お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

RENESAS

ユーザーズ・マニュアル

RX850 Pro Ver.3.20

リアルタイム・オペレーティング・システム

インストレーション編

対象ツール

RX850 Pro Ver.3.20

資料番号　U17421JJ1V0UM00（第1版）
発行年月　April 2005　CP(K)

〔メ　モ〕

目次要約

MS-DOS, Windows, Windows NT，およびWindows XP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。

Green Hills Software, MULTIは米国Green Hills Software, Inc.の商標です。

PC/ATは米国IBM社の商標です。

UNIXはX/Openカンパニーリミテッドがライセンスしている米国ならびに他の国における登録商標です。

TRONは，The Real-time Operating system Nucleusの略称です。

ITRONは，Industrial TRONの略称です。

μITRONは，Micro Industrial TRONの略称です。

TRON, ITRON，およびμITRONは，コンピュータの仕様に対する名称であり，特定の商品ないし商品群を指すものではありません。

μITRON3.0仕様は，（社）トロン協会が策定したオープンなリアルタイムカーネル仕様です。

μITRON3.0仕様の仕様書は，「μITRON3.0標準ハンドブック改訂新版」（パーソナルメディア株式会社刊）として発行されています。

μITRON仕様の著作権は（社）トロン協会に属しています。

- 本資料に記載されている内容は2005年4月現在のもので，今後，予告なく変更することがあります。量産設計の際には最新の個別データ・シート等をご参照ください。
- 文書による当社の事前の承諾なしに本資料の転載複製を禁じます。当社は，本資料の誤りに関し，一切その責を負いません。
- 当社は，本資料に記載された当社製品の使用に関連し発生した第三者の特許権，著作権その他の知的財産権の侵害等に関し，一切その責を負いません。当社は，本資料に基づき当社または第三者の特許権，著作権その他の知的財産権を何ら許諾するものではありません。
- 本資料に記載された回路，ソフトウエアおよびこれらに関する情報は，半導体製品の動作例，応用例を説明するものです。お客様の機器の設計において，回路，ソフトウエアおよびこれらに関する情報を使用する場合には，お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し，当社は，一切その責を負いません。
- 当社は，当社製品の品質，信頼性の向上に努めておりますが，当社製品の不具合が完全に発生しないことを保証するものではありません。当社製品の不具合により生じた生命，身体および財産に対する損害の危険を最小限度にするために，冗長設計，延焼対策設計，誤動作防止設計等安全設計を行ってください。
- 当社は，当社製品の品質水準を「標準水準」，「特別水準」およびお客様に品質保証プログラムを指定していただく「特定水準」に分類しております。また，各品質水準は，以下に示す用途に製品が使われることを意図しておりますので，当社製品の品質水準をご確認ください。

  標準水準：コンピュータ，OA機器，通信機器，計測機器，AV機器，家電，工作機械，パーソナル機器，産業用ロボット

  特別水準：輸送機器（自動車，電車，船舶等），交通用信号機器，防災・防犯装置，各種安全装置，生命維持を目的として設計されていない医療機器

  特定水準：航空機器，航空宇宙機器，海底中継機器，原子力制御システム，生命維持のための医療機器，生命維持のための装置またはシステム等

  当社製品のデータ・シート，データ・ブック等の資料で特に品質水準の表示がない場合は，標準水準製品であることを表します。意図されていない用途で当社製品の使用をお客様が希望する場合には，事前に当社販売窓口までお問い合わせください。

(注)

（1）本事項において使用されている「当社」とは，NECエレクトロニクス株式会社およびNECエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいう。

（2）本事項において使用されている「当社製品」とは，（1）において定義された当社の開発，製造製品をいう。

M8E　02.11

〔メ　モ〕

はじめに

対象者　このマニュアルは，V850シリーズの応用システムを設計，開発するユーザを対象とします。

目　的　このマニュアルは，次の構　成に示すRX850 Proの機能をユーザに理解していただくことを目的としています。

構　成　このマニュアルは，大きく分けて次の内容で構成しています。

- ●概　説
- ●インストレーション
- ●システム構築
- ●インタフェース・ライブラリ
- ●メモリとその容量の見積もり
- ●システム・コンフィギュレーション・ファイル
- ●コンフィギュレータ CF850 Pro

読み方　このマニュアルの読者には，電気，論理回路，マイクロコンピュータ，C言語，アセンブリー言語に関する一般知識が必要です。

V850シリーズのハードウエア機能を知りたいとき
→各製品の**ユーザーズ・マニュアル　ハードウエア編**を参照してください。

V850シリーズの命令機能を知りたいとき
→V850ES　**ユーザーズ・マニュアル　アーキテクチャ編**（U15943J）または
　V850E1　**ユーザーズ・マニュアル　アーキテクチャ編**（U14559J）を参照してください。

凡　例　**注**　：本文中につけた注の説明
注意　：気をつけて読んでいただきたい内容
備考　：本文の補足説明
数の表記　：2進数 …XXXXまたはB’XXXX
10進数…XXXX
16進数…0xXXXまたはH’XXXX
2のべき数を示す接頭語（アドレス空間，メモリ容量）：
K（キロ）：$2^{10}=1024$
M（メガ）：$2^{20}=1024^2$
G（ギガ）：$2^{30}=1024^3$

関連資料　**このマニュアルを使用する場合は，次の資料もあわせてご覧ください。**
関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。
あらかじめご了承ください。

開発ツールに関する資料（ユーザーズ・マニュアル）

資　料　名		資料番号	
		和文	英文
CA850 Ver.3.00 **Cコンパイラ・パッケージ**	**操作編**	U17293J	U17293E
	C言語編	U17291J	U17291E
	アセンブリー言語編	U17292J	U17292E
	リンク・ディレクティブ編	U17294J	U17294E
ID850 Ver.3.00　**統合デバッガ**	**操作編**	U17358J	U17358E
ID850NW Ver.3.00, 3.10　**統合デバッガ**	**操作編**	U17369J	U17369E
ID850NWC Ver.2.51　**統合デバッガ**	**操作編**	U16525J	U16525E
ID850QB Ver.3.10　**統合デバッガ**	**操作編**	U17435J	U17435E
SM+　**システム・シミュレータ**	**操作編**	U17246J	U17246E
	ユーザ・オープン・インタフェース編	U17247J	U17247E
SM850 Ver.2.50　**システム・シミュレータ**	**操作編**	U16218J	U16218E
SM850 Ver.2.00**以上**　**システム・シミュレータ**	**外部部品ユーザ・オープン・インタフェース仕様編**	U14873J	U14873E
RX850 Pro Ver.3.20　**リアルタイム**OS	**基礎編**	U13773J	U13773E
	インストレーション編	**このマニュアル**	U17421E
	テクニカル編	U13772J	U13772E
	タスク・デバッガ編	U17422J	U17422E
RX-NET　**ネットワーク・ライブラリ**（TCP/IP）		U15083J	－
RX-NET　**ネットワーク・ライブラリ**（PPP）		U15303J	－
RX-NET　**ネットワーク・ライブラリ**（DNS）		U15304J	－
RX-NET　**ネットワーク・ライブラリ**（DHCP）		U15382J	－
RX-NET　**ネットワーク・ライブラリ**（SMTP）		U15505J	－
RX-NET　**ネットワーク・ライブラリ**（POP）		U15539J	－
RX-NET Ver.1.00 **ネットワーク・ライブラリ**（telnet）		U16085J	－
AZ850 Ver.3.30　**システム・パフォーマンス・アナライザ**		U17423J	U17423E
PG-FP4　**フラッシュ・メモリ・プログラマ**		U15260J	U15260E
TW850 Ver.2.00　**性能解析チューニング・ツール**		U17241J	U17241E
PM+ Ver.6.00　**プロジェクト・マネージャ**		U17178J	U17178E

目次

図目次

表目次

第 1 章　概説

1.1　概要

マイクロプロセッサは半導体技術の進歩にしたがって急速に普及し，今日では，あらゆる分野で利用されるようになってきました。しかし，マイクロプロセッサを取り巻く処理プログラム量は増大し，各種ハードウエアにあわせた固有のプログラムをそのつど作成することが困難となりました。

そこで，高性能，多機能化へと進むマイクロプロセッサの能力を完全に引き出すために，オペレーティング・システム (Operating System：OS) の重要性が高まってきました。

厳密な分け方ではありませんが，OS にはプログラム開発用と制御用の 2 種類があります。プログラム開発用の OS は，開発に使用するハードウエア構成をある程度固定 (パーソナル・コンピュータ等) できるため，標準的な OS (MS-DOS，Windows，UNIX 等) が流通しやすい環境にあります。

これに対し，制御用の OS は，制御機器に組み込まれて使用します。つまり，各々のシステムによってハードウエア構成が異なり，しかも，用途に応じた効率の良い動作が要求されるため，標準的な OS が流通しにくい環境にあります。

NEC エレクトロニクスでは，V850 シリーズを開発，発売し，より強力なマイクロプロセッサを提供する一方で，このような市場状況を考慮し，高機能なマイクロプロセッサが持つ機能を十分に引き出すため，また，将来にわたっての体系的なソフトウエアの構築を支援するために，RX850 Pro を開発，発売しました。

RX850 Pro は高性能，高機能なマイクロプロセッサの応用範囲を拡大し，いっそうの汎用性を持たせるために開発されたリアルタイム，マルチタスク処理を実現する制御用 OS です。

RX850 Pro は，効率の良いリアルタイム，マルチタスク処理環境を提供するとともに，対象プロセッサの制御機器分野における応用範囲を拡大することを目的として開発された，組み込み型制御用リアルタイム・マルチタスク OS です。

また，ターゲット・システムに組み込んで使用することを前提として開発されているため，ROM 化を意識し，高速かつコンパクトな OS となっています。

1.2　特徴

次に，RX850 Pro の特徴を示します。

1) μITRON3.0 仕様に準拠
RX850 Pro は，組み込み型制御用 OS のアーキテクチャとして代表的な μITRON3.0 仕様に準拠した設計が行われており，レベル E までの機能を実装しています。
なお，μITRON3.0 仕様とは，組み込み型制御用リアルタイム・システムのオペレーティング・システム仕様です。

2) 高い汎用性
RX850 Pro は，μITRON3.0 仕様で規定されているシステム・コールのほかに，RX850 Pro オリジナルのシステム・コールも提供し，アプリケーション・システムの汎用性を高めています。
なお，RX850 Pro では，アプリケーション・システムが使用する機能 (システム・コール) のみをコンフィギュレーション時に選択できるため，コンパクトでありながら，ユーザのニーズに最適なリアルタイム・マルチタスク OS を構築できます。

3) リアルタイム処理，マルチタスク処理の実現
完全なリアルタイム処理，マルチタスク処理を実現するために，次に示す機能を提供しています。

- タスク管理機能
- タスク付属同期機能
- 同期通信管理機能
- 割り込み処理管理機能
- メモリ・プール管理機能
- 時間管理機能
- システム管理機能
- スケジューリング機能

4) スケジューリングのロック機能
ディスパッチ処理 (タスクのスケジューリング処理) を禁止 / 再開する機能を提供しています。
これにより，ユーザは，処理プログラム・レベルからのディスパッチ処理の禁止 / 再開が可能となります。

5) ROM 化の実現
ターゲット・システムに組み込んで使用することを想定したリアルタイム・マルチタスク OS であるため，ROM 化を意識し，コンパクトに設計されています。

6) オリジナル命令の活用
V850 シリーズ マイクロプロセッサの高速な命令実行速度と，オリジナル命令の活用により，高速処理を実現しています。

7) 内蔵 ROM/RAM の活用
V850 シリーズの内蔵 ROM/RAM の活用により，高速な命令実行，高速なデータ・アクセスを実現しています。

8) アプリケーション・ユーティリティの提供
アプリケーション・システムを構築するうえで有益なユーティリティ・ツール，および，インタフェース・ライブラリを提供しています。

- コンフィギュレータ CF850 Pro
- タスク・デバッガ RD850 Pro
- システム・パフォーマンス・アナライザ AZ850
- 高級言語インタフェース・ライブラリ

注意　RX850 Pro 用タスク・デバッガを “ RD850 Pro ” と呼んでいます。

9) C コンパイラ・パッケージ
RX850 Pro は，次に示す V850 シリーズ用 C コンパイラ・パッケージに対応しています。

- CA850 (NEC エレクトロニクス製)
- CCV850/CCV850E (米国 Green Hills Software, Inc. 製)

1.3　実行環境

RX850 Pro は，組み込み型制御用の OS として開発されているため，次に示すハードウエアを備えたターゲット・システム上で動作します。

1) 動作対象 CPU
V851，V852，V853，V854，V850/SA1，V850/SBx，V850/SV1，
V850E/MS1，V850E/MA1，V850E/MA2，NB85E コア，V850E/IA1，V850E/IA2，

2) 周辺コントローラ
RX850 Pro では，様々な実行環境に対応するために，ニュークリアス内のハードウエア依存部を切り出し，サンプル・ソース・ファイルで提供しています。このため，サンプル・ソース・ファイルを各ターゲット・システム用に書き換えることで対応が可能です。そのため特定の周辺コントローラを要求しません。

1.4　開発環境

次に，システムを開発するうえで必要となるハードウエア環境とソフトウエア環境を示します。

1.4.1　ハードウエア環境

1) **ホスト・マシン**

- IBM PC/AT **互換機**

2) **インサーキット・エミュレータ**

- IE-703002-MC (V851，V852，V853，V854，V850/SA1，V850/SBx，V850/SV1)
- IE-703102-MC (V850E/MS1)
- IE-V850E-MC-A (V850E/MA1，V850E/MA2，NB85E **コア**)
- IE-V850E-MC (V850E/IA1，V850E/IA2)

1) **インサーキット・エミュレータ用** I/O **ボード**

- IE-703003-MC-EMI (V853)
- IE-703008-MC-EMI (V854)
- IE-703017-MC-EMI (V850/SA1)
- IE-703037-MC-EM1 (V850/SBx)
- IE-703040-MC-EM1 (V850/SV1)
- IE-703102-MC-EM1 (V850E/MS1 5V)
- IE-703102-MC-EM1-A (V850E/MS1 3.3V)
- IE-703107-MC-EM1 (V850E/MA1，V850E/MA2)
- IE-703116-MC-EM1 (V850E/IA1)
- IE-703114-MC-EM1 (V850E/IA2)
- IE-V850E-MC-EM1-A (NB85E **コア** 5V)
- IE-V850E-MC-EM1-B (NB85E **コア** 3.3V)

注意　これらの I/O **ボードは，インサーキット・エミュレータと組み合わせて使用します。**

2) PC **インタフェース・ボード**

- IE-70000-98-IF-C (PC-9800 **シリーズ** C **バス用**)
- IE-70000-PC-IF-C (IBM PC/AT **互換機** ISA **バス用**)
- IE-70000-CD-IF-A (PCMCIA **ソケット用**)
- IE-70000-PCI-IF (PCI **バス用**)

1.4.2　ソフトウエア環境

1) OS (**かっこ内はホスト・マシン**)

- Windows 98，Me，NT 4.0，2000，XP (IBM PC/AT **互換機**)

2) **クロス・ツール**

- CA850 (NEC **エレクトロニクス製**)
- CCV850/CCV850E (**米国** Green Hills Software, Inc. **製**)

3) **デバッガ**

- ID850 (NEC **エレクトロニクス製**)
- SM850 (NEC **エレクトロニクス製**)
- MULTI，MULTI2000 (**米国** Green Hills Software, Inc. **製**)

- PARTNER (**京都マイクロコンピュータ製**)

4) **タスク・デバッガ**

- RD850 Pro (NEC **エレクトロニクス製**)

5) **システム・パフォーマンス・アナライザ**

- AZ850 (NEC **エレクトロニクス製**)

第 2 章　インストレーション

この章では，RX850 Pro のインストール / アンインストールについて説明します。

2.1　インストール

ここでは，RX850 Pro をインストールする方法について説明します。なお，RX850 Pro をインストールし直す場合は，あらかじめアンインストールしてから行ってください。

RX850 Pro の提供媒体は，オブジェクト・リリース版，ソース・リリース版ともに，それぞれ CD-ROM (1 枚) です。また，RX850 Pro のパッケージには，RD850 Pro，AZ850，および，オンライン・ヘルプも含まれています。これらも同時にインストールすることができます。

次に，インストールの手順を示します。

1) Windows を起動します。

2) CD ドライブに CD-ROM を挿入します。そのあと，自動的にセットアップ・プログラムが起動されます。もし自動的にセットアップ・プログラムが起動されない場合は，エクスプローラを起動し，CD ドライブにある INSTALL.EXE をダブルクリックして起動してください。以降，モニタ画面に表示されるメッセージにしたがってインストール作業を実行します。

3) Windows の標準アプリケーション Explorer 等を用いて，RX850 Pro の提供媒体に格納されていたファイル群がホスト・マシンにインストールされたことを確認します。なお，各フォルダについては，「2.3　フォルダ構成」を参照してください。

2.2　アンインストール

ここでは，RX850 Pro をアンインストールする方法について説明します。

1) Windows を起動します。

2) コントロール・パネルの “ アプリケーションの追加と削除 ” を起動し，アンインストールしたい項目 (“ NEC EL RX850PRO NEC EL (オブジェクト版) ” 等) を選択してください。

2.3　フォルダ構成

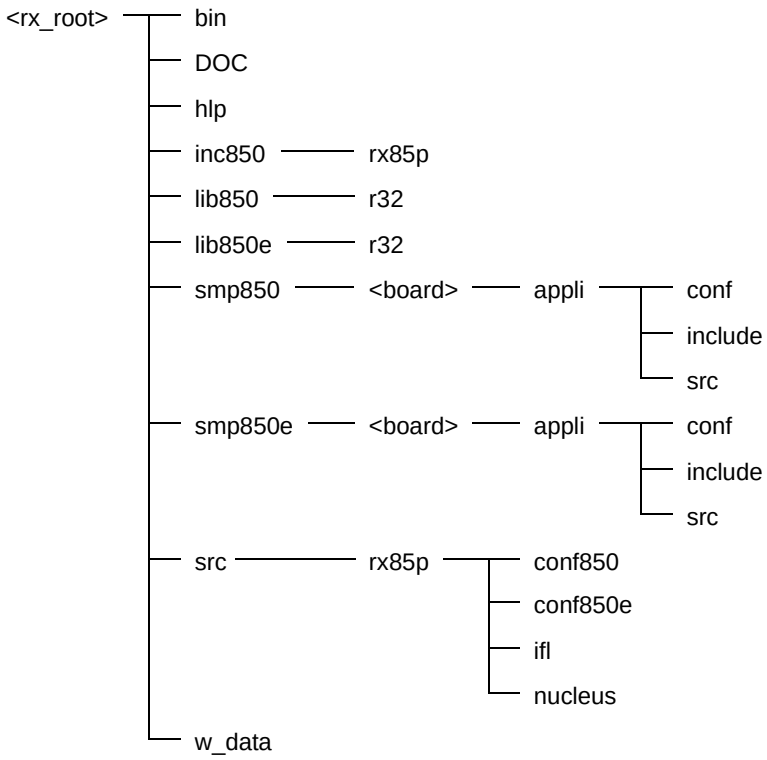
RX850 Pro をインストールした結果，提供媒体から読み込まれた各種ファイルのフォルダ構成について説明します。RX850 Pro の提供形態には，オブジェクト・リリース版とソース・リリース版があり，それぞれ CA850 対応版と GHS 製コンパイラ対応版があります。

- オブジェクト・リリース版 / CA850 対応版
- オブジェクト・リリース版 / GHS 製コンパイラ対応版
- ソース・リリース版 / CA850 対応版
- ソース・リリース版 / GHS 製コンパイラ対応版

2.3.1　オブジェクト・リリース版 / CA850 対応版

図 2-1 に，RX850 Pro の提供媒体に格納されているファイル群 (オブジェクト・リリース版 /CA850 対応版) をホスト・マシン上にインストールした際に生成されるフォルダ構成を示します。

図 2-1　フォルダ構成 (オブジェクト・リリース版 /CA850 対応版)



以下に，各フォルダの詳細を示します。

1) <rx_root>
インストール時に指定した「RX850 Pro のインストール・フォルダ」です。

2) <rx_root>\bin
RX850 Pro のユーティリティ・ツールが格納されているフォルダです。

cf850pro.exe　：コンフィギュレータ CF850 Pro
rx703100p.dll　：CF850 Pro 用 DLL ファイル

3) <rx_root>\DOC
RX850 Pro のドキュメント・ファイルが格納されているフォルダです。

4) <rx_root>\hlp
RX850 Pro のヘルプ・ファイルが格納されているフォルダです。

5) <rx_root>\inc850
RX850 Pro の標準ヘッダ・ファイルが格納されているフォルダです。

stdrx85p.h	： 標準ヘッダ・ファイル (C 言語用)
stdrx85p.inc	： 標準ヘッダファイル (アセンブリー言語用)

6) <rx_root>\inc850\rx85p
RX850 Pro のヘッダ・ファイルが格納されているフォルダです。

7) <rx_root>\lib850\r32
RX850 Pro のライブラリ・ファイル (V850 コア，32 レジスタ・モード) が格納されているフォルダです。

librxp.a	： ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要あり)
librxpm.a	： ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要なし)
libchp.a	： インタフェース・ライブラリ (パラメータ・チェック機能あり)
libncp.a	： インタフェース・ライブラリ (パラメータ・チェック機能なし)
libdbp.a	： インタフェース・ライブラリ (パラメータ・チェック機能あり，デバッグ用システム・コールを含む)
rxcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可)
rxtmcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可)
rxdbcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可，デバッグ用システム・コールを含む)
rxdbtmcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可，デバッグ用システム・コールを含む)

8) <rx_root>\lib850e\r32
RX850 Pro のライブラリ・ファイル (V850E1/V850E2/V850ES コア，32 レジスタ・モード) が格納されているフォルダです。

librxp.a	： ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要あり)
librxpm.a	： ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要なし)
libchp.a	： インタフェース・ライブラリ (パラメータ・チェック機能あり)
libncp.a	： インタフェース・ライブラリ (パラメータ・チェック機能なし)
libdbp.a	： インタフェース・ライブラリ (パラメータ・チェック機能あり，デバッグ用システム・コールを含む)
rxcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可)
rxtmcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可)
rxdbcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可，デバッグ用システム・コールを含む)
rxdbtmcore.o	： ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可，デバッグ用システム・コールを含む)

9) <rx_root>\smp850\<board>
RX850 Pro のサンプル・プログラム (V850 コア) が格納されているフォルダです。

10) <rx_root>\smp850\<board>\appli\conf
RX850 Pro のロード・モジュールを生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ロード・モジュール sample.out が本フォルダに生成されます。

sample.prj	： ロード・モジュール用プロジェクト・ファイル

sample.prw　：ロード・モジュール用ワーク・スペース・ファイル

11) <rx_root>\smp850\<board>\appli\include
サンプル・プログラムのヘッダ・ファイルが格納されているフォルダです。

12) <rx_root>\smp850\<board>\appli\src
サンプル・プログラムのソース・ファイル，および，リンク・ディレクティブ・ファイルが格納されているフォルダです。

13) <rx_root>\smp850e\<board>
RX850 Pro **のサンプル・プログラム** (V850E1/V850E2/V850ES **コア**) **が格納されているフォルダです。**

14) <rx_root>\smp850e\<board>\appli\conf
RX850 Pro **のロード・モジュールを生成するためのコマンド・ファイルが格納されているフォルダです。**
なお，本フォルダのコマンド・ファイルを用いることにより，ロード・モジュール sample.out **が本フォルダに生成されます。**

sample.prj　：ロード・モジュール用プロジェクト・ファイル
sample.prw　：ロード・モジュール用ワーク・スペース・ファイル

15) <rx_root>\smp850e\<board>\appli\include
サンプル・プログラムのヘッダ・ファイルが格納されているフォルダです。

16) <rx_root>\smp850e\<board>\appli\src
サンプル・プログラムのソース・ファイル，および，リンク・ディレクティブ・ファイルが格納されているフォルダです。

17) <rx_root>\src\rx85p\conf850
インタフェース・ライブラリ (V850 **コア，**32 **レジスタ・モード**) **を生成するためのコマンド・ファイルが格納されているフォルダです。**
なお，本フォルダのコマンド・ファイルを用いることにより，インタフェース・ライブラリが <rx_root>\lib850\r32 **に生成されます。**

libchp.prj　：インタフェース・ライブラリ libchp.a 用プロジェクト・ファイル
libncp.prj　：インタフェース・ライブラリ libncp.a 用プロジェクト・ファイル
library.prw　：インタフェース・ライブラリ libchp.a，libncp.a 用ワーク・スペース・ファイル

18) <rx_root>\src\rx85p\conf850e
インタフェース・ライブラリ (V850E1/V850E2/V850ES **コア，**32 **レジスタ・モード**) **を生成するためのコマンド・ファイルが格納されているフォルダです。**
なお，本フォルダのコマンド・ファイルを用いることにより，インタフェース・ライブラリが <rx_root>\lib850e\r32 **に生成されます。**

libchp.prj　：インタフェース・ライブラリ libchp.a 用プロジェクト・ファイル
libncp.prj　：インタフェース・ライブラリ libncp.a 用プロジェクト・ファイル
library.prw　：インタフェース・ライブラリ libchp.a，libncp.a 用ワーク・スペース・ファイル

19) <rx_root>\src\rx85p\ifl
インタフェース・ライブラリ libchp.a，libncp.a，libdbp.a **のソース・ファイルが格納されているフォルダです。**

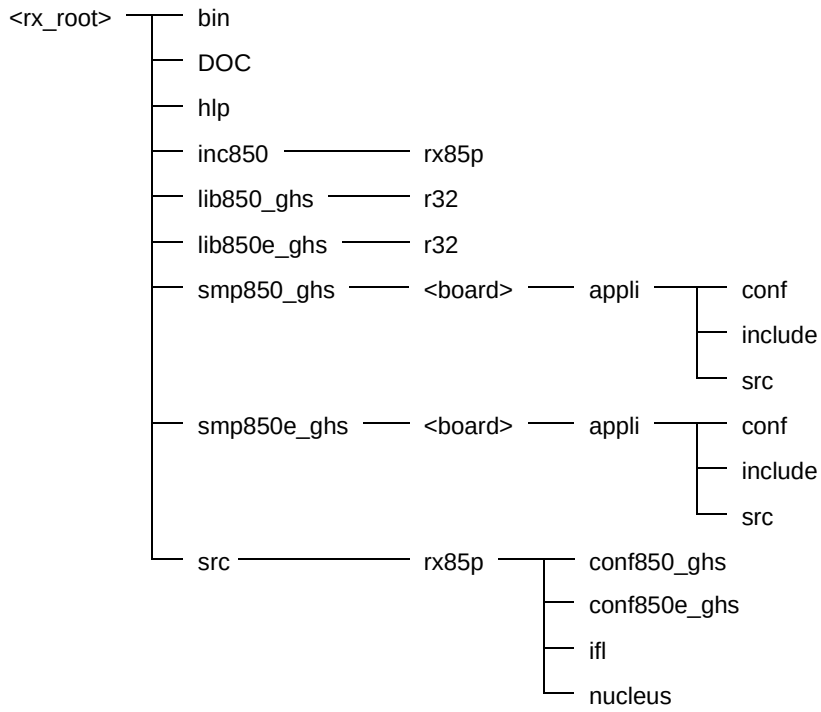
20) <rx_root>\src\rx85p\nucleus
AZ850 **のトレース機能を有効にするためのソース・ファイルが格納されているフォルダです。**

21) <rx_root>\w_data
PM+ **用の** RX850 Pro **を使用した場合のサンプル・リンク・ディレクティブ・ファイルが格納されているフォルダです。**

2.3.2　オブジェクト・リリース版 / GHS 製コンパイラ対応版

図 2-2 に，RX850 Pro の提供媒体に格納されているファイル群 (オブジェクト・リリース版 /GHS 製コンパイラ対応版) をホスト・マシン上にインストールした際に生成されるフォルダ構成を示します。

図 2-2　フォルダ構成 (オブジェクト・リリース版 /GHS 製コンパイラ対応版)



以下に，各フォルダの詳細を示します。

1) <rx_root>
インストール時に指定した「RX850 Pro のインストール・フォルダ」です。

2) <rx_root>\bin
RX850 Pro のユーティリティ・ツールが格納されているフォルダです。

cf850pro_ghs.exe：コンフィギュレータ CF850 Pro

3) <rx_root>\DOC
RX850 Pro のドキュメント・ファイルが格納されているフォルダです。

4) <rx_root>\hlp
RX850 Pro のヘルプ・ファイルが格納されているフォルダです。

5) <rx_root>\inc850
RX850 Pro の標準ヘッダ・ファイルが格納されているフォルダです。

stdrx85p.h　　：標準ヘッダ・ファイル

6) <rx_root>\inc850\rx85p
RX850 Pro のヘッダ・ファイルが格納されているフォルダです。

7) <rx_root>\lib850_ghs\r32
RX850 Pro のライブラリ・ファイル (V850 コア，32 レジスタ・モード) が格納されているフォルダです。

librxp.a　　：ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要あり)

librxpm.a　：ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要なし)
libchp.a　：インタフェース・ライブラリ (パラメータ・チェック機能あり)
libncp.a　：インタフェース・ライブラリ (パラメータ・チェック機能なし)
libdbp.a　：インタフェース・ライブラリ (パラメータ・チェック機能あり，デバッグ用システム・コールを含む)
rxcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可)
rxtmcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可)
rxdbcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可，デバッグ用システム・コールを含む)
rxdbtmcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可，デバッグ用システム・コールを含む)

8) <rx_root>\lib850e_ghs\r32
RX850 Pro のライブラリ・ファイル (V850E1/V850E2/V850ES コア , 32 レジスタ・モード) が格納されているフォルダです。

librxp.a　：ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要あり)
librxpm.a　：ニュークリアス・ライブラリ (rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要なし)
libchp.a　：インタフェース・ライブラリ (パラメータ・チェック機能あり)
libncp.a　：インタフェース・ライブラリ (パラメータ・チェック機能なし)
libdbp.a　：インタフェース・ライブラリ (パラメータ・チェック機能あり，デバッグ用システム・コールを含む)
rxcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可)
rxtmcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可)
rxdbcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可，デバッグ用システム・コールを含む)
rxdbtmcore.o　：ニュークリアス共通オブジェクト (周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可，デバッグ用システム・コールを含む)

9) <rx_root>\smp850_ghs\<board>
RX850 Pro のサンプル・プログラム (V850 コア) が格納されているフォルダです。

10) <rx_root>\smp850_ghs\<board>\appli\conf
RX850 Pro のロード・モジュールを生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ロード・モジュール sample.out が本フォルダに生成されます。

sample.bld　：ロード・モジュール用ビルド・ファイル

11) <rx_root>\smp850_ghs\<board>\appli\include
サンプル・プログラムのヘッダ・ファイルが格納されているフォルダです。

12) <rx_root>\smp850_ghs\<board>\appli\src
サンプル・プログラムのソース・ファイル，および，リンク・ディレクティブ・ファイルが格納されているフォルダです。

13) <rx_root>\smp850e_ghs\<board>
RX850 Pro のサンプル・プログラム (V850E1/V850E2/V850ES コア) が格納されているフォルダです。

14) <rx_root>\smp850e_ghs\<board>\appli\conf
RX850 Pro のロード・モジュールを生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ロード・モジュール sample.out が本フォルダに生成されます。

sample.bld　：ロード・モジュール用ビルド・ファイル

15) <rx_root>\smp850e_ghs\<board>\appli\include
サンプル・プログラムのヘッダ・ファイルが格納されているフォルダです。

16) <rx_root>\smp850e_ghs\<board>\appli\src
サンプル・プログラムのソース・ファイル，および，リンク・ディレクティブ・ファイルが格納されているフォルダです。

17) <rx_root>\src\rx85p\conf850_ghs
インタフェース・ライブラリ (V850 **コア，**32 **レジスタ・モード**) **を生成するためのコマンド・ファイルが格納されているフォルダです。**
なお，本フォルダのコマンド・ファイルを用いることにより，インタフェース・ライブラリが <rx_root>\lib850_ghs\r32 **に生成されます。**

library.bld　　**：インタフェース・ライブラリ** libchp.a**，**libncp.a **用ビルド・ファイル**

18) <rx_root>\src\rx85p\conf850e_ghs
インタフェース・ライブラリ (V850E1/V850E2/V850ES **コア，**32 **レジスタ・モード**) **を生成するためのコマンド・ファイルが格納されているフォルダです。**
なお，本フォルダのコマンド・ファイルを用いることにより，インタフェース・ライブラリが <rx_root>\lib850e_ghs\r32 **に生成されます。**

library.bld　　**：インタフェース・ライブラリ** libchp.a**，**libncp.a **用ビルド・ファイル**

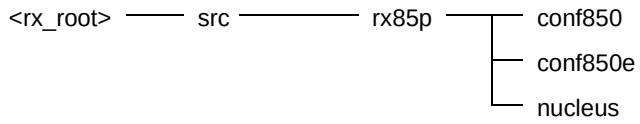
19) <rx_root>\src\rx85p\ifl
インタフェース・ライブラリ libchp.a**，**libncp.a**，**libdbp.a **のソース・ファイルが格納されているフォルダです。**

20) <rx_root>\src\rx85p\nucleus
AZ850 **のトレース機能を有効にするためのソース・ファイルが格納されているフォルダです。**

2.3.3　ソース・リリース版 / CA850 対応版

図 2-3 に，RX850 Pro の提供媒体に格納されているファイル群 (ソース・リリース版 /CA850 対応版) をホスト・マシン上にインストールした際に生成されるフォルダ構成を示します。

図 2-3　フォルダ構成 (ソース・リリース版 /CA850 対応版)



以下に，各フォルダの詳細を示します。

1) <rx_root>
インストール時に指定した「RX850 Pro のインストール・フォルダ」です。

2) <rx_root>\src\rx85p\conf850
ライブラリ・ファイル (V850 コア，32 レジスタ・モード) を生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ライブラリ・ファイルが <rx_root>\lib850\r32 に生成されます。

makefile　：ライブラリ・ファイル librxp.a，librxpm.a，libchp.a，libncp.a，libdbp.a，rxcore.o，rxtmcore.o，rxdbcore.o，rxdbtmcore.o 用メイク・ファイル

3) <rx_root>\src\rx85p\conf850e
ライブラリ・ファイル (V850E1/V850E2/V850ES コア，32 レジスタ・モード) を生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ライブラリ・ファイルが <rx_root>\lib850e\r32 に生成されます。

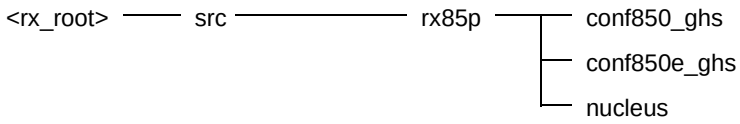
makefile　：ライブラリ・ファイル librxp.a，librxpm.a，libchp.a，libncp.a，libdbp.a，rxcore.o，rxtmcore.o，rxdbcore.o，rxdbtmcore.o 用メイク・ファイル

4) <rx_root>\src\rx85p\nucleus
ライブラリ・ファイル librxp.a，librxpm.a，rxcore.o，rxtmcore.o，rxdbcore.o，rxdbtmcore.o のソース・ファイルが格納されているフォルダです。

2.3.4　ソース・リリース版 / GHS 製コンパイラ対応版

図 2-4 に，RX850 Pro の提供媒体に格納されているファイル群 (ソース・リリース版 /GHS 製コンパイラ対応版) をホスト・マシン上にインストールした際に生成されるフォルダ構成を示します。

図 2-4　フォルダ構成 (ソース・リリース版 / GHS 製コンパイラ対応版)



以下に，各フォルダの詳細を示します。

1) <rx_root>
インストール時に指定した「RX850 Pro のインストール・フォルダ」です。

2) <rx_root>\src\rx85p\conf850_ghs
ライブラリ・ファイル (V850 コア，32 レジスタ・モード) を生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ライブラリ・ファイルが <rx_root>\lib850_ghs\r32 に生成されます。

nucleus.bld　：ライブラリ・ファイル librxp.a，librxpm.a，libchp.a，libncp.a，libdbp.a，rxcore.o，rxtmcore.o，rxdbcore.o，rxdbtmcore.o 用ビルド・ファイル

3) <rx_root>\src\rx85p\conf850e_ghs
ライブラリ・ファイル (V850E1/V850E2/V850ES コア，32 レジスタ・モード) を生成するためのコマンド・ファイルが格納されているフォルダです。
なお，本フォルダのコマンド・ファイルを用いることにより，ライブラリ・ファイルが <rx_root>\lib850e_ghs\r32 に生成されます。

nucleus.bld　：ライブラリ・ファイル librxp.a，librxpm.a，libchp.a，libncp.a，libdbp.a，rxcore.o，rxtmcore.o，rxdbcore.o，rxdbtmcore.o 用ビルド・ファイル

4) <rx_root>\src\rx85p\nucleus
ライブラリ・ファイル librxp.a，librxpm.a，rxcore.o，rxtmcore.o，rxdbcore.o，rxdbtmcore.o のソース・ファイルが格納されているフォルダです。

第 3 章　システム構築

この章では，RX850 Pro を使用したアプリケーション・システムの構築手順について説明します。

3.1　概要

システム構築とは，RX850 Pro の提供媒体からユーザの開発環境（ホスト・マシン）上に転送したファイル群を用いて，ロード・モジュールを作成したあと，ターゲット・システムへ組み込むことです。
次に，システムを構築する際の手順を示します。

1）**システム・コンフィギュレーション・ファイルの作成**

2）**情報ファイルの生成**

- システム情報テーブル・ファイル
- システム・コール・テーブル・ファイル
- システム情報ヘッダ・ファイル

備考　これらの情報テーブルは，CF850 Pro を利用して作成します。

3）**システム初期化部の作成**

- **ブート処理**
- **ハードウエア初期化部**
- **ソフトウエア初期化部**
- **割り込みエントリ**

4）**処理プログラムの作成**

- タスク
- 割り込みハンドラ
- 周期起動ハンドラ
- 拡張 SVC ハンドラ
- 拡張 SVC ハンドラ用インタフェース・ライブラリ

備考　処理プログラムは，C 言語やアセンブリー言語を用いて作成します。

5）**初期化データの退避領域の作成** (CA850 使用時のみ)

6）**リンク・ディレクティブ・ファイルの作成**

7）**ロード・モジュールの作成**

8）**システムへの組み込み**

図 3-1 に，CA850 対応版のシステム構築手順の例を，図 3-2 に，GHS 製コンパイラ対応版のシステム構築手順の例をそれぞれ示します。

図 3-1　システム構築手順 (CA850 対応版)

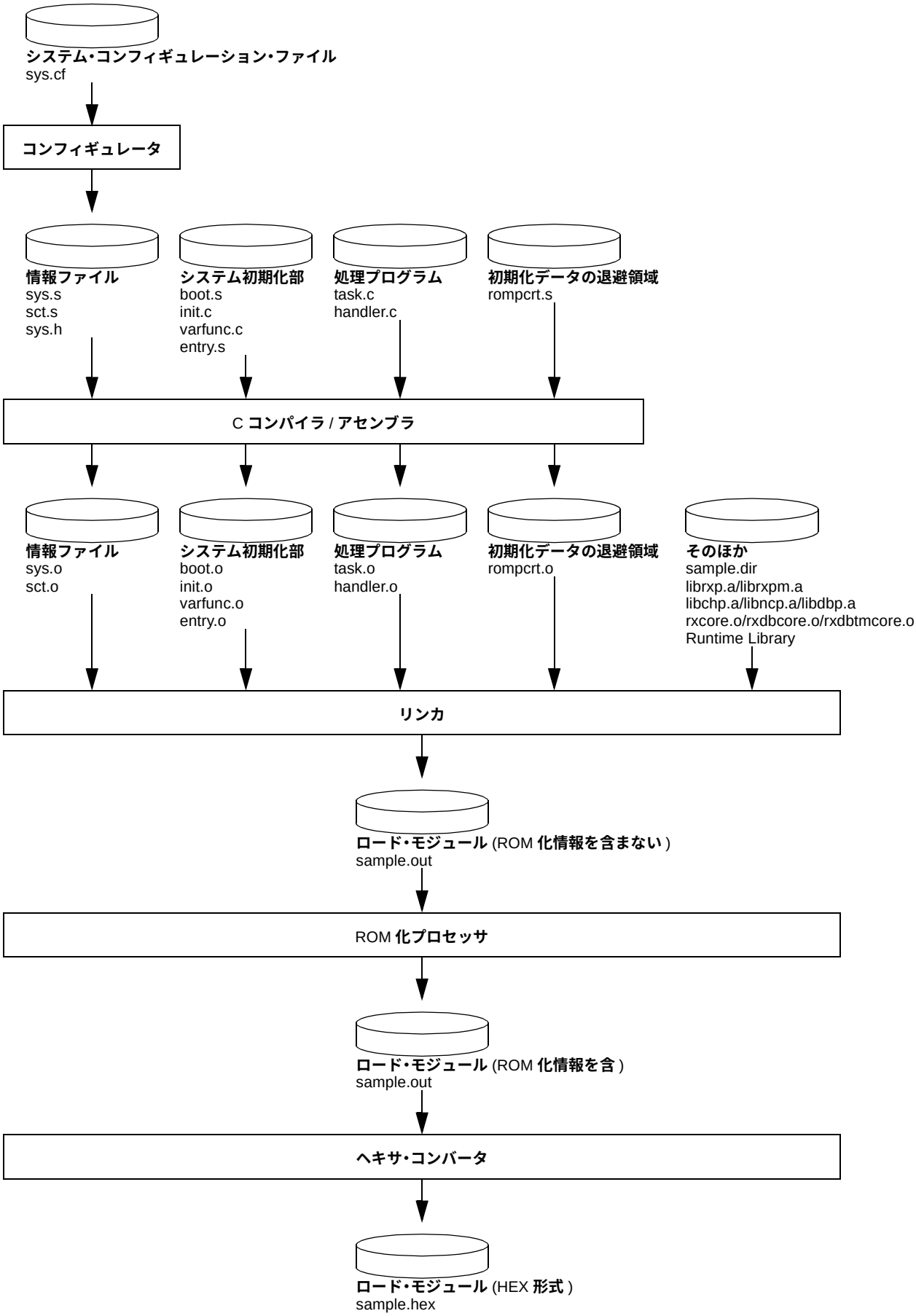
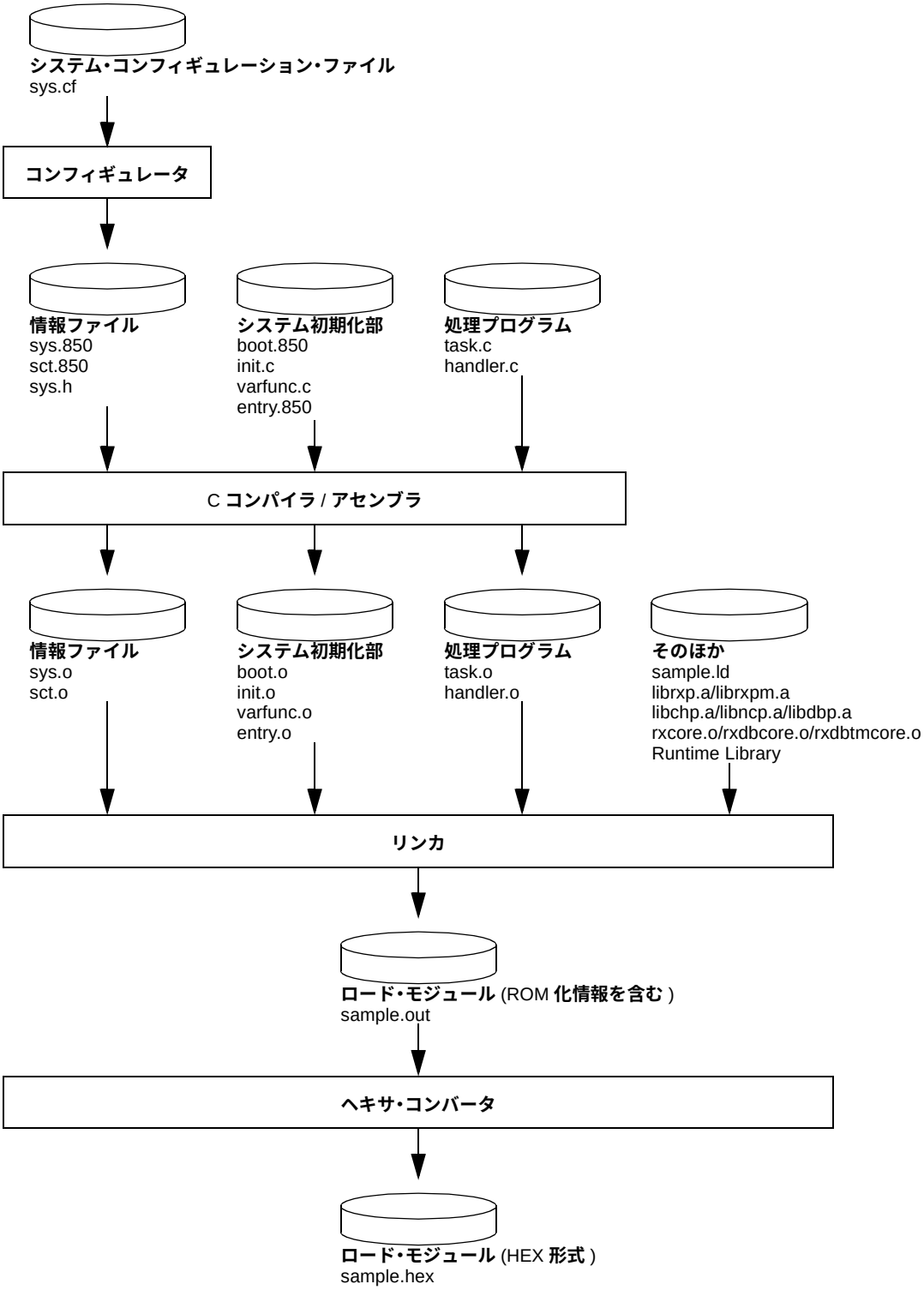


図 3-2　システム構築手順 (GHS 製コンパイラ対応版)



次に，パッケージに添付されているサンプル・プログラムを基に，システム構築の流れを説明します。
なお，サンプル・プログラムの格納フォルダは，RX850 Pro のインストール・フォルダが <rx_root> とした場合，次のようになります。

表 3-1　サンプル・プログラムの格納フォルダ

コンパイラ種別 / 対象 CPU	格納フォルダ
CA850 / V850 コア	<rx_root>\smp850\rx85p\src
CA850 / V850E1/V850E2/V850ES コア	<rx_root>\smp850e\rx85p\src
GHS 製コンパイラ / V850 コア	<rx_root>\smp850_ghs\rx85p\src
GHS 製コンパイラ / V850E1/V850E2/V850ES コア	<rx_root>\smp850e_ghs\rx85p\src

それぞれご使用のコンパイラ種別，対象 CPU にあわせて参照してください。

3.2　システム・コンフィギュレーション・ファイルの作成

RX850 Pro で使用する各種データを保持した「システム・コンフィギュレーション・ファイル」という情報テーブルを作成します。
このファイルは，CF850 Pro を使用し，次のものを作成するために必要となります。

- システム情報テーブル・ファイル
- システム・コール・テーブル・ファイル
- システム情報ヘッダ・ファイル

作成方法については，「3.2.1　情報ファイルの生成」を参照してください。
システム情報テーブル・ファイルは，タスク，セマフォ，メモリ・プール等の RX850 Pro の資源情報，システム・コール・テーブル・ファイルは，アプリケーションで使用しているシステム・コールの一覧を保持するテーブルです。システム情報ヘッダ・ファイルは，システム情報テーブル・ファイルで作成したタスク，セマフォ等，資源 ID として指定されたシンボル名と実際の ID 番号を，#define 命令で対応させる記述があります。
サンプルのシステム・コンフィギュレーション・ファイルは

- sys.cf

です。システム・コンフィギュレーション・ファイルの内容と書式については，「第 6 章　システム・コンフィギュレーション・ファイル」を参照してください。

3.2.1　情報ファイルの生成

「3.2　システム・コンフィギュレーション・ファイルの作成」で生成したシステム・コンフィギュレーション・ファイルから，CF850 Pro を使用してシステム情報テーブル・ファイル，システム・コール・テーブル・ファイル，システム情報ヘッダ・ファイルを作成します。
それぞれ，次のファイル名を推奨しています。

【 システム情報テーブル・ファイル 】

- CA850 対応版：sys.s
- GHS 製コンパイラ対応版：sys.850

【 システム・コール・テーブル・ファイル 】

- CA850 対応版：sct.s
- GHS 製コンパイラ対応版：sct.850

【 システム情報ヘッダ・ファイル 】

- sys.h

RX850 Pro を使用したアプリケーションを構築する際は，sys.s (，sys.850)，sct.s (，sct.850) をアセンブルして生成したオブジェクトをリンクし，C 言語ソース・ファイルでは，sys.h をインクルードする必要があります。
これらのファイルを作るための CF850 Pro の使用方法については，「第 7 章　コンフィギュレータ CF850 Pro」を参照してください。

3.3　システム初期化部の作成

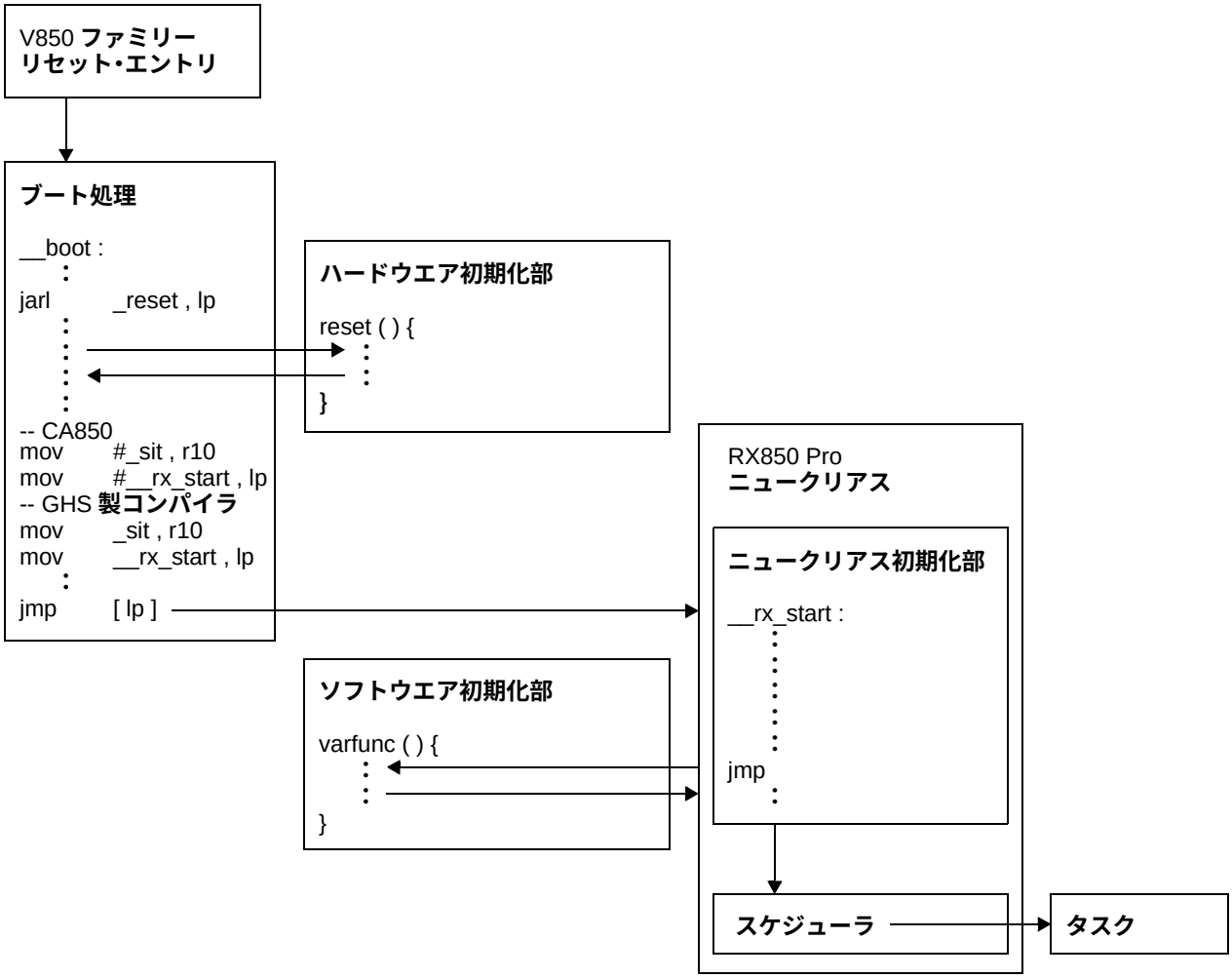
システム初期化部とは，ユーザのターゲット・システムに依存する部分を切り出し，移植性の向上をはかるとともに，カスタマイズを容易にするために用意された関数です。
システム初期化部の構成は，次のとおりです。

表 3-2　システム初期化部の構成

サンプル・ファイル名	種別	関数名	機能
boot.s (CA850 対応版) boot.850 (GHS 製コンパイラ対応版)	ブート処理	_boot	システムのブート処理
init.c	ハードウエア初期化部	reset	ハードウエアの初期化処理
varfunc.c	ソフトウエア初期化部	varfunc	ソフトウエアの初期化処理
entry.s (CA850 対応版) entry.850 (GHS 製コンパイラ対応版)	割り込みエントリ	無し	割り込み処理への分岐処理

また，システム初期化部のおおまかな流れは次のようになります。

図 3-3　システム初期化部の流れ



3.3.1　ブート処理

ブート処理は，V850 シリーズのリセット・エントリ (ハンドラ・アドレス：0x0 番地) に割りつけられ，システム初期化処理の中で最初に実行されます。
サンプルでは，boot.s (，boot.850) 内の _boot というラベル以降が，ブート処理の本体になります。リセット・エントリからこのラベルにジャンプさせている部分は，entry.s (，entry.850) 内にある次の命令に当たります。

【 CA850 対応版 】

```
.section        "RESET"
.extern         __boot

mov             #__boot , lp
jmp             [ lp ]
```

【 GHS 製コンパイラ対応版 】

```
.org            0x00000000
.extern         __boot

mov             __boot , lp
jmp             [ lp ]
```

それぞれ下 2 行分が，ハンドラ・アドレス「0x0 番地」に割りつけられます。つまり，リセットされると，この命令が実行され，_boot へジャンプしてブート処理が実行されます。
ブート処理内では，次のことをする必要があります。

1) tp (テキスト・ポインタ)，gp (グローバル・ポインタ)，ep (エレメント・ポインタ) の設定
2) ブート処理内で使う sp (スタック・ポインタ) の設定
3) r10 へ _sit シンボルのアドレスを設定
4) __rx_start シンボルをレジスタにセットして jmp 命令を発行し，RX850 Pro ニュークリアス初期化部へ制御移行

このほか，サンプルでは，ハードウエア初期化部 reset にジャンプする処理 (jarl _reset , lp) が 2) と 3) の間にあります。
2）で設定するスタック・ポインタは，タスクや割り込みハンドラのスタックとは無関係です。RX850 Pro 起動後は，タスクや割り込みハンドラ等が使用するスタックは，システム情報テーブル・ファイルにより RX850 Pro 自身が管理しており，タスク・スイッチや割り込みにより自動的にスタック・ポインタを切り替えます。
つまり，ブート処理内で指定するスタック・ポインタは RX850 Pro 起動前に使用するもので，たとえば関数にジャンプし，その関数内でスタックに保存すべきデータがある場合等に使用されます。サンプルの reset 関数でスタックを使用する必要があるときには，このスタック・ポインタが使用されます。
また，サンプルでは，スタック・サイズとして 0x28000 バイトを取っていますが，これほど必要ないことがほとんどです。ただ，この領域は RX850 Pro で使用されるシステム・メモリ領域システム・メモリ情報で定義されたサイズにあわせてあり，RX850 Pro 起動後は，その領域として使用されるようになっています。
また，CA850 対応版のサンプルではブート処理内で RAM 上の bss 領域の初期化 (0 クリア) が行われています。GHS 製コンパイラ対応版のサンプルでは，ソフトウエア初期化処理 varfunc.c 内で bss 領域の初期化処理，および，初期値データのコピー処理を行っています。
なお，CA850 対応版の「初期値データのコピー処理」は，初期値データの退避領域の作成 rompcrt.s と _rcopy 関数の使用によって行われます。詳細については，「CA850 ユーザーズ・マニュアル 操作編」を参照してください。
そしてブート処理の最後では，3）と 4）の処理が必要になります。次の処理を行ってください。

【 CA850 対応版 】

```
.extern         _sit
mov             #_sit , r10

.extern         __rx_start
mov             #__rx_start , lp
jmp             [ lp ]
```

【 GHS 製コンパイラ対応版 】

```
.extern     _sit
mov         _sit , r10

.extern     __rx_start
mov         __rx_start , lp
jmp         [ lp ]
```

RX850 Pro 内部にあるシンボル __rx_start 以降が RX850 Pro のニュークリアス初期化処理に当たります。ブート処理が完了した際は jmp 命令によってニュークリアス初期化処理に移行します。その際に r10 レジスタに _sit シンボルのアドレスを代入してから移行してください。初期化処理では，この r10 レジスタに代入されたアドレスと，システム・コンフィギュレーション・ファイルから作ったシステム情報テーブル・ファイルを基に，資源生成や初期化を行っているためです。

ブート処理の記述は，サンプルのブート処理を基に，ユーザにあわせた環境へ変更していくことを推奨します。

3.3.2　ハードウエア初期化部

ハードウエア初期化部は，サンプルでは，ブート処理から呼び出される関数です。ブート処理の一環としてターゲット・システム上のハードウエアの初期化を行うために用意しているものです。

サンプルでは，reset 関数が，このハードウエア初期化処理を行っています。ブート処理から，この reset 関数を呼び出していますが，ハードウエア初期化処理が特に必要ない場合，または，ほかで行うという場合は，この関数を使用しなくても問題ありません。

サンプルのハードウエア初期化部では，次のような処理を行っています。

1) 割り込みコントローラ，タイマ・コントローラの初期化
2) 周辺 I/O レジスタ，コントローラの初期化
3) ブート処理へ制御を戻す

3.3.3　ニュークリアス初期化部

ニュークリアス初期化部は，ブート処理終了後に実行される RX850 Pro 内部のルーチンです。ここでは，システム・コンフィギュレーション・ファイルより作成されたシステム情報テーブル・ファイルより，RX850 Pro のシステム管理ブロックの作成，タスクやセマフォ，メモリ・プール等の情報を作成，初期化を行っています。

RX850 Pro では，このニュークリアス初期化部において，初期化が正常に行われなかった場合，CPU を HALT 状態にします。ブート処理から初期化処理にジャンプしたあと，RX850 Pro が起動せずに HALT 状態になる場合は，RX850 Pro の管理ブロック作成等に使用するシステム・メモリ領域「システム・メモリ情報」が不足していることが考えられますので，メモリが十分に確保されているか確認してください。

ニュークリアス初期化部において初期化が終了すると，ソフトウエア初期化部が呼び出されます。これは，「初期化ハンドラ情報」で指定されたもので，サンプルでは，varfunc 関数となっています。この関数については，「3.3.4　ソフトウエア初期化部」を参照してください。

ソフトウエア初期化部から制御が戻ると，スケジューラが起動され，RX850 Pro が起動します。

3.3.4　ソフトウエア初期化部

ソフトウエア初期化部は，ニュークリアス初期化部から呼び出される関数です。ここでは，RX850 Pro 起動直前にやっておきたい処理を記述します。

ソフトウエア初期化部は，初期化ハンドラ情報で指定された関数を呼び出します。サンプルでは，varfunc 関数となっています。この関数は必ず指定する必要がありますので，特に必要ない場合でも，何もしない関数として生成してください。また，ハンドラの最後では，return 命令によってニュークリアス初期化処理へ戻してください。

CA850 対応版のサンプルでは何もしない関数として定義されていますが，GHS 製コンパイラ対応版のサンプルでは RAM 上の bss 領域の初期化 (0 クリア) と初期値データのコピー処理を行っています。このルーチンは GHS 製コンパイラで推奨されている方法をそのまま使用していますので，詳細については GHS 製コンパイラのマニュアル等を参照してください。なお，CA850 対応版で初期値データのコピー処理を，このソフトウエア初期化部で行うことも可能です。初期値データのコピー処理については，「CA850 ユーザーズ・マニュアル 操作編」を参照してください。

3.3.5　割り込みエントリ

割り込みエントリは，割り込みが発生した際に実行される命令を記述したもので，V850 シリーズが持つ割り込みハンドラ・アドレスに割りつけられます。ユーザが使用するすべての割り込みに対し，割り込みエントリを定義する必要があります。これらはアセンブリー言語で記述する必要があり，サンプルでは，CA850 対応版は entry.s に，GHS 製コンパイラ対応版は entry.850 に記述されています。

RX850 Pro の割り込み処理には，直接起動割り込みハンドラ，間接起動割り込みハンドラの 2 種類があり，それぞれエントリの記述方法が違います。

直接起動割り込みハンドラの場合，通常の割り込みエントリと同様に分岐命令を記述します。サンプルでは，割り込み INTP110 (ハンドラ・アドレス：0x180 番地) を直接起動割り込みハンドラの例としています。

CA850 版では .section 疑似命令を，GHS 製コンパイラ対応版では .org 命令を使用します。それぞれの命令については，「CA850 ユーザーズ・マニュアル アセンブリー言語編」，または，GHS 製コンパイラの言語関連のマニュアルを参照してください。直接起動割り込みハンドラのエントリは，次のようになります。

【 CA850 対応版 】

```
        .section        "INTP110"
        jr              _intp110_entry
```

【 GHS 製コンパイラ対応版 】

```
        .org            0x00000180
        jr              _intp110_entry
```

なお，飛び先のラベル _intp110_entry も同じファイルで定義されており，直接起動割り込みハンドラの前処理，後処理の記述 (マクロ記述) をしたあと，ハンドラ本体 intp130 へジャンプしています。直接起動割り込みハンドラの記述方法については，「RX850 Pro ユーザーズ・マニュアル 基礎編」を参照してください。

間接起動割り込みハンドラの場合，RX850 Pro で用意しているマクロを使用します。つまり，マクロ内容をハンドラ・アドレスに割りつける必要があります。マクロ名は RTOS_IntEntry_Indirect です。サンプルでは，割り込み INTP120 (ハンドラ・アドレス：0x1c0 番地) を間接起動割り込みハンドラの例としています。

CA850 対応版では .section 疑似命令を，GHS 製コンパイラ対応版では .org 命令を使用します。それぞれの命令については，「CA850 ユーザーズ・マニュアル アセンブリー言語編」，または，GHS 製コンパイラの言語関連のマニュアルを参照してください。間接起動割り込みハンドラのエントリは，次のようになります。

【 CA850 対応版 】

```
        .section        "INTP120"
        RTOS_IntEntry_Indirect
```

【 GHS 製コンパイラ対応版 】

```
        .org            0x000001c0
        RTOS_IntEntry_Indirect
```

また，RX850 Pro で使用するタイマ割り込みに関しても，間接起動割り込みハンドラと同じように割り込みエントリを登録する必要があります。サンプル (V850E 用) では，INTCMD0 (ハンドラ・アドレス：0x240 番地) をタイマ割り込みとして使用しているので，次のように記述します。

【 CA850 対応版 】

```
        .section        "INTCMD0"
        RTOS_IntEntry_Indirect
```

【 GHS 製コンパイラ対応版 】

```
        .org            0x00000240
        RTOS_IntEntry_Indirect
```

備考　システム・コンフィギュレーション・ファイルで定義されている割り込みハンドラについては，コンフィギュレータが該当割り込みエントリをシステム情報テーブルに自動出力しているため,ユーザが該当割り込みエントリを記述する必要がありません。

3.4　処理プログラムの作成

処理プログラム (アプリケーション本体) を作成します。
RX850 Pro で必要となるアプリケーションの処理単位は，大きく分けて次のようになります。

- タスク
- 直接起動割り込みハンドラ
- 間接起動割り込みハンドラ
- 周期起動ハンドラ
- 拡張 SVC ハンドラ

サンプルでは，拡張 SVC ハンドラ以外の 4 つを使用しています。次の表がサンプルの内容です。

表 3-3　処理プログラムの構成

サンプル・ファイル名	種別	関数名	機能
task.c	タスク	task1 task2	タスク本体
handler.c	周期起動ハンドラ 間接起動割り込みハンドラ 直接起動割り込みハンドラ	cychdr0 cychdr1 intp120 intp130	各種ハンドラ処理

C 言語で記述された処理プログラムで，システム・コールを発行している場合には，RX850 Pro が提供しているヘッダ・ファイル stdrx85p.h をインクルードしてください。システム・コールを使用するうえで必要な定義が含まれています。また，サンプルにあるヘッダ・ファイル usr.h は，必要に応じて関数が使用する定数等を定義し，プログラム中でインクルードします。CA850 対応版のサンプルでは定数のマクロ定義のみですが，GHS 製コンパイラ対応版のサンプルでは V850 シリーズの周辺 I/O ポートの名前とアドレスのマクロ定義も記述されています。
なお，拡張 SVC ハンドラの記述方法については，「RX850 Pro ユーザーズ・マニュアル 基礎編」を参照してください。

3.5　初期化データの退避領域の作成

CA850 対応版を使用するとき，この初期化データの退避領域を作成する必要があります。
これは，初期化データを ROM に格納しておき，プログラム実行前にその初期値を RAM にコピーする必要があるためです。格納元である ROM 領域を確保するのが，この初期化データの退避領域の作成に当たります。
この作成方法については，「CA850 ユーザーズ・マニュアル 操作編」の「ROM 化プロセッサ」の項を参照してください。
GHS 製コンパイラ対応版では，この処理はサンプルのソフトウエア初期化部 varfunc 内で行っています。

3.6　リンク・ディレクティブ・ファイルの作成

リンカがリンク時に参照するセクション情報，アドレス情報が記述されたリンク・ディレクティブ・ファイルを作成します。サンプルでは，次のファイルがこれに当たります。

- sample.dir (CA850 対応版)
- sample.ld (GHS 製コンパイラ対応版)

RX850 Pro において必須のセクションが存在し，それは次のようになっています。

表 3-4　RX850 Pro の必須セクション

セクション名	領域の種類
.sit	システム情報領域
.system	RX850 Pro 共通部分配置領域
.system_cmn	RX850 Pro スケジューラ関連配置領域
.system_int	RX850 Pro 割り込み処理関連配置領域
.text	RX850 Pro インタフェース・ライブラリ / システム・コール配置領域

これらの属性は .sit が const 属性，残りは text 属性になっています。これらのセクション情報を，リンク・ディレクティブ・ファイルに定義する必要があります。サンプルのファイルに記述してあるように，これらは位置情報を定義します。次がサンプルの該当部分です。

【 CA850 対応版 】

```
const :     !LOAD       ?R              V0x00001000 {
            .sit        = $PROGBITS     ?A          .sit ;
            .const      = $PROGBITS     ?A          .const ;
} ;
TEXT :      !LOAD       ?RX {
            .system     = $PROGBITS     ?AX         .system ;
            .system_cmn = $PROGBITS     ?AX         .system_cmn ;
            .system_int = $PROGBITS     ?AX         .system_int ;
            .text       = $PROGBITS     ?AX         .text ;
} ;
            :
            :
```

【 GHS 製コンパイラ対応版 】

```
-sec {
            .sit        0x00001000      :
            .system                     :
            .system_int                 :
            .system_cmn                 :
            .text                       :
            :
            :
```

そのほか，.data / .bss セクション等の RAM 領域に関するセクション，const セクション等の ROM 領域に関するセクションについて，必要なものも定義していきます。リンク・ディレクティブ・ファイルの記述は，サンプルを基に，ユーザにあわせた環境へ変更していくことを推奨します。

なお，リンク・ディレクティブ・ファイルの記述方法については，「CA850 ユーザーズ・マニュアル 操作編」，または，GHS 製コンパイラの言語関連のマニュアルを参照してください。

3.7　ロード・モジュールの作成

次に，ロード・モジュール (実行可能モジュール) を作成します。

これまで作成した C 言語ソース・ファイル，アセンブリー言語ソース・ファイルをコンパイル，アセンブルし，できたオブジェクト (.o ファイル) 群を「3.6　リンク・ディレクティブ・ファイルの作成」で作成したリンク・ディレクティブ・ファイルに基づいてリンクします。

RX850 Pro を使用したアプリケーションをリンクする際，次のライブラリの参照，および，オブジェクトのリンクが必要になります。

- ニュークリアス・ライブラリ

librxp.a	：rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要あり
librxpm.a	：rel_blk の発行直前で対象メモリ・ブロックの先頭 4 バイトをゼロ・クリアする必要なし

- インタフェース・ライブラリ

libchp.a	：パラメータ・チェック機能あり
libncp.a	：パラメータ・チェック機能なし
libdbp.a	：パラメータ・チェック機能あり，デバッグ用システム・コールを含む

- ニュークリアス共通オブジェクト

rxcore.o	：周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可
rxtmcore.o	：周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可
rxdbcore.o	：周期起動ハンドラ内ではすべてのマスカブル割り込みを受付許可，デバッグ用システム・コールを含む
rxdbtmcore.o	：周期起動ハンドラ内ではタイマ割り込みよりも優先度の高いマスカブル割り込みを受け付け許可，デバッグ用システム・コールを含む

ニュークリアス・ライブラリ，インタフェース・ライブラリ，ニュークリアス共通オブジェクトは複数の種類が用意されています。それぞれの用途に応じて使用するライブラリを決定してください。なお，インタフェース・ライブラリの詳細については，「第 4 章　インタフェース・ライブラリ」を，ニュークリアス共通オブジェクトについては，「RX850 Pro ユーザーズ・マニュアル 基礎編」の「時間管理機能」の項を参照してください。

リンクに成功すると実行可能モジュール (.out ファイル) が完成します。この段階でデバッガに読み込んでアプリケーションを実行させることができます。

リンカにより作成されたロード・モジュールは，初期化データに関しては RAM に正しく配置された状態になっています。そのため，CA850 対応版の場合で，かつ初期化データがアプリケーションに存在する場合，初期化データ退避領域を確保し，コピー・ルーチンを組み込んだモジュールを作成する必要があります。この場合はリンカが生成したロード・モジュールに対し，ROM 化プロセッサを通したロード・モジュールを作成する必要があります。

ROM 化プロセッサの使用方法，および，コピー・ルーチンについては，「CA850 ユーザーズ・マニュアル 操作編」の「ROM 化プロセッサ」の項を参照してください。

3.8　システムへの組み込み

次に，完成したロード・モジュールをシステムに組み込みます。

組み込むためには，「3.7　ロード・モジュールの作成」で作成したロード・モジュールをヘキサ・ファイルにコンバートする必要があります。

CA850 対応版，および，GHS 製コンパイラ対応版それぞれに用意されているヘキサ・コンバータを使って，必要なフォーマットのヘキサ・ファイルにします。そして，ROM ライタ等を使用してシステムに組み込みます。

第 4 章　インタフェース・ライブラリ

この章では，RX850 Pro が提供するインタフェース・ライブラリについて説明します。

4.1　概要

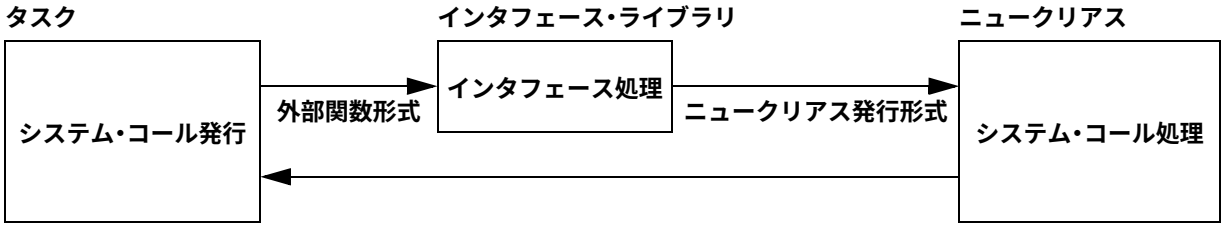
RX850 Pro では，ユーザの処理プログラムと RX850 Pro のニュークリアスの中間に位置するインタフェース・ライブラリを提供しています。インタフェース・ライブラリは，ニュークリアスが処理を行ううえで必要な各種情報の設定等を行ったあとに制御を移す機能を持っています。

処理プログラム (タスク，非タスク) を C 言語で記述した場合，システム・コールの発行形式や拡張 SVC ハンドラの呼び出し形式は外部関数形式となります。しかし，ニュークリアスが理解可能な発行形式 (ニュークリアス発行形式) と外部関数形式には相違があります。

そこで，システム・コールの発行形式や拡張 SVC ハンドラの呼び出し形式を外部関数形式からニュークリアス発行形式に変換する手続き (インタフェース) が必要になります。このような，処理プログラムとニュークリアスの仲介役を行うインタフェースは各システム・コール毎にあり，これを集めたものがインタフェース・ライブラリです。

図 4-1 に，インタフェース・ライブラリの位置づけを示します。

図 4-1　インタフェース・ライブラリの位置づけ



4.2　インタフェース・ライブラリ内での処理

インタフェース・ライブラリでは，次のような処理を行っています。

- ニュークリアスが管理しているテーブルに必要な情報を設定する
- 必要になるデータをレジスタに設定する
- システム・コールのエラー値を設定 (ただし，ニュークリアス内で設定されるエラーは除く) したあと，処理プログラムへ処理を戻す

インタフェース・ライブラリを用意することにより，ニュークリアスとユーザの処理プログラムの分離が容易になります。インタフェース・ライブラリはユーザ・アプリケーション側にリンクされることになるので，ニュークリアス本体を ROM 化したあとでユーザの処理プログラムを変更する必要が生じた場合でも，ニュークリアス本体が格納されている ROM を変更する必要がなくなります。また，ロード・モジュールを分割して作成するといったことも可能になります。

「4.5　システム・コール用インタフェース・ライブラリ」では，システム・コール用インタフェース・ライブラリの，「4.6　拡張 SVC ハンドラ用インタフェース・ライブラリ」では，拡張 SVC ハンドラのインタフェース・ライブラリの具体的な書式を掲載していますので，詳細はそちらを参照してください。

4.3　インタフェース・ライブラリの種類

RX850 Pro が提供するインタフェース・ライブラリには，システム・コールのパラメータをチェックする機能を持つものと持たないものに大別されます。システムにどちらを組み込むかは，リンク時に指定します。

パラメータ・チェック機能のあるライブラリを使用すると，システム・コール発行時のパラメータの指定が不正だった場合，必ず戻り値が返されます。一方，パラメータ・チェック機能のないライブラリを使用すると，システム・コール発行時のパラメータの指定が不正だった場合，戻り値が返されない場合があります。

これらのライブラリは，用途に応じて使い分けることができます。たとえば，デバッグ時はパラメータ・チェック機能のあるものを使用し,実際の組み込み時はパラメータ･チェック機能のないものを使用することで,プログラムのパフォーマンス向上や容量の節減等を実現します。

パラメータ・チェック機能のないライブラリで戻り値が返されるエラーは，「RX850 Pro ユーザーズ・マニュアル 基礎編」の「第 11 章 システム・コール」の各システム・コールの戻り値の欄で，* を付けて示しています。なお，パラメータ・チェック機能のないライブラリを使用したとき，戻り値が返されないエラーが発生した場合は，アプリケーション・システムの動作は保証されません。

4.4　提供されているインタフェース・ライブラリ

RX850 Pro が提供しているインタフェース・ライブラリは，次の 2 種類です。

- CA850 対応版
- GHS 製コンパイラ対応版

このほかのコンパイラを使用する場合，または，ユーザが必要に応じてインタフェース・ライブラリを変更したい場合は，インタフェース・ライブラリを書き換える必要があります。

変更したインタフェース・ライブラリは，アセンブルし，あらためてライブラリ化する必要があります。

4.5　システム・コール用インタフェース・ライブラリ

システム・コール用インタフェース・ライブラリでは，次のような処理を行います。
ただし，システム・コールのパラメータの引き渡し方法については，使用する C コンパイラに準じます。

- システム・コールの機能コードを r10 レジスタに設定
- システム・コールからの戻り番地を lp レジスタに設定
- システム・コールのパラメータをチェック
- システム・コールの入り口のアドレス (hp レジスタ + 0x100 番地) を獲得
- システム・コールの入り口にジャンプ

また，システム・コールのパラメータをチェックした際にエラーを検出した場合，次の処理を行います。

- 検出したエラーに対応したエラー・コードを r10 レジスタに設定

図 4-2 に，システム・コール用インタフェース・ライブラリの記述例を示します。

図 4-2　システム・コール用インタフェース・ライブラリの記述例

```
                .text
                .globl          _syscall_name
                .align          2
_syscall_name
                -- 機能コード設定
                mov             func_code , r10

                -- パラメータをチェック
                ……
                ……

                jz              _syscall_err

                -- システム・コールの入り口のアドレスを獲得
                ld.w            0x100 [ hp ] , r12

                -- システム・コールの入り口にジャンプ
                jmp             [ r12 ]
```

4.6　拡張 SVC ハンドラ用インタフェース・ライブラリ

拡張 SVC ハンドラ用インタフェース・ライブラリでは，次のような処理を行います。
ただし，拡張 SVC ハンドラのパラメータの引き渡し方法については，使用する C コンパイラに準じます。

- 拡張 SVC ハンドラの機能コードを r10 レジスタに設定
- 拡張 SVC ハンドラからの戻り番地を lp レジスタに設定
- 拡張 SVC ハンドラのパラメータをチェック
- 拡張 SVC ハンドラのパラメータ領域のサイズを r11 レジスタに設定

  例　　int 型のパラメータが 4 個の場合，0x10 を r11 レジスタに設定

- 拡張 SVC ハンドラの入り口のアドレスを獲得 (hp レジスタ + 0x108 番地) を獲得
- 拡張 SVC ハンドラの入り口にジャンプ

また，拡張 SVC ハンドラのパラメータをチェックした際にエラーを検出した場合，次のような処理を行います。

- 検出したエラーに対応したエラー・コードを r10 レジスタに設定

図 4-3 に，拡張 SVC ハンドラ用インタフェース・ライブラリの記述例を示します。

図 4-3　拡張 SVC ハンドラ用インタフェース・ライブラリの記述例

```
                .text
                .globl          _svchdr_name
                .align          2
_svchdr_name
                -- 機能コード設定
                mov             func_code , r10

                -- パラメータをチェック
                ·····························
                ·····························

                jz              _svchdr_err

                -- パラメータ領域のサイズを設定
                mov             prm_siz , r11

                -- 拡張 SVC ハンドラの入り口のアドレスを獲得
                ld.w            0x108 [ hp ] , r12

                -- 拡張 SVC ハンドラの入り口にジャンプ
                jmp             [ r12 ]
```

第 5 章　メモリとその容量の見積もり

この章では，RX850 Pro におけるメモリ (RAM) の管理と，使用する容量について説明します。

5.1　SPOL と UPOL

RX850 Pro で使用する RAM 領域は，次の 4 つです。

SPOL0　…　System Memory Pool 0 (システム・メモリ・プール 0)
SPOL1　…　System Memory Pool 1 (システム・メモリ・プール 1)
UPOL0　…　User Memory Pool 0 (ユーザ・メモリ・プール 0)
UPOL1　…　User Memory Pool 1 (ユーザ・メモリ・プール 1)

これらの配置情報は，システム・コンフィギュレーション・ファイルにて「先頭アドレス」と「サイズ」を指定することによって決定されます。つまり，使用できる RAM 領域上のアドレスとそのサイズを指定することになります。
また，これらのメモリ・プールは用途が決まっており，次のような分類になります。

表 5-1　メモリ・プールの種類と割り付けられるもの

メモリ・プール名	割りつけられるもの
SPOL0	システム・ベース・テーブル レディー・キュー 各種管理ブロック タスク・スタック 割り込みハンドラ用スタック
SPOL1	タスク・スタック 割り込みハンドラ用スタック 可変長メモリ・プール 固定長メモリ・プール
UPOL0	可変長メモリ・プール
UPOL1	可変長メモリ・プール

SPOL0 は，RX850 Pro のシステムに関する情報が配置されるため，生成が必須なメモリ・プールです。SPOL1 は SPOL0 だけでまかなえる場合は，作成しなくても良いメモリ・プールです。また，管理ブロックが配置される SPOL0 を内蔵 RAM に配置し，比較的大きいサイズを必要とするスタックを SPOL1 として外部 RAM に配置することによって，システムのパフォーマンスを向上させる使用方法もあります。
UPOL0 は RX850 Pro のメモリ管理機能を使用する場合に必要となります。その場合は UPOL0 から作成してください。UPOL1 だけの作成はできません。

5.2　管理領域のメモリ容量

RX850 Pro における，システム・ベース・テーブル，各種管理ブロックが使用するサイズについて説明します。
これらは SPOL0 より確保されます。表 5-2 に，各オブジェクト 1 つ当たりが使用する管理領域のサイズとその算出方法を示します。

表 5-2　オブジェクト管理領域のサイズ

オブジェクト名	1 オブジェクト当たりの管理領域サイズ (単位:バイト)	サイズの算出方法 (単位：バイト)
オペレーティング・システム管理テーブル	520 ～ 1048	504 + Align32 (優先度数 + 4) / 8 + Align4 ((優先度数 + 4) * 2) 優先度数は，システム最大値情報 *pri_lvl* と同値。
システム・メモリ領域管理テーブル	8	8 * 4 = 32 SPOL0，SPOL1，UPOL0，UPOL1 それぞれにつき 8 バイト。4 つすべてを生成しなくても，必ず 4 つ分確保されるため，常に 32 バイト確保される。
タスク管理テーブル	56	56 * タスクの最大生成数 タスクの最大生成数は，システム最大値情報 *maxtsk* と同値。
セマフォ管理テーブル	20	20 * セマフォの最大生成数 セマフォの最大生成数は，システム最大値情報 *maxsem* と同値。
イベントフラグ管理テーブル	20	20 * イベントフラグの最大生成数 イベントフラグの最大生成数は，システム最大値情報 *maxflg* と同値。
メールボックス管理テーブル	20	20 * メールボックスの最大生成数 メールボックスの最大生成数は，システム最大値情報 *maxmbx* と同値。
割り込みハンドラ管理テーブル	16	16 *割り込みハンドラの最大生成数+ Align4 (最大割り込み要因番号) 割り込みハンドラの最大生成数は，システム最大値情報 *maxint* と同値。 最大割り込み要因番号は，システム最大値情報 *maxintfactor* と同値。
周期起動ハンドラ管理テーブル	40	40 * 周期起動ハンドラの最大生成数 周期起動ハンドラの最大生成数は，システム最大値情報 *maxcyc* と同値。
メモリ・プール管理テーブル	24	24 * メモリ・プールの最大生成数 メモリ・プールの最大生成数は，システム最大値情報 *maxmpl* と同値。
拡張 SVC ハンドラ管理テーブル	16	16 * 拡張 SVC ハンドラの最大生成数 拡張 SVC ハンドラの最大生成数は，システム最大値情報 *maxsvc* と同値。

5.3　タスク・スタックの容量

タスク・スタック領域は，次の 4 つに分類されます。

- タスク・スタック管理テーブル
- コンテキスト領域
- 割り込みスタック・フレーム
- タスク使用領域

タスク生成時（ コンフィギュレーション時 ）に，タスク・スタックのサイズを指定しますが，その値は「割り込みスタック・フレーム」と「タスク使用領域」の合計サイズとなります。実際にメモリに確保されるサイズは，上記の「タスク・スタック管理テーブル」と「コンテキスト領域」のサイズがさらに加算されたサイズになります (さらに，その値を 4 バイトでアラインした値)。
「タスク使用領域」はユーザのアプリケーションによって変動しますが，「タスク・スタック管理テーブル」「コンテキスト領域」，および，「割り込みスタック・フレーム」のサイズは決まっており，そのサイズは次のようになっています。

表 5-3　タスクで使用するタスク・スタックのサイズ

タスク・スタック領域	サイズ (単位：バイト)	
	V850E 用	V850 用
タスク・スタック管理テーブル	28	
コンテキスト領域 (割り込みスタック・フレームを含む：72 バイト)	148	140
タスク使用領域	アプリケーション依存	

タスク生成時 (コンフィギュレーション時) に，タスク・スタックのサイズを 100 バイトと指定した場合，実際にメモリに確保されるスタック・サイズは，次のようになります。

【 V850E 用 】
100 + 28 + 148 = 276 バイト

【 V850 用　】
100 + 28 + 140 = 268 バイト

また，タスクから拡張 SVC ハンドラを起動している場合，ハンドラ実行のためのレジスタ退避領域と，拡張 SVC ハンドラ自身が消費するスタック領域が必要となります。これらのサイズは次のようになります。

表 5-4　拡張 SVC ハンドラで使用するタスク・スタックのサイズ

タスク・スタック領域	サイズ (単位：バイト)	
	V850E 用	V850 用
拡張 SVC ハンドラ用レジスタ退避領域	28	20
拡張 SVC ハンドラ使用領域	アプリケーション依存	

タスク・スタック領域は，タスク生成時に SPOL0，または，SPOL1 から確保され，タスク削除 (del_tsk，exd_tsk，ter_tsk) によって解放されます。したがって，すべてのタスクが同時に生成される可能性のある場合は，すべてのタスクにおいてサイズを計算し，それらを合計したサイズが確保可能となるように，SPOL0，SPOL1 のサイズを決定する必要があります。すべてのタスクが同時に生成されない場合は，同時に生成された状態となるタスクの組み合わせの中で，スタックのサイズの合計が最大となる組み合わせのサイズを計算し，そのサイズを基に，SPOL0，SPOL1 のサイズを決定します。

次に，タスク・スタック・サイズの計算方法をまとめたものを示します。全サイズの合計がメモリ領域として確保する必要のあるサイズ，網掛け部分のサイズの合計が，タスク作成時（ コンフィギュレーション時 ）に指定するタスク・スタック・サイズです。なお，メモリ領域に確保される値は 4 バイトでアラインされた値です。

表 5-5　タスク・スタックで使用されるサイズのまとめ

タスク・スタック領域	サイズ (単位：バイト)		備考
	V850E 用	V850 用	
タスク・スタック管理テーブル	28		
コンテキスト領域 (割り込みスタック・フレームを含む：72 バイト)	148	140	
タスク使用領域	アプリケーション依存		タスクがプッシュ・ポップするスタック・サイズを算出して指定。 使用している変数の数等を考慮。 タスクからシステム・コールが発行された際に RX850 Pro は lp (r31) をプッシュするので 4 バイト加算。
拡張 SVC ハンドラ用レジスタ退避領域	28	20	拡張 SVC ハンドラを使用していない場合は不要。
拡張 SVC ハンドラ使用領域	アプリケーション依存		拡張 SVC ハンドラを使用していない場合は不要。 拡張 SVC ハンドラからシステム・コールが発行された際に RX850 Pro は lp (r31) をプッシュするので 4 バイト加算。

5.4　割り込みハンドラ用スタックの容量

割り込みハンドラ用スタック領域は，次の 4 つにおいて使用されます。

- 初期化ハンドラ
- アイドル・タスク
- 割り込みハンドラ
- 周期起動ハンドラ

それぞれ，使用されるサイズについて説明します。

1) 初期化ハンドラ
初期化ハンドラとして記述した関数が消費 (プッシュ・ポップ) するサイズ分の領域を確保します。初期化ハンドラ実行中は，タスクや割り込みハンドラが起動することがないため，「2) アイドル・タスク」以降で説明している領域サイズよりも，その消費サイズが小さい場合は，初期化ハンドラで消費するスタック・サイズを考慮する必要はありません。

2) アイドル・タスク
アイドル・タスク実行中，および，タスク終了後 (del_tsk，exd_tsk，ter_tsk 発行後) に，次タスクが実行されるまでの間に入る割り込み用のスタック消費分です。このサイズとして 72 バイトが必要です。スタックがさらに伸長するケースがありますが，その場合については，「3) 割り込みハンドラ」の説明以降を参照してください。
なお，この 72 バイト分の領域は，初期化処理時に加算されていますので，コンフィギュレーション時に考慮に入れる必要はありません。つまり，コンフィギュレーション時に指定する割り込みハンドラ用スタックのサイズに 72 バイト加えた値が，実際に確保されるメモリ・サイズとなります。

3) 割り込みハンドラ
割り込みが発生する場合，初回割り込み発生時 (タスクに割り込みが入ったとき) は，タスク・スタック，または，「2) アイドル・タスク」で示したアイドル・タスク用の領域に生成されます。以後，多重割り込みが発生する可能性がある場合は，最大ネスト回数分の割り込みスタック・フレームのサイズを加算する必要があります。
割り込みハンドラが起動する際は，割り込みスタック・フレームとは別に，レジスタ退避領域として，V850E の場合 28 バイト，V850 の場合 20 バイトが必要となります。たとえば，タスク・スタックに割り込みスタック・フレーム分の情報を積んだあと，SP (スタック・ポインタ) を割り込みハンドラ用スタックに切り替えて，さらに積まれる分に当たります。多重割り込みを許可するシステムの場合，このサイズを考慮に入れる必要があります。よって，「割り込みの最大ネスト回数 + 1 (初回割り込み分)」に，V850E の場合は 28，V850E の場合は 20 を掛けた値を，割り込みハンドラ用スタックのサイズに加算します。
さらに，割り込みハンドラとして記述した関数が，プッシュ・ポップによって消費するスタック・サイズを，割り込みが最大にネストした場合を考慮して加算します。これには，割り込みハンドラ内でシステム・コールを発行する場合 4 バイトを，拡張 SVC ハンドラを発行する場合は，V850E の場合 28 バイト，V850 の場合 20 バイトを加算します。また，拡張 SVC ハンドラ内でシステム・コールを発行している場合は，さらに 4 バイトを加算します。
タイマ割り込みハンドラは，プッシュ・ポップによってスタックを消費しない割り込みハンドラとして扱います。これを考慮したサイズを算出します。

4) 周期起動ハンドラ
周期起動ハンドラは，タイマ割り込みハンドラから呼び出されるサブルーチンとして実装されています。
また，周期起動ハンドラ実行中に，新たなタイマ割り込みが発生し，別の周期起動ハンドラが起動するような態になっても，実行中だった周期起動ハンドラの処理が優先されます。したがって，周期起動ハンドラとして記述した関数の中で，その関数自身が消費するスタック・サイズの一番大きいものを，ハンドラ用スタックのサイズに加算します。

次に，割り込みハンドラ用スタックのサイズの計算方法をまとめたものを示します。全サイズの合計がメモリ領域として確保する必要のあるサイズ，網掛け部分のサイズの合計が，コンフィギュレーション時に指定する割り込みハンドラ用スタックのサイズです。なお，メモリ領域に確保される値は 4 バイトでアラインされた値です。

表 5-6　多重割り込みを受け付けないシステムにおける割り込みハンドラ用スタック・サイズ

割り込みハンドラ用スタック領域	サイズ (単位：バイト)		備考
	V850E 用	V850 用	
アイドル・タスク用領域	72		
割り込みハンドラ用レジスタ退避領域	28	20	
割り込みハンドラ使用領域	アプリケーション依存		割り込みハンドラがプッシュ・ポップするスタック・サイズを算出して指定。 使用している変数の数等を考慮。 割り込みハンドラからシステム・コールが発行された際に RX850 Pro は lp (r31) をプッシュするので 4 バイト加算。 使用する割り込みハンドラ中で最もスタックを使用するハンドラ (周期起動ハンドラを含む) の使用スタック・サイズを指定。
拡張 SVC ハンドラ用レジスタ退避領域	28	20	拡張SVCハンドラを割り込みハンドラが呼び出していない場合は不要。
拡張 SVC ハンドラ使用領域	アプリケーション依存		拡張SVCハンドラを割り込みハンドラが呼び出していない場合は不要。 拡張 SVC ハンドラからシステム・コールが発行された際に RX850 Pro は lp (r31) をプッシュするので 4 バイト加算。

この表は多重割り込みを受け付けない場合のものですが，周期起動ハンドラに対して割り込みが入り，その割り込みを受け付けてしまう場合は，多重割り込みと同等になります。つまり，この表に該当するケースとしては，ニュークリアス共通部オブジェクトとして rxtmcore.o (周期起動ハンドラ内でタイマ割り込みより高い割り込み受け付け可能版) を使用し，タイマ割り込みより高い割り込みを使用していないこと，および，割り込みハンドラはすべて割り込み禁止状態で動作するアプリケーションのときにのみ該当します。

表 5-7　多重割り込みを受け付けるシステムにおける割り込みハンドラ用スタック・サイズ

割り込みハンドラ用スタック領域	サイズ (単位：バイト)		備考
	V850E 用	V850 用	
アイドル・タスク用領域	72		
割り込みスタック・フレーム	72 * n		
割り込みハンドラ用レジスタ退避領域	28 * (n+1)	20 * (n+1)	
割り込みハンドラ使用領域	アプリケーション依存		割り込みハンドラがプッシュ・ポップするスタック・サイズを算出して指定。 使用している変数の数等を考慮。 割り込みハンドラからシステム・コールが発行された際に RX850 Pro は lp (r31) をプッシュするので 4 バイト加算。 使用する割り込みハンドラ毎 (周期起動ハンドラを含む) に使用スタック・サイズを算出し，その合計を使用スタック・サイズを指定。

割り込みハンドラ用スタック領域	サイズ (単位：バイト)		備考
	V850E 用	V850 用	
拡張 SVC ハンドラ用レジスタ退避領域	28 * m	20 * m	拡張SVCハンドラを割り込みハンドラが呼び出していない場合は不要。
拡張 SVC ハンドラ使用領域	アプリケーション依存		拡張SVCハンドラを割り込みハンドラが呼び出していない場合は不要。 拡張 SVC ハンドラからシステム・コールが発行された際に RX850 Pro は lp (r31) をプッシュするので 4 バイト加算。

上記の表で，n は割り込みの最大ネスト回数を，m は拡張 SVC ハンドラを使用している割り込みハンドラの数を指します。

5.5　メモリ・プールの容量

メモリ・プールの容量について説明します。メモリ・プールの容量は，メモリ・プール生成時 (コンフィギュレーション時) に指定しますが，実際に割り当てられるメモリ・サイズは，その指定サイズに 8 を加えたものを 4 バイトでアラインした値となります。

加える 8 バイトは，メモリ・プールの先頭に設けられ，メモリ・プールの管理領域になっています。メモリ・プールとして実際に確保されるサイズは大きくなることに注意が必要です。

表 5-8　メモリ・プールのサイズ

オブジェクト名	サイズの算出方法 (単位：バイト)
メモリ・プール	Align4 (メモリ・プールのサイズ + 8) メモリ・プールのサイズは，システム・コール cre_mpl 発行時，または，メモリ・プール情報 *mpl_siz* と同値。

5.6　メモリ容量見積もりの例

次に，RX850 Pro の管理領域 SPOL，UPOL として使用されるメモリ容量の見積もり方法の例を示します。
ここでは，対象 CPU が V850E であり，システム・コール cre_tsk，cre_mpl は発行されないものとして見積もります。

【 アプリケーション情報 】

情報	値 (単位：バイト)
割り込みハンドラ用スタック領域 *intstk_siz*	SPOL0 より 256
タスクの優先度範囲 *pri_lvl*	15
タスクの最大生成数 *maxtsk*	2
セマフォの最大生成数 *maxsem*	1
イベントフラグの最大生成数 *maxflg*	2
メールボックスの最大生成数 *maxmbx*	3
割り込みハンドラの最大生成数 *maxint*	4
メモリ・プールの最大生成数 *maxmpl*	2
周期起動ハンドラの最大登録数 *maxcyc*	1
拡張 SVC ハンドラの最大登録数 *maxsvc*	1
割り込み要因番号の最大値 *maxintfactor*	56
タスク・スタック情報	SPOL0 より 256 SPOL0 より 256
メモリ・プール情報	UPOL0 より 4096 UPOL1 より 8192

【 見積もり方法 】

オブジェクト情報	サイズ (単位：バイト)
オペレーティング・システム管理テーブル	SPOL0 より 504 + Align32 (15 + 4) / 8 + Align4 ((15 + 4) * 2) = 548
システム・メモリ管理テーブル	SPOL0 より 8 * 4 = 32
タスク管理テーブル	SPOL0 より 56 * 2 = 112
セマフォ管理テーブル	SPOL0 より 20 * 1 = 20
イベントフラグ管理テーブル	SPOL0 より 20 * 2 = 40
メールボックス管理テーブル	SPOL0 より 20 * 3 = 60
割り込みハンドラ管理テーブル	SPOL0 より 16 * 4 + Align4 (56) = 120
メモリ・プール管理テーブル	SPOL0 より 24 * 2 = 48
周期起動ハンドラ管理テーブル	SPOL0 より 40 * 1 = 40

オブジェクト情報	サイズ (単位：バイト)
拡張 SVC ハンドラ管理テーブル	SPOL0 より 16 * 1 = 16
タスク・スタック	SPOL0 より Align4 (100 + 256) + Align4 (100 + 256) = 712
割り込みハンドラ用スタック	SPOL0 より Align4 (256 + 80) = 336
メモリ・プール	UPOL0 より 4096 + 8 = 4104 UPOL1 より 8192 + 8 = 8200

上記の計算結果より

SPOL0 ： 548 + 32 + 112 + 20 + 40 + 60 + 120 + 48 + 40 + 16 + 712 + 336 = 2084 バイト
SPOL1 ： 0 バイト
UPOL0 ： 4104 バイト
UPOL1 ： 8200 バイト

がそれぞれ必要であることになります。

第 6 章　システム・コンフィギュレーション・ファイル

この章では，システム・コンフィギュレーション・ファイルと，その記述方法について説明します。

6.1　概要

RX850 Pro を使ったシステムを構築する場合，必要とする各種データ (システム情報や資源情報等) と使用するシステム・コールの種類を保持した情報が必要となります。この情報は，システム情報テーブル・ファイル，および，システム・コール・テーブル・ファイルと呼ばれます。

システム情報テーブル・ファイル，および，システム・コール・テーブル・ファイルはアセンブリー言語で記述されており，規定された形式のデータ羅列となっています。これらは，各種エディタで記述することは可能ですが，変更や追加が非常に難しいものであるため，記述にはかなりの労力を必要とします。

そこで，コンフィギュレータ CF850 Pro というアプリケーションを提供しています。

これは，システム・コンフィギュレーション・ファイルという，RX850 Pro のシステムや資源，使用するシステム・コールに関する情報を記述した，独自のフォーマットのファイルを，システム情報テーブル・ファイルとシステム・コール・テーブル・ファイルに変換するアプリケーションです。つまり，ユーザはシステム・コンフィギュレーション・ファイルを作成し，CF850 Pro によって，システム情報テーブル・ファイルとシステム・コール・テーブル・ファイルを得ることができます。

CF850 Pro は，システム・コンフィギュレーション・ファイルからシステム情報テーブル・ファイル，システム・コール・テーブル・ファイル，システム情報ヘッダ・ファイルの 3 つのファイルを出力します。システム情報ヘッダ・ファイルは，作成したタスク，セマフォ等，資源 ID として指定されたシンボル名と実際の ID 番号を，#define 命令で対応させる記述があるファイルです。

CF850 Pro の起動方法に関しては，「第 7 章　コンフィギュレータ CF850 Pro」を参照してください。

次に，システム・コンフィギュレーション・ファイルの記述方法について説明します。

6.2　表記方法

次に，システム・コンフィギュレーション・ファイルを記述する際の表記方法を示します。

1) 構成要素と文字コード

a) 文字コード

システム・コンフィギュレーション・ファイルは，ASCII コードで記述します。なお，漢字コード (SJIS，EUC のみ) はコメント中にのみ使用できます。

b) 単語

システム・コンフィギュレーション・ファイルの中では，空白 (スペース・コード，タブ・コード，改行コード) を含まない一連の文字列を単語と呼びます。以降の説明中の数値，シンボル名，キー・ワードはすべて単語の一種です。

CF850 Pro は単語内の大文字と小文字を区別します。したがって，“ ABC ”，“ Abc ”，“ abc ” はそれぞれ別の単語として扱われます。

c) 文

システム・コンフィギュレーション・ファイルの中では，一つ以上の空白で区切られた一連の単語を文と呼びます。文の区切りは改行で行います。

なお，システム・コンフィギュレーション・ファイルでは，行末に “ \ ” を記述すると，次の行との併合を意味します。ただし，“ \ ” の直前 1 文字は，“ スペース ”，または，“ タブ ” でなければなりません。

2) 数値

単語の中で，先頭が数字コードで始まる単語は数値として扱われます。先頭の数字コードにより，表 6-1 に示すように数値が分類されます。

なお，数値は指定のないかぎり 32 ビット幅 (0x0 ～ 0xFFFFFFFF) の範囲で記述できます。

表 6-1　数値の種別

種別	先頭の数字コード	含まれる文字	例
8 進数	0	0 ～ 7	0123，0，056712

種別	先頭の数字コード	含まれる文字	例
10 進数	0 以外の数字	0 ～ 9	123，0，689525
16 進数	0x	0 ～ 9，a ～ f (，または，A ～ F)，x，X	0x12C，0X0，0xabcdef

3) **シンボル名**
名前とシンボル名は文脈によって区別します。シンボル名はユーザ・プログラム上の関数名や変数名を示します。ただし，シンボル名の先頭 1 文字は，“ _ ”，または，“ 英字 ” でなければなりません。

4) **コメント**
システム・コンフィギュレーション・ファイルでは，“ -- ” 以降行末までがコメントとして扱われます。

5) **キー・ワード**
CF850 Pro **では，次に示す文字列をキー・ワードとして予約しています。**
したがって，これらの文字列をほかの用途には使用できません。

clkhdr	clktim	cyc	defstk	flg	flgsvc
ini	inthdr	intstk	intsvc	maxcyc	maxflg
maxint	maxintfactor	maxmbx	maxmpl	maxpri	maxsem
maxsvc	maxtsk	mbx	mbxsvc	mem	mpl
mplsvc	no_use	prtflg	prtmbx	prtmpl	prtsem
prttsk	RX850PRO	rxsers	sct_def	sem	semsvc
ser_def	sit_def	SPOL0	SPOL1	svc	syssvc
TA_ASM	TA_DISINT	TA_ENAINT	TA_HLNG	TA_MFIFO	TA_MPRI
TA_TFIFO	TA_TPRI	TA_WMUL	TA_WSGL	TCY_OFF	TCY_ON
timsvc	tsk	tsksvc	TTS_DMT	TTS_RDY	UPOL0
UPOL1	V850	V850E1	V850E2	V850ES	

6.3　コンフィギュレーション情報

システム・コンフィギュレーション・ファイルに記述するコンフィギュレーション情報は，次の 3 **種類に大別されます。**

- **リアルタイム** OS **情報**
  使用するリアルタイム OS **に関するデータ**

- SIT(System Information Table) **情報**
  RX850 Pro **が動作するうえで必要となるデータ**

- SCT(System Call Table) **情報**
  システムで使用する機能 (システム・コール) であるか否かを選択したデータ

6.3.1　リアルタイム OS 情報

システム・コンフィギュレーション・ファイルに記述するリアルタイム OS **情報は，次に示す** 1 **項目です。**

1) RX **シリーズ情報**
RX **シリーズ情報として，次のデータを定義します。**

- **リアルタイム** OS **名**
- **バージョン番号**

6.3.2　SIT(System Information Table) 情報

システム・コンフィギュレーション・ファイルに記述する SIT 情報は，次に示す 12 項目です。

1) **システム情報**
システム情報として，次のデータを定義します。

- プロセッサ種別
- 基本クロック周期
- タイマの割り込み要因番号
- デフォルト・スタック・サイズ
- 割り込みハンドラ用スタック情報
  割り込みハンドラ用スタックのサイズ
  割り込みハンドラ用スタックを割り付けるシステム・メモリの種類
- ID 番号の保護範囲
  タスクの ID 番号保護範囲
  セマフォの ID 番号保護範囲
  イベントフラグの ID 番号保護範囲
  メールボックスの ID 番号保護範囲
  メモリ・プールの ID 番号保護範囲

2) **システム最大値情報**
システム最大値情報として，次のデータを定義します。

- タスクの優先度範囲
- 管理オブジェクトの最大生成数 / 最大登録数
  タスクの最大生成数
  セマフォの最大生成数
  イベントフラグの最大生成数
  メールボックスの最大生成数
  メモリ・プールの最大生成数
  周期起動ハンドラの最大登録数
  拡張 SVC ハンドラの最大登録数
  割り込みハンドラの最大登録数
  割り込み要因番号の最大値

3) **システム・メモリ情報**
個々のシステム・メモリ (System Memory Pool 0，System Memory Pool 1，User Memory Pool 0，User Memory Pool 1) に対して，次のデータを定義します。

- システム・メモリの種類
- セクション名
- サイズ

4) **タスク情報**
個々のタスクに対して，次のデータを定義します。

- ID 番号
- 初期状態
- 起動コード
- 拡張情報
- 記述言語
- 起動アドレス
- 初期優先度
- 割り込み状態
- タスク用スタック情報
  タスク用スタックのサイズ
  タスク用スタックを割り付けるシステム・メモリの種類
- 固有 GP レジスタ値
- 固有 TP レジスタ値
- キー ID 番号

5) **セマフォ情報**
個々のセマフォに対して，次のデータを定義します。

- ID 番号
- **拡張情報**
- **タスクのキューイング方式**
- **初期資源数**
- **最大資源数**
- **キー** ID **番号**

6) **イベントフラグ情報**
個々のイベントフラグに対して，次のデータを定義します。

- ID 番号
- **拡張情報**
- **複数タスクの待ち許可 / 禁止**
- **初期ビット・パターン**
- **キー** ID **番号**

7) **メールボックス情報**
個々のメールボックスに対して，次のデータを定義します。

- ID 番号
- **拡張情報**
- **タスクのキューイング方式**
- **メッセージのキューイング方式**
- **キー** ID **番号**

8) **割り込みハンドラ情報**
個々の割り込みハンドラに対して，次のデータを定義します。

- **割り込み要因番号**
- **記述言語**
- **起動アドレス**
- **固有** GP **レジスタ値**
- **固有** TP **レジスタ値**

9) **メモリ・プール情報**
個々のメモリ・プールに対して，次のデータを定義します。

- ID 番号
- **拡張情報**
- **タスクのキューイング方式**
- **メモリ・プール情報**
  メモリ・プールのサイズ
  メモリ・プールを割り付けるシステム・メモリの種類
- **キー** ID **番号**

10) **周期起動ハンドラ情報**
個々の周期起動ハンドラに対して，次のデータを定義します。

- **指定番号**
- **拡張情報**
- **記述言語**
- **起動アドレス**
- **初期活性状態**
- **起動時間間隔**
- **固有** GP **レジスタ値**
- **固有** TP **レジスタ値**

11) **拡張 SVC ハンドラ情報**
個々の拡張 SVC ハンドラに対して，次のデータを定義します。

- **拡張機能コード**
- **記述言語**
- **起動アドレス**
- **固有 GP レジスタ値**
- **固有 TP レジスタ値**

12) **初期化ハンドラ情報**
個々の初期化ハンドラに対して，次のデータを定義します。

- **記述言語**
- **起動アドレス**
- **固有 GP レジスタ値**
- **固有 TP レジスタ値**

6.3.3　SCT(System Call Table) 情報

システム・コンフィギュレーション・ファイルに記述する SCT 情報は，次に示す 8 項目です。

1) **タスク管理 / タスク付属同期管理機能システム・コール情報**
タスク管理 / タスク付属同期管理機能システム・コール情報として，ユーザ処理プログラム内で使用するタスク管理 / タスク付属同期管理機能システム・コールを定義します。
次に，RX850 Pro が提供しているタスク管理 / タスク付属同期管理機能システム・コールを示します。

cre_tsk	del_tsk	sta_tsk	ext_tsk	exd_tsk	ter_tsk
dis_dsp	ena_dsp	chg_pri	rot_rdq	rel_wai	get_tid
ref_tsk	vget_tid	sus_tsk	rsm_tsk	frsm_tsk	slp_tsk
tslp_tsk	wup_tsk	can_wup			

2) **同期通信 (セマフォ) 管理機能システム・コール情報**
同期通信 (セマフォ) 管理機能システム・コール情報として，ユーザ処理プログラム内で使用する同期通信 (セマフォ) 管理機能システム・コールを定義します。
次に，RX850 Pro が提供している同期通信 (セマフォ) 管理機能システム・コールを示します。

cre_sem	del_sem	sig_sem	preq_sem	wai_sem	twai_sem
ref_sem	vget_sid				

3) **同期通信 (イベントフラグ) 管理機能システム・コール情報**
同期通信 (イベントフラグ) 管理機能システム・コール情報として，ユーザ処理プログラム内で使用する同期通信 (イベントフラグ) 管理機能システム・コールを定義します。
次に，RX850 Pro が提供している同期通信 (イベントフラグ) 管理機能システム・コールを示します。

cre_flg	del_flg	set_flg	clr_flg	wai_flg	pol_flg
twai_flg	ref_flg	vget_fid			

4) **同期通信 (メールボックス) 管理機能システム・コール情報**
同期通信 (メールボックス) 管理機能システム・コール情報として，ユーザ処理プログラム内で使用する同期通信 (メールボックス) 管理機能システム・コールを定義します。
次に，RX850 Pro が提供している同期通信 (メールボックス) 管理機能システム・コールを示します。

cre_mbx	del_mbx	snd_msg	rcv_msg	prcv_msg	trcv_msg
ref_mbx	vget_mid				

5) **割り込み処理管理機能システム・コール情報**
割り込み処理管理機能システム・コール情報として，ユーザ処理プログラム内で使用する割り込み処理管理システム・コールを定義します。
次に，RX850 Pro が提供している割り込み処理管理機能システム・コールを示します。

def_int	ret_int	ret_wup	ena_int	dis_int	loc_cpu
unl_cpu	chg_icr	ref_icr			

6) **メモリ・プール管理機能システム・コール情報**

メモリ・プール管理機能システム・コール情報として，ユーザ処理プログラム内で使用するメモリ・プール管理機能システム・コールを定義します。
次に，RX850 Pro **が提供しているメモリ・プール管理機能システム・コールを示します。**

cre_mpl	del_mpl	get_blk	pget_blk	tget_blk	rel_blk
ref_mpl	vget_pid				

7) **時間管理機能システム・コール情報**

時間管理機能システム・コール機能情報として，ユーザ処理プログラム内で使用する時間処理管理システム・コール機能を定義します。
次に，RX850 Pro **が提供している時間管理機能システム・コールを示します。**

set_tim	get_tim	dly_tsk	def_cyc	act_cyc	ref_cyc

8) **システム管理機能システム・コール情報**

システム管理機能システム・コール情報として，ユーザ処理プログラム内で使用するシステム管理機能システム・コールを定義します。
次に，RX850 Pro **が提供しているシステム管理機能システム・コールを示します。**

get_ver	ref_sys	def_svc	viss_svc

6.4　リアルタイム OS 情報の記述形式

システム・コンフィギュレーション・ファイルに記述するリアルタイム OS 情報の記述形式を示します。
表記中の太字は予約語であることを，イタリック書体はユーザが該当する数値，シンボル名，キー・ワードを記述する部分であることを表しています。

6.4.1　RX シリーズ情報

RX シリーズ情報では，使用するリアルタイム OS の「リアルタイム OS 名，バージョン番号」を定義します。
なお，システム・コンフィギュレーション・ファイルに RX シリーズ情報を記述することは必須です。
以下に，RX シリーズ情報の記述形式を示します。

rxsers	*rtos_nam*	*rtos_ver*

次に，RX シリーズ情報で記述する各項目について示します。

- *rtos_nam*
  リアルタイム OS の名前 (リアルタイム OS 名) を指定します。
  なお，*rtos_nam* として指定可能なキー・ワードは，RX850PRO に限られます。

- *rtos_ver*
  リアルタイム OS のバージョン番号を指定します。
  なお，*rtos_ver* として指定可能なキー・ワードは，V32x (x は任意の数字) に限られます。

6.5　SIT 情報の記述形式

システム・コンフィギュレーション・ファイルに記述する SIT 情報の記述形式を示します。
表記中の太字は予約語であることを，イタリック書体はユーザが該当する数値，シンボル名，キー・ワードを記述する部分であることを表しています。

6.5.1　システム情報

システム情報では，「プロセッサ種別，基本クロック周期，タイマの割り込み要因番号，デフォルト・スタック・サイズ，割り込みハンドラ用スタック情報，ID 番号の保護範囲」を定義します。
なお，システム・コンフィギュレーション・ファイルにシステム情報を記述することは必須です。
以下に，システム情報の記述形式を示します。

[**cputype**	*chip_type*]
clktim	*time*
clkhdr	*clk_intno*
defstk	*stk_siz*
intstk	*intstk_siz* : *mem_nam*
prttsk	*tsk_idlmt*
prtsem	*sem_idlmt*
prtflg	*flg_idlmt*
prtmbx	*mbx_idlmt*
prtmpl	*mpl_idlmt*

次に，システム情報で記述する各項目について示します。

- *chip_type*

ターゲット・デバイスのプロセッサ種別を指定します。
なお，*chip_type* として指定可能なキー・ワードは，V850，V850E1，V850E2，V850ES に限られます。

V850	：V850 コア
V850E1	：V850E1 コア
V850E2	：V850E2 コア
V850ES	：V850ES コア

省略時　ターゲット・デバイスのプロセッサ種別は，“ V850E1 ” となります。

- *time*

使用するタイマの基本クロック周期 (単位：ms) を指定します。
なお，*time* として指定可能な値は，0x1 ～ 0x7fff に限られます。

備考　基本クロック周期とは，RX850 Pro が時間管理機能 (周期起床，遅延起床，タイムアウト等) を実現するうえで必要になるタイマ割り込みの発生周期を意味します。

- *clk_intno*

タイマの割り込み要因番号を指定します。
なお，*clk_intno* として指定可能な値は，使用する C コンパイラ・パッケージにより異なります。

【 CA850 対応版 】
デバイス・ファイルで規定されている割り込み要因名，または，“ (例外コード - 0x80) / 0x10 ” で計算される値に限られます。

【 GHS 製コンパイラ対応版 】
“ (例外コード - 0x80) / 0x10 ” で計算される値に限られます。

- *stk_siz*

デフォルト・スタック・サイズ (単位：バイト) を指定します。
なお，*stk_siz* として指定可能な値は，0x0 ～ 0x7ffffffc の 4 バイト境界値に限られます。

備考　デフォルト・スタック・サイズとは，システム内で存在可能なタスク用スタックの最小サイズを意味します。したがって，システム起動時に行われる静的なタスクの生成や，cre_tsk システム・コールの発行により行われる動的なタスクの生成において，デフォルト・スタック・サイズ以下のスタック・サイズが指定さ

れた場合には，そのサイズは無視され，デフォルト・スタック・サイズがスタック・サイズとして使用されます。

- *intstk_siz* : *mem_nam*

割り込みハンドラ用スタックのサイズ (単位：バイト)，および，割り込みハンドラ用スタックを割り付けるシステム・メモリの種類を指定します。
なお，*intstk_siz* として指定可能な値は，0x0 ～ 0x7ffffffc の 4 バイト境界値に限られます。
また，*mem_nam* として指定可能なキー・ワードは，SPOL0，SPOL1 に限られます。

SPOL0	： System Memory Pool 0 に割り込みハンドラ用スタックを割り付けます。
SPOL1	： System Memory Pool 1 に割り込みハンドラ用スタックを割り付けます。

- *tsk_idlmt*

タスクの ID 番号無指定生成が行われた際に保護する ID 番号の範囲を指定します。
なお，*tsk_idlmt* として指定可能な範囲は，0x0 ～ *tsk_cnt* に限られます。

注意　*tsk_idlmt* に 0x0 を指定した場合，ID 番号を保護しません。*tsk_cnt* はシステム最大値情報のタスクの最大生成数 maxtsk に定義された値となります。

- *sem_idlmt*

セマフォの ID 番号無指定生成が行われた際に保護する ID 番号の範囲を指定します。
なお，*sem_idlmt* として指定可能な範囲は，0x0 ～ *sem_cnt* に限られます。

注意　*sem_idlmt* に 0x0 を指定した場合，ID 番号を保護しません。*sem_cnt* はシステム最大値情報のセマフォの最大生成数 maxsem に定義された値となります。

- *flg_idlmt*

イベントフラグの ID 番号無指定生成が行われた際に保護する ID 番号の範囲を指定します。
なお，*flg_idlmt* として指定可能な範囲は，0x0 ～ flg_cnt に限られます。

注意　*flg_idlmt* に 0x0 を指定した場合，ID 番号を保護しません。*flg_cnt* はシステム最大値情報のイベントフラグの最大生成数 maxflg に定義された値となります。

- *mbx_idlmt*

メールボックスの ID 番号無指定生成が行われた際に保護する ID 番号の範囲を指定します。
なお，*mbx_idlmt* として指定可能な範囲は，0x0 ～ *mbx_cnt* に限られます。

注意　*mbx_idlmt* に 0x0 を指定した場合，ID 番号を保護しません。*mbx_cnt* はシステム最大値情報のメールボックスの最大生成数 maxmbx に定義された値となります。

- *mpl_idlmt*

メモリ・プールの ID 番号無指定生成が行われた際に保護する ID 番号の範囲を指定します。
なお，*mpl_idlmt* として指定可能な範囲は，0x0 ～ *mpl_cnt* に限られます。

注意　*mpl_idlmt* に 0x0 を指定した場合，ID 番号を保護しません。*mpl_cnt* はシステム最大値情報のメモリ・プールの最大生成数 maxmpl に定義された値となります。

6.5.2　システム最大値情報

システム最大値情報では，「タスクの優先度範囲，管理オブジェクトの最大生成数 / 最大登録数」を定義します。
なお，システム・コンフィギュレーション・ファイルにシステム最大値情報を記述することは必須です。
以下に，システム最大値情報の記述形式を示します。

maxpri	*pri_lvl*
maxtsk	*tsk_cnt*
maxsem	*sem_cnt*
maxflg	*flg_cnt*
maxmbx	*mbx_cnt*
maxmpl	*mpl_cnt*
maxcyc	*cyc_cnt*
maxsvc	*svc_cnt*
maxint	*ith_cnt*
maxintfactor	*itf_cnt*

次に，システム最大値情報で記述する各項目について示します。

- *pri_lvl*
  タスクの優先度範囲 (優先度数) を指定します。
  なお，*pri_lvl* として指定可能な値は，0x1 ～ 0xfc に限られます。

- *tsk_cnt*
  タスクの最大生成数を指定します。
  なお，*tsk_cnt* として指定可能な値は，0x1 ～ 0x7fff に限られます。

- *sem_cnt*
  セマフォの最大生成数を指定します。
  なお，*sem_cnt* として指定可能な値は，0x0 ～ 0x7fff に限られます。

- *flg_cnt*
  イベントフラグの最大生成数を指定します。
  なお，*flg_cnt* として指定可能な値は，0x0 ～ 0x7fff に限られます。

- *mbx_cnt*
  メールボックスの最大生成数を指定します。
  なお，*mbx_cnt* として指定可能な値は，0x0 ～ 0x7fff に限られます。

- *mpl_cnt*
  メモリ・プールの最大生成数を指定します。
  なお，*mpl_cnt* として指定可能な値は，0x0 ～ 0x7fff に限られます。

- *cyc_cnt*
  周期起動ハンドラの最大登録数を指定します。
  なお，*cyc_cnt* として指定可能な値は，0x0 ～ 0x7fff に限られます。

- *svc_cnt*
  拡張 SVC ハンドラの最大登録数を指定します。
  なお，*svc_cnt* として指定可能な値は，0x0 ～ 0x7fff に限られます。

- *ith_cnt*
  割り込みハンドラの最大登録数を指定します。
  なお，*ith_cnt* として指定可能な値は，0x0 ～ *itf_cnt* + 1 に限られます。

- *itf_cnt*
  割り込み要因番号の最大値を指定します。
  なお，*itf_cnt* として指定可能な値は，“ (対象プロセッサで既定されている例外コードの最大値 - 0x80) / 0x10 ” で計算される値に限られます。

6.5.3　システム・メモリ情報

システム・メモリ情報では，「システム・メモリの種類，セクション名，サイズ」を個々のシステム・メモリ (System Memory Pool 0，System Memory Pool 1，User Memory Pool 0，User Memory Pool 1) に対して定義します。

なお，システム・コンフィギュレーション・ファイルに System Memory Pool 0 に関するデータを記述することは必須です。

また，システム・コンフィギュレーション・ファイルに User Memory Pool 1 に関するデータを記述する場合，User Memory Pool 0 に関するデータを記述することは必須です。

以下に，システム・メモリ情報の記述形式を示します。

mem	*mem_id*	*sec_nam*	*mem_siz*

次に，システム・メモリ情報で記述する各項目について示します。

- *mem_id*

  システム・メモリの種類を指定します。
  なお，*mem_id* として指定可能なキー・ワードは，SPOL0，SPOL1，UPOL0，UPOL1 のいずれかに限られます。

  SPOL0　：システム・メモリの種類は，System Memory Pool 0 となります。
  SPOL1　：システム・メモリの種類は，System Memory Pool 1 となります。
  UPOL0　：システム・メモリの種類は，User Memory Pool 0 となります。
  UPOL1　：システム・メモリの種類は，User Memory Pool 1 となります。

- *sec_nam*

  システム・メモリが割り付けられるメモリ領域のセクション名を指定します。
  なお，*sec_nam* として指定可能な値は“ リンク・ディレクティブ・ファイルで定義されているセクション名 *.sec_nam* から . を除いた名前 ” に限られます。

- *mem_siz*

  システム・メモリのサイズ (単位：バイト) を指定します。
  なお，*mem_siz* として指定可能な値は，0x100 ～ 0x7ffffffc の 4 バイト境界値に限られます。

6.5.4　タスク情報

タスク情報では，「ID 番号，初期状態，起動コード，拡張情報，記述言語，起動アドレス，初期優先度，割り込み状態，タスク用スタック情報，固有 GP レジスタ値，固有 TP レジスタ値，キー ID 番号」を個々のタスクに対して定義します。
なお，システム・コンフィギュレーション・ファイルにタスク情報は最低 1 個必要です。
また，タスク情報を定義可能な数は，1 ～システム最大値情報で指定したタスクの最大生成数 *tsk_cnt* の範囲に限られます。
以下に，タスク情報の記述形式を示します。

```
tsk     tsk_id      sts         sta_code    ext_inf     lang    \
        sta_adr     pri         intr        stk_siz : mem_nam   \
        data        text        key_id
```

次に，タスク情報で記述する各項目について示します。

- *tsk_id*

  タスクの ID 番号を指定します。
  なお，*tsk_id* として指定可能な値は，0x0 ～ *tsk_cnt*，シンボル名に限られます。
  tsk_id に 0x0，または，シンボル名を指定した場合，CF850 Pro が *tsk_idlmt* ～ *tsk_cnt* の範囲で未使用の ID 番号を自動的に割り当てます。
  なお，*tsk_idlmt* は，システム情報のタスクの ID 番号保護範囲 prttsk に定義された値となります。
  また，*tsk_cnt* は，システム最大値情報のタスクの最大生成数 maxtsk に定義された値となります。

- *sts*

  タスクの初期状態を指定します。
  なお，*sts* として指定可能なキー・ワードは，TTS_DMT，TTS_RDY のいずれかに限られます。

  TTS_DMT　：システム起動時，dormant 状態へと遷移します。
  TTS_RDY　：システム起動時，ready 状態へと遷移します。

  注意　すべての静的タスクの初期状態を TTS_DMT にすると，システム起動時に動作しているタスクがないことになってしまいます。この場合，初期化ハンドラで適当なタスクを起動する必要があります。

- *sta_code*

  タスクの起動コードを指定します。
  なお，*sta_code* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*sta_code* は，sts に TTS_RDY を指定した場合にのみ有効となり，TTS_DMT を指定した場合には無効となります。

- *ext_inf*

  タスクの拡張情報を指定します。
  なお，*ext_inf* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*ext_inf* は，対象タスクに関する “ ユーザ独自の情報 ” を指定するためのものであり，ユーザが自由に指定できます。
  なお，*ext_inf* に指定された値は，処理プログラム (タスク / 非タスク) から ref_tsk システム・コールを発行することにより，動的に獲得できます。

- *lang*

  タスクの記述言語を指定します。
  なお，*lang* として指定可能なキー・ワードは，TA_HLNG，TA_ASM のどちらかに限られます。

  TA_HLNG　：タスクを C 言語で記述します。
  TA_ASM　：タスクをアセンブリー言語で記述します。

- *sta_adr*

  タスクの起動アドレスを指定します。
  なお，*sta_adr* として指定可能な値は，0x0 ～ 0xfffffffe の 2 バイト境界値，または，シンボル名に限られます。

- *pri*

  タスクの初期優先度を指定します。
  なお，*pri* として指定可能な値は，0x1 **～ *pri_lvl* に限られます。**
  また，pri_lvl **は，システム最大値情報のタスクの優先度範囲** maxpri **に定義された値となります。**

- *intr*

  タスク起動時の割り込み状態を指定します。
  なお，*intr* として指定可能なキー・ワードは，TA_ENAINT**，**TA_DISINT **のどちらかに限られます。**

  TA_ENAINT　：**タスク起動時，すべてのマスカブル割り込みを許可します。**
  TA_DISINT　：**タスク起動時，すべてのマスカブル割り込みを禁止します。**

- *stk_siz* : *mem_nam*

  タスク用スタックのサイズ (単位：バイト)，および，タスク用スタックを割り付けるシステム・メモリの種類を指定します。
  なお，*stk_siz* として指定可能な値は，0x0 **～** 0x7ffffffc **の** 4 **バイト境界値に限られます。**
  また，*mem_nam* として指定可能なキー・ワードは，SPOL0**，**SPOL1 **のいずれかに限られます。**

  SPOL0　：**システム・メモリの種類は，**System Memory Pool 0 **となります。**
  SPOL1　：**システム・メモリの種類は，**System Memory Pool 1 **となります。**

- *data*

  タスクの固有 GP **レジスタ値を指定します。**
  なお，*data* として指定可能な値 / キー・ワードは，0x0 **～** 0xffffffff**，シンボル名，**no_use **に限られます。**

  no_use　：**固有** GP **レジスタ値の設定は行われません。**

- *text*

  タスクの固有 TP **レジスタ値を指定します。**
  なお，*text* として指定可能な値 / キー・ワードは，0x0 **～** 0xffffffff**，シンボル名，**no_use **に限られます。**

  no_use　：**固有** TP **レジスタ値の設定は行われません。**

- *key_id*

  タスクのキー ID **番号を指定します。**
  なお，*key_id* として指定可能な値は，0x0 **～** 0x7fff **に限られます。**

  注意　*key_id* **に** 0x0 **を設定した場合，**CF850 Pro **はキー** ID **番号の割り付けを行いません。**

6.5.5　セマフォ情報

セマフォ情報では，「ID 番号，拡張情報，タスクのキューイング方式，初期資源数，最大資源数，キー ID 番号」を個々のセマフォに対して定義します。
なお，セマフォ情報として定義可能な数は，0 ～システム最大値情報で登録したセマフォの最大生成数 *sem_cnt* の範囲に限られます。
以下に，セマフォ情報の記述形式を示します。

sem	*sem_id*	*ext_inf*	*twai_opt*	*init_cnt*	*max_cnt*	\
	key_id					

次に，セマフォ情報で記述する各項目について示します。

- *sem_id*
  セマフォの ID 番号を指定します。
  なお，*sem_id* として指定可能な値は，0x0 ～ *sem_cnt*，シンボル名に限られます。
  sem_id に 0x0，または，シンボル名を指定した場合，CF850 Pro が *sem_idlmt* ～ *sem_cnt* の範囲で未使用の ID 番号を自動的に割り当てます。
  なお，*sem_idlmt* は，システム情報のセマフォの ID 番号保護範囲 prtsem に定義された値となります。
  また，*sem_cnt* は，システム最大値情報のセマフォの最大生成数 maxsem に定義された値となります。

- *ext_inf*
  セマフォの拡張情報を指定します。
  なお，*ext_inf* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*ext_inf* は，対象セマフォに関する “ ユーザ独自の情報 ” を指定するためのものであり，ユーザが自由に指定できます。
  なお，*ext_inf* に指定された値は，処理プログラム (タスク / 非タスク) から ref_sem システム・コールを発行することにより，動的に獲得できます。

- *twai_opt*
  タスクのキューイング方式を指定します。
  なお，*twai_opt* として指定可能なキー・ワードは，TA_TFIFO，TA_TPRI のいずれかに限られます。

  TA_TFIFO　：タスクのキューイング方式は，資源の獲得要求を行った順になります。
  TA_TPRI　：タスクのキューイング方式は，タスクの優先度順になります。

- *init_cnt*
  セマフォの初期資源数を指定します。
  なお，*init_cnt* として指定可能な数値は，0x0 ～ *max_cnt* に限られます。

- *max_cnt*
  セマフォの最大資源数を指定します。
  なお，*max_cnt* として指定可能な数値は，0x1 ～ 0x7fffffff に限られます。

- *key_id*
  セマフォのキー ID 番号を指定します。
  なお，*key_id* として指定可能な値は，0x0 ～ 0x7fff に限られます。

  注意　*key_id* に 0x0 を設定した場合，CF850 Pro はキー ID 番号の割り付けを行いません。

6.5.6　イベントフラグ情報

イベントフラグ情報では，「ID 番号，拡張情報，複数タスク待ちの許可 / 禁止，初期ビット・パターン，キー ID 番号」を個々のイベントフラグに対して定義します。
なお，イベントフラグ情報として定義可能な数は，0 ～システム最大値情報で登録したイベントフラグの最大生成数 *flg_cnt* に限られます。
以下に，イベントフラグ情報の記述形式を示します。

flg	*flg_id*	*ext_inf*	*twai_opt*	*init_ptn*	*key_id*

次に，イベントフラグ情報で記述する各項目について示します。

- *fig_id*
  イベントフラグの ID 番号を指定します。
  なお，*fig_id* として指定可能な値は，0x0 ～ *flg_cnt*，シンボル名に限られます。
  flg_id に 0x0，または，シンボル名を指定した場合，CF850 Pro が *flg_idlmt* ～ *flg_cnt* の範囲で未使用の ID 番号を自動的に割り当てます。
  なお，*flg_idlmt* は，システム情報のイベントフラグの ID 番号保護範囲 prtflg に定義された値となります。
  また，*flg_cnt* は，システム最大値情報のイベントフラグの最大生成数 maxflg に定義された値となります。

- *ext_inf*
  イベントフラグの拡張情報を指定します。
  なお，*ext_inf* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*ext_inf* は，対象イベントフラグに関する “ ユーザ独自の情報 ” を指定するためのものであり，ユーザが自由に指定できます。
  なお，*ext_inf* に指定された値は，処理プログラム (タスク / 非タスク) から ref_flg システム・コールを発行することにより，動的に獲得できます。

- *twai_opt*
  複数タスク待ちの許可 / 禁止を指定します。
  なお，*twai_opt* として指定可能なキー・ワードは，TA_WSGL，TA_WMUL のいずれかに限られます。

  TA_WSGL　：複数タスク待ちは禁止になります。
  TA_WMUL　：複数タスク待ちは許可になります。

- *init_ptn*
  イベントフラグの初期ビット・パターン (32 ビット幅) を指定します。
  なお，*init_ptn* として指定可能な値は，0x0 ～ 0xffffffff に限られます。

- *key_id*
  イベントフラグのキー ID 番号を指定します。
  なお，*key_id* として指定可能な値は，0x0 ～ 0x7fff に限られます。

  注意　*key_id* に 0x0 を設定した場合，CF850 Pro はキー ID 番号の割り付けを行いません。

6.5.7　メールボックス情報

メールボックス情報では，「ID 番号，拡張情報，タスクのキューイング方式，メッセージのキューイング方式，キー ID 番号」を個々のメールボックスに対して定義します。
なお，メールボックス情報として定義可能な数は，0 ～システム最大値情報で登録したメールボックスの最大生成数 *mbx_cnt* に限られます。
以下に，メールボックス情報の記述形式を示します。

mbx	*mbx_id*	*ext_inf*	*twai_opt*	*mwai_opt*	*key_id*

次に，メールボックス情報で記述する各項目について示します。

- *mbx_id*
  メールボックスの ID 番号を指定します。
  なお，*mbx_id* として指定可能な値は，0x0 ～ *mbx_cnt*，シンボル名に限られます。
  mbx_id に 0x0，または，シンボル名を指定した場合，CF850 Pro が *mbx_idlmt* ～ *mbx_cnt* の範囲で未使用の ID 番号を自動的に割り当てます。
  なお，*mbx_idlmt* は，システム情報のメールボックスの ID 番号保護範囲 prtmbx に定義された値となります。
  また，*mbx_cnt* は，システム最大値情報のメールボックスの最大生成数 maxmbx に定義された値となります。

- *ext_inf*
  メールボックスの拡張情報を指定します。
  なお，*ext_inf* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*ext_inf* は，対象メールボックスに関する “ ユーザ独自の情報 ” を指定するためのものであり，ユーザが自由に指定できます。
  なお，*ext_inf* に指定された値は，処理プログラム (タスク / 非タスク) から ref_mbx システム・コールを発行することにより，動的に獲得できます。

- *twai_opt*
  タスクのキューイング方式を指定します。
  なお，*twai_opt* として指定可能なキー・ワードは，TA_TFIFO，TA_TPRI のいずれかに限られます。

  TA_TFIFO　：タスクのキューイング方式は，メッセージの受信要求を行った順になります。
  TA_TPRI　：タスクのキューイング方式は，タスクの優先度順になります。

- *mwai_opt*
  メッセージのキューイング方式を指定します。
  なお，*mwai_opt* として指定可能なキー・ワードは，TA_MFIFO，TA_MPRI のいずれかに限られます。

  TA_MFIFO　：メッセージのキューイング方式は，メッセージの送信を行った順になります。
  TA_MPRI　：メッセージのキューイング方式は，メッセージの優先度順になります。

- *key_id*
  メールボックスのキー ID 番号を指定します。
  なお，*key_id* として指定可能な値は，0x0 ～ 0x7fff に限られます。

  注意　*key_id* に 0x0 を設定した場合，CF850 Pro はキー ID 番号の割り付けを行いません。

6.5.8　割り込みハンドラ情報

割り込みハンドラ情報では，「割り込み要因番号，記述言語，起動アドレス，固有 GP レジスタ値，固有 TP レジスタ値」を個々の割り込みハンドラに対して定義します。
なお，割り込みハンドラ情報として定義可能な数は，0 ～システム最大値情報で登録した割り込みハンドラの最大生成数 *ith_cnt* に限られます。
以下に，割り込みハンドラ情報の記述形式を示します。

inthdr	*int_no*	*lang*	*hdr_adr*	*data*	*text*

次に，割り込みハンドラ情報で記述する各項目について示します。

- *int_no*
  割り込みハンドラの割り込み要因番号を指定します。
  なお，*int_no* として指定可能な値は，使用する C コンパイラ・パッケージにより異なります。

  【 CA850 対応版 】
  デバイス・ファイルで規定されている割り込み要因名，または，“ (例外コード - 0x80) / 0x10 ” で計算される値に限られます。

  【 GHS 製コンパイラ対応版 】
  “ (例外コード - 0x80) / 0x10 ” で計算される値に限られます。

  注意　*int_no* に *clk_intno* と同じ割り込み要因番号を指定することはできません。
  なお，*clk_intno* は，システム情報のタイマの割り込み要因番号 clkhdr に定義された値となります。

- *lang*
  割り込みハンドラの記述言語を指定します。
  なお，*lang* として指定可能なキー・ワードは，TA_HLNG，TA_ASM のどちらかに限られます。

  TA_HLNG　：割り込みハンドラを C 言語で記述します。
  TA_ASM　：割り込みハンドラをアセンブリー言語で記述します。

- *hdr_adr*
  割り込みハンドラの起動アドレスを指定します。
  なお，*hdr_adr* として指定可能な値は，0x0 ～ 0xfffffffe の 2 バイト境界値，シンボル名に限られます。

- *data*
  割り込みハンドラの固有 GP レジスタ値を指定します。
  なお，*data* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 GP レジスタ値の設定は行われません。

- *text*
  割り込みハンドラの固有 TP レジスタ値を指定します。
  なお，*text* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 TP レジスタ値の設定は行われません。

6.5.9　メモリ・プール情報

メモリ・プール情報では，「ID 番号，拡張情報，タスクのキューイング方式，メモリ・プール情報，キー ID 番号」を個々のメモリ・プールに対して定義します。
なお，メモリ・プール情報として定義可能な数は，0 ～システム最大値情報で登録したメモリ・プールの最大生成数 *mpl_cnt* に限られます。
以下に，メモリ・プール情報の記述形式を示します。

mpl	*mpl_id*	*ext_inf*	*twai_opt*	*mpl_siz* : *mem_nam*	\
	key_id				

次に，メモリ・プール情報で記述する各項目について示します。

- *mpl_id*
  メモリ・プールの ID 番号を指定します。
  なお，*mpl_id* として指定可能な値は，0x0 ～ *mpl_cnt*，シンボル名に限られます。
  mpl_id に 0x0，または，シンボル名を指定した場合，CF850 Pro が *mpl_idlmt* ～ *mpl_cnt* の範囲で未使用の ID 番号を自動的に割り当てます。
  なお，*mpl_idlmt* は，システム情報のメモリ・プールの ID 番号保護範囲 prtmpl に定義された値となります。
  また，*mpl_cnt* は，システム最大値情報のメモリ・プールの最大生成数 maxmpl に定義された値となります。

- *ext_inf*
  メモリ・プールの拡張情報を指定します。
  なお，*ext_inf* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*ext_inf* は，対象メモリ・プールに関する “ ユーザ独自の情報 ” を指定するためのものであり，ユーザが自由に指定できます。
  なお，*ext_inf* に指定された値は，処理プログラム (タスク / 非タスク) から ref_mpl システム・コールを発行することにより，動的に獲得できます。

- *twai_opt*
  タスクのキューイング方式を指定します。
  なお，*twai_opt* として指定可能なキー・ワードは，TA_TFIFO，TA_TPRI のいずれかに限られます。

  TA_TFIFO　：タスクのキューイング方式は，メモリ・ブロックの要求を行った順になります。
  TA_TPRI　：タスクのキューイング方式は，タスクの優先度順になります。

- *mpl_siz* : *mem_nam*
  メモリ・プールのサイズ (単位：バイト),および,メモリ・プールを割り付けるシステム・メモリの種類を指定します。
  なお，*mpl_siz* として指定可能な値は，0x4 ～ 0x7ffffffc の 4 バイト境界値に限られます。
  また，*mem_nam* として指定可能なキー・ワードは，UPOL0，UPOL1 のいずれかに限られます。

  UPOL0　：システム・メモリの種類は，User Memory Pool 0 となります。
  UPOL1　：システム・メモリの種類は，User Memory Pool 1 となります。

- *key_id*
  メモリ・プールのキー ID 番号を指定します。
  なお，*key_id* として指定可能な値は，0x0 ～ 0x7fff に限られます。

  注意　*key_id* に 0x0 を設定した場合，CF850 Pro はキー ID 番号の割り付けを行いません。

6.5.10　周期起動ハンドラ情報

周期起動ハンドラ情報では，「指定番号，拡張情報，記述言語，起動アドレス，初期活性状態，起動時間間隔，固有 GP レジスタ値，固有 TP レジスタ値」を個々の周期起動ハンドラに対して定義します。

なお，周期起動ハンドラ情報として定義可能な数は，0 ～システム最大値情報で登録した周期起動ハンドラの最大登録数 *cyc_cnt* の範囲に限られます。

以下に，周期起動ハンドラ情報の記述形式を示します。

cyc	*cyc_no*	*ext_inf*	*lang*	*hdr_adr*	*act*	\
	intvl	*data*	*text*			

次に，周期起動ハンドラ情報で記述する各項目について示します。

- *cyc_no*

  周期起動ハンドラの指定番号を指定します。
  なお，*cyc_no* として指定可能な値は，0x1 ～ *cyc_cnt*，シンボル名に限られます。
  *cyc_no*にシンボル名を指定した場合,CF850 Proが0x1～*cyc_cnt*の範囲で未使用のID番号を自動的に割り当てます。
  なお，*cyc_cnt* は，システム最大値情報の周期起動ハンドラの最大登録数 maxcyc に定義された値となります。

- *ext_inf*

  周期起動ハンドラの拡張情報を指定します。
  なお，*ext_inf* として指定可能な値は，0x0 ～ 0xffffffff，シンボル名に限られます。

  注意　*ext_inf* は，対象周期起動ハンドラに関する “ ユーザ独自の情報 ” を指定するためのものであり，ユーザが自由に指定できます。
  なお，*ext_inf* に指定された値は，処理プログラム (タスク / 非タスク) から ref_cyc システム・コールを発行することにより，動的に獲得できます。

- *lang*

  周期起動ハンドラの記述言語を指定します。
  なお，*lang* として指定可能なキー・ワードは，TA_HLNG，TA_ASM のどちらかに限られます。

  TA_HLNG　：周期起動ハンドラを C 言語で記述します。
  TA_ASM　：周期起動ハンドラをアセンブリー言語で記述します。

- *hdr_adr*

  周期起動ハンドラの起動アドレスを指定します。
  なお，*hdr_adr* として指定可能な値は，0 ～ 0xfffffffe の 2 バイト境界値，シンボル名に限られます。

- *act*

  周期起動ハンドラの初期活性状態を指定します。
  なお，*act* として指定可能なキー・ワードは，TCY_ON，TCY_OFF のどちらかに限られます。

  TCY_ON　：システム起動時，活性状態は ON 状態になります。
  TCY_OFF　：システム起動時，活性状態は OFF 状態になります。

- *intvl*

  周期起動ハンドラの起動時間間隔 (単位：基本クロック周期) を指定します。
  なお，*intvl* として指定可能な値は，0x1 ～ 0xffffffff に限られます。

- *data*

  周期起動ハンドラの固有 GP レジスタ値を指定します。
  なお，*data* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 GP レジスタ値の設定は行われません。

- *text*

  周期起動ハンドラの固有 TP レジスタ値を指定します。
  なお，*text* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 TP レジスタ値の設定は行われません。

6.5.11　拡張 SVC ハンドラ情報

拡張 SVC ハンドラ情報では，「拡張機能コード，記述言語，起動アドレス，固有 GP レジスタ値，固有 TP レジスタ値」を個々の拡張 SVC ハンドラに対して定義します。
なお，拡張 SVC ハンドラ情報として定義可能な数は，0 ～システム最大値情報で登録した拡張 SVC ハンドラの最大登録数 *scv_cnt* に限られます。
以下に，拡張 SVC ハンドラ情報の記述形式を示します。

svc	*svc_no*	*lang*	*hdr_adr*	*data*	*text*

次に，拡張 SVC ハンドラ情報で記述する各項目について示します。

- *svc_no*
  拡張 SVC ハンドラの拡張機能コードを指定します。
  なお，*svc_no* として指定可能な値は，0x1 ～ *svc_cnt*，シンボル名に限られます。
  *svc_no*にシンボル名を指定した場合,CF850 Proが0x1～*svc_cnt*の範囲で未使用のID番号を自動的に割り当てます。
  なお，*svc_cnt* は，システム最大値情報の拡張 SVC ハンドラの最大登録数 maxsvc に定義された値となります。

- *lang*
  拡張 SVC ハンドラの記述言語を指定します。
  なお，*lang* として指定可能なキー・ワードは，TA_HLNG，TA_ASM のどちらかに限られます。

  TA_HLNG　：拡張 SVC ハンドラを C 言語で記述します。
  TA_ASM　：拡張 SVC ハンドラをアセンブリー言語で記述します。

- *hdr_adr*
  拡張 SVC ハンドラの起動アドレスを指定します。
  なお，*hdr_adr* として指定可能な値は，0x0 ～ 0xfffffffe の 2 バイト境界値，シンボル名に限られます。

- *data*
  拡張 SVC ハンドラの固有 GP レジスタ値を指定します。
  なお，*data* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 GP レジスタ値の設定は行われません。

- *text*
  拡張 SVC ハンドラの固有 TP レジスタ値を指定します。
  なお，*text* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 TP レジスタ値の設定は行われません。

6.5.12　初期化ハンドラ情報

初期化ハンドラ情報では，初期化ハンドラの「記述言語，起動アドレス，固有 GP レジスタ値，固有 TP レジスタ値」を定義します。

なお，システム・コンフィギュレーション・ファイルにて，初期化ハンドラの情報を省略することが可能です。省略した場合，RX850 Pro は初期化ハンドラがないものと判断し，処理を続けます。

以下に，初期化ハンドラ情報の記述形式を示します。

ini	*lang*	*hdr_adr*	*data*	*text*

次に，初期化ハンドラ情報で記述する各項目について示します。

- *lang*

  初期化ハンドラの記述言語を指定します。
  なお，*lang* として指定可能なキー・ワードは，TA_HLNG，TA_ASM のどちらかに限られます。

  TA_HLNG　：初期化ハンドラを C 言語で記述します。
  TA_ASM　：初期化ハンドラをアセンブリー言語で記述します。

- *hdr_adr*

  初期化ハンドラの起動アドレスを指定します。
  なお，*hdr_adr* として指定可能な値は，0x0 ～ 0xfffffffe の 2 バイト境界値，シンボル名に限られます。

- *data*

  初期化ハンドラの固有 GP レジスタ値を指定します。
  なお，*data* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 GP レジスタ値の設定は行われません。

- *text*

  初期化ハンドラの固有 TP レジスタ値を指定します。
  なお，*text* として指定可能な値 / キー・ワードは，0x0 ～ 0xffffffff，シンボル名，no_use に限られます。

  no_use　：固有 TP レジスタ値の設定は行われません。

6.6　SCT 情報の記述形式

システム・コンフィギュレーション・ファイルに記述する SCT 情報の記述形式を示します。
表記中の太字は予約語であることを，イタリック書体はユーザが該当する数値，シンボル名，キー・ワードを記述する部分であることを表しています。

6.6.1　タスク管理 / タスク付属同期管理機能システム・コール情報

タスク管理 / タスク付属同期管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「タスク管理 / タスク付属同期管理機能システム・コール」を個々のシステム・コールに対して定義します。
ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。
以下に，タスク管理 / タスク付属同期管理機能システム・コール情報の記述形式を示します。

tsksvc	*svc_nam*

次に，タスク管理 / タスク付属同期管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  タスク管理機能システム・コール名を指定します。
  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

cre_tsk	del_tsk	sta_tsk	ext_tsk	exd_tsk	ter_tsk
dis_dsp	ena_dsp	chg_pri	rot_rdq	rel_wai	get_tid
ref_tsk	vget_tid	sus_tsk	rsm_tsk	frsm_tsk	slp_tsk
tslp_tsk	wup_tsk	can_wup			

6.6.2　同期通信 (セマフォ) 管理機能システム・コール情報

同期通信 (セマフォ) 管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「同期通信 (セマフォ) 管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，同期通信 (セマフォ) 管理機能システム・コール情報の記述形式を示します。

semsvc	*svc_nam*

次に，同期通信 (セマフォ) 管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  同期通信 (セマフォ) 管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

cre_sem	del_sem	sig_sem	preq_sem	wai_sem	twai_sem
ref_sem	vget_sid				

6.6.3　同期通信 (イベントフラグ) 管理機能システム・コール情報

同期通信 (イベントフラグ) 管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「同期通信 (イベントフラグ) 管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，同期通信 (イベントフラグ) 管理機能システム・コール情報の記述形式を示します。

flgsvc	*svc_nam*

次に，同期通信 (イベントフラグ) 管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  同期通信 (イベントフラグ) 管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

cre_flg	del_flg	set_flg	clr_flg	wai_flg	pol_flg
twai_flg	ref_flg	vget_fid			

6.6.4　同期通信 (メールボックス) 管理機能システム・コール情報

同期通信 (メールボックス) 管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「同期通信 (メールボックス) 管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，同期通信 (メールボックス) 管理機能システム・コール情報の記述形式を示します。

mbxsvc	*svc_nam*

次に，同期通信 (メールボックス) 管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  同期通信 (メールボックス) 管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

  cre_mbx　del_mbx　snd_msg　rcv_msg　prcv_msg　trcv_msg
  ref_mbx　vget_mid

6.6.5　割り込み処理管理機能システム・コール情報

割り込み処理管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「割り込み処理管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，割り込み処理管理機能システム・コール情報の記述形式を示します。

intsvc	*svc_nam*

次に，割り込み処理管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  割り込み処理管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

def_int	ret_int	ret_wup	ena_int	dis_int	loc_cpu
unl_cpu	chg_icr	ref_icr			

6.6.6　メモリ・プール管理機能システム・コール情報

メモリ・プール管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「メモリ・プール管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，メモリ・プール管理機能システム・コール情報の記述形式を示します。

mplsvc	*svc_nam*

次に，メモリ・プール管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  メモリ・プール管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

cre_mpl	del_mpl	get_blk	pget_blk	tget_blk	rel_blk
ref_mpl	vget_pid				

6.6.7　時間管理機能システム・コール情報

時間管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「時間管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，時間管理機能システム・コール情報の記述形式を示します。

timsvc	*svc_nam*

次に，時間管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  時間管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

  set_tim　get_tim　dly_tsk　def_cyc　act_cyc　ref_cyc

6.6.8　システム管理機能システム・コール情報

システム管理機能システム・コール情報では，ユーザ処理プログラム内で使用する「システム管理機能システム・コール」を個々のシステム・コールに対して定義します。

ここで定義せずにアプリケーションでシステム・コールを使用した場合，システム・コールの戻り値として E_NOSPT(-17) が入ります。

以下に，システム管理機能システム・コール情報の記述形式を示します。

syssvc	*svc_nam*

次に，システム管理機能システム・コールで記述する各項目について示します。

- *svc_nam*

  システム管理機能システム・コール名を指定します。

  なお，*svc_nam* として指定可能なキー・ワードは，次のものに限られます。

  get_ver　　ref_sys　　def_svc　　viss_svc

6.7　記述上の注意点

システム・コンフィギュレーション・ファイルにコンフィギュレーション情報 (リアルタイム OS 情報，SIT 情報，SCT 情報) を記述する場合，次の順序で行います。

1) リアルタイム OS 情報の記述が開始されることの宣言
2) リアルタイム OS 情報の記述
3) SIT(System Information Table) 情報の記述が開始されることの宣言
4) SIT(System Information Table) 情報の記述
5) SCT(System Call Table) 情報の記述が開始されることの宣言
6) SCT(System Call Table) 情報の記述

図 6-1 に，システム・コンフィギュレーション・ファイルの記述イメージを示します。

図 6-1　システム・コンフィギュレーション・ファイルの記述イメージ

```
-- リアルタイム OS 情報の記述が開始されることの宣言
ser_def

-- リアルタイム OS 情報の記述
……
……
……

-- SIT(System Information Table) 情報の記述が開始されることの宣言
sit_def

-- SIT(System Information Table) 情報の記述
……
……
……

-- SCT(System Call Table) 情報の記述が開始されることの宣言
sct_def

-- SCT(System Call Table) 情報の記述
……
……
……
```

6.8　記述例

図 6-2 に，システム・コンフィギュレーション・ファイルの記述例 (CA850 対応版) を示します。
図 6-2 の記述例では，次に示すデータが記述されています。

【 リアルタイム OS 情報 】

1) RX シリーズ情報

リアルタイム OS 名	： RX850PRO
バージョン番号	： V320

【 SIT(System Information Table) 情報 】

1) システム情報

プロセッサ種別	： V850E1
基本クロック周期	： 0x1 ms
タイマの割り込み要因番号	： 0x20 (INTCM4)
デフォルト・スタック・サイズ	： 0x100 バイト
割り込みハンドラ用スタック情報	： System Memory Pool 0 から 0x100 バイトのメモリ領域を確保
タスクの ID 番号保護範囲	： 0x1
セマフォの ID 番号保護範囲	： 0x1
イベントフラグの ID 番号保護範囲	： 0x1
メールボックスの ID 番号保護範囲	： 0x1
メモリ・プールの ID 番号保護範囲	： 0x1

2) システム最大値情報

タスクの優先度範囲	： 0xf
タスクの最大生成数	： 0x2
セマフォの最大生成数	： 0x1
イベントフラグの最大生成数	： 0x2
メールボックスの最大生成数	： 0x3
メモリ・プールの最大生成数	： 0x2
周期起動ハンドラの最大登録数	： 0x1
拡張 SVC ハンドラの最大登録数	： 0x1
割り込みハンドラの最大登録数	： 0x5
割り込み要因番号の最大値	： 0x48

3) システム・メモリ情報

System Memory Pool 0	： .syspol0 セクションから 0x2000 バイトのメモリ領域を確保
System Memory Pool 1	： .syspol1 セクションから 0x1000 バイトのメモリ領域を確保
User Memory Pool 0	： .usrpol0_0 セクションから 0x7000 バイトのメモリ領域を確保
User Memory Pool 0	： .usrpol0_1 セクションから 0x2500 バイトのメモリ領域を確保
User Memory Pool 1	： .usrpol1 セクションから 0x1500 バイトのメモリ領域を確保

4) タスク情報

ID 番号	： 0x1
初期状態	： ready 状態
起動コード	： 0x0
拡張情報	： 0x0
記述言語	： アセンブリー言語
起動アドレス	： _task01
初期優先度	： 0x8
割り込み状態	： すべてのマスカブル割り込みを許可
タスク用スタック情報	： System Memory Pool 0 から 0x100 バイトのメモリ容量を確保
固有 GP レジスタ値	： 設定しない
固有 TP レジスタ値	： 設定しない
キー ID 番号	： 0x1

ID 番号	： TASK02
初期状態	： dormant 状態
起動コード	： 0x0
拡張情報	： 0x0
記述言語	： C 言語
起動アドレス	： _task02

初期優先度　：0xf
割り込み状態　：すべてのマスカブル割り込みを禁止
タスク用スタック情報　：System Memory Pool 0 から 0x100 バイトのメモリ容量を確保
固有 GP レジスタ値　：設定しない
固有 TP レジスタ値　：設定しない
キー ID 番号　：0x2

5) セマフォ情報
ID 番号　：0x1
拡張情報　：0x0
タスクのキューイング方式　：資源の獲得要求を行った順 (FIFO 順)
初期資源数　：0xff
最大資源数　：0xff
キー ID 番号　：0x1

6) イベントフラグ情報
ID 番号　：0x1
拡張情報　：0x0
複数タスクの待ち許可 / 禁止　：禁止
初期ビット・パターン　：0x0
キー ID 番号　：0x1

7) メールボックス情報
ID 番号　：0x1
拡張情報　：0x0
タスクのキューイング方式　：メッセージの受信要求を行った順 (FIFO 順)
メッセージのキューイング方式　：メッセージの送信を行った順 (FIFO 順)
キー ID 番号　：0x1

8) 割り込みハンドラ情報
割り込み要因番号　：0x0 (INTP0)
記述言語　：アセンブリー言語
起動アドレス　：_inthdr01
固有 GP レジスタ値　：設定しない
固有 TP レジスタ値　：設定しない

9) メモリ・プール情報
ID 番号　：0x1
拡張情報　：0x0
タスクのキューイング方式　：タスクの優先度順
メモリ・プール情報　：User Memory Pool 0 から 0x2000 バイトのメモリ領域を確保
キー ID 番号　：0x1

10) 周期起動ハンドラ情報
指定番号　：0x1
拡張情報　：0x0
記述言語　：C 言語
起動アドレス　：_cychdr01
初期活性状態　：OFF 状態
起動時間間隔　：0x100 基本クロック周期
固有 GP レジスタ値　：設定しない
固有 TP レジスタ値　：設定しない

11) 拡張 SVC ハンドラ情報
拡張機能コード　：0x1
記述言語　：C 言語
起動アドレス　：_svchdr01
固有 GP レジスタ値　：設定しない
固有 TP レジスタ値　：設定しない

12) 初期化ハンドラ情報
記述言語　：C 言語
起動アドレス　：_varfunc
固有 GP レジスタ値　：設定しない
固有 TP レジスタ値　：設定しない

【 SCT(System Call Table) 情報 】

1) **タスク管理 / タスク付属同期管理機能システム・コール情報**
ユーザ処理プログラムで使用するタスク管理 / タスク付属同期管理機能システム・コール情報として，次のシステム・コールを定義

sta_tsk　　exd_tsk

2) **同期通信 (セマフォ) 管理機能システム・コール情報**
ユーザ処理プログラムで使用する同期通信 (セマフォ) 管理機能システム・コール情報として，次のシステム・コールを定義

sig_sem　　wai_sem

3) **同期通信 (イベントフラグ) 管理機能システム・コール情報**
ユーザ処理プログラムで使用する同期通信 (イベントフラグ) 管理機能システム・コール情報として，次のシステム・コールを定義

cre_flg　　del_flg　　set_flg　　wai_flg

4) **同期通信 (メールボックス) 管理機能システム・コール情報**
ユーザ処理プログラムで使用する同期通信 (メールボックス) 管理機能システム・コール情報として，次のシステム・コールを定義

cre_mbx　　del_mbx　　snd_msg　　rcv_msg

5) **割り込み処理管理機能システム・コール情報**
ユーザ処理プログラムで使用する割り込み処理管理機能システム・コール情報として，次のシステム・コールを定義

ret_int

6) **メモリ・プール管理機能システム・コール情報**
ユーザ処理プログラムで使用するメモリ・プール管理機能システム・コール情報として，次のシステム・コールを定義

cre_mpl　　del_mpl　　get_blk　　rel_blk

7) **時間管理機能システム・コール情報**
ユーザ処理プログラムで使用する時間管理機能システム・コール情報として，次のシステム・コールを定義

act_cyc　　ref_cyc

8) **システム管理機能システム・コール情報**
ユーザ処理プログラムで使用するシステム管理機能システム・コール情報として，次のシステム・コールを定義

viss_svc

図 6-2　システム・コンフィギュレーション・ファイルの記述例 (CA850 対応版)

```
------------------------------
-- リアルタイム OS 情報の記述が開始されることの宣言
------------------------------
ser_def

------------------------------
-- リアルタイム OS 情報の記述
------------------------------

-- RX シリーズ情報
rxsers          RX850PRO        V320

------------------------------
-- SIT(System Information Table) 情報の記述が開始されることの宣言
------------------------------
sit_def

------------------------------
-- SIT(System Information Table) 情報の記述
------------------------------

-- システム情報
cputype         V850E1
clktim          0x1
clkhdr          INTCM4
defstk          0x100
intstk          0x100 : SPOL0
prtstk          0x1
prtsem          0x1
prtflg          0x1
prtmbx          0x1
prtmpl          0x1

-- システム最大値情報
maxpri          0xf
maxtsk          0x2
maxsem          0x1
maxflg          0x2
maxmbx          0x3
maxmpl          0x2
maxcyc          0x1
maxsvc          0x1
maxint          0x5
maxintfactor    0x48

-- システム・メモリ情報
mem             SPOL0           syspol0         0x2000
mem             SPOL1           syspol1         0x1000
mem             UPOL0           usrpol0_0       0x7000
mem             UPOL0           usrpol0_1       0x2500
mem             UPOL1           usrpol1         0x1500

- タスク情報
tsk             0x1             TTS_RDY         0x0             0x0             TA_ASM          \
                _task01         0x8             TA_ENAINT       0x100 : SPOL0   no_use          \
                no_use          0x1
tsk             TASK02          TTS_DMT         0x0             0x0             TA_HLNG         \
                _task02         0xf             TA_DISINT       0x100 : SPOL0   no_use          \
```

```
                no_use          0x2

-- セマフォ情報
sem             0x1             0x0             TA_TFIFO        0xff            0xff            \
                0x1

-- イベントフラグ情報
flg             0x1             0x0             TA_WSGL         0x0             0x1

-- メールボックス情報
mbx             0x1             0x0             TA_TFIFO        TA_MFIFO        0x1

-- 割り込みハンドラ情報
inthdr          INTP0           TA_ASM          _inthdr01       no_use          no_use

-- メモリ・プール情報
mpl             0x1             0x0             TA_TPRI         0x2000 : UPOL0                  \
                0x1

-- 周期起動ハンドラ情報
cyc             0x1             0x0             TA_HLNG         _cychdr01       TCY_OFF         \
                0x100           no_use          no_use

-- 拡張 SVC ハンドラ情報
svc             0x1             TA_HLNG         _svchdr01       no_use          no_use

-- 初期化ハンドラ情報
ini             TA_HLNG         _varfunc        no_use          no_use

------------------------------
-- SCT(System Call Table) 情報の記述が開始されることの宣言
------------------------------
sct_def

------------------------------
-- SCT(System Call Table) 情報の記述
------------------------------

-- タスク管理 / タスク付属同期管理機能システム・コール情報
tsksvc          sta_tsk
tsksvc          exd_tsk

-- 同期通信 ( セマフォ ) 管理機能システム・コール情報
semsvc          sig_sem
semsvc          wai_sem

-- 同期通信 ( イベントフラグ ) 管理機能システム・コール情報
flgsvc          cre_flg
flgsvc          del_flg
flgsvc          set_flg
flgsvc          wai_flg

-- 同期通信 ( メールボックス ) 管理機能システム・コール情報
mbxsvc          cre_mbx
mbxsvc          del_mbx
mbxsvc          snd_msg
mbxsvc          rcv_msg

-- 割り込み処理管理機能システム・コール情報
intsvc          ret_int
```

```
-- メモリ・プール管理機能システム・コール情報
mplsvc          cre_mpl
mplsvc          del_mpl
mplsvc          get_blk
mplsvc          rel_blk

-- 時間管理機能システム・コール情報
timsvc          act_cyc
timsvc          ref_cyc

-- システム管理機能システム・コール情報
syssvc          viss_svc
```

第 7 章　コンフィギュレータ CF850 Pro

この章では，コンフィギュレータ CF850 Pro を操作して，システム・コンフィギュレーション・ファイルから情報ファイル (システム情報テーブル・ファイル，システム・コール・テーブル・ファイル，システム情報ヘッダ・ファイル) を生成する方法について解説しています。

7.1　概要

RX850 Pro が提供している機能を利用したシステム (ロード・モジュール) を構築をする場合，RX850 Pro に提供するデータを保持した情報が必要となります。

基本的に情報ファイルは，規定された形式のデータ羅列であるため，各種エディタを用いて記述することは可能です。しかし，情報ファイルは，記述性 / 可読性の面で劣ったものとなっているため，記述に際してはかなりの時間と労力を必要としてます。

そこで，RX850 Pro では，記述性 / 可読性の面で優れたシステム・コンフィギュレーション・ファイルから情報ファイルへと変換するユーティリティ・ツール (コンフィギュレータ CF850 Pro) を提供しています。

CF850 Pro は，システム・コンフィギュレーション・ファイルを入力ファイルとして読み込んだあと，情報ファイルを出力します。

以下に，CF850 Pro が出力する情報ファイルについて示します。

- システム情報テーブル・ファイル
  RX850 Pro が動作するうえで必要なるデータ (タスク，セマフォ，イベントフラグ等の RX850 Pro の資源情報) を保持した情報ファイルです。

- システム・コール・テーブル・ファイル
  ユーザの処理プログラム内で使用するシステム・コールの種類に関するデータを保持した情報ファイルです。

- システム情報ヘッダ・ファイル
  システム・コンフィギュレーション・ファイルに記述されたオブジェクト名 (タスク名，セマフォ名，イベントフラグ名など) と ID 番号の対応付けを保持した情報ファイルです。

表 7-1 に，CF850 Pro の動作環境を示します。

表 7-1　CF850 Pro の動作環境

ホスト・マシン	オペレーティング・システム
Windows ベース - IBM PC/AT 互換機	Windows 98，Me，NT 4.0，2000，XP

7.2　起動方法

7.2.1　コマンド・ラインからの起動

以下に，CF850 Pro をコマンド・ラインから起動する際の起動方法を示します。
ただし，入力例中の “ C> ” はコマンド・プロンプトを，“ △ ” はスペース・キーの入力を表しています。
また，“ [　] ” で囲まれた起動オプションは，省略可能な起動オプションであることを表しています。

【 CA850 対応版の場合 】

C>　cf850pro △ [*@cmd_file*] △ [-cpu △ *name*] △ [-devpath=*path*] △ [-i △ *sit_file*] [-c △ *sct_file*] [-d △ *h_file*] [-ni] [-nc] [-nd] [-V] [-help] *cf_file*

【 GHS 製コンパイラ対応版の場合 】

C>　cf850pro △ [*@cmd_file*] △ [-i △ *sit_file*] [-c △ *sct_file*] [-d △ *h_file*] [-ni] [-nc] [-nd] [-V] [-help] *cf_file*

次に，CF850 Pro の起動オプションを示します。

- *@cmd_file*
  コマンド・ファイル名を指定します。
  省略時　コマンド・ライン上で指定された起動オプションが有効となります。
  注意 1　コマンド・ファイル名 *cmd_file* として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。
  注意 2　コマンド・ファイルについての詳細は，「7.2.3　コマンド・ファイル」を参照してください。

- -cpu △ *name*
  ターゲット・デバイスの品種指定名を指定します。
  省略時　CF850 Pro はデバイス・ファイルの読み込みを行いません。このため，システム・コンフィギュレーション･ファイル内で‘ デバイス･ファイルで規定されている割り込み要因名 ”を用いた定義が行えなくなります。
  注意　　本起動オプションは，CA850 対応版でのみ指定可能となります。

- -devpath=*path*
  -cpu △ *name* で指定されたターゲット・デバイスに対応したデバイス・ファイルを *path* フォルダから検索します。
  省略時　カレント・フォルダ，..\..\..\dev の順序で検索処理を行います。
  注意　　本起動オプションは，CA850 対応版でのみ指定可能となります。

- -i △ *sit_file*
  CF850 Pro からの出力ファイル (システム情報テーブル・ファイル名) を指定します。
  省略時　以下に示した起動オプションが指定されていたものとして処理を行います。

CA850 対応版	： -i △ sit.s
GHS 製コンパイラ対応版	： -i △ sit.850

  注意 1　出力ファイル名 *sit_file* として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。
  注意 2　本起動オプションと -ni を同時に指定した場合，前者に入力した起動オプションは無効となり，後者に入力した起動オプションが有効となります。

- -c △ *sct_file*
  CF850 Pro からの出力ファイル (システム・コール・テーブル・ファイル名) を指定します。
  省略時　以下に示した起動オプションが指定されていたものとして処理を行います。

CA850 対応版	： -c △ sct.s
GHS 製コンパイラ対応版	： -c △ sct.850

  注意 1　出力ファイル名 *sct_file* として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。

注意 2　**本起動オプションと** -nc **を同時に指定した場合，前者に入力した起動オプションは無効となり，後者に入力した起動オプションが有効となります。**

- -d △ *h_file*

CF850 Pro **からの出力ファイル (システム情報ヘッダ・ファイル名) を指定します。**

省略時　*cf_file* **で指定されたシステム・コンフィギュレーション・ファイル名の拡張子を“** .h **”に置き換えたシステム情報ヘッダ・ファイルを出力します。**

注意 1　**出力ファイル名** *h_file* **として指定可能な文字数は，パス名を含めて** 255 **文字以内に限られます。**

注意 2　**本起動オプションと** -nd **を同時に指定した場合，前者に入力した起動オプションは無効となり，後者に入力した起動オプションが有効となります。**

- -ni

システム情報テーブル・ファイルを出力しません。

省略時　**以下に示した起動オプションが指定されていたものとして処理を行います。**

CA850 **対応版**　：-i △ sit.s
GHS **製コンパイラ対応版**　：-i △ sit.850

注意　**本起動オプションと** -i △ *sit_file* **を同時に指定した場合，前者に入力した起動オプションは無効となり，後者に入力した起動オプションが有効となります。**

- -nc

システム・コール・テーブル・ファイルを出力しません。

省略時　**以下に示した起動オプションが指定されていたものとして処理を行います。**

CA850 **対応版**　：-c △ sct.s
GHS **製コンパイラ対応版**　：-c △ sct.850

注意　**本起動オプションと** -c △ *sct_file* **を同時に指定した場合，前者に入力した起動オプションは無効となり，後者に入力した起動オプションが有効となります。**

- -nd

システム情報ヘッダ・ファイルを出力しません。

省略時　*cf_file* **で指定されたシステム・コンフィギュレーション・ファイル名の拡張子を“** .h **”に置き換えたシステム情報ヘッダ・ファイルを出力します**

注意　**本起動オプションと** -d △ *h_file* **を同時に指定した場合，前者に入力した起動オプションは無効となり，後者に入力した起動オプションが有効となります。**

- -V

CF850 Pro **のバージョン情報を標準出力に出力します。**

省略時　**バージョン情報を出力しません。**

注意　**本起動オプションを指定すると，ほかの起動オプションはすべて無効となります。**

- -help

CF850 Pro **の起動オプションの使い方を標準出力に出力します。**

省略時　CF850 Pro **の起動オプションの使い方を出力しません。**

注意　**本起動オプションを指定すると，ほかの起動オプションはすべて無効となります。**

- *cf_file*

CF850 Pro **の入力ファイル (システム・コンフィギュレーション・ファイル名) を指定します。**

省略時　**入力ファイルの指定は，省略できません。**

注意　**入力ファイル名** *cf_file* **として指定可能な文字数は，パス名を含めて** 255 **文字以内に限られます。**

7.2.2　PM+ からの起動

以下に，CF850 Pro の起動オプションを CA850 が提供する統合開発環境プラットホーム PM+ から設定する方法を示します。
なお，本項では，既存のプロジェクト・ファイルに対する設定方法を例にとり，説明を行っています。

1) PM+ の起動
ショート・カット (デフォルト：[スタート] → [プログラム (P)] → [NEC Electronics Tools] → [PM+] → [V*x.xx*] → [PM+ V*x.xx*]) をクリック，または，実行形式ファイル (デフォルト：C:\Program Files\NEC Electronics Tools\PM+\V*x.xx*\bin\PMplus.exe) をダブル・クリックし，PM+ を起動します。

2) [ワークスペースを開く] ダイアログのオープン
[ファイル (F)] → [ワークスペースを開く (W)...] を選択し，[ワークスペースを開く] ダイアログをオープンします。

注意　[ワークスペースを開く] ダイアログについての詳細は，「PM+ ユーザーズ・マニュアル」を参照してください。

3) ワークスペース・ファイルの指定
[ファイルの場所 (I)] エリア，[ファイル名 (N)] エリア，および，[ファイルの種類 (T)] エリアをそれぞれに選択 / 指定したあと，< 開く (O) > ボタンをクリックし，CF850 Pro の起動オプションを設定するワークスペース・ファイル (，または，プロジェクト・ファイル) を指定します。

4) [OS の選択] ダイアログのオープン
[ツール (T)] → [OS の選択 (O)...] を選択し，[OS の選択] ダイアログをオープンします。

注意　[OS の選択] ダイアログについての詳細は，「付録 A　ウインドウ・リファレンス」を参照してください。

5) [RX850 Pro 詳細設定] ダイアログのオープン
[OS の選択 (O)] エリアのリスト・ボックスから " RX850 Pro V3.*xx* " を選択したあと，< OK > ボタンをクリックし，[RX850 Pro 詳細設定] ダイアログをオープンします。

注意　[RX850 Pro 詳細設定] ダイアログについての詳細は，「付録 A　ウインドウ・リファレンス」を参照してください。

6) システム・コンフィギュレーション・ファイルの指定
[システム・コンフィギュレーション・ファイル (C)] エリアにおいて，入力ファイル名 (システム・コンフィギュレーション・ファイル名) を指定します。

7) システム情報テーブル・ファイルの指定
[システム情報テーブル・ファイルを出力する [-i/-ni] (T)] チェック・ボックスをチェック状態としたあと，[ファイル名 (F)] エリアにおいて，出力ファイル名 (システム情報テーブル・ファイル名) を指定します。

8) システム・コール・テーブル・ファイルの指定
[システム・コール・テーブル・ファイルを出力する [-c/-nc] (S)] チェック・ボックスをチェック状態としたあと，[ファイル名 (L)] エリアにおいて，出力ファイル名 (システム・コール・テーブル・ファイル名) を指定します。

9) システム情報ヘッダ・ファイルの指定
[システム情報ヘッダ・ファイルを出力する [-d/-nd] (H)] チェック・ボックスをチェック状態としたあと，[ファイル名 (A)] エリアにおいて，出力ファイル名 (システム情報ヘッダ・ファイル名) を指定します。

10) ニュークリアス・ライブラリの指定
[ニュークリアス・ライブラリをリンクする (U)] チェック・ボックスをチェック状態としたあと，[ファイル名 (N)] コンボ・ボックスにおいて，" ロード・モジュールを作成する際 (システム構築時) にリンクするニュークリアス・ライブラリ：librxp.a，librxpm.a など " を指定します。

11) インタフェース・ライブラリの指定
[インタフェース・ライブラリをリンクする (R)] チェック・ボックスをチェック状態としたあと，[ファイル名 (A)] コンボ・ボックスにおいて，" ロード・モジュールを作成する際 (システム構築時) にリンクするインタフェース・ライブラリ：libchp.a，libncp.a など " を指定します。

12) ニュークリアス共通オブジェクトの指定
[ニュークリアス共通オブジェクトをリンクする (O)] チェック・ボックスをチェック状態としたあと，[ファイル名 (M)] コンボ・ボックスにおいて，" ロード・モジュールを作成する際 (システム構築時) にリンクするニュークリアス共通オブジェクト：rxcore.o，rxtmcore.o など " を指定します。

13) **起動オプションの確認**
6) 〜 12) **の指定により設定される** CF850 Pro **の起動オプション指定形式が** [**コマンドライン・オプション**] **エリアに表示されます。これにより，**6) 〜 12) **の指定結果が意図したものと一致しているか否かを明示的に確認することができます。**

14) **プロジェクト・ファイルに対する反映**
< OK > **ボタンをクリックし，上記の操作結果をプロジェクト・ファイルに対して反映します。**

7.2.3　コマンド・ファイル

CF850 Pro では，コマンド・ライン上で指定可能な起動オプションの文字数制限を解消する目的からコマンド・ファイル対応を行っています。
以下に，コマンド・ファイルの記述形式を示します。

1）**コメント行**
行頭に # **が記述された行については，コメント行として扱われます。**

2）**起動オプション**
起動オプションと起動オプションは，スペース・コード，または，改行コードで区切って記述します。
なお，-cpu，-i，-c，-d **といった** *-xxx* **部とパラメータ部から構成される起動オプションについては，** *-xxx* **部とパラメータ部をスペース・コード，または，改行コードで区切って記述します。**

注意　-devpath **のパラメータ部（パス名** *path*) **にスペース・コードを含むフォルダ名が存在する場合は，パラメータ部をダブル・クォーテーション** " **でくくる必要があります。**

```
-devpath="Program Files\DEV"
```

3）**文字数制限**
コマンド・ファイル内の 1 **行に対する文字数制限は** 4096 **文字となっています。**

図 7-1 **に，**CA850 **対応版におけるコマンド・ファイルの記述例を示します。**
なお，図 7-1 **の記述例では，次に示す起動オプションが記述されています。**

ターゲット・デバイス	**：** μPD703000
デバイス・ファイルの検索フォルダ	**：** C:\NECTools32\DEV
システム情報テーブル・ファイル	**：** sys.s
システム・コール・テーブル・ファイル	**：** sct.s
システム情報ヘッダ・ファイル	**：** sys.h
システム・コンフィギュレーション・ファイル	**：** sys.cf

図 7-1　**コマンド・ファイルの記述例** (CA850 **対応版**)

```
# Command File
-cpu
3000
-devpath=C:\NECTools32\DEV
-i
sys.s
-c
sct.s
-d
sys.h
sys.cf
```

7.3　コマンド入力例

以下に，CA850 対応版における CF850 Pro のコマンド入力例を示します。
なお，この例では，ターゲット・デバイスとして μPD703000 を指定しています。

- cf850pro -cpu 3000 -devpath=C:\NECTools32\DEV -i sitfile.s -c sctfile.s -d hfile.h cffile.cf
  システム・コンフィギュレーション・ファイル cffile.cf をカレント・フォルダから，品種指定名 3000 に対応したデバイス・ファイルを C:\NECTools32\DEV フォルダから入力ファイルとして読み込んだあと，システム情報テーブル・ファイル sitfile.tbl，システム・コール・テーブル・ファイル sctfile.tbl，システム情報ヘッダ・ファイル hfile.h を出力します。

- cf850pro -cpu 3000 -devpath=C:\NECTools32\DEV -i sitfile.s cffile.cf
  システム・コンフィギュレーション・ファイル cffile.cf をカレント・フォルダから，品種指定名 3000 に対応したデバイス・ファイルを C:\NECTools32\DEV フォルダから入力ファイルとして読み込んだあと，システム情報テーブル・ファイル sitfile.tbl，システム・コール・テーブル・ファイル cffilec.tbl，システム情報ヘッダ・ファイル cffile.h を出力します。

- cf850pro -cpu 3000 -devpath=C:\NECTools32\DEV -c sctfile.s cffile.cf
  システム・コンフィギュレーション・ファイル cffile.cf をカレント・フォルダから，品種指定名 3000 に対応したデバイス・ファイルを C:\NECTools32\DEV フォルダから入力ファイルとして読み込んだあと，システム情報テーブル・ファイル cffilei.tbl，システム・コール・テーブル・ファイル sctfile.tbl，システム情報ヘッダ・ファイル cffile.h を出力します。

- cf850pro -cpu 3000 -devpath=C:\NECTools32\DEV -d hfile.h cffile.cf
  システム・コンフィギュレーション・ファイル cffile.cf をカレント・フォルダから，品種指定名 3000 に対応したデバイス・ファイルを C:\NECTools32\DEV フォルダから入力ファイルとして読み込んだあと，システム情報テーブル・ファイル cffilei.tbl，システム・コール・テーブル・ファイル sctfilec.tbl，システム情報ヘッダ・ファイル hfile.h を出力します。

- cf850pro -cpu 3000 -devpath=C:\NECTools32\DEV cffile.cf
  システム・コンフィギュレーション・ファイル cffile.cf をカレント・フォルダから，品種指定名 3000 に対応したデバイス・ファイルを C:\NECTools32\DEV フォルダから入力ファイルとして読み込んだあと，システム情報テーブル・ファイル cffilei.tbl，システム・コール・テーブル・ファイル cffilec.tbl，システム情報ヘッダ・ファイル cffile.h を出力します。

- cf850pro -V
  CF850 Pro のバージョン情報を標準出力に出力します。

- cf850pro -help
  CF850 Pro の起動オプションの使い方を標準出力に出力します。

7.4　メッセージ

メッセージは，CF850 Pro が処理を実行中に，「システム・コンフィギュレーション・ファイルに誤った定義が行われている」等といった記述ミスを検出した際に生成され，標準出力に出力されます。
なお，CF850 Pro のフォーマッタ部では，メッセージを 3 つのレベル (致命的なエラー，致命的でないエラー，警告) に分けており，メッセージを出力する際には，その先頭にレベルを表す英字が付けられます。

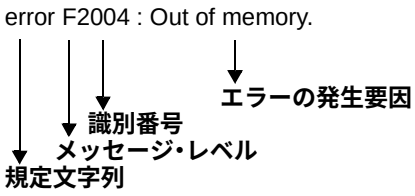
F　：致命的なエラー
致命的なエラーが発生した場合，メッセージを出力後，コンフィギュレーション処理は中止されます。
例　メモリ領域が足りない。

E　：致命的でないエラー
致命的でないエラーが発生した場合，メッセージを出力後，コンフィギュレーション処理は中止されます。
例　二重定義が行われている。

W　：警告
警告が発生した場合，メッセージを出力後，コンフィギュレーション処理は続行されます。
例　デフォルト・サイズ以下のスタック・サイズを指定した。

図 7-2 に，メッセージの出力形式を示します。

図 7-2　メッセージの出力形式



CF850 Pro では，処理が完了した際，以下に示した終了メッセージを出力します。
なお，%d は，エラー，および，警告の検出個数を意味します。

```
total error(s) : %d total warning(s) : %d
```

7.4.1　致命的なエラー

次に，致命的なエラーが発生した際に出力されるメッセージを示します。
なお，メッセージ内のイタリック表記 (*file_name* 等) は，致命的なエラーが発生した際に決定されるものです。

F2001 :　Usage: cf850pro [@<cfile>] [-cpu <name>] [-devpath=<path>] [-i <SITfile>] [-c <SCTfile>] [-d <includefile>] [-ni] [-nc] [-nd] [-V] [-help] <file>
CA850 **に対応した** CF850 Pro **の起動オプションの指定が不正です。**

F2001 :　Usage: cf850pro_ghs [@<cfile>] [-i <SITfile>] [-c <SCTfile>] [-d <includefile>] [-ni] [-nc] [-nd] [-V] [-help] <file>
GHS **製コンパイラに対応した** CF850 Pro **の起動オプションの指定が不正です。**

F2002 :　Can't allocate memory.
メモリが足りません。

F2003 :　Can't open file *file_name*.
ファイル *file_name* **がオープンできません。**

F2004 :　Out of memory.
メモリが足りません。

F2005 :　Can't open device file.
デバイス・ファイルがオープンできません。

F2006 :　Can't read device file.
デバイス・ファイルが読み込めません。

F2007 :　Unknown device file format.
未対応のデバイス・ファイルが指定されています。

F2008 :　Output file "*file_name*" names are the same.
起動オプションで同一出力ファイル名 *file_name* **が指定されています。**

7.4.2　致命的でないエラー

次に，致命的でないエラーが発生した際に出力されるメッセージを示します。
なお，メッセージ内のイタリック表記 (*symbol_name* 等) は，致命的でないエラーが発生した際に決定されるものです。

E2001 :　ser_def not defined.
リアルタイム OS 情報の開始宣言 ser_def が先頭行で行われていません。

E2002 :　Illegal ser_def.
リアルタイム OS 情報の開始宣言 ser_def が誤った場所で行われています。

E2003 :　ser_def already defined.
リアルタイム OS 情報の開始宣言 ser_def が二重定義されています。

E2004 :　sit_def not defined.
SIT 情報の開始宣言 sit_def が定義されていません。

E2005 :　Illegal sit_def.
SIT 情報の開始宣言 sit_def が誤った場所で行われています。

E2006 :　sit_def already defined.
SIT 情報の開始宣言 sit_def が二重定義されています。

E2007 :　Out of sit_def division.
SIT 情報に含まれるデータが SIT 情報の開始宣言 sit_def より前に定義されています。

E2008 :　sct_def not defined.
SCT 情報の開始宣言 sct_def が定義されていません。

E2009 :　Illegal sct_def.
SCT 情報の開始宣言 sct_def が誤った場所で行われています。

E2010 :　sct_def already defined.
SCT 情報の開始宣言 sct_def が二重定義されています。

E2011 :　Out of sct_def division.
SCT 情報に含まれるデータが SCT 情報の開始宣言 sct_def より前に定義されています。

E2012 :　rxsers not defined.
RX シリーズ情報 rxsers が定義されていません。

E2013 :　Illegal rxsers.
RX シリーズ情報 rxsers の定義場所，または，バージョン番号に指定した値が不正です。

E2014 :　rxsers already defined.
RX シリーズ情報 rxsers が二重定義されています。

E2101 :　Integer overflow.
数値が 32 ビット・データ範囲を越えています。

E2102 :　Syntax error.
システム・コンフィギュレーション・ファイルの記述形式が誤っています。

E2103 :　Word too long.
シンボル名が最大文字数を越えています。

E2104 :　Address out of range.
アドレスに範囲外の値が指定されています。

E2105 :　Address must be aligned by 2.
アドレスには 2 バイト境界値を指定してください。

E2106 :　Symbol *symbol_name* already defined.
シンボル名 *symbol_name* が二重定義されています。

E2107 :　Illegal system memorypool for stack.
スタック確保領域に指定したメモリの種類が不正です。

E2108 :　Memory block size out of range.
システム・メモリのサイズに範囲外の値が指定されています。

E2109 :　Memory block size must be aligned by 4.
システム・メモリのサイズに 4 **バイト境界値以外の値が指定されています。**

E2201 :　System clock time not defined.
基本クロック周期が定義されていません。

E2202 :　System clock time out of range.
基本クロック周期に範囲外の値が指定されています。

E2203 :　System clock time already defined.
基本クロック周期が二重定義されています。

E2204 :　Task default stack size not defined.
デフォルト・スタック・サイズが定義されていません。

E2205 :　Task default stack size out of range.
デフォルト・スタック・サイズに範囲外の値が指定されています。

E2206 :　Task default stack size already defined.
デフォルト・スタック・サイズが二重定義されています。

E2207 :　System stack size not defined.
割り込みハンドラ用スタック (システム・スタック) のサイズが定義されていません。

E2208 :　System stack size out of range.
割り込みハンドラ用スタック (システム・スタック) のサイズに範囲外の値が指定されています。

E2209 :　System stack size already defined.
割り込みハンドラ用スタック (システム・スタック) のサイズが二重定義されています。

E2210 :　Protect task id not defined.
タスクの ID **番号保護範囲が定義されていません。**

E2211 :　Protect task id out of range.
タスクの ID **番号保護範囲に範囲外の値が指定されています。**

E2212 :　Protect task id already defined.
タスクの ID **番号保護範囲が二重定義されています。**

E2213 :　Protect task id greater than max task.
タスクの ID **番号保護範囲にタスクの最大生成数よりも大きな値が指定されています。**

E2214 :　Protect semaphore id not defined.
セマフォの ID **番号保護範囲が指定されていません。**

E2215 :　Protect semaphore id out of range.
セマフォの ID **番号保護範囲に範囲外の値が指定されています。**

E2216 :　Protect semaphore id already defined.
セマフォの ID **番号保護範囲が二重定義されています。**

E2217 :　Protect semaphore id greater than max semaphore.
セマフォの ID **番号保護範囲にセマフォの最大生成数よりも大きな値が指定されています。**

E2218 :　Protect eventflag id not defined.
イベントフラグの ID **番号保護範囲が定義されていません。**

E2219 :　Protect eventflag id out of range.
イベントフラグの ID 番号保護範囲に範囲外の値が指定されています。

E2220 :　Protect eventflag id already defined.
イベントフラグの ID 番号保護範囲が二重定義されています。

E2221 :　Protect eventflag id greater than max eventflag.
イベントフラグの ID 番号保護範囲にイベントフラグの最大生成数よりも大きな値が指定されています。

E2222 :　Protect mailbox id not defined.
メールボックスの ID 番号保護範囲が定義されていません。

E2223 :　Protect mailbox id out of range.
メールボックスの ID 番号保護範囲に範囲外の値が指定されています。

E2224 :　Protect mailbox id already defined.
メールボックスの ID 番号保護範囲が二重定義されています。

E2225 :　Protect mailbox id greater than max mailbox.
メールボックスの ID 番号保護範囲にメールボックスの最大生成数よりも大きな値が指定されています。

E2226 :　Protect memorypool id not defined.
メモリ・プールの ID 番号保護範囲が定義されていません。

E2227 :　Protect memorypool id out of range.
メモリ・プールの ID 番号保護範囲に範囲外の値が指定されています。

E2228 :　Protect memorypool id already defined.
メモリ・プールの ID 番号保護範囲が二重定義されています。

E2229 :　Protect memorypool id greater than max memorypool.
メモリ・プールの ID 番号保護範囲にメモリ・プールの最大生成数よりも大きな値が指定されています。

E2230 :　Max priority level not defined.
タスクの優先度範囲が指定されていません。

E2231 :　Max priority level out of range.
タスクの優先度範囲に範囲外の値が指定されています。

E2232 :　Max priority level already defined.
タスクの優先度範囲が二重定義されています。

E2233 :　Max task not defined.
タスクの最大生成数が定義されていません。

E2234 :　Max task out of range.
タスクの最大生成数に範囲外の値が指定されています。

E2235 :　Max task already defined.
タスクの最大生成数が二重定義されています。

E2236 :　Max semaphore not defined.
セマフォの最大生成数が定義されていません。

E2237 :　Max semaphore out of range.
セマフォの最大生成数に範囲外の値が指定されています。

E2238 :　Max semaphore already defined.
セマフォの最大生成数が二重定義されています。

E2239 :　Max eventflag not defined.
イベントフラグの最大生成数が定義されていません。

E2240 :　Max eventflag out of range.
イベントフラグの最大生成数に範囲外の値が指定されています。

E2241 :　Max eventflag already defined.
イベントフラグの最大生成数が二重定義されています。

E2242 :　Max mailbox not defined.
メールボックスの最大生成数が定義されていません。

E2243 :　Max mailbox out of range.
メールボックスの最大生成数に範囲外の値が指定されています。

E2244 :　Max mailbox already defined.
メールボックスの最大生成数が二重定義されています。

E2245 :　Max memorypool not defined.
メモリ・プールの最大生成数が定義されていません。

E2246 :　Max memorypool out of range.
メモリ・プールの最大生成数に範囲外の値が指定されています。

E2247 :　Max memorypool already defined.
メモリ・プールの最大生成数が二重定義されています。

E2248 :　Max cyclic handler not defined.
周期起動ハンドラの最大生成数が定義されていません。

E2249 :　Max cyclic handler out of range.
周期起動ハンドラの最大生成数に範囲外の値が指定されています。

E2250 :　Max cyclic handler already defined.
周期起動ハンドラの最大生成数が二重定義されています。

E2251 :　Max svc handler not defined.
拡張 SVC **ハンドラの最大生成数が定義されていません。**

E2252 :　Max svc handler out of range.
拡張 SVC **ハンドラの最大生成数に範囲外の値が指定されています。**

E2253 :　Max svc handler already defined.
拡張 SVC **ハンドラの最大生成数が二重定義されています。**

E2254 :　System memorypool "*mem_id*" not defined.
システム・メモリ *mem_id* **が定義されていません。**

E2255 :　System memorypool id out of range.
システム・メモリの種類に範囲外の値が指定されています。

E2256 :　Illegal system memorypool id.
システム・メモリの種類が不正です。

E2257 :　Memory section "*sec_nam*" already defined.
システム・メモリが割り付けられるメモリ領域のセクション名 *sec_nam* **が二重定義されています。**

E2258 :　Memory block address must be symbol.
システム・メモリが割り付けられるメモリ領域のセクション名が不正です。

E2259 :　Not enough system memorypool "*mem_id*" block size.
システム・メモリ *mem_id* **に管理オブジェクト，スタック，または，メモリ・プールを割り付けるのに十分なサイズを指定していません。または，システム・メモリ** *mem_id* **が小さく不連続な領域に分割されており，管理オブジェクト，スタック，または，メモリ・プールに必要なサイズを確保できません。**

E2260 :　System memorypool size exceeds 4Gbytes.
システム・メモリの合計サイズが 4G **バイトを越えています。**

E2261 :　Memory block overlap.
システム・メモリ領域が重なっています。

E2262 :　Task not defined.
タスク情報が定義されていません。

E2263 :　Task id "*tsk_id*" already defined.
タスクの ID **番号** *tsk_id* **が二重定義されています。**

E2264 :　Non-protect task id all assigned.
タスクの ID **番号として自動的に割り付けが可能な値がすべて使用済みです。**

E2265 :　Too many tasks.
タスク情報の数がタスクの最大生成数を越えています。

E2266 :　Task id out of range.
タスクの ID **番号に範囲外の値が指定されています。**

E2267 :　Task id greater than max task.
タスクの ID **番号にタスクの最大生成数よりも大きな値が指定されています。**

E2268 :　Task priority greater than max priority.
タスクの初期優先度にタスクの優先度範囲よりも大きな値が指定されています。

E2269 :　Task priority out of range.
タスクの初期優先度に範囲外の値が指定されています。

E2270 :　Task stack size out of range.
タスク用スタックのサイズに範囲外の値が指定されています。

E2271 :　Task key-id out of range.
タスクのキー ID **番号に範囲外の値が指定されています。**

E2272 :　Task key-id "*key_id*" already defined.
タスクのキー ID **番号** *key_id* **が二重定義されています。**

E2273 :　Semaphore id "*sem_id*" already defined.
セマフォの ID **番号** *sem_id* **が二重定義されています。**

E2274 :　Non-protect semaphore id all assigned.
セマフォの ID **番号として自動的に割り付けが可能な値がすべて使用済みです。**

E2275 :　Too many semaphores.
セマフォ情報の数がセマフォの最大生成数を越えています。

E2276 :　Semaphore id out of range.
セマフォの ID **番号に範囲外の値が指定されています。**

E2277 :　Semaphore id greater than max semaphore.
セマフォの ID **番号にセマフォの最大生成数よりも大きな値が指定されています。**

E2278 :　Initial resource count out of range.
セマフォの初期資源数に範囲外の値が指定されています。

E2279 :　Max resource count out of range.
セマフォの最大資源数に範囲外の値が指定されています。

E2280 :　Initial resource count greater than max resource count.
セマフォの初期資源数にセマフォの最大資源数よりも大きな値が指定されています。

E2281 :　Semaphore key-id out of range.
セマフォのキー ID **番号に範囲外の値が指定されています。**

E2282 :　Semaphore id is 0, but semaphore key-id not specified.
ID **番号自動生成のセマフォにシンボル名とキー** ID **番号のどちらも指定していません。**

E2283 :　Semaphore key-id "*key_id*" already defined.
セマフォのキー ID **番号** *key_id* **が二重定義されています。**

E2284 :　Eventflag id "*flg_id*" already defined.
イベントフラグの ID **番号** *flg_id* **が二重定義されています。**

E2285 :　Non-protect eventflag id all assigned.
イベントフラグの ID **番号として自動的に割り付けが可能な値がすべて使用済みです。**

E2286 :　Too many eventflags.
イベントフラグ情報の数がイベントフラグの最大生成数を越えています。

E2287 :　Eventflag id out of range.
イベントフラグの ID **番号に範囲外の値が指定されています。**

E2288 :　Eventflag id greater than max eventflag.
イベントフラグの ID **番号にイベントフラグの最大生成数よりも大きな値が指定されています。**

E2289 :　Initial pattern out of range.
イベントフラグの初期ビット・パターンに範囲外の値が指定されています。

E2290 :　Eventflag key-id out of range.
イベントフラグのキー ID **番号に範囲外の値が指定されています。**

E2291 :　Eventflag id is 0, but eventflag key-id not specified.
ID **番号自動生成のイベントフラグにシンボル名とキー** ID **番号のどちらも指定していません。**

E2292 :　Eventflag key-id "*key_id*" already defined.
イベントフラグのキー ID **番号** *key_id* **が二重定義されています。**

E2293 :　Mailbox id "*mbx_id*" already defined.
メールボックスの ID **番号** *mbx_id* **が二重定義されています。**

E2294 :　Non-protect mailbox id all assigned.
メールボックスの ID **番号として自動的に割り付け可能な値がすべて使用済みです。**

E2295 :　Too many mailboxes.
メールボックス情報の数がメールボックスの最大生成数を越えています。

E2296 :　Mailbox id out of range.
メールボックス ID **番号に範囲外の値が指定されています。**

E2297 :　Mailbox id greater than max mailbox.
メールボックス ID **番号にメールボックスの最大生成数よりも大きな値が指定されています。**

E2298 :　Mailbox key-id out of range.
メールボックスのキー ID **番号に範囲外の値が指定されています。**

E2299 :　Mailbox id is 0, but mailbox key-id not specified.
ID **番号自動生成のメールボックスにシンボル名とキー** ID **番号のどちらも指定していません。**

E2300 :　Mailbox key-id "*key_id*" already defined.
メールボックスのキー ID **番号** *key_id* **が二重定義されています。**

E2301 :　Memorypool id "*mpl_id*" already defined.
メモリ・プールの ID **番号** *mpl_id* **が二重定義されています。**

E2302 :　Non-protect memorypool id all assigned.
メモリ・プールの ID **番号として自動的に割り付け可能な値がすべて使用済みです。**

E2303 :　Too many memorypools.
メモリ・プール情報の数がメモリ・プールの最大生成数を越えています。

E2304 :　Memorypool id out of range.
メモリ・プールの ID **番号に範囲外の値が指定されています。**

E2305 :　Memorypool id greater than max memorypool.
メモリ・プールの ID **番号にメモリ・プールの最大生成数よりも大きな値が指定されています。**

E2306 :　Illegal system memorypool for memorypool.
システム・メモリの種類が不正です。

E2307 :　Memorypool key-id out of range.
メモリ・プールのキー ID **番号に範囲外の値が指定されています。**

E2308 :　Memorypool id is 0, but memorypool key-id not specified.
ID **番号自動生成のメモリ・プールにシンボル名とキー** ID **番号のどちらも指定していません。**

E2309 :　Memorypool key-id "*key_id*" already defined.
メモリ・プールのキー ID **番号** *key_id* **が二重定義されています。**

E2310 :　Interrupt handler number "*int_no*" already defined.
割り込みハンドラの割り込み要因番号 *int_no* **が二重定義されています。**

E2311 :　Interrupt handler number out of range.
割り込みハンドラの割り込み要因番号に範囲外の値が指定されています。

E2312 :　Cyclic handler number "*cyc_no*" already defined.
周期起動ハンドラの指定番号 *cyc_no* **が二重定義されています。**

E2313 :　Too many cyclic handlers.
周期起動ハンドラ情報の数が周期起動ハンドラの最大登録数を越えています。

E2314 :　Cyclic handler number out of range.
周期起動ハンドラの指定番号に範囲外の値が指定されています。

E2315 :　Cyclic handler number greater than max cyclic handler.
周期起動ハンドラの指定番号に周期起動ハンドラの最大登録数よりも大きな値が指定されています。

E2316 :　Interval time out of range.
周期起動ハンドラの起動時間間隔に範囲外の値が指定されています。

E2317 :　Svc handler number "*svc_no*" already defined.
拡張 SVC **ハンドラの拡張機能コード** *svc_no* **が二重定義されています。**

E2318 :　Too many svc handlers.
拡張 SVC **ハンドラ情報の数が拡張** SVC **ハンドラの最大登録数を越えています。**

E2319 :　Svc handler number out of range.
拡張 SVC **ハンドラの拡張機能コードに範囲外の値が指定されています。**

E2320 :　Svc handler number greater than max svc handler.
拡張 SVC **ハンドラの拡張機能コードに拡張** SVC **ハンドラの最大登録数よりも大きな値が指定されています。**

E2321 :　Initial handler not defined.
初期化ハンドラ情報が定義されていません。

E2322 :　Initial handler already defined.
初期化ハンドラが二重定義されています。

E2323 :　Illegal system call name.
システム・コール名が不正，または，別のグループのシステム・コールの利用を宣言しています。

E2324 :　Max interrupt handler not defined.
割り込みハンドラの最大登録数が定義されていません。

E2325 :　Max interrupt handler out of range.
割り込みハンドラの最大登録数に範囲外の値が指定されています。

E2326 :　Max interrupt handler already defined.
割り込みハンドラの最大登録数が二重定義されています。

E2327 :　Max interrupt handler greater than (max interrupt factor + 1).
割り込みハンドラの最大登録数に，最大割り込み要因番号 + 1 **よりも大きな値が指定されています。**

E2328 :　Too many interrupt handlers.
割り込みハンドラ情報の数が割り込みハンドラの最大登録数を越えています。

E2329 :　Interrupt factor is already assigned by clkhdr.
割り込みハンドラに指定した割り込み要因番号が，タイマの割り込み要因番号として指定されています。

E2330 :　Max interrupt factor not defined.
割り込み要因番号の最大値が定義されていません。

E2331 :　Max interrupt factor out of range.
割り込み要因番号の最大値に範囲外の値が指定されています。

E2332 :　Max interrupt factor already defined.
割り込み要因番号の最大値が二重定義されています。

E2333 :　Clock handler number not defined.
タイマの割り込み要因番号が定義されていません。

E2334 :　Clock handler number out of range.
タイマの割り込み要因番号に範囲外の値が指定されています。

E2335 :　Clock handler number already defined.
タイマの割り込み要因番号が二重定義されています。

E2336 :　"*chip_type*" cannot define.
システム情報に指定したプロセッサ種別の種類が不正です。

E2337 :　CPU type already defined.
ターゲット・デバイスのプロセッサ種別が二重定義されています。

7.4.3　警告

次に，警告が発生した際に出力されるメッセージを示します。
なお，メッセージ内のイタリック表記 (*name* 等) は，警告が発生した際に決定されるものです。

W2201 :　Task id is 0, but task key-id not specified.
タスクの ID **番号，および，キー** ID **番号に** 0x0 **が指定されています。**
CF850 Pro **は，タスクの** ID **番号保護範囲** *tsk_idlmt* **～タスクの最大生成数** *tsk_cnt* **の範囲で未使用の** ID **番号を自動的に割り当てます。**

W2202 :　System call "*svc_nam*" already defined.
SCT **情報中のユーザ処理プログラム内で使用するシステム・コールの宣言で** *svc_nam* **が二重定義されています。**
CF850 Pro **は，***svc_nam* **の二重定義を無視して処理を続行します。**

W2203 :　Cannot open command file "*cmd_file*".
コマンド・ファイル *cmd_file* **がオープンできません。**
CF850 Pro **は，起動オプション** *@cmd_file* **を無視して処理を続行します。**

W2204 :　Nested command file "*file_name*".
コマンド・ファイル *file_name* **で不正な起動オプション** *@cmd_file* **が定義されています。**
CF850 Pro **は，定義されている** *@cmd_file* **を無視して処理を続行します。**

W2205 :　cputype in CF file is different device file. (device file assumed)
起動オプション -cpu △ *name* **で指定された品種指定名とシステム情報で定義されたプロセッサ種別の整合性がとれていません。**
CF850 Pro **は，起動オプション** -cpu △ *name* **を有効情報として扱い処理を続行します。**

付録 A　ウインドウ・リファレンス

本付録では，CF850 Pro の起動オプションを CA850 が提供する統合開発環境プラットホーム PM+ から設定する際に必要となる各種ダイアログについて説明しています。

A.1　概要

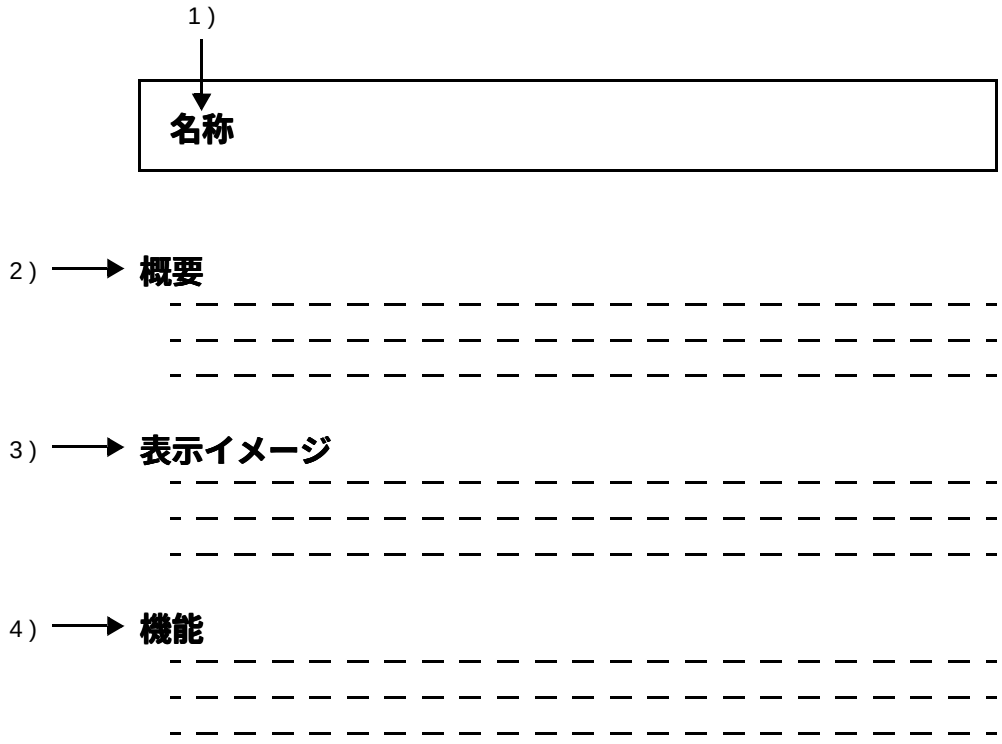
表 A-1 に，各種ダイアログの一覧を示します。

表 A-1　ダイアログの一覧

ダイアログ名	機能概要
[OS の選択] ダイアログ	以下に示した情報を入力するためのダイアログです。 - リアルタイム OS 名
[RX850 Pro 詳細設定] ダイアログ	以下に示した情報を CF850 Pro に対する起動オプションとして設定，統合開発環境プラットホーム PM+ に通知するためのダイアログです。 - システム・コンフィギュレーション・ファイル名 - システム情報テーブル・ファイル名 - システム・コール・テーブル・ファイル名 - システム情報ヘッダ・ファイル名 - ニュークリアス・ライブラリ名 - インタフェース・ライブラリ名 - ニュークリアス共通オブジェクト名
[システム・コンフィギュレーション・ファイルの指定] ダイアログ	既存のシステム・コンフィギュレーション・ファイルを読み込むためのダイアログです。
[RX850 Pro ERROR] ダイアログ	エラー情報を表示するためのダイアログです。

A.2　ウインドウ解説

次頁から，各種ダイアログについて，以下の記述フォーマットにしたがって解説します。



1）名称
ダイアログの名称を示しています。

2）概要
ダイアログの機能概要，および，オープン方法を示しています。

3）表示イメージ
ダイアログの表示イメージを示しています。

4）機能
ダイアログの機能詳細を構成要素別に示しています。

[OS の選択] ダイアログ

概要

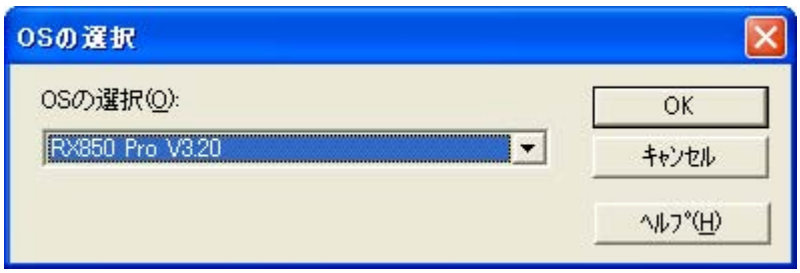
以下に示した情報を入力するためのダイアログです。

- リアルタイム OS 名

なお，本ダイアログは，次の方法でオープンすることができます。

- PM+ のメイン・ウインドウにおいて，[ツール (T)] → [OS の選択 (O)...] を選択

表示イメージ



機能

1) [OS の選択 (O)] エリア

- リスト・ボックス
  リアルタイム OS の名前 (リアルタイム OS 名) を選択します。
  なお，本リスト・ボックスにおいて指定可能なメニューは，RX850 Pro V3.*xx* に限られます。

2) 機能ボタン

- < OK > ボタン
  [RX850 Pro 詳細設定] ダイアログをオープンします。
- < キャンセル > ボタン
  本ダイアログを閉じます。
- < ヘルプ (H) > ボタン
  本ダイアログのオンライン・ヘルプをオープンします。

[RX850 Pro 詳細設定] ダイアログ

概要

以下に示した情報を CF850 Pro に対する起動オプションとして設定，統合開発環境プラットホーム PM+ に通知するためのダイアログです。

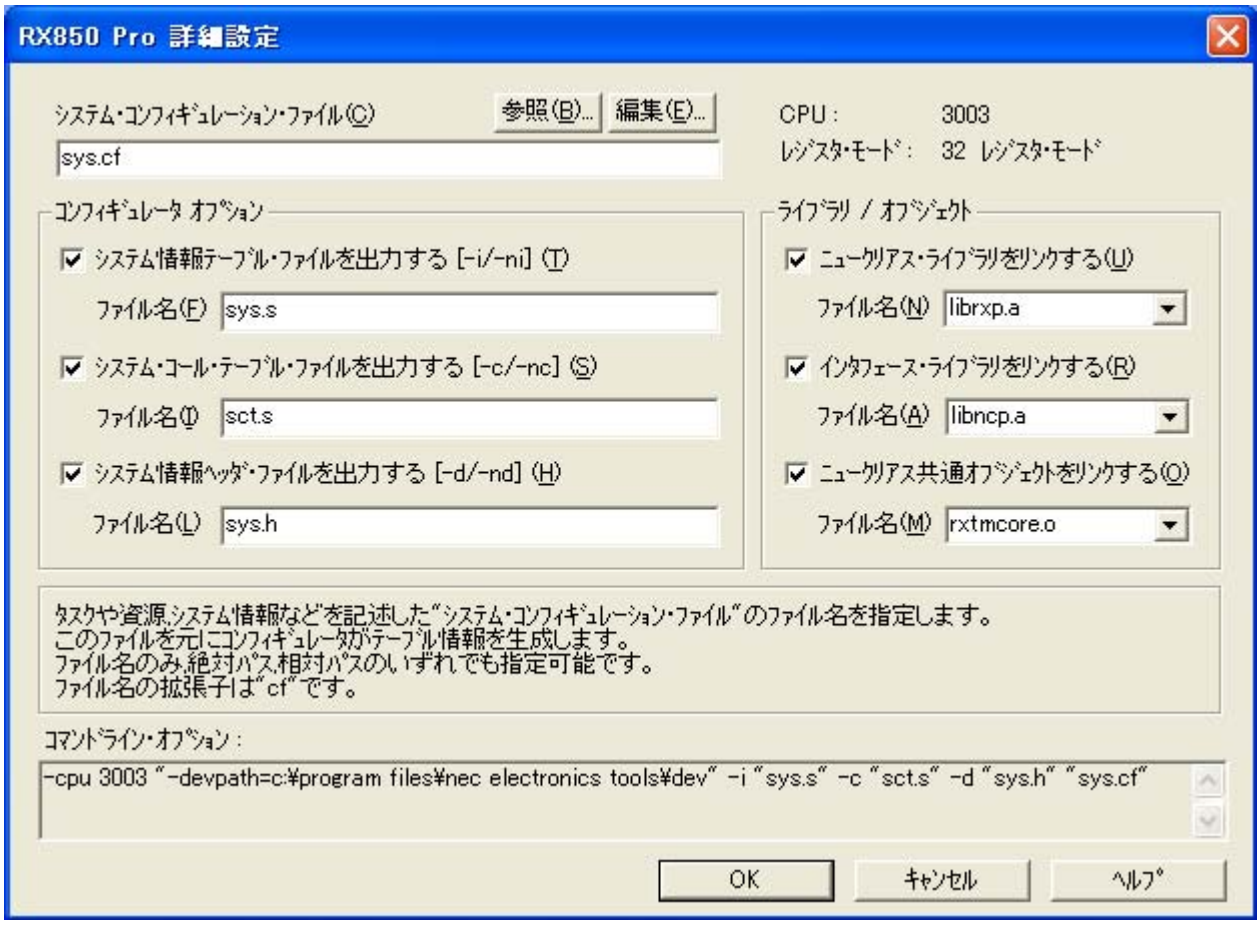
- システム・コンフィギュレーション・ファイル名
- システム情報テーブル・ファイル名
- システム・コール・テーブル・ファイル名
- システム情報ヘッダ・ファイル名
- ニュークリアス・ライブラリ名
- インタフェース・ライブラリ名
- ニュークリアス共通オブジェクト名

なお，本ダイアログは，次の方法でオープンすることができます。

- [OS の選択] ダイアログにおいて，“ RX850 Pro V3.*xx* ” を選択したあと，< OK > ボタンをクリック

備考　品種指定名，および，デバイス・ファイルの検索対象フォルダ名については，CA850 が提供する統合開発環境プラットホーム PM+ の [プロジェクトの設定] ダイアログで指定された情報が反映されるため，本ダイアログにおいて設定する必要がありません。
なお，[プロジェクトの設定] ダイアログについての詳細は，「PM+ ユーザーズ・マニュアル」を参照してください。

表示イメージ



機能

1) [システム・コンフィギュレーション・ファイル (C)] エリア

- テキスト・ボックス
  CF850 Pro への入力ファイル名 (システム・コンフィギュレーション・ファイル名) を指定します。

  注意　入力ファイル名として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。

- < 参照 (B)... > ボタン
  [システム・コンフィギュレーション・ファイルの指定] ダイアログをオープンします。

- < 編集 (E)... > ボタン
  [編集] ウインドウをオープンします。

  注意 1　[編集] ウインドウについての詳細は，「PM+ ユーザーズ・マニュアル」を参照してください。

  注意 2　[編集] ウインドウに表示されているシステム・コンフィギュレーション・ファイルを PM+ 付属のエディタ idea-L を用いて操作する際には，本ダイアログ，および，[OS の選択] ダイアログを閉じる必要があります。

2) [コンフィギュレータ・オプション] エリア

- [システム情報テーブル・ファイルを出力する [-i/-ni] (I)] チェック・ボックス
  CF850 Pro を起動した際，システム情報テーブル・ファイルを出力するか否かを指定します。

  チェック状態　：[ファイル名 (F)] エリアで指定されたファイル名でシステム情報テーブル・ファイルを出力します。
  非チェック状態　：システム情報テーブル・ファイルを出力しません。

- [ファイル名 (F)] エリア
  CF850 Pro を起動した際の出力ファイル名 (システム情報テーブル・ファイル名) を指定します。

  注意　出力ファイル名として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。

- [システム・コール・テーブル・ファイルを出力する [-c/-nc] (S)] チェック・ボックス
  CF850 Pro を起動した際，システム・コール・テーブル・ファイルを出力するか否かを指定します。

  チェック状態　：[ファイル名 (L)] エリアで指定されたファイル名でシステム・コール・テーブル・ファイルを出力します。
  非チェック状態　：システム・コール・テーブル・ファイルを出力しません。

- [ファイル名 (L)] エリア
  CF850 Pro を起動した際の出力ファイル名 (システム・コール・テーブル・ファイル名) を指定します。

  注意　出力ファイル名として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。

- [システム情報ヘッダ・ファイルを出力する [-d/-nd] (H)] チェック・ボックス
  CF850 Pro を起動した際，システム情報ヘッダ・ファイルを出力するか否かを指定します。

  チェック状態　：[ファイル名 (A)] エリアで指定されたファイル名でシステム情報ヘッダ・ファイルを出力します。
  非チェック状態　：システム情報ヘッダ・ファイルを出力しません。

- [ファイル名 (A)] エリア
  CF850 Pro を起動した際の出力ファイル名 (システム情報ヘッダ・ファイル名) を指定します。

  注意　出力ファイル名として指定可能な文字数は，パス名を含めて 255 文字以内に限られます。

3) [ライブラリ / オブジェクト] エリア

- [ニュークリアス・ライブラリをリンクする (U)] チェック・ボックス
  CF850 Pro を起動した際，[ファイル名（N)] コンボ・ボックスで指定されたニュークリアス・ライブラリを“ リンカ ld850 に対するリンク・オプション ” として PM+ に反映するか否かを指定します。

- [ファイル名（N)] コンボ・ボックス
  ロード・モジュールを作成する際にリンクするニュークリアス・ライブラリ名 (librxp.a，librxpm.a など) を指定します。

- [インタフェース・ライブラリをリンクする (R)] チェック・ボックス
  CF850 Pro を起動した際，[ファイル名（A)] コンボ・ボックスで指定されたインタフェース・ライブラリを“ リンカ ld850 に対するリンク・オプション ” として PM+ に反映するか否かを指定します。

- [ファイル名 (A)] コンボ・ボックス
  ロード・モジュールを作成する際にリンクするインタフェース・ライブラリ名 (libchp.a，libncp.a など) を指定します。
- [ニュークリアス共通オブジェクトをリンクする (O)] チェック・ボックス
  CF850 Pro を起動した際，[ファイル名 (M)] コンボ・ボックスで指定されたニュークリアス共通オブジェクト・ライブラリを “ リンカ ld850 に対するリンク・オプション ” として PM+ に反映するか否かを指定します。
- [ファイル名 (M)] コンボ・ボックス
  ロード・モジュールを作成する際にリンクするニュークリアス共通オブジェクト名 (rxcore.o，rxtmcore.o など) を指定します。

4) [コマンドライン・オプション] エリア
[システム・コンフィギュレーション・ファイル (C)] エリア，および，[コンフィギュレータ・オプション] エリアで指定された情報 (統合開発環境プラットホーム PM+ の [プロジェクトの設定] ダイアログで指定された情報を含む) を，CF850 Pro に対するコマンド入力形式として表示します。

5) 機能ボタン

- < OK > ボタン
  [システム・コンフィギュレーション・ファイル (C)] エリア，および，[コンフィギュレータ・オプション] エリアにおいて指定した情報 (統合開発環境プラットホーム PM+ の [プロジェクトの設定] ダイアログで指定された情報を含む) を “ CF850 Pro に対する起動オプション ” として，[ライブラリ / オブジェクト] エリアにおいて指定した情報を “ リンカ ld850 に対するリンク・オプション ” として PM+ に反映したあと，本ダイアログを閉じます。
- < キャンセル > ボタン
  本ダイアログを閉じます。
- < ヘルプ > ボタン
  本ダイアログのオンライン・ヘルプをオープンします。

[システム・コンフィギュレーション・ファイルの指定] ダイアログ

概要

既存のシステム・コンフィギュレーション・ファイルを読み込むためのダイアログです。
なお，本ダイアログは，次の方法でオープンすることができます。

- [RX850 Pro 詳細設定] ダイアログにおいて，< 参照 (B)... > ボタンをクリック

表示イメージ

機能

1) [ファイルの場所 (I)] エリア

- コンボ・ボックス
  システム・コンフィギュレーション・ファイルの格納されているフォルダ名を指定します。

2) ファイル名表示エリア
[ファイルの場所 (I)] エリア，および，[ファイルの種類 (T)] エリアで指定された情報をもとに，該当ファイルを表示します。

3) [ファイル名 (N)] エリア

- テキスト・ボックス
  読み込むシステム・コンフィギュレーション・ファイルのファイル名を指定します。

4) [ファイルの種類 (T)] エリア

- コンボ・ボックス
  ファイル名表示エリアに表示させるファイルの種類 (ファイル・タイプ) を指定します。

5) 機能ボタン

- < 開く (O) > ボタン
  [ファイルの場所 (I)] エリア，および，[ファイル名 (N)] エリアで指定されたシステム・コンフィギュレーション・ファイルを読み込みます。
- < キャンセル > ボタン
  本ダイアログを閉じます。

[RX850 Pro ERROR] ダイアログ

概要

エラー情報を表示するためのダイアログです。
なお，本ダイアログは，[RX850 Pro 詳細設定] ダイアログなどにおいて不正な情報が設定された際，自動的にオープンします。

表示イメージ

機能

1) メッセージ表示エリア
検出したエラーに対応したメッセージ (エラー情報：メッセージ・レベル / 識別番号，エラーの発生要因) を表示します。
以下に，本エリアに表示されるメッセージの一覧を示します。

メッセージ・レベル / 識別番号	エラーの発生要因
E1006:	ファイル名が長すぎます。
E1009:	同じファイル名に変更しようとしました。別のファイル名を指定してください。
E1010:	システム情報テーブル・ファイル名とシステム・コール・テーブル・ファイル名が同じです。どちらかを別のファイル名にしてください。
E1012:	ニュークリアス・ライブラリ・ファイル名とインタフェース・ライブラリ・ファイル名が同じです。どちらかを別のファイル名にしてください。
E1022:	システム・コンフィギュレーション・ファイル名，またはパス名が不正です。
E1023:	システム情報テーブル・ファイル名，またはパス名が不正です。
E1024:	システム情報ヘッダ・ファイル名，またはパス名が不正です。
E1025	システム・コール・テーブル・ファイル名，またはパス名が不正です。
E1027:	ニュークリアス・ライブラリ名が不正です。
E1028:	インタフェース・ライブラリ名が不正です。
E1029:	ニュークリアス共通オブジェクト名が不正です。
E2011:	システム・コンフィギュレーション・ファイル名が指定されていません。
E2012:	システム情報テーブル・ファイル名が指定されていません。
E2013	システム・コール・テーブル・ファイル名が指定されていません。
E2014:	システム情報ヘッダ・ファイル名が指定されていません。
E2016:	ニュークリアス・ライブラリ名が指定されていません。

メッセージ・レベル / 識別番号	エラーの発生要因
E2017:	インタフェース・ライブラリ名が指定されていません。
E2018:	ニュークリアス共通オブジェクト名が指定されていません。
E2030:	32 レジスタ・モードが指定されていません。RX850 Pro を使用する場合は 32 レジスタ・モードに変更してください。
E2040:	オンライン・ヘルプ・ファイルが見つかりません。

2) **機能ボタン**

- < OK > **ボタン**
  本ダイアログを閉じます。

索引

C

M

O

P

R

S

T

あ

い

う

か

き

く

け

こ

し

せ

そ

た

ち

て

と

に

は

ひ

ふ

ほ

み

め

り

ろ

わ

〔メ　モ〕

【発　行】
NECエレクトロニクス株式会社
〒211-8668　神奈川県川崎市中原区下沼部1753
電話（代表）：044(435)5111

お問い合わせ先

【ホームページ】

NECエレクトロニクスの情報がインターネットでご覧になれます。
URL(アドレス)　http://www.necel.co.jp/

【営業関係，技術関係お問い合わせ先】

半導体ホットライン
（電話：午前 9:00～12:00，午後 1:00～5:00）
電　話　：044−435−9494
E-mail　：info@necel.com

【資料請求先】

NECエレクトロニクスのホームページよりダウンロードいただくか，NECエレクトロニクスの販売特約店へお申し付けください。