

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3614907号

(P3614907)

(45) 発行日 平成17年1月26日(2005.1.26)

(24) 登録日 平成16年11月12日(2004.11.12)

(51) Int. Cl.⁷

F I

H O 4 L 12/56

H O 4 L 12/56

Z

H O 4 L 1/16

H O 4 L 1/16

H O 4 L 1/18

H O 4 L 1/18

請求項の数 4 (全 69 頁)

(21) 出願番号	特願平6-326911	(73) 特許権者	000003078
(22) 出願日	平成6年12月28日(1994.12.28)		株式会社東芝
(65) 公開番号	特開平8-186570		東京都港区芝浦一丁目1番1号
(43) 公開日	平成8年7月16日(1996.7.16)	(74) 代理人	100058479
審査請求日	平成13年12月25日(2001.12.25)		弁理士 鈴江 武彦
		(72) 発明者	金井 久美子
			神奈川県川崎市幸区小向東芝町1番地 株
			式会社東芝研究開発センター内
		(72) 発明者	角田 啓治
			神奈川県川崎市幸区小向東芝町1番地 株
			式会社東芝研究開発センター内
		審査官	清水 稔

最終頁に続く

(54) 【発明の名称】 データ再送制御方法及びデータ再送制御システム

(57) 【特許請求の範囲】

【請求項1】

F E C による誤り訂正機能および再送による誤り訂正機能を用いた通信におけるデータ再送制御方法であって、

送信端末において、受信端末から、該受信端末にて F E C による誤り訂正機能によっては訂正できない誤りを伴って受信された誤りのある通信データに対する再送を要求する再送要求を受信する第1の手順と、

前記第1の手順にて受信された前記再送要求および F E C による前記誤り訂正機能の誤り訂正能力に基づいて、前記誤りのある通信データを、前記 F E C による誤り訂正機能および前記再送による誤り訂正機能を用いることによってリカバーするのに必要な再送データの最小量を決定する第2の手順と、

前記第2の手順にて決定された前記最小量の再送データを、前記送信端末から前記受信端末へ再送する第3の手順とを有することを特徴とするデータ再送制御方法。

【請求項2】

前記通信は、前記送信端末から複数の受信端末へのマルチキャスト通信であり、

前記第1の手順では、送信端末において、少なくとも一つの受信端末から前記再送要求を受信し、

前記第3の手順では、前記最小量の再送データを、前記送信端末から前記少なくとも一つの受信端末へ再送することを特徴とする請求項1に記載のデータ再送制御方法。

【請求項3】

10

20

F E Cによる誤り訂正機能および再送による誤り訂正機能を用いた通信におけるデータ再送制御システムであって、

送信端末において、受信端末から、該受信端末にてF E Cによる誤り訂正機能によっては訂正できない誤りを伴って受信された誤りのある通信データに対する再送を要求する再送要求を受信する第1の手段と、

前記第1の手段にて受信された前記再送要求およびF E Cによる前記誤り訂正機能の誤り訂正能力に基づいて、前記誤りのある通信データを、前記F E Cによる誤り訂正機能および前記再送による誤り訂正機能を用いることによってリカバーするのに必要な再送データの最小量を決定する第2の手段と、

前記第2の手段にて決定された前記最小量の再送データを、前記送信端末から前記受信端末へ再送する第3の手段とを有することを特徴とするデータ再送制御システム。

10

【請求項4】

前記通信は、前記送信端末から複数の受信端末へのマルチキャスト通信であり、

前記第1の手段では、送信端末において、少なくとも一つの受信端末から前記再送要求を受信し、

前記第3の手段では、前記最小量の再送データを、前記送信端末から前記少なくとも一つの受信端末へ再送することを特徴とする請求項3に記載のデータ再送制御システム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】

20

本発明は、F E Cによる誤り訂正機能及び再送による誤り訂正機能を用いた通信におけるデータ再送制御方法及びデータ再送制御システムに関する。

【0002】

【従来の技術】

多種多様な情報をA T Mセルを用いた通信形態に適應させるには、A T M網と上位レイヤとのインタワークが必要であり、その機能を備えたA A LがA T Mレイヤと上位レイヤの間に存在する。

【0003】

図31に、A T Mアダプテーションレイヤの一例としてA A Lタイプ5の位置づけを示す。タイプ5の場合、A A Lは大きく分けてC Sレイヤ(Convergence Sub layer)とS A Rレイヤ(Segmentation and Reassemble)の2つのサブレイヤからなっている。C Sレイヤはさらに2つのサブレイヤからなっており、C P C Sレイヤ(Common Part C S)はサービスの種類に係わらず必要となる処理を、また、S S C Sレイヤ(Service Specific C S)はサービスに依存して行われる処理を受け持っている。従って、S S C Sレイヤがヌルの場合も有る。S A Rレイヤは、主にデータの分解/組立を行っており、送信時にはA T MレイヤでA T Mヘッダを付与することができるサイズ、つまりペイロードサイズである48オクテットにデータを区切って下位レイヤに渡す処理を行う。それとは反対に、受信時には下位レイヤからペイロードサイズのデータを受け取り、C S - P D Uに組み立ててC Sレイヤに渡す。

30

40

【0004】

現在までに、A A Lは扱う情報の種類に応じて4つのタイプがI T U - Tなどにより標準化されている。具体的には、A A L 1は固定ビットレートの音声・画像などのリアルタイム通信、A A L 2は可変ビットレートのリアルタイム通信、A A L 3/4と5は可変データ通信用(但し、A A L 3と5はコネクションオリエンテッド(C O)通信用、A A L 4はコネクションレス(C L))となっている。

【0005】

この中でも、特に、A A L 3/4と5はデータ通信用であることから、わずか1ビットのビット誤りでさえ許容されないようなデータの正確さ、換言すると受信したデータが送信したデータと完全に同一のものであるということを保証できるような信頼性が要求される

50

アプリケーションの存在が考えられる。また、AAL1、2といったリアルタイムアプリケーションで高いQOSが要求されるような場合にも、データ自体の信頼性が必要となってくることは言うまでもない。

【0006】

しかし、現状ではATM技術についてはプリミティブな部分の標準化に留まっており、AALにて誤り制御機能が十分には提供されていない。このため、正確さが要求されるデータ通信をAAL3/4、あるいはAAL5のユーザが求める場合、例えばAAL5であれば、標準の決まっていないCPCSレイヤよりも上位のレイヤに、信頼性を保証させるプロトコルを実装する必要がある。AAL5のCPCSレイヤ以下では仕様が固まっており、すでに標準化されているため、サービスに依存した新しい機能を付加する場合にはCPCSレイヤよりも上位のレイヤでなければならない。

10

【0007】

これまで知られている例としては、SSCSレイヤ(Service Specific Convergence Sublayer; サービス依存CSレイヤ)に再送制御を基本としたプロトコルを実装する方法(例えば、ITU-TのQ.SAALプロトコル)や、SSCSレイヤをヌルとし、その上位にトランスポートレイヤプロトコル(例えば、TCPやOSI/TP4など)を実装する方法などがある。

【0008】

上記のような方法は、全て再送によりその信頼性を確保する方法である。

【0009】

しかしながら、(1)大容量トランザクションなど、大容量で実時間性が求められ(すなわち再送制御が基本的に許されない)、かつ信頼性も要求されるアプリケーションが今後ますます増えていくと考えられる。

20

【0010】

また、(2)LAN、MAN、WANなど、通信網の広域化がなされている。通信網が広域であればあるほど、上記のレイテンシの確保と、再送による誤り制御とは相いれないものとなってくる(情報の再送には莫大な時間がかかる)。

【0011】

また、(3)一般に、再送による誤り制御を行う方式においては、再送の単位は上位レイヤパケット群(具体的には、例えばTCPパケットや、レイヤ4パケットグループ)であり、セルの廃棄といった比較的小さな単位でのデータ消失に対し、大きなデータの再送を求めるものとなっており、網資源の浪費につながる場合がある。

30

【0012】

このような理由により、上述した再送を基本とする誤り制御方法をATM端末上を実装するのみでは対処が難しいデータ通信領域が、今後ますます増大していくことが考えられる。特に、リアルタイム性が重視される場合には、リアルタイム性の確保が必須であり、再送を基本とする誤り制御方法による対処では極めて難しい。

【0013】

そこで、1994年電子情報通信学会春季大会にて本発明者らは、“B-786 ATM網における誤り制御方式、FEC(Forward Error Correction) on AAL Type 5”と題してFECをAAL5に実装する方法を提案している。この方式は、送信するデータに対して誤り訂正符号を付与して通信を行い、網内にて多少の情報の廃棄や誤りが生じた場合でも受信側において消失情報の再生を行うという、信頼性のある通信を行うための別のアプローチを適用したものである。より具体的には、AAL1のCSレイヤにおけるオプションとして用意されている誤り訂正符号と同様の符号をAAL5のSSCSレイヤ以上の部分に適用するものである。AAL1では、送信側から受信側へ音声・映像などのリアルタイム通信を行うため、再送が現実的にできないことから、該データに誤り制御符号を付与して、伝送中のエラーに対する耐性を強くするという工夫が行なわれている。この方法と同様のインタリーブ方式を用いて、FEC冗長コードをAAL5のデータに対して付加することによって(再送を必要としない)リアル

40

50

タイム通信や、信頼性のある通信を保証することができる。

【0014】

ここでAAL3/4でなく、AAL5を用いた理由としては、例えば以下の点があげられる。

【0015】

(1) 次世代通信方式としてATM方式が本命視される中、ITU-TやATMフォーラムなどの標準化機関により標準化が進められているデータ通信用AAL(具体的にはAAL5)が、今後の高速端末(マルチメディア端末を含む)に実装されていく可能性が極めて高いこと。

【0016】

(2) AAL5は、高スループット、低レイテンシを目的としていること。

【0017】

ただし、上記方式をタイプ5以外のAALにも実装可能であることはもちろんのことである。

【0018】

この実装にあたって、AAL1で行われているようなFECを行うためには、送信側において、上位レイヤから渡されたデータを一度マトリクス状のインタリーバに書き込んで、冗長用の符号を付与した後に転送する必要がある。また、受信側では到着すべきであったが廃棄されてしまったり、到着が許容範囲以上に遅くなってしまったデータ(セル)が送信時にインタリーバのどの位置にあったかを特定する必要がある。AAL1では、デフォルトの処理としてSARレイヤでセルにシーケンスナンバ(SN)を付与することになっているので、オプションとしてFECを行う場合にもこれを用いることができ、必然的に誤り位置の特定を行うことができた。一方、AAL5には位置を特定するための処理は行われていない。そこで、FEC冗長符号およびSNの付与を行う機能を、現状ではヌルであるSSCSレイヤに持たせることを提案した。

【0019】

ところで、上記学会発表にて想定しているインタリーバはAAL1で用いられているものと同じであり、データを書き込めるサイズは固定であった。AAL1では、転送するデータが音声等のCBR(Constant Bit Rate; 固定ビットレート)であり、一定の割合で連続して上位レイヤから渡されることが予めわかっているため、大きめの固定サイズのインタリーバでも問題はなかった。しかしながら、データ転送においては、どのようなタイミングで、どのような大きさのデータが上位レイヤから渡されるかわからない。よって、大きさの固定されたインタリーバでは、インタリーバのサイズよりもかなり小さいデータを転送する場合には、残りの領域にダミーデータをパディングしてダミーセルを送出しなければならない。データのサイズをインタリーバのサイズで割ったときの剰りが小さい場合、ダミーデータが増えてしまうので、特に問題になる。

【0020】

したがって、このような方法では、帯域を有効に使うことができず、余分なトラフィックを増やしてしまうことになりかねない。加えて、従来のAAL1と異なり、ダミーデータがインタリーバ毎に入るときには、どこからがダミーデータであるかを示すための表示部分がインタリーバ毎に必要なのであるため、この部分を受信側で的確に取り出せるための機構も必要となる。

【0021】

また、既存のインタリーバは、上位レイヤから渡されたデータを一度メモリに書き込んでから読み出しを行っているため、書き込みと読み出しの方向が垂直になる場合には、レイテンシが少なくとも1つのインタリーバに書き込むのに必要な時間だけかかった。例えば、ITU-T勧告I.363においてAAL1のオプションとなっているインタリーバ1つあたりのサイズはデータ領域で $47 \times 124 = 5828$ オクテットあり、この領域に対して最初の列にデータを書き込み始めてから最終列の書き込みに入る所までの間、下位レイヤに対しての送出手は全くできないことになる。

10

20

30

40

50

【 0 0 2 2 】

他のインタリーブ方式としては、最近、ITU-T, SG XIII, Working Document TD/27 (Rev.) (13-2)-E, March, 1994 at Geneva, "Status Report on AAL1/2 for video signal transport"にあるように、斜めインタリーブ（読み出しと書き込みの方向が斜めに交差する）という手法が提案されている。しかしながら、この方法を用いても、垂直に交差する場合と比べてレイテンシは半分にしかないという問題があった。

【 0 0 2 3 】

この遅延への対策として、松崎らにより示される文献（1994年春 電子情報通信学会 春季全国大会 B-853；「ATMセルとMPEG-2システム規格との整合性を考慮した誤り訂正手法の検討」）において、データの読み出しと書き込みが同じ方向であるかのごときインタリーブが図示されており、そのように書き込みと読み出しを行えば上記のような遅延自体は改善される。当該図に対しては何ら説明が与えられていないが、一般的に該インタリーブへはデータを書き込み、その後読み出すという操作が必要であり、またその読み書きのためのメモリ領域もインタリーブの大きさが必要である。すなわち、この方法においても、なお、インタリーブの大きなメモリ領域の確保が必要であるとともに、該メモリへの読み書きのための遅延が回避できないという問題があった。

【 0 0 2 4 】

また、受信側で受信するデータにFEC冗長符号が付与されている場合、既存の方法（AAL1等）では、従来送信時にFEC冗長符号を付与するのと同程度のあるいはそれ以上の遅延がすべてのインタリーブ処理で生じてしまうという問題もあった。それは、受信時には到着したデータを、データ領域かFEC冗長領域かにかかわらず、全て順次インタリーブ内に格納していくという処理が必要であったからである。AAL1では、セル廃棄をSNで検出することは可能であるが、データ内のビット誤りを検出する機能がFEC冗長符号にしかないため、データが正しいことを確認するには、受信側は全てのデータに対してFECによる復号処理を行わなければならない。これらの処理に要する遅延は、ビットエラーがセル廃棄と比較して極めてまれであってほとんどのデータが正しいような場合には、特に無駄が大きくなってしまっていた。

【 0 0 2 5 】

【 発明が解決しようとする課題 】

以上述べたように、従来のFECによる誤り訂正は、本来CBR伝送等に適切な方式であって、転送すべきデータが連続的に上位レイヤから到着することを前提としている。よって、特にAAL3/4、5のようなデータ転送を行うプロトコルに、従来のようなFECによる誤り訂正を実装するのは不向きであった。すなわち、AAL1で採用されている固定サイズのインタリーブでは、送りたいデータがそのサイズと比べて小さい場合には、データでインタリーブのデータ領域を満たすことができないので、残りの領域にはダミーデータを入れてダミーセルを送出するしかなかった。このような方法では、帯域を有効に使うことができず、余分なトラフィックを増やしてしまい、輻輳を招くことになりかねない。加えて、どこからがダミーデータであるかをインタリーブ毎に明確にし、それを受信側にわかるように伝送する機能が必要となるが、従来のAAL1のインタリーブでは全く考えられていなかった。

【 0 0 2 6 】

また、既存の方法ではインタリーブに対して書き込みと読み出しの方向が異なるために、メモリに書き込んでいる間、メモリからの読み出しを待つことで遅延が存在する。この遅延はインタリーブの大きさにおよそ比例するものであり、比較的低遅延であると言われている斜めインタリーブの方法を用いても遅延の大きさを半分にすることしかできない。

【 0 0 2 7 】

さらに、これへの解決法と考えられるデータの読み書きが同じとなるような既存の方法においても、インタリーブのメモリ領域の確保が必要で、また該メモリへの読み書きのため

10

20

30

40

50

の遅延時間が必要であった。

【0028】

また、AAL1等で行われている既存の方法では、誤り検出や復号のために受信時にも従来送信時と同程度あるいはそれ以上の遅延が生じてしまうという問題もあった。というのは、受信時には、到着したデータをデータ領域のものだけでなくFEC冗長領域のものも含めて全て、順次デインタリーバ内に格納し、FECにより誤り検出、さらには訂正を行なっていたからである。

【0029】

本発明は、FECによる誤り訂正機能及び再送による誤り訂正機能を用いた通信におけるデータ再送制御方法及びデータ再送制御システムを提供することを目的とする。

10

【0030】

【課題を解決するための手段】

本発明は、FECによる誤り訂正機能および再送による誤り訂正機能を用いた通信におけるデータ再送制御方法であって、送信端末において、受信端末から、該受信端末にてFECによる誤り訂正機能によっては訂正できない誤りを伴って受信された誤りのある通信データに対する再送を要求する再送要求を受信する第1の手順と、前記第1の手順にて受信された前記再送要求およびFECによる前記誤り訂正機能の誤り訂正能力に基づいて、前記誤りのある通信データを、前記FECによる誤り訂正機能および前記再送による誤り訂正機能を用いることによってリカバーするのに必要な再送データの最小量を決定する第2の手順と、前記第2の手順にて決定された前記最小量の再送データを、前記送信端末から前記受信端末へ再送する第3の手順とを有することを特徴とする。

20

好ましくは、前記通信は、前記送信端末から複数の受信端末へのマルチキャスト通信であり、前記第1の手順では、送信端末において、少なくとも一つの受信端末から前記再送要求を受信し、前記第3の手順では、前記最小量の再送データを、前記送信端末から前記少なくとも一つの受信端末へ再送するようにしてもよい。

また、本発明は、FECによる誤り訂正機能および再送による誤り訂正機能を用いた通信におけるデータ再送制御システムであって、送信端末において、受信端末から、該受信端末にてFECによる誤り訂正機能によっては訂正できない誤りを伴って受信された誤りのある通信データに対する再送を要求する再送要求を受信する第1の手段と、前記第1の手段にて受信された前記再送要求およびFECによる前記誤り訂正機能の誤り訂正能力に基づいて、前記誤りのある通信データを、前記FECによる誤り訂正機能および前記再送による誤り訂正機能を用いることによってリカバーするのに必要な再送データの最小量を決定する第2の手段と、前記第2の手段にて決定された前記最小量の再送データを、前記送信端末から前記受信端末へ再送する第3の手段とを有することを特徴とする。

30

好ましくは、前記通信は、前記送信端末から複数の受信端末へのマルチキャスト通信であり、前記第1の手段では、送信端末において、少なくとも一つの受信端末から前記再送要求を受信し、前記第3の手段では、前記最小量の再送データを、前記送信端末から前記少なくとも一つの受信端末へ再送するようにしてもよい。

【0034】

【作用】

40

本発明では、送信端末において、受信端末から、該受信端末にてFECによる誤り訂正機能によっては訂正できない誤りを伴って受信された誤りのある通信データに対する再送を要求する再送要求を受信し、受信された前記再送要求およびFECによる前記誤り訂正機能の誤り訂正能力に基づいて、前記誤りのある通信データを、前記FECによる誤り訂正機能および前記再送による誤り訂正機能を用いることによってリカバーするのに必要な再送データの最小量を決定し、決定された前記最小量の再送データを、前記送信端末から前記受信端末へ再送する。これによって、信頼性の高いデータ通信を効果的に実現することができる。

【0043】

(4)また、本発明によると、インタリーバのデータ部分と冗長部分を独立したものとす

50

ることが可能となる。これによって、ネットワーク上での誤り率が比較的小さい場合にはほとんどのインタリーバのデータ部分には誤りが無いと判断されるのでF E C処理が不要となり、よってS S C Sレイヤでの処理時間を更に短縮することができる。

【0048】

【実施例】

以下、図面を参照しながら本発明の実施例を説明する。

【0049】

本実施例では、送信側端末と受信側端末との間にてA T M網を介してA A Lタイプ5を用いたデータ通信を行う場合を示すが、本発明は他のタイプの方式にも同様に適用できることは言うまでもない。

10

【0050】

この送受信端末間では、例えば大容量トランザクション処理が行われており、両端末間におけるデータのやりとりは、低レイテンシ（再送制御が基本的に許されない）、高スループット、かつ信頼性のあるデータ通信が求められている場合を想定している。例えば、端末が公衆網を介してW A N接続されており、物理的に遠い距離にあるような場合も含まれる。

【0051】

ここでは、両端末間にてやりとりするデータにF E C（F o r w a r d E r r o r C o r r e c t i o n）のための誤り訂正ビットを付与し、送信中の伝送路やA T Mスイッチ内でセル廃棄が生じた場合に、再送せずに正しいデータを再生できる手法を示す。

20

【0052】

本実施例では、A T Mの特徴であるセル廃棄のみを主に仮定して説明を進めているが、実際のデータ通信ではビット誤りも生じる可能性はある。このビット誤りに対する訂正方法としては、後述するようなS S C Sヘッダ/トレイラにC R Cフィールドを設け、セルごとにビット誤り検出を行うことによって誤りの検出されたセルは廃棄し、セル廃棄されたものとみなす方法がある。また、誤り訂正を行う符号として、インタリーバ内のどの位置にビット誤りが生じているのかわからなくても訂正可能であるリードソロモン符号（R S符号）を用いる方法もある。ただし、この機能を適用するには、受信側で必ず誤り位置の特定のための復号演算を行う必要がある。

【0053】

本実施例では、送信側端末と受信側端末は、それぞれA T M端末でありA A L5を用いている。エラーフリーなデータを高いスループットで、かつ低いレイテンシで送受信できるように、A A L5のC P C Sレイヤの上位レイヤであるS S C SレイヤにF E C機能を実装している。

30

【0054】

（第1の実施例）

まず、第1の実施例を説明する。本実施例は、1対1の端末が通信を行なう場合の例である。

【0055】

図1に、本実施例の送受信端末のプロトコルスタックの構造を示す。これは、Uプレーン通信（アプリケーション間の通信）を行なう場合のプロトコルスタックである。図中、1はA T M網、3はA T M交換機を表している。

40

【0056】

送信側端末2 tおよび受信側端末2 rでは、ともに大容量トランザクション処理のアプリケーションが動作しており、送信側端末2 tから受信側端末2 rに対して大容量・リアルタイムのデータが転送される。両端末2 t, 2 rは、直接接続されているものとする。このエンド-エンドのA T Mコネクションは、アプリケーション間に予め確立される。従って、レイヤ2、3、4は通常ヌルとなっている。

【0057】

もし、何らかの理由で送信側端末2 tと受信側端末2 rのコネクション間に1つでもルー

50

タが介されている場合には、そのルータはSSCSレイヤの処理として同様のFEC処理を行えることが必要である。

【0058】

AALレイヤではすでに標準化が決まっているAAL5を実装しており、標準のAAL5の上のSSCSレイヤに、本発明のインタリーブ処理とFEC処理を実装している。これにより、エンド-エンド間で信頼性のある通信を実現している。

【0059】

ただし、より信頼性を増すために、上記でヌルであったトランスポートプロトコルのレイヤ4にもライトウェイトな再送制御プロトコルを実装していてもよい。FECには冗長の程度に応じて訂正能力限界があり、通常はエラーは訂正能力の範囲内におさまるように設計されるべきであるが、万一その範囲を越える場合(すなわち訂正不可能な場合)も考えられる。再送制御プロトコルを実装すれば、FECの能力を越えてしまうほど多量の廃棄が生じるようなことが仮にあった場合、遅延の増大は生じるとしても、データの信頼性だけは保つことができる。

【0060】

図2に、送信側端末2tおよび受信側端末2rの内部構成を示す。

【0061】

送信側端末2tは、アプリケーション処理(例えば広域トランザクション処理)を行い、受信側端末2rに対してアプリケーションデータを送信する。

【0062】

アプリケーション処理モジュール4tは、アプリケーションデータをSSCS処理モジュール6tに渡す処理を行う。

【0063】

SSCS処理モジュール6tでは、渡されたデータに対してインタリーブ操作を行い、FEC冗長コードを付与した後、そのインタリーブの列に対してSSCSヘッダ/トレイラを付与し、これをCPCS処理モジュール14tに渡す。

【0064】

CPCS処理モジュール14tでは、SSCS処理モジュール6tから受けとったデータに対して、AAL5のCPCS処理、すなわち長さ表示とCRCの2つのフィールドを各々演算して付与する処理を含むトレイラ付与作業を行ない、SAR処理モジュール20tに渡す。

【0065】

SAR処理モジュール20tでは、通常のAAL5のSARレイヤ処理としてCPCS処理モジュール14tから渡されたデータに対してセグメンテーションを行なう。

【0066】

セグメンテーションされたデータはATMヘッダ付与モジュール22tに渡され、ATMセルヘッダが付与され、さらに図示しない下位レイヤの処理が行なわれてATM網1にセルとなって投入される。

【0067】

一方、受信側端末2rは、ATM網1から入力されてきたセル流に対して、図示しない物理レイヤおよびATMレイヤの処理を行い、自端末宛のセルを選択して取り出す。そして、ATMヘッダ削除モジュール22rは、ATMヘッダを削除した後にSAR処理モジュール20rに渡す。その際、ATMセルヘッダのPT(ペイロードタイプ)のUI(ユーザユーザ情報)領域に、CS-PDUの最後部を示すビットが表示されていたら、これをSAR処理モジュール20rに通知する。

【0068】

SAR処理モジュール20rは、受け取ったデータ(セルのペイロード)に対して、CS-PDUの最後部の情報を参照しながらAAL5のリアセンブリ処理を行なう。

【0069】

CPCS処理モジュール14rは、SAR処理モジュール20rから受け取ったCS-P

10

20

30

40

50

D Uに対してA A L 5のC P C Sレイヤ処理、すなわちC R Cと長さ表示のチェック等を行ない、C S - S D Uを再生する。

【0070】

S S C Sヘッダ/トレイラ確認モジュール12rは、C S - S D Uに付与されたS S C Sヘッダ/トレイラを確認しながら削除し、デインタリーバモジュール8rに渡していく。その際、C P C Sトレイラで誤りが検出されている場合には、その事実をF E C訂正モジュール10rに通知する。

【0071】

デインタリーバ8rで正しく再生されたデータは、S S C S処理モジュール6rからアプリケーションデータとしてアプリケーション処理モジュール4rへ渡されていく。

10

【0072】

ところで、上記した個々のモジュールは、ハードウェア処理あるいはソフトウェア処理のいずれで行なっても構わない。例えば、アプリケーション処理モジュール4t, 4rおよびS S C S処理モジュール6t, 6rは、各々の端末2t, 2r内部に設けたC P Uによるソフトウェア処理で、また、それらより下層のC P C S処理モジュール14t, 14r、S A R処理モジュール20t, 20rおよびA T Mヘッダ付与/削除モジュール22t, 22rは、各々端末2t, 2r内にハードウェアとして実装する方法もある。

【0073】

次に、送信側端末2tにおける処理をレイヤの順に従って説明する。

【0074】

20

(A) アプリケーション処理モジュール4tによる処理

まず、アプリケーション処理モジュール4で、例えば図3のようなアプリケーションデータが生成される。アプリケーションデータの長さは、アプリケーションによって異なり、一般的には広いレンジを持つ可変長のデータである。アプリケーションデータの発生間隔もアプリケーションに依存し、連続する場合や断続的、断片的な場合など様々なケースが考えられる。

【0075】

アプリケーション処理モジュール4tは、インタリーバモジュール8tに対してアプリケーションデータ(A A L - I D U)を渡していくが、1アプリケーションデータが終了した時点で、その旨をデリミタとしてインタリーバモジュール8tに通知する。

30

【0076】

(B) インタリーバモジュール8tによる処理

インタリーバモジュール8tは、アプリケーション処理モジュール4tから渡されたアプリケーションデータであるA A L - I D Uを、マトリクス状メモリであるインタリーバのデータ領域の部分に順次書き込んでいく。書き込み方は、例えば図4の様に左上端から垂直方向に1列ずつ行う。

【0077】

インタリーバへの書き込みの方法として、ここでは図4に示した方向、順序に従う例を挙げるが、図5に示すような方法など、様々な書き込み順序、方向が可能である。

【0078】

40

F E C冗長領域内にはデータは書き込まず、書き込み終了後に各行のシンボルごとに対して予め決まった演算を行った結果を、対応するF E C冗長領域内の各行に書き込む。なお、演算を行った結果をメモリ内のインタリーバ領域に具体的に書き込まずに、論理的に別のメモリに格納し必要に応じて読み出し下位レイヤに渡すという方法も可能である。

【0079】

インタリーバのサイズは様々なものが考えられるが、例えば、A A L 1では以下のようなサイズのインタリーバがオプションとしてI T U - T勧告I . 363によって標準化されている。横(列)のサイズとしては、1シンボルが1オクテットであって、データ領域124シンボル、F E C冗長領域4シンボルである。この場合の縦(行)のサイズは、セル長からS A Rヘッダ長を差し引いた長さ、47オクテットである。また、A A L 1でのオ

50

プシヨンの他の値として、同じシンボルサイズで縦のサイズが8シンボル、横のサイズがデータ88シンボル、FEC冗長領域6シンボルであるものも標準化されている。ただし、ここではこれらの値にこだわる必要はない。

【0080】

本実施例において、いくつかのインタリーバサイズの値を提示しているが、これらはいずれも、インタリーバがデータ領域としてとることのできる最大値を示している。すなわち、アプリケーションデータのサイズによって、それよりも小さく変化させることが可能である。アプリケーションデータのサイズがインタリーバの横のサイズを最大値まで用いた場合でも、その領域に入らないときには、2枚目のインタリーバ、3枚目のインタリーバと言うように、AAL1でのCBR転送と同じように複数枚のインタリーバを用いて転送することとなる。また、縦のサイズはセルのペイロードのサイズとの関係でいくつか実装に適切なサイズがある。

10

【0081】

例えば、インタリーバの1列に対して1つのSSCSヘッダ/トレイラとAAL5のCPCSTレイラを付与する方法をとったものとする。この場合には、縦の長さはセルのペイロード長からSSCSヘッダ/トレイラ分(ヘッダ/トレイラ部分によるオーバーヘッドをなるべく小さくするように、1オクテット等が考えられる)とCPCSTレイラ分(8オクテット)を除いた長さ($48 - 1 - 8 =$)39オクテット等が考えられる。また、1列を複数セル、例えば2セル分とすれば縦のサイズは($39 + 48 =$)87オクテット等が適当であると考えられる。

20

【0082】

縦のサイズが大きい場合と小さい場合を比べてみる。上述したものと同じ、インタリーバ1列をペイロードに持つSSCS-PDUについて比較すると、SSCSヘッダ/トレイラ長が同じならば、縦のサイズの大きい方が上位レイヤデータの転送効率(packaging efficiency)があがるという利点がある。一方、縦のサイズが長いと転送すべきデータが列の途中で終了してしまう可能性も高く、その場合にはその列の残りの部分をパディングで満たさなければならず、逆に転送効率が下がってしまうという問題点もある。縦のサイズとして適当な値を選ぶには、このようなトレードオフを加味すべきである。さらに、あまりに大き過ぎると、ランダムなビット誤りを含み易い問題あるいはFEC符号のオーバーヘッドが上がる問題などが生じる。

30

【0083】

横のサイズに関しては、従来と同じ誤り訂正符号として1シンボルがnビットのリード・ソロモン符号(以下RS符号と記す)を用いるとして、通常、最大長が $(2^n) - 1$ シンボルに限定される。拡張RS符号を利用すると、それよりも1あるいは2シンボル長いものも可能であるが、拡張RS符号は通常のものと比較して、受信後の復号処理、誤り訂正処理が複雑である。従って本実施例では、あえて拡張RS符号を用いないこととする。nの値としては、ソフトウェアによる処理の容易さも考慮して、8またはその倍数を用いることが一般的である。

【0084】

図6に一例を示すように、上記の値を越えない範囲で、インタリーバのサイズとして、横のサイズを4オクテットの倍数にする方法がある。この方法では、例えばRS符号で1シンボルをnビットとしたときに、nと列シンボル数との積が32の倍数になることを意味している。FEC機能をソフト処理で実現する場合を想定すると、計算機で仮想的にインタリーバへのデータの書き込み、読み出しを行うことになるが、計算機のCPUはメモリへのアクセスが4オクテット単位の方が扱いやすいので、有効なデータが足りない場合には4オクテット単位になるようにパディングをしてでもアライメントをとった方が良い場合もある。

40

【0085】

後で詳しく述べるが、今度はCPCSTレイラをデータ領域とFEC冗長領域の各々に1つずつ付ける例を考える。この場合、インタリーバの縦の長さはSSCSヘッダ/トレイ

50

ラ分のみを考慮すれば十分である。よって、SSCSレイヤのオーバーヘッドを例えば1オクテットとして、 $(48 - 1 =)$ 47バイトが良いと考えられる。しかしこの方法では、CPCSTレイヤによって増加する8オクテットの増加分もセルのペイロードとして伝送しなければいけないことを考慮していないので、そのための8バイト分をインタリーバのデータ領域内に確保する方法か、あるいはCPCSレイヤのアライメントをとる機能に任せる方法のいずれかをとる必要が生じる。CPCSレイヤのアライメントをとる機能に任せる場合には、CPCSTレイヤのみを1セルに格納することになるので40バイトのパディングが必要となり、オーバーヘッドが大きくなってしまふ問題点がある。従って、インタリーバのデータ領域にあらかじめCPCSレイヤ用に8オクテットのスペースを確保しておく方がパッキング効率は良い。

10

【0086】

前述したように、データ転送の場合、AAL-IDUのサイズは一定でなく、SSCSモジュール6tではこのサイズを認識し、さらに受信側SSCSモジュール6rにも知らせる必要がある。多くのプロトコルでは、SSCSモジュール6tでAAL-IDUの終了(デリミタ)を検出することができ、SSCSレイヤでは予めサイズを認識することが可能となっている。従って、送信側のSSCSレイヤにおいてインタリーバのどの位置までデータを書き込んだかを、何らかの手段で明記し受信側SSCSレイヤにわかるようにしなければならない。

【0087】

次に、これまで述べてきた種々の例に基づいて任意のサイズのAAL-IDUをインタリーバに格納していった最終的な様子をいくつかの例について説明する。これらの例では、あらかじめコネクション毎に用いるインタリーバの最大サイズ、FEC領域の大きさ、SSCS-PDUの大きさやフォーマットなどが呼設定時にネゴシエーションされているものとする。

20

【0088】

図7には、インタリーバの縦のサイズが39オクテットである例を示した。このサイズは、1列ごとに対して1オクテットのSSCSヘッダ/トレイラ(図7(b)のようにヘッダを付与している)とCPCSTレイラが1つつ付いている例である。インタリーバの最大列数は128オクテットであり、各行に4オクテット分のFEC領域がある。

【0089】

また、この例は比較的大きなサイズのアプリケーションデータが格納された様子を示しており、複数枚(#1から#Nまで)のインタリーバに渡っている。もちろん、小さなサイズのアプリケーションデータであっても、図7に示したインタリーバの状態と同じように格納できる。図7(a)の#1は、データ領域がこの例で許容される最大値まで使用されているインタリーバであり、最後の1オクテットを用いてその列に含まれている有効なデータ量を長さ表示(LI)フィールドに表示する。また、図7(a)の#Nはデータ領域に未使用部分が残っている(アプリケーションデータが途中で終了した)インタリーバであり、最終列はパディングを含む場合もある。このインタリーバでも同様に最終列の最後の1バイトを用いて、その列の有効データ量をLIフィールドに表示する。よって、最終列の有効データ長とLIフィールドの1オクテットの和が39オクテットに満たない場合、その足りない分だけパッドを付加する。

30

40

【0090】

次に、図8に、インタリーバの縦のサイズが47バイトである例を示す。図9には、この場合のSSCSレイヤの処理の流れを説明するための図を示す。

【0091】

AAL-IDUが到着すると(ステップS11)、インタリーバへの書き込みを行う(ステップS13)。

【0092】

インタリーバが満たされると(ステップS14)、必要に応じてPADを挿入し、LIを書き込む(ステップS15)。そして、FEC冗長コードの計算を行う(ステップS16

50

）。

【0093】

インタリーバからは、順次、列ごとのデータが読みだされ、SSCSヘッダが付与される（ステップS17）。

【0094】

そして、CPCS処理（ステップS18）、SAR処理（ステップS19）が行われ、ATMヘッダが付与され（ステップS18）、ATMセルとしてATM網を転送される。

【0095】

さて、図8は、インタリーバのデータ領域とFEC領域それぞれに対して1つずつCPCSTレイラが付与されている例である。図7の例と同じように、各列に対しては、1バイトのSSCSヘッダ/トレイラが付与される（図8（b））。また、この例も比較的大きなサイズのアプリケーションデータが格納された様子を示しており、複数枚のインタリーバ（#1～#N）に渡っている。もちろん、小さなサイズのアプリケーションデータであっても、図8に示したインタリーバの状態と同じように格納できる。

10

【0096】

図8（a）の#1は、データ領域が最大値まで使用されているインタリーバであり、最後の9バイトを用いてその列に含まれているパディングの量をLIフィールド（1オクテット）に表示し、残り（8オクテット）は未使用とする。この8オクテットCPCSTレイラが付与される際に余分なパディングを行わなくてすむように、予めトレイラのサイズ分を確保しておくための領域である。また、図8（a）の#Nは、データ領域に未使用部分が残っている（アプリケーションデータが途中で終了した）インタリーバであり、最終列はパディングを含む場合もある。このインタリーバでも同様に最終列の最後の1バイトを用いてその列のパディングの量をLIフィールドに表示する。よってパッドは、最終列の有効データ長とLIフィールドの1オクテットの和が47オクテットに満たない場合、その足りない分だけ付加される。

20

【0097】

次に、図7と図8で用いられているLI表示の方法について詳しく説明する。図10には、その例を示した。（a）は図7に対応する場合で、データ領域内の途中の列の最下行でアプリケーションデータが終了した場合を表している。この場合最終列には有効なデータは含まれていないので、LIフィールドは0となり、38オクテット分のパディングをする。また、（b）は図8に対応する場合で、アプリケーションデータを書き込んで行って終了した行が38～47オクテット目であった場合の表示方法を示している。この時は、LIフィールドにはパディングの量を示すようにする。このような方法をとることで、送信側から受信側に正確にアプリケーションデータの量を伝えることが可能となる。

30

【0098】

同じアプリケーションのデータであれば、複数のAAL-IDUを1つのインタリーバ内に格納することも可能である。よって、1つのAAL-IDUのデリミタ検出後に、タイマを起動させ、ある一定時間次のAAL-IDUが到着するまでそのインタリーバの送待を待つこともできる。

【0099】

タイマーの設定は、アプリケーションがどれくらいのレイテンシを要求するかによっても様々な設定が可能である。例えば、遅延に対する要求条件の厳しくないファイル転送等では数秒待つことも可能であり、またそれとは逆で、動画像などの場合にはタイマーのしきい値を0に設定し帯域を最大限まで使用することもできる。

40

【0100】

アプリケーションデータをデータ領域に書き込んだ後には、レイテンシを小さく抑えるためにも、できるだけ速やかにFEC冗長領域を計算することが望ましい。

【0101】

図11に一例を示すように、インタリーバからの読み出しは書き込んだ方向、順序と同じに、1シンボル書き込み終了後引き続いて1シンボル読み出しをすぐに行う。書き込みと

50

読み出しを交代に、または並列に行うことによって、今まで書き込み終わるまで読み出しができなかったというボトルネックが解消され、SSCSレイヤの上位レイヤから下位レイヤに渡すまでにかかる時間が大きく減少する。データ読み出し後は、順次SSCSヘッダ/トレイラ付与モジュール12tへ渡していく。

【0102】

以上のように行くと、インターリーブにより生じていた遅延がほとんど無視でき、SSCSレイヤがヌルの場合とかなり近い速度が得られる。以前のインターリーブでは、書き込みは水平方向に、読み出しは垂直方向にと、1つのデータ領域内全てにデータが書き込まれてから（全てが埋まらない場合にはPADをしてから）読み出しを行っていた。これに対して本実施例の方法では、下位レイヤに渡すデータの読み出し方向を書き込み方向と同一にし、特にデータ領域に関しては書き込みと読み出しがほぼ同時に行われるので、データ領域全てを書き込むまでデータの送待を待たなくてもよくなり、遅延がかなり小さくなる。

10

【0103】

(C) FEC付与モジュール10tによる処理

FEC付与モジュール10tでは、図12に例を示すように、データ領域の範囲内に書き込まれたデータの各行に対して、それぞれのFEC冗長領域を計算し対応する位置に書き込んでいく。n行目に対するFEC冗長領域は、n行1列、n行2列、・・・、n行m列のm個のシンボルを用いて演算することによって得られた結果である。

【0104】

パディングがなされていないインターリーブの場合は図12(a)のように、また、インターリーブの途中でデータが終了している場合は図12(b)のように未使用部分を除いて各行に対応するFEC冗長領域を計算していく。(b)の場合に、n行目に対するFEC冗長領域を計算するには、n行1列からn行4列までのシンボルを用いて、未使用部分が無い場合と同じ演算方法でFEC冗長領域を計算することが可能である。

20

【0105】

図13に、1列が1つのCPCS-SDUとなる場合の例を示す。インターリーブは、ALL-IDUであるデータが書き込み終わったら、図13(a)のように、その列の残りの部分をPAD（例えばシンボル中のビットがa11“0”またはa11“1”）で埋めて、さらにその列の最後の1シンボルの部分に、その列に含まれる有効なデータの長さを書き込む。もし、ある列の最後までデータが埋まっていたら、図13(b)のように、次の列を使って長さ表示を行う。要するに、列の最後に長さ表示が含まれている列の前の列はデータで満たされることになる。

30

【0106】

データが全く書かれていない残りの列に対しては、PAD等を行なわない。データ領域の最後は次のステップで付与されるSSCSヘッダ/トレイラで受信側が認識できるので、余分なPADをしてインターリーブの最大サイズに合わせる必要はない。また、データを書き込んでいない部分はセルとなって転送されることもない。このような操作を行うと、CBR転送に適していたインターリーブをデータ転送にも容易に適用することが可能となるのである。

40

【0107】

FEC冗長領域を計算する際、データが書き込まれていずダミーの入っていないデータ領域の各列はFEC冗長コード演算のために代入するパラメータが存在しないので、演算数が少なくて済むことになる。しかしながら、ダミーとして0または1がパディングされていて、しかもそれをFECの計算のためのデータとして用いる従来の場合には、その値を用いてデータがある列と同様の計算を各シンボルに対して行わなければならないので、データ領域が有効なデータで満たされている場合と同じだけの計算時間が必要になってしまう。したがって、残ったデータ領域にPADを入れず、かつその領域をFECのための演算領域からはずす場合は、演算量が減少するため簡単に短時間でFEC冗長の計算が行えることになり、より高速にプロトコル処理できることになる。

50

【0108】

図13は、データを書き込むという観点から、仮想的にインタリーブ行列をにおいて、そこに左上から書き込む様子を示しているが、FECのための冗長を演算するような立場からは、実際には図14のように書かれていることと等価になる。すなわち、マトリクスの左側（すなわちFECのための情報部分の上位の方）はダミーのPADがつけられていて、データ領域の右端につめられて実データが置かれている構成である。実データ領域の中では、番号の若い（すなわち早く上位レイヤから来た）データほど左側にある。すなわち、FEC演算をする符号語、という点からは、下位の実データ分の情報部分とFEC冗長部分とがあるように見える。この様子を、図15(a)に示す。

【0109】

他には、例えば図16の様にデータが全く入らない列にダミーデータを入れてFEC冗長領域を計算するが、それらを含むダミーセルは送出しない方法もある。網に送出されるデータ量は図13の場合と同様に最低限に抑えることが可能である。

【0110】

それ以外にも、前述した図5(a)のようにシンボル毎にインタリーブのデータ領域中にランダムに配置したり、図5(b)のように行の順序をデータ領域の中で自由に設定することも可能である。これらの場合、同様にデータの全く入らない列にいたダミーデータ列は送出しないことにより、網の送出されるデータ量はやはり図13や図16と同じく最低限に抑えることが可能である。

【0111】

これらの例では、あらかじめ通信前に送受信双方でインタリーブ行列への挿入方法についてあらかじめネゴシエーションしておくことは言うまでもない。

【0112】

図16および図5は、図13とは異なり、FEC演算の立場から見たインタリーブマトリクス構造はそれぞれの図と同一である。すなわち、各符号語は、図16の場合は、図15(b)にあるように、上位に実データの情報、その下にダミーの（シンボル値0の）情報があるような情報部分とFEC冗長部分とから構成される。また、図5(b)のようにランダムに実データ列を配する場合は、図15(c)のように実データとダミーデータがまだらに入ったような情報部分と、その右側の冗長部分から成る。

【0113】

図13、図16および図5共に、実データ部分はセルとして伝送されるが、ダミー部分は送出されない。受信側では、ダミー部分には0が送信されたものとみなして復号が行われる。

【0114】

図13および図16に関するFEC冗長領域の作成方法については、後でより詳細に説明する。以下のプロトコル処理の流れでは、代表例として図13のインタリーブマトリクスへの書き込み方法であるものとして説明を行う。

【0115】

FEC冗長領域を演算し始めた時には、データ領域に一度書き込まれたデータはSSCSヘッダ/トレイラを付与するためにSSCSヘッダ/トレイラ付与モジュール12tにほとんど渡されている（図11）。図7および図8のように、FEC冗長領域に書き込まれた計算結果も、演算終了次第、データ領域と同様に列方向に1列ずつ読み出され、SSCSヘッダ/トレイラ付与モジュール12tへと順次渡されていく。FEC冗長領域の計算にかかる時間は、一般にデータを水平方向に書き込むのを待つ時間よりもはるかに小さいことがわかっている。

【0116】

(D)SSCSヘッダ/トレイラ付与モジュール12tによる処理
SSCSヘッダ/トレイラ付与モジュール12tでは、インタリーブモジュール8tから列単位で読み出されたデータに対して、SSCSヘッダ/トレイラを付与しCPCS処理モジュール14tに渡していく。SSCSヘッダ/トレイラを付与する単位は、図17(

10

20

30

40

50

a)のようにインタリーバの1列ごとでもよいし、図17(b)のように、一定の列数ごと、例えば2列ごとをひとまとめでしたものなども考えられる。すなわち、インタリーバの複数列にSSCSヘッダ/トレイラを付与したものがSSCS-PDUとなる。複数のインタリーバ列をSSCS-PDUとする場合には、ひとまとめでする列数が最低限FECの訂正能力を越えないようにする。FEC方式の誤り訂正では、付与している誤り訂正符号のオーバーヘッド(冗長部分)の量によって、訂正できるシンボル数(インタリーバの列数)に限界がある。例えば、前述の図8におけるインタリーバのFEC冗長領域の列サイズが4の場合には、データとFEC領域全体に対して、4列までの消失誤りを訂正することができる。

【0117】

ここで付与するSSCSヘッダ/トレイラには、それが付与されているデータ列がインタリーバのどの位置(何列目)にあったかを確実に特定するための情報が含まれていなければならない。

【0118】

代表的なFECを行う誤り訂正符号としては、Reed-Solomon符号があるが、この誤り訂正符号を用いて容易に訂正するための前提として、「誤り位置が特定されていること」という条件がある。もし、誤り位置が特定されていない場合、その位置特定のための計算量を多く必要とし、また、訂正能力も位置特定時のおよそ半分のシンボルエラーしか訂正できない。誤り位置(ここでは誤りが生じているデータを含むセルのインタリーバ内の位置に相当する)を特定するには、何らかの形で送信時にセル毎の識別子を付けておく必要がある。その識別子として最も有効であろう方法は、例えばセル単位にSNを付与することである。また、SNにもある程度の信頼性を持たせる必要がある。

【0119】

SSCSヘッダ/トレイラとしては、例えば図18のようなフィールドを持たせることができる。ここでは、1バイトの長さのヘッダの例を示している。

【0120】

インタリーバの横のサイズの一例として、CPUを用いてのソフト処理のためには、データ長を該CPUがアクセスする単位として4バイトにアライメントすることが望ましいことを前述したが、ここでも同様のことが言え、ヘッダ/トレイラ長を4バイトとすることも可能である。

【0121】

ここで、図18のSSCSヘッダの一例について説明する。

【0122】

(a)は、同期ビットであり、インタリーバの先頭のSSCSヘッダ/トレイラであることを示す。例えば、先頭の時は“1”、それ以外は“0”と表示する。

【0123】

(b)は、シーケンスナンバであり、インタリーバ中の順序を表示する。例えば、セル廃棄された時の抜けが検出できる。

【0124】

(c)は、SNP(Sequence Number Protection)ビット、すなわちシーケンスナンバに対するプロテクトのためのパリティビットであり、シーケンスナンバの誤りを検出する。

【0125】

(d)は、D/F(Data/Flag)ビットであり、ペイロードの中身がデータであるかそれともFEC冗長コードであるかを示す。

【0126】

(e)は、B/E(Begin/End)ビットであり、データとFEC冗長コード領域各々の最後であることを示す。例えばインタリーバ内で、データ領域およびFEC冗長領域の最後にあたるSSCSヘッダ/トレイラのこのフラグを“1”、それ以外を“0”に設定する。

10

20

30

40

50

【 0 1 2 7 】

インタリーバからは、データ領域に引き続いて F E C 冗長領域が読み出され、S S C S ヘッダ/トレイラ付与モジュール 1 2 t に渡されてくる。S S C S ヘッダ/トレイラの付与は、これらのデータ列に対して、上述したようなヘッダ/トレイラを一定間隔に挿入していくことで実現される。ここでいう一定間隔は、インタリーバにおける 1 列分の長さであったり、もしくは複数列分の長さである。

【 0 1 2 8 】

図 1 9 には、インタリーバ 1 つ分に対する S S C S ヘッダ/トレイラの例を示す。ここで示した値は、誤りの無い正しい S S C S ヘッダ/トレイラの一例である。

【 0 1 2 9 】

同期ビットは、インタリーバの先頭にある S S C S - P D U を示し、シーケンスナンバはこの場合 4 ビットで循環している。図 1 9 の例では、シーケンスナンバはデータ領域に付与されている最初の S S C S ヘッダ/トレイラを 0 としてスタートし、1 6 を法として増加しており、データ領域の最後の S S C S ヘッダ/トレイラと F E C 冗長領域の最初の S S C S ヘッダ/トレイラは連続している。また、別の方法として F E C 冗長領域の最初の S S C S ヘッダ/トレイラでシーケンスナンバをクリアし再度 0 からスタートさせた場合でも実現は可能である。ただし、シーケンスナンバを途中でクリアせずに図 1 9 の例のように連続していた方が、データ領域と F E C 冗長領域の境界でセル廃棄が生じた場合もセル廃棄の検出が容易に行えるという利点がある。また、1 つの A A L - I D U のサイズがかなり大きく、複数のインタリーブに渡って格納される場合に対しては、連続したインタリーブ間に連続したシーケンスナンバを付与する方法と、データ領域の先頭の S S C S ヘッダ/トレイラには必ず 0 を付与する方法との 2 通りが考えられる。データ領域の列数は固定長でないが、S S C S ヘッダ/トレイラの別のフィールド (D / F ビットや B / E) の表示や、F E C 冗長領域のサイズが予めエンドエンド間でネゴシエーションされていることから、データ領域と F E C 冗長領域が交代して転送されてくる境界を正しく識別することが可能である。

【 0 1 3 0 】

シーケンスナンバプロテクトは 4 ビットのシーケンスナンバに対して偶パリティを計算するものとする。もちろん、その代わりに奇数パリティを用いても構わない。

【 0 1 3 1 】

本実施例では、D / F ビットは 1 であればインタリーバ内のデータ領域であることを示し、0 であれば F E C 冗長領域であることを示している。B / E ビットは、E N D を 1 として、データ領域の S S C S - P D U の最後の P D U と F E C 領域の S S C S - P D U の最後の P D U にマークをしている。

【 0 1 3 2 】

図 1 9 で示した S S C S ヘッダ/トレイラで、B / E フィールド = 1 , D / F フィールド = データ、の値を持つヘッダ/トレイラが、そのインタリーバのデータ領域の最終列に含まれる有効なデータ長を示す L I フィールドを含む S S C S - S D U に付与される。

【 0 1 3 3 】

(E) C P C S 処理モジュール 1 4 t の処理
 順次 C P C S 処理モジュール 1 4 t に渡されたデータには通常の A A L 5 処理が行われ、長さ表示と C P C S トレイラが付けられる。C P C S - S D U となる単位の例として、図 7 と図 8 の 2 種類を挙げる。図 7 では、S S C S ヘッダ/トレイラとそのペイロードが 1 組となったものが、C P C S - S D U となる単位である。図 8 では、インタリーバのデータ領域とインタリーバの F E C 冗長領域のそれぞれ (合計 2 つ) が、C P C S - S D U となる単位である。

【 0 1 3 4 】

図 7 の場合は、インタリーバの 1 列ごとに S S C S ヘッダ/トレイラを付与する方法をとり、それぞれに対して A A L 5 の C P C S トレイラが付与される。この方法では A A L 5 の C R C がインタリーバの 1 列に対して適用されるので、もし受信側でエラーが発見され

10

20

30

40

50

た場合には、該インタリーバの特定の一行にエラーのあることが認識される。また、もしセル廃棄が発生したとしても、SSCSヘッダ/トレイラにおけるシーケンス番号の抜けから、どのインタリーブ列が受信側に到着しなかったかを把握できるようになっている。このようにして、セル廃棄やビット誤りの位置を正確に特定することが可能となる。

【0135】

図7の応用として、インタリーバの1列ではなく、複数列毎にSSCSヘッダ/トレイラを付与する方法もある。図7ではSSCSレイヤのオーバヘッドとインタリーバデータの一行とCPCSTレイラの和が例えば48オクテットになるようにインタリーバを設計して、レイヤ毎に余分なパディングを防ぐような工夫をしている。これを応用した場合においても、インタリーバの縦の長さを工夫して、SSCS-PDUの長さと同様にCPCSTレイラの長さの和が48オクテットの倍数か、あるいはそれが通信を行う端末の操作上不可能な場合でも、なるべくCPCSTレイヤでのパディングを少なくするような設計が、データ転送効率をあげる上で望ましい。

10

【0136】

図8の場合は、AAL5トレイラも1つのインタリーバに対して2つだけ付与する方法をとる。2つとは、データ領域全体に対して1つのCPCSTレイラを付与し、FEC冗長領域全体に対してもう1つのCPCSTレイラを付与することを意味している。前者については、パディングを最小限にするために、予めインタリーバモジュール8tでインタリーバのデータ領域の最後の8バイト分は未使用にしてあるので、SSCSレイヤから渡されたデータにCPCSTレイラを付与すると、ちょうど48バイトの倍数になり、SAR 20 処理を行うときにパディングの必要がなくなる。一方、FEC冗長領域に対してCPCSTレイラを付与する場合には、予めインタリーバモジュール8tで未使用部分を確保しておくことができないので、図8のようにインタリーバの一行を47オクテットとした場合には、40オクテットのパディングが必要になってしまう。しかし、例えば図7と比較して、AAL5のトレイラ数が少ない分のオーバーヘッドを小さくすることができ、転送効率があがることになる。また、上記パディングを減らしたい場合には、例えば図8において、インタリーバの長さを変更し、FEC冗長領域を含むSSCS-PDUとCPCSTレイラの長さの和を48オクテットで割った余りが0であるかあるいは、48オクテットに十分近い値であるようにすることも可能である。詳しくは、受信側のSSCSヘッダ/トレイラ確認モジュール12rの説明において述べる。

20

30

【0137】

(F) SAR処理モジュール20tによる処理

SAR処理モジュール20tでは、CPCST処理モジュール14tから受け取ったデータ(CPCST-PDU)のセグメンテーション処理が行なわれる。ただし、インタリーバの縦のサイズ、SSCSヘッダ/トレイラのサイズ、CPCSTレイラのサイズの合計が48オクテットになっていれば、セグメンテーションは行なわれない。一方、例えば、CPCST-PDUが図8のようになっていると、CPCSTレイラがデータ領域、FEC冗長領域のそれぞれに対して1つずつ付与されているような場合には、セグメンテーションが行なわれる。

【0138】

48オクテットに区切られたSAR-PDUは、ATMヘッダ付与モジュール22tに渡される。その際、ATMヘッダ付与モジュール22tに渡されるSAR-PDUがCS-PDUの最後部分にあたるデータである場合、その旨をATMヘッダ付与モジュール22tに通知する。

40

【0139】

(G) ATMヘッダ付与モジュール22tによる処理

ATMヘッダ付与モジュール22tでは、呼設定時に定められたATMセルヘッダを付与した後、物理レイヤの処理を介して、ATMセル化されたデータをATM網1に対して送出する。SAR処理モジュール20tからデータ(SAR-PDU)を受け取る際、これはCPCST-PDUの最後部分である旨の通知を受けている場合は、ATMセルヘッダのペ 50

イロードのUUIフィールドに、これを告げる旨のビットを立てておく。

【0140】

以上が、送信側端末2tにおける処理の流れである。

【0141】

次に、受信側端末2rにおける処理をレイヤの昇順、すなわち各処理モジュールの動作する順に従って説明する。

【0142】

(H) ATMヘッダ削除モジュール22rによる処理

ATMヘッダ削除モジュール22rでは、ATM網1から受け取ったセル流について、物理レイヤ処理を行った後に呼設定時に定められたATMセルヘッダを有したセルを受け取った場合、これをフィルタリングして受け取り、その他のセルは無視し、ATMセルヘッダを削除してSAR処理モジュール20rに渡す。ここで、ATMセルヘッダのペイロードタイプのUUIフィールドに、CPCS-PDUの最後部であることを示すビットが立てられていたらその旨をSAR処理モジュール20rに対して通知する。

10

【0143】

CPCS-PDUの最後部を構成するセルであることを示すビットを含むセルが廃棄された場合、図20のように、次に到着するCPCS-PDUと同一のCPCS-PDUとして上位レイヤに渡されてしまう。しかしながら、CPCSトレイラのチェックによりこの誤りは検出される。さらに、その誤りを含むCPCS-PDUをSSCS処理モジュール6tまで渡すことで、FECを用いて訂正することが可能なのである。

20

【0144】

(I) SAR処理モジュール20rによる処理

SAR処理モジュール20rでは、ATMセルヘッダ削除モジュール22rから渡されたデータ(SAR-PDU)に対し、CPCS-PDUの最後部であるという表示を参照してデータのアセンブリを行う。アセンブリ終了後、CPCS処理モジュール14rに該データが渡される。

【0145】

(J) CPCS処理モジュール14rによる処理

CPCS処理モジュール14rでは、SAR処理モジュール20rから受け取ったCPCS-PDUに対し、AAL5の処理であるCRC演算処理、および長さ表示の確認を順次行う。

30

【0146】

ここで、CRC演算処理、および長さ表示の値のいずれかに誤りが検出されたかどうか、結果をSSCS処理モジュール6rに通知する。この場合、どちらの演算で誤りが生じたかを知らせる必要は必ずしもない。

【0147】

(K) SSCSヘッダ/トレイラ確認モジュール12rによる処理

SSCSヘッダ/トレイラ確認モジュール12rでは、主にSNに抜けがないか、さらに、CPCSトレイラで誤りが検出されている場合にはその通知を受け取る等の確認しながら、ヘッダ/トレイラを取り除いていく。

40

【0148】

CPCSトレイラで誤りが検出されていることを通知されている場合には、そのCPCS-PDUに含まれるSSCS-PDUのいずれかが廃棄されている可能性が高いので、SNをチェックしセル廃棄の位置を特定しなければならない。

【0149】

SSCSヘッダ/トレイラの誤り(ビット誤りも含む)としては様々な状況が考えられる。ここでは、図18の例に示したフィールドを持ったヘッダ/トレイラの誤りの対処について説明する。

【0150】

(a)の同期ビットはインタリーブの先頭を示すビットである。そのSSCS-PDUが

50

インタリーバの先頭のデータを含んでいる場合にはそのビットをたてて1とし、その他の場合には0とするものである。もし、この同期ビットが反転してしまうような誤りが起きた場合には、そのSSCS-PDUから新規のインタリーバであることがわからないかも知れない。しかし、その前のSSCS-PDUの(d)のFEC冗長コードである表示と(e)B/Eビットが最後であることを示していることによって、同期ビットが反転していても新規のインタリーバの先頭を検出することが可能である。

【0151】

(b)のシーケンスナンバでビット誤りが生じた場合には、その後の(c)のシーケンスナンバプロテクション(例えば、偶数パリティ)を用いて誤り検出することができる。前後のシーケンスナンバが正しいことがわかれば、ヘッダ/トレイラを取り除いた後、SSCSのペイロード部はインタリーバの相当する位置に書き込むことができる。また、その代わりに、疑わしいデータを廃棄してしまってセル廃棄と見なしてFEC冗長コードを用いて正しいデータを再生させることも可能である。

10

【0152】

(d)のD/Fは、ペイロード部が含むビット列がデータであれば1、FEC冗長コードであれば0というように、ペイロードのタイプを示す。また、(e)のB/Eは、データを含む複数の連続したSSCS-PDUの中でそのSSCS-PDUが最後であることを示す。FEC冗長部についても、同様に、連続したSSCS-PDUの中で最後であることを示す。従って、B/Eビットが1の(要するにそのSSCS-PDUが最後のデータである)場合に、この表示がビット誤りを起こしたとする。しかし、次のSSCSヘッダ/トレイラのD/Fビットが1から0になり、そのペイロードにはFEC冗長コードを含むことが知られるので、SSCS確認モジュール12rでは正しく認識することが可能である。

20

【0153】

上記では、伝送路においてビット誤り率が例えば 10^{-10} 以下といった微小なネットワークであるような場合を想定していることから、該SSCSヘッダ/トレイラおよびその前後のSSCSヘッダ/トレイラには同時に2ビット以上のビット誤りが存在しないことを前提として記述をしている(ただし、 \wedge はべき乗を意味する)。しかしながら、実際には、極めて珍しいことではあるが、その範囲に2ビット以上の誤りが含まれることもあり、その場合には、例えば、インタリーバのFEC冗長領域の大きさが固定であることや、シーケンス番号が連続していること等、上記以外の条件を加味して正確なSSCSヘッダ/トレイラを再現することが可能である。さらには、該SSCSヘッダ/トレイラ全体の長さを例えば2オクテット長にし、その一部または全体を誤り訂正符号化することによって、ほぼ完全にビット誤り対策が施されることになる。

30

【0154】

次に、以上のような対誤り性を考慮した上で、具体例として図7と図8で示した送信データを受信した場合の処理について説明する。また、各々の場合で、CPCS処理モジュール14rからCPCSTレイラに誤りが検出された対処の方法について述べる。

【0155】

まず、図7の場合について説明する。

40

【0156】

この場合は、インタリーバ内の比較的小さな単位、例えば1列ごとに1つのSSCSデータについて、1つのCPCSTレイラが付与されている。それぞれの列に関してCRCおよび長さ表示を用いたチェックを行うことができるので、それらが正しい場合にはそれぞれのペイロード部の信頼性が保証され、ペイロードに含まれているSSCSヘッダ/トレイラの信頼性も保証されることになる。もちろん、CRCチェックによって漏れがあるような誤りパターンが存在しない訳ではない。しかしながら、そのようなパターンの発生確率は、該データに4ビット以上の誤りが発生し、かつ、その誤りパターンが偶然CRCチェックにより検出できないという、極めて微小なケースであり、これはこのレイヤの誤り検出レベルとしては、十分無視できる程度のものである。

50

【 0 1 5 7 】

また、このC P C Sトレイラの付与の方法では、セル単位にC R Cが演算されることになり、セル内のビット誤りを検出することも可能となる。そこで、ビット誤りの検出されたセルを廃棄されたと見なすことによって、誤り位置不明のビット誤りを位置のわかった消失誤りの形で誤り訂正することができる。F E C符号では、位置不明のランダム誤りを訂正するよりも、位置のわかった消失誤りに対しての方がおよそ2倍の訂正能力があるため、この手法はより少ないF E C冗長部で同じ訂正能力を有することができるという点で極めて有効である。

【 0 1 5 8 】

図7の応用として、インタリーバの数列に対してC P C Sトレイラを1つ付加する場合を上記した。そのような場合に、もし1つのC P C S - P D Uが1つのセルに入らない場合は、複数セルに対して、C P C Sトレイラによるチェックが行われる。そのケースで、C R C等のチェックによってビットエラーが発見されたとすると、そのエラーがインタリーバの数列のどこにあるかがわからないため、該当するC P C S - P D Uに属するインタリーバの数列全部を廃棄されたものとみなす方法と、位置の特定できないランダム誤りが発生したとして訂正を行う方法がある。もちろん、これらいずれの場合においても、訂正できるだけの十分なF E C冗長領域を有していることが必要である。

10

【 0 1 5 9 】

図7において、1つのインタリーバに含まれるデータ領域を含む全てのC P C S - P D Uに対してC P C Sトレイラの誤りが全く通知されていない場合には、上記したように、該インタリーバのデータ領域には誤りがないと判断できる。この場合、S S C Sヘッダ/トレイラ確認モジュール1 2 rはそれらのデータに対して以下の手順で処理を行う。

20

【 0 1 6 0 】

先頭S S C S - P D Uから順にS S C Sヘッダ/トレイラのD / FビットがDであることを確認しながら、デインタリーバモジュール8 rに渡していく。C P C Sトレイラで誤りが検出されていないので、デインタリーバモジュール8 rのデータ領域に格納せずに、直接上位レイヤであるアプリケーションレイヤまで渡すように通知する。

【 0 1 6 1 】

そして、データ領域に相当する部分を含む複数のC P C S - S D Uより以降に到着する、F E C冗長符号を含むC P C S - S D Uは廃棄するように命令を出す。というのは、もはや誤り訂正をする必要がないので、次以降に到着するF E C冗長領域のデータを含むC P C S - S D Uは不要だからである。

30

【 0 1 6 2 】

もちろん、S Nのチェックを行い、セル廃棄が無いことを確認しながら、S S C Sヘッダ/トレイラを取り除くことはいうまでもない。これらの処理は、S S C Sヘッダ/トレイラ確認モジュール1 2 rの基本動作である。よって、以下に説明するケースでも必ず行われる。

【 0 1 6 3 】

次に、図7において、C P C S - P D Uに誤りが検出された場合を考える。

【 0 1 6 4 】

先頭のS S C S - P D Uから順にS S C Sヘッダ/トレイラのD / FビットがDであることを確認しながら、デインタリーバのデータ領域に格納するようにデインタリーバモジュール8 rに渡していく。データ領域の最後の列を含むS S C S - P D Uに付与されたヘッダ/トレイラはエンドビットがマークされており、更に次のS S C S - P D UはF / DビットがDからFに変わる。このP D Uからは、F E Cを含むデータであることをデインタリーバモジュール8 rに通知する。この通知に従って、デインタリーバモジュール8 rは、データとF E C冗長符号をデインタリーバの正しい位置に格納していく。途中で生じているセル廃棄は、S S C Sヘッダ/トレイラ中のS Nにより廃棄されたセルの位置の特定が可能である。

40

【 0 1 6 5 】

50

上記のようにデインタリーバ上に置かれたデータに関する誤り訂正の詳細については、デインタリーバモジュール 8 r による処理の説明のところで行う。

【 0 1 6 6 】

次に、図 8 の場合について説明する。これは、インタリーバ内のデータ領域と F E C 冗長領域に対してそれぞれ 1 つずつの C P C S トレイラが付与されている例である。図 2 1 には、この場合の処理の流れを説明するための図を示す。

【 0 1 6 7 】

A T M 網から転送されてきた A T M セルの A T M ヘッダを削除し (ステップ S 3 1)、S A R 処理 (ステップ S 3 2)、C P C S R 処理 (ステップ S 3 3) を行う。C P C S - P D U に対して行われた C R C および長さ表示チェックの結果が C P C S 処理モジュール 1 4 r から S S C S レイヤに対して通知される。 10

【 0 1 6 8 】

図 7 の場合と同様に、誤りが検出されていなければ、その旨の通知を S S C S ヘッダ / トレイラ確認モジュール 1 2 r が C P C S 処理モジュール 1 4 r から受けている。この場合には、次に F E C 冗長領域を含む C P C S - P D U が C P C S レイヤから渡されたときには、誤りを含む含まないに係わらずその C P C S - S D U を廃棄してよい。よって、図 2 1 で「F E C 冗長コードの廃棄命令を出す」というオペレーションが動作し (ステップ S 3 5)、次に到着する C P C S - S D U を廃棄する (ステップ S 3 6)。このように処理を行えば、F E C を付与していない場合と比較しても S S C S ヘッダ / トレイラを削除するだけのオーバーヘッドで実現が可能である。 20

【 0 1 6 9 】

すなわち、F E C 冗長領域とデータ領域が別々の C P C S - P D U に属していることにより、上述したように、データ領域の C P C S - P D U にエラーのない場合に、F E C 領域の S S C S 処理を行う無駄がなくなり、結果として処理の軽減を図ることができるのである。これは、必ずしもデータ領域と F E C 冗長領域に 1 つずつの C P C S - P D U を割り当てる場合にとどまらず、図 7 の例などにもあるように、1 つの C P C S - P D U 内に、インタリーバのデータ領域と F E C 冗長領域が混ざって存在しなければ、同様にあてはまることである。

【 0 1 7 0 】

次に、データ領域を含む C P C S - S D U にエラーが発見された場合について説明する。 30
C P C S トレイラで誤りが検出されたデータ列の先頭に位置する S S C S ヘッダ / トレイラの中を見て (ステップ S 3 4)、ペイロード内のデータ列がデータ領域を示している場合、すなわち D / F ビットが D である場合、誤り訂正を行わなければならない (ステップ S 3 8, S 4 0)。ペイロードの中身が F E C 冗長領域、すなわち D / F ビットが F である場合には、それとペアであるデータ領域を含む C P C S - S D U で誤りが検出されていなければ上述の例にあてはまるので、F E C 冗長符号を含む C P C S - S D U はもはや不要であり、訂正の必要はない (ステップ S 3 9, S 4 0)。

【 0 1 7 1 】

先頭の S S C S - P D U から順に S S C S ヘッダ / トレイラの D / F ビットが D であることを確認しながら、デインタリーバのデータ領域に格納するようにデインタリーバモジュール 8 r に渡していく。データ領域の最後の列を含む S S C S - P D U に付与されたヘッダ / トレイラはエンドビットがマークされており、更に次の S S C S - P D U は F / D ビットが D から F に変わる。この P D U からは、F E C を含むデータであることをデインタリーバモジュール 8 r に通知する。この通知に従って、デインタリーバモジュール 8 r は、データと F E C 冗長符号をデインタリーバの正しい位置に格納していく。途中で生じているセル廃棄は、S S C S ヘッダ / トレイラ中の S N により廃棄されたセルの位置の特定が可能である。 40

【 0 1 7 2 】

しかしながら、図 7 の場合と異なり、通常のランダム誤りが含まれている場合、その誤り位置をインタリーブのどの一列あるいは数列であるかを特定することができない。従って 50

、この場合は、位置のわからないランダム誤りが含まれることを前提とした誤り訂正処理を行わなければならない。

【0173】

このランダム誤り訂正は、一般に消失誤り訂正よりも処理量が多くなるため、それを防ぐための工夫を行うことができる。

【0174】

セル内に含まれるランダム誤り（位置のわからないビット誤り）を検出するためには、セル毎にセル全体に対するCRCを設けることが必要である。このようなCRCを設けるには、SSCSヘッダ/トレイラに含めることが適切であると考えられ、その場合には、SSCSヘッダ/トレイラは、図18に示した例よりも大きなサイズになる可能性もある。SSCSヘッダ/トレイラのCRCにより、あるセル内に誤りが生じていることが発見された場合には、そのセルは消失したものとみなし、デインタリーバにはそのセルは廃棄されたと通知する手段をとる。このように、ランダム誤りを消失誤りにみだてて訂正を行うことにより、処理量が多くなるのを避けることができる。

10

【0175】

ところで、受信側のCPCS処理とSSCS処理を協調動作させたソフトウェアで実現する場合には、さらなる高速化を図ることができる。すなわち、データ領域に対するCPCS処理時に行う通常の長さのカウントとCRC演算を同時に行うのと並行に、SSCS処理としてSSCSヘッダ/トレイラの削除を行うものである。

【0176】

このように行くと、CPCS処理で誤りが検出されなかった場合には、すでにメモリにはSSCSヘッダ/トレイラの取り除かれた状態で書き込まれているので、上位レイヤへの受け渡しはポイント渡しで瞬時に行えるのである。もちろんこの後に渡されるFEC冗長領域を含むCPCS-SDUは、すでに不必要であるので廃棄して構わない。

20

【0177】

データ領域を格納したCPCS-PDUについてのチェックで誤りが検出されたならば、その旨をSSCS処理モジュール6rに通知し、続いてCPCS処理モジュール14rで処理されるFEC冗長領域を含むCPCS-PDUをチェックし、両CPCS-PDUをSSCS処理モジュール6rに渡さなければならない。SSCS処理モジュール6rでも誤りの検出された通知を受け、それに従いシーケンスナンバ等ヘッダ/トレイラのを確認し、セルの抜けている位置をデインタリーバに通知し誤り訂正をさせることとなる。

30

【0178】

(L) デインタリーバモジュール8rによる処理

デインタリーバモジュール8rでは、SSCSヘッダ/トレイラ確認モジュール12rからの情報により、データ領域に誤りがなければデータ列をインタリーバに書き込みしなおすことなく、スルーな状態でアプリケーション処理モジュール4rに渡していく。例えば図5のように、インタリーブ上ではランダムな並びであるようなデータもあるが、これらデータの送受信の順序は上位アプリケーションから見て順序透過性がある。すなわち、CPCSレイヤからのデータについて、SSCSレイヤでのオーバーヘッドを除けば、そのままスルーしてアプリケーション処理モジュール4rに渡すことが可能である。

40

【0179】

SSCSヘッダ/トレイラ確認モジュール12rからの情報により、データ領域に誤りが検出されていれば、誤りを含むデータをインタリーバの所定の位置に戻していき、該データ領域に対応して付随されているFEC冗長領域のビット列をSSCSヘッダ/トレイラ確認モジュール12rから受け取り、同じくインタリーバのFEC冗長領域に戻し、FEC演算を行って誤り訂正を行う。

【0180】

この際、データ領域の最後のSSCS-SDUは全てがデータではなく、一部がダミーデータである可能性が高い。1列のどこまでがデータであるかは、SSCSヘッダ/トレイラによりデータを含む最後のSSCS-SDUであることを認識した結果を、デインタリ

50

ーバまで通知してもらうことにより検出する。

【0181】

図8の場合、デインタリーバではデータ領域の最後のSSCS-SDUが到着したら、その列の最後から9シンボル目(LIフィールド)を見る。そこに示された値は、LIフィールドから逆方向に何オクテットのパディングが挿入されているかを示している。また、図7の場合には、データ領域の最終列の最後のシンボルがLIフィールドとなっており、そのLIフィールドから逆方向に何オクテットのパディングが挿入されているか、またはその最終列内の最上行から何オクテットの実データが含まれているかのいずれかの情報を表す。

【0182】

その後、データ領域に書き込まれている訂正されたデータをアプリケーション処理モジュール4rに渡していく。渡す順序は送信側端末2tでインタリーバモジュール8tに書き込んだ順序と同様に列方向に読み出しながら渡していく。デインタリーバ処理を行ってデータ長を正確に把握しているので、上位レイヤに渡す場合にもパッドのダミーデータを上位に渡してしまうことはない。データ領域のSSCS-PDUに誤りが検出されていない場合にも、上位にパッド以外のデータのみを渡さなければならないので、データの正確な長さを把握する必要がある。この場合は、FEC冗長コードの廃棄命令を出す際に、先に説明したデインタリーバを行った場合と同じようにデータ領域を含む最後のSSCS-SDUをSSCSヘッダ/トレイラ確認時に通知してもらい、それによって長さを正確に知る。パッドであった部分は上位レイヤに渡す前に取り除く。

【0183】

(M)アプリケーション処理モジュール4rによる処理

最後に、アプリケーション処理モジュール4rが、アプリケーションデータをSSCSヘッダ/トレイラ確認モジュール12rから受け取り、アプリケーション処理を行う。

【0184】

以上が、受信側端末2rにおける処理の流れである。

【0185】

次に、より具体的な例として、FEC付与モジュール10tおよびFEC訂正モジュール10rにおけるFEC手法の動作例をあげる。

【0186】

送信側において、FEC付与モジュール10tでは、インタリーバモジュール8tからのデータを見ながらインタリーバのFEC冗長領域に付加すべきデータを計算し、その結果をすぐにインタリーバモジュール8tに返す役割を担う。インタリーバモジュール8tでは、受け取ったFEC冗長領域用のデータを通常のデータに続けてSSCSヘッダ/トレイラ付与モジュール12tに渡す。SSCSヘッダ/トレイラ付与モジュール12tでは、それぞれの領域に従って、ヘッダ/トレイラを付与する。

【0187】

また、受信側において、FEC訂正モジュール10rは、そのインタリーバについて訂正する必要のある時のみ動作する。すなわち、下位のCPCS処理モジュール14rにおいてセル抜けやビット誤りによるエラーが検出された時、そのデータはSSCS処理モジュール6rに渡されてデインタリーバモジュール8rに蓄えられる。そして、SSCSヘッダ/トレイラ確認モジュール12rにおいて、多くの場合、デインタリーバ内のどの位置にセル抜けあるいはビット誤りがあるのかが検出される。プロトコル仕様によっては、位置の正確に特定できないビット誤りが含まれることがあるが、その場合でも、訂正能力は誤り位置の特定された場合よりも限定されるものの、FEC訂正モジュール10rにおいては訂正可能な構成をとることができる。しかしながら、この実施例では簡単のため、上記SSCSヘッダ/トレイラに付与された種々のビットが有する能力によって、ビット誤りが発生した場合においても、その誤りシンボル位置が特定できるものと仮定する。このようなエラーに対して、FEC訂正モジュール10rでは、デインタリーバモジュール8rのデータを参照しながら、データ抜け部分の復元、あるいはビット誤りの訂正を行う。

10

20

30

40

50

そして、訂正情報をインタリーバモジュール 8 t に戻す。

【 0 1 8 8 】

F E C 付与モジュール 1 0 t および F E C 訂正モジュール 1 0 r の構成は、これまでに述べてきたインタリーバモジュール自身のサイズや、それぞれのインタリーバについて、1 つの S S C S ヘッダ/トレイラを付与する列数や、またインタリーバへの書き込み位置等によって、少しずつ異なることになる。

【 0 1 8 9 】

従って、これらのモジュールの例を挙げる際には、インタリーバモジュール 8 t およびインタリーバモジュール 8 r と関連させながら説明することとなる。はじめに、送信側における F E C 冗長シンボル構成法の例をあげ、その後それぞれの方法に対する、受信側での誤り訂正方法の例を述べることとする。

10

【 0 1 9 0 】

まず、図 7 および図 8 に示されるようなインタリーバの一行がすべて 1 セルのペイロードに収容され、また該 1 列のみで S S C S レイヤのペイロードとなる場合について説明する。このような場合は、インタリーバの縦の長さは 3 9 オクテットまたは 4 7 オクテット、横の最大長は R S 符号の 1 シンボルを k ビットとして $2^k - 1$ シンボル長以下になる。このようにシンボルの大きさを可変にすると、それに従ってインタリーバの大きさが変わるが、これらインタリーブ処理には、主としてソフトウェア上での操作を考慮する必要があるので、ソフトウェア処理に適したインタリーバ長にするため、1 シンボルのビット数は 2 の n 乗 (n は整数)、すなわち 4 ビット、8 ビット、1 6 ビット等が望ましい。また

20

【 0 1 9 1 】

そのようにすると、図 7 のような例では、インタリーバの縦の長さが 3 9 オクテット、横の長さが 2 5 5 オクテット長以下の任意のオクテット長となる。また図 8 では、横は同じで縦のみが 4 7 オクテットとなる。どちらの図でも、データ領域に最大 1 2 4 オクテット、F E C 冗長領域に 4 オクテット分を割り当てている。実際にはこれらの合計が 2 5 5 オクテットを越えないように設計できることになるが、本実施例では、A A L 1 におけるインタリーバの横の大きさに合わせた例として、図 7 および図 8 の場合について詳細に示す

30

【 0 1 9 2 】

具体的には、 α を 8 次の原始多項式の根とし、生成多項式は、

$$\begin{aligned} G(X) &= (X - \alpha^{120}) (X - \alpha^{121}) (X - \alpha^{122}) (X - \alpha^{123}) \\ &= X^3 - \alpha^{162} \cdot X^2 + \alpha^{35} \cdot X^2 - \alpha^{150} \cdot X + \alpha^{231} \\ &\quad \dots (1) \end{aligned}$$

とする。ただし、(1) 式およびそれ以降の式はすべて $G F (2 ^ 8)$ 上での演算によるものとする。ここで、 α^b は、 α の b 乗を表すものとする。

40

【 0 1 9 3 】

図 7 および図 8 における実際の F E C 演算の方法としては、図 1 3 および図 1 4 による方法、図 1 6 による方法、図 5 による方法等がある。

【 0 1 9 4 】

以下、それぞれの方法を示す。

【 0 1 9 5 】

(1) 送信側 F E C 処理方法 1

ここでは、図 1 3 や図 1 4 の送信側におけるインタリーバへのデータ挿入方法に対する F E C 冗長データの生成方法を示す。

【 0 1 9 6 】

50

図14にあるように、この方式の場合、インタリーバのなるべく右側につめてデータが挿入されていることが特徴である。従って、図7のインタリーバで具体的に説明すると、データがインタリーバのデータ領域に全部入っている場合には、 39×124 シンボル分のデータが入るが、もしデータが m 列分しかない場合、インタリーバの左から $(124 - m)$ 列は空(データがはいっていない)となる。そして、残り m 列の先頭からデータが入るが、データ領域の最後の列は実データが全部入っている場合と、パディング用のデータの混じっている場合がある。

【0197】

データの入っていない先頭の $(124 - m)$ 列に関しては、この部分をセルに合成して送信することはないため、FEC用の符号化をするに際しても、この部分は、符号化のための情報部に含めない。ただし、装置化するにあたって、送受信の少なくとも一方がインタリーバの大きさを可変長にすることが困難であるような場合があるかもしれない。そのときは、あらかじめ送受信間でネゴシエーションし、実際にはデータを送らないとしても、インタリーバの中で空きがでる部分にどのようなダミーデータをいれてFECの符号化および復号化を行うかについて決めておく方法もある。

10

【0198】

インタリーバについては、各行毎に符号化を行うこととする。この場合、 m シンボル(オクテット)の情報部分と4シンボル(オクテット)の冗長部分からなる符号語が39行分できることになる。各行は並列に符号化/復号化が行われ、それぞれの行で同じ動作が行われるため、以下では一行のみについて示すこととする。

20

【0199】

インタリーバの一行には情報シンボルは m 個($1 \quad m \quad 124$)あるので、それをインタリーバの左側から表して、

$$M = (M[m-1], M[m-2], M[m-3], \dots, M[0]) \quad \dots (2)$$

と表記することにする。たまに情報シンボルの存在しない行が存在するが、その場合は $M[*]$ はない。

【0200】

また、符号語のシンボルは $(m + 4)$ 個あるので、それもインタリーバの左側から表すと、

30

$$C = (C[m+3], C[m+2], C[m+1], \dots, C[4], C[3], C[2], C[1], C[0]) \quad \dots (3)$$

となる。情報 M と符号語 C とは一対一であるので、必ずしも同じシンボル値を共有する必要はないが、今回のようにインタリーブすることによって遅延が増えないようにするためには、なるべくシンボル値を共有することが望ましい。

【0201】

従って、上記(2)式と(3)式において、情報シンボルはそのまま符号シンボルとして使用し、

$$C[m+3] = M[m-1]$$

40

$$C[m+2] = M[m-2]$$

$$C[m+1] = M[m-3]$$

・

・

・

$$C[4] = M[0] \quad \dots (4)$$

50

とする。そして、 $C[3]$ 、 $C[2]$ 、 $C[1]$ 、 $C[0]$ を冗長部分とする。

【0202】

このようにすると、送信時に、上位レイヤからのメッセージをそのまま下位に流すことができる。すなわち、一旦インタリーバに書き込むことなく、上位レイヤからのメッセージをスルーして流せるため、遅延が小さくて済むというメリットが生まれる。ただし、上位から下位へ流すデータは、冗長シンボル生成のための計算処理のためには使用される。

【0203】

一方、受信時にはデインタリーバのデータ領域の全体、あるいはその領域をいくつかに分割したそれぞれの部分に対して、下位レイヤにおいてCRCによる誤り検査が行われる。この検査の結果そのデインタリーバのデータ領域に誤りがないと判定された場合には、データ領域のデータは、SSCSレイヤにおいて、デインタリーバに改めて書き込むことなく、SSCSヘッダ/トレイラ処理のみを行ってスルーして上位レイヤに送ることができる。また、冗長部分は不必要なので、これも特に処理を行わず即座に廃棄できる。このようにして、受信側処理全体のスループットを向上させるというメリットがある。ところが、もし同じ値を元の情報と符号語とで共有している式(4)のような構成になっていなければ、符号語全体を受信しないと元のメッセージが復元できないため、上述のメリットが生まれないことになる。

【0204】

以上は、主として図14に関するインタリーバモジュール8t中のデータについて述べた。式(4)のようにして、 $C[m+3]$ から $C[4]$ までは簡単に生成することができる。

【0205】

次に、残りの冗長部分の生成方法を述べる。この冗長部分は、上述した構成のインタリーバモジュール8tからのデータを受けて、FEC付与モジュール10tにおいて生成される。

【0206】

ここでは、AAL1と同じようなRS符号を用いることとしているので、演算としては、まず情報Mを多項式表現した $M(X)$ を、

$$M(X) = M[m-1] \cdot X^{m-1} + M[m-2] \cdot X^{m-2} + \dots + M[1] \cdot X + M[0] \quad \dots (5)$$

と表して

$$M(X) \cdot X^4 = Q(X) \cdot G(X) + a[m] \cdot X^3 + b[m] \cdot X^2 + c[m] \cdot X + d[m] \quad \dots (6)$$

となるような、 $a[m]$ 、 $b[m]$ 、 $c[m]$ 、 $d[m]$ の係数を求めることを行えばよい。

【0207】

ここで、 $Q(X)$ は、 $M(X) \cdot X^4$ を $G(X)$ で割った商にあたり、式(6)の $Q(X) \cdot G(X)$ 以外の右辺部分は、その余りに相当する。よって、 $M(X)$ が決まると、式(6)の左辺を $G(X)$ でわり算すればよい。 $a[m]$ 、 $b[m]$ 、 $c[m]$ 、 $d[m]$ はそれぞれ、

$$\begin{aligned} C[3] &= a[m] \\ C[2] &= b[m] \\ C[1] &= c[m] \\ C[0] &= d[m] \end{aligned} \quad \dots (7)$$

10

20

30

40

50

に相当する。

【0208】

繰り返し述べているように、このインタリーバはある最大サイズまでは可変であり、しかもそれがどれ位の大きさになるかは、上位からのデータ量次第である。従ってデータが多い場合も少ない場合も同じような計算手法を使わないと、計算に無駄が生じることになる。式(6)をそのまま用いることは、 $M(X)$ が確定してからならばよいが、通常は上位レイヤからいきなりメッセージ M を全部一度に与えられることはない。上位レイヤからは、例えば数バイト長ずつのデータが流れてきている。それを全部受け取った後に、はじめて上位レイヤからのメッセージ長がどれだけの量であった、ということが判明するのである。従って、もし $M(X)$ が確定するまでFEC演算処理の開始を待つとすると、それは従来のAAL1のような垂直方向のデータの読み書きをする場合と同じ程度の遅延を要することになってしまう。そのような訳で、上位レイヤからデータが流れ始めると、すぐにFEC冗長領域の計算が開始でき、しかもいつ上位レイヤからのデータが終わったとしても、常に式(6)の形になっているような計算方法を用いる必要がある。

10

【0209】

そこで、本実施例においては、ハードウェアにおいて符号化に用いられているシフトレジスタを用いた方式を(ソフトウェアで実現する場合においても)採用するのがよい。すなわち、まず $M[m-1]$ が上位レイヤより来るため、これに対してあたかもこれが最終列のシンボルであるかのように以下の計算を行う。

【0210】

$$M[m-1] \cdot X^4 = (a[1] \cdot X^3 + b[1] \cdot X^2 + c[1] \cdot X + d[1]) \bmod G(X) \quad \dots (8)$$

20

ここで、式(1)のそれぞれの係数を利用すると、式(8)の値は、

$$\begin{aligned} a[1] &= M[m-1] \cdot \alpha^{162} \\ b[1] &= M[m-1] \cdot \alpha^{35} \\ c[1] &= M[m-1] \cdot \alpha^{150} \\ d[1] &= M[m-1] \cdot \alpha^{231} \end{aligned} \quad \dots (9)$$

30

により簡単に求められる。これは、実際に計算してもよいし、かけ算のテーブルをひく方法をとってもよい。

【0211】

もし、 $m=1$ であって、 $M[m-1]$ が最後であったとすると、右辺の4つの係数の値がそれぞれ式(6)の a, b, c, d に相当する。すなわち、 $M[0]$ は $C[4]$ にそのまま変わり、

$$\begin{aligned} C[3] &= a[1] \\ C[2] &= b[1] \\ C[1] &= c[1] \\ C[0] &= d[1] \end{aligned} \quad \dots (10)$$

40

となる。このように1シンボル目では式(8)と式(6)とでは整合がとれている。

【0212】

次に、2シンボル目が到着した場合を考える。式(6)に合わせるように記述すると、

$$\begin{aligned} &M[m-1] \cdot X^5 + M[m-2] \cdot X^4 \\ &= (a[2] \cdot X^3 + b[2] \cdot X^2 + c[2] \cdot X + d[2]) \bmod G(X) \end{aligned} \quad \dots (11)$$

50

となる。この場合は、 X^5 の項と X^4 の項に分けて、別々に $G(X)$ で割った剰余を計算し、それを後で加える方法がある。しかし、そのような方法では、シンボル数が段々増えていくにつれて、その計算回数がシンボル数に比例して増えていく、という問題が生じる。当然テーブルを引く場合にも、テーブル参照回数が増えていってしまう。

【0213】

そこで、式(8)をうまく利用することを考える。

式(11)は、式(8)を用いて、

$$\begin{aligned} & (a[1] \cdot X^3 + b[1] \cdot X^2 + c[1] \cdot X + d[1]) \cdot X + M[m-2] \cdot X^4 \\ = & (a[2] \cdot X^3 + b[2] \cdot X^2 + c[2] \cdot X + d[2]) \pmod{G(X)} \end{aligned} \quad \dots (12) \quad 10$$

と変形できる。

【0214】

式(12)の左辺を展開して、

$$\begin{aligned} & (a[1] + M[m-2]) \cdot X^4 + b[1] \cdot X^3 + c[1] \cdot X^2 + d[1] \cdot X \\ = & [\{ (a[1] + M[m-2]) \cdot \alpha^{162} + b[1] \} \cdot X^3 \\ & + \{ (a[1] + M[m-2]) \cdot \alpha^{35} + c[1] \} \cdot X^2 \\ & + \{ (a[1] + M[m-2]) \cdot \alpha^{150} + d[1] \} \cdot X \\ & + (a[1] + M[m-2]) \cdot \alpha^{231}] \pmod{G(X)} \end{aligned} \quad \dots (13) \quad 20$$

となる。

【0215】

従って、

$$\begin{aligned} a[2] &= (a[1] + M[m-2]) \cdot \alpha^{162} + b[1] \\ b[2] &= (a[1] + M[m-2]) \cdot \alpha^{35} + c[1] \\ c[2] &= (a[1] + M[m-2]) \cdot \alpha^{150} + d[1] \\ d[2] &= (a[1] + M[m-2]) \cdot \alpha^{231} \end{aligned} \quad \dots (14) \quad 30$$

により、メッセージが2シンボルの場合のFEC冗長領域のシンボル値が求められる。

【0216】

式(14)は、帰納的に用いることができる。すなわち、2シンボル目以降は毎回その以前のシンボル値から容易に計算できる。よって k シンボル目が到着した時には、

$$\begin{aligned} a[k] &= (a[k-1] + M[m-k]) \cdot \alpha^{162} + b[k-1] \\ b[k] &= (a[k-1] + M[m-k]) \cdot \alpha^{35} + c[k-1] \\ c[k] &= (a[k-1] + M[m-k]) \cdot \alpha^{150} + d[k-1] \\ d[k] &= (a[k-1] + M[m-k]) \cdot \alpha^{231} \quad (k=2, 3, \dots, m) \end{aligned} \quad \dots (15) \quad 40$$

という計算を行えばよい。

【0217】

このようにして最後のシンボルが到着するまで、あるいはインタリーバに収容されるデータ領域の最大列数(本実施例では124)までのシンボルが計算されると、終了し、その次のタイミングで、他の行において計算されたものと一緒に、FEC付与モジュール10tからインタリーバモジュールにFEC冗長領域データとして送られる。送る順序は、a 50

[m]、b [m]、c [m]、d [m]の順である。

【 0 2 1 8 】

図 1 0 においては、最後の列にはパディングされた部分があるが、この部分については符号化のための情報に加える方法も加えない方法も実現可能である。しかし、少なくとも L I 表示のシンボルは F E C 符号化のためのデータに加えないので、パディング部分も（たとえ a 1 1 “ 0 ” のデータでも）一応、加えた方が装置化が楽になるであろう。もし、パディング部分を含まないような F E C 符号化を行うとすると、全くデータの無い（つまり情報 M がヌルである）行が存在し得る。このような場合は、F E C 付与モジュールは、a [0] = b [0] = c [0] = d [0] = 0 のデータをインタリーバモジュールに返す。

【 0 2 1 9 】

(2) 送信側 F E C 処理方法 2

ここでは、図 1 6 にあるように、送信側にてデータをインタリーバの左上から正確に挿入するような場合の F E C 冗長データの生成方法を示す。

【 0 2 2 0 】

図 1 6 にあるように、本方式の場合、今度はインタリーバのなるべく左側につめてデータが挿入されているという特徴を有している。従って、図 7 のインタリーバで具体的に説明すると、データがインタリーバのデータ領域に全部入っている場合には、前述した例と同様に、3 9 × 1 2 4 シンボル分のデータが入るが、もしデータが m 列分しかない場合、インタリーバの右から (1 2 4 - m) 列は空（データがはいっていない）となる。そして、m 列目は実データが全部入っている場合と、パディング用のデータの混ざっている場合がある。

【 0 2 2 1 】

データ領域の残りの (1 2 4 - m) 列に関しては、この部分をセルに合成して送信することはない。それはすなわちトラヒックの無駄を意味するからである。F E C 用の符号化をするに際しては、この部分は、あたかも全て “ 0 ” のデータが入っているかのように扱われる。

【 0 2 2 2 】

別の手法として、あらかじめ送受信間でネゴシエーションし、実際にはデータを送らないとしても、インタリーバの中で空きがでる部分にどのようなダミーデータをいれて F E C の符号化および復号化を行うかについて決めておく手法もある。この場合は必ずしも空き領域を全て “ 0 ” にするような必要はなく、お互いの決めたビットパターンを用いて符号化すればよい。しかし、一般にそのようなビットパターンを決めておくと、F E C のための符号化にはかえって演算時間を増加させることになってしまので、ここでは a 1 1 “ 0 ” を仮定することとする。

【 0 2 2 3 】

インタリーバにおいては、前述した例と同様に各行毎に符号化を行うこととする。すると、この場合は m シンボル（オクテット）の情報部分と、(1 2 4 - m) シンボルの “ 0 ” のデータと、4 シンボル（オクテット）の冗長部分からなる符号語が 3 9 行分できることになる。各行は並列に符号化 / 復号化が行われ、それぞれの行で同じ動作が行われるため、そのうち一列について見ることにする。

【 0 2 2 4 】

インタリーバの一行には情報シンボルは m 個 (1 m 1 2 4) あるので、それをインタリーバの左側から表して、

$$M = (M[m-1] , M[m-2] , M[m-3] , \dots , M[0]) \quad \dots (1 6)$$

と表記することにする。たまに情報シンボルの存在しない行が存在するが、その場合は M [*] はない。

【 0 2 2 5 】

また、符号語のシンボルは 1 2 8 個あるので、それもインタリーバの左側から表すと、

10

20

30

40

$$C = (C[127], C[126], C[125], \dots, C[3], C[2], C[1], \\ C[0]) \quad \dots (17)$$

となる。情報Mと符号語Cとは一対一であるので、必ずしも同じシンボル値を共有する必要はないが、インタリーブすることによって遅延が増えないようにするためには、情報シンボル値を共有することが望ましい。

【0226】

従って、上記(16)式と(17)式において、情報シンボルはそのまま符号シンボルとして使用し、

$$\begin{aligned} C[127] &= M[m-1] && 10 \\ C[126] &= M[m-2] \\ C[125] &= M[m-3] \\ &\cdot \\ &\cdot \\ C[127-m+2] &= M[1] \\ C[127-m+1] &= M[0] \\ C[127-m] &= 0 && 20 \\ C[127-m-1] &= 0 \\ &\cdot \\ &\cdot \\ C[4] &= 0 && \dots (18) \end{aligned}$$

とする。そして、C[3]、C[2]、C[1]、C[0]を冗長部分とする。ただし、式(18)において、C[127-m]からC[4]までは実際にはネットワークには送出不しい。

【0227】

このようにすると、送信時には、上位レイヤからのメッセージをそのまま下位に流すことができる。すなわち、一旦インタリーブに書き込むことなく、上位レイヤからのメッセージをスルーして流せるため、遅延が小さくて済むというメリットが生まれる。ただし、上位から下位へ流すデータは、冗長部分生成のための計算処理のためには使用される。

【0228】

受信時には、下位レイヤのCRCによる誤り検査の結果、そのデインタリーブのデータ領域に誤りがない場合、データはSSCSヘッダ/トレイラ処理のみでスルーして上位レイヤに渡せる。冗長部分は即座に廃棄でき、デインタリーブへ改めて誤り訂正のための書き込みを行う必要はない。このようにして、全体の処理遅延を減らすことができるという利点が得られる。もし、式(18)のように同じ値を元情報と符号語とで共有しているような構成になっていなければ、符号語全体を受信しないと元のメッセージが復元できないため、上述の利点は得られない。

【0229】

以上は、主として図16に関するインタリーブモジュール8t中のデータについて述べた。式(18)のようにして、C[127]からC[4]までは簡単に生成することができる。

【0230】

次に、残りの冗長部分の生成方法を述べる。この冗長部分は、上述した構成のインタリーブモジュール8tからのデータを受けて、FEC付与モジュール10tにおいて生成される

10

20

30

40

50

。

【 0 2 3 1 】

ここでは、A A L 1と同じようなR S符号を用いることとしているので、演算としては、まず情報Mを多項式表現したM(X)を、

$$M(X) = M[m-1] \cdot X^{123} + M[m-2] \cdot X^{122} + \dots + M[1] \cdot X^{(123-m+2)} + M[0] \cdot X^{(123-m+1)} \dots (19)$$

と表して、

$$M(X) \cdot X^4 = Q(X) \cdot G(X) + e[m] \cdot X^3 + f[m] \cdot X^2 + g[m] \cdot X + h[m] \dots (20)$$

10

となるような、e[m]、f[m]、g[m]、h[m]の係数を求めることを行えばよい。

【 0 2 3 2 】

ここで、Q(X)は、M(X)・X⁴をG(X)で割った商にあたり、式(20)のQ(X)・G(X)以外の右辺部分は、その余りに相当する。インタリーバのデータ領域中のダミー部分は、値が“0”であり、符号化の計算には関わらないため、式(20)では既に省略されている。よって、M(X)が決まると、式(20)の左辺ををG(X)でわ

20

り算すればよい。e[m]、f[m]、g[m]、h[m]はそれぞれ、

$$\begin{aligned} C[3] &= e[m] \\ C[2] &= f[m] \\ C[1] &= g[m] \\ C[0] &= h[m] \end{aligned} \dots (21)$$

に相当する。

【 0 2 3 3 】

前述した例とは異なり、M(X)を構成する個々の項において、M(X)におけるmの値によらず項の係数と次数は変化しない。すなわち、インタリーバの1列目のデータM[m-1]に対しては、必ずX¹²³であり、これは2列目、3列目とデータが増えていっても変化しない。もちろん2列目以降についても同様である。すると、式(20)において、

30

$$\begin{aligned} \text{左辺} &= M(X) \cdot X^4 \\ &= M[m-1] \cdot X^{127} + M[m-2] \cdot X^{126} + \dots + M[1] \cdot X^{(127-m+2)} + M[0] \cdot X^{(127-m+1)} \dots (22) \end{aligned}$$

となるので、mの値にかかわらず、インタリーバの最初のデータから順にXのべき乗をかけたもの、すなわち式(22)における各項をを個別にG(X)で割ることが可能である。従って、このような形での冗長領域の計算方法としては、インタリーバのデータが1シンボル到着する度に、そのデータに対してG(X)によるわり算を実行し、その余りをそれまでに計算された余りに加える方法が考えられる。

40

【 0 2 3 4 】

以下、この方法を具体的に示す。

【 0 2 3 5 】

まず、インタリーバの最初のデータが到着すると、

$$M[m-1] \cdot X^{127} = (e[1] \cdot X^3 + f[1] \cdot X^2 + g[1] \cdot X + h[1]) \pmod{G(X)} \quad \dots (23)$$

となる $e[1]$ 、 $f[1]$ 、 $g[1]$ 、 $h[1]$ を計算する。次に 2 番目のデータが到着すると、式 (23) と同様に

$$M[m-2] \cdot X^{126} = (e'[2] \cdot X^3 + f'[2] \cdot X^2 + g'[2] \cdot X + h'[2]) \pmod{G(X)} \quad \dots (24)$$

を計算し、その後、

$$\begin{aligned} e[2] &= e[1] + e'[2] \\ f[2] &= f[1] + f'[2] \\ g[2] &= g[1] + g'[2] \\ h[2] &= h[1] + h'[2] \end{aligned} \quad \dots (25)$$

のように排他的論理和を行って 2 シンボルのデータに対する冗長を計算する。以下も同様に行い、最終的に $e[m]$ 、 $f[m]$ 、 $g[m]$ 、 $h[m]$ が得られる。このように 1 つずつ計算することによって、最終のシンボルが到着した時点でそのときの計算値が求められることになる。

【0236】

式 (23) および式 (24) のような $G(X)$ でのモジュロ計算は、毎回行うとかなり計算時間をとられることになる。従って、通常は、各 X のべき乗に対する $G(X)$ のモジュロ値のテーブルを持つのがよい。

【0237】

テーブルには、

$$X^{[128-k]} = (E[k] \cdot X^3 + F[k] \cdot X^2 + G[k] \cdot X + H[k]) \pmod{G(X)} \quad (k=1, 2, \dots, 124) \quad \dots (26)$$

の計算結果である、 $E[k]$ 、 $F[k]$ 、 $G[k]$ 、 $H[k]$ の値とそれに対応する X のべき乗の数が書かれている。すなわち、例えば図 22 に示す表 a のようになる。

【0238】

この表 a を用いると、任意の入力データ $M[m-k+1]$ に対して、

$$\begin{aligned} &M[m-k] \cdot X^{[128-k]} \\ &= M[m-k] \cdot (E[k] \cdot X^3 + F[k] \cdot X^2 + G[k] \cdot X + H[k]) \\ &\pmod{G(X)} \\ &= (e'[k] \cdot X^3 + f'[k] \cdot X^2 + g'[k] \cdot X + h'[k]) \pmod{G(X)} \end{aligned} \quad (k=1, 2, \dots, m) \quad \dots (27)$$

となることから、

$$\begin{aligned} e'[k] &= M[m-k] \cdot E[k] \\ f'[k] &= M[m-k] \cdot F[k] \\ g'[k] &= M[m-k] \cdot G[k] \\ h'[k] &= M[m-k] \cdot H[k] \end{aligned} \quad (k=1, 2, \dots, m) \quad \dots (28)$$

という計算によって、各シンボルに対する剰余の値が簡単に算出できる。ただし、 $e[1] = e'[1]$ 、 $f[1] = f'[1]$ 、 $g[1] = g'[1]$ 、 $h[1] = h'[1]$ である。

10

20

30

40

50

【 0 2 3 9 】

このようにして、最後のシンボルが到着するまで、あるいはインタリーバに收容されるデータ領域の最大列数（本実施例では124）までのシンボルが計算されると終了し、その次のタイミングで、計算結果を他の行において計算されたものと一緒に、FEC付与モジュール10tからインタリーバモジュールにFEC冗長領域データとして送る。送る順序は、e[m]、f[m]、g[m]、h[m]の順である。

【 0 2 4 0 】

図16においても、前述した例と同様に、最後の列にはパディングされた部分があるが、この部分については符号化のための情報に加える方法も加えない方法も実現可能である。しかし、少なくともLI表示のシンボルはFEC符号化のためのデータに加えたいため、パディング部分も（たとえa11“0”のデータでも）データとして加えた方が、装置化が容易になる。もし、パディング部分を含まないようなFEC符号化を行うとすると、全くデータのない（つまり情報Mがヌルである）行が存在し得る。このような場合は、FEC付与モジュールは、e[0]=f[0]=g[0]=h[0]=0のデータをインタリーバモジュールに返す。

【 0 2 4 1 】

(3) 送信側FEC処理方法3

ここでは、図5の送信側におけるインタリーバへのデータ挿入方法に対するFEC冗長データの生成方法を示す。

【 0 2 4 2 】

図5のようにランダムにインタリーバへの書き込みが行われる場合には、送信側FEC処理方法1のような規則的な手法を用いることはできない。そこで、送信側FEC処理方法2のようなテーブルを用いて、各シンボルに対する剰余を計算し、それを加えることによって、FEC冗長領域の値を計算する手法が用いられる。

【 0 2 4 3 】

以下に、図5のそれぞれについて簡単にFEC冗長領域の生成方法を述べる。

【 0 2 4 4 】

送信側FEC処理方法においては、インタリーバの左端から規則正しくデータが挿入されていたので、テーブルの内容についても、図22の表aのように規則正しく並んでいた。しかし、図5のそれぞれの実施例については、その規則性がないため、それぞれ送信側と受信側とであらかじめネゴシエーションした後に、はじめて上記のような表が書けることになる。

【 0 2 4 5 】

図5(b)の場合には、ランダムであるのは書き込む列の順序だけであり、縦方向の書き込み方は決められているので、前述した例と同様に、図22の表aだけを用いればよいが、図5(a)の方は、書き込む列の順序だけでなく、その一列の中でのシンボルの書き込み順序が異なるため、その順序情報についての表を追加して書かなければならない。

【 0 2 4 6 】

まず、図5(b)について説明する。ここでは、図に従いデータ領域が右端7行、FEC冗長領域が4行あることとする。左端の行から、5番目、2番目、7番目（実際はヌルでデータなし）、4番目、1番目、3番目、6番目（実際はヌル）、といったデータとなる。

【 0 2 4 7 】

すると、送信側FEC処理方法2と同様にして、情報の多項式表現M(X)は、

$$M(X) = M[4] \cdot X^6 + M[3] \cdot X^9 + M[2] \cdot X^5 + M[1] \cdot X^7 + M[0] \cdot X^{10} \quad \dots (29)$$

となる。

【 0 2 4 8 】

以下の処理は、送信側FEC処理方法2における式(20)～式(28)の経過と比較し

10

20

30

40

50

て、 $M[*]$ 、 $e[*]$ 、 $f[*]$ 、 $g[*]$ 、 $h[*]$ の係数を除くとほとんど同じである。従って、図22の表aを用いて、まず $M[4]$ に対する e の係数を式(28)の方法で、

$$e[122] = M[4] \cdot E[122] \quad \dots (30)$$

で求め、それ以下のものは、順に、

$$e[119] = e[122] + M[3] \cdot E[119]$$

$$e[123] = e[119] + M[2] \cdot E[123]$$

$$e[121] = e[123] + M[1] \cdot E[121]$$

$$e[118] = e[121] + M[0] \cdot E[118] \quad \dots (31)$$

の排他的論理和演算を行うことによって最終の $e[118]$ を得ることができる。係数 f 、 g 、 h についても全く同様の計算により得られる。こうして、 $e[118]$ 、 $f[118]$ 、 $g[118]$ 、 $h[118]$ がそれぞれ求める冗長シンボルとなる。

【0249】

次に、図5(a)の場合を示す。この場合も図に従ってシンボル単位に書き込みが行われるとすると、図22の表a1の他に、図23の表bを用いることによって計算することができる。ここでも、図に示されているようなデータ領域が右端6行、FEC冗長領域が4行ある具体例を用いて説明する。なお、行番号は左端および上端から昇べきの順につけるものとする。

【0250】

この図23の表bに従うと、図で例えば上から4行目の情報の多項式表現 $M(X)$ は、各シンボルを $M[*]$ で表すと、

$$M(X) = M[1] \cdot X^7 + M[11] \cdot X^5 + M[18] \cdot X^8 + M[19] \cdot X^9 \quad \dots (32)$$

と表せる。ここからは、図5(b)の例と同様にして図22の表aを用い、 X^3 の係数 e は、

$$e[121] = M[1] \cdot E[121]$$

$$e[124] = e[121] + M[11] \cdot E[124]$$

$$e[120] = e[124] + M[18] \cdot E[120]$$

$$e[119] = e[120] + M[19] \cdot E[119] \quad \dots (33)$$

の排他的論理和演算を順に行うことによって、得られる。他の係数 f 、 g 、 h も同様である。

【0251】

こうして、 $e[119]$ 、 $f[119]$ 、 $g[119]$ 、 $h[119]$ がそれぞれ上から4行目の冗長シンボルとなる。他の行については、式(33)において情報Mのシンボル番号が異なるだけで、それ以外は同様の演算により得られる。

【0252】

以上のように、図5(a)、(b)の例でのFEC冗長領域の計算は、送信側FEC処理方法2を多少変形した方法を用いることによって実行することができる。

【0253】

図7および図8のような構成以外の例として、インタリーブの一行が1セルよりも大きい場合、また、インタリーブの複数列に対してSSCSヘッダを付加する場合等が考えられるが、これらの場合においても、図13、図16、図5のそれぞれの例に合わせて、全く同様の演算でFEC冗長シンボルの計算が行える。

【0254】

10

20

30

40

50

これまでは3種類の例をあげて、送信側におけるFEC冗長領域の計算方法について説明してきたが、今度はそれぞれに関する受信側での復号方法および誤り訂正方法について説明する。

【0255】

送信側のインタリーバモジュール8tから出力されたデータはセルとして伝送され、受信側に到達し、セル分解の後、受信側のデインタリーバモジュール8rに入る。ある送信側インタリーバからの受信データは、データ領域の何列かをペイロードに含む1つ以上のCPCS-PDUと、FEC冗長領域の何列かをペイロードに含むやはり1つ以上のCPCS-PDUから成っている。このうち、データ領域をペイロードに含む全CPCS-PDUが、CPCSレイヤにおけるCRCチェック等によって、全てエラーなしと判定された場合、SSCSレイヤにおいては、SSCSヘッダ/トレイラを除去するのみで残りのデータをそのまま上位のアプリケーションモジュール4rに渡す。そして、そのデインタリーバに関して、FEC冗長領域をペイロードに含むCPCS-PDUは全て、デインタリーバ用のメモリに書き込むことなく廃棄して構わない。

10

【0256】

ただし、データ領域側データをペイロードに含む複数のCPCS-PDUにおいて、1つでもCRCチェック等によるエラーがあるときには、そのデインタリーバに関連する全てのデータは、FEC冗長領域のものを含めて、誤り訂正のために必要なものとなる。

【0257】

図7の場合は、デインタリーバの1列から1つのCPCS-PDUを構成しており、しかもそれが1つのセルとなる。このため、受信したCPCS-PDUにおいてCPCSトレイラのCRCチェック等が異常であるという場合には、該CPCS-PDU内にビット誤りが発生していることを意味している。この時、エラーの発見された列はデインタリーバモジュール8r上で、伝送中にセル廃棄された場合と同じ扱いにしてしまう。すなわち、その1列にはセルが到着しなかったと同等にみなす。そしてその結果として生じたセル抜けの位置をSSCSヘッダのSNチェックにより確認する。

20

【0258】

一方、図8のような場合は、デインタリーバの1列は1セルであるが、CPCS-PDUは複数列にまたがって構成される。従って、CPCSトレイラによるチェックのみでは、ランダム誤りおよびセル抜けの状態が不明確である。仮にあるCPCS-PDUにおいてセル抜けがあったとすると、CPCSトレイラの機能だけでは、そのCPCS-PDUに属する他のデインタリーバ列がエラーフリーである保証がなされない。

30

【0259】

そこで、まずセル抜けに対しては、SSCSヘッダ/トレイラのSNを用いて確認する。SN自体に誤り検査ビットを加えておくと、抜けた位置を特定する際の確実性が増す。また、ランダムなビット誤りへの対策としては、SSCSヘッダ/トレイラがデインタリーバの1列毎についているので、この部分にSSCS-PDU全体の誤り検出機能を付加する。そして、伝送中のセル廃棄によって、CPCSトレイラの誤り検出機能が有効に働かなかった場合、あるいはビット誤りによってCPCSトレイラのCRCチェックにエラーが発見された場合に限って、SSCSレイヤにおけるデインタリーバ1列単位の該誤り検査を行い、各列のエラーの有無を検査するという方法が考えられる。これによって、CPCS-PDUのサイズによらず、実際にビット誤りおよびセル廃棄の発見されたインタリーバ列に属するシンボルのみを訂正すればよいことになる。

40

【0260】

また、別の場合として、デインタリーバの1列が複数セルで構成され、その1列をペイロードに含んでSSCS-PDUおよびCPCS-PDUを構成することも考えられる。ここでも、1つのセルが伝送路中に廃棄された場合には、その他の部分のシンボルエラーはCPCSトレイラを用いても検出不可能になる。この場合には、計算を簡単にするために、該当する1列全てが廃棄されたとみなして誤り訂正を試みる方法をとることができる。このような方法を用いることによって、例えば図7の例に比べて誤り訂正能力は若干落ち

50

るが、処理遅延は増加させることなく小さく保持できる。廃棄されたとみなされるデインタリーバ列の位置は、SSCSヘッダ/トレイラによって特定される。

【0261】

SSCSヘッダを除いたデータは、インタリーバと全く同様の順番でデインタリーバ領域に書き込む。この書き込み順は、あらかじめ送受信間でネゴシエーションされているものとする。その際、抜けのある部分については、そのことを別領域に明示するか、あるいはダミーのペイロードデータを書き込む。そして、そのデータをFEC訂正モジュール10rに渡し、シンボル抜けの部分再生する。その後、情報データをデインタリーバモジュール8tから書き込んだ順に読み出して、受信側のアプリケーション処理モジュール4rに送る。

10

【0262】

以下では、上記FEC演算の送信側FEC処理方法1, 2, 3に従って、受信側のFEC訂正モジュール10rにて行われる夫々の場合に対応する誤り訂正操作を、受信側FEC処理方法1, 2, 3として説明する。このモジュールが動作するのは、デインタリーバの任意の一行において、訂正可能な範囲のシンボルエラーがあり、かつその位置が判明している場合であり、以下では、そのような場合を例にとって説明する。なお、図7にあるように、インタリーバの一行がすべて1セルのペイロードに収容され、また該1列のみでSSCSレイアのペイロードを構成する場合について説明するが、それ以外のデインタリーバの構成にも適用可能である。

【0263】

(4) 受信側FEC処理方法1

ここでは、送信側FEC処理方法1として述べたインタリーバに対するデインタリーバを用いた誤り訂正方法を記す。

20

【0264】

送信側FEC処理方法1ではFEC冗長領域が4シンボル分である場合について記してあるので、この受信側FEC処理方法1でもそれに従って記述する。FEC冗長領域が4シンボルの場合、誤りシンボル位置が明確であれば、4シンボルまでの誤りを訂正することが可能である。

【0265】

いま、s番目、t番目、u番目、v番目のシンボルに誤りまたは抜けがあるとする。ただし、 $0 < s < t < u < v < (m + 1)$ であるとする。

30

【0266】

図14は、送信側FEC処理方法1において説明したように、デインタリーバのできるだけ右側につめてシンボルを挿入する構成をしている。デインタリーバ上に埋まるデータ量は毎回可変なので、受信が開始された時点では、どの位のデータが到着するかの予測ができない。従って、送信側と同様に、可変シンボル長に柔軟に対応できるような復号処理を行う必要がある。

【0267】

受信シンボル列は、情報シンボルと冗長シンボルとの合計がmシンボルあるとすると、(17)式から、

40

$$R = (C[m-1], C[m-2], \dots, C[m-s+1], E[m-s], C[m-s-1], \dots, C[m-t+1], E[m-t], C[m-t-1], \dots, C[m-u+1], E[m-u], C[m-u-1], \dots, C[m-v+1], E[m-v], C[m-v-1], \dots, C[0]) \quad \dots (34)$$

のように表され、E[*]の部分はエラーがある部分を意味する。それ以外のC[*]で書かれる部分は、送信側のデータがそのまま正しく受信されていることを意味する。式(34)では一般的な書き方をしたが、E[*]は、先頭あるいは最後尾のシンボルにあっても構わないし、また連続していても構わない。

50

【 0 2 6 8 】

さて、F E C 訂正モジュール 1 0 r での処理を行うまでに、既にシンボル単位にエラーの有無が明確になっている。もし 1 シンボルもエラーがない場合には、式 (3 4) は、

$$C = (C[m-1], C[m-2], \dots, C[0]) \quad \dots (35)$$

と表され、これを用いた多項式 $C(X)$ は、生成多項式 $G(X)$ で割り切れる。これは、すなわち、

$$C(\alpha^{*120}) = C(\alpha^{*121}) = C(\alpha^{*122}) = C(\alpha^{*123}) = 0 \quad \dots (36)$$

10

となることを意味する。しかし、エラーシンボルの含まれる受信シンボル列 R を用いた多項式 $R(X)$ では、 $G(X)$ による余りは 0 とはならない。そこで、これらをシンδροーム $S[i]$ ($i = 1, 2, 3, 4$) として次のように表すこととする。

【 0 2 6 9 】

$$\begin{aligned} S[1] &= R(\alpha^{*120}) \\ S[2] &= R(\alpha^{*121}) \\ S[3] &= R(\alpha^{*122}) \\ S[4] &= R(\alpha^{*123}) \quad \dots (37) \end{aligned}$$

20

ここで、 $E[*]$ の部分の係数を訂正を簡単にするために、 $R(X)$ においては、

$$E[m-s] = E[m-t] = E[m-u] = E[m-v] = 0 \quad \dots (38)$$

とおいておくこととする。

【 0 2 7 0 】

すると、シンδροーム $S[i]$ ($i = 1, 2, 3, 4$) は、以下のようにして計算される。

【 0 2 7 1 】

手順 1 : $S[1] = S[2] = S[3] = S[4] = 0$ とおく。

【 0 2 7 2 】

手順 2 : シンボルが 1 つ来る度に (すなわちデインタリーバの 1 列が到着する度に)

30

$$S[i] \leftarrow S[i] \cdot \alpha^{*(119+i)} + C[m-k] \quad (\text{到着シンボルが正しい場合})$$

または

$$S[i] \leftarrow S[i] \cdot \alpha^{*(119+i)} + E[m-k] \quad (\text{到着シンボルが誤っている})$$

場合)

$$(i=1, 2, 3, 4) \quad (k=1 \sim m) \quad \dots (39)$$

40

のように代入していく。ここで + は排他的論理和を表す。ただし、式 (3 8) によって、 $E[m-k] = 0$ である。

【 0 2 7 3 】

手順 3 : 手順 2 を最後のシンボルまで計算した時の $S[i]$ ($i = 1, 2, 3, 4$) の値が、求めるシンδροーム値である。

【 0 2 7 4 】

手順 2 を詳細に説明する。最終的には式 (3 7) の値を求めたいのであるが、受信を終了するまで、シンボル数がどれだけあるのかがわからない。よって、シンボルを受信する度に計算を行い、突然シンボル受信が終了したとしても、シンδροーム $S[i]$ ($i = 1, 2, 3, 4$) の計算結果が最終的に求める値になっている必要がある。

50

【 0 2 7 5 】

実際に、手順 2 では、以下の計算と同じことを行っている。まず、最初のシンボル $C[m-1]$ が到着すると、シンドローム値は、

$$S[i] = C[m-1] \quad (i=1, 2, 3, 4) \quad \dots (40)$$

となる。

【 0 2 7 6 】

次に、シンボル $C[m-2]$ が到着すると、

$$S[i] = C[m-1] \cdot \alpha^{(119+i)} + C[m-2] \quad (i=1, 2, 3, 4) \quad \dots (41)$$

10

となる。

【 0 2 7 7 】

同様にしていくと、最終的に

$$S[i] = C[m-1] \cdot (\alpha^{(119+i)})^{(m-1)} + C[m-2] \cdot (\alpha^{(119+i)})^{(m-2)} + \dots + C[1] \cdot (\alpha^{(119+i)}) + C[0] \quad (i=1, 2, 3, 4) \quad \dots (42)$$

となり、この式は (37) 式と同じである。

20

【 0 2 7 8 】

このような方法をとることによって、 k 個のシンボルが到着した時には、必ずシンドローム $S[i]$ ($i = 1, 2, 3, 4$) は、 $(k-1)$ 次の多項式に $\alpha^{(119+i)}$ ($i = 1, 2, 3, 4$) を代入した形をとっているので、デインタリーバモジュール 8r のシンボル到着がいつ終了しても、常にその時点の本実施例に基づくデインタリーバ構成での正しいシンドローム演算値を示していることになる。なお、式 (40) ~ (42) において、エラーのあるシンボルを受信した部分については、 $C[*]$ から $E[*]$ に表記を書き換えるとよい。

【 0 2 7 9 】

次に、このようにして得られるシンドローム値の意味を考える。誤りのあるシンボルがもし正しく受信されていたとすると、これらはそれぞれ、 $C[m-k]$ ($k = s, t, u, v$) と表される。すると、これらの記号を用いて式 (37) のシンドローム $S[1]$ の値は、

30

$$\begin{aligned} S[1] &= R(\alpha^{120}) \\ &= C(\alpha^{120}) + (E[m-s] - C[m-s]) \cdot \alpha^{120(m-s)} \\ &\quad + (E[m-t] - C[m-t]) \cdot \alpha^{120(m-t)} \\ &\quad + (E[m-u] - C[m-u]) \cdot \alpha^{120(m-u)} \\ &\quad + (E[m-v] - C[m-v]) \cdot \alpha^{120(m-v)} \\ &= -C[m-s] \cdot \alpha^{120(m-s)} - C[m-t] \cdot \alpha^{120(m-t)} \\ &\quad - C[m-u] \cdot \alpha^{120(m-u)} - C[m-v] \cdot \alpha^{120(m-v)} \\ &\quad \dots (43) \end{aligned}$$

40

となる。

【 0 2 8 0 】

なお、本実施例では、 $\wedge b c$ は、 $(b \times c)$ 乗を表すものとする。例えば、上記の $\wedge 120(m-s)$ は、 $(120 \times (m-s))$ 乗を表す。

【 0 2 8 1 】

50

他のシンδροームも同様に、

$$S[2] = -C[m-s] \cdot \alpha^{121(m-s)} - C[m-t] \cdot \alpha^{121(m-t)} - C[m-u] \cdot \alpha^{121(m-u)} - C[m-v] \cdot \alpha^{121(m-v)}$$

$$S[3] = -C[m-s] \cdot \alpha^{122(m-s)} - C[m-t] \cdot \alpha^{122(m-t)} - C[m-u] \cdot \alpha^{122(m-u)} - C[m-v] \cdot \alpha^{122(m-v)}$$

$$S[4] = -C[m-s] \cdot \alpha^{123(m-s)} - C[m-t] \cdot \alpha^{123(m-t)} - C[m-u] \cdot \alpha^{123(m-u)} - C[m-v] \cdot \alpha^{123(m-v)}$$

… (44)

10

となる。ただし、ここでは2進信号(0, 1)のみを仮定しているので、+の演算と-の演算はどちらも同じ排他的論理和となる。すなわち式(43)と(44)は合わせて、

$$\begin{aligned} S[i] = & C[m-s] \cdot (\alpha^{(119+i)(m-s)}) \\ & + C[m-t] \cdot (\alpha^{(119+i)(m-t)}) \\ & + C[m-u] \cdot (\alpha^{(119+i)(m-u)}) \\ & + C[m-v] \cdot (\alpha^{(119+i)(m-v)}) \end{aligned}$$

(i=1, 2, 3, 4) … (45)

20

と表記できる。

【0282】

このことから、式(39)の手順によってS[i] (i=1, 2, 3, 4)の値が求められると、式(45)は、C[m-s]、C[m-t]、C[m-u]、C[m-v]に関する連立方程式となる。そしてこれを解くことによって、それぞれの正しいシンボル値が求められる。本実施例では、エラーシンボルの見落としがない(すなわち、SSCSレイヤ以下によってエラーのあるシンボルが確実に検出され、それ以外には存在しない)ことを想定しているので、これらの連立方程式には、必ず一意な解が存在する。

30

【0283】

解は、それぞれ、

$$\begin{aligned}
 C[m-s] = & \{ S[4] + (\alpha^{*(m-t)} + \alpha^{*(m-u)} + \alpha^{*(m-v)}) \cdot S[3] \\
 & + (\alpha^{*(2m-t-u)} + \alpha^{*(2m-t-v)} + \alpha^{*(2m-u-v)}) \cdot S[2] \\
 & + \alpha^{*(3m-t-u-v)} \cdot S[1] \} \\
 & / \{ \alpha^{*120(m-s)} \cdot (\alpha^{*(m-t)} + \alpha^{*(m-v)}) \cdot (\alpha^{*(m-t)} + \alpha^{*(m- \\
 & u)) \cdot (\alpha^{*(m-u)} + \alpha^{*(m-v)}) \}
 \end{aligned}$$

$$\begin{aligned}
 C[m-t] = & \{ S[4] + (\alpha^{*(m-s)} + \alpha^{*(m-u)} + \alpha^{*(m-v)}) \cdot S[3] \\
 & + (\alpha^{*(2m-s-u)} + \alpha^{*(2m-s-v)} + \alpha^{*(2m-u-v)}) \cdot S[2] \\
 & + \alpha^{*(3m-s-u-v)} \cdot S[1] \} \\
 & / \{ \alpha^{*120(m-t)} \cdot (\alpha^{*(m-s)} + \alpha^{*(m-v)}) \cdot (\alpha^{*(m-s)} + \alpha^{*(m- \\
 & u)) \cdot (\alpha^{*(m-u)} + \alpha^{*(m-v)}) \}
 \end{aligned}$$

10

$$\begin{aligned}
 C[m-u] = & \{ S[4] + (\alpha^{*(m-t)} + \alpha^{*(m-s)} + \alpha^{*(m-v)}) \cdot S[3] \\
 & + (\alpha^{*(2m-t-s)} + \alpha^{*(2m-t-v)} + \alpha^{*(2m-s-v)}) \cdot S[2] \\
 & + \alpha^{*(3m-t-s-v)} \cdot S[1] \} \\
 & / \{ \alpha^{*120(m-t)} \cdot (\alpha^{*(m-t)} + \alpha^{*(m-v)}) \cdot (\alpha^{*(m-t)} + \alpha^{*(m- \\
 & s)) \cdot (\alpha^{*(m-s)} + \alpha^{*(m-v)}) \}
 \end{aligned}$$

20

$$\begin{aligned}
 C[m-v] = & \{ S[4] + (\alpha^{*(m-t)} + \alpha^{*(m-u)} + \alpha^{*(m-s)}) \cdot S[3] \\
 & + (\alpha^{*(2m-t-u)} + \alpha^{*(2m-t-s)} + \alpha^{*(2m-u-s)}) \cdot S[2] \\
 & + \alpha^{*(3m-t-u-s)} \cdot S[1] \} \\
 & / \{ \alpha^{*120(m-v)} \cdot (\alpha^{*(m-t)} + \alpha^{*(m-s)}) \cdot (\alpha^{*(m-t)} + \alpha^{*(m- \\
 & u)) \cdot (\alpha^{*(m-u)} + \alpha^{*(m-s)}) \} \quad \dots (46)
 \end{aligned}$$

となる。

30

【0284】

式(45)において、エラーシンボルがもし1つ($E[m-s]$)のみであれば、上式は

$$S[i] = C[m-s] \cdot \alpha^{*(119+i)(m-s)} \quad (i=1, 2, 3, 4) \quad \dots (47)$$

であり、これから、

$$C[m-s] = (\alpha^{*(119+i)(m-s)}) / S[i] \quad (i=1, 2, 3, 4) \quad \dots (48)$$

40

となる。これは*i*の値によらず同じなので、どれか1つを計算すればよい。とすると、式(37)において、例えば*S*[1]のみ計算すればよく、他のシンδροーム計算を行う必要がないので、その分の演算量を減らすことができる。

【0285】

また、もしエラーシンボルが2つ($E[m-s]$ と $E[m-t]$)であれば、式(45)は、

$$\begin{aligned}
 S[i] &= C[m-s] \cdot (\alpha^{*(119+i)(m-s)}) \\
 &\quad + C[m-t] \cdot (\alpha^{*(119+i)(m-t)}) \\
 &\hspace{15em} (i=1, 2, 3, 4) \quad \dots (49)
 \end{aligned}$$

となる。これも2つのシンδροーム、例えばS[1]とS[2]のみを計算すればよく、他のシンボルに対してかかる演算の分を減らすことができる。

【0286】

そして、解は、

$$\begin{aligned}
 C[m-s] &= (S[2] + \alpha^{*(m-t)} \cdot S[1]) && 10 \\
 &\quad / \{ \alpha^{*120(m-s)} \cdot (\alpha^{*(m-s)} + \alpha^{*(m-t)}) \} \\
 C[m-t] &= (S[2] + \alpha^{*(m-s)} \cdot S[1]) \\
 &\quad / \{ \alpha^{*120(m-t)} \cdot (\alpha^{*(m-s)} + \alpha^{*(m-t)}) \} \quad \dots (50)
 \end{aligned}$$

となる。

【0287】

また、エラーシンボルが3つ(E[m-s]とE[m-t]とE[m-u])であるとき、式(45)は、

$$\begin{aligned}
 S[i] &= C[m-s] \cdot (\alpha^{*(119+i)(m-s)}) && 20 \\
 &\quad + C[m-t] \cdot (\alpha^{*(119+i)(m-t)}) \\
 &\quad + C[m-u] \cdot (\alpha^{*(119+i)(m-u)}) \\
 &\hspace{15em} (i=1, 2, 3, 4) \quad \dots (51)
 \end{aligned}$$

となり、この場合は3つのシンδροーム、例えばS[1]、S[2]、S[3]のみを計算すればよい。すなわち、S[4]に対してかかる演算の分を減らすことができる。

【0288】

解は、

$$\begin{aligned}
 C[m-s] &= \{ S[3] + (\alpha^{*(m-t)} + \alpha^{*(m-u)}) \cdot S[2] + \alpha^{*(2m-t-u)} \cdot S && 30 \\
 [1] \} \\
 &\quad / \{ \alpha^{*120(m-s)} \cdot (\alpha^{*(m-s)} + \alpha^{*(m-t)}) \cdot (\alpha^{*(m-s)} + \alpha^{*(m- \\
 u)) \} \\
 C[m-t] &= \{ S[3] + (\alpha^{*(m-s)} + \alpha^{*(m-u)}) \cdot S[2] + \alpha^{*(2m-s-u)} \cdot S && \\
 [1] \} \\
 &\quad / \{ \alpha^{*120(m-t)} \cdot (\alpha^{*(m-s)} + \alpha^{*(m-t)}) \cdot (\alpha^{*(m-t)} + \alpha^{*(m- && 40 \\
 u)) \} \\
 C[m-u] &= \{ S[3] + (\alpha^{*(m-t)} + \alpha^{*(m-s)}) \cdot S[2] + \alpha^{*(2m-t-s)} \cdot S && \\
 [1] \} \\
 &\quad / \{ \alpha^{*120(m-u)} \cdot (\alpha^{*(m-u)} + \alpha^{*(m-t)}) \cdot (\alpha^{*(m-s)} + \alpha^{*(m- && \\
 u)) \} &&& \dots (52)
 \end{aligned}$$

となる。

【0289】

これらの演算には、 のべき乗の計算が多数必要となる。常に出てくる演算(例えば ^ 50

120等)に関しては、表を作っておくと高速化を図ることができる。

【0290】

(5) 受信側FEC処理方法2

ここでは、送信側FEC処理方法2で述べたインタリーバに対するデインタリーバを用いた誤り訂正方法を説明する。

【0291】

受信側FEC処理方法1と同様に、FEC冗長領域の大きさと同じ4シンボルまで訂正可能である。そこで、 $0 < s < t < u < v < (m + 5)$ であるような、s番目、t番目、u番目、v番目にそれぞれシンボル誤り、またはシンボル抜けがあるとする。

【0292】

図16の場合は、送信側FEC処理方法2で説明したように、データ領域内のシンボルはデインタリーバのできるだけ左側に詰められており、一方FEC冗長領域のシンボルは、デインタリーバの右端4列に入れられているような構成である。受信側FEC処理方法1と同様に、データが終了するまで、デインタリーバのデータ領域のどこまでデータが入るかは毎回に異なり、全くわからない。従って、シンボル受信開始と同時に復号演算を開始し、シンボル受信がいつが終了したとしても、即座に誤り訂正処理に入れるような復号処理を行う必要がある。

【0293】

受信シンボルは、受信側FEC処理方法1の場合とは異なり、データがmシンボル($0 < m < 125$)、冗長部分が4シンボルあるとする。このとき、受信シンボル列Rは(17

$$R = (C[127], C[126], \dots, C[128-s+1], E[128-s], C[128-s-1], \dots, C[128-t+1], E[128-t], C[128-t-1], \dots, C[128-u+1], E[128-u], C[128-u-1], \dots, C[128-v+1], E[128-v], C[128-v-1], \dots, C[128-m], 0, 0, \dots, 0, C[3], C[2], C[1], C[0]) \quad \dots (53)$$

のように表され、E[*]の部分はエラーがあるシンボルを意味する。それ以外のC[*]で書かれる部分は、送信側のデータがそのまま正しく受信されていることを意味する。式(53)は一般的な書き方であり、E[*]はデータ領域の先頭や最後尾、あるいはFEC冗長領域にあっても構わないし、また連続していても構わない。C[128-m]とC[3]の間は、デインタリーバのマトリクス上では、空のデータ領域(0データの部分)があるが、送受信はされないことを意味している。もちろん、m=124の場合は、この空データ領域は存在しない。

【0294】

FEC訂正モジュール10rでの処理を行うまでに、既にシンボル単位にエラーの有無が明確になっている。シンボルの訂正を簡単にするために、式(53)において、E[128-s]、E[128-t]、E[128-u]、E[128-v]は全て値が0であると

【0295】

受信側FEC処理方法1に従って、式(37)のようにシンδροームS[i](i=1, 2, 3, 4)を定義すると、これらは以下のように計算される。

【0296】

手順1: S[i] = 0 (i = 1, 2, 3, 4)とおく。

【0297】

手順2: シンボルが1つ到着する度に、

10

20

30

40

$S[i] \leftarrow S[i] + C[128-k] \cdot \alpha^{*(119+i)(128-k)}$ (到着シンボルが正しい場合)

$S[i] \leftarrow S[i]$ (到着シンボルが誤りの場合)

$$(k=1, 2, \dots, m \text{ または } k=125, 126, 127, 128) \quad (i=1, 2, 3, 4) \quad \dots (54)$$

のように代入していく。

【0298】

手順3：手順2で最後のシンボルまで計算した時の $S[i]$ ($i=1, 2, 3, 4$) の値が求めるシンドローム値である。

【0299】

上記手順において $C[127]$ が到着した時点で、仮に正しいデータ領域シンボルの到着が終わったとすると、

$$S[i] = C[127] \cdot \alpha^{*127(119+i)} + C[3] \cdot \alpha^{*3(119+i)} + C[2] \cdot \alpha^{*2(119+i)} + C[1] \cdot \alpha^{*(119+i)} + C[0] \quad (i=1, 2, 3, 4) \quad \dots (55)$$

となる。これはまさしく、図16にある $C[127]$ のシンボルがデインタリーバの左端にある場合のシンドローム計算の結果を示している。もし正しいデータがこの1シンボルのみであったとしても、その後 FEC 冗長領域部分の値を加えると、本実施例に基づくデインタリーバ構成のシンドローム値となる。すなわち、シンボル到着がいつ終了しても常に正しいシンドローム値が求められるような計算方法となっている。

【0300】

このようにして得られるシンドローム値は、もし受信シンボルに全く誤りがなければ全て0となる。実際は、受信側 FEC 処理方法1の式(45)にならって、

$$S[i] = C[128-s] \cdot (\alpha^{*(119+i)(128-s)}) + C[128-t] \cdot (\alpha^{*(119+i)(128-t)}) + C[128-u] \cdot (\alpha^{*(119+i)(128-u)}) + C[128-v] \cdot (\alpha^{*(119+i)(128-v)}) \quad (i=1, 2, 3, 4) \quad \dots (56)$$

という値を持つことになる。式(54)の手順で求められたシンボル値 $S[i]$ ($i=1, 2, 3, 4$) を使って、式(56)の各シンボル $C[128-k]$ ($k=s, t, u, v$) を求めることが、 FEC 訂正モジュール10rでの誤り訂正処理となる。

【0301】

これらは4次の連立方程式となり、解は、

10

20

30

40

$$C[128-s] = \{ S[4] + (\alpha*(128-t) + \alpha*(128-u) + \alpha*(128-v)) \cdot S[3] \\ + (\alpha*(256-t-u) + \alpha*(256-t-v) + \alpha*(256-u-v)) \cdot S[2] \\ + \alpha*(384-t-u-v) \cdot S[1] \}$$

$$\begin{aligned} & \diagdown \{ \alpha*120(128-s) \cdot (\alpha*(128-t) + \alpha*(128-v)) \\ & \quad \cdot (\alpha*(128-t) + \alpha*(128-u)) \cdot (\alpha*(128-u) + \alpha*(128-v)) \} \end{aligned}$$

$$C[128-t] = \{ S[4] + (\alpha*(128-s) + \alpha*(128-u) + \alpha*(128-v)) \cdot S[3] \\ + (\alpha*(256-s-u) + \alpha*(256-s-v) + \alpha*(256-u-v)) \cdot S[2] \\ + \alpha*(384-s-u-v) \cdot S[1] \}$$

$$\begin{aligned} & \diagdown \{ \alpha*120(128-t) \cdot (\alpha*(128-s) + \alpha*(128-v)) \\ & \quad \cdot (\alpha*(128-s) + \alpha*(128-u)) \cdot (\alpha*(128-u) + \alpha*(128-v)) \} \end{aligned}$$

$$C[128-u] = \{ S[4] + (\alpha*(128-t) + \alpha*(128-s) + \alpha*(128-v)) \cdot S[3] \\ + (\alpha*(256-t-s) + \alpha*(256-t-v) + \alpha*(256-s-v)) \cdot S[2] \\ + \alpha*(384-t-s-v) \cdot S[1] \}$$

$$\begin{aligned} & \diagdown \{ \alpha*120(128-t) \cdot (\alpha*(128-t) + \alpha*(128-v)) \\ & \quad \cdot (\alpha*(128-t) + \alpha*(128-s)) \cdot (\alpha*(128-s) + \alpha*(128-v)) \} \end{aligned}$$

$$C[128-v] = \{ S[4] + (\alpha*(128-t) + \alpha*(128-u) + \alpha*(128-s)) \cdot S[3] \\ + (\alpha*(256-t-u) + \alpha*(256-t-s) + \alpha*(256-u-s)) \cdot S[2] \\ + \alpha*(384-t-u-s) \cdot S[1] \}$$

$$\begin{aligned} & \diagdown \{ \alpha*120(128-v) \cdot (\alpha*(128-t) + \alpha*(128-s)) \\ & \quad \cdot (\alpha*(128-t) + \alpha*(128-u)) \cdot (\alpha*(128-u) + \alpha*(128-s)) \} \end{aligned}$$

… (57)

となる。

【0302】

式(56)において、エラーシンボルがもし1つ(E[128-s])のみであれば、上式は、

$$S[i] = C[128-s] \cdot \alpha*(119+i)(128-s) \quad (i=1, 2, 3, 4) \dots (58)$$

であり、これから、

$$C[128-s] = (\alpha*(119+i)(128-s)) \diagdown S[i] \quad (i=1, 2, 3, 4) \dots (59)$$

となる。これは、iの値によらず同じなので、どれか1つを計算すればよい。すると、式(54)において、例えばS[1]のみ計算すればよく、他のシンドローム計算を行う必要がないので、その分の演算量を減らすことができる。

【0303】

また、もしエラーシンボルが2つ(E[128-s]とE[128-t])であれば、式(45)は、

$$S[i] = C[128-s] \cdot (\alpha*(119+i)(128-s)) + C[128-t] \cdot (\alpha*(119+i)(128-t)) \quad (i=1, 2, 3, 4) \dots (60)$$

となる。

【0304】

これも2つのシンδροーム、例えばS[1]とS[2]のみを計算すればよく、他のシンδροームに対してかかる分の演算を減らすことができる。

【0305】

そして解は、

$$\begin{aligned}
 C[128-s] &= (S[2] + \alpha*(128-t) \cdot S[1]) \\
 &\quad / \{ \alpha*120(128-s) \cdot (\alpha*(128-s) + \alpha*(128-t)) \} \\
 C[128-t] &= (S[2] + \alpha*(128-s) \cdot S[1]) \\
 &\quad / \{ \alpha*120(128-t) \cdot (\alpha*(128-s) + \alpha*(128-t)) \} \\
 &\quad \dots (61)
 \end{aligned}$$

となる。

【0306】

また、エラーシンボルが3つ(E[128-s]とE[128-t]とE[128-u])であるとき、式(56)は、

$$\begin{aligned}
 S[i] &= C[128-s] \cdot (\alpha*(119+i)(128-s)) + C[128-t] \cdot (\alpha*(119+i) \\
 (128-t)) &+ C[128-u] \cdot (\alpha*(119+i)(128-u)) \quad (i=1, 2, 3, 4) \\
 &\quad \dots (62)
 \end{aligned}$$

となり、この場合は3つのシンδροーム、例えばS[1]、S[2]、S[3]のみを計算すればよいので、S[4]に対してかかる演算の分を減らすことができる。

【0307】

解は、

$$\begin{aligned}
 C[128-s] &= \{ S[3] + (\alpha*(128-t) + \alpha*(128-u)) \cdot S[2] \\
 &\quad + \alpha*(256-t-u) \cdot S[1] \} \\
 &\quad / \{ \alpha*120(128-s) \cdot (\alpha*(128-s) + \alpha(128-t)) \\
 &\quad \cdot (\alpha*(128-s) + \alpha*(128-u)) \} \\
 C[128-t] &= \{ S[3] + (\alpha*(128-s) + \alpha*(128-u)) \cdot S[2] \\
 &\quad + \alpha*(256-s-u) \cdot S[1] \} \\
 &\quad / \{ \alpha*120(128-t) \cdot (\alpha*(128-s) + \alpha(128-t)) \\
 &\quad \cdot (\alpha*(128-t) + \alpha*(128-u)) \} \\
 C[128-u] &= \{ S[3] + (\alpha*(128-t) + \alpha*(128-s)) \cdot S[2] \\
 &\quad + \alpha*(256-t-s) \cdot S[1] \} \\
 &\quad / \{ \alpha*120(128-u) \cdot (\alpha*(128-u) + \alpha(128-t)) \\
 &\quad \cdot (\alpha*(128-s) + \alpha*(128-u)) \} \quad \dots (63)
 \end{aligned}$$

となる。

【0308】

これらの演算を行う場合には、 $\alpha^{(119+i)(128-k)}$ ($k=1 \sim 128$)、($i=1, 2, 3, 4$)の演算に多少の時間がかかってしまうので、 i および k をパラメータとした表をあらかじめ作成しておく、毎回のべき乗を計算するよりもより高

速に演算ができる。

【0309】

(6) 受信側 F E C 処理方法 3

ここでは、送信側 F E C 処理方法 3 で述べたインタリーバに対するデインタリーバを用いた誤り訂正方法を示す。

【0310】

受信側 F E C 処理方法 1、2 と同様にエラーシンボルは最大 4 つまで訂正可能である。そこで、それらのエラーシンボル位置をそれぞれ s 番目、t 番目、u 番目、s 番目 ($0 < s < t < u < v < (m + 5)$) とする。

【0311】

図 5 の場合は、到着したシンボル毎にデータ領域のシンボルはランダムに、また F E C 冗長領域のシンボルは、受信側 F E C 処理方法 1、2 と同様にデインタリーバの右端 4 列に位置する。この場合も、デインタリーバによっていつシンボル受信が終了するかわからない。よって、そのような可変シンボル長に柔軟に対応できるような復号処理を行う必要がある。

【0312】

一般的には、エラーがのった受信シンボルは式 (34) で、エラーフリーの場合の受信シンボルは式 (35) で夫々表されるが、以下では、図 5 にある具体例を用いて説明する。

【0313】

まず、図 5 (b) についてみると、全てのシンボルが正しく受信できたときの受信系列 C は、

$$C = (C[8], C[7], \dots, C[0]) \quad \dots (64)$$

であり、そのうち C[3]、C[2]、C[1]、C[0] は冗長シンボルである。これを多項式表現すると、

$$C(X) = C[8] \cdot X^6 + C[7] \cdot X^9 + C[6] \cdot X^5 + C[5] \cdot X^7 + C[4] \cdot X^{10} + C[3] \cdot X^3 + C[2] \cdot X^2 + C[1] \cdot X + C[0] \quad \dots (65)$$

となる。この例では 9 つのシンボルを受信するが、実際には最大 128 シンボルまで受信することがあり得る。これらのシンボルのそれぞれについて、何番目のシンボルがどの列に属するか (すなわち式 (65) を作成するにあたって、各シンボルに対応する X のべき乗の値は何であるか) といった一覧表を、送信側と同様に保持している必要がある。

【0314】

さて、式 (65) のシンボルのうち、最大 4 つのシンボルがエラーを起こしているとする。ただし、冗長シンボルのみのエラーである場合は、誤り訂正処理を行わないので、ここでは、少なくとも 1 つのエラーシンボルは、データ領域に属するシンボルであるとする。

【0315】

受信側 F E C 処理方法 1、2 と同様にエラー数に応じて、シンドローム $S[i]$ の計算を行う。エラーシンボル数と同じだけのシンドロームの計算を行えば、誤り訂正が可能である。シンドロームの計算は、

手順 1 : $S[i] = 0$ ($i = 1, 2, 3, 4$) とおく。

【0316】

手順 2 : 受信シンボルに応じて、以下の計算を行う。

【0317】

$S[i] \leftarrow S[i] + C[k] \cdot (\alpha^{*(119+i)} f(k))$ (受信シンボルが正しい
場合)

$S[i] \leftarrow S[i]$ (受信シンボルが誤りの
場合)

($i=1, 2, 3, 4$) ... (66)

手順3：最後のシンボルが終了すると、その時の $S[i]$ ($i = 1, 2, 3, 4$) が求めるシンドロームの値である。

【0318】

手順2において、 $f(k)$ は $C[k]$ と対になる、多項式上のべき乗数を示す。例えば図5(b)において、 $C[8]$ に相当するべき乗数は $f(8) = 6$ である。

【0319】

上記演算により得られる $S[i]$ ($i = 1, 2, 3, 4$) の別の表現は、式(45)の変形で与えられる。誤りのあるシンボルの訂正後の値を $C[m-k]$ ($k = s, t, v, u$) とすると、本実施例の具体例では、 $m = 8$ なので、

$S[i] = C[8-s] \cdot (\alpha^{*(119+i)} f(8-s)) + C[8-t] \cdot (\alpha^{*(119+i)} f(8-t)) + C[8-u] \cdot (\alpha^{*(119+i)} f(8-u)) + C[8-v] \cdot (\alpha^{*(119+i)} f(8-v))$
($i=1, 2, 3, 4$) ... (67)

となる。

【0320】

式(66)と式(67)から、4次の連立方程式ができ、これから $C[8-k]$ ($k = s, t, u, v$) の値を求めることができる。エラーシンボルが4つあるときの解は、それぞれ

10

20

$$\begin{aligned}
C[8-s] &= \{ S[4] + (\alpha * f(8-t) + \alpha * f(8-u) + \alpha * f(8-v)) \cdot S[3] \\
&\quad + (\alpha * (f(8-t)+f(8-u)) + \alpha * (f(8-t)+f(8-v)) + \alpha * (f(8-u)+f(8-v) \\
&\quad)) \\
&\quad \cdot S[2] + \alpha * (f(8-t)+f(8-u)+f(8-v)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-s) \cdot (\alpha * f(8-t) + \alpha * f(8-v)) \cdot (\alpha * f(8-t) + \\
&\quad \alpha * f(8-u)) \\
&\quad \cdot (\alpha * f(8-u) + \alpha * f(8-v)) \}
\end{aligned}$$

10

$$\begin{aligned}
C[8-t] &= \{ S[4] + (\alpha * f(8-s) + \alpha * f(8-u) + \alpha * f(8-v)) \cdot S[3] \\
&\quad + (\alpha * (f(8-s)+f(8-u)) + \alpha * (f(8-s)+f(8-v)) + \alpha * (f(8-u)+f(8-v) \\
&\quad)) \\
&\quad \cdot S[2] + \alpha * (f(8-s)+f(8-u)+f(8-v)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-t) \cdot (\alpha * f(8-s) + \alpha * f(8-v)) \cdot (\alpha * f(8-s) + \\
&\quad \alpha * f(8-u)) \\
&\quad \cdot (\alpha * f(8-u) + \alpha * f(8-v)) \}
\end{aligned}$$

20

$$\begin{aligned}
C[8-u] &= \{ S[4] + (\alpha * f(8-t) + \alpha * f(8-s) + \alpha * f(8-v)) \cdot S[3] \\
&\quad + (\alpha * (f(8-t)+f(8-s)) + \alpha * (f(8-t)+f(8-v)) + \alpha * (f(8-s)+f(8-v) \\
&\quad)) \\
&\quad \cdot S[2] + \alpha * (f(8-t)+f(8-s)+f(8-v)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-t) \cdot (\alpha * f(8-t) + \alpha * f(8-v)) \cdot (\alpha * f(8-t) + \\
&\quad \alpha * f(8-s)) \\
&\quad \cdot (\alpha * f(8-s) + \alpha * f(8-v)) \}
\end{aligned}$$

30

$$\begin{aligned}
C[8-v] &= \{ S[4] + (\alpha * f(8-t) + \alpha * f(8-u) + \alpha * f(8-s)) \cdot S[3] \\
&\quad + (\alpha * (f(8-t)+f(8-u)) + \alpha * (f(8-t)+f(8-s)) + \alpha * (f(8-u)+f(8-s) \\
&\quad)) \\
&\quad \cdot S[2] + \alpha * (f(8-t)+f(8-u)+f(8-s)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-v) \cdot (\alpha * f(8-t) + \alpha * f(8-s)) \cdot (\alpha * f(8-t) + \\
&\quad \alpha * f(8-u)) \\
&\quad \cdot (\alpha * f(8-u) + \alpha * f(8-s)) \} \quad \dots (68)
\end{aligned}$$

40

となる。

【 0 3 2 1 】

式(67)において、エラーシンボルがもし1つ(s番目)のみであれば、上式は、

$$S[i] = C[8-s] \cdot \alpha * (119+i)f(8-s) \quad (i=1, 2, 3, 4) \dots (69)$$

であり、これから、

$$C[8-s] = (\alpha * (119+i)f(8-s)) / S[i] \quad (i=1, 2, 3, 4) \dots (70)$$

となる。これは、iの値によらず同じなので、どれか1つを計算すればよい。とすると、

50

例えば $S[1]$ のみ計算すればよく、他のシンドロームに対する計算を行わずに済むことになる。

【0322】

また、もしエラーシンボルが2つ (s 番目と t 番目) であれば、式(67)は、

$$\begin{aligned} S[i] = & C[8-s] \cdot (\alpha^{119+i} f(8-s)) \\ & + C[8-t] \cdot (\alpha^{119+i} f(8-t)) \end{aligned} \quad (i=1, 2, 3, 4) \quad \dots (71)$$

となる。これも2つのシンドローム、例えば $S[1]$ と $S[2]$ のみを計算すればよく、他のシンドロームに対する演算を行わずに済むことになる。 10

【0323】

そして、解は、

$$\begin{aligned} C[8-s] = & (S[2] + \alpha f(8-t) \cdot S[1]) \\ & / \{ \alpha^{120} f(8-s) \cdot (\alpha f(8-s) + \alpha f(8-t)) \} \\ C[8-t] = & (S[2] + \alpha f(8-s) \cdot S[1]) \\ & / \{ \alpha^{120} f(8-t) \cdot (\alpha f(8-s) + \alpha f(8-t)) \} \end{aligned} \quad \dots (72) \quad 20$$

となる。

【0324】

また、エラーシンボルが3つ (s 番目と t 番目と u 番目) であるとき、式(67)は、

$$\begin{aligned} S[i] = & C[8-s] \cdot (\alpha^{119+i} f(8-s)) \\ & + C[8-t] \cdot (\alpha^{119+i} f(8-t)) \\ & + C[8-u] \cdot (\alpha^{119+i} f(8-u)) \end{aligned} \quad (i=1, 2, 3, 4) \quad \dots (73) \quad 30$$

となり、この場合は3つのシンドローム、例えば $S[1]$ 、 $S[2]$ 、 $S[3]$ のみを計算すればよい。すなわち $S[4]$ の演算を必要とせずに済む。

【0325】

このときの解は、

$$\begin{aligned}
C[8-s] &= \{ S[3] + (\alpha * f(8-t) + \alpha * f(8-u)) \cdot S[2] \\
&\quad + \alpha * (f(8-t) + f(8-u)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-s) \cdot (\alpha * f(8-s) + \alpha * f(8-t)) \\
&\quad \cdot (\alpha * f(8-s) + \alpha * f(8-u)) \} \\
C[8-t] &= \{ S[3] + (\alpha * f(8-s) + \alpha * f(8-u)) \cdot S[2] \\
&\quad + \alpha * (f(8-s) + f(8-u)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-t) \cdot (\alpha * f(8-s) + \alpha * f(8-t)) \\
&\quad \cdot (\alpha * f(8-t) + \alpha * f(8-u)) \} \\
C[8-u] &= \{ S[3] + (\alpha * f(8-t) + \alpha * f(8-s)) \cdot S[2] \\
&\quad + \alpha * (f(8-t) + f(8-s)) \cdot S[1] \} \\
&\quad / \{ \alpha * 120f(8-u) \cdot (\alpha * f(8-u) + \alpha * f(8-t)) \\
&\quad \cdot (\alpha * f(8-s) + \alpha * f(8-u)) \} \quad \dots (74)
\end{aligned}$$

10

となる。

【0326】

20

これらの演算には、 α のべき乗の計算が多数必要となる。常に出てくる演算（例えば α^{120} 等）に関しては、表を作っておくと高速化を図ることができる。

【0327】

次に、図5(a)の場合を説明する。これは基本的に図5(b)と同じであるので、図に示した具体例のみを説明することとする。

【0328】

図5(a)の場合は、受信側でも送信側と同じ表2を持っていて、シンボル到着の度に、そのシンボルがデインタリーバのどの位置にくるのかを把握しておく必要がある。ここでも、送信側と同じく上から4行目を例にとり、この行がもし全くエラーがないとすると、受信シンボルを使った多項式は、式(32)から

30

$$\begin{aligned}
C(X) &= C[1] \cdot X^7 + C[11] \cdot X^5 + C[18] \cdot X^8 + C[19] \cdot X^9 \\
&\quad + C[125] \cdot X^3 + C[126] \cdot X^2 + C[127] \cdot X + C[128] \\
&\quad \dots (75)
\end{aligned}$$

と表される。本来は、何番目のシンボルまで受信するかは、あらかじめ送信側から知らされることはなく、CPCS-PDUの終わりを検出することによって、初めて知ることになる。

【0329】

この場合も、最大4シンボルのエラーを想定する。シンδροームの計算方法は、式(66)にある手順と全く同じである。ただし、 k の値は式(75)にある $C[k]$ から取り出されているので、ランダムな値をとる。そして、 $f(k)$ は式(75)において、その k の値に対する $C[k]$ を係数とする X の次数を表す。例えば、 $k=1$ に対して、 $f(1)=7$ である。

40

【0330】

エラーがある場合のシンδροームの値は0にはならず、式(67)にならって記述すると

$$S[i] = C[s] \cdot (\alpha^{(119+i)} f(s)) + C[t] \cdot (\alpha^{(119+i)} f(t)) \\ + C[u] \cdot (\alpha^{(119+i)} f(u)) + C[v] \cdot (\alpha^{(119+i)} f(v)) \\ (i=1, 2, 3, 4) \quad \dots (76)$$

となる。ただし、 $0 < s < t < u < v$ であり、 $\{s, t, u, v \mid k \mid k \text{ は 4 行目に属するシンボル番号}\}$ であるとする。そして、 s, t, u, v 番目のシンボルにエラーが発生しているとする。また、 $C[k]$ ($k = s, t, u, v$) は、それぞれのシンボルの訂正後の値を意味する。

【0331】

式(66)と式(76)を解くことによって、訂正後のシンボル値が計算できる。まず、4つのエラーがある場合は、

$$C[s] = \{S[4] + (\alpha f(t) + \alpha f(u) + \alpha f(v)) \cdot S[3] \\ + (\alpha(f(t)+f(u)) + \alpha(f(t)+f(v)) + \alpha(f(u)+f(v))) \cdot S[2] \\ + \alpha(f(t)+f(u)+f(v)) \cdot S[1]\} \\ / \{\alpha^{120} f(s) \cdot (\alpha f(t) + \alpha f(v)) \cdot (\alpha f(t) + \alpha f(u)) \\ \cdot (\alpha f(u) + \alpha f(v))\}$$

$$C[t] = \{S[4] + (\alpha f(s) + \alpha f(u) + \alpha f(v)) \cdot S[3] \\ + (\alpha(f(s)+f(u)) + \alpha(f(s)+f(v)) + \alpha(f(u)+f(v))) \cdot S[2] \\ + \alpha(f(s)+f(u)+f(v)) \cdot S[1]\} \\ / \{\alpha^{120} f(t) \cdot (\alpha f(s) + \alpha f(v)) \cdot (\alpha f(s) + \alpha f(u)) \\ \cdot (\alpha f(u) + \alpha f(v))\}$$

$$C[u] = \{S[4] + (\alpha f(t) + \alpha f(s) + \alpha f(v)) \cdot S[3] \\ + (\alpha(f(t)+f(s)) + \alpha(f(t)+f(v)) + \alpha(f(s)+f(v))) \cdot S[2] \\ + \alpha(f(t)+f(s)+f(v)) \cdot S[1]\} \\ / \{\alpha^{120} f(u) \cdot (\alpha f(t) + \alpha f(v)) \cdot (\alpha f(t) + \alpha f(s)) \\ \cdot (\alpha f(s) + \alpha f(v))\}$$

$$C[v] = \{S[4] + (\alpha f(t) + \alpha f(u) + \alpha f(s)) \cdot S[3] \\ + (\alpha(f(t)+f(u)) + \alpha(f(t)+f(s)) + \alpha(f(u)+f(s))) \cdot S[2] \\ + \alpha(f(t)+f(u)+f(s)) \cdot S[1]\} \\ / \{\alpha^{120} f(v) \cdot (\alpha f(t) + \alpha f(s)) \cdot (\alpha f(t) + \alpha f(u)) \\ \cdot (\alpha f(u) + \alpha f(s))\}$$

… (77)

となる。

【0332】

式(67)において、エラーシンボルがもし1つ(s 番目)のみであれば、上式は、

$$S[i] = C[s] \cdot \alpha^{(119+i)} f(s) \quad (i=1, 2, 3, 4) \quad \dots (78)$$

であり、これから、

10

20

30

40

$$C[s] = (\alpha * (119+i)f(s)) / S[i] \quad (i=1, 2, 3, 4) \dots (79)$$

となる。これは i の値によらず同じなので、どれか 1 つを計算すればよい。とすると、例えば $S[1]$ のみ計算すればよく、他のシンδροーム計算を行う必要がないので、その分の演算量を減らすことができる。

【0333】

また、もしエラーシンボルが 2 つ (s 番目と t 番目) であれば、式 (67) は

$$S[i] = C[s] \cdot (\alpha * (119+i)f(s)) + C[t] \cdot (\alpha * (119+i)f(t)) \quad (i=1, 2, 3, 4) \dots (80)$$

10

となる。これも 2 つのシンδροーム、例えば $S[1]$ と $S[2]$ のみを計算すればよく、他のシンδροームに対する演算を行わなくて済む。

【0334】

そして、解は、

$$C[s] = (S[2] + \alpha * f(t) \cdot S[1]) / \{ \alpha * 120f(s) \cdot (\alpha * f(s) + \alpha * f(t)) \}$$

$$C[t] = (S[2] + \alpha * f(s) \cdot S[1]) / \{ \alpha * 120f(t) \cdot (\alpha * f(s) + \alpha * f(t)) \} \quad \dots (81)$$

20

となる。

【0335】

また、エラーシンボルが 3 つ (s 番目と t 番目と u 番目) であるとき、式 (67) は、

$$S[i] = C[s] \cdot (\alpha * (119+i)f(s)) + C[t] \cdot (\alpha * (119+i)f(t)) + C[u] \cdot (\alpha * (119+i)f(u)) \quad (i=1, 2, 3, 4) \dots (82)$$

となり、この場合は 3 つのシンδροーム、例えば $S[1]$ 、 $S[2]$ 、 $S[3]$ のみを計算すればよい。すなわち $S[4]$ の演算を行わなくて済む。

30

【0336】

このときの解は、

$$\begin{aligned} C[s] = & \{ S[3] + (\alpha * f(t) + \alpha * f(u)) \cdot S[2] + \alpha * (f(t) + f(u)) \cdot S \\ [1] \} \\ & \diagdown \{ \alpha * 120f(s) \cdot (\alpha * f(s) + \alpha * f(t)) \cdot (\alpha * f(s) + \alpha * f(u)) \} \\ & \} \end{aligned}$$

$$\begin{aligned} C[t] = & \{ S[3] + (\alpha * f(s) + \alpha * f(u)) \cdot S[2] + \alpha * (f(s) + f(u)) \cdot S \\ [1] \} \\ & \diagdown \{ \alpha * 120f(t) \cdot (\alpha * f(s) + \alpha * f(t)) \cdot (\alpha * f(t) + \alpha * f(u)) \} \end{aligned}$$

10

$$\begin{aligned} C[u] = & \{ S[3] + (\alpha * f(t) + \alpha * f(s)) \cdot S[2] + \alpha * (f(t) + f(s)) \cdot \\ S[1] \} \\ & \diagdown \{ \alpha * 120f(u) \cdot (\alpha * f(u) + \alpha * f(t)) \cdot (\alpha * f(s) + \alpha * f(u)) \} \\ & \} \end{aligned}$$

… (83)

となる。

20

【0337】

本実施例では、データの信頼性を保証するためのプロトコルとしてはAALタイプ5のSSCSレイヤのFEC処理以外を想定していない。しかしながら、実際にはFECの能力には限界があり、一定数以上セルが廃棄された場合にはもはやFECによる誤り訂正は不可能となってしまう。このような状況では再送による誤り訂正を行なうしかなく、そのためには予めSSCSレイヤのFECによる誤り訂正の他に上位レイヤにライトウェイトな再送制御を実装しなければならない。FECで訂正できない場合、元のデータを得るには再送を行なうしかないので、本実施例では、SSCSの上位にライトウェイトな再送制御のためのプロトコルは実装していると想定している。また、予めATM網の状態をみて誤りの多く発生する状態であれば能力の高いFEC用符号を用いる、またほとんど誤りの発生しない状態であれば軽いFEC符号を用いるなど、パラメータを選択した上で通信を開始できる。

30

【0338】

なお、前述のように、本実施例ではAALタイプ5に関するものについて説明したが、本発明の誤り訂正方法はデータ転送に限定されるものではなく、同様の方法で他のタイプにも適用可能である。

【0339】

逆に、例えば、AAL5機能のみが実装された端末装置において、該端末装置がマルチメディア/コンティニューアスメディア処理機能（音声、動画等）を持っているとすれば、このようなアプリケーションを実現する場合にAAL5の処理機能に本実施例の誤り制御方式を併用させることによって柔軟に対処し様々な状況で様々なデータの信頼性を保証することができるのである。

40

【0340】

(第2の実施例)

次に、第2の実施例を説明する。第1の実施例は、1対1の端末が通信を行なう場合の例であったが、第2の実施例では、1対多通信の場合について説明する。

【0341】

本実施例では、図24に示すように、1台の送信側端末2tが、複数台の受信側端末2r (2r - 1 ~ 2r - n) に対して通信を行なっているものとする。なお、図中の受信側端末2rのいずれか複数個が受信側端末かつ送信側端末となり、同じようなコネクションを

50

張り同じように1対多通信を行なうことによって多対多通信を実現することもできるが、このような場合にも、本発明を適用することが可能である。

【0342】

以下、FECを付与しマルチキャストを行なう場合の例について説明する。

【0343】

図25は、本実施例の送受信端末のプロトコルスタックの一例である。基本的な構成は、第1の実施例のプロトコルスタックとほぼ同様である。図1と異なる点は、本実施例では、いずれかのレイヤにマルチキャスト機能を提供する能力のあるプロトコルが必要になることである。

【0344】

一般的に広く知られたマルチキャスト機能を提供できるプロトコルとしては、マルチキャストIPアドレスを使ったIP転送などがある。現状のシステムでは、IPマルチキャストを行なう場合に、その上位レイヤプロトコルとしてUDP(User Datagram Protocol)が用いられるのが通常である。UDPはTCP(Transmission Control Protocol)と異なり、エンド-エンドに信頼性の保証を行なわないプロトコルであるので、転送途中でパケットが廃棄されても、ビット誤り等が生じて、訂正は行なわない。しかしながら、ここで転送しようとしているアプリケーションデータはマルチメディアで放送形態をとるデジタルメディアを想定しているので、IP+UDPのようなデータの信頼性を保証しないプロトコルでは不都合である。そこで、本実施例では、MTP(Multicast Transport Protocol)やRMP(Reliable Multicast Protocol)をレイヤ4の

10

20

【0345】

MTPは、IPでマルチキャストを行なう場合、転送途中に生じた誤りを訂正する機能を持ったプロトコルである。現在はインターネットドラフトのRFC1301に実現の方法に関する詳細が記述されている。該ドラフトの誤り訂正の方法とは、受信者による再送要求により送信者がデータを再送するという方法である。RMPは、カリフォルニア大学バークレイ校のB. Wettenらにより提案されたものであり、MTPとほぼ同様の動作及び機能を持ったプロトコルである。しかし、両者ともCS-CWなどをアプリケーション

30

【0346】

しかしながら、例で挙げたようなアプリケーションを実現するにはリアルタイム性の保持を行なう必要がある。その場合には、受信者が誤りを検出してから送信者に対して再送要求を行っていたのでは、遅延が許容範囲を越えてしまう可能性が高い。このような場合にも、前述した第1の実施例のように、転送するデータに予め誤り訂正を行なうための冗長な符号を付与しておくことによって、上記のような問題点を解消できるのである。

【0347】

さて、IPのような上位レイヤで提供されるマルチキャストは、ATMレベルでは図25のようにスイッチ3のコピー機能などを用いて実現される。また、接続の張り方としては、主に以下の2通りが考えられる。

40

【0348】

(1) 送信側端末から全ての受信側端末に対してポイント-マルチポイントのATM接続を張る。

【0349】

(2) 送信側端末から一部の受信側端末と一部の中継サーバに対してポイント-マルチポイントのATM接続を張る。中継サーバよりも(マルチキャストツリー内で)下流の受信側端末と中継サーバに関しては、該中継サーバが担当する。中継サーバとは、物理的または論理的にネットワークやサブネットを接続しているような装置を指し、AALよりも上位のレイヤで接続を一度終端することもある。

50

【0350】

上記2の例を図26に示す。Hは送信側端末(2t)、H1~H25は受信側端末(2r)、S1~S5は中継サーバである。ネットワーク1-1~1-7は、ATM網である。図26のように、送信側端末Hから受信側端末H1、H2、および中継サーバS1、S2、S3に対して、コネクションを設定し、中継サーバS1、S2、S3は、それぞれの下流に位置するATM網1-2、1-3、1-4内のマルチキャストグループに属する受信側端末やさらに下流にメンバーが存在する中継サーバに対して同じようにコネクションを張る。

【0351】

上記1の方法は、受信側端末の総数が比較的少ない場合には適切な方法である。これに対して、上記2の方法は受信側端末の総数がかなり多い場合でも、階層的に下流の中継サーバにコネクションを一度終端させることで、広域に渡るマルチキャストが実現できる方法である。

10

【0352】

次に、送信側端末2tおよび受信側端末2rを説明する。図27に、本実施例の送信側端末2tおよび受信側端末2rの内部構成を示す。本実施例の送信側端末2tおよび受信側端末2rの内部構成は、基本的には、第1の実施例とほぼ同様である。したがって、以下では、本実施例の送信側端末2tおよび受信側端末2rについて、第1の実施例との相違点を中心として説明する。なお、ここでも、AALとしてタイプ5を例にあげる。

【0353】

20

まず、送信側端末2tのアプリケーション処理モジュール104t、MTP/RMP処理を行うMTP/RMP処理モジュール130t、IP処理を行うIP処理モジュール131tでは、マルチキャストしたい送信側端末2rに対して、それぞれのプロトコルに特有な処理が行なわれる。データの種類としては、動画、音声、テキストデータなどATMの特徴を活かして様々なメディアを転送するが、ここまでの処理は、本発明に限られた方法ではない。

【0354】

次に、SSCS処理モジュール106t、CPCS処理モジュール114t、SAR処理モジュール120tでは、第1の実施例で説明したものと同様の処理が施される。

【0355】

30

ただし、FECを行なうSSCSレイヤで、上位レイヤに対してエラーフリーな通信を提供する場合、そのためには、SSCSレイヤ以下のレイヤでトランスペアレント(透過的)なデータの受渡しが可能であることが必要となる。というのは、送信側のSSCSレイヤのエンティティに対して、受信側のSSCSレイヤのエンティティから再送要求が送られた場合に、セル単位の再送を行なうためには、SSCSレイヤで、個々のセルの識別ができることが必須になるからである。例えば、SSCSレイヤでセル単位の再送を行なう場合、再送要求、あるいは再送に関係あるデータはSSCSレイヤよりも下のレイヤではトランスペアレントに受渡しされなければならない。これらは、受信側端末2rの処理の説明において詳しく述べる。SSCSレイヤ以下に関しても、第1の実施例と同様の処理が行なわれる。

40

【0356】

次に、受信側端末2rについて説明する。

【0357】

マルチキャストを行なう場合には、多くの宛先に対してメッセージを送信するため、いずれかの受信側端末2rで正しくデータを受信できない可能性が高くなる。言い換えると、送信側にいずれかの送信側端末2tから再送要求が送られる確率が高くなる。しかし、本発明によれば、予めFECを付与するレイヤ(AAL5の場合であればSSCSレイヤ)に、データの信頼性を保証する機能を持たせることで、その確率を激減させることができるのである。

【0358】

50

受信側端末 2 r での誤り検出は、まず、CPCS 処理モジュール 114 r で行なわれる。その際、誤りが検出されなければ、第 1 の実施例で示したように、SSCS 処理モジュール 106 r を省略して上位レイヤにデータを渡す方法も可能である。その場合には、送信側で付与した FEC 冗長符号のオーバーヘッド程度ですみ、スループット、レイテンシは、AAL5 にかなり近い性能を得ることができる。

【0359】

前述したように、マルチキャストの場合、IP レベルでマルチキャストグループに該当する複数の IP アドレスに対してデータグラムを送出するので、一般的な IP マルチキャストでは、送信側端末 2 t で再送要求を受けとる確率がポイント - ポイントの場合よりも大きい。マルチキャストグループに属する端末数が大きければ大きいほどその確率は大きくなる。例えば、受信側端末数が 10^4 、データリンクの誤り率が無線程度に大きい場合には、1 に近い確率で送信側端末 2 t は再送要求を受けとる。

10

【0360】

誤りのレベルとしては、大きく 2 種類に分けられる。

- (1) 予め付与した誤り訂正符号 (FEC コード) により訂正可能な程度である場合
- (2) 上記 1 のレベルを上回り、FEC コードによる訂正が不可能、よって、再送以外に訂正は無理である場合

もちろん、上記 1 の場合には、第 1 の実施例の 1 対 1 通信の場合と同様に誤り訂正を行なって、正しいデータを上位レイヤに渡すことができる。

【0361】

しかしながら、上記 2 のケースが発生した場合には再送により誤りを訂正しなければならず、その方法としては以下のような候補がある。

A) AAL よりも上位レイヤで再送制御を行なう (例えば、図 25 のレイヤ 4 に相当する MTP/RMP 等)。

B) AAL のいずれかのレイヤで再送制御を行なう。

20

【0362】

上記 A のケースでは、それぞれの再送プロトコルの手順にしたがって誤り訂正が行なわれるので、第 1 の実施例で説明した手順をもちることができるが、上記 B の方法で再送制御を行なうには、第 1 の実施例で説明した手順に加え、いくつかの処理が必要になる。以下、上記 2 のようなレベルの誤りが生じ、それを上記 B のような方法で誤り回復をはかろうとした場合の手順について説明する。

30

【0363】

FEC の訂正能力を上回るような誤りが転送中に生じた場合に、SSCS レイヤで信頼性保証のために再送制御を行なう方法について説明する。再送制御を行なうとすると、再送の単位は様々な方法が可能である。

- 1) ATM レイヤレベルのセル再送
- 2) CPCS - PDU を単位とする再送
- 3) SSCS - PDU (インタリーブ) を単位とする再送
- 4) SSCS - PDU (インタリーブ) 内の列 (任意の列数) を単位とする再送
- 5) レイヤ 3 以上のパケットを単位とする再送

40

ここでは、一例として、マルチキャストを行う場合で、かつ、上記 3) の SSCS - PDU (インタリーブ) 内の列 (任意の列数) を単位として再送により誤り訂正を行なうケースについて述べる。

【0364】

第 1 の実施例で述べたように、送信側端末 2 t の SSCS 処理モジュール 106 t では、セル単位に SN を含む SSCS ヘッダが付与されている。受信側端末 2 r の SSCS 処理モジュール 106 r では、この SSCS ヘッダに従ってセル廃棄されたセルを特定するのである。これ以前で、何らかの誤りが生じているという事実は、CPCS レイヤの CRC により検出済みである。検出した誤りが広範囲に及んでおり、送信時に付与された FEC の訂正能力を越えている場合には、送信側の SSCS レイヤに対して、再送要求を出す。

50

再送要求のメッセージは、マルチキャストグループのメンバーである受信側端末のいずれかから送られることが予想され、その台数はわからない。1台だけかも知れないし、または数台から送られることもある。

【0365】

例えば、受信側端末 2 r - 1 のみが訂正能力以上の誤りを検出した場合、送信側端末 2 t の S S C S 処理モジュール 1 0 6 t に対して、受信側端末 2 r - 1 は少なくとも自分のアドレス、またはマルチキャストアドレス、再送して欲しいセルの S N を送ることによって誤り訂正を実現する（図 2 8 参照）。

【0366】

受信側端末 2 r - 1 が送信した再送要求は、送信側端末 2 t の S S C S 処理モジュール 1 0 6 t に届けられる。送信側端末 2 t では再送要求メッセージの内容に応じて、それに対応する S N を持つセルを再送する。その際、マルチキャストコネクションとは別に設定されたポイント - ポイントコネクションを利用して、再送要求を送信した受信側端末にのみ再送データを送信しても構わないし、再度マルチキャストを行なっても構わない。

【0367】

廃棄されたセルを S S C S - P D U 内のどの列に対応するかということ S N から特定できても、この例では A A L 5 を前提としているので、通常、その廃棄されたセルのみを転送することはできない。A A L 5 のトレイラが付与され、1セル分の実データでも2セル分の A A L - P D U となってしまう。再送データを転送する方法としては、以下のようなものが可能である。

(1) 実データを独立したインタリーバに格納し直して、新たに F E C を付与し、独立した A A L - P D U として送出する。この場合は、廃棄された時の S N も新たな S S C S ヘッダ/トレイラに記す必要がある。

(2) A A L をヌルにして送出する。S S C S ヘッダ/トレイラは以前のものを付与する。

(3) 再送要求を受けとった時点で、その時、またはその次に転送するインタリーバの最初または最後に含ませてしまう。他のデータと混ざらないように、再送データであることの表示が必要。

【0368】

また、上記の方法での、再送の宛先としては、以下のそれぞれがある。

A) マルチキャストグループに属する全ての受信側端末（マルチキャストによって転送する）

B) 再送要求を送ってきた受信側端末のみ（何らかの方法でポイント - ポイントのコネクションを設定する必要がある）

C) 再送用のコネクションを個々の受信側端末に対して予め張っておいて、再送要求を送ってきた受信側端末に対してのみ再送する。S S C S ヘッダ/トレイラは以前のものを付与する。

D) マルチキャストグループ内の一部

上記 1 の場合には、廃棄されたセル数が小さいと、F E C コードの冗長性が高くなってしまふ可能性がある。

【0369】

上記 2 の場合には、最小限のデータ量を網に送出することになりオーバーヘッドが小さい。

【0370】

上記 A のように、マルチキャストで再送データを送出すると、特定の受信側端末だけにコネクションを張り直すという処理を送信側端末が行なわなくて済むという利点がある。

【0371】

マルチキャストデータに再送セルを含ませる場合、受信したデータが再送データであるという識別は、S S C S ヘッダ/トレイラに再送データであることを示すビットを立てるようにすれば良い。すでに正しく受信できている場合には受信側の S S C S 処理モジュール

10

20

30

40

50

106rで廃棄してしまえば良いので実現は比較的簡単である。

【0372】

図18のSSCSヘッダ/トレイラの例にはそのようなフィールドがなく、SSCSレイヤでエラーフリーな通信を提供する場合には、図29のように拡張したSSCSヘッダ/トレイラが通常の転送時から必要になる。

【0373】

(1)は、図18では1バイトであったSSCSヘッダ/トレイラに、再送データであることを表示するフィールドを1ビット付加した。

【0374】

(2)は、バイト単位が望ましいという見地から、SSCSヘッダ/トレイラのSNをより大きな値までとれるようにし、インタリーバ内のデータ領域と冗長コードの領域がより確実に識別できるようフィールドを増やした。

10

【0375】

(3)は、(2)とは逆に、SNフィールドを小さくすることによって1バイトのSSCSヘッダ/トレイラ内に再送データ表示フィールドを1ビット設けた。

【0376】

再送セル数は、FECの訂正能力に依存して減らすことが可能である。例えば、FECの訂正能力が128セル中4セルだった場合、再送要求で8個のセルを再送するようにリクエストされたとする。このFECコードでは4セル以内の廃棄ならば訂正できるので、再送するセル数は少なくとも4セル送ってやれば済む。このように、最小限のセルを送ることによって、網の輻輳を引き起こさないように防止することができる。このような利点も、FECを用いている場合の特徴の一つである。

20

【0377】

次に、送信側端末に複数の受信側端末から再送要求が送られてきた場合について説明する。ここでは、FECの訂正能力としては、1インタリーバのサイズが16セル、その内、1セルまでの訂正が可能なコードであるものとして説明する。

【0378】

廃棄されたセルを全て再送する方法の他に、次のようにして再送セルの数を削減できる方法がある。

【0379】

図30のように、受信側端末 $2r-1$ 、 $2r-3$ から各々次のようなセルの再送要求が送信側端末 $2t$ のSSCS処理モジュール106tに到着したとする。

30

受信側端末 $2r-1$ セルのSN... 1, 2, 3

受信側端末 $2r-3$ セルのSN... 2, 3, 4

どの受信側端末で、どのセルが廃棄されたのかが、送信側端末で上記のように特定できた場合、その後の処理として以下のような方法がある。

(1) 受信側端末が受信できなかったセルのSNを全て再送する。

(2) FECの能力に応じて復号に必要な最小限のセルを再送する

ここでもそれぞれ、マルチキャストで送る方法と、ポイント-ポイントで送る方法がある。

40

【0380】

上記の廃棄されたセルのSNを比較すると、2と3が共通であることがわかる。上記2の方法に従えば、それぞれの受信側端末で3セルずつ廃棄が生じているので、それぞれ、あと2セルずつ正しく受信できれば、残りの1セルはFEC符号により回復することができる。よって、前で説明したように最小限のセルを再送する、というポリシーに基づいて、送信側端末 $2t$ は、受信側端末 $2r-1$ 、 $2r-3$ に対して共通の廃棄セルであるSN = 2, 3を送信するという方法がとれる。もちろん、受信側端末 $2r-1$ に対してSN = 1, 2, 3のセル、受信側端末 $2r-3$ に対してSN = 2, 3, 4のセルと、個々に最小限のセルを送ることもできる。しかし、マルチキャストグループ内の特定の端末に対して再送を行なう場合には、先にも述べたが対応するポイント-ポイントのコネクションが必要

50

になる。

【0381】

送信先は、前述した例と同様に、受信側端末 $2r - 1$, $2r - 3$ に限ることもできるし、再度マルチキャストによってこれらの2セルのみ、または $SN = 1, 2, 3, 4$, の4個のセルを送ることもできる。

【0382】

マルチキャストによる場合、すでに正しく復号されている受信側端末ではデータが再送セルであることをSSCSレイヤで検出することができるのでそのセルは廃棄することができる。よって、重複して別のデータに混入してしまって、影響を及ぼす恐れはない。

【0383】

この時の再送セルもオーバーヘッドを最小限にするために、再送セルのみをCPCS-SDUにせず、SSCSレイヤのレベルで別のインタリーブに挿入してしまふことが可能である。

【0384】

マルチキャストグループ内の特定の受信側端末(ここでは受信側端末 $2r - 1$, $2r - 3$)にだけデータを送るようになるには、送信側端末か、もしくは送信側端末と受信側端末の中継点で特定の受信側端末にだけコネクションを設定して必要最小限のセルのみを転送するという方法が考えられる。

【0385】

以上の例では、セル廃棄を検出した受信側が再送要求を出すという、Negative ACK (否定的な応答確認)を用いた方法であったが、Positive ACK (肯定的な応答確認)を用いる方法もとることができる。

【0386】

このような場合には、送信側端末 $2t$ がマルチキャストデータを送信後、SSCS処理モジュール106tでタイマを起動して、マルチキャストグループの全ての受信側端末 $2r$ からACKが返ってくるのを待つ。ACKには、受信側端末 $2r$ のアドレス、受信できたデータの識別子などが記されている。

【0387】

タイマの上限値の設定は、ラウンドトリップタイムに応じて設定される。このタイマの上限値を越えてもACKが返ってこない場合には、その返事のない受信側端末に対してそれに応じたデータを再送する。この再送の方法は、今まで述べてきた方法と同様である。

【0388】

このデータ送出のメカニズムはTCPのウィンドウ制御によく似ている方法であり、同じようにスライディングウィンドウを用いることもできる。ただし、マルチキャストという環境下でPositive ACKを用いているので、全ての受信側端末 $2r$ からACKが返ってこないとそれに対応するデータは再送用のメモリから排除できなくなってしまう。よって、ACKが経路の途中の輻輳を起こしているノード等で送れてしまっている時には、送信者は次のデータを送出できないという状況に陥ってしまい、スループットの低下が起こる可能性もある。

【0389】

受信側のAALより上位のレイヤには、SSCSレイヤでのFECと再送による誤り制御の効果で、エラーフリーな正しいデータが渡されることになる。その後は、レイヤ3, 4の通常の処理が行なわれ、アプリケーションレイヤにはもちろん正しいデータが届けられる。

【0390】

受信側の説明に含めて、再送時の送信側の説明を述べた。このように行なうことで、マルチキャストを行なっている時に、FECの能力を越えてしまうほどの誤りが転送中に生じてても、最小限のオーバーヘッドと遅延でサービスを提供できる。

【0391】

上記したように送受信側端末の処理を行なうことで、AALレベルにエラーフリーな通信

10

20

30

40

50

を提供することが可能である。マルチキャストを行なう場合は、受信側端末の数が大きければ大きいほど、いずれかの受信側端末から送信側端末に対して再送要求が生じる可能性が高い。しかし、AALレベルにFEC機能を持たせ、さらに、再送による誤り訂正を提供することによって、信頼性が高く、リアルタイム性のより保証されたシステムを構築することができる。

【0392】

以上、各実施例にて説明したように、本発明によれば、ATM通信網で高速通信を行う場合に、AALのいずれかのレイヤにインタリーバとFECを用いた誤り制御方法を実装し、さらに、インタリーバのサイズを柔軟にアプリケーションデータのサイズに合わせることでデータ転送の高効率化、高速化、省資源化を図ることができる。特に、AALタイプ5で高速データ転送を行う場合には、CPCSよりも上位のレイヤに本発明の誤り制御方法を適用することによって、本来FECが持つオーバーヘッドを最小限に抑えつつ、AALタイプ5の高速性を最大限に発揮することが可能となる。また、AAL1、2、3/4に対しても、より高効率にFECを実装することが可能である。

【0393】

また、本発明は、上述した各実施例に限定されるものではなく、その要旨を逸脱しない範囲で、種々変形して実施することができる。

【0394】

【発明の効果】

本発明によれば、信頼性の高いデータ通信を効果的に実現することができる。

【図面の簡単な説明】

【図1】本発明の第1の実施例の送受信端末のプロトコルスタックの構造を示す図

【図2】送信側端末2tおよび受信側端末2rの内部構成を示す図

【図3】AAL-IDUとして上位から渡されるアプリケーションデータを示す図

【図4】インタリーバへのアプリケーションデータの書き込みの様子を示す図

【図5】インタリーバへの書き込み順序の例を示す図

【図6】4オクテット単位でデータ長のアライメントをとる場合の様子を示す図

【図7】インタリーバにデータを格納した様子を示す図

【図8】他のインタリーバにデータを格納した様子を示す図

【図9】図8の場合の処理の流れを説明するための図

【図10】図7および図8の場合のLIFフィールドの例を示す図

【図11】インタリーバからのデータの読み出しの例を示す図

【図12】FEC冗長コードの付与の例を示す図

【図13】データ領域内の最終列の長さ表示の例を示す図

【図14】図13のFEC演算の立場から見たマトリクスの状態を示す図

【図15】マトリクスの1ラインの符号語構成の例を示す図

【図16】データ領域にダミーを入れる場合のインタリーバの状態を示す図

【図17】SSCSヘッダ/トレイラを付与する単位を示す図

【図18】SSCSヘッダ/トレイラの一例の図

【図19】インタリーバ1つに対するSSCSヘッダ/トレイラを示す図

【図20】 $UUI = 1$ を含むセルが廃棄された場合の受信側でのレイヤ処理を説明するための図

【図21】図8の場合の処理の流れを説明するための図

【図22】モジュロ値のテーブルを示す図

【図23】冗長シンボルのテーブルを示す図

【図24】本発明の第2の実施例における1対多通信の形態を示す図

【図25】同実施例の送受信端末のプロトコルスタックの構造を示す図

【図26】ATMコネクションの張り方を説明するための図

【図27】送信側端末2tおよび受信側端末2rの内部構成を示す図

【図28】受信側端末2r-1において再送要求が出された場合について説明するための図

10

20

30

40

50

【図29】SSCSヘッダ/トレイラの例を示す図

【図30】複数の端末から再送要求が送られた場合について説明するための図

【図31】ATMアダプテーションレイヤを説明するための図

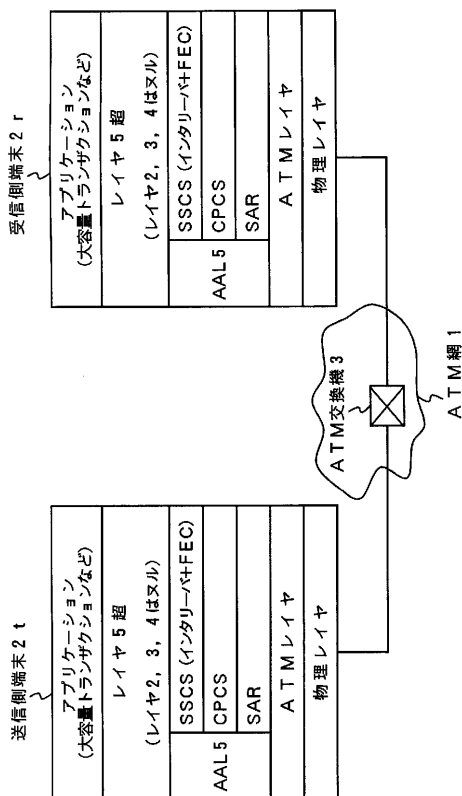
【符号の説明】

1 ... ATM網、3 ... ATM交換機、2 t ... 送信側端末、2 r ... 受信側端末、4 t ... アプリケーション処理モジュール、4 r ... アプリケーション処理モジュール、6 t ... SSCS処理モジュール、6 r ... SSCS処理モジュール、8 t ... インタリバモジュール、8 r ... デインタリバモジュール、10 t ... FEC付与モジュール、10 r ... FEC訂正モジュール、12 t ... SSCSヘッダ/トレイラ付与モジュール、12 r ... SSCSヘッダ/トレイラ確認モジュール、14 t ... CPCCS処理モジュール、14 r ... CPCCS処理モジュール、16 t ... LI付与モジュール、16 r ... LI確認モジュール、18 t ... CRC付与モジュール、18 r ... CRC確認モジュール、20 t ... SAR処理モジュール(セグメンテーション)、20 r ... SAR処理モジュール(リアセンブリ)、22 t ... ATMヘッダ付与モジュール、22 r ... ATMヘッダ削除モジュール、1-1 ~ 1-7 ... ATM網、2 r-1 ~ 2 r-5 ... 受信側端末、30 ... マルチキャストサーバ、104 t ... アプリケーション処理モジュール、104 r ... アプリケーション処理モジュール、130 t ... MTP/RMP処理モジュール、130 r ... MTP/RMP処理モジュール、131 t ... IP処理モジュール、131 r ... IP処理モジュール、106 t ... SSCS処理モジュール、106 r ... SSCS処理モジュール、114 t ... CPCCS処理モジュール、114 r ... CPCCS処理モジュール、120 t ... SAR処理モジュール、120 r ... SAR処理モジュール、122 t ... ATMヘッダ処理モジュール、122 r ... ATMヘッダ処理モジュール、H ... 送信側端末、H1 ~ H25 ... 受信側端末、S1 ~ S5 ... 中継サーバ

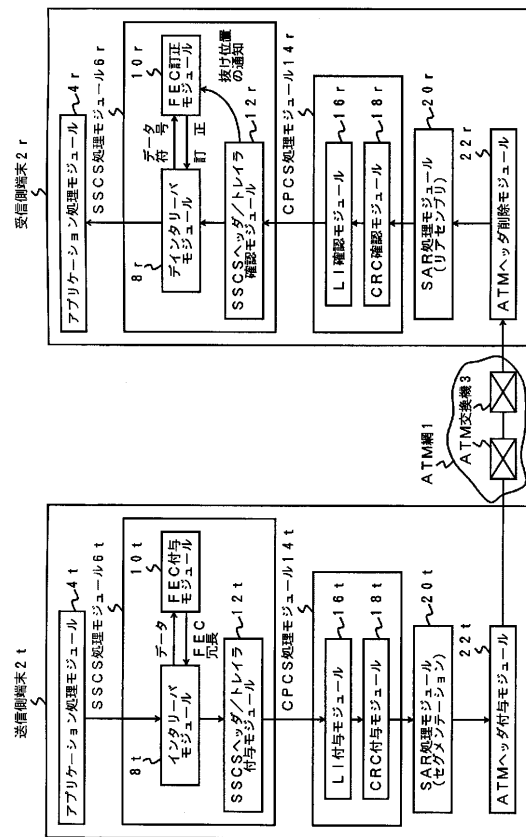
10

20

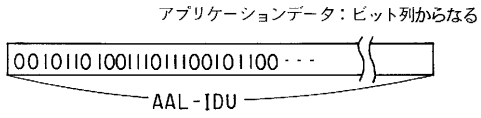
【図1】



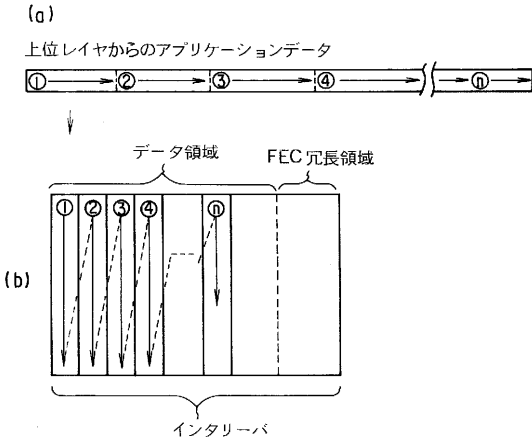
【図2】



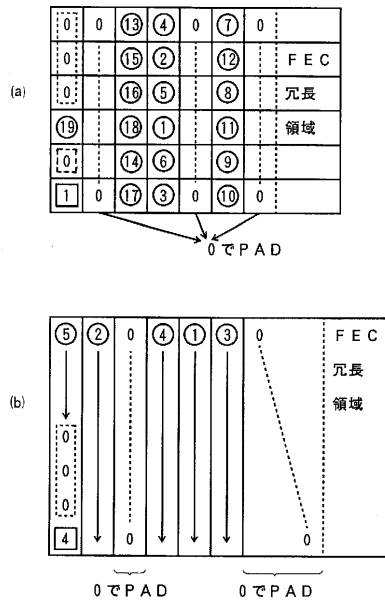
【 図 3 】



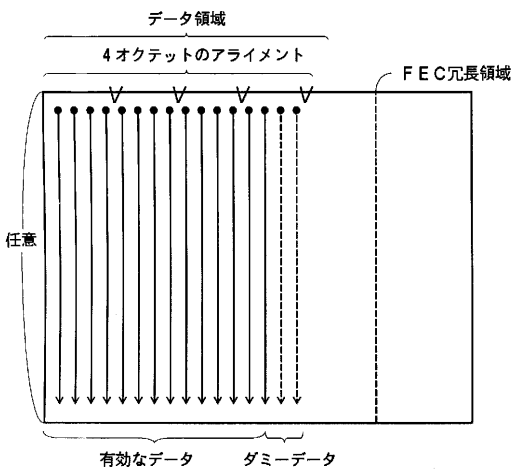
【 図 4 】



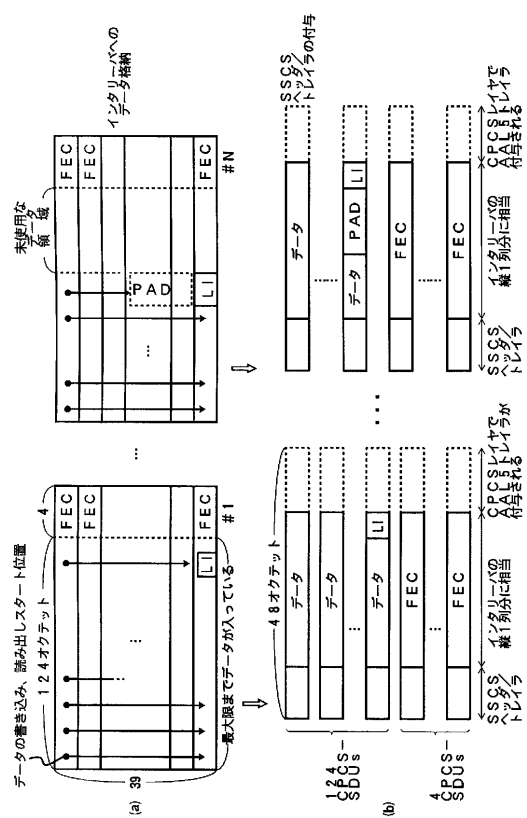
【 図 5 】



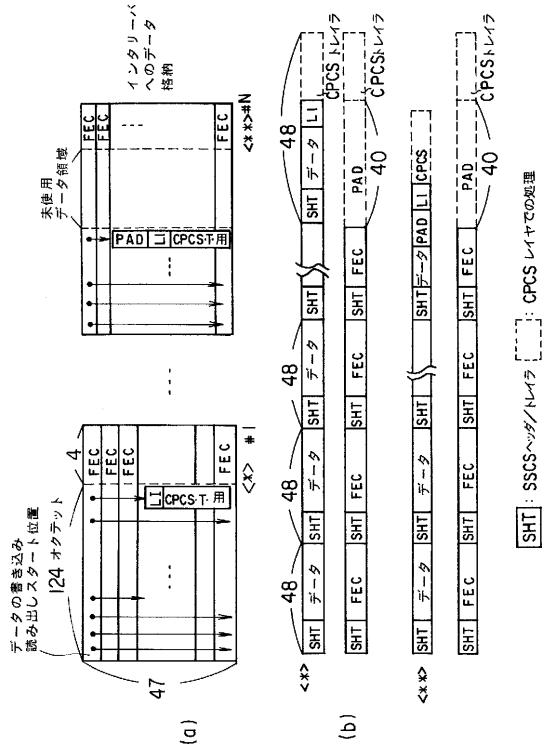
【 図 6 】



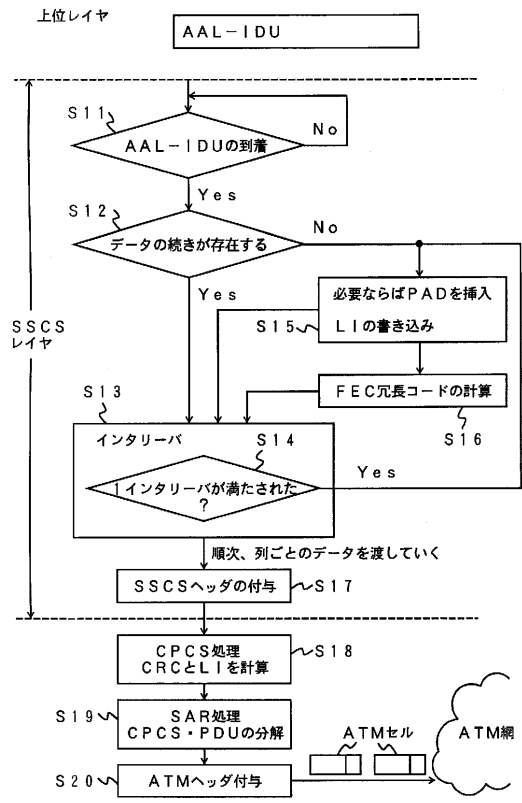
【 図 7 】



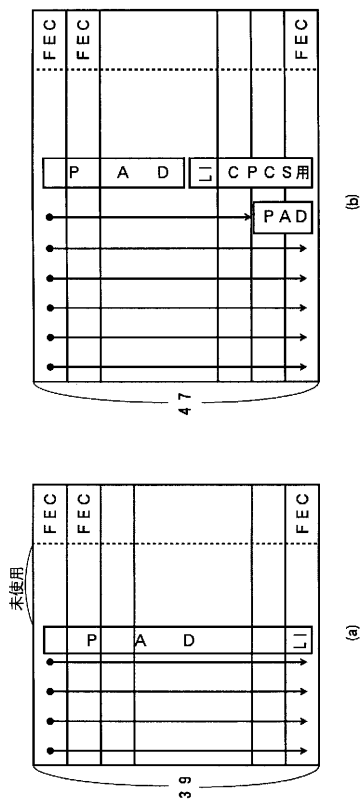
【図 8】



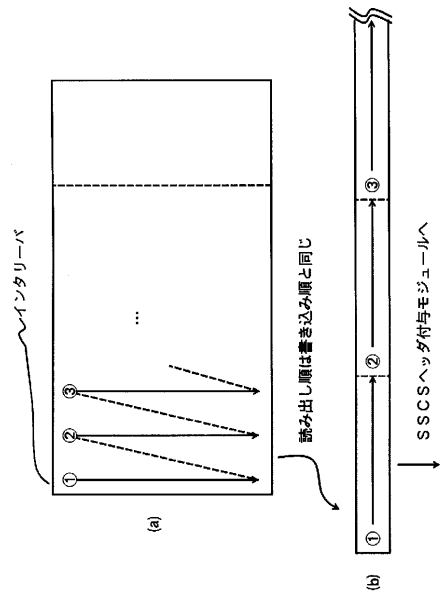
【図 9】



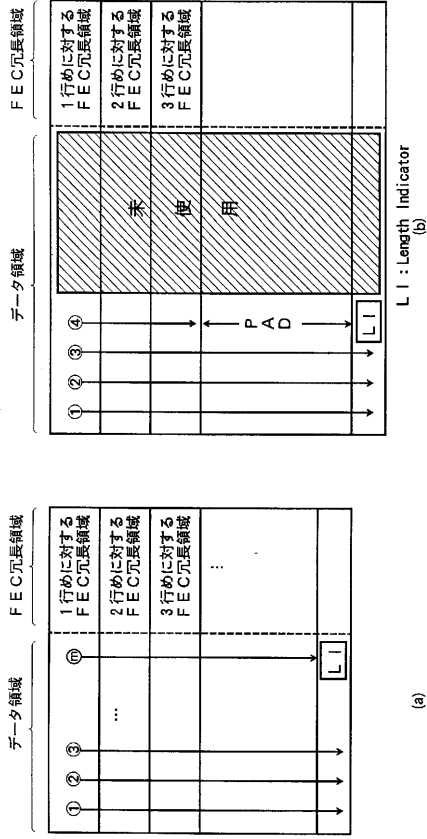
【図 10】



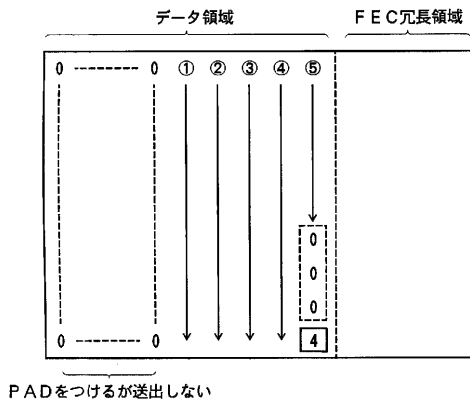
【図 11】



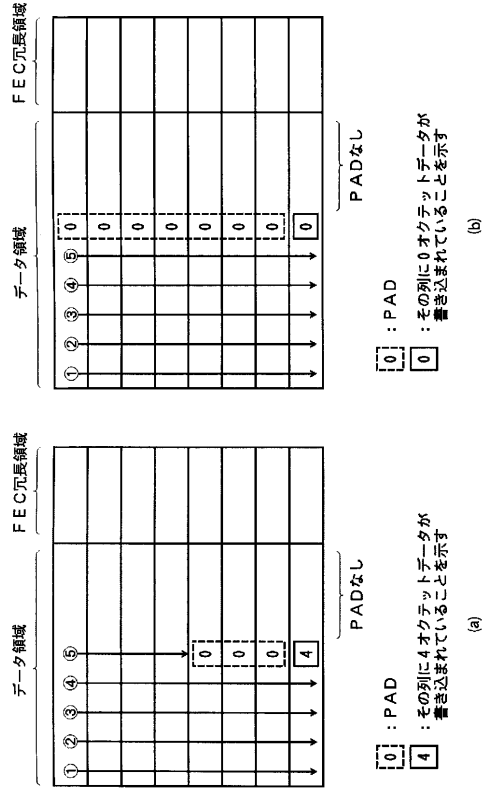
【 図 1 2 】



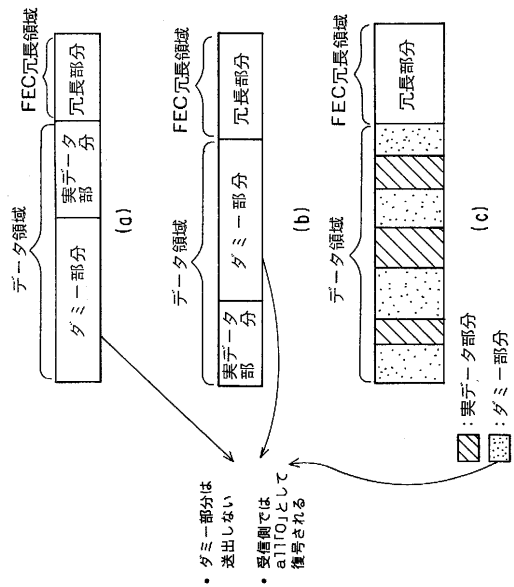
【 図 1 4 】



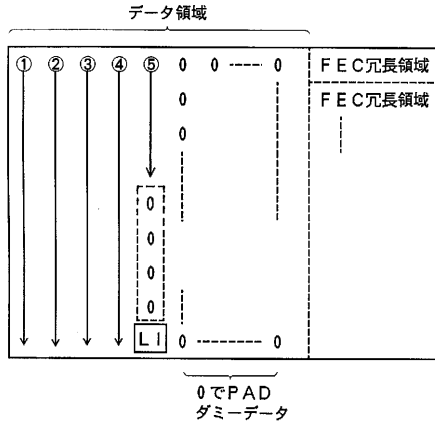
【 図 1 3 】



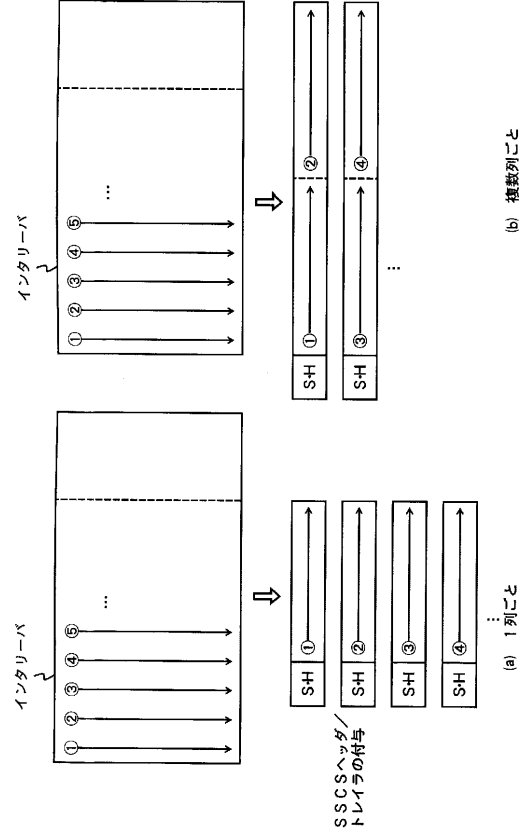
【 図 1 5 】



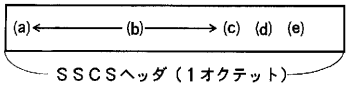
【 図 16 】



【 図 17 】

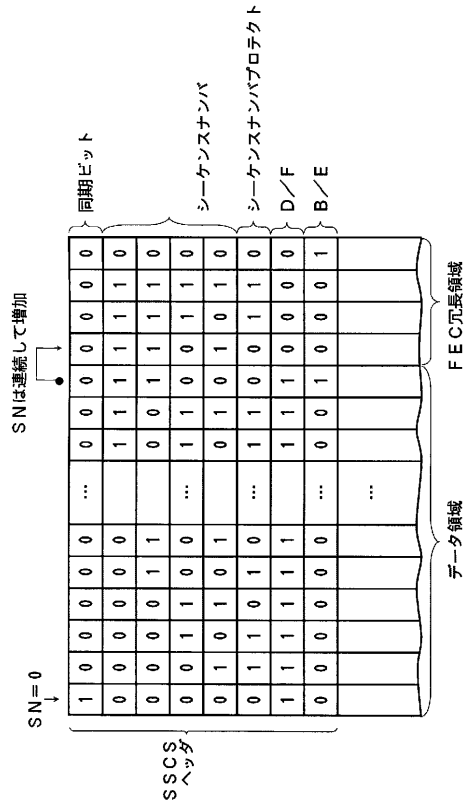


【 図 18 】

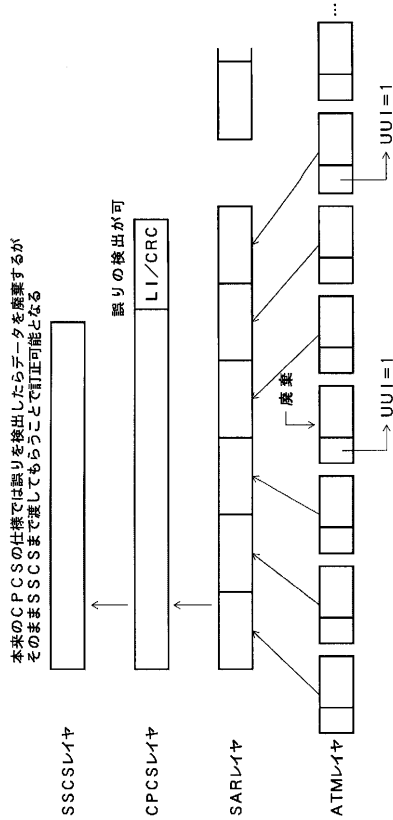


- (a): 同期ビット (インタリバーの先頭を示す)
- (b): シーケンスナンバ (インタリバー中のどの列かを示す)
- (c): SNP (シーケンスナンバのプロテクト用)
- (d): D/F (データであるか、FECコードであるかを示す)
- (e): B/E (データとFECコードの最後であることを示す)

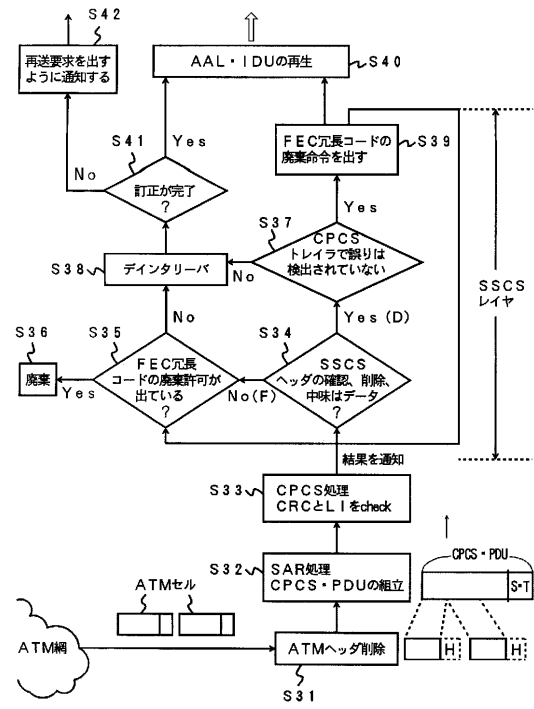
【 図 19 】



【 図 2 0 】



【 図 2 1 】

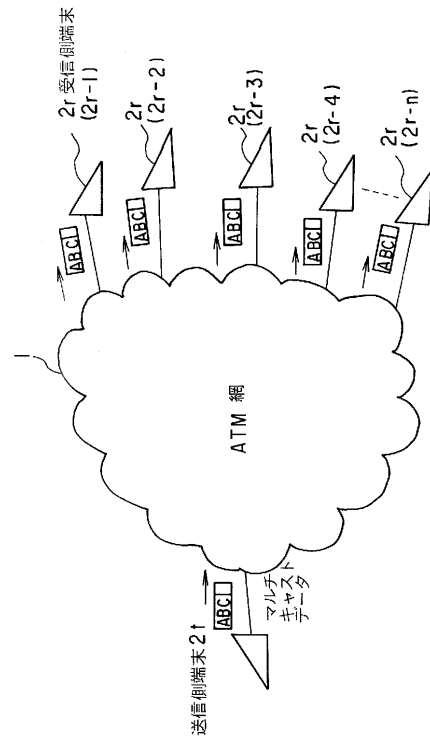


【 図 2 2 】

表 a

Xのべき乗	X ³ の係数	X ² の係数	Xの係数	定数
127	E [1]	F [1]	G [1]	H [1]
126	E [2]	F [2]	G [2]	H [2]
⋮	⋮	⋮	⋮	⋮
4	E [124]	F [124]	G [124]	H [124]

【 図 2 4 】

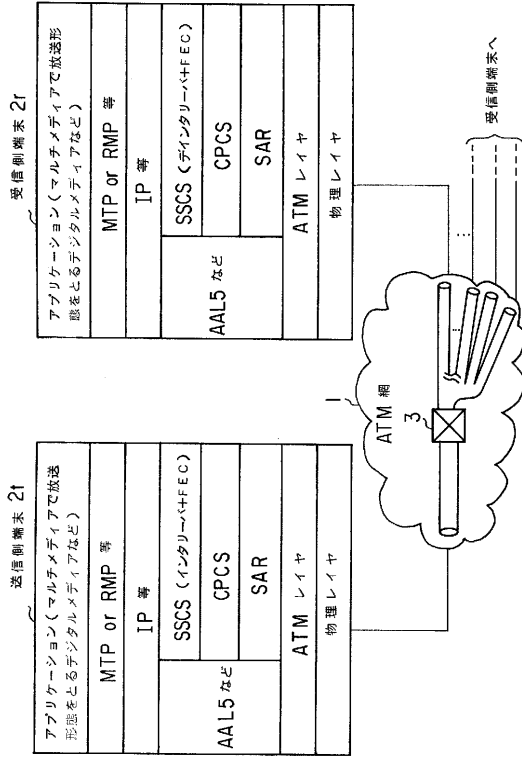


【 図 2 3 】

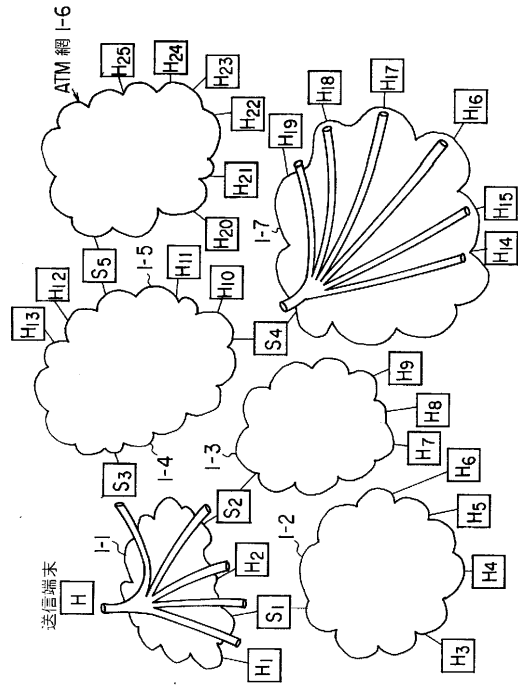
表 b

シンボル番号	行番号	列番号
1	121	4
2	121	2
⋮	⋮	⋮
19	118	2

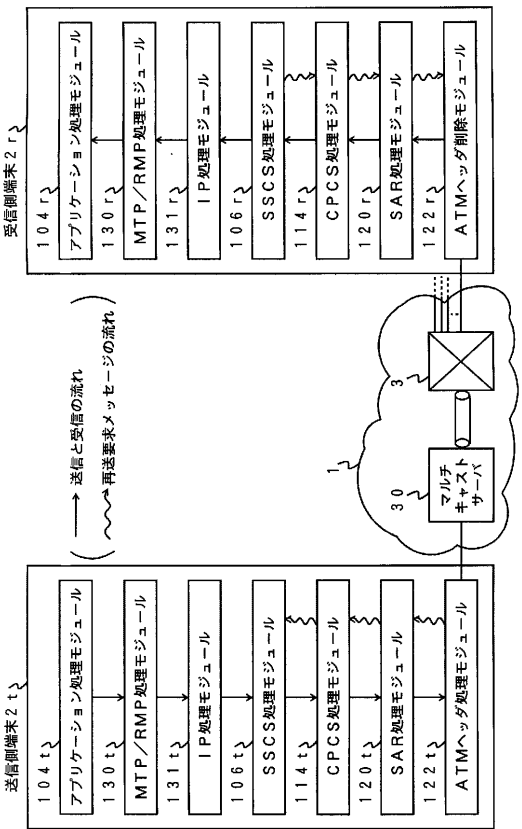
【 図 2 5 】



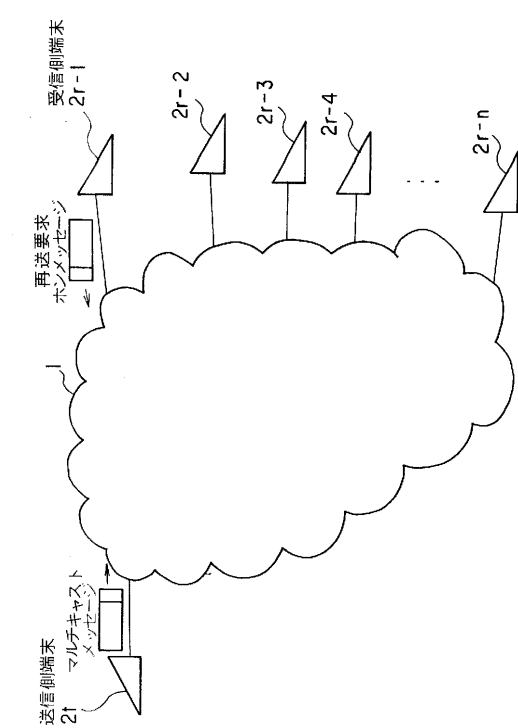
【 図 2 6 】



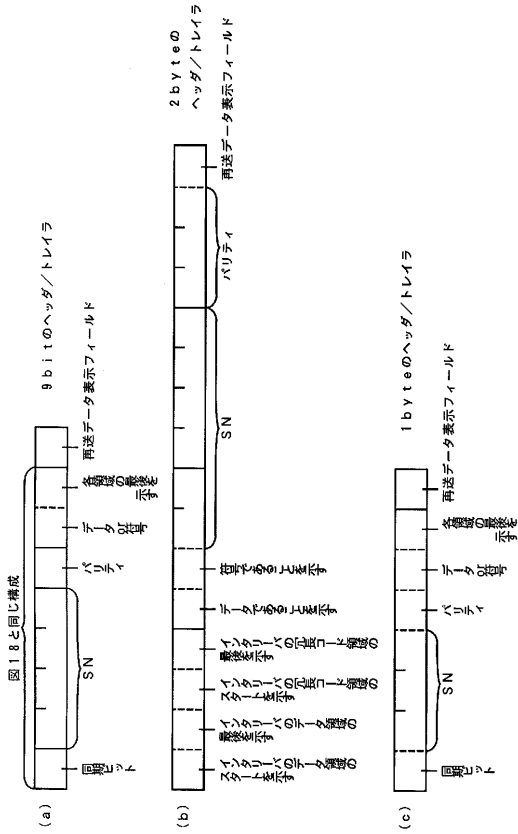
【 図 2 7 】



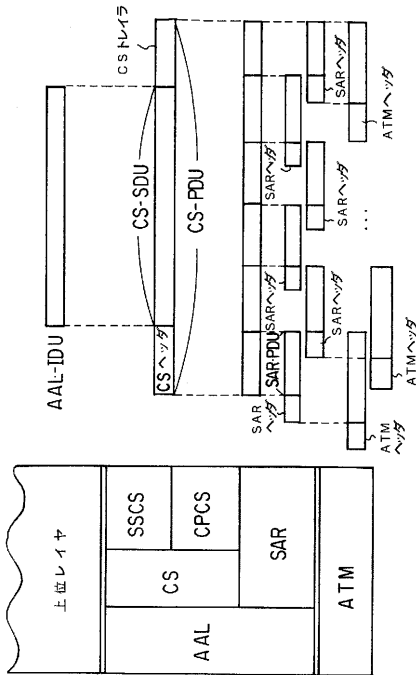
【 図 2 8 】



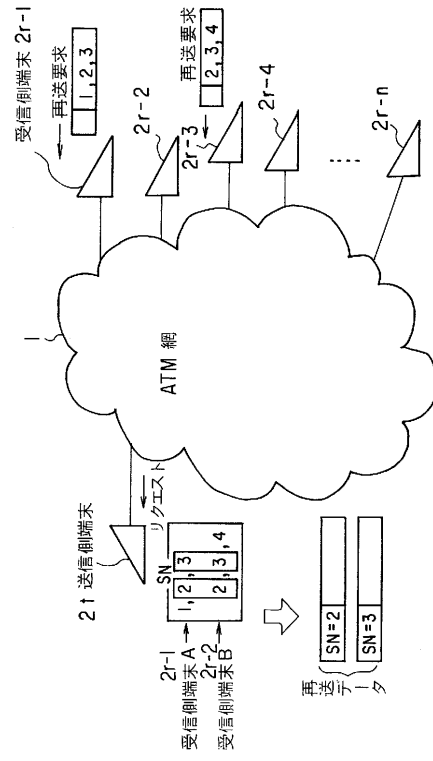
【 図 2 9 】



【 図 3 1 】



【 図 3 0 】



フロントページの続き

(56)参考文献 特開平5 - 91091 (JP, A)

金井他, ATM網における誤り制御方式、FEC on AAL Type5, 1994年電子情報通信学会春季大会講演論文集, 1994年 3月26日, 通信3 B - 786, p. 3-281

佐藤陽一, 上田裕巳, HIPPI-ATMアダプタの構成法の検討, 信学技報, 1994年 9月, SSE94-100, pp.57-62

(58)調査した分野(Int.Cl.⁷, DB名)

H04L 12/56

H04L 1/16

H04L 1/18