お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

M16C/6C

USB Mass Storage Class (MSC) Bulk-Only Transport サンプルプログラム

・概要

このアプリケーションノートは、M16C/6C 内蔵の USB ファンクションモジュールを使用した Mass Storage Class Bulk-Only Transport ファームウェアについて説明し、お客様が USB ファンクションモジュールファームウェア作成時に、参考にしていただけるようまとめたものです。このアプリケーションノートの記述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を示すものであり、その内容を保証するものではありません。

なお、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参照してください。

・ 適用マイコン： M16C/6C グループ

本アプリケーションノートは、上記グループと同様のSFR(周辺機能制御レジスタ)を持つM16Cファミリマイコンでも使用できます。ただし、一部の機能を変更している場合がありますのでマニュアルで確認してください。また、本アプリケーションノートで説明しているプログラムを使用される場合は十分な評価を行ってください。

【関連マニュアル】

- Universal Serial Bus Specification Revision 2.0
- Universal Serial Bus Mass Storage Class Specification Overview Revision 1.1
- Universal Serial Bus Mass Storage Class Bulk-Only Transport Revision 1.0
- M16C/6Cグループ ハードウェアマニュアル
- E8aエミュレータ ユーザーズマニュアル

【注】 このアプリケーションノートに記載してい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ェアは準備しておりません。インタラプトの転送タイプをご使用になる場合は、別途プログラムを作成してください(M16C/6C グループハードウェアマニュアル参照)。

1. USBファンクションモジュール

M16C/6C 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。
表1.1にエンドポイントの構成を示します。

- エンドポイント0に対するUSB標準コマンド(注1)の自動実行
- フルスピード（12Mbps）転送対応
- USB送受信に必要な各種割り込み信号を生成
- USB動作クロック生成用のPLL回路内蔵
- バストランシーバ内蔵
- バスパワーモードまたはセルフパワーモードをUSBコントロールレジスタ(USBCTLR)で選択可能
- Set_Configuration割り込みで現在のConfiguration値がチェック可能

注1. 一部コマンドはファームウェアで処理する必要があります。

表 1.1　エンドポイントの構成

エンドポイント	シンボル	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 バイト	8 バイト	×
	EP0I	コントロール IN	16 バイト	16 バイト	×
	EP0O	コントロール OUT	16 バイト	16 バイト	×
エンドポイント 1	EP1	バルク OUT	64 バイト	64×2（128 バイト）	○
エンドポイント 2	EP2	バルク IN	64 バイト	64×2（128 バイト）	○
エンドポイント 3	EP3	インタラプト IN	16 バイト	16 バイト	×
エンドポイント 4	EP4	バルク OUT	64 バイト	64×2（128 バイト）	○
エンドポイント 5	EP5	バルク IN	64 バイト	64×2（128 バイト）	○
エンドポイント 6	EP6	インタラプト IN	16 バイト	16 バイト	×

図 1.1 にシステム構成例を示します。

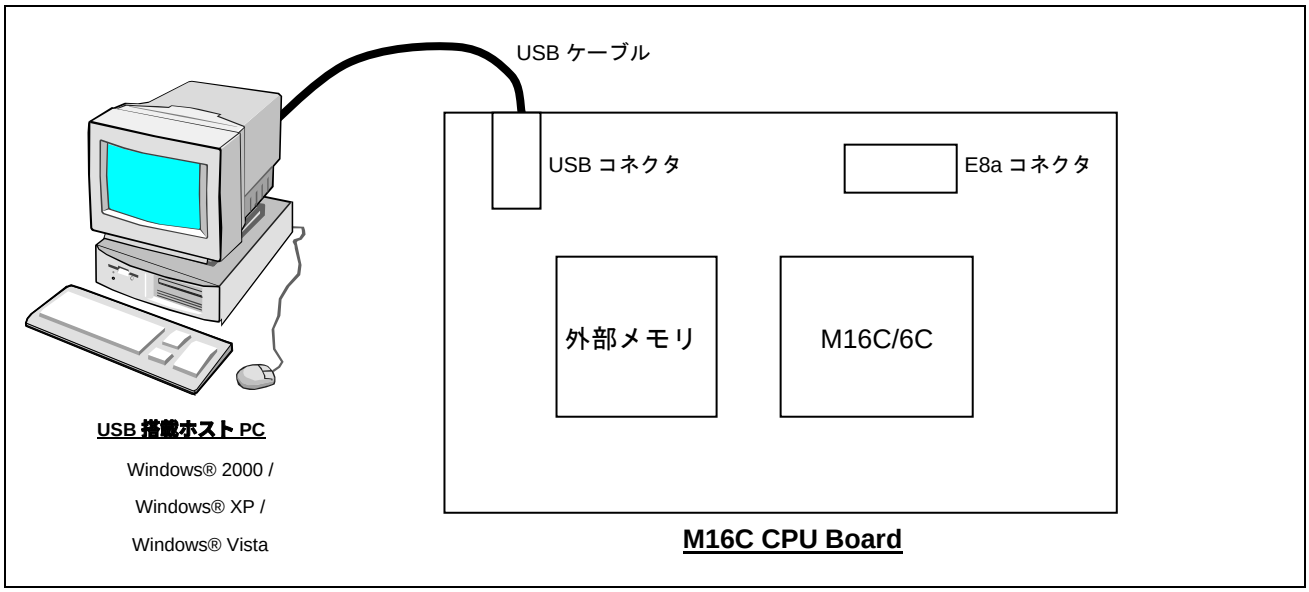


図 1.1　システム構成

本システムは、M16C/6C を搭載した M16C CPU Board、「Windows® 2000/Windows® XP/Windows® Vista」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、USB ホスト PC と M16C CPU Board を USB で接続し、M16C CPU Board 上の外部メモリを RAM Disk として動作させることにより、USB ホスト PC から M16C CPU Board の外部メモリ上へデータの保存、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することが可能です。

システムの特長を以下に示します。

サンプルプログラムによりM16C/6CのUSBファンクションモジュールを短期間で評価可能

コントロール転送、バルク転送をサポート

E8a Emulator(ルネサステクノロジ製)に対応しており効果的なデバッグが可能

プログラムを追加作成することによりインタラプト転送(注1)に対応可能

注 1. インタラプト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時に参照してください。

規格の詳細については、以下の関連マニュアルを参照してください。

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

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

2.1 USB Mass Storage Class（MSC）

USB Mass Storage Class とは、大規模記憶装置（ストレージ）を USB ホストに接続しデータの書き込み、読み出しなど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。

USBホストに、このクラスのファンクションであることを伝えるには、インタフェースディスクリプタのbInterfaceClassフィールドに値 0x08 を記述します。また、Mass Storage Classではストリングディスクリプタを用いてシリアルナンバーをホストへ伝える必要があります。このサンプルプログラムでは、Unicodeで 0000 0000 0001 を返信しています。

データ転送プロトコルは、USB ホスト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）の内、どの転送方法をどのように使用するかを定めています。

データ転送プロトコルは次の 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。

この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。

プロトコルコードは使用するデータ転送プロトコルを指定します。

USB ホストはファンクションに対してデータのリードやライト要求がある場合、ファンクションに対し命令（コマンド）を送信します。ファンクションは受信したコマンドを実行します。

サブクラスコードは USB ホストからファンクションに対して送られるコマンドフォーマットを定義しています。

2.2 サブクラスコード

サブクラスコードとは、USB ホスト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は 6 種類あります。**表 2.1** に内容を示します。

表 2.1 コマンドフォーマット

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

USB ホストに、機器が対応しているコマンドフォーマットを伝えるには、インタフェースディスクリプタの bInterfaceSubClass フィールドにサブクラスコード値を記述します。

この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

Bulk-Only Transport は USB ホストとファンクション間でデータの転送にバルク転送のみを使用します。

バルク転送には、以下の 2 種類があります。

バルク OUT 転送 ：USB ホストからファンクションにデータを送信するバルク転送

バルク IN 転送 ： ファンクションから USB ホストにデータを送信するバルク転送

Bulk-Only Transport では、バルク OUT 転送とバルク IN 転送をあらかじめ定めた組み合わせにすることにより、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み合わせになります。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。

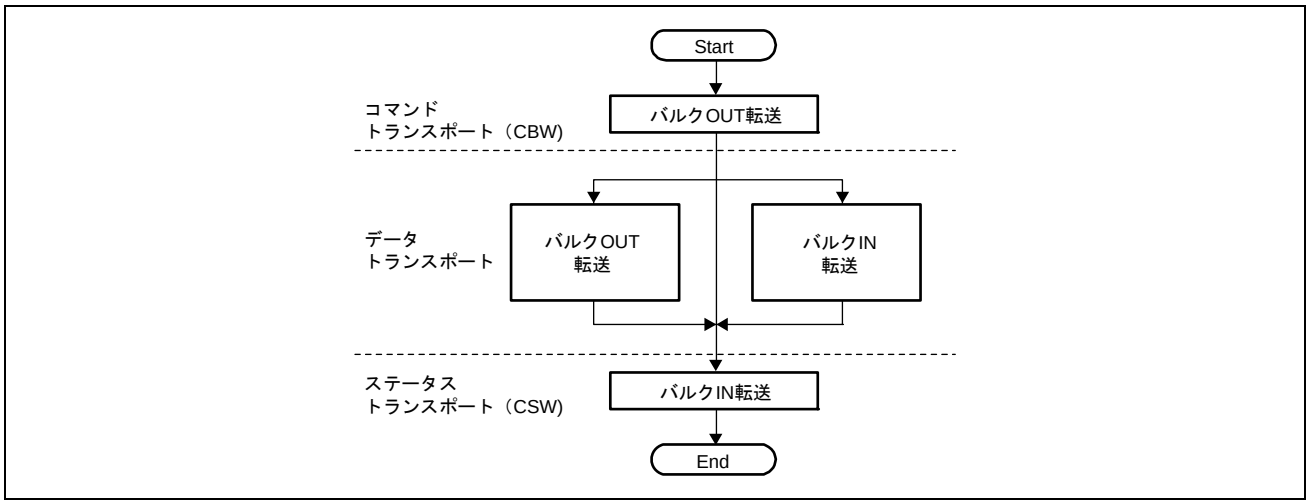


図 2.1 転送方法とトランスポートの関係

USBホストに、使用するデータ転送プロトコルを伝えるには、インタフェースディスクリプタのbInterfaceProtocolフィールドにプロトコルコード値を記述します。

このサンプルプログラムでは、プロトコルコード値 0x50 のBulk-Only Transportプロトコルを使用します。

2.3.1 コマンドトランスポート

コマンドトランスポートは USB ホストがファンクションに送るコマンドです。

コマンドファーマットは Command Block Wrapper（CBW）として定義されており、Bulk-Only Transport は CBW から始まります。

CBW は、31 バイト長のバルク OUT 転送で構成されます。

表 2.2 に CBW のフォーマットを示します。

表 2.2　CBW の詳細

bit / byte	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							

以下にCBWの各フィールド内容を示します。

dCBWSignature　：このデータパケットが CBW であることを示す識別子です。

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。USB ホストがコマンド毎に発行する識別子です。USB ホストはこの値を使用して一組の CBW と CSW を結び付けます。

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　：データトランスポートの転送方向

このフィールドのビット 7 が“0”の場合、データトランスポートはバルク OUT 転送で行われ、“1”の場合、バルク IN 転送で行われます。ビット 6～0 は“0”固定です。

bCBWLUN　：コマンドを送信する先の論理ユニット番号（Logical Unit Number）

一つのファンクション内に複数の論理ユニットを持つことができます。

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 USB ホストが実行したいコマンド（この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

ステータストランスポートではファンクションが USB ホストにコマンド実行結果を送ります。

ステータスフォーマットは Command Status Wrapper（CSW）として定義されており、Bulk-Only Transport は CSW で終わります。

CSW は、13 バイト長のバルク IN 転送で構成されます。

表 2.3 に CSW のフォーマットを示します。

表 2.3　CSW の詳細

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

以下にCSWの各フィールド内容を示します。

dCSWSignature　：このデータパケットが CSW であることを示す識別子。
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。ファンクションは、対応する CBW の dCBWTag フィールドと同じ値を送信します。USB ホストはこの値を使用して一組の CBW と CSW を結び付けます。

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差を示します。

bCSWStatus　：コマンドの実行結果を示します。内容は以下の通りです。
0x00 : コマンドが正常完了した
0x01 : コマンド FAIL によりコマンドが異常完了した
0x02 : フェーズエラーによりコマンドが異常完了した

2.3.3 データトランスポート

データトランスポートは、USB ホスト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、リード/ライトコマンド（「4.6 サポートする SCSI コマンドの動作」参照）では、データトランスポートで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 1 バイト長以上のバルク転送で構成されます。

データトランスポートで行われるデータ転送はバルク OUT 転送かバルク IN 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OUT転送）

データトランスポートがバルク OUT 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 コマンドを実行時に必要なデータです。

(2) データトランスポート（バルクIN転送）

データトランスポートがバルク IN 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 USB ホストに送信します。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

クラスコマンドとは、USB の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で転送されます。

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。この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

ファンクションは USB ホストより送信される CBW 内のサブクラスコマンドを処理する必要があります。

このサンプルプログラムでは、SCSI コマンドの中から表 2.5 に示す 11 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、USB ホストに対し CSW を使用し「コマンド FAIL である」と報告しています。

表 2.5 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
23	READ FORMAT CAPACITY	メディアのフォーマット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とき、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（6）	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
1B	STOP/START UNIT	メディアの着脱を制御する

3. 使用デバイス

- M16C CPU Board(Renesas Starter Kit for M16C/6C)
- E8a Emulator(ルネサステクノロジ製)
- E8a付属の接続ケーブル(ルネサステクノロジ製)
- E8a用PC (Windows® 2000/Windows® XP/Windows® Vista)
- USBホスト用PC (Windows® 2000/Windows® XP/Windows® Vista)
- USBケーブル
- High-performance Embedded Workshop4 (HEW4)(ルネサステクノロジ製)

3.1 ハードウェア環境

図 3.1 に各デバイスの接続形態を示します。

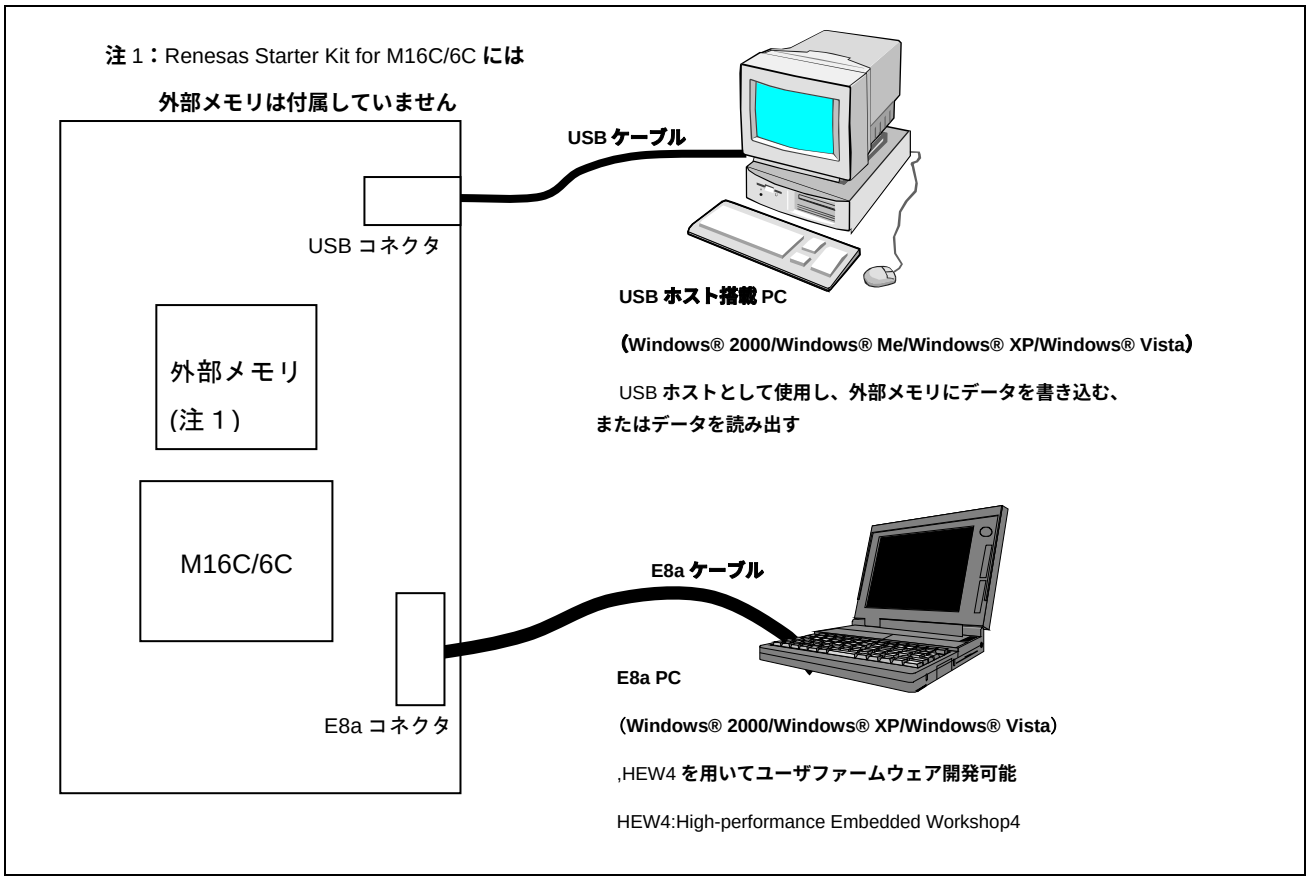


図 3.1　デバイス接続形態

(1) M16C CPU Board

表 3.1 に M16C CPU Board の各ポートに割り当てる機能を示します。

表 3.1　ポート割り当て

端子名	信号名	入出力	機能
P4_0～P4_3	P4_0～P4_3	出力	通信状態出力

(2) USBホストPC

USB ポートを搭載した、Windows® 2000/Windows® XP/Windows® Vista のいずれかをインストールした PC を、USB ホスト PC として使用してください。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。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) E8a用PC

E8a 用 PC の USB コネクタに E8a エミュレータを接続し、接続用のケーブルを介して M16C CPU Board と接続してください。接続後、HEW4 を起動してエミュレーションを行います。

3.2 ソフトウェア環境

サンプルプログラムと、コンパイラ、リンカ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および“MSC.hws”というワークスペースファイルは、MSC フォルダに格納されています。サンプルプログラムを使用するには、このフォルダを HEW4 がインストールされた PC にコピーしてください。

図 3.2 に MSC フォルダ内のファイルを示します。

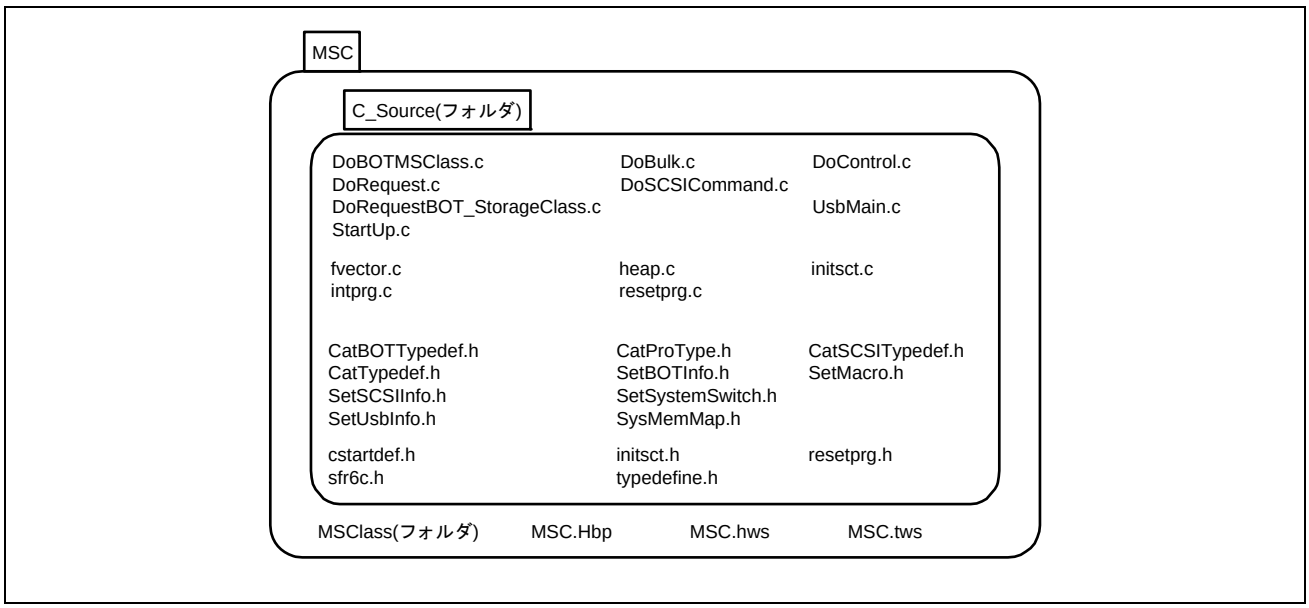


図 3.2 フォルダ内ファイル

3.2.2 コンパイルおよびリンク

サンプルプログラムは、High-performance Embedded Workshop4(以下、HEW4)によりコンパイルし、リンクします。以下に手順を示します。

(1)MSC フォルダを任意の場所にコピーしてください。

(2)“MSC.hws”というワークスペースファイルをダブルクリックし、HEW4 を起動してください。

(3)起動した HEW4 で、「ビルド」から「全てをビルド」を選択すると、コンパイルおよびリンクが行われます。

(3)が完了すると、“MSC¥MSCClass¥Debug”フォルダ内に、実行ファイルである“MSC.mot”と、マップファイルである“MSC.map”が作成されます。MSC.mot はモトローラ S タイプフォーマットファイルです。MSC.map にはプログラムのサイズ、変数のアドレスが格納されています。

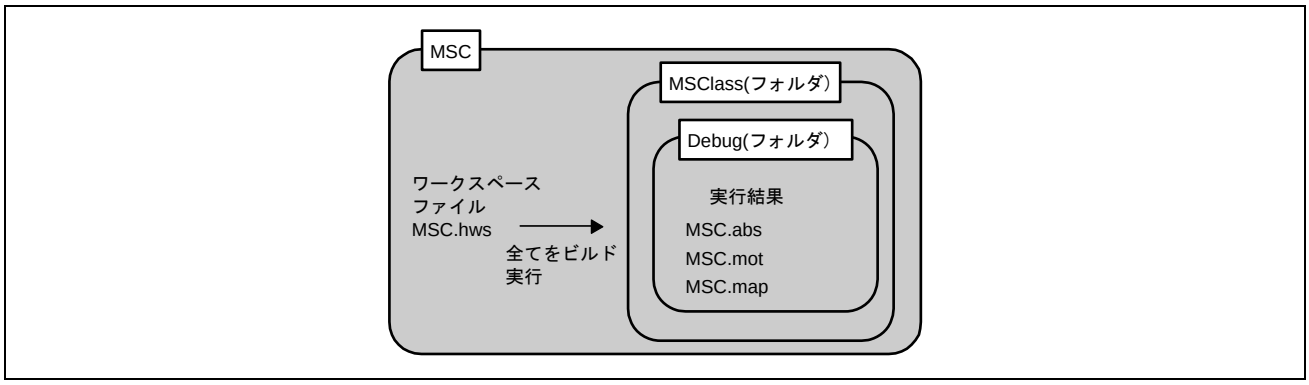


図 3.3 コンパイル結果

3.3 プログラムのダウンロードと実行方法

図 3.4 にサンプルプログラムのメモリマップを示します。

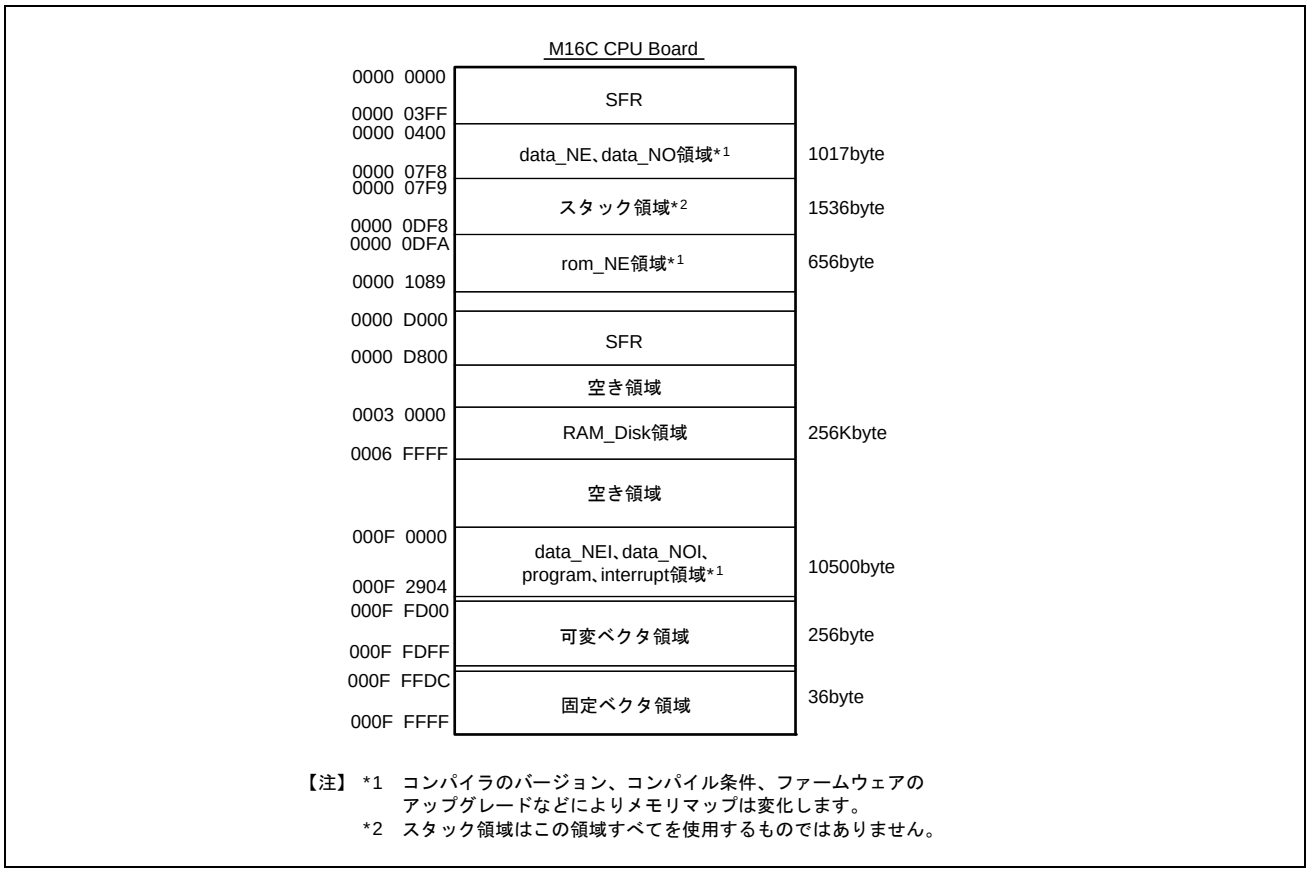


図 3.4 メモリマップ

図 3.4 のように、このサンプルプログラムはベクタ、data_NEI、data_NOI、program、interrupt を内蔵 ROM 領域上に配置。スタック、data_NE、data_NO、rom_NE を内蔵 RAM 上に配置、RAM_Disk として使用する領域を外部メモリ上に配置しています。

プログラムの配置を変更する場合は、HEW4 で変更してください。配置の結果は MSC¥MSCClass¥Debug フォルダ内の MSC.map に出力されます。

3.3.1 プログラムのダウンロードと実行

以下に、サンプルプログラムの実行手順を示します。

- HEW4をインストールしたE8a用PCにE8aを接続してください。
- E8a付属の接続ケーブルでE8aとM16C CPU Boardを接続してください。
- M16C CPU Boardの電源を投入してください。
- MSCフォルダ内のMSC.hwsを実行してください。
- デバッグ/接続 を選択してください。
- デバッグ/ダウンロード/ALL Download Modules を選択してください。プログラムがダウンロードされます。
- デバッグ/リセット実行 を選択してください。プログラムが実行されます。

3.4 RAM Diskの使用方法

Windows® XP を用いた場合を例に以下に説明します。

(1) 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M16C CPU Board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メレーション終了後、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 Renesas EX RAM Disk USB Device が表示されます。その結果、USB ホスト PC は M16C CPU Board を記憶デバイスとして認識し、マイコンピュータの中にリムーバブルディスクがマウントされます。

(2) 次にリムーバブルディスクをフォーマットします。

リムーバブ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。

(3) ドライブのフォーマット選択ウインドウが開くので、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であることを確認し、開始ボタンをクリックしてください。

(4) フォーマットの実行確認ウインドウが開くので、OKボタンをクリックしてください。

(5) 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開くので、OKボタンをクリックしてください。

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

以上で M16C CPU Board を USB 接続の RAM-Disk として使用できます。

【注】なお、Renesas Starter Kit for M16C/6C には外部 RAM が搭載されていないため、マイコンピュータの中にリムーバブルディスクがマウントされる所まで実行が可能です。

3.5 RAM Diskの設定変更

このサンプルプログラムで使用する RAM Disk の設定の変更方法について説明します。

3.5.1 リムーバブル・固定ディスクの選択

このサンプルプログラムでは、RAM Disk をリムーバブルディスクとして使用しています。

SetSysytemSwitch.h 内の「#define REMOVABLE_DISK」を無効にし、「#undef REMOVABLE_DISK」を有効にすることにより、固定ディスクとして使用することができます。

3.5.2 RAM Diskの容量の変更

このサンプルプログラムでは、256K バイトの外部メモリを RAM Disk として使用しています。RAM Disk の容量を変更するには、SysMemMap.h の内容を変更する必要があります。まず、RAM Disk として使用する全体のバイト数*1 を、DISK_ALL_BYTE で指定します。次に、RAM Disk として使用する領域の始まりと終わりを RAM_DISK_S、RAM_DISK_E*2 で指定します。

【注】*1 FAT 情報などで領域を消費するため、PC から見える容量は若干減少します。このサンプルプログラムでは、約 16M バイトまでを FAT12、約 2G バイトまでを FAT16 として FAT 情報を構成します。その他の FAT システムの FAT 情報はお客様で用意していただく必要があります。

*2 RAM_DISK_S から RAM_DISK_E で指定する領域は、DISK_ALL_BYTE で指定するサイズ以上必要です。

なお、Renesas Starter Kit for M16C/6C には外部 RAM が搭載されていないため、リムーバブルディスクから変更しないでください。

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。このサンプルプログラムは M16C CPU Board 上で動作し、M16C CPU Board 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。M16C/6C 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、USB 割り込み 0、 USB 割り込み 1、USB RESUME 割り込みの 3 種類です。

このサンプルプログラムは、M16C CPU Board 上で動作します。

サンプルプログラムの特長を以下に示します。

- コントロール転送可能
- バルクOUT転送でUSBホストからデータを受信可能
- バルクIN転送でUSBホストにデータを送信可能
- SCSIコマンドに対応するRAM-Diskとして動作

4.1 状態遷移

図 4.1 にこのサンプルプログラムの状態遷移図を示します。

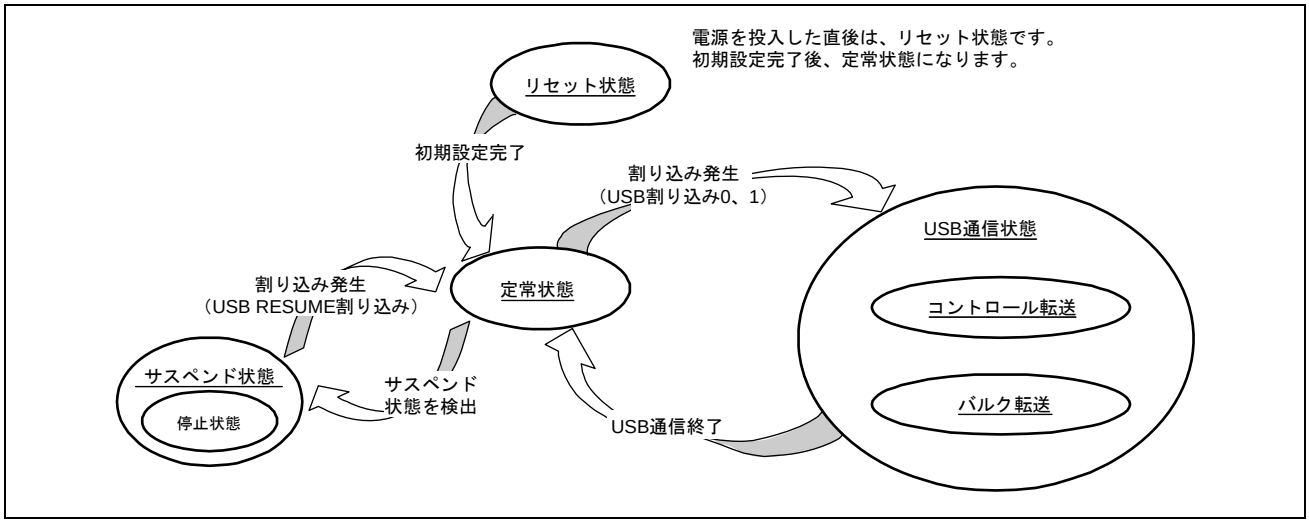


図 4.1 状態遷移図

- リセット状態
  ハードウェアリセット後、この状態になります。リセット状態では、主にM16C/6Cの初期設定を行います。
- 定常状態
  初期設定が完了すると、メインルーチンで定常状態となります。
- USB通信状態
  定常状態でUSB割り込みが受け付けられるとこの状態になります。USB割り込み要因は12種類あります。割り込み要求が発生すると、USB割り込みフラグレジスタ0、1、2、3の対応するビットが“1”になります。USB通信状態では、割り込み要因に応じた転送方式でデータを送受信します。
- 停止状態
  ホストのサスペンド状態を検出すると、停止状態に遷移します。USB RESUME割り込みが発生すると停止状態から定常状態に復帰します。

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、まず 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。

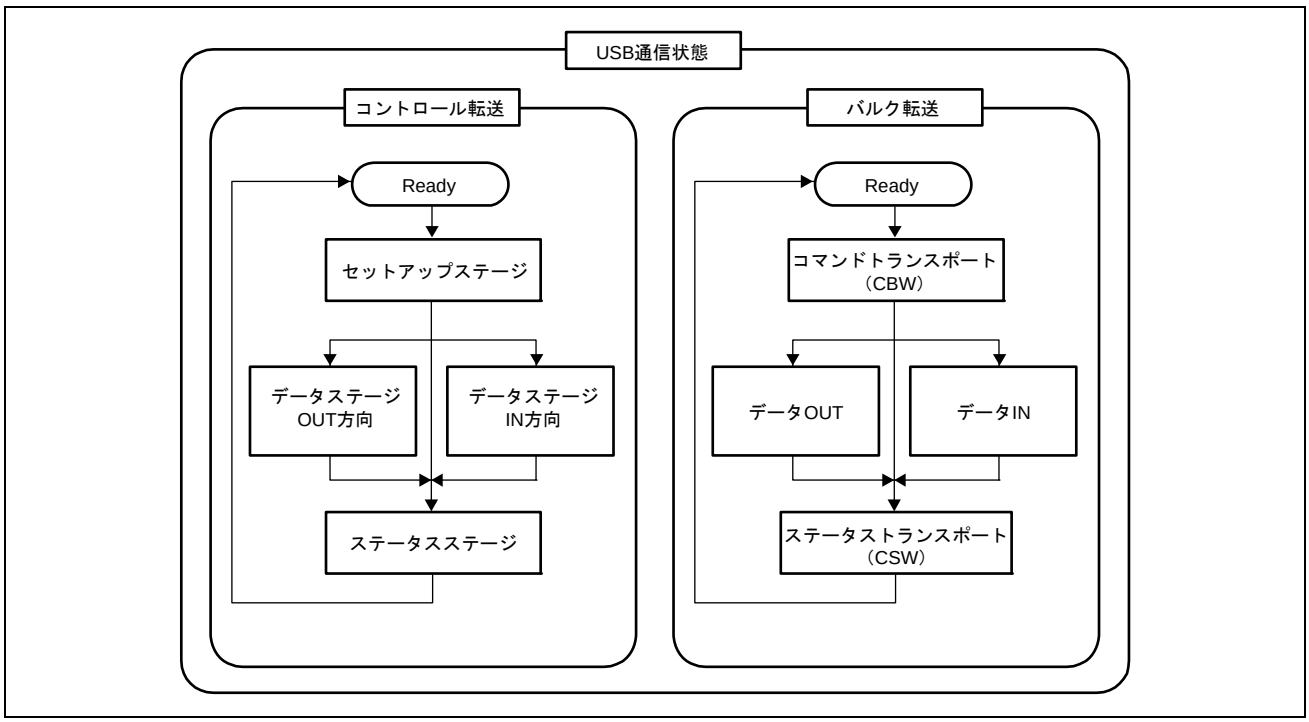


図 4.2　USB 通信状態

4.2.1 コントロール転送

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、USB ホスト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

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、USB ホスト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

このサンプルプログラムは、13 個のソースファイルと 15 個のヘッダファイルで構成されています。関数は、転送方式または機能ごとに一つのファイルにまとめてあります。**表 4.1** に構成ファイルを示します。**図 4.3** に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
DoBOTMSClass.c	Mass Storage Class Bulk-Only Transport を実行
DoBulk.c	バルク転送を実行
DoControl.c	コントロール転送を実行
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
DoSCSICommand.c	SCSI コマンドの解析および処理
UsbMain.c	割り込み要因の判定 パケットの送受信
StartUp.c	USB ファンクションの初期設定
fvector.c	固定ベクタテーブル
heap.c	ヒープエリアの割り付け
initsct.c	各セクションのクリアと割り付け
intprg.c	可変ベクタテーブル
resetprg.c	電源投入時のマイコン初期設定
CatBOTTypedef.h	Bulk-Only Transport 用構造体定義
CatProType.h	プロトタイプ宣言
CatSCSITypedef.h	SCSI 用構造体定義
CatTypedef.h	USB ファームウェアで使用する基本の構造体定義
SetBOTInfo.h	Bulk-Only Transport 対応に必要な変数の初期設定
SetMacro.h	マクロ定義
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
SetUsbInfo.h	USB 対応に必要な変数の初期設定
SysMemMap.h	メモリマップのアドレス定義
cstartdef.h	スタックおよびヒープサイズ設定ファイル
initscth	セクション定義ファイル
resetprg.h	スタックサイズの定義ファイル
sfr6c.h	M16C/6C のレジスタ定義
typedefine.h	各アクセスサイズの定義

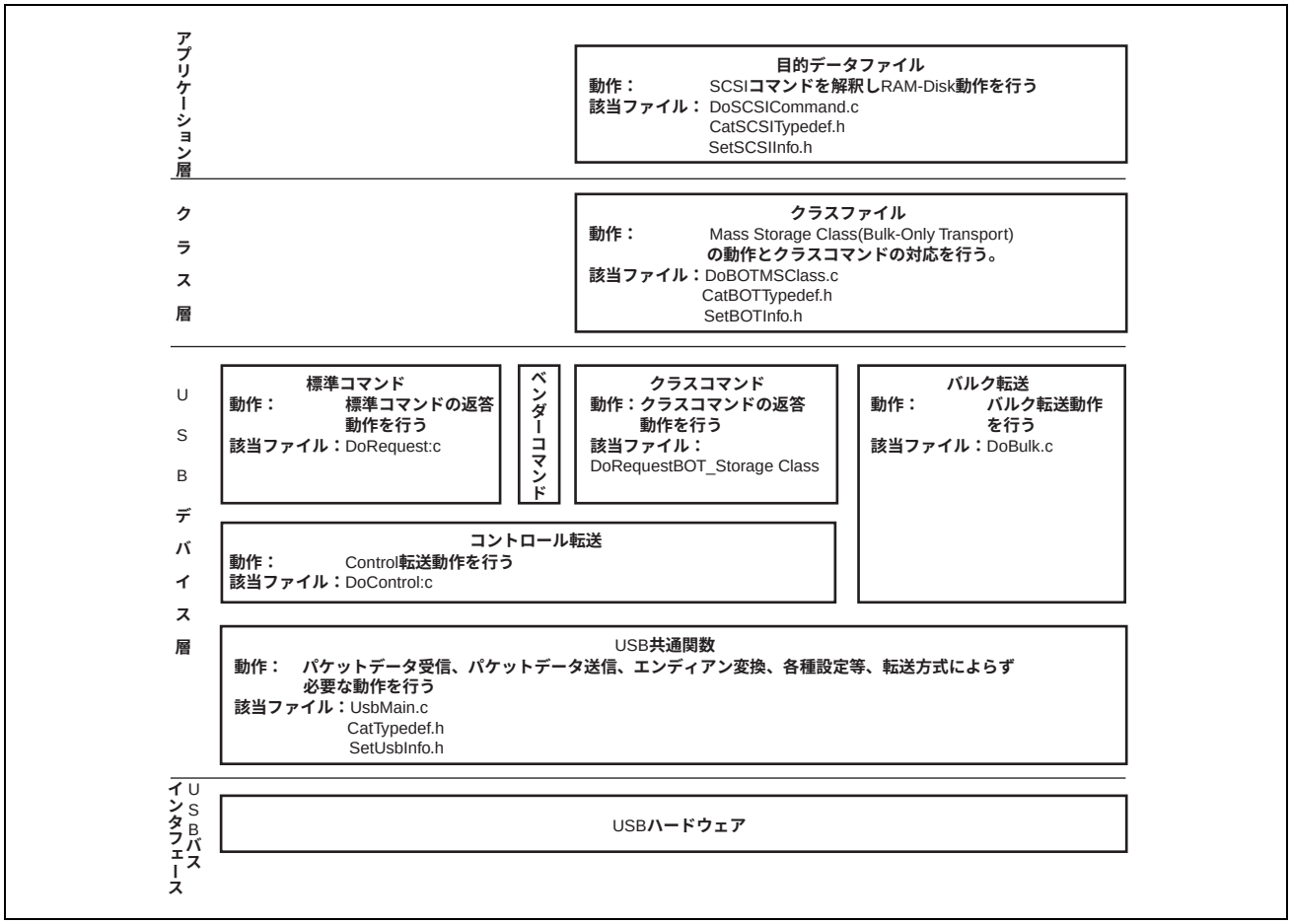


図 4.3　ストレージクラス（Bulk-Only Transport）ファームウェアの階層構造

4.4 関数の機能

表 4.2～表 4.9 に、ファイルに含まれる関数とその機能を示します。

ハードウェアリセット時、StartUp.c の SetPowerOnSection が呼び出されます。ここでは M16C/6C 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　StartUp.c

格納ファイル	関数名	機　　能
StartUp.c	SetPowerOnSection	バス、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チンへ移行
	_INITSCT	初期値がある変数を、RAM のワークエリアにコピー
	InitMemory	バルク転送で使用する 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
	SetEPInfoR	エンドポイント情報の書き込み
	error	エラー発生時、CPU をスリープモードに遷移
	SuspendResume	停止状態、定常状態の判定を行う
	Resume_int	停止状態からの復帰

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を呼び出します。また、USB ホストと USB 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 4.3　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および割り込みに応じた関数の呼び出し
	BranchOfInt0	割り込み要因の判定および割り込みに応じた関数の呼び出し
	BranchOfInt1	割り込み要因の判定および割り込みに応じた関数の呼び出し
	GetPacket	USB ホストから転送されたデータを RAM に書く
	GetPacket4	USB ホストから転送されたデータをロングワードサイズで RAM に書く（リングバッファ対応版）。Mass Storage Class では使用しません。
	GetPacket4S	USB ホストから転送されたデータをロングワードサイズで RAM に書く（リングバッファ非対応、高速版）。
	PutPacket	USB ホストに転送するデータを USB ファンクションモジュールに書く
	PutPacket4	USB ホストに転送するデータをロングワードサイズで USB ファンクションモジュールに書く（リングバッファ対応版）。Mass Storage Class では使用しません。
	PutPacket4S	USB ホストに転送するデータをロングワードサイズで USB ファンクションモジュールに書く（リングバッファ非対応、高速版）。
	SetControlOutContents	ホストから送られたデータの書き換え
	SetUsbModule	USB ファンクション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 ファンクションモジュールの制御を行う（このサンプルプログラムでは使用しません）
	ConvRealn	指定した番地から指定バイト長のデータを読む
	ConvReflexn	指定した番地から指定バイト長のデータを逆順に読む

(注)本サンプルファームでは、BranchOfInt は使用しません。

コントロール転送時に、USB ホスト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。このサンプルプログラムでは、ベンダ ID の値に 045B（ベンダ：Renesas Technology Corp.）を使用します。製品開発時には「USB Implementers Forum」にてベンダ ID を取得してください。また、このサンプルプログラムでは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時には、別途プログラムを作成してください。

表 4.4　DoRequest.c

格納ファイル	関数名	機　　能
DoRequest.c	DecStandardCommands	USB ホスト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に応じた処理を行う
	DecVenderCommands	ベンダコマンドに応じた処理を行う

コントロール転送時にホストから送られてくるコマンドをデコードし、Mass Storage Class Bulk-Only Transport コマンド（Bulk-Only Mass Storage Reset と Get Max LUN）であれば対応する処理を行います。

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。この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 4.5　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 コマンドに応じた処理を行う

コントロール転送の割り込み（SETUPTS）が受け付けられると、ActControl がコマンドを取得し、DecStandardCommands でデコードを行いコマンドの転送方向を判別します。その後、コントロール転送の割り込み（EP0OTS、EP0ITR、EP0ITS）が発生すると、ActControlInOut がコマンドの転送方向により、ActControlIn または ActControlOut を呼び出します。呼び出された関数は、データステージと、ステータスステージを制御します。

表 4.6　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を制御
	ActControlIn	コントロール IN 転送（データステージが IN 方向の転送）のデータステージとステータスステージを制御
	ActControlOut	コントロール OUT 転送（データステージが OUT 方向の転送）のデータステージとステータスステージを制御
	ActControlInOut	コントロール転送のデータステージとステータスステージを、ActControlIn と ActControlOut に振り分ける

バルク転送に関する処理を行います。

表 4.7　DoBulk.c

格納ファイル	関数名	機　　能
DoBulk.c	ActBulkOut	バルク OUT 転送を行う
	ActBulkIn	バルク IN 転送を行う
	ActBulkInReady	バルク IN 転送の準備を行う

DoBOTMSClass.c では、Mass Storage Class Bulk-Only Transport の 2 または 3 つのステージ制御と仕様にしたがった動作を行います。

表 4.8　DoBOTMSClass.c

格納ファイル	関数名	機　　能
DoBOTMSClass.c	ActBulkOnly	Bulk-Only Transport のステージ別に振り分ける
	ActBulkOnlyCommand	Bulk-Only Transport の CBW を制御
	ActBulkOnlyIn	（データステージが IN 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を制御
	ActBulkOnlyOut	（データステージが OUT 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を制御

DoSCSICommand.c では、USB ホストから送られてきた 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。

表 4.9　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 コマンドに応じた処理を行う
	SetBotCmdErr	SCSI コマンドのエラー時の処理を行う

図 4.4 に**表 4.2**～**表 4.9** で説明した関数の相関関係を示します。上側の関数が、下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ます。USB 割り込みの発生によって遷移する USB 通信状態では、BranchOfInt0、BranchOfInt1 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すもので、関数が呼び出される順序を示すものではありません。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参照してください。

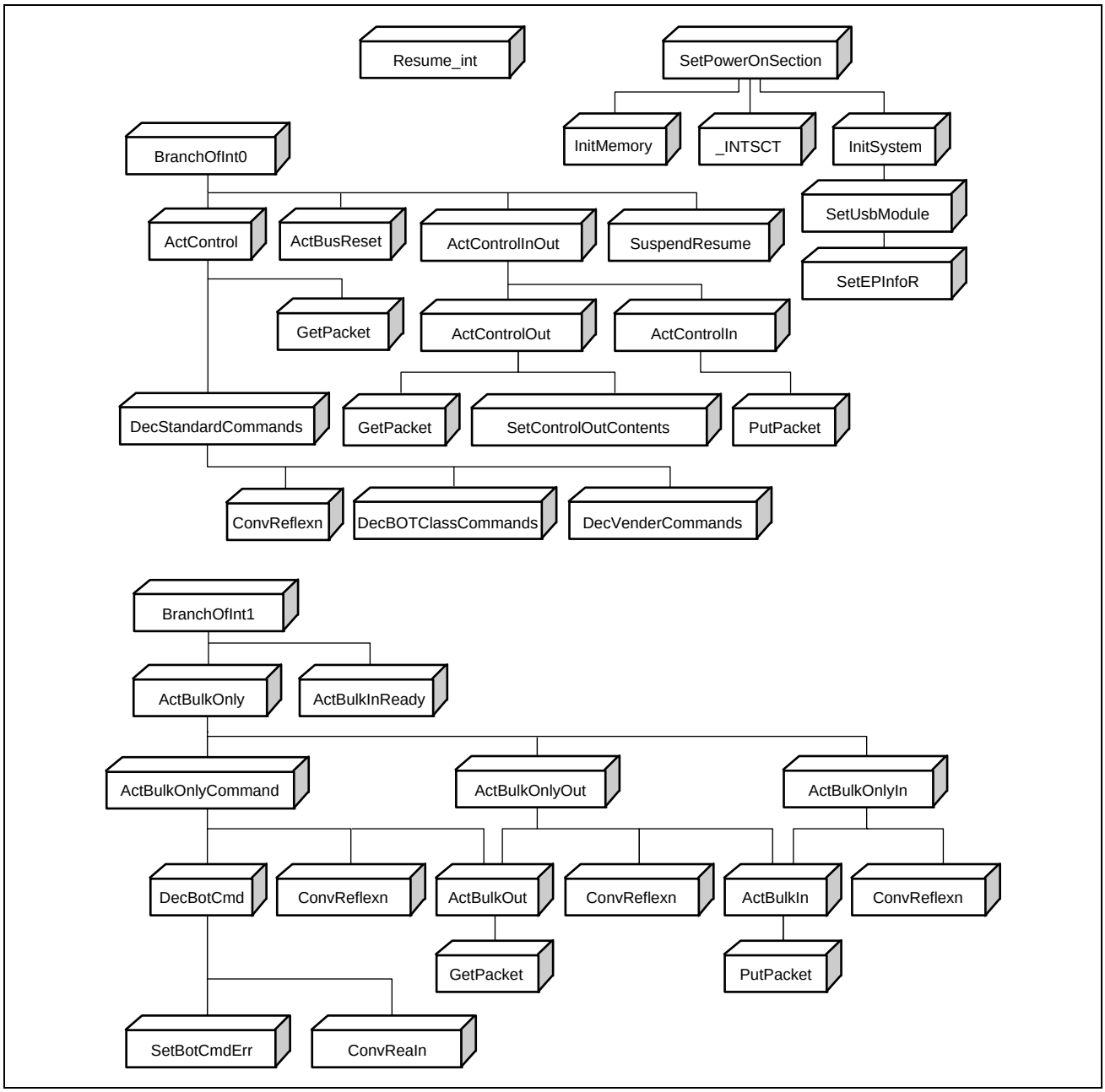


図 4.4　関数の相関関係

4.5 RAM-Disk

このサンプルプログラムでは M16C CPU Board 上の外部メモリを Disk 装置に見立て、USB ホストに対し M16C CPU Board（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外部メモリ上の 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。

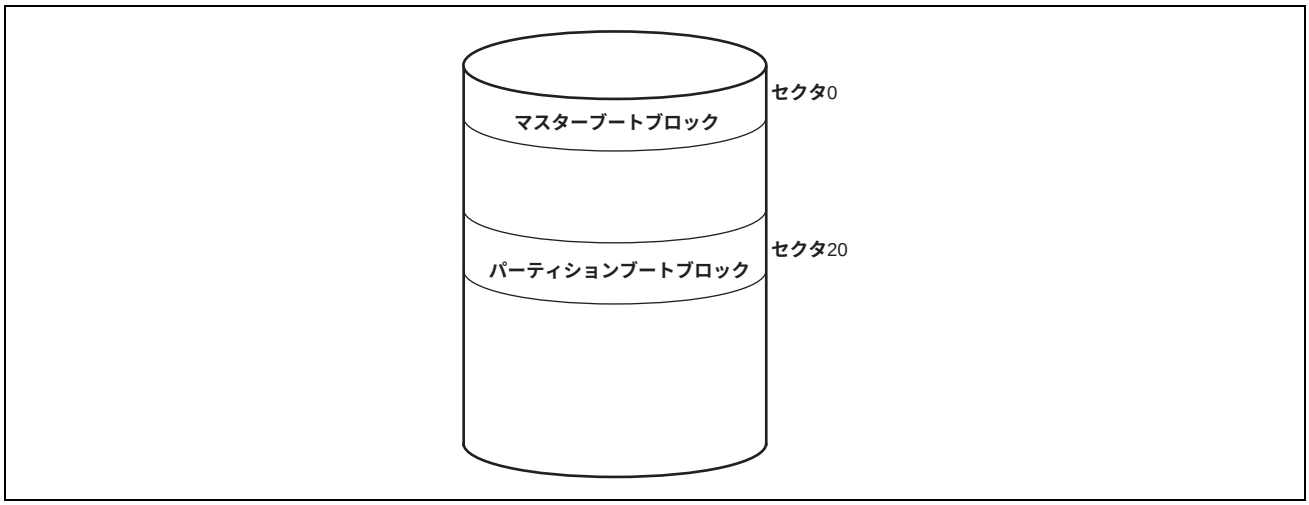


図 4.5　Disk の構造

USB ホスト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。

【注】なお、Renesas Starter Kit for M16C/6C には外部 RAM が搭載されていないため、RAM-Disk 領域へのアクセスはできません。

4.6 サポートするSCSIコマンドの動作

表 4.10 にこのサンプルプログラムがサポートする SCSI コマンドの動作を示します。

表 4.10 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	USB ホストに対し INQUIRY 情報をバルク IN 転送にて送信します。
	CSW	USB ホスト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 FORMAT CAPACITY	CBW	コマンドをデコードし、READ FORMAT CAPACITY コマンドであることを認識後、READ FORMAT CAPACITY 情報(20 バイト)の送信準備を行います。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）ファンクションはデータ転送なしとして扱い、4.7 エラー時の処理（4）にしたがいます。また、REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	USB ホストに対し READ CAPACITY 情報をバルク IN 転送にて送信します。 メディアがアクセス不能状態の場合、ホストが要求したデータと同量のデータ（0x00）を返信します。
	CSW	USB ホストに対しコマンド実行結果を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。
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、外部メモリ上に展開してある 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。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）ファンクションはデータ転送なしとして扱い、4.7 エラー時の処理（4）にしたがいます。また、REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	USB ホストに対し READ CAPACITY 情報をバルク IN 転送にて送信します。 メディアがアクセス不能状態の場合、ホストが要求したデータと同量のデータ（0x00）を返信します。
	CSW	USB ホストに対しコマンド実行結果を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。
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、外部メモリ上に展開してある 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。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）ファンクションはデータ転送なしとして扱い、4.7 エラー時の処理（4）にしたがいます。また、REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	USB ホストに対し読み出しセクタのデータをバルク IN 転送にて送信します。 メディアがアクセス不能状態の場合、ホストが要求したデータと同量のデータ（0x00）を返信します。
	CSW	USB ホストに対し READ（10）コマンド実行結果を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。

コマンド名	トランスポート名	動作内容
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、外部メモリ上に展開してある 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。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）ファンクションはデータ転送なしとして扱い、4.7 エラー時の処理（9）にしたがいます。また、REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	USB ホストから書き込みセクタのデータをバルク OUT 転送にて受信します。 メディアがアクセス不能状態の場合、ホストから送られたデータを空読みします。
	CSW	USB ホストに対し正常終了を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。
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	USB ホストに対し返答値をバルク IN 転送にて送信します。
	CSW	USB ホストに対しこのコマンド実行結果を送信します。送信データが 18 バイト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、USB ホストに対し正常終了の送信準備を行います。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）コマンドを FAIL に設定し、REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	このコマンドにデータトランスポートは存在しません。
	CSW	USB ホストに対し正常終了を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。
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、USB ホストに対し正常終了の送信準備を行います。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）コマンドを FAIL に設定し、REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	このコマンドにデータトランスポートは存在しません。
	CSW	USB ホストに対し正常終了を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。

コマンド名	トランスポート名	動作内容
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、USB ホストに対し正常終了の送信準備を行います。 メディアがアクセス不能状態の場合（unit_state[0]の最下位ビットが“1”）コマンドを FAIL に設定し､REQUEST SENSE で返信する値を NOT READY（用意ができていない）に設定します。
	データ	このコマンドにデータトランスポートは存在しません。
	CSW	USB ホストに対し正常終了を送信します。 メディアがアクセス不能状態の場合、コマンド FAIL（CSW ステータス 0x01）を返信します。
STOP/START UNIT	CBW	コマンドをデコードし、STOP/START UNIT であることを認識後、コマンドがメディアの取り出し、もしくは停止を指定していたときにはグローバル変数 unit_state[0]の最下位ビットを“1”にセットします。その他の場合にはグローバル変数 unit_state[0]の最下位ビットを“0”にセットします。 ユーザーがアクセス不能状態から復帰させたい場合には unit_state[0]の最下位ビットを“0”にしてください。
	データ	このコマンドにデータトランスポートは存在しません。
	CSW	USB ホストに対し正常終了を送信します。
MODE SENSE（6）	CBW	コマンドをデコードし、MODE SENSE（6）コマンドであることを認識後、要求された MODE SENSE 情報の送信準備を行います。
	データ	USB ホストに対し MODE SENSE 情報をバルク IN 転送にて送信します。
	CSW	USB ホストに対しコマンド実行結果を送信します。
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	USB ホストがバルク IN 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。 USB ホストがバルク OUT 転送にてデータを転送した場合、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	USB ホストに対しコマンド FAIL（CSW ステータス 0x01）を送信します。

4.7 エラー時の処理

Mass Storage Class Bulk-Only Transport の転送を行う際、USB ホスト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 の規格では次に挙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 4.11** と**表 4.12** 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 4.11 ホスト-ファンクション間のデータ転送状態

		ホストの期待		
		Hn (データ転送なしを期待)	Hi (ファンクションからのデータ受信を期待)	Ho (ファンクションへのデータ送信を期待)
ファンクションの意図	Dn (データ転送なしを意図)	（1）Hn = Dn	（4）Hi > Dn	（9）Ho > Dn
	Di (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	
	Do (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 4.12 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 4.13 に発生する可能性のあるエラー状況例を示します。

表 4.13　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

表 4.14 にエラー状況に対するファンクションの対応動作を示します。

表 4.14　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
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 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 CSW のステータスに 0x00 を設定
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 CSW のステータスに 0x02 を設定
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 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 CSW のステータスに 0x01 を設定
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 CSW のステータスに 0x02 を設定

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

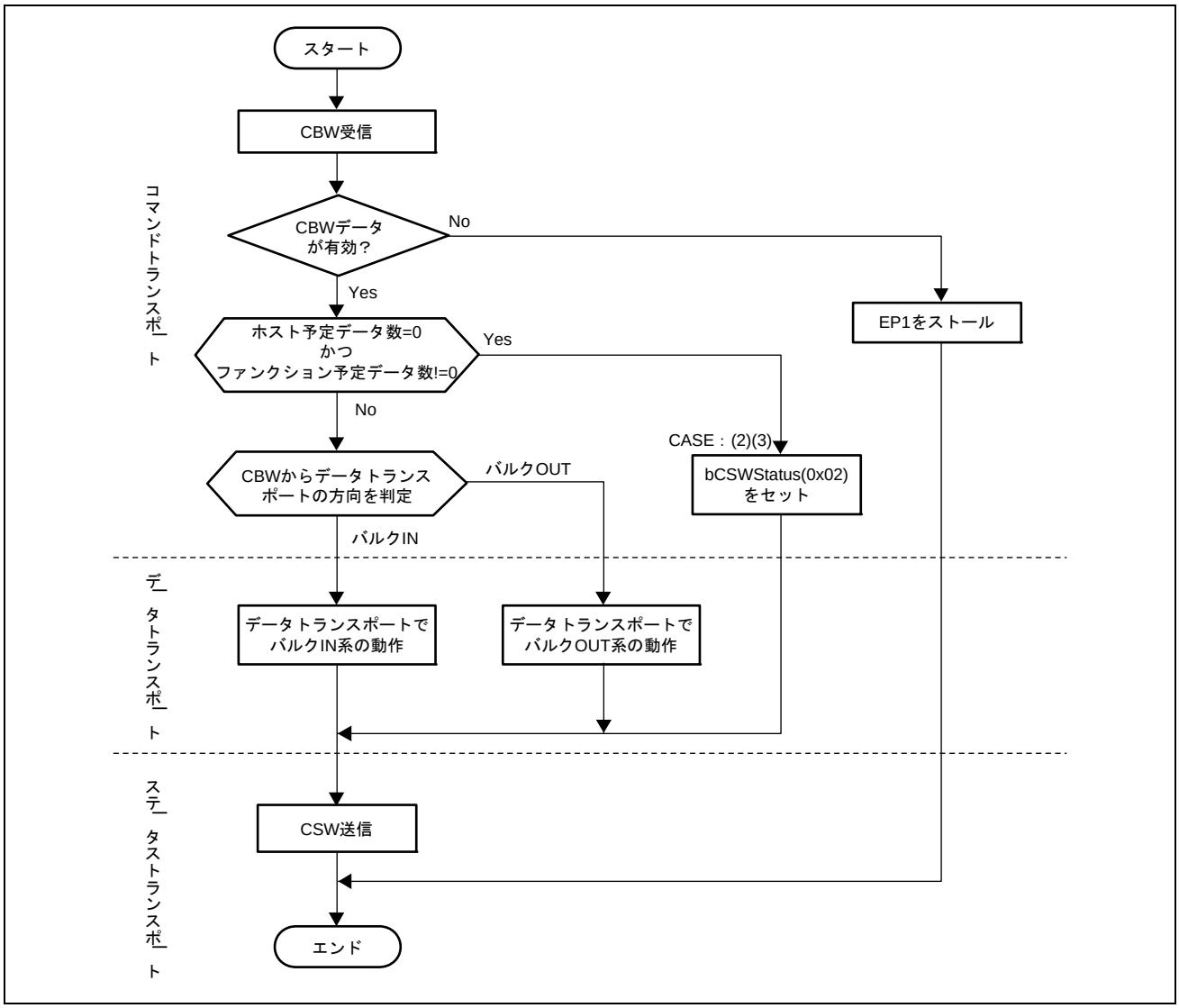


図 4.6　データ転送時のエラー処理フロー（1）

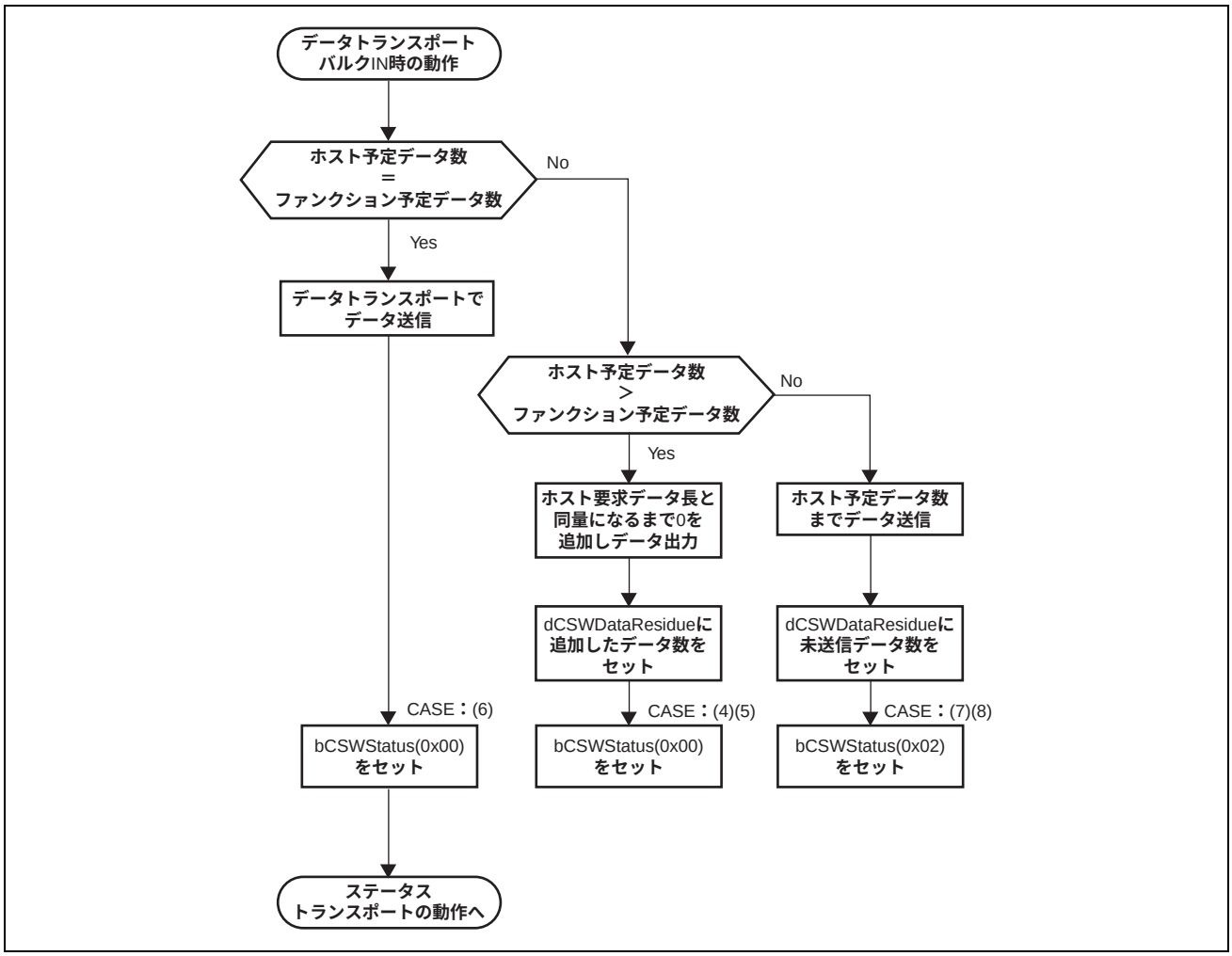


図 4.7　データ転送エラー発生時の処理フロー（2）

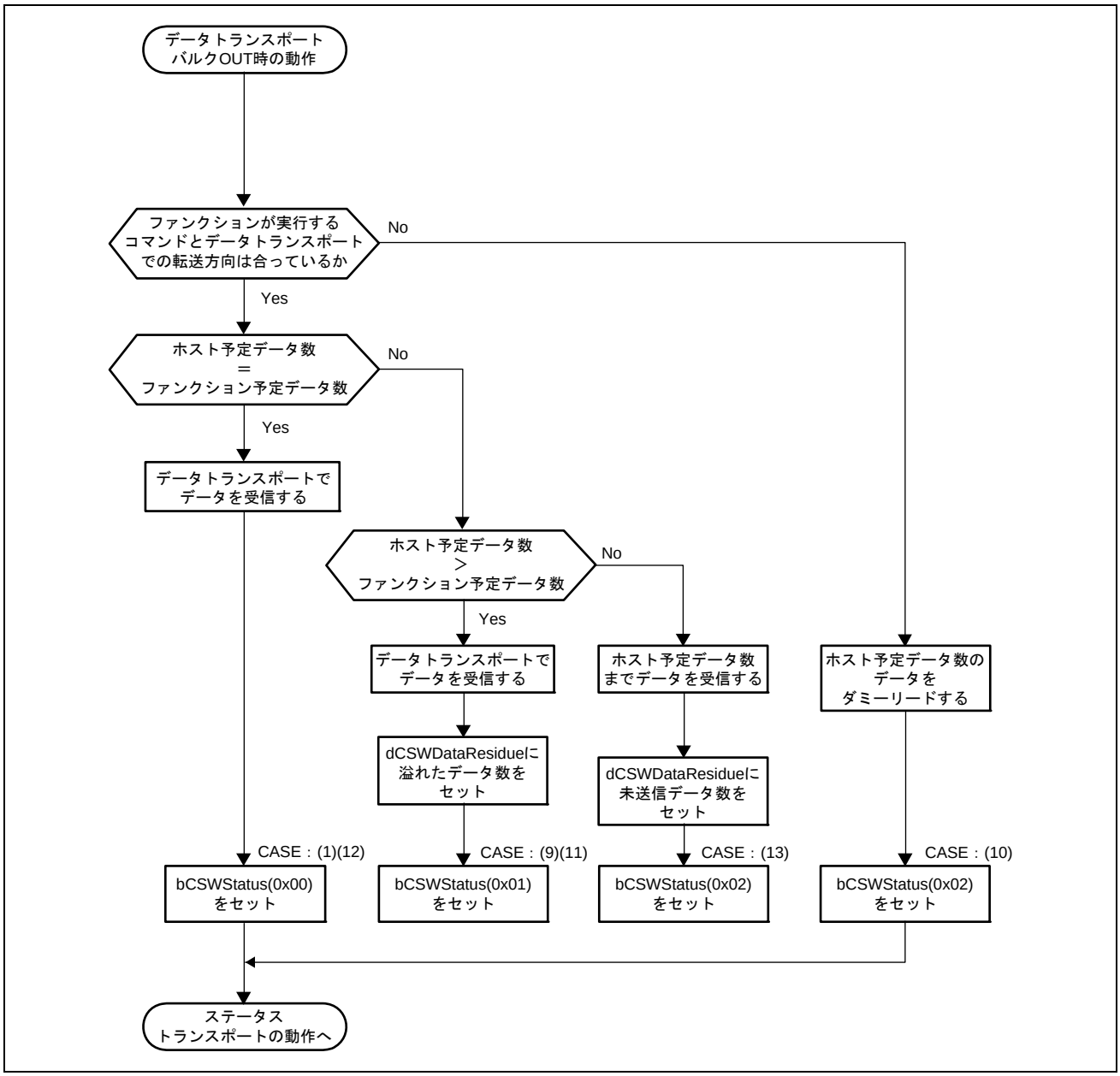


図 4.8　データ転送エラー発生時の処理フロー（3）

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、USB ホストに 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。

この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dCBWDataTransferLengthフィールド
- CSWパケットのdCSWDataResidueフィールド

を使用します。

CBW パケットの dCBWDataTransferLength フィールドは USB ホスト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 dCSWDataResidue 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、dCBWDataTransferLength フィールドと dCSWDataResidue 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時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ときは、データトランスポートでデータ転送するたびに dCBWDataTransferLength フィールドと dCSWDataResidue 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 dCSWDataResidue 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。

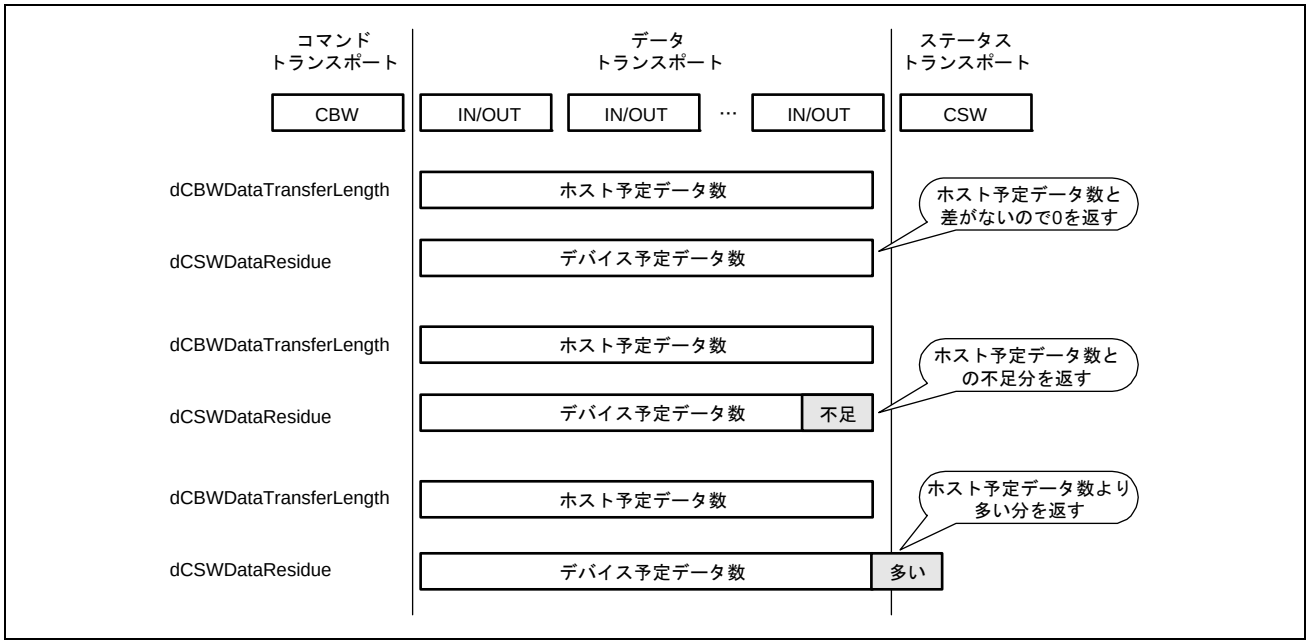


図 4.9　Bulk-Only Transport における各ステージ

5. サンプルプログラムの動作

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づけて説明します。

5.1 メインルーチン

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 を初期化します。図 5.1 に SetPowerOnSection のフローチャートを示します。

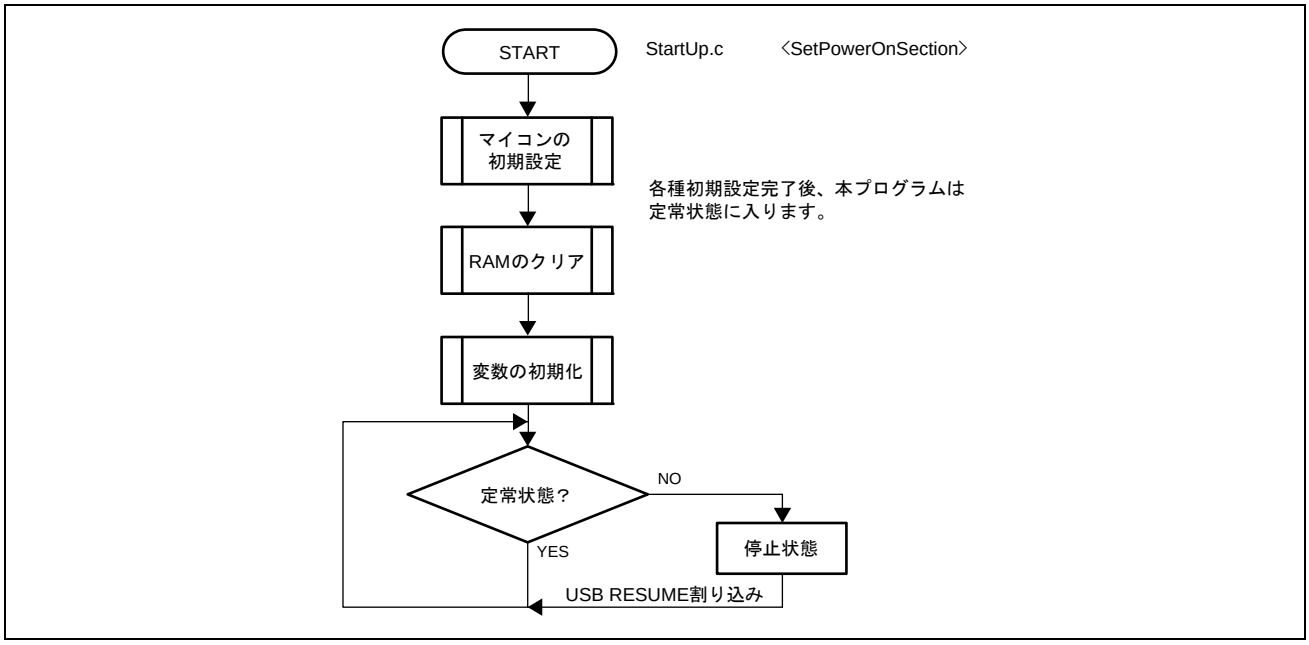


図 5.1 メインルーチン

5.2 割り込みの種類

4 章で説明したように、このサンプルプログラムが USB 通信状態で使用する割り込みは、USB 割り込みフラグレジスタ 0～3（USBIFR0～3）によって示される計 12 種類です。割り込み要求が発生すると、USB 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、“USB 割り込み 0”、“USB 割り込み 1” 割り込みが発生します。サンプルプログラムでは、この割り込みによって USB 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 USB 割り込みフラグレジスタと USB 通信との関係を示します。

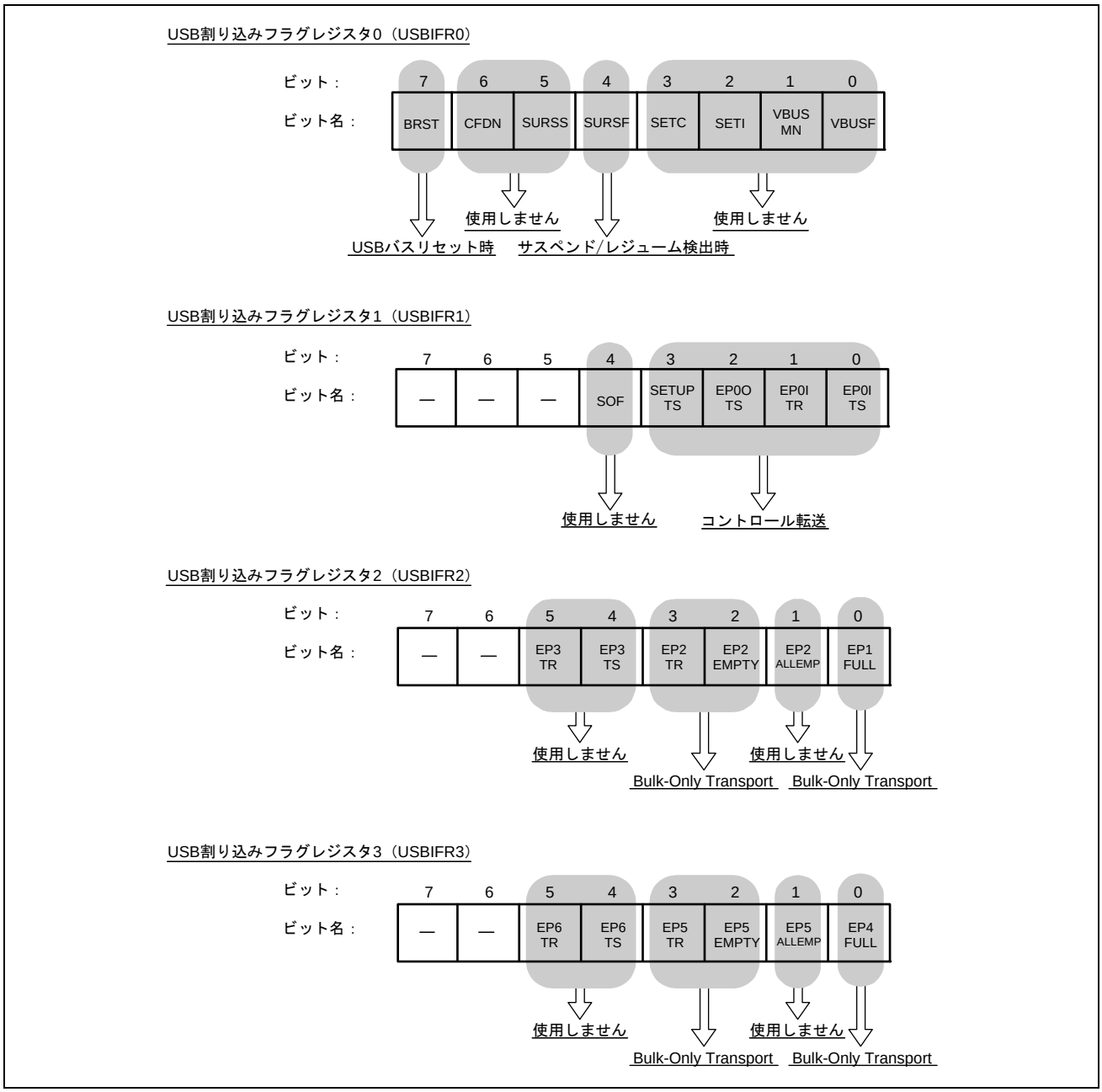


図 5.2 割り込みフラグの種類

5.2.1 各転送への分岐方法

サンプルプログラムでは、USB ファンクションモジュールから発生する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 BranchOfInt0、BranchOfInt1 が行います。**表 5.1** に割り込みの種類と、BranchOfInt0、BranchOfInt1 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
USBIFR0	7	BRST	ActBusReset
	6	CFDN	−
	5	SURSS	−
	4	SURSF	SuspendResume
	3	SETC	−
	2	SETI	−
	1	VBUSMN	−
	0	VBUSF	−
USBIFR1	7	−	−
	6	−	−
	5	−	−
	4	SOF	−
	3	SETUP TS	ActControl
	2	EP0O TS	ActControlInOut
	1	EP0I TR	ActControlInOut
	0	EP0I TS	ActControlInOut
USBIFR2	7	−	−
	6	−	−
	5	EP3 TR	−
	4	EP3 TS	−
	3	EP2TR	ActBulkOnly
	2	EP2 EMPTY	ActBulkInReady
	1	EP2 ALLEMP	−
	0	EP1FULL	ActBulkOnly
USBIFR3	7	−	−
	6	−	−
	5	EP6 TR	−
	4	EP6 TS	ActBulkOnly
	3	EP5TR	ActBulkInReady
	2	EP5 EMPTY	−
	1	EP5 ALLEMP	−
	0	EP4FULL	ActBulkOnly

EP0ITSとEP0OTS割り込みは、コントロールIN、コントロールOUT転送の両方で使用します。したがって、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 つのコントロール転送ステートをもっています。詳細は、**「5.6 コントロール転送」**を参照してください。

5.3 バスリセット時（BRST）割り込み

USB ホストは、USB データバスにファンクションが接続されると、バスリセット信号を出力します。M16C CPU Board が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。

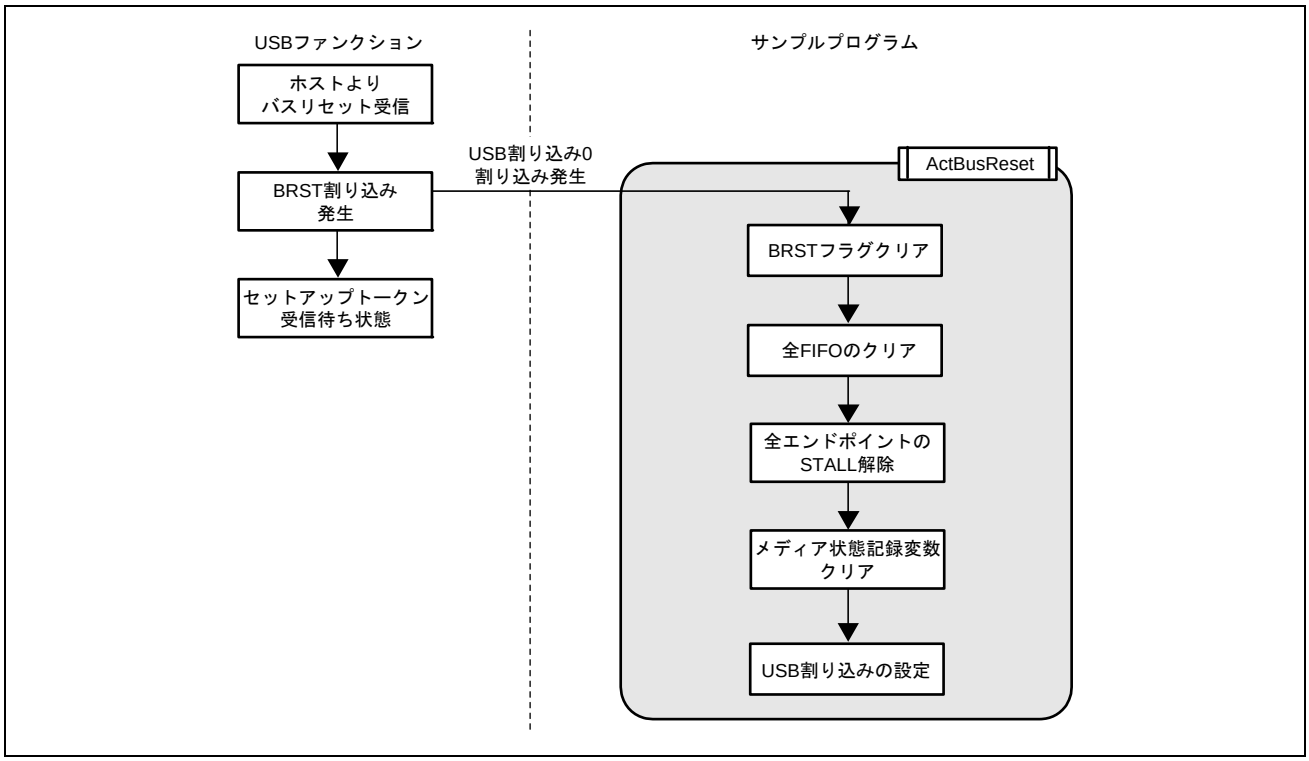


図 5.3　バスリセット割り込み

5.4　サスペンド/レジューム検出時（SURSF）割り込み

サスペンド/レジューム状態を検出時に発生します。

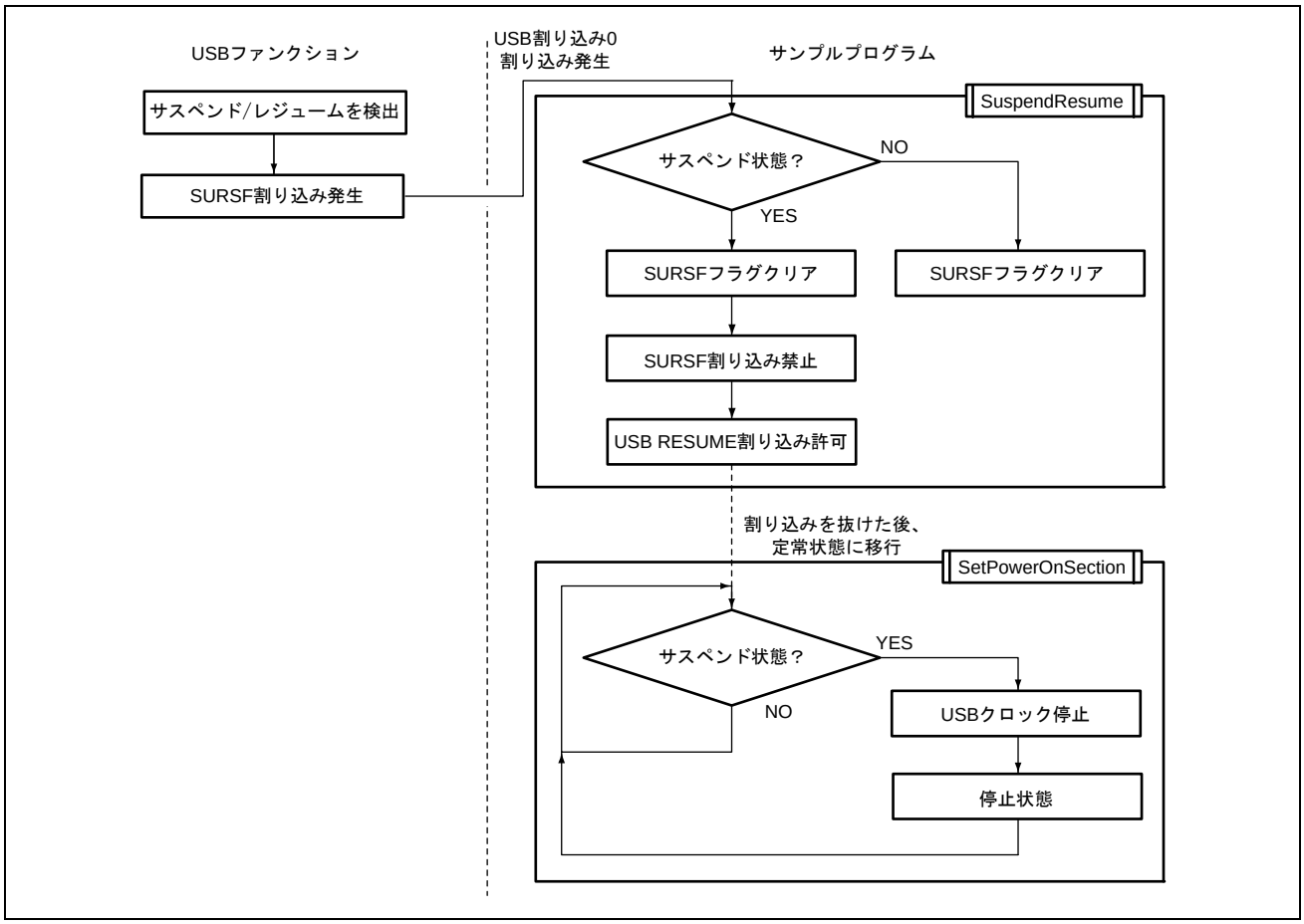


図 5.4　サスペンド/レジューム割り込み

5.5 エンドポイント設定

M16C/6C の USB ファンクションモジュールは、初期化時に、ソフトウェアでエンドポイント構成を設定する必要があります。設定可能な転送タイプを以下に示します。

- コントロール転送 ：1系統
- バルクOUT転送 ：2系統
- バルクIN転送 ：2系統
- インタラプトIN転送 ：2系統

コントロール転送以外は USB エンドポイント情報レジスタ(以下 USBEPIR)で、EndPoint 番号、Interface 番号、Alternate 番号を設定できます。

図 5.5 にこのサンプルプログラムのエンドポイント構成を示します。

表 5.2 にこのサンプルプログラムのエンドポイント構成を実現する USBEPIR の設定値を示します。詳細は M16C/6C ハードウェアマニュアルを参照してください。

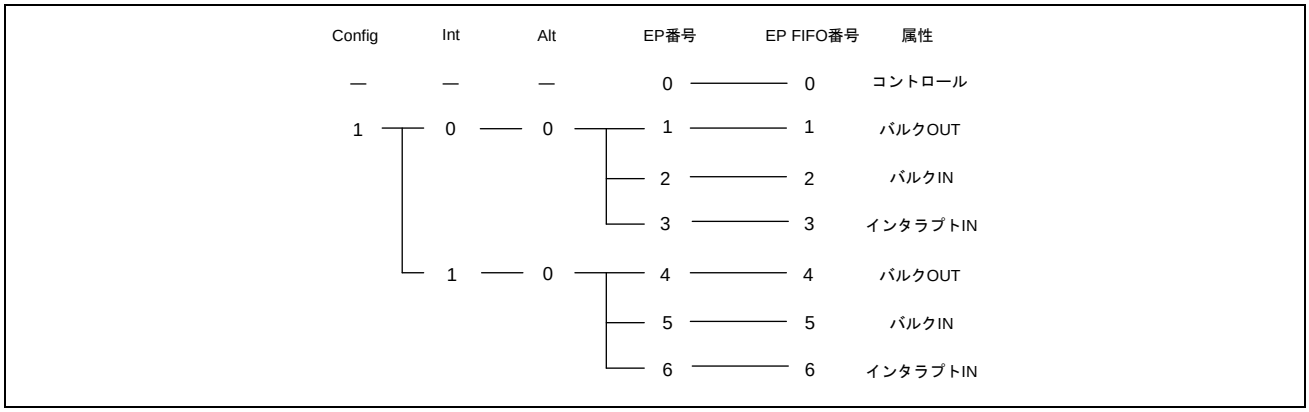


図 5.5 エンドポイント構成

表 5.2 USBEPIR の設定値

EPIR	設定値(16 進数)	転送タイプ	EP 番号	Conf	Int	Alt	MaxPacket Size	EPFIFO 番号
0	00_00_20_00_00	コントロール	0	—	—	—	16 バイト	0
1	14_20_80_00_01	バルク OUT	1	1	0	0	64 バイト	1
2	24_28_80_00_02	バルク IN	2	1	0	0	64 バイト	2
3	34_38_20_00_03	インタラプト IN	3	1	0	0	16 バイト	3
4	45_20_80_00_04	バルク OUT	4	1	1	0	64 バイト	4
5	55_28_80_00_05	バルク IN	5	1	1	0	64 バイト	5
6	65_38_20_00_06	インタラプト IN	6	1	1	0	16 バイト	6

5.6 コントロール転送

コントロール転送では、USB 割り込みフラグレジスタ 1 のビット 3～0 を使用します。コントロール転送は、セットアップステージ、データステージ（ない場合もあります）、ステータスステージで構成されています。データステージは、複数のトランザクションで構成されています。

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。USB ホストから USB ファンクションへデータ転送するのがコントロール OUT 転送、その逆がコントロール IN 転送です。

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データステージからステータスステージへ切り替わります。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 IN 転送または、コントロール OUT 転送を行う関数を呼び出します。このため、現在 IN、OUT 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ェアがコントロール転送ステートによって管理し（図 5.7 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コントロール転送ステート（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。

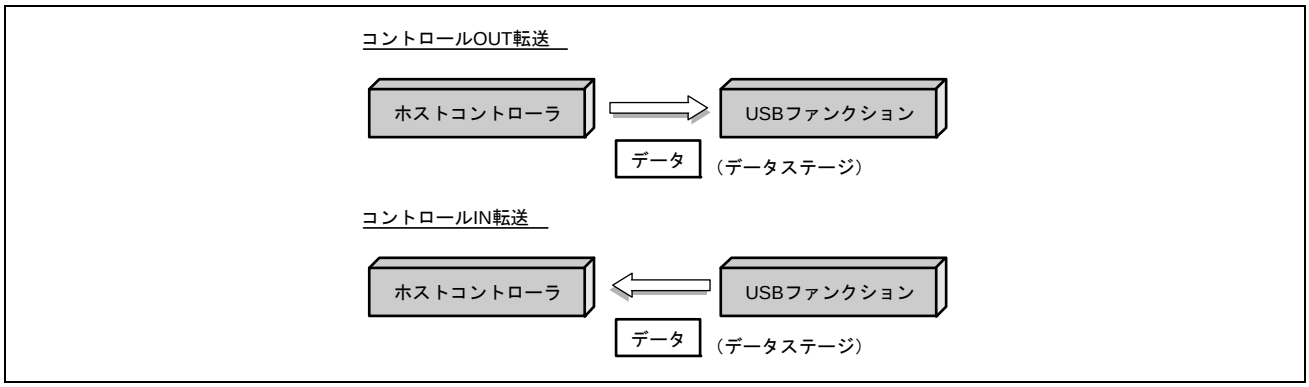


図 5.6 データステージにおけるデータの方向(コントロール転送)

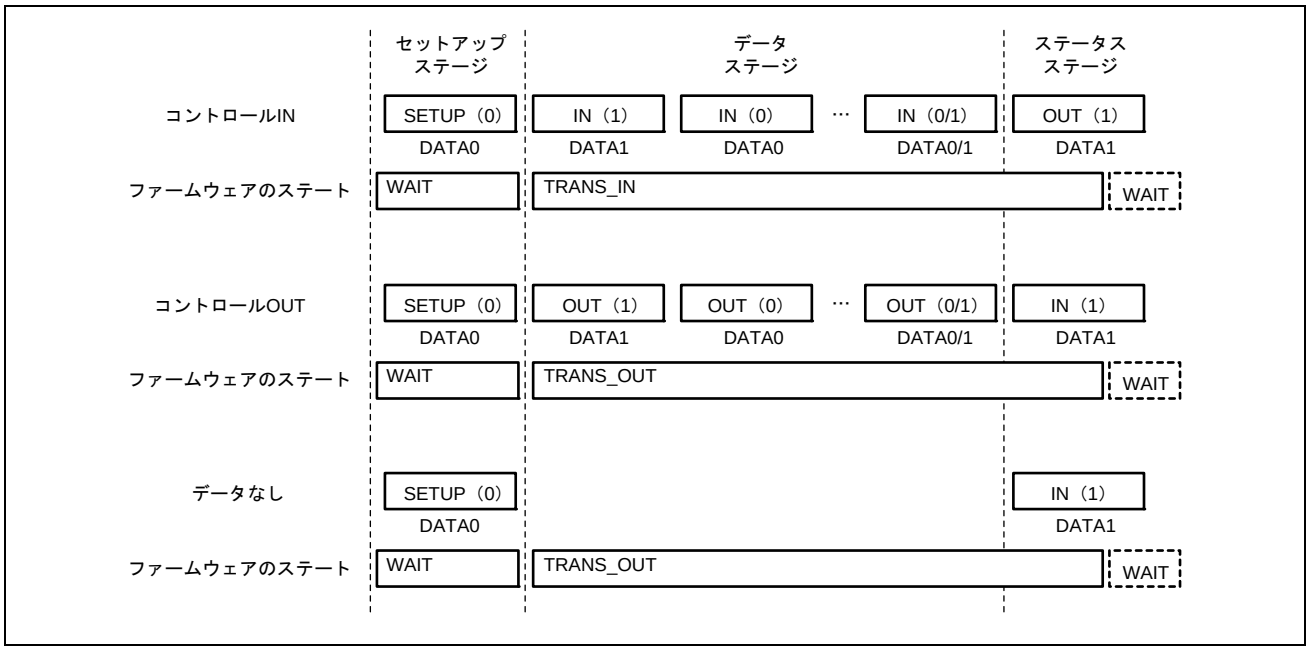


図 5.7　コントロール転送における各ステージの構成

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、USB ホストから USB ファンクションへコマンドを送信します。ファームウェアのコントロール転送ステートは WAIT です。また発行されるコマンドの種類によって、コントロール IN 転送またはコントロール OUT 転送の区別を行い、データステージにおけるファームウェアのコントロール転送ステート(TRANS_IN、TRANS_OUT)を決定します。

- コントロールINとなるコマンド　GetDescriptor　(TRANS_IN)　標準コマンド
  Get Max LUN　(TRANS_IN)　クラスコマンド
- コントロールOUTとなるコマンド　Bulk-Only Mass Storage Reset　(TRANS_OUT)　クラスコマンド

図 5.8 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。

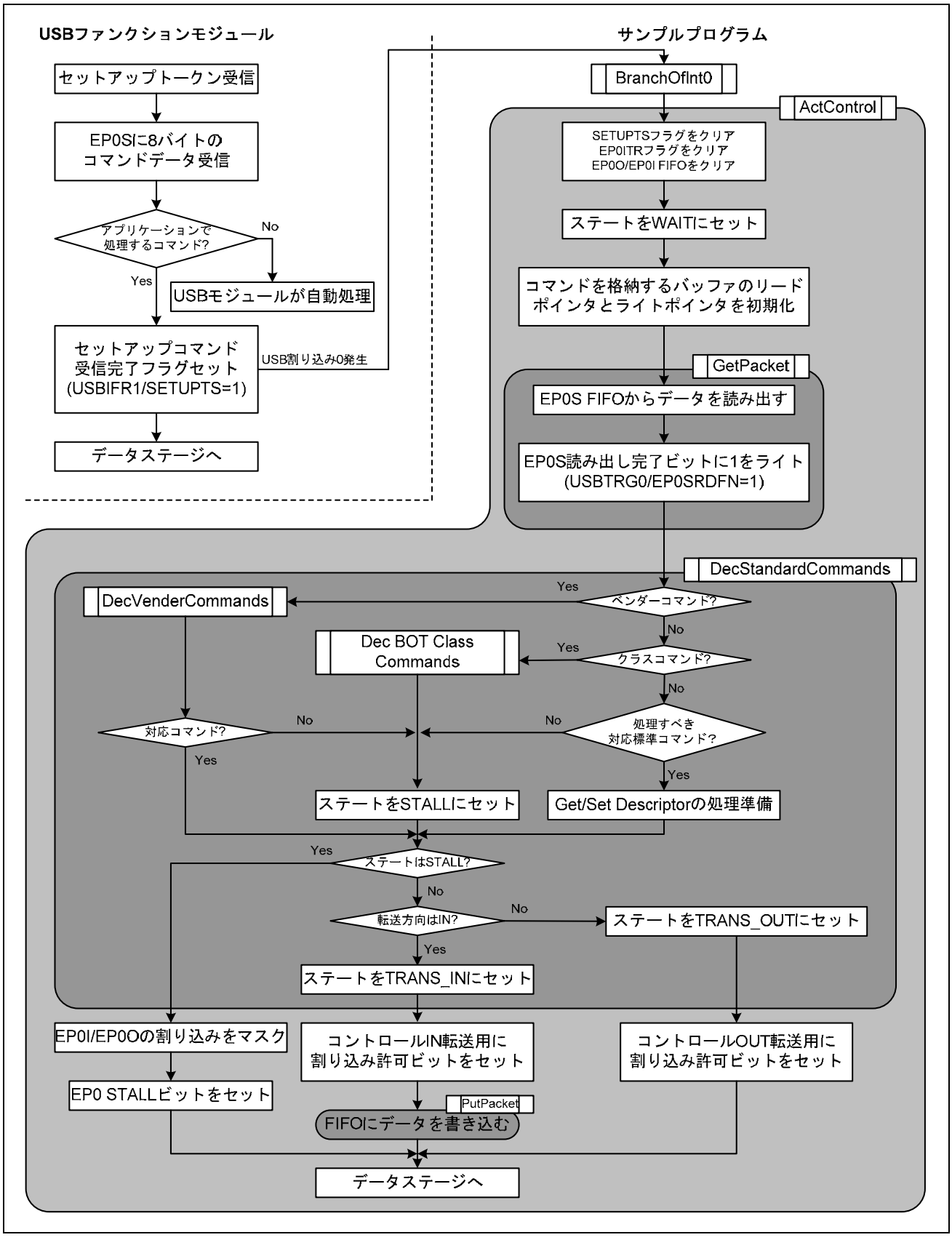


図 5.8　セットアップステージ

5.6.2 データステージ

データステージでは、USB ホストと USB ファンクションがデータの送受信を行います。ファームウェアのコントロール転送ステートは、セットアップステージで行ったコマンドのデコードの結果により、コントロール IN 転送であれば TRANS_IN に、コントロール OUT 転送であれば TRANS_OUT になります。

図 5.9、図 5.10 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。

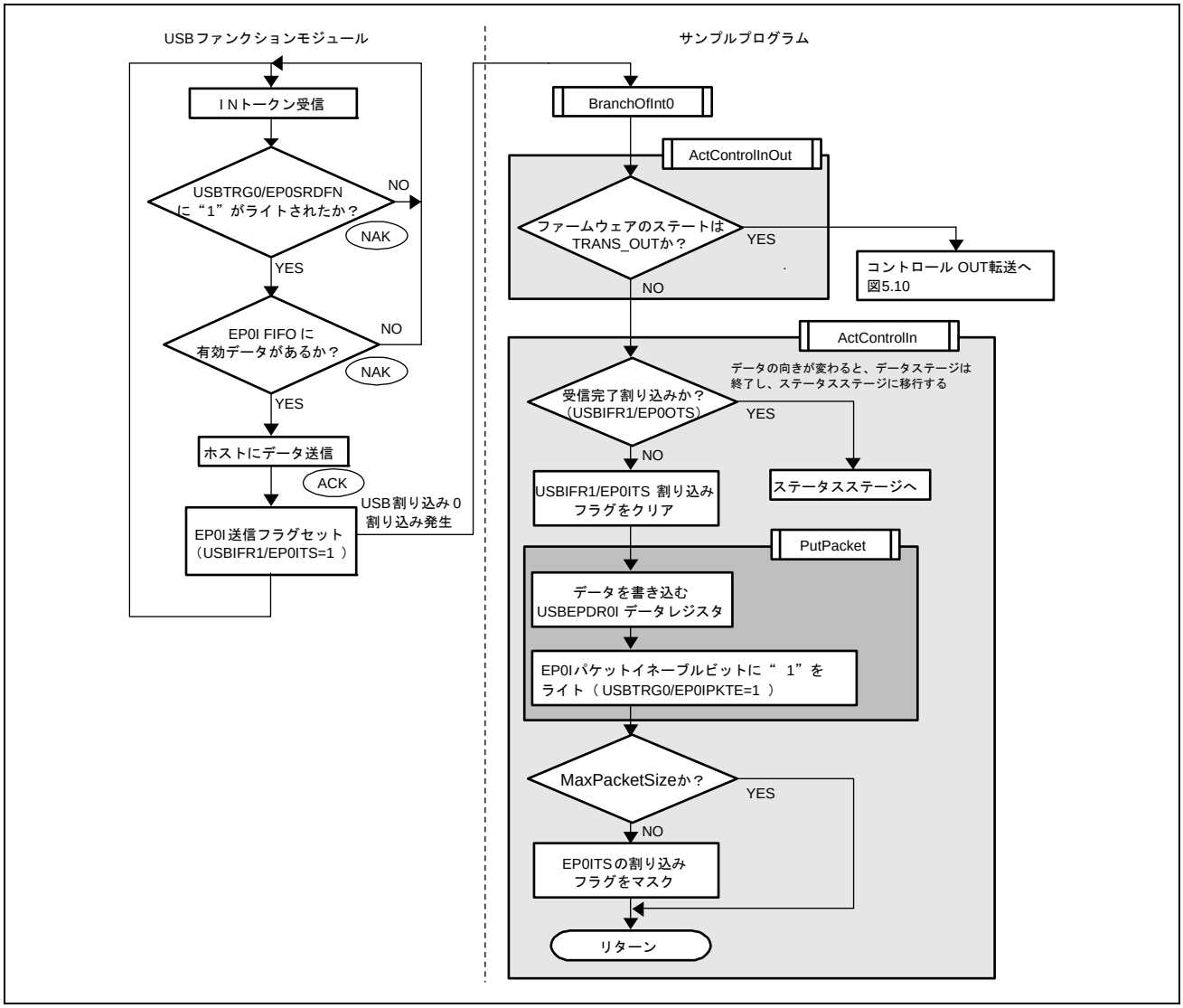


図 5.9　データステージ（コントロール IN 転送）

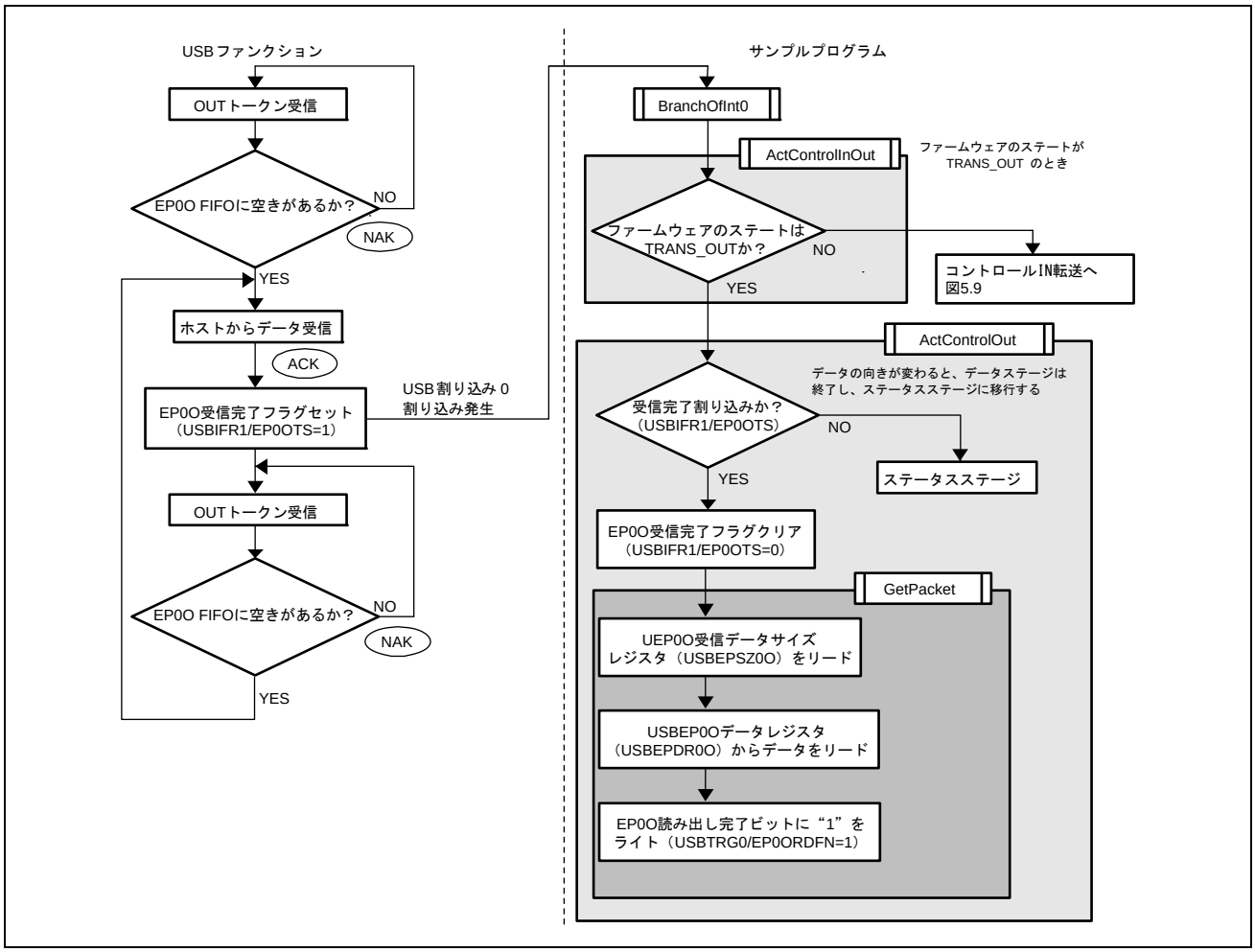


図 5.10　データステージ（コントロール OUT 転送）

5.6.3 ステータスステージ

ステータスステージは、データステージと逆方向のトークンによって開始されます。コントロール IN 転送では、USB ホストからの OUT トークンによってステータスステージへ移行します。コントロール OUT 転送では、USB ホストからの IN トークンによってステータスステージへ移行します。

図 5.11、図 5.12 にコントロール転送のステータス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。

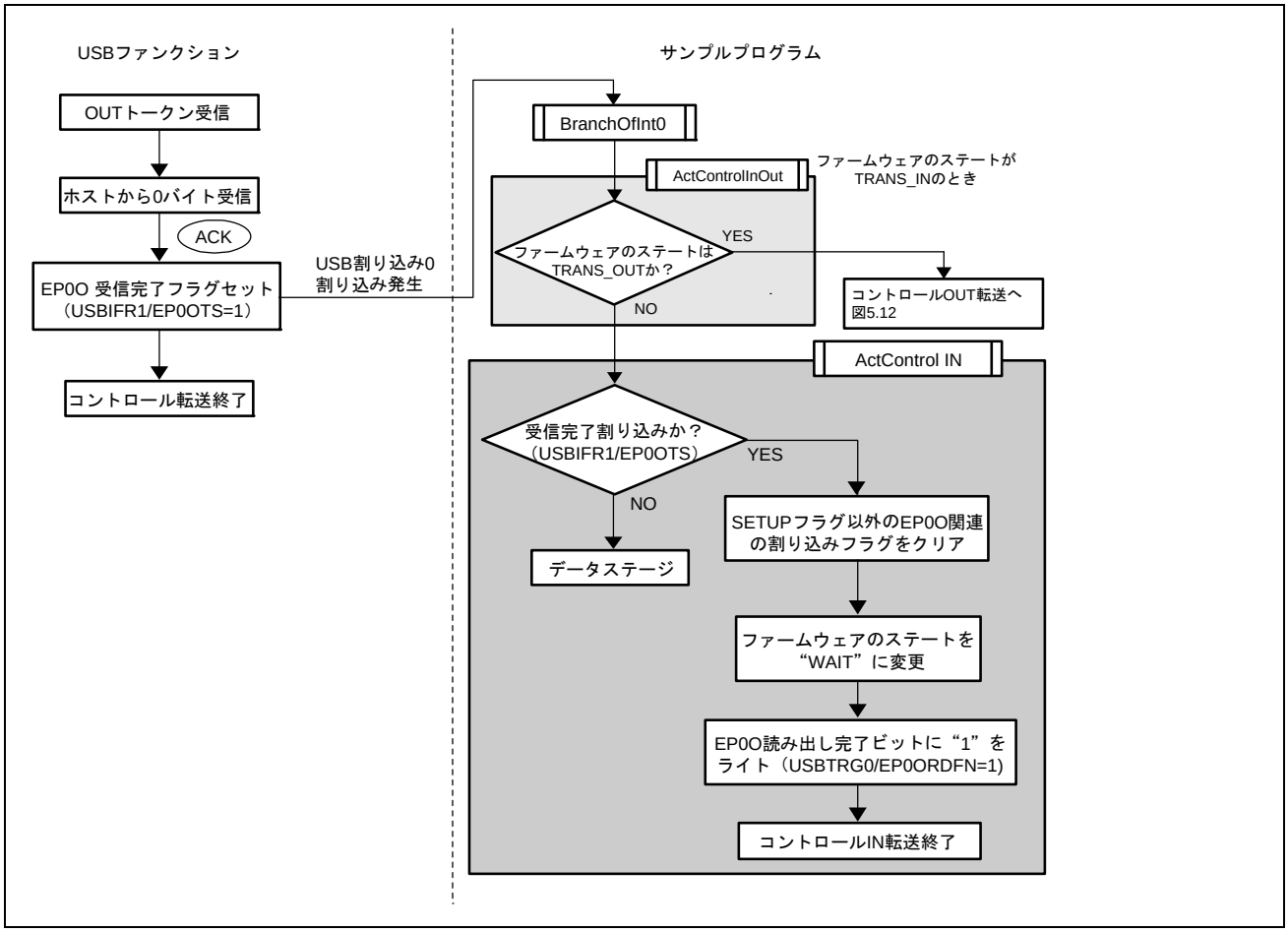


図 5.11　ステータスステージ（コントロール IN 転送）

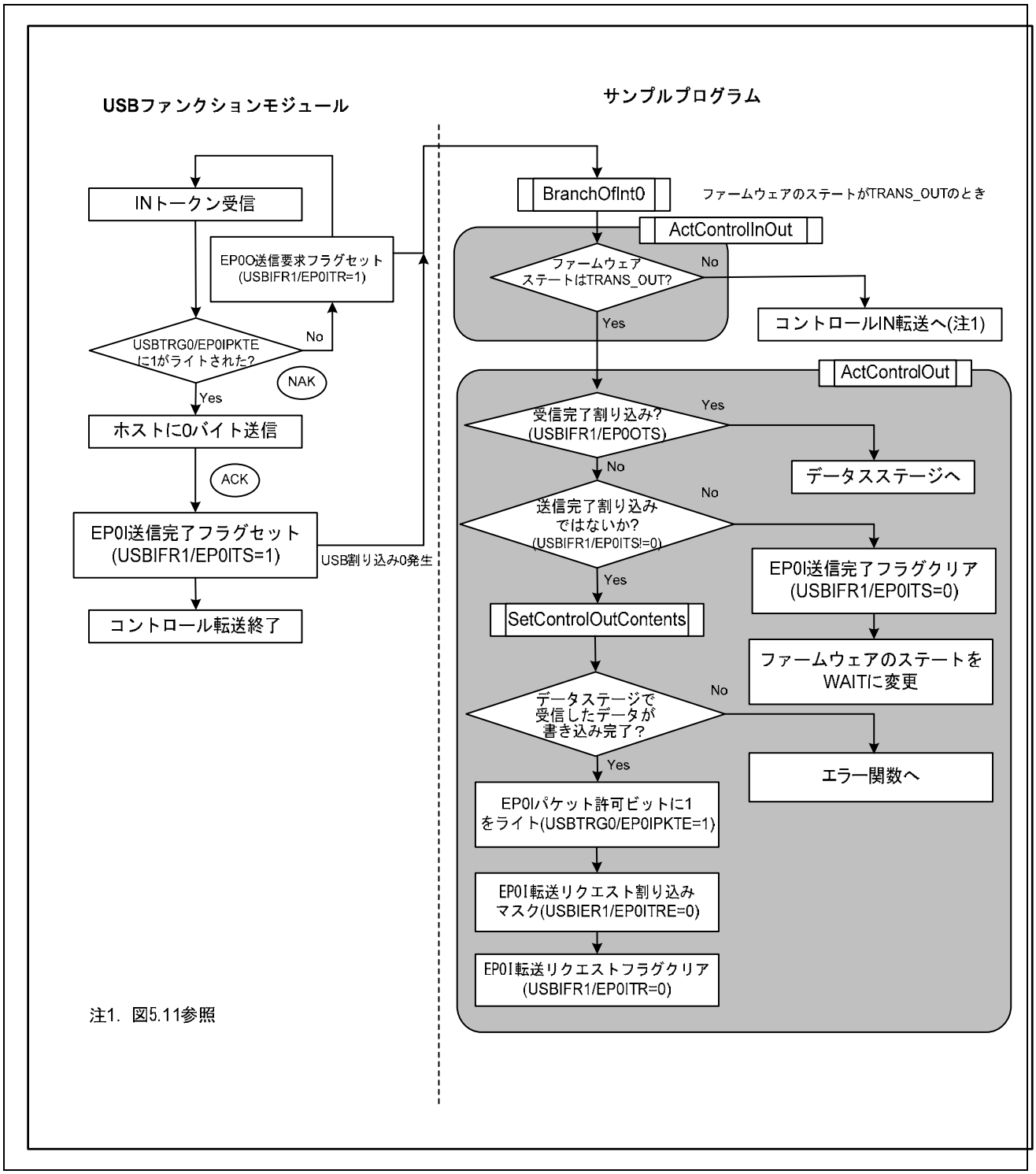


図 5.12　ステータスステージ（コントロール OUT 転送）

5.7 バルク転送

バルク転送では、USB 割り込みフラグレジスタ 2 のビット 0、2、3 を使用します。

バルク転送は、データの方向によって、2 つに分けることができます。USB ホストから USB ファンクションへデータを送信する場合がバルク OUT 転送、その逆がバルク IN 転送です。尚、以降の説明ではエンドポイント 1 と 2 を使用した場合を例に挙げ説明します。

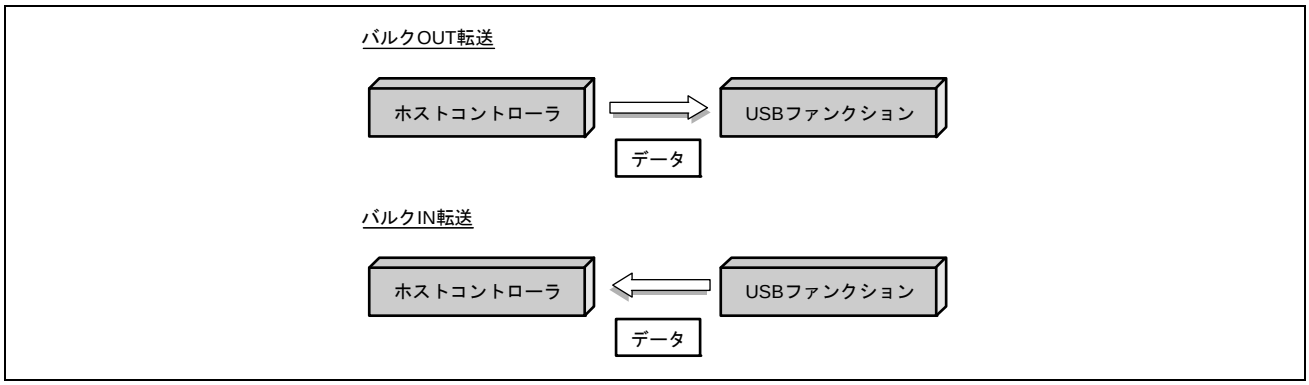


図 5.13　バルク転送

USB Mass Storage Class の Bulk-Only Transport は、バルク IN 転送とバルク OUT 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」（ない場合もあります）「ステータストランスポート（CSW）」の 2 または 3 つのステージで構成されます（図 5.14）。また、データトランスポートは、複数の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 OUT 転送、ステータストランスポート（CSW）はバルク IN 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 IN 転送、バルク OUT 転送の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 IN、バルク OUT 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。ファームウェアはデータトランスポートがバルク IN 転送、バルク OUT 転送のどちらの転送が行われるかを Bulk-Only Transport ステート（TRANS_IN、TRANS_OUT）で管理し（図 5.14 参照）、適切な関数を呼び出す必要があります。

また、コマンドトランスポート（CBW）で USB ホストが指定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完了することで、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遷移が行われます。ファームウェアはデータトランスポートで送信または受信したデータ長からステータストランスポートへの遷移を検出し、ステータスを USB ホスト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。

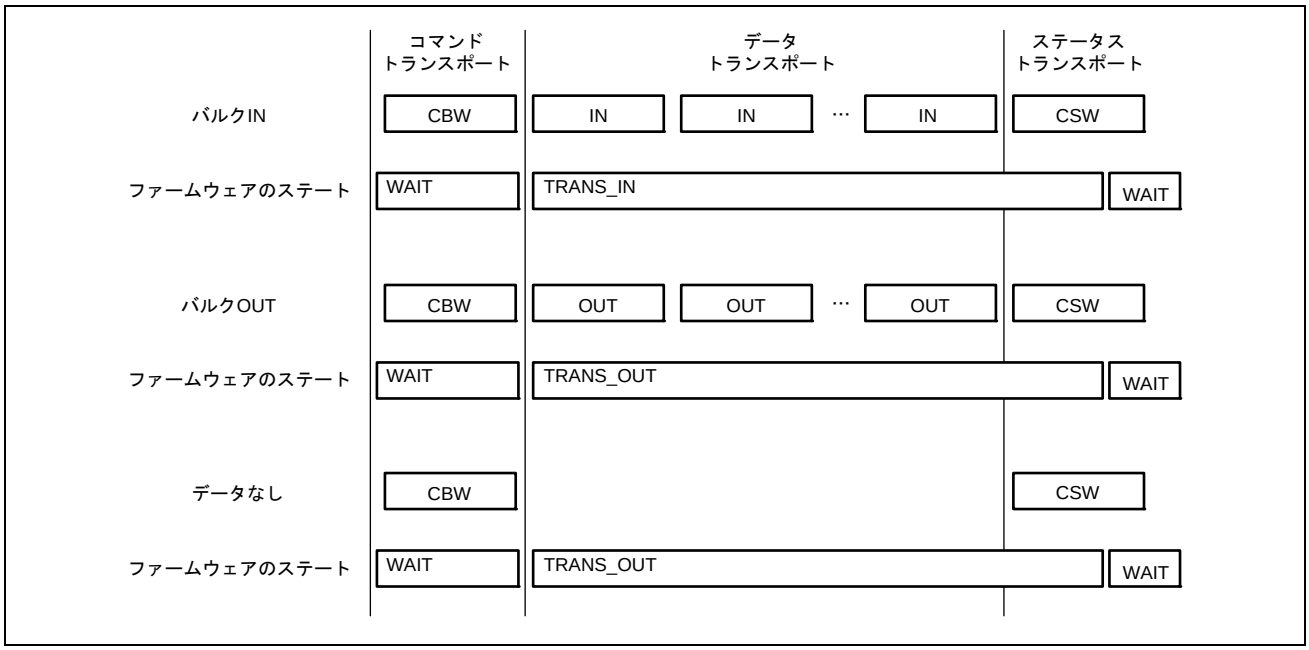


図 5.14　Bulk-Only Transport における各ステージ

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。

コマンドトランスポートでは、ファームウェアの Bulk-Only Transport ステートは WAIT です。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

CBWデータの有効判定

CSWデータの準備

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）

データトランスポートがバルクINまたはバルクOUT転送どちらかの区別を行い、ファームウェアの Bulk-Only Transportステート（TRANS_IN、TRANS_OUT）を決定

図 5.15 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表しています。

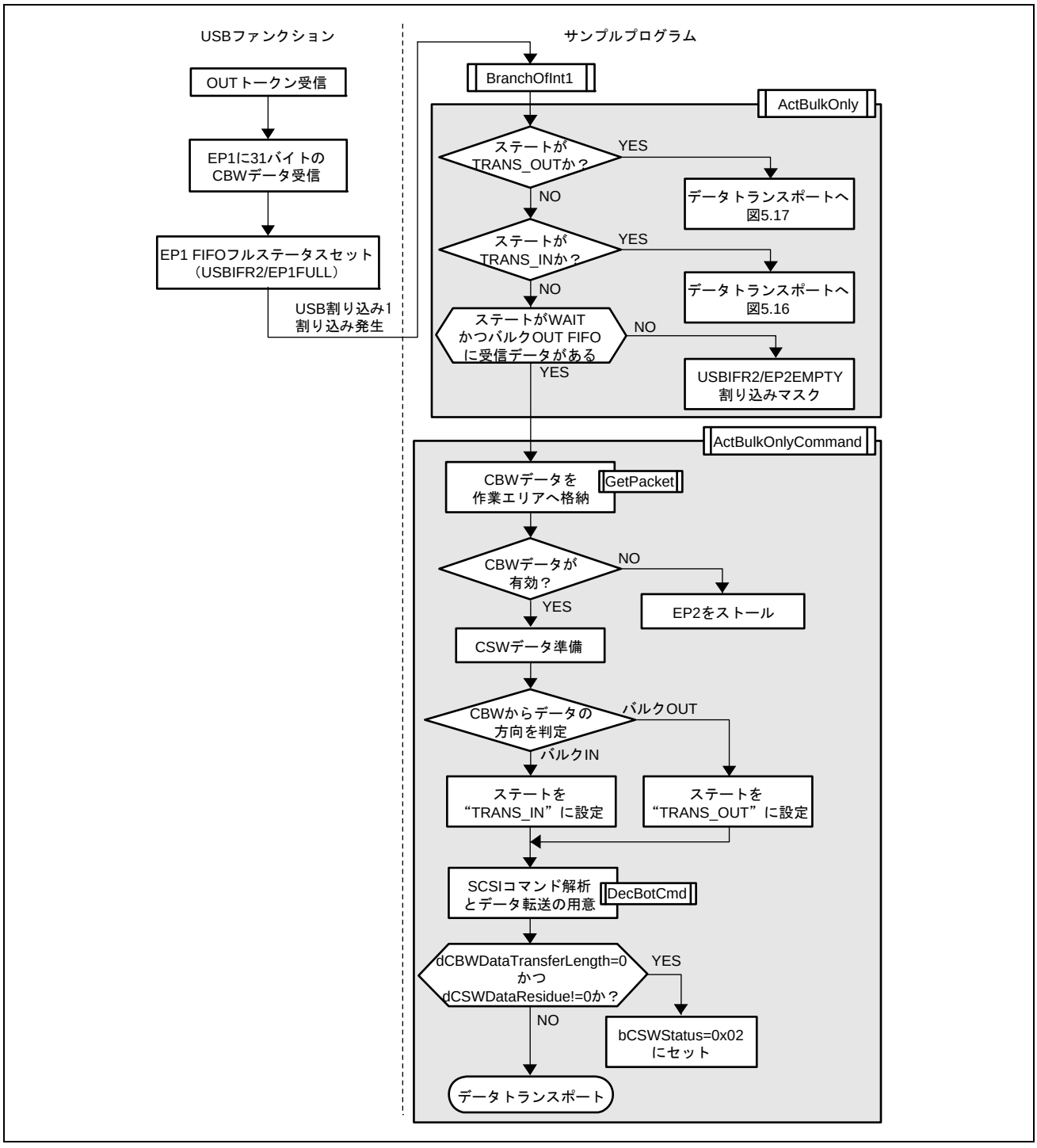


図 5.15　コマンドトランスポート

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。

ファームウェアの Bulk-Only Transport ステートは TRANS_IN または TRANS_OUT のどちらかです。

ファームウェアの Bulk-Only Transport ステートが TRANS_IN 状態の場合（バルク IN 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

図 5.16 にサンプルプログラムのデータトランスポート（バルク IN 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表しています。

このサンプルプログラムでは、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。

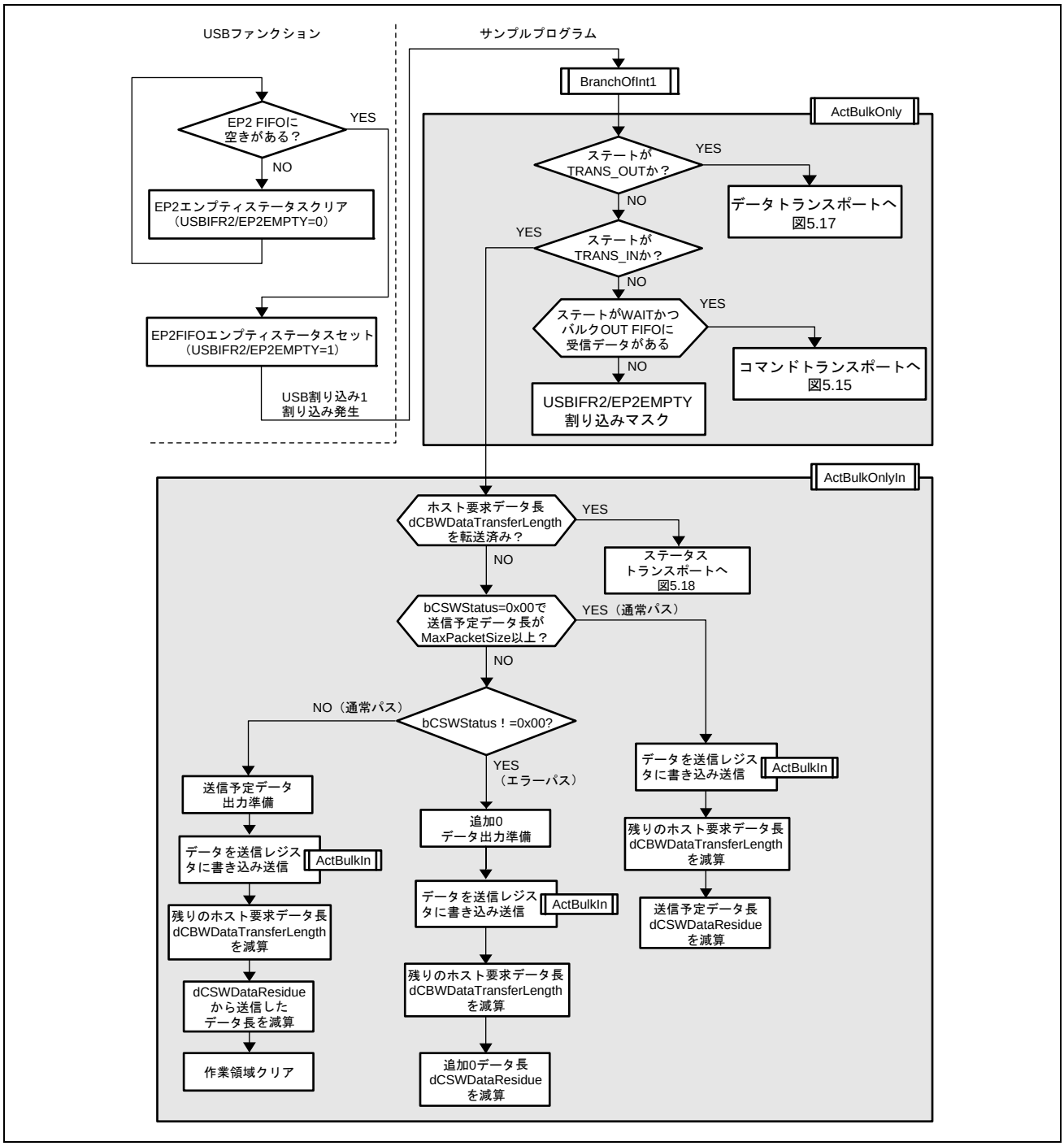


図 5.16　データトランスポート（バルク IN 転送）

図 5.17 にサンプルプログラムのデータトランスポート（バルク OUT 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表しています。

ファームウェアの Bulk-Only Transport ステートが TRANS_OUT 状態の場合（バルク OUT 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

このサンプルプログラムでは、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。

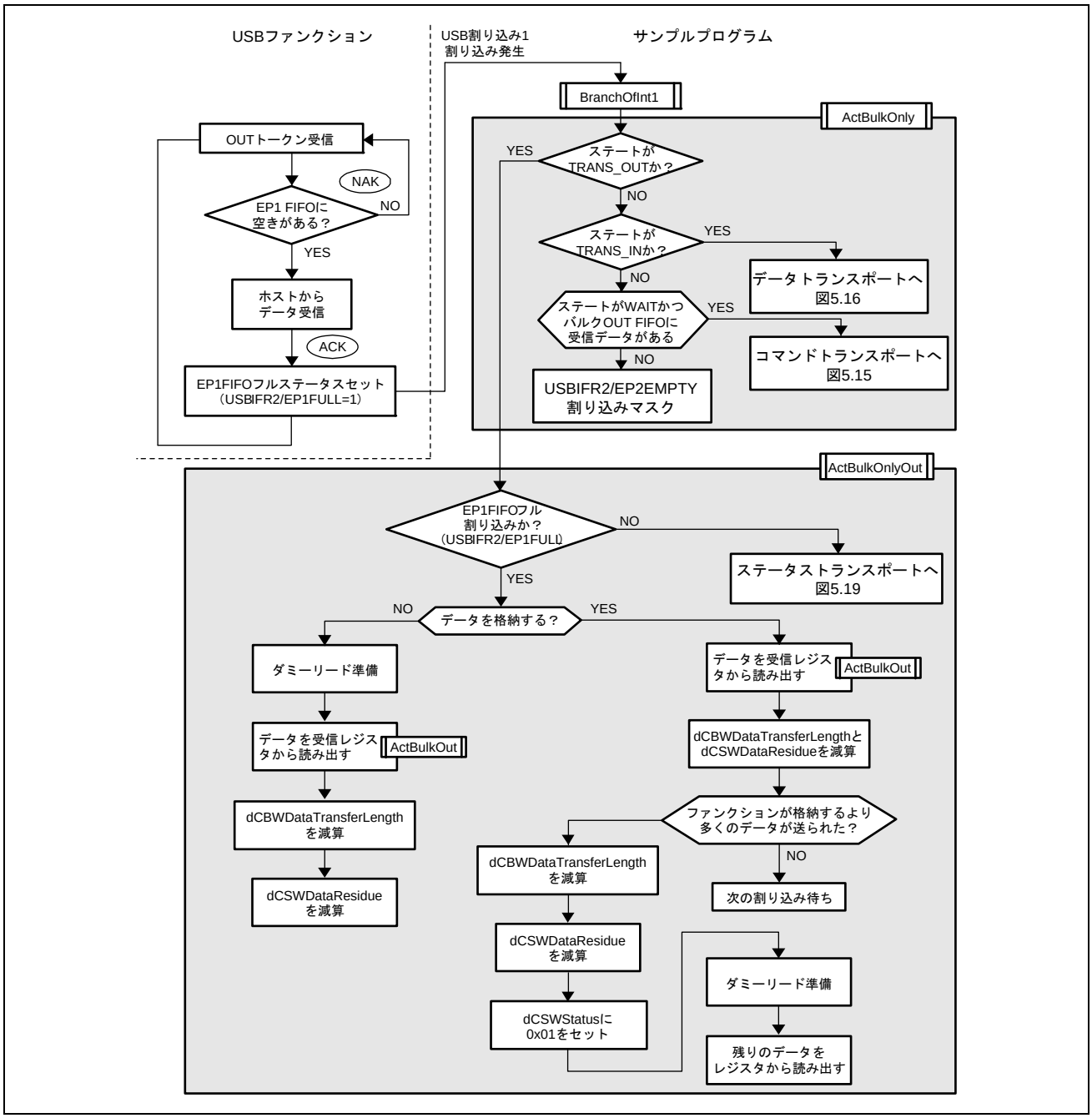


図 5.17　データトランスポート（バルク OUT 転送）

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。

ファームウェアの Bulk-Only Transport ステートは TRANS_IN または TRANS_OUT のどちらかです。ファームウェアの Bulk-Only Transport ステートが TRANS_IN 状態の場合（バルク IN 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. ファームウェアのBulk-Only TransportステートをWAITに設定

図 5.18 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はバルク IN 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表しています。

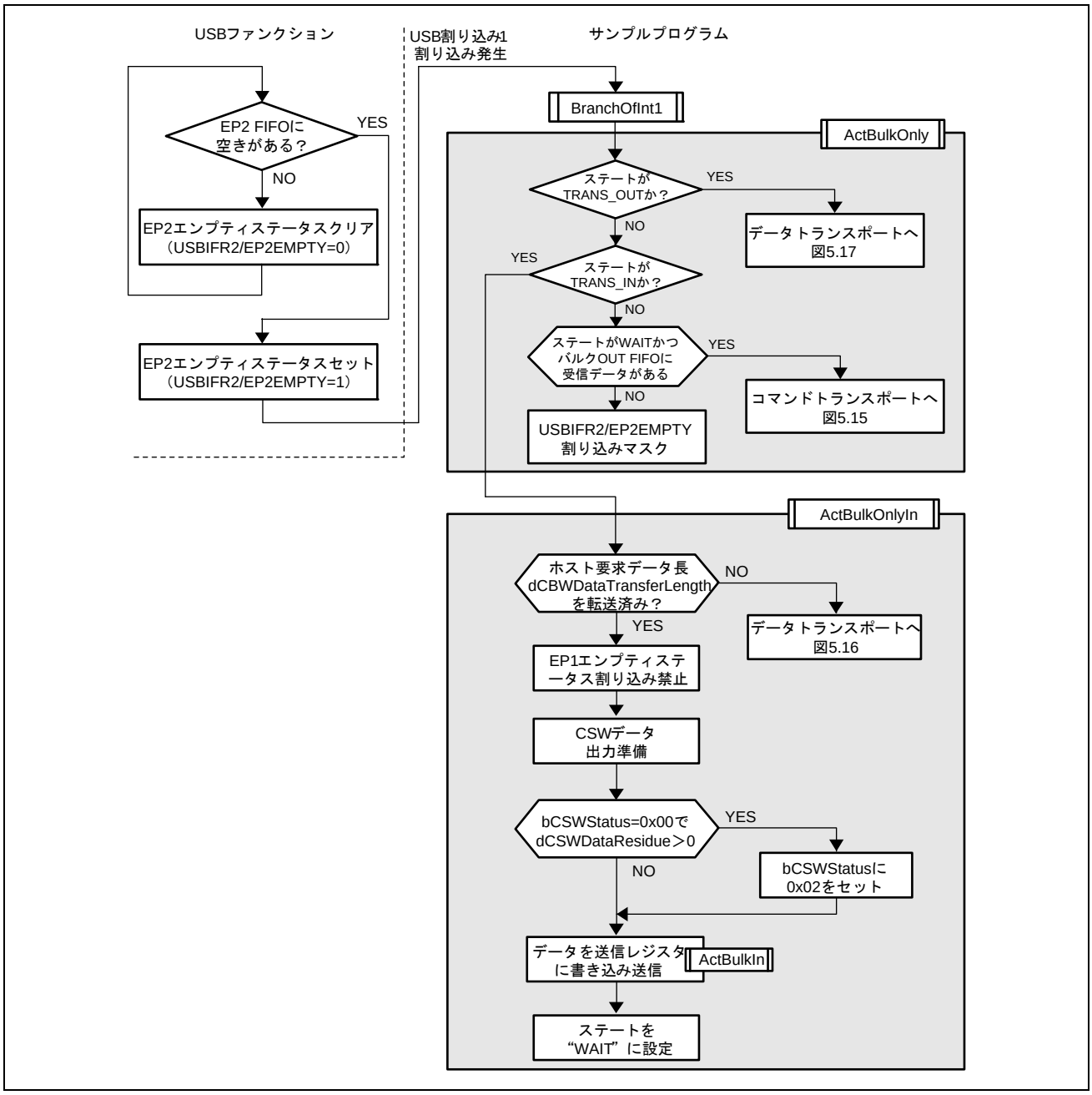


図 5.18　ステータストランスポート（データトランスポートはバルク IN 転送）

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はバルク OUT 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表しています。

ファームウェアの Bulk-Only Transport ステートが TRANS_OUT 状態の場合（バルク OUT 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. ファームウェアのBulk-Only TransportステートをWAITに設定

このサンプルプログラムでは、ホストが指定した予定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、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ます。ファンクションは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。

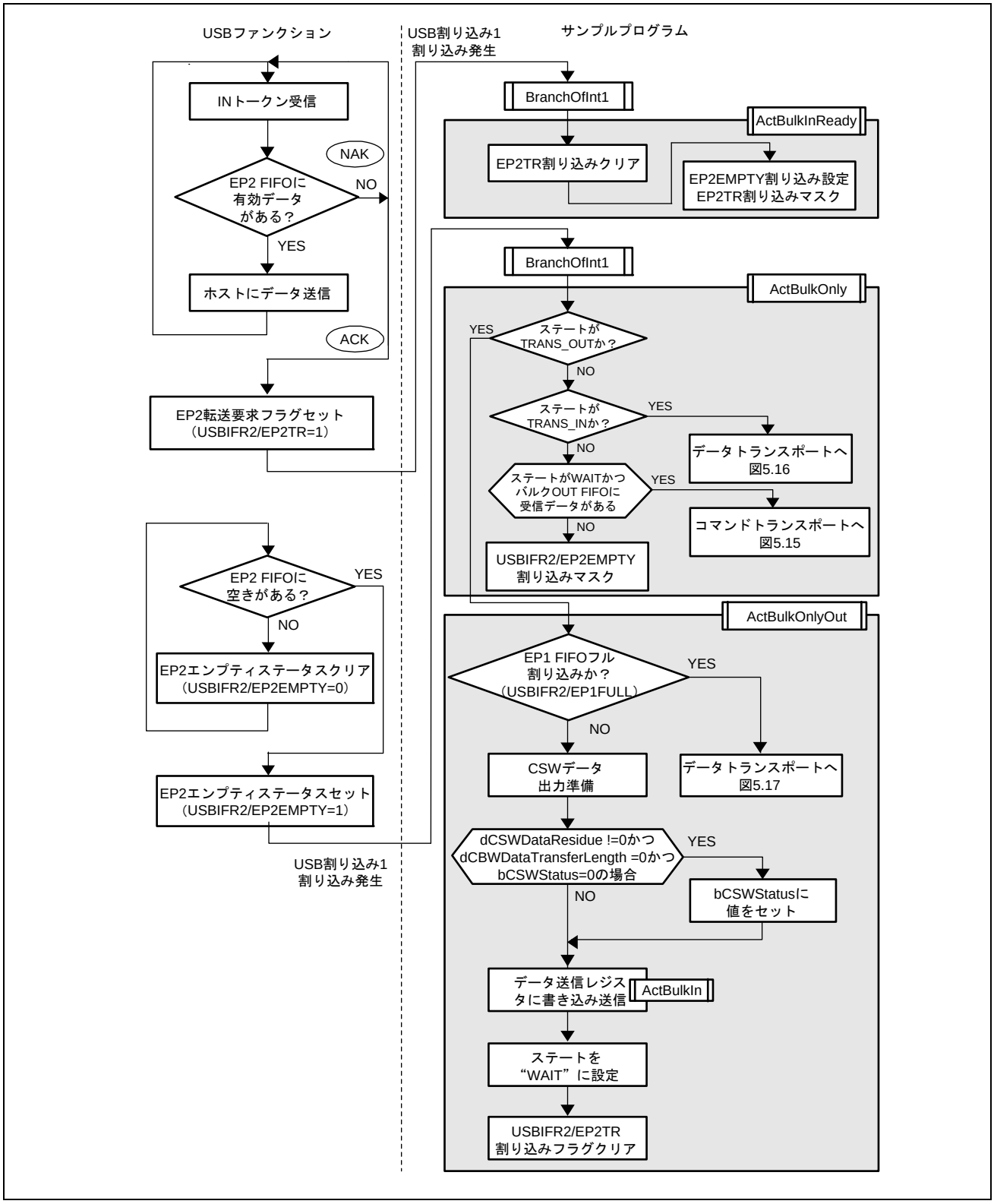


図 5.19　ステータストランスポート（データトランスポートはバルク OUT 転送）

6. アナライザのデータ

この章では、M16C/6C内蔵 USB ファンクションモジュールを使用して、ヒロテック社製 USB プロトコルアナライザ「HUSB200」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。

なお、各パケットの前部にある「No.」は測定時のトランザクションおよびパケット通し番号です。

以下に示す**図 6.1** は INQUIRY コマンドを測定したものです。

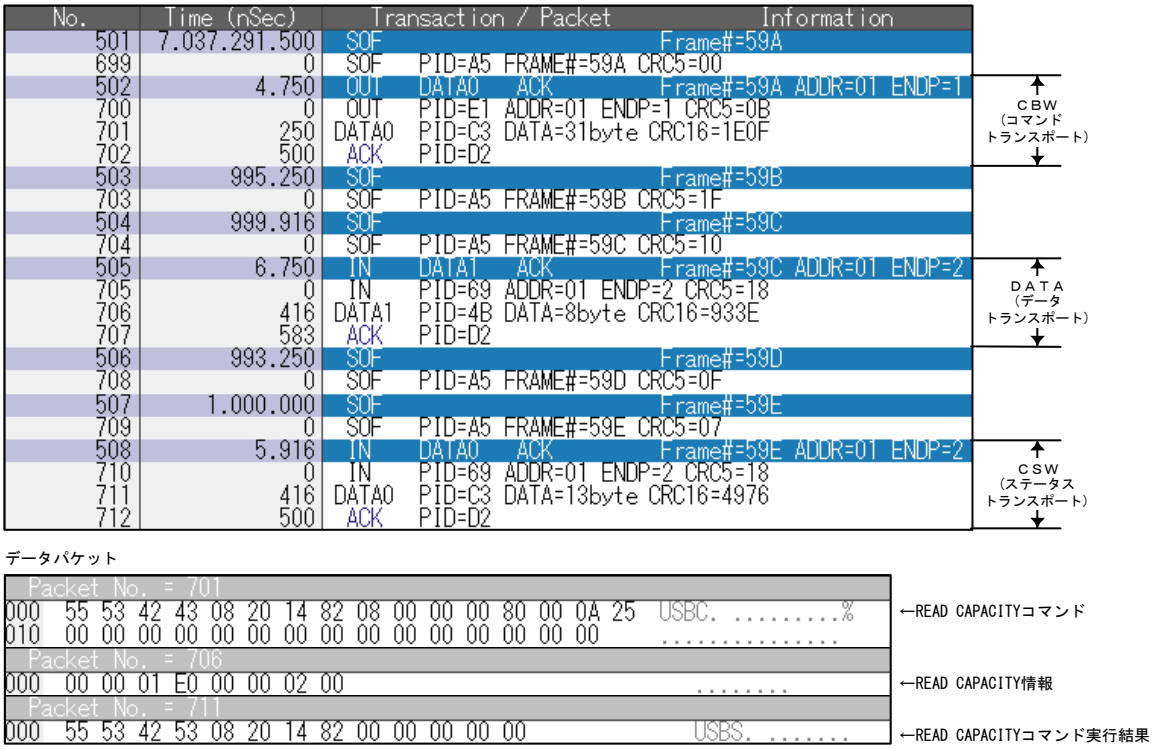
No.	Time (nSec)	Transaction / Packet	Information	
432	7.024.291.833	SOF	Frame#=58D	
565	0	SOF PID=A5 FRAME#=58D CRC5=13		
433	10.333	OUT DATA0 ACK	Frame#=58D ADDR=01 ENDP=1	↑ CBW（コマンドトランスポート）
566	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B		
567	250	DATA0 PID=C3 DATA=31byte CRC16=C0D5		
568	500	ACK PID=D2		↓
434	989.666	SOF	Frame#=58E	
569	0	SOF PID=A5 FRAME#=58E CRC5=1B		
435	1.000.000	SOF	Frame#=58F	
570	0	SOF PID=A5 FRAME#=58F CRC5=04		
436	10.166	IN DATA0 ACK	Frame#=58F ADDR=01 ENDP=2	↑ DATA（データトランスポート）
571	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
572	416	DATA0 PID=C3 DATA=36byte CRC16=0481		
573	500	ACK PID=D2		↓
437	989.750	SOF	Frame#=590	
574	0	SOF PID=A5 FRAME#=590 CRC5=19		
438	1.000.000	SOF	Frame#=591	
575	0	SOF PID=A5 FRAME#=591 CRC5=06		
439	10.583	IN DATA1 ACK	Frame#=591 ADDR=01 ENDP=2	↑ CSW（ステータストランスポート）
576	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
577	416	DATA1 PID=4B DATA=13byte CRC16=9AFD		
578	583	ACK PID=D2		↓

データパケット

```
Packet No. = 567
000  55 53 42 43 90 23 14 82 24 00 00 00 80 00 06 12  USBC.#..$.......   ← INQUIRYコマンド
010  00 00 00 24 00 00 00 00 00 00 00 00 00 00 00 00  ...$............
Packet No. = 572
000  00 80 02 02 5B 00 00 00 52 65 6E 65 73 61 73 20  ....[...Renesas    ←INQUIRY情報
010  45 58 20 52 41 4D 20 44 69 73 6B 20 20 20 20 20  EX RAM Disk
020  31 2E 31 32                                      1.12
Packet No. = 577
000  55 53 42 53 90 23 14 82 00 00 00 00 00           USBS.#.....       ←INQUIRY実行結果
```

図 6.1　INQUIRY コマンド

以下に示す図 6.2 は READ CAPACITY コマンド（通常時）を測定したものです。



No.	Time (nSec)	Transaction / Packet	Information	
501	7.037.291.500	SOF	Frame#=59A	
699	0	SOF PID=A5 FRAME#=59A CRC5=00		
502	4.750	OUT DATA0 ACK	Frame#=59A ADDR=01 ENDP=1	CBW（コマンドトランスポート）
700	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B		
701	250	DATA0 PID=C3 DATA=31byte CRC16=1E0F		
702	500	ACK PID=D2		
503	995.250	SOF	Frame#=59B	
703	0	SOF PID=A5 FRAME#=59B CRC5=1F		
504	999.916	SOF	Frame#=59C	
704	0	SOF PID=A5 FRAME#=59C CRC5=10		
505	6.750	IN DATA1 ACK	Frame#=59C ADDR=01 ENDP=2	DATA（データトランスポート）
705	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
706	416	DATA1 PID=4B DATA=8byte CRC16=933E		
707	583	ACK PID=D2		
506	993.250	SOF	Frame#=59D	
708	0	SOF PID=A5 FRAME#=59D CRC5=0F		
507	1.000.000	SOF	Frame#=59E	
709	0	SOF PID=A5 FRAME#=59E CRC5=07		
508	5.916	IN DATA0 ACK	Frame#=59E ADDR=01 ENDP=2	CSW（ステータストランスポート）
710	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
711	416	DATA0 PID=C3 DATA=13byte CRC16=4976		
712	500	ACK PID=D2		

データパケット

```
Packet No. = 701
000  55 53 42 43 08 20 14 82 08 00 00 00 80 00 0A 25  USBC. .........%
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 706
000  00 00 01 E0 00 00 02 00                          ........
Packet No. = 711
000  55 53 42 53 08 20 14 82 00 00 00 00 00           USBS. .......
```

←READ CAPACITYコマンド

←READ CAPACITY情報

←READ CAPACITYコマンド実行結果

図 6.2　READ CAPACITY コマンド（通常時）

以下に示す図 6.3 は READ CAPACITY コマンド（メディア取り出し状態時）を測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
4467	4.317.767.166	SOF	Frame#=3B5
4654	0	SOF PID=A5	FRAME#=3B5 CRC5=0E
4468	5.916	OUT DATA1 ACK	Frame#=3B5 ADDR=01 ENDP=1
4655	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
4656	250	DATA1 PID=4B	DATA=31byte CRC16=133C
4657	416	ACK PID=D2	
4469	994.083	SOF	Frame#=3B6
4658	0	SOF PID=A5	FRAME#=3B6 CRC5=06
4470	999.916	SOF	Frame#=3B7
4659	0	SOF PID=A5	FRAME#=3B7 CRC5=19
4471	6.416	IN DATA1 ACK	Frame#=3B7 ADDR=01 ENDP=2
4660	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4661	416	DATA1 PID=4B	DATA=8byte CRC16=F4BF
4662	666	ACK PID=D2	
4472	993.583	SOF	Frame#=3B8
4663	0	SOF PID=A5	FRAME#=3B8 CRC5=18
4473	999.916	SOF	Frame#=3B9
4664	0	SOF PID=A5	FRAME#=3B9 CRC5=07
4474	6.500	IN DATA0 ACK	Frame#=3B9 ADDR=01 ENDP=2
4665	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4666	500	DATA0 PID=C3	DATA=13byte CRC16=5EC5
4667	500	ACK PID=D2	

CBW（コマンドトランスポート）

DATA（データトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 4656
000  55 53 42 43 58 32 1A 82 08 00 00 00 80 00 0A 25  USBCX2.........%
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 4661
000  00 00 00 00 00 00 00 00                          ........
Packet No. = 4666
000  55 53 42 53 58 32 1A 82 08 00 00 00 01           USBSX2.....
```

←READ CAPACITYコマンド

←READ CAPACITY情報（無効データ）

←READ CAPACITYコマンド実行結果（コマンドフェイル）

図 6.3　READ CAPACITY コマンド（メディア取り出し状態時）

以下に示す図 6.4 は READ（10）コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
510	7.043.291.333	SOF	Frame#=5A0
714	0	SOF PID=A5 FRAME#=5A0 CRC5=14	
511	5.833	OUT DATA1 ACK	Frame#=5A0 ADDR=01 ENDP=1
715	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B	
716	250	DATA1 PID=4B DATA=31byte CRC16=38CE	
717	416	ACK PID=D2	
512	994.166	SOF	Frame#=5A1
718	0	SOF PID=A5 FRAME#=5A1 CRC5=0B	
513	999.916	SOF	Frame#=5A2
719	0	SOF PID=A5 FRAME#=5A2 CRC5=03	
514	7.166	IN DATA1 ACK	Frame#=5A2 ADDR=01 ENDP=2
720	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
721	500	DATA1 PID=4B DATA=64byte CRC16=D0BF	
722	583	ACK PID=D2	
515	57.500	IN DATA0 ACK	Frame#=5A2 ADDR=01 ENDP=2
723	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
724	500	DATA0 PID=C3 DATA=64byte CRC16=D0BF	
725	583	ACK PID=D2	
		⋮	
571	217.750	SOF	Frame#=5A3
840	0	SOF PID=A5 FRAME#=5A3 CRC5=1C	
572	5.916	IN DATA1 ACK	Frame#=5A3 ADDR=01 ENDP=2
841	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
842	416	DATA1 PID=4B DATA=64byte CRC16=10DE	
843	416	ACK PID=D2	
581	146.916	IN DATA0 ACK	Frame#=5A3 ADDR=01 ENDP=2
860	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
861	416	DATA0 PID=C3 DATA=64byte CRC16=34E6	
862	416	ACK PID=D2	
582	847.166	SOF	Frame#=5A4
863	0	SOF PID=A5 FRAME#=5A4 CRC5=13	
583	999.916	SOF	Frame#=5A5
864	0	SOF PID=A5 FRAME#=5A5 CRC5=0C	
584	6.166	IN DATA1 ACK	Frame#=5A5 ADDR=01 ENDP=2
865	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
866	500	DATA1 PID=4B DATA=13byte CRC16=4976	
867	500	ACK PID=D2	

CBW（コマンドトランスポート）

DATA（データトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 716
000  55 53 42 43 08 20 14 82 00 02 00 00 80 00 0A 28  USBC. .........(
010  00 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00  ................
Packet No. = 721
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 724
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
                          ⋮
Packet No. = 842
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 00  ................
Packet No. = 861
000  00 00 01 00 00 00 20 00 00 00 E0 01 00 00 00 00  ...... ........
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA  ..............U.
Packet No. = 866
000  55 53 42 53 08 20 14 82 00 00 00 00 00           USBS. .......
```

←READ(10)コマンド

←READ(10)情報

←READ(10)コマンド実行結果

図 6.4　READ（10）コマンド

以下に示す図 6.5 は WRITE（10）コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
2920	2.851.273.083	SOF	Frame#=680
3008	0	SOF PID=A5	FRAME#=680 CRC5=0A
2921	12.416	OUT DATA0 ACK	Frame#=680 ADDR=01 ENDP=1
3009	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
3010	250	DATA0 PID=C3	DATA=31byte CRC16=3058
3011	416	ACK PID=D2	
2922	987.500	SOF	Frame#=681
3012	0	SOF PID=A5	FRAME#=681 CRC5=15
2923	1.000.000	SOF	Frame#=682
3013	0	SOF PID=A5	FRAME#=682 CRC5=1D
2924	13.083	OUT DATA1 ACK	Frame#=682 ADDR=01 ENDP=1
3014	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
3015	250	DATA1 PID=4B	DATA=64byte CRC16=3141
3016	416	ACK PID=D2	
2925	60.750	OUT DATA0 ACK	Frame#=682 ADDR=01 ENDP=1
3017	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
3018	250	DATA0 PID=C3	DATA=64byte CRC16=D0BF
3019	583	ACK PID=D2	
⋮			
3152	240.916	OUT DATA1 ACK	Frame#=68F ADDR=01 ENDP=1
3672	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
3673	250	DATA1 PID=4B	DATA=64byte CRC16=D0BF
3674	583	ACK PID=D2	
3156	241.166	OUT DATA0 ACK	Frame#=68F ADDR=01 ENDP=1
3684	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
3685	250	DATA0 PID=C3	DATA=64byte CRC16=D0BF
3686	583	ACK PID=D2	

CBW（コマンドトランスポート）

DATA（データトランスポート）

データパケット

```
Packet No. = 3010
000  55 53 42 43 78 15 92 81 00 10 00 00 00 00 0A 2A  USBCx.........*
010  00 00 00 00 23 00 00 08 00 00 00 00 00 00 00 00      ....#.........
Packet No. = 3015
000  4D 53 43 20 20 20 20 20 54 58 54 20 18 22 77 86  MSC     TXT .”w.
010  52 3A 52 3A 00 00 78 86 52 3A 00 00 00 00 00 00  R:R:..x.R:......
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 3018
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 3030
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
                        ⋮
Packet No. = 3673
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 3685
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

←WRITE(10)コマンド

←WRITE(10)情報

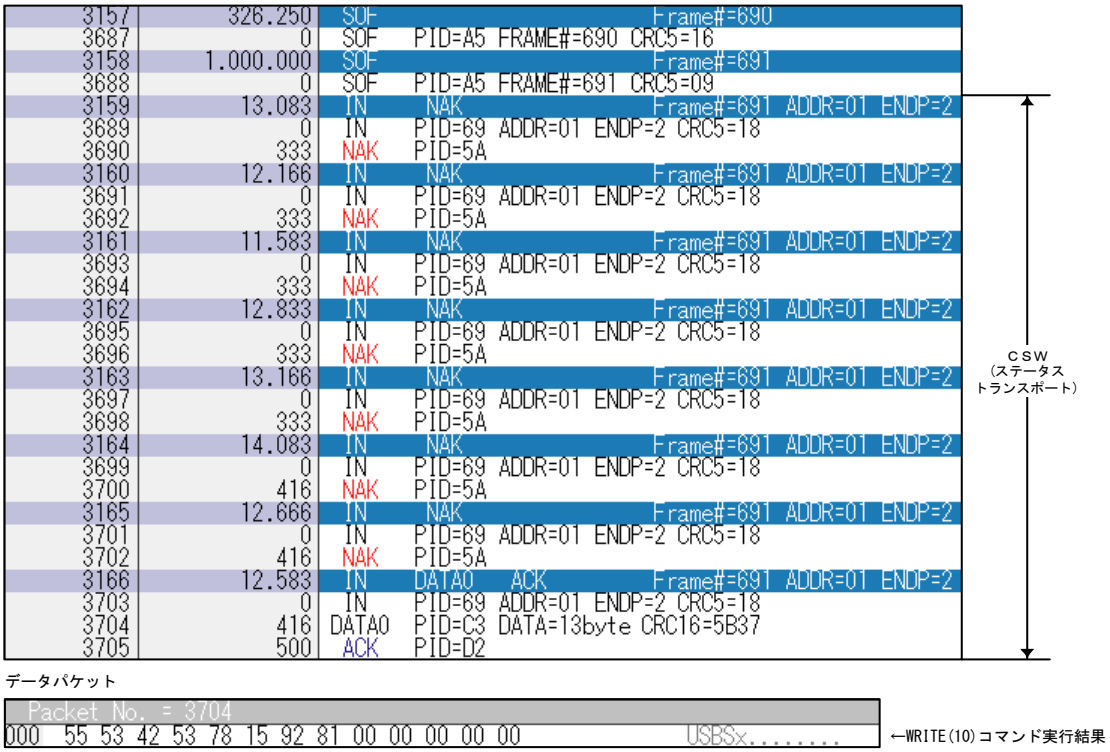
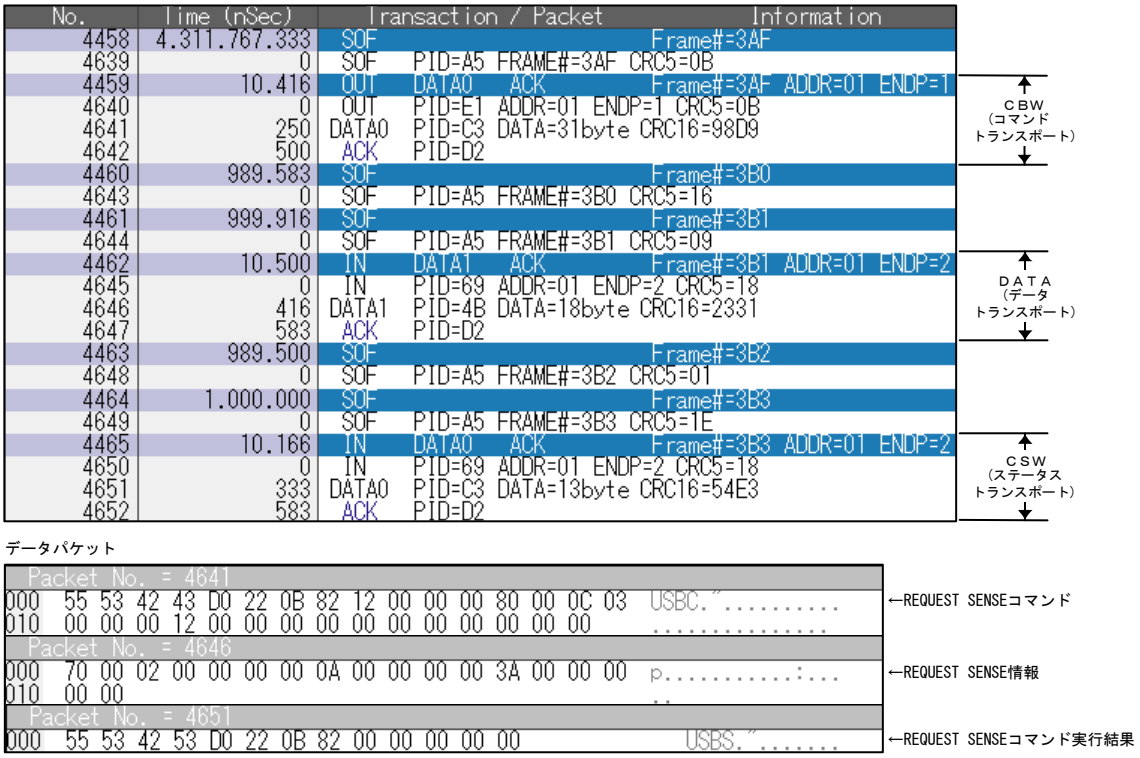


図 6.5　WRITE（10）コマンド

以下に示す図 6.6 は REQUEST SENSE コマンドを測定したものです。



No.	Time (nSec)	Transaction / Packet	Information
4458	4.311.767.333	SOF	Frame#=3AF
4639	0	SOF PID=A5	FRAME#=3AF CRC5=0B
4459	10.416	OUT DATA0 ACK	Frame#=3AF ADDR=01 ENDP=1
4640	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
4641	250	DATA0 PID=C3	DATA=31byte CRC16=98D9
4642	500	ACK PID=D2	
4460	989.583	SOF	Frame#=3B0
4643	0	SOF PID=A5	FRAME#=3B0 CRC5=16
4461	999.916	SOF	Frame#=3B1
4644	0	SOF PID=A5	FRAME#=3B1 CRC5=09
4462	10.500	IN DATA1 ACK	Frame#=3B1 ADDR=01 ENDP=2
4645	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4646	416	DATA1 PID=4B	DATA=18byte CRC16=2331
4647	583	ACK PID=D2	
4463	989.500	SOF	Frame#=3B2
4648	0	SOF PID=A5	FRAME#=3B2 CRC5=01
4464	1.000.000	SOF	Frame#=3B3
4649	0	SOF PID=A5	FRAME#=3B3 CRC5=1E
4465	10.166	IN DATA0 ACK	Frame#=3B3 ADDR=01 ENDP=2
4650	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4651	333	DATA0 PID=C3	DATA=13byte CRC16=54E3
4652	583	ACK PID=D2	

CBW（コマンドトランスポート）

DATA（データトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 4641
000  55 53 42 43 D0 22 0B 82 12 00 00 00 80 00 0C 03  USBC."..........
010  00 00 00 12 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 4646
000  70 00 02 00 00 00 00 0A 00 00 00 00 3A 00 00 00  p...........:...
010  00 00                                            ..
Packet No. = 4651
000  55 53 42 53 D0 22 0B 82 00 00 00 00 00           USBS."........
```

←REQUEST SENSEコマンド

←REQUEST SENSE情報

←REQUEST SENSEコマンド実行結果

図 6.6　REQUEST SENSE コマンド

以下に示す図 6.7 は PREVENT ALLOW MEDIUM REMOVAL コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
3974	3.854.780.916	SOF	Frame#=1E6
4122	0	SOF PID=A5	FRAME#=1E6 CRC5=1B
3975	6.500	OUT DATA1 ACK	Frame#=1E6 ADDR=01 ENDP=1
4123	0	OUT PID=E1	ADDR=01 ENDP=1 CRC5=0B
4124	250	DATA1 PID=4B	DATA=31byte CRC16=B187
4125	416	ACK PID=D2	
3976	993.416	SOF	Frame#=1E7
4126	0	SOF PID=A5	FRAME#=1E7 CRC5=04
3977	1.000.000	SOF	Frame#=1E8
4127	0	SOF PID=A5	FRAME#=1E8 CRC5=05
3978	10.500	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4128	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4129	416	NAK PID=5A	
3979	12.666	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4130	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4131	416	NAK PID=5A	
3980	12.166	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4132	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4133	416	NAK PID=5A	
3981	13.166	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4134	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4135	416	NAK PID=5A	
3982	12.666	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4136	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4137	416	NAK PID=5A	
3983	12.166	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4138	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4139	333	NAK PID=5A	
3984	13.250	IN NAK	Frame#=1E8 ADDR=01 ENDP=2
4140	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4141	333	NAK PID=5A	
3985	12.000	IN DATA0 ACK	Frame#=1E8 ADDR=01 ENDP=2
4142	0	IN PID=69	ADDR=01 ENDP=2 CRC5=18
4143	416	DATA0 PID=C3	DATA=13byte CRC16=CDD7
4144	500	ACK PID=D2	

CBW（コマンドトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 4124
000  55 53 42 43 F8 7A 25 82 00 00 00 00 00 00 06 1E  USBC.z%.........
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 4143
000  55 53 42 53 F8 7A 25 82 00 00 00 00 00           USBS.z%.....
```

←PREVENT ALLOW MEDIUM REMOVALコマンド

←PREVENT ALLOW MEDIUM REMOVALコマンド実行結果

図 6.7　PREVENT ALLOW MEDIUM REMOVAL コマンド

以下に示す図 6.8 は TEST UNIT READY コマンド（通常時）を測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
912	7.119.289.166	SOF	Frame#=5EC
1504	0	SOF PID=A5 FRAME#=5EC CRC5=16	
913	5.750	OUT DATA1 ACK	Frame#=5EC ADDR=01 ENDP=1
1505	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B	
1506	250	DATA1 PID=4B DATA=31byte CRC16=1FB5	
1507	583	ACK PID=D2	
914	994.250	SOF	Frame#=5ED
1508	0	SOF PID=A5 FRAME#=5ED CRC5=09	
915	999.916	SOF	Frame#=5EE
1509	0	SOF PID=A5 FRAME#=5EE CRC5=01	
916	5.666	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1510	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1511	416	NAK PID=5A	
917	11.250	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1512	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1513	416	NAK PID=5A	
918	10.750	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1514	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1515	416	NAK PID=5A	
919	10.916	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1516	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1517	416	NAK PID=5A	
920	11.416	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1518	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1519	416	NAK PID=5A	
921	10.750	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1520	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1521	416	NAK PID=5A	
922	10.500	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1522	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1523	416	NAK PID=5A	
923	11.000	IN NAK	Frame#=5EE ADDR=01 ENDP=2
1524	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1525	500	NAK PID=5A	
924	12.000	IN DATA1 ACK	Frame#=5EE ADDR=01 ENDP=2
1526	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
1527	500	DATA1 PID=4B DATA=13byte CRC16=9C9B	
1528	583	ACK PID=D2	

CBW（コマンドトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 1506
000  55 53 42 43 08 10 0B 82 00 00 00 00 00 00 06 00  USBC............
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 1527
000  55 53 42 53 08 10 0B 82 00 00 00 00 00           USBS.........
```

←TEST UNIT READYコマンド

←TEST UNIT READYコマンド実行結果

図 6.8　TEST UNIT READY コマンド（通常時）

以下に示す**図 6.9** は TEST UNIT READY コマンド（メディア取り出し状態時）を測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
4445	4.307.767.416	SOF	Frame#=3AB
4615	0	SOF PID=A5 FRAME#=3AB CRC5=0C	
4446	7.000	OUT DATA1 ACK	Frame#=3AB ADDR=01 ENDP=1
4616	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B	
4617	250	DATA1 PID=4B DATA=31byte CRC16=58FA	
4618	500	ACK PID=D2	
4447	993.000	SOF	Frame#=3AC
4619	0	SOF PID=A5 FRAME#=3AC CRC5=03	
4448	1.000.000	SOF	Frame#=3AD
4620	0	SOF PID=A5 FRAME#=3AD CRC5=1C	
4449	11.250	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4621	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4622	333	NAK PID=5A	
4450	12.833	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4623	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4624	333	NAK PID=5A	
4451	11.416	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4625	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4626	416	NAK PID=5A	
4452	12.666	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4627	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4628	416	NAK PID=5A	
4453	12.666	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4629	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4630	416	NAK PID=5A	
4454	12.750	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4631	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4632	333	NAK PID=5A	
4455	12.666	IN NAK	Frame#=3AD ADDR=01 ENDP=2
4633	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4634	333	NAK PID=5A	
4456	12.416	IN DATA0 ACK	Frame#=3AD ADDR=01 ENDP=2
4635	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
4636	416	DATA0 PID=C3 DATA=13byte CRC16=9422	
4637	500	ACK PID=D2	

CBW（コマンドトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 4617
000  55 53 42 43 D0 22 0B 82 00 00 00 00 00 00 06 00  USBC."..........
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 4636
000  55 53 42 53 D0 22 0B 82 00 00 00 00 01           USBS."........
```

←TEST UNIT READYコマンド

←TEST UNIT READYコマンド実行結果(コマンドフェイル)

図 6.9　TEST UNIT READY コマンド（メディア取り出し状態時）

以下に示す図 6.10 は VERIFY(10)コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
9859	9.374.013.333	SOF	Frame#=381
10599	0	SOF PID=A5 FRAME#=381 CRC5=04	
9860	13.250	OUT DATA0 ACK	Frame#=381 ADDR=01 ENDP=1
10600	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B	
10601	250	DATA0 PID=C3 DATA=31byte CRC16=E931	
10602	416	ACK PID=D2	
9861	986.666	SOF	Frame#=382
10603	0	SOF PID=A5 FRAME#=382 CRC5=0C	
9862	1.000.000	SOF	Frame#=383
10604	0	SOF PID=A5 FRAME#=383 CRC5=13	
9863	12.250	IN NAK	Frame#=383 ADDR=01 ENDP=2
10605	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10606	416	NAK PID=5A	
9864	12.750	IN NAK	Frame#=383 ADDR=01 ENDP=2
10607	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10608	416	NAK PID=5A	
9865	12.666	IN NAK	Frame#=383 ADDR=01 ENDP=2
10609	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10610	416	NAK PID=5A	
9866	12.666	IN NAK	Frame#=383 ADDR=01 ENDP=2
10611	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10612	416	NAK PID=5A	
9867	11.833	IN NAK	Frame#=383 ADDR=01 ENDP=2
10613	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10614	416	NAK PID=5A	
9868	12.500	IN NAK	Frame#=383 ADDR=01 ENDP=2
10615	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10616	416	NAK PID=5A	
9869	13.833	IN NAK	Frame#=383 ADDR=01 ENDP=2
10617	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10618	333	NAK PID=5A	
9870	12.583	IN DATA1 ACK	Frame#=383 ADDR=01 ENDP=2
10619	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
10620	416	DATA1 PID=4B DATA=13byte CRC16=2BC9	
10621	500	ACK PID=D2	

CBW（コマンドトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 10601
000  55 53 42 43 40 2D AD 81 00 00 00 00 00 00 0A 2F  USBC@-........./
010  00 00 00 00 20 00 00 19 00 00 00 00 00 00 00 00  .... ..........
Packet No. = 10620
000  55 53 42 53 40 2D AD 81 00 00 00 00 00           USBS@-.......
```

←VERIFY(10)コマンド

←VERIFY(10)コマンド実行結果

図 6.10　VERIFY(10)コマンド

以下に示す図 6.11 は STOP/START UNIT コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
3817	3.694.791.166	SOF	Frame#=2CD
3968	0	SOF PID=A5 FRAME#=2CD CRC5=03	
3818	4.666	OUT DATA0 ACK	Frame#=2CD ADDR=01 ENDP=1
3969	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B	
3970	250	DATA0 PID=C3 DATA=31byte CRC16=DDF0	
3971	416	ACK PID=D2	
3819	995.250	SOF	Frame#=2CE
3972	0	SOF PID=A5 FRAME#=2CE CRC5=0B	
3820	1.000.000	SOF	Frame#=2CF
3973	0	SOF PID=A5 FRAME#=2CF CRC5=14	
3821	4.583	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3974	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3975	416	NAK PID=5A	
3822	11.250	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3976	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3977	416	NAK PID=5A	
3823	11.833	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3978	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3979	416	NAK PID=5A	
3824	12.000	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3980	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3981	416	NAK PID=5A	
3825	12.000	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3982	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3983	416	NAK PID=5A	
3826	11.583	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3984	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3985	416	NAK PID=5A	
3827	11.666	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3986	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3987	333	NAK PID=5A	
3828	11.166	IN NAK	Frame#=2CF ADDR=01 ENDP=2
3988	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3989	333	NAK PID=5A	
3829	11.416	IN DATA0 ACK	Frame#=2CF ADDR=01 ENDP=2
3990	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
3991	416	DATA0 PID=C3 DATA=13byte CRC16=6966	
3992	500	ACK PID=D2	

CBW（コマンドトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 3970
000  55 53 42 43 00 BE 1B 82 00 00 00 00 00 00 06 1B  USBC............
010  00 00 00 02 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 3991
000  55 53 42 53 00 BE 1B 82 00 00 00 00 00           USBS.........
```

←STOP/START UNITコマンド

←STOP/START UNITコマンド実行結果

図 6.11　STOP/START UNIT コマンド

以下に示す図 6.12 は MODE SENSE(6)コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information
586	7.050.291.166	SOF	Frame#=5A7
869	0	SOF PID=A5 FRAME#=5A7 CRC5=1B	
587	5.750	OUT DATA0 ACK	Frame#=5A7 ADDR=01 ENDP=1
870	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B	
871	250	DATA0 PID=C3 DATA=31byte CRC16=12EF	
872	416	ACK PID=D2	
588	994.166	SOF	Frame#=5A8
873	0	SOF PID=A5 FRAME#=5A8 CRC5=1A	
589	1.000.000	SOF	Frame#=5A9
874	0	SOF PID=A5 FRAME#=5A9 CRC5=05	
590	5.500	IN DATA0 ACK	Frame#=5A9 ADDR=01 ENDP=2
875	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
876	500	DATA0 PID=C3 DATA=64byte CRC16=AEDD	
877	416	ACK PID=D2	
591	58.250	IN DATA1 ACK	Frame#=5A9 ADDR=01 ENDP=2
878	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
879	416	DATA1 PID=4B DATA=64byte CRC16=D0BF	
880	583	ACK PID=D2	
618	346.750	IN DATA0 ACK	Frame#=5A9 ADDR=01 ENDP=2
933	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
934	416	DATA0 PID=C3 DATA=64byte CRC16=D0BF	
935	500	ACK PID=D2	
619	589.500	SOF	Frame#=5AA
936	0	SOF PID=A5 FRAME#=5AA CRC5=0D	
620	999.916	SOF	Frame#=5AB
937	0	SOF PID=A5 FRAME#=5AB CRC5=12	
621	9.750	IN DATA1 ACK	Frame#=5AB ADDR=01 ENDP=2
938	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18	
939	500	DATA1 PID=4B DATA=13byte CRC16=5B0D	
940	500	ACK PID=D2	

CBW（コマンドトランスポート）

DATA（データトランスポート）

CSW（ステータストランスポート）

データパケット

```
Packet No. = 871
000  55 53 42 43 30 24 AD 81 C0 00 00 00 80 00 06 1A  USBC0$..........
010  00 1C 00 C0 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 876
000  17 00 00 08 00 00 00 00 00 00 02 00 9C 0A 08 00  ................
010  00 00 02 00 00 00 02 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 879
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 934
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 939
000  55 53 42 53 30 24 AD 81 A8 00 00 00 00           USBS0$.......
```

←MODE SENSE(6)コマンド

←MODE SENSE(6)情報

←MODE SENSE(6)コマンド実行結果

図 6.12　MODE SENSE(6)コマンド

以下に示す図 6.13 は READ FORMAT CAPACITIES コマンドを測定したものです。

No.	Time (nSec)	Transaction / Packet	Information	
441	7.030.291.666	SOF	Frame#=593	
580	0	SOF PID=A5 FRAME#=593 CRC5=11		
442	10.083	OUT DATA1 ACK	Frame#=593 ADDR=01 ENDP=1	↑ CBW（コマンドトランスポート）
581	0	OUT PID=E1 ADDR=01 ENDP=1 CRC5=0B		
582	250	DATA1 PID=4B DATA=31byte CRC16=D94C		
583	666	ACK PID=D2		↓
443	989.916	SOF	Frame#=594	
584	0	SOF PID=A5 FRAME#=594 CRC5=1E		
444	1.000.000	SOF	Frame#=595	
585	0	SOF PID=A5 FRAME#=595 CRC5=01		
445	10.333	IN DATA0 ACK	Frame#=595 ADDR=01 ENDP=2	↑ DATA（データトランスポート）
586	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
587	416	DATA0 PID=C3 DATA=64byte CRC16=A690		
588	500	ACK PID=D2		
446	58.583	IN DATA1 ACK	Frame#=595 ADDR=01 ENDP=2	
589	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
590	416	DATA1 PID=4B DATA=64byte CRC16=D0BF		
591	583	ACK PID=D2		
470	347.416	IN DATA0 ACK	Frame#=595 ADDR=01 ENDP=2	
638	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
639	416	DATA0 PID=C3 DATA=64byte CRC16=D0BF		
640	583	ACK PID=D2		
495	355.333	IN DATA1 ACK	Frame#=595 ADDR=01 ENDP=2	
689	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
690	500	DATA1 PID=4B DATA=60byte CRC16=2400		
691	500	ACK PID=D2		↓
496	228.250	SOF	Frame#=596	
692	0	SOF PID=A5 FRAME#=596 CRC5=09		
497	1.000.000	SOF	Frame#=597	
693	0	SOF PID=A5 FRAME#=597 CRC5=16		
498	10.166	IN DATA0 ACK	Frame#=597 ADDR=01 ENDP=2	↑ CSW（ステータストランスポート）
694	0	IN PID=69 ADDR=01 ENDP=2 CRC5=18		
695	500	DATA0 PID=C3 DATA=13byte CRC16=8D9D		
696	500	ACK PID=D2		↓

データパケット

```
Packet No. = 582
000  55 53 42 43 90 23 14 82 FC 00 00 00 80 00 0A 23  USBC.#.........#   ←READ FORMAT CAPACITIESコマンド
010  00 00 00 00 00 00 00 FC 00 00 00 00 00 00 00 00  ................
Packet No. = 587
000  00 00 00 08 00 00 01 E0 02 00 02 00 00 00 01 E0  ................   ←READ FORMAT CAPACITIES情報
010  00 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 590
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 639
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
Packet No. = 690
000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
030  00 00 00 00 00 00 00 00 00 00 00 00              ............
Packet No. = 695
000  55 53 42 53 90 23 14 82 E8 00 00 00 00           USBS.#.......   ←READ FORMAT CAPACITIESコマンド実行結果
```

図 6.13　READ FORMAT CAPACITIES コマンド

参考ドキュメント

ハードウェアマニュアル
M16C/6C ハードウェアマニュアル
（最新版をルネサス テクノロジホームページから入手してください。）

C コンパイラマニュアル M16C シリーズ, R8C ファミリ用 C コンパイラパッケージ V.5.44
C コンパイラユーザーズマニュアル Rev.1.00
(最新版をルネサス テクノロジホームページから入手してください。)

ホームページとサポート窓口

ルネサス テクノロジホームページ
http://japan.renesas.com/

お問合せ先
http://japan.renesas.com/inquiry
csc@renesas.com

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2009.12.29	-	初版発行

本資料ご利用に際しての留意事項

１．本資料は、お客様に用途に応じた適切な弊社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弊社または第三者の知的財産権その他の権利の実施、使用を許諾または保証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など全ての情報の使用に起因する損害、第三者の知的財産権その他の権利に対する侵害に関し、弊社は責任を負いません。
３．本資料に記載の製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的、あるいはその他軍事用途の目的で使用しないでください。また、輸出に際しては、「外国為替および外国貿易法」その他輸出関連法令を遵守し、それらの定めるところにより必要な手続を行ってください。
４．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などの全ての情報は本資料発行時点のものであり、弊社は本資料に記載した製品または仕様等を予告なしに変更することがあります。弊社の半導体製品のご購入およびご使用に当たりましては、事前に弊社営業窓口で最新の情報をご確認いただきますとともに、弊社ホームページ(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。
５．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、万一本資料の記述の誤りに起因する損害がお客様に生じた場合においても、弊社はその責任を負いません。
６．本資料に記載の製品データ、図、表などに示す技術的な内容、プログラム、アルゴリズムその他応用回路例などの情報を流用する場合は、流用する情報を単独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。弊社は、適用可否に対する責任は負いません。
７．本資料に記載された製品は、各種安全装置や運輸・交通用、医療用、燃焼制御用、航空宇宙用、原子力、海底中継用の機器・システムなど、その故障や誤動作が直接人命を脅かしあるいは人体に危害を及ぼすおそれのあるような機器・システムや特に高度な品質・信頼性が要求される機器・システムでの使用を意図して設計、製造されたものではありません（弊社が自動車用と指定する製品を自動車に使用する場合を除きます）。これらの用途に利用されることをご検討の際には、必ず事前に弊社営業窓口へご照会ください。なお、上記用途に使用されたことにより発生した損害等について弊社はその責任を負いかねますのでご了承願います。
８．第７項にかかわらず、本資料に記載された製品は、下記の用途には使用しないでください。これらの用途に使用されたことにより発生した損害等につきましては、弊社は一切の責任を負いません。
　１）生命維持装置。
　２）人体に埋め込み使用するもの。
　３）治療行為（患部切り出し、薬剤投与等）を行うもの。
　４）その他、直接人命に影響を与えるもの。
９．本資料に記載された製品のご使用につき、特に最大定格、動作電源電圧範囲、放熱特性、実装条件およ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ください。弊社保証値を越えて製品をご使用された場合の故障および事故につきましては、弊社はその責任を負いません。
１０．弊社は製品の品質および信頼性の向上に努めておりますが、特に半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。弊社製品の故障または誤動作が生じた場合も人身事故、火災事故、社会的損害などを生じさせないよう、お客様の責任において冗長設計、延焼対策設計、誤動作防止設計などの安全設計（含むハードウェアおよびソフトウェア）およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特にマイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
１１．本資料に記載の製品は、これを搭載した製品から剥がれた場合、幼児が口に入れて誤飲する等の事故の危険性があります。お客様の製品への実装後に容易に本製品が剥がれることがなきよう、お客様の責任において十分な安全設計をお願いします。お客様の製品から剥がれた場合の事故につきましては、弊社はその責任を負いません。
１２．本資料の全部または一部を弊社の文書による事前の承諾なしに転載または複製することを固くお断りいたします。
１３．本資料に関する詳細についてのお問い合わせ、その他お気付きの点等がございましたら弊社営業窓口までご照会ください。

D039444