お客様各位

資料中の「日立製作所」、「日立XX」等名称の株式会社ルネサス テクノロジへの変更について

2003年4月1日を以って三菱電機株式会社及び株式会社日立製作所のマイコン、ロジック、アナログ、ディスクリート半導体、及びDRAMを除くメモリ(フラッシュメモリ・SRAM等)を含む半導体事業は株式会社ルネサス テクノロジに承継されました。従いまして、本資料中には・「日立製作所」、「株式会社日立製作所」、「日立半導体」、「日立XX」といった表記が残っておりますが、これらの表記は全て「株式会社ルネサス テクノロジ」に変更されておりますのでご理解の程お願い致します。尚、会社商標・ロゴ・コーポレートステートメント以外の内容については一切変更しておりませんので資料としての内容更新ではありません。

ルネサステクノロジ ホームページ (http://www.renesas.com)

2003年4月1日
株式会社ルネサス テクノロジ
カスタマサポート部

ご注意

安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。

ADJ-502-091

日立シングルチップマイクロコンピュータ

H8S/2215　USB ファンクションモジュール Mass Storage Class（Bulk-Only Transport）アプリケーションノート

H8S/2215	HD64F2215

H8S/2215　USB ファンクションモジュール Mass Storage Class（Bulk-Only Transport）アプリケーションノート	
発行年月日	2002 年 3 月　第 1 版
発行	株式会社　日立製作所 半導体グループビジネス企画本部
編集	株式会社日立小平セミコン 技術ドキュメントグループ
©株式会社　日立製作所	2002

ご注意

1　本書に記載の製品及び技術のうち「外国為替及び外国貿易法」に基づき安全保障貿易管理関連貨物・技術に該当するものを輸出する場合，または国外に持ち出す場合は日本国政府の許可が必要です。

2　本書に記載された情報の使用に際して，弊社もしくは第三者の特許権，著作権，商標権，その他の知的所有権等の権利に対する保証または実施権の許諾を行うものではありません。また本書に記載された情報を使用した事により第三者の知的所有権等の権利に関わる問題が生じた場合，弊社はその責を負いませんので予めご了承ください。

3　製品及び製品仕様は予告無く変更する場合がありますので，最終的な設計，ご購入，ご使用に際しましては，事前に最新の製品規格または仕様書をお求めになりご確認ください。

4　弊社は品質・信頼性の向上に努めておりますが，宇宙，航空，原子力，燃焼制御，運輸，交通，各種安全装置，ライフサポート関連の医療機器等のように，特別な品質・信頼性が要求され，その故障や誤動作が直接人命を脅かしたり，人体に危害を及ぼす恐れのある用途にご使用をお考えのお客様は，事前に弊社営業担当迄ご相談をお願い致します。

5　設計に際しては，特に最大定格，動作電源電圧範囲，放熱特性，実装条件及びその他諸条件につきましては，弊社保証範囲内でご使用いただきますようお願い致します。
保証値を越えてご使用された場合の故障及び事故につきましては，弊社はその責を負いません。また保証値内のご使用であっても半導体製品について通常予測される故障発生率，故障モードをご考慮の上，弊社製品の動作が原因でご使用機器が人身事故，火災事故，その他の拡大損害を生じないようにフェールセーフ等のシステム上の対策を講じて頂きますようお願い致します。

6　本製品は耐放射線設計をしておりません。

7　本書の一部または全部を弊社の文書による承認なしに転載または複製することを堅くお断り致します。

8　本書をはじめ弊社半導体についてのお問い合わせ，ご相談は弊社営業担当迄お願い致します。

はじめに

本アプリケーションノートは、H8S/2215 内蔵の USB ファンクションモジュールを用いた Mass Storage Class（Bulk-Only Transport）ファームウエアについて説明したものであり、お客様が USB ファンクションモジュールファームウエア作成の際に、御参考として役立てて頂けるようにまとめました。

本アプリケーションノートの内容およびソフトウェアは、USB ファンクションモジュールの使用例として説明しているものであり、その内容を保証するものではありません。

また、開発に際しましては、本書のほかに以下の関連マニュアルもあわせて御覧ください。

【関連マニュアル】

- Universal Serial Bus Specification Revision 1.1
- Universal Serial Bus Mass Storage Class Specification Overview Revision 1.1
- Universal Serial Bus Mass Storage Class（Bulk-Only Transport）Revision 1.0
- H8S/2215ハードウエアマニュアル
- H8S/2215 Solution Engine CPUボード（MS2215CP01-C/S）取扱説明書
- H8S/2215シリーズTFP120用ユーザシステムインタフェースケーブル（HS2215ECN61H）取扱説明書
- E6000（HS2214EPI61H）エミュレータユーザーズマニュアル

【ご注意】　本アプリケーションノートに記載してあるサンプルプログラムでは、USB の転送タイプのうち「インタラプト」に関するファームウエアは準備しておりません。「インタラプト」「アイソクロナス」（H8S/2215 ハードウエアマニュアル 15-1 参照）の転送タイプを御使用になる場合は、別途お客様でプログラムを作成していただく必要があります。

また、本アプリケーションノートには、上記システムの開発時に必要と思われる H8S/2215、H8S/2215 Solution Engine のハードウエア仕様を記載してありますが、詳細は H8S/2215 のハードウエアマニュアル、ならびに H8S/2215 Solution Engine の取扱説明書を御覧ください。

目次

1. 概要

本アプリケーションノートは、H8S/2215 の USB ファンクションモジュールの使用方法、およびファームウエアの作成例について説明したものです。

H8S/2215 内蔵 USB ファンクションモジュールの特長を以下に示します。

- USB1.1に準拠したUDC（USB Device Controller）を内蔵
- USBプロトコルを自動処理
- エンドポイント0に対するUSB標準コマンドを自動処理（一部コマンドはファームウエアで処理する必要があります。）
- フルスピード（12Mbps）転送に対応
- USB送受信に必要な各種割り込み信号を生成
- クロック発振器内のUSBクロック選択回路により、USB動作クロックを内部システムクロック（16MHz）3逓倍／外部入力（48MHz）を選択可能
- バストランシーバを内蔵
- 任意のエンドポイント構成が設定可能

エンドポイントの構成

エンドポイント名	名称	転送タイプ	最大パケットサイズ	FIFO バッファ容量	DMA 転送
エンドポイント 0	EP0s	セットアップ	8 Byte	8 Byte	―
	EP0i	コントロールイン	64 Byte	64 Byte	―
	EP0o	コントロールアウト	64 Byte	64 Byte	―
エンドポイント任意	EPn	インタラプト(イン)	64 Byte	64 Byte（可変）	―
エンドポイント任意	EPn	バルクイン	64 Byte	64×2（128 Byte）	可能
エンドポイント任意	EPn	バルクアウト	64 Byte	64×2（128 Byte）	可能
エンドポイント任意	EPn	アイソクロナス(イン)	128 Byte	128×2（可変）	―
エンドポイント任意	EPn	アイソクロナス(アウト)	128 Byte	128×2（可変）	―
エンドポイント任意	EPn	バルクイン	64 Byte	64×2（128 Byte）	可能
エンドポイント任意	EPn	バルクアウト	64 Byte	64×2（128 Byte）	可能
エンドポイント任意	EPn	インタラプト(イン)	64 Byte	64 Byte（可変）	―

システム構成例を**図** 1.1 に示します。

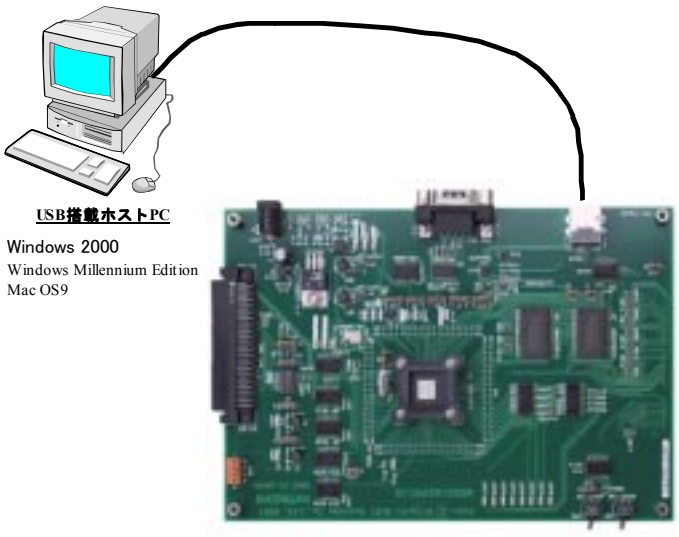


図1.1 システム構成例

本システムは、H8S/2215 を搭載した日立超 LSI システムズ社製の H8S/2215 Solution Engine（以下 MS2215CP）、「Windows 2000/Windows Millennium Edition」または「Mac OS9」OS を搭載した PC によって構成されています。

本システムは、ホスト PC と MS2215CP を USB で接続し、MS2215CP 上の SRAM を RAM Disk として動作させる事により、ホスト PC から MS2215CP の SRAM 上へデータの保存、読み出しができます。

また、上記 OS に標準で付属している USB Mass Storage Class（Bulk-Only Transport）のデバイスドライバを使用することが可能です。

本システムの特長を以下に示します。

1. サンプルプログラムにより、H8S/2215のUSBモジュールを短期間で評価可能
2. サンプルプログラムは、USBのコントロール転送、バルク転送をサポート
3. E6000を使用することができ、効率的なデバッグが可能
4. プログラムを追加作成することにより、インタラプト、アイソクロナス転送についても対応可能*

【注】* インタラプト、アイソクロナス転送のプログラムは、お客様で作成していただく必要があります。

2. USB Mass Storage Class（Bulk-Only Transport）の概要

この章では、USB Mass Storage Class（Bulk-Only Transport）について説明します。

USB のストレージ関連システムを開発する際に、ご参考として御使用ください。なお、規格の詳細については、

「Universal Serial Bus Mass Storage Class Specification Overview Revision 1.1」

「Universal Serial Bus Mass Storage Class Bulk-Only Transport Revision 1.0」

をご覧下さい。

2.1 USB Mass Storage Class について

USB Mass Storage Class とは、大規模記憶装置（ストレージ）をホスト PC に接続しデータの書き込み、読み出し等の動作を行う機器に適合するよう規格化されたクラスです。

ホスト PC に、このクラスのファンクションである事を伝えるためには、Interface Descriptor の bInterfaceClass フィールドに値 0x08 を記述する事が必要です。

ホスト PC とファンクション間でデータ転送をする場合、USB に規定されている 4 つの転送方法（コントロール転送、バルク転送、インタラプト転送、アイソクロナス転送）を用いてデータの転送を行います。どの転送方法をどのように使用するかは、プロトコルコードとして定められています。

データ転送プロトコルとして次の 2 種類があります。

- USB Mass Storage Class Bulk-Only Transport
- USB Mass Storage Class Control/Bulk/Interrupt（CBI）Transport

USB Mass Storage Class Bulk-Only Transport は名前の示すとおり、バルク転送のみ使用したデータ転送プロトコルです。

USB Mass Storage Class Control/Bulk/Interrupt（CBI）Transport は、コントロール転送、バルク転送、インタラプト転送を使用したデータ転送プロトコルです。CBI Transport は、更にインタラプト転送を使用するデータ転送プロトコル、使用しないデータ転送プロトコルの 2 種類に分かれています。

本サンプルプログラムでは、USB Mass Storage Class Bulk-Only Transport をデータ転送プロトコルとして使用します。

ホスト PC がデータのロードやセーブをするために機器を使用する場合、ホスト PC からファンクションに対して命令（コマンド）を与えます。ファンクションは送られたコマンドを実行することによりデータのロードやセーブが行えます。ホスト PC からファンクションに対して送られるコマンドはサブクラスコードとして定められています。

2.2 サブクラスコードについて

サブクラスコードとは、ホスト PC からコマンドトランスポートでファンクションに送られるコマンドフォーマットを表す値です。コマンドフォーマットの種類としては 7 種類あり、**表** 2.1 に示すサブクラスコードが定められています。

表 2.1

サブクラスコード	コマンドの規格
0x01	Reduced Block Commands（RBC）、T10/1240-D
0x02	Attachment Packet Interface（ATAPI）for CD-ROMs. SFF-8020i、 Multi-Media Command Set 2（MMC-2）
0x03	Attachment Packet Interface（ATAPI）for Tape. QIC-157
0x04	USB Mass Storage Class UFI Command Specification
0x05	Attachment Packet Interface（ATAPI）for Floppies. SFF-8070i
0x06	SCSI Primary Commands –2（SPC-2）、Revision 3 or later

ホスト PC に、機器が対応しているコマンドフォーマットを伝えるためには、Interface Descriptor の bInterfaceSubClass フィールドにサブクラスコード値を記述することが必要です。

本サンプルプログラムでは、サブクラスコード値 0x06 の SCSI Primary Commands を使用します。

2.3 Bulk-Only Transport について

Bulk-Only Transport はバルク転送のみ使用し、ホスト PC とファンクション間でデータの転送が行われます。

バルク転送は、データを送信する向きにより 2 つに分ける事が出来ます。ホストコントローラからファンクションにデータを送信する転送をバルクアウト転送。ホストコントローラにファンクションからデータを送信する転送をバルクイン転送と言います。

Bulk-Only Transport では、バルクアウト転送とバルクイン転送を予め定めた組み合わせにする事により、ホスト-ファンクション間のデータ転送を行います。Bulk-Only Transport は必ず**図** 2.1 に示すバルク転送の組合せになります。それぞれのバルク転送には異なった意味がありステージ（トランスポート）として管理します。

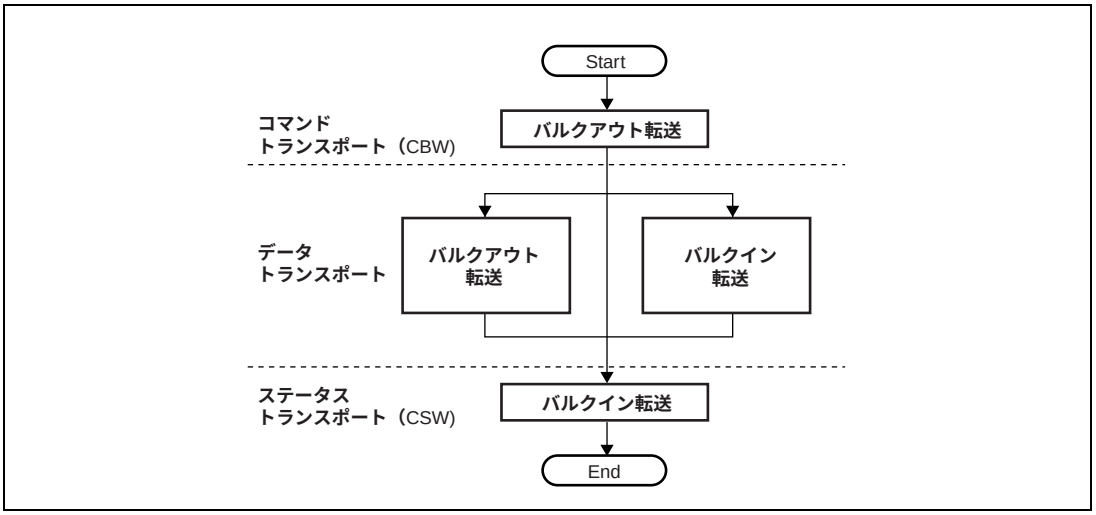


図 2.1　**転送方法とトランスポートの関係**

ホスト PC に、Bulk-Only Transport プロトコルの使用を伝えるためには、Interface Descriptor の bInterfaceProtocol フィールドに値 0x50 を記述することが必要です。

2.3.1　コマンドトランスポートについて

コマンドトランスポートはホスト PC がファンクションにバルクアウト転送を用いてコマンドを送ります。このコマンドパケットが Command Block Wrapper（CBW）として定義されており、Bulk-Only Transport は必ず CBW から始まります。

CBW は、ホスト PC からバルクアウト転送を使用して 31 バイト長のパケットで送られてきます。

内容は**表** 2.2 に示すフォーマットで送られます。

表 2.2

	7	6	5	4	3	2	1	0
00-03	dCBWSignature							
04-07	dCBWTag							
08-0B	dCBWDataTransferLength							
0C	bmCBWFlags							
0D	**リザーブ（0）**				bCBWLUN			
0E	**リザーブ（0）**			bCBWCBLength				
0F-1E	CBWCB							

各フィールドを下記に説明いたします。

dCBWSignature ：データパケットが CBW であると認知するためのフィールド。
値は 43425355h（リトルエンディアン）です。

dCBWTag ：コマンドブロックタグ。CBW と対応する CSW を結びつけるために存在し、
ホスト PC が指定します。

dCBWDataTransferLength：データトランスポートの予定データ長。
ここが 0 の場合データトランスポートは存在しません。

bmCBWFlags ：このフィールドのビットは、ビット 7 が 0 の場合、データトランス
ポートはバルクアウト転送で行われ、1 の場合、バルクイン転送で行われます。ビット
6～0 は 0 固定です。

bCBWLUN ：コマンドブロックが送られている装置の論理ユニット番号（Logical Unit Number）。

bCBWCBLength ：次の CBWCB フィールドの有効バイト数を表します。

CBWCB ：ファンクションによって実行されるコマンドブロックを格納するフィールド。ここにホ
スト PC が実行したいコマンド（本サンプルプログラムでは SCSI コマンド）が入りま
す。

2.3.2 ステータストランスポートについて

ステータストランスポートはファンクションがホスト PC にバルクイン転送を用いてコマンド実行結果を送ります。

このステータスパケットが Command Status Wrapper（CSW）として定義されており、Bulk-Only Transport は必ず CSW で終わります。

CSW は、ホスト PC へバルクイン転送を使用して 13 バイト長のパケットで送ります。

内容は**表** 2.3 に示すフォーマットで送られます。

表 2.3

	7	6	5	4	3	2	1	0
0-3	dCSWSignature							
4-7	dCSWTag							
8-B	dCSWDataResidue							
C	bCSWStatus							

各フィールドを下記に説明いたします。

dCSWSignature ：データパケットが CSW であると認知するためのフィールド。値は 53425355h（リトルエンディアン）です。

dCSWTag ：コマンドブロックタグ。CBW に CSW を結びつけるために存在し、CBW の dCBWTag フィールドと同じ値が入ります。

dCSWDataResidue：CBW の dCBWDataTransferLength 値と実際にファンクションが処理したデータ量の相違を報告します。

bCSWStatus ：コマンドの成功あるいは失敗を示します。コマンドが正常に完了した場合ファンクションはこのフィールドを 0x00 にセットします。ゼロ以外の値は次の通りとし、コマンド実行時の不具合を示します。

コマンドフェイルは 0x01、フェーズエラーは 0x02。

2.3.3 データトランスポートについて

データトランスポートは、ホスト PC とファンクション間のデータ転送を行うトランスポートです。例えば、Read/Write コマンド（4.6 章参照）では、データトランスポートにてストレージ各セクタの実データを送信します。

データトランスポートは複数のバストランザクションで構成されます。

データトランスポートで行われるデータ転送はバルクアウト転送かバルクイン転送のどちらか一方です。どちらになるかは CBW データの bmCBWFlags フィールドで決定されます。

（1） データトランスポート（バルクアウト転送）について

データトランスポートがバルクアウト転送の場合について説明します。

この状態になるのは、CBW データの bmCBWFlags フィールドのビット 7 が 0 であり、CBW データの dCBWDataTransferLength フィールドが 0 ではない場合です。

ここでは CBW データの dCBWDataTransferLength フィールドで予定した長さのデータをファンクションが受信します。ここで転送されるデータは、CBW データの CBWCB フィールドで指定された SCSI コマンドを実行する際に必要なデータです。

（2） データトランスポート（バルクイン転送）について

データトランスポートがバルクイン転送の場合について説明します。

この状態になるのは、CBW データの bmCBWFlags フィールドのビット 7 が 1 であり、CBW データの dCBWDataTransferLength フィールドが 0 ではない場合です。

ここでは CBW データの dCBWDataTransferLength フィールドで予定した長さのデータをファンクションがホスト PC に送信します。ここで転送されるデータは、CBW データの CBWCB フィールドで指定された SCSI コマンドを実行した結果のデータです。

2.3.4 クラスコマンドについて

クラスコマンドとは、USB の各クラス定義ごとに定められているコマンドです。クラスコマンドはコントロール転送を使用します。

USB Mass Storage Class Bulk-Only Transport をデータ転送プロトコルとして使用する場合にサポートしなければならないコマンドは 2 種類あります。**表** 2.4 にクラスコマンドを示します。

表 2.4 **クラスコマンド一覧**

bRequest **フィールド値**	**コマンド**	**コマンドの意味**
255（0xFF）	Bulk-Only Mass Storage Reset	**インタフェースをリセットする**
254（0xFE）	Get Max LUN	**サポートする** LUN **の数を調べる**

Bulk-Only Mass Storage Reset コマンドを受信した場合、ファンクションは USB Mass Storage Class Bulk-Only Transport で使用する全てのインタフェースをリセットします。

Get Max LUN コマンドを受信した場合、ファンクションは使用できる最大の論理ユニット番号を返答します。当サンプルシステムの場合、論理ユニットは 1 つなので返答値は 0 をホストに返答します。

2.4 サブクラスコード SCSI transparent command set について

ファンクションはホスト PC より送信される CBW 内サブクラスコマンドに対応し、各コマンドを処理する必要があります。

本サンプルプログラムでは、SCSI コマンドの中から**表** 2.6 に示す 9 コマンドをサポートしています。また、未サポートのコマンドについては、ホスト PC に対し CSW を使用し「コマンドフェイルである」と報告しています。

表 2.6 **サポートコマンド一覧**

Operation Code	**コマンド名**	**コマンドの動作**
12	INQUIRY	**ドライブに関する情報をホストに伝える**
25	READ CAPACITY	**メディアのセクタに関する情報をホストに伝える**
28	READ（10）	**指定された読み出しセクタから、指定セクタ量のデータを読み出す**
2A	WRITE（10）	**指定された書き込みセクタから、指定セクタ量のデータを書き込む**
03	REQUEST SENSE	**前のコマンドでエラーが発生した時どのようなエラーが発生したかをホストに伝える**
1A	MODE SENSE（10）	**ドライブの状態をホストに伝える**
1E	PREVENT ALLOW MEDIUM REMOVAL	**メディアの着脱を禁止／許可します**
00	TEST UNIT READY	**メディアが使用可能か否かを調べる**
2F	VERIFY（10）	**メディア上のデータにアクセス可能を確かめる**

3. 開発環境

この章では、本システムの開発に使用した開発環境について説明します。本システムの開発は、以下のデバイス（ツール）を使用しました。

- H8S/2215 Solution Engine（以下MS2215CP　型名MS2215CP01-C/S）日立超LSIシステムズ社製
- E6000（型名 HS2214EPI61H）Emulator　日立製作所製
- H8S/2215シリーズTFP120用ユーザシステムインタフェースケーブル（以下H8S/2215ユーザケーブル型名 HS2215ECN61H）日立製作所製
- ISA（またはPCI/PCMCIA）スロット搭載のPC（Windows95/98）
- USBホスト用PC（Windows 2000/Windows Millennium EditionまたはMac OS9）
- USBケーブル
- Hitachi Debugging Interface（以下HDI）日立製作所製
- Hitachi Embedded Workshop（以下HEW）日立製作所製

3.1 ハードウエア環境

図 3.1 に各デバイスの接続形態を示します。

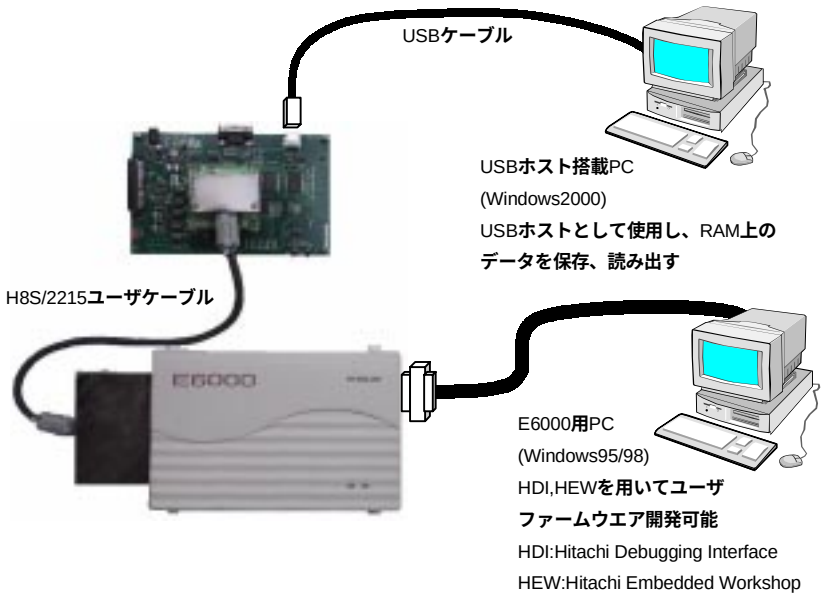


図 3.1　**デバイスの接続形態**

(1) MS2215CP

MS2215CP ボードのジャンパピンのいくつかを出荷時の設定から変更する必要があります。電源を投入する前に、これらの設定をよくご確認ください。その他のジャンパピンを変更する必要はありません。

表 3.1 ジャンパピンの設定

出荷時	変更後	ジャンパピンの機能
J9 1-2 ショート	J9 2-3 ショート	EXTAL48 端子切り換え

(2) USB ホスト PC

USB ポート搭載の、Windows 2000/Windows Millennium Edition または Mac OS9 をインストールしたパソコンを USB ホストとして使用します。本システムでは、上記 OS に標準で搭載されている USB Mass Storage Class（Bulk-Only Transport）のデバイスドライバを使用しますので、新たにドライバをインストールする必要はありません。

(3) E6000

E6000 用 PC と E6000 エミュレータの接続インタフェースは ISA を使用しました。

ISA スロットに E6000I/F ボードを挿入し、接続用のケーブルを介して E6000 と E6000I/F ボードを接続してください。次に H8S/2215 ユーザケーブルを用いて、E6000 と MS2215CP を接続してください。接続後、HDI を起動してエミュレーションを行います。

3.2 ソフトウエア環境

サンプルプログラムと、今回使用したコンパイラおよびリンクについて説明します。

3.2.1 サンプルプログラム

サンプルプログラムとして必要なファイルは、すべて H8S2215 フォルダ内に収められています。HEW、HDI がインストールされたパソコンに、このフォルダごと移動して頂くと、すぐにサンプルプログラムを使用することができます。フォルダに含まれるファイルを以下**図** 3.2 に示します。

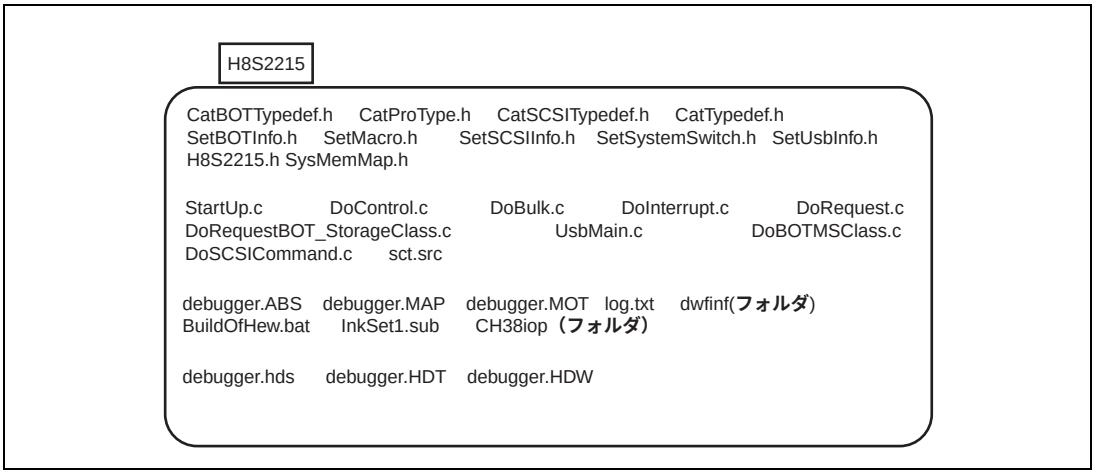


図 3.2 **フォルダに含まれるファイル**

3.2.2 コンパイルおよびリンク

サンプルプログラムのコンパイルおよびリンクは、以下のソフトウエアにより行いました。

Hitachi Embedded Workshop Version1.0（release9） （以下 HEW）

HEW を C:¥Hew にインストール゛した場合、コンパイルおよびリンクの手順は以下のようになります。

まず、コンパイル時に作業用として、Tmp という名前のフォルダを C:¥Hew のフォルダ内に作成してください（**図** 3.3）。

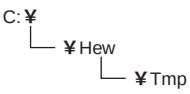


図 3.3 **作業フォルダの作成**

次に、サンプルプログラムが格納されているフォルダ（H8S2215）を、C:¥Usr にコピーしてください（もしくは、任意の場所にコピーし、フォルダ内に含まれている debugger.hds ファイルに記述されている“ C:¥Usr¥h8s2215 ”を、フォルダをコピーしたパスに変更してください。この中には、サンプルプログラムと共に BildOfHew.bat というバッチファイルが含まれています。このバッチファイルでは、パスの設定、コンパイルオプションの指定、コンパイルおよびリンク結果を示すログファイルの指定等を行っています。BildOfHew.bat を実行すると、コンパイルおよびリンクが行われます。その結果、フォルダ内にはファイル名 debugger.ABS の実行ファイルが作成されます。このとき同時にマップファイル debugger.MAP とログファイル log.txt が作成されます。マップファイルにはプログラムのサイズ、および変数のアドレスが示されています。コンパイルの結果（エラーの有無等）はログファイルに記録されます（**図** 3.4）。

【注】 HEW **を** C:¥HEW **以外にインストールした場合、**BildOfHew.bat **内の「コンパイラパスの設定」と「コンパイラが使用する環境変数の設定」、**lnkSet1.sub **内の「ライブラリーの指定」を変更する必要があります。この場合、コンパイラパスの設定は** ch38.exe **のパス、コンパイラが使用する環境変数** ch38 **の設定は** machine.h **のフォルダ、**ch38tmp **の設定はコンパイル作業フォルダをそれぞれ指定してください。また、ライブラリーの指定は** c8s2ba.lib **のパスを指定してください。**

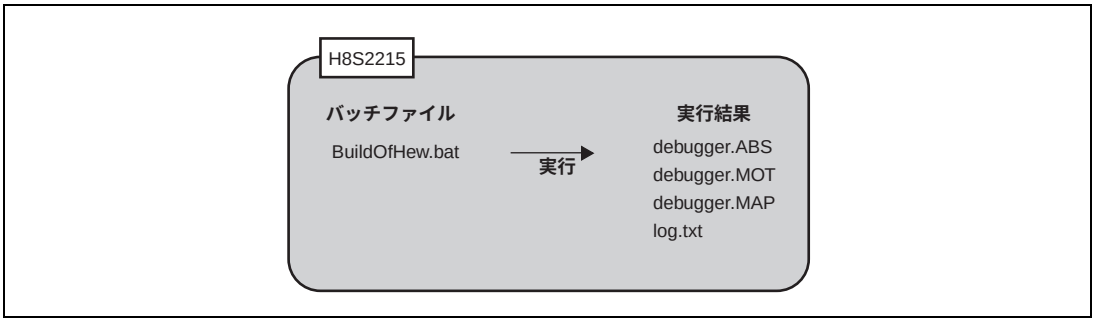
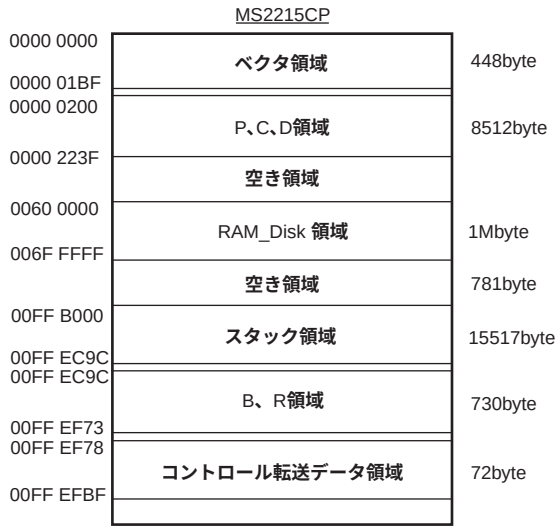


図 3.4 **コンパイル結果**

3.3 プログラムのロードと実行方法

図 3.5 にサンプルプログラムのメモリマップを示します。



【注】 コンパイラバージョン、コンパイル条件、ファームウェアのアップグレード等によりメモリマップは変化します。

図 3.5 **メモリマップ**

図 3.5 のように、本サンプルプログラムはベクタ P、C、D の各領域をエリア 1 内蔵 ROM 領域（E6000 の貸し出しメモリ）上に配置。スタック、B、R、コントロール転送の各領域を内蔵 RAM 上に配置、RAM_Disk として使用する領域を SRAM 上に配置しています。これらのメモリへの割り付けは、H8S2215 フォルダ内に含まれる lnkSet1.sub で指定します。プログラムの配置を変更する場合は、このファイルを変更してください。

3.3.1 プログラムのロード

MS2215CP にサンプルプログラムをロードするには、以下のような手順で行います。

- HDIをインストールしたE6000用PCとE6000を接続してください。
- H8S/2215ユーザケーブルでE6000とMS2215CPを接続してください。
- E6000用PCの電源を投入し、起動してください。
- E6000とMS2215CPの電源を投入してください。
- H8S2215フォルダ内のdeugger.hdsを実行してください。

以上の操作で、サンプルプログラムを MS2215CP にロードする事ができます。

3.3.2 プログラムの実行

「3.3.1 プログラムのロード」でロードしたプログラムを実行するためには、プログラムカウンタ（PC）を設定する必要があります。

メニューバーの View→Register Window を選択し、Registers ウインドを開きます。ウインド内の該当レジスタ（PC）の数値エリアをダブルクリックすると、ダイアログボックスが開きレジスタの値を変更する事ができます。このダイアログボックスで PC を H'0000 0200 に設定してください。

以上の設定後、メニューバーの Run→Go を選択するとプログラムが実行されます。

3.4 RAM Disk の使用方法

Windows 2000 を用いた場合を例に以下に説明します。

プログラムを実行した状態で、USB ケーブルのシリーズ B コネクタを MS2215CP に挿入し、反対側のシリーズ A コネクタを USB ホスト PC に接続します。

コントロール転送およびバルク転送を用いたエニュミレーション終了後、デバイスマネージャーの USB コントローラの下に USB 大容量記憶装置デバイスが表示され、ディスクドライブの下に HITACHI EX RAM Disk USB Device が表示されます。その結果、ホスト PC は MS2215CP を記憶デバイスとして認識し、マイコンピュータの中にローカルディスクがマウントされます。

次にローカルディスクをフォーマットします。

ローカルディスクを選択し、マウスの右ボタンをクリックし、フローティングメニュー内のフォーマットを選択します。ドライブのフォーマット選択ウインドが開かれるので、フォーマットの設定を行います。ファイルシステム選択項目が FAT である事を確認し、開始ボタンをクリックしてください。

フォーマットの実行確認ウインドウが出力されるので、OK ボタンをクリックしてください。

フォーマットが完了するとフォーマット完了のメッセージウインドウが出力されるので、OK ボタンをクリックしてください。

ドライブのフォーマット選択ウインドに戻るので、閉じるボタンをクリックしてウインドを閉じてください。

以上で MS2215CP を USB 接続の RAM-Disk として使用できます。

4. サンプルプログラム概要

この章ではサンプルプログラムの特長やその構成について説明します。本サンプルプログラムは MS2215CP 上で動作し MS2215CP が RAM-Disk として動作します。USB 転送は USB ファンクションモジュールからの割り込みによって開始します。H8S/2215 内蔵モジュールの割り込みのうち、USB ファンクションモジュールに関連する割り込みは、EXIRQ0、EXIRQ1、IRQ6 の 3 種類ですが、本サンプルでは EXIRQ0 のみ使用しています。

本サンプルプログラムの特長を以下に示します。

- コントロール転送を行うことができます。
- バルクアウト転送でホストコントローラからデータを受信することができます。
- バルクイン転送でホストコントローラにデータを送信することができます。
- SCSIコマンドに対応するRAM-Diskとして動作します。

4.1 状態遷移図

図 4.1 に、本サンプルプログラムの状態遷移図を示します。本サンプルプログラムは、**図** 4.1 のように 3 つの状態に遷移します。

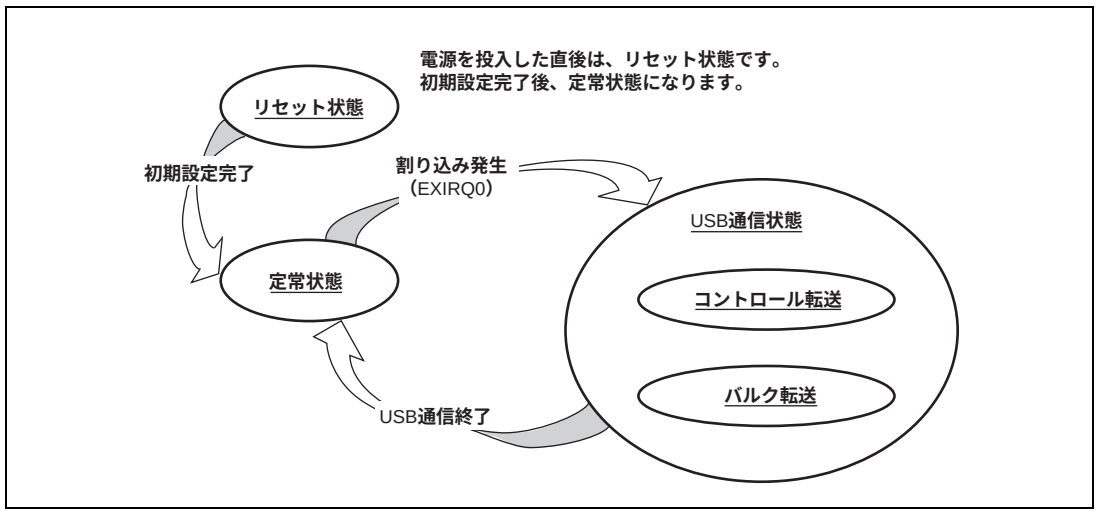


図 4.1　**状態遷移図**

- **リセット状態**

  パワーオンリセット・マニュアルリセットの際には、この状態になります。リセット状態では、主にH8S/2215の初期設定を行います。

- **定常状態**

  初期設定が完了すると、メインループで定常状態となります。

- **USB通信状態**

  定常状態において、USBモジュールから割り込みが発生するとこの状態になります。USB通信状態では、割り込みの種類に応じた転送方式によるデータ転送を行います。本サンプルプログラムで使用する割り込みは割り込みフラグレジスタ0～3（UIFR0～3）によって示される計9種類です。割り込み要因が発生すると、UIFR0～3の対応するビットに1がセットされます。

4.2 USB 通信状態

USB 通信状態は、転送方式ごとに 2 つの状態に分類することができます（**図** 4.2 参照）。割り込みが発生すると、先ず USB 通信状態へと遷移し、さらに割り込みの種類に応じて各転送状態へ分岐します。分岐の方法については**「第 5 章　サンプルプログラムの動作」**で説明します。

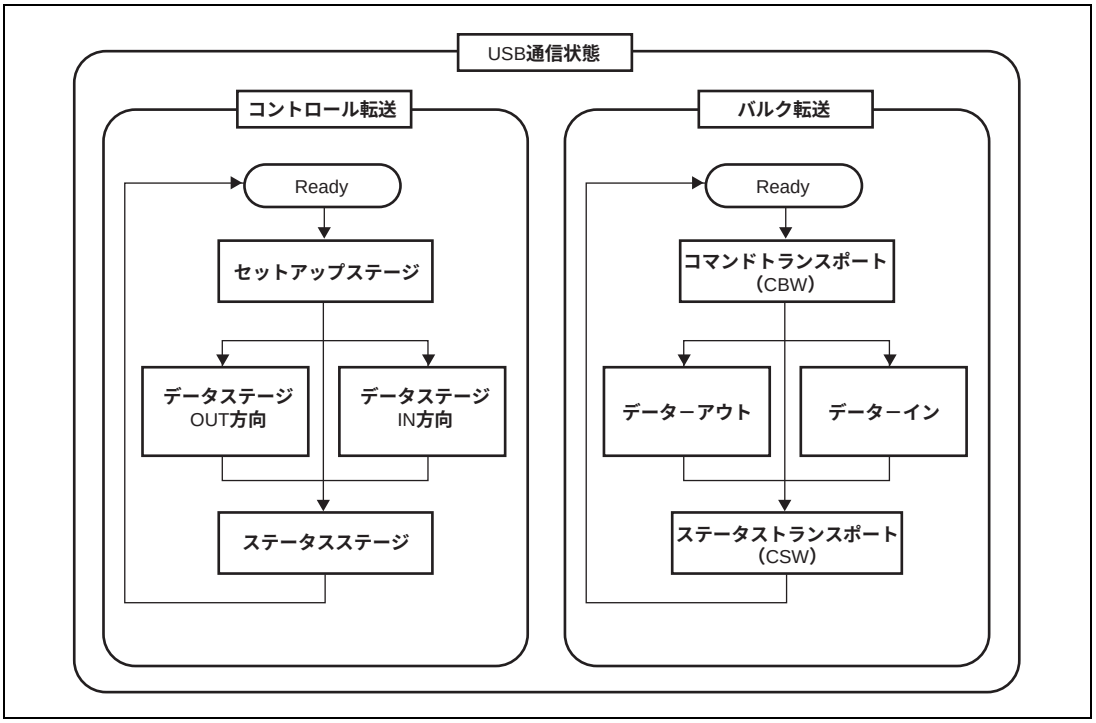


図 4.2　USB **通信状態**

4.2.1 コントロール転送について

コントロール転送は主に、デバイス情報の取得、デバイスの動作状態を設定する際などに使用されます。そのため、ホスト PC にファンクションを接続した際、最初に行われる転送でもあります。

コントロール転送の一連の転送処理は、2 または 3 つのステージから構成されます。コントロール転送のステージは、「セットアップステージ」「データステージ」「ステータスステージ」に分類することができます。

4.2.2 バルク転送について

バルク転送は時間的制約がない大量のデータを、エラーなく転送する場合に使用します。データの転送速度は保証されませんが、データの内容は保証されます。USB Mass Storage Class（Bulk-Only Transport）ではバルク転送を使用し、ホスト PC とファンクション間でストレージデータを転送します。

USB Mass Storage Class（Bulk-Only Transport）の一連の転送処理（リードやライトなど）は、2 または 3 つのステージから構成されます。USB Mass Storage Class（Bulk-Only Transport）のステージは「コマンドトランスポート（CBW）」「データトランスポート」「ステータストランスポート（CSW）」に分類することができます。

4.3 ファイル構成

本サンプルプログラムは、8 個のソースファイルと 11 個のヘッダーファイルで構成されています。全構成ファイルを**表** 4.1 に示します。各関数は、転送方式または機能ごとに 1 つのファイルにまとめてあります。**図** 4.3 に各ファイルの関係を階層構造で示します。

表 4.1 **ファイル構成**

ファイル名	主な役割
StartUp.c	**マイコンの初期設定**
UsbMain.c	**割り込み要因の判定** **パケットの送受信**
DoRequest.c	**ホストが発行するセットアップコマンドの処理**
DoRequestBOT_StorageClass.c	Mass Storage Class（Bulk-Only Transport）**クラスコマンドの処理**
DoControl.c	**コントロール転送を実行**
DoBulk.c	**バルク転送を実行**
DoBOTMSClass.c	Mass Storage Class（Bulk-Only Transport）**を実行**
DoSCSICommand.c	SCSI **コマンドの解析および処理**
h8s2215.h	H8S/2215 **レジスタ定義**
SysMemMap.h	MS2215CP **のメモリマップのアドレス定義**
CatProType.h	**プロトタイプ宣言**
CatTypedef.h	USB **ファームウエアで使用する基本の構造体定義**
CatBOTTypedef.h	Bulk-Only Transport **用構造体定義**
CatSCSITypedef.h	SCSI **用構造体定義**
SetUsbInfo.h	USB **対応に必要な変数の初期設定**
SetBOTInfo.h	Bulk-Only Transport **対応に必要な変数の初期設定**
SetSCSIInfo.h	SCSI **コマンド対応に必要な変数の初期設定**
SetSystemSwitch.h	**システムの動作設定**
SetMacro.h	**マクロ定義**

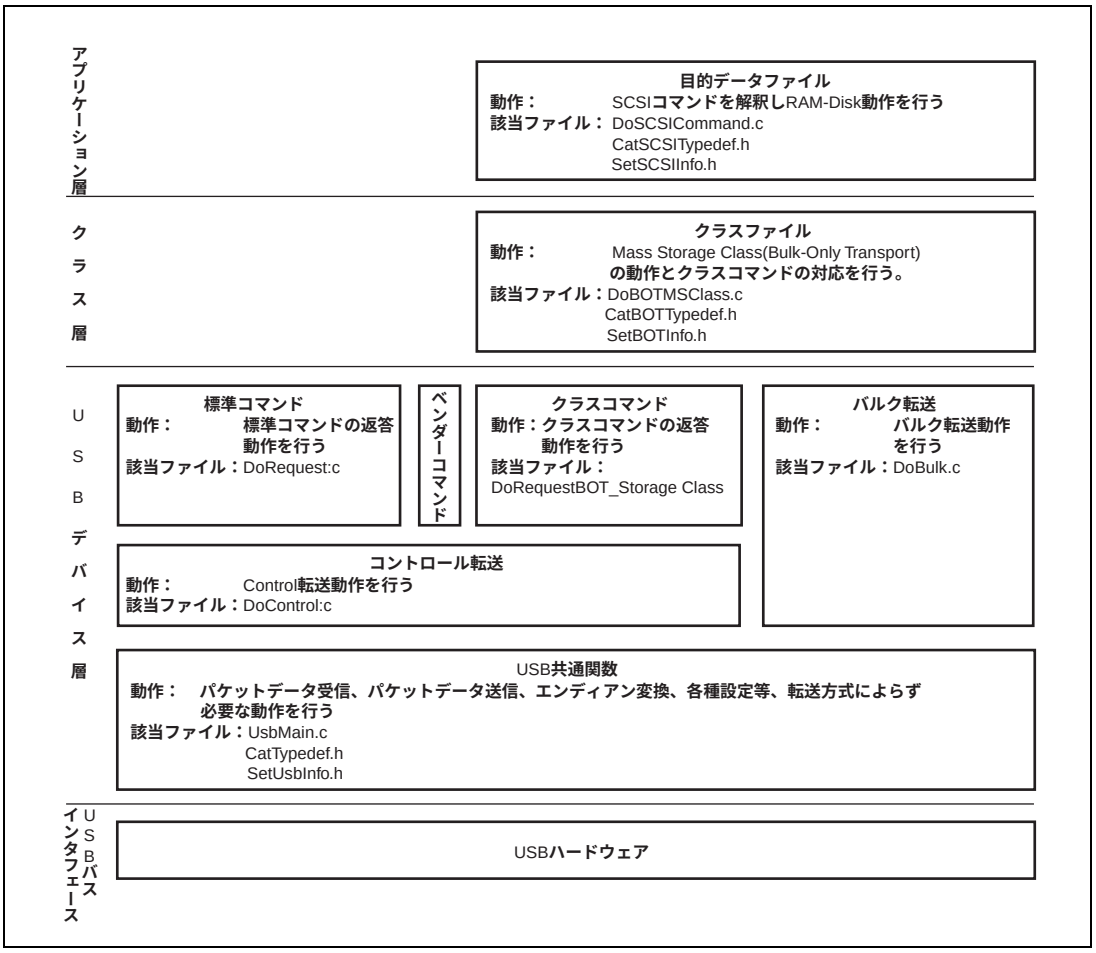


図 4.3 **ストレージクラス（**BOT**）ファームウェアの階層構造**

4.4 関数の機能

表 4.2 に各ファイルに含まれる関数とその機能を示します。

表 4.2-1 StartUp.c

格納ファイル	関数名	機　　能
StartUp.c	SetPowerOnSection	BSC、端子、割り込みコントローラの設定、各初期化ルーチン呼び出しを行いメインループへ移行
	_INITSCT	初期値がある変数を、RAM のワークエリアにコピー
	InitMemory	バルク通信で使用する RAM 領域をクリア
	InitSystem	USB クロックの設定、システム割り込み、マスクの設定

パワーオンリセット、またはマニュアルリセットの際には、StartUp.c の SetPowerOnSection が呼び出されます。ここでは H8S/2215 の初期設定や、バルク転送に使用する RAM 領域のクリアを行います。

表 4.2-2 UsbMain.c

格納ファイル	関数名	機　　能
UsbMain.c	BranchOfInt	割り込み要因の判定と，割り込みに応じた関数を呼び出す
	GetPacket	ホストコントローラから転送されたデータを、RAM に書き込む
	GetPacket4	ホストコントローラから転送されたデータを、ロングワードサイズで RAM に書き込む。リングバッファ対応版（本サンプルプログラムでは使用しません）
	GetPacket4S	ホストコントローラから転送されたデータを、ロングワードサイズで ROM に書き込む。高速版
	PutPacket	ホストコントローラに転送するデータを USB モジュールに書き込む
	PutPacket4	ホストコントローラに転送するデータをロングワードサイズで USB モジュールに書き込む。リングバッファ対応版（本サンプルプログラムでは使用しません）
	PutPacket4S	ホストコントローラに転送するデータを、ロングワードサイズで USB モジュールに書き込む。高速版
	SetControlOutContents	ホストから送られたデータに書き換える
	SetUsbModule	USB モジュールの初期設定
	ActBusReset	バスリセット受信時に FIFO のクリアを行う
	ActBusVcc	USB ケーブル接続、切断時に D+プルアップと USB モジュールの制御を行う
	ConvRealn	指定した番地から指定バイト長のデータを読み出す
	ConvReflexn	指定した番地から指定バイト長のデータを逆順に読み出す

UsbMain.c では、主に USB 割り込みフラグレジスタによって割り込み要因を判定し、割り込みの種類に応じた関数の呼び出しを行います。また、ホストコントローラとファンクションモジュール間におけるパケットの送受信を行います。

表 4.2-3 DoRequest.c

格納ファイル	関数名	機　　能
DoRequest.c	DecStandardCommands	ホストコントローラが発行したコマンドをデコードし、そのうち標準コマンドの対応を行う
	DecVenderCommands	ベンダコマンドの対応を行う

コントロール転送時に、ホストコントローラから送られてくるコマンドをデコードし、コマンドに応じた処理を行います。本サンプルプログラムでは、ベンダ ID の値に 045B（ベンダ：日立）を使用しています。お客様にて製品を開発される場合は「USB Implementers Forum」にてお客様のベンダ ID を取得願います。また、ベンダコマンドは使用していないため、DecVenderCommands では何も行っていません。ベンダコマンドを使用する際には、お客様でプログラムを作成願います。

表 4.2-4　DoRequestBOT_StorageClass.c

格納ファイル	関数名	機　　能
DoRequestBOT_ StorageClass.c	DecBOTClass Commands	USB Mass Storage Class（Bulk-Only Transport）**コマンドの対応を行う**

Mass Storage Class（Bulk-Only Transport）コマンド（Bulk-Only Mass Storage Reset と Get Max LUN）に応じた処理を行います。

Bulk-Only Mass Storage Reset コマンドは Bulk-Only Transport で使用している全てのインターフェースをリセットします。

Get Max LUN コマンドは周辺装置が使用する最大の論理ユニット番号を返答します。当サンプルシステムの場合、論理ユニットは 1 つなので返答値は 0 をホストに返答します。

表 4.2-5　DoControl.c

格納ファイル	関数名	機　　能
DoControl.c	ActControl	**コントロール転送のセットアップステージの制御を行う**
	ActControlIn	**コントロールイン転送（データステージがイン方向の転送）のデータステージとステータスステージの制御を行う**
	ActControlOut	**コントロールアウト転送（データステージがアウト方向の転送）のデータステージとステータスステージの制御を行う**

コントロール転送の割り込み（EP0oTS）が入ると、ActControl がコマンドを取得し、DecStandardCommands でデコードを行います。その後コマンドの種類に応じて、ActControlIn または ActControlOut によってデータステージと、ステータスステージを行います。

表 4.2-6　DoBulk.c

格納ファイル	関数名	機　　能
DoBulk.c	ActBulkOut	**バルクアウト転送を行う**
	ActBulkIn	**バルクイン転送を行う**
	ActBulkInReady	**バルクイン転送の準備を行う**

バルク転送に関する処理を行います。Mass Storage Class（Bulk-Only Transport）では ActBulkInReady を使用しません。

表 4.2-7　DoBOTMSClass.c

格納ファイル	関数名	機　　能
DoBOTMSClass.c	ActBulkOnly	Bulk-Only Transport のステージ別に振分けを行う
	ActBulkOnlyCommand	Bulk-Only Transport の CBW の制御を行う
	ActBulkOnlyIn	（データステージがイン方向の転送）のデータトランスポートとステータストランスポートの制御を行う
	ActBulkOnlyOut	（データステージがアウト方向の転送）のデータトランスポートとステータストランスポートの制御を行う

DoBOTMSClass.c では、Mass Storage Class（Bulk-Only Transport）の 2 ないし 3 つのステージ制御と仕様に従った動作を行います。

表 4.2-8　DoSCSICommand.c

格納ファイル	関数名	機　　能
DoSCSI Command.c	DecBotCmd	ホストから Bulk-Only Transport で送られる SCSI コマンドの対応を行う

DoSCSICommand.c では、ホスト PC から送られてきた SCSI コマンドを解析し、次のデータトランスポートまたはステータストランスポートの準備を行います。

図 4.4 に、**表** 4.2 で説明した関数の相関関係を示します。上位側の関数が、下位側の関数を呼び出すことができます。また、複数の関数が同一の関数を呼び出すこともあります。定常状態では、SetPowerOnSection が他の関数を呼び出し、割り込みの発生によって遷移する USB 通信状態では、BranchOfInt が他の関数を呼び出します。**図** 4.4 は、関数の上下関係を示しているもので、関数が呼び出される順序は示していません。関数がどのような順序で呼び出されるかについては、**「第 5 章　サンプルプログラムの動作」**のフローチャートをご覧ください。

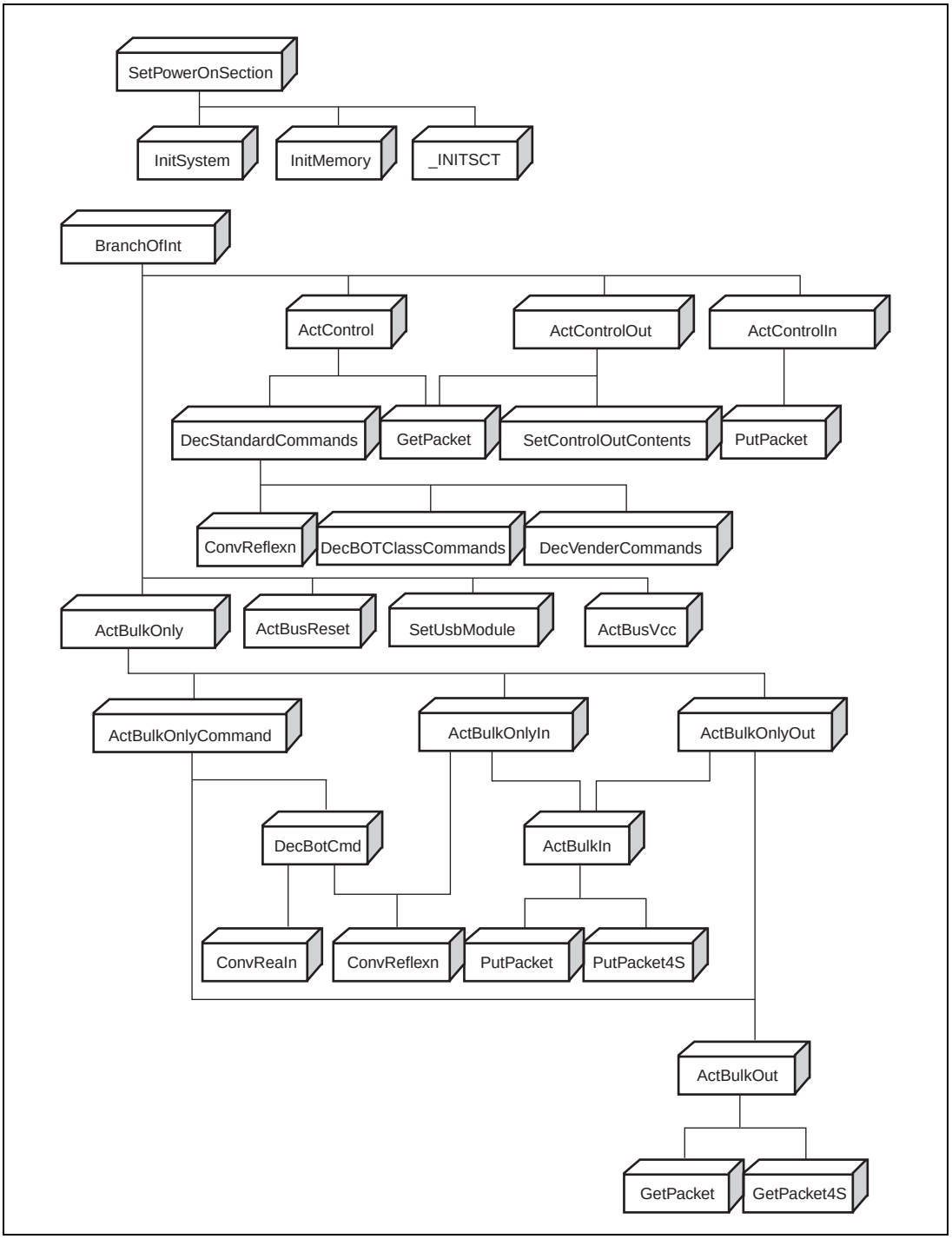

SetPowerOnSection
InitSystem
InitMemory
_INITSCT
BranchOfInt
ActControl
ActControlOut
ActControlIn
DecStandardCommands
GetPacket
SetControlOutContents
PutPacket
ConvReflexn
DecBOTClassCommands
DecVenderCommands
ActBulkOnly
ActBusReset
SetUsbModule
ActBusVcc
ActBulkOnlyCommand
ActBulkOnlyIn
ActBulkOnlyOut
DecBotCmd
ActBulkIn
ConvReaIn
ConvReflexn
PutPacket
PutPacket4S
ActBulkOut
GetPacket
GetPacket4S

図 4.4　**関数の相関関係**

4.5 RAM-Disk について

本サンプルプログラムでは MS2215CP 上の SRAM を Disk 装置に見立て、ホスト PC に対し MS2215CP（ファンクション）は Disk であると報告しています。

ファンクションの Disk 装置には**図** 4.5 に示すようにマスターブートブロックと、パーティションブートブロックが存在しています。システム立ち上げ時に初期化ルーチンを用いて SRAM 上の RAM-Disk 領域にマスターブートブロックと、パーティションブートブロックを書き込みます。

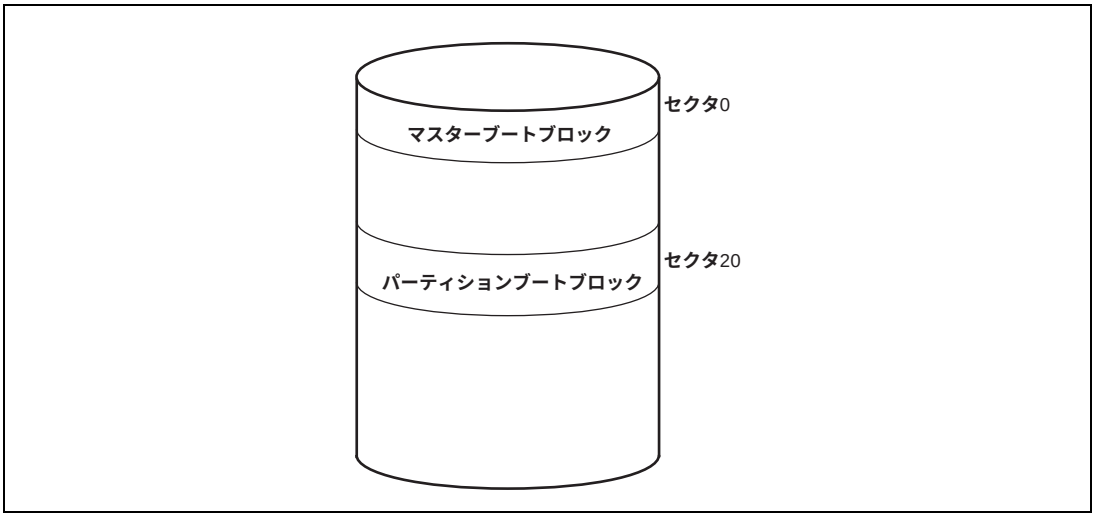


図 4.5 Disk **の構造**

ホスト PC からファンクションに対するアクセス（データの保存、読み出し）は SCSI コマンドを使用します。SCSI コマンドの動作を行う場合、**図** 4.5 の構造を理解し動作を書く必要があります。

4.6 サポートする SCSI コマンドの動作について

本サンプルプログラムがサポートする SCSI コマンドの動作を表 4.3 に示します。

表 4.3 SCSI コマンド動作表

コマンド名	トランスポート名	動作内容
INQUIRY	CBW	コマンドをデコードし、INQUIRY コマンドであることを認識後、ROM に格納してある INQUIRY 情報（96 バイト）の送信準備を行います。
	データ	ホスト PC に対し INQUIRY 情報をバルクイン転送にて送信します。
	CSW	ホスト PC に対しコマンド実行結果を送信します。送信データが 96 バイト以下であれば正常終了を送信します。
READ CAPACITY	CBW	コマンドをデコードし、READ CAPACITY コマンドであることを認識後、SRAM 上に展開してある Disk 装置内にあるパーティションブートブロック内の 1 セクタ当りのバイト数と、ディスクの総セクタ数に格納されている値を読み出し READ CAPACITY 情報（8 バイト）の送信準備を行います。
	データ	ホスト PC に対し READ CAPACITY 情報をバルクイン転送にて送信します。
	CSW	ホスト PC に対しコマンド実行結果を送信します。送信データが 8 バイト以下であれば正常終了を送信します。
READ（10）	CBW	コマンドをデコードし、READ（10）コマンドであることを認識後、SRAM 上に展開してある Disk 装置内の指定された読み出しセクタから、指定セクタ量のデータ送信準備を行います。
	データ	ホスト PC に対し読み出しセクタのデータをバルクイン転送にて送信します。
	CSW	ホスト PC に対し READ（10）コマンド実行結果を送信します。送信データが読み出しバイト数以下であれば正常終了を送信します。
WRITE（10）	CBW	コマンドをデコードし、WRITE（10）コマンドであることを認識後、SRAM 上に展開してある Disk 装置内の指定された書き込みセクタから、指定セクタ量のデータ受信準備を行います。
	データ	ホスト PC から書き込みセクタのデータをバルクアウト転送にて受信します。
	CSW	ホスト PC に対し正常終了を送信します。
REQUEST SENSE	CBW	コマンドをデコードし、REQUEST SENSE コマンドであることを認識後、返答値（直前の SCSI コマンドを実行した結果）の送信準備を行います。
	データ	ホスト PC に対し返答値をバルクイン転送にて送信します。
	CSW	ホスト PC に対し本コマンド実行結果を送信します。送信データが 18 バイト数以下であれば正常終了を送信します。
PREVENT ALLOW MEDIUM REMOVAL	CBW	コマンドをデコードし、PREVENT ALLOW MEDIUM REMOVAL コマンドであることを認識後、ホスト PC に対し正常終了の送信準備を行います。本サンプルソフトの記録メディアは SRAM であり常に着脱不能です。このため本コマンドの返答は常に正常終了となります。
	データ	本コマンドにデータトランスポートは存在しません。
	CSW	ホスト PC に対し正常終了を送信します。

コマンド名	トランスポート名	動作内容
TEST UNIT READY	CBW	コマンドをデコードし、TEST UNIT READY コマンドであることを認識後、ホスト PC に対し正常終了の送信準備を行います。本サンプルソフトの記録メディアは SRAM でありプログラム実行中は常にアクセス可能となっています。このため本コマンドの返答は常に正常終了となります。
	データ	本コマンドにデータトランスポートは存在しません。
	CSW	ホスト PC に対し正常終了を送信します。
VERIFY（10）	CBW	コマンドをデコードし、VERIFY（10）コマンドであることを認識後、ホスト PC に対し正常終了の送信準備を行います。本サンプルソフトの記録メディアは SRAM でありプログラム実行中は常にアクセス可能となっています。このため本コマンドの返答は常に正常終了となります。
	データ	本コマンドにデータトランスポートは存在しません。
	CSW	ホスト PC に対し正常終了を送信します。
MODE SENSE（10）	CBW	コマンドをデコードし、MODE SENSE（10）コマンドであることを認識後、ホスト PC に対しモードパラメータの送信準備を行います。 本サンプルソフトではモードパラメータヘッダのみ値を用意しています。
	データ	ホスト PC に対しモードパラメータをバルクイン転送にて送信します。
	CSW	ホスト PC に対しコマンド実行結果を送信します。送信データが 8 バイト以下であれば正常終了を送信します。
未サポートコマンド	CBW	コマンドをデコードし、未サポートコマンドであれば、REQUEST SENSE の返答値に INVALID FIELD IN CDB を設定後、データトランスポートの準備を行います。
	データ	ホスト PC がバルクイン転送にてデータを要求した場合、ホストが要求した量と同量のデータ（0x00）を送信します。
		ホスト PC がバルクアウト転送にてデータを送信した場合、受信バイト数のカウントを行います。
		データトランスポートがない場合、何も動作は行いません。
	CSW	ホスト PC に対しコマンドフェイルを送信します。

4.7 エラー時の処理について

Mass Storage Class（Bulk-Only Transport）の転送を行う際、ホスト PC とファンクション間で発生するエラーとエラー時のファンクション側の対応動作を示します。

Bulk-Only Transport の規格では次に上げるエラーケースが規定されています。

- CBWが有効でない場合。
- ホストの期待とファンクションが意図する動作（SCSIコマンドで指定された動作）の相違（10ケース）。

以上の 2 種類があります。これ以外の状態については規格書には定められていません。

ホスト-ファンクション間のデータ転送については**表** 4.4 と**表** 4.5 に示す 13 種類の状態が存在します。このうち CASE（1）（6）（12）は正常な転送状態です。

表 4.4 **ホスト-ファンクション間のデータ転送状態**

		ホストは		
		データ転送なしを期待	ファンクションからのデータ受信を期待	ファンクションへのデータ送信を期待
ファンクションは	データ転送なしを意図	(1) Hn = Dn	(4) Hi > Dn	(9) Ho > Dn
	ホストへのデータ送信を意図	(2) Hn < Di	(5) Hi > Di	(10) Ho < > Di
			(6) Hi = Di	
			(7) Hi < Di	
	ホストからのデータ受信を意図	(3) Hn < Do	(8) Hi < > Do	(11) Ho > Do
				(12) Ho = Do
				(13) Ho < Do

表 4.5 **ホスト-ファンクション間データ転送状態解説**

CASE	ホスト-ファンクション間での関係
1	ホストはデータ転送なしを期待し、ファンクションもデータ転送なしを意図する場合
2	ホストはデータ転送なしを期待し、ファンクションはホストへのデータ送信を意図する場合
3	ホストはデータ転送なしを期待し、ファンクションはホストからのデータ受信を意図する場合
4	ホストはファンクションからのデータ受信を期待し、ファンクションはホストへのデータ転送なしを意図する場合
5	ホストが期待したファンクションからのデータ受信数より、ファンクションがホストへ送信するデータ数が少ない場合
6	ホストが期待したファンクションからのデータ受信数と、ファンクションがホストへ送信するデータ数が同じ場合
7	ホストが期待したファンクションからのデータ受信数より、ファンクションがホストへ送信するデータ数が多い場合
8	ホストはファンクションからのデータ受信を期待し、ファンクションはホストからのデータ受信を意図する場合
9	ホストはファンクションへのデータ送信を期待し、ファンクションはデータ転送なしを意図する場合
10	ホストはファンクションへのデータ送信を期待し、ファンクションはホストへのデータ送信を意図する場合
11	ホストが期待したファンクションへのデータ送信数より、ファンクションがホストから受信するデータ数が少ない場合
12	ホストが期待したファンクションへのデータ送信数と、ファンクションがホストから受信するデータ数が同じ場合
13	ホストが期待したファンクションへのデータ数より、ファンクションがホストから受信するデータ数が多い場合

表 4.6 に発生する可能性のあるエラー状況例を示します。

表 4.6　エラー状況例

CASE	エラー状況
2	ホストから READ コマンドが発行される際、USB のデータトランスポートで転送するデータ数が 0 で、SCSI コマンドで指定されたデータ数が 0 以外の場合
3	ホストから WRITE コマンドが発行される際、USB のデータトランスポートで転送するデータ数が 0 で、SCSI コマンドで指定されたデータ数が 0 以外の場合
4	ホストから READ コマンドが発行される際、USB のデータトランスポートで転送するデータ数が 0 で、SCSI コマンドで指定されたデータ数が 0 の場合
5	ホストから READ コマンドが発行される際、USB のデータトランスポートで転送するデータ数より、SCSI コマンドで指定されたデータ数が少ない場合
7	ホストから READ コマンドが発行される際、USB のデータトランスポートで転送するデータ数より、SCSI コマンドで指定されたデータ数が多い場合
8	ホストから WRITE コマンドが発行されたのに、ホストが USB のデータトランスポートでデータを要求する場合
9	ホストから WRITE コマンドが発行される際、USB のデータトランスポートで転送するデータ数が 0 以外で、SCSI コマンドで指定されたデータ数が 0 の場合
10	ホストから READ コマンドが発行されたのに、ホストが USB のデータトランスポートでデータを送ってくる場合
11	ホストから WRITE コマンドが発行される際、USB のデータトランスポートで転送するデータ数より、SCSI コマンドで指定されたデータ数が少ない場合
13	ホストから WRITE コマンドが発行される際、USB のデータトランスポートで転送するデータ数より、SCSI コマンドで指定されたデータ数が多い場合

エラー状況に対するファンクションの対応動作は表 4.7 のようになります。

表 4.7　エラー対応動作表

CASE	エラー時におけるデータトランスポートでのファンクション対応動作
2、3	• CSW のステータスに 0x02 を設定する。
4、5	• ファンクションは dCBWDataTransferLength で示されたデータ長になるようにデータを付加し、ホストにデータを送信する。 • CSW の dCBWDataResidue にデータトランスポートで付加したデータ数を設定する。 • CSW のステータスに 0x01 を設定する。
7、8	• ファンクションは dCBWDataTransferLength で示されたデータ長まで、ホストにデータを送信する。 • CSW のステータスに 0x02 を設定する。
9、11	• ファンクションは dCBWDataTransferLength で示されたデータ長分、データを受信する。 • データトランスポートで受信したデータ数とファンクションで処理したデータ数の差を CSW の dCBWDataResidue に設定する。 • CSW のステータスに 0x01 を設定する。
10、13	• ファンクションは dCBWDataTransferLength で示されたデータ長分、データを受信する。 • CSW のステータスに 0x02 を設定する。

データ転送時のエラー処理フローは、**図** 4.6、4.7、4.8 のようになります。

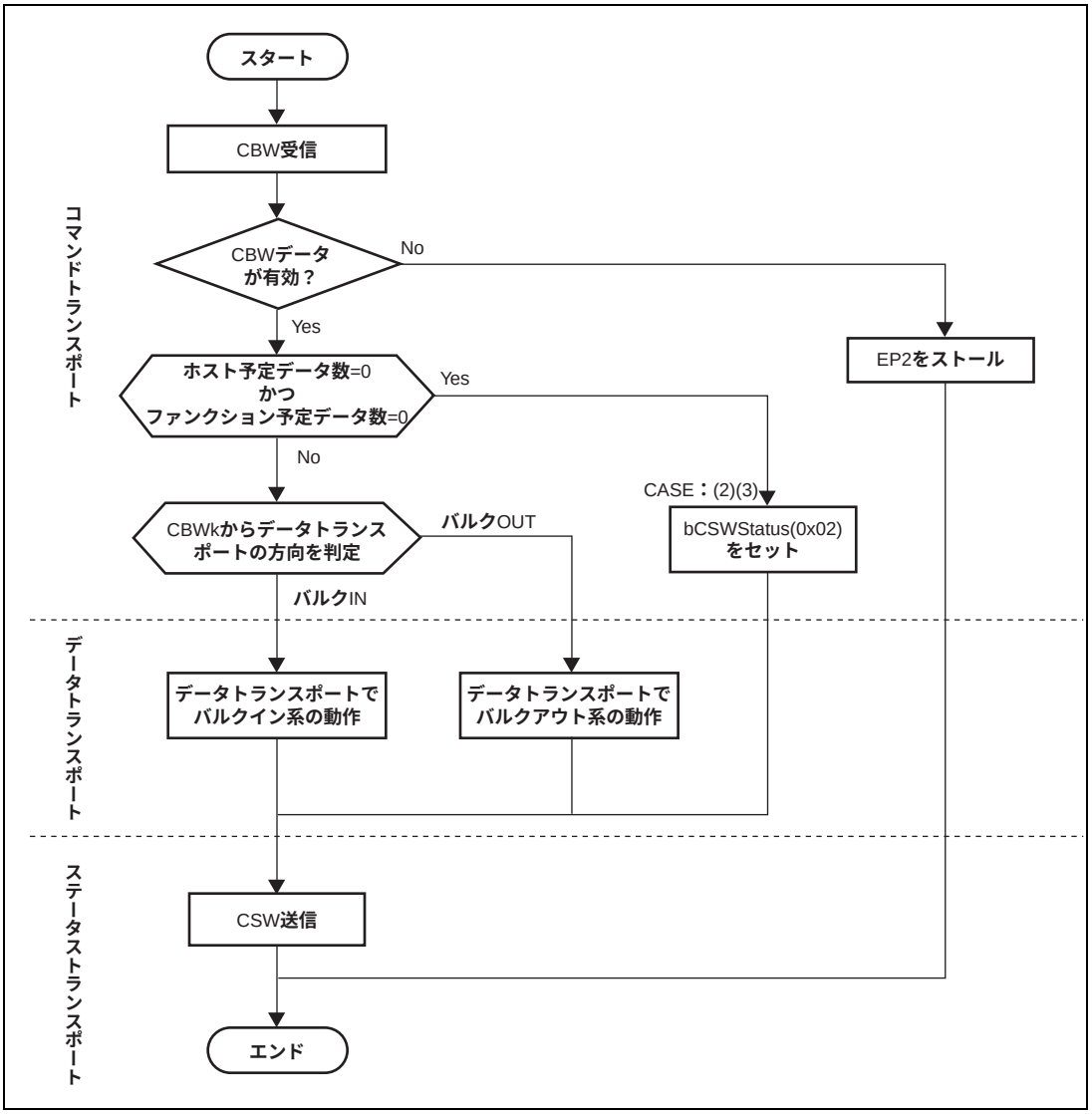


図 4.6　**データ転送時のエラー処理フロー（1）**

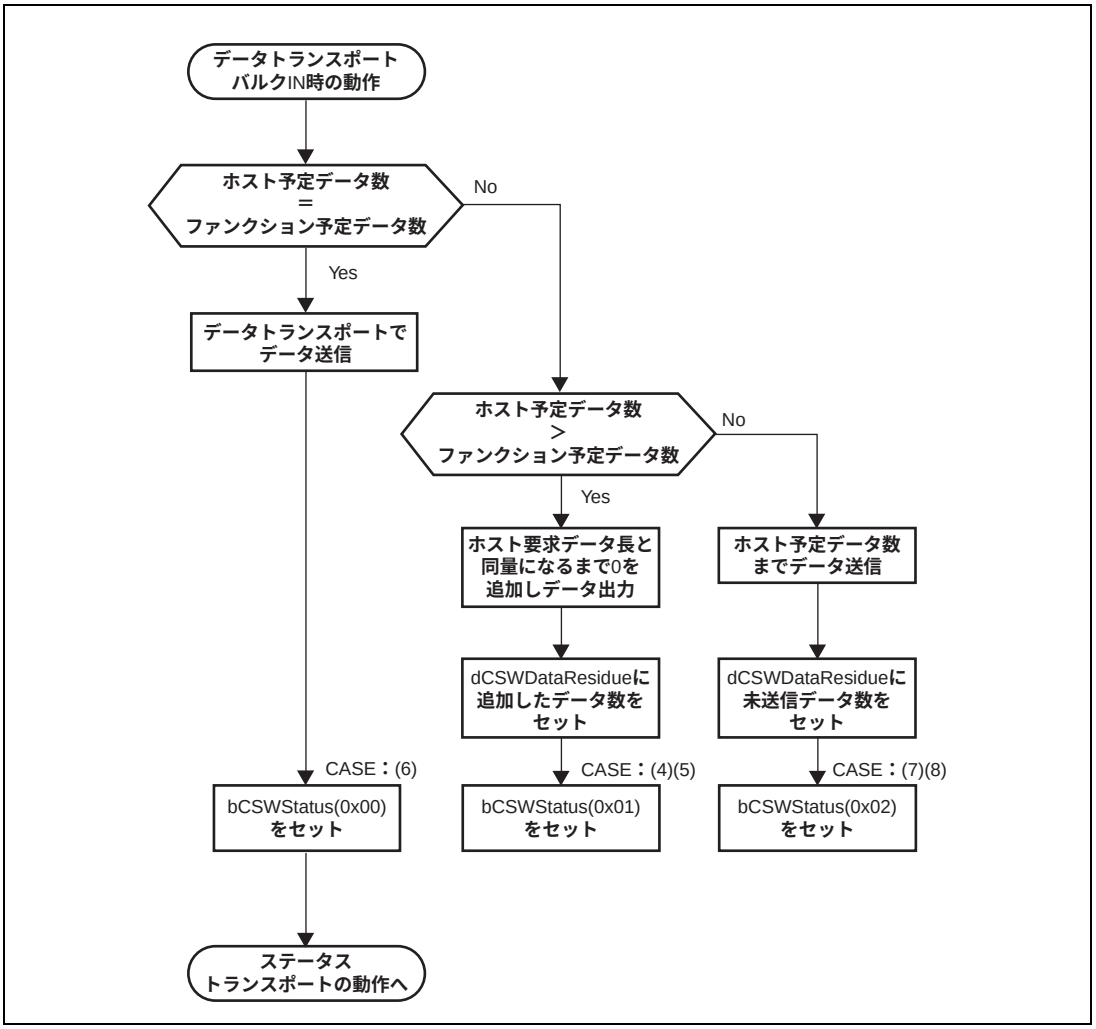


図 4.7　**データ転送エラー発生時の処理フロー（2）**

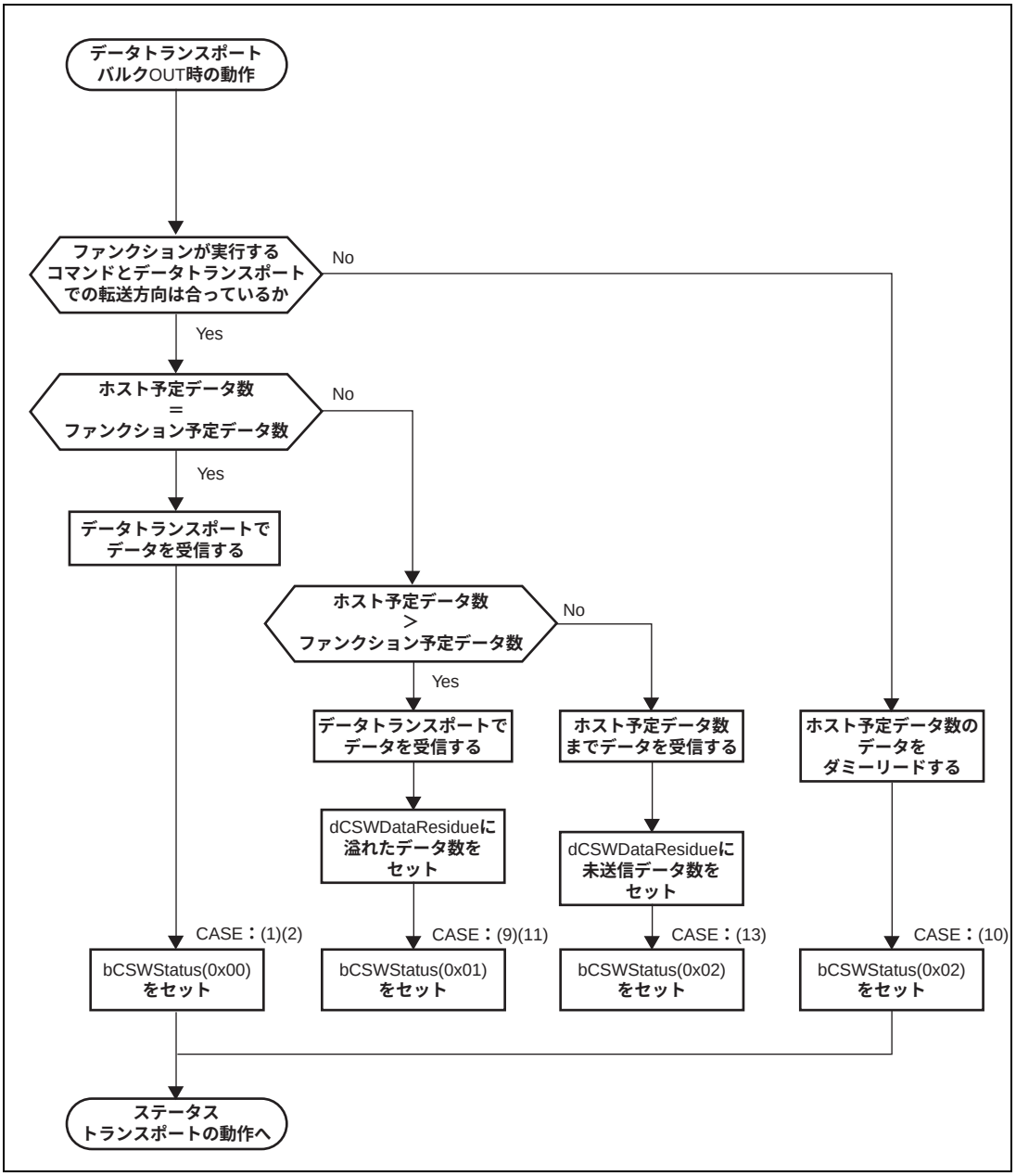


図 4.8　**データ転送エラー発生時の処理フロー（3）**

Mass Storage Class（Bulk-Only Transport）の転送を行う際、CBW トランスポートで一連のデータ転送が始まり、ホスト PC に CSW トランスポートで一連の転送結果（ステータス）を返します。このためデータ転送処理を行う際に、CSW トランスポートで返答する内容も作成します。返答内容としては２項目あり、転送処理の結果を表す dCSWStatus と、データ転送エラーバイト数を表す dCSWSDataResidue があります。

本サンプルプログラムでは、この２項目を作成するために、

- CBWパケットのdCSWDataTransferLengthフィールド
- CSWパケットのdCSWDataTransferResidueフィールド

を使用します。

CBW パケットの dCSWDataTransferLength フィールドはホスト PC が指定するデータトランスポートで扱うデータバイト数を入れる変数として使用します。

CSW パケットの dCSWDataTransferResidue フィールドはファンクションがデータトランスポートで扱うデータバイト数を入れる変数として使用します。

CBW トランスポートが終了すると、dCSWDataTransferLength フィールドと dCSWDataTransferResidue フィールドにはデータトランスポートで扱う予定データバイト数がそれぞれ格納されます。

データトランスポートでデータ転送する際にはフロー図で示した流れで動作を行います。

ホスト-ファンクション間でエラーなく処理が行われる時は、データトランスポートでデータ転送する度に dCSWDataTransferLength フィールドと dCSWDataTransferResidue フィールドの値を転送バイト数分減算します。それ以外の場合は、PC が要求するデータトランスポートで扱うデータバイト数とファンクションがデータトランスポートで扱ったデータバイト数の「差」を、CSW パケットの dCSWDataTransferResidue フィールドに設定し、ステータストランスポートに移行します。

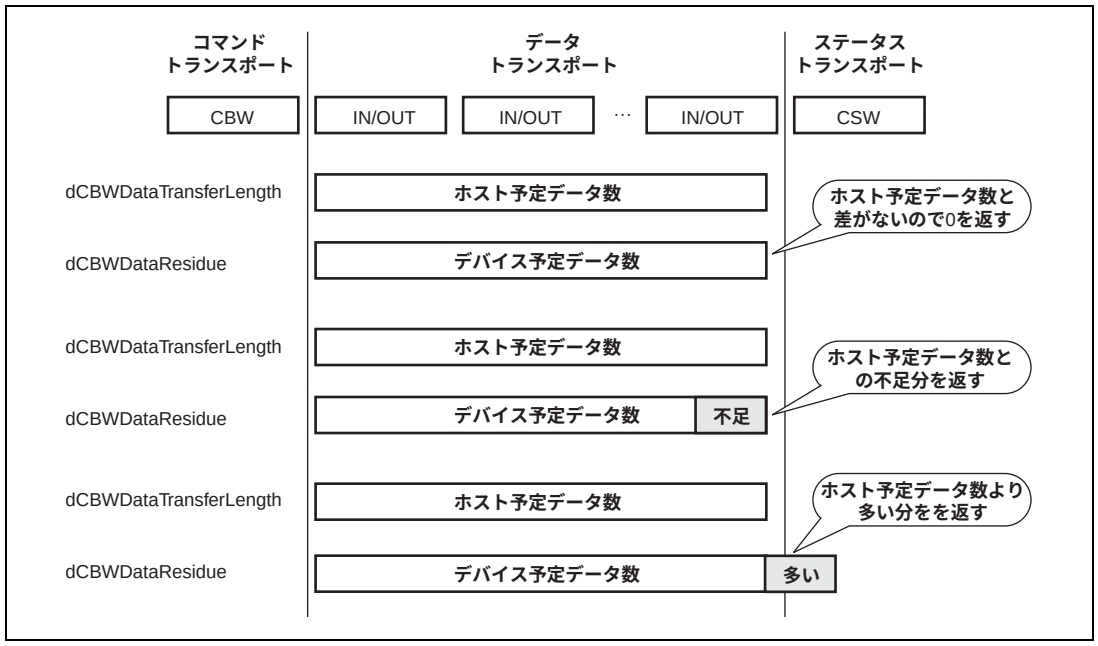


図 4.9　Bulk-Only Transport **における各ステージ**

5. サンプルプログラムの動作

この章ではサンプルプログラムの動作を、USB ファンクションモジュールの動作と関連付けて説明します。

5.1 メインループ

マイコンがリセット状態になると、CPU の内部状態と内蔵周辺モジュールのレジスタが初期化されます。次に StartUp.c の関数 SetPowerOnSection が呼び出され、CPU の初期化をします。**図** 5.1 に SetPowerOnSection のフローチャートを示します。

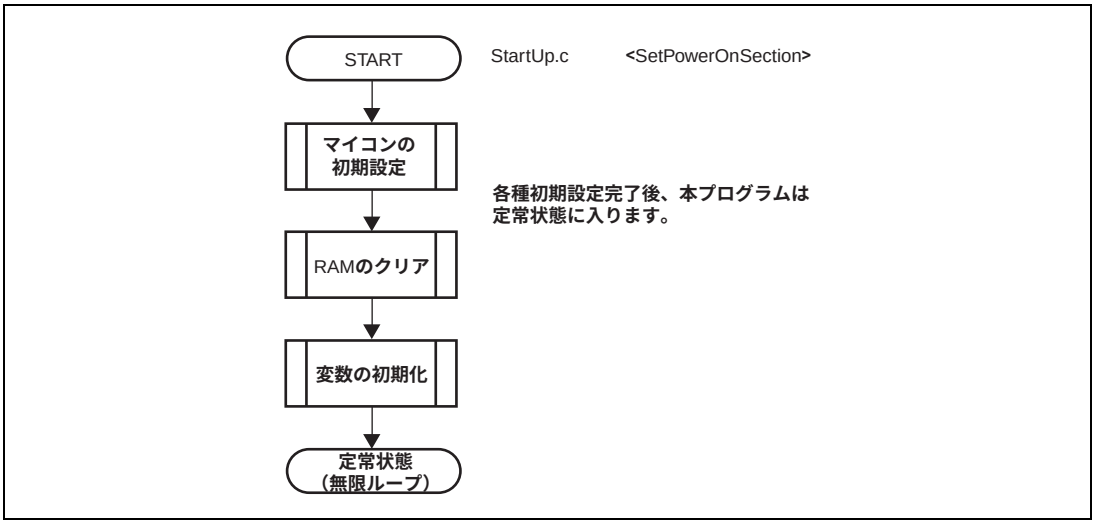
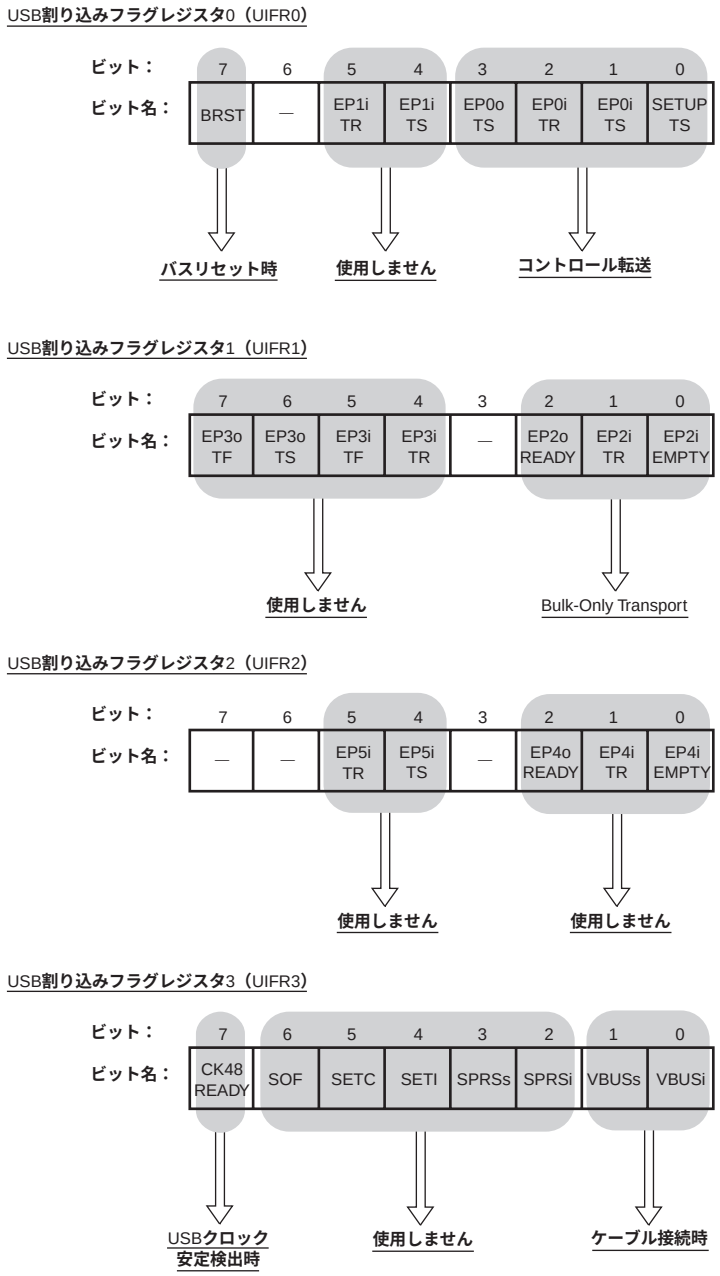


図 5.1 **メインループ**

5.2 割り込みの種類

4 章で説明したように本サンプルプログラムで使用する割り込みは割り込みフラグレジスタ 0～3（UIFR0～3）によって示される計 9 種類です。割り込み要因が発生すると、割り込みフラグレジスタの対応するビットに 1 がセットされ、CPU に対して EXIRQ0 割り込みを要求します。サンプルプログラムでは、この割り込み要求によって割り込みフラグレジスタをリードし、それに対応する USB 通信を行います。**図** 5.2 に割り込みフラグレジスタと、USB 通信の関係を示します。



【注】 サンプルプログラムはインタラプト転送、アイソクロナス転送はサポートしていません。

図 5.2 **割り込みフラグの種類**

5.2.1 各転送への分岐方法

サンプルプログラムでは、USB モジュールからの割り込みの種類によって転送方式を決定しています。各転送方式への分岐は、UsbMain.c の BranchOfInt が実行します。**表 5.1** に割り込みの種類と、BranchOfInt が呼び出す関数の関係を示します。

表 5.1 割り込みの種類と分岐先関数

レジスタ名	ビット	ビット名	呼び出す関数名
UIFR0	7	BRST	ActBusReset
	6	−	−
	5	EP1i TR	−
	4	EP1i TS	−
	3	EP0o TS	ActControlIn、ActControlOut
	2	EP0i TR	ActControlOut
	1	EP0i TS	ActControlIn、ActControlOut
	0	SETUP TS	ActControl
UIFR1	7	EP3o TF	−
	6	EP3o TS	−
	5	EP3i TF	−
	4	EP3i TR	−
	3	−	−
	2	EP2o READY	ActBulkOut
	1	EP2i TR	ActBulkOut
	0	EP2i EMPTY	ActBulkOut
UIFR3	7	CK48 READY	SetUSBModule
	6	SOF	−
	5	SETC	−
	4	SETI	−
	3	SPRSs	−
	2	SPRSi	−
	1	VBUSs	−
	0	VBUSi	ActBusVcc

EP0i TS と EP0o TS 割り込みは、コントロールイン、アウト転送の両方で使用します。従って、コントロール転送の方向とステージを管理するために、サンプルプログラムは TRANS_IN、TRANS_OUT、WAIT の 3 つのステートを持っています。詳細は、**「5.4 コントロール転送」**をご覧ください。

H8S/2215 のハードウエアマニュアルには、割り込み発生時の USB ファンクションモジュールの動作と、アプリケーション側の動作概略が示してあります。次節からは、アプリケーション側ファームウエアの詳細を USB の転送方式ごとに説明します。

5.3 USB 動作クロック安定割り込み

このモジュールは USB モジュールストップ解除後、48MHz の USB 動作クロック安定時間を自動的にカウントした後、発生します。割り込み受信後、EndPoint 構成情報を USB エンドポイントインフォメーションレジスタ（UEPIR00_0～22_4 に書き込み、各割り込みを設定して、USB ケーブル接続待ち状態へ移行します。

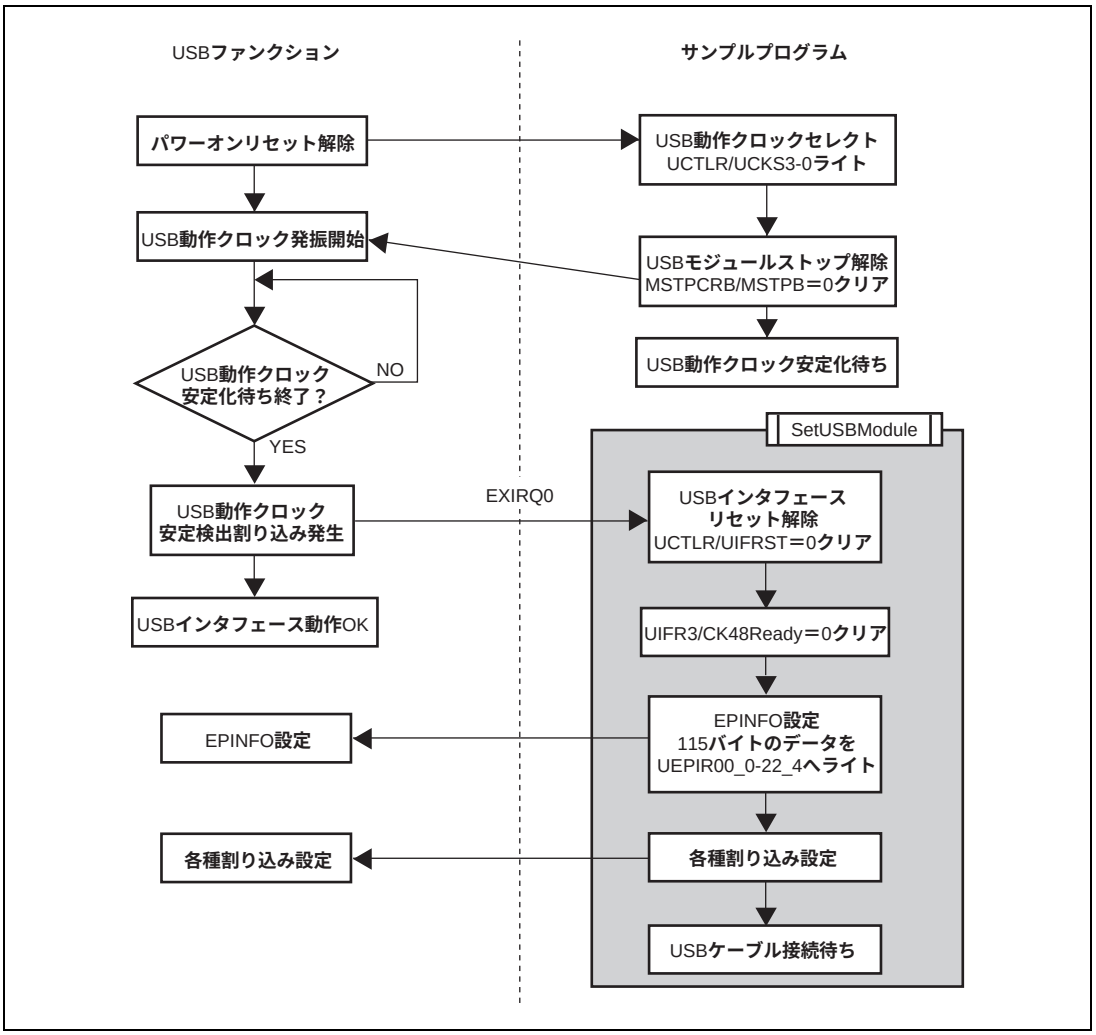


図 5.3 USB 動作クロック安定検出割り込み

5.3.1 EPINFO

H8S/2215搭載のUSBファンクションモジュールは初期化時にエンドポイント構成をソフトウエアにて設定することが可能です。設定可能な転送タイプを次に示します。

- Control転送　　：1系統
- BulkIn転送　　：2系統
- BulkOut転送　　：2系統
- Interrupt In転送　　：1系統
- Isochronous In転送　：1系統
- Isochronous Out転送　：1系統

USBエンドポイントインフォメーションレジスタ（以下UEPIR）を使用し、Control転送以外はEndPoint番号、Interface番号、Alternate番号、MaxPacketSizeの設定が可能です。

表 5.2 に転送タイプと対応するUEPIRを示します。

表 5.2　**転送タイプと** UEPIR **との関係**

転送タイプ	**数**	**対応する** UEPIR
Control **転送**	1	00
Interrupt In **転送**	2	01、22
BulkIn **転送**	2	02、20
BulkOut **転送**	2	03、21
Isochronous In **転送**	1	04,06,08,10,12,14,16,18
Isochronous Out **転送**	1	05,07,09,11,13,15,17,19

H8S/2215ハードウエアマニュアルでは、エンドポイント情報をBluetooth規格に設定した場合でエンドポイント番号を説明しています。

本サンプルプログラムで使用するエンドポイント構成とH8S/2215ハードウエアマニュアル記載のエンドポイント番号を比較したものを**図** 5.4 に示します。

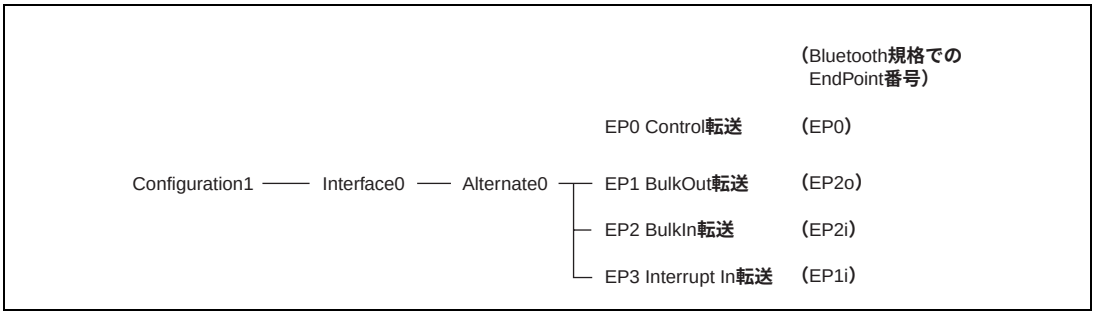


図 5.4　**サンプルプログラムの** EndPoint **構成**

図 5.4　に示す EndPoint 構成を実現する UEPIR00_0～22_4 の設定値を**表** 5.3 に示します。使用しない EndPoint についても必ずダミーデータ 0 をライトしてください。

表 5.3　UEPIR **の設定値**

UEPIR	**設定値** **(16 進数)**	**転送** **タイプ**	EP **番号**	Interface **番号**	Alternate **番号**	MaxPacket Size（Byte）
00	00_00_40_00_00	Control	0	0	0	64
01	34_1C_08_00_01	Interrupt In	3	0	0	8
02	24_14_40_00_02	BulkIn	2	0	0	64
03	14_10_40_00_03	BulkOut	1	0	0	64
04	04_1C_00_00_04	Isochronous In	0	0	0	0
05	04_08_00_00_05	Isochronous Out	0	0	0	0
06	04_1C_00_00_06	Isochronous In	0	0	0	0
07	04_08_00_00_07	Isochronous Out	0	0	0	0
08	04_1C_00_00_08	Isochronous In	0	0	0	0
09	04_08_00_00_09	Isochronous Out	0	0	0	0
10	04_1C_00_00_0A	Isochronous In	0	0	0	0
11	04_08_00_00_0B	Isochronous Out	0	0	0	0
12	04_1C_00_00_0C	Isochronous In	0	0	0	0
13	04_08_00_00_0D	Isochronous Out	0	0	0	0
14	04_1C_00_00_0E	Isochronous In	0	0	0	0
15	04_08_00_00_0F	Isochronous Out	0	0	0	0
16	04_1C_00_00_10	Isochronous In	0	0	0	0
17	04_08_00_00_11	Isochronous Out	0	0	0	0
18	04_1C_00_00_12	Isochronous In	0	0	0	0
19	04_08_00_00_13	Isochronous Out	0	0	0	0
20	04_14_00_00_14	BulkIn	0	0	0	0
21	04_10_00_00_15	BulkOut	0	0	0	0
22	04_10_00_00_16	Interrupt In	0	0	0	0

5.4 ケーブル接続時（VBUS）割り込み

USB ファンクションモジュールのケーブルを、ホストコントローラに接続した際に発生します。アプリケーション側はマイコンの初期設定完了後、汎用出力ポートを使用して USB データバスの D+をプルアップします。このプルアップによって、ホストコントローラはデバイスが接続されたことを認識します。（**図** 5.5 参照）

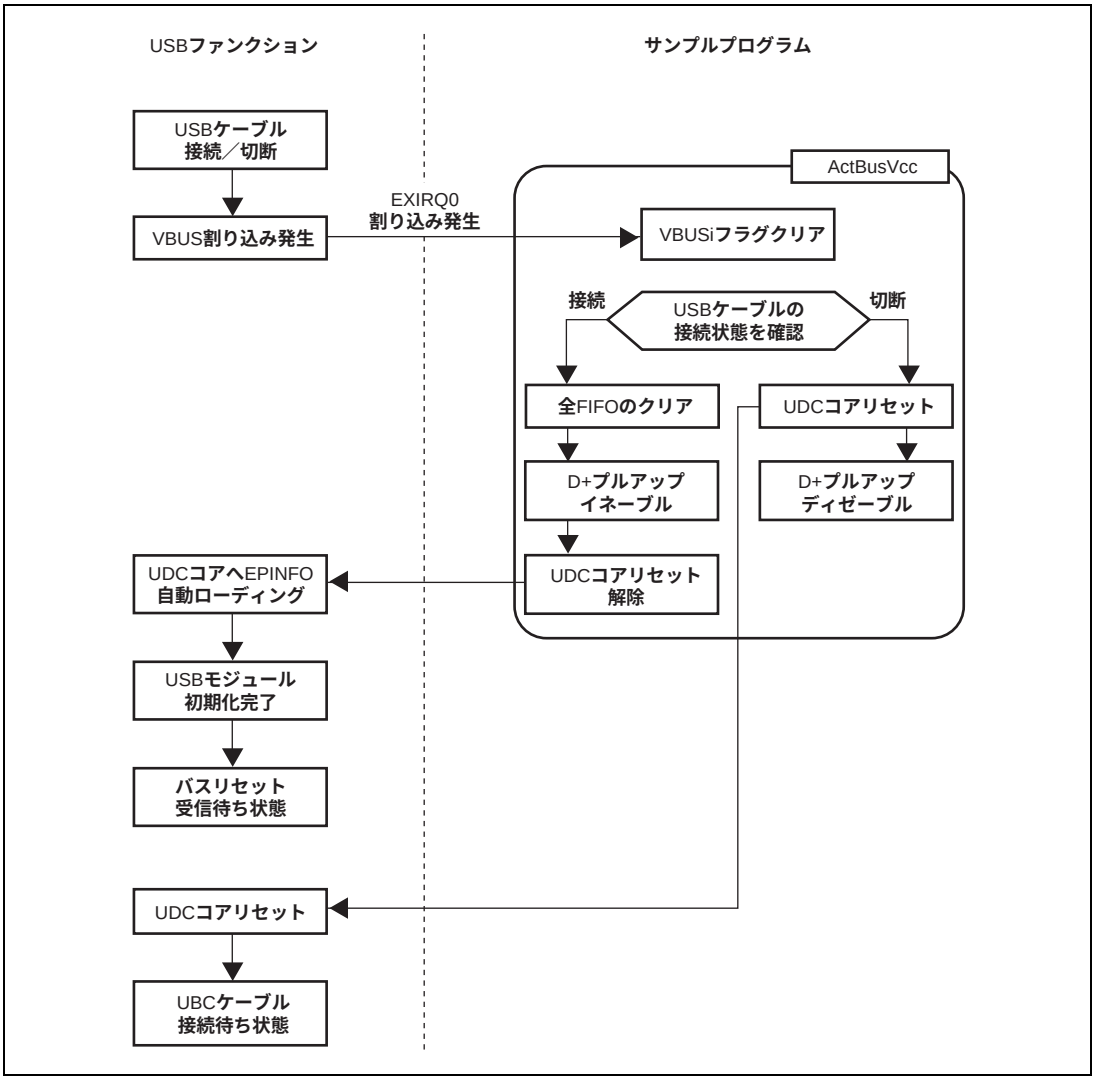


図 5.5 **ケーブル接続時割り込み**

5.5 バスリセット時（BRST）割り込み

ホストコントローラがUSBデータバスにデバイスが接続されたことを認識するとバスリセット信号を出力します。ホストからのバスリセット信号を受信するとバスリセット割り込みが発生します。

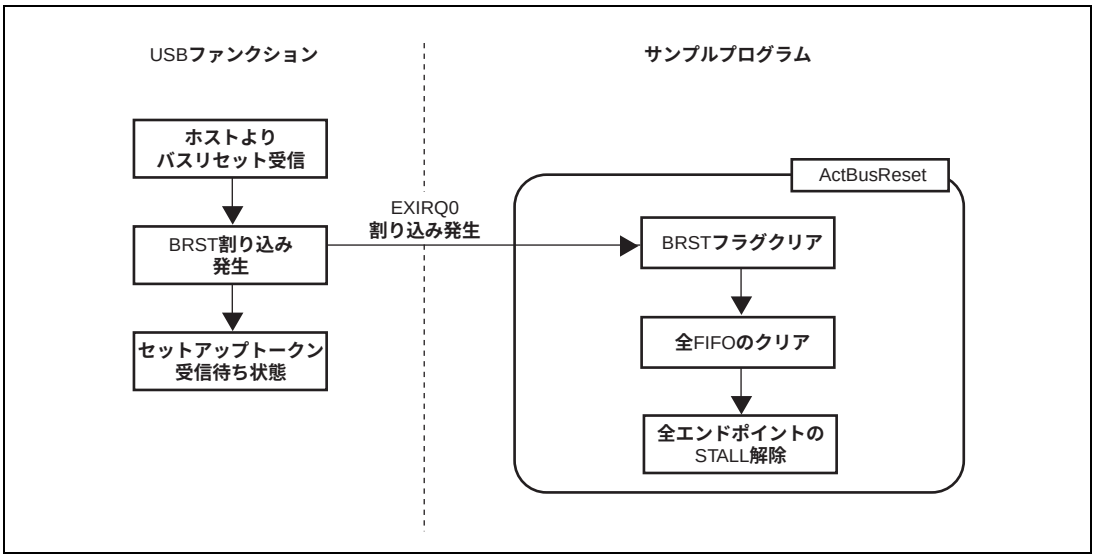


図 5.6 **バスリセット割り込み**

5.6 コントロール転送

コントロール転送には、割り込みフラグレジスタのビット 0～3 を使用します。コントロール転送は、データステージにおけるデータの向きによって、2 つに分けることができます。（**図** 5.7 参照）

データステージにおいて、ホストコントローラから USB ファンクションへデータ転送する場合がコントロールアウト転送、反対の場合がコントロールイン転送です。

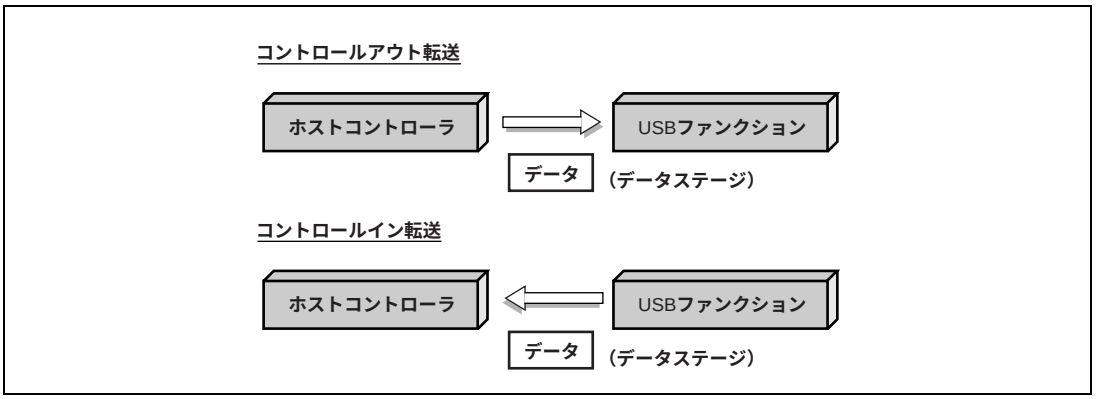


図 5.7 **コントロール転送**

コントロール転送は、セットアップ、データ（ない場合もある）、ステータスの 3 つのステージで構成されます（**図** 5.8）。また、データステージは、複数のバストランザクションで構成されます。

コントロール転送では、データの向きが反転することによってステージが切り替わったことを認識します。したがって同じ割り込みフラグを使用して、コントロールイン転送または、コントロールアウト転送を行う関数を呼び出します（**表** 5.1 参照）。このため、現在イン、アウトどちらのコントロール転送が行われているかをファームウエアがステートによって管理し（**図** 5.8 参照）、適切な関数を呼び出す必要があります。データステージにおけるステート（TRANS_IN、TRANS_OUT）は、セットアップステージで受信するコマンドによって決定します。

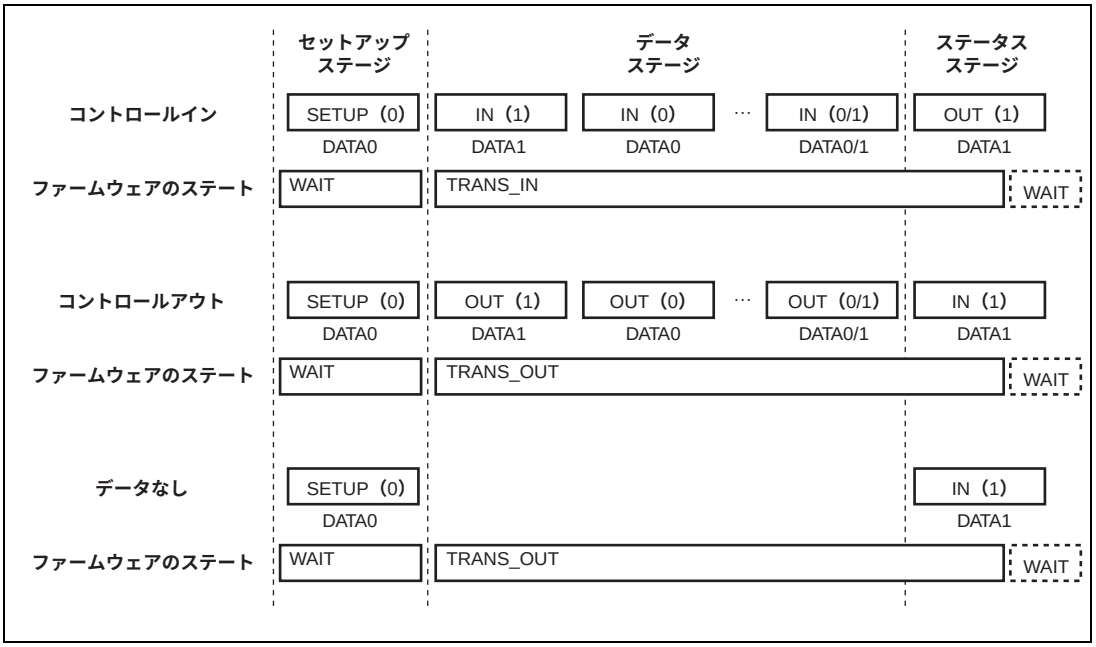


図 5.8　**コントロール転送における各ステージ**

5.6.1　セットアップステージ

セットアップステージでは、ホストとファンクションがコマンドの送受信を行います。コントロールイン転送、コントロールアウト転送共に、ファームウエアのステートは WAIT になります。また発行されるコマンドの種類によって、コントロールイン転送またはアウト転送の区別を行い、データステージにおけるファームウエアのステート（TRANS_IN、TRANS_OUT）を決定します。

- TRANS_INとなるコマンド・・・・・・・GetDescriptor（標準コマンド）
  Get Max LUN（クラスコマンド）
- TRANS_OUTとなるコマンド・・・・・・Bulk-Only Mass Storage Reset（クラスコマンド）

図 5.9 にセットアップステージにおけるサンプルプログラムの動作を示します。図の左側は、USB ファンクションモジュールの動作を示しています。

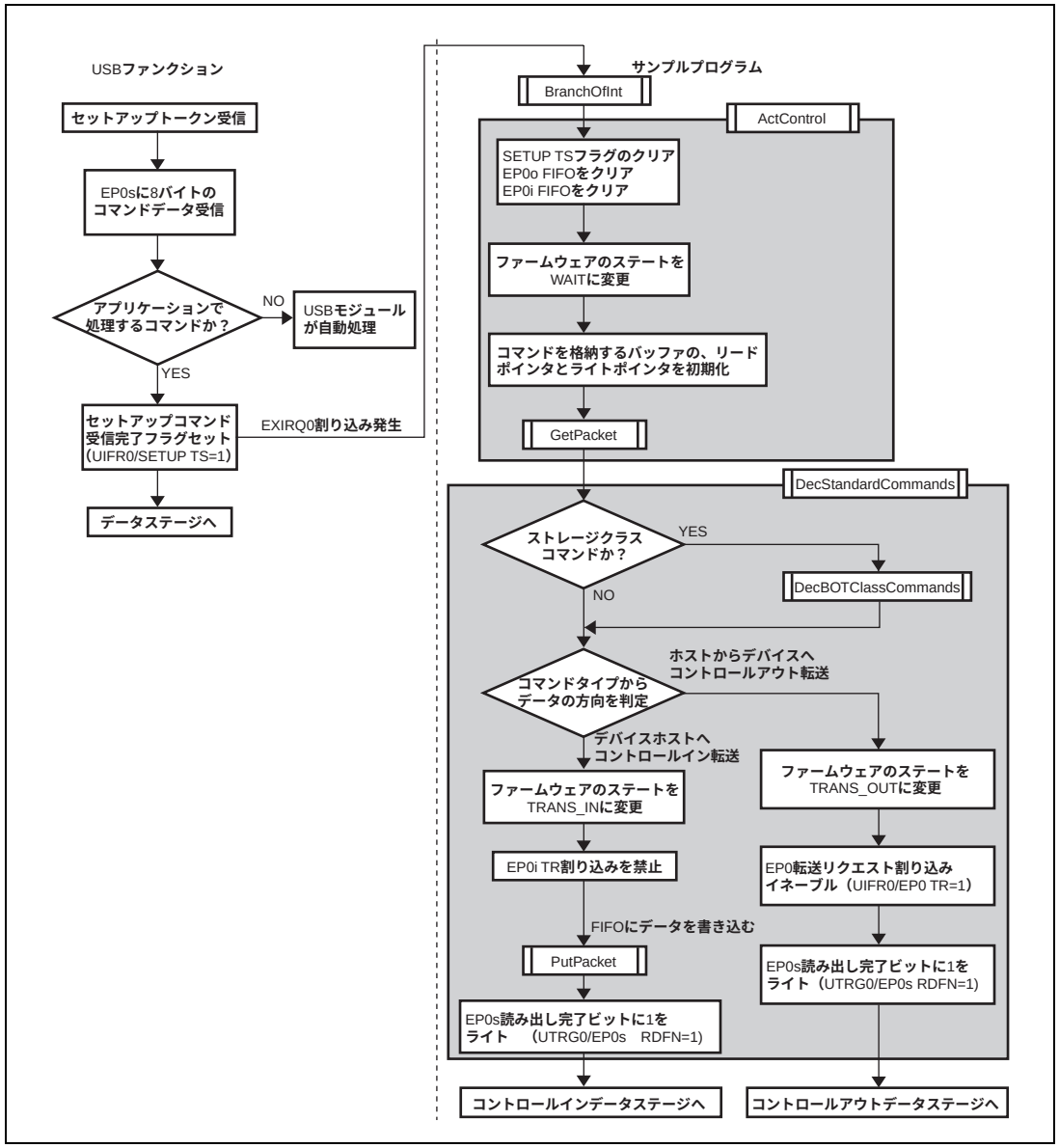


図 5.9 **セットアップステージ**

5.6.2 データステージ

データステージでは、ホストとファンクションがデータの送受信を行います。ファームウエアのステートは、セットアップステージで行ったコマンドのデコード結果によって、コントロールイン転送の場合は TRANS_IN に、コントロールアウト転送の場合は TRANS_OUT になります。**図** 5.10、**図** 5.11 にコントロール転送のデータステージにおけるサンプルプログラムの動作を示します。

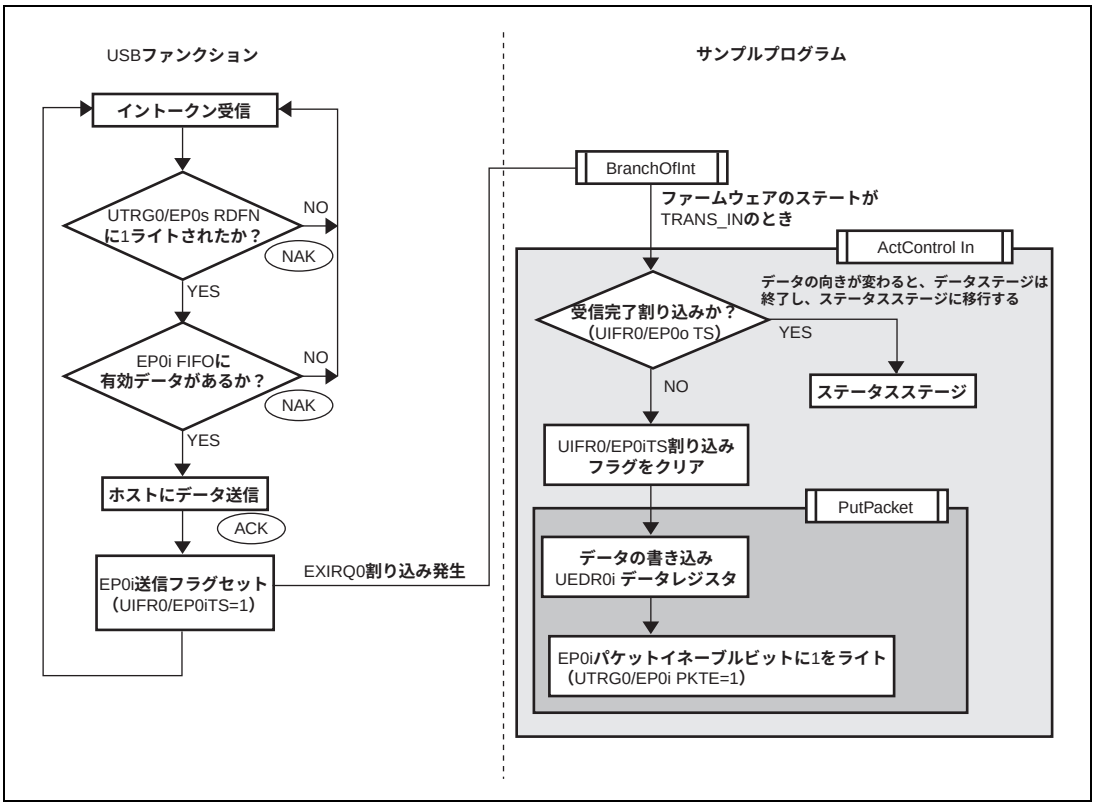


図 5.10 **データステージ（コントロールイン転送）**

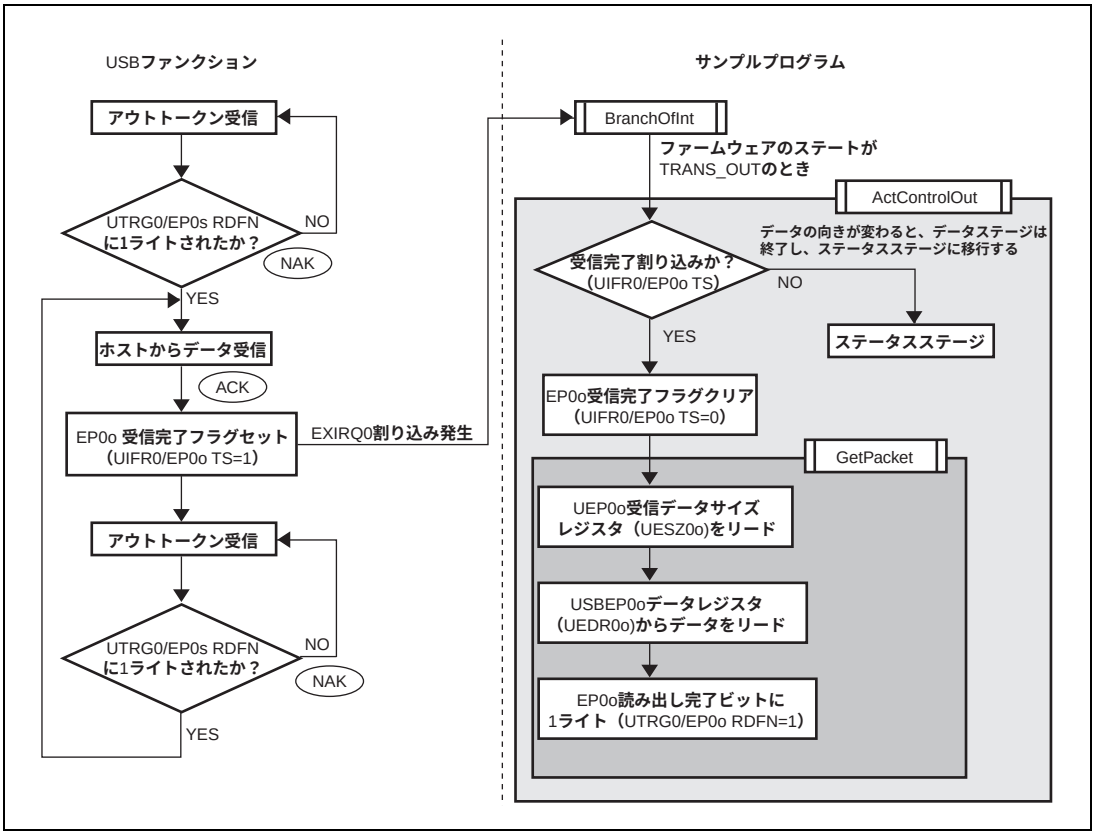


図 5.11　**データステージ（コントロールアウト転送）**

5.6.3 ステータスステージ

ステータスステージは、データステージと反対方向のトークンによって開始されます。つまり、コントロールイン転送では、ホストコントローラからのアウトトークンによってステータスステージが開始され、コントロールアウト転送では、ホストコントローラからのイントークンによってステータスステージが開始されます。

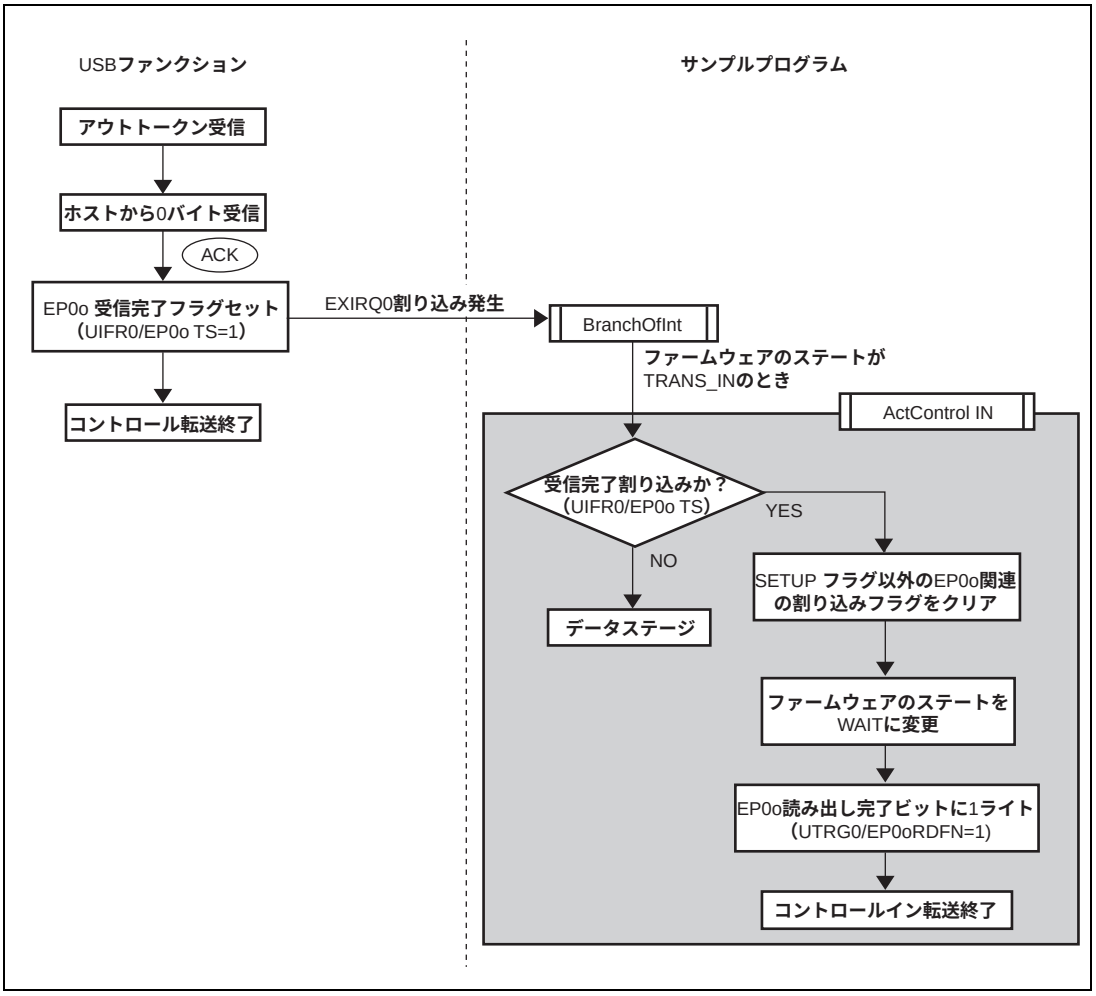


図 5.12 **ステータスステージ（コントロールイン転送）**

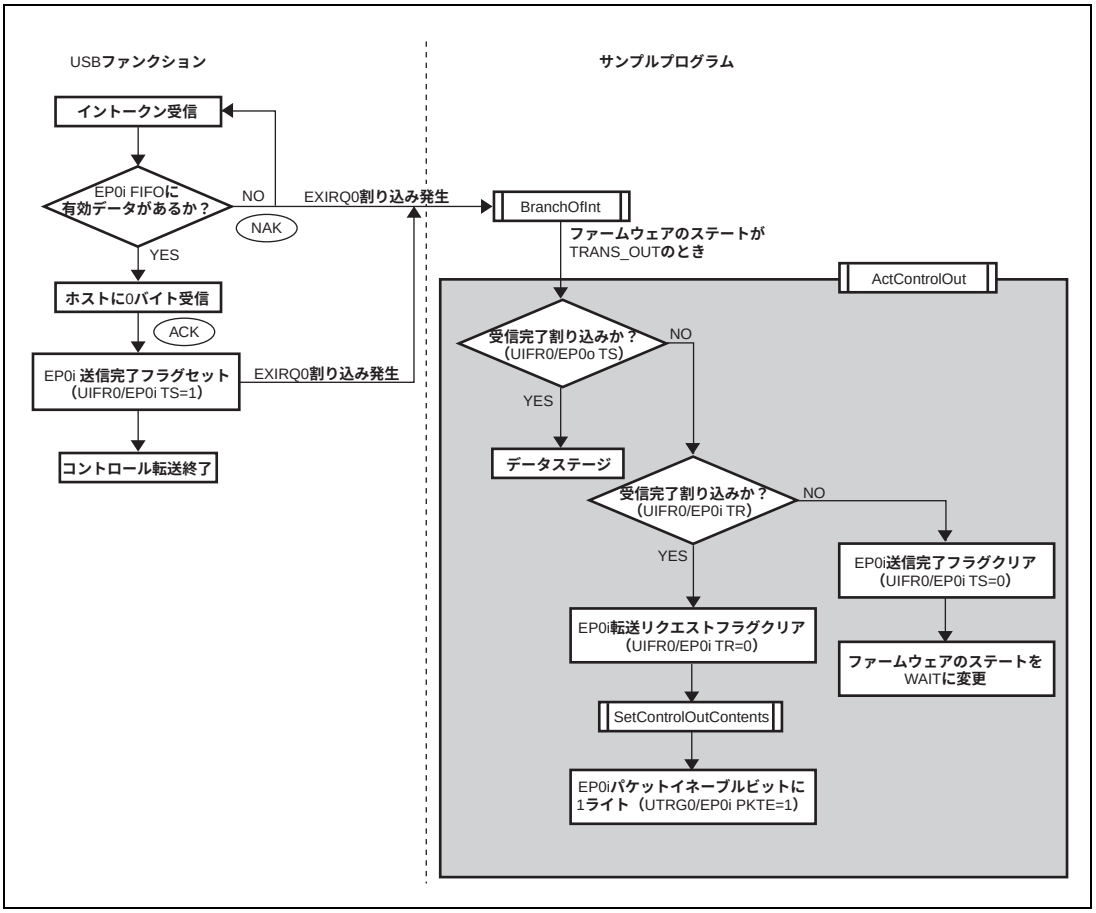


図 5.13　**ステータスステージ（コントロールアウト転送）**

5.7 バルク転送

バルク転送には、割り込みフラグレジスタ 1 のビット 0～2 を使用します。バルク転送もデータを送信する向きによって、2 つに分けることができます。（**図** 5.14 参照）

ホストコントローラから USB ファンクションへデータ転送する場合をバルクアウト転送、反対の場合をバルクイン転送と呼びます。

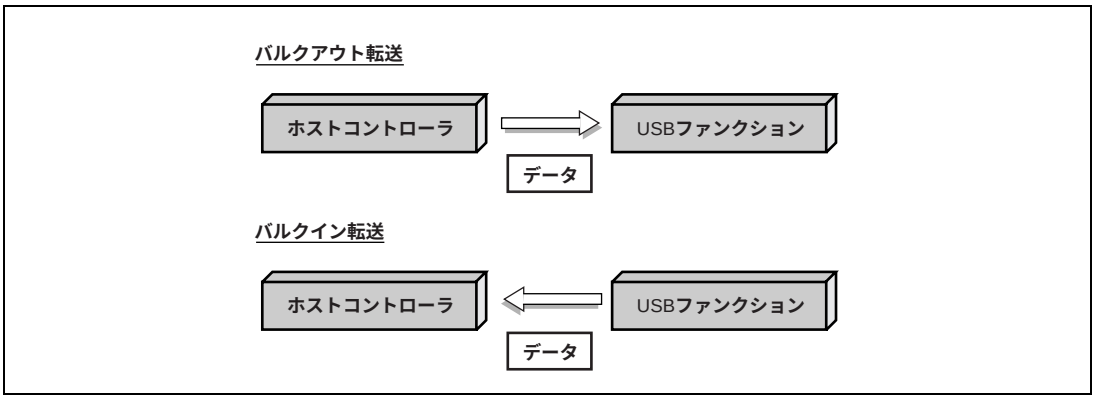


図 5.14 **バルク転送**

USB Mass Storage Class の Bulk-Only Transport は、バルクイン転送とバルクアウト転送で構成されています。

Bulk-Only Transport は、「コマンドトランスポート（CBW）」「データトランスポート」（ない場合もある）「ステータストランスポート（CSW）」の 2 ないし 3 つのステージで構成されます（**図** 5.15）。また、データトランスポートは、複数のバストランザクションで構成されます。

Bulk-Only Transport では、コマンドトランスポート（CBW）はバルクアウト転送、ステータストランスポート（CSW）はバルクイン転送、データトランスポートはデータを送信する向きによってバルクイン転送、バルクアウト転送どちらかの転送が行われます。

データトランスポートでバルクイン、バルクアウトどちらの転送が行われるかは、コマンドトランスポートで受信する CBW データにより決定します。ファームウエアはデータトランスポートがバルクイン転送、バルクアウト転送どちらの転送が行われるかをステート（TRANS_IN、TRANS_OUT）で管理し（**図** 5.15 参照）、適切な関数を呼び出す必要があります。

また、データトランスポートからステータストランスポートへのステージの遷移は、ホスト PC がリクエストしたデータトランスポートでの予定データ長のデータを送信または受信することで、ステータストランスポートへステージが遷移します。したがって、ファームウエアがデータトランスポートで送信または受信したデータ長を管理し、ステージの遷移後ステータストランスポートでデータをホスト PC に送信する必要があります。

もし、コマンドトランスポートで受信する CBW データが有効と認められない場合、エンドポイントをストールし一切のバルク転送を行いません。

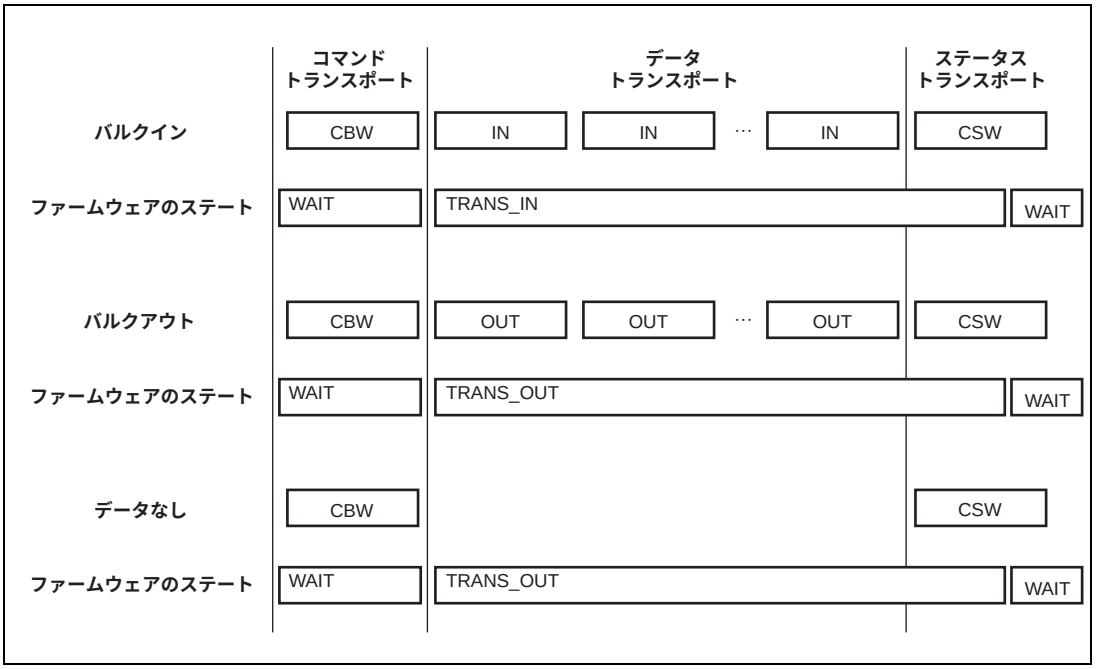


図 5.15　Bulk-Only Transport **における各ステージ**

5.7.1　　コマンドトランスポート

コマンドトランスポートでは、ホストからファンクションへの CBW データを受信します。

この時ファームウエアのステートは WAIT 状態で実行されます。CBW データ受信後このステージでは、次に示す 5 点の処理を行います。

1. CBWデータをEP1データレジスタからワーク領域に格納。
2. CBWデータの有効判定。
3. CSWデータの準備。
4. CBWデータの内容をデコードし、データトランスポートで転送するデータがある場合は、データの準備を行う。（関数DecBotCmd内にて処理）
5. データトランスポートがバルクインまたはバルクアウト転送どちらかの区別を行い、ファームウエアのステート（TRANS_IN、TRANS_OUT）を決定。

図 5.16 にサンプルプログラムのコマンドトランスポートにおける動作を示します。図の左側は、USB ファンクションモジュールの動作を示しています。

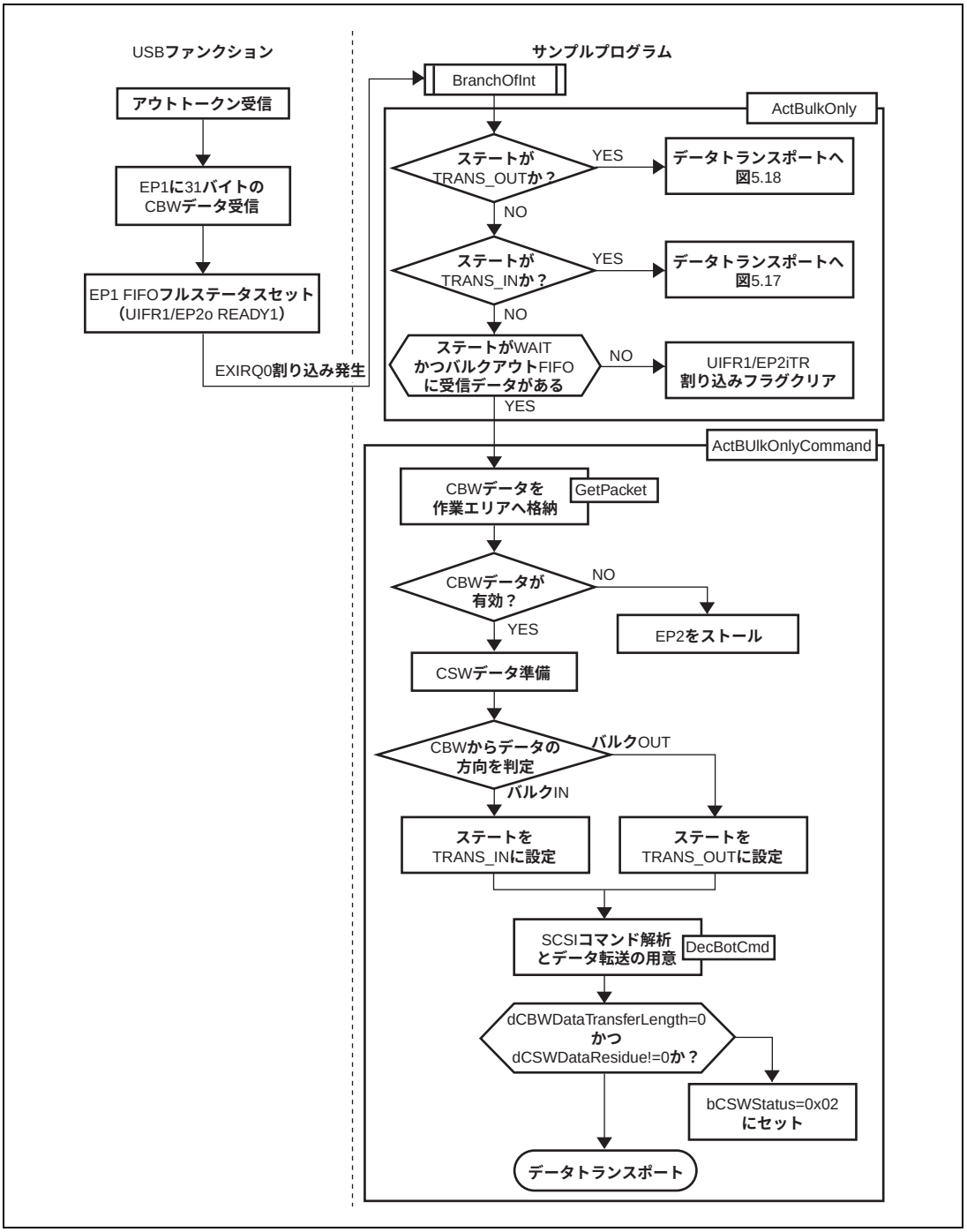


図 5.16　**コマンドトランスポート**

5.7.2 データトランスポート

データトランスポートでは、ホストとファンクションがデータの送受信を行います。

この時ファームウエアのステートは TRANS_IN または TRANS_OUT どちらかの状態で実行されます。

ファームウエアのステートが TRANS_IN 状態の場合（バルクイン転送）、次に示す 3 点の処理を行います。

1. ファンクションからホストへ向けデータ送信処理。
2. ホストが予定したデータ長に対しファンクションが送信するデータ長が短い場合の0付加処理。
3. CSWで送信する情報の作成。

サンプルプログラムのデータトランスポート（バルクイン転送）における動作を**図 5.14** に示します。図の左側は、USB ファンクションモジュールの動作を示しています。

このサンプルソフトでは、ホストが要求するデータ長に対しファンクションが送信するデータ長が短い場合、USB Mass Storage Class の Bulk-Only Transport に記載されている通り、ファンクションが送信するデータの後に 0 を付加しホストが要求する長さのデータを送信後、ステータストランスポートにおいて 0 を何バイト付加したかを報告しています。

この動作を行うために、CBW データの dCBWDataTransferLength、CSW データの dCSWDataResidue、CSW データの bCSWStatus をグローバル変数として使用しています。

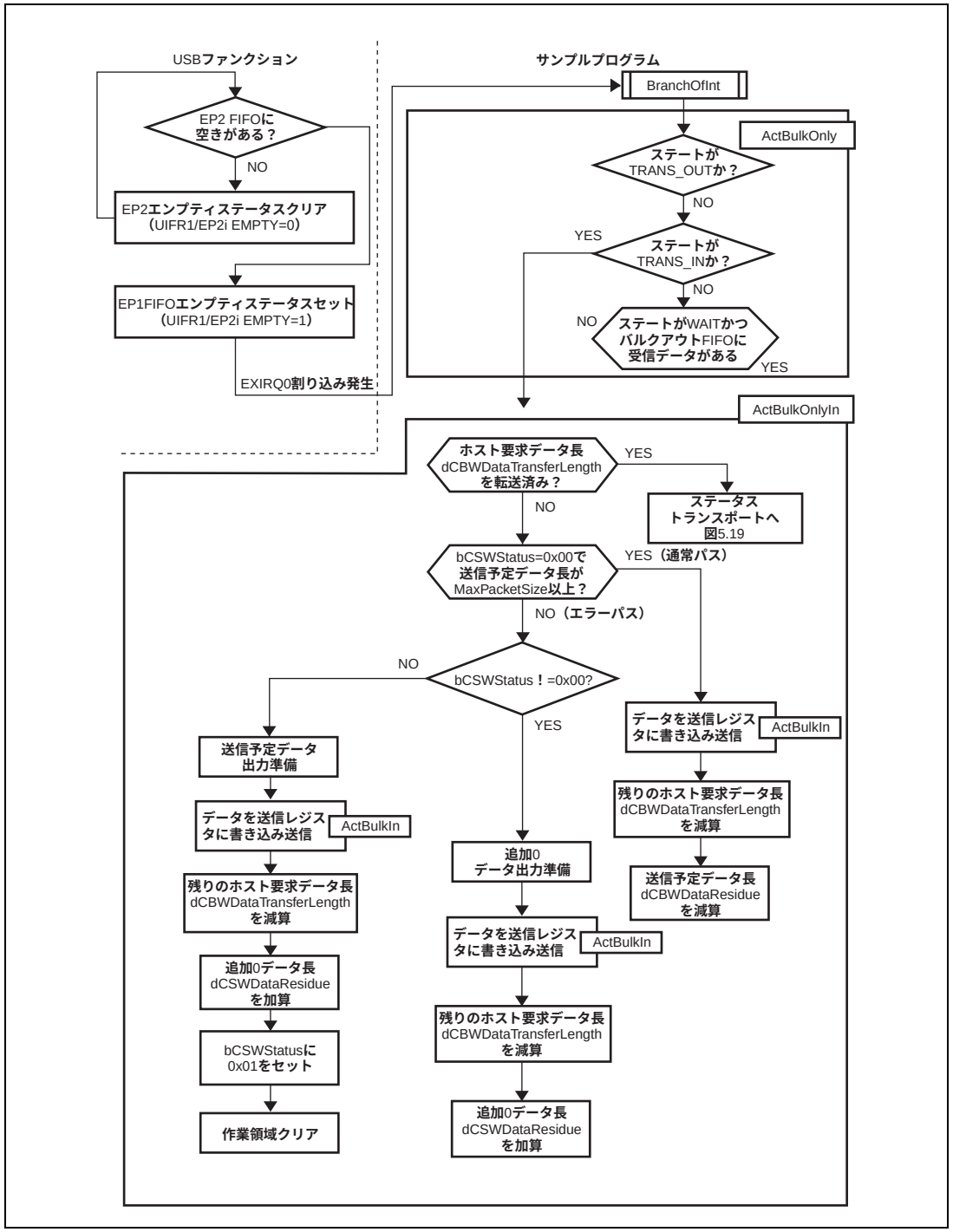


図 5.17　**データトランスポート（バルクイン転送）**

サンプルプログラムのデータトランスポート（バルクアウト転送）における動作を**図** 5.18 に示します。図の左側は、USB ファンクションモジュールの動作を示しています。

ファームウエアのステートが TRANS_OUT 状態の場合（バルクアウト転送）、次に示す 3 点の処理を行います。

1. ホストからファンクションに対するデータ受信処理。
2. データ長演算処理
3. CSWで送信する情報の作成。

このサンプルソフトでは、ホストが予定したデータ長に対しファンクションが受信したデータ長が短い場合、USB Mass Storage Class の Bulk-Only Transport に記載されている通り、データトランスポートにおいてファンクションが受信する不足データ長をステータストランスポートにおいて報告しています。

この動作を行うために、CBW データの dCBWDataTransferLength と、CSW データの dCSWDataResidue をグローバル変数として使用しています。

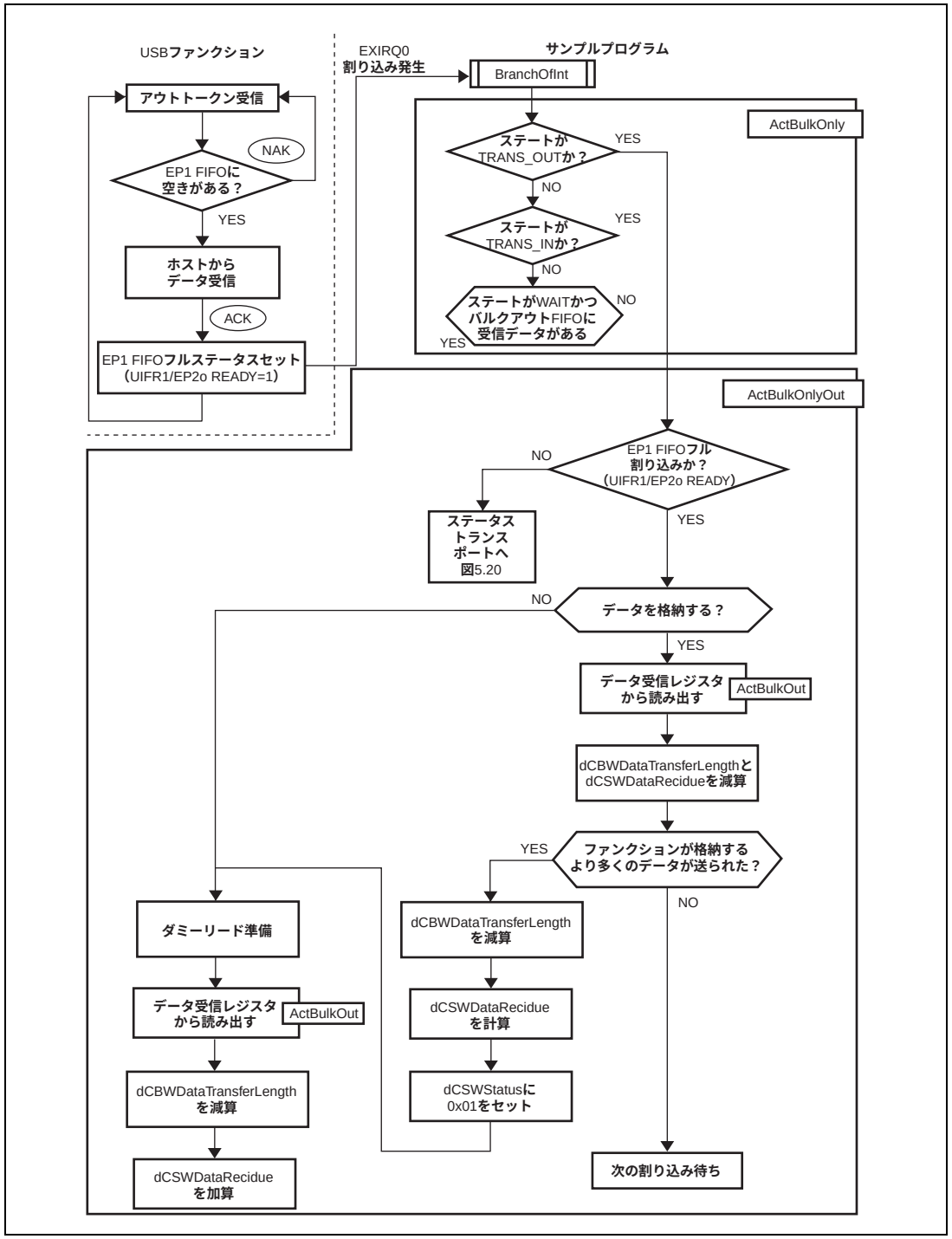


図 5.18　**データトランスポート（バルクアウト転送）**

5.7.3 ステータストランスポート

ステータストランスポートでは、ファンクションからホストに対しデータの送信を行います。

この時ファームウエアのステートは TRANS_IN または TRANS_OUT どちらかの状態で実行されます。ファームウエアのステートが TRANS_IN 状態の場合（バルクイン転送）、次に示す 4 点の処理を行います。

1. EP2エンプティステータス割り込みの禁止。
2. CSWデータの転送準備。
3. CSWデータ発行。
4. ファームウエアのステートをWAITに設定。

図 5.19 にサンプルプログラムのステータストランスポート（データトランスポートバルクイン転送）における動作を示します。図の左側は、USB ファンクションモジュールの動作を示しています。

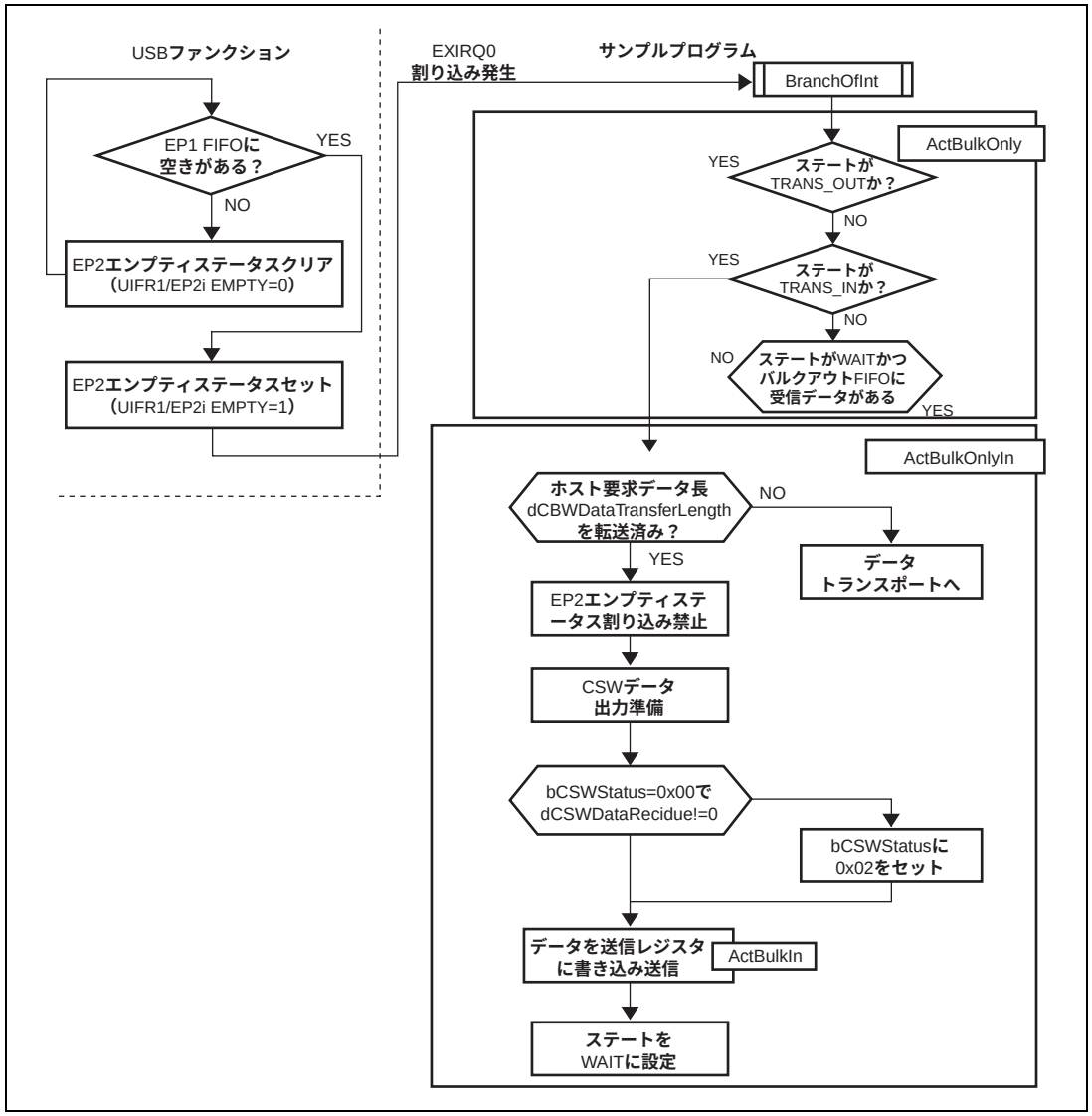


図 5.19　**ステータストランスポート（データトランスポートバルクイン転送）**

サンプルプログラムのステータストランスポート（データトランスポートバルクアウト転送）における動作を**図 5.20** に示します。図の左側は、USB ファンクションモジュールの動作を示しています。

ファームウエアのステートが TRANS_OUT 状態の場合（バルクアウト転送）、次に示す 4 点の処理を行います。

1. CSWデータの転送準備。
2. 受信不足データチェック。
3. CSWデータ発行。
4. ファームウエアのステートをWAITに設定。

このサンプルソフトでは、ホストが予定したデータ長に対しファンクションが受信したデータ長が短い場合、USB Mass Storage Class の Bulk-Only Transport に記載されている通り、データトランスポートにおいてファンクションが受信する不足データ長をステータストランスポートにおいて報告する動作を行うために、ファンクションが受信する不足データ長をチェックし不足発生時は CSW データの bCSWStatus の値を 0x02（フェーズエラー）に設定します。

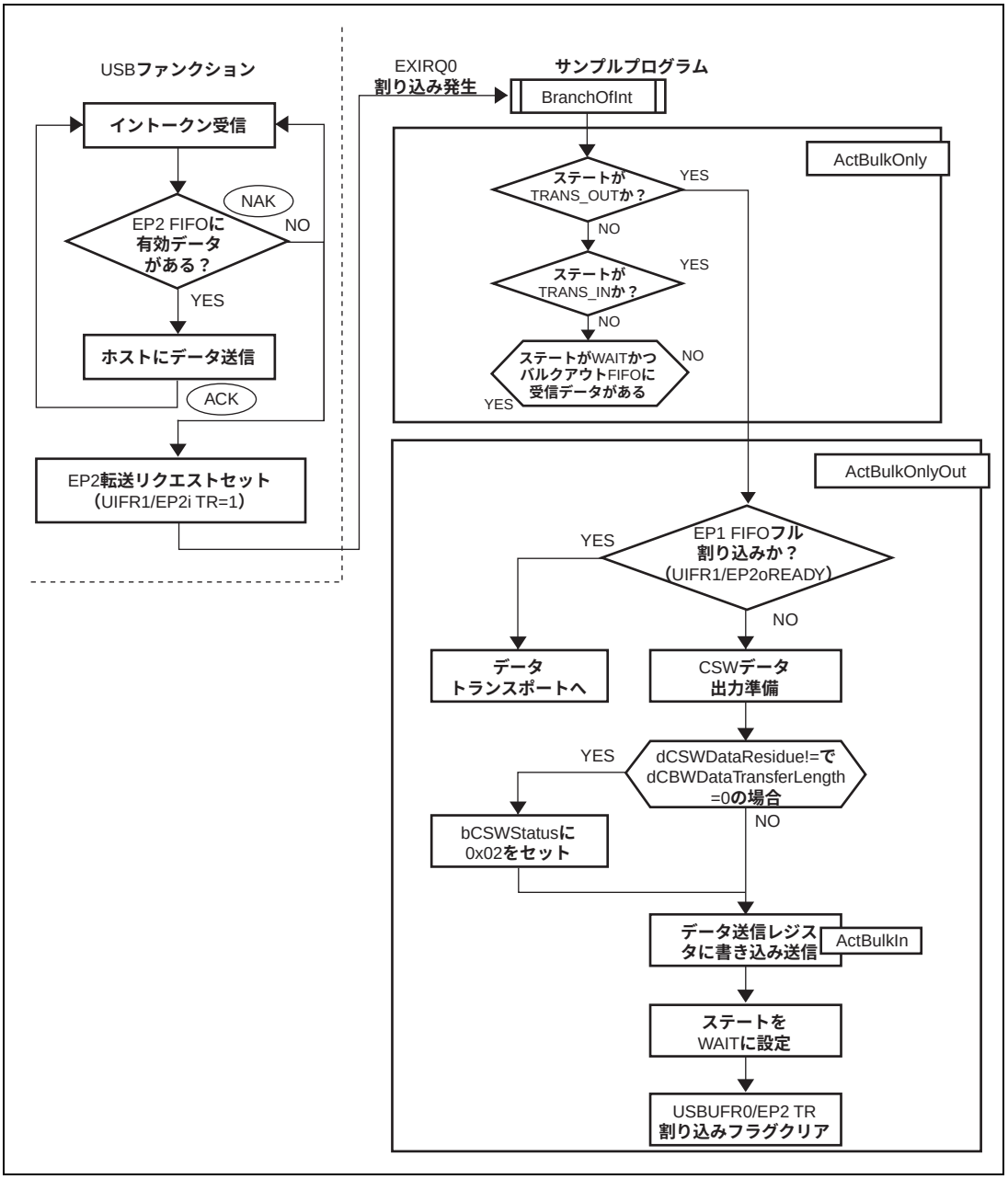


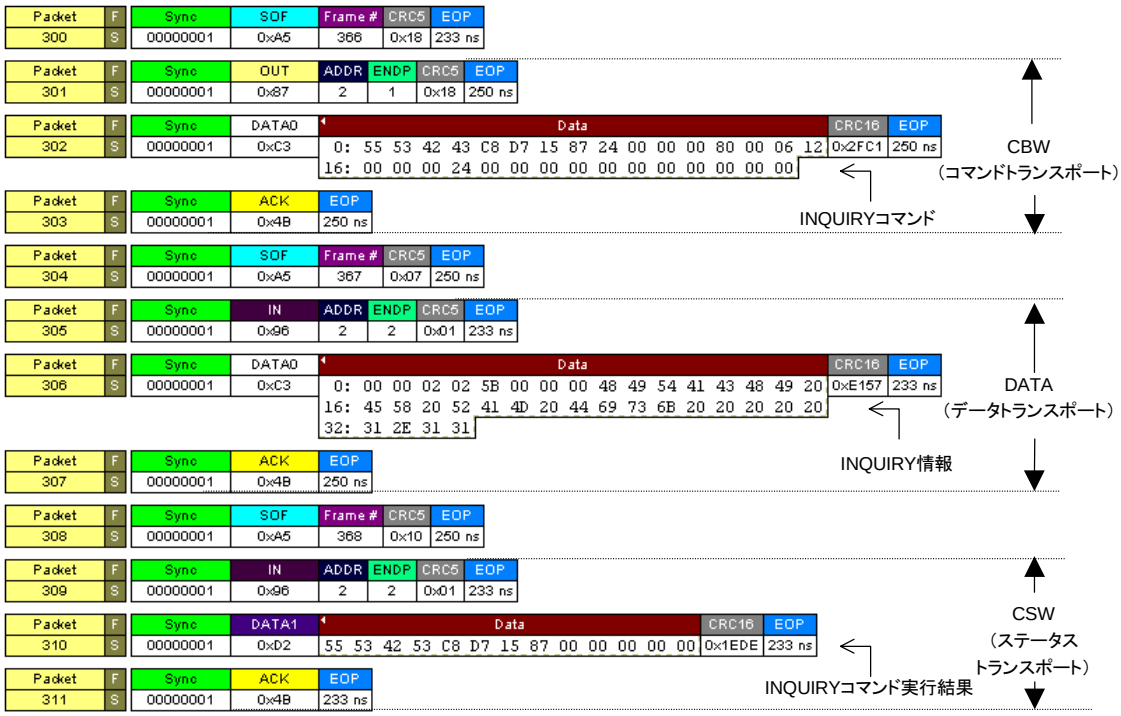
図 5.20　**ステータストランスポート（データトランスポートバルクアウト転送）**

6. アナライザのデータ

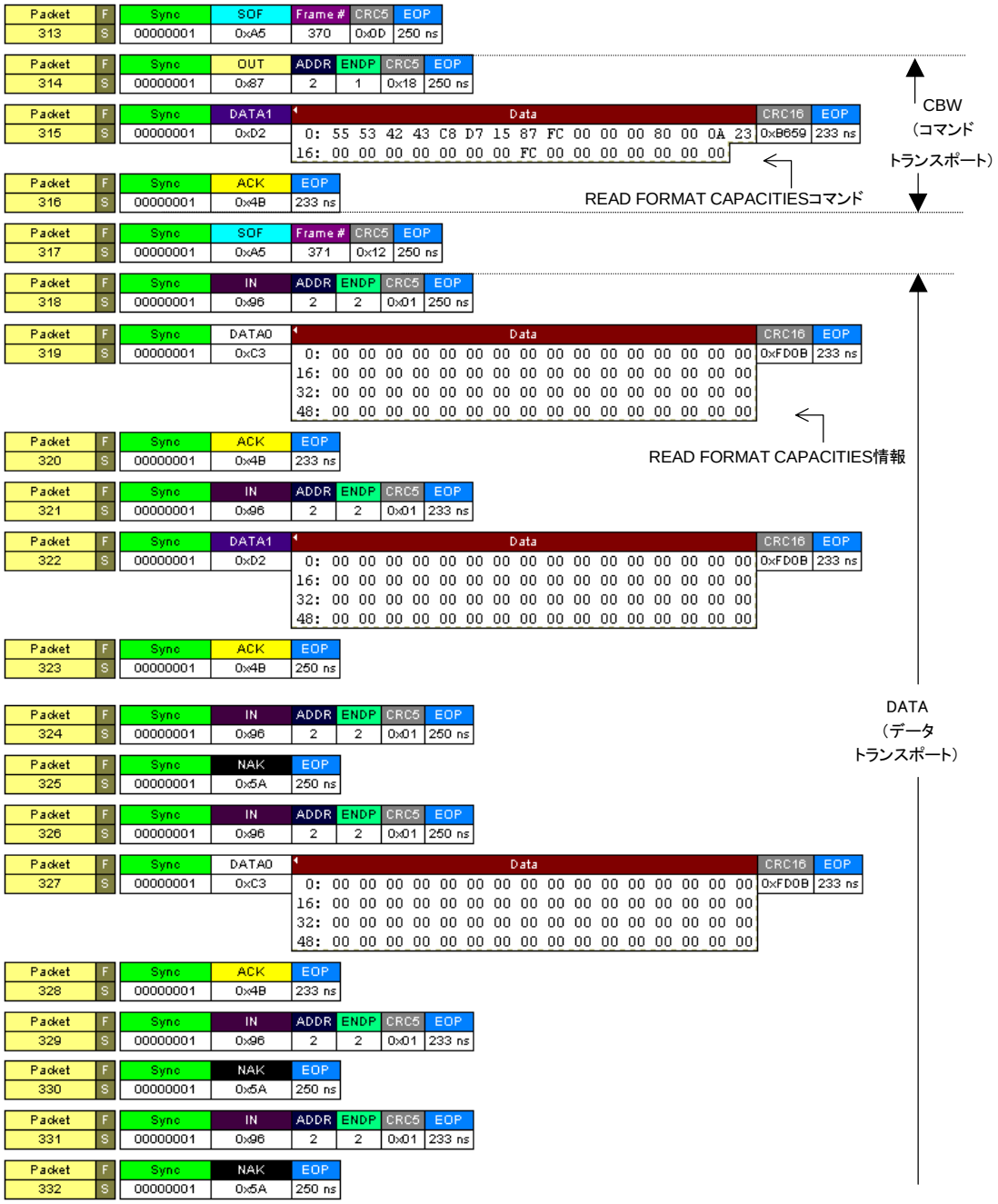
この章では、H8S/2215 内蔵 USB ファンクションモジュールを使用して、CATC 社製 USB プロトコルアナライザ「USB Inspector」（国内：（株）東陽テクニカ（http://www.toyo.co.jp/））を用いた測定を行い、実際にバスを流れているデータについて説明します。なお、パケットの詳細につきましては 2.3 章をご参照下さい。

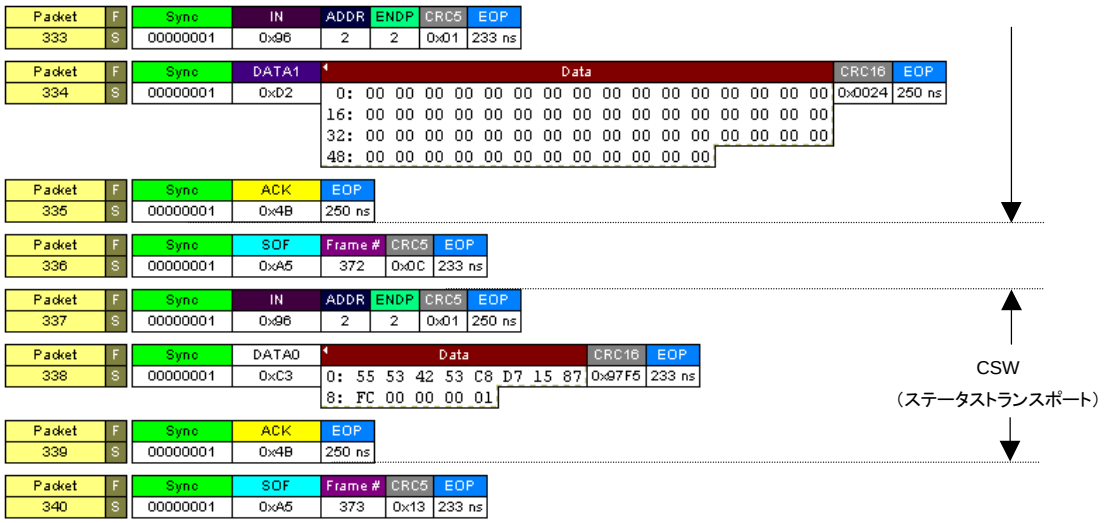
【注】各パケットの前部にある「Packet」は測定時のパケット通し番号です。

- INQUIRY**コマンド**

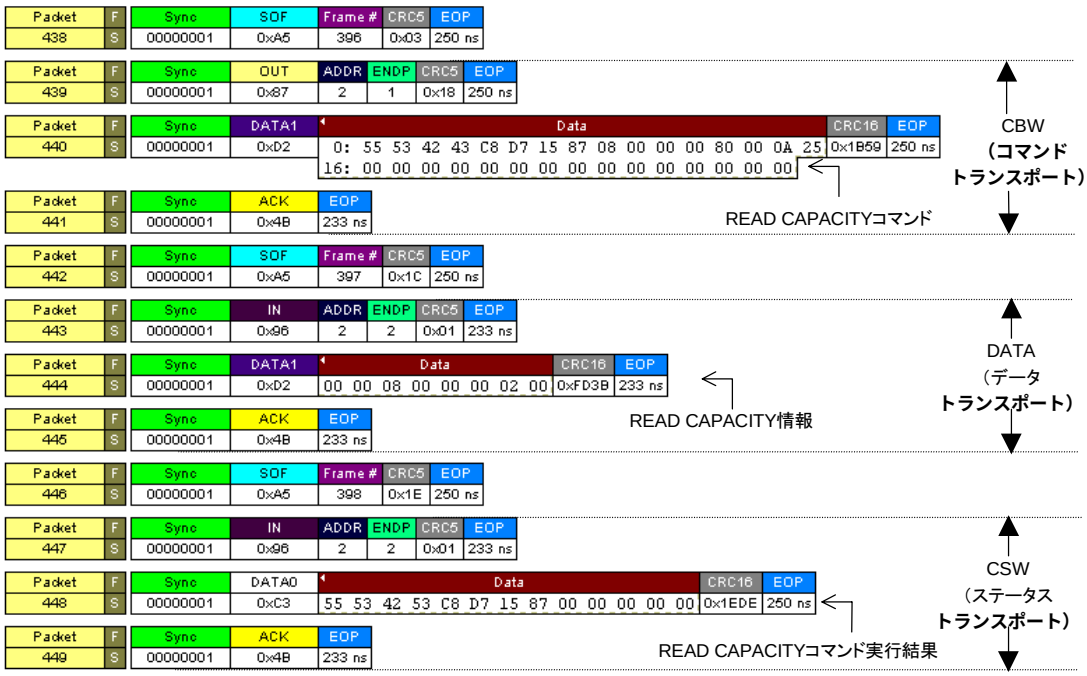


- READ FORMAT CAPACITIES**コマンド**

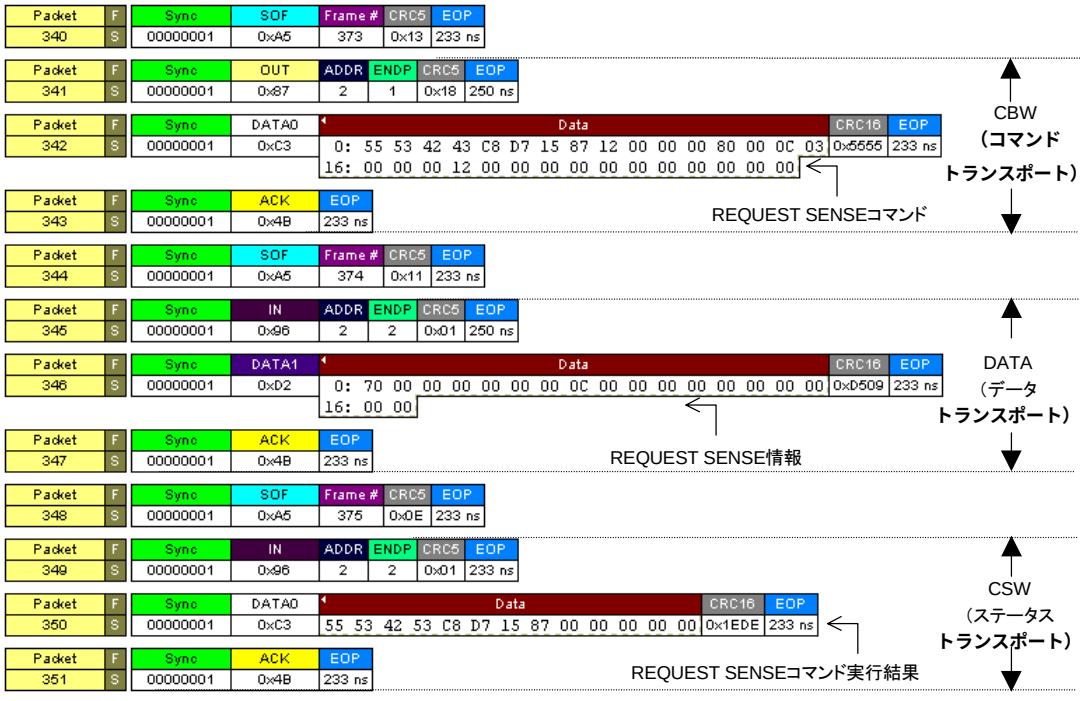



Packet 333 F S Sync 00000001 IN 0x96 ADDR 2 ENDP 2 CRC5 0x01 EOP 233 ns
Packet 334 F S Sync 00000001 DATA1 0xD2 Data 0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CRC16 0x0024 EOP 250 ns
Packet 335 F S Sync 00000001 ACK 0x4B EOP 250 ns
Packet 336 F S Sync 00000001 SOF 0xA5 Frame # 372 CRC5 0x0C EOP 233 ns
Packet 337 F S Sync 00000001 IN 0x96 ADDR 2 ENDP 2 CRC5 0x01 EOP 250 ns
Packet 338 F S Sync 00000001 DATA0 0xC3 Data 0: 55 53 42 53 C8 D7 15 87 8: FC 00 00 00 01 CRC16 0x97F5 EOP 233 ns
Packet 339 F S Sync 00000001 ACK 0x4B EOP 250 ns
Packet 340 F S Sync 00000001 SOF 0xA5 Frame # 373 CRC5 0x13 EOP 233 ns
CSW
（ステータストランスポート）

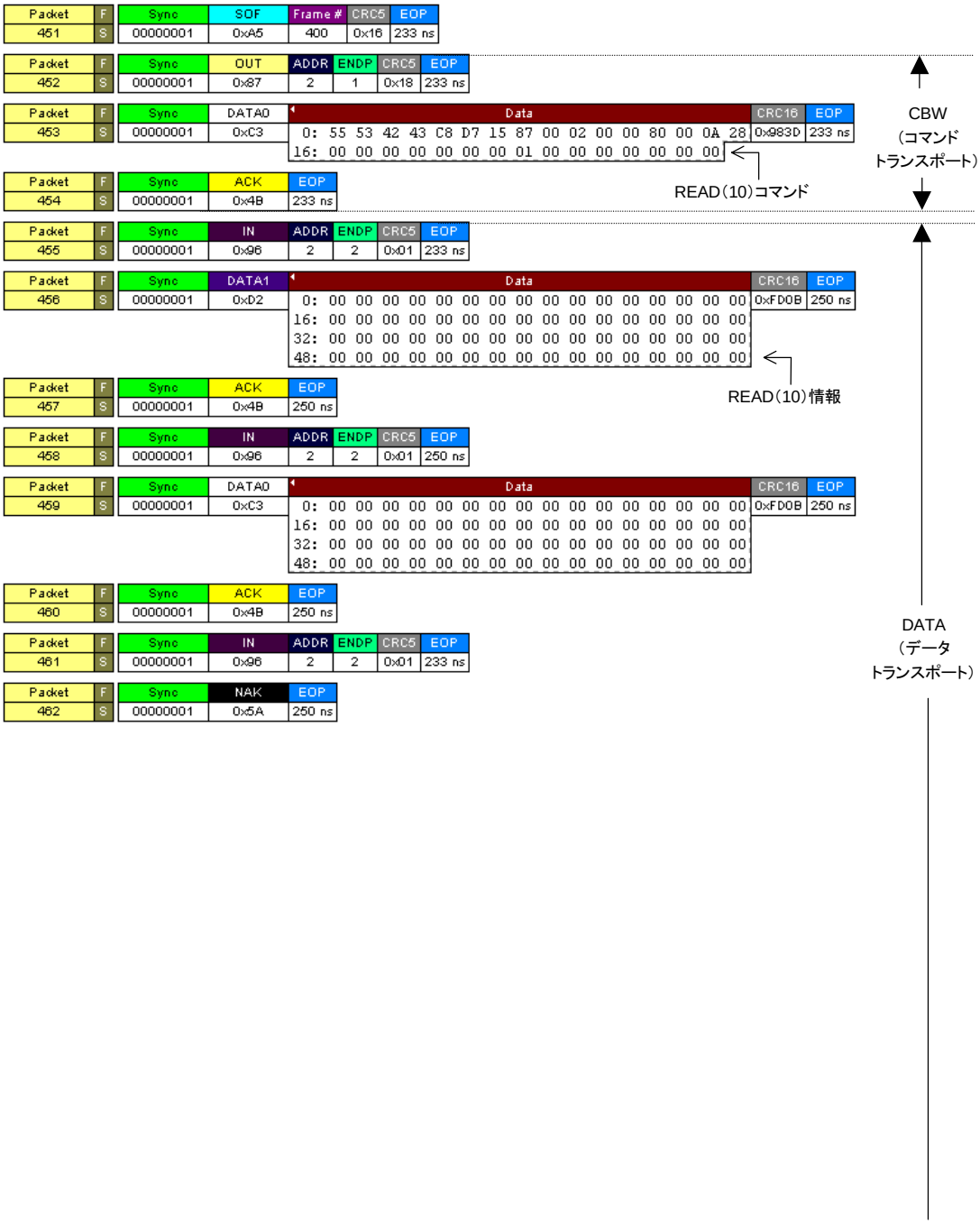
・READ CAPACITY コマンド

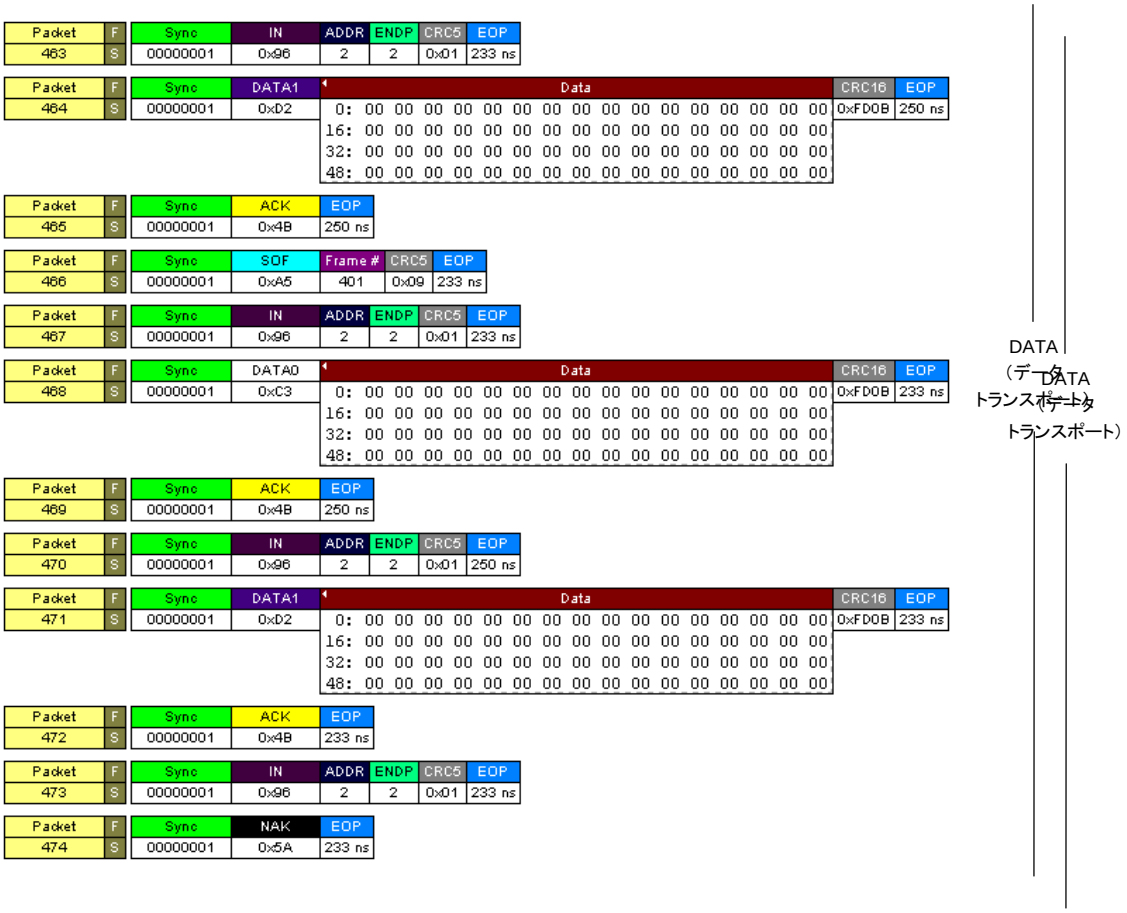


・ EQUEST SENSEコマンド



- READ（10）コマンド


Packet 451 S | Sync 00000001 | SOF 0xA5 | Frame # 400 | CRC5 0x16 | EOP 233 ns
Packet 452 S | Sync 00000001 | OUT 0x87 | ADDR 2 | ENDP 1 | CRC5 0x18 | EOP 233 ns
Packet 453 S | Sync 00000001 | DATA0 0xC3 | Data | CRC16 0x983D | EOP 233 ns
0: 55 53 42 43 C8 D7 15 87 00 02 00 00 80 00 0A 28
16: 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00
Packet 454 S | Sync 00000001 | ACK 0x4B | EOP 233 ns
READ(10)コマンド
CBW
（コマンド
トランスポート）
Packet 455 S | Sync 00000001 | IN 0x96 | ADDR 2 | ENDP 2 | CRC5 0x01 | EOP 233 ns
Packet 456 S | Sync 00000001 | DATA1 0xD2 | Data | CRC16 0xFD0B | EOP 250 ns
0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Packet 457 S | Sync 00000001 | ACK 0x4B | EOP 250 ns
READ(10)情報
Packet 458 S | Sync 00000001 | IN 0x96 | ADDR 2 | ENDP 2 | CRC5 0x01 | EOP 250 ns
Packet 459 S | Sync 00000001 | DATA0 0xC3 | Data | CRC16 0xFD0B | EOP 250 ns
0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Packet 460 S | Sync 00000001 | ACK 0x4B | EOP 250 ns
Packet 461 S | Sync 00000001 | IN 0x96 | ADDR 2 | ENDP 2 | CRC5 0x01 | EOP 233 ns
Packet 462 S | Sync 00000001 | NAK 0x5A | EOP 250 ns
DATA
（データ
トランスポート）



Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
463	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	DATA1	Data	CRC16	EOP
464	S	00000001	0xD2	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0xFD0B	250 ns

Packet	F	Sync	ACK	EOP
465	S	00000001	0x4B	250 ns

Packet	F	Sync	SOF	Frame #	CRC5	EOP
466	S	00000001	0xA5	401	0x09	233 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
467	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	DATA0	Data	CRC16	EOP
468	S	00000001	0xC3	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0xFD0B	233 ns

Packet	F	Sync	ACK	EOP
469	S	00000001	0x4B	250 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
470	S	00000001	0x96	2	2	0x01	250 ns

Packet	F	Sync	DATA1	Data	CRC16	EOP
471	S	00000001	0xD2	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0xFD0B	233 ns

Packet	F	Sync	ACK	EOP
472	S	00000001	0x4B	233 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
473	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	NAK	EOP
474	S	00000001	0x5A	233 ns

DATA（データトランスポート）

DATA（データトランスポート）

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
475	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	DATA0	Data	CRC16	EOP
476	S	00000001	0xC3	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0xFD0B	233 ns

Packet	F	Sync	ACK	EOP
477	S	00000001	0x4B	250 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
478	S	00000001	0x96	2	2	0x01	250 ns

Packet	F	Sync	NAK	EOP
479	S	00000001	0x5A	250 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
480	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	NAK	EOP
481	S	00000001	0x5A	250 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
482	S	00000001	0x96	2	2	0x01	233 ns

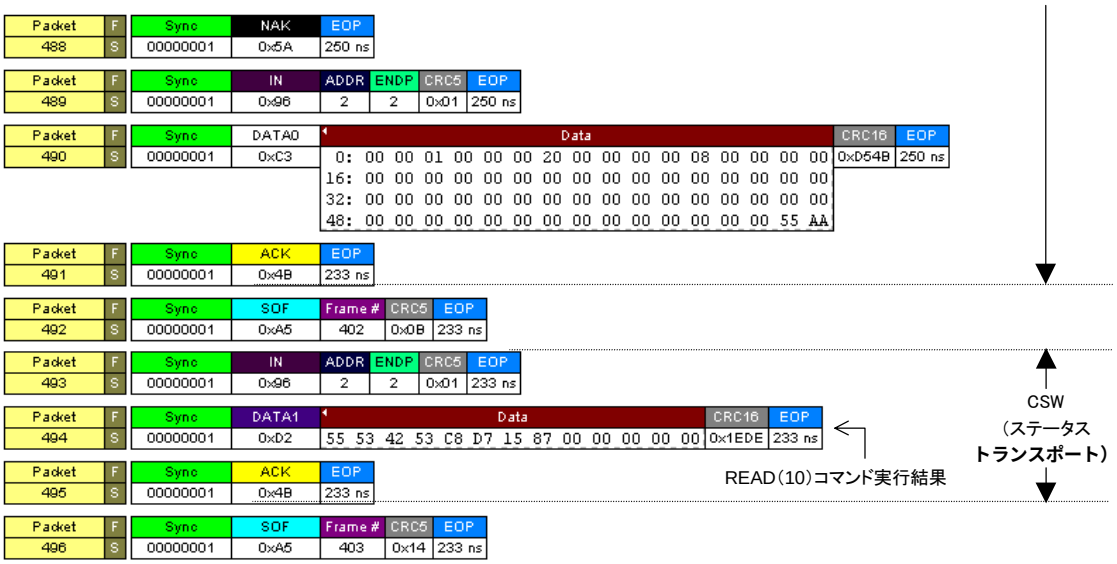
Packet	F	Sync	DATA1	Data	CRC16	EOP
483	S	00000001	0xD2	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 00	0x7B08	250 ns

Packet	F	Sync	ACK	EOP
484	S	00000001	0x4B	250 ns

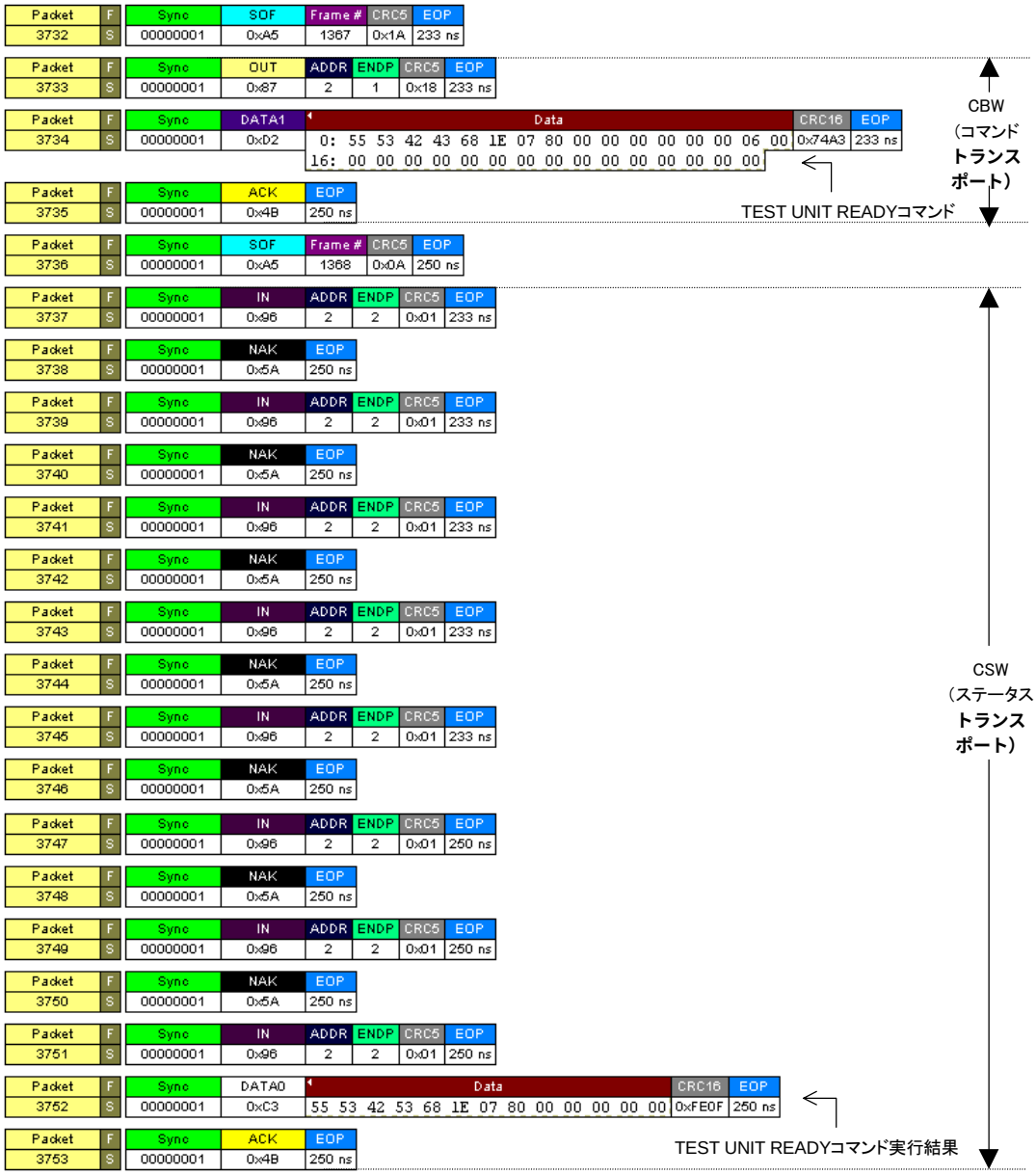
Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
485	S	00000001	0x96	2	2	0x01	250 ns

Packet	F	Sync	NAK	EOP
486	S	00000001	0x5A	250 ns

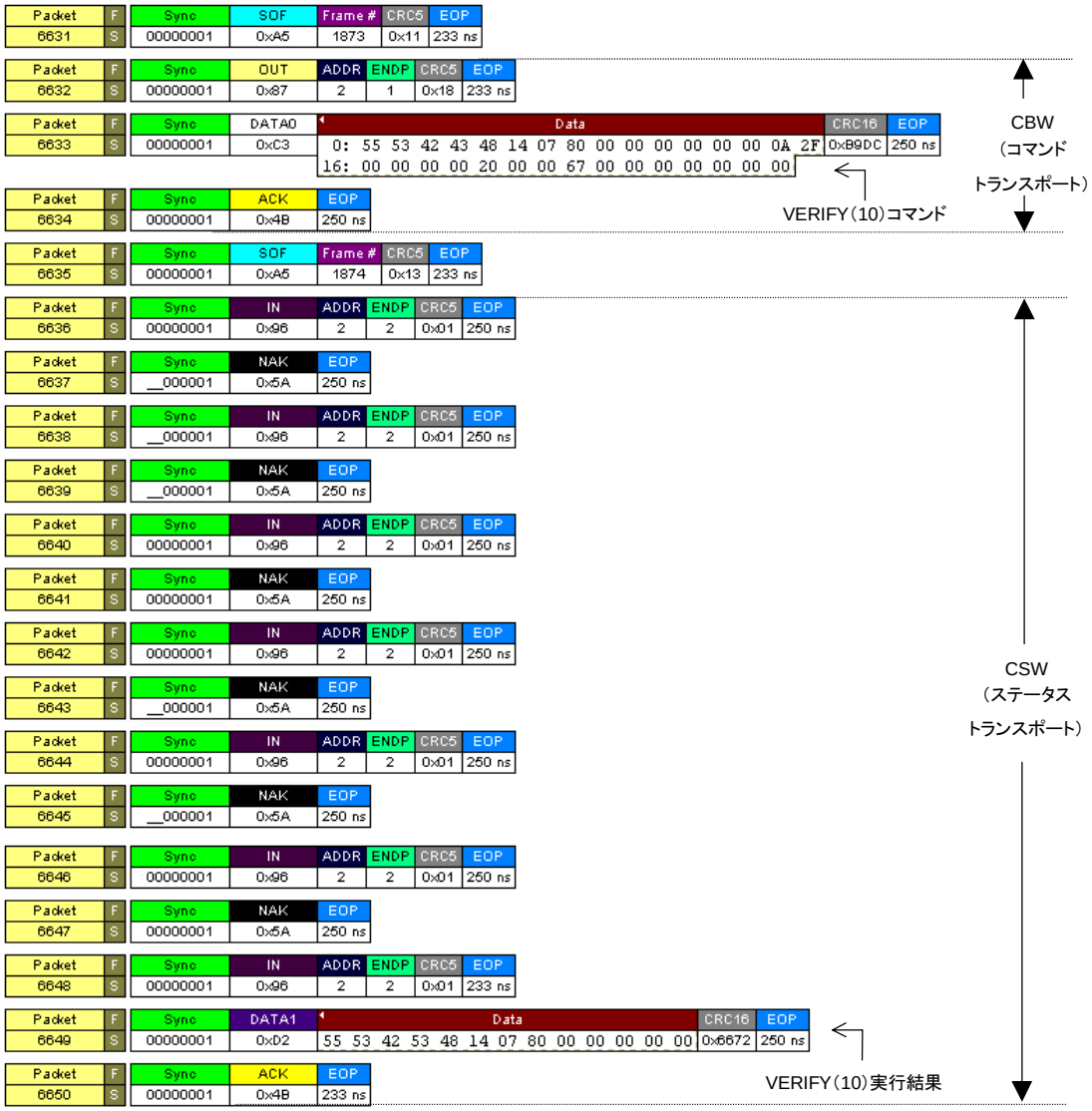
Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
487	S	00000001	0x96	2	2	0x01	250 ns


Packet 488 F S | Sync 00000001 | NAK 0x5A | EOP 250 ns
Packet 489 F S | Sync 00000001 | IN 0x96 | ADDR 2 | ENDP 2 | CRC5 0x01 | EOP 250 ns
Packet 490 F S | Sync 00000001 | DATA0 0xC3 | Data | CRC16 0xD54B | EOP 250 ns
0: 00 00 01 00 00 00 20 00 00 00 00 08 00 00 00 00
16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA
Packet 491 F S | Sync 00000001 | ACK 0x4B | EOP 233 ns
Packet 492 F S | Sync 00000001 | SOF 0xA5 | Frame # 402 | CRC5 0x0B | EOP 233 ns
Packet 493 F S | Sync 00000001 | IN 0x96 | ADDR 2 | ENDP 2 | CRC5 0x01 | EOP 233 ns
Packet 494 F S | Sync 00000001 | DATA1 0xD2 | Data 55 53 42 53 C8 D7 15 87 00 00 00 00 00 00 | CRC16 0x1EDE | EOP 233 ns
READ(10)コマンド実行結果
Packet 495 F S | Sync 00000001 | ACK 0x4B | EOP 233 ns
Packet 496 F S | Sync 00000001 | SOF 0xA5 | Frame # 403 | CRC5 0x14 | EOP 233 ns
CSW
(ステータス
トランスポート)

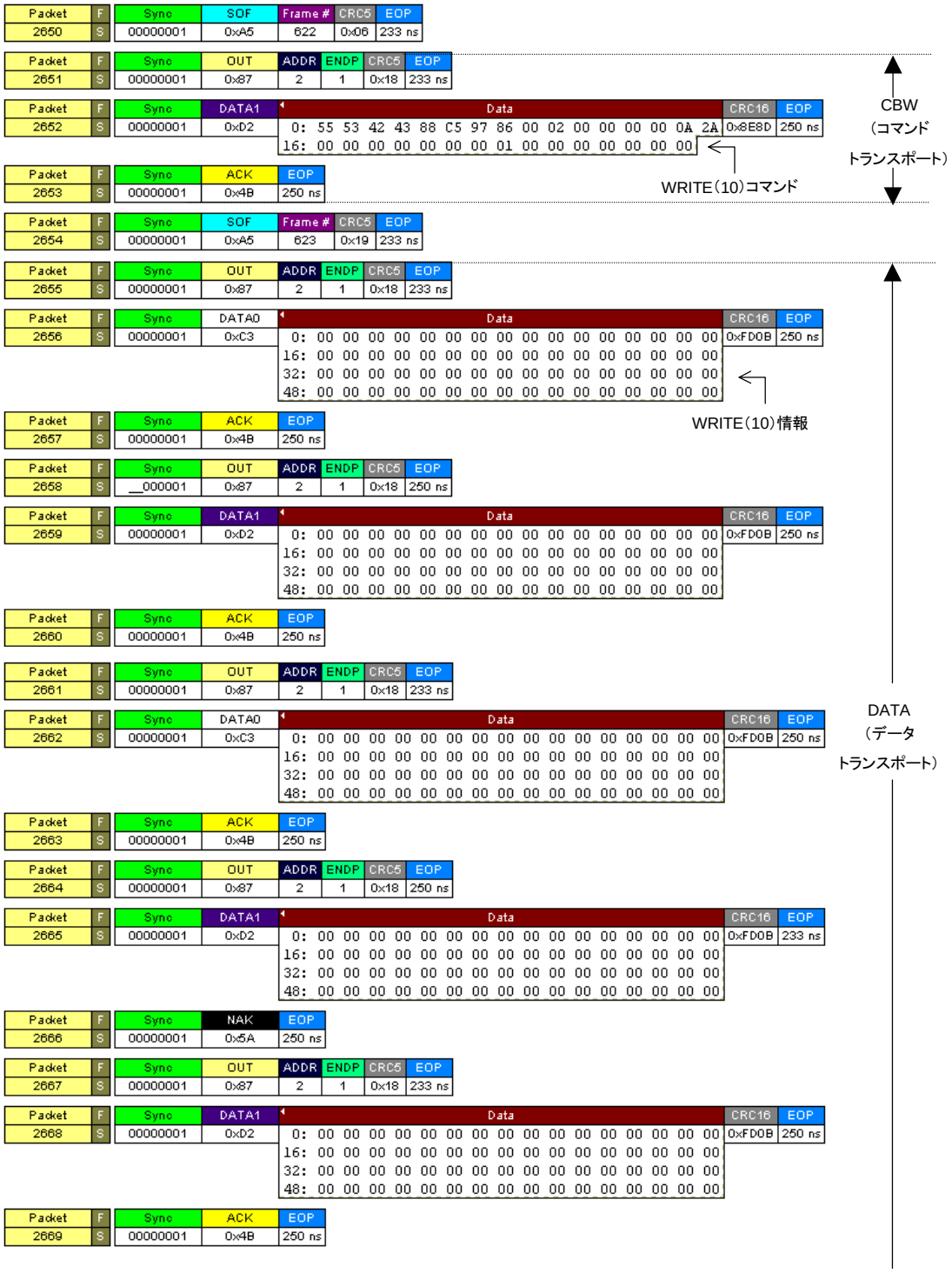
- TEST UNIT READYコマンド

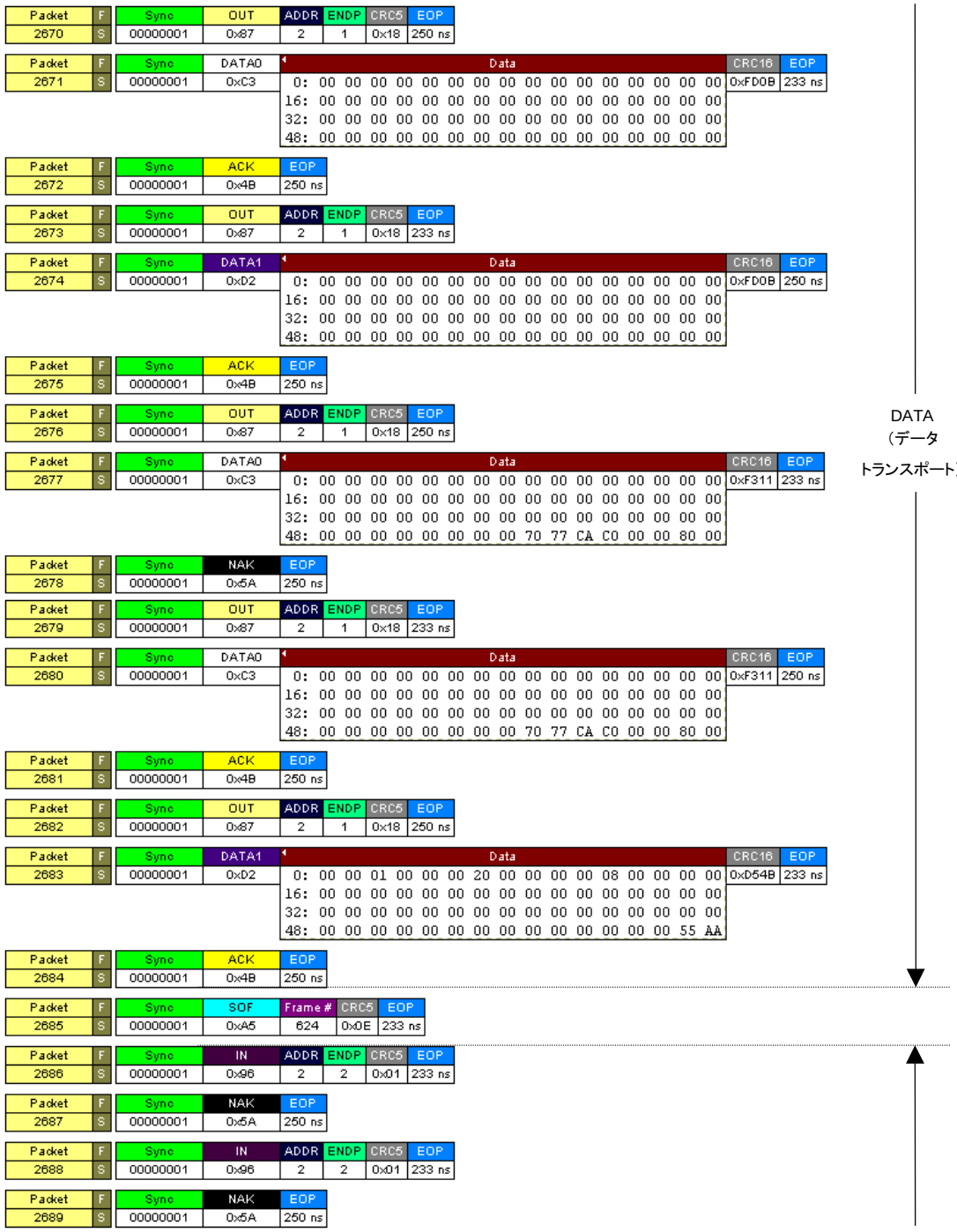


- VERIFY（10）コマンド



- WRITE（10）コマンド





Packet	F	Sync	OUT	ADDR	ENDP	CRC5	EOP
2670	S	00000001	0x87	2	1	0x18	250 ns

Packet	F	Sync	DATA0	Data	CRC16	EOP
2671	S	00000001	0xC3	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0xFD0B	233 ns

Packet	F	Sync	ACK	EOP
2672	S	00000001	0x4B	250 ns

Packet	F	Sync	OUT	ADDR	ENDP	CRC5	EOP
2673	S	00000001	0x87	2	1	0x18	233 ns

Packet	F	Sync	DATA1	Data	CRC16	EOP
2674	S	00000001	0xD2	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0xFD0B	250 ns

Packet	F	Sync	ACK	EOP
2675	S	00000001	0x4B	250 ns

Packet	F	Sync	OUT	ADDR	ENDP	CRC5	EOP
2676	S	00000001	0x87	2	1	0x18	250 ns

Packet	F	Sync	DATA0	Data	CRC16	EOP
2677	S	00000001	0xC3	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 70 77 CA C0 00 00 80 00	0xF311	233 ns

Packet	F	Sync	NAK	EOP
2678	S	00000001	0x5A	250 ns

Packet	F	Sync	OUT	ADDR	ENDP	CRC5	EOP
2679	S	00000001	0x87	2	1	0x18	233 ns

Packet	F	Sync	DATA0	Data	CRC16	EOP
2680	S	00000001	0xC3	0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 70 77 CA C0 00 00 80 00	0xF311	250 ns

Packet	F	Sync	ACK	EOP
2681	S	00000001	0x4B	250 ns

Packet	F	Sync	OUT	ADDR	ENDP	CRC5	EOP
2682	S	00000001	0x87	2	1	0x18	250 ns

Packet	F	Sync	DATA1	Data	CRC16	EOP
2683	S	00000001	0xD2	0: 00 00 01 00 00 00 20 00 00 00 00 08 00 00 00 00 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA	0xD54B	233 ns

Packet	F	Sync	ACK	EOP
2684	S	00000001	0x4B	250 ns

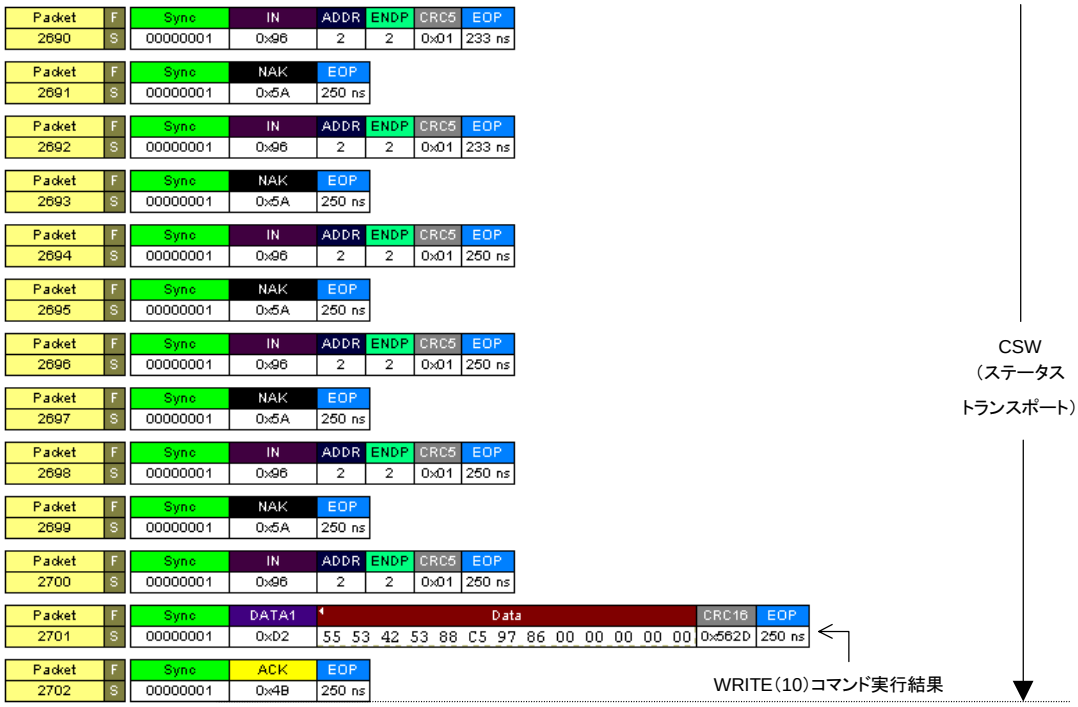
Packet	F	Sync	SOF	Frame #	CRC5	EOP
2685	S	00000001	0xA5	624	0x0E	233 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
2686	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	NAK	EOP
2687	S	00000001	0x5A	250 ns

Packet	F	Sync	IN	ADDR	ENDP	CRC5	EOP
2688	S	00000001	0x96	2	2	0x01	233 ns

Packet	F	Sync	NAK	EOP
2689	S	00000001	0x5A	250 ns


Packet F Sync IN ADDR ENDP CRC5 EOP
2690 S 00000001 0x96 2 2 0x01 233 ns
Packet F Sync NAK EOP
2691 S 00000001 0x5A 250 ns
Packet F Sync IN ADDR ENDP CRC5 EOP
2692 S 00000001 0x96 2 2 0x01 233 ns
Packet F Sync NAK EOP
2693 S 00000001 0x5A 250 ns
Packet F Sync IN ADDR ENDP CRC5 EOP
2694 S 00000001 0x96 2 2 0x01 250 ns
Packet F Sync NAK EOP
2695 S 00000001 0x5A 250 ns
Packet F Sync IN ADDR ENDP CRC5 EOP
2696 S 00000001 0x96 2 2 0x01 250 ns
Packet F Sync NAK EOP
2697 S 00000001 0x5A 250 ns
Packet F Sync IN ADDR ENDP CRC5 EOP
2698 S 00000001 0x96 2 2 0x01 250 ns
Packet F Sync NAK EOP
2699 S 00000001 0x5A 250 ns
Packet F Sync IN ADDR ENDP CRC5 EOP
2700 S 00000001 0x96 2 2 0x01 250 ns
Packet F Sync DATA1 Data CRC16 EOP
2701 S 00000001 0xD2 55 53 42 53 88 C5 97 86 00 00 00 00 00 0x562D 250 ns
Packet F Sync ACK EOP
2702 S 00000001 0x4B 250 ns
CSW
（ステータス
トランスポート）
WRITE（10）コマンド実行結果

H8S/2215 USBファンクションモジュール
Mass Storage Class（Bulk-Only Transport）
アプリケーションノート

発行年月　2002年3月　第1版
発　　行　株式会社　日立製作所
　　半導体グループビジネス企画本部
編　　集　株式会社　日立小平セミコン
　　技術ドキュメントグループ