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 Human Interface Devices（HID）Class 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 HID クラスについて説明し、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。具体的には HID クラスの通信を例に、M16C/6C 内蔵 USB ファンクションモジュールの構成を解説しています。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 Device Class Definition for Human Interface Devices(HID)
- 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、バルクに関するファームウェアは準備しておりません。バルク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
- バストランシーバを内蔵
- 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

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。

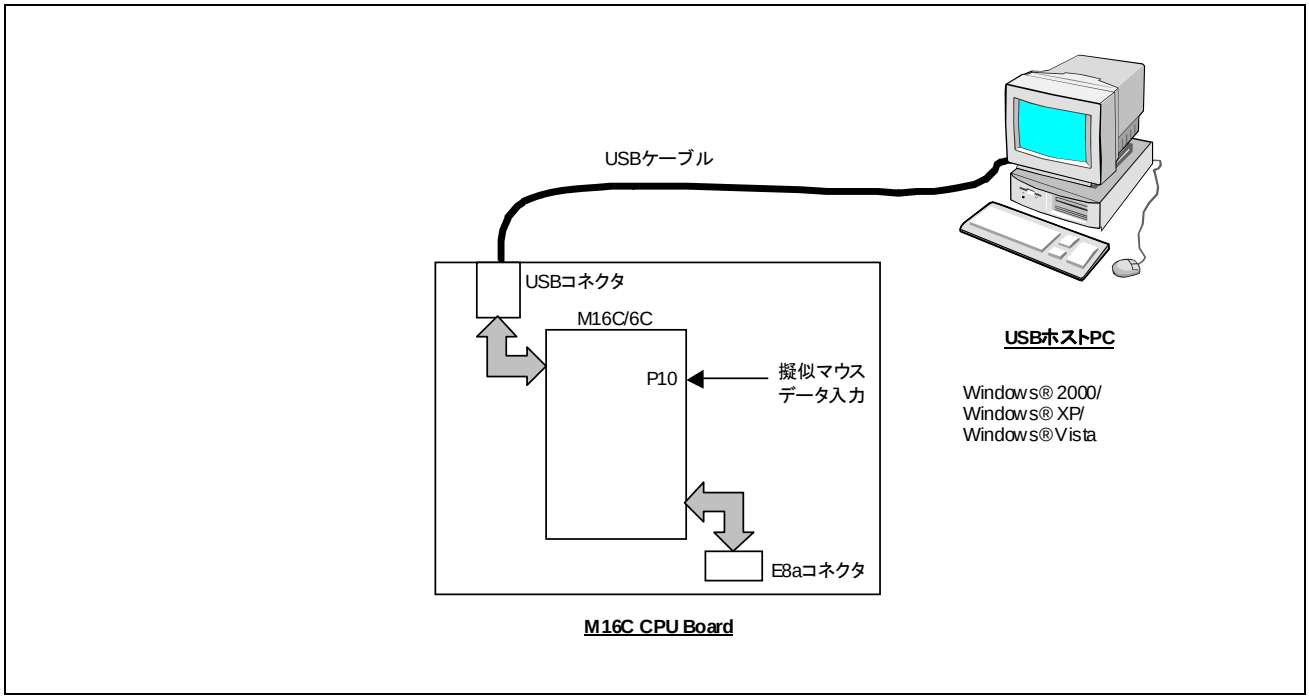


図 1.1　システム構成

本システムは、M16C/6C を搭載した M16C CPU Board、Windows® 2000/ Windows® XP/ Windows® Vista を OS とした PC によって構成されています。

本システムは、M16C CPU Board 上で擬似マウスデータを自動生成し、USB を通じて USB ホストへマウスデータ（以下 HID データ）を出力する HID クラスファームウェアです。

なお、Windows® 2000/ Windows® XP/ Windows® Vista に標準で搭載されている USB HID クラス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、インタラプト転送をサポート
- 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

2. Human Interface Devices（HID）クラスの概要

この章では、Human Interface Devices（HID）クラスについて説明します。

USB HID クラスのシステムを開発時に参照してください。

規格の詳細につきましては、以下の関連マニュアルを参照してください。

「Device Class Definition for Human Interface Devices（HID）Version 1.11」

「HID Usage Tables Version 1.11」

2.1 HID クラス

USB HID クラスとは、人が PC 操作に必要な機器を使いこなせるよう規格化されたクラスのことです。代表的なものとしてはマウス、キーボード、ジョイスティックなどがあります。

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 0x03 を記述します。

2.2 サブクラスコード

サブクラスは、もともと HID クラスのファンクションが使用する特定のプロトコルを識別するように考えられていましたが、人が使用する機器は種類が多く、サブクラスのプロトコル定義は現実とかけ離れています。よって、HID クラスは、ほとんどのプロトコル定義にはサブクラスを使用しません。その代わりに HID クラスファンクションでは、レポートディスクリプタでプロトコルを判別します。

一方、BIOS がサポートするファンクション（ブートデバイス）では、使用するプロトコルの判別に単純な方法が適しています。そこで、HID クラスファンクションがマウスあるいはキーボード（すなわち、ブートデバイスとして使用可能）の場合、前もって定められたプロトコル（ブートプロトコル）をサポートするファンクションの判別にサブクラスを使用します。

サブクラスコードは、ファンクションがブートプロトコルに対応しているかを示します。

USB ホストに、ファンクションがブートプロトコルに対応していると伝えるには、インタフェースディスクリプタの bInterfaceSubClass フィールドに 0 x 01 を記述します。

2.3 プロトコルコード

プロトコルコードは、ファンクションがブートプロトコルに対応している（サブクラスコードが 0 以外）場合、対象ファンクションを示すために使用します。対象となるのはキーボード（値：0x01）とマウス（値：0x02）です。これにより、ホストはそれぞれのファンクションに適合したプロトコルが使用可能だとわかります。

USB ホストに、ファンクションが何であるかを伝えるには、インタフェースディスクリプタの bInterfaceProtocol フィールドに値を記述します。

2.4 HID クラスのディスクリプタについて

HID クラスのファンクションには、他の USB ファンクションがもっているディスクリプタ情報に加え「HID ディスクリプタ」、「レポートディスクリプタ」と「フィジカルディスクリプタ」（オプション）が必要になります。**図 2.1** に HID クラスファンクションのディスクリプタ構成を示します。

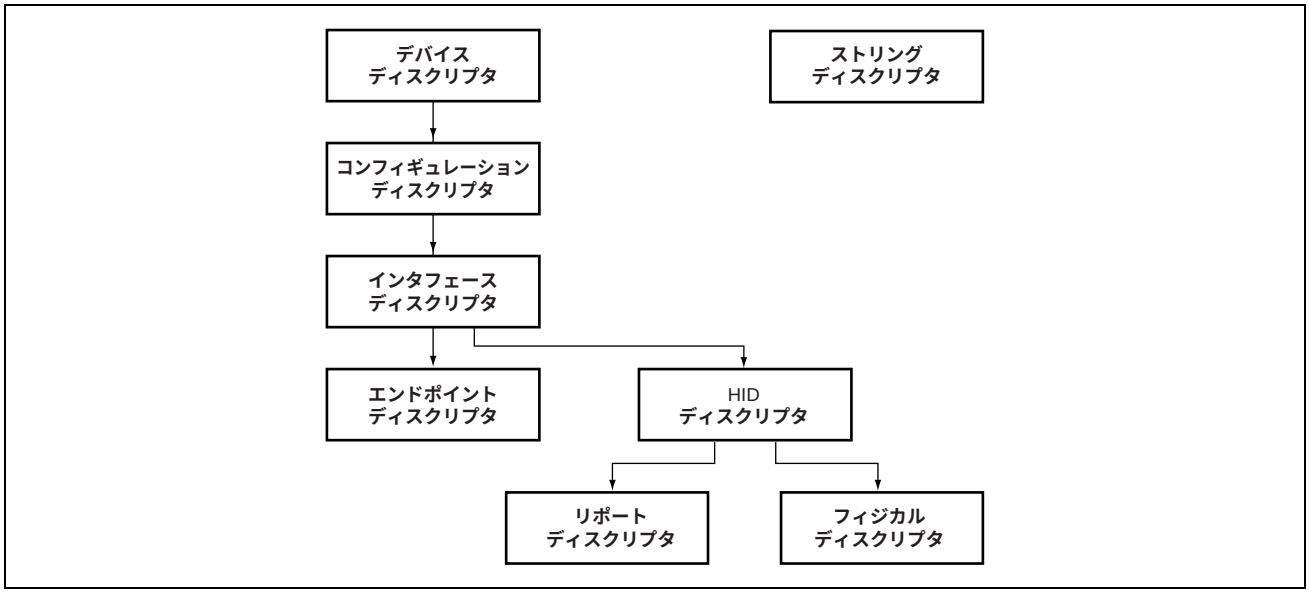


図 2.1　ディスクリプタ構成

2.5 HIDディスクリプタ

HID ディスクリプタは、レポートディスクリプタとフィジカルディスクリプタ（オプション）をまとめる働きをします。**表 2.1** に HID ディスクリプタのフォーマットを示します。

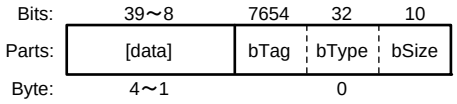
表 2.1　HID ディスクリプタ

フィールド	サイズ（バイト）	内　　容
bLength	1	ディスクリプタのサイズ（0x09 で固定）
bDescriptorType	1	ディスクリプタのタイプ（0x21 で固定）
bcdHID	2	BCD 表現の HID バージョン
bCountryCode	1	地域固有デバイスのための国識別番号（不要なら 0）
bNumDescriptors	1	クラスディスクリプタの数
bDescriptorType	1	クラスディスクリプタの型 （HIDREPORT の場合 0x22）
wDescriptorLength	2	レポートディスクリプタのサイズ

2.6 レポートディスクリプタ

レポートディスクリプタは、USB ホストとファンクション間で転送するデータのフォーマットを決める働きをします。レポートディスクリプタには、他のディスクリプタのように規定されたフォーマットはありません。ファンクションからの報告の種類と数により、レポートディスクリプタの長さと内容が変化します。

レポートディスクリプタは、ファンクションに関する情報を提供するまとまり（アイテム）によって構成されます。アイテムには、ショートアイテムとロングアイテムの 2 種類があります。ここではショートアイテムを用いて説明します。



bTypeとbSizeの値が示す意味を次に示します。

アイテムタイプ	
bType値	タイプ
00	Main
01	Global
10	Local
11	Reserve

アイテムサイズ	
bSize値	データサイズ
00	0byte
01	1byte
10	2byte
11	4byte

図 2.2 レポートディスクリプタアイテム

アイテムはデータ、アイテムタグ、アイテムタイプとアイテムサイズの 4 種類から構成されます。アイテムはフィールドを使って、情報がどのような種類であるかを示します。

アイテムタイプにはメインアイテム、グローバルアイテム、ローカルアイテムの 3 種類があり、メインアイテムタイプ（レポートディスクリプタ内のデータフィールドの定義もしくはグループ化に使用）は 5 種類、グローバルアイテムタイプ（データの記述に使用）は 12 種類、ローカルアイテムタイプ（特性の定義づけに使用）は 10 種類のアイテムタグがあります。

これらのアイテムを組み合わせることにより、USB ホストとファンクション間で転送するデータのフォーマットを作成します。

2.6.1 メインアイテム

Main item type の item tag は 5 種類あります。表 2.2 に内容を示します。

表 2.2 Main item type の item tag

item tag	bTag	bType	bSize	意　　味
Input	1000	00	nn	1 つ以上の物理的なコントロールが提供するデータに関する情報について記述
Output	1001	00	nn	出力データフィールドを定義するために使用
Feature	1011	00	nn	デバイスに送ることができるデバイスコンフィギュレーション情報について記述
Collection	1010	00	nn	2 つ以上のデータ（Input、Output あるいは Feature）のグループ化を開始
End Collection	1100	00	nn	Collection に対応し、2 つ以上のデータ（Input、Output あるいは Feature）のグループ化を終了

(1) Input item tag

Input item tag のパラメータ（データフィールド）には 8 つの項目があり、それぞれ 1 ビットの値で指定します。表 2.3 に内容を示します。

表 2.3 Input item tag のパラメータ

ビット	値	シンボル	機　　能
0	0	Data	アイテムはデータを報告
	1	Constant	アイテムは定数を報告
1	0	Array	アイテムは配列を報告
	1	Variable	アイテムは変数を報告
2	0	Absolute	アイテムは絶対値を報告
	1	Relative	アイテムは最後の報告からの偏差を報告
3	0	No Wrap	アイテムが報告する値はロールオーバーしない
	1	Wrap	アイテムが報告する値はロールオーバーする（たとえば : 0～10 の値を出力するダイヤルで、ダイヤルを回し続けると値 10 の次は 0 を出力する）
4	0	Linear	アイテムはコントロールする物の状態をリニアに報告する
	1	Non Linear	アイテムは生データを処理し、コントロールする物の状態をリニアに報告しない
5	0	Preferred State	アイテムはユーザがコントロールしていない時、戻る状態がある
	1	No Preferred	アイテムはユーザがコントロールしていない時、戻る状態がない
6	0	No Null position	アイテムは無意味なデータを送る状態がない
	1	Null state	アイテムは無意味なデータを送る状態がある
7	0	Reserved	リザーブ
8	0	Bit Field	アイテムはビットフィールドを発する
	1	Buffered Bytes	アイテムは 1 バイト固定サイズのストリームを発する
9-31	0	Reserved	リザーブ

(2) Output item tag & Feature item tag

Output および Feature item tag のパラメータ（データフィールド）には 9 つの項目があり、内容は Bit7 を除いて Input item tag と同じです。**表 2.4** に Output および Feature item tag のパラメータを示します。

表 2.4　Output および Feature item tag のパラメータ

ビット	値	シンボル	機　　能
0-6	—	—	Input item tag と同様
7	0	Non Volatile	アイテムの値はホスト要求によってのみ変更できる
	1	Volatile	アイテムの値はホスト要求の有無にかかわらず変更できる
8-31	—	—	Input item tag と同様

(3) Collection item tag

Collection item tag のパラメータ（データフィールド）は 8 種類あり、1 バイトの値で指定します。**表 2.5** に内容を示します。

表 2.5　Collection item tag のパラメータ

値	シンボル	機　　能
0x00	Physical	1 つに集められたデータアイテム。つまり、単一のポイントに正確なデータあるいは感知したデータを関連させる必要がある装置に使用。 たとえばデータがキーボードのような 1 つのデバイスから来るのではなく、多数のセンサーポジションを報告するデバイスなどで、いずれのデータもそれぞれの別のセンサーから来ることを示す。
0x01	Application	アプリケーションレベルでだけ使われる Usage を識別、このコレクションが HID デバイスあるいは複雑なデバイスの機能的な下位グループであることを示す。オペレーティング・システムでは、デバイスをコントロールするアプリケーションあるいはドライバにリンクするために、このコレクションと結び付けられた Usage を使用。
0x02	Logical	データアイテムが複合してデータ構造となるときに使用
0x03	Report	フィールドをすべて包む、論理的な収集を定義する。 レポート ID はこの収集に含まれる。アプリケーションが容易にデバイスのある機能をサポートするかどうかを決定することができる。
0x04	Named Array	データアイテムが複合しデータ構造となり、命名するときに使用
0x05	Usage Switch	これが含んでいる Usage の意味を変更する、論理的な収集 その収集中の Usage の目的を修正し、論理的な収集に適用された Usage を識別
0x06	Usage Modifier	包含するコレクションに付けられた Usage の意味を修正します。Usage は、基本的にはコントロールのために単一操作のモードを定義する。コントロールの操作方法を拡張可能にする。
0x07-7F	Reserved	リザーブ
0x80-FF	Vendor-defined.	ベンダ定義

2.6.2 グローバルアイテム

Global item type の item tag は 12 種類あります。表 2.6 に内容を示します。

表 2.6 Global item type の item tag

item tag	bTag	bType	bSize	意　味
Usage Page	0000	01	nn	現在の Usage Page を指定している値。アイテム使用法のインデックスを定義
Logical Minimum	0001	01	nn	変数または配列のアイテムが報告する最小値。たとえば、0～128 まで X 位置価値を報告するマウスは、0 の論理的な最小値をもつ。
Logical Maximum	0010	01	nn	変数または配列のアイテムが報告する最大値。たとえば、0～128 まで X 位置価値を報告するマウスは、128 の論理的な最大値をもつ。
Physical Minimum	0011	01	nn	可変アイテムの物理的な最小範囲値
Physical Maximum	0100	01	nn	可変アイテムの物理的な最大範囲値
Unit Exponent	0101	01	nn	10 の指数乗値
Unit	0110	01	nn	単位値
Report Size	0101	01	nn	レポートフィールドのサイズをビットで指定する符号なし値
Report ID	1000	01	nn	レポート ID を指定する符号なし値
Report Count	1001	01	nn	アイテムのデータフィールド数を指定する符号なし整数 何個のフィールドがこの特定のアイテムのためにレポートに含められるかを決定する（したがってビットが何個であるかがレポートに付け加えられる）
Push	1010	01	nn	Global アイテムステートテーブルをスタックに退避する
Pop	1011	01	nn	アイテムステートテーブルをスタックの先頭から復帰する

2.6.3 ローカルアイテム

Local item type の item tag は 10 種類あります。表 2.7 に内容を示します。

表 2.7 Local item type の item tag

item tag	bTag	bType	bSize	意　　味
Usage	0000	10	nn	アイテムまたはコレクションの用法（Usage）インデックスを定義
Usage Minimum	0001	10	nn	配列あるいはビットマップと関連づけた用法（Usage）の開始を定義
Usage Maximum	0010	10	nn	配列あるいはビットマップと関連づけた用法（Usage）の終了を定義
Designator Index	0011	10	nn	制御に使う身体部分を決定(Designator)
Designator Minimum	0100	10	nn	配列あるいはビットマップと関連づけた Designator の開始インデックスを定義
Designator Maximum	0101	10	nn	配列あるいはビットマップと関連づけた Designator の終了インデックスを定義
String Index	0111	10	nn	アイテムまたはコントロールに対してストリングを関連付ける
String Minimum	1000	10	nn	配列またはビットマップのコントロールに対して、ひとまとまりの連続するストリングを割り付けるときの始めのストリングのインデックス
String Maximum	1001	10	nn	配列またはビットマップのコントロールに対して、ひとまとまりの連続するストリングを割り付けるときの終わりのストリングのインデックス
Delimiter	1010	10	nn	ローカルアイテムの開始か終了を定義

2.6.4 レポートディスクリプタの例

図 2.3 にこのサンプルのレポートディスクリプタを示します。

```
Usage Page (Generic Desktop),                          : 05  01
Usage (Mouse),                                         : 09  02
Collection (Application),                              : A1  01
    Usage (Pointer),                                   : 09  01
    Collection (Physical),                             : A1  00
        Usage Page (Buttons),                          : 05  09
        Usage Minimum (01),                            : 19  01
        Usage Maximum (03),                            : 29  03
        Logical Minimum (0),                           : 15  00
        Logical Maximum (1),                           : 25  01
        Report Count (3),                              : 95  03
        Report Size (1),                               : 75  01
        Input (Data, Variable, Absolute), ; 3 button bits : 81  02
        Report Count (1),                              : 95  01
        Report Size (5),                               : 75  05
        Input (Constant), ; 5 bit padding              : 81  01
        Usage Page (Generic Desktop),                  : 05  01
        Usage (X),                                     : 09  30
        Usage (Y),                                     : 09  31
        Usage (Wheel),                                 : 09  38
        Logical Minimum (-127),                        : 15  81
        Logical Maximum (127),                         : 25  7F
        Report Size (8),                               : 75  08
        Report Count (3),                              : 95  03
        Input (Data, Variable, Relative), ; 2 position bytes (X & Y) : 81  06
    End Collection,                                    : C0
End Collection                                         : C0
```

図 2.3　レポートディスクリプタ

2.6.5 レポートディスクリプタの説明

表 2.8 にこのサンプルで使用するレポートディスクリプタの説明を示します。

表 2.8 レポートディスクリプタ

Item	Value（Hex）	Item 区分	意　味
Usage Page（Generic Desktop Control）	0x05 01	Global	用法ページ（Usage Page）を示す値。0x01 は"Generic Desktop Control"を表す
Usage（Mouse）	0x09 02	Local	アイテム使用法のインデックス。0x02 は"Mouse"を表す。オペレーティング・システムがデバイスをアクティブなアプリケーションあるいはドライバにマウスとしてリンクする。"Mouse"の使用法タイプは”Collection Application”。
Collection（Application）	0xA1 01	Main	アプリケーションにマウスとして” Pointer”を伝える。
Usage（Pointer）	0x09 01	Local	アイテム使用法のインデックス。0x01 は"Pointer"を表す。"Pointer"の使用法タイプは”Collection Physical”。
Collection（Physical）	0xA1 00	Main	ポインタとして多数のセンサーのポジション（ボタン、X 軸、Y 軸、ロータリー・コントロール）を 1 つに集める。
Usage Page（Button）	0x05 09	Global	用法ページ（Usage Page）を示す値。0x09 は"Button"を表す
Usage Minimum（1）	0x19 01	Local	アレイあるいはビットマップと関連づけた用法（Usage）を、1 からスタートと定義
Usage Maximum（3）	0x29 03	Local	アレイあるいはビットマップと関連づけた用法（Usage）を、3 で終わると定義
Logical Minimum（0）	0x15 00	Global	アイテムが報告する最小値は 0
Logical Maximum（1）	0x25 01	Global	アイテムが報告する最大値は 1
Report Count（3）	0x95 03	Global	アイテムにデータフィールドをいくつ使うかを示す。ここでは 3 個のレポートフィールドを使用
Report Size（1）	0x75 01	Global	レポートフィールドの大きさを示す。ここでは 1 ビット使用
Input（Data, Variable, Absolute）	0x81 02	Main	入力するアイテムがどのようなものかを示す。入力は可変データで、絶対値を報告。
Report Count （1）	0x95 01	Global	アイテムにデータフィールドをいくつ使うかを示す。ここでは 1 個のレポートフィールドを使用
Report Size （5）	0x75 05	Global	レポートフィールドの大きさを示す。ここでは 5 ビット使用
Input（Constant）	0x81 01	Main	入力するアイテムがどのようなものかを示す。入力は定数を報告
Usage Page（Generic Desktop Control）	0x05 01	Global	用法ページ（Usage Page）を示す値。0x01 は"Generic Desktop Control"を表す
Usage（X）	0x09 30	Local	アイテム使用法のインデックス。0x30 は"X"を表す。コントローラが報告する値は X 方向の値で、ユーザから見て「左」から「右」にコントローラを動かしたとき、リニアに値が増加。
Usage（Y）	0x09 31	Local	アイテム使用法のインデックス。0x31 は"Y"を表す。コントローラが報告する値は Y 方向の値で、ユーザから見て「離れている」から「近い」にコントローラを動かしたとき、リニアに値が増加。

Item	Value（Hex）	Item 区分	意　　味
Usage（Wheel）	0x09 38	Local	アイテム使用法のインデックス。0x38 は"Wheel"を表す。ダイヤルとは異なり、回転され可変値を生成するロータリー・コントロール。コントローラが前方に、（ユーザから遠ざかるように）回転すると、値は増加。
Logical Minimum（-127）	0x15 81	Global	アイテムが報告する最小値は-127
Logical Maximum（127）	0x25 7F	Global	アイテムが報告する最大値は 127
Report Size（8）	0x75 08	Global	レポートフィールドの大きさを示す。ここでは 8 ビット使用
Report Count（3）	0x95 03	Global	アイテムにデータフィールドをいくつ使うかを示す。ここでは 3 個のレポートフィールドを使用。
Input（Data, Variable, Relative）	0x81 06	Main	入力するアイテムがどのようなものかを示します。入力は可変データで、前回の入力からの変化分を報告。
End Collection	0xC0	Main	データセットとして 1 つにまとめる際の終点を表す（Physical）
End Collection	0xC0	Main	データセットとして 1 つにまとめる際の終点を表す（Application）

2.7　フィジカルディスクリプタ

フィジカルディスクリプタは、ファンクションをコントロールしている人の体（あるいは体の特定の部分）に関する情報を提供するために存在します。このディスクリプタはオプションであり、省略することが可能です。このサンプルプログラムでは省略しています。

2.8　HIDデータの転送フォーマット

USB ホストとファンクション間では、HID データを転送する場合、主にインタラプト転送を使用します（コントロール転送も使用可）。

ブートデバイスが使用可能なプロトコルは、レポートプロトコルとブートプロトコルの 2 種類で、非ブートデバイスが使用可能なプロトコルは、レポートプロトコルの 1 種類です。

レポートプロトコルがデータ転送に使用するフォーマットはレポートディスクリプタで作成します。

ブートプロトコルがデータ転送に使用するフォーマットは規格書にデータフォーマットが記されています。

ブートデバイスのデフォルトプロトコルはレポートプロトコルですが、ブートプロトコルまたはレポートプロトコルのどちらを使用するかはクラスコマンドで指定することができます。図 2.4 にこのサンプルプログラムのレポートプロトコルフォーマットを示します。

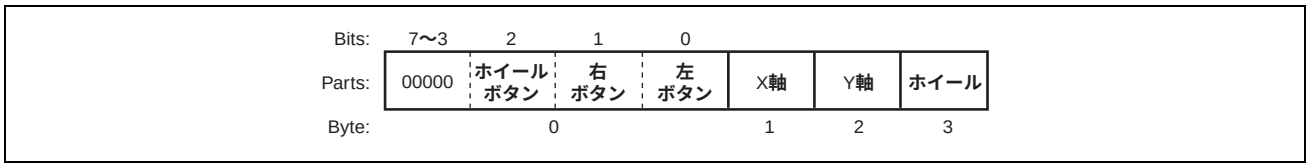


図 2.4　レポートプロトコルフォーマット

2.9 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 HID Class のクラスコマンドは 6 種類あります。表 2.9 にクラスコマンド一覧を示します。

表 2.9 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
0x01	GET_REPORT	コントロール転送を使い、ファンクションから USB ホストに HID データを転送要求
0x02	GET_IDLE	インタラプト転送の間隔(アイドルレート)の現在値を報告要求
0x03	GET_PROTOCOL	現在選択されているプロトコル（ブートプロトコルまたはレポートプロトコル）を報告要求
0x09	SET_REPORT	コントロール転送を使い、USB ホストからファンクションに HID データを転送
0x0A	SET_IDLE	インタラプト転送の間隔(アイドルレート)を設定
0x0B	SET_PROTOCOL	使用するプロトコル（ブートプロトコルまたはレポートプロトコル）を設定

【注】*1 すべてのファンクションは GET_REPORT をサポートする必要があります。

*2 ブートデバイスは GET_PROTOCOL と SET_PROTOCOL をサポートする必要があります。

GET_REPORT コマンドを受信した場合、ファンクションはコントロール転送のデータステージを使用して HID データをホストに送信します。セットアップデータ内 wValue フィールドの上位 1 バイトでレポートタイプを指定し、wValue フィールドの下位 1 バイトでレポート ID を指定します。レポート ID を使用しない場合は値 0 が指定されます。

GET_IDLE コマンドを受信した場合、ファンクションはインタラプト転送の間隔(アイドルレート)を返答します。返答には、4ms を 1 単位とするタイムレートを用います。ホストはセットアップデータ内 wValue フィールドの下位 1 バイトで返答するレポート ID を指定します。この値が 0 の場合、当該ファンクションの全インタラプト転送のタイムレートを返答します。

GET_PROTOCOL コマンドを受信した場合、ファンクションはコントロール転送のデータステージを使用して現在選択されているプロトコル (ブートプロトコルまたはレポートプロトコル) をホストに返答します。返答値 0 はブートプロトコル、返答値 1 はレポートプロトコルを表します。

SET_REPORT コマンドを受信した場合、ファンクションはコントロール転送のデータステージを使用した HID データを受信します。しかし、ファンクションはホストからの指示を無視する可能性があります。

SET_IDLE コマンドを受信した場合、ファンクションはインタラプト転送の間隔(アイドルレート)を設定します。ファンクションは設定時間が経過するか、新しいイベントが発生するまでインタラプト転送を抑止します。セットアップデータ内 wValue フィールドの上位 1 バイトで時間を指定します。なお、時間は、4ms を 1 単位とするタイムレートで表されます。wValue フィールドの下位 1 バイトにはレポート ID が指定されます。この値が 0 以外の場合、指定されたレポート ID の転送を停止します。この値が 0 の場合、当該ファンクションの全インタラプト転送を停止します。

SET_PROTOCOL コマンドを受信した場合、ファンクションはそれ以降に使用するプロトコル（ブートプロトコルまたはレポートプロトコル）を設定します。設定値（値 0 はブートプロトコル、値 1 はレポートプロトコル）はセットアップデータ内 wValue フィールドで指定されます。なお、ファンクションはレポートプロトコルを初期値としています。

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。

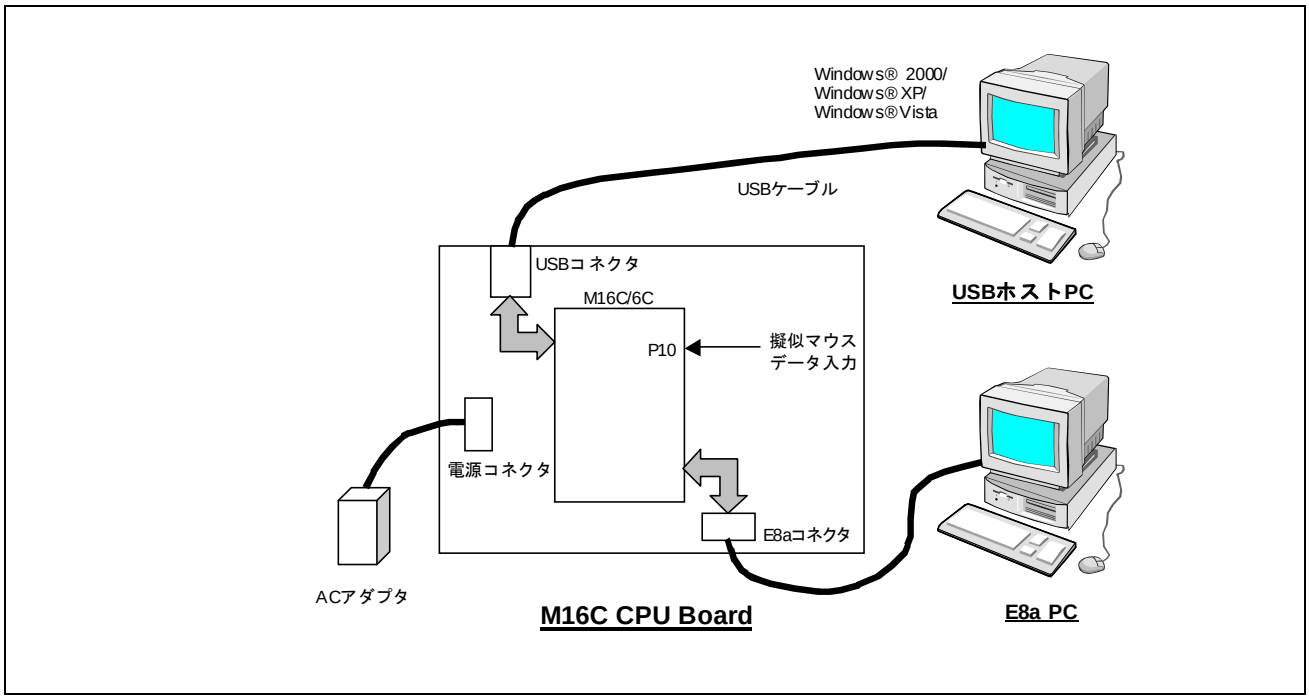


図 3.1 デバイスの接続形態

(1) M16C CPU Board

表 3.1 に M16C CPU Board の擬似マウスデータ入力ポートを示します。

表 3.1 擬似マウスデータ入力ポート

端子名	信号名	端子の状態	機能
P8_4	LEFT_CLICK	0= ON	マウスを左クリックしたときのデータを生成。
P8_3	MOVE_X	0= ON	ポインタがX方向の移動データを生成。移動方向は、スイッチを一度離してから再度押すと変更します。
P8_2	MOVE_Y	0= ON	ポインタがY方向の移動データを生成。移動方向は、スイッチを一度離してから再度押すと変更します。

【注】端子の状態は、“H”レベルの場合を“1”、“L”レベルの場合を“0”とし、以降“1”または“0”で記します。

(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 HID クラス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“HID.hws”というワークスペースファイルは、HID 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 HID フォルダ内のファイルを示します。

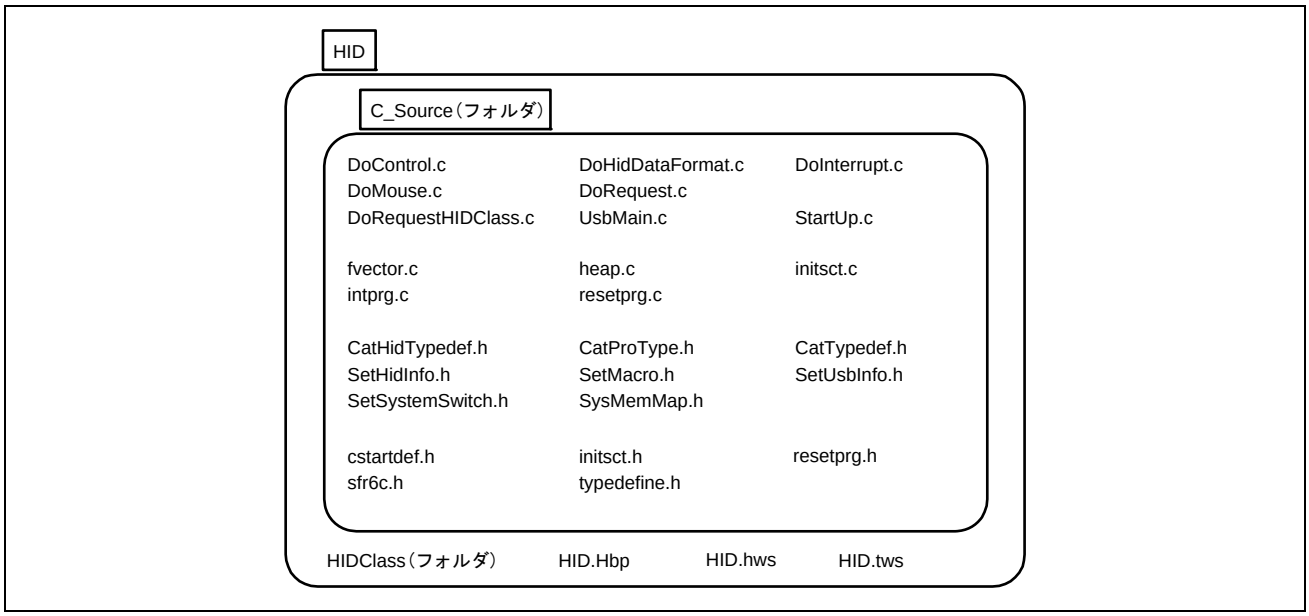


図 3.2 フォルダ内ファイル

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)HID フォルダを任意の場所にコピーしてください。

(2)“HID.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、“HID¥HIDClass¥Debug”フォルダ内に、実行ファイルである“HID.mot”と、マップファイルである“HID.map”が作成されます。HID.mot はモトローラ S タイプフォーマットファイルです。HID.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。

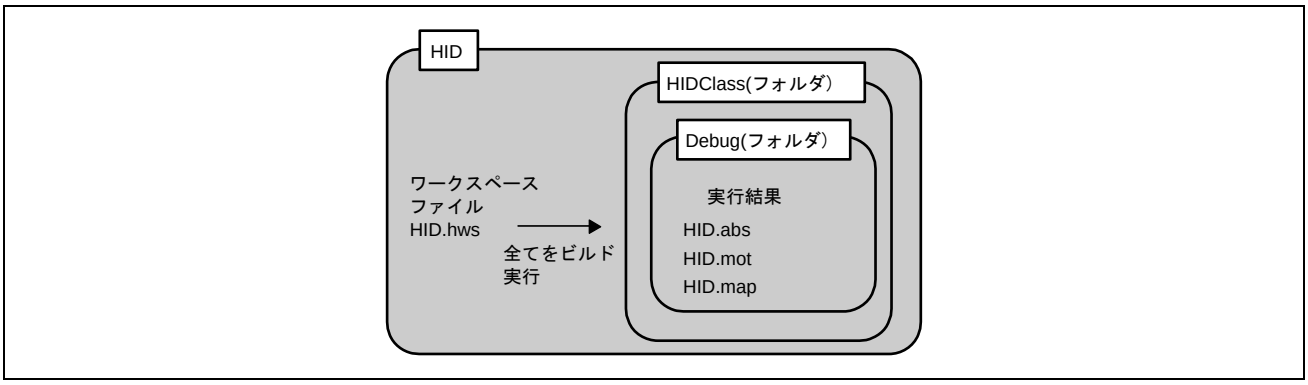
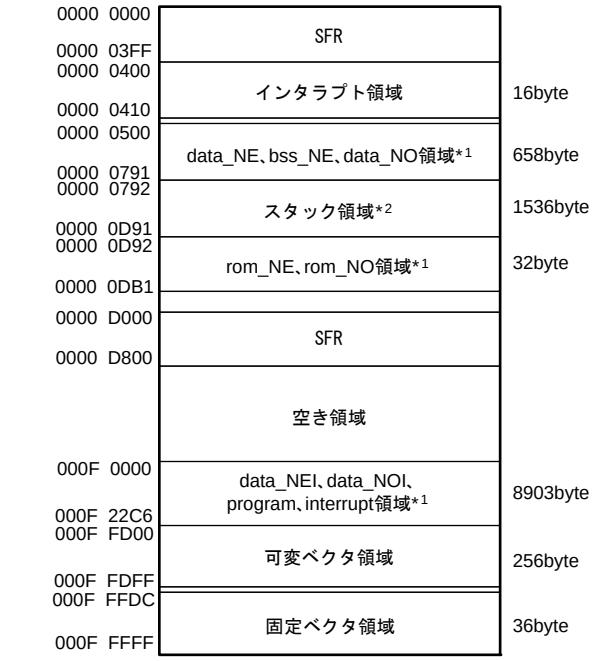


図 3.4　コンパイル結果

3.3 プログラムのロードと実行方法

図 3.5 にサンプルプログラムのメモリマップを示します。



【注】 *1 コンパイラのバージョン、コンパイル条件、ファームウェアの
アップグレードなどによりメモリマップは変化します。
*2 スタック領域はこの領域すべてを使用するものではありません。

図 3.5 メモリマップ

図 3.5 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 を内蔵フラッシュメモリ領域上に配置、スタック、data_NE、bss_NE、data_NO、rom_NE、rom_NO を内蔵 RAM 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 HID¥HIDClass¥Debug フォルダ内の HID.map に出力されます。

3.3.1 プログラムのロードと実行

以下に、サンプルプログラムのロード手順を示します。

1. HEW4をインストールしたE8a用PCにE8aを接続してください。
2. 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。
3. M16C CPU Boardの電源を投入してください。
4. HIDフォルダ内のHID.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。
6. 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。
7. 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 擬似マウス操作（カーソル移動）の実行方法

このサンプルプログラムでは、マウスを接続せずに擬似マウス操作（カーソル移動）のデモンストレーションを行います。

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 M16C CPU Board に、シリーズ A コネクタを USB ホスト PC に接続します。コントロール転送終了後、デバイスマネージャにヒューマンインタフェースデバイス/USB ヒューマンインタフェースデバイスが表示され、USB ホスト PC は M16C CPU Board をマウスデバイスとして認識します。

USB ホスト PC に接続後、M16C CPU Board の P8_2～P8_4 を“0”にすることにより、マウスの移動データが生成されます。

P8_2 を“0”にするとマウスを左クリックした際のデータを生成。

P8_3 を“0”にするとポインタが X 方向に移動するデータを生成。移動方向は、スイッチを一度離してから再度押すと変更します。

P8_4 を“0”にするとポインタが Y 方向に移動するデータを生成。移動方向は、スイッチを一度離してから再度押すと変更します。

USB ホストからのインタラプト IN 転送に応じて M16C CPU Board がマウス擬似データを送出します。その結果、USB ホスト PC 上のカーソルが移動を開始します。

4. サンプルプログラム概要

この章ではサンプルプログラムの特長やその構成について説明します。このサンプルプログラムは M16C CPU Board 上で動作する HID クラスファームウェアです。本ボード上でマウスデータを生成することにより、マウスの動作をエミュレートします。また、USB ホストからのトークンによって USB 転送を開始します。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
- インタラプトIN転送でUSBホストPCにマウス擬似データを送信可能

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。

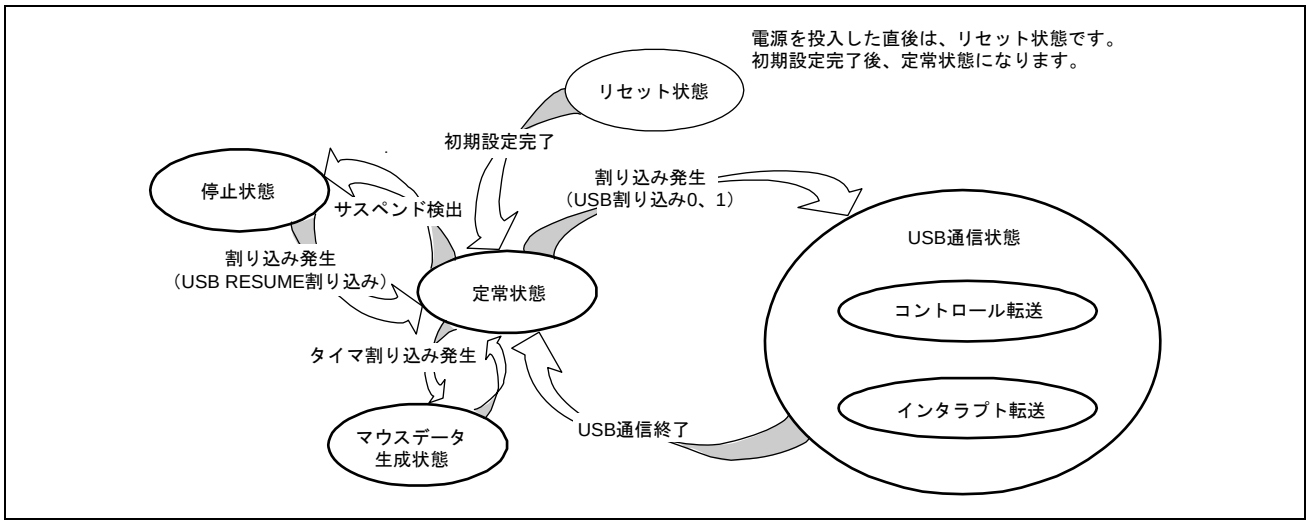


図 4.1 状態遷移図

-
- 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 9 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。
- マウスデータ生成状態
  定常状態で 16 ビットタイマ TGRA_2 のコンペアマッチ割り込みが発生すると、この状態になります。マウスデータ生成状態ではボード上のボタンを使用し、マウスの代わりにマウスデータを生成もしくは、移動データを自動生成します。コンペアマッチ割り込みは 16 または 10ms 間隔で発生します。
- 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。

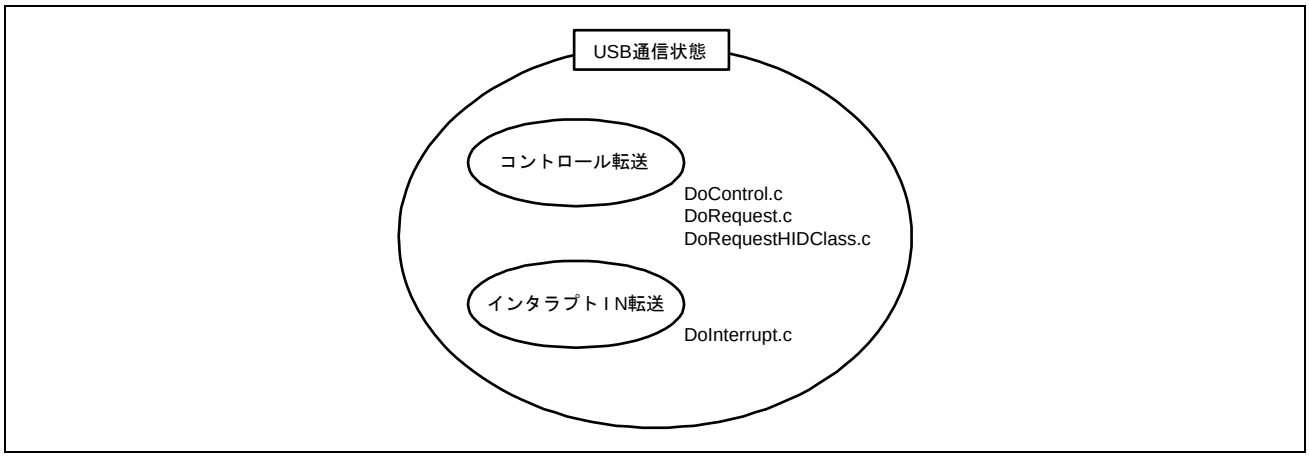


図 4.2　USB 通信状態

4.3 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 13 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。

表 4.1 ファイル構成

ファイル名	主な役割
DoControl.c	コントロール転送実行
DoHidDataFormat.c	転送する HID データのフォーマット処理
DoInterrupt.c	インタラプト IN 転送実行
DoMouse.c	マウスデータ生成処理
DoRequest.c	USB ホストが発行するセットアップコマンドの処理
DoRequestHIDClass.c	HID クラスコマンドの処理
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
CatHidTypedef.h	HID クラス固有の型、構造体定義
CatProType.h	プロトタイプ宣言
CatTypedef.h	USB ファームウェアで使用する基本の構造体定義
SetHidInfo.h	HID クラス対応に必要な変数の初期設定
SetMacro.h	マクロ定義
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
initsct.h	セクション定義ファイル
resetprg.h	スタックサイズ定義ファイル
sfr6c.h	M16C/6C のレジスタ定義
typedefine.h	各アクセスサイズの定義

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。

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 4.2　UsbMain.c

格納ファイル	関数名	機　　能
UsbMain.c	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
	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
	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（このサンプルプログラムでは使用しません）
	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版高速版
	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（このサンプルプログラムでは使用しません）
	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版高速版 （このサンプルプログラムでは使用しません）
	SetControlOutContents	USB 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
	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。

StartUp.c

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、USB クロックなどの初期設定を行います。

表 4.3　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
	InitMemory	メモリ領域の設定
	InitSystem	USB クロックの設定、システム割り込みマスクの設定、タイマの設定
	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

DoRequest.c

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、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、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

DoRequestHIDClass.c

HID Class コマンド（GET_REPORT,GET_IDLE,GET_PROTOCOL, SET_REPORT, SET_IDLE, SET_PROTOCOL）に応じた処理を行います。

- GET_REPORTコマンドは、コントロール転送を使い、ファンクションからUSBホストにHIDデータを転送要求します。
- GET_IDLEコマンドは、インタラプト転送の間隔(アイドルレート)の現在値を報告要求します。
- GET_PROTOCOLコマンドは、現在選択されているプロトコル（ブートプロトコルまたはレポートプロトコル）を報告要求します。
- SET_REPORTコマンドは、コントロール転送を使い、USBホストからファンクションにHIDデータを転送します。このサンプルではHIDデータのOUT方向の通信をサポートしないので、データの受信のみ行います。
- SET_IDLEコマンドは、インタラプト転送の間隔(アイドルレート)を設定します。
- SET_PROTOCOLコマンドは、使用するプロトコル（ブートプロトコルまたはレポートプロトコル）を設定します。

表 4.5　DoRequestHIDClass.c

格納ファイル	関数名	機　　能
DoRequestHIDClass.c	DecHIDClassCommands	HID クラスコマンドに応じた処理を行う
	ActIdleCount	SOF 割り込みで呼び出され、インタラプト転送の間隔(アイドルレート)を計算する

DoControl.c

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応じて、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

DoInterrupt.c

USB ホストからのインタラプト転送 IN トークンに対応し、インタラプト転送バッファが空き次第、次に転送するデータの準備を行います。

表 4.7　DoInterrupt.c

格納ファイル	関数名	機　　能
DoInterrupt.c	ActInterruptIn	インタラプト転送の IN トークンに対応し FIFO が空き次第データ転送バッファからデータを取り出しインタラプト転送の準備を行う

DoHidDataFormat.c

USB ホストに送信する HID データの送信準備を行います。

表 4.8　DoHidDataFormat.c

格納ファイル	関数名	機　　能
DoHidDataFormat.c	ActMakeHidData	この関数は HID データ伝送のプログラム・インタフェース ActReportProtocol 関数を呼び出した後、インタラプト転送が停止しているなら、ActInterruptIn 関数を呼び出す
	ActReportProtocol	転送するデータの並びをレポートディスクリプタで決められたフォーマットに整え、送信バッファにデータを書く

DoMouse.c では、タイマ割り込みを使用してマウスのデータを生成します。

表 4.9　DoMouse.c

格納ファイル	関数名	機　　能
DoMouse.c	MousePushedDataInput	操作モード、デモモードの判定を行う
	MousePushedDataInput1	キースキャンを行い、マウスのデータを作成する
	MousePushedDataInput2	タイマでの時間経過に応じてマウス移動データを生成する

図 4.3 に表 4.2～4.9 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 4.3 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。

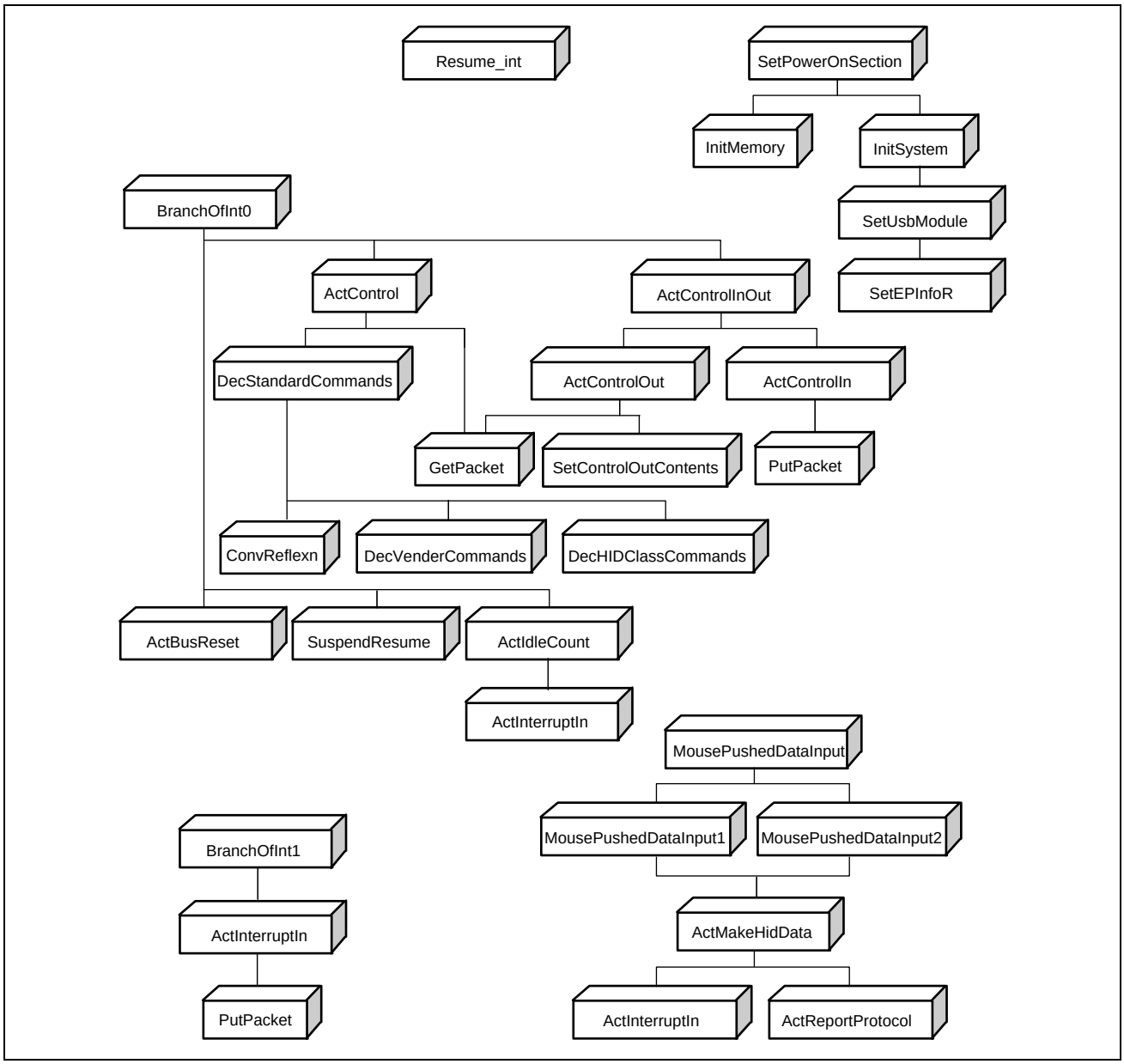


図 4.3　関数の相関関係

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。

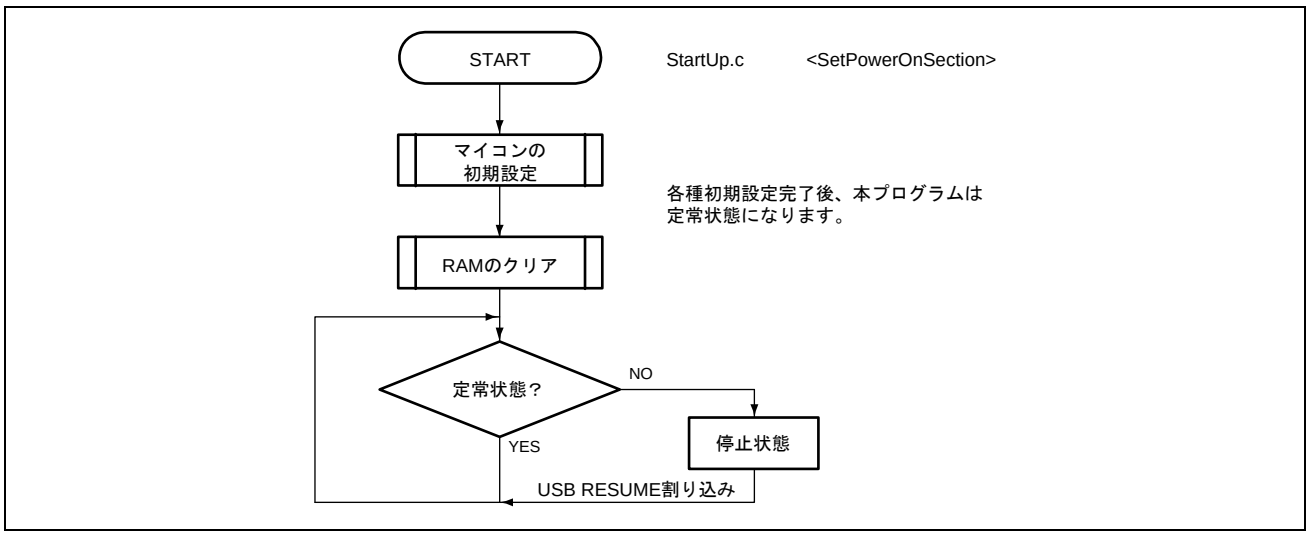


図 5.1 メインルーチン

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 9 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。

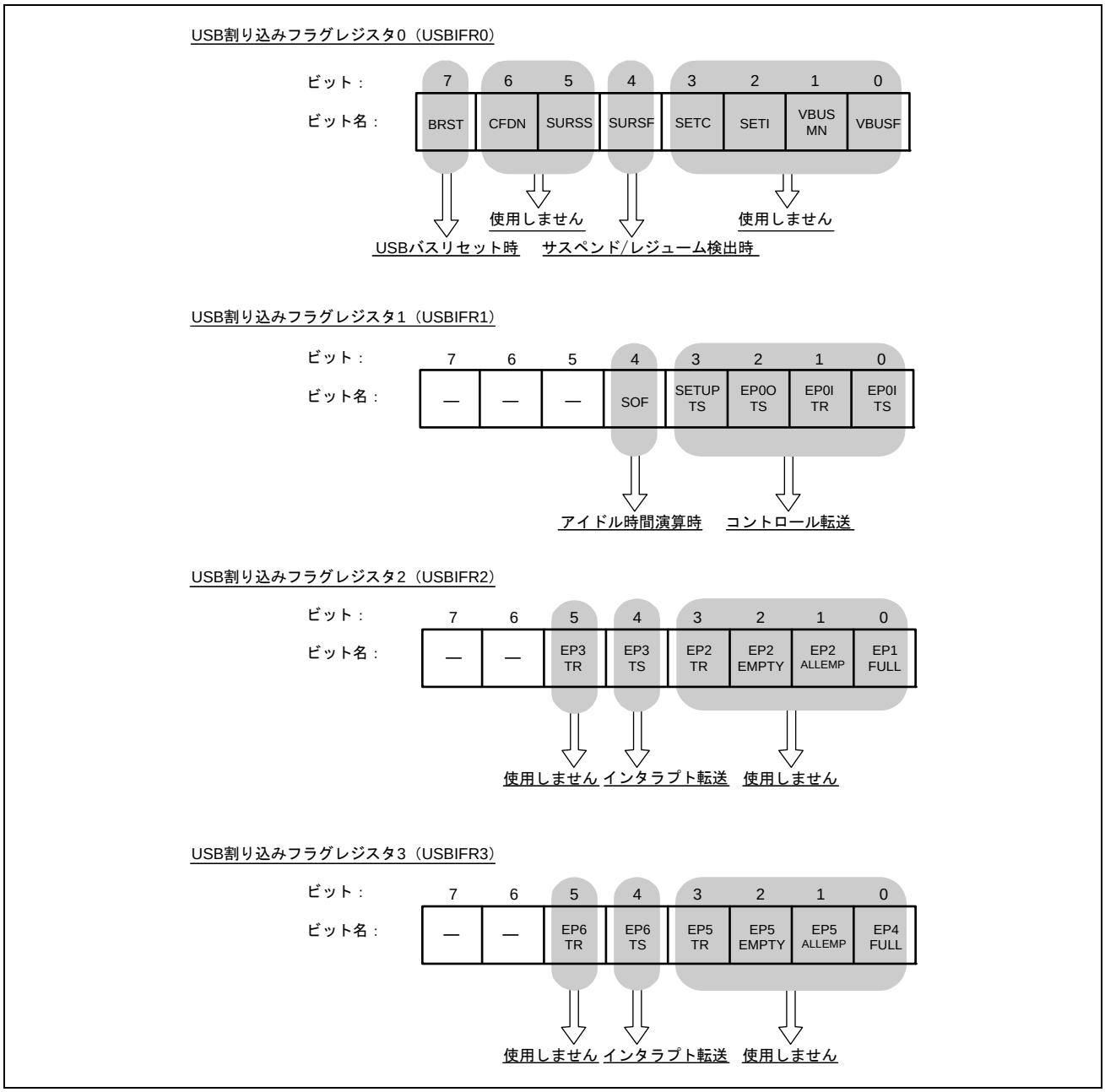


図 5.2 割り込みフラグの種類

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	ActIdleCount
	3	SETUP TS	ActControl
	2	EP0O TS	ActControlInOut
	1	EP0I TR	ActControlInOut
	0	EP0I TS	ActControlInOut
USBIFR2	7	−	−
	6	−	−
	5	EP3 TR	ActInterruptIn
	4	EP3 TS	−
	3	EP2TR	−
	2	EP2 EMPTY	−
	1	EP2 ALLEMP	−
	0	EP1FULL	−
USBIFR3	7	−	−
	6	−	−
	5	EP6 TR	ActInterruptIn
	4	EP6 TS	−
	3	EP5TR	−
	2	EP5 EMPTY	−
	1	EP5 ALLEMP	−
	0	EP4FULL	−

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。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。

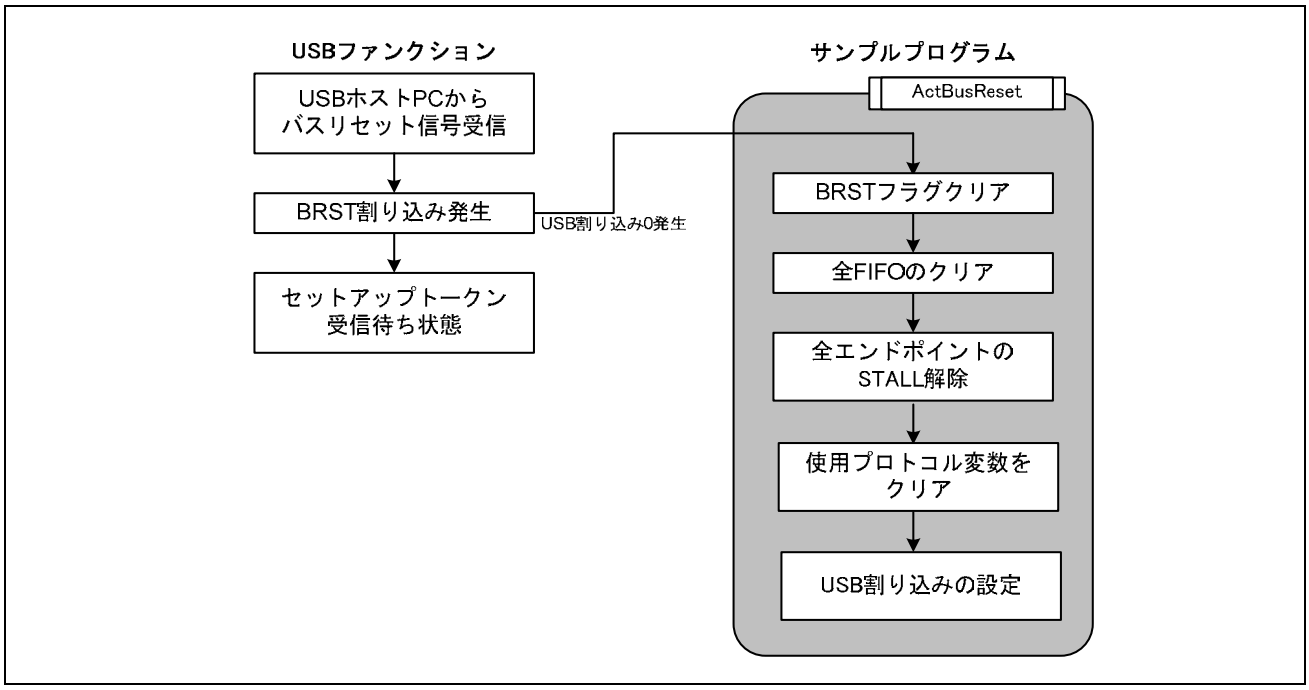


図 5.3　バスリセット割り込み

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。

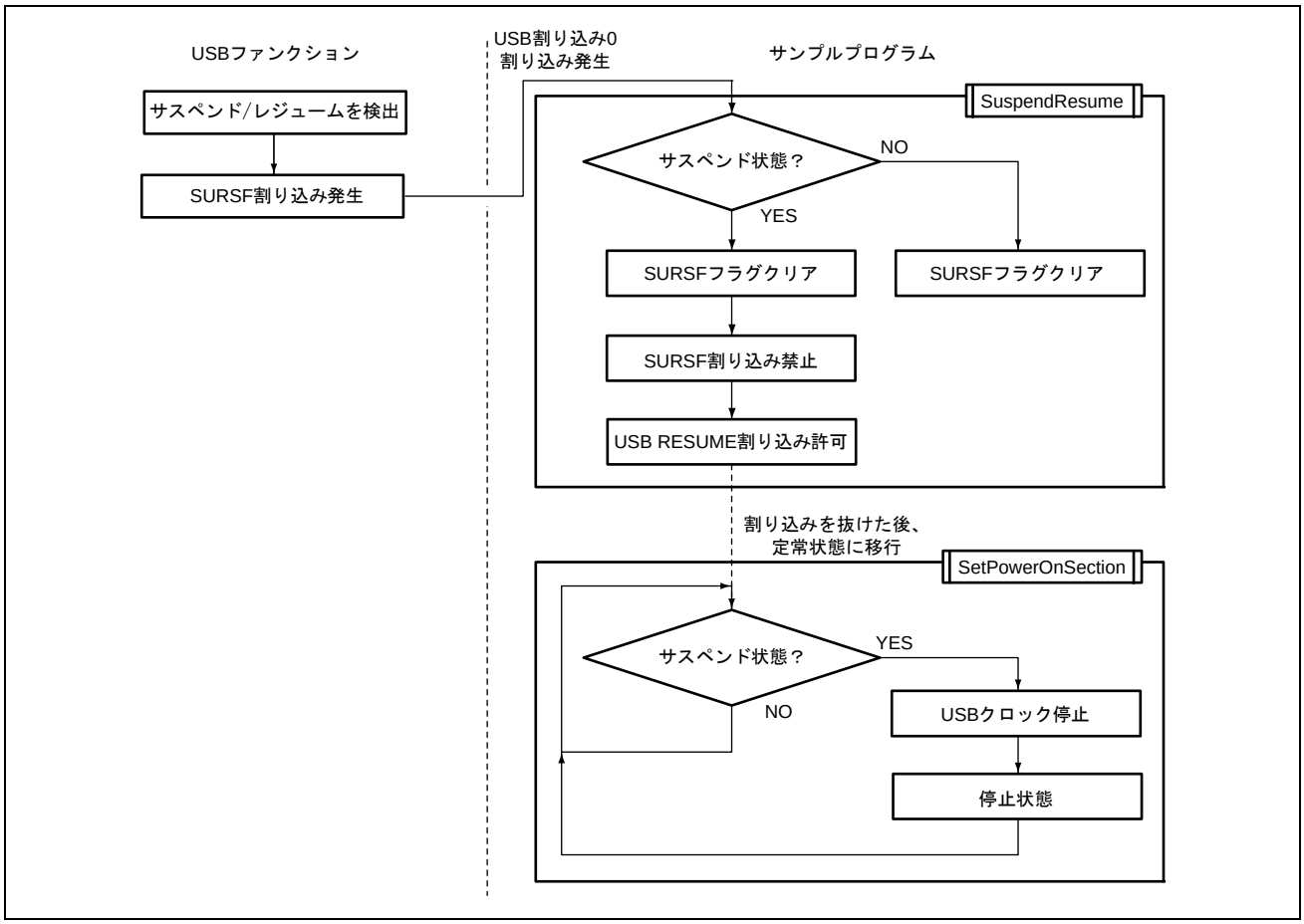


図 5.4　サスペンド/レジューム割り込み

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。

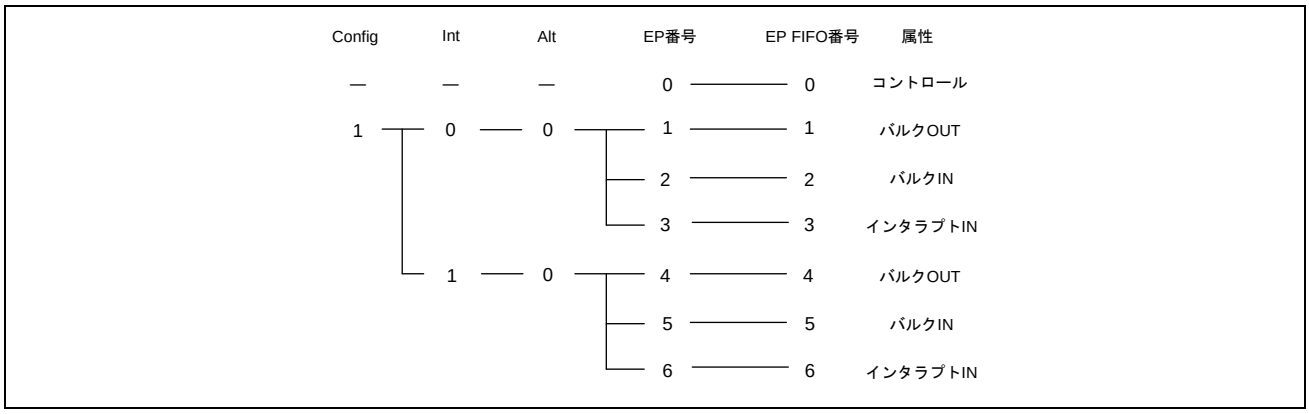


図 5.5 エンドポイント構成

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。

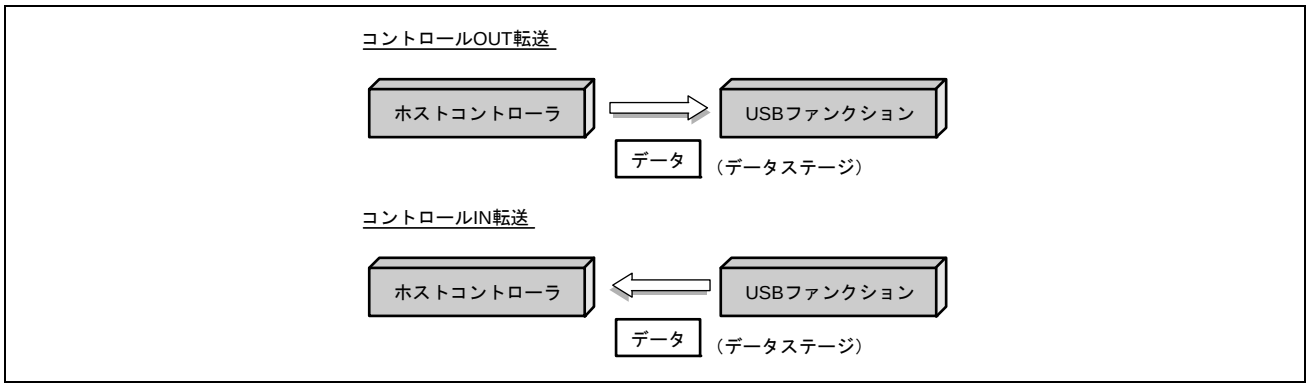


図 5.6　データステージにおけるデータの方向(コントロール転送)

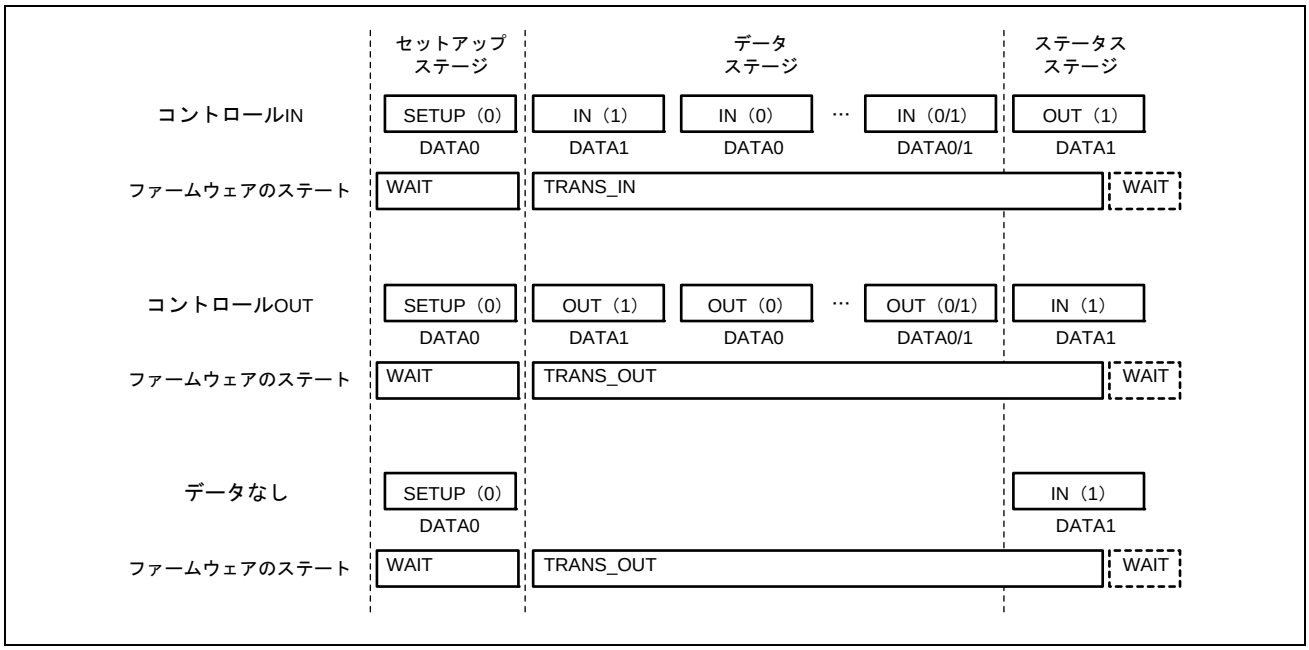


図 5.7　コントロール転送における各ステージの構成

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。セットアップステージでは、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

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。

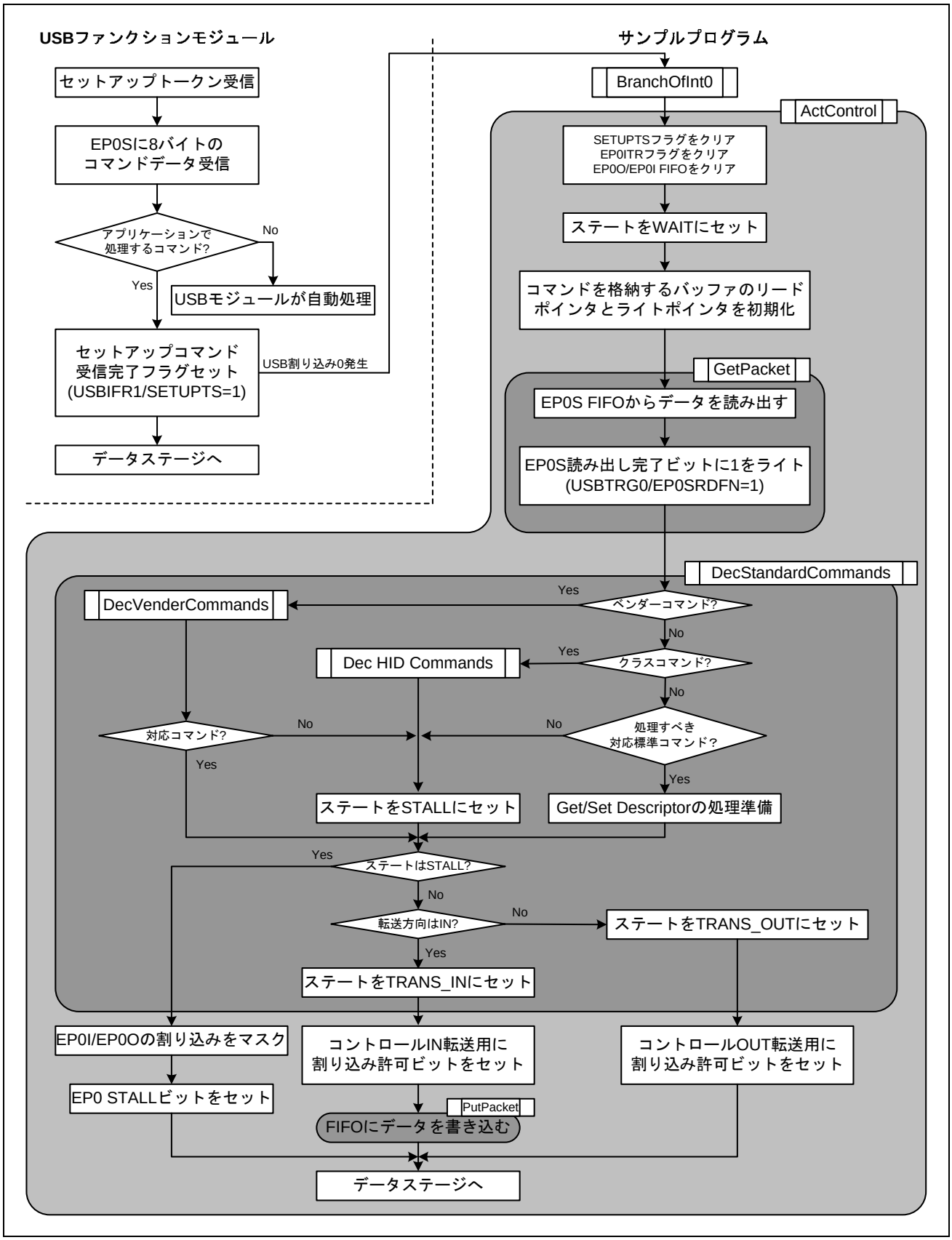


図 5.8　セットアップステージ

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。

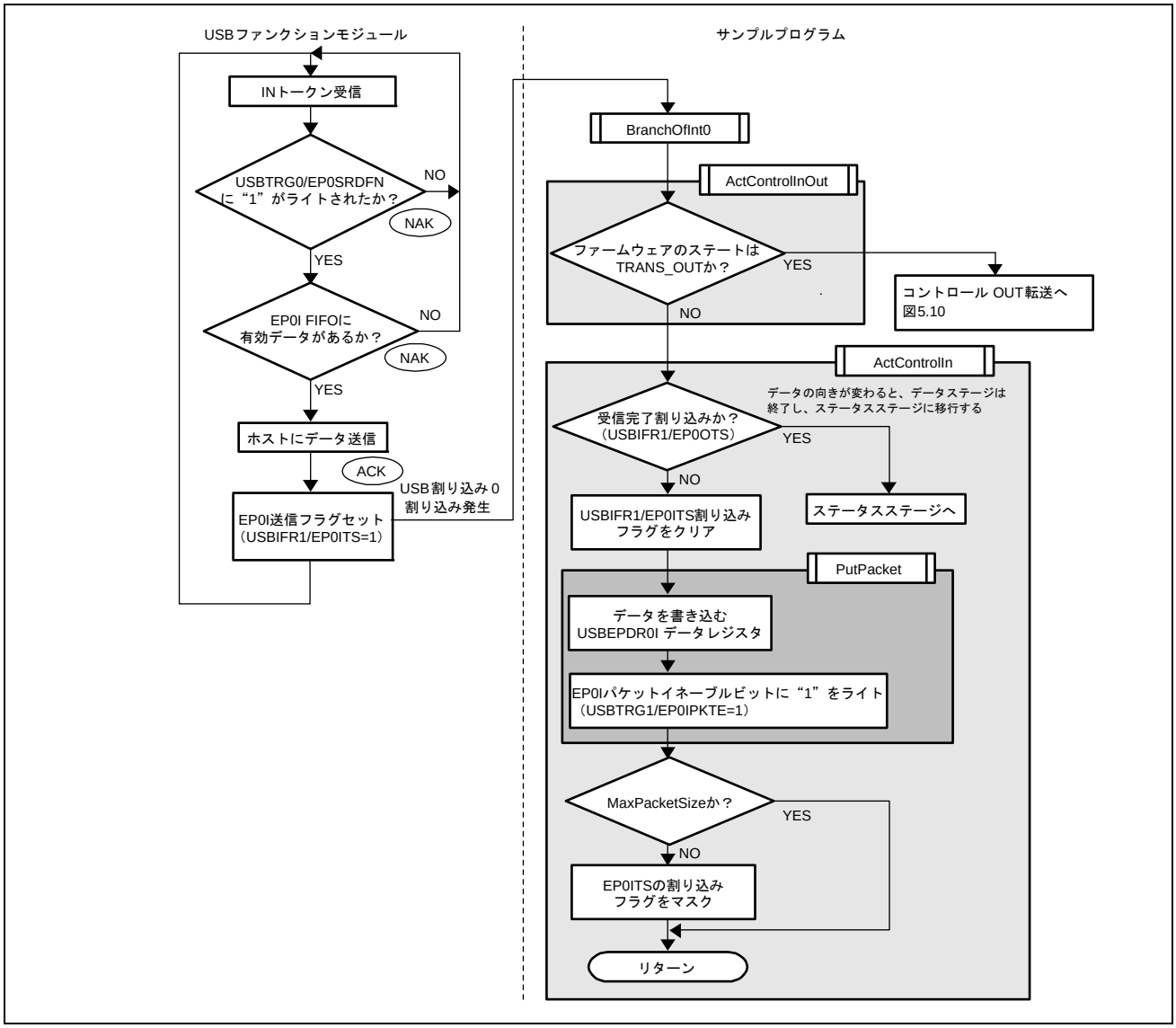


図 5.9　データステージ（コントロール IN 転送）

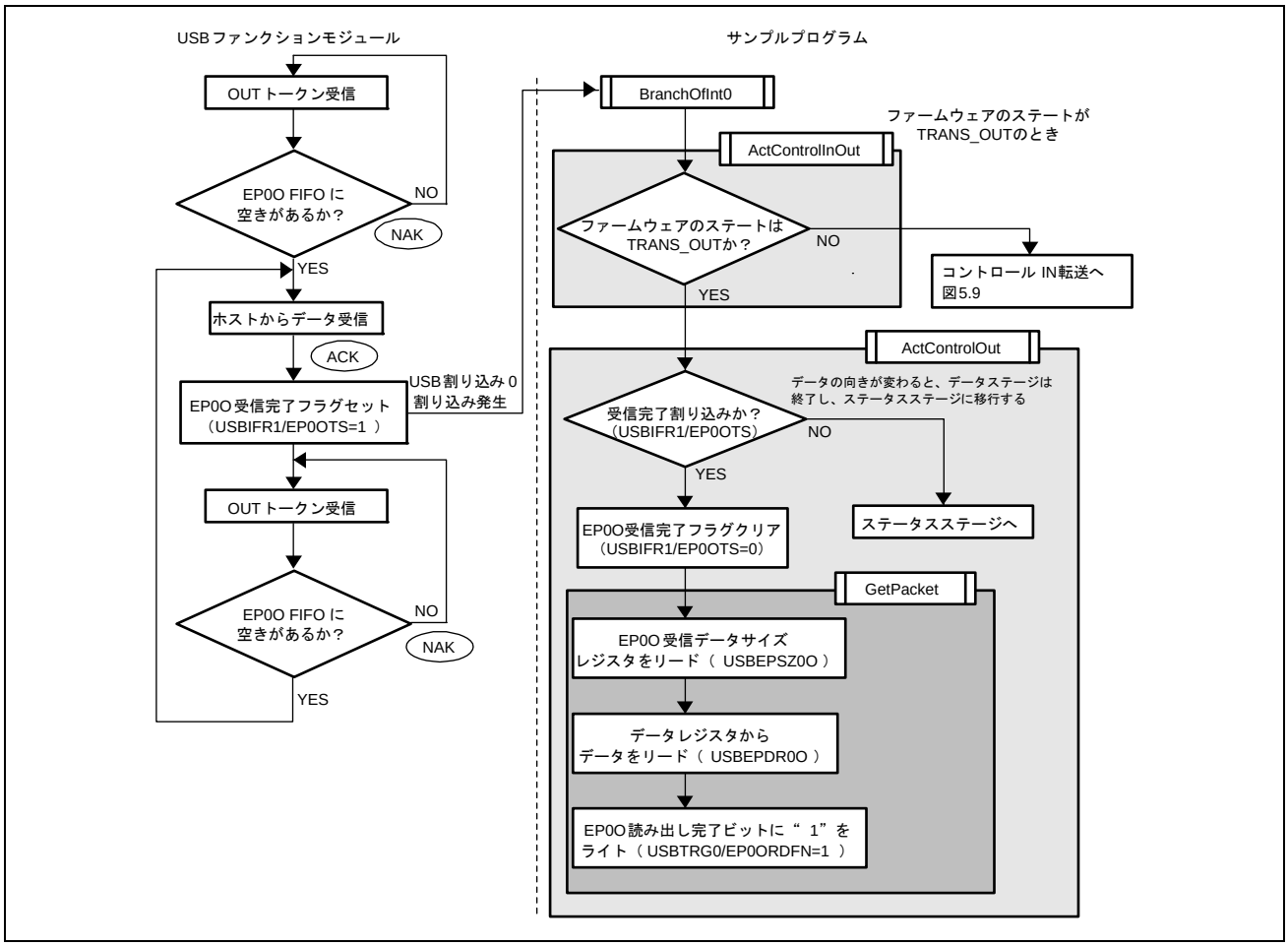


図 5.10　データステージ（コントロール OUT 転送）

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。

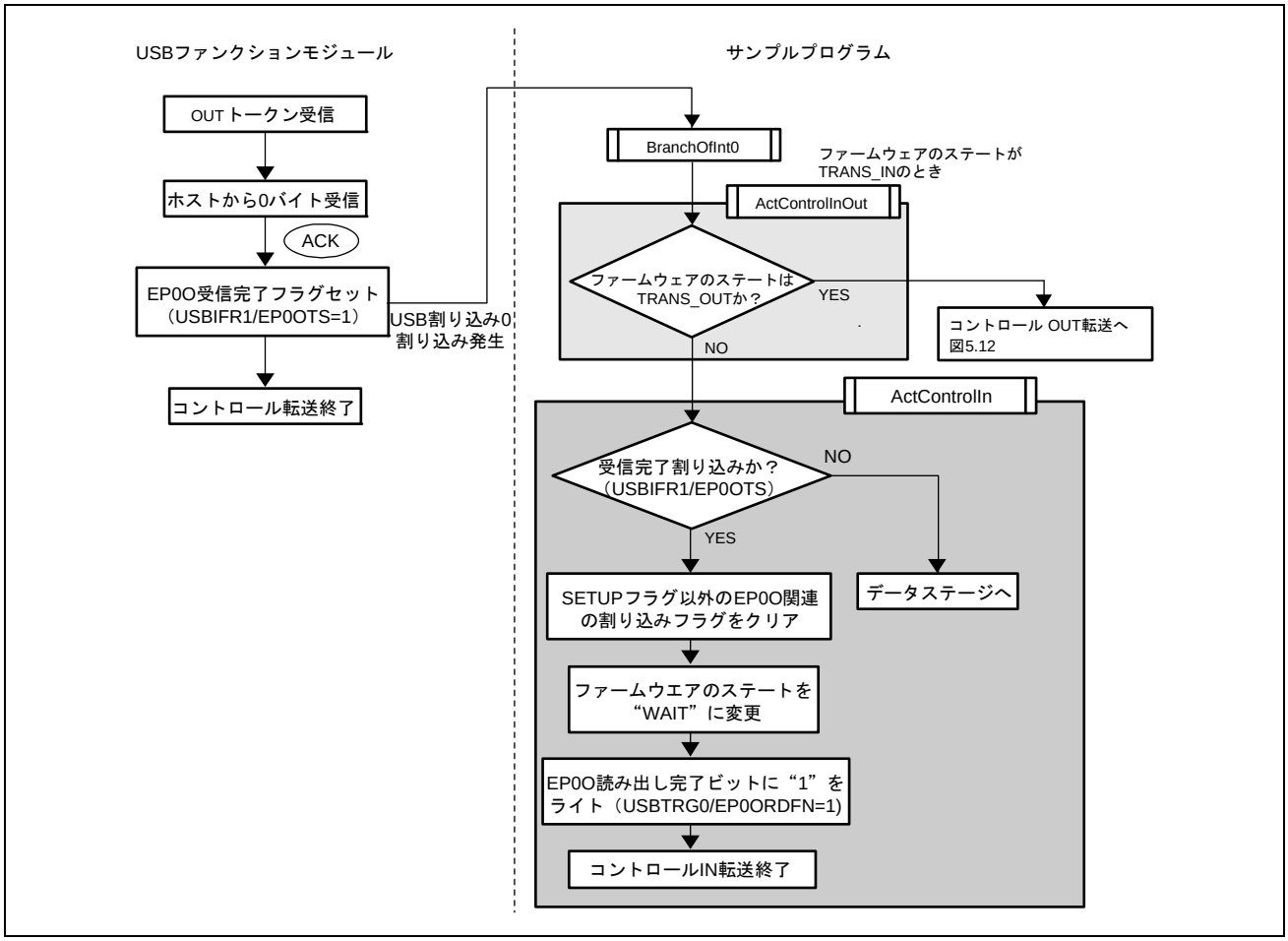


図 5.11　ステータスステージ（コントロール IN 転送）

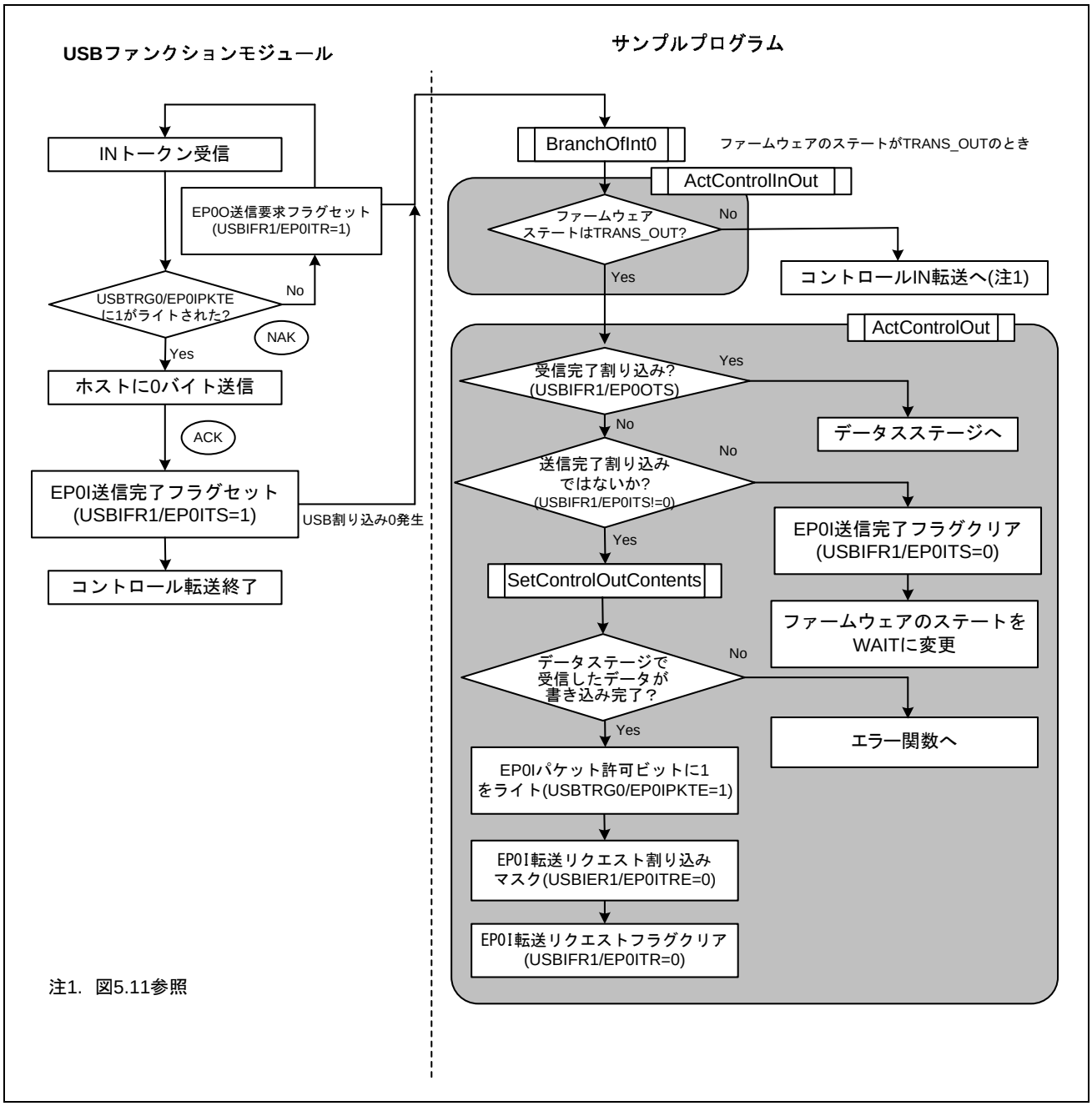


図 5.12　ステータスステージ（コントロール OUT 転送）

5.7 インタラプト転送

インタラプト転送は、データの方向によって 2 つに分けることができます。USB ファンクションから USB ホストへデータを送信する場合がインタラプト IN 転送、その逆がインタラプト OUT 転送です。M16C/6C はインタラプト IN 転送のみサポートしています。尚、以降の説明ではエンドポイント 3 を使用した場合を例に挙げ説明します。

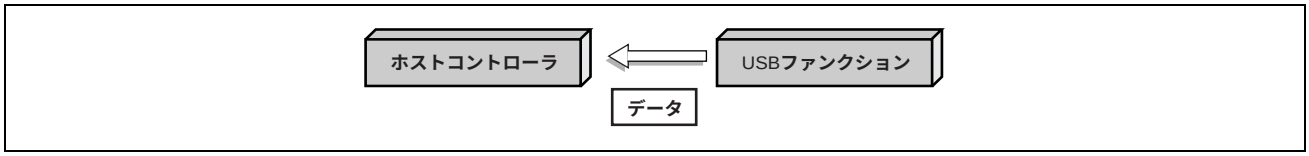


図 5.13 インタラプト転送

5.7.1 インタラプトIN転送

このサンプルプログラムでは、インタラプト IN 転送の処理に USB 割り込みフラグレジスタ 2 のビット 4（EP3TS）を使用します。

USB ホストからの IN トークン受信時、USB ファンクションは以下の状態になります。

EP3 FIFO に有効データがなかった場合: NAK ハンドシェークを送信する、EP3TR フラグがセットされます。

EP3 FIFO に有効データがあった場合: USB ホストにデータを送信する、その後 USB ホストからの ACK ハンドシェークを受信すると EP3TS フラグがセットされます。

USB ファンクションは EP3TS フラグがセットされると ActInterruptIn 関数を実行します。ActInterruptIn 関数では送信する HID データがある場合、USB エンドポイントデータレジスタ 3（USBEPDR3）にデータをライトします。ライトしたデータは次の IN トークン受信時に USB ホストに送信されます。この時、ファームウェアは“WAIT”、“TRANS_IN”どちらかのインタラプト転送ステートです。

図 5.14 にサンプルプログラムのインタラプト IN 転送処理を示します。図の左側は、USB ファンクションモジュールの動作を示しています。

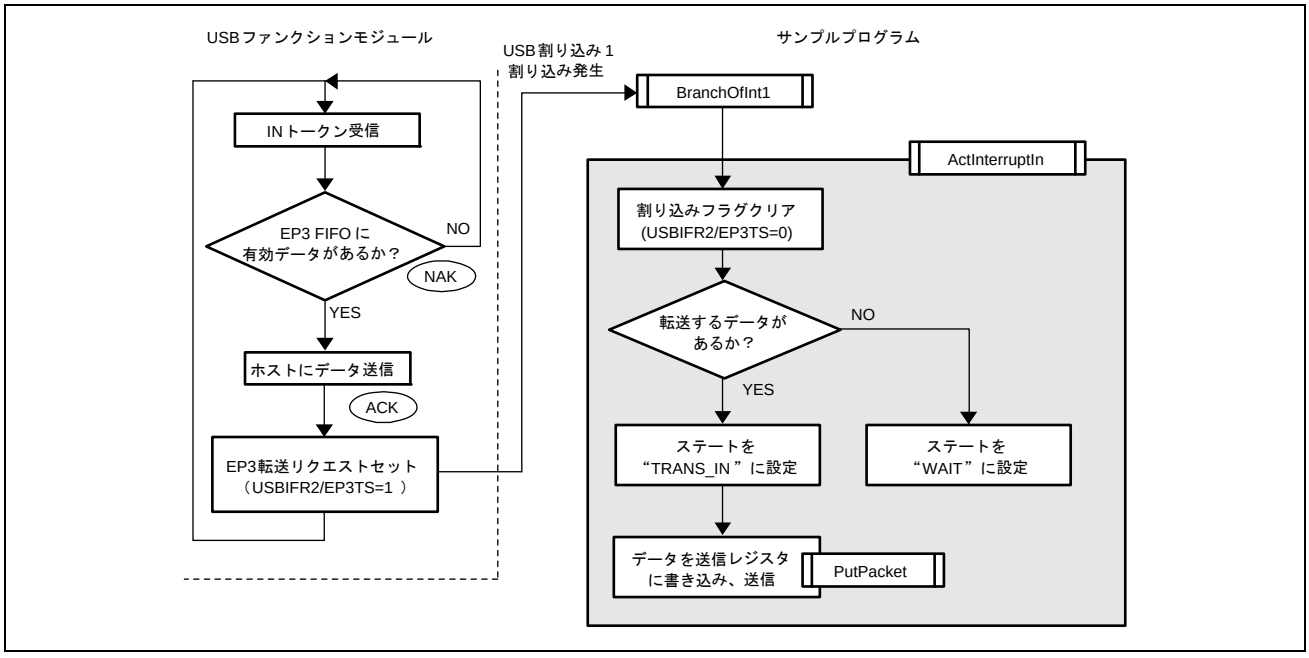


図 5.14 インタラプト IN 転送

5.8 マウスデータの生成

M16C CPU Board にはマウスが接続できないため、このサンプルプログラムでは M16C CPU Board 上の P8 を使用して USB マウスの疑似データ（HID データ）を生成し、USB マウスとして動作させています。

8 ビットタイマ割り込みを使い、M16C CPU Board 上の P8_2～P8_4 のキースキャンを行うことにより、HID データを生成します。

P8_2 を"0"にするとマウスを左クリックしたときのデータを生成。

P8_3 を"0"にするとポインタが X 方向に移動するデータを生成。移動方向は、スイッチを一度離してから再度押すと変更します。

- P8_4を"0"にするとポインタがY方向に移動するデータを生成。移動方向は、スイッチを一度離してから再度押すと変更します。

これらのソフトウェアを使用して生成したデータを ActMakeHidData 関数に引き渡すと、インタラプト転送によって USB ホストに HID データを送信します。**図 5.15** にサンプルプログラムの HID データ生成動作を示します。

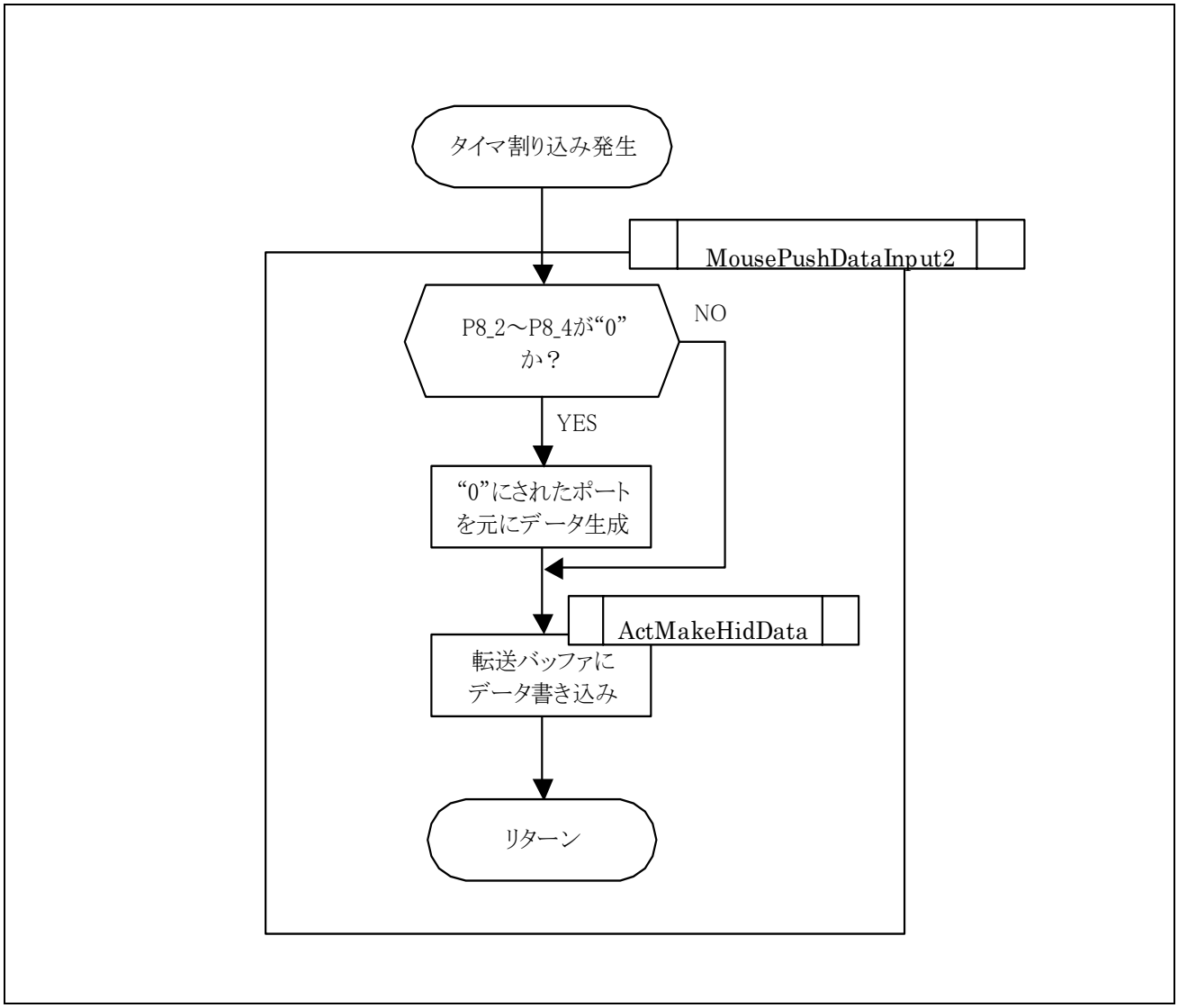


図 5.15　マウスデータ生成

6. アナライザのデータ

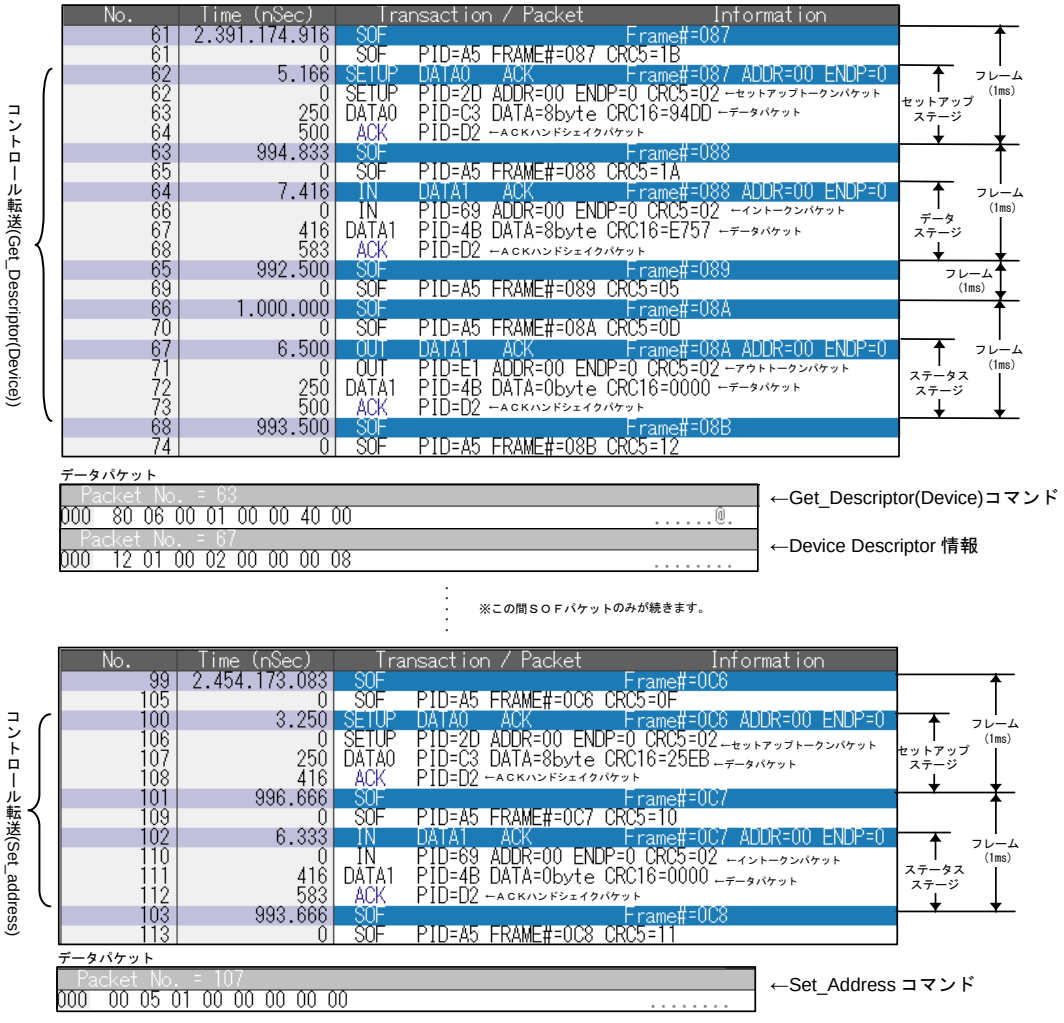
この章では、M16C/6C内蔵 USB ファンクションモジュールを使用して、ヒロテック社製 USB プロトコルアナライザ「HUSB200」を用いた測定を行い、実際にバスを流れているデータについて「デバイス接続時のコントロール転送」および「HID データのインタラプト IN 転送」を例に説明します。

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。

6.1 デバイス接続時のコントロール転送

以下に示す図 6.1 は本デバイスを USB ホストに接続し、Vbus に電源が供給されている（電源投入ステート）状態から、デバイスが使用可能になる状態（構成ステート）に至るまでを測定したものです。

なお、USB ホストによりパケットのスケジューリングが本図と異なる場合がありますが、構成ステートに至るまでのコマンドの流れは同一です。



※この間ＳＯＦパケットのみが続きます。

コントロール転送(Get_Descriptor(Device))

No.	Time (nSec)	Transaction / Packet	Information	
163	2.516.171.250	SOF	Frame#=104	
173	0	SOF PID=A5 FRAME#=104 CRC5=00		
164	5.000	SETUP DATA0 ACK	Frame#=104 ADDR=01 ENDP=0	フレーム(1ms)
174	0	SETUP PID=2D ADDR=01 ENDP=0 CRC5=1D ←セットアップトークンパケット		セットアップステージ
175	250	DATA0 PID=C3 DATA=8byte CRC16=F4E0 ←データパケット		
176	500	ACK PID=D2 ←ACKハンドシェイクパケット		
165	994.916	SOF	Frame#=105	
177	0	SOF PID=A5 FRAME#=105 CRC5=1F		
166	7.500	IN DATA1 ACK	Frame#=105 ADDR=01 ENDP=0	フレーム(1ms)
178	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット		データステージ
179	500	DATA1 PID=4B DATA=8byte CRC16=E757 ←データパケット		
180	500	ACK PID=D2 ←ACKハンドシェイクパケット		
167	992.500	SOF	Frame#=106	
181	0	SOF PID=A5 FRAME#=106 CRC5=17		
168	7.500	IN DATA0 ACK	Frame#=106 ADDR=01 ENDP=0	フレーム(1ms)
182	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット		データステージ
183	416	DATA0 PID=C3 DATA=8byte CRC16=EBED ←データパケット		
184	500	ACK PID=D2 ←ACKハンドシェイクパケット		
169	992.500	SOF	Frame#=107	
185	0	SOF PID=A5 FRAME#=107 CRC5=08		
170	7.500	IN DATA1 ACK	Frame#=107 ADDR=01 ENDP=0	フレーム(1ms)
186	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット		データステージ
187	416	DATA1 PID=4B DATA=2byte CRC16=7F3F ←データパケット		
188	666	ACK PID=D2 ←ACKハンドシェイクパケット		
171	992.416	SOF	Frame#=108	
189	0	SOF PID=A5 FRAME#=108 CRC5=09		
172	7.500	OUT DATA1 ACK	Frame#=108 ADDR=01 ENDP=0	フレーム(1ms)
190	0	OUT PID=E1 ADDR=01 ENDP=0 CRC5=1D ←アウトトークンパケット		ステータスステージ
191	250	DATA1 PID=4B DATA=0byte CRC16=0000 ←データパケット		
192	500	ACK PID=D2 ←ACKハンドシェイクパケット		
173	992.500	SOF	Frame#=109	
193	0	SOF PID=A5 FRAME#=109 CRC5=16		

データパケット

```
Packet No. = 175
000  80 06 00 01 00 00 12 00                          ........
Packet No. = 179
000  12 01 00 02 00 00 00 08                          ........
Packet No. = 183
000  5B 04 10 00 00 01 00 00                          [.......
Packet No. = 187
000  03 01                                            ..
```

←Get_Descriptor(Device)コマンド

←Device Descriptor 情報

コントロール転送(Get_Descriptor(Configuration))

No.	Time (nSec)	Transaction / Packet	Information	
174	2.522.171.083	SOF	Frame#=10A	
194	0	SOF PID=A5 FRAME#=10A CRC5=1E		
175	8.833	SETUP DATA0 ACK	Frame#=10A ADDR=01 ENDP=0	フレーム(1ms)
195	0	SETUP PID=2D ADDR=01 ENDP=0 CRC5=1D ←セットアップトークンパケット		セットアップステージ
196	250	DATA0 PID=C3 DATA=8byte CRC16=04AE ←データパケット		
197	500	ACK PID=D2 ←ACKハンドシェイクパケット		
176	991.083	SOF	Frame#=10B	
198	0	SOF PID=A5 FRAME#=10B CRC5=01		
177	7.500	IN DATA1 ACK	Frame#=10B ADDR=01 ENDP=0	フレーム(1ms)
199	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット		データステージ
200	500	DATA1 PID=4B DATA=8byte CRC16=B00A ←データパケット		
201	500	ACK PID=D2 ←ACKハンドシェイクパケット		
178	992.500	SOF	Frame#=10C	
202	0	SOF PID=A5 FRAME#=10C CRC5=0E		
179	7.500	IN DATA0 ACK	Frame#=10C ADDR=01 ENDP=0	フレーム(1ms)
203	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット		データステージ
204	416	DATA0 PID=C3 DATA=1byte CRC16=6AC1 ←データパケット		
205	500	ACK PID=D2 ←ACKハンドシェイクパケット		
180	992.500	SOF	Frame#=10D	
206	0	SOF PID=A5 FRAME#=10D CRC5=11		
181	7.500	OUT DATA1 ACK	Frame#=10D ADDR=01 ENDP=0	フレーム(1ms)
207	0	OUT PID=E1 ADDR=01 ENDP=0 CRC5=1D ←アウトトークンパケット		ステータスステージ
208	250	DATA1 PID=4B DATA=0byte CRC16=0000 ←データパケット		
209	416	ACK PID=D2 ←ACKハンドシェイクパケット		
182	992.416	SOF	Frame#=10E	
210	0	SOF PID=A5 FRAME#=10E CRC5=19		

データパケット

```
Packet No. = 196
000  80 06 00 02 00 00 09 00                          ........
Packet No. = 200
000  09 02 22 00 01 01 00 C0                          .."....
Packet No. = 204
000  32                                               2
```

←Get_Descriptor(Configuration)コマンド

←Configuration Descriptor 情報

コントロール転送(Get_Descriptor(String))

No.	Time (nSec)	Transaction / Packet	Information
183	2.527.170.916	SOF	Frame#=10F
211	0	SOF PID=A5 FRAME#=10F CRC5=06	
184	7.666	SETUP DATA0 ACK	Frame#=10F ADDR=01 ENDP=0
212	0	SETUP PID=2D ADDR=01 ENDP=0 CRC5=1D ←セットアップトークンパケット	
213	250	DATA0 PID=C3 DATA=8byte CRC16=64D4 ←データパケット	
214	500	ACK PID=D2 ←ACKハンドシェイクパケット	
185	992.333	SOF	Frame#=110
215	0	SOF PID=A5 FRAME#=110 CRC5=1B	
186	6.250	IN DATA1 ACK	Frame#=110 ADDR=01 ENDP=0
216	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
217	416	DATA1 PID=4B DATA=4byte CRC16=7809 ←データパケット	
218	583	ACK PID=D2 ←ACKハンドシェイクパケット	
187	993.666	SOF	Frame#=111
219	0	SOF PID=A5 FRAME#=111 CRC5=04	
188	1.000.000	SOF	Frame#=112
220	0	SOF PID=A5 FRAME#=112 CRC5=0C	
189	7.416	OUT DATA1 ACK	Frame#=112 ADDR=01 ENDP=0
221	0	OUT PID=E1 ADDR=01 ENDP=0 CRC5=1D ←アウトトークンパケット	
222	250	DATA1 PID=4B DATA=0byte CRC16=0000 ←データパケット	
223	500	ACK PID=D2 ←ACKハンドシェイクパケット	
190	992.583	SOF	Frame#=113
224	0	SOF PID=A5 FRAME#=113 CRC5=13	

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　フレーム(1ms)

データパケット

```
Packet No. = 213
000  80 06 00 03 00 00 FF 00                          ........
Packet No. = 217
000  04 03 09 04                                      ....
```

←Get_Descriptor(String)コマンド

←String Descriptor 情報

コントロール転送(Get_Descriptor(String))

No.	Time (nSec)	Transaction / Packet	Information
191	2.532.170.750	SOF	Frame#=114
225	0	SOF PID=A5 FRAME#=114 CRC5=1C	
192	4.416	SETUP DATA0 ACK	Frame#=114 ADDR=01 ENDP=0
226	0	SETUP PID=2D ADDR=01 ENDP=0 CRC5=1D ←セットアップトークンパケット	
227	250	DATA0 PID=C3 DATA=8byte CRC16=0A96 ←データパケット	
228	583	ACK PID=D2 ←ACKハンドシェイクパケット	
193	995.583	SOF	Frame#=115
229	0	SOF PID=A5 FRAME#=115 CRC5=03	
194	7.500	IN DATA1 ACK	Frame#=115 ADDR=01 ENDP=0
230	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
231	416	DATA1 PID=4B DATA=8byte CRC16=7713 ←データパケット	
232	500	ACK PID=D2 ←ACKハンドシェイクパケット	
195	992.416	SOF	Frame#=116
233	0	SOF PID=A5 FRAME#=116 CRC5=0B	
196	7.500	IN DATA0 ACK	Frame#=116 ADDR=01 ENDP=0
234	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
235	416	DATA0 PID=C3 DATA=8byte CRC16=10A2 ←データパケット	
236	583	ACK PID=D2 ←ACKハンドシェイクパケット	
197	992.500	SOF	Frame#=117
237	0	SOF PID=A5 FRAME#=117 CRC5=14	
198	7.500	IN DATA1 ACK	Frame#=117 ADDR=01 ENDP=0
238	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
239	500	DATA1 PID=4B DATA=8byte CRC16=10A2 ←データパケット	
240	500	ACK PID=D2 ←ACKハンドシェイクパケット	
199	992.500	SOF	Frame#=118
241	0	SOF PID=A5 FRAME#=118 CRC5=15	
200	5.916	IN DATA0 ACK	Frame#=118 ADDR=01 ENDP=0
242	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
243	416	DATA0 PID=C3 DATA=2byte CRC16=DFEB ←データパケット	
244	583	ACK PID=D2 ←ACKハンドシェイクパケット	
201	994.000	SOF	Frame#=119
245	0	SOF PID=A5 FRAME#=119 CRC5=0A	
202	1.000.000	SOF	Frame#=11A
246	0	SOF PID=A5 FRAME#=11A CRC5=02	
203	7.500	OUT DATA1 ACK	Frame#=11A ADDR=01 ENDP=0
247	0	OUT PID=E1 ADDR=01 ENDP=0 CRC5=1D ←アウトトークンパケット	
248	250	DATA1 PID=4B DATA=0byte CRC16=0000 ←データパケット	
249	500	ACK PID=D2 ←ACKハンドシェイクパケット	
204	992.500	SOF	Frame#=11B
250	0	SOF PID=A5 FRAME#=11B CRC5=1D	

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　フレーム(1ms)

データパケット

```
Packet No. = 227
000  80 06 03 03 09 04 FF 00                          ........
Packet No. = 231
000  1A 03 30 00 30 00 30 00                          ..0.0.0.
Packet No. = 235
000  30 00 30 00 30 00 30 00                          0.0.0.0.
Packet No. = 239
000  30 00 30 00 30 00 30 00                          0.0.0.0.
Packet No. = 243
000  31 00                                            1.
```

←Get_Descriptor(String)コマンド

←String Descriptor 情報

コントロール転送(Get_Descriptor(Configuration))

No.	Time (nSec)	Transaction / Packet			Information	
205	2.540.170.500	SOF			Frame#=11C	
251	0	SOF	PID=A5 FRAME#=11C CRC5=12			
206	7.666	SETUP	DATA0	ACK	Frame#=11C ADDR=01 ENDP=0	
252	0	SETUP	PID=2D ADDR=01 ENDP=0 CRC5=1D		←セットアップトークンパケット	セットアップステージ
253	250	DATA0	PID=C3 DATA=8byte CRC16=A4E9		←データパケット	
254	500	ACK	PID=D2		←ACKハンドシェイクパケット	
207	992.333	SOF			Frame#=11D	フレーム(1ms)
255	0	SOF	PID=A5 FRAME#=11D CRC5=0D			
208	7.583	IN	DATA1	ACK	Frame#=11D ADDR=01 ENDP=0	
256	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D		←イントークンパケット	データステージ
257	500	DATA1	PID=4B DATA=8byte CRC16=B00A		←データパケット	
258	500	ACK	PID=D2		←ACKハンドシェイクパケット	
209	992.416	SOF			Frame#=11E	フレーム(1ms)
259	0	SOF	PID=A5 FRAME#=11E CRC5=05			
210	5.750	IN	DATA0	ACK	Frame#=11E ADDR=01 ENDP=0	
260	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D		←イントークンパケット	データステージ
261	416	DATA0	PID=C3 DATA=8byte CRC16=4D35		←データパケット	
262	500	ACK	PID=D2		←ACKハンドシェイクパケット	
211	994.166	SOF			Frame#=11F	フレーム(1ms)
263	0	SOF	PID=A5 FRAME#=11F CRC5=1A			
212	5.916	IN	DATA1	ACK	Frame#=11F ADDR=01 ENDP=0	
264	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D		←イントークンパケット	データステージ
265	500	DATA1	PID=4B DATA=8byte CRC16=B312		←データパケット	
266	500	ACK	PID=D2		←ACKハンドシェイクパケット	
213	994.083	SOF			Frame#=120	フレーム(1ms)
267	0	SOF	PID=A5 FRAME#=120 CRC5=16			
214	7.583	IN	DATA0	ACK	Frame#=120 ADDR=01 ENDP=0	
268	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D		←イントークンパケット	データステージ
269	416	DATA0	PID=C3 DATA=8byte CRC16=210C		←データパケット	
270	500	ACK	PID=D2		←ACKハンドシェイクパケット	
215	992.416	SOF			Frame#=121	フレーム(1ms)
271	0	SOF	PID=A5 FRAME#=121 CRC5=09			
216	5.916	IN	DATA1	ACK	Frame#=121 ADDR=01 ENDP=0	
272	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D		←イントークンパケット	データステージ
273	416	DATA1	PID=4B DATA=2byte CRC16=89FF		←データパケット	
274	583	ACK	PID=D2		←ACKハンドシェイクパケット	
217	994.000	SOF			Frame#=122	フレーム(1ms)
275	0	SOF	PID=A5 FRAME#=122 CRC5=01			
218	1.000.000	SOF			Frame#=123	フレーム(1ms)
276	0	SOF	PID=A5 FRAME#=123 CRC5=1E			
219	7.500	OUT	DATA1	ACK	Frame#=123 ADDR=01 ENDP=0	
277	0	OUT	PID=E1 ADDR=01 ENDP=0 CRC5=1D		←アウトトークンパケット	ステータスステージ
278	250	DATA1	PID=4B DATA=0byte CRC16=0000		←データパケット	
279	416	ACK	PID=D2		←ACKハンドシェイクパケット	
220	992.416	SOF			Frame#=124	フレーム(1ms)
280	0	SOF	PID=A5 FRAME#=124 CRC5=11			

データパケット

```
Packet No. = 253
000  80 06 00 02 00 00 FF 00                          ........
Packet No. = 267
000  09 02 22 00 01 01 00 C0                          .."....
Packet No. = 261
000  32 09 04 00 00 01 03 01                          2.......
Packet No. = 265
000  02 00 09 21 00 01 00 01                          ...!....
Packet No. = 269
000  22 34 00 07 05 83 03 04                          "4......
Packet No. = 273
000  00 08                                            ..
```

←Get_Descriptor(Configuration)コマンド

←Configuration Descriptor 情報

コントロール転送(Get_Descriptor(Device_Qualifier))

No.	Time (nSec)	Transaction / Packet			Information	
221	2.549.170.250	SOF			Frame#=125	
281	0	SOF	PID=A5 FRAME#=125 CRC5=0E			
222	8.916	SETUP	DATA0	ACK	Frame#=125 ADDR=01 ENDP=0	
282	0	SETUP	PID=2D ADDR=01 ENDP=0 CRC5=1D		←セットアップトークンパケット	
283	250	DATA0	PID=C3 DATA=8byte CRC16=345F		←データパケット	
284	500	ACK	PID=D2		←ACKハンドシェイクパケット	
223	991.083	SOF			Frame#=126	
285	0	SOF	PID=A5 FRAME#=126 CRC5=06			
224	7.250	IN	STALL		Frame#=126 ADDR=01 ENDP=0	
286	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D		←イントークンパケット	
287	333	STALL	PID=1E		←STALLハンドシェイクパケット	
225	992.666	SOF			Frame#=127	
288	0	SOF	PID=A5 FRAME#=127 CRC5=19			

データパケット

```
Packet No. = 283
000  80 06 00 06 00 00 0A 00                          ........
```

←Get_Descriptor(Device Qualifier)コマンド

※この間ＳＯＦパケットのみが続きます。

コントロール転送(Get_Descriptor(Device))

No.	Time (nSec)	Transaction / Packet	Information
228	2.554.170.083	SOF	Frame#=12A
291	0	SOF PID=A5 FRAME#=12A CRC5=0F	
229	3.250	SETUP DATA0 ACK	Frame#=12A ADDR=01 ENDP=0
292	0	SETUP PID=2D ADDR=01 ENDP=0 CRC5=1D ←セットアップトークンパケット	
293	250	DATA0 PID=C3 DATA=8byte CRC16=F4E0 ←データパケット	
294	500	ACK PID=D2 ←ACKハンドシェイクパケット	
230	996.750	SOF	Frame#=12B
295	0	SOF PID=A5 FRAME#=12B CRC5=10	
231	3.250	IN DATA1 ACK	Frame#=12B ADDR=01 ENDP=0
296	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
297	500	DATA1 PID=4B DATA=8byte CRC16=E757 ←データパケット	
298	416	ACK PID=D2 ←ACKハンドシェイクパケット	
232	996.750	SOF	Frame#=12C
299	0	SOF PID=A5 FRAME#=12C CRC5=1F	
233	5.250	IN DATA0 ACK	Frame#=12C ADDR=01 ENDP=0
300	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
301	416	DATA0 PID=C3 DATA=8byte CRC16=EBED ←データパケット	
302	500	ACK PID=D2 ←ACKハンドシェイクパケット	
234	994.666	SOF	Frame#=12D
303	0	SOF PID=A5 FRAME#=12D CRC5=00	
235	3.250	IN DATA1 ACK	Frame#=12D ADDR=01 ENDP=0
304	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
305	500	DATA1 PID=4B DATA=2byte CRC16=7F3F ←データパケット	
306	666	ACK PID=D2 ←ACKハンドシェイクパケット	
236	996.750	SOF	Frame#=12E
307	0	SOF PID=A5 FRAME#=12E CRC5=08	
237	4.000	OUT DATA1 ACK	Frame#=12E ADDR=01 ENDP=0
308	0	OUT PID=E1 ADDR=01 ENDP=0 CRC5=1D ←アウトトークンパケット	
309	250	DATA1 PID=4B DATA=0byte CRC16=0000 ←データパケット	
310	416	ACK PID=D2 ←ACKハンドシェイクパケット	
238	996.000	SOF	Frame#=12F
311	0	SOF PID=A5 FRAME#=12F CRC5=17	

セットアップステージ　データステージ　データステージ　データステージ　ステータスステージ　フレーム(1ms)

データパケット

```
Packet No. = 293
000  80 06 00 01 00 00 12 00                          ........
Packet No. = 297
000  12 01 00 02 00 00 00 08                          ........
Packet No. = 301
000  5B 04 10 00 00 01 00 00                          [.......
Packet No. = 305
000  03 01                                            ..
```

←Get_Descriptor(Device)コマンド

←Device Descriptor 情報

コントロール転送(Get_Descriptor(Configuration))

No.	Time (nSec)	Transaction / Packet	Information
239	2.560.169.916	SOF	Frame#=130
312	0	SOF PID=A5 FRAME#=130 CRC5=0A	
240	9.000	SETUP DATA0 ACK	Frame#=130 ADDR=01 ENDP=0
313	0	SETUP PID=2D ADDR=01 ENDP=0 CRC5=1D ←セットアップトークンパケット	
314	250	DATA0 PID=C3 DATA=8byte CRC16=04AE ←データパケット	
315	500	ACK PID=D2 ←ACKハンドシェイクパケット	
241	991.000	SOF	Frame#=131
316	0	SOF PID=A5 FRAME#=131 CRC5=15	
242	7.500	IN DATA1 ACK	Frame#=131 ADDR=01 ENDP=0
317	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
318	416	DATA1 PID=4B DATA=8byte CRC16=B00A ←データパケット	
319	583	ACK PID=D2 ←ACKハンドシェイクパケット	
243	992.500	SOF	Frame#=132
320	0	SOF PID=A5 FRAME#=132 CRC5=1D	
244	7.583	IN DATA0 ACK	Frame#=132 ADDR=01 ENDP=0
321	0	IN PID=69 ADDR=01 ENDP=0 CRC5=1D ←イントークンパケット	
322	416	DATA0 PID=C3 DATA=1byte CRC16=6AC1 ←データパケット	
323	500	ACK PID=D2 ←ACKハンドシェイクパケット	
245	992.333	SOF	Frame#=133
324	0	SOF PID=A5 FRAME#=133 CRC5=02	
246	7.166	OUT DATA1 ACK	Frame#=133 ADDR=01 ENDP=0
325	0	OUT PID=E1 ADDR=01 ENDP=0 CRC5=1D ←アウトトークンパケット	
326	250	DATA1 PID=4B DATA=0byte CRC16=0000 ←データパケット	
327	500	ACK PID=D2 ←ACKハンドシェイクパケット	
247	992.833	SOF	Frame#=134
328	0	SOF PID=A5 FRAME#=134 CRC5=0D	

セットアップステージ　データステージ　データステージ　データステージ　ステータスステージ　フレーム(1ms)

データパケット

```
Packet No. = 314
000  80 06 00 02 00 00 09 00                          ........
Packet No. = 318
000  09 02 22 00 01 01 00 C0                          .."....
Packet No. = 322
000  32                                               2
```

←Get_Descriptor(Configuration)コマンド

←Configuration Descriptor 情報

コントロール転送(Get_Descriptor(Configuration))

No.	Time (nSec)	Transaction / Packet			Information	
248	2.565.169.750	SOF			Frame#=135	
329	0	SOF	PID=A5 FRAME#=135 CRC5=12			
249	8.833	SETUP	DATA0	ACK	Frame#=135 ADDR=01 ENDP=0	
330	0	SETUP	PID=2D ADDR=01 ENDP=0 CRC5=1D	←セットアップトークンパケット		
331	250	DATA0	PID=C3 DATA=8byte CRC16=F4B0	←データパケット		
332	500	ACK	PID=D2	←ACKハンドシェイクパケット		
250	991.166	SOF			Frame#=136	
333	0	SOF	PID=A5 FRAME#=136 CRC5=1A			
251	7.583	IN	DATA1	ACK	Frame#=136 ADDR=01 ENDP=0	
334	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D	←イントークンパケット		
335	416	DATA1	PID=4B DATA=8byte CRC16=B00A	←データパケット		
336	500	ACK	PID=D2	←ACKハンドシェイクパケット		
252	992.416	SOF			Frame#=137	
337	0	SOF	PID=A5 FRAME#=137 CRC5=05			
253	7.500	IN	DATA0	ACK	Frame#=137 ADDR=01 ENDP=0	
338	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D	←イントークンパケット		
339	416	DATA0	PID=C3 DATA=8byte CRC16=4D35	←データパケット		
340	583	ACK	PID=D2	←ACKハンドシェイクパケット		
254	992.416	SOF			Frame#=138	
341	0	SOF	PID=A5 FRAME#=138 CRC5=04			
255	7.500	IN	DATA1	ACK	Frame#=138 ADDR=01 ENDP=0	
342	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D	←イントークンパケット		
343	500	DATA1	PID=4B DATA=8byte CRC16=B312	←データパケット		
344	500	ACK	PID=D2	←ACKハンドシェイクパケット		
256	992.500	SOF			Frame#=139	
345	0	SOF	PID=A5 FRAME#=139 CRC5=1B			
257	5.916	IN	DATA0	ACK	Frame#=139 ADDR=01 ENDP=0	
346	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D	←イントークンパケット		
347	416	DATA0	PID=C3 DATA=8byte CRC16=210C	←データパケット		
348	500	ACK	PID=D2	←ACKハンドシェイクパケット		
258	994.083	SOF			Frame#=13A	
349	0	SOF	PID=A5 FRAME#=13A CRC5=13			
259	7.583	IN	DATA1	ACK	Frame#=13A ADDR=01 ENDP=0	
350	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D	←イントークンパケット		
351	416	DATA1	PID=4B DATA=2byte CRC16=89FF	←データパケット		
352	500	ACK	PID=D2	←ACKハンドシェイクパケット		
260	992.333	SOF			Frame#=13B	
353	0	SOF	PID=A5 FRAME#=13B CRC5=0C			
261	6.916	OUT	DATA1	ACK	Frame#=13B ADDR=01 ENDP=0	
354	0	OUT	PID=E1 ADDR=01 ENDP=0 CRC5=1D	←アウトトークンパケット		
355	250	DATA1	PID=4B DATA=0byte CRC16=0000	←データパケット		
356	500	ACK	PID=D2	←ACKハンドシェイクパケット		
262	993.083	SOF			Frame#=13C	
357	0	SOF	PID=A5 FRAME#=13C CRC5=03			

セットアップステージ
データステージ
データステージ
データステージ
データステージ
ステータスステージ
フレーム(1ms)

データパケット

```
Packet No. = 331
000  80 06 00 02 00 00 22 00                          ......".
Packet No. = 335
000  09 02 22 00 01 01 00 C0                          .."..... ←Configuration Descriptor 情報
Packet No. = 339
000  32 09 04 00 00 01 03 01                          2.......
Packet No. = 343
000  02 00 09 21 00 01 00 01                          ...!....
Packet No. = 347
000  22 34 00 07 05 83 03 04                          "4......
Packet No. = 351
000  00 08                                            ..
```

←Get_Descriptor(Configuration)コマンド

コントロール転送(Set_Configuration)

No.	Time (nSec)	Transaction / Packet			Information	
263	2.573.169.583	SOF			Frame#=13D	
358	0	SOF	PID=A5 FRAME#=13D CRC5=1C			
264	8.916	SETUP	DATA0	ACK	Frame#=13D ADDR=01 ENDP=0	
359	0	SETUP	PID=2D ADDR=01 ENDP=0 CRC5=1D	←セットアップトークンパケット		
360	250	DATA0	PID=C3 DATA=8byte CRC16=2527	←データパケット		
361	500	ACK	PID=D2	←ACKハンドシェイクパケット		
265	991.000	SOF			Frame#=13E	
362	0	SOF	PID=A5 FRAME#=13E CRC5=14			
266	7.500	IN	DATA1	ACK	Frame#=13E ADDR=01 ENDP=0	
363	0	IN	PID=69 ADDR=01 ENDP=0 CRC5=1D	←イントークンパケット		
364	500	DATA1	PID=4B DATA=0byte CRC16=0000	←データパケット		
365	583	ACK	PID=D2	←ACKハンドシェイクパケット		
267	992.500	SOF			Frame#=13F	
366	0	SOF	PID=A5 FRAME#=13F CRC5=0B			

セットアップステージ
ステータスステージ
フレーム(1ms)

データパケット

```
Packet No. = 360
000  00 09 01 00 00 00 00 00                          ........ ←Set Configuration 情報
```

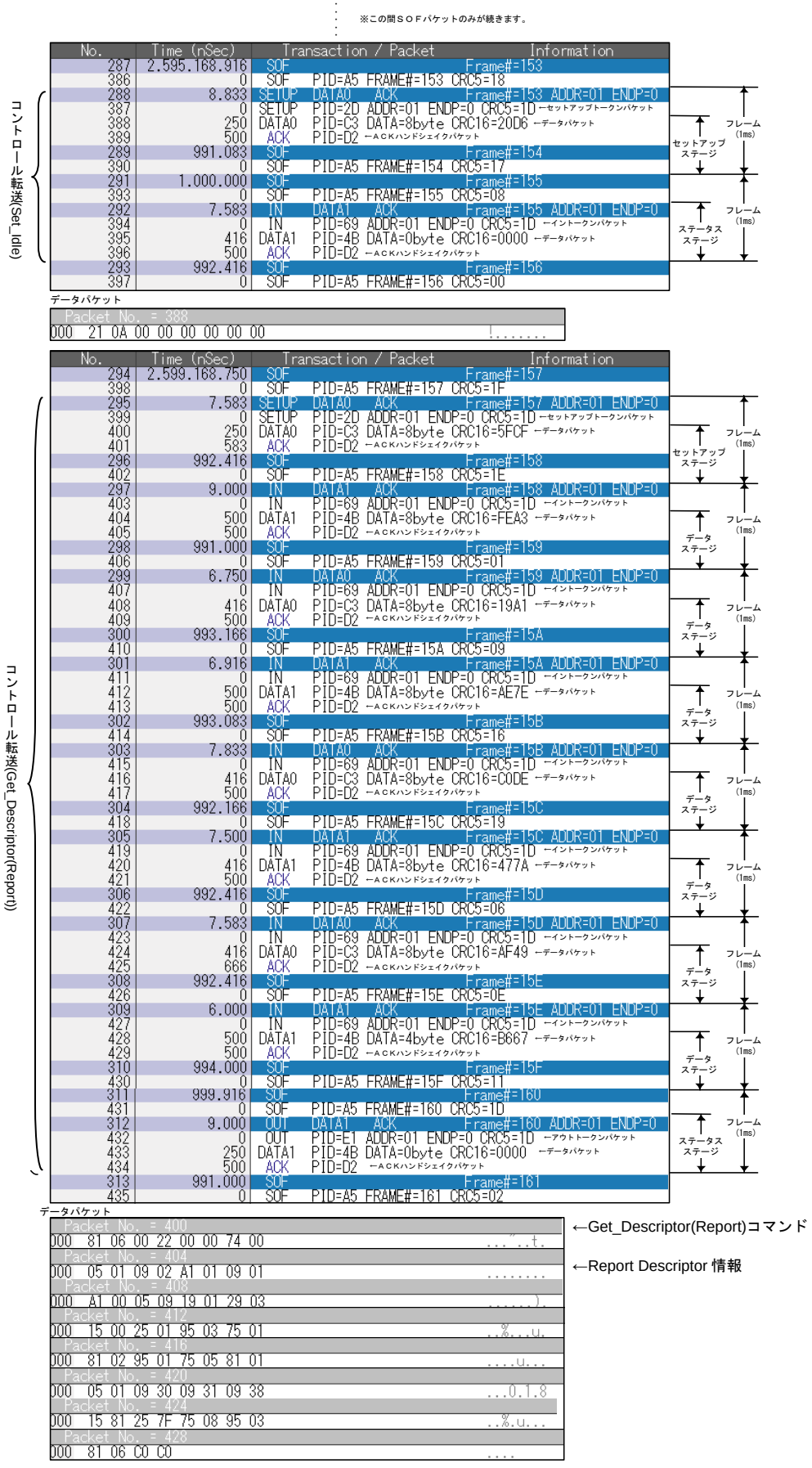


図 6.1　デバイス接続時のコントロール転送

6.2 HIDデータのインタラプトIN転送

以下に示す図 6.2 は、本デバイスから USB ホストに向けてインタラプト IN 転送で HID データの送信を行っている際の測定結果です。USB ホストが発行するインタラプト IN 転送に対し、ファンクションは送信可能なデータがない場合は NAK を送信します。送信可能なデータがある場合は 4 バイトの HID データを送信します。USB ホストは HID データを受信すると ACK を発行します。

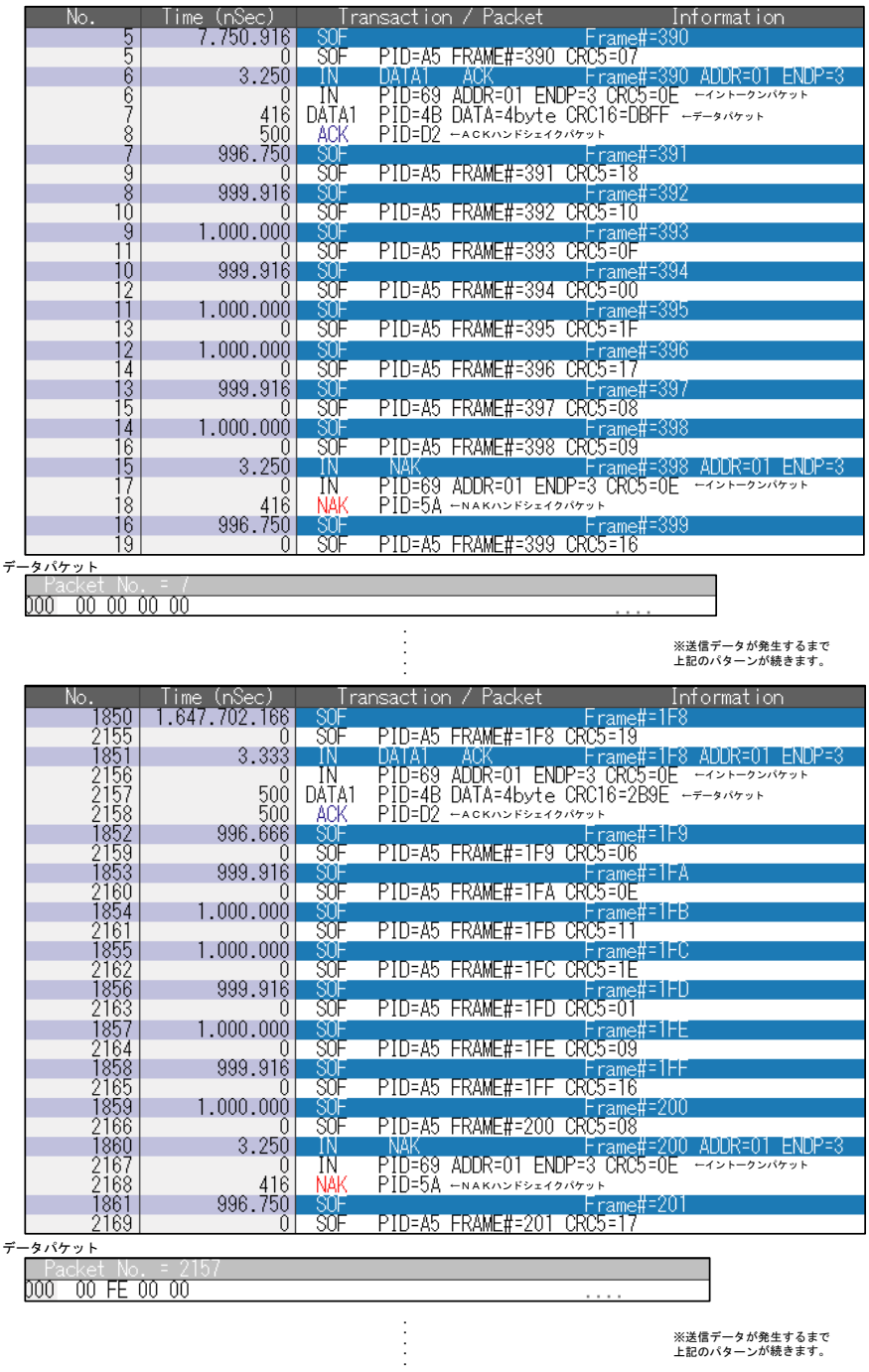


図 6.2　データ送信時のインタラプト IN 転送

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