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

(10) **DE 699 25 151 T2** 2006.02.16

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 127 411 B1**
(21) Deutsches Aktenzeichen: **699 25 151.6**
(86) PCT-Aktenzeichen: **PCT/US99/26102**
(96) Europäisches Aktenzeichen: **99 962 698.9**
(87) PCT-Veröffentlichungs-Nr.: **WO 00/27034**
(86) PCT-Anmeldetag: **04.11.1999**
(87) Veröffentlichungstag der PCT-Anmeldung: **11.05.2000**
(97) Erstveröffentlichung durch das EPA: **29.08.2001**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **04.05.2005**
(47) Veröffentlichungstag im Patentblatt: **16.02.2006**

(51) Int Cl.[8]: ***H03M 13/00*** (2006.01)

(30) Unionspriorität:
186753 **05.11.1998** **US**

(84) Benannte Vertragsstaaten:
DE, FR, GB

(73) Patentinhaber:
Qualcomm, Inc., San Diego, Calif., US

(72) Erfinder:
ULMER, J., Elisha, 14131 Tiberias, IL; STEIN, M., Jeremy, 34670 Haifa, IL

(74) Vertreter:
WAGNER & GEYER Partnerschaft Patent- und Rechtsanwälte, 80538 München

(54) Bezeichnung: **EFFIZIENTE NORMALISIERUNG VOM TRELLISZUSTANDSMETRISCHEM WERT**

Beschreibung

Gebiet der Erfindung

[0001] Die vorliegende Erfindung bezieht sich allgemein auf Decodierverfahren und insbesondere auf die schnelle Decodierung von Codes unter Verwendung eines Trellis bzw. eines Trellisverfahrens.

Hintergrund der Erfindung

[0002] Die Übertragung oder Sendung von Digitaldaten ist von Natur aus empfindlich gegenüber Störungen bzw. Interferenzen, die Fehler in die übertragenen Daten einführen können. Fehlerdetektionsschemata wurden vorgeschlagen, um so zuverlässig wie möglich festzustellen, ob Fehler in die übertragenen Daten eingeführt wurden.

[0003] Wenn die übertragenen Daten nicht "on-line" verwendet werden, so ist es möglich, die erneute Übertragung (re-transmission) von fehlerhafter Daten, wenn Fehler detektiert werden, anzufordern. Wenn jedoch die Übertragung "on-line" durchgeführt wird, wie beispielsweise bei Telefonleitungen, Zellentelefonen (Handys), entfernt angeordneten Videosystemen und so weiter, kann es nicht möglich oder praktikabel sein, die erneute Übertragung anzufordern.

[0004] Faltungs- bzw. Convolution-Codes und andere ähnliche Codes wurden eingeführt, um Empfängern von Digitaldaten zu gestatten, die übertragenen Daten genau oder korrekt zu bestimmen, selbst dann, wenn Fehler während der Übertragung bzw. der Sendung aufgetreten sind. Die Faltungscodes führen in die übertragenen Daten eine Redundanz ein, und zwar dadurch, dass jedes Eingangs- oder Eingabebit von Daten durch mehr als ein codiertes Bit in den gesendeten Daten repräsentiert wird. Normalerweise sind die codierten Bits der übertragenen Daten in Pakete gepackt, in denen der Wert jedes codierten Bits von früheren Bits in der Folge oder Sequenz abhängig ist. Wenn somit einige wenige Fehler auftreten, kann der Empfänger noch immer die ursprünglichen oder Originaldaten ableiten, und zwar durch Zurückverfolgung möglicher Sequenzen in den empfangenen Daten.

[0005] In einigen Decodierern weist der Empfänger jedem der Signale ein Wort zu, welches einen Wert auf einer Multi-Pegel- oder Multi-Level-Skala besitzt, repräsentativ für die Wahrscheinlichkeit, dass das codierte Bit Eins ist, und zwar geschieht dies anstelle der unmittelbaren Bestimmung ob die Empfangssignale, die in einem einzigen übertragenen codierten Bit ihren Ursprung finden, Null oder Eins sind. Eine beispielhafte Skala, auf die als LLR-Wahrscheinlichkeiten Bezug genommen wird, repräsentiert jedes gesendete oder übertragene codierte Bit durch ein 6-Bit-Wort, was eine ganze Zahl in dem Bereich {–32, 31} repräsentiert. Wenn Worte einer unterschiedlichen Anzahl von Bits verwendet werden, so wird der Bereich dementsprechend eingestellt. Die LLR-Wahrscheinlichkeit eines codierten Bits wird berechnet, und zwar dadurch, dass man den Logarithmus des Verhältnisses der Wahrscheinlichkeit bildet, dass das Bit Eins ist und zwar zur Wahrscheinlichkeit, dass das Bit Null ist, oder aber den Logarithmus des Reziproken des Verhältnisses. Der Wert von 31 kennzeichnet, dass das übertragenen Bit mit sehr hoher Wahrscheinlichkeit Null war und der Wert von –32 bedeutet, dass das übertragene Bit mit einer sehr hohen Wahrscheinlichkeit Eins war. Ein Wert von Null zeigt an, dass der Wert unbestimmt ist.

[0006] Wenn ein Sender ein codiertes Paket von Bits überträgt, in dem jedes codierte Bit einen definierten Wert von "1" oder "0" besitzt, und zwar über einen "rauschenden" Kanal, so empfängt der Empfänger ein Paket in dem jedes Bit einen variablen Spannungswert besitzt, der als eine LLR-Wahrscheinlichkeit interpretiert werden kann, und zwar in Folge der Interferenz, die durch den Kanal eingeführt wird. Ein Decodierer muss basierend auf dem empfangenen Paket das übertragene oder gesendete Paket bestimmen. Ein einfaches Verfahren umfasst die Bestimmung einer "Differenz" zwischen dem empfangenen Paket und allen möglichen Paketen und die Bestimmung welches mögliche Paket die kleinste Differenz besitzt. Wegen der großen Anzahl unterschiedlicher möglicher Werte der Pakete ist dieses Verfahren jedoch normalerweise nicht praktikabel.

[0007] Bei Faltungscodierverfahren und anderen verwandten Verfahren wird ein nicht codiertes Eingangspaket in einen Codierer eingespeist, der eine Anzahl von möglichen Zuständen besitzt. Wenn jedes Datenbit des nicht codierten Paketes in den Decodierer eingespeist wird, bewirkt es eine Änderung des Zustands des Codierers und liefert eine Gruppe von einem oder mehreren codierten Ausgangsbits (output coded bits), die eine Funktion des Zustandes und des Eingangs sind. Die Gruppen von codierten Ausgangsbits bilden ein codiertes Paket, welches übertragen wird. Die Anzahl der Bits in jeder Gruppe ist ein Faktor der Redundanz, eingeführt durch den Code und zwar als Folge der Faltung. Ein Code in dem jede Gruppe beispielsweise zwei Bits umfasst, besitzt eine Coderate von 1/2, was bedeutet, dass der tatsächliche Informationsgehalt eines Pakets gleich der Hälfte der Anzahl der codierten Bits im Paket ist.

[0008] Faltungscodes werden im Allgemeinen gemäß Decodierschemata decodiert, die ein Trellis verwenden, wie beispielsweise MAP-Decodieren (oder APP-Decodieren), SOVA-Decodieren und Viterbi-Decodieren. Ein Decodierer empfängt die Wahrscheinlichkeiten der empfangenen codierten Bits in

dem codierten Paket (zusammen mit während der Übertragung eingeführtem Rauschen) und decodiert das codierte Paket durch Zurückverfolgung der Schritte des Codierers. Der Decodierer berechnet für jeden möglichen Zustand des Codierers einen Unterschied oder eine Differenz zwischen den das empfangene Paket repräsentierenden Worten und einem bevorzugten übertragenen Paket, welches den Codierer in diesen Zustand gebracht haben würde. Diese Differenz wird als eine Zustandsmetrik ("state metric") bezeichnet. Für jede Gruppe von Worten, die eine Gruppe von empfangenen Bits repräsentiert, aktualisiert der Decodierer die Zustandsmetrik jedes möglichen Zustands, und zwar entsprechend der Differenz zwischen den Wahrscheinlichkeitswerten der empfangenen Bits und den idealen hypothetischen Werten, die für einen bestimmten oder spezifischen Zustandsübergang (state transition) (als ein "Trellisübergang" bzw. Trellistransition bezeichnet) des Codierers erforderlich wären. Beim Viterbi-Decodieren, wenn die Übergänge (transitions) von unterschiedlichen Zuständen zu dem gleichen resultierenden Zustand führen, herrscht der Übergang der sich aus einer Zustandsmetrik mit der höchsten Wahrscheinlichkeit ergibt vor. Beim MAP- und APP-Decodieren ist der Wert der neuen Zustandsmetrik eine Funktion aller Übergänge (transitions), die in den Zustand führen, und zwar beispielsweise eine Summe davon.

[0009] Die Zustandsmetrik steigt schnell mit jedem Bit das verarbeitet ist an, kann bei Paketen von Tausenden von Bits Repräsentation mit 15–20 Bits oder selbst mehr erfordern. Wenn das Decodieren in Software ausgeführt wird, beispielsweise in einem Digitalsignalprozessor, bewirken diese Größen an sich kein großes Problem. Da jedoch die Decodierung unter kritischen zeitlichen Einschränkungen ausgeführt werden muss, werden dieser Aufgabe gewidmete Hardwareprozessoren vorzugsweise eingesetzt und verwendet. Bei derartigen Prozessoren ist es notwendig, die Anzahl von Bits zu reduzieren, die verwendet wird, um die Zustandsmetrik zu repräsentieren, und zwar um schnelles Decodieren ohne exzessive Hardwarekosten zu erreichen.

[0010] Eine übliche Lösung umfasst Folgendes: Verwendung von 8-Bit-Registern zur Speicherung der Zustandsmetriken. Um die Sättigung zu verhindern wird eine Normalisierungsmetrik (NM) periodisch berechnet, die die Minimumzustandsmetrik umfasst, und zwar geschieht dies vorzugsweise nach jedem aufeinanderfolgenden Trellisübergang (trellis transition) und die NM wird von sämtlichen Registern abgezogen. Die Berechnung der minimalen Zustandsmetrik ist jedoch zeitraubend. Im Allgemeinen wird die NM parallel mit dem Betrieb des Decodierers berechnet und die NM wird später an einer verzögerten Stufe subtrahiert, wenn das Minimum bestimmt wird. In der Zwischenzeit können die Register sich sättigen und wertvolle Daten verlieren. Zudem erfordert eine solche Lösung zusätzliche Hardware zur Aufbewahrung der NM, die an einer späteren Stufe verwendet wird und zum Subtrahieren von dem NM an der späteren Stufe, wobei vorherige Werte von der NM bereits von den Zustandsmetrikregistern während der Verzögerung subtrahiert wurden.

[0011] A. P. Hekstra schlägt in "An Alternative to Metric Rescaling in Viterbi Decoders", IEEE Trans. Commun., Band 37, Nr. 11 (Nov. 1989), Seiten 1220–1222, die Verwendung eines Modularberechnungsverfahrens vor, um die Sättigung der Zustandsmetriken zu verhindern. Um die Sättigung zu verhindern, werden einige wenige zusätzliche Bits dazu verwendet, um die Zustandsmetrik zu repräsentieren, und zwar zusammen mit dem modularen Berechnungsergebnis. Beispielsweise werden für einen Vierzustand, 1/2-Ratencode mit 6-Bit-Datenworten, 11-Bit-Register zur Speicherung der Zustandsmetriken verwendet. Jedes zusätzliche Bit in den Registern benötigt jedoch mehr Rechenzeit und erhöht die Kosten des Decodierers.

[0012] C. B. Shung, P. H. Siegel, G. Ungerbroeck und H. K. Thapar in "VLSI Architectures for Metric Normalisation in the Viterbi Altorithm", IEEE International Conference on Communications ICC" 90, 15.–19. April 1990, Seiten 1723–1728, beschreiben das subtrahieren einer minimalen Survivor- oder Überlebensmetrik von allen Survivor- oder Überlebensmetriken nachdem eine feste Anzahl von Rekursionen vorgenommen wurde. Ein Subtrahierer ist erforderlich, um die minimale Survivormetrik abzuziehen.

Zusammenfassung der Erfindung

[0013] Es ist ein Ziel einiger Aspekte der vorliegenden Erfindung Verfahren und Vorrichtungen vorzusehen zur schnellen Zustandsnormalisierung in Decodern unter Verwendung eines Trellis.

[0014] Somit ist gemäß einem ersten Aspekt der vorliegenden Erfindung ein Verfahren vorgesehen zur Normalisierung einer Vielzahl von Zustandsmetrikregistern in einem Decodierer unter Verwendung eines Trellis, wie in Anspruch 1 angegeben.

[0015] Gemäß einem zweiten Aspekt wird eine Zustandsmetrikberechnungseinheit zur Verwendung in einem Decodierer vorgesehen, der unter Verwendung eines Trellis, wie in Anspruch 9 angegeben, decodiert.

[0016] Gemäß einem dritten Aspekt wird ein iterativer Decodierprozessor vorgesehen, der eine Vielzahl von Berechnungseinheiten aufweist, und zwar gemäß dem zweiten Aspekt nach Anspruch 16.

[0017] In bevorzugten Ausführungsbeispielen der

vorliegenden Erfindung berechnet ein ein Trellis verwendender Decodierer, wie beispielsweise ein "A Posteriori-Probability (APP) Decodierer = (oder ein Maximal-A-Posteriori- (MAP-) – Decodierer), ein Viterbi-Decodierer oder ein Soft Output-Viterbi-Algorithmus (SOVA-)-Decodierer, eine Minimumzustandsmetrik, und zwar verwendet bei der Normalisierung nur annähernd. Der Schaden wegen der angenäherten Berechnung ist vernachlässigbar, und zwar relativ zu den Zeiteinsparungen, die sich durch die Annäherung ergeben. Vorzugsweise ist das angenäherte Minimum kleiner als oder gleich dem tatsächlichen Minimum derart, dass dann, wenn das berechnete Minimum von allen Zustandsmetrikregistern abgezogen wird, und zwar unter Verwendung vorzeichenloser Arithmetik, keine Daten verloren gehen.

[0018] Vorzugsweise wird das angenäherte Minimum berechnet durch Bestimmung des Minimums einiger weniger am meisten signifikanten Bits der Zustandsmetriken. Vorzugsweise ist die Zahl der Bits verwendet zur Berechnung des angenäherten Minimums zwischen 30% und 60% der Bits in den Zustandsmetriken. In einem bevorzugten Ausführungsbeispiel der vorliegenden Erfindung sind acht Bits in den Zustandsmetrikregistern und die vier am meisten signifikanten Bits (MSBs = most significant bits) werden bei der Berechnung des Minimums zur Normalisierung verwendet.

[0019] Gemäß einem bevorzugten Ausführungsbeispiel der vorliegenden Erfindung wird ein Verfahren zur Normalisierung einer Vielzahl von Zustandsmetrikregistern in einem Decoder vorgesehen, und zwar unter Verwendung eines Trellis einschließlich der Bestimmung eines angenäherten Minimums von entsprechenden Zustandsmetrikwerten, gespeichert in der Vielzahl von Zustandsmetrikregistern, und durch Subtrahieren des angenäherten Minimums von diesen Werten.

[0020] Vorzugsweise ist das angenäherte Minimum stets gleich oder weniger als ein tatsächliches Minimum der Werte in der Vielzahl von Zustandsmetrikregistern.

[0021] Die Bestimmung des angenäherten Minimums umfasst Folgendes: Bestimmung des Minimums einer vorbestimmten Anzahl der am meisten signifikanten Bits in der Vielzahl von Zustandsmetrikregistern.

[0022] Vorzugsweise schließt die vorbestimmte Anzahl der am meisten signifikanten Bits zwischen 30% und 60% der Anzahl der Bits im Register ein.

[0023] Vorzugsweise umfasst das Verfahren das Berechnen von neuen Zustandsmetrikwerten für die Register, wobei das Subtrahieren des angenäherten Minimums im Wesentlichen gleichzeitig mit der Berechnung der neuen Werte ausgeführt wird.

[0024] Vorzugsweise umfasst das Verfahren das Speichern in den Zustandsmetrikregistern während einer Vielzahl von aufeinander folgenden Clock- oder Taktzyklen einer Vielzahl von entsprechenden neuen Werten, wobei die Bestimmung des angenäherten Minimums die Bestimmung eines angenäherten Minimums der neuen Werte in im Wesentlichen jedem Zyklus umfasst.

[0025] Vorzugsweise umfasst die Bestimmung des angenäherten Minimums Folgendes: Bestimmung eines angenäherten Minimums während eines ersten einer Vielzahl von Clock- oder Taktzyklen, und wobei das Subtrahieren des angenäherten Minimums das Subtrahieren während eines zweiten der Vielzahl von Taktzyklen nach dem ersten Taktzyklus umfasst.

[0026] Vorzugsweise umfasst das Verfahren das Einstellen des angenäherten Minimums ansprechend auf einen Zeitspalt oder eine Zeitunterbrechung zwischen den ersten und zweiten Taktzyklen.

[0027] Vorzugsweise umfasst das Einstellen des angenäherten Minimums Folgendes: Subtrahieren von dem angenäherten Minimum berechnet während des zweiten Zyklus, das angenäherte Minimum berechnet während des ersten Zyklus.

[0028] Ferner ist gemäß einem bevorzugten Ausführungsbeispiel der vorliegenden Erfindung eine Zustandsmetrikberechnungseinheit vorgesehen, und zwar zur Verwendung in einem Decodierer, der unter Verwendung eines Trellis decodiert und zwar Folgendes einschließend: eine Vielzahl von Zustandsmetrikregistern, die entsprechende Zustandsmetrikwerte speichern, eine Minimumberechnungseinheit, die ein angenähertes Minimum der Vielzahl von Registern bestimmt, und eine Vielzahl von Subtrahierern, die das angenäherte Minimum von den Werten in der Vielzahl der Register subtrahiert.

[0029] Die Minimumberechnungseinheit berechnet das Minimum einer vorbestimmten Anzahl der am meisten signifikanten Bits in den Registern und der Subtrahierer subtrahiert das angenäherte Minimum von der vorbestimmten Anzahl der am meisten signifikanten Bits in den Registern.

[0030] Vorzugsweise weist die Einheit eine Vielzahl von rekursiven Kombinierern auf, die für jedes Register eine nächste Zustandsmetrik des Registers berechnen, und wobei jeder rekursive Kombinierer einen entsprechenden Subtrahierer der Subtrahierer umfasst.

[0031] Vorzugsweise ist der Decodierer ein Viterbi-Decodierer, ein APP-Decodierer, ein MAP-Decodierer, ein Trellis-Decodierer und/oder ein Soft-Out-

put-Viterbi-Decodierer.

[0032] Gemäß einem bevorzugten Ausführungsbeispiel der Erfindung ist ein iterativer Decodierprozessor für die iterative Decodierung einer Sequenz von Signalpaketen codiert gemäß einem Multikomponentencodierschema vorgesehen, und zwar einschließlich einer Vielzahl von Berechnungseinheiten, wie oben beschrieben.

[0033] Die vorliegende Erfindung wird vollständiger anhand der folgenden detaillierten Beschreibung von bevorzugten Ausführungsbeispielen der Erfindung verstanden, und zwar im Zusammenhang mit den Zeichnungen:

Kurze Beschreibung der Zeichnungen

[0034] Fig. 1 ist ein schematisches Blockdiagramm eines Codierers **10**, der codierte Pakete erzeugt;

[0035] Fig. 2 ist ein schematisches Blockdiagramm eines APP-Decodierers gemäß einem bevorzugten Ausführungsbeispiel der Erfindung;

[0036] Fig. 3 ist ein schematisches Blockdiagramm einer Vorwärtszustandsmetrikberechnungseinheit in dem Decodierer der Fig. 2 gemäß einem bevorzugten Ausführungsbeispiel der Erfindung;

[0037] Fig. 4 ist eine graphische Darstellung möglicher Zustandsübergänge (transitions) im Codierer der Fig. 1;

[0038] Fig. 5 ist ein Blockdiagramm eines rekursiven Metrikkombinierers in der Zustandsberechnungseinheit der Fig. 3, und zwar gemäß einem bevorzugten Ausführungsbeispiel der Erfindung;

[0039] Fig. 6 ist ein Blockdiagramm einer Minimum-Berechnungseinheit in der Zustandsmetrikberechnungseinheit der Fig. 3, und zwar gemäß einem bevorzugten Ausführungsbeispiel der Erfindung;

[0040] Fig. 7 ist ein Blockdiagramm eines Decodierprozessors, der zwei Decodierer ähnlich dem Decodierer der Fig. 2 aufweist, und zwar gemäß einem bevorzugten Ausführungsbeispiel der Erfindung; und

[0041] Fig. 8 ist ein Blockdiagramm einer Minimum-Berechnungseinheit gemäß einem weiteren bevorzugten Ausführungsbeispiel der Erfindung.

Detaillierte Beschreibung bevorzugter Ausführungsbeispiele

[0042] Fig. 1 ist ein schematisches Blockdiagramm eines Codierers **10**, der codierte Pakete erzeugt. Der durch den Codierer **10** erzeugte Code besitzt eine Generator- oder Erzeugermatrix (G[D]), gegeben durch die Gleichung (1):

$$G[D] = [1,(1 + D + D^2)/(1 + D)] \quad (1)$$

wobei D ein Verzögerungselement **12** repräsentiert.

[0043] Fig. 2 ist ein schematisches Blockdiagramm eines APP-Decodierers **20** gemäß einem bevorzugten Ausführungsbeispiel der Erfindung. Der Decodierer **20** verwendet eine Trellis zur Decodierung des durch den Codierer **10** erzeugten Codes.

[0044] Der Decodierer **20** weist eine einzige Eingangsleitung auf, durch die ein Eingangs- oder Eingabepaket an den Decodierer geliefert wird. Ein wahlweise oder optionaler Depunktierer (de-puncturer) **24** addiert LLR-Wahrscheinlichkeitsworte mit Null oder Nullwerten (null or zero values) wie erfordert, wenn der Code decodiert durch den Decodierer **20** Punktierung ("Puncturing") verwendete, wie dies im Stand der Technik bekannt ist. Die "Null- oder Zerowerte" zeigen an, dass die Codebits repräsentiert durch die LLR-Worte die gleiche Wahrscheinlichkeit besitzen "1" oder "0" zu sein. Für jede Gruppe von LLR-Worten in dem Eingangs- oder Eingabepaket berechnet eine Zweig- oder Branchmetrik-(BM)-Recheneinheit **26**, wie im Stand der Technik bekannt, eine BM (branch metrik, Zweigmetrik) für jeden möglichen hypothetischen Wert der Codebits repräsentiert durch die LLR-Worte in der Gruppe. Für ½-Raten-Codes (d.h. für Codes mit Rate ½), wie beispielsweise den durch den Codierer **10** erzeugten Code, entspricht jede Gruppe zwei codierten Bits und hat daher vier mögliche hypothetische Werte.

[0045] Vorzugsweise gibt, wie unten weiterhin beschrieben, die Einheit **26** vier 8-Bit-BMs für jede neue Gruppe von LLR-Worten in den Eingang. Jede BM zeigt die Wahrscheinlichkeit an, dass die Gruppe von Bits ursprünglich den entsprechenden hypothetischen Wert hatte. Die BM-Einheit **26** kann beispielsweise eine solche sein, wie sie in der folgenden Literaturstelle beschrieben ist: "Implementation and Performance of a Turbo/MAP-Decoder" von Steven S. Pietrobon im International Journal of Satellite Communications, Band 16 (1998), Seiten 23–46.

[0046] Eine oder mehrere, vorzugsweise drei, ZustandsmetrikBerechnungseinheiten (SM = state-metric calculation units)-**30**, **32** und **34** empfangen die BMs und berechnen demgemäß die Zustandsmetriken jedes der Zustände des Codes wie weiter unten beschrieben. Vorzugsweise empfangen die SM-Berechnungseinheiten **30**, **32** und **34** die BMs von einer oder mehreren Speichereinheiten **28**, wo die Branch- oder Zweigmetriken (BM) gespeichert sind. Ferner besitzt jede SM-Berechnungseinheit ihre eigene BM-Speichereinheit **28**, so dass die SM-Berechnungseinheiten parallel arbeiten können, ohne miteinander in Interferenz zu geraten oder sich zu

stören. Vorzugsweise unterscheiden sich die SM-Einheiten **30**, **32** und **34** in der Richtung, in der das Paket verarbeitet wird. Die Einheit **30** verarbeitet das Paket in einer ersten (Vorwärts-)Richtung und zwar vom Beginn oder Anfang bis zum Ende, wohingegen die Einheiten **32** und **34** das Paket in der entgegengesetzten (umgekehrten; reverse) Richtung vom Ende zum Anfang verarbeiten.

[0047] Zwei Reverse- oder Umkehreinheiten **32** und **34** sind vorzugsweise in Verwendung, um die Verwendung einer Näherungs- oder Approximationsmethode zu verwenden, die auf der Umkehr von Segmenten des Eingangspaketes basieren anstelle des Umkehrens des gesamten Pakets. Diese Annäherung gestattet die Durchführung der umgekehrten (reverse) Decodierung direkt nach dem jedes Datensegment in dem Paket empfangen wurde, anstelle dass man auf ein ganzes oder gesamtes Paket wartet, das umgekehrt werden muss. Das Verfahren ist in der folgenden Literaturstelle beschrieben: "An Intuitive Justification and Simplified Implementation of the MAP Decoder for Convolutional Codes" von Andrew J. Viterbi, IEEE Journal on Selected Areas in Communications, Band 16, Nr. 2, Seiten 260–264 (Februar 1998). Jede der Einheiten **32** und **34** liefert zuverlässige SMs während der Hälfte der Betriebszeit, so dass sie zusammen zuverlässig SMs für das gesamte Paket liefern. Ein Multiplexer **35** wählt die richtigen Ergebnisse gemäß der Approximationsmethode aus. Eine Speichereinheit **36** wird dazu verwendet, um die Reihenfolge des verarbeiteten Pakets oder der Segmente umzukehren, und zwar zurück zur richtigen Vorwärtsreihenfolge oder Ordnung, so dass die Verwendung zusammen mit den Resultaten von Einheit **30** erfolgen kann. Eine LLR-Berechnungseinheit **40** berechnet einen Ausgangswahrscheinlichkeitswert für jedes LLR-Wort, und zwar basierend auf den SMs von den Einheiten **30**, **32** und **34** und auf den BMs von der Einheit **26**. Die LLR-Einheit **40** ist vorzugsweise von einer Art, wie sie in dem oben genannten Artikel von Steven S. Pietrobon, oder alternativ, wie es in dem oben genannten Artikel von Anthony J. Viterbi beschrieben ist. Das Resultat oder das Ergebnis von Einheit **40** kann auf einer Ausgangs- oder Ausgabeleitung **42** ausgegeben werden und/oder kann durch einen Punktierer (puncterer) **44** für die weitere Verarbeitung weiter geleitet werden.

[0048] Fig. 3 ist ein schematisches Blockdiagramm einer Vorwärts-SM-Berechnungseinheit **30** gemäß einem bevorzugten Ausführungsbeispiel der Erfindung. Die Umkehr-(Reverse-)-SM-Einheiten **32** und **34** sind ähnlich im Aufbau zur Einheit **30** und zwar mit den notwendigen Modifikationen, wie sie im Stand der Technik bekannt sind. Die Einheit **30** weist vier rekursive Metrikkombinierer **50** auf, und zwar entsprechend jedem der vier möglichen Zustände des Codierers. Jeder der Kombinierer **50** berechnet rekursiv das SM des Zustands. Für jede Bitgruppe (n + 1) im Paket aktualisieren die Kombinierer **50** ihre SMs basierend auf den SMs der vorhergehenden Bitgruppe (n) und auf den entsprechenden BMs, wie weiter unten beschrieben. Die berechneten SMs werden zurück auf Rückkopplungsleitungen **52** geleitet, und zwar für die nächste Berechnungsiteration der nächsten Bitgruppe des Pakets. Vorzugsweise liefern eine Vielzahl von Ausgangsleitungen **54** die SMs an die LLR-Berechnungseinheit **40**. Zudem bestimmt eine Minimumberechnungseinheit **56** nach jeder oder nach mehreren rekursiven Iterationen der Kombinierer **50**, das Minimum der am meisten signifikanten Bits der SMs. Das berechnete Minimum wird als eine Normalisierungsmetrik (NM) verwendet und wird zurück zu den Kombinierern **50** zur Normalisierung geleitet.

[0049] Es sei bemerkt, dass in den bevorzugten Ausführungsbeispielen der Erfindung die Einheit **30** mehr als vier Kombinierer **50** umfasst, vorzugsweise 16 Kombinierer entsprechend den 16 Zuständen. Aus Gründen der Klarheit veranschaulichen die Fig. 2–Fig. 4 ein einfaches Vier-Zustands-Ausführungsbeispiel. Die Erweiterung der hier beschriebenen Konzepte zum Betrieb von 16 oder sogar einer größeren Anzahl von Zuständen ist dem Fachmann unmittelbar einleuchtend.

[0050] Fig. 4 ist eine graphische Darstellung, die mögliche Zustandsübergänge in dem Code decodiert durch den Decodierer **20** zeigt, wobei der Code entsprechend der obigen Gleichung (1) erzeugt wurde. Die Knoten **60** repräsentieren die möglichen Zustände des Codierers für eine Bitgruppe (n) von zwei Bits. Die Knoten **62** repräsentieren die möglichen Zustände des Codierers für eine darauf folgende Bitgruppe (n + 1). Gemäß dem Code kann der Codierer einen Übergang machen, und zwar von einem der Knoten **60** der Bitgruppe (n) auf bestimmte der Knoten **62** der Bitgruppe (n + 1) und zwar nur entlang eines einer Vielzahl von Zweigen (branches) **64**. Um einen solchen Übergang zu machen, muss die Bitgruppe (n) den Wert haben, der oben durch den entsprechenden Zweig **64** angegeben ist.

[0051] Daher gilt, zurückgreifend auf Fig. 3 Folgendes: jeder Kombinierer **50** empfängt zwei SMs, die SMs von denjenigen Knoten **60** sind, die zu dem Zustand führen können, der mit dem speziellen Kombinierer assoziiert ist, und zwar repräsentiert durch Knoten **62** des Kombinierers **50**. Zudem empfängt jeder Kombinierer **50** zwei BMs, die den BMs der Zweige **64** entsprechen, die von den zwei Knoten **60** zum Knoten **62** entsprechend dem speziellen Kombinierer führen. Beispielsweise entspricht der obere Kombinierer **50** in Fig. 2 dem Zustand '00'. Er empfängt daher die SM des Zustands '00' und die BM '00', welches die Bitgruppe des Zweiges (branch) **64** ist, der vom Knoten **60** des Zustandes '00' zum Knoten **62** des Zustands '00' führt. Zudem empfängt der obere

Kombinierer **50** die SM des Zustands '01' und die BM '01', welche die Bitgruppe des Zweiges **64** ist, der vom Knoten **60** des Zustandes '01' zum Knoten **62** des Zustandes '00' führt.

[0052] Fig. 5 ist ein schematisches Blockdiagramm des rekursiven Kombinierers **50** gemäß einem bevorzugten Ausführungsbeispiel der Erfindung. Zwei Addierer **70** berechnen jeweils zwei mögliche nächste-Stufe-(n + 1)-Zustandsmetriken des Zustands repräsentiert durch den speziellen Kombinierer **50** und zwar durch Addieren der SM der richtigen oder ordnungsgemäßen Knoten **60** zu entsprechenden BMs der entsprechenden Zweige **64**. Ein Subtrahierer **72** subtrahiert die zwei möglichen nächste-Stufe-Zustandsmetriken voneinander und basierend auf dem Vorzeichen der Subtraktion wird die niedrigere Zustandsmetrik durch einen MUX **74** gewählt. Die Zustandsmetrik vom MUX **74** ist vorzugsweise 9 Bits breit. Die Normalisierungsmetrik (NM), berechnet in der Einheit **56**, wird von den fünf am meisten signifikanten Bits der gewählten SM in einem Subtrahierer **78** abgezogen. Die vier am wenigsten signifikanten Bits der Zustandsmetrik werden vorzugsweise intakt auf einer Leitung **80** geleitet, und zwar den Subtrahierer **78** umgehend.

[0053] Vorzugsweise bestimmt eine Nachschautabelle **82** eine Versetzung (offset) basierend auf der Differenz zwischen den zwei möglichen nächste-Stufe-Zustandsmetriken (a, b) und zwar berechnet durch Subtrahierer **72**. Vorzugsweise ist die Ausgangsgröße der Nachschautabelle **82** so wie in Gleichung (2) beschrieben:

$$LUT = Const \cdot (\ln^2 - \ln(1 + e^{-|a-b|})) \qquad (2)$$

in der Folgendes gilt: Const ist eine Skaliervariable abhängig von der Anzahl der Bits, die die SMs repräsentieren oder genauer gesagt abhängig von dem Quantisierungsschritt der SMs. Die Gleichung (2) ist auf dem Gebiet der APP-Decodierer bekannt. Es sei bemerkt, dass Viterbi-Decodierer auch einen Kombinierer ähnlich dem Kombinierer **50** enthalten, aber ohne LUT **82**. Die Versetzung (offset) wird der normalisierten Zustandsmetrik am Addierer **84** hinzu addiert, wobei eine korrigierte Zustandsmetrik geliefert wird. Vorzugsweise ist die korrigierte Zustandsmetrik auf 8 Bits gesättigt, und zwar an einer Klammer (clamp) **86** und wird sodann in einem 8-Bit-Register **88** zur Ausgabe vom rekursiven Kombinierer **50** gespeichert.

[0054] Fig. 6 ist ein Blockdiagramm der Minimum-Recheneinheit **56** gemäß einem bevorzugten Ausführungsbeispiel der Erfindung. Die Einheit **56** empfängt vorzugsweise die vier am meisten signifikanten Bits (MSB) der SMs von jedem der rekursiven Kombinierer **50**. Auswahleinheiten **90** wählen die vier MSBs aus, die den niedrigsten Wert besitzen, wobei dieser als die Normalisierungsmetrik (NM) ausgegeben wird.

[0055] Fig. 7 ist ein Blockdiagramm eines Decodierprozessors **100** der zwei Decodierer ähnlich dem Decodierer **20** gemäß einem bevorzugten Ausführungsbeispiel der Erfindung verwendet. Der Prozessor **100** ist beispielsweise zweckmäßig bei einer Turbodecodierung und bei einer turboartigen Decodierung, wie dies in der anhängigen Anmeldung mit dem Titel "Efficient Parallel Iterative Decoding" beschrieben ist, die auf den Anmelder der vorliegenden Erfindung übertragen ist. Der Prozessor **100** weist einen ersten Decodierer **102** auf, und zwar ähnlich dem Decodierer **20** zum Decodieren eines ersten Codes und einen zweiten Decodierer **106** ebenfalls ähnlich oder gleich dem Decodierer **20** zum Decodieren eines zweiten Codes, wobei beide zur Codierung von Eingangsdaten angewandt werden. Vorzugsweise sind die ersten und zweiten Codes unterschiedlich. Eine Steuereinheit **104** steuert vorzugsweise den Betrieb der Decodierer. Vorzugsweise wird ein codiertes Paket iterativ zwischen den Decodierern **102** und **106** zurück und vorwärts hindurchgeleitet, bis das Paket hinreichend decodiert ist.

[0056] Es sei nunmehr bemerkt, dass Decodierer ähnlich oder gleich dem Decodierer **20** in einer großen Verschiedenheit von Decodierprozessoren verwendet werden können, wobei der Prozessor **100** nur eine bevorzugte Bauart des Prozessors ist.

[0057] Fig. 8 ist ein Blockdiagramm einer Minimum-Recheneinheit **108**, die anstelle der Einheit **56** verwendet werden kann, und zwar gemäß einem weiteren bevorzugten Ausführungsbeispiel der Erfindung. Eine Vielzahl von Auswahleinheiten **90** bestimmt die NM, wie dies oben unter Bezugnahme auf die Einheit **56** in Fig. 6 beschrieben wurde. Die zum Berechnen des Minimums erforderliche Zeit kann jedoch die Zeit eines Takt- oder Clockzyklus übersteigen, der erforderlich ist zur Berechnung der neuen Zustandsmetriken durch die Kombinierer **50**. Um daher den Taktzyklus nicht zu verlängern und den Betrieb des Decodierers **20** zu verlangsamen, wird das Minimum durch Einheit **108** in einem einzigen Taktzyklus berechnet aber wird nicht durch die Kombinierer **50** in dem gleichen Zyklus verwendet. Vielmehr verwenden die Kombinierer **50** vorzugsweise eine NM berechnet in einem vorhergehenden Takt- oder Clockzyklus.

[0058] Vorzugsweise wird die berechnete NM in einem Register **114** gespeichert. In einem darauf folgenden Zyklus werden die Inhalte des Registers **114** zu den Kombinierern **50** geleitet und zwar zur Verwendung bei der Normalisierung. In der Zwischenzeit wurde jedoch das Minimum von dem vorhergehenden Taktzyklus von den Zustandsmetriken in den Kombinierern **50** subtrahiert. Daher bewahrt das Re-

gister **114** vorzugsweise das Minimum von dem vorhergehenden Taktzyklus, welches sodann von dem Minimum von dem laufenden Taktzyklus durch einen Subtrahierer **112** subtrahiert wird. Die Differenz im Register **114** wird zu den Kombinierern **50** geleitet. Es sei bemerkt, dass andere Ausbildungen von einem oder mehreren Registern zum Speichern des Minimums alternativ verwendet werden können.

[0059] Obwohl die obige Beschreibung die Verwendung eines APP-Decodierers eines bestimmten Codes mit nur zwei Codebits in jeder Gruppe verwendet, erkennt der Fachmann, dass die Prinzipien der vorliegenden Erfindung auch auf einen weiten Bereich von iterativen Decodierern angewandt werden können, und zwar einschließlich SOVA- und MAP-Decodierern und ferner auch bei Decodierern, die irgendeine Anzahl von Codebits in jeder Gruppe besitzen.

[0060] Es sei bemerkt, dass die oben beschriebenen bevorzugten Ausführungsbeispiele Beispiele sind und dass der vollständige Bereich der Erfindung nur durch die Ansprüche begrenzt ist.

Patentansprüche

1. Ein Verfahren zur Normalisierung einer Vielzahl von Zustandsmetrikregistern (**30**, **32**, **34**) in einem Decoder (**20**, **102**, **106**) unter Verwendung eines Trellis, wobei das Verfahren die folgende Schritte aufweist:
Bestimmen eines Minimums von jeweiligen Zustandsmetrikwerten, die in der Vielzahl von Zustandsmetrikregistern (**30**, **32**, **34**) gespeichert sind; und
Subtrahieren des Minimums von den Werten;
dadurch gekennzeichnet, dass:
das Minimum ein ungefähres Minimum ist;
das Bestimmen das Bestimmen des Minimums einer vorbestimmten Anzahl von höchstwertigen Bits in der Vielzahl von Zustandsmetrikregistern (**30**, **32**, **34**) aufweist, wobei die vorbestimmte Anzahl geringer ist als die Gesamtanzahl von Bits in den Registern; und
das Subtrahieren das Subtrahieren des ungefähren Minimums von der vorbestimmten Anzahl von höchstwertigen Bits der Register (**30**, **32**, **34**) aufweist.

2. Verfahren gemäß Anspruch 1, wobei das ungefähre Minimum immer gleich oder weniger als ein tatsächliches Minimum der Werte in der Vielzahl von Zustandsmetrikregistern (**30**, **32**, **34**) ist.

3. Verfahren gemäß einem der vorhergehenden Ansprüche, wobei die vorbestimmte Anzahl von höchstwertigen Bits zwischen 30% und 60% der Anzahl von Bits in dem Register (**30**, **32**, **34**) aufweist.

4. Verfahren gemäß Anspruch 1, das weiterhin den Schritt des Berechnens neuer Zustandsmetrikwerte für die Register (**30**, **32**, **34**) aufweist und wobei das Subtrahieren des ungefähren Minimums im Wesentlichen gleichzeitig mit der Berechnung der neuen Werte ausgeführt wird.

5. Verfahren gemäß Anspruch 1, das weiterhin den Schritt des Speicherns in den Zustandsmetrikregistern (**30**, **32**, **34**) während der Vielzahl von aufeinander folgenden Taktzyklen von einer Vielzahl von jeweiligen neuen Werten aufweist, wobei das Bestimmen des ungefähren Minimums das Bestimmen eines ungefähren Minimums der neuen Werte im Wesentlichen in jedem Zyklus aufweist.

6. Ein Verfahren gemäß Anspruch 5, wobei das Bestimmen des ungefähren Minimums das Bestimmen eines ungefähren Minimums während einem ersten Zyklus der Vielzahl von Taktzyklen aufweist und wobei das Subtrahieren des ungefähren Minimums das Subtrahieren während eines zweiten Zykluses der Vielzahl von Taktzyklen nach dem ersten aufweist.

7. Verfahren gemäß Anspruch 6, das weiterhin das Anpassen des ungefähren Minimums ansprechend auf eine Zeitlücke zwischen den ersten und zweiten Taktzyklen aufweist.

8. Ein Verfahren gemäß Anspruch 7, wobei das Anpassen des ungefähren Minimums das Subtrahieren des ungefähren Minimums, das während des ersten Zykluses berechnet wurde, von dem ungefähren Minimum, das während des zweiten Zykluses berechnet wurde, aufweist.

9. Eine Zustandsmetrikberechnungseinheit (**30**, **32**, **34**) zur Verwendung in einem Decoder (**20**, **102**, **106**), der mittels eines Trellis decodiert, wobei die Einheit Folgendes aufweist:
Eine Vielzahl vom Zustandsmetrikregistern (**30**, **32**, **34**), die jeweilige Zustandsmetrikwerte aufweist;
eine Minimumberechnungseinheit (**56**, **108**), die betriebsmäßig ein Minimum der jeweiligen Zustandmetrikwerte bestimmt; und
eine Vielzahl von Subtrahierern (**78**), die betriebsmäßig das Minimum von den Werten subtrahiert;
dadurch gekennzeichnet, dass
das Minimum ein ungefähres Minimum ist;
die Minimumberechnungseinheit (**56**, **108**) betriebsmäßig ist, um das Minimum einer vorbestimmten Anzahl von höchstwertigen Bits in der Vielzahl von Zustandsmetrikregistern (**30**, **32**, **34**) zu bestimmen, wobei die vorbestimmte Anzahl geringer ist als die Gesamtanzahl von Bits in den Registern (**30**, **32**, **34**); und
der Subtrahierer (**78**) betriebsmäßig ist, um das ungefähre Minimum von der vorbestimmten Anzahl von höchstwertigen Bits der Register (**30**, **32**, **34**) abzuziehen.

10. Einheit (**30**, **32**, **34**) gemäß Anspruch 9, die weiterhin eine Vielzahl von Rekursivkombinierern (**50**) aufweist, die betriebsmäßig für jedes Register (**30**, **32**, **34**) eine Metrik für den nächsten Zustand des Registers (**30**, **32**, **34**) berechnet und wobei jeder Rekursivkombinierer (**50**) einen jeweiligen Subtrahierer der Subtrahierer (**78**) aufweist.

11. Einheit (**30**, **32**, **34**) gemäß Anspruch 9, wobei der Decoder (**20**, **102**, **106**) einen Viterbi-Decoder aufweist.

12. Einheit (**30**, **32**, **34**) gemäß Anspruch 9, wobei der Decoder (**20**, **102**, **106**) einen APP Decoder aufweist.

13. Einheit (**30**, **32**, **34**) gemäß Anspruch 9, wobei der Decoder (**20**, **102**, **106**) einen MAP Decoder aufweist.

14. Einheit (**30**, **32**, **34**) gemäß Anspruch 9, wobei der Decoder (**20**, **102**, **106**) einen Trellis-Decoder aufweist.

15. Einheit (**30**, **32**, **34**) gemäß Anspruch 9, wobei der Decoder (**20**, **102**, **106**) einen Soft Output Viterbi Decodierer bzw. einen Viterbi-Decodierer mit weicher Ausgabe aufweist.

16. Ein iterativer Decodierungsprozessor (**100**) zum iterativen Decodieren einer Sequenz von Signalpaketen, die gemäß einem Multikomponenten-Codierungsschema codiert werden, wobei der Prozessor eine Vielzahl von Berechnungseinheiten (**102**, **106**), wie sie in Anspruch 9 beschrieben sind, aufweist.

Es folgen 7 Blatt Zeichnungen

Anhängende Zeichnungen

FIG. 1

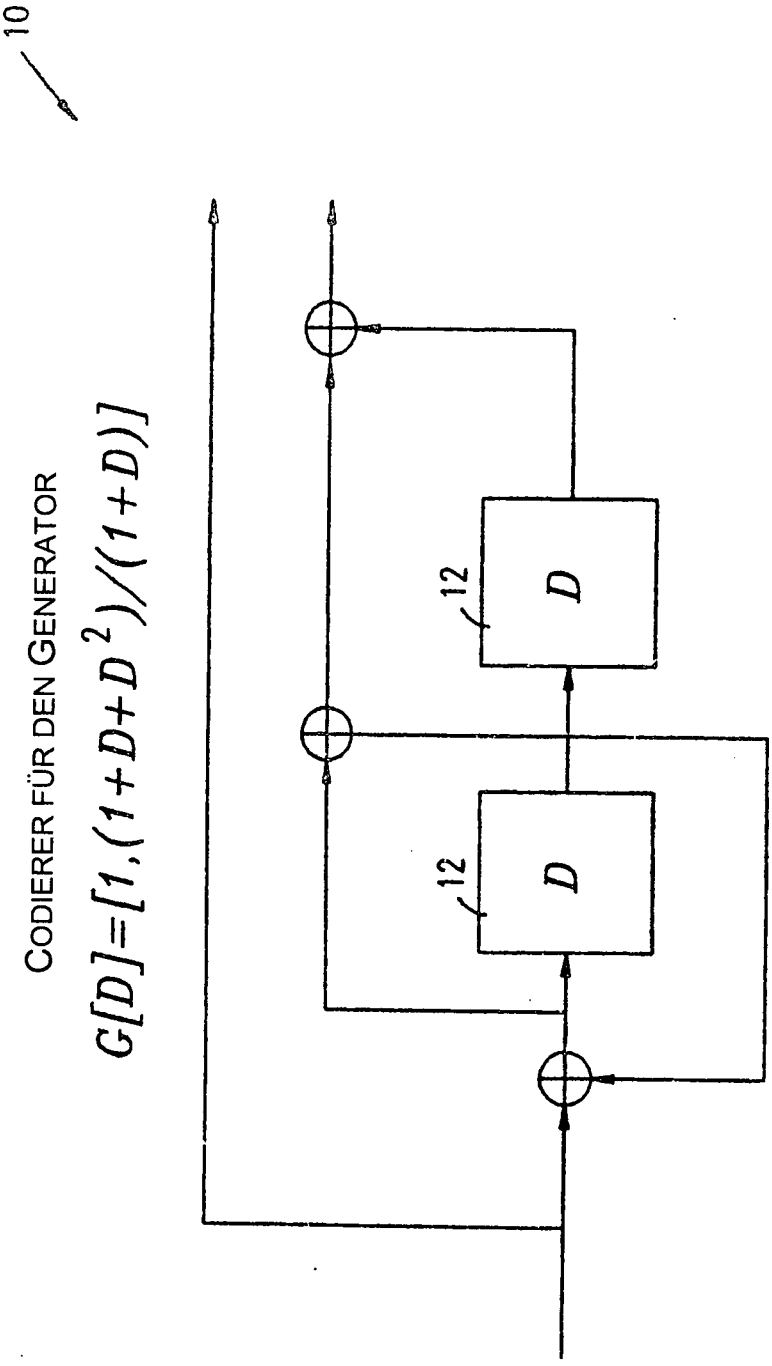

FIG. 2

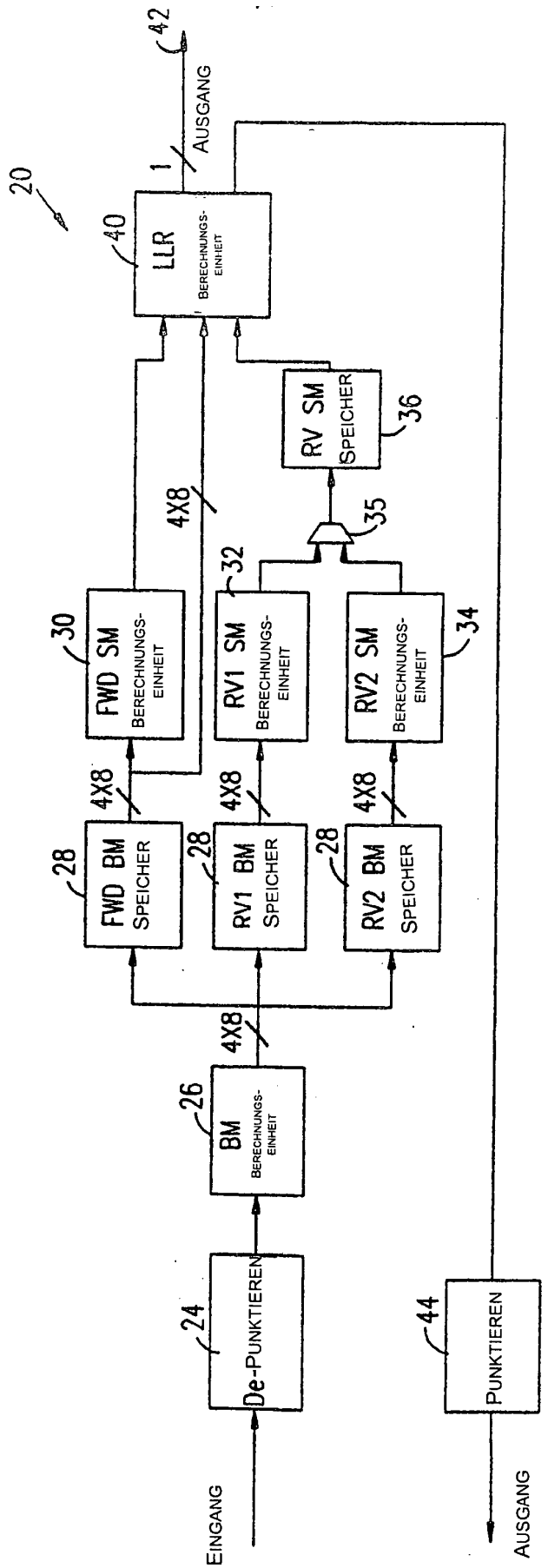
20
42
AUSGANG
1
40
LLR
BERECHNUNGS-EINHEIT
RV SM SPEICHER
36
35
4X8
32
30
FWD SM
BERECHNUNGS-EINHEIT
RV1 SM
BERECHNUNGS-EINHEIT
RV2 SM
BERECHNUNGS-EINHEIT
34
4X8
4X8
4X8
28
FWD BM SPEICHER
28
RV1 BM SPEICHER
28
RV2 BM SPEICHER
4X8
26
BM
BERECHNUNGS-EINHEIT
24
De-PUNKTIEREN
44
PUNKTIEREN
EINGANG
AUSGANG

FIG. 4

$C[D]=[1,((1+D+D^2)/(1+D))]$

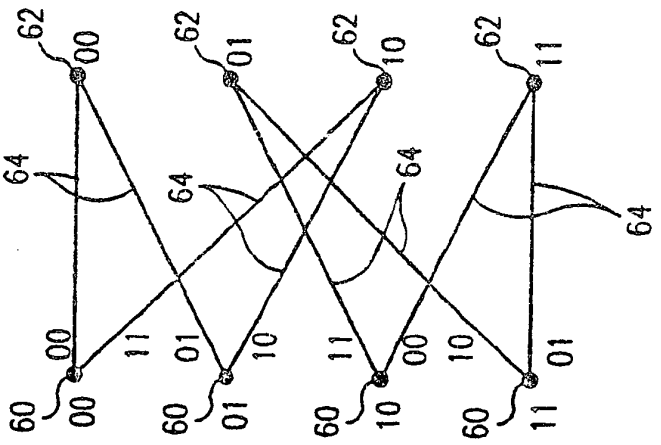

FIG. 3

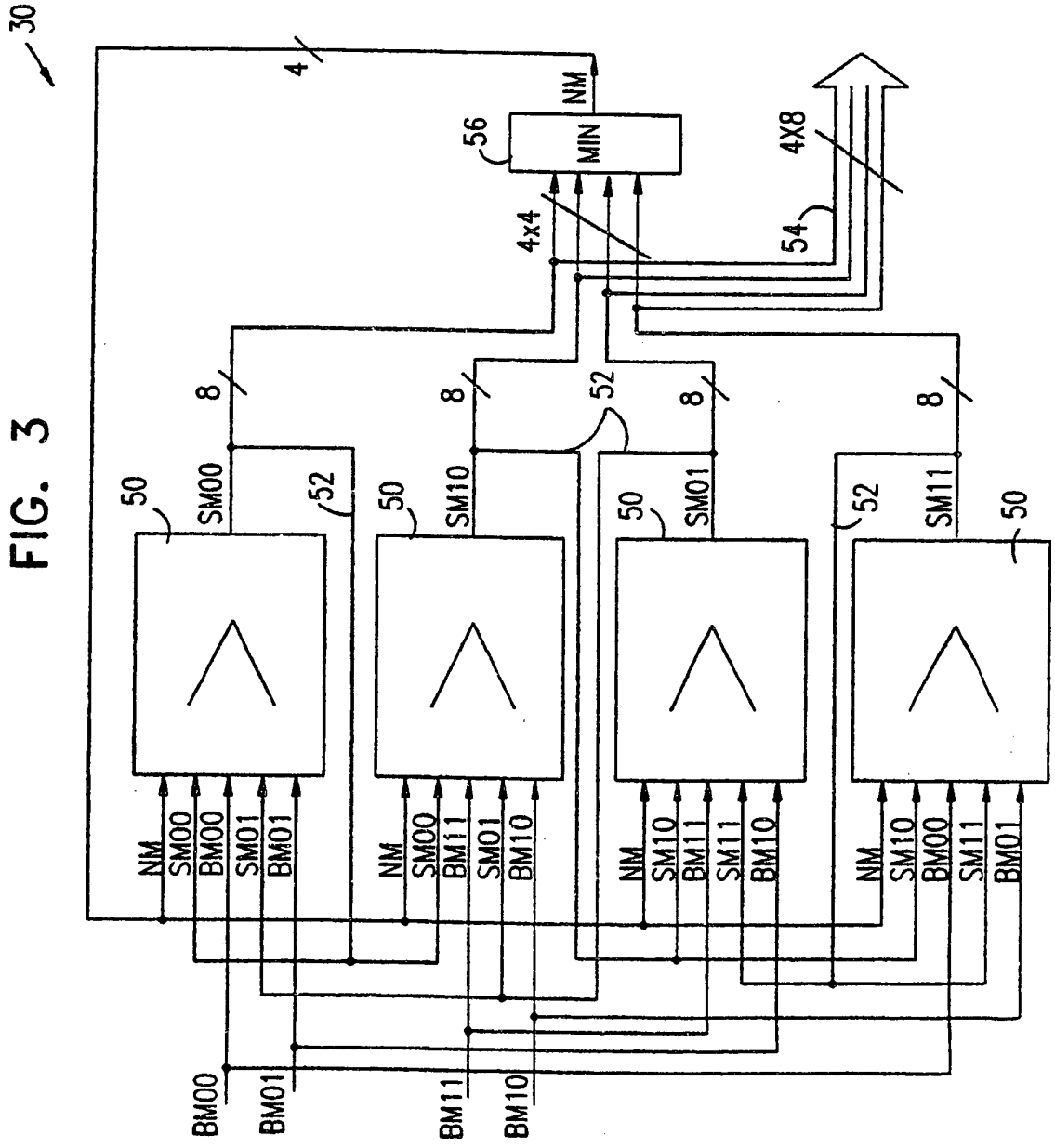

FIG. 5

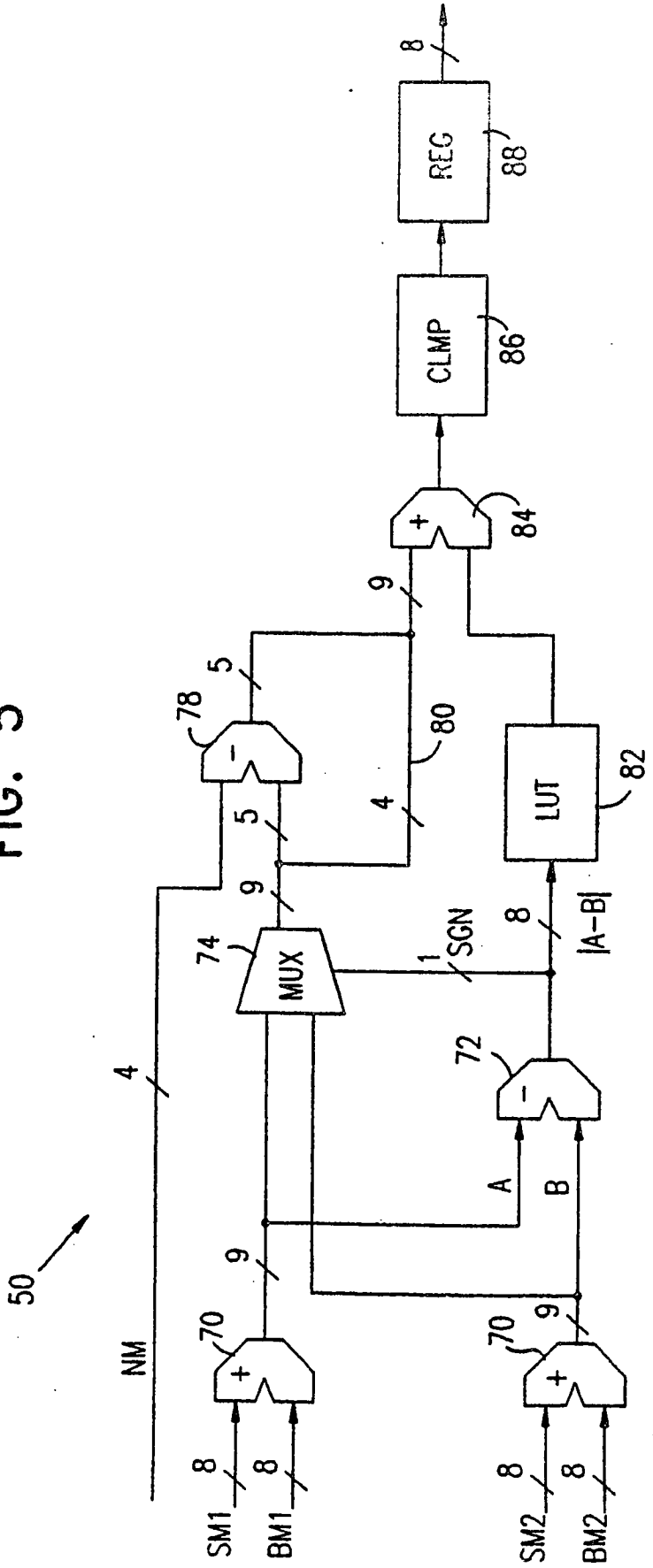
50
NM
4
SM1
8
BM1
8
70
+
9
SM2
8
BM2
8
70
+
9
A
B
72
-
9
|A-B|
8
SGN
1
74
MUX
9
5
78
-
5
4
80
LUT
82
9
+
84
CLMP
86
REG
88
8

FIG. 6

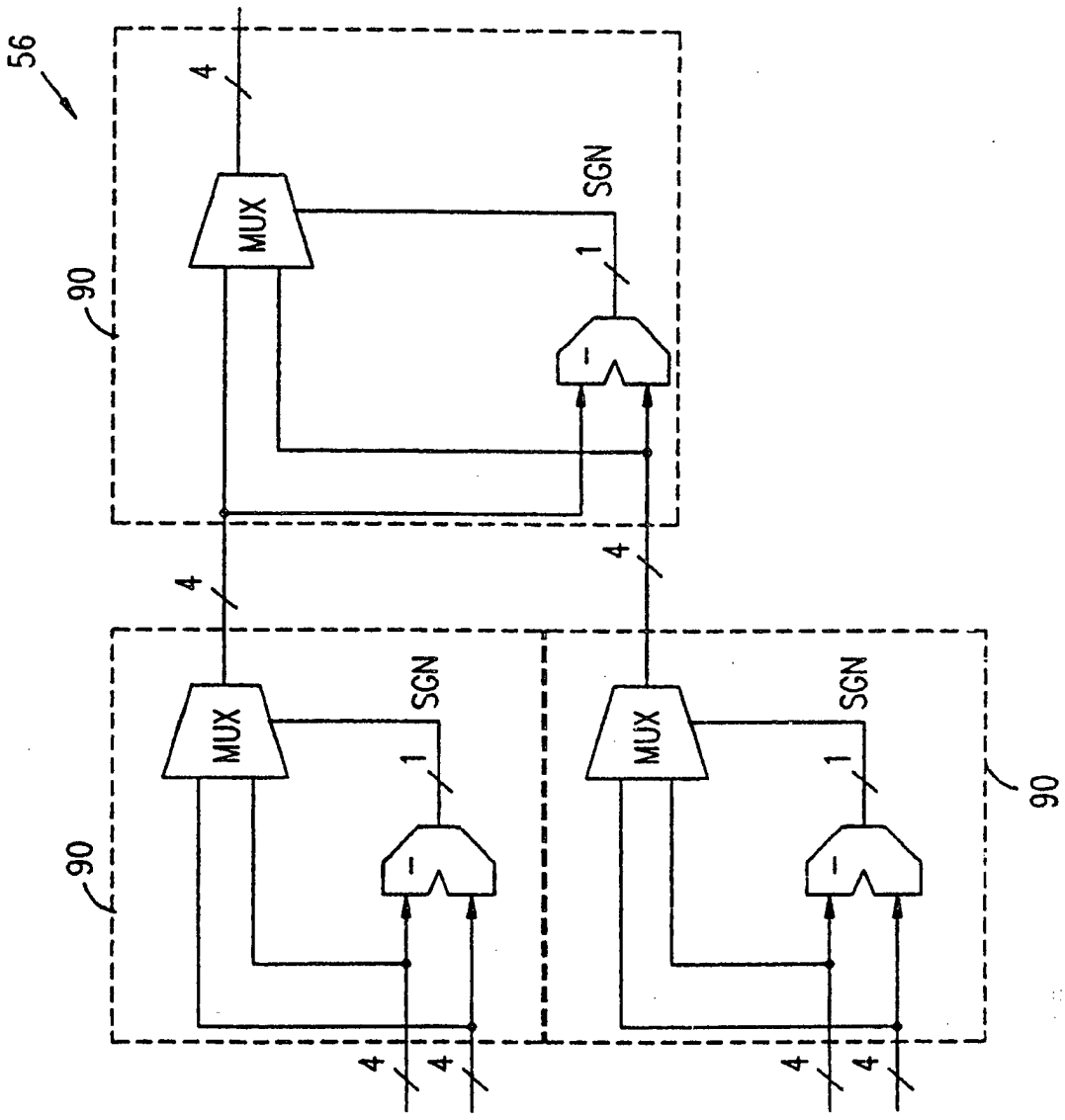

FIG. 7

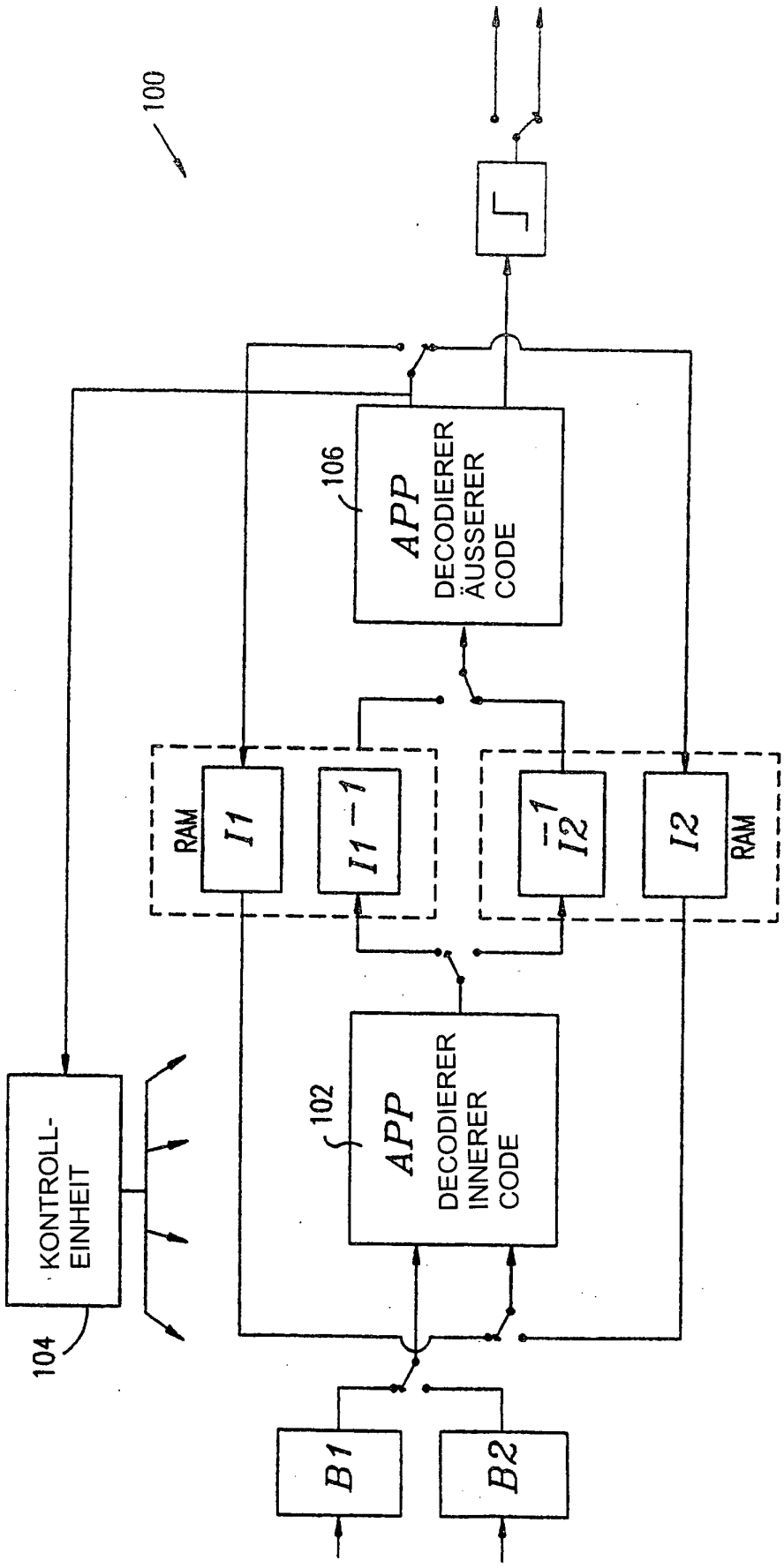
100
106
APP
DECODIERER
ÄUSSERER
CODE
RAM
I1
I1⁻¹
I2⁻¹
I2
RAM
102
APP
DECODIERER
INNERER
CODE
KONTROLL-
EINHEIT
104
B1
B2

FIG. 8