

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2004-248331

(P2004-248331A)

(43) 公開日 平成16年9月2日(2004.9.2)

(51) Int.Cl.⁷
H04J 13/04F I
H04J 13/00G
テーマコード (参考)
5K022

審査請求 有 請求項の数 26 O L (全 68 頁)

(21) 出願番号	特願2004-163766 (P2004-163766)	(71) 出願人	594001292
(22) 出願日	平成16年6月1日 (2004.6.1)		サムスン エレクトロニクス カンパニー
(62) 分割の表示	特願2001-508108 (P2001-508108)		リミテッド
原出願日	平成12年7月6日 (2000.7.6)		大韓民国 キョンギード スウォンーシ
(31) 優先権主張番号	1999/27932	(74) 代理人	100064908
(32) 優先日	平成11年7月6日 (1999.7.6)		弁理士 志賀 正武
(33) 優先権主張国	韓国 (KR)	(74) 代理人	100089037
			弁理士 渡邊 隆
		(72) 発明者	ジェーヨル・キム
			大韓民国・キョンギード・435-042
			・クンポーシ・サンボン・2ードン (番地
			なし)・サンボン・9ーダンジ・ベクトゥ
			ー・エーピーティ・#960-1401
最終頁に続く			

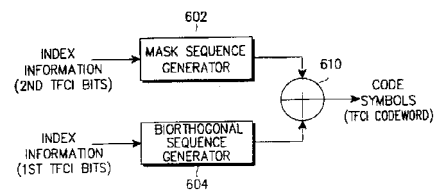
(54) 【発明の名称】 符号分割多重接続移動通信システムの伝送率情報符号化／復号化装置及び方法

(57) 【要約】

【課題】 符号分割多重接続移動通信システムの伝送率情報符号化／復号化装置及び方法を提供する。

【解決手段】 同一シンボルのシーケンスを発生する1ビット発生器と、複数の基底直交シーケンスを発生する基底直交シーケンス発生器と、複数の基底マスクシーケンスを発生する基底マスクシーケンス発生器と、相互直交シーケンスへの変換を示す第1情報部分と、直交シーケンスへの変換を示す第2情報部分と、マスクシーケンスへの変換を示す第3情報部分を含む伝送率情報ビットを受信し、基底直交シーケンスの中、第2情報部分により選択される直交シーケンスと、選択された直交シーケンスと第1情報部分により選択される同一シンボルの組み合わせに構成される相互直交シーケンスと、相互直交シーケンスと第3情報部分により選択されるマスクシーケンスとを組み合わせる伝送率情報シーケンスに発生する演算部を含む。

【選択図】 図6



【特許請求の範囲】

【請求項 1】

第 1 情報ビットと第 2 情報ビットとを含む符号化装置において、
複数の基底シーケンスを生成し、複数のシーケンスの中、前記第 1 情報ビットにより選択されたシーケンスを出力するシーケンス発生器と、

複数の基底マスクシーケンスを生成し、複数のマスクシーケンスの中、前記第 2 情報ビットにより選択されたマスクシーケンスを出力するマスクシーケンス発生器と、

前記シーケンス発生器と前記マスクシーケンス発生器とから受信された前記シーケンスと前記マスクシーケンスとを加算する加算器と

を含むことを特徴とする通信システムの符号化装置。

10

【請求項 2】

前記複数のシーケンスは、ウォルシュ符号と前記ウォルシュ符号の相互直交補数シーケンスであることを特徴とする請求項 1 に記載の符号化装置。

【請求項 3】

前記マスクシーケンス発生器は、ゴールドコードを形成するために共に加われる第 1 m シーケンス及び第 2 m シーケンスを備え、前記第 1 m シーケンスを循環して形成されるシーケンスを有する第 1 シーケンスグループと、前記第 2 m シーケンスを循環して形成されるシーケンスを有する第 2 シーケンスグループとを形成し、前記第 1 グループの前記シーケンスを直交シーケンスに変換するために、列置換関数を前記第 1 グループの前記シーケンスに印加し、前記第 2 グループの前記シーケンスの先に 0 の列を挿入し、前記第 2 グループの前記シーケンスを前記マスクシーケンスに変換するために、逆列置換関数を発生し前記第 2 グループの前記シーケンスに印加することを特徴とする請求項 1 に記載の符号化装置。

20

【請求項 4】

前記複数の基底シーケンスは、長さ 32 の第 1 ウォルシュ符号、第 2 ウォルシュ符号、第 4 ウォルシュ符号、第 8 ウォルシュ符号、第 16 ウォルシュ符号及び全て “1” のシーケンスであることを特徴とする請求項 1 に記載の符号化装置。

【請求項 5】

前記基底マスクシーケンスは、“001010000110001111110000
01110111”、“000000001110011010110110111000
111”、“00001010111110010001101100101011”、
“00011100001101110010111101010001”であることを
特徴とする請求項 1 に記載の符号化装置。

30

【請求項 6】

前記加算器の出力信号は、#0 及び #16 の位置で穿孔されることを特徴とする請求項 4 または請求項 5 のいずれかに記載の符号化装置。

【請求項 7】

前記符号化装置は、長さ 32 のウォルシュ符号から #0 及び #16 シンボルが除かれた長さ 30 のシーケンスを出力することを特徴とする請求項 1 から請求項 5 のいずれかに記載の符号化装置。

40

【請求項 8】

前記符号化装置は、長さ 32 の基底マスクシーケンスから #0 及び #16 シンボルが除かれた長さ 30 のシーケンスを出力することを特徴とする請求項 7 に記載の符号化装置。

【請求項 9】

前記基底ウォルシュ符号 W1 は “010101010101010101010101010101010101010101” であり、

前記基底ウォルシュ符号 W2 は “001100110011001100110011001100110011001100” であり、

前記基底ウォルシュ符号 W4 は “000011110000111100001111000011110000111100001111” であり、

50

前記基底ウォルシュ符号W8は“000000001111111100000000
11111111”であり、

前記基底ウォルシュ符号W16は“0000000000000000000011111111
11111111”である

ことを特徴とする請求項7に記載の符号化装置。

【請求項10】

前記基底ウォルシュ符号W1は“0101010101010101010101010101
01010101”であり、

前記基底ウォルシュ符号W2は“0011001100110011001100110011
00110011”であり、

前記基底ウォルシュ符号W4は“000011110000111100001111
00001111”であり、

前記基底ウォルシュ符号W8は“000000001111111100000000
11111111”であり、

前記基底ウォルシュ符号W16は“0000000000000000000011111111
11111111”である

ことを特徴とする請求項8に記載の符号化装置。

【請求項11】

前記基底マスクシーケンスは、第1マスクシーケンスM1“001010000110
00111111000001110111”、第2マスクシーケンスM2“000000
001110011010110110111000111”、第3マスクシーケンスM
4“00001010111110010001101100101011”及び第4マ
スクシーケンスM8“000111000011011100101111010100
01”であることを特徴とする請求項9に記載の符号化装置。

【請求項12】

前記基底マスクシーケンスは、第1マスクシーケンスM1“001010000110
00111111000001110111”、第2マスクシーケンスM2“000000
001110011010110110111000111”、第3マスクシーケンスM
4“00001010111110010001101100101011”及び第4マ
スクシーケンスM8“000111000011011100101111010100
01”であることを特徴とする請求項10に記載の符号化装置。

【請求項13】

前記符号化装置は、前記基底相互直交シーケンスと前記基底マスクシーケンスとの合計
を求めることによって生成されたシーケンスから#0及び#16シンボルが除かれた長さ
30のシーケンスを出力することを特徴とする請求項5に記載の符号化装置。

【請求項14】

前記基底相互直交シーケンスは、

“0101010101010101010101010101010101010101010101010101010101010101”、

“0011001100110011001100110011001100110011001100110011001100110011”、

“0000111100001111000011110000111100001111000011110000111100001111”、

“0000000011111111000000001111111100000000111111110000000011111111”であり、

前記基底マスクシーケンスは、

“001010000110001111110000001110111”、

“00000001110011010110110110111000111”、

“00001010111110010001101100101011”

“00011100001101110010111101010001”であることを
特徴とする請求項13に記載の符号化装置。

【請求項15】

第1情報ビットと第2情報ビットとを含む情報ビットを符号化する方法において、

複数の基底シーケンスを生成し、複数のシーケンスの中、前記第1情報ビットにより選

10

20

30

40

50

択されたシーケンスを出力する過程と、

複数の基底マスクシーケンスを生成し、複数のマスクシーケンスの中、前記第 2 情報ビットにより選択されたマスクシーケンスを出力する過程と、

前記選択されたシーケンスと前記選択されたマスクシーケンスとを加算する過程とを含むことを特徴とする通信システムの符号化方法。

【請求項 16】

前記複数のシーケンスは、長さ 32 のウォルシュ符号と前記ウォルシュ符号の補数符号であることを特徴とする請求項 15 に記載の符号化方法。

【請求項 17】

前記複数の基底マスクシーケンスは、第 1 m シーケンスと第 2 m シーケンスとを加算することによって生成されるゴールドシーケンスから生成され、前記第 1 m シーケンスを循環して形成されるシーケンスを有する第 1 シーケンスグループと、前記第 2 m シーケンスを循環して形成されるシーケンスを有する第 2 シーケンスグループとを形成し、前記第 1 グループの前記シーケンスを直交シーケンスに変換するために、列置換関数を前記第 1 グループの前記シーケンスに印加し、前記第 2 グループの前記シーケンスの先に 0 の列を挿入し、前記第 2 グループの前記シーケンスを前記マスクシーケンスに変換するために、逆列置換関数を前記第 2 グループの前記シーケンスに印加することを特徴とする請求項 16 に記載の符号化方法。 10

【請求項 18】

前記複数の基底シーケンスは、長さ 32 の第 1 ウォルシュ符号 (W1)、第 2 ウォルシュ符号 (W2)、第 4 ウォルシュ符号 (W4)、第 8 ウォルシュ符号 (W8)、第 16 ウォルシュ符号 (W16) 及び全て “1” のシーケンスであることを特徴とする請求項 15 に記載の符号化方法。 20

【請求項 19】

前記複数の基底マスクシーケンスは、“0010100001100011111100001110111”、“00000001110011010110110111000111”、“00001010111110010001101100101011”、“00011100001101110010111101010001”であることを特徴とする請求項 15 に記載の符号化方法。

【請求項 20】

前記符号化は、長さ 32 のウォルシュ符号から #0 及び #16 シンボルが除かれた長さ 30 のシーケンスを出力することを特徴とする請求項 15 に記載の符号化方法。 30

【請求項 21】

前記符号化は、長さ 32 の基底マスクシーケンスから #0 及び #16 シンボルが除かれた長さ 30 のシーケンスを出力することを特徴とする請求項 15 に記載の符号化方法。

【請求項 22】

前記基底ウォルシュ符号 W1 は “010101010101010101010101010101010101010101” であり、

前記基底ウォルシュ符号 W2 は “001100110011001100110011001100110011001100” であり、 40

前記基底ウォルシュ符号 W4 は “0000111100001111000011110000111100001111” であり、

前記基底ウォルシュ符号 W8 は “000000001111111111110000000011111111” であり、

前記基底ウォルシュ符号 W16 は “000000000000000000001111111111111111111111” である

ことを特徴とする請求項 18 に記載の符号化方法。

【請求項 23】

前記基底マスクシーケンスは、第 1 マスクシーケンス M1 “0010100001100011111111000001110111”、第 2 マスクシーケンス M2 “00000 50

001110011010110110111000111”、第3マスクシーケンスM4 “00001010111110010001101100101011”及び第4マスクシーケンスM8 “00011100001101110010111101010001”であることを特徴とする請求項22に記載の符号化方法。

【請求項24】

前記符号化は、前記基底相互直交シーケンスと前記基底マスクシーケンスとの合計を求めることによって生成されたシーケンスから#0及び#16シンボルが除かれた長さ30のシーケンスを出力することを特徴とする請求項15に記載の符号化方法。

【請求項25】

前記基底相互直交シーケンスは、

“010101010101010101010101010101010101010101010101”、
 “001100110011001100110011001100110011001100110011”、
 “0000111100001111000011110000111100001111”、
 “0000000011111111100000000011111111”であり、

前記基底マスクシーケンスは、

“001010000110001111110000001110111”、
 “00000001110011010110110110111000111”、
 “00001010111110010001101100101011”
 “00011100001101110010111101010001”であることを

特徴とする請求項24に記載の符号化方法。

【請求項26】

同一シンボルを発生する過程と、

複数の基底シーケンスを発生する過程と、

複数の基底マスクシーケンスを発生する過程と、

前記同一シンボルと前記複数の基底シーケンスと前記複数の基底マスクシーケンスとを対応する伝送率情報ビットとそれぞれ乗算する過程と、

各乗算から生成されたシーケンスを加算する過程と

を含むことを特徴とする符号分割多重接続移動通信システムの伝送率情報符号化方法。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は符号分割多重接続移動通信システムで情報を伝送するための装置及び方法に関するもので、特に、伝送率情報(Transport Format Combination Indicator、以下、TFCI)を伝送するための装置及び方法に関する。

【背景技術】

【0002】

一般的に符号分割多重接続方式を使用する移動通信システム(以下、IMT2000システム)では一つの物理的チャネル、例えば、専用物理データチャネル(DPDCH)内に音声サービス、画像サービス、文字サービスなどのような各種サービスを提供するフレームを伝送する。前記サービスフレームは固定されたデータ伝送率、または可変的なデータ伝送率に伝送される。固定された伝送率にサービスフレームを伝送する場合には、伝送する各サービスフレームの拡散レート(spreading rate)を受信側に知らせる必要がない。しかし、可変的な伝送率にサービスフレームを伝送する場合には、伝送される各サービスフレームが異なる伝送率により伝送されることができるので、送信側は各サービスフレームの伝送率により決定される拡散レートを受信側に知らせるべきである。この時、通常的なIMT2000システムで伝送率とデータ伝送速度は比例関係であり、前記データ伝送速度とデータ拡散レートは反比例関係である。

【0003】

上述した可変的な伝送率にサービスフレームを伝送する場合に、現在伝送されているサービスフレームの伝送率を受信側に知らせる役割をすることが専用物理制御チャネル(D

10

20

30

40

50

P C C H)内に含まれたT F C Iフィールドである。前記T F C Iフィールドにはサービスフレームの伝送率を含む情報を示す伝送率情報(T F C I)が含まれる。前記T F C Iはそれぞれの伝送率による音声、またはデータサービスが正しく遂行されるようにする情報である。

【0004】

図1 A乃至図1 Dに前記T F C Iの使用例を示す。前記図1 Aの例はアップリンク(up-link)専用物理データチャネル(D P D C H)とアップリンク(up-link)専用物理制御チャネル(D P C C H)での適用を示している。前記図1 Bの例はランダムアクセスチャネル(R A C H)での適用を示している。前記図1 Cの例はダウンリンク(down-link)専用物理データチャネル(D P D C H)とダウンリンク専用物理制御チャネル(D P C C H)での適用を示している。前記図1 Dの例は第2共通物理制御チャネル(S C C P C H)での適用を示している。 10

【0005】

前記図1 A乃至図1 Dを参照すると、一つのフレームは16個のスロットに構成され、前記各スロットはT F C Iフィールドを有する。従って、一つのフレームは16個のT F C Iフィールドを含む。前記T F C Iフィールドは $N_{T F C I}$ ビットに構成される。通常的に伝送率情報(T F C I)は32ビットになされる。一つのフレームに32ビットのT F C Iを送送するためには、一つのフレームを構成する16スロット($T_{s l o t} = 0.625 \text{ ms}$)ごと2ビットのT F C Iビットが割り当てられる。

【0006】

図2は通常的なI M T - 2000システムの基地局送信器構造を示している図である。 20

【0007】

前記図2を参照すると、乗算器211、231、232は入力信号を利得係数 G_1 、 G_3 、 G_5 とかけて出力する。乗算器221、241、242は対応するT F C I符号器から受信されたT F C I符号語(符号化シンボル)と利得係数 G_2 、 G_4 、 G_6 をかけて出力する。前記利得係数 G_1 、 G_2 、 G_3 、 G_4 、 G_5 、 G_6 はサービス種類やハンドオーバー状況などによってそれぞれ異なる値を有することもできる。前記入力信号はD P D C HデータとD P C C H信号を構成するパイロット信号(Pilot)、電力制御信号(T P C)などを含む。マルチプレクサ212は図1 Cに示したように、前記乗算器221から受信される32ビットの符号化シンボルを前記T F C Iフィールドに挿入して出力する。マルチプレクサ242は前記乗算器241から受信される32ビットの符号化シンボルを前記T F C Iフィールドに挿入して出力する。マルチプレクサ252は前記乗算器242から受信される32ビットの符号化シンボルを前記T F C Iフィールドに挿入して出力する。前記T F C Iフィールドに符号化シンボルを挿入する例は図1 A乃至図1 Dに示したようである。前記32ビットの符号化シンボルは各チャネルのデータ信号による伝送速度情報などを定義するT F C Iビット(情報ビット)を符号化することにより得られるシンボルである。第1、第2及び第3直並列変換器(以下、S/P)213、233、234は、前記マルチプレクサ212、242、252からの出力をIチャネルとQチャネルに分離する。乗算器214、222、235、236、237、238は前記直並列変換器213、233、234からの出力をチャネル区分コード(Channelization code) $C_{c h 1}$ 、 $C_{c h 2}$ 、 $C_{c h 3}$ とかける。前記チャネル区分コードは直交符号が使用される。前記乗算器214、235、237の出力は第1合算器215で合算されIチャネル信号に生成される。前記乗算器222、236、238の出力は第2合算器223で合算されQチャネル信号に生成される。位相遷移器224は前記第2合算器223から受信されたQチャネル信号の位相を 90° シフトする。合算器216は第1合算器215の出力と位相遷移器224の出力を合算して複素信号 $I + j Q$ を生成する。乗算器217は前記複素信号を各基地局に割り当てられたP Nシーケンス($C_{s c r a m b}$)によりスクランブリングする。第4 S/P 218は前記スクランブリングされた信号を直/並列変換してIチャネル及びQチャネルに分配する。前記第4 S/P 218の出力はIチャネル及びQチャネル別に低域ろ波器219と225を通過して帯域幅が制限された信号が生成される。前記低域ろ波器219 40 50

と 2 2 5 の出力信号は乗算器 2 2 0 と 2 2 6 で搬送波 $\cos(2\pi f_c t)$ 、 $\sin(2\pi f_c t)$ とかけられて高周波帯域に遷移される。合算器 2 2 7 は前記高周波帯域に遷移された I チャンネルと Q チャンネルの信号を合わせて出力する。

【0008】

図 3 は通常的な I M T - 2 0 0 0 システムの移動局送信器の構造を示す図である。

前記図 3 を参照すると、乗算器 3 1 1、3 2 1、3 2 3 は対応する信号をチャンネル区分コード(Channelization code) C_{ch1} 、 C_{ch2} 、 C_{ch3} とかけて出力する。信号 1、2、3 は第 1、第 2、第 3 D P D C H 信号である。入力信号 4 は D P C C H のパイロット信号(Pilot)と電力制御信号(T P C)などを含む。T F C I 情報ビットは T F C I 符号器 3 0 9 により 3 2 ビット T F C I 符号語シンボルに符号化される。乗算器 3 1 0 は図 1 A に示したように、3 2 ビット T F C I 符号化シンボルを信号 4 に挿入する。乗算器 3 2 5 は前記乗算器 3 1 0 からの T F C I 符号化シンボルを含む D P C C H 信号とチャンネル区分コード C_{ch4} をかける。前記チャンネル区分コード C_{ch1} 、 C_{ch2} 、 C_{ch3} 、 C_{ch4} は直交符号が使用される。前記 3 2 ビットの符号化シンボルは D P D C H 信号の伝送速度情報などを定義する T F C I 情報ビットを符号化することにより得られるシンボルである。前記乗算器 3 1 1、3 2 1、3 2 3、3 2 5 の出力は乗算器 3 1 2、3 2 2、3 2 4、3 2 6 で利得係数 G_1 、 G_2 、 G_3 、 G_4 とかけられる。前記利得係数 G_1 、 G_2 、 G_3 、 G_4 はそれぞれ異なる値を有することもできる。前記乗算器 3 1 2、3 2 2 の出力は第 1 合算器 3 1 3 で合算され I チャンネル信号に出力され、乗算器 3 2 4、3 2 6 の出力は第 2 合算器 3 2 7 で合算され Q チャンネル信号に出力される。位相遷移器 3 2 8 は第 2 合算器 3 2 7 から受信される前記 Q チャンネル信号の位相を 90° シフトする。合算器 3 1 4 は第 1 合算器 3 1 3 の出力と位相遷移器 3 2 8 の出力を合算して複素信号 $I + jQ$ を生成する。乗算器 3 1 5 は前記複素信号を各基地局に割り当てられた P N シーケンス(C_{scramb})によりスクランブリングする。第 4 S / P 3 2 9 は前記スクランブリングされた信号を直／並列変換して I チャンネル及び Q チャンネルに分配する。前記第 4 S / P 3 2 9 の出力は I チャンネル及び Q チャンネル別に低域ろ波器 3 1 6 と 3 3 0 を通過して帯域幅が制限された信号が生成される。前記低域ろ波器 3 1 6、3 3 0 の出力信号は乗算器 3 1 7 と 3 3 1 で搬送波 $\cos(2\pi f_c t)$ 、 $\sin(2\pi f_c t)$ とかけられて高周波帯域に遷移される。合算器 3 1 8 は前記高周波帯域に遷移された I チャンネルと Q チャンネルの信号を合わせて出力する。

【0009】

上述したように伝送率情報を伝送する T F C I は基本型 T F C I と拡張型 T F C I に区別されることができる。前記基本型 T F C I は 6 ビットに構成された T F C I 情報ビットを利用して対応するデータチャンネルの伝送速度を含む 1 乃至 6 4 の相異なる情報を指定することができる。前記拡張型 T F C I は 7 乃至 1 0 ビットに構成された T F C I 情報ビットを利用して 1 乃至 1 2 8、1 乃至 2 5 6、1 乃至 5 1 2 及び 1 乃至 1 0 2 4 の相異なる情報を指定することができる。前記拡張型 T F C I は符号分割多重接続移動通信システムで、より多様なサービスの支援要求を満足させるために提案された。前記 T F C I ビットは送信側で伝送したサービスフレームを受信側で正しく解釈するため必ず必要な情報である。このため、前記 T F C I ビットが伝送誤りにより受信側に伝達されないと、受信側では受信したサービスフレームを正しく解釈できないようになる。従って、前記 T F C I ビットは伝送中に誤りが発生しても受信側で発生した誤りを訂正することができるように、送信側は前記 T F C I ビットを誤り訂正符号により符号化して伝送するようになる。

【0010】

図 4 A は従来の符号分割多重接続移動通信システムで基本型 T F C I ビットを符号化するための構成を概念的に示しており、図 4 B は図 4 A の相互直交符号器に適用される符号化テーブルの一例を示している。上述したように、基本型 T F C I は 1 乃至 6 4 の異なる情報を指定する 6 ビットの T F C I ビット(以下、基本型 T F C I ビット)に構成される。

【0011】

前記図 4 A 及び 4 B を参照すると、基本型 T F C I ビットは相互直交(Biorthogonal)符

号器 402 に入力され、32 シンボル(またはビット)の T F C I 符号語に出力される。前記基本型 T F C I ビットは上述したように 6 ビット表現形式を基本としている。従って、6 ビットより小さいビット数を有する基本型 T F C I ビットが前記相互直交符号器 402 に印加される場合には、前記入力された基本型 T F C I ビットの M S B (most significant bit、最左のビット)から 0 の値を入れて 6 ビット表現形式を有するようにする。一方、前記相互直交符号器 402 は入力される 6 ビット表現形式の基本型 T F C I ビットにより 32 シンボルの T F C I 符号語を出力するため所定の符号化テーブルを有する。その一例は図 4 B に示されている。前記図 4 B のように、符号化テーブルには 32 ビットの長さを有する 32 個の直交符号語 $c_{32,1}$ 乃至 $c_{32,32}$ と、前記 32 個の符号語 $c_{32,1}$ 乃至 $c_{32,32}$ の

10

【数 1】

補数 $C_{32.1}$ 乃至 $C_{32.32}$

に該当する 32 個の相互直交符号語が登録されている。従って、前記相互直交符号器 402 は基本型 T F C I ビットの L S B (Least significant bit、最下位ビット)が 1 であると、32 個の相互直交符号語を選択し、前記 L S B が 0 であると、32 個の直交符号語を選択する。前記 L S B により直交符号、または相互直交符号が選択されると、前記 L S B を除外したその以外の T F C I ビットにより前記選択された直交符号、または相互直交符号のいずれか一つを選択して T F C I 符号語に出力する。

20

【0012】

一方、T F C I 符号語は上述したように誤り訂正能力を有すべきであるが、一般的に 2 進線形符号(Binary Linear Codes)の誤り訂正能力は 2 進線形符号間の最小距離(dmin:Minimum Distance)によって決定される。最適符号(optimal code)になるための 2 進線形符号間の最小距離(dmin:Minimum Distance)は下記参考文献[1]で開示している。

【0013】

参考文献[1]: An Updated Table of Minimum-Distance Bounds for Binary Linear Codes(A.E. Brouwer and Tom Verhoeff, IEEE Transactions on Information Theory, Vol. 39, No.2, March 1993)

【0014】

前記参考文献[1]で 6 ビットの入力により 32 ビットが出力される 2 進線形符号で要求される最小距離(dmin:Minimum Distance)は 16 に提案している。このような観点からみる時、前記相互直交符号器 402 の出力である T F C I 符号語は 16 の最小距離(dmin:Minimum Distance)を有する。従って、前記相互直交符号器 402 から出力される T F C I 符号語は最適符号といえることができる。

30

【0015】

一方、図 5 A は従来の符号分割多重接続移動通信システムで拡張型 T F C I ビットを符号化するための構成を概念的に示しており、図 5 B は図 5 A の制御器が伝送率情報を分配するため遂行するアルゴリズムの一例を示している。また、図 5 C は図 5 A の相互直交符号器に適用される符号化テーブルの一例を示している。拡張型 T F C I は T F C I ビット数により定義されることができる。即ち、前記拡張型 T F C I は 1 乃至 128、1 乃至 256、1 乃至 512 及び 1 乃至 1024 の相異なる情報を指定する 7 ビット乃至 10 ビットの T F C I ビット(以下、拡張型 T F C I ビット)に構成される。

40

【0016】

前記図 5 A 乃至図 5 C を参照すると、拡張型 T F C I ビットは制御器 500 に入力され所定アルゴリズムにより両分され出力される。例えば、10 ビットの拡張型 T F C I ビットが入力されると、前記制御器 500 は前記 10 ビットの拡張型 T F C I ビット中の 5 ビットを第 1 T F C I ビット(word1)に出力し、その他の 5 ビットを第 2 T F C I ビット(word2)に出力する。前記拡張型 T F C I ビットは上述したように 10 ビット表現形式を基本にしている。従って、前記制御器 500 は 10 ビットより小さいビット数を有する拡張

50

型 T F C I ビットが入力される場合には、前記入力された拡張型 T F C I ビットの M S B から 0 の値を入れて 1 0 ビット表現形式を有するようにする。その後、前記制御器 5 0 0 は前記 1 0 ビット表現形式の拡張型 T F C I ビットを上述した動作により word 1 と word 2 に分割して出力する。

【0017】

前記 5 ビットに分割された word 1 と word 2 はそれぞれ相互直交符号器 5 0 2、5 0 4 に入力される。前記制御器 5 0 0 で入力される拡張型 T F C I ビット (a_1 乃至 a_{10}) を word 1 と word 2 に分割して出力する方法の一例は図 5 B のようである。

【0018】

前記相互直交符号器 5 0 2 は前記制御器 5 0 0 から受信される word 1 を符号化して長さ 1 6 シンボルの第 1 T F C I 符号語を出力し、前記相互直交符号器 5 0 4 は前記制御器 5 0 0 から受信される word 2 を符号化して長さ 1 6 のシンボルの第 2 T F C I 符号語を出力する。一方、前記相互直交符号器 5 0 2 と 5 0 4 は入力される 5 ビットの T F C I ビット (word 1、word 2) により 1 6 シンボルの T F C I 符号語を出力するため所定の符号化テーブルを有する。その例は図 5 に示されている。前記図 5 C のように符号化テーブルには 1 6 ビットの長さを有する 1 6 個の直交符号語 ($c_{16,1}$ 乃至 $c_{16,16}$) と前記 1 6 個の直交符号語の

【数 2】

補数($c_{16,1}$ 乃至 $c_{16,16}$)

20

に構成された相互直交符号語が登録されている。従って、前記相互直交符号器 5 0 2 と 5 0 4 は入力される 5 ビットの T F C I ビットの L S B (Least significant bit、最下位ビット) が 1 であると、1 6 個の相互直交符号語を選択し、前記 L S B が 0 であると、1 6 個の直交符号語を選択する。その後、前記相互直交符号器は、前記 L S B を除外したその他の T F C I ビットに基づいて前記選択された直交符号、または相互直交符号中のいずれか一つを選択して、第 1、または第 2 T F C I 符号語に出力する。

【0019】

前記相互直交符号器 5 0 2 から出力される 1 6 シンボルの第 1 T F C I 符号語と前記相互直交符号器 5 0 4 から出力される 1 6 シンボルの第 2 T F C I 符号語はマルチプレクサ 5 1 0 に提供される。前記マルチプレクサ 5 1 0 は前記第 1 T F C I 符号語と前記第 2 T F C I 符号語をマルチプレクシングして 3 2 シンボルの T F C I 符号語を出力する。

【0020】

受信側では前記 3 2 シンボル(またはビット)の T F C I 符号語を受信して 1 6 シンボル(またはビット)ずつ分けて、それぞれ 5 ビットの T F C I ビット (word 1、word 2) に復号した後、これを合わせて 1 0 ビットの正確な T F C I ビットを得る。従って、上述した復号過程で復号された 5 ビットの T F C I ビット (word 1、または word 2) の中の一つでも誤りが発生する時には 1 0 ビットの全体 T F C I ビットに誤りが発生する。

【0021】

一方、拡張型 T F C I 符号語も上述したように誤り訂正能力を有すべきである。このためには前記拡張型 T F C I 符号語は前記参考文献 [1] で提案している最小距離を有すべきである。

【0022】

送信側での T F C I ビットが 1 0 ビットであり、前記 T F C I ビットを符号化した T F C I 符号語が 3 2 シンボル(またはビット)であることを考慮すると、前記参考文献 [1] では最適符号のための最小距離を 1 2 に提案している。しかし、前記図 5 A のような構造から出力される T F C I 符号語は 8 の最小距離 (dmin: Minimum Distance) を有する。その理由は上述したように復号過程で復号された 5 ビットの T F C I ビット (word 1、または word 2) の一つでも誤りが発生する時には、1 0 ビットの全体 T F C I ビットに誤りが発生するためである。即ち、T F C I ビットが 5 ビットずつ分割され符号化されるとしても、最終出力される T F C I 符号語間の最小距離は相互直交符号器 5 0 2 と 5 0 4 の符号語間の

50

最小距離である 8 になる。

【0023】

従って、前記図 5 A に示した符号化構造から伝送される T F C I 符号語は最適符号を有することができなく、これは同一の無線チャネル環境で T F C I ビットの誤りが発生する確率が高くなる原因になる。一方、前記 T F C I ビットの誤り発生確率が高くなると、受信側で受信されたデータフレームの伝送率を誤判断するようになり、これはデータフレームの復号時、エラー率を増加させるようになって、符号分割多重接続移動通信システムの効率を低下させる問題をもたらした。

【0024】

また、前記従来技術によると、基本型 T F C I と拡張型 T F C I をサービスするために 10
は基本型 T F C I をサービスするためのハードウェア構成と前記拡張型 T F C I をサービスするためのハードウェア構成を別に設けるべきであった。これは符号分割多重接続移動通信システムを具現することにおいて、費用とシステムの大きさにおいて不合理な問題を発生させた。

【発明の開示】

【発明が解決しようとする課題】

【0025】

本発明は上述した従来の問題点を解決するためのもので、本発明の目的は符号分割多重 20
接続移動通信システムで、拡張型伝送率情報 (T F C I) を符号化する符号化装置及び方法を提供することにある。

【0026】

本発明の他の目的は符号分割多重接続移動通信システムで、拡張型伝送率情報 (T F C I) の符号化と共に、基本型伝送率情報 (T F C I) の符号化に対して互換性を有する符号 20
化装置及び方法を提供することにある。

【0027】

本発明のさらに他の目的は符号分割多重接続移動通信システムで、拡張型伝送率情報 (T F C I) を復号化する復号化装置及び方法を提供することにある。

【0028】

本発明のさらに他の目的は符号分割多重接続移動通信システムで、拡張型伝送率情報 (T F C I) の復号化と、基本型伝送率情報 (T F C I) の復号化に対して互換性を有する復 30
号化装置及び方法を提供することにある。

【0029】

本発明のさらに他の目的は符号分割多重接続移動通信システムで、拡張型伝送率情報 (T F C I) を符号化することによって、最適符号を生成することができる符号化装置及び 30
方法を提供することにある。

【0030】

本発明のさらに他の目的は符号分割多重接続移動通信システムで、拡張型伝送率情報 (T F C I) を符号化及び復号化するため使用されるマスクシーケンスを生成する方法を提 40
供することにある。

【課題を解決するための手段】

【0031】

前記目的を達成するために本発明は、符号分割多重接続移動通信システムで T F C I 符 50
号化／復号化装置及び方法を提供する。前記 T F C I 符号化装置は、同一シンボルのシーケンスを発生する 1 ビット発生器と、複数の基底直交シーケンスを発生する基底直交シーケンス発生器と、複数の基底マスクシーケンスを発生する基底マスクシーケンス発生器と、相互直交シーケンスへの変換を示す第 1 情報部分と、直交シーケンスへの変換を示す第 2 情報部分と、マスクシーケンスへの変換を示す第 3 情報部分とを含む伝送率情報ビットを受信し、前記基底直交シーケンスの中、前記第 2 情報部分により前記基底直交シーケンスから選択される直交シーケンスと、前記選択される直交シーケンスと前記第 1 情報部分により選択される前記同一シンボルの組み合わせに構成される相互直交シーケンスと、前

記相互直交シーケンスと前記第 3 情報部分により選択されるマスクシーケンスとを組み合わせ、伝送率情報シーケンスに発生する演算部と、を含む。

【発明を実施するための最良の形態】

【0032】

以下、本発明の望ましい実施形態を添付図を参照しつつ詳細に説明する。下記の発明において、本発明の要旨のみを明瞭にする目的で、関連した公知機能又は構成に関する具体的な説明は省略する。

【0033】

先ず、本発明では第 1 T F C I ビットにより出力される第 1 符号化シンボル(第 1 T F C I 符号語)と第 2 T F C I ビットにより出力される第 2 符号化シンボル(第 2 T F C I 符号語)を加算して完全な符号化シンボル(T F C I 符号語)を出力する符号分割多重接続移動通信システムでの伝送率情報(T F C I)符号化概念を提案している。前記 T F C I 符号化概念は図 6 に示されている。ここで、相互直交シーケンス及びマスクシーケンスは、第 1 T F C I 符号語及び第 2 T F C I 符号語をそれぞれ開示している。

10

【0034】

前記図 6 を参照すると、伝送率情報(T F C I)ビットは第 1 T F C I ビットと第 2 T F C I ビットに分割され、マスクシーケンス発生部 602 と相互直交シーケンス発生部 604 にそれぞれ入力される。前記マスクシーケンス発生部 602 は受信された第 2 T F C I ビットを符号化して所定マスクシーケンスを生成する。前記相互直交シーケンス発生部 604 は受信された第 1 T F C I ビットを符号化して所定相互直交シーケンスを生成する。加算器 610 は前記マスクシーケンスと前記相互直交シーケンスを加算して、前記 T F C I ビットが符号化された完全な符号化シンボル(T F C I 符号語)を出力する。この時、前記マスクシーケンス発生部 602 は入力される第 2 T F C I ビットのすべての場合数に対応してマスクシーケンスを符号化テーブルに有していることができる。前記相互直交シーケンス発生部 604 も入力される第 1 T F C I ビットのすべての場合数に対応して相互直交シーケンスを符号化テーブルに有していることができる。

20

【0035】

上述したように本発明の実施のためにはマスクシーケンス及びマスクシーケンス生成方法が定義されるべきである。一方、本発明の実施形態では前記直交シーケンスの一例にウォルシュ符号を開示している。

30

【0036】

1. マスクシーケンス生成方法

本発明は上述したように、I M T システムで T F C I ビットを符号化及び復号化する技術に関するもので、拡張されたリードミュラー(Extended Reed Muller)コードを使用するものである。このためには所定シーケンスが使用されるが、この時のシーケンスは誤り訂正性能が優秀な最小距離を有すべきである。

【0037】

通常的に線形誤り訂正符号(Linear Error Correcting Code)の性能、または能力を示す重要な尺度(measure)、即ちパラメータは誤り訂正符号の符号語(code word)間の最小距離(minimum distance)である。その符号語のハミングウェイト(Hamming weight)は“0”ではないシンボルの個数を意味する。即ち、符号語が“0111”であると、ハミングウェイトは 3 である。各コードに対して“0”ではない符号語の一番小さいハミングウェイトは最小ウェイト(minimum weight)と言われ、各 2 進線形コードの前記最小距離は前記最小ウェイトと同一である。前記線形誤り訂正符号(Linear Error Correcting Code)で最小距離が大きいほど誤り訂正性能が優秀である。これは参考文献 2 “The Theory of Error-Correcting Codes” -F.J.Macwilliams, and N.J.A. Sloane, North-Hollandで詳細に開示している。

40

【0038】

前記拡張されたリードミュラーコードは任意のシーケンスと m シーケンスの和になされるシーケンスの集合であるシーケンス群から類推することができる。前記シーケンス群を

50

線形誤り訂正符号に使用するためには、前記シーケンス群の最小距離が大きくなければならない。このようなシーケンス群はカサミシーケンス(Kasami sequence)群、ゴールドシーケンス(Gold sequence)群、カドックシーケンス(Kerdock sequence)群を含む。前記シーケンスは全体長さ $L = 2^{2^m}$ である時、最小距離が $(2^{2^m} - 2^m)/2$ であり、 $L = 2^{2^m + 1}$ である時、最小距離が $(2^{2^m + 1} - 2^{2^m})/2$ である。即ち、全体長さが $3 \cdot 2$ である時、最小距離は $1 \cdot 2$ である。

【0039】

以下、本発明では前記のようなシーケンス群を使用して優秀な性能を有する線形誤り訂正符号である拡張された誤り訂正符号(ウォルシュ符号及びマスクシーケンス)を生成する方法に対して説明する。

10

【0040】

コード理論(Coding Theory)によると、元の m シーケンスを一回から n 回循環することにより形成されるグループの m シーケンスからウォルシュ符号を形成する列置換関数は存在する。ここで n は m シーケンスの長さである。言い換えれば、各 m シーケンスは元の m シーケンスを所定回数循環させることにより形成される。前記列置換関数は前記 m シーケンス内のシーケンスをウォルシュ符号に置換するための関数である。ここで、元の m シーケンスと他の元の m シーケンスを加算することにより形成させるシーケンスにはゴールドシーケンス、またはカサミシーケンスがあると仮定する。 m シーケンスのさらに他のグループは他の元の m シーケンスを一回から n 回循環させることにより類似に形成される。ここで、 n は予め決定されたシーケンスの長さである。その後、逆列置換関数は他の元の m シーケンスから形成された m シーケンスの第2グループに印加される。前記逆列置換関数の m シーケンスの第2グループへの印加はマスクシーケンスと定義できる他のシーケンス群を生成する。

20

【0041】

本実施形態ではゴールドシーケンス群を使用して、 $(2^n, n+k)$ コード(拡張されたリードミュラーコード)(ただ、 $k = 1, \dots, n+1$)を生成する場合、上述したマスクシーケンスの生成方法を説明する。前記 $(2^n, n+k)$ コードとは $n+k$ 個の T F C I ビット(入力情報ビット)を入力として 2^n 個の符号化された T F C I 符号語(符号化シンボル)を出力するためのコードを意味する。実際に、ゴールドシーケンスは二つの相異なる特定 m シーケンスの和に表現されることは公知の事実である。従って、前記 $(2^n, n+k)$ コードを生成するためには、まず長さ $2^n - 1$ であるゴールドシーケンスを生成すべきである。ここで、生成多項式 $f_1(x)$ と $f_2(x)$ から生成される二つの m シーケンス $m_1(t)$ と $m_2(t)$ の和がゴールドシーケンスになる。この時、 m シーケンス $m(t)$ 、即ち $m_1(t)$ と $m_2(t)$ は、生成多項式 $f_1(x)$ と $f_2(x)$ が定まると、下記式(1)のようにトレイス関数(Trace function)を利用して求めることができる。

30

【0042】

【数3】

$$m_1(t) = \text{Tr}(A\alpha^t) \quad t = 0, 1, \dots, 30$$

$$\text{Tr}(a) = \sum_{k=0}^{n-1} \alpha^{2^k}, \quad a \in \text{GF}(2^n) \quad \dots\dots (1)$$

40

【0043】

前記式(1)で A は m シーケンスの初期値によって決定される値、 α は生成多項式の根、 n は生成多項式の次数を示す。

【0044】

図7は前記シーケンス群の中、ゴールドシーケンス群を使用して、 $(2^n, n+k)$ コード(即ち、 $n+k$ ビットの情報ビットを入力して 2^n ビットの符号化シンボルを出力するためのコード)を生成する場合に使用されるマスクシーケンスの生成過程を示す。

【0045】

50

前記図7を参照してマスクシーケンスの生成過程を説明すると、710段階では前記式(1)により生成多項式 $f_1(x)$ と生成多項式 $f_2(x)$ を使用して m シーケンス $m_1(t)$ と m シーケンス $m_2(t)$ を求める。すると、712段階では前記 m シーケンス $m_2(t)$ を0から $n-2$ 回循環させ形成された m シーケンスを有するシーケンス群からウォルシュ符号を形成するためにシーケンス列置換関数 $\sigma(t)$ を求める。ここで、下記のように $m_2(t)$ から形成された前記 m シーケンスの先にすべて0の列を挿入する。

【0046】

【数4】

$$\sigma: \{0, 1, 2, \dots, 2^n - 2\} \rightarrow \{1, 2, 3, \dots, 2^n - 1\}$$

$$\sigma(t) = \sum_{i=0}^{n-1} m_2(t+i)2^{n-1-i} \quad t = 0, 1, 2, \dots \quad \dots\dots (2)$$

10

【0047】

すると、730段階では前記 m シーケンス $m_1(t)$ を0から30回まで循環(cyclically shifting)させ得られる31個のシーケンス群を前記 $m_2(t)$ をウォルシュ符号に形成する列置換関数 $\sigma(t)$ の逆関数を利用した $\sigma^{-1}(t)+2$ に列置換する。そして、前記列置換により形成されたそれぞれのシーケンスのスタート部分に“0”を加えることにより長さ 2^n に形成して長さ 2^n である $2^n - 1$ 個のシーケンス群 $d_i(t)$ ($i = 0, \dots, 2^n - 2, t = 1, \dots, 2^n$)を生成する。前記のように730段階で生成されるシーケンス群は下記式(3)のように表現することができる。

20

【0048】

【数5】

$$\{d_i(t) | t = 1, \dots, 2^n, \quad i = 0, \dots, 2^n - 2\}$$

$$d_i(t) = \begin{pmatrix} 0, & \text{if, } t=1 \\ m_1(\sigma^{-1}(t+i)+2), & \text{if, } t=2,3,\dots,2^n \end{pmatrix} \quad \dots\dots (3)$$

【0049】

30

複数の $d_i(t)$ は前記マスク関数であり、31個のマスクに使用することができる。

$d_i(t)$ の性質の中、一つの性質は前記マスクの中、2個の相異なるマスクを加えると、前記二つのマスクを除いた $2^n - 1$ 個のマスク中一つのマスクになるものである。より一般化させると、前記 $2^n - 1$ 個のマスクのそれぞれは、特定な n 個のマスクの少なくとも二つ以上の和に表現されることができる。前記特定 n 個のマスクを基底マスクシーケンスと定義する。

【0050】

前記(2^n 、 $n+k$)コード(拡張されたリードミュラーコード)を生成する時、必要とする符号語(Code word)の個数は入力情報ビット(TFCIビット) $n+k$ に対して 2^{n+k} 個である。 2^n 個の直交シーケンス(ウォルシュシーケンス)とその補数を示す相互直交シーケンス(Biorthogonal sequence)の個数は $2^n \times 2 = 2^{n+1}$ であり、(2^n 、 $n+k$)コードを生成するため必要な“0”ではないマスクの個数は $(2^{n+k} / 2^{n+1}) - 1 = 2^{k-1} - 1$ 個である。また、この時、 $2^{k-1} - 1$ 個のマスクは、上述したように $k-1$ 個の基底マスクシーケンスにより表現されることができる。

40

【0051】

以下、前記 $k-1$ 個の基底マスクシーケンスを選択する方法を説明する。前記図7の730段階で $m_1(t)$ を0から $2^n - 1$ 回まで循環(cyclically shift)させシーケンス群を生成するが、この時、 $m_1(t)$ を i 回循環(cyclic shift)させた m シーケンスは前記式(1)を使用して表現すると $\text{Tr}(\alpha^i \cdot \alpha^t)$ になる。即ち、 $m_1(t)$ を初期値 $A = 1$ 、 α 、 $\dots$ 、 $\alpha^{2^n - 2}$ について0から30回まで循環(cyclic shift)させることにより生成

50

されるシーケンス群である、この時、ガロアの元素 $1, \alpha, \dots, \alpha^{2^n-2}$ の中、線形独立である $k-1$ 個の基底元素を探し、前記 $k-1$ 個の基底元素を初期値にするトレイス関数の出力シーケンスに対応するマスクシーケンスが基底マスクシーケンスになる。前記過程の中、線形独立の条件を式に示すと式 (4) のようである。

【0052】

【数6】

$\alpha_1, \dots, \alpha_{k-1}$: 線形独立

$$\Leftrightarrow c_1\alpha_1 + c_2\alpha_2 + \dots + c_{k-1}\alpha_{k-1} \neq 0, \quad \forall c_1, c_2, \dots, c_{k-1} \quad \dots\dots (4)$$

10

【0053】

上述した一般化されたマスク関数生成方法を具体的に説明するために、下記では前記図7を参照してゴールドシーケンス群を使用して、(32、10)コードを生成する場合を一例に説明する。実際に、ゴールドシーケンスは相異なる特定の m シーケンスの和に表現されることは公知の事実である。従って、得ようとする(32、10)コードを生成するためには、先ず、長さ31であるゴールドシーケンスを生成すべきである。前記ゴールドシーケンスは生成多項式 $x^5 + x^2 + 1$ と $x^5 + x^4 + x + 1$ からそれぞれ生成される二つの m シーケンスの和である。また、 m シーケンス $m_1(t)$ と $m_2(t)$ のそれぞれは、対応する生成多項式が定まると、下記の式 (5) のようにトレイス関数を利用して求めることができる。

20

【0054】

【数7】

$$m_1(t) = \text{Tr}(A\alpha^t) \quad t = 0, 1, \dots, 30$$

$$\text{Tr}(a) = \sum_{n=0}^4 \alpha^{2^n}, \quad a \in \text{GF}(2^5) \quad \dots\dots (5)$$

【0055】

前記式 (5) で A は m シーケンスの初期値によって決定される値であり、 α は生成多項式の根、 n は生成多項式の次数である。ここで、 n は“5”である。

30

【0056】

図7は前記シーケンス群の中、ゴールドシーケンス群を使用して、(32、10)コード(即ち、10ビットの入力情報ビット(TFCIビット)を入力して32ビットの符号化シンボル(TFCI符号語)を出力するためのコード)を生成する場合、前記のようなマスク関数の生成過程を示す。

【0057】

前記図7を説明すると、710段階では前記式 (1) により前記生成多項式 $f_1(x)$ と $f_2(x)$ を使用して m シーケンス $m_1(t)$ と $m_2(t)$ をそれぞれ生成する。すると、712段階では前記 m シーケンス $m_2(t)$ を下記式 (6) に適用して前記 $m_2(t)$ をウォルシュ符号に形成する列置換関数 $\sigma(t)$ を求める。

40

【0058】

【数8】

$$\sigma: \{0, 1, 2, \dots, 30\} \rightarrow \{1, 2, 3, \dots, 31\}$$

$$\sigma(t) = \sum_{i=0}^4 m_2(t-i)2^{4-i} \quad \dots\dots (6)$$

【0059】

すると、730段階では前記 m シーケンス $m_1(t)$ を0から30回まで循環(cyclic shift)させ得られる31個のシーケンス群を前記式 (6) により求められた列置換関数 σ (

50

t)の逆関数を利用した“ $\sigma^{-1}(t)+2$ ”に列置換する。そして、前記列置換されたそれぞれのシーケンスのスタート部分に0を加えることにより、長さ32である31個のシーケンス群 $d_i(t)$ を生成する。ここで、 $i=0, \dots, 31$ であり、 $t=1, \dots, 32$ である。前記のように730段階で生成されるシーケンス群は下記式(7)のように表現することができる。

【0060】

【数9】

$$\{d_i(t) | t=1, \dots, 32, i=0, \dots, 30\}$$

$$d_i(t) = \begin{cases} 0, & \text{if, } t=1 \\ m_1(\sigma^{-1}(t+i)+2), & \text{if, } t=2,3,\dots,32 \end{cases} \quad \dots\dots (7)$$

10

【0061】

前記式(7)により求められた複数の $d_i(t)$ は31個のマスクシーケンスに使用することができる。

【0062】

$d_i(t)$ の性質中の一つは、前記マスクシーケンスの中、二つの相異なるマスクシーケンスを加えると、前記二つのマスクを除いた31個のマスク中の一つになるものである。言い換えれば、前記31個のマスクシーケンスは31個のマスクシーケンスの中、特定な5個のマスクシーケンスの和に表現されることができる。前記5個のマスクシーケンスは基底マスクシーケンスである。

20

【0063】

前記(32、10)コードを生成する時、必要な符号語(Code word)の総数はすべての場合の入力情報ビット(TFCIビット)の数である $2^{10} = 1024$ 個である。この時、長さ32である相互直交(Biorthogonal)シーケンスの個数は $32 \times 2 = 64$ であり、この時、(32、10)コードを生成するため必要なマスクシーケンスの個数は $(1024 / 64) - 1 = 15$ 個である。また、この時、15個のマスクシーケンスは4個の基底マスクシーケンスの組み合わせに表現されることができる。従って、前記4個の基底マスクシーケンスを選択する方法が必要である。

30

【0064】

前記4個の基底マスクシーケンスを選択する方法を説明すると、前記図7の730段階で $m_1(t)$ を0から30回まで循環(cyclic shift)させシーケンス群を生成するが、この時、 $m_1(t)$ を i 循環(cyclic shift)させた m シーケンスは前記式(1)を使用して表現すると $\text{Tr}(\alpha^i \cdot \alpha^t)$ になる。即ち、前記 $m_1(t)$ を0から30回まで循環(cyclic shift)させ生成されたシーケンス群は初期値 A が1、 α 、 $\dots$ 、 α^{30} である場合、それぞれに対応して生成されるシーケンスである。この時、ガロア(Galois)の元素1、 α 、 $\dots$ 、 α^{30} の中、線形独立である4個の基底元素を探し、前記4個の基底元素を初期値にするトレリス関数の出力シーケンスに対応するマスクシーケンスが基底マスクシーケンスになる。この、線形独立の条件を式に示すと数(8)のようである。

40

【0065】

【数10】

$\alpha, \beta, \gamma, \delta$: 線形独立

$$\Leftrightarrow c_1\alpha + c_2\beta + c_3\gamma + c_4\delta \neq 0, \quad \forall c_1, c_2, c_3, c_4 \quad \dots\dots (8)$$

【0066】

実際、ガロア $GF(2^5)$ で1、 α 、 α^2 、 α^3 は前記のような4個の線形独立元素に知らせた基底多項式(polynomial basis)である。従って、前記基底多項式を前記式(1)の変数 A の代わり代入することによって、求められた4個の基底マスクシーケンス $M1$ 、 M

50

2、M4、M8は次のようである。

【0067】

M1 = 00101000011000111111000001110111

M2 = 00000001110011010110110111000111

M4 = 00001010111110010001101100101011

M8 = 00011100001101110010111101010001

【0068】

以下、本発明の実施形態による符号分割多重接続移動通信システムで、上述した方式により求められた基底マスクシーケンスを利用してTFCIを符号化／復号化する装置及び方法に対して説明する。

10

【0069】

2. 符号化／復号化装置及び方法の第1実施形態

図8及び図9は本発明の一実施形態による符号分割多重接続移動通信システムで、伝送率情報を符号化及び復号化するための構成をそれぞれ示している。

【0070】

先ず、図8を参照して本発明の一実施形態による符号化装置を説明すると、10ビットの入力情報ビット(TFCIビット)が入力されると、それぞれのビットa0、a1、a2、a3、a4、a5、a6、a7、a8、a9は対応する乗算器840、841、842、843、844、845、846、847、848、849にそれぞれ入力される。1ビット発生器800は持続的に所定の符号ビットを発生する。即ち、本発明では相互直交シーケンスを対象にしているので、直交シーケンスを相互直交シーケンスに使用するため要求されるビットを発生する。例えば、前記1ビット発生器800は持続的に“1”の値を有するビットを発生することにより、基底ウォルシュ符号発生器810から発生される直交シーケンス(ウォルシュ符号)が反転され相互直交シーケンスが発生するようにする。前記基底ウォルシュ符号発生器810は所定長さを有する基底ウォルシュ符号を発生する。前記基底ウォルシュ符号は任意の和により使用しようとするウォルシュ符号をすべて生成することができるウォルシュ符号を意味する。例えば、長さ32のウォルシュ符号を使用する場合、基底ウォルシュ符号は第1ウォルシュ符号(W1)、第2ウォルシュ符号(W2)、第4ウォルシュ符号(W4)、第8ウォルシュ符号(W8)、第16ウォルシュ符号(W16)になる。前記W1、W2、W3、W4は下記のようなものである。

20

30

【0071】

W1: 01010101010101010101010101010101

W2: 00110011001100110011001100110011

W4: 00001111000011110000111100001111

W8: 00000000111111110000000011111111

W16: 00000000000000001111111111111111.

【0072】

基底マスクシーケンス発生器820は所定長さを有する基底マスクシーケンスを発生する。前記基底マスクシーケンス発生方法は上述したので詳細な説明は省略する。例えば、長さ32のマスクシーケンスを使用する場合、基底マスクシーケンスは第1マスクシーケンス(M1)、第2マスクシーケンス(M2)、第4マスクシーケンス(M4)、第8マスクシーケンス(M8)になる。前記M1、M2、M4、M8は下記のようなものである。

40

【0073】

M1: 00101000011000111111000001110111

M2: 00000001110011010110110111000111

M4: 00001010111110010001101100101011

M8: 00011100001101110010111101010001

【0074】

前記1ビット発生器800から出力されるシンボル1は前記乗算器840に入力され前記入力情報ビット中a0とシンボル単位にかけられて出力される。

50

“ 1 ”を前記基底マスクシーケンス発生器 8 2 0 から発生される M 1 とシンボル単位にかけて “ 0 0 1 0 1 0 0 0 0 1 1 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 0 1 1 1 ” の符号化シンボルを出力する。乗算器 8 4 7 は a 7 に該当する “ 0 ” を前記基底マスクシーケンス発生器 8 2 0 から発生される M 2 とシンボル単位にかけて長さ 3 2 であるすべてのシンボルが “ 0 ” である符号化シンボルを出力する。乗算器 8 4 8 は a 8 に該当する “ 0 ” を前記基底マスクシーケンス発生器 8 2 0 から発生される M 4 とシンボル単位にかけて長さ 3 2 であるすべてのシンボルが “ 0 ” である符号化シンボルを出力する。乗算器 8 4 9 は a 9 に該当する “ 0 ” を前記基底マスクシーケンス発生器 8 2 0 から発生される M 8 とシンボル単位にかけて長さ 3 2 であるすべてのシンボルが “ 0 ” である符号化シンボルを出力する。一方、前記加算器 8 6 0 は乗算器 8 4 0 乃至 8 4 9 から受信される符号化シンボルを加算して “ 0 1 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 1 1 0 0 1 1 0 1 1 1 0 0 0 0 1 ” の最終符号化シンボルを出力する。前記最終符号化シンボルは “ 1 ” を有する入力情報ビットに対応する基底ウォルシュ符号 (W 1、W 2、W 4、W 1 6) と基底マスクシーケンス (M 1) をシンボル単位に加算した結果である。即ち、基底ウォルシュ符号 W 1、W 2、W 4、W 1 6 を加算することにより W 2 3 が生成され、前記生成された W 2 3 に M 1 を加算した形態 (W 2 3 + M 1) の T F C I 符号語 (符号化シンボル) が前記加算器 8 6 0 の最終出力になる。

【0080】

図 1 1 は本発明の一実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するための制御流れを示している図である。

【0081】

前記図 1 1 を参照すると、先ず、1 1 0 0 段階で 1 0 ビットの入力情報ビット (T F C I ビット) を受信し、変数 s u m と j を 0 に初期化する。前記変数 s u m はシンボル単位に加算され出力される最終符号化シンボルを示す変数であり、前記変数 j はシンボル単位への加算により出力される最終符号化シンボルの個数をカウントするための変数である。1 1 1 0 段階で、変数 j が 3 2 であるかを判断する。前記変数 j が 3 2 であるかを判断することは前記入力情報ビットの符号化のため使用されるウォルシュ符号及びマスクシーケンスの長さが 3 2 シンボルであるためである。即ち、1 1 1 0 段階は前記入力情報ビットがウォルシュ符号及びマスクシーケンスの 3 2 シンボルそれぞれに対応して符号化されたかを判断するために遂行される。

【0082】

前記 1 1 1 0 段階で j が 3 2 ではないと、ウォルシュ符号及びマスクシーケンスのすべてのシンボルに対応した符号化動作が遂行されなかったと判断して 1 1 2 0 段階に進行する。前記 1 1 2 0 段階に進行すると、基底ウォルシュ符号 W 1、W 2、W 4、W 8、W 1 6 の j 番目シンボルである W 1 (j)、W 2 (j)、W 4 (j)、W 8 (j)、W 1 6 (j) と基底マスクシーケンス M 1、M 2、M 4、M 8 の j 番目シンボルである M 1 (j)、M 2 (j)、M 4 (j)、M 8 (j) を入力する。上述したような入力が遂行されると、1 1 3 0 段階では前記入力されたシンボルを前記入力情報ビット (T F C I ビット) とシンボル単位にかけた後、その結果値を全部合わせて s u m を求める。

【0083】

前記 1 1 3 0 段階でなされる動作を式に表現すると、下記式 (9) のように示すことができる。

【0084】

【数 1 1】

$$\begin{aligned} \text{s u m} = & a_0 + a_1 \cdot W_1(j) + a_2 \cdot W_2(j) + a_3 \cdot W_4(j) + a_4 \cdot W_8(j) \\ & + a_5 \cdot W_{16}(j) + a_6 \cdot M_1(j) + a_7 \cdot M_2(j) + a_8 \cdot M_4(j) + a_9 \cdot M_8(j) \\ & \dots\dots (9) \end{aligned}$$

【0085】

前記式(9)から分かるように、各入力情報ビットは対応する基底ウォルシュ符号、または基底マスクシーケンスのシンボルとかけられ、前記かけられた結果を加算することにより所望する符号化シンボルを得るようになる。

【0086】

1140段階では前記1130段階で求められたj番目符号化シンボルであるsumを出力し、1150段階に進行してjを1だけ増加させた後、さらに1110段階を進行して上述した動作を反復して遂行する。一方、上述した動作の反復遂行を通じて前記1110段階でjが32であると、すべての符号化動作を完了するようになる。

【0087】

一方、図8を参照して説明した本発明の一実施形態による符号化装置は、基本型TFCIだけでなく、すべての拡張型TFCIも支援することができる。前記拡張型TFCIを支援するための符号器には(32、10)符号器、(32、9)符号器、(32、8)符号器、(32、7)符号器がある。 10

【0088】

前記(32、10)符号器の出力は、10ビットの入力情報ビットを入力として長さ32である32個のウォルシュ符号と前記ウォルシュ符号それぞれを反転した32個の相互直交符号と15種類のマスクシーケンスの組み合わせに示される。前記32個のウォルシュ符号は5個の基底ウォルシュ符号の組み合わせから生成され得るすべてのウォルシュ符号を意味する。前記32個の相互ウォルシュ符号は前記32個のウォルシュ符号それぞれを構成する32シンボルに1を加算することにより求められる。これは前記32個のウォルシュ符号を実数に仮定した時、-1をかけることと同一の効果を有する。前記15個のマスクシーケンスは5個の基底マスクシーケンスの組み合わせに求められる。従って、前記(32、10)符号器から得られる総符号語の数は1024個である。 20

【0089】

前記(32、9)符号器の出力は、9ビットの入力情報ビットを受信し、長さ32である32個のウォルシュ符号と前記ウォルシュ符号それぞれを反転した32個の相互直交符号と4種類のマスクシーケンスの組み合わせに示される。前記4種類のマスクシーケンスは4個の基底マスクシーケンス中、2個の基底マスクシーケンスの組み合わせに求められる。

【0090】 30

前記(32、7)符号器の出力は、7ビットの入力情報ビットを受信し、前記1024個の符号語中、長さ32である32個のウォルシュ符号と、前記ウォルシュ符号それぞれを反転した32個の相互直交符号と、4個の基底マスクシーケンス中、1個の基底マスクシーケンスの組み合わせに示される。

【0091】

一方、拡張型TFCIをサービスするため上述した符号器はすべて最小距離12を有すべきであり、前記基底マスクシーケンス発生器820からの4種類の基底マスクシーケンス中、少なくとも一つの基底マスクシーケンスに対する入力と出力を遮断することにより具現されることができる。

【0092】 40

即ち、(32、9)符号器は前記図8の基底マスクシーケンス発生器820からの4種類基底マスクシーケンス中、いずれか一つの基底マスクシーケンスに対する入力と出力を遮断することにより具現されることができる。(32、8)符号器は前記基底マスクシーケンス発生器820からの4種類基底マスクシーケンス中、二つの基底マスクシーケンスに対する入力と出力を遮断することにより具現されることができる。また、(32、7)符号器は前記基底マスクシーケンス発生器820からの4種類基底マスクシーケンス中、三つの基底マスクシーケンスに対する入力と出力を遮断させることにより具現されることができる。上述したように本発明の一実施形態による符号器は入力情報ビット、即ち伝送しようとするTFCIのビット数に従って流動的に符号化ができ、符号器の性能を左右する最小距離を最大限高めることにより優秀な性能を有するようになる。 50

【0093】

前記符号器は長さ32である32個のウォルシュ符号と前記ウォルシュ符号にすべて1を加えた32個の相互直交符号になされる総64個の符号に、長さ15である15個のマスクシーケンスを組み合わせた数列を符号語に使用するが、これに対する構造は図13に示されている。

【0094】

上述したTFCIビットを符号化する過程に対する理解を助けるため、下記<表1A>乃至<表1F>では10ビットのTFCIビットそれぞれに対応して符号化された符号化シンボル(TFCI符号語)を示している。

【0095】

【表 1 A】

0000000000 :	0000000001 :
00000000000000000000000000000000	11111111111111111111111111111111
0000000010 :	0000000011 :
01010101010101010101010101010101	10101010101010101010101010101010
0000000100 :	0000000101 :
00110011001100110011001100110011	11001100110011001100110011001100
0000000110 :	0000000111 :
01100110011001100110011001100110	10011001100110011001100110011001
0000001000 :	0000001001 :
00001111000011110000111100001111	11110000111100001111000011110000
0000001010 :	0000001011 :
01011010010110100101101001011010	10100101101001011010010110100101
0000001100 :	0000001101 :
00111100001111000011110000111100	11000011110000111100001111000011
0000001110 :	0000001111 :
01101001011010010110100101101001	10010110100101101001011010010110
0000010000 :	0000010001 :
00000000111111110000000011111111	11111111000000001111111100000000
0000010010 :	0000010011 :
01010101101010100101010110101010	10101010010101011010101001010101
0000010100 :	0000010101 :
00110011110011000011001111001100	11001100001100111100110000110011
0000010110 :	0000010111 :
01100110100110010110011010011001	10011001011001101001100101100110
0000011000 :	0000011001 :
00001111111100000000111111110000	11110000000011111111000000001111
0000011010 :	0000011011 :
01011010101001010101101010100101	10100101010110101010010101011010
0000011100 :	0000011101 :
00111100110000110011110011000011	11000011001111001100001100111100
0000011110 :	0000011111 :
01101001100101100110100110010110	10010110011010011001011001101001
0000100000 :	0000100001 :
00000000000000001111111111111111	11111111111111110000000000000000
0000100010 :	0000100011 :
01010101010101011010101010101010	10101010101010100101010101010101

10

20

30

40

0000100100 :	0000100101 :
00110011001100111100110011001100	11001100110011000011001100110011
0000100110 :	0000100111 :
01100110011001101001100110011001	10011001100110010110011001100110
0000101000 :	0000101001 :
00001111000011111111000011110000	11110000111100000000111100001111
0000101010 :	0000101011 :
01011010010110101010010110100101	10100101101001010101101001011010
0000101100 :	0000101101 :
00111100001111001100001111000011	11000011110000110011110000111100
0000101110 :	0000101111 :
01101001011010011001011010010110	10010110100101100110100101101001
0000110000 :	0000110001 :
00000000111111111111111100000000	11111111000000000000000011111111
0000110010 :	0000110011 :
01010101101010101010101001010101	10101010010101010101011010101010
0000110100 :	0000110101 :
00110011110011001100110000110011	11001100001100110011001111001100
0000110110 :	0000110111 :
01100110100110011001100101100110	10011001011001100110011010011001
0000111000 :	0000111001 :
00001111111100001111000000001111	11110000000011110000111111110000
0000111010 :	0000111011 :
010110101001011010010101011010	10100101010110100101101010100101
0000111100 :	0000111101 :
00111100110000111100001100111100	11000011001111000011110011000011
0000111110 :	0000111111 :
01101001100101101001011001101001	10010110011010010110100110010110
0001000000 :	0001000001 :
00101000011000111111000001110111	1101011110011100000011110001000
0001000010 :	0001000011 :
01111101001101101010010100100010	10000010110010010101010101101101
0001000100 :	0001000101 :
00011011010100001100001101000100	11100100101011110011110010111011
0001000110 :	0001000111 :
01001110000001011001011000010001	1011000111110100110100111101110

10

20

30

40

0001001000 :	0001001001 :
0010011101101100111111101111000	11011000100100110000000010000111
0001001010 :	0001001011 :
01110010001110011010101000101101	10001101110001100101010111010010
0001001100 :	0001001101 :
00010100010111111100110001001011	11101011101000000011001110110100
0001001110 :	0001001111 :
01000001000010101001100100011110	10111110111101010110011011100001
0001010000 :	0001010001 :
00101000100111001111000010001000	11010111011000110000111101110111
0001010010 :	0001010011 :
01111101110010011010010111011101	10000010001101100101101000100010
0001010100 :	0001010101 :
00011011101011111100001110111011	11100100010100000011110001000100
0001010110 :	0001010111 :
01001110111110101001011011101110	10110001000001010110100100010001
0001011000 :	0001011001 :
00100111100100111111111110000111	11011000011011000000000001111000
0001011010 :	0001011011 :
01110010110001101010101011010010	10001101001110010101010100101101
0001011100 :	0001011101 :
00010100101000001100110010110100	11101011010111110011001101001011
0001011110 :	0001011111 :
01000001111101011001100111100001	10111110000010100110011000011110
0001100000 :	0001100001 :
00101000011000110000111110001000	11010111100111001111000001110111
0001100010 :	0001100011 :
01111101001101100101101011011101	10000010110010011010010100100010
0001100100 :	0001100101 :
00011011010100000011110010111011	11100100101011111100001101000100
0001100110 :	0001100111 :
01001110000001010110100111101110	10110001111110101001011000010001
0001101000 :	0001101001 :
00100111011011000000000010000111	11011000100100111111111101111000
0001101010 :	0001101011 :
011100100011100101010111010010	10001101110001101010101000101101

10

20

30

40

0001101100 :	0001101101 :
00010100010111110011001110110100	11101011101000001100110001001011
0001101110 :	0001101111 :
01000001000010100110011011100001	10111110111101011001100100011110
0001110000 :	0001110001 :
00101000100111000000111101110111	11010111011000111111000010001000
0001110010 :	0001110011 :
01111101110010010101101000100010	10000010001101101010010111011101
0001110100 :	0001110101 :
00011011101011110011110001000100	11100100010100001100001110111011
0001110110 :	0001110111 :
01001110111110100110100100010001	10110001000001011001011011101110
0001111000 :	0001111001 :
00100111100100110000000001111000	1101100001101100111111110000111
0001111010 :	0001111011 :
01110010110001100101010100101101	10001101001110011010101011010010
0001111100 :	0001111101 :
00010100101000000011001101001011	11101011010111111100110010110100
0001111110 :	0001111111 :
01000001111101010110011000011110	10111110000010101001100111100001
0010000000 :	0010000001 :
00000001110011010110110111000111	11111110001100101001001000111000
0010000010 :	0010000011 :
01010100100110000011100010010010	10101011011001111100011101101101
0010000100 :	0010000101 :
0011001011111100101111011110100	11001101000000011010000100001011
0010000110 :	0010000111 :
01100111101010110000101110100001	10011000010101001111010001011110
0010001000 :	0010001001 :
00001110110000100110001011001000	11110001001111011001110100110111
0010001010 :	0010001011 :
01011011100101110011011110011101	10100100011010001100100001100010
0010001100 :	0010001101 :
00111101111100010101000111111011	11000010000011101010111000000100
0010001110 :	0010001111 :
01101000101001000000010010101110	10010111010110111111101101010001

10

20

30

40

0010010000 :	0010010001 :
00000001001100100110110100111000	1111110110011011001001011000111
0010010010 :	0010010011 :
01010100011001110011100001101101	10101011100110001100011110010010
0010010100 :	0010010101 :
00110010000000010101111000001011	11001101111111101010000111110100
0010010110 :	0010010111 :
01100111010101000000101101011110	10011000101010111111010010100001
0010011000 :	0010011001 :
00001110001111010110001000110111	11110001110000101001110111001000
0010011010 :	0010011011 :
01011011011010000011011101100010	10100100100101111100100010011101
0010011100 :	0010011101 :
00111101000011100101000100000100	11000010111100011010111011111011
0010011110 :	0010011111 :
01101000010110110000010001010001	10010111101001001111101110101110
0010100000 :	0010100001 :
00000001110011011001001000111000	11111110001100100110110111000111
0010100010 :	0010100011 :
01010100100110001100011101101101	10101011011001110011100010010010
0010100100 :	0010100101 :
00110010111111101010000100001011	11001101000000010101111011110100
0010100110 :	0010100111 :
01100111101010111111010001011110	10011000010101000000101110100001
0010101000 :	0010101001 :
00001110110000101001110100110111	11110001001111010110001011001000
0010101010 :	0010101011 :
01011011100101111100100001100010	10100100011010000011011110011101
0010101100 :	0010101101 :
00111101111100011010111000000100	11000010000011100101000111111011
0010101110 :	0010101111 :
01101000101001001111101101010001	10010111010110110000010010101110
0010110000 :	0010110001 :
00000001001100101001001011000111	11111110110011010110110100111000
0010110010 :	0010110011 :
01010100011001111100011110010010	10101011100110000011100001101101

10

20

30

40

0010110100 :	0010110101 :
00110010000000011010000111110100	11001101111111100101111000001011
0010110110 :	0010110111 :
01100111010101001111010010100001	10011000101010110000101101011110
0010111000 :	0010111001 :
00001110001111011001110111001000	11110001110000100110001000110111
0010111010 :	0010111011 :
01011011011010001100100010011101	10100100100101110011011101100010
0010111100 :	0010111101 :
00111101000011101010111011111011	11000010111100010101000100000100
0010111110 :	0010111111 :
01101000010110111111101110101110	10010111101001000000010001010001
0011000000 :	0011000001 :
00101001101011101001110110110000	11010110010100010110001001001111
0011000010 :	0011000011 :
01111100111110111100100011100101	10000011000001000011011100011010
0011000100 :	0011000101 :
00011010100111011010111010000011	11100101011000100101000101111100
0011000110 :	0011000111 :
01001111110010001111101111010110	10110000001101110000010000101001
0011001000 :	
00100110101000011001001010111111	

10

20

【表 1 B】

0000000000 :	0000000001 :
00000000000000000000000000000000	11111111111111111111111111111111
0000000010 :	0000000011 :
01010101010101010101010101010101	10101010101010101010101010101010
0000000100 :	0000000101 :
00110011001100110011001100110011	11001100110011001100110011001100
0000000110 :	0000000111 :
01100110011001100110011001100110	10011001100110011001100110011001
0000001000 :	0000001001 :
00001111000011110000111100001111	11110000111100001111000011110000
0000001010 :	0000001011 :
01011010010110100101101001011010	10100101101001011010010110100101
0000001100 :	0000001101 :
00111100001111000011110000111100	11000011110000111100001111000011
0000001110 :	0000001111 :
01101001011010010110100101101001	10010110100101101001011010010110
0000010000 :	0000010001 :
00000000111111110000000011111111	11111111000000001111111100000000
0000010010 :	0000010011 :
01010101101010100101010110101010	10101010010101011010101001010101
0000010100 :	0000010101 :
00110011110011000011001111001100	11001100001100111100110000110011
0000010110 :	0000010111 :
01100110100110010110011010011001	10011001011001101001100101100110
0000011000 :	0000011001 :
00001111111100000000111111110000	11110000000011111111000000001111
0000011010 :	0000011011 :
01011010101001010101101010100101	10100101010110101010010101011010
0000011100 :	0000011101 :
00111100110000110011110011000011	11000011001111001100001100111100
0000011110 :	0000011111 :
01101001100101100110100110010110	10010110011010011001011001101001
0000100000 :	0000100001 :
00000000000000001111111111111111	11111111111111111100000000000000
0000100010 :	0000100011 :
01010101010101101010101010101010	10101010101010010101010101010101

10

20

30

40

0000100100 :	0000100101 :
00110011001100111100110011001100	11001100110011000011001100110011
0000100110 :	0000100111 :
01100110011001101001100110011001	10011001100110010110011001100110
0000101000 :	0000101001 :
00001111000011111111000011110000	11110000111100000000111100001111
0000101010 :	0000101011 :
01011010010110101010010110100101	10100101101001010101101001011010
0000101100 :	0000101101 :
00111100001111001100001111000011	11000011110000110011110000111100
0000101110 :	0000101111 :
01101001011010011001011010010110	10010110100101100110100101101001
0000110000 :	0000110001 :
000000001111111111111100000000	11111111000000000000000011111111
0000110010 :	0000110011 :
010101011010101010101001010101	10101010010101010101010110101010
0000110100 :	0000110101 :
00110011110011001100110000110011	11001100001100110011001111001100
0000110110 :	0000110111 :
01100110100110011001100101100110	10011001011001100110011010011001
0000111000 :	0000111001 :
00001111111100001111000000001111	11110000000011110000111111110000
0000111010 :	0000111011 :
010110101001011010010101011010	10100101010110100101101010100101
0000111100 :	0000111101 :
00111100110000111100001100111100	11000011001111000011110011000011
0000111110 :	0000111111 :
01101001100101101001011001101001	10010110011010010110100110010110
0001000000 :	0001000001 :
00101000011000111111000001110111	11010111100111000000111110001000
0001000010 :	0001000011 :
01111101001101101010010100100010	10000010110010010101101011011101
0001000100 :	0001000101 :
00011011010100001100001101000100	11100100101011110011110010111011
0001000110 :	0001000111 :
01001110000001011001011000010001	10110001111110100110100111101110

10

20

30

40

0001001000 :	0001001001 :
0010011101101100111111101111000	11011000100100110000000010000111
0001001010 :	0001001011 :
01110010001110011010101000101101	10001101110001100101010111010010
0001001100 :	0001001101 :
00010100010111111100110001001011	11101011101000000011001110110100
0001001110 :	0001001111 :
01000001000010101001100100011110	10111110111101010110011011100001
0001010000 :	0001010001 :
00101000100111001111000010001000	11010111011000110000111101110111
0001010010 :	0001010011 :
01111101110010011010010111011101	10000010001101100101101000100010
0001010100 :	0001010101 :
00011011101011111100001110111011	11100100010100000011110001000100
0001010110 :	0001010111 :
01001110111110101001011011101110	10110001000001010110100100010001
0001011000 :	0001011001 :
00100111100100111111111110000111	110110000110110000000000001111000
0001011010 :	0001011011 :
01110010110001101010101011010010	10001101001110010101010100101101
0001011100 :	0001011101 :
00010100101000001100110010110100	11101011010111110011001101001011
0001011110 :	0001011111 :
01000001111101011001100111100001	10111110000010100110011000011110
0001100000 :	0001100001 :
00101000011000110000111110001000	11010111100111001111000001110111
0001100010 :	0001100011 :
01111101001101100101101011011101	10000010110010011010010100100010
0001100100 :	0001100101 :
00011011010100000011110010111011	11100100101011111100001101000100
0001100110 :	0001100111 :
01001110000001010110100111101110	10110001111110101001011000010001
0001101000 :	0001101001 :
00100111011011000000000010000111	11011000100100111111111101111000
0001101010 :	0001101011 :
011100100011100101010111010010	10001101110001101010101000101101

10

20

30

40

0001101100 :	0001101101 :
00010100010111110011001110110100	11101011101000001100110001001011
0001101110 :	0001101111 :
01000001000010100110011011100001	10111110111101011001100100011110
0001110000 :	0001110001 :
00101000100111000000111101110111	11010111011000111111000010001000
0001110010 :	0001110011 :
01111101110010010101101000100010	10000010001101101010010111011101
0001110100 :	0001110101 :
00011011101011110011110001000100	11100100010100001100001110111011
0001110110 :	0001110111 :
01001110111110100110100100010001	10110001000001011001011011101110
0001111000 :	0001111001 :
00100111100100110000000001111000	11011000011011001111111110000111
0001111010 :	0001111011 :
01110010110001100101010100101101	10001101001110011010101011010010
0001111100 :	0001111101 :
00010100101000000011001101001011	11101011010111111100110010110100
0001111110 :	0001111111 :
01000001111101010110011000011110	10111110000010101001100111100001
0010000000 :	0010000001 :
00000001110011010110110111000111	11111110001100101001001000111000
0010000010 :	0010000011 :
01010100100110000011100010010010	10101011011001111100011101101101
0010000100 :	0010000101 :
0011001011111100101111011110100	11001101000000011010000100001011
0010000110 :	0010000111 :
01100111101010110000101110100001	10011000010101001111010001011110
0010001000 :	0010001001 :
00001110110000100110001011001000	11110001001111011001110100110111
0010001010 :	0010001011 :
01011011100101110011011110011101	10100100011010001100100001100010
0010001100 :	0010001101 :
00111101111100010101000111111011	11000010000011101010111000000100
0010001110 :	0010001111 :
01101000101001000000010010101110	10010111010110111111101101010001

10

20

30

40

0010010000 :	0010010001 :
00000001001100100110110100111000	11111110110011011001001011000111
0010010010 :	0010010011 :
01010100011001110011100001101101	10101011100110001100011110010010
0010010100 :	0010010101 :
00110010000000010101111000001011	11001101111111101010000111110100
0010010110 :	0010010111 :
01100111010101000000101101011110	10011000101010111111010010100001
0010011000 :	0010011001 :
00001110001111010110001000110111	11110001110000101001110111001000
0010011010 :	0010011011 :
01011011011010000011011101100010	10100100100101111100100010011101
0010011100 :	0010011101 :
00111101000011100101000100000100	1100001011110001101011101111011
0010011110 :	0010011111 :
01101000010110110000010001010001	10010111101001001111101110101110
0010100000 :	0010100001 :
00000001110011011001001000111000	11111110001100100110110111000111
0010100010 :	0010100011 :
01010100100110001100011101101101	10101011011001110011100010010010
0010100100 :	0010100101 :
00110010111111101010000100001011	11001101000000010101111011110100
0010100110 :	0010100111 :
01100111101010111111010001011110	10011000010101000000101110100001
0010101000 :	0010101001 :
00001110110000101001110100110111	11110001001111010110001011001000
0010101010 :	0010101011 :
01011011100101111100100001100010	10100100011010000011011110011101
0010101100 :	0010101101 :
00111101111100011010111000000100	11000010000011100101000111111011
0010101110 :	0010101111 :
01101000101001001111101101010001	10010111010110110000010010101110
0010110000 :	0010110001 :
00000001001100101001001011000111	11111110110011010110110100111000
0010110010 :	0010110011 :
01010100011001111100011110010010	10101011100110000011100001101101

10

20

30

40

0010110100 :	0010110101 :
00110010000000011010000111110100	11001101111111100101111000001011
0010110110 :	0010110111 :
01100111010101001111010010100001	10011000101010110000101101011110
0010111000 :	0010111001 :
00001110001111011001110111001000	11110001110000100110001000110111
0010111010 :	0010111011 :
01011011011010001100100010011101	10100100100101110011011101100010
0010111100 :	0010111101 :
00111101000011101010111011111011	11000010111100010101000100000100
0010111110 :	0010111111 :
01101000010110111111101110101110	10010111101001000000010001010001
0011000000 :	0011000001 :
00101001101011101001110110110000	11010110010100010110001001001111
0011000010 :	0011000011 :
01111100111110111100100011100101	10000011000001000011011100011010
0011000100 :	0011000101 :
00011010100111011010111010000011	11100101011000100101000101111100
0011000110 :	0011000111 :
01001111110010001111101111010110	10110000001101110000010000101001
0011001000 :	
00100110101000011001001010111111	

10

20

30

40

0000000000 :	0000000001 :
000000000000000000000000000000	11111111111111111111111111111111
00000000010 :	00000000011 :
010101010101010101010101010101	10101010101010101010101010101010
00000000100 :	00000000101 :
00110011001100110011001100110011	11001100110011001100110011001100
00000000110 :	00000000111 :
01100110011001100110011001100110	10011001100110011001100110011001
00000001000 :	00000001001 :
00001111000011110000111100001111	11110000111100001111000011110000
00000001010 :	00000001011 :
01011010010110100101101001011010	10100101101001011010010110100101
00000001100 :	00000001101 :
00111100001111000011110000111100	11000011110000111100001111000011
00000001110 :	00000001111 :
01101001011010010110100101101001	10010110100101101001011010010110
00000010000 :	00000010001 :
00000000011111110000000011111111	11111111000000001111111100000000
00000010010 :	00000010011 :
01010101101010100101010110101010	10101010010101011010101001010101
00000010100 :	00000010101 :
00110011110011000011001111001100	11001100001100111100110000110011
00000010110 :	00000010111 :
01100110100110010110011010011001	10011001011001101001100101100110
00000011000 :	00000011001 :
00001111111100000000111111110000	11110000000011111111000000001111
00000011010 :	00000011011 :
01011010101001010101101010100101	10100101010110101010010101011010
00000011100 :	00000011101 :
00111100110000110011110011000011	11000011001111001100001100111100
00000011110 :	00000011111 :
01101001100101100110100110010110	10010110011010011001011001101001
00001000000 :	00001000001 :
00000000000000000011111111111111	11111111111111110000000000000000
00001000010 :	00001000011 :
01010101010101011010101010101010	10101010101010100010101010101010

0000100100 :	0000100101 :
00110011001100111100110011001100	11001100110011000011001100110011
0000100110 :	0000100111 :
01100110011001101001100110011001	10011001100110010110011001100110
0000101000 :	0000101001 :
00001111000011111111000011110000	11110000111100000000111100001111
0000101010 :	0000101011 :
01011010010110101010010110100101	10100101101001010101101001011010
0000101100 :	0000101101 :
00111100001111001100001111000011	11000011110000110011110000111100
0000101110 :	0000101111 :
01101001011010011001011010010110	10010110100101100110100101101001
0000110000 :	0000110001 :
000000001111111111111100000000	11111111000000000000000011111111
0000110010 :	0000110011 :
010101011010101010101001010101	101010100101010101010110101010
0000110100 :	0000110101 :
00110011110011001100110000110011	11001100001100110011001111001100
0000110110 :	0000110111 :
01100110100110011001100101100110	10011001011001100110011010011001
0000111000 :	0000111001 :
00001111111100001111000000001111	11110000000011110000111111110000
0000111010 :	0000111011 :
01011010101001011010010101011010	10100101010110100101101010100101
0000111100 :	0000111101 :
00111100110000111100001100111100	11000011001111000011110011000011
0000111110 :	0000111111 :
01101001100101101001011001101001	10010110011010010110100110010110
0001000000 :	0001000001 :
00101000011000111111000001110111	11010111100111000000111110001000
0001000010 :	0001000011 :
01111101001101101010010100100010	10000010110010010101101011011101
0001000100 :	0001000101 :
00011011010100001100001101000100	11100100101011110011110010111011
0001000110 :	0001000111 :
01001110000001011001011000010001	10110001111110100110100111101110

10

20

30

40

0001001000 :	0001001001 :
0010011101101100111111101111000	11011000100100110000000010000111
0001001010 :	0001001011 :
01110010001110011010101000101101	10001101110001100101010111010010
0001001100 :	0001001101 :
00010100010111111100110001001011	11101011101000000011001110110100
0001001110 :	0001001111 :
01000001000010101001100100011110	10111110111101010110011011100001
0001010000 :	0001010001 :
00101000100111001111000010001000	11010111011000110000111101110111
0001010010 :	0001010011 :
01111101110010011010010111011101	10000010001101100101101000100010
0001010100 :	0001010101 :
00011011101011111100001110111011	11100100010100000011110001000100
0001010110 :	0001010111 :
01001110111110101001011011101110	10110001000001010110100100010001
0001011000 :	0001011001 :
0010011110010011111111110000111	110110000110110000000000001111000
0001011010 :	0001011011 :
01110010110001101010101011010010	10001101001110010101010100101101
0001011100 :	0001011101 :
000101001010000001100110010110100	11101011010111110011001101001011
0001011110 :	0001011111 :
01000001111101011001100111100001	10111110000010100110011000011110
0001100000 :	0001100001 :
00101000011000110000111110001000	11010111100111001111000001110111
0001100010 :	0001100011 :
01111101001101100101101011011101	10000010110010011010010100100010
0001100100 :	0001100101 :
00011011010100000011110010111011	11100100101011111100001101000100
0001100110 :	0001100111 :
01001110000001010110100111101110	10110001111110101001011000010001
0001101000 :	0001101001 :
001001110110110000000000010000111	11011000100100111111111101111000
0001101010 :	0001101011 :
011100100011100101010111010010	10001101110001101010101000101101

10

20

30

40

0001101100 :	0001101101 :
00010100010111110011001110110100	11101011101000001100110001001011
0001101110 :	0001101111 :
01000001000010100110011011100001	1011110111101011001100100011110
0001110000 :	0001110001 :
00101000100111000000111101110111	11010111011000111111000010001000
0001110010 :	0001110011 :
01111101110010010101101000100010	10000010001101101010010111011101
0001110100 :	0001110101 :
00011011101011110011110001000100	11100100010100001100001110111011
0001110110 :	0001110111 :
01001110111110100110100100010001	10110001000001011001011011101110
0001111000 :	0001111001 :
00100111100100110000000001111000	1101100001101100111111110000111
0001111010 :	0001111011 :
01110010110001100101010100101101	10001101001110011010101011010010
0001111100 :	0001111101 :
00010100101000000011001101001011	11101011010111111100110010110100
0001111110 :	0001111111 :
01000001111101010110011000011110	10111110000010101001100111100001
0010000000 :	0010000001 :
00000001110011010110110111000111	11111110001100101001001000111000
0010000010 :	0010000011 :
01010100100110000011100010010010	10101011011001111100011101101101
0010000100 :	0010000101 :
00110010111111100101111011110100	11001101000000011010000100001011
0010000110 :	0010000111 :
01100111101010110000101110100001	10011000010101001111010001011110
0010001000 :	0010001001 :
00001110110000100110001011001000	11110001001111011001110100110111
0010001010 :	0010001011 :
01011011100101110011011110011101	10100100011010001100100001100010
0010001100 :	0010001101 :
00111101111100010101000111111011	11000010000011101010111000000100
0010001110 :	0010001111 :
01101000101001000000010010101110	10010111010110111111101101010001

10

20

30

40

0010010000 :	0010010001 :
00000001001100100110110100111000	1111110110011011001001011000111
0010010010 :	0010010011 :
01010100011001110011100001101101	10101011100110001100011110010010
0010010100 :	0010010101 :
00110010000000010101111000001011	1100110111111101010000111110100
0010010110 :	0010010111 :
01100111010101000000101101011110	10011000101010111111010010100001
0010011000 :	0010011001 :
00001110001111010110001000110111	11110001110000101001110111001000
0010011010 :	0010011011 :
01011011011010000011011101100010	10100100100101111100100010011101
0010011100 :	0010011101 :
00111101000011100101000100000100	11000010111100011010111011111011
0010011110 :	0010011111 :
01101000010110110000010001010001	10010111101001001111101110101110
0010100000 :	0010100001 :
00000001110011011001001000111000	1111110001100100110110111000111
0010100010 :	0010100011 :
01010100100110001100011101101101	10101011011001110011100010010010
0010100100 :	0010100101 :
00110010111111101010000100001011	11001101000000010101111011110100
0010100110 :	0010100111 :
01100111101010111111010001011110	10011000010101000000101110100001
0010101000 :	0010101001 :
00001110110000101001110100110111	11110001001111010110001011001000
0010101010 :	0010101011 :
01011011100101111100100001100010	10100100011010000011011110011101
0010101100 :	0010101101 :
00111101111100011010111000000100	11000010000011100101000111111011
0010101110 :	0010101111 :
01101000101001001111101101010001	10010111010110110000010010101110
0010110000 :	0010110001 :
00000001001100101001001011000111	11111110110011010110110100111000
0010110010 :	0010110011 :
01010100011001111100011110010010	10101011100110000011100001101101

10

20

30

40

0010110100 :	0010110101 :
00110010000000011010000111110100	11001101111111100101111000001011
0010110110 :	0010110111 :
01100111010101001111010010100001	10011000101010110000101101011110
0010111000 :	0010111001 :
00001110001111011001110111001000	11110001110000100110001000110111
0010111010 :	0010111011 :
01011011011010001100100010011101	10100100100101110011011101100010
0010111100 :	0010111101 :
00111101000011101010111011111011	11000010111100010101000100000100
0010111110 :	0010111111 :
01101000010110111111101110101110	10010111101001000000010001010001
0011000000 :	0011000001 :
00101001101011101001110110110000	11010110010100010110001001001111
0011000010 :	0011000011 :
0111110011110111100100011100101	10000011000001000011011100011010
0011000100 :	0011000101 :
00011010100111011010111010000011	11100101011000100101000101111100
0011000110 :	0011000111 :
01001111110010001111101111010110	10110000001101110000010000101001
0011001000 :	
00100110101000011001001010111111	

10

20

【表 1 D】

1001011011 :	1001011100 :
10010001000011100111101001111100	00001000100101111110001111100101
1001011101 :	1001011110 :
11110111011010000001110000011010	01011101110000101011011010110000
1001011111 :	1001100000 :
10100010001111010100100101001111	0110100010101000010000011011001
1001100001 :	1001100010 :
11001011101010111101111100100110	1100001000000010111010110001100
1001100011 :	1001100100 :
10011110111111101000101001110011	00000111011001110001001111101010
1001100101 :	1001100110 :
1111000100110001110110000010101	01010010001100100100011010111111
1001100111 :	1001101000 :
10101101110011011011100101000000	00111011010110110010111111010110
1001101001 :	1001101010 :
1000100101001001101000000101001	01101110000011100111101010000011
1001101011 :	1001101100 :
10010001111100011000010101111100	0001000011010000001110011100101
1001101101 :	1001101110 :
11110111100101111110001100011010	01011101001111010100100110110000
1001101111 :	1001110000 :
10100010110000101011011001001111	0110100101010110010000000100110
1001110001 :	1001110010 :
11001011101010100110111111011001	01100001111111100111010101110011
1001110011 :10011110000000011000101	1001110100 :
010001100	0000111100110000001001100010101
1001110101 :	1001110110 :
11111000011001111110110011101010	1010010110011010100011001000000
1001110111 :	1001111000 :
10101101001100101011100110111111	00111011101001000010111100101001
1001111001 :	1001111010 :
11000100010110111101000011010110	01101110111100010111101001111100
1001111011 :	1001111100 :
0010001000011101000010110000011	00001000100101110001110000011010
1001111101 :	1001111110 :
11110111011010001110001111100101	01011101110000100100100101001111

10

20

30

40

1001111111 :	1010000000 :
101000100011110110110110110000	00011101111110100100001010010110
1010000001 :	1010000010 :
1100010000001011011110101101001	0100100010101111000101111000011
1010000011 :	1010000100 :
0110111010100001110100000111100	0101110110010010111000110100101
1010000101 :	1010000110 :
1010001001101101000111001011010	01111011100111000010010011110000
1010000111 :	1010001000 :
10000100011000111101101100001111	0010010111101010100110110011001
1010001001 :	1010001010 :
1101101000010101011001001100110	1000111101000000001100011001100
1010001011 :	1010001100 :
1011100001011111110011100110011	00100001110001100111111010101010
1010001101 :	1010001110 :
1011110001110011000000101010101	01110100100100110010101111111111
1010001111 :	1010010000 :
00010110110110011010100000000000	0011101000001010100001001101001
1010010001 :	1010010010 :
11100010111110101011110110010110	1001000010100000001011100111100
1010010011 :	1010010100 :
10110111101011111110100011000011	0101110001101100111000101011010
1010010101 :	1010010110 :
1010001110010011000111010100101	01111011011000110010010000001111
1010010111 :	1010011000 :
10000100100111001101101111110000	0010010000010100100110101100110
1010011001 :	1010011010 :
11101101111101011011001010011001	01000111010111110001100000110011
1010011011 :	1010011100 :
10111000101000001110011111001100	00100001001110010111111001010101
1010011101 :	1010011110 :
1011110110001101000000110101010	01110100011011000010101100000000
1010011111 :	1010100000 :
10001011100100111101010011111111	00011101111110101011110101101001
1010100001 :	1010100010 :
1100010000001010100001010010110	01001000101011111110100000111100

10

20

30

40

1010100011 :	1010100100 :
011011101010000000101111000011	0101110110010011000111001011010
1010100101 :	1010100110 :
1010001001101100111000110100101	01111011100111001101101100001111
1010100111 :	1010101000 :
0000100011000110010010011110000	0010010111101011011001001100110
1010101001 :	1010101010 :
1101101000010100100110110011001	01000111101000001110011100110011
1010101011 :	1010101100 :
0111000010111110001100011001100	0100001110001101000000101010101
1010101101 :	1010101110 :
11011110001110010111111010101010	11101001001001111101010000000000
1010101111 :	1010110000 :
1000101101101100001010111111111	00011101000001011011110110010110
1010110001 :	1010110010 :
1100010111110100100001001101001	1001000010100001110100011000011
1010110011 :	1010110100 :
10110111101011110001011100111100	0101110001101101000111010100101
1010110101 :	1010110110 :
1010001110010010111000101011010	0111101101100011110110111110000
1010110111 :	1010111000 :
0000100100111000010010000001111	0010010000010101011001010011001
1010111001 :	1010111010 :
11101101111101010100110101100110	0100011101011111110011111001100
1010111011 :	1010111100 :
0111000101000000001100000110011	0100001001110011000000110101010
1010111101 :	1010111110 :
11011110110001100111111001010101	01110100011011001101010011111111
1010111111 :	1011000000 :
0001011100100110010101100000000	00110101100110011011001011100001
1011000001 :	1011000010 :
1001010011001100100110100011110	01100000110011001110011110110100
1011000011 :	1011000100 :
0011111001100110001100001001011	000011010101010101000000111010010
1011000101 :	1011000110 :
11111001010101010111111000101101	01010011111111111101010010000111

10

20

30

40

1011000111 :	1011001000 :
0101100000000000010101101111000	00111010100101101011110111101110
1011001001 :	1011001010 :
1000101011010010100001000010001	01101111110000111110100010111011
1011001011 :	1011001100 :
0010000001111000001011101000100	00001001101001011000111011011101
1011001101 :	1011001110 :
1110110010110100111000100100010	01011100111100001101101110001000
1011001111 :	1011010000 :
10100011000011110010010001110111	00110101011001101011001000011110
1011010001 :	1011010010 :
1001010100110010100110111100001	01100000001100111110011101001011
1011010011 :	1011010100 :
10011111110011000001100010110100	0000110010101011000000100101101
1011010101 :	1011010110 :
11111001101010100111111011010010	1010011000000001101010001111000
1011010111 :	1011011000 :
1010110011111110010101110000111	00111010011010011011110100010001
1011011001 :	1011011010 :
1000101100101100100001011101110	01101111001111001110100001000100
1011011011 :	1011011100 :
10010000110000110001011110111011	0001001010110101000111000100010
1011011101 :	1011011110 :
11110110101001010111000111011101	01011100000011111101101101110111
1011011111 :	1011100000 :
10100011111100000010010010001000	00110101100110010100110100011110
1011100001 :	1011100010 :
11001010011001101011001011100001	01100000110011000001100001001011
1011100011 :	1011100100 :
10011111001100111110011110110100	00000110101010100111111000101101
1011100101 :	1011100110 :
11111001010101011000000111010010	01010011111111110010101101111000
1011100111 :	1011101000 :
10101100000000001101010010000111	00111010100101100100001000010001
1011101001 :	1011101010 :
11000101011010011011110111101110	01101111110000110001011101000100

10

20

30

40

1011101011 :	1011101100 :
10010000001111001110100010111011	00001001101001010111000100100010
1011101101 :	1011101110 :
11110110010110101000111011011101	01011100111100000010010001110111
1011101111 :	1011110000 :
10100011000011111101101110001000	00110101011001100100110111100001
1011110001 :	1011110010 :
11001010100110011011001000011110	01100000001100110001100010110100
1011110011 :	1011110100 :
10011111110011001110011101001011	00000110010101010111111011010010
1011110101 :	1011110110 :
11111001101010101000000100101101	010100110000000000010101110000111
1011110111 :	1011111000 :
1010110011111111101010001111000	00111010011010010100001011101110
1011111001 :	1011111010 :
11000101100101101011110100010001	01101111001111000001011110111011
1011111011 :	1011111100 :
10010000110000111110100001000100	00001001010110100111000111011101
1011111101 :	1011111110 :
11110110101001011000111000100010	01011100000011110010010010001000
1011111111 :	1100000000 :
10100011111100001101101101110111	00010110110011100011010001111010
1100000001 :	1100000010 :
11101001001100011100101110000101	01000011100110110110000100101111
1100000011 :	1100000100 :
10111100011001001001111011010000	00100101111111010000011101001001
1100000101 :	1100000110 :
11011010000000101111100010110110	01110000101010000101001000011100
1100000111 :	1100001000 :
10001111010101111010110111100011	00011001110000010011101101110101
1100001001 :	1100001010 :
11100110001111101100010010001010	01001100100101000110111000100000
1100001011 :	1100001100 :
10110011011010111001000111011111	00101010111100100000100001000110
1100001101 :	1100001110 :
1101010100001101111101110111001	01111111101001110101110100010011

10

20

30

40

1100001111 :	1100010000 :
10000000010110001010001011101100	00010110001100010011010010000101
1100010001 :	1100010010 :
11101001110011101100101101111010	01000011011001000110000111010000
1100010011 :	1100010100 :
10111100100110111001111000101111	0100101000000100000011110110110
1100010101 :	1100010110 :
1101101011111011111100001001001	01110000010101110101001011100011
1100010111 :	1100011000 :
10001111101010001010110100011100	00011001001111100011101110001010
1100011001 :	1100011010 :
11100110110000011100010001110101	01001100011010110110111011011111
1100011011 :10110011100101001001000	1100011100 :
100100000	0101010000011010000100010111001
1100011101 :	1100011110 :
11010101111100101111011101000110	01111111010110000101110111101100
1100011111 :	1100100000 :
10000000101001111010001000010011	00010110110011101100101110000101
1100100001 :	1100100010 :
11101001001100010011010001111010	01000011100110111001111011010000
1100100011 :	
10111100011001000110000100101111	

10

20

【表 1 E】

1100100100:	1100100101:
0010010111111101111100010110110	11011010000000100000011101001001
1100100110:	1100100111:
01110000101010001010110111100011	10001111010101110101001000011100
1100101000:	1100101001:
00011001110000011100010010001010	11100110001111100011101101110101
1100101010:	1100101011:
01001100100101001001000111011111	10110011011010110110111000100000
1100101100:	1100101101:
00101010111100101111011110111001	11010101000011010000100001000110
1100101110:	1100101111:
01111111101001111010001011101100	10000000010110000101110100010011
1100110000:	1100110001:
000101100011000111001101101111010	11101001110011100011010010000101
1100110010:	1100110011:
01000011011001001001111000101111	10111100100110110110000111010000
1100110100:	1100110101:
00100101000000101111100001001001	11011010111111010000011110110110
1100110110:	1100110111:
011100000010101111010110100011100	10001111101010000101001011100011
1100111000:	1100111001:
00011001001111101100010001110101	11100110110000010011101110001010
1100111010:	1100111011:
01001100011010111001000100100000	10110011100101000110111011011111
1100111100:	1100111101:
00101010000011011111011101000110	11010101111100100000100010111001
1100111110:	1100111111:
01111111010110001010001000010011	10000000101001110101110111101100
1101000000:	1101000001:
00111110101011011100010000001101	11000001010100100011101111110010
1101000010:	1101000011:
01101011111110001001000101011000	10010100000001110110111010100111
1101000100:	1101000101:
00001101100111101111011100111110	11110010011000010000100011000001
1101000110:	1101000111:
01011000110010111010001001101011	10100111001101000101110110010100

10

20

30

40

1101001000:	1101001001:
00110001101000101100101100000010	11001110010111010011010011111101
1101001010:	1101001011:
01100100111101111001111001010111	10011011000010000110000110101000
1101001100:	1101001101:
00000010100100011111100000110001	11111101011011100000011111001110
1101001110:	1101001111:
01010111110001001010110101100100	10101000001110110101001010011011
1101010000:	1101010001:
00111110010100101100010011110010	11000001101011010011101100001101
1101010010:	1101010011:
01101011000001111001000110100111	10010100111110000110111001011000
1101010100:	1101010101:
00001101011000011111011111000001	11110010100111100000100000111110
1101010110:	1101010111:
01011000001101001010001010010100	10100111110010110101110101101011
1101011000:	1101011001:
00110001010111011100101111111101	11001110101000100011010000000010
1101011010:	1101011011:
01100100000010001001111010101000	10011011111101110110000101010111
1101011100:	1101011101:
00000010011011101111100011001110	11111101100100010000011100110001
1101011110:	1101011111:
01010111001110111010110110011011	10101000110001000101001001100100
1101100000:	1101100001:
00111110101011010011101111110010	11000001010100101100010000001101
1101100010:	1101100011:
01101011111110000110111010100111	10010100000001111001000101011000
1101100100:	1101100101:
00001101100111100000100011000001	11110010011000011111011100111110
1101100110:	1101100111:
01011000110010110101110110010100	10100111001101001010001001101011
1101101000:	1101101001:
00110001101000100011010011111101	11001110010111011100101100000010
1101101010:	1101101011:
01100100111101110110000110101000	10011011000010001001111001010111

10

20

30

40

1101101100:	1101101101:
00000010100100010000011111001110	111111010110111011111000000110001
1101101110:	1101101111:
01010111110001000101001010011011	10101000001110111010110101100100
1101110000:	1101110001:
00111110010100100011101100001101	11000001101011011100010011110010
1101110010:	1101110011:
01101011000001110110111001011000	10010100111110001001000110100111
1101110100:	1101110101:
00001101011000010000100000111110	11110010100111101111011111000001
1101110110:	1101110111:
01011000001101000101110101101011	10100111110010111010001010010100
1101111000:	1101111001:
00110001010111010011010000000010	11001110101000101100101111111101
1101111010:	1101111011:
01100100000010000110000101010111	10011011111101111001111010101000
1101111100:	1101111101:
000000100110111100000011100110001	11111101100100011111100011001110
1101111110:	1101111111:
01010111001110110101001001100100	10101000110001001010110110011011
1110000000:	1110000001:
00010111000000110101100110111101	11101000111111001010011001000010
1110000010:	1110000011:
01000010010101100000110011101000	10111101101010011111001100010111
1110000100:	1110000101:
00100100001100000110101010001110	11011011110011111001010101110001
1110000110:	1110000111:
0111000101100101001111111011011	10001110100110101100000000100100
1110001000:	1110001001:
00011000000011000101011010110010	11100111111100111010100101001101
1110001010:	1110001011:
01001101010110010000001111100111	10110010101001101111110000011000
1110001100:	1110001101:
00101011001111110110010110000001	1101010011000000100110100111110
1110001110:	1110001111:
01111110011010100011000011010100	10000001100101011100111100101011

10

20

30

40

1110010000:	1110010001:
000101111111000101100101000010	1110100000000111010011010111101
1110010010:	1110010011:
01000010101010010000110000010111	10111101010101101111001111101000
1110010100:	1110010101:
00100100110011110110101001110001	11011011001100001001010110001110
1110010110:	1110010111:
01110001100110100011111100100100	10001110011001011100000011011011
1110011000:	1110011001:
00011000111100110101011001001101	11100111000011001010100110110010
1110011010:	1110011101:
01001101101001100000001100011000	10110010010110011111110011100111
1110011100:	1110011101:
00101011110000000110010101111110	11010100001111111001101010000001
1110011110:	1110011111:
01111110100101010011000000101011	10000001011010101100111111010100
1110100000:	1110100001:
00010111000000111010011001000010	11101000111111000101100110111101
11101000010:	1110100011:
01000010010101101111001100010111	10111101101010010000110011101000
1110100100:	1110100101:
00100100001100001001010101110001	11011011110011110110101010001110
1110100110:	1110100111:
01110001011001011100000000100100	1000111010011010001111111011011
1110101000:	1110101001:
00011000000011001010100101001101	11100111111100110101011010110010
1110101010:	1110101011:
01001101010110011111110000011000	10110010101001100000001111100111
1110101100:	1110101101:
00101011001111111001101001111110	11010100110000000110010110000001
1110101110:	1110101111:
01111110011010101100111100101011	10000001100101010011000011010100
1110110000:	1110110001:
00010111111111001010011010111101	11101000000000110101100101000010
1110110010:	1110110011:
01000010101010011111001111101000	10111101010101100000110000010111

10

20

30

40

1110110100:	1110110101:
00100100110011111001010110001110	11011011001100000110101001110001
1110110110:	1110110111:
01110001100110101100000011011011	10001110011001010011111100100100
1110111000:	1110111001:
00011000111100111010100110110010	11100111000011000101011001001101
1110111010:	1110111011:
01001101101001101111110011100111	10110010010110010000001100011000
1110111100:	1110111101:
00101011110000001001101010000001	11010100001111110110010101111110
1110111110:	1110111111:
01111110100101011100111111010100	10000001011010100011000000101011
1111000000:	1111000001:
00111111011000001010100111001010	11000000100111110101011000110101
1111000010:	1111000011:
01101010001101011111110010011111	1001010111001010000000110110000
1111000100:	1111000101:
00001100010100111001101011111001	11110011101011000110010100000110
1111000110:	1111000111:
01011001000001101100111110101100	10100110111110010011000001010011
1111001000:	1111001001:
00110000011011111010011011000101	11001111100100000101100100111010
1111001010:	1111001011:
01100101001110101111001110010000	10011010110001010000110001101111
1111001100:	1111001101:
00000011010111001001010111110110	11111100101000110110101000001001
1111001110:	1111001111:
01010110000010011100000010100011	101010011111011000011111101011100
1111010000:	1111010001:
00111111100111111010100100110101	11000000011000000101011011001010
1111010010:	1111010011:
01101010110010101111110001100000	10010101001101010000001110011111
1111010100:	1111010101:
00001100101011001001101000000110	11110011010100110110010111111001
1111010110:	1111010111:
01011001111110011100111101010011	10100110000001100011000010101100

10

20

30

40

1111011000:	1111011001:
00110000100100001010011000111010	11001111011011110101100111000101
1111011010:	1111011011:
01100101110001011111001101101111	10011010001110100000110010010000
1111011100:	1111011101:
00000011101000111001010100001001	11111100010111000110101011110110
1111011110:	1111011111:
010101101111011011000000011100	10101001000010010011111110100011
1111100000:	1111100001:
00111111011000000101011000110101	11000000100111111010100111001010
1111100010:	1111100011:
01101010001101010000001101100000	10010101110010101111110010011111
1111100100:	1111100101:
00001100010100110110010100000110	11110011101011001001101011111001
1111100110:	1111100111:
01011001000001100011000001010011	10100110111110011100111110101100
1111101000:	1111101001:
00110000011011110101100100111010	11001111100100001010011011000101
1111101010:	1111101011:
01100101001110100000110001101111	10011010110001011111001110010000
1111101100:	
00000011010111000110101000001001	

10

20

【 0 1 0 0 】

30

【表 1 F】

```

1111101101 :
11111100101000111001010111110110
1111101110 :
01010110000010010011111101011100
1111101111 :
10101001111101101100000010100011
1111110000 :
00111111100111110101011011001010
1111110001 :
11000000011000001010100100110101
1111110010 :
01101010110010100000001110011111
1111110011 :
10010101001101011111110001100000
1111110100 :
00001100101011000110010111111001
1111110101 :
11110011010100111001101000000110
1111110110 :
01011001111110010011000010101100
1111110111 :
10100110000001101100111101010011
1111111000 :
00110000100100000101100111000101
1111111001 :
11001111011011111010011000111010
1111111010 :
01100101110001010000110010010000
1111111011 :
1001101000111010111001101101111
1111111100 :
00000011101000110110101011110110
1111111101 :
11111100010111001001010100001001
1111111110 :
0101011011110110001111110100011
1111111111 :
10101001000010011100000001011100

```

10

20

30

40

【0 1 0 1】

次に、図 9 を参照して本発明の一実施形態による復号化装置を説明すると、先ず入力信号 $r(t)$ は 15 個の乗算器 902、904、...、906 と相関度計算器 920 に印加

50

される。前記入力信号 $r(t)$ は送信側で所定のウォルシュ符号と所定のマスクシーケンスにより符号化された信号である。マスクシーケンス発生器 910 は 15 個のすべての場合のマスクシーケンス $M1$ 、 $M2$ 、 $\dots$ 、 $M15$ を生成して前記乗算器 902、904、 $\dots$ 、906 に出力する。従って、前記乗算器 902、904、 $\dots$ 、906 それぞれは前記マスクシーケンス発生器 910 から固有のマスクシーケンスを受信して、前記入力信号 $r(t)$ をかけて出力する。乗算器 902 は入力信号 $r(t)$ とマスクシーケンス生成器 910 からの $M1$ をかけて相関度計算器 922 に出力する。乗算器 904 は入力信号 $r(t)$ とマスクシーケンス生成器 910 からの $M2$ をかけて相関度計算器 924 に出力する。乗算器 906 は入力信号 $r(t)$ とマスクシーケンス生成器 910 からの $M15$ をかけて相関度計算器 926 に出力する。従って、送信側で所定マスクシーケンスを使用して TFCI ビットを符号化すると、前記乗算器 902、904、 $\dots$ 、906 からの出力の中のいずれか一つはマスクシーケンスが除去された信号である。例えば、送信側で $M2$ を使用して TFCI ビットを符号化すると、 $M2$ と入力信号 $r(t)$ をかける乗算器 904 の出力がマスクシーケンスの除去された信号である。前記マスクシーケンスが除去された信号は所定ウォルシュ符号のみにより符号化が遂行された TFCI ビットである。相関度計算器 920、922、924、 $\dots$ 、926 は入力信号 $r(t)$ 及び前記乗算器 902、904、 $\dots$ 、906 からの出力と 64 個の相互直交符号との相関度を計算する。前記 64 個の相互直交符号はすでに定義されている。相関度計算器 920 は入力信号 $r(t)$ と長さ 32 である 64 個の相互直交符号との相関値を求める。そして、前記求められた 64 個の相関値中、最大相関値を選択して、前記選択された相関値と、前記選択された相関値に対応する相互直交符号インデックス及び自分の固有インデックス 0000 を相関度比較器 940 に出力する。

【0102】

相関度計算器 922 は前記乗算器 902 の出力と 64 個の相互直交符号との相関値を求める。そして、前記求められた 64 種類の相関値中、一番大きな相関値を選択して、前記選択された相関値と、前記選択された相関値に対応する相互直交符号インデックス及び自分の固有インデックス 0001 を相関度比較器 940 に出力する。相関度計算器 924 は前記乗算器 904 の出力と 64 個の相互直交符号との相関値を求める。そして、前記求められた 64 種類の相関値中、一番大きな相関値を選択して、前記選択された相関値と、前記選択された相関値に対応する相互直交符号インデックス及び自分の固有インデックス 0010 を相関度比較器 940 に出力する。その以外の相関度計算器 (図示さず) は対応する乗算器の出力と 64 個の相互直交符号との相関値をそれぞれ計算し、上述した相関度計算器と同一の動作を遂行する。

【0103】

最後に、相関度計算器 926 は前記乗算器 906 の出力と 64 個の相互直交符号との相関値を求める。そして、前記求められた 64 種類の相関値中、一番大きな相関値を選択して、前記選択された相関値と、前記選択された相関値に対応する相互直交符号インデックス及び自分の固有インデックス 1111 を相関度比較器 940 に出力する。

【0104】

上述したように前記相関度計算器 920、922、924、 $\dots$ 、926 から出力される固有インデックスは、前記乗算器 902、 $\dots$ 、906 の入力信号 $r(t)$ によりかけられた所定マスクシーケンスのインデックスと同一である。前記乗算器によりかけられる 15 個のマスクシーケンスとマスクシーケンスを使用しない場合に対応するマスクシーケンスインデックスの一例を示すと、下記<表 2>のようである。

【0105】

【表 2】

マスクシーケンス	マスクシーケンス インデックス	マスクシーケンス	マスクシーケンス インデックス
使用しない	0000	M8	1000
M1	0001	M9	1001
M2	0010	M10	1010
M3	0011	M11	1011
M4	0101	M12	1100
M5	0101	M13	1101
M6	0110	M14	1110
M7	0111	M15	1111

10

【0106】

前記<表 2>から分かるように、入力信号 $r(t)$ に M1 がかけられた信号を受信する相関度計算器 922 は、固有インデックスに “0001” を出力する。入力信号 $r(t)$ に M15 がかけられた信号を受信する相関度計算器 926 は、固有インデックスに “1111” を出力する。しかし、マスクシーケンスがかけられない入力信号 $r(t)$ のみを受信する相関度計算器 920 は、固有インデックスに “0000” を出力する。

20

【0107】

一方、前記相互直交符号インデックスは 2 進符号形態に表現して使用することができる。例えば、W4 の補数である

【数 12】

$$\overline{W4}$$

に対する相関値が一番大きいと、対応する相互直交符号インデックス (a0 乃至 a9) は 001001 である。

30

【0108】

前記相関度比較器 940 は前記それぞれの相関度計算器 920 乃至 926 から受信された 16 種類の最大相関値を比較して、その中で最大の相関値を選択する。前記最大の相関値が選択されると、前記選択された相関値に対応して該当相関度計算器から受信された相互直交符号インデックスとマスクシーケンスインデックスに基づいて送信側で送信した TFCI ビットを出力する。前記 TFCI ビットは、前記相互直交符号インデックスと前記マスクシーケンスインデックスを結合することにより決定することができる。例えば、前記マスクシーケンスインデックスが M4 (0100) に対応するインデックスであり、前記相互直交符号インデックスが

40

【数 13】

$$\overline{W4}(001001)$$

に対応するインデックスであると、前記 TFCI ビット (a9 乃至 a0) は

【数 14】

$$\text{“M4 に対応するインデックス(0100) + } \overline{W4} \text{ に対応するインデックス(001001)”}$$

になる。即ち、前記 TFCI ビット (a9 乃至 a0) は “0100001001” である。

50

【0109】

例えば、送信側でTFCIビット(a0乃至a9)“1011000010”に対応する符号化シンボルを伝送したと仮定すると、前記送信側ではTFCIビットを

【数15】

$$\overline{W6}$$

とM4に符号化して伝送する。これは上述した符号化による動作部分ですでに説明された。一方、受信側では

【数16】

$$\overline{W6}$$

10

とM4に符号化された入力信号r(t)に対してすべてのマスクシーケンスをかけることにより、前記入力信号r(t)がM4により符号化されたことを判断できるようになる。また、前記M4がかけられた入力信号r(t)に対してすべての相互直交符号との相関度を測定することにより、前記r(t)が

【数17】

$$\overline{W6}$$

により符号化されたことが分かる。上述した例に基づいて、前記第5相関度計算器(図示さず)は最大相関値、即ち

20

【数18】

$$\overline{W6}$$

のインデックス101100と固有インデックス0011を出力する。すると、受信側は前記

【数19】

$$\overline{W6}$$

30

に対応するインデックス101100とM4に対応するインデックス“0010”を加算することにより、復号されたTFCIビット(a0乃至a9)“1011000010”を出力する。

【0110】

本発明の一実施形態による復号化装置は、入力信号r(t)をマスクシーケンスの個数に対応して並列に処理する構成を示している。しかし、他の実施形態として入力信号r(t)に対してすべてのマスクシーケンスを順次的にかけ、これに対する相関度を順次的に計算する構成を有するように復号化装置を具現することができる。

【0111】

図17は復号化装置の他の実施形態を示している図である。

40

前記図17を参照すると、メモリ1720は32シンボルの入力信号r(t)を貯蔵する。マスクシーケンス発生器1710は送信側で使用する16種類のマスクシーケンスを生成して順次的に出力する。乗算器1730は前記マスク生成器1710から受信される16種類のマスクシーケンス中の一つと前記メモリ1720から受信される入力信号r(t)をかけて相関度計算器1740に出力する。前記相関度計算器1740は64種類の長さ32である相互直交符号(ウォルシュ符号)と前記乗算器1730の出力の相関度を計算した後、前記計算によるすべての相関値の中、最大相関値と、前記最大相関値に対応する相互直交符号のインデックスを相関度比較器1750に出力する。前記相関度比較器1750は前記相関度計算器1740から受信された最大相関値及び相互直交符号インデックスと、前記マスクシーケンス発生器1710から受信されたマスクシーケンスのインデック

50

スを貯蔵する。

【0112】

上述した動作により所定マスクシーケンスに対応した復号化動作が完了されると、前記メモリ1720は前記貯蔵された入力信号 $r(t)$ を乗算器1730に出力する。前記乗算器1730は前記マスクシーケンス発生器1710で発生される16種類のすべてのマスクシーケンス中、すでに使用されたマスクシーケンスを除外し、残ったマスクシーケンス中のいずれか一つを前記 $r(t)$ とかけて前記相関度計算器1740に出力する。前記相関度計算器1740は前記乗算器1730の出力と長さ32である64種類の相互直交符号との相関度を計算した後、最大相関値と、前記最大相関値に対応する相互直交符号のインデックスを出力する。前記相関度比較器1750は前記最大相関値と、前記最大相関値に対応する相互直交符号インデックスと、前記マスクシーケンス発生器1710から受信されたマスクシーケンスインデックスを貯蔵する。 10

【0113】

上述したような動作は前記マスクシーケンス発生器1710から発生される16種類のすべてのマスクシーケンスに対して遂行される。すると、前記相関度比較器1750には16種類の最大相関値と前記最大相関値に対応する相互直交符号のインデックスが貯蔵される。前記相関度比較器1750は上述した動作の16回反復遂行が完了されると、貯蔵されている16種類の相関値を比較して最大相関値を決定する。前記最大相関値が決定されると、前記相関度比較器1750は前記決定された最大相関値に対応する相互直交符号インデックスとマスクシーケンスインデックスを組み合わせるTFCIビットに出力する。 20
上述した動作により、TFCIビットの復号過程が終了すると、前記メモリ1720に貯蔵されていた入力信号 $r(t)$ は削除され、次の入力信号 $r(t+1)$ が貯蔵される。

【0114】

上述した前記図17に対する動作では相関度比較器1750は16回入力される最大相関値を一度に比較することに説明した。しかし、さらに他の方法には相関度比較を実時間に遂行することができる。即ち、最初に入力される最大相関値と次に入力される最大相関値を比較して、より大きな相関値とそれに対するマスクシーケンスインデックス及び相互直交符号インデックスを貯蔵する。次に最大相関値が入力されると、すでに貯蔵されている最大相関値と比較して、より大きな相関値を選択する。これによって所定相関値が選択されると、前記選択された相関値とこれに対応するマスクシーケンスインデックス及び相互直交符号インデックスを貯蔵する。このような動作を前記マスクシーケンス発生器1710で発生されるマスクシーケンスの総個数だけ反復遂行するようになる。前記マスクシーケンスの総個数に対応して上述した動作をすべて遂行するようになると、前記相関度比較器1750は最後に貯蔵されている相互直交符号インデックス(a6乃至a6)とマスクシーケンスインデックス(a7乃至a9)を組み合わせるTFCIビットに出力する。 30

【0115】

図10は図9の相関度比較器940の動作を示した流れ図である。前記相関度比較器940は16種類の最大相関値を貯蔵し、前記16種類の最大相関値中に一番大きな最大相関値を選択し、前記選択された最大相関値に対応する相互直交符号インデックスとマスクシーケンスインデックスにより入力情報ビット(TFCIビット)を出力する。前記16種類の最大相関値は比較され、前記TFCIビットは前記最大相関値に対応する相互直交符号インデックスとマスクシーケンスインデックスに基づいて出力される。 40

【0116】

前記図10を参照して説明すると、1000段階では最大相関インデックス i は1に、検索しようとする最大相関値、相互直交符号、マスクシーケンスのインデックスは0に初期化する。1010段階で前記相関度比較器940は相関度計算器920から1番目最大相関値、1番目相互直交符号インデックス、1番目マスクシーケンスインデックスを受信する。前記受信が完了されると、1020段階で前記相関度比較器940は前記1番目最大相関値と既存の最大相関値を比較する。前記1番目最大相関値が既存の最大相関値より大きいと、1030段階に進行し、前記1番目最大相関値が既存の最大相関値と同じであ 50

るか、小さいと1040段階に進行する。前記1030段階に進行すると、前記相関度比較器940は前記1番目最大相関値を最終最大相関値に指定し、1番目相互直交符号インデックスとマスクシーケンスインデックスをそれぞれ最終直交符号インデックスとマスクシーケンスインデックスとして貯蔵する。前記1040段階に進行すると、前記相関度比較器940は前記1番目最大相関値インデックスと相関度計算器の個数である16を比較して、16個のすべての相関値に対する比較が完了されたかを判断する。前記1040段階でiが16ではないと、1060段階に進行して前記iを1増加させた後、前記1010段階に進行して上述した過程を反復遂行する。

【0117】

上述したような反復遂行動作により16番目最大相関値、16番目相互直交符号インデックス、16番目マスクシーケンスインデックスに対する比較まで完了されると、1040段階で1050段階に進行する。前記1050段階に進行すると、前記相関度比較器940は前記変数である相互直交符号インデックスとマスクシーケンスインデックスに対応する復号ビット(入力情報ビット、TFCIビット)を出力する。前記復号ビットに対応する相互直交符号インデックスとマスクシーケンスインデックスは、16個の相関度計算器から受信された16種類の最大相関値の中、最終最大相関値に対応するインデックスである。

【0118】

3. 符号化／復号化装置及び方法の第2実施形態

上述した第1実施形態は電力制御の単位であるスロットの個数が16であるので、32シンボルのTFCI符号語を出力する(32、10)TFCI符号化器に対して説明した。最近、IMT-2000標準規格は一つのフレームが15スロットを有している。下記の第2実施形態ではスロットの個数が15であるので、30シンボルのTFCI符号語を出力する(30、10)TFCI符号化器である。従って、本発明の2実施形態は前記第1実施形態による(32、10)TFCI符号化器を通じて符号化された32個の符号化シンボルの中、2シンボルを穿孔して30シンボルを出力する符号化装置及び方法を提案する。

【0119】

本発明の第2実施形態による符号化装置の構造は上述した第1実施形態の符号化装置の構造と同一である。ただ、第2実施形態での1ビット発生器、基底ウォルシュ符号発生器及び基底マスクシーケンス発生器で出力される数列は、前記第1実施形態の数列の#0(第1シンボル)と#16(第17シンボル)のシンボルを穿孔することにより長さ30を有する。

【0120】

前記図8を参照して第2実施形態の符号化装置を説明すると、10ビットの入力情報ビット(TFCIビット)が入力されると、それぞれの入力情報ビットa0、a1、a2、a3、a4、a5、a6、a7、a8、a9は乗算器840、841、842、843、844、845、846、847、848、849にそれぞれ入力される。1ビット発生器800は常に1であるシンボルを乗算器840に出力する。前記乗算器840は前記1ビット発生器800から受信される32個のシンボルそれぞれに入力情報ビットa0を乗算する。基底ウォルシュ符号発生器810は長さ32の基底ウォルシュ符号W1、W2、W4、W8、W16を同時に発生する。前記乗算器841は前記基底ウォルシュ符号W1のシンボル(10101010101010101010101010101010)と前記入力情報ビットa1をかける。前記乗算器842は前記基底ウォルシュ符号W2のシンボル(01100110011001100110011001100110)と前記入力情報ビットa2をかける。前記乗算器843は前記基底ウォルシュ符号W4のシンボル(00011110000111100001111000011110)と前記入力情報ビットa3をかける。前記乗算器844は前記基底ウォルシュ符号W8のシンボル(00000001111110000000111110000000)と前記入力情報ビットa4をかける。前記乗算器845は前記基底ウォルシュ符号W16のシンボル(00000000000000000000111111111111)と前記入力情報ビットa5をかける。

10

20

30

40

50

【0121】

基底マスクシーケンス発生器820は長さ32の基底マスクシーケンスM1、M2、M3、M4を同時に発生する。前記乗算器846は基底マスクシーケンスM1のシンボル(010100001100011111000001110111)と入力情報ビットa6をかける。前記乗算器847は基底マスクシーケンスM2のシンボル(0000000111001101110110111000111)と入力情報ビットa7をかける。前記乗算器848は基底マスクシーケンスM4のシンボル(0001010111111001001101100101011)と入力情報ビットa8をかける。前記乗算器849は基底マスクシーケンスM8のシンボル(0011100001101110101011101010001)と入力情報ビットa9をかける。前記乗算器840乃至849は前記各基底ウォルシュ符号、各基底マスクシーケンス、1ビット発生器からのビットの発生及び出力を制御するスイッチのように機能する。

【0122】

前記加算器860は前記乗算器840、841、842、843、844、845、846、847、848、849の出力をシンボル単位に加算して32符号化シンボル(TFCI符号語)を出力する。前記32符号化シンボル中に二つのシンボルは所定位置に穿孔される(前記加算器860のシンボル#0(第1シンボル)及びシンボル#16(第17シンボル)は穿孔された。)。その以外の30シンボルは30TFCIシンボルになる。これは本発明の第2実施形態により容易に変更できる。例えば、1ビット発生器800、基底ウォルシュ発生器810、基底マスクシーケンス発生器820は#0及び#16シンボルを除いた30シンボルを発生することができる。前記加算器860は前記1ビット発生器800、基底ウォルシュ発生器810及び基底マスクシーケンス発生器820の出力をビット単位に加算して、TFCIシンボルとして30符号化されたシンボルを出力する。

【0123】

図12は本発明の第2実施形態によるスロットの個数が15である場合の符号化装置の動作に対する制御流れを示している図である。

図12を参照すると、1200段階で10ビットの入力情報ビットa0、a1、...、a9を入力し、変数sumとjを0に初期化する。1210段階で、変数jが30であるかを判断する。前記1210段階で前記jが30ではないと、1220段階で、基底ウォルシュ符号W1、W2、W4、W8、W16のj番目シンボルであるW1(j)、W2(j)、W4(j)、W8(j)、W16(j)と基底マスク符号M1、M2、M4、M8のj番目シンボルであるM1(j)、M2(j)、M4(j)、M8(j)を入力する。1230段階では前記1220段階で入力されたシンボルを前記入力情報ビットとシンボル単位にかけ、その結果値を全部合わせてsumを求める。1240段階では前記1230段階で求められたj番目符号化シンボルであるsumを出力し、1250段階でjを1だけ増加させた後、前記1210段階に進行して上述した動作を反復して遂行する。一方、上述した動作の反復遂行を通じて前記1210段階でjが30であると、すべての符号化動作を完了するようになる。

【0124】

前記(30、10)符号器はシンボル#0と#16が穿孔された前記(32、10)符号器の符号語に相当する1024個の符号語を出力する。従って、総符号語の数は1024個である。

【0125】

前記(30、9)符号器は前記1024個の符号語中、長さ32である32個の各ウォルシュ符号の#0と#16シンボルが穿孔された長さ30である32個のウォルシュ符号と、前記穿孔されたウォルシュ符号の各シンボルに1を加えた(実数の場合、-1をかけた)32個の相互直交符号と、前記4個の穿孔された基底マスクシーケンス中、任意の3個の基底マスクシーケンスの組み合わせにより求められる総8種類のマスクシーケンスとの組み合わせに示される。

【0126】

前記(30、8)符号器は前記1024個の符号語中、長さ32である32個の各ウォルシュ符号の#0と#16シンボルが穿孔された長さ30である32個のウォルシュ符号と、前記穿孔されたウォルシュ符号の各シンボルに1を加算した(実数の場合、-1をかけた)32個の相互直交符号と、前記4個の穿孔された基底マスクシーケンスの中、任意の2個の基底マスクシーケンスの組み合わせにより求められる総4種類のマスクシーケンスとの組み合わせに示される。

【0127】

前記(30、7)符号器は前記1024個の符号語中、長さ32である32個の各ウォルシュ符号の#0と#16シンボルが穿孔された長さ30である32個のウォルシュ符号と、前記穿孔されたウォルシュ符号の各シンボルに1を加算した(実数の場合、-1をかけた)32個の相互直交符号と、前記4個の穿孔された基底マスクシーケンスの中、1個の基底マスクシーケンスとの組み合わせに示される。

10

【0128】

一方、拡張型TFCIをサービスするために上述した符号器はすべて最小距離が10である。前記拡張型TFCIをサービスするため上述した(30、9)、(30、8)、(30、7)符号器は、前記図8の基底マスクシーケンス発生器820からの4種類基底マスクシーケンス中、少なくとも一つの基底マスクシーケンスに対する入力と出力を遮断することにより具現されることができる。

【0129】

前記符号器は入力情報ビット(TFCIビット)数に応じて流動的に符号化が可能であり、符号器の性能を左右する最小距離を最大限高めることにより優秀な性能を有するようになる。

20

【0130】

一方、本発明の第2実施形態による復号化装置は、符号化されたシンボルの信号長さが異なるだけで、上述した第1実施形態で説明された復号化装置と同一構成(図9参照)を有し、その動作も同一である。即ち、上述した第1実施形態による復号化装置は長さ32のウォルシュ符号とマスクシーケンスを使用することに対して、第2実施形態では長さ30のウォルシュ符号とマスクシーケンスを使用する。従って、第2実施形態による復号化装置を構成するマスクシーケンス発生器は0番目シンボルと16番目シンボルが穿孔された15種類のマスクシーケンスを発生すべきである。また、相関度計算器は相関度を計算することにおいて、0番目シンボルと16番目シンボルが穿孔された64個のウォルシュ符号を使用すべきである。

30

【0131】

本発明による第2実施形態の復号化装置を構成する相関度比較器で遂行する段階も、上述した第1実施形態で図10を参照して詳細に説明したものと同一であるので、追加的な説明は省略する。

【0132】

4. 符号化／復号化装置及び方法の第3実施形態

上述した第2実施形態は第1実施形態で使用しているウォルシュ符号とマスクシーケンスの#0シンボルと#16シンボルを穿孔した長さ30のウォルシュ符号とマスクシーケンスを使用する符号化装置を示した。また、前記第2実施形態の符号化装置は4個の基底マスクシーケンスの使用を制御することにより、(30、7)、(30、8)、(30、9)、(30、10)符号器に使用できるようにしている。

40

【0133】

本発明の第3実施形態は前記第2実施形態の(30、7)、(30、8)、(30、9)、(30、10)(以下、(30、7-10))符号器で1ビット発生器の出力を遮断する符号化装置を提供する。また、この代わりさらに他のマスクシーケンスを発生させることにより、最小距離が11になるようにする符号化装置を提供する。前記(30、7-10)符号器は7乃至10ビットの入力情報ビット(TFCIビット)を入力として30シンボルのTF CI符号語を出力する符号器を意味する。

50

【0134】

図14は本発明の第3実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するため符号化装置の構成を示している図である。即ち、前記図14の(30、7-10)符号器は、最小距離が11になるように設計された。

【0135】

前記第3実施形態の符号化装置の構造は、上述した第2実施形態の符号化装置の構造と類似である。ただ、本発明の第3実施形態による符号化装置は、基底マスクシーケンスM16を発生するマスクシーケンス発生器1480と、前記マスクシーケンス発生器1480と1ビット発生器1400を乗算器1440にスイッチングするスイッチ1470をさらに設けている。

10

【0136】

前記図14で使用する2ビット穿孔された基底マスクシーケンスM1、M2、M4、M8、M16は次のようである。

M1 = 000001011111000010110100111110
 M2 = 000110001100110001111010110111
 M4 = 010111100111101010000001100111
 M8 = 011011001000001111011100001111
 M16 = 100100011110011111000101010011

【0137】

前記図14を参照して説明すると、(30、6)符号器が使用される場合には、スイッチ1470は前記1ビット発生器1400を乗算器1440に連結し、基底マスク発生器1480で発生するすべての基底マスクシーケンスの使用を遮断する。一方、前記1ビット発生器1400で発生されるシンボルは乗算器1440により入力情報ビットa0とシンボル単位にかけられて出力される。

20

【0138】

これに対して、前記(30、7-10)符号器が使用される場合には、スイッチ1470はマスクシーケンス発生器1480と乗算器1440を連結し、前記基底マスクシーケンス発生器1420から出力される4種類の基底マスクシーケンスを選択的に使用する。この場合には5種類の基底マスクシーケンスの組み合わせにより31個のマスクシーケンスM1、M2、M3、M4、M5、...、M29、M30、M31が発生され得る。

30

【0139】

一方、乗算器1440乃至1449を使用して入力情報ビット(a0乃至a9)を符号化シンボルに符号化して出力する構成及び動作は、上述した第1及び第2実施形態と同一であるので、詳細な説明は省略する。

【0140】

上述したように、(30、7-10)符号器をサービスするためには、スイッチ1470はマスクシーケンス発生器1480を乗算器1440に連結すべきであり、(30、6)符号器をサービスするためには、スイッチ1470は1ビット発生器1400を乗算器1440に連結すべきである。

【0141】

(30、6)符号器の出力は、6ビットの入力情報ビットを入力として長さ30である32個のウォルシュ符号と前記1ビット発生器1400の出力により前記ウォルシュ符号が反転された32個の相互ウォルシュ符号の組み合わせに示される。

40

【0142】

(30、10)符号器の出力は、10ビットの入力情報ビットを入力として長さ30である32個のウォルシュ符号と5個の基底マスクシーケンスにより発生される32個のマスクシーケンスの組み合わせに示される。この時、前記5個の基底マスクシーケンスは、上述したようにM1、M2、M4、M8、M16であり、前記M16は本発明の第3実施形態による符号化装置に追加されたマスクシーケンス発生器1480から出力されるマスクシーケンスである。従って、前記(30、10)符号器から得られる総符号語の数は102

50

4である。(30、9)符号器の出力は、9ビットの入力情報ビットを入力として前記1024個の符号語中、長さ30である32個のウォルシュ符号と16種類のマスクシーケンスの組み合わせに示される。前記16種類のマスクシーケンスは5個の基底マスクシーケンスの中、4個の基底マスクシーケンスの組み合わせに求められる。(30、8)符号器の出力は8ビットの入力情報ビットを入力として前記1024個の符号語の中、長さ30である32個のウォルシュ符号と8種類のマスクシーケンスの組み合わせに示される。前記8種類のマスクシーケンスは5個の基底マスクシーケンスの中、3個の基底マスクシーケンスの組み合わせに求められる。(30、7)符号器の出力は7ビットの入力情報ビットを入力として前記1024個の符号語の中、長さ30である32個のウォルシュ符号と4種類のマスクシーケンスの組み合わせに示される。前記4種類のマスクシーケンスは5個の基底マスクシーケンスの中、2個の基底マスクシーケンスの組み合わせに求められる。

【0143】

一方、拡張型TFCIをサービスするために、上述した(30、7-10)符号器はすべて最小距離が11である。前記拡張型TFCIをサービスするために、上述した(32、7-10)符号器は図14の基底マスクシーケンス発生器1420とマスクシーケンス発生器1480からの5種類の基底マスクシーケンスの中、少なくとも一つの基底マスクシーケンスに対する使用を制御することにより具現されることができる。

【0144】

図16は本発明の第3実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するための制御流れを示している図である。

【0145】

前記図16を参照すると、先ず1600段階で10ビットの入力情報ビット(TFCIビット)a0乃至a9を入力し、変数sumとjを0に初期化する。前記変数sumはシンボル単位に加算された後、出力される最終符号化シンボルを示す変数であり、前記変数jはシンボル単位に加算された後、出力される最終符号化シンボルの個数をカウントするための変数である。前記1600段階が終了されると、1610段階に進行して変数jが30であるかを判断する。前記変数jが30であるかを判断するのは、前記入力情報ビットの符号化のため使用される穿孔されたウォルシュ符号及びマスクシーケンスの長さが30シンボルであるためである。即ち、1610段階を遂行するのは、前記入力情報ビットがウォルシュ符号及びマスクシーケンスの30シンボルそれぞれに対応して符号化されたかを判断するためである。

【0146】

前記1610段階でjが30ではないと、ウォルシュ符号及びマスクシーケンスのすべてのシンボルに対応した符号化動作が遂行されなかったと判断して1620段階に進行する。前記1620段階に進行すると、基底ウォルシュ符号W1、W2、W4、W8、W16のj番目シンボルであるW1(j)、W2(j)、W4(j)、W8(j)、W16(j)と基底マスクシーケンスM1、M2、M4、M8、M16のj番目シンボルであるM1(j)、M2(j)、M4(j)、M8(j)、M16(j)を入力する。1630段階では前記入力されたシンボルを前記入力情報ビット(TFCIビット)とシンボル単位にかけた後、その結果値を全部合わせてsumを求める。

【0147】

前記1630段階を下記式(10)のように表現することができる。

【0148】

式(10)

$$\begin{aligned} \text{sum} = & a_0 + M_{16}(j) + a_1 W_1(j) + a_2 \cdot W_2(j) + a_3 \cdot W_4(j) + a_5 \cdot W_{16}(j) \\ & + a_6 \cdot M_1(j) + a_7 \cdot M_2(j) + a_8 \cdot M_4(j) + a_9 \cdot M_8(j) \\ & \dots \dots (10) \end{aligned}$$

【0149】

前記式(10)から分かるように、各入力情報ビットは対応する基底ウォルシュ符号、または基底マスクシーケンスのシンボルとかけられて、前記かけられた結果を加算するこ

とにより所望する符号化シンボルを得るようになる。

【0150】

1640段階では前記1630段階で求められたj番目符号化シンボルであるsumを出力し、1650段階に進行してjを1だけ増加させた後、1610段階を進行して上述した動作を反復して遂行する。一方、1610段階でjが30であると、すべての符号化動作を完了するようになる。

【0151】

次に、図15を参照して本発明の第3実施形態による復号化装置を説明する。送信側により伝送された30符号化されたシンボルを含む入力信号 $r(t)$ と、前記符号器により穿孔された位置に挿入された二つのダミーシンボルは、31個の乗算器1502、1504、...、1506と相関度計算器1520に入力される。マスクシーケンス発生器1500は長さ32である31個のすべての場合のマスクシーケンスM1、M2、...、M31を生成して前記乗算器1502、1504、...、1506に出力する。従って、前記乗算器1502、1504、...、1506それぞれは前記マスクシーケンス発生器1500から受信されたマスクシーケンスと前記入力信号 $r(t)$ をかけて出力する。従って、送信側で所定マスクシーケンスを使用してTFCIビットを符号化すると、前記乗算器1502、1504、...、1506の出力の中のいずれか一つは、マスクシーケンスが除去された信号である。例えば、送信側でM31を使用してTFCIビットを符号化すると、M31と入力信号 $r(t)$ をかける乗算器1506の出力が、マスクシーケンスが除去された信号である。しかし、送信側でマスクシーケンスを使用しない場合には、相関度計算器1520に印加される入力信号 $r(t)$ 自体がマスクシーケンスの除去された信号である。前記相関度計算器1520、1522、1524、...、1526は入力信号 $r(t)$ 及び前記乗算器1502、1504、...、1506からの出力に構成された32個の信号を受信して、長さ32である64個の相互直交符号との相関度を計算する。一方、前記相関度計算器1520、1522、1524、...、1526は、前記計算した64種類の相関値の中、一番大きな相関値を決定する。前記相関値が決定されると、前記決定された相関値と、前記決定された相関値に対応する相互直交符号のインデックス及び自分の固有インデックスを相関度比較器1540に出力する。

【0152】

前記相関度比較器1540は前記それぞれの相関度計算器1520乃至1526から受信された32種類の最大相関値を比較して、その中で最大の相関値を最終の最大相関値として決定する。すると、前記相関度比較器1540は前記最終最大値に対応する相互直交符号及びマスクシーケンスのインデックスにより送信側から伝送される符号化されたTFCIビットを出力する。図17のように、本発明の前記第3実施形態は相互直交符号の相関値を計算する単一相関値計算器と、前記マスクと $r(t)$ をかける単一乗算器により具現されることもできる。

【0153】

〔発明の効果〕

上述したように本発明は基本型TFCIと拡張型TFCIを可変的にサービスする符号化／復号化装置及び方法を提案することにより、ハードウェアを簡単化させる利点と、前記基本型TFCIと拡張型TFCIに対する誤り訂正符号化方式をすべて支援することにより、サービスの安定性を向上させる利点がある。また、符号化装置の性能を決定する最小距離を符号分割多重接続移動通信システムで要求される条件に満足するようにすることにより、優秀な性能を有するようにする効果がある。

【0154】

以上、本発明の実施形態を添付図面を参照して説明したが、本発明はこの特定の実施形態に限るものでなく、各種の変形及び修正が本発明の範囲を逸脱しない限り、該当分野における通常の知識を持つ者により可能なのは明らかである。

【図面の簡単な説明】

【0155】

【図 1 A】 通常的な符号分割多重接続移動通信システムで伝送率情報の使用例を示すためのフレーム構造を示した図である。

【図 1 B】 通常的な符号分割多重接続移動通信システムで伝送率情報の使用例を示すためのフレーム構造を示した図である。

【図 1 C】 通常的な符号分割多重接続移動通信システムで伝送率情報の使用例を示すためのフレーム構造を示した図である。

【図 1 D】 通常的な符号分割多重接続移動通信システムで伝送率情報の使用例を示すためのフレーム構造を示した図である。

【図 2】 通常的な符号分割多重接続移動通信システムの基地局送信器構造を示している図である。

【図 3】 通常的な符号分割多重接続移動通信システムの移動局送信器の構造を示している図である。

【図 4 A】 従来の符号分割多重接続移動通信システムで基本的な伝送率情報を符号化するための構成を概念的に示している図である。

【図 4 B】 図 4 A の相互直交符号器に適用される符号化テーブルの一例を示している図である。

【図 5 A】 従来の符号分割多重接続移動通信システムで拡張された伝送率情報を符号化するための構成を概念的に示している図である。

【図 5 B】 図 5 A の制御器が伝送率情報を分配するため遂行するアルゴリズムの一例を示している図である。

【図 5 C】 図 5 A の相互直交符号器に適用される符号化テーブルの一例を示している図である。

【図 6】 本発明の一実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するための構成を概念的に示している図である。

【図 7】 本発明の一実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するため要求されるマスクシーケンスを生成する制御流れを示している図である。

【図 8】 本発明の一実施形態による符号分割多重接続移動通信システムで伝送率情報符号化装置の構成を示している図である。

【図 9】 本発明の一実施形態による符号分割多重接続移動通信システムで伝送率情報復号化装置の構成を示している図である。

【図 10】 図 9 の相関度比較器で遂行する制御流れを示している図である。

【図 11】 本発明の一実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するための制御流れを示している図である。

【図 12】 本発明の他の実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するための制御流れを示している図である。

【図 13】 本発明の一実施形態による伝送率情報により決定される直交シーケンス及びマスクシーケンスの構造を示している図である。

【図 14】 本発明の実施形態による符号分割多重接続移動通信システムで伝送率情報符号化装置の他の例を示している図である。

【図 15】 本発明の実施形態による符号分割多重接続移動通信システムで伝送率情報復号化装置の他の例を示している図である。

【図 16】 本発明の他の実施形態による符号分割多重接続移動通信システムで伝送率情報を符号化するための制御流れの例を示している図である。

【図 17】 本発明のさらに他の実施形態による符号分割多重接続移動通信システムで伝送率情報復号化装置の例を示している図である。

【符号の説明】

【0156】

602 マスクシーケンス発生部

604 相互直交シーケンス発生部

610 加算器

10

20

30

40

50

902、904、...、906、1502、1504、...、1506、1730
乗算器

910 マスクシーケンス発生器

920、922、924、926、1520、1522、1524、...、1526

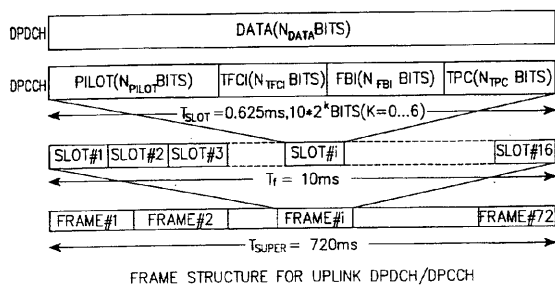
、1740 相関度計算器

940、1540、1750 相関度比較器

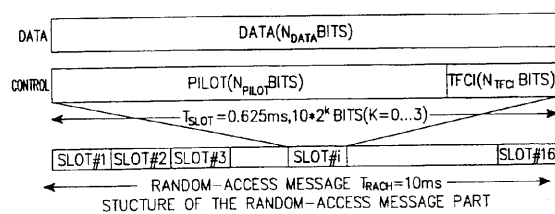
1500、1710 マスクシーケンス発生器

1720 メモリ

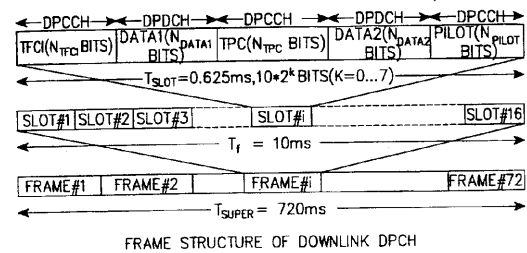
【図1A】



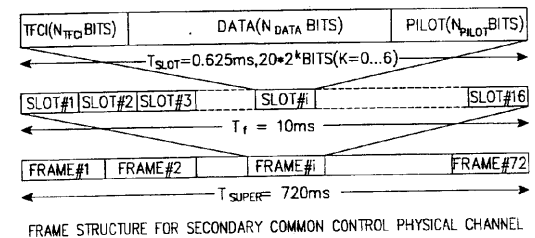
【図1B】



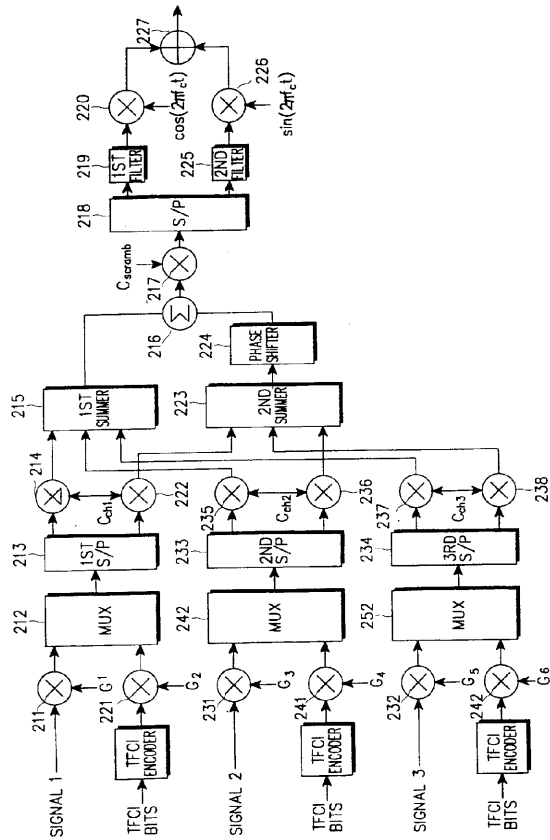
【図1C】



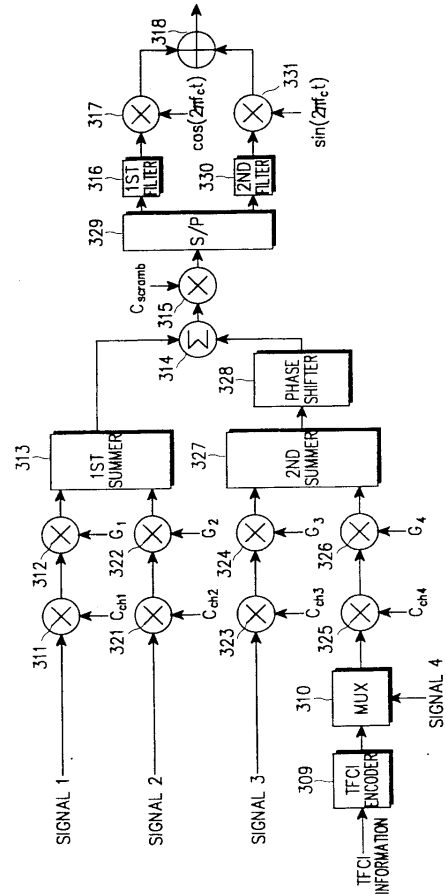
【図1D】



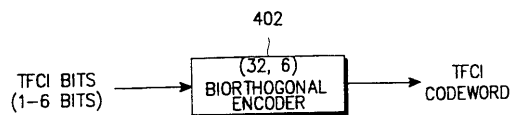
【図 2】



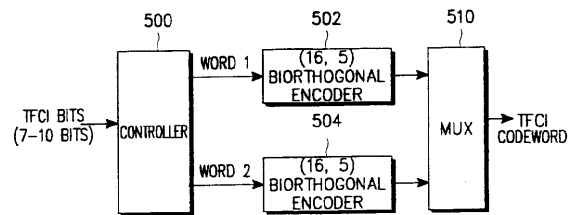
【図 3】



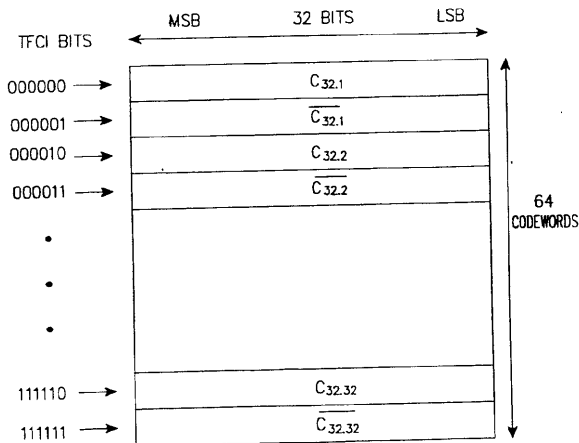
【図 4 A】



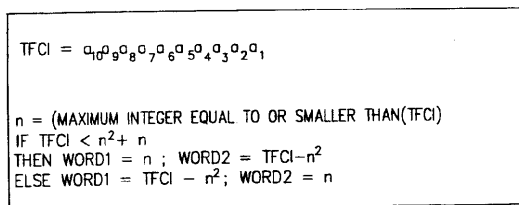
【図 5 A】



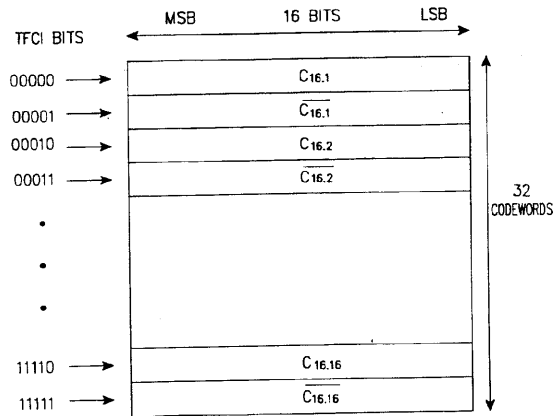
【図 4 B】



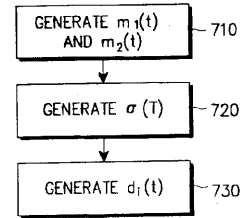
【図 5 B】



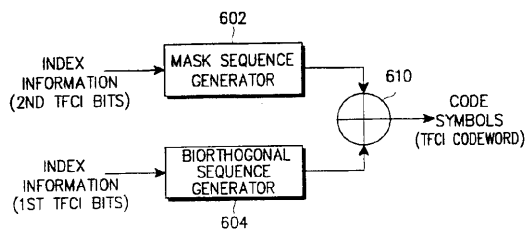
【図 5 C】



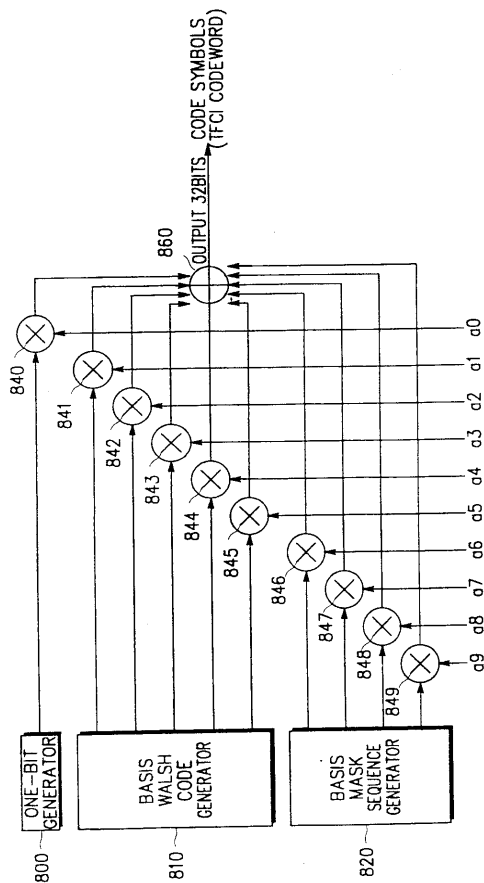
【図 7】



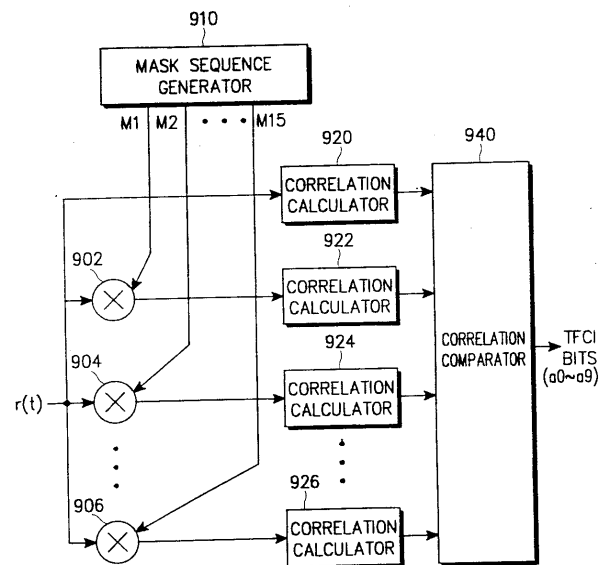
【図 6】



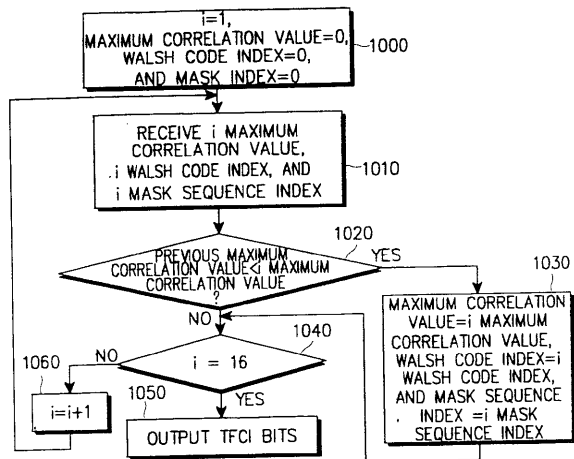
【図 8】



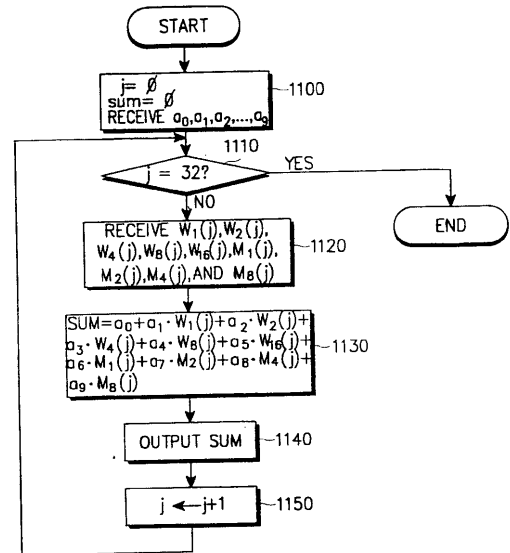
【図 9】



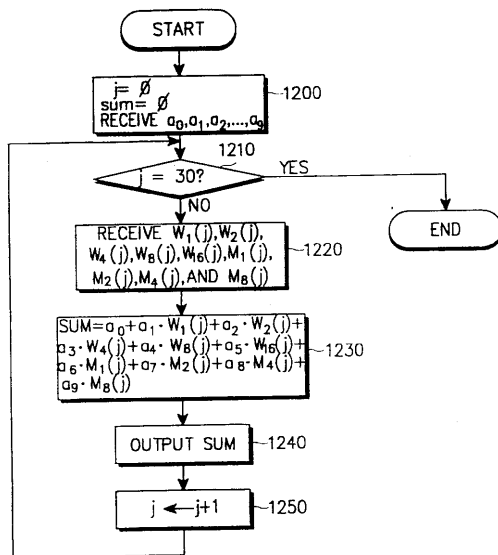
【図 10】



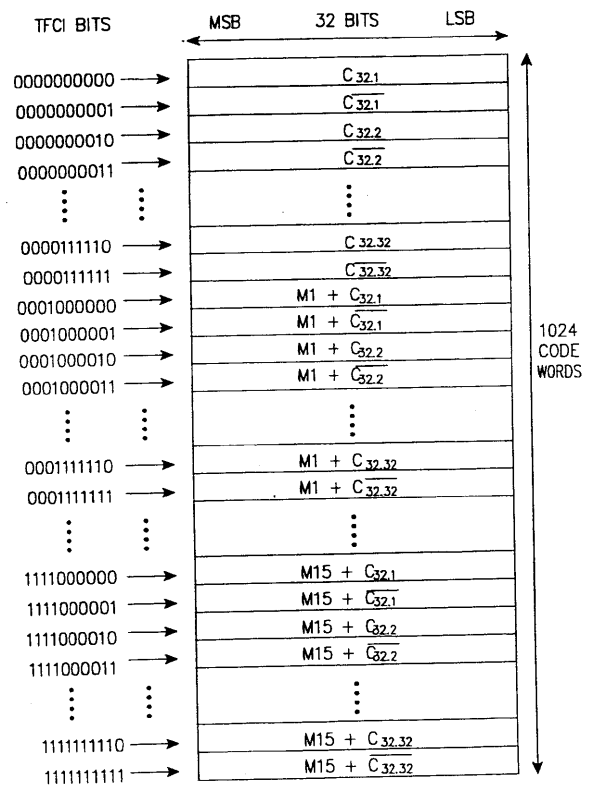
【図 11】



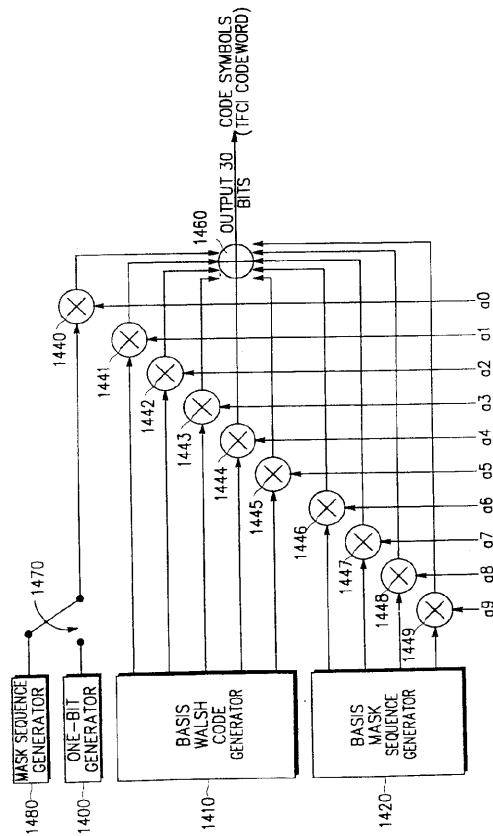
【図 12】



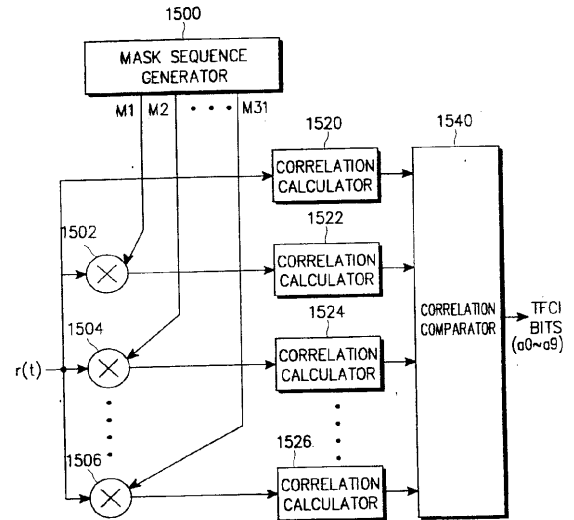
【図 13】



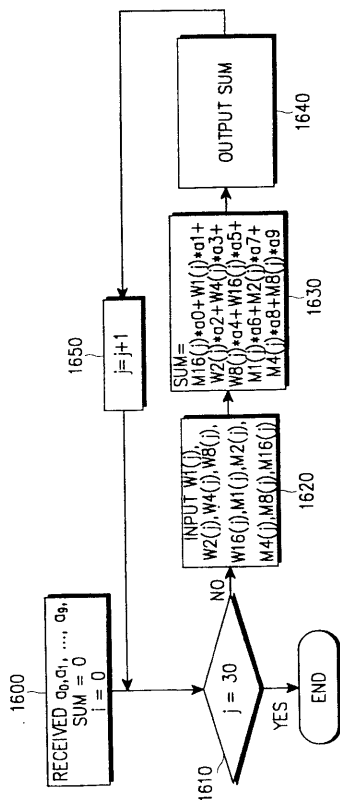
【図 14】



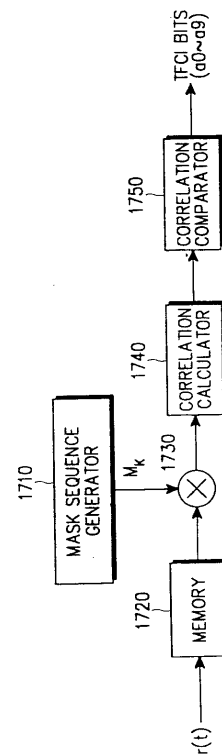
【図 15】



【図 16】



【図 17】



フロントページの続き

(72)発明者 ヒーウォン・カン

大韓民国・ソウル・131-207・チュンナング・ミョンモク・7ードン・1499

Fターム(参考) 5K022 EE02 EE22 EE25