(19) Deutsches Patent- und Markenamt

(10) **DE 101 96 007 B4** 2010.08.12

(12) # Patentschrift

(21) Deutsches Aktenzeichen: **101 96 007.7**
(86) PCT-Aktenzeichen: **PCT/US01/08891**
(87) PCT-Veröffentlichungs-Nr.: **WO 2001/075564**
(86) PCT-Anmeldetag: **21.03.2001**
(87) PCT-Veröffentlichungstag: **11.10.2001**
(43) Veröffentlichungstag der PCT Anmeldung in deutscher Übersetzung: **09.10.2003**
(45) Veröffentlichungstag der Patenterteilung: **12.08.2010**

(51) Int Cl.[8]: ***G06F 1/00*** (2006.01)
G06F 12/14 (2006.01)

Innerhalb von drei Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten(§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 1 Patentkostengesetz).

(30) Unionspriorität:
09/541,108 **31.03.2000** **US**

(73) Patentinhaber:
Intel Corporation, Santa Clara, Calif., US

(74) Vertreter:
ZENZ Patent- und Rechtsanwälte, 45128 Essen

(72) Erfinder:
Herbert, Howard C., Phoenix, Ariz., US; Grawrock, David W., Aloha, Oreg., US; Ellison, Carl M., Portland, Oreg., US; Golliver, Roger A., Beaverton, Oreg., US; Lin, Derrick C., San Mateo, Calif., US; McKeen, Francis X., Portland, Oreg., US; Neiger, Gilbert, Portland, Oreg., US; Reneris, Ken, Wilbraham, Mass., US; Sutton, James A., Portland, Oreg., US; Thakkar, Shreekant S., Portland, Oreg., US; Mittal, Millind, Palo Alto, Calif., US

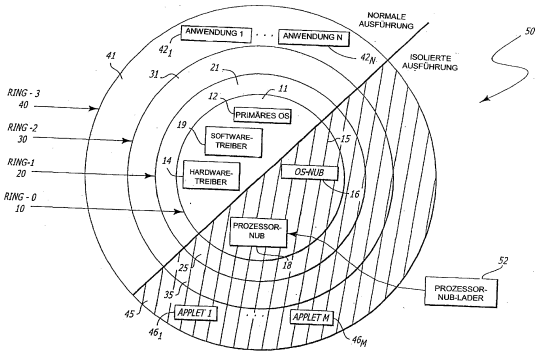
(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:

US	**59 53 502**	**A**
EP	**08 49 657**	**A1**
WO	**98/15 082**	**A1**

(54) Bezeichnung: **Plattform und Verfahren zum Fernattestieren einer Plattform**

(57) Hauptanspruch: Eine Plattform, enthaltend:
einen Prozessor (110), der entweder in einem normalen Ausführungsmodus oder in einem isolierten Ausführungsmodus arbeitet;
einen Systemspeicher (140), der einen isolierten Bereich und einen nicht-isolierten Bereich enthält, wobei der Prozessor nur dann auf den isolierten Bereich zugreifen kann, wenn er in dem isolierten Ausführungsmodus arbeitet;
einen Chipsatz (130, 150; 310; 400, 420) mit einer Komponente (310; 420; 400; 150), die einen geschützten Speicher (152) aufweist, der ein Prüfprotokoll (156) oder, wenn das Prüfprotokoll (156) in einem nicht-geschützten Speicher gespeichert ist, einen Gesamt-Hash-Wert des Prüfprotokolls (156) speichert, wobei das Prüfprotokoll (156) repräsentative Daten von nach dem Einschalten in der Plattform geladenen Software-Modulen enthält;
wobei in dem Prozessor (110) oder in einer Komponente (400) des Chipsatzes oder in einem mit dem Chipsatz über einen Token-Bus (185) gekoppelten Token (180; 182) eine Fernattestiereinheit (300) enthalten ist,

wobei zwischen dem die Fernattestiereinheit (300) enthaltenden Prozessor...



Beschreibung

[0001] Die Erfindung betrifft eine Plattform mit einem Prozessor, einem Systemspeicher, einem Chipsatz und mit einer eine Attestierung der Plattform unterstützenden Attestiereinheit sowie ein Verfahren zum Attestieren einer solchen Plattform.

[0002] Fortschritte in Mikroprozessor- und Kommunikationstechnologien bei einer Plattform eröffneten viele Möglichkeiten für Anwendungen, die über traditionelle Wege der geschäftlichen Betätigung hinausgehen. Elektronischer Handel (e-commerce) und Business-to-Business(B2B)-Transaktionen werden jetzt populär, wobei sie die globalen Märkte mit hoher Geschwindigkeit erreichen. Während die moderne Mikroprozessortechnologie Benutzern zweckmäßige und leistungsfähige Methoden zum Handel treiben, Kommunizieren und Ausführen von Transaktionen zur Verfügung stellt, unterstützt diese Technologie keine Fernattestierung (remote attestation). Fernattestierung ist eine Methode zur Feststellung des Betriebszustandes einer entfernt angeordneten Plattform in einer generell sicheren Weise. Durch Bestimmung des Betriebszustandes der Plattform vor dem Ausführen von E-Commerce- oder B2B-Transaktionen mit dieser Plattform, wird dem Benutzer größeres Vertrauen in die Sicherheit der Transaktion vermittelt.

[0003] Die US 5,953,502 A befaßt sich mit der Computersicherheit. Aus der Druckschrift ist ein Computersystem (eine Plattform) bekannt, das die physische Isolation einer CPU durch Montage auf einer Tochterplatine und Befestigen der Tochterplatine in dem CPU-Steckplatz des Motherboards beschreibt. Die Tochterplatine enthält ein Coprozessor-Subsystem in Form eines RISC-Mikroprozessorchips und eines Multiprozessor-Logik-Controllers. Der Multiprozessor-Logik-Controller bestimmt, wann Signalleitungen zur und aus der CPU getrennt bzw. wieder verbunden werden. Durch das Trennen der Verbindungsleitungen zwischen dem Rest des Computers und dem Mikroprozessor wird eine Sicherheitsüberprüfung ermöglicht. Somit kann der Prozessor durch die zwischengeschaltete Platine physisch isoliert werden. Die Druckschrift beschreibt ferner die Speicherung digitaler Signaturen für das BIOS in einem Speicher auf der Tochterplatine, wobei dann, wenn später das BIOS geladen wird, die digitale Signatur von dem RISC-Coprozessor neu berechnet und mit der gespeicherten verglichen wird. Wenn die berechnete und die gespeicherte Signatur übereinstimmen, wird die Verbindung zwischen CPU und dem restlichen PC freigegeben (wobei zu beachten ist, daß der Adreßbus und der Datenbus des Prozessors ständig mit dem Rest des Computers verbunden sind, während bestimmte Signale des Steuerbusses modifiziert bzw. unterbrochen werden).

[0004] Aus der EP 0 849 657 A1 ist ein sicheres Computersystem bekannt, wobei die Authentizität mehrerer Komponenten des Systems bestimmt wird, indem in jeder Komponente bereits herstellerseitig eine digitale Signatur gespeichert wird, ein zugehöriger öffentlicher Schlüssel von dem Hersteller geliefert und in einem Speicher einer Sicherheitsschaltung des Computersystems abgelegt wird und dieser dann in einem kryptographischen Algorithmus verwendet wird, um die Authentizität der Komponente zu prüfen.

[0005] Die WO 98/15082 A1 offenbart eine Vorrichtung und ein Verfahren, wobei eine digitale Signatur eines BIOS-Programms in einem kryptographischen Coprozessor erzeugt und gespeichert wird.

[0006] Aufgabe der Erfindung ist es, eine zuverlässige Attestierung, d. h. eine Feststellung des Betriebszustandes und der Integrität der Plattform (einschließlich der Integrität sämtlicher in der Plattform geladener Software-Module) aus der Ferne zu ermöglichen, um einem Benutzer als Anforderer dieser Attestierung vor der Einleitung von Transaktionen zu dieser Plattform größeres Vertrauen in die Sicherheit der Transaktion zu vermitteln.

[0007] Diese Aufgabe wird erfindungsgemäß durch eine Plattform mit den Merkmalen des Anspruchs 1 sowie durch ein Verfahren mit den Merkmalen des Anspruch 24 gelöst. Vorteilhafte und/oder bevorzugte Weiterbildungen der Erfindung sind in den Unteransprüchen gekennzeichnet.

[0008] Die Merkmale und Vorteile der vorliegenden Erfindung werden aus der folgenden Detailbeschreibung der Erfindung deutlich, in der:

[0009] Fig. 1A ein Schema zur Veranschaulichung eines Ausführungsbeispiels der logischen Betriebsarchitektur für die IsoX™-Architektur der Plattform ist.

[0010] Fig. 1B ist ein veranschaulichendes Schema, das die Zugreifbarkeit verschiedener Elemente in dem Betriebssystem und dem Prozessor nach einem Ausführungsbeispiel der Erfindung zeigt.

[0011] Fig. 1C ist eine erste Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Plattform, die die vorliegende Erfindung benutzt.

[0012] Fig. 2 ist ein Ablaufdiagramm der veranschaulichenden Operationen der Plattform zur Erzeugung eines Ausführungsbeispiels des geschützten Prüfprotokolls.

[0013] Fig. 3 ist eine Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Fernattestiereinheit, die in dem Prozessor gemäß Fig. 1C zur Gewinnung einer geschützten Kopie des Prüfprotokolls verwendet wird.

[0014] Fig. 4 ist eine Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Fernattestiereinheit, die in dem Chipsatz gemäß Fig. 1C zum Gewinnen einer geschützten Kopie des Prüfprotokolls außerhalb des Chipsatzes verwendet wird.

[0015] Fig. 5 ist eine Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Fernattestiereinheit, die in dem Chipsatz gemäß Fig. 1C zum Gewinnen einer geschützten Kopie des Prüfprotokolls in dem Chipsatz verwendet wird.

[0016] Fig. 6 ist eine Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Fernattestiereinheit, die in dem festen Token gemäß Fig. 1C zum Gewinnen einer geschützten Kopie des Prüfprotokolls verwendet wird.

[0017] Fig. 7 ist eine Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Fernattestiereinheit, die in dem entnehmbaren Token gemäß Fig. 1C zum Gewinnen einer geschützten Kopie des Prüfprotokolls verwende wird.

[0018] Die vorliegende Erfindung bezieht sich auf eine Plattform und ein Verfahren zum Fernattestieren einer Plattform. Die Fernattestierung kann durchgeführt werden, während die Plattform in einem speziellen Betriebsmodus arbeitet. Ein Beispiel dieses speziellen Modus umfaßt einen isolierten Ausführungs-Prozessor-Modus (”IsoX”-Modus), wie unten beschrieben wird. Insbesondere verwendet ein im IsoX-Modus arbeitender Prozessor hardwaregeschütztes Verschlüsselungsmaterial, das kryptographisch einzigartig ist, um eine digitale Signatur zu erzeugen, die die Betriebsumgebung der Plattform betreffende Informationen enthält. Die den Schutz des Verschlüsselungsmaterials zur Verfügung stellende Hardware, die hier als ”Fernattestiereinheit” (RAU; Remote Attestation Unit) bezeichnet wird, kann in einem Kernlogikbauelement (z. B. einem Prozessor oder einer Chipsatzkomponente) oder in einem Nicht-Kernlogikbauelement (z. B. Token) integriert sein.

[0019] In der folgenden Beschreibung wird eine bestimmte Terminologie zur Erörterung von Merkmalen der vorliegenden Erfindung verwendet. Beispielsweise umfaßt eine ”Plattform” Komponenten, die unterschiedliche Funktionen an gespeicherten Informationen ausführen. Beispiele einer Plattform umfassen, ohne darauf beschränkt zu sein, einen Computer (z. B. Desktop, einen Laptop, einen Handheld, einen Server, eine Workstation u. s. w.), Desktop-Büroausrüstung (z. B. Drucker, Scanner, ein Telefaxgerät u. s. w.), ein schnurloses Telefongerät, eine Fernseh-Set-Top-Box u. dgl. Beispiele einer ”Komponente” umfassen Hardware (z. B. eine integrierte Schaltung u. s. w.) und/oder einen oder mehrere Softwaremodule. Ein ”Softwaremodul” ist ein Code, der bei Ausführung eine gewisse Funktion erfüllt. Dieser Code kann ein Betriebssystem, eine Anwendung, ein Applet oder sogar ein Nub, das eine Reihe von Codebefehlen, möglicherweise eine Untermenge eines Codes aus einer Applet darstellt, enthalten. Eine ”Link” (Verbindung) ist in breiter Definition mindestens ein informationsführendes Medium (z. B. Draht, Lichtleiter, Kabel, Bus oder Luft in Verbindung mit drahtloser Signaltechnologie), um einen Kommunikationsweg zu etablieren. Dieser Weg wird als ”geschützt” angesehen, wenn es nahezu unmöglich ist, über diesen Weg geleitete Informationen zu modifizieren, ohne daß dies bemerkt würde.

[0020] Außerdem ist der Begriff ”Informationen” als ein oder mehrere Bits von Daten, Adressen und/oder Steuersignalen definiert, und ein ”Segment” ist mindestens ein Byte von Informationen. Eine ”Nachricht” ist eine Gruppierung von Informationen, möglicherweise von Paketinformationen. ”Verschlüsselungsmaterial” umfaßt beliebige für einen speziellen kryptographischen Algorithmus, beispielsweise einen digitalen Signaturalgorithmus benötigte Information. Eine ”Einweg-Funktion” ist eine Funktion, mathematische oder andere, welche Informationen von einer variablen Länge in Informationen einer festen Länge umsetzt (die als ”Hash-Wert” oder ”Digest” bezeichnet wird). Der Ausdruck ”Einweg” zeigt an, daß es nicht wirklich eine Umkehrfunktion zum Wiedergewinnen eines erkennbaren Teils der Ursprungsinformationen aus dem Hash-Wert fester Länge gibt. Beispiele einer Hash-Funktion umfassen MD5, geliefert von der RSA Data Security aus Redwood City, Kalifornien, oder Secure Hash Algorithm (SHA-1), spezifiziert in der 1995er Veröffentlichung Secure Hash Standard FIPS 180-1 mit dem Titel ”Federal Information Processing Standards Publication” (April 17, 1995).

I. Architekturübersicht

[0021] Die die vorliegende Erfindung benutzende Plattform ist mit einer Isolierte-Ausführungs-Architektur (ISoX™-Architektur) konfiguriert. Die IsoX™-Architektur umfaßt logische und physikalische Definitionen von Hardware- und Software-Komponenten, die direkt oder indirekt mit einem Betriebssystem der Plattform in Wechselwirkung stehen. Hierbei können das Betriebssystem und ein Prozessor der Plattform verschiedene Hierarchieniveaus, bezeichnet als Ringe, haben, die verschiedenen Operationsmodi entsprechen. Ein ”Ring” ist eine logische Unterteilung von Hardware- und Software-Komponenten, die zum Ausführen vorgegebener Tasks innerhalb der Plattform konzipiert sind. Die Unterteilung basiert typischerweise auf dem Privilegierungsgrad oder -niveau, nämlich der Fähigkeit zum Vornehmen von Änderungen an der Plattform. Beispielsweise ist der Ring-0 der innerste Ring, der auf dem höchsten Hierarchieniveau ist. Ring-0 umgibt die kritischsten und privilegiertesten Komponenten. Ring-3 ist der äu-

ßerste Ring, der sich auf dem niedrigsten Hierarchieniveau befindet. Ring-3 umfaßt typischerweise Anwendungen auf Benutzerniveau, denen normalerweise das niedrigste Privilegierungsniveau gegeben wird. Ring-1 und Ring-2 stellen die Zwischenringe mit abnehmenden Privilegierungsniveaus dar.

[0022] Fig. 1A ist ein Schema, das ein Ausführungsbeispiel einer logischen Betriebsarchitektur **50** der IsoX™-Architektur veranschaulicht. Die logische Betriebsarchitektur **50** ist eine Abstraktion von Komponenten des Betriebssystems und Prozessors. Die logische Betriebsarchitektur **50** enthält Ring-0 **10**, Ring-1 **20**, Ring 2- **30**, Ring-3 **40** und einen Prozessor-Nub-Lader **52**. Jeder Ring in der logischen Betriebsarchitektur **50** kann in entweder (i) einem normalen Ausführungsmodus oder (ii) einem IsoX-Modus arbeiten. Der Prozessor-Nub-Lader **52** ist eine Beispiel eines Prozessor-Exekutive(PE)-Handlers.

[0023] Ring-0 **10** enthält zwei Abschnitte: einen Normale-Ausführung-Ring-0 **11** und einen Isolierte-Ausführung-Ring-0 **15**. Der Normale-Ausführung-Ring-0 **11** enthält Softwaremodule, die für das Betriebssystem kritisch sind und gewöhnlich als das ”Kernel” bezeichnet werden. Diese Softwaremodule enthalten ein primäres Betriebssystem **12** (z. B. Kernel), Softwaretreiber **13** und Hardwaretreiber **14**. Der Isolierte-Ausführung-Ring-0 **15** enthält einen Betriebssystem(OS)-Nub **16** und einen Prozessor-Nub **18**, wie unten beschrieben werden wird. Der OS-Nub **16** und der Prozessor-Nub **18** sind Instanzen einer OS-Exekutive (OSE) bzw. Prozessor-Exekutive (PE). Die OSE und die PE sind Teil der Exekutiveentitäten, die in einer geschützten Umgebung in Zuordnung zu dem isolierten Bereich **70** und dem IsoX-Modus arbeiten. Der Prozessor-Nub-Lader **52** ist ein Bootstrap-Ladercode, der für das Laden des Prozessorhubs **18** aus dem Prozessor oder Chipsatz in einen isolierten Bereich verantwortlich ist, wie weiter unten erläutert werden wird.

[0024] In ähnlicher Weise enthalten Ring-1 **20**, Ring-2 **30** und Ring-3 **40** einen Normale-Ausführung-Ring-1 **21**, -Ring-2 **31** bzw. -Ring-3 **41** und einen Isolierte-Ausführung-Ring-1 **25**, -Ring-2 **35** bzw. -Ring-3 **45**. Insbesondere enthält der Normale-Ausführung-Ring-3 N Anwendungen $\mathbf{42_1}$–$\mathbf{42_N}$, und der Isolierte-Ausführung-Ring-3 enthält M Applets $\mathbf{46_1}$–$\mathbf{46_M}$ (wobei ”N” und ”M” positive ganze Zahle sind).

[0025] Ein Konzept der IsoX™-Architektur ist die Bildung eines isolierten Gebiets im Systemspeicher, welches durch Komponenten der Plattform (z. B. dem Prozessor und Chipsatz) geschützt ist. Dieses isolierte Gebiet, das hier als ”isolierter Bereich” bezeichnet wird, kann auch im Cache-Speicher sein, der von einer Übersetzungsnachschlage(TLB)-Zugriffsprüfung geschützt wird. Zugriff auf diesen isolierten Bereich wird nur von einem Frontseitenbus (FSB) des Prozessors unter Verwendung spezieller Buszyklen (bezeichnet als ”isolierte Lese- und Schreibzyklen”), die von dem im IsoX-Modus ausführenden Prozessor ausgegeben werden, gewährt.

[0026] Es ist beabsichtigt, daß der ausschließlichen Unterstützung spezieller Zyklen während der Fernattestierung (bezeichnet als ”Attestierungszyklen”) zugeordnete Links (Verbindungen) innerhalb der Plattform verwendet werden können. Diese Attestierungszyklen können auf den isolierten Lese- und Schreibzyklen basieren oder von den isolierten Lese- und Schreibzyklen unabhängig sein. Anstelle spezieller Links können gemeinsam genutzte Links innerhalb der Plattform zum Unterstützen einer Fernattestierung verwendet werden. Beispiele für diese gemeinsam benutzten Links umfassen einen Peripheriekomponentenverbindungs(PCI)-Bus, einen Beschleuniger-Graphik-Port(AGP)-Bus, einen Industrie-Standard-Architektur(ISA)-Bus, einen universellen seriellen Bus (USB) u. dgl. Die Attestierzyklen werden ausgegeben, um die Lokalität zu beweisen, nämlich, daß ein Bauelement oder Gerät mit dem Verschlüsselungsmaterial und einer signierenden Maschine auf Informationen zugreift (z. B. ein Prüfprotokoll), die in dem geschützten Speicher innerhalb der Plattform gespeichert sind. Dies vermindert die Gefahr einer Software, die z. B. die Gewinnung des Prüfprotokolls simuliert.

[0027] Der IsoX-Modus wird unter Verwendung einer privilegierten Anweisung im Prozessor, kombiniert mit dem Prozessor-Nub-Lader **52** initiiert. Der Prozessor-Nub-Lader **52** verifiziert und lädt ein Ring-0-Nub-Softwaremodul (z. B. Prozessor-Nub **18**) in den isolierten Bereich. Aus Sicherheitsgründen ist der Prozessor-Nub-Lader **52** nicht modifizierbar, eingriffssicher und nicht austauschbar. Bei einem Ausführungsbeispiel ist der Prozessor-Nub-Lader **52** in einem Nur-Lese-Speicher (ROM) implementiert.

[0028] Eine Aufgabe des Prozessor-Nubs **18** ist das Verifizieren und Laden des Ring-0 OS-Nubs **16** in den isolierten Bereich. Der OS-Nub **16** stellt Links zu Diensten in dem primären Betriebssystem **12** (z. B. den ungeschützten Segmenten des Betriebssystems) zur Verfügung, liefert ein Seitenmanagement innerhalb des isolierten Bereichs und trägt die Verantwortung für das Laden von Ring-3-Anwendungsmodulen **45**, einschließlich Applets $\mathbf{46_1}$ bis $\mathbf{46_M}$, in geschützte Seiten, die in dem isolierten Bereich zugeteilt sind. Der OS-Nub **16** kann auch das Paging von Daten zwischen dem isolierten Bereich und gewöhnlichem (z. B. nicht-isoliertem) Speicher unterstützen. Ist dies der Fall, ist der OB-Nub **16** auch für die Integrität und Vertrauenswürdigkeit der Seiten des isolierten Bereichs vor der Ausräumung der Seite in den gewöhnlichen Speicher und für das Prüfen der Seiteninhalte bei Wiederherstellung der Seite verant-

wortlich.

[0029] Im folgenden wird auf Fig. 1B Bezug genommen, in der ein Schema der dem Betriebssystem **10** und dem Prozessor zugeordneten veranschaulichenden Elemente bei einem Ausführungsbeispiel der Erfindung gezeigt ist. Aus Darstellungsgründen sind nur Elemente des Rings-0 **10** und Rings-3 **40** gezeigt. Die verschiedenen Elemente in der logischen Betriebsarchitektur **50** greifen auf einen zugreifbaren physikalischen Speicher **60** entsprechend ihrer Ringhierarchie und dem Ausführungsmodus zu.

[0030] Der zugreifbare physikalische Speicher **60** umfaßt einen isolierten Bereich **70** und einen nicht-isolierten Bereich **80**. Der isolierte Bereich **70** enthält Applet-Seiten **72** und Nub-Seiten **74**. Der nicht-isolierte Bereich **80** enthält Anwendungsseiten **82** und Betriebssystemseiten **84**. Der isolierte Bereich **70** ist nur für Komponenten des Betriebssystems und des im IsoX-Modus arbeitenden Prozessors zugänglich. Der nicht-isolierte Bereich **80** ist für alle Elemente des Ring-0-Betriebssystems und Prozessors zugänglich.

[0031] Der Normale-Ausführung-Ring-0 **11**, der das primäre OS **12**, die Softwaretreiber **13** und die Hardwaretreiber **14** enthält, kann sowohl auf die OS-Seiten **84** als auch die Anwendungsseiten **82** zugreifen. Der Normale-Ausführung-Ring-3, der Anwendungen $\mathbf{42_1}$–$\mathbf{42_N}$ enthält, kann nur auf die Anwendungsseiten **82** zugreifen. Sowohl der Normale-Ausführung-Ring-0 **11** als auch -Ring-3 **41** können jedoch nicht auf den isolierten Bereich **70** zugreifen.

[0032] Der Isolierte-Ausführung-Ring-0 **15**, einschließlich des OS-Nubs **16** und des Prozessor-Nubs **18**, kann sowohl auf den isolierten Bereich **70**, einschließlich der Applet-Seiten **72** und der Nub-Seiten **74**, als auch auf den nicht-isolierten Bereich **80**, einschließlich der Anwendungsseiten **82** und der OS-Seiten **84**, zugreifen. Der Isolierte-Ausführung-Ring-3 **45**, einschließlich der Applets $\mathbf{46_1}$ bis $\mathbf{46_M}$, kann nur auf die Anwendungsseiten **82** und die Applet-Seiten **72** zugreifen. Die Applets $\mathbf{46_1}$ bis $\mathbf{46_M}$ liegen im isolierten Bereich **70**.

[0033] Im folgenden wird auf Fig. 1C Bezug genommen, in der ein Blockdarstellung eines veranschaulichenden Ausführungsbeispiels einer Plattform gezeigt ist, die die vorliegende Erfindung benutzt. Bei diesem Ausführungsbeispiel enthält die Plattform **100** einen Prozessor **110**, einen Chipsatz **120**, einen Systemspeicher **140** und periphere Komponenten (z. B. Tokens **180**/**182**, gekoppelt mit einer Token-Verbindung **185** und/oder einem Tokenleser **190**), die miteinander kommunizieren. Es ist ferner vorgesehen, daß die Plattform **100** optionale Komponenten, wie einen nicht-flüchtigen Speicher (z. B. Flash) **160** und zusätzliche Peripheriekomponenten enthält. Beispiele dieser zusätzlichen Peripheriekomponenten enthalten, ohne Beschränkung hierauf, ein Massenspeichergerät **170** und mindestens eine Eingabe/Ausgabe(I/O)-Gerät **175**. Der Übersichtlichkeit halber sind die speziellen Verbindungen für diese Peripheriekomponenten (z. B. PCI-Bus, AGB-Bus, ISA-Bus, USB-Bus, drahtlose Sende-Empfänger-Kombinationen u. s. w.) nicht gezeigt.

[0034] Generell stellt der Prozessor **110** eine zentrale Verarbeitungseinheit irgendeiner Architekturart dar, wie beispielsweise eine Komplexer-Befehlssatz-Computer(CISC)-Architektur, eine Reduzierter-Befehlssatz-Computer(RISC)-Architektur, eine Sehr-Langes-Befehlswort(VLIW)-Architektur oder eine Hybridarchitektur dar. Bei einem Ausführungsbeispiel enthält der Prozessor **110** mehrere logische Prozessoren. Ein ”logischer Prozessor”, manchmal bezeichnet als ein Thread, ist eine Funktionseinheit innerhalb eines physikalischen Prozessors mit einem Architekturzustand und physikalischen Ressourcen, die nach einer speziellen Aufteilungsfunktionalität zugeordnet sind. Daher enthält ein Multi-Threaded-Prozessor mehrere logische Prozessoren. Der Prozessor **110** ist mit dem Intel-Architektur(IA)-Prozessor, z. B. einem Prozessor einer Pentium®-Serie, dem IA-32™ und IA-64™ kompatibel. Es ist für Fachleute klar, daß die grundsätzliche Beschreibung und Operation des Prozessors **110** sowohl für eine Einzelprozessorplattform als auch für eine Multiprozessorplattform gilt.

[0035] Der Prozessor **110** kann in einem normalen Ausführungsmodus oder einem IsoX-Modus arbeiten. Insbesondere stellt eine Isolierte-Ausführung-Schaltung **115** einen Mechanismus zur Verfügung, um den Prozessor **110** das Arbeiten in einem IsoX Modus zu ermöglichen. Die Isolierte-Ausführung-Schaltung **115** stellt Hardware- und Software-Unterstützung für den IsoX-Modus bereit. Diese Unterstützung umfaßt eine Konfiguration für isolierte Ausführung, Definition des isolierten Bereichs, Definition (z. B. Decodierung und Ausführung) von Isoliert-Befehlen, Erzeugen von isolierten Bus-Zugriffszyklen und Erzeugung von Isolierter-Modus-Interrupts. Bei einem Ausführungsbeispiel, das in Fig. 3 gezeigt ist, kann die RAU als Teil des Prozessors **110** implementiert sein.

[0036] Wie in Fig. 1C gezeigt ist, ist eine Host-Verbindung **116** ein Frontseitenbus, der Schnittstellensignale zur Verfügung stellt, um dem Prozessor **110** die Kommunikation mit anderen Prozessoren oder dem Chipsatz **120** zu ermöglichen. Zusätzlich zum Normalmodus unterstützt die Host-Verbindung **116** einen Isolierter-Zugriff-Verbindungs-Modus mit entsprechenden Schnittstellensignalen für isolierte Lese- und Schreibzyklen, wenn der Prozessor **110** im IsoX Modus konfiguriert ist. Der Isolierter-Zugriff-Verbindungs-Modus wird bei Speicherzugriffen ange-

legt, die initiiert werden, während sich der Prozessor **110** im IsoX-Modus befindet, wenn die physikalische Adresse in den Adreßraum des isolierten Bereichs fällt. Der Isolierter-Zugriff-Verbindungs-Modus wird auch aktiviert bei Befehls-Vorabruf- und Cache-Rückschreib-Zyklen, wenn die Adresse innerhalb des Adreßbereichs des isolierten Bereichs liegt. Der Prozessor **110** antwortet auf Snoop-Zyklen an eine Cacheadresse innerhalb des Adreßbereichs des isolierten Bereichs, wenn der isolierte Zugriffsbuszyklus angelegt ist.

[0037] Hier enthält der Chipsatz **120** ein Speicher-Steuer-Hub (MCH) **130** und ein Eingabe/Ausgabe-Steuer-Hub (ICH) **150**, das weiter unten beschrieben ist. Das MCH **130** und das ICH **150** können in dasselbe Chip integriert sein oder auf zusammenarbeitenden separaten Chips angeordnet sein. Bei einem anderen Ausführungsbeispiel, das in Fig. 4 gezeigt ist, kann die RAU als Teil des Chipsatzes **120** implementiert sein.

[0038] Mit Bezug auf den Chipsatz **120** stellt ein MCH **130** Steuerung und Konfiguration von Speicher und Eingabe/Ausgabe-Geräten, wie dem Systemspeicher **140** und dem ICH **150** zur Verfügung. Das MCH **130** stellt Schnittstellenschaltungen zur Verfügung, um Attestierzyklen und/oder isolierte Speicher-Lese- und Schreibzyklen zu erkennen und zu bedienen. Außerdem hat das MCH **130** Speicherbereichsregister (z. B. Basis- und Längen-Register) zum Darstellen des isolierten Bereichs in dem Systemspeicher **140**. Sobald es konfiguriert ist, bricht das MCH **130** jeden Zugriff auf den isolierten Bereich ab, wenn der Isolierter-Zugriff-Verbindungs-Modus nicht angelegt ist.

[0039] Der Systemspeicher **140** speichert Code und Daten. Der Systemspeicher **140** ist typischerweise als dynamischer Speicher mit wahlfreiem Zugriff (DRAN) oder statischer Speicher mit wahlfreiem Zugriff (SRAM) implementiert. Der Systemspeicher **140** enthält den zugreifbaren physikalischen Speicher **60** (gezeigt in Fig. 1B). Der zugreifbare physikalische Speicher **60** enthält den isolierten Bereich **70** und den nicht-isolierten Bereich **80**, wie in Fig. 1B gezeigt. Der isolierte Bereich **70** ist der Speicherbereich, der vom Prozessor **110** beim Betrieb im IsoX-Modus definiert wird. Der Zugriff auf den isolierten Bereich **70** ist beschränkt und wird vom Prozessor **110** und/oder dem Chipset **120** erzwungen, der die Isolierter-Bereich-Funktionalität integriert. Der nicht-isolierte Bereich **80** enthält ein geladenes Betriebssystem (OS). Das geladene OS **142** ist der Abschnitt des Betriebssystems, der typischerweise von dem Massenspeichergerät **170** über einen Anfangsladecode in einem Bootspeicher (z. B. einem Boot-Nur-Lese-Speicher (ROM)) geladen wird. Selbstverständlich kann der Systemspeicher **140** auch weitere Programme oder Daten enthalten, die hier nicht gezeigt sind.

[0040] Wie in Fig. 1C gezeigt ist, unterstützt das ICH **150** die isolierte Ausführung zusätzlich zu herkömmlichen I/O-Funktionen. Bei diesem Ausführungsbeispiel enthält das ICH **150** wenigstens den Prozessor-Nub-Lader **52** (gezeigt in Fig. 1A), einen Hardware-geschützten Speicher **152**, einen logischen Verarbeitungsmanager **154** für isolierte Ausführung und eine Token-Link-Schnittstelle **158**. Der Klarheit halber ist nur ein ICH **150** gezeigt, obwohl Plattform **100** mit mehreren ICHS implementiert sein kann. Wenn es mehrere ICHs gibt, wird ein bestimmtes ICH ausgewählt, um die Konfiguration und den Status des isolierten Bereichs zu steuern. Diese Auswahl kann von einem externen Strapping-Pin ausgeführt werden. Wie dem Fachmann bekannt ist, können andere Auswahlmethoden verwendet werden.

[0041] Der Prozessor-Nub-Lader **52**, wie er in den Fig. 1A und Fig. 1C gezeigt ist, weist einen Prozessor-Nub-Ladercode und seinen Hash-Wert (oder Digest) auf. Nach dem Aufruf durch Ausführung eines geeigneten Isoliert-Befehls (z. B. ISO_INIT) durch den Prozessor **110** wird der Prozessor-Nub-Lader **52** in den isolierten Bereich **70** übertragen. Danach kopiert der Prozessor-Nub-Lader **52** das Prozessor-Nub **18** aus dem nicht-flüchtigen Speicher **160** in den isolierten Bereich **70**, verifiziert und bringt eine Darstellung des Prozessor-Nubs **18** (z. B. einen Hash-Wert) in den geschützten Speicher **152** ein. Hier ist der geschützte Speicher **152** als ein Speicher-Array mit Einzelschreib- und Mehrfachlese-Befähigung implementiert. Diese nicht-modifizierbare Befähigung wird logikgesteuert oder ist Teil der eigenen Natur des Speichers selbst. Wie gezeigt, kann der geschützte Speicher **152** beispielsweise eine Mehrzahl von Einzelschreib-, Mehrfachleseregistern enthalten.

[0042] Wie in den Fig. 1C und Fig. 2 gezeigt ist, ist der geschützte Speicher **152** so konfiguriert, daß er ein Prüfprotokoll **156** unterstützt. Ein "Prüfprotokoll" **156** enthält Informationen betreffend die Betriebsumgebung der Plattform **100**; nämlich eine Liste von Daten, die darstellt, welche Informationen erfolgreich in den Systemspeicher **140** geladen wurden, und zwar nach dem Einschalten der Plattform **100**. Beispielsweise können die repräsentativen Daten Hash-Werte jedes in den Systemspeicher **140** geladenen Softwaremoduls sein. Diese Softwaremodule können den Prozessor-Nub **18**, den OS-Nub **16** und/oder irgendwelche anderen kritischen Softwaremodule (z. B. Ring-0-Module sein), die in den isolierten Bereich **70** geladen worden sind. Daher kann das Prüfprotokoll **156** als ein Fingerabdruck wirken, der in die Plattform geladene Informationen identifiziert (z. B. der Ring-0-Code, der die Konfiguration und Operation der isolierten Ausführung steuert), und es dient zum Attestieren oder Belegen des Zustandes der derzeitigen isolierten Ausführung.

[0043] Bei einem anderen Ausführungsbeispiel können sowohl der geschützte Speicher **152** als auch der ungeschützte Speicher (z. B. ein Speicherarray in dem nicht-isolierten Bereich **80** des Systemspeichers **140** der Fig. 1C) gemeinsam ein geschütztes Prüfprotokoll **156** zur Verfügung stellen. Das Prüfprotokoll **156** wird im Speicherarray gespeichert, während Informationen betreffend den Zustand des Prüfprotokolls **156** (z. B. ein Gesamt-Hash-Wert für die repräsentativen Daten innerhalb des Prüfprotokolls **156**) in dem geschützten Speicher **152** gespeichert werden.

[0044] Es wird wiederum auf Fig. 1C Bezug genommen; der nicht-flüchtige Speicher **160** speichert nicht-flüchtige Informationen. Typischerweise ist der nicht-flüchtige Speicher **160** in Flash-Speicher implementiert. Der nicht-flüchtige Speicher **160** enthält den Prozessor-Nub **18**, wie oben beschrieben wurde. Zusätzlich kann der Prozessor-Nub **18** auch Anwendungsprogrammschnittstellen(API)-Abstraktionen an Low-Level-Sicherheitsdienste liefern, die von anderer Hardware zur Verfügung gestellt werden, und der kann von dem ursprünglichen Ausrüstungshersteller (OEM) oder Betriebssystemkäufer (OSV) über eine Boot-Diskette verteilt werden.

[0045] Das Massenspeichergerät **170** speichert Archivinformationen, wie z. B. Code (z. B. Prozessor-Nub **18**), Programme, Dateien, Daten, Anwendungen (z. B. Anwendungen $\mathbf{42_1}$–$\mathbf{42_N}$), Applets (z. B. Applets $\mathbf{46_1}$ bis $\mathbf{46_M}$) und Betriebssysteme. Das Massenspeichergeräte **170** kann eine Compactdisk(CD)-ROM **172**, einen Plattenspeicher **176** oder irgendein anderes magnetisches oder optisches Speichergerät enthalten. Das Massenspeichergerät **170** stellt auch einen Mechanismus zum Lesen von plattform-lesbaren Medien zur Verfügung. Bei Implementierung in Software werden die Elemente der vorliegenden Erfindung in einem prozessor-lesbaren Medium gespeichert. Das ”prozessor-lesbare Medium” kann irgendein Medium umfassen, welches Informationen zu speichern oder zu übertragen vermag. Beispiele des prozessor-lesbaren Mediums schließen eine elektronische Schaltung, ein Halbleiterspeicherbauelement, einen Nur-Lese-Speicher (ROM), einen Flash-Speicher, einen löschbaren programmierbaren ROM (EPROM), ein Lichtleitermedium, eine Hochfrequenz(HF)-Verbindung und beliebige plattform-lesbare Medien, wie eine Diskette, eine CD-ROM, eine optische Platte, ein Plattenspeicher u. s. w., ein.

[0046] Bei der Kommunikation mit der Plattform **100** enthalten I/O-Geräte **175** stationäre oder tragbare Benutzer-Eingabegeräte, von denen jedes eine oder mehrere I/O-Funktionen ausführt. Beispiele eines stationären Benutzer-Eingabegeräts umfassen eine Tastatur, ein Keypad, eine Maus, einen Trackball, ein Touch-Pad und einen Stift. Beispiele für ein tragbares Benutzer-Eingabegerät umfassen einen Handapparat, einen Pieper, einen Hand-held (z. B. PDA) oder irgendein drahtloses Gerät. Die I/O-Geräte **175** ermöglichen die Fernattestierung der Plattform **100**, die weiter unten beschrieben wird.

[0047] Die Token-Verbindung **185** bildet ein Schnittstelle zwischen dem ICH **150** und einem festen Token **180** (z. B. einem Mutterplatinen-Token) und/oder einem Token-Leser **190** in Verbindung mit einem entnehmbaren Token **182**, das Charakteristiken ähnlich einer Smart Card hat. Generell sind beiden Arten von Token Geräte, welche vorgegebene I/O-Funktionen ausführen. Bei Ausführungsbeispielen, gezeigt in Fig. 6 und Fig. 7, enthalten Token **180** und/oder **182** Verschlüsselungsmaterial (z. B. spezielle kryptographische Identifizierer, wie ein öffentliches/privates Schlüsselpaar) und die Funktionsfähigkeit, das Prüfprotokoll (oder dessen Darstellung) mit dem privaten Schlüssel oder Schlüsselpaar zu signieren. Die Token-Verbindungs-Schnittstelle **158** in dem ICH **150** bildet eine logische Kopplung zwischen der Token-Verbindung **185** und dem ICH **150** und unterstützt eine Fernattestierung zur Wiederherstellung des Inhalts des Prüfprotokoll **156**.

II. Erzeugen und Benutzen eines geschützten Prüfprotokoll

[0048] Im folgenden wird auf Fig. 2 Bezug genommen, in der ein Ablaufdiagramm der veranschaulichenden Operationen der Plattform zum Erzeugen eines Ausführungsbeispiels des geschützten Prüfprotokolls (audit log) gezeigt ist. Nach dem Einschalten der Plattform werden Informationssegmente in den Systemspeicher zum Verarbeiten durch einen Prozessor geladen (Block **200**). Beispiele für diese Informationssegmente umfassen das Prozessor-Nub und das OS-Nub. Gleichzeitig mit dem Laden der Informationssegmente in den Systemspeicher werden Kopien jedes Informationssegments einer kryptographischen Hash-Operation unterzogen, um einen Hash-Wert der Segmente zu erzeugen. Diese Hash-Werte bilden ein im geschützten Speicher gespeichertes Prüfprotokoll (Blöcke **205** und **210**). Bei einem Ausführungsbeispiel, das in Fig. 1C gezeigt ist, wird der geschützte Speicher innerhalb des ICH implementiert. Der Speicher wird als ”geschützt” angesehen, wenn der Inhalt des Speichers lesbar und nicht-modifizierbar ist, wie oben beschrieben wurde. Wenn nachfolgende Informationssegmente zum Speichern in das Prüfprotokoll ausgewählt werden, werden ihre Hash-Werte an das Prüfprotokoll hinter die zuvor berechneten Hash-Werte angehängt (Block **215**). Es ist vorgesehen, daß nur Hash-Werte von ausgewählten Nubs in dem Prüfprotokoll gespeichert werden können.

III. Fernattestierung

A. Beginn der Fernattestierung

[0049] Bei einem Ausführungsbeispiel wird die Fernattestierung durch Ausgabe einer Attestierungsanforderung initiiert. Die Attestierungsanforderung kann von einer fernen Quelle oder von einem Teilnehmer am Ort der Plattform ausgehen, der als Vertretung für die ferne Quelle tätig ist oder nicht. Normalerweise enthält die Attestierungsanforderung eine Primärabfrage und/oder mindestens eine optionale Sekundärabfrage. Jede Abfrage bewirkt die Ausgabe der Attestierungszyklen, die zur Gewinnung des Inhalts des Prüfprotokolls konzipiert sind. Zumindest können die Inhalte des Prüfprotokolls zum Verifizieren der Integrität des IsoX™-Prozessor und des OS-Nub der Plattform verwendet werden. Die sekundäre Abfrage gewinnt zusätzlich zu dem Prüfprotokoll einen Hash-Wert eines ausgewählten IsoX-Applets, das von der Plattform geladen ist, um die Integrität des Applets zu verifizieren. Der Hash-Wert des Applets wird im laufenden Betrieb (on-the-fly) vom OS-Nub generiert. Dies vermeidet die Notwendigkeit der Speicherung jedes und aller geladenen Applets in dem Prüfprotokoll. Bei primären Abfragen erzeugt die RAU eine Nachricht, die das Prüfprotokoll, eine digitale Signatur, die das Prüfprotokoll abdeckt, und mindestens ein digitales Zertifikat für das RAU-Verschlüsselungsmaterial enthalten kann, und leitet die Nachricht zum Anfordernden zurück. Bei sekundären Abfragen erzeugt die RAU eine Nachricht, die das Applet-Hash, das Prüfprotokoll, eine an das Applet-Hash und das Prüfprotokoll abdeckende digitale Signatur und mindestens ein digitales Zertifikat für das RAU-Verschlüsselungsmaterial enthalten kann, und leitet die Nachricht zum Anfordernden zurück, um verschiedene oben angegebene Informationen wiederzugewinnen.

B. Prozessorintegrierte RAU

[0050] Gemäß Fig. 3, auf die im folgenden Bezug genommen wird, ist die RAU (Remote Attestation Unit; Fernattestiereinheit) **300** in den Prozessor **110** integriert. Der Prozessor **110** führt lokalen Code aus. Bei Feststellung einer Attestierungsanforderung entwickelt der Prozessor **110** einen Kommunikationsweg zu einer für die Speicherung des Prüfprotokoll **156** verantwortlichen Komponente **310**. Genauer gesagt, führt bei einem Ausführungsbeispiel der lokale Code einen physikalischen Befehl in Abhängigkeit einer Attestierungsanforderung aus. Der physikalische Befehl bewirkt bei Ausführung durch den Prozessor **110** die Ausgabe von Attestierungszyklen durch den Prozessor **110** zum Lesen des Inhalts des Prüfprotokolls **156**.

[0051] Um dies zu veranschaulichen, die Komponente **310** könnte das ICH **150** der Fig. 1C sein, obwohl andere Komponenten innerhalb der Plattform **100** verwendet werden können. Die Kommunikation zwischen dem Prozessor **110** und der Komponente **310** erfolgt über eine oder mehrere Verbindungen, wie z. B. eine erste Verbindung **310** und eine zweite Verbindung **320**. Diese Verbindung **310** und **320** können als spezielle Verbindungen zum Verarbeiten von Attestierungszyklen oder gemeinsam genutzte Verbindungen (z. B. Host-Verbindung, PCI-Bus u. s. w.), die zur Verarbeitung der Attestierungszyklen erweitert sind, konfiguriert sein. Diese Attestierungszyklen signalisieren der Komponente **310**, ein Lesen des Prüfprotokolls **156** zu akzeptieren.

[0052] Bei Empfang des Prüfprotokolls **156** erzeugt die RAU **300** im Prozessor **110** eine digitale Signatur **330** durch digitales Signieren des Prüfprotokolls **156** mit dem Verschlüsselungsmaterial **340** (z. B. einem im voraus gespeicherten privaten Schlüssel). Das Prüfprotokoll **156**, die digitale Signatur **330** und möglicherweise digitale Zertifikate aus dem RAU-Verschlüsselungsmaterial werden paketisiert und als eine Nachricht von der RAU **300** zu dem Anforderer oder zu einem Bereich **350** gesendet, der für den lokalen Code zugreifbar ist.

[0053] Selbstverständlich wird angenommen, daß dann, wenn das Prüfprotokoll **156** in ungeschütztem Speicher gespeichert ist, das ICH **150** eine (nicht gezeigte) Komponente enthalten kann, um zu verifizieren, daß der Inhalt des Prüfprotokolls **156** vor Ausgabe des Prüfprotokolls **156** an den Prozessor **110** nicht modifiziert worden ist. Dies kann von der Komponente **310** dadurch erreicht werden, daß ein Hash-Wert des Prüfprotokolls **156**, das aus ungeschütztem Speicher wiederhergestellt ist, erzeugt und mit dem Gesamthashwert, der im geschützten Speicher gespeichert ist, verglichen wird.

[0054] Als eine optionale Ausführungsform kann der Benutzer zu steuern wünschen, wann das Verschlüsselungsmaterial **340** verwendet wird. Beispielsweise kann die Plattform eine Anforderungsnachricht über eine Kommunikationseinrichtung **360** an eine Benutzer-opt-in-Einrichtung **380** über einen geschützten Kommunikationspfad ausgegeben. Bei einem Ausführungsbeispiel ist die Kommunikationseinheit **360** mit dem Token-Bus **185** gekoppelt und wird mit einem drahtlosen Empfänger **385** und einem drahtlosen Sender **370** (gemeinsam hier als ”drahtloser Sendeempfänger” bezeichnet) benutzt. Der drahtlose Empfänger und Sender **365** und **370** werden zur Herstellung und Aufrechterhaltung einer direkten Kommunikation mit der Benutzer-opt-in-Einrichtung **380** verwendet. Selbstverständlich kann die Benutzer-opt-in-Einrichtung **380** mit der Kommunikationseinrichtung **360** über irgendeine Art einer Verbindung gekoppelt sein.

[0055] Bei Empfang der Anforderungsnachricht gibt

die Kommunikationseinrichtung **360** eine Nachricht an die Benutzer-opt-in-Einrichtung **380** aus, welche dem Benutzer ermöglicht, seinen Wunsch auf Freigabe des Verschlüsselungsmaterials **340** zur Erzeugung der digitalen Signatur **330** zu bestätigen. Auf der Basis einer Eingabe durch den Benutzer oder deren Fehlen (z. B. Drücken einer der Benutzer-opt-in-Einrichtung **380** zugeordneten Taste, fehlende Betätigung durch den Benutzer u. s. w.) wird eine Antwortnachricht an die Kommunikationseinrichtung **360** zurückgeschickt, welche den Inhalt der Antwortnachricht an die RAU **300** über einen geschützten Kommunikationspfad weiterleitet. Bei Empfang der Antwortnachricht fährt die RAU **300** fort mit der Generierung der digitalen Signatur **330** und/oder der digitalen Zertifikate für das RAU-Verschlüsselungsmaterial und die Eingabe in den Bereich **350**, der für den lokalen Code zugreifbar ist, wenn die Benutzung des Verschlüsselungsmaterials **340** vom Benutzer autorisiert worden ist.

C. Chipsatz-Integrierte RAU

[0056] Im folgenden wird auf Fig. 4 Bezug genommen, gemäß der die RAU **300** in ein Kernlogikbauelement **400** integriert ist. Wie gezeigt ist, führt der Prozessor **110** lokalen Code aus. Bei Feststellung einer Attestierungsanforderung richtet das Kernlogikbauelement **400** einen Kommunikationspfad zu einer Komponente **420**, die für die Speicherung des Prüfprotokolls **156** verantwortlich ist. Insbesondere sendet bei einem Ausführungsbeispiel der lokale Code eine Nachricht an das Kernlogikbauelement **400** auf der Basis einer Attestierungsanforderung. Die Nachricht veranlaßt das Kernlogikbauelement **400**, Attestierungszyklen zum Lesen des Inhalts der Prüfprotokoll **156** auszugeben.

[0057] Beispielsweise leitet das Kernlogikbauelement **400** in Erwiderung der Attestierungsanforderung die Attestierungszyklen zur Komponente **420** über die Verbindung **430**, um ein Lesen des Inhalts des gespeicherten Prüfprotokolls **156** zu ermöglichen. Die Verbindung **430** kann dem Unterstützen der Fernattestierung speziell gewidmet sein oder mehrere Funktionen, einschließlich der Attestierungszyklen, die von dem Kernlogikbauelement **400** erzeugt werden, unterstützen. Bei Empfang des Inhalts des gespeicherten Prüfprotokolls **156** erzeugt das Kernlogikbauelement **400**, welches die RAU **300** enthält, eine digitale Signatur **300** für das Prüfprotokoll **156** (wie oben beschrieben) und schreibt die digitale Signatur **300** in einen für den lokalen Code zugreifbaren Bereich.

[0058] Wie jedoch in Fig. 5 gezeigt ist, werden dann, wenn das Kernlogikbauelement **400** auch das Prüfprotokoll **156** enthält, interne Signale **450** innerhalb des Kernlogikbauelements **400** dazu verwendet, der RAU **300** den Zugriff auf das Prüfprotokoll **156** zu ermöglichen. Bei Empfang des Inhalts des Prüfprotokolls **156** erzeugt die RAU **300** des Kernlogikbauelements **400** wiederum die digitale Signatur **330** des Prüfprotokolls und möglicherweise eine oder mehrere digitale Zertifikate für das RAU-Verschlüsselungsmaterial (nicht gezeigt). Diese Informationen werden als Nachricht dem Anforderer zur Verfügung gestellt oder in den für den lokalen Code zugreifbaren Bereich geschrieben.

[0059] Als optionale Ausführungsform könnte der Benutzer zu Kontrollieren wünschen, wann das Verschlüsselungsmaterial **340** verwendet wird. So kann die Plattform beispielsweise eine Anforderungsnachricht **470** über eine Kommunikationseinrichtung **460** an eine Benutzer-opt-in-Einrichtung **490** über einen geschützten Kommunikationspfad ausgeben. Bei einem Ausführungsbeispiel ist die Kommunikationseinrichtung **460** mit dem Token-Bus **185** gekoppelt und wird als drahtloser Sendeempfänger **465** verwendet, um eine direkte Kommunikationen mit der Benutzer-opt-in-Einrichtung **490** herzustellen und aufrechtzuerhalten.

[0060] In Erwiderung des Empfangs der Anforderungsnachricht **470** gibt die Kommunikationseinrichtung **460** eine Nachricht an die Benutzer-opt-in-Einrichtung **490** aus, die den Benutzer veranlaßt, seinen Wunsch auf Freigabe des Verschlüsselungsmaterials **340** zur Erzeugung der digitalen Signatur **330** zu bestätigen. Auf der Basis einer Benutzereingabe oder deren Fehlen (z. B. Betätigen einer der Benutzer-opt-in-Einrichtung **490** zugeordneten Taste, Untätigkeit des Benutzers u. s. w.) wird eine Antwortnachricht **480** an die Kommunikationseinrichtung **460** zurückgegeben, welche den Inhalt der Antwortnachricht **480** an die RAU **300** des Kernlogikbauelements **400** über einen geschützten Kommunikationspfad weiterleitet. Bei Empfang der Antwortnachricht **480** fährt die RAU **300** mit der Erzeugung der digitalen Signatur **330** und möglicherweise digitaler Zertifikate in der oben beschriebenen Weise fort und plaziert sie in den für den lokalen Code zugänglichen Bereich, wenn die Benutzung des Verschlüsselungsmaterials **340** vom Benutzer autorisiert worden ist.

D. In ein festes Token integrierte RAU

[0061] Im folgenden wird auf Fig. 6 Bezug genommen. Wenn die RAU **300** in dem festen Token **180** integriert ist, kommuniziert das feste Token **180** mit einer Komponente (z. B. ICH **150**), die das Prüfprotokoll **156** hält, über die Token-Verbindung **185**. Die Funktion der Token-Verbindung **185** kann erweitert werden auf die Unterstützung von Attestierungszyklen, die nur dann vom festen Token **180** erzeugt werden, wenn eine Fernattestierung angefordert ist. Diese Attestierungszyklen werden zum ICH **150** geleitet, um eine Akzeptanz eines Lesens zum Prüfprotokoll **156** anzufordern. Bei Empfang des Inhalts des Prüf-

protokolls **156** erzeugt die RAU **300**, die in dem festen Token **180** implementiert ist, eine digitale Signatur **330** durch digitales Signieren des Prüfprotokolls **156** mit dem in der RAU gespeicherten Verschlüsselungsmaterial **340**. Danach schreibt die RAU **300** die digitale Signatur **330** und möglicherweise digitale Zertifikate für das Verschlüsselungsmaterial **340** zum Anforderer oder in einen für den lokalen Code zugänglichen Bereich.

[0062] Als ein optionales Ausführungsbeispiel kann der Benutzer zu steuern wünschen, wann das in der RAU **300** gespeicherte Verschlüsselungsmaterial **610** verwendet wird. Beispielsweise kann der Benutzer aufgefordert werden, seinen Wunsch auf Freigabe des Verschlüsselungsmaterials **340** zur Erzeugung der digitalen Signatur **330** zu bestätigen. Die Aufforderung kann beispielsweise durch Übertragung einer Nachricht **620** über einen in dem Token **180** angeordneten drahtlosem Sendeempfänger **630** ausgeführt werden. Die Bestätigung eines Wunsches auf Freigabe des Verschlüsselungsmaterials **340** kann entweder (1) durch Übertragen einer Rücklaufnachricht (**640**) aus einer Benutzer-opt-in-Einrichtung zu dem Token **180**, wie es gezeigt ist, oder (2) durch Eingabe von Zugriffsinformation über eine Benutzer-opt-in-Einrichtung (nicht gezeigt), die mit dem Token **180** physikalisch verbunden ist, vorgenommen werden. Danach fährt die RAU **300** fort mit der Erzeugung der digitalen Signatur **330** und/oder mindestens einem digitalen Zertifikat für das Verschlüsselungsmaterial **340**. Danach werden diese Informationen zusammen mit dem Prüfprotokoll **156** zu dem Anforderer gesendet oder in den für den lokalen Code zugänglichen Bereich plaziert, wenn die Benutzung des Verschlüsselungsmaterials **340** von dem Benutzer autorisiert worden ist. Selbstverständlich können opt-in-Nachrichten **620** und **640** über das I/O-Gerät **175** geleitet werden, sofern die Nachrichten geschützt sind.

E. Entnehmbares Token mit integrierter RAU

[0063] Im folgenden wird auf Fig. 7 Bezug genommen; wenn die RAU **300** in dem entnehmbaren Token **182** integriert ist, kommuniziert das entnehmbare Token **182** mit einer Komponente (z. B. ICH **150**), die das Prüfprotokoll **156** hält, über die Token-Verbindung **185**. Die Funktionsfähigkeit der Token-Verbindung **185** kann auf eine Unterstützung von Attestierungszyklen erweitert werden, die nur dann vom Token-Leser bei Einsetzen oder Verbinden (z. B. drahtloses Token) des entnehmbaren Token **182** erzeugt werden, wenn eine Fernattestierung angefordert wird. Diese Attestierungszyklen werden vom Token-Leser an die das Prüfprotokoll **156** (z. B. ICH **150**) speichernde Hardware erzeugt, um eine Akzeptanz des Lesens des Prüfprotokolls **156** anzufordern. Bei Empfang des Inhalts des Prüfprotokolls **156** erzeugt die im entnehmbaren Token **182** implementierte RAU **300** die digitale Signatur **300** durch digitales Signieren des Prüfprotokolls **156** mit dem in der RAU **300** gespeicherten Verschlüsselungsmaterial **340**. Danach schreibt die RAU **300** die digitale Signatur **330** und/oder wenigstens ein digitales Zertifikat für das Verschlüsselungsmaterial **340** in einen für den lokalen Code zugreifbaren Bereich.

[0064] Als optionales Ausführungsbeispiel kann der Benutzer zu steuern wünschen, wann das in der RAU **300** gespeicherte Verschlüsselungsmaterial **340** verwendet wird. Beispielsweise kann der Benutzer aufgefordert werden, seinen Wunsch zur Freigabe des Verschlüsselungsmaterials **340** zum Erzeugen der digitalen Signatur **330** zu bestätigen. Die Aufforderung kann beispielsweise durch Übertragung einer Nachricht **720** über einen drahtlosen Sendeempfänger **730** erfolgen, der im Token **182** angeordnet ist. Die Bestätigung eines Wunsches zur Freigabe des Verschlüsselungsmaterials **340** kann erfolgen entweder (1) durch Senden einer Rücklaufnachricht **740** von einer (nicht gezeigten) Benutzer-opt-in-Einrichtung zum Token **182**, wie gezeigt, oder (2) durch Eingabe von Zugriffsinformationen über eine Benutzer-opt-in-Einrichtung, die mit dem Token **182** in einer nicht gezeigten Weise physikalisch verbunden ist. Danach fährt die RAU **300** fort mit der Erzeugung der digitalen Signatur **330** und/oder digitaler Zertifikate für das Verschlüsselungsmaterial **340**, der Weiterleitung über den Token-Leser **190** und dem Anordnen in den für den lokalen Code zugreifbaren Bereich, wenn die Benutzung des Verschlüsselungsmaterials **340** von dem Benutzer autorisiert worden ist. Selbstverständlich können opt-in-Nachrichten **620** und **640** über das I/O-Gerät **175** geleitet werden, sofern die Nachrichten geschützt sind.

[0065] Während die Erfindung unter Bezugnahme auf veranschaulichende Ausführungsbeispiele beschrieben worden ist, soll diese Beschreibung nicht in einem einschränkenden Sinne verstanden werden. Verschiedene Abwandlungen der veranschaulichenden Ausführungsbeispiele sowie andere Ausführungsbeispiele der Erfindung, die für Fachleute offensichtlich sind, sollen dem Wesen und Schutzumfang der Erfindung zugerechnet werden.

Patentansprüche

1. Eine Plattform, enthaltend:
einen Prozessor (**110**), der entweder in einem normalen Ausführungsmodus oder in einem isolierten Ausführungsmodus arbeitet;
einen Systemspeicher (**140**), der einen isolierten Bereich und einen nicht-isolierten Bereich enthält, wobei der Prozessor nur dann auf den isolierten Bereich zugreifen kann, wenn er in dem isolierten Ausführungsmodus arbeitet;
einen Chipsatz (**130**, **150**; **310**; **400**, **420**) mit einer Komponente (**310**; **420**; **400**; **150**), die einen ge-

schützten Speicher (**152**) aufweist, der ein Prüfprotokoll (**156**) oder, wenn das Prüfprotokoll (**156**) in einem nicht-geschützten Speicher gespeichert ist, einen Gesamt-Hash-Wert des Prüfprotokolls (**156**) speichert, wobei das Prüfprotokoll (**156**) repräsentative Daten von nach dem Einschalten in der Plattform geladenen Software-Modulen enthält;
wobei in dem Prozessor (**110**) oder in einer Komponente (**400**) des Chipsatzes oder in einem mit dem Chipsatz über einen Token-Bus (**185**) gekoppelten Token (**180**; **182**) eine Fernattestiereinheit (**300**) enthalten ist,
wobei zwischen dem die Fernattestiereinheit (**300**) enthaltenden Prozessor (**110**), der die Fernattestiereinheit (**300**) enthaltenden Komponente (**400**) bzw. dem die Fernattestiereinheit (**300**) enthaltenden Token (**180**; **182**) einerseits und der das Prüfprotokoll (**156**) oder dessen Gesamt-Hash-Wert speichernden Komponente (**310**; **420**; **400**; **150**) andererseits eine Kommunikationsverbindung errichtet ist, wobei die Kommunikationsverbindung zum Unterstützen spezieller Attestierungszyklen dient, um das Prüfprotokoll (**156**) zu lesen, wenn eine Attestierungsanforderung aus einem fernen oder lokalen Anforderer erfasst worden ist;
wobei die Fernattestiereinheit (**300**) das Prüfprotokoll (**156**) mit in der Fernattestiereinheit (**300**) gespeichertem Verschlüsselungsmaterial (**340**) digital unterzeichnet, und eine das digital unterzeichnete Prüfprotokoll enthaltende Nachricht zur Rückgabe an den Anforderer erzeugt.

2. Die Plattform nach Anspruch 1, wobei die Fernattestiereinheit (**300**) in dem Prozessor (**110**) enthalten ist.

3. Die Plattform nach Anspruch 1 oder 2, wobei die Fernattestiereinheit eine digitale Signatureinheit zum digitalen Signieren des Prüfprotokolls mit dem Verschlüsselungsmaterial enthält.

4. Die Plattform nach Anspruch 3, wobei das Verschlüsselungsmaterial innerhalb der Fernattestiereinheit einen privaten Schlüssel enthält.

5. Die Plattform nach Anspruch 2, wobei der Chipsatz enthält:
ein Speichersteuer-Hub (**130**), das über eine teilweise die Kommunikationsverbindung bildende erste Verbindung (**315**) mit dem Prozessor (**110**) gekoppelt ist; und
ein Eingabe/Ausgabe-Steuer-Hub (**150**; **310**), das über eine teilweise die Kommunikationsverbindung bildende zweite Verbindung (**320**) mit dem Speichersteuer-Hub (**130**) gekoppelt ist, wobei das Eingabe/Ausgabe-Steuer-Hub (**150**; **310**) Einzelschreib-Mehrfachlese-Speicher zum Speichern des Prüfprotokolls enthält.

6. Die Plattform nach Anspruch 5, ferner enthaltend eine Kommunikationseinrichtung (**360**), die mit dem Eingabe/Ausgabe-Steuer-Hub (**150**; **310**) gekoppelt ist und Kommunikationen mit einer Benutzer-opt-in-Einrichtung (**380**) ermöglicht.

7. Die Plattform nach Anspruch 6, wobei die Kommunikationseinrichtung (**360**) einen drahtlosen Sender und einen drahtlosen Empfänger zur Kommunikation mit der Benutzer-opt-in-Einrichtung (**380**) enthält.

8. Die Plattform nach Anspruch 6, wobei die Benutzer-opt-in-Einrichtung (**380**) es einem Benutzer ermöglicht, eine Betriebsstufe der Fernattestierung zu steuern, indem die Erzeugung der digitalen Signatur verhindert wird.

9. Die Plattform nach Anspruch 1, wobei die Fernattestieranforderung eine primäre Abfrage umfaßt.

10. Die Plattform nach Anspruch 9, wobei die Fernattestiereinheit (**300**) eine Nachricht an einen Anforderer in Erwiderung der primären Abfrage zurückschickt, wobei die Nachricht das Prüfprotokoll und wenigstens eine digitale Signatur, die das mit dem Verschlüsselungsmaterial (**340**) digital signierte Prüfprotokoll ist, enthält.

11. Die Plattform nach Anspruch 10, wobei die Nachricht ferner ein digitales Zertifikat für das Verschlüsselungsmaterial (**340**) enthält.

12. Die Plattform nach Anspruch 9, wobei die Fernattestieranforderung eine sekundäre Abfrage enthält.

13. Die Plattform nach Anspruch 12, wobei die Fernattestiereinheit (**300**) eine Nachricht an einen Anforderer in Erwiderung der zweiten Abfrage zurückschickt, wobei die Nachricht einen Hash-Wert eines ausgewählten Applets, das Prüfprotokoll und eine digitale Signatur mit dem Hash-Wert und dem Prüfprotokoll enthält.

14. Die Plattform nach Anspruch 13, wobei die Nachricht ferner ein digitales Zertifikat für das Verschlüsselungsmaterial (**340**) enthält.

15. Die Plattform nach Anspruch 1, wobei die Fernattestiereinheit (**300**) in einer Komponente (**400**) des Chipsatzes oder in einem mit dem Chipsatz über einen Token-Bus (**185**) gekoppelten Token enthalten ist.

16. Die Plattform nach Anspruch 15, ferner enthaltend den Prozessor zum Feststellen der Fernattestieranforderung von den Anforderer und zum Ausgeben von Zyklen an die Komponente (**400**) bzw. das Token (**180**; **182**), um der Komponente (**400**) bzw. dem Token (**180**; **182**) den Zugriff auf das Prüfproto-

koll (**156**) zu ermöglichen.

17. Die Plattform nach Anspruch 15, wobei die Fernattestiereinheit (**300**) in einer Komponente (**400**) des Chipsatzes enthalten ist.

18. Die Plattform nach Anspruch 16, wobei die Fernattestiereinheit (**300**) in einem mit dem Chipsatz über einen Token-Bus (**185**) gekoppelten Token (**180**; **182**) enthalten ist, wobei der Chipsatz eine mit dem Token-Bus gekoppelte Token-Verbindungs-Schnittstelle aufweist.

19. Die Plattform nach Anspruch 15, wobei das Token ein festes Token (**180**) ist, das mit dem Token-Bus (**185**) gekoppelt ist.

20. Die Plattform nach Anspruch 19, ferner enthaltend eine Benutzer-opt-in-Einrichtung, die mit dem festen Token (**180**) in Verbindung steht, wobei die Benutzer-opt-in-Einrichtung dem Benutzer das Beenden von Operationen der Fernattestiereinheit (**300**) ermöglicht.

21. Die Plattform nach Anspruch 18, ferner enthaltend einen Token-Leser (**190**), der mit dem Token-Bus (**185**) gekoppelt ist.

22. Die Plattform nach Anspruch 21, wobei das Token ein entnehmbares Token (**182**) ist, das mit dem Token-Leser (**190**) in Verbindung steht.

23. Die Plattform nach Anspruch 22, ferner enthaltend eine Benutzer-opt-in-Einrichtung, die mit dem entnehmbaren Token (**182**) in Verbindung steht, wobei die Benutzer-opt-in-Einrichtung einem Benutzer das Beenden von Operationen der Fernattestiereinheit (**300**) ermöglicht.

24. Ein Verfahren zum Fernattestieren einer Plattform, umfassend:
Speichern eines Prüfprotokolls in einem geschützten Speicher einer Komponente eines Chipsatzes einer Plattform, wobei das Prüfprotokoll eine Liste von Daten ist, die jedes aus einer Mehrzahl von IsoX-Softwaremodulen darstellen, die in die Plattform geladen sind;
Wiedergewinnen des Prüfprotokolls aus dem geschützten Speicher in Erwiderung des Empfangs einer Fernattestieranforderung von einer entfernt gelegenen Plattform.
Digitales Signieren des Prüfprotokolls zum Erzeugen einer digitalen Signatur vor dem Übertragen an die entfernt gelegene Plattform.

25. Das Verfahren nach Anspruch 24, wobei die für jeden der Mehrzahl von Softwaremodulen repräsentativen Daten als ein kryptographischer Hash-Wert ausgebildet werden.

Es folgen 8 Blatt Zeichnungen

Anhängende Zeichnungen

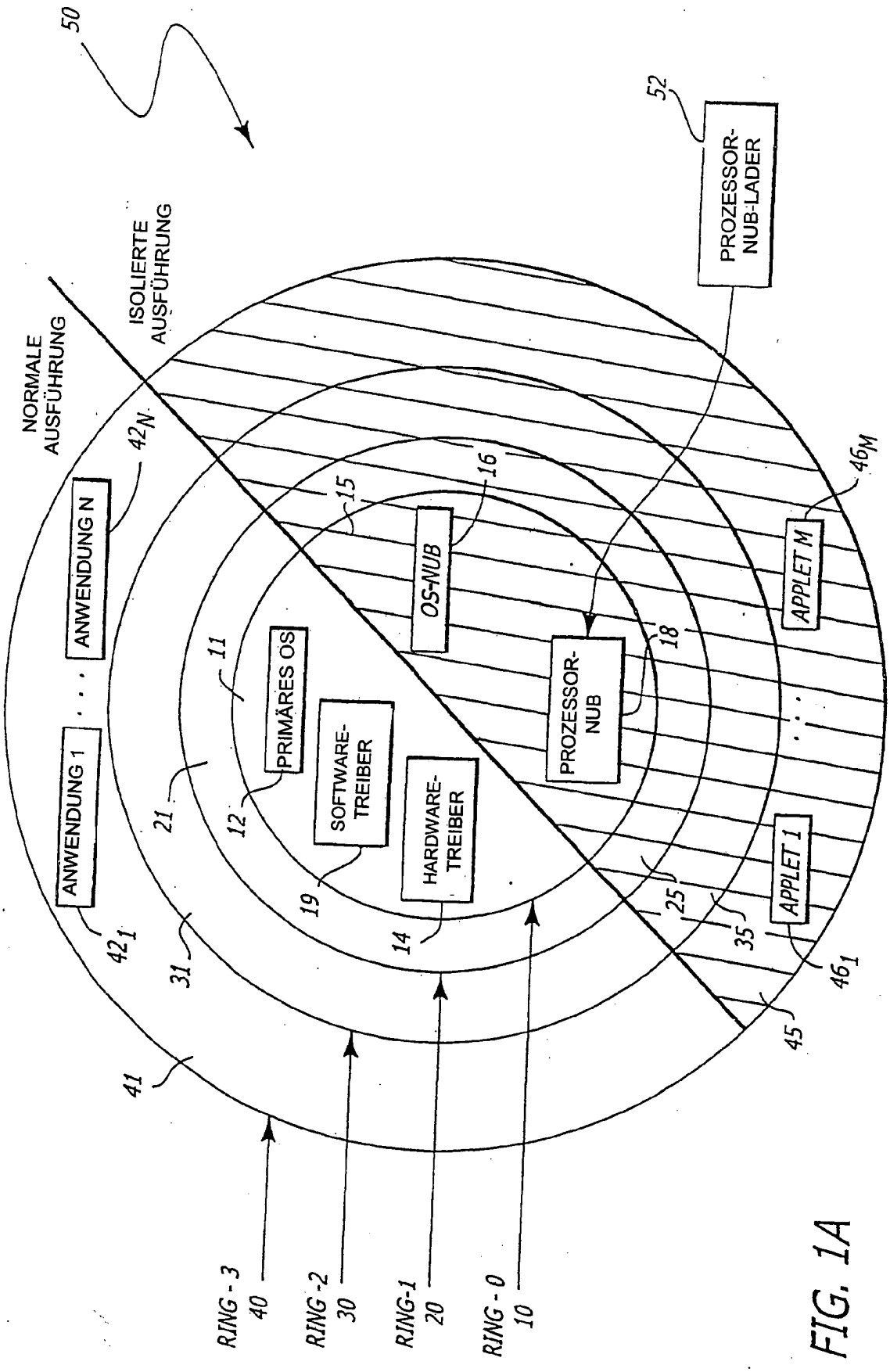


FIG. 1A

FIG. 1B

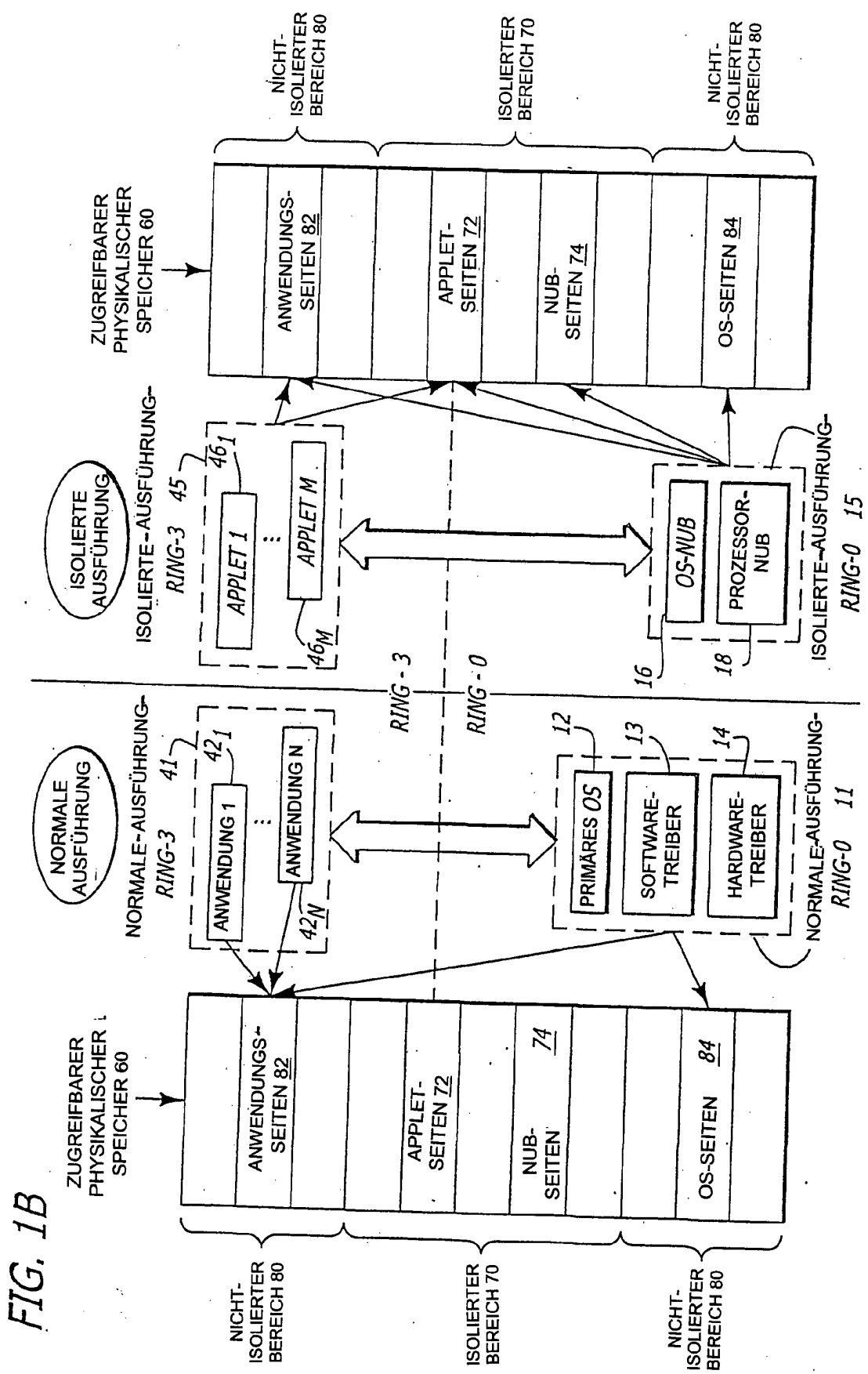
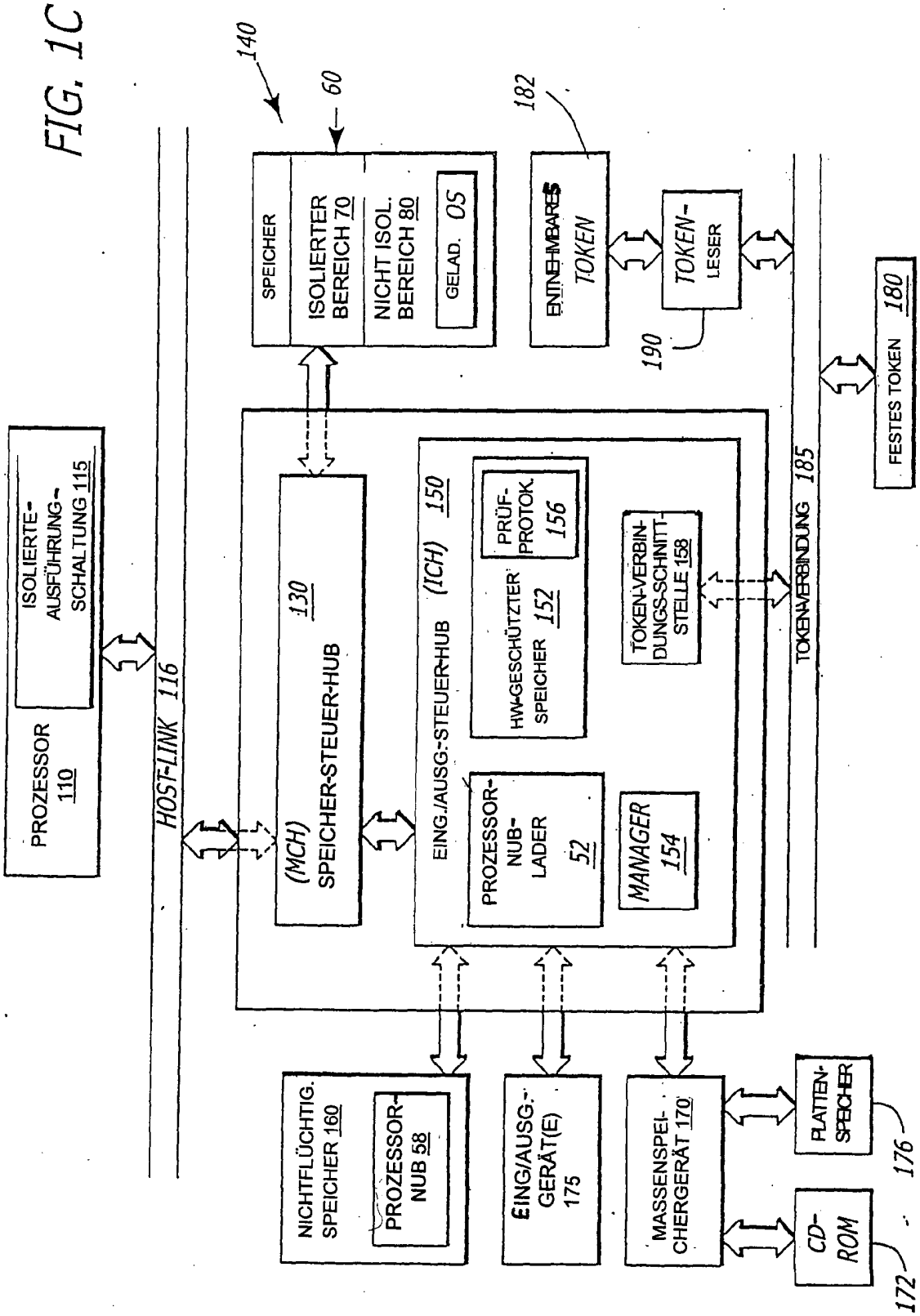

ZUGREIFBARER PHYSIKALISCHER SPEICHER 60
NICHT-ISOLIERTER BEREICH 80
ISOLIERTER BEREICH 70
NICHT-ISOLIERTER BEREICH 80
ANWENDUNGS-SEITEN 82
APPLET-SEITEN 72
NUB-SEITEN 74
OS-SEITEN 84
ISOLIERTE AUSFÜHRUNG
ISOLIERTE-AUSFÜHRUNG-RING-3 45
APPLET 1
APPLET M
461
46M
OS-NUB
PROZESSOR-NUB
16
18
ISOLIERTE-AUSFÜHRUNG-RING-0 15
RING - 3
RING - 0
NORMALE AUSFÜHRUNG
NORMALE-AUSFÜHRUNG-RING-3 41
ANWENDUNG 1
ANWENDUNG N
421
42N
PRIMÄRES OS
SOFTWARE-TREIBER
HARDWARE-TREIBER
12
13
14
NORMALE-AUSFÜHRUNG-RING-0 11
ZUGREIFBARER PHYSIKALISCHER SPEICHER 60
ANWENDUNGS-SEITEN 82
APPLET-SEITEN 72
NUB-SEITEN 74
OS-SEITEN 84
NICHT-ISOLIERTER BEREICH 80
ISOLIERTER BEREICH 70
NICHT-ISOLIERTER BEREICH 80

FIG. 1C


PROZESSOR 110
ISOLIERTE-AUSFÜHRUNG-SCHALTUNG 115
HOST-LINK 116
140
60
SPEICHER
ISOLIERTER BEREICH 70
NICHT ISOL. BEREICH 80
GELAD. OS
182
ENTNEHMBARES TOKEN
190
TOKEN-LESER
FESTES TOKEN 180
TOKENVERBINDUNG 185
(MCH) SPEICHER-STEUER-HUB 130
(ICH) EING./AUSG.-STEUER-HUB 150
HW-GESCHÜTZTER SPEICHER 152
PRÜF-PROTOK 156
TOKEN-VERBINDUNGS-SCHNITTSTELLE 158
PROZESSOR-NUB-LADER 52
MANAGER 154
NICHTFLÜCHTIG. SPEICHER 160
PROZESSOR-NUB 58
EING/AUSG.-GERÄT(E) 175
MASSENSPEICHERGERÄT 170
PLATTEN-SPEICHER
176
CD-ROM
172

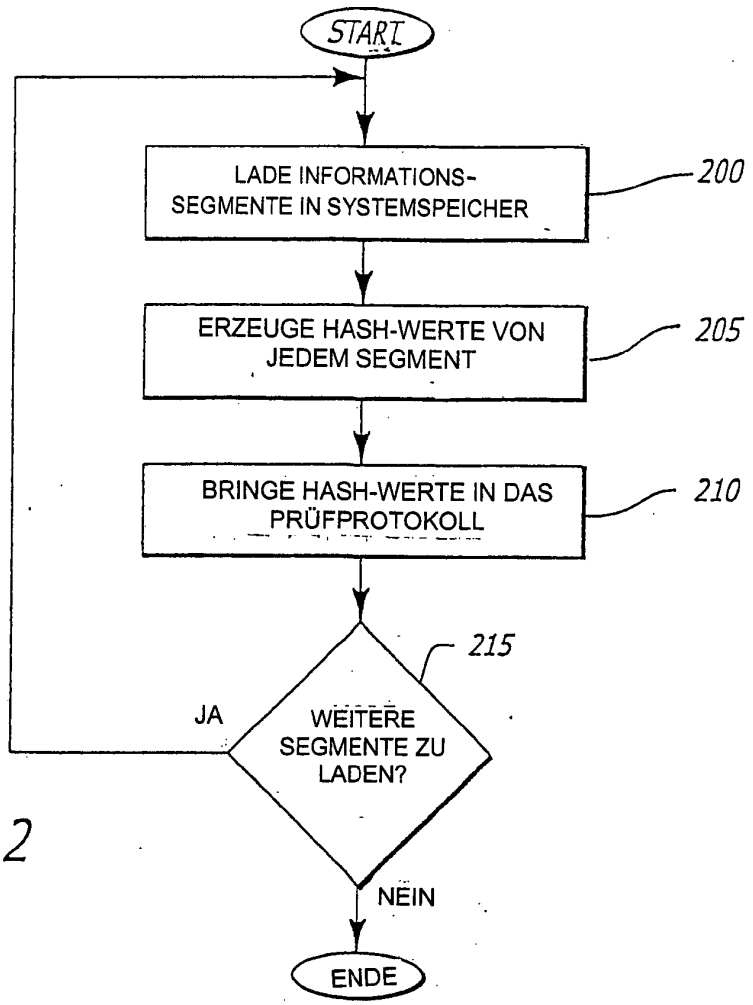

START
LADE INFORMATIONS-SEGMENTE IN SYSTEMSPEICHER
200
ERZEUGE HASH-WERTE VON JEDEM SEGMENT
205
BRINGE HASH-WERTE IN DAS PRÜFPROTOKOLL
210
215
JA
WEITERE SEGMENTE ZU LADEN?
NEIN
ENDE

FIG. 2

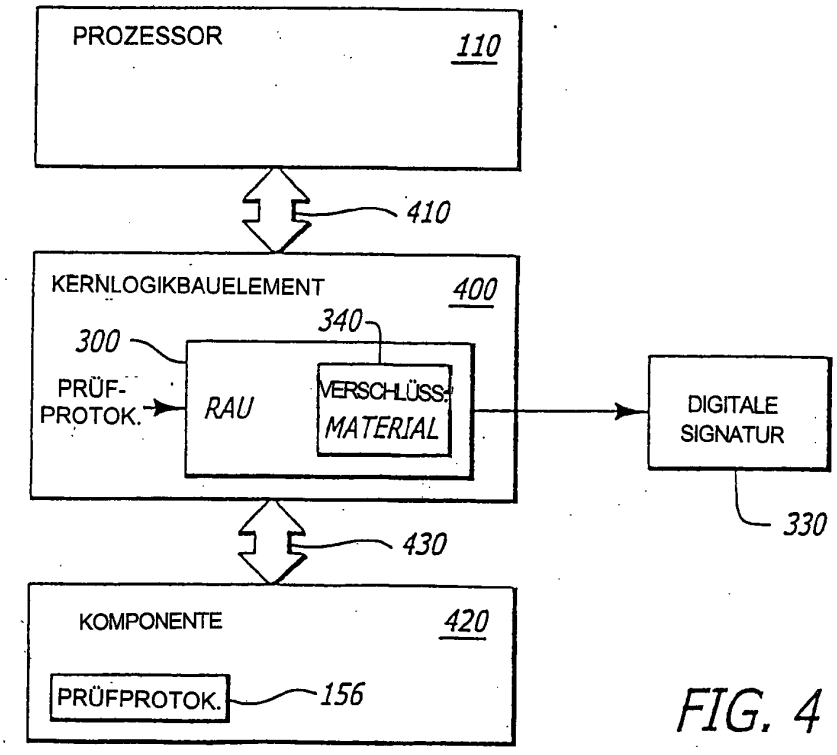
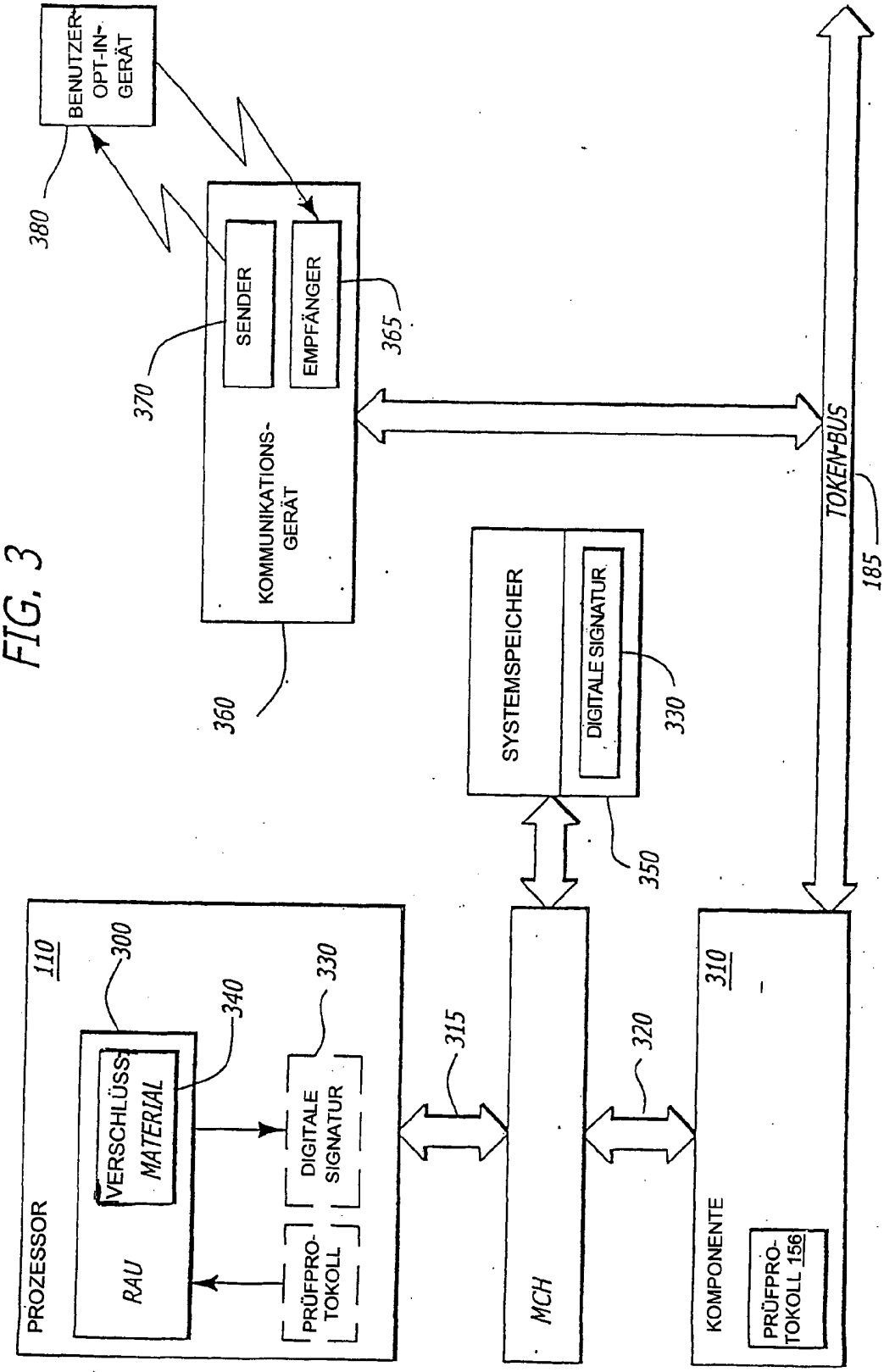

PROZESSOR
110
410
KERNLOGIKBAUELEMENT
400
340
300
PRÜF-PROTOK.
RAU
VERSCHLÜSS-MATERIAL
DIGITALE SIGNATUR
330
430
KOMPONENTE
420
PRÜFPROTOK.
156

FIG. 4

FIG. 3


BENUTZER-OPT-IN-GERÄT
380
SENDER
370
EMPFÄNGER
365
KOMMUNIKATIONS-GERÄT
360
TOKEN-BUS
185
SYSTEMSPEICHER
DIGITALE SIGNATUR
330
350
PROZESSOR
110
300
VERSCHLÜSS. MATERIAL
340
RAU
DIGITALE SIGNATUR
330
PRÜFPRO-TOKOLL
315
MCH
320
310
KOMPONENTE
PRÜFPRO-TOKOLL 156

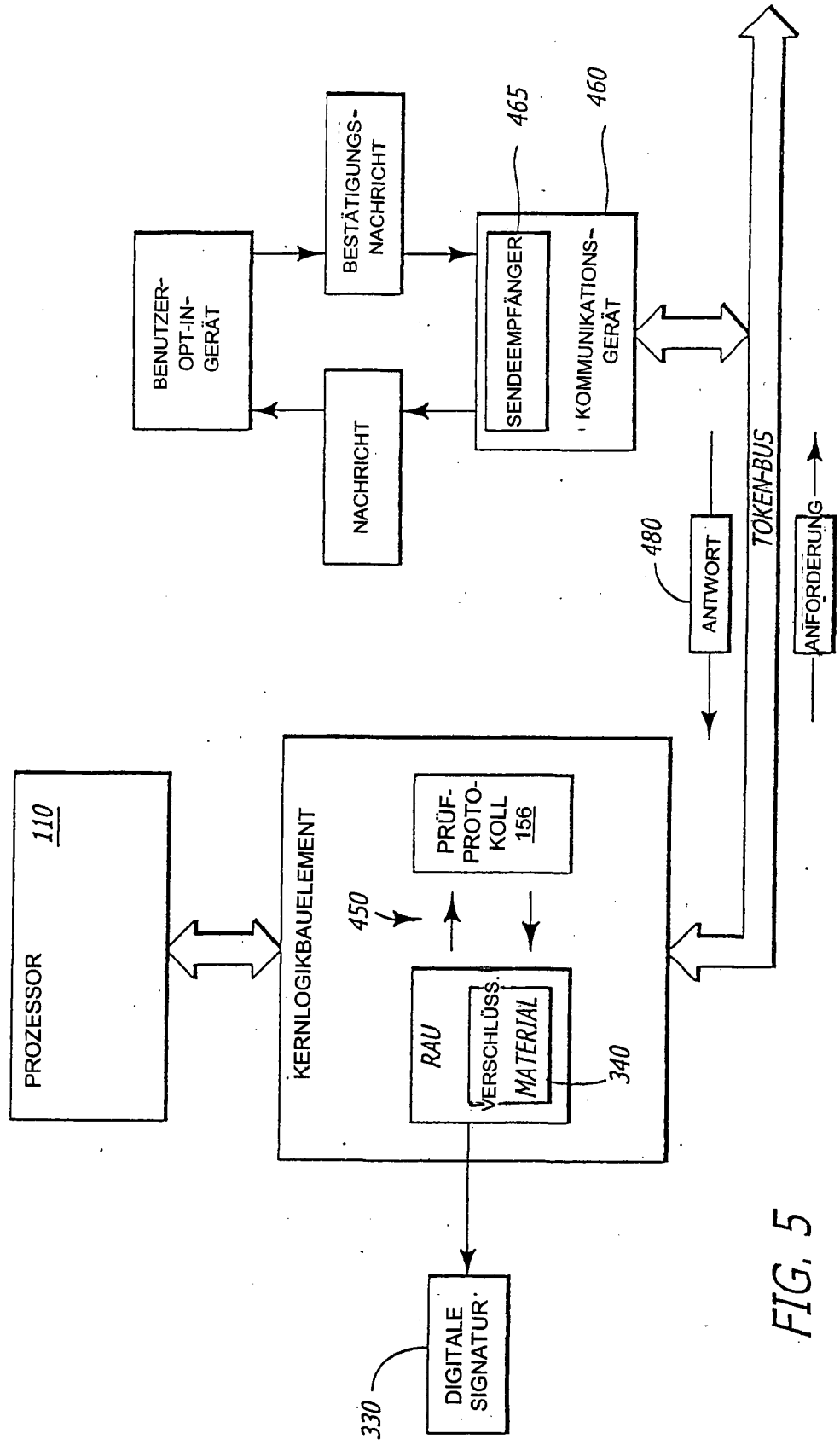

PROZESSOR
110
KERNLOGIKBAUELEMENT
PRÜF-PROTO-KOLL
156
450
RAU
VERSCHLÜSS. MATERIAL
340
DIGITALE SIGNATUR
330
BENUTZER-OPT-IN-GERÄT
BESTÄTIGUNGS-NACHRICHT
NACHRICHT
SENDEEMPFÄNGER
465
KOMMUNIKATIONS-GERÄT
460
TOKEN-BUS
ANTWORT
480
ANFORDERUNG

FIG. 5

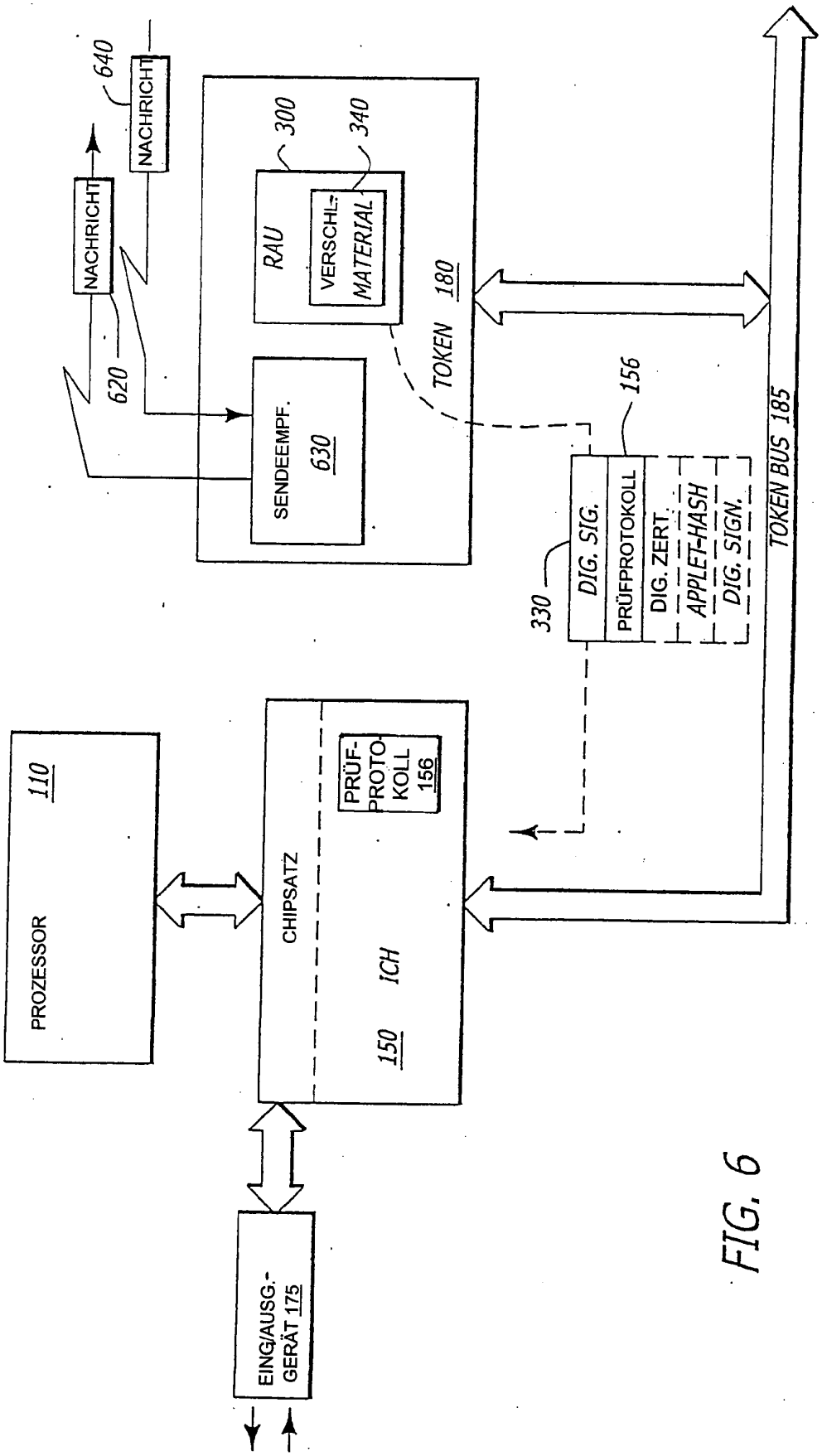

640
NACHRICHT
NACHRICHT
620
300
340
RAU
VERSCHL-
MATERIAL
TOKEN 180
SENDEEMPF.
630
156
330
DIG. SIG.
PRÜFPROTOKOLL
DIG. ZERT.
APPLET-HASH
DIG. SIGN.
TOKEN BUS 185
PROZESSOR
110
CHIPSATZ
PRÜF-
PROTO
KOLL
156
ICH
150
EING/AUSG.-
GERÄT 175

FIG. 6

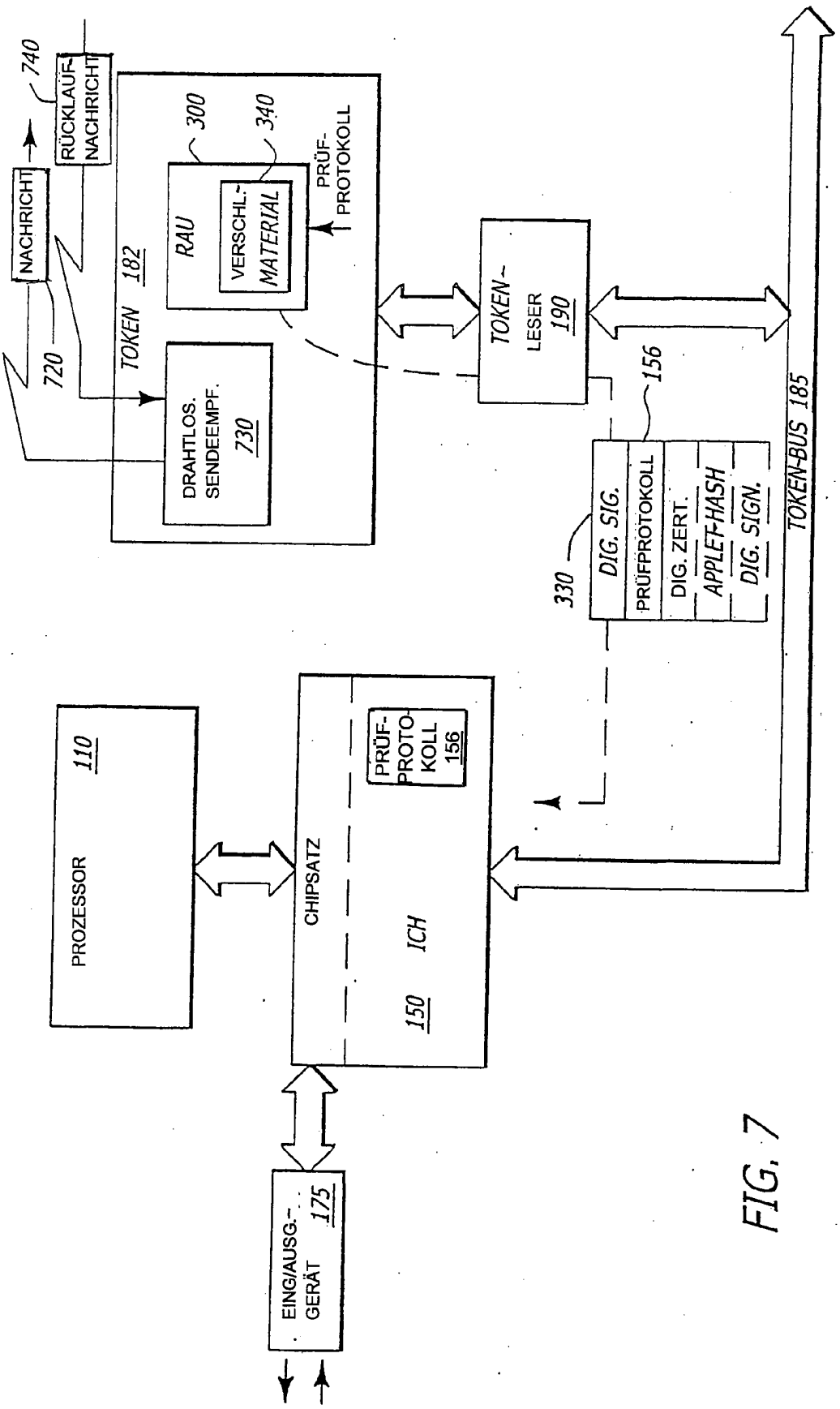

740
RÜCKLAUF-NACHRICHT
NACHRICHT
720
TOKEN 182
300
RAU
340
VERSCHL.-MATERIAL
PRÜF-PROTOKOLL
DRAHTLOS. SENDEEMPF. 730
TOKEN-LESER 190
156
330
DIG. SIG.
PRÜFPROTOKOLL
DIG. ZERT.
APPLET-HASH
DIG. SIGN.
TOKEN-BUS 185
PROZESSOR 110
CHIPSATZ
PRÜF-PROTO-KOLL 156
ICH 150
EING/AUSG.-GERÄT 175

FIG. 7