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

保守／廃止

RX830 （μITRON Ver.3.0）

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

対象デバイス

V830 ファミリ™

資料番号 U13151JJ2V0UM00 （第2版）
発行年月 June 1998 CP(K)

保守／廃止

〔メ　モ〕

保守／廃止

商標等について

- V800 シリーズ ™，および，V830 ファミリ ™ は，日本電気株式会社の商標です。
- TRON は，The Realtime Operating system Nucleus の略称です。
- ITRON は，Industrial TRON の略称です。
- その他，記載の会社名／製品名は，各社の商標あるいは登録商標です。

保守／廃止

はじめに

マイクロプロセッサは半導体技術の進歩に伴って急速に普及し，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合わせた固有のプログラムをその都度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・コンピュータなど）することができるため，標準的なOS(Windows 95，UNIX OSなど）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されます。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では，このような市場状況を考慮し，V800シリーズ™V830ファミリ™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，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，RX830を開発，発売しました。

RX830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，一層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。

保守／廃止

〔メ　モ〕

保守／廃止

目次

表目次

保守／廃止

図目次

保守／廃止

第１章　概説

1.1　概要

RX830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

以下に，RX830の特徴を示します。

(1) μITRON3.0仕様に準拠

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

μITRON3.0仕様で規定されているシステム・コール(66種類)のほかに，RX830オリジナルのシステム・コール(7種類)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，RX830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)のみをシステム構築時に選択することができるため，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を構築することができます。

(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，豊富な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
- 同期通信機能
- 割り込み管理機能
- 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，コンパクトな設計が行われています。

(6) オリジナル命令の活用

V830 ファミリ ™ 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 RAM の活用

V830 ファミリ ™ の内蔵 RAM(内蔵命令メモリ，内蔵データ・メモリ）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) 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 2 つのユーティリティを提供しています。

- コンフィギュレータ CF830
- 高級言語インタフェース・ライブラリ

(9) C コンパイラ

以下に示す V830 ファミリ ™ 用 C コンパイラに対応しています。

- CA830　　NEC 製
- CCV830　　米国 Green Hills Software, Inc. 製

1.3 実行環境

以下に，RX830が処理を実行するうえで必要となる環境を示します。

- プロセッサ

  V830ファミリ™

- 周辺コントローラ

  RX830では，実行環境のハードウエア構成に依存する部分（システム初期化処理：ブート処理，ハードウエア初期化部）については，サンプル・ソース・ファイルを提供しています。

  このため，システム初期化処理を各ターゲット・システム用に書き替えることで，特定の周辺コントローラは要求しません。

- メモリ容量

  以下に，RX830が処理を実行するうえで必要となるメモリ容量を示します。

  ニュークリアス・テキスト部 ： 約4 ～ 11 Kバイト
  ニュークリアス・データ部 ： 約1 ～ 2 Kバイト

  なお，上記に示したメモリ容量の最大値は，コンフィギュレーション時に，「優先度の指定範囲を最大に設定する」，「RX830が提供する機能（システム・コール）をすべて使用する」を指定した場合のものであり，優先度の指定範囲や機能に制限を設けることにより，必要となるメモリ容量を小さくすることが可能です。

1.4 開発環境

以下に，アプリケーション・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，および，ソフトウエア環境を示します。

- ハードウエア環境
  - ホスト・マシン
    - * PC-9800 シリーズ　Windows 95
    - * IBM-PC/AT 互換機　Windows 95
    - * SPARC station™　SunOS™ Rel 4.1.x
    - * SPARC station™　Solaris™ Rel 2.5.x
  - インサーキット・エミューレータ
    - * IE-705100-MC-EM1　V830 用
    - * IE-70000-MC-NW　V831 用
  - ネットワーク・モジュール
    - * IE-70000-MC-SV2
- ソフトウエア環境
  - C コンパイラ
    - * CA830　NEC 製
    - * CCV830　米国 Green Hills Software, Inc. 製
  - ディバッガ
    - * ID830　NEC 製
    - * MULTI　米国 Green Hills Software, Inc. 製
  - タスク・ディバッガ
    - * RD830　NEC 製
  - システム・パフォーマンス・アナライザ
    - * AZ830　NEC 製

第 2 章　インストレーション

この章では，RX830 の提供媒体に格納されているファイル群をユーザの開発環境上にインストールする際の手順について解説しています。

2.1　概要

RX830 の提供媒体は，ユーザの開発環境（ホスト・マシン）にあわせて，Windows ベースと UNIX ベースの 2 種類が用意されています。

表 2-1に，開発環境と提供媒体の対応を示します。

表 2-1　提供媒体の種類

開発環境（ホスト・マシン）	提供媒体
Windows ベース • PC-9800 シリーズ • IBM-PC/AT 互換機	フロッピィ・ディスク（FD）
UNIX ベース • SPARC station™	フロッピィ・ディスク（FD） または カートリッジ磁気テープ（CGMT）

2.2　インストール手順

RX830 の提供媒体に格納されているファイル群をユーザの開発環境上にインストールする際の手順は，ユーザの開発環境により異なります。

そこで，次節以降に，ユーザの開発環境が Windows ベースの場合のインストール手順と，UNIX ベースの場合のインストール手順をそれぞれに示します。

2.3 Windows ベースのインストール

ユーザの開発環境が Windows ベース (PC-9800 シリーズ, IBM-PC/AT 互換機) の場合, 次の手順で RX830 のインストールを行います。

ただし, 入力例中の "A>" はシェル・プロンプトを, "Δ" は 1 個以上の空白を, "↵" はリターン・キーの入力を表しています。

(1) Windows 95 の起動

パソコン本体, および, 周辺機器等の電源を投入し, Windows 95 を起動します。

(2) MS-DOS プロンプトのオープン

MS-DOS プロンプトをオープンします。

注意 RX830 のインストール作業は, MS-DOS プロンプト上で行います。

(3) ディレクトリの移動

cd コマンドを実行し, インストール用ディレクトリに移動します。

なお, 以下の入力例では, インストール用ディレクトリに A ドライブのルート・ディレクトリ A:¥ を指定しています。

注意 RX830 のインストールに際しては, 約 1 Mbyte の空きディスク領域が必要となります。

```
A>  cd Δ a:¥ ↵
```

(4) 提供媒体のセット

提供媒体 (フロッピィ・ディスク) をフロッピィ・ディスク・ドライブにセットします。

なお, ここでは, 提供媒体を C ドライブにセットしたものとします。

(5) ファイル群のインストール

xcopy コマンドを実行し, 提供媒体に格納されているファイル群を, ユーザの開発環境上にインストールします。

ただし, 提供媒体には, 以下に示した 2 種類の RX830 が格納されています。

- CA830 対応版
- CCV830 対応版

そこで, 使用する C コンパイラ・パッケージが CA830 の場合は CA830 対応版の RX830 を, CCV830 の場合は CCV830 対応版の RX830 をインストールします。

[CA830 対応版の場合]

```
A>  xcopy Δ c:¥nectools Δ . Δ /s Δ /e Δ /v ↵
```

[CCV830 対応版の場合]

```
A>  xcopy Δ c:¥ghstools Δ . Δ /s Δ /e Δ /v ↵
```

(6) ファイル群の確認

dir コマンドを実行し，提供媒体に格納されていたファイル群がユーザの開発環境上にインストールされたことを確認します。

なお，各ディレクトリについての詳細は，「2.5　ディレクトリ構成」を参照してください。

[CA830 対応版の場合]

```
A>  dir Δ /s Δ a:¥nectools ↵
```

[CCV830 対応版の場合]

```
A>  dir Δ /s Δ a:¥ghstools ↵
```

(7) コマンド・サーチ・パスの設定

RX830 が提供するユーティリティ・ツール（コンフィギュレータ CF830）に対するコマンド・サーチ・パスを設定します。

なお，以下の記述例では，環境設定ファイル autoexec.bat 内の path 変数に設定する際の方法を示しています。

[CCV830 対応版の場合]

```
path = %path%;a:¥ghstools¥bin
```

注意　CA830 対応版のコンフィギュレータは VSH からの起動となるため，NEC 製 V830 ファミリ™ 用 C コンパイラ・パッケージ CA830 を使用した場合，“コマンド・サーチ・パスの設定”に関する操作を行う必要はありません。

2.4 UNIX ベースのインストール

ユーザの開発環境が UNIX ベース (SPARC station™) の場合，次の手順で RX830 のインストールを行います。

ただし，入力例中の "%" はシェル・プロンプトを，"Δ" は 1 個以上の空白を，"↵" はリターン・キーの入力を表しています。

(1) 開発環境へのログイン

開発環境にログインします。

```
%
```

(2) ディレクトリの移動

cd コマンドを実行し，インストール用ディレクトリに移動します。

なお，以下の入力例では，インストール用ディレクトリに /usr/local を指定しています。

注意 RX830 のインストールに際しては，約 1 Mbyte の空きディスク領域が必要となります。
また，インストール用ディレクトリのパーミッション (read, write, execute) が許可状態であることを確認してください。

```
% cd Δ /usr/local ↵
```

(3) 提供媒体のセット

提供媒体 (フロッピィ・ディスク，または，カートリッジ磁気テープ) を該当する装置 (フロッピィ・ディスク・ドライブ，カートリッジ磁気テープ装置など) にセットします。

(4) ファイル群のインストール

tar コマンドを実行し，提供媒体に格納されているファイル群を，ユーザの開発環境上にインストールします。

ただし，提供媒体には，以下に示した 2 種類の RX830 が格納されています。

- CA830 対応版
- CCV830 対応版

そこで，使用する C コンパイラ・パッケージが CA830 の場合は CA830 対応版の RX830 を，CCV830 の場合は CCV830 対応版の RX830 をインストールします。

なお，以下の入力例では，スペシャル・ファイル名に /dev/rst8 を指定しています。

[CA830 対応版の場合]

```
% tar △ -xvof △ /dev/rst8 △ nectools ↵
```

[CCV830 対応版の場合]

```
% tar △ -xvof △ /dev/rst8 △ ghstools ↵
```

(5) ファイル群の確認

ls コマンドを実行し，提供媒体に格納されていたファイル群がユーザの開発環境上にインストールされたことを確認します。

なお，各ディレクトリについての詳細は，「2.5　ディレクトリ構成」を参照してください。

[CA830 対応版の場合]

```
% ls △ -CFR △ /usr/local/nectools ↵
```

[CCV830 対応版の場合]

```
% ls △ -CFR △ /usr/local/ghstools ↵
```

(6) コマンド・サーチ・パスの設定

RX830 が提供するユーティリティ・ツール (コンフィギュレータ CF830) に対するコマンド・サーチ・パスを設定します。

なお，以下の記述例では，環境設定ファイル.cshrc 内の path 変数に設定する際の方法を示しています。

[CA830 対応版の場合]

```
set  path = (  $path /usr/local/nectools/bin  )
```

[CCV830 対応版の場合]

```
set  path = (  $path /usr/local/ghstools/bin  )
```

2.5 ディレクトリ構成

RX830の提供媒体には，ソース・ファイル提供形式と，オブジェクト・ファイル提供形式の2種類があり，それぞれにディレクトリ構成が異なります。

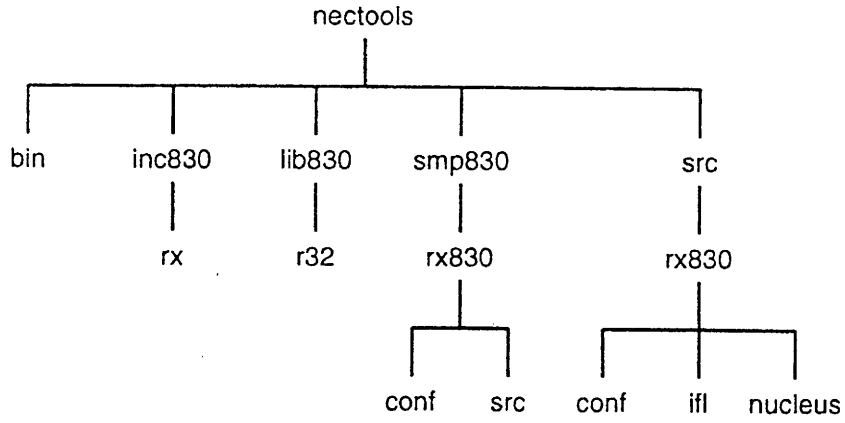
そこで，次節以降に，ソース・ファイル提供形式のディレクトリ構成と，オブジェクト・ファイル提供形式のディレクトリ構成をそれぞれに示します。

2.6 ソース・ファイル提供形式

2.6.1 CA830対応版の場合

図2-1に，ソース・ファイル提供形式のディレクトリ構成を示します。

図 2-1 ソース・ファイル提供形式のディレクトリ構成 CA830対応版



(1) nectools

CA850対応版のRX830が格納されているディレクトリです。

(2) nectools¥bin

RX830が提供するユーティリティ・ツール(コンフィギュレータ CF830)が格納されているディレクトリです。

(3) nectools¥inc830

ヘッダ・ファイルが格納されているディレクトリです。

stdrx.h ：標準ヘッダ・ファイル(C言語ソース・ファイル用)
本ヘッダ・ファイルをインクルードすることにより，RX830を使用するうえで必要となるすべてのヘッダ・ファイルがインクルードされます。

stdrx.inc：標準ヘッダ・ファイル(アセンブリ言語ソース・ファイル用)
本ヘッダ・ファイルをインクルードすることにより，RX830を使用するうえで必要となるすべてのヘッダ・ファイルがインクルードされます。

(4) nectools¥inc830¥rx

RX830 のヘッダ・ファイルが格納されているディレクトリです。

(5) nectools¥lib830

オブジェクト・ファイル，および，ライブラリ・ファイルが格納されているディレクトリです。

(6) nectools¥lib830¥r32

RX830 のオブジェクト・ファイル (32 レジスタ・モード)，および，ライブラリ・ファイル (32 レジスタ・モード) が格納されているディレクトリです。

rxcore.o：ニュークリアス共通部
librx.a ：ニュークリアス・ライブラリ
libch.a ：システム・コール用インタフェース・ライブラリ (チェック機能あり)
libnc.a ：システム・コール用インタフェース・ライブラリ (チェック機能なし)

(7) nectools¥smp830

サンプル・プログラムが格納されているディレクトリです。

(8) nectools¥smp830¥rx830

RX830 のサンプル・プログラムが格納されているディレクトリです。

(9) nectools¥smp830¥rx830¥conf

サンプル・プログラムからロード・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ディレクトリです。

なお，本ディレクトリにおいて，コマンド・ファイルを実行することにより，以下に示すロード・モジュールがカレント・ディレクトリ nectools¥smp830¥rx830¥conf に生成されます。

sample.out：ロード・モジュール (ROM 化情報を含まない)
sample.rom：ロード・モジュール (ROM 化情報を含む)
sample.hex：ロード・モジュール (HEX 形式)

(10) nectools¥smp830¥rx830¥src

サンプル・プログラムのソース・ファイルが格納されているディレクトリです。

(11) nectools¥src

ソース・ファイルが格納されているディレクトリです。

(12) nectools¥src¥rx830

RX830 のソース・ファイルが格納されているディレクトリです。

(13) nectools¥src¥rx830¥conf

RX830 のオブジェクト・ファイル (32 レジスタ・モード), および, ライブラリ・ファイル (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ディレクトリです。

なお, 本ディレクトリにおいて, コマンド・ファイルを実行することにより, 以下に示すオブジェクト・ファイル (32 レジスタ・モード), および, ライブラリ・ファイル (32 レジスタ・モード) がオブジェクト・ファイル／ライブラリ・ファイル格納ディレクトリ nectools¥lib830¥r32 に生成されます。

rxcore.o：ニュークリアス共通部
librx.a ：ニュークリアス・ライブラリ
libch.a ：システム・コール用インタフェース・ライブラリ (チェック機能あり)
libnc.a ：システム・コール用インタフェース・ライブラリ (チェック機能なし)

(14) nectools¥src¥rx830¥ifl

システム・コール用インタフェース・ライブラリ (チェック機能あり, および, チェック機能なし) のソース・ファイルが格納されているディレクトリです。

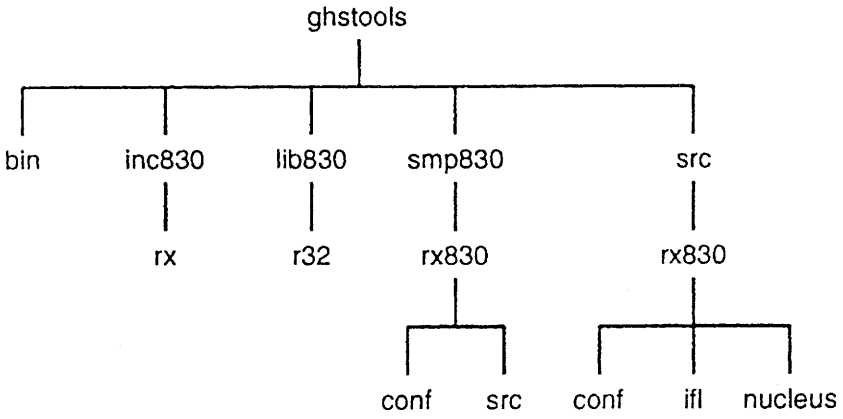
(15) nectools¥src¥rx830¥nucleus

ニュークリアス共通部, および, ニュークリアス・ライブラリのソース・ファイルが格納されているディレクトリです。

2.6.2　CCV830 対応版の場合

図 2-2に，ソース・ファイル提供形式のディレクトリ構成を示します。

図 2-2　ソース・ファイル提供形式のディレクトリ構成　CCV830 対応版



(1) ghstools

CCV850 対応版の RX830 が格納されているディレクトリです。

(2) ghstools¥bin

RX830 が提供するユーティリティ・ツール（コンフィギュレータ　CF830）が格納されているディレクトリです。

(3) ghstools¥inc830

ヘッダ・ファイルが格納されているディレクトリです。

stdrx.h：標準ヘッダ・ファイル
本ヘッダ・ファイルをインクルードすることにより，RX830 を使用するうえで必要となるすべてのヘッダ・ファイルがインクルードされます。

(4) ghstools¥inc830¥rx

RX830 のヘッダ・ファイルが格納されているディレクトリです。

(5) ghstools¥lib830

オブジェクト・ファイル，および，ライブラリ・ファイルが格納されているディレクトリです。

(6) ghstools¥lib830¥r32

RX830 のオブジェクト・ファイル（32 レジスタ・モード），および，ライブラリ・ファイル（32 レジスタ・モード）が格納されているディレクトリです。

rxcore.o：ニュークリアス共通部
librx.a ：ニュークリアス・ライブラリ
libch.a ：システム・コール用インタフェース・ライブラリ（チェック機能あり）
libnc.a ：システム・コール用インタフェース・ライブラリ（チェック機能なし）

(7) ghstools¥smp830

サンプル・プログラムが格納されているディレクトリです。

(8) ghstools¥smp830¥rx830

RX830 のサンプル・プログラムが格納されているディレクトリです。

(9) ghstools¥smp830¥rx830¥conf

サンプル・プログラムからロード・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ディレクトリです。

なお，本ディレクトリにおいて，コマンド・ファイルを実行することにより，以下に示すロード・モジュールがカレント・ディレクトリ ghstools¥smp830¥rx830¥conf に生成されます。

sample.rom：ロード・モジュール（ROM 化情報を含む）
sample.hex：ロード・モジュール（HEX 形式）

(10) ghstools¥smp830¥rx830¥src

サンプル・プログラムのソース・ファイルが格納されているディレクトリです。

(11) ghstools¥src

ソース・ファイルが格納されているディレクトリです。

(12) ghstools¥src¥rx830

RX830 のソース・ファイルが格納されているディレクトリです。

(13) ghstools¥src¥rx830¥conf

RX830 のオブジェクト・ファイル（32 レジスタ・モード），および，ライブラリ・ファイル（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ディレクトリです。

なお，本ディレクトリにおいて，コマンド・ファイルを実行することにより，以下に示すオブジェクト・ファイル（32 レジスタ・モード），および，ライブラリ・ファイル（32 レジスタ・モード）がオブジェクト・ファイル／ライブラリ・ファイル格納ディレクトリ ghstools¥lib830¥r32 に生成されます。

rxcore.o：ニュークリアス共通部
librx.a ：ニュークリアス・ライブラリ
libch.a ：システム・コール用インタフェース・ライブラリ（チェック機能あり）
libnc.a ：システム・コール用インタフェース・ライブラリ（チェック機能なし）

(14) ghstools¥src¥rx830¥ifl

システム・コール用インタフェース・ライブラリ（チェック機能あり，および，チェック機能なし）のソース・ファイルが格納されているディレクトリです。

(15) ghstools¥src¥rx830¥nucleus

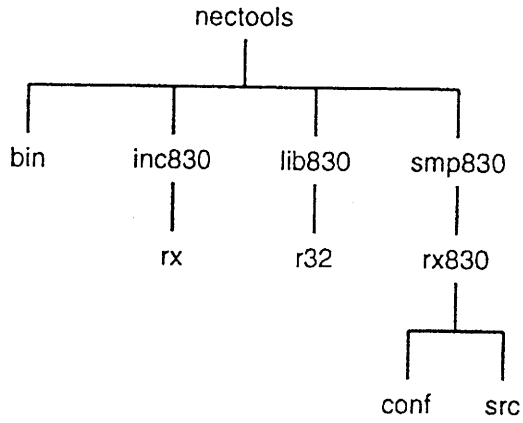
ニュークリアス共通部，および，ニュークリアス・ライブラリのソース・ファイルが格納されているディレクトリです。

2.7　オブジェクト・ファイル提供形式

2.7.1　CA830 対応版の場合

図 2-3に，オブジェクト・ファイル提供形式のディレクトリ構成を示します。

図 2-3　オブジェクト・ファイル提供形式のディレクトリ構成　CA830 対応版



(1) nectools

CA850 対応版の RX830 が格納されているディレクトリです。

(2) nectools¥bin

RX830 が提供するユーティリティ・ツール（コンフィギュレータ　CF830）が格納されているディレクトリです。

(3) nectools¥inc830

ヘッダ・ファイルが格納されているディレクトリです。

stdrx.h　：標準ヘッダ・ファイル（C 言語ソース・ファイル用）
本ヘッダ・ファイルをインクルードすることにより，RX830 を使用するうえで必要となるすべてのヘッダ・ファイルがインクルードされます。

stdrx.inc：標準ヘッダ・ファイル（アセンブリ言語ソース・ファイル用）
本ヘッダ・ファイルをインクルードすることにより，RX830 を使用するうえで必要となるすべてのヘッダ・ファイルがインクルードされます。

(4) nectools¥inc830¥rx

RX830 のヘッダ・ファイルが格納されているディレクトリです。

(5) nectools¥lib830

オブジェクト・ファイル，および，ライブラリ・ファイルが格納されているディレクトリです。

(6) nectools¥lib830¥r32

RX830 のオブジェクト・ファイル (32 レジスタ・モード), および, ライブラリ・ファイル (32 レジスタ・モード) が格納されているディレクトリです。

rxcore.o：ニュークリアス共通部
librx.a ：ニュークリアス・ライブラリ
libch.a ：システム・コール用インタフェース・ライブラリ (チェック機能あり)
libnc.a ：システム・コール用インタフェース・ライブラリ (チェック機能なし)

(7) nectools¥smp830

サンプル・プログラムが格納されているディレクトリです。

(8) nectools¥smp830¥rx830

RX830 のサンプル・プログラムが格納されているディレクトリです。

(9) nectools¥smp830¥rx830¥conf

サンプル・プログラムからロード・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ディレクトリです。

なお, 本ディレクトリにおいて, コマンド・ファイルを実行することにより, 以下に示すロード・モジュールがカレント・ディレクトリ nectools¥smp830¥rx830¥conf に生成されます。

sample.out：ロード・モジュール (ROM 化情報を含まない)
sample.rom：ロード・モジュール (ROM 化情報を含む)
sample.hex：ロード・モジュール (HEX 形式)

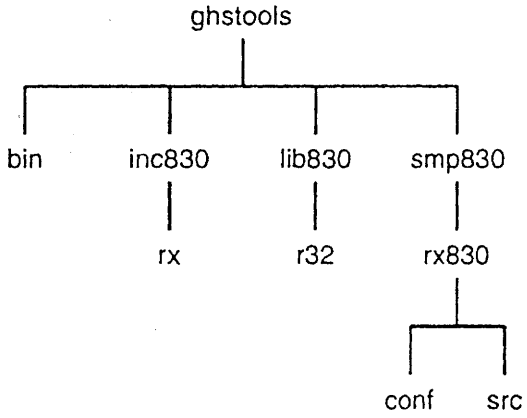
(10) nectools¥smp830¥rx830¥src

サンプル・プログラムのソース・ファイルが格納されているディレクトリです。

2.7.2 CCV830 対応版の場合

図 2-4に，オブジェクト・ファイル提供形式のディレクトリ構成を示します。

図 2-4 オブジェクト・ファイル提供形式のディレクトリ構成 CCV830 対応版



(1) ghstools

CCV850 対応版の RX830 が格納されているディレクトリです。

(2) ghstools¥bin

RX830 が提供するユーティリティ・ツール（コンフィギュレータ CF830）が格納されているディレクトリです。

(3) ghstools¥inc830

ヘッダ・ファイルが格納されているディレクトリです。

stdrx.h：標準ヘッダ・ファイル
本ヘッダ・ファイルをインクルードすることにより，RX830 を使用するうえで必要となるすべてのヘッダ・ファイルがインクルードされます。

(4) ghstools¥inc830¥rx

RX830 のヘッダ・ファイルが格納されているディレクトリです。

(5) ghstools¥lib830

オブジェクト・ファイル，および，ライブラリ・ファイルが格納されているディレクトリです。

(6) ghstools¥lib830¥r32

RX830 のオブジェクト・ファイル（32 レジスタ・モード），および，ライブラリ・ファイル（32 レジスタ・モード）が格納されているディレクトリです。

rxcore.o：ニュークリアス共通部
librx.a ：ニュークリアス・ライブラリ
libch.a ：システム・コール用インタフェース・ライブラリ（チェック機能あり）
libnc.a ：システム・コール用インタフェース・ライブラリ（チェック機能なし）

(7) ghstools¥smp830

サンプル・プログラムが格納されているディレクトリです。

(8) ghstools¥smp830¥rx830

RX830 のサンプル・プログラムが格納されているディレクトリです。

(9) ghstools¥smp830¥rx830¥conf

サンプル・プログラムからロード・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ディレクトリです。

なお，本ディレクトリにおいて，コマンド・ファイルを実行することにより，以下に示すロード・モジュールがカレント・ディレクトリ ghstools¥smp830¥rx830¥conf に生成されます。

sample.rom：ロード・モジュール（ROM 化情報を含む）
sample.hex：ロード・モジュール（HEX 形式）

(10) ghstools¥smp830¥rx830¥src

サンプル・プログラムのソース・ファイルが格納されているディレクトリです。

第 3 章　システム構築

この章では，RX830 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解説しています。

3.1　概要

システム構築とは，RX830 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のち，ターゲット・システムへ組み込むことです。

以下に，システムを構築する際の手順を示します。

(1) CF 定義ファイルの作成

(2) ユーザ・オウン・コーディング部の作成

- 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／例外エントリ
- クロック割り込みの後処理
- I/O ポート操作

(3) 処理プログラムの作成

- 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
- 拡張 SVC ハンドラ用インタフェース・ライブラリ

(4) 初期化データの退避領域の作成

(5) リンク・ディレクティブ・ファイルの作成

(6) ロード・モジュールの作成

- ROM 化情報を含まない
- ROM 化情報を含む

- HEX 形式

(7) ターゲット・システムへの組み込み

注意　米国 Green Hills Software, Inc. 製 C クロス V800™ コンパイラ・パッケージ CCV830 を使用した場合，“初期化データの退避領域”を作成する必要はありません。

図 3-1～図 3-2に，システムを構築する際の手順を示します。

ただし，図 3-1は，NEC 製 V830 ファミリ™ 用 C コンパイラ・パッケージ CA830 を使用した場合のシステム構築手順であり，図 3-2は，米国 Green Hills Software, Inc. 製 C クロス V800™ コンパイラ・パッケージ CCV830 を使用した場合のシステム構築手順です。

保守／廃止

図 3-1　システム構築手順 CA830

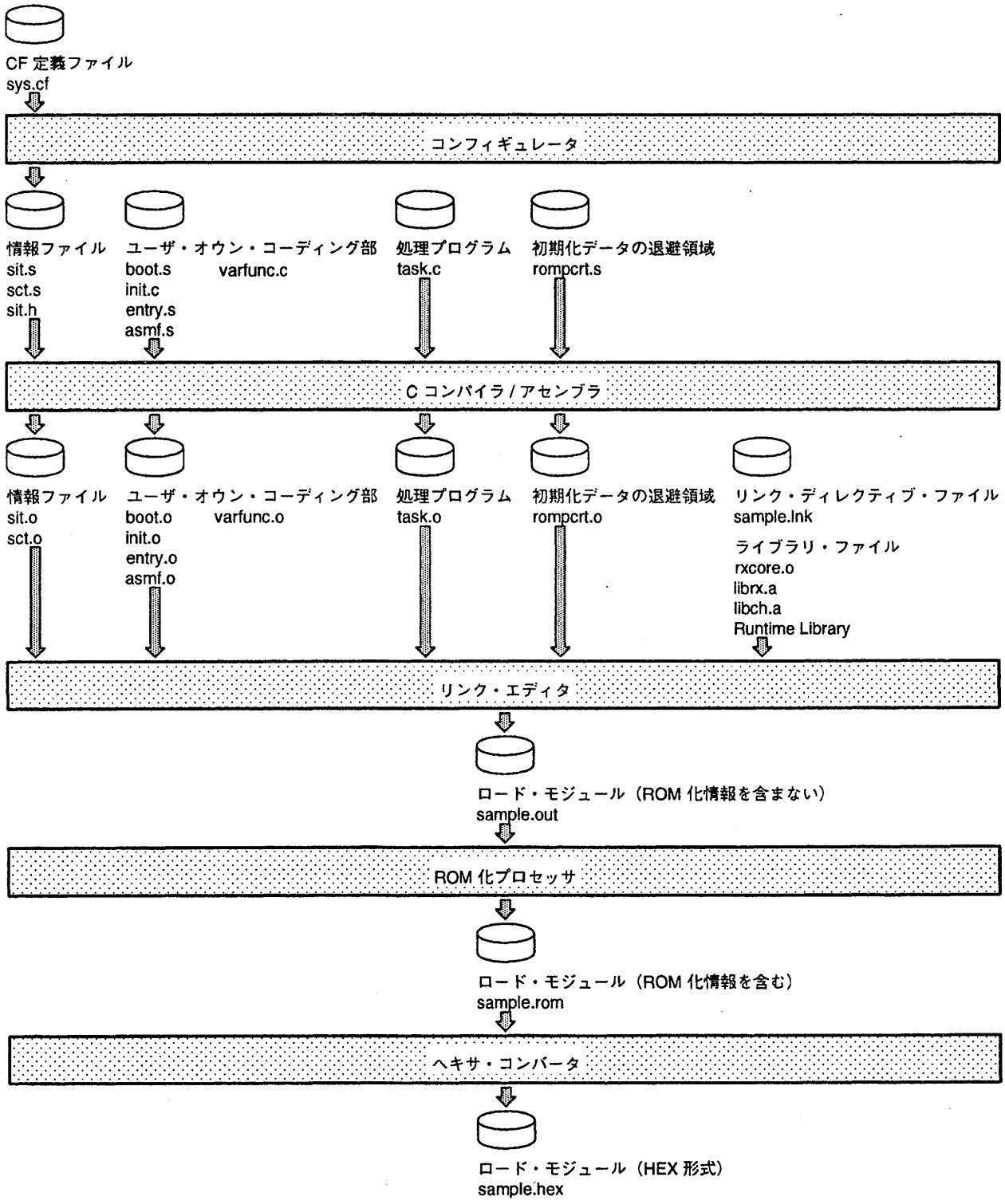
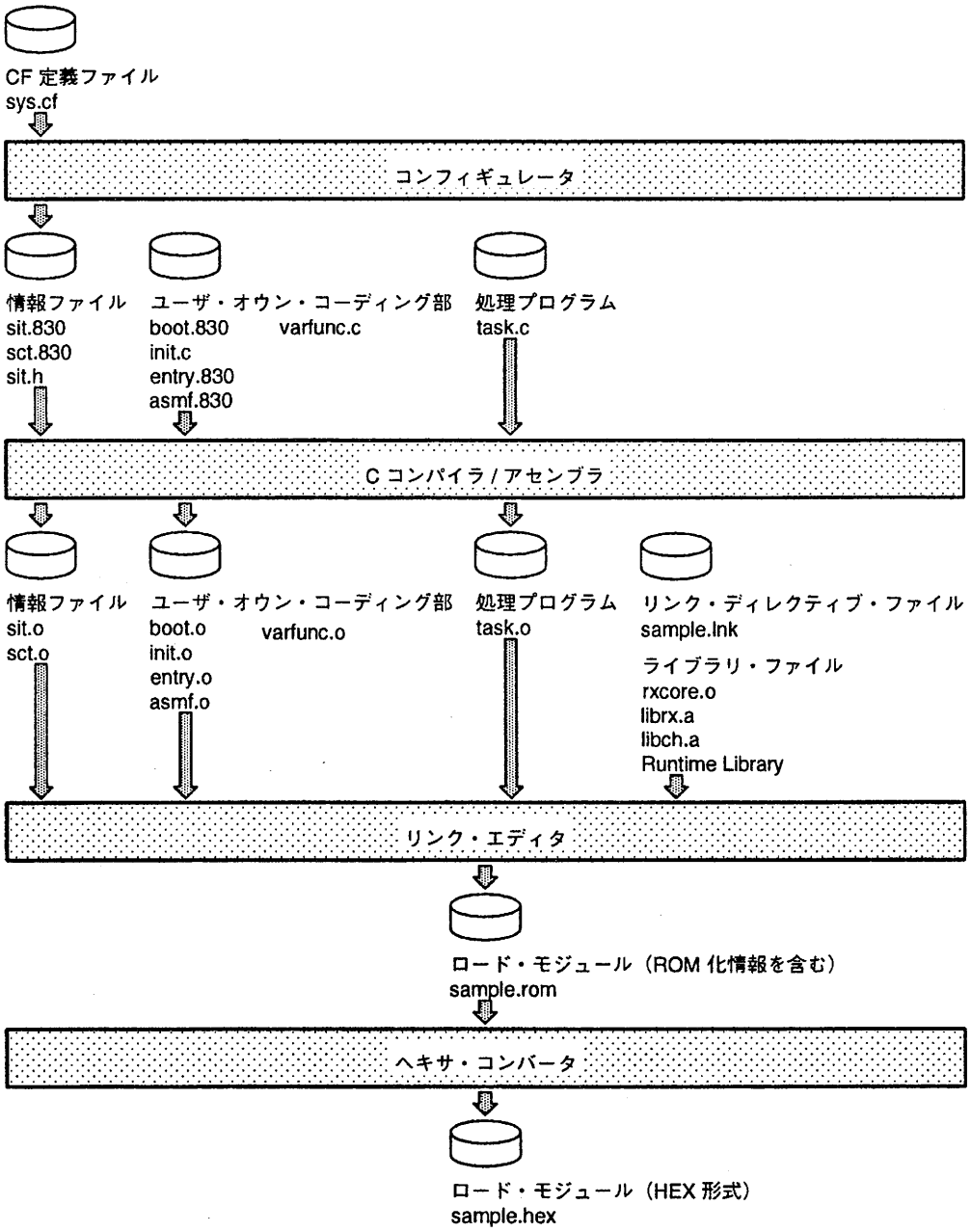


図 3-2　システム構築手順 CCV830



3.2 CF 定義ファイルの作成

RX830 に提供する各種データを保持したファイル（CF 定義ファイル）を作成します。
以下に，CF 定義ファイルを作成する際に必要となるデータを示します。

- リアルタイム OS 情報
  - RX シリーズ情報
- SIT 情報
  - システム情報
  - システム最大値情報
  - メモリ情報
  - タスク情報
  - セマフォ情報
  - イベント・フラグ情報
  - メールボックス情報
  - 割り込みハンドラ情報
  - メモリ・プール情報
  - 周期起動ハンドラ情報
  - 拡張 SVC ハンドラ情報
  - 初期化ハンドラ情報
- SCT 情報
  - タスク管理／タスク付属同期機能情報
  - タスク通信（セマフォ）機能情報
  - タスク通信（イベント・フラグ）機能情報
  - タスク通信（メールボックス）機能情報
  - 割り込み管理機能情報
  - メモリ・プール管理機能情報
  - 時間管理機能情報
  - システム管理機能情報

注 1　標準添付の CF 定義ファイルは，

`nectools¥smp830¥rx830¥src¥sys.cf` … CA830 対応版

`ghstools¥smp830¥rx830¥src¥sys.cf` … CCV830 対応版

に記述されています。

注 2　CF 定義ファイルの記述形式についての詳細は，「第 8章　CF 定義ファイルの記述方法」を参照してください。

3.3 ユーザ・オウン・コーディング部の作成

RX830の本体処理から実行環境（ターゲット・システム）に依存する部分を切り出した処理，および，ユーザのソフトウエア環境を快適なものとするため用意された処理を作成します。

- ブート処理

  V830ファミリ™のリセット・エントリに割り付けられた関数です。

- ハードウエア初期化部

  実行環境（ターゲット・システム）上のハードウエアを初期化するために用意された関数です。

- ソフトウエア初期化部

  ユーザのソフトウエア環境を快適なものとするために用意された関数です。

- 割り込み／例外エントリ

  割り込み／例外が発生した際に起動される割り込み／例外処理管理への分岐処理専用ルーチンとして用意された関数です。

- クロック割り込みの後処理

  クロック割り込みが発生した際に起動される時間管理用割り込みハンドラ（クロック・ハンドラ）の後処理専用ルーチンとして用意された関数です。

- I/Oポート操作

  ポート・アドレスからのデータの読み込み／書き込みといった操作をC言語レベルで実現するために用意された関数です。

注1　標準添付のブート処理は，

```
nectools¥smp830¥rx830¥src¥boot.s     ··· CA830対応版
ghstools¥smp830¥rx830¥src¥boot.830 ··· CCV830対応版
```

に記述されています。

注2　標準添付のハードウエア初期化部，ソフトウエア初期化部，クロック割り込みの後処理は，

```
nectools¥smp830¥rx830¥src¥init.c ··· CA830対応版
ghstools¥smp830¥rx830¥src¥init.c ··· CCV830対応版
```

に記述されています。

注3　標準添付の割り込み／例外エントリは，

```
nectools¥smp830¥rx830¥src¥entry.s     ··· CA830対応版
ghstools¥smp830¥rx830¥src¥entry.830 ··· CCV830対応版
```

に記述されています。

注４　標準添付のI/Oポート操作は，

nectools¥smp830¥rx830¥src¥asmf.s　・・・ CA830対応版

ghstools¥smp830¥rx830¥src¥asmf.830 ・・・ CCV830対応版

に記述されています。

注５　ユーザ・オウン・コーディング部についての詳細は，「第4章　ユーザ・オウン・コーディング部」を参照してください。

3.4 処理プログラムの作成

システムで実現すべき処理（処理プログラム）を作成します。
なお，RX830では，処理プログラムを用途別に，以下のように分類しています。

- タスク

  RX830の管理下で実行可能な処理プログラムの最小単位です。

- 直接起動割り込みハンドラ

  マスカブル割り込みが発生した際，RX830を介在させることなく起動される割り込み処理専用ルーチンであり，タスクとは独立したものとして扱われます。このため，マスカブル割り込みが発生した際には，システム内で最高優先度を持つタスクが実行中であっても，その処理は中断され，直接起動割り込みハンドラに制御が移ります。

  なお，直接起動割り込みハンドラは，RX830を介在されることなく起動される割り込み処理専用ルーチンであるため，ハードウエアの限界に近い高速な応答性が期待される処理プログラムです。

- 間接起動割り込みハンドラ

  マスカブル割り込みが発生した際，RX830による割り込み前処理（レジスタの退避，スタックの切り替えなど）を行わせたのちに起動される割り込み処理専用ルーチンであり，タスクとは独立したものとして扱われます。このため，マスカブル割り込みが発生した際には，システム内で最高優先度を持つタスクが実行中であっても，その処理は中断され，間接起動割り込みハンドラに制御が移ります。

  なお，間接起動割り込みハンドラは，直接起動割り込みハンドラに比べて応答性の面では劣ったものとなりますが，RX830による割り込み前処理が行われているため，ハンドラ内での処理が簡素化された処理プログラムとなります。

- 周期起動ハンドラ

  一定の起動時間間隔に達した際，ただちに起動される周期処理専用ルーチンであり，タスクとは独立したものとして扱われます。このため，起動時間に達した際には，システム内で最高優先度を持つタスクが実行中であっても，その処理は中断され，周期起動ハンドラに制御が移ります。

  なお，周期起動ハンドラは，ユーザが記述する周期的な処理プログラムの中で，実行開始までのオーバヘッドが最も小さな処理プログラムです。

- 拡張SVCハンドラ

  ユーザが記述した関数を拡張システム・コールとしてRX830に登録した処理ルーチンです。

  なお，RX830では，拡張SVCハンドラを“拡張SVCハンドラを呼び出した実行プロシージャの延長線”として位置付けています。このため，タスク内から拡張SVCハンドラを呼

び出した場合には，“タスクから発行可能なシステム・コール”のみが発行可能となり，非タスク内から拡張SVCハンドラを呼び出した場合には，“非タスクから発行可能なシステム・コール”のみが発行可能となるため注意が必要です。

注1　標準添付の処理プログラムは，

nectools¥smp830¥rx830¥src¥task.c … CA830対応版

ghstools¥smp830¥rx830¥src¥task.c … CCV830対応版

に記述されています。

注2　処理プログラムの記述形式についての詳細は，「RX830 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

システムをROM化する際に必要となる初期化データ（.data, .sdata, .itextセクションなど）の退避領域を作成します。

注1 標準添付の初期化データの退避領域は，

nectools¥smp830¥rx830¥src¥rompcrt.s ··· CA830対応版

に記述されています。

注2 米国Green Hills Software, Inc.製CクロスV800™コンパイラ・パッケージCCV830を使用した場合，“初期化データの退避領域”を作成する必要はありません。

3.6 リンク・ディレクティブ・ファイルの作成

リンク・エディタが行うアドレス割り付けをユーザが固定化するためのファイル（リンク・ディレクティブ・ファイル）を作成します。

なお，RX830では，RX830のテキスト領域を以下に示した3つのセクションに分割して提供しています。これにより，大きなメモリ・サイズが要求されるメモリ領域（標準処理部）については外部RAMに，高速なメモリ・アクセスが要求されるメモリ領域（割り込み処理部，スケジューリング処理部）については内蔵命令RAMに割り付けるといったことが可能となります。

- 標準処理部（.systemセクション）

  RX830の本体処理（タスク管理機能，同期通信機能など）から構成されています。

  注意　標準処理部が必要とするメモリ・サイズは，内蔵命令RAMのサイズよりも大きくなります。このため，標準処理部を内蔵命令RAMに割り付けることはできません。

- 割り込み処理部（.system_intセクション）

  間接起動割り込みハンドラに制御を移す際に行われる割り込み前処理，および，間接起動割り込みハンドラから復帰する際に行われる割り込み後処理から構成されています。

  したがって，割り込み処理部を内蔵命令RAMに割り付けることにより，マスカブル割り込みの発生から間接起動割り込みハンドラの起動までの応答性が向上します。

- スケジューリング処理部（.system_cmnセクション）

  タスクの起床処理，および，タスクのスケジューリング処理から構成されています。

  したがって，スケジューリング処理部を内蔵命令RAMに割り付けることにより，タスクの起床処理，および，タスクのスケジューリング処理が高速化されるほかに，スケジューリング処理を伴ったシステム・コールの処理も高速化されます。

注1　標準添付のリンク・ディレクティブ・ファイルは，

```
nectools¥smp830¥rx830¥src¥sample.lnk  ･･･ CA830対応版
ghstools¥smp830¥rx830¥src¥sample.lnk  ･･･ CCV830対応版
```

に記述されています。

注2　リンク・ディレクティブ・ファイルの記述形式についての詳細は，使用するCコンパイラ・パッケージのユーザーズ・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

以下に，「3.2 CF 定義ファイルの作成」から「3.6 リンク・ディレクティブ・ファイルの作成」に示した各種ファイルからロード・モジュールを作成するまでの流れを示します。

(1) 情報ファイルの作成

コンフィギュレータを CF 定義ファイルに対して実行することにより，情報ファイル (システム情報テーブル，ブランチ・テーブル，システム情報ヘッダ・ファイル) を作成します。

(2) オブジェクト・ファイルの作成

C コンパイラ／アセンブラを下記ソース・ファイルに対して実行することにより，オブジェクト・ファイルを作成します。

- 情報ファイル
  - システム情報テーブル
  - ブランチ・テーブル
- システム初期化処理
  - ブート処理
  - ハードウエア初期化部
  - 割り込み／例外エントリ
  - クロック割り込みの後処理
  - I/O ポート操作
  - ソフトウエア初期化部
- 処理プログラム
  - 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
  - 拡張 SVC ハンドラ用インタフェース・ライブラリ
- 初期化データの退避領域

(3) ロード・モジュール (ROM 化情報を含まない) の作成

リンク・エディタを下記オブジェクト・ファイル，リンク・ディレクティブ・ファイル，ライブラリ・ファイルに対して実行することにより，ロード・モジュール (ROM 化情報を含まない) を作成します。

- 情報ファイル
  - システム情報テーブル
  - ブランチ・テーブル
- システム初期化処理
  - ブート処理
  - ハードウエア初期化部
  - 割り込み／例外エントリ
  - クロック割り込みの後処理
  - I/O ポート操作
  - ソフトウエア初期化部
- 処理プログラム
  - 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
  - 拡張 SVC ハンドラ用インタフェース・ライブラリ
- 初期化データの退避領域
- リンク・ディレクティブ・ファイル
- ライブラリ・ファイル
  - ニュークリアス共通部
  - ニュークリアス・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
  - ランタイム・ライブラリ

(4) ロード・モジュール (ROM 化情報を含む) の作成

ROM 化プロセッサをロード・モジュール (ROM 化情報を含まない) に対して実行することにより，ロード・モジュール (ROM 化情報を含む) を作成します。

(5) ロード・モジュール (HEX 形式) の作成

ヘキサ・コンバータをロード・モジュール (ROM 化情報を含む) に対して実行することにより，ロード・モジュール (HEX 形式) を作成します。

注意　米国 Green Hills Software, Inc. 製 C クロス V800™ コンパイラ・パッケージ CCV830 を使用した場合，“初期化データの退避領域” に関する操作を行う必要はありません。

第 4 章　ユーザ・オウン・コーディング部

この章では，ユーザ・オウン・コーディング部の機能，および，設計方法について説明しています。

4.1　概要

ユーザ・オウン・コーディング部は，ユーザの実行環境（ターゲット・システム）に依存する部分を RX830 内から切り出し，移植性の向上をはかるとともに，カスタマイズ化を容易にするために用意された関数群です。

なお，標準添付のユーザ・オウン・コーディング部は，ユーザの実行環境として，（株）電産製 V830 ファミリ™ ボード［DVE-V830/20］を想定した処理が記述してあります。このため，ユーザの実行環境が標準構成でない場合や，各種ハードウエアのモード設定を変更する場合は，ユーザ・オウン・コーディング部を書き替える必要があります。

表 4-1に，ユーザ・オウン・コーディング部の構成を示します。

表 4-1　ユーザ・オウン・コーディング部の構成

種　別	関数名	機　能
ブート処理	boot	システムのブート処理
ハードウエア初期化部	reset	ハードウエアの初期化
ソフトウエア初期化部	varfunc	ソフトウエア環境の整備
割り込み／例外エントリ	entry	割り込み／例外処理管理への分岐処理
クロック割り込みの後処理	clkhdr	時間管理用割り込みハンドラの後処理
I/O ポート操作	inpb	バイト・データの読み込み
	inph	ハーフワード・データの読み込み
	inpw	ワード・データの読み込み
	outpb	バイト・データの書き込み
	outph	ハーフワード・データの書き込み
	outpw	ワード・データの書き込み
ヘッダ・ファイル	—	ポート・アドレスなどの定義

4.2 ブート処理

ブート処理は，V830 ファミリ™ のリセット・エントリに割り付けられた関数であり，システム初期化処理の中でも，まず最初に処理が実行されるものです。

なお，標準添付のブート処理（関数名：boot）では，以下のような処理を行っています。

- gp，tp レジスタの設定
- 初期値なしメモリ領域の初期化
- ハードウエア初期化部の呼び出し
- ニュークリアス初期化部に制御を移す

注意　標準添付のブート処理は，

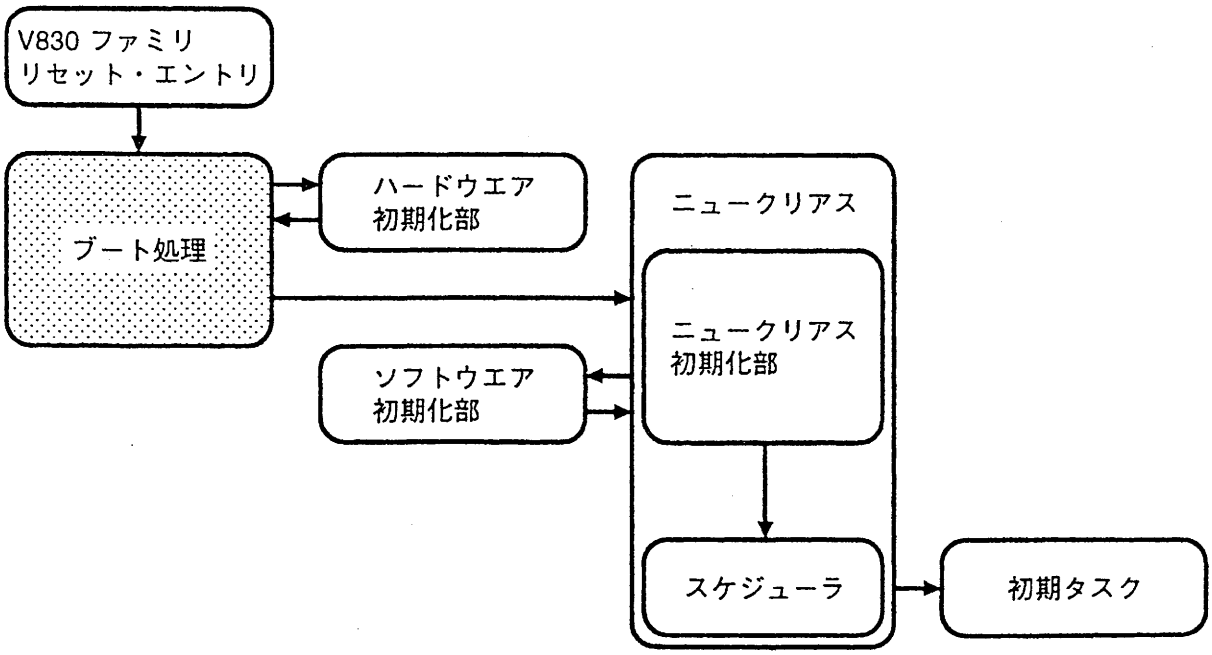
nectools¥smp830¥rx830¥src¥boot.s　・・・ CA830 対応版

ghstools¥smp830¥rx830¥src¥boot.830 ・・・ CCV830 対応版

に記述されています。

図 4-1に，ブート処理の位置付けを示します。

図 4-1　ブート処理の位置付け



4.3 ハードウエア初期化部

ハードウエア初期化部は，実行環境（ターゲット・システム）上のハードウエアを初期化するために用意された関数であり，ブート処理から呼び出されるものです。

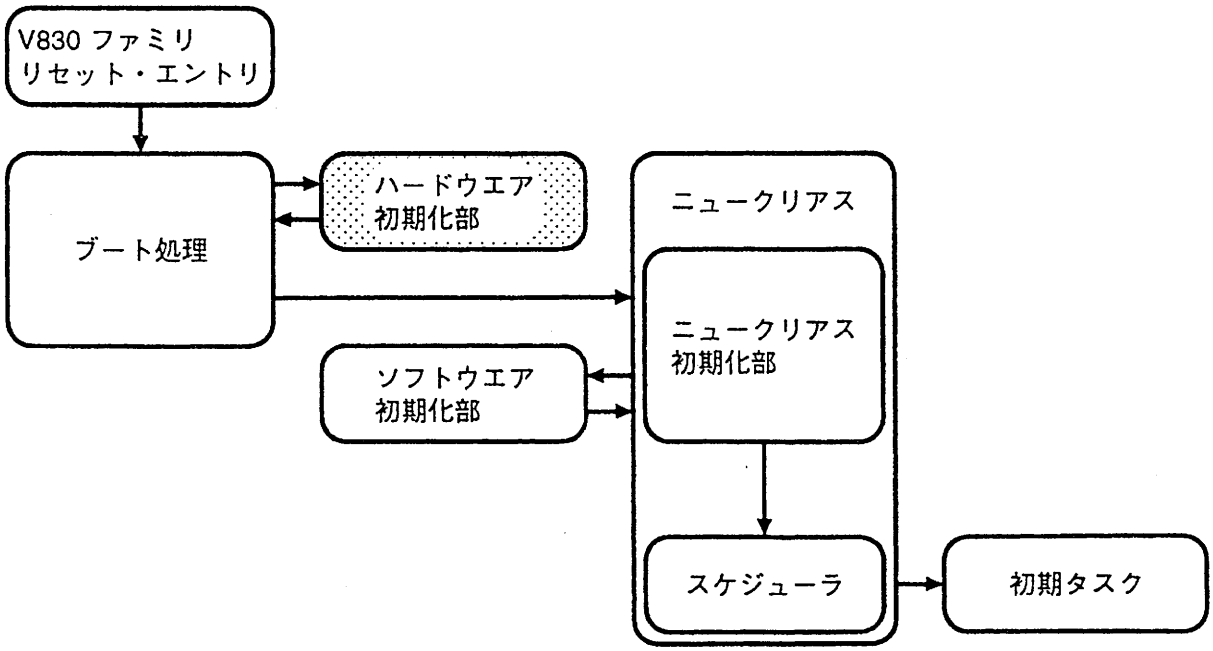
なお，標準添付のハードウエア初期化部（関数名：reset）では，以下のような処理を行っています。

- 内部ユニットの初期化
  - 割り込みコントローラの初期化
  - クロック・コントローラの初期化
- 周辺コントローラの初期化
- ブート処理に制御を戻す

注意　標準添付のハードウエア初期化部は，
nectools¥smp830¥rx830¥src¥init.c ･･･ CA830 対応版
ghstools¥smp830¥rx830¥src¥init.c ･･･ CCV830 対応版
に記述されています。

図 4-2に，ハードウエア初期化部の位置付けを示します。

図 4-2　ハードウエア初期化部の位置付け



4.4 ソフトウエア初期化部

ソフトウエア初期化部は，ユーザのソフトウエア環境を快適なものとするために用意された関数であり，ニュークリアス初期化部から呼び出されるものです。

なお，標準添付のソフトウエア初期化部（関数名： varfunc）では，以下のような処理を行っています。

- 初期化データのコピー
- クロック割り込みの許可
- ニュークリアス初期化部に制御を戻す

注意　標準添付のソフトウエア初期化部は，

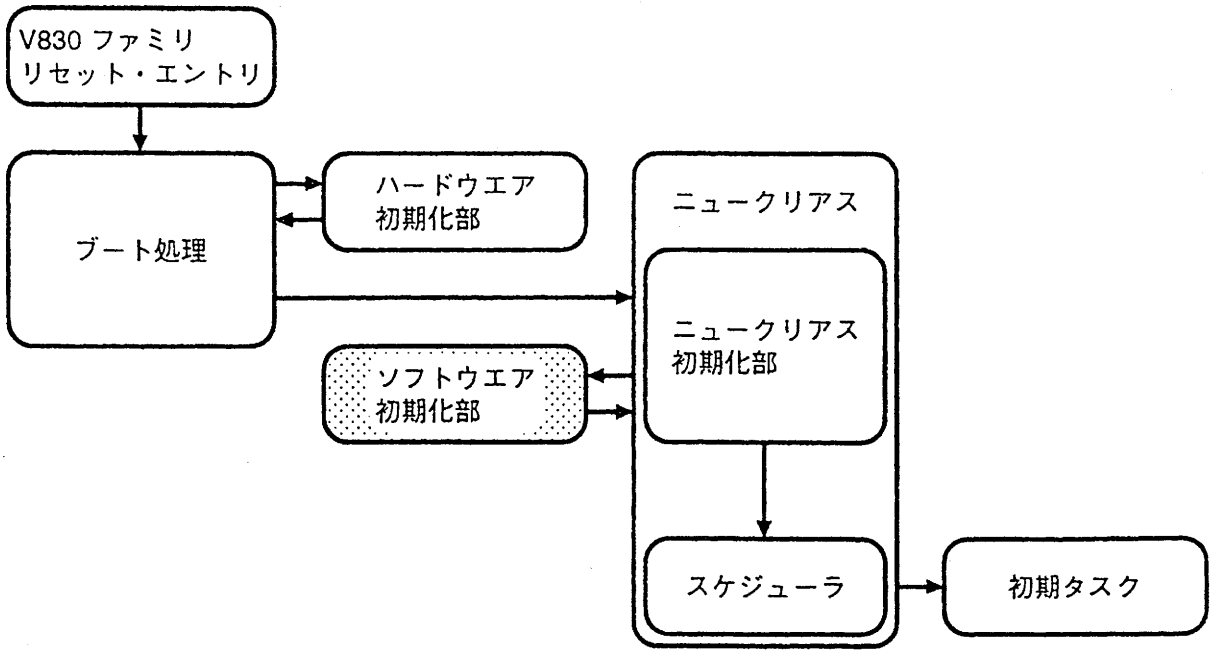
nectools¥smp830¥rx830¥src¥init.c ･･･ CA830 対応版

ghstools¥smp830¥rx830¥src¥init.c ･･･ CCV830 対応版

に記述されています。

図 4-3に，ソフトウエア初期化部の位置付けを示します。

図 4-3　ソフトウエア初期化部の位置付け



4.5 割り込み／例外エントリ

割り込み／例外エントリは，割り込み／例外が発生した際に起動される割り込み／例外処理管理への分岐命令専用ルーチンとして用意された関数であり，V830 ファミリ ™ が持つハンドラ・アドレスに割り付けられたものです。

なお，標準添付の割り込み／例外エントリ（関数名： entry）では，以下のような処理を行っています。

- 割り込み／例外処理管理（RX830 内）の入り口にジャンプ

注意　標準添付の割り込み／例外エントリは，

nectools￥smp830￥rx830￥src￥entry.s　… CA830 対応版

ghstools￥smp830￥rx830￥src￥entry.830 … CCV830 対応版

に記述されています。

4.6 クロック割り込みの後処理

クロック割り込みの後処理は，クロック割り込みが発生した際に起動される時間管理用割り込みハンドラ（クロック・ハンドラ）への分岐命令専用ルーチンとして用意された関数であり，V830 ファミリ ™ が持つハンドラ・アドレス（クロック割り込み用のエントリ）に割り付けられたものです。

なお，標準添付のクロック割り込みの後処理（関数名： clkhdr）では，以下のような処理を行っています。

- クロック割り込みのラッチ・クリア
- 割り込み終了コマンドの発行
- 時間管理用割り込みハンドラ（クロック・ハンドラ）に制御を移す

注意　標準添付のクロック割り込みの後処理は，

nectools￥smp830￥rx830￥src￥init.c … CA830 対応版

ghstools￥smp830￥rx830￥src￥init.c … CCV830 対応版

に記述されています。

4.7 I/O ポート操作

I/O ポート操作は，ポート・アドレスからのデータの読み込み／書き込みといった操作を C 言語レベルで実現するために用意された関数です。

なお，標準添付の I/O ポート操作（関数名：inpb, inph, inpw, outpb, outph, outpw）では，以下のような処理を行っています。

- inpb

  パラメータで指定されたポート・アドレスからバイト・データを読み込みます。

- inph

  パラメータで指定されたポート・アドレスからハーフワード・データを読み込みます。

- inpw

  パラメータで指定されたポート・アドレスからワード・データを読み込みます。

- outpb

  パラメータで指定されたポート・アドレスへバイト・データを書き込みます。

- outph

  パラメータで指定されたポート・アドレスへハーフワード・データを書き込みます。

- outpw

  パラメータで指定されたポート・アドレスへワード・データを書き込みます。

次頁以降に，各関数のインタフェース仕様を示します。

注意　標準添付の I/O ポート操作は，

nectools¥smp830¥rx830¥src¥asmf.s　　… CA830 対応版

ghstools¥smp830¥rx830¥src¥asmf.830 … CCV830 対応版

に記述されています。

保守／廃止

inpb

概　要

バイト・データを読み込む。

形　式

```
int     inpb(int prtadr);
```

パラメータ

I/O	パラメータ	説　明
I	int　*prtadr*;	データを読み込むポート・アドレス

説　明

prtadr で指定されたポート・アドレスからバイト・データを読み込みます。
なお，読み込んだバイト・データは，戻り値として返却されます。

戻り値

読み込んだバイト・データ

保守／廃止

inph

概　要

ハーフワード・データを読み込む。

形　式

```
int       inph(int prtadr);
```

パラメータ

I/O	パラメータ	説　明
I	int　*prtadr*;	データを読み込むポート・アドレス

説　明

prtadr で指定されたポート・アドレスからハーフワード・データを読み込みます。
なお，読み込んだハーフワード・データは，戻り値として返却されます。

戻り値

読み込んだハーフワード・データ

保守／廃止

inpw

概　要

ワード・データを読み込む。

形　式

```
int       inpw(int prtadr);
```

パ ラ メ ー タ

I/O	パラメータ	説　明
I	int　*prtadr*;	データを読み込むポート・アドレス

説　明

prtadr で指定されたポート・アドレスからワード・データを読み込みます。
なお，読み込んだワード・データは，戻り値として返却されます。

戻　り　値

読み込んだワード・データ

保守／廃止

outpb

概　要

バイト・データを書き込む。

形　式

```
void      outpb(int prtadr, int data);
```

パラメータ

I/O	パラメータ	説　明
I	int　*prtadr*;	データを書き込むポート・アドレス
I	int　*data*;	書き込むデータ

説　明

prtadr で指定されたポート・アドレスへ *data* で指定されたバイト・データを書き込みます。

戻り値

なし

保守／廃止

outph

概　要

ハーフワード・データを書き込む。

形　式

```
void      outph(int prtadr, int data);
```

パラメータ

I/O	パラメータ	説　明
I	int　*prtadr*;	データを書き込むポート・アドレス
I	int　*data*;	書き込むデータ

説　明

prtadr で指定されたポート・アドレスへ *data* で指定されたハーフワード・データを書き込みます。

戻り値

なし

保守／廃止

outpw

概　要

ワード・データを書き込む。

形　式

```
void      outpw(int prtadr, int data);
```

パラメータ

I/O	パラメータ	説　明
I	int　*prtadr*;	データを書き込むポート・アドレス
I	int　*data*;	書き込むデータ

説　明

prtadr で指定されたポート・アドレスへ *data* で指定されたワード・データを書き込みます。

戻り値

なし

4.8　ヘッダ・ファイル

RX830では，ユーザ・オウン・コーディング部の関数が使用するポート・アドレスなどといった定数を，専用のヘッダ・ファイルにマクロ定義しています。

なお，標準添付のヘッダ・ファイルでは，以下のようなマクロ定義を行っています。

- VMEバス・インタフェース・ゲート・アレイのポート・アドレス
- ボード・コントロール・レジスタのポート・アドレス
- ティック・タイマ制御コマンド
- 割り込み制御コマンド
- 割り込みアクノリッジ・サイクル・レジスタのポート・アドレス

注意　標準添付のヘッダ・ファイルは，

nectools¥smp830¥rx830¥src¥user.h … CA830対応版

ghstools¥smp830¥rx830¥src¥user.h … CCV830対応版

に記述されています。

第 5 章　インタフェース・ライブラリ

5.1　概要

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，および，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，外部関数形式で発行されたシステム・コール，または，外部関数形式で呼び出された拡張SVCハンドラをニュークリアス発行形式に変換し，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・ライブラリが必要となります。

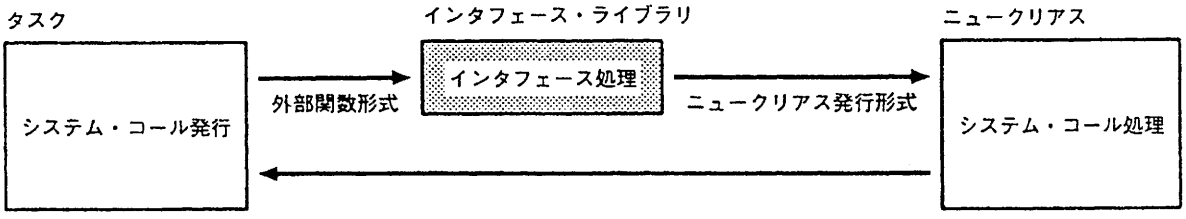
つまり，インタフェース・ライブラリは，処理プログラムとニュークリアスの中間に位置し，ニュークリアスが対応した処理を行ううえで必要となる各種情報を設定したのち，ニュークリアスに制御を移す機能を持ちます。

なお，RX830が提供するインタフェース・ライブラリは，NEC製V830ファミリ™用Cコンパイラ CA830用，および，米国 Green Hills Software, Inc. 製Cクロス V800™ コンパイラ CCV830用の2種類が用意されています。

また，RX830が提供するシステム・コール用インタフェース・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）と，システム・コールのパラメータをチェックしないインタフェース・ライブラリ（パラメータ・チェック機能なし）の2種類があり，用途に応じて使い分けることができます。

図5-1に，インタフェース・ライブラリの位置付けを示します。

図 5-1　インタフェース・ライブラリの位置付け



5.2 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，以下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，以下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
- システム・コール例外処理の戻り番地にジャンプ

図5-2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 5-2　システム・コール用インタフェース・ライブラリ

```
        .text
        .globl  _syscall_name
        .align  2
_syscall_name
        -- 機能コードを設定
        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]


-- エラーを検出した際の処理
_syscall_err
        -- エラー・コードを設定
        movea   lo(err_code), r0, r10
        -- システム・コール例外処理の戻り番地にジャンプ
        jmp     [lp]
```

5.3 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，以下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(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，以下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
- 拡張SVCハンドラ例外処理の戻り番地にジャンプ

図5-3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 5-3　拡張 SVC ハンドラ用インタフェース・ライブラリ

```
        .text
        .globl  _svchdr_name
        .align  2
_svchdr_name
        -- 機能コードを設定
        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]


-- エラーを検出した際の処理
_svchdr_err
        -- エラー・コードを設定
        movea   lo(err_code), r0, r10
        -- 拡張 SVC ハンドラ例外処理の入り口にジャンプ
        jmp     [lp]
```

第 6 章　メモリ容量の見積り

この章では，RX830 の管理領域 (システム・メモリ領域，ユーザ・メモリ領域) として使用されるメモリ容量を見積る際に必要となる情報について解説しています。

6.1　概要

RX830 では，管理領域を用途別に 2 種類に分類し，管理しています。
以下に，管理領域 (システム・メモリ領域，ユーザ・メモリ領域) について示します。

- システム・メモリ領域

  システム・メモリ領域は，System Memory Pool 0，System Memory Pool 1 から構成されており，管理オブジェクト，タスク用スタック領域，割り込みハンドラ用スタック領域が割り当てられます。

  注意　RX830 では，管理オブジェクトの割り付けを System Memory Pool 0 に限定しています。

- ユーザ・メモリ領域

  ユーザ・メモリ領域は，User Memory Pool 0，User Memory Pool 1 から構成されており，メモリ・プールが割り当てられます。

6.2　メモリ容量計算式

6.2.1　管理オブジェクト

以下に，管理オブジェクトのメモリ容量計算式 (単位： byte) を示します。
なお，管理オブジェクトとして割り当てられる領域は，下記計算式から算出された値を合計したものとなります。

- オペレーティング・システム管理テーブル

  OBT_SIZ = align8(744 + align32(*pri_lvl* + 4) / 8) + align8(*pri_lvl* x 2 + 8)

  注意　*pri_lvl* は，コンフィギュレーション時，システム最大値情報のタスクの優先度範囲に定義された値となります。

- タスク管理ブロック

  TSK_SIZ = *tsk_cnt* x 56

  注意　*tsk_cnt* は，コンフィギュレーション時，システム最大値情報のタスクの最大生成数に定義された値となります。

- セマフォ管理ブロック

  SEM_SIZ = *sem_cnt* x 20

  注意　*sem_cnt* は，コンフィギュレーション時，システム最大値情報のセマフォの最大生成数に定義された値となります。

- イベント・フラグ管理ブロック

  FLG_SIZ = *flg_cnt* x 20

  注意　*flg_cnt* は，コンフィギュレーション時，システム最大値情報のイベント・フラグの最大生成数に定義された値となります。

- メールボックス管理ブロック

  MBX_SIZ = *mbx_cnt* x 20

  注意　*mbx_cnt* は，コンフィギュレーション時，システム最大値情報のメールボックスの最大生成数に定義された値となります。

- メモリ・プール管理ブロック

  MPL_SIZ = *mpl_cnt* x 24

  注意　*mpl_cnt* は，コンフィギュレーション時，システム最大値情報のメモリ・プールの最大生成数に定義された値となります。

- 周期起床用時間管理ブロック

  CYC_SIZ = *cyc_cnt* x 40

  注意　*cyc_cnt* は，コンフィギュレーション時，システム最大値情報の周期起動ハンドラの最大登録数に定義された値となります。

- 拡張SVCハンドラ管理ブロック

  SVC_SIZ = *svc_cnt* x 16

  注意 *svc_cnt*は，コンフィギュレーション時，システム最大値情報の拡張SVCハンドラの最大登録数に定義された値となります。

6.2.2 タスク用スタック領域

以下に，1タスク当たりのタスク用スタック領域のメモリ容量計算式(単位： byte)を示します。

なお，タスク用スタック領域として割り当てられる領域は，下記計算式から算出された1タスク当たりの値を合計したものとなります。

STK_SIZ = align8(*stk_siz* + 88)

注意 *stk_siz*は，コンフィギュレーション時，タスク情報のタスク用スタック情報に定義された値，または，**cre_tsk**システム・コール発行時，タスク生成情報のスタック・サイズに定義された値となります。また割り込み用のスタック・サイズ(76Byte)を含みます。

6.2.3 割り込みハンドラ用スタック領域

以下に，割り込みハンドラ用スタック領域のメモリ容量計算式(単位： byte)を示します。

INTSTK_SIZ = *intstk_siz*

注意 *intstk_siz*は，コンフィギュレーション時，システム情報の割り込みハンドラ用スタック情報に定義された値となります。

6.2.4 メモリ・プール

以下に，1メモリ・プール当たりのメモリ・プールのメモリ容量計算式(単位： byte)を示します。

なお，メモリ・プールとして割り当てられる領域は，下記計算式から算出された1メモリ・プール当たりの値を合計したものとなります。

MPL_SIZ = *mpl_siz*

注意 *mpl_siz*は，コンフィギュレーション時，メモリ・プール情報のメモリ・プール情報に定義された値，または，**cre_mpl**システム・コール発行時，メモリ・プール生成情報のメモリ・プール・サイズに定義された値となります。

6.3 メモリ容量の見積り例

以下に，RX830 の管理領域 (システム・メモリ領域，ユーザ・メモリ領域) として使用されるメモリ容量の見積り方法を示します。

[前提条件]

- システム情報

  割り込みハンドラ用スタック情報 ： System Memory Pool 0 から 0x100 byte のメモリ領域を確保

- 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

- タスク情報

  タスク用スタック情報 ： System Memory Pool 0 から 0x100 byte のメモリ領域を確保

  タスク用スタック情報 ： System Memory Pool 0 から 0x100 byte のメモリ領域を確保

- メモリ・プール情報

  メモリ・プール情報 ： User Memory Pool 0 から 0x1000 byte のメモリ領域を確保

  メモリ・プール情報 ： User Memory Pool 0 から 0x2000 byte のメモリ領域を確保

注意　ユーザの処理プログラム内では，cre_tsk，cre_mpl システム・コールの発行が行われないものとします。

[見積り方法]

- 管理オブジェクト

```
OBJ_SIZ = {align8(744 + align32(0xf + 4) / 8) + align8(0xf x 2 + 8)}
          + {0x2 x 56} + {0x1 x 20} + {0x2 x 20} + {0x3 x 20}
          + {0x2 x 24} + {0x1 x 40} + {0x1 x 20}
        = 1288
```

- タスク用スタック領域

  STK_SIZ = align8(0x100 + 88) + align8(0x100 + 88)
  = 688

- 割り込みハンドラ用スタック領域

  INTSTK_SIZ = 256

- メモリ・プール

  MPL_SIZ = 0x1000 + 0x2000
  = 12288

上記計算結果から，

システム・メモリ領域

System Memory Pool 0 ··· 2072 byte
System Memory Pool 1 ··· 0 byte

ユーザ・メモリ領域

User Memory Pool 0 ··· 12288 byte
User Memory Pool 1 ··· 0 byte

の管理領域が必要であることがわかります。

第 7 章　コンフィギュレータ CF830

この章では，コンフィギュレータ CF830 の役割，動作環境について解説しています。

7.1　概要

RX830 を使ったシステムを構築する場合，RX830 に提供する各種データを保持した情報ファイル（システム情報テーブル，ブランチ・テーブル，システム情報ヘッダ・ファイル）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，これら情報ファイルは，記述性／可読性の面で劣ったものであるため，記述に際しては，かなりの労力を必要とします。

そこで，RX830 では，記述性／可読性に優れた独自の記述形式で作成されたファイル（CF 定義ファイル）を情報ファイルへと変換するユーティリティ・ツールを提供しています。

このユーティリティ・ツールがコンフィギュレータであり，コンフィギュレータは，独自の記述形式で作成された CF 定義ファイルを入力ファイルとして読み込んだのち，システム情報テーブル，ブランチ・テーブル，システム情報ヘッダ・ファイルといった情報ファイルを出力します。

以下に，コンフィギュレータ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

  RX830 が動作するうえで必要となるデータ（基本クロック周期，タスクの優先度範囲など）を保持した情報ファイルです。

- ブランチ・テーブル

  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を保持した情報ファイルです。

  なお，RX830 では，システムで使用する機能に制限を設けることにより，必要となるメモリ容量を小さくすることが可能です。

- システム情報ヘッダ・ファイル

  CF 定義ファイルに記述された各種オブジェクト名（タスク名，セマフォ名など）と ID 番号の対応が定義されている情報ファイルです。

7.2 動作環境

表 7-1, コンフィギュレータが動作するうえで必要となる環境を示します。

表 7-1 コンフィギュレータの動作環境

開発環境（ホスト・マシン）	オペレーティング・システム
PC-9800 シリーズ	Windows 95 (MS-DOS プロンプト)
IBM-PC/AT 互換機	Windows 95 (MS-DOS プロンプト)
SPARC station™	SunOS™ Rel 4.1.x
SPARC station™	Solaris™ Rel 2.5.x

第 8 章　CF 定義ファイルの記述方法

この章では，コンフィギュレータ CF830 を使用して情報ファイル（システム情報テーブル，ブランチ・テーブル，システム情報ヘッダ・ファイル）を作成する際に必要となる CF 定義ファイルの記述方法について解説しています。

8.1　表記方法

以下に，CF 定義ファイルを記述する際の表記方法を示します。

(1) 文字コード

CF 定義ファイルは，ASCII コードで記述します。

なお，英小文字と英大文字の区別は行われます。

また，単語（数値，シンボル名，キー・ワードなど）の区切りはスペース，または，タブで行い，文の区切りは改行で行います。

注意　日本語は，EUC コード，または，シフト JIS コードのものをコメント内でのみ記述することができます。

(2) 数値

数値は，指定のない限り 32 ビット幅（0x0 ～ 0xffffffff）の範囲で記述することができます。

(3) シンボル名

シンボル名は，256 文字までの英数字で記述することができます。

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

CF 定義ファイルでは，“--”以降行末までがコメントとして扱われます。

(5) 行の併合

CF 定義ファイルでは，行末に“\”を記述した場合，次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。

(6) キー・ワード

コンフィギュレータでは，以下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，これらの文字列を他の用途に使用することは禁止されています。

clktim	cyc	defstk	flg	flgsvc
ini	inthdr	intstk	intsvc	maxcyc
maxflg	maxmbx	maxmpl	maxpri	maxsem
maxsvc	maxtsk	mbx	mbxsvc	mem
mpl	mplsvc	no_use	prtflg	prtmbx
prtmpl	prtsem	prttsk	RX830	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	V300	

8.2 コンフィギュレーション情報

CF 定義ファイルに記述するコンフィギュレーション情報は，以下の 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
  RX830 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

8.2.1 リアルタイム OS 情報

CF 定義ファイルに記述するリアルタイム OS 情報は，以下に示す 1 項目です。

(1) RX シリーズ情報

RX シリーズ情報として，以下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

8.2.2 SIT 情報

CF 定義ファイルに記述する SIT 情報は，以下に示す 12 項目です。

(1) システム情報

システム情報として，以下のデータを定義します。

- 基本クロック周期
- ディフォールト・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
- 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，以下のデータを定義します。

- タスクの優先度範囲
- 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

(3) 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，以下のデータを定義します。

- システム・メモリの種類
- システム・メモリの先頭アドレス
- システム・メモリのサイズ

(4) タスク情報

個々のタスクに対して，以下のデータを定義します。

- タスクの ID
- タスクの初期状態
- タスクの起動コード
- タスクの拡張情報
- タスクの記述言語
- タスクの起動アドレス
- タスクの起動時優先度（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
- タスクの固有 GP レジスタ値
- タスクの固有 TP レジスタ値
- タスクのキー ID 番号

(5) セマフォ情報

個々のセマフォに対して，以下のデータを定義します。

- セマフォの ID
- セマフォの拡張情報
- タスクのキューイング方式
- セマフォの初期資源数
- セマフォの最大資源数
- セマフォのキー 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，以下のデータを定義します。

- イベント・フラグの ID
- イベント・フラグの拡張情報
- 複数タスクの待ち許可／禁止
- イベント・フラグの初期ビット・パターン
- イベント・フラグのキー 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，以下のデータを定義します。

- メールボックスの ID
- メールボックスの拡張情報
- 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
- メールボックスのキー 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，以下のデータを定義します。

- 割り込みレベル
- 割り込みハンドラの記述言語
- 割り込みハンドラの起動アドレス
- 割り込みハンドラの固有 GP レジスタ値
- 割り込みハンドラの固有 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，以下のデータを定義します。

- メモリ・プールの ID
- メモリ・プールの拡張情報
- 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
- メモリ・プールのキー 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，以下のデータを定義します。

- 周期起動ハンドラの指定番号
- 周期起動ハンドラの拡張情報
- 周期起動ハンドラの記述言語
- 周期起動ハンドラの起動アドレス
- 周期起動ハンドラの初期活性状態
- 周期起動ハンドラの起動時間間隔

- 周期起動ハンドラの固有 GP レジスタ値
- 周期起動ハンドラの固有 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，以下のデータを定義します。

- 拡張 SVC ハンドラの機能番号
- 拡張 SVC ハンドラの記述言語
- 拡張 SVC ハンドラの起動アドレス
- 拡張 SVC ハンドラの固有 GP レジスタ値
- 拡張 SVC ハンドラの固有 TP レジスタ値

(12) 初期化ハンドラ情報

初期化ハンドラ情報として，以下のデータを定義します。

- 初期化ハンドラの記述言語
- 初期化ハンドラの起動アドレス
- 初期化ハンドラの固有 GP レジスタ値
- 初期化ハンドラの固有 TP レジスタ値

8.2.3 SCT 情報

CF 定義ファイルに記述する SCT 情報は，以下に示す 8 項目です。

(1) タスク管理／タスク付属同期管理システム・コール情報

タスク管理／タスク付属同期管理システム・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システム・コールを定義します。

以下に，RX830 が提供しているタスク管理／タスク付属同期管理システム・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）管理システム・コール情報

セマフォ管理システム・コール情報として，ユーザ処理プログラム内で使用するセマフォ管理システム・コールを定義します。

以下に，RX830 が提供しているセマフォ管理システム・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）管理システム・コール情報

イベント・フラグ管理システム・コール情報として，ユーザ処理プログラム内で使用するイベント・フラグ管理システム・コールを定義します。

以下に，RX830 が提供しているイベント・フラグ管理システム・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）管理システム・コール情報

メールボックス管理システム・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ールボックス管理システム・コールを定義します。

以下に，RX830 が提供しているメールボックス管理システム・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) 割り込み処理管理システム・コール情報

割り込み処理管理システム・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。

以下に，RX830 が提供している割り込み処理管理システム・コールを示します。

```
def_int     ret_int     ret_wup     vret_clk     loc_cpu
unl_cpu     chg_ilv     ref_ilv
```

(6) メモリ・プール管理システム・コール情報

メモリ・プール管理システム・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・プール管理システム・コールを定義します。

以下に，RX830 が提供しているメモリ・プール管理システム・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) 時間管理システム・コール情報

時間管理システム・コール情報として，ユーザ処理プログラム内で使用する時間管理システム・コールを定義します。

以下に，RX830 が提供している時間管理システム・コール情報を示します。

```
set_tim     get_tim     dly_tsk     def_cyc     act_cyc
ref_cyc
```

(8) システム管理システム・コール情報

システム管理システム・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システム・コールを定義します。

以下に，RX830 が提供しているシステム管理システム・コールを示します。

```
get_ver     ref_sys     def_svc     viss_svc
```

8.3 リアルタイムOS情報の記述形式

CF定義ファイルに記述するリアルタイムOS情報の記述形式を以下に示します。

なお，表記中のゴシック書体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。

8.3.1 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，CF定義ファイルに，RXシリーズ情報を記述することは必須です。

図8-1に，RXシリーズ情報の記述形式を示します。

図8-1 RXシリーズ情報の記述形式

```
rxsers  rtos_nam  rtos_ver
```

RXシリーズ情報で記述する各項目について，以下に示します。

rtos_nam　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，RX830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，V300に限られます。

8.4 SIT 情報の記述形式

CF 定義ファイルに記述する SIT 情報の記述形式を以下に示します。
なお，表記中のゴシック書体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。

8.4.1 システム情報

システム情報では，「基本クロック周期，ディフォールト・スタック・サイズ，スタック情報，ID 番号の保護範囲」を定義します。
なお，CF 定義ファイルに，システム情報を記述することは必須です。
図 8-2に，システム情報の記述形式を示します。

図 8-2 システム情報の記述形式

```
clktim  time
defstk  stk_siz
intstk  intstk_siz: mem_nam
prttsk  tsk_idlmt
prtsem  sem_idlmt
prtflg  flg_idlmt
prtmbx  mbx_idlmt
prtmpl  mpl_idlmt
```

システム情報で記述する各項目について，以下に示します。

time
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 ～ 0xffff に限られます。

備考 基本クロック周期とは，RX830 が時間管理機能 (周期起床，遅延起床，タイムアウトなど) を実現するうえで必要になるクロック割り込みの発生周期を意味します。

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

備考 ディフォールト・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，ディフォールト・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，ディフォールト・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：byte），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 ～ 0x7fffffac の 4 byte 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 ： 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* として指定可能な値は，0x0 ～ *tsk_cnt* に限られます。

注意　0x0 を指定した場合，ID 番号の保護を行いません。
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* として指定可能な値は，0x0 ～ *sem_cnt* に限られます。

注意　0x0 を指定した場合，ID 番号の保護を行いません。
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* として指定可能な値は，0x0 ～ *flg_cnt* に限られます。

注意　0x0 を指定した場合，ID 番号の保護を行いません。
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* として指定可能な値は，0x0 ～ *mbx_cnt* に限られます。

注意　0x0 を指定した場合，ID 番号の保護を行いません。
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* として指定可能な値は，0x0 ～ *mpl_cnt* に限られます。

注意　0x0 を指定した場合，ID 番号の保護を行いません。
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。

8.4.2 システム最大値情報

システム最大値情報では，システム内で使用する「タスクの優先度範囲（優先度数），オブジェクト（タスクなど）の最大生成数，オブジェクト（周期起動ハンドラなど）の最大登録数」を定義します。

なお，CF 定義ファイルに，システム最大値情報を記述することは必須です。

図 8-3に，システム最大値情報の記述形式を示します。

図 8-3　システム最大値情報の記述形式

```
maxpri  pri_lvl
maxtsk  tsk_cnt
maxsem  sem_cnt
maxflg  flg_cnt
maxmbx  mbx_cnt
maxmpl  mpl_cnt
maxcyc  cyc_cnt
maxsvc  svc_cnt
```

システム最大値情報で記述する各項目について，以下に示します。

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。

保守／廃止

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。

8.4.3 メモリ情報

メモリ情報では，「種類，先頭アドレス，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，CF 定義ファイルに，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，CF 定義ファイルに，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が必須となります。

注意　個々のシステム・メモリは，複数のメモリ領域に分割して割り付けることができます。

図 8-4に，メモリ情報の記述形式を示します。

図 8-4　メモリ情報の記述形式

```
mem   mem_id   mem_adr   mem_siz
```

メモリ情報で記述する各項目について，以下に示します。

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。

mem_adr　システム・メモリの先頭アドレスを指定します。
なお，*mem_adr* として指定可能な値は，0x0 ～ 0xfffffffc の 4byte 境界値に限られます。

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

注意　システム・メモリの見積り方法についての詳細は，**「第 6章　メモリ容量の見積り」**を参照してください。

8.4.4 タスク情報

タスク情報では，「ID，初期状態，起動コード，拡張情報，記述言語，起動アドレス，起動時優先度，割り込み状態，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，CF 定義ファイルに，タスク情報を 1 個以上記述することは必須です。

また，タスク情報として定義可能な数は，1 ～システム最大値情報で登録したタスクの最大生成数 *tsk_cnt* の範囲に限られます。

図 8-5に，タスク情報の記述形式を示します。

図 8-5　タスク情報の記述形式

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

タスク情報で記述する各項目について，以下に示します。

tsk_id

タスクの ID 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。

注意　0x0，または，シンボル名を指定した場合，コンフィギュレータが *tsk_idlmt* ～ *tsk_cnt* の未使用 ID 番号を自動的に割り付けます。
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。
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。

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，ユーザが自由に使用することができます。
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，ダイナミックに獲得することができます。

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のいずれかに限られます。

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2byte境界値，シンボル名に限られます。

pri

タスクの起動時優先度（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。

注意　*pri_lvl*は，システム最大値情報の「タスクの優先度範囲max pri」に定義された値となります。

intr

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のいずれかに限られます。

TA_ENAINT ： タスク起動時，すべての割り込みを許可します。
TA_DISINT ： タスク起動時，すべての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： byte），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4byte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 ： System Memory Pool 0にタスク用スタックを割り付けます。

SPOL1 ： System Memory Pool 1 にタスク用スタックを割り付けます。

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 ： システム起動時，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 ： システム起動時，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　0x0 を指定した場合，コンフィギュレータはキー ID 番号の割り付けを行いません。

8.4.5 セマフォ情報

セマフォ情報では，「ID，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。

図 8-6に，セマフォ情報の記述形式を示します。

図 8-6 セマフォ情報の記述形式

```
sem  sem_id  ext_inf  twai_opt  init_cnt  max_cnt  key_id
```

セマフォ情報で記述する各項目について，以下に示します。

sem_id セマフォの ID 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。

注意 0x0，または，シンボル名を指定した場合，コンフィギュレータが *sem_idlmt* ～ *sem_cnt* の未使用 ID 番号を自動的に割り付けます。
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。
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，ユーザが自由に使用することができます。
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，ダイナミックに獲得することができます。

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　0x0 を指定した場合，コンフィギュレータはキー ID 番号の割り付けを行いません。

8.4.6 イベント・フラグ情報

イベント・フラグ情報では，「ID，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」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* の範囲に限られます。

図8-7に，イベント・フラグ情報の記述形式を示します。

図 8-7 イベント・フラグ情報の記述形式

```
flg  flg_id  ext_inf  twai_opt  init_ptn  key_id
```

イベント・フラグ情報で記述する各項目について，以下に示します。

flg_id イベント・フラグのIDを指定します。
なお，*flg_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。

注意 0x0，または，シンボル名を指定した場合，コンフィギュレータが *flg_idlmt* ～ *flg_cnt* の未使用ID番号を自動的に割り付けます。
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。
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，ユーザが自由に使用することができます。
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，ダイナミックに獲得することができます。

twai_opt 複数タスクの待ち許可／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 ： 複数タスクの待ちは，禁止になります。

TA_WMUL ： 複数タスクの待ちは，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　0x0 を指定した場合，コンフィギュレータはキー ID 番号の割り付けを行いません。

8.4.7　メールボックス情報

メールボックス情報では，「ID，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* の範囲に限られます。

図 8-8に，メールボックス情報の記述形式を示します。

図 8-8　メールボックス情報の記述形式

```
mbx   mbx_id   ext_inf   twai_opt   mwai_opt   key_id
```

メールボックス情報で記述する各項目について，以下に示します。

mbx_id　　メールボックスの ID 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。

注意　0x0，または，シンボル名を指定した場合，コンフィギュレータが *mbx_idlmt* ～ *mbx_cnt* の未使用 ID 番号を自動的に割り付けます。
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。
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，ユーザが自由に使用することができます。
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，ダイナミックに獲得することができます。

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　0x0 を指定した場合，コンフィギュレータはキー ID 番号の割り付けを行いません。

8.4.8 割り込みハンドラ情報

割り込みハンドラ情報では，「割り込みレベル，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，各割り込みレベルに対して 1 個に限られます。

図 8-9に，割り込みハンドラ情報の記述形式を示します。

図 8-9 割り込みハンドラ情報の記述形式

```
inthdr  int_lvl  lang  hdr_adr  data  text
```

割り込みハンドラ情報で記述する各項目について，以下に示します。

int_lvl 割り込みレベルを指定します。
なお，*int_lvl* として指定可能な値は，0x0 ～ 0xf に限られます。

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 のいずれかに限られます。

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 2byte 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 ： システム起動時，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 ： システム起動時，固有 TP レジスタ値の設定は行われません。

8.4.9 メモリ・プール情報

メモリ・プール情報では，「ID，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* の範囲に限られます。

図8-10に，メモリ・プール情報の記述形式を示します。

図 8-10 メモリ・プール情報の記述形式

```
mpl  mpl_id  ext_inf  twai_opt  mpl_siz:mem_nam  key_id
```

メモリ・プール情報で記述する各項目について，以下に示します。

mpl_id

メモリ・プールのID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。

注意 0x0，または，シンボル名を指定した場合，コンフィギュレータが *mpl_idlmt* ～ *mpl_cnt* の未使用ID番号を自動的に割り付けます。
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。
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，ユーザが自由に使用することができます。
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，ダイナミックに獲得することができます。

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。

mpl_siz:*mem_nam*

メモリ・プールのサイズ (単位： byte)，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，0x100 ～ 0x7ffffffc の 4 byte 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 ： User Memory Pool 0 にメモリ・プールを割り付けます。
UPOL1 ： User Memory Pool 1 にメモリ・プールを割り付けます。

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 0x0 を指定した場合，コンフィギュレータはキー ID 番号の割り付けを行いません。

8.4.10 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。

図 8-11に，周期起動ハンドラ情報の記述形式を示します。

図 8-11 周期起動ハンドラ情報の記述形式

```
cyc  cyc_no  ext_inf  lang  hdr_adr  act  intvl  data  text
```

周期起動ハンドラ情報で記述する各項目について，以下に示します。

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。

注意 シンボル名を指定した場合，コンフィギュレータが 0x1 ～ *cyc_cnt* の未使用指定番号を自動的に割り付けます。
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，ユーザが自由に使用することができます。
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，ダイナミックに獲得することができます。

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 のいずれかに限られます。

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，0x0 ～ 0xfffffffe の 2byte 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 のいずれかに限られます。

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 ～ 0x7fffffff 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 ：システム起動時，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 ：システム起動時，固有 TP レジスタ値の設定は行われません。

8.4.11 拡張SVCハンドラ情報

拡張SVCハンドラ情報では，「機能番号，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 *svc_cnt* の範囲に限られます。

図8-12に，拡張SVCハンドラ情報の記述形式を示します。

図 8-12 拡張SVCハンドラ情報の記述形式

```
svc   svc_no   lang   hdr_adr   data   text
```

拡張SVCハンドラ情報で記述する各項目について，以下に示します。

svc_no　拡張SVCハンドラの機能番号を指定します。
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。

注意　シンボル名を指定した場合，コンフィギュレータが0x1 ～ *svc_cnt* の未使用機能番号を自動的に割り付けます。
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のいずれかに限られます。

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2byte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 ： システム起動時，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 ： システム起動時，固有 TP レジスタ値の設定は行われません。

8.4.12 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，CF 定義ファイルに，初期化ハンドラ情報を記述することは必須です。

図 8-13に，初期化ハンドラ情報の記述形式を示します。

図 8-13　初期化ハンドラ情報の記述形式

```
ini   lang   hdr_adr   data   text
```

初期化ハンドラ情報で記述する各項目について，以下に示します。

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 のいずれかに限られます。

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 2byte 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 ： システム起動時，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 ： システム起動時，固有 TP レジスタ値の設定は行われません。

8.5 SCT 情報の記述形式

CF 定義ファイルに記述する SCT 情報の記述形式を以下に示します。

なお，表記中のゴシック書体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。

8.5.1 タスク管理／タスク付属同期管理システム・コール情報

タスク管理／タスク付属同期管理システム・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。

図 8-14に，タスク管理／タスク付属同期管理システム・コール情報の記述形式を示します。

図 8-14 タスク管理／タスク付属同期管理システム・コール情報の記述形式

```
tsksvc  svc_nam
```

タスク管理／タスク付属同期管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　タスク管理／タスク付属同期管理システム・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，以下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			

8.5.2 同期通信（セマフォ）管理システム・コール情報

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

図 8-15に，セマフォ管理システム・コール情報の記述形式を示します。

図 8-15　セマフォ管理システム・コール情報の記述形式

```
semsvc  svc_nam
```

セマフォ管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　　セマフォ管理システム・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，以下のものに限られます。

cre_sem	del_sem	sig_sem	wai_sem
preq_sem	twai_sem	ref_sem	vget_sid

8.5.3 同期通信（イベント・フラグ）管理システム・コール情報

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

図 8-16に，イベント・フラグ管理システム・コール情報の記述形式を示します。

図 8-16　イベント・フラグ管理システム・コール情報の記述形式

```
flgsvc   svc_nam
```

イベント・フラグ管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　　イベント・フラグ管理システム・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，以下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			

8.5.4 同期通信（メールボックス）管理システム・コール情報

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

図 8-17に，メールボックス管理システム・コール情報の記述形式を示します。

図 8-17　メールボックス管理システム・コール情報の記述形式

```
mbxsvc   svc_nam
```

メールボックス管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　　メールボックス管理システム・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，以下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

8.5.5 割り込み処理管理システム・コール情報

割り込み処理管理システム・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。

図 8-18に，割り込み処理管理システム・コール情報の記述形式を示します。

図 8-18 割り込み処理管理システム・コール情報の記述形式

```
intsvc  svc_nam
```

割り込み処理管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam 割り込み処理管理システム・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，以下のものに限られます。

def_int	ret_int	ret_wup	vret_clk
loc_cpu	unl_cpu	chg_ilv	ref_ilv

8.5.6　メモリ・プール管理システム・コール情報

メモリ・プール管理システム・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。

図 8-19に，メモリ・プール管理システム・コール情報の記述形式を示します。

図 8-19　メモリ・プール管理システム・コール情報の記述形式

```
mplsvc   svc_nam
```

メモリ・プール管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　　メモリ・プール管理システム・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，以下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

8.5.7 時間管理システム・コール情報

時間管理システム・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。

図 8-20に，時間管理システム・コール情報の記述形式を示します。

図 8-20 時間管理システム・コール情報の記述形式

```
timsvc  svc_nam
```

時間管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　時間管理システム・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，以下のものに限られます。

set_tim	get_tim	dly_tsk	def_cyc
act_cyc	ref_cyc		

8.5.8　システム管理システム・コール情報

システム管理システム・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。

図 8-21に，システム管理システム・コール情報の記述形式を示します。

図 8-21　システム管理システム・コール情報の記述形式

```
syssvc   svc_nam
```

システム管理システム・コール情報で記述する各項目について，以下に示します。

svc_nam　　システム管理システム・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，以下のものに限られます。

`get_ver`　　`ref_sys`　　`def_svc`　　`viss_svc`

8.6 記述上の注意点

CF 定義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，以下の順番で行います。

(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 情報の記述が開始されることの宣言
(4) SIT 情報の記述
(5) SCT 情報の記述が開始されることの宣言
(6) SCT 情報の記述

図 8-22に，CF 定義ファイルの記述イメージを示します。

図 8-22 CF 定義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 情報の記述が開始されることの宣言
sit_def

-- SIT 情報の記述
.....
.....
.....

-- SCT 情報の記述が開始されることの宣言
sct_def

-- SCT 情報の記述
.....
.....
.....
```

8.7 記述例

図 8-23に，CF 定義ファイルの記述例を示します。
なお，図 8-23の記述例では，以下に示すデータが記述されています。

- リアルタイム OS 情報
  - RX シリーズ情報

    リアルタイム OS 名 ： RX830
    バージョン番号 ： V300

- SIT 情報
  - システム情報

    基本クロック周期 ： 0x1 ms
    ディフォールト・スタック・サイズ ： 0x100 byte
    割り込みハンドラ用スタック情報 ： System Memory Pool 0 から 0x100 byte 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

  - 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

  - メモリ情報

    System Memory Pool 0 ： 0x0 番地から 0x2000 byte のメモリ領域を確保
    System Memory Pool 1 ： 0x2000 番地から 0x1000 byte のメモリ領域を確保
    User Memory Pool 0 ： 0x3000 番地から 0x7000 byte のメモリ領域を確保
    User Memory Pool 0 ： 0x20000 番地から 0x2500 byte のメモリ領域を確保
    User Memory Pool 1 ： 0x30000 番地から 0x1500 byte のメモリ領域を確保

－ タスク情報

ID	: 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	: すべての割り込みを許可
スタック情報	: System Memory Pool 0 から 0x100 byte のメモリ領域を確保
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	: 0x2
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	: すべての割り込みを禁止
スタック情報	: System Memory Pool 0 から 0x100 byte のメモリ領域を確保
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

－ セマフォ情報

ID	: 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

－ イベント・フラグ情報

ID	: 0x1
拡張情報	: 0x0
複数タスクの待ち	: 禁止
初期ビット・パターン	: 0x0
キー ID 番号	: 0x1

– メールボックス情報

ID	: 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

– 割り込みハンドラ情報

割り込みレベル	: 0x0
記述言語	: アセンブリ言語
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

– メモリ・プール情報

ID	: 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 byte のメモリ領域を確保
キー ID 番号	: 0x1

– 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

– 拡張 SVC ハンドラ情報

機能番号	: 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

– 初期化ハンドラ情報

記述言語　　：アセンブリ言語
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 情報

– タスク管理／タスク付属同期管理システム・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システム・コールとして，以下のシステム・コールを定義

sta_tsk　　exd_tsk

– 同期通信（セマフォ）管理システム・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）管理システム・コールとして，以下のシステム・コールを定義

sig_sem　　wai_sem

– 同期通信（イベント・フラグ）管理システム・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）管理システム・コールとして，以下のシステム・コールを定義

cre_flg　　del_flg　　set_flg　　wai_flg

– 同期通信（メールボックス）管理システム・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）管理システム・コールとして，以下のシステム・コールを定義

cre_mbx　　del_mbx　　snd_msg　　rcv_msg

– 割り込み処理管理システム・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システム・コールとして，以下のシステム・コールを定義

vret_clk

– メモリ・プール管理システム・コール情報

ユーザ処理プログラムで使用するメモリ・プール管理システム・コールとして，以下のシステム・コールを定義

cre_mpl　　del_mpl　　get_blk　　rel_blk

– 時間管理システム・コール情報

ユーザ処理プログラムで使用する時間管理システム・コールとして，以下のシステム・コールを定義

```
act_cyc          ref_cyc
```

– システム管理システム・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システム・コールとして，以下のシステム・コールを定義

```
viss_svc
```

図 8-23　CF 定義ファイルの記述例（1/4）

```
------------------------------------------------
-- 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  RX830  V300

-----------------------------------
-- SIT 情報の記述が開始されることの宣言
-----------------------------------
sit_def

----------------
-- SIT 情報の記述
----------------
-- システム情報
clktim  0x1
defstk  0x100
intstk  0x100:SPOL0
prttsk  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
```

図 8-23　CF 定義ファイルの記述例 (2/4)

```
maxmpl  0x2
maxcyc  0x1
maxsvc  0x1

-- メモリ情報
mem     SPOL0  0x0      0x2000
mem     SPOL1  0x2000   0x1000
mem     UPOL0  0x3000   0x7000
mem     UPOL0  0x20000  0x2500
mem     UPOL1  0x30000  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     0x2  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  0x0  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
```

図 8-23　CF 定義ファイルの記述例 (3/4)

```
-- 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_ASM  _varfunc  no_use  no_use

------------------------------------
-- SCT 情報の記述が開始されることの宣言
------------------------------------
sct_def

-----------------
-- SCT 情報の記述
-----------------
-- タスク管理／タスク付属同期管理システム・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

-- 割り込み処理管理システム・コール情報
```

図 8-23　CF 定義ファイルの記述例 (4/4)

```
intsvc  vret_clk

-- メモリ・プール管理システム・コール情報
mplsvc  cre_mpl
mplsvc  del_mpl
mplsvc  get_blk
mplsvc  rel_blk

-- 時間管理システム・コール情報
timsvc  act_cyc
timsvc  ref_cyc

-- システム管理システム・コール情報
syssvc  viss_svc
```

第 9 章　情報ファイルの生成

この章では，CF 定義ファイルから情報ファイル（システム情報テーブル，ブランチ・テーブル，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。

9.1　概要

CF 定義ファイルから情報ファイル（システム情報テーブル，ブランチ・テーブル，システム情報ヘッダ・ファイル）を生成するには，コンフィギュレータをコマンド・ラインから起動することにより行われます。

以下に，コンフィギュレータをコマンド・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“A>”はシェル・プロンプトを，“↵”はリターン・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，省略が可能な起動オプションであることを表しています。

注意　CA830 対応版のコンフィギュレータは VSH からの起動，CCV830 対応版のコンフィギュレータは MS-DOS　プロンプトからの起動となります。

```
A> cf830 [-i sit_file][-c sct_file][-d h_file][-ni][-nc][-nd][-V][-help] cf_file ↵
```

コンフィギュレータの起動オプションを，以下に示します。

-i *sit_file*　　コンフィギュレータからの出力ファイル（システム情報テーブル名）を指定します。

省略時　*cf_file* で指定された CF 定義ファイル名の末尾に“i”を付加し，拡張子を“.tbl”に置き換えたシステム情報テーブルを出力します。

注　意　本起動オプションと -ni オプションを同時に指定した場合，入力が前者のオプションは無効となり，入力が後者のオプションは有効となります。

-c *sct_file*　　コンフィギュレータからの出力ファイル（ブランチテーブル名）を指定します。

省略時　*cf_file* で指定された CF 定義ファイル名の末尾に“c”を付加し，拡張子を“.tbl”に置き換えたブランチ・テーブルを出力しま

す。

注意　本起動オプションと -nc オプションを同時に指定した場合，入力が前者のオプションは無効となり，入力が後者のオプションは有効となります。

-d *h_file*　コンフィギュレータ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 CF 定義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 -nd オプションを同時に指定した場合，入力が前者のオプションは無効となり，入力が後者のオプションは有効となります。

-ni　システム情報テーブルを出力しません。

省略時　*cf_file* で指定された CF 定義ファイル名の末尾に“i”を付加し，拡張子を“.tbl”に置き換えたシステム情報テーブルを出力します。

注　意　本起動オプションと -i *sit_file* オプションを同時に指定した場合，入力が前者のオプションは無効となり，入力が後者のオプションは有効となります。

-nc　ブランチ・テーブルを出力しません。

省略時　*cf_file* で指定された CF 定義ファイル名の末尾に“c”を付加し，拡張子を“.tbl”に置き換えたブランチ・テーブルを出力します。

注　意　本起動オプションと -c *sct_file* オプションを同時に指定した場合，入力が前者のオプションは無効となり，入力が後者のオプションは有効となります。

-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 CF 定義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* オプションを同時に指定した場合，入力が前者のオプションは無効となり，入力が後者のオプションは有効となります。

-V　コンフィギュレータのバージョン情報を標準出力（または，標準エラー出力）に出力します。

省略時　コンフィギュレータの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した場合，他の起動オプションはすべて無効となります。

-help　　コンフィギュレータの起動オプションの使い方を標準出力（または，標準エラー出力）に出力します。

省略時 ： コンフィギュレータ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した場合，他の起動オプションはすべて無効となります。

cf_file　　コンフィギュレータへの入力ファイル（CF 定義ファイル名）を指定します。

省略時 ： 本起動オプションを省略することはできません。

9.2 コマンド入力例

コンフィギュレータのコマンド入力例を，以下に示します。

- `cf830 -i sitfile.tbl -c sctfile.tbl -d hfile.h cffile.cf`

  CF 定義ファイル `cffile.cf` を読み込み，システム情報テーブル `sitfile.tbl`，ブランチ・テーブル `sctfile.tbl`，システム情報ヘッダ・ファイル `hfile.h` を出力します。

- `cf830 -i sitfile.tbl cffile.cf`

  CF 定義ファイル `cffile.cf` を読み込み，システム情報テーブル `sitfile.tbl`，ブランチ・テーブル `cffilec.tbl`，システム情報ヘッダ・ファイル `cffile.h` を出力します。

- `cf830 -c sctfile.tbl cffile.cf`

  CF 定義ファイル `cffile.cf` を読み込み，システム情報テーブル `cffilei.tbl`，ブランチ・テーブル `sctfile.tbl`，システム情報ヘッダ・ファイル `cffile.h` を出力します。

- `cf830 -d hfile.h cffile.cf`

  CF 定義ファイル `cffile.cf` を読み込み，システム情報テーブル `cffilei.tbl`，ブランチ・テーブル `cffilec.tbl`，システム情報ヘッダ・ファイル `hfile.h` を出力します。

- `cf830 cffile.cf`

  CF 定義ファイル `cffile.cf` を読み込み，システム情報テーブル `cffilei.tbl`，ブランチ・テーブル `cffilec.tbl`，システム情報ヘッダ・ファイル `cffile.h` を出力します。

- `cf830 -nc -nd cffile.cf`

  CF 定義ファイル `cffile.cf` を読み込み，システム情報テーブル `cffilei.tbl` を出力します。

- `cf830 -V`

  コンフィギュレータのバージョン情報を標準出力（または，標準エラー出力）に出力します。

- `cf830 -help`

  コンフィギュレータの起動オプションの使い方を標準出力（または，標準エラー出力）に出力します。

9.3 メッセージ

メッセージは，コンフィギュレータが処理を実行中に，「CF 定義ファイルに誤った定義が行われている」などといった記述ミスを検出した際に生成され，標準出力（または，標準エラー出力）に出力されます。

なお，コンフィギュレータでは，メッセージを 3 つのレベル（致命的エラー，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 ･･･ 致命的エラー

致命的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 ： パラメータの記述が省略されている。

索引

N

O

P

R

S

T

U

V

W

い

お

か

き

保守／廃止

保守／廃止

お問い合わせ先

【技術的なお問い合わせ先】

ＮＥＣ半導体テクニカルホットライン（インフォメーションセンター）	電　話 FAX E-mail	：044-548-8899 ：044-548-7900 ：s-info@saed.tmg.nec.co.jp

【営業関係お問い合わせ先】

半導体第一販売事業部 半導体第二販売事業部 半導体第三販売事業部	〒108-8001	東京都港区芝５－７－１　（日本電気本社ビル）	(03)3454-1111
中部支社 半導体第一販売部 半導体第二販売部	〒460-8525	愛知県名古屋市中区錦１－17－１　（日本電気中部ビル）	(052)222-2170 (052)222-2190
関西支社 半導体第一販売部 半導体第二販売部 半導体第三販売部	〒540-8551	大阪府大阪市中央区城見１－４－24　（日本電気関西ビル）	(06) 945-3178 (06) 945-3200 (06) 945-3208

北海道支社	札　幌	(011)231-0161	宇都宮支店	宇都宮	(028)621-2281	北陸支社	金　沢	(076)232-7303
東北支社	仙　台	(022)267-8740	小山支店	小　山	(0285)24-5011	富山支店	富　山	(0764)31-8461
岩手支店	盛　岡	(019)651-4344	甲府支店	甲　府	(0552)24-4141	福井支店	福　井	(0776)22-1866
郡山支店	郡　山	(0249)23-5511	長野支社	松　本	(0263)35-1662	京都支社	京　都	(075)344-7824
いわき支店	いわき	(0246)21-5511	静岡支社	静　岡	(054)254-4794	神戸支社	神　戸	(078)333-3854
長岡支店	長　岡	(0258)36-2155	立川支社	立　川	(042)526-5981,6167	中国支社	広　島	(082)242-5504
水戸支店	水　戸	(029)226-1717	埼玉支社	大　宮	(048)649-1415	鳥取支店	鳥　取	(0857)27-5311
土浦支店	土　浦	(0298)23-6161	千葉支社	千　葉	(043)238-8116	岡山支店	岡　山	(086)225-4455
群馬支店	高　崎	(027)326-1255	神奈川支社	横　浜	(045)682-4524	松山支店	松　山	(089)945-4149
太田支店	太　田	(0276)46-4011	三重支店	津	(059)225-7341	九州支社	福　岡	(092)261-2806

保守／廃止

アンケート記入のお願い

お手数ですが，このドキュメントに対するご意見をお寄せください。今後のドキュメント作成の参考にさせていただきます。

［ドキュメント名］RX830（μITRON Ver.3.0） ユーザーズ・マニュアル インストレーション編
（U13151JJ2V0UM00（第2版））

［お名前など］（さしつかえのない範囲で）

御社名（学校名，その他） （ ）
ご住所 （ ）
お電話番号 （ ）
お仕事の内容 （ ）
お名前 （ ）

1. ご評価（各欄に○をご記入ください）

項　　目	大変良い	良　い	普　通	悪　い	大変悪い
全体の構成					
説明内容					
用語解説					
調べやすさ					
デザイン，字の大きさなど					
その他（　　）					
（　　）					

2. わかりやすい所（第　章，第　章，第　章，第　章，その他　　）

理由［　　］

3. わかりにくい所（第　章，第　章，第　章，第　章，その他　　）

理由［　　］

4. ご意見，ご要望

5. このドキュメントをお届けしたのは
NEC販売員，特約店販売員，NEC半導体ソリューション技術本部員，
その他（　　）

ご協力ありがとうございました。
下記あてにFAXで送信いただくか，最寄りの販売員にコピーをお渡しください。

NEC半導体テクニカルホットライン
FAX：（044）548-7900

キリトリ