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

(10) **DE 698 14 465 T2** 2004.04.01

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 048 034 B1**
(21) Deutsches Aktenzeichen: **698 14 465.1**
(86) PCT-Aktenzeichen: **PCT/GB98/00055**
(96) Europäisches Aktenzeichen: **98 900 557.4**
(87) PCT-Veröffentlichungs-Nr.: **WO 99/036913**
(86) PCT-Anmeldetag: **16.01.1998**
(87) Veröffentlichungstag der PCT-Anmeldung: **22.07.1999**
(97) Erstveröffentlichung durch das EPA: **02.11.2000**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **07.05.2003**
(47) Veröffentlichungstag im Patentblatt: **01.04.2004**

(51) Int Cl.[7]: **G11B 20/18**

(73) Patentinhaber:
Hewlett-Packard Co. (n.d.Ges.d.Staates Delaware), Palo Alto, Calif., US; Sony Corp., Tokio/Tokyo, JP

(74) Vertreter:
Schoppe, Zimmermann, Stöckeler & Zinkler, 82049 Pullach

(84) Benannte Vertragsstaaten:
DE, FR, GB

(72) Erfinder:
MORLEY, Stephen, Thornbury, Bristol BS35 2PX, GB; WILLIAMS, Robert, Bradley Stoke, GB; OSAKI, Shinya, Atsugi-shi, Kanagawa-ken 243, JP; WATKINS, Robert, Mark, Sneyd Park, GB; HIROSE, Toshiyuki, 6-chome, JP

(54) Bezeichnung: **VERFAHREN UND GERÄT ZUR DATENSPEICHERUNG AUF MAGNETISCHEN MEDIEN, DIE FEHLERKORREKTURKODES ENTHALTEN**

Beschreibung

Gebiet der Erfindung

[0001] Diese Erfindung bezieht sich auf Vorrichtungen und Verfahren zum Speichern von Daten auf Magnetmedien und insbesondere auf solche Vorrichtungen und Verfahren, die zumindest zwei Fehlerkorrekturebenen implementieren.

Hintergrund der Erfindung

[0002] Es gibt eine ständig steigende Nachfrage nach Magnetdatenspeichervorrichtungen, die in der Lage sind, große Mengen an digitalen Daten, wie z. B. Computerdaten, durch das DDS- (digitale Datenspeicherung-) Format auf Magnetband zu speichern. In einem DDS-Lese/Schreibmechanismus, der das obige Format verwendet, werden Daten in einer Schrägspurweise auf einem länglichen Aufzeichnungsmedium aufgezeichnet, das ein Band umfaßt, das mit einem Magnetmedium beschichtet ist, durch eine Drehtrommel, die einen oder mehrere elektromagnetische Köpfe trägt. Das Band wird durch eine motorgetriebene Antriebsrolle entlang einem Weg bewegt, der sich zwischen zwei Spulen oder Rollen erstreckt und teilweise um die Trommel gewickelt ist. Die Rotationsebene der Köpfe der Trommel ist in einem Winkel zu der Bewegungsebene des Bands angeordnet, so daß jeder Kopf das Band entlang aufeinanderfolgenden Spuren überquert, die sich über die Breite des Bands erstrecken, in einem Winkel zu der Mittellinie desselben. Der Mechanismus umfaßt eine geeignete Schaltungsanordnung zum Codieren von Daten in Signale, die zum Aufzeichnen auf Band geeignet sind, einschließlich Fehlererfassungs- und Korrekturcode, und zum Konditionieren dieser Signale in eine Form, die optimal an die Charakteristika des Aufzeichnungsmedium angepaßt ist. Für die Datenwiedergewinnung ist eine zusätzliche Schaltungsanordnung vorgesehen, zum Erfassen von Magnetfeldschwankungen, die auf dem Band gespeichert sind, zum Ableiten von entsprechenden Signalen, zum Konditionieren dieser Signale in eine Form für die nachfolgende Verarbeitung, zum Decodieren der codierten Daten und zum Erfassen und Korrigieren von Fehlern.

[0003] Eine aktuelle Entwicklung ist das DDS3-Format (definiert in ECMA Standard ECMA-236 vom Juni 1996 „3,81 mm Wide Magnetic Tape Cartridge for Information Interchange – Helical Scan Recording – DDS-3 Format using 125 m Length Tapes", dessen gesamte Inhalte hierin durch Bezugnahme aufgenommen sind).

[0004] Bei DDS3 wird eine Fehlerprüfung und -erfassung erreicht durch Verwenden eines verschachtelten Mehrebenen-Reed-Solomon-Codes, der zumindest zwei Ebenen (C1, C2) und optional eine dritte Ebene (C3) von Fehlerkorrekturcodierung (ECC = error correction coding) liefert. Um außerdem eine endgültige Datenprüfung beim Lesen zu liefern, werden Spurprüfsummen erzeugt, die jeder Datenspur entsprechen, die auf Band geschrieben werden soll, und in den Fragmentanfangsblöcken gespeichert, von denen mehrere in jeder Spur enthalten sind. Somit wird beim Lesen der Spurprüfsummenalgorithmus an die Datenbytes angelegt, die von dem Band wiedergewonnen werden, und falls diese Spursumme nicht mit der übereinstimmt, die berechnet und gespeichert wurde, als das Band aufgezeichnet wurde, wird die Spur zurückgewiesen.

Zusammenfassung der Erfindung

[0005] Es ist wünschenswert, daß die Spurprüfsumme arbeitet, um einen so großen Anteil wie möglich der ausgefallenen Spuren (das sind diejenigen, die entweder nicht korrigierbare oder falsch korrigierte Codewörter aufweisen – wie es nachfolgend beschrieben wird) zurückzuweisen. Obwohl die bestehende Spurprüfsumme relativ gut arbeitet, haben die Erfinder herausgefunden, daß es einen unerwarteten und doch wesentlichen Nachteil bei diesem System gibt, so daß die Spurprüfsumme einen großen Anteil von Fehlkorrekturen nicht identifiziert. Die Analyse der Anmelder hat offenbart, daß bei einem falsch korrigierten Codewort, bei dem die „Korrekturen" nur in den Datenbytes des Codeworts auftreten, die Spurprüfsumme diesen Ausfall nicht offenbaren wird. Dieses Phänomen ergibt sich, weil die Codewörter aufgebaut sind, um die inhärente Eigenschaft aufzuweisen, daß das Ergebnis einer XOR-Verknüpfung aller Bytes zusammen (sowohl Daten als auch Parität) 0 ist. Außerdem werden bei DDS3 die Spurprüfsummen durch XOR-Verknüpfung der Datenbytes der relevanten Spur berechnet.

[0006] Vorausgesetzt, daß die Bytes eines guten (oder falsch korrigierten) Codeworts einer XOR-Verknüpfung zu 0 unterzogen werden, falls die Paritätsbytes eines speziellen guten oder falsch korrigierten Codeworts einer XOR-Verknüpfung zu beispielsweise einem binären Wert A unterzogen werden, müssen die Datenbytes auch einer XOR-Verknüpfung zu dem gleichen Wert A unterzogen werden (so daß die Datenbytes und die Paritätsbytes zusammen einer XOR-Verknüpfung zu 0 unterzogen werden). Bei einer Fehlkorrektur dieses speziellen Codeworts, bei dem die „Korrekturen" nur in den Datenbytes enthalten sind, werden die Paritätsbytes wie vorher einer XOR-Verknüpfung zu A unterzogen, weil dieselben unverändert sind. Obwohl sich die Datenbytes geändert haben, werden die Datenbytes nach wie vor einer XOR-Verknüpfung zu A unterzogen, um die

Anforderung zu erfüllen, daß die Fehlkorrektur einer XOR-Verknüpfung zu 0 unterzogen wird. Da die Datenbytes jedoch alle zu der gleichen Prüfsumme (also dem Ergebnis einer XOR-Operation) beitragen, bleibt die Spurprüfsumme unverändert, und daher wird die Spurprüfsumme für diesen Typ von Fehlkorrektur kein falsch korrigiertes Codewort offenbaren, bei dem alle die „Korrekturen" in den Datenbytes erscheinen, aufgrund der Korrelation zwischen der inhärenten XOR-Eigenschaft des Codeworts und der Operation, die verwendet wird, um die Prüfsumme zu berechnen.

[0007] Die Anmelder haben bestimmt, daß das Modifizieren des Spurprüfsummenalgorithmus, so daß derselbe nicht mit der XOR-Operation korreliert, ein zuverlässiges Verfahren zum Erfassen von Fehlkorrekturen bei dem vorhergehenden Codewort liefert. Die Fähigkeit, Fehlkorrekturen zuverlässig zu erfassen, hat auch wichtige vorteilhafte Konsequenzen, wenn eine Dritte-Ebene-C3-Korrektur implementiert wird.

[0008] In der Vergangenheit haben die Anmelder versucht, die Spurprüfsumme zu verwenden, um C2-Codewortversagen zu identifizieren und dieselben entsprechend zu markieren. Falls ein Codewort an der C1- oder der C2-Ebene versagt (bei einem 3-Ebenen-System) identifiziert die Kenntnis der Position dieses Codeworts in dem Array, aufgrund der verschachtelten und Mehrebenenstruktur der Fehlerkorrekturcodierung, in den Codewörtern der nächsten Ebene die Positionen einer Anzahl von Bytes, die verdächtig sind. Die Struktur der Fehlerkorrekturcodierung bedeutet, daß die Bytes in einem speziellen Codewort auf Positionen in den nachfolgenden Codewörtern gemäß einer bekannten Abbildung abbilden. Somit kann ein ausgefallenes Codewort an der C2-Stufe verwendet werden, um spezielle Bytes in dem C3-Codewort für den C3-Korrekturalgorithmus zu kennzeichnen, so daß diese als „Löschungen" anstatt als Fehler behandelt werden.

[0009] Wenn somit ein vollständiges C2-Codewort als ein Versagen markiert ist, können die entsprechenden Datenbytepositionen, die das Codewort bilden bestimmt werden, so daß der nächsten C3-Ebene, die Position der Fehler bekannt ist. Ein Reed-Solomon-Code mit N Paritätsbytes ist in der Lage, „e" Fehler und „v" Löschungen zu korrigieren, wobei $2e + v \leq N$, und eine Löschung ist ein schlechtes Byte in einer bekannten Position. Ein typisches C3-Korrekturcodewort hat lediglich zwei Paritätsbytes (d. h. $N = 2$), und daher kann der C3-Korrekturalgorithmus entweder einen einzigen Fehler ($e = 1$) oder zwei Löschungen ($v = 2$) korrigieren. Wo zwei Korrekturen durchgeführt werden, wird dies als Doppelfehlerkorrektur bezeichnet. Somit würde die Fähigkeit, falsch korrigierte C2-Codewörter zuverlässig als Löschungen zu markieren, bedeuten, daß der C3-Algorithmus eine Doppelfehlerkorrektur durchführen könnte. Bis jetzt hat das Nichtvorhandensein eines zuverlässigen Verfahrens zum Erfassen von Fehlkorrekturen bedeutet, daß es nicht realistisch war, eine Doppelfehlerkorrektur an der C3-Stufe zu versuchen.

[0010] Folglich haben die Anmelder ein Verfahren und eine Vorrichtung geliefert, bei der die Spurprüfsumme ein wesentlich zuverlässigeres Prüfen von Fehlkorrekturen liefert.

[0011] Bei einem Aspekt liefert diese Erfindung eine Vorrichtung zum Speichern eines Stroms von Datenaufzeichnungen auf Magnetmedien, wobei die Vorrichtung folgende Merkmale umfaßt:

eine Gruppenformatiereinrichtung zum Gruppieren der Datenaufzeichnungen in Gruppen von Datenbytes;

eine Untergruppenverarbeitungseinrichtung zum Teilen jeder der Gruppen in Untergruppen, wobei jede Untergruppe Datenbytes umfaßt, die einer oder mehreren Datenspuren entsprechen;

eine Spurprüfsummenberechnungseinrichtung zum Anlegen der Datenbytes von jeder Datenspur an einen Prüfsummenalgorithmus zum Berechnen einer oder mehrerer Prüfsummen für jede Datenspur;

eine Einrichtung zum Transformieren jeder Untergruppe in zumindest ein jeweiliges Array, wobei jedes einer Datenspur entspricht,

eine Erste-Ebene-Fehlerkorrekturcodierungs- (ECC = error correction coding) Codiereinrichtung zum Codieren von Spalten des oder jedes Arrays zum Liefern erster (C1) ECC-Codewörter, die jeweils Datenbytes und Paritätsbytes umfassen, die eine Erste-Ebene-Korrektur von zumindest einem der Datenbytes in jedem der Erste-Ebene-ECC-Codewörter umfaßt;

eine Zweite-Ebene-Fehlerkorrekturcodierungs- (ECC-) Codiereinrichtung zum Codieren von Zeilen des oder jedes Arrays zum Liefern zweiter (C2) ECC-Codewörter, die Datenbytes und Paritätsbytes umfassen, die eine Zweite-Ebene-Korrektur von zumindest einem der Datenbytes in den Zweite-Ebene-ECC-Codewörtern ermöglichen, und wobei die Datenbytes und die Paritätsbytes, die jedes gegebene zweite ECC-Codewort bilden, zumindest einer vorbestimmten ECC-Regel gehorchen;

wobei der Algorithmus, der durch die Spurprüfsummenberechnungseinrichtung angelegt wird, ausgewählt ist, um nicht mit der vorbestimmten ECC-Regel zu korrelieren, und dadurch eine Prüfsumme zu liefern, die die Eigenschaft aufweist, daß beim Decodieren der ECC-Codewörter und beim Neuberechnen und Prüfen der Spurprüfsumme die Wahrscheinlichkeit, daß ein falsch korrigiertes Codewort ein Spurprüfsummenversagen bewirkt, erhöht ist.

[0012] Es wird betont, daß es die Begriffe „erster" und „zweiter" nicht erforderlich machen, daß die erste Fehlerkorrekturcodiereinrichtung vor der zweiten Fehlerkorrekturcodierungseinrichtung arbeitet.

[0013] Auf diese Weise bedeutet das Auswählen eines Spurprüfsummenalgorithmus, der keine Korrelation mit den Codeworterzeugungsregeln hat, daß die Spurprüfsumme ein zuverlässigerer Indikator für vorhergehende Fehlerkorrekturen ist als der vorhergehende Algorithmus. Dies macht es wiederum möglich, die Spur-

prüfsummen zu verwenden, um falsch korrigierte Codewörter mit wesentlich verbesserten Korrekturraten zu markieren.

[0014] Diese erste und zweite Fehlerkorrekturcodiereinrichtung legen vorzugsweise jeweilige Reed-Solomon-Codieralgorithmen an.

[0015] Es gibt viele Algorithmen, die verwendet werden können, um die Spurprüfsumme zu erzeugen; bei einem speziellen Beispiel, wo Reed-Solomon-Codewörter mit einer Wurzel bei α^0 verwendet werden, und das erste und das zweite ECC-Codewort die Regel befolgen, daß alle die Bytes in einem bestimmten Codewort einer XOR-Verknüpfung zu 0 unterzogen werden, kann die Spurprüfsumme auf der Basis der arithmetischen Addition der Datenbytes in der relevanten Spur berechnet werden.

[0016] Die Codewörter und die Prüfsummen können vor dem Schreiben auf Band in einer Vielzahl von Möglichkeiten verarbeitet werden, aber es wird bevorzugt, daß die Vorrichtung eine Einrichtung zum Transformieren der codierten Arrays umfaßt, um jeder Spur eine Mehrzahl von Datenfragmenten zu liefern, die auf das Magnetmedium geschrieben werden, und eine Fragmenganfangsblockeinrichtung zum Versehen jedes der Datenfragmente mit einem Fragmentanfangsblock, wobei zumindest einige der Fragmentanfangsblöcke Datenbytes umfassen, die die entsprechende Spurprüfsumme für die aktuelle Spur identifizieren.

[0017] Die Bereitstellung eines zuverlässigeren Spurprüfsummenschemas zum Erfassen von Fehlkorrekturen ermöglicht eine Doppelfehlerkorrektur auf einer dritten Ebene. Folglich umfaßt die Vorrichtung vorzugsweise eine dritte Fehlerkorrekturcodiereinrichtung zum Berechnen von Codewörtern von jeweiligen entsprechenden Bytepositionen über jede der Spuren, die eine Gruppe bilden.

[0018] Die Erfindung erstreckt sich auch auf ein Verfahren zum Speichern eines Stroms von Datenaufzeichnungen auf einem Magnetband, das folgende Schritte umfaßt:

Gruppieren der Datenaufzeichnungen in Gruppen von Datenbytes;

Dividieren jeder der Gruppen in Untergruppen von Datenbytes, wobei jede Untergruppe zumindest einer Datenspur entspricht;

Anlegen der Datenbytes von jeder Datenspur an einen Prüfsummenalgorithmus zum Berechnen einer oder mehrerer Spurprüfsummen für die oder für jede Datenspur;

Speichern der einen oder mehreren Spurprüfsummen;

Transformieren der Untergruppe in zumindest ein jeweiliges Array, wobei jedes Array einer Datenspur entspricht;

Codieren von Spalten des oder jedes Arrays, um Erste-Ebene-ECC-Codewörter zu bilden, die Datenbytes und Paritätsbytes umfassen, die eine Erste-Ebene-Korrektur von zumindest einem der Datenbytes in den Erste-Ebene-ECC-Codewörtern ermöglichen;

Codieren von Zeilen der Arrays, um Zweite-Ebene-Fehlerkorrekturcodierwörter zu bilden, die Datenbytes und Paritätsbytes umfassen, die eine Zweite-Ebene-Korrektur von zumindest einem der Datenbytes in dem Zweite-Ebene-ECC-Codewort erlauben, und wobei die Datenbytes und die Paritätsbytes, die jedes gegebene ECC-Codewort bilden, zumindest einer vorbestimmten ECC-Regel gehorchen;

wobei der Algorithmus, der durch die Spurprüfsummenberechnungseinrichtung angelegt wird, ausgewählt ist, um nicht mit der vorbestimmten ECC-Regel zu korrelieren, um eine Prüfsumme zu liefern, bei der beim Decodieren der ECC-Codewörter und beim Neuberechnen und Prüfen der Spurprüfsumme die Wahrscheinlichkeit, daß ein falsch korrigiertes Codewort ein Spurprüfsummenversagen bewirkt, erhöht ist.

[0019] Die Erfindung erstreckt sich auch auf ein Verfahren zum Lesen von Daten, die gemäß dem obigen Verfahren gespeichert sind, was das Wiedergewinnen von Daten von Band, das Extrahieren der Spurprüfsummen von denselben, das Decodieren der Codewörter, um Datenbytes für jede Spur zu erhalten, das Berechnen einer Spurprüfsumme für die decodierten Datenbytes und das Kennzeichnen eines Versagens umfaßt, falls die Prüfsummen nicht übereinstimmen.

[0020] Vorzugsweise umfassen die Daten drei Ebenen von Fehlerkorrektur, und die Spurprüfsumme wird nach der zweiten Ebene der Fehlerkorrektur verwendet, um alle zweiten Codewortversagen zu kennzeichnen, um dadurch alle Datenbytes in dem Dritte-Ebene-Codewort, die einem Zweite-Ebene-Codewortversagen entsprechen, zu identifizieren, und für den Dritte-Ebene-Korrekturalgorithmus als Löschungen zu markieren.

Kurze Beschreibung der Zeichnungen

[0021] Die Erfindung kann auf verschiedene Weisen durchgeführt werden, und ein Ausführungsbeispiel derselben wird nun detailliert beschrieben, wobei Bezugnahme auf die beiliegenden Zeichnungen genommen wird. Es zeigen:

[0022] **Fig.** 1 ein schematisches Blockpipelinediagramm, das die Speicherung und Wiedergewinnung von Informationen von einem Magnetband unter Verwendung eines modifizierten DDS3-Formats gemäß dieser Erfindung darstellt;

[0023] **Fig.** 2 die G2 Untergruppen, die von der Basisgruppe in dem Datenformat abgesplittet sind, das in dem DDS-3-Format implementiert ist;

[0024] **Fig.** 3 die G3 Untergruppen, die nach der Randomisierung und Neuanordnung der G1-Untergruppen erhalten werden;
[0025] **Fig.** 4 die Struktur der G4 Untergruppen, die von den G3-Untergruppen erhalten werden;
[0026] **Fig.** 5 die Struktur eines Hauptdatenfragments; und
[0027] **Fig.** 6 die Struktur des Stapelelements (Pack Item), das die Spurprüfsummen enthält.

Detaillierte Beschreibung des bevorzugten Ausführungsbeispiels

[0028] Das nachfolgend beschriebene Ausführungsbeispiel eines Bandspeichersystems basiert auf dem DDS-3-Standard, der in ECMA 236 beschrieben ist, der einen Dritte-Ebene- (C3-) Korrekturalgorithmus und einen modifizierten Spurprüfsummenalgorithmus umfaßt, der eine Prüfsumme liefert, die ein zuverlässigerer Indikator von C2-Fehlkorrekturen ist. Die Spurprüfsumme, die gemäß dem modifizierten Algorithmus bestimmt wird, kann als Endprüfung der Primärdaten verwendet werden, nach der C2-Korrektur (falls keine C3-Korrektur verwendet wird) oder nach der C3-Korrektur. Wie es nachfolgend näher beschrieben wird, in dem Fall, wo eine C3-Korrektur nicht verwendet wird, liefert die Verwendung des modifizierten Spurprüfsummenalgorithmus ein sehr viel zuverlässigeres Verfahren zum Erfassen von C2-Fehlkorrekturen als eine Enddatenprüfung. In dem Fall, wo eine C3-Korrektur verwendet wird, bedeutet die Fähigkeit, C2-Fehlkorrekturen zuverlässig zu prüfen und zu markieren, daß sowohl unkorrigierbare C2-Codewörter als auch C2-Fehlkorrekturen als Löschungen für C3 markiert werden können, um es zu ermöglichen, daß die volle Leistung des C3-Korrekturcodes verwendet wird.
[0029] Mit anfänglicher Bezugnahme auf **Fig.** 1 wird ein Strom von Primärdaten, die auf dem Band gespeichert werden sollen, durch ein Basisgruppenmodul **10** in Basisgruppen von 384.296 Bytes gruppiert. Nach dem Basisgruppenmodul **10** folgt ein C3-Korrekturprozessor **14** und ein Spurprüfsummengenerator **20**. Da der C3-Korrekturprozessor **14** und der Prüfsummengenerator **20** leichter unter Verwendung der Bytenomenklatur beschrieben werden können, die durch die Untergruppen eingeführt wird, werden diese Komponenten der Zweckmäßigkeit halber nachfolgend näher beschrieben.
[0030] Wenn eine Basisgruppe fertiggestellt ist, wird dieselbe durch das G1-Modul **12** in **22** G1-Untergruppen aufgeteilt, jede mit 17.648 Bytes, numeriert von 0 bis 17.467. Jede G1-Untergruppe hat eine laufende Nummer in dem Bereich von 1 bis 22 (siehe **Fig.** 2). Ein Fehlerkorrekturcode- (ECC3-) Prozessor **14** leitet Daten von jeder der **22** G1-Untergruppen ab, um eine 23te G1-Untergruppe zu bilden. Der Fehlerkorrekturcode C3 ist ein GF(2^8) Reed-Solomon-Code (46, 44, 3). Die Berechnung in einem GF(2^8) soll definiert werden durch:

$$G(x) = x^8 + x^9 + x^3 + x^2 + 1$$

[0031] Ein primitives Element α in GF(2^8) ist 00000010.
[0032] Die Verschachtelungstiefe von ECC3 soll eine Spur tief sein. Die ECC-Bytes sollen folgende Gleichung erfüllen:

$$H_R \times V_R = 0$$

[0033] Das Generatorpolynom soll wie folgt sein:

$$G_R(X) = \prod_{i=0}^{i=1} \left(x - \alpha^i\right)$$

$$H_R \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 & 1 & 1 \\ \alpha^{45} & \alpha^{44} & \alpha^{43} & \ldots & \alpha^2 & \alpha & 1 \end{bmatrix}$$

$$V_R = \begin{bmatrix} D_{n,1} \\ D_{n+8734,1} \\ D_{n,2} \\ D_{n+8734,2} \\ \cdots \\ D_{n+8734,22} \\ R_{n,23} \\ R_{n+8734,23} \end{bmatrix}$$

n = 0, 1, 2, ..., 8 733
$D_{x,y}$ x = Benutzerdatenbytenummer in einer G1-Untergruppe,
y = G1-Untergruppennummer
$R_{x,23}$ x = Paritätsbytezahl in der ECC2-G2-Untergruppe

[0034] Der Fehlerkorrekturcode C3 hat die Fähigkeit, beliebige zwei Spuren zu korrigieren, die in einer aufgezeichneten Datengruppe schlecht sind.

[0035] Die Bytes jeder G1-Untergruppe werden dann durch ein G2-Modul **16** randomisiert, um eine G2-Untergruppe zu bilden, in der alle Bytes nach wie vor von D_0 bis D_{17467} numeriert sind. Die Sequenz derselben ist wie in der G1-Untergruppe. Ein G3-Modul **18** arbeitet dann auf den G2-Untergruppen, so daß jede G2-Untergruppe von 17.468 Bytes angeordnet ist, um Bytes D_0 bis D_{8733} einer G2-Untergruppe in der Spur A der E3-Untergruppe und die Bytes D_{8734} bis D_{17467} in der Spur B der G3-Untergruppe zu gruppieren (wie es in **Fig.** 3 ersichtlich ist). In jeder Spur werden die Bytes in Wörter zugeordnet. Die geradzahligen Bytes werden einem tieferen Byte zugeteilt, während ungerade Bytes einem oberen Byte zugeteilt werden. Das erste Wort jeder Spur (Wort Nr. 0) enthält zwei Bytes, die als die logische Rahmenidentifikation (LFID = Logical Frame Identification) und die Datenrahmenidentifikation (DFID = Data Frame Identification) bekannt sind.

[0036] Die DFID zeigt das DDS-Format an. Die LFID zeigt die Rahmennummer und das Vorliegen oder anderes eines 23ten Rahmens (des C3-Rahmens) an.

[0037] Jedes Byte einer G3-Untergruppe wird dann identifiziert durch:
seine Spur (A oder B);
seinen Bytenamen (oberer oder unterer);
seine Wortnummer (von 0 bis 4367)

[0038] Bei der nachfolgenden Beschreibung wird die folgende Schreibweise verwendet:
A_{il} zeigt das Byte an, das durch das untere Byte der Spur
A in dem i-ten Wort identifiziert wird;
A_{iu} zeigt das Byte an, das durch das obere Byte der Spur A in dem i-ten Wort identifiziert ist;
B_{il} zeigt das Byte an, das durch das untere Byte der Spur
B in dem i-ten Wort angezeigt ist;
B_{iu} zeigt das Byte an, das durch das obere Byte der Spur B in dem i-ten Wort identifiziert wird.

[0039] Der Spurprüfsummengenerator **20** erzeugt eine Spurprüfsumme gemäß einem Algorithmus, der keine Korrelation mit den Reed-Solomon-Codeworterzeugungsregeln aufweist. Somit verwendet der Spurprüfsummenalgorithmus eine arithmetische Summe als die Prüfsumme für jede Spur. Die Bytes in der Spur werden aufaddiert, um ein 16-Bit-Ergebnis zu bilden, wobei jeder Übertrag über 16 Bit ignoriert wird (d. h. die Summe wird modulo 2^{16} berechnet). Daher ist für die Spur A die Spurprüfsumme (TCS(A)) gegeben durch:

$$TCS(A) = [LFID] + [DFID] + \sum_{i=0}^{8733} D(i)$$

und für die Spur B:

$$TCS(B) = [LFID] + [DFID] + \sum_{i=8734}^{17467} D(i)$$

[0040] LFID und DIFD sind die Logische-Rahmen-ID und die Datenformat-ID, die Überwachungsbytes sind, wie sie oben in Verbindung mit der G3-Untergruppe beschrieben sind. Es wird angemerkt, daß alle der Primärdatenbytes in jedem C2-Codewort in einer Spur zu der gleichen Spurprüfsumme beitragen. Die Spurprüfsummenberechnung ist eine arithmetische Summe und hat daher keine Korrelation mit der XOR-Operation, die beim Berechnen der C2-Paritätsbytes verwendet wird.

[0041] Mit einer 16-Bit-Arithmetikprüfsumme ist die Wahrscheinlichkeit, daß sich die Spurprüfsumme nicht ändert, wenn C2 falsch korrigiert, im wesentlichen zufällig (d. h. $1/2^{16}$) = $1{,}5 \times 10^{-6}$.

[0042] Jede G3-Untergruppe wird dann durch ein G4-Modul **22** in G4-Untergruppen transformiert (die die C1-, C2-Codeworterzeugung umfaßt). Jede G4-Untergruppe besteht aus zwei Zwillingsarrays, wie folgt (siehe **Fig.** 4). Ein Vorzeichen, eine Fragmentnummer und eine Seriennummer sind jedem Byte zugeteilt, unter Verwendung der folgenden Formel:

VORZEICHEN: $(-1)^a$

FRAGMENTNUMMER: $i \pmod{78} + 9$

SERIENNUMMER : $\left(2\left(u + Integer\,\frac{i}{78}\right)\right) - \left(Integer\,\frac{i}{78}\right) \pmod 2$

wobei
Integer = gleich der Ganzzahlteil des Quotienten ist:
i = 0 bis 4 367
a = 0 für die A_{iu} und A_{il} Bytes
a = 1 für die B_{iu} und B_{il} Bytes
u = 0 für die A_{iu} und B_{iu} Bytes
u = 1 für die A_{il} und B_{il} Bytes

[0043] Mit Bezugnahme auf **Fig.** 4 liefert dies zwei Arrays, die mit plus und minus bezeichnet sind, was den Spuren A und B entspricht. In jedem Array ist jedes Byte durch seine Fragmentnummer (0 bis 95) und seine Seriennummer (0 bis 123) identifiziert.

[0044] Das Anwenden der obigen Formel verteilt Datenbytes in den nicht schraffierten Teil der Arrays, die in **Fig.** 4 ersichtlich sind. Die Datenbytes in dem nicht schraffierten Teil der Tabelle werden dann unter Verwendung von zwei Reed-Solomon-Fehlererfassungs- und Korrekturcodes C1 und C2 codiert, die über den Bytes berechnet werden, die bereits in den Arrays zugeteilt sind. Die C1-Codewörter erstrecken sich in der Spaltenrichtung, während sich die C2-Codewörter in der Zeilenrichtung erstrecken.

[0045] Die C2-Bytes werden für die Bytepositionen mit Seriennummern in dem Bereich von 0 bis 111 in jedem Fragment mit einer Fragmentnummer in dem Bereich von 0 bis 8 oder 87 bis 95 von den Bytes mit der gleichen Seriennummer in allen anderen Fragmenten berechnet.

[0046] Die C1-Bytes werden dann für Bytepositionen mit Seriennummern in dem Bereich von 112 bis 123 in allen Fragmenten von all den anderen Bytes in dem gleichen Fragment berechnet. In Fragmenten mit einer Fragmentnummer in dem Bereich von 0 bis 8 oder dem Bereich 87 bis 95 werden die C1-Bytes von den vorher berechneten C2-Bytes berechnet. C1 ist ein GF(2^8) Reed-Solomon-Code (62, 56, 7). Dieser Ausdruck zeigt an, daß das Codewort 62 Bytes lang ist, wovon 56 Bytes Datenbytes sind und eine Trennung von 7 Bytes aufweisen. C2 ist ein GF(2^8) Reed-Solomon (32, 26, 7).

[0047] Die Berechnung in einem GF(2^8) ist definiert durch:

$$G(x) = x^8 + x^4 + x^3 + x^2 + 1$$

[0048] Ein primitives Element α in GF(2^8) ist 00000010.

[0049] Die Verschachtelungstiefe von C1 ist 2 Bytes; somit gehören in jeder Spalte eines Arrays Datenbytes mit geraden Seriennummern zu einem der C1-Codewörter dieser Spalte, und Datenbytes mit ungeraden Seriennummern gehören zu dem anderen C1-Codewort in dieser Spalte. Die Verschachtelungstiefe von C2 ist drei Fragmente und daher gehört in jeder Zeile jedes dritte Fragment zu dem gleichen Codewort, wobei es drei C2-Codewörter in einer Zeile gibt. Die Fehlerkorrekturcode- (ECC-) Bytes erfüllen die folgenden Beziehungen:
$H_P \times V_P = 0$
$H_Q \times V_Q = 0$

[0050] Die Generatorpolynome sind:

$$G_P(X) = \prod_{i=0}^{i=5} (x - \alpha^i)$$

$$G_Q(X) = \prod_{i=0}^{i=5} (x - \alpha^i)$$

$$H_P \begin{bmatrix} 1 & 1 & 1 & \dots & 1 & 1 & 1 \\ \alpha^{61} & \alpha^{60} & \alpha^{59} & \dots & \alpha^2 & \alpha & 1 \\ \alpha^{122} & \alpha^{120} & \alpha^{118} & \dots & \alpha^4 & \alpha^2 & 1 \\ \alpha^{183} & \alpha^{180} & \alpha^{177} & \dots & \alpha^6 & \alpha^3 & 1 \\ \alpha^{244} & \alpha^{240} & \alpha^{236} & \dots & \alpha^8 & \alpha^4 & 1 \\ \alpha^{50} & \alpha^{45} & \alpha^{40} & \dots & \alpha^{10} & \alpha^5 & 1 \end{bmatrix}$$

$$H_Q \begin{bmatrix} 1 & 1 & 1 & \dots & 1 & 1 & 1 \\ \alpha^{31} & \alpha^{29} & \alpha^{28} & \dots & \alpha^2 & \alpha & 1 \\ \alpha^{62} & \alpha^{60} & \alpha^{58} & \dots & \alpha^4 & \alpha^2 & 1 \\ \alpha^{93} & \alpha^{90} & \alpha^{87} & \dots & \alpha^6 & \alpha^3 & 1 \\ \alpha^{124} & \alpha^{120} & \alpha^{116} & \dots & \alpha^8 & \alpha^4 & 1 \\ \alpha^{155} & \alpha^{150} & \alpha^{145} & \dots & \alpha^{10} & \alpha^5 & 1 \end{bmatrix}$$

(V_P und V_Q erscheinen auf der folgenden Seite).
wobei P_{ij} = C1-Bytes
Q_{ij} = C2-Bytes
i = Fragmentnummer
j = Seriennummer
[0051] Für C1: k = 0, 1, ..., 95
l = 0,1
falls k = 0, 1, ..., 8 oder k = 87, 88, ..., 95, wird D_{ij} in V_P gelesen als Q_{ij}
[0052] Für C2: m = 0, 1, 2
n = 0, 1, ..., 111
[0053] Jedes Fragment einer G4-Untergruppe wird durch einen Anfangsblockprozessor **24** in ein 132-Byte-Hauptdatenfragment transformiert, durch Voranstellen eines 8-Byte-Anfangsblocks. Der Anfangsblock enthält Steuerung und

$$V_P = \begin{bmatrix} D_{k,l} \\ D_{k,l+2} \\ D_{k,l+4} \\ D_{k,l+6} \\ D_{k,l+8} \\ D_{k,l+10} \\ D_{k,l+12} \\ D_{k,l+14} \\ D_{k,l+16} \\ D_{k,l+18} \\ D_{k,l+20} \\ D_{k,l+22} \\ D_{k,l+24} \\ D_{k,l=26} \\ D_{k,l+28} \\ D_{k,l+30} \\ \cdots \\ D_{k,l=94} \\ D_{k,l=96} \\ D_{k,l+98} \\ D_{k,l+100} \\ D_{k,l+102} \\ D_{k,l+104} \\ D_{k,l+106} \\ D_{k,l=108} \\ D_{k,l+110} \\ P_{k,l+112} \\ P_{k,l+114} \\ P_{k,l+116} \\ P_{k,l+118} \\ P_{k,l+120} \\ P_{k,l+122} \end{bmatrix} \qquad V_Q = \begin{bmatrix} Q_{m,n} \\ Q_{m+3,n} \\ Q_{m+6,n} \\ D_{m+9,n} \\ D_{m+12,n} \\ D_{m+15,n} \\ D_{m+18,n} \\ D_{m+21,n} \\ D_{m+24,n} \\ D_{m+27,n} \\ D_{m+30,n} \\ D_{m+33,n} \\ D_{m+36,n} \\ D_{m+39,n} \\ D_{m+42,n} \\ D_{m+45,n} \\ D_{m+48,n} \\ D_{m+51,n} \\ D_{m+54,n} \\ D_{m+57,n} \\ D_{m+60,n} \\ D_{m+63,n} \\ D_{m+66,n} \\ D_{m+69,n} \\ D_{m+72,n} \\ D_{m+75,n} \\ D_{m+78,n} \\ D_{m+81,n} \\ D_{m+84,n} \\ D_{m+87,n} \\ Q_{m+90,n} \\ Q_{m+93,n} \end{bmatrix}$$

Überwachungsdaten, wie es in **Fig.** 5 angezeigt ist, aber nur die Untercodebytenummer 0 bis 3 sind erwähnenswert. Die Untercodebytes enthalten Untercodeinformationen, die als sechzehn 4-Byte-Stapelelemente angeordnet sind. Jede Spur enthält 96 Fragmente, jedes mit seinem eigenen Anfangsblock, der ein Stapele-

lement umfaßt, und daher wird jedes Stapelelement sechsmal auf einer Spur wiederholt. Bei dieser Implementierung ist das Stapelelementnummer **5** den Spurprüfsummen zugewiesen, für die Spuren A und B, wie es in **Fig.** 6 gezeigt ist.

[0054] Folglich gruppiert die Vorrichtung Primärdaten anfangs in Basisgruppen, die dann in 22 Untergruppen aufgeteilt werden, wobei jede Untergruppe auf Band in einem Rahmen geschrieben wird. Jeder Rahmen besteht aus zwei Spuren (A und B), die jeweils Primärdaten plus andere Überwachungs- und Fehlerkorrekturdaten enthalten, die durch das Format addiert werden. Eine Gruppe auf Band besteht aus **22** Rahmen plus einem zusätzlichen Fehlerkorrekturcoderahmen, falls eine C3-Korrektur implementiert ist.

[0055] C3-Paritätsbytes werden über die 44 Spuren in einer Gruppe berechnet, wobei ein Byte von jeder Spur genommen wird, um ein C3-Codewort mit zwei Paritätsbytes zu erzeugen. Die C3-Paritätsbytes werden in dem oben erwähnten 23ten Rahmen gespeichert, der der Gruppe auf dem Band angehängt wird. Die Verwendung von C3 ist optional.

[0056] Es sollte angemerkt werden, daß das Generatorpolynom für die C1-, C2- und C3-Codes eine Wurzel bei α^0 aufweist, und daher wird bei einem guten Codewort eine XOR-Verknüpfung aller Bytes in dem Codewort immer ein 0-Ergebnis ergeben.

[0057] Folglich wird jede Basisgruppe in **22** G4-Untergruppen transformiert (oder **23**, falls eine C3-Korrektur vorliegt), wobei jede G4-Untergruppe aus zwei Spuren von jeweils 96 Fragmenten besteht.

[0058] Die Fragmente (mit Anfangsblöcken) werden dann einer 8-Bit-zu-10-Bit-Codierung unterworfen, bei einem 8 : 10-Codierer 26 und dann auf herkömmliche Weise auf Band geschrieben.

[0059] Beim Lesen der Daten wird die Mehrebenen-Verschachtelungs-Reed-Solomon-Fehlerkorrekturcodierung verwendet, um Fehler in den Daten zu erfassen und zu korrigieren, während dieselben gelesen werden.

[0060] Reed-Solomon-Codes ermöglichen es, daß Fehler in den Codewörtern, die vom Band gelesen werden, korrigiert werden. Während der Korrektur wird ein Reed-Solomon-Codewort mit N Paritätsbytes immer korrigiert, falls $2e + v \leq N$, wobei „e" die Anzahl von Zufallsfehlern ist und v die Anzahl von Fehlern in bekannten Positionen innerhalb des Codeworts ist (Löschungen). Falls $2e + v > N$, ist das Codewort nicht korrigierbar und kann falsch korrigiert werden. Falschkorrekturen sind während der Korrektur nicht erfaßbar.

[0061] Viele Standardtexte beschreiben Algorithmen zum Korrigieren von Reed-Solomon-Codewörtern, beispielsweise „Theory and Practice of Error Control Codes", Richard E Blahut, ISDN 0-201-10102-5, Addison-Wesley Publishing Company Inc., dessen Inhalt hierin durch Bezugnahme aufgenommen ist.

[0062] Beispiele geeigneter Algorithmen umfassen den Euklidischen Divisionsalgorithmus.

[0063] Wenn solche Algorithmen an ein Codewort angelegt werden, ist das Ergebnis entweder „korrekt", was anzeigt, daß das Codewort gut ist, und keine Korrekturen durchgeführt wurden; „korrigierbar", was anzeigt, daß das Codewort Fehler hatte, die durch den Algorithmus korrigiert wurden, „falsch korrigiert", was anzeigt, daß der richtige Algorithmus ein Codewort korrigiert hat, um ein Codewort zu erzeugen, das gültig aber falsch ist; und „nicht korrigierbar", was anzeigt, daß das Codewort schlecht ist und als solches durch den Algorithmus identifiziert wurde. Fehlkorrekturen und nicht-korrigierbare werden als „Versagen" bezeichnet.

[0064] Bei der Implementierung von DDS3 durch die Anmelder werden nicht-korrigierbare C1-Codewörter als Löschungen für C2-Korrektur markiert.

[0065] Mit Bezugnahme auf **Fig.** 4 wird daran erinnert, daß die C1-Codewörter eine Verschachtelungstiefe von 2 aufweisen und in den Spalten verlaufen. Falls daher der C2-Korrekturalgorithmus das erste Codewort z. B. in Fragment **9**, als ein nicht-korrigierbares identifiziert, bedeutet dies, daß die Bytes an mehreren Seriennummern in Fragment **9** als Löschungen in den entsprechenden C2-Codewörtern markiert werden können, in denen sie auftreten.

[0066] Fehlkorrekturen sollten ebenfalls durch die nächste Ebene erfaßt (und korrigiert) werden. Das verschachtelte Mehrebenen-Fehlerkorrekturschema liefert daher eine hohe Toleranz gegenüber den Fehlern bei den Daten, die von Band gelesen werden.

[0067] Die Spurprüfsumme wird als eine Endprüfung der Primärdaten nach einer C2- (falls C3 nicht verwendet wird) oder einer C3-Korrektur verwendet. Es ist wichtig, daß falsche Primärdaten durch die Vorrichtung erfaßt werden, die das Band liest.

[0068] In dem Fall, wo C3 nicht verwendet wird, können nichtkorrigierbare C2-Codewörter ohne weiteres erfaßt werden, wenn eine C2-Korrektur durchgeführt wird. C2 Fehlkorrekturen sind während einer C2-Korrektur jedoch nicht erfaßbar, und es ist wünschenswert, daß diese durch die Spurprüfsummen erfaßt werden. Der Algorithmus, der zum Berechnen der Spurprüfsumme beschrieben ist, hat keine Korrelation mit den Reed-Solomon-Codeworterzeugungsregeln und liefert daher ein zuverlässiges Verfahren zum Erfassen von C2-Fehlkorrekturen.

[0069] In dem Fall, wo C3 verwendet wird, ist es wünschenswert, nicht-korrigierbare C2-Codewörter für C3 zu markieren, und C2-Fehlkorrekturen als Löschungen, um es zu ermöglichen, daß die volle Leistung des C3-Codes bei der Korrektur verwendet wird. Da der Spurprüfsummenalgorithmus wiederum keine Korrelation mit den Reed-Solomon-Codeworterzeugungsregeln aufweist, kann dies erreicht werden.

[0070] Folglich werden die Daten, die von Band gelesen werden, einer 10 : 8-Decodierung durch einen De-

coder **28** unterworfen, um die Datenfragmente zu erhalten, die die Anfangsblöcke umfassen. Die Anfangsblöcke werden durch ein Reformatiermodul **30** extrahiert und die Spurprüfsummenwerte durch den Spurprüfsummenprüfer **32** wiedergewonnen. Die Datenfragmente werden dann durch einen G4-Neuformatierer **34** in ein G4-Untergruppenformat zurückgeführt, und die G4-Untergruppe wird dann C1- und C2-Korrekturalgorithmen unterworfen, zum Erfassen und, wo möglich, Korrigieren von Fehlern in den C1- und C2-Codewörtern. Dies umfaßt das Markieren von C1-Unkorrigierbaren als Löschungen für den C2-Algorithmus, wie es oben beschrieben ist, um eine gute Verwendung der Leistung zu erreichen, die von den sechs C2-Paritätsbytes verfügbar ist.

[0071] Falls es nach der C2-Korrektur keine C3-Korrekturstufe gibt, werden die Codewörter durch die Reformatierermodule **36** bis **40** zu dem G1-Untergruppenformat zurückgeführt. Die Spursummen werden neu kalkuliert und durch den Spursummenprüfer **32** mit denjenigen verglichen, die von dem Anfangsblock wiedergewonnen werden, um C2 Fehlkorrekturen zu identifizieren (C2-Unkorrigierbare wurden durch den C2-Korrekturalgorithmus markiert).

[0072] Falls es dann nach der C1- und C2-Korrektur eine C3-Korrektur gibt, markiert der Prüfsummenprüfer C2-Fehlkorrekturen als Löschungen für das C3-Korrekturalgorithmusmodul **42**, um dadurch die Möglichkeit für den C3-Korrekturalgorithmus zu liefern, eine Doppelfehlerkorrektur durchzuführen, falls zwei Löschungen markiert werden, wodurch es ermöglicht wird, daß die volle Leistung des C3-Codes bei der Korrektur verwendet wird. Danach kann die Spurprüfsumme auf den C3-korrigerten Daten als eine Enddatenprüfung verwendet werden.

Fehlerkorrekturversagen

[0073] Die folgende Analyse (nur für Zufallsfehler gültig) zeigt die Verbesserung bei den Korrekturraten an, die durch zuverlässiges Markieren von C2-Fehlkorrekturen als Löschungen erreicht wird, und dadurch eine C2-Doppelfehlerkorrektur ermöglicht.

[0074] P ist als die Wahrscheinlichkeit eines Zufallsbitfehlers in dem Kanal definiert, und Pc1 als die Wahrscheinlichkeit einer guten C1-Korrektur.

[0075] $\overline{Pc1} = 1 - Pc1$, die Wahrscheinlichkeit eines C1-Versagen (ein Versagen, das ein Unkorrigierbares oder eine Fehlkorrektur sein kann).

[0076] Pc2 und $\overline{Pc2}$ sind ähnlich definiert.

[0077] Die C2-Versagensrate kann durch die folgende Formel geschätzt werden:

$$\overline{P_{c2}} = \begin{bmatrix} 32 \\ 7 \end{bmatrix} \overline{P_{c1}}^{7} = \begin{bmatrix} 32 \\ 7 \end{bmatrix} \begin{bmatrix} 62 \\ 4 \end{bmatrix}^{7} p^{28}$$

[0078] Die Annahme hierbei ist, daß in C1-Korrektur keine Löschungen verwendet werden (somit können in einem 62-Byte-C1-Codewort nur drei Bytes korrigiert werden). Tabelle 1 zeigt Figuren für Pc2 für verschiedene Werte von p. Die C1-Fehlerrate ist ebenfalls gezeigt, d. h. die Wahrscheinlichkeit, daß ein C1-Codewort, einen oder mehrere Fehler enthält.

Tabelle 1: Wahrscheinlichkeit eines C2-Versagens gegen Fehlerrate

p	C1-Fehlerrate	$\overline{Pc2}$
10^{-2}	0,46	$10*10^{-9}$
$5*10^{-3}$	0,27	$2,1*10^{-18}$
$2*10^{-3}$	0,12	$1,5*10^{-29}$
10^{-3}	0,06	$5,7*10^{-38}$

[0079] Mit einer C3-Einzelfehlerkorrektur ist die C3-Versagenrate gegeben durch:

$$\overline{P_{c3}}(1) = \begin{bmatrix} 46 \\ 2 \end{bmatrix} \overline{P_{c2}}^{2}$$

[0080] Mit einer C3-Doppelfehlerkorrektur ist die C3-Versagenrate gegeben durch:

$$\frac{}{P_{c3}}(2) = \begin{bmatrix} 46 \\ 3 \end{bmatrix} \frac{}{P_{c2}} 3$$

[0081] Wenn man die obere Zeile von Tabelle 1 nimmt, gibt die obige Formel die folgende Zahl für C2-Versagen:

Tabelle 2: Wahrscheinlichkeit eines C3-Versagens

P	C1-Fehlerrate	Pc2	Pc3 (1)	Pc3 (2)
10^{-2}	0,46	$1*10^{-9}$	$3,3*10^{-16}$	$2,8*10^{-24}$

[0082] Eine C3-Doppelfehlerkorrektur ergibt dann Verbesserungen bei der Versagenrate von zumindest acht Größenordnungen (die anderen Zeilen von Tabelle 1 zeigen noch bessere Verbesserungen), dies ist offensichtlich ein wesentlicher Gewinn.

[0083] Um den Gesamtgewinn von dem obigen Algorithmus zu berechnen, ist es notwendig, die Wahrscheinlichkeit einer C2-Fehlkorrektur zu bestimmen, und dies nicht in der Prüfsumme aufgenommen wird. Dies ist die „Fehlerrate" in den C2-Löschungsflags, die zu C3 geleitet werden. Die obige Tabelle 1 gibt einige Zahlen für C2-Versagen in einem schlechten Fall, aber es ist notwendig, zu schätzen, welcher Prozentsatz derselben Fehlkorrekturen sind. Dies ist schwierig zu berechnen, aber wenn angenommen wird, daß die Fehlerpositionen, die in einer Fehlkorrektur erzeugt werden, in dem Bereich von 0 bis 255 zufällig sind, dann ist die Wahrscheinlichkeit, daß vier in dem Bereich von 0 bis 31 sind (32 ist die Länge eines C2-Codewort) $(32/256)^4 = 2{\cdot}4*10^{-4}$. Basierend darauf zeigt Tabelle 3 einen Vergleich für den aktuellen (herkömmlichen) Spurprüfsummenalgorithmus und den hierin beschriebenen.

Tabelle 3: C2-Flag-Fehlerrate

P	C1-Fehlerrate	Aktuelle (herkömmliche) Prüfsummen	Neue Prüfsummen
10^{-2}	0,46	$6*10^{-14}$	$2*10^{-18}$

[0084] Folglich liefert die obige Anordnung zumindest eine Verbesserung von zumindest vier Größenordnungen im Vergleich zu dem bestehenden Schema. Es erfordert keine zusätzlichen Bytes in dem Format. Es ist leicht in Hardware zu implementieren, weil ein 16-Bit-Addierer nicht viele Gatter verwendet, und es gibt keine Verschachtelung.

Patentansprüche

1. Vorrichtung zum Speichern eines Stroms von Datenaufzeichnungen auf Magnetmedien, wobei die Vorrichtung folgende Merkmale umfaßt:
eine Gruppenformatiereinrichtung zum Gruppieren der Datenaufzeichnungen in Gruppen von Datenbytes;
eine Untergruppenverarbeitungseinrichtung zum Teilen jeder der Gruppen in Untergruppen, wobei jede Untergruppe Datenbytes umfaßt, die einer oder mehreren Datenspuren entsprechen;
eine Spurprüfsummenberechnungseinrichtung zum Anlegen der Datenbytes von jeder Datenspur an einen Prüfsummenalgorithmus zum Berechnen einer oder mehrerer Prüfsummen für jede Datenspur;
eine Einrichtung zum Transformieren jeder Untergruppe in zumindest ein jeweiliges Array, wobei jedes einer Datenspur entspricht;
eine Erste-Ebene-Fehlerkorrekturcodierungs- (ECC = error correction coding) Codiereinrichtung zum Codieren von Spalten des oder jedes Arrays zum Liefern erster (C1) ECC-Codewörter, die jeweils Datenbytes und Paritätsbytes umfassen, die eine Erste-Ebene-Korrektur von zumindest einem der Datenbytes in den Erste-Ebene-ECC-Codewörtern umfaßt;
eine Zweite-Ebene-Fehlerkorrekturcodierungs- (ECC-) Codiereinrichtung zum Codieren von Zeilen des oder jedes Arrays zum Liefern zweiter (C2) ECC-Codewörter, die Datenbytes und Paritätsbytes umfassen, die eine Zweite-Ebene-Korrektur von zumindest einem der Datenbytes in den Zweite-Ebene-ECC-Codewörtern ermög-

lichen, und wobei die Datenbytes und die Paritätsbytes, die jedes gegebene zweite ECC-Codewort bilden, zumindest einer vorbestimmten ECC-Regel gehorchen;
wobei der Algorithmus, der durch die Spurprüfsummenberechnungseinrichtung angelegt wird, ausgewählt ist, um nicht mit der vorbestimmten ECC-Regel zu korrelieren, um dadurch eine Prüfsumme zu liefern, die die Eigenschaft aufweist, daß beim Decodieren der ECC-Codewörter und beim Neuberechnen und Prüfen der Spurprüfsumme die Wahrscheinlichkeit, daß ein falsch korrigiertes Codewort ein Spurprüfsummenversagen bewirkt, erhöht ist.

2. Vorrichtung gemäß Anspruch 1, bei der die Fehlerkorrekturcodiereinrichtung eine Reed-Solomon-Codierung anwendet, und die ECC-Regel erfordert, daß alle Bytes in einem bestimmten Codewort einer XOR-Verknüpfung zu Null unterzogen werden, und daß der Spurprüfsummenalgorithmus ausgewählt ist, um anders als eine XOR-Operation auf den Datenbytes der Spur oder das logische Äquivalent derselben zu sein.

3. Vorrichtung gemäß Anspruch 2, bei der der Prüfsummenalgorithmus eine arithmetische Addition der Datenbytes in der Spur ist.

4. Vorrichtung gemäß einem der vorhergehenden Ansprüche, die eine Einrichtung zum Transformieren der codierten Arrays, um für jede Spur eine Mehrzahl von Datenfragmenten zu liefern, die auf die Magnetmedien geschrieben werden, und eine Fragmentanfangsblockeinrichtung zum Versehen jedes der Datenfragmente mit einem Fragmentanfangsblock umfaßt, wobei zumindest einige der Fragmentanfangsblöcke Datenbytes umfassen, die die entsprechende Spurprüfsumme für die aktuelle Spur identifizieren.

5. Vorrichtung gemäß einem der vorhergehenden Ansprüche, die eine Dritte-Ebene-Fehlerkorrekturcodierungseinrichtung zum Berechnen von Codewörtern von jeweiligen entsprechenden Bytepositionen über jede der Spuren, die eine Gruppe bilden, umfaßt.

6. Ein Verfahren zum Speichern eines Stroms von Datenaufzeichnungen auf Magnetmedien, das folgende Schritte umfaßt:
Gruppieren der Datenaufzeichnungen in Gruppen von Datenbytes;
Dividieren jeder der Gruppen in Untergruppen von Datenbytes, wobei jede Untergruppe zumindest einer Datenspur entspricht;
Anlegen der Datenbytes von jeder Datenspur an einen Prüfsummenalgorithmus zum Berechnen einer oder mehrerer Spurprüfsummen für die oder für jede Datenspur;
Speichern der einen oder der mehreren Spurprüfsummen; Transformieren der Untergruppe in zumindest ein jeweiliges Array, wobei jedes Array einer Datenspur entspricht;
Codieren von Spalten des oder jedes Arrays, um Erste-Ebene-ECC-Codewörter zu bilden, die Datenbytes und Paritätsbytes umfassen, die eine Erste-Ebene-Korrektur von zumindest einem der Datenbytes in den Erste-Ebene-ECC-Codewörtern ermöglichen;
Codieren von Zeilen der Arrays, um Zweite-Ebene-Fehlerkorrekturcodiercodewörter zu bilden, die Datenbytes und Paritätsbytes umfassen, die eine Zweite-Ebene-Korrektur von zumindest einem der Datenbytes in dem Zweite-Ebene-ECC-Codewort erlauben, und wobei die Datenbytes und die Paritätsbytes, die jedes gegebene ECC-Codewort bilden, zumindest einer vorbestimmten ECC-Regel gehorchen;
wobei der Algorithmus, der durch die Spurprüfsummenberechnungseinrichtung angelegt wird, ausgewählt ist, um nicht mit der vorbestimmten ECC-Regel zu korrelieren, um eine Prüfsumme zu liefern, bei der beim Decodieren der ECC-Codewörter und beim Neuberechnen und Prüfen der Spurprüfsumme die Wahrscheinlichkeit, daß ein falsch korrigiertes Codewort ein Spurprüfsummenversagen bewirkt, erhöht ist.

7. Ein Verfahren gemäß Anspruch 6, bei dem zumindest die Zweite-Ebene-Fehlerkorrekturcodierung eine Reed-Solomon-Codierung anwendet, und die ECC-Regel erfordert, daß alle Bytes in einem bestimmten Codewort einer XOR-Verknüpfung zu Null unterzogen werden, und daß der Spurprüfsummenalgorithmus ausgewählt ist, um anders als eine XOR-Operation auf den Datenbytes der Spur oder das logische Äquivalent derselben zu sein.

8. Ein Verfahren gemäß Anspruch 7, bei dem der Spurprüfsummenalgorithmus eine arithmetische Addition der Datenbytes in der Spur ist.

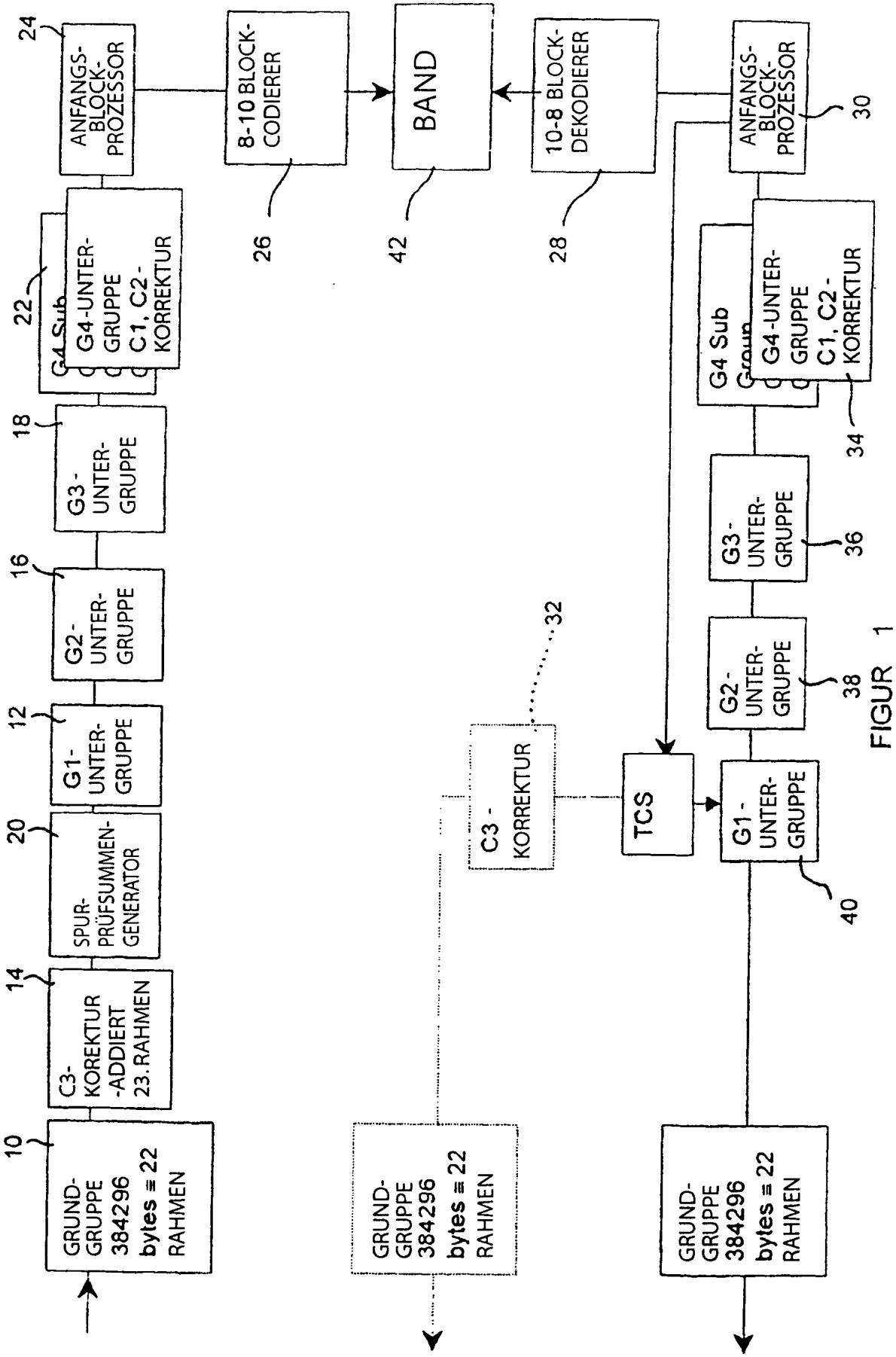
9. Ein Verfahren gemäß einem der Ansprüche 6 bis 8, das das Anlegen einer Dritte-Ebene-Fehlerkorrekturcodierung durch Berechnen von Codewörtern von entsprechenden Bytepositionen über jede der Spuren, die eine Gruppe bilden, umfaßt.

10. Ein Verfahren zum Lesen von Daten, die gemäß dem Verfahren von einem der Ansprüche 6 bis 9 gespeichert sind, das das Wiedergewinnen von Daten von einem Band, das Extrahieren der Spurprüfsummen von denselben, das Decodieren der Codewörter, um Datenbytes für jede Spur zu erhalten, das Berechnen einer Spurprüfsumme für die codierten Datenbytes und das Kennzeichnen eines Versagens umfaßt, falls die Prüfsummen nicht übereinstimmen.

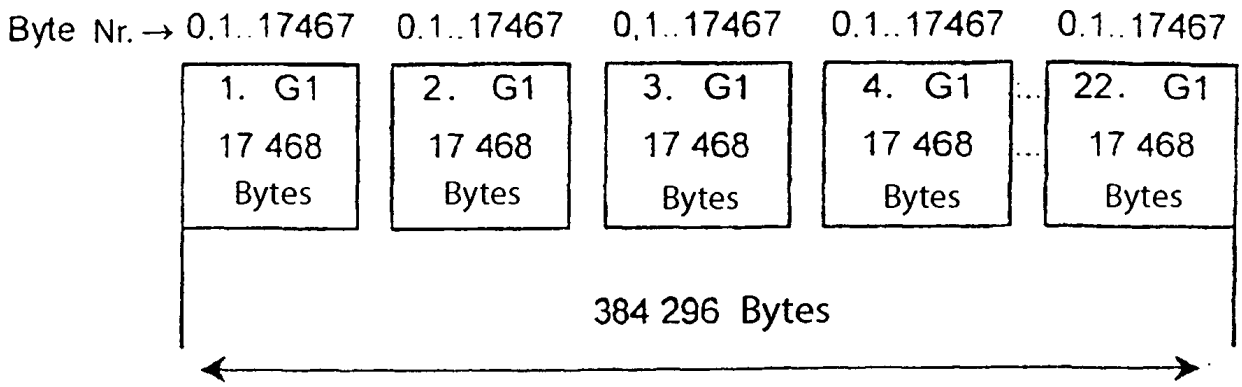
11. Ein Verfahren gemäß Anspruch 10, bei dem die Daten drei Ebenen von Fehlerkorrektur umfassen, und die Spurprüfsumme nach der zweiten Ebene der Fehlerkorrektur verwendet wird, um alle zweiten Codewortausfälle zu kennzeichnen, um dadurch alle Datenbytes in dem Dritte-Ebene-Codewort, die einem Zweite-Ebene-Codewortversagen entsprechen, zu identifizieren und für den End-Ebene-Korrekturalgorithmus als Löschungen zu markieren.

Es folgen 4 Blatt Zeichnungen

Anhängende Zeichnungen



FIGUR 1

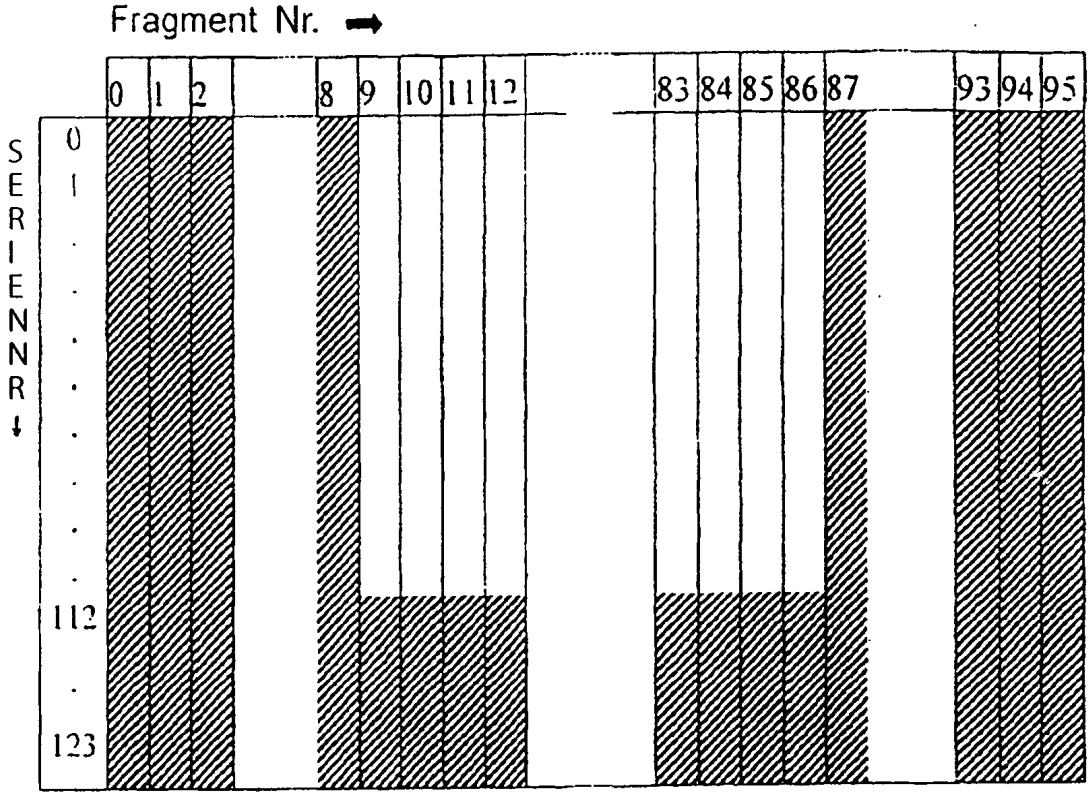


FIGUR 2

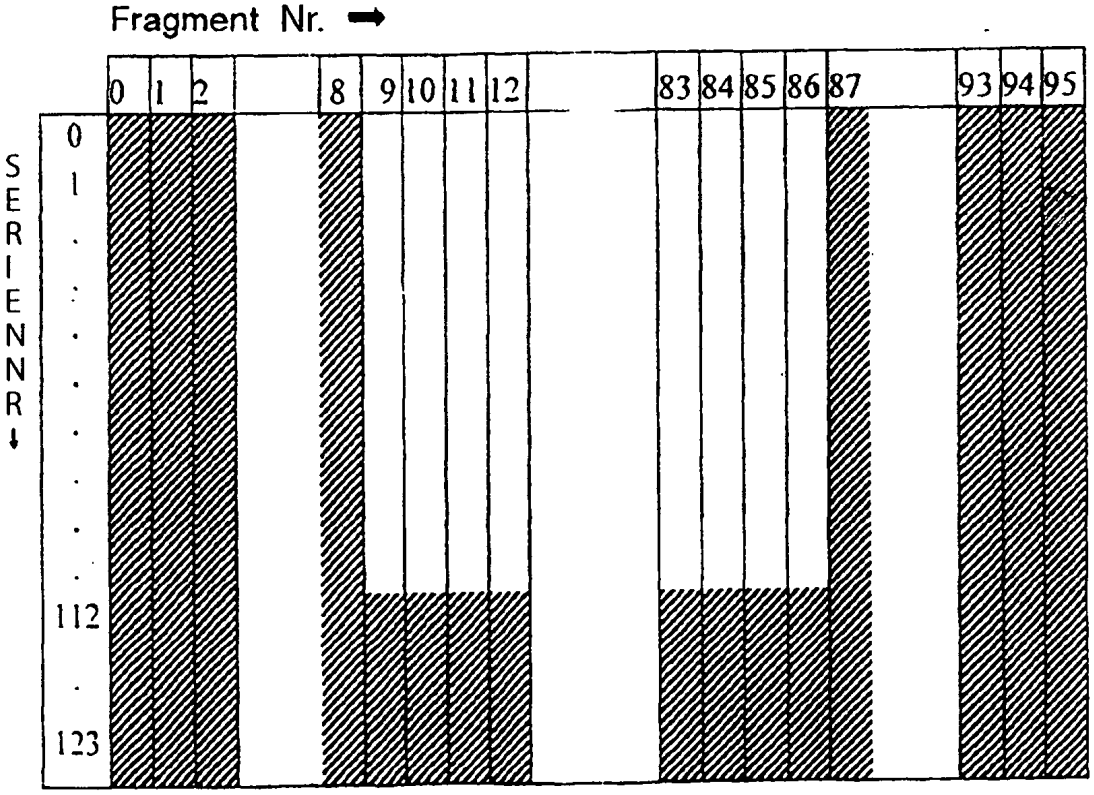
SPURNAME		A		B	
BYTENAME		UNTERE	OBERE	UNTERE	OBERE
BITZAHL		8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
ANFANGS-BLOCK	0	0 0 0 0 DF-ID	LF-ID	0 0 0 0 DF-ID	LF-ID
	1	D_0	D_1	$D_{8\,734}$	$D_{8\,735}$
	2	D_2	D_3	$D_{8\,736}$	$D_{8\,737}$
	3	D_4	D_5	$D_{8\,738}$	$D_{8\,739}$
	. . .	. . .	. . .	. . .	. . .
	4 366	$D_{8\,730}$	$D_{8\,731}$	$D_{17\,464}$	$D_{17\,465}$
	4 367	$D_{8\,732}$	$D_{8\,733}$	$D_{17\,466}$	$D_{17\,467}$
		↑ WORT-NUMMER			

FIGUR 3

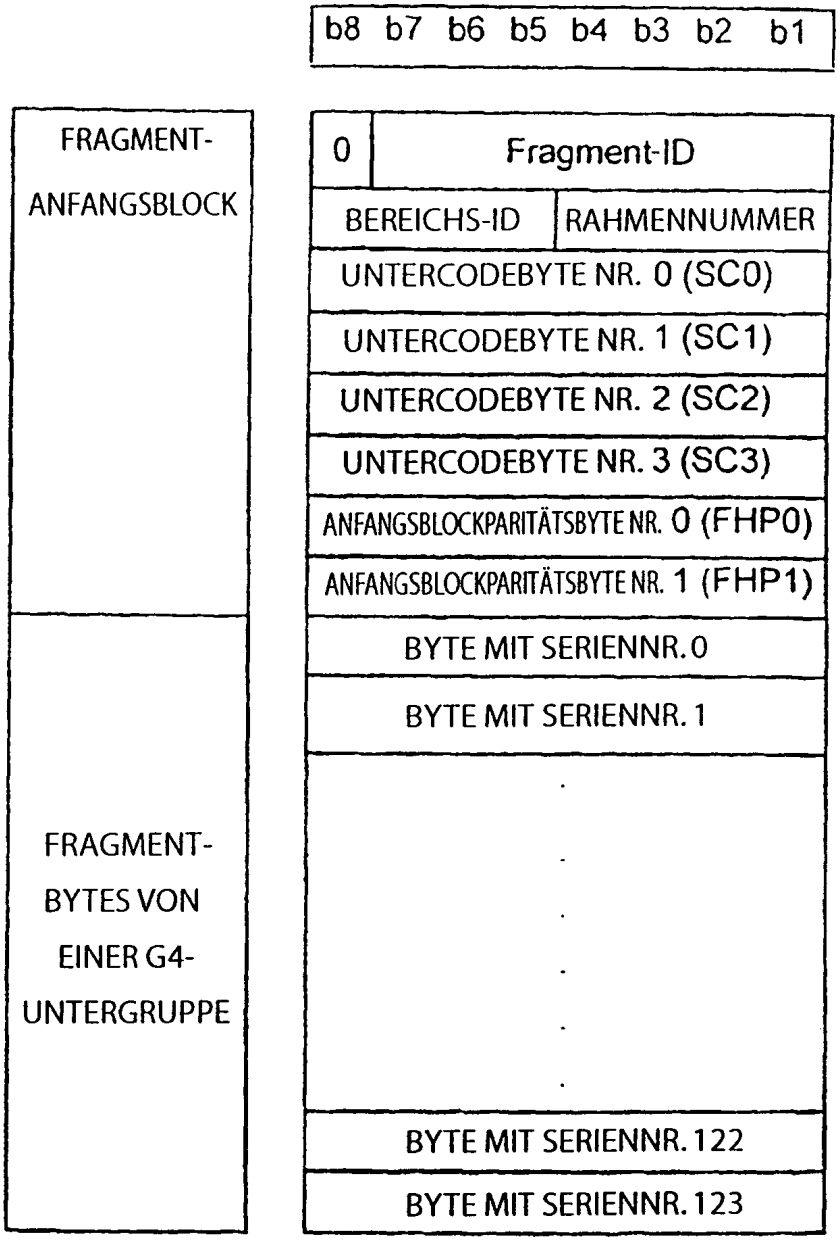
Array PLUS



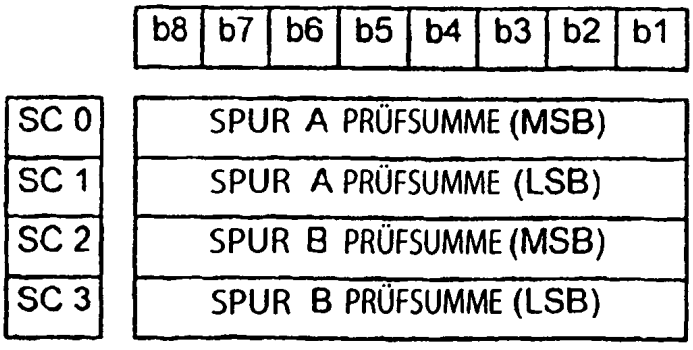
Array MINUS



FIGUR 4



FIGUR 5



FIGUR 6