(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 693 32 889 T2** 2004.03.04

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 0 706 692 B1**
(21) Deutsches Aktenzeichen: **693 32 889.4**
(86) PCT-Aktenzeichen: **PCT/AU93/00552**
(96) Europäisches Aktenzeichen: **93 923 424.1**
(87) PCT-Veröffentlichungs-Nr.: **WO 94/010657**
(86) PCT-Anmeldetag: **26.10.1993**
(87) Veröffentlichungstag der PCT-Anmeldung: **11.05.1994**
(97) Erstveröffentlichung durch das EPA: **17.04.1996**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **16.04.2003**
(47) Veröffentlichungstag im Patentblatt: **04.03.2004**

(51) Int Cl.[7]: **G07F 7/10**
G06K 19/073

(30) Unionspriorität:
PL552092 26.10.1992 AU

(84) Benannte Vertragsstaaten:
AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LI, LU, MC, NL, PT, SE

(73) Patentinhaber:
Intellect Australia Pty. Ltd., Bentley, AU

(72) Erfinder:
BERTINA, Johannes Marinus George, Canning Vale, AU; OLIVER, Rees Quentin, South Perth, AU

(74) Vertreter:
Vossius & Partner, 81675 München

(54) Bezeichnung: **HOST-BENUTZER-TRANSAKTIONSSYSTEM**

Beschreibung

[0001] Die vorliegende Erfindung betrifft ein Host- und Benutzer-Transaktionssystem, das insbesondere, jedoch nicht ausschließlich, zum Ausführen kommerzieller Transaktionen geeignet ist, bei denen mehrere Service-Provider und mehrere Service-User involviert sind.

[0002] In der vorliegenden Beschreibung bezeichnet der Ausdruck "Transaktion" einen beliebigen Austausch von Daten oder Information, der für die Ausführung oder Abwicklung einer kommerziellen Transaktion spezifisch oder nicht spezifisch sein kann. Außerdem sind die Ausdrücke "Service-Provider" und "Service-User" ähnlicherweise nicht auf die Bereitstellung und Verwendung von Dienstleistungen ausschließlich kommerzieller Natur beschränkt, sondern betreffen die Identität zweier beliebiger Entitäten, die in einem beliebigen Daten- oder Informationsaustausch involviert sind, für den die Erfindung anwendbar ist.

[0003] Die Erfindung basiert auf der Verwendung einer eigenständigen, tragbaren, intelligenten Vorrichtung mit einem Mikroprozessor zum Ausführen von Datenverarbeitungen. Solche Vorrichtungen sind gegenwärtig in der Form von IC-Karten verfügbar. Diese Karten haben in ihrer Grundform das Erscheinungsbild einer Standard-Kreditkarte, beinhalten jedoch verschiedenartige integrierte Schaltungen (ICs), die eine On-Board-Speicherfähigkeit bereitstellen und die Verarbeitung von Daten über einen Ein-/Ausgabeport ermöglichen.

[0004] Die Entwicklung dieser IC-Karten ist begrenzt gewesen, so daß diese Karten an sich eher als Einrichtung zum Speichern von Daten dienen als daß "Anwendungs"-programme darauf laufen. Daher arbeiten diese Karten unter der Steuerung einer externen Vorrichtung, die physikalisch Halbduplexvorrichtungen sind, die grundsätzlich Befehle empfangen und auf Abfragen antworten.

[0005] IC-Karten werden grundsätzlich in synchrone und asynchrone Karten eingeteilt. Synchrone Karten bilden im wesentlichen eine serielle Speichervorrichtung, die keine Befehle enthält. Die meisten Karten weisen keine Lese-/Schreibsteuerungsmerkmale auf, und nur einige weisen eine Zugriffssteuerung auf. Es sind Hardware-Treiberroutinen erforderlich, um den bitweisen Zugriff auf derartige IC-Karten zu implementieren.

[0006] Asynchrone Karten sind komplizierter und weisen im wesentlichen einen Einzelchip-Mikrocomputer auf, der auf der Karte vollständig eigenständig bereitgestellt wird. Der Mikrocomputer weist eine Zentraleinheit, einen flüchtigen und einen nichtflüchtigen Speicher und einen Ein-/Ausgabe-Port auf. Das Hauptprogramm des Mikrocomputers ist allgemein ein spezifisches Programm zum Partitionieren, Speichern und Abrufen von Daten im nichtflüchtigen Speicher, normalerweise mit gewissen Lese-/Schreibsteuerungsmerkmalen, die optional und ziemlich flexibel sind. Das Hauptprogramm weist ein Betriebssystem auf, das einen Befehls-Executor aufweist, der auf Befehle anspricht, um Dateien zu erzeugen, in Dateien zu schreiben, von Dateien zu lesen und Paßwörter zu handhaben. Daher weisen die meisten derartigen Karten Sicherheitsmerkmale auf, die den Zugriff auf die Karte und/oder spezifische Datenbereiche steuern und Paßwortänderungen ermöglichen.

[0007] Ein wichtiger Aspekt bezüglich der Verwendung dieser tragbaren, eigenständigen, intelligenten Vorrichtungen ist die Fähigkeit, eine sichere Datenspeicherung in diesen Vorrichtungen bereitzustellen sowie die Fähigkeit, eine sichere Übertragung dieser Daten zu und von diesen Vorrichtungen zu ermöglichen.

[0008] Eine sichere Datenspeicherung kann durch zwei Grundverfahren erreicht werden. Ein Verfahren besteht darin, die Daten in einer verschlüsselten Form im Speicher der tragbaren intelligenten Vorrichtung zu speichern, und das andere Verfahren besteht darin, die Daten als reine Daten in der Vorrichtung zu speichern, jedoch Sicherheitsmaßnahmen bereitzustellen, um den Zugriff auf diese Daten zu beschränken, z. B. einen Paßwort-Zugriffsbeschränkungsmechanismus.

[0009] Im ersten Verfahren ist ein Sicherheitsmodul erforderlich, das Code- oder Verschlüsselungsschlüssel aufweist, die durch die Sicherheitsmerkmale des Sicherheitsmoduls geheim gehalten werden, das sowohl für die tragbare Vorrichtung als auch für das Host-System verwendet wird, mit dem der Service-Provider verbunden ist. Daher würden die zu speichernden Daten durch das Sicherheitsmodul unter Verwendung eines spezifischen Verschlüsselungsschlüssels (z. B. des mit dem Speicherbereich, in den die Daten gespeichert werden sollen, in Beziehung stehenden Schlüssels) verschlüsselt und dann zum Speichern weitergeleitet.

[0010] Für das zweite Verfahren müssen die Daten in einem Sicherheitsmodul gespeichert werden, so daß der Zugriff auf dieses Modul beschränkt ist. Ein Einzelchip-Microcomputer ohne externe Busse, der in asynchronen IC-Karten dieses Typs verwendet wird, bildet ein geeignetes Sicherheitsmodul, so daß derartige IC-Karten für die vorliegende Erfindung besonders geeignet sind.

[0011] Derartige IC-Karten werden durch Programme gesteuert, die aus einem im Masken-ROM-Speicher des Mikrocomputers gespeicherten Maschinencode für den Mikrocomputer bestehen. Ein solches Programm wird beim Einschalten nach einem Reset-Vorgang ausgeführt und steuert alle Zugriffe auf den Daten-Speicherbereich des Mikrocomputers. Das Programm steuert die seriellen Kommunikationen und erkennt mehrere Hochpegel-Befehlsrahmen der seriellen Kommunikationen und spricht darauf an. Diese Befehlsrahmen dienen zum Erzeugen von Speicherbereichen, Öffnen eines Speicherbereichs zum Lesen oder Schreiben, Bereitstellen eines Paßwortes für einen Speicherbereich, durch das eine Karte freigegeben wird, nachdem zu viele falsche Paßwörter eingegeben worden sind, und möglicherweise zum Verschlüsseln. Durch Handhaben nur spezifischer Befehle auf sehr spezifische Weisen kann durch die Karte eine sichere Datenspeicherung bereitge-

stellt werden.

[0012] Gegenwärtige Techniken zum Bereitstellen einer sicheren Datenübertragung basieren auf der Verwendung geheimer Verschlüsselungsschlüssel zum Verschlüsseln der Daten. Diese Schlüssel müssen sicher gespeichert und sicher verwendet werden, weil die Datenübertragung ansonsten nicht sicher ist.

[0013] Mit der Weiterentwicklung derartiger tragbarer, eigenständiger, intelligenter Vorrichtungen und der erweiterten Anwendung der Computertechnologie zum Bereitstellen von Ferntransaktionen für Benutzer und Hosts über Verkaufsplatzvorrichtungen, Geldausgabeautomaten und ähnliche Geräte, die zum Ausführen finanzieller Transaktionen bereits vorhanden sind, ergibt sich eine geeignete Gelegenheit für weitere Modifikationen der gesamten kommerziellen und Informations-Transaktions-Schnittstelle sowohl im Interesse der Service-Provider als auch der Service-User.

[0014] In der EP-A-190733 wird ein System zum Ausführen von Transaktionen beschrieben. Es weist jedoch keinen Interpreter auf. Ein in der FR-A-2667177 beschriebenes System weist einen Interpreter auf. Es wird jedoch in diesem Dokument nicht erwähnt, daß der Interpreter begrenzte Steuerungsfunktionen zum Einschränken des Datendateizugriffs aufweist.

[0015] Obwohl durch IC-Karten mit Einzelchip-Microcomputern geeignet Transaktionen zwischen Service-Usern und Service-Providern ermöglicht werden, besteht aufgrund des Fehlens einer Normierung des Befehlssatzes dieser Karten, der fehlenden Sicherheit bei der Bereitstellung eines Paßwortes und der mangelnden Flexibilisierung in der Verwendung dieser Karten immer noch eine Abneigung, die vollen Fähigkeiten dieser Karten zum Ausführen von Transaktionen auszunutzen, insbesondere wenn vertrauliche oder sensitive Daten und Information auf der Karte gespeichert werden müssen.

[0016] Es ist eine Aufgabe der vorliegenden Erfindung, ein System bereitzustellen, daß es einem Service-User ermöglicht, mehrere Transaktionen bezüglich verschiedenen Service-Providern auszuführen, während ein hoher Sicherheitsgrad bei der Ausführung von Transaktionen sowohl bezüglich des Service-Providers als auch bezüglich des Service-Users bereitgestellt wird.

[0017] Es ist eine bevorzugte Aufgabe der vorliegenden Erfindung, ein System für einen sicheren Austausch von Daten und Information zwischen einem Service-Provider und einem Service-User bereitzustellen, um Transaktionen zwischen dem Service-Provider und dem Service-User zu ermöglichen, die den Austausch kommerzieller Daten und/oder Information mit einem hohen Sicherheitsgrad beinhalten können.

[0018] Es ist eine weitere bevorzugte Aufgabe der vorliegenden Erfindung, eine tragbare, eigenständige, intelligente Vorrichtung bereitzustellen, die ein sicheres Modul zum Speichern von Daten und Information bilden kann, die ihr von einem Service-Provider zugeführt werden.

[0019] Gemäß einem Aspekt der vorliegenden Erfindung wird ein System zum Ausführen mehrerer Transaktionen bereitgestellt, mit.

(i) einer eigenständigen, tragbaren, intelligenten Vor-Richtung, die einen Einzelchip-Mikrocomputer mit einem Ein/Ausgabe-Kommunikationsport, einem nichtflüchtigen Speicher und einem RAM-Speicher oder Direktzugriffsspeicher aufweist; und

(ii) einer Schnittstellenvorrichtung, die einem Speicher zum Speichern von Daten zugeordnet ist und einen Koppler aufweist, der dazu mit der intelligenten Vorrichtung verbindbar ist, um Kommunikationen damit einzurichten;

[0020] wobei der nichtflüchtige Speicher so strukturiert ist, daß ein Teil davon eine Betriebssystemmaske aufweist, die in Maschinencode für den Mikrocomputer programmiert ist, um Grundfunktionen auszuführen, und ein zweiter Teil davon dazu geeignet ist, Datendateien zu speichern, denen verschiedene Zugriffsbeschränkungsgrade zugeordnet sind,

wobei der RAM-Speicher zur Verwendung durch das Betriebssystem, wenn eine Funktion ausgeführt wird, und zum Speichern von Daten geeignet ist, die vom Kommunikationsport empfangen werden oder für eine Übertragung über den Kommunikationsport bereit sind; und

wobei die Schnittstellenvorrichtung ein Programmmodul innerhalb des Speichers aufweist, das eine oder mehrere Instruktionen aufweist, die mit vorgeschriebenen Datendateien innerhalb des zweiten Teils des nichtflüchtigen Speichers gemäß dem Betriebssystem arbeiten, nachdem die intelligente Vorrichtung mit dem Koppler verbunden ist;

wobei das Betriebssystem aufweist: (a) einen Befehls-Executor zum Empfangen eines Befehls, zum Ausführen einer vorgeschriebenen Funktion bezüglich des Befehls und zum Bereitstellen eines Ergebnisses oder Status für den Befehl; und (b) einen Programm-Interpreter zum Ausführen des Programmmoduls, um eine Transaktion auszuführen; wobei

ein System zum Ausführen einer Transaktion aufweist:

(i) eine eigenständige, tragbare, intelligente Vorrichtung mit einem Einzelchip-Mikrocomputer mit einem Ein/Ausgabe-Kommunikationsport, einem nichtflüchtigen Speicher und einem RAM-Speicher oder Direktzugriffsspeicher; und

(ii) eine Schnittstellenvorrichtung, die einem Speicher zum Speichern von Daten zugeordnet ist und einen

Koppler aufweist, der mit der intelligenten Vorrichtung verbindbar ist, um Kommunikationen damit einzurichten;

[0021] wobei der nichtflüchtige Speicher so strukturiert ist, daß ein Teil davon eine Betriebssystemmaske aufweist, die in Maschinencode für den Mikrocomputer programmiert ist, um Grundfunktionen auszuführen, und ein zweiter Teil davon dazu geeignet ist, Datendateien zu speichern, denen verschiedene Zugriffsbeschränkungsgrade zugeordnet sind,
wobei der RAM-Speicher zur Verwendung durch das Betriebssystem, wenn eine Funktion ausgeführt wird, und zum Speichern von Daten geeignet ist, die vom Kommunikationsport empfangen werden oder für eine Übertragung über den Kommunikationsport bereit sind; und
wobei die Schnittstellenvorrichtung ein Programmmodul innerhalb des Speichers aufweist, das eine oder mehrere Instruktionen aufweist, die mit vorgeschriebenen Datendateien innerhalb des zweiten Teils des nichtflüchtigen Speichers gemäß dem Betriebssystem arbeiten, nachdem die intelligente Vorrichtung mit dem Koppler verbunden ist;
wobei das Betriebssystem aufweist: (a) einen Befehls-Executor zum Empfangen eines Befehls, zum Ausführen einer vorgeschriebenen Funktion bezüglich des Befehls und zum Bereitstellen eines Ergebnisses oder Status für den Befehl; und (b) einen Programm-Interpreter zum Ausführen des Programmmoduls, um eine Transaktion auszuführen; wobei
die eine oder mehreren Instruktionen einen Teil des vorgeschriebenen Instruktionssatzes bilden, der vom Maschinencode getrennt ist und begrenzte Steuerungsfunktionen zum Einschränken des Datendateizugriffs aufweist.
[0022] Vorzugsweise wird das Programmmodul durch den Befehls-Executor über den Kommunikationsport in den RAM-Speicher geladen, wobei das Programmmodul durch den Befehls-Executor in Antwort auf einen vorgeschriebenen Befehl über den Kommunikationsport in den RAM-Speicher geladen wird, der empfangen und ausgeführt wird, wenn die intelligente Vorrichtung mit dem Koppler verbunden ist, um durch den Programm-Interpreter ausgeführt zu werden.
[0023] Alternativ kann das Programmmodul über das System nach Anspruch 1 in einen dritten Teil des nichtflüchtigen Speichers geladen werden, wobei das Programmmodul durch den Befehls-Executor in Antwort auf einen vorgeschriebenen Befehl über den Kommunikationsport in einen dritten Teil des nichtflüchtigen Speichers geladen wird, der empfangen und ausgeführt wird, wenn die intelligente Vorrichtung mit dem Koppler verbunden ist, um durch den Programm-Interpreter ausgeführt zu werden.
[0024] Vorzugsweise ist das Programmmodul verschlüsselt und weist das Betriebssystem ein Datenverschlüsselungs- und -entschlüsselungsprogramm zum Entschlüsseln des Programmmoduls gemäß einem vorgeschriebenen Algorithmus auf.
[0025] Vorzugsweise weist die intelligente Vorrichtung eine Tastatur und ein Display auf.
[0026] Vorzugsweise ist der Algorithmus ein DES- (Data Encryption Standard) Algorithmus oder ein RSA-Algorithmus. Vorzugsweise werden mehrere Programmmodule sequentiell in den RAM-Speicher geladen und miteinander verkettet, um die Leistungsfähigkeit des Systems zu erweitern. Vorzugsweise sind die Programmmodule verschlüsselt und weist das Betriebssystem ein Datenverschlüsselungs- und -entschlüsselungsprogramm zum Entschlüsseln der Programmmodule im Blockverkettungsmodus gemäß einem vorgeschriebenen Algorithmus auf.
[0027] Vorzugsweise ist der Algorithmus ein DES- (Data Encryption Standard) Algorithmus oder ein RSA-Algorithmus. Vorzugsweise weisen die Datendateien jeweils Datensätze auf, so daß verschiedenen Datensätzen innerhalb einer Datei verschiedene Zugriffsbeschränkungsgrade zugeordnet werden können.
[0028] Vorzugsweise ist das Programmmodul dazu geeignet, zu ermöglichen, daß ein für eine spezifische Anwendung geeignetes Datendarstellungsverfahren verwendbar ist.
[0029] Gemäß einem anderen Aspekt der Erfindung wird eine eigenständige, tragbare, intelligente Vorrichtung zum Ausführen einer Transaktion mit einer Schnittstellenvorrichtung bereitgestellt, der ein Speicher zum Speichern von Daten zugeordnet ist und die einen Koppler für eine Verbindung mit der intelligenten Vorrichtung zum Einrichten von Kommunikationen mit der intelligenten Vorrichtung aufweist, mit:
einem Einzelchip-Mikrocomputer mit einem Ein-/Ausgabe-Kommunikationsport, einem nichtflüchtigen Speicher und einem RAM-Speicher; wobei

(i) der nichtflüchtige Speicher so strukturiert ist, daß ein Teil davon eine Betriebssystemmaske aufweist, die in Maschinencode für den Mikrocomputer programmiert ist, um Grundfunktionen auszuführen, und ein zweiter Teil davon dazu geeignet ist, Datendateien zu speichern, denen verschiedene Zugriffsbeschränkungsgrade zugeordnet sind;

(ii) der RAM-Speicher zur Verwendung durch das Betriebssystem geeignet ist, wenn eine Funktion ausgeführt wird, und zum Speichern von Daten, die über den Kommunikationsport empfangen wurden oder für eine Übertragung über den Kommunikationsport bereit sind;

(iii) das Betriebssystem aufweist: (a) einen Befehls-Executor zum Empfangen eines Befehls, zum Ausfüh-

ren einer vorgeschriebenen Funktion bezüglich des Befehls und zum Bereitstellen eines Ergebnisses oder Status für den Befehl; und (b) einen Programm-Interpreter zum Ausführen eines ihm zugeführten Programmmoduls zum Abwickeln einer Transaktion;
(iv) das Programmmodul eine oder mehrere Instruktionen zum Arbeiten mit vorgeschriebenen Datendateien innerhalb des zweiten Teils des nichtflüchtigen Speichers gemäß dem Betriebssystem aufweist; und
(v) die eine oder mehreren Instruktionen einen Teil eines vorgeschriebenen Instruktionssatzes biden, der vom Maschinencode getrennt ist und eingeschränkte Steuerungsfunktionen zum Einschränken des Datendateizugriffs aufweist.

[0030] Gemäß einem anderen Aspekt der Erfindung wird ein Verfahren zum Ausführen einer Transaktion zwischen einer intelligenten Vorrichtung und einer Schnittstellenvorrichtung bereitgestellt, mit den Schritten:
Erzeugen eines Programmmoduls, das eine oder mehrere Instruktionen zum Arbeiten mit vorgeschriebenen Datendateien innerhalb der intelligenten Vorrichtung aufweist;
Speichern des Programmmoduls im Speicher der Schnittstellenvorrichtung;
Verbinden der intelligenten Vorrichtung mit der Schnittstellenvorrichtung; und
Aufrufen eines Programm-Interpreters, der einen Teil des Betriebssystems der intelligenten Vorrichtung bildet, um die Instruktionen des Programmmoduls zu interpretieren und auszuführen;
wobei der eine oder die mehreren Instruktionen einen Teil eines vorgeschriebenen Instruktionssatzes bilden, der vom Maschinencode für den Mikrocomputer der intelligenten Vorrichtung getrennt ist und eingeschränkte Steuerungsfunktionen zum Einschränken des Datendateizugriffs aufweist.
[0031] Die Erfindung wird anhand der folgenden Beschreibung einer spezifischen Ausführungsform der Erfindung verdeutlicht. Die Beschreibung erfolgt unter Bezug auf die beigefügten Zeichnungen; es zeigen:
[0032] **Fig.** 1 ein Blockdiagramm der intelligenten Vorrichtung;
[0033] **Fig.** 2 eine schematische Ansicht zum Darstellen der Weise, auf die die intelligente Vorrichtung in Verbindung mit einem Host verwendet wird, um eine Transaktion auszuführen;
[0034] **Fig.** 3 ein Ablaufdiagramm zum Darstellen der Menü-Verarbeitungszustände des Betriebssystemprogramms;
[0035] **Fig.** 4 ein Ablaufdiagramm zum Darstellen der Menüpunkteroutine des Betriebssystems;
[0036] **Fig.** 5 ein Blockdiagramm zum Darstellen der logischen Strukturen des Befehls-Executors;
[0037] **Fig.** 6 ein Ablaufdiagramm zum Darstellen des Arbeitsablaufs der Unterroutine des Befehls-Executors;
[0038] **Fig.** 7 ein Blockdiagramm zum Darstellen der logischen Strukturen des Programm-Interpreters;
[0039] **Fig.** 8 ein Ablaufdiagramm zum Darstellen der Verarbeitungszustände des Programm-Interpreters; und
[0040] **Fig.** 9 ein Ablaufdiagramm zum Darstellen des Arbeitsablaufs der Unterroutine des Programm-Interpreters.
[0041] Die Ausführungsform betrifft ein Host- und Benutzer-Transaktionssystem. Der Host weist mehrere Service-Provider auf, die Dienste oder Services für mehrere Benutzer bereitstellen, die in der Lage sind, wechselseitig als geeignet betrachtete Transaktionen mit bestimmten Service-Providern unabhängig einzurichten und auszuführen.
[0042] Um Transaktionen einzurichten und auszuführen verfügt jeder der Service-Provider über eine Schnittstellenvorrichtung **11**, die einer zentralen Host-Verarbeitungsvorrichtung zugeordnet ist, und jeder der Service-User besitzt eine eigenständige, tragbare, intelligente Vorrichtung **13**.
[0043] Die Schnittstellenvorrichtung **11** des Service-Providers ist einem (nicht dargestellten) Speicher zum Speichern von Daten zugeordnet und weist einen Koppler **14** auf, der dazu geeignet ist, mit einer intelligenten Vorrichtung **13** eines Service-Users verbunden zu werden, um Kommunikationen damit einzurichten. Die Schnittstellenvorrichtung **11** kann ein Endgerät, eine Kommunikationsvorrichtung oder eine Maschine sein, mit der die intelligente Vorrichtung während ihres Betriebs elektrisch verbindbar ist und die entfernt von oder in der Nähe der Host-Verarbeitungsvorrichtung angeordnet sein kann.
[0044] In der vorliegenden Ausführungsform ist die Schnittstellenvorrichtung **11** ein Verkaufsplatz- (POS) Endgerät mit einer Kartenaufnahmeeinrichtung (nicht dargestellt), die einen Koppler **14** aufweist, um Kommunikationen mit der intelligenten Vorrichtung **13** einzurichten.
[0045] Das POS-Endgerät ist über eine herkömmliche Kommunikationsleitung mit der Host-Verarbeitungsvorrichtung verbunden und weist eine begrenzte Verarbeitungskapazität durch einen Mikroprozessor und einen Speicher auf, um Kommunikationen zwischen der Host-Verarbeitungsvorrichtung und der intelligenten Vorrichtung sowie zwischen dem POS-Endgerät und der intelligenten Vorrichtung unabhängig von der Host-Verarbeitungsvorrichtung zu ermöglichen.
[0046] Die Host-Verarbeitungsvorrichtung weist ein Computersystem auf, das ein vorgeschriebenes Verschlüsselungsverarbeitungssystem aufweist, das zur Verwendung mit mehreren Schlüsseln geeignet ist, einschließlich eines damit gespeicherten geheimen Schlüssels, der zu einem in einer spezifischen intelligenten Vorrichtung gespeicherten geheimen Schlüssel paßt.

[0047] Die in **Fig.** 1 dargestellte intelligente Vorrichtung **13** weist auf: einen Mikrocomputer mit einem seriellen Ein/Ausgabe-Kommunikationsport 15, einem nichtflüchtigen Speicher in der Form eines Masken-ROM-Speichers **17** und eines EEPROM **19**, einen RAM-Speicher **21**, eine Tastatur **23**, ein Display **25**, eine Zentraleinheit (CPU) **27**, eine Oszillatorschaltung **29** und eine Stromversorgung in der Form einer Batterie **31**, die alle in einer Identifizierungskarte in der Form einer allgemein bekannten visuellen IC-Karte eingebettet sind.
[0048] Die intelligente Vorrichtung **13** ist batteriebetrieben und kann daher verwendet werden ohne daß andere elektronische Geräte erforderlich sind. Die intelligente Vorrichtung **13** ist jedoch nicht darauf beschränkt, sondern durch die Bereitstellung des seriellen Ein-/Ausgabe-Kommunikationsports **15** kann sie auch mit dem Koppler **14** elektrisch verbunden werden, um Kommunikationen mit der Schnittstellenvorrichtung **11** zu ermöglichen.
[0049] Der Mikrocomputer ist in einer einzelnen monolithischen integrierten Schaltung eingebettet, in der sowohl die CPU **27** als auch alle flüchtigen und nichtflüchtigen Speicher des Mikrocomputers aufgenommen sind. Der Mikrocomputer ist so strukturiert, daß er nur in einem Einzelchipmodus durch ein Betriebssystemprogramm betreibbar ist, das zum Zeitpunkt der Fertigung im ROM-Speicher **17** der integrierten Schaltung maskiert wird. Dieses als "Masken"-programm bekannte Betriebssystemprogramm bildet die Basis für den Betrieb der intelligenten Vorrichtung und ruft mehrere diskrete Spezialroutinen auf, die ebenfalls im ROM-Speicher maskiert sind, um ein Gesamt-"Masken"-programm zu bilden, durch das ein hoher Sicherheitsgrad bereitgestellt wird.
[0050] Das Betriebssystem des "Masken"-Programms weist die üblichen Standardroutinen auf, die Funktionen ausführen, wie beispielsweise Erfassen von Tastenbetätigungen oder -anschlägen auf der Tastatur **23**, Schreiben von Daten auf das Display **25**, Kommunizieren mit einem externen System über den seriellen Kommunikationsport **15**, Speicherzuweisung und Verschlüsselung. Insbesondere stehen die Spezialroutinen mit der vorliegenden Erfindung in Beziehung und werden später ausführlicher beschrieben.
[0051] Die Tastatur **23** weist mehrere numerische Tasten "0 bis 9" und mehrere Funktionstasten "Neue Zeile" (NL), "Eingabe" oder "Enter" (E), "Löschen" (*) und "Menü" (M) auf, deren Zweck später ausführlicher beschrieben wird.
[0052] Das Display **25** ist ein herkömmliches 16-Zeichen-Flüssigkristall-Display.
[0053] Der Mikrocomputer-IC ist ein speziell für visuelle IC-Karten geeigneter IC mit einem Verschlüsselungsverarbeitungssystem.
[0054] Die Batterie **31** ist eine Lithiumbatterie, und die Oszillatorschaltung **29** ist ein Standard-Quarzoszillator, und beide sind für den Mikrocomputer-IC geeignet.
[0055] Der EEPROM **19** des nichtflüchtigen Speichers ist dazu geeignet, mehrere Datendateien **33** zu speichern, wie in **Fig.** 2 schematisch dargestellt ist. Datendateien sind grundsätzlich Partitionen im nichtflüchtigen Speicher, die zur Datenspeicherung verwendet werden. Diese Dateien können mit verschiedenen Zugriffsbeschränkungen erzeugt werden, z. B. "Lesen", "Schreiben", "nur Lesen", "nur Schreiben", "kein Zugriff", "Paßwortzugriff". Die Datendateien **33** weisen jeweils einen Daten-Kopf oder Header und einen oder mehrere Datensätze **35** auf. Der Header enthält Information zum Identifizieren der Datei, die Anfangsadresse der Datei, die nächste freie Adresse nach dem Ende der Datei (die der Header der neuen Datei wird), die Datensatzgröße und -anzahl, ihre Zugriffsmerkmale und ihre Paßwortanzahl.
[0056] In der vorliegenden Ausführungsform können sechzehn Paßwörter verwendet werden, wobei jedes Paßwort eine Länge von bis zu acht Zeichen haben kann. Wenn das Paßwort weniger als acht Zeichen hat, wird ein Byte mit dem Wert 0 verwendet, um die Paßworteingabe abzuschließen.
[0057] Die Partitionen des nichtflüchtigen Speichers, in denen die Dateien gespeichert sind, werden erzeugt, wenn die Header-Information geschrieben wird, wodurch die Adresse des Anfangs der Datei und die nächste freie Adresse durch das Betriebssystem erkannt werden.
[0058] Wenn eine Datendatei **33** mehrere Datensätze **35** aufweist, sind die Datensätze selbst zusätzlich durch Datensatz-Header definiert, um verschiedene Zugriffsbeschränkungen bereitzustellen, so daß verschiedenen Datensätzen in einer Datendatei verschiedene Zugriffsbeschränkungen zugeordnet sein können, ohne daß für die gesamte Datei die gleichen Zugriffsbedingung gelten muß. Beispielsweise kann ein Paßwort erforderlich sein, um einen spezifischen Datensatz einer Datei zu lesen, und zum Lesen der restlichen Daten der Datei kann eine persönliche Identifikationsnummer (PIN) plus ein Paßwort erforderlich sein.
[0059] Auf diese Weise kann die Anzahl von Dateien, die innerhalb des verfügbaren Speicherplatzes des EEPROM **19** gespeichert werden müssen, reduziert werden, wodurch Speicherplatz eingespart wird, weil jede Datendatei zusätzlich zu ihren Datensätzen auch eine Header-Information am Anfang der Datei aufweisen muß, die Speicherplatz benötigt. Außerdem wird eine erhöhte Flexibilität und eine einfachere Struktur bereitgestellt, indem die Datendateien **33** in erster Linie durch die Funktion und nicht zwangsweise durch Zugriffsbeschränkung unterschieden werden.
[0060] Die Spezialroutinen des Maskenprogramms des ROM-Speichers **17** weisen eine Hauptmenüroutine, einen Befehls-Executor und einen Programm-Interpreter auf. Wie in **Fig.** 2 dargestellt ist, ist das Maskenprogramm des ROM-Speichers **17** in drei Funktionsblöcke geteilt. Der erste Funktionsblock **37** ist das Betriebs-

systeme und die Hauptmenüroutine, die eng miteinander in Beziehung stehen, der zweite Funktionsblock **39** ist die Befehls-Executor-Routine, und der dritte Funktionsblock ist der Programm-Interpreter **41**.

[0061] Die Hauptmenüroutine betrifft eingebaute Funktionen, die durch Aufrufen des Menümodus und Scrollen durch den Speicher durch aufeinanderfolgendes Drücken der M-Taste ausgewählt werden. Die Menüpunkte werden aufeinanderfolgend dargestellt und können durch die E-Taste ausgewählt werden.

[0062] Wie in **Fig.** 3 dargestellt ist, weist das Betriebssystem grundsätzlich sechs Betriebs- oder Operationszustände auf, die jeweils durch die in **Fig.** 4 dargestellte Hauptmenüroutine ausgewählt werden können. Diese Zustände sind grundsätzlich in zwei Gruppen geteilt, wobei für eine Gruppe eine PIN und für die andere keine PIN erforderlich ist. Die Zustände, für die eine PIN erforderlich ist, weisen einen "PIN-Eingabe"-Zustand **61**, einen "Fernidentifizierung über seriellen Port"-Zustand **63**, einen "Befehlsmodus"-Zustand **65** und einen "PIN-Änderung"-Zustand **67** auf. Die übrige Gruppe, für die keine PIN erforderlich ist, weist einen "Fernidentifizierung über seriellen Port"-Zustand **69** und einen "Befehlsmodus"-Zustand **71** auf. Jede dieser Gruppen kann ausgewählt werden, nachdem die Vorrichtung eingeschaltet wurde, wie durch einen Startblock **72** dargestellt ist.

[0063] Nach dem Eintritt in einen spezifischen Zustand wird eine geeignete Unterroutine ausgeführt. Im Fall der beiden "Befehlsmodus"-Zustände **65** und **71** werden die speziellen Befehls-Executor- und Programm-Interpreter-Routinen **73** bzw. **75** aufgerufen, wobei die Programm-Interpreter-Routine **75** auf eine später ausführlicher beschriebene Weise aufgerufen wird, wenn der Befehls-Executor **73** einen vorgegebenen Befehl empfängt.

[0064] Der "PIN-Eingabe"-Zustand beinhaltet die Verwendung einer Standardroutine zum Decodieren einer PIN, die durch einen Benutzer in die Vorrichtung eingegeben wird. In der vorliegenden Ausführungsform wird durch Auswählen des Punkts "PIN-Eingabe" veranlaßt, daß ein anderer Verschlüsselungsschlüssel für den "Fernidentifizierung über seriellen Port"-Zustand oder den "Befehlsmodus"-Zustand verwendet wird. Auf diese Weise können in Abhängigkeit davon, ob ein Paßwort verwendet wird oder nicht, bezüglich den durch die Vorrichtung im "Fernidentifizierung über seriellen Port"-Zustand oder im "Befehlsmodus"-Zustand ausgeführten Funktionen verschiedene Sicherheitsgrade bereitgestellt werden. Daher können in Abhängigkeit davon, ob der "Fernidentifizierung über seriellen Port"-Zustand oder der "Befehlsmodus"-Zustand aufgerufen wird, in Abhängigkeit von den Zugriffsbeschränkungen der Datendatei und der Datensätze auf verschiedene Datendateien und/oder Datensätze davon zugegriffen werden, so daß die Vorrichtung in Abhängigkeit vom Paßwortzugriff bequem für Anwendungen mit niedrigem Sicherheitsgrad sowie für Anwendungen verwendbar ist, für die ein hoher Sicherheitsgrad erforderlich ist.

[0065] Der Zugriff auf diese verschiedenen Zustände wird durch das in **Fig.** 4 dargestellte Menüablaufdiagramm demonstriert. Die Hauptmenüroutine beginnt in Block **77**, in dem drei Anfangszustände **61**, **69** und **71** wählbar sind. Der erste Zustand "PIN-Eingabe" **61** wird als Menüpunkt angezeigt, wie durch Block **79** dargestellt ist, woraufhin die Routine auf eine Tasteneingabe wartet, wie durch Block **81** dargestellt ist. Es folgt ein Entscheidungsblock **83** zum Bestimmen, welche Taste betätigt wurde, wobei, wenn die M-Taste betätigt wurde, die Routine den nächsten Menüpunkt anzeigt, wie durch Block **85** dargestellt ist, und in Block **81** erneut auf eine Tasteneingabe wartet. Wenn die E-Taste betätigt wird, wodurch angezeigt wird, daß der dargestellte Menüpunkt ausgewählt wurde, wird die geeignete Unterroutine für diesen Punkt ausgeführt, wie durch Block **87** dargestellt ist. Wenn eine von der M- und der E-Taste verschiedene Taste betätigt wird, wird dies als ungültige Tastenbetätigung interpretiert, woraufhin die Unterroutine zurückspringt, um auf eine andere Tasteneingabe zu warten, wie durch Block **81** dargestellt ist.

[0066] In Block **87** wird, wenn der "PIN-Eingabe"-Zustand ausgewählt wird, die geeignete Unterroutine ausgeführt, wofür die korrekte PIN eingegeben werden muß. Nach Abschluß dieser Unterroutine, wie durch Block **89** dargestellt, schreitet die Hauptmenüroutine fort und bestimmt durch einen Entscheidungsblock **91**, ob die vorangehenede Unterroutine die "PIN-Eingabe"-Routine ist, wobei, wenn dies der Fall ist, die Unterroutine die übrigen Zustände **63**, **65** und **67** verfügbar macht, wie in Block **93** dargestellt ist. Wenn die in Schritt **87** abgearbeitete Unterroutine eine mit dem "Fernidentifizierung über seriellen Port"-Zustand **69** oder dem "Befehlsmodus"-Zustand **71** in Beziehung stehende Unterroutine ist, bestimmt die Routine nach Abschluß der Unterroutine von Block **89** in Block **91**, ob einer PIN-Eingabe einer der vorstehend erwähnten Zustände folgt. Wenn dies nicht der Fall ist, springt die Routine zum Anfangsmenüblock **77** zurück, und anschießend wird der erste Menüpunkt gemäß Block **79** dargestellt.

[0067] Der "Fernidentifizierung über seriellen Port"-Zustand ist so strukturiert, daß eine Fernidentifizierung eines Benutzers der intelligenten Vorrichtung ermöglicht wird, wobei die Vorrichtung mit einer Schnittstellenvorrichtung eines Service-Providers verbunden sein kann oder nicht. Durch den Identifizierungszustand wird ein Algorithmus aufgerufen, der einem in der Host-Verarbeitungsvorrichtung gespeicherten Algorithmus entspricht. Dieser Algorithmus weist einen Abfrage/Antwort-Modus (Challenge/Response-Modus) auf, der eine Verschlüsselungsverarbeitung unter Verwendung eines Verschlüsselungsschlüssels beinhaltet. Wie vorstehend beschrieben wurde, unterscheidet sich dieser Verschlüsselungsschlüssel in Abhängigkeit davon, ob in den "Fernidentifizierung über seriellen Port"-Zustand **63** oder **69** eingetreten wird. Diese Verschlüsselungs-

schlüssel stimmen mit entsprechenden, in der Host-Verarbeitungsvorrichtung gespeicherten Verschlüsselungsschlüsseln überein, so daß, wenn ein übereinstimmender Schlüssel ausgewählt wird, durch einen Abfrage- oder Challenge-Code, der in die intelligente Vorrichtung und die Host-Verarbeitungsvorrichtung eingegeben wird, veranlaßt wird, daß durch die durch den Algorithmus in jeder Vorrichtung ausgeführte Verschlüsselungsverarbeitung der gleiche Identifizierungscode erhalten wird. Dadurch kann die Identität des Kartenbenutzers oder alternativ des Service-Providers durch eine andere Partei bestimmt und bestätigt werden, die einen geeigneten Challenge-Code ausgibt und die Übereinstimmungsantwort empfängt.

[0068] Derartige Algorithmen sind in der Industrie bekannt und werden hierin nicht näher beschrieben.

[0069] Im "Befehlsmodus"-Zustand steht die Befehls-Executor-Routine mit Kommunikationen in Beziehung, die über den seriellen Kommunikationsport **15** der intelligenten Vorrichtung ausgeführt werden. Befehle für die Vorrichtung werden verarbeitet, und es wird eine Antwort oder ein Status zurückgegeben, um anzuzeigen, ob die Kommunikation abgeschlossen ist oder mehr Daten erforderlich sind.

[0070] Das Betriebssystem wechselwirkt mit der Befehls-Executor-Routine, ist jedoch davon getrennt. Diese Wechselwirkung kommt ins Spiel, wenn Befehle am seriellen Kommunikationsport **15** der Vorrichtung empfangen werden.

[0071] Die Befehls-Executor-Routine **39** ist den für die meisten IC-Karten und visuellen IC-Karten konstruierten Befehls-Executor-Routinen ähnlich. Sie ist so konstruiert, daß sie die CPU **27** veranlaßt auf den Empfang eines Befehls von der Betriebssystemebene **37** zu warten, in Abhängigkeit vom Befehl eine bestimmte Funktion oder einen Satz von Funktionen ausführt und ein Ergebnis und/oder einen Status für den Befehl bereitstellt, um das Betriebssystem zu veranlassen, die Verarbeitung fortzusetzen, bevor die CPU auf den nächsten Befehl wartet. In diesem Sinne wird der Befehls-Executor **39** für bestimmte Verarbeitungen verwendet, z. B. zum Erzeugen von Datendateien **33** oder zum Setzen von Zugriffsbeschränkungen für die Datendateien **33** und ihre Datensätze **35** auf einen von der Betriebssystemebene **37** erhaltenen Zugriffsbeschränkungsgrad.

[0072] In der vorliegenden Ausführungsform wird folgende Befehlsliste verwendet:

Auf Display darstellen
Von Tastatur empfangen
Paßwort eingeben
Öffne Datenbereich (Datei)
Lies Daten
Schreibe Daten
Ändere Paßwort
Erzeuge Datenbereich (Datei)
Lade Programmmodul
Führe Programmmodul aus
Verschlüsseln
Entschlüsseln

[0073] Diese Befehle sind mit Ausnahme der Befehle "Lade Programmmodul" und "Führe Programmmodul aus" , die für die nachstehend beschriebene Programm-Interpreter-Routine relevant sind, im wesentlich selbsterklärend.

[0074] Die Programm-Interpreter-Routine steht mit einem Programmmodul oder einer Liste von Befehlen in Beziehung, die über den seriellen Kommunikationsport **15** empfangen werden.

[0075] Die Programm-Interpreter-Routine **41** stellt eine Betriebsebene bereit, die vom Betriebssystem **37** und vom Befehls-Executor **39** unabhängig und getrennt ist. Der Programm-Interpreter **41** ist im wesentlichen so konstruiert, daß er ausgewählte Programmmodule **43** ausführt, die in der vorliegenden Ausführungsform im RAM-Speicher **21** gespeichert sind.

[0076] Instruktionen werden von der Liste von im Programmmodul erscheinenden Instruktionen sequentiell abgerufen und so interpretiert, daß die CPU **27** die den Instruktionen entsprechenden Funktionen oder Verarbeitungen ausführt.

[0077] Der Instruktionssatz des Programm-Interpreters enthält 38 verschiedene Instruktionen, von denen einige Details nachstehend in Kurzfassung aufgelistet sind.

Get Key [n][t](p1)	Speichere (n) Tastenanschläge, die durch die Taste (t) abgeschlossen sind, an Stellen, die bei (p1) beginnen. (n) liegt im Bereich von 1 bis 16. (t) kann ein beliebiger Tastencode oder "keine Taste" (null) sein, wenn keine Abschlußtaste erforderlich ist.
Put LCD (p1)(p2)	Setze das Zeichen an der Stelle (p1) an der Position (q) auf dem Display.
Get LCD (p2)(p1)	Erfasse das Zeichen an der Displayposition (p2) und speichere es an der Stelle (p1).
Swap LCD	Tausche verborgene & sichtbare Displayzeilen (das Display weist 1 physikalische Zeile und 2 logisch Zeilen auf).
Put Comm (p1)	Übertrage das Zeichen an der Stelle (p1) an den seriellen Port.
Get Comm (p1)	Übernimm das Zeichen vom seriellen Port und speichere es an der Stelle (p1).
Set rate [n]	Setze Datenrate des seriellen Ports. (n) = 0 für 9600, 1 für 4800, 2 für 2400 und 3 für 1200 bps. (Default ist 9600.)
Put Record (p1)[n][f]	Übernimm Daten beginnend an der Stelle (p1) und speichere sie im Datensatz (n) der Datei (f).
Get Record (p1)[n][f]	Übernimm Daten von Datensatz (n) der Datei (f) und speichere sie beginnend an der Stelle (p1).
Open [f]	Öffne Datei (f) für einen Zugriff.
Close	Schließe die offene Datei (es kann jeweils nur eine Datei geöffnet sein).
Create [f] [n] [s] [a]	Erzeuge Datei (f) mit (n) Datensätzen der Größe (s) mit Zugriffsregeln (a). (Wenn (r) = 0 ist, muß Add Rec verwendet werden.)
Add Rec [s] [a]	Füge Datensatz zur gerade erzeugten Datei hinzu, wobei eine Datensatzgröße (r) und Zugriffsregeln (a) verwendet werden. (Kann nur verwendet werden, wenn (r) in Create den Wert 0 hat.)
Pres PW [n](p1)	Stelle Daten beginnend an der Stelle (p1) als Paßwort [n] dar.
Comp (p1)(p2)	Vergleiche Daten an den Stellen (p1) und (p2) und setze Flags, die das Ergebnis "=", "<" oder ">" anzeigen.
Branch X[n]	Wenn der Flagzustand "X" vorliegt: ändere Programmausführung, um das Programm von der aktuellen Position (n) abzuarbeiten. X kann die Zustände haben: EQ, NE, GT, LT & AW. EQ ist "=", NE ist "≠", GT ist ">", LT ist "<"und AW ist "immer".
ST Loop [n]	Führe die folgende Liste von Instruktionen (n)-mal aus. Der Liste muß eine Instruktion "Loop Bk" folgen.
Loop Bk	Wird verwendet, um das Ende eines Instruktionsblocks zu signalisieren, der durch eine "St Loop"-Instruktion begonnen hat.
Call [e]	Aufruf einer Unterroutine; (e) bezeichnet die Eintrittstelle.
Ret	Rücksprung (Return) von einem Unterroutineaufruf.

Add (p1)(p2)	Füge Zeichen an den Stellen (p1) und (p2) hinzu, das Ergebnis wird in (p1) dargestellt und jeglicher Übertrag setzt ein Flag, das ">" anzeigt.
Sub (p1)(p2)	Substrahiere Zeichen an den Stellen (p1) und (p2), das Ergebnis wird in (p1) dargestellt, und jeglicher Übertrag setzt ein Flag, das "<" anzeigt.
Shf R (p1)	Bitverschiebungszeichen an der Stelle (p1) "rechts" (MSB (höchstwertiges Bit) auf LSB (niedrigstwertiges Bit)), wobei das LSB ausgesondert wird und das MSB mit "0" gefüllt wird.
Shf L (p1)	Bitverschiebungszeichen an der Stelle (p1) "links" (LSB auf MSB), wobei das MSB ausgesondert wird und das LSB mit "0" gefüllt wird.
AND (p1)(p2)	Logische UND-Zeichen an den Stellen (p1) und (p2).
OR (p1)(p2)	Logische ODER-Zeichen an den Stellen (p1) und (p2).
XOR (p1)(p2)	Logische XOR-Zeichen an den Stellen (p1) und (p2).
BCD2Bin (p1)[L]	Wandle numerische Daten beginnend an der Stelle (p1) von BCD- in Binärformat um. (L) bezeichnet die Länge der BCD-Daten.
Bin2BCD (p1)[L]	Wandle Binärdaten beginnend an der Stelle (p1) in BCD-Format um. (L) bezeichnet die Länge der Binärdaten.
Bin2ASC (p1)	Wandle Daten an der Stelle (p1) von Binär-Format in eine ASCII-Darstellung um. (ASCII Hex.)
ASC2BIN (p1)	Wandle zwei Zeichen beginnend an der Stelle (p1) von einer ASCII Hex-Darstellung in Binär-Format um.
ENC (p1)(p2)	Verschlüssele 8 Byte Daten beginnend an der Stelle (pl) unter Verwendung des DES-Verschlüsselungsschlüssels beginnend an der Stelle (p2).
DEC (p1)(p2)	Entschlüssele 8 Byte Daten beginnend an der Stelle (pl) unter Verwendung des DES-Verschlüsselungsschlüssels beginnend an der Stelle (p2).
PUT data Px	Übertrage der Instruktion folgende Daten an P1 oder P2.
MOV (p1)(p2)	Übertrage Daten an der Stelle (p1) zu (p2).
ICR Px	Inkrementiere P1 oder P2.
DCR Px	Dekrementiere P1 oder P2.
LDPRG	Lade ein anderes Codemodul in den Ausführungspuffer.

[0078] Bezüglich der vorstehend aufgeführten Liste sollte erwähnt werden:
1. Die Ausdrücke "Stellen" bezeichnen jeweils Speicherstellen.
2. Die Parameter [a], [f], [L], [n], [s] und [t] sind 8-Bit-Werte.
3. Die Parameter [e], (p1) und (p2) sind 12- oder 16-Bit-Werte.
4. Die Dateiidentifizierungen [f] liegen im Bereich von 0 bis 127.
5. Die Verzweigungs-"Offset"-Werte [n] liegen im Bereich von –128 bis 127, das MSB-Bit hat einen positiven oder negativen Wert. (MSB = 1 = –Ve).
6. Für Verarbeitungen, in denen (p1) und (p2) verwendet wird, wird das Ergebnis in (p1) dargestellt, außer für den Befehl Comp, bei dem kein Ergebnis erhalten wird.
7. (p1) und (p2) sind logische Namen für Werte, die auf Speicherstellen zeigen.
8. Die St Loop-Instruktion erstellt eine Kopie von [n] für die Loop Bk-Instruktion, um [n] zu dekrementieren, wenn dieser Wert erreicht wird. Obwohl die Kopie von [n] nicht "null" ist, wird durch Loop Bk eine Programmausführung zu den Instruktionen zu St Loop [n] zurückübertragen. Das Verschachteln von Schleifen ist auf 3 Ebenen beschränkt.
9. Die Unterroutine Nesting (Verschachteln) wird nicht bereitgestellt, es ist nur eine Ebene erlaubt.
10. Die Instruktionen PUT, ICR und DCR beeinflussen den Wert des logischen Namens und nicht die Speicherstelle, auf die sie zeigen.
11. Es sind 16 Paßwörter vorgesehen, so daß [n] in Pres P einen Wert zwischen 0 und 15 hat.
12. Wenn in Create [s] = 0 ist, wird das MSB der Datei-ID so gesetzt, daß angezeigt wird, daß die Datei Datensätze mit variabler Länge aufweist, so daß jeder Datensatz seine eigenen Zugriffsregeln aufweisen kann.
13 . Der Instruktion Add Rec muß nach einer Create-Instruktion mit [s] = 0 folgen, um die Dateistruktur zu erzeugen. Sie muß [n]-mal ausgeführt werden, um die Dateistruktur abzuschließen. Nachdem die Create-Instruktion mit [s] = 0 ausgegeben wurde, tritt die Vorrichtung in einen Zustand ein, in der die korrekte Anzahl

von Add Rec-Instruktionen ausgeführt werden muß, auch wenn die Stromversorgung unterbrochen ist. Alle anderen Dateioperationen werden mit einem Status zurückgewiesen, der die aktuelle Situation anzeigt.

[0079] Der RAM-Speicher **21** weist einen Kommunikationspuffer **44** auf, in dem diskrete Befehle zusammen mit ihren Parametern gespeichert sind, einen Ausführungspuffer **45** mit typischerweise 64 bis 128 Byte, in dem Programmmodule **43** gespeichert sind, und ein durch die Unterroutinen verwendetes Hilfsregister **47** für allgemeine Verarbeitungen.
[0080] Nachstehend werden die logischen Operationen der Spezialroutinen und ihre relative Wechselwirkung ausführlicher beschrieben, wobei der Betriebssystemkern so konstruiert ist, daß er auf eine Meldung vom seriellen Kommunikationsport **15** der Vorrichtung wartet. Wenn eine Meldung empfangen wird, überträgt der Betriebssystemkern sie in den Befehlspuffer **44**, und die Steuerung wird dann an den Befehls-Executor übergeben. Es wird nun vorausgesetzt, daß die Meldung den Format-Befehlscode aufweist, dem mit dem Befehlscode in Beziehung stehende Parameter folgen, wie in **Fig.** 4 dargestellt ist.
[0081] Die durch den Logikblock **49** dargestellte Verarbeitung des Befehls-Executors **49** veranlaßt die CPU **27**, den Befehlscode **51** im Puffer zu betrachten, ihn zu prüfen und ihn als realen Befehl zu validieren. Die dem Befehlscode folgenden Parameter **53** und der Puffer werden dann geprüft und es wird validiert, daß sie für den spezifischen Befehl **51** eine korrekte Länge und einen korrekten Datentyp aufweisen. Nach einer erfolgreichen Prüfung und Validierung wird, wie durch den Logikblock **55** dargestellt ist, der Befehlscode decodiert, indem er als ein Index in einer Tabelle von Unterroutineadressen verwendet wird, wobei die korrekte Adresse für eine anschließende Ausführung des Befehls ausgewählt wird, wie durch den Logikblock **57** dargestellt ist.
[0082] Das Ablaufdiagramm der Befehls-Executor-Unterroutine ist in **Fig.** 6 dargestellt und beginnt mit einem Schritt zum Warten auf einen Befehl, wie durch Block **101** dargestellt ist. Nachdem Information vom seriellen Kommunikationsport empfangen wurde, wie durch Block **103** dargestellt ist, bestimmt die Unterroutine anschließend im Entscheidungsblock **105**, ob die Information einen gültigen Befehl darstellt oder nicht. Wenn der Befehl kein gültiger Befehl ist, setzt die Unterroutine anschließend das Statusregister des Mikricomputers, um anzuzeigen, daß ein Befehlsfehler aufgetreten ist, wie durch Block **107** dargestellt ist, und gibt dann den Status über den Kommunikationsport **15** aus, wie durch Block **109** dargestellt ist. An diesem Punkt springt die Unterroutine zu Block **101** zurück, um auf einen anderen Befehl zu warten.
[0083] Wenn in Block **105** entschieden wird, daß der Befehl ein gültiger Befehl ist, bestimmt die Unterroutine anschließend, ob die dem Befehl zugeordneten Daten gültig sind oder nicht, wie durch Block **111** dargestellt ist. Wenn die Daten nicht gültig sind, setzt die Unterroutine das Statusregister des Mikrocomputers, um anzuzeigen, daß ein Parameterfehler aufgetreten ist, wie durch Block **113** dargestellt ist, und gibt dann den Status über den seriellen Kommunikationsport aus, wie durch Block **109** dargestellt ist, bevor sie zu Block **101** zurückspringt, um auf einen anderen Befehl zu warten.
[0084] Wenn die Unterroutine in Block **111** bestimmt, daß die Daten gültig sind, führt sie den Befehl aus, wie durch Block **115** dargestellt ist, und gibt dann die Antwort über die serielle Kommunikationsschnittstelle **15** aus, wie durch Block **109** dargestellt ist, bevor sie zu Block **101** zurückspringt, um auf den nächsten Befehl zu warten.
[0085] In der vorliegenden Ausführungsform wird ein Programmmodul **43** an die intelligente Vorrichtung **13** übertragen, wenn sie mit einer Schnittstellenvorrichtung **11**, z. B. einem POS-Endgerät, verbunden ist, und durch den Befehls-Executor **39** in den Ausführungspuffer **45** geladen. Diese Programmmodule **43** können in verschlüsselter Form übertragen werden und sind dazu geeignet, mit einer oder mehreren Datendateien **33** oder Datensätzen **33** davon zu arbeiten, die im EEPROM **19** gespeichert sind.
[0086] Ein Beispiel eines Programmmoduls ist das folgende:
1. Nimm 4 Tastenanschläge an.
2. Formatiere Daten in 8 Bytes.
3. Verschlüssele unter Verwendung der in Datei 2 gespeicherten Schlüsselnummer 3.
4. Vergleiche Ergebnis mit Schlüsselnummer 3 in Datei.
5. Inkrementiere Wert der Schlüsselnummer 3.
6. Gib Vergleichsergebnis zurück.

[0087] Gegebenenfalls können mehrere Programmmodule **43** miteinander verkettet werden, um eine erweiterte Leistungsfähigkeit des Systems bereitzustellen. Eine Verkettung ist aufgrund des zum Speichern von Programmmodulen verfügbaren begrenzten Speicherplatzes in der intelligenten Vorrichtung erforderlich. Daher ist die Menge der ausführbaren Verarbeitungen begrenzt, Programmmodule können jedoch sequentiell geladen werden, um eine erweiterte Verarbeitung zu ermöglichen.
[0088] Diese Programmmodule **43** sind bezüglich des Typs der Transaktion, die zwischen dem Service-Provider und dem Service-User eingerichtet werden soll, genauso anwendungsspezifisch wie die Datendateien **33**. Daher kann ein Programmmodul von der Schnittstellenvorrichtung **11** über den Koppler **14** in die intelligente Vorrichtung **13** geladen und durch den Programm-Interpreter **41** automatisch ausgeführt werden, ohne daß

der Programmablauf durch den Service-Provider oder den Service-User beobachtet wird, so daß ein hoher Sicherheitsgrad für jegliche sensitiven Daten bereitgestellt wird, die die Karte enthalten kann. Dieser hohe Sicherheitsgrad wird dadurch bereitgestellt, daß der Mikrocomputer im Einzelchip-Modus betrieben wird. Daher treten an den Anschlüssen oder Pins des Mikrocomputers keine "Instruktionsabruf"- und "Datenlese oder -schreib"-zyklen auf. Die Programme des Mikrocomputers werden vom internen Speicher, Masken-ROM-Speicher, RAM-Speicher oder EEPROM abgearbeitet. Keine der Programmausführungen kann elektrisch, elektronisch oder visuell beobachtet werden.

[0089] Es sind zwar komplizierte Geräte zum Beobachten von Programmausführungen in Mikroprozessorsystemen für Prüf- und Diagnosezwecke verfügbar, die zum Erfassen von Information, wie beispielsweise von Paßwörtern und Zugriffscodes, verwendbar sind, es ist jedoch nicht möglich, diese Geräte in Verbindung mit einem Einzelchip-Microcomputer zu verwenden.

[0090] Obwohl in der vorliegenden Ausführungsform Programmmodule im RAM-Speicher gespeichert sind, sind für bestimmte Transaktionen spezifische Programmmodule in einem nichtflüchtigen Speicher gespeichert, der entweder eine Zwischenspeicherfunktion oder eine permanente Speicherfunktion besitzt.

[0091] Der Programm-Interpreter führt ein geladenes Programmmodul nicht automatisch aus, es wird stattdessen nur in Antwort auf einen durch den Befehls-Executor ausgeführten spezifischen Befehl ausgeführt. Der Befehls-Executor **37** ist so konstruiert, daß, wenn das Programmmodul ausgeführt wird, die Steuerung der CPU **27** an den Programm-Interpreter **41** übergeben wird.

[0092] Der Programm-Interpreter **41** ist so konstruiert, daß, wenn er durch den Befehls-Executor **39** der intelligenten Vorrichtung angewiesen wird, die Instruktionsliste des Programmmoduls **43** auszuführen, er automatisch jede Instruktion vom Ausführungspuffer **45** abruft, ihn decodiert und entsprechende Verarbeitungen oder Operationen ausführt. Die Instruktionsliste kann bedingte Instruktionen enthalten, die den Programmablauf ändern, so daß die Instruktionsliste des Programmmoduls **43** nicht notwendigerweise in einer linearen Folge ausgeführt wird.

[0093] **Fig.** 7 zeigt die logische Struktur des Programm-Interpreters. Wie dargestellt, sind die Instruktionscodes und Daten im Ausführungspuffer **45** gespeichert, wo sie durch den Programm-Interpreter abgerufen und decodiert werden können. Diesbezüglich weist der Programm-Interpreter einen Adressenzähler **121** und ein Address-Latch **123** zum Adressieren des geeigneten Speicher-Bytes des Ausführungspuffers auf, in dem die abgerufene Instruktion oder die abgerufenen Daten gespeichert sind. Das adressierte Byte des Ausführungspuffers **45** wird in Abhängigkeit davon, ob vermutet wird, daß das Byte eine Instruktion oder Daten enthält, in einen Instruktions-Latch **125** oder einen Daten-Latch **127** geladen.

[0094] Der Programm-Interpreter ist so konstruiert, daß vorausgesetzt wird, daß das erste Byte des im Ausführungspuffer **45** gespeicherten Programmmoduls ein Instruktionscode ist. Dieser Instruktionscode würde dann im Instruktions-Latch gespeichert, wodurch eine Instruktionstyptabelle **129** zunächst die Instruktion decodiert, um zu bestimmen, ob durch den Adressenzähler **121** und den Address-Latch **123** mehr Adressen erzeugt werden müssen, um Daten im Daten-Latch **127** zu speichern. In Abhängigkeit von der Bestimmung der Instruktion durch die Instruktionstyptabelle **129** wird der Instruktions-Latch **125** oder der Daten-Latch **127** gesteuert, wie durch die Latch-Steuerungslinien **131** und **133** dargestellt ist. Dann wird ein Instruktionsdecodierer **135** aktiviert, um unter Verwendung des im Instruktions-Latch **125** gespeicherten Instruktionscodes als Index eine Tabelle von Unterroutineadressen zu durchsuchen. Dann wird die relevante Unterroutine ausgeführt, und wenn sie beendet ist, inkrementiert die Instruktionstyptabelle den Adressenzähler **121**, um auf die nächste Instruktion des im Ausführungspuffer **45** gespeicherten Programmmoduls zuzugreifen.

[0095] Instruktionen werden sequentiell vom Puffer abgerufen, bis durch das Ergebnis einer Instruktion veranlaßt wird, daß der Adressenzähler mit einem anderen Wert neu geladen wird, der den Weg des Instruktionsablaufs ändert. Instruktionen werden automatisch abgerufen, bis das Ende des Puffers erreicht ist oder eine Ende-Instruktion erfaßt wird. In beiden Fällen wird die Steuerung wieder an den Befehls-Executor übergeben, wobei durch die Ende-Instruktion veranlaßt wird, daß der Befehls-Executor wieder auf den Wartezustand eingestellt wird, in dem er auf eine weitere über den seriellen Kommunikationsport **15** zugeführte Kommunikation wartet, wohingegen durch die "Pufferende"-Instruktion veranlaßt würde, daß der Befehls-Executor mehr Daten für den Ausführungspuffer erhält, woraufhin die Steuerung wieder an den Programm-Interpreter übergeben wird.

[0096] Der Adressenzähler **121** führt dem Ausführungspuffer **45** sequentiell Adressen über den Address-Latch **123** zu, wobei er normalerweise durch die Instruktionstyptabelle **129** inkrementiert wird. Der Adressenzähler **121** kann jedoch durch die Instruktionstyptabelle **129** oder den Decodierer **135** neu geladen werden, und in den Address-Latch **123** kann durch den Decodierer **135** direkt geschrieben werden.

[0097] Der Instruktionsdecodierer **135** aktiviert die Verarbeitungen für einen Code, z. B. Lese- oder Schreiboperationen oder Ausführen eines Tests oder Herbeiführen einer Entscheidung, und aktualisiert kontinuierlich den Status-Latch **137**, um seinen Status zu einem beliebigen Zeitpunkt anzuzeigen.

[0098] **Fig.** 8 zeigt die Operations- oder Verarbeitungszustände des Programm-Interpreters **41**, gemäß denen der Programm-Interpreter entweder nicht abgearbeitet wird, wie durch Block **141** dargestellt, eine Instruktion

abruft, wie durch Block **143** dargestellt, weitere Daten nach Erfordernis abruft, wie durch Block **145** dargestellt, oder eine Instruktion oder Instruktionen ausführt, wie durch Block **147** dargestellt, wodurch er in einem Zyklus zum Abrufen einer Instruktion in Block **143** zurückspringt, bis alle im Programmmodul enthaltenen Instruktionen ausgeführt sind, wie durch Block **149** dargestellt ist. Nach Abschluß der Ausführung des Programmmoduls wird der Programm-Interpreter zu Block **141** zurückspringen, gemäß dem er nicht abgearbeitet wird.

[0099] **Fig.** 9 zeigt das tatsächliche Ablaufdiagramm, das die Operation oder Verarbeitung des Programm-Interpreters der Unterroutine darstellt. Nach dem Start der Unterroutine, wie durch Block **151** dargestellt, wird der erste Instruktionscode vom Ausführungspuffer abgerufen, wie durch Block **153** dargestellt ist. Die Unterroutine bestimmt dann, ob er ein gültiger Instruktionscode ist, wie durch den Entscheidungsblock **155** dargestellt ist, und wenn dies nicht der Fall ist, setzt die Unterroutine den Ausführungsstatus des Mikrocomputers, um anzuzeigen, daß der Instruktionscode "ungültig" war, wie durch Block **157** dargestellt, woraufhin die Verarbeitung der Unterroutine abgeschlossen ist, wie durch Block **159** dargestellt, und die Steuerung wieder an den Befehls-Executor übergeben wird. Wenn festgestellt wird, daß der Instruktionscode gültig ist, bestimmt die Unterroutine anschließend, ob irgendwelche Daten vorhanden sind, die dem Instruktionscode zugeordnet sind, wie durch den Entscheidungsblock **161** dargestellt ist. Wenn dies nicht der Fall ist, schreitet die Unterroutine direkt fort, um die Instruktionen auszuführen, wie durch Block **163** dargestellt ist, wenn jedoch Daten vorhanden sind, ruft die Unterroutine Daten vom Puffer und das Byte oder die Bytes ab, die der Adresse der aktuellen Instruktion folgen, wie durch Block **165** dargestellt ist, und führt anschließend die durch Block **163** dargestellte Instruktion aus. Nachdem die Ausführung der Instruktion abgeschlossen ist, bestimmt die Unterroutine, wie durch Entscheidungsblock **167** dargestellt ist, ob das Ende des Ausführungspuffers erreicht ist, wobei, wenn dies nicht der Fall ist, die Unterroutine den nächsten Instruktionscode vom Ausführungspuffer abruft, wie durch Block **153** dargestellt ist. Wenn die Unterroutine feststellt, daß das Ende des Ausführungspuffers erreicht ist, setzt die Unterroutine den Ausführungsstatus auf "gut", wie durch Block **169** dargestellt ist, und die Unterroutine wird dann beendet, wie durch Block **159** dargestellt ist, woraufhin die Steuerung wieder an den Befehls-Executor übergeben wird.

[0100] Der Programm-Interpreter übernimmt tatsächlich eine Liste von Binärcodes und verwendet sie, wie durch den Instruktionssatz dargestellt wird, entweder als auszuführende Instruktionen, als Daten, mit denen gearbeitet werden soll, Adressen, von denen Daten gewonnen werden sollen, als Daten, gemäß denen Entscheidungen getroffen oder Änderungen im Ausführungsweg vorgenommen werden sollen, usw. Jedes 8-Bit-Byte wird eine Instruktion oder einen Datenteil oder einen Teil einer Adressenspeicherstelle oder eine Offset-Adresse darstellen. Gemäß diesem System wird das Programmmodul **43** unter Verwendung von Instruktionen geschrieben, die interpretiert werden und auf denen basierend der Programm-Interpreter **41** arbeitet und die keinen Maschinencode der CPU **27** enthalten, wodurch ansonsten möglicherweise eine Sicherheitslücke entstehen könnte. Daher wird jedes Programmmodul gezwungen, nur die Operationen oder Verarbeitungen auszuführen, die der Programm-Interpreter **43** zuläßt. Der Grund hierfür ist, daß der Maschinencode des Mikrocomputers alle Verarbeitungen ausführen kann, die er möchte, wohingegen der Programm-Interpreter so konstruiert ist, daß er die Verarbeitungen bezüglich Daten und den Zugriff auf Daten und die Darstellung von Daten auf eine Weise beschränkt, gemäß der keine Sicherheitslücke entstehen kann. Dies wird durch den vorstehend beschriebenen spezifischen Instruktionssatz erreicht, und indem veranlaßt wird, daß der Interpreter den Speicher auf die gleiche Weise wie der Befehls-Executor den Speicher betrachten würde, nicht als eine Speicherstelle, sondern eher als Register, die vom Speicher logisch abgebildet werden, und als Dateispeicherbereiche betrachtet.

[0101] Der Maschinencode könnte verwendet werden, um auf einfache Weise Daten von einer Datei zu lesen, die Verschlüsselungsschlüssel enthält, weil die Datei aufgrund des Maschinencodes nur ein anderer Satz Speicherstellen im Speicher wäre. Andererseits würde der Programm-Interpreter sehen, daß die Datei, die diese Schlüssel enthält, eine Nur-Schreib-Datei ist, und könnte sie daher nicht lesen, sondern anstatt des Schlüssels selbst lediglich die Adresse des Schlüssels in der Datei als Eingabe für eine Verschlüsselungsroutine verwenden.

[0102] Obwohl ein Programm in Maschinencode geschrieben werden könnte, das Sicherheitsinformation lesen und übertragen könnte, z. B. Verschlüsselungsschlüssel, verfügt die erfindungsgemäße intelligente Vorrichtung über keine Einrichtungen oder Mittel, um diesen Maschinencode auszuführen, mit Ausnahme ihres Masken-ROM-Speichers, so daß ein solches Programm von einer externen Quelle in der intelligenten Vorrichtung nicht ausgeführt werden kann.

[0103] Der Instruktionssatz für den Programm-Interpreter **41** weist in der vorliegenden Ausführungsform Instruktionen insbesondere zum Verschlüsseln und für ein Schlüsselmanagement zusammen mit Instruktionen zur Datenspeicherung, -manipulation und -Prüfung auf, wobei der letztgenannte Instruktionssatz in seiner Verarbeitung durch Zugriffsbeschränkungen eingeschränkt ist, die durch die Datendateien **33** und ihre Datensätze **35** bereitgestellt werden.

[0104] Nachstehend wird die Funktionsweise des Systems zum Ausführen von Transaktionen, während die IC-Karte **13** mit einer Schnittstellenvorrichtung **11** verbunden ist, zunächst bezüglich der Datenspeicherung

und -manipulation beschrieben, wobei anfangs Datendateien **33** erzeugt werden, die mehrere Datensätze aufweisen, die Daten und Information enthalten, die für eine zwischen einem Service-Provider und einem Service-User auszuführenden spezifischen Transaktion relevant sind. Sicherheit wird durch die Fähigkeit bereitgestellt, Beschränkungen für den Zugriff auf und die Erzeugung von Dateien aufzuerlegen, so daß die Anforderungen für den PIN- und/oder den Challenge-Code so konfiguriert werden können, daß Dateien gelesen oder geschrieben oder erzeugt werden können. Individuellen Dateien und ihren Datensätzen können verschiedene Beschränkungsgrade zugeordnet werden. Beispielsweise kann eine Datei als Nur-Schreib-Datei mit einem Zugriff nur über den Challenge-Code oder als Nur-Lese-Datei mit einem Zugriff konfiguriert sein, der nur über die PIN und die Challenge-Code-Sequenz freigegeben wird.

[0105] Ein praktisches Beispiel hierfür ist, wenn eine Datei mit Zugriffsbeschränkungen Daten enthält, die eine Postadresse zum Weiterleiten der Karte enthält, wenn diese verloren geht. Der Header dieser Datei wäre codiert, um einen freien Zugriff darauf zu ermöglichen. Anderen Dateien, die den Namen, die Adresse und die Telefonnummer des Benutzers enthalten, könnten jedoch verschiedene Zugriffsgrade zugeordnet sein, um zu ermöglichen, daß vielleicht nur auf den Namen und die Adresse, nicht jedoch auf die Telefonnummer zugegriffen werden kann. Die Header dieser Dateien oder Datensätze wären geeignet mit Zugriffsbeschränkungen codiert, um verschiedene Zugriffsgrade zu erhalten.

[0106] Es können separate Dateispeicherbereiche im EEPROM **19** erzeugt werden, um zu ermöglichen, daß die IC-Karte **13** für verschiedene Anwendungen auf sichere Weise verwendet werden kann. Diese Partitionierung des EEPROM **19** wird unter Verwendung eines "Datei"-Verzeichnisses oder der Adresse der nächsten Datei im Header der vorangehenden Datei realisiert, wodurch gewährleistet wird, daß jede Anwendung nur auf die relevanten Dateien **33** zugreifen kann.

[0107] Beispielsweise könnten die in der folgenden Liste aufgeführten Anwendungen in einer IC-Karte verarbeitet werden:

Banking-Transaktionen und EFTPOS
Bezahlung von Rechnungen
Verschreibung von Medikamenten
Erstattung von Beihilfen im Gesundheitswesen
Zusammenfassung des Krankheitsverlaufs
Club-Mitgliedschaften
Ticketing-Systeme
Bonuspunkt-Systeme
Fahrzeugwartungsverlauf, etc.

[0108] Wie vorstehend beschrieben wurde, können den individuellen Datensätzen **35** einer Datei **33** Zugriffsbeschränkungen getrennt von der Gesamtdatei selbst zugeordnet sein, wodurch Speicherplatz eingespart werden kann. Beispielsweise würden bezüglich des vorstehend erwähnten Beispiels, in dem das Speichern von Daten für die Postadresse, den Namen, die Wohnadresse und die Telefonnummer beschrieben wurde, diese Daten in vorhandenen IC-Karten normalerweise in separaten Dateien gespeichert sein, um zu ermöglichen, daß ihnen verschiedene Zugriffsbeschränkungen zugeordnet werden können. In der erfindungsgemäßen visuellen IC-Karte könnten diese Daten alle in einer Datei vermischt und als separate Datensätze in dieser Datei behandelt werden, wodurch den separaten Datensätzen verschiedene Zugriffsbeschränkungen zugeordnet werden, um die Sicherheit des Systems zu gewährleisten, jedoch durch Reduzieren der Anzahl von Dateien Speicherplatz einzusparen.

[0109] Die Ausführung einer Transaktion dreht sich um die Verwendung von Programmmodulen **43**, die für die Zwecke des Service-Providers und des Service-Users vorprogrammiert wurden und die permanent im der Schnittstellenvorrichtung zugeordneten Speicher gespeichert sind. Diese Programmmodule **43** werden auf die IC-Karte downgeloadet, um die spezifische Transaktion zu steuern, die abgewickelt soll, nachdem die Transaktion anfangs eingerichtet worden ist, wie vorstehend beschrieben wurde.

[0110] In der vorliegenden Ausführungsform sind die Programmmodule **43** als verschlüsselte Daten im Koppler **14** gespeichert, und nachdem sie über den Ein-/Ausgangs-Port 15 zum RAM-Speicher **21** der Karte übertragen wurden, werden sie unter Verwendung des relevanten Algorithmus des Verschlüsselungsverarbeitungssystems für eine anschließende Interpretation durch die Programm-Interpreter-Routine **41** entschlüsselt. In anderen Ausführungsformen kann der gesamte Transaktionsprozeß von einem entfernten Host aus gesteuert werden, wobei die Programmmodule **43** nach Erfordernis über eine Leitung übertragen werden, wobei sie in diesem Fall nicht im Koppler gespeichert werden müssen.

[0111] Nach Abschluß der Verarbeitung des Programmmoduls löscht das Betriebssystem das Programmmodul automatisch vom RAM-Speicher **21**, um eine nachfolgende, nicht autorisierte Verwendung des Programmmoduls zu verhindern und dadurch die Sicherheitsanforderungen des Service-Providers zu erfüllen. Die automatische Verarbeitung des Programm-Interpreters und das Löschen des Programmmoduls (der Programmmodule) durch das Betriebssystem können aufgrund der Einzelchip-Ausführungsform des Mikrocomputers in der

intelligenten Vorrichtung und der nachfolgenden Verarbeitungs- oder Operationszustände des Mikrocomputers im "Befehlsmmodus"-Zustand weder durch den Service-Provider noch durch den Service-User außer Kraft gesetzt werden, nachdem die Transaktion eingerichtet und ausgeführt wurde. Außerdem führt der Mikrocomputer eine der folgenden Funktionen oder Operationen aus:
1. Warten auf eine Menüpunktauswahl.
2. Warten auf einen Befehl.
3. Ausführen eines Befehls.
4. Interpretieren eines Programmmoduls.

[0112] Wenn der Programm-Interpreter ein Programmmodul interpretiert, kann der Programmablauf nicht umgeleitet oder abgeändert werden, sondern der Programm-Interpreter führt das Programm bis zum Ende aus oder bis ein Fehler auftritt oder bis ein Abbruchbefehl ausgegeben wird.
[0113] Ein wichtiger Vorteil der vorliegenden Erfindung ist, daß Programmmodule geändert werden können, wenn Updates oder Erweiterungen an der Seite des Service-Providers erforderlich sind, ohne daß die IC-Karte selbst aktualisiert oder erweitert werden muß. Außerdem wird die gesamte Transaktion durch das Programmmodul gesteuert, wodurch die Fehlerwahrscheinlichkeit minimiert und die Ausführung der Transaktion erleichtert wird. Außerdem kann für jede "Anwendung", die durch die IC-Karte unterstützt wird, ein andersartiges Datendarstellungsverfahren vorgesehen sein, das so angepaßt ist, daß es den Erfordernissen der spezifischen Anwendung entspricht. Die Datendarstellung ist nicht auf den ASCII-Standard beschränkt, bei dessen Verwendung in einigen Fällen, z. B. bei Verwendung einer IC-Karte, Speicherplatz verschwendet wird, so daß dieses Problem durch Verwendung anwendungsspezifischer Datendarstellungen reduziert werden kann. Außerdem können durch verschiedene "Anwendungen", z. B. Schlüsselmanagement, usw. andere Sicherheitsschemas implementiert werden.
[0114] Die Funktionsweise und die Vorteile der Verwendung des beschriebenen Systems zum Einrichten und Ausführen einer Transaktion werden unter Bezug auf das folgende Beispiel einer Mehrzweckanwendung der IC-Karte in Verbindung mit mehreren Service-Providern verdeutlicht. In diesem Beispiel ist die IC-Karte zum Ausführen einer Transaktion im medizinischen Bereich vorgesehen, wodurch separate Datendateien erzeugt werden, von denen eine eine Zusammenfassung des Krankheitsverlaufs des Benutzers, eine andere Information über Rezepte und eine andere Information über finanzielle Transaktionen enthält.
(1) Eine Person, die eine visuelle IC-Karte besitzt, geht zu einem Arzt, z. B. zu einem Arzt, den sie normalerweise nicht aufsucht. Nachdem die Anfangsidentifizierungsroutine unter Verwendung der Karte ausgeführt wurde, kann der Arzt die Karte mit seiner Schnittstellenvorrichtung verbinden, die ein vorgegebenes Programmmodul an die Karte überträgt. Dieses Programmmodul ermöglicht es dem Arzt anschließend, die Zusammenfassung des aktuellen Krankheitsverlaufs anzuschauen. Basierend auf Information vom Patienten und der in der Karte gespeicherten Zusammenfassung des Krankheitsverlaufs kann der Arzt eine Diagnose erstellen und ein Rezept verordnen. Der Arzt kann dann seine Diagnose, das Rezept und eine Konsultationsgebühr dokumentieren und diese Daten über die Schnittstellenvorrichtung für eine anschließende Aufzeichnung in auf der Karte bereitgestellten relevanten Datendateien übertragen. All diese Verarbeitungen werden durch ein in der Karte residentes Programmmodul gesteuert, so daß diese Information von der Schnittstellenvorrichtung zur Karte übertragen und anschließend durch den Programm-Interpreter automatisch verarbeitet wird. Nach Abschluß der Verarbeitung des Programmmoduls oder der Programmmodule wird das Programmmodul (die Programmmodule) vollständig von der Karte gelöscht.
(2) Die Person kann dann eine Apotheke aufsuchen und erneut die IC-Karte vorlegen, um das Rezept vorzulegen. Der Benutzer durchläuft erneut die Identifizierungsroutine und verbindet die Karte mit der Schnittstellenvorrichtung der Apotheke. Durch Übertragen und Interpretieren eines Programmmoduls auf eine ähnliche Weise wie vorstehend unter Bezug auf den Arzt beschrieben wurde kann die Apotheke die relevante Rezeptinformation lesen und die für die Person verschriebenen relevanten Medikamente ausgeben. Einrichtungen, die von der Apotheke beauftragt sind, die Anzahl von Wiederholungsrezepten und die durch die Apotheke eingegebenen Kosten aufzuzeichnen, werden durch die Steuerung des Programmmoduls in die Karte eingegeben.
(3) Die Person geht schließlich zu ihrer Krankenversicherungsagentur, um eine Erstattung der Konsultationsgebühr des Arztes und der Medikamentenkosten zu beantragen, wobei ähnliche Identifizierungs- und Datenaustauschverarbeitungen ausgeführt werden.

[0115] Die durch die Verwendung der visuellen IC-Karte und das Szenario bereitgestellten wichtigen Vorteile sind: 1) bezüglich des Arztes, daß es dem Arzt durch eine Zusammenfassung des Krankheitsverlaufs ermöglicht wird, bessere Entscheidungen hinsichtlich der Verschreibung von Medikamenten zu treffen, um zu gewährleisten, daß sie nicht mit Medikamenten, die der Patient bereits einnimmt, oder mit einem anderen Gesundheitszustand, den der Patient möglicherweise hat, in Konflikt stehen; 2) bezüglich der Medikamentenausgabe kann die visuelle IC-Karte den Patienten und den Arzt identifizieren und somit einen Schutz vor betrügerisch ausgestellten Rezepten bereitgestellt werden, und außerdem wird das Medikament und die Dosierung

klar identifiziert, wodurch verhindert wird, daß ein falsches Medikament und falsche Dosierungsanweisungen ausgegeben werden; und 3) wird durch Aufzeichnen der Konsultationsgebühr und der Medikamentenkosten in der visuellen IC-Karte ein bequemes und möglicherweise ein sichereres Verfahren für eine Bezahlung und zum Erhalten einer Rückerstattung von der Krankenversicherungsgesellschaft bereitgestellt.

[0116] Die Datendatei für eine finanzielle Transaktion kann mit einem spezifischen Finanzinstitut verknüpft sein, so daß das Institut dem Benutzer ein Guthaben zur Verfügung stellen kann, nachdem er dieses bezahlt hat, von dem die Konsultationsgebühr und die Medikamentenkosten durch den entsprechenden Arzt und die Apotheke abgebucht werden und die anschließend teilweise oder vollständig durch die Krankenversicherungsgesellschaft wieder gutgeschrieben werden.

[0117] In diesem Szenario werden aufgrund der Anforderung zur Bewahrung der Privatsphäre und der Sicherheitsanforderungen verschiedene Datendateien verschiedene Zugriffsbeschränkungsanforderungen aufweisen, wie nachstehend dargestellt ist.

Medizinisch	Aktueller Krankheitsverlauf	Rezept	Kosten
Arzt	Lesen und Hinzufügen	Lesen und Hinzufügen	Nur Hinzufügen
Apotheke	Kein Zugriff	Lesen und Modifizieren	Nur Hinzufügen
Kranken-versicherung	Kein Zugriff	Kein Zugriff	Lesen und Modifizieren

[0118] Die Zugriffsbeschränkung "Hinzufügen" dient dazu, zu gewährleisten, daß der Arzt in den Krankheitsverlauf-, Rezept- oder Krankheitskostendateien keine Änderung in den vorhandenen Datensätzen vornehmen kann.

[0119] Die Zugriffsbeschränkung "Modifizieren" dient dazu, zu gewährleisten, daß die Apotheke lediglich die Anzahl von Wiederholungen eines bestimmten verordneten Rezepts reduzieren kann oder die Krankenversicherungsgesellschaft lediglich eine Markierung, wie beispielsweise "Gebühr erstattet" für eine bestimmte Gebühr erstellen kann.

[0120] Wie anhand des Beispiels ersichtlich ist, werden, weil unter Verwendung der visuellen IC-Karte mehrere Anwendungen unterstützt werden, Anforderungen für die Datendarstellung, die Verarbeitung, die Bewahrung der Privatsphäre und der Sicherheit bereitgestellt, die für jede Anwendung verschieden und spezifisch sind. Weil die IC-Karte Programmmodule ausführt, die vom Koppler in die IC-Karte geladen werden, ist ein System denkbar, das die Anforderungen mehrerer Anwendungen in einer Karte mit einer hohen Flexibilität und einem hohen Sicherheitsgrad leicht berücksichtigen kann, ohne daß der Koppler kundenspezifisch programmiert werden muß.

[0121] Der Schutzumfang der vorliegenden Erfindung ist nicht auf die spezifische beschriebene Ausführungsform beschränkt.

Patentansprüche

1. System zum Vornehmen einer Transaktion mit:

(i) einer selbständigen, tragbaren, intelligenten Vorrichtung (**13**) mit einem Einzelchip-Mikrocomputer (**27**) mit einem Ein-/Ausgabe-Kommunikationsanschluß (**15**), einem nichtflüchtigen Speicher (**17**, **19**) und einem Direktzugriffsspeicher (**21**) und

(ii) einer Schnittstellenvorrichtung (**11**), die einem Speicher zum Speichern von Daten zugeordnet ist, und einem Koppler (**14**) zum Anschließen an die intelligente Vorrichtung, um eine Kommunikation mit dieser herzustellen,

wobei der nichtflüchtige Speicher so eingerichtet ist, daß ein Teil von diesem (**17**) eine Betriebssystemmaske aufweist, die in Maschinencode für den Mikrocomputer programmiert ist, um Grundfunktionen auszuführen, und ein zweiter Teil von diesem (**19**) Datendateien (**33**) speichern kann, die verschiedene Zugriffsbeschränkungsebenen aufweisen,

wobei der Direktzugriffsspeicher (**21**) zur Verwendung durch das Betriebssystem (**37**), wenn eine Funktion ausgeführt wird, und zum Speichern von Daten, die vom Kommunikationsanschluß empfangen wurden oder zur Übertragung über den Kommunikationsanschluß bereit sind, vorgesehen ist, und

wobei die Schnittstellenvorrichtung ein Programmodul (**43**) innerhalb des Speichers aufweist, das eine oder mehrere Instruktionen zum Bearbeiten vorgeschriebener Datendateien innerhalb des zweiten Teils des nichtflüchtigen Speichers entsprechend dem Betriebssystem nach dem Anschluß der intelligenten Vorrichtung an den Koppler aufweist,

wobei das Betriebssystem aufweist: (a) einen Befehlsausführer (**39**) zum Empfangen eines Befehls, zum Ausführen einer vorgeschriebenen Funktion bezüglich des Befehls und zum Bereitstellen eines Ergebnisses oder

Status für den Befehl und (b) einen Programminterpretierer (**41**) zum Ausführen des Programmoduls, um eine Transaktion vorzunehmen, wobei die eine Instruktion oder die mehreren Instruktionen einen Teil eines vorgeschriebenen Instruktionssatzes bilden, der von dem Maschinencode abgesetzt ist, und wobei sie beschränkte Steuerfunktionen aufweisen, um einen Datendateizugriff zu beschränken.

2. System nach Anspruch 1, wobei das Programmodul über den Kommunikationsanschluß ansprechend auf einen vorgeschriebenen, vom Befehlsausführer empfangenen Befehl durch den Befehlsausführer in den Direktzugriffsspeicher geladen wird und von diesem ausgeführt wird, wenn die intelligente Vorrichtung an den Koppler angeschlossen wird, um ihn nachfolgend durch den Programminterpretierer auszuführen.

3. System nach Anspruch 1, wobei das Programmodul über den Kommunikationsanschluß ansprechend auf einen vorgeschriebenen, vom Befehlsausführer empfangenen Befehl durch den Befehlsausführer in einen dritten Teil des nichtflüchtigen Speichers geladen wird und von diesem ausgeführt wird, wenn die intelligente Vorrichtung an den Koppler angeschlossen wird, um ihn durch den Programminterpretierer auszuführen.

4. System nach einem der vorhergehenden Ansprüche, wobei das Programmodul verschlüsselt ist und das Betriebssystem ein Datenverschlüsselungs- und -entschlüsselungsprogramm zum Entschlüsseln des Programmoduls entsprechend einem vorgeschriebenen Algorithmus aufweist.

5. System nach Anspruch 4, wobei der Algorithmus der Datenverschlüsselungsstandard-Algorithmus (D.E.S) oder der RSA-Algorithmus ist.

6. System nach einem der vorhergehenden Ansprüche, wobei die intelligente Vorrichtung eine Tastatur (**23**) und eine Anzeige (**26**) aufweist.

7. System nach einem der vorhergehenden Ansprüche, wobei mehrere der Programmodule nacheinander in den Direktzugriffsspeicher geladen und miteinander verkettet werden, um die Fähigkeiten des Systems zu erweitern.

8. System nach Anspruch 7, wobei die Programmodule verschlüsselt sind und das Betriebssystem ein Datenverschlüsselungs- und -entschlüsselungsprogramm zum Entschlüsseln der Programmodule im Blockverkettungsmodus entsprechend einem vorgeschriebenen Algorithmus aufweist.

9. System nach einem der vorhergehenden Ansprüche, wobei die Datendateien jeweils Datensätze aufweisen, wodurch verschiedene Datensätze innerhalb einer Datei verschiedene Zugriffsbeschränkungsebenen aufweisen können.

10. System nach Anspruch 9, wobei die Datendateien und Datensätze jeweils Kopfteile zum Spezifizieren einer Zugriffsbeschränkungsebene aufweisen.

11. System nach Anspruch 10, wobei die Zugriffsbeschränkungsebene ein Paßwort aufweist.

12. System nach einem der vorhergehenden Ansprüche, wobei das Programmodul ein Datendarstellungsverfahren zulassen kann, das zur Verwendung für die spezifische Anwendung geeignet ist.

13. System nach einem der vorhergehenden Ansprüche, wobei der Betrieb des Programminterpretierers durch den Befehlsausführer ansprechend auf einen von diesem empfangenen und ausgeführten vorgeschriebenen Befehl aufgerufen wird.

14. Selbständige, tragbare, intelligente Vorrichtung (**13**) zum Vornehmen einer Transaktion mit einer Schnittstellenvorrichtung (**15**), die einem Speicher zum Speichern von Daten zugeordnet ist, und einem Koppler (**14**) zum Anschließen an die intelligente Vorrichtung, um eine Kommunikation mit dieser herzustellen, mit einem Einzelchip-Mikrocomputer (**27**) mit einem Ein/Ausgabe-Kommunikationsanschluß (**15**), einem nichtflüchtigen Speicher (**17**, **19**) und einem Direktzugriffsspeicher (**21**), wobei.
(i) der nichtflüchtige Speicher so eingerichtet ist, daß ein Teil von diesem (**17**) eine Betriebssystemmaske aufweist, die in Maschinencode für den Mikrocomputer programmiert ist, um Grundfunktionen auszuführen, und ein zweiter Teil von diesem (**19**) Datendateien (**33**) speichern kann, die verschiedene Zugriffsbeschränkungsebenen aufweisen,
(ii) der Direktzugriffsspeicher (**21**) zur Verwendung durch das Betriebssystem (**37**), wenn eine Funktion ausgeführt wird, und zum Speichern von Daten, die vom Kommunikationsanschluß empfangen wurden oder zur

Übertragung über den Kommunikationsanschluß bereit sind, vorgesehen ist,
(iii) das Betriebssystem aufweist: (a) einen Befehlsausführer (**39**) zum Empfangen eines Befehls, zum Ausführen einer vorgeschriebenen Funktion bezüglich des Befehls und zum Bereitstellen eines Ergebnisses oder Status für den Befehl und (b) einen Programminterpretierer (**41**) zum Ausführen des Programmoduls, um eine Transaktion vorzunehmen,
(iv) das Programmodul (**43**) eine oder mehrere Instruktionen zum Bearbeiten vorgeschriebener Datendateien innerhalb des zweiten Teils des nichtflüchtigen Speichers entsprechend dem Betriebssystem aufweist und
(v) die eine Instruktion oder die mehreren Instruktionen einen Teil eines vorgeschriebenen Instruktionssatzes bilden, der von dem Maschinencode abgesetzt ist, und wobei sie beschränkte Steuerfunktionen aufweisen, um den Datendateizugriff zu beschränken.

15. Intelligente Vorrichtung nach Anspruch 14, wobei das Programmodul über den Kommunikationsanschluß vom Koppler der Schnittstellenvorrichtung in den Direktzugriffsspeicher geladen wird, wenn die intelligente Vorrichtung an den Koppler angeschlossen wird, um es durch das Betriebssystem auszuführen.

16. Intelligente Vorrichtung nach Anspruch 14, wobei das Programmodul über den Kommunikationsanschluß vom Koppler der Schnittstellenvorrichtung in einen dritten Teil des nichtflüchtigen Speichers geladen wird, wenn die intelligente Vorrichtung an den Koppler angeschlossen wird, um es durch das Betriebssystem auszuführen.

17. Intelligente Vorrichtung nach einem der Ansprüche 14 bis 16, wobei das Programmodul verschlüsselt ist und das Betriebssystem ein Datenverschlüsselungs- und -entschlüsselungsprogramm zum Entschlüsseln des Programmoduls entsprechend einem vorgeschriebenen Algorithmus aufweist.

18. Intelligente Vorrichtung nach einem der Ansprüche 14 bis 17, wobei mehrere der Programmodule nacheinander in den Direktzugriffsspeicher geladen und miteinander verkettet werden, um die Fähigkeiten des Systems zu erweitern.

19. Intelligente Vorrichtung nach Anspruch 18, wobei die Programmodule verschlüsselt sind und das Betriebssystem ein Datenverschlüsselungs- und -entschlüsselungsprogramm zum Entschlüsseln der Programmodule im Blockverkettungsmodus entsprechend einem vorgeschriebenen Algorithmus aufweist.

20. Intelligente Vorrichtung nach einem der Ansprüche 14 bis 19, wobei die Datendateien jeweils Datensätze aufweisen, wodurch verschiedene Datensätze innerhalb einer Datei verschiedene Zugriffsbeschränkungsebenen aufweisen können.

21. Intelligente Vorrichtung nach Anspruch 20, wobei die Datendateien und Datensätze jeweils Kopfteile zum Spezifizieren einer Zugriffsbeschränkungsebene aufweisen.

22. Intelligente Vorrichtung nach Anspruch 21, wobei die Zugriffsbeschränkungsebene ein Paßwort aufweist.

23. Intelligente Vorrichtung nach einem der Ansprüche 14 bis 22, wobei das Programmodul ein Datendarstellungsverfahren zulassen kann, das zur Verwendung für die spezifische Anwendung geeignet ist.

24. Intelligente Vorrichtung nach einem der Ansprüche 14 bis 23, wobei der Betrieb des Programminterpretierers durch den Befehlsausführer ansprechend auf einen von diesem empfangenen und ausgeführten vorgeschriebenen Befehl aufgerufen wird.

25. Verfahren zum Vornehmen einer Transaktion zwischen einer intelligenten Vorrichtung (**13**) nach einem der Ansprüche 14 bis 24 und der dort angegebenen Schnittstellenvorrichtung (**11**), aufweisend: Erzeugen eines Programmoduls mit einer oder mehreren Instruktionen zum Bearbeiten vorgeschriebener Datendateien innerhalb der intelligenten Vorrichtung, Speichern des Programmoduls in dem Speicher (**21**) der Schnittstellenvorrichtung, Koppeln der intelligenten Vorrichtung mit der Schnittstellenvorrichtung und Aufrufen eines Programminterpretierers (**41**), der Teil des Betriebssystems (**37**) der intelligenten Vorrichtung ist, der die Instruktionen des Programmoduls interpretiert und ausführt, wobei die eine Instruktion oder die mehreren Instruktionen Teil eines vorgeschriebenen Instruktionssatzes sind, der von dem Maschinencode für den Mikrocomputer der intelligenten Vorrichtung abgesetzt ist, und wobei sie beschränkte Steuerfunktionen aufweisen, um den Datendateizugriff zu beschränken.

26. Verfahren nach Anspruch 25, bei dem das Programmodul in den Speicher der intelligenten Vorrichtung geladen wird, bevor der Programminterpretierer aufgerufen wird.

27. Verfahren nach Anspruch 26, wobei das Laden das Aufrufen eines Befehlsausführers beinhaltet, der auch Teil des Betriebssystems der intelligenten Vorrichtung ist, um einen Ladebefehl von der Schnittstellenvorrichtung zu empfangen und auszuführen, und wobei das Aufrufen des Programminterpretierers durch den Befehlsausführer nach dem Empfangen und Ausführen eines Programmodul-Ausführungsbefehls von der Schnittstellenvorrichtung ausgeführt wird.

28. Verfahren nach einem der Ansprüche 25 bis 27, wobei die Datendateien Kopfteile aufweisen, die verschiedene Zugriffsbeschränkungsebenen bereitstellen, die auf verschiedene Programmodule anwendbar sind, und wobei die Instruktionen den Kopfteil der vorgeschriebenen Datendatei decodieren, bevor auf sie zugegriffen wird, um zu bestimmen, ob auf sie zugegriffen werden kann.

Es folgen 9 Blatt Zeichnungen

Anhängende Zeichnungen

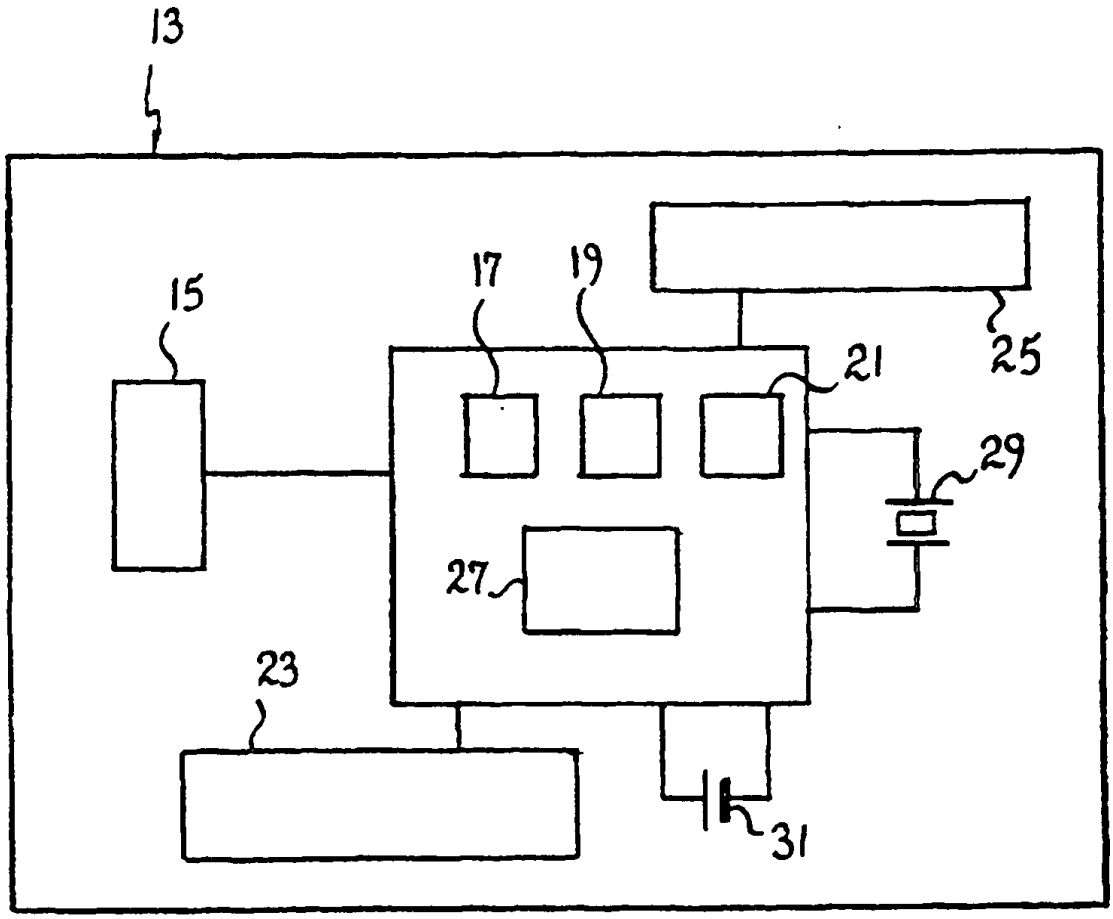

Fig. 1.

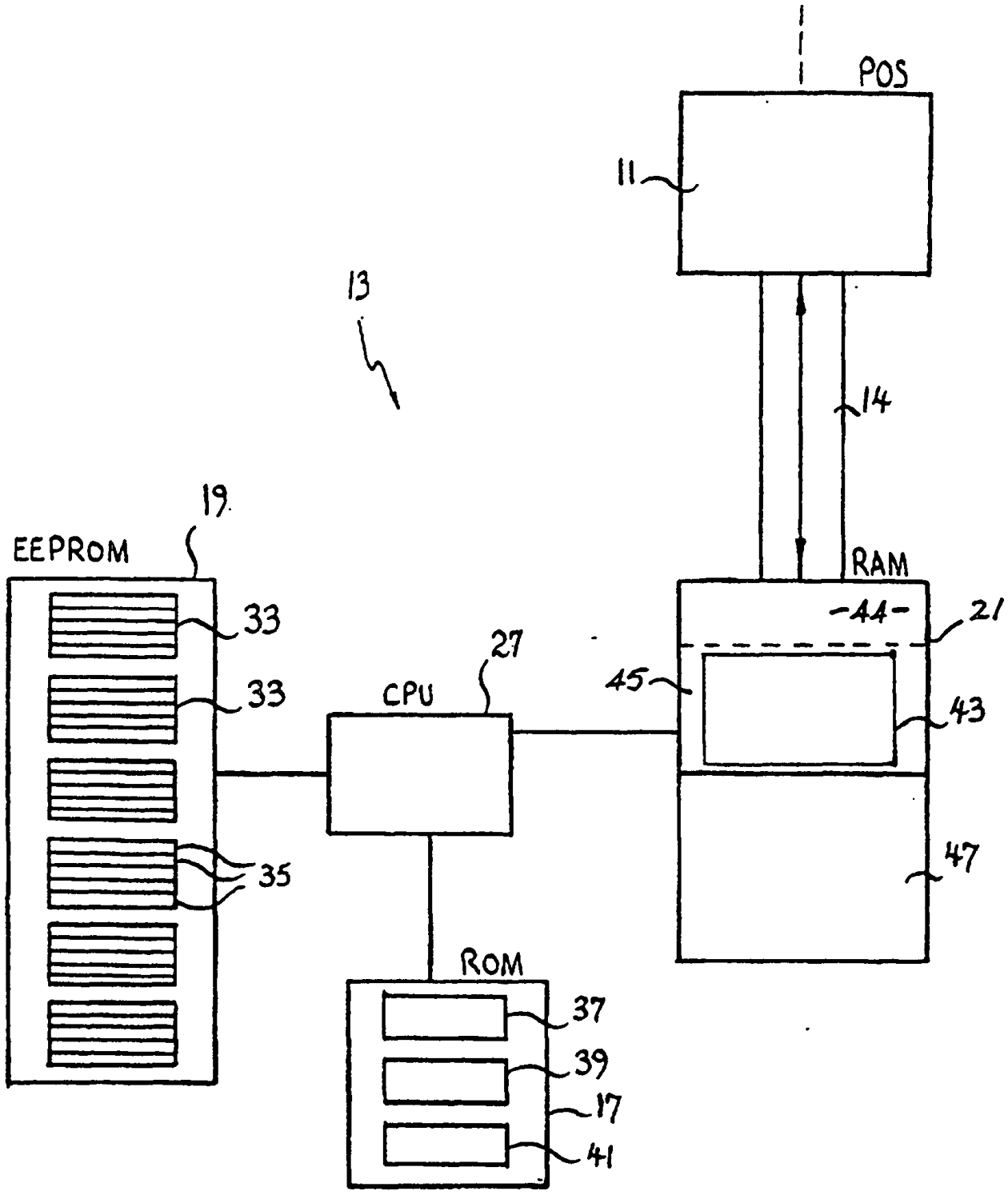
POS
11
13
14
19
EEPROM
33
33
35
RAM
44
21
27
CPU
45
43
47
ROM
37
39
17
41

Fig. 2

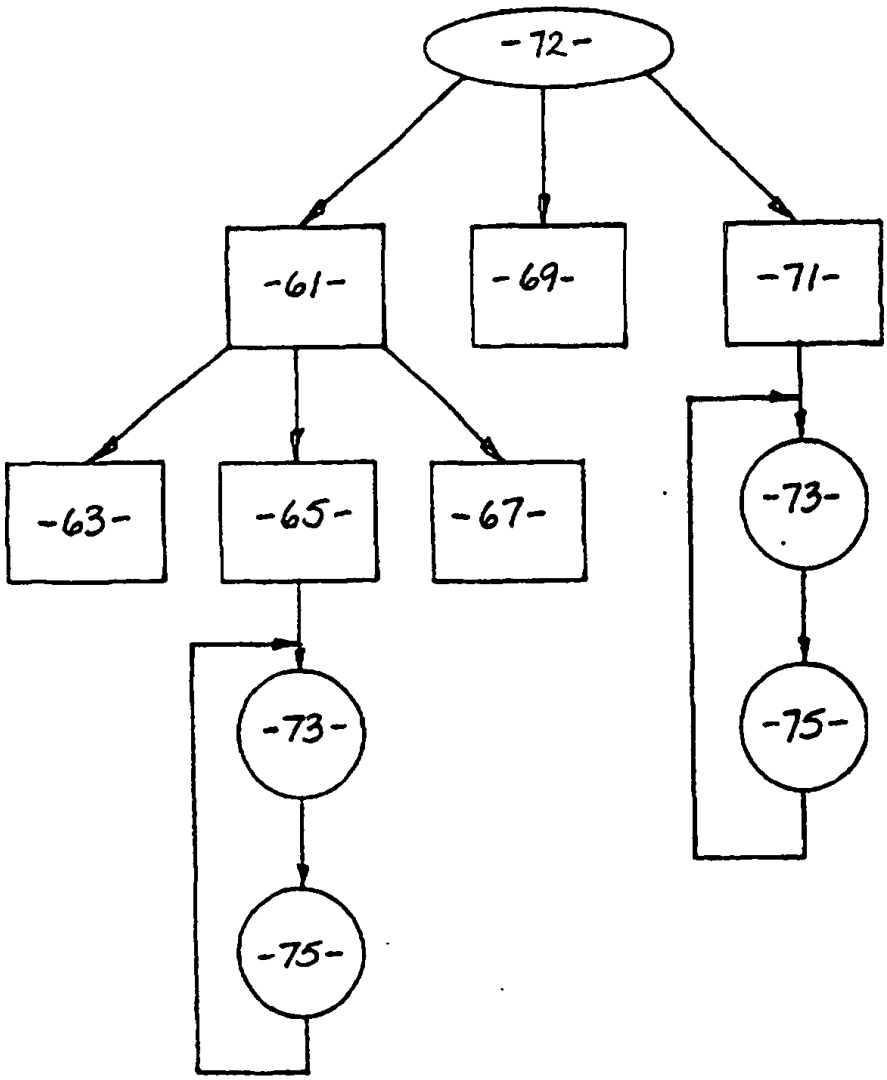

Fig 3

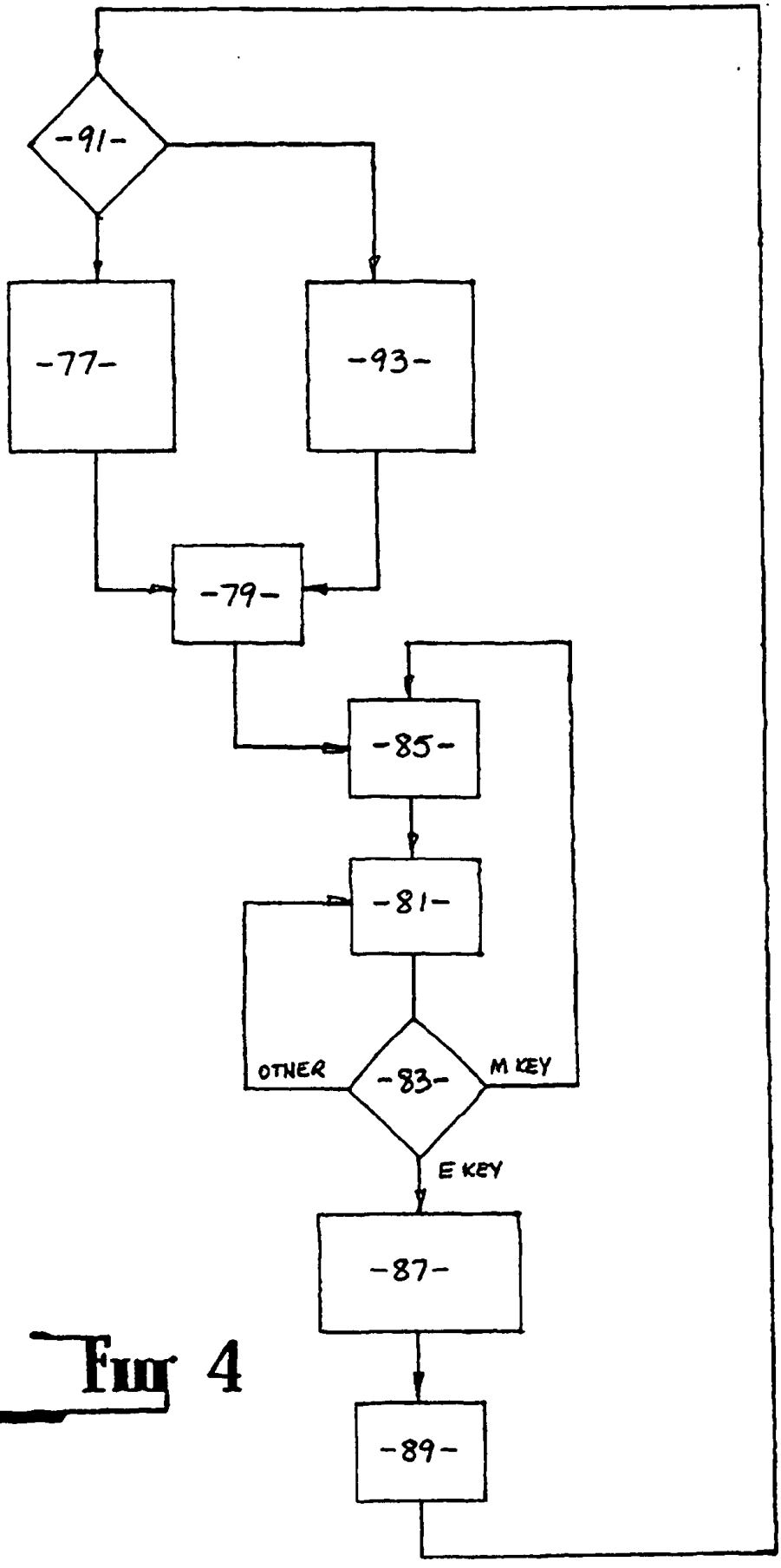

Fig 4

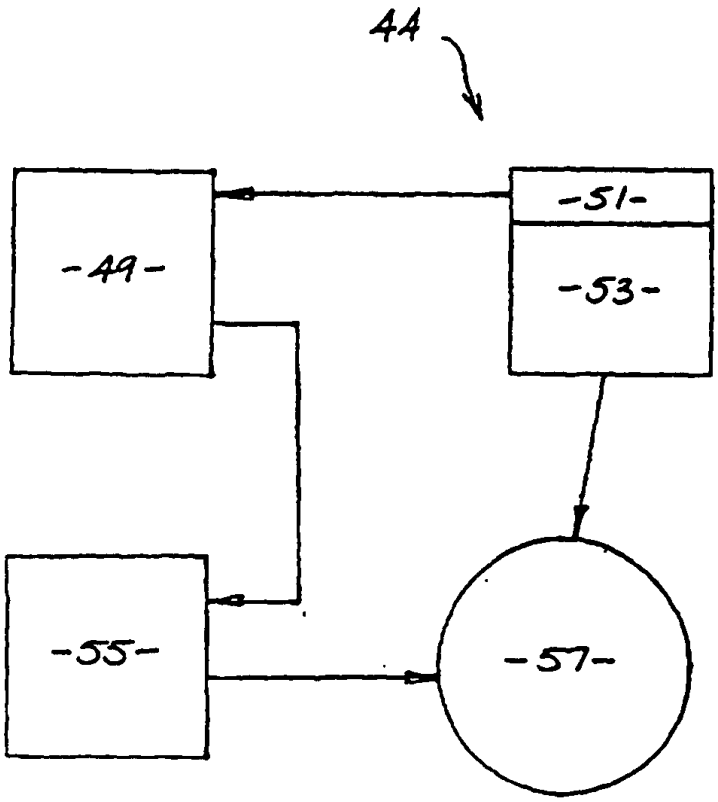

Fig 5

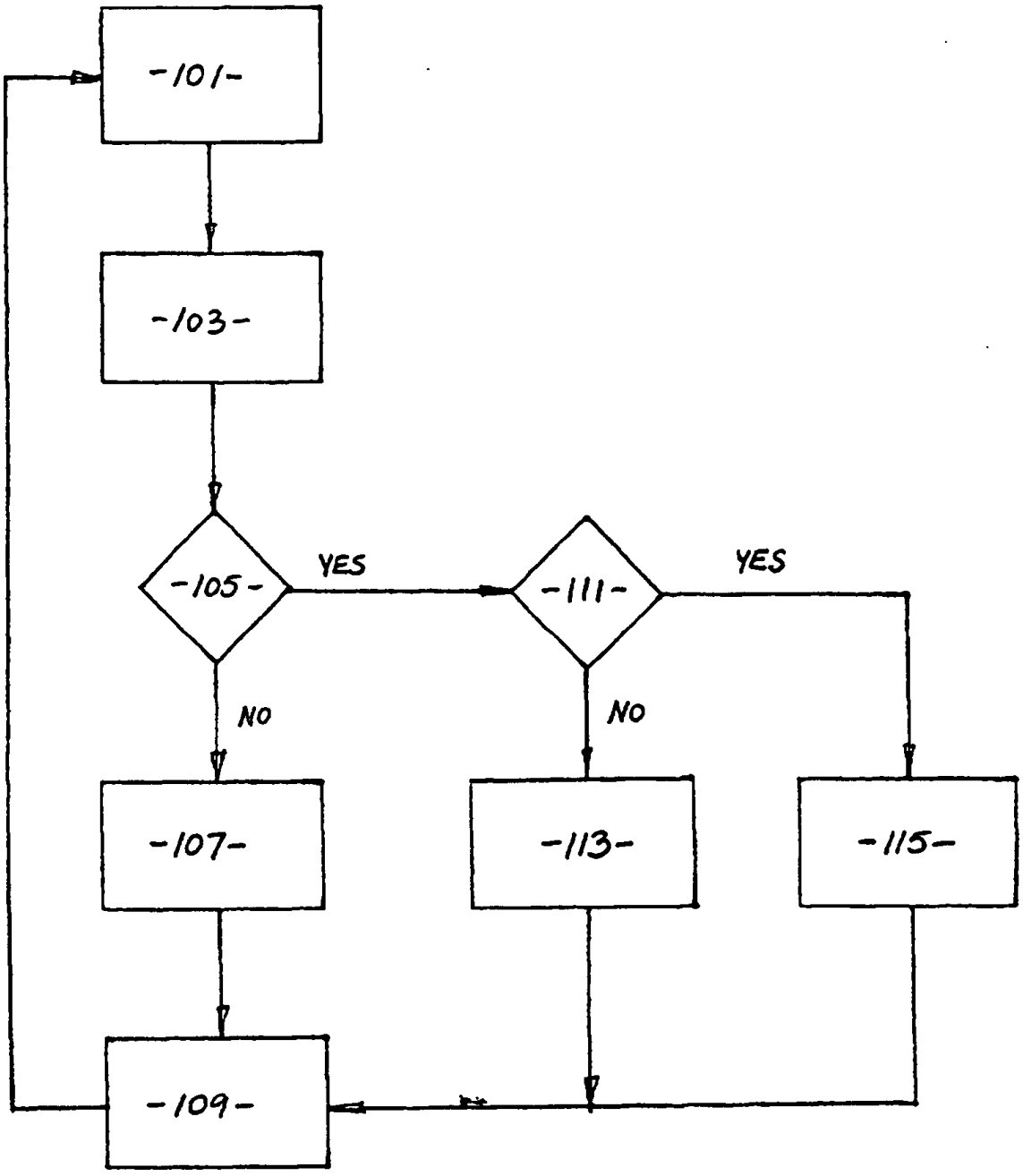

Fig 6

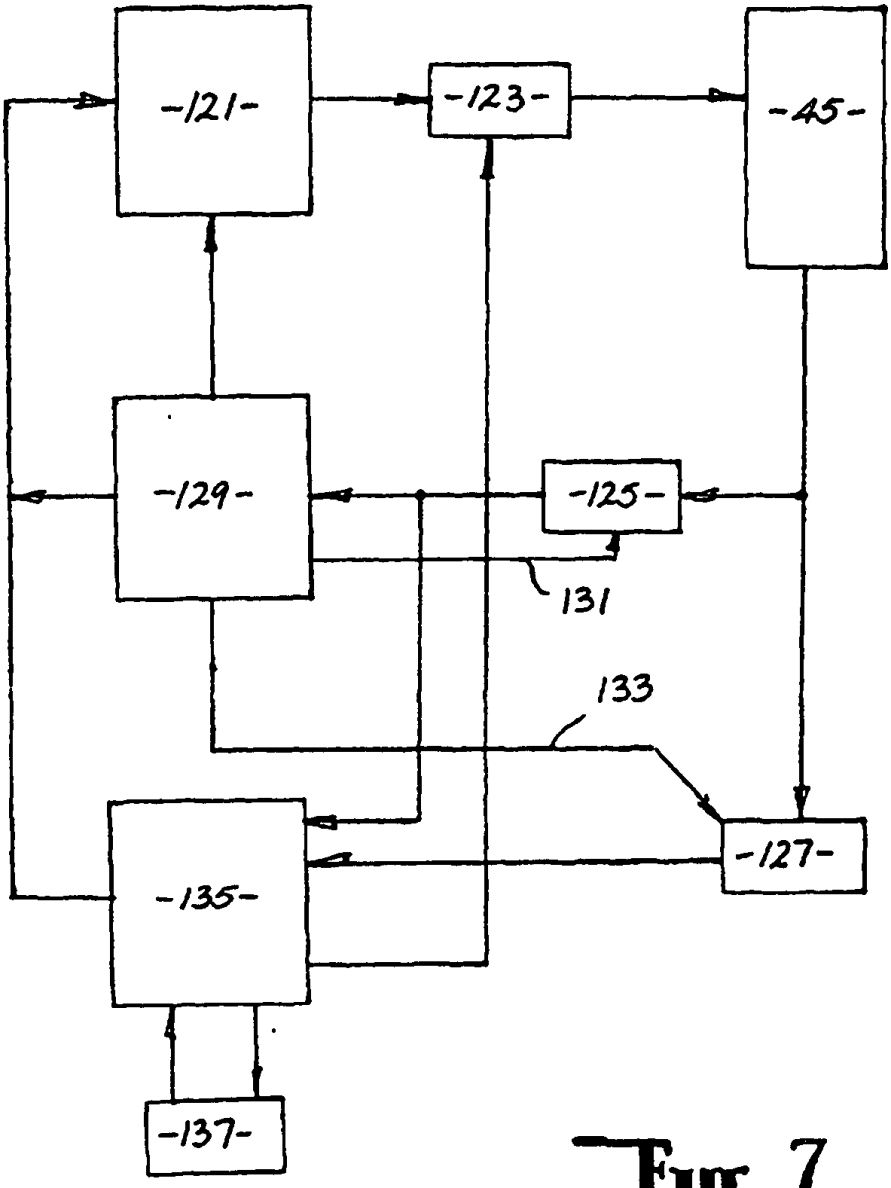

Fig 7

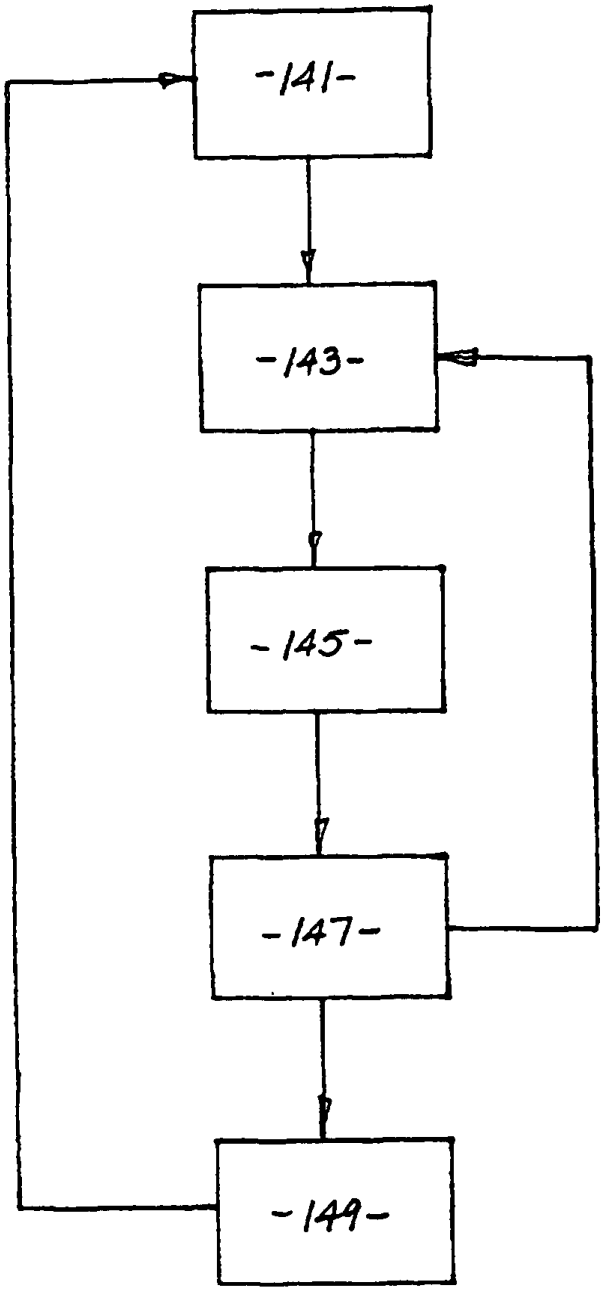

Fig 8

Fig. 9