(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4381483号
(P4381483)

(45)発行日　平成21年12月9日(2009.12.9)　(24)登録日　平成21年10月2日(2009.10.2)

(51)Int.Cl.		F I		
G06K 9/62	(2006.01)	G06K	9/62	G
G06F 17/22	(2006.01)	G06F	17/22	

請求項の数 3　(全 27 頁)

(21)出願番号	特願平7-180052
(22)出願日	平成7年7月17日(1995.7.17)
(65)公開番号	特開平9-34881
(43)公開日	平成9年2月7日(1997.2.7)
審査請求日	平成14年6月18日(2002.6.18)
審判番号	不服2006-9096(P2006-9096/J1)
審判請求日	平成18年5月8日(2006.5.8)

(73)特許権者　000002185
ソニー株式会社
東京都港区港南１丁目７番１号
(74)代理人　100067736
弁理士　小池　晃
(74)代理人　100096677
弁理士　伊賀　誠司
(72)発明者　篠塚　英明
東京都品川区北品川６丁目７番３５号　ソニー株式会社内
(72)発明者　脇田　能宏
東京都品川区北品川６丁目７番３５号　ソニー株式会社内

最終頁に続く

(54)【発明の名称】文字情報処理装置及び文字情報処理方法

(57)【特許請求の範囲】

【請求項１】

予め規格化された標準文字コードにより表現される文字を入力するための第１の入力手段と、

文字に対応するパターンデータを入力するための第２の入力手段と、

上記第２の入力手段を介して入力されたパターンデータを１文字分毎に分けて、パターンデータ指示制御コードに続くパターンデータ番号として記録するパターンデータ記録手段と、

上記パターンデータ記録手段からの上記パターンデータ指示制御コードに続く上記パターンデータ番号と、上記第１の入力手段を介した入力により生成される上記標準文字コードとから文字列データを形成して記憶する文字列データ記憶手段と、

上記文字列データ記憶手段から読み出した文字列データの１文字毎に標準文字コードにより表現される文字であるか又はパターンデータであるかを判定する判定手段と、

上記判定手段の判定結果に応じて、１文字毎にデータ処理を行う文字列データ処理手段とを備えることを特徴とする文字情報処理装置。

【請求項２】

上記文字列データ処理手段で処理された文字列データを出力する出力手段をさらに備えることを特徴とする請求項１記載の文字情報処理装置。

【請求項３】

予め規格化された標準文字コードにより表現される文字を入力するための第１の入力工

程と、
文字に対応するパターンデータを入力するための第２の入力工程と、
上記第２の入力工程において入力されたパターンデータを１文字分毎に分けて、パターンデータ指示制御コードに続くパターンデータ番号としてパターンデータ記録手段に記録するパターンデータ記憶工程と、
上記パターンデータ記録手段からの上記パターンデータ指示制御コードに続く上記パターンデータ番号と、上記第１の入力工程における入力により生成される上記標準文字コードとから文字列データを形成して文字列データ記憶手段に記憶する文字列データ記憶工程と、
上記文字列データ記憶手段から読み出した文字列データの１文字毎に標準文字コードにより表現される文字であるか又はパターンデータであるかを判定する判定工程と、
上記判定工程の判定結果に応じて、１文字毎にデータ処理を行う文字列データ処理工程とを備えること
を特徴とする文字情報処理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、予め規格化された標準文字コードにより表現される文字と文字に対応するパターンデータとを共に扱う文字情報処理装置及び文字情報処理方法に関する。

【０００２】

【従来の技術】

最近、手帳に文字を書き込むように、例えばタブレットに文字、記号等の情報を手書き入力し、内蔵の記憶部に格納しておくような、文字情報処理装置が考えられている。この文字情報処理装置で上記文字、記号等を手書き入力する場合には、上記文字、記号等をピクセルデータ、ストロークデータ等のようなグラフィックデータ又はパターンデータとして取り扱う方法や、文字認識をして文字コードで取り扱う方法がある。

【０００３】

このような従来の文字情報処理装置について図２２のフローチャートを参照しながら説明する。既にメモリに記憶されている文字列データに対して、ステップＳ１０に示すように例えば、コピー、表示、検索等のデータ処理が開始されると、ステップＳ２０では、１文字分のデータを読む。文字コードは任意の符号化が行われていてよい。現在、計算機上で日本語の文字を表現するためには２バイト（１６ビット）の文字コードを使用するのが一般的であるが、その場合は２バイト分のデータを読む。ステップＳ４０では、ステップＳ２０で読み込んだ文字コードの文字に対して所望の処理を行う。ステップＳ８０では、文字列データの最後か否かを確認し、続く文字データがある場合には次の文字に対する処理に移る。ステップＳ９０では、文字列データに対する処理を終了する。なお、ステップＳ８０の文字列データの終端チェックは、ステップＳ２０の前に行うこともできる。

【０００４】

ところで、上記文字情報処理装置では、講演の記録や議事録を取る時等の文字認識や候補選択をしている時間的余裕が無い場合、取り合えず手書き文字のまま上記記憶部に格納しておき、後で時間をかけて文字認識と候補選択、場合によっては文字単位の書き直しができるとよい。また、文字認識をしている時間的余裕があっても、文字認識モジュールとの相性が悪い場合に手書き文字を保存しておき、後で別の認識モジュールを使えるとよい。

【０００５】

【発明が解決しようとする課題】

しかし、上記文字情報処理装置では、手書き入力された上記パターンデータとして保存しているような場合、該パターンデータをそのまま予め規格化された標準文字コードにより表現される文字列中に埋め込めなかった。

【０００６】

そこで、ワードプロセッサで扱われる外字のように既存の文字コードの空き領域のみを上

記パターンデータで表現される手書き文字として使う方法が考えられる。しかし、この場合には、ＪＩＳの漢字第１水準、第２水準のみを用いている場合で二千数百、他のコード体系でも約六千位の手書き文字しか取り扱えなかった。

【０００７】

また、例えば異なる種類の二つの文字情報処理装置間で作られた異なったデータ形式の外字を、どちらか一つの文字情報処理装置で繋げる場合には、一方の形式に統一するために、データの変換や外字の再登録を行う必要があった。

【０００８】

本発明は、上記実情に鑑みてなされたものであり、文字に対応するパターンデータを予め規格化された標準文字コードにより表現される文字列中に埋め込める文字情報処理装置及び文字情報処理方法の提供を目的とする。

【０００９】

【課題を解決するための手段】

本発明の文字情報処理装置は、上記課題を解決するために、予め規格化された標準文字コードにより表現される文字を入力するための第１の入力手段と、文字に対応するパターンデータを入力するための第２の入力手段と、上記第２の入力手段を介して入力されたパターンデータを１文字分毎に分けて、パターンデータ指示制御コードに続くパターンデータ番号として記録するパターンデータ記録手段と、上記パターンデータ記録手段からの上記パターンデータ指示制御コードに続く上記パターンデータ番号と、上記第１の入力手段を介した入力により生成される上記標準文字コードとから文字列データを形成して記憶する文字列データ記憶手段と、上記文字列データ記憶手段から読み出した文字列データの１文字毎に標準文字コードにより表現される文字であるか又はパターンデータであるかを判定する判定手段と、上記判定手段の判定結果に応じて、１文字毎にデータ処理を行う文字列データ処理手段とを備えることを特徴としている。

また、本発明の文字情報処理方法は、上記課題を解決するために、予め規格化された標準文字コードにより表現される文字を入力するための第１の入力工程と、文字に対応するパターンデータを入力するための第２の入力工程と、上記第２の入力工程において入力されたパターンデータを１文字分毎に分けて、パターンデータ指示制御コードに続くパターンデータ番号としてパターンデータ記録手段に記録するパターンデータ記憶工程と、上記パターンデータ記録手段からの上記パターンデータ指示制御コードに続く上記パターンデータ番号と、上記第１の入力工程における入力により生成される上記標準文字コードとから文字列データを形成して文字列データ記憶手段に記憶する文字列データ記憶工程と、上記文字列データ記憶手段から読み出した文字列データの１文字毎に標準文字コードにより表現される文字であるか又はパターンデータであるかを判定する判定工程と、上記判定工程の判定結果に応じて、１文字毎にデータ処理を行う文字列データ処理工程とを備えることを特徴としている。

【００１０】

【発明の実施の形態】

以下、本発明に係る文字情報処理装置のいくつかの実施の形態について説明する。いずれの実施の形態もキーボードからのキー入力で予め規格化された標準文字コードにより表現される通常文字を、またタブレットからの専用ペンを用いての手書き入力で文字に対応するパターンデータを入力できる。そして、これらの通常文字及びパターンデータに複写、挿入等のデータ処理を施して表示できる文字情報処理装置である。なお、タブレットの代わりに、タッチパネルから手書き入力を行っても良い。

【００１１】

先ず、第１の実施の形態は、図１に示すように、データキー等のキーによって上記標準文字コードにより表現される通常文字を入力するキーボード１や上記パターンデータを手書き入力するタブレット２を備える文字入力部３と、タブレット２からの上記パターンデータを１文字分毎に分けて図２の（Ｂ）に示す手書き文字指示制御コード（以下、制御コードという。）に続く手書き文字番号（合わせて手書き文字コードという。）として記録す

る手書き文字データベース５と、手書き文字データベース５からの上記手書文字指示制御コードに続く上記手書き文字番号と、キーボード１から入力された図２の（Ａ）に示す上記標準文字コード（図中、文字コードと示す。）とから文字列データを形成して記憶する文字列データ記憶部６と、文字列データ記憶部６から読み出した文字列データの１文字毎に上記標準文字コードにより表現される通常文字であるか又はパターンデータであるかを判定し、それぞれに応じたデータ処理を行う文字列データ処理部４と、文字列データ処理部４で処理された文字列データを表示する文字表示部１１とを備えて成る。

【００１２】

図２の（Ｂ）に示した制御コードは、図２の（Ａ）に示した文字コードと区別することができるコードを使用する。例えば文字コードとしてシフトＪＩＳコードを使っている場合には、ＦＦＦＦ等が考えられる。この制御コードは、手書き文字処理部１０で付加されてもよいし、手書き文字データベース５で付加されてもよい。また、手書き文字番号は、該当する手書き文字の手書き文字データベース５での登録番号である。また、計算機内部での手書き文字番号の表現形式は、番号との１対１の対応さえ取れればどのような符号化でもよい。

【００１３】

文字列データ処理部４は、文字列データ記憶部６から読み出した上記文字列データに共通の処理を施す共通処理部７と、上記文字列データの１文字分が上記パターンデータ指示制御コードを含む文字コードであるか否かを判定する文字コード判定部８と、文字コード判定部８での判定結果により上記標準文字コードにより表現される通常文字に所望のデータ処理を施す通常文字処理部９と、文字コード判定部８での判定結果により上記パターンデータ指示制御コードに続く手書き文字番号に応じた手書き文字に所望のデータ処理を施す手書き文字処理部１０とを備えて成る。

【００１４】

キーボード１によってキー入力される上記標準文字コードにより表現される通常文字は、通常文字処理部９を介して文字列データ記憶部６に供給される。一方、タブレット２によって入力された上記パターンデータは、手書き文字処理部１０によって１文字分毎に分けられ、手書き文字データベース５に供給される。手書き文字データベース５は、上記パターンデータを手書き文字番号に対応づけて１文字分毎に管理する。この手書き文字データベース５に記録された上記パターンデータは、文字列データ記憶部６に供給される。文字列データ記憶部６には、通常文字処理部９を介した上記標準文字コードにより表現される通常文字も供給されている。文字列データ記憶部６は、上記パターンデータ指示制御コードに続く上記手書き文字番号と上記標準文字コードとから文字列データを形成する。

【００１５】

文字列データ処理部４は、文字列データ記憶部６から読み出した上記文字列データの１文字分が図２の（Ａ）に示す文字コード（標準文字コード）であるか又は図２の（Ｂ）に示す手書き文字コードであるかを判定し、それぞれに応じたデータ処理を行う。この文字列データ処理部４は、データを処理するためのＣＰＵ、プログラムのような処理手順を記憶しているＲＯＭ、処理中のデータを憶えるＲＡＭから構成され、上述したように共通処理部７、文字コード判定部８、通常文字処理部９、手書き文字処理部１０をソフトウェア的に実現している。

【００１６】

共通処理部７は、文字列データ記憶部６から読み出した上記文字列データに共通の処理を実行する。ここで、上記通常文字と上記パターンデータで処理を変更する必要がある場合には、文字の種類に応じてデータを渡す処理部を指定してデータを文字コード判定部８に渡す。パターンデータ、すなわち手書き文字以外の新しい種類の文字を含むデータを扱う場合でも、文字の種類に関わらず行う処理と、文字の種類による分岐制御のある処理を行うという機能は変化しない。

【００１７】

文字コード判定部８は、共通処理部７から渡された上記文字列の１文字分が上記パターン

データ指示制御コードを含む文字コードであるか否かを判定する。そして、データの種類に応じて、所望されるデータ処理に制御を移す。上記手書き文字以外の新しい種類の文字が増えても文字コードか制御コードかを判定して文字の種類を判断するという機能は変化しない。

【００１８】

通常文字処理部９は、受け取ったデータを全て標準文字コードとして扱う。手書き文字処理部１０は、手書き文字データベース５を利用し、受け取ったデータを全て手書き文字データベース５での手書き文字番号として扱う。ここで、手書き文字処理部１０が複数のデータベースを利用する場合には、一つの手書き文字処理部１０が一括して管理することもできるし、データベース毎に異なった手書き文字処理部を用意して、文字コード判定部８で処理を分岐することもできる。

【００１９】

また、手書き文字として複数の形式のグラフィックデータを扱う場合には、手書き文字データベース５の中にデータ形式を示すフィールドを追加する。或いは、データ形式毎に異なる手書き文字データベースを用意し、異なった制御コードを使用することもできる。

【００２０】

また、手書き文字以外にも新しい種類の文字（例えば音声入力による音声文字等）が増えた場合には増えた種類の文字専用の処理部が追加され、専用のデータベースを利用し、文字の種類毎に制御コードを別にして管理することができる。

【００２１】

そして、データベースの数が増えて必要な制御コードが増えてきた場合には、データベースのデータベースとでもいうデータベースの管理機構をもう一段増やすことにより、通常の文字コードとの区別のためのコードは一つで済ませることができる。

【００２２】

以上に構成を説明した第１の実施の形態の文字情報処理装置の動作を図３のフローチャートを用いて説明する。ここでは、文字入力部３のキーボード１及びタブレット２から入力された上記通常文字及び上記手書き文字からなる文字列データは、文字列データ記憶部６に既に記憶されているとする。なお、ここでは、タブレット２から入力された上記パターンデータにより表現される上記手書き文字は、１文字１文字の大きさが正規化されているものとするが、タブレット２上に枠を設け入力時に文字の大きさを制限したり、指定してもよい。また、入力された文字はその度に、あるいは指定された文字列毎に文字表示部１１に表示される。

【００２３】

先ず、文字入力部３内のキーボード１又はタブレット２での入力状態、もしくは文字列データ処理部４内の上記ＣＰＵが実行中のプログラムによって、ステップＳ１０に示すように文字列データの処理開始指示が発生する。文字列データ処理部４として行うべき処理により、共通処理部７、通常文字処理部９、手書き文字処理部１０の行う処理はそれぞれ変化する。

【００２４】

ステップＳ２０では、共通処理部７の制御により文字列データ記憶部６から１文字分のデータが読み込まれ、文字コード判定部８に供給される。

【００２５】

ステップＳ３０は、文字コード判定部８で行われる処理である。文字列データ記憶部６から共通処理部７の制御により文字コード判定部８に供給された１文字分のデータが図２の（Ｂ）に示した手書き文字指示制御コードであるか否かを見分ける。手書き文字指示制御コードは、上記手書き文字に用いられるコードであるので、ここでは読み込まれた１文字分のデータが手書き文字であるか否かを判定している。具体的に、文字コード判定部８は、内部にＲＯＭを有し、このＲＯＭ内のコードを参照して判定を行っている。上記１文字分のデータが手書き文字指示制御コードでない、すなわち通常文字であればステップ４０に進む。また、上記１文字分のデータが手書き文字指示制御コードであれば、ステップ５

０に進む。
【００２６】
ステップＳ４０は、通常文字処理部９で実行される処理である。処理内容は、文字列データ処理部４自体で行うべき処理により異なる。
【００２７】
ステップＳ５０は、手書き文字処理部１０で実行される処理であり、手書き文字コードの内の手書き文字指示制御コードに続く手書き文字番号を読む。なお、この処理は、共通処理部７に依頼してもよい。
【００２８】
ステップ６０も手書き文字処理部１０で実行される処理であり、処理内容は、文字列データ処理部４自体で行うべき処理により異なるが、手書き文字データベース５を利用して行われる。ステップＳ５０で読み込んだ手書き文字番号の手書き文字に対して、通常文字データで行われている処理と見た目が同様の処理を行う。例えば、表示等でグラフィックデータが必要な場合に、文字コードと指定されたフォントからビットマップデータ等を得る代わりに、指定された手書き文字データベース５を参照して該当する手書き文字番号のグラフィックデータを得る。
【００２９】
ステップＳ７０は、通常文字であるか手書き文字であるかに関わらず共通処理部７で実行される処理である。処理内容は、文字列データ処理部４で行うべき処理によって異なる。例えば、表示の際の表示位置の変更（カーソル移動）等の処理である。通常文字と手書き文字で共通な部分が無い処理や、区別する必要がない処理も有り得る。
【００３０】
ステップＳ８０は、共通処理部７で行われる処理であり、文字列データの最後かどうかを確認し、続く文字データがある場合には次の文字に対するステップＳ２０の処理に移る。文字列データの最後を確認する方法としては、処理文字数のカウンタを持ち指定文字数になったか否かを判定する方法や、文字列データの終端を示すデータを読み込んだか否かを判定する方法がある。この処理は、文字列データの格納方法によって決まり、文字列データ処理部４で行うべき処理が異なっても変化しない。特殊な処理が必要な場合は、ステップＳ４０，Ｓ６０，Ｓ７０で行われる。
【００３１】
ステップＳ９０は、文字列データ処理部４全体としての処理が終了したことを示す。これは、共通処理部７で行われる。
【００３２】
ところで、キーボード１からの入力の場合には、キーを押す毎に通常の文字コードが生成される。文字コードは、任意の符号化が行われていて良い。現在、計算機上で日本語の文字を表現するためには２バイト（１６ビット）の文字コードを使用するのが一般的だが、その場合は２バイト分のデータを読む。
【００３３】
これに対してタブレット２から上記手書き文字を入力する場合はもう少し複雑となる。タブレット２から入力されるのは点列のデータであり、タブレット２上に書いた時点ではパターンデータに過ぎず手書き文字ではない。手書き文字処理部１０で１文字分のデータとして指示された時点で手書き文字となる。この手書き文字は、手書き文字データベース５に登録されて手書き文字番号を得る。タブレット２の表面に設けられた枠の中に文字を書くことで自動的に一文字分のデータと判別し手書き文字として登録されることも考えられるが、何れにせよ手書き文字データベース５に登録され、手書き文字番号を得るまでは手書き文字ではなく単なるパターンデータである。
【００３４】
逆に単なるパターンデータとして保存していたものの一部を手書き文字として登録することも可能である。タブレット２やマウス等のポインティングデバイスとＲＡＭ内のデータが文字入力部となる。

【００３５】
同様に、音声文字等が有る場合にも音声文字データベース等に登録され固有の文字番号を得るまでは音声データ等であり音声文字等の文字では無いし、既に音声データ等として保存してあるものを音声文字等として登録することもできる。
【００３６】
次に、通常文字と手書き文字が混在した文字列データの具体例について図４を参照しながら説明する。この具体例は、通常の文字コードとしてシフトＪＩＳコードを使い、手書き文字指示制御コードとしてＦＦＦＦ、手書き文字番号が２バイトの場合の文字列データであり、図４に示すように、ＪＩＳコード“８２Ａ０”及び“８２Ａ２”の通常文字「あ」及び「い」に続いて、手書き文字コード“ＦＦＦＦ０００１”の手書き文字＃１が続き、さらにＪＩＳコード“８２Ａ４”の「う」の後に、手書き文字コード“ＦＦＦＦ０００２”の手書き文字＃２が続く。さらに、手書き文字コード“ＦＦＦＦ０００３”の手書き文字＃３が続き、最後にＪＩＳコード“８２Ａ６”の「え」が並んでいるような文字列である。この文字列は、文字列データ記憶部６内に記憶されている。例えば、手書き文字＃１の手書き文字コード“ＦＦＦＦ０００１”の内、“ＦＦＦＦ”は手書き文字指示制御コード、“０００１”は手書き文字番号である。手書き文字番号＃２及び＃３も、手書き文字指示制御コード“ＦＦＦＦ”の他に、“０００２”及び“０００３”のような手書き文字番号を有している。このため、手書き文字番号には、既存の文字のシフトＪＩＳコードとは全く別の文字コードを使用できる。このため、例えば手書き文字番号が２バイト（１６ビット）の場合、扱える文字数は６５５３６文字となる。
【００３７】
この図４に示した文字列データの具体例を扱った場合の、図１に示す第１の実施の形態の文字情報処理装置の動作について図５のフローチャートを参照しながら説明する。図５のフローチャートは、基本的に図３のフローチャートと同様であるので、各ステップ番号は同一とする。なお、ステップＳ１５については、ステップＳ１０とステップＳ２０との間に新たに設ける。
【００３８】
先ず、ステップＳ１０では、図４に示した文字列データの具体例が格納されている文字列データ記憶部６のアドレスaddrと、処理をする文字数Ｎ（図４の具体例では７）と、通常文字に対する処理内容char_funcと、手書き文字に対する処理内容hw_funcと、両者に共通な処理内容common_funcとを指定して文字列処理を開始する。例えば、通常文字に対する処理内容char_funcと、手書き文字に対する処理内容hw_funcとは、後述するように指定文字数のデータのサイズを確認するという処理であったり、指定文字数のデータを表示するという処理であってもよい。もちろん、手書き文字を文字認識して通常文字のように清書するような処理でもよい。これは、時間がない場合に、キーボード１からの通常入力とタブレット２からの手書き入力とを混在させて、時間があるときに手書き文字を清書しようとする場合に適用できる。
【００３９】
ステップＳ１５では、通常文字に対する処理内容char_funcや、手書き文字に対する処理内容hw_funcを行うために必要な初期化を行う。この初期化は、処理によって異なる。例えば、表示するという処理では、画面のどの位置から表示を始めるかを決める。また、手書き入力を清書するという処理では、手書き文字モジュールを起動させて、辞書等を準備する。
【００４０】
ステップＳ２０では、共通処理部７の制御により文字列データ記憶部６から１文字分の２バイトのデータｃを読み込む。アドレスaddrは、２バイト毎に移動する。
【００４１】
ステップＳ３０では、文字コード判定部８により１文字分の２バイトのデータｃが制御コード（手書き文字指示制御コード）ＦＦＦＦであるか否かの判定を行う。
【００４２】

ステップＳ４０では、ステップＳ３０での判定がＮＯの場合、すなわち上記データｃが制御コードＦＦＦＦではないとの判定を受けて、上記データｃに対して通常文字に対する処理char_funcを実行する。

【００４３】

ステップＳ５０では、ステップＳ３０での判定がＹＥＳの場合、すなわち上記データｃが制御コードＦＦＦＦであると判定されたのを受けて、上記データｃの２バイトの手書き文字番号ｎを読み込む。この場合のアドレスaddrは、２バイト毎に移動する。

【００４４】

ステップＳ６０では、ステップＳ５０で読み込んだ手書き文字番号ｎの手書き文字に対して、処理hw_funcを実行する。

【００４５】

ステップＳ７０では、両者に共通な処理内容common_funcを実行する。未処理の文字数Ｎを１減らす処理を必ず行う。

【００４６】

ステップＳ８０では、文字列データの終わりか否かを判断する。すなわち、未処理の文字数Ｎが０か否かを判断する。未処理の文字Ｎが残っていれば、ステップＳ２０に戻って、次の文字の処理を続ける。ステップＳ９０では、文字列データの処理を終了する。

【００４７】

このフローチャートでは、ステップＳ１５、ステップＳ４０、ステップＳ６０、ステップＳ７０の個々の処理が、それぞれの文字に対しての上記データ処理の種類に応じて異なる。

【００４８】

次に、データ処理の例として、指定文字数のデータのサイズを確認するという処理を行わせる場合の第１の実施の形態の動作について、ステップＳ１５、ステップＳ４０、ステップＳ６０の個々の処理を書き換えた図６を参照しながら説明する。図５と同様のステップには、同一符号を付し説明を省略する。

【００４９】

ステップＳ１５では、char_funcや、hw_funcのための初期化を行う。具体的にはバイト単位のデータサイズのためのカウンタsizeを初期化して０にする。

【００５０】

ステップＳ４１乃至ステップＳ４３は、図５のステップＳ４０をデータサイズ確認という処理に適用したステップである。ステップＳ４１で文字コードｃの処理を開始し、ステップＳ４２でデータサイズのカウンタsizeを２増やす。これは、文字コードが２バイトのためである。そして、ステップＳ４３で文字コードｃの処理を終了する。

【００５１】

ステップＳ６１乃至ステップＳ６３も、図５のステップＳ６０をデータサイズ確認という処理に適用したステップである。ステップＳ６１で手書き文字番号ｎの処理を開始し、ステップＳ６２でデータサイズのカウンタsizeを４増やす。これは、制御コードが２バイト、手書き文字番号が２バイトで手書き文字コードが計４バイトとなるためである。

【００５２】

そして、ステップＳ７０で１文字毎に行う処理は、未処理の文字数を減らすだけで特別な処理は無い。

【００５３】

なお、ステップＳ９０では、文字列処理終了後、sizeにＮ文字分のデータのバイト数が残っている。

【００５４】

次に、処理の例として、指定文字数のデータを表示するという処理を行わせる場合の第１の実施の形態の動作について説明する。先ず、この指定文字数のデータを表示するという処理の概念について図７を参照しながら説明しておく。

【００５５】

文字の位置を示す座標は文字の左下であり、一文字目の位置が（ｘ０，Ｙ０）とする。文字列を左から右に横書きで表示し、隣の文字はwidthだけ位置が違う。次の行の文字との位置の違いは、heightとする。

【００５６】

これは、図５のフローチャートのステップＳ１５、ステップＳ４０、ステップＳ６０、ステップＳ７０の個々の処理を書き換えてなる図８のフローチャートによって説明できる。

【００５７】

ステップＳ１５では、char_funcやhw_funcのための初期化を行う。具体的には、指定文字数のデータを表示するという処理で必要なパラメータの設定を行う。先ず、列の最小位置をｘ０、最大位置をｘ１とする。また、列中での隣の文字との差ｄｘをwidth、行中の前後の差ｄｙをheightとする。さらに、一文字目の開始位置を設定する。

【００５８】

ステップＳ４１～ステップＳ４４は、図５のステップＳ４０を指定文字数のデータを確認するという処理に適用したステップである。ステップＳ４１で文字コードｃの処理を開始し、ステップＳ４２で文字コードｃのフォントデータを取得してから、ステップＳ４３でフォントを（ｘ，ｙ）の位置に表示する。このステップＳ４２とステップＳ４３の処理は、文字を表示するための処理である。そして、ステップＳ４４で文字コードｃの処理を終了する。

【００５９】

ステップＳ６１～ステップＳ６４も、図５のステップＳ６０を指定文字数のデータを確認するという処理に適用したステップである。ステップＳ６１で手書き文字番号ｎの処理を開始し、ステップＳ６２で手書き文字データベース６から番号ｎのデータを取得し、この番号ｎのデータをステップＳ６３で（ｘ，ｙ）の位置からグラフィックデータとして表示する。そして、ステップＳ６４で手書き文字番号ｎの処理を終了する。

【００６０】

ステップＳ７１～ステップＳ７４も図５のステップＳ７０を指定文字数のデータを確認するという処理に適用したステップである。ステップＳ７１では、１文字分の共通の処理を行う。未処理の文字数Ｎを減らす（Ｎ－１）他、固有の処理として表示位置の変更、すなわちカーソルの移動をｘ＝ｘ＋ｄｙとして行う。ステップＳ７２では、文字の行方向の値が最大値max（＝ｘ１）より大きくなったか否かを判断する。ここで、大きいと判断すれば、ステップＳ７３に進み、行方向の最小値min（＝ｘ０）に戻り、列を一段下げる（ｙ＝ｙ＋ｄｙ）。すなわち、改行する。そして、ステップＳ７４に進む。一方、このステップＳ７２で文字の行方向の値が最大値max（＝ｘ１）以下であると判断した場合には、ステップＳ７３の処理を飛ばして、ステップＳ７４に進む。ステップＳ７４では、１文字分の共通の処理を終了する。

【００６１】

そして、ステップＳ８０の判断により未処理の文字Ｎがなくなるまで処理を繰り返し、ステップＳ９０で処理を終了すると、上記図７のような表示を文字表示部４に表示する。

【００６２】

ここで、ステップＳ６２で行われる手書き文字データベース６からの番号ｎのデータの取得について説明しておく。

【００６３】

先ず、手書き文字データベース５の具体例について図９を参照しながら説明する。この手書き文字データベース５の具体例は、手書き文字番号とグラフィックデータの先頭アドレスのテーブルを持っている。テーブルの最終行は、手書き文字番号、アドレスとも０とする。テーブルは、連続したメモリ上に実現され、１エントリー６バイト（手書き文字番号２バイト、先頭アドレス４バイト）とする。グラフィックデータ自体は、図１０に示すように、連続して配置されている場合も、連続していない場合も有り、文字番号順になっているとは限らない。

【００６４】

手書き文字データベース５から番号ｎのデータを取得するというステップＳ６２に示す処理は、図１１に示すように、ステップ６２－１～ステップＳ６２－６に分割できる。先ず、ステップＳ６２－１では、手書き文字データベース５の先頭のエントリーを読む準備をする。具体的には、手書き文字データベース５のテーブルアドレスの先頭adt1に着目する。次に、ステップＳ６２－２では、エントリーの手書き文字番号を読む。すなわち、着目したテーブルアドレスadt1から文字番号を次々と読んでいく。ステップＳ６２－３では、エントリーされている手書き文字番号（entry_num）と手書き入力された手書き文字の手書き番号ｎ（search_num）が等しくなったか否かを判断する。ここで、ＮＯと判断する、すなわちまだエントリーされている手書き文字番号（entry_num）と手書き入力された手書き文字の手書き番号ｎ（search_num）が一致していないと判断すると、ステップＳ６２－４に進み、次のエントリーの手書き文字番号（entry_num）を読む準備をする。具体的には、テーブルアドレスを４バイトづつ移動する。そして、ステップＳ６２－２に戻り、エントリーの手書き番号をさらに読む。このループは、ステップＳ６２－３でＹＥＳを判断するまで繰り返される。一方、ステップＳ６２－３でＹＥＳと判断する、すなわちエントリーの手書き文字番号と手書き入力された手書き文字の手書き番号ｎのデータが等しくなったと判断すると、ステップＳ６２－５に進み、エントリーの先頭アドレスを読む。そして、ステップＳ６２－６に進み図１０から手書き文字番号ｎのグラフィックデータを取得する。

【００６５】

次に、図８に示すステップＳ６３の処理、すなわち（ｘ，ｙ）の位置からグラフィックデータを表示するという処理の詳細を説明する。

【００６６】

タブレット２を使って手書き入力したデータを保存する方法として、最終的な絵の状態で保存するビットマップデータ方式と、書き順などの時間情報を含んだストロークデータ方式がある。ビットマップデータ方式で保存してある場合には、通常の文字のフォントのビットマップデータと同様に手書きフォントを扱えばいいので、ここではストロークデータのフォーマット例と、フローチャートを示す。

【００６７】

ストロークデータ方式で保存する場合の手書き入力によるパターンデータの具体例を図１２に示す。この図１２に記した（ａ），（ｂ），（ｃ），（ｄ），（ｅ），（ｆ），（ｇ）の各点は、手書き文字の例を示す図１３上の（ａ），（ｂ），（ｃ），（ｄ），（ｅ），（ｆ），（ｇ）の各点と対応している。また、図１２のendで区分した“０”，“１”は、一文字分のデータの終了を示すための１ビットのフラグであり、“０”は一文字分のデータの途中を、“１”は一文字分のデータの終了を示している。また、u/dで区分した“０”，“１”は、ペンがタブレット上にある（すなわち線を書いている）のか、タブレット２から離れている（移動している）のかを示す１ビットのフラグであり、“０”はタブレットから離れていることを、“１”はタブレット上にあることを示している。また、dx,dyで区分している７ビットの数値はそれぞれ水平方向、垂直方向のペンの移動量を示す。ここでは、１文字のデータの始めはペンの枠の左下の位置（ｓ）にあるように表現している。なお、文字の大きさや一文字分のデータ長等をヘッダーとしてつけることもできるが、ここではヘッダーの無い単純な例を示す。

【００６８】

（ｘ，ｙ）の位置から手書き文字を１文字表示するという図８のステップＳ６３に示す処理は、図１４に示すように、ステップＳ６３－１～ステップＳ６３－８に分割できる。先ず、ステップＳ６３－１では、グラフィックデータが無いか否かを判断する。ここで、グラフィックデータがあれば、ステップＳ６３－２に進み、無ければステップＳ６３－８に進む。ステップＳ６３－２では、２バイトのデータを読む。ステップＳ６３－３では、データの解釈を行う。すなわち、図１２に示したストロークデータのend，u/d，dx，dyを解釈する。ステップＳ６３－４では、u/dが０であるか否か、つまりペンがタブレット２から離れているか否かを判断する。ここで、ペンがタブレット２上にあると判断するとステ

ップＳ６３－５に進み、（ｘ，ｙ）から（ｄｘ，ｄｙ）分の線を描画する。一方、ペンがタブレット２から離れていると判断するとステップＳ６３－６に進む。ステップＳ６３－６では、ペンの位置を移動させる。そして、ステップＳ６３－７では、endが１であるか否か、つまり一文字分のデータが終了したか否かを判断する。ここで、一文字分のデータが終了したと判断するとステップＳ６３－８に進み、一文字分のデータが終了していない、つまり一文字分のデータの途中であると判断すると、ステップＳ６３－２に戻る。すなわち、endが１になるまでステップＳ６３－２からの処理を繰り返す。ステップＳ６３－８では、一文字分のグラフィクデータの表示処理を終了する。

【００６９】

以上、第１の実施の形態で、図６に示したフローにより指定文字数のデータのサイズを確認するという処理と、図８に示したフローにより指定文字数のデータを表示するという処理の具体例を説明した。

【００７０】

なお、この第１の実施の形態の変形例について説明しておく。図３に示した基本的なフローを含め、図５、図６及び図８にも示したステップＳ５０の手書き文字番号を読むという処理は、手書き文字番号を読み込む時にのみ行われるので、制御コード（もしくはそれに続くデータ）で手書き文字番号のバイト数（もしくはビット数）を示すようになっていれば、手書き文字番号はバイト数（もしくはビット数）を可変にすることもできる。つまり、手書き文字番号の符号化の際に、符号長を含めることも考えられる。

【００７１】

手書き文字番号が小さいものは少ないバイト数（もしくはビット数）で表現し、手書き文字番号が大きいものは多いバイト数（もしくはビット数）で表現することにより、データ量を抑えながら、扱える手書き文字数の制限をなくすことができる。

【００７２】

手書き文字番号にバイト数（もしくはビット数）の情報を含め、例えば１バイト表現の場合には数値表現そのもので、２バイト表現の場合には数値表現に２５６を足して扱うことにすれば、少ないバイト数で更に多くの手書き文字を扱うこともできる。

【００７３】

例えば、制御コードに続くデータで手書き文字番号のビット数を示す方法としては、手書き文字情報の部分を図１５に示すような形式とすることも考えられる。この場合、制御コードは一つで良く、Ｌを表現するためのビット数を決めておく必要がある。

【００７４】

具体的には、例えば８ビットでビット数（Ｌ）をそのまま表現する場合には、

【００７５】

【表１】

制御コード	情報長	手書き文字番号
XXXXXXXX XXXXXXXX	00001000	mmmmmmmm
XXXXXXXX XXXXXXXX	00010000	mmmmmmmm mmmmmmmm

【００７６】
のようになり、４ビットでＬ／４を表現する場合には、
【００７７】
【表２】

制御コード	情報長	手書き文字番号
XXXXXXXX XXXXXXXX	0001	mmmm
XXXXXXXX XXXXXXXX	0011	mmmm mmmmmmmm
XXXXXXXX XXXXXXXX	0101	mmmm mmmmmmmm mmmmmmmm

【００７８】
のようになる。
【００７９】
また、例えば制御コードに続くデータで手書き文字番号のビット数を示す他の方法としては、手書き文字情報の部分を図１６に示すような形式にすることも考えられる。この場合も、制御コードは一つで良く、Ｌを表現するためのビット数を決めておく必要がある。
【００８０】
具体的には、例えば８ビットでビット数（Ｌ）をそのまま表現する場合には、
【００８１】
【表３】

制御コード	情報長	手書き文字番号
XXXXXXXX XXXXXXXX	00010000	mmmmmmmm
XXXXXXXX XXXXXXXX	00011000	mmmmmmmm mmmmmmmm

【００８２】
のようになり、４ビットでＬ／４を表現する場合には、
【００８３】
【表４】

制御コード	情報長	手書き文字番号
XXXXXXXX XXXXXXXX	0010	mmmm
XXXXXXXX XXXXXXXX	0100	mmmm mmmmmmmm
XXXXXXXX XXXXXXXX	0110	mmmm mmmmmmmm mmmmmmmm

【００８４】
のようになる。
【００８５】
また、例えば制御コードに続くデータで手書き文字番号のビット数を示す他の方法としては、手書き文字情報の部分を図１７に示すような形式にすることも考えられる。この場合も、制御コードは一つで良く、Ｌを表現するためのビット数と制御コードのビット数を決めておく必要がある。
【００８６】
具体的には、例えば８ビットでビット数（Ｌ）をそのまま表現する場合には、
【００８７】
【表５】

制御コード	情報長	手書き文字番号
XXXXXXXX XXXXXXXX	00100000	mmmmmmmm
XXXXXXXX XXXXXXXX	00101000	mmmmmmmm mmmmmmmm

【００８８】
のようになり、４ビットでＬ／４を表現する場合には、
【００８９】
【表６】

制御コード	情報長	手書き文字番号
XXXXXXXX XXXXXXXX	0110	mmmm
XXXXXXXX XXXXXXXX	1000	mmmm mmmmmmmm
XXXXXXXX XXXXXXXX	1010	mmmm mmmmmmmm mmmmmmmm

【００９０】
のようになる。
【００９１】
また、例えば制御コードで、手書き文字番号のバイト数を示すには、例えば、文字コードが１６ビットの場合で手書き文字番号が１～４バイトの場合、具体的に、
【００９２】
【表７】

データ形成（ビット表現）
XXXXXXXXXXXXXX 00 mmmmmmmmm
XXXXXXXXXXXXXX 01 mmmmmmmmmmmmmmmmm
XXXXXXXXXXXXXX 10 mmmmmmmmmmmmmmmmmmmmmmm
XXXXXXXXXXXXXX 10 mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm

【００９３】
のようになる。この例では、通常の文字コードとして使われていないコードが連続して４つ必要になるが、手書き文字番号長毎に異なった制御コードを割り当てるだけなので、通常の文字コードとして使われていないコードが４つあれば、連続している必要はない。
【００９４】
また、表７に示したデータ形成のうち、最初の１４ビットが制御コードであり、次の２ビットを手書き文字番号の表現の一部とすることも考えられる。つまり、図３、図５、図６及び図８のステップＳ２０では制御コードと通常の文字コードのビット数を同じにしたが、通常の文字コードと制御コードとを確実に区別できることが重要なことであり、通常の文字コードと制御コードのビット数が同じか否かは重要でない。制御コード自体は通常の

文字コードであるか否かを判断するために必要な部分であり、通常の文字コードよりもビット数が少なくても構わない。

【００９５】

この場合の処理は、図１８に示すようになる。すなわち、ステップＳ３５で文字コードの不足分を読むという処理が追加される。

【００９６】

さらに、図３に示すフローのステップＳ４０とステップＳ６０の内容の両方にステップＳ７０の内容を含めることによって、図１９に示すように、ステップＳ７０を省略することもできる。この場合、ステップＳ１０で、common_funcを指定する必要が無くなる。

【００９７】

以上説明したように、この第１の実施の形態の文字情報処理装置によれば、制御コードと手書き文字番号をまとめて、通常の文字コードと同様に扱い、表示の際の処理としては対応するグラフィックデータ（パターンデータ）を通所の文字に対するフォントと同様に扱うことにより、手書き文字を文字列中に埋め込むことが可能になる。

【００９８】

また、ユーザインターフェースはもちろんプログラム処理でも移動、コピー等の文字列操作を通常の文字に対するのと同様に手書き文字に対して行うことができる。

【００９９】

また、グラフィックデータ自体を手書き文字として扱うのに比べ、コピーの時などに行う作業量を減らせる。

【０１００】

また、外字の様に使われていないコードを使うのとは異なり、文字コードと手書き文字番号は完全に独立して扱うことができるので、お互いに制限を受けることがない。

【０１０１】

このため、例えば手書き文字番号が１６ビットの場合、扱える手書き文字番号は65536文字となる。

【０１０２】

また、手書き文字番号のビット数を可変にし、制御コードに続く手書き文字番号のビット数を含めることにより、扱える手書き文字数を更に増やすことも容易にできる。これは既存コードの空き領域を利用する外字との大きな違いであり、制御コードを使った方法で取り扱える手書き文字数は、２４ビットで約1600万、３２ビットで約40億であり、文字数制限は事実上無いといえる。

【０１０３】

また、手書き文字データベースに各文字のデータ形式を示すフィールドを含めることにより、複数のデータ形式の手書き文字を混在して扱う場合でも、扱う文字のデータ形式を容易に知ることができ、異なった種類のデータ形式の手書き文字が混在していても適切に扱うことができる。

【０１０４】

また、複数の手書き文字データベースに登録された文字を混在させて使う場合には、同じ手書き文字番号が使われている可能性があるので、手書き文字指示制御コードを複数用意することにより対応することができる。

【０１０５】

また、参照する手書き文字データベースの数が増えてきた場合には、もう一段管理の階層を増やし、手書き文字データベースのデータベースを作ることによって対応することができる。

【０１０６】

また、手書き文字以外の音声文字等を追加する場合にも音声文字指示制御コード等を追加したり、拡張文字データベースのデータベースに手書き文字データベースや音声文字データベースを登録することによって容易に対応できる。

【０１０７】

次に、第２の実施の形態について図２０を参照しながら説明する。この第２の実施の形態の構成は、上記第１の実施の形態の構成とほぼ同一である。異なるのは、図１の文字列データ処理部４に対する文字列データ処理部１２の内部構成である。文字コード判定部８が文字の種類を共通処理部７に返し、その種類に応じて共通処理部７が適切な処理部（通常文字処理部９又は手書き文字処理部１０）にデータを渡す構成となっている。この場合には、文字の種類による分岐制御がある処理を実行することになる。文字コード判定部８でデータの種類を判断し、結果を共通処理部７に返し、通常文字処理部９又は手書き文字処理部１０で処理を行う。

【０１０８】

この図２０に示す第２の実施の形態の処理動作は、上述した図３、図５、図６及び図８等の各フローチャートのステップＳ３０の処理を若干変更することにより実行される。すなわち、文字コード判定部８が得た判定結果を共通処理部７に一回返して、この共通処理部７を介して通常文字処理部９又は手書き文字処理部１０を動作させればよい。

【０１０９】

なお、上記図１に示した第１の実施の形態及び上記図２０に示した上記第２の実施の形態の具体的ハードウェアを図２１に示す。差異となる文字列データ処理部４及び１２の構成は、特に区別していない。図１及び図２０で用いた符号に対応させて説明する。キーボード１は、キースイッチ群２０と、電源２１と、キーインターフェースからなる。タブレット２は、ペン２３と、タブレット板２４と、タブレットインターフェース２５からなる。文字表示部１１は、ディスプレイ部２６と、ディスプレイコントローラ２７と、Ｖ－ＲＡＭ２８とからなる。文字列データ処理部４及び１２は、ＣＰＵ２９と、ＲＯＭ３０と、ＲＡＭ３１とからなる。ここで、ＲＯＭ３０とＲＡＭ３１の一部は、文字列データ記憶部６にも使われる。また、ＲＡＭ３１の一部は、手書きデータベース５にも使われる。

【０１１０】

図２１には、マイク３２と、固体録音処理部３３と、スピーカ３４と、ダイアラー３５も記載している。入力音声を装置内部で文字コードして記憶する音声文字を扱うの場合には、マイク３２を文字入力部３に含めてもよい。ここで、マイク３２のような音声入力手段は、手書き文字以外の音声文字等を追加する場合に、音声文字指示制御コード等を追加したり、拡張文字データベースのデータベースに手書き文字データベースや音声文字データベースを登録することによって有効に使用できる。

【０１１１】

また、装置内部では文字コードを記憶しておき、音声合成等によりスピーカ３４から出力する音声文字も扱うような場合には、スピーカ３４等を文字出力部として用いてもよい。

【０１１２】

【発明の効果】

本発明に係る文字情報処理装置及び文字情報処理方法は、パターンデータを１文字分毎に分けて手書き文字指示制御コードに続く手書き文字番号として記録し、これと通常文字の文字コードとから文字列データを形成し、この文字列データの１文字毎に通常文字又は手書き文字であるかを判定し、それぞれの文字に応じたデータ処理を行い出力処理を行っているので、制御コードと手書き文字番号をまとめて、通常の文字コードと同様に扱い、表示の際の処理としては対応するパターンデータを通常の文字に対するフォントと同様に扱うことにより、手書き文字を文字列中に埋め込むことができる。また、通常の文字コードとは異なるコードで手書き文字を扱えるので、その数を増やすことができる。

【０１１３】

また、手書き文字番号のビット数を可変にし、制御コードに続く手書き文字番号のビット数を含めることにより、扱える手書き文字数を更に増やすことも容易にできる。

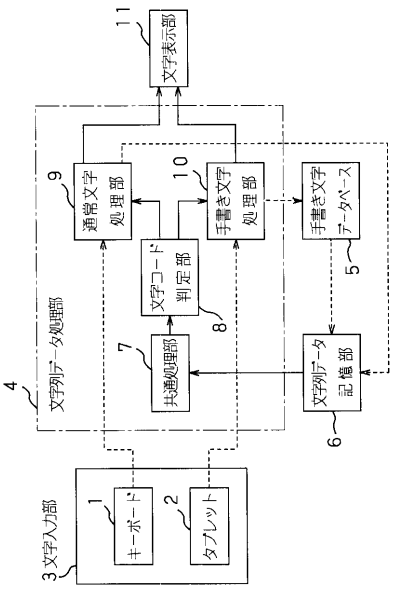
【０１１４】

また、手書き文字データベースに各文字のデータ形式を示すフィールドを含めることにより、複数のデータ形式の手書き文字を混在して扱う場合でも、扱う文字のデータ形式を容易に知ることができ、異なった種類のデータ形式の手書き文字が混在していても適切に扱

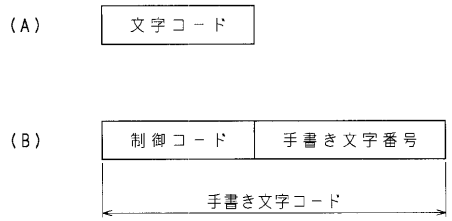
うことができる。
【図面の簡単な説明】
【図１】本発明に係る文字情報処理装置の第１の実施の形態のブロック図である。
【図２】予め規格化された標準文字コードと、文字に対応するパターンデータの手書き文字コードとを示す模式図である。
【図３】上記第１の実施の形態の基本的動作を説明するためのフローチャートである。
【図４】　通常文字と手書き文字が混在した文字列データの具体例を示す模式図である。
【図５】図４に示した文字列データの具体例を扱った場合の上記第１の実施の形態の動作を説明するためのフローチャートである。
【図６】指定文字数のデータのサイズを確認するという処理を上記第１の実施の形態に行わせた場合の動作を説明するためのフローチャートである。
【図７】指定文字数のデータを表示するという処理の概念を説明するための模式図である。
【図８】指定文字数のデータを表示するという処理を上記第１の実施の形態に行わせた場合の動作を説明するためのフローチャートである。
【図９】上記第１の実施の形態に使われる手書き文字データベースの具体例を示す模式図である。
【図１０】グラフィックデータの配置を示す模式図である。
【図１１】図８のフローチャートのステップＳ６２の処理を分割して説明するためのフローチャートである。
【図１２】ストロークデータ方式で保存する場合のデータの具体例を示す模式図である。
【図１３】手書き入力のパターンデータの具体例を示す模式図である。
【図１４】上記図８のフローチャートのステップＳ６３の処理を分割して説明するためのフローチャートである。
【図１５】制御コードに続くデータで手書き文字番号のビット数を示す方法を説明するための模式図である。
【図１６】制御コードに続くデータで手書き文字番号のビット数を示す他の方法を説明するための模式図である。
【図１７】制御コードに続くデータで手書き文字番号のビット数を示すさらに他の方法を説明するための模式図である。
【図１８】文字コードの不足分を読むという処理を追加した第１の実施の形態の動作を示すフローチャートである。
【図１９】１文字分の共通の処理を省略した第１の実施の形態の動作を示すフローチャートである。
【図２０】本発明に係る文字情報処理装置の第２の実施の形態のブロック図である。
【図２１】上記図１に示した第１の実施の形態及び上記図２０に示した上記第２の実施の形態の具体的ハードウェアを示すブロック図である。
【図２２】文字コードで取り扱う方法で行われていた従来の文字列処理を説明するためのフローチャートである。
【符号の説明】
１　キーボード
２　タブレット
３　文字入力部
４　文字列データ処理部
５　手書き文字データベース
６　文字列データ記憶部
７　共通処理部
８　文字コード判定部
９　通常文字処理部
１０　手書き文字処理部

１１　文字表示部

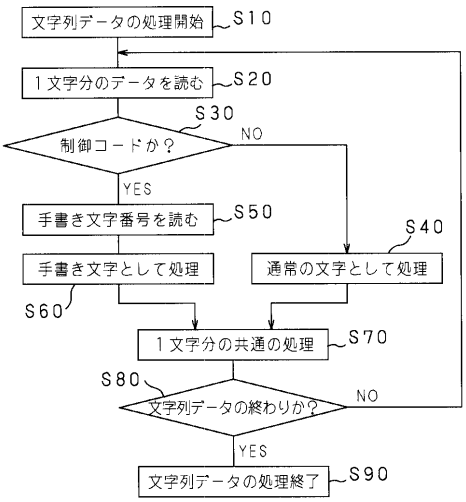
【図１】



【図２】



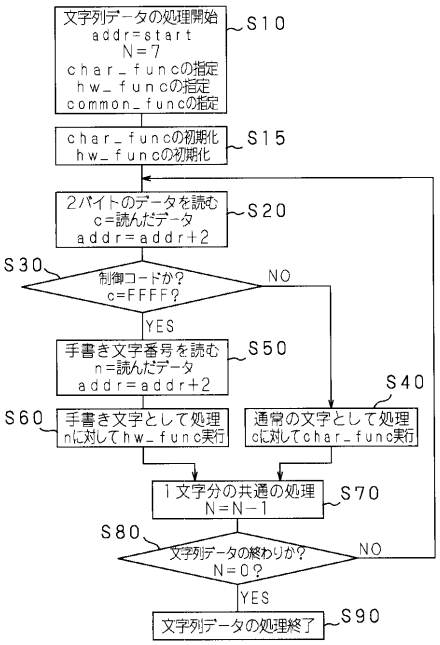
【図３】



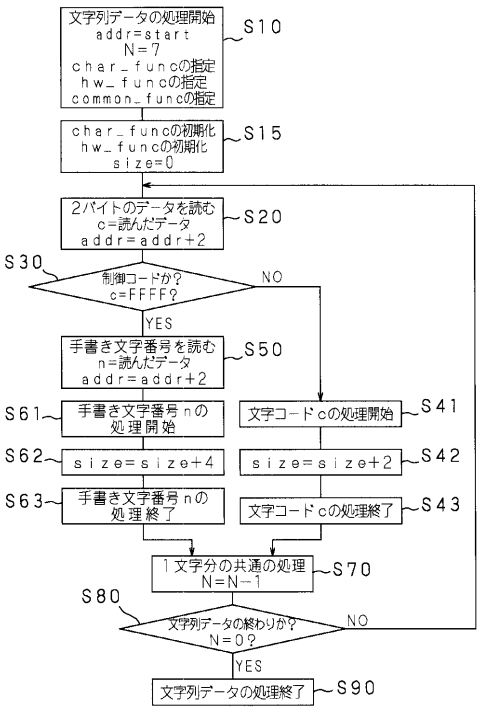
【図４】


「あ」
「い」
手書き＃1
「う」
手書き＃2
手書き＃3
「え」
"82A0"
"82A2"
"FFFF 0001"
"82A4"
"FFFF 0002"
"FFFF 0003"
"82A6"

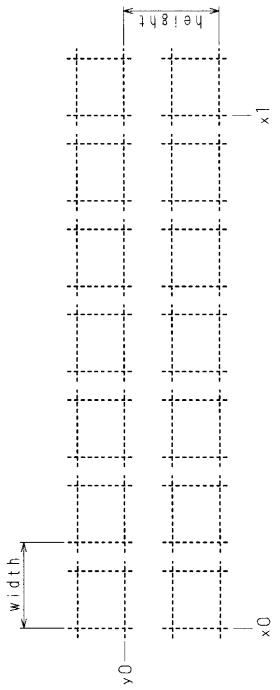
【図５】


文字列データの処理開始
addr=start
N=7
char_funcの指定
hw_funcの指定
common_funcの指定
S10
char_funcの初期化
hw_funcの初期化
S15
2バイトのデータを読む
c=読んだデータ
addr=addr+2
S20
S30
制御コードか?
c=FFFF?
NO
YES
手書き文字番号を読む
n=読んだデータ
addr=addr+2
S50
S40
S60
手書き文字として処理
nに対してhw_func実行
通常の文字として処理
cに対してchar_func実行
1文字分の共通の処理
N=N−1
S70
S80
文字列データの終わりか?
N=0?
NO
YES
文字列データの処理終了
S90

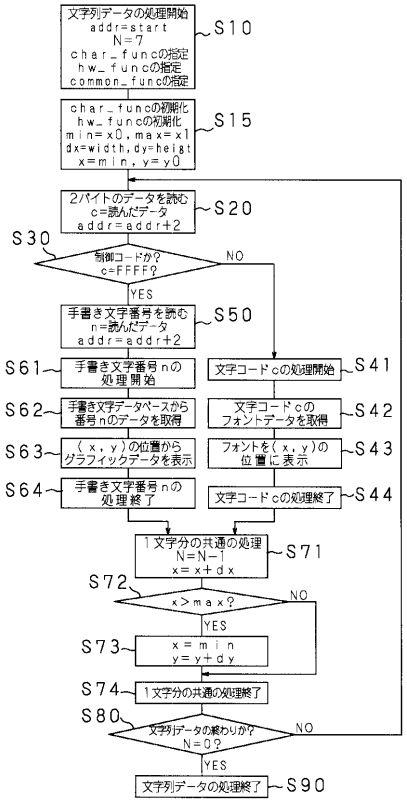
【図６】


文字列データの処理開始
addr=start
N=7
char_funcの指定
hw_funcの指定
common_funcの指定
S10
char_funcの初期化
hw_funcの初期化
size=0
S15
2バイトのデータを読む
c=読んだデータ
addr=addr+2
S20
S30
制御コードか?
c=FFFF?
NO
YES
手書き文字番号を読む
n=読んだデータ
addr=addr+2
S50
S61
手書き文字番号nの
処理開始
文字コードcの処理開始
S41
S62
size=size+4
size=size+2
S42
S63
手書き文字番号nの
処理終了
文字コードcの処理終了
S43
1文字分の共通の処理
N=N−1
S70
S80
文字列データの終わりか?
N=0?
NO
YES
文字列データの処理終了
S90

【図７】


height
x1
width
x0
y0

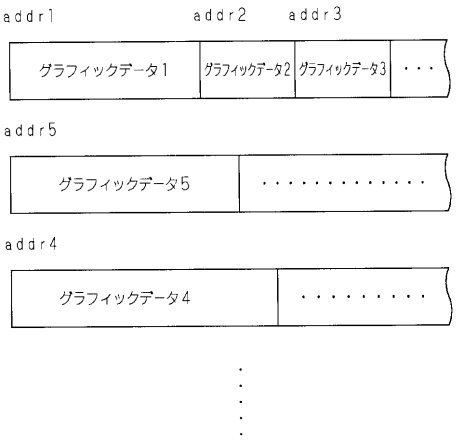
【図８】



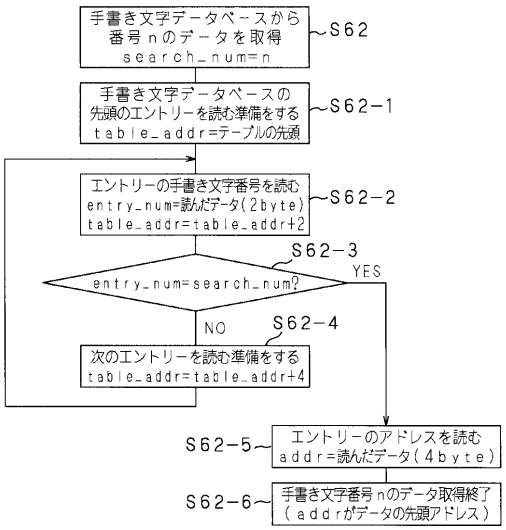
【図９】

テーブルアドレス	手書き文字番号	グラフィックデータの先頭アドレス
adt1	0001	addr1
adt2	0002	addr2
adt3	0003	addr3
adt4	0004	addr4
adt5	0005	addr5
・・・・・	・・・・・	・・・・・
0000	0000	0

【図１０】



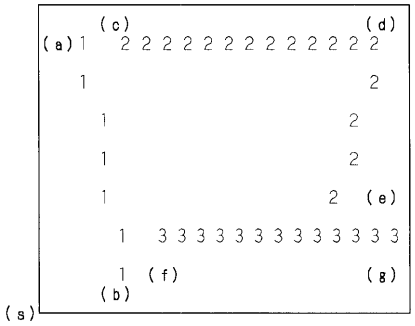
【図１１】



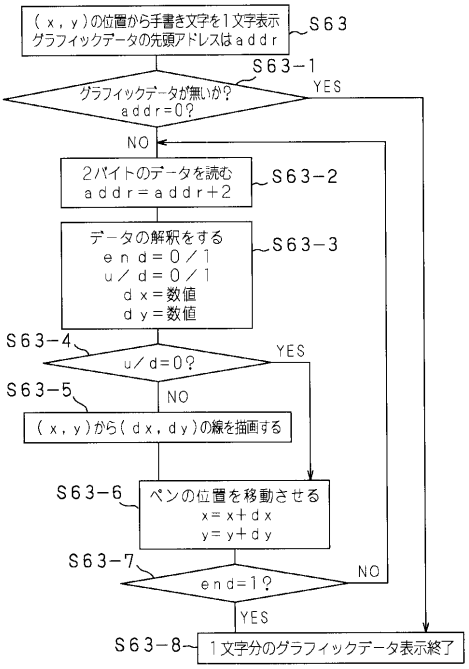
【図１２】

	end	u／d	dx	dy
(a)	0	0	0000011	1110000
(b)	0	1	0000010	0001100
(c)	0	0	0000000	1110100
(d)	0	1	0001100	0000000
(e)	0	1	1111110	0001000
(f)	0	0	1111000	0000010
(g)	1	1	0001011	0000000

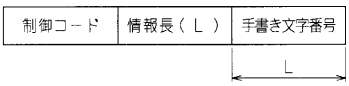
【図１３】



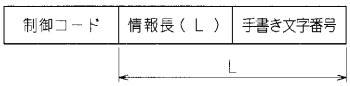
【図１４】



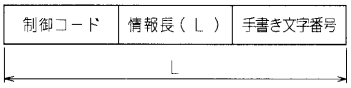
【図１５】



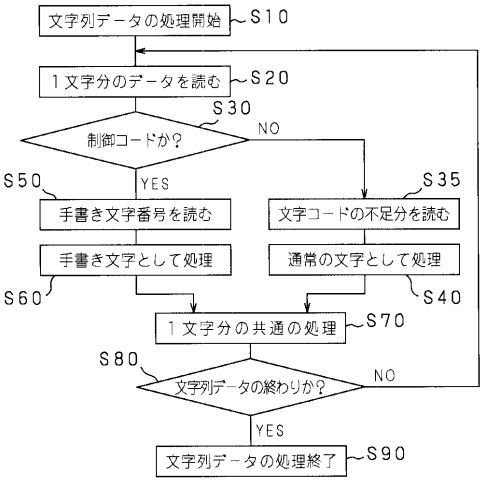
【図１６】



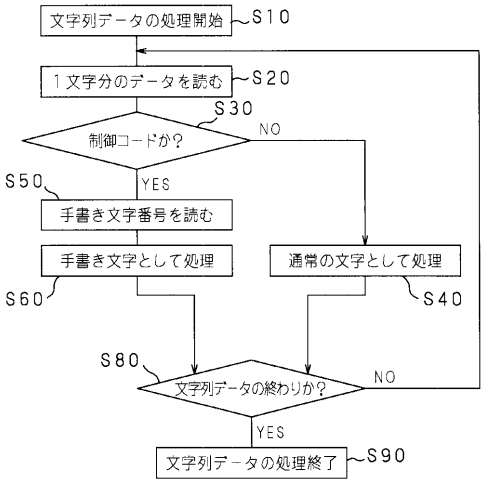
【図１７】



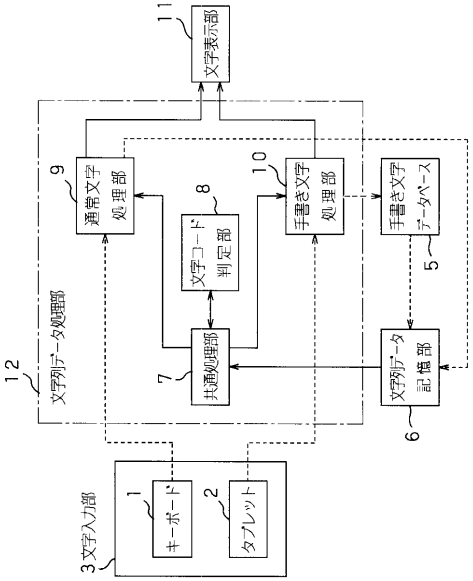
【図１８】



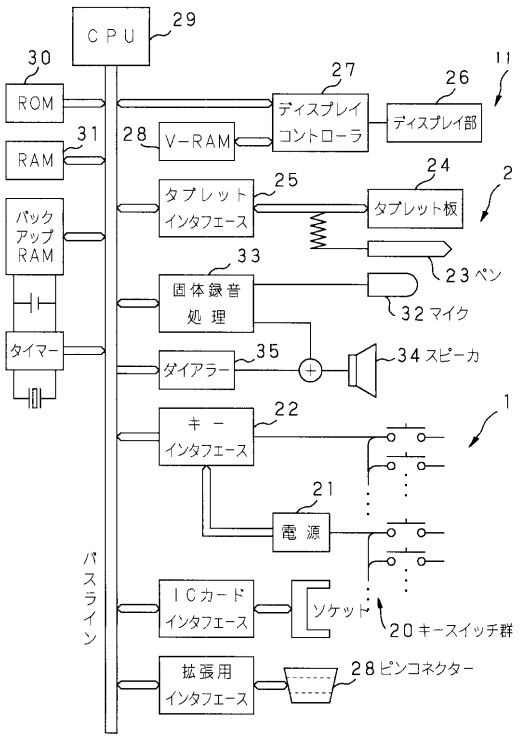
【図１９】



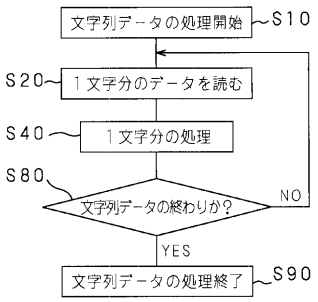
【図２０】



【図２１】



【図２２】



フロントページの続き

合議体
審判長　田口　英雄
審判官　飯田　清司
審判官　真木　健彦

(56)参考文献　特開平７－１７５５８１（ＪＰ，Ａ）
特開平５－１２８３２５（ＪＰ，Ａ）
特開平５－１５１２２０（ＪＰ，Ａ）
特開平８－２３５１１９（ＪＰ，Ａ）
特開平５－１７３５３６（ＪＰ，Ａ）
特開平４－９５９９７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 17/21-26
G06K 9/62