

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3943722号

(P3943722)

(45) 発行日 平成19年7月11日(2007.7.11)

(24) 登録日 平成19年4月13日(2007.4.13)

(51) Int. Cl.

F I

H O 4 L 12/28 (2006.01)

H O 4 L 12/28 2 O O Z

H O 4 L 29/08 (2006.01)

H O 4 L 13/00 3 O 7 Z

H O 4 L 13/08 (2006.01)

H O 4 L 13/08

G O 6 F 13/00 (2006.01)

G O 6 F 13/00 3 5 3 Q

G O 6 F 13/38 (2006.01)

G O 6 F 13/38 3 5 O

請求項の数 18 (全 54 頁) 最終頁に続く

(21) 出願番号 特願平10-220689
 (22) 出願日 平成10年8月4日(1998.8.4)
 (65) 公開番号 特開2000-59402(P2000-59402A)
 (43) 公開日 平成12年2月25日(2000.2.25)
 審査請求日 平成16年11月30日(2004.11.30)

(73) 特許権者 000001007
 キヤノン株式会社
 東京都大田区下丸子3丁目30番2号
 (74) 代理人 100076428
 弁理士 大塚 康德
 (74) 代理人 100112508
 弁理士 高柳 司郎
 (74) 代理人 100115071
 弁理士 大塚 康弘
 (74) 代理人 100116894
 弁理士 木村 秀二
 (74) 代理人 100093908
 弁理士 松本 研一
 (74) 代理人 100101306
 弁理士 丸山 幸雄

最終頁に続く

(54) 【発明の名称】 データ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体

(57) 【特許請求の範囲】

【請求項1】

ホストデバイスとターゲットデバイス間で利用されるデータ転送方法であって、
 前記ターゲットデバイスのバッファに格納されるべき第1のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第1の転送ステップと、

前記第1のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから前記ホストデバイスへ返信する第1の返信ステップと、

前記バッファ情報が示す空き容量が所定の容量より大きくない場合、特定のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第2の転送ステップと、

前記特定のデータを前記バッファに格納することなく、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから前記ホストデバイスへ返信する第2の返信ステップと、

前記特定のデータが前記ホストデバイスから前記ターゲットデバイスへ転送された後、前記バッファ情報が示す空き容量が所定の容量より大きい場合、前記バッファに格納されるべき第2のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第3の転送ステップとを有することを特徴とするデータ転送方法。

【請求項2】

前記バッファは、複数のバッファよりなり、

前記バッファ情報は、前記複数のバッファのうちデータを格納可能な空きバッファの数を示すことを特徴とする請求項1記載のデータ転送方法。

10

20

【請求項 3】

前記特定のデータは、前記空きバッファが所定数以下である場合に転送されることを特徴とする請求項 2 記載のデータ転送方法。

【請求項 4】

前記特定のデータは、前記空きバッファ数が 1 個である場合に転送されることを特徴とする請求項 2 或いは 3 記載のデータ転送方法。

【請求項 5】

前記第 1 のデータ及び前記第 2 のデータは、画像データを転送するためのコマンドによって転送される画像データを含むことを特徴とする請求項 1 乃至 4 のいずれかに記載のデータ転送方法。

10

【請求項 6】

前記特定のデータは、前記バッファに格納されるべきデータを含まないダミーデータを転送するためのコマンドであることを特徴とする請求項 1 乃至 5 のいずれかに記載のデータ転送方法。

【請求項 7】

前記特定のデータは、ダミー転送制御を行なうための転送制御コマンドであることを特徴とする請求項 1 乃至 5 のいずれかに記載のデータ転送方法。

【請求項 8】

前記バッファ情報が示す空き容量が所定の容量より大きくない場合、前記ターゲットデバイスのステータス情報を取得するための第 3 のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第 4 の転送ステップと、

20

前記第 3 のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから前記ホストデバイスへ返信する第 3 の返信ステップとを有することを特徴とする請求項 1 乃至 7 のいずれかに記載のデータ転送方法。

【請求項 9】

前記第 3 のデータは、前記ターゲットデバイスのステータス情報を取得するためのコマンドであることを特徴とする請求項 8 記載のデータ転送方法。

【請求項 10】

前記特定のデータの 1 回の転送に応じて応答が返信されるまでの時間は、前記第 1 のデータの転送に応じて応答が返信される場合よりも長いことを特徴とする請求項 1 乃至 9 のいずれかに記載のデータ転送方法。

30

【請求項 11】

前記第 1 のデータ及び前記第 2 のデータの転送の際に、コマンドとデータとをレジスタに書き込むことを特徴とする請求項 5 記載のデータ転送方法。

【請求項 12】

前記第 1 のデータ及び前記第 2 のデータに対する応答は、前記レジスタへの書き込みに対して返信されることを特徴とする請求項 11 記載のデータ転送方法。

【請求項 13】

前記シリアルバスは I E E E 1 3 9 4 規格に適合または準拠するバス、または U S B 規格に適合または準拠するバスであることを特徴とする請求項 1 乃至 12 の何れかに記載のデータ転送方法。

40

【請求項 14】

データ受信装置であって、
データを格納するバッファと、
前記バッファに格納されるべき第 1 のデータをホストデバイスから受信する第 1 の受信手段と、

前記第 1 のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ホストデバイスへ送信する第 1 の送信手段と、

前記バッファ情報が示す空き容量が所定の容量より大きくない場合、特定のデータを前記ホストデバイスから受信する第 2 の受信手段と、

50

前記特定のデータを前記バッファに格納することなく、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ホストデバイスへ送信する第2の送信手段と、

前記特定のデータを前記ホストデバイスから受信した後、前記バッファ情報が示す空き容量が所定の容量より大きい場合、前記バッファに格納されるべき第2のデータを前記ホストデバイスから受信する第3の受信手段とを有し、

前記ホストデバイスは、前記バッファ情報が示す空き容量が所定の容量より大きくない場合、前記特定のデータを前記データ受信装置へ転送することを特徴とするデータ受信装置。

【請求項15】

データ転送装置であって、

ターゲットデバイスのバッファに格納されるべき第1のデータを前記ターゲットデバイスへ転送する第1の転送手段と、

前記第1のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから受信する第1の受信手段と、

前記バッファ情報が示す空き容量が所定の容量より大きくない場合、特定のデータを前記ターゲットデバイスへ転送する第2の転送手段と、

前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから受信する第2の受信手段と、

前記特定のデータが前記ターゲットデバイスへ転送された後、前記バッファ情報が示す空き容量が所定の容量より大きい場合、前記バッファに格納されるべき第2のデータを前記ターゲットデバイスへ転送する第3の転送手段とを有し、

前記ターゲットデバイスは、前記特定のデータを前記バッファに格納することなく、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記データ転送装置に送信することを特徴とするデータ転送装置。

【請求項16】

転送されてきた画像データをバッファに格納して応答を送信する外部装置と通信するデータ転送装置であって、

前記バッファの空き容量に関するバッファ情報を含む応答を前記外部装置から受信する受信手段と、

前記バッファ情報によって示される前記バッファの空き容量が所定より大きい場合、画像データを前記外部装置に転送する画像データ転送手段と、

前記バッファ情報によって示される前記バッファの空き容量が所定より大きくない場合、前記画像データ転送手段による画像データの転送を中断して、特定のコマンドを前記外部装置に転送するコマンド転送手段とを有し、

前記外部装置は、前記特定のコマンドを前記バッファに格納することなく、前記バッファ情報を含む応答を送信し、

前記コマンド転送手段が前記特定のコマンドを転送した後、前記バッファ情報によって示される前記バッファの空き容量が所定より大きい場合、前記画像データ転送手段は画像データを前記外部装置に転送することを特徴とするデータ転送装置。

【請求項17】

第1の装置と、前記第1の装置から転送されてきた画像データをバッファに格納して応答を送信する第2の装置との間のデータ転送方法であって、

前記第1の装置から前記第2の装置へ画像データを転送するステップと、

前記バッファの空き容量に関するバッファ情報を含む応答を前記第2の装置から前記第1の装置へ転送するステップと、

前記バッファ情報によって示される前記バッファの空き容量が所定より大きくない場合、特定のコマンドを前記第1の装置から前記第2の装置へ転送するステップとを有し、

前記第2の装置は、前記特定のコマンドを前記バッファに格納することなく、前記バッファ情報を含む応答を転送し、

10

20

30

40

50

前記特定のコマンドが転送された後、前記バッファ情報によって示される前記バッファの空き容量が所定より大きい場合、画像データを前記第1の装置から前記第2の装置へ転送することを特徴とするデータ転送方法。

【請求項18】

画像処理装置であって、

外部装置から転送されてきた画像データを格納する格納手段と、

前記外部装置からの画像データに対して、前記格納手段の空き容量に関する格納手段情報を含む応答を前記外部装置に送信する送信手段と、

前記外部装置から特定のコマンドを受信する受信手段と、

前記特定のコマンドを前記格納手段に格納することなく、前記特定のコマンドに対して、前記格納手段の空き容量に関する格納手段情報を含む応答を前記外部装置に送信する第2の送信手段とを有し、

10

前記外部装置は、前記格納手段情報によって示される前記格納手段の空き容量が所定より大きい場合、画像データを前記画像処理装置に転送し、前記格納手段情報によって示される前記格納手段の空き容量が所定より大きくない場合、前記特定のコマンドを前記画像処理装置に転送することを特徴とする画像処理装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明はデータ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体に関し、例えば、IEEE1394などにより規定されるシリアルインタフェイスを介して、画像供給デバイスと画像処理デバイスとを直結する場合のデータ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体に関するものである。

20

【従来の技術】

プリンタは、セントロニクスやRS232Cと言ったパラレルあるいはシリアルインタフェイスを介して、ホストデバイスであるパーソナルコンピュータ(PC)と接続されている。

【0002】

また、スキャナ、デジタルスチルカメラ、デジタルビデオカメラといった画像供給デバイスであるデジタル機器もPCに接続されている。各デジタル機器により取込まれた画像データは、一旦PC上のハードディスクなどに取込まれた後、PC上のアプリケーションソフトウェアなどにより処理されてプリンタ用の印刷データに変換され、上記のインタフェイスを経由してプリンタに送られる。

30

【0003】

上記のようなシステムでは、各デジタル機器やプリンタなどを制御するためのドライバソフトウェアがそれぞれ独立にPCに存在し、デジタル機器から出力された画像データは、それらドライバソフトウェアによりPC上で使い易くかつ表示し易い形式のデータとして保存される。保存されたデータは、入力機器の画像特性と出力機器の画像特性とを考慮した画像処理方法により印刷データに変換される。

【0004】

今日、IEEE1394により規定されるインタフェイス(以下「1394シリアルバス」と呼ぶ)のような新しいインタフェイスでは、画像供給デバイスとプリンタとを直結することも可能である。1394シリアルバスにより画像供給デバイスとプリンタとを直結する場合、FCP(Function Control Protocol)のオペランドに印刷データを含める方法が考えられる。また、1394シリアルバスでは、データ転送のためのレジスタ領域を設けて、そのレジスタ領域にデータを書込むことでデータ転送を行う方法も考えられる。

40

【0005】

また、データ転送の開始を指示するコマンドと、コマンドに対するレスポンスとを用いて、データ転送を指示する方法も考えられる。

【発明が解決しようとする課題】

しかし、上述した技術においては、次のような問題点がある。前述したように、画像供給

50

デバイスから出力される画像データは、PCにより印刷データに変換されてプリンタにより印刷されるものであるから、画像供給デバイスとプリンタを直結したとしても、PCが無ければ印刷を行うことができない。デジタルビデオカメラから出力される画像データを直接印刷するビデオプリンタと呼ばれるプリンタもあるが、特定の機種間で接続ができるだけであり、多数の画像供給デバイスと直結して使える汎用性の高いビデオプリンタはない。つまり、1394シリアルバスなどの特徴であるデバイス間を直結する機能を生かし、画像供給デバイスからプリンタへ画像データを直接送って印刷することはできない。

【0006】

1394シリアルバスにより画像供給デバイスとプリンタを直結し、FCPのオペランドに印刷データを含める前述した方法は、制御コマンドと印刷データとを分離することができない問題がある上、コマンドに対して常にレスポンスが必要なため転送効率が低いという問題もある。また、前述したデータ転送のためのレジスタ領域を設ける方法は、そのレジスタ領域にデータを書き込むことが可能であるかどうかを判定するための処理がデータ転送の度に必要になる。従って、この判定処理のオーバーヘッドが大きくなり、やはり転送効率が低下するという問題が発生する。

10

【0007】

この問題を解決するために、データとコマンドとを分離せずに同じレジスタ領域を用いる方法が考えられる。この方法では使われるレジスタ領域を減らすことができ、より簡単なデータ転送方式を提供できる。さらにデータを書き込むレジスタ領域が書き込み可能であるかどうかの判定は行わず、データをレジスタに書き込んだ後に、該書き込みに対する正否の応答のみを行い、成功の応答があれば次のデータを転送する方法も考えられる。

20

【0008】

この方法により、データ転送手順を簡略化するという利点があるものの、反面データ受け取り側でデータを書き込んだ領域を保存するバッファが一杯になるという問題が発生する。データ受け取り側は、レジスタにデータが書き込まれて自分のバッファに保存できると、すぐにデータの受け取り可の応答を返すことになる。受け取り可の応答を受け取ったデータ転送側は、次のデータの転送をすぐに開始しようとする。従って、データ転送側に対してデータ受け取り側の処理が遅い場合、データ受け取り側のバッファはいつも一杯の状態となる。

【0009】

30

データ受け取り側は、最後の空きバッファのデータを保存した後、処理が進み空きバッファが出来るまで、データ送り側に受け取り可の応答を返す事ができなくなる。この場合、コマンドとデータにより同じレジスタ領域が使用されているため、データを転送するコマンドの実行中には、データ受け取り側のバッファに空きが無いと、データ転送以外のコマンドが実行できなくなる。

【0010】

これは、データ受け取り側としてプリンタを考えた場合、データ転送中にプリンタのステータス（紙なし、エラーなど）を得るコマンドが実行できる場合と実行できない場合とが発生することを意味する。すなわち、プリンタに空きバッファがあり、ステータスを得るコマンドが受け取れるときには該コマンドが実行できるが、空きバッファが無いときには該コマンドは実行できないか、空きバッファができるまで該コマンドは待たされることになる。従って、特にリアルタイム性を必要とするようなステータス取得を行う場合には問題が発生する。また、必要とされる時点でのステータスであるか否かの判断がつかなくなるという問題も発生する。

40

【0011】

また、データを転送した後に、該転送に対する応答がほぼ一定の短い時間で行われるため、データバス上を流れるデータ量が増し、バスの有効利用という点に対して問題があった。

【0012】

また、前述したデータ転送の開始を指示するコマンドと、コマンドに対するレスポンスと

50

を用いてデータ転送の指示する場合は、ある単位 of データ転送ごとにコマンドおよびレスポンスのやり取りが発生し、やはり転送効率を低下させるという問題が発生する。

【0013】

本発明は、上述した問題を個々にまたはまとめて解決するためのものであり、1394シリアルバスなどによりホストデバイスとターゲットデバイスを接続し、ホストデバイスからターゲットデバイスへ送られるデータの転送について、コマンドとデータとで同じレジスタ領域を使用し、かつレジスタへのデータの書き込みに対する応答のみを行う際に、データ転送以外のコマンドを任意に実行可能なデータ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体を提供することを目的とする。

【0014】

また、データ転送に対する応答時間を調整することにより、データバス上のトラフィックの効率化を実現するデータ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体を提供することを目的とする。

【0015】

【課題を解決するための手段】

本発明は、前記の目的を達成する一手段として、以下の構成を備える。

【0016】

本発明にかかるデータ転送方法は、ホストデバイスとターゲットデバイスの間で利用されるデータ転送方法であって、前記ターゲットデバイスのバッファに格納されるべき第1のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第1の転送ステップと、前記第1のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから前記ホストデバイスへ返信する第1の返信ステップと、前記バッファ情報が示す空き容量が所定の容量より大きくない場合、特定のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第2の転送ステップと、前記特定のデータを前記バッファに格納することなく、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから前記ホストデバイスへ返信する第2の返信ステップと、前記特定のデータが前記ホストデバイスから前記ターゲットデバイスへ転送された後、前記バッファ情報が示す空き容量が所定の容量より大きい場合、前記バッファに格納されるべき第2のデータを前記ホストデバイスから前記ターゲットデバイスへ転送する第3の転送ステップとを有することを特徴とする。

【0017】

また、本発明にかかるデータ受信装置は、前記バッファに格納されるべき第1のデータをホストデバイスから受信する第1の受信手段と、前記第1のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ホストデバイスへ送信する第1の送信手段と、前記バッファ情報が示す空き容量が所定の容量より大きくない場合、特定のデータを前記ホストデバイスから受信する第2の受信手段と、前記特定のデータを前記バッファに格納することなく、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ホストデバイスへ送信する第2の送信手段と、前記特定のデータを前記ホストデバイスから受信した後、前記バッファ情報が示す空き容量が所定の容量より大きい場合、前記バッファに格納されるべき第2のデータを前記ホストデバイスから受信する第3の受信手段とを有し、前記ホストデバイスは、前記バッファ情報が示す空き容量が所定の容量より大きくない場合、前記特定のデータを前記データ受信装置へ転送することを特徴とする。

【0018】

また、本発明にかかるデータ転送装置は、ターゲットデバイスのバッファに格納されるべき第1のデータを前記ターゲットデバイスへ転送する第1の転送手段と、前記第1のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記ターゲットデバイスから受信する第1の受信手段と、前記バッファ情報が示す空き容量が所定の容量より大きくない場合、特定のデータを前記ターゲットデバイスへ転送する第2の転送手段と、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前

10

20

30

40

50

記ターゲットデバイスから受信する第2の受信手段と、前記特定のデータが前記ターゲットデバイスへ転送された後、前記バッファ情報が示す空き容量が所定の容量より大きい場合、前記バッファに格納されるべき第2のデータを前記ターゲットデバイスへ転送する第3の転送手段とを有し、前記ターゲットデバイスは、前記特定のデータを前記バッファに格納することなく、前記特定のデータに対する応答として前記バッファの空き容量を示すバッファ情報を前記データ転送装置に送信することを特徴とする。

【0019】

【IEEE1394の概要】

家庭用デジタルVTRやデジタルビデオディスク(DVD)の登場に伴い、ビデオデータやオーディオデータ(以下、まとめて「AVデータ」と呼ぶ)など、リアルタイムかつ情報量の多いデータを転送する必要があるが生じている。AVデータをリアルタイムに、PCへ転送したり、その他のデジタル機器に転送するには、高速のデータ転送能力をもつインタフェースが必要になる。そういった観点から開発されたインタフェースが1394シリアルバスである。

【0020】

図2に1394シリアルバスを用いて構成されるネットワークシステムの例を示す。このシステムは機器AからHを備え、A-B間、A-C間、B-D間、D-E間、C-F間、C-G間、およびC-H間がそれぞれ1394シリアルバス用のツイストペアケーブルで接続されている。これらの機器AからHの例としては、パソコンなどのホストコンピュータ装置、および、コンピュータ周辺機器である。コンピュータ周辺機器としては、デジタルVCR、DVDプレーヤ、デジタルスチルカメラ、ハードディスクや光ディスクなどのメディアを用いる記憶装置、CRTやLCDのモニタ、チューナ、イメージスキャナ、フィルムスキャナ、プリンタ、MODEM、ターミナルアダプタ(TA)などコンピュータ周辺機器のすべてが対象になる。なお、プリンタの記録方式は、レーザビームやLEDを用いた電子写真方式、インクジェット方式、インク溶解型や昇華型の熱転写方式、感熱記録方式など、どんな方式でも構わない。

【0021】

各機器間の接続は、ディジーチェーン方式とノード分岐方式との混在が可能であり、自由度の高い接続を行うことができる。また、各機器はそれぞれIDを有し、互いにIDを認識し合うことによって、1394シリアルバスで接続された範囲において、一つのネットワークを構成している。例えば、各機器間をそれぞれ一本の1394シリアルバス用ケーブルでディジーチェーン接続するだけで、それぞれの機器が中継の役割を担うので、全体として一つのネットワークを構成することができる。

【0022】

また、1394シリアルバスはPlug and Play機能に対応し、1394シリアルバス用ケーブルを機器に接続するだけで自動的に機器を認識し、接続状況を認識する機能を有している。また、図2に示すようなシステムにおいて、ネットワークからある機器が外されたり、または新たに追加されたときなど、自動的にバスをリセット(それまでのネットワークの構成情報をリセット)して、新たなネットワークを再構築する。この機能によって、その時々ネットワークの構成を常時設定、認識することができる。

【0023】

また、1394シリアルバスのデータ転送速度は、100/200/400Mbpsが定義されていて、上位の転送速度をもつ機器が下位の転送速度をサポートすることで、互換性が保たれている。データ転送モードとしては、コントロール信号などの非同期データを転送する非同期(Asynchronous)転送モード(ATM)と、リアルタイムなAVデータ等の同期データを転送する同期(Isochronous)転送モードがある。この非同期データと同期データは、各サイクル(通常125μs/サイクル)の中で、サイクル開始を示すサイクルスタートパケット(CSP)の転送に続き、同期データの転送を優先しつつ、サイクル内で混在して転送される。

【0024】

図3は1394シリアルバスの構成例を示す図である。1394シリアルバスはレイヤ構造で構成されている。図3に示すように、コネクタポート810には、1394シリアルバス用のケーブル813の先端のコネクタが接続される。コネクタポート810の上位には、ハードウェア部800

10

20

30

40

50

で構成されるフィジカルレイヤ811とリンクレイヤ812がある。ハードウェア部800はインタフェース用チップで構成され、そのうちフィジカルレイヤ811は符号化やコネクション関連の制御等を行い、リンクレイヤ812はパケット転送やサイクルタイムの制御等を行う。

【0025】

ファームウェア部801のトランザクションレイヤ814は、転送（トランザクション）すべきデータの管理を行い、Read、Write、Lockの命令を出す。ファームウェア部801のマネージメントレイヤ815は、1394シリアルバスに接続されている各機器の接続状況やIDの管理を行い、ネットワークの構成を管理する。上記のハードウェアとファームウェアまでが、1394シリアルバスの実質的な構成である。

10

【0026】

また、ソフトウェア部802のアプリケーションレイヤ816は、利用されるソフトによって異なり、インタフェース上でどのようにしてデータを転送するかは、プリンタやAV/Cプロトコルなどのプロトコルによって定義される。

【0027】

図4は1394シリアルバスにおけるアドレス空間の一例を示す図である。1394シリアルバスに接続された各機器（ノード）には必ずノードに固有の64ビットアドレスをもたせる。そして、このアドレスは機器のメモリに格納されていて、自分や相手のノードアドレスを常時認識することで、通信相手を指定したデータ通信を行うことができる。

【0028】

20

1394シリアルバスのアドレッシングは、IEEE1212規格に準じた方式であり、アドレス設定は、最初の10ビットがバスの番号の指定用に、次の6ビットがノードIDの指定用に使われる。

【0029】

それぞれの機器内で使用される48ビットのアドレスについても、20ビットと28ビットに分けられ、256Mバイト単位の構造をもって利用される。最初の20ビットのアドレス空間のうち0～0xFFFFDはメモリ空間、0xFFFFEはプライベート空間、0xFFFFFはレジスタ空間とそれぞれ呼ばれる。プライベート空間は機器内で自由に利用できるアドレスであり、レジスタ空間にはバスに接続された機器間で共通な情報が置かれ、各機器間のコミュニケーションに使われる。

30

【0030】

レジスタ空間の、最初の512バイトにはCSRアーキテクチャのコアになるレジスタ（CSRコア）が、次の512バイトにはシリアルバスのレジスタが、その次の1024バイトにはコンフィグレーションROMが、残りはユニット空間で機器固有のレジスタが、それぞれ置かれる。

【0031】

一般的には異種バスシステムの設計の簡略化のため、ノードは初期ユニット空間の最初の2048バイトだけを使うべきであり、この結果としてCSRコア、シリアルバスのレジスタ、コンフィグレーションROMおよびユニット空間の最初の2048バイトを合わせて4096バイトで構成することが望ましい。

40

【0032】

以上が、1394シリアルバスの概要である。次に、1394シリアルバスの特徴をより詳細に説明する。

【0033】

【1394シリアルバスの詳細】

[1394シリアルバスの電氣的仕様]

図5は1394シリアルバス用のケーブルの断面を示す図である。1394シリアルバス用ケーブルには、二組のツイストペア信号線の他に、電源ラインが設けられている。これによって、電源を持たない機器や、故障などにより電圧が低下した機器にも電力の供給が可能になる。電源線により供給される直流電力の電圧は8～40V、電流は最大電流1.5Aに規定されて

50

いる。なお、DVケーブルと呼ばれる規格では、電源ラインを省いた四線で構成される。

〔DS-Link方式〕

図6は1394シリアルバスで採用されている、データ転送方式のDS-Link(Data/Strobe Link)方式を説明するための図である。

【0034】

DS-Link方式は、高速なシリアルデータ通信に適し、二組の信号線を必要とする。つまり、二組のより対線のうち一組でデータ信号を送り、もう一組でストロブ信号を送る構成になっている。受信側では、このデータ信号と、ストロブ信号との排他的論理和をとることによってクロックを生成することができるという特徴がある。このため、DS-Link方式を用いると、データ信号中にクロック信号を混入させる必要がないので他のシリアルデータ転送方式に比べて転送効率が高い、クロック信号を生成できるので位相ロックループ(PLL)回路が不要になり、その分コントローラLSIの回路規模を小さくすることができる。さらに、転送すべきデータが無いときにアイドル状態であることを示す情報を送る必要が無いので、各機器のトランシーバ回路をスリープ状態にすることができ、消費電力の低減が図れる、などが挙げられる。

10

〔バスリセットのシーケンス〕

1394シリアルバスに接続されている各機器(ノード)にはノードIDが与えられ、ネットワークを構成するノードとして認識される。例えば、ネットワーク機器の接続分離や電源のオン/オフなどによるノード数の増減、つまりネットワーク構成に変化があり、新たなネットワーク構成を認識する必要があるとき、その変化を検知した各ノードはバス上にバスリセット信号を送信して、新たなネットワーク構成を認識するモードに入る。このネットワーク構成の変化の検知は、コネクタポート810においてバイアス電圧の変化を検知することによって行われる。

20

【0035】

あるノードからバスリセット信号が送信されると、各ノードのフィジカルレイヤ811はこのバスリセット信号を受けると同時にリンクレイヤ812にバスリセットの発生を伝達し、かつ他のノードにバスリセット信号を伝達する。最終的にすべてのノードがバスリセット信号を受信した後、バスリセットのシーケンスが起動される。なお、バスリセットのシーケンスは、ケーブルが抜き差しされた場合や、ネットワークの異常等をハードウェアが検出した場合に起動されるとともに、プロトコルによるホスト制御などフィジカルレイヤ811に直接命令を与えることによっても起動される。また、バスリセットのシーケンスが起動されると、データ転送は、一時中断され、バスリセットの間は待たされ、バスリセット終了後、新しいネットワーク構成の基で再開される。

30

〔ノードID決定のシーケンス〕

バスリセットの後、各ノードは新しいネットワーク構成を構築するために、各ノードにIDを与える動作に入る。このときの、バスリセットからノードID決定までの一般的なシーケンスを図7から図9に示すフローチャートを用いて説明する。図7は、バスリセット信号の発生から、ノードIDが決定し、データ転送が行えるようになるまでの一連のシーケンス例を示すフローチャートである。各ノードは、ステップS101でバスリセット信号の発生を常時監視し、バスリセット信号が発生するとステップS102に移り、ネットワーク構成がリセットされた状態において新たなネットワーク構成を得るために、互いに直結されているノード間で親子関係が宣言される。そしてステップS103の判定により、すべてのノード間で親子関係が決ったと判定されるまでステップS102が繰り返される。

40

【0036】

親子関係が決定するとステップS104へ進みルート(root)ノードが決定され、ステップS105で各ノードにIDを与えるノードIDの設定作業が行われる。ルートノードから所定のノード順にノードIDの設定が行われ、ステップS106の判定により、すべてのノードにIDが与えられたと判定されるまでステップS105が繰り返される。

【0037】

ノードIDの設定が終了すると、新しいネットワーク構成がすべてのノードにおいて認識さ

50

れたことになるのでノード間のデータ転送が行える状態になり、ステップS107でデータ転送が開始されるとともに、シーケンスはステップS101へ戻り、再びバスリセット信号の発生が監視される。

【 0 0 3 8 】

図8はバスリセット信号の監視(S101)からルートノードの決定(S104)までの詳細例を示すフローチャート、図9はノードID設定(S105,S106)の詳細例を示すフローチャートである。

【 0 0 3 9 】

図8において、ステップS201でバスリセット信号の発生が監視され、バスリセット信号が発生すると、ネットワーク構成は一旦リセットされる。次に、ステップS202で、リセットされたネットワーク構成を再認識する作業の第一歩として、各機器はフラグFLをリーフノードであることを示すデータでリセットする。そして、ステップS203で、各機器はポート数、つまり自分に接続されている他ノードの数を調べ、ステップS204で、ステップS203の結果に応じて、これから親子関係の宣言を始めるために、未定義（親子関係が決定されていない）ポートの数を調べる。ここで、未定義ポート数は、バスリセットの直後はポート数に等しいが、親子関係が決定されて行くにしたがって、ステップS204で検知される未定義ポートの数は減少する。

【 0 0 4 0 】

バスリセットの直後に親子関係の宣言を行えるのは実際のリーフノードに限られている。リーフノードであるか否かはステップS203のポート数の確認結果から知ることができ、つまりポート数が「1」であればリーフノードである。リーフノードは、ステップS205で、接続相手のノードに対して親子関係の宣言「自分は子、相手は親」を行い動作を終了する。

【 0 0 4 1 】

一方、ステップS203でポート数が「2以上」であったノード、つまりブランチノードは、バスリセットの直後は「未定義ポート数>1」であるからステップS206へ進み、フラグFLにブランチノードを示すデータをセットし、ステップS207で他ノードから親子関係が宣言されるのを待つ。他ノードから親子関係が宣言され、それを受けたブランチノードはステップS204に戻り、未定義ポート数を確認するが、もし未定義ポート数が「1」になっていれば残るポートに接続された他ノードに対して、ステップS205で「自分は子、相手は親」の親子関係を宣言することができる。また、未だ未定義ポート数が「2以上」あるブランチノードは、ステップS207で再び他ノードから親子関係が宣言されるのを待つことになる。

【 0 0 4 2 】

何れか一つのブランチノード（または例外的に、子宣言を行えるのにもかかわらず、すばやく動作しなかったリーフノード）の未定義ポート数が「0」になると、ネットワーク全体の親子関係の宣言が終了したことになる、未定義ポート数が「0」になった唯一のノード、つまりすべてノードの親に決まったノードは、ステップS208でフラグFLにルートノードを示すデータをセットし、ステップS209でルートノードとして認識される。

【 0 0 4 3 】

このようにして、バスリセットから、ネットワーク内のすべてのノード間における親子関係の宣言までの手順が終了する。

【 0 0 4 4 】

次に、各ノードにIDを与える手順を説明するが、最初にIDの設定を行うことができるのはリーフノードである。そして、リーフ ブランチ ルートの順に若い番号（ノード番号：0）からIDを設定する。

【 0 0 4 5 】

図9のステップS301で、フラグFLに設定されたデータを基にノードの種類、つまりリーフ、ブランチおよびルートに応じた処理に分岐する。

【 0 0 4 6 】

まずリーフノードの場合は、ステップS302でネットワーク内に存在するリーフノードの数（自然数）を変数Nに設定した後、ステップS303で各リーフノードがルートノードに対し

10

20

30

40

50

て、ノード番号を要求する。この要求が複数ある場合、ルートノードはステップS304でアービトレーションを行い、ステップS305である一つのノードにノード番号を与え、他のノードにはノード番号の取得失敗を示す結果を通知する。

【0047】

ステップS306の判断により、ノード番号を取得できなかったリーフノードは、再びステップS303でノード番号の要求を繰り返す。一方、ノード番号を取得できたリーフノードは、ステップS307で、取得したノード番号を含むID情報をブロードキャストすることで全ノードに通知する。ID情報のブロードキャストが終わるとステップS308で、リーフ数を表す変数Nがデクリメントされる。そして、ステップS309の判定により変数Nが「0」になるまでステップS303からS308の手順が繰り返され、すべてのリーフノードのID情報がブロードキャストされた後、ステップS310へ進んで、ブランチノードのID設定に移る。

10

【0048】

ブランチノードのID設定もリーフノードとほぼ同様に行われる。まず、ステップS310でネットワーク内に存在するブランチノードの数（自然数）を変数Mに設定した後、ステップS311で各ブランチノードがルートノードに対して、ノード番号を要求する。この要求に対してルートノードは、ステップS312でアービトレーションを行い、ステップS313である一つのブランチノードにリーフノードに続く若い番号を与え、ノード番号を取得できなかったブランチノードには取得失敗を示す結果を通知する。

【0049】

ステップS314の判定により、ノード番号の取得に失敗したことを知ったブランチノードは、再びステップS311でノード番号の要求を繰り返す。一方、ノード番号を取得できたブランチノードはステップS315で、取得したノード番号を含むID情報をブロードキャストすることで全ノードに通知する。ID情報のブロードキャストが終わるとステップS316で、ブランチ数を表す変数Mがデクリメントされる。そして、ステップS317の判定により、変数Mが「0」になるまでステップS311からS316の手順が繰り返され、すべてのブランチノードのID情報がブロードキャストされた後、ステップS318へ進んで、ルートノードのID設定に移る。

20

【0050】

ここまで終了すると、最終的にIDを取得していないノードはルートノードのみなので、ステップS318では、他のノードに与えていない最も若い番号を自分のノード番号に設定し、ステップS319でルートノードのID情報をブロードキャストする。

30

【0051】

以上で、すべてのノードのIDが設定されるまでの手順が終了する。次に、図10に示すネットワーク例を用いてノードID決定のシーケンスの具体的な手順を説明する。

【0052】

図10に示すネットワークは、ルートであるノードBの下位にはノードAとノードCが直結され、ノードCの下位にはノードDが直結され、ノードDの下位にはノードEとノードFが直結された階層構造を有する。この、階層構造やルートノード、ノードIDを決定する手順は以下ようになる。

【0053】

40

バスリセットが発生した後、各ノードの接続状況を認識するために、各ノードの直結されているポート間において、親子関係の宣言がなされる。ここでいう親子とは、階層構造の上位が「親」、下位が「子」という意味である。図10では、バスリセットの後、最初に親子関係を宣言したのはノードAである。前述したように、一つのポートだけが接続されたノード（リーフ）から親子関係の宣言を開始することができる。これは、ポート数が「1」であればネットワークツリーの末端、つまりリーフノードであることが認識され、それらリーフノードの中で最も早く動作を行ったノードから親子関係が決定されて行くことになる。こうして親子関係の宣言を行ったノードのポートが、互いに接続された二つのノードの「子」と設定され、相手ノードのノードが「親」と設定される。こうして、ノードA-B間、ノードE-D間、ノードF-D間で「子-親」の関係が設定される。

50

【 0 0 5 4 】

さらに、階層が一つ上がって、複数のポートをもつノード、つまりブランチノードのうち他ノードから親子関係の宣言を受けたノードから順次、上位のノードに対して親子関係を宣言する。図10ではまずノードD-E間、D-F間の親子関係が決定された後、ノードDがノードCに対して親子関係を宣言し、その結果、ノードD-C間で「子-親」の関係が設定される。ノードDから親子関係の宣言を受けたノードCは、もう一つのポートに接続されているノードBに対して親子関係を宣言し、これによってノードC-B間で「子-親」の関係が設定される。

【 0 0 5 5 】

このようにして、図10に示すような階層構造が構成され、最終的に接続されているすべてのポートにおいて親となったノードBが、ルートノードと決定される。なお、ルートノードは一つのネットワーク構成中に一つしか存在しない。また、ノードAから親子関係を宣言されたノードBが、速やかに、他のノードに対して親子関係を宣言した場合は、例えばノードCなどの他のノードがルートノードになる可能性もあり得る。すなわち、親子関係の宣言が伝達されるタイミングによっては、どのノードもルートノードになる可能性があり、ネットワーク構成が同一であっても、特定のノードがルートノードになるとは限らない。

10

【 0 0 5 6 】

ルートノードが決定されると、各ノードIDの決定モードに入る。すべてのノードは、決定した自分のID情報を、他のすべてのノードに通知するブロードキャスト機能をもっている。なお、ID情報は、ノード番号、接続されている位置の情報、もっているポートの数、接続のあるポートの数、各ポートの親子関係の情報などを含むID情報としてブロードキャストされる。

20

【 0 0 5 7 】

ノード番号の割当ては、前述したようにリーフノードから開始され、順に、ノード番号=0,1,2,...が割当てられる。そしてID情報のブロードキャストによって、そのノード番号は割当て済みであることが認識される。

【 0 0 5 8 】

すべてのリーフノードがノード番号を取得し終わると、次はブランチノードへ移りリーフノードに続くノード番号が割当てられる。リーフノードと同様に、ノード番号が割当てられたブランチノードから順にID情報がブロードキャストされ、最後にルートノードが自己のID情報をブロードキャストする。従って、ルートノードは常に最大のノード番号を所有することになる。

30

【 0 0 5 9 】

以上のようにして、階層構造全体のID設定が終わり、ネットワーク構成が構築され、バスの初期化作業が完了する。

[ノード管理のための制御情報]

ノード管理を行うためのCSRアーキテクチャの基本的な機能として、図4に示したCSRコアがレジスタ上に存在する。それらレジスタの位置と機能を図11に示すが、図中のオフセットは0xFFFFF0000000からの相対位置である。

40

【 0 0 6 0 】

CSRアーキテクチャでは、0xFFFFF0000200からシリアルバスに関するレジスタが配置されている。それらのレジスタの位置と機能を図12に示す。

【 0 0 6 1 】

また、0xFFFFF0000800から始まる場所には、シリアルバスのノード資源に関する情報が配置されている。それらのレジスタの位置と機能を図13に示す。

【 0 0 6 2 】

CSRアーキテクチャでは、各ノードの機能を表すためコンフィグレーションROMをもっているが、このROMには最小形式と一般形式があり、0xFFFFF0000400から配置される。最小形式では図14に示すようにベンダIDを表すだけであり、このベンダIDは24ビットで表される

50

全世界で固有の値である。

【 0 0 6 3 】

また、一般形式は図15に示すような形式で、ノードに関する情報をもっているが、この場合、ベンダIDはルートディレクトリ(root_directory)にもつことができる。また、バス情報ブロック(bus_info_block)とルートリーフ(root_leaf)にはベンダIDを含む64ビットの全世界で固有な装置番号をもっている。この装置番号は、バスリセットなどの再構成後に継続してノードを認識するために使用される。

【 シリアルバス管理 】

1394シリアルバスのプロトコルは、図3に示したように、フィジカルレイヤ811、リンクレイヤ812およびトランザクションレイヤ814から構成されている。この中で、バス管理は、CSRアーキテクチャに基づくノードの制御とバス資源管理のための基本的な機能を提供している。

10

【 0 0 6 4 】

バス管理を行うノード（以下「バス管理ノード」と呼ぶ）は、同一バス上に唯一存在し、シリアルバス上の他のノードに管理機能を提供するが、この管理機能にはサイクルマスタの制御や、性能の最適化、電源管理、伝送速度管理、構成管理などがある。

【 0 0 6 5 】

バス管理機能は、バスマネージャ、同期（アイソクロノス）リソースマネージャおよびノード制御の三つの機能に大きく別けられる。ノード制御は、CSRによってフィジカルレイヤ811、リンクレイヤ812、トランザクションレイヤ814およびアプリケーションにおけるノード間通信を可能にする管理機能である。同期リソースマネージャは、シリアルバス上で同期型のデータ転送を行うために必要になる管理機能で、同期データの転送帯域幅とチャネル番号の割当てを管理するものである。この管理を行うためにバス管理ノードは、バスの初期化後に、同期リソースマネージャ機能をもつノードの中から動的に選出される。

20

【 0 0 6 6 】

また、バス上にバス管理ノードが存在しない構成では、電源管理やサイクルマスタの制御のようなバス管理の一部の機能を同期リソースマネージャ機能をもつノードが行う。さらにバス管理は、アプリケーションに対してバス制御のインタフェースを提供するサービスを行う管理機能であり、その制御インタフェースにはシリアルバス制御要求(SB_コントロール.request)、シリアルバスイベント制御確認(SB_コントロール.confirmation)、シリアルバスイベント通知(SB_EVENT.indication)がある。

30

【 0 0 6 7 】

シリアルバス制御要求は、バスのリセット、バスの初期化、バスの状態情報などを、アプリケーションからバス管理ノードに要求する場合に利用される。シリアルバスイベント制御確認は、シリアルバス制御要求の結果で、バス管理ノードからアプリケーションに確認通知される。シリアルバスイベント通知は、バス管理ノードからアプリケーションに対して、非同期に発生されるイベントを通知するためのものである。

【 データ転送プロトコル 】

1394シリアルバスのデータ転送は、周期的に送信する必要のある同期データ（同期パケット）と、任意タイミングのデータ送受信が許容される非同期データ（非同期パケット）とが同時に存在し、なおかつ、同期データのリアルタイム性を保証している。データ転送では、転送に先立ってバス使用权を要求し、バスの使用許可を得るためのバスアービトレーションが行われる。

40

【 0 0 6 8 】

非同期転送においては、送信ノードIDおよび受信ノードIDが転送データと一緒にパケットデータとして送られる。受信ノードは、自分のノードIDを確認してパケットを受取るとアクノリッジ信号を送信ノードに返すことで、一つのトランザクションが完了する。

【 0 0 6 9 】

同期転送においては、送信ノードが伝送速度とともに同期チャネルを要求し、チャネルIDが転送データと一緒にパケットデータとして送られる。受信ノードは、所望するチャネル

50

IDを確認してデータパケットを受取る。必要になるチャネル数と伝送速度はアプリケーションレイヤ816で決定される。

【0070】

これらのデータ転送プロトコルは、フィジカルレイヤ811、リンクレイヤ812およびトランザクションレイヤ814の三つのレイヤによって定義される。フィジカルレイヤ811は、バスとの物理的・電氣的インタフェース、ノード接続の自動認識、ノード間のバス使用权のバスアービトレーションなどを行う。リンクレイヤ812は、アドレッシング、データチェック、パケット送受信、そして同期転送のためのサイクル制御を行う。トランザクションレイヤ814は、非同期データに関する処理を行う。以下、各レイヤにおける処理について説明する。

10

〔フィジカルレイヤ〕

次に、フィジカルレイヤ811におけるバスアービトレーションを説明する。

【0071】

1394シリアルバスは、データ転送に先立って、必ず、バス使用权のアービトレーションを行う。1394シリアルバスに接続された各機器は、ネットワーク上を転送される信号をそれぞれ中継することによって、ネットワーク内のすべての機器に同信号を伝える論理的なバス型ネットワークを構成するので、パケットの衝突を防ぐ意味でバスアービトレーションが必要である。これによって、ある時間には、一つのノードだけが転送を行うことができる。

【0072】

20

図16はバス使用权の要求を説明する図、図17はバス使用の許可を説明する図である。バスアービトレーションが始まると、一つもしくは複数のノードが親ノードに向かって、それぞれバスの使用权を要求する。図16においては、ノードCとノードFがバス使用权を要求している。この要求を受けた親ノード（図16ではノードA）は、さらに親ノードに向かって、バスの使用权を要求することで、ノードFによるバスの使用权の要求を中継する。この要求は最終的に、アービトレーションを行うルートノードに届けられる。

【0073】

バスの使用权の要求を受けたルートノードは、どのノードにバスの使用权を与えるかを決める。このアービトレーション作業はルートノードのみが行えるものであり、アービトレーションに勝ったノードにはバスの使用許可を与えるられる。図17は、ノードCにバスの使用許可が与えられ、ノードFのバスの使用权の要求は拒否された状態を示している。

30

【0074】

ルートノードは、バスアービトレーションに負けたノードに対してはDP(data prefix)パケットを送り、そのバスの使用权の要求が拒否されたことを知らせる。バスアービトレーションに負けたノードのバスの使用权の要求は、次のバスアービトレーションまで待たされることになる。

【0075】

以上のようにして、アービトレーションに勝ってバス使用の許可を得たノードは、以降、データ転送を開始することができる。ここで、バスアービトレーションの一連の流れを図18に示すフローチャートにより説明する。

40

【0076】

ノードがデータ転送を開始できるためには、バスがアイドル状態であることが必要である。先に開始されたデータ転送が終了し、現在、バスがアイドル状態にあることを確認するためには、各転送モードで個別に設定されている所定のアイドル時間ギャップ長（例えば、サブアクションギャップ）の経過を検出し、所定のギャップ長が検出された場合、各ノードはバスがアイドル状態になったと判断する。各ノードは、ステップS401で、非同期データ、同期データなどそれぞれ転送するデータに応じた所定のギャップ長が検出されたか否かを判断する。所定のギャップ長が検出されない限り、転送を開始するために必要なバス使用权を要求することはできない。

【0077】

50

各ノードは、ステップS401で所定のギャップ長が検出されると、ステップS402で転送すべきデータがあるか判断し、ある場合はステップS403でバスの使用权を要求する信号をルートに対して発信する。このバスの使用权の要求を表す信号は、図16に示すように、ネットワーク内の各機器に中継されながら、最終的にルートノードに届けられる。ステップS402で転送するデータがないと判断した場合は、ステップS401に戻る。

【0078】

ルートノードは、ステップS404でバスの使用权を要求する信号を一つ以上受信したら、ステップS405で使用权を要求したノードの数を調べる。ステップS405の判定により、使用权を要求したノードが一つだったら、そのノードに、直後のバス使用許可が与えられることになる。また、使用权を要求したノードが複数だったら、ステップS406で直後のバス使用許可を与えるノードを一つに絞るアービトレーション作業が行われる。このアービトレーション作業は、毎回同じノードばかりにバスの使用許可を与えるようなことはなく、平等にバスの使用許可を与えるようになっている（フェア・アービトレーション）。

10

【0079】

ルートノードの処理は、ステップS407で、ステップS406のアービトレーションに勝った一つのノードと、敗れたその他のノードとに応じて分岐する。アービトレーションに勝った一つのノード、またはバスの使用权を要求したノードが一つの場合は、ステップS408でそのノードに対してバスの使用許可を示す許可号が送られる。この許可信号を受信したノードは、直後に転送すべきデータ（パケット）の転送を開始する（ステップS410）。また、アービトレーションに敗れたノードにはステップS409で、バス使用权の要求が拒否されたことを示すDP(data prefix)パケットが送られる。DPパケットを受取ったノードの処理は、再度、バスの使用权を要求するためにステップS401まで戻る。ステップS410におけるデータの転送が完了したノードの処理もステップS401へ戻る。

20

【トランザクションレイヤ】

トランザクションの種類には、リードトランザクション、ライトトランザクションおよびロックトランザクションの三種類がある。

【0080】

リードトランザクションでは、イニシエータ（要求ノード）がターゲット（レスポンスノード）のメモリの特定アドレスからデータを読取る。ライトトランザクションでは、イニシエータがターゲットのメモリの特定アドレスにデータを書込む。また、ロックトランザクションでは、イニシエータからターゲットに参照データと更新データを転送する。その参照データは、ターゲットのアドレスのデータと組み合わせられて、ターゲットの特定のアドレスを指示する指定アドレスになる。そして、この指定アドレスのデータが更新データにより更新される。

30

【0081】

図19はトランザクションレイヤ814におけるCSRアーキテクチャに基づくリード、ライト、ロックの各コマンドの要求・レスポンスプロトコルを示す図で、図に示す要求、通知、レスポンスおよび確認は、トランザクションレイヤ814でのサービス単位である。

【0082】

トランザクション要求(TR_DATA.request)はレスポンスノードに対するパケットの転送、トランザクション通知(TR_DATA.indication)はレスポンスノードに要求が届いたことの通知、トランザクションレスポンス(TR_DATA.レスポンス)はアクノリッジの送信、トランザクション確認(TR_DATA.confirmation)はアクノリッジの受信である。

40

【リンクレイヤ】

図20はリンクレイヤ812におけるサービスを示す図で、レスポンスノードに対するパケットの転送を要求するリンク要求(LK_DATA.request)、レスポンスノードにパケット受信を通知するリンク通知(LK_DATA.indication)、レスポンスノードからのアクノリッジ送信のリンクレスポンス(LK_DATA.レスポンス)、要求ノードのアクノリッジ送信のリンク確認(LK_DATA.confirmation)のサービス単位に分けられる。一つのパケット転送プロセスはサブアクションと呼ばれ、非同期サブアクションと同期サブアクションの二つの種類がある。

50

以下では、各サブアクションの動作について説明する。

〔非同期サブアクション〕

非同期サブアクションは非同期データ転送である。図21は非同期転送における時間的な遷移を示す図である。図21に示す最初のサブアクションギャップは、バスのアイドル状態を示すものである。このアイドル時間が所定値になった時点で、データ転送を希望するノードがバス使用权を要求し、バスアービトレーションが実行される。

【0083】

バスアービトレーションによりバスの使用が許可されると、次に、データがパケット転送され、このデータを受信したノードは、ACKギャップという短いギャップの後、受信確認用返送コードACKを返してレスポンスするか、レスポンスパケットを返送することでデータ転送が完了する。ACKは4ビットの情報と4ビットのチェックサムからなり、成功、ビジー状態またはペンディング状態であることを示す情報を含み、すぐにデータ送信元のノードに返される。

【0084】

図22は非同期転送用パケットのフォーマットを示す図である。パケットには、データ部および誤り訂正用のデータCRCのほかにヘッダ部があり、そのヘッダ部には目的ノードID、ソースノードID、転送データ長や各種コードなどが書込まれている。

【0085】

また、非同期転送は送信ノードから受信ノードへの一対一の通信である。送信元ノードから送り出されたパケットは、ネットワーク中の各ノードに行き渡るが、各ノードは自分宛てのパケット以外は無視するので、宛先に指定されたノードだけがそのパケットを受取ることになる。

〔スプリットトランザクション〕

トランザクションレイヤ814におけるサービスは、図19で示したトランザクション要求およびトランザクションレスポンスのセットで行われる。ここで、ターゲット（レスポンスノード）のリンクレイヤ812およびトランザクションレイヤ814における処理が充分高速であれば、要求とレスポンスをリンクレイヤ812のそれぞれ独立したサブアクションで処理せず、一つのサブアクションで処理することが可能になる。しかし、ターゲットの処理速度が遅い場合は、要求とレスポンスを個別のトランザクションで処理する必要がある。そして、この動作をスプリットトランザクションと呼ぶ。

【0086】

図23はスプリットトランザクションの動作例を示す図で、イニシエータ（要求ノード）のコントローラからのライト要求に対して、ターゲットはペンディングを返す。これにより、ターゲットは、コントローラのライト要求に対する確認情報を返すことができ、データを処理するための時間を稼ぐことができる。そして、データ処理に十分な時間が経過した後、ターゲットは、ライトレスポンスをコントローラに通知してライトトランザクションを終了させる。なお、このときの要求とレスポンスのサブアクションの間には、他のノードによるリンクレイヤ812の操作が可能である。

【0087】

図24はスプリットトランザクションを行う場合の転送状態の時間的遷移例を示す図で、サブアクション1は要求サブアクションを、サブアクション2はレスポンスサブアクションをそれぞれ表している。

【0088】

サブアクション1で、イニシエータはライト要求を表すデータパケットをターゲットに送り、これを受けたターゲットはアクノリッジパケットにより上記の確認情報を示すペンディングを返すことで要求サブアクションが終了する。

【0089】

そして、サブアクションギャップが挿入された後、サブアクション2で、ターゲットはデータパケットが無データであるライトレスポンスを送り、これを受けたイニシエータはアクノリッジパケットでコンプリーレスポンスを返すことでレスポンスサブアクションが

10

20

30

40

50

終了する。

【 0 0 9 0 】

なお、サブアクション1の終了からサブアクション2の開始に至る時間は、最小はサブアクションギャップに相当する時間であり、最大はノードに設定された最大待ち時間まで伸ばすことが可能である。

[同期サブアクション]

1394シリアルバスの最大の特徴であるともいえるこの同期転送は、とくにAVデータなどのリアルタイム転送を必要とするデータの転送に適している。また、非同期転送が一对一の転送であるのに対し、この非同期転送はブロードキャスト機能によって、一つの送信元ノードから他のすべてのノードへ様にデータを転送することができる。

10

【 0 0 9 1 】

図25は同期転送における時間的な遷移を示す図で、同期転送はバス上で一定時間毎に実行され、この時間間隔を同期サイクルと呼ぶ。同期サイクル時間は125 μ sである。この同期サイクルの開始を示し、各ノードの動作を同期させる役割を担っているのがサイクルスタートパケット(CSP)2000である。CSP2000を送信するのは、サイクルマスタと呼ばれるノードであり、一つ前のサイクル内の転送が終了し、所定のアイドル期間(サブアクションギャップ2001)を経た後、本サイクルの開始を告げるCSP2000を送信する。つまり、このCSP2000が送信される時間間隔が125 μ sになる。

【 0 0 9 2 】

また、図25にチャンネルA、チャンネルBおよびチャンネルCと示すように、一つの同期サイクル内において複数種のパケットにチャンネルIDをそれぞれ与えることにより、それらのパケットを区別して転送することができる。これにより、複数ノード間で、略同時に、リアルタイム転送が可能であり、また、受信ノードは所望するチャンネルIDのデータのみを受信すればよい。このチャンネルIDは、受信ノードのアドレスなどを表すものではなく、データに対する論理的な番号に過ぎない。従って、送信されたあるパケットは、一つの送信元ノードから他のすべてのノードに行き渡る、つまりブロードキャストされることになる。

20

【 0 0 9 3 】

同期転送によるパケット送信に先立ち、非同期転送と同様に、バスアービトレーションが行われる。しかし、非同期転送のように一对一の通信ではないので、同期転送には受信確認用の返送コードのACKは存在しない。

30

【 0 0 9 4 】

また、図25に示したisoギャップ(同期ギャップ)は、同期転送を行う前にバスがアイドル状態であることを確認するために必要なアイドル期間を表している。この所定のアイドル期間を検出したノードは、バスがアイドル状態にあると判断し、同期転送を行いたい場合はバス使用权を要求するのでバスアービトレーションが行われることになる。

【 0 0 9 5 】

図26は同期転送用のパケットフォーマット例を示す図である。各チャンネルに分けられた各種のパケットには、それぞれデータ部および誤り訂正用のデータCRCのほかにヘッダ部があり、そのヘッダ部には図27に示すような、転送データ長、チャンネル番号、その他各種コードおよび誤り訂正用のヘッダCRCなどが書込まれている。

40

[バス・サイクル]

実際に、1394シリアルバスにおいては、同期転送と非同期転送が混在できる。図28は同期転送と非同期転送が混在するときの転送状態の時間的遷移を示す図である。

【 0 0 9 6 】

ここで、前述したように同期転送は非同期転送より優先して実行される。その理由は、CSPの後、非同期転送を起動するために必要なアイドル期間のギャップ(サブアクションギャップ)よりも短いギャップ(アイソクロナスギャップ)で、同期転送を起動できるからである。従って、非同期転送より同期転送は優先して実行されることになる。

【 0 0 9 7 】

図28に示す一般的なバスサイクルにおいて、サイクル#mのスタート時にCSPがサイクルマ

50

スタから各ノードに転送される。CSPによって、各ノードの動作が同期され、所定のアイドル期間（同期ギャップ）を待ってから同期転送を行おうとするノードはバスアービトレーションに参加し、パケット転送に入る。図28ではチャンネルe、チャンネルsおよびチャンネルkが順に同期転送されている。

【0098】

このバスアービトレーションからパケット転送までの動作を、与えられているチャンネル分繰り返し行った後、サイクル#mにおける同期転送がすべて終了すると、非同期転送を行うことができるようになる。つまり、アイドル時間が、非同期転送が可能なサブアクションギャップに達することによって、非同期転送を行いたいノードはバスアービトレーションに参加する。ただし、非同期転送が行えるのは、同期転送の終了から、次のCSPを転送すべき時間(cycle synch)までの間に、非同期転送を起動するためのサブアクションギャップが検出された場合に限られる。

10

【0099】

図28に示すサイクル#mでは、三つのチャンネル分の同期転送の後、非同期転送によりACKを含む2パケット（パケット1、パケット2）が転送されている。この非同期パケット2の後、サイクルm+1をスタートすべき時間(cycle synch)に至るので、サイクル#mにおける転送はこれで終わる。ただし、非同期または同期転送中に次のCSPを送信すべき時間(cycle synch)に至ったら、転送を無理に中断せず、その転送が終了した後にアイドル期間を経て次の同期サイクルのCSPを送信する。すなわち、一つの同期サイクルが125 μ s以上続いたときは、その延長分、次の同期サイクルは基準の125 μ sより短縮される。このように同期サイクルは125 μ sを基準に超過、短縮し得るものである。

20

【0100】

しかし、同期転送はリアルタイム転送を維持するために、必要であれば毎サイクル実行され、非同期転送は同期サイクル時間が短縮されたことによって次以降の同期サイクルに延期されることもある。サイクルマスタは、こういった遅延情報も管理する。

【0101】

【データ転送処理】

[データ転送プロトコル]

図29は、1394シリアルバス上におけるデータ転送のためのプロトコルスタックを説明するための図である。

30

【0102】

アプリケーションは、しばしば画像データといった大量のデータを、デバイス間でやり取りする必要がある。そのような場合に、データ転送に関わるハードウェア技術の細部、制限事項、データのエラー再転送の処理等をアプリケーションプログラムから分離するために、アプリケーションに信頼性のあるデータ転送サービスを提供し、汎用的なインタフェースを定める必要がある。

【0103】

そこで本実施形態においては、アプリケーションプログラムがデータ転送を行なう際に、図29に示すような階層化されたプロトコル体系を用いる。図29は、上から順にアプリケーションレイヤ29-1、セッションレイヤ29-2、トランザクションレイヤ29-3、以下、IEEE1394で定められた1394トランザクションレイヤ29-4、1394フィジカルレイヤ29-5を示す。ここで、1394トランザクションレイヤ29-4が上述した図3に示すトランザクションレイヤ814に、1394フィジカルレイヤ29-5がリンクレイヤ812及びフィジカルレイヤ811に相当する。

40

【0104】

本実施形態においてデータを送信するデバイスは、以下のような動作をする。まずデータ転送アプリケーション29-1は、画像データ等の大量のデータを、データ転送APIのような抽象化されたインタフェースを用いて下位のセッションレイヤ29-2に渡す。セッションレイヤ29-2は、アプリケーションデータを図30で定義されているブロックレジスタ30-3の単位に分割し、下位のトランザクションレイヤ29-3に順次渡していく。トランザクションレイヤ29-3は、ブロックレジスタ30-3単位のアプリケーションデータを、下位の1394トラン

50

ザクションレイヤ29-4のライトトランザクションに適した単位に分割し、1394トランザクション29-4のライトトランザクションサービスインタフェースを呼び出す。1394トランザクションレイヤ29-4及び1394フィジカルレイヤ29-5は、IEEE1394で定義された手段で、上記分割されたアプリケーションデータを他のデバイスに転送する。

【0105】

また、データを受信するデバイスは、以下のような動作をする。送信側から転送されてきた適切に分割されたデータは、1394フィジカルレイヤ29-5及び1394トランザクションレイヤ29-4により、トランザクションレイヤ29-3で定義されるデータ受信デバイスのブロックレジスタ30-3に、順次書き込まれる。トランザクションレイヤ29-3は、ブロックレジスタ30-3に順次書き込まれてくるデータを再構成し、1ブロックレジスタ単位のデータに組み立て、上位のセッションレイヤ29-2に渡す。セッションレイヤ29-2は、上記1ブロックレジスタ単位のデータを順次受け取り、データストリームの形に再構成してアプリケーション29-1に渡す。データ受信側アプリケーション29-1は、抽象化されたインタフェースを介して、データ送信側デバイスから画像データ等の大量のデータを受け取る。

[レジスタ構成]

図30は、図29のトランザクションレイヤ29-3がデバイス同士でデータ転送を行うためのレジスタ構成を示した図であり、1394シリアルバスの初期ユニット空間(図4のユニット空間で示される)に配置される。デバイスはデータ転送を行うに先立って、転送相手のレジスタが配置されているアドレス、サイズを予め知っており、データ転送を行うためにこれらのレジスタを使用する。レジスタは実際のデータを書き込むブロックレジスタ30-3と、ブロックレジスタ30-3への書き込み制御(書き込み完了通知)を行うためのコントロールレジスタ30-1と、コントロールレジスタ30-1への書き込み完了通知に対して、書き込まれたデータの正否(ACK/NACK)を返答するためのレスポンスレジスタ30-2から構成される。コントロールレジスタ30-1とレスポンスレジスタ30-2を合わせて、ブロックマネージメントレジスタと呼ぶ。これらのレジスタは、画像データを転送する画像供給デバイスと、画像データを受けて印字を行うプリンタの双方に準備され、相互にデータ転送を行うことが可能である。

【0106】

図31に、図30に示したレジスタを用いたコマンドの転送の概要を示す。これは、画像供給デバイスからプリンタへのコマンド転送の例であり、画像供給デバイスは、プリンタに送るべきコマンドデータをプリンタ側のブロックレジスタ31-6に書き込む(コマンド31-9)。次に、画像供給デバイスはプリンタ側のコントロールレジスタ31-4にコマンドデータの転送終了を通知するための書き込みを行う(コントロール31-7)。プリンタは、正常にデータを受け取れたか否かを画像供給デバイスに返答するために、画像供給デバイスのレスポンスレジスタにACKあるいはNACKに相当する内容を書き込む(レスポンス31-8)。

【0107】

以上の手順で、画像供給デバイスからプリンタのレジスタへの、コマンドの書き込みと、該書き込みに対する返答が行われる。プリンタからACKの返答が画像供給デバイスのレスポンスレジスタに書き込まれれば、コマンドは正常にプリンタに受け取られたことを示し、NACKが戻された場合には、何らかの理由でデータが正しくプリンタへ送られていないことを示す。返答がNACKである場合には、画像供給デバイスはエラーに対する終了処理やコマンドの再転送処理等、何らかのエラー処理を行う必要がある。

【0108】

図32は、図31でレジスタに書き込まれたコマンドに対する返答を、プリンタから画像供給デバイスに戻す場合の手順を示している。プリンタは画像供給デバイスに送るべきリプライ(返答)を画像供給デバイス側のブロックレジスタ32-3に書き込む(リプライ32-9)。次にプリンタは、画像供給デバイス側のコントロールレジスタ32-1にリプライの転送終了を通知するための書き込みを行う(コントロール32-7)。画像供給デバイスは、正常にデータを受け取れたか否かをプリンタに返答するために、画像供給デバイスのレスポンスレジスタ32-5にACKあるいはNACKに相当する内容を書き込む(レスポンス32-8)。

【 0 1 0 9 】

以上の手順で、プリンタから画像供給デバイスのレジスタへの、リプライの書き込みと、該書き込みに対する返答が行われる。画像供給デバイスからACKの返答がプリンタのレスポンスレジスタに書き込まれば、リプライは正常に画像供給デバイスに受け取られたことを示し、NACKが戻された場合には、何らかの理由でデータが正しく画像供給デバイスへ送られていないことを示す。返答がNACKである場合には、プリンタはエラーに対する終了処理やコマンドの再転送処理等、何らかのエラー処理を行う必要がある。

【 0 1 1 0 】

以上、図31、図32を参照して、画像供給デバイスからプリンタへのコマンド及びそれに対するリプライの転送手順を示した。これらが一般的なコマンドとリプライの手順である。10
なお、コマンドによってはリプライを必要としないものも考えられ、この場合には図32に示した手順が実行されないことになる。これらはコマンドごとに決める事ができる。

【 0 1 1 1 】

図33は、ブロックレジスタ33-1と、装置が内部的に有するバッファとの関係を示した図である。ブロックレジスタ33-1と同じ大きさを持つ複数のバッファBlockbuffer[1] ~ [n] (33-2 ~ 33-7)が装置内部に用意されており、ブロックレジスタ33-1に対する書き込みが行われると、Blockbuffer[1] ~ [n]に保存される。ブロックレジスタ33-1に対する書き込みは、内部バッファに空きがある場合、書き込まれたデータをバッファに保存した後、次に保存ができる空きバッファがある場合に、ACKを画像供給デバイスに返す。バッファに空きがなくなると、最後のバッファに対する保存を行った後、空のバッファができるまでACK 20
は画像供給デバイスに返されない。

【 0 1 1 2 】

画像供給デバイスにおいては、プリンタ側にバッファの空きができてACKが戻されるまで、次のコマンドの転送は行えないことになる。プリンタにおいてバッファに空きができるのは、印字を行うデータの場合、例えばBlockbuffer[1]のデータを印字のためのデータに変換し、該バッファ内のデータが全部処理されるとバッファは空になり、再びブロックレジスタ33-1に書き込まれたデータを保存することができるようになる。

【 0 1 1 3 】

図34は、上部が図30に示したコントロールレジスタ30-1の構成を示しており、コントロールコマンド34-1にその制御内容が設定される。例えば、コントロールコマンド34-1が01h 30
であればBLOCK COMPLETEを表し、すなわちブロックレジスタへの書き込み完了を示す。

【 0 1 1 4 】

また、図34の下部が図30に示したレスポンスレジスタ30-2の構成を示しており、レスポンスコマンド34-2にその返答内容が設定される。例えば、レスポンスコマンド34-2が01hであればBLOCK ACKを表し、即ちコントロールレジスタ30-1へBLOCK COMPLETEが書き込まれた後、データが正しく受け取られたことを示し、02hであればBLOCK NACKを表し、即ちコントロールレジスタ30-1へBLOCK COMPLETEが書き込まれた後、データが正しく受け取られなかったことを示す。

【 0 1 1 5 】

これらのコントロールコマンドとレスポンスコマンドを用いることにより、ブロックレジスタへのデータの書き込み完了の通知とそれに対する返答を行うことができる。 40

【 0 1 1 6 】

図35は、ブロックレジスタ30-3に書き込まれるコマンドの一般形式を示した図である。コマンドは、その識別子であるCommandID35-1と、コマンドに付随するParameter35-2からなる。

【 0 1 1 7 】

図36は、実際のコマンドの例を示す図である。同図において、左側は画像データを転送するためのコマンド(SENDコマンド)であり、CommandIDをSEND(36-1)とし、Parameterとして、転送する画像データであるImage Data(36-2)を有する。このコマンドにより、画像供給デバイスからプリンタへ印字を行うための画像データの転送を行うことができる。 50

【 0 1 1 8 】

また、図36の右側は画像供給デバイスがプリンタの状態を示すステータスをプリンタから受け取るためのコマンド(GETSTATUSコマンド)であり、CommandIDをGETSTATUS(36-3)とし、Parameterは有しない。即ち、GETSTATUSコマンドではParameterを必要としない。

【 0 1 1 9 】

これらのコマンドが、図31のコマンド31-9として画像供給デバイスからプリンタへ転送される。但し、上記SENDコマンドの場合、画像データはブロックレジスタ30-3の大きさに比べて大きい場合が普通であり、この場合にはブロックレジスタ30-3に合ったサイズでの書き込みを複数回行うことになる。

【 0 1 2 0 】

図37は、図36に示したコマンドに対するリプライを示す図である。同図において、左側はSENDコマンドに対するリプライ(SENDリプライ)である。このリプライは、SENDコマンドに対してプリンタから画像供給デバイスにコマンドの実行ステータスを戻すものであり、SENDリプライ37-1と、コマンドの実行ステータスを示すSENDリプライステータス37-2から構成される。

【 0 1 2 1 】

また、図37の右側はGETSTATUSコマンドに対するリプライである。このリプライは、GETSTATUSコマンドに対してプリンタから画像供給デバイスにプリンタのステータスを戻すものである。

【 0 1 2 2 】

これらのリプライが、図32のリプライ32-9としてプリンタから画像供給デバイスへ転送される。

【 0 1 2 3 】

図36、図37に示したコマンドとリプライにより、図29に示した画像供給デバイスのアプリケーションが、プリンタのアプリケーションに対してイメージデータの転送による印字要求、及びプリンタのステータス取得を行なうことが可能となる。

【 0 1 2 4 】

図38は、図37のGETSTATUSリプライで戻される、プリンタステータス38-1の詳細を示した図であり、例えば、紙なし、エラー、ビジー等のステータスを持つことができる。画像供給デバイスはこのステータスにより、プリンタの現在の状態を知ることができる。例えば、画像供給デバイスにプリンタの用紙切れを知ることができ、この場合には画像供給デバイスのユーザにその旨を報知することもできる。また、プリンタにエラーが発生しているような場合には、印字を行わないように制御することも可能となる。

【 0 1 2 5 】

図39は、SENDコマンドにより画像データ39-1を送る際に、ブロックレジスタ31-3に対して画像データ39-1のサイズが大きい場合を示す。この場合、一度には転送できないため、同図に示すように画像データ39-1をブロックレジスタ30-3のサイズに合わせて分割し、複数のコマンド39-2～39-5として転送する。この処理は、図29に示すセッションレイヤ29-2で行われる。以降、これら分割された一回のコマンド転送を、WriteBlockと称する。従って、大量の画像データを転送する場合、WriteBlockが複数回実行されることになる。尚、画像供給デバイスからプリンタのブロックレジスタ31-6に転送された画像データは、図33に示したように、プリンタ側の内部的なバッファに保存される。

【 0 1 2 6 】

そして、セッションレイヤ29-2で制御されてWriteBlockされる画像データはさらに、トランザクションレイヤ29-3において、IEEE1394規格に準拠したデータ転送単位(1934トランザクション39-6～39-9)に分解される。即ち1394シリアルバス上において、1394トランザクション39-6単位でのデータ転送が複数回行われることにより、ブロックレジスタ31-3の全ての領域への画像データの書き込みが遂行される。

[一般的なデータ転送制御]

図40は、Write Blockを繰り返し行なう場合の、画像供給デバイスとプリンタの一般的な

10

20

30

40

50

制御手順を示す図である。画像供給デバイスはプリンタに対して、分割されたコマンド単位でWriteBlockを行う。具体的には、SENDコマンドが繰り返し転送される。

【 0 1 2 7 】

まず、ステップS40-1において、画像供給デバイスは第1番目のコマンドをプリンタのブロックレジスタ31-6に書き込むためのWriteBlockを行なう。そしてブロックレジスタ31-3の分のデータ書き込みが終了すると、ステップS40-2においてWriteBlockの完了を通知するBLOCK COMPLETEをプリンタのコントロールレジスタ31-4に書き込む。

【 0 1 2 8 】

これに対して、プリンタはバッファの空きがあり、ブロックレジスタに書き込まれたコマンドが正常であるため、ステップS40-3でBLOCK ACKを画像供給デバイスへのレスポンスレジスタ31-2に書き込む。この場合の、BLOCK COMPLETEの発生からBLOCK ACKが戻されるまでの応答時間をT1で示す。このような、WriteBlockからBLOCK ACKまでの一連の処理(ステップS40-1～S40-3)により、一つのブロック単位での転送が終了する。以降、ステップS40-4～S40-6に示されるように、プリンタの空きバッファがなくなるまで、WriteBlockが繰り返される。

10

【 0 1 2 9 】

プリンタは、印字動作に従って送られたデータを消費してバッファを空とし、バッファの再利用を行う。しかし、画像供給デバイスからのデータ転送がプリンタのバッファ消費速度よりも速い場合、空きバッファが無くなるため、ステップS40-7に示すプリンタの最終バッファに対するWriteBlockが行われることになる。これは、画像供給デバイスはプリンタのバッファに対する残り数などの情報を持たないため、BLOCK ACKが戻されれば直ちにWriteBlockを実行するためである。尚、図40は、プリンタ側がn個のバッファを有している場合を示している。

20

【 0 1 3 0 】

ステップS40-7において最終バッファに対してWriteBlockが行われ、更にステップS40-8でBLOCK COMPLETEが転送されると、この転送によってプリンタのバッファが一杯となるため、バッファに空きができるまで、プリンタはBLOCK ACKを画像供給デバイスに転送できなくなる。そのため、通常はBLOCK COMPLETEからBLOCK ACKまでの応答時間はT1で済んでいたのに対し、バッファが空になるまでのT2時間の間、画像供給デバイスはデータ転送ができないことになる。そしてT2が経過してプリンタからBLOCK ACKが戻された後、画像供給デバイスは次のデータを転送することができるようになる。即ち、応答時間がT1よりもはるかに長いT2となってしまう。

30

【 0 1 3 1 】

図41は、本実施形態におけるGETSTATUSコマンドの制御手順を示す図であり、SENDコマンドにより画像データを転送している途中でGETSTATUSコマンドを実行する場合処理の流れを示している。

【 0 1 3 2 】

WriteBlockを繰り返し行っている間に、プリンタのステータスを取る必要がある場合、GETSTATUSコマンドは定期的に行われる。即ち、GETSTATUSコマンドがSENDコマンドの途中で実行される。まずステップS41-1～41-6までSENDコマンドが実行され、所定時間になると、ステップS41-7でGETSTATUSコマンドが実行される。

40

【 0 1 3 3 】

以下、GETSTATUSコマンド制御について詳細に説明する。まずステップS41-8において、WriteBlockでGETSTATUSコマンドがプリンタのブロックレジスタに書き込まれる。次に、ステップS41-9でBLOCK COMPLETEがプリンタのコントロールレジスタに書き込まれる。これに対して、ステップS41-10でプリンタから画像供給デバイスのレスポンスレジスタにBLOCK ACKが戻され、ステップS41-11でプリンタ側からGETSTATUSリプライが画像供給デバイスのブロックレジスタに書き込まれる。そしてステップS41-12において、プリンタからBLOCK COMPLETEが画像供給デバイスのコントロールレジスタに書き込まれ、ステップS41-13でそれに対するBLOCK ACKがプリンタのレスポンスレジスタに戻される。

50

【0134】

ステップS41-7に示す一連の処理により、GETSTATUSコマンドがSENDコマンド中に割り込み実行される。そしてGETSTATUSコマンドの実行後、ステップS41-14～S41-16において、中断されていたSENDコマンドが再開される。即ち、SENDコマンドの間にGETSTATUSコマンドが割り込んで実行され、GETSTATUSコマンドの終了を待って、SENDコマンドが再開される。

【0135】

従って画像供給デバイスは、SENDコマンドの実行中であってもGETSTATUSコマンドによってプリンタの現在のステータスを得ることができ、プリンタの状況を確認しながらデータ転送を行うことができる。このGETSTATUSコマンドの割り込み処理は、図29に示すアプリケーションレイヤ29-1から指示され、セッションレイヤ29-2で処理される。即ち、画像供給デバイスのアプリケーションがSENDコマンドをセッションレイヤに指示した後、特定の時間ごとに、プリンタのステータスを得るために、アプリケーションがセッションレイヤに対してGETSTATUSコマンドを発行することを意味している。

10

【0136】

しかし、図41に示した例においてGETSTATUSコマンドが実行できるのは、プリンタのバッファに空きがある場合に限られる。例えば、図40のステップS40-8～S40-9に示した、WriteBlockが行えない応答時間T2の間に、GETSTATUSコマンドを実行する時間になってしまった場合には、GETSTATUSコマンドが転送できないという不都合が生じる。これは上述したように、プリンタ側に空きバッファが無いためであり、この応答時間T2が長ければ、一定の時間毎にプリンタのステータスを得ることができなくなってしまう。このように、プリンタ側におけるバッファの残り数によっては、プリンタステータスが定期的に得られないという問題が生じてしまうことがある。

20

【0137】

【本実施形態におけるデータ転送処理】

[ダミー処理概要]

本実施形態は、上述した図29～図41を参照して説明した一般的なコマンド転送における問題を解決する。以下、この解決方法について詳細に説明する。尚、以下の説明は、上述した図34、図40を適当に変更したものであり、図29～図41の他の図については同様である。

【0138】

本実施形態においては、上述した図34で示したコントロールレジスタ30-1及びレスポンスレジスタ30-3の構成を、図42に示すように変更したことを特徴とする。図42の下部はレスポンスレジスタ30-3の構成を示すが、BLOCK count42-3が追加されていることを特徴とする。BLOCK count42-3は、プリンタ側が持つ空きバッファの数を設定する部分であり、従って本実施形態においては、プリンタはWriteBlockに対するACK/NACKの他に、BLOCK countを画像供給デバイスに戻すことができる。

30

【0139】

図43は、上述した図36で示したSENDコマンド及びGETSTATUSコマンドの他に、本実施形態で追加されるDUMMYBLOCKコマンド(以下、単にDUMMYコマンドと称する)を示す。これは画像供給デバイスからプリンタに送られるコマンドであり、画像供給デバイスはプリンタから戻されるレスポンスにおいてBLOCK count42-3で示される空きブロック数が予め定められた数よりも少なくなった場合に、SENDのコマンドに代わってプリンタに送り出される。プリンタ側は、このDUMMYコマンドを受け取ると、内部のブロックバッファ33-2～33-7に保存を行わず、かつ空きバッファの数を変えずに、BLOCK ACK, NACKを画像供給デバイスのレスポンスレジスタ31-2に戻す。このように、DUMMYコマンドは対応するリプライを持たないコマンドである。尚、DUMMYコマンド処理の詳細について後述する。

40

[DUMMYコマンド制御手順]

本実施形態においては、上述した図40に示したWriteBlockのリピート処理を、図44に示すように変更したことを特徴とし、その差異は以下の点にある。まず、図40のステップS40-3等で示したBLOCK ACK返答に対して、図44のステップS44-3のBLOCK ACK返答においては残

50

りバッファ数を示すBLOCK count42-3がセットされている点である。そして更に、図40では、ステップS40-7で示した最終バッファへのWriteBlockとステップS40-9でBLOCK ACKが戻されるまでの応答時間T2は転送待ちの状態となるのに対して、図44では、ステップS44-9でBLOCK countが1としてBLOCK ACKが戻された場合、BLOCK countが m (m は1以外の値)でBLOCK ACKが戻されるまで、ステップS44-10のダミー処理を行う点である。

【0140】

以下、本実施形態におけるダミー処理について説明する。まずステップS44-11において、画像供給デバイスがBLOCK ACKのBLOCK countが1となったことを判断して、SENDコマンドに代えてDUMMYコマンドをプリンタのブロックレジスタに書き込む。そしてステップS44-12においてWriteBlockの完了を通知するBLOCK COMPLETEをプリンタのコントロールレジスタに書き込む。するとステップS44-13で画像供給デバイスのレスポンスレジスタにBLOCK ACKが戻されるが、このBLOCK ACKにセットされたBLOCK countが1である限り、ステップS44-13～S44-14に示すように、プリンタのブロックレジスタへのDUMMYコマンドの書き込みを継続する。

【0141】

尚、プリンタ側においては、ブロックレジスタにDUMMYコマンドが書き込まれるとDUMMYコマンドであることを判定し、該DUMMYコマンドをバッファに保存せずにBLOCK ACKを戻す。従って、返答する時点の空きバッファの数がBLOCK countとして戻されるため、空きバッファが増えていればBLOCK countは増え、空きバッファが増えていなければ引き続きBLOCK countに1をセットしてBLOCK ACKを戻す。

【0142】

画像供給デバイスにおいては、ステップS44-16でBLOCK ACKのBLOCK countに1以外の数がセットされて戻されると、ステップS44-17～S44-22において、ダミー処理で中断されていたSENDコマンドの転送処理を再開する。

【0143】

以上、図44において示したように本実施形態においては、プリンタ側のバッファが空くまでダミー処理を行うことができる。従って、プリンタ側のバッファに空きがない状態であっても、以下に示すようにGETSTATUSコマンドを実行することができる。

[DUMMY, GETSTATUSコマンド制御手順]

以下、図45に本実施形態におけるDUMMYコマンド及びGETSTATUSコマンドの制御手順を示す。上述したように、図44のステップS44-9においてBLOCK countが1となった場合に、本実施形態のダミー処理が実行される。図45においては、ステップS45-5がダミー処理を、ステップS45-8がダミー処理中におけるGETSTATUSコマンド実行処理を示す。以下、ステップS45-8におけるダミー処理中のGETSTATUSコマンド実行処理について詳細に説明する。

【0144】

ステップS45-5に示すダミー処理の間に、画像供給デバイスがプリンタのステータスを取り込むべき時期となった場合、画像供給デバイスはプリンタのブロックレジスタに書き込んでいたDUMMYコマンドに代えて、GETSTATUSコマンドの書き込み(ステップS45-9)を行う。尚、GETSTATUSコマンドの実行処理は、上述した図41と同様であるため、説明を省略する。

【0145】

そしてステップS45-11において、GETSTATUSコマンドに対して戻されたBLOCK ACKにセットされたBLOCK countが依然として1であれば、GETSTATUSコマンド処理の実行後も、ステップS45-15～S45-17に示すように、BLOCK countが1以外になるまでDUMMYコマンド処理が実行される。

【0146】

ステップS45-17においてBLOCK countが1以外の数が戻されると、ステップS45-18以降、上述した図44において説明したのと同様に、SENDコマンドの処理が再開される。

【0147】

以上、図44, 図45を参照して説明したように本実施形態においては、まずプリンタ側のバ

10

20

30

40

50

ッファの残り数を知ることができ、その残り数が1となった場合にダミー処理が実行される。ダミー処理中、プリンタ側ではDUMMYコマンドがバッファに保存されることが無いため、残りバッファ数が減ることは無い。従って、ダミー処理中にGETSTATUSコマンド等の他のコマンドをブロックレジスタに書き込むことができる。即ち、図40に示す例においては、空きバッファがない間、即ち応答時間T2の間は、GETSTATUSコマンドを割り込んで発行することができなかったが、本実施形態ではDUMMYコマンドを設けたことにより、空きバッファがなくてもGETSTATUSコマンドを割り込み発行することができる。尚、この際の画像供給デバイス、プリンタの各処理の詳細については、後述する。

[SEND前のDUMMYコマンド制御手順]

図46は、本実施形態におけるSENDコマンド前のDUMMYコマンドの制御手順を示す図であり、即ち、一番最初のSENDコマンドを実行する前にDUMMYコマンドを送る場合の手順を示す。

10

【0148】

図46において、ステップS46-4が最初のSENDコマンドであり、それに先立ってステップS46-1～S46-3においてDUMMY処理が行われる。このように、まずDUMMYコマンド処理を最初に行うことにより、バッファがまだSENDコマンドによって使用されていない状態であるため、ステップS46-3で戻されるBLOCK countがプリンタの持つバッファの総数を示すため、SENDコマンドを開始する前に、プリンタが持つバッファの総数を知ることができる。即ち、バッファの使用を開始するのに先立って、バッファの総数を知ることが可能となる。

【0149】

20

例えばプリンタがバッファを1つしか持たない場合にダミー処理を実行すると、SENDコマンドがいつまでも実行されず、DUMMYコマンドが無限に繰り返されてしまうことが考えられる。従って、図46に示すように、SENDコマンドに先立ってDUMMYコマンドを実行することで、プリンタのバッファ数が1である場合にはダミー処理を行わないように制御することができる。

[トラフィック効率化]

図47は、上述した図44でも説明した、DUMMYコマンドの制御手順を示す図であり、BLOCK COMPLETEに対するBLOCK ACKの応答時間がT1であることを示す。このように、画像供給デバイスはBLOCK ACKがBLOCK countが1にセットされて戻る限りにおいて、プリンタ側のブロックレジスタへのDUMMYコマンドの書き込みを繰り返す。

30

【0150】

ここで、応答時間T1が短いとダミー処理が数多く実行されることになり、バス上のトラフィックを無駄に増加させることになる。これを画像供給デバイス側において解決するには、BLOCK ACK後、次のDUMMYコマンドをブロックレジスタに書き込むまでにタイマによって時間計測を行い、ブロックレジスタにDUMMYコマンドが書き込まれる回数を調整する方法が考えられる。しかし、この方法では画像供給デバイスの処理が増え、複雑になってしまう。

【0151】

そこで本実施形態においては以下に示す方法により、1394シリアルバス上のトラフィックの効率化を実現する。

40

【0152】

図48は、本実施形態においてトラフィックの増加を抑制するためのDUMMYコマンド制御手順を示す図である。図48においては、プリンタがBLOCK COMPLETEに対してBLOCK ACKを戻すまで(ステップS48-2～S48-3等)の応答時間を、T1よりも長いT1'に調整することを特徴とする。これにより、画像供給デバイス側ではBLOCK ACKが戻ればすぐにDUMMYコマンドをプリンタのブロックレジスタへ書き込むことができる。

【0153】

また、応答時間T1'をプリンタ側で制御できるため、印字速度やバッファの消費速度等、プリンタ側の条件を考慮してT1'を調整することができる。

このように本実施形態においては、プリンタ側で応答時間をT1'に調整するだけで、画像

50

供給デバイス側に処理を追加することなく、1394シリアルバス上のトラフィックの効率化を実現することができる。

【0154】

以下、本実施形態におけるデータ転送処理に関し、画像供給デバイス側及びプリンタ側のそれぞれについて詳細に説明する。

[画像供給デバイスにおける画像転送処理]

図49は、本実施形態の画像供給デバイスにおける画像転送処理を示すフローチャートである。ここで画像転送処理は、画像供給デバイスがプリンタに対して画像データを転送しプリンタで印字を行う処理であり、まずステップS500において、DUMMYコマンド書込み処理を行う。これは上述した図46において説明したように、相手のバッファの数を確認するための処理である。次にステップS501において、ステップS500で戻されたBLOCK countが1であるか否かを判定し、1であればステップS502へ進んでENADUMMYSENDflagを0にセットする。ここでENADUMMYSENDflagは、DUMMY処理を行うか否かを判定するためのフラグであり、0ならばダミー処理は行わず、1であればダミー処理を行うことを示す。ステップS501においてBLOCK countが1でなければ、ステップS503へ進んでENADUMMYSENDflagを1にセットする。そしてステップS504に進み、後述するSENDコマンド処理を実行する。

10

【0155】

図50は、上記ステップS504における画像供給デバイスのSENDコマンド処理の詳細を示すフローチャートである。まずステップS600において、SNEDコマンド処理中にGETSTATUSコマンドの要求があったか否かの判定を行う。本実施形態においては、タイマ処理により一定時間ごとにGETSTATUSコマンドの割り込みがかかり、GETSTATUSコマンド要求が設定される。このGETSTATUSコマンドの割り込みの詳細については後述する。ステップS600でGETSTATUSコマンド要求があれば、ステップS601へ進んでGETSTATUSコマンド処理を実行する。これは上述した図38で説明したプリンタのステータスを取り出す処理であり、SNEDコマンド実行中にプリンタにエラーや異常がないかを確認するために利用される。

20

【0156】

ステップS601のGETSTATUSコマンド処理が終了すると、処理はステップS600にもどる。ステップS600でGETSTATUSコマンド要求がなければ、ステップS602へ進んでBLOCK countが1であることをチェックする。BLOCK countが1であれば、プリンタの持つ空きバッファが残り1つであることを示しており、ステップS603へ進んでENADUMMYflagが1であることをチェックする。ENADUMMYBLOCKflagはDUMMYコマンド処理が実行可能であるか否かを示すフラグであり、上述した図49のステップS502、S503において、プリンタ側の持つ空きバッファの総数に応じてその設定がなされる。ステップS603でENADUMMYBLOCKflagが1でなければ、DUMMYコマンド処理は行わずにステップS600へ戻る。この処理は、本実施形態において図46に示した、プリンタ側のバッファの総数が1である場合にDUMMYコマンド処理を行わないようにする処理に相当する。ステップS603でENADUMMYBLOCKflagが1であれば、ステップS604へ進んで、上述した図45においてステップS45-4で示したDUMMYコマンドの転送処理を実行する。即ち、プリンタ側のバッファ総数が1以上でBLOCK countが1となった（プリンタの残り空きバッファが1となった）場合に、DUMMYコマンドの転送処理を行う。そして次にステップS606へ進む。

30

40

【0157】

一方、ステップS602においてBLOCK countが1でなければステップS605へ進み、SENDコマンドの転送処理を実行する。これは、上述した図45においてステップS45-1、S45-18、S45-21に相当し、プリンタへ画像データを送る処理である。ここで、画像データが1つのバッファに収まらないような場合、上述した図39で示したように、該イメージデータは複数のSNEDコマンドに分割されてプリンタのブロックレジスタに書き込まれる。ステップS605は分割された画像データの1つを書き込む処理に相当し、ステップS605が繰り返し実行されることにより、画像データ全体が転送される。

【0158】

次にステップS606へ進み、プリンタ側のコントロールレジスタにBLOCK COMPLETEを書き込

50

む。このBLOCK COMPLETEの書き込みにより、ブロックレジスタへのデータ書き込みの終了がプリンタ側に通知される。

【0159】

次にステップS607へ進み、タイムアウトかどうかをチェックする。即ち、ステップS606でプリンタのコントロールレジスタにBLOCK COMPLETEを書き込んでから、画像供給デバイスのレスポンスレジスタにBLOCK ACK/NACKが戻されるまでの時間をタイマにより計測し、所定時間が経過してもBLOCK ACK/NACKが戻されない場合に、タイムアウトとなる。ステップS607でタイムアウトとなった場合、ステップS608へ進みタイムアウト処理を行なう。ここでタイムアウト処理は、ユーザに何らかの理由でタイムアウトが発生し、プリンタへのデータ送信がうまくいかなかったことを報知したり、プリンタへのデータ転送処理を終了する処理である。タイムアウト処理後、SENDコマンド処理は終了する。

10

【0160】

ステップS607においてタイムアウトでない場合、ステップS609へ進んでプリンタから画像供給デバイスのレスポンスレジスタにBLOCK NACKが戻されたか否かをチェックする。BLOCK NACKが戻された場合、プリンタへ送ったデータに何らかの不具合があったため、ステップS610のNACK処理へ進む。ステップS610のNACK処理は、タイムアウト処理と同様に何らかの理由でプリンタへのデータ送信がうまくいかなかったため、プリンタへのデータ転送処理を終了することを行う。NACK処理後、SENDコマンド処理は終了する。

【0161】

ステップS609においてBLOCK NACKでない場合、ステップS611へ進んでプリンタからBLOCK ACKが戻されたか否かをチェックし、BLOCK ACKでなければステップS607へ戻る。BLOCK ACKならばステップS612へ進み、SNEDコマンドが終了したか否かをチェックする。終了でなければステップS600へ戻り、上述した処理を繰り返す。一方、ステップS612でSENDコマンドが終了していれば、SENDコマンド処理は終了となる。

20

【0162】

以上、図50を参照して説明したように、本実施形態の画像供給デバイスにおいては、SNED、DUMMYコマンドの実行とSENDコマンド実行中のGETSTATUSコマンドの処理が行えることが分かる。これは、上述した図45で説明したDUMMY、GETSTATUSコマンド制御手順において画像供給デバイス側の手順に相当する処理である。

【0163】

図51は、上述した図50のステップS601に示したGETSTATUSコマンド処理の詳細を示すフローチャートであり、即ち、画像供給デバイスが、プリンタからステイタスを受け取るためのコマンドをプリンタ側のブロックレジスタに書き込む処理である。まずステップS700において、GETSTATUSコマンドをプリンタ側のブロックレジスタに書き込む。これは、上述した図45のステップS45-9に相当する。次にステップS701へ進み、プリンタのコントロールレジスタにBLOCK COMPLETEを書き込む。これは、図45のステップS45-10に相当する。次にステップS702へ進み、タイムアウトか否かをチェックする。即ち、ステップS701でプリンタのコントロールレジスタにBLOCK COMPLETEを書き込んでからプリンタからレスポンスレジスタにBLOCK ACK/NACKが戻されるまでの応答時間を計測し、所定時間が経過してもBLOCK ACK/NACKが戻されない場合にタイムアウトとなる。ステップS702でタイムアウトとなった場合、ステップS703へ進んでタイムアウト処理を行なう。このタイムアウト処理は、図50に示したステップS608と同様である。タイムアウト処理後、GETSTATUSコマンド処理は終了する。

30

40

【0164】

ステップS702でタイムアウトでなければステップS704へ進み、プリンタから画像供給デバイスのレスポンスレジスタにBLOCK NACKが戻されたか否かをチェックする。BLOCK NACKが戻された場合、プリンタへ送ったデータに何らかの不具合があったため、ステップS705のNACK処理へ進む。このNACK処理は、図50のステップS610と同様である。NACK処理後、GETSTATUSコマンド処理は終了する。ステップS704でBLOCK NACKでなければステップS706へ進み、プリンタからBLOCK ACKが戻されたか否かをチェックする。BLOCK ACKでなければステ

50

ップS702へ戻る。以上、ステップS702～S706までの処理は、図45のステップS45-11に相当する。

【0165】

次にステップS707へ進み、ステップS707～S709によりBLOCK COMPLETEまでのタイムアウトをチェックする。これは、図45のステップS45-12～S45-13のプリンタからのBLOCK COMPLETE応答時間を判定し、タイムアウトであればBLOCK COMPLETEが所定時間内に届かなかったことを意味し、処理はステップS708へ進む。即ち、ステップS708はステップS703と同様のタイムアウト処理であり、その後、GETSTATUSコマンド処理は終了する。ステップS707でタイムアウトでなければステップS709へ進み、画像供給デバイスのコントロールレジスタにBLOCK COMPLETEが書き込まれたか否かをチェックする。BLOCK COMPLETEでなければステ

10

【0166】

ステップS709においてBLOCK COMPLETEであればステップS710へ進み、GETSTATUSリプライを取り込む、これは、図37の右側に示されたGETSTATUSコマンドに対する返答であり、その内容は図38に示される。次にステップS711で、ステップS710のリプライが正常であったか、即ち、戻されたステータスが画像供給デバイスのブロックレジスタに正しく書き込まれたか否かを判定する。正しければ、ステップS712でプリンタのレスポンスレジスタにBLOCK ACKを書き込み、正しくなければ、ステップS713でBLOCK NACKを書き込む。このステップS712、S713の処理が、図45のステップS45-14に相当する。そしてGETSTATUSコマンド処理は終了となる。

20

【0167】

以上、図51で示したGETSTATUSコマンド処理(図45のステップS45-8)により、画像供給デバイスはプリンタからステータスを得ることができる。

【0168】

図52は、画像供給デバイスにおけるタイマ割り込み処理の詳細を示すフローチャートである。画像供給デバイスでは、一定時間ごとにこのタイマ割り込み処理が呼び出される。まずステップS800でGETSTATUSコマンド処理を行う時間であるか否かをチェックする。GETSTATUS実行時間であればステップS801へ進み、GETSTATUSコマンド要求をセットする。この要求が図50のステップS600で判定されることにより、上述したGETSTATUSコマンド処理が実行される。ステップS801で要求セット後、タイマ処理は終了する。

30

[プリンタにおける画像転送処理]

図53は、プリンタ側でのSENDコマンド処理の詳細を示すフローチャートである。まずステップS900で、プリンタ側のコントロールレジスタにBLOCK COMPLETEが書き込まれたかチェックする。BLOCK COMPLETEが書き込まれていなければステップS901へ進み、印字処理を実行した後ステップS900へ戻る。この印字処理については後述する。ステップS900でコントロールレジスタにBLOCK COMPLETEが書き込まれていれば、ステップS902へ進んで書き込まれたコマンドを取り込む。このコマンドは、上述した図35に示したように、CommandID35-1とParameter35-2からなり、このCommandIDを調べることによりコマンドの種類を知ることができる。

【0169】

40

次にステップS903へ進み、ブロックレジスタに書き込まれたコマンドがGETSTATUSコマンドであるか否か、即ち、CommandIDがGETSTATUSコマンドを示しているかチェックする。GETSTATUSコマンドであればステップS904へ進み、後述するプリンタ側のGETSTATUSコマンド処理を実行する。GETSTATUSコマンド処理の実行後は、ステップS900へ戻る。ステップS903でGETSTATUSコマンドでなければ、次にDUMMYコマンドか否かをチェックする。DUMMYコマンドでなければ、該コマンドはSENDコマンドであるとしてステップS906へ進み、SENDコマンドによって送られた画像データが正常であるか否かをチェックする。正常でなければステップS907へ進み、画像供給デバイスのレスポンスレジスタにBLOCK NACKを書き込む。これにより、画像データが正しく送られてこなかったことが画像供給デバイスに通知されたことになる。

50

【 0 1 7 0 】

ステップS906で画像データが正常であればステップS908へ進み、ブロックレジスタへ書き込まれた画像データを内部の適当なバッファへ移動する。これは即ち、ブロックレジスタ内の画像データを図33の33-2～33-7で示したバッファのいずれかへ移動する処理である。そしてステップS909へ進み、残りの空きバッファ数を示すBLOCK countを1減算する。次にステップS910へ進み、画像供給デバイスのレスポンスレジスタにBLOCK ACKを書き込んでステップS900へ戻る。

【 0 1 7 1 】

ステップS905においてコマンドがDUMMYコマンドであれば、ステップS911へ進んでBLOCK countが1より大きいかなかをチェックする。大きければ、最終バッファでのDUMMYコマンドではないため、ステップS910へ進んで画像供給デバイスのレスポンスレジスタにBLOCK ACKを書き込む。一方、ステップS911でBLOCK countが1であれば、ステップS912へ進んでT1'時間のウェイトを行う。ここでT1'は、上述した図48に示すBLOCK COMPLETEに対してBLOCK ACKを戻すまでの応答時間であり、図47に示すT1よりも長い時間が設定される。このT1'時間により、BLOCK ACKが戻されるまでの時間が調整され、結果として画像供給デバイスからプリンタへ転送されるDUMMYコマンド数を減らすことができる。

【 0 1 7 2 】

以上、図53に示したSENDコマンド処理により、プリンタ側での画像データ受け取り処理が実行され、SENDコマンド実行中にGETSTATUS、DUMMYコマンドの処理が可能となり、BLOCK countで示される空きバッファ数によりACKを返す時間の調整ができる。

【 0 1 7 3 】

図54は、上述した図53のステップS901で示した印字処理を示すフローチャートである。まずステップS1000で、バッファにデータがあるかなかをチェックする。バッファにデータが無ければ印字処理は終了するが、データがあればステップS1001へ進み、先頭バッファのデータを印字するための処理を行う。ここで先頭バッファとは、まだ印字処理がされていないデータの先頭にあるバッファを意味し、ステップS1002において該先頭バッファが空であるかなかをチェックする。先頭バッファが空でなければ印字処理は終了し、空であればステップS1003へ進んでBLOCK countに1加算する。即ち、先頭バッファのデータがすべて印字処理されて空きとなった場合に、空きバッファ数を1つ増やす。そして印字処理は終了する。以上の印字処理により、本実施形態における印字の実行と空きバッファ管理が行える。

【 0 1 7 4 】

図55は、上述した図53のステップS904で示された、プリンタにおけるGETSTATUSコマンド処理の詳細を示すフローチャートである。これは、画像供給デバイスからのGETSTATUSコマンドに対してプリンタ側からその返答を戻す処理であり、まずステップS1100で戻すべきステータスを作成する。このステータスの内容は上述した図38に示した通りであり、プリンタの状態を調査して、対応する内容を設定する。次にステップS1101において、GETSTATUSリプライを画像供給デバイスのブロックレジスタに書き込む。これは、上述した図45のステップS45-12に相当する。

【 0 1 7 5 】

次にステップS1102において、画像供給デバイスのコントロールレジスタにBLOCK COMPLETEを書き込む。これは図45のステップS45-13に相当する。そして次にステップS1103へ進み、タイムアウトをチェックする。即ち、ステップS1102で画像供給デバイスのコントロールレジスタにBLOCK COMPLETEを書き込んでから、プリンタのレスポンスレジスタにBLOCK ACK/NACKが戻されるまでの時間を計測し、所定時間が経過してもBLOCK ACK/NACKが戻されない場合に、タイムアウトとなる。ステップS1103でタイムアウトとなればステップS1104へ進み、タイムアウト処理を行なう。これは上述した図50のステップS608と同様の処理である。タイムアウト処理後は、GETSTATUSコマンド処理を終了する。

【 0 1 7 6 】

ステップS1103でタイムアウトでなければ、ステップS1105へ進んで画像供給デバイスから

10

20

30

40

50

プリンタのレスポンスレジスタにBLOCK NACKが戻されたか否かをチェックする。BLOCK NACKが戻された場合、画像供給デバイスへ送ったデータに何らかの不具合があったため、ステップS1106へ進んで図50のステップS610と同様のNACK処理を行なう。NACK処理後、GETSTATUSコマンド処理は終了する。一方、ステップS1105でBLOCK NACKでなければステップS1107へ進み、画像供給デバイスからBLOCK ACKが戻されたか否かをチェックする。BLOCK ACKでなければステップS1103へ戻る。以上ステップS1103～S1107までの処理は、図45のステップS45-14に相当する。そしてステップS1107でBLOCK ACKであれば、GETSTATUS処理は終了する。

【0177】

以上、図55に示したプリンタ側でのGETSTATUSコマンド処理により、画像供給デバイスにプリンタのステータスが戻される。

10

【0178】

以上、図42～図55を参照して説明したように本実施形態によれば、画像供給デバイスは、プリンタからレスポンスレジスタに書き込まれるBLOCK ACK/NACK及びBLOCK countにより、プリンタが現在有する空きバッファ数を知ることができ、空きバッファ数が1となった場合に、SENDコマンドに代えてDUMMYコマンドを送る処理が実行できる。また、DUMMYコマンド実行中にGETSTATUSコマンドが実行できるため、SENDコマンドが実行中であっても、任意にGETSTATUSコマンドを実行することができる。このDUMMYコマンドは、SENDコマンド等の他のコマンドに比べて少ないデータ量で構成することができるため、1394シリアルバス上のトラフィックに与える影響を最小限に留めることができる。

20

【0179】

また、プリンタ側でDUMMYコマンドが送られてくる時間間隔を調整することができるため、画像供給デバイス側に負荷をかけることなく、1394シリアルバス上のトラフィックを効率化することができる。

【0180】

また、プリンタの総バッファ数が1である場合にはDUMMYコマンド処理を行わないようにすることで、必要なSENDコマンド処理を妨げないようにすることができる。

【第2実施形態】

以下、本発明に係る第2実施形態について説明する。

〔レジスタ構成〕

30

図56は、上述した第1実施形態において図34に示した一般的なコントロールレジスタ及びレスポンスレジスタの構成を、第2実施形態において改良した構成を示す図である。第2実施形態においては、図56の下部に示すレスポンスレジスタ構成において、レスポンスコマンド56-2として03hのBLOCK RETRYが追加されていることを特徴とする。ここでBLOCK RETRYとは、例えばプリンタ側が持つ空きバッファ数が1となった場合等において、画像供給デバイスへ画像データの再転送を要求することを示す返答である。従って、レスポンスコマンド56-2が03hであればBLOCK RETRYを表し、即ち画像データの再転送を要求することを示す。

【0181】

図57は、第2実施形態におけるRETRY、GETSTATUSコマンド制御手順を示す図であり、上述した第1実施形態において図45に示したDUMMY、GETSTATUSコマンド制御手順に対応するものであるが、以下の点において図45と異なることを特徴とする。まず、図45のステップS45-3ではBLOCK countを1で戻していたのに対し、図57のステップS57-3ではBLOCK RETRYを戻している点が異なる。そして、図45のステップS45-4ではブロックレジスタにDUMMYコマンドを書き込んでいるのに対して、図57のステップS57-4では、ステップS57-1に示す直前のSENDコマンドと同じデータを書き込んでいる点が異なる。尚、図57における他の処理は図45と同様であるため、説明を省略する。

40

【0182】

以下、第2実施形態における画像供給デバイス側の処理及びプリンタ側の処理を、図58～図60を参照して詳細に説明する。

50

[画像供給デバイスにおける画像転送処理]

図58は、第2実施形態における画像供給デバイスの画像転送処理を示すフローチャートであり、実質的にステップS1200におけるSENDコマンド処理を行なう。

【0183】

図59は、上述したステップS1200のSENDコマンド処理の詳細を示すフローチャートである。まずステップS1300でGETSTATUSコマンド要求があるか否かの判定を行う。尚、GETSTATUSコマンドの起動は、第1実施形態で示した図52と同様に行われる。ステップS1300で要求があればステップS1301へ進み、GETSTATUSコマンド処理を実行する。このGETSTATUSコマンド処理は、第1実施形態において図51で示したプリンタのステータスを取り出す処理と同様である。ステップS1301のGETSTATUSコマンド処理が終了すると、処理はステップS1300 10

【0184】

ステップS1300でGETSTATUSコマンド要求がなければステップS1302へ進み、レスポンスレジスタにBLOCK_RETRYが書かれたか否かをチェックする。BLOCK_RETRYが書かれていれば、プリンタのもつ空きバッファが残り1つであり、データの再転送を要求していることを示しているため、処理はステップS1303へ進んで、プリンタ側のコントロールレジスタに対して1つ前のSENDコマンドで書き込まれたデータを再度書き込む。これは、図57のステップS57-4、S57-15に相当する。ステップS1302でBLOCK_RETRYでなければ、ステップS1304へ進んでSENDコマンドの転送処理を実行する。これは、図57のステップS57-1、S57-18、S57-21に相当し、プリンタへ画像データを送る処理となる。ここで、画像データが1つのバッ 20

ファに収まらないような場合、第1実施形態において図39で示したように、該イメージデータは複数のSNEDコマンドに分割されてプリンタのブロックレジスタに書き込まれる。ステップS1303、S1304は分割された画像データの1つを書き込む処理に相当し、ステップS1304が繰り返し実行されることにより、画像データ全体が転送される。

【0185】

次にステップS1305へ進み、プリンタ側のコントロールレジスタにBLOCK_COMPLETEを書き込む。このBLOCK_COMPLETEの書き込みにより、ブロックレジスタへのデータ書き込みの終了がプリンタ側に通知される。次にステップS1306へ進み、タイムアウトかどうかをチェックする。即ち、ステップS1305でプリンタのコントロールレジスタにBLOCK_COMPLETEを書き込んでから画像供給デバイスのレスポンスレジスタにBLOCK_ACK/NACK又はRETRYが戻さ 30

れるまでの時間をタイマにより計測し、所定時間が経過してもBLOCK_ACK/NACK及びRETRYが戻されない場合に、タイムアウトとなる。ステップS1306でタイムアウトとなった場合、ステップS1307へ進みタイムアウト処理を行なう。ここでタイムアウト処理は、ユーザに何らかの理由でタイムアウトが発生し、プリンタへのデータ送信がうまくいかなかったことを報知したり、プリンタへの画像転送処理を終了する処理である。タイムアウト処理後、SENDコマンド処理は終了する。

【0186】

ステップS1306においてタイムアウトでない場合、ステップS1308へ進んでプリンタから画像供給デバイスのレスポンスレジスタにBLOCK_NACKが戻されたか否かをチェックする。BLOCK_NACKが戻された場合、プリンタへ送ったデータに何らかの不具合があったため、ステ 40

ップS1309のNACK処理へ進む。ステップS1309のNACK処理は、タイムアウト処理と同様に何らかの理由でプリンタへのデータ送信がうまくいかなかったため、プリンタへの画像転送処理を終了することを行う。NACK処理後、SENDコマンド処理は終了する。

【0187】

ステップS1308においてBLOCK_NACKでない場合、ステップS1310へ進んでプリンタからBLOCK_RETRYが戻されたか否かをチェックし、BLOCK_RETRYが戻されればステップS1300へ戻る。BLOCK_RETRYが戻されなければステップS1311へ進み、BLOCK_ACKがプリンタから戻されたか否かをチェックし、BLOCK_ACKでなければステップS1306へ戻る。BLOCK_ACKならばステップS1312へ進み、SNEDコマンドが終了したか否かをチェックする。終了でなければステップS1300へ戻り、上述した処理を繰り返す。一方、ステップS1312でSENDコマンドが終 50

了していれば、SENDコマンド処理は終了となる。

【0188】

以上、図59を参照して説明したように、第2実施形態の画像供給デバイスにおいては、SENDコマンドの実行とBLOCK RETRYに対する処理、及びSENDコマンド実行中のGETSTATUSコマンドの処理が行えることが分かる。これは、上述した図57で説明したRETRY、GETSTATUSコマンド制御手順において画像供給デバイス側の手順に相当する処理である。

[プリンタにおける画像転送処理]

図60は、第2実施形態のプリンタにおけるSENDコマンド処理を詳細に示すフローチャートである。

【0189】

まずステップS1400で、プリンタ側のコントロールレジスタにBLOCK COMPLETEが書き込まれたかをチェックする。BLOCK COMPLETEが書き込まれていなければステップS1401へ進み印字処理を実行し、ステップS1400へ戻る。この印字処理は、第1実施形態で説明した図53のステップS901と同様である。ステップS1400でBLOCK COMPLETEが書き込まれていれば、ステップS1402へ進んで書き込まれたコマンドを取り込む。このコマンドは、上述した図35に示したように、CommandID35-1とParameter35-2からなり、このCommandIDを調べることでによりコマンドの種類を知ることができる。

【0190】

次にステップS1403へ進み、書き込まれたコマンドがGETSTATUSコマンドであるか否か、即ち、CommandIDがGETSTATUSコマンドを示しているかをチェックする。GETSTATUSコマンドであればステップS1404へ進み、プリンタ側のGETSTATUSコマンド処理を実行した後、ステップS1400へ戻る。このGETSTATUSコマンド処理は、第1実施形態で示した図55と同様である。ステップS1403でGETSTATUSコマンドでなければ、該コマンドはSENDコマンドであるとしてステップS1405へ進み、BLOCK countが1より大きいかな否かをチェックする。

【0191】

BLOCK countが1より大きければステップS1408へ進み、SENDコマンドによって送られた画像データが正常であるか否かをチェックする。正常でなければステップS907へ進み、画像供給デバイスのレスポンスレジスタにBLOCK NACKを書き込む。これにより、画像データが正しく送られてこなかったことが画像供給デバイスに通知されたことになる。ステップS1406で画像データが正常であればステップS1408へ進み、ブロックレジスタへ書き込まれた画像データを内部の適当なバッファへ移動する。これは即ち、ブロックレジスタ内の画像データを図33の33-2～33-7で示したバッファのいずれかへ移動する処理である。そしてステップS1409へ進み、残りの空きバッファ数を示すBLOCK countを1減算する。次にステップS1410へ進み、画像供給デバイスのレスポンスレジスタにBLOCK ACKを書き込んでステップS1400へ戻る。

【0192】

一方、ステップS1405でBLOCK countが1であれば、ステップS1411へ進んでT1'時間のウェイトを行う。ここでT1'は、上述した図48に示すBLOCK COMPLETEに対してBLOCK ACKを戻すまでの応答時間であり、図47に示すT1よりも長い時間が設定される。そしてT1'時間の経過後、ステップS1412でSENDコマンドの再送を促すBLOCK RETRYを画像供給デバイスのレスポンスレジスタに書き込んでステップS1400に戻る。第2実施形態では、上述した第1実施形態において図47、図48で示したDUMMYコマンドの転送処理に代えて、SENDコマンドの再送を行なうことを特徴とする。従って、このT1'時間によりBLOCK RETRYが戻されるまでの時間が調整され、結果として画像供給デバイスからプリンタへ再送されるSENDコマンド数を減らすことができる。

【0193】

以上、図60に示したSENDコマンド処理により、プリンタ側での画像データ受け取り処理が実行され、SENDコマンド実行中にGETSTATUSコマンド処理が可能となるばかりでなく、BLOCK countで示される空きバッファ数によりSENDコマンドの再送処理が可能となる。

【0194】

10

20

30

40

50

以上、図56～図60を参照して説明したように第2実施形態によれば、画像供給デバイスはプリンタから返送されるBLOCK ACK/NACK及びBLOCK RETRYにより、プリンタが現在有する空きバッファ数が1となった場合にSENDコマンドの再転送処理が実行できる。また、SNEDコマンドの再転送実行中にGETSTATUSコマンドが実行できるため、SENDコマンド実行中であっても任意にGETSTATUSコマンドを実行することができる。

【0195】

また、このSENDコマンドの再送処理においては、既存のバスリセットやエラー処理等を利用することができるため、新たな処理を追加することなく実現できる。

【0196】

また、プリンタ側でSENDコマンドが再送される時間間隔を調整することができるため、画像供給デバイスに負荷をかけることなく、1394シリアルバス上のトラフィックを効率化することができる。

10

【第3実施形態】

以下、本発明に係わる第3実施形態について説明する。

〔レジスタ〕

図61は、上述した第1実施形態において図34に示した一般的なコントロールレジスタ及びレスポンスレジスタの構成を、第3実施形態において改良した構成を示す図である。

【0197】

第3実施形態においては、図61に示すコントロールレジスタ構成において、コントロールコマンドとして02hのBLOCK DUMMY COMPLETEが追加されていることを特徴とする。ここで、BLOCK DUMMY COMPLETEとは、第1実施形態においては、DUMMYコマンドを書き込んだ後、BLOCK COMPLETEを書き込んでDUMMYコマンドをプリンタ側に通知するものであったのに対して、コントロールレジスタのコマンドにBLOCK DUMMY COMPLETEを持つことでプリンタ側はBLOCK DUMMY COMPLETEがコントロールレジスタに書き込まれたことを検知することで、第1実施形態と同様の働きを行わせるものである。

20

【0198】

図62は、第3実施形態の実施例におけるDUMMY処理の制御手順を示す図であり、上述した第1実施形態において図44に示したDUMMY処理の制御手順に対応するものであるが、以下の点において図44と異なることを特徴とする。まず、図44のステップS44-11，S44-14ではWriteBlockでDUMMYコマンドを書き込んでいたのに対して図62ではこのステップがなくなっている点異なる。そして、図44のステップS44-12，S44-15ではBLOCK COMPLETEをコントロールレジスタに書き込んでいるのに対して、図62ではステップS62-11，S62-13でBLOCK DUMMY COMPLETEを書き込んでいる点異なる。尚、図62における他の処理は図44と同様であるため、説明は省略する。

30

【0199】

図63は、第3実施形態の実施例におけるDUMMY処理、GETSTATUSコマンドの制御手順を示す図であり、上述した第1実施形態において図45に示したDUMMY処理、GETSTATUSコマンドの制御手順に対応するものであるが、以下の点において図45と異なることを特徴とする。まず、図45のステップS45-4，S45-15ではWriteBlockでDUMMYコマンドを書き込んでいたのに対して図63ではこのステップがなくなっている点異なる。そして、図45のステップS44-6，S44-16ではBLOCK COMPLETEをコントロールレジスタに書き込んでいるのに対して、図63ではステップS63-5，S63-14でBLOCK DUMMY COMPLETEを書き込んでいる点異なる。尚、図62における他の処理は図44と同様であるため、説明は省略する。

40

【0200】

図64は、第3実施形態の実施例におけるイメージ送信処理に先立って行われるDUMMY処理の制御手順を示す図であり、上述した第1実施形態において図46に示したDUMMY処理の制御手順に対応するものであるが、以下の点において図46と異なることを特徴とする。まず、図46のステップS46-1ではWriteBlockでDUMMYコマンドを書き込んでいたのに対して図64ではこのステップがなくなっている点異なる。そして、図46のステップS46-2ではBLOCK COMPLETEをコントロールレジスタに書き込んでいるのに対して、図64ではステップS64-1でBL

50

OCK DUMMY COMPLETEを書き込んでいる点が異なる。尚、図64における他の処理は図46と同様であるため、説明は省略する。

【0201】

以下、第3実施形態における画像供給デバイス側の処理及びプリンタ側の処理を、図65～図67を参照して詳細に説明する。

〔画像供給デバイスにおける画像転送処理〕

図65は、第3実施形態の実施例におけるイメージ送信処理の詳細を示すフローチャートであり、上述した第1実施形態において図49に示したイメージ送信処理の詳細を示すフローチャートに対応するものであるが、以下の点において図49と異なることを特徴とする。まず、図49のステップS500では、WriteBlockでDUMMYコマンドを書き込んでいたのに対して図65ではBLOCK DUMMY COMPLETEを書き込んでいる点が異なる。尚、図65における他の処理は図49と同様であるため、説明は省略する。

10

【0202】

図66は、第3実施形態の実施例におけるSENDコマンド処理の詳細を示すフローチャートであり、上述した第1実施形態において図50に示したSENDコマンド処理の詳細を示すフローチャートに対応するものであるが、以下の点において図50と異なることを特徴とする。まず、図50のステップS604ではWriteBlockでDUMMYコマンドを書き込んでステップS606へ進んでいたのに対して図66ではステップS1604でBLOCK DUMMY COMPLETEを書き込んでステップS1607へ進んでいる点が異なる。尚、図66における他の処理は図50と同様であるため、説明は省略する。

20

〔プリンタにおける画像転送処理〕

図67は、第3実施形態の実施例におけるプリンタ側でのSENDコマンド処理の詳細を示すフローチャートであり、上述した第1実施形態において図53に示したプリンタ側でのSENDコマンド処理の詳細を示すフローチャートに対応するものである。

【0203】

まず、ステップS1700で、プリンタ側のコントロールレジスタにBLOCK DUMMY COMPLETEが書き込まれたかどうかをチェックする。BLOCK DUMMY COMPLETEが書き込まれていればステップS1701へ進み、画像供給デバイスのレスポンスレジスタにBLOCK ACKを書き込んでステップS1700に戻る。ステップS1700でBLOCK DUMMY COMPLETEが書き込まれていなければ、ステップS1702へ進み、BLOCK COMPLETEが書き込まれたかどうかをチェックする。BLOCK COMPLETEが書き込まれていなければステップS1703へ進み、印字処理を実行した後ステップS1700へ戻る。印字処理は第1実施形態の図54で説明したものと同様である。ステップS1702でBLOCK COMPLETEが書き込まれていれば、ステップS1704へ進んで書き込まれたコマンドを書き込む。このコマンドは第1実施形態の図35に示したように、CommandID35-1とParameter35-2からなり、このCommandIDを調べることによりコマンドの種類を知ることができる。次にステップS1705へ進み、ブロックレジスタに書き込まれたコマンドがGETSTATUSコマンドであるか否か、すなわち、CommandIDがGETSTATUSコマンドを示しているかどうかをチェックする。GETSTATUSコマンドであればステップS1706へ進み、プリンタ側のGETSTATUS処理を実行する。GETSTATUS処理は第1実施形態の図55で説明したものと同様である。GETSTATUS処理の実行後は、ステップS1700へ戻る。ステップS1705でGETSTATUSコマンドでなければ、ステップS1707へ進み、該コマンドがSENDコマンドであるためSENDコマンドによって送られてきた画像データが正常であるか否か进行检查する。正常でなければ、ステップS1708へ進み、画像供給デバイスのレスポンスレジスタにBLOCK NACKを書き込む。これにより、画像データが正しく送られてこなかったことが画像供給デバイスに通知されたことになる。ステップS1707で画像データが正常であればステップS1709へ進み、ブロックレジスタへ書き込まれた画像データを内部の適当なバッファへ移動する。これは即ち、ブロックレジスタ内の画像データを図33の33-2～33-7で示したバッファのいずれかへ移動する処理である。そしてステップS1710へ進み、残りの空きバッファ数を示すBLOCK countを1減算する。次にステップS1711へ進み、画像供給デバイスのレスポンスレジスタにBLOCK ACKを書き込んでステップS1700へ戻る。

30

40

50

【0204】

以上、図61～図67を参照して説明したように第3実施形態によれば、画像供給デバイスは、プリンタからのレスポンスレジスタに書き込まれるBLOCK ACK/NACK及びBLOCK countにより、プリンタが現在有する空きバッファ数を知ることができ、空きバッファ数が1となった場合に、SENDコマンドに変えてプリンタ側のコントロールレジスタにBLOCK DUMMY COMPLETEを書き込む処理が実行できる。また、DUMMY処理実行中にGETSTATUSコマンドが実行できるため、SENDコマンド実行中であっても、任意にGETSTATUSコマンドを実行することができる。

【実施形態の変形例】

なお、上述した第1及び第2実施形態においては、プリンタ内の空きバッファ数が1となった場合に、DUMMYコマンド処理やBLOCK RETRYの処理を行なう例について説明したが、この空きバッファ数は1に限られるわけではない。SENDコマンド処理を実行するのに不都合が生じるような数であればどのような数であっても良く、例えば2や3であっても構わない。

【0205】

また、各実施形態においては画像供給デバイスからプリンタへデータ転送を行なうことによって印字処理を実現する例について説明したが、本発明は画像データを転送して利用する機器であれば、どのような装置にも適用可能である。

【0206】

また、上述した第1実施形態においては、DUMMYコマンドをプリンタ内の空きバッファ数が1となった場合に転送するとして説明したが、画像供給デバイスからプリンタへ送られるコマンドであって、プリンタ内の空きバッファ数が得られ、プリンタでの処理負荷の少ないものであれば、どのようなコマンドを使用しても良い。

【0207】

また、各実施形態においては、BLOCK countに空きバッファ数をセットして通知する例について説明したが、空きバッファの数そのものでなく、「空きバッファあり」、「空きバッファ少ない」等のバッファ情報をセットする方法も考えられる。この方法でもバッファの状態を画像供給デバイスに通知することができるため、上述した各実施形態と同様の効果を得られる。

【0208】

また、上述した各実施形態においてはIEEE1934に規定されるシリアルインタフェースを用いてネットワークを構成する例を説明したが、本発明はこれに限定されるものではなく、Universal Serial Bus(USB)と呼ばれるシリアルインタフェースなど、任意のシリアルインタフェースを用いて構成されるネットワークにも適用することができる。

【他の実施形態】

なお、本発明は、複数の機器（例えばホストコンピュータ、インタフェース機器、リーダー、プリンタなど）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機、ファクシミリ装置など）に適用してもよい。

【0209】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはCPUやMPU）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記憶媒体から読出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。プログラムコードを供給するための記憶媒体としては、例えば、フロッピディスク、ハードディスク、光ディスク、光磁気ディスク、CD-ROM、CD-R、磁気テープ、不揮発性のメモリカード、ROMなどを用いることができる。

【0210】

また、コンピュータが読出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ

10

20

30

40

50

上で稼働しているOS（オペレーティングシステム）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0211】

さらに、記憶媒体から読出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0212】

10

【発明の効果】

以上説明したように、本発明によれば、1394シリアルバスなどによりホストデバイスとターゲットデバイスを接続し、ホストデバイスからターゲットデバイスへ送られるデータの転送について、コマンドとデータとで同じレジスタ領域を使用し、かつレジスタへのデータの書き込みに対する応答のみを行う際に、データ転送以外のコマンドを任意に実行可能なデータ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体を提供することができる。すなわち、本発明によれば、ホストデバイスがターゲットデバイスにおける空きバッファ数に応じてダミー処理を実行することにより、データ転送中であっても他のコマンドが任意に実行できる。

【0213】

20

また、レジスタへのデータの書き込みに対する応答時間を調整することにより、データバス上のトラフィックの効率化を実現するデータ転送装置、データ転送システムおよびその方法、画像処理装置、並びに、記録媒体を提供することができる。

【0214】

【図面の簡単な説明】

【図1】本発明を適用するシステムの一般的な構成例を示す図、

【図2】 1394シリアルバスによるネットワークの構成例を示す図、

【図3】 1394シリアルバスの構成例を示す図、

【図4】 1394シリアルバスにおけるアドレス空間の一例を示す図、

【図5】 1394シリアルバス用のケーブルの断面を示す図、

30

【図6】 1394シリアルバスで採用されている、データ転送方式のDS-Link方式を説明するための図、

【図7】バスリセット信号の発生から、ノードIDが決定し、データ転送が行えるようになるまでの一連のシーケンス例を示すフローチャート、

【図8】バスリセット信号の監視からルートノードの決定までの詳細例を示すフローチャート、

【図9】ノードID設定の詳細例を示すフローチャート、

【図10】 1394シリアルバスのネットワーク動作例を示す図、

【図11】 1394シリアルバスのCSRアーキテクチャの機能を示す図、

【図12】 1394シリアルバスに関するレジスタを示す図、

40

【図13】 1394シリアルバスのノード資源に関するレジスタを示す図、

【図14】 1394シリアルバスのコンフィギュレーションROMの最小形式を示す図、

【図15】 1394シリアルバスのコンフィギュレーションROMの一般形式を示す図、

【図16】バス使用权の要求を説明する図、

【図17】バス使用の許可を説明する図、

【図18】 1394シリアルバスにおけるアービトレーションの流れを示すフローチャート、

【図19】トランザクションレイヤにおけるCSRアーキテクチャに基づくリード、ライト、ロックの各コマンドの要求・レスポンスプロトコルを示す図、

【図20】リンクレイヤにおけるサービスを示す図、

50

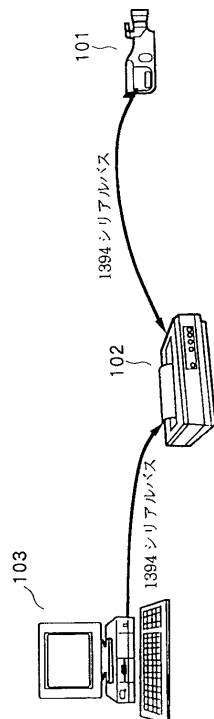
- 【図 2 1】非同期転送における時間的な遷移を示す図、
- 【図 2 2】非同期転送用パケットのフォーマットを示す図、
- 【図 2 3】スプリットランザクションの動作例を示す図、
- 【図 2 4】スプリットランザクションを行う場合の転送状態の時間的な遷移例を示す図、
- 【図 2 5】同期転送における時間的な遷移を示す図、
- 【図 2 6】同期転送用のパケットフォーマット例を示す図、
- 【図 2 7】 1394シリアルバスにおける同期転送のパケットフォーマットのフィールドの詳細を示す図、
- 【図 2 8】同期転送と非同期転送が混在するときの転送状態の時間的な遷移を示す図、
- 【図 2 9】 1394シリアルバスのレイヤ構造を示す図、 10
- 【図 3 0】データ転送用のレジスタ構成を示す図、
- 【図 3 1】画像供給デバイスからプリンタに対するコマンド制御の概要を示す図、
- 【図 3 2】プリンタから画像供給デバイスに対するリプライ制御の概要を示す図、
- 【図 3 3】プリンタにおけるブロックレジスタと内部バッファ構成を示す図、
- 【図 3 4】コントロール、レスポンスレジスタの一般的な構成を示す図、
- 【図 3 5】コマンド構成を示す図、
- 【図 3 6】 SEND、GETSTATUSコマンドの構成を示す図、
- 【図 3 7】 SEND、GETSTATUSリプライの構成を示す図、
- 【図 3 8】 GETSTATUSコマンドのステータスを示す図、
- 【図 3 9】 SENDコマンドにおける画像データ分割の様子を示す図、 20
- 【図 4 0】画像供給デバイスとプリンタの一般的な制御手順を示す図、
- 【図 4 1】 GETSTATUSコマンドの制御手順を示す図、
- 【図 4 2】コントロール、レスポンスレジスタの構成を示す図、
- 【図 4 3】 DUMMYコマンドの構成を示す図、
- 【図 4 4】 DUMMYコマンドの制御手順を示す図、
- 【図 4 5】 DUMMY、GETSTATUSコマンドの制御手順を示す図、
- 【図 4 6】 SENDコマンド前のDUMMYコマンドの制御手順を示す図、
- 【図 4 7】 DUMMYコマンドの一般的な制御手順を示す図、
- 【図 4 8】トラフィック効率化のためのDUMMYコマンドの制御手順を示す図、
- 【図 4 9】画像供給デバイスにおける画像転送処理を示すフローチャート、 30
- 【図 5 0】画像供給デバイスにおけるSENDコマンド処理を示すフローチャート、
- 【図 5 1】画像供給デバイスにおけるGETSTATUSコマンド処理を示すフローチャート、
- 【図 5 2】画像供給デバイスにおけるタイマ割り込み処理を示すフローチャート、
- 【図 5 3】プリンタにおけるSENDコマンド処理を示すフローチャート、
- 【図 5 4】プリンタにおける印字処理を示すフローチャート、
- 【図 5 5】プリンタにおけるGETSTATUSコマンド処理を示すフローチャート、
- 【図 5 6】第2実施例におけるコントロール、レスポンスレジスタの構成を示す図、
- 【図 5 7】第2実施例におけるRETRY、GETSTATUSコマンドの制御手順を示す図、
- 【図 5 8】第2実施例の画像供給デバイスにおける画像転送処理を示すフローチャート、
- 【図 5 9】第2実施例の画像供給デバイスにおけるSENDコマンド処理を示すフローチャート、 40
- ト、
- 【図 6 0】第2実施例のプリンタにおけるSENDコマンド処理を示すフローチャートである。
- 【図 6 1】本発明に係る第3実施形態におけるコントロールレジスタ、レスポンスレジスタの構成を示す図である。
- 【図 6 2】第3実施形態におけるDUMMY処理の制御手順を示す図である。
- 【図 6 3】第3実施形態におけるDUMMY処理、GETSTATUSコマンドの制御手順を示す図である。
- 【図 6 4】第3実施形態におけるSENDコマンド前のDUMMY処理の制御手順を示す図である。
- 【図 6 5】第3実施形態の画像供給デバイスにおける画像転送処理を示すフローチャート 50

である。

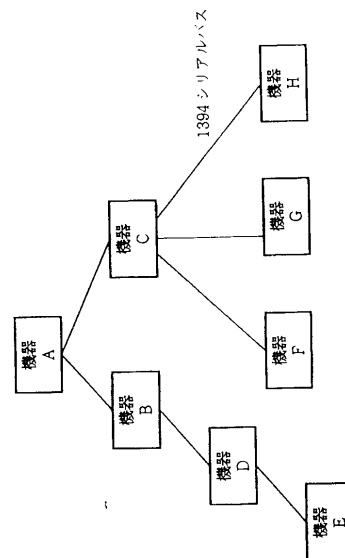
【図 6 6】第3実施形態の画像供給デバイスにおけるSENDコマンド処理を示すフローチャートである。

【図 6 7】第3実施形態のプリンタにおけるSEND処理を示すフローチャートである。

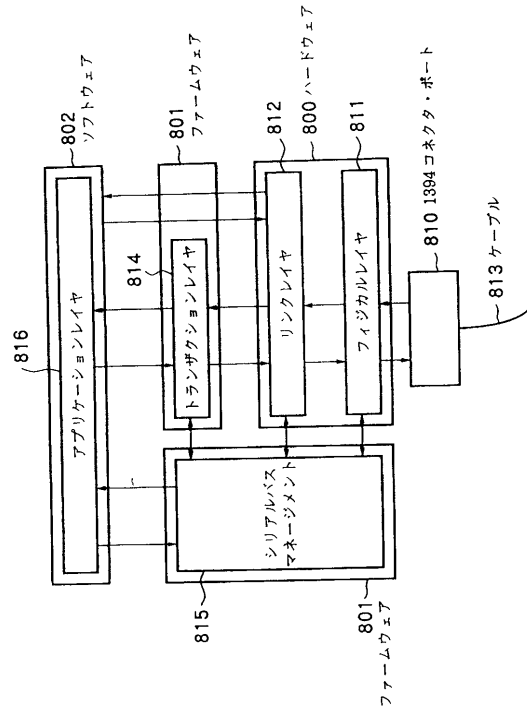
【図 1】



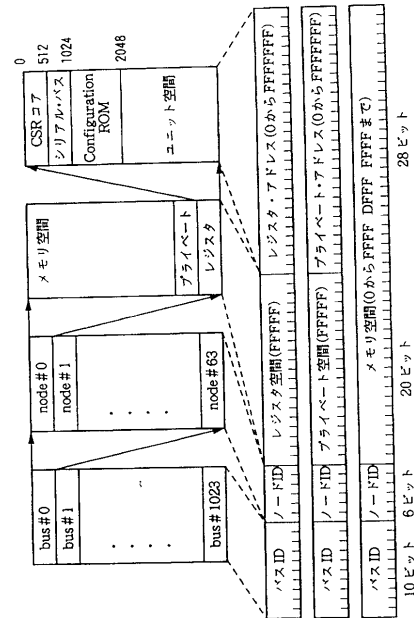
【図 2】



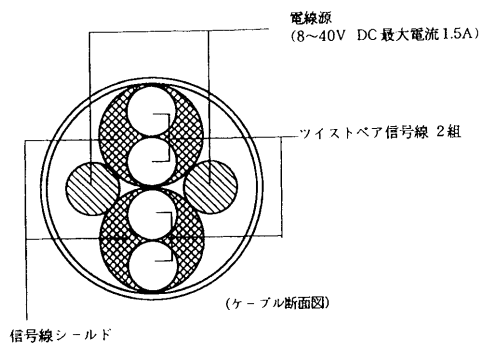
【図 3】



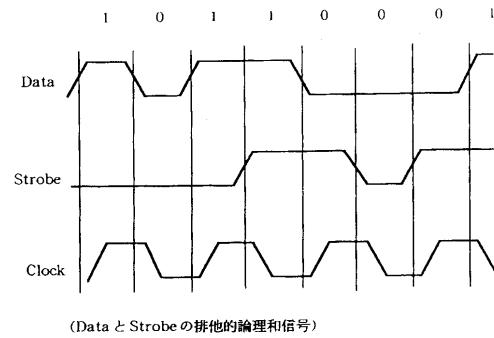
【図 4】



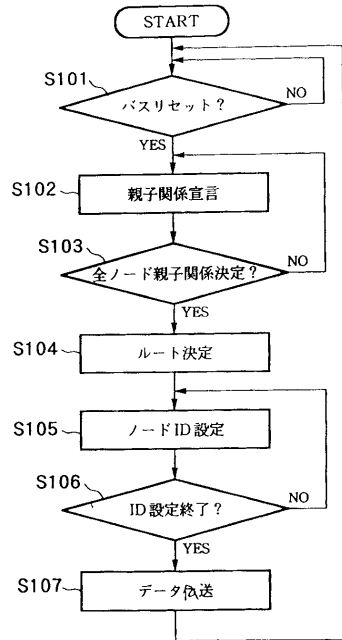
【図 5】



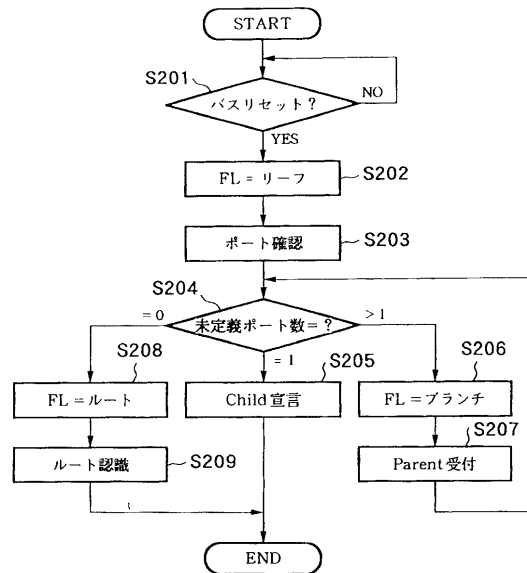
【図 6】



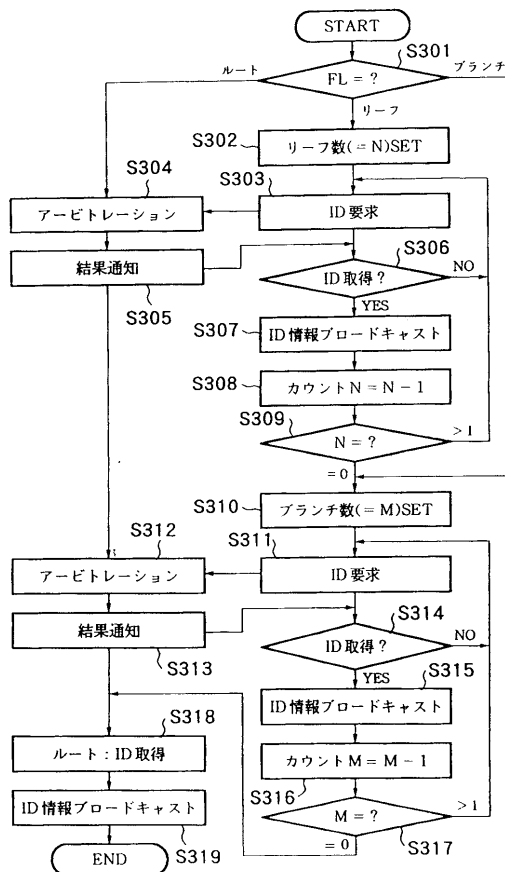
【図 7】



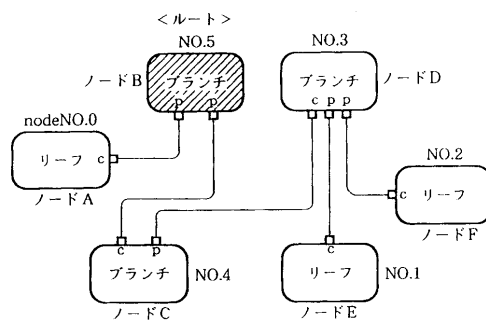
【図 8】



【図 9】



【図 10】



ブランチ: 2つ以上のノード接続があるノード
 リーフ: 1つのポートのみ接続があるノード

□: ポート
 c: 子のノードに相当するポート
 p: 親のノードに相当するポート

【図 1 1】

CSR コア・レジスタ

オフセット (16進数)	レジスタ名称	機 能
000	STATE_CLEAR	状態と制御の情報
004	STATE_SET	STATE_CLEARの書き込み可否を示す情報
008	NODE_IDS	バスID + ノードID
00C	RESET_START	この領域に対する書き込みでバスをリセット
010~014	INDIRECT_ADDRESS, INDIRECT_DATA	1Kより大きいROMをアクセスするための レジスタ
018~01C	SPLIT_TIMEOUT	スプリット・トランザクションのタイムアウトを 検出するタイマの値
020~02C	ARGUMENT, TEST_START, TEST_STATUS	診断用のレジスタ
030~04C	UNITS_BASE, UNITS_BOUND, MEMORY_BASE, MEMORY_BOUND	IEEE1394では、実施しない
050~054	INTERRUPT_TARGET, INTERRUPT_MASK	割り込み通知レジスタ
058~07C	CLOCK_VALUE, CLOCK_TICK_PERIOD, CLOCK_STOROB, ARRIVED, CLOCK_INFO	IEEE1394では、実施しない
080~0FC	MESSAGE_REQUEST, MESSAGE_RESPONSE	メッセージ通知レジスタ
100~17C		予約
180~1FC	ERROR_LOG_BUFFER	IEEE1394用に予約

【図 1 2】

シリアル・バス・レジスタ

オフセット (16進数)	レジスタ名称	機 能
200	CYCLE_TIME	アイソクロナス転送のためのカウンタ
204	BUS_TIME	時間を同期するためのレジスタ
208	POWER_FAIL_IMMINENT	電源供給に関するレジスタ
20C	POWER_SOURCE	
210	BUSY_TIMEOUT	トランザクション層の再試行を制御
214 218		予約
21C	BUS_MANAGER_ID	バス・マネージャのノードID
220	BANDWIDTH_AVAILABLE	アイソクロナス転送の帯域を管理
224 228	CHANNELS_AVAILABLE	アイソクロナス転送のチャネル番号を 管理
22C	MAINT_CONTROL	
230	MAINT_UTILITY	診断用レジスタ
234 3FC		予約

【図 1 3】

シリアル・バス装置レジスタ

オフセット (16進数)	レジスタ名称	機 能
800 FFC		予約
1000 13FC	TOPOLOGY_MAP	シリアル・バスの構成情報
1400 1FFC		予約
2000 2FFC	SPEED_MAP	シリアル・バスの転送速度の 情報
3000 FFFC		予約

【図 1 4】

最小形式の Configuration ROM

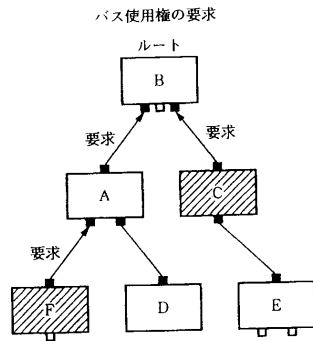
01	ベンダID
----	-------

【図 1 5】

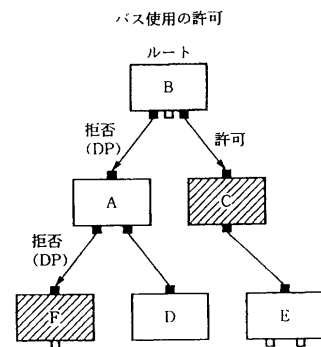
一般形式の Configuration ROM

bus_info_blockの長さ	ROMの長さ	CRC
bus_info_block (1394のASCIIコードと、ノードがアイソクロナス資源管理、 サイクル・マスタ、バス・マネージャの能力をもっているかの情報)		
root_directory (ベンダIDとノードの機能を表す)		
unit_directories (ユニットの種類とドライバ・ソフトのバージョン)		
root & unit leaves		
vender_dependent_information		

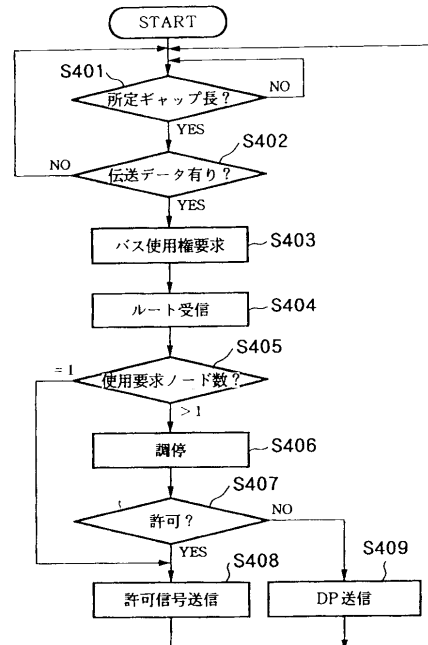
【図 16】



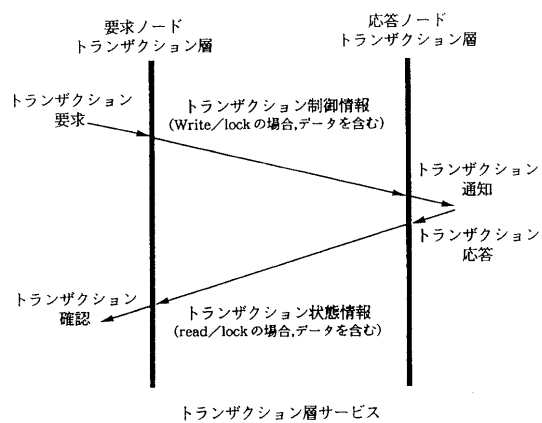
【図 17】



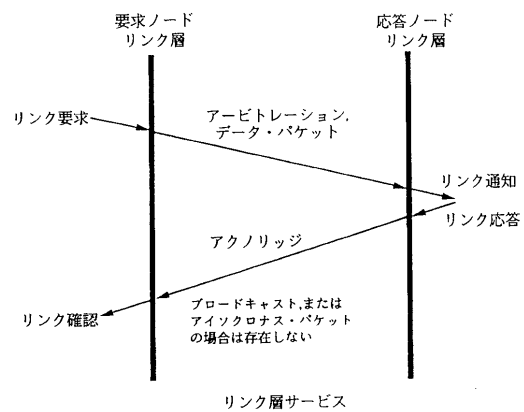
【図 18】



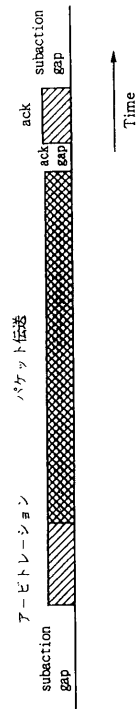
【図 19】



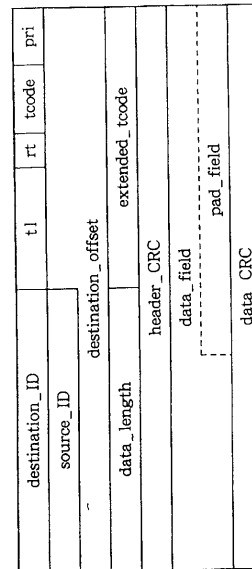
【図 20】



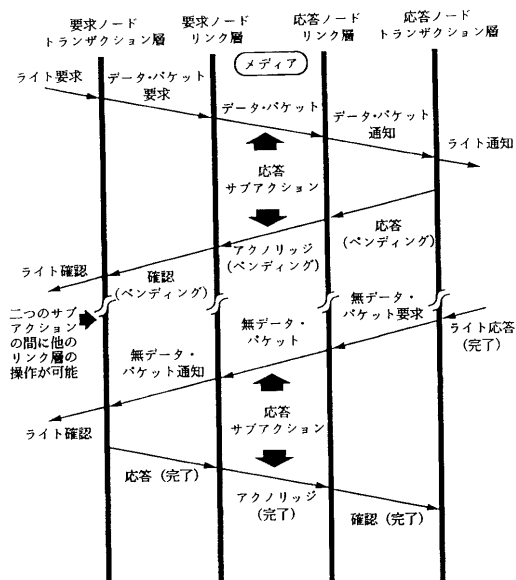
【図 2 1】



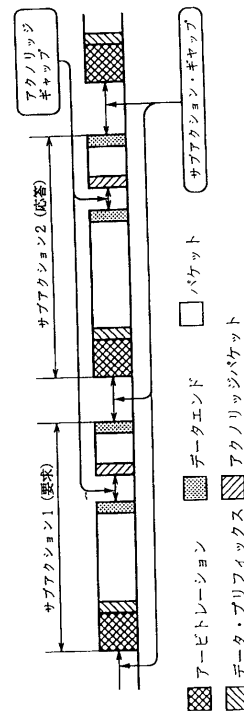
【図 2 2】



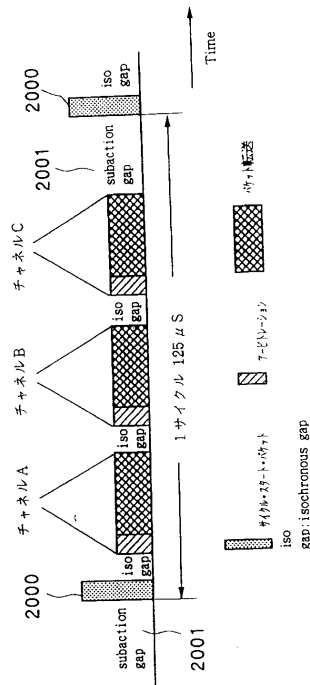
【図 2 3】



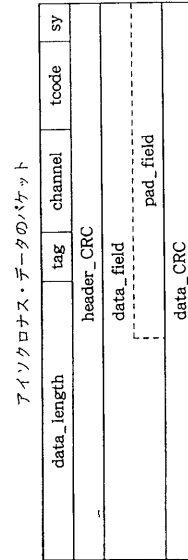
【図 2 4】



【図 25】



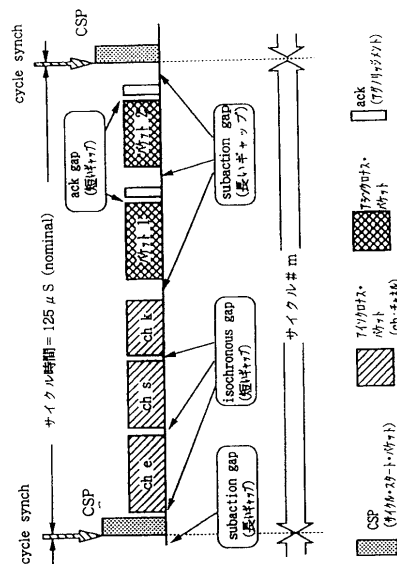
【図 26】



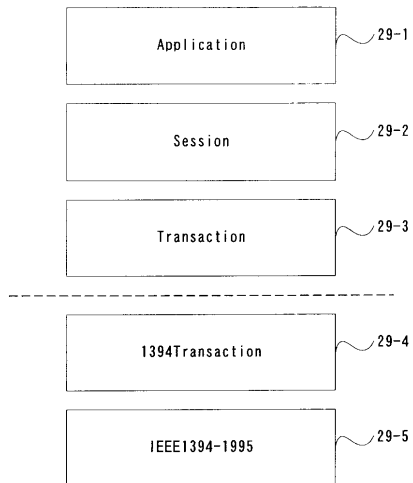
【図 27】

略称	名称	内容
destination_ID	destination identifier	着信先ノードのIDを示す (アシンクロナスのみ)
tl	transaction label	一連のトランザクションを示すためのラベル (アシンクロナスのみ)
rt	retry code	再送ステータスを示すコード (アシンクロナスのみ)
tcode	transaction code	パケットの種類を示すコード (アシンクロナスのみ)
pri	priority	優先順位 (アシンクロナスのみ)
source_ID	source identifier	発信元ノード (アシンクロナスのみ)
destination_offset	destination memory address	着信先ノードのメモリ・アドレス (アシンクロナスのみ)
rcode	response code	応答ステータス (アシンクロナスのみ)
quadiet_data	quadiet(4bytes) data	4バイト長のデータ (アシンクロナスのみ)
data_length	length of data	data_fieldの長さ (pad bytesは除く)
extended_tcode	extended transaction code	拡張トランザクション・コード (アシンクロナスのみ)
chanel	isochronous identifier	アイソクロナス・パケットの識別を行う
sy	synchronization code	映像・音声などの同期に使われる (アシンクロナスのみ)
cycle_time_data	contents of the CYCLE_TIME register	サイクル・マスタ・ノードのサイクル・タイム・レジスタの値 (サイクル・パケットのみ)
data_field	data + pad bytes	データが格納される (アイソクロナスとアシンクロナス)
header_CRC	CRC for header field	ヘッダ部分に対するCRC
data_CRC	CRC for data field	データ部分に対するCRC
tag	tag label	アイソクロナス・パケットのフォーマット

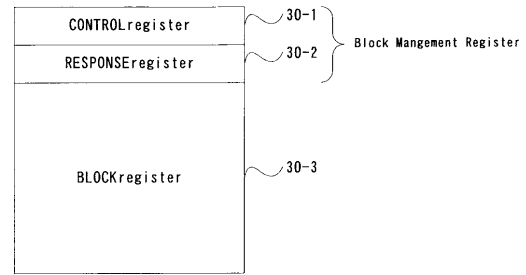
【図 28】



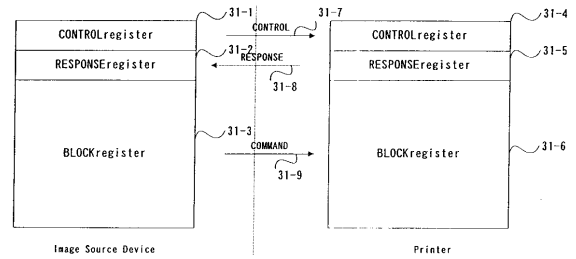
【図 29】



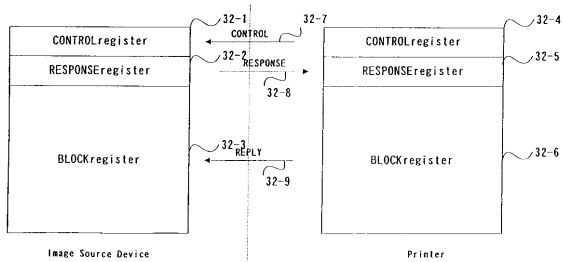
【図 30】



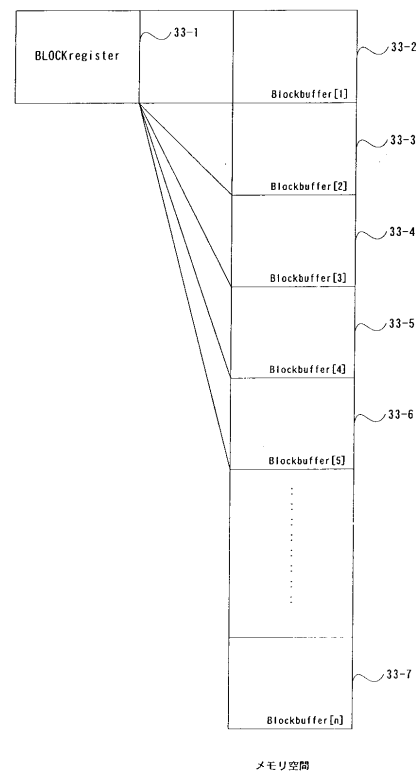
【図 31】



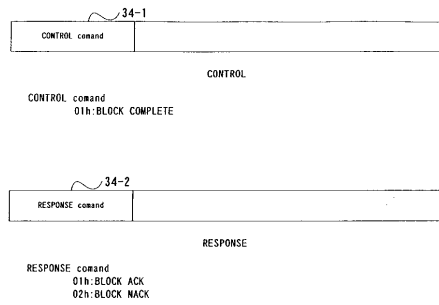
【図 32】



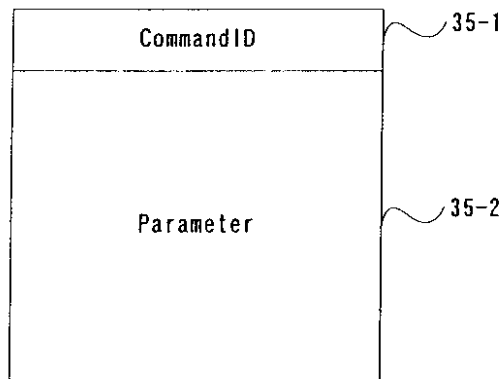
【図 33】



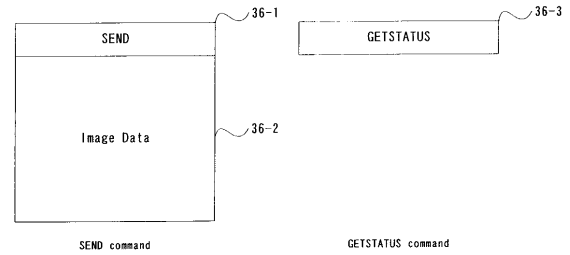
【図 34】



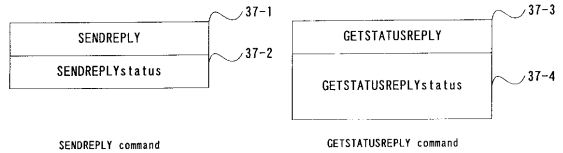
【図 35】



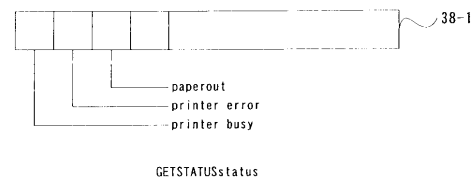
【図 36】



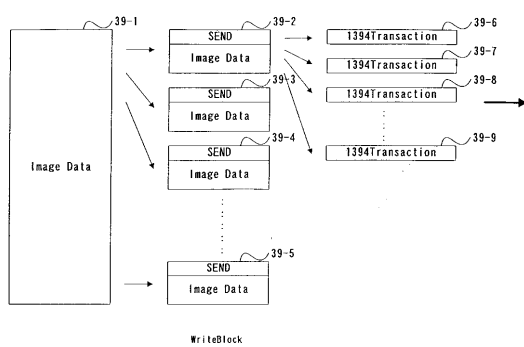
【図 37】



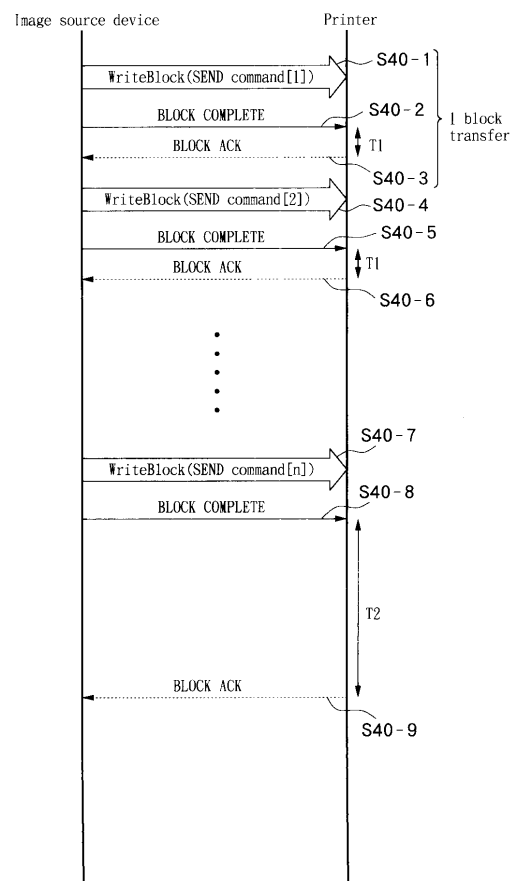
【図 38】



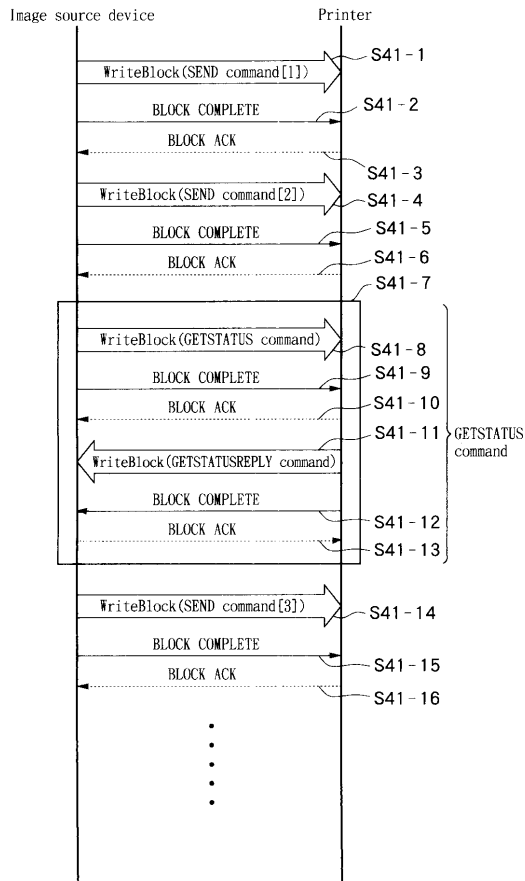
【図 39】



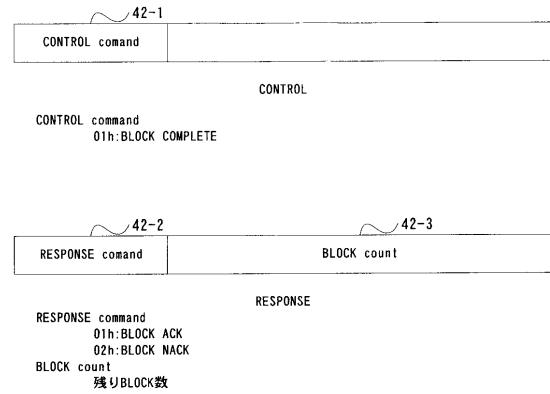
【図 40】



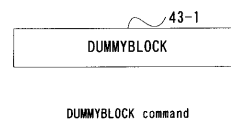
【図 4 1】



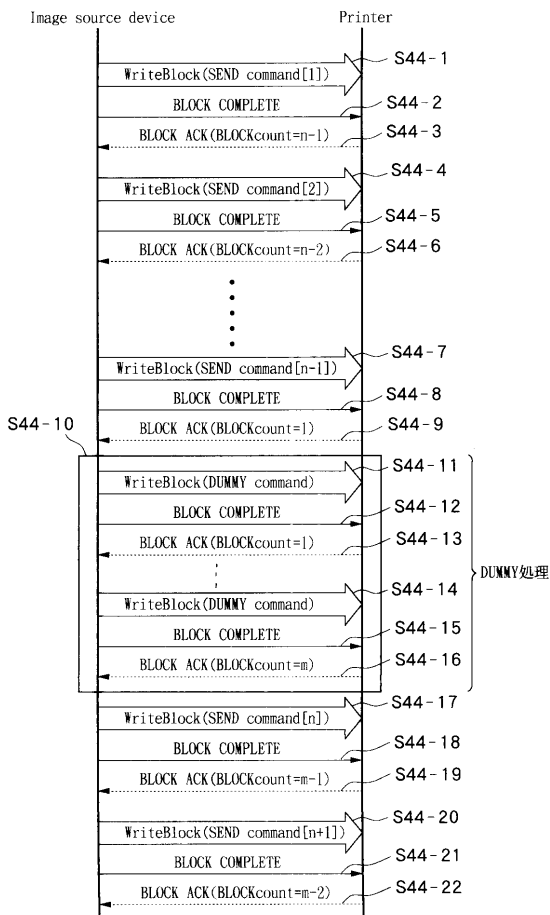
【図 4 2】



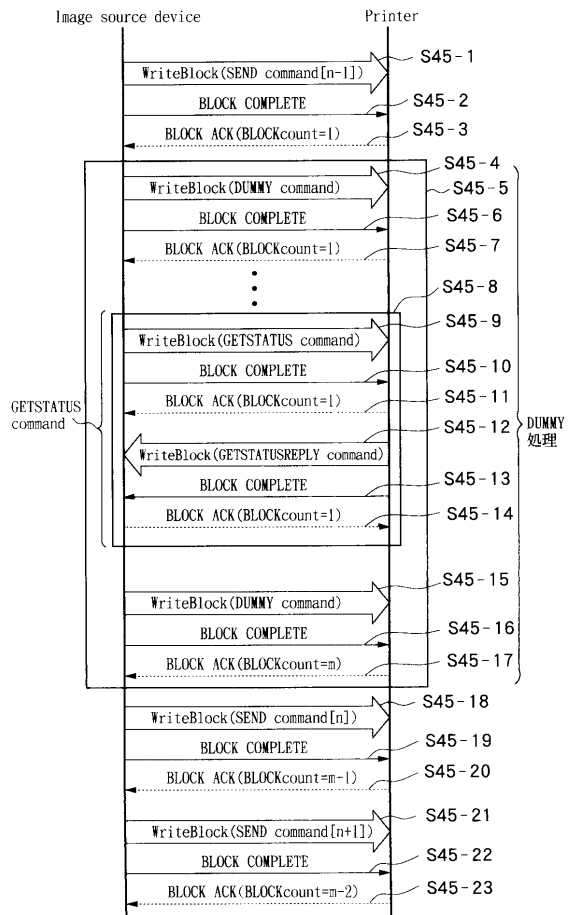
【図 4 3】



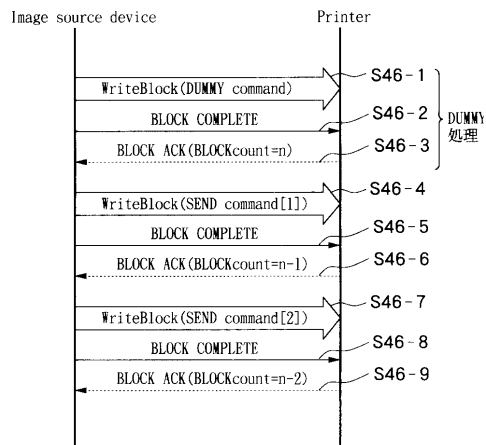
【図 4 4】



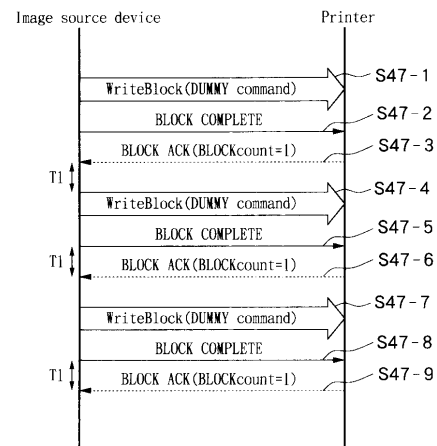
【図 4 5】



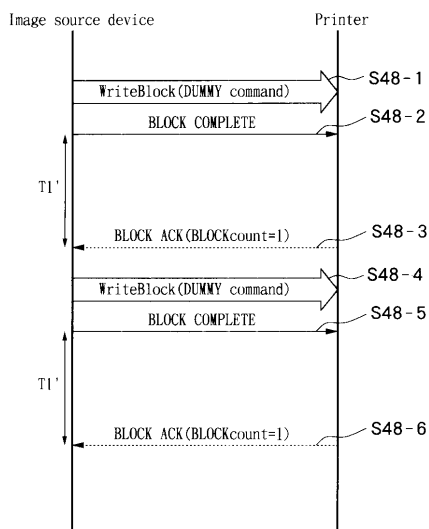
【図 4 6】



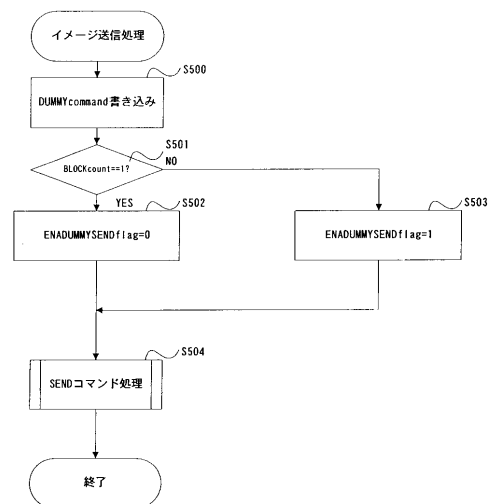
【図 4 7】



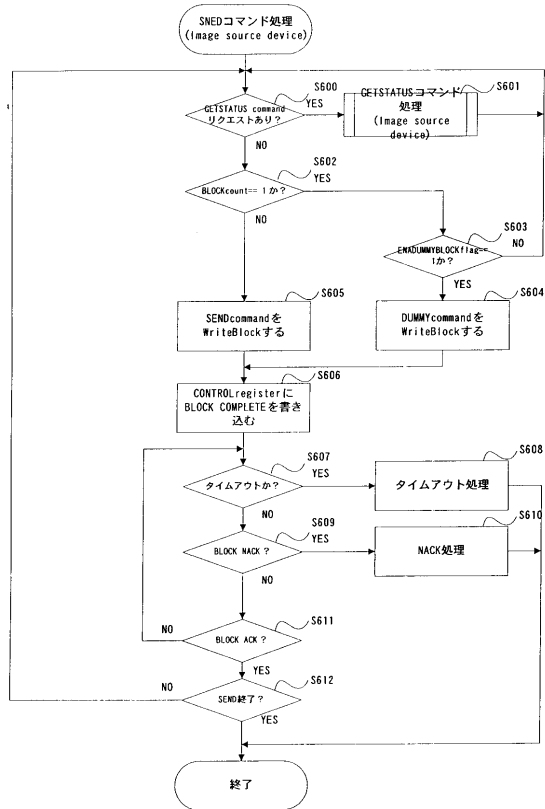
【図 4 8】



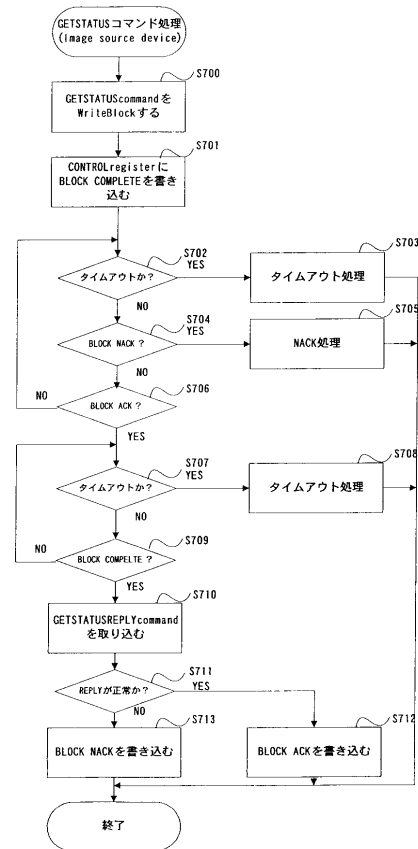
【図 4 9】



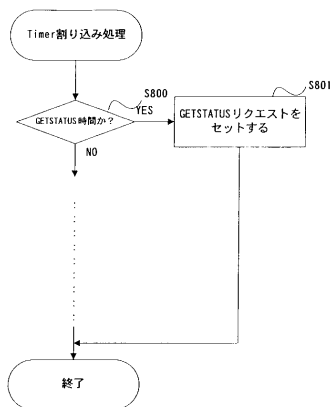
【図 50】



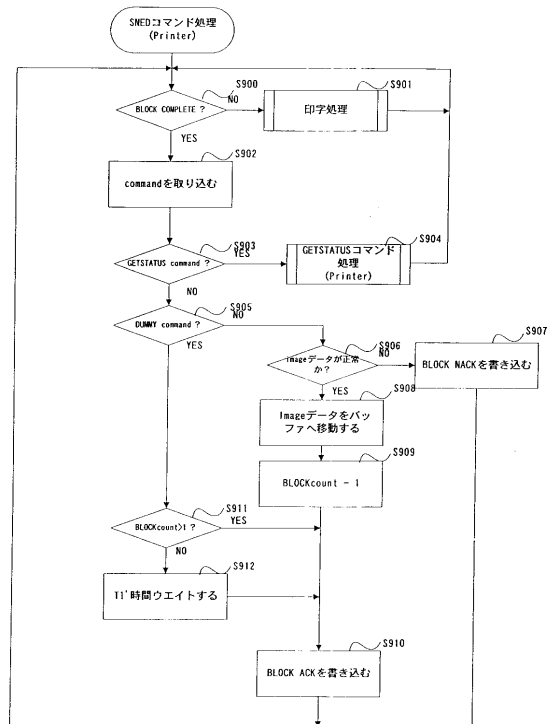
【図 51】



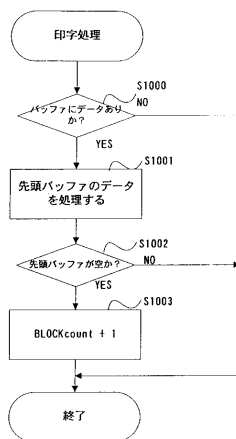
【図 52】



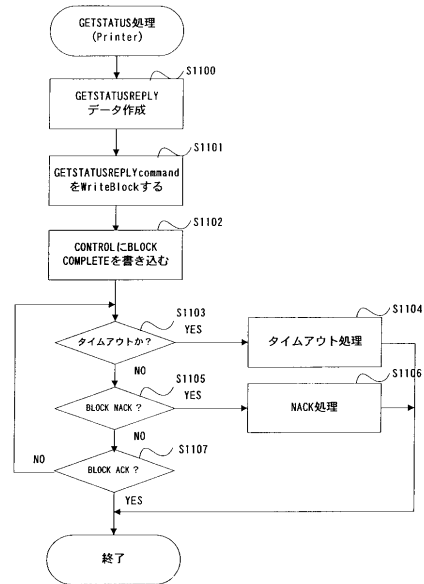
【図 53】



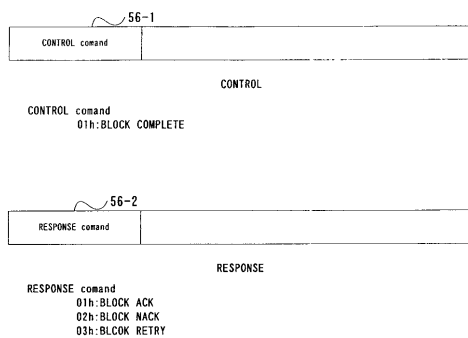
【図 54】



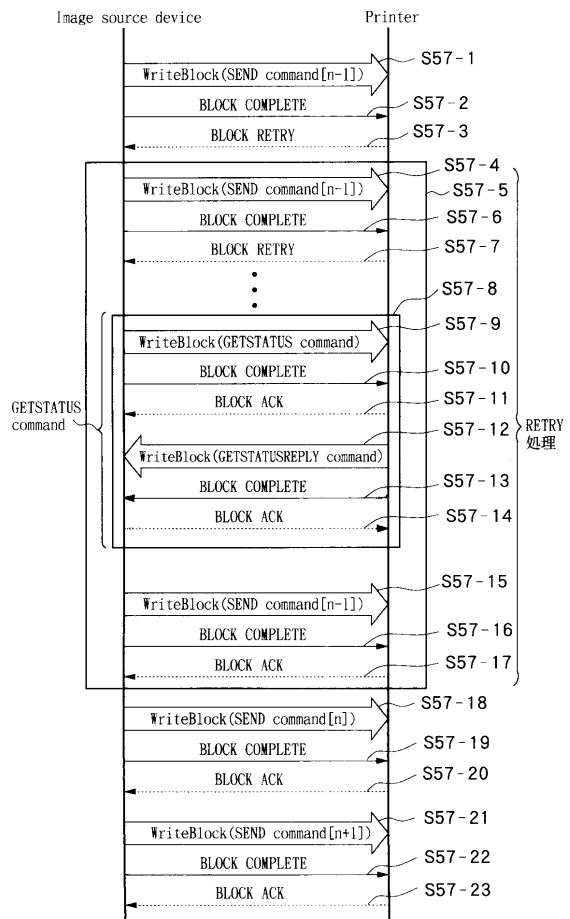
【図 55】



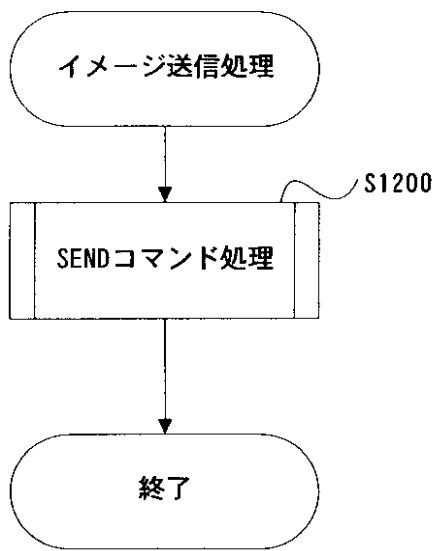
【図 56】



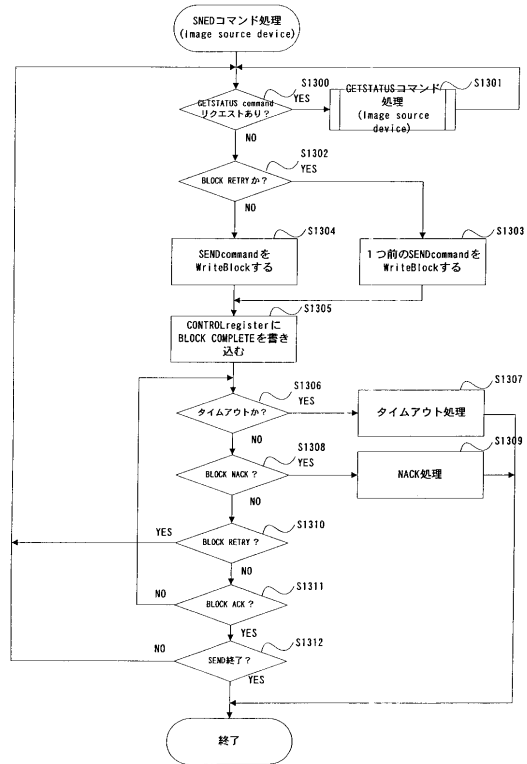
【図 57】



【図 58】

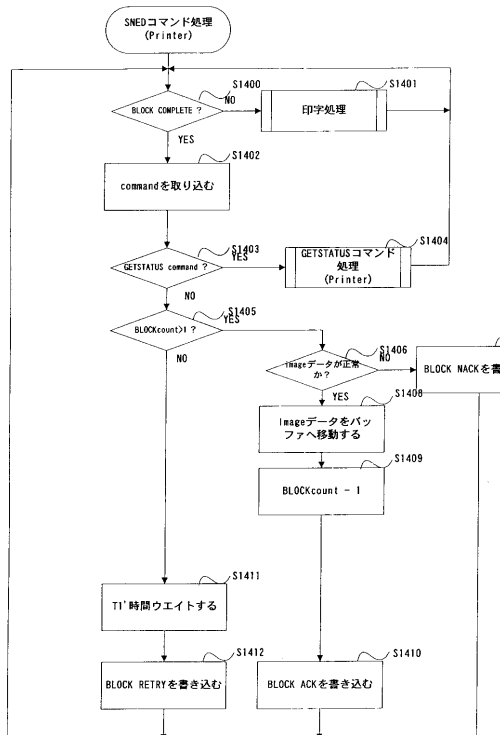


【図 59】

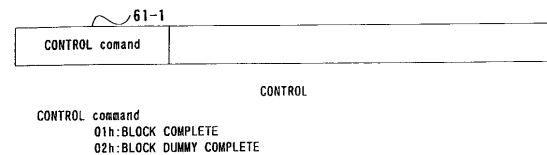


【図 60】

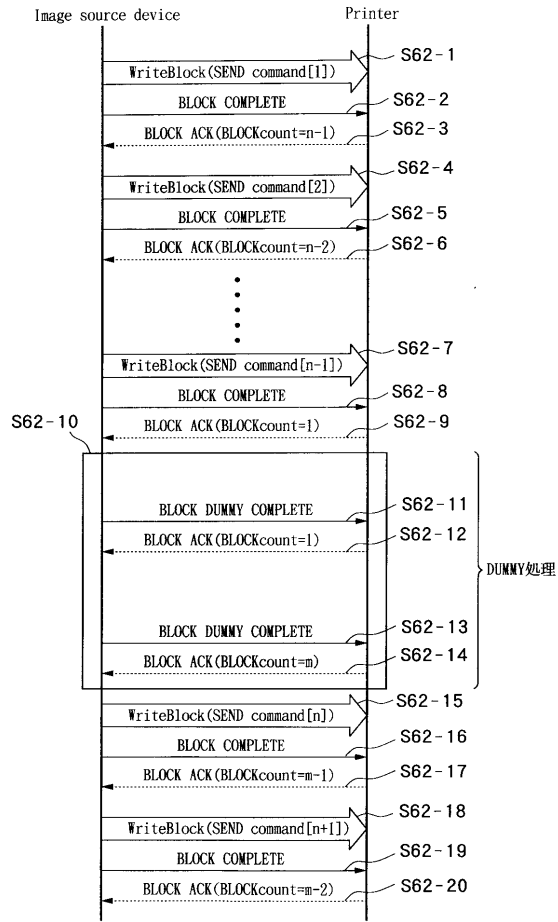
図 60



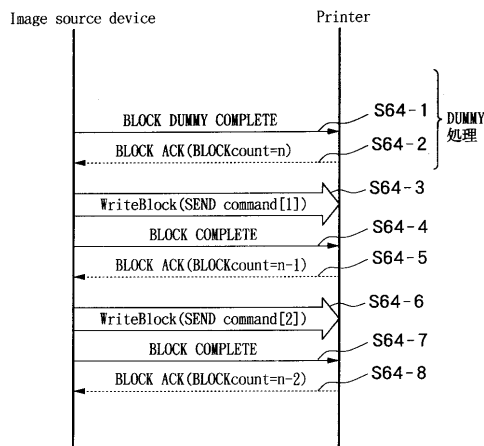
【図 61】



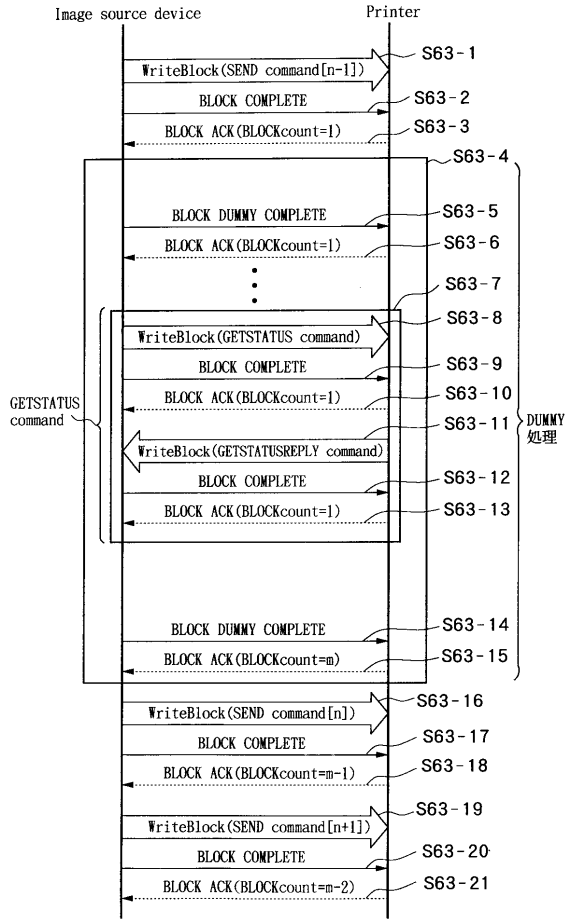
【図 6 2】



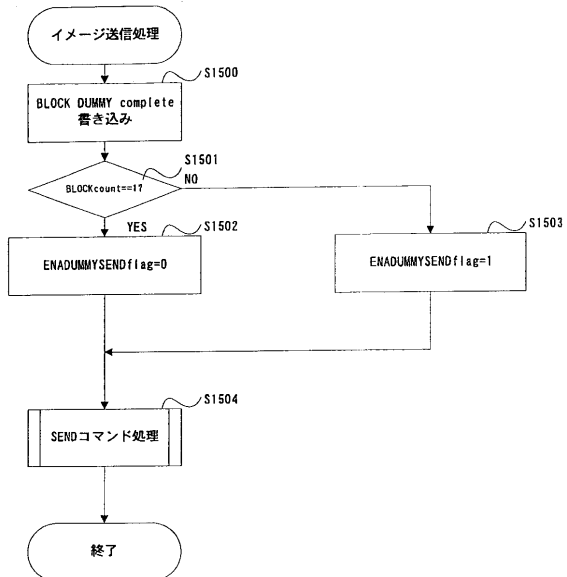
【図 6 4】



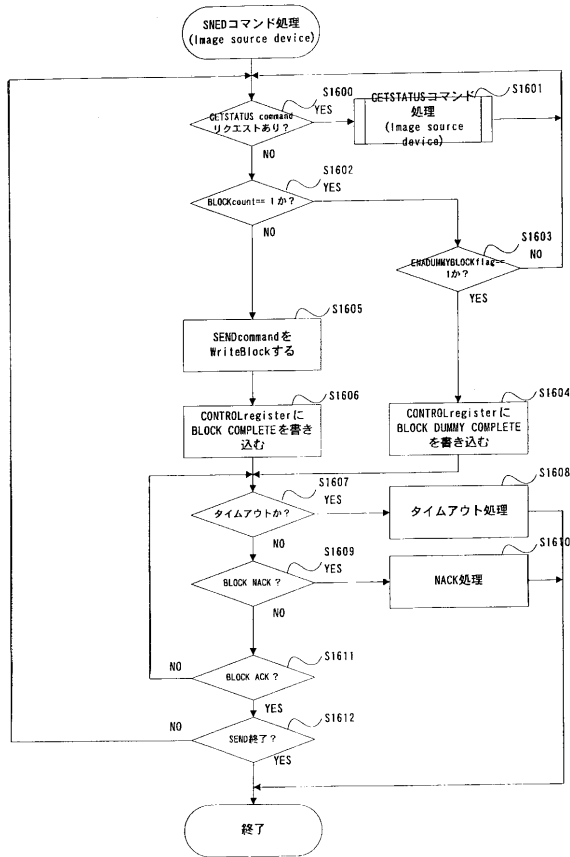
【図 6 3】



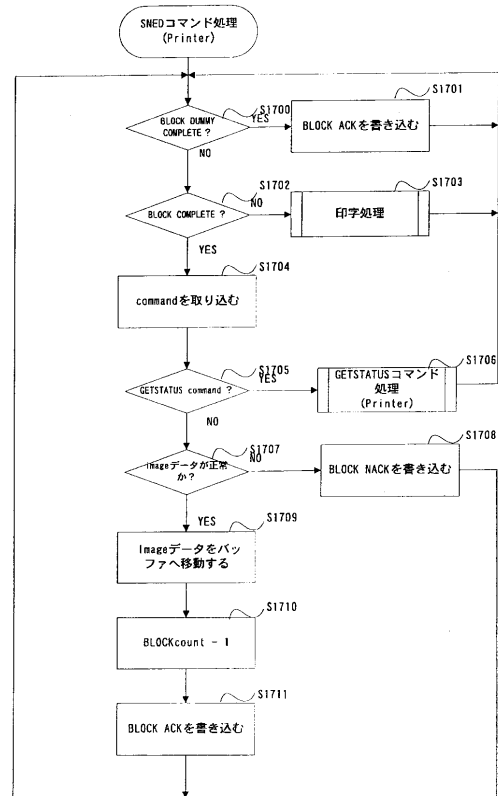
【図 6 5】



【図 66】



【図 67】



フロントページの続き

(51) Int.Cl. F I
G 0 6 F 13/42 (2006.01) G 0 6 F 13/42 3 1 0

(72)発明者 福長 耕司
東京都大田区下丸子3丁目30番2号 キヤノン株式会社内

(72)発明者 小林 真琴
東京都大田区下丸子3丁目30番2号 キヤノン株式会社内

審査官 羽岡 さやか

(56)参考文献 特開昭60-157354(JP,A)
特開平05-276344(JP,A)
特開昭58-034650(JP,A)
特開平05-022369(JP,A)
特開平04-127255(JP,A)

(58)調査した分野(Int.Cl., DB名)

H04L 12/28-12/417

H04L 12/44-12/46

H04L 13/00-13/18

G06F 13/00

G06F 13/38-13/42