SH7286 グループ

USB ファンクションモジュール　USB マスストレージクラス

R01AN0063JJ0100
Rev. 1.00
2010.08.20

要旨

本アプリケーションノートは、SH7286 の USB ファンクションモジュールの使用方法、および USB マスストレージクラスに対応したファームウェアの作成例について説明します。

本アプリケーションノートの内容とソフトウェアは USB ファンクションモジュールの応用例を説明するもので、その内容を保証するものではありません。

動作確認デバイス

SH7286

目次

1. はじめに

1.1 仕様

SH7286 の USB ファンクションモジュールの使用方法、および USB マスストレージクラスに対応したファームウェアの作成例を掲載しています。

1.2 使用機能

- 割り込みコントローラ (INTC)
- ピンファンクションコントローラ (PFC)
- USB ファンクションモジュール

1.3 適用条件

- マイコン: SH7286
- 動作周波数: 内部クロック　100 MHz
  バスクロック　50 MHz
  周辺クロック　50 MHz
- 統合開発環境: ルネサス エレクトロニクス製
  High-performance Embedded Workshop Ver.4.04.01
- C コンパイラ: ルネサス エレクトロニクス製
  SuperH RISC engine ファミリ C/C++ コンパイラパッケージ Ver.9.01 Release 01
- コンパイルオプション: High-performance Embedded Workshop でのデフォルト設定
  (-cpu=sh2a -object="$(CONFIGDIR)¥$(FILELEAF).obj"
  -debug -gbr=auto -chgincpath -errorpath -global_volatile=0
  -opt_range=all -infinite_loop=0 -del_vacant_loop=0
  -struct_alloc=1 -nologo)

1.4 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。合わせて参照してください。

- SH7285 グループ　初期設定例
- SH7285/SH7286　USB ファンクションモジュール　USB・シリアル変換アプリケーションノート

1.5 "L"アクティブ端子（信号）の表記について

端子名（信号名）末尾の＃は“L”アクティブ端子（信号）であることを示します。

2. 概要

本プログラムでは USB ファンクションモジュール (USB) を使用したコントロール転送、バルク転送、およびマスストレージクラスコマンド対応処理を行います。

SH7286 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。

- USB プロトコルの自動処理
- 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ェアで処理する必要があります。）
- 転送スピード：フルスピード
- 割り込み要求：USB 送受信に必要な各種割り込み信号を生成
- クロック：外部入力 (48MHz) 、内部入力（EXTAL 12MHz 使用時のみ）
- 低消費電力モードを搭載
- バストランシーバを内蔵
- エンドポイント構成：表 1に示す構成

表 1　エンドポイント構成

エンドポイント名	名称	転送方式	最大パケットサイズ	FIFO バッファ容量	DMA/DTC 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	コントロールイン	8 バイト	8 バイト	—
	EP0o	コントロールアウト	8 バイト	8 バイト	—
エンドポイント 1	EP1	バルクアウト	64 バイト	64×2 (128) バイト	可能
エンドポイント 2	EP2	バルクイン	64 バイト	64×2 (128) バイト	可能
エンドポイント 3	EP3	インタラプト	8 バイト	8 バイト	—

システム構成例を図 1に示します。

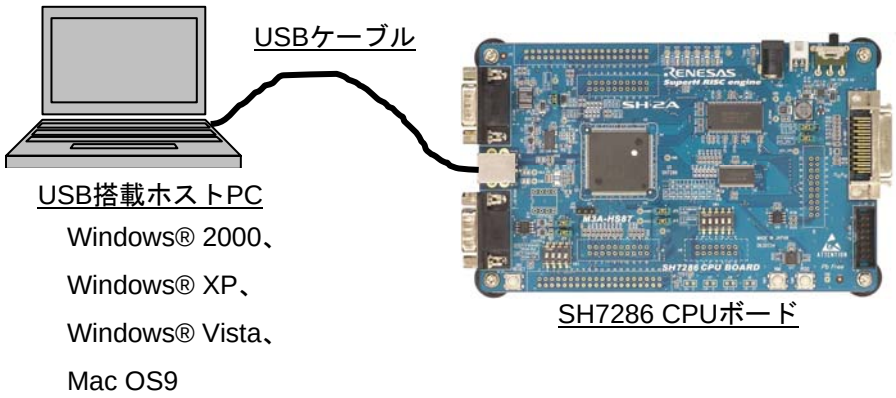

図 1　システム構成例

本システムは、SH7286 を搭載したルネサス エレクトロニクス製の SH7286 CPU ボード、Windows® 2000、Windows® XP、 Windows® Vista、 または Mac OS9 を OS として搭載する PC によって構成されています。

本システムは、ホスト PC と SH7286 CPU ボードを USB で接続し、SH7286 CPU ボード上の SDRAM を RAM Disk として動作させることにより、ホスト PC から SH7286 CPU ボードの SDRAM へのデータ書き込み、および SDRAM からの読み出しができます。

上記 OS に標準で付属している USB マスストレージクラス (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、SH7286 の USB モジュールを評価可能
2. サンプルプログラムは USB のコントロール転送とバルク転送をサポート
3. E10A-USB（USB 接続型エミュレータ）によるデバッグが可能
4. プログラムを追加作成することでインタラプト転送*にも対応可能

【注】 * 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。なお、SH7286 はアイソクロナス転送には対応していません。

3.　USB マスストレージクラス (Bulk-Only Transport) の概要

この章では、USB マスストレージクラス (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場合の参考にしてください。なお、規格の詳細は、「616. 参考ドキュメント」の(3)と(4)を参照してください。

3.1　USB マスストレージクラスについて

USB マスストレージクラスとは、大規模記憶装置をホスト 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ことを伝えるためには、Interface DescriptorのbInterfaceClassフィールドに値H'08 を記述することが必要です。また、USBマスストレージクラスではString Descriptorを用いてSerial Numberをホストへ伝える必要があります。本サンプルプログラムではUnicodeで000000000001 を返信しています。

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。

USB マスストレージクラスでは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、さら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を行うことができます。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。

3.2　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。7 種類のコマンドフォーマットがあり、表 2に示すサブクラスコードが定められています。

表 2　サブクラスコード

サブクラスコード	コマンドの規格
H'01	Reduced Block Commands (RBC)、 T10/1240-D
H'02	Attachment Packet Interface (ATAPI) for CD-ROMs. SFF-8020i、 Multi-Media Command Set 2 (MMC-2)
H'03	Attachment Packet Interface (ATAPI) for Tape. QIC-157
H'04	USB Mass Storage Class UFI Command Specification
H'05	Attachment Packet Interface (ATAPI) for Floppies. SFF-8070i
H'06	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H'06 のSCSI Primary Commandsを使用します。

3.3　Bulk-Only Transport

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 つに分けることができます。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定義しています。

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。

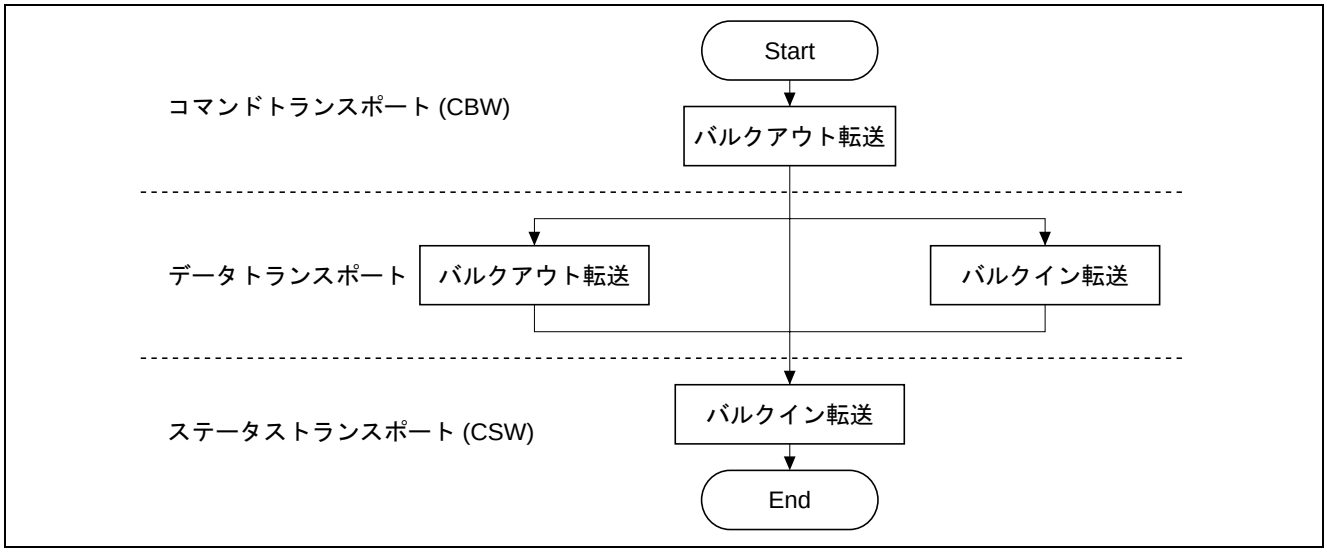

図 2　転送方法とトランスポートの関係

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 H'50 を記述することが必要です。

3.3.1　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 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ます。

内容は表 3に示すフォーマットで送られます。

表 3　コマンドトランスポートフォーマット

	7	6	5	4	3	2	1	0
H'00–H'03	dCBWSignature							
H'04–H'07	dCBWTag							
H'08–H'0B	dCBWDataTransferLength							
H'0C	bmCBWFlags							
H'0D	予約(0)				bCBWLUN			
H'0E	予約(0)			bCBWCBLength				
H'0F–H'1E	CBWCB							

各フィールドの内容は以下の通りです。

- dCBWSignature:
  データパケットが CBW であると識別するためのフィールド。値は H'43425355（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 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) を表します。
- 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です。このフィールドにホスト 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。

3.3.2　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 4に示すフォーマットで送られます。

表 4　ステータストランスポートフォーマット

	7	6	5	4	3	2	1	0
H'0–H'3	dCSWSignature							
H'4–H'7	dCSWTag							
H'8–H'B	dCSWDataResidue							
H'C	bCSWStatus							

各フィールドの内容は以下の通りです。

- dCSWSignature:
  データパケットが CSW であると識別するためのフィールド。値は H'53425355（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 H'00 にセットします。0 以外の値はコマンド実行時の不具合を示します。H'01 はコマンド失敗、H'02 はフェーズエラーを示します。

3.3.3　データトランスポート

データトランスポートとは、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。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。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。

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 2 種類のコマンドをサポートする必要があります。表 5にクラスコマンドを示します。

表 5　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 (H'FF)	Bulk-Only Mass Storage Reset	インタフェースをリセットする
254 (H'FE)	Get Max LUN	サポートする LUN の数を判別する

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 で使用するすべ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。本サンプルの場合、論理ユニットは 1 つなので返答値を 0 としてホストに返答します。

3.5　サブクラスコード (SCSI transparent command set)

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 6に示す 11 個の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「コマンド失敗」と報告しています。

表 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
H'00	TEST UNIT READY	メディアが使用可能か否かを確認する
H'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
H'12	INQUIRY	ドライブに関する情報をホストに伝える
H'1A	MODE SENSE (6)	ドライブの状態をホストに伝える
H'1B	START/STOP UNIT	メディアの着脱を制御する
H'1E	PREVENT ALLOW MEDIUM REMOVAL	メディアの着脱を禁止／許可する
H'23	READ FORMAT CAPACITIES	メディアのフォーマット情報をホストに伝える
H'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
H'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
H'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
H'2F	VERIFY (10)	メディア上のデータにアクセス可能かどうかを確認する

4.　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）を使用します。

- SH7286 CPU ボード（型名 M3A-HS87）ルネサス エレクトロニクス製
- E10A-USB Emulator ルネサス エレクトロニクス製
- E10A-USB 用 PC (Windows® 2000、 Windows® XP)
- USB ホスト PC (Windows® 2000、 Windows® XP、 Windows® Vista、Mac OS9)
- USB ケーブル
- High-performance Embedded Workshop 4 ルネサス エレクトロニクス製

4.1　ハードウェア環境

図 3に各デバイスの接続形態を示します。

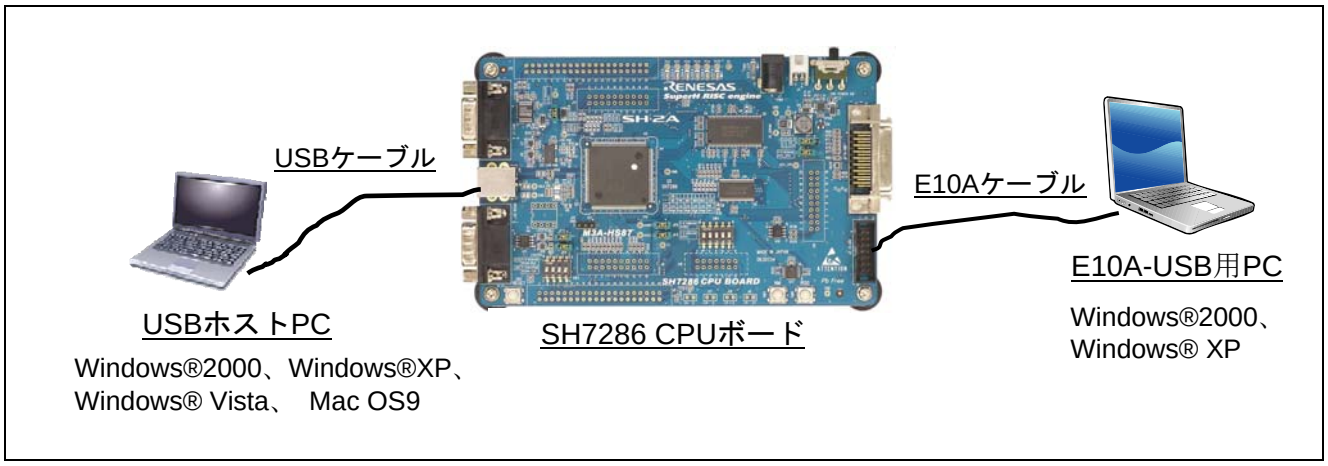

図 3　デバイスの接続形態

(1) SH7286 CPU ボード

本システムではE10A-USBを使用します。SH7286CPUボードでE10A-USBを使用するために、ディップスイッチ（SW4）を表 7のように設定してください。なお、SW4 の切り換えは、必ず電源をオフにした状態で行ってください。

表 7　ディップスイッチの設定

設定値（モード 6）	ディップスイッチの機能
SW4-1 (FWE)　OFF	内蔵フラッシュメモリの書き込み/消去プロテクト
SW4-2 (MD1)　OFF	MD1 端子状態
SW4-3 (MD0)　ON	MD0 端子状態

(2) USB ホスト PC

USB ポート搭載の Windows® 2000、Windows® XP、Windows® Vista または Mac OS9 をインストールしたパソコンを USB ホスト PC 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 マスストレージクラス (Bulk-Only Transport) のデバイスドライバを使用するため、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) E10A-USB 用 PC

USB ポート搭載の Windows® 2000、Windows® XP をインストールしたパソコンを E10A-USB 用 PC として使用します。E10A-USB 用 PC の USB コネクタに E10A-USB エミュレータを接続し、接続用のケーブルを介して E10A-USB と CPU ボードを接続してください。接続後、High-performance Embedded Workshop 4 を起動してエミュレーションを行います。

4.2　ソフトウェア環境

ソースコードのコンパイル、リンク、およびデバッグは High-performance Embedded Workshop 4 で行ってください。High-performance Embedded Workshop 4 は、sh7286_usb_msc.hws をダブルクリックすることで起動します。

4.2.1　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 sh7286_usb_msc フォルダ内に収められています。High-performance Embedded Workshop 4 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することで、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図 4に示します。

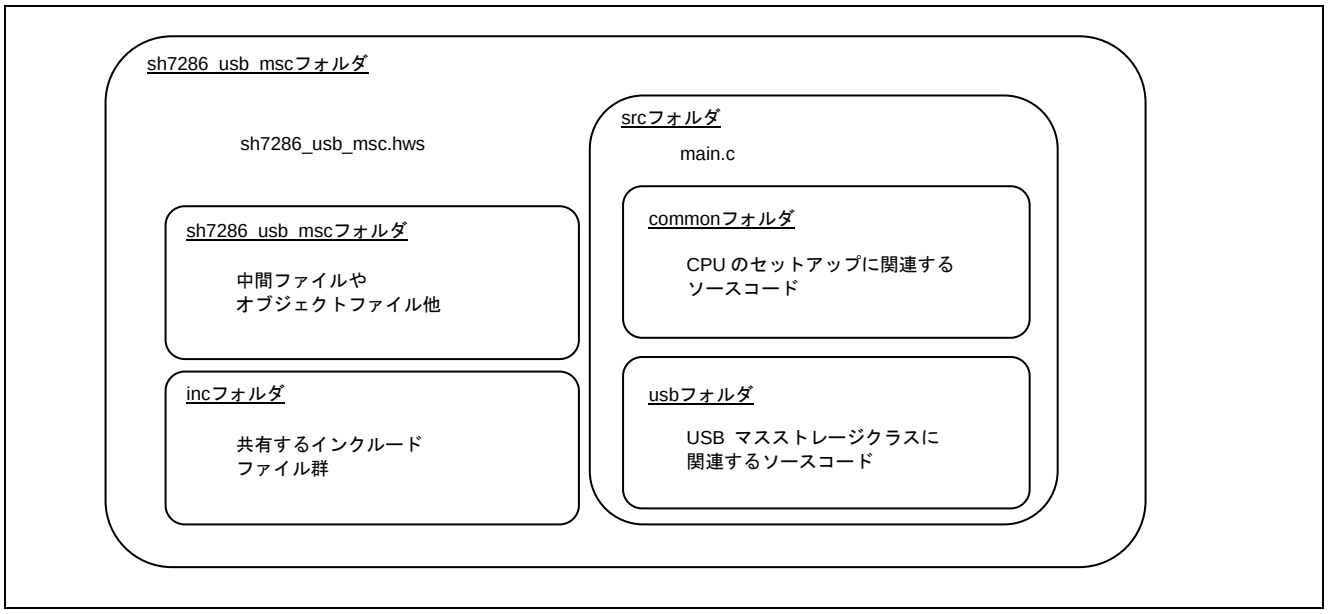

図 4　フォルダに含まれるファイル

4.2.2　コンパイルおよびリンク

ソースコードのコンパイルには High-performance Embedded Workshop 4 を使用してください。

4.3　プログラムのロードと実行方法

プログラムのロードと実行方法について説明します。

4.3.1　プログラムのロード

SH7286 CPU ボード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。

(1) High-performance Embedded Workshop 4 をインストールした E10A-USB 用 PC に E10A-USB エミュレータを接続してください。
(2) ユーザケーブルで E10A-USB エミュレータと SH7286 CPU ボードを接続してください。
(3) SH7286 CPU ボードの電源を投入してください。
(4) sh7286_usb_msc フォルダ内の sh7286_usb_msc.hws を実行してください。
(5) 「デバッグ > 接続」を選択してください。エミュレータのモードの選択を要求されるので、「SH7286」、「E10A-USB Emulator」を選択してください。
(6) SH7286 CPU ボードのリセットスイッチを押し、「OK」ボタンを押してください。
(7)動作周波数の入力を求められるので、実装している水晶発振子 (10.00MHz) の周波数を入力してください。
(8) ID Code の入力を求められるので、「E10A」を入力してください。
(9) 「デバッグ＞ダウンロード > All Download Modules」を選択することでプログラムのダウンロードが行われます。

4.3.2　プログラムの実行

「デバッグ > リセット後実行」を選択することでプログラムが実行されます。

4.4　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。

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 SH7286 CPU ボードに挿入し、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、ホスト PC は SH7286 CPU ボード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ィンドウが開かれるので、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 であることを確認し、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画面が出力されるので、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ィ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ィンドウに戻るので、閉じるボタンをクリックしてウィンドウを閉じてください。

以上で SH7286 CPU ボードを USB 接続の RAM Disk として使用できます。

4.5　RAM Disk の設定変更について

本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。

4.5.1　リムーバブル・固定ディスク

本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。

SetSystemSwitch.h 内の「#define　REMOVABLE_DISK」をコメント化し、コメント化されている「#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。

4.5.2　RAM Disk の容量変更

本サンプルプログラムでは、16Mバイト分のSDRAMをRAM Diskとして使用しています。RAM Diskの容量を変更する場合はSysMemMap.hの内容を変更します。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. 1.5M バイト以上の値を指定してください。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。本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。

5.　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 SH7286 CPU ボード上で動作し、また、SH7286 CPU ボードが 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。

5.1　状態遷移図

図 5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 5のように 3 つの状態に遷移します。

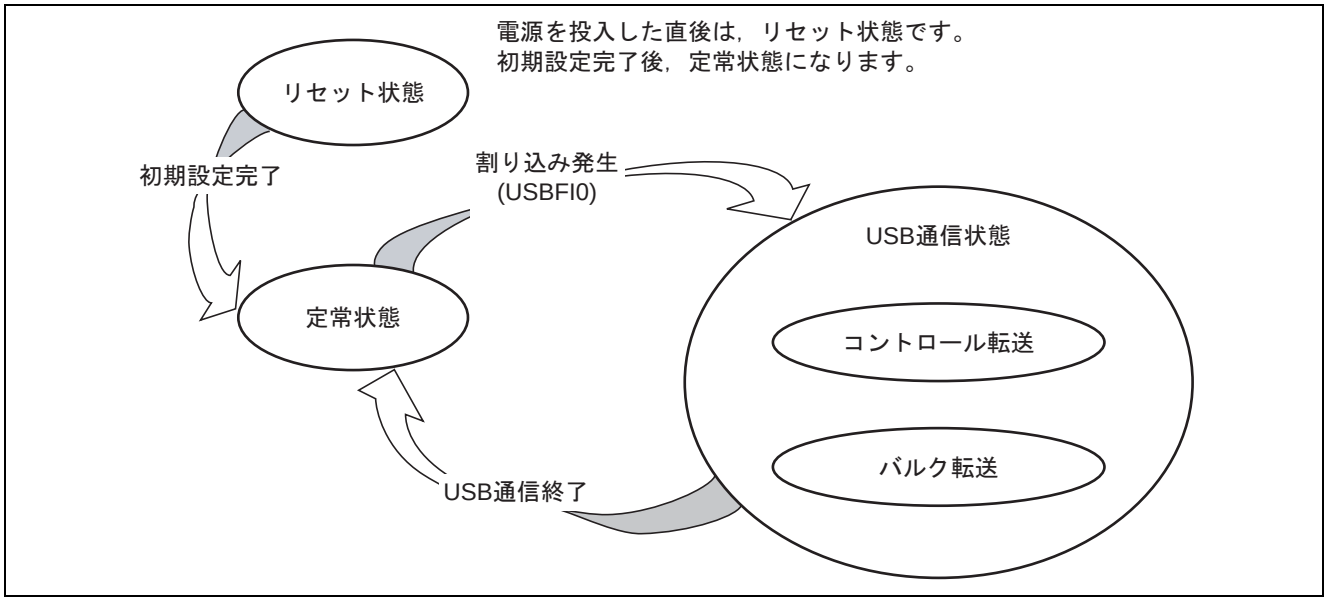

図 5　状態遷移図

- 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 SH7286 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遷移します。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、 1 (USBIFR0、 1) によって示されます。割り込み要因が発生すると、USBIFR0、 1 の対応するビットに 1 がセットされます。

5.2　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 6参照）。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。

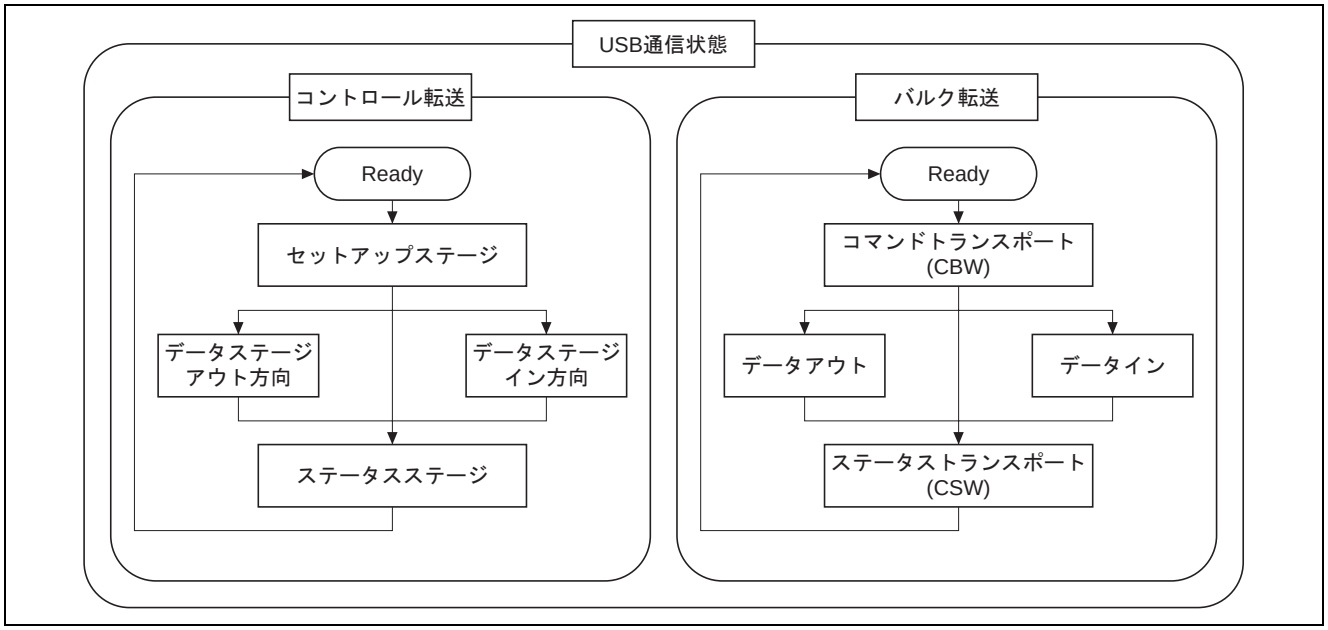

図 6　USB 通信状態

5.2.1　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します。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す。

コントロール転送の一連の転送処理は、2 つまたは 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。

5.2.2　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 マスストレージクラス (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 マスストレージクラス (Bulk-Only Transport) の一連の転送処理（リードやライトなど）は、2 つまたは 3 つのステージから構成されます。USB マスストレージクラス (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。

5.3　ファイル構成

本サンプルプログラムは、15 個のソースファイルと 14 個のヘッダファイルで構成されています。全構成ファイルを表 8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。

表 8　ファイル構成

ファイル名	主な機能
main.c	USB ファンクションの初期設定
usb フォルダ	
UsbMain.c	割り込み要因の判定、パケットの送受信
DoRequest.c	ホスト PC が発行するセットアップコマンドの処理
DoRequestBOT_StorageClass.c	USB マスストレージクラス (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	USB マスストレージクラス (Bulk-Only Transport) のトランスポートを実行
DoSCSICommand.c	SCSI コマンドの解析および処理
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、FAT 情報構成のためのマクロ定義
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 ファームウェアで使用する変数の初期設定
SysMemMap.h	メモリマップのアドレス定義
common フォルダ	
bscsdram.c	SDRAM インタフェースの設定
cpg.c	CPG の設定
dbsct.c	初期化セクション（B、R）の定義
hwsetup.c	ハードウェア初期化処理
intprg.c	割り込みプログラム
resetprg.c	CPU 初期化プログラム
stacksct.h	スタック領域の定義
vect.h	割り込みプログラムの宣言
vecttbl.c	割り込み例外処理ベクタテーブルの定義
inc フォルダ	
iodefine.h	SH7286 のレジスタ定義
typedefine.h	共通型定義

5.4　関数の機能

各ファイルに含まれる関数とその機能を表 9～表 16に示します。

- main.c
  パワーオンリセットの際には、resetprg.c の CPU 初期化プログラムが実行された後、main.c の main 関数が呼び出されます。ここでは SH7286 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 9　main.c

格納ファイル	関数名	機能
main.c	main	モジュールおよびメモリの初期化、USB バスのプルアップ制御を行い、メインループへ移行

- UsbMain.c
  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 10　UsbMain.c

格納ファイル	関数名	機能
UsbMain.c	BranchOfInt	バスリセットおよびエンドポイント 0 の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、USB マスストレージクラスでは使用しません）
	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 RAM に書き込む（リングバッファ非対応、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、USB マスストレージクラスでは使用しません）
	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（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 のクリア等を行う
	ActBusVcc	USB ケーブル接続割り込み動作を行う（本サンプルアプリケーションでは使用しません）
	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

- DoRequest.c
  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 H'045B（ベンダ：ルネサス）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」でお客様のベンダ ID を取得してください。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してください。

表 11　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に対応する
	DecVenderCommands	ベンダコマンドに対応する

- DoRequestBOT_StorageClass.c
  USB マスストレージクラス (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 で使用しているすべ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。本サンプルの場合、論理ユニットは 1 つなので返答値として 0 をホストに返答します。

表 12　DoRequestBOT_StorageClass.c

格納ファイル	関数名	機能
DoRequestBOT_StorageClass.c	DecBOTClassCommands	USB マスストレージクラス (Bulk-Only Transport) コマンドに対応する

- DoControl.c
  コントロール転送の割り込み (SETUP TS) が入ると、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 (EP0o TS、 EP0i TR、 EP0i TS) 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 を呼び出しデータステージとステータスステージを行います。

表 13　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する
	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を制御する
	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を制御する
	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

- DoBulk.c
  バルク転送に関する処理を行います。USB マスストレージクラス (Bulk-Only Transport) では ActBulkInReady を使用しません。

表 14　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

- DoBOTMSClass.c
  DoBOTMSClass.c では、USB マスストレージクラス (Bulk-Only Transport) の 2 つまたは 3 つのステージ制御と仕様に従った動作を行います。

表 15　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を行う
	ActBulkOnlyCommand	Bulk-Only Transport の CBW の制御を行う
	ActBulkOnlyIn	Bulk-Only Transport 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（データステージがイン方向の場合）
	ActBulkOnlyOut	Bulk-Only Transport 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（データステージがアウト方向の場合）

- DoSCSICommand.c
  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。

表 16　DoSCSICommand.c

格納ファイル	関数名	機能
DoSCSICommand.c	DecBotCmd	ホストから Bulk-Only Transport で送られる SCSI コマンドに対応する
	SetBotCmdErr	SCSI コマンドのエラー時の処理

5.5　RAM Disk について

本サンプルプログラムではSH7286 CPUボード上のSDRAMをDisk装置に見立て、ホストPCに対しSH7286 CPU ボード（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 7に示すようにマスタブートブロックと、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SDRAM上のRAM Disk領域にマスタブートブロックと、パーティションブートブロックを書き込みます。

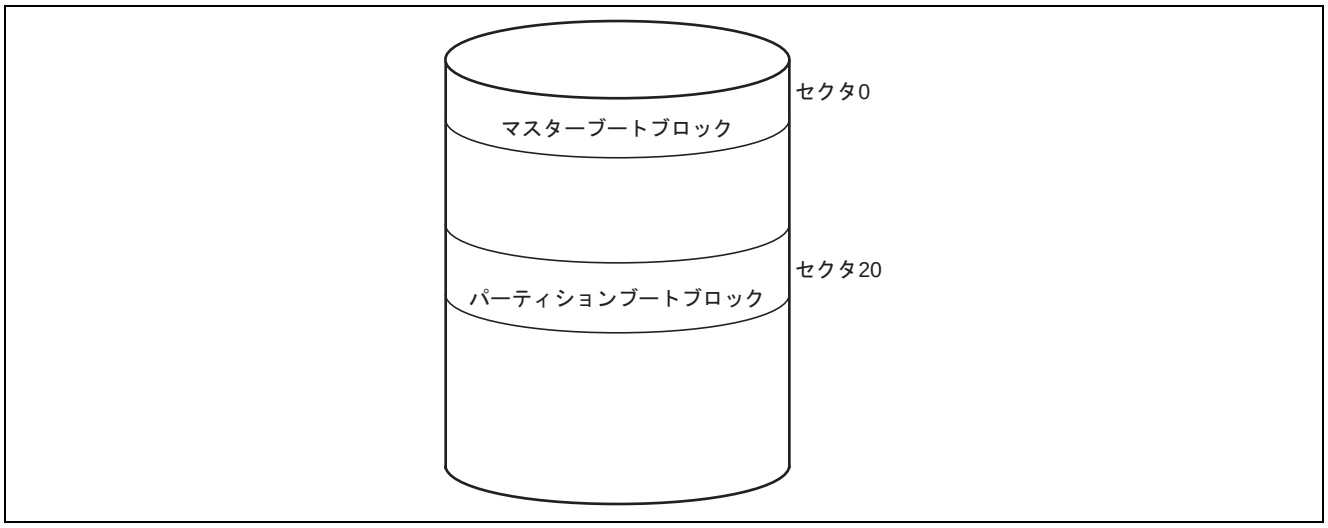

図 7　Disk の構造

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 7の構造を理解する必要があります。

5.6　サポートする SCSI コマンドの動作について

表 17～表 19に本サンプルプログラムがサポートするSCSIコマンドの動作を示します。

表 17　SCSI コマンド動作表 (1/3)

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 コマンドである事を認識後、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コマンドである事を認識後、SDRAM上に展開してあるDisk装置の 1 セクタ当りのバイト数とディスクの総セクタ数からなる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、"5.7　エラー時の処理についてのCase (4)"に従います。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	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。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 (H'00)を返信します。
	CSW	ホスト PC に対しコマンド実行結果を送信します。メディアがアクセス不能状態の場合、コマンド失敗（CSW ステータス H'01）を返信します。
READ (10)	CBW	コマンドをデコードし、READ (10)コマンドである事を認識後、SDRAM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、"5.7　エラー時の処理についてのCase (4)"に従います。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	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。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 (H'00)を返信します。
	CSW	ホスト PC に対し READ (10)コマンド実行結果を送信します。メディアがアクセス不能状態の場合、コマンド失敗（CSW ステータス H'01）を返信します。
WRITE (10)	CBW	コマンドをデコードし、WRITE (10)コマンドである事を認識後、SDRAM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"5.7　エラー時の処理についてのCase (9)"に従います。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	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。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	ホスト PC 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、コマンド失敗（CSW ステータス H'01）を返信します。

表 18　SCSI コマンド動作表 (2/3)

コマンド名	トランスポート名	動作内容
REQUEST SENSE	CBW	コマンドをデコードし、REQUEST SENSE コマンドである事を認識後、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 コマンドである事を認識後、ホスト PC に対し正常終了の送信準備を行います。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「失敗」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	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。メディアがアクセス不能状態の場合、コマンド失敗（CSW ステータス H'01）を返信します。
TEST UNIT READY	CBW	コマンドをデコードし、TEST UNIT READY コマンドである事を認識後、ホスト PC に対し正常終了の送信準備を行います。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「失敗」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	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。メディアがアクセス不能状態の場合、コマンド失敗（CSW ステータス H'01）を返信します。
VERIFY (10)	CBW	コマンドをデコードし、VERIFY (10)コマンドである事を認識後、ホスト PC に対し正常終了の送信準備を行います。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「失敗」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	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。メディアがアクセス不能状態の場合、コマンド失敗（CSW ステータス H'01）を返信します。
START/STOP 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時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	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 (6)	CBW	コマンドをデコードし、MODE SENSE (6)コマンドである事を認識後、要求された MODE SENSE 情報の送信準備を行います。
	データ	ホスト PC に対し MODE SENSE 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。

表 19　SCSI コマンド動作表 (3/3)

コマンド名	トランスポート名	動作内容
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、SDRAM上に展開してあるDisk装置がフォーマット済みであるものとして 1 セクタ当りのバイト数やディスクの総セクタ数などからなる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、"5.7　エラー時の処理についてのCase (4)"に従います。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	ホスト PC に対し READ FORMAT CAPACITY 情報をバルクイン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 (H'00)を返信します。
	CSW	ホスト PC に対しコマンド実行結果を送信します。メディアがアクセス不能状態の場合、コマンド失敗（CSW ステータス H'01）を返信します。
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 (H'00) 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無い場合、何も動作は行いません。
	CSW	ホスト PC に対しコマンド失敗（CSW ステータス H'01）を返信します。

5.7　エラー時の処理について

USB マスストレージクラス (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 の規格では次にあ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)。

上記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 20と表 21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 20　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 21　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 22に発生する可能性のあるエラー状況例を示します。

表 22　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 23のようになります。

表 23　ファンクションの対応動作

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 H'02 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 H'00 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 H'02 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 H'01 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 H'02 を設定する。

データ転送時のエラー処理フローは、図 8～図 10のようになります。

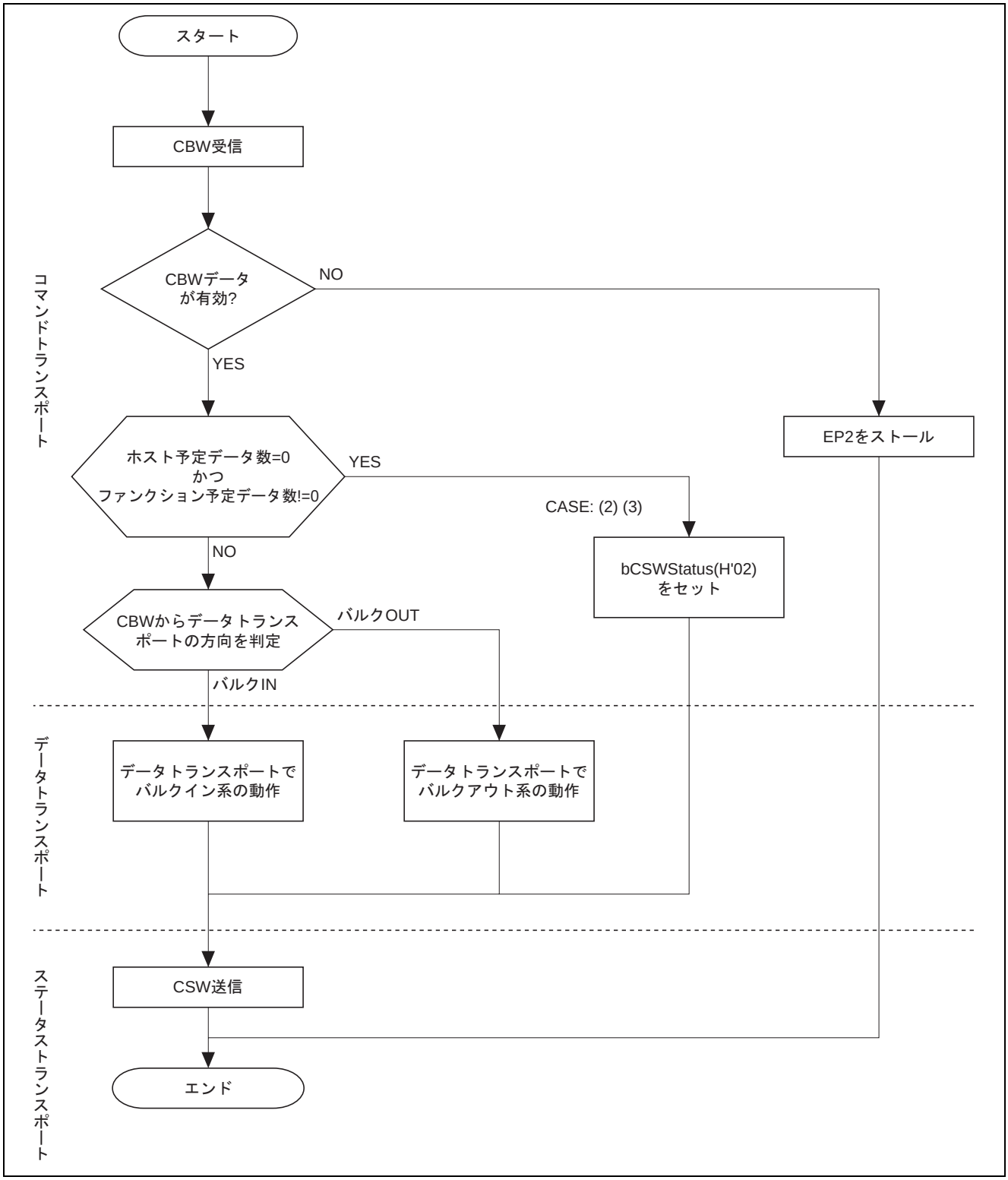

図 8　データ転送時のエラー処理フロー (1)

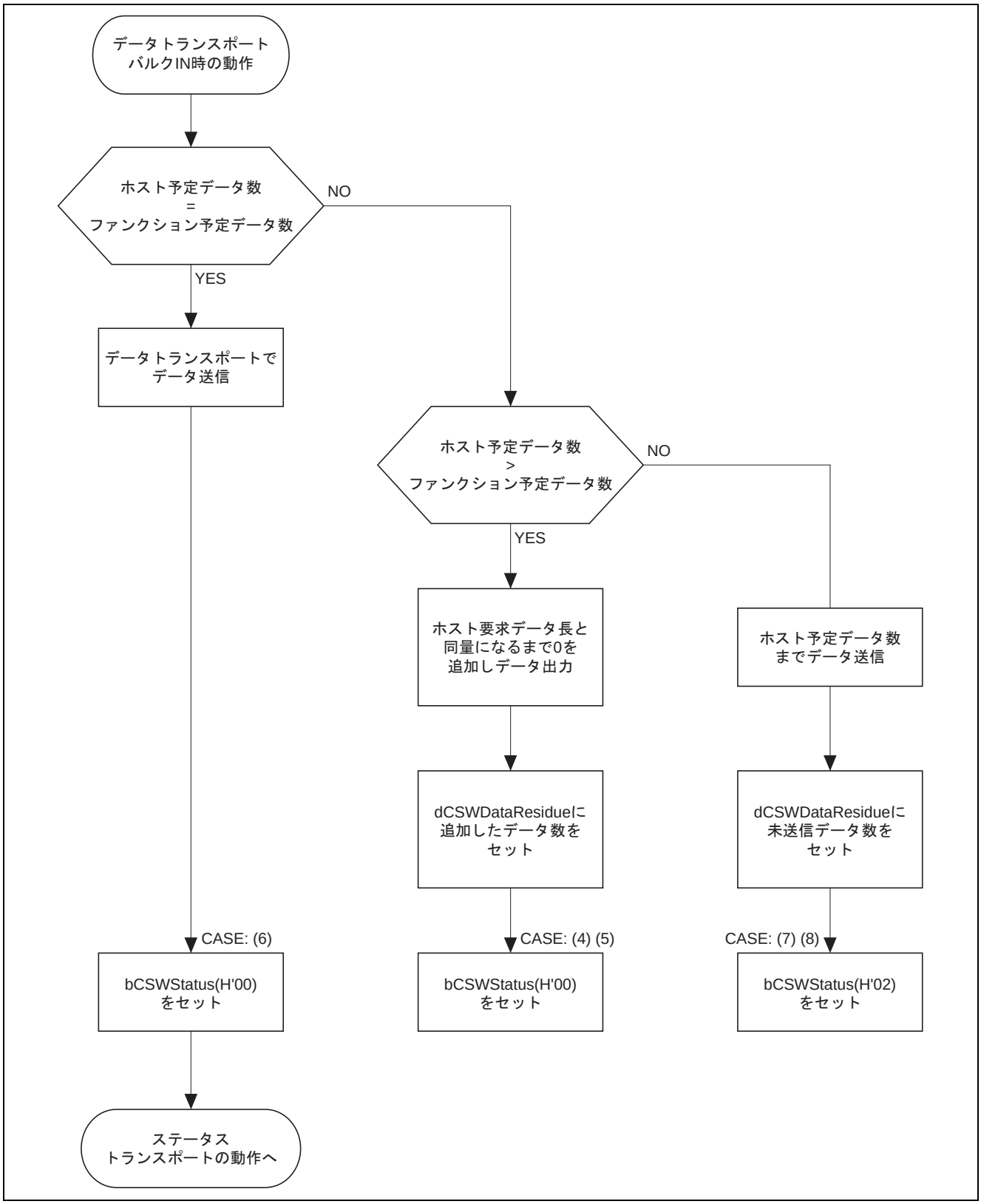

図 9　データ転送時のエラー処理フロー (2)

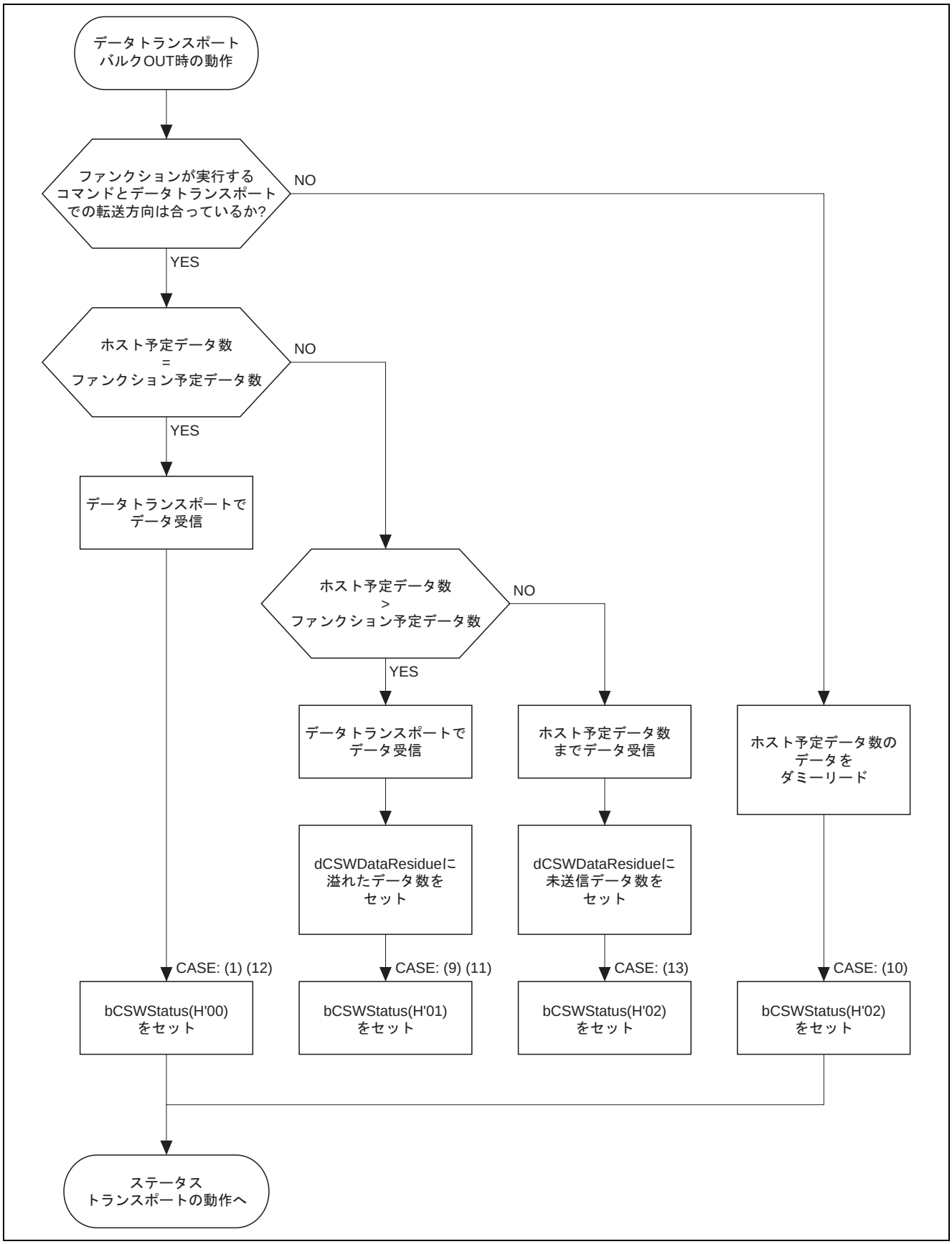

図 10　データ転送時のエラー処理フロー (3)

USB マスストレージクラス (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 dCBWDataTransferLength 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 dCBWDataTransferLength 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、dCBWDataTransferLength 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ときは、データトランスポートでデータ転送する度に dCBWDataTransferLength 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。

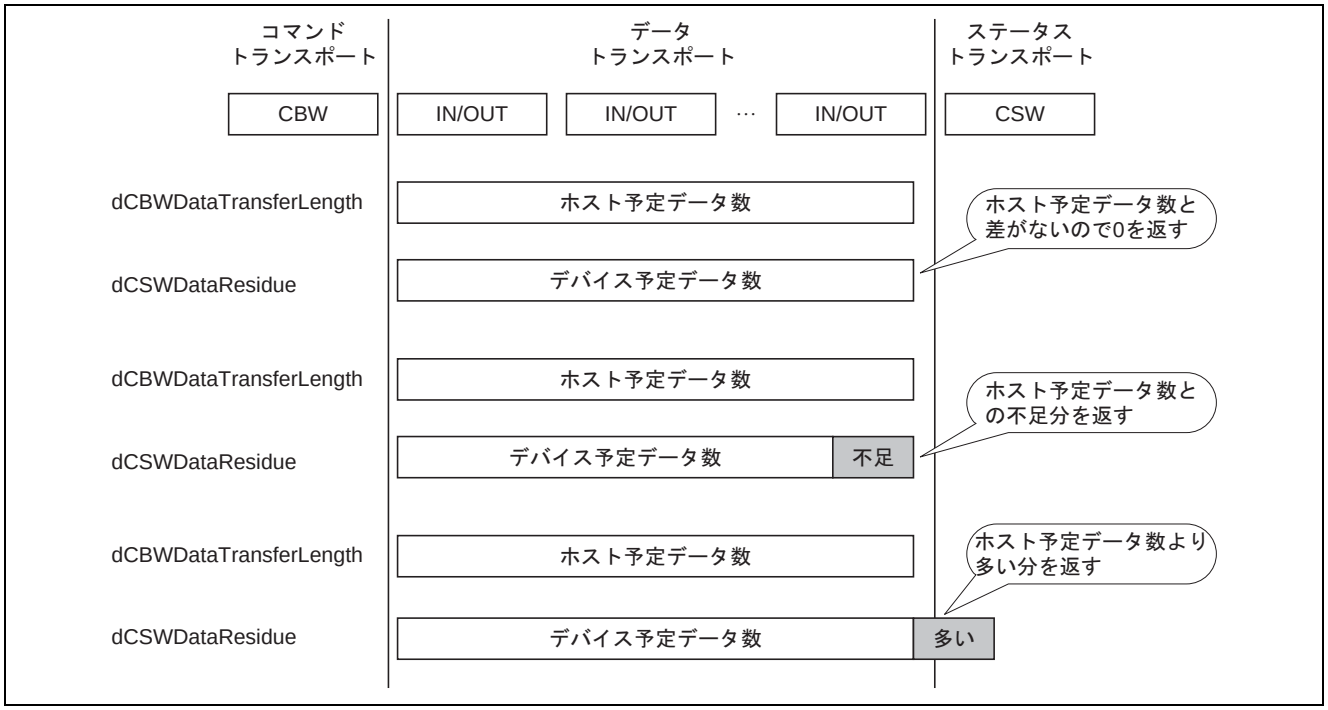

図 11　Bulk-Only Transport における各ステージ

6. 参考ドキュメント

- ソフトウェアマニュアル
  (1) SH-2A、SH2A-FPU ソフトウェアマニュアル Rev. 3.00
  （最新版をルネサスエレクトロニクスホームページから入手してください。）

- ハードウェアマニュアル
  (2) SH7280 グループハードウェアマニュアル Rev. 2.00
  （最新版をルネサスエレクトロニクスホームページから入手してください。）

- USB 規格関連
  (3) Universal Serial Bus Specification, Revision 2.0
  (4) Universal Serial Bus Mass Storage Class Specification Overview
  (5) Universal Serial Bus Mass Storage Class (Bulk-Only Transport)
  — USB 開発者向けホームページ
  http://www.usb.org/developers

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com/

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

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2010.08.20	—	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理

   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。

   CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

2. 電源投入時の処置

   【注意】電源投入時は、製品の状態は不定です。

   電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。

   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。

   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. リザーブアドレスのアクセス禁止

   【注意】リザーブアドレスのアクセスを禁止します。

   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

4. クロックについて

   【注意】リセット時は、クロックが安定した後、リセットを解除してください。

   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。

   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

5. 製品間の相違について

   【注意】型名の異なる製品に変更する場合は、事前に問題ないことをご確認下さい。

   同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。

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。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/inquiry

Colophon 1.0