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

μSAP703000-B09
μSAP70732-B09

手書き文字認識ミドルウエア　Ver.2

対象デバイス

μSAP703000-B09：V850 ファミリ ™

μSAP70732-B09：V810 ファミリ ™

資料番号　U13253JJ2V0UM00（第２版）
発行年月　August 1998 NS　CP(K)

保守／廃止

〔メ　モ〕

保守／廃止

目次要約

保守／廃止

M7A 94.11

保守／廃止

本版で改訂された主な箇所

箇 所	内　容
p.18	1. 3. 3　動作環境に V810 ファミリ追加
p.18	Crg_SetPoint の使用するワーク・メモリの容量変更
p.19	1. 3. 3　(4) サポート・ツール修正
p.19	1. 3. 4　性能に V810 ファミリ追加
p.27	2. 2. 1　(2) 構造体定義修正
p.45	図 3-1　パッケージの内容（V810 ファミリ）追加
p.47	表 3-1　セクション名（V810 ファミリ）追加
p.47	3. 2　(1) NEC 版 修正
p.51	3. 5　サンプル・プログラムの作成（V810 ファミリ）追加
p.71	付録 G　サンプル・ソース修正

本文欄外の★印は，本版で改訂された主な箇所を示しています。

保守／廃止

はじめに

対象者　このマニュアルは，V810 ファミリ，V850 ファミリの応用システムを設計，開発するユーザを対象としています。

目　的　V810 ファミリ，V850 ファミリの応用，開発をする際にサポートするミドルウエアを，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
- ライブラリ仕様
- インストレーション
- システム例

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およびＣ言語に関する一般知識を必要とします。

V810 ファミリ，V850 ファミリ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・マニュアルのハードウエア編を参照してください。

V810 ファミリ，V850 ファミリの命令機能を知りたいとき
→各製品のユーザーズ・マニュアルのアーキテクチャ編を参照してください。

凡　例　注　：本文中に付けた注の説明
注意：気を付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 xxxxB
10 進数…xxxx
16 進数…xxxxH または 0x 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

保守／廃止

関連資料 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。

V810 ファミリに関する資料

製品名		データ・シート	ユーザーズ・マニュアル	
愛称	品名		ハードウエア編	アーキテクチャ編
V810™	μPD70732	U10691J	U10038J	U10082J
V821™	μPD70741	U11678J	U10077J	

V850 ファミリに関する資料

製品名		データ・シート	ユーザーズ・マニュアル	
愛称	品名		ハードウエア編	アーキテクチャ編
V852™	μPD703002	U11826J	U10038J	U10243 J
	μPD70P3002	U11827J		
V853™	μPD703003	U12261J	U10913J	
	μPD703003A,703004A,703025A	U13188J		
	μPD70F3003	U12036J		
	μPD70F3003A,70F3025A	U13189J		
V854™	μPD703008	作成予定	U11969J	
	μPD703008Y	作成予定		
	μPD70F3008	U12756J		
	μPD70F3008Y	U12755J		
V850/SA1™	μPD703015	作成予定	U12768J	
V850E/MS1™	μPD703100,703101,703102	作成予定	U12688J	U12197J
	μPD70F3102	作成予定		

保守／廃止

○V810 ファミリ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
CA732（C コンパイラ）	操作編(UNIX™ベース)	U11013J
	操作編(Windows™ベース)	U11068J
	アセンブリ言語編	U11016J
	C 言語編	U11010J
RX732（リアルタイム OS）	基礎編	U10346J
	テクニカル編	U10347J
	ニュークリアス・インストレーション編	U10490J

○V850 ファミリ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
IE-703002-MC（V852,V853,V854,V850/SA1 用インサーキット・エミュレータ）		U11595J
IE-703003-MC-EM1（V853 用インサーキット・エミュレータ・オプション・ボード）		U11569J
IE-703008-MC-EM1（V854 用インサーキット・エミュレータ・オプション・ボード）		U12420J
IE-703017-MC-EM1（V850/SA1 用インサーキット・エミュレータ・オプション・ボード）		U12898J
IE-703100-MC（V850E/MS1 用インサーキット・エミュレータ）		U12887J
CA850（C コンパイラ）	操作編(UNIX ベース)	U12839J
	操作編(Windows ベース)	U12827J
	C 言語編	U12840J
	アセンブリ言語編	U10543J
	プロジェクト・マネージャ編(Windows ベース)	U11991J
ID850（C ソース・ディバッガ）	操作編(Windows ベース)	U11196J
RX850（リアルタイム OS）	基礎編	U13430J
	テクニカル編	U13431J
	ニュークリアス・インストレーション編	U11038J
	インストレーション編(Windows ベース)	U13410J
	ディバッガ編(Windows ベース)	U11158J
RD850（タスク・ディバッガ）		U11158J
RD850（Ver.3.0）（タスク・ディバッガ）		U13737J
AZ850（システム・パフォーマンス・アナライザ）操作編		U11181J

Green Hills Software™, Inc.（GHS 社）製ツールに関する資料

GHS 社製ツールは，日本国内では下記で取り扱っております。各種製品とそれに関する資料については，下記へお問い合わせください。

(株)アドバンスド データ コントロールズ　TEL（03）3576-5351

保守／廃止

〔メ　モ〕

保守／廃止

目　次

保守／廃止

図の目次

保守／廃止

表の目次

第1章　概　　説

この章では，ミドルウエアと手書き文字認識の概要について説明します。

1.1　ミドルウエア

ミドルウエアとは，プロセッサの性能を最大限に引き出すようにチューニングされたソフトウエア群のことです。

現在では高性能 RISC プロセッサが比較的安く市場に投入され，従来，専用ハードウエアに頼っていた処理を「高性能 RISC プロセッサ」＋「ソフトウエア」というアプローチで実現できるようになりました。この「ソフトウエア」をミドルウエアと呼んでいます。

NEC では，ヒューマン・マシン・インタフェースおよび信号処理技術をミドルウエアの形で用意しています。さまざまなユーザのニーズに対応して，優れたシステム・ソリューションを提供しています。

備考　RISC：Reduced Instruction Set Computer

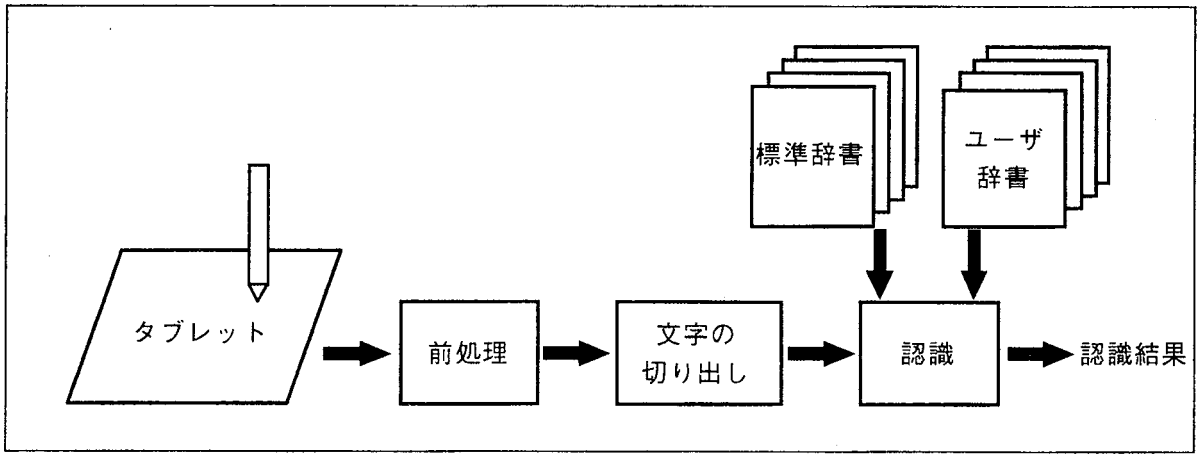
1.2　オンライン手書き文字認識

オンライン手書き文字認識は，タブレット上にペンで入力された手書き文字を直接認識するものです。オンラインで得られるペンの軌跡から文字を認識します。

1.2.1　システム構成

オンライン手書き文字認識のシステム構成を図 1-1 に示します。

図 1−1　システム構成



(1) 前処理

タブレットから文字の軌跡が，一定時間間隔でサンプリングされた筆点情報（x 座標，y 座標，ペン状態[注]）の時系列として入力されます。入力された筆点情報に対して，冗長なデータを取り除くなどの前処理を行い，座標データに変換します。この処理により，認識対象文字の座標データをセットする文字バッファのサイズを小さくすることができます。

注　ペンがタブレットから離れている状態，またはペンがタブレットに接触している状態

(2) 文字の切り出し

前処理された座標データを文字単位に分割します。文字の区切れの判断は，文字を入力する領域を示す枠の設定や，文字間でペンがタブレットから離れている時間などを用いて行うことができます。ただし，このミドルウエアは文字の切り出しは行わないので，アプリケーション側で行ってください。前者の文字枠を設定することにより，文字の切り出しを行うのが一般的です。

(3) 認　識

切り出された 1 文字分の座標データに対して，認識対象文字の情報をセットした辞書を用いて認識を行います。辞書は，NEC から提供する標準辞書のほかに，ユーザが入力した文字を登録するユーザ辞書を同時に使用することができます。辞書は複数あってもかまいません。

保守／廃止

1.3 システム概要

1.3.1 特　徴

○ 認識対象文字は約 4200 字種（JIS 第 1 水準約 3400 字種，JIS 第 2 水準約 800 字種）
○ 筆順，画数の変動に強い
○ 複数のユーザ辞書を作成可能
○ NEC 製/GHS 社製の C コンパイラの C 言語からの呼び出しが可能
○ ワーク・エリアをダイナミックにアロケート可能

1.3.2 機　能

(1) 文字認識

入力された 1 文字分の座標データに対して認識を行い，最大 10 位の候補文字のシフト JIS コードを結果として返します。

(2) ユーザ辞書登録

一度に一字種を登録できます。最初に登録する字種のシフト JIS コードを入力して，1 つの字種に対して 1 回以上文字を入力することで行います。同じシフト JIS コードに形の違う文字を登録することもできます。ユーザ辞書は複数作成することができ，1 つのユーザ辞書には複数の字種が登録できます。

また，ユーザ辞書に登録された全文字のメモリ・サイズを取得することができます。これにより，アプリケーション側でユーザ辞書のデータを取り出し，ROM 化などすることができます。ROM 化などした複数のユーザ辞書も認識に用いることができます。

保守／廃止

1.3.3 動作環境

(1) 対象 CPU

★ ・V810 ファミリ（V810,V821）

・V850 ファミリ（V852,V853,V854,V850/SA1,V850E/MS1）

(2) 必要メモリ

手書き文字認識ミドルウエアを動作させるために必要な ROM/RAM の容量を次に示します。

ROM/RAM	用　途	容　量
ROM	プログラム	約 60K バイト
	データ	約 60K バイト
	標準辞書	約 450K バイト
RAM	データ領域	32K バイト[注1]
	スタック領域	約 2K バイト
	ユーザ辞書	約 300 バイト／パターン[注2]

注 1. 座標データをセットする文字バッファの領域は除きます。また，各ライブラリ関数実行後も保持しなければならない RAM 領域のリザーブ・エリア（**2.2.1 構造体**参照）の容量は，400 バイトです。残りは各ライブラリ関数を実行していない間は，開放してもかまいません。各ライブラリ関数で使用する RAM 領域のワーク・エリアを次に示します。

ライブラリ関数名	使用するワーク・メモリの容量
Crg_Initialize	0 バイト
Crg_SetGridParam	0 バイト
★ Crg_SetPoint	0 バイト
Crg_RecogChar	32K バイト
Crg_Uninitialize	0 バイト
Crg_StopProc	0 バイト
Crg_DicClearDic	0 バイト
Crg_DicCreate	0 バイト
Crg_DicEntryChar	32K バイト
Crg_DicRegister	0 バイト
Crg_DicEraseChar	0 バイト
Crg_DicInformation	0 バイト

2. ユーザ辞書では，通常 1 文字に対して複数の文字を入力しても，1 つのパターンが作成されますが，入力した文字の書き方によって，認識性能向上のため複数のパターンが作成されることがあります（最大 4 パターン：約 1.2 K バイト／文字）。なお，1 回の文字入力で作成されるパターンは最大で 1 つです。1 回だけ文字を入力した場合は，作成されるパターンは必ず 1 つになります。

(3) 推奨するタブレット

【 分 解 能 】 0.05 mm／point
【 読み取り速度 】 25 ms／point （40 points／秒）
【 出力データ 】 筆点情報（x 座標，y 座標，ペン状態）
左上隅を原点として，右方向を x 軸，下方向を y 軸とします。
ペン状態は，ペンがタブレットから離れている状態では 0，タブレットに接触している状態では 1 となります。
【文字枠のサイズ】 縦 1.4 cm×横 1.4 cm

(4) サポート・ツール

★ ・NEC 製 C コンパイラ・パッケージ：CA732（Ver.1.10 以上）
：CA850（Ver.1.10 以上）
★ ・GHS 社製 C コンパイラ ：CCV732（Ver.1.8.8 Release 3.0.1 以上）
：CCV850（Ver.1.8.8 Release 3.0.1 以上）

1.3.4 性　能

★ (1) V810 ファミリ

【 認 識 率 】 平均約 94%（NEC のデータによる）
【 認識時間 】 平均約 0.1 秒（NEC のデータによる）
【 条　件 】 CPU：V810（動作周波数：25MHz），各 32 ビット・バスの場合
RAM：0 ウエイト
ROM：0 ウエイト

(2) V850 ファミリ

【 認 識 率 】 平均約 94%（NEC のデータによる）
【 認識時間 】 平均約 0.1 秒（NEC のデータによる）
【 条　件 】 CPU：V852（動作周波数：25MHz），各 16 ビット・バスの場合
RAM：3 ウエイト
ROM：3 ウエイト

保守／廃止

1.4 認識対象文字

1.4.1 認識対象字種（4213字種）

(1) ひらがな：83字種

ぁあぃいぅうぇえぉおかがきぎくぐけげこごさざしじすずせぜそぞただちぢっつづてでとどなにぬねの
はばぱひびぴふぶぷへべぺほぼぽまみむめもゃやゅゆょよらりるれろゎわゐゑをん

(2) カタカナ：86字種

ァアィイゥウェエォオカガキギクグケゲコゴサザシジスズセゼソゾタダチヂッツヅテデトドナニヌネノ
ハバパヒビピフブプヘベペホボポマミムメモャヤュユョヨラリルレロヮワヰヱヲンヴヵヶ

(3) 漢字：3807字種

○ JIS第1水準：2965字種（付録A　JIS第1水準漢字を参照）

○ JIS第2水準：842字種（付録B　JIS第2水準漢字を参照）

(4) 数字：10字種

0 1 2 3 4 5 6 7 8 9

(5) 英字：52字種（小文字：26字種，大文字：26字種）

a b c d e f g h i j k l m n o p q r s t u v w x y z

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

(6) 記号：175字種

(a) 手書きで入力できる記号：158字種

，。、．・：；？！￣＿ゝゞ〃仝々〆〇ー－／＼～‥ ‘ ’ “ ” （） 〔〕 ［］ ｛｝ 〈〉 《》 「」 『』 ＋－
±×÷＝≠＜＞≦≧∞∴♂♀° ′ ″ ℃ ¥ ＄ ¢ £ ％ ＃ ＆ ＊ ＠ § ☆ ○ ◎ ◇ □ △ ▽ ※ 〒 → ← ↑ ↓ ≡ ≒ √ ∫ Γ
Λ Π Σ Φ Ψ Ω α β γ δ ε ζ η θ ι κ λ μ ν ξ π ρ σ τ υ φ χ ψ ω ① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩ ⑪ ⑫ ⑬ ⑭ ⑮ ⑯
⑰ ⑱ ⑲ ⑳ II III IV VI VII VIII IX XI XII ii iii iv vi vii viii ix xi xii

(b) 空白文字として認識される文字：1字種

␣

(c) 他の文字の認識候補となる記号：16字種

★●◆■▲▼【】Δ o l V X i v x（付録F　他の文字の認識候補になる文字を参照）

保守／廃止

1.4.2 文字コード

認識結果として出力される文字コードは，シフトJISコードです。ただし，シフトJISコードが割り当てられていない記号に関しては，次のように割り当ててあります。

①～⑳：8740～8753
Ⅰ～Ⅹ：8754～875D
Ⅺ～Ⅻ：8776～8777
ⅰ～ⅹ：EEEF～EEF8
ⅺ～ⅻ：8778～8779

1.4.3 筆記法

○ 漢字の字体は，新JIS字体です。
○ 英数字の書き方は，**付録C 英数字の書き方**を参照してください。
○ 大文字／小文字などの同型文字の書き分け方は，**付録D 同型文字の書き分け方**を参照してください。
○ 略字の書き方は，**付録E 略字の書き方**を参照してください（**付録E**に示す31種の略字体に対応）。
ただし，「風(9597)」の略字である「凡」は，「凡(967d)」の次候補として認識結果に出力します。
○ 空白文字として認識される文字（␣）は，枠の横幅いっぱいに，左から右へ1画で書きます。

保守／廃止

〔メ　モ〕

第２章　ライブラリ仕様

2.1　処理概要

2.1.1　ライブラリ関数

ライブラリ関数の一覧を次に示します。

表2−1　ライブラリ関数一覧

処　理	関　数		説　明
文字認識／辞書登録共通	初期化	crg_Initialize	手書き文字認識システムを初期化します。
	終了	crg_Uninitialize	手書き文字認識システムを終了します。
	処理停止	crg_StopProc	処理を中断させます。
文字認識	文字枠セット	crg_SetGridParam	文字枠の大きさの情報をセットします。
	筆点単位処理	crg_SetPoint	筆点情報を基に，座標データをセットします。
	認識処理	crg_RecogChar	認識処理を行い，認識結果を出力します
辞書登録	ユーザ辞書クリア	crg_DicClearDic	ユーザ辞書をクリアします。
	登録開始	crg_DicCreate	登録する文字コードを決め,登録の準備を行います。
	登録用文字入力	crg_DicEntryChar	学習に必要な特徴パラメータを抽出して，累積していきます。
	文字登録	crg_DicRegister	辞書データを作成して，ユーザ辞書に登録します。
	登録文字消去	crg_DicEraseChar	ユーザ辞書から指定された文字を消去します。
	登録文字数出力	crg_DicInformation	ユーザ辞書に登録されている文字数とメモリ・サイズを返します。

2.1.2　機能概要

(1) 文字認識

(a) 前処理

筆点情報は，割り込みを利用してタブレットから入力されます。入力された筆点情報を，１つの筆点ごとに「筆点単位処理」により座標データに変換します。変換された座標データは，文字バッファに順次セットされます。

(b) 認識処理

文字バッファにセットされている１文字分の文字データは，「認識処理」により認識され，認識結果を出力します。

保守／廃止

(c) 認識処理の中断

認識処理を中断する場合は，割り込みを利用して「処理停止」により中断することができます。

(2) ユーザ辞書登録

(a) 文字登録

１つのユーザ辞書に，複数の字種の文字を登録できます。既存のユーザ辞書に追加登録することもできます。文字登録は，「登録開始」「登録用文字入力」「文字登録」を用いて行います。

最初に「登録開始」で登録する文字に対するシフトJISコードを入力します。

続いて登録したい文字がタブレットから入力されると，「登録用文字入力」処理によりユーザ辞書に保存される形式に変換され保存されます。認識性能を向上させるためには，複数回入力することにより，文字のばらつきをカバーすることができます。ただし，文字の形は統一する必要があります。

入力が終了すると，「文字登録」により文字を登録します。新規にユーザ辞書を作成する場合は，文字登録をする前に「ユーザ辞書クリア」を一度行ってください。

(b) 文字消去

ユーザ辞書に登録されている文字は，「登録文字消去」に対象文字のユーザ辞書中での位置を与えると消去できます。

(c) ユーザ辞書のROM化

「登録文字数出力」により，ユーザ辞書に登録された全文字のメモリ・サイズを取得できます。アプリケーション側で，取得したメモリ・サイズと，ユーザ辞書の先頭アドレスを用いてユーザ辞書のデータを取り出し，ROM化などをすることができます。

(3) 文字枠

文字枠に対するデータの関係を図 2-1 に示します。

図 2−1　文字枠とデータの関係

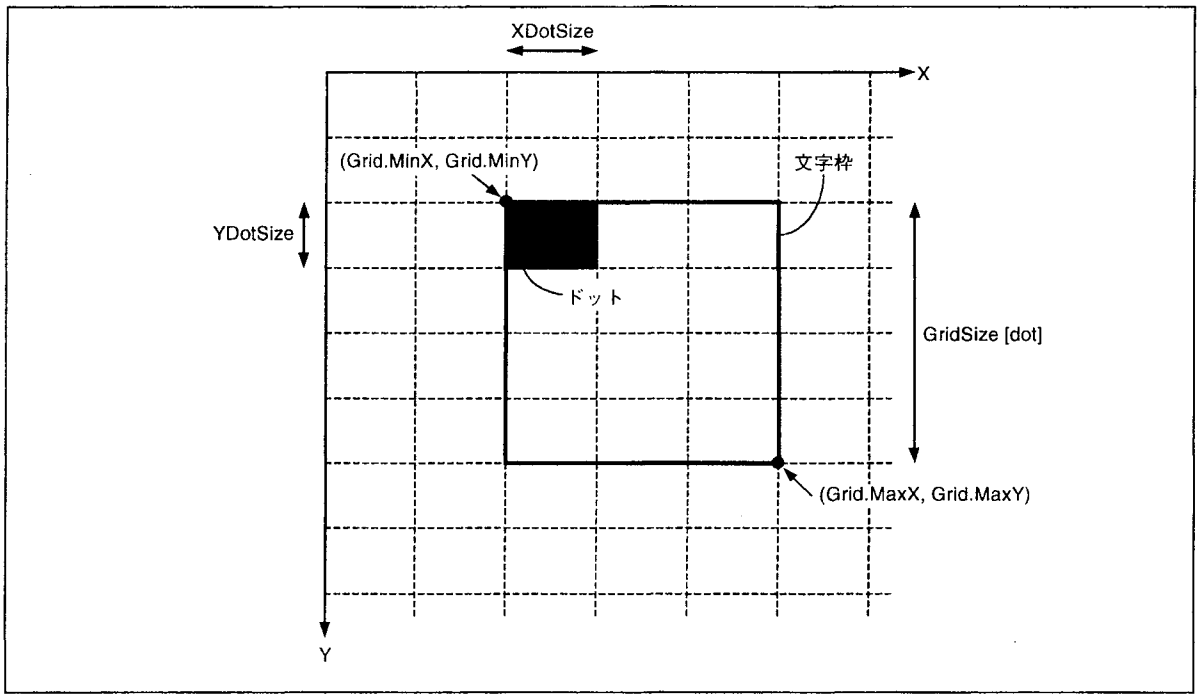


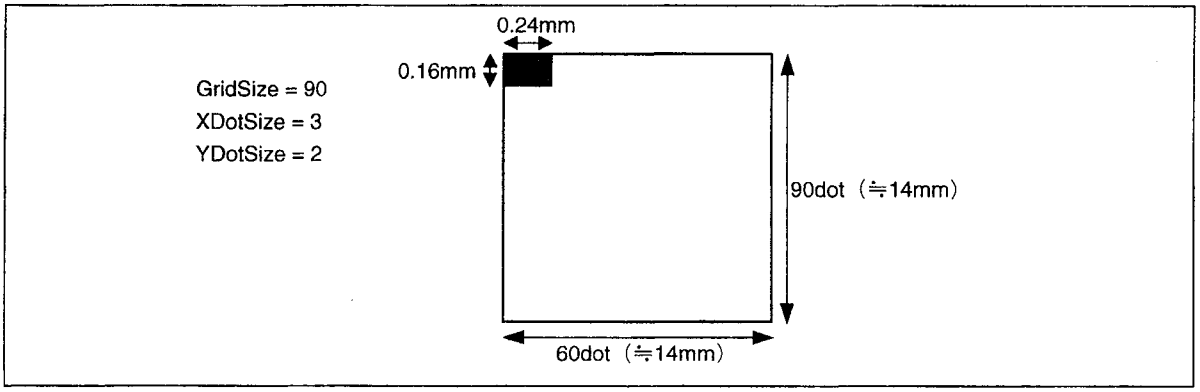
図 2-1 のように，横右方向が x 方向，縦下方向が y 方向である座標系を用います。

GridSize は，文字枠の大きさを表します。タブレット上のドット数単位で表された縦，横の大きい方の長さです。

XDotSize と YDotSize は，ドットの x 方向と y 方向の大きさの比を整数値で表したものです（x 方向の大きさ：y 方向の大きさ ＝XDotSize：YdotSize）。x 方向と y 方向の大きさが等しい場合は，XDotSize = 1，YDotSize = 1 とします。

Grid は，現在文字が書かれている文字枠の左上，右下の座標値をタブレット上のドット数単位で表したものです。図 2-2 に文字枠の例を示します。

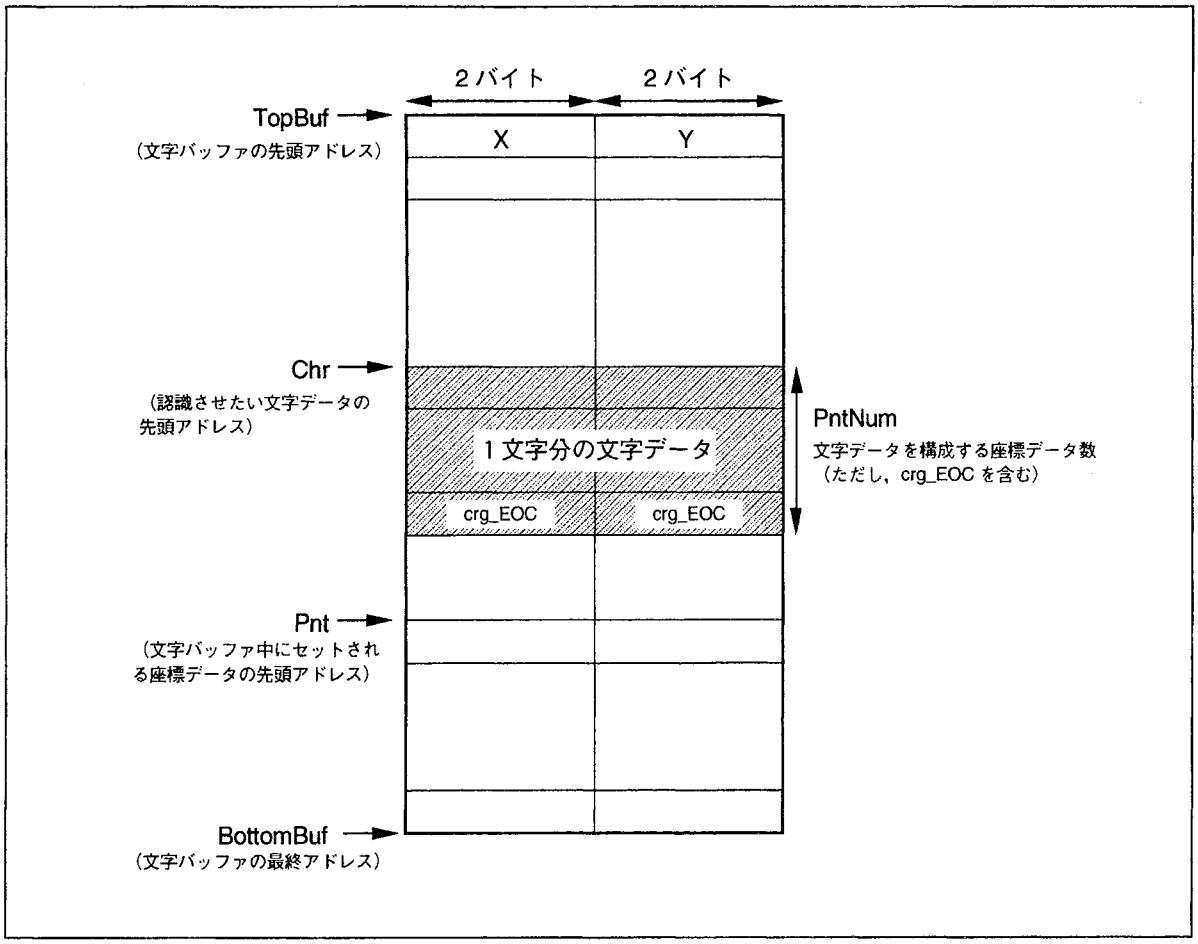
図 2−2　文字枠の例



(4) 文字バッファ

Ultwriter-V で認識される文字データは，文字バッファにセットされます。文字バッファは，ライブラリ関数を呼ぶ側で確保してください（文字バッファは，**1.3.3 (2) 必要メモリ**のワーク・メモリの容量には含まれません）。 文字バッファには，ライブラリ関数「筆点単位処理」により，冗長な座標データが取り除かれた座標データ列が保存され，ストロークの切れ目には，crg_EOC コードが書かれます。文字バッファは，リング・バッファになっています。構造については図 2-3 に示します。推奨する文字バッファ・サイズは，4000 バイト（1000 点分の座標データを格納）です。

図 2−3　文字バッファの構造



TopBuf, BottomBuf の設定，Chr, Pnt，PntNum の初期設定と更新は，ライブラリ関数を呼ぶ側のメイン・プログラムで行ってください。なお，BottomBuf-1 が指している領域まで，x 座標，y 座標は格納されます。

保守／廃止

2.2　関数仕様

各ライブラリ関数で用いる構造体と，各ライブラリ関数の仕様（C 言語記述）について説明します。

注意 1. ライブラリ関数実行中に，crg_StopProc 以外の他のライブラリ関数を呼ばないでください。
2. ライブラリ関数の引き数の各アドレスは，必ず 4 の倍数にしてください。

2.2.1　構造体

(1) 定数定義

次に必要な定数を示します。

表 2−2　定　　数

定数名	数　値	説　　明
crg_MAXCAND	10	認識結果の最大候補数
crg_EOC	0x7FFF	ストロークの切れ目
crg_SPOINTSIZE	1000	内部バッファに格納できる座標データ数

★　(2) 構造体定義

次にこのライブラリ関数で使用する構造体を示します。

(a) V810 ファミリ，V850 ファミリ共通に用いる構造体

```
typedef struct {
  short   X;                /* 座標データの x 座標 */
  short   Y;                /* 座標データの y 座標 */
} CRGPOINT;

typedef struct {
  CRGPOINT     Point;     /* 座標データ */
  short        Pen;       /* ペン状態 */
} CRGHWDATA;

typedef struct {
  CRGPOINT     *TopBuf;          /* 文字バッファの先頭アドレス */
  CRGPOINT     *BottomBuf;       /* 文字バッファの最終アドレス */
 } CRGBUF;
```

保守／廃止

```
typedef struct {
    short           Warning;  /* ワーニング・フラグ（0：OK, 1：ワーニング） */
    short           CandNo;   /* 文字の候補数（最大値は crg_MAXCAND） */
    CRGRECT         InData;   /* 入力文字の大きさ */
    CRGCAND         Candidate[crg_MAXCAND];    /* 認識候補 */
} CRGRESULT;

typedef struct {
    short           MaxX;     /* 最大 x 座標 */
    short           MaxY;     /* 最大 y 座標 */
    short           MinX;     /* 最小 x 座標 */
    short           MinY;     /* 最小 y 座標 */
} CRGRECT;

typedef struct {
    short           Value;    /* 距離値 */
    unsigned short  Index;    /* 文字コード（シフト JIS コード） */
} CRGCAND;

typedef struct {
    CRGPOINT        *Chr;     /* 認識させたい文字データの先頭アドレス */
    Short           PntNum;   /* 文字データを構成する座標データ数（crg_EOC もカウントする） */
    CRGBUF          *ChrBuf;  /* 文字バッファのアドレス情報へのアドレス */
} CRGCHRPAT;

typedef struct {
    long  *DicArea;           /* 辞書バッファ先頭アドレス */
    short DicState;           /* 辞書状態 */
} CRGDICSET;
```

(b) V810 ファミリ専用の構造体

```
typedef struct {
    long  *ReserveArea;      /* リザーブ・エリア （ライブラリ関数実行後は常に保持）先頭アドレス */
    long  *WorkArea;         /* ワーク・エリア（ライブラリ関数ごとに使用）先頭アドレス */
} CRGMEM;
```

(c) V850 ファミリ専用の構造体

```
typedef struct {
    long  *ReserveArea;      /* リザーブ・エリア （ライブラリ関数実行後は常に保持）先頭アドレス */
    long  *WorkArea;         /* ワーク・エリア（ライブラリ関数ごとに使用）先頭アドレス */
    short IDRAMSw;           /* 内蔵データ RAM スイッチ */
    short *IDRAMAdrs;        /* 内蔵データ RAM 先頭アドレス */
} CRGMEM;
```

保守／廃止

2.2.2　外部仕様

(1) crg_Initialize 関数

【分　類】　文字認識／辞書登録共通
【関 数 名】　crg_Initialize
【機能概要】　手書き文字認識システムを初期化します。
【形　式】　int　crg_Initialize (CRGMEM *MemInfo);
【引 き 数】　CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）
【返 り 値】　0x00　　正常
　　0x0e　　与えられたアドレスが４の倍数でない
【機　能】　手書き文字認識システムを初期化します。手書き文字認識システムを使用する前に一度呼び出す必要があります。

メモリ領域（RAM）は，各ライブラリ関数を実行したあとに開放できるワーク・エリア（WorkArea）と，常に保持しなければならないリザーブ・エリア（ReserveArea）から構成されています。各ライブラリ関数は，これらのメモリ領域のアドレスを格納している構造体 CRGMEM を引き数として受け取り，その領域を使用します。なお，メモリ領域の確保は，ライブラリ関数を呼ぶ側で行ってください。確保するメモリの容量は，**1.3.3（2）必要メモリ**を参照してください。

MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

V850 ファミリを使用する場合は，MemInfo→IDRAMsw （内蔵 RAM スイッチ）は次のように指定してください。内蔵データ RAM を 2K バイト使用する場合は，使用する領域の先頭アドレスを指定してください（現在未対応）。

0：内蔵データ RAM（全領域開放）
1：内蔵データ RAM（2K バイト使用）

保守／廃止

(2) crg_Uninitialize 関数

【 分　類 】　文字認識／辞書登録共通
【関 数 名】　crg_Uninitialize
【機能概要】　手書き文字認識システムを終了します。
【 形　式 】　int　crg_Uninitialize (CRGMEM *MemInfo);
【引 き 数】　CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）
【返 り 値】　0x00　正常
　　0x0e　与えられたアドレスが４の倍数でない
【 機　能 】　手書き文字認識システムを終了します。システム使用後に一度呼び出す必要があります。
　　MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

(3) crg_StopProc 関数

【 分　類 】　文字認識／辞書登録共通
【関 数 名】　crg_StopProc
【機能概要】　処理を中断させます。
【 形　式 】　int　crg_StopProc (CRGMEM *MemInfo);
【引 き 数】　CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）
【返 り 値】　0x00　正常
　　0x0e　与えられたアドレスが４の倍数でない
【 機　能 】　認識処理（crg_RecogChar()）および登録用文字入力（crg_DicEntryChar()）を中断させます。
　　MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

(4) crg_SetGridParam 関数

【分　類】　文字認識

【関 数 名】　crg_SetGridParam

【機能概要】　文字枠の大きさの情報をセットします。

【形　式】　int　crg_SetGridParam (short GridSize, short XDotSize, short YDotSize, CRGMEM *MemInfo);

【引 き 数】

short	GridSize	文字枠の大きさ（入力）
short	XdotSize	x 方向のドットの大きさ（入力）
short	YdotSize	y 方向のドットの大きさ（入力）
CRGMEM	*MemInfo	メモリ情報へのアドレス（入力）

【返 り 値】

0x00	正常
0x01	文字枠の大きさが範囲外
0x02	指定された文字枠のドットの大きさが範囲外
0x0e	与えられたアドレスが 4 の倍数でない

【機　能】　文字枠の大きさの情報より必要なパラメータを入力します。crg_SetGridParam 関数を呼ばない場合，パラメータは次に示すディフォールト値になります。

GridSize = 280
XDotSize =1
YDotSize = 1

文字枠を使用しない場合は，GridSize に通常書かれる文字の大きさを入力してください。入力できる文字枠の大きさを次に示します。この条件を満たさない場合，エラー（返り値：0x01）になります。

$0 \leqq \text{GridSize} \leqq 0x7FFF$

入力可能な文字枠のドットの大きさを次に示します。この条件を満たさない場合，エラー（返り値：0x02）になります。

0 < XdotSize < 512
0 < YDotSize < 512
XDotSize と YDotSize の比が 64 以下

MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず 4 の倍数にしてください。4 の倍数でない場合，エラー（返り値：0x0e）になります。

(5) crg_SetPoint 関数

【分　類】　文字認識

【関 数 名】　crg_SetPoint

【機能概要】　筆点情報を基に，座標データをセットします。

【形　式】　int　crg_SetPoint (CRGHWDATA HwData,CRGPOINT *Pnt, CRGBUF Buf,
short *Num, CRGMEM *MemInfo);

【引 き 数】

CRGHWDATA　HwData	筆点情報（入力）
CRGPOINT　*Pnt	文字バッファ中にセットされる座標データの先頭アドレス（入力）
CRGBUF　Buf	文字バッファのアドレス情報（入力）
short　*Num	セットされた座標データ数へのアドレス（出力）
CRGMEM　*MemInfo	メモリ情報へのアドレス（入力）

【返 り 値】

0x00	正常
0x0e	与えられたアドレスが４の倍数でない
0x10	ペン状態が不正（1/0 以外）

【機　能】　タブレットからの筆点情報（x 座標，y 座標，ペン状態[注]）を１つ受け取り，ペン状態が１のときは，筆点情報中の座標データを Pnt の位置にセットします。ペン状態が０のときは，crg_EOC をセットします。ただし，２回続けて crg_EOC はセットされません。

注　ペンがタブレットから離れている状態は 0，タブレットに接触している状態は 1

座標データまたは crg_EOC が Pnt にセットされた場合は，*Num=1 を返します。何もセットされなかった場合は，*Num=0 を返します。

Pnt，文字バッファの先頭アドレス（Buf.TopBuf），最終アドレス（Buf.BottomBuf），MemInfo→ReserveArea，MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

Buf.BottomBuf-1 が指している領域まで x 座標と y 座標を格納するので，Pnt の値は Buf.TopBuf,Buf.TopBuf+1,Buf.TopBuf+2,…,Buf.BottomBuf-1 のいずれかです。

ペン状態（HwData.Pen）は，０または１の値をセットします。それ以外の場合，エラー（返り値：0x10）になります。

(6) crg_RecogChar 関数

【分　類】　文字認識

【関 数 名】　crg_RecogChar

【機能概要】　認識処理を行い，認識結果を出力します。

【形　式】　int crg_RecogChar (CRGRECT Grid, CRGCHRPAT ChrPat, unsigned short LastIndex, CRGRESULT *Result, long DicInfoNum, CRGDICSET *DicSelect, CRGMEM *MemInfo);

【引 き 数】

CRGRECT　Grid	現在の文字が書かれている文字枠の左上と右下の座標値（入力）
CRGCHRPAT　ChrPat	1 文字分のデータ（入力）
unsigned short　LastIndex	1 つ前に入力された文字のシフト JIS コード（入力）
CRGRESULT　*Result	認識結果へのアドレス（出力）
Long　DicInfoNum	辞書情報テーブル中の辞書情報の数（入力）
CRGDICSET　*DicSelect	辞書情報テーブルへのアドレス（入力）
CRGMEM　*MemInfo	メモリ情報へのアドレス（入力）

【返 り 値】

0x00	正常
0x04	座標データ数（CharPat.PntNum）が 1 点以下，または crg_SPOINTSIZE より大きい
0x0e	与えられたアドレスが 4 の倍数でない
0x0f	辞書情報テーブル中の辞書情報の数（DicInfoNum）が負
0x11	辞書状態 DicState が不正（1/0 以外）
0x21	認識結果が得られなかった
0x22	処理停止（crg_StopProc()が呼ばれた場合）

【機　能】　文字バッファ中の ChrPat.Chr から ChrPat.PntNum 個の座標データ（crg_EOC を含む）からなる文字データに対して認識を行い，認識結果 Result を出力します。

文字バッファはリング・バッファであり，文字バッファの先頭アドレス（ChrPat.ChrBuf →TopBuf）と最終アドレス（ChrPat.ChrBuf→BottomBuf）を利用して文字データを取り出します。

座標データ数（CharPat.PntNum）が 1 点以下，または crg_SPOINTSIZE より大きい場合は，エラー（返り値：0x04）となります。

1 文字分のデータ(ChrPat)中の各アドレス，MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず 4 の倍数にしてください。4 の倍数でない場合，エラー（返り値：0x0e）になります。

辞書情報テーブル中の辞書情報の数（DicInfoNum）が負の場合，エラー（返り値：0x0f）になります。

crg_RecogChar()は，CRGRESULT 型の先頭アドレス（Result）を引き数として受け取り，*Result に認識結果を書き込みます。*Result の領域の確保は，crg_RecogChar()を呼ぶ側で行ってください。

保守／廃止

認識結果の信頼性が低い場合はワーニングとなり，Result→Warning に 1 がセットされます。

入力文字の座標データ数が極端に少ない，ストローク数が多すぎるまたは少なすぎる，DicInfoNum=0 など，crg_StopProc()が呼ばれた場合などにより認識結果が得られない場合は，距離値（Result→Candidate[ ].Value）と，文字コード（Result→Candidate[ ].Index）のすべてにそれぞれ 0x7FFF と 0x0000 がセットされます。

候補数（Result→CandNo）が crg_MAXCAND 個未満の場合，Result→Candidate[Result→CandNo]～Result→Candidate[crg_MAXCAND-1]の距離値と文字コードにはそれぞれ 0x7FFF と 0x0000 がセットされます。

引き数の DicSelect は，DicInfoNum 個の要素を持つ CRGDICSET 型配列へのアドレスです。１つの要素が，１つの辞書情報を持っています。辞書状態（DicState）を次に示すように指定すると，DicInfoNum 個の複数の辞書から，認識に使用する辞書を自由に指定することができます。

0：認識に使用しない。
1：認識に使用する。

辞書状態（DicState）が上記以外を指定した場合，エラー（返り値：0x11）になります。

辞書バッファ先頭アドレス（DicArea）は，必ず 4 の倍数にしてください。4 の倍数でない場合，エラー（返り値：0x0e）になります。

認識結果の候補中に「凡（967d）」がある場合，その次候補は必ず「風（9597）」になります。

Grid，LastIndex を用いて大文字／小文字などの同型文字の判別を行います。なお，１つ前に入力された文字がない場合は，LastIndex=0 にしてください。

文字枠を用いなかった場合などで，Grid の各パラメータに 0 をセットすると，認識結果に同型文字（**付録 D　同型文字の書き分け方**参照）が存在する場合，認識結果は，ひらがな→カタカナ→漢字→英字→数字→記号の順（ひらがな，カタカナ：大文字→小文字）（漢，英，記：シフト JIS コード順）にセットされます。

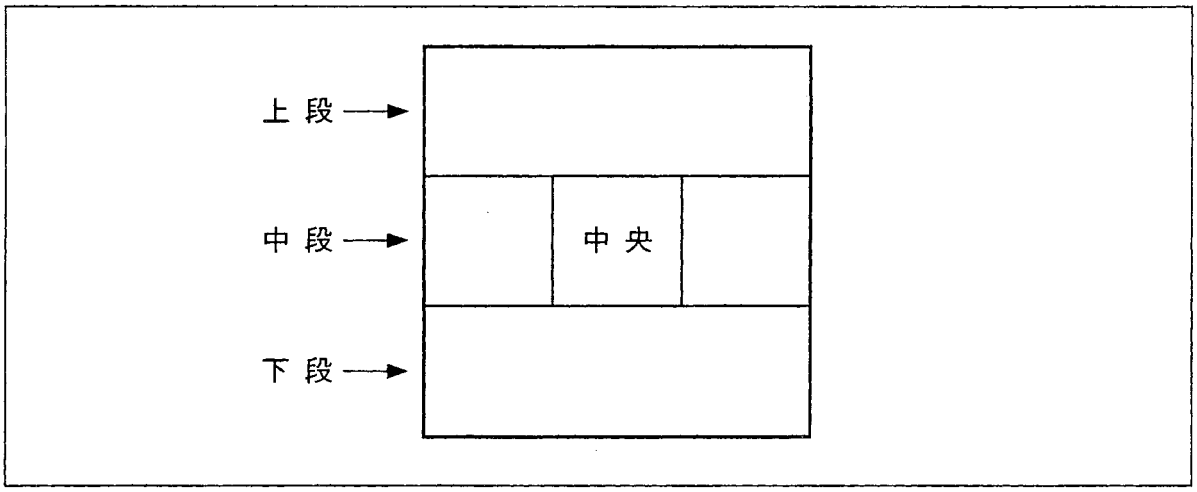
保守／廃止

認識結果の１位候補が次に示す同型文字（**付録Ｄ　同型文字の書き分け方**参照）の場合，Grid と LastIndex の値により，認識結果は次のような順でセットされます。

長音	― (815b)
いち	― (88ea)
ダッシュ	― (817c)
マイナス	- (815d)
上 線	￣ (8150)
下 線	＿ (8151)

なお，文字枠を縦方向に上から上段，中段，下段に３等分し，さらに中段を横方向に３等分し，真中の領域を中央と呼ぶことにします（図 2-4 参照）。

図 2−4　文字枠の領域



大文字／小文字などの同型文字により，認識に使用した辞書に登録されていない字種が認識結果（Result）のなかに表示されることがあります（**付録Ｄ　同型文字の書き分け方**参照）。これを回避するためには，アプリケーション側で認識結果（Result）のなかの不要な文字を消去してください（**付録Ｇ　サンプル・ソース**参照）。

(a) Grid のパラメータ≠0，LastIndex≠0 の場合

○ 上段に書かれた場合　　上線→長音→いち→ダッシュ→マイナス→下線
○ 下段に書かれた場合　　下線→長音→いち→ダッシュ→マイナス→上線
○ 中段に書かれた場合
　中央に書かれた場合　　マイナス→長音→いち→ダッシュ→上線→下線
　中央からはみ出して書かれた場合
　　LastIndex がひらがな／カタカナのコードの場合
　　長音→いち→ダッシュ→マイナス→上線→下線
　　LastIndex が漢字のコードの場合
　　いち→長音→ダッシュ→マイナス→上線→下線
　　LastIndex が上記以外（英数字，記号）の場合
　　ダッシュ→長音→いち→マイナス→上線→下線

(b) Grid のパラメータ≠0，LastIndex＝0 の場合

○ 上段に書かれた場合　　上線→長音→いち→ダッシュ→マイナス→下線
○ 下段に書かれた場合　　下線→長音→いち→ダッシュ→マイナス→上線
○ 中段に書かれた場合
　中央に書かれた場合　　マイナス→長音→いち→ダッシュ→上線→下線
　中央からはみ出して書かれた場合　　長音→いち→ダッシュ→マイナス→上線→下線

(c) Grid のパラメータ＝0，LastIndex≠0 の場合

○ LastIndex がひらがな／カタカナのコードの場合
　長音→いち→ダッシュ→マイナス→上線→下線
○ LastIndex が漢字のコードの場合
　いち→長音→ダッシュ→マイナス→上線→下線
○ LastIndex が数字のコードの場合
　マイナス→長音→いち→ダッシュ→上線→下線
○ LastIndex が上記以外（記号，英字）の場合
　ダッシュ→長音→いち→マイナス→上線→下線

(d) Grid のパラメータ＝0，LastIndex＝0 の場合
　長音→いち→ダッシュ→マイナス→上線→下線

保守／廃止

(7) crg_DicClearDic 関数

【分　類】　辞書登録
【関 数 名】　crg_DicClearDic
【機能概要】　ユーザ辞書をクリアします。
【形　式】　int　crg_DicClearDic (long *DicArea, long DicSize, CRGMEM *MemInfo);
【引 き 数】　long　*DicArea　　辞書バッファ先頭アドレス（入力）
long　DicSize　　辞書バッファ・サイズ（バイト）（入力）
CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）
【返 り 値】　0x00　　正常
0x05　　辞書バッファ・サイズが不足
0x0e　　与えられたアドレスが４の倍数でない
【機　能】　ユーザ辞書をクリアします。
新規にユーザ辞書を作成する際，crg_DicCreate()を呼ぶ直前に crg_DicClearDic()を呼んでください。

DicSize が８バイト未満の場合，エラー（返り値：0x05）になります。

ユーザ辞書の辞書バッファの先頭アドレス（DicArea），MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

(8) crg_DicCreate 関数

【分　類】　辞書登録
【関 数 名】　crg_DicCreate
【機能概要】　登録する字種を決めて，登録の準備を行います。
【形　式】　int　crg_DicCreate (unsigned short UserJIS, long *DicArea,long DicSize,
CRGMEM *MemInfo);
【引 き 数】　unsigned short　UserJIS　　登録する字種のシフト JIS コード（入力）
long　*DicArea　　辞書バッファ先頭アドレス（入力）
long　DicSize　　辞書バッファ・サイズ（バイト）（入力）
CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）
【返 り 値】　0x00　　正常
0x05　　辞書バッファ・サイズが不足
0x06　　辞書バッファ・サイズの値が不正
0x0c　　UserJIS に 0x0000 が与えられた
0x0e　　与えられたアドレスが４の倍数でない

保守／廃止

【 機　能 】　登録する字種を決め，登録の準備を行います。１字種の登録を開始するごとに一度呼ぶ必要があります。

１字種の登録は，最初に crg_DicCreate()を呼び，続けて複数回の crg_DicEntryChar()を呼んだあと，crg_DicRegister()を呼ぶことで行います。crg_DicCreate()から crg_DicRegister()の間は，crg_DicEraseChar()を呼ぶことはできません。呼んだ場合，crg_DicEraseChar()でエラー（返り値：0x0b）を返します。

登録するユーザ辞書の辞書バッファの先頭アドレスとサイズを DicArea と DicSize で与えます。認識時にユーザ辞書を使用する場合は，crg_DicCreate()に与えた DicArea のアドレスを crg_RecogChar()に与えてください。

すでに存在するユーザ辞書の辞書バッファ・サイズ（DicSize）を再設定することができます。たとえば，ユーザ辞書に何文字か登録して辞書バッファ・サイズが不足してきた場合，crg_DicCreate の引き数（DicSize）にさらに大きな値をセットすることで，辞書バッファ・サイズの再設定ができます。

登録するユーザ辞書の辞書バッファ・サイズが不足している場合は，エラー（返り値：0x05）になります。

あるユーザ辞書に字種を追加登録するとき，そのユーザ辞書に登録済みの全文字のメモリ・サイズより小さいサイズを辞書バッファ・サイズとして与えた場合，エラー（返り値：0x06）になります。

UserJIS に 0x0000 を与えた場合は，エラー（返り値：0x0c）になります。

ユーザ辞書の辞書バッファの先頭アドレス（DicArea），MemInfo→ReserveArea と MemInfo →WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

(9) crg_DicEntryChar 関数

【分　類】　辞書登録

【関 数 名】　crg_DicEntryChar

【機能概要】　学習に必要な特徴パラメータを抽出し，累積していきます。

【形　式】　int　crg_DicEntryChar (CRGCHRPAT ChrPat, CRGMEM *MemInfo);

【引 き 数】　CRGCHRPAT　ChrPat　　文字分のデータ（入力）

CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）

【返 り 値】　0x00　正常

0x03　特徴パラメータが抽出されなかった

0x04　座標データ数（CharPat.PntNum）が 1 点以下，または crg_SPOINTSIZE より大きい

0x07　255 回以上連続して crg_DicEntryChar()を呼んだ

0x0b　コマンドのシーケンシャル・エラー

0x0d　入力文字の画数が 31 画以上である

0x0e　与えられたアドレスが 4 の倍数でない

0x22　処理停止（crg_StopProc()が呼ばれた場合）

【機　能】　文字バッファ中の ChrPat.Chr から ChrPat.PntNum 個の座標データからなる文字データ（crg_EOC を含む）から学習に必要な特徴パラメータを抽出し，累積していきます。

文字バッファは，文字認識時に使用するものと同じ形式のものを使用します。

登録用文字を 1 つ入力するごとに，crg_DicEntryChar()を呼びます。

1 字種の登録において，複数回入力する（複数回 crg_DicEntryChar()を呼ぶ）ことにより，認識性能の向上が期待できます。ただし，1 字種の登録において，crg_DicEntryChar()を 255 回以上連続して呼ぶとエラー（返り値：0x07）になります。

1 文字分のデータ(ChrPat)中の各アドレス，MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず 4 の倍数にしてください。4 の倍数でない場合，エラー（返り値：0x0e）になります。

座標データ数（CharPat.PntNum）が 1 点以下，または crg_SPOINTSIZE より大きい場合はエラー（返り値：0x04）になります。

1 字種の登録の始めに crg_DicCreate()を呼ぶ必要があります。呼ばなかった場合，エラー（返り値：0x0b）になります。1 字種の登録で，crg_DicEntryChar()を複数回呼ぶ場合は，そのたびに crg_DicCreate()を呼ばないでください。

1 字種の登録に使用する文字の字形は，同じものとします。たとえば，アルファベット「b」の筆記体とブロック体を同一の字種登録で入力しないでください。それぞれ個別に字種登録をしてください。

（10）crg_DicRegister 関数

【 分　類 】　辞書登録

【関 数 名】　crg_DicRegister

【機能概要】　辞書データを作成して，ユーザ辞書に登録します。

【 形　式 】　int　crg_DicRegister (short *DicNum, CRGMEM *MemInfo);

【引 き 数】　short　*DicNum　　ユーザ辞書に登録されている文字数へのアドレス（出力）

CRGMEM　*MemInfo　　メモリ情報へのアドレス（入力）

【返 り 値】　0x00　正常

0x09　特徴パラメータが累積されていない

0x0b　コマンドのシーケンシャル・エラー

0x0e　与えられたアドレスが４の倍数でない

【 機　能 】　登録用文字入力（crg_DicEntryChar()）で累積された特徴パラメータを用いて辞書データを作成して，crg_DicCreate()の引き数である文字コード（UserJIS）とともに DicArea で示されている辞書バッファ領域にユーザ辞書を登録します。ただし，特徴パラメータや文字コード（UserJIS）は，crg_DicRegister()を呼ぶ直前のものだけ有効になります。つまり，crg_DicCreate()→crg_DicEntryChar()→crg_DicCreate()→crg_DicEntryChar()→crg_DicEntryChar()→crg_DicRegister()の順でライブラリ関数を呼んだ場合，1,2 番目の crg_DicCreate()，crg_DicEntryChar()は無視され，3 番目の crg_DicCreate()から有効になります。

ユーザ辞書に登録されている文字数を*DicNum に返します。

登録された文字は，すでにユーザ辞書に登録されている文字の最後に追加され，ユーザ辞書に登録されている文字数が*DicNum に出力されます。

1 字種の登録の最後に crg_DicRegister()を呼ぶので，crg_DicCreate()が呼ばれていない場合は，エラー（返り値：0x0b）になります。

crg_DicEntryChar()が呼ばれていないなど，特徴パラメータが累積されていないために登録ができない場合は，エラー（返り値：0x09）になります。

MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

保守／廃止

（11）crg_DicEraseChar 関数

【分　類】　辞書登録

【関 数 名】　crg_DicEraseChar

【機能概要】　ユーザ辞書から指定された文字を消去します。

【形　式】　int　crg_DicEraseChar (short DicPos, short *DicNum, long*DicArea,
CRGMEM *MemInfo);

【引 き 数】

short	DicPos	ユーザ辞書中での消去文字の位置（入力）
short	*DicNum	ユーザ辞書に登録されている文字数（出力）
long	*DicArea	辞書バッファ先頭アドレス（入力）
CRGMEM	*MemInfo	メモリ情報へのアドレス（入力）

【返 り 値】

0x00	正常
0x0a	消去文字の指定位置が正しくない
0x0b	コマンドのシーケンシャル・エラー
0x0e	与えられたアドレスが４の倍数でない

【機　能】　DicArea が示すユーザ辞書中の DicPos 番目の文字を消去します。

DicPos は，ユーザ辞書に登録された文字の順番を示し，１から始まる値です。たとえば，最初に登録された文字は１となります。DicPos の値は，ユーザ辞書に登録されている文字数までです。それ以外の値を入力した場合は，エラー（返り値：0x0a）になります。

文字消去後のユーザ辞書に登録されている文字数が*DicNum に出力され，残ったユーザ辞書内の文字はメモリ上で詰め合わされます。消去された文字以降に登録された文字の順番も繰り上がります。

１つのシフト JIS コードに形の違う文字を複数登録することができます。登録文字消去では，シフト JIS コードによらずに，登録された順番のみにより消去の対象を指定します。

登録中（crg_DicCreate()から crg_DicRegister()）に crg_DicEraseChar()を呼ぶとエラー（返り値：0x0b）になります。その場合，crg_DicEraseChar()でエラーを返しますが登録は引き続き行えます。

ユーザ辞書の辞書バッファの先頭アドレス（DicArea），MemInfo→ReserveArea と MemInfo→WorkArea の各アドレスは，必ず４の倍数にしてください。４の倍数でない場合，エラー（返り値：0x0e）になります。

保守／廃止

（12）crg_DicInformation 関数

【 分　類 】　辞書登録

【関 数 名】　crg_DicInformation

【機能概要】　ユーザ辞書に登録されている文字数とメモリ・サイズを返します。

【 形　式 】　int　crg_DicInformation (short *DicNum, long *UsedDicMemSize, long *DicArea);

【引 き 数】

short	*DicNum	ユーザ辞書に登録されている文字数（出力）
long	*UsedDicMemSize	ユーザ辞書に登録された全文字のメモリ・サイズ（バイト）（出力）
long	*DicArea	辞書バッファ先頭アドレス（入力）

【返 り 値】　0x00　　正常

0x0e　　与えられたアドレスが４の倍数でない

【 機　能 】　ユーザ辞書に登録されている文字数を*DicNum に返します。何も登録されていない場合は，*DicNum = 0 になります。

DicArea が示すユーザ辞書に登録された全文字のメモリ・サイズを，*UsedDicMemSize に返します。全文字のメモリ・サイズは，crg_DicCreate()の引き数である DicSize のうち，実際に辞書に使われているメモリ・サイズです。何も登録されていない場合は，*UsedDicMemSize= 0 になります。

ユーザ辞書の辞書バッファの先頭アドレス（DicArea）は，必ず４の倍数にしてください。4 の倍数でない場合，エラー（返り値：0x0e）になります。

保守／廃止

〔メ　モ〕

保守／廃止

第３章　インストレーション

3.1　提供形態

手書き文字認識ミドルウエアでは，NEC 製または GHS 社製ツールを使用してアプリケーションを開発するためのライブラリを２種類（NEC 製ツール用と GHS 社製ツール用）提供しています。

パッケージの内容について図 3-1，図 3-2 に示します。

★　図 3−1　パッケージの内容（V810 ファミリ）

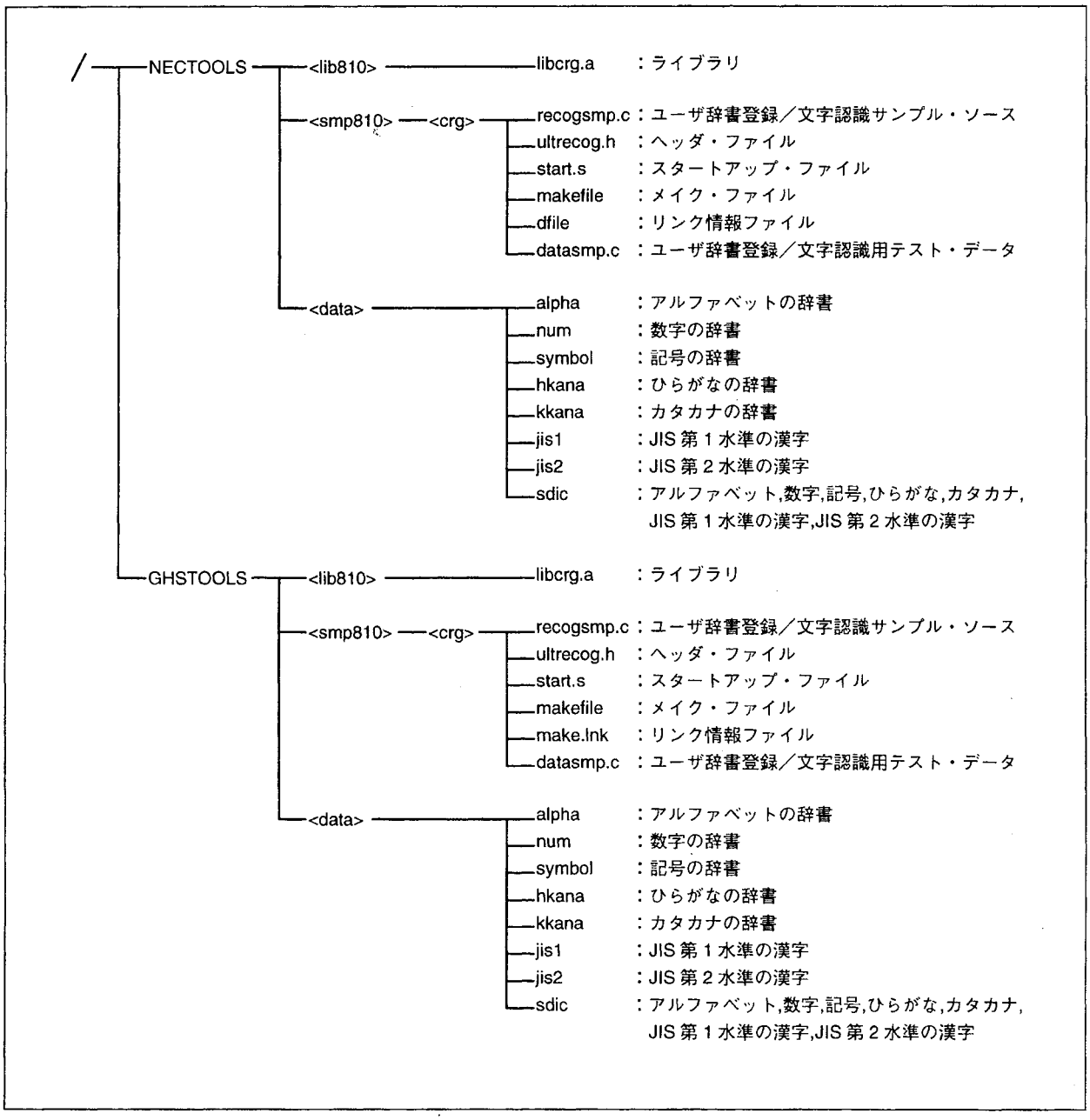
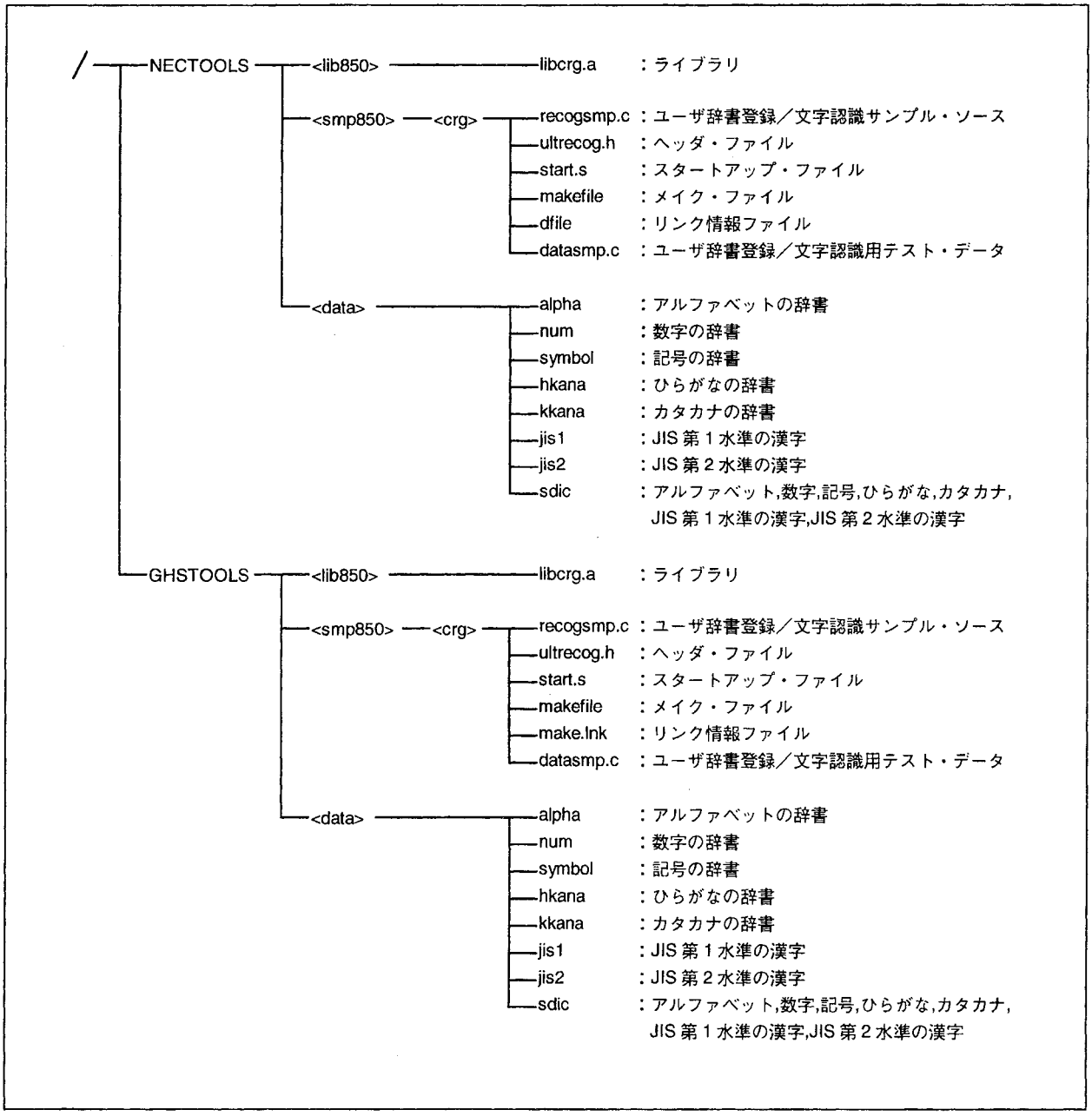


図 3−2　パッケージの内容（V850 ファミリ）



保守／廃止

3.2　リンク手順

このライブラリ内で使用しているセクション名を示します。

★

表3−1　セクション名（V810ファミリ）

セクション名	属　性	機　　能
.crgtext	text	文字認識／辞書登録プログラム
.crgdata	data	定数データ

表3−2　セクション名（V850ファミリ）

セクション名	属　性	機　　能
.crgtexti	text	内部ROMへの配置を推奨する文字認識／辞書登録プログラム
.crgtext	text	.crgtexti以外の文字認識／辞書登録プログラム
.crgdata	data	定数データ

次にリンク手順を示します。

(1) NEC版

★

(a) V810ファミリの場合（CA732　Ver.1.10以上）

ld732 -D ＜リンク情報＞ -cpu ＜devicename[注1]＞ ＜オブジェクト＞ libcrg.a -lc -o ＜出力ファイル＞

(b) V850ファミリの場合（CA850　Ver.1.10以上）

ld850 -D ＜リンク情報＞ -cpu ＜devicename[注2]＞ ＜オブジェクト＞ libcrg.a -lc -o ＜出力ファイル＞

注1. devicenameの設定例を次に示します。

V810の場合：732　　V821の場合　　：741

2. devicenameの設定例を次に示します。

V852の場合：3002　　V850/SA1の場合　：3015

V853の場合：3003　　V850E/MS1の場合：3100

V854の場合：3008

(2) GHS版（GHS　Ver.1.8.8 Release 3.0.1以上）

lx -o ＜出力ファイル＞ -sec ＜リンク情報＞ ＜オブジェクト＞ -lcrg -larch

保守／廃止

3.3　シンボル名規約

このライブラリ内のグローバル・シンボルは，表3-3に示す規約に従って命名されています。ユーザ・アプリケーション内のシンボル名と重複しないように注意してください。

表3−3　シンボル名規約

分　類	規　約
関　数	先頭に"crg_"を付加しています。
変　数	
定　数	
タグ名	先頭に"CRG"を付加しています。

3.4　ホスト・マシンへのファイル展開

提供媒体からホスト・マシン上にファイル群を転送する手順を，UNIX版（SUN4™）とWindows版に分けて説明します。

3.4.1　UNIX版

UNIX版の提供媒体は，CGMTと3.5インチ・フロッピィ・ディスク（2枚）の2種類あります。この2種類の媒体には，tar形式でファイル群を格納しています。ホスト・マシンへのインストール手順を次に示します。

① 手書き文字認識ライブラリをインストールするためのディレクトリを作成します。
ここでは，mw_crgという名前のディレクトリを作成します。

```
% mkdir mw_crg <CR>
```

② 作成したディレクトリに移動します。

```
% cd mw_crg <CR>
```

③ 提供媒体を磁気テープ装置にセットします。
CGMTの場合：磁気テープ装置にセットします。
フロッピィ・ディスクの場合：フロッピィ・ディスク装置にセットします。フロッピィ・ディスクは，NEC製ツール用とGHS社製ツール用の2枚あります。
どちらか必要な方のフロッピィ・ディスクをセットしてください。

④ tar コマンドを実行して，ファイル群をディスク上に展開します。なお，ホスト・マシンにより指定するスペシャル・ファイル名は異なります。

(a) CGMT の場合

ここでは/dev/rst8 であるとして，NEC 製ツール用と GHS 社製ツール用のファイル群を展開する場合をそれぞれ次に示します。

NEC 製ツール用

```
% tar -xvof /dev/rst8 nectools <CR>
```

GHS 社製ツール用

```
% tar -xvof /dev/rst8 ghstools <CR>
```

(b) フロッピィ・ディスクの場合

ここでは/dev/rfd0c であるとして，NEC 製ツール用と GHS 社製ツール用のファイル群を展開する場合をそれぞれ次に示します。

NEC 製ツール用

```
% tar -xvof /dev/rfd0c nectools <CR>
```

GHS 社製ツール用

```
% tar -xvof /dev/rfd0c ghstools <CR>
```

⑤ ファイルがインストールされたことを確認します。

```
% ls -CFR <CR>
```

3.4.2　Windows 版

Windows 版の提供媒体は，フロッピィ・ディスク（3.5 インチ）で提供しています。ホスト・マシンへのインストールの手順を次に示します。

① MS-DOS™ プロンプトを起動します。

② 手書き文字認識ライブラリをインストールするためのディレクトリを作成します。ここでは A ドライブに mw_crg という名前のディレクトリを作成します。

```
A:¥> md mw_crg <CR>
```

③ 作成したディレクトリに移動します。

```
A:¥> cd mw_crg <CR>
```

④ 提供媒体をフロッピィ・ディスク装置にセットします。
フロッピィ・ディスクは，NEC 製ツール用と GHS 社製ツール用の 2 枚あります。どちらか必要な方のフロッピィ・ディスクをセットしてください。
ここではフロッピィ・ディスク・ドライブは C ドライブとしています。

⑤ ファイル群を展開します。
NEC 製ツール用と GHS 社製ツール用のファイル群を展開する場合をそれぞれ次に示します。

(a) NEC 製ツール用の場合

（ i ）インストールするディレクトリを作成します（ここでは nectools という名前のディレクトリを作成）。

```
A:¥mw_crg> md nectools
```

（ ii ）作成したディレクトリに移動します。

```
A:¥mw_crg> cd nectools
```

（ iii ）xcopy コマンドを実行して，ファイル群を展開します。

```
A:¥mw_crg¥nectools> xcopy c:¥nectools . /s /e /v <CR>
```

(b) GHS 社製ツール用の場合

（ i ）インストールするディレクトリを作成します（ここでは ghstools という名前のディレクトリを作成）。

```
A:¥mw_crg> md ghstools
```

（ ii ）作成したディレクトリに移動します。

```
A:¥mw_crg> cd ghstools
```

（ iii ）xcopy コマンドを実行して，ファイル群を展開します。

```
A:¥mw_crg¥ghstools> xcopy c:¥ghstools . /s /e /v <CR>
```

⑥ ファイルがインストールされたことを確認します。

NEC 製ツールの場合

```
A:¥mw_crg¥nectools> dir /s /w <CR>
```

GHS 社製ツールの場合

```
A:¥mw_crg¥ghstools> dir /s /w <CR>
```

★3.5　サンプル・プログラムの作成（V810 ファミリ）

smp810 ディレクトリには，ユーザ辞書登録および文字認識を行うサンプル・プログラムを格納しています。サンプル・プログラムの内容については，第４章　システム例を参照してください。

3.5.1　UNIX 版（NEC 製ツール用）

NEC 製ツールを使用して，UNIX 版のサンプル・プログラムを make する例を示します。

① サンプル・プログラムを格納しているディレクトリに移動します。

```
% cd mw_crg/nectools/smp810/crg <CR>
```

② エディタを使用して，makefile の TOOLDIR にコンパイラのあるディレクトリを記述します。

③ make コマンドを実行して，recogsmp.out を生成します。

```
% make <CR>
```

④ インサーキット・エミュレータなどを使用して，recogsmp.out をダウンロードして実行します。

3.5.2　UNIX 版（GHS 社製ツール用）

GHS 社製ツールを使用して，UNIX 版のサンプル・プログラムを make する例を示します。

① サンプル・プログラムを格納しているディレクトリに移動します。

```
% cd mw_crg/ghstools/smp810/crg <CR>
```

② エディタを使用して，makefile の ROOTDIR にコンパイラのあるディレクトリを記述します。

③ make コマンドを実行して，recogsmp.elf を生成します。

```
% make <CR>
```

④ インサーキット・エミュレータなどを使用して，recogsmp.elf をダウンロードして実行します。

3.5.3　Windows 版（NEC 製ツール用，VSH 使用時）

NEC 製ツールを使用して，Windows 版のサンプル・プログラムを make する例を示します。

① VSH をダブル・クリックして，起動します。

② サンプル・プログラムを格納しているディレクトリに移動します。

```
A:¥> cd a:¥mw_crg¥nectools¥smp810¥crg <CR>
```

③ エディタを使用して，makefile の TOOLDIR にコンパイラのあるディレクトリを指定します。

④ vmake.exe を実行して，recogsmp.out を生成します。

```
A:¥mw_crg¥nectools¥smp810¥crg> vmake <CR>
```

⑤ インサーキット・エミュレータなどを使用して，recogsmp.out をダウンロードして実行します。

3.5.4　Windows 版（GHS 社製ツール用）

GHS 社製ツールを使用して，Windows 版のサンプル・プログラムを make する例を示します。

① make.exe（GNU の make プログラム）を用意します。

② サンプル・プログラムを格納しているディレクトリに移動します。

```
A:¥> cd a:¥mw_crg¥ghstools¥smp810¥crg <CR>
```

③ エディタを使用して，makefile の ROOTDIR にコンパイラのあるディレクトリを指定します。

④ make.exe を実行して，recogsmp.elf を生成します。

```
A:¥mw_crg¥ghstools¥smp810¥crg> make <CR>
```

⑤ インサーキット・エミュレータなどを使用して，recogsmp.elf をダウンロードして実行します。

3.6　サンプル・プログラムの作成（V850 ファミリ）

smp850 ディレクトリには，ユーザ辞書登録および文字認識を行うサンプル・プログラムを格納しています。サンプル・プログラムの内容については，**第４章　システム例**を参照してください。

3.6.1　UNIX 版（NEC 製ツール用）

NEC 製ツールを使用して，UNIX 版のサンプル・プログラムを make する例を示します。

① サンプル・プログラムを格納しているディレクトリに移動します。

```
% cd mw_crg/nectools/smp850/crg <CR>
```

② エディタを使用して，makefile の TOOLDIR にコンパイラのあるディレクトリを記述します。

③ make コマンドを実行して，recogsmp.out を生成します。

```
% make <CR>
```

④ インサーキット・エミュレータなどを使用して，recogsmp.out をダウンロードして実行します。

3.6.2　UNIX 版（GHS 社製ツール用）

GHS 社製ツールを使用して，UNIX 版のサンプル・プログラムを make する例を示します。

① サンプル・プログラムを格納しているディレクトリに移動します。

```
% cd mw_crg/ghstools/smp850/crg <CR>
```

② エディタを使用して，makefile の ROOTDIR にコンパイラのあるディレクトリを記述します。

③ make コマンドを実行して，recogsmp.elf を生成します。

```
% make <CR>
```

④ インサーキット・エミュレータなどを使用して，recogsmp.elf をダウンロードして実行します。

3. 6. 3　Windows 版（NEC 製ツール用，VSH 使用時）

NEC 製ツールを使用して，Windows 版のサンプル・プログラムを make する例を示します。

① VSH をダブル・クリックして，起動します。

② サンプル・プログラムを格納しているディレクトリに移動します。

```
A:¥> cd a:¥mw_crg¥nectools¥smp850¥crg <CR>
```

③ エディタを使用して，makefile の TOOLDIR にコンパイラのあるディレクトリを指定します。

④ vmake.exe を実行して，recogsmp.out を生成します。

```
A:¥mw_crg¥nectools¥smp850¥crg> vmake <CR>
```

⑤ インサーキット・エミュレータなどを使用して，recogsmp.out をダウンロードして実行します。

3. 6. 4　Windows 版（GHS 社製ツール用）

GHS 社製ツールを使用して，Windows 版のサンプル・プログラムを make する例を示します。

① make.exe（GNU の make プログラム）を用意します。

② サンプル・プログラムを格納しているディレクトリに移動します。

```
A:¥> cd a:¥mw_crg¥ghstools¥smp850¥crg <CR>
```

③ エディタを使用して，makefile の ROOTDIR にコンパイラのあるディレクトリを指定します。

④ make.exe を実行して，recogsmp.elf を生成します。

```
A:¥mw_crg¥ghstools¥smp850¥crg> make <CR>
```

⑤ インサーキット・エミュレータなどを使用して，recogsmp.elf をダウンロードして実行します。

第４章　システム例

この章ではユーザ辞書登録および文字認識のシステム例を示します。システム例のメイン・ソースについては，**付録 G　サンプル・ソース**を参照してください。

システム例は，図 4-1 に示す手順でユーザ辞書登録および文字認識を行います。

(1) メモリ情報セット

ReserveArea，WorkArea などの情報をセットします。

(2) 初期化

crg_Initialize によりシステムを初期化します。

(3) ユーザ辞書初期化

新規にユーザ辞書を作成するため，crg_DicClearDic を呼びます。

(4) 初期設定

バッファや，認識に必要なパラメータを準備します。

(5) 文字枠セット

認識に用いる文字枠の大きさの情報をセットするため，crg_SetGridParam を呼びます。

(6) ユーザ辞書登録

次の(a)～(d)を 2 回繰り返して，配列 UserData の 2 文字分の文字データをユーザ辞書に登録します。

(a) 登録開始

登録に必要なパラメータを設定するため，crg_DicCreate を呼びます。

(b) 筆点単位処理

配列 UserData に格納されている 1 文字分の文字データの座標点ごとに，筆点単位処理（crg_SetPoint）を行います。

(c) 登録用文字入力

筆点単位処理後の文字データに対して，学習に必要なパラメータを抽出し，累積していきます（crg_DicEntryChar）。

(d) 文字登録

crg_DicRegister を呼んでユーザ辞書に登録します。

(7) 辞書情報セット

使用する複数の辞書に関する情報をセットします。

（８）文字認識

配列 RecogData の文字データに対して認識を行います。

(a) 初期設定

認識に必要なパラメータを準備します。

(b) 筆点単位処理

配列 RecogData に格納されている文字データの座標点ごとに筆点単位処理（crg_SetPoint）を行います。

(c) 文字認識

crg_RecogChar により文字認識を行います。

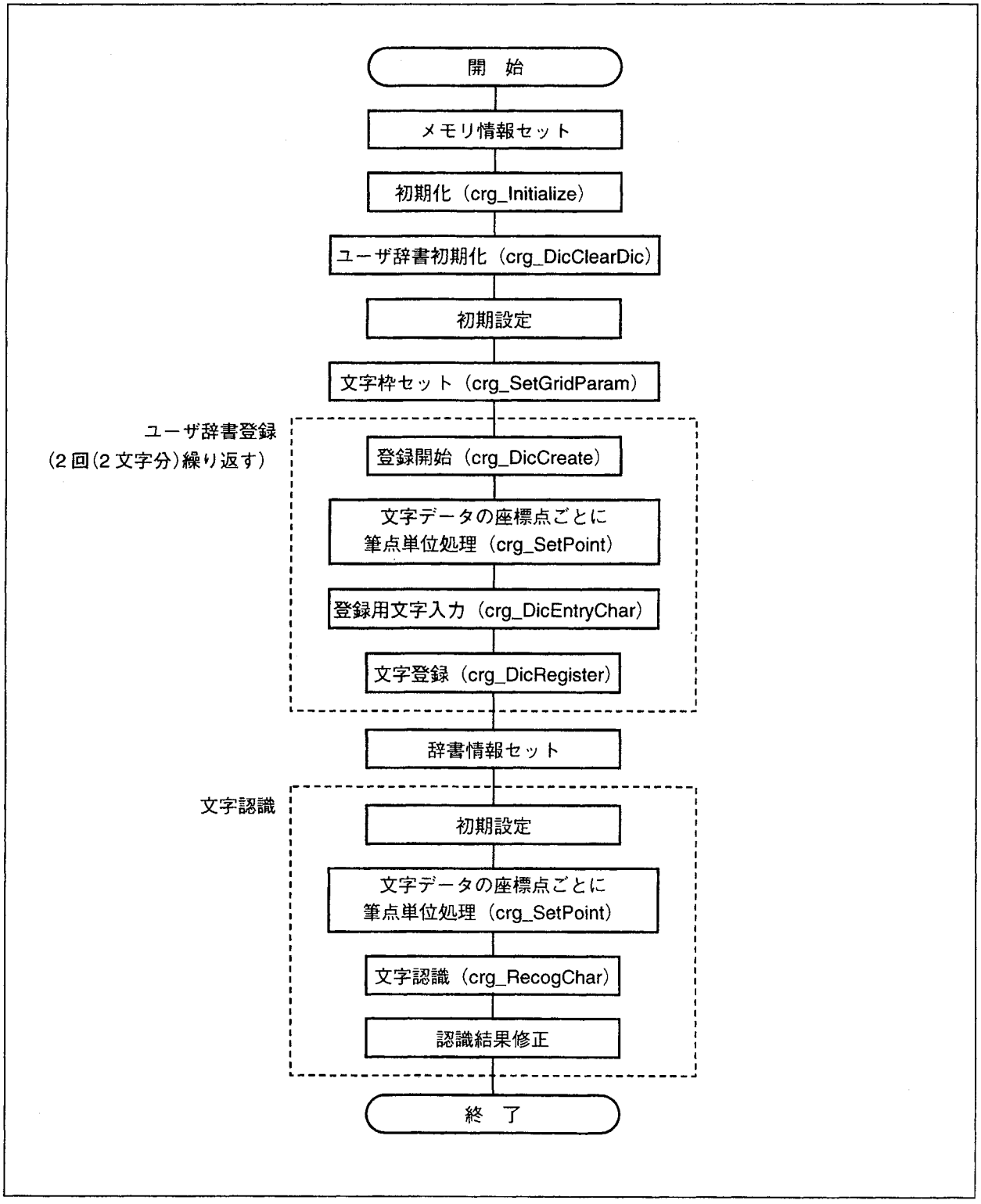
(d) 認識結果修正

認識結果のなかの不要な文字を消去します。

(9) 終　了

crg_Uninitialize を呼んでシステムを終了します。

図 4−1　システム例のフロー・チャート



〔メ　モ〕

保守／廃止

付録A　JIS第1水準漢字

亜唖娃阿哀愛挨姶逢葵茜穐悪握渥旭葦芦鯵梓圧斡扱宛姐虻飴絢綾鮎或粟袷安庵按暗案闇鞍杏以伊位依偉囲夷委威尉惟意慰易椅為畏異移維緯胃萎衣謂違遺医井亥域育郁磯一壱溢逸稲茨芋鰯允印咽員因姻引飲淫胤蔭院陰隠韻吋右宇烏羽迂雨卯鵜窺丑碓臼渦嘘唄欝蔚鰻姥厩浦瓜閏噂云運雲荏餌叡営嬰影映曳栄永泳洩瑛盈穎頴英衛詠鋭液疫益駅悦謁越閲榎厭円園堰奄宴延怨掩援沿演炎焔煙燕猿縁艶苑薗遠鉛鴛塩於汚甥凹央奥往応押旺横欧殴王翁襖鴬鴎黄岡沖荻億屋憶臆桶牡乙俺卸恩温穏音下化仮何伽価佳加可嘉夏嫁家寡科暇果架歌河火珂禍禾稼箇花苛茄荷華菓蝦課嘩貨迦過霞蚊俄峨我牙画臥芽蛾賀雅餓駕介会解回塊壊廻快怪悔恢懐戒拐改魁晦械海灰界皆絵芥蟹開階貝凱劾外咳害崖慨概涯碍蓋街該鎧骸浬馨蛙垣柿蛎鈎劃嚇各廓拡撹格核殻獲確穫覚角赫較郭閣隔革学岳楽額顎掛笠樫橿梶鰍潟割喝恰括活渇滑葛褐轄且鰹叶椛樺鞄株兜竃蒲釜鎌噛鴨栢茅萱粥刈苅瓦乾侃冠寒刊勘勧巻喚堪姦完官寛干幹患感慣憾換敢柑桓棺款歓汗漢澗潅環甘監看竿管簡緩缶翰肝艦莞観諌貫還鑑間閑関陥韓館舘丸含岸巌玩癌眼岩翫贋雁頑顔願企伎危喜器基奇嬉寄岐希幾忌揮机旗既期棋棄機帰毅気汽畿祈季稀紀徽規記貴起軌輝飢騎鬼亀偽儀妓宜戯技擬欺犠疑祇義蟻誼議掬菊鞠吉吃喫桔橘詰砧杵黍却客脚虐逆丘久仇休及吸宮弓急救朽求汲泣灸球究窮笈級糾給旧牛去居巨拒拠挙渠虚許距鋸漁禦魚亨享京供侠僑兇競共凶協匡卿叫喬境峡強彊怯恐恭挟教橋況狂狭矯胸脅興蕎郷鏡響饗驚仰凝尭暁業局曲極玉桐粁僅勤均巾錦斤欣欽琴禁禽筋緊芹菌衿襟謹近金吟銀九倶句区狗玖矩苦躯駆駈駒具愚虞喰空偶寓遇隅串櫛釧屑屈掘窟沓靴轡窪熊隈粂栗繰桑鍬勲君薫訓群軍郡卦袈祁係傾刑兄啓圭珪型契形径恵慶慧憩掲携敬景桂渓畦稽系経継繋罫茎荊蛍計詣警軽頚鶏芸迎鯨劇戟撃激隙桁傑欠決潔穴結血訣月件倹倦健兼券剣喧圏堅嫌建憲懸拳捲検権牽犬献研硯絹県肩見謙賢軒遣鍵険顕験鹸元原厳幻弦減源玄現絃舷言諺限乎個古呼固姑孤己庫弧戸故枯湖狐糊袴股胡菰虎誇跨鈷雇顧鼓五互伍午呉吾娯後御悟梧檎瑚碁語誤護醐乞鯉交佼侯候倖光公功効勾厚口向后喉坑垢好孔孝宏工巧巷幸広庚康弘恒慌抗拘控攻昂晃更杭校梗構江洪浩港溝甲皇硬稿糠紅紘絞綱耕考肯肱腔膏航荒行衡講貢購郊酵鉱砿鋼閤降項香高鴻剛劫号合壕拷濠豪轟麹克刻告国穀酷鵠黒獄漉腰甑忽惚骨狛込此頃今困坤墾婚恨懇昏昆根梱混痕紺艮魂些佐叉唆嵯左差査沙瑳砂詐鎖裟坐座挫債催再最哉塞妻宰彩才採栽歳済災采犀砕砦祭斎細菜裁載際剤在材罪財冴坂阪堺榊肴咲崎埼碕鷺作削咋搾昨朔柵窄策索錯桜鮭笹匙冊刷察拶撮擦札殺薩雑皐鯖捌錆鮫皿晒三傘参山惨撒散桟燦珊産算纂蚕讃賛酸餐斬暫残仕仔伺使刺司史嗣四士始姉姿子屍市師志思指支孜斯施旨枝止死氏獅祉私糸紙紫肢脂至視詞詩試誌諮資賜雌飼歯事似侍児字寺慈持時次滋治爾璽痔磁示而耳自蒔辞汐鹿式識鴫竺軸宍雫七叱執失嫉室悉湿漆疾質実蔀篠偲柴芝屡蕊縞舎写射捨赦斜煮社紗者謝車遮蛇邪借勺尺杓灼爵酌釈錫若寂弱惹主取守手朱殊狩珠種腫趣酒首儒受呪寿授樹綬需囚収周宗就州修愁拾洲秀秋終繍習臭舟蒐衆襲讐蹴輯週酋酬集醜什住充十従戎柔汁渋獣縦重銃叔夙宿淑祝縮粛塾熟出術述俊峻春瞬竣舜駿准循旬楯殉淳準潤盾純巡遵醇順処初所暑曙渚庶緒署書薯藷諸助叙女序徐恕鋤除傷償勝匠升召哨商唱嘗奨妾娼宵将小少尚庄床廠彰承抄招掌捷昇昌昭晶松梢樟樵沼消渉湘焼焦照症省硝礁祥称章笑粧紹肖菖蒋蕉衝裳訟証詔詳象賞醤鉦鍾鐘障鞘上丈丞乗冗剰城場壌嬢常情擾条杖浄状畳穣蒸譲醸錠嘱埴飾拭植殖燭織職色触食蝕辱尻伸信侵唇娠寝審心慎振新晋森榛浸深申疹真神秦紳臣芯薪親診身辛進針震人仁刃塵壬尋甚尽腎訊迅陣靭笥諏須酢図厨逗吹垂帥推水炊睡粋翠衰遂酔錐錘随瑞髄崇嵩数枢趨雛据杉椙菅頗雀裾澄摺寸世瀬畝是凄制勢姓征性成政整星晴棲栖正清牲生盛精聖声製西誠誓請逝醒青静斉税脆隻席惜戚斥昔析石積籍績脊責赤跡蹟碩切拙接摂折設窃節説雪絶舌蝉仙先千占宣専尖川戦扇撰栓栴泉浅洗染潜煎煽旋穿箭線繊羨腺舛船薦詮賤践選遷銭銑閃鮮前善漸然全禅繕膳糎噌塑岨措曾曽楚狙疏疎礎祖租粗素組蘇訴阻遡鼠僧創双叢倉喪壮奏爽宋層匝惣想捜掃挿掻操早曹巣槍槽漕燥争痩相窓糟総綜聡

保守／廃止

草荘葬蒼藻装走送遭鎗霜騒像増憎臓蔵贈造促側則即息捉束測足速俗属賊族続卒袖其揃存孫尊損村遜他多太汰詑 唾堕妥惰打柁舵楕陀駄騨体堆対耐岱帯待怠態戴替泰滞胎腿苔袋貸退逮隊黛鯛代台大第醍題鷹滝瀧卓啄宅托択拓沢濯琢託鐸濁諾茸凧蛸只叩但達辰奪脱巽竪辿棚谷狸鱈樽誰丹単嘆坦担探旦歎淡湛炭短端箪綻耽胆蛋誕鍛団壇弾断暖檀段男談値知地弛恥智池痴稚置致蜘遅馳築畜竹筑蓄逐秩窒茶嫡着中仲宙忠抽昼柱注虫衷註酎鋳駐樗瀦猪苧著貯丁兆凋喋寵帖帳庁弔張彫徴懲挑暢朝潮牒町眺聴脹腸蝶調諜超跳銚長頂鳥勅捗直朕沈珍賃鎮陳津墜椎槌追鎚痛通塚栂掴槻佃漬柘辻蔦綴鍔椿潰坪壷嬬紬爪吊釣鶴亭低停偵剃貞呈堤定帝底庭廷弟悌抵挺提梯汀碇禎程締艇訂諦蹄逓邸鄭釘鼎泥摘擢敵滴的笛適鏑溺哲徹撤轍迭鉄典填天展店添纏甜貼転顛点伝殿澱田電兎吐堵塗妬屠徒斗杜渡登菟賭途都鍍砥砺努度土奴怒倒党冬凍刀唐塔塘套宕島嶋悼投搭東桃梼棟盗淘湯涛灯燈当痘祷等答筒糖統到董蕩藤討謄豆踏逃透鐙陶頭騰闘働動同堂導憧撞洞瞳童胴萄道銅峠鴇匿得徳涜特督禿篤毒独読栃橡凸突椴届鳶苫寅酉瀞噸屯惇敦沌豚遁頓呑曇鈍奈那内乍凪薙謎灘捺鍋楢馴縄畷南楠軟難汝二尼弐迩匂賑肉虹廿日乳入如尿韮任妊忍認濡禰祢寧葱猫熱年念捻撚燃粘乃廼之埜嚢悩濃納能脳膿農覗蚕巴把播覇杷波派琶破婆罵芭馬俳廃拝排敗杯盃牌背肺輩配倍培媒梅楳煤狽買売賠陪這蝿秤矧萩伯剥博拍柏泊白箔粕舶薄迫曝漠爆縛莫駁麦函箱硲箸肇筈櫨幡肌畑畠八鉢溌発醗髪伐罰抜筏閥鳩噺塙蛤隼伴判半反叛帆搬斑板氾汎版犯班畔繁般藩販範釆煩頒飯挽晩番盤磐蕃蛮匪卑否妃庇彼悲扉批披斐比泌疲皮碑秘緋罷肥被誹費避非飛樋簸備尾微枇毘琵眉美鼻柊稗匹疋髭彦膝菱肘弼必畢筆逼桧姫媛紐百謬俵彪標氷漂瓢票表評豹廟描病秒苗錨鋲蒜蛭鰭品彬斌浜瀕貧賓頻敏瓶不付埠夫婦富冨布府怖扶敷斧普浮父符腐膚芙譜負賦赴阜附侮撫武舞葡蕪部封楓風葺蕗伏副復幅服福腹複覆淵弗払沸仏物鮒分吻噴墳憤扮焚奮粉糞紛雰文聞丙併兵塀幣平弊柄並蔽閉陛米頁僻壁癖碧別瞥蔑箆偏変片篇編辺返遍便勉娩弁鞭保舗鋪圃捕歩甫補輔穂募墓慕戊暮母簿菩倣俸包呆報奉宝峰峯崩庖抱捧放方朋法泡烹砲縫胞芳萌蓬蜂褒訪豊邦鋒飽鳳鵬乏亡傍剖坊妨帽忘忙房暴望某棒冒紡肪膨謀貌貿鉾防吠頬北僕卜墨撲朴牧睦穆釦勃没殆堀幌奔本翻凡盆摩磨魔麻埋妹昧枚毎哩槙幕膜枕鮪柾鱒桝亦俣又抹末沫迄侭繭麿万慢満漫蔓味未魅巳箕岬密蜜湊蓑稔脈妙粍民眠務夢無牟矛霧鵡椋婿娘冥名命明盟迷銘鳴姪牝滅免棉綿緬面麺摸模茂妄孟毛猛盲網耗蒙儲木黙目杢勿餅尤戻籾貰問悶紋門匁也冶夜爺耶野弥矢厄役約薬訳躍靖柳薮鑓愉愈油癒諭輸唯佑優勇友宥幽悠憂揖有柚湧涌猶猷由祐裕誘遊邑郵雄融夕予余与誉輿預傭幼妖容庸揚揺擁曜楊様洋溶熔用窯羊耀葉蓉要謡踊遥陽養慾抑欲沃浴翌翼淀羅螺裸来莱頼雷洛絡落酪乱卵嵐欄濫藍蘭覧利吏履李梨理璃痢裏裡里離陸律率立葎掠略劉流溜琉留硫粒隆竜龍侶慮旅虜了亮僚両凌寮料梁涼猟療瞭稜糧良諒遼量陵領力緑倫厘林淋燐琳臨輪隣鱗麟瑠塁涙累類令伶例冷励嶺怜玲礼苓鈴隷零霊麗齢暦歴列劣烈裂廉恋憐漣煉簾練聯蓮連錬呂魯櫓炉賂路露労婁廊弄朗楼榔浪漏牢狼篭老聾蝋郎六麓禄肋録論倭和話歪賄脇惑枠鷲亙亘鰐詫藁蕨椀湾碗腕

付録B　JIS 第 2 水準漢字

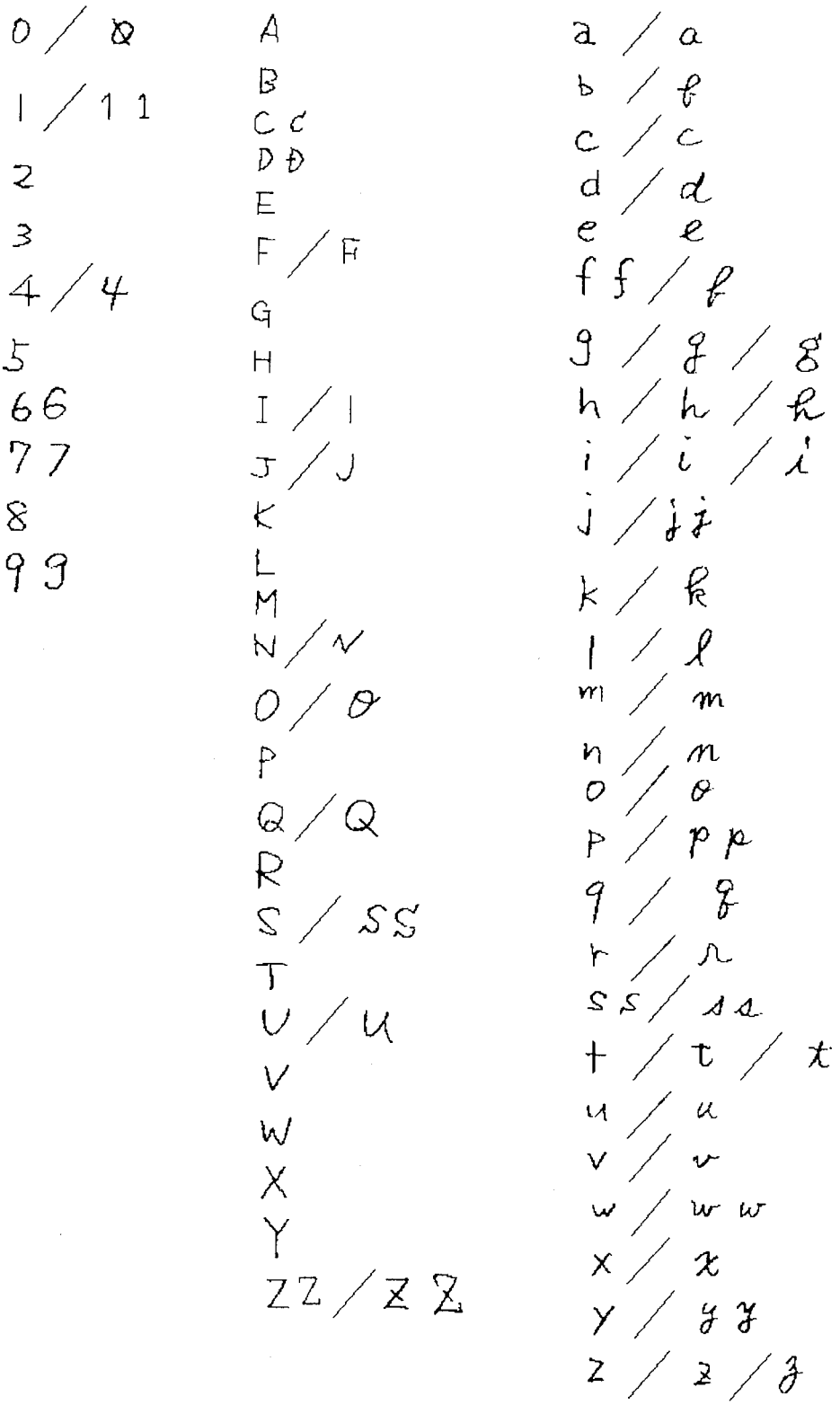
丼豫亊亞亢仆仞仭佚佛佇侏侑儘俎俤倅伜俯倆偕傀傅傳僥僭偺僂儡儼兒兔兢冤冦冪冽凅凜几凩凭凰
凾刎刹剋剌剪剳剽辦勁匆匈匳卮厖厠厦厮曼吽呵咎呟呷咀咄咬咤哭哺啖唸喀咯喘喩嗚嗅嗟嗜嘔嗾嘴
嘲嘯噪嚆嚔嚏嚥嚼囁囃囓囮國圓畬址垪埃埒埓堡毀墟壙壤壹壺壽夛夥夭奎奘奢奠奸妁佞侫姨姜婉娶
媚嫂嫺嫻嬌嬲孕孵寃寇寐寳屁屎屓屏屹峙崗崑崚崙崘嵌嶮嶽巌巫巳巵幀幟廈廣廝廚彝彜彌彎弯彗彙
彷彿徊徘徨忖忿怱恍恣恙悍惧悃悛悖悴忰愕慇愾愴慄慙慚慟憬憔憚憊憑憫懃懴懺懼戛截戮扁扣扠扱
抒拗拈拌拇拋拉挾掖掉掟揆揣揉插揶揄搦搏摧摯撓撥撼擅撻擡抬揹擲攣攫敲斂斟旁旛旱旻昂晏晧晨
晟晰暉暝曄暸曖曠昿曰朦朧覇杁杞杤枷柩檜栞梳條梃檮棘棕椒棹椚楷楔椰楡榮槁槃榜榴樂樅槭檣橙
樸橄檗蘗檻櫃櫂檸檬櫟櫞櫻欒鬱歇歿殲殱毬毫毯麾气氣汪沐泄泗沮泪洽洸洵洒洌浣涎涕濤涅渕涸淆
淺渾游溂溪溘滉溷漑灌滸滲漱澁潘澳澤澹濘濱瀑濾瀦瀝瀘瀟炙炒烔炬炸烟烋烙焉焜焙煥煕熙煌熏燻
熄熨燗熾燎燧燵燿爛爬爼犂犁犢犲狢狡猊猜猥猾獏獪獨珈珀琅瑯琥琲瑕瑙瑶璧瓏瓔琰瓣瓷甍甕甦畸
疆疇畴疣疳疵疽疸疼痍痒痙痣痺瘍瘉瘤瘻癇癈癪皓皸皹皺盪蘯眈眩眞眥眦眸睨睛瞑瞞瞰瞼矜矮砌砒
礦礪硅磋磔磧礒礙礫祀祓禧齋稟禀稻穢窖竈笙筐筍筵筧筬箋箒箏筝篏篝簀籠簗籃籔籐籤籖籬粃糒糴
紮紆絆絨絣絛綺綸緻縢繃縷繹繩繻纃纃罐罠羃羈羇羞羹羮翔翳耆聚聟聰肛肚肬胥胚胱脛脩腋隋脾腑
腱膊膀膠膣腟膓膾臀臘臚舅舍艫艪芻苟苺茉荅茲茹茫茗莉菫萃菁萍萠萼蕚萬蒡蔘蕁蘂薊薔藪薇蕾藉
藏蘊蘆虱蛉蠣蛛蛯蜀蜃蜷蜻蜚蝮螢蟄蟇螟蟒蠎蠢蠧蠹衄衾衽袵袂袮裔裹褄褪襄褻褶褌襌襞覺覿觴訃
訛訶詈詭詢誂誡誦諄諫諧諤諠諷謐謚謳謦謨證譜譛譟讓谺豺貉貎貌犾貘貪賤贄贅贓賍贔贖赭赳趙趺
跛跪踝蹂踵蹊蹉躄踪蹶躁躇躊躙躪躾軋軏輙輛輌輳輻轢辜辟辣辯辷迪邇逅逡逞適遒邂遽邁邀邊郁邱
郛鄂鄙酊酣酪釉鈑鉤銓鋏鋺錮錢鐵鐡鑪鈩鑚鑽鑿閂閙閏閤閾闊濶闕阡阯陞陟隘隕雹霙霰霹靭鞋鞏韋
齏韲竟頌頸頷頽顆顰颯颱飄飩飫餃餉餘餞饉饅饒馭駝駱騙騷驀驕驩髀髯鬘鬧魄魃鮨鯵鯰鰮鰛鳬鳧鶯
鴟鵄鵝鵞鵐鵯鵯鶇鶫麒麸麩靡黎黴鼾齟齧齬堯槇遙瑤凜熙

保守／廃止

〔メ　モ〕

保守／廃止

付録Ｃ　英数字の書き方



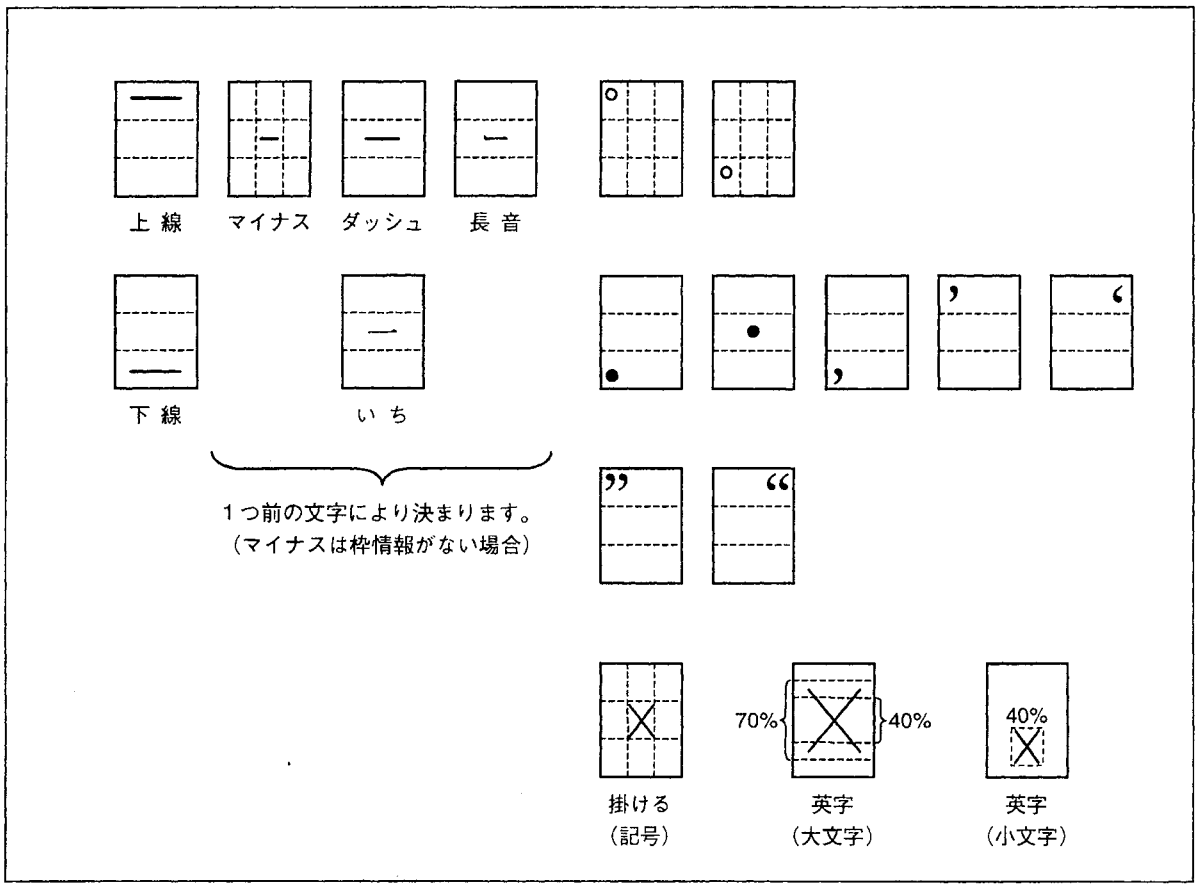
保守／廃止

〔メ　モ〕

付録D　同型文字の書き分け方

D.1　位置による書き分け方

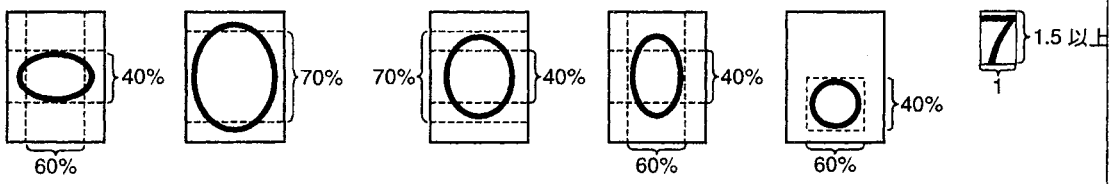
図D−1　位置による書き分け方



保守／廃止

D.2　大きさによる書き分け方

図D−2　大きさによる書き分け方



大きさ	高さ 40%未満 かつ 幅 60%以上	高さ 70%以上	高さ 70%未満 かつ 高さ 40%以上	高さ 40%以上 または 幅 60%以上	高さ 40%未満 かつ 幅 60%未満
形　状	高さが幅の 1.5 倍未満			高さが幅の 1.5 倍 以上	—
対　象	あ い う お つ や ゆ よ わ				あ い う お つ や ゆ よ わ
	ア イ ウ オ ツ ヤ ユ ヨ ケ				ア イ ウ オ ツ ヤ ユ ヨ ケ
	〉 テ 十 井				＞ 〒 十 #
	C S V W Z				c s v w z
	ワ			7	ワ
	之 力 二 口 く		え カ ニ 口 く		え カ ＝ ロ く
	工		工	I	エ
	O	O	O	0	o
	タ ト ハ へ べ ぺ リ	タ ト 八 へ べ ぺ り	タ ト ハ へ べ ぺ リ		

保守／廃止

付録Ｅ　略字の書き方

営		機		撰		点		問	
械		権		選		働		門	
間		個		前		品		曜	
関		事		層		風		様	
喜		質		卒		聞		歴	
器		職		第		簿		開	
閉									

〔メ　モ〕

保守／廃止

付録Ｆ　他の文字の認識候補になる文字

他の認識文字となる文字を次に示します。これらの文字は，表 F-1 の右に示す文字を入力して，その文字の認識候補の中から選択します。

表 F−1　他の文字の認識候補になる文字

文　字	認識候補になる文字
★	☆
●	○
◆	◇
■	□
▲	△
▼	▽
【	（
】	）
Δ（ギリシャ文字）	△（記号）
ο（ギリシャ文字）	o（英字）
Ⅰ（ローマ数字）	I（英字）
Ⅴ（ローマ数字）	V（英字）
Ⅹ（ローマ数字）	X（英字）
ⅰ（ローマ数字）	i（英字）
ⅴ（ローマ数字）	v（英字）
ⅹ（ローマ数字）	x（英字）

〔メ　モ〕

★ 付録Ｇ　サンプル・ソース

```
/*****************************************************************************
        Ultwriter-V Ver.2      外部共通ヘッダ (ultrecog.h)
                        Copyright (c) NEC Corporation 1997,1998
 *****************************************************************************/

/*****************************************************************************
        define
 *****************************************************************************/
#define   crg_MAXCAND          10      /* 認識結果の最大候補数 */
#define   crg_EOC     (short)0x7FFF    /* ストローク終点記号（short 用）*/
#define   crg_SPOINTSIZE   1000       /* 最大点数 */

/*****************************************************************************
        typedef
 *****************************************************************************/
/* V850 ファミリの場合は，以下の構造体 CRGMEM を使用する。
 V810 ファミリの場合は，以下の構造体 CRGMEM コメント・アウトする */
  typedef struct {
        long      *ReserveArea;      /* リザーブ・エリア先頭アドレス */
        long      *WorkArea;         /* ワーク・エリア先頭アドレス */
        short     IDRAMSw;           /* 内蔵データ RAM スイッチ */
        short     *IDRAMAdrs;        /* 内蔵データ RAM 先頭アドレス */
  } CRGMEM;
/* V810 ファミリの場合は，以下の構造体 CRGMEM を使用する。
 V850 ファミリの場合は，以下の構造体 CRGMEM コメント・アウトする */
  typedef struct {
        long      *ReserveArea;      /* リザーブ・エリア先頭アドレス */
        long      *WorkArea;         /* ワーク・エリア先頭アドレス */
  } CRGMEM;

typedef   struct {
        short     X;                 /* 座標データの x 座標 */
        short     Y;                 /* 座標データの y 座標 */
} CRGPOINT;

typedef struct {
        CRGPOINT        Point;       /* 座標データ */
        short           Pen;         /* ペン状態 */
} CRGHWDATA;
```

```
typedef struct {
        CRGPOINT        *TopBuf;        /* 文字バッファの先頭アドレス */
        CRGPOINT        *BottomBuf;     /* 文字バッファの最終アドレス */
} CRGBUF;

typedef struct {
        short   MaxX;           /* 最大 x 座標 */
        short   MaxY;           /* 最大 y 座標 */
        short   MinX;           /* 最小 x 座標 */
        short   MinY;           /* 最小 y 座標 */
} CRGRECT;

typedef struct {
        CRGPOINT        *Chr;   /* 認識させたい文字データの先頭アドレス */
        short           PntNum; /* 文字データを構成する座標データ数 */
        CRGBUF  *ChrBuf;        /* 文字バッファのアドレス情報のアドレス */
} CRGCHRPAT;

typedef struct {
        short           Value;  /* 距離値 */
        unsigned short  Index;  /* 文字データ(シフト JIS コード) */
} CRGCAND;

typedef struct {
        short           Warning;                /* ワーニング・フラグ */
        short           CandNo;                 /* 文字の候補数 */
        CRGRECT         InData;                 /* 入力文字の大きさ */
        CRGCAND         Candidate[crg_MAXCAND]; /* 認識候補 */
} CRGRESULT;

typedef struct {
        long    *DicArea;       /* 辞書バッファ先頭アドレス */
        short   DicState;       /* 辞書状態 */
} CRGDICSET;

typedef struct {
        int     Version;        /* 製品のバージョン番号 */
        char    *Serial;        /* 製品シリアル番号 */
        int     *Others;        /* その他の情報 */
} CRGVERSION;

/*****************************************************************************
        FILE END (ultrecog.h)
                Copyright (c) NEC Corporation 1997,1998
 *****************************************************************************/
```

```
/*****************************************************************************
        手書き文字認識ライブラリ用メイン・プログラム
                recogsam.c
                  Copyright (c) NEC Corporation 1997,1998
 *****************************************************************************/
#include <stdio.h>
#include "ultrecog.h"


/*****************************************************************************
        define
 *****************************************************************************/
#define   BSIZE    1000              /* 文字格納バッファ・サイズ */
#define   DSIZE    5000              /* ユーザ辞書サイズ */


/*****************************************************************************
        data
 *****************************************************************************/
CRGDICSET       DicSelect[3];       /* 辞書情報 */
CRGPOINT        RingBuf[BSIZE];     /* 座標格納用リング・バッファ */
CRGPOINT        *Pnt;               /* 座標データの先頭アドレス */
CRGMEM MemInfo;                     /* メモリ情報 */
CRGBUF  Buf;                        /* 文字バッファのアドレス */
long            Reserve[100];       /* リザーブ・データ領域 */
long            Work[8*1024];       /* ワーク領域 */
long            DicInfoNum;         /* 辞書情報数 */
long            UDicArea[DSIZE];    /* ユーザ辞書領域 */


/*****************************************************************************
        extern data
 *****************************************************************************/
extern short    RecogData[300];     /* 認識用文字データ */
extern short    UserData[300];      /* 登録用文字データ */


/*****************************************************************************
        prototype model
 *****************************************************************************/
void    main(void);                 /* メイン関数 */
void    sample_recog(void);         /* 文字認識関数 */
void    sample_user(void);          /* ユーザ登録関数 */
void    error_stop(void);           /* エラー・ストップ */
void    main_stop(void);            /* 正常終了 */
void    result_mod(CRGRESULT *);    /* 認識結果修正 */
```

```
/*****************************************************************************
        extern model
 *****************************************************************************/
/* 初期化*/
extern int  crg_Initialize(CRGMEM *);
/* 終了 */
extern int  crg_Uninitialize(CRGMEM *);
/* 処理停止 */
extern int  crg_StopProc(CRGMEM *);
/* 文字枠セット */
extern int  crg_SetGridParam(short, short, short, CRGMEM *);
/* 筆点単位処理 */
extern int  crg_SetPoint(CRGHWDATA, CRGPOINT *, CRGBUF, short *, CRGMEM *);
/* 認識処理 */
extern int  crg_RecogChar(CRGRECT,  CRGCHRPAT,  unsigned  short,  CRGRESULT  *,  long,  CRGDICSET  *,
CRGMEM *);
/* ユーザ辞書クリア */
extern int  crg_DicClearDic(long *, long, CRGMEM *);
/* 登録開始 */
extern int  crg_DicCreate(unsigned short, long *, long, CRGMEM *);
/* 登録用文字入力 */
extern int  crg_DicEntryChar(CRGCHRPAT, CRGMEM *);
/* 文字登録 */
extern int  crg_DicRegister(short *, CRGMEM *);
/* 登録文字消去 */
extern int  crg_DicEraseChar(short, short *, long *, CRGMEM *);
/* 登録文字数出力 */
extern int  crg_DicInformation(short *, long *, long *);
/* バージョン情報出力 */
extern int  crg_GetVersion(CRGVERSION *);


/*****************************************************************************
        main()    メイン
 *****************************************************************************/
void    main(void)
{
        short    XDotSize; /* X 方向のドットの大きさ */
        short    YDotSize; /* Y 方向のドットの大きさ */
        short    GridSize;  /* 文字枠の大きさ */
        long     DicSize;   /* ユーザ辞書領域サイズ */
```

```
/* メモリ情報セット */
MemInfo.ReserveArea = &Reserve[0];
MemInfo.WorkArea    = &Work[0];
/* V810 ファミリの場合は以下の 2 行をコメント・アウトする */
MemInfo.IDRAMSw     = 0;
MemInfo.IDRAMAdrs   = (short *)NULL;

/* 初期化 */
if(crg_Initialize(&MemInfo) != 0)
{
        error_stop();
}

/* ユーザ辞書初期化 */
DicSize = DSIZE * sizeof(long);
if(crg_DicClearDic(UDicArea, DicSize, &MemInfo) != 0)
{
        error_stop();
}

/* 初期設定 */
Buf.TopBuf    = &RingBuf[0];
Buf.BottomBuf = &RingBuf[BSIZE -1] + 1;
Pnt           = &RingBuf[0];
XDotSize = 1;       YDotSize = 1;
GridSize = 64;

/* 文字枠セット */
if(crg_SetGridParam(GridSize, XDotSize, YDotSize, &MemInfo) != 0)
{
        error_stop();
}

/* ユーザ辞書登録 */
sample_user();

/* 辞書情報セット */
DicInfoNum = 3;
/* 通常辞書 */
DicSelect[0].DicArea = (long *)0x00018000;          /* 先頭アドレス例 */
DicSelect[0].DicState = 1;
/* ユーザ辞書 */
DicSelect[1].DicArea = UDicArea;                    /* 先頭アドレス */
DicSelect[1].DicState = 1;
```

```
	/* 予備バッファ */
	DicSelect[2].DicArea = NULL;
	DicSelect[2].DicState = 0;

	/* 文字認識 */
	sample_recog();

	/* 終了 */
	if(crg_Uninitialize(&MemInfo) != 0)
	{
		error_stop();
	}

	main_stop();
}


/*****************************************************************************
	sample_recog()	認識処理
 *****************************************************************************/
void	sample_recog(void)
{
	CRGRESULT	Result;		/* 認識結果 */
	CRGHWDATA	HwData;	/* 筆点情報 */
	CRGCHRPAT	ChrPat;		/* 文字データ */
	CRGRECT		Grid;		/* 文字枠情報 */
	unsigned short	LastIndex;/* 前入力文字コード */
	short		*p, ptnum, Num;
	int		i;


	/* 初期設定 */
	LastIndex = 0;
	Grid.MaxX = 64;	Grid.MinX = 1;
	Grid.MaxY = 64;	Grid.MinY = 1;

	/* 文字コード、座標点数取得 */
	p = &RecogData[0];
	p++;	ptnum = *p++;

	/* 筆点単位処理 (座標点数分繰り返す) */
	ChrPat.PntNum = 0;
	ChrPat.Chr = Pnt;
	ChrPat.ChrBuf = &Buf;
	for(i=0; i<ptnum; i++)
```

```
        {
                HwData.Point.X = *p++;
                HwData.Point.Y = *p++;
                HwData.Pen     = *p++;

                if(crg_SetPoint(HwData, Pnt, Buf, &Num, &MemInfo) != 0)
                {
                        error_stop();
                }

                ChrPat.PntNum += Num;
                if(Num != 0)
                {
                        Pnt++;
                        if(Pnt == Buf.BottomBuf) Pnt = Buf.TopBuf;
                }
        }

        /* 文字認識 */
        if((i = crg_RecogChar(Grid, ChrPat, LastIndex, &Result, DicInfoNum, DicSelect, &MemInfo)) != 0)
        {
                error_stop();
        }

        /* 認識結果修正 */
        if(DicSelect[0].DicState != 0)
        {
                result_mod(&Result);
        }
}


/*****************************************************************************
        sample_user()       ユーザ辞書登録
 *****************************************************************************/
void    sample_user(void)
{
        CRGHWDATA       HwData;             /* 筆点情報 */
        CRGCHRPAT       ChrPat;                     /* 文字情報 */
        unsigned short  catg;                       /* 文字コード */
        short           *p, ptnum, Num, DicNum;
        long            DicSize;
        int             i, j;
```

```
/* 2 文字の登録 */
ChrPat.ChrBuf = &Buf;
for(i=0; i<2; i++)
{
        /* 文字コード、座標点数取得 */
        p = &UserData[0];
        catg = *p++;        ptnum = *p++;

        /* 登録開始 */
        DicSize = DSIZE * sizeof(long);
        if(crg_DicCreate(catg, UDicArea, DicSize, &MemInfo) != 0)
        {
                error_stop();
        }

        /* 筆点単位処理 (座標点数分繰り返す) */
        ChrPat.PntNum = 0;
        ChrPat.Chr = Pnt;
        for(j=0; j<ptnum; j++)
        {
                HwData.Point.X = *p++;
                HwData.Point.Y = *p++;
                HwData.Pen     = *p++;

                if(crg_SetPoint(HwData, Pnt, Buf, &Num, &MemInfo) != 0)
                {
                        error_stop();
                }

                ChrPat.PntNum += Num;
                if(Num != 0)
                {
                        Pnt++;
                        if(Pnt == Buf.BottomBuf) Pnt = Buf.TopBuf;
                }
        }

        if(crg_DicEntryChar(ChrPat, &MemInfo) != 0)
        {
                error_stop();
        }

        if(crg_DicRegister(&DicNum, &MemInfo) != 0)
```

```
		{
			error_stop();
		}
	}
}

/*****************************************************************************
	result_mod()		認識結果修正（ひらがなの場合）
 *****************************************************************************/
void	result_mod(CRGRESULT *Result)
{
	int		r,w;

	for( r = 0; r < crg_MAXCAND; r++ )
	{
		/* ひらがな以外の場合 */
		if( (Result->Candidate[r].Index < 0x829f) ||
		   (0x82f1 < Result->Candidate[r].Index) )
		{
			w = r;
			r++;
			break;
		}
	}
	for( ; r < crg_MAXCAND; r++ )
	{
		/* ひらがなだけを書き込む */
		if( (Result->Candidate[r].Index >= 0x829f) &&
		   (0x82f1 >= Result->Candidate[r].Index) )
		{
			Result->Candidate[w++].Index = Result->Candidate[r].Index;
		}
	}
	/* 余分な配列要素に 0x0000 を入れる */
	if(w)
	{
		for( ; w < crg_MAXCAND; w++)
		{
			Result->Candidate[w].Index = 0x0000;
		}
	}
}
```

```
/************************************************************************************
        error_stop()            エラー処理
 ************************************************************************************/
void    error_stop(void)
{
        while(1)
        {
                __asm("nop");
        }
}


/************************************************************************************
        main_stop()             終了処理
 ************************************************************************************/
void    main_stop(void)
{
        while(1)
        {
                __asm("nop");
        }
}
/************************************************************************************
    FILE END        Copyright (c) NEC Corporation 1997,1998
 ************************************************************************************/
```

```
/*****************************************************************************
        サンプル用データ
                        Copyright (c) NEC Corporation 1997,1998
 *****************************************************************************/


/*****************************************************************************
        認識用文字データ
 *****************************************************************************/
short RecogData[300] =
{
        0x82a0,   /* 文字コード */
        0x0031,   /* 座標点数 */
/*      x 座標     y 座標     ペン状態 */
        0x000d, 0x0017, 0x0001,
        0x000c, 0x0016, 0x0001,
        0x000c, 0x0016, 0x0001,
        0x000d, 0x0016, 0x0001,
        0x0015, 0x0015, 0x0001,
        0x001c, 0x0013, 0x0001,
        0x0023, 0x0012, 0x0001,
        0x0026, 0x0011, 0x0001,
        0x0026, 0x0011, 0x0001,
        0x0026, 0x0011, 0x0001,
        0x0026, 0x0011, 0x0001,
        0x0000, 0x0000, 0x0000,
        0x001a, 0x000d, 0x0001,
        0x0019, 0x000c, 0x0001,
        0x0019, 0x000f, 0x0001,
        0x0018, 0x0016, 0x0001,
        0x0018, 0x0020, 0x0001,
        0x0018, 0x0028, 0x0001,
        0x001b, 0x002e, 0x0001,
        0x001b, 0x0031, 0x0001,
        0x001c, 0x0031, 0x0001,
        0x0000, 0x0000, 0x0000,
        0x0029, 0x001a, 0x0001,
        0x0028, 0x0019, 0x0001,
        0x0028, 0x0019, 0x0001,
        0x0028, 0x001a, 0x0001,
        0x0028, 0x001b, 0x0001,
        0x0027, 0x0020, 0x0001,
        0x0023, 0x0027, 0x0001,
        0x001e, 0x002d, 0x0001,
        0x001a, 0x0032, 0x0001,
```

```
        0x0016, 0x0035, 0x0001,
        0x0014, 0x0036, 0x0001,
        0x0011, 0x0036, 0x0001,
        0x000e, 0x0034, 0x0001,
        0x000c, 0x0032, 0x0001,
        0x000b, 0x0030, 0x0001,
        0x000c, 0x002c, 0x0001,

        0x0010, 0x0028, 0x0001,
        0x0018, 0x0022, 0x0001,
        0x0020, 0x001f, 0x0001,
        0x0026, 0x001f, 0x0001,
        0x002b, 0x0021, 0x0001,
        0x002d, 0x0026, 0x0001,
        0x002c, 0x002c, 0x0001,
        0x002b, 0x0031, 0x0001,
        0x0029, 0x0034, 0x0001,
        0x0027, 0x0034, 0x0001,
        0x0000, 0x0000, 0x0000,
};


/*****************************************************************************
        登録用文字データ(2 文字分)
 *****************************************************************************/
short UserData[300] =
{
        0x82a2,   /* 文字コード */
        0x001c,   /* 座標点数 */
/*      x 座標     y 座標     ペン状態 */
        0x0009, 0x0010, 0x0001,
        0x0009, 0x0010, 0x0001,
        0x0009, 0x0010, 0x0001,
        0x0009, 0x0010, 0x0001,
        0x0009, 0x0013, 0x0001,
        0x0009, 0x0018, 0x0001,
        0x000b, 0x001e, 0x0001,
        0x000c, 0x0025, 0x0001,
        0x000f, 0x002b, 0x0001,
        0x0011, 0x002f, 0x0001,
        0x0015, 0x0033, 0x0001,
        0x0017, 0x0036, 0x0001,
        0x0018, 0x0036, 0x0001,
        0x001a, 0x0036, 0x0001,
        0x001c, 0x0034, 0x0001,
        0x001e, 0x0030, 0x0001,
```

```
        0x001f, 0x002c, 0x0001,
        0x0000, 0x0000, 0x0000,
        0x002c, 0x0012, 0x0001,
        0x002c, 0x0012, 0x0001,
        0x002d, 0x0012, 0x0001,
        0x002e, 0x0014, 0x0001,
        0x0030, 0x0017, 0x0001,
        0x0032, 0x001b, 0x0001,
        0x0034, 0x001f, 0x0001,
        0x0035, 0x0022, 0x0001,
        0x0035, 0x0022, 0x0001,
        0x0000, 0x0000, 0x0000,
        0x82a4,   /* 文字コード */
        0x0018,   /* 座標点数 */

/*          x 座標      y 座標      ペン状態 */
        0x0017, 0x000a, 0x0001,
        0x0016, 0x000a, 0x0001,
        0x0016, 0x000a, 0x0001,
        0x0018, 0x000a, 0x0001,
        0x001d, 0x000a, 0x0001,
        0x0020, 0x000a, 0x0001,
        0x0000, 0x0000, 0x0000,
        0x0013, 0x001b, 0x0001,
        0x0012, 0x001b, 0x0001,
        0x0012, 0x001b, 0x0001,
        0x0012, 0x001b, 0x0001,
        0x0012, 0x001b, 0x0001,
        0x0013, 0x001a, 0x0001,
        0x0017, 0x0017, 0x0001,
        0x001c, 0x0016, 0x0001,
        0x0023, 0x0015, 0x0001,
        0x0028, 0x0018, 0x0001,
        0x002b, 0x001d, 0x0001,
        0x0029, 0x0025, 0x0001,
        0x0025, 0x002e, 0x0001,
        0x001f, 0x0034, 0x0001,
        0x001a, 0x0038, 0x0001,
        0x0018, 0x0038, 0x0001,
        0x0000, 0x0000, 0x0000,
};
/*****************************************************************************
    FILE END        Copyright (c) NEC Corporation 1997,1998
 *****************************************************************************/
```

〔メ　モ〕

保守／廃止

付録H　総合索引

H.1　50音で始まる語句の索引

保守／廃止

H. 2　アルファベットで始まる語句の索引

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：午前 9:00～12:00，午後 1:00～5:00）

電　話　：044-548-8899
FAX　：044-548-7900
E-mail　：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)251-5599	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	京都支社	京　都 (075)344-7824
岩手支店	盛　岡	(019)651-4344	甲府支店	甲　府	(0552)24-4141	神戸支社	神　戸 (078)333-3854
郡山支店	郡　山	(0249)23-5511	長野支社	松　本	(0263)35-1662	中国支社	広　島 (082)242-5504
いわき支店	いわき	(0246)21-5511	静岡支社	静　岡	(054)254-4794	鳥取支店	鳥　取 (0857)27-5311
長岡支店	長　岡	(0258)36-2155	立川支社	立　川	(042)526-5981,6167	岡山支店	岡　山 (086)225-4455
水戸支店	水　戸	(029)226-1717	埼玉支社	大　宮	(048)649-1415	松山支店	松　山 (089)945-4149
土浦支店	土　浦	(0298)23-6161	千葉支社	千　葉	(043)238-8116	九州支社	福　岡 (092)261-2806
群馬支店	高　崎	(027)326-1255	神奈川支社	横　浜	(045)682-4524		
太田支店	太　田	(0276)46-4011	三重支店	津	(059)225-7341		

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］ μSAP703000-B09，μSAP70732-B09 ユーザーズ・マニュアル
（U13253JJ2V0UM00（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

保守／廃止