(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 1 278 164 B1**

(12) **EUROPÄISCHE PATENTSCHRIFT**

(45) Veröffentlichungstag und Bekanntmachung des Hinweises auf die Patenterteilung:
16.01.2013 Patentblatt 2013/03

(51) Int Cl.:
G07B 17/04 *(2006.01)* ***G06F 9/445*** *(2006.01)*
G06F 1/00 *(2006.01)*

(21) Anmeldenummer: **02090220.1**

(22) Anmeldetag: **22.06.2002**

(54) **Anordnung und Verfahren zum Andern der Funktionalität eines Sicherheitsmoduls**

System and method for changing the functionality of a security module

Système et méthode pour changer la fonctionnalité d'un module de sécurité

(84) Benannte Vertragsstaaten:
CH DE FR GB IT LI

(30) Priorität: **16.07.2001 DE 10137505**

(43) Veröffentlichungstag der Anmeldung:
22.01.2003 Patentblatt 2003/04

(73) Patentinhaber: **Francotyp-Postalia GmbH**
16547 Birkenwerder (DE)

(72) Erfinder:
- **Baum, Volker**
  13189 Berlin (DE)
- **Rosenau, Dirk**
  13469 Berlin (DE)

(56) Entgegenhaltungen:
EP-A- 1 087 294
WO-A-98/20461
US-A- 4 849 927
US-A- 5 844 986

EP 1 278 164 B1

Printed by Jouve, 75001 PARIS (FR)

Beschreibung

[0001] Die Erfindung betrifft eine Anordnung zum Ändern der Funktionalität eines Sicherheitsmoduls, gemäß des Anspruchs 1 angegebenen Art und ein entsprechendes Verfahren, gemäß des Anspruchs 3 angegebenen Art. Sicherheitsmodule arbeitet in einer potentiell unfreundlichen Umgebung in Geldautomaten, Fahrkartenautomaten, Registrierkassen, elektronischen Geldbörsen, Computern für den persönlichen Gebrauch (Palmtops, Notebooks, Organizers), Handys und Geräten, die mehrere dieser Funktionalitäten kombinieren. Die Baugruppen sind mit einer Vergussmasse vergossen. In Form eines postalischen Sicherheitsmoduls ist ein Einsatz in einer Frankiermaschine bzw. Postbearbeitungsmaschine oder Computer mit Postbearbeitungs-funktion (PC-Frankierer) möglich.

[0002] Es ist bereits aus EP 417 447 B1 bekannt, in elektronischen Datenverarbeitungsanlagen besondere Module einzusetzen und mit Mitteln zum Schutz vor einem Einbruch in ihre Elektronik auszustatten. Solche Module zählen zu den Sicherheitsmodulen.

[0003] Moderne Frankiermaschinen, oder andere Einrichtungen zum Frankieren von Postgut, sind mit einem Drucker zum Drucken des Postwertstempels auf das Postgut, mit einer Steuerung zum Steuern des Druckens und der peripheren Komponenten der Frankiermaschine, mit einer Abrecheneinheit zum Abrechnen von Postgebühren, die in nichtflüchtigen Speichern gehalten werden, und einer Einheit zum kryptografischen Absichern der Postgebührendaten ausgestattet. Ein Sicherheitsmodul (EP 789 333 A2) kann eine Hardware-Abrecheneinheit und/oder die Einheit zum Absichern des Druckens der Postgebührendaten aufweisen. Beispielsweise kann ersterer als Anwenderschaltkreis ASIC und letzterer als OTP-Prozessor (One Time Programmable) realisiert werden. Ein interner OTP-Speicher speichert auslesesicher sensible Daten (kryptografische Schlüssel), die beispielsweise zum Nachladen eines Guthabens erforderlich sind. Eine Kapselung durch ein Sicherheitsgehäuse bietet einen weiteren Schutz.

[0004] Zum Schutz eines Sicherheitsmoduls vor einem Angriff, auf die in ihm gespeicherten Daten, wurden weitere Maßnahmen vorgeschlagen, in der DE 198 16 572 A1 mit dem Titel: Anordnung für ein Sicherheitsmodul und DE 198 16 571 A1 mit dem Titel: Anordnung für den Zugriffsschutz für Sicherheitsmodule, im EP 1 035 516 A2 mit dem Titel: Anordnung für ein Sicherheitsmodul, im EP 1 035 517 A2 und EP 1 035 518 A2 beide mit dem Titel: Verfahren zum Schutz eines Sicherheitsmoduls und Anordnung zur Durchführung des Verfahrens, im EP 1 035 513 A2 mit dem Titel: Sicherheitsmodul mit Statussignalisierung, sowie im deutschen Gebrauchsmuster DE 200 20 635 U1, mit dem Titel: Anordnung zur Stromversorgung für einen Sicherheitsbereich eines Gerätes.

[0005] Die bisher zum Einsatz kommenden verschiedene Techniken, wie die Umhüllung mit einem sicheren Gehäuse und der Einsatz von verschiedenen Ereignis-Detektoren, die ggf. das Sicherheitsmodul zum Löschen von sicherheitsrelevanten Daten veranlassen (EP 1 035 518 A2 und DE 200 20 635 U1) können nur für die eine vorgesehene Anwendungsfunktionalität einen sichern Schutz vor Manipulation bieten.

[0006] Aus dem US 4.528.644 ist ein Verfahren zum kundenspezifischen Einstellen der Firmware einer elektronischen Frankiermaschine nach deren Zusammenbau bekannt, wobei eine Eingabe einer Konfigurationsmitteilung in einem nichtflüchtigen Speicher gespeichert wird und mit dem Betriebsprogramm zusammenwirkt, um die Frankiermaschine den Kundenwünschen anzupassen. Nach Abschluß der Konfigurierung wird der weitere Zugriff zum Konfigurationsprogramm verhindert. Außerhalb der sicheren Umgebung beim Hersteller ist es jedoch schwierig, einen sicheren Schutz vor Manipulation zu bieten. Deshalb werden außerhalb der sicheren Umgebung beim Hersteller keine sicherheitsrelevanten Programmdaten in Frankiermaschinen installiert, um eine unterschiedliche Anwendungsfunktionalität zu erreichen.

[0007] In modernen Postgeräten werden heute bereits als Programmspeicher sogenannte Flash-EEPROM's eingesetzt. Letztere gestatten ein sektorweises Löschen und Speichern von Daten sowie ein byteweises Einfügen von einzelnen Daten in einen Speicherbereich (Sektor). So wurde bereits im EP 724 141 B1 ein Verfahren zur Dateneingabe in eine Waage vorgeschlagen, wobei die betreffenden Speicherbereiche im Flash-EEPROM der Waage gelöscht werden, bevor eine Re-programmierung vorgenommen wird, beispielsweise um eine Portotariftabelle mindestens partiziell zuändern. Die vorzugsweise via Modem einer Frankiermaschine, beispielsweise JetMail®, geladenen Daten werden im Flash-EEPROM komprimiert gespeichert und vor Anwendung dekomprimiert und in einem separaten Anwendungsspeicher gespeichert. In der Waage ist außerdem ein programmierbares Sicherheitsmittel vorgesehen, dass ein unbefugtes Löschen von Datenblöcken in den Flash-EEPROM-Speicherbereichen verhindert. Für die Frankiermaschine sind Teilbilddateien und eine Steuerdatei bestimmt, welche zugleich mit den für die Waage bestimmten Daten von einer Datenzentrale in den Speicher der Frankiermaschine heruntergeladen werden. Neben einem Datensatz, der u.a. auch eine Versionsinformation enthält, wird auch der Bearbeitungsstatus gespeichert, um den erreichten Programmzustand bei Programmabbruch nichtflüchtig zu konservieren. Jedoch weder in der Frankiermaschine noch in der Waage werden scherheitsrelevante Programmdaten installiert.

[0008] Ein elektronisches Gerät mit Flash-Speicher und ein Verfahren zur Reprogrammierung des Flash-Programmspeichers wurde in der EP 788 115 A2 vorgeschlagen. Die Programmierung des Flash-Programmspeicher-Bausteins erfolgt durch die Abarbeitung eines für diesen Zweck in einer Speicher-Bank dieses Bausteins enthaltenen Teil-Programms, wobei die betreffen-

den Speicherbereiche der jeweils anderen Speicher-Bank gelöscht werden, bevor eine Reprogrammierung vorgenommen wird. Meist ist das Programm länger oder kürzer als der durch Löschung geschaffene freie Speichersektor, der somit nicht voll ausgenutzt werden kann. Neben der o.g.Einschränkung bezüglich der vollständigen Nutzung des Speicherplatzes ist ein solcher Baustein auch noch teuerer als ein vergleichbarer ohne mehrere Speicher-Bänke. Durch ein Überprüfen einer Checksumme wird festgestellt, ob die Reprogrammierung komplett abgeschlossen ist. Jedoch kann damit nicht ausgeschlossen werden, dass das Gerät mit manipulierten Daten reprogrammiert wurde.

[0009] Auch bei den postalischen Sicherheitsmodulen kommen für die funktionsspezifische Programmspeicherung reprogrammierbare Speicherbausteine (FLASH oder EEPROM) zum Einsatz. Die Programmierung dieser Bausteine kann in bekannter Art nach unterschiedliche Verfahren beim Hersteller vorgenommen werden:

- Programmierung eines Programm-Bausteins durch einen Programmieradapter vor dem Einbau in das Sicherheitsmodul,
- Programmierung des Programm-Bausteins durch die Abarbeitung eines für diesen Zweck in einer Speicher-Bank dieses Programm-Bausteins enthaltenen Teil-Programms.

[0010] Das erste Verfahren hat gegenüber dem zweiten Verfahren den Nachteil, dass ein fehlerhaftes Programm nicht mehr ausgetauscht werden kann. Das zweite Verfahren erfordert in nachteiliger Weise einen Baustein, der mindestens zwei unterschiedliche Speicher-Bänke besitzen, was ihn bei den o.g. starken Einschränkungen der Nutzung des Speicherplatzes teuerer macht. An die postalischen Sicherheitsmodule werden besondere Anforderungen gestellt, was den Austausch bzw. die Erweiterbarkeit von Funktionen angeht. Die Programmierung von o.a. Programmbausteinen darf nicht zu jedem Zeitpunkt und insbesondere nicht von jedem Bediener ausgeführt werden können.

[0011] Aus der Veröffentlichung EP 1087294 A2 ist bereits Kommunikationsgerät mit einem Flash-Speicher bekannt, der zwei Partitionen zum Speichern eines Programms aufweist und Firmware nachladen eingesetzt wird.

[0012] Es ist auch ein FLASH Typ bekannt, der nur eine Partition aufweist, wie der Spansion™-16Mbit-MBM29LV160TE, welcher von der Firma Advanced Micro Devices (AMD) oder von der Firma Fujitsu angeboten wird, der reprogrammierbar ist und einen freien Speicherplatz aufweist.

[0013] Der Erfindung liegt die Aufgabe zugrunde, mit geringem Aufwand die oben genannten besondere Anforderungen, unter Vermeidung der Nachteile, zu erfüllen und eine Anordnung und Verfahren für ein Sicherheitsmodul zu schaffen, dass ein Austauschen der Funktionalität zustandsabhängig und autorisiert gewährleistet.

[0014] Die Aufgabe wird mit den Merkmalen der Anordnung nach Anspruch 1 bzw. mit den Merkmalen des Verfahrens nach Anspruch 3 gelöst.

[0015] Bei dem entwickelten Sicherheitsmodul kommt ein Mikroprozessor zum Einsatz, der die Ausführung seines Programms in einem Arbeitsspeicher ermöglicht. Neben diesem Arbeitsspeicher kommt ebenfalls ein FLASH-Programm-Speicher für das anwendungsspezifische Programm zum Einsatz. Beide Speicher sind über den Bus an den Prozessor angeschlossen.

[0016] Zum Zeitpunkt der Fertigung des Sicherheitsmoduls wird nach der oben genannten ersten bekannten Methode ein sog. "Bootloader" als Startladeprogramm in den Programmspeicher eingebracht. Eine spezielle Funktionalität zur Änderung der Funktionalität des übrigen freien Programm-Speichers ermöglicht:

a) Das Kopieren eines Programmteils des Startladeprogramms in den Arbeitsspeicher,
b) Das Ausführen dieses Programmteils im Arbeitsspeicher, zum Programmieren des freien Teils des Programmspeichers,
c) Das Verifizieren eines beim Programmieren erreichten Programmzustandes, um die Programmfunktionalität zustandsabhängig ausführen zu können,
d) Das Autorisieren der geänderten Funktionsweise des nachgeladenen Programms bei dessen Authentizität.

[0017] Das Sicherheitsmodul wird somit während seiner Produktion mit Programmdaten programmiert und erhält eine Kennung für einen ersten Grundzustand. Nach dem Einschalten wird mittels eines Start up Programmes ein erster Programmteil aus dem Speicherbereich des Programmspeichers in den Arbeitsspeicher kopiert. Der erreichte Programmzustand wird verifiziert, um die Programmfunktionalität zustandsabhängig ausführen zu können. Eine Zustandsvariable für den erreichten Programmzustand kann beispielsweise im Programmspeicher oder in einem nichtflüchtigen Speicher des Sicherheitsmoduls gespeichert sein. Eine Lichtemitterdiode (LED) signalisiert, dass der Mikroprozessor einen zweiten Programmteil abarbeitet und auf die Änderung der Programmfunktionalität des freien Programmspeichers wartet. Über das im Sicherheitsmodul enthaltene Kommunikationsinterface werden mindestens Anwendungsprogrammdaten in einen freien bzw. nicht aktiven Speicherbereich des Programmspeichers geladen. Außerdem werden zugehörige Kennungsdaten und eine kryptographische Signatur des Anwendungsprogramms in einen nichtflüchtigen Speicher des Sicherheitsmoduls oder ebenfalls in ein und denselben oder anderen freien bzw. nicht aktiven Speicherbereich desselben Programmspeichers geladen. Hierzu verifiziert der durch das zweite Programmteil gesteuerte Mikroprozessor zunächst die Kennung des vorherig gespeicherten Programms. Die Kennung beschreibt die Eigenschaften der

Programmdaten und ist auf einem Speicherplatz mit einer bestimmten Adresse gespeichert. Stellt die an dieser Adresse gespeicherte Kennung einen gültigen Vorgänger der Kennung der neuen Anwendungsprogrammdaten dar, so wird die in den Arbeitsspeicher kopierte Funktionalität des ersten Programmteils benutzt, um die über das Kommunikationsinterface erhaltenen Anwendungsprogrammdaten in den freien Speicherbereich des Programmspeichers zu laden. Vor jeder Programmierung des Programmspeichers wird zusätzlich sichergestellt, dass keine Daten in den z.Z. aktiven Bootloader-Speicher-Bereich gelangen können, um ein Überschreiben des Startladeprogramms (Bootloaders) zu verhindern. Nachdem auf diese Weise alle Anwendungsprogrammdaten in den freien Speicherbereich des Programmspeichers gespeichert wurden, wird bei Echtheit des Anwendungsprogramms dessen Anwendung freigegeben. Beispielsweise wird ein Zertifikatcode, vorzugsweise die kryptographische Signatur der geladenen Anwendungsprogrammdaten verifiziert und bei Erfolg wird das geladene Anwendungsprogramm durch ein Flag als gültig gekennzeichnet bzw. der erreichte Zustand des Anwendungsprogramms auf andere geeignete Weise fortgeschrieben. Außerdem wird die zugehörige Kennung gespeichert. Die Änderung der Funktionalität ist somit abgeschlossen. Nachdem das Sicherheitsmodul neu gebootet wurde, stellt nun das Startladeprogramm (Bootloader) fest, das der neue Programmzustand eine gültige Anwendungsprogrammfunktionalität anzeigt und führt diese nun aus. Dies wird zusätzlich durch eine andersfarbige LED angezeigt. Eine Änderung der jetzigen Funktionalität des Programmspeichers ist nun nicht mehr möglich solange der Programmzustand nicht wieder geeignet modifiziert wird.

[0018] Um diese Änderung der Programmfunktionalität weiterhin gewährleisten zu können, enthält jede nachgeladene Funktionalität ebenfalls ein Teilprogramm zum Kopieren und Ausführen von Programmieranweisungen in den Arbeitsspeicher. Diese Funktionalität kann ebenfalls über das im Sicherheitsmodul befindliche Kommunikationsinterface aufgerufen werden. Bei Aufruf ändert es die Zustandvariable derart, dass zwar die Kennung des Programms erhalten bleibt, beim nächsten Booten dem Bootloader jedoch kenntlich gemacht wird, dass die anwendungsspezifische Software jetzt wieder einen freien Programmspeicherbereich darstellt. Als Folge wird beim nächsten Booten der Bootloader wieder aktiv sein und Anwendungsprogrammdaten empfangen.

[0019] Die Erfindung geht weiterhin davon aus, dass mittels einem schnellen Mikroprozessor und weiteren teilweise bekannten Funktionseinheiten ein Sicherheitsmodul geschaffen wird, das allen Anforderungen genügt. Der schnelle Prozessor ermöglicht symmetrische und/oder asymmetrische Verschlüsselungsverfahren für unterschiedliche Einsatzfälle einzusetzen. Entsprechend dem jeweiligem Einsatzfall wird eine Echtzeitverarbeitung von Ereignissen sowie eine Aufzeichnung bzw. Buchung ermöglicht. Eine interne Batterie des Sicherheitsmoduls übernimmt die Spannungsversorgung für eine Echtzeituhr und für Bauelemente zur nichtflüchtigen Speicherung der Nutzdaten, zur permanenten Überwachung aller sicherheitsrelevanten Funktionen sowie der Betriebsbereitschaft des Sicherheitsmoduls bei ausgeschalteter Systemspannung des Gerätes. Im Fehlerfall und bei Entfernung des Sicherheitsmoduls wird eine Zustandsänderung abfragbar gespeichert. Der Status des Sicherheitsmoduls ist auch nach dem Löschen vom Gerät abfragbar. Zur Signalisierung des Zustandes kann eine vorhandene Anzeigeeinheit des Gerätes oder ein Signalisierungsmittel des Sicherheitsmoduls mitbenutzt werden.

[0020] Vorteilhafte Weiterbildungen der Erfindung sind in den Unteransprüchen gekennzeichnet bzw. werden nachstehend zusammen mit der Beschreibung der bevorzugten Ausführung der Erfindung anhand der Figuren näher dargestellt. Es zeigen:

Figur 1, Blockschaltbild des Sicherheitsmoduls,

Figur 2, Darstellung der Mehrschicht-Programm Architektur,

Figur 3, Flußplan zur Änderung der Funktionalität des Sicherheitsmoduls.

[0021] Die Figur 1 zeigt ein Blockschaltbild des Sicherheitsmoduls, umfassend die Baugruppen:

- einen Mikroprozessor 120 mit interner Echtzeituhr,
- einen Programmspeicher 128, zum Beispiel ein FLASH 512K x32,
- einen Arbeitsspeicher SRAM 121, zum Beispiel ein SRAM 64K x32,
- zwei nichtflüchtige Speicher NVRAM I & NVRAM II,
- einen Arbeitsspeicher SRDI-RAM 122 (Secure Relevant Data Items) mit Lösch-Hardware und BUS-Treibereinheit 127,
- eine Langzeit-Batterie 134, zum Beispiel eine Lithium-Batterie,
- eine Leistungsverwaltungs-& Überwachungseinheit (Power Manager) 11 mit Spannungsüberwachungseinheit 12, mit Schnittstellen zur Zuführung der Systemspannung (Main Power Supply Interface) und zur Batteriespannungszuführung (Host Battery Interface),
- Ereignisdetektoren(Event Detectors), einschließlich einer Zerstörungs-Detektionseinheit 15, die mit einer in einer Vergussmasse 105 eingebetten Membrane 153 verbunden ist, und einer Ungestecktsein-Detektionseinheit 13,
- einen speziellen Schaltkreis FPGA 160 mit einem I/O Interface 150 zur Herstellung einer Kommunikationsverbindung mit einem Gerät. Das Kommunikationsinterface 150 enthält eine interne Steuerung und einen 8 Byte-Kommunikations-Puffer, aus welchem zuerst eingelesene Daten zuerst ausgelesen

und weitergeleitet werden.

[0022] Das Herstellergerät liefert eine Systemspannung und optional eine zweite Batteriespannung. Das Sicherheitsmodul wird bei eingeschaltetem Herstellergerät mit Systemspannung betrieben. Zum Ändern der Funktionalität ist das Sicherheitsmodul 100 mit einem reprogrammierbaren FLash-Programmspeicher 128 ausgestattet, der ein Startladeprogramm speichert, und mit einem Mikroprozessor 120, der das Startladeprogramm teilweise in den Arbeitsspeicher SRAM 121 kopiert. Das integrierte Kommunikationsinterface 150 des speziellen Schaltkreises 160 ermöglicht die Herstellung einer Kommunikationsverbindung mit dem Herstellergerät, welches Anwendungsprogrammdaten für das Sicherheitsmodul bereit stellt. Der Mikroprozessor 120 steht über einen BUS mit dem Arbeitsspeicher SRAM 121, mit dem FLASH-Programmspeicher 128 und mit dem Kommunikationsinterface 150 in kommunikativer Verbindung. Es ist vorgesehen, dass das Kommunikationsinterface 150 zur Bereitstellung von Daten mindestens eines Teils eines Anwendungsprogramms, eines zugehörigen Zertifikatcodes und Kennungsdaten ausgebildet ist und dass der Mikroprozessor 120 durch das teilweise in den Arbeitsspeicher 121 kopierte Startladeprogramm programmiert ist, die Daten des Teils des Anwendungsprogramms auf einem freien Speicherplatz des FLASH-Programmspeichers zu speichern, wenn die Kennungsdaten einen Nachfolger zur gespeicherten Vorgängerkennung kennzeichnen, und die Authentizität des geladenen mindestens einen Teils des Anwendungsprogramms mittels des Zertifikatcodes zu überprüfen und bei Authentizität des geladenen Teils des Anwendungsprogramms letzteres als gültig zu speichern.

[0023] Der Mikroprozessor 120 stellt fest, ob die Kennungsdaten einen Nachfolger zur gespeicherten Vorgängerkennung kennzeichnen, indem die Kennungsdaten mit entsprechenden Vergleichsdaten verglichen werden, die in einem weiteren Speicherbereich des FLASH-Programmspeichers 128 gespeichert sind, in welchem Informationsdaten zu einem bereits geladenen Programm gelistet sind. Zu den Kennungsdaten gehören der Programmtyp, die Versions- und die Revisionsdaten. Vorteilhaft ist ein Mikroprozessor-Typ vorgesehen, der die Ausführung seines Programms in einem Arbeitsspeicher 121 ermöglicht, um den FLASH zu reprogrammieren. Damit kann auf den Einsatz eines teueren FLASH-Programmspeicher-Bausteins mit separaten Speicherbänken verzichtet werden.

[0024] Die Leistungsverwaltungseinheit (Power Manager) 11 hat eine Vielzahl an Funktionseinheiten, die bei einem geringen Leistungsverbrauch die Betriebsfähigkeit des Sicherheitsmoduls auch bei abgeschaltetem Gerät sichern. Die Leistungsverwaltungseinheit 11 weist einen Gleichstrom/Gleichstrom-Wandler (nicht gezeigt) und einen Spannungsregler (nicht gezeigt) für die entsprechenden Betriebsspannungen (3V, 5V und 8V), eine Temperatur-und Spannungsüberwachungsschaltung (nicht gezeigt) auf. Die letzteren beiden können ein Reset-Signal erzeugen. Die gelieferte Systemspannung wird auf Über- bzw. Unterschreitung von Grenzwerten überwacht. Innerhalb letzterer sorgt ein Gleichstrom/Gleichstrom-Wandler für eine vorbestimmte Betriebsspannung U_B. Eine Spannungsgenerierung sorgt für die Erzeugung aller notwendigen Spannungen, die die Funktionseinheiten des Sicherheitsmoduls benötigen.

[0025] Bei ausgeschalteten Gerät werden neben den Überwachungsschaltungen und der Zerstörungs-Detektionseinheit nur eine Echtzeituhr RTC und die Arbeitsspeicher mit Batteriespannung versorgt. Eine ununterbrochene Versorgung der batteriebetriebenen Einheiten ist auch in DE 200 20 635 U1 mitgeteilt worden. Zu letzteren gehört mindestens einer der Post-Speicher, einige der Detektoren und der SRDI-Speicher. An das Sicherheitsmodul können zwei unabhängige Batterien angeschlossen werden. Die erste Batteriespannung stammt aus der internen Batterie 134, welch optional durch eine zweite separate Batterie gestützt werden kann.

[0026] Alternativ zur internen Echtzeituhr kann eine separate Echtzeituhr RTC 124 angeschlossen werden. Der Mikroprozessor 120 ist beispielsweise vom Typ ARM7 und die separate Echtzeituhr vom Typ EPSON RTC-4543. Der Mikroprozessor 120 ist über einen BUS mit dem Programmspeicher FLASH 128, dem Arbeitsspeicher SRAM 121, dem Arbeitsspeicher SRDI-RAM 122 und dem speziellen Schaltkreis FPGA 160 verbunden. Der Bus ist mit breiten weißen Pfeilen dargestellt. Der spezielle Schaltkreis FPGA 160 ist ein anwenderspezifisch programmiertes FPGA (one time programmable). Der FPGA enthält eine Hardware-Abrechnungseinheit (nicht gezeigt), eine Ansteuerschaltung für zwei weitere Speicher NVRAM I und II sowie eine Ein/Ausgabe-Schnittstelle (digitale Interface des Sicherheitsmoduls nicht gezeigt) zum Gerät (nicht gezeigt). Der spezielle Schaltkreis FPGA 160 ist mit zwei nichtflüchtigen Speichern 114 (NVRAM I) & 116 (NVRAM II) verbunden, die unter anderem die postalisch relevanten Daten enthalten. Die beiden nichtflüchtigen Speicher NVRAM I und II sind physikalisch getrennt und in verschiedenen Technologien ausgeführt. Sie sind vom Prozessor schreibend und lesend ansprechbar, vom FPGA modifizierbar und von außerhalb des Sicherheitsmodules lesbar. Einer der nichtflüchtigen Speicher ist in einer gemischten EEPROM-SRAM-Technologie ausgeführt, der andere ist ein SRAM mit herkömmlicher Technologie.

[0027] Die Zuführung der Systemspannung (Main Power Supply Interface) und der Batteriespannungen zur Schnittstelle ist mit breiten schwarzen Pfeilen gekennzeichnet worden. Dünne schwarze Pfeile gekennzeichnen die Versorgung von Baugruppen mit einer entsprechenden Betriebsspannung aus der Leistungsverwaltungs-und Überwachungseinheit 11 bzw. aus der Überwachungseinheit 12. Dünne weiße Pfeile kennzeichnen Abfrage- und Steuerleitungen.

[0028] Zur Lösch-Hardware gehören teilweise Mittel der Leistungsverwaltungs-& Überwachungseinheit, eine

Steuerleitung CL und eine Bus-Treibereinheit 127. Die Steuerleitungen von der Zerstörungs-Detektionseinheit 15 und der Spannungsüberwachungseinheit 12 sind zu einer gemeinsamen Steuerleitung CL verschaltet, welche gestrichelt dargestellt ist. Die Einheiten 12 oder 15 steuern über die gemeinsame Steuerleitung CL einen elektronischen Umschalter S an, welcher wahlweise Betriebsspannung U_B oder Löschspannung U_C bzw. Massepotential U_M an den VCC-Pin des SRDI-Arbeitsspeicher 122 anlegt. Dieser SRDI-RAM-Speicher ist nicht direkt an dem Prozessorbus angeschlossen. Alle digitalen Signale werden über Treiberschaltkreise der Bus-Treibereinheit 127 geführt, die über Ausgänge verfügen, die hochohmig geschaltet werden können. Damit kann der BUS vom SRDI-Arbeitsspeicher 122 entkoppelt werden. Die Bus-Treibereinheit 127 wird ebenfalls von der gemeinsamen Steuerleitung CL angesteuert.

[0029] Folgende Detektor- und Überwachungseinheiten überwachen den sachgemäßen Betrieb des Sicherheitsmoduls:

- Spannungsüberwachungseinheit 12, die zur Batteriespannungsüberwachung mit Selbsthaltung ausgebildet ist,
- Zerstörungsdetektionseinheit 15 zur Detektion gegen mechanische Zerstörung des Sicherheitsmoduls mit Selbsthaltung,
- Ungestecktsein-Detektioneinheit 13 (Host-System-Loop) mit Selbsthaltung.
- Temperatursensor und weitere
- Spannungsüberwachungseinheiten zur Überwachung aller Spannungen im System, insbesondere der Systemspannung.

[0030] Die beiden ersten führen bei Ansprechen (oder-Verknüpfung) zum Löschen der Daten im SRDI-Speicher.

[0031] Der dritte Detektor kann nur einen Zustandswechsel hervorrufen und vom Prozessor während des Betriebes bzw. beim Systemstart vom Programm des Sicherheitsmoduls abgefragt werden.

[0032] Der Temperatursensor überwacht die Betriebstemperatur des Moduls und löst einen Reset aus, wenn die Temperatur unter oder über einen vorherbestimmten Wert sinkt bzw. steigt. Auch damit wird ein unsachgemäßer Gebrauch verhindert und die Nutzerdaten gesichert. Ein Reset wird ebenfalls ausgelöst, wenn die Eingangsspannung des Moduls zu klein oder zu groß wird oder wenn die interne Betriebsspannung unter einen bestimmten Pegel sinkt. Der Zustand aller anderen Spannungen können von der Systemsoftware abgefragt werden. Das Sicherheitsmodul enthält - nicht gezeigte - LED zur Statusausgabe und wird mit einer harten, undurchsichtigen Vergußmasse 105 vergossen, in welche eine Sensor-Membrane 153 eingebettet ist. Einer der Ereignisdetektoren, die Zerstörungs-Detektioneinheit 15, ist mit Leiterschleifen der Sensor-Membrane 153 verbunden.

[0033] Die Figur 2 zeigt eine Darstellung der Mehrschicht-Programm-Architektur. In der obersten Schicht befinden sich ein Vorinitalisierungsprogramm und ein Anwendungsprogramm. Das Vorinitalisierungsprogramm wird nach der Herstellung der Hardware des Sicherheitsmoduls über eine Hersteller-Anwendungsprogramm-Lade-Schnittstelle (Manufacturing Application Programming Interface) geladen und veranlaßt die Generierung eines öffentlichen Schlüsselpaares, welches eine einzigartige Identität schafft. Letztere ermöglicht zu jeder Zeit eine Wiedererkennung des Sicherheitsmoduls. Die initiale kryptographisch einzigartige Identität kann später durch die kryptographische Identität des Kunden ersetzt werden. Das Anwendungsprogramm bestimmt die reguläre Funktionalität während des Betriebes des Sicherheitsmoduls. Sie steht über eine betiebsmäßige Anwendungsprogramm-Schnittstelle (Operational Application Programming Interface) zur Verfügung und kann beispielsweise dem PKCS#11 oder einem anderem kryptographischen Standard entsprechen.

[0034] In der mittleren Schicht befindet sich eine offene Sicherheitssockelschichtsammlung (Open Secure Sokket Layer Library), welche die darüber liegenden Schichten (Pre Initializer & Application-Software) nutzen können. Die Sammlung (OpenSSL-Library) enthält eine große Anzahl an Sätzen von kryptographischen Algorithmen (DES, tripel-DES, RSA, DAS, SHA-1, HMAC, usw.) und solche PKCS- und ASN.1-Formatierungswerkzeuge, wie beispielsweise der X.509v3 Zertifizierungs-Standard. Die OpenSSL-Library enthält weiterhin eine kleine und effiziente Sammlung von Elliptic Curve Digital Signature Algorithms (ECDSA), welche eine Auswahl von ein oder mehreren verschiedenen elliptischen Kurven - die vom NIST empfohlen werden - gestattet.

[0035] Die untere Schicht enthält ein Startladeprogramm (boot loader) mit einem integrierten Code-Überprüfungsprogramm. Das Startladeprogramm unternimmt zuerst ein Laden des Vorinitalisierungsprogramms, welches erst einmal geladen und ausgeführt, nicht durch ein anderes Vorinitalisierungsprogramm ersetzt werden kann, sondern höchstens durch ein Teil des Anwendungsprogramms. Bevor das Startladeprogramm den Zustand eines geladenen Teils des Anwendungsprogramms als gültig speichert, wird letzteres mittels Zertifikatcode überprüft. Der Zertifikatcode wird zusammen mit jedem Teil des Anwendungsprogramms bereitgestellt. Zur Überprüfung wird ein Code-Überprüfungsschlüssel benötigt, welcher während der Herstellung im Rahmen einer Vorinitialisierung geladen wird.

[0036] Aus den Daten des Anwendungsprogramms wird ein Hashwert gebildet, welcher beispielsweise mit einem Schlüssel nach dem bekannten DES-Verfahren (Data Encryption Standard) zu einem Message Authorization Code (MAC) verschlüsselt wird. Der MAC wird als Zertifikatcode dem Anwendungsprogramm beigefügt. Der Code-Überprüfungsschlüssel muß im Sicherheitsmodul jedoch auslesesicher gespeichert werden, wenn der Code-Überprüfungsschlüssel ein Schlüssel eines

symmetrischen Verschlüsselungsverfahrens (DES) ist.

[0037] Ein auslesesicheres Speichern entfällt, wenn ein öffentlicher Code-Überprüfungsschlüssel geladen wird. Vorzugsweise ist vorgesehen, dass der Code-Überprüfungsschlüssel ein öffentlicher Verifizierschlüssel ist, dass der öffentliche Verifizierschlüssel und ein zugehöriger geheimer Signierschlüssel ein Schlüsselpaar bilden sowie dass der Zertifikatcode mit Hilfe des geheimen Signierschlüssels vom Hersteller zugehörig zu den Daten mindestens eines Teils eines Anwendungsprogramms erzeugt wird. Aus den Daten des Anwendungsprogramms wird dazu ein Hashwert gebildet, welcher beispielsweise mit einem geheimen Signierschlüssel nach dem bekannten RSA-Verfahren (Rivest, Shamir und Adleman) zu einer digitalen Signatur verschlüsselt wird. Der Code-Überprüfungsschlüssel wird von von einer vertrauenswürdigen Zentrale des Herstellers generiert, gespeichert und ständig auf Echtheit überprüft, wobei der Hersteller eine weltweite Public Key Infrastrukture einsetzt. Zur verwendeten Public Key Infrastrukture existieren folgende Standards:

[1] American National Standards Institute: Public Key Infrastructure-Practices and Policy Framework; ANSI X9.79, 2000

[2] ISO/CCITT Directory Convergence Document: The Directory - Authentication Framework; CCITT Recommendation X.509 and ISO 9594-8, "Information Processing Systems - Open Systems Interconnection - the Directory-Authentication Framework".

[3] ISO_9594-8a 95 ISO/IEC 9594-8: Information technology - Open Systems Interconnection - Specification - The Directory: Authentication framework; ISO/IEC International Standard, Second edition 15.09.1995.

[4] ISO_10181-2 96 ISO/IEC 10181-2: Information technology - Open Systems Interconnection - Security frameworks for open systems: Authentication framework; ISO International Standard 10181-2, Ist edition, 96.05.15, 1996.

[5] Bruce Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C; (2nd ed.) John Wiley & Sons, New York 1996, Chapter 24.9

[6] Simson Garfinkel, Gene Spafford: Web Security & Commerce (Section III Digital Certificates; O'Reilly & Associates, Cambridge 1997.

[0038] In der Figur 3 ist ein Flußplan zur Änderung der Funktionalität des Sicherheitsmoduls dargestellt.

[0039] Nach dem Einschalten eines - nicht gezeigten - Herstellergerätes wird Energie zur Verfügung gestellt und im Schritt 200 wird geprüft, ob das Einschalten den beabsichtigten Erfolg hatte, so dass am Sicherheitsmodul eine Systemspannung anliegt. Falls nicht, dann wird auf eine Warteschleife verzweigt und die Abfrage ständig wiederholt. Falls am Sicherheitsmodul die Systemspannung anliegt, wird im Schritt 201 ein Start up Programm gestartet und in den Arbeitsspeicher SRAM 121 wird mindestens ein erster Teil des Startladeprogramms mit der Programmierfunktionalität kopiert. Der Mikroprozessor 120 ist durch das Startladeprogramm programmiert, dass derjenige Speicherbereich des FLASH-Programmspeichers, in welchem sich das Startladeprogramm befindet, nur kopiert aber nicht überschrieben werden kann. Informationen zu einem bereits geladenen Anwendungsprogramm können in einem weiteren Speicherbereich des FLASH-Programmspeichers 128 oder an anderer Stelle nichtflüchtig gespeichert sein. Die Infomationen schließen eine Zustandsvariable ein. Bei der anschließenden Programmausführung, stellt der Mikroprozessor im Schritt 202 anhand der vorgenannten Informationen fest, ob ein gültiger Zustand eines Anwendungsprogramms vorliegt.

[0040] Ist letzteres der Fall, dann wird im Schritt 209 das Anwendungsprogramm gestartet. Anschließend wird ständig im Schritt 210 überprüft, ob im Kommunikationsinterface Daten zum Löschen des Anwendungsprogramms vorhanden sind. Ist das nicht der Fall, dann wird auf den Schritt 209 zurückverzweigt und das Anwendungsprogramm gestartet. Anderenfalls wird vom Schritt 210 auf einen Schritt 211 verzweigt, in welchem das vorhandene Anwendungsprogramm mittels einer Zustandsvariablen als 'ungültig' gekennzeichnet wird.

[0041] Beim nächsten Booten wird das Startladeprogramm (Bootloader) wieder aktiv und kann neue Anwendungsprogrammdaten speichern.

[0042] Zunächst wird vom Mikroprozessor im Schritt 202 festgestellt, dass das vorhandene Anwendungsprogramm als 'ungültig' gekennzeichnet wurde bzw. kein gültiger Zustand des Anwendungsprogramms vorliegt, dann wird vom Schritt 202 auf einen Schritt 203 verzweigt, in welchem ein zweiter Teil des Startladeprogramms mit einer Kommunikationsschnittstellenabruf und Überprüfungsfunktionalität gestartet wird. In einem nachfolgenden Abfrageschritt 204 wird überprüft, ob Anwendungsprogrammdaten und Kennungsdaten in der Kommunikationsschnittstelle vorliegen. Falls das nicht der Fall ist, dann wird auf eine Warteschleife verzweigt und die Abfrage ständig wiederholt. Ist letzteres aber der Fall, dann wird auf einen Abfrageschritt 205 verzweigt, in welchem geprüft wird, ob die Kennungsdaten einen Nachfolger vom gespeicherten Vorgänger kennzeichnen. Der Mikroprozessor vergleicht dazu die gelieferten Kennungsdaten mit gespeicherten Kennungsdaten. Letztere können im vorgenannten weiteren Speicherbereich des FLASH-Programmspeichers, in welchem alle Informationen zu einem bereits geladenen Programm gelistet sind, gespeichert werden. An die Kommunikationsschnittstelle werden vom Hersteller auch zu den Anwendungsprogrammdaten zugehörige Informationsdaten geliefert, wie: Start- und Endadresse des Program-

mes, Überprüfungssumme (CRC), Programmtyp, Version, Revision. Zu den Kennungsdaten gehören der Programmtyp, die Versions- und die Revisionsdaten.

[0043] Falls die in der Kommunikationsschnittstelle vorliegenden Kennungsdaten keinen Nachfolger vom gespeicherten Vorgänger betreffen, dann wird auf eine Warteschleife zum Schritt 204 zurückverzweigt. Falls die in der Kommunikationsschnittstelle vorliegenden Kennungsdaten einen Nachfolger vom gespeicherten Vorgänger kennzeichnen, dann wird auf einen Schritt 206 verzweigt. Der Mikroprozessor wird entsprechend dem o.g. ersten Teilprogramm des Startladeprogramms mit der Programmierfunktionalität gesteuert. Die kopierten Anwendungsprogrammdaten werden auf einen für das Anwendungsprogramm vorgesehenen Speicherplatz des Programmspeichers gespeichert.

[0044] Ein zum Anwendungsprogramm zugehöriges Echtheitszertifikat, beispielsweise eine kyptographische Signatur, wird im nachfolgenden Abfrageschritt 207 zur Überprüfung der Echtheit des Anwendungsprogramms benutzt. Liegt jedoch keine Echtheit vor, dann wird auf den Abfrageschritt 204 zurückverzweigt. Ein echtes Anwendungsprogramm veranlaßt im nachfolgenden Schritt 208, dass eine Information über einen gültigen Zustand nichtflüchtig gespeichert und dann zum Abfrageschritt 204 zurückverzweigt wird. Bei Authentizität des geladenen Programmteils, der mindestens einen Teil des Anwendungsprogramms einschließt, wird beispielsweise eine Zustandsvariable im nichtflüchtigen Speicher des Sicherheitsmoduls gespeichert oder auf den o.g. weiteren Speicherplatz für Informationsdaten eingeschrieben, die den vorgenannten geladenen Programmteil als gültig kennzeichnet. Vorzugsweise ist die Zustandsvariable ein Flag, mit welchem das geladene Anwendungsprogramm als gültig gekennzeichnet wird, nachdem eine kryptographische Signatur verifiziert wurde, welche die Authentizität des geladenen Anwendungsprogrammes beweist.

[0045] Neue gültige Programmdaten, deren zugehörige Kennungsdaten einen Nachfolger kennzeichnen, werden nur dann auf einen Speicherplatz eingeschrieben, wenn das bereits vorhandene Programm zuvor im Schritt 211 mit der Zustandsvariablen 'ungütig' gekennzeichnet wurde. Letzteres setzt voraus, dass Daten zum Löschen des Anwendungsprogramms im Kommunikationsinterface vorhanden sind (Schritt 210).

[0046] Aufgrund der dadurch möglichen Änderung seiner Funktionalität ist das Sicherheitsmodul an unterschiedliche Geräte anpaßbar und kann zur Lösung einer Vielfalt an Aufgaben eingesetzt werden.

[0047] Das Sicherheitsmodul, welches zum Einsatz in postalischen Geräten, insbesondere zum Einsatz in einer Frankiermaschine, bestimmt ist, wird als postalisches Sicherheitsmodul (Postal Security Device) oder als sicheres Abrechnungsgerät (Security Accounting Device) bezeichnet. Ein PSD beruht wie ein SAD auf einer gleichen Hardware. Das PSD nutzt einen asymmetrischen Verschlüsselungsalgorithmus (RSA, ECDSA), aber das SAD nutzt einen symmetrischen Verschlüsselungsalgorithmus (DES, triple-DES). Das Sicherheitsmodul kann auch eine andere Bauform aufweisen, die es ermöglicht, daß es in unterschiedlichen Geräten arbeiten kann. Somit wird es ermöglicht, dass es beispielsweise auf die Hauptplatine eines Personalcomputers gesteckt werden kann, der als PC-Frankierer einen handelsüblichen Drucker ansteuert.

[0048] Die Erfindung ist nicht auf die vorliegenden Ausführungsform beschränkt, da offensichtlich weitere andere Anordnungen bzw. Ausführungen der Erfindung entwickelt bzw. eingesetzt werden können, die - vom gleichen Grundgedanken der Erfindung ausgehend - von den anliegenden Schutzansprüchen umfaßt werden.

Patentansprüche

1. Anordnung zum Ändern der Funktionalität eines Sicherheitsmoduls, der einen Mikroprozessor (120) und einen reprogrammierbaren Programmspeicher (128) aufweist, der ein Startladeprogramm speichert, welches teilweise in einen Arbeitsspeicher (121) kopierbar ist, und dass der reprogrammierbare Programmspeicher (128) einen freien Speicherplatz für ein nachfolgend ladbares Anwendungsprogramm aufweist, auf dem ein bisheriges Anwendungsprogramm gespeichert ist, dass der Sicherheitsmodul einen spezellen Schaltkreis (160) aufweist, der mit einem Kommunikationsinterface (150) zur Herstellung einer Kommunikationsverbindung mit einem Herstellergerät ausgestattet ist, welches eine Systemspannung und Anwendungsprogrammdaten für das Sicherheitsmodul bereitstellt, wobei der Mikroprozessor (120) über einen BUS mit dem Arbeitsspeicher (121), mit dem Programmspeicher (128) und mit einem Kommunikations-interface (150) in kommunikativer Verbindung steht, dadurch gekenn-zeichne t, dass das Kommunikationsinterface (150) zur Bereitstellung von Daten mindestens eines Teils eines Anwendungsprogramms, eines zugehörigen Zertifikatcodes und Kennungsdaten ausgebildet ist und dass der Mikroprozessor (120) durch das teilweise in einen Arbeitsspeicher (121) kopierte Startladeprogramm programmiert ist,

   - den Programmzustand des bisherigen Anwendungsprogramms anhand einer Zustandsvariablen zu verifizieren und dessen Kennungsdaten zu verifizieren,
   - die Daten des bereitgestellten Teils des nachfolgend ladbaren Anwendungsprogramms auf dem für das nachfolgend ladbare Anwendungsprogramm freien Speicherplatz des Programmspeichers (128) zu speichern, wenn die Zustandsvariable den Programmzustand des bisherigen Anwendungsprogramms als ungültig sowie die Kennungsdaten des geladenen mindestens einen Teils des modifizierten Anwen-

dungsprogramms einen Nachfolger zur gespeicherten Vorgängerkennung kennzeichnen,
- die Authentizität des geladenen mindestens einen Teils des nachfolgend ladbaren Anwendungsprogramms mittels des Zertifikatcodes zu überprüfen und bei Authentizität des geladenen Teils des nachfolgend ladbaren Anwendungsprogramms letzteres mit der Zustandsvariablen als gültig zu speichern.

2. Anordnung, nach Anspruch 1, **dadurch gekennzeichnet, dass** der Programmspeicher (128) ein FLASH-Programmspeicher ist, dass das Kommunikationsinterface (150) eine interne Steuerung und einen Kommunikations-Puffer einschließt, aus welchem zuerst eingelesene Daten zuerst ausgelesen und weitergeleitet werden sowie dass ein Mikroprozessor-Typ vorgesehen ist, der die Ausführung seines Programms in einem Arbeitsspeicher (121) ermöglicht.

3. Verfahren zum Ändern der Funktionalität eines Sicherheitsmoduls, dessen Programmspeicher (128) reprogrammierbar ist und einen freien Speicherplatz für ein nachfolgend ladbares Anwendungsprogramm aufweist, auf dem ein bisheriges Anwendungsprogramm gespeichert ist, wobei die Funktionalität geändert wird, mittels Reprogrammierung des Programmspeichers unter Verwendung eines im Programmspeicher gespeicherten Startladeprogramms, welches zum Ausführen eines Programmteils teilweise in einen Arbeitsspeicher kopiert wird, **gekennzeichnet durch** die Schritte:

- Ausführen eines Programmteils, welches teilweise in einen Arbeitsspeicher kopiert wurde,
- Verifizieren eines beim Programmieren erreichten Programmzustandes des bisherigen Anwendungsprogramms anhand einer Zustandsvariablen, wobei der Programmzustand nichtflüchtig gespeichert ist, um die Programmfunktionalität zustandsabhängig ausführen zu können und Verifizieren der Kennungsdaten des bisherigen Anwendungsprogramms,
- Speichern der Daten des Teils des nachfolgend ladbaren Anwendungsprogramms auf dem für ein Anwendungsprogramm freien Speicherplatz des Programmspeichers (128), wenn die Zustandsvariable den Programmzustand des bisherigen Anwendungsprogramms als ungültig sowie wenn die Kennungsdaten des geladenen mindestens einen Teils des nachfolgend ladbaren Anwendungsprogramms einen Nachfolger zur gespeicherten Vorgängerkennung kennzeichnen,
- Autorisieren der geänderten Funktionsweise des nachgeladenen Teils des nachfolgend ladbaren Anwendungsprogramms bei dessen Authentizität und Speichern des nachgeladenen Teils des neuen Anwendungsprogramms als gültig bei dessen Authentizität und Speichern der Zustandsvariablen.

4. Verfahren, nach Anspruch 3, **dadurch gekennzeichnet, dass** in einem Kommunikationsinterface (150) Daten mindestens eines Teils eines Anwendungsprogramms, ein zugehöriger Zertifikatcode und zugehörige Kennungsdaten bereitgestellt und dass die Daten mindestens des einen Teils des Anwendungsprogramms auf einem freien Speicherplatz des Programmspeichers gespeichert werden, wenn die Kennungsdaten einen Nachfolger zur gespeicherten Vorgängerkennung kennzeichnen, sowie dass die Authentizität des geladenen mindestens einen Teils des Anwendungsprogramms mittels des Zertifikatcodes überprüft wird, wobei bei Authentizität des geladenen mindestens einen Teils des Anwendungsprogramms letzteres als gültig gespeichert wird.

5. Verfahren, nach Anspruch 4, **dadurch gekennzeichnet, dass** bei Authentizität des geladenen mindestens einen Teils des Anwendungsprogramms eine Zustandsvariable gespeichert wird, die vorgenannten geladenen Programmteil als gültig kennzeichnet.

6. Verfahren, nach Anspruch 3, **dadurch gekennzeichnet, dass** ein Code-Überprüfungsschlüssel zur Überprüfung der Authentizität des geladenen Teils des Anwendungsprogramms bereitgestellt wird.

7. Verfahren, nach Anspruch 6, **dadurch gekennzeichnet, dass** das Bereitstellen ein Laden des Code-Überprüfungsschlüssels einschließt, welcher während der Herstellung im Rahmen einer Vorinitialisierung geladen wird, dass der Code-Überprüfungsschlüssel im Sicherheitsmodul auslesesicher gespeichert wird und ein geheimer Schlüssel eines symmetrischen Verschlüsselungsverfahren ist.

8. Verfahren, nach Anspruch 6, **dadurch gekennzeichnet, dass** das Bereitstellen ein Laden des Code-Überprüfungsschlüssels einschließt, welcher während der Herstellung im Rahmen einer Vorinitialisierung geladen wird, dass der Code-Überprüfungsschlüssel ein öffentlicher Verifizierschlüssel ist, dass der öffentliche Verifizierschlüssel und ein zugehöriger geheimer Signierschlüssel ein Schlüsselpaar bilden, wobei der Zertifikatcode mit Hilfe des geheimen Signierschlüssels vom Hersteller zugehörig zu den Daten mindestens eines Teils eines Anwendungsprogramms erzeugt wird.

9. Verfahren, nach Anspruch 4, **dadurch gekenn-**

zeichnet, dass der Mikroprozessor (120) feststellt, ob die Kennungsdaten einen Nachfolger zur gespeicherten Vorgängerkennung kennzeichnen, indem die Kennungsdaten mit entsprechenden Vergleichsdaten verglichen werden, die in einem weiteren Speicherbereich des Programmspeichers (128) gespeichert sind, in welchem Informationsdaten zu einem bereits geladenen Programm gelistet sind.

10. Verfahren, nach Anspruch 9, **dadurch gekennzeichnet, dass** die Informationsdaten vom Hersteller zu den Anwendungsprogrammdaten zugehörig an die Kommunikationsschnittstelle (150) geliefert werden, wobei die Informationsdaten, die Start- und Endadresse des Programmes, Überprüfungssumme (CRC) und Kennungsdaten umfassen.

11. Verfahren, nach den Ansprüchen 9 bis 10, **dadurch gekennzeichnet, dass** zu den Kennungsdaten der Programmtyp, die Versions- und die Revisionsdaten gehören.

12. Verfahren, nach den Ansprüchen 3 bis 5, **dadurch gekennzeichnet, dass** die zum Verifizieren eines beim Programmieren erreichten Programmzustandes benötigte Zustandsvariable im Programmspeicher (128) oder in einem nichtflüchtigen Speicher des Sicherheitsmoduls gespeichert vorliegt.

13. Verfahren, nach Anspruch 12, **dadurch gekennzeichnet, dass** die Zustandsvariable ein Flag ist, mit welchem das geladene Anwendungsprogramm als gültig gekennzeichnet wird, nachdem eine kryptographische Signatur verifiziert wurde, welche die Authentizität des geladenen Anwendungsprogrammes beweist.

Claims

1. An arrangement for changing the functionality of a security module having a microprocessor (120) and a reprogrammable program memory (128) that stores a start loading program that is in part copyable into a working memory (121), wherein the reprogrammable program memory (128) has a free memory space for a subsequently loadable application program in which a current application program is stored, wherein the security module has a special circuit (160) equipped with a communication interface (150) for establishing a communication connection with a manufacturer unit providing a system voltage and application program data for the security module, the microprocessor (120) being, via a BUS, in a communicative connection with the working memory (121), with the program memory (128) and with a communication interface (150), **characterized in that** the communication interface (150) is designed for providing data of at least part of an application program, of a related certificate code and identification data and that the microprocessor (120) is programmed by the start loading program partially copied into a working memory (121) to

- verify the program status of the current application program on the basis of a status variable and verify its identification data,
- store the data of the provided part of the subsequently loadable application program in the memory space of the program memory (128) free for the subsequently loadable application program when the status variable marks the program status of the current application program as invalid and the identification data of the at least one part of the modified application program loaded mark a successor to the stored previous identification,
- check the authenticity of the at least one part loaded of the subsequently loadable application program by means of the certificate code and, in case of authenticity of the loaded part of the subsequently loadable application program, store the latter with the status variable as valid.

2. An arrangement according to Claim 1, **characterized in that** the program memory (128) is a FLASH-type program memory, that the communication interface (150) includes an internal control and a communication buffer from which data read in first are read out and forwarded first and that there is provided a microprocessor type that allows the execution of its program in a working memory (121).

3. A method for changing the functionality of a security module the program memory (128) of which is reprogrammable and has a free memory space for a subsequently loadable application program in which a current application program is stored, wherein the functionality is changed by reprogramming the program memory using a start load program stored in the program memory copied in part into a working memory for execution of a part of the program, **characterized by** the following steps:

- execution of a program part partially copied into a working memory,
- verification of a program status of the current application program reached by programming on the basis of a status variable, wherein the program status is stored in a non-volatile manner for being able to execute the program functionality in a status-dependent manner, and verification of the identification data of the current application program;
- storing of the data of the part of the subsequently loadable application program in the

memory space of the program memory (128) free for an application program when the status variable marks the program status of the current application program as invalid and when the identification data of the loaded at least one part of the subsequently loadable application program mark a successor to the stored previous identification,

- authorization of the changed mode of functioning of the reloaded part of the subsequently loadable application program when it is authentic and storage of the loaded part of the new application program as valid when it is authentic and storage of the status variable.

4. A method according to Claim 3, **characterized in that** data of at least part of an application program, a related certification code and related identification data are provided in a communication interface (150), and that the data of the at least one part of the application program are stored in a free memory space of the program memory when the identification data mark a successor to the stored predecessor identification, and that the authenticity of the loaded at least one part of the application program is checked by means of the certificate code, wherein the latter is stored as valid when the loaded at least one part of the application program is authentic.

5. A method according to Claim 4, **characterized in that**, when the at least one part of the application program loaded is authentic, there is stored a status variable marking the above-mentioned loaded program part as valid.

6. A method according to Claim 3, **characterized in that** there is provided a code verification key for verifying the authenticity of the part of the application program loaded.

7. A method according to Claim 6, **characterized in that** the provision includes a loading of the code verification key that is loaded in the course of manufacturing within the scope of a pre-initialization, that the code verification key is stored in the security module in a readout-proof manner and is a secret key of a symmetrical encoding method.

8. A method according to Claim 6, **characterized in that** the provision includes a loading of the code verification key that is loaded in the course of manufacturing within the scope of a pre-initialization, that the code verification key a public verification key, that the public verification key and a related secret signing key form a key pair, wherein the certificate code is generated by the manufacturer with the help of the secret signing key assigned to the data of the at least one part of an application program.

9. A method according to Claim 4, **characterized in that** the microprocessor (120) determines whether the identification data mark a successor to the predecessor identification stored by comparing the identification data with respective reference data stored in a further memory area of the program memory (128) in which information data for an already loaded program are enlisted.

10. A method according to Claim 9, **characterized in that** the information data are delivered by the manufacturer related to the application program data to the communication interface (150), said information data comprising the start and end address of the program, the checksum (CRC) and identification data.

11. A method according to Claims 9 to 10, **characterized in that** the identification data include program type, version and revision data.

12. A method according to Claims 3 to 5, **characterized in that** the status variable required for the verification of a program status reached upon programming is available stored in the program memory (128) or in a non-volatile memory of the security module.

13. A method according to Claim 12, **characterized in that** the status variable is a flag by which the application program loaded is marked as valid after a cryptographic signature proving the authenticity of the application program loaded has been verified.

Revendications

1. Système pour changer la fonctionnalité d'un module de sécurité, qui comporte un microprocesseur (120) et une mémoire de programme (128) reprogrammable, qui mémorise un chargeur d'amorçage pouvant partiellement être copié dans une mémoire vive (121), ladite mémoire de programme (128) reprogrammable comportant un espace de mémoire libre pour le programme d'application chargeable subséquemment, sur lequel est enregistré un ancien programme d'application, que le module de sécurité comporte un circuit (160) spécial équipé d'une interface de communication (150) destinée à l'établissement d'une liaison de communication avec un appareil d'un fabricant, qui met à disposition du module de sécurité une tension de système et des données du programme d'application, et dont le microprocesseur (120) est en liaison de communication par l'intermédiaire d'un bus avec la mémoire vive (121), la mémoire de programme (128) et une interface de communication (150), **caractérisé en ce, que** l'interface de communication (150) pour la mise à disposition de données soit formée par au moins une

partie d'un programme d'application, d'un code de certification correspondant et de données d'identification et que le microprocesseur (120) soit programmé par le chargeur d'amorçage copié partiellement dans une mémoire vive (121) pour,

- vérifier, à l'aide d'une variable d'état, l'état de programme de l'ancien programme d'application et pour vérifier les données d'identification correspondantes,
- enregistrer les données de la partie mise à disposition du programme d'application chargeable subséquemment sur l'espace de mémoire libre de la mémoire de programme (128), ledit espace de mémoire libre étant destiné au programme d'application chargeable subséquemment, si la variable d'état caractérise l'invalidité de l'état de programme de l'ancien programme d'application ainsi que si les données d'identification de l'au moins une partie chargée du programme d'application modifié caractérisent un successeur par rapport à l'identificateur du prédécesseur enregistré,
- contrôler l'authenticité de l'au moins une partie chargée du programme d'application chargeable subséquemment à l'aide du code de certification et d'enregistrer comme valide le programme d'application chargeable subséquemment en cas d'authenticité de la partie chargée dudit programme d'application chargeable subséquemment.

2. Système, selon la revendication 1, **caractérisé en ce, que** la mémoire de programme (128) soit une mémoire de programme FLASH, que l'interface de communication (150) comporte une commande interne et une mémoire tampon de communication, à partir de laquelle les données enregistrées en premier lieu soient délivrées et transmises en premier lieu, ainsi que soit prévu un type de microprocesseur qui permet d'exécuter son programme dans une mémoire vive (121).

3. Méthode pour changer la fonctionnalité d'un module de sécurité, dont la mémoire de programme (128) est reprogrammable et qui comporte un espace de mémoire libre pour un programme d'application chargeable subséquemment, sur lequel est enregistré un ancien programme d'application, et dont la fonctionnalité peut être modifiée à l'aide d'une reprogrammation de la mémoire de programme en utilisant un chargeur d'amorçage enregistré dans la mémoire de programme, ledit chargeur d'amorçage étant copié partiellement dans une mémoire vive pour l'exécution d'une partie de programme, **caractérisée par** les étapes :

- exécution d'une partie de programme qui a été partiellement copiée dans une mémoire vive,
- vérification à l'aide d'une variable d'état d'un état de programme de l'ancien programme d'application, ledit état de programme ayant été atteint lors de la programmation, et dont cet état de programme est enregistré de manière non volatile, afin de pouvoir exécuter la fonctionnalité du programme en fonction de l'état et vérifier les données d'identification de l'ancien programme d'application,
- enregistrer les données de la partie du programme d'application chargeable subséquemment sur l'espace de mémoire libre de la mémoire de programme (128), ledit espace de mémoire libre étant destiné à un programme d'application, si la variable d'état caractérise l'invalidité de l'état de programme de l'ancien programme d'application ainsi que si les données d'identification de l'au moins une partie chargée du programme d'application chargeable subséquemment caractérisent un successeur par rapport à l'identificateur enregistré du prédécesseur,
- autorisation du mode de fonctionnement modifié de la partie chargée subséquemment du programme d'application chargeable subséquemment en cas d'authenticité, enregistrement comme valide de la partie chargée subséquemment du nouveau programme d'application en cas d'authenticité de celui-ci et enregistrement des variables d'état.

4. Méthode, selon la revendication 3, **caractérisée en ce, que** soient mises à disposition dans une interface de communication (150) des données d'au moins une partie d'un programme d'application, un code de certification correspondant et des données d'identification correspondantes, et que les données de cette au moins une partie du programme d'application soient enregistrées sur un espace de mémoire libre de la mémoire de programme, si les données d'identification caractérisent un successeur par rapport à l'identificateur du prédécesseur, ainsi que l'authenticité de l'au moins une partie chargée du programme d'application soit vérifiée à l'aide du code de certification, et dont cette au moins une partie chargée du programme d'application soit enregistrée comme valide en cas d'authenticité.

5. Méthode, selon la revendication 4, **caractérisée en ce, qu'**en cas d'authenticité de l'au moins une partie chargée du programme d'application, une variable d'état, qui caractérise comme valide la partie de programme chargée précitée, soit enregistrée.

6. Méthode, selon la revendication 3, **caractérisée en ce, qu'**une clé de contrôle de code soit mise à disposition pour vérifier l'authenticité de la partie char-

gée du programme d'application.

7. Méthode, selon la revendication 6, **caractérisée en ce, que** la mise à disposition intègre un chargement de la clé de contrôle de code, laquelle étant chargée pendant la fabrication dans le cadre d'une pré-initialisation, que la clé de contrôle de code soit enregistrée dans le module de sécurité de manière sûre contre la lecture et qu'elle soit une clé secrète d'un procédé de cryptographie symétrique.

8. Méthode, selon la revendication 6, **caractérisée en ce, que** la mise à disposition intègre un chargement de la clé de contrôle de code, laquelle étant chargée pendant la fabrication dans le cadre d'une pré-initialisation, que la clé de contrôle de code soit une clé publique de vérification, que la clé publique de vérification et une clé secrète de signature correspondante forment une paire de clés, et dont le code de certification soit créé par le fabricant à l'aide de la clé secrète de signature en correspondance aux données de l'au moins une partie du programme d'application.

9. Méthode, selon la revendication 4, **caractérisée en ce, que** le microprocesseur (120) détermine si les données d'identification caractérisent un successeur par rapport à l'identificateur enregistré du prédécesseur en comparant les données d'identification avec des données comparatives correspondantes, qui sont enregistrées dans un autre espace de mémoire de la mémoire de programme (128), dans laquelle sont listées des données informatives relatives à un programme déjà chargé.

10. Méthode, selon la revendication 9, **caractérisée en ce, que** les données informatives soient livrées par le fabricant à l'interface de communication (150) en correspondance avec les données du programme d'application, les données informatives comportant l'adresse initiale et finale du programme, la somme de contrôle (CRC / contrôle par redondance cyclique) et les données d'identification.

11. Méthode, selon les revendications 9 à 10, **caractérisée en ce, que** le type de programme, les données de la version et de la révision fassent partie des données d'identification.

12. Méthode, selon les revendications 3 à 5, **caractérisée en ce, que** la variable d'état, nécessaire pour la vérification d'un état de programme atteint lors de la programmation, soit enregistrée et disponible dans la mémoire de programme (128) ou dans une mémoire non volatile du module de sécurité.

13. Méthode, selon la revendication 12, **caractérisée en ce, que** la variable d'état soit un drapeau qui permet de caractériser le programme d'application chargé comme valide après la vérification d'une signature cryptographique, qui prouve l'authenticité du programme d'application chargé.

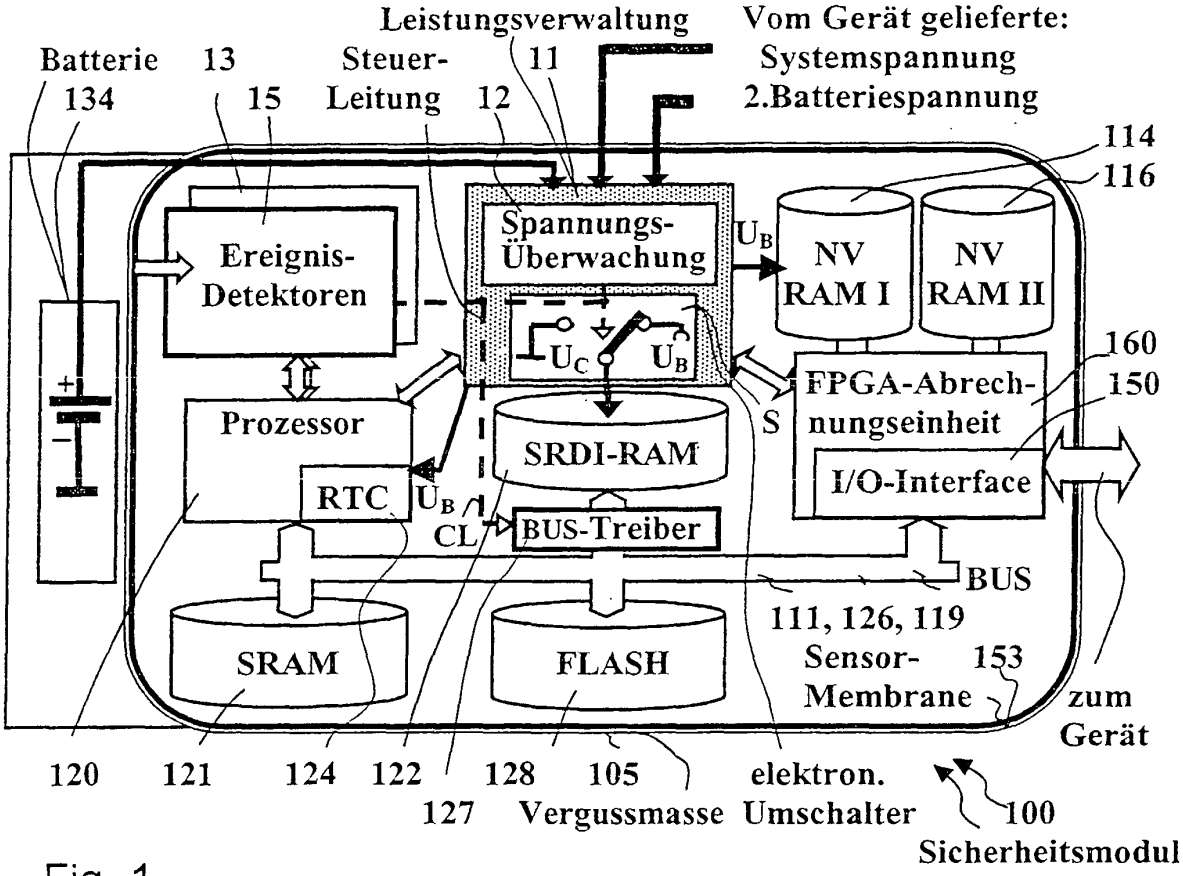
Leistungsverwaltung
Vom Gerät gelieferte:
Systemspannung
2.Batteriespannung
Batterie
134
13
15
Steuer-
Leitung
12
11
114
116
Spannungs-
Überwachung
U_B
NV
RAM I
NV
RAM II
Ereignis-
Detektoren
U_C
U_B
160
150
FPGA-Abrech-
nungseinheit
S
Prozessor
SRDI-RAM
I/O-Interface
RTC
U_B
CL
BUS-Treiber
BUS
111, 126, 119
Sensor-
Membrane
153
zum
Gerät
SRAM
FLASH
120
121
124
122
128
105
127
Vergussmasse
elektron.
Umschalter
100
Sicherheitsmodul

Fig. 1

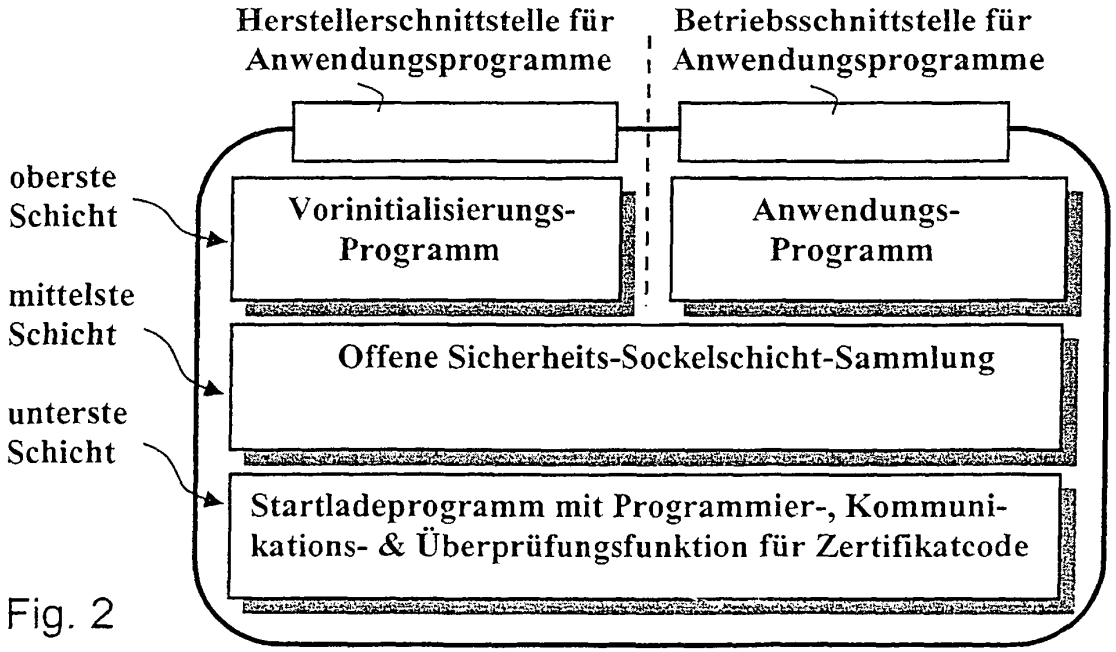
Herstellerschnittstelle für
Anwendungsprogramme
Betriebsschnittstelle für
Anwendungsprogramme
oberste
Schicht
mittelste
Schicht
unterste
Schicht
Vorinitialisierungs-
Programm
Anwendungs-
Programm
Offene Sicherheits-Sockelschicht-Sammlung
Startladeprogramm mit Programmier-, Kommuni-
kations- & Überprüfungsfunktion für Zertifikatcode

Fig. 2

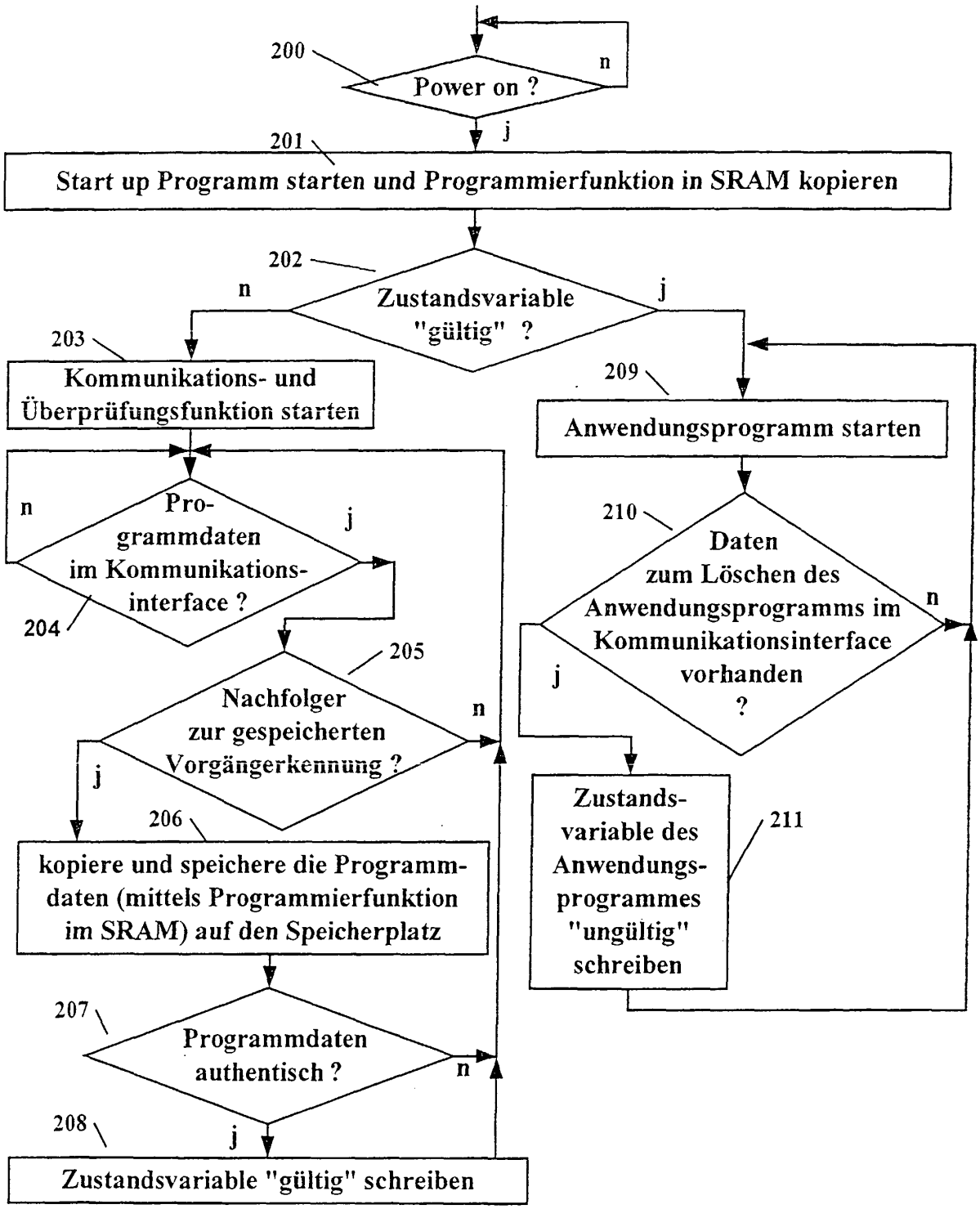
200
Power on ?
n
j
201
Start up Programm starten und Programmierfunktion in SRAM kopieren
202
Zustandsvariable "gültig" ?
n
j
203
Kommunikations- und Überprüfungsfunktion starten
209
Anwendungsprogramm starten
Pro-grammdaten im Kommunikations-interface ?
n
j
204
205
Nachfolger zur gespeicherten Vorgängerkennung ?
n
j
206
kopiere und speichere die Programm-daten (mittels Programmierfunktion im SRAM) auf den Speicherplatz
207
Programmdaten authentisch ?
n
208
j
Zustandsvariable "gültig" schreiben
210
Daten zum Löschen des Anwendungsprogramms im Kommunikationsinterface vorhanden ?
n
j
Zustands-variable des Anwendungs-programmes "ungültig" schreiben
211

Fig. 3

IN DER BESCHREIBUNG AUFGEFÜHRTE DOKUMENTE

Diese Liste der vom Anmelder aufgeführten Dokumente wurde ausschließlich zur Information des Lesers aufgenommen und ist nicht Bestandteil des europäischen Patentdokumentes. Sie wurde mit größter Sorgfalt zusammengestellt; das EPA übernimmt jedoch keinerlei Haftung für etwaige Fehler oder Auslassungen.

In der Beschreibung aufgeführte Patentdokumente

- EP 417447 B1 **[0002]**
- EP 789333 A2 **[0003]**
- DE 19816572 A1 **[0004]**
- DE 19816571 A1 **[0004]**
- EP 1035516 A2 **[0004]**
- EP 1035517 A2 **[0004]**
- EP 1035518 A2 **[0004] [0005]**
- EP 1035513 A2 **[0004]**
- DE 20020635 U1 **[0004] [0005] [0025]**
- US 4528644 A **[0006]**
- EP 724141 B1 **[0007]**
- EP 788115 A2 **[0008]**
- EP 1087294 A2 **[0011]**

In der Beschreibung aufgeführte Nicht-Patentliteratur

- **BRUCE SCHNEIER.** Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley & Sons, 1996 **[0037]**
- **SIMSON GARFINKEL ; GENE SPAFFORD.** Web Security & Commerce. O'Reilly & Associates, 1997 **[0037]**