラムの先頭から実行されます。**main**から実行されるとは限りません。

プログラムのロード時は、メインルーチンのオブジェクトファイルを最初に指定してください。

R700のcchcでコンパイルしたオブジェクトは、S10VEにはローディングしないでください。

<スタック容量>

プログラムでスタックエリアを使用する場合、スタックエリアの容量を指定してください。

第4章 ローダ

<システム/ユーザの外部参照チェック>

システムからユーザの情報を参照できません。ユーザからシステムのサブプログラムだけを参照できます。参照できる組み合わせを下表に示します。

参照 \ 被参照		サブプログラム		グローバル (CM含む)		バリュ	
		S	U	S	U	S	U
プログラム	S	○	×	○	×	○	×
	U	○	○	×	○	×	○
サブプログラム	S	○	×	○	×	○	×
	U	○	○	×	○	×	○
グローバル (CM含む)	S	○	×	○	×	○	×
	U	○	○	×	○	×	○

S : システム U : ユーザ

○ : 参照できます。 × : 参照できません。

(注) グローバルからサブプログラムを参照する場合、名称に対応する間接リンクテーブルの番号がグローバル内に埋め込まれます。グローバルからグローバルを参照する場合、絶対アドレスがグローバル内に埋め込まれます。グローバルからバリュを参照する場合、バリュ値がグローバル内に埋め込まれます。

<注意事項>

- IRSUBやマルチタスクは、リエントラントなプログラムであるため、BSSエリアを持つことはできません。BSSエリアを持つと、警告メッセージが出力されます。
- ロードするプログラム内に複数グローバルが存在する場合、ローカルなラベルのアドレスは解決されません。この場合、グローバルを複数のファイルに分割しローディングしてください。
- タスクはプログラムの先頭から実行されます。mainから実行されるとは限りません。
- オプションの組み合わせは、下記となります。

		-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
プログラム	+P	◎	◎	◎	○	○	○	○	○	○	○	○	○	○	○	×	○	○
IRSUB	+I	◎	◎	◎	○	○	○	○	○	×	○	○	○	○	○	○	○	○
ULSUB	+U	◎	◎	◎	○	○	○	○	○	×	○	○	○	○	○	×	○	○
データ	+D	×	×	×	×	○	○	×	×	×	×	○	○	×	×	×	○	×
バイナリデータ	+B	×	×	×	×	○	○	×	×	×	×	×	×	×	×	×	×	×

◎ : 必須指定 ○ : 選択指定 × : 指定不可

<終了コード>

svloadコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

<クロスリファレンス情報>

シンボルの参照情報（クロスリファレンス情報）を出力します。クロスリファレンス情報の出力例を以下に示します。

```

** 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 ユニット番号。オブジェクト単位の識別番号。
- ② オブジェクト名。リンク時の入力指定順に表示されます。
- ③ シンボル名。セクションごとに昇順に出力されます。
- ④ シンボルの配置アドレス。
- ⑤ 外部シンボルを参照している場所のアドレスを表します。出力形式は以下のようになります。
<ユニット番号> (<アドレスまたはセクション内オフセット> : <セクション名>)

<スタック容量算出方法>

スタックエリアの使用量は、プログラムを構成する各関数のスタック使用量を算出し、関数の呼び出し関係から全体のスタック使用量を算出します。

(1) 各関数の使用するスタック領域の算出

各関数の使用するスタック領域の大きさは、コンパイラが出力するオブジェクトリストのframe sizeから分かります。

以下に具体例を示します。

■ ソースコード

```
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;
    }
}
```

■ オブジェクトリスト

SCT	OFFSET	CODE	C LABEL	INSTRUCTION	OPERAND	COMMENT
P	00000000		_h:			; function: h ; <u>frame size=12</u>
	00000000	2FE6		MOV.L	R14, @-R15	
	00000002	4F22		STS.L	PR, @-R15	

上記の例では関数hの使用するスタック領域のサイズは、オブジェクトリスト中の項目“COMMENT”の“frame size”の値12バイトとなります。

(2) 呼び出し関係からの全体スタック容量の算出

関数呼び出しの関係から使用するスタック領域のサイズを算出します。

関数呼び出しの関係からのスタック使用量算出方法の例を図2-1に示します。



図2-1 関数呼び出しの関係とスタック使用量

上記の場合、関数fを介して関数gが呼ばれた場合のスタック領域のサイズは、表2-5に示すようになります。

表2-5 スタックサイズの計算例

呼び出し経路	スタックサイズ (バイト)
main(24)→f(32)→g(24)	80
main(24)→g(24)	48

このように、呼び出しレベルの一番深い関数についてスタック容量のサイズを計算し、その最大サイズのスタック領域を最低制限り当てなければなりません。

標準ライブラリ関数を使用する場合には、ライブラリ関数を使用するスタックサイズも考慮する必要があります。標準ライブラリ関数の使用するスタックサイズは、「付録H ライブラリの使用するスタックサイズ一覧」を参照してください。

再帰的に呼び出される関数のスタックサイズは、「関数のスタックサイズ×再帰的に呼び出される回数の最大値」で算出してください。

また、ソースプログラム上でライブラリ関数を使用していなくても、プログラムの実行に必要な実行時ルーチンがリンクされている場合があります。この場合は、実行時ルーチンの使用するスタックサイズも考慮する必要があります。実行時ルーチンの使用するスタックサイズは、次ページのスタック使用量確認方法に示すスタック解析ツールで確認することができます。

<スタック使用量確認方法>

プログラム、サブプログラムのロード時に-sオプションを指定すると、スタック使用量情報ファイルを生成することができます。

ローダで生成したスタック使用量情報ファイルをコンパイラパッケージ付属のスタック解析ツールで解析することにより、プログラム、サブプログラムの全体スタック使用量を求めることができます。

● スタック使用量情報ファイルの生成

プログラム、サブプログラムのロード時に-sオプションを指定すると、スタック使用量情報ファイルを生成します。

スタック使用量情報ファイルは、サイトディレクトリ下にあるPGM/SUBディレクトリ下に、プログラム、サブプログラムの名称に、“_sni”を付加したファイル名で生成します。

(例) svload +P -o pgm01 -a tskarea -w 4096 pgm01.obj -s

上記の例では、サイトディレクトリ¥PGM¥pgm01_sniが生成されます。

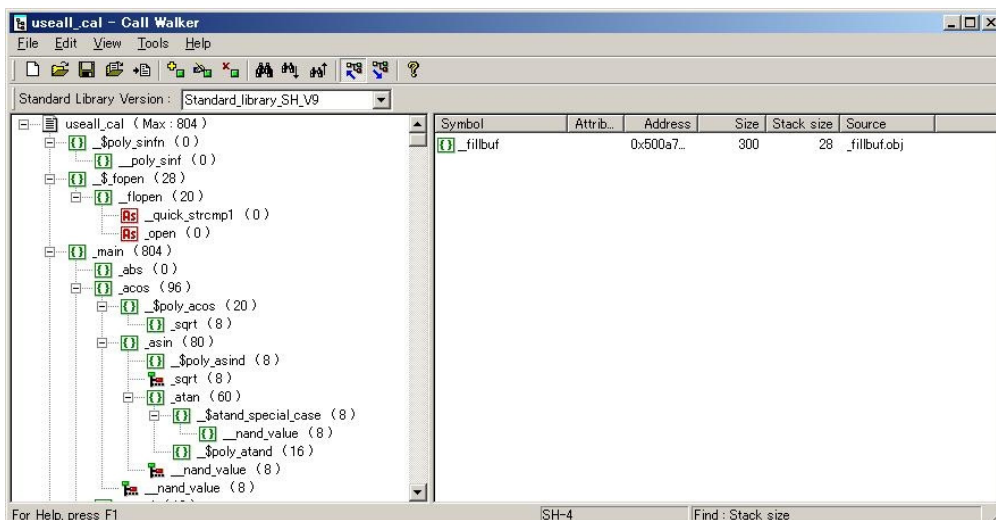
● スタック解析ツール使用方法

以下に示す操作を行うとスタック解析ツールを起動し、プログラム、サブプログラムのスタック使用量を表示することができます。

スタック解析ツールの使用方法の詳細は、コンパイラパッケージのマニュアルおよびスタック解析ツールのヘルプを参照してください。

- ① Windows® 7の場合は、Windows®の [スタート] メニューから [Renesas] → [High-performance Embedded Workshop] → [Call Walker] を選択、Windows® 10の場合は、Windows®の [スタート] メニューから [Renesas] → [Call Walker] を選択し、スタック解析ツールを起動します。
- ② スタック解析ツールの [File] メニューから [Import Stack file...] を選択し、表示されたダイアログボックスの「ファイル名(N)」にローダで生成したスタック使用量情報ファイルを指定し、[開く(O)] ボタンをクリックします。

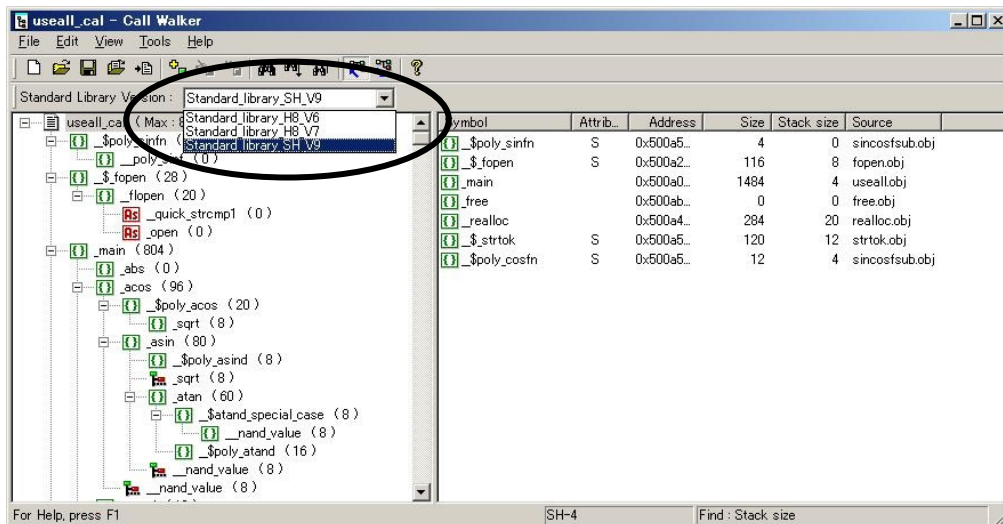
■ スタック解析ツールの表示例



<注意事項>

スタック解析ツールの [Standard Library Version] は [Standard_Library_SH_V9] を使用してください。

Standard_Library_H8_V6およびStandard_Library_H8_V7は、CPU種別が異なるため使用しないでください。



- ロードしたプログラム、サブプログラムのスタック使用量解析時の注意点

スタック解析ツールで算出するスタック使用量では、アセンブラで記述されたプログラム、サブプログラムのスタックサイズは0バイトとして表示されます。このため、RPDPでロードしたプログラム、サブプログラムのスタック使用量をスタック解析ツールで解析する場合には、以下に示す注意事項があります。

- memcpy()のスタックサイズ

ロードしたプログラム、サブプログラムでは、memcpy()関数はC標準ライブラリではなくCPMSライブラリのmemcpy()関数がリンクされます。スタック解析ツールではCPMSライブラリのmemcpy()が使用するスタックサイズは0として表示されますが、CPMSライブラリのmemcpy()はスタックを28バイト使用します。

スタック解析ツールの [Edit] メニューから [Modify] コマンドを使用してmemcpy()関数のスタックサイズを28バイトに変更し、スタック使用量を再計算してください。

ロードしたプログラム、サブプログラムがmemcpy()を使用しているか否かは、スタック解析ツールの検索機能でmemcpyを検索することで識別することができます。

■ IRSUBのスタックサイズ

ロードしたプログラム、サブプログラムからIRSUBを呼び出している場合、スタック解析ツールではIRSUBが使用するスタックは0バイトとして算出されます。呼び出しているIRSUBが使用するスタックサイズを含んだスタック使用量を算出する場合は、`memcpy()`と同様に、スタック解析ツールの [Edit] メニューから [Modify] コマンドを使用してスタックサイズを変更し、スタック使用量を再計算してください。

IRSUBが使用するスタックサイズは、IRSUB自身のロード時にスタック使用量情報ファイルを生成し、スタック解析ツールで解析することで求めることができます。

ローダに指定する自分自身が使用する（呼び出しているIRSUBを含まない）スタックサイズを求めるためにはIRSUBのスタックサイズの再計算は必要ありません。

■ その他アセンブラで記述されたプログラム、サブプログラムのスタックサイズ

アセンブラで記述されたプログラム、サブプログラムのスタックサイズも同様に、スタック解析ツールの [Edit] メニューから [Modify] コマンドを使用してスタックサイズを変更し、スタック使用量を再計算してください。

(3) svloadへのスタックサイズの指定方法

svloadで指定するスタックサイズは下記としてください。

-w n m

n : ロードするプログラム、サブプログラム自身が使用するスタックサイズ

m : ロードするプログラム、サブプログラムから呼び出しているIRSUBが使用するスタックサイズを含んだサイズ

プログラムのロード時にこの値を指定すると指定サイズでスタックを確保します。この値を省略すると、n+呼び出しているIRSUBが使用するスタックサイズの値でスタックを確保します。

この値を省略すると、n+呼び出しているIRSUBが使用するスタックサイズの値でスタックを確保します。

mの値にn+呼び出しているIRSUBが使用するスタックサイズの値よりも小さい値を指定した場合は、ローダは下記のWarningメッセージを出力します。

Warning : Stack size (name) = xxxx (zzzz) byte [Max refered (sname) size = yyyy byte] Err

name : ロードするプログラムまたはサブプログラム名

xxxx : 呼び出しているIRSUBが使用するスタックサイズを含んだサイズ

zzzz : mに指定したスタックサイズ

sname : 呼び出しているIRSUBの名称 (最大スタックサイズのもの)

yyyy : snameの使用するスタックサイズ

それぞれのプログラム、サブプログラムが使用するスタックサイズは、「(2) 呼び出し関係からの全体スタック容量の算出」に示す手順に従って算出してください。

以下に「図2-1 関数呼び出しの関係とスタック使用量」に示す、呼び出し関係のプログラムを例にsvloadに指定するスタックサイズの指定例を示します。

● IRSUBを使用していない場合

main : プログラムのメイン

f : ISUB

g : ISUB

の場合は下記のように指定します。

svload +P -o main -w 80 4096

ロードするプログラム自身が使用するスタックサイズの最大値は80バイトであるため、nの値には80を指定します。

mには実際に確保するスタックのサイズを指定してください。mの値を省略するとスタックサイズは80バイトとなります。プログラムの改修時に使用するスタックサイズが増加したり、IRSUBを呼び出すようになった場合に備え余裕を持って指定してください。

プログラムのスタック/BSSは、テキスト/データとは別ページに配置されるため、スタックサイズを大きめに指定しても、BSSサイズ+スタックサイズが4096バイトの境界を超えなければプログラムサイズは増加しません。

● IRSUBを使用している場合

main : プログラムのメイン

f : IRSUB

g : IRSUB

の場合は下記のように指定します。

```
svload +I -o g ..... -w 24 .....  
svload +P -o main ..... -w 56 4096 .....
```

IRSUBとプログラムは別々にロードします。スタックサイズはIRSUB、プログラムの両方に指定します。

• IRSUBのロード時

IRSUB (g) が使用するスタックサイズは24バイトであるため、スタックサイズには24バイトを指定します。mの値は省略することを推奨します。mの値を指定した場合は、IRSUBの呼び出し側で計算するIRSUBのスタックサイズはmが使用されます。

• プログラムのロード時

プログラム自身が使用するスタックサイズは56バイトであるため、nの値には56を指定します。

mの値は56と呼び出しているIRSUB (g) の使用するスタックサイズ24を加算した80バイト以上を指定してください。IRSUBを使用しない場合と同様に、プログラムの改修に備えてmの値は余裕を持って指定してください。

mの値に80バイトよりも小さい値を指定した場合は、ローダは下記のWarningメッセージを出力します。

```
Warning : Stack size (main) = 80 (m) byte [Max refered (g) size = 24 byte] Err
```

mを省略すると、ローダがIRSUB (g) が使用するスタックサイズ24を加算し、80バイトでスタックを確保します。

● IRSUBからIRSUBを使用している場合

main : プログラムのメイン

f : IRSUB

g : IRSUB

の場合は下記のように指定します。

```
svload +I -o g ..... -w 24 .....
svload +I -o f ..... -w 32 .....
svload +P -o main ..... -w 24 4096 .....
```

• IRSUB (g) のロード時

IRSUB (g) が使用するスタックサイズは24バイトであるため、スタックサイズには24バイトを指定します。mの値は省略することを推奨します。mの値を指定した場合は、IRSUBの呼び出し側で計算するIRSUBのスタックサイズはmが使用されます。

• IRSUB (f) のロード時

IRSUB (f) が使用するスタックサイズは32バイトであるため、スタックサイズには32バイトを指定します。mの値を省略した場合、IRSUBの呼び出し側で計算するIRSUB (f) のスタックサイズは、IRSUB (f) から呼び出しているIRSUB (g) のスタックサイズを加算した56バイトとなります。mの値を指定した場合は、IRSUBの呼び出し側で計算するIRSUB (f) のスタックサイズはmが使用されます。mの値は省略することを推奨します。

mの値に56バイトよりも小さい値を指定した場合は、ローダは下記のWarningメッセージを出力します。

```
Warning: Stack size (f) = 56 (m) byte [Max refered (g) size = 24 byte] Err
```

• プログラムのロード時

プログラム自身が使用するスタックサイズは24バイトであるため、nの値には24を指定します。

mの値は24と呼び出している最大スタック容量のIRSUB (f) の使用するスタックサイズ56を加算した80バイト以上を指定してください。IRSUBを使用しない場合と同様に、プログラムの改修に備えてmの値は余裕を持って指定してください。

mの値に80バイトよりも小さい値を指定した場合は、ローダは下記のWarningメッセージを出力します。

```
Warning: Stack size (main) = 80 (m) byte [Max refered (f) size = 56 byte] Err
```

mを省略すると、ローダがIRSUB (f) が使用するスタックサイズ56を加算し、80バイトでスタックを確保します。

第4章 ローダ

<ライブラリの整合性チェック>

コンパイル時に`shc -denormalization=off -round=zero`オプション指定でコンパイルした場合は、ロード時にライブラリ`libsh4nbmzz.lib(-lsh4nbmzz)`を指定しなければなりません。`libsh4nbmzz.lib`を指定しない場合、ローダは下記エラーメッセージを出力します。

```
svload : Error : Undefined symbols
svload :      _use_libsh4nbmzz
```

同様にコンパイル時に`shc -denormalization=off -round=zero`オプション指定でコンパイルしていないときに、`libsh4nbmzz.lib`が指定された場合は、下記のエラーメッセージを出力します。

```
svload : Error : Undefined symbols
svload :      _use_libsh4nbmdn
```

また、両方のオブジェクトを混在させて、`-lsh4nbmzz`と`-lsh4nbmdn`の両方を指定した場合は、下記のエラーメッセージを出力します。

```
svload : rpdpload: Inconsistent object was mixed (NO:2004-25)
```

<名前>

svdload

<形式>

svdload pname [オプション]

<機能説明>

svdloadは、svloadコマンドで登録したプログラム、サブプログラムを、開発環境下から削除します。ただし、バックアップファイルの0クリアはしません。

<引数説明>

pname : 削除するプログラム、サブプログラム名称を指定します。

<オプション>

- S : 処理モードがシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）として扱います。
- u site : ローダの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
- +P : プログラム（タスク）の削除を指定します。
- +I : 間接リンクサブルーチン（IRSUB）の削除を指定します。
- +U : 組み込みサブルーチン（ULSUB）の削除を指定します。

<終了コード>

svdloadコマンドは次の終了コードを返します。

- 0 : 正常終了
- 0以外 : 異常終了

第4章 ローダ

<名前>

svcomp

<形式>

svcomp [オプション] file...

<機能説明>

svcompはローダで登録済みのプログラム、サブプログラム、データのバックアップファイルの内容とロードモジュールを比較して、その結果を出力します。

<引数説明>

file : 結合するオブジェクトファイル、ライブラリを指定します。ファイルは複数指定できます。

<オプション>

- S : 処理モードがシステムであることを指定します。このオプション省略時のアクセス権はあらかじめ設定されたデフォルト（環境変数RSUTYP）として扱います。
- u site : ローダの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
- C n : svcompでは意味を持ちません。
- p n : svcompでは意味を持ちません。
- a area : svcompでは意味を持ちません。
- +P : プログラム（タスク）としてローディングしたものを比較する指定です。
- +I : 間接リンクサブルーチン（IRSUB）としてローディングしたものを比較する指定です。
- +U : 組み込みサブルーチン（ULSUB）としてローディングしたものを比較する指定です。
- +D : グローバル、CMデータとしてローディングしたものを比較する指定です。
データ種別は属する分割領域の属性に従います。
- +B : グローバル、CMデータとしてローディングしたものを、データジェネレータ（svdatagen）で生成したバイナリデータと比較する指定です。fileにはsvdatagenで生成したバイナリデータ（*.bin）を指定します。
- M n : svcompでは意味を持ちません。
- m n[n...] : svcompでは意味を持ちません。
- Z : svcompでは意味を持ちません。
- d : svcompでは意味を持ちません。
- llib : 結合するライブラリ（lib）を指定します。なお、libcpms.libとlibsh4nbmdn.libは自動で結合します。
- P [file] : svcompでは意味を持ちません。
- o obj : 比較するプログラム名（obj）を指定します。
- E n : svcompでは意味を持ちません。

- r : svcompでは意味を持ちません。
- w n [m] : svcompでは意味を持ちません。
- Xref : svcompでは意味を持ちません。

<ライブラリサーチパス>

ローダのライブラリのサーチパス (-lオプションで指定されたライブラリのサーチ順) は、shcコンパイラパッケージの最適化リンケージエディタの入力ファイルを検索する順序に従います。

最適化リンケージエディタの入力ファイルの検索順は下記となります。

(1) カレントディレクトリ

(2) RPDP動作環境設定ファイルのHLNK_DIRで指定されたディレクトリ

RPDP動作環境設定ファイルのHLNK_DIRには複数のパスが設定できます。

複数のパスを設定する場合はセミコロンで区切って指定してください。

<終了コード>

svcompコマンドは次の終了コードを返します。

0 : 相違なし

1 : 相違あり

101 : コマンドオプション指定誤り

上記以外 : コンペア失敗

<svcompの表示結果>

svcompコマンドは比較結果に相違がない場合とある場合で、それぞれ下記のメッセージを出力します。

(1) 比較結果に相違がない場合

svcompコマンドは比較結果に相違がない場合、下記のメッセージを出力します。

compare OK (type = X name = xxx)

X : 比較するリソースのタイプを表します。

pgm : プログラム

irsub : IRSUB

ulsub : 組み込みサブプログラム

data : データ (GLB、CM)

xxx : 比較するリソースの名称を表します。

データの比較の場合は、**data**と表示します。

(2) 比較結果に相違がある場合

svcompコマンドは比較結果に相違がある場合、下記のメッセージを出力します。

<表示フォーマット>

svcompコマンドの比較結果の相違点表示内容は、以下のフィールドから構成されます。

(1) ヘッダ

(2) 詳細情報

(3) フッタ

プログラム、サブプログラムの場合のフォーマットを図2-2に示します。データの場合のフォーマットを図2-3に示します。斜体で表記している部分は、実行環境によって変わる部分です。

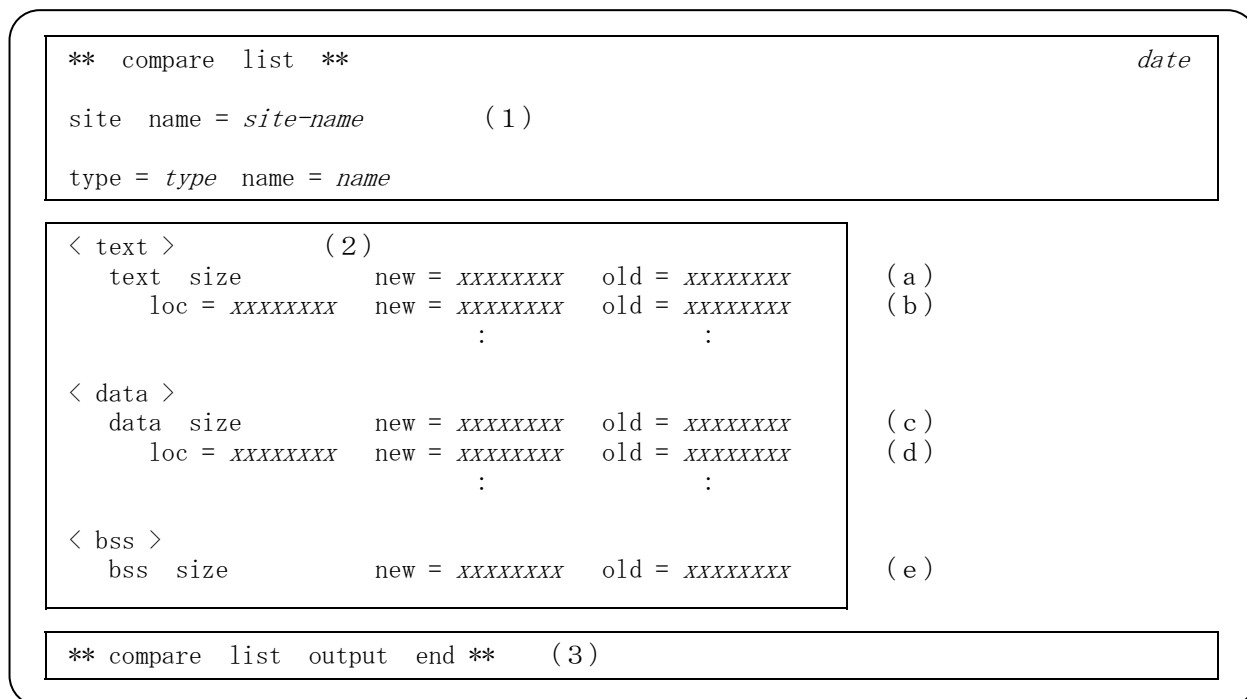


図2-2 svcomp (プログラム、サブプログラム) のフォーマット

各フィールドの詳細は以下のとおりです。

(1) ヘッダフィールド

date : svcompコマンドを起動した時刻を表示します。

site-name : 処理対象としているサイト名称を表示します。

type : 比較対象の種別を示します。

- pgm* - タスク用プログラム
- irsub* - 間接リンクサブルーチン
- ulsub* - 組み込みサブルーチン

name : 比較対象の名称です。

(2) 詳細情報フィールド

このフィールドには、テキスト、データ、BSSそれぞれのサイズの新/旧比較結果と、テキスト、データの内容の新/旧比較結果を表示します。新/旧サイズが異なる場合の比較は、小さい方のサイズにあわせて行います。

(a)、(c)、(e) : テキスト、データ、BSSのサイズの新/旧比較結果を16進数で表示します。比較結果が同じである場合には表示しません。

(b)、(d) : テキスト、データの内容の新/旧比較結果を16進数で表示します。比較結果が同じである場合には表示しません。loc = の直後に表示される16進数は、比較内容が異なっているアドレスであり、テキスト、データ先頭からの相対値です。

(3) フッタフィールド

比較完了を示します。


```

** 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)

```

図2-3 svcomp (GLB、CM) のフォーマット

各フィールドの詳細は以下のとおりです。

(1) ヘッダフィールド

date : svcompコマンドを起動した時刻を表示します。

site-name : 処理対象としているサイト名称を表示します。

(2) 詳細情報フィールド

このフィールドには、GLB、CMそれぞれのサイズの新／旧比較結果と、テキスト、データの内容の新／旧比較結果を表示します。新／旧サイズが異なる場合の比較は、小さい方のサイズにあわせて行います。

(a) : GLB、CMのサイズの新／旧比較結果を16進数で表示します。比較結果が同じである場合には表示しません。

(b) : GLB、CMの内容の新／旧比較結果を16進数で表示します。比較結果が同じである場合には表示しません。loc =の直後に表示される16進数は、比較内容が異なっているアドレスであり、データ先頭からの相対値です。

(3) フッタフィールド

比較完了を示します。

第5章 ビルダ

<名前>

svctask

<形式>

svctask pname tname tn [オプション]

<機能説明>

svctaskは、ローダで格納されたプログラムをリソースとして、タスクを生成します。

<引数説明>

pname : 生成すべきタスクのリソースとなるプログラム名を指定します。

tname : 生成すべきタスク名を指定します。

tn : タスク番号を指定します。ユーザタスクの場合1~224まで、システムタスクの場合は225~300までが指定できます。指定タスク番号が使用中の場合はエラーとなります。

<オプション>

-S : 処理モードがシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）として扱います。

-u site : ビルダの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

-l lvl : タスクの初期起動時の実行レベル（lvl）を指定します。ユーザタスクの場合は4~27まで、システムタスクの場合は0~31までが指定できます。このオプション省略時、ユーザタスクの場合は、lvl=27、システムタスクの場合は、lvl=0が指定されたものとみなします。

-r n : リソースとなるプログラムから複数のタスクを生成する場合の、使用スタックエリア番号を指定します。この数値は、ロードコマンドで指定するマルチタスクの個数（-M n）を超えて指定することはできません。このオプション省略時は、未使用のスタックエリア番号の最小値が指定されたものとみなします。

<注意事項>

ユーザタスクとはTN=1~224で登録したタスク、システムタスクとはTN=225~300で登録したタスクを示します。

RSUTYP=sで登録する場合、システムタスクを生成することができます。

RSUTYP=uで登録する場合、-Sオプションを指定することでシステムタスクが生成できます。

RSUTYP=uで-Sオプションを指定しない場合、システムタスクは生成できません。

タスク種別とオプションの組み合わせを次ページに示します。

第5章 ビルダ

タスク種別 パラメータ		シングルタスク	マルチタスク
		pname	◎
tname	◎	◎	
tn	◎	◎	
オプション	-u site	○	○
	-l lvl	○	○
	-S	○	○
	-r n	×	◎

◎：必須 ○：指定できます。×：指定できません。

使用者種別とプログラム所有者種別との関係を以下に示します。

使用者種別	プログラム所有者種別	
	システム	ユーザ
システム	○	○
ユーザ	×	○

○：タスク生成できます。×：タスク生成できません。

<終了コード>

svctaskコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

<名前>

svdtask

<形式>

svdtask tname [オプション]

<機能説明>

svdtaskは、svctaskで生成したタスクを削除します。

<引数説明>

tname : 削除すべきタスク名を指定します。

<オプション>

-S : 処理モードがシステムであることを指定します。このオプション省略時のアクセス権は、あらかじめ設定されたデフォルト（環境変数RSUTYP）として扱います。

-u site : ビルダの処理対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

<注意事項>

使用者種別と削除タスクの所有者種別との関係を以下に示します。

使用者種別	削除タスク所有者種別	
	システム	ユーザ
システム	○	○
ユーザ	×	○

○ : タスク削除できます。× : タスク削除できません。

<終了コード>

svdtaskコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

第5章 ビルダ

<名前>

svbuild (間接リンクサブプログラムの登録)

<形式>

svbuild name -ir -e irno [オプション]

<機能説明>

svbuildは、間接リンクサブプログラムを登録します。

<引数説明>

name : 間接リンクサブプログラム名称 (マルチエントリローディング時はエントリ名称) を指定します。

-ir -e irno : 間接リンクサブプログラムの登録番号 (irno) を指定します。irnoは1~7935の範囲で指定できます。

<オプション>

-u site : ビルダの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

種別 パラメータ		間接リンク サブプログラム
		◎
オプション	-u site	○

◎ : 必須 ○ : 指定できます。

<終了コード>

svbuildコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

<名前>

svbuild (組み込みサブルーチンの登録)

<形式>

svbuild name point en -ul [オプション]

<機能説明>

svbuildは、組み込みサブルーチンを登録します。

<引数説明>

name : 組み込みサブルーチン名称を指定します。

point en -ul : 組み込みサブルーチンの組み込み箇所 (point) およびエントリ番号 (en) を指定します。エントリ番号として1~4を、組み込み箇所には以下の文字列を指定してください。ただし、エントリ番号1はOS用に、エントリ番号2はNXACP用に予約されています。

CPES : CPES組み込みサブルーチンを示します。

IES : IES組み込みサブルーチンを示します。

EAS : EAS組み込みサブルーチンを示します。

INS : INS組み込みサブルーチンを示します。

EXS : EXS組み込みサブルーチンを示します。

ABS : ABS組み込みサブルーチンを示します。

PCKS : PCKS組み込みサブルーチンを示します。

MODES : MODES組み込みサブルーチンを示します。

WDTES : WDTES組み込みサブルーチンを示します。

XEAS : XPUのエラー発生時にリンクするサブルーチンを示します。

<オプション>

-u site : ビルダの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE” に設定されたサイトに対して処理します。

第5章 ビルダ

<注意事項>

同一のサブプログラムを複数の組み込み箇所 (point)、複数のエントリ番号 (en) に組み込むことはできません。

オプションの組み合わせを以下に示します。

種別		組み込み
		サブルーチン
パラメータ		
name point en -ul		◎
オプション	-u site	○

◎：必須 ○：指定できます。

<終了コード>

svbuildコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

<名前>

svdbuild (間接リンクサブプログラムの削除)

<形式>

svdbuild name -ir [オプション]

<機能説明>

svdbuildは、間接リンクサブプログラムを削除します。

<引数説明>

name : 間接リンクサブプログラム名称 (マルチエントリローディング時はエントリ名称) を指定します。

-ir : 間接リンクサブプログラムの削除指定です。

<オプション>

-u site : ビルダの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

		種別	間接リンク サブプログラム
パラメータ			
name -ir			◎
オプション	-u site		○

◎ : 必須 ○ : 指定できます。

<終了コード>

svdbuildコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

第5章 ビルダ

<名前>

svdbuild (組み込みサブルーチンの削除)

<形式>

svdbuild name point -ul [オプション]

<機能説明>

svdbuildは、組み込みサブルーチンを削除します。

<引数説明>

name : 組み込みサブルーチン名称を指定します。

point -ul : 組み込みサブルーチンの削除指定です。

組み込み箇所 (point) には、以下の文字列を指定してください。

CPES : CPES組み込みサブルーチンを示します。

IES : IES組み込みサブルーチンを示します。

EAS : EAS組み込みサブルーチンを示します。

INS : INS組み込みサブルーチンを示します。

EXS : EXS組み込みサブルーチンを示します。

ABS : ABS組み込みサブルーチンを示します。

PCKS : PCKS組み込みサブルーチンを示します。

MODES : MODES組み込みサブルーチンを示します。

WDTES : WDTES組み込みサブルーチンを示します。

XEAS : XPUのエラー発生時にリンクするサブルーチンを示します。

<オプション>

-u site : ビルダの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

<注意事項>

オプションの組み合わせを以下に示します。

種別		組み込み サブルーチン
パラメータ		
name point -ul		◎
オ プ シ ョ ン	-u site	○

◎：必須 ○：指定できます。

<終了コード>

svdbuildコマンドは次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

第5章 ビルダ

<名前>

svirglb

<形式>

svirglb idxnum name [オプション]

<機能説明>

svirglbは、svdfsで確保した細分割領域を間接リンクグローバルとして登録または削除します。

<引数説明>

idxnum : 間接リンクグローバルテーブルの登録番号を指定します (1~7935の範囲で指定できます)。

name : 間接リンクグローバル名称を指定します (-s、-aを指定しない場合には間接リンクグローバル名称は、svdfsで確保済みの細分割領域名称を指定します)。

<オプション>

-u site : ビルダの処理対象となるサイト名称 (site) を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

-s name : 間接リンクグローバルテーブルに格納するアドレスとして、間接リンクグローバル+オフセットで格納する場合に、細分割領域名称を指定します。

-o offset : 間接リンクグローバルテーブルに格納するアドレスとして、間接リンクグローバル+オフセットで格納する場合に、オフセット (offset) を16進数または10進数で指定します。0xで始めると16進数として扱います。

-a adr : 間接リンクグローバルテーブルに格納するアドレスを16進数または10進数の絶対アドレスで指定します。0xで始めると16進数として扱います。

-d : 間接リンクグローバルテーブルから指定した登録番号を削除します。

<注意事項>

- -s、-oオプションは同時に指定してください。
- -s、-oオプションと-aオプションを同時に指定するとエラーになります。
オプションの組み合わせを以下に示します。

種別		間接リンクグローバル	
		登録	削除
パラメータ			
idxnum		◎	◎
name		◎	◎
オプション	-u site	○	○
	-s name	○	○
	-o offset	○	○
	-a	○	○
	-d	×	◎

◎：必須 ○：指定できます。×：指定できません。

- -aオプションにおいて指定できるアドレスは、CM、GLB内のアドレスになります。
- -sオプションにおいて指定した細分割領域を含む分割領域の範囲外にオフセットを指定するとエラーとなります。

<終了コード>

svirglbコマンドは次の終了コードを返します。

- 0 : 正常終了
- 0以外 : 異常終了

第6章 管理ツール

<名前>

svmap

<形式>

- ・ 名称指定

```
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]
```

- ・ 番号指定

```
svmap num [num...] [-site sitename] -irs [-f] [-CON] [-osrsv]
           [-site sitename] -irg [-f] [-CON] [-osrsv]
           [-site sitename] -uls [-f] [-CON] [-osrsv]
```

- ・ 全体表示

```
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]
```

<機能説明>

svmapは、RPDPで管理しているリソースの情報を出力します。リソースの情報はRSUTYPの設定によらずシステム/ユーザの情報を表示します。

<引数説明>

- name** : 表示するリソースの名称を指定します。このとき名称の種別を表すオプションは省略できません。名称は複数指定することができます。
- num** : 表示するリソースの番号を指定します。番号は複数指定することができます。このとき番号の種別を表すオプションは省略できません。
- irs、-irg指定のときnumには表示したいIRSUB、IRGLBの番号を指定します。
- uls指定のときnumには組み込みポイントとエントリ番号を指定します。組み込みポイントとエントリ番号はpnt、typ、entの形式で指定してください。（例：eas, os, l）

<オプション>

- site sitename : マップ情報を出力するサイト名称を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
- G : 大分割領域の情報を出力します。-a、-eと組み合わせることで大分割領域／分割領域／細分割領域の階層マップを出力することができます。
- a : 分割領域の情報を出力します。-eと組み合わせることで分割領域／細分割領域の階層マップを出力することができます。
- e : 細分割領域（GLB、CMのsarea、プログラム、サブプログラム）の情報を出力します。
- p : プログラムの情報を出力します。
- s : サブプログラム（IRSUBとULSUB）の情報を出力します。
- t : タスクの情報を出力します。
- g : グローバル（GLB、CM）の情報を出力します。
- v : VAL情報を出力します。
- irs : IRSUBのエントリ情報を出力します。
- irg : IRGLBのエントリ情報を出力します。
- uls : ULSUBのエントリ情報を出力します。
- en : IRSUB、GLB（CM含む）、VALの登録数に関する情報を出力します。
- s、-g、-v、-irs、-irgと同時に指定すると、指定に対応するものの登録数に関する情報だけを表示します。
- f : 詳細情報を表示します。
- fm : 物理メモリの空き情報を出力します。
- CON : S10VEメモリ上のマップを出力します。このオプション省略時は開発系マシン側で管理するリソースのマップ情報を出力します。
- osrsv : OSリザーブのリソースを表示します。このオプション省略時はOSリザーブのリソースは表示しません。osreserveで始まる名称をOSリザーブとして扱います。ユーザは使用しないでください。
- help : コマンドの起動形式を説明するリストを表示します。
- +a : 結果をアドレス順にソートして出力します。

第6章 管理ツール

- +n : 結果を名称順にソートして出力します。
- +e : 結果をエントリ番号順にソートして出力します。
- +gn gname : garea/area/sareaの階層マップ出力時に、特定の大分割領域の情報だけを出力する場合、大分割領域名称を指定します。大分割領域名称はTASK、IRSUBなどのように\$のつかない名称で指定してください。

表示するリソースの種別を指定するオプション (-G、-a、-e、-p、-s、-t、-g、-v、-irs、-irg、-uls) と結果の出力順を指定するオプション (+a、+n、+e) の組み合わせの可否と、+a、+n、+eを省略した場合のデフォルトの出力順を表2-6に示します。

表示するリソースの種別を表すオプション (-G、-a、-e、-p、-s、-t、-g、-v、-irs、-irg、-uls) および-en、-fm、-helpをすべて省略すると、下記が指定されたものとして扱います。

-G -a -e -t -v -irs -irg -uls

<終了コード>

- 0 : 正常終了
- 0以外 : 異常終了

表2-6 出力リソース指定と出力順指定の組み合わせ可否とデフォルトの出力順

出力リソース指定 出力順指定	-G	-a	-e	-p	-s	-t	-g	-v	-irs	-irg	-uls
+a	○	○	○	×	×	×	×	×	×	×	×
+n	○	○	○	○	○	○	○	○	○	○	○
+e	×	×	×	×	○	○	○	×	○	○	○
なし (デフォルト)	+a	+a	+a	+n	+n	+e	+n	+n	+e	+e	+e

○ : 組み合わせ可 × : 組み合わせ不可

+a : デフォルトはアドレス順 +n : デフォルトは名称順 +e : デフォルトは番号順

<名前>

svadm

<形式>

svadm [addr] [オプション]

<機能説明>

svadmは、指定した論理アドレスに登録されているリソース（グローバル、IRSUB）の名称など（詳細は<表示フォーマット>参照）の情報を出力します。

論理アドレスを省略した場合は、会話形式で論理アドレスを取り込み、名称などの情報を出力します。

<引数説明>

addr：論理アドレス（addr）を指定します。省略時は、会話形式で論理アドレスを取り込みます。

アドレスの範囲は0x30000000～0x7fffffff（タスク空間、GLB空間、サブプログラム空間、CM、LADDER、USRFUNC、HI-FLOW空間）です。

<オプション>

-u site：論理アドレス情報の表示対象となるサイト名称（site）を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

-o file：オペレーション結果出力先のファイル名称（file）を指定します。

<終了コード>

svadmコマンドは、次の終了コードを返します。

0：正常終了

0以外：異常終了

<オペレーション>

引数でアドレス指定した場合

```
#svadm addr[Enter]
```

情報表示

```
#
```

引数でアドレス指定しなかった場合

```
#svadm[Enter]
```

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

```
addr : addr[Enter]
```

情報表示

```
addr : q[Enter]
```

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

```
#
```

<説明>

アンダーライン部はユーザが入力してください。

XXXXX : サイト名称

addr : 情報を得たいアドレスを指定します。

q : コマンドを終了します。

[Enter] : [Enter] キーを押します。

<表示フォーマット>

表示内容は以下の3種類です。

- 指定アドレスにリソースが登録されている場合のフォーマット

```
name = NAME type = TYPE raddr = XXXXXXXX
```

NAME : リソースの名称 (sarea名、プログラム名、サブプログラム名) を表します。

TYPE : リソースの種別を表します。

task(TEXT) : タスクのテキスト部です。

task(DATA) : タスクのデータ部です。

task(BSS) : タスクのBSS部です。

task(STACK) : タスクのスタックです。

sub : サブプログラムです。

data : GLBのsareaです。

CM : CM領域のsareaです。

XXXXXXXX : リソースの先頭からのオフセットです。タスクの場合はそれぞれTEXT、DATA、BSS、STACK領域の先頭からの相対となります。サブプログラムの場合はサブプログラム先頭からの相対、GLB、CMの場合はそれぞれのsareaの先頭からの相対となります。

- 指定アドレスにリソースが登録されていない場合のフォーマット

```
lspace = SPACE external name is not defined
```

SPACE : 指定したアドレスのGAREA名を表示します。

- 指定アドレスがGAREA範囲外の場合のフォーマット

```
address error (0xXXXXXXXX)
```

XXXXXXXX : 指定したアドレスを表示します。

LADDER、USRFUNC、HIFLOWの空間のアドレスを指定した場合も、エリアが定義されていないため、このフォーマットの表示となります。

第6章 管理ツール

<名前>

svsitectl

<形式>

svsitectl [オプション]

<機能説明>

svsitectlは、指定サイトの状態を表示します。また、-rsrcvオプション指定時は指定サイトのリカバリを抑制します。

<オプション>

-query [-sort] : 開発系マシンに登録されているサイトの一覧を表示します。

- “-query” の表示

-queryオプション指定時、S10VEのサイトを表示します。

以下に出力フォーマットを示します。

-sort指定時表示は、機種ごとにアルファベット順にソートして出力します。

```
+++++++ S10VE site ++++++
0001cp is active
0001hp is active
0002cp is active
0002hp is active
1000cp is active
1000hp is active
```

-rsrcv site : svdebugコマンドのldサブコマンドで中断時に、RPDPのリカバリ処理なしにコマンドを利用できるようにします。

ただし、svdebugのldサブコマンドは、利用できません。

(詳細は、「付録E RPDP使用上の注意事項」を参照してください。)

<終了コード>

svsitectlコマンドは、次の終了コードを返します。

0 : 正常終了

0以外 : 異常終了

第7章 立ち上げ／PU制御

<名前>

svrpl — リモートローディング

<形式>

svrpl [{-u site | -U unit} {-s}] [-all] [-r] [{-time|-notime}] [-ROMSV | -NOROMSV] [-setpc sno]

<機能説明>

svrplは、指定サイト（PU）をストップさせ、バックアップファイルの内容をS10VE側指定サイト（PU）の主メモリに転送し、指定サイトを立ち上げます。オプションには以下のものがあります。

<オプション>

- u site : ローディング対象となるサイト名称を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
CPU名称（CPのサイト名称と同一名称）を指定してください。
- U UNIT : 全サイトローディング時のユニットを指定します。
-uオプションと-Uオプションを同時に指定できません。
- all : 全バックアップファイルをダウンロードします。

オプションによるダウンロードファイルは下表のようになります。

ファイル オプション	OS	TASK、SUB、GLB	CM
-all	○	○	○
指定なし	○	○	○

○：ダウンロードします。 ×：ダウンロードしません。

- r : ローディング終了後、指定サイト（PU）をスタートしません。
- s : 指定サイト（PU）をストップするかどうかの確認応答を取らないでストップします。
-uオプションと同時に指定してください。また、-sオプションを指定しなかった場合はオペレータ操作とみなし、ダウンロード時のストップ確認を会話形式で行います。
- time : 指定サイトがCPUのとき、時刻を設定します。
- notime : 指定サイトがCPUのとき、時刻を設定しません。
時刻を設定したくない場合には、このオプションを設定してください。

- ROMSV : リモートローディングしたSDRAMの内容をROM (NAND-Flash) にセーブします。
- NOROMSV : リモートローディングしたSDRAMの内容をROM (NAND-Flash) にセーブしません。
 - ROMSVも-NOROMSVも指定しなかったときは-ROMSVが指定されたものとして扱います。
- setpcsn0 : サイト名とPCs番号の設定機能を有効にします。

svrplは、最初に指定サイトが属するユニットに実装されている全PUの状態を表示します。

続いて、ダウンロードするサイト (PU) をストップさせてもよいかどうかを確認します。ここで、yesを入力すると指定サイト (PU) をストップさせダウンロードを開始します。noを入力すると何もしないで終了します。

<使用上の留意点>

- svrplではLADDERとHI-FLOWのプログラムはダウンロードされません。
- svrplでローディングできるS10VEは、BASE SYSTEM/S10VEの「接続PCs変更」機能で接続されているS10VEのみです。
- BASE SYSTEM/S10VEの「CPMSダウンロード」で一度もOSをダウンロードしたことのないS10VEをsvrplで立ち上げることはできません。最初は必ずBASE SYSTEM/S10VEの「CPMSダウンロード」でOSをローディングしてください。
- サイト名称にHPのサイト名称を指定した場合は、エラーメッセージを表示したあとコマンドを終了します。
- ローディング中にエラーが起きた場合、指定サイト (PU) をストップさせたまま終了します。
- ダウンロード後、指定PUをスタートさせない (-rオプション指定) ときは、-timeオプション指定があっても無視します。時刻設定を行いません。
- 状態制御 (svcpuctl) コマンドでCPUに対してRUN要求をするとき、-timeオプションを指定すると時刻を設定できます。
- CPUに対してダウンロード時、-timeオプションを指定しなかった場合は、時刻を設定しません。

<サイト名とPCs番号の設定機能>

-setpcsnoオプションを指定すると、CPMSのSYSCBテーブルローディング時、サイト名 (sy_site) をローディングしたサイト名で書き換えます。これにより、BASE SYSTEM/S10VEがベースサイトをコピーして生成したPCs番号ごとのサイトを実機にローディングすると、実機上のサイト名をPCs番号に書き換えます。

-setpcsnoオプションを指定しない場合、CPMSのSYSCBテーブルローディング時、サイト名 (sy_site) をローディングしません。このとき、実機上のサイト名はsvrplコマンドでローディングする前に (BASE SYSTEM/S10VEのCPMSダウンロードで) ローディングされていたサイト名のままとなります。

また、-setpcsnoオプションを指定してPCs番号ごとのサイトをローディングすると、PCs番号をMRAMに書き込みます。PCs番号ごとのサイトとして認識されるサイト名は、CPのサイト名が10進4桁の数字 (0000~9998) +cpの場合です。CPのサイト名が10進4桁の数字+cp以外および9999cpである場合、PCs番号ごとのサイトとして認識されないため、MRAMへの書き込みは行いません。

-setpcsnoオプションを指定しない場合、PCs番号をMRAMに書き込みません。

-setpcsnoオプションを指定した場合、-NOROMSVオプションは指定できません。

<終了コード>

svrplコマンドは、次の終了コードを返します。

0 : 正常終了

1 : 異常終了

2 : 通信異常

3 : [Ctrl] + [C] キーを押すことによって中断

<名前>

svcpuctl - リモート状態制御

<形式>

svcpuctl [{-u site} {-s{-stop | -run}} [-time] (状態制御)

svcpuctl [-u site] -ss (状態表示)

<機能説明>

svcpuctlは、指定サイト (PU) の状態を制御します。また、指定サイト (PU) の状態を表示します。オプションには以下のものがあります。

<状態制御オプション>

-u site : 処理対象となるサイト名称を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

CPU名称 (CPのサイト名称と同一名称) を指定してください。

-s : 本当にコマンドを実行してよいかどうかの確認応答を取りません。

-uオプションと同時に指定してください。

-sオプションを指定しなかった場合は、オペレータ操作とみなし、状態 (stop/run) 指定を会話形式により指定します。

-stop : 指定サイト (PU) をCPU STOP状態にします。

-run : 指定サイト (PU) をCPU RUN状態にします。

-stop、-runオプションは-sオプションと同時に指定してください。

また、-stop、-runは同時に指定できません。

-time : CPU RUN要求時、指定サイトがCPUのときの時刻設定をします。

svcpuctlは、最初に指定サイトが属するユニットの実装されている全PUの状態を表示します。次に制御したい状態 (runまたはstop) を入力することにより、指定サイト (PU) の状態を変更させてもよいかどうかを確認します。ここで、yesを入力すると指定サイト (PU) の状態を変更します。noを入力すると何もしないで終了します。

<状態表示オプション>

-ss : 指定サイトのPU状態 (CPU RUNまたはSTOP) を表示します。

CPがCPU RUN状態でHPがCPU STOP状態のときは、RUN(HP STOP)と表示します。

<使用上の留意点>

- svcpuctlで状態を制御/表示できるS10VEは、BASE SYSTEM/S10VEの「接続PCs変更」機能で接続されているS10VEのみです。
- サイト名称にHPのサイト名称を指定した場合は、エラーメッセージを表示したあとコマンドを終了します。
- 状態制御オプションと状態表示オプションは同時に指定できません。

<終了コード>

svcpuctlは、次の終了コードを返します。

- 0：正常終了
- 1：異常終了
- 2：通信異常
- 3：[Ctrl] + [C] キーを押すことによって中断

第8章 svdebug (オンラインデバッガ) とデバッグ支援コマンド

<名前>

svdebug - S10VE用オンラインデバッガ

<形式>

svdebug [オプション]

<機能説明>

svdebug コマンドは、S10VE用オンラインデバッグ機能を提供します。

svdebug コマンドは、起動、終了時に下記メッセージを出力し、デバッガ起動後、プロンプト“サイト名称”が表示されると、各サブコマンドが受け付けできます。

svdebug 起動時にS10VEにブレークポイントが設定されていた場合、それぞれの設定内容を表示します。

<デバッガ起動時のメッセージ>

```
++ debugger start ++
```

break point

name = プログラム名称 raddr = プログラム内相対アドレス object = 機械語命令パターン

・
・

name = プログラム名称 raddr = プログラム内相対アドレス object = 機械語命令パターン

サイト名称>

<デバッガ終了時のメッセージ>

```
++ debugger end ++
```

(注) -s オプションを指定して起動した場合、上記のメッセージは出力されません。

<オプション>

-i fname : キー入力結果をファイルに出力するときのファイル名を指示します。

-o fname : 指定されたファイルにオペレーションの日付および結果を出力します。

-r fname : 指定されたコマンドファイル内のサブコマンド行を実行します。すべてのサブコマンド行を実行したあと、svdebug コマンドは自動的に終了します。

コマンドファイルは、-i オプションで作成したファイルを利用できます。

-s sub command :

このオプションに指定したサブコマンドをダイレクト実行します。サブコマンド実行後、svdebug コマンドは自動的に終了します。

-u site : デバッガの処理対象となるサイト名を指定します。このオプション省略時は、環境変数
“RSSITE” に設定されたサイトに対して処理します。

-debug : デバッグモードを指定します。拡張サブコマンドが使用できます。

<終了コード>

svdebugコマンドは、正常終了した場合戻り値として0を返し、異常終了した場合戻り値として1を返します。ただし、-sオプションで指定したサブコマンドがエラーになった場合は、戻り値として255を返します。

<注意事項>

- 複数オプション指定時には、-sオプション以降のオプション指定はサブコマンドとみなされ、無視されます。

(例)

svdebug -i fname -s サブコマンド : -iはオプションとして判断されます。

svdebug -s サブコマンド -i fname : -iは-sのサブコマンドの一部としてみなされます。

- svdebugコマンドに指定したオプションが、<機能説明>の項で説明されているオプションのいずれにも該当しない場合は、以下の使用例が出力されます。

```
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
```

- -rオプションで使用するコマンドファイルを、-iオプション指定時のキー入力結果ファイルを使用しないで作成する場合は、以下の点に注意してください。
 - ・サブコマンド仕様に準じないデータが設定されている場合でも、チェックが行われずにその行を実行します。
 - ・対話型インタフェースを持つサブコマンドについては、オペレーション手順ごとにプロンプト番号に対応するデータを行単位に指定します。
 - ・コマンドファイル内に現れた空行は、サブコマンド行が現れなかったものとしてその行を無視します。
- -rオプション、-sオプションを使用して、ブレイクポイントのサブコマンド (br、rb、rr、rd、go、stickybr) は使用できません。

- サブコマンド一覧
svdebugの機能を表2-7に示します。

表2-7 svdebug機能一覧

分類	サブコマンド	機能
タスク起動/停止	qu	タスクの起動要求
	ab	タスクの起動禁止
	re	タスクの起動禁止解除
	ta	タスクの状態表示
	su	タスクの実行抑止
	rs	タスクの実行抑止解除
	tm	タスクの周期起動
	ct	タスクの周期起動解除
	sht	タスクの周期起動表示
	メモリプリント/ パッチ	md
sd		名称指定によるメモリ内容の表示/変更
bs		指定ビットへのデータ設定
bg		指定ビットのデータ表示
mcp		メモリ内容のコピー
mmv		メモリ内容の移動
mf		メモリへのパターン値設定
ブレークポイント	br/stickybr	ブレークポイントの設定/表示
	rb	ブレークポイントの解除
	rd	レジスタの表示
	rr	レジスタの内容変更
	go	ブレークポイントからの実行再開
システムエラー表示	el	エラーログの表示
	ss	システムの状態表示
現在時刻設定/表示	st	現在時刻の設定
	gt	現在時刻の表示
アップ/ダウンロー ディング、コンペア	ld	バックアップファイルの内容をコントローラのメモリに転送
	sv	コントローラのメモリの内容をバックアップファイルに転送
	cm	バックアップファイルとコントローラメモリの内容比較
DHP記録許可/禁止	dr	DHP記録許可
	ds	DHP記録禁止
ラダーのデバッグ機 能	lbr	ブレークポイントの設定/表示
	lrb	ブレークポイントの解除
	lrd	レジスタの表示
	lrr	レジスタの書き換え
	lgo	ブレークポイントからの実行再開
	s	ステップ実行
その他	si	スタック初期化
	sp	スタック使用量の表示
	svdhp	DHPの表示
	svadm	アドレスに対するリソース名称の表示
	ps	デバッグ文の表示開始
	pe	デバッグ文の表示終了
	ver	CPMSのバージョン表示
	help	サブコマンド一覧表示
	q	デバッガの終了
	!	svdebug実行時の開始系マシン上のコマンド実行

<名前>

qu - タスクの起動要求

<形式>

qu tn[,fact]

qu tname[,fact]

<機能説明>

quは、指定されたタスクを起動します。指定するパラメータは以下のとおりです。

tn : タスク番号 (1~最大タスク番号)

fact : 起動要因 (1~32)

tname : タスク名

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

- factを省略した場合、fact=0が設定されたものとして扱われます。
- このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。
また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn[,fact] or tname[,fact]
```

```
:
```

<名前>

ab - タスクの起動禁止

<形式>

ab tn1[-tn2]

ab tname

<機能説明>

abは、指定されたタスクを起動禁止状態にします。指定するパラメータは以下のとおりです。

tn1 : 先頭タスク番号 (1~最大タスク番号)

tn2 : 最終タスク番号 (tn1~最大タスク番号)

tname : タスク名

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

ただし、tn1-tn2指定時はすべて正常終了となります。

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。

また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn1[-tn2] or tname
```

```
:
```

<名前>

re - タスクの起動禁止解除

<形式>

re tn1[-tn2]

re tname

<機能説明>

reは、指定されたタスクの起動禁止状態を解除します。指定するパラメータは以下のとおりです。

tn1 : 先頭タスク番号 (1~最大タスク番号)

tn2 : 最終タスク番号 (tn1~最大タスク番号)

tname : タスク名

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

ただし、tn1-tn2指定時はすべて正常終了となります。

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。

また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn1[-tn2] or tname
```

```
:
```

<名前>

ta - タスクの状態表示

<形式>

ta tn1[-tn2 [-s|-r]] [-susp]

ta tname [-susp]

<機能説明>

taは、指定されたタスクの状態を表示します。指定されたタスクが実行抑止状態の場合、レジスタの内容も表示します。指定するパラメータおよびオプションは以下のとおりです。

tn1 : 先頭タスク番号 (1~最大タスク番号)

tn2 : 最終タスク番号 (tn1~最大タスク番号)

tname : タスク名

-s : タスク番号、タスク名、タスクステータスだけが表示されます。

-r : NON EXISTENT、DORMANT、IDLE以外のタスクの状態が表示されます。

-susp : 指定タスクの状態がWAIT状態でないまたはSUSPENDED状態でない場合、強制的にSUSPENDED状態にしてレジスタ情報を表示します。

このオプションはsvdebug起動時に-debugオプションを指定した場合に使用できます。それ以外はエラーとなります。

taサブコマンドの表示フォーマットは以下のとおりです。

```
tn=*** (0x**)  tname=***** task state=***** (0x*****)  
tcb top=0x***** fact=0x***** level=** (**)  
task top=0x***** stack=0x*****-0x*****  
(レジスタ情報)
```

● 表示情報の説明

tn : タスク番号 (10進および16進で表示)

tname : タスク名

task state : タスク状態 (ステータスビットを16進表示)

fact : タスクの起動要因

level : カレントタスクレベル (カッコ内はタスクレベル初期値)

tcb top : TCBの先頭アドレス

task top : タスク先頭アドレス

stack : タスクスタック範囲

レジスタ情報 : タスクが実行抑止状態のとき、レジスタ情報も表示されます。

レジスタ情報の出力フォーマットは以下のとおりです。

```

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*****
    
```

タスク状態として表示する情報は、表 2-8 に示すいずれかとなります。

表 2-8 タスクの状態

状態	意味
NON EXISTENT	未登録
DORMANT	起動抑止状態
IDLE	起動待ち状態
READY	実行中または実行待ち状態
WAIT	イベント待ち状態
SUSPENDED	実行抑止状態

ステータスビットの構成とその意味は、表 2-9 のとおりです。

表 2-9 ステータスビットの構成

ステータスビット値	意味
0x1	多重起動あり
0x10	DELAYによる実行抑止中
0x20	SUSPによる実行抑止中
0x40	RSERV、PRSRVの資源解放待ち
0x80	ブレークポイントによる実行抑止中
0x1000	EXIT処理実行中
0x2000	RLEAS処理ペンディング中
0x4000	ABORT処理実行中
0x8000	QUEUE処理ペンディング中

(注) ステータスビットは、複数のビットが同時にONになる場合があります。

<注意事項>

- このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。
また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn1[-tn2 [-s|-r]] or tname
:
```

- -suspオプションを指定した場合、タスクの範囲指定は行えません。
- -suspオプションを指定時にタスク状態が、DORMANTまたはNON EXISTである場合、エラーメッセージを出力し、サブコマンドを終了します。
- -suspオプションを指定した場合、レジスタ情報を取得するために、タスクの状態がSUSPENDEDでなくても、一時的にSUSPENDEDにします。

<名前>

su - タスクの実行抑止

<形式>

su tn1[-tn2]

su tname

<機能説明>

suは、指定されたタスクの実行を抑止します。指定するパラメータは以下のとおりです。

tn1 : 先頭タスク番号 (1~最大タスク番号)

tn2 : 最終タスク番号 (tn1~最大タスク番号)

tname : タスク名

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

ただし、tn1-tn2指定時はすべて正常終了となります。

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。

また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn1[-tn2] or tname
```

```
:
```

<名前>

rs - タスクの実行抑止解除

<形式>

rs tn1[-tn2]

rs tname

<機能説明>

rsは、指定されたタスクの実行抑止状態を解除します。指定するパラメータは以下のとおりです。

tn1 : 先頭タスク番号 (1~最大タスク番号)

tn2 : 最終タスク番号 (tn1~最大タスク番号)

tname : タスク名

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

ただし、tn1-tn2指定時はすべて正常終了となります。

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。

また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn1[-tn2] or tname
```

```
:
```

<名前>

tm — タスクの周期起動

<形式>

tm

id:

tn[,fact]: (または tname[,fact])

t,cyct:

または

tm

id:

tn[,fact]: (または tname[,fact])

t:

<機能説明>

tmは、指定されたタスクに対して周期起動をかけます。tmが起動するとサブプロンプトが表示されます。指定するパラメータは以下のとおりです。

id : 起動するタイマタスクの種類 (1~4)

tn : タスク番号 (1~最大タスク番号)

tname : タスク名

fact : 起動要因 (1~32)

t : 初回のタイマイベントの時刻または現在からの相対時間
相対時間はミリ秒で指定します。

idとして1または3を指定したとき、相対時間として1~86400000が指定できます。

idとして2または4を指定したとき、相対時間として0~86399999が指定できます。

cyct : 周期的にイベントを発生させる場合の周期時間

周期時間はミリ秒で指定します。周期時間として1~86400000が指定できます。

id、t、cyctの詳細は表 2-10を参照してください。

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

- factを省略した場合、fact=0が設定されたものとして扱います。
- パラメータの入力待ちの状態、[e] または [Enter] キーを押した場合、サブコマンドを終了します。

表 2-10 id、t、cyctの説明

タイマイベント	id	t	cyct	説明
時間指定	1	現時刻から起動までの 相対時間		パラメータtで指定された時間経過 後、パラメータtn、tnameで指定さ れたタスクを起動します。
時刻指定	2	午前0時を起点にした 起動時刻		パラメータtで指定された時刻に、 パラメータtn、tnameで指定された タスクを起動します。
時間周期指定	3	現時刻から起動までの 相対時間 (初回の起動までの相 対時間)	初回の起動後、周期的 に起動する周期時間	パラメータtで指定された時間経過 後、パラメータtn、tnameで指定さ れたタスクを起動します。 その後、パラメータcyctで指定され た周期で、パラメータtn、tnameで 指定されたタスクを起動します。
時刻周期指定	4	午前0時を起点にした 起動時刻 (初回の起動時刻)	初回の起動後、周期的 に起動する周期時間	パラメータtで指定された時刻に、 パラメータtn、tnameで指定された タスクを起動します。 その後、パラメータcyctで指定され た周期で、パラメータtn、tnameで 指定されたタスクを起動します。

<名前>

ct — タスクの周期起動解除

<形式>

ct tn[,fact]

ct tname[,fact]

<機能説明>

ctは、指定されたタスクの周期起動を解除します。指定するパラメータは以下のとおりです。

tn : タスク番号 (1~最大タスク番号)

fact : 起動要因 (1~32)

tname : タスク名

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

(≠0) の部分はマクロのリターンコードが表示されます。

<注意事項>

- factを省略した場合、fact=0が設定されたものとして扱います。
- このサブコマンド起動時にパラメータを何も指定しなかった場合、または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。
また、この状態で [e] または [Enter] キーを押した場合にはサブコマンド処理を終了します。

```
input tn[,fact] or tname[,fact]
```

```
:
```

第8章 svdebug（オンラインデバッガ）とデバッグ支援コマンド

<名前>

sht — タスクの周期起動表示

<形式>

sht

<機能説明>

shtは、現在設定されているタスクの周期起動をすべて表示します。

<結果>

以下の形式でタスク周期起動を表示します。

ID	TN	FACT	TIME	CYT
*	***	**	****/**/** **:*:*:**.***	*****
			.	
			.	
			.	

ID : タイマ種別

TN : タスク番号

FACT : 起動要因

TIME : 起動時刻（年／月／日 時：分：秒. ミリ秒）

CYT : 周期時間（ミリ秒）

<注意事項>

shtサブコマンドは、同一サイトに対して同時に複数の起動はできません。

<名前>

md - アドレス指定によるメモリ内容の表示/変更

<形式>

md

```

1 storage(s,m,*) : {s}
                   {m}
                   {*}
                   {e}
                   {return}

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}
                        {return}
                        {e}
    
```

(注) アンダーライン部はユーザが入力してください。

<機能説明>

mdは、論理アドレスを指定することでメモリ内容を表示/変更します。起動するとサブプロンプトが表示されます。各々のサブプロンプトに対する入力は以下のようになります。

1 storage(s,m,*)

- s : バックアップファイルを変更・表示の対象とします。
- m : 実機メモリを変更・表示の対象とします。
- return : 実機メモリを変更・表示の対象とします。
- * : バックアップファイル、実機メモリを変更の対象とします。
- e : サブコマンドの終了を指示します。

(注) storage=* 指定時のデータ表示は、バックアップファイルを対象とします。

2 addr

addr1-addr2 : 表示先頭アドレスaddr1から表示最終アドレスaddr2までを表示することを指示します。

addr1,len : 表示先頭アドレスaddr1からlenで指定したバイト数を表示することを指示します。

-h : データ出力形式を16進表示とします。

-d : データ出力形式を10進表示とします。

-f : データ出力形式を単精度浮動小数点表示とします。

-fd : データ出力形式を倍精度浮動小数点表示とします。

-l : データ長を4バイトとします。

-w : データ長を2バイトとします。

-b : データ長を1バイトとします。

-all : 同一データが行単位で連続した場合、省略表示をしないように指示します。

-omit : 同一データが行単位で連続した場合、省略表示をするように指示します。

*n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。

e : サブコマンドの終了を指示します。

0xaaaaaaaa 0xdddddddd

data : 変更するデータの値を指定します。

return : データの変更がないことを指示します。

e : 2 addrの入力に戻ります。

<注意事項>

- 2 addrの指定で、addr2およびlenのパラメータ指定がないときデータ変更 (パッチ) モードとなり、このパラメータ指定があるときはデータ表示 (プリント) モードとなります。
また、データ表示終了後は、再度2 addrの入力待ちとなります。
- データ出力形式とデータ長の両オプションを省略した場合、このサブコマンド内で最後に指定したものが有効となります。デフォルトは16進4バイト (-h、-l) です。
- データ変更 (パッチ) 時に指定できる値は、データ出力形式とデータ長の両オプションの指定に依存します。指定できる値とオプションの組み合わせを表2-11に示します。

表 2-11 指定できる値とオプションの組み合わせ

fmt \ size	-l	-w	-b	指定なし
-h	○	○	○	—
-d	○	○	○	—
-f	◎	○	○	—
-fd	×	×	×	◎
指定なし	—	—	—	—

○ : 8進、10進、16進が指定できます。
 ◎ : 実数が指定できます。
 — : 前回指定したfmt, sizeに依存します。
 × : 指定できないオプションの組み合わせです。

(fmt : データ出力形式 size : データ長)

- データ出力形式に-fd (倍精度浮動小数点形式) を指定しデータ表示・変更をしたあとで、データ出力形式またはデータ長のオプションを省略して、データ表示・変更を行う場合、データ出力形式のオプション省略時は-h (16進形式)、データ長のオプション省略時は-l (4バイト) が指定されたものとして扱います。
- 1 storageの指定で's'を指定し、2 addrの指定でバックアップファイルの存在しないアドレスを指定した場合は、エラーメッセージを表示し、2 addrの入力待ちとなります。
- データ表示 (プリント) の表示フォーマットは、データ出力形式とデータ長の両オプションの組み合わせにより、表 2-12 のようになります。

表 2-12 オプションの組み合わせによる表示フォーマット

fmt \ size	-l	-w	-b	指定なし
-h	h/4	h/2	h/1	—
-d	d/4	d/2	d/1	—
-f	f/4	h/2	h/1	—
-fd	×	×	×	f/8
指定なし	—	—	—	—

fmt/size : fmt : h (16進形式)
 d (10進形式)
 f (浮動小数点形式)
 size : バイトサイズ
 — : 前回指定したfmt, sizeに依存します。
 × : 指定できないオプションの組み合わせです。

(fmt : データ出力形式 size : データ長)

- mdの表示 (READ) /変更 (WRITE) 時のメモリアクセス範囲は、下図の網掛けで示される範囲内となります。ただし、物理メモリにマッピングされていない空間はアクセスできません。

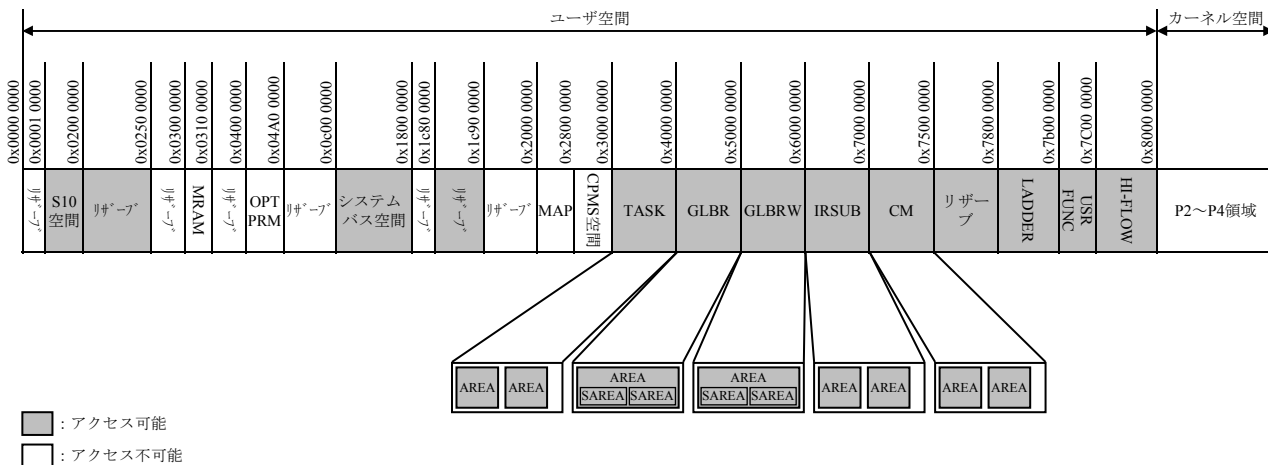
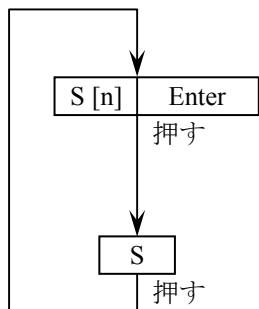


図 2-4 メモリアクセス範囲

- mdではデータ表示 (プリント) モードでのダイナミック表示機能をサポートしています。ダイナミック表示のオペレーションは、図 2-5 を参照してください。



[S] と [n] [Enter] キーを押すとダイナミック表示を開始します。[n] にはダイナミック表示のインターバルをms単位で指定します。nを省略するとインターバルは0msとなります。nには0~60000までの値が指定できます。モニタ中は、[S] キーだけが受け付けられます。ダイナミック表示を終了します (キー操作ができます)。

図 2-5 ダイナミック表示のオペレーション

<名前>

sd - 名称指定 (タスク名、サブプログラム名、プログラム名、グローバル名) による
メモリ内容の表示/変更

<形式>

sd

1 name : {name [-t | -s | -g | -b | -w]}
{e}

2 storage(s,m,*) : {s}
{m}
{*}
{*n}
{e}
{return}

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(0xllllll) 0xdddddddd : {data}
{return}
{e}

(注) アンダーライン部はユーザが入力してください。

<機能説明>

sdは、名称を指定 (タスク名、サブプログラム名、プログラム名、グローバル名) するとメモリ内容を表示/変更します。起動するとサブプロンプトが表示されます。各々のサブプロンプトに対する入力は以下のとおりです。

1 name

- name : 表示/変更の対象となる名称を指定します。
- t : nameで指定した名称がプログラム名であることを指示します。
相対アドレスをテキスト先頭からの相対として扱います。
- s : nameで指定した名称がサブプログラム名であることを指示します。
- g : nameで指定した名称がグローバルの細分割領域 (SAREA) 名であることを指示します。
- b : nameで指定したプログラム名称に対応するプログラムのBSS領域であることを指示します。
相対アドレスを指定プログラムのBSS領域先頭からの相対として扱います。
- w : nameで指定したタスク名称に対応するタスクのスタック領域であることを指示します。
相対アドレスを指定タスクのスタック領域先頭からの相対として扱います。
- e : サブコマンドの終了を指示します。

(注) -t、-s、-g、-b、-wオプションの指定を省略した場合、-gが指定されたものとして扱います。

2 storage(s,m,*)

- s : バックアップファイルを変更/表示の対象とします。
- m : 実機メモリを変更/表示の対象とします。
- return : 実機メモリを変更/表示の対象とします。
- * : バックアップファイル、実機メモリを変更の対象とします。
- *n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。
- e : サブコマンドの終了を指示します。

(注) storage=* 指定時のデータ表示は、バックアップファイルが対象となります。

3 baddr

- addr : 変更・表示の対象領域先頭からの相対アドレスを指定します。
- *n : nに指定したプロンプト番号の前処理に戻ります ($1 \leq n \leq 2$ の範囲で指定できます)。
- h : データ出力形式を16進表示とします。
- d : データ出力形式を10進表示とします。
- f : データ出力形式を単精度浮動小数点表示とします。
- fd : データ出力形式を倍精度浮動小数点表示とします。
- l : データ長を4バイトとします。
- w : データ長を2バイトとします。
- b : データ長を1バイトとします。
- all : 同一データが行単位で連続しても省略表示をしないように指示します。
- omit : 同一データが行単位で連続した場合に省略表示をするように指示します。
- e : サブコマンドの終了を指示します。

4 raddr

- addr1-addr2 : 表示先頭アドレスaddr1から表示最終アドレスaddr2までを表示することを指示します (アドレスはbaddrのaddrを起点として計算します)。
- addr1,len : 表示先頭アドレスaddr1からlenで指定したバイト数を表示することを指示します (アドレスはbaddrのaddrを起点として計算します)。
- addr1,* : 表示先頭アドレスaddr1から指定シンボルの残りの領域をすべて表示することを指示します (アドレスはbaddrのaddrを起点として計算します)。
- addr1-* : 表示先頭アドレスaddr1から指定シンボルの残りの領域をすべて表示することを指示します (アドレスはbaddrのaddrを起点として計算します)。
- * : 指定シンボルの該当領域すべてを表示することを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (1 ≤ n ≤ 3の範囲で指定できます)。
- e : サブコマンドの終了を指示します。

0xaaaaaaaa(0xl1111) 0xdddddddd

- data : 変更するデータの値を指定します。
- return : データの変更がないことを指示します。
- e : 4 raddrの入力に戻ります。

<注意事項>

- 4 raddrの入力でaddr1単独でパラメータ指定されたときデータ変更 (パッチ) モードとなり、これ以外のパラメータ指定のときはデータ表示 (プリント) モードとなります。また、データ表示終了後は、再度4 raddrの入力待ちとなります。
- データ出力形式/データ長の両オプションを省略した場合、このサブコマンド内で最後に指定したものを有効とします。デフォルトは16進4バイト (-h、-l) とします。
- sdで表示/変更できるエリアの領域範囲は、グローバルの場合、グローバルに割り付けた細分割領域 (SAREA) の範囲内で、プログラム/サブプログラムの場合、TEXT+DATA部、BSS部、STACK部の範囲内とします。
- データ変更 (パッチ) 時に指定できる値は、データ出力形式/データ長の両オプションの指定に依存します。指定できる値とオプションは表2-11を参照してください。
- データ表示 (プリント) 時の表示フォーマットの、データ出力形式とデータ長の両オプションの組み合わせはmdサブコマンドと同様です。
- sdサブコマンドは、mdサブコマンド同様にデータ表示 (プリント) モードでのダイナミック表示機能をサポートします。ダイナミック表示のオペレーションは、mdサブコマンドを参照してください。
- 1 nameの指定でバックアップファイルの存在しない領域を指定し、2 storageの指定で 's' を指定した場合は、エラーメッセージを表示し、2 storageの入力待ちとなります。

<名前>

bs - 指定ビットへのデータ設定

<形式>

bs

1 storage(s,m,*) : {s}
 {m}
 {*}
 {e}
 {return}

2 addr : {addr}
 {*n}
 {e}

3 bit : {bit1, len}
 {bit1-bit2}
 {*n}
 {e}

4 data : {data}
 {*n}
 {e}

(注) アンダーライン部はユーザが入力してください。

<機能説明>

bsは、指定されたアドレスのビット位置へデータを設定します。起動するとサブプロンプトが表示されます。各々のサブプロンプトに対する入力は以下のようになります。

1 storage(s,m,*)

- s : ビットセットの対象がバックアップファイルであることを指示します。
- m : ビットセットの対象が実機メモリであることを指示します。
- return : ビットセットの対象が実機メモリであることを指示します。
- * : ビットセットの対象がバックアップファイル、実機メモリであることを指示します。
- e : サブコマンドの終了を指示します。

2 addr

- addr : セット対象メモリのアドレスを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。
- e : コマンドの終了を指示します。

3 bit

bit1, len : 先頭ビット番号bit1からビット長lenの範囲に設定することを指示します。

bit1-bit2 : 先頭ビット番号bit1から最終ビット番号bit2の範囲に設定することを指示します。

*n : nに指定したプロンプト番号の前処理に戻ります ($1 \leq n \leq 2$ の範囲で指定できます)。

e : コマンドの終了を指示します。

4 data

data : 設定するデータを指示します。

‘0x’ または ‘0X’ で始めると16進数として処理し、それ以外は2進数として処理します。
指示したビット数分のパターンを指定してください。

*n : nに指定したプロンプト番号の前処理に戻ります ($1 \leq n \leq 3$ の範囲で指定できます)。

e : コマンドの終了を指示します。

<注意事項>

- メモリ内容を変更できる範囲はmdサブコマンドと同じです。
- 1 storageの指定で ‘s’ を指定し、2 addrの指定でバックアップファイルの存在しないアドレスを指定した場合は、エラーメッセージを表示し、2 addrの入力待ちとなります。

<名前>

bg — 指定ビットのデータ表示

<形式>

bg

1 storage(s,m,*) : {s}
 {m}
 {*}
 {e}
 {return}

2 addr : {addr}
 {*n}
 {e}

3 bit : {bit1, len}
 {bit1-bit2}
 {*n}
 {e}

(注) アンダーライン部はユーザが入力してください。

<機能説明>

bgは、指定されたビット位置のデータを表示します。起動するとサブプロンプトが表示されます。各々のサブプロンプトに対する入力は以下ようになります。

1 storage(s,m,*)

- s : ビット取り出し対象がバックアップファイルであることを指示します。
- m : ビット取り出し対象が実機メモリであることを指示します。
- return : ビット取り出し対象が実機メモリであることを指示します。
- * : ビット取り出し対象がバックアップファイル、実機メモリであることを指示します。
- e : サブコマンドの終了を指示します。

(注) storage=*指定時のデータ表示は、バックアップファイルを対象とします。

2 addr

- addr : 取り出し対象メモリのアドレスを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。
- e : コマンドの終了を指示します。

3 bit

- bit1, len : 先頭ビット番号bit1からビット長lenの範囲を表示することを指示します。
- bit1-bit2 : 先頭ビット番号bit1から最終ビット番号bit2の範囲を表示することを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (1 ≤ n ≤ 2の範囲で指定できます)。
- e : コマンドの終了を指示します。

<結果>

メモリ内容は以下のフォーマットで出力されます。

a d d r	0 1 2 3 4 5 6 7 8 9 a b c d e f
0xNNNNNNNN	c c c c c c c c c c c c c c c c

0xNNNNNNNN : アドレス

c : 1、0、または*。ビット位置が指定範囲外の場合は*を表示します。

メモリ内容出力後、2 addrの入力に戻ります。

<注意事項>

- メモリ内容を参照できる範囲はmdサブコマンドと同じです。
- 1 storageの指定で's'を指定し、2 addrの指定でバックアップファイルの存在しないアドレスを指定した場合は、エラーメッセージを表示し、2 addrの入力待ちとなります。

<名前>

mcp — メモリ内容のコピー

<形式>

mcp

```
1 storage(s,m,*) : {s}
                  {m}
                  {*}
                  {e}
                  {return}
2 s_addr : {addr1,len}
          {addr1-addr2}
          {*n}
          {e}
3 d_addr : {addr}
          {*n}
          {e}
```

(注) アンダーライン部はユーザが入力してください。

<機能説明>

mcpは、メモリの内容やバックアップファイルの内容を指定アドレスにコピーします。

1 storage(s,m,*)

- s : コピーの対象がバックアップファイルであることを指示します。
- m : コピーの対象が実機メモリであることを指示します。
- return : コピーの対象が実機メモリであることを指示します。
- * : バックアップファイル、実機メモリをコピーの対象とします。
- e : サブコマンドの終了を指示します。

2 s_addr

- addr1,len : コピー元先頭アドレスaddr1からlenで指定したバイト数をコピーすることを指示します。
- addr1-addr2 : コピー元先頭アドレスaddr1からコピー元最終アドレスaddr2までをコピーすることを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。
- e : サブコマンドの終了を指示します。

3 d_addr

- addr : コピー元先頭アドレスを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (1 ≤ n ≤ 2の範囲で指定できます)。
- e : サブコマンドの終了を指示します。

<注意事項>

- mcpサブコマンドは、3 d_addrのコピー先アドレス入力後、コピーを開始する前に、下記フォーマットで確認メッセージを表示します。

```

                コピー元アドレス範囲      コピー先アドレス範囲
copy : 0x*****-0x***** -> 0x*****-0x*****
memory data copy ok ? (y/n) :
    
```

storage種別に対応した
メッセージとなります。

storage種別=s : backup file
m : memory
* : memory and backup file

ここで、‘y’ または ‘Y’ が入力された場合、コピーを開始し、コピー開始、終了時に下記メッセージを表示します。これ以外の文字が入力された場合は、コピーを行わないで2 s_addrの入力に戻ります。

```

コピー開始時      ***** memory copy start *****
コピー終了時      ***** memory copy end *****
    
```

storage種別に対応したメッセージとなります。

- 奇数アドレスが指定された場合は、マイナス方向の偶数アドレスに補正します。
- メモリコピー終了後は、2 s_addrの入力に戻ります。
- 1 storageの指定で ‘s’ または ‘*’ を指定し、バックアップファイルの存在しないアドレスを指定した場合は、エラーメッセージを表示し、2 s_addr入力待ちとなります。
- mcpのメモリアクセス範囲は、下図に網掛けで示される範囲内となります。
mcpはmd、bs、bgと異なり、R700システムバス空間、S10空間はアクセス対象外となります。また、物理メモリにマッピングされていない空間はアクセスできません。

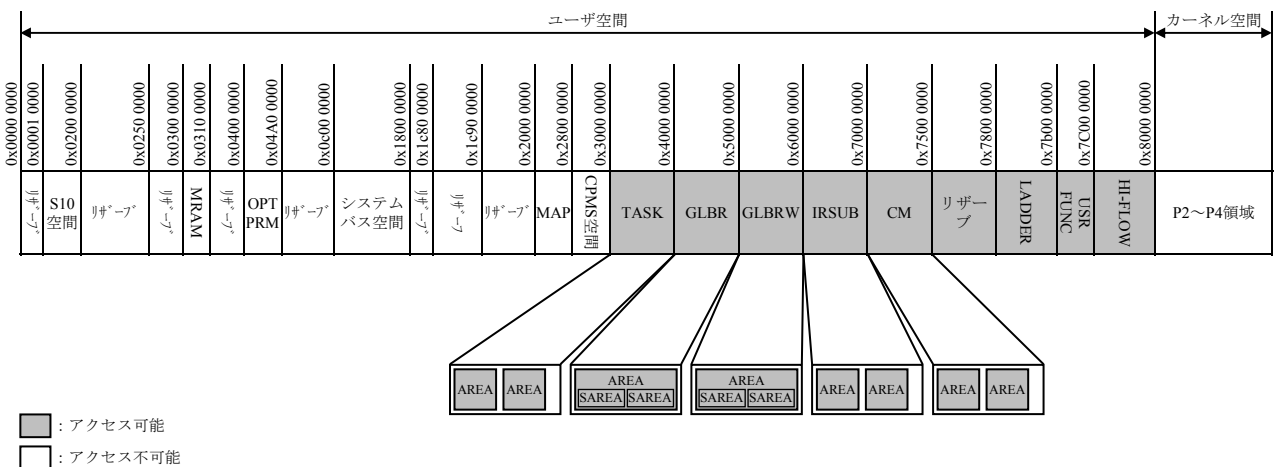


図2-6 メモリアクセス範囲

<名前>

mmv ー メモリ内容の移動

<形式>

mmv

```
1 storage(s,m,*) : {s}
                  {m}
                  {*}
                  {e}
                  {return}
2 s_addr : {addr1,len}
          {addr1-addr2}
          {*n}
          {e}
3 d_addr : {addr}
          {*n}
          {e}
```

(注) アンダーライン部はユーザが入力してください。

<機能説明>

mmvは、メモリ内容を移動します。移動元は0クリアします。

1 storage(s,m,*)

- s : 移動の対象がバックアップファイルであることを指示します。
- m : 移動の対象が実機メモリであることを指示します。
- return : 移動の対象が実機メモリであることを指示します。
- * : バックアップファイルと実機メモリの両方を移動の対象とします。
- e : サブコマンドの終了を指示します。

2 s_addr

- addr1,len : 移動元先頭アドレスaddr1からlenで指定したバイト数を移動することを指示します。
- addr1-addr2 : 移動元先頭アドレスaddr1から移動元最終アドレスaddr2までを移動することを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。
- e : サブコマンドの終了を指示します。

3 d_addr

- addr : 移動先先頭アドレスを指示します。
- *n : nに指定したプロンプト番号の前処理に戻ります (1 ≤ n ≤ 2の範囲で指定できます)。
- e : サブコマンドの終了を指示します。

<注意事項>

- mmvサブコマンドは、3 d_addrの移動先アドレス入力後、移動を開始する前に、下記フォーマットで確認メッセージを表示します。

```

                移動元アドレス範囲      移動先アドレス範囲
move : 0x*****-0x***** -> 0x*****-0x*****
memory data move ok ? (y/n) :
    
```

↑
storage種別に対応した
メッセージとなります。

storage種別=s : backup file
m : memory
* : memory and backup file

ここで、‘y’ または ‘Y’ が入力された場合、移動を開始し、移動開始、終了時に下記メッセージを表示します。これ以外の文字が入力された場合は、移動を行わないで2 s_addrの入力に戻ります。

```

移動開始時   ***** memory move start *****
移動終了時   ***** memory move end   *****
    
```

↑
storage種別に対応したメッセージとなります。

- 奇数アドレスが指定された場合は、マイナス方向の偶数アドレスに補正します。
- 指定できるアドレスの指定範囲は、mcpサブコマンドのメモリアクセス範囲と同じです。
- メモリ移動終了後は、2 s_addrの入力に戻ります。
- 1 storageの指定で ‘s’ または ‘*’ を指定し、バックアップファイルの存在しないアドレスを指定した場合は、エラーメッセージを表示し、2 s_addrの入力待ちとなります。

<名前>

mf — メモリへのパターン値設定

<形式>

mf

1 storage(s,m,*) : {s}

{m}

{*}

{e}

{return}

2 addr : {addr1,len [-l/-w/-b]}

{addr1-addr2 [-l/-w/-b]}

{*n}

{e}

3 data : {data}

{*n}

{e}

(注) アンダーライン部はユーザが入力してください。

<機能説明>

mfは、与えられたアドレス範囲に、一定のパターン値をセットします。

1 storage(s,m,*)

s : セット対象メモリがバックアップファイルであることを指示します。

m : セット対象メモリが実機メモリであることを指示します。

return : セット対象メモリが実機メモリであることを指示します。

* : セット対象メモリがバックアップファイルと実機メモリの両方であることを指示します。

e : サブコマンドの終了を指示します。

2 addr

addr1,len : セット先頭アドレスaddr1からlenで指定したケース数の範囲にセットすることを指示します。

addr1-addr2 : セット先頭アドレスaddr1からセット最終アドレスaddr2までの範囲にセットすることを指示します。

*n : nに指定したプロンプト番号の前処理に戻ります (n=1だけを指定できます)。

e : サブコマンドの終了を指示します。

-l : データ長を4バイトとします。

-w : データ長を2バイトとします。

-b : データ長を1バイトとします。

3 data

data : セットするデータを指示します。

*n : nに指定したプロンプト番号の前処理に戻ります (1 ≤ n ≤ 2の範囲で指定できます)。

e : サブコマンドの終了を指示します。

<注意事項>

- データ長を省略した場合は、このコマンド内で最後に指定したものを有効とします。
デフォルトは-1 (4バイト) です。
- mfサブコマンドは、3 dataのパターンデータ設定後、セットを開始する前に、下記フォーマットで確認メッセージを表示します。

セット先アドレス範囲	パターンデータ (16進)
write : 0x*****-0x*****	pattern data = 0x*****
memory data write ok ? (y/n) :	

↑
storage種別に対応した
メッセージとなります。

storage種別=s : backup file
m : memory
* : memory and backup file

ここで、‘y’ または ‘Y’ が入力された場合、パターンデータのセットを開始します。これ以外の文字が入力された場合は、パターンデータのセットを行わないで2 addrの入力に戻ります。

- 指定アドレスがデータ長の境界ではない場合、マイナス方向の境界へ補正します。
- 指定できるアドレスの指定範囲は、mcpのメモリアクセス範囲と同じです。
- メモリセット終了後は、2 addrの入力に戻ります。
- 1 storageの指定で ‘s’ または ‘*’ を指定し、2 addrの指定でバックアップファイルの存在しないアドレスを指定した場合は、エラーメッセージを表示し、2 addrの入力待ちとなります。

第8章 svdebug (オンラインデバッガ) とデバッグ支援コマンド

<名前>

el - エラーログの表示

<形式>

el [-u site] [-f {s|m|l}] [-logno] [+count] [-o fname]

<機能説明>

elは、svelogコマンドを起動し、エラーログを表示します。

elサブコマンドの詳細は、svelogコマンドのコマンド仕様を参照してください。

<名前>

ss - システムの状態表示

<形式>

ss

<機能説明>

ssは、svcpuctlコマンドを起動し、システムの状態を表示します。

ssサブコマンドの詳細は、svcpuctlコマンドのコマンド仕様を参照してください。

<名前>

st - 現在時刻の設定

<形式>

st [yyyy.mm.dd.hh:mt:ss]

<機能説明>

stは、S10VEの管理している現在時刻を設定します。指定するパラメータは以下のとおりです。

yyyy 年 (西暦4桁)

mm 月

dd 日

hh 時

mt 分

ss 秒

(注) 各時刻データは10進で入力します。

<結果>

OK(0) : 正常終了

<注意事項>

- サブコマンドのコマンド行の設定時刻 (yyyy.mm.dd.hh:mt:ss) を省略した場合は、下記メッセージを出力し入力待ちとなります。この状態で [e] または [Enter] キーを押した場合には、サブコマンドを終了します。

YYYY.MM.DD.HH:MT:SS :

- このサブコマンドが-sオプションによりダイレクト実行されたときは、エラーがあってもエラーメッセージは表示されません。

<名前>

gt - 現在時刻の表示

<形式>

gt

<機能説明>

gtは、S10VEの管理している現在時刻を表示します。

<結果>

以下の形式で現在時刻を表示します。

yyyy.mm.dd.hh:mt:ss

yyyy 年 (西暦4桁)

mm 月

dd 日

hh 時

mt 分

ss 秒

<名前>

br — ブレークポイントの設定/表示
 stickybr

<形式>

br [pname break1 ... break5 [-t|-s]]
 stickybr [pname break1 ... break5 [-t|-s]]

<機能説明>

brおよびstickybrは、ブレークポイントの設定および設定されているブレークポイントを表示します。

ブレークポイントを設定できるのは、タスクのプログラム (TEXT空間) および間接リンクサブプログラム (TEXT空間) だけです。

ブレークが発生したタスクは、WAIT状態となります。このとき、タスクの周期起動タイマは継続され、タイマ起動は実行されますが、ブレークポイントからの実行を再開しない限り、タスクはWAIT状態のままとなります。

ブレーク発生により停止しているタスクが存在する場合、その他のブレークポイントではブレークは発生しません。ブレーク発生により停止しているタスクが再開後、その他のブレークポイントが有効となります。

指定するパラメータは以下のとおりです。

- pname : ブレークポイントを設定するプログラム名称を指定します。
- break1~break5 : ブレークポイント (プログラム内相対アドレス) を指定します。
- t : 指定したプログラム名称がタスクのプログラム名称であることを示します。
- s : 指定したプログラム名称がサブプログラム名称であることを示します。

(注) -t、-sオプションを省略した場合は、タスク、サブプログラムの順に検索されます。

<結果>

ブレークポイントが正常に設定された場合は、下記メッセージを表示します。

break point set

name = プログラム名称 raddr = プログラム内相対アドレス object = 機械語命令パターン

pname, breakを指定しない場合は、現在設定されているブレークポイントを下記のように表示します。

break point

name = プログラム名称 raddr = プログラム内相対アドレス object = 機械語命令パターン

name = プログラム名称 raddr = プログラム内相対アドレス object = 機械語命令パターン ***BREAK***

·
·

↑
 現在ブレーク中の設定
 に表示されます。

他端末からブレークポイントを設定中の場合は、エラーメッセージを表示します。

<注意事項>

- ブレークポイントは、サイトごとに最大5箇所まで設定できます。
- ブレークポイントに達すると、以下のようなメッセージが表示されます。

break!!

tn = タスク番号 pname = プログラム名称 raddr = プログラム内相対アドレス

- rb、rd、rr、goなどのサブコマンドが失敗した場合は、brだけを発行してブレークポイントの状態を確認してください。これにより、開発系マシンのブレークポイント情報とS10VEの情報が異なっている場合に、開発系マシンの情報をS10VEに合わせるすることができます。
- S10VEの再立ち上げ後には、brサブコマンドによりS10VEの停止前に設定されていたブレークポイント情報は保存されません。この場合、brだけを発行して開発系マシンのブレークポイント情報をS10VEの情報に合わせてください。stickybrサブコマンドで設定したブレークポイントは、S10VEのリセットスタートによる再立ち上げでも解除されません。
- ブレークポイントは、RPCサーバタスク、システムイニシャルスタートタスク (SIST)、および組み込みサブルーチンには設定できません。
- このサブコマンドは、svdebug起動時に-debugオプションを指定した場合に使用できます。それ以外はエラーとなります。
- S10VEの再立ち上げやデバッガの異常終了により実行系マシンと開発系マシンのブレークポイントの設定が異なってしまった場合は、下記を実行してください。
brサブコマンドをパラメータなしで実行し、その後、ブレークポイントの設定が残っていた場合は、rbサブコマンドで解除してください。

<使用例>

以下にブレーク設定ポイントの設定手順を示します。

- 下記のprog1.c、prog2.cからprogというタスクプログラムを生成した場合のポイント①に、ブレークポイントを設定する場合を示します。

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;
}
```

- ・ソースコンパイル時に、“-SH=SO -l” オプションを指定してコンパイルすることで、アセンブラソースにC言語のソースファイルを挿入することができます。prog1.cのソース内にブレークポイントを設定するため、prog1.lstを参照します。

ブレークポイントを設定するC言語のソースは (1) なので、その直後のアセンブラ命令 (2) がC言語のソースに対応しています。

該当する命令のソース内 (prog1.c) でのオフセットは、0x00000026となります。

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 )
```

次ページへ続く

```

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

```


- svloadコマンドで実行プログラム生成時に、“-P” オプションを設定することで、マップファイルを生
成します。

マップのSECTION=Pを確認すると、prog1.objは、0x3003c014から始まることが分かります。

ブレークポイントを設定するソース内相対アドレスは0x00000014なので、ブレークポイントを設定する
アドレスは0x3003c028 (=0x3003c014 + 0x00000014) となります。

プログラム内相対アドレスは、ブレークポイント設定アドレスとプログラムの先頭アドレスから求めれ
ばよいので、0x00000028 (=0x3003c028 - 0x3003c000) となります。

prog.map

SECTION=P					
FILE=C:\WINDOWS\system32\cmd.exe					
		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	*

<名前>

rb - ブレークポイントの解除

<形式>

rb [pname break1 ... break5 [-t|-s]]

<機能説明>

rbは、現在設定されているブレークポイントを解除します。パラメータを指定しない場合、現在設定されているブレークポイントをすべて解除します。指定するパラメータは以下のとおりです。

- pname : ブレークポイントを設定するプログラム名称を指定します。
 - break1~break5 : ブレークポイント (プログラム内相対アドレス) を指定します。
 - t : 指定したプログラム名称がタスクのプログラム名称であることを示します。
 - s : 指定したプログラム名称がサブプログラム名称であることを示します。
- (注) -t、-sオプションを省略した場合は、タスク、サブプログラムの順に検索されます。

<結果>

正常にブレークポイントを解除した場合には、以下のようなメッセージを表示します。

break point reset

name = プログラム名称 raddr = プログラム内相対アドレス object = 機械語命令パターン

他端末からブレークポイントを設定中の場合は、エラーメッセージを表示します。

<注意事項>

- このサブコマンドは、svdebug起動時に-debugオプションを指定した場合に使用できます。それ以外はエラーです。
- ブレーク中のタスクが存在した場合、該当ブレークポイントの設定は解除できません。

<名前>

rd - レジスタの表示

<形式>

rd [-f|-h]

<機能説明>

rdは、ブレーク時の各レジスタの内容を表示します。パラメータを指定しない場合、浮動小数点レジスタは表示されません。指定するパラメータは以下のとおりです。

- f : 浮動小数点レジスタを実数で表示します。
- h : 浮動小数点レジスタを16進数で表示します。

<結果>

OK(0) : 正常終了

各レジスタの内容が下記のように表示されます。
異常終了時は、エラーメッセージが表示されます。

```
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*****
```

-fオプションが指定された場合は、以下のように浮動小数点レジスタを実数で表示します。

```
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**
```

-hオプションが指定された場合は、以下のように浮動小数点レジスタを16進数で表示します。

```
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*****
```

<レジスタ説明>

- SR : ステータスレジスタです。
- PC : プログラムカウンタです。
- GBR : グローバルベースレジスタです。ディスプレイメント付きGBR間接およびインデックス付きGBR間接の、アドレッシング使用するベースアドレスを格納するレジスタです。
- PR : プロシジャレジスタです。プロシジャレジスタはサブルーチンの呼び出しに使われます。実行されていたプログラムがサブルーチン呼び出し関係の末尾である場合、このレジスタに戻りアドレスが格納されています。
- MACH : システムレジスタ (積和上位レジスタ) です。MAC命令 (積和演算) の加算値およびMAC命令、MUL命令の結果を格納するために使用するレジスタです。演算結果が64ビット値だった場合、上位32ビットが格納されます。演算結果が32ビット値だった場合、32ビットが格納されます。
- MACL : システムレジスタ (積和下位レジスタ) です。演算結果が64ビット値だった場合、下位32ビットが格納されます。
- FPUL : 浮動小数点通信レジスタです。
- FPSCR : 浮動小数点ステータス/コントロールレジスタです。
- R0~R15
汎用レジスタです (R15がスタックポインタとして使われます)。
- FR0~FR15
単精度浮動小数点レジスタです。FPSCR.PR (31-0ビット値の19ビット目) =0の場合、FPRxx_BANK0の値。FPSCR.PR=1の場合、FPRxx_BANK1の値。

第8章 svdebug (オンラインデバッガ) とデバッグ支援コマンド

- XF0～XF15
単精度浮動小数点拡張レジスタです。FPSCR.PR (31-0ビット値の19ビット目) =0の場合、FPRxx_BANK1の値。FPSCR.PR=1の場合、FPRxx_BANK0の値。
- DR0～DR15
倍精度浮動小数点レジスタです。FPSCR.PR (31-0ビット値の19ビット目) =0の場合、FPRxx_BANK0の値。FPSCR.PR=1の場合、FPRxx_BANK1の値。
- XD0～XD15
倍精度浮動小数点レジスタです。FPSCR.PR (31-0ビット値の19ビット目) =0の場合、FPRxx_BANK1の値。FPSCR.PR=1の場合、FPRxx_BANK0の値。

<注意事項>

- このサブコマンドは、svdebug起動時に-debugオプションを指定した場合に使用できます。それ以外はエラーです。
- 浮動小数点データが以下のような場合は、実数表示を指定された場合でも、16進数に変換して表示します。また、16進数表示後に対応する文字列を表示します。

浮動小数点データ	文字列	表示例	
		単精度	倍精度
非数	Na	FR0 =0x7fffffff:Na	DR0 =0xfff 00000 0x00000001:Na
無限大	In	FR0 =0x7f800000:In	DR0 =0xfff 00000 0x00000000:In
表現できる最大値	Ma	FR0 =0x7f7fffff:Ma	DR0 =0x7fefffff 0xffffffff:Ma
表現できる最小値	Mi	FR0 =0xff7fffff:Mi	DR0 =0xffefffff 0xffffffff:Mi

<名前>

rr - レジスタの内容変更

<形式>

rr

register : rx

data : datax

(注) アンダーライン部はユーザが入力してください。

<機能説明>

rrは、ブレーク中のレジスタの内容を変更します。

rx : 内容を変更するレジスタの略称を指定します。

レジスタ略称は、rdサブコマンドで表示される名称を指定してください。

datax : 変更データを指定します (8進、10進、16進、実数の指定が可能)。

倍精度浮動小数点レジスタのデータ入力の実数でだけ可能です。

<結果>

OK(0) : 正常終了

異常終了時は、エラーメッセージが表示されます。

<注意事項>

- rrサブコマンドは、タスクがブレークポイントで中断しているときにだけ有効です。
- このサブコマンドは、svdebug起動時に-debugオプションを指定した場合に使用できます。それ以外はエラーです。

<名前>

go - ブレークポイントからの実行再開

<形式>

go

<機能説明>

goは、ブレークポイントで中断していたアドレスから、プログラムを再開させます。プログラムの再開後、該当ブレークポイントの設定は解除されます。

<結果>

OK(0) : 正常終了

異常終了時は、エラーメッセージが表示されます。

<注意事項>

- goサブコマンドは、タスクがブレークポイントで中断しているときにだけ有効です。
- このサブコマンドは、svdebug起動時に-debugオプションを指定した場合に使用できます。それ以外はエラーです。

<名前>

ld - バックアップファイルの内容をSIOVEのメモリに転送

<形式>

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}

<機能説明>

ldは、バックアップファイルの内容をSIOVEの主メモリに転送します。転送する対象は以下に示すオプションで指示します。

- t tname : tnameで指定されたプログラム本体をダウンロードし、SIOVE上でタスクを生成します。
tnameがマルチタスクとして生成されている場合は、SIOVE上でもマルチタスクとして生成されます。
- s sname : snameで指定されたサブプログラムをローディングします。
サブプログラム本体のダウンロードをします。また、間接リンクサブプログラムアドレステーブルのエントリまたは組み込みサブルーチン管理テーブルのエントリをダウンロードします。snameで指定されたサブプログラムがマルチエントリIRSUBとして登録されている場合は、該当IRSUBのエントリポイントに対応するすべての間接リンクサブプログラムアドレステーブルのエントリをダウンロードします。
- g gname : gnameで指定されたグローバルをローディングします。
グローバルデータをダウンロードします。間接リンクグローバルである場合は、間接リンクグローバルアドレステーブルのエントリもダウンロードします。
- a aname : anameで指定された分割領域の内容をローディングします。
anameで指定する分割領域はGLB領域、CM領域に定義されている分割領域でなければなりません。

- m addr,len : 先頭アドレス (addr) 、バイト数 (len) 指定でローディングすることを指示します。
指定するアドレスの範囲がアロケートされていない空間を含む場合は、ローディングしません。
- T [tno] : tnoで指定されたタスク番号のタスクをS10VE上で削除または生成します。
tnoが開発系マシン上で生成されているタスクである場合、タスクを生成します。
tnoが開発系マシン上で生成されていないタスクである場合、タスクを削除します。
tnoが省略された場合は、開発系マシン上の生成/削除操作がS10VEに反映されていないタスクの一覧を表示します。
- U [point,ent] : point、entで指定される組み込みサブルーチン管理テーブルのエントリをローディングします。
pointおよびentは以下のように指定します。
point : 組み込みサブルーチンの組み込み箇所を表す文字列であり、CPES、IES、EAS、INS、EXS、ABS、PCKS、MODES、WDTES、XEASのいずれかを指定します。
ent : エントリ番号を表す1から4の数値です。
point、entが省略された場合は、開発系マシン上での登録/削除操作がS10VEに反映されていない組み込みサブルーチン管理テーブルの一覧を表示します。
- S [sno] : snoで指定された間接リンクサブプログラムアドレステーブルエントリをローディングします。
snoが省略された場合は、開発系マシン上の登録/削除がS10VEに反映されていない間接リンクサブプログラムアドレステーブルの一覧を表示します。
- G [gno] : gnoで指定された間接リンクグローバルアドレステーブルをローディングします。
gnoが省略された場合は、開発系マシン上での登録/削除操作がS10VEに反映されていない間接リンクグローバルアドレステーブルの一覧を表示します。
- dcm : DCM上に確保された領域だけをローディングします。拡張オプションのため、S10VEでは利用できません。
- cm : CM上に確保された領域だけをローディングします。
- f fname : 指定ファイルfnameの内容をローディングします。
fnameにはsvサブコマンドで出力したファイルだけを指定できます。

<結果>

ldによってリソースのローディングを行った場合、ローディング終了時に状況を表示します。

ldオプションごとの結果について以下に説明します。

(1) -t tname (プログラムの個別ロード)

1行目は本体のアドレス、2行目以降はタスクの生成/削除によりダウンロードを行ったTCBのアドレスが表示されます(旧タスクが存在する場合は、1行目に旧TCB、2行目に本体、3行目に新しいTCBアドレスが表示されます)。

タスクがマルチタスクの場合は、ダウンロードするすべてのTCBのアドレスが表示されます。

```
address : 0x*****-0x*****
address : 0x*****-0x*****
```

(2) -s sname (サブプログラムの個別ロード)

ローディング対象のサブプログラムの種別により、結果表示が異なります。

● IRSUBの場合

ダウンロードしたアドレスの範囲が下記フォーマットで表示されます。

1行目は本体のアドレス、2行目以降は間接リンクサブプログラムアドレステーブルのアドレスが表示されます。

IRSUBがマルチエントリ化されている場合は、ダウンロードするすべての間接リンクサブプログラムアドレステーブルのアドレスが表示されます。

```
address : 0x*****-0x*****
address : 0x*****-0x*****
```

● 組み込みサブルーチンの場合

ダウンロードしたアドレスの範囲および組み込みサブルーチン管理テーブルのエントリが、下記フォーマットで表示されます。

addressはダウンロードした組み込みサブルーチン本体のアドレスを、point、entはダウンロードした組み込みサブルーチン管理テーブルのエントリを表す組み込み箇所およびエントリ番号を表示します。

```
address : 0x*****-0x*****
point,ent : POINT,N
```

POINTには、組み込み箇所を表す文字列のうちCPES、IES、EAS、INS、EXS、ABS、PCKS、MODES、WDTES、XEASのどれかを表示します。

Nには、エントリ番号を表す1~4の数値を表示します。

(3) `-g gname` (グローバルの個別ロード)

ダウンロードしたアドレスの範囲が下記フォーマットで表示されます。

間接リンクグローバルの場合は、間接リンクグローバルアドレステーブルのアドレスも表示されます。

`gname`が複数の間接リンクグローバルとして登録されている場合は、すべての間接リンクグローバルアドレステーブルのアドレスが表示されます。

```
address : 0x*****-0x*****
```

(4) `-a aname` (分割名称の個別ロード)

ダウンロードしたアドレスの範囲が下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

(5) `-m addr,len` (アドレス指定のロード)

ダウンロードしたアドレスの範囲が下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

(6) `-T tno` (タスク生成/削除のコントローラメモリへの反映)

- タスクの生成/削除のコントローラメモリへの反映を行った場合

タスクの生成/削除によりダウンロードを行ったTCBのアドレスが表示されます。

```
address : 0x*****-0x*****
```

- コントローラメモリに反映が必要なタスクの一覧表示を行った場合

`-T`オプションだけを指定した場合は、下記フォーマットでコントローラメモリに反映が必要なタスクの一覧を表示します。

TN	TNAME	PNAME	STATUS
tn	tname	pname	stat

上記一覧で、`tn`はタスク番号、`tname`はタスク名称、`pname`はプログラム名称、`stat`はタスクの管理状態を表します。管理状態の意味は、表2-13を参照してください。

タスクの管理状態が`unmatch`であり、開発系マシン上の管理とS10VE上の管理で、異なるTNが与えられている場合は、開発系マシン上のタスクのTNが表示されます。

表 2-13 リソースの管理状態

No.	状態	意味
1	non-exist	開発系マシンにもS10VEにも未登録の状態。
2	not-build	サブプログラムでロードされているが、buildされていない状態。
3	defined-POC	開発系マシンだけに登録されている状態。
4	defined	開発系マシンにもS10VEにも登録されている状態。
5	defined-CON	S10VEだけに登録されている状態。 S10VEにダウンロード後に開発系マシンから削除した状態。
6	unmatch	開発系マシンにもS10VEにも登録されているが、整合の取れていない状態。 開発系マシンで削除・再登録を行い、ダウンロードしていない状態。
7	non-exist2	defined-CON状態のサブプログラムで、S10VEメモリ上のbuild情報を削除したが、開発系マシンからはdloadしていない状態。
8	defined-CON2	defined-CON状態のサブプログラムで、開発系マシンではdloadしたが、S10VEのメモリにはまだbuild情報が残っている状態。

(7) -U point, ent (組み込みサブルーチン管理テーブルのダウンロード)

- 組み込みサブルーチン管理テーブルをダウンロードした場合

更新した組み込みサブルーチン管理テーブルのエントリは下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

- S10VEメモリに反映が必要な組み込みサブルーチン管理テーブル一覧表示を行った場合-Uオプションだけを指定した場合は、下記フォーマットでコントローラメモリに反映の必要な組み込みサブルーチン管理テーブルの一覧が表示されます。

```
POINT  ENT  SUBNAME  STATUS
pnt    eno  subname  stat
```

上記表示で、pntは組み込み箇所、enoはエントリ番号、subnameはサブプログラム名、statは組み込みサブルーチン管理テーブルのエントリの管理状態を表します。管理状態の意味については、表 2-13を参照してください。

組み込みサブルーチン管理テーブルのエントリの管理状態には、defined-POC、defined-CON、not-build、non-exist2、unmatchのいずれかが表示されます。

組み込みサブルーチン管理テーブルのエントリの管理状態がunmatchであり、開発系マシン上の管理とコントローラ上の管理で、異なる組み込み箇所が与えられている場合は、開発系マシン上の組み込み箇所が表示されます。

(8) -S sno (間接リンクサブプログラムアドレステーブルのダウンロード)

- 間接リンクサブプログラムアドレステーブルをダウンロードした場合

ダウンロードした間接リンクサブプログラムアドレステーブルのアドレスは、下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

- S10VEメモリに反映が必要な間接リンクサブプログラムアドレステーブルの一覧表示を行った場合

-Sオプションだけを指定した場合は、下記フォーマットでS10VEメモリに反映の必要な間接リンクサブプログラムアドレステーブル一覧が表示されます。

IRSBNO	ENTNAME	SUBNAME	STATUS
sno	entname	subname	stat

上記一覧で、snoはIRSUB番号、entnameはエントリ名、subnameはサブプログラム名、statはIRSUBTのエントリの管理状態を表示しています。管理状態の意味については、表2-13を参照してください。

IRSUBTのエントリの管理状態がunmatchであり、開発系マシン上の管理とS10VE上の管理で、異なるIRSUB番号が与えられている場合は、開発系マシン上のIRSUB番号が表示されます。また、サブプログラム名が異なる場合は、開発系マシン上のサブプログラム名が表示されません。

(9) -G gno (間接リンクグローバルアドレステーブルのダウンロード)

- 間接リンクグローバルアドレステーブルのダウンロード

ダウンロードした間接リンクグローバルアドレステーブル (間接リンクグローバルアドレステーブル) のアドレスが下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

- S10VEメモリに反映が必要な間接リンクグローバルアドレステーブルの一覧表示を行った場合、`-G`オプションだけを指定した場合は、下記フォーマットでS10VEメモリに反映の必要な間接リンクグローバルの一覧が表示されます。

IRGLBNO	ENTNAME	SNAME	STATUS
gno	entname	sname	stat

上記一覧で、`gno`は間接リンクグローバル番号、`entname`はエントリ名、`sname`は細分割領域名、`stat`は間接リンクグローバルアドレステーブルのエントリの管理状態を表示します。管理状態の意味については表2-13を参照してください。

間接リンクグローバルアドレステーブルのエントリの管理状態が`unmatch`であり、開発系マシン上の管理とS10VE上の管理で、異なる間接リンクグローバル番号が与えられている場合は、開発系マシン上の間接リンクグローバル番号が表示されます。

また、細分割領域名が異なる場合は、開発系マシン上の細分割領域名が表示されます。

(10) `-cm` (CMのダウンロード)

ダウンロードしたアドレスの範囲は下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

CMへのダウンロードはS10VEがSTOP状態のときだけ可能です。

CMへのダウンロードは、実行したサイトが持つCMのバックアップだけとなります。

(11) `-f fname` (指定ファイルからのロード)

ダウンロードしたアドレスの範囲が下記フォーマットで表示されます。

```
address : 0x*****-0x*****
```

<注意事項>

- リソースの入れ換え時には、リソースを入れ換えても問題が発生しないように、リソースを参照するタスクをすべてabortするなどの操作が必要です。
- -aオプション、-mオプションでリソースをダウンロードした場合、-T、-U、-S、-Gオプションで管理テーブルをダウンロードおよびタスクを生成／削除してください。
- -fオプションでファイル名を指定しないで、バックアップファイルの存在しないリソースまたは領域を指定した場合はエラーメッセージを表示し、サブコマンドを終了します。
- 障害回復処理をスキップした場合、svrplコマンドで一括ロードするまで、ldサブコマンドによる個別ロード機能は使用できなくなります。
- ブレークポイントを設定しているプログラムまたはサブプログラムはダウンロードできません。

<名前>

sv - SIOVEのメモリの内容をバックアップファイルに転送

<形式>

```
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]}
```

<機能説明>

svは、SIOVEのメモリの内容をバックアップファイルに転送します。転送する対象は以下に示すオプションで指示します。

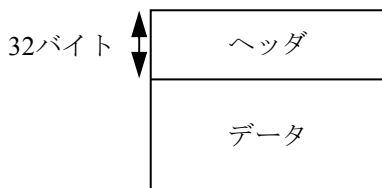
- t tname : tnameで指定されたプログラムのtext部およびdata部を転送します。
- s sname : snameで指定されたサブプログラムのtext部およびdata部を転送します。
- g gname : gnameで指定された細分割領域の内容を転送します。
- a aname : anameで指定された分割領域の内容を転送します。
anameで指定する領域はGLB領域、CM領域に定義された分割領域にしてください。
- m addr,len : 先頭アドレス (addr) 、バイト数 (len) 指定で転送します。
指定するアドレスの範囲がマッピングされていない空間を含む場合は、転送しません。
- f fname : 指定ファイルfnameに転送します。
この指定を省略した場合は、バックアップファイルに転送します。
セーブ途中に異常が発生した場合は指定ファイルを削除して終了します。

<結果>

転送したアドレスの範囲が下記フォーマットで表示されます。

address : 0x*****-0x*****

バックアップファイル以外に転送した場合のファイルのフォーマットを以下に示します。



32バイトのヘッダ+バイナリデータです。
ヘッダの内容は下記文字列となります。

```
svdebug△*****△*****¥0¥0...¥0¥0
          先頭アドレス   容量
```

*****は8桁の16進数、¥0はNULL (0) を、
△は空白文字 (0X20) を表します。

<注意事項>

- CMのデータをバックアップファイルに転送する場合、転送できるのは自サイトに割り当てられたCMだけとなります。
- -fオプションでファイル名称を指定しないで、バックアップファイルが存在しない領域を指定した場合は、エラーメッセージを出力し、サブコマンドを終了します。

<名前>

cm - バックアップファイルとSIOVEメモリの内容比較

<形式>

```
cm {-t tname}
cm {-s sname}
cm {-g gname}
cm {-a aname}
cm {-m addr,len}
cm {-f fname}
```

<機能説明>

cmは、バックアップファイルとSIOVEの主メモリの内容を比較します。比較する対象は以下に示すオプションで指示します。

- t tname : tnameで指定されたプログラムのtext部およびdata部を比較します。
- s sname : snameで指定されたサブプログラムのtext部およびdata部を比較します。
- g gname : gnameで指定された細分割領域の内容を比較します。
- a aname : anameで指定された分割領域の内容を比較します。
anameとして指定する領域は、GLB領域、CM領域に定義された分割領域にしてください。
- m addr,len : 先頭アドレス (addr)、バイト数 (len) 指定で比較します。
指定するアドレスの範囲がアロケートされていない空間を含む場合は、比較しません。
- f fname : 指定ファイルfnameとメモリの内容を比較します (svで出力したファイルだけ使用できます)。

<結果>

比較結果が正常の場合、アドレス範囲を下記フォーマットで表示します。

```
address : 0x*****-0x*****
++ compare OK ++
```

比較で違いが見つかった場合には、ワード (2バイト) 単位で以下のように表示します。

```
address : 0x*****-0x*****
address = 0x***** memory data = 0x**** backup data = 0x****
```

<注意事項>

- CMのデータとバックアップファイル内データは、自サイトに割り当てられたCMだけで比較できません。
- -fオプションでファイル名称を指定しないで、バックアップファイルが存在しない領域を指定した場合は、エラーメッセージを出力し、サブコマンドを終了します。

<名前>

dr — DHP記録許可

<形式>

dr

<機能説明>

drは、svdhpコマンドを起動し、DHPの記録を許可モードにします。

drサブコマンドの詳細は、svdhpコマンドの“-on” オプションの仕様を参照してください。

<名前>

ds — DHP記録禁止

<形式>

ds

<機能説明>

dsは、svdhpコマンドを起動し、DHPの記録を禁止モードにします。

dsサブコマンドの詳細は、svdhpコマンドの“-off” オプションの仕様を参照してください。

第8章 svdebug (オンラインデバッガ) とデバッグ支援コマンド

<名前>

svdhp — DHPの表示

<形式>

svdhp [-u site] [+count] [-on|-off] [-o fname] [-f fname] [-all [fname]]

<機能説明>

svdhpサブコマンドは、svdhpコマンドを起動し、DHPを表示します。

svdhpサブコマンドの詳細は、svdhpコマンドを参照してください。

<名前>

svadm — アドレスに対するリソース名称の表示

<形式>

svadm [addr] [-u site]

<機能説明>

svadmサブコマンドは、svadmコマンドを起動し、指定した論理アドレスに対して名称などの情報を表示します。

svadmサブコマンドの詳細は、svadmコマンドを参照してください。

<名前>

si - スタック初期化

<形式>

si tn1[-tn2][,data]

si tname[,data]

<機能説明>

siは、指定されたタスクのスタックを固定パターンで初期化します。指定するパラメータは以下のとおりです。

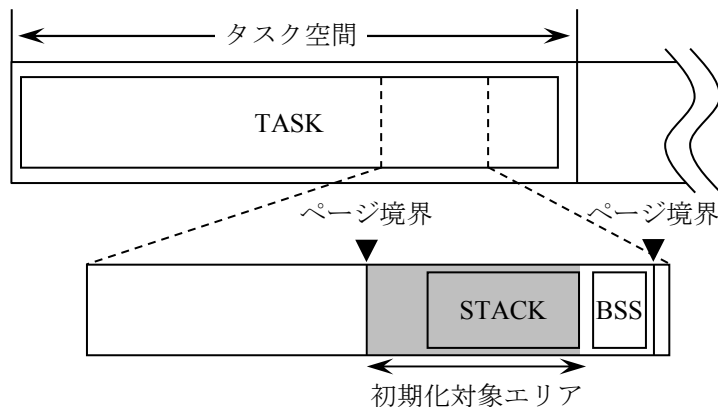
- tn1 : 先頭タスク番号 (1~最大タスク番号)
- tn2 : 最終タスク番号 (tn1~最大タスク番号)
- tname : タスク名称
- data : 初期化データ (0~9、a~f)

<結果>

OK(0) : 正常終了

<注意事項>

- 初期化データ指定 (data) を省略した場合、すべてfで初期化します。
- 初期化の対象となるタスクは、DORMANT状態にしておいてください。
- スタックの初期化は、指定したタスクのスタックが存在するページ領域のうち、ページ先頭アドレスからスタック最終アドレスまでの範囲を対象とします (下図参照)。



- このサブコマンド起動時に、パラメータを指定しなかった場合または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。また、この状態で [e] または [Enter] キーを押した場合には、サブコマンド処理を終了します。

```
input tn1[-tn2] [,data] or tname [,data]
:
```

<名前>

sp - スタック使用量の表示

<形式>

sp tn1[-tn2][,data]

sp tname[,data]

<機能説明>

spは、指定されたタスクのスタック使用量を表示します。指定するパラメータは以下のとおりです。

- tn1 : 先頭タスク番号 (1~最大タスク番号)
- tn2 : 最終タスク番号 (tn1~最大タスク番号)
- tname : タスク名称
- data : チェックパターン指定 (0~9, a~f)

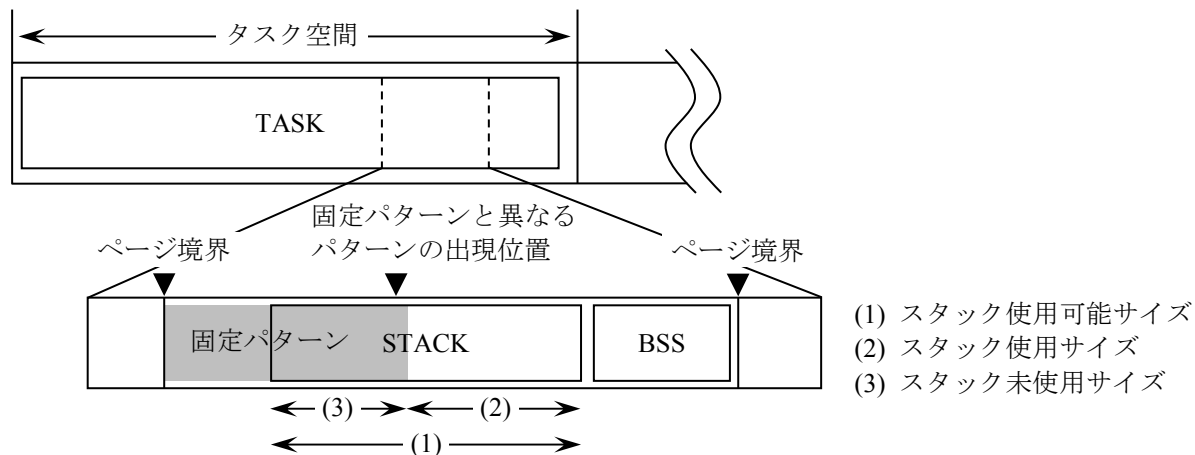
<結果>

タスクのスタック使用量は、下記フォーマットで表示します。

<u>tn=***</u>	<u>total:*****bytes</u>	<u>use:*****bytes</u>	<u>rest:*****bytes</u>
タスク番号	スタック使用可能サイズ	スタック使用サイズ	スタック未使用サイズ

<注意事項>

- チェックパターン指定 (data) には、siの初期化データに対応した値を指定してください。
チェックパターン指定 (data) を省略した場合、0xfが指定されたものとして扱います。
- スタック使用量は、指定したタスクが使用するスタックのページ領域に対して、チェックパターン指定 (data) で指定した値と、異なるパターンが現れたアドレスを基に算出します。このため、スタックの先頭が、チェックパターンと同一のパターンであった場合、正しいスタック使用量の表示はできなくなります。
- spが表示する内容と、タスクの動作空間との対応を下図に示します。



- このサブコマンド起動時に、パラメータを指定しなかった場合または正しくない引数を指定した場合は、下記メッセージを出力し “:” のあとでパラメータの入力待ちとなります。
また、この状態で [e] または [Enter] キーを押した場合には、サブコマンド処理を終了します。

```
input tn[-tn2] [-data] or tname [-data]
```

```
:
```


<名前>

ps - デバッグ文の表示開始

<形式>

ps

<機能説明>

psは、プログラム内のrs_printf()で出力するメッセージの端末表示を開始することを指示します。

psサブコマンドの実行前に出力されたデバッグ文は表示されません。

rs_printf()は、「付録B ライブラリ」を参照してください。

<結果>

処理が正常に終了後、デバッグ文の表示を開始します。

<注意事項>

デバッグ文を格納するバッファが足りなくなった場合、デバッグ文が出力されない場合があります。

<名前>

pe - デバッグ文の表示終了

<形式>

pe

<機能説明>

peは、プログラム内のrs_printf()で出力するメッセージの端末表示を終了することを指示します。

rs_printf()は、「付録B ライブラリ」を参照してください。

<結果>

処理が正常に終了後、デバッグ文の表示を終了します。

<名前>

ver - CPMSのバージョン表示

<形式>

ver [-m|-s]

<機能説明>

verは、CPMS、KROMのバージョンを表示します。オプションには、以下のものがあります。

-m : S10VEメモリ上のバージョン情報を表示します。

-s : 開発系マシン上のバージョン情報を表示します。

オプションを省略した場合は、両方のバージョン情報を出力します。

<結果>

処理が正常に終了したとき、CPMS、MPファームのバージョンが下記フォーマットで表示されます。

```
Secondary version:
CPMS      Ver.** Rev.** XXXX
Memory version:
CPMS      Ver.** Rev.** XXXX
KROM      Rev.**
```

<注意事項>

- 開発系マシン上のバージョン表示を行った場合は、MPファームのバージョンは表示されません。
- MPファームのバージョン表示はレビジョンを示す文字列だけが表示されます。
- CPMSのバージョン表示のあとに“-A”など、任意の文字列XXXXが表示されます。
文字列が設定されていない場合は、空白を表示します。
- CPMSがS10VEにダウンロードされていない場合は、“Ver.-- Rev.--”と表示されます。

<名前>

lbr - ラダーのブレークポイントの設定/表示

<形式>

lbr addr [tn]

<機能説明>

ラダープログラムにブレークポイントを設定します。設定可能なブレークポイントは1つのみです。addrにはブレークポイントを設定するLADDER空間のアドレスを指定してください。

tnは特定のタスクから実行したラダープログラムに対してのみブレークポイントを設定するときに指定します。tnを省略または、tnに0を指定すると、すべてのタスクから実行したラダープログラムにブレークポイントを設定します。

ラダープログラムはブレークポイントに到達すると、ブレークポイントの命令を実行する直前で処理を中断します。ラダーのプログラムのブレーク中はすべてのラダープログラムの実行が抑止されます。

addr, tnを省略すると現在設定されているブレークポイントと、ブレークの発生状況を表示します。

<結果>

正常にブレークポイントを設定した場合、以下のメッセージを表示します。

```
break point set
addr = 0xXXXXXXXX tn = xxx
```

0xXXXXXXXX : ブレークポイントを設定したアドレス

xxx : ブレークポイントの到達を検出するタスクのタスク番号

設定中のブレークポイント表示の場合、現在設定中のブレークポイント以下のように表示します。

```
break point
addr = 0xXXXXXXXX tn = xxx
```

0xXXXXXXXX : ブレークポイントが設定されているアドレス

xxx : ブレークポイントの到達を検出するタスクのタスク番号

ブレーク中のラダープログラムがある場合、以下のメッセージを表示します。

```
break on 0xXXXXXXXX (tn = xxx)
0xXXXXXXXX 0xYYYYYYYY
```

0xXXXXXXXX : ブレーク中のアドレス

xxx : ブレークポイントの到達を検出したタスクのタスク番号

0xYYYYYYYY : 次に実行される命令

<名前>

lrb - ブレークポイントの解除

<形式>

lrb

<機能説明>

ブレークポイントの設定を解除します。

<結果>

正常にブレークポイントの設定を解除した場合、以下のメッセージを表示します。

```
break point reset  
addr = 0XXXXXXXXX tn = xxx
```

0XXXXXXXXX : ブレークポイントが設定されているアドレス
xxx : ブレークポイントの到達を検出するタスクのタスク番号

<名前>

lrd - ラダープロセッサのレジスタ表示

<形式>

lrd [-h]

<機能説明>

ブレーク中のラダープログラムのレジスタ内容を表示します。-hオプションを指定すると、浮動小数点レジスタを16進数で表示します。

<結果>

```

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*****
    
```

浮動小数点データが以下のような場合は、16進数に変換して表示します。また、16進数表示のあとに対応する文字列を表示します。

浮動小数点データ	文字列	表示例
非数	Na	FR0 =0x7fffffff:Na
無限大	In	FR0 =0x7f800000:In
表現できる最大値	Ma	FR0 =0x7f7fffff:Ma
表現できる最小値	Mi	FR0 =0xff7fffff:Mi

-hオプションが指定された場合は、以下のように浮動小数点レジスタを16進数で表示します。

```

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*****
    
```

<名前>

lrr - ラダープロセッサのレジスタ変更

<形式>

lrr

register : reg

data : datax

(注) アンダーライン部はユーザが入力してください。

<機能説明>

ブレーク中のラダープログラムのレジスタ内容を書き換えます。

reg : 内容を変更するレジスタの名称を指定します。

レジスタの名称はlrdで表示される名称を指定してください。

datax : 変更データを指定します。8進、10進、16進で指定できます。

浮動小数点レジスタは実数で指定します。

<結果>

OK(0) : 正常終了

NG(≠0) : マクロエラー

(≠0) の部分はマクロのリターンコードが表示されます。

<名前>

lgo - ラダープログラムの実行再開

<形式>

lgo

<機能説明>

ブレーク中のラダープログラムの実行を再開します。実行を再開してもブレークポイントは解除されません。

<結果の表示>

OK(0) : 正常終了。

no break : ブレーク中のラダープログラムがありません。

<名前>

s - ラダープログラムのステップ実行

<形式>

s

<機能説明>

ブレーク中のラダープログラムをステップ実行します。ステップ実行によりラダープログラムが終了した場合、ブレークにより実行を抑制されていたラダープログラムは実行を再開します。

<結果の表示>

正常にステップを実行した場合、以下のメッセージを表示します。

0xXXXXXXXX 0xYYYYYYYY

0xXXXXXXXX : 次に実行する命令のアドレス

0xYYYYYYYY : 次に実行される命令

異常終了時は、以下のメッセージを表示します。

no break : ブレーク中のラダープログラムがありません。

<名前>

help — サブコマンド一覧表示

<形式>

help

<機能説明>

helpサブコマンドは、svdebugのサブコマンド一覧を表示します。

サブコマンド名と機能概要が、以下のフォーマットで表示されます。

<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

<名前>

q — デバッガの終了

<形式>

q

<機能説明>

qは、デバッガを終了させます。ただし、ブレークポイントが設定されている場合には、設定されているブレークポイントを表示してキー入力待ちになります。

<注意事項>

“ブレークポイントが設定されています”と表示された場合、ブレーク中の場合は、goサブコマンド、ブレーク中でない場合はrbサブコマンドでブレークポイントを解除してから、このサブコマンドを再発行してください。

<名前>

! — svdebug実行時の開発系マシン上のコマンド実行

<形式>

! 開発系マシン上のコマンド

<機能説明>

!以降を開発系マシン上のコマンドとして実行します。

<名前>

svelog - エラーログ情報出力

<形式>

```
svelog [-u site] [-f {s|m|l}] [-logno] [+case] [-d fname] [-o fname]
```

```
}
```

エラーログ1画面分表示

```
}
```

```
{ p }
```

```
{ - }
```

```
{ ±nl }
```

```
{ n }
```

```
{ 何も入れず }
```

```
{ q }
```

<機能説明>

svelogは、SIOVE内のエラーログバッファからエラーログ情報を読み出し、エラーログ情報を出力します。オプションには以下のものがあります。

- u site : 処理対象となるサイト名称を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
- f {s|m|l} : エラーログ情報の出力形式を指定します。
以下の形式があります。省略時は“m”になります。
s : エラー情報を簡略化した短い形式で出力します。
m : エラー情報をすべて出力します。
l : エラー情報に加えDHPトレース情報も出力します。
- logno : lognoで指定されたログ番号のエラーログ情報を出力します。
- +case : 表示ログケース数を指定します。このオプション省略時は、最新のケースから順にすべてのエラーログ情報を表示します。
- d fname : 画面操作履歴 (オペレーション結果) を格納するファイルを指定します。
同名のファイルがすでに存在した場合、そのファイルに画面操作履歴を追加して格納します。
- o fname : エラーログ情報を格納するファイルを指定します。
同名のファイルがすでに存在した場合、そのファイルを消去し、新しいファイルを作ります。

<使用上の留意点>

- svelogは、CPUがRUN/STOP状態において動作できます。
- -lognoで指定されたログ番号が、最新のエラーログ情報より大きい場合は、最新のエラーログ情報が表示されます。
- -logno、+caseが同時に指定された場合、lognoで指定されたログ番号のケースから+caseで指定されたケース数のログ情報を表示します。
- デフォルトフォーマットは、SIOVEでは“-fm”です。

<終了コード>

svelogは、次の終了コードを返します。

- 0 : 正常終了
- 1 : パラメータエラー
- 2 : 通信エラー
- 3 : [Ctrl] + [C] キーを押すことによって中断

<名前>

svdhp - DHPトレース情報の表示

<形式>

svdhp [-u site] [+count] [-on|-off|-stat] [-d fname] [-o fname] [-all [fname] | -f fname] [-freeze]

DHPの1画面分表示

{ p }

{ - }

{ ±nl }

{ n }

{ 何も入れず }

{ q }

<機能説明>

svdhpは、S10VE内DHPトレースバッファに記録されているDHPトレース情報を時刻の新しい順に表示します。オプションには以下のものがあります。

- u site : 処理対象となるサイト名称を指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。
- +count : countで指定されたトレース情報を出力します。
このオプション省略時は、すべてのトレース情報を出力します。
- on : DHPの記録を許可モードにします。
- off : DHPの記録を禁止モードにします。
- stat : DHPの記録モードを表示します。記録許可モードのときはDHP ON、記録禁止モードのときは、DHP OFFと表示します。
- all [fname] : CP、HPのDHPログを同時に取り出して表示します。
このオプションを指定すると、データは表示されなくて、CP、HPの各データを別々のファイルに出力します。ファイル名はfnameに指定された名称にそれぞれ“_cp.txt”、“_hp.txt”を付加して出力します。また、ファイル名は省略することができます。省略された場合のファイル名はそれぞれ、“CPサイト名.txt”、“HPサイト名.txt”とします。
このオプションは、-fオプション、-freezeオプションと同時に指定できません。
- d fname : 画面操作履歴（オペレーション結果）を格納するファイルを指定します。
同名のファイルがすでに存在した場合、そのファイルに画面表示を追加して格納します。
- o fname : DHP表示結果を格納するファイルを指定します。
同名のファイルがすでに存在した場合、そのファイルを消去し、新しいファイルを作ります。
- f fname : DHPログ入力ファイル名称を指定します。
入力ファイルは、dhpreadマクロでDHPログをGLBに格納し、デバッガのsvサブコマンドでGLBのDHPログを格納したファイルだけが対象となります。-allオプション、-freezeオプションと同時に指定できません。

- freeze : CPU時間を占有せずにDHPトレース情報を収集します。
このオプションでDHPトレース情報を収集する場合、収集中は記録モードが禁止モードとなるため、収集中のDHPトレース情報は記録されません。DHPトレース情報収集後、収集前の記録モードが許可モードであれば、DHPの記録を再開します。禁止モードであった場合はDHPの記録は停止したままです。DHPトレース情報収集後のDHP記録モードについては、-statオプションで確認してください。
-allオプションまたは-fオプションと同時に指定することはできません。

また、DHPトレース情報の表示は、以下に示すように閲覧用のコマンドによって制御します。

- p, 何も入れず : 次ページ表示
- : 前ページ表示
- ±nl : nl行前 (-) または後 (+) の行から表示します。
- n : n番目の行から表示します。
- q : DHP表示を終了します。

svdhpで出力する情報を以下に示します。

```

Debugging helper trace list      [X X X X X]   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 xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
 ①   ②           ③           ⑤ ⑥ xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
                                   ④
```

- ① DHPトレース情報の表示番号
- ② トレース時刻
tt.tttttt
| |
秒 1マイクロ秒まで出力
- ③ トレースポイント種別
- ④ トレースデータ (16進で出力)
- ⑤ タスク番号
- ⑥ 優先レベル
- ⑦ サイト名または-fオプション指定時のファイル名
- ⑧ プロセッサ種別

<使用上の留意点>

- svdhpは、CPUがRUN状態のときだけ動作できます。
- -on、-offオプション指定時に、DHPトレース情報は出力しません。

<終了コード>

svdhpは、次の終了コードを返します。

0 : 正常終了

1 : パラメータエラー

2 : 通信エラー

3 : [Ctrl] + [C] キーを押すことによって中断

<名前>

svcpunow - PU負荷率の表示

<形式>

svcpunow [-u site] [-t second]

<機能説明>

svcpunowは、指定サイト (PU) のIDLE時間の累積と時刻を取り込み、PU負荷率を表示します。

(計算式)

PU負荷率 = (測定時間 - IDLE時間) / 測定時間

オプションには以下のものがあります。

-u site : 対象となるサイトを指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

-t second : PU負荷率測定時間を秒 (1~3600) で指定します。

省略時のデフォルトは1秒です。

<使用上の留意点>

svcpunowコマンドがすでに実行されているときに、再びこのコマンドを実行しても要求は受け付けられません。

<終了コード>

svcpunowは、次の終了コードを返します。

0 : 正常終了

1 : 異常終了

2 : 通信異常

3 : [Ctrl] + [C] キーを押すことによって中断

<出力フォーマット>

出力結果は、下記ようになります。

```
2018/02/07 17:57:33 SITE=0001cp ** 1 second wait **
CPU(0001cp) load ratio = 0.06%
```


<名前>

svtimex — タスク稼働率表示

<形式>

```
svtimex [-u site] [tname] [-t second]
        [tn]
```

<機能説明>

svtimexは、測定時間内におけるタスクの実行回数、実行時間の累積と時刻を取り込んだタスク稼働率を表示します。

オプションには以下のものがあります。

-u site : 対象となるサイトを指定します。このオプション省略時は、環境変数“RSSITE”に設定されたサイトに対して処理します。

tn : タスク番号 (1~300) を10進または16進 (0xを前置) で指定します。

tname : タスク名称を指定します。

tnまたはtnameを省略すると会話形式となり最初に測定時間の入力を促します。ここで、1~86400の間で測定時間を入力すると、次にタスク名称または番号の入力を促します。ここでの入力は最大10タスク分の設定ができます。svtimexコマンドを実行させたいときは何も入力しないで [Enter] キーを押してください。

-t second : タスク稼働率測定時間を秒 (1~86400) で指定します。
省略時のデフォルトは1秒です。

<使用上の留意点>

- -tオプションによって測定時間を指定した場合は、タスク番号 (tn) またはタスク名称 (tname) を合わせて指定してください。
- svtimexコマンドがすでに実行されているときに、再びこのコマンドを実行しても要求は受け付けられません。
- tnオプションとtnameオプションは、同時には指定できません。会話形式によるタスク名称またはタスク番号の指定は最大10個まで指定できます。

<終了コード>

svtimexは、次の終了コードを返します。

0 : 正常終了

1 : 異常終了

2 : 通信異常

3 : [Ctrl] + [C] キーを押すことによって中断

<出力フォーマット>

出力結果は、下記のようになります。

```
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
```

このページは白紙です。

付録

付録A プログラムで使用できる名称

あらかじめシステムで用意されたサブルーチンの名称と同一名称のプログラムを使用する場合は、注意が必要です。システムで用意されたサブルーチンは、すべてライブラリファイルに納められています。svloadにおいて-lオプションを指定すると簡単に結合できます。しかし、システムのサブルーチンと同一名称である場合、そのサブルーチンを定義するオブジェクトファイルをsvloadで引数に指定しないとライブラリファイルから同一名称のサブルーチンが結合されてしまいます。

以下に各システム用のライブラリファイルとそこで定義されている名称を示します。名称が重複しないようにプログラミングしてください。

重複した名称を使用する場合、ライブラリファイルの指定順序を結合したいオブジェクトファイルのあとにすればライブラリファイルから結合されません。

以降にシステムで用意されたサブルーチンをライブラリ別に示します。

なお、_（アンダーライン）で始まる名称はシステムで予約した名称ですので使用しないでください。ライブラリの構成は下記のようになっています。

ライブラリ名	ライブラリの内容	備考
libsh4nbmzz.lib	C言語用のサブルーチン群 コンパイラ：バージョン9 非正規化数：非正規化数 値の丸め方：切り捨て	詳細はバージョン9のshcコンパイラマニュアルを参照してください。
libsh4nbmdn.lib	C言語用のサブルーチン群 コンパイラ：バージョン9 非正規化数：0 値の丸め方：切り捨て	
libfirad.lib	間接リンクアドレス参照サブルーチン群	詳細は各マニュアルを参照してください。
libcpms.lib	CPMSマクロリンケージサブルーチン群	
libsysctl.lib	システム制御用サブルーチン群	
libcyem.lib (*1)	サイクリック通信サブルーチン群	
libnet.lib	ソケット通信サブルーチン群	
libcrs.lib	IEEE浮動小数点処理環境制御サブルーチン群	

(*1) S10VEではIRSUB提供

<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
isctrl	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_strcmp1	quick_strcpy1	rand
realloc	rewind	scanf	setbuf	setjmp
setjmp_a	setvbuf	SFTRL1	sin	sinf
sinh	sinhf	sml_buf	sprintf	sqrt
sqrtf	srand	sscanf	streat	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
---------	---------

付録A プログラムで使用できる名称

<libcpmns.lib>

abort	arsum	asusp	atmadd	atmand
atmcas	atmor	atmswap	atmtas	atmxor
cfread	cfwrite	chap	chkbmem	chaktaer
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				

<libcym.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		

付録B ライブラリ

(1) ライブラリファイルの指定条件

ライブラリファイルをsvloadで指定する場合、表A-1に示すようにライブラリを指定してください。

表A-1 ライブラリの指定条件

条件	ライブラリ名	svloadでの指定方法	備考
作成したプログラムがC言語で記述されている場合（付録Aに示す該当のライブラリ内の関数を使用している場合）	libcrs.lib libnet.lib	-lcrs -lnet	
作成したプログラムでCPMSマクロを使用している場合	libcpms.lib libsysctl.lib	-lcpms -lsysctl	「S10VE ソフトウェアマニユアル CPMS 概説&マクロ仕様（マニュアル番号SEJ-3-201）」を参照してください。
間接リンクアドレスを参照している場合、または間接リンクサブルーチンを参照している場合	libfirad.lib	-lfirad	「(3) 間接リンクアドレス参照サブルーチン」を参照してください。
ユーザ専用のライブラリを使用する場合	ユーザライブラリ	-l文字列またはライブラリ名	

(2) ライブラリの指定順序

svloadでライブラリを指定する場合、下記の点に注意してください。

- 共通サブルーチンを含むライブラリはできるだけあとの方で指定してください。
- 指定した複数のライブラリの中に同一名称がある場合、結合したいオブジェクトファイルのあるライブラリを前に指定してください。

(3) 間接リンクアドレス参照サブルーチン

下記モジュールを利用する場合は、libfirad.libをリンクします。

<名前>

irglbad

<形式>

```
int *irglbad (no)
```

```
int no;
```

<機能説明>

irglbadは、noが間接リンクグローバル番号（1～最大グローバル番号）のとき、対応するグローバルアドレスを返します。

<リターンコード>

noが登録済みの間接リンクグローバル番号のとき、対応するグローバルアドレスを返します。

noが未登録の間接リンクグローバル番号のとき、0を返します。

<名前>

irsubad

<形式>

```
int *irsubad (no)
```

```
int no;
```

<機能説明>

irsubadは、noがIRSUB番号（1～最大IRSUB番号）のとき、対応するIRSUBアドレスを返します。

<リターンコード>

noが登録済みのIRSUB番号のとき、対応するIRSUBアドレスを返します。

noが未登録のIRSUB番号のとき、0を返します。

(4) メッセージ出力ルーチン

下記モジュールを利用する場合は、libcpms.libをリンクします。

<名前>

rs_printf

<形式>

```
int rs_printf (buf, fmt, p1, p2, ...,p10)
```

```
char *buff;
```

```
char *fmt;
```

```
long p1, p2, ...,p10;
```

<機能説明>

rs_printfは、データを書式に従ってメッセージに変換し、OS内のメッセージ用バッファエリアにメッセージを書き込みます。

メッセージは、svdebugコマンドのpsサブコマンドで読み出すことができます。

変換後のメッセージの文字列サイズ（バイト数）が1～1024の範囲となるように指定してください。変換後のメッセージの文字列サイズ（バイト数）が1～1024の範囲以外の場合は、パラメータ異常となり書き込みは行われません。

<パラメータ>

buff : 書き込むメッセージを一時的に格納するメモリ領域の先頭アドレスを指定します。

fmt : 書式を示す文字列を格納したメモリ領域の先頭アドレスを指定します。

p1～p10 : データを指定します。

<リターンコード>

正常終了した場合は、変換後のメッセージの文字列サイズ（バイト数）が返されます。

それ以外の場合は、以下のリターンコードが返されます。

0 : OS内のメッセージ用バッファエリアが満杯で書き込みできません。

-1 : パラメータ異常で書き込みできませんでした。

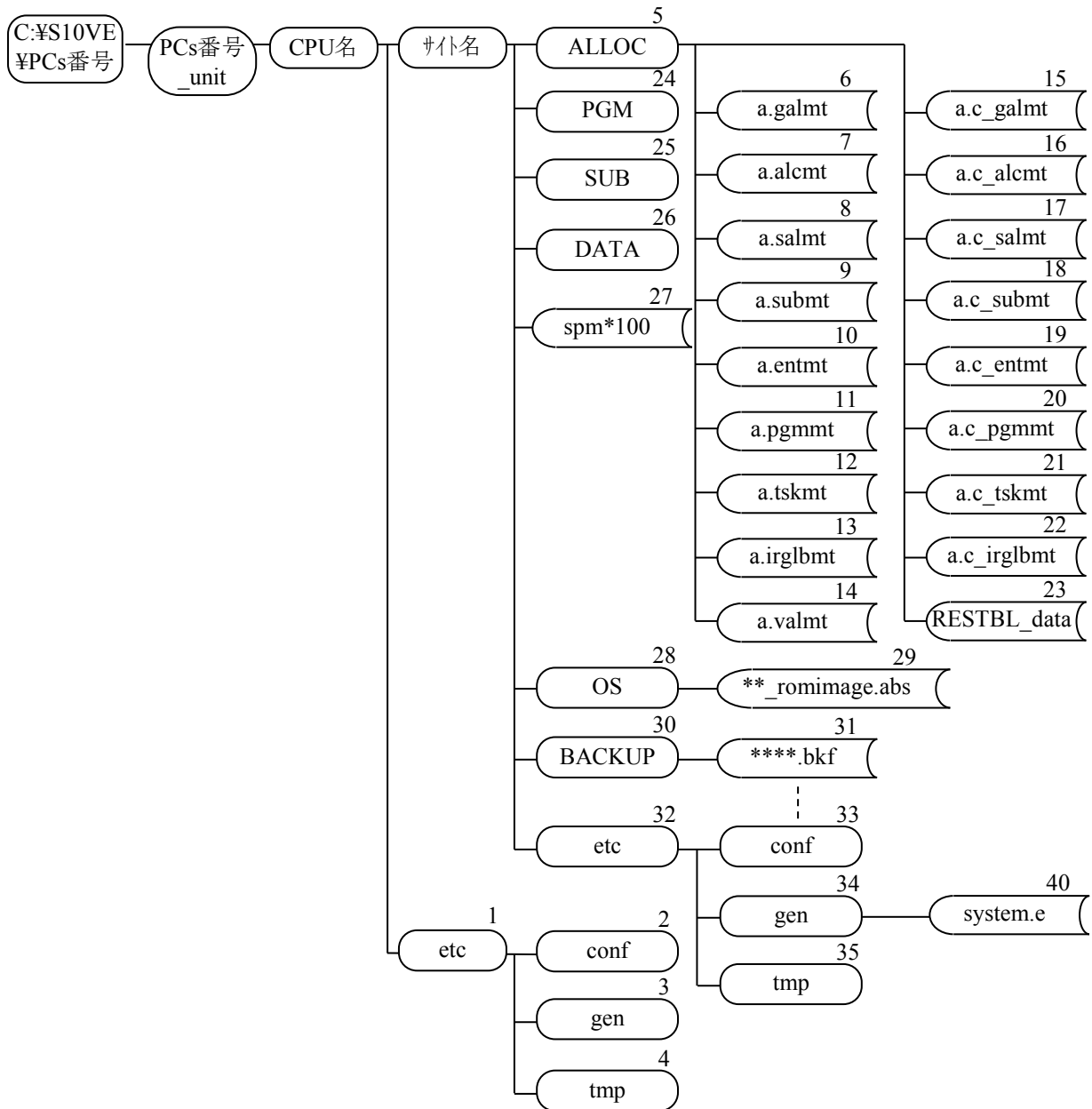
-2 : メッセージの書き込み異常で書き込みできませんでした。

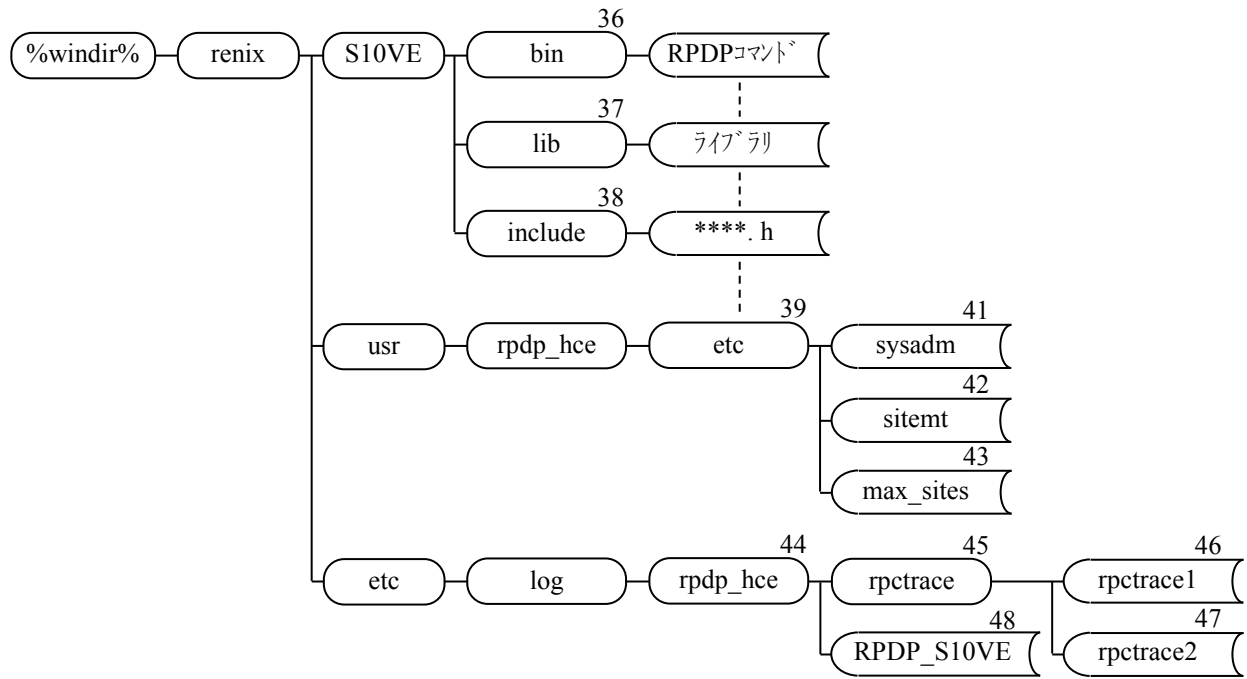
<注意事項>

デバッグのときだけ使用してください。

付録C サイト管理ファイル

以下にサイトを管理するファイルのディレクトリ構成とファイルの説明を示します。





No.	ファイル名/ ディレクトリ名	名称	説明	備考
1	etc	CPUシステムジェネレーションファイル格納ディレクトリ	CPU (CP、HP共通) のシステムジェネレーションファイルを格納するディレクトリです。	
2	conf	CPU定義情報ファイル格納ディレクトリ	CPU (CP、HP共通) のシステムジェネレーション時にユーザが定義するファイルを含むディレクトリです。	***.uファイルがユーザ定義対象ファイルです。 ***.sファイルがシステム用定義ファイルです。 svgenコマンドがテンプレートファイルを作成します。
3	gen	CPU実参照定義情報出力ファイル格納ディレクトリ	svconfコマンドが出力するCPU (CP、HP共通) の構成定義情報ファイルを格納するディレクトリです。	
4	tmp	CPU定義情報出力ファイル格納ディレクトリ	conf下の情報を基に作成した定義情報出力ファイルの格納ディレクトリです。	
5	ALLOC	アロケータ管理テーブル格納ディレクトリ	アロケータ管理テーブルを格納するディレクトリです。	
6	a.galmt	garea管理ファイル	論理空間内の領域の名称、大きさなどを管理します。	
7	a.alcmt	area管理ファイル	タスクのtext、dataサブプログラムのtext、dataグローバル (CM含む) 内に確保したAREA (分割領域) を管理します。	system.uファイル内MAXAREAで定義した数のエントリを含みます。
8	a.salmt	sarea管理ファイル	グローバル (CM含む) 内AREA (分割領域) に確保したSAREA (細分割領域) を管理します。	system.uファイル内MAXSAREAで定義した数を加えたエントリを含みます。
9	a.submt	サブプログラム管理ファイル	サブプログラム (IRSUB、組み込みサブルーチン) を管理します。	system.uファイル内ENTMT_MAXENTで定義した数に組み込みサブルーチンの最大数を加えた分のエントリを含みます。
10	a.entmt	IRSUB管理ファイル	間接リンクサブプログラム (IRSUB) を管理します。	system.uファイル内ENTMT_MAXENTで定義した数分のエントリを含みます。

付録C サイト管理ファイル

No.	ファイル名/ ディレクトリ名	名称	説明	備考
11	a.pgmmt	プログラム管理ファイル	タスクとして登録されるプログラムを管理します。	system.uファイル内 PGM_MAXNUMで定義した数分のエントリを含みます。
12	a.taskmt	タスク管理ファイル	タスクを管理します。	
13	a.irglbmt	間接リンクグローバル管理ファイル	間接リンクグローバルの登録を管理します。	system.uファイル内 IRG_MAXENTで定義した数分のエントリを含みます。
14	a.valmt	バリュ管理ファイル	バリュの登録を管理します。	system.uファイル内 MVAL_MAXNUMで定義した数を加えたエントリを含みます。
15	a.c_galmt	S10VE側garea管理ファイル	S10VE側の論理空間内の領域の名称、大きさなどを管理します。	svrplコマンドまたはsvdebugコマンドのldサブコマンドでS10VEにダウンロードされます。
16	a.c_alcmt	S10VE側area管理ファイル	S10VE側のタスクのtext、dataサブプログラムのtext、dataグローバル（CM含む）内に確保したAREA（分割領域）を管理します。	
17	a.c_salmt	S10VE側sarea管理ファイル	S10VE側のグローバル（CM含む）内AREA（分割領域）に確保したSAREA（細分割領域）を管理します。	
18	a.c_submt	S10VE側サブプログラム管理ファイル	S10VE側のサブプログラム（IRSUB、組み込みサブルーチン）を管理します。	
19	a.c_entmt	S10VE側IRSUB管理ファイル	S10VE側の間接リンクサブプログラム（IRSUB）を管理します。	
20	a.c_pgmmt	S10VE側プログラム管理ファイル	S10VE側のタスクとして登録されるプログラムを管理します。	
21	a.c_taskmt	S10VE側タスク管理ファイル	S10VE側のタスクを管理します。	
22	a.c_irglbmt	S10VE側間接リンクグローバル管理ファイル	S10VE側の間接リンクグローバルの登録を管理します。	

No.	ファイル名/ ディレクトリ名	名称	説明	備考
23	RESTBL_data	リソース管理テーブルデータファイル	サイトのリソースを管理するテーブルデータファイルです。	
24	PGM	プログラム格納ディレクトリ	プログラムのロードモジュールを格納するディレクトリです。	svloadで-dオプションを指定した場合に格納されます。
25	SUB	サブプログラム格納ディレクトリ	サブプログラムのロードモジュールを格納するディレクトリです。	
26	DATA	グローバル初期値データ格納ディレクトリ	グローバルエリアの初期値データを格納するディレクトリです。	
27	spm*100 (CP側 : spmd100) (NP側 : spmc100)	SPMファイル	S10VEの主メモリ (SPM領域) 内のOS、ドライバ用データファイルです。 CP、HPでファイル名称が異なります。	svrplコマンドでS10VE主メモリにローディングされます。
28	OS	OS格納ディレクトリ	S10VEのOS基本部を格納するディレクトリです。	
29	**_romimage.abs	OSファイル	S10VEのOSファイルです。 I/O、ネットワークドライバ機能も含まれます。	svrplコマンドでS10VE主メモリにローディングされます。
30	BACKUP	バックアップファイル格納ディレクトリ	バックアップファイルを格納するディレクトリです。	svdfaで作成したバックアップファイルが格納されます。
31	****.bkf	バックアップファイル	分割領域単位のS10VEメモリ初期値ファイルです。	svrplコマンドでS10VE主メモリにローディングされます。
32	etc	システムジェネレーションファイル格納ディレクトリ	システムジェネレーションファイルを格納するディレクトリです。	
33	conf	サイト定義情報ファイル格納ディレクトリ	システムジェネレーション時にユーザが定義するファイルを含むディレクトリです。	***.uファイルがユーザ定義対象ファイルです。 ***.sファイルがシステム用定義ファイルです。 svgenコマンドがテンプレートファイルを作成します。

No.	ファイル名/ ディレクトリ名	名称	説明	備考
34	gen	構成定義情報ファイル格納ディレクトリ	svconfコマンドが出力する構成定義情報ファイルを格納するディレクトリです。	
35	tmp	構成定義情報出力ファイル格納ディレクトリ	conf下の情報を基に作成した定義情報出力ファイルの格納ディレクトリです。	
36	bin	RPDPコマンド格納ディレクトリ	RPDPコマンドが格納されているディレクトリです。	
37	lib	S10VE用ライブラリ格納ディレクトリ	S10VE用ライブラリ群が格納されているディレクトリです。	
38	include	S10VE用インクルードファイル格納ディレクトリ	S10VE用インクルードファイルが格納されているディレクトリです。	
39	etc	システム管理ファイル格納ディレクトリ	システム管理ファイルを格納するディレクトリです。	
40	system.e	サイト定数管理ファイル	サイト定数情報格納ファイルです。	
41	sysadm	システム管理ファイル	サイト情報格納ファイルです。	
42	sitemt	システム管理ファイル	サイト情報格納ファイルです。	
43	max_sites	最大サイト数管理ファイル	最大サイト数情報格納ファイルです。	
44	rpdp_hce	ログファイル格納ディレクトリ	ログファイルが格納されているディレクトリです。	
45	rpctrace	RPCライブラリ用ログファイル格納ディレクトリ	RPCライブラリ用のログファイルが格納されているディレクトリです。	
46	rpctrace1	RPCライブラリ用ログファイル	RPCライブラリ用のログファイルです。	ログの格納数が最大になった場合、ログの収集先をrpctrace2に切り替えます。
47	rpctrace2	RPCライブラリ用ログファイル	RPCライブラリ用のログファイルです。	ログの格納数が最大になった場合、ログの収集先をrpctrace1に切り替えます。
48	RPDP_S10VE	RPDP用ログファイル	RPDP用のログファイルです。	

付録D エラーメッセージ

各コマンドで出力するエラーメッセージの意味とユーザの対応を示します。

エラーメッセージ内の%s、%d、%xにはエラーに関連する数値、文字列が表示されます。

No.	項目	コマンド	対応ページ
(1)	アロケータ、ローダ、ビルダコマンドのエラーメッセージ	svdfa	A-16
		svdla	
		svdfs	
		svdls	
		svdfv	
		svdlv	
		svload	
		svdload	
		svcomp	
		svctask	
		svdtask	
		svbuild	
		svdbuild	
svirglb			
(2)	svdebugエラーメッセージ一覧	svdebug	A-24
(3)	svrplコマンドエラーメッセージ一覧	svrpl	A-32
(4)	svcpuctlコマンドエラーメッセージ一覧	svcpuctl	A-34
(5)	svelogコマンドエラーメッセージ一覧	svelog	A-35
(6)	svdhpコマンドエラーメッセージ一覧	svdhp	A-36
(7)	svcpunowコマンドエラーメッセージ一覧	svcpunow	A-39
(8)	svtimexコマンドエラーメッセージ一覧	svtimex	A-42
(9)	svdatagenコマンドエラーメッセージ一覧	svdatagen	A-45

(1) アロケータ、ローダ、ビルダコマンドのエラーメッセージ

● エラーメッセージ

アロケータ、ローダ、ビルダ、およびマップ表示コマンドが出力するエラーメッセージを
表A-2に示します。各コマンドは異常検出後、これらのメッセージを出力し終了します。

表A-2 エラーメッセージ

(1/8)

エラーコード	エラーメッセージ	意味	ユーザの対応
システム管理者による復旧が必要な内部エラー			
1001-1	Abnormal allocator management table (%s)	アロケータ管理テーブルが異常です。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
リソース不足に関連するエラー			
1002-4	Not enough physical memory allocated	物理メモリの容量が不足しています。	不要なリソースを削除し、再試行してください。
6	Not enough area allocated (%s)	GLB領域の容量が不足しています。	
8	No task number available	タスク番号の空きがありません。	
10	No free table to make new entry (%s)	アロケータ管理テーブルの空きがありません。	
13	Cannot get RSSITE (%s)	環境変数のRSSITEが取り込めません。	環境変数を設定後、再試行してください。
15	Please set environment variable (%s)	環境変数が設定されていません。	
システムリソース不足に関連するエラー			
1003-1	Memory allocation error (malloc, %s)	mallocでメモリが確保できません。	メモリの空きを確認し再試行してください。
3	Cannot create temporaries: %s	テンポラリファイルが作成できません。	ディスクの空きを確認し再試行してください。
リトライにより復旧可能なエラー			
1004-1	Allocator management table is busy	アロケータ管理テーブルが他コマンドによって使用中です。	再試行してください。
原因を特定できるシステムコールエラー			
1005-1	Cannot open %s (%s)	ファイルを開けません。	ファイルの有無アクセス権を見直して再試行してください。
2	Cannot read %s (%s)	ファイルを読み出せません。	
3	Cannot write %s (%s)	ファイルに書き込めません。	
4	Cannot stat %s (%s)	ファイルのステータス情報が取り込めません。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
5	Cannot lseek %s (%s)	ファイルポインタをシークできません。	

(2/8)

エラーコード	エラーメッセージ	意味	ユーザの対応
原因を特定できないシステムコールエラー			
1006-1	syscall error (%s, errno=%d) (*2)	システムコールのエラーが発生しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
2	コマンド名 : WIN32API error (API名称, EC=エラーコード) (*3)	API名称の関数でエラーコードのエラーが発生しました。	
パラメータ不足			
1007-3	Not enough parameter	引数が足りません。	引数を見直してください。
入力パラメータ範囲異常関連のエラー			
2001-2	Align number is out of range	アライン数値が範囲外です。	入力可能データを確認後、再試行してください。
3	Task number is out of range (1 to %d)	タスク番号が範囲外です。	
5	Priority level is out of range (%d to %d)	ユーザタスク用実行レベルが範囲外です。	
6	Priority level for system is out of range (%d to %d)	システムタスク用実行レベルが範囲外です。	
9	Bad align type	アライン種別が間違っています。	
10	Illegal point number (%d)	組み込みサブルーチン用ポイント番号が間違っています。	
11	Entry number is out of range (1 to %d)	組み込みサブルーチン用エントリ番号が範囲外です。	
12	Specified index number is out of range (1 to %d)	指定の番号が範囲外です。	
13	Invalid name (%s)	名称が間違っています。	
18	Illegal point name (%s)	組み込みサブルーチン用ポイント名称が間違っています。	
19	Numeric value is out of range	指定の数値が範囲外です。	
23	Limit size for stack is out of range (0 to 2097152)	指定のスタックサイズが範囲外です。	
24	User task number is out of range (1 to 224)	指定のユーザタスク番号が範囲外です。	
25	Specified system index number is out of range (1 to %d)	指定のシステム用番号が範囲外です。	
26	Specified number with -r is out of range	-rオプションで指定した番号が範囲外です。	
27	Loading data is empty	ローディングデータがありません。	
28	Number of user task is over (タスク数)	ユーザタスク数がシステム定数を超えました。	不要なリソースを削除し、再試行してください。
33	ULSUB stack size (%d) is out of range (0 to 512)	組み込みサブプログラムのスタックサイズが512バイトを超えています。	スタックサイズを見直してください。
35	Program/IRSUB stack size (%d) is out of range (0 to 8388608)	タスクのスタックサイズが8MBを超えています。	

エラーコード	エラーメッセージ	意味	ユーザの対応
入力パラメータ未定義関連のエラー			
2002-1	Specified name is undefined (%s)	指定の名称は未定義です。	入力可能データを 確認後、再試行し てください。
4	Specified point number in the entry number is empty	指定の組み込みサブルーチン用ポイント番号は未登録です。	
5	Specified IRSUB is not built (%s)	指定のIRSUBはビルドされていません。	
7	Specified IRSUB is already built (%s)	指定のIRSUBはビルド済みです。	
8	Specified number is undefined	指定の番号は未定義です。	
10	%s is undefined	オブジェクト内の名称が未定義です。	
11	Loading data is empty	ローディングするデータがありません。	オブジェクトファイル指定を見直してください。
13	Area (%s) kind is wrong	エリア種別が誤っています。	指定エリアを見直してください。
14	Can not load data in GLBW, CMW, and DCMW (%s)	初期値なしGLB、CMに初期値をロードすることはできません。	
15	Can not load CM, DCM data from NP or HP site	HPサイトから、CMにロードすることはできません。	
入力パラメータ二重定義関連のエラー			
2003-1	Specified name is already defined (%s)	指定名称は登録済みです。	入力可能データを 確認後、再試行し てください。
5	Task number is already defined	指定のタスク番号は登録済みです。	
7	Point number is already defined	指定の組み込みサブルーチン用ポイント番号は登録済みです。	
8	Specified IRSUB number is already defined	指定したIRSUB番号は使用済みです。	IRSUB番号を変えて再試行してください。
13	Unmatched reserved index number	指定の番号は不一致です。	入力可能データを 確認後、再試行し てください。
15	PN=%s is already defined	プログラム管理番号が重なっています。	
19	Specified number is already defined	指定の番号は登録済みです。	
21	Can not specify -s or -a with SAREA (%s)	細分割領域に-sまたは-aオプションは指定できません。	
22	Specified pgmname is already defined as TASK (%s)	指定のプログラム名称はタスクとして登録済みです。	
23	PN=%d is already loaded for single task	プログラム管理番号は定義済みです。	プログラム管理番号を変えて再試行してください。

(4/8)

エラーコード	エラーメッセージ	意味	ユーザの対応
入力パラメータ属性不一致関連のエラー			
2004-1	Unmatched owner type	所有者タイプが不一致です。	入力可能データを 確認後、再試行し てください。
2	Illegal user type (%s)	使用者タイプが不一致です。	
3	Specified area is not global (%s)	エリアの種別がGLB以外です。	
5	Illegal program type	プログラム種別が不一致です。	
8	Unmatched entry number	組み込みサブルーチン用エントリ番号 が不一致です。	
10	Area type is not GLBI	エリア種別が初期値ありGLB以外で す。	
11	Multi task attribute error	マルチタスク属性が不一致です。	
16	Unmatched entry type	組み込みサブルーチン用エントリセッ ト番号が不一致です。	
17	Specified name is defined as GLB, CM or DCM (%s)	指定の名称はGLB、CMとして定義さ れています。	
19	Specified name is defined as VAL (%s)	指定の名称はVALとして定義されて います。	
21	4096 aline error (%s)	指定アドレスが4096バイト境界ではあ りません。	入力可能データを 確認後、再試行し てください。
22	Physical address error (%s)	不当な論理アドレスを指定していま す。	
23	Area (%s) kind is wrong	エリアの属性が誤っています。	
24	Loading data is too large (sname=%s)	データサイズがエリアのサイズを超え ています。	
25	Inconsistent object was mixed	ライブラリ指定で-lsh4nbmzzと -lsh4nbmdn両方を指定しています。	
操作誤り関連のエラー			
2005-1	Cannot delete area which is already used	タスクまたはサブプログラムが登録済 みのためエリアを削除できません。	svdloadしてから削 除してください。
2	Cannot delete program which is registered as task	タスクとして登録済みのため削除でき ません。	svdtaskしてから削 除してください。
3	Cannot delete built subprogram (%s)	ビルド済みのため削除できません。	svdbuildしてから削 除してください。
6	Cannot delete defined %s (%s)	GLBまたはVALとして登録済みのた め削除できません。	svdlsまたはsvdlvし てから削除してく ださい。
7	Specified name (%s) is referenced by PROG or SUB	指定のリソースはプログラムまたはサ ブプログラムから参照されています。	参照しているプロ グラムまたはサブ プログラムを削除 してから再試行し てください。

エラーコード	エラーメッセージ	意味	ユーザの対応
svloadコマンドでのパラメータ指定方法誤り関連のエラー			
2006-1	Invalid subargument: -W%s	使用できないサブ引数を指定しました。	入力可能データを 確認後、再試行し てください。
5	Too few arguments	引数の個数が不足しています。	
8	Missing operand (%s)	オペランドが不足しています。	
9	Bad option (%s)	使用できないオプションを指定しまし た。	
10	Invalid name (%s)	名称が間違っています。	
svloadコマンドの内部コマンドのエラー			
2008-1	Error in %s ; Status終了コード	内部コマンド (%s) でエラーが発生 しました。 内部コマンドのエラー要因は同時に 出力される内部コマンドのエラーメッ セージを参照してください。内部コマ ンド (リンク) のエラーメッセージ は、コンパイラパッケージのマニユア ル (PDFファイル) を参照してくださ い (終了コードは意味を持ちませ ん)。	指定したライブラ リが正しいか、オ ブジェクトファイ ルまたはそのソー スを見直し後、再 試行してくださ い。
2008-2	Fatal error in %s ; Status終了コード	内部コマンド (%s) で致命的なエ ラーが発生しました。 内部コマンドのエラー要因は同時に 出力される内部コマンドのエラーメッ セージを参照してください。内部コマ ンド (リンク) のエラーメッセージ は、コンパイラパッケージのマニユア ル (PDFファイル) を参照してくださ い (終了コードは意味を持ちませ ん)。	調査用のデータを 収集し、開発系マ シンを再立ち上げ してください。 (*4)
アロケータ管理テーブルの異常			
200-1	RMphase, (0x%02x)(*)	処理フェーズが異常です。	開発系マシンを再 立ち上げしてくだ さい。
入力データ誤りの関連のエラー			
—	Argument list too long	引数の数が多すぎます。	入力可能データを 確認後、再試行し てください。
—	Argument data too large	引数のデータ値が大きすぎます。	
—	File open error	指定ファイルが開けません。	
—	Illegal format in ファイル名 line 行	指定ファイルのline行目のフォーマッ トに誤りがあります。	
—	Illegal format of name	指定名称フォーマットに誤りがありま す。	
—	Illegal format of numeric value	数値データの指定に誤りがあります。	
—	Illegal format of task name	タスク名称の指定フォーマットに誤り があります。	

(6/8)

エラーコード	エラーメッセージ	意味	ユーザの対応
入力データ誤りの関連のエラー (続き)			
—	Illegal operand	オペランドの指定に誤りがあります。	入力可能データを 確認後、再試行し てください。
—	Program text is empty	プログラムテキストサイズが0です。	
—	%s is different from subprogram top name	サブプログラム名称が誤っています。	
—	%s is refered from system type	システムからユーザを参照していま す。	
—	%s is refered from user type	ユーザからシステムを参照していま す。	
—	Illegal option	許されないオプションを指定しまし た。	
—	Illegal option combination	オプションの組み合わせに誤りがあり ます。	
—	Missing option parameter	オプションパラメータの指定に誤りが あります。	
—	Numeric value is out of range	指定された数値が許容範囲外です。	
—	Parameter error	パラメータの指定誤りです。	
—	Specified number is undefined (指定name)	svmapコマンドでnameおよび-nオプ ションを指定したとき、指定name (エントリ番号) は未定義でした。	
—	Specified number is illegal (指定name)	svmapコマンドでnameおよび-nオプ ションを指定したとき、name指定 (エントリ番号) に誤りがありまし た。	
—	Specified name is undefined (指定name)	svmapコマンドでnameを指定したと き、指定nameは未定義でした。	
—	Task number error	タスク番号の指定方法に誤りがありま す。	
—	Bad file name	svadmコマンドでオペレーション結果 出力先のファイル名称が256文字以上 です。	
—	Bad site name	svadmコマンドで指定サイト名称が14 文字以上です。	
—	Parameter is too long	引数の数が多すぎます。	
—	Illegal parameter	引数のパラメータに誤りがあります。	
—	Sitename max length is 14 character (-u)	サイト名が14文字を超えています。	
—	Multi task count is out of range (2 to 128)	マルチタスク数に誤りがあります。	2~128の値を指定 してください。
—	Can not get RSSITE	環境変数を取り込めません。	RSSITE環境変数を 指定してくださ い。
—	No such site (%s)	指定サイトはありません。	指定サイトを見直 してください。

エラーコード	エラーメッセージ	意味	ユーザの対応
入力データ誤りの関連のエラー (続き)			
—	Some system constants are not defined	指定定数がシステム定数にありません。	指定定数を見直してください。
—	-w option argument is not 8 byte align	スタックサイズに8の倍数以外が指定されています。	指定サイズを見直してください。
—	Total stack size (%d) is too small	確保するスタックサイズがプログラムで使用するスタックサイズよりも小さな値です。	スタックサイズを見直してください。
—	-C option argument is not %d byte align	-Cオプションで指定された値が適切な値ではありません。	プログラムの場合は4096の倍数を、サブプログラムの場合は32の倍数を指定してください。
—	Bad realtime environment	環境変数が異常です。	環境変数を見直してください。
—	Program text is empty	プログラムのテキストサイズが0です。	指定オブジェクトファイルを見直してください。
—	%s is different from subprogram top name	サブプログラム名称が誤っています。	
—	.rodata cannot locate GLB area	const宣言されたデータは、ロードできません。	初期値データプログラムのconst宣言を見直してください。
—	%s is not defined (Sarea)	オブジェクトに未定義GLB、CMの初期値データが含まれます。	初期値データを見直してください。
—	Program has BSS area	マルチタスク、IRSUBがBSSエリアを持っています。	プログラムを確認してください。
—	Stack size (%s) = %d (%d) byte [MAX refered (%s) size %d byte] Err	累積スタックサイズが指定スタックサイズを超えました。	指定スタックサイズを見直してください。
—	Can not get site information	サイト情報が取り込めません。	サイト定義内容を確認のうえ、再試行してください。
—	CM area address or size is different form another CPU	他方のCPU (CP、HP) とアドレスまたはサイズが異なります。	CMのアドレスとサイズを合わせてください。
実行環境異常関連のエラー			
—	Please set RSSITE	環境変数RSSITEが設定されていません。	RSSITEを設定後、再試行してください。
—	Unknown RSUTYP	環境変数RSUTYPに指定外のパラメータを設定しています。	RSUTYPに 's' または 'u' を設定してください。

(8/8)

エラーコード	エラーメッセージ	意味	ユーザの対応
コマンド内部のエラー			
—	cannot perform malloc	mallocまたはreallocによる作業エリアの確保に失敗しました。	再試行してください。
—	Internal error (timeout detected)	通信タイムアウトを検出しました。	
—	Internal error (no valid data)	通信データに異常が発生しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
—	cannot open %s	svmapコマンドでアロケータ管理テーブルファイルのオープンに失敗しました。	
—	Specified site is undefined	指定サイトが存在しません。	サイト名を見直してください。

(*1) 調査用のデータは下記のファイルです。

- ・%windir%\¥renix¥etc¥log¥rpdp_hce¥RPDP_S10VE
- ・%windir%\¥renix¥etc¥log¥rpdp_hce¥1_RPDP_S10VE

(*2) エラーコード1006-1に表示するerrnoの意味

errno	意味
2	ファイルが見つかりません。
3	パスが異常です。
9	ファイルが異常です。
12	メモリが不足しています。
13	アクセス権がありません。
24	オープンしているファイルが多すぎます。
28	ディスクが不足しています。

(*3) エラーコード1006-2に表示するエラーコードの意味

エラーコード	意味
2	ファイルが見つかりません。
3	パスが異常です。
4	オープンしているファイルが多すぎます。
5	アクセス権がありません。
6	ハンドルが異常です。
8、14	メモリが不足しています。

(*4) 処理フェーズの意味

- 0x01 : 主メモリ上管理テーブル更新中
- 0x02 : 管理テーブルファイル更新中
- 0x03 : ハッシュテーブル更新中
- 0xff : システム状態異常

(2) svdebugエラーメッセージ一覧

(1/7)

No.	エラーメッセージ	意味	ユーザの対応
1	File "ファイル名" already exists	指定したファイルはすでに存在しています。	正しいファイル名を指定してください。
2	Site "サイト名" not found	指定したサイトがありません。	サイト名を見直してください。
3	Cannot open "ファイル名"	ファイルのオープンに失敗しました。	正しいファイル名を指定してください。
4	No filename given for -i/-o/-r option	ファイル名の指定がありません。	ファイル名を指定してください。
5	No sitename given for -u option	サイト名の指定がありません。	サブコマンドを指定してください。
6	Task No error	タスク番号の指定が誤っています。	タスク番号を見直してください。
7	Task name error	タスク名の指定が誤っています。	タスク名を見直してください。
8	Factor error	起動要因の指定が誤っています。	起動要因を見直してください。
9	Cannot Specify RPC-server task	RPCサーバのタスクは指定できません。	タスク番号、タスク名を見直してください。
10	Unknown sub command	解釈不能なサブコマンド名が指定されています。	サブコマンドを見直してください。
11	Name error	解釈不能な名称が指定されています。	名称を見直してください。
12	Option error	解釈不能なオプションが指定されています。	オプションを見直してください。
13	Storage error	解釈不能な記憶装置が指定されています。	storage指定を見直してください。
14	Invalid address set	アクセス不可能なアドレスを指定しています。	アドレスを見直してください。
15	Misformed patch data	変更となるデータの設定に誤りがあります。	8進、10進、16進、実数のいずれかで入力してください。
16	Unknown RSSITE	RSSITE環境変数が設定されていません。	RSSITE環境変数を設定してください。
17	Unknown RSUTYP	ユーザモードでのシステムリソースへのアクセスはできません。	RSUTYP環境変数に 's' または 'u' を設定してください。

(2/7)

No.	エラーメッセージ	意味	ユーザの対応
18	Break point already used by another process	別のデバッグプロセスでブレークポイントを使用中です。	他のユーザの終了を待って、使用してください。
19	Input error	入力書式に誤りがあります。	入力書式を見直してください。
20	Type or length error	md、sdのオプション指定に誤りがあります。	オプションを見直してください。
21	RPDP library error (ライブラリ名:エラーコード)	RPDPのライブラリでエラーが発生しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1) (*3)
22	Allocator management table busy	アロケータ管理テーブルがbusy状態です。	コマンドを再実行してください。
23	Unmatch resource status	開発系マシンとS10VE間でリソースの状態が不一致です。	名称指定を見直してください。
24	Id NO error	tmサブコマンドのidの指定が誤っています。	idを見直してください。
25	Time error	tmサブコマンドのtの指定が誤っています。	tを見直してください。
26	Cycle time error	tmサブコマンドのcyctの指定が誤っています。	cyctを見直してください。
27	Initial/check data error	si、spサブコマンドの初期化／チェックパターンの指定が誤っています。	初期化／チェックパターンの指定を見直してください。
28	Addr error	br、rb、asサブコマンドのアドレスの指定が誤っています。	アドレスを見直してください。
29	Break point is used	ブレークポイントを使用中です。	ブレークポイントを解除してからデバッグを終了してください。
30	Bit data error	bs、bgサブコマンドのビットの指定が誤っています。	ビットの指定を見直してください。
31	Sht sub command is already executed by another process	別のデバッグプロセスでshtサブコマンドを実行中です。	他のユーザの終了を待って、使用してください。
32	Task no error (NO.2100-01)	タスク番号の指定が誤っています。	タスク番号を見直してください。
33	Task name error (NO.2100-02)	タスク名の指定が誤っています。	タスク名を見直してください。

No.	エラーメッセージ	意味	ユーザの対応
34	Factor error (NO.2100-03)	起動要因の指定が誤っています。	起動要因を見直してください。
35	Cannot Specify RPC-server task (NO.2100-04)	RPCサーバのタスクは指定できません。	タスク番号、タスク名を見直してください。
36	Specified task is dormant (NO.2100-05)	指定されたタスクは、DORMANT状態です。	タスクの状態を確認し、再試行してください。
37	Specified task is not dormant (NO.2100-06)	指定されたタスクは、DORMANT状態ではありません。	
38	Specified task is already suspend (NO.2100-07)	指定されたタスクは、すでに実行抑止状態です。	
39	Specified task is not suspend (NO.2100-08)	指定されたタスクは、すでに実行抑止状態ではありません。	
40	Specified task is not registered (NO.2100-09)	指定されたタスクは、未登録です。	タスク番号、タスク名を見直してください。
41	Backup file access error (NO.2100-10)	バックアップファイルのアクセスに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
42	Unmatched RSUTYP (NO.2100-11)	ユーザモードでのシステムリソースへのアクセスはできません。	システムモードでアクセスしてください。
43	Unmatch resource status (NO.2100-12)	開発系マシン、S10VE間でリソースの状態が不一致です。	名称指定を見直してください。
44	Specified task is undefined (NO.2100-13)	指定したタスク名が未定義です。	タスクの指定を見直してください。
45	CPMS not running (NO.2100-14)	CPMSがRUNNING状態ではありません。	CPMSをRUNNING状態にしてください。
46	Task is not dormant (tn=%d) (NO.2100-15)	タスクがDORMANT状態ではありません。	タスクをDORMANT状態にして再試行してください。
47	Invalid address error (NO.2100-16)	アクセス不可能なアドレスをアクセスしようとしてしました。	アドレスを見直してください。
48	Invalid address set (NO.2100-17)	アクセス不可能なアドレスを指定しています。	アドレスを見直してください。
49	Cannot open ”ファイル名” (NO.2100-18)	ファイルのオープンに失敗しました。	正しいファイル名を指定してください。

(4/7)

No.	エラーメッセージ	意味	ユーザの対応
50	Processor connection table is full (NO.2100-19)	プロセス間連絡テーブルが満杯です。	他ユーザの使用終了を待ち、再試行してください。
51	Specified task is not idle (NO.2100-20)	指定タスクがIDLE状態ではありません。	タスクの状態を確認し再試行してください。
52	Timer event is not registered (NO.2100-21)	タイマイイベントが登録されていません。	指定タスクを見直してください。
53	Time error (NO.2100-22)	tmサブコマンドのtの指定が誤っています。	tを見直してください。
54	Cycle time error (NO.2100-23)	tmサブコマンドのcyctの指定が誤っています。	cyctを見直してください。
55	Cannot get system constant (NO.2100-24)	システム定数の取り出しに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
56	Ent error (NO.2100-25)	ldサブコマンドのentの指定が誤っています。	entの指定を見直してください。
57	Irsub No error (NO.2100-26)	間接リンクサブルーチン番号が誤っています。	間接リンクサブルーチン番号を見直してください。
58	Irglobal No error (NO.2100-27)	間接リンクグローバル番号の指定が誤っています。	間接リンクグローバル番号を見直してください。
59	Task suspend failed (NO.2100-28)	taサブコマンドでタスクの実行抑止に失敗しました。	タスクの状態を確認し、再試行してください。
60	Point error (NO.2100-29)	ldサブコマンドのポイントの指定が誤っています。	pointを見直してください。
61	Cannot get register information (NO.2100-30)	ta、rrサブコマンドでレジスタの取り出しに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
62	Specified name (名称) is undefined (NO.2100-31)	指定された名称が未定義です。	名称指定を見直してください。
63	Cannot register timer event in TRB (NO.2100-32)	タイマイイベントの登録に失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
64	File "ファイル名" already exists (NO.2100-33)	ファイルは、すでに存在しています。	ファイル名の指定を見直してください。
65	File "ファイル名" create error (NO.2100-34)	ファイルの作成に失敗しました。	
66	Cannot save "ファイル名" (NO.2100-35)	ファイルの保存に失敗しました。	

No.	エラーメッセージ	意味	ユーザの対応
67	File "ファイル名" read error (NO.2100-36)	ファイルの読み出しに失敗しました。	ファイル名の指定を見直してください。
68	File "ファイル名" format error (NO.2100-37)	ld、cmサブコマンドで指定したファイルのフォーマットが誤っています。	
69	Pname "ファイル名" not found (NO.2100-38)	プログラム名が見つかりません。	プログラム名を見直してください。
70	Must specify address in text space (NO.2100-39)	text空間内のアドレスを指定してください。	アドレス指定を見直してください。
71	Specified address is already set (NO.2100-40)	指定アドレスは、すでに設定済みです。	
72	Must specify break point address (NO.2100-41)	ブレークポイントのアドレスを指定してください。	
73	Cannot get TCB (NO.2100-42)	taサブコマンドでTCBの取り出しに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
74	Cannot set break point beyond the max (NO.2100-43)	すでにブレークポイントの最大設定可能数分 (5つ) を指定しています。	ブレークポイントを解除してから再試行してください。
75	Cannot use ld sub command after RSSRCV set (NO.2100-44)	RSSRCVの設定後にldサブコマンドは使用できません。	svrplコマンドで一括ダウンロードしてください。
76	Inconsistency detected %s (NO.2100-46)	アロケータ管理テーブルに不整合があります。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
77	CM is not defined (NO.2100-48)	CM領域は定義されていません。	オプション指定を見直してください。
78	DCM is not defined (NO.2100-49)	DCM領域は定義されていません。	
79	Specified area is not defined for glb (NO.2100-50)	指定した分割領域はグローバルのエリアではありません。	指定分割領域名を見直してください。
80	Cannot specify another CM space (NO.2100-51)	他のCM空間を指定することはできません。	指定パラメータを見直してください。
81	Specified pgm number is out of range (1 to 255) (NO.2100-52)	指定プログラム番号は、範囲外です。	
82	Cannot load CM/DCM when PU is running (NO.2100-53)	S10VEがRUN状態のとき、CMへローディングできません。	S10VEの状態を確認して、再試行してください。

(6/7)

No.	エラーメッセージ	意味	ユーザの対応
83	ADT channel is already set (NO.2100-56)	ADTはすでに設定されています。	ADTを解除してから実行してください。
84	Illegal break point address (laddr=論理アドレス) (NO.2100-58)	ブレークポイントに設定されている論理アドレスは、プログラムとして登録されているアドレスではありません。	CPUをリスタートし、ブレークポイントの設定を解除してください。
85	Specified raddr is not break point address (raddr=相対アドレス) (NO.2100-59)	指定したraddrはブレークポイントが設定されているアドレスではありません。	指定アドレスを見直してください。
86	Break task is not found (NO.2100-60)	ブレーク中のタスクが見つかりませんでした。	ブレークポイントの設定を確認してください。
87	Specified name (プログラム名称) is used break point (NO.2100-61)	ldサブコマンドで指定したプログラム名称は、ブレークポイントが設定されています。	ブレークポイントを解除してから再実行してください。
88	Specified area is not initialize data area (NO.2100-62)	指定したアドレスはバックアップファイルの存在しない領域を含んでいます。	指定アドレスを見直してください。
89	Specified address is not initialize data area (NO.2100-63)	指定したエリアはバックアップファイルが存在しません。	指定エリアを見直してください。
90	Specified name (%s) is not GLB/CM/DCM area (NO.2100-66)	指定した名称は、GLB、CMエリアではありません。	GLB、CMエリアを指定してください。
91	Cannot access CM/DCM backupfile from NP or HP site (NO.2100-67)	CMのバックアップファイルはHPサイトからはアクセスできません。	CMのバックアップファイルはCPサイトからアクセスしてください。
92	Specified sub command can use in HP site only (NO.2100-73)	指定したサブコマンドはHPサイト以外では使用できません。	HPサイトを指定してください。
93	Communication error (catch signal) (NO.2101-01)	シグナルを受信しました。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再実行してください。
94	Communication error (connection timeout) (NO.2101-02)	タイムアウトが発生しました。	
95	Communication error (connection refused) (NO.2101-03)	RPCサーバが不在です。	
96	Communication error (connection cut) (NO.2101-04)	RPCサーバが切断されています。	
97	Communication error (connection reset) (NO.2101-05)	コネクションがリセットされています。	

No.	エラーメッセージ	意味	ユーザの対応
98	Communication error (server closed) (NO.2101-06)	RPCサーバがクローズしています。	ネットワークの接続状態、SIOVE、開発系マシンのIPアドレスを確認して再試行してください。
99	Communication error (port busy) (NO.2101-07)	回線ポートがbusy状態です。	他ユーザの使用終了を待ち再実行してください。
100	Communication error (bad socket specified) (NO.2101-08)	指定ソケットが異常です。	ネットワークの接続状態、SIOVE、開発系マシンのIPアドレスを確認して再試行してください。
101	Communication error (socket creat error) (NO.2101-09)	ソケットの生成に失敗しました。	
102	Communication error (no buffer) (NO.2101-10)	メモリの確保に失敗しました。	
103	Communication error (network not reached) (NO.2101-11)	ネットワークが未接続状態です。	
104	Communication error (network down) (NO.2101-12)	ネットワーク接続インタフェースがダウンしています。	
105	Communication error (port No error) (NO.2101-13)	ポート番号の取り込みに失敗しました。	
106	Communication error (IP address error) (NO.2101-14)	IPアドレスの取り込みに失敗しました。	
107	Communication error (memory attach failed) (NO.2101-15)	共有メモリのアタッチに失敗しました。	
108	Communication error (trace file cannot open) (NO.2101-16)	トレースファイルのオープンに失敗しました。	
109	Communication error (trace file cannot copy) (NO.2101-17)	トレースファイルのコピーに失敗しました。	
110	Communication error (fatal error) (NO.2101-18)	致命的なエラーが発生しました。	
111	Communication error (ライブラリ名:エラーコード) (NO.2101-19) (*2)	RPL、RRBライブラリでエラーが発生しました。	
112	Communication error (inter PU communication time out) (NO.2101-20)	内部タイムアウトエラーが発生しました。	
113	Communication error (rc=%d) (NO.2101-21)	RPCライブラリエラーが発生しました。	CPMSとRPDPのバージョンを確認してください。 (*4)

(*1) 調査用のデータは下記のファイルです。

- ・ %windir%\¥renix¥etc¥log¥rpdp_hce¥RPDP_S10VE
- ・ %windir%\¥renix¥etc¥log¥rpdp_hce¥1_RPDP_S10VE

(*2) Communication error発生時は、下記を参照してください。

<Communication errorのエラーコードの意味>

0x11 : ソケット異常	0x04 : フレーム作成用メモリ確保失敗
0x12 : IPアドレス異常	0x05 : データ送信失敗
0x14 : 格納エリアアドレス異常 (0指定、dta)	0x06 : レスポンス受信待ち異常
0x15 : 格納エリアアドレス異常 (0指定、wka)	0x07 : レスポンス未受信でリトライオーバー
0x16 : サイズ異常 (0以下または16KB以上)	0x08 : データ受信失敗
0x17 : サイズ異常 (非ロングワードサイズ)	0x18 : 格納エリアアドレス異常 (0指定、dmaia)
0x03 : 相手アダプタ種別異常	0x19 : 格納エリアアドレス異常 (0指定、reta)
0x8000000X : レスポンスで異常報告 (CPU制御ヘッダ内ステータスコード)	
X : ステータスコード、4 : $\mu\Sigma 1000$ のネットワーク未設定	
0xFFFFFFFF : 環境ファイル設定誤り	

(*3) RPDP library errorのエラーコードの意味

- 1 : システムファイルのオープンに失敗しました。
- 2 : システムファイルが読み込めません。
- 3 : サイトディレクトリのパスが長すぎます。
- 4 : システムルートが取り込めません。

(*4) RPCライブラリエラーのrcの意味

- 22 : 不当なリクエストが発生しました。
- 23 : CPMSに対応する機能がありません。

(3) svrplコマンドエラーメッセージ一覧

(1/2)

No.	エラーメッセージ	意味	ユーザの対応
1	No sitename given for -u option	サイト名称が指定されていません。	サイト名称を指定してください。
2	No unitname given for -U option	ユニット名称が指定されていません。	ユニット名称を指定してください。
3	unknown RSSITE	指定サイト名称がありません。	指定サイトを確認してください。
4	Site=%s not found		
5	Unit=%s not found	指定ユニット名称がありません。	指定ユニットを確認してください。
6	%s cannot open	ファイルがオープンできません。	ファイルが正常か確認してください。
7	%s file access error	ファイルに対するアクセスができません。	
8	Internal error (%s)	内部エラーが発生しました。	再試行してください。
9	download file (%s) not found	ダウンロードするバックアップファイルが見つかりません。	環境を確認してください。
10	site (%s) lock busy	サイトは、他の処理で使われています。	再試行してください。
11	site (%s) lock error		
12	communication error (%s,RC=0x%x,エラー発生アドレス)	通信エラーが発生しました。	RCを基に原因を確認してください。 (*1) ST#指定誤り(%s:rrw_rpl_p)。または、イーサケーブル未接続です。指定を見直してください。
13	communication error (%s,RC=0x%x)		
14	site (%s) allocator management tables modify error	アロケータ管理テーブル更新でエラーが発生しました。	svmkrestblコマンドでアロケータ管理テーブルを修復してください。
15	IP ADDRESS SET ERROR (RC=0x%x)	IPアドレスを設定時にエラーが発生しました。	RCを基に原因を確認してください。 (*1) 接続PCs変更で、IPアドレス指定誤りまたはイーサケーブル未接続です。指定を見直してください。
16	%s (slot=%d) NON EXIST	スロットが存在しません。	システムジェネレーション情報を確認してください。
17	File mapping error (%s)	RPDP用リソーステーブルがセッティングできませんでした。	再試行してください。
18	site (%s) unlock error	サイトの占有解除でエラーが発生しました。	

(2/2)

No.	エラーメッセージ	意味	ユーザの対応
19	Can not specified NP or HP site(%s)	指定サイトは、HPのサイトです。	CPのサイトを指定してください。
20	Usage:svrpl [{-u site -U unit} {-s}] [-all] [-r] [{-time -notime}][{-ROMSV - NOROMSV}] [-setpcsno]	—	—
21	Can not get site information	サイトの情報を取得できません。	指定サイトの状態を確認してください。 (*2)
22	Can not load data until finish initializing module-hardware	モジュールハードウェアの初期化中のため、データをダウンロードできません。	ハードウェアの初期化完了を待って、再試行してください。(ハードウェアの初期化中は、モジュールのSTBY LEDが点滅しています。完了するとSTBY LEDが点灯状態になります。)
23	Command I/F (command code=0x%x) time out	コマンドI/Fでタイムアウトが発生しました。 (*3)	ネットワークの接続状態を確認して再試行してください。
24	Not RPDUsers	RPDUsers権限を持たないユーザです。	RPDUsers権限を持つユーザで実行してください。
25	-NOROMSV cannot be specified with - setpcsno	-NOROMSVと-setpcsnoは同時に指定できません。	オプションの指定を見直してください。
26	CPMS has not been downloaded. CPMS must be downloaded from BASE SYSTEM in advance.	CPMSダウンロードが未実施です。	BASE SYSTEM/S10VEでCPMSダウンロードを実行してください。

(*1) Communication error発生時は、下記を参照してください。

<Communication errorのRCの意味>

0x11 : ソケット異常	0x04 : フレーム作成用メモリ確保失敗
0x12 : IPアドレス異常	0x05 : データ送信失敗
0x14 : 格納エリアアドレス異常 (0指定、dta)	0x06 : レスポンス受信待ち異常
0x15 : 格納エリアアドレス異常 (0指定、wka)	0x07 : レスポンス未受信でリトライオーバー
0x16 : サイズ異常 (0以下または16KB以上)	0x08 : データ受信失敗
0x17 : サイズ異常 (非ロングワードサイズ)	0x18 : 格納エリアアドレス異常 (0指定、dmaia)
0x03 : 相手アダプタ種別異常	0x19 : 格納エリアアドレス異常 (0指定、reta)
0x8000000X : レスポンスで異常報告 (CPU制御ヘッダ内ステータスコード)	
X : ステータスコード、4 : μΣ1000のネットワーク未設定	
0xFFFFFFFF : 環境ファイル設定誤り	

(*2) A-10ページのNo.41のファイルを確認してください。

(*3) コマンドI/Fでタイムアウトが発生した場合に表示されるコマンドコードの意味は、下記のとおりです。

0x01C00000 : 物理アドレスアクセス
0x11C00000 : 時刻設定
0x21C00000 : 一括ROMセーブ

(4) svcpuctlコマンドエラーメッセージ一覧

No.	エラーメッセージ	意味	ユーザの対応
1	No sitename given for -u option	ユニット名称が指定されていません。	ユニット名称を指定してください。
2	unknown RSSITE	指定サイト名称がありません。	指定サイトを確認してください。
3	Site=%s not found		
4	Internal error (%s)	内部エラーが発生しました。	再試行してください。
5	site (%s) lock busy	サイトは、他の処理で使われています。	
6	site (%s) lock error		
7	communication error (%s, RC=0x%x, エラー発生アドレス)	通信エラーが発生しました。	RCを基に原因を確認してください。 (*) ST#指定誤り(%s:rrw_rpl_p)。または、イーサケーブル未接続 (%s:set_ip) です。指定を見直してください。
8	communication error (%s, RC=0x%x)		
9	%s (slot=%d) NON EXIST	スロットが存在しません。	システムジェネレーション情報を確認してください。
10	site (%s) unlock error	サイトの占有解除でエラーが発生しました。	再試行してください。
11	Site=%s is NP or HP site	指定サイトは、HPのサイトです。	CPのサイトを指定してください。
12	Command I/F (command code=0x11C00000) time out	コマンドI/Fの時刻設定でタイムアウトが発生しました。	ネットワークの接続状態を確認して再試行してください。
13	Not RPDUsers	RPDUsers権限を持たないユーザです。	RPDUsers権限を持つユーザで実行してください。
14	Usage:svcpuctl [{-u site} {-s {-stop -run}}] [-time] Usage:svcpuctl [-u site] -ss	—	—

(*) Communication error発生時は、下記を参照してください。

<Communication errorのRCの意味>

- | | |
|---|--------------------------------|
| 0x11 : ソケット異常 | 0x04 : フレーム作成用メモリ確保失敗 |
| 0x12 : IPアドレス異常 | 0x05 : データ送信失敗 |
| 0x14 : 格納エリアアドレス異常 (0指定、dta) | 0x06 : レスポンス受信待ち異常 |
| 0x15 : 格納エリアアドレス異常 (0指定、wka) | 0x07 : レスポンス未受信でリトライオーバー |
| 0x16 : サイズ異常 (0以下または16KB以上) | 0x08 : データ受信失敗 |
| 0x17 : サイズ異常 (非ロングワードサイズ) | 0x18 : 格納エリアアドレス異常 (0指定、dmaia) |
| 0x03 : 相手アダプタ種別異常 | 0x19 : 格納エリアアドレス異常 (0指定、reta) |
| 0x8000000X : レスポンスで異常報告 (CPU制御ヘッダ内ステータスコード) | |
| X : ステータスコード、4 : μΣ1000のネットワーク未設定 | |
| 0xFFFFFFFF : 環境ファイル設定誤り | |

(5) svelogコマンドエラーメッセージ一覧

No.	エラーメッセージ	意味	ユーザの対応
1	Usage: svelog [-u site] [-f {s m l}] [-logno] [+case] [-d fname] [-o fname]	オプションに誤りがあります。	正しいオプションで起動してください。
2	Unknown RSSITE	RSSITE環境変数が設定されていません。	RSSITE環境変数を設定してください。
3	logno error. logno is 1-999	指定したログ番号が範囲外です。	ログ番号を見直してください。
4	unknown site (サイト名)	指定されたサイトがありません。	サイト名を見直してください。
5	communication error (エラーコード)	開発系マシンとS10VE間の送受信に失敗しました。	エラーコードをもとに原因を確認してください。 (*2)
6	memory allocate error	メモリが確保できませんでした。	再試行してください。
7	logno "ログ番号": not found	指定されたログ番号のエラーログがありません。	ログ番号を見直してください。
8	no error log.	エラーログがありません。	エラーは発生していません。
9	cannot open "ファイル名"	ファイルのオープンに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
10	cannot read "ファイル名"	ファイルの読み出し時に異常を検出しました。	
11	specified logno is not found	指定したログ番号のエラーログがありません。	ログ番号を見直してください。
12	svelog : Invalid file name (XXXX)	XXXXに出力されたファイル名称に誤りがあります。	正しいファイル名称で再起動してください。

(*1) 調査用のデータは下記のファイルです。

- ・ %windir%\¥renix¥etc¥log¥rpdh_hce¥RPDP_S10VE
- ・ %windir%\¥renix¥etc¥log¥rpdh_hce¥1_RPDP_S10VE

(*2) communication error発生時は、下記を参照してください。

< communication errorのエラーコードの意味 >

- | | |
|---|--------------------------------|
| 0x11 : ソケット異常 | 0x04 : フレーム作成用メモリ確保失敗 |
| 0x12 : IPアドレス異常 | 0x05 : データ送信失敗 |
| 0x14 : 格納エリアアドレス異常 (0指定、dta) | 0x06 : レスポンス受信待ち異常 |
| 0x15 : 格納エリアアドレス異常 (0指定、wka) | 0x07 : レスポンス未受信でリトライオーバー |
| 0x16 : サイズ異常 (0以下または16KB以上) | 0x08 : データ受信失敗 |
| 0x17 : サイズ異常 (非ロングワードサイズ) | 0x18 : 格納エリアアドレス異常 (0指定、dmaia) |
| 0x03 : 相手アダプタ種別異常 | 0x19 : 格納エリアアドレス異常 (0指定、reta) |
| 0x8000000X : レスポンスで異常報告 (CPU制御ヘッダ内ステータスコード) | |
| X : ステータスコード、4 : μΣ1000のネットワーク未設定 | |
| 0xFFFFFFFF : 環境ファイル設定誤り | |

(6) svdhpコマンドエラーメッセージ一覧

(1/3)

No.	エラーメッセージ	意味	ユーザの対応
1	usage: svdhp [-u site] [+count] [-on -off]-stat] [-d fname] [-o fname] [-all [fname]]-f fname] [-freeze]	オプションに誤りがあります。	正しいオプションで起動してください。
2	Unknown RSSITE	RSSITE環境変数が設定されていません。	RSSITE環境変数を設定してください。
3	No such site (サイト名)	指定されたサイトがありません。	サイト名を見直してください。
4	No such PUName (CPU名)	指定されたCPUがありません。	CPU名を見直してください。
5	specified CPU (CPU名) is CP only	指定されたCPUはCPだけです。	
6	Some system constants are not defined	定義されていないシステム定数があります。	システム定数の定義を見直してください。
7	Bad realtime environment	システム環境が異常です。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*)
8	memory allocate error	メモリが確保できませんでした。	一時的に空きメモリが不足しエラーが発生した可能性があります。メモリに空きがあることを確認後に再実行してください。
9	cannot open "ファイル名"	ファイルのオープンに失敗しました。	ファイルやディレクトリのセキュリティなどの状態を確認してください。
10	cannot read "ファイル名"	ファイルの読み出し時に異常を検出しました。	
11	cannot write "ファイル名"	ファイルの書き込み時に異常を検出しました。	
12	No such AREA (DHP_RD) in salmt	グローバルエリアにDHP読み出しエリア (DHP_RD) がありません。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*)
13	Memory access error	S10VEメモリのアクセスに失敗しました。	

(2/3)

No.	エラーメッセージ	意味	ユーザの対応
14	Memory allocation error (malloc, dhp read area)	dhp読み出しエリアが確保できませんでした。	一時的に空きメモリが不足しエラーが発生した可能性があります。メモリに空きがあることを確認後に再実行してください。
15	Communication error (catch signal)	シグナルを受信しました。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再実行してください。
16	Communication error (connection timeout)	コネクションタイムアウトが発生しました。	
17	Communication error (connection refused)	RPCサーバが存在しません。	
18	Communication error (connection cut)	コネクションが切断されました。	
19	Communication error (connection reset)	コネクションがリセットされました。	
20	Communication error (server closed)	RPCサーバがクローズされました。	調査用のデータを収集し、開発系マシンを再起動してください。 (*)
21	Communication error (port busy)	回線ポートがbusy状態です。	他の通信の終了を待ち再実行してください。
22	Communication error (socket create error)	ソケットの生成に失敗しました。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再実行してください。
23	Communication error (no buffer)	メモリ獲得に失敗しました。	
24	Communication error (network not reached)	ネットワークが未接続状態です。	
25	Communication error (network down)	ネットワーク接続インターフェースがダウンしました。	
26	Communication error (port No error)	ポート番号の取り込みに失敗しました。	
27	Communication error (IP address error)	IPアドレスの取り込みに失敗しました。	
28	Communication error (memory attach failed)	共有メモリのアタッチに失敗しました。	
29	Communication error (trace file cannot open)	トレースファイルのオープンに失敗しました。	
30	Communication error (trace file cannot copy)	トレースファイルのコピーに失敗しました。	

No.	エラーメッセージ	意味	ユーザの対応
31	Communication error (fatal error)	致命的なエラーが発生しました。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再試行してください。
32	dhp data read error	dhpトレースデータを読み出し時にエラーを検出しました。	調査用のデータを収集し、開発系マシンを再起動してください。 (*)
33	Cannot dhp trace ON/OFF	dhpトレース制御時にエラーを検出しました。	
34	svdhp : Invalid file name (XXXX)	XXXXに出力されたファイル名称に誤りがあります。	正しいファイル名称で再起動してください。
35	svdhp : DHP data illegal	ファイルの内容が異常です。	正しいファイルを指定してください。
36	Not RPDUsers	RPDUsers権限を持たないユーザです。	RPDUsers権限を持つユーザで実行してください。

(*) 調査用のデータは下記のファイルです。

- %windir%\¥renix¥etc¥log¥rpdp_hce¥RPDP_S10VE
- %windir%\¥renix¥etc¥log¥rpdp_hce¥1_RPDP_S10VE

(7) svcpunowコマンドエラーメッセージ一覧

(1/3)

No.	エラーメッセージ	意味	ユーザの対応
1	Usage:svcpunow [-u site] [-t second]	オプションに誤りがあります。	正しいオプションで起動してください。
2	Unknown RSSITE	RSSITE環境変数が設定されていません。	環境変数RSSITEを設定後、再試行してください。
3	memory allocation error (malloc, puloadinfo area)	PU負荷率情報の読み出しエリア確保に失敗しました。	一時的に空きメモリが不足しエラーが発生した可能性があります。メモリに空きがあることを確認後に再実行してください。
4	cannot get PU load information	PU負荷率の取り込みに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。
5	Not available parameter	使用できないパラメータを検出しました。	(*1)
6	No such AREA (puloadinfo) in salmt	PU負荷率読み出しエリアが検索できませんでした。	
7	Communication error (catch signal)	シグナルを受信しました。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再試行してください。
8	Communication error (connection timeout)	タイムアウトが発生しました。	
9	Communication error (connection refused)	RPCサーバが存在しません。	
10	Communication error (connection cut)	RPCサーバが切断されています。	
11	Communication error (connection reset)	コネクションがリセットされています。	
12	Communication error (server closed)	RPCサーバがクローズされています。	
13	Communication error (port busy)	回線ポートがbusy状態です。	他の通信の終了を待ち、再試行してください。
14	Communication error (bad socket specified)	ソケット記述子の指定に誤りがあります。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再試行してください。
15	Communication error (socket create error)	ソケットの生成に失敗しました。	

No.	エラーメッセージ	意味	ユーザの対応
16	Communication error (no buffer)	メモリ獲得に失敗しました。	一時的に空きメモリが不足しエラーが発生した可能性があります。メモリに空きがあることを確認後に再実行してください。
17	Communication error (network not reached)	ネットワークが未接続状態です。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1) (*2)
18	Communication error (network down)	ネットワーク接続インタフェースがダウンしています。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再実行してください。
19	Communication error (port No error)	ポート番号の取り込みに失敗しました。	
20	Communication error (IP address error)	IPアドレスの取り込みに失敗しました。	
21	Communication error (memory attach failed)	共有メモリのアタッチに失敗しました。	
22	Communication error (trace file cannot open)	トレースファイルのオープンに失敗しました。	
23	Communication error (trace file cannot copy)	トレースファイルのコピーに失敗しました。	
24	Communication error (fatal error)	致命的なエラーが発生しました。	
25	Communication error (cannot connection errno = %x) (*2)	通信回線の確立に失敗しました。	
26	Communication error (RRB errno = %x) (*2)	メモリの読み出しに失敗しました。	
27	Memory access error	S10VEメモリの読み出し、書き込みに失敗しました。	
28	target status error	S10VEコマンドサポートタスクの起動に失敗しました。	
29	Cannot get TCB	TCB情報の読み出しに失敗しました。	
30	command is already execution	他でPU負荷率を測定中のため実行できません。	再実行してください。

(3/3)

No.	エラーメッセージ	意味	ユーザの対応
31	No sitename given for -u option	サイト名称の指定がありません。	入力可能データを确认后、再試行してください。
32	Site=%s not found	指定したサイトがありません。	
33	PU load measuring period error [second = 1-3600]	測定時間の指定に誤りがあります。	
34	Not RPDUsers	RPDUsers権限を持たないユーザです。	RPDUsers権限を持つユーザで実行してください。

(*1) 調査用のデータは下記のファイルです。

- %windir%\¥renix¥etc¥log¥rpdp_hce¥RPDP_S10VE
- %windir%\¥renix¥etc¥log¥rpdp_hce¥1_RPDP_S10VE

(*2) Communication error発生時のerrnoは、下記を参照してください。

- | | |
|---|--------------------------------|
| 0x11 : ソケット異常 | 0x04 : フレーム作成用メモリ確保失敗 |
| 0x12 : IPアドレス異常 | 0x05 : データ送信失敗 |
| 0x14 : 格納エリアアドレス異常 (0指定、dta) | 0x06 : レスポンス受信待ち異常 |
| 0x15 : 格納エリアアドレス異常 (0指定、wka) | 0x07 : レスポンス未受信でリトライオーバー |
| 0x16 : サイズ異常 (0以下または16KB以上) | 0x08 : データ受信失敗 |
| 0x17 : サイズ異常 (非ロングワードサイズ) | 0x18 : 格納エリアアドレス異常 (0指定、dmaia) |
| 0x03 : 相手アダプタ種別異常 | 0x19 : 格納エリアアドレス異常 (0指定、reta) |
| 0x8000000X : レスポンスで異常報告 (CPU制御ヘッダ内ステータスコード) | |
| X : ステータスコード、4 : $\mu\Sigma$ 1000のネットワーク未設定 | |
| 0xFFFFFFFF : 環境ファイル設定誤り | |

(8) svtimexコマンドエラーメッセージ一覧

(1/3)

No.	エラーメッセージ	意味	ユーザの対応
1	Usage:svtimex [-u site] [tname] [-t second] [tn]	オプションに誤りがあります。	正しいオプションで起動してください。
2	Unknown RSSITE	RSSITE環境変数が設定されていません。	環境変数RSSITEを設定後、再試行してください。
3	memory allocation error (malloc, taskloadinfo area)	PU負荷率情報の読み出しエリア確保に失敗しました。	一時的に空きメモリが不足しエラーが発生した可能性があります。メモリに空きがあることを確認後に再実行してください。
4	cannot get task load information	タスク稼働率の取り込みに失敗しました。	調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
5	cannot get task load information Taskname=%s (%s)	タスク稼働率の取り込みに失敗しました (タスクが特定できる場合)。	
6	Not available parameter	使用できないパラメータを検出しました。	
7	No such AREA (puploadinfo) in salmt	タスク稼働率読み出しエリアが検索できませんでした。	
8	Communication error (catch signal)	シグナルを受信しました。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再試行してください。
9	Communication error (connection timeout)	タイムアウトが発生しました。	
10	Communication error (connection refused)	RPCサーバが不在です。	
11	Communication error (connection cut)	RPCサーバが切断されています。	
12	Communication error (connection reset)	コネクションがリセットされています。	
13	Communication error (server closed)	RPCサーバがクローズしています。	
14	Communication error (port busy)	回線ポートがbusy状態です。	

(2/3)

No.	エラーメッセージ	意味	ユーザの対応	
15	Communication error (bad socket specified)	ソケット記述子の指定に誤りがあります。	ネットワークの接続状態、S10VE、開発系マシンのIPアドレスを確認して再試行してください。	
16	Communication error (socket creat error)	ソケットの生成に失敗しました。		
17	Communication error (no buffer)	メモリ獲得に失敗しました。		
18	Communication error (network not reached)	ネットワークが未接続状態です。		
19	Communication error (network down)	ネットワーク接続インタフェースがダウンしています。		
20	Communication error (port No error)	ポート番号の取り込みに失敗しました。		
21	Communication error (IP address error)	IPアドレスの取り込みに失敗しました。		
22	Communication error (memory attach failed)	共有メモリのアタッチに失敗しました。		
23	Communication error (trace file cannot open)	トレースファイルのオープンに失敗しました。		
24	Communication error (trace file cannot copy)	トレースファイルのコピーに失敗しました。		
25	Communication error (fatal error)	致命的なエラーが発生しました。		
26	Communication error (cannot connection errno = %x) (*2)	通信回線の確立に失敗しました。		
27	Communication error (RRB errno = %x) (*2)	メモリの読み出しに失敗しました。		
28	Memory access error	メモリの読み出し、書き込みに失敗しました。		調査用のデータを収集し、開発系マシンを再立ち上げしてください。 (*1)
29	target status error	コマンドサポートタスクの起動に失敗しました。		
30	Cannot get TCB	TCB情報の読み出しに失敗しました。		

No.	エラーメッセージ	意味	ユーザの対応
31	command is already execution	他でタスク稼働率を測定中のため実行できません。	再試行してください。
32	No sitename given for -u option	サイト名称の指定がありません。	入力可能データを確認後、再試行してください。
33	Site=%s not found	指定したサイトがありません。	
34	Task measuring period error [second = 1-86400]	測定時間の指定に誤りがあります。	
35	taskname or number set data over (max=10)!!	タスク名称または番号の指定が10個を超えました。	
36	taskname or number error (%s)	タスク名称または番号の指定に誤りがあります。	
37	%s (%s) task non exist or unmatched	指定されたタスクは開発系マシンとS10VE両方に登録されていません。	タスクを開発系マシンとコントローラ両方に登録してください。
38	Not RPDUsers	RPDUsers権限を持たないユーザです。	RPDUsers権限を持つユーザで実行してください。

(*1) 調査用のデータは下記のファイルです。

- ・ %windir%\¥renix¥etc¥log¥rpdp_hce¥RPDP_S10VE
- ・ %windir%\¥renix¥etc¥log¥rpdp_hce¥1_RPDP_S10VE

(*2) Communication error発生時のerrorは、下記を参照してください。

- | | |
|------------------------------|--------------------------------|
| 0x11 : ソケット異常 | 0x04 : フレーム作成用メモリ確保失敗 |
| 0x12 : IPアドレス異常 | 0x05 : データ送信失敗 |
| 0x14 : 格納エリアアドレス異常 (0指定、dta) | 0x06 : レスポンス受信待ち異常 |
| 0x15 : 格納エリアアドレス異常 (0指定、wka) | 0x07 : レスポンス未受信でリトライオーバー |
| 0x16 : サイズ異常 (0以下または16KB以上) | 0x08 : データ受信失敗 |
| 0x17 : サイズ異常 (非ロングワードサイズ) | 0x18 : 格納エリアアドレス異常 (0指定、dmaia) |
| 0x03 : 相手アダプタ種別異常 | 0x19 : 格納エリアアドレス異常 (0指定、reta) |
- 0x8000000X : レスポンスで異常報告 (CPU制御ヘッダ内ステータスコード)
 X : ステータスコード、4 : μΣ1000のネットワーク未設定
 0xFFFFFFFF : 環境ファイル設定誤り

(9) svdatagenコマンドエラーメッセージ一覧

(1/2)

No.	エラーメッセージ	意味	ユーザの対応	制限
—	Usage: svdatagen [-u site] file	コマンドの起動形式に誤りがあります。	オプションの指定形式を見直してください。	
—	Cannot open input file (%s)	入力ファイルがオープンできません。	指定ファイルのパスとアクセス権を見直してください。	
—	Cannot open include file (%s)	インクルードファイルがオープンできません。	インクルードファイルの指定を見直してください。	プ
—	sitename is too long	サイト名の指定が14文字を超えています。	サイト名の指定を見直してください。	
—	No such site (%s)	指定サイトが存在しません。		
—	No type nor storage class	宣言文に型指定子か記憶クラス指定子がありません。	宣言文の記述を見直してください。	制(14)
—	Illegal token (%s)	構文が誤っています。または、未サポート構文を使用しています。	入力ファイルの内容を見直してください。	制(7), (9), (10), (11), (12), (13), (15), (16), (17)
—	Multiple storage classes	記憶クラス指定子の記述が誤っています。	記憶クラス指定子の記述を見直してください。	制(3)
—	Unexpected token appeared (%s)	未サポート構文を検出しました。	入力ファイルの内容を見直してください。	制(2), (3), (4), (5), (6), (10)
—	Illegal array size (%s)	配列の要素数は省略できません。	配列の記述を見直してください。	制(8)
—	Invalid initializer (%s)	変数の型に変換できない初期値が現れました。	初期値の記述を見直してください。	初
—	Illegal type combination	型指定子の組み合わせが誤っています。	型指定子の記述を見直してください。	制(4)
—	#define expected. (%s)	インクルードファイルの中に#define以外の記述が現れました。	インクルードファイルの内容を見直してください。	プ
—	Invalid define value (%s)	define値の記述が誤っています。	#defineの記述を見直してください。符号付きの8進数、16進数がないかどうか確認してください。	プ

制限は第1編「4. 4 データジェネレータ」との対応を示しています。

詳細は対応する制限事項を参照してください。

制：(b) C言語の宣言文との相違点と制限事項

プ：(e) プリプロセッサ機能の制限事項

初：(f) 初期値の型変換の仕様

No.	エラーメッセージ	意味	ユーザの対応	制限
—	EOF encounterd in a comment	コメントの終わりが見つかりません。	コメントの記述を見直してください。	制(18)
—	Too few initializer (%s)	配列、構造体の初期値が中カッコ ({}) で囲まれていません。	配列、構造体の初期値を見直してください。	制(17)
—	Too many initializer (%s)	初期値が多すぎます。	初期値の記述を見直してください。	
—	Incomplete tag used in declaration (%s)	未定義の構造体タグを使用しています。	構造体の宣言を見直してください。	制(4)
—	Illegal void type	ポインタでないvoid型の変数に初期値が設定されています。	void型の指定を見直してください。	
—	Undeclared name (%s)	指定define値またはGLB、VAL名が見つかりません。	入力ファイルの内容を見直してください。サイトのGLB、VALの登録状況を確認してください。	
—	Site is not specified	GLB、VAL名のアドレスを解決しようとしていますが、サイトが指定されていません。	サイトを指定して再試行してください。	
—	Cannot alloc memory	mallocでメモリが確保できません。	メモリの空きを確認し再試行してください。	
—	Allocator management table is busy	アロケータ管理テーブルが他のコマンドによって使用中です。	他のコマンドの終了を待って再試行してください。	
—	Cannot make temporary file (%s)	テンポラリファイル名が生成できませんでした。	出力ファイルが生成されるディレクトリ、環境変数 (%TMP%) で示されるディレクトリのアクセス権と、ディスクの空き容量を確認して再試行してください。	
—	Cannot open temporary file (%s)	テンポラリファイルが生成できませんでした。		
—	Cannot write output file	出力ファイルに書き込みできませんでした。		
—	Cannot rename (%s)	出力ファイルにrenameできませんでした。		

制限は第1編「4. 4 データジェネレータ」との対応を示しています。

詳細は対応する制限事項を参照してください。

制：(b) C言語の宣言文との相違点と制限事項

プ：(e) プリプロセッサ機能の制限事項

初：(f) 初期値の型変換の仕様

付録E RPD P使用上の注意事項

(1) svdebug ldサブコマンド中断時のリカバリ処理

RPDPでは、svdebugのldサブコマンド中断時（通信エラーなどによる）のS10VE主メモリと開発系マシンの対象サイトのバックアップファイルの不整合を防止するため、ldサブコマンドのリカバリ処理を行います。ここでいうリカバリ処理とは、ldサブコマンドを実行したあとの状態に移行するため、中断したldサブコマンドを再実行することです。リカバリ処理は、RPDPコマンド実行時または開発系マシンの立ち上げ時に行われます。ただし、通信エラーなどが回復しない場合、“Communication error (connection timeout)”、“Communication error (inter PU communication time out)”などのエラーメッセージが出力されRPDPコマンドが使用できなくなります。

この場合の対処法として以下のオペレーションがあります。

このオペレーション後、RPDPのリカバリ処理を実行しなくてもコマンドの実行を行えるようになります。ただし、ldサブコマンドのリカバリ処理が未実行の状態ですので、このオペレーション以降、対象サイトに対しldサブコマンドの実行は禁止されます。

エラー内容は、下記となります。

Cannot use ld sub comannd after RSSRCV set (NO.2100-44)

この状態は、svrplコマンドによって対象サイトに対しダウンロードが行われるまで継続されます。

(オペレーション)

```
svsitectl -rsrcv サイト名称<CR>
```

(2) 使用上の注意事項

- デバッガコマンド (svdebug) に入力するテキストファイルを他マシンからftp転送して使用する場合、asciiモード指定でファイルを転送してください。

<制限事項>

- 以下の名称は、予約名であるため、サイト名、ユニット名、ディレクトリ名、ファイル名に使用できません。また、以下の名称に拡張子 (.c.obj.txtなど) を付加した場合も同様です。

• AUX	• CON	• NUL
• COM1	• LPT1	• PRN
• COM2	• LPT2	
• COM3	• LPT3	
• COM4	• LPT4	
• COM5	• LPT5	
• COM6	• LPT6	
• COM7	• LPT7	
• COM8	• LPT8	
• COM9	• LPT9	

- svdebugのld、cm、svサブコマンドの-fオプションで、他のディスク上のファイルを指定できません。

付録F マップの表示フォーマット

マップ情報は、以下に示す情報を出力します。

- (1) ヘッダー、フッター
- (2) 大分割領域情報
- (3) 分割領域情報
- (4) 細分割領域情報
- (5) プログラム情報
- (6) サブプログラム情報
- (7) タスク情報
- (8) グローバル情報
- (9) VAL情報
- (10) IRSUBエン트리情報
- (11) IRGLBエン트리情報
- (12) ULSUBエン트리情報
- (13) 物理メモリの空き情報

<マップ情報の出力形式>

マップ情報は、以下に示す形式で出力できます。

- (1) 階層マップ出力
- (2) アドレス順リスト出力
- (3) 名称順リスト出力
- (4) 番号順リスト出力
- (5) 名称指定出力

階層マップ出力は、指定大分割領域、分割領域単位に、論理空間上に配置されるリソースのマップ情報を階層的に出力します。

リスト出力は、指定情報を、アドレス順、名称順、番号順に並べて出力します。

また、リソースの名称を指定し、その名称単独の情報を出力することもできます。

付録F マップの表示フォーマット

マップ情報の出力フォーマットを以下に示します。

以下に示す表示フォーマット中の下線 () 部分は、出力するマップ情報であり、マップ出力対象によって変化することを表します。

(1) ヘッダー、フッターフィールド

マップ情報は、出力情報の前後にヘッダー、フッターを出力します。

ヘッダーおよびフッターのフォーマットは下記のとおりです。

(a) ヘッダー

```
** allocator map **                                YYYY/MM/DD hh:mm:ss  
  
site name = site
```

**** allocator map ** :**

ヘッダースtringを表示します。

**** allocator map **** : 通常のマップ出力時のヘッダーです。

**** allocator map (CON) **** : コントローラ側の論理空間マップ出力 (-CON指定) 時のヘッダーです。

YYYY/MM/DD hh:mm:ss :

マップ出力コマンド (svmap) を起動した時刻を表示します。

YYYY : 年 (西暦4桁)

MM : 月

DD : 日

hh:mm:ss : 時分秒

site : マップ情報を表示するサイト名を表示します。

(b) フッター

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

(2) 大分割領域情報

システムジェネレーションで定義した大分割領域のマップを表示します。

大分割領域の論理空間上の先頭アドレスは固定です。

```

< garea >
gname      laddr      paddr      size
$MAP       20000000  paddr     size
$CPMS      28000000  paddr     size
$TASK      30000000  paddr     size
$GLBR      40000000  paddr     size
$GLBRW     50000000  paddr     size
$IRSUB     60000000  paddr     size
$CM        70000000  paddr     size
$DCM       75000000  paddr     size
$LADDER    78000000  paddr     size
$USRFUNC   7B000000  paddr     size
$HIFLOW    7C000000  paddr     size

```

gname：大分割領域の名称です。

laddr：大分割領域先頭の論理アドレスです。

paddr：大分割領域先頭の物理アドレスです。

size：大分割領域のサイズです。

表A-3 リアルタイムリソースの管理状態

シンボル	状態	意味
@	not-build	バックアップファイルにだけロードされている状態です。
+	defined-POC	
.	defined	バックアップファイルにも実機メモリにもロードされている状態です。
-	defined-CON	実機メモリにだけロードされている状態です。 ダウンロード後、開発系マシン側だけを削除した状態です。
*	unmatch	バックアップファイルにも実機メモリにもロードされているが、整合の取れていない状態です。
-	non_exist2	dbuildしたIRSUB、組み込みサブプログラムをdloadしないでダウンロードした状態です。

-CON指定時、s、date、lddate、svdateは表示されません。

(3) 分割領域情報

svdfaで確保した分割領域の情報を表示します。

< area >

garea/aname

garea/aname

garea/_____

raddr

raddr

raddr

size

size

size

laddr

laddr

laddr

kind bkupfile

kind bkupfile

kind bkupfile

date

YYYY/MM/DD

hh:mm:ss

lddate

YYYY/MM/DD

hh:mm:ss

svdate

YYYY/MM/DD

hh:mm:ss

garea : 親大分割領域のGAREA名称を表示します。

\$MAP : マップ情報格納領域を表示します。

\$TASK : タスク格納領域を表示します。

\$CM : PU間共有メモリ格納領域を表示します。

\$DCM : 二重化共有メモリ格納領域を表示します。

\$GLBR : 読み出し専用GLB格納領域を表示します。

\$GLBRW : 読み書き両用GLB格納領域を表示します。

\$IRSUB : サブプログラム格納領域を表示します。

\$LADDER : LADDERプログラム格納領域を表示します。

\$USRFUNC : ユーザ演算ファンクション領域を表示します。

\$HIFLOW : HI-FLOWプログラム格納領域を表示します。

aname : 分割領域名称を表示します。

分割領域名称の空白は、空き領域であることを表します。

s : リソースの状態を表します。

リソースの状態については、表A-3を参照してください。

k : 所有者種別を表示します。(s: システム、u: ユーザ)

raddr : 分割領域先頭の大分割領域先頭からの相対アドレスを16進8桁固定で表示します。

size : 分割領域のサイズを16進8桁固定で表示します。

laddr : 分割領域の先頭論理アドレスを16進8桁固定で表示します。

kind : 分割領域種別。

gbi : 初期値あり読み書き両用グローバル領域を表示します。

glbw : 初期値なし読み書き両用グローバル領域を表示します。

glbr : 初期値あり読み出し専用グローバル領域を表示します。

cmi : 初期値ありPU (プロセッサ) 間共有メモリ領域を表示します。

cmw : 初期値なしPU (プロセッサ) 間共有メモリ領域を表示します。

dcmi : 初期値あり二重化 (コントローラ間) 共有メモリ領域を表示します。

dcmw : 初期値なし二重化 (コントローラ間) 共有メモリ領域を表示します。

task : タスク格納領域を表します。

sub : サブプログラム格納領域を表します。

ostbl : OSリザーブ領域を表します。

bkupfile : 分割領域の初期値を格納したバックアップファイル名を表示します。

初期値なしGLB、CM、DCMのエリアの場合は空白で表示します。

date : svdfaで分割領域を生成した時刻を表示します。

lddate : SIOVEメモリにダウンロードした時刻を表示します。

ダウンロードされていない場合は空白で表示します。

svdate : デバッグのsvサブコマンドでバックアップファイルに保存した時刻を表示します。

保存されていない場合は空白で表示します。

date、**lddate**、**svdate**は、-fオプション指定時だけ表示します。

(-CON指定時、s、date、lddate、svdateは表示されません。)

(4) 細分割領域情報

細分割領域の情報を表示します。

```

< sarea >
garea/aname/aname
garea/aname/aname
garea/aname/_____

```

garea : 親大分割領域のGAREA名称を表示します。

表示するGAREA名称の意味は、分割領域情報の表示と同様です。

aname : 分割領域名称を表示します。

sname : 細分割領域名称を表示します。

細分割領域名称の空白は、空き領域であることを表します。

s : リソースの状態を表します。

リソースの状態については、表A-3を参照してください。

k : 所有者種別を表示します。(s : システム、u : ユーザ)

raddr : 細分割領域の先頭アドレスを、分割領域先頭からの相対バイトアドレス (16進8桁固定) で表示します。

size : 細分割領域のサイズを16進8桁固定で表示します。

laddr : 細分割領域の先頭論理アドレスを16進8桁固定で表示します。

date : svdfsで細分割領域を確保した時刻またはsvloadでプログラム、サブプログラムをバックアップファイルにロードした時刻を表示します。

lddate : S10VEメモリにダウンロードした時刻を表示します。

ダウンロードされていない場合は空白で表示します。

svdate : デバッグのsvサブコマンドでバックアップファイルに保存した時刻を表示します。

保存されていない場合は空白で表示します。

date、**lddate**、**svdate**は、-fオプション指定時だけ表示します。

(-CON指定時、s、date、lddate、svdateは表示されません。)

(7) タスク情報
タスクに関する情報を表示します。

<pre>< task-program > tn tname tnox rmtn lvl sp pname st mtn texttop lastaddr tsize dsiz ssize (part) bsiz extra oswork datatop bstop date lddate s k st mtn texttop lastaddr tsize dsiz ssize (part) bsiz extra oswork datatop bstop tn tname s k st mtn texttop lastaddr tsize dsiz ssize (part) bsiz extra oswork datatop bstop YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss</pre>																	
← (a) タスク情報フィールド						→ (b) プログラム情報フィールド						← (c) 詳細情報フィールド					

(a) タスク情報フィールド

tn : タスク番号を10進4桁固定で表示します。

tname : タスク名称を表示します。

s : タスクの状態を表します。

リソースの状態については、表A-3を参照してください。

k : 所有者種別を表示します。(s : システム、u : ユーザ)

tnox : タスク番号を16進4桁固定で表示します。

rmtn : マルチタスク番号 (使用スタック位置) を16進4桁固定で表示します。

シングルタスクの場合は0001と表示します。

lvl : タスクのレベルを16進2桁固定で表示します。

sp : タスクが使用するstackの最終論理アドレスを16進8桁固定で表示します。

(b) プログラム情報フィールド

pname : プログラム名称を表示します。

s : プログラムの状態を表します。

リソースの状態については、表A-3を参照してください。

k : 所有者種別を表示します。(s : システム、u : ユーザ)

st : プログラムの利用状況を表示します。

ls : タスク登録していない状態を表します。

lm : マルチタスクとしてロードし、タスク登録していない状態を表します。

cs : タスク登録した状態を表します。

cm : マルチタスクとしてロードし、タスク登録した状態を表します。

(注) プログラムがタスク生成されていない場合、タスク情報フィールドは空白となります。

mtn : マルチタスクの個数を16進4桁固定で表示します。

シングルタスクの場合は0001と表示します。

texttop : テキスト部の先頭論理アドレスを16進8桁固定で表示します。

lastaddr : プログラムの最終論理アドレスを16進8桁固定で表示します。

tsize : テキスト部のサイズを16進6桁固定で表示します。

dsiz : データ部のサイズを16進6桁固定で表示します。

ssize (part) : stackのサイズを16進6桁固定で表示します。

(part)にはローダに指定したプログラム自身の使用するスタックサイズ表示します。

bsiz : bss部のサイズを16進6桁固定で表示します。

extra : load時に指定した冗長バイトサイズを16進6桁固定で表示します。

oswork : OSワークのサイズを16進6桁固定で表示します。

(c) 詳細情報フィールド

datatop : データ部の先頭論理アドレスを16進8桁固定で表示します。

bsstop : BSS部の先頭論理アドレスを16進8桁固定で表示します。
date : svctaskでタスク生成した時刻を表示します。

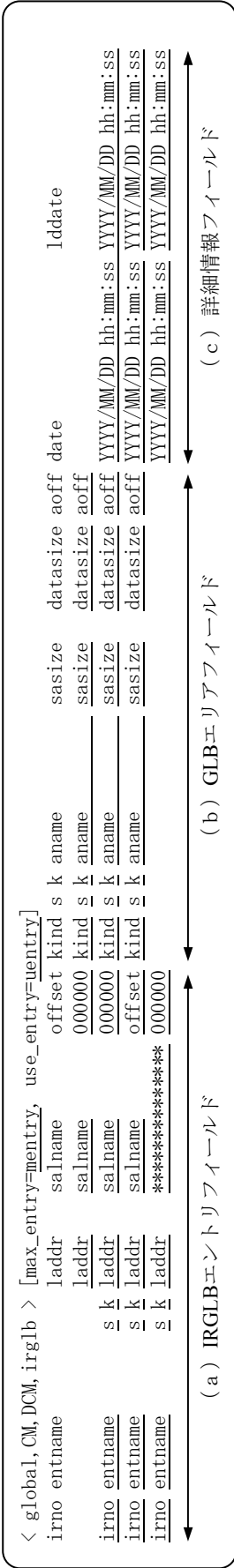
lddate : SIOVEメモリにダウンロードした時刻を表示します。

ダウンロードされていない場合は空白で表示します。

詳細情報フィールドは、-fオプションを指定した場合だけ表示します。

(8) グローバル情報

グローバル (GLB、CM、DCM) に関する情報を表示します。



グローバル情報の表示フォーマットは、「(11) IRGLBエントリー情報」に示すフォーマットと同一です。各フィールドの意味は、「(11) IRGLBエントリー情報」を参照してください。

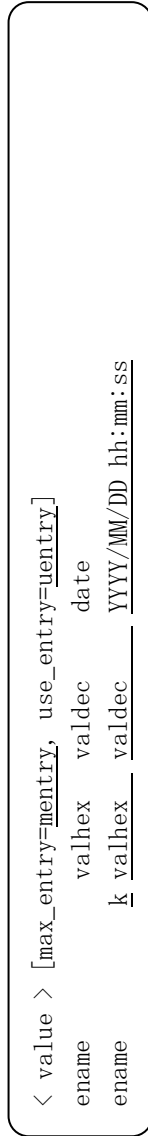
グローバル情報とIRGLBエントリー情報の相違点は以下の2点です。

- ・デフォルトのソート順がグローバル指定 (-g) ではグローバル名称順に、IRGLBエントリー情報 (-irg) ではIRGLB番号順となります。
- ・名称指定の表示で、グローバル指定 (-g) では指定名称をグローバル名として扱います。

IRGLBエントリー (-irg) では指定番号をIRGLB番号として扱います。

(9) VAL情報

バリュ (VAL) に関する情報を表示します。



ename : バリュ名称を表示します。

k : 所有者種別を表示します。(s : システム、u : ユーザ)

valhex : バリュ値を16進8桁固定長で表示します。

valdec : バリュ値を10進10桁可変長で表示します。

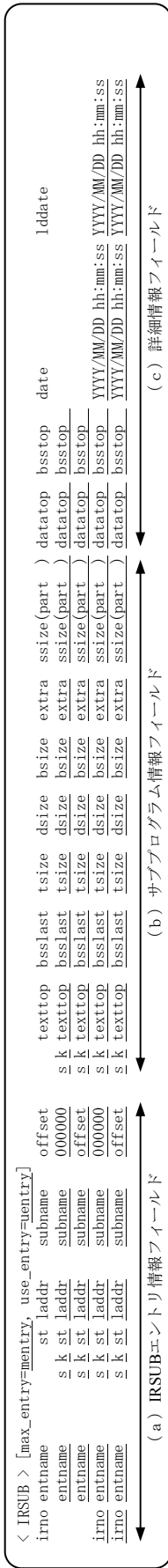
date : 登録時刻を表示します。

mentry : VALの登録可能エントリー数を10進6桁固定で表示します。

uentry : VALの使用中のエントリー数を10進6桁固定で表示します。

dateは、-fオプションを指定した場合だけ表示します。

(10) IRSUBエン트리情報
IRSUBのエン 트리に関する情報を表示します。



mentry : IRSUBの登録可能エン트리数を10進6桁固定で表示します。
umentry : IRSUBの使用中のエン트리数を10進6桁固定で表示します。

(a) IRSUBエン트리情報フィールド

irno : 間接リンクページのエン트리番号。
IRSUBがビルドされていない場合、**irno**は空白となります。
entname : IRSUBのエン트리名称。

s : IRSUBのエントリの状態を表します。
 リソースの状態については、表A-3を参照してください。
k : 所有者種別を表示します。(s : システム、u : ユーザ)
 (IRSUBがビルドされていない場合、空白を表示します。)
st : IRSUBのエントリ種別および割り当て状態を表示します。

il : ビルドされていないIRSUBのトップエン트리関数を表します。
ml : ビルドされていないIRSUBのマルチエン트리関数を表します。
ib : ビルドされているIRSUBのトップエン트리関数を表します。
mb : ビルドされているIRSUBのマルチエン트리関数を表します。

laddr : エントリポイントの論理空間上のアドレスを16進8桁固定で表示します。
subtype : エントリの含まれるサブプログラムの名称を表示します。
offset : サブプログラムの先頭からエントリポイントまでの相対値を16進6桁固定で表示します。

(b) サブプログラム情報フィールド

s : サブプログラムの状態を表します。
 リソースの状態については、表A-3を参照してください。

k : 所有者種別を表示します。(s : システム、u : ユーザ)
texttop : テキスト部の先頭論理アドレスを16進8桁固定で表示します。

bsslast : bss部の最終論理アドレスを16進8桁固定で表示します。
 最終論理アドレスは冗長バイトサイズ (**extra**) を含んだ値です。

tsize : テキスト部のサイズを16進6桁固定で表示します。

dsiz : データ部のサイズを16進6桁固定で表示します。

bsize : bss部のサイズを16進6桁固定で表示します。

extra : load時に指定した冗長バイトサイズを16進6桁固定で表示します。

ssize(part) : stackのサイズを16進6桁固定で表示します。

(part)にはローダに指定したサブプログラム自身の使用するスタックサイズを表示します。

(c) 詳細情報フィールド

datatop : データ部の先頭論理アドレスを16進8桁固定で表示します。

bsstop : BSS部の先頭論理アドレスを16進8桁固定で表示します。
date : svbuildでビルドした時刻を表示します。

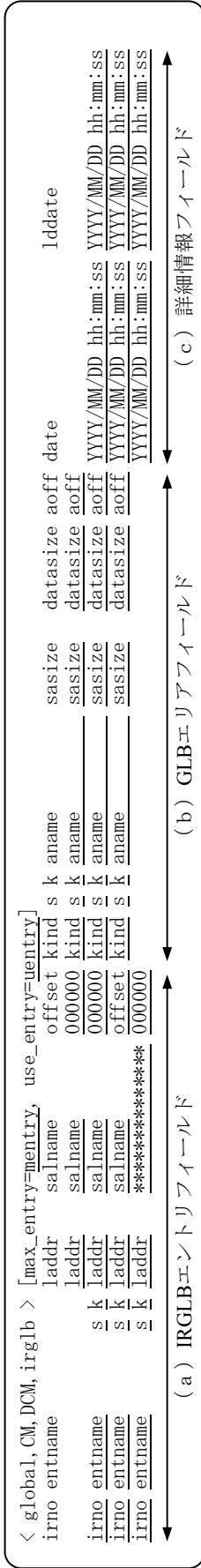
ビルドされていない場合は空白で表示します。

lddate : SIOVEメモリにダウンロードした時刻を表示します。

ダウンロードされていない場合は空白で表示します。

詳細情報フィールドは、-fオプションを指定した場合だけ表示します。

(11) IRGLBエン트리情報



mentry : GLB (CM, DCM含む) の登録可能エントリ数を10進6桁固定で表示します。

uentry : GLB (CM, DCM含む) の使用中のエントリ数を10進6桁固定で表示します (OS使用分の14個を含みます)。

(a) IRGLBエン트리フィールド

irno : 間接リンクテーブルのエントリ番号。
(IRGLBとしてビルドされていない場合は空白を表示します。)

entname : IRGLBのエントリ名称。
(IRGLBとしてビルドされていない場合は空白を表示します。)

s : IRGLBエントリの状態を表します。
リソースの状態については、表A-3を参照してください。
(IRGLBとしてビルドされていない場合は空白を表示します。)

k : 所有者種別を表します。(s: システム, u: ユーザ)
(IRGLBとしてビルドされていない場合は空白を表示します。)

laddr : エントリポイントの論理空間上のアドレスを16進8桁固定で表示します。
(IRGLBとしてビルドされていない場合はGLB(sarea)のアドレスを表示します。)

salname : エントリの含まれる細分割領域の名称を表示します。
エントリのアドレスが絶対アドレスで与えられている場合は、* (アスタリスク) で表示します。

offset : 細分割領域の先頭からエントリポイントまでの相対値を16進6桁固定で表示します。

(b) GLBエリアフィールド

kind : 細分割領域種別。
glbi : 初期値あり読み書き両用グローバル領域を表します。

glbw : 初期値なし読み書き両用グローバル領域を表します。

glbr : 初期値あり読み出し専用グローバル領域を表します。

cmi : 初期値ありPU (プロセッサ) 間共有メモリ領域を表します。

cmw : 初期値なしPU (プロセッサ) 間共有メモリ領域を表します。

dcmi : 初期値あり二重化 (コントローラ間) 共有メモリ領域を表します。

dcmw : 初期値なし二重化 (コントローラ間) 共有メモリ領域を表します。

s : GLBの状態を表します。
リソースの状態については、表A-3を参照してください。

k : 所有者種別を表します。(s: システム, u: ユーザ)
aname : 親分割領域の名称を表示します。

asize : 領域のサイズを16進8桁固定で表示します。
datasize : データのサイズを16進8桁固定で表示します。

aoff : 分割領域先頭からの相対バイトアドレスを16進8桁固定で表示します。

(c) 詳細情報フィールド

date : **svirglb**または**svdfs-e**でIRGLBとしてビルドした時刻を表示します。
IRGLBとしてビルドされていない場合は空白で表示します。

lddate : **S10VE**メモリにダウンロードした時刻を表示します。
ダウンロードされていない場合は空白で表示します。
詳細情報フィールドは、**-f**オプションを指定した場合だけ表示します。

(12) ULSUB エン트리 情報

```

< ULSUB >
pnt typ ent      subname      texttop  bsslast  tsize  dsize  bsize  extra  ssize(part)  datatop  bsstop  lddate
pnt typ ent b subname      s k texttop  bsslast  tsize  dsize  bsize  extra  ssize(part)  datatop  bsstop  bssstop  YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss

```

- pnt** : 組み込みサブルーチンの組み込みポイントを表示します。
- typ** : 組み込みサブルーチンのタイプ (osまたはuser) を表示します。
- ent** : 組み込みサブルーチンのエン트리番号を表示します。
- b** : 組み込みサブルーチンのビルド状態を表します。
 リソースの状態については、表 A-3 を参照してください。
- subname** : 組み込みサブルーチン名称を表示します。
- s** : 組み込みサブルーチンの状態を表します。
 リソースの状態については、表 A-3 を参照してください。
- k** : 所有者種別を表示します。 (s : システム、u : ユーザ)
- texttop** : テキスト部の先頭論理アドレスを16進8桁固定で表示します。
- bsslast** : bss部の最終論理アドレスを16進8桁固定で表示します。
 最終論理アドレスは冗長バイトサイズ (extra) を含んだ値です。
- tsize** : テキスト部のサイズを16進6桁固定で表示します。
- dsize** : データ部のサイズを16進6桁固定で表示します。
- bsize** : bss部のサイズを16進6桁固定で表示します。
- extra** : load時に指定した冗長バイトサイズを16進6桁固定で表示します。
- ssize(part)** : stackのサイズを16進6桁固定で表示します。
 (part)にはローダに指定したサブプログラム自身の使用するスタックサイズを表示します。
- datatop** : データ部の先頭論理アドレスを16進8桁固定で表示します。
- bsstop** : BSS部の先頭論理アドレスを16進8桁固定で表示します。
- date** : svbuildで組み込みサブルーチンとしてビルドした時刻を表示します。
 ビルドされていない場合はloadした時刻を表示します。
- lddate** : S10VEメモリにダウンロードした時刻を表示します。
 ダウンロードされていない場合は空白で表示します。
 詳細情報フィールドは、-fオプションを指定した場合だけ表示します。

- (15) デフォルトの表示フォーマット
 (a) デフォルトの表示フォーマット (詳細情報なし)
 -u オプション以外のすべてのオプションを省略した場合は、分割領域・細分割領域のアドレス順リスト、タスク・プログラム、IRSUB、組み込みサブルーチン、IRGLBの番号順リスト、VALの名称順リストを詳細情報なしで出力します。表示形式を以下に示します。

```

** allocator map **
site name = site
< garea >
gname laddr paddr size
gname laddr paddr size
< area >
garea/aname raddr size kind bkupfile
garea/aname s k raddr laddr kind bkupfile
garea/aname raddr laddr
< sarea >
garea/aname/ raddr size laddr
garea/aname/ s k raddr laddr
garea/aname/ raddr
< task-program >
tn tname tnox rmtn lvl sp
tn tname s k tn tnox rmtn lvl sp
< IRSUB > [max_entry=entry, use_entry=entry]
irno entname st laddr subname offset
entname s k st laddr subname 000000
entname s k st laddr subname offset
irno entname s k st laddr subname 000000
irno entname s k st laddr subname offset
< ULSUB >
pnt typ ent subname texttop bsslast tsize dsiz extra ssize (part)
pnt typ ent b subname s k texttop bsslast tsize dsiz extra ssize (part)
< global, CM, DCM, irglb > [max_entry=entry, use_entry=entry]
irno entname laddr salname offset kind s k aname datasize aoff
irno entname s k laddr salname 000000 kind s k aname datasize aoff
irno entname s k laddr salname 000000 kind s k aname datasize aoff
irno entname s k laddr salname ***** kind s k aname datasize aoff
< value > [max_entry=entry, use_entry=entry]
ename valhex valdec
ename k valhex valdec
** map output end **
    
```

(b) デフォルトの表示フォーマット (詳細表示)

-u、-fオプション以外のすべてのオプションを省略した場合は、分割領域・細分割領域のアドレス順リスト、タスク・プログラム、IRSUB、組み込みサブルーチン、IRGLBの番号順リスト、VALの名称順リストを詳細情報付きで出力します。表示形式を以下に示します。

```

** allocator map **
site name = site
< garea >
gname laddr paddr size
gname laddr paddr size
< area >
gares/ aname laddr size kind bkupfile svdate
gares/ aname laddr size laddr kind bkupfile YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
gares/ aname/ raddr size laddr laddr date lddate
gares/ aname/ s k raddr size laddr laddr size lddate
gares/ aname/ raddr size laddr laddr size lddate
gares/ aname/ s k raddr size laddr laddr size lddate
< task-program >
tn tname tnox rmtn lvl sp pname
tn tname s k tnox rmtn lvl sp pname
< IRSUB > [max_entry=entry, use_entry=entry]
irno entname laddr subname offset
irno entname s k laddr subname 000000
irno entname s k laddr subname offset
irno entname s k laddr subname 000000
irno entname s k laddr subname offset
< ULSUB >
pnt typ ent subname texttop bsslast tsize dsize bsize extra ssize(part) datatop bsstop date lddate
pnt typ ent b subname s k texttop bsslast tsize dsize bsize extra ssize(part) datatop bsstop YYYY/MM/DD hh:mm:ss YYYY/MM/DD hh:mm:ss
< IRGLE_GLE > [max_entry=entry, use_entry=entry]
irno entname laddr salname offset kind s k aname sasize datasize aoff date
irno entname s k laddr salname 000000 kind s k aname sasize datasize aoff
irno entname s k laddr salname 000000 kind s k aname sasize datasize aoff
irno entname s k laddr salname ***** kind s k aname sasize datasize aoff
irno entname s k laddr ***** kind s k aname sasize datasize aoff
< value > [max_entry=entry, use_entry=entry]
ename valhex valdec valdec YYYY/MM/DD hh:mm:ss
** map output end **

```

付録G svdebug (オンラインデバッガ) md、sdの表示フォーマット

(1) mdサブコマンドの表示フォーマット

- データ表示 (プリント) 時の表示フォーマット

```

0xaaaaaaaa dddddddd dddddddd dddddddd dddddddd '.....'
  アドレス      表示データ      文字コード
    
```

アドレス：表示データの先頭アドレスを16進で表示します。

表示データ：指定されたデータ出力形式、データ長に従い、アドレスの内容を表示します。

1行で最大16バイト分のデータを表示します。

データ出力形式が浮動小数点 (-f-l、-fd) のとき、アドレスの内容が下記データである場合は、16進数に変換して表示します。また、16進数表示のあとに対応する文字列を表示します。

浮動小数点データ	文字列	表示例	
		単精度	倍精度
非数	Na	0x7fffffff : Na	0xffff00000 0x00000001 : Na
無限大	In	0x7f800000 : In	0xffff00000 0x00000000 : In
表現できる最大値	Ma	0x7f7fffff : Ma	0x7fefffff 0xffffffffff : Ma
表現できる最小値	Mi	0xff7fffff : Mi	0xffefffff 0xffffffffff : Mi

文字コード：データ出力形式が16進 (-h) のときだけデータの内容を文字コードに置き換えて画面右部に表示します。置き換えできないデータは“.”で表示します。

直前に表示した行の内容と同一データが行単位で連続する場合、下記のメッセージを表示します (-allオプション指定時は、連続するデータすべてを表示します)。

```

0xaaaaaaaa-0xaaaaaaaa as previous
  |
  |-----> 同一データの先頭アドレス
  |
  |-----> 同一データの最終アドレス
    
```

- データ変更 (パッチ) 時の表示フォーマット

```

0xaaaaaaaa dddddddd :
  アドレス 表示データ
    
```

● 表示例

データ出力形式とデータ長の両オプションの組み合わせによるmdの表示例を、下図に示します。

```

*****|*****|*****|*****|*****|*****|*****|*****|
16進4バイト 0x00ec0000 0000000a 00000064 000003e8 00002710 ‘.....d.....’.‘
表示 (-h, -l) 0x00ec0010 000186a0 ‘....’

16進2バイト 0x00ec0000 0000 000a 0000 0064 0000 03e8 0000 2710 ‘.....d.....’.‘
表示 (-h, -w) 0x00ec0010 0001 86a0 ‘....’

16進1バイト 0x00ec0000 00 00 00 0a 00 00 00 64 00 00 03 e8 00 00 27 10 ‘.....d.....’.‘
表示 (-h, -b) 0x00ec0010 00 01 86 a0 ‘....’

10進4バイト 0x00ec0000          10          100          1000          10000
表示 (-d, -l) 0x00ec0010          100000

10進2バイト 0x00ec0000          0          10          0          100          0          1000          0          10000
表示 (-d, -w) 0x00ec0010          1 -31072

10進1バイト 0x00ec0000          0          0          0          10          0          0          0          100          0          0          3 -24          0          0
表示 (-d, -b)          39          16
0x00ec0010          0          1 -122 -96

単精度実数 0x00ec0020          1.1200000          2.1229999          10.1230001          20.1233997
表示 (-f, -l) 0x00ec0030          100.123451

倍精度実数 0x96000000          1.0000000000000000E+00          2.0000000000000000E+00
表示 (-fd) 0x96000010          -1.0000000000000000E+00          1.0000000000000000E+100
0x96000020          0x7fefffff 0xfffffff:Ma          0xfffffff 0xfffffff:Mi
    
```

(2) sdの表示フォーマット

● データ表示 (プリント) 時の表示フォーマット

0xaaaaaaaa(0x111111) dddddddd dddddddd dddddddd dddddddd '.....'
 アドレス オフセット 表示データ 文字コード

アドレス：表示データの先頭アドレスを16進で表示します。

オフセット：データ先頭からのオフセットを表示します。

表示データ：指定されたデータ出力形式、データ長に従い、アドレスの内容を表示します。

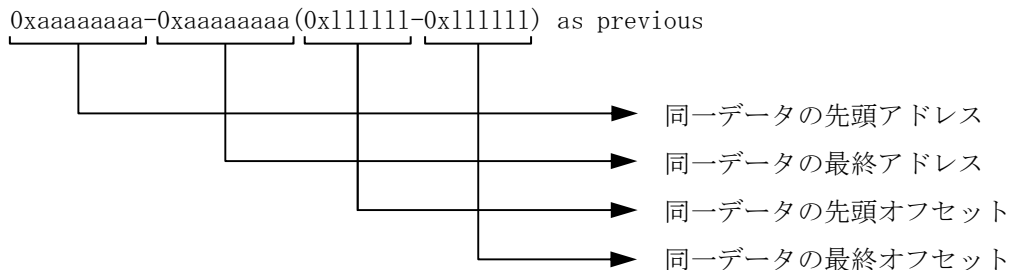
1行で最大16バイト分のデータを表示します。

データ出力形式が浮動小数点 (-f-l、-fd) のとき、アドレスの内容が下記データである場合は、16進数に変換して表示します。また、16進数表示のあとに対応する文字列を表示します。

浮動小数点データ	文字列	表示例	
		単精度	倍精度
非数	Na	0x7fffffff : Na	0xffff00000 0x00000001 : Na
無限大	In	0x7f800000 : In	0xffff00000 0x00000000 : In
表現できる最大値	Ma	0x7f7fffff : Ma	0x7fefffff 0xffffffff : Ma
表現できる最小値	Mi	0xff7fffff : Mi	0xffefffff 0xffffffff : Mi

文字コード：データ出力形式が16進 (-h) のときだけデータの内容を文字コードに置き換えて画面右部に表示します。置き換えできないデータは“.”で表示します。

直前に表示した行の内容と同一データが行単位で連続する場合、下記のメッセージを表示します (-allオプション指定時は連続するデータすべてを表示します)。



● データ変更 (パッチ) 時の表示フォーマット

0xaaaaaaaa(0x111111) dddddddd :
 アドレス オフセット 表示データ

● 表示例

データ出力形式とデータ長の両オプションの組み合わせによるsdの表示例を、下図に示します。

	*****	*****	*****	*****	*****	*****	*****	*****	*****				
16進4バイト 表示 (-h, -l)	0x00ec0000(0x000000)	0000000a	00000064	000003e8	00002710	‘.....d.....’	‘						
	0x00ec0010(0x000010)	000186a0				‘....	‘						
16進2バイト 表示 (-h, -w)	0x00ec0000(0x000000)	0000 000a	0000 0064	0000 03e8	0000 2710	‘.....d.....’	‘						
	0x00ec0010(0x000010)	0001 86a0				‘....	‘						
16進1バイト 表示 (-h, -b)	0x00ec0000(0x000010)	00 00 00 0a	00 00 00 64	00 00 03 e8	00 00 27 10	‘.....d							
	0x00ec0010(0x000010)	00 01 86 a0				‘....							
10進4バイト 表示 (-d, -l)	0x00ec0000(0x000000)		10	100	1000	10000							
	0x00ec0010(0x000010)		100000										
10進2バイト 表示 (-d, -w)	0x00ec0000(0x000000)	0	10	0	100	0	1000	0	10000				
	0x00ec0010(0x000010)	1	-31072										
10進1バイト 表示 (-d, -b)	0x00ec0000(0x000000)	0	0	0	10	0	0	0	100	0	0	3	-24
		0	0	39	16								
	0x00ec0010(0x000010)	0	1	-122	-96								
単精度実数 表示 (-f, -l)	0x00ec0020(0x000000)	1.1200000	2.1229999	10.1230001	20.1233997								
	0x00ec0030(0x000010)	100.123451											
倍精度実数 表示 (-fd)	0x96000000(0x000000)	1.0000000000000000E+00	2.0000000000000000E+00										
	0x96000010(0x000010)	-1.0000000000000000E+00	1.0000000000000000E+100										
	0x96000020(0x000020)	0x7fefffff	0xffffffff:Ma	0xffefffff	0xffffffff:Mi								

付録H ライブラリの使用するスタックサイズ一覧

ライブラリが使用するスタックサイズの一覧を以下に示します。

(1) C標準ライブラリのスタックサイズ一覧

No.	関数名	スタックサイズ	
		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	strncpy	0	
14	strlen	0	
15	strncat	4	
16	strncmp	4	
17	strncpy	0	
18	strpbrk	0	
19	strrchr	12	
20	strspn	0	
21	strtod	440	436

No.	関数名	スタックサイズ	
		libsh4nbmdn.lib	libsh4nbmzz.lib
22	strtol	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) libfhrad.libのスタックサイズ一覧

No.	関数名	スタックサイズ
1	irglbad	0
2	irsubad	0

(3) libcrs.libのスタックサイズ一覧

No.	関数名	スタックサイズ
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) libcpms.libのスタックサイズ一覧

No.	関数名	スタックサイズ
1	memcpy (*)	28

(*) ローダでロードしたプログラム、サブプログラムにおいて、memcpy()はC標準ライブラリではなく、CPMSライブラリのmemcpy()が使用されます。

このページは白紙です。