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

(10) **DE 601 13 395 T2** 2006.06.14

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 273 129 B1**
(21) Deutsches Aktenzeichen: **601 13 395.1**
(86) PCT-Aktenzeichen: **PCT/US01/40507**
(96) Europäisches Aktenzeichen: **01 927 441.4**
(87) PCT-Veröffentlichungs-Nr.: **WO 01/080483**
(86) PCT-Anmeldetag: **11.04.2001**
(87) Veröffentlichungstag der PCT-Anmeldung: **25.10.2001**
(97) Erstveröffentlichung durch das EPA: **08.01.2003**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **14.09.2005**
(47) Veröffentlichungstag im Patentblatt: **14.06.2006**

(51) Int Cl.[8]: ***H04L 9/32*** (2006.01)

(30) Unionspriorität:

197152	**13.04.2000**	**US**
261425 P	**13.01.2001**	**US**
827882	**04.04.2001**	**US**

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

(73) Patentinhaber:
Broadcom Corp., Irvine, Calif., US

(72) Erfinder:
BUER, Mark, Gilbert, US; LAW, Y., Patrick, Milpitas, US; QI, Zheng, Milpitas, US

(74) Vertreter:
Bosch, Graf von Stosch, Jehle Patentanwaltsgesellschaft mbH, 80639 München

(54) Bezeichnung: **VERFAHREN UND ARCHITEKTUR ZUR AUTHENTIFIZIERUNG**

Beschreibung

HINTERGRUND DER ERFINDUNG

[0001] Die vorliegende Erfindung betrifft allgemein das Gebiet der Kryptografie und im Besonderen eine Architektur und ein Verfahren in Bezug auf eine beschleunigte Kryptografieerstellung. Insbesondere richtet sich die Erfindung auf eine Hardware-Implementierung für die Erhöhung der Geschwindigkeit, mit der in den Datenpaketen, die über ein Computernetz übertragen werden, entsprechende Authentizitätsverfahren durchgeführt werden können.

[0002] Nach dem Stand der Technik sind zahlreiche Verfahren zur Kryptografieausführung allgemein bekannt und beispielsweise in der „Angewandten Kryptografie" [Applied Cryptography] von dem Kryptografie-Experten Bruce Schneier, John Wiley & Sons, Inc. (1996, 2. Auflage) erläutert, welche hierin als Bestandteil mit Referenzangabe enthalten sind. Um eine Geschwindigkeitsverbesserung in der Kryptografieverarbeitung zu erreichen, sind spezielle Kryptografie-Beschleunigerchips entwickelt worden. Kryptografie-Beschleunigerchips können beispielsweise in Routern [Leitwegvermittlungsrechnern] oder in Gateways [Netzübergangskopplungsstellen] zur Bereitstellung einer automatischen Verschlüsselung/Entschlüsselung von IP-Paketen enthalten sein. Durch ein Integrieren der Kryptografiefunktion in einer Netzwerk-Hardware werden sowohl die Systemleistung als auch die Datensicherheit verbessert.

[0003] In der Regel beinhalten Kryptografie-Protokolle sowohl das Verschlüsseln/Entschlüsseln als auch die Authentizitätsfunktion. Das Verschlüsseln/Entschlüsseln bezieht sich auf das Chiffrieren und Dechiffrieren von Daten, wobei die Authentizität die Datenintegrität und das Bestätigen der Identität des übertragenden Teilnehmers betrifft sowie das Sicherstellen einschließt, dass unterwegs an den Empfänger keine unerlaubten Änderungen in einem Datenpaket vorgenommen worden sind. Es ist allgemein bekannt, dass durch das Integrieren sowohl von den Verschlüsselungs- als auch von den Authentizitätsfunktionalitäten in einem einzigen Beschleunigerchip die gesamte Systemleistung verbessert werden kann.

[0004] Ausführungsbeispiele für Kryptografie-Protokolle, die Verschlüsselungs/Entschlüsselungs- sowie Authentizitätsfunktionalitäten integrieren, beinhalten auch ein Sicherheitsverschlüsselungsprotokoll für das Internet, d. h. ein SSL = Security Sockets Layer-Protokoll (Netscape Communications Corporation), das hauptsächlich für elektronische Geschäftstransaktionen verwendet wird, sowie das in jüngster Zeit weit verbreitete, branchenübliche Internetsicherheitsstandard-Protokoll, das als „IPSec." bekannt geworden ist. Diese Protokolle und deren zugehörige Algorithmen sind auf dem Gebiet der Kryptografie sehr bekannt und werden in den Normvorschriften des „National Institute of Standards and Technology (NIST), IETF und anderen Spezifikationen in den Einzelheiten beschrieben, von denen einige (zum Beispiel IETF RFC#) nachfolgend für praktische Ausführungszwecke in Verbindung gebracht werden. Diese Spezifikationen sind hierin unter Referenzangabe für alle entsprechenden Ausführungsbeispiele enthalten.

[0005] Das SSL-Protokoll (v3) verwendet zur Authentifizierung eine Algorithmusvariante des HMAC (RFC2104). Der zugrunde liegende Hash-Algorithmus kann entweder ein MD5 (RFC1321) oder ein SHA1 (NIST) sein. Hinzu kommt, dass sich der in dem SSL vorhandene Schlüsselgenerierungsalgorithmus außerdem auf eine Sequenzfolge von MD5- und SHA1-Operationen aufbaut. Das SSL setzt für die Verschlüsselungs-/Entschlüsselungsvorgänge entsprechende Algorithmen wie RC4, DES [Data Encryption Standard = Datenverschlüsselungsstandard] und Triple-DES ein.

[0006] Das IP-Schichtsicherheitsstandard-Protokoll – IPSec. (RFC 2406) – spezifiziert zwei Standardalgorithmen zur Durchführung der Authentifizierungsvorgänge, nämlich HMAC-MD5-96 (RFC2403) und HMAC-SHA1-96 (RFC2404). Diese Algorithmen basieren auf den zugrunde liegenden MD5- bzw. SHA1-Algorithmen. Ziel der Authentizitätsberechnung ist es, eine eindeutige, digitale Darstellung für die Eingangsdaten zu generieren, die Digest genannt wird.

[0007] Sowohl MD5 als auch SHA1 spezifizieren, dass die Daten in 512-Bit-Blöcken verarbeitet werden müssen. Falls die Daten in einem zu verarbeitenden Paket mit einem Vielfachen nicht 512 Bits ergeben, wird ein Padding [Auffüllen] genutzt, um die Datenlänge mit einem Vielfachen auf 512 Bits aufzurunden. Falls daher somit ein Datenpaket, das von einem Chip zur Authentizitätsprüfung empfangen wird, größer als 512 Bits ist, wird das Paket in 512-Bits-Datenblöcke für eine Authentifizierungsverarbeitung umgebrochen. Wenn die Paketgröße nicht genau ein Vielfaches von 512 Bits ist, müssen die übrig bleibenden Daten nach dem Aufteilen des Paketes in komplette 512-Bit-Blöcke aufgefüllt werden, um eine 512-Bit-Block-Bearbeitungsgröße zu erhalten. Dasselbe trifft zu, wenn ein Paket weniger als 512 Datenbits enthält. Zur Information, ein typisches

Ethernet-Paket besteht aus bis zu 1.500 Bytes. Wenn ein solches Paket in 512-Bit-Blöcke aufgeteilt wird, wird nur der letzte Block aufgefüllt und so, dass insgesamt nur ein relativ kleiner Prozentanteil einer Padding-Befehlsverarbeitungszeit erforderlich ist. Jedoch kann für sehr viel kleinere Paketlängen die Padding-Befehlsverarbeitungszeit sehr viel größer sein. Wenn ein Paket zum Beispiel knapp über 512 Bits groß ist, muss es in zwei 512-Bit-Blöcke unterteilt werden, von denen der zweite Block zum größten Teil aufgefüllt werden muss, so dass der Padding-Aufwand auf 50% der Befehlsverarbeitungsdaten herankommt. Die Authentifizierung von solch kleinen Datenpaketen ist besonders mühsam und zeitaufwändig, wenn man die herkömmlich implementierten MD5- und SHA1-Authentizitätsalgorithmen anwendet.

[0008] Für jeden 512-Bit-Datenblock wird ein Satz von Operationen mit nichtlinearen Funktionen, Verschiebefunktionen und Additionen, eine so genannte „Durchlaufrunde" auf den Block wiederholend angewendet. Die Algorithmen MD5 und SHA1 spezifizieren 64 Runden bzw. 80 Runden, die auf unterschiedlichen, nichtlinearen und verschiebenden Funktionen sowie auf den unterschiedlichen Schaltsequenzen basieren. In jeder Durchlaufrunde startet der Betriebsvorgang mit bestimmten Hash-Zuständen (wird auch als "Kontext" bezeichnet), die in Hash-Zustandsregistern (in der Hardware) oder in Hash-Variablen (in der Software) vorhanden sind, und endet mit einem neuen Satz von Hash-Zuständen (d. h. mit einem Ausgangs-„Satz" mit Hash-Zuständen und mit einem Endsatz. Ein „Satz" kann aus 4 oder 5 zählenden Registern bestehen, den von den Algorithmen MD5 bzw. SHA1 verwendet werden). Die Algorithmen MD5 und SHA1 spezifizieren jeweils einen Satz mit Konstanten für den ersten 512-Bit-Block als Hash-Ausgangszustände. Die nachfolgenden Datenblöcke verwenden Hash-Ausgangszustände, die aus den Additionen der Hash-Ausgangszustände mit den Hash-Endzuständen der vorausgegangen Blöcke resultieren.

[0009] In der Regel werden die MD5- und SHA1-Durchlaufrunden in die Hardware-Implementierungen mit Taktzyklen übertragen. Die Additionen der Hash-Zustände bis zu dem Größenumfang, dass sie nicht mit anderen Rundenoperationen parallel durchgeführt werden können, machen während der gesamten Berechnungszeit (aufwändige) Überhangstaktzyklen notwendig. Die Computerberechnung des Datenanteils für das Padding wird außerdem allgemein als aufwändiger Leistungsüberhang betrachtet, da dieses nicht Bestandteil der echten Daten ist. Demzufolge wird die Betriebsleistung von MD5 und SHA1 am meisten herabgesetzt, wenn die Länge des Padding in etwa der Länge der Daten entspricht (d. h. wie vorstehend beschrieben, wenn ein Paket viel weniger als 512 Datenbits aufweist und die Padding-Logik ein 512-Extra-Bit benötigt, das für das Parken der Padding-Werte hinzugefügt werden muss).

[0010] Darüber hinaus weiten die in dem Protokoll IPSec. verwendeten Algorithmen HMAC-MD5-96 bzw. HMAC-SHA1-96 die Algorithmen MD5 und SHA1 durch das Durchführen von zwei Schleifenoperationen aus. Der HMAC-Algorithmus entweder für den MD5 oder den SHA1 (HMAC-x Algorithmus) wird in **Fig. 1** dargestellt. Die Hash-Inneneinheit (innere Schleife) und die Hash-Außeneinheit (äußere Schleife) verwenden unterschiedliche Hash-Ausgangszustände. Der Außen-Hash wird zur Berechnung eines Digests verwendet, der auf dem Ergebnis des inneren Hashwertes basiert. Da das Ergebnis des Innen-Hash für MD5 eine Länge von 128 Bits und für SHA1 eine 160 Bit Länge aufweist, muss das Ergebnis immer auf 512 Bits aufgefüllt werden, denn die Hash-Außeneinheit kann nur einen 512-Bit-Datenblock verarbeiten. Die erweiterten Algorithmen HMAC-MD5-96 und HMAC-SHA1-96 bieten eine höhere Sicherheitsstufe, jedoch wird zusätzliche Zeit für die Durchführung der Außen-Hash-Operation benötigt. Dieser zusätzliche Zeitaufwand wird von Bedeutung, wenn die Länge der zu verarbeitenden Daten sehr klein ist, wobei in diesem Fall die benötigte Zeit zur Durchführung der Außen-Hash-Operation mit der benötigten Zeit vergleichbar ist, die für die Durchführung der Innen-Hash-Operation aufzuwenden ist.

[0011] Die Authentifizierungsprüfung stellt einen erheblichen Zeitanteil dar, der für die Komplettierung der kryptografischen Betriebsvorgänge unter Anwendung von Kryptografie-Protokollen erforderlich ist, die sowohl das Verschlüsseln/Entschlüsseln als auch die MD5- und/oder SHA1-Authentizitätsfunktionalitäten einbeziehen. Im Falle des Sicherheitsprotokolls IPSec. ist die Authentifizierung häufig ein zeitlich eingeschränkter Betriebsschritt, insbesondere bei der Verarbeitung von sehr kleinen Paketen, und sie verursacht infolgedessen einen Datenverarbeitungsengpass. Dementsprechend sind Techniken zur Beschleunigung der Authentifizierung und somit zur Umgehung dieses Engpasses erwünscht. Ferner wären schnellere Implementierungen der Mehrfach-Durchlaufrunden-Algorithmen zur Authentifizierung für jede Anwendung von diesbezüglichen Authentizitätsalgorithmen von Vorteil.

[0012] Der bereits vorstehend zitierte Kryptografie-Experte Schneier bezieht sich auf Implementierungen von Mehrfach-Durchlaufrunden-Algorithmen für eine Authentifizierung unter Anwendung von entsprechenden Kryptografievorgängen.

[0013] Touch, J. D.: Die Veröffentlichung "Performance Analysis of MD5" in Computer Communications Review von dem Verband Association for Computing Machinery, New York/USA, Band 25, Nr. 4, vom 1. Oktober 1995, bezieht sich auf eine Analyse des MD5, der ein spezieller Mehrfach-Durchlaufrunden-Algorithmus zur Authentifizierung ist.

ZUSAMMENFASSUNG DER ERFINDUNG

[0014] Die vorliegende Erfindung stellt im Wesentlichen eine Architektur (Hardware-Implementierung) für eine Authentifizierungsfunktionseinheit zur Erhöhung der Geschwindigkeit bereit, mit der mehrschleifige und/oder mehrfache Rundenauthentizitätsalgorithmen in Datenpaketen, die über ein Computernetzwerk übermittelt wurden, durchgeführt werden können. Wie in dieser Patentanmeldung beschrieben ist, weist die vorliegende Erfindung ein spezielles Anwenderprogramm bezüglich der Varianten der SHA1- und MD5-Authentizitätsalgorithmen auf, die in dem IPSec.-Kryptografiestandard-Protokoll spezifiziert sind. Übereinstimmend mit der Normvorschrift des IPSec-Standards kann die vorliegende Erfindung in Verbindung mit Verschlüsselungs-/Entschlüsselungsarchitekturen und -Protokollen für Datenübertragungen eingesetzt werden. Sie ist jedoch außerdem für die Anwendung in Verbindung mit anderen Kryptografie-Algorithmen ohne IPSec.-Protokoll vorgesehen und ferner für Anwendungen geeignet, in denen keine Verschlüsselung/Entschlüsselung (mit oder ohne IPSec) durchgeführt wird, in denen es rein um die Authentifizierung geht, die es zu beschleunigen gilt. Neben anderen Vorteilen stellt die erfindungsgemäße Authentifizierungsfunktionseinheit eine verbesserte Betriebsleistung im Hinblick auf das Verarbeiten von sehr kleinen Datenpaketen bereit.

[0015] Authentifizierungsfunktionseinheiten gemäß der vorliegenden Erfindung setzen eine Vielfalt an Techniken ein, die – in verschiedenen Anwendungen – das Kollabieren/Herunterbrechen von zwei Authentifizierungsalgorithmus-Mehrfach-Durchlaufrunden (z. B. SHA1 bzw. MD5 oder Varianten davon) und das Verarbeiten der Runden zu einer Runde beinhalten, das Reduzieren des betrieblichen Befehlsverarbeitungsaufwands durch die Planungserstellung der Additionen, die von einem mehrere Runden durchlaufenden Authentifizierungsalgorithmus benötigt werden, so dass die gesamten kritischen Zeittaktpfade verringert werden können („Übergehen von Additionen"), und die in Bezug auf eine mehrschleifige Variante (d. h. HMAC) eines mehrere Runden durchlaufenden Authentifizierungsalgorithmus die Befehlsverknüpfung [Kanalisieren] der Innen- und Außenschleifen umfasst. In einem speziellen Ausführungsbeispiel wendet die vorliegende Erfindung in einer Authentifizierungsfunktionseinheit den HMAC-SHA1-Algorithmus des IPSec.-Protokolls an, wobei das Herunterbrechen von den herkömmlichen 80 SHA-1-Runden in 40 Durchlaufrunden, das Umgehen von Additionen und das Befehlsverknüpfen/Kanalisieren der Innen- und Außenschleifen zugelassen wird, so dass der HMAC-SHA1 nahezu in der gleichen Zeit wie der herkömmliche SHA1 durchgeführt werden kann.

[0016] In einer Ausführungsform bezieht sich die vorliegende Erfindung auf eine Authentifizierungsfunktionseinheit mit einer Architektur für einen mehrschleifigen, mehrere Runden durchlaufenden Authentifizierungsalgorithmus. Die Architektur umfasst eine erste Instanz von einem Mehrfach-Durchlaufrunden-Authentifzierungsalgorithmus mit einer Hash-Rundenlogik in einer Hash-Innenfunktionseinheit und einer zweiten Instanz von einem Mehrfach-Durchlaufrunden-Authentifzierungsalgorithmus mit einer Hash-Rundenlogik in einer Hash-Außenfunktionseinheit. Es sind außerdem ein Doppelpaketnutzlast-Dateneingangspuffer, der während der Verarbeitung eines anderen Datenblocks in der Hash-Innenfunktionseinheit zum Laden eines neuen Datenblocks konzipiert ist, eine Hash-Ausgangszustandseingabe-Pufferkonfiguration zum Laden der Hash-Ausgangszustände in die Hash-Innen- und Außenfunktionseinheiten für gleichzeitige Hash-Operationen der inneren und äußeren Hashwerte und ein Dual-Port-Rom-Speicher, der für simultane Konstantensuchverfahren sowohl für die Innen- als auch für die Außen-Hash-Einrichtungen konfiguriert ist, enthalten. Der mehrschleifige Mehrfach-Durchlaufrunden-Algorithmus zur Authentifizierung kann sowohl ein HMAC-MD5 als auch ein HMAC-SHA1 sein.

[0017] In einer weiteren Ausführungsform bezieht sich die vorliegende Erfindung auf eine Authentifizierungsfunktionseinheit mit einer Architektur für einen mehrschleifigen, mehrere Runden durchlaufenden Authentifizierungsalgorithmus. Die Architektur enthält eine Hash-Funktionseinheit, die zur Implementierung der Hash-Rundenlogik für einen Mehrfach-Durchlaufrunden-Algorithmus ausgelegt ist. Das Implementieren der Hash-Rundenlogik schließt mindestens ein Additionsmodul ein, das eine Mehrzahl von Übertragsspeicheraddierwerken (CSA) für die Berechnung von Teilprodukten und ein Übertragsvorausberechnungsaddierwerk (CLA) für die Berechnung und Reproduktion einer Endsumme umfasst. Der Mehrfach-Durchlaufrunden-Algorithmus zur Authentifizierung kann ein MD5 oder ein SHA1 sein.

[0018] In einer weiteren Ausführungsform bezieht sich die vorliegende Erfindung auf eine Authentifizierungsfunktionseinheit mit einer Architektur für einen SHA1-Authentifizierungsalgorithmus. Die Architektur enthält

mindestens eine Hash-Funktionseinheit, die zur Implementierung der Hash-Rundenlogik konfiguriert ist. Die Logikimplementierung umfasst fünf Hash-Zustandsregister, einen kritischen und vier nichtkritische Datenpfade, die zu den fünf Registern gehören. In den nachfolgenden SHA1-Durchlaufrunden sind die Register mit dem kritischen Pfad eine Ausweichalternative.

[0019] In einer anderen Ausführungsform bezieht sich die vorliegende Erfindung auf ein Verfahren zur Authentifizierung von Daten, die über ein Computernetz übertragen werden. Das Verfahren umfasst das Empfangen eines Datenpaketstroms, das Aufteilen des Datenpaketstroms in Datenblöcke mit festgelegter Größe und das Verarbeiten der Datenblöcke mit festgelegter Größe unter Verwendung einer mehrschleifigen, mehrere Runden durchlaufenden Authentifizierungseinrichtungsarchitektur, die eine Hash-Kernfunktionseinheit mit einer Innen-Hash-Einrichtung und einer Außen-Hash-Einrichtung aufweist. Die Architektur ist zur Kanalisierung [Befehlsverknüpfung] der Hashwert-Operationen in den Innen-Hash- und Außen-Hash-Einrichtungen, zum Kollabieren und Neuarrangieren der Mehrfach-Durchlaufrundenlogik für eine Reduzierung der Runden im Hinblick auf die Hashwert-Operationen und zum Implementieren der Mehrfach-Durchlaufrunden-Logik für die Planungserstellung der Additionscomputerberechnungen ausgelegt, welche zu den Rundenoperationen parallel durchgeführt werden sollen. Der mehrschleifige Mehrfach-Durchlaufrunden-Algorithmus zur Authentifizierung kann ein HMAC-MD5 oder ein HMAC-SHA1 sein.

[0020] In einer anderen Ausführungsform bezieht sich die vorliegende Erfindung auf ein Verfahren zur Authentifizierung von Daten, die über ein Computernetz übertragen werden. Das Verfahren umfasst das Empfangen eines Datenpaketstroms, das Aufteilen des Datenpaketstroms in Datenblöcke mit festgelegter Größe und das Verarbeiten der Datenblöcke mit festgelegter Größe unter Verwendung einer mehrere Runden durchlaufenden Authentifizierungseinrichtungsarchitektur. Die Architektur implementiert eine Hash-Rundenlogik für einen mehrere Runden durchlaufenden Authentifizierungsalgorithmus, die für die Planungserstellung von Additionscomputerberechnungen ausgelegt ist, welche zu den Rundenoperationen parallel durchzuführen sind. Der Mehrfach-Durchlaufrunden-Algorithmus zur Authentifizierung kann ein MD5 oder ein SHA1 sein.

[0021] In einer weitergehenden, anderen Ausführungsform betrifft die vorliegende Erfindung ein Verfahren für eine Authentifizierung von Daten, die über ein Computernetz übertragen werden, unter Anwendung eines SHA1-Authenzitätsalgorithmus. Das Verfahren umfasst die Bereitstellung von fünf Hash-Zustandsregistern und die Bereitstellung von Datenpfaden aus den fünf Zustandsregistern dergestalt, dass in jeder SHA1-Runde vier von den fünf Datenpfaden aus den Registern zeitlich nicht kritisch sind.

[0022] Diese und weitere kennzeichnenden Merkmale und Vorteile gemäß der vorliegenden Erfindung werden in der nachstehenden Beschreibung in den erfindungsgemäßen Einzelheiten und in den zugehörigen Figuren dargelegt, die anhand von Ausführungsbeispielen die Grundsätze der Erfindung veranschaulichen sollen.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

[0023] Die vorliegende Erfindung wird aufgrund der nachfolgenden, detaillierten Beschreibung in Verbindung mit den zugehörigen Zeichnungen leicht verstanden, wobei die gleichen Bezugszeichen die gleichen Strukturelemente benennen, und in denen:

[0024] Fig. 1 ein Oberstufen-Blockdiagramm ist, das den HMAC-x-Algorithmus (HMAC entweder für MD5 oder für SHA1) darstellt, der in dem IPSec.-Standard-Protokoll implementiert ist;

[0025] Fig. 2 ein Oberstufen-Blockdiagramm von einer Authentifizierungsfunktionseinheit mit einer Architektur gemäß einer Ausführungsform der vorliegenden Erfindung ist;

[0026] Fig. 3 ein Zeitstudiendiagramm ist, das den kritischen Pfad von der herkömmlichen Rundenlogik für den SHA1-Authentifizierungsalgorithmus veranschaulicht;

[0027] Fig. 4 ein Zeitstudiendiagramm ist, das den kritischen Pfad von der Rundenlogik für den SHA1-Authentifizierungsalgorithmus gemäß einer Ausführungsform der vorliegenden Erfindung veranschaulicht;

[0028] Fig. 5 ein Oberstufen-Blockdiagramm von einer SHA1-Hash-Funktionseinheit ist, das die wesentlichen Elemente des Rundenlogikaufbaus gemäß einer Ausführungsform der vorliegenden Erfindung veranschaulicht;

[0029] Fig. 6 ein Unterstufen-Blockdiagramm ist, das die Einzelheiten von der Planungserstellung der Additionen innerhalb des Rundenlogikaufbaus der Fig. 5 veranschaulicht.

DETAILLIERTE BESCHREIBUNG DER AUSFÜHRUNGSFORMEN GEMÄSS DER VORLIEGENDEN ERFINDUNG

[0030] Es wird nun auf ein paar spezielle Ausführungsformen der Erfindung detailliert Bezug genommen, welche nach Ansicht der Erfinder die bestmöglichen Betriebsfunktionen zur Ausführung der Erfindung beinhalten. In den beigefügten Zeichnungen werden Beispiele für diese spezifischen Ausführungsformen veranschaulicht. Obgleich die Erfindung in Verbindung mit diesen besonderen Ausführungsformen beschrieben wird, muss man davon ausgehen, dass die Erfindung nicht auf die beschriebenen Ausführungsformen eingeschränkt werden darf. Im Gegenteil, es ist lediglich beabsichtigt, Alternativen und Modifikationen aufzuzeigen, die außerdem im Schutzumfang der Erfindung gemäß den anhängenden Patentansprüchen enthalten sind. In der nun folgenden Beschreibung sind viele spezifische Einzelheiten dargelegt, um für die vorliegende Erfindung ein volles Verständnis zu erhalten. Die vorliegende Erfindung kann mit einigen oder aber auch mit allen dargestellten, spezifischen Details ausgeführt werden. Bei manchen Gelegenheiten sind allgemein bekannte Betriebsvorgänge nicht detailliert beschrieben, um die vorliegende Erfindung nicht unnötigerweise schwer verständlich werden zu lassen.

[0031] Die vorliegende Erfindung stellt im Wesentlichen eine Architektur (Hardware-Implementierung) für eine Authentifizierungsfunktionseinheit zur Erhöhung der Geschwindigkeit bereit, mit der mehrschleifige und/oder mehrfach durchlaufende Rundenauthentizitätsalgorithmen in Datenpaketen, die über ein Computernetzwerk übermittelt wurden, durchgeführt werden können. Authentifizierungsfunktionseinheiten gemäß der vorliegenden Erfindung setzen eine Vielfalt an Techniken ein, die – in verschiedenen Anwendungen – das Kollabieren/Herunterbrechen von zwei Mehrfach-Durchlaufrundenalgorithmen zur Authentifizierung (z. B. SHA1 bzw. MD5 oder Varianten davon) und das Verarbeiten der Runden zu einer Runde beinhalten, das Reduzieren des betrieblichen Befehlsverarbeitungsaufwands durch die Planungserstellung der Additionen, die von einem mehrere Runden durchlaufenden Authentifizierungsalgorithmus (z. B. SHA1 oder Varianten) benötigt werden, so dass die gesamten, kritischen Zeittaktpfade verringert werden („Übergehen von Additionen"), und die in Bezug auf eine HMAC-Variante (Mehrfachschleife) eines Mehrfach-Durchlaufrunden-Authentifizierungsalgorithmus die Befehlsverknüpfung [Kanalisieren] der Innen- und Außenschleifen umfasst. Neben anderen Vorteilen stellt die Authentifizierungsfunktionseinheit gemäß der vorliegenden Erfindung eine verbesserte Betriebsleistung im Hinblick auf das Verarbeiten von sehr kleinen Datenpaketen bereit.

[0032] In dieser Patentschrift enthalten die Singularformen „ein/eine/eines/einen" und „der/die/das" auch einen Pluralbezug, außer wenn der Kontext eindeutig etwas anderes aussagt. Wenn nichts anderes definiert ist, haben sämtliche technischen und wissenschaftlichen Begriffe, die hierin verwendet werden, dieselbe Bedeutung, wie sie allgemein von den Fachleuten auf diesem Gebiet bekannt sind, an die diese Erfindung gerichtet ist.

[0033] Die vorliegende Erfindung kann vielfältig eingesetzt werden. Wie in dieser Patentanmeldung beschrieben ist, weist die Erfindung ein spezielles Anwenderprogamm bezüglich der Varianten der SHA1- und MD5-Authentizitätsalgorithmen auf, die in dem IPSec.-Kryptogafiestandard-Protokoll spezifiziert sind. Die Erfindung wird in der nachfolgenden Beschreibung in erster Linie in Zusammenhang mit dem IPSec.-Protokoll erläutert. Die Fachleute auf diesem Gebiet werden jedoch erkennen, dass außerdem verschiedene Ausführungsformen gemäß der Erfindung auf mehrschleifige und/oder mehrfach durchlaufende Rundenauthentizitätsalgorithmen allgemein angewendet werden können, ganz gleich, ob sie mit dem IPSec.-Protokoll, oder ob sie überhaupt in Verbindung mit kryptogafischen Vorgängen zum Einsatz kommen. Obgleich ferner die Ausführungsformen gemäß der vorliegenden Erfindung, die nachstehend beschrieben werden, gemeinsam in einem bevorzugten Ausführungsbeispiel der Erfindung Verwendung finden, können unabhängig davon verschiedene Ausführungsaspekte zum Beschleunigen von Authentifizierungsvorgängen eingesetzt werden. Es sind zum Beispiel die Befehlsverknüpfungsvorgänge insbesondere auf mehrschleifige und mehrere Runden durchlaufende Authentifizierungsalgorithmen anwendbar, wobei das Herunterbrechen/Kollabieren der Rundenvorgänge speziell auf den SHA1 und Varianten der Authentifizierungsalgorithmen anwendbar ist, während die Planungserstellung von Additionen auf irgendeinen Mehrfach-Durchlaufrunden-Algorithmus erfolgen kann.

Befehlsverknüpfung für Innen und Außenoperationen der Hashwerte

[0034] Fig. 2 ist ein Blockdiagamm auf höherer Ebene von einer Authentifizierungsfunktionseinheit mit einer Architektur gemäß einer Ausführungsform der vorliegenden Erfindung. Die Funktionsarchitektur hat eine vor-

bereitende Befehlsverknüpfungseinrichtung zur Umgehung des Zeitraums implementiert, der zur Durchführung der äußeren Hash-Operation erforderlich ist, wenn laufend viele Datennutzlasten in die Funktionsarchitektur eingespeist werden. Die Funktionseinheitsarchitektur umfasst einen Kernteil, der zwei Instanzen der Hash-Rundenlogik aufweist; in diesem Beispiel eine Hash-Innen- und eine Hash-Außenfunktionseinheit (innere und äußere Schleifen) jeweils für die MD5-Hash-Rundenlogik als auch für die SHA1-Hash-Rundenlogik, die von dem IPSec.-Protokoll unterstützt werden. Eine Befehlsverknüpfungssteuerlogik stellt sicher, dass die äußere Hash-Operation für eine Datennutzlast parallel zu der inneren Hash-Operation mit der nächsten Datennutzlast in dem Paketstrom durchgeführt werden kann, welcher in die Authentifizierungsfunktionseinheit eingespeist wird. Ein Doppelpaketnutzlast-Dateneingangspuffer kommt in Bezug auf die Hash-Innenfunktionseinheit zum Einsatz und ermöglicht somit, dass ein neuer 512-Bit-Datenblock geladen werden kann, während ein anderer gerade verarbeitet wird, wobei die Hash-Ausgangszustände für die der simultanen Innen- und Außenoperationen bezüglich der Hashwerte doppelt gepuffert werden. Hinzu kommt, dass der Dual-Port-Rom-Speicher für die simultanen Konstantensuchverfahren sowohl von der Hash-Innenfunktionseinheit als auch von der Hash-Außenfunktionseinheit verwendet wird.

[0035] Mit Bezug auf Fig. 2 umfasst die Funktionseinheit **200** einen Doppelpaketnutzlast-Dateneingangspuffer **201**, der in dieser Instanz einen linken Paketrahmen **202** und einen rechten Paketrahmen **204** aufweist. Die von der Funktionseinheit empfangenen Eingangsdatennutzlasten, beispielsweise von Datenpaketen, die aus einem Netzwerk auf einem Chip empfangen werden, auf dem die Funktionseinheitsarchitektur implementiert ist, werden zwischen den Paketrahmen **202**, **204** des Doppelpaketnutzlast-Dateneingangspuffers **201** aufgeteilt, so dass ein Datenblock in den Puffer aufgenommen werden kann, während zu dem Datenstrom nachgeschaltet der andere Datenblock verarbeitet wird. Da die Fig. 2 ein Implementierungsbeispiel gemäß der vorliegenden Erfindung für die Verarbeitung von IPSec.-Paketen veranschaulicht, enthält die Architektur Hash-Funktionseinheiten für die MD5- und SHA1-Authentifizierungsprotokolle, die von IPSec. unterstützt werden. Übereinstimmend mit den MD5- und SHA1-Protokollen werden die Eingangsdaten-Nutzlasten in den Dual-Paketrahmen des Eingangsdatenpuffers **201** geladen, in 512-Bit-Datenblöcke aufgeteilt und falls notwendig, werden sie mit Bits [Padding] aufgefüllt (d. h. wenn der Datenblock weniger als 512 Bits aufweist) sowie abgespeichert, bevor er zur Verarbeitung an eine Hash-Innenfunktionseinheit weitergeleitet wird. Ein Multiplexer **206** steuert den Strom der 512-Bit-Datenblöcke aus den Paketrahmen des Eingangspuffers an eine Hash-Innenfunktionseinheit.

[0036] Die Hash-Ausgangszustände werden für den ersten Datenblock von jedem Paket auf pro Paketbasis benötigt. Die Hash-Ausgangszustände werden mittels einer Software auf Basis eines Authentifizierungsschlüssels und einigen Konstantenvorgaben auf Basis des HMAC-Algorithmus (Voraus-Hashwert) generiert, und zwar in Übereinstimmung mit den Spezifikationen für diese Algorithmenarten. Dies erfolgt normalerweise pro Authentifizierungsschlüssel. Alternativ dazu können die Ausgangszustände von den Vorgabekonstantenzuständen und dem Authentifizierungsschlüssel abgeleitet werden, wobei für jedes Paket die gleiche Hardware verwendet wird, für das eine Authentifizierung erforderlich ist.

[0037] Die Hash-Ausgangszustände bezüglich des inneren Hashwertes von einem vorhandenen Datenblock werden in einen Puffer **214** aufgenommen, der mit der/den Hash-Innenfunktionseinrichtung/en **210**, **212** in Verbindung steht. Die Hash-Ausgangszustände bezüglich des äußeren Hashwertes von diesem vorhandenen Datenblock werden in dem ersten Puffer **215** von den zwei Puffern **215**, **216** eingespeist (die auch als HMAC-Phasenpuffer bezeichnet werden), die mit der/den Hash-Außenfunktionseinrichtung/en **220**, **222** in Verbindung steht/stehen. Wenn die Hash-Ausgangszustände an die Hash-Innenfunktionseinheit für ein Verarbeiten des Datenblocks weitergeleitet worden sind, werden für diesen Block die äußeren Hash-Zustände in den zweiten Puffer **216** eingespeist, wobei die für das nächste, zu verarbeitende Paket inneren und äußeren Hash-Ausgangszustände wieder in die Puffer **214** bzw. **215** aufgenommen werden. Auf diese Weise bleibt das synchronisierende Gleichlaufverfahren bezüglich der inneren und äußeren Hashwertzustände für einen vorhandenen Datenblock aufrechterhalten, wobei die Hash-Ausgangszustände für simultane Hash-Innenoperationen als auch für Hash-Außenoperationen verfügbar sind. Ferner erlaubt das doppelte Puffern der Hash-Zustandswerte das Laden der Hash-Ausgangszustände für das zweite Paket, während das erste Paket noch verarbeitet wird, so dass von Paket zu Paket die Datenverarbeitung kontinuierlich fortgesetzt werden kann, wodurch die Leistungsfähigkeit und Verarbeitungseffizienz der Hash-Funktionseinheit maximiert wird.

[0038] Die Funktionseinheit **200** enthält ferner einen Dual-Port-Rom-Speicher **218**. Der Dual-Port-Rom-Speicher **218** ermöglicht ferner die parallelen inneren und äußeren Hash-Operationen, wobei das simultane Konstantensuchverfahren sowohl von der Hash-Innenfunktionseinheit als auch von der Hash-Außenfunktionseinheit zugelassen wird.

[0039] Der innere Hashwert wird in allen 512-Bit-Blöcken eines vorhandenen Datenpaketes ermittelt. Das Ergebnis des inneren Hashwertes weist eine Länge von 128 Bits für MD5 und für SHA1 eine Länge von 160 Bits auf. Das Ergebnis wird auf 512 Bits aufgefüllt und die äußere Hash-Einheit verarbeitet den einen 512-Bit-Datenblock auf Basis des Ergebnisses des inneren Hashwertes und errechnet so einen Digest. Ein Ausgabepuffer **230** speichert den Digest und gibt ihn über einen Multiplexer **232** aus.

Das Kollabieren der Verarbeitungsrunden im Mehrfach-Durchlaufrunden-Algorithmus

[0040] Von den zwei vom IPSec.-Protokoll unterstützten Algorithmen ist der HMAC-SHA1-96 ungefähr um 25 Prozent langsamer als der HMAC-MD5-96 in Bezug auf die Gesamtsumme aus den Computerberechnungsrunden. Ein Weg zur Verbesserung des HMAC-SHA1-96 in einer IPSec.-unterstützten Hardware-Implementierung besteht darin, mehrere Logikrunden in einen einzigen Taktzyklus herunterzubrechen, so dass folglich die für die HMAC-SHA1-96-Operation erforderliche Anzahl der Taktimpulse verringert wird. Der gleiche Lösungsansatz kann für jeden Mehrfach-Durchlaufrunden-Algorithmus angewendet werden. Jedoch kann das einfache Kollabieren der Logik bezüglich mehrerer Durchlaufrunden in einen einzigen Taktzyklus bei der Berechnung der kollabierten Logik für eine Leistungssteigerung eine Verzögerung verursachen und daher die maximale Taktfrequenz reduzieren.

[0041] Fig. 3 ist ein Zeitstudiendiagramm, das den zeitlich kritischen Pfad von der herkömmlichen Rundenlogik für den SHA1-Authentifizierungsalgorithmus veranschaulicht. Die Register a, b, c, d und e parken während der Durchlaufrunden die Zwischenzustände der Hashwerte. In dieser Figur werden diese Hash-Zwischenzustände für eine eindeutige Demonstrierung der Stoppstellen in den Logikpfaden doppelt markiert dargestellt. Die Pfade werden in der konkreten Konzeption zum gleichen Registersatz zurückgeführt, da die Rundenlogik 80mal erneut angewendet wird. Die "+" Symbole werden mit standardisierten Addiergliedern in Verbindung gebracht, die in Übertragsvorausberechnungsaddierwerken (CLAs = Carry Look-Ahead Adders) implementiert sind. Das Zeichen W; repräsentiert die eingehenden Nutzdaten. K_i stellt eine Konstante dar, die von dem ROM-Speicher erhalten wird, der bei den Authentifizierungscomputerberechnungen zum Einsatz kommt. In dieser Figur wird dargestellt, wie die zeitlich kritischen Pfade von den Registern b, c und d herstammen, eine nichtlineare Funktion durchlaufen (definiert durch die SHA1-Spezifikation), und wie die Addierglieder im Register a enden. Die Register b, c, d und e empfangen jeweils eine nichtkritische Eingabe (b empfängt a, usw.).

[0042] Fig. 4 ist ein Zeitstudiendiagramm, das den zeitlich kritischen Pfad von der kollabierten Rundenlogik für den SHA1-Authentifizierungsalgorithmus gemäß einer Ausführungsform der vorliegenden Erfindung veranschaulicht. Der SHA1-Algorithmus wird mit fünf Registern spezifiziert. Wie zuvor bereits dargestellt wurde, ist in jeder SHA1-Runde der Datenpfad von vier der fünf Registern nicht kritisch (zeitlich eingeschränkt). Gemäß der vorliegenden Erfindung sind die Register, die den kritischen Pfad aufweisen, in den nachfolgenden SHA1-Runden eine Ausweichalternative, so dass vier Register mit wertvollen Daten stets zur nächsten Runde weitergeleitet werden können, und zwar vor Vollendung des kritischen Pfades in der momentanen Runde. Wenn folglich zwei Runden des SHA1 zusammengefasst werden, ist die kritische Pfad-Computerberechnung für die zweite Runde unabhängig von jener der ersten Runde, da die empfangenden Register des kritischen Pfades der ersten Runde (d. h. Register a) nicht das Steuerregister des kritischen Pfades der zweiten Runde (d. h. Register e) bildet. Dieser Lösungsansatz demonstriert, wie zwei SHA1-Runden kollabierend zusammengelegt werden können, während für den zeitlich kritischen Pfad derselbe Verzögerungsumfang beibehalten wird, und wie durch das Wechseln des kritischen Pfades von Register zu Register während der Durchlaufrunden auf diese Weise die addierenden Operationen „umgangen" werden können.

[0043] Bei einer bevorzugten Ausführungsform werden die achtzig Runden einer SHA1-Durchlaufschleife zu vierzig Runden herunter gebrochen. Wie vorstehend beschrieben und veranschaulicht wurde, wird das Herunterbrechen von Durchlaufrunden mit einem einzigen Registersatz (die bevorzugte Ausführungsform weist fünf Register auf, wie in dem IPSec.-Protokoll definiert ist) und zwei Logikrunden ausgeführt. Es ist auch vorstellbar, dass die hierin beschriebenen Techniken der Erfindung des Weiteren angewendet werden können, um in einer SHA1-Schleife die Anzahl der SHA-1-Runden in zwanzig oder sogar noch weniger Runden herunter zu brechen.

Planungserstellung der Additionen

[0044] Wie bereits vorstehend beschrieben, spezifizieren sowohl der MD5- als auch der SHA1-Algorithmus, dass die finalen Hash-Zustandswerte [Endzustände] von jedem 512-Bit-Block mit den anfänglichen Hash-Zustandswerten [Ausgangszuständen] zusammen addiert werden müssen. Diese Ergebnisse werden anschließend als Ausgangszustandswerte für den nächsten 512-Bit-Block verwendet. Beim MD5 müssen die Werte

von vier 32-Bit-Registerpaaren und beim SHA1 müssen 5 Paare addiert werden. Wenn man annimmt, dass jede 32-Bit-Addition einen Taktzyklus beansprucht, würde eine typische Hardware-Implementation beim MD5 vier extra Zyklen und beim SHA1 fünf extra Zyklen anwenden.

[0045] Wie vorstehend mit Bezug auf die Fig. 3 und Fig. 4 erläutert wurde, wird sowohl beim MD5 als auch beim SHA1 nur ein Registerzustandswert in jeder Durchlaufrunde erneut berechnet. Die restlichen Zustandsregister verwenden verschobene oder nicht verschobene Inhalte aus den benachbarten Registern. Daher werden die finalen Hash-Zustandswerte nicht in der finalen Runde erzeugt, sondern vielmehr in den letzten vier konsekutiven MD5-Runden bzw. letzten fünf konsekutiven SHA1-Runden. Die vorliegende Erfindung nutzt diese vorausgesetzte Annahme aus, indem sie eine Architektur und eine Logik bereitstellt, welche die Planungserstellung der Additionen ermöglicht, sobald der finale Hash-Zustandswert verfügbar ist und umgeht die Computerberechnungszeiten im Hintergrund der Rundenoperationen komplett. Dies wird in den nachfolgenden Aufstellungstabellen veranschaulicht, in denen "T1" einen Taktzyklus und „rnd 1" eine Rundenoperation darstellt usw. = Ti/rndi. Die anfänglichen Hash-Ausgangszustände werden mit ia, ib, ic, id und ie repräsentiert. Die parallelen Operationen sind in der gleichen Spalte aufgelistet.

MD5-Durchlaufrunden

T1	T2	T3	---------	T61	T62	T63	T64	T1
rnd 1	rnd 2	rnd 3		rnd 61	rnd 62	rnd 63	rnd 64	rnd 1
					a+ia	d+id	c+ic	b+ib

Original SHA1-Durchlaufrunden

T1	T2	T3	---------	T77	T78	T79	T80	T1
rnd 1	rnd 2	rnd 3		rnd 77	rnd 78	rnd 79	rnd 80	rnd 1
				e+ie	d+id	c+ic	b+ib	a+ia

Kollabierte SHA-Durchlaufrunden

T1	T2	T3	---------	T38	T39	T40	T1
rnd 1	rnd 2	rnd 3		rnd 38	rnd 39	rnd 40	rnd 1
					e+ie	d+id c+ic	b + ib a+ia

[0046] Bei einer Ausführungsform der Erfindung kann in einem einzigen Taktzyklus eine Vielzahl von Additionen mit den finalen Hash-Zustandswerten ausgeführt werden. Ein Ausführungsbeispiel wird in der "kollabierten SHA1-Tabelle" gezeigt, in der in nur drei Taktzyklen – T39, T40 und T1 – der nächstfolgenden Schleife die fünf Additionen durchgeführt werden. Die Fachleute auf diesem Gebiet realisieren, dass in einem Taktzyklus – in Übereinstimmung mit den Grundsätzen der hierin beschriebenen Erfindung – mehr als zwei Additionen parallel durchgeführt werden können. Darüber hinaus sei angemerkt, dass – wie in den Tabellen veranschaulicht

wird – gemäß der vorliegenden Erfindung dieser Aspekt sowohl auf kollabierte als auch auf nicht herunter gebrochene Mehrfach-Durchlaufrunden-Algorithmen anwendbar ist. Die Implementierung von diesem erfindungsgemäßen Aspekt in Verbindung mit einem herunter gebrochenen Mehrfach-Durchlaufrunden-Algorithmus ist insbesondere von Vorteil, da das Umgehen von Additionsschritten immer mehr von Bedeutung ist, wenn dabei die Anzahl der Durchlaufrunden verringert werden kann. Additionen, die nicht in der Art und Weise gemäß diesem Aspekt der vorliegenden Erfindung verborgen werden können, würden sogar einen noch größeren programmtechnischen Befehlsverarbeitungszeitaufwand für eine kollabierte Rundenimplementierung darstellen, als wie wenn sie für eine Implementierung mit einer höheren Anzahl von Durchlaufrunden erforderlich sind.

Die Konzeption der Rundenlogik

[0047] Fig. 5 ist ein Blockdiagramm auf höherer Ebene von einer SHA1-Hash-Funktionseinheit, das die wesentlichen Elemente eines kollabierenden Rundenlogikaufbaus gemäß einer Ausführungsform der vorliegenden Erfindung veranschaulicht, das zu dem zeitlich kritischen Pfadstudien-Diagramm der Fig. 4 konform ist. Das Design macht Gebrauch von den Übertragsspeicheraddierwerken (CSA; eine Verzögerung entspricht einem 1-Bit-Addierglied), indem es deren Kapazitäten in Bezug auf das Zusammenaddieren von vielen Quantitäten vorteilhaft ausnutzt. Die CSAs addieren effizient viele Quantitäten zusammen, um Teilprodukte zu erzeugen, die nicht reproduziert werden sollen. Zwei in der Figur dargestellte, umfangreiche Additionsmodule – das Add5to1-Addiermodul und das Add4to1-Addiermodul – wenden jeweils mehrere CSA-Stufen an, denen ein Übertragsvorausberechnungsaddierwerk (CLA) folgt, wie es in Bezug auf Fig. 6 noch nachstehend in den Einzelheiten beschrieben wird.

[0048] Die Hash-Funktionseinheit weist fünf Register – A, B, C, D und E – auf. Der anfängliche Hash-Ausgangszustandswert in Register A (a_1) durchläuft einen 5-Bit-Ringschieber und wird mit dem anfänglichen Hash-Ausgangszustandswert in Register E (e_1), dem Nutzlastdatenwert (W_i), einer Konstante (K_i) und dem Ergebnis eines Funktionswertes (F_t) der Hash-Ausgangszustandswerte in den Registern B, C und D mittels eines Add5to1-Addiermoduls addiert, das mit den Addiergliedern des CSA und CLA aufgebaut ist. Der anfängliche Hash-Ausgangszustandswert in Register D (d_1) wird mit dem Nutzlastdatenwert (W_{i+1}), einer Konstante (K_{i+1}) und dem Ergebnis eines Funktionswertes (F_t) der Hash-Ausgangszustandswerte in den Registern A, B (die einen 30-Bit-Ringschieber durchlaufen) und C mittels eines Add4to1-Addiermoduls addiert, das mit den Addiergliedern des CSA und CLA aufgebaut ist.

[0049] Die Addiermodule schließen mit einem Addierglied des Übertragsvorausberechnungsaddierwerks (CLA = Carry Look-Ahead Adder) ab. Die Summe eines jeden Addiermoduls wird mittels eines CLA-Addiergliedes addiert, um eine Endsumme für die Durchlaufrunde zu erzeugen und zu reproduzieren, die anschließend in das Register A für die nächste Runde zurückgeführt wird. Die zeitlich kritischste Eingabe dieser beiden Module muss nur die letzte CLA-Phase durchlaufen.

[0050] Fig. 6 ist ein Blockdiagramm auf einer unteren Ebene, das die Einzelheiten von der Planungserstellung der Additionen innerhalb des Rundenlogikaufbaus der Fig. 5 veranschaulicht. Das Abziehen von zwei Runden der SHA1-Operation führt zu folgender Pfadbeschleunigung:

$$S = ((a <<< 5) + f(b, c, d) + e + w_i + k_i) <<< 5 + f(a, b <<< 30, c) + d + W_{i+1} + k_{i+1},$$

wobei a, b, c, d, e, w und k die 32-Bit-Größen sind. Gemäß der in Fig. 5 dargestellten Ausführungsform der vorliegenden Erfindung wird diese Operation in zwei Schritten ausgeführt.

[0051] Schritt **1** wendet das Add5to1-Addiermodul an zur Generierung von:

$$Si = (a <<< 5) + f(b, c, d) + e + w + k.$$

[0052] Schritt **2** wendet das Add4to1-Addiermodul und ein 32-Bit-Übertragsvorausberechnungsaddierwerk (CLA) an zur Generierung von:

$$S = Si <<< 5 + f(a, b <<< 30, c) + d + w_{i+1} + k_{i+1}.$$

[0053] In jedem Schritt werden Addierglieder des Übertragsspeicheraddierwerks (CSA) angewendet, um 3 bzw. 2 Eingabereduzierungen durchzuführen, bevor das 32-Bit-CLA zum Einsatz kommt. Die gesamte Verzögerungszeit entspricht den zwei 32-Bit-CLA-Verzögerungen plus einer 32-Bit-CSA-Verzögerung plus des Ver-

zögerungszeitraums für den Funktionswert „f" in Bezug auf den zeitlich kritischsten Pfad. Nachdem über die CSAs [Übertragsspeicheraddierwerke] alle Reduzierungen beendet worden sind, ergeben Schritt **1** und Schritt **2**:

S = (A + B) <<< 5 + C + D.

[0054] Erfindungsgemäße Implementierungen, die diese Logikkonzeption in einer Authentifizierungsfunktionseinheit mit dem HMAC-SHA1-Algorithmus des IPSec.-Protokolls anwenden, wobei das Herunterbrechen von den herkömmlichen 80 SHA-1-Runden in 40 Durchlaufrunden, das Umgehen von Additionen und das Befehlsverknüpfen der Innen- und Außenschleifen zum Einsatz kommt, haben es ermöglicht, dass der erweiterte HMAC-SHA1 nahezu in der gleichen Zeit wie der herkömmliche SHA1 durchgeführt werden kann.

Zusammenfassung

[0055] Obgleich die vorstehende Erfindung in den verschiedenen Einzelheiten zum Zwecke eines klaren und leichteren Verständnisses beschrieben worden ist, werden die Fachleute auf diesem Gebiet zustimmen, dass verschiedene Adaptionen und Modifikationen zu den vorstehend erläuterten, bevorzugten Ausführungsformen vorgenommen werden können, ohne vom Schutzumfang der Erfindung abzuweichen. Auch wenn die vorliegende Erfindung in erster Linie beispielsweise in Zusammenhang mit dem IPSec.-Protokoll beschrieben worden ist, können die erfindungsgemäßen Grundsätze auch auf allgemeine, mehrfach durchlaufende Rundenauthentizitätsalgorithmen angewendet werden, ganz gleich, ob sie mit kryptografischen Betriebsvorgängen zum Einsatz kommen oder nicht. Daher sollten die beschriebenen Ausführungsformen nur veranschaulichend und nicht restriktiv zur Kenntnis genommen werden, wobei die Erfindung nicht auf die hierin aufgeführten Einzelheiten eingeschränkt werden darf. Sie ist in den anhängenden Patentansprüchen mit deren vollem Schutzumfang definiert.

Patentansprüche

1. Authentifizierungseinrichtung für einen Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung (SHA1, MD5, HMAC-MD5, HMAC-SHA1), welche umfasst:
– eine Hash-Einrichtung, die zur Implementierung einer Zufallszeichen-Rundenlogik [Hash-Rundenlogik] für den Mehrfachdurchlaufrunden-Authentifizierungs-Algorithmus (SHA1, MD5, HMAC-MD5, HMAC-SHA1) konfiguriert ist, wobei das Implementieren der Hash-Rundenlogik mindestens ein Additionsmodul einschließt, welches umfasst:
– eine Mehrzahl von Übertragsspeicheraddierwerken (CSA) für die Berechnung von Teilprodukten, und
– ein Übertragsvorausberechnungsaddierwerk (CLA) für die Berechnung und Reproduktion eines Endbetrages;
dadurch gekennzeichnet, dass
– die Hash-Einrichtung ferner zur Durchführung von Additionsberechnungen konfiguriert ist, die End- und Ausgangszustände von Hashwerten umfassen, welche zu den Rundenoperationen parallel verlaufen.

2. Authentifizierungseinrichtung gemäß Anspruch 1, wobei der Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung ein MD5-Algorithmus ist.

3. Authentifizierungseinrichtung gemäß Anspruch 1, wobei der Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung ein SHA1-Algorithmus ist.

4. Authentifizierungseinrichtung gemäß Anspruch 3, wobei die Hash-Rundenlogik-Implementierung folgendes umfasst:
– fünf Hash-Zustandsregister (A, B, C, D, E);
– einen 5-Bit-Ringschieber (<<< 5);
– ein Add5to1-Addiermodul (Add5to1) mit einer Mehrzahl von Übertragsspeicheraddierwerken (CSA) und einem Übertragsvorausberechnungsaddierwerk (CLA);
– einen 30-Bit-Ringschieber (<<< 30); und
– ein Add4to1-Addiermodul (Add4to1) mit einer Mehrzahl von Übertragsspeicheraddierwerken (CSA) und einem Übertragsvorausberechnungsaddierwerk (CLA).

5. Authentifizierungseinrichtung gemäß Anspruch 3, wobei die Hash-Rundenlogik-Implementierung folgendes umfasst:
– fünf Hash-Zustandsregister (A, B, C, D, E);

– einen kritischen und vier nichtkritische Datenpfade, die den fünf Registern (A, B, C, D, E) zugeordnet sind;
– wobei die sukzessiven, ersten und zweiten Runden des SHA1-Authentifizierungsalgorithmus so zusammengefasst werden, dass der kritische Datenpfad der zweiten Runde von dem kritischen Datenpfad der ersten Runde unabhängig ist.

6. Authentifizierungseinrichtung gemäß Anspruch 1, die außerdem für einen mehrschleifigen Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung (SHA1, MD5, HMAC-MD5, HMAC-SHA1) ausgelegt ist, wobei die Authentifizierungseinrichtung ferner aufweist:
– eine erste Hash-Rundenlogik für den Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung in einer Innen-Hash-Einrichtung (**210**, **212**);
– eine zweite Hash-Rundenlogik für den Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung in einer Außen-Hash-Einrichtung (**220**, **222**);
– einen Doppelpaketnutzlast-Dateneingabepuffer (**201**), der für ein Laden eines neuen Datenblocks konfiguriert ist, während ein anderer Datenblock in der Innen-Hash-Einrichtung (**210**, **212**) verarbeitet wird;
– eine Hash-Ausgangszustandseingabe-Pufferkonfiguration (**214**) für ein Laden der Hash-Ausgangszustände in die Innen- und Außen-Hash-Einrichtungen (**210**, **212**, **220**, **222**) für simultane Innen- und Außen-Hashwert-Operationen, sowie
– einen Dual-Port-ROM-Speicher (**218**), der für simultane Konstantensuchverfahren sowohl für die Innen- als auch für die Außen-Hash-Einrichtungen (**210**, **212**, **220**, **222**) konfiguriert ist.

7. Authentifizierungseinrichtung gemäß Anspruch 6, wobei der mehrschleifige Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung ein HMAC-MD5-Algorithmus ist.

8. Authentifizierungseinrichtung gemäß Anspruch 6, wobei der mehrschleifige Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung ein HMAC-SHA1-Algorithmus ist.

9. Authentifizierungseinrichtung gemäß Anspruch 8, wobei mindestens eine der Innen- und Außen-Hash-Einrichtungen (**210**, **212**, **220**, **222**) für das Implementieren einer Hash-Rundenlogik konfiguriert ist, welche folgendes umfasst:
– fünf Hash-Zustandsregister (A, B, C, D, E);
– einen kritischen und vier nichtkritische Datenpfade, die den fünf Registern zugeordnet sind;
– wobei die sukzessiven, ersten und zweiten Runden des SHA1-Authentifizierungsalgorithmus so zusammengefasst werden, dass der kritische Datenpfad der zweiten Runde von dem kritischen Datenpfad der ersten Runde unabhängig ist.

10. Authentifizierungseinrichtung gemäß Anspruch 8, wobei die Hash-Rundenlogik-Implementierung folgendes umfasst:
– fünf Hash-Zustandsregister (A, B, C, D, E);
– einen 5-Bit-Ringschieber (<<< 5);
– ein Add5to1-Addiermodul (Add5to1) mit einer Mehrzahl von Übertragsspeicheraddierwerken (CSA) und einem Übertragsvorausberechnungsaddierwerk (CLA);
– einen 30-Bit-Ringschieber (<<< 30); und
– ein Add4to1-Addiermodul (Add4to1) mit einer Mehrzahl von Übertragsspeicheraddierwerken (CSA) und einem Übertragsvorausberechnungsaddierwerk (CLA).

11. Verfahren für die Authentifizierung von Daten, die über ein Computernetzwerk übermittelt worden sind, welches die folgenden Schritte umfasst:
– Empfangen eines Datenpaketstroms;
– Aufteilen des Datenpaketstroms in Datenblöcke mit festgelegter Größe, und
– Verarbeiten der Datenblöcke mit festgelegter Größe unter Verwendung einer Mehrfachdurchlauf-Authentifizierungseinrichtung; dadurch gekennzeichnet, dass
– eine Authentifizierungseinrichtung, die eine Hash-Rundenlogik für einen Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung (SHA1, MD5, HMAC-MD5, HMAC-SHA1) implementiert, so konfiguriert ist, dass Additionsberechnungen, welche die Hash-End- und Ausgangszustände umfassen, parallel zu den Rundenoperationen durchgeführt werden können, und wobei
– das Implementieren der Hash-Rundenlogik mindestens ein Additionsmodul einschließt, das
– eine Mehrzahl von Übertragsspeicheraddierwerken (CSA) für die Berechnung von Teilprodukten umfasst, und
– ein Übertragsvorausberechnungsaddierwerk (CLA) für die Berechnung und Reproduktion eines Endbetrages.

12. Verfahren gemäß Anspruch 11, wobei die Hash-Rundenlogik die folgenden Schritte umfasst:
– Durchführen einer logischen 5-Bit-Verschiebung der Daten (a_1) aus einem ersten Schieberegister (A);
– Addieren eines Hash-Ausgangszustands (e_1) in ein zweites Register (E), einen ersten Nutzlastdatenblock (W_i), eine erste Konstante (K_i), und das Ergebnis einer ersten Funktion (F_t) aus den Hash-Ausgangszuständen in den dritten, vierten und fünften Zusatzregistern (B, C, D) mit einem Add5to1-Addiermodul (Add5to1), das eine Mehrzahl von Übertragsspeicheraddierern (CSA) und einen Übertragsvorausberechnungsaddierer (CLA) aufweist;
– Durchführen einer logischen 30-Bit-Verschiebung der Daten aus dem dritten Zusatzregister (B); und
– Addieren des Hash-Ausgangszustands in dem fünften Zusatzregister (D) mit einem zweiten Nutzlastdatenblock (W_{i+1}), einer zweiten Konstante (K_{i+1}), und das Ergebnis einer Funktion (F_t) aus den Hash-Ausgangszuständen in den ersten und vierten Registern (A, C) und des verschobenen Hash-Zustands des dritten Registers (B) mit einem Add4to1-Addiermodul (Add4to1), das eine Mehrzahl von Übertragsspeicheraddierern (CSA) und einen Übertragsvorausberechnungsaddierer (CLA) aufweist.

13. Verfahren gemäß Anspruch 11, welches ferner die folgenden Schritte umfasst:
– Verarbeiten der Datenblöcke mit festgelegter Größe unter Verwendung einer mehrschleifigen Mehrfachdurchlauf-Authentifizierungseinrichtung mit einem Hash-Kernelement, das eine Innen-Hash-Einrichtung (**210**, **212**) und eine Außen-Hash-Einrichtung (**220**, **222**) aufweist,
– wobei die Authentifizierungseinrichtung ferner so konfiguriert ist, um
– Hashwert-Operationen der Innen-Hash- und Außen-Hash-Einrichtungen (**210**, **212**, **220**, **222**) kanalisieren zu können,
– Kollabieren und Neuarrangieren der Mehrfachdurchlaufrundenlogik zur Reduzierung der Runden im Hinblick auf die Hashwert-Operationen.

14. Verfahren gemäß Anspruch 13, wobei das Kanalisieren die Durchführung einer Außenhashwert-Operation für eine Datennutzlast parallel zu einer Innen-Hashwert-Operation von einer zweiten Datennutzlast in einem Paketstrom umfasst, mit dem die Authentifizierungseinrichtung gespeist ist.

15. Verfahren gemäß Anspruch 14, wobei ein Doppelpaketnutzlast-Dateneingabepuffer (**201**) in Bezug auf die Innen-Hash-Einrichtung (**210**, **212**) zum Einsatz kommt.

16. Verfahren gemäß Anspruch 15, wobei die Hash-Ausgangszustände für die Hashwert-Operationen bezüglich simultaner Innen-Hashwert- und Außen-Hashwert-Operationen doppelt gepuffert werden.

17. Verfahren gemäß Anspruch 16, wobei simultane Konstantensuchverfahren von einem Dual-Port-ROM-Speicher (**218**) mit beiden Innen- und Außen-Hash-Einrichtungen (**210**, **212**, **220**, **222**) durchgeführt werden.

18. Verfahren gemäß Anspruch 13, wobei der mehrschleifige Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung ein MD5-Algorithmus ist.

19. Verfahren gemäß Anspruch 13, wobei der mehrschleifige Mehrfachdurchlaufrunden-Algorithmus zur Authentifizierung ein SHA1-Algorithmus ist.

20. Verfahren gemäß Anspruch 19, wobei die Additionsablauffolge die folgenden Schritte umfasst:
– Durchführen einer logischen 5-Bit-Verschiebung der Daten (a_1) aus einem ersten Schieberegister (A);
– Addieren eines Hash-Ausgangszustands (e_1) in ein zweites Register (E), einen ersten Nutzlastdatenblock (W;), eine erste Konstante (K_i), und das Ergebnis einer Funktion (F_t) aus den Hash-Ausgangszuständen in den dritten, vierten und fünften Zusatzregistern (B, C, D) mit einem Add5to1-Addiermodul (Add5to1), das eine Mehrzahl von Übertragsspeicheraddierern (CSA) und einen Übertragsvorausberechnungsaddierer (CLA) aufweist;
– Durchführen einer logischen 30-Bit-Verschiebung der Daten aus dem dritten Zusatzregister (B); und
– Addieren des Hash-Ausgangszustands in dem fünften Zusatzregister (D) mit einem zweiten Nutzlastdatenblock (W_{i+1}), einer zweiten Konstante (K_{i+1}), und das Ergebnis einer Funktion (Ft) aus den Hash-Ausgangszuständen in den ersten und vierten Registern (A, C) und des verschobenen Hash-Zustands des dritten Registers (B) mit einem Add4to1-Addiermodul (Add4to1), das eine Mehrzahl von Übertragsspeicheraddierern (CSA) und einen Übertragsvorausberechnungsaddierer (CLA) aufweist.

21. Verfahren gemäß Anspruch 19, wobei das Kollabieren und Neuarrangieren der Mehrfachdurchlaufrunden-Logik die folgenden Schritte umfasst:

– Bereitstellen von fünf Hash-Zustandsregistern (A, B, C, D, E) und
– Bereitstellen von Datenpfaden aus den fünf Zustandsregistern (A, B, C, D, E) dergestalt, dass in jeder SHA1-Runde vier von den fünf Datenpfaden aus den Registern zeitlich nicht kritisch koordiniert sind;
– wobei die sukzessiven, ersten und zweiten Runden des SHA1-Authentifizierungsalgorithmus so zusammengefasst werden, dass der kritische Datenpfad der zweiten Runde von dem kritischen Datenpfad der ersten Runde unabhängig ist.

Es folgen 6 Blatt Zeichnungen

Anhängende Zeichnungen

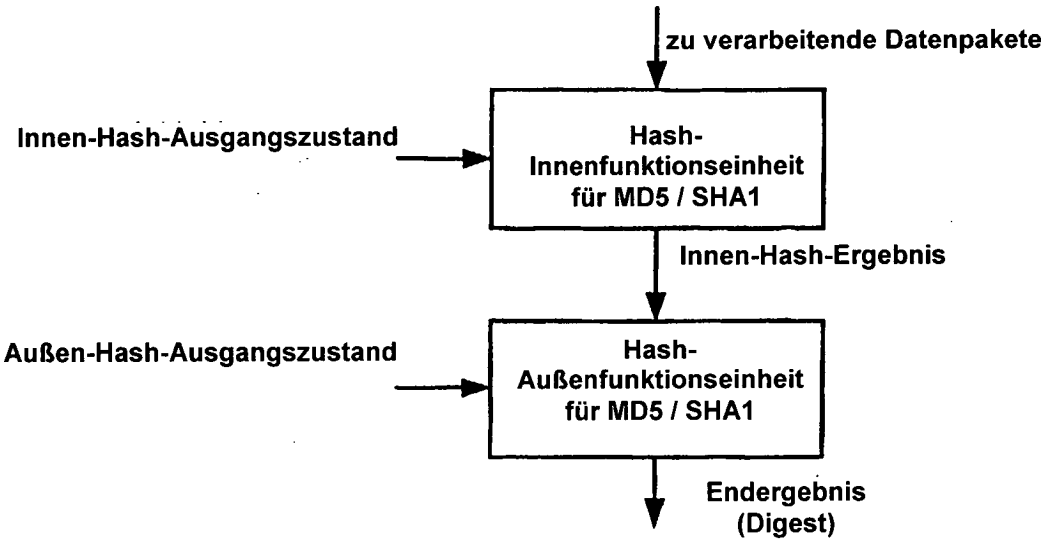

FIG. 1

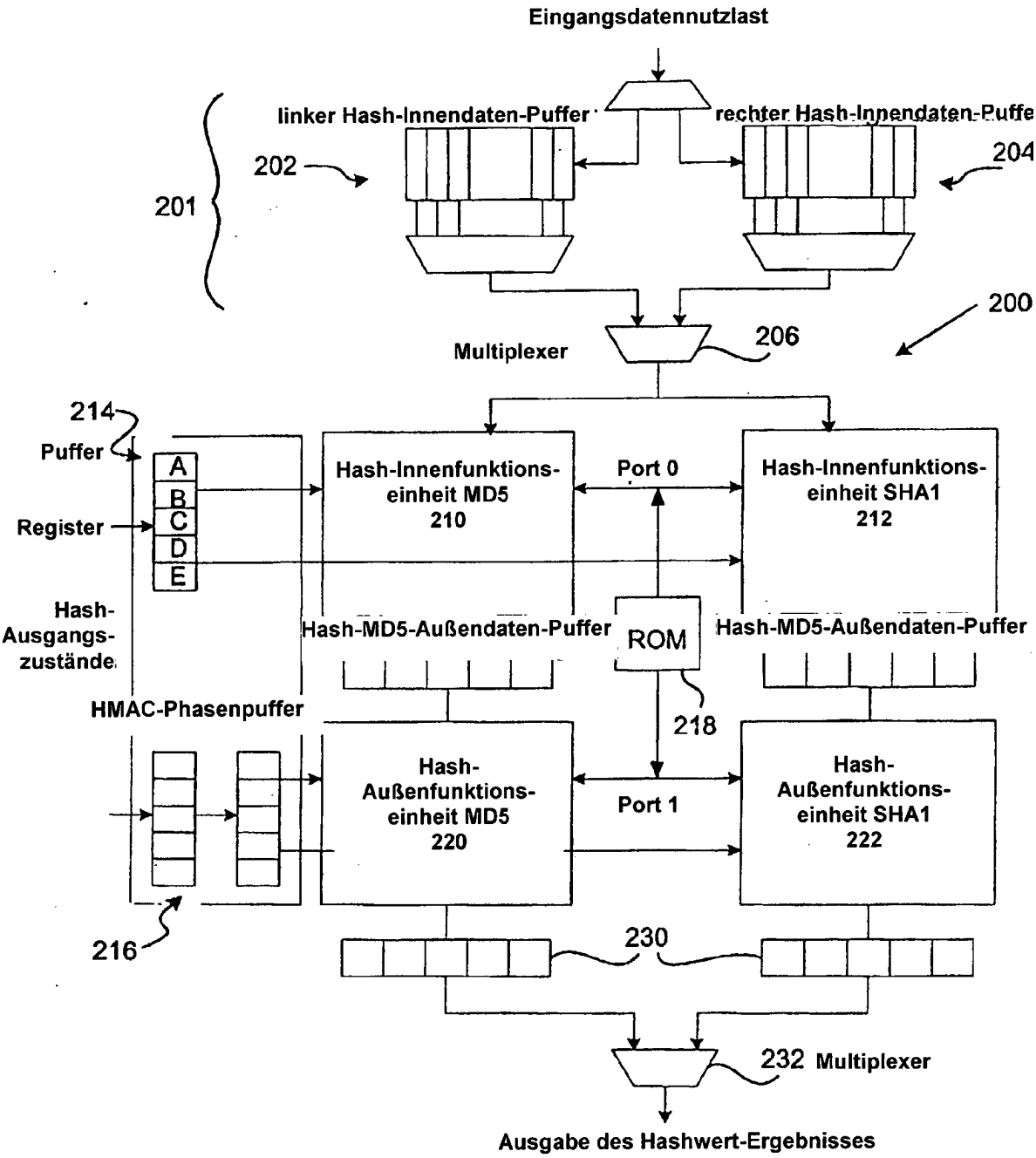

FIG. 2

200 = Funktionseinheit

201 = Doppelpaketnutzlast-Dateneingangspuffer
218 = Dual-Port-Rom-Speicher
230 = Ausgabepuffer

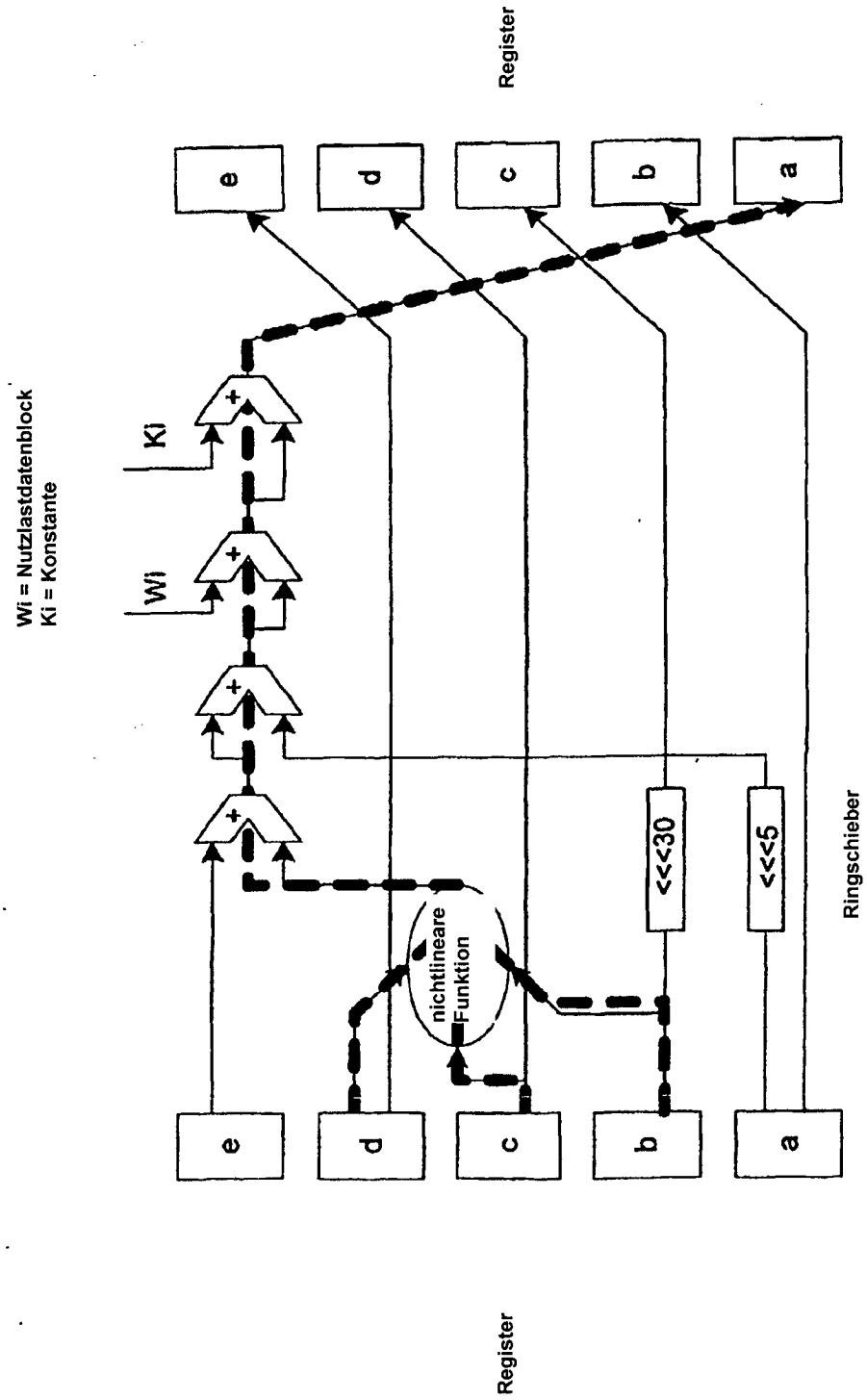
Register
e
d
c
b
a
Ki
Wi
Wi = Nutzlastdatenblock
Ki = Konstante
nichtlineare Funktion
<<<30
<<<5
Ringschieber
Register

FIG. 3

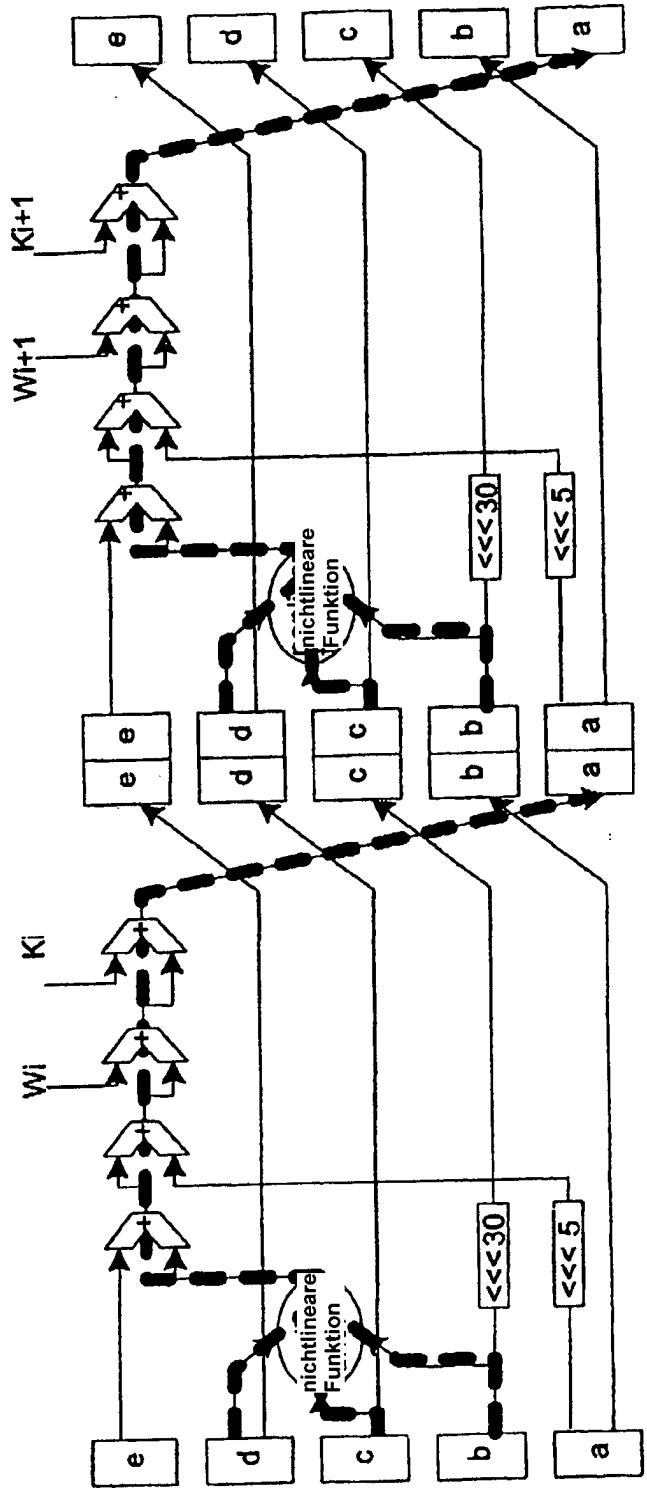

FIG. 4

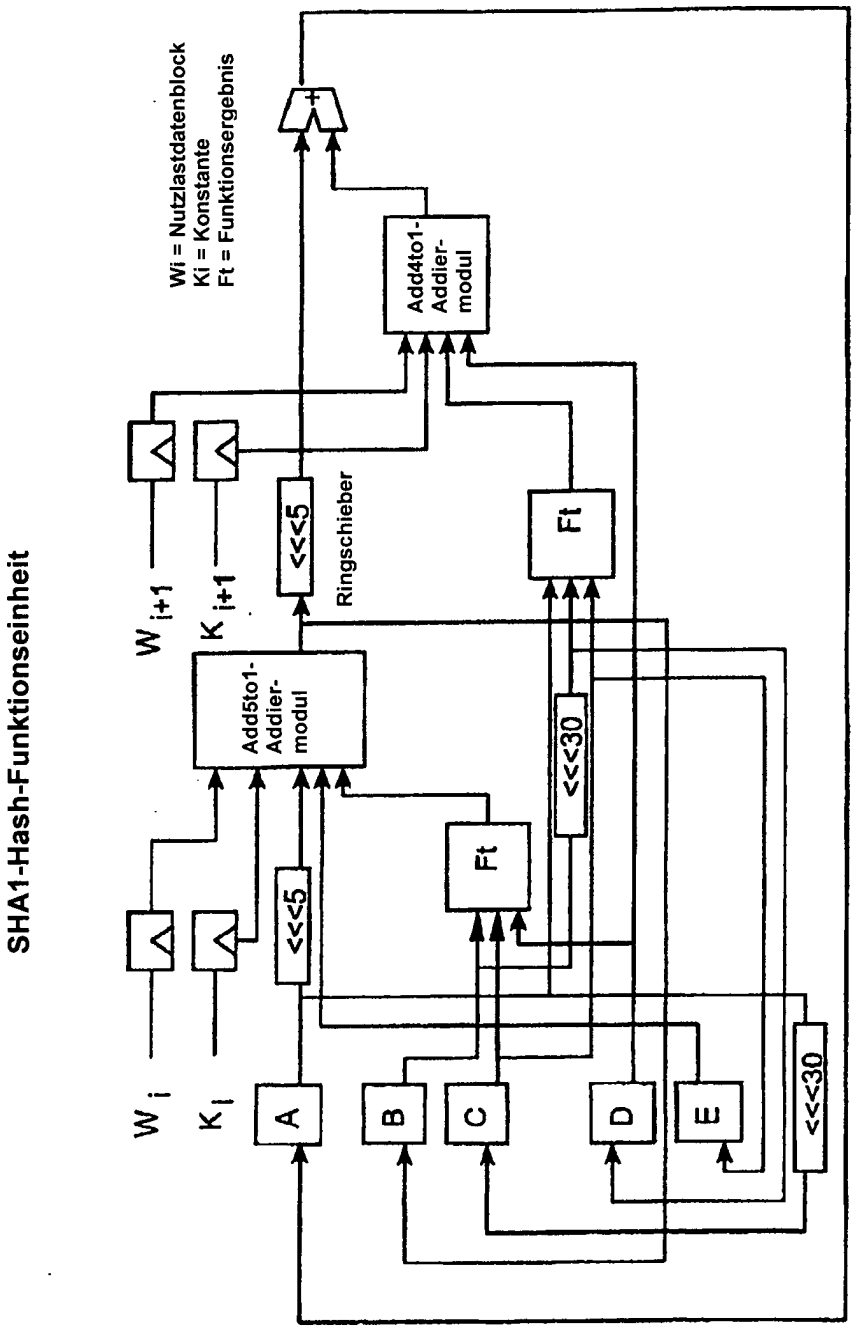
SHA1-Hash-Funktionseinheit
Wi = Nutzlastdatenblock
Ki = Konstante
Ft = Funktionsergebnis
Add4to1-Addier-modul
Add5to1-Addier-modul
Ringschieber
<<<5
<<<30
Ft
A
B
C
D
E
W i
K i
W i+1
K i+1
Ringschieber

FIG. 5

ai	=	Register a x bzw. (A,B,C,D,E)ei
ft	=	Funktionswert
ki	=	Konstante x
shift	=	Verschiebung
S	=	Schrittpfad
Si	=	Schritt x
wi	=	Nutzlastdatenblock x
CLA	=	Übertragsvorausberechnungsaddierwerk
CSA	=	Übertragsspeicheraddierwerk

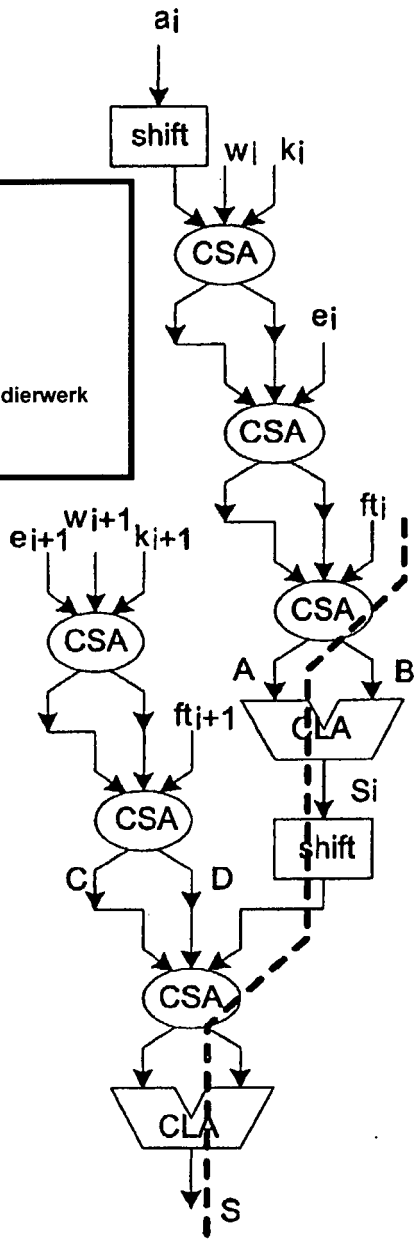

FIG. 6