CH 666 973 A5

(19) SCHWEIZERISCHE EIDGENOSSENSCHAFT
BUNDESAMT FÜR GEISTIGES EIGENTUM

(11) CH 666 973 A5

(51) Int. Cl.[4]: G 06 F 15/40

Erfindungspatent für die Schweiz und Liechtenstein
Schweizerisch-liechtensteinischer Patentschutzvertrag vom 22. Dezember 1978

(12) PATENTSCHRIFT A5

(21) Gesuchsnummer: 2275/87

(62) Teilgesuch von: 5946/83

(22) Anmeldungsdatum: 03.11.1983

(30) Priorität(en): 03.11.1982 US 439210

(24) Patent erteilt: 31.08.1988

(45) Patentschrift veröffentlicht: 31.08.1988

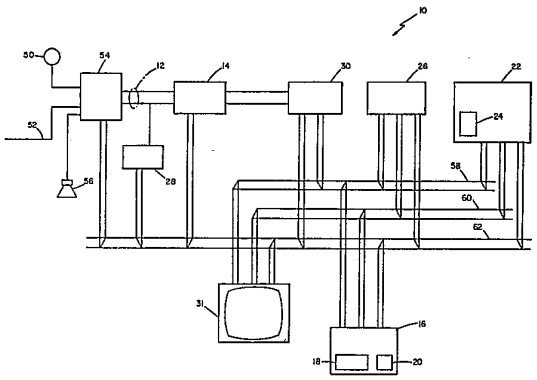
(73) Inhaber:
Wang Laboratories, Inc., Lowell/MA (US)

(72) Erfinder:
Stapleford, Gary N., Londonderry/NH (US)
Osborne, Deane C., Brookline/NH (US)

(74) Vertreter:
E. Blum & Co., Zürich

(54) Informationsverarbeitungsanlage für Dokumente mit geschriebenen und gesprochenen Bestandteilen.

(57) Die Anlage besitzt einen Speicher (22) zum Speichern von Dokumenten, Anzeigemittel (31) für Bestandteile eines Dokumentes, und Befehlseingabemittel (16). Sie Besitzt auch Organe (50, 54, 56, 28, 14, 30) zur Ein- und Ausgabe von akustischen Signalen und zur Umwandlung zwischen analogen und digitalen Darstellungen dieser Signale. Die Anlage ist eingerichtet, um Textbestandteile des Dokumentes als Text (92), und um gesprochene Bestandteile der Dokumente als Sprachsymbole (90) darzustellen. Letztere zeigen sowohl die zeitliche Lage wie auch die Dauer der gesprochenen Bestandteile relativ zum Text. Zum Bearbeiten einer bestimmten Stelle des Dokumentes kann diese durch Einrücken einer Markierung (94) in die passende Stellung, entweder der Text- oder der Sprachdarstellung, gekennzeichnet werden.



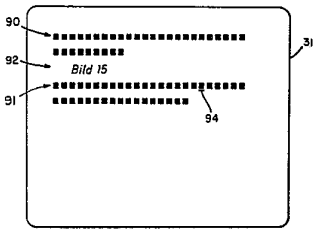


Bild 15

PATENTANSPRÜCHE

1. Informationsverarbeitungsanlage für Dokumente, die sowohl Text-Bestandteile wie gesprochene Bestandteile aufweisen, gekennzeichnet durch Dokumenten-Verarbeitungsmittel, die folgende Teile umfassen:

Anzeigemittel (31), zur Anzeige einer Darstellung eines Bestandteiles des Dokumentes,

eine in den Anzeigemitteln vorhandene, bewegliche Markierung (94), um eine, der augenblicklichen Stellung in diesem Bestandteil entsprechende Lage in der Darstellung zu markieren,

Befehlseingabemittel (16), zum Empfang eines Befehles für die Durchführung einer auf irgendeine Stelle des Dokumentes anwendbaren Operation, und

Befehls-Ausführungsmittel (26), um in Abhängigkeit des Befehles und der augenblicklichen Stellung die befohlene Operation an der durch die augenblickliche Stellung bestimmten Stelle des Dokumentes auszuführen.

2. Verarbeitungsanlage nach Anspruch 1, dadurch gekennzeichnet, dass die Befehlseingabemittel und die Befehls-Ausführungsmittel für die Eingabe bzw. die Durchführung von Änderungsoperationen eingerichtet sind.

3. Verarbeitungsanlage nach Anspruch 1, dadurch gekennzeichnet, dass die Befehls-Ausführungsmittel eingerichtet sind, um die Operation an einer Stelle innerhalb des Text-Bestandteiles oder des gesprochenen Bestandteiles durchzuführen, die in einem bestimmten Verhältnis zur augenblicklichen Stellung steht.

4. Verarbeitungsanlage nach Anspruch 2, dadurch gekennzeichnet, dass sie Mittel zur Eingabe von Text-Bestandteilen und von gesprochenen Bestandteilen aufweist, und dass die Befehls-Ausführungsmittel eingerichtet sind, um diese Bestandteile an der durch die augenblickliche Stellung bezeichneten Stelle des Dokumentes einzufügen.

5. Verarbeitungsanlage nach Anspruch 4, dadurch gekennzeichnet, dass die Befehls-Ausführungsmittel eingerichtet sind, um an einer durch die augenblickliche Stellung bezeichneten Stelle befindliche Bestandteile durch die eingegebenen Bestandteile zu ersetzen.

6. Verarbeitungsanlage nach Anspruch 2, dadurch gekennzeichnet, dass die Befehls-Ausführungsmittel eingerichtet sind, um an einer durch die augenblickliche Stellung bezeichneten Stelle befindliche Bestandteile zu löschen.

7. Verarbeitungsanlage nach Anspruch 2, dadurch gekennzeichnet, dass die Befehls-Ausführungsmittel eingerichtet sind, um durch die augenblickliche Stellung bestimmte Bestandteile von einer ersten Stelle an eine durch diese Stellung bestimmte zweite Stelle zu verschieben.

8. Verarbeitungsanlage nach Anspruch 2, dadurch gekennzeichnet, dass die Befehls-Ausführungsmittel eingerichtet sind, um durch die augenblickliche Stellung bestimmte Bestandteile in eine durch diese Stellung bestimmte Stelle zu kopieren.

BESCHREIBUNG

Die Erfindung bezieht sich auf eine Anlage zur Bearbeitung von Dokumenten, welche sowohl Text-Bestandteile, wie auch gesprochene Bestandteile aufweisen.

Um diese Bearbeitung besonders bequem zu gestalten, ist die Erfindung wie in Anspruch 1 beschrieben, definiert.

Es liefert die Erfindung dem Autor eines Textes eine sichtbare Darstellung des Aufbaues seines Diktates, mitsamt Anzeigen, welche er bezüglich Paragrapheneinteilung oder anderer Dinge einfügen mag. Sie gestattet dem Autor sein Diktat in sehr flexibler Weise abzuändern: verschieben, einsetzen, auslöschen, und rückspielen bei gleichzeitiger Beobachtung der Anzeige. Sie hilft ihm sein Korrekturlesen zu überschauen und genau zu bestimmen, wo Änderungen anzubringen sind. Ausserdem kann die Erfindung dem Autor auch gestatten, mittels einer Tastatur Zwischenbemerkungen und Instruktionen in sein Diktat einzufügen.

Fig. 1 zeigt ein Blockdiagramm einer Ausführungsform der Erfindung.

Fig. 2 zeigt die Darstellung eines Dokumentes auf der Anzeige-Einheit 31.

Die erfindungsgemässe Anlage 10 umfasst Anschlüsse 12, um kontinuierlich variierende elektrische Signale, welche gesprochenen Texten entsprechen, zu empfangen und abzugeben. Ein empfangenes Signal kann von einem Mikrophon 50 oder einer Telephonleitung 52 über die Schnittstellenschaltung 54 empfangen werden, wie in der Figur beispielsweise gezeigt, dies kann aber auch auf andere Weise geschehen. Das gelieferte Signal kann benützt werden um einen Lautsprecher 56 zu betreiben, oder auf andere Weise. Die Anschlüsse 12 sind mit einem Analogdigitalwandler 14 verbunden, welcher in beiden Richtungen arbeitet. Der Wandler 14 seinerseits ist mit einem Serie-Parallelwandler 30 verbunden, der in beiden Richtungen arbeitet. Ein akustischer Sensor 28 ist mit den Anschlüssen 12 verbunden und erzeugt ein Kontrollsignal, welches davon abhängt, ob der Textempfangskanal aktiv ist oder nicht. Die Anlage 10 umfasst noch eine Anzeigeeinheit 31, welche vorzugsweise eine Kathodenstrahlröhre enthält, sowie eine Tastatur 16, die einen Sektor 18 für die Eingabe numerischer Zeichen, sowie einen Sektor 20, für die Eingabe von Textverarbeitungs- und Steuersignalen aufweist.

Die Anlage 10 umfasst auch einen Processor 26, welcher der von Zilog hergestellte Typ Z-80 sein kann, sowie einen Speicher 22 zum Speichern von Daten in Bitform, welcher Speicher einen Teil 24 aufweist, der ein darin gespeichertes Steuerprogramm enthält. Alle Elemente der soeben beschriebenen Anlage werden durch eine Datensammelschiene 58, eine Adress-Sammelschiene 60, und Steuerleitungen 62 verbunden, wie in der Figur gezeigt. Alle Elemente der oben beschriebenen Anlage 10 sind handelsübliche Gegenstände, und es ist dem Fachmann in der Textverarbeitung wohlbekannt, wie sie gegenseitig verbunden sein müssen.

Zusammen mit dem Processor 26 steuert das im Speicher gespeicherte Sprachbearbeitungsprogramm die Funktion der Anlage zur Durchführung aller Textverarbeitungsschritte. Wenn ein die Anlage benutzender Autor in ein Mikrophon spricht, wird der von der Anlage als analoges Signal empfangende Text digitalisiert und in diskreter Form in den Speicher eingelesen. Gleichzeitig wird, unter Verwendung einer Reihe von Sprech-Kennzeichen, von denen jedes eine Sekunde des gesprochenen Textes darstellt, eine Darstellung des gesprochenen Textes erzeugt und auf dem Bildschirm gezeigt. Um eine Vergeudung der Speicherkapazität zu vermeiden, wird während der Sprachpausen das Einlesen von Daten unterdrückt. Es kann der Autor gleichzeitig mit dem Diktieren Unterbruchssignale mittels einer Klaviatur eingeben, welche Signale im Speicher Zeiger erzeugen, die angeben, an welcher Stelle des Datenflusses die Eingabe durchgeführt wurde. Dadurch werden nachfolgende Sprech-Kennzeichen am Anfang der nächsten Zeile gezeigt, was eine paragraphenähnliche Einteilung bewirkt. Gleichzeitig wird eine Randziffer erzeugt, um eine bequeme Identifikation des Unterbruches zu ermöglichen. Es kann der Autor auch das Diktieren mittels eines über die Klaviatur eingegebenen Signals unterbrechen, und alphanumerischen Text via die Klaviatur eingegeben. Dieser Text wird im Speicher aufgenommen und auf dem Bildschirm gezeigt.

Die durch das Programm gesteuerte Anlage stellt eine Liste auf, die eine vereinheitlichte Sequenz von Sprechdaten, Textdaten, und Unterbruchskennzeichnungen enthält. Ursprünglich entspricht die Reihenfolge dieser Sequenz der zeitlichen Reihenfolge in welcher die Daten durch die Anlage aufgenommen

wurden. Die Anlage erzeugt auch einen Zeiger im Speicher, der eine Stellung innerhalb der Datensequenz hervorhebt. Eine Markierung zeigt die entsprechende Stelle in der visuellen Anzeige an. Der Autor kann den Zeiger und die Markierung, welche gegenseitig verbunden sind, bewegen, um irgendeinen bestimmten Punkt innerhalb der gemeinsamen Datensequenz zu bezeichnen. Unter Verwendung der Markierung und der Textverarbeitungsbefehle der Klaviatur, inklusive «Einsetzen», «Löschen», «Ersetzen», «Bewegen», und «Kopieren», kann der Autor all diese Textverarbeitungsschritte durchführen, und sie dabei genau gleich benützen, ob es sich nun um Sprechdaten, um Text oder um Kennzeichen handelt. Die visuelle Anzeige in den Anzeigemitteln wiederspiegelt alle durchgeführten Textverarbeitungsschritte. Es kann der Autor auch, unter Verwendung der Markierung und von über die Klaviatur eingegebenen Signalen, das Überspielen des gesprochenen Textes auf irgendein angeschlossenes akustisches Gerät bewirken.

Im folgenden wird eine ausführlichere Beschreibung der Wirkungsweise des Programms gegeben.

Im Speicher 22 ist ein Programm zur Steuerung des Sprechverarbeitungsprogramms gespeichert, welches zusammen mit dem Processor 26 die Funktion der Anlage in bezug auf alle Sprachverarbeitungsfunktionen steuert. Es benützt das Programm zur Verarbeitung gesprochener Texte eine aus Routinen gebildete Warteschlange, und vom Sprachverarbeiter aufgerufene Subroutinen werden zunächst in die Routinen-Warteschlange eingeordnet, und in der Folge dann durchgeführt, wenn der Processor sie erreicht. Bei solch einer Warteschlange setzt ein Unterbruchsverarbeiter eine Subroutine in die Schlange ein, um den Unterbruch zu bearbeiten, und aktiviert sofort danach die Unterbrüche und die Rückkehrvorgänge. Die Subroutinen werden in die Schlange eingereiht, und werden zu gegebener Zeit durch den Processor behandelt. Ein Modul für Routineschlangen enthält Subroutinen zum Bearbeiten der Routinen-Warteschlange des Sprachverarbeiters. Es sind diese:

RTNQUEINIT:
Initialisieren der Routine-Warteschlange.

RTNQUEPUSH:
Fügt die Adresse und ein Adressparameter einer Prozedur einer Routineschlange an.

RTNQUERUN:
Prüft, ob ein Paar Prozedur/Parameter ansteht. Wenn ja, wird die Prozedur aufgerufen, und der einzelne Adressparameter durchgegeben.

Der Hauptfluss des Programms zur Sprachbearbeitung ist wegen der Sprachbearbeitungs-Routinen-Warteschlangen recht einfach. Der Hauptfluss des Sprachbearbeiters führt zwei Funktionen durch: 1) Er ruft eine Initialisierungsroutine, voice$editor$init, auf, um alle durch den Sprachverarbeiter gebrauchten Datenstrukturen sowie festverdrahtete Ein- Ausgangsgeräte zu initialisieren. 2) Er geht dann in eine Endlosschleife, und ruft dabei RTNQUERUN auf, um etwaige Subroutinen in der Subroutinenschlange durchzuführen. Falls der Benützer beispielsweise bekanntgibt, dass er aus dem Sprachverarbeiter aussteigen will, wird die Prozedur EXIT$EDITOR an die Routineschlange angefügt. Der Processor ruft diese Routine auf sobald er kann, wodurch der Sprachverarbeiter in die aufrufende Funktion zurückkehrt.

Aus obiger Beschreibung ergibt sich, dass nach dem Einsteigen in den Sprachverarbeiter und dem Initialisieren der variablen und der fest verdrahteten Schaltungen, der Sprachverarbeiter sich in einer Endlosschleife befindet und wartet, bis etwas in der Routine-Warteschlange auftaucht. Es werden Unterbruchsprozeduren verwendet, um etwas an diese Schlange anzufügen. Wenn ein Hardware-Unterbruch stattfindet, werden Unterbruchsprozeduren durchgeführt. Falls dies geschieht schaltet der Processor Unterbrüche aus, schiebt die laufende Programmzählung auf das Stack und springt zu einer Prozedur, die den Unterbruch bearbeitet.

Der Sprachverarbeiter arbeitet im Unterbruchsmodus 2 des Z 80, und erhält Unterbruchsbefehl von den folgenden Geräten, welche in der Reihenfolge ihrer Unterbruchspriorität aufgezählt sind.

1. CTC Kanal 0
   Blockzählung — dieser Kanal erzeugt einen Unterbruch wenn die akustische Schaltung gerade mit der Aufnahme oder dem Abspielen eines Puffers mit digitalisierten Audiosignalen fertig ist.
2. CTC Kanal 1
   Telephonanruf — dieser Kanal erzeugt einen Unterbruch jedesmal wenn das Telephon klingelt.
3. CTC Kanal 2
   Anschlag — der Sprachverarbeiter programmiert diesen Kanal, so dass er jedesmal dann einen Unterbruch erzeugt, wenn ein Anschlag der Tastatur empfangen wird.
4. CTC Kanal 3
   Takter — der Sprachverarbeiter programmiert diesen Kanal, um alle 10 Millisekunden (0,010 Sekunden) einen Unterbruch zu erzeugen.

Die Adressen der Unterbruchsverarbeiter für obige Geräte sind im Gedächtnis in einer Vectortabelle für die Unterbrüche enthalten. Wenn irgendeines der obigen Geräte einen Unterbruch erzeugt, wird die entsprechende Adresse in der Tabelle der Unterbruchsvectoren aufgerufen.

Es befinden sich die Unterbruchsverarbeiter in zwei Einheiten, der Unterbruchseinheit und der Eingang- Ausgangeinheit.

Es ist die Unterbruchseinheit einfach ein Bündel von Routinen auf Assemblerniveau, nämlich eine für jedes der unterbrechenden Geräte. Sie retten alle die Register auf dem Stack, rufen eine PLM Prozedur auf, und stellen dann die Register wieder her, aktivieren die Unterbrüche und kehren zurück. Die Verarbeiter sind:

Audio:
CTC Kanal 0 Verarbeiter, ruft die PLM Prozedur AUDIO$INTERUPT.

Klingel:
CTC Kanal 1 Verarbeiter, ruft die PLM Prozedur RING$INTERUPT auf.

KEYHNDLR:
CTC Kanal 2 Verarbeiter, führt ein IN (00) durch um eingegebene Tastaturanschläge zu erhalten, speichert diese in einer variablen RAWKEY, und ruft die PLM Prozedur GOT$KEY auf.

Takter:
CTC Kanal 3 Verarbeiter, ruft die PLM Prozedur TENMSTIMER auf.

Es enthalten die Ein- Ausgangsverarbeitungsmodule PLM-Prozeduren, welche den Grossteil der Unterbruchsverarbeitung besorgen. Zudem sind darin noch diverse weitere Routinen enthalten. Die Unterbruchsroutinen sind nachstehend kurz beschrieben:

RING$INTERUPT:
Fügt an die Warteschlange eine Prozedur an, welche folgenden Text zur Anzeige bringen wird «Ihr Telephon klingelt, bitte drücken Sie TAB».

GOT$KEY:
Wird meist nur die Prozedur KEY$DISPATCH an die Warteschlange anschliessen. Der Tastaturanschlag selbst wird von KEY$DISPATCH behandelt.

TENMSTIMER:

Ruft andere PLM Prozeduren auf, was die periodische Prüfung gewisser Bedingungen bewirkt.

Sozusagen alle Funktionen des Sprachverarbeiters werden initialisiert wenn der Benützer eine Taste drückt. Der Sprachverarbeiter verwendet einen tabellengesteuerten Mechanismus um zu beschliessen, welche Prozedur in Beantwortung des Anschlages einer gewissen Taste aufgerufen werden soll.

Die Tasten der Arbeitsstelle sind in 16 verschiedene Klassen aufgeteilt. Jeder Klasse ist eine Zahl zwischen 0 und 15 zugeordnet. Keine Taste kann in mehr als einer Klasse liegen. Die Nummern der Klassen und die Tasten jeder Klasse sind im folgenden aufgezählt.

Klassen-Nummer	Beschreibung	Tasten
1	Aufnahme	INSERT
2	Stop	STOP
3	Start/Stop	Abstand, (HOME)
4	Schirm-Markierung	Markierung nach: Norden, Osten, Süden, Westen
5	Sprünge	Sprung auf Seite
6	Zahlen	0 bis 9
7	Text	A - Z, a - z, Komma, Punkt, ! # $ % ¢ & * ( ) – = +] [; : ' " / ?
8	Rückschalten	Rücktaste
9	Kennzeichen	Wagenrücklauf, Anmerkung
10	Umnumerieren	
11	Abändern	ELETE, REPLIC, MOVE, COPY
12	Ausführen	EXECUTE
13	Löschen	CANCEL
14	Bedienungshilfe	COMMAND, (HELP)
15	Telephon	TAB
0	Ungültig	Alle anderen Tasten

Es besteht eine Übersetzungstabelle, die ursprünglich verdrahtete Tastencode in die entsprechende Klassennummer (0 bis 15) übersetzt. Diese Tabelle befindet sich im Sektor 0 der Datei «VOICE.CLASSTBL» Der Sektor 1 dieser Datei enthält die standardmässige Übersetzungstabelle für pre-WISCII Tastatur. Es muss bemerkt werden, dass die Klassentabelle von der Umschaltung unabhängig ist. So liegen beispielsweise sowohl CANCEL wie SHIFT CANCEL in der Lösch-Klasse (13). Dies berührt jedoch nicht die gross- und kleingeschriebenen Buchstaben, da sich beide in der Text-Klasse (7) befinden.

Es ist der Verarbeiter in verschiedene Funktionszustände aufgeteilt. Die Tasten können je nach dem laufenden Zustand verschiedene Bedeutung haben, und daher ist für jeden Zustand eine Prozedurtabelle definiert. Diese Prozedurtabellen werden Zustandstabellen genannt. Die Zustandstabellen sind im Modul für Zustandstabellen definiert.

Die Zustandstabellen des Sprachbearbeiters enthalten Indexe für eine grosse Prozedurtabelle. Diese Tabelle kann in einem mit 36 Eingängen versehenen Modul der Routinetabelle gefunden werden.

Beim ersten Aufruf des Verarbeiters ist der Hauptzustand der laufende Funktionszustand. Wenn neue Funktionszustände wirksam werden, dann werden die alten Zustände zusammenmit einem Index der jeweiligen Lage der Markierung auf dem Bildschirm, auf einem Zustands-Stack abgelegt. Nehmen wir beispielsweise an, dass der Benützer die Löschtaste drückt, während sich die Anlage im Hauptzustand befindet. Der Hauptzustand wird dann auf den Zustands-Stack abgelegt, und der Zustand zur Definition von Segmenten wird zum laufenden Zustand. Die Aufforderung «Was Löschen?» erscheint auf dem Bildschirm.

Nehmen wir jetzt an, der Benützer drücke die Taste «Sprung auf Seite». Dann wird der Zustand zur Definition von Segmenten auf den Stack abgelegt, und die Aufforderung wird auch auf den Zustands-Stack abgelegt. Der neue Zustand ist der Sprung Zustand. Die Aufforderung «Sprung Wohin» erscheint auf dem Schirm. Dann gibt der Benützer eine Zahl ein und drückt die Taste EXECUTE. Es wird eine Prozedur für das Springen auf die Zahl aufgerufen.

In diesem Augenblick werden der Zustand zur Definition eines Segmentes und die Aufforderung vom Stack heruntergeschoben. Es wird die Aufforderung «Was Löschen?» wieder auf dem Schirm erscheinen. Der Benützer drückt die Taste EXECUTE, und es wird eine Prozedur zum Löschen des bezeichneten Teiles der Sprechdatei aufgerufen. Darnach wird der Hauptzustand vom Stack geschoben, und wir befinden uns wieder im ursprünglichen Funktionszustand.

Zusätzlich zu den Zustandstabellen selbst enthält das Modul der Zustandstabellen auch noch Prozeduren zur Bearbeitung des Zustands-Stack. Diese Prozeduren sind:

INIT$STATE:

Zustands-Stack initialisieren.

NEW$STATE:

Legt den alten Zustand auf dem Stack ab, und macht den genannten Zustand zum laufenden Zustand.

POP$STATE:

Schiebt einen Zustand vom Stack, wodurch er zum laufenden Zustand wird.

Es enthält das Modul für Zustandstabellen eine Routine welche, wenn eine Klassennummer gegeben ist, die Adresse der Prozedur liefert, die dem laufenden Zustand dieser Klasse entspricht:

ROUTINE$ADDR:

Eine Klasse sei gegeben, dann sucht diese Prozedur in der laufenden Zustandstabelle die Adresse der Prozedur, welche dieser Klasse entspricht.

Der Entscheid eine bestimmte Prozedur aufzurufen ist folgendermassen zusammengefasst:

1) Unterbruch durch Tastenanschlag
2) KEYHNDLR rettet Register, legt den fest verdrahteten Tastencode in der variablen RAWKEY fest, und ruft den GOT$KEY auf.
3) GOT$KEY führt folgendes durch:
   a) Abbruch bei tödlichem Irrtum.
   b) Rohe Ausgabe falls SHIFT$PAGE eingetippt wurde.
   c) Falls die vorhergehende Taste noch nicht verarbeitet wurde, wird diese ausgelassen.
   d) Adresse der Prozedur KEY$DISPATCH zusammen mit dem Parameter RAWKEY an die Routine-Warteschlange anfügen.
4) KEY$DISPATCH wird aus der Routineschlange entnommen und ausgeführt, unter Durchführung des folgenden:
   a) Übersetzung der Tastenbezeichnung, unter Verwendung der Übersetzungstabelle.
   b) Finden der Klassennummer für diese Taste, unter Verwendung der Klassentabelle.
   c) Falls das höchste Bit der Klassennummer Null ist, einklinken dieses Anschlages.

d) Löschen etwaiger Fehlanzeige.
e) Ausschalten der Akustik, ausgenommen für die RETURN und die Ein/Aus-Klasse.
f) Aufruf von ROUTINE$ADDR, und Übergabe der Klasse an dieselbe, um die Adresse der zu erreichenden Prozedur zu erhalten.
g) Anfügen der Adresse dieser Prozedur und der übersetzten Tastenbezeichnung in die Routineschlange.

5) Es wird die geeignete Routine zusammen mit der übersetzten Tastenbezeichnung aus der Warteschlange entnommen und durchgeführt.

Weitere Prozeduren können grob in zwei Teile geteilt werden. Es gibt für jede Datenstruktur Module der unteren Stufe, welche Operationen an diesen Strukturen vollführen. Typische Module der tiefen Stufe sind die Dateiindexe (akustische Index, Kennzeichentabelle, Anmerkungstabelle), akustische Funktionen, und der Bildschirm.

Den zweiten Teil bilden die Routinen der höheren Stufe. Diese Prozeduren werden in der Regel durch den Tastenanschlag-Weiterleitungsmechanismus aufgerufen (ihre Adressen befinden sich in der Routinentabelle) und rufen ihrerseits die Routinen dere niedrigeren Stufe auf, welche den Grossteil der Arbeit machen. Sie können daher als eine Schnittstelle zwischen den Routinen zur Bearbeitung der Tastenanschläge und den Routinen der unteren Stufe betrachtet werden.

Das Modul der Benützerschnittstelle (V:voice.rrr.plm.ve.userint) enthält Prozeduren der höheren Stufe für akustische Verarbeitung, das Markieren von Abschnitten und das Umnumerieren:

PLAY$STOP:
Er wird immer dann aufgerufen, wenn eine Taste der Ein/Aus-Klasse betätigt wird. Wenn die akustische Bearbeitung in diesem Augenblick ausgeschaltet ist, bewegt die Prozedur die Markierung an den Beginn des nächsten Akustikabschnittes und beginnt mit dem Abspielen. Falls die akustische Bearbeitung in diesem Augenblick aufnimmt oder abspielt hält die Prozedur die akustische Verarbeitung an.

INSERT$MARK:
Wird aufgerufen, wenn eine Taste der Kennzeichenklasse angeschlagen wird. Falls eine Abschnittskennzeichnung eingegeben wurde, bestimmt die Prozedur ihre genaue Stellung auf dem Bildschirm und ruft das Modul mit der passenden Fensterroutine auf, um die Markierung einzutragen. Falls die NOTE Taste gedrückt wurde, prüft die Routine ob sich die Markierung in diesem Augenblick auf einer Fussnote befindet. Falls nicht, wird eine solche geschaffen. In beiden Fällen wird in den Textmodus gegangen.

RENUMBER:
Wird aufgerufen, wenn eine Taste der Umnumerierungsklasse gedrückt ist. Der Verarbeiter wird in den Umnumerierungszustand gebracht, und die Aufforderung «Umnumerierungsmarken?» wird angezeigt.

REN$EXECUTE:
Wird aufgerufen, wenn im Umnumerierungszustand die Taste EXECUTE gedrückt wird. Es wird eine Tabellenprozedur zum Umnumerieren der Marken aufgerufen, der Bildschirm neu aufgefüllt und der vorhergehende Zustand vom Stack abgerufen.

REN$CANCEL:
Wird aufgerufen, wenn im Umnumerierungszustand die Taste CANCEL gedrückt wird. Dies ruft den vorhergehenden Zustand vom Stack ab.

Das Rücktastenmodul führt das Rückschalten durch. Ein Druck auf die Rückschaltetaste bewirkt dass die Markierung um fünf Sekunden zurückversetzt wird und dass während fünf Sekunden abgespielt wird. Einmaliges Drücken bewirkt dass die Markierung fünfmal N Sekunden zurückversetzt wird, und dass die gleiche Zeitlänge abgespielt wird. Während des Abspielens stoppt der Druck auf irgendeine Taste das Abspielen, wodurch die Rückschaltfunktion völlig annulliert wird. Wenn die Rückschalttaste gedrückt ist, vergehen 350 Millisekunden bevor mit dem Abspielen begonnen wird. Dadurch wird dem Benützer Zeit gelassen, die Rückschalttaste mehrmals zu drücken, bevor das Abspielen beginnt. Das Rücktastenmodul verwendet zur Durchführung dieser Funktionen drei Variablen:

bs$mode
Ist WAHR wenn zurückgetastet wird, und andernfalls FALSCH.

bs$time
Die Zeit der Markierung, in dem Augenblick wo der Benützer zuerst die Rücktaste drückt. Gleichgültig wie oft die Taste gedrückt wird, wird bis zu dieser Stelle abgespielt, und nicht weiter.

bs$play$cnt
Ein durch die tenmstimer abwärts geschalterer Zähler. Wird verwendet um die Wartezeit von 350 Millisekunden abzuzählen.

Die Rückschaltfunktion liefert die folgenden Prozeduren:

BS:
Wird aufgerufen, wenn die Rückschalttaste gedrückt wird. Wenn sie zum ersten Mal gedrückt wird, wird das bs$mode auf WAHR gesetzt, und bs$time gespeichert. Es wird bs$wait$time auf 350 Millisekunden initialisiert.

BS$WAIT$COUNTER:
Wird alle zehn Millisekunden durch TENMSTIMER aufgerufen. Diese Prozedur erniedrigt bs$wait$time und nachdem 350 Millisekunden abgelaufen sind, fügt es an die Warteschlange eine Routine an, die von der laufenden Stellung der Markierung bis zu bs$time abspielen wird.

BSKEYCHECK:
Diese, von KEY$DISPATCH aufgerufene Prozedur annuliert den Rückschaltmodus wenn eine andere Taste als die Rückschalttaste gedrückt wird.

Das Markierungsmodul setzt alle Markierungsfunktionen der oberen Stufe. Es sind diese Prozeduren wiederum nur Schnittstellen zwischen der Tastenbetätigungsverarbeitung und den Bildschirmroutinen, welche die Markierung effektiv auf dem Bildschirm bewegen.

CURSOR$RTN:
Wird in den meisten Zuständen dann aufgerufen, wenn eine Taste der Markierungsklasse gedrückt wird. Sie ruft nur eine der vier Bildschirmroutinen auf, in Abhängigkeit davon, welche Markierungstaste gedrückt wurde.

GOTORTN:
Wird bei Druck auf die GO TO PAGE Taste aufgerufen. Schiebt den alten Zustand auf das Stack, und bewirkt dass der laufende Zustand der «go to» Zustand ist. Zeigt die Aufforderung «gehe wohin?» und positioniert die Markierung direkt hinter der Aufforderung. Es ist zu bemerken, dass bei Übersetzung eines Dateitextes diese Markierung am rechten Ende ausgerichtet sein sollte.

GOTOEXIT:
Diese Prozedur wird aufgerufen, wenn CANCEL im GOTOSTATE gedrückt wird. Sie setzt die Markierung in den Audioteil des Bildschirms zurück, und ruft den vorhergehenden Zustand vom Stack ab.

GOTOCURSOR:
Wird aufgerufen, wenn eine der Markierungstasten im

«go to» Zustand gedrückt wird. Sie ruft eine der vier Bildschirmroutinen auf, in Abhängigkeit davon, welche Markierungstaste gedrückt wurde. Danach ruft sie GOTOEXIT auf, um in den vorhergehenden Zustand zurückzukehren.

GOTOACCEPT$NUM:
Wird aufgerufen, wenn im «go to» Zustand eine Taste der Zahlenklasse betätigt wird. Diese Prozedur zeigt auf dem Schirm, gleich hinter der Markierung, die Zahl an und setzt die Stellung der Markierung neu.

GOTOEXECUTE:
Wird aufgerufen, wenn im GOTOSTATE die Taste EXECUTE gedrückt wird. Wenn sich auf dem Schirm eine Zahl befindet, wird diese aus der ASCII-Kodierung in die binäre Form übersetzt, und es wird eine Bildschirmroutine aufgerufen um die Markierung unter das passende Zeichen zusetzen. Danach ruft die Prozedur GOTOEXIT auf, um in den vorhergehenden Zustand zurückzukehren.

Das Modul zur Texteingabe enthält Routinen zum Eingeben von Anmerkungen, während man sich im Textmodus befindet.

Es werden die folgenden Variablen verwendet:

text$buffer (60)
Puffer zum Aufbewahren der Anmerkung während ihrer Eingabe.

tindex
Laufende Stellung (0-59) im Textpuffer.

tcursor
Gegenwärtige Stellung der Markierung auf dem Bildschirm.

note$index
Index innerhalb der Anmerkungstabelle zur Bestimmung der im Augenblick bearbeiteten Textanmerkung.

first
Eine Flagge, WAHR wenn die eingegebene Anmerkung gerade erstellt wurde. Ist dies der Fall, dann wird diese Anmerkung gelöscht falls CANCEL gedrückt wird. Falls es eine alte, in Abänderung befindliche Anmerkung ist, dann bewirkt CANCEL nur, dass die Anmerkung in ihrer ursprünglichen Form erhalten bleibt.

Es werden die folgenden Routinen ausgeführt:

TEXTSETFIRST:
Wird durch INSERT$MARK aufgerufen, um dem Texteingabemodul mitzuteilen, dass diese Anmerkung gerade eingefügt wurde.

TEXT$MODE$ENTER:
Wird durch INSERT$MARK aufgerufen, wenn die Anmerkungstaste gedrückt wird. Dadurch wird der alte Zustand abgeschoben und ein neuer Textzustand kreiert. Die Aufforderung «Text Eingeben» wird angezeigt. Es wird die Anmerkung aus der Anmerkungstabelle erfasst und in den Textpuffer abgelegt.

TXT$CANCEL:
Wird aufgerufen, wenn im Textzustand die Taste CANCEL gedrückt wird. Falls man daran war eine neue Anmerkung einzugeben, wird diese Anmerkung gelöscht. Andernfalls wird der Textpuffer annulliert, und der Schirm wird mit der alten, unveränderten Anmerkung beschickt. Es wird der vorhergehende Zustand wieder eingesetzt.

TEXT$EXECUTE:
Wird aufgerufen, wenn im Textzustand die Taste EXECUTE gedrückt wird. Ersetzt die alte Anmerkung durch den Inhalt des Textpuffers. Stellt den vorhergehenden Zustand wieder her.

TEXT$CURSOR:
Wird aufgerufen, wenn im Textzustand eine Markierungstaste gedrückt wird. Bewegt die Markierung vorwärts oder rückwärts. Zeigt eine Fehlmeldung an, wenn an die Taste Markierung nach Norden oder Markierung nach Süden gedrückt wird.

TXT$BACK$SPACE:
Wird aufgerufen, wenn im Textzustand die Rückschalttaste gedrückt wird. Bewegt die Markierung um eine Stelle zurück und löscht das Zeichen unter dem sie sich befindet.

TXT$ENTRY:
Wird aufgerufen, wenn eine Taste in der Text-, Zahlen- oder Ein/Aus-Klasse gedrückt wird. Er liest das Zeichen in den Textpuffer und auf den Schirm ein, und bewegt die Markierung um eine Stelle vorwärts.

TEXT:
Wird aufgerufen, wenn im «Hauptzustand» eine Texttaste gedrückt wird. Falls die Markierung sich auf einer Anmerkung befindet, wird in den Textmodus gegangen und die gedrückte Taste wird in den Textpuffer eingelesen, und auf dem Schirm angezeigt. Falls sich die Markierung nicht in einer Anmerkung befindet, wird die Bemerkung «Markierung Bewegen» angezeigt.

Das Verarbeitungsmodul liefert eine Schnittstelle zwischen dem Tasteneingabe-Mechanismus und den Schirm und Dateiindex Routinen der niederen Stufe, welche die eigentliche Dateiverarbeitung besorgen.

Das Verarbeitungsmodul merkt sich, welche Teile der Datei verarbeitet werden. Es wird eine Punktstruktur verwendet, um Stellungen in der Datei zu charakterisieren. Diese Struktur hat die folgende Form:

Punkt-Struktur
(Zeit Adresse, Index Byte)
wobei die Zeit die in der Datei abgelaufene Zeit ist, und wobei Index der Markierungsindex der laufenden Marke, oder falls sich keine Marke an dieser Stelle befindet, diejenige der nächsten Markierung in der Datei ist.

Es werden die folgenden Punkt-Strukturen verwendet, um während der Verarbeitung einzelne Stellen festzuhalten:

begpoint
Anfang eines Segmentes, das gelöscht/bewegt/kopiert werden soll.

endpoint
Ende eines Segmentes, das gelöscht/bewegt/kopiert werden soll.

destpoint
Zielpunkt für ein Bewegen/Kopieren.

Um einen Teil der Datei zu löschen wird das Segment zwischen begpoint und endpoint (inklusive) aus der Datei entfernt:

Um einen Teil der Datei zu bewegen oder zu kopieren wird das Segment zwischen begpoint und endpoint (inklusive) nach destpoint bewegt:

Beim Einsetzen in die Datei wird despoint zum Einsetzpunkt. Das augenblickliche Ende der Datei in begpoint wird aufgenommen und am Ende der Datei gestartet:

Wenn der Benützer STOP drückt, benimmt sich das Programm wie oben beschrieben, und bewegt das durch (begpoint, endpoint) begrenzte Segment nach destpoint.

Um ein Segment der Datei zu ersetzen werden drei weitere Punktstrukturen verwendet:

rbegpoint
Enthält den Anfang des zu löschenden Segmentes.

rendpoint
Enthält das Ende des zu löschenden Segmentes.

rbegpoint
Enthält den Anfang des einzusetzenden Segmentes.

Der Ersatzvorgang arbeitet wie folgt: Zunächst wird der zu ersetzende Abschnitt zwischen begpoint und endpoint bestimmt. Nachdem der Abschnitt definiert ist, wird begpoint nach rdestpoint und endpoint nach rendpoint kopiert, und es wird rbegpoint an das Ende der Datei gesetzt. Danach wird die übliche Einsetzprozedur durchgeführt, und am Ende der Datei aufgenommen. Falls STOP gedrückt ist, wird, wie beim Einsetzen, das neue Material, nämlich der Abschnitt (begpoint, endpoint) zur Einsetzstelle (destpoint) bewegt, womit das Einsetzen durchgeführt ist. Während des Ersetzens kann der Benützer einsetzen, abspielen, die Markierungstasten betätigen, und Abschnittsmarken sowie Textanmerkungen eingeben. Alle Einschübe werden in üblicher Weise durchgeführt, unter Verwendung von begpoint, endpoint und destpoint. Es sind natürlich alle Einschübe auf jenseits von rbegpoint beschränkt.

Falls der Benützer CANCEL drückt, wird das Ersetzen annulliert indem das Ende der Sprechdateizeit auf rbegpoint gesetzt wird, so dass die ursprüngliche Form der Datei wieder hergestellt ist.

Falls der Benützer die Taste EXECUTE drückt, wird der Ersatz so durchgeführt, dass zuerst der Abschnitt (rdestpoint, rendpoint) gelöscht wird, und danach rdestpoint auf destpoint und das Ende der Datei auf endpoint gesetzt wird, wonach das Einsetzen durch normale Verschiebung des Abschnittes (begpoint, endpoint) nach destpoint erreicht wird.

Das Modul für akustische Funktionen enthält Routinen zum Aufnehmen und Abspielen in und von Sprechdateien. Es benützt ein zugehöriges Modul, das Ein/Aus Modul, welches Datenstrukturen und Prozeduren enthält, die die Puffer und Warteschlange-Anfragen an die Steuerung verarbeitet.

Wenn aufgenommen oder abgespielt wird, müssen akustische Daten gepuffert werden, so dass das Abspielen oder Aufnehmen nicht dadurch unterbrochen wird, dass auf einen Ein- oder Auslesevorgang eines Puffers gewartet werden muss. Die Software der Stimmverarbeitungseinheit ist für den Gebrauch von mindestens zwei Puffern ausgelegt, es können jedoch mehr Puffer verwendet werden, wenn es der Platz erlaubt. Zurzeit verwendet die Stimmverarbeitungseinheit sechs akustische Puffer.

Der Stimmverarbeiter verwendet Puffer welche eine Länge von 1 bis 16 Sektoren haben. Diese Puffer sind seitenweise im Speicher angeordnet. Jeder Puffer entspricht einem akustischen Block in der Stimmdatei. Das Ein/Ausgangsmodul enthält als Informationsstrukturen bezeichnete Strukturen, welche die akustischen Puffer steuern. Das Ein/Ausgangsmodul enthält eine Ein/Ausgangs-Warteschlange, welche verwendet wird, um RCBs aufzureihen. Der Zehnmillisekunden-Takter prüft diese Schlange alle 10 Millisekunden. Falls sich etwas in der Schlange befindet, wird die Takterprozedur selbst die Abfrage aus der Schlange entnehmen und sie der Steuerung vorzeigen.

Es verwendet die Ein/Aus Anfrageschlange die folgenden Datenstrukturen:

queue
Eine Tabelle von Adressen, dies ist die Ein/Aus Anfrageschlange.

top
Index des Oberteils der Schlange.

bottom
Index des unteren Endes der Schlange.

count
Die Anzahl Elemente in der Schlange.

Die folgenden Routinen verarbeiten die Warteschlange:

IO$PUSH:
Schiebt die Adresse eines RCB auf die Ein/Aus Anfrageschlange.

POPANDSEND:
Falls sich etwas in der Schlange befindet und das SCA klar ist, wird die RCB Adresse aus der Schlange entnommen und in das SCA eingelesen. Diese Prozedur wird immer dann eingelesen, wenn man etwas zum ersten Mal an die Warteschlange anfügt (versucht es sofort wieder abzurufen). Es wird ausserdem alle zehn Millisekunden durch die Prozedur TENMSTIMER aufgerufen.

Da der Stimmverarbeiter nur aufgenommene Daten einfügt, wird er nicht durchstreichen, und die Aufnahme wird immer am Ende der Datei beginnen. Eingesetzte Daten werden am Ende der Datei aufgenommen, und dann zur Einsetzstelle verschoben.

Für die Aufnahme werden die folgenden Schritte durchgeführt:

1) Start mit der sechsten Informationsstruktur.
   a) Einsetzen der ersten Pufferadresse.
   b) Einsetzen der Puffergrösse.
   c) Wenn im letzten Block der Datei eingesetzt wird, wird die Stop Flagge gesetzt.
2) Zuführen der Adresse des ersten Puffers an die Hardware.
3) Der Hardware den Anfang der Aufnahme zu befehlen.
4) Durchführen der folgenden Prozedur:
   a) Der Hardware die Grösse des Puffers mitteilen, in welchen sie momentan einliest.
   b) Eine Einleseanfrage für den vorhergehenden Puffer in die Schlange anfügen, falls dies nicht der erste Puffer ist.
   c) Wenn für diesen Puffer die Stop Flagge gesetzt ist, dann stoppen.
   d) Prüfen um sicherzustellen, dass alle vorangehenden Anfragen für Einlesen in diesen Puffer befriedigt wurden, und widrigenfalls die akustische Verarbeitung unterbrechen bis die Anfragen behandelt wurden.
   e) das RCB für diesen Puffer auffüllen.
   f) die Variablen so erhöhen, dass die Bereitschaft zum Verarbeiten des nächsten Puffers hergestellt ist.

Nachdem die Hardware das Einlesen in den ersten Puffer beendet hat, wird ein Blockzählungsunterbruch erzeugt (CTC Kanal 0). Wenn dies geschieht, wird die Prozedur AUDIO$ INTERUPT aufgerufen. Diese Prozedur prüft ob ein Abspiel- oder ein Aufnahme-Modus in Kraft ist, und ruft eine Abspiel- oder Aufnahme-Unterbruchsprozedur auf. Der obige Schritt 4 ist die Aufnahme-Unterbruchsprozedur RECORD$INTERUPT. Bei fortschreitender Aufnahme wird sie jedesmal aufgerufen, wenn ein Puffer abgefertigt ist.

Das Abspielen spricht der Aufnahme. Es wird einige Initialisierung durchgeführt, und dann der Hardware mitgeteilt, dass das Abspielen begonnen werden soll. Es wird sofort die PLAY$ INTERUPT Routine aufgerufen. So wie jeder Puffer abgespielt ist, wird PLAY$INTERUPT wieder aufgerufen, um den nächsten Puffer für das Abspielen vorzubereiten, und eine Anfrage in die Warteschlange einzufügen, um einen weiteren Puffer von der Speicherplatte abzulesen.

Bei der Aufnahme wird die Abtastfrequenz immer entsprechend SMP$RATE gesetzt, welches die Abtastfrequenz bestimmt. Während des Abspielens kann jedoch die Abtastrate geändert werden. Alle zehn Millisekunden wird die Prozedur SET$RATE durch den TEN$MS$TIMER aufgerufen. Diese Prozedur ruft eine Routine auf, um die laufende Einstellung der Geschwindigkeitskontrolle auf den passenden Wert der Abtastfrequenz zu setzen. Es wird dann der Hardware der Wert dieser Abtastfrequenz zugeführt.

Der Bildschirm des Stimmverarbeiters ist in zwei Teile geteilt, den Zustandsteil und den Akustikmarkierungsteil. Der Zustandsteil besteht aus den ersten zwei und der letzten Zeile des Schirmes. Diese Stellen werden zur Anzeige von Aufforderungen, der Markierungszeit, der Länge usw. verwendet. Der Akustikmarkierungsteil besteht aus den Zeilen 3 bis 21 und wird verwendet um den Inhalt der Stimmdatei, d.h. des akustischen Blocks, Textanmerkungen und der Abschnittsmarken, anzuzeigen.

Das Anzeigemodul steuert den Zustandsteil des Bildschirmes. Zusätzlich befinden sich in diesem Modul alle MENUPACK Prozeduren. Es enthält Prozeduren zum Initialisieren des folgenden: menupack, Anzeige der Markierungszeit, akustischer Modus, Hilfsanzeigen, Telephonmodus, Titel, Aufforderungen, Länge und Fehlmeldungen.

Das Fenstermodul enthält Routinen zum Anzeigen und zum Auffrischen des Akustikmarkierungsteiles des Schirmes. Dieses Modul wird durch folgende Module unterstützt:

Umformen	(V:voice.rrr.plm.ve. convert)	Lage-Änderungsroutinen
Zeit	(V:voice.rrr.plm.ve. time)	Zeit- und Lage-Änderungsroutinen
Zeile	(V:voice.rrr.plm.ve. line)	Zeilenformat-Bearbeitung
Ort	(V:voice.rrr.plm.ve. region)	Index-Aufsuchen
Blättern	(V:voice.rrr.plm.ve. scroll)	Fenster-Bearbeitung der unteren Stufe

Die Stimmdatei umfasst einen Kopf, eine Markierungstabelle, eine Anmerkungstabelle, einen Sektor-Plan und einen Block-Plan. Die folgenden Module enthalten Module zum Aufrufen der Stimmdatei:

fileindx	(V:voice.rrr.plm.ve. fileindx)	Datei-Index aufstellen
editindx	(V:voice.rrr.plm.ve. editindx)	Datei-Index abändern
mark	(V:voice.rrr.plm.ve. mark)	Marken-Tabelle aufstellen
note	(V:voice.rrr.plm.ve. note)	Anmerkungs-Tabelle aufstellen
voicegrm	(V:voice.rrr.plm.ve. voicegrm)	Stimmdatei : Routinen für Aufstellung, Initialisieren und Ausputzen
extend	(V:voice.rrr.plm.ve. extend)	Erweitern und Beschneiden der Stimmdateien
fatal	Tödlicher Fehler, ABEND-Verarbeitung	

Das Fehlermodul enthält Prozeduren für ABEND, tödliche und nicht tödliche Fehler. Eine Flagge, DUMPFLAG, die in die Verbindung gesetzt ist, wird verwendet um zu bestimmen, ob ein Fehler zu einem Rohausdruck führt oder nicht. Wenn DUMPFLAG auf OFFh steht, dann werden Rohausdrucke aktiviert. Wenn sie auf 0 steht, dann werden Rohausdrucke unterdrückt.

Die ausgeführten Prozeduren sind:

NON$FATAL$ERROR:
Listet aus wenn die Flagge gesetzt ist, zeigt einen VE Fehler an: XXX, wobei XXX eine durchgegebene Fehlernummer ist. Diese Fehlernummern werden in (V:voice.rrr.lit.ve.ERR) definiert. Zeigt auch einen 16 Byte langen Datenabschnitt (in der Regel an RCB) wenn es als Parameter durchgegeben wird.

INFORM$ERROR:
Zeigt eine nicht-VE-Fehlermeldung an; beim Drücken irgendeiner Taste kehrt es zum aufzurufenden Teil zurück. Nicht-VE Fehlermeldungen sind einfach die Standardfehler, wie etwa «Markierung Bewegen», welche im unteren Teil des Bildschirmes angezeigt werden. Diese werden in (V:voice.rrr.lit.ve.MERROR) definiert.

FATAL$ERROR:
Ist mit NON$FATAL$ERROR identisch, ausser dass dieser Fehler nicht korrigierbar ist. Nachdem der Benützer irgendeine Taste gedrückt hat, kehrt der Verarbeiter zum aufzurufenden Teil zurück.

Der Wiederherstellungsprozess des Stimmverarbeiters wird bei Netzunterbruch oder zufälligen IPLs während des Aufnahmevorganges eine Wiederherstellung bewirken. Es verwendet der Stimmverarbeiter einige gemeinsame Datenstrukturen, und es enthalten drei Module Routinen, um diese Strukturen zu verarbeiten.

Die Routine-Warteschlange verwendet folgende Prozeduren:

QUE$INIT:
Diese Prozedur definiert eine Warteschlange. Der Benützer gibt die Adresse der Schlange, die Grösse der Schlange, die Grösse jedes Elementes darin, und einen Zeiger an, der auf eine Struktur hinweist, die alle wesentlichen Charakteristiken der Schlange enthält. Diese Struktur identifiziert die Schlange. Sie muss als Parameter den Einfüge- und Abrufroutinen zugeführt werden, die weiter unten beschrieben sind.

QUE$PUSH:
Diese Prozedur fügt ein Element an eine bestimmte Schlange ein.

QUE$POP:
Diese Prozedur ruft ein Element vom Kopf einer bestimmten Schlange ab.

Das Stackmodul (V:voice.rrr.plm.ve.stack) ist eine Stack-Ausführung mit Einfüge- und Abrufroutinen. Es verwendet das Stack des Zustandstabellenmoduls Prozeduren aus dem Stackmodul, um den Zustandsstack aufzustellen. Im Gegensatz zum Warteschlangenmodul, können die Routinen des Stackmoduls nur auf ein bestimmtes Stack einwirken, welches folgendermassen im Modul bestimmt ist:

stack (12)
Byte Das ist der für das Stack reservierte Raum.

sp
Der Zeiger des Stacks.

Zwei Routinen bearbeiten das Stack:

PUSH:
Legt ein Element auf das Stack ab.

POP:
Ruft ein Element vom Stack ab.

Es kann das Modul für den Bitplan (V:voice.rrr.plm.ve.bit) in einer durch den Benützer bestimmten Bitaufzeichnung Bits setzen, löschen und testen. Die Aufzeichnung kann nicht mehr als 256 Bytes umfassen. Die Tabelle der Marken verwendet eine Bitaufzeichnung, um die Nummer der nächsten zu schaffenden Abschnittsmarke zu bestimmen. Das Modul für die Verarbeitung der Dateiindexe verwendet eine Bitaufzeichnung, um alle freien Blocks im Index so zu ordnen, dass die Dateierweiterungen optimal durchgeführt werden. Das Modul für Bitaufzeichnungen enthält folgende Prozeduren:

BIT$SET:
Setzt ein Bit in einer Bitaufzeichnung.

BIT$CLR:

Löscht ein Bit in einer Bitaufzeichnung.

BIT$TEST:

Prüft ein Bit um zu sehen, ob es gesetzt oder gelöscht ist.

Es sind alle PLM INPUT und OUTPUT Bestimmungen für den Stimmverarbeiter im Hardware-Modul für akustische Steuerung (V:voice.rrr.plm.ve.audioctl) enthalten. Dieses Modul enthält kleine Prozeduren welche als Schnittstelle zwischen der Hardware und dem Hauptteil des PLM Code des Stimmverarbeiters wirken.

Das Modul zum Setzen des Unterbruchsmoduls (V:voice.rrr.z80.ve.setimode) enthält zwei Prozeduren, eine um die Anlage in den Unterbrechungsmodus 2 zu setzen, und die andere um sie in den Unterbrechungsmodus 0 zurückzusetzen. Die PLM Routinen INIT$WORKSTATION und RESET$WORKSTATION, welche sich im Modul für akustische Steuerung befinden, rufen die zwei Routinen im Modul für das Setzen von Unterbrechungsmoden auf. Es enthalten die allerersten Bytes dieses Moduls die Unterbrechungstabellen für das CTC und das PIO. Diese Tabellen müssen im Speicher auf einer Grenze liegen, die durch einen Faktor 8 bestimmt ist, und es muss daher in dem Zusammenstellungsplan dafür Sorge getragen werden, dass dies der Fall ist.

666 973
1 Blatt *

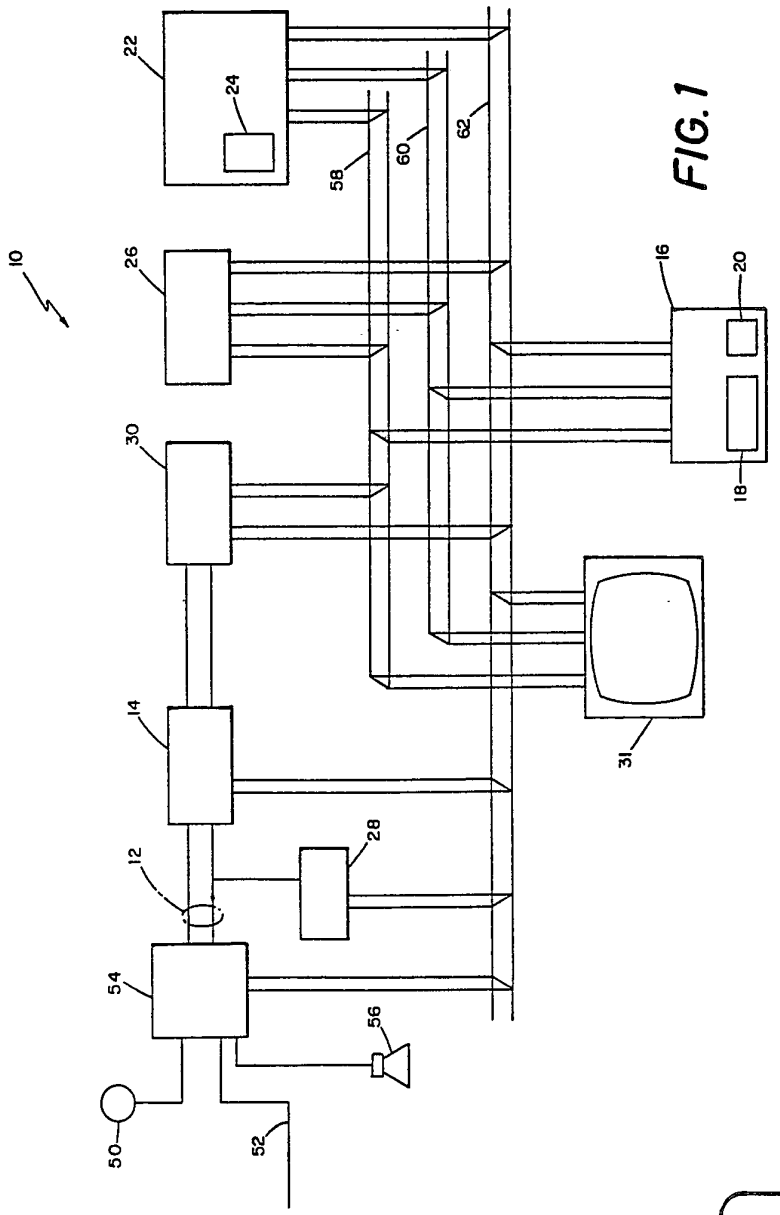


FIG. 1

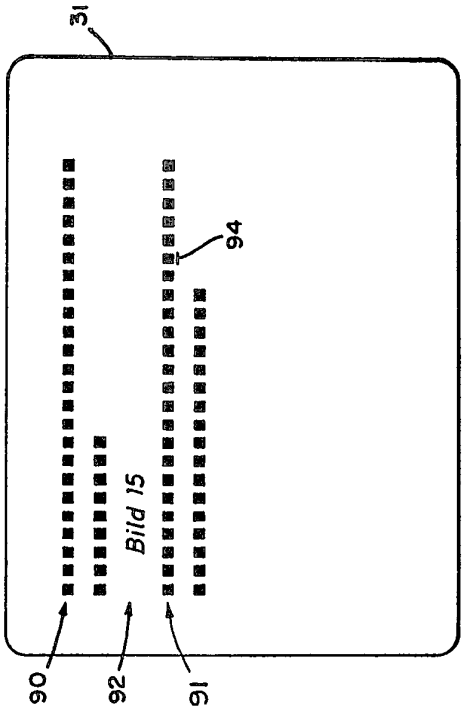


FIG. 2