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

(10) **DE 699 20 779 T2** 2005.05.25

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 345 333 B1**
(21) Deutsches Aktenzeichen: **699 20 779.7**
(96) Europäisches Aktenzeichen: **03 013 368.0**
(96) Europäischer Anmeldetag: **27.05.1999**
(97) Erstveröffentlichung durch das EPA: **17.09.2003**
(97) Veröffentlichungstag
der Patenterteilung beim EPA: **29.09.2004**
(47) Veröffentlichungstag im Patentblatt: **25.05.2005**

(51) Int Cl.[7]: **H03M 13/00**

(30) Unionspriorität:

14619198	**27.05.1998**	**JP**
14619298	**27.05.1998**	**JP**
34601698	**04.12.1998**	**JP**

(84) Benannte Vertragsstaaten:
DE, FR, GB, IT

(73) Patentinhaber:
NTT Mobile Communications Network Inc., Tokio/Tokyo, JP

(74) Vertreter:
HOFFMANN & EITLE, 81925 München

(72) Erfinder:
Kawahara, Toshio, Yokosuka-shi, Kanagawa 238-0315, JP; Miki, Toshio, Yokohama-shi, Kanagawa 236-0057, JP; Hotani, Sanae, Yokohama-shi, Kanagawa 232-0025, JP

(54) Bezeichnung: **Fehlerschutzverfahren und -vorrichtung**

Beschreibung

Technisches Gebiet

[0001] Die vorliegende Erfindung betrifft ein Fehlerschutzverfahren und eine Fehlerschutzvorrichtung, die eine Vielfalt von Daten vor Signalfehlern in dem Prozess einer Übertragung schützt.

Stand der Technik

[0002] Fehlerschutz wurde in Daten durchgeführt, wie etwa Sprache und Bildern, um sie vor Signalfehlern zu schützen, während sie übertragen werden. Es wurde eine Vielfalt von Fehlerschutzverfahren eingesetzt, wie etwa PDC-Halbrate, Twin-VQ und dergleichen.

[0003] Außerdem wurden Fehlerschutzverfahren des Standes der Technik auf Rahmen fixierter Länge mit der gleichen Zahl von Bits pro Rahmen oder Rahmen quasi-fixierter Länge angewendet, wo nur Rahmen einer begrenzten Vielfalt von Bitlängen herauszufinden waren.

[0004] In dem Fehlerschutzverfahren des Standes der Technik wurde Fehlerempfindlichkeit für jeden Parameter (inkludierend Bits), die den Rahmen ausmachen, untersucht, und in dem Fall von Parametern mit hoher Empfindlichkeit wurde Fehlerschutz eines spezifizierten Typs (z. B. die Hinzufügung eines Fehlerschutzcodes oder Fehlerkorrekturkodierung) in Rahmen, die Parameter mit hoher Empfindlichkeit enthalten, zur Zeit einer Übertragung der Rahmen durchgeführt. Hier ist die Fehlerempfindlichkeit des Parameters ein Ausmaß, das die Ergebnisse einer Kodierung verschlechtern würde, die auf der empfangenden Seite empfangen werden, in dem Fall, wo ein einzelner Bitfehler in dem Prozess einer Übertragung des Parameters aufgetreten ist.

[0005] Fig. 13 zeigt ein Beispiel der Konfiguration einer Fehlerschutzvorrichtung des Standes der Technik. In dieser Figur nimmt eine Klassensortiervorrichtung **401** den Rahmen, der vor einem Fehler zu schützen ist, und sortiert die Parameter des Rahmens in Klassen gemäß ihren jeweiligen Ausmaßen von Fehlerempfindlichkeit.

[0006] In dem Fall dieser Fehlerschutzvorrichtung wird das Fehlerschutzverfahren, nämlich entweder Fehlerkorrekturkodierung bei einem spezifizierten Kodierungsverhältnis oder die Hinzufügung eines spezifizierten Fehlererfassungscodes für jede Klasse bestimmt.

[0007] Ein Klassendaten-Fehlerschutz-Verarbeitungsteil **403** implementiert die Fehlerkorrekturkodierung oder fügt einen Fehlererfassungscode hinzu gemäß dem Fehlerschutzverfahren, das als Reaktion auf die verschiedenen Klassen bestimmt wird, in Bezug auf die verschiedenen Parameter, die in die Klassen klassifiziert sind. Speziell wird Faltung für Fehlerkorrekturkodierung ausgeführt, und zyklische Redundanzkodierung wird ausgeführt, um einen Fehlererfassungscode zu erzeugen. Des weiteren führt eine Ausgabevorrichtung **404** Verschachtelung und dergleichen in den Daten durch, die Fehlerkorrekturkodierung unterzogen werden oder denen ein Fehlercodeerfassungscode hinzugefügt wird, und sendet die Daten zu dem Empfänger.

[0008] Übrigens wurde in dem Fall der Fehlerkorrekturvorrichtung des Standes der Technik zuvor festgesetzter Fehlerschutz für jede Klasse ausgeführt. Somit hat die Fehlerkorrekturvorrichtung des Standes der Technik die Nachteile von geringer Flexibilität in Schutzverfahren und schlechter allgemeiner Anwendbarkeit.

[0009] Außerdem hat die Fehlerschutzvorrichtung des Standes der Technik auch den Nachteil, dass sie nicht in der Lage ist, einen Rahmen variabler Länge zu schützen, dessen Zahl von Bestandteilbits mit der Zeit schwankt, da sie nur die Rahmen fixierter Länge oder Rahmen quasi-fixierter Länge schützen könnte, wie oben erwähnt.

[0010] Zusätzlich zu dem oben erwähnten Stand der Technik offenbart die Veröffentlichung mit dem Titel "Optimierung der Empfangsqualität durch angepassten Fehlerschutz bei DAB" von C. Beck und veröffentlicht in ITG-Fachberichte, Bd. 118, 18. Februar 1992 auf den Seiten 102–112 ein dynamisches ungleiches Fehlerschutz-(DUEP)-Schema für MUSICAM-Datenrahmen, umfassend die Schritte eines Sortierens der Rahmen in der Reihenfolge einer abnehmenden Fehlerempfindlichkeit, eines Teilens jedes Rahmens in Unterblöcke mit abnehmenden Fehlerkorrekturanforderungen, wodurch für jeden Datenrahmen ein angepasstes Fehlerschutzprofil dynamisch aufgebaut wird, eines Durchführens einer Fehlerkorrekturkodierung und eines Übertragens des Fehlerschutzprofils und des codierten Rahmens.

[0011] DE-A-196 05 418 offenbart ein adaptives ungleiches Fehlerschutzschema für eine Videotelephonie, in welchem Eingangsrahmen sortiert und in Fehlerkorrekturklassen abgelegt werden, welchen jeweils ein Gewichtsfaktor zum Steuern der Kodierungsrate zugeordnet ist.

[0012] Die Veröffentlichung mit dem Titel "Still Image Transmission Using Unequal Error Protection Coding in Mobile Radio Channel" von N. Matoba et al., und veröffentlicht in Electronics und Communications in Japan, Teil 1, Bd. 79, Nr. 4, Seiten 75 bis 85 in 1996 offenbart ein ungleiches Fehlerschutzschema für JPEG-Daten inkludierend Parameter einer variablen Länge, in welchen Datenrahmen sortiert und

in Fehlerkorrekturklassen einer variablen Länge abgelegt sind, wobei die Anzahl von Datenblöcken jeder Fehlerkorrekturklasse mit einem Fehlerkorrekturcode einer hohen Redundanz übertragen wird.

Offenlegung der Erfindung

[0013] Die vorliegende Erfindung wurde angesichts des oben beschriebenen Standes der Technik erdacht, und hat als ihr Ziel, ein Fehlerschutzverfahren und eine Fehlerschutzvorrichtung vorzusehen, die es möglich machen, eine Vielfalt von Fehlerkorrekturalgorithmen einzusetzen, selbst wenn die Rahmen eine flexible Länge aufweisen.

[0014] Um das Ziel zu erreichen, inkludiert das Fehlerschutzverfahren in Bezug auf die vorliegende Erfindung einen Prozess zum Sortieren einer Vielzahl von Parametern, die einen Rahmen ausmachen, in eine Vielzahl von Klassen, einen Prozess zum Generieren von Rahmenbildungsdaten in Bezug auf die Bildung des Rahmens und Daten mit einem Fehlerschutzverfahren, das für jede Klasse bestimmt wird, um auf die Parameter angewendet zu werden, einen Prozess zum Implementieren des spezifizierten Fehlerschutzes für die Rahmenbildungsdaten, einen Prozess zum Implementieren des Fehlerschutzes, der für jede Klasse spezifiziert ist, gemäß den Rahmenbildungsdaten mit Bezug auf die Daten, die in die Vielzahl von Klassen sortiert sind, und einen Prozess zum Übertragen der Rahmenbildungsdaten, die dem spezifizierten Fehlerschutz unterzogen wurden, und Parametern, die für jede Klasse fehlergeschützt wurden.

[0015] Da die Parameter in Klassen unterteilt sind, und da die Vorrichtungen getrennt angeordnet sind, sodass es eine Vorrichtung zum Generieren von Rahmenbildungsdaten gibt, die den Inhalt des Fehlerschutzes anzeigen, der für jede Klasse durchzuführen ist, und eine Vorrichtung zum Durchführen eines Schutzes entsprechend jeder Klasse gemäß den Rahmenbildungsdaten ist es in Übereinstimmung mit der Erfindung möglich, auf Änderungen in dem Inhalt des Fehlerschutzes, der auf jede Klasse anzuwenden ist, abzustimmen.

[0016] Da die Rahmenbildungsdaten übertragen werden, die anzeigen, welcher Typ eines Fehlerschutzes für jede Klasse geeignet ist, wird des weiteren der Inhalt des Fehlerschutzes, der auf jede Klasse anzuwenden ist, aus diesen Rahmenbildungsdaten durch die Vorrichtung der empfangenden Seite entnommen, was es möglich macht, angemessen zu reagieren. Deshalb gibt es keine Notwendigkeit, den Inhalt des Fehlerschutzes festzusetzen, und es wird möglich, den Fehlerschutzinhalt zwischen der Vorrichtung der Übertragungsseite und der Vorrichtung der empfangenen Seite umzuschalten, wie es die Zweckmäßigkeit diktieren kann. Da die Rahmenbildungsdaten Daten enthalten, die die Bildung von Rahmen betreffen, ist es außerdem möglich, die Verarbeitung von Rahmen variabler Länge zu implementieren, die einem Fehlerschutz durch die Vorrichtung der empfangenden Seite unterzogen wurden.

Kurze Beschreibung der Zeichnungen

[0017] In den Zeichnungen zeigen:

[0018] Fig. 1 ein Blockdiagramm, das eine Konfiguration einer Fehlerschutzvorrichtung zeigt, die sich auf eine erste Ausführungsform eines Fehlerschutzsystems in Bezug auf die vorliegende Erfindung bezieht;

[0019] Fig. 2A–Fig. 2C Zeitdiagramme, die den Betrieb der Ausführungsform zeigen;

[0020] Fig. 3 ein Blockdiagramm, das eine Konfiguration einer Empfangsseitenvorrichtung in der Ausführungsform zeigt;

[0021] Fig. 4 ein Blockdiagramm, das eine Konfiguration einer Verschachtelungsvorrichtung zeigt, die einer Synthetisierungsvorrichtung der Fehlerschutzvorrichtung bereitgestellt ist;

[0022] Fig. 5 ein Blockdiagramm, das eine Beispielkonfiguration des Schreibadressenzufuhrteils der Verschachtelungsvorrichtung zeigt;

[0023] Fig. 6 ein Blockdiagramm, das eine Beispielkonfiguration des Leseadressenzufuhrteils der Verschachtelungsvorrichtung zeigt;

[0024] Fig. 7 ein Diagramm, das die Sequenz eines Schreibens der Bits, die einen Rahmen bilden, zu der Zeit eines Verschachtelns in die verschiedenen Speicherbereiche des Betriebsspeichers und die Sequenz eines Lesens der Bits, die einen Rahmen bilden, zu der Zeit eines Entschachtelns aus den verschiedenen Speicherbereichen eines Betriebsspeichers in der Ausführungsform zeigt;

[0025] Fig. 8 ein Diagramm, das die Sequenz eines Lesens der Bits, die einen Rahmen bilden, zu der Zeit eines Verschachtelns von den verschiedenen Speicherbereichen des Betriebsspeichers und die Sequenz eines Schreibens der Bits, die einen Rahmen bilden, zu der Zeit eines Entschachtelns in die verschiedenen Speicherbereiche eines Betriebsspeichers in der Ausführungsform zeigt;

[0026] Fig. 9A und Fig. 9B Zeitdiagramme, die den Betrieb der Verschachtelungsvorrichtung der Ausführungsform zeigen;

[0027] Fig. 10 ein Blockdiagramm, das eine Konfiguration der Entschachtelungsvorrichtung, die der

Empfangsseitenvorrichtung bereitgestellt ist, in der Ausführungsform zeigt;

[0028] Fig. 11 ein Blockdiagramm, das eine Konfiguration der Fehlerschutzvorrichtung in Bezug auf eine zweite Ausführungsform eines Fehlerschutzsystems in Bezug auf die vorliegende Erfindung betrifft, zeigt;

[0029] Fig. 12 ein Blockdiagramm, das eine Konfiguration der Fehlerschutzvorrichtung in Bezug auf die dritte Ausführungsform der vorliegenden Erfindung, die den besten Modus zum Ausführen der Erfindung darstellt; und

[0030] Fig. 13 ein Blockdiagramm, das eine Konfiguration einer Schutzvorrichtung nach dem Stand der Technik zeigt.

Bester Modus zum Ausführen der Erfindung

[0031] Das Folgende ist eine Beschreibung von Ausführungsformen der vorliegenden Erfindung, die unter Bezugnahme auf die Zeichnungen dargestellt sind.

A. Erste Ausführungsform

(1) Fehlerschutzvorrichtung

[0032] Fig. 1 ist ein Blockdiagramm, das eine Konfiguration einer Fehlerschutzvorrichtung zeigt, die die erste Ausführungsform der vorliegenden Erfindung betrifft.

[0033] Diese Fehlerschutzvorrichtung ist innerhalb bereitgestellt oder stromabwärts mit einem Kodierer verbunden, der eine Kompressionskodierung von Sprachsignalen durchführt.

[0034] Probensequenzen von Sprachsignalen, die zu übertragen sind, können zu jedweder Zeit von diesem Kodierer erzeugt werden. Zusätzlich wird, wie in den Fig. 2A und Fig. 2B gezeigt, eine Kompressionskodierung von Probensequenzen von Sprachsignalen, die zu den verschiedenen Rahmenzyklen gehören, mit einem fixierten Rahmenzyklus als eine Verarbeitungseinheit durchgeführt, wobei eine Vielzahl von Typen von Parametern erzeugt wird. Hier variiert das Datenvolumen der gebildeten Parametergruppen für jeden Rahmenzyklus.

[0035] Bezüglich der verschiedenen Parametergruppen führt die Fehlerschutzvorrichtung einen Fehlerschutz durch, der für jeden Parameter geeignet ist, und führt ihre Ausgabe durch.

[0036] Wie in Fig. 1 gezeigt, besitzt die Fehlerschutzvorrichtung in Bezug auf die vorliegende Ausführungsform einen Rahmenbildungs-Datenberechnungsteil **101**, einen Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102**, einen Klassendaten-Fehlerkorrektur-Verarbeitungsteil **103** und einen Synthetisierer **104**.

[0037] Der Rahmenbildungs-Datenberechnungsteil **101** dient primär dazu, die nächste Verarbeitung durchzuführen.

a. Klassenteilung von Parametergruppen

[0038] Die Parameter, die von dem Kodierer erzeugt werden, inkludieren manche mit einer hohen Fehlerempfindlichkeit und manche mit einer niedrigen Fehlerempfindlichkeit. Hier ist es, da Parameter mit einer hohen Fehlerempfindlichkeit nie ein niedriges akzeptables Restfehlerverhältnis aufweisen, empfehlenswert, das Restfehlerverhältnis auf der Empfangsseite durch Implementieren einer Kodierung mittels eines Fehlerkorrekturcodes mit einem hohen Kodierungsverhältnis zu reduzieren. Andererseits sollte bezüglich der Parameter mit einer niedrigen Fehlerempfindlichkeit kein Bedarf nach Fehlerkorrekturkodierung bei einem derartig hohen Kodierungsverhältnis vorhanden sein. Überdies werden in Abhängigkeit von dem Parameter manche Fälle vorhanden sein, wo es ausreichend ist, einfach einen Fehlererfassungscode hinzuzufügen, anstelle eine derartige Fehlerkorrekturkodierung durchzuführen. Außerdem müssen manche dieser Parameter verschachtelt werden und manche nicht. Somit sind unterschiedliche Fehlerschutzverfahren auf unterschiedliche Fälle anzuwenden, in Abhängigkeit von dem Typ eines Parameters.

[0039] Dementsprechend werden in der vorliegenden Ausführungsform Parameter, die von dem Kodierer erzeugt werden, in Klassen gemäß dem Fehlerschutzverfahren, das auf jeden Fall anzuwenden ist, sortiert. Zusätzlich führt in der Fehlerschutzvorrichtung, die in Fig. 1 gezeigt ist, der Rahmenbildungs-Datenberechnungsteil **101** das Sortieren in Klassen durch.

[0040] In dem in Fig. 1 gezeigten Beispiel sind n Klassen eingerichtet, von einer Klasse 1 bis zu einer Klasse n. Der Typ eines Fehlerschutzes, der auf die Parameter angewandt wird, die zu diesen Klassen gehören, wird zuvor in der Sendeseitenvorrichtung bestimmt. Überdies wird der Typ von Parametern, die zu den Klassen gehören, auch zuvor bestimmt. Überdies ist es auch möglich, wie gewünscht den Inhalt des Fehlerschutzes, der auf die verschiedenen Klassen mittels der Sendeseitenvorrichtung angewendet wird, zu variieren. Der Rahmenbildungs-Datenberechnungsteil **101** beurteilt die Klasse, die zu jedem Parameter gehört, der durch den Kodierer gebildet wird, gemäß dem Typ jedes Parameters, indem er Entscheidungskriterien in der Sendeseitenvorrichtung folgt. Es sei darauf hingewiesen, dass die Para-

meter, die an den verschiedenen Klassen durch ein Sortieren angebracht worden sind, aus Bequemlichkeitsgründen nachstehend als die Klassendaten jeder Klasse bezeichnet werden sollten.

b. Erzeugung von Rahmenbildungsdaten

[0041] Der Rahmenbildungs-Datenberechnungsteil **101** bestimmt die gesamte Anzahl von Bits der Parameter, die von dem Kodierer innerhalb eines Rahmenzyklus erzeugt werden, und gibt die Daten Cb_1–Cb_n, die die Anzahl von Bits von Klassendaten jeder Klasse 1–n anzeigen, aus.

[0042] Überdies gibt, zusätzlich zu einem Ausgeben der Daten Cb_1–Cb_n, die die Anzahl von Bits von Klassendaten jeder Klasse anzeigen, der Rahmenbildungs-Datenberechnungsteil **101** auch die Daten CC_1–CC_n aus, die den Fehlerschutzinhalt anzeigen, der auf die Klassendaten jeder Klasse angewendet wird.

[0043] Sämtliche der Daten CC_1–CC_n inkludieren beispielsweise einen Kodierungsalgorithmus zur Fehlerkodierung und Daten, ob ein Verschachteln durchgeführt wurde oder nicht und betreffend den Typ des Verschachtelns, falls durchgeführt.

[0044] Außerdem gibt der Rahmenbildungs-Datenberechnungsteil **101** die Daten ECR_1–ECR_n aus, die eine Kodierungsrate in Fällen anzeigen, wo eine Fehlerkorrektur für Klassendaten jeder Klasse 1–n implementiert ist. Die Daten ECR_1–ECR_n werden auf der Grundlage eines Restfehlerverhältnisses für Klassendaten jeder Klasse bestimmt.

[0045] Außerdem gibt der Rahmenbildungs-Datenberechnungsteil **101** die Daten EDB_1–EDB_n, die die Anzahl von Bits eines Fehlererfassungscodes anzeigen, in Fällen aus, wo ein Fehlererfassungscode (z. B. Bits eines zyklischen Redundanzcodes) Klassendaten für jede Klasse 1–n hinzugefügt sind. Die Anzahl dieser Bits wird auf der Grundlage der Fehlerempfindlichkeit der Klassendaten jeder Klasse bestimmt.

[0046] Beispielsweise wird, in Fällen, wo die Fehlerempfindlichkeit der Klassendaten einer Klasse niedrig ist, und wo es absehbar ist, dass auch dann, wenn ein Bitfehler auftritt, die Wirkung auf der Empfangsseite gering sein würde, somit die Anzahl von Bits des Fehlererfassungscodes für diese Klasse auf Null gesetzt. Andererseits wird in Fällen, wo ein Klassendaten-Bitfehler auftritt, wenn absehbar ist, dass die Wirkung auf der Empfangsseite groß sein würde, dann die Anzahl von Bits des Fehlererfassungscodes für diese Klasse in Bezug auf die Anzahl von zu schützenden Bits (d. h. die Anzahl von Bits dieser Klasse) bestimmt.

[0047] Die Daten Cb_1–Cb_n, CC_1–CC_n, ECR_1–ECR_n und EDB_1–EDB_n, die entsprechend durch den Rahmenbildungs-Datenberechnungsteil **101** ausgegeben werden, sind sämtlich Daten, die die Klasse betreffen, die den Rahmen bildet, und werden somit generisch als Rahmenbildungsdaten bezeichnet.

[0048] Als nächstes führt der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102** eine spezifizierte Fehlerkorrektur bezüglich der Rahmenbildungsdaten aus, die von dem Rahmenbildungs-Datenberechnungsteil **101** ausgegeben werden, das heißt, er führt eine Fehlerkorrektur eines Typs durch, der mit dem Empfänger vorbestimmt ist. In diesem Fall kann der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102** eine Fehlerkorrekturkodierung von Rahmenbildungsdaten durchführen, oder er kann einfach einen Fehlererfassungscode hinzufügen.

[0049] Die Klassendaten der Klassen, die in Klassen 1–n von dem Rahmenbildungs-Datenberechnungsteil **101** sortiert sind, werden dem Klassendaten-Fehlerschutz-Verarbeitungsteil **103** zugeführt. Fig. 2c zeigt Beispiele derartiger Klassendaten. Der Klassendaten-Fehlerschutz-Verarbeitungsteil **103** führt in den empfangenen Klassendaten jeder Klasse eine Fehlerkorrekturkodierung des Typs durch, der in den Rahmenbildungsdaten bezeichnet ist, oder er fügt einen Fehlererfassungscode zu Klassendaten jeder Klasse 1–n hinzu.

[0050] Das heißt, in Fällen, wo eine Fehlerkorrekturkodierung bezüglich der Klassendaten einer bestimmten Klasse k über die Verwendung eines Kodierungsalgorithmus, der durch die Daten CC_k, gekennzeichnet ist, durchzuführen ist, führt der Klassendaten-Fehlerschutz-Verarbeitungsteil **103** eine Fehlerkorrekturkodierung der Klassendaten durch den Kodierungsalgorithmus bei einem Kodierungsverhältnis aus, das durch die Daten ECR_k gekennzeichnet ist, die der Klasse k entsprechen. Überdies erzeugt in Fällen, wo die Daten EDB_k, die einer bestimmten Klasse k entsprechen, nicht 0 sind, der Klassendaten-Fehlerschutz-Verarbeitungsteil **103** einen Fehlererfassungscode mit der Anzahl von Bits, die durch die Daten EDB_k gekennzeichnet sind, aus den Klassendaten der Klasse k, und fügt ihn den Klassendaten hinzu.

[0051] Der Synthetisierer **104** sammelt die Rahmenbildungsdaten, die einen Fehlerschutz durch den Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102** durchlaufen haben, und die Klassendaten, die eine Fehlerkorrektur durch den Klassendaten-Fehlerschutz-Verarbeitungsteil **103** durchlaufen haben, synthetisiert sie und überträgt sie zu der Empfangsseitenvorrichtung über den Übertragungspfad.

[0052] Unter den Klassen 1 bis n sind Klassen vor-

handen, die für ein Verschachteln gekennzeichnet worden sind. Ein Verschachteln dieser Klassen wird durch diesen Synthetisierer **104** durchgeführt. Die folgenden beiden Typen von Verschachtelungsverfahren sind für Fälle wie diesen verfügbar.

a. Die Anordnung von Bits in jede Klasse, die für ein Verschachteln gekennzeichnet ist;

b. Die Bits, die zu der Klasse gehören, die für ein Verschachteln gekennzeichnet sind, werden gestreut und in der Bitsequenz einer anderen Klasse angeordnet.

[0053] Die Daten CC_k, die einer Klasse k entsprechen, kennzeichnen, welcher Typ eines Verschachtelns durchgeführt wird. Überdies ist hinsichtlich der Klasse k bei einem Durchführen eines Verschachtelns gemäß dem Verfahren b die andere Klasse (z. B. Klasse j), mit welcher die Bits, die die Klasse k bilden, gestreut und angeordnet werden, durch die Daten CC_k gekennzeichnet.

[0054] Es sei darauf hingewiesen, dass eine detaillierte Beschreibung betreffend die Vorrichtung zum Verschachteln gemäß dem Verfahren b folgen wird.

[0055] Die Rahmenbildungsdaten, die den Fehlerschutz durchlaufen haben, werden den Klassen als Header hinzugefügt, und die Rahmen variabler Länge, die von den Headern und den Parametern gebildet werden, werden zu der Empfangsseitenvorrichtung über den Übertragungspfad übertragen.

[0056] Es sei auch darauf hingewiesen, dass die Rahmenbildungsdaten, anstatt dass sie als Header übertragen werden, auch zu einer Empfangsseitenvorrichtung über Kommunikationskanäle übertragen werden, die sich von jenen unterscheiden, über welche die Rahmen, die aus den Klassendaten der verschiedenen Klassen gebildet sind, übertragen werden.

(2) Die Empfangsseitenvorrichtung

[0057] Wie in Fig. 3 gezeigt, ist die Empfangsseitenvorrichtung eine Vorrichtung, die aus einem Separator **105**, einem Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106**, einer Entschachtelungsvorrichtung **107** und einem Klassendaten-Fehlerschutz-Verarbeitungsteil **108** gebildet sind, und stromaufwärts von dem Decoder befindlich.

[0058] Der Separator **105** trennt die Rahmen, die durch den Synthetisierer **104** der Senderseite ausgegeben werden, in Rahmenbildungsdaten und Klassendaten.

[0059] Der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106** führt Fehlerkorrekturdekodierung oder Fehlererfassung für die Rahmenbildungsdaten aus, die aus dem Separator **105** ausgegeben werden. Eine Verarbeitung, die durch diesen Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106** durchgeführt wird, entspricht der Verarbeitung, die von dem Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102** auf der Sendeseite durchgeführt wird. Das heißt in Fällen, wo der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102** der Sendeseite das Codieren der Rahmenbildungsdaten mittels eines spezifizierten Fehlerkorrekturcodes durchführt, führt dieser Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106** in den Rahmenbildungsdaten eine Fehlerdekodierung durch, die dem Fehlerkorrekturcode entspricht. Überdies setzt in Fällen, wo der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102** der Sendeseite einen Fehlererfassungscode einer spezifizierten Anzahl von Bits zu den Rahmenbildungsdaten hinzufügt, dieser Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106** den Fehlererfassungscode ein, um Fehler in den Rahmenbildungsdaten zu erfassen.

[0060] Der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106** gibt Rahmenbildungsdaten aus, für welche Fehlerkorrekturdekodierung oder Fehlererfassung vollendet worden ist.

[0061] Die Entschachtelungsvorrichtung **107** führt Entschachtelung bezüglich Klassendaten durch, die verschachtelt worden sind, wobei Bezug genommen wird auf die Daten CC_1–CC_n unter diesen Rahmenbildungsdaten.

[0062] Das heißt in Fällen, wo aus den Daten CC_k, die der Klasse k entsprechen, bestimmt worden ist, dass ein Verschachteln des Typs vorgenommen wurde, das Bits in der Klasse k anordnet, führt die Entschachtelungsvorrichtung **107** den umgekehrten Betriebsschritt eines Anordnens dieser Bits durch und stellt die Klassendaten in ihren ursprünglichen Zustand vor der Bitanordnung wieder her.

[0063] Überdies entfernt in Fällen, wo aus den Daten CC_k, die der Klasse k entsprechen, bestimmt worden ist, dass ein Verschachteln des Typs vorgenommen worden ist, der die Bits, die die Klasse k bilden, streut und sie innerhalb einer Bitsequenz einer anderen Klasse j anordnet, die Entschachtelungsvorrichtung **107** die Bits, die der Klasse k entsprechen, aus der Bitsequenz der Klasse j und stellt die Klassendaten der Klassen k und m in den Zustand vor dem Verschachteln wieder her.

[0064] Die Vorrichtung zum Ausführen des letzteren Typs eines Entschachtelns wird im Detail untenstehend beschrieben.

[0065] Die Entschachtelungsvorrichtung **107** führt der Klassendaten-Fehlerschutz-Verarbeitungsvorrichtung **108** die entschachtelten Klassendaten für jede Klasse zu.

[0066] Die Daten, die für eine Klassendaten-Fehlerkorrekturdekodierung oder Fehlererfassung benötigt werden, wie etwa die Anzahl von Bits jeder Klasse, das Kodierungsverhältnis jeder Klasse, die Anzahl von Bits jedes Klassenfehler-Erfassungscodes, der Kodierungsalgorithmus jeder Klasse und dergleichen, werden dieser Klassendaten-Fehlerschutz-Verarbeitungsvorrichtung **108** von dem Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **106** zugeführt.

[0067] Der Klassendaten-Fehlerschutz-Verarbeitungsteil **108** implementiert eine Fehlerkorrekturdekodierung oder eine Fehlererfassung für jeweilige Klassendaten in Übereinstimmung mit diesen Daten und liefert die Klassendaten, die verarbeitet worden sind, zu dem Decoder.

[0068] Wenn die Fehlerschutzvorrichtung in Bezug auf die vorliegende Ausführungsform eingesetzt wird, ist es möglich, mit einer Flexibilität auf Änderungen in dem Fehlerschutzinhalt, der auf jede Klasse anzuwenden ist, zu antworten, da die Konfiguration derart ist, dass sie in eine Vorrichtung, die Rahmenbildungsdaten erzeugt, die den Inhalt des an jede Klasse, die durch das Sortieren der Parameter erhalten wird, anzulegenden Fehlerschutzes anzeigen, und eine Vorrichtung, die einen Fehlerschutz entsprechend jeder Klasse in Übereinstimmung mit diesen Rahmenbildungsdaten durchführt, getrennt ist.

[0069] Überdies wird gemäß der vorliegenden Ausführungsform, da die Rahmenbildungsdaten, die Daten enthalten, die den Inhalt des Fehlerschutzes, der auf jede Klasse anzuwenden ist, anzeigen, von der Sendeseiten- zu der Empfangsseiten-Vorrichtung übertragen werden, der Inhalt des Fehlerschutzes, der an jede Klasse anzulegen ist, wie er durch die Rahmenbildungsdaten angezeigt wird, von der Empfangsseitenvorrichtung bestimmt. Es ist somit möglich, eine Fehlerkorrekturdekodierung, eine Fehlererfassung oder eine Entschachtelung der Klassendaten durchzuführen. Deswegen besteht kein Bedarf, den Inhalt des Fehlerschutzes, der auf jeden Parameter anzuwenden ist, der übertragen wird, zu fixieren, und es wird möglich, Rahmen zu übertragen, indem der Fehlerschutzinhalt zwischen der Wendeseitenvorrichtung und der Empfangsseitenvorrichtung umgeschaltet wird, wie es eine Zweckmäßigkeit vorgeben kann. Zusätzlich ist es, da die Empfangsseitenvorrichtung über die Konfiguration mittels der Rahmenbildungsdaten informiert wird, für die Empfangsseitenvorrichtung möglich, eine Fehlerkorrekturdekodierung, eine Fehlererfassung oder eine Entschachtelung von Klassendaten für jede Klasse geeignet auszuführen, auch wenn der Rahmen eine variable Rahmenlänge aufweist, wobei die Anzahl von Bits über der Zeit fluktuiert.

(3) Die Verschachtelungsvorrichtung und die Entschachtelungsvorrichtung

[0070] Fig. 4 ist ein Blockdiagramm, das eine Konfiguration einer Verschachtelungsvorrichtung, die der Synthetisierungsvorrichtung **104** der Sendeseite in der Ausführungsform bereitgestellt ist, zeigt.

[0071] Diese Verschachtelungsvorrichtung ist eine Vorrichtung, die ein Verschachteln durch ein Streuen der Bits, die die Klassendaten einer Klasse k bilden, durchführt, indem sie innerhalb der Bitsequenz der Klassendaten einer anderen Klasse j angeordnet werden.

[0072] Hier werden, in dem Fall, wo die Bitsequenz der Klassendaten der Klasse k D_1 und die Bitsequenz der Klassendaten der Klasse j D_0 ist, die Bits, die die Bitsequenzen D_1 und D_0 bilden, der Verschachtelungsvorrichtung synchron mit dem Bittakt CLK zugeführt.

[0073] Wie in Fig. 4 gezeigt, inkludiert diese Verschachtelungsvorrichtung als primäre Strukturelemente einen Schreibadressen-Zufuhrteil **510**, einen Betriebsspeicher **510** und einen Leseadressen-Zufuhrteil **530**.

[0074] Hier ist der Betriebsspeicher **520** eine Speichervorrichtung zum Speichern der Bitsequenzen D_1 und D_0, die zu verschachteln sind. In der vorliegenden Ausführungsform wird eine Vielzahl von Speicherberechen mit kontinuierlichen Adressen zur Verwendung aus sämtlichen der Speicherbereiche des Betriebsspeichers **520** gewählt, und die Bits werden in ihren jeweiligen Speicherbereichen gespeichert.

[0075] Es sei darauf hingewiesen, dass in der folgenden Diskussion die Bitsequenz D_1 einer Klasse k aus p Bits gebildet ist, und die Bitsequenz D_0 aus m × p Bits gebildet ist.

[0076] Ein Verschachteln besteht aus dem Prozess eines sequentiellen Schreibens der Bits der Bitsequenzen D_1 und D_0 in den Betriebsspeicher **520**, und der Prozess eines Lesens der Bits aus diesem Betriebsspeicher **520** in einer Sequenz, die sich von davon unterscheidet, wenn sie geschrieben werden.

[0077] Der Schreibadress-Zufuhrteil **510** ist eine Vorrichtung, die ein spezifiziertes Adresserzeugungsprogramm durchführt, um synchron zu dem Zeittakt CLK die Schreibadresse WADh, die die Schreibbestimmung der Bits, die die Bitsequenz D1 bilden, spezifiziert, und die Schreibadresse WADd zu erzeugen, die die Schreibbestimmung der Bits, die die Bitsequenz D_0 bilden, spezifiziert, und führt sie dem Betriebsspeicher **520** zu, wenn die Bits in den Betriebsspeicher **520** geschrieben werden.

[0078] Überdies wird, wenn sämtliche der Schreibadressen, die zum Schreiben der Bits in den Betriebsspeicher **520** benötigt werden, nicht länger erzeugt werden, dem Leseadressenteil **530** das END-Symbol, das dies anzeigt, zugeführt.

[0079] Der Leseadressen-Zufuhrteil **530** ist eine Vorrichtung, die die Leseadresse RAD zum Lesen der Bits, die in dem Betriebsspeicher **520** gespeichert sind, erzeugt, indem sie synchron zu dem Bittakt CLK durch ein Ausführen eines spezifizierten Adressenerzeugungsprogramms erzeugt wird, wenn das END-Symbol empfangen wird, und führt die Leseadresse RAD dem Betriebsspeicher **520** zu.

[0080] Fig. 5 ist ein Blockdiagramm, das die Adressenerzeugungsalgorithmen, die von dem Schreibadressen-Zufuhrteil **510** ausgeführt wird, von dem Standpunkt einer Hardware zeigt.

[0081] Der Adressenerzeugungsalgorithmus gemäß diesem Schreibadressen-Zufuhrteil **510** ist aus einem p-skaligen Zähler **510A**, einem p-skaligen Zähler **510b**, einem m-skaligen Zähler **510C**, einem UND-Gatter **510D** und einem Addierer **510E** gebildet. Hier zählt der p-skalige Zähler **510A** den Bittakt CLK. Die Schreibadresse WADh für die Bits, die die Bitsequenz D_1 bilden, weist eine untere Adresse mit einem Zählwert des p-skaligen Zählers **510A** und eine obere Adresse, die als "0" gekennzeichnet ist, auf.

[0082] Der p-skalige Zähler **510A** stoppt das Zählen des Bittakts CLK, wenn sich der Zählwert eine Anzahl p von Malen ändert, was eine Überzählung ergibt.

[0083] Das UND-Gatter **510D** führt dem m-skaligen Zähler **510C** den Bittakt CLK zu, nachdem der p-skalige Zähler **510A** eine Überzählung ergibt.

[0084] Der m-skalige Zähler **510c** führt ein Zählen des Bittakts CLK, der über dieses UND-Gatter **510D** zugeführt wird, aus. Der Zählwert des m-skaligen Zählers **510C** ergibt eine Überzählung, wenn er sich m-mal ändert, aber danach, solange der Bittakt CLK zugeführt wird, wird die Zählung wieder von dem Anfangswert "0" wiederholt. Der p-skalige Zähler **510B** fügt Zählwerte "1" jedes Mal dann hinzu, wenn der m-skalige Zähler **510C** eine Überzählung ergibt. Der Addierer **510E** addiert "1" zu dem Zählwert des m-skaligen Zählers **510C** und überträgt das Ergebnis.

[0085] Die Schreibadresse WADd, die die Schreibbestimmung der Bits, die die Bitsequenz D_0 bilden, spezifiziert, ist derart, dass der Zählwert des p-Zählers **510B** eine untere Adresse ist, und die Ausgangsdaten des Addierers **510E** eine obere Adresse sind.

[0086] Fig. 6 ist ein Blockdiagramm, das eine Beispielkonfiguration des Leseadressen-Zufuhrteils **530** zeigt. Dieser Leseadressen-Zufuhrteil **530** ist aus einem p-skaligen Zähler **530A** und einem (m + 1)-skaligen Zähler **530B** gebildet. Der (m + 1)-skalige Zähler **530B** führt ein Zählen für den Bittakt CLK aus. Der p-skalige Zähler **530A** inkrementiert seinen Zählwert um "1" jedes Mal dann, wenn sich die Zählung des (m + 1)-skaligen Zählers **530B** m + 1 mal ändert und kehrt auf den Anfangswert "0" zurück.

[0087] Die Leseadresse RAD ist derart, dass der Zählwert dieses p-skaligen Zählers **530A** eine untere Adresse ist, und der Zählwert des (m + 1)-skaligen Zählers **530B** eine obere Adresse ist.

[0088] Das obige ist eine detaillierte Beschreibung der Konfiguration der Verschachtelungsvorrichtung.

[0089] Es folgt eine Beschreibung einer Verschachtelung, die durch diese Verschachtelungsvorrichtung implementiert ist, unter Bezugnahme auf Fig. 7 und Fig. 8.

[0090] In dieser Verschachtelungsvorrichtung wird ein Verschachteln unter Verwendung von (m + 1) × p kontinuierlichen Speicherbereichen in dem Betriebsspeicher **520** implementiert.

[0091] Fig. 7 und Fig. 8 zeigen den Speicherbereich zum Verschachteln, der als ein zweidimensionaler Speicherraum ausgedrückt ist. Die verschiedenen individuellen Adressen entsprechen den Speicherbereichen in diesem Speicherraum. Eine Adresse wird aus einer unteren Adresse, die p-Kombinationen von Werten annehmen kann, und einer oberen Adresse, die (m + 1)-Kombinationen von Werten annehmen kann, gebildet.

[0092] In Fig. 7 und Fig. 8 sind die Speicherbereiche, die identische obere Adressen aufweisen müssen, von links nach rechts in einer Reihenfolge der unteren Adresse angeordnet, und die Speicherbereiche, die identische untere Adressen aufweisen müssen, sind von oben nach unten in einer oberen Adressreihenfolge angeordnet. Wie oben bemerkt, wird eine Reihe von Speicherbereichen mit identischen oberen Adressen als Zeilen bezeichnet werden, und eine Reihe von Speicherbereichen mit identischen unteren Adressen wird als Spalten bezeichnet werden.

[0093] Bei einem Implementieren des Verschachtelns werden zuallererst die Bitsequenzen, die zu verschachteln sind, sequentiell, jede in ihren jeweiligen m × n-Speicherbereichen bitweise geschrieben. Wenn dieses Schreiben ausgeführt ist, werden die Schreibadressen von dem Schreibadressen-Zufuhrteil **510** erzeugt, der obenstehend unter Bezugnahme auf Fig. 5 beschrieben ist.

[0094] Während die p Bits, die die Bitsequenz D_1 bil-

den, dem Betriebsspeicher **520** synchron zu dem Bittakt CLK zugeführt werden, wird das Bittakt CLK-Zählen gemäß dem p-skaligen Zähler **510A** in dem Schreibadressen-Zufuhrteil **510** durchgeführt. Auch wird die Schreibadresse WADh erzeugt, die den Zählwert des p-skaligen Zählers **510A** als eine untere Adresse und "0" als eine obere Adresse enthält, und wird dem Betriebsspeicher **520** synchron zu dem Bittakt CLK zugeführt.

[0095] Folglich werden, wie in Fig. 7 gezeigt, die p-Bits, die die Bitsequenz D_1 bilden, in die Anfangszeile innerhalb des Speicherbereichs des Betriebsspeichers **520** zum Verschachteln geschrieben.

[0096] Als nächstes werden, folgend auf die Bitsequenz D_1, die m × p Bits, die Bitsequenz D_0 bilden, dem Betriebsspeicher **520** synchron zu dem Bittakt CLK zugeführt.

[0097] Unterdessen wird das Zählen des Bittakts CLK durch den m-skaligen Zähler **510C** in dem Schreibadressen-Zufuhrteil **510** ausgeführt, und zusätzlich wird eine Zählwerthochsetzung des p-skaligen Zählers **510B** jedes Mal dann ausgeführt, wenn der m-skalige Zähler **510C** eine Überzählung ergibt. Außerdem wird eine Schreibadresse WADd erzeugt, die den Zählwert des p-skaligen Zählers **510B** als eine untere Adresse enthält, und eine "1" wird dem Zählwert des m-skaligen Zählers **510C** als eine obere Adresse hinzugefügt, und diese Schreibadresse wird dem Betriebsspeicher **520** synchron zu dem Bittakt CLK zugeführt.

[0098] Folglich werden, wie in Fig. 7 gezeigt, die m Anfangsbits von den m × p Bits, die die Bitsequenz D_0 bilden, in jeden Speicherbereich entsprechend der (m + 1)-ten Zeile von der zweiten Zeile der ersten Spalte, und in den Betriebsspeicher **520** geschrieben, und dann werden die m Bits in jeden Speicherbereich, der der (m + 1)-ten Zeile entspricht, von der zweiten Zeile der zweiten Spalte und so weiter geschrieben, bis die letzten m Bits (d. h. das p-te Bit) in jeden Speicherbereich, der der (m + 1)-ten Zeile entspricht, von der zweiten Spalte der p-ten Spalte geschrieben.

[0099] Als nächstes werden die Bits, die somit in den Betriebsspeicher **520** geschrieben sind, in einer Sequenz gelesen, die sich von derjenigen unterscheidet, wenn sie geschrieben werden.

[0100] In diesem Lesebetrieb werden Leseadressen durch den Leseadressen-Zufuhrteil **530** erzeugt, der unter Bezugnahme auf Fig. 6 beschrieben ist.

[0101] Das heißt, das Zählen des Bittakts CLK wird durch den (m + 1)-skaligen Zähler **530B** ausgeführt, und zusätzlich wird ein Zählwertinkrement mit einem Zählwert von "1" nur des p-skaligen Zählers **530A** jedes Mal ausgeführt, wenn der (m + 1)-skalige Zähler **530B** eine Überzählung ergibt. Außerdem wird dem Betriebsspeicher **520** synchron zu dem Bittakt CLK eine Leseadresse, die den Zählwert des p-skaligen Zählers **530A** als eine untere Adresse und den Zählwert des (m + 1)-skaligen Zählers **530B** als eine obere Adresse enthält, zugeführt.

[0102] Folglich werden, wie in Fig. 8 gezeigt, zuallererst die (m + 1) Bits, die in jedem Speicherbereich der ersten Spalte in dem Betriebsspeicher **520** aufgezeichnet sind, gelesen, und dann werden die (m + 1) Bits, die in jedem Speicherbereich der zweiten Spalte aufgezeichnet sind, gelesen, und so weiter, bis schließlich die (m + 1) Bits, die in jedem Speicherbereich der p-ten Spalte aufgezeichnet sind, gelesen werden.

[0103] Aufgrund des oben beschriebenen Verschachtelns werden die p-Bits, die die Bitsequenz D_1 bilden, gestreut und in gleichen Intervallen innerhalb der Bitsequenz D_0 angeordnet, wie in den Fig. 9A und Fig. 9B gezeigt.

[0104] Das obige war eine detaillierte Beschreibung der Verschachtelungsvorrichtung.

[0105] Es folgt eine Beschreibung der Entschachtelungsvorrichtung, die der Empfangsseitenvorrichtung bereitgestellt ist.

[0106] Fig. 10 ist ein Blockdiagramm, das eine Konfiguration dieser Entschachtelungsvorrichtung zeigt.

[0107] Wenn ein Verschachteln auf der Sendeseitenvorrichtung durch ein Streuen der Bits, die die Klassendaten der Klasse k bilden, implementiert worden ist, indem sie innerhalb der Bitsequenz der Klassendaten einer anderen Klasse j angeordnet werden, stellt diese Entschachtelungsvorrichtung die Klassendaten der nicht-verschachtelten Klasse durch ein Entfernen der Bits der Klassendaten der Klasse k aus der Bitsequenz der Klassendaten der Klasse j wieder her.

[0108] Dieser Entschachtelungsvorrichtung werden synchron zu dem Bittakt CLK die (m + 1) × p Bits, die die Bitsequenz D_1 der Klassendaten der Klasse K bilden, und die Bitsequenz D_0 Klassendaten der Klasse j zugeführt.

[0109] Diese Entschachtelungsvorrichtung umfasst einen Schreibadressen-Zufuhrteil **610**, einen Betriebsspeicher **620** und eine Leseadressen-Zufuhrteil **630**.

[0110] Hier ist der Betriebsspeicher **620** eine Speichervorrichtung ähnlich zu dem Betriebsspeicher **520** in der Verschachtelungsvorrichtung **1**.

[0111] Entschachteln besteht aus dem Prozess eines Schreibens dieser zugeführten Bits in den Betriebsspeicher **620** und dem Prozess eines Lesens dieser Bits aus dem Betriebsspeicher **620** in einer Sequenz, die sich von derjenigen unterscheidet, wenn sie geschrieben werden.

[0112] Während diese Bits, die den empfangenen Rahmen bilden, dem Betriebsspeicher **620** synchron zu dem Bittakt CLK zugeführt werden, führt der Schreibadressen-Zufuhrteil **610** ein spezifiziertes Adressenerzeugungsprogramm aus, um die Schreibadresse WAD zu erzeugen, indem sie den Betriebsspeicher **620** synchron zu dem Bittakt CLK zugeführt wird.

[0113] Dieser Schreibadressen-Zufuhrteil **610** besitzt eine Konfiguration, die identisch zu jener des Leseadressen-Zufuhrteils **520** in der Verschachtelungsvorrichtung (siehe Fig. 6) ist.

[0114] Außerdem wird in dem Fall dieses Schreibadressen-Zufuhrteils **610** das Zählen des Bittakts durch den (m + 1)-skaligen Zähler ausgeführt, und zusätzlich inkrementiert der p-skalige Zähler seinen Zählwert um "1" jedes Mal, wenn der (m + 1)-skalige Zähler eine Überzählung ergibt. Außerdem wird dem Betriebsspeicher **620** synchron zu dem Bittakt eine Schreibadresse WAD, die den Zählwert des p-skaligen Zählers als eine untere Adresse und den Zählwert des (m + 1)-skaligen Zählers als eine obere Adresse enthält, zugeführt.

[0115] Folglich werden, wie in Fig. 8 gezeigt, die (m + 1)-Anfangsbits in jeden Speicherbereich der ersten Spalte in dem Betriebsspeicher **620** geschrieben, und dann werden die (m + 1)-Bits in jeden Speicherbereich der zweite Spalte geschrieben, und so weiter, bis die letzten (m + 1)-Bits (d. h. das p-te Bit) in jeden Speicherbereich der p-ten Spalte geschrieben werden.

[0116] Die Platzierung jedes Bits in dem Betriebsspeicher **620** zu dieser Zeit ist in Übereinstimmung mit der Position dieser Bits, wenn sie in den Betriebsspeicher **620** der Verschachtelungsvorrichtung geschrieben werden, bevor sie von der Sendeseitenvorrichtung gesendet werden.

[0117] Wenn der Prozess eines Schreibens sämtlicher der Bits, die den empfangenen Rahmen bilden, in den Betriebsspeicher **620** vollendet ist, wird dem Leseadressen-Zufuhrteil **630** von dem Schreibadressen-Zufuhrteil **610** das END-Signal, das dieses anzeigt, zugeführt.

[0118] Dieser Leseadressen-Zufuhrteil **620** besitzt eine Konfiguration, die identisch zu jener des Schreibadressen-Zufuhrteils **510** in der Verschachtelungsvorrichtung (siehe Fig. 5) ist.

[0119] Außerdem wird in dem Fall dieses Leseadressen-Zufuhrteils **630** das Zählen des Bittakts CLK von dem p-skaligen Zähler ausgeführt. Außerdem wird eine Leseadresse RADh, die den Zählwert des p-skaligen Zählers als eine untere Adresse und "0" als eine obere Adresse enthält, erzeugt und dem Betriebsspeicher **620** synchron zu dem Bittakt zugeführt.

[0120] Folglich werden, wie in Fig. 7 gezeigt, die p-Bits, die die Bitsequenz D_1 bilden, von jedem Speichereich gelesen, der der Anfangszeile in dem Betriebsspeicher **620** entspricht.

[0121] Als nächstes wird in dem Fall des Leseadressen-Zufuhrteils **630** das Zählen des Bittakts durch den m-skaligen Zähler ausgeführt, und zusätzlich wird ein Zählwertinkrement mit einem Zählwert des p-skaligen Zählers jedes Mal ausgeführt, wenn der m-skalige Zähler eine Überzählung ergibt. Außerdem wird eine Leseadresse RADd, die den Zählwert des p-skaligen Zählers als eine untere Adresse und den Zählwert des m-skaligen Zählers plus "1" als eine obere Adresse enthält, erzeugt und dem Betriebsspeicher **620** synchron zu dem Bittakt CLK zugeführt.

[0122] Folglich werden, wie in Fig. 7 gezeigt, die m Anfangsbits unter den m × p Bits, die die Bitsequenz D_0 bilden, von jedem Speicherbereich entsprechend der (m + 1)-ten Zeile von der zweiten Zeile der ersten Spalte in dem Betriebsspeicher **620** gelesen. Dann werden die folgenden m-Bits von jedem Speichereich entsprechend der (m + 1)-ten Zeile von der zweiten Zeile der zweiten Spalte in dem Betriebsspeicher **620** gelesen, und die folgenden m-Bits werden von jedem Speichereich entsprechend der der (m + 1)-ten Zeile von der zweiten Zeile der dritten Spalte gelesen, und so weiter, bis die letzten m-Bits (d. h. das p-te Bit) von jedem Speicherbereich entsprechend der (m + 1)-ten Zeile von der zweiten Zeile der p-ten Spalte gelesen sind.

[0123] Entsprechend führt die Entschachtelungsvorrichtung einen Betrieb durch, der vollständig der umgekehrte der Verschachtelung ist, der in dem Fall der Verschachtelungsvorrichtung durchgeführt wird, und gewinnt den ursprünglichen, nicht-verschachtelten Rahmen wieder.

[0124] Es sei darauf hingewiesen, dass in dem Beispiel oben, obwohl der Fall, wo die Bitsequenz D_1 der Klasse k aus p-Bits gebildet ist, und die Bitsequenz D_0 der Klasse j aus m × p Bits erläutert ist, der Fall, wo die Anzahl von Bits der Bitsequenz D_0 nicht ein geradzahliges Vielfaches von p ist, auch möglich ist. In diesem Fall ist es beispielsweise möglich, ein Verfahren einzusetzen, wie etwa, dass Blindbits der Bitsequenz D_0 hinzugefügt werden, um eine Bitlänge zu bilden, die ein geradzahliges Vielfaches der Bitlänge ist, und dann die Blindbits zu entfernen, nachdem

ein Verschachteln beendet ist.

[0125] Außerdem sind die Konfigurationen der Vorrichtungen zum Verschachteln und Entschachteln, die obenstehend beschrieben sind, in jedem Fall Beispiele, und Vorrichtungen, die in der Lage sind, das Verschachteln und Entschachteln durchzuführen, das in der Ausführungsform benötigt wird, sind nicht darauf beschränkt.

[0126] Zusätzlich zu der oben beschriebenen grundlegenden Ausführungsform kann ein Verschachteln in einer Vielfalt von Ausführungsformen implementiert werden, wovon Beispiele untenstehend gegeben sind. In jedem Fall ist es möglich, auf unterschiedliche Umstände durch eine wiederholte Verwendung der Verschachtelungsvorrichtung, wie obenstehend beschrieben ist, anzupassen. Das gleiche trifft auf die Entschachtelung zu, die benötigt wird, wenn verschiedene Typen eines Verschachtelns implementiert sind.

① Wenn beispielsweise Klassen 1–3 Klassen sind, die zu verschachteln sind, und Klassen 4–6 Klassen sind, die nicht zu verschachteln sind, werden die Bits der Klassendaten der Klasse 1 gestreut und in der Bitsequenz der Klassendaten der Klasse 4 angeordnet, die Bits der Klassendaten der Klasse 2 werden gestreut und in der Bitsequenz der Klassendaten der Klasse 5 angeordnet, und die Bits der Klassendaten der Klasse 3 werden gestreut und in der Bitsequenz der Klassendaten der Klasse 6 angeordnet.

② Wenn beispielsweise die Klassen 1–3 Klassen sind, die zu verschachteln sind, und die Klassen 4–6 Klassen sind, die nicht zu verschachteln sind, werden die Bits der Klassendaten der Klasse 1 gestreut und in der Bitsequenz der Klassendaten der Klasse 4 angeordnet, und dann werden die Bits der Klassendaten der Klasse 2 gestreut und in der sich ergebenden Bitsequenz angeordnet, und dann werden die Bits der Klassendaten der Klasse 3 gestreut und in der sich ergebenden Bitsequenz angeordnet.

B. Zweite Ausführungsform (in Bezug auf die Erfindung)

[0127] Es folgt eine Beschreibung der Fehlerschutzvorrichtung in Bezug auf die zweite Ausführungsform der vorliegenden Erfindung.

[0128] In der oben beschriebenen ersten Ausführungsform, passiert es in Fällen, wo eine Vielzahl von Rahmen dem Übertragungspfad sequentiell geliefert werden, oft, dass unter den Rahmenbildungsdaten identische Ausdrücke sind, die überall in den Rahmen vorhanden sind. Beispielsweise fällt die Anzahl von Bits, die die verschiedenen Klassen bilden, in eine Anzahl von Kategorien in den verschiedenen Rahmen, aber das Kodierungsverhältnis der verschiedenen Klassen und die Anzahl von Fehlererfassungsbits der verschiedenen Klassen sind in den verschiedenen Klassen fixiert. Entsprechend wird, wie in der oben beschriebenen ersten Ausführungsform, wenn die fixierten Daten zu jedem Rahmen als Rahmenbildungsdaten übertragen werden, die Anzahl von Bits, die dem Übertragungspfad von dem Synthetisierer **104** geliefert wird, zunehmen und vergeudet werden.

[0129] Entsprechend werden in der Fehlerschutzvorrichtung in Bezug auf diese zweite Ausführungsform die fixierten Daten unter den Rahmenbildungsdaten nur zu dem Beginn der Übertragung gesendet.

[0130] Fig. 11 ist ein Blockdiagramm, das diese Konfiguration zeigt.

[0131] In der Figur berechnet ein Festdaten-Berechnungsteil **201** die gemeinsamen fixierten Daten in jedem Rahmen unter den Rahmenbildungsbits, und diese fixierten Daten werden zu der Empfangsseitenvorrichtung bei dem Beginn der Übertragung übertragen. Eine Übertragung dieser fixierten Daten wird durch die Verwendung eines Telekommunikationskanals getrennt von demjenigen, der für die Rahmen verwendet wird, erreicht.

[0132] Spezifisch sind die fixierten Daten der vorliegenden Ausführungsform die Daten ECR_1–ECR_n, die das Kodierungsverhältnis des Fehlerkorrekturcodes, der auf jede Klasse angewandt wird, anzeigen, und die Daten EDB_1–EDB_n, die die Anzahl der Bits des Fehlererfassungscodes, der auf jede Klasse angewandt wird, anzeigen. Überdies sind die fixierten Daten vorzugsweise so konfiguriert, dass die andere Seite unter Verwendung einer ARQ (automatische Wiederholanforderung) oder dergleichen sicher sein wird, diese zu empfangen.

[0133] Als nächstes ist ein Festdaten-Speicherteil **202** eine Vorrichtung zum Speichern von fixierten Daten, die von dem Festdaten-Berechnungsteil **201** ausgegeben werden.

[0134] Ein Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102'** entfernt die fixierten Daten, die in dem Festdaten-Speicherteil **202** gespeichert sind, von den Rahmenbildungsdaten, die von dem Rahmenbildungs-Datenberechnungsteil **101** ausgegeben werden, und führt einen spezifizierten Fehlerschutz (Fehlerkorrekturkodierung, das Hinzufügen eines Fehlererfassungscodes, oder beides) bezüglich der übrigen Rahmenbildungsdaten aus. Das heißt, der Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102'** führt einen Fehlerschutz nur für die Daten CB_1–CB_n und CC_1–CC_n von den Rahmenbildungsdaten aus, die von dem Rahmenbildungs-Datenberechnungsteil **101** ausgegeben werden.

[0135] Der Klassendaten-Fehlerschutz-Verarbeitungsteil **103'** führt eine Fehlerkorrekturkodierung bei einem Kodierungsverhältnis aus, das durch die Daten ECR_1–ECR_n spezifiziert ist, die fixierte Daten sind, die in dem Festdaten-Speicherteil **202** gespeichert sind, oder fügt einen Fehlererfassungscode einer Anzahl von Bits, die durch die Daten EDB_1–EDB_n spezifiziert sind, zu den Klassendaten der Klassen 1–n hinzu.

[0136] Der Synthetisierer **104** sammelt die Rahmenbildungsdaten außer den fixierten Daten, die einen Fehlerschutz durch den Rahmenbildungs-Datenfehlerschutz-Verarbeitungsteil **102'** durchlaufen haben und den Klassendaten, die einen Fehlerschutz durch den Klassendaten-Fehlerschutz-Verarbeitungsteil **103'** durchlaufen haben, synthetisiert sie und überträgt sie zu dem Empfänger über den Übertragungspfad. Es sei darauf hingewiesen, dass das Verschachteln, das bezüglich der Klassendaten in diesem Synthetisierer **104** durchgeführt wird, von dem gleichen Typ ist wie in der ersten Ausführungsform.

[0137] Wie oben beschrieben, werden in der vorliegenden Ausführungsform die fixierten Daten (die Daten ECR_1–ECR_n, EDB_1–EDB_n) der Rahmenbildungsdaten nur zu dem Beginn der Übertragung gesendet, während während der Übertragung die Rahmenbildungsdaten außer den fixierten Daten (die Daten CB_1–CB_n, CC_1–CC_n) in jedem Rahmen gesendet werden.

[0138] Die Empfangsseitenvorrichtung empfängt die fixierten Daten zu dem Beginn der Übertragung, nach welcher die Daten, die von dem Synthetisierer **104** ausgegeben werden, empfangen werden. Dann wird die Konfiguration die Rahmens der empfangenen Daten gemäß den fixierten Daten, die anfänglich empfangen werden, und den Rahmenbildungsdaten außer den fixierten Daten, die danach empfangen werden, bestimmt. Das heißt das Kodierungsverhältnis der Klassen, wie auch die Anzahl von Fehlererfassungsbits der Klassen werden durch fixierte Daten bestimmt, und außerdem werden die Anzahl von Bits jeder Klasse, wie auch der Kodierungsalgorithmus jeder Klasse, ein Vorhandensein oder Nicht-Vorhandensein oder ein Typ eines Verschachtelns durch Rahmenbildungsdaten außer den fixierten Daten bestimmt. Deswegen werden in Übereinstimmung mit diesen Daten Daten außer den Rahmenbildungsdaten unter den empfangenen Daten, d. h. die Daten, die eine Fehlerkorrektur durch den Klassendaten-Fehlerschutz-Verarbeitungsteil **103'** durchlaufen haben, dekodiert, was es ermöglicht, die ursprünglichen Daten wiederzugewinnen.

[0139] In Übereinstimmung mit der vorliegenden Ausführungsform werden die fixierten Daten, die sämtlichen Rahmen gemeinsam sind, nur zu dem Beginn der Übertragung anstelle bei jedem Rahmen gesendet, wodurch es ermöglicht wird, ein Hinzufügen einer zusätzlichen Zahl von Bits, die von dem Synthetisierer **104** zu dem Übertragungspfad zu senden sind, zu unterlassen.

[0140] Es sei darauf hingewiesen, dass hinsichtlich der fixierten Daten, obwohl der Fall der Daten ECR_1–ECR_n, die das Kodierungsverhältnis jeder Klasse anzeigen, und der Fall der Daten EDB_1–EDB_n, die die Fehlererfassungsbitzahl jeder Klasse anzeigen, beschrieben wurden, die zweite Ausführungsform nicht darauf beschränkt ist. Beispielsweise können, wenn die Daten CB_1–CB_n, die die Anzahl von Bits anzeigen, die jede Klasse bilden, auch fixierte Daten sind, diese auch als fixierte Daten inkludiert sein. Zusätzlich können, wenn nur entweder die Daten ECR_1–ECR_n, die das Kodierungsverhältnis jeder Klasse anzeigen, oder die Daten EDB_1–EDB_n, die die Anzahl der Fehlererfassungsbits jeder Klasse anzeigen, fixiert sind, diese Daten allein als fixierte Daten verwendet werden.

C. Ausführungsform der Erfindung

[0141] Es folgt eine Beschreibung der Fehlerschutzvorrichtung in Bezug auf die Ausführungsform der vorliegenden Erfindung.

[0142] In der oben beschriebenen ersten oder zweiten Ausführungsform ist der Inhalt eines Abschnitts der Rahmenbildungsdaten auf mehrere Typen begrenzt. Z. B. gibt es den Fall, wo die Kodierungsverhältnisse für Fehlerkorrekturkodierung, die fähig ist, auf die Klassendaten von einigen Klassen angewendet zu werden, nur von wenigen Typen sind, oder den Fall, wo die Zahl von Bits eines Fehlererfassungscodes, die den Klassendaten von einigen Klassen hinzuzufügen sind, auf wenige Typen begrenzt ist. Entsprechend wird in Fällen, wo ein Abschnitt des Inhalts von Rahmenbildungsdaten auf wenige Typen begrenzt ist, gedacht, dass die Zahl von Bits, die zu dem Übertragungspfad gesendet werden, sich erhöht und verschwendet wird, wie in der ersten und zweiten Ausführungsform, die eine Konfiguration haben, die jeden Typen von Rahmenbildungsdaten zu jedem Rahmen sendet.

[0143] Entsprechend werden in der Fehlerschutzvorrichtung in Bezug auf die Ausführungsform der Rahmenbildungsdaten der Erfindung die Typen, die durch den Kodierungsalgorithmus begrenzt sind, der verwendet wird, durch die folgenden Übertragungsverfahren übertragen.

a. Bezüglich Rahmenbildungsdaten von begrenzten Typen werden die Rahmenbildungsdaten von jedem von diesen Typen zum Beginn der Übertragung zusammen mit einem Identifikator, der jeden zugeordnet ist, gesendet.

b. Während einer Übertragung werden die Klas-

sendaten jeder Klasse, die einen Fehlerschutz durchlaufen haben, zusammen mit einem Identifikator zum Bezeichnen der Rahmenbildungsdaten entsprechend jeden und zusätzlichen Daten gesendet.

[0144] Fig. 12 ist ein Blockdiagramm, das eine Konfiguration einer Fehlerschutzvorrichtung in Bezug auf die vorliegende Ausführungsform der Erfindung zeigt.

[0145] In dieser Figur gibt ein Festdaten-Übertragungsteil **301** die Daten ein, die einem Fehlerschutz zu unterziehen sind, und er sendet auch die Rahmenbildungsdaten des Typs, die begrenzt sind, zu dem Kodierungsalgorithmus, der verwendet wird, ebenso wie den Identifikator, der jeden zugeordnet ist, zu der Empfangsseitenvorrichtung unter Verwendung eines Telekommunikationskanals, der sich von dem unterscheidet, der für die Rahmen zum Beginn der jeweiligen Übertragungen verwendet wird.

[0146] Falls z. B. aus den Rahmenbildungsdaten die Kodierungsverhältnisse der Klassen auf zwei Typen begrenzt sind, dann werden zwei Typen von Daten ECR, die den Inhalt davon anzeigen, und zwei Identifikatoren für ihre Identifikation gesendet, und falls des weiteren die Fehlererfassungsbitzahlen der Klassen auf zwei Typen begrenzt sind, werden dann zwei Typen von Daten EDBR, die den Inhalt davon anzeigen, und zwei Identifikatoren für ihre Identifikation gesendet. Es sollte fest gehalten werden, dass es wünschenswert ist, dass diese Daten so konfiguriert sind, dass die andere Seite sicher sein wird, sie unter Verwendung von ARQ oder dergleichen zu empfangen.

[0147] Der Rahmenbildungsdaten-Berechnungsteil **101** ist von dem gleichen Typ wie in den ersten und zweiten Ausführungsformen. Da jedoch die Klassenkonfiguration der Rahmen nicht divergent, sondern begrenzt ist, wie oben erörtert wird, wird auch ein Abschnitt der Rahmenbildungsdaten begrenzt sein.

[0148] Ein Rahmenbildungsdatenkonverter **302** führt die folgende Verarbeitung durch Speichern der Rahmenbildungsdaten, die der Übertragungsinhalt des Festdaten-Übertragungsteils **301** sind, und des Identifikators, der jeden zugeordnet ist, durch. Das heißt der Rahmenbildungsdatenkonverter **302** gibt zusätzliche Daten aus in Bezug darauf, was zu Beginn der Übertragung gesendet wurde, aus den Rahmenbildungsdaten gemäß dem Rahmenbildungsdaten-Berechnungsteil **101**, und bezüglich darauf, was zum Beginn der Übertragung nicht gesendet wurde, nämlich was mit den entsprechenden Identifikatoren ersetzt wird. Das heißt, aus den Rahmenbildungsdaten, bezüglich dessen, was zu Beginn der Übertragung gesendet wurde und durch den Kodierungsalgorithmus begrenzt ist, der verwendet wird, sind es nicht die Daten, die den Inhalt anzeigen, sondern vielmehr was mit dem entsprechenden Identifikator ersetzt wird, was ausgegeben wird, ebenso wie andere Elemente als zusätzliche Daten.

[0149] Die Identifikatoren und ein zusätzlicher Datenfehlerschutz-Verarbeitungsteil **303** führen eine spezifizierte Fehlerkorrektur mit Bezug auf die zusätzlichen Daten und die Identifikatoren von dem Rahmenbildungsdatenkonverter **302** durch. Zu dieser Zeit kann ein Fehlererfassungscode zu der Konfiguration hinzugefügt werden.

[0150] Andererseits bestimmt, wie in der ersten Ausführungsform, der Klassendaten-Fehlerschutz-Verarbeitungsteil **103** den Fehlererfassungscode und Fehlerkorrekturcode in Bezug auf die Klassen 1–n getrennt durch den Rahmenbildungsdaten-Berechnungsteil **101**.

[0151] Außerdem sammelt der Synthetisierer **104** die zusätzlichen Daten und die Identifikatoren, die einer Fehlerkorrektur unterzogen wurden, zusammen mit den Klassen, die einer Fehlerkorrektur unterzogen wurden, synthetisiert sie und überträgt sie zu dem Empfänger über den Übertragungspfad.

[0152] Somit werden in der vorliegenden Ausführungsform die Rahmenbildungsdaten von Typen, die durch die Kodierungsalgorithmen begrenzt sind, die verwendet werden, und die dazu zugeordneten Identifikatoren nur zu Beginn der Übertragung übertragen, während während der Übertragung danach die Daten jeder Klasse, die einer Fehlerkorrektur unterzogen wurden, zusammen mit den Identifikatoren, die ihre Konfigurationen anzeigen, und zusätzlichen Daten übertragen werden.

[0153] Deshalb werden auf der Empfangseite in Bezug auf diese unter den Rahmenbildungsdaten von Typen, die durch den Kodierungsalgorithmus begrenzt sind, der verwendet wird, die Rahmenbildungsdaten dieser Typen und die Identifikatoren, die jeden von ihnen zugeordnet sind, zu Beginn der Übertragung empfangen, sodass es, wenn die Daten von Klassen 1–n zusammen mit den Identifikator und zusätzlichen Daten empfangen werden, möglich wird, die Klassenkonfiguration, die durch diesen Identifikator gekennzeichnet wird, zu kennen. Außerdem empfängt der Empfänger zusätzliche Daten, die nicht mit der anfänglichen Übertragung gesendet wurden, aus den Rahmenbildungsdaten. Somit wird es möglich, Klassen 1–n zu den ursprünglichen Daten im Empfängerende wiederherzustellen.

[0154] In Übereinstimmung mit der vorliegenden Erfindung wird es hinsichtlich der Rahmenbildungsdaten, die von begrenzten Typen sind, da es nicht die Daten selbst sind, die den Inhalt davon anzeigen, sondern der dazu zugeordnete Identifikator, der über-

tragen wird, möglich, die Zahl von Bits stark zu reduzieren, die von dem Synthetisierer **104** zu dem Übertragungspfad gesendet werden.

[0155] Es sollte vermerkt werden, dass in den oben beschriebenen Ausführungsformen der Rahmenbildungsdaten-Fehlerschutz-Verarbeitungsteil **102'** in der oben beschriebenen zweiten Ausführungsform durch den Rahmenbildungsdatenkonverter **302** und Identifikator und den zusätzlichen Datenfehlerschutz-Verarbeitungsteil **303** ersetzt wird, aber der Rahmenbildungsdaten-Fehlerschutz-Verarbeitungsteil **102** der ersten Ausführungsform kann auch durch den Rahmenbildungsdatenkonverter **302** und Identifikator und den zusätzlichen Datenfehlerschutz-Verarbeitungsteil **303** ersetzt werden. In diesem Fall konvertiert, da fixierte Daten nicht gesendet werden, der Rahmenbildungsdatenkonverter **302** zu einem Identifikator diesen Teil der Daten, die zusammen mit dem Identifikator anfangs aus den Rahmenbildungsdaten gesendet werden, die für jeden Rahmen von dem Rahmenbildungsdaten-Berechnungsteil **101** ausgegeben werden, und die anderen Teile werden als zusätzliche Daten ausgegeben.

[0156] Zusätzlich ist es bezüglich der Fehlerschutzvorrichtung in Bezug auf die ersten bis dritten Ausführungsformen zusätzlich zu der Hardware-Typ-Konfiguration, die in den Zeichnungen gezeigt wird, auch möglich, eine Konfiguration aus einem Software-Standpunkt zu realisieren. Um eine Konfiguration von einem Software-Standpunkt zu realisieren, ist es möglich, ein Programm mit der gleichen Operation wie oben unter Verwendung eines Personalcomputers und einer Arbeitsstation auszuführen.

[0157] Überdies werden in der beschriebenen Fehlerschutzvorrichtung in Bezug auf die Ausführungsformen die Rahmen klassenmäßig gemäß der Fehlerempfindlichkeit verschiedener Parameter sortiert, aber die vorliegende Anmeldung ist darauf nicht beschränkt, und die Rahmen können in einem weiten Bereich auf Klassen zutreffen, das heißt, auf Blöcke, die durch Schlitze und dergleichen partitioniert sind.

[0158] Außerdem können die Rahmenbildungsdaten auch Daten enthalten, die die Konfiguration der Klasse, wie etwa die Anzahl von Bits jeder Klasse, wie auch Daten, die die Position jeder Klasse anzeigen, enthalten.

Patentansprüche

1. Ein Fehlerschutzverfahren, die Schritte umfassend:
Sortieren einer Vielzahl von Typen von Parametern, die einen Rahmen bilden, in eine Vielzahl von Klassen;
Generieren von Rahmenbildungsdaten, inkludierend Daten, die für jede Klasse ein Fehlerschutzverfahren spezifizieren, das auf jeden der Parameter und Daten in Bezug auf eine Konfiguration der Rahmen anzuwenden ist;
Ausführen eines Fehlerschutzes, der für jede Klasse spezifiziert ist, gemäß den Rahmenbildungsdaten in Bezug auf Parameter, die in die Vielzahl von Klassen sortiert sind, und
wobei ein Abschnitt der Rahmenbildungsdaten auf einen oder mehr Typen von Daten begrenzt ist,
Übertragen des einen oder mehr Typen von Daten in Bezug auf den Abschnitt der Rahmenbildungsdaten und eines Identifikators, der jedem zugewiesen ist, zum Beginn der Übertragung;
Bestimmen unter den Rahmenbildungsdaten, die für jeden Rahmen generiert werden, welche Elemente mit beliebigen der Daten übereinstimmen, die bereits zusammen mit den Identifikatoren übertragen sind,
Konvertieren der Elemente zu entsprechenden Identifikatoren für eine Übertragung und Ausgeben der Rahmenbildungsdaten außer den bestimmten Elementen als zusätzliche Daten;
Ausführen eines vorbestimmten Fehlerschutzes für die Indikatoren und die ausgegebenen zusätzlichen Daten; und
Übertragen der Parameter, die den Fehlerschutz durchlaufen haben, spezifiziert für jede Klasse zusammen mit den zusätzliche Daten und den Identifikatoren, die den vorbestimmten Fehlerschutz durchlaufen haben.

2. Fehlerschutzverfahren nach Anspruch 1, wobei die Daten, die für jede Klasse ein Fehlerschutzverfahren spezifizieren, das auf jeden der Parameter anzuwenden ist, mindestens eines von einem Kodierungsverhältnis eines Fehlerkorrekturcodes, um auf die Parameter angewendet zu werden, die zu jeder Klasse gehören, einer Zahl von Bits eines Fehlererfassungscodes und Daten in Bezug auf die Ausführung einer Verschachtelung in beliebiger Kombination enthalten.

3. Eine Fehlerschutzeinrichtung, umfassend:
einen Rahmenbildungsdatenberechnungsteil (**101**) zum Sortieren einer Vielzahl von Typen von Parametern, die einen Rahmen bilden, in eine Vielzahl von Klassen, und zum Generieren von Rahmenbildungsdaten, die Daten inkludieren, die für jede Klasse ein Fehlerschutzverfahren spezifizieren, anzuwenden auf jeden der Parameter, und Daten in Bezug auf eine Konfiguration der Rahmen;
einen Klassendatenfehlerschutzverarbeitungsteil (**103**) zum Implementieren eines Fehlerschutzes, spezifiziert für jede Klasse gemäß den Rahmenbildungsdaten, in Bezug auf die Parameter, die in die Vielzahl von Klassen sortiert sind, und
wobei ein Abschnitt der Rahmenbildungsdaten auf einen oder mehr Typen von Daten begrenzt ist,
wobei die Einrichtung ferner umfasst:
einen Festdatenübertragungsteil (**301**) zum Übertragen des einen oder mehr Typen von Daten in Bezug auf den Abschnitt der Rahmenbildungsdaten und ei-

nes Identifikators, der jedem zugeteilt ist, zum Beginn der Übertragung;
einen Rahmenbildungsdatenkonverter (**302**) zum Bestimmen unter den Rahmenbildungsdaten, die für jeden Rahmen generiert sind, welche Elemente mit beliebigen der Daten, die bereits zusammen mit den Identifikatoren übertragen sind, übereinstimmen, Konvertieren der Elemente zu entsprechenden Identifikatoren für eine Übertragung und Ausgeben von Rahmenbildungsdaten außer den bestimmten Elementen als zusätzliche Daten;
ein Mittel (**303**) zum Ausführen eines spezifizierten Fehlerschutzes in den Identifikatoren und in den zusätzliche Daten, die von dem Rahmenbildungsdatenkonverter ausgegeben werden; und
einen Syntheseteil (**104**) zum Übertragen von Parametern, die einen Fehlerschutz durchlaufen haben, der für jede Klasse spezifiziert ist, und Rahmenbildungsdaten, die den spezifizierten Fehlerschutz durchlaufen haben.

4. Eine Fehlerschutzeinrichtung nach Anspruch 3, ferner umfassend:
einen Festdatenberechnungsteil zum Bestimmen fester Daten, die für jeden Rahmen von unter den Rahmenbildungsdaten gemeinsam sind, und zum Übertragen der festen Daten beim Beginn der Übertragung; und
wobei der Rahmenbildungsdatenkonverter die festen Daten aus den Rahmenbildungsdaten entfernt, die für jeden Rahmen durch den Rahmenbildungsdatenberechnungsteil ausgegeben werden, einen Abschnitt, der mit beliebigen der Daten übereinstimmt, die bereits gemeinsam mit den Identifikatoren übertragen sind, unter Rahmenbildungsdaten außer den festen Daten in entsprechende Identifikatoren konvertiert und Rahmenbildungsdaten außer den festen Daten und den konvertierten Abschnitt als zusätzliche Daten ausgibt.

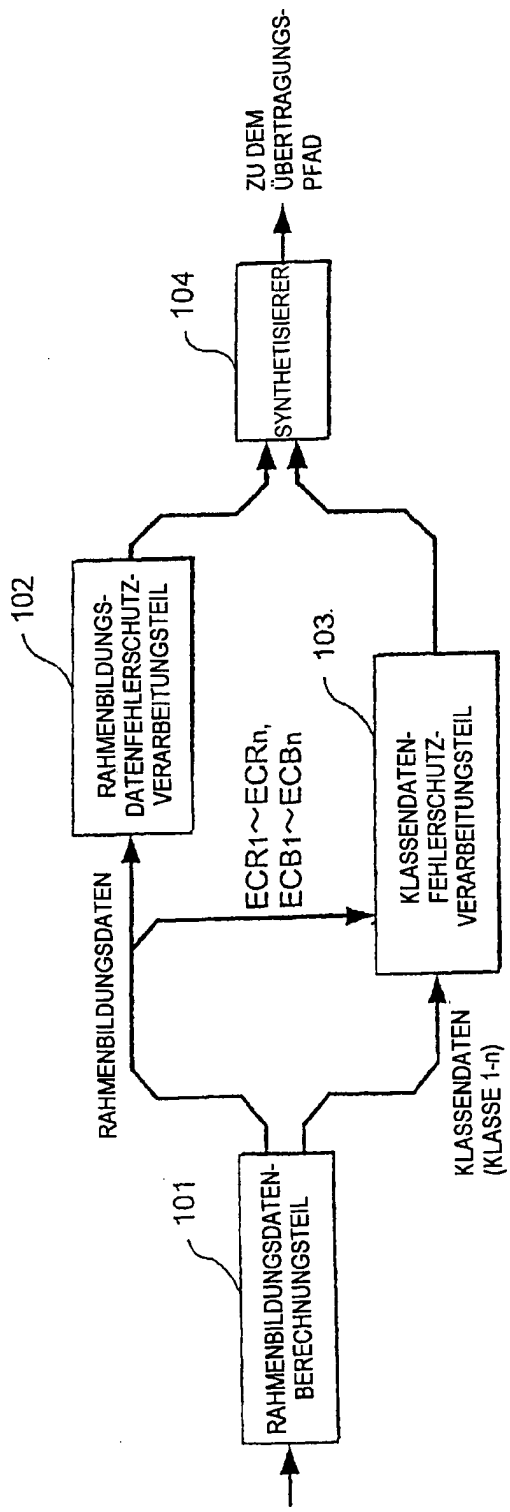
5. Eine Fehlerschutzeinrichtung nach Anspruch 3 oder 4, ferner umfassend:
eine Verschachtelungseinrichtung zum Durchführen der Verschachtelung von Parametern jeder Klasse, die einen Fehlerschutz durch den Klassendatenfehlerschutzverarbeitungsteil durchlaufen haben, und
wobei der Rahmenbildungsdatenberechnungsteil die Rahmenbildungsdaten ausgibt, die Daten enthalten, die den Inhalt der Verschachtelung bestimmen, die auf jede Klasse anzuwenden ist; und
wobei die Verschachtelungseinrichtung die Verschachtelung der Parameter jeder Klasse in Übereinstimmung mit den Rahmenbildungsdaten durchführt.

6. Eine Fehlerschutzeinrichtung nach Anspruch 5, wobei die Verschachtelungseinrichtung Mittel für eine Streuung von Bits, umfassend Parameter einer bestimmten Klasse, und Anordnen der Bits in eine Bitfolge von Parametern einer anderen Klasse umfasst.

Es folgen 12 Blatt Zeichnungen

Anhängende Zeichnungen

FIG. 1



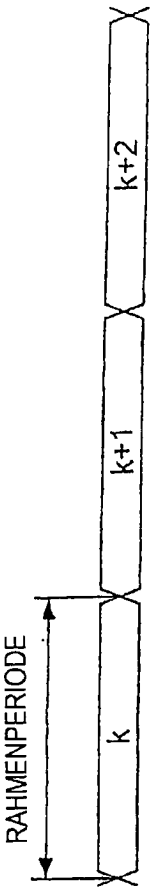


FIG. 2A SPRACHPROBE

FIG. 2B PARAMETERGRUPPEN

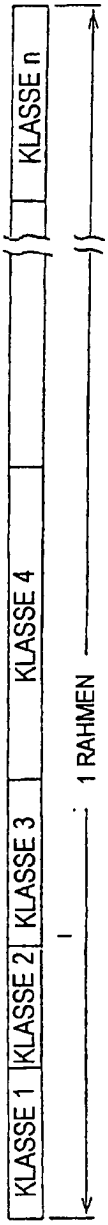


FIG. 2C

FIG. 3

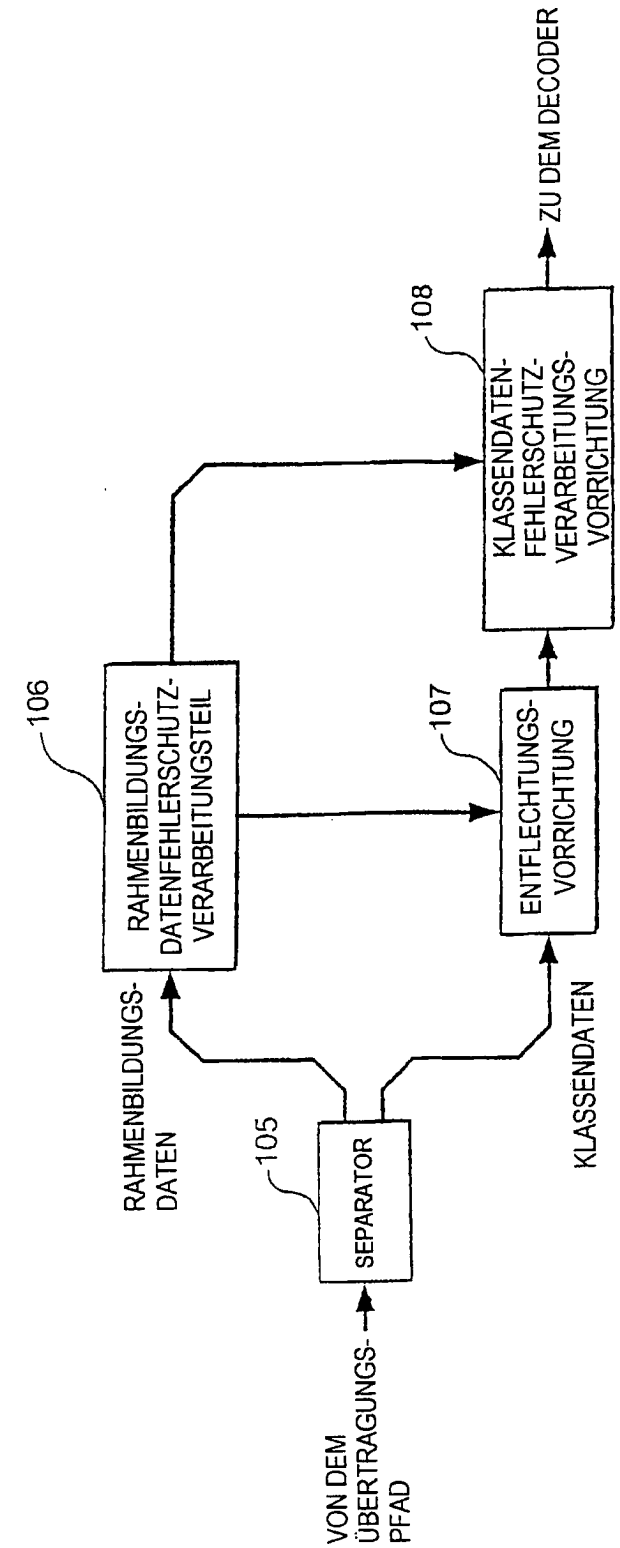

VON DEM ÜBERTRAGUNGS-PFAD
SEPARATOR
105
RAHMENBILDUNGS-DATEN
KLASSENDATEN
RAHMENBILDUNGS-DATENFEHLERSCHUTZ-VERARBEITUNGSTEIL
106
ENTFLECHTUNGS-VORRICHTUNG
107
KLASSENDATEN-FEHLERSCHUTZ-VERARBEITUNGS-VORRICHTUNG
108
ZU DEM DECODER

FIG. 4

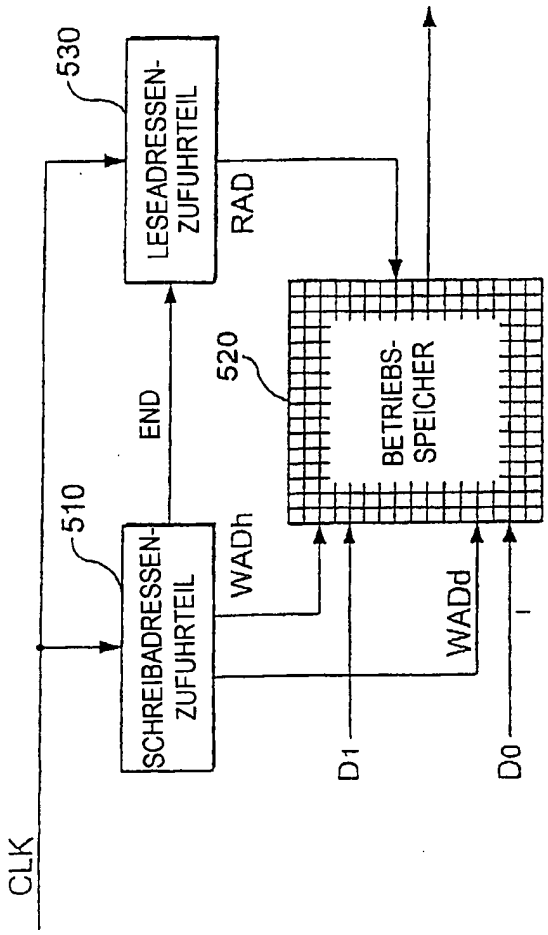

530
LESEADRESSEN-ZUFUHRTEIL
RAD
END
520
BETRIEBS-SPEICHER
510
SCHREIBADRESSEN-ZUFUHRTEIL
WADh
WADd
D1
D0
CLK

FIG. 5

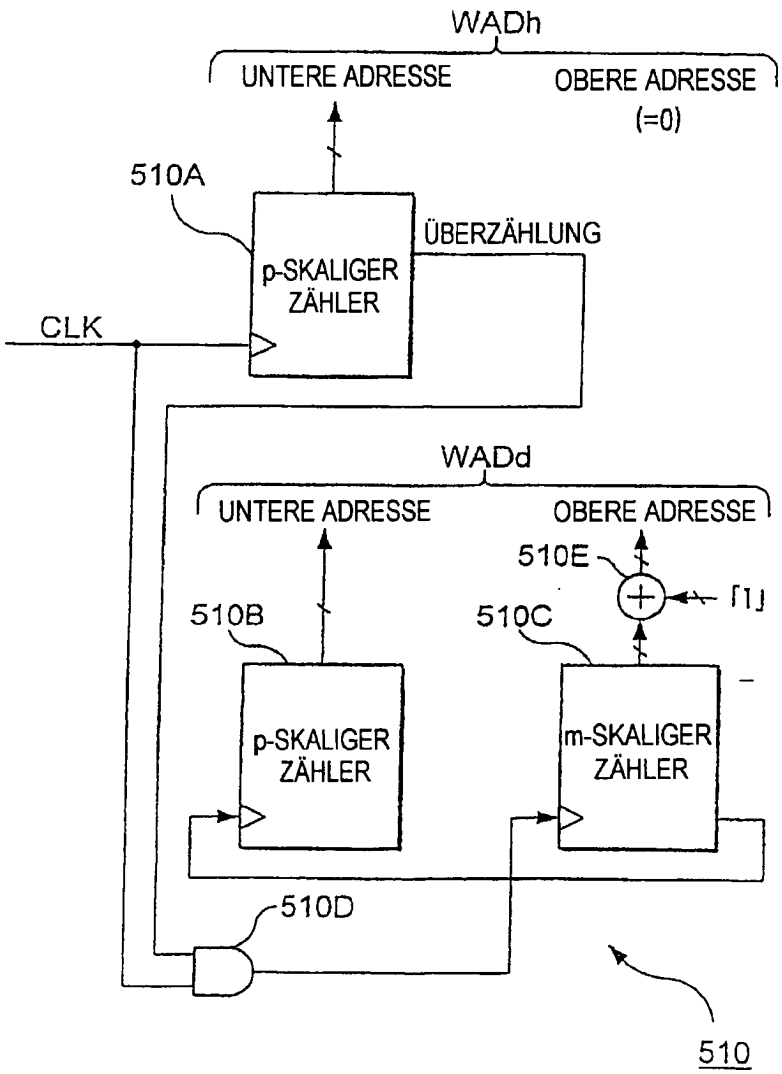

WADh
UNTERE ADRESSE
OBERE ADRESSE
(=0)
510A
p-SKALIGER ZÄHLER
ÜBERZÄHLUNG
CLK
WADd
UNTERE ADRESSE
OBERE ADRESSE
510E
⌈1⌋
510B
510C
p-SKALIGER ZÄHLER
m-SKALIGER ZÄHLER
510D
510

FIG. 6

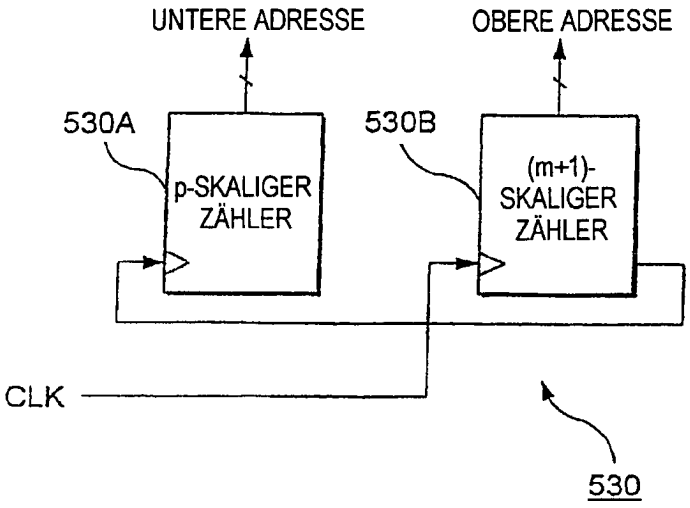

UNTERE ADRESSE
OBERE ADRESSE
530A
p-SKALIGER ZÄHLER
530B
(m+1)-SKALIGER ZÄHLER
CLK
530

FIG. 7

SCHREIBREIHENFOLGE ZU DER ZEIT EINES VERSCHACHTELNS
(LESEREIHENFOLGE ZU DER ZEIT EINES ENTFLECHTENS)

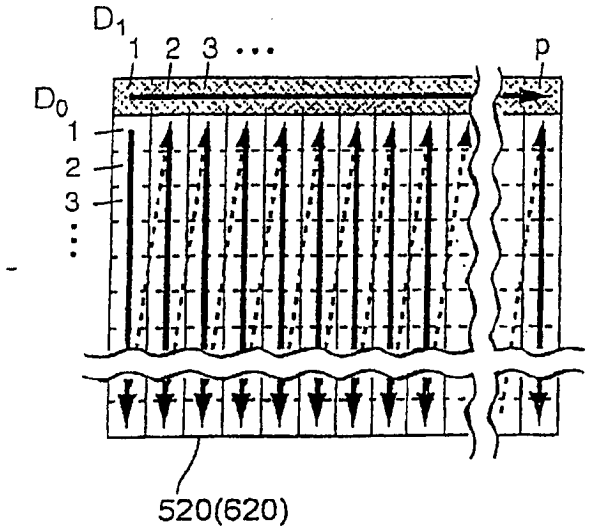


FIG. 8

LESEREIHENFOLGE ZU DER ZEIT EINES VERSCHACHTELNS
(SCHREIBREIHENFOLGE ZU DER ZEIT EINES ENTFLECHTENS)

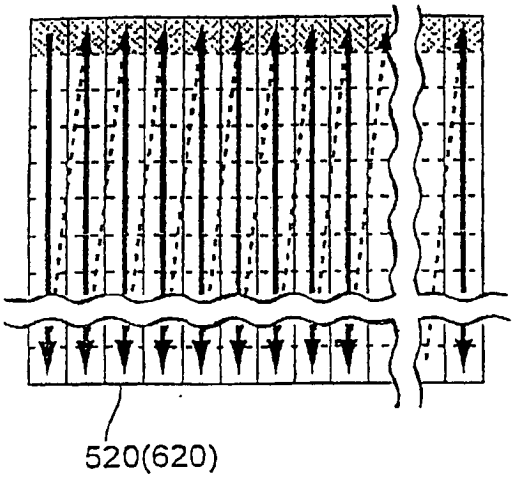


FIG. 9A

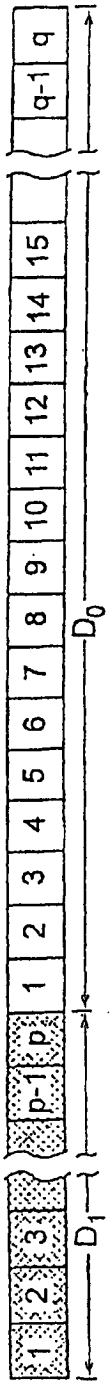

VOR VERSCHACHTELN
1
2
3
p-1
p
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
q-1
q
D_1
D_0

FIG. 9 B

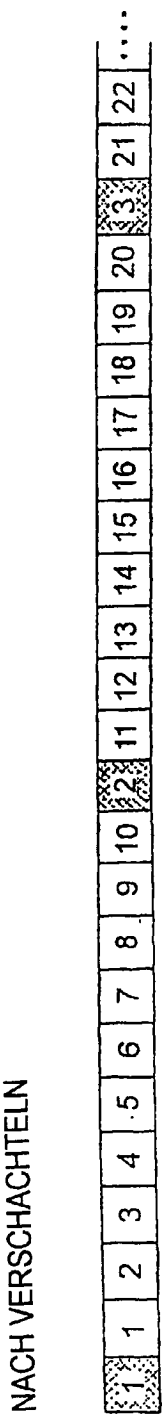

NACH VERSCHACHTELN
1
1
2
3
4
5
6
7
8
9
10
23
11
12
13
14
15
16
17
18
19
20
3
21
22
....

FIG. 10

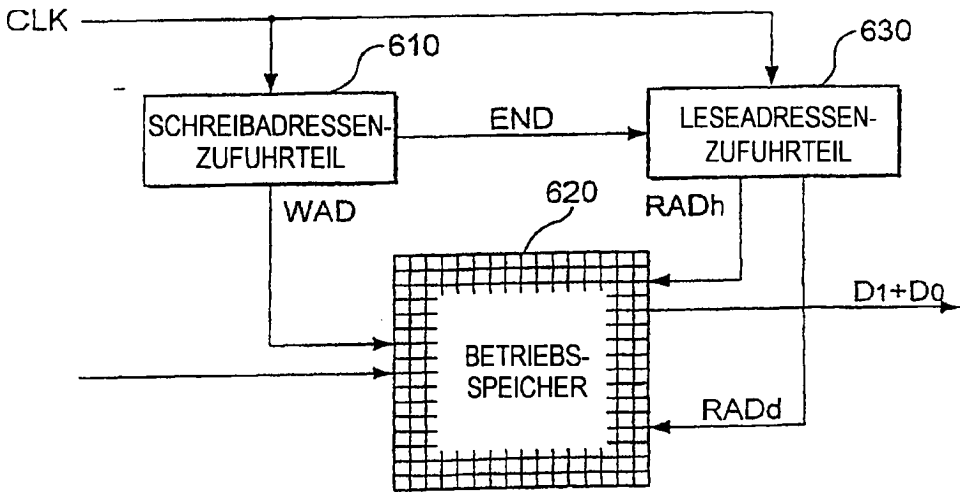

CLK
610
630
SCHREIBADRESSEN-ZUFUHRTEIL
END
LESEADRESSEN-ZUFUHRTEIL
WAD
620
RADh
D1+D0
BETRIEBS-SPEICHER
RADd

FIG. 11

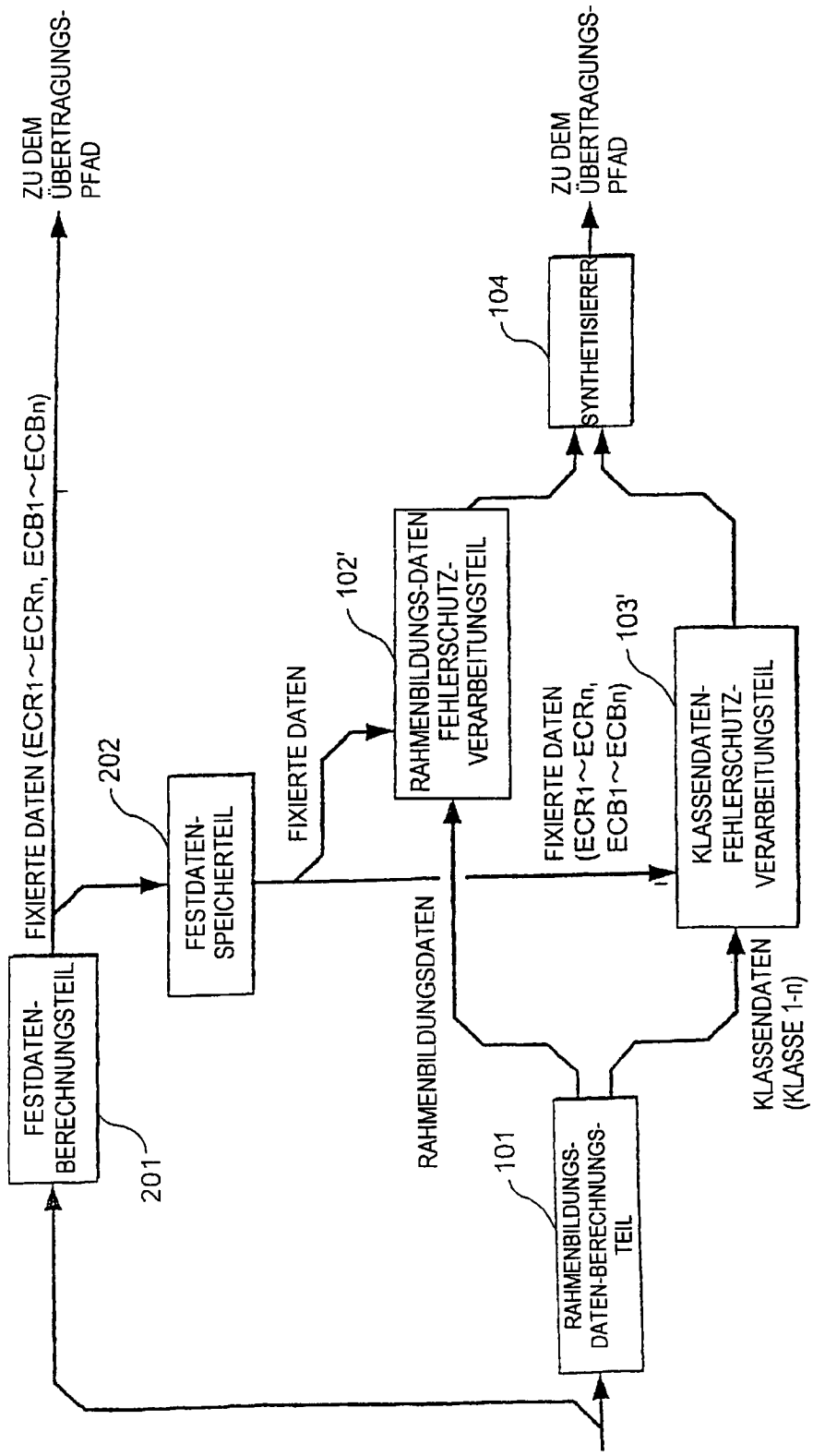

ZU DEM ÜBERTRAGUNGS-PFAD
FIXIERTE DATEN (ECR1~ECRn, ECB1~ECBn)
FESTDATEN-BERECHNUNGSTEIL
201
202
FESTDATEN-SPEICHERTEIL
FIXIERTE DATEN
102'
RAHMENBILDUNGS-DATEN FEHLERSCHUTZ-VERARBEITUNGSTEIL
104
SYNTHETISIERER
ZU DEM ÜBERTRAGUNGS-PFAD
FIXIERTE DATEN (ECR1~ECRn, ECB1~ECBn)
103'
KLASSENDATEN-FEHLERSCHUTZ-VERARBEITUNGSTEIL
RAHMENBILDUNGSDATEN
101
RAHMENBILDUNGS-DATEN-BERECHNUNGS-TEIL
KLASSENDATEN (KLASSE 1-n)

FIG. 12

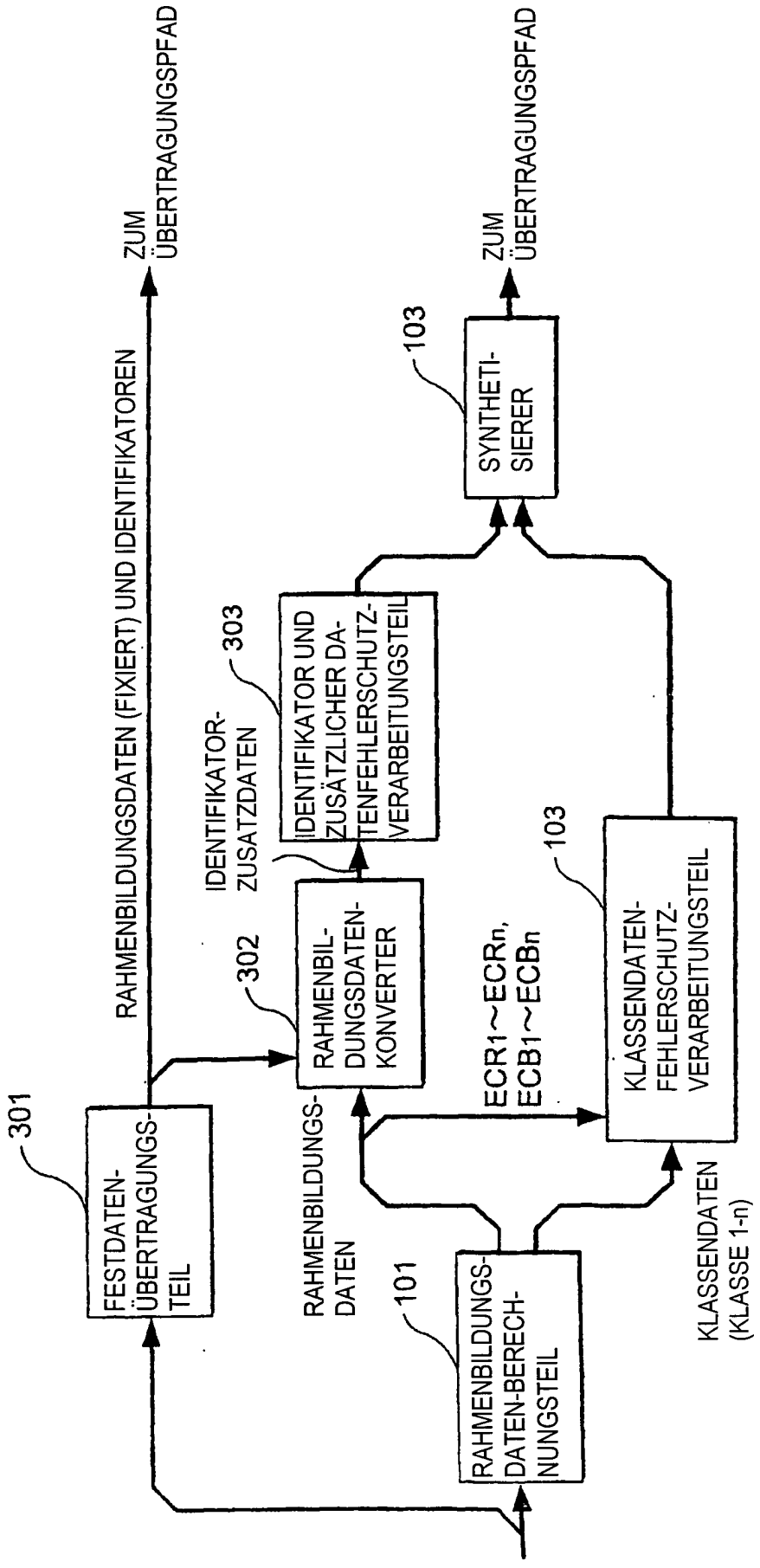
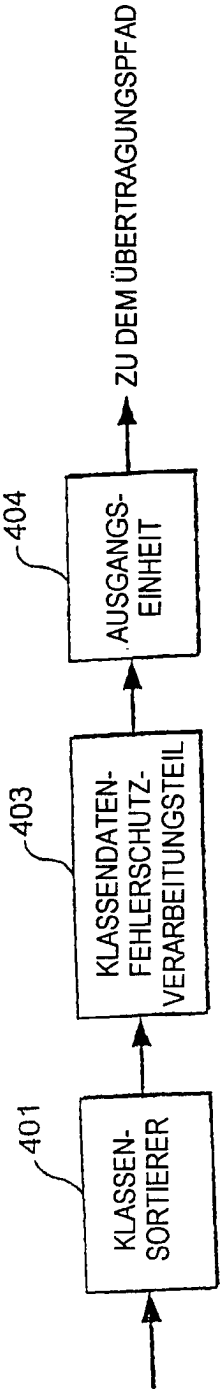

ZUM ÜBERTRAGUNGSPFAD
RAHMENBILDUNGSDATEN (FIXIERT) UND IDENTIFIKATOREN
301
FESTDATEN-ÜBERTRAGUNGS-TEIL
302
RAHMENBIL-DUNGSDATEN-KONVERTER
IDENTIFIKATOR-ZUSATZDATEN
303
IDENTIFIKATOR UND ZUSÄTZLICHER DA-TENFEHLERSCHUTZ-VERARBEITUNGSTEIL
103
SYNTHETI-SIERER
ZUM ÜBERTRAGUNGSPFAD
RAHMENBILDUNGS-DATEN
ECR1~ECRn, ECB1~ECBn
103
KLASSENDATEN-FEHLERSCHUTZ-VERARBEITUNGSTEIL
101
RAHMENBILDUNGS-DATEN-BERECH-NUNGSTEIL
KLASSENDATEN (KLASSE 1-n)

FIG. 13


401
KLASSEN-SORTIERER
403
KLASSENDATEN-FEHLERSCHUTZ-VERARBEITUNGSTEIL
404
AUSGANGS-EINHEIT
ZU DEM ÜBERTRAGUNGSPFAD