

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ひ申し上げます。

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事務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、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。

お客様各位

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

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

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

2003年4月1日
株式会社ルネサス テクノロジー
カスタマサポート部

ご注意

安全設計に関するお願い

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

本資料ご利用に際しての留意事項

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

H8S/2215 USB 機能モジュール

USB シリアル変換アプリケーションノート
ルネサスシングルチップマイクロコンピュータ

H8S/2215

HD64F2215

ご注意

- 1 本書に記載の製品及び技術のうち「外国為替及び外国貿易法」に基づき安全保障貿易管理関連貨物・技術に該当するものを輸出する場合、または国外に持ち出す場合は日本国政府の許可が必要です。
- 2 本書に記載された情報の使用に際して、弊社もしくは第三者の特許権、著作権、商標権、その他の知的所有権等の権利に対する保証または実施権の許諾を行うものではありません。また本書に記載された情報を使用した事により第三者の知的所有権等の権利に関わる問題が生じた場合、弊社はその責を負いませんので予めご了承ください。
- 3 製品及び製品仕様は予告無く変更する場合がありますので、最終的な設計、ご購入、ご使用に際しましては、事前に最新の製品規格または仕様書をお求めになりご確認ください。
- 4 弊社は品質・信頼性の向上に努めておりますが、宇宙、航空、原子力、燃焼制御、運輸、交通、各種安全装置、ライフサポート関連の医療機器等のように、特別な品質・信頼性が要求され、その故障や誤動作が直接人命を脅かしたり、人体に危害を及ぼす恐れのある用途にご使用をお考えのお客様は、事前に弊社営業担当迄ご相談をお願い致します。
- 5 設計に際しては、特に最大定格、動作電源電圧範囲、放熱特性、実装条件及びその他諸条件につきましては、弊社保証範囲内でご使用いただきますようお願い致します。
保証値を越えてご使用された場合の故障及び事故につきましては、弊社はその責を負いません。また保証値内のご使用であっても半導体製品について通常予測される故障発生率、故障モードをご考慮の上、弊社製品の動作が原因でご使用機器が人身事故、火災事故、その他の拡大損害を生じないようにフェールセーフ等のシステム上の対策を講じて頂きますようお願い致します。
- 6 本製品は耐放射線設計をしておりません。
- 7 本書の一部または全部を弊社の文書による承認なしに転載または複製することを堅くお断り致します。
- 8 本書をはじめ弊社半導体についてのお問い合わせ、ご相談は弊社営業担当迄お願い致します。

はじめに

本アプリケーションノートは、H8S/2215 内蔵の USB ファンクションモジュールを用いた USB シリアル変換ファームウェアについて説明したものであり、お客様が USB ファンクションモジュールファームウェア作成の際に、御参考として役立てて頂ける様にまとめました。本アプリケーションノートの内容およびソフトウェアは、USB ファンクションモジュールの使用例を説明しているものであり、その内容を保証するものではありません。

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

【関連マニュアル】

- Universal Serial Bus Specification Revision 1.1
- H8S/2215シリーズハードウェアマニュアル
- H8S/2215 Solition Engine CPUボード (MS2215CP01-C/S) 取扱説明書
- H8S/2215シリーズTFP120用ユーザシステムインターフェイスケーブル (HS2215ECN61H) 取扱説明書
- E6000 (HS2214EPI61H) エミュレータユーザズマニュアル

【ご注意】

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

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

目次

1. 概要	1-1
1.1 概要	1-1
1.2 本システムの目的	1-3
2. 開発環境	2-1
2.1 ハードウェア環境	2-2
2.2 ソフトウェア環境	2-3
2.2.1 サンプルプログラム	2-3
2.2.2 コンパイルおよびリンク	2-3
2.2.3 USB シリアル変換ドライバ	2-4
2.3 プログラムのロードと実行方法	2-5
2.3.1 プログラムのロードと実行	2-5
2.4 PC間通信の方法	2-6
2.4.1 USB ホスト PC の設定	2-6
2.4.2 シリアル接続 PC の設定	2-11
2.4.3 PC 間通信	2-11
3. サンプルプログラム概要	3-1
3.1 状態遷移図	3-1
3.2 PC間通信概要	3-3
3.3 ファイル構成	3-4
3.4 関数の機能	3-4
4. サンプルプログラムの動作	4-1
4.1 メインループ	4-1
4.2 割り込みの種類	4-2
4.2.1 各転送への分岐方法	4-4
4.3 USB動作クロック安定検出割り込み	4-7
4.3.1 EPINFO	4-8
4.4 ケーブル接続時 (BRST、VBUS) 割り込み	4-10
4.5 コントロール転送	4-11
4.5.1 セットアップステージ	4-12
4.5.2 データステージ	4-14
4.5.3 ステータスステージ	4-16

4.6	バルク転送	4-18
4.6.1	バルクアウト転送	4-19
4.6.2	バルクイン転送	4-19
4.7	シリアル転送	4-21
4.7.1	シリアルアウト転送	4-21
4.7.2	シリアルイン転送	4-23
4.8	ベンダーコマンド	4-25
4.8.1	SetLineCoding	4-25
4.8.2	GetLineCoding	4-26
4.8.3	SetControlLineState	4-27
4.8.4	SendBreak	4-27
5.	アナライザのデータ	5-1
5.1	デバイス接続時のコントロール転送	5-1
5.2	ベンダコマンド転送時のコントロール転送	5-6

1. 概要

1.1 概要

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

H8S/2215内蔵USBファンクションモジュールの特長を以下に示します。

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

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

システム構成例を図 1.1 に示します。

1. 概要

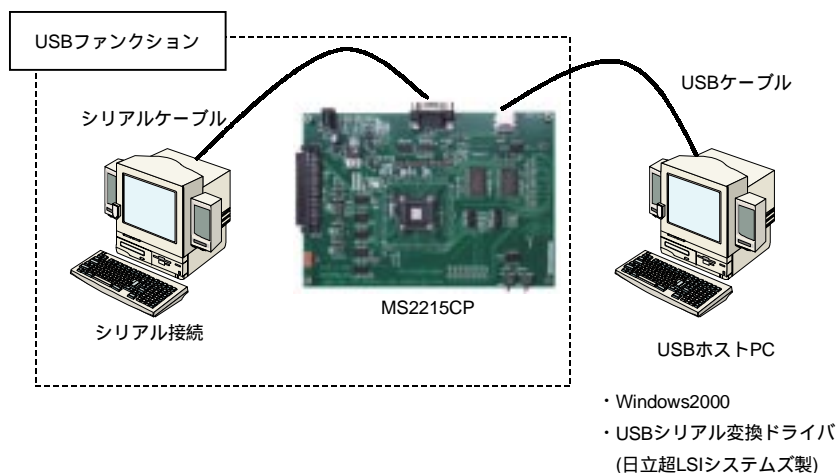


図 1.1 システム構成

本システムは、H8S/2215を搭載した日立超 LSI システムズ社製の H8S/2215 CPU ボード（以下、MS2215CP）、シリアル接続 PC、USB シリアル変換ドライバ*¹（日立超 LSI システムズ製）を搭載した USB ホスト PC（Windows2000）によって構成されています。

本システムは、USB ホスト PC から USB パケットにて送信されるデータを、MS2215CP によって受信しシリアルに変換後、シリアル接続 PC へ送信することができます。また、その逆の、シリアル接続 PC からのシリアルデータを MS2215CP によって受信し USB パケットに変換後、USB ホスト PC へ送信することができます。

本システムの特長を以下に示します。

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

【注】 1. システム（サンプルプログラム、USB シリアル変換ドライバ）に関するお問い合わせは、弊社営業担当までお願いします。

USB シリアル変換ドライバは、日立製作所のベンダ ID：045B 以外では動作しません。お客様が製品化される場合は、別途日立超 LSI システムズ社と USB シリアル変換ドライバに関し契約していただく必要があります。

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

1.2 本システムの目的

最近、レガシーフリーPC(シリアルポートなどの旧規格のポートがなく、USB(Universal Serial Bus)などのPlug & Playに対応した新しい規格のポートしかないPC)が、PCの低価格化が進むなか、数多く出荷され始めています。これに伴い、既存のシリアル機器がPCに接続できなくなり、数多く存在するシリアル機器が使用できなくなるおそれがあります。この問題を解決するためには、既存のシリアル回線をUSBに変換する装置が必要となってきます。本アプリケーションノートは、この問題を解決するためのUSBシリアル変換機能の実現例を提供することを目的としています。

本アプリケーションノートでは、既存のシリアル機器を新たにUSB機器に置き換えたとき、シリアルのAPIを用意することで、既存のシリアルアプリケーションから見てあたかもUSBが存在しないように見せかけます。これにより、アプリケーションプログラムに変更を加える事なく使用することができるようになります。

図1.2に、PCとシリアル機器を既存のシリアル回線で接続した場合のハードウェア構成およびソフトウェア構成を示します。また、図1.3にUSBシリアル変換装置を使用してPCとシリアル機器を接続した場合のハードウェア構成およびソフトウェア構成を示します。

図1.2(a)に示すように、既存のシステムでは、シリアル機器とPCをシリアルケーブルで接続します。これに対し、図1.3(a)に示すように既存シリアル機器をUSB経由でPCに接続する場合、PCとシリアル機器の間にUSBシリアル変換装置が必要となります。USBシリアル変換装置は、USB信号とシリアル信号を相互に変換する機能を持ちます。接続構成は、PCとUSBシリアル変換装置をUSBケーブルで接続し、USBシリアル変換装置とシリアル機器をシリアルケーブルで接続する構成となります。これにより、PCとシリアル機器が相互に通信可能となります。

図1.2(b)、図1.3(b)は、ソフトウェア構成を階層構造で表わしたものです。破線で示した接続は、論理的に接続されるイメージを表わします。

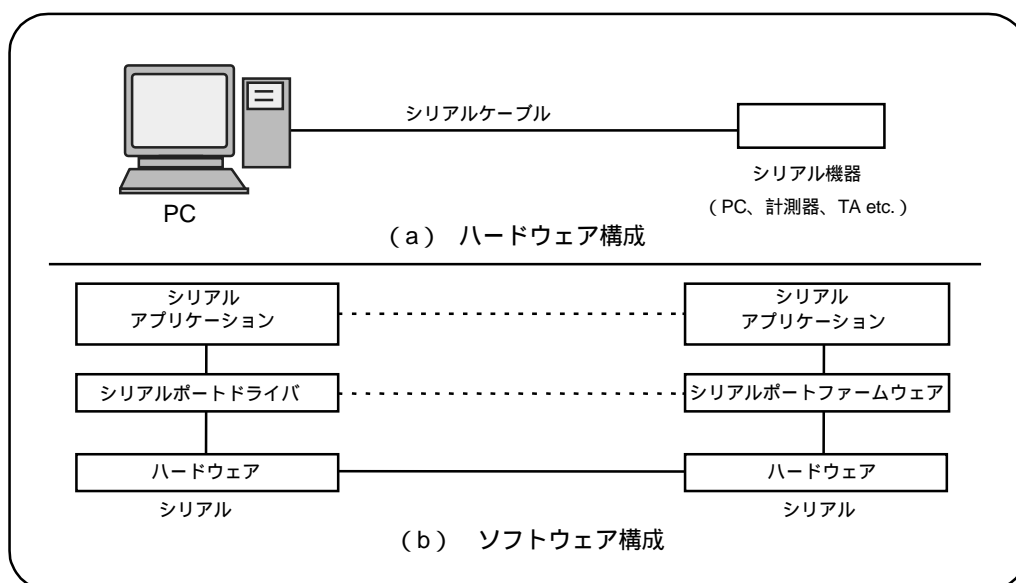


図 1.2 既存シリアル回線でのPCとシリアル機器接続構成例

1. 概要

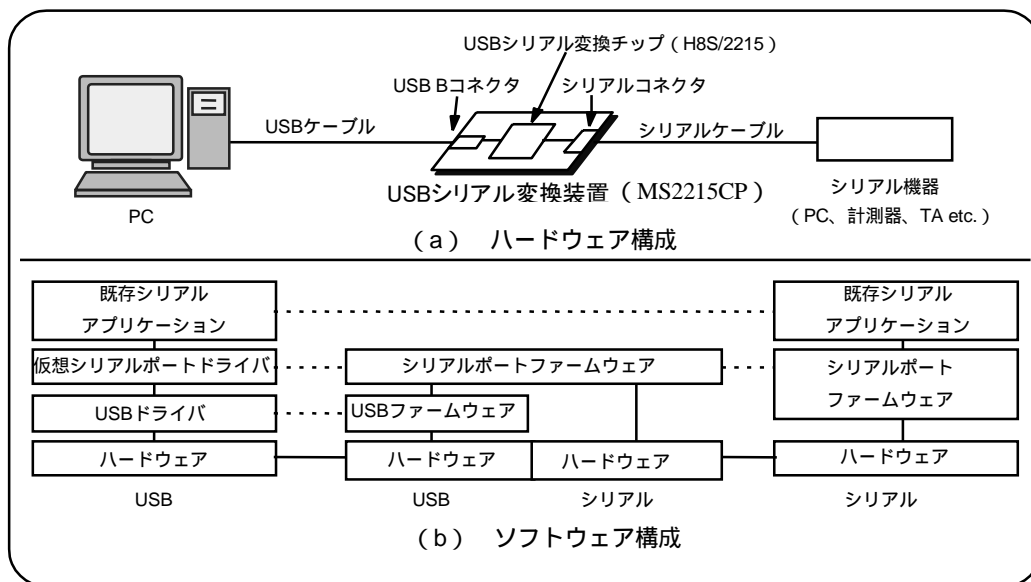


図 1.3 USB 経由での PC とシリアル機器接続構成例

図 1.2 (b) では、PC のシリアルアプリケーションからの送信データは、シリアルポートドライバに渡され、シリアルポートドライバは、そのデータを PC のシリアルハードウェアに送ります。シリアルハードウェアは、シリアル回線を通して、相手側のシリアルハードウェアにデータを送信します。シリアル機器のシリアルポートファームウェアは、データを受信したハードウェアからデータを取り出し、シリアルアプリケーションにデータを渡します。これにより、シリアルアプリケーション同士で、データのやり取りが可能となります。

一方、図 1.3 (b) では、PC のシリアルアプリケーションからの送信データは、仮想シリアルポートドライバに渡されます。この仮想シリアルポートドライバは、既存のシリアルポートドライバと同じアプリケーションインタフェースを持っています。そのため、既存のシリアルアプリケーションは、アプリケーションから見て、あたかも USB が存在しない様に見せかける事が可能となり、既存のシリアルアプリケーションに変更を加える事なくデータ通信が可能となります。仮想シリアルポートドライバは、アプリケーションからのデータを下位の USB ドライバに渡し、USB ドライバは、そのデータを PC の USB ハードウェアに送ります。USB ハードウェアは、USB バスを通して、USB シリアル変換装置の USB ハードウェアにデータを送信します。USB シリアル変換装置では、受信した USB データをシリアルデータに変換し、シリアル機器にデータを送信します。USB シリアル変換装置とシリアル機器間の通信は、図 1.2 と同じ形となります。これにより、既存のシリアルアプリケーション同士で、データのやり取りが可能となります。

本アプリケーションノートは、図 1.3 (b) の USB シリアル変換装置上のファームウェアに相当する、MS2215CP で動作するファームウェアの実現例を示します。

2. 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。本システムの開発には、以下のデバイス（ツール）を使用しました。

- H8S/2215 CPUボード（型名MS2215CP01-C/S）日立超LSIシステムズ社製
- H8S/2215シリーズTFP120用ユーザシステムインターフェイスケーブル
（以下、H8S/2215ユーザケーブル 型名HS2215ECN61H）日立製作所製
- E6000（型名HS2214EPI61H）エミュレータ日立製作所製
- ISA（またはPCI/PCMCIA）スロット搭載のPC（Windows95/98）
- USBホスト用PC（Windows2000）
- USBシリアル変換ドライバ 日立超LSIシステムズ社製
- シリアル接続PC
- USBケーブル
- シリアルケーブル（クロスケーブル）
- Hitachi Debugging Interface（以下HDI）日立製作所製
- Hitachi Embedded Workshop（以下HEW）日立製作所製

2.1 ハードウェア環境

図 2.1 に各デバイスの接続形態を示します。

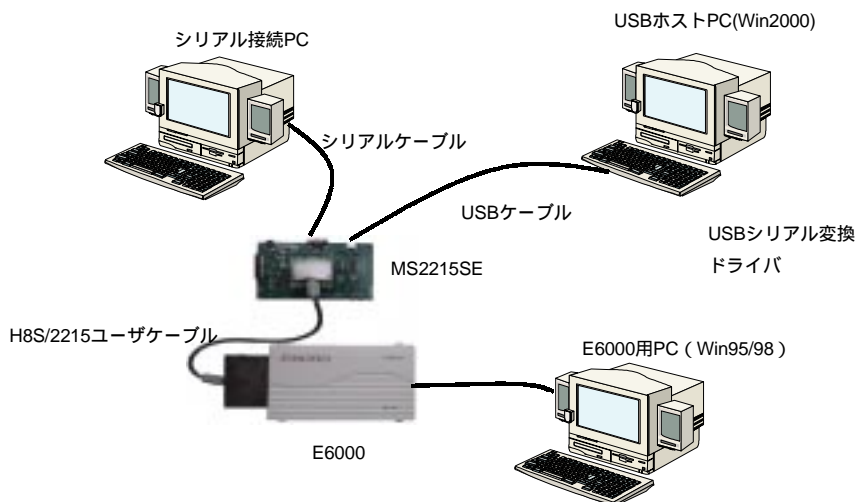


図 2.1 デバイスの接続形態

(1) MS2215CP

表 2.1 に示す MS2215CP ボードのジャンパピンを出荷時の設定から変更する必要があります。電源を投入する前に、このジャンパピンの設定をよくご確認ください。その他のジャンパピン、ディップスイッチを変更する必要はありません。

表 2.1 ディップスイッチの設定

出荷時	変更後	ディップスイッチの機能
J9 1-2 ショート	J9 2-3 ショート	PLLVCC 端子切り換え

(2) USB ホスト PC

USB ポート搭載の、Windows2000 をインストールしたパソコンを USB ホストとして使用します。このパソコンに USB シリアル変換ドライバ(日立超 LSI システムズ社製)をインストールします。

(3) シリアル接続 PC

シリアルポート搭載のパソコンをシリアルデータ送受信用のパソコンとして使用します。

(4) E6000 用 PC

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

2.2 ソフトウェア環境

サンプルプログラムと、今回使用したコンパイラおよびリンク、および、USB シリアル変換ドライバについて説明します。

2.2.1 サンプルプログラム

サンプルプログラムとして必要なファイルは、すべて「H8S2215」フォルダ内に収められています。HEW、HDI がインストールされたパソコンに、このフォルダごと移動して頂くと、すぐにサンプルプログラムを使用することができます。フォルダに含まれるファイルを以下図 2.2 に示します。

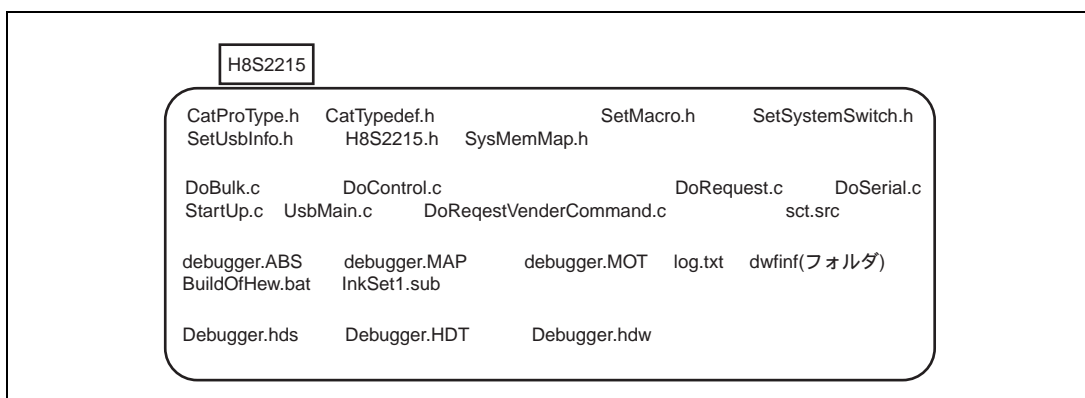


図 2.2 H8S2215 フォルダに含まれるファイル

2.2.2 コンパイルおよびリンク

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

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

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

まず、コンパイル時に作業用として、Tmp という名前のフォルダを C:\Hew のフォルダ内に作成してください。(図 2.3)

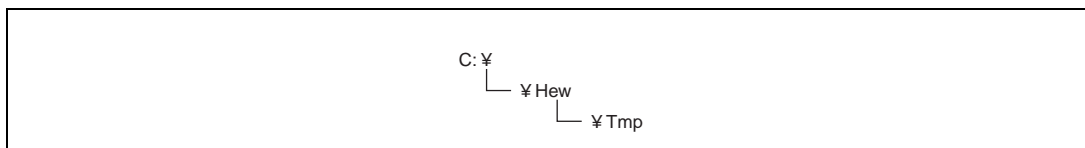


図 2.3 作業フォルダの作成

2. 開発環境

次に、サンプルプログラムが格納されているフォルダ（H8S2215）を、C:\Usr にコピーしてください（もしくは、任意の場所にコピーし、フォルダ内に含まれている debugger.hds ファイルに記載されている “C:\Usr\H8s2215” をフォルダをコピーしたパスに変更してください）。この中には、サンプルプログラムと共に “BuildOfHew.bat” というバッチファイルが含まれています。このバッチファイルでは、パスの設定、コンパイルオプションの指定、コンパイルおよびリンク結果を示すログファイルの指定等を行っています。BuildOfHew.bat を実行すると、コンパイルおよびリンクが行われます。その結果、フォルダ内にはファイル名 “debugger.MOT” のモトローラ S タイプフォーマットファイルが作成されます。これが実行ファイルとなります。このとき同時にマップファイル “debugger.MAP” とログファイル “log.txt” が作成されます。マップファイルにはプログラムのサイズ、および変数のアドレスが示されています。コンパイルの結果（エラーの有無等）はログファイルに記録されます。

【注】* HEW を “C:\HEW” 以外にインストールした場合、BuildOfHew.bat 内の「コンパイラパスの設定」と「コンパイラが使用する環境変数の設定」、lnkSet1.sub 内の「ライブラリーの指定」を変更する必要があります。この場合、コンパイラパスの設定は “ch38.exe” のパス、コンパイラが使用する環境変数 “ch38” の設定は “machine.h” のフォルダ、 “ch38_tmp” の設定はコンパイル作業フォルダをそれぞれ指定してください。また、ライブラリーの指定は “c8s26a.lib” のパスを指定してください。

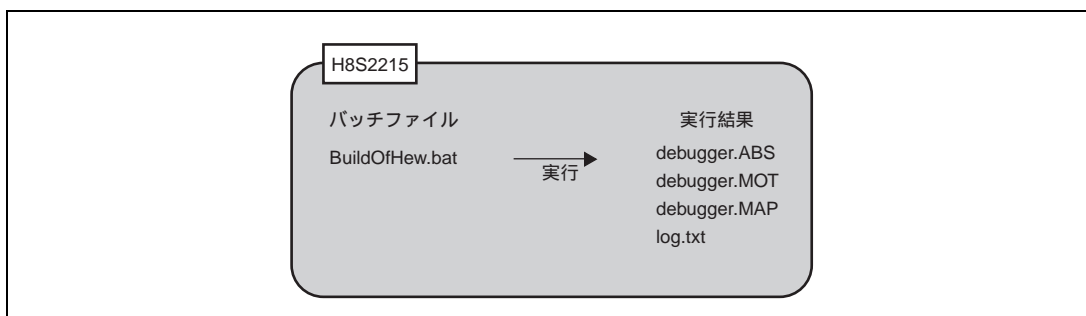


図 2.4 コンパイル結果

2.2.3 USB シリアル変換ドライバ

USB シリアル変換ドライバに関するファイルは、すべて “UST-03” フォルダ内に収められています。

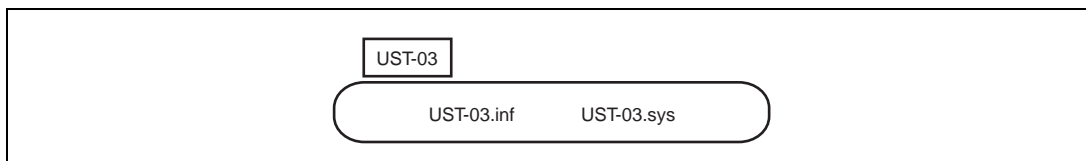


図 2.5 UST-03 フォルダに含まれるファイル

2.3 プログラムのロードと実行方法

図 2.6 にサンプルプログラムのメモリマップを示します。

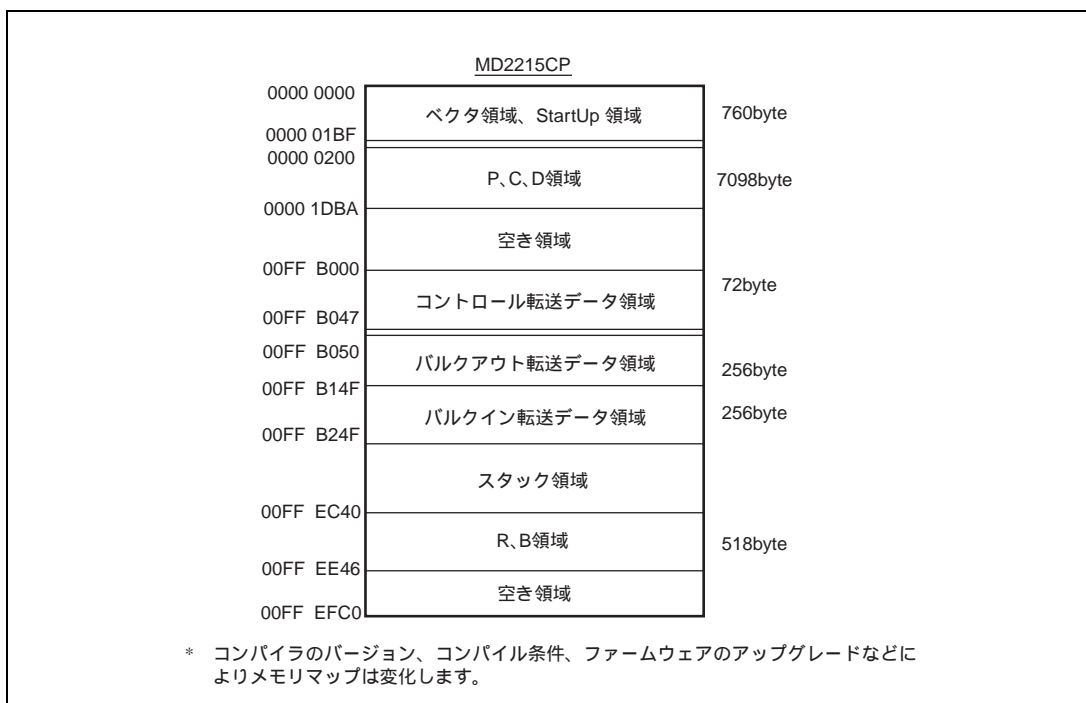


図 2.6 メモリマップ

図 2.6 のように、本サンプルプログラムは P、C、D を内蔵フラッシュメモリ領域に、R、B を内蔵 RAM 領域に配置しています。これらのメモリへの割り付けは、H8S2215 フォルダ内に含まれる “InkSet1.sub” で指定します。

2.3.1 プログラムのロードと実行

サンプルプログラムをロードするには、以下のような手順で行います。

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

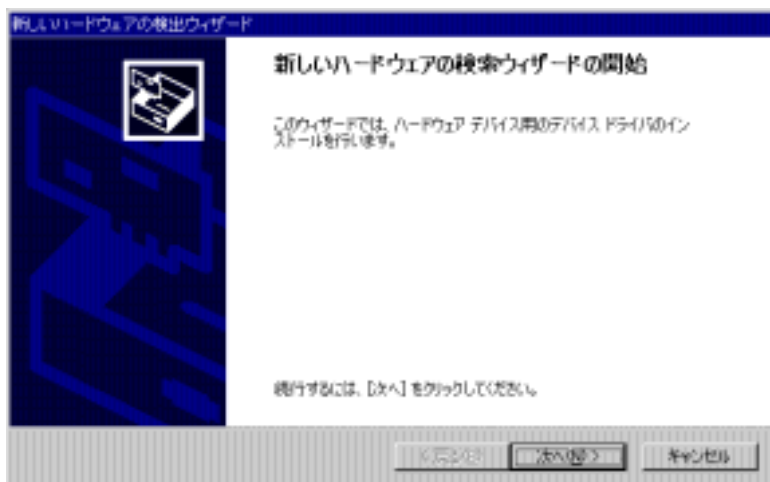
以上の操作で、サンプルプログラムを MS2215CP 上にロードする事ができます。

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

2.4 PC 間通信の方法

2.4.1 USB ホスト PC の設定

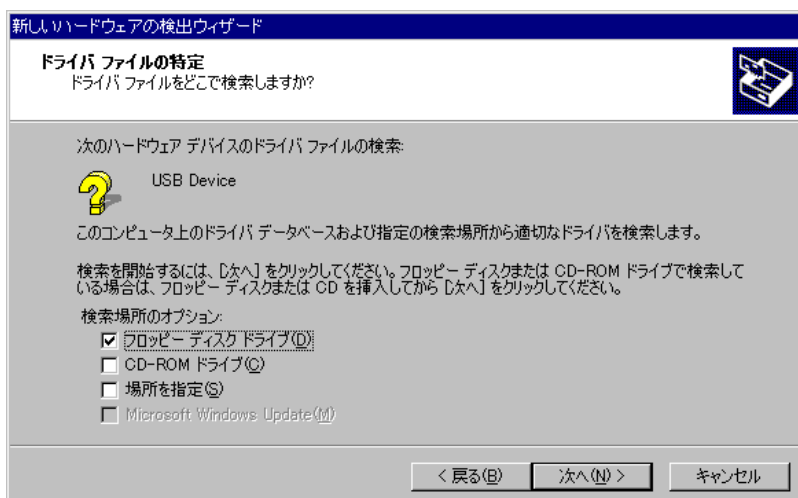
- 2.3.1、2.3.2に従い、サンプルプログラムを実行してください。サンプルプログラムが正常に起動すると、MS2215CP上の8ビットLEDが0xAAと表示されます。
- USBケーブルのシリーズBコネクタをMS2215CPに挿入し、反対側のシリーズAコネクタをUSBホストPCに接続してください。
- 画面に以下のダイアログが表示されるので、[次へ]をクリックします。



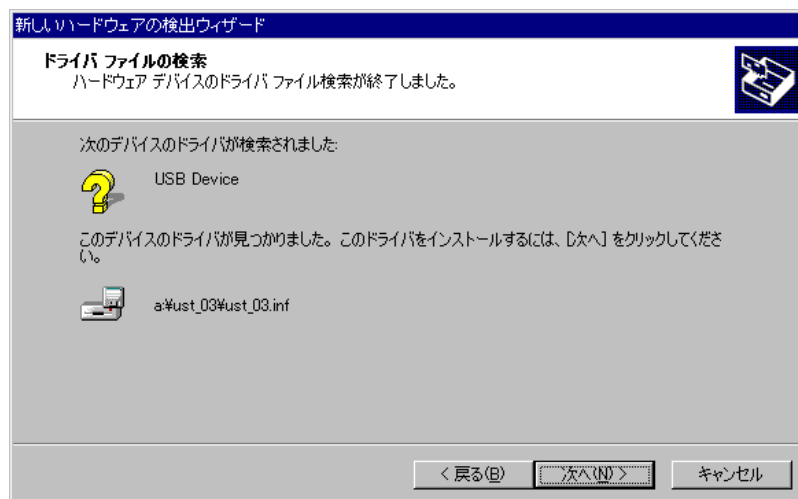
- 「デバイスに最適なドライバを検索する（推奨）」を選択し、[次へ]をクリックします。



- 「フロッピーディスクドライブ」を選択し、[次へ]をクリックします。



- UST-03.infを読み込むようになっていることを確認し、[次へ]をクリックします。



2. 開発環境

- [完了]をクリックします。



以上で、ドライバのインストール作業が終了し、USB ホスト PC は MS2215CP をシリアル COM ポートとして認識するようになります。

次に、WindowsOS 標準添付の通信ソフト、ハイパーターミナルを起動します。

- Windowsキーを押し、“スタート プログラム アクセサリ(もしくはさらに『通信』の下)”の順に選択し、ハイパーターミナルを起動します。
- 名前を入力し(任意、画面では、USB-Serialと入力)、「OK」を押します。



- 接続方法を「COM3」を選択し、「OK」をクリックします。



- シリアルポートの設定を行います。表2.2に示す値の範囲で所望の設定を行います。下図は、本プログラムのデフォルト値を入力した場合の例です。入力後、「OK」をクリックします。

これで、ハイパーターミナルが起動します。表 2.2 に示す設定可能値以外の設定を行うと、MS2215CP 上の 8 ビット LED が 0x30 を表示して、表 2.2 に示す本プログラムのデフォルト値を設定します。設定可能値を設定すると、8 ビット LED は 0xAA を表示し続けます。

表 2.2 シリアルポートの設定可能値

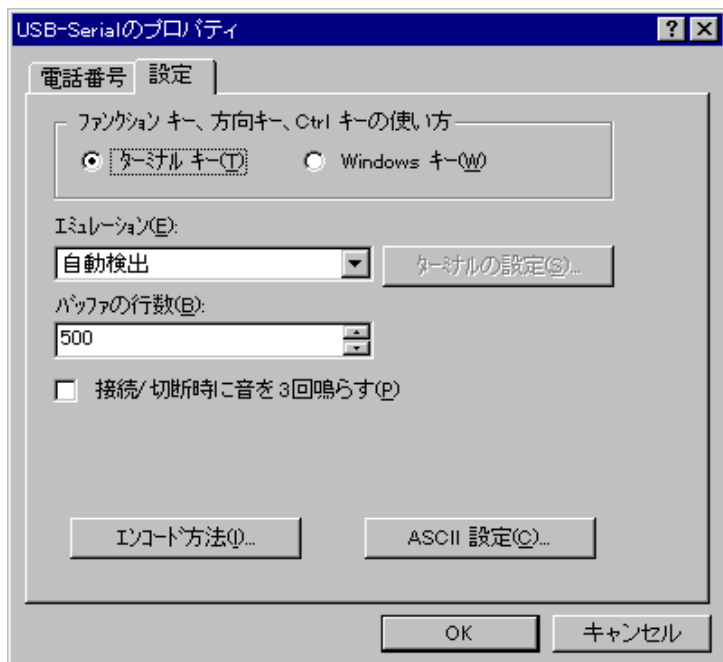
	本プログラムのデフォルト設定	設定可能値
ビット / 秒[bps]	38400	9600、19200、38400*
データビット	8	8、7
パリティ	なし	なし、奇数、偶数
ストップビット	1	1、2
フロー制御	Xon/Xoff	Xon/Xoff のみ

【注】 * 本サンプルプログラムでは CPU を 16MHz で動作させているため、57600bps、115200bps では誤差が大きくなり、正常動作しないおそれがあります。本サンプルプログラムでは 57600bps、115200bps は設定可能ですが、動作保障していません。

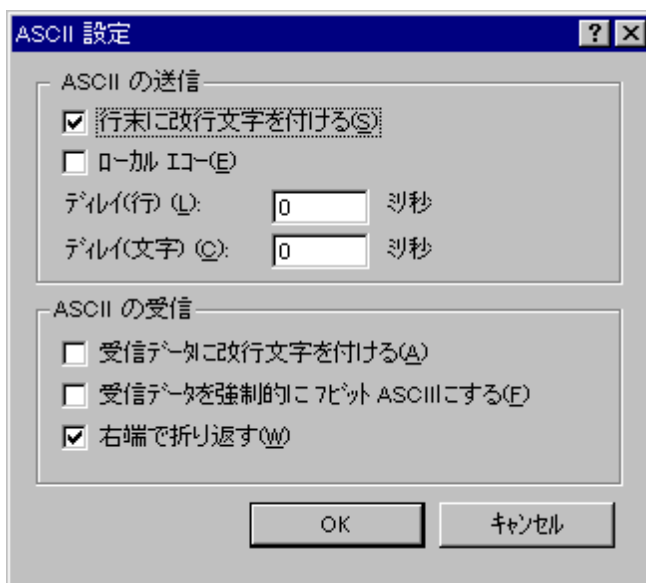
2. 開発環境



- ハイパーターミナル起動後、通信を始める前に、「ファイルメニュー プロパティ 設定」と順に選択して、「ASCII設定」をクリックします。



- ASCII送信欄の「行末に改行文字を付ける」にチェックを入れて、「OK」をクリックします。



2.4.2 シリアル接続 PC の設定

USB ホスト PC と同様にハイパーターミナルを起動します。シリアル通信の設定（ビット / 秒、データビット、パリティ、ストップビット、フロー制御）は必ず USB ホスト PC と同じ設定にしてください。

2.4.3 PC 間通信

USB ホスト PC とシリアル接続 PC の両者のハイパーターミナルが起動すると、この両者間で、キー入力文字の転送、テキストファイルの転送、および、バイナリファイルの転送が行えます。

USB ホスト PC 側でキー入力することで、その入力文字がシリアル接続 PC へ転送されます。また、その逆で、シリアル接続 PC 側でキー入力することで、その文字を USB ホスト PC へと転送されます。

「転送 テキストファイルの転送」を選択することで、テキストファイルを相手側 PC へ送信することができます。

受信側 PC で「転送 ファイルの受信 ZMODEM」を選択して受信側 PC を受信待ち受け状態にしてから、送信側 PC で「転送 ファイルの送信 ZMODEM」と選択します。これによって、テキストファイル、バイナリファイルを受信側 PC へと送信できます。

【注】 本アプリケーションノートでは、PC 上で動作するシリアルアプリケーションとして、ハイパーターミナルを使用しております。そのため、その他のシリアルアプリケーションでは、別途動作確認が必要となります。

本サンプルプログラムでは、ソフトフロー制御 (X-ON/X-OFF) を行っております。そのため、ファイルの送信は、ZMODEM などのソフトフロー制御 (X-ON/X-OFF) 対応プロトコルを必ず選択してください。

3. サンプルプログラム概要

この章ではサンプルプログラムの特長やその構成について説明します。本サンプルプログラムはMS2215CP上で動作し、USBファンクションモジュールからの割り込みもしくはメインループからの分岐によってUSB転送を開始します。また、SCIIの割り込みもしくはメインループからの分岐によってシリアル転送を開始します。H8S/2215内蔵モジュールの割り込みのうち、USBファンクションモジュールに関する割り込みは、EXIRQ0、EXIRQ1、IRQ6の3種類ですが、本サンプルプログラムではEXIRQ0のみを使用します。また、SCIIモジュールに関する割り込みは、ERII（受信エラー）、RXII（受信データフル）、TXII（送信データエンプティ）、TEII（送信終了）の4種類のうち、ERII、RXIIの2種類を使用します。

本サンプルプログラムの特長を以下に示します。

- コントロール転送を行う事ができます。
- バルクアウト転送でホストコントローラからデータを受信する事ができます。
- バルクイン転送でホストコントローラにデータを送信する事ができます。
- シリアル接続PCからシリアルデータを受信する事ができます。
- シリアル接続PCへシリアルデータを送信する事ができます。
- バルクアウト転送で受信したデータをシリアル送信する事ができます。
- シリアル受信したデータをバルクイン転送する事ができます。

3.1 状態遷移図

図3.1に、本サンプルプログラムの状態遷移図を示します。本サンプルプログラムは、図3.1のように4つの状態に遷移します。

1. リセット状態

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

2. 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。ここでは、USBホストPC、シリアル接続PCからのデータの有無を常に監視し、データがあればそれぞれ相手側PCへデータを出力します。つまり、MS2215CPへの入力データを常に監視し、入力データがあればそれぞれの相手側PCへデータ出力します。

3. USB通信状態

定常状態において、USBモジュールから割り込みが発生するとこの状態になります。USB通信状態では、割り込みの種類に応じた転送方式によるデータ転送を行います。本サンプルプログラムで使用する割り込み要因は“割り込みフラグレジスタ0~3 (UIFR0~3)”によって示され、計5種類です。割り込み要因が発生すると、

3. サンプルプログラム概要

UIFR0~3の対応するビットに1がセットされます。

4. シリアル通信状態

定常状態において、SCI1モジュールから割り込みが発生するとこの状態になります。本サンプルプログラムで使用する割り込み要因は、シリアルステータスレジスタ (SSR1) によって示され、ERI1、RXI1の計2種類です。

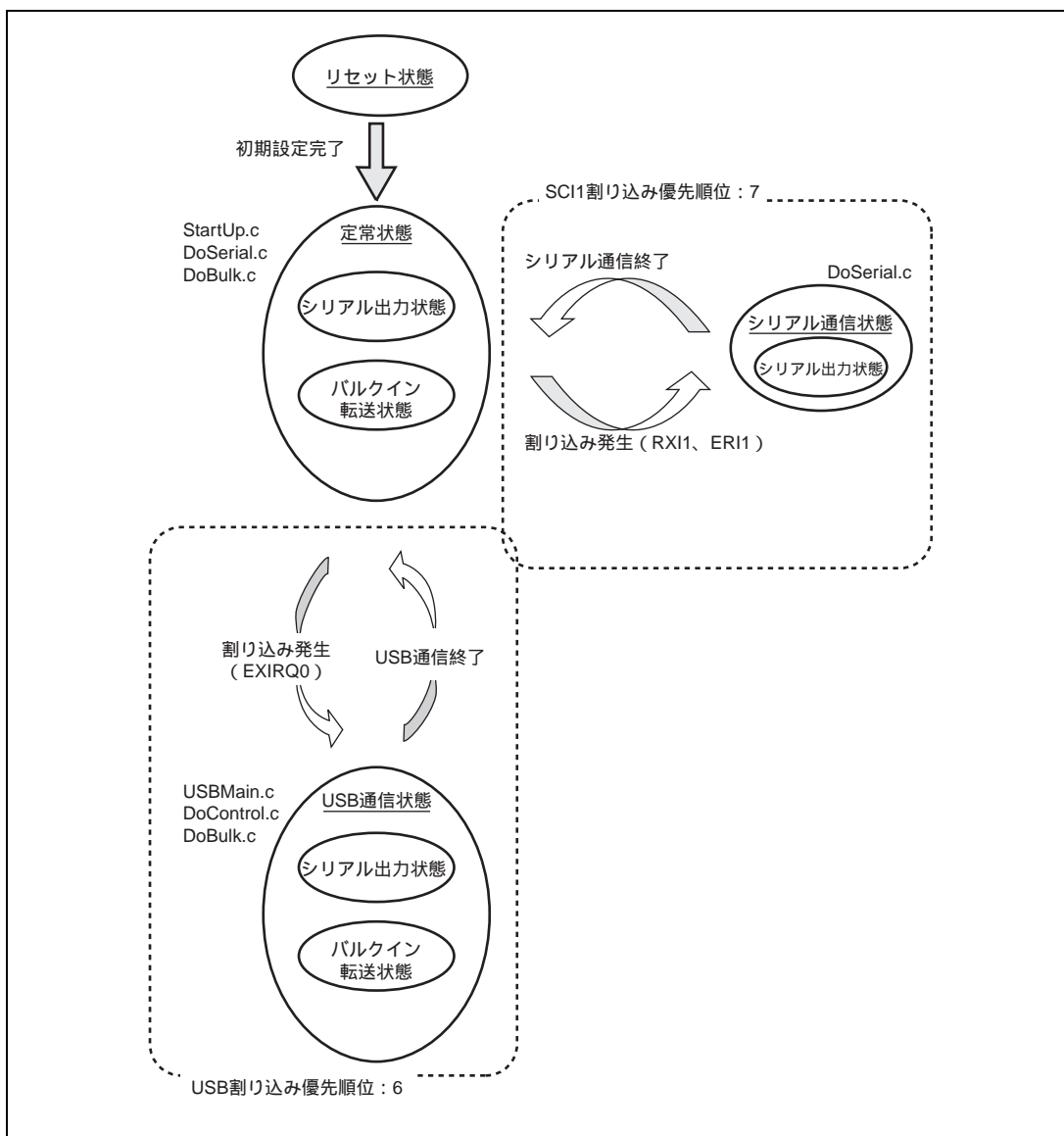


図 3.1 状態遷移図

本サンプルプログラムでは、USB の割り込み優先順位を 6、SCII の割り込み優先順位を 7 に設定します。この設定により、SCII 割り込み処理中は USB 割り込みを受け付けることができなく、USB 割り込みによってシリアル受信処理が待たされることを防止します。

3.2 PC 間通信概要

図 3.2 に PC 間通信の概要を示します。本サンプルプログラムでは、通信形態は大きく分けて USB 通信とシリアル通信の二つあります。データの送受信を考えると、USB 通信はバルクイン転送、バルクアウト転送のさらに二つの通信形態に分ける事ができ、シリアル通信の方も同様にシリアル入力、シリアル出力の 2 つに分ける事ができるので、本サンプルプログラムでは計 4 つの通信形態があることになります。

本サンプルプログラムでは、データの流れを、『バルクアウト転送 シリアル出力』系と『シリアル入力 バルクイン転送』系の二つに分け、それぞれに 256Byte のバッファを持たせてあります。それぞれの系のバッファへの入力系は割り込み処理をし、バッファからの出力系はメインループからの分岐で行うという制御を行います。メインループでは、常に両者のバッファであるバルクイン / バルクアウト転送用 RAM 領域を監視し、データが存在していればそのデータをバッファから出力します。

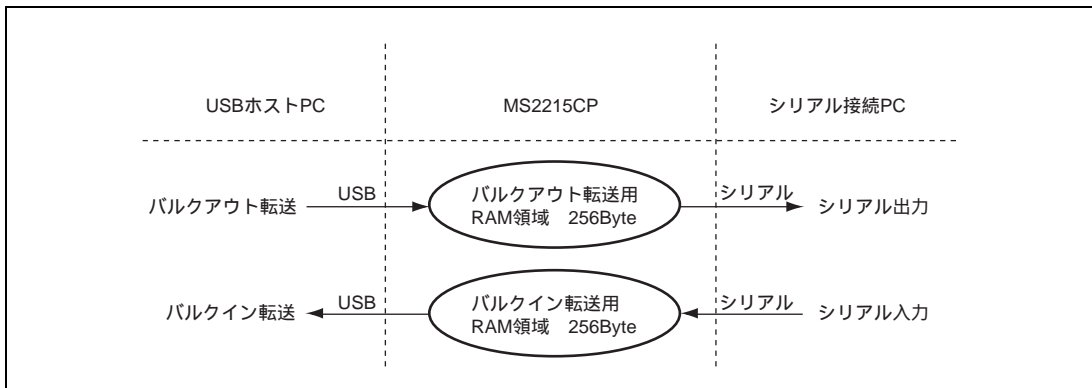


図 3.2 PC 間通信概要

3. サンプルプログラム概要

3.3 ファイル構成

本サンプルプログラムは、7個のソースファイルと7個のヘッダーファイルで構成されています。全構成ファイルを表 3.1 に示します。各関数は、転送方式または機能毎に1つのファイルにまとめてあります。

表 3.1 ファイル構成

ファイル名	主な役割
StartUp.c	ベクタテーブルの設定、マイコンの初期設定、リングバッファのクリア
DoSerial.c	シリアル送受信を実行、SCI1 モジュールの制御
UsbMain.c	割り込み要因の判定、パケットの送受信
DoRequest.c	ホストが発行するセットアップコマンドの処理
DoControl.c	コントロール転送を実行
DoBulk.c	バルク転送を実行
DoRequestVenderCommand.c	ベンダコマンドの処理
SysMemMap.h	MS2215CP のメモリマップのアドレス定義
SetUsbInfo.h	USB の構成を定義
SetMacro.h	マクロ定義
SetSystemSwitch.h	システムの動作設定
H8S2215.h	H8S/2215 レジスタ定義
CatTypedef.h	構造体定義
CatProType.h	プロトタイプ宣言

3.4 関数の機能

表 3.2 に、各ファイルに含まれる関数と、その機能を示します。

表 3.2-1 UsbMain.c

格納ファイル	関数名	機能
UsbMain.c	BranchOfInt	割り込み要因の判定と、割り込みに応じた関数の呼び出し
	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
	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
	SetControlOutContents	ホストから送られたデータを書き換える
	BE2ByteRead	2Byte のデータを、ビッグエンディアンに変換する
	LE2ByteRead	2Byte のデータを、リトルエンディアンに変換する
	ActBusReset	バスリセット受信時にバッファとフラグ、FIFO のクリアを行う
	SetUsbModule	USB モジュールの初期設定
	USBclear	リングバッファ、フラグのクリアを行う

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 3.2-2 StartUp.c

格納ファイル	関数名	機能
StartUp.c	SetPowerOnSection	BSC の設定、モジュール及びメモリの初期化を行い、メインループへ移行
	_INIT_SCT	初期値がある変数を、RAM のワークエリアにコピー
	InitMemory	バルク通信で使用する RAM 領域をクリア
	InitSystem	USB バスのプルアップ制御
	Error	エラー発生時、CPU をスリープモードに移行する
	Scilnit	SCI1 の初期化
	Set_SMR	SCI1 の SMR1 レジスタの初期設定を行う
	ActBusVcc	VBUS 受信時の処理

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

表 3.2-3 DoSerial.c

格納ファイル	関数名	機能
DoSerial.c	ActSerialOut	リードポインタからデータを読み出し、1 バイトづつ ExSerialOut に引数として渡す
	ActSerialIn	シリアル入力したデータをバルクイン転送用領域に書き込む
	WriteBulkInArea	データをバルクイン転送用領域に書き込む
	ExSerialOut	1 バイトデータを SCI1 からシリアル出力する

DoSerial.c では、シリアル送受信を行います。SCI1 モジュールの制御も行います。

表 3.2-4 DoRequest.c

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

コントロール転送時に、ホストコントローラから送られてくるコマンドをデコードし、コマンドに応じた処理を行います。本サンプルプログラムでは、ベンダ ID の値に"045B"（ベンダ：日立）を使用しています。お客様にて製品を開発される場合は「USB Implementers Forum」にてお客様のベンダ ID を取得願います。

3. サンプルプログラム概要

表 3.2-5 DoControl.c

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

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

表 3.2-6 DoBulk.c

格納ファイル	関数名	機 能
DoBulk.c	ActBulkOut	バルクアウト転送を制御する
	ActBulkIn	バルクイン転送を制御する
	ExBulkOut	GetPacket を実行する
	ExBulkIn	PutPacket を実行する

バルク転送に関する処理を行います。データ送受信および、フローの制御を行います。

表 3.2-7 DoRequestVenderCommand.c

格納ファイル	関数名	機 能
DoRequestVenderCommand.c	DecVenderCommands	ベンダコマンドの対応を行う

ベンダ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、日立超 LSI システムズ社製の USB シリアル変換ドライバがサポートする 4 つのベンダコマンドに対する処理を行います。詳細は「4.8 ベンダコマンド」を参照ください。

図 3.3 に、表 3.2 で説明した関数の相関関係を示します。上位側の関数が、下位側の関数を呼び出しています。また、複数の関数が同一の関数を呼び出す事もあります。定常状態では、SetPowerOnSection が他の関数を呼び出し、USB 割り込みの発生によって遷移する USB 通信状態では、BranchOfInt が他の関数を呼び出します。さらに、SC11 割り込みでは、ActSerialIn 関数を呼び出します。図 3.3 は、関数の上下関係を示しているもので、関数が呼び出される順序は示していません。関数がどのような順序で呼び出されるかについては、「第 4 章 サンプルプログラムの動作」のフローチャートをご覧ください。

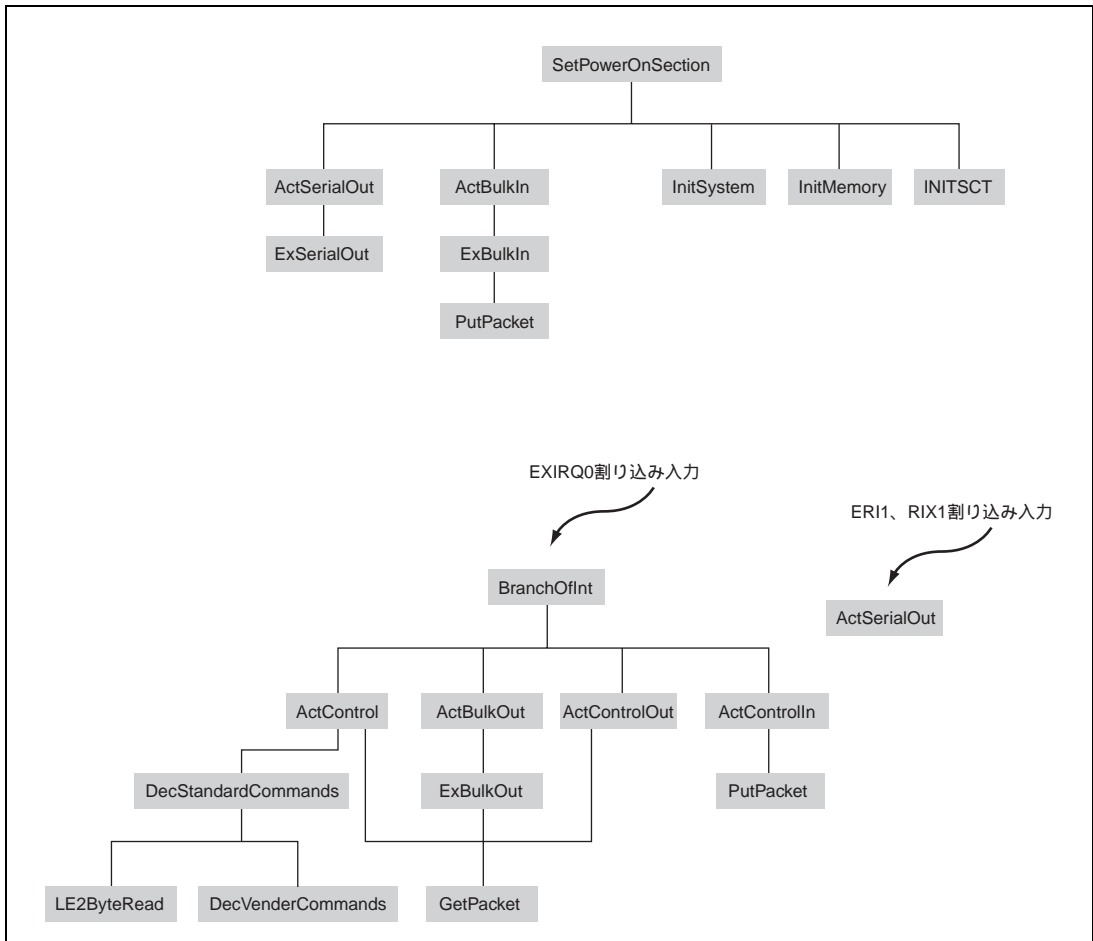


図 3.3 関数の相関関係

3. サンプルプログラム概要

4. サンプルプログラムの動作

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

4.1 メインループ

マイコンがリセット状態になると、CPU の内部状態と内蔵周辺モジュールのレジスタが初期化されます。次に StartUp.c の関数 SetPowerOnSection が呼び出され、CPU の初期化をします。図 4.1 に SetPowerOnSection のフローチャートを示します。

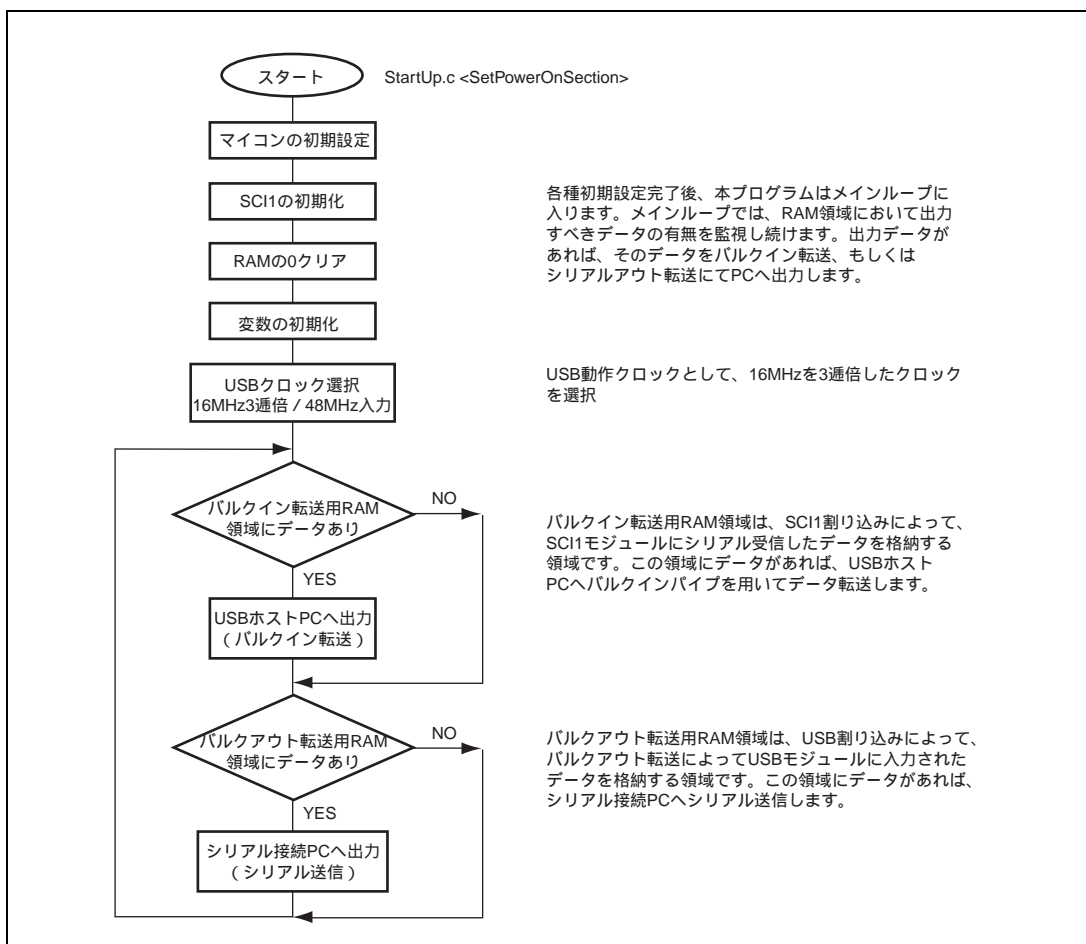


図 4.1 メインループ

4.2 割り込みの種類

「3.1 状態遷移図」で説明したように、本サンプルプログラムで使用する割り込み要因は"割り込みフラグレジスタ0~3 (UIFR0~3)"、"シリアルステータスレジスタ (SSR1)"によって示され、USB系が5種類、シリアル系が2種類です。

USB割り込み要因が発生すると、割り込みフラグレジスタの対応するビットに1がセットされ、CPUに対してEXIRQ0割り込みを要求します。サンプルプログラムでは、この割り込みによって割り込みフラグレジスタをリードし、それに対応するUSB通信を行います。図4.2に割り込みフラグレジスタと、USB通信の関係を示します。

なお、バルクイン転送は、本サンプルプログラムでサポートしていますが、割り込み動作ではなく、メインルーチンからの分岐によって動作します。そのため、バルクイン割り込みはディゼーブルに設定し、EP2i EMPTY フラグを見ることでバルクイン転送を起動します。EP2i TR は使用しません。

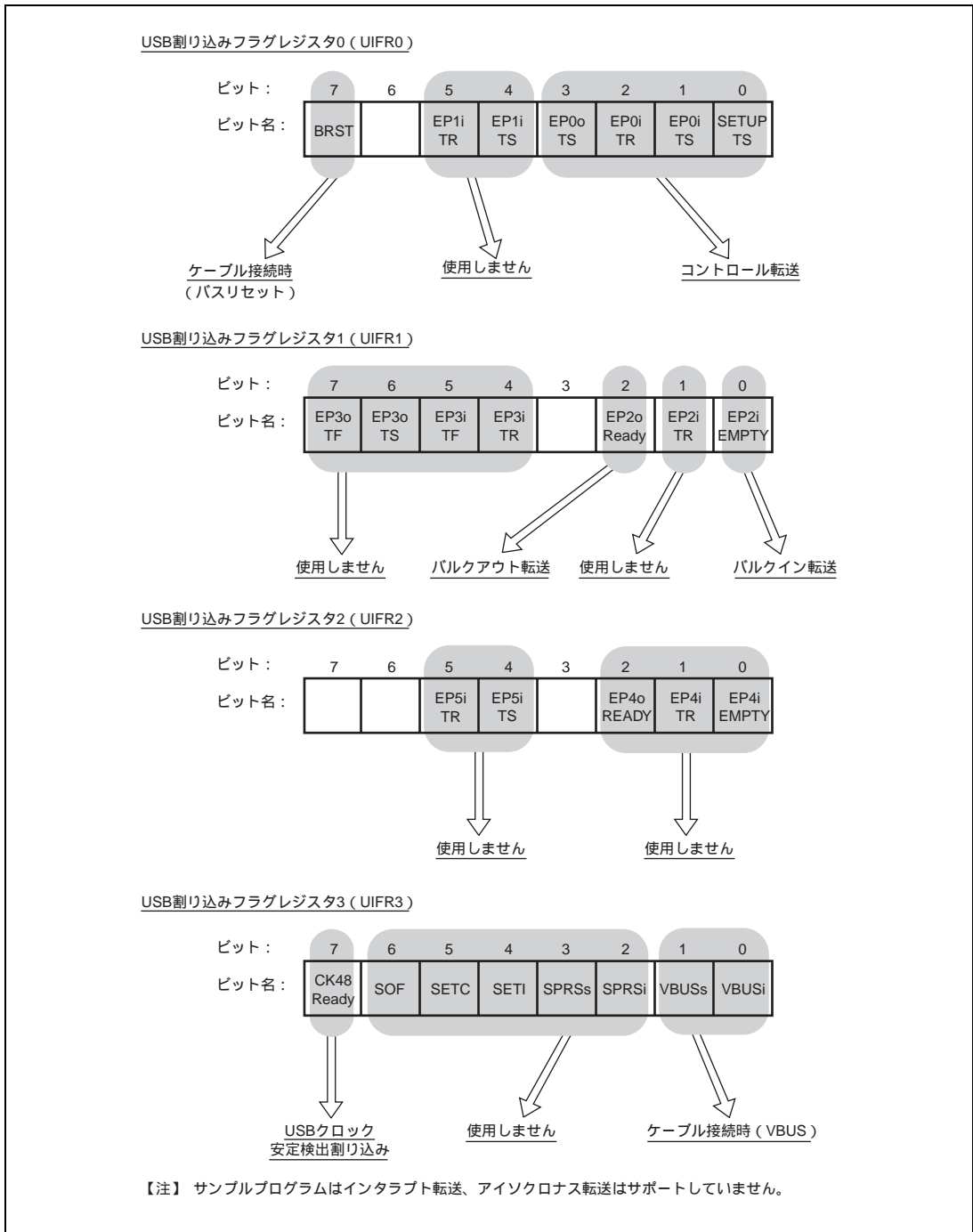


図 4.2 USB 割り込みフラグの種類

4. サンプルプログラムの動作

シリアル割り込み要因が発生した場合も同様にシリアルステータスレジスタの対応するフラグに1がセットされ、CPU に対して割り込み要求します。本サンプルプログラムでは、送信データエンプティ、受信データフル、つまりシリアル送信、シリアル受信機能をサポートします。しかし、シリアル送信は割り込み動作ではなくメインループからの分岐により動作するため、フラグとして使用し、割り込み機能は使用しません。

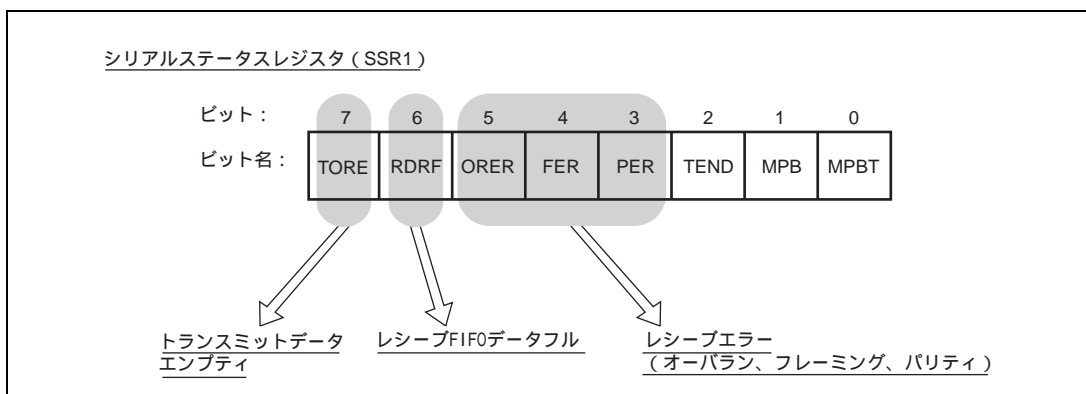


図 4.3 シリアル割り込みフラグの種類

4.2.1 各転送への分岐方法

本サンプルプログラムでは、メインループからの分岐、もしくは USB モジュール、SC11 モジュールからの割り込みによって転送方式を決定しています。表 4.1 に各転送方式の呼び出し方法をまとめます。

メインループからの分岐の場合は、直接関数が呼ばれます。シリアルアウト転送 (ActSerialOut)、バルクイン転送 (ActBulkIn) がこれに当たります。

USB 割り込みによる各転送方式への分岐は UsbMain.c の BranchOfInt が行います。USB 動作クロック安定検出時 (SetUsbModule)、ケーブル接続時 (ActBusReset、ActBusVcc)、コントロール転送 (ActControl)、バルクアウト転送 (ActBulkOut) がこれに当たります。

SC11 割り込みによる各転送方式への分岐は、SC11 割り込みは ER12、RX12、TX12 と転送方式毎に割り込み要因が設定されているため、直接関数を呼び出します。シリアルイン転送 (ActSerialIn) がこれに当たります。

表 4.1 各転送方式の呼び出し方法

モジュール	転送方式	呼び出し方法
USB	USB 動作クロック安定検出	USB 割り込み
	ケーブル接続 (バスリセット)	USB 割り込み
	ケーブル接続 (BusVcc)	USB 割り込み
	コントロール転送	USB 割り込み
	バルクアウト転送	USB 割り込み
	バルクイン転送	メインループからの分岐
SC11	シリアルイン転送	SC11 割り込み
	シリアルアウト転送	メインループからの分岐

表 4.2 に USB 割り込みの種類と、BranchOfInt が呼び出す関数の関係を示します。

表 4.2 USB 割り込みの種類と分岐先関数

レジスタ名	ビット	ビット名	呼び出す関数名
UIFR0	0	BRST	ActBusReset
	1	-	-
	2	EP1i TR	-
	3	EP1i TS	-
	4	EP0o TS	ActControlIn, ActControlOut
	5	EP0i TR	ActControlOut
	6	EP0i TS	ActControlIn, ActControlOut
	7	SETUP TS	ActControl
UIFR1	7	EP3o TF	-
	6	EP3o TS	-
	5	EP3i TF	-
	4	EP3i TR	-
	3	-	-
	2	EP2o Ready	ActBulkOut
	1	EP2i TR	-
	0	EP2i EMPTY	- (メインループからの分岐)
UIFR3	7	CK48 Ready	SetUSBModule
	6	SOF	-
	5	SETC	-
	4	SETI	-
	3	SPRSs	-
	2	SPRSi	-
	1	VBUSs	-
	0	VBUSi	ActBusVcc

EP0i TS と EP0o TS 割り込みは、コントロールイン、アウト転送の両方で使用します。従って、コントロール転送の方向とステージを管理するために、サンプルプログラムは TRANS_IN、TRANS_OUT、WAIT の 3 つのステートを持っています。詳細は、「4.4 コントロール転送」をご覧ください。

表 4.3 に SCIF2 割り込みの種類と、呼び出す関数を示します。

4. サンプルプログラムの動作

表 4.3 SCI1 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
SSR1	7	TDRE	- (メインループからの分岐)
	6	RDRF	ActSerialOut
	5	ORER	ActSerialOut
	4	FER	ActSerialOut
	3	PER	ActSerialOut
	2	TEND	-
	1	MPB	-
	0	MPBT	-

以下の項では、USB、SCI1 の各転送方式ごとにアプリケーション側ファームウェアの動作詳細を説明します。

4.3 USB 動作クロック安定検出割り込み

USB モジュールストップ解除後の 48MHz の USB 動作クロック安定時間を自動的にカウントした後に発生します。割り込み受信後、EndPoint 情報を UEPIR00_0 ~ 22_4 に書き込み、各割り込みを設定して、USB ケーブル接続待ち状態へ以降します。

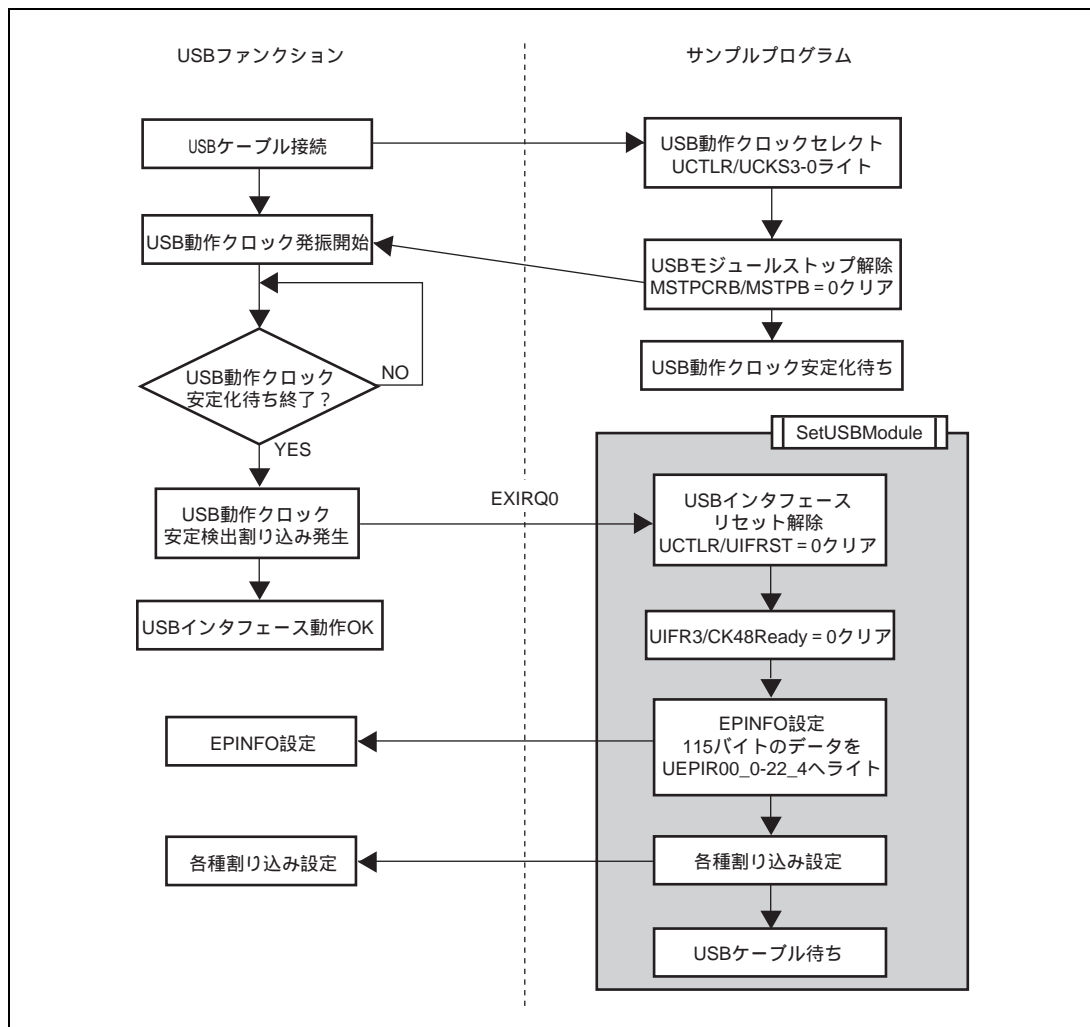


図 4.4 USB 動作クロック安定検出割り込み

4. サンプルプログラムの動作

4.3.1 EPINFO

H8S/2215 搭載の USB ファンクションモジュールは、表 4.4 に示すように、Control 転送：1 系統、BulkIn 転送：2 系統、BulkOut 転送：2 系統、Interrupt In 転送：2 系統、Isochronous In 転送：1 系統、Isochronous out 転送：1 系統の転送タイプをサポートしています。UEPIR レジスタにより、Control 転送以外は任意にその EndPoint 番号、Interface 番号、Alternate 番号、MaxPacketSize 設定が可能となっています。

表 4.4 各転送タイプと UEPIR との関係

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

本アプリケーションノートでは、EndPoint 構成を図 4.5 のように設定します。

なお、H8S/2215 のハードウェアマニュアルでは、Bluetooth 規格に準拠した場合の EndPoint 構成で説明されており、それぞれ、図のように対応します。

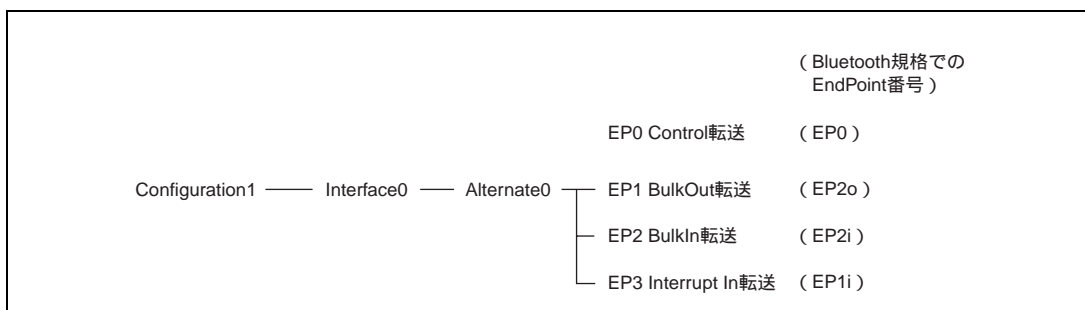


図 4.5 本アプリケーションノートの EndPoint 構成

図 4.5 の EndPoint 構成を実現するための、UEPIR00_0 ~ 22_4 の設定を表 4.5 に示します。表に示すように、使用しない EndPoint についても必ずダミーデータをライトしてください。

表 4.5 UEPIR の設定値

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

4. サンプルプログラムの動作

4.4 ケーブル接続時 (BRST、VBUS) 割り込み

USB ファンクションモジュールのケーブルを、ホストコントローラに接続した際に発生します。アプリケーション側はマイコンの初期設定完了後、汎用出力ポートを使用して USB データバスの D+ をプルアップします。このプルアップによって、ホストコントローラはデバイスが接続された事を認識します。(図 4.6 参照)

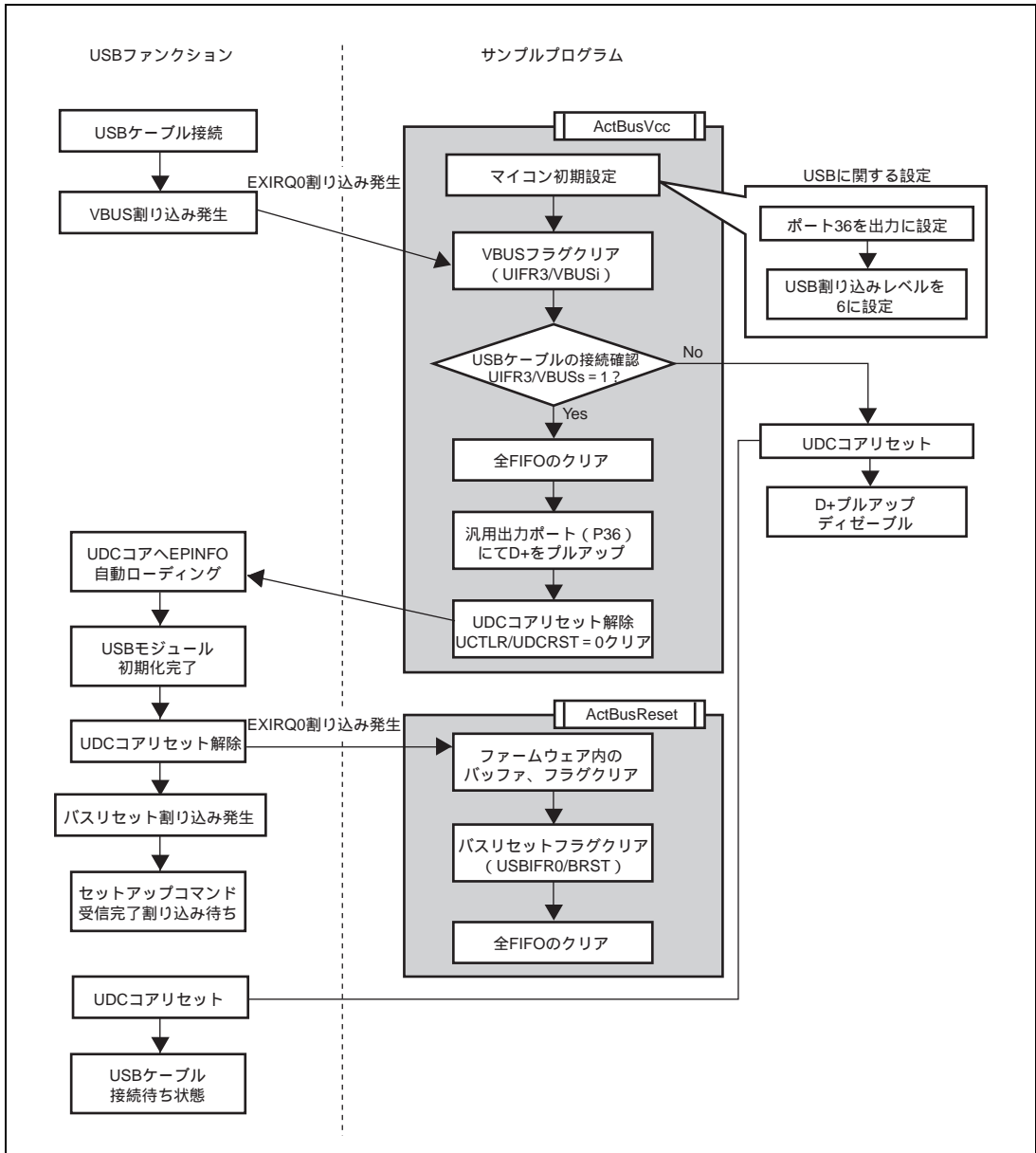


図 4.6 ケーブル接続時割り込み

4.5 コントロール転送

コントロール転送には、割り込みフラグレジスタのビット 0~3 を使用します。コントロール転送は、データステージにおけるデータの向きによって、2 つに分ける事ができます。(図 4.7 参照)

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

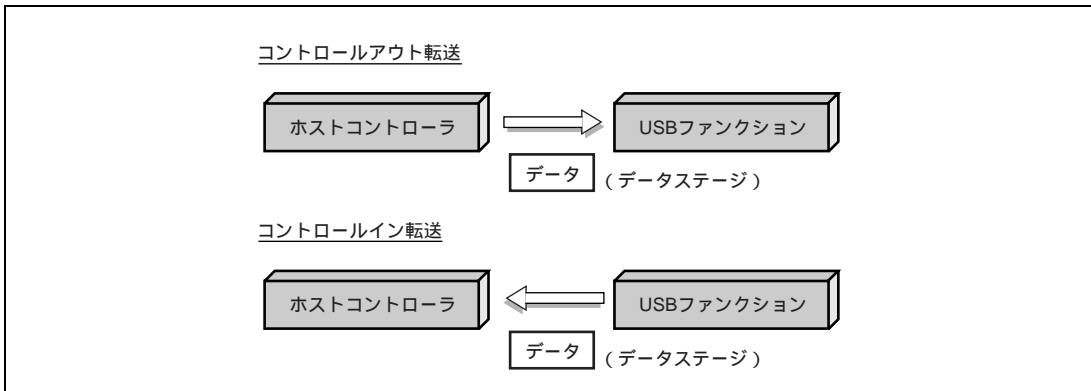


図 4.7 コントロール転送

コントロール転送は、セットアップ、データ(ない場合もある)、ステータスの 3 つのステージで構成されます(図 4.8)。また、データステージは、複数のバストランザクションで構成されます。

コントロール転送では、データの向きが反転する事によってステージが切り替わったことを認識します。したがって同じ割り込みフラグを使用して、コントロールイン転送または、コントロールアウト転送を行う関数を呼び出します(表 4.1 参照)。このため、現在イン、アウトどちらのコントロール転送が行われているかをファームウェアがスタートによって管理し(図 4.8 参照)、適切な関数を呼び出す必要があります。データステージにおけるステート (TRANS_IN、TRANS_OUT) は、セットアップステージで受信するコマンドによって決定します。

4. サンプルプログラムの動作

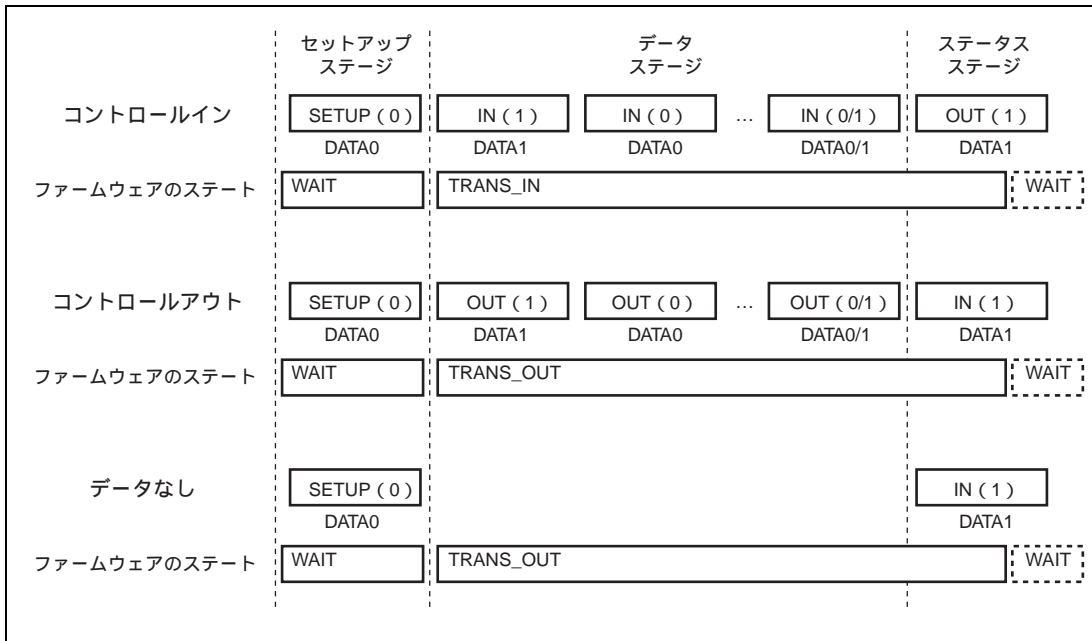


図 4.8 コントロール転送における各ステージ

4.5.1 セットアップステージ

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

- コントロールインとなるコマンド

GetDescriptor	(TRANS_IN)	標準コマンド
GetLineCoding	(TRANS_IN)	ベンダコマンド
- コントロールアウトとなるコマンド

SetLineCoding	(TRANS_OUT)	ベンダコマンド
SetControlLineState	(TRANS_OUT)	ベンダコマンド
SendBreak	(TRANS_OUT)	ベンダコマンド

図 4.9 にセットアップステージにおけるサンプルプログラムの動作を示します。図の左側は、USB ファンクションモジュールの動作を示しています。

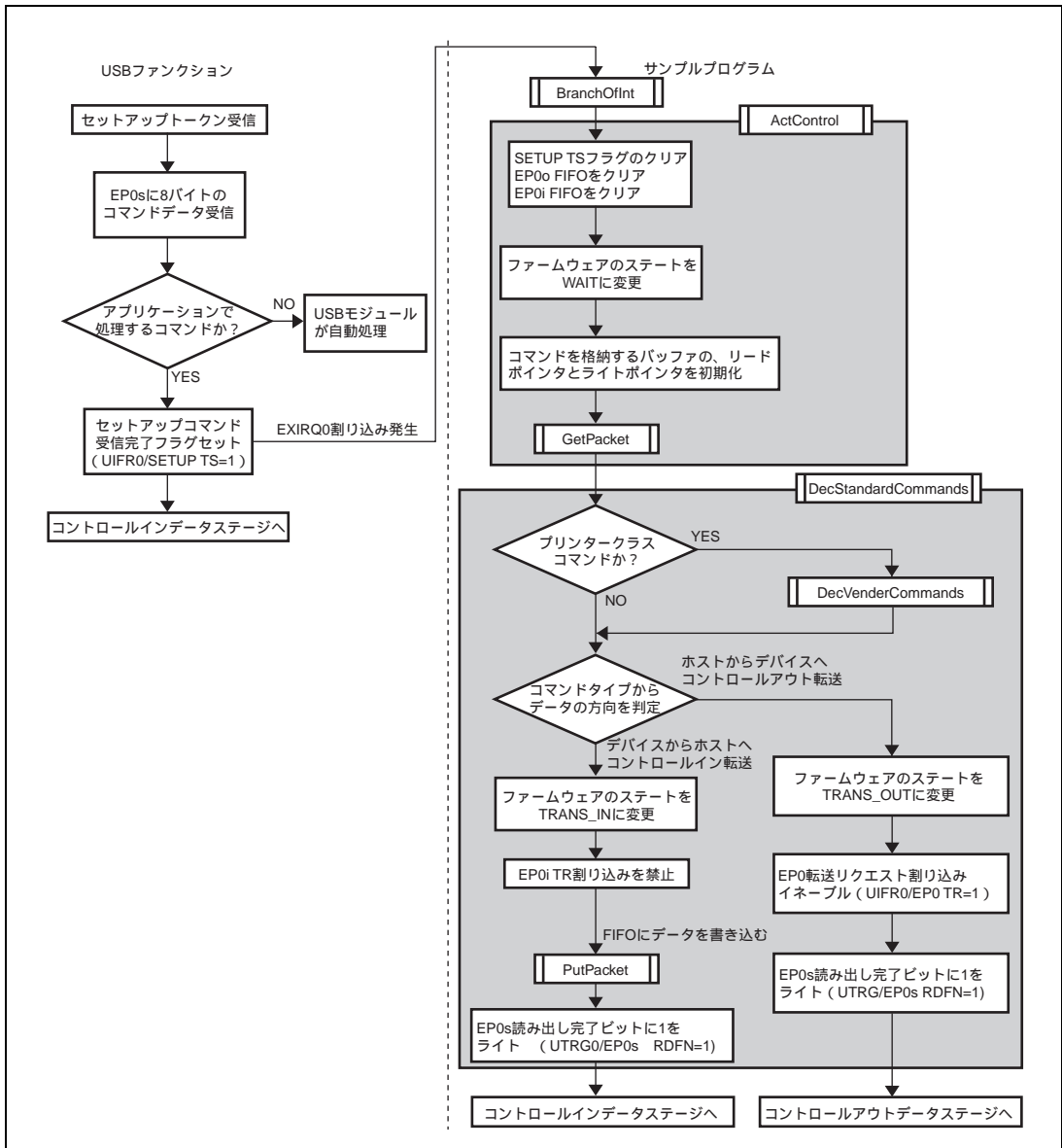


図 4.9 セットアップステージ

4. サンプルプログラムの動作

4.5.2 データステージ

データステージでは、ホストとファンクションがデータの送受信を行います。ファームウェアのステータスは、セットアップステージで行ったコマンドのデコード結果によって、コントロールイン転送の場合は“TRANS_IN”に、コントロールアウト転送の場合は“TRANS_OUT”になります。

図 4.10、図 4.11 にコントロール転送のデータステージにおけるサンプルプログラムの動作を示します。

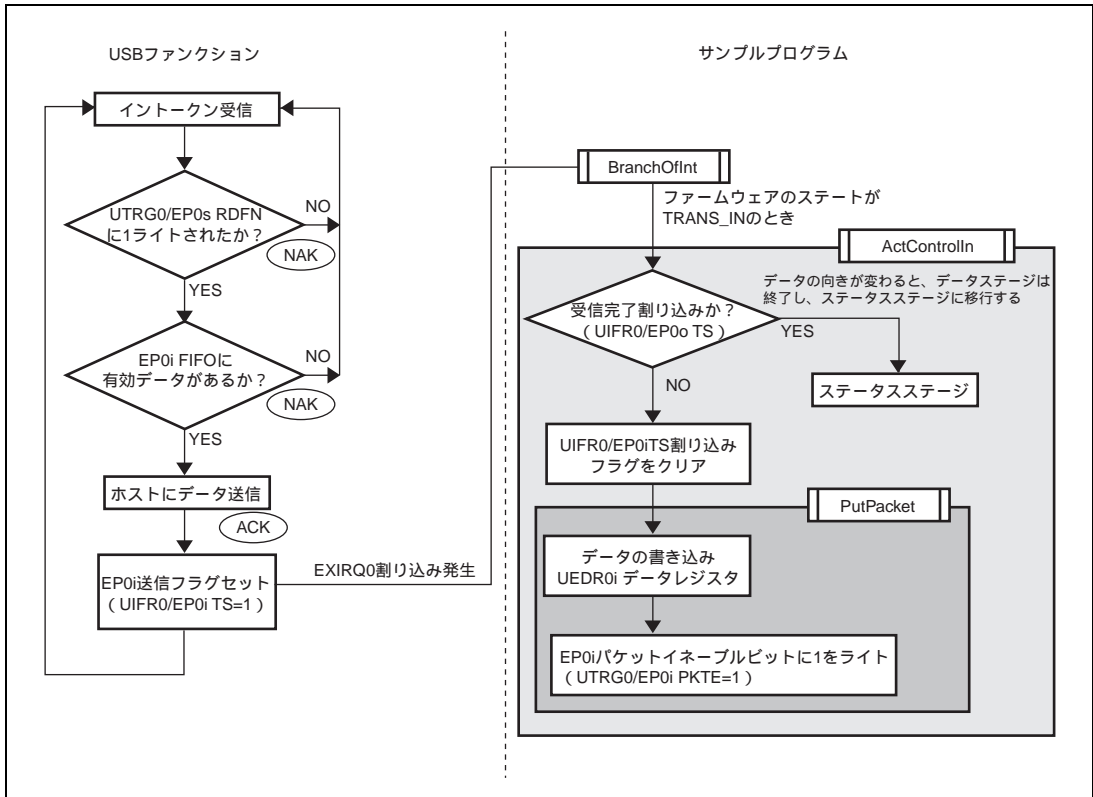


図 4.10 データステージ (コントロールイン転送)

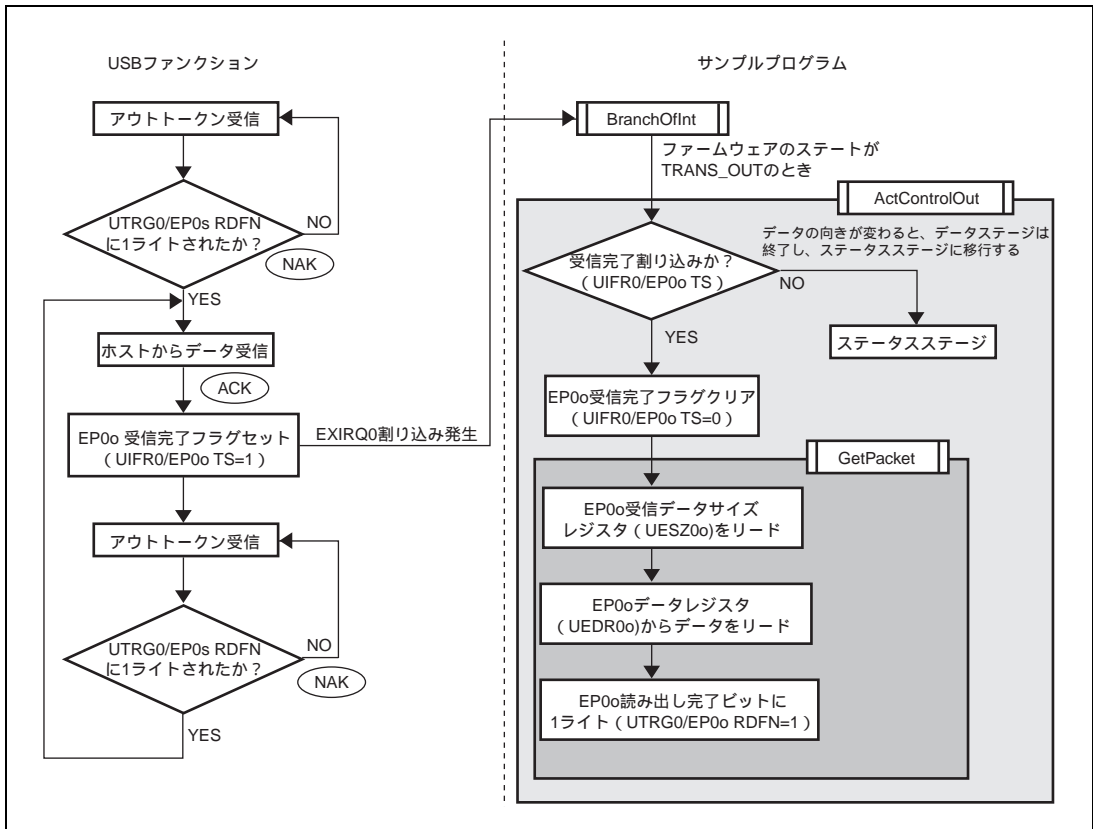


図 4.11 データステージ (コントロールアウト転送)

4. サンプルプログラムの動作

4.5.3 ステータスステージ

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

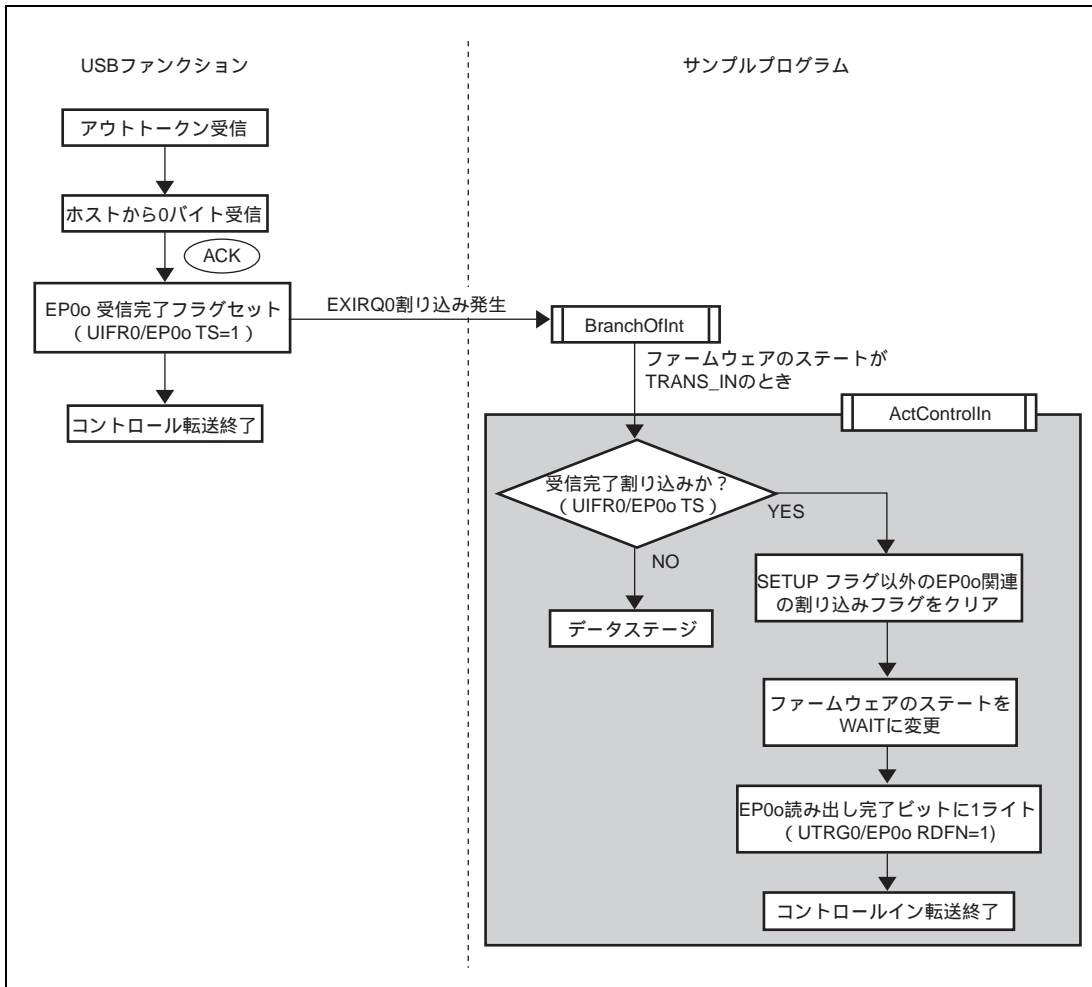


図 4.12 ステータスステージ (コントロールライン転送)

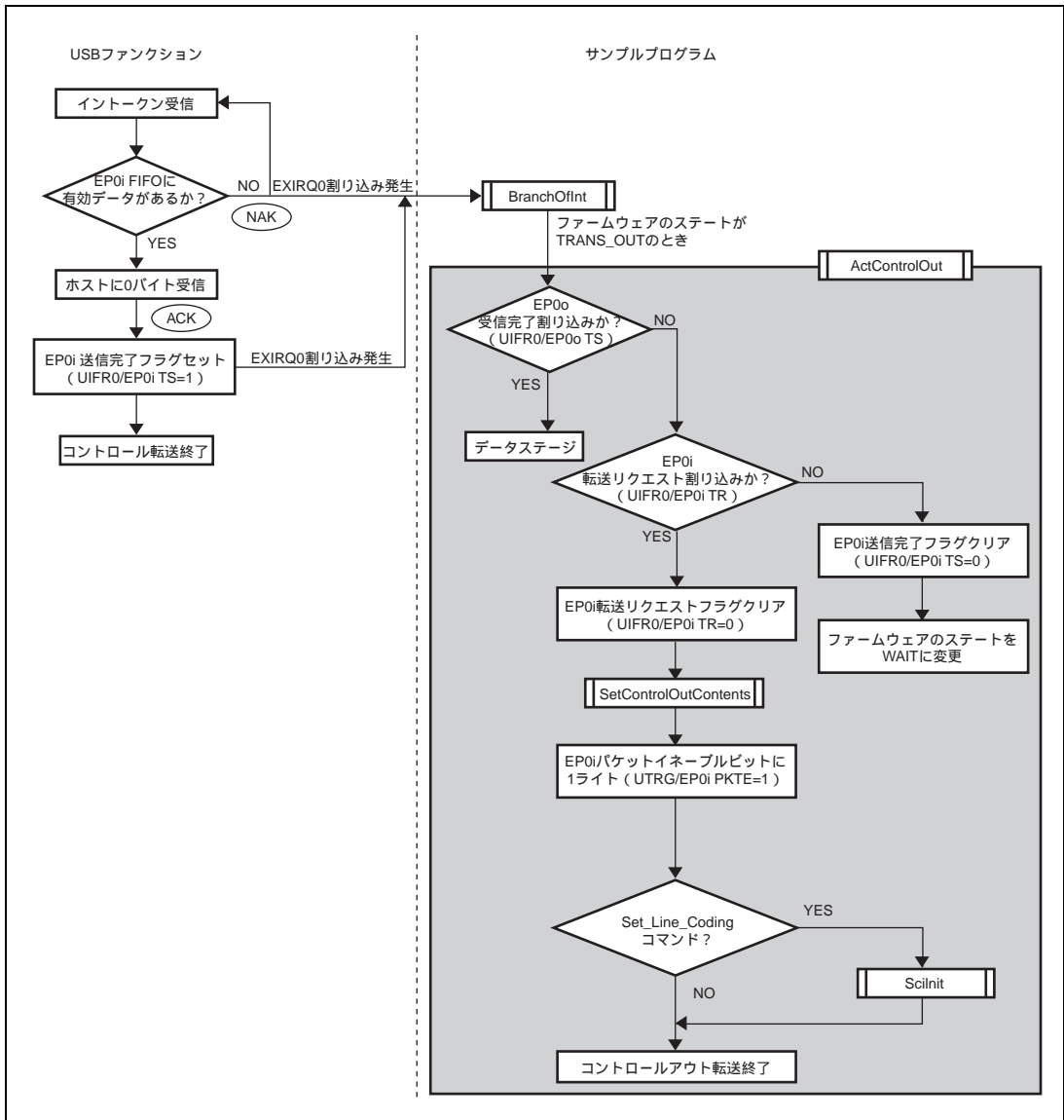


図 4.13 ステータスステージ (コントロールアウト転送)

4.6 バルク転送

バルク転送には、割り込みフラグレジスタ1のビット0~2を使用します(本プログラムではバルクイン転送は割り込み起因での起動ではないためビット0、1は使用しません)。バルク転送はデータを送信する向きによって、2つに分けることができます。(図4.14参照)

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

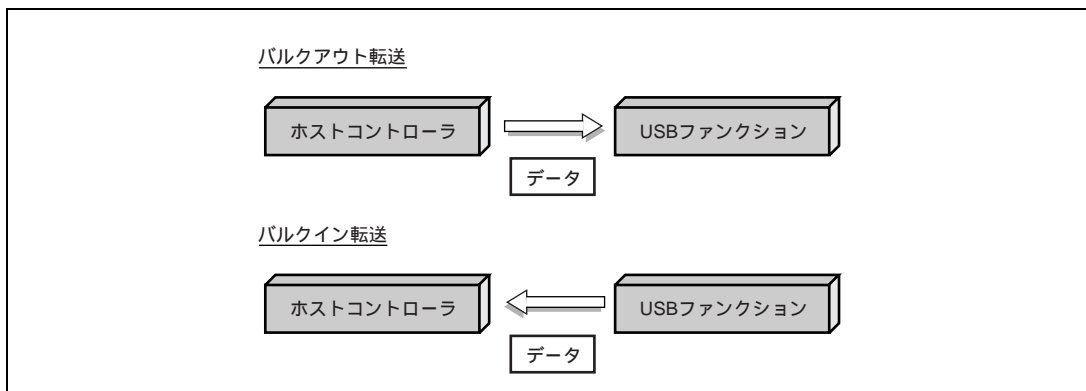


図 4.14 バルク転送

4.6.1 バルクアウト転送

図 4.15 にバルクアウト転送におけるサンプルプログラムの動作を示します。

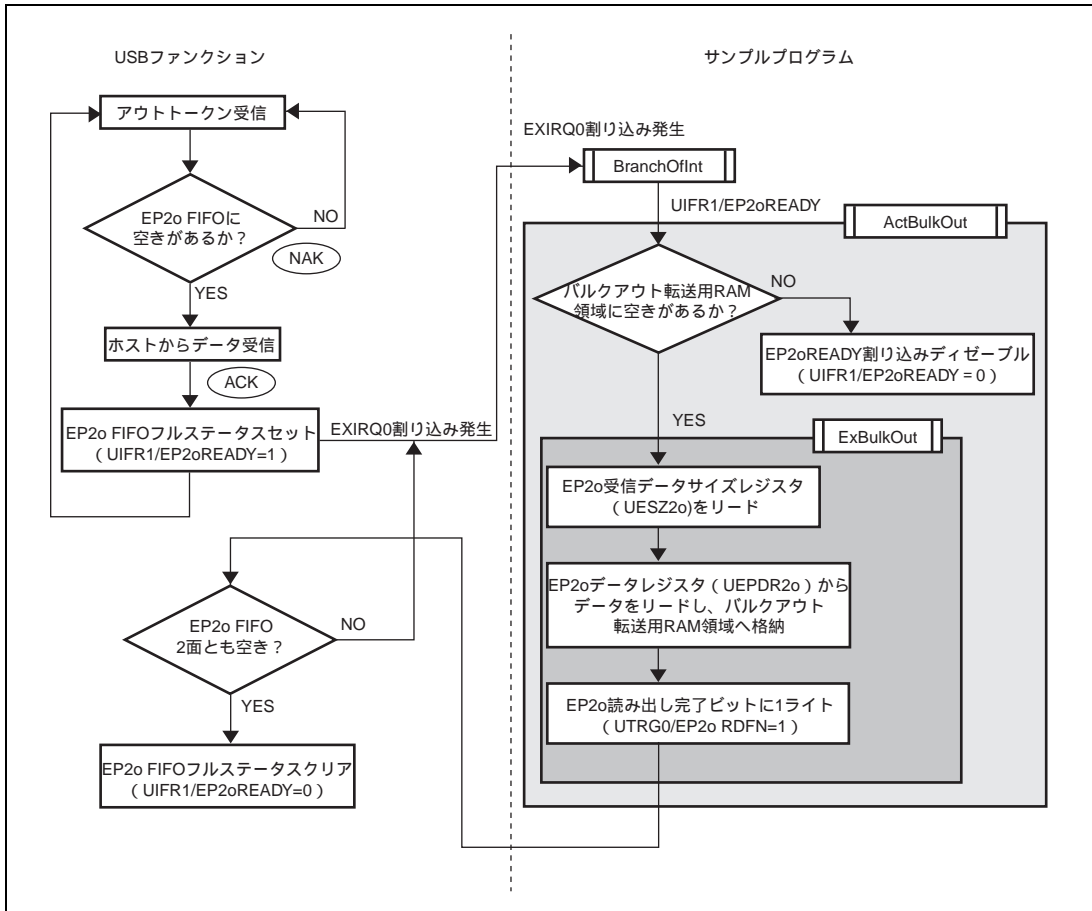


図 4.15 バルクアウト転送

4.6.2 バルクイン転送

図 4.16 にバルクイン転送におけるサンプルプログラムの動作を示します。バルクアウト転送と異なり、割り込み起因で起動するのではなく、メインループから呼ばれます。

バルクイン転送用 RAM 領域に格納されているデータを USBEP2 データレジスタに書き込みます。その後、この RAM 領域に空き領域がなくシリアルイン転送がディゼーブル状態の場合は、UEDR2I データレジスタにデータを書き込んだことで、この RAM 領域に空き領域ができたか確認します。空き領域ができた場合はシリアルイン転送をイネーブルにします。

4. サンプルプログラムの動作

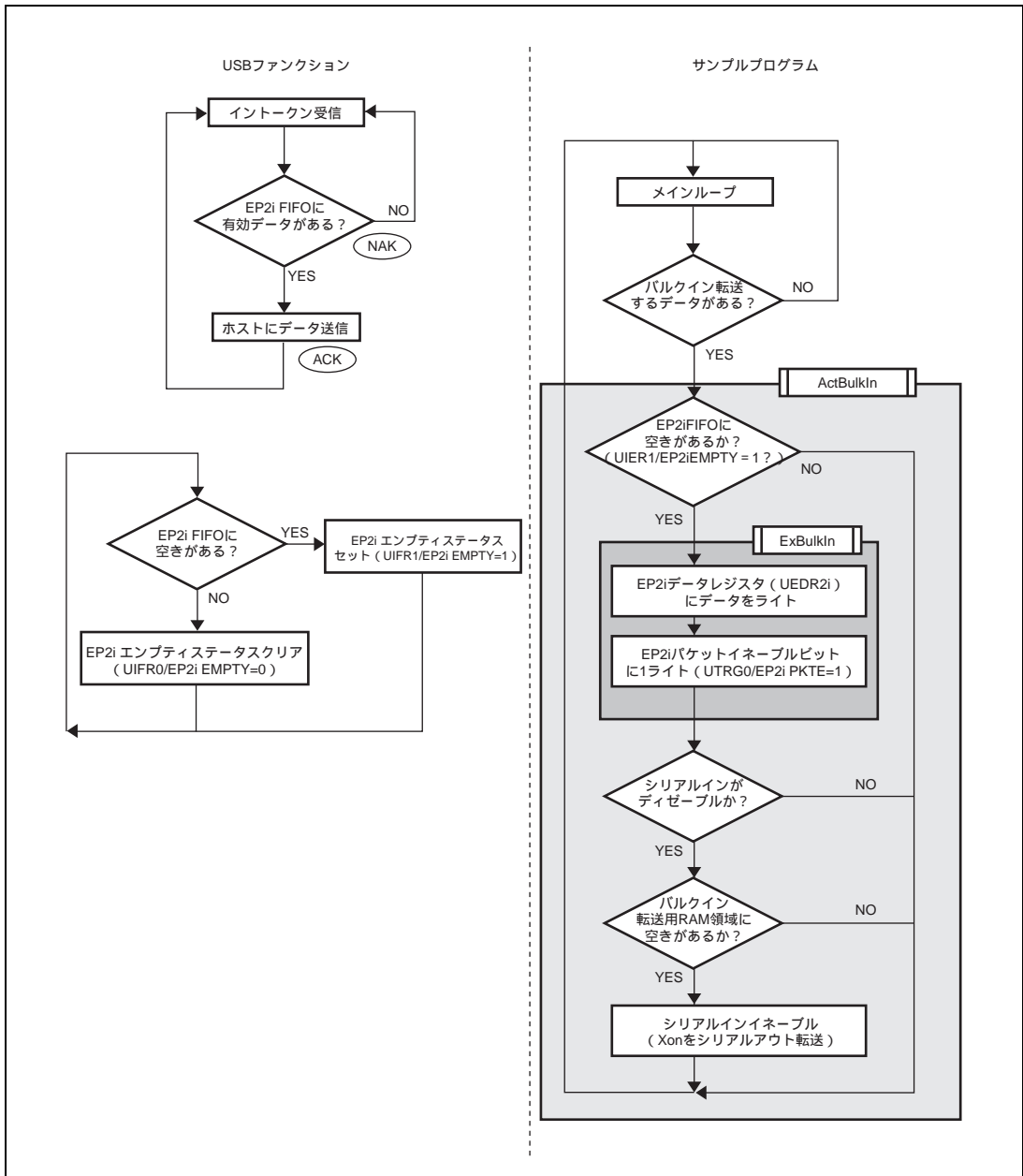


図 4.16 バルクイン転送

4.7 シリアル転送

シリアル転送には、SCII モジュールを使います。シリアルアウト転送はメインループからの分岐によって実行され、シリアルイン転送は割り込みによって実行されます。シリアルイン転送は、シリアルステータスレジスタ(SSR1)のRDRF フラグを使用します。

4.7.1 シリアルアウト転送

図 4.17 にシリアルアウト転送におけるサンプルプログラムの動作を示します。シリアルアウト転送は、バルクアウト転送用 RAM 領域にデータがあれば、メインループから分岐して ActSerialOut 関数を呼び出し、SCII モジュールを制御してシリアルアウト転送によりデータを送信します。もし、バルクアウト転送用 RAM 領域に空き容量がない状態でバルクアウト転送がディゼーブル状態である場合、このシリアルアウト転送によって空き容量が確保できたか確認し、確保できた場合は、バルクアウト転送をイネーブルにします。

4. サンプルプログラムの動作

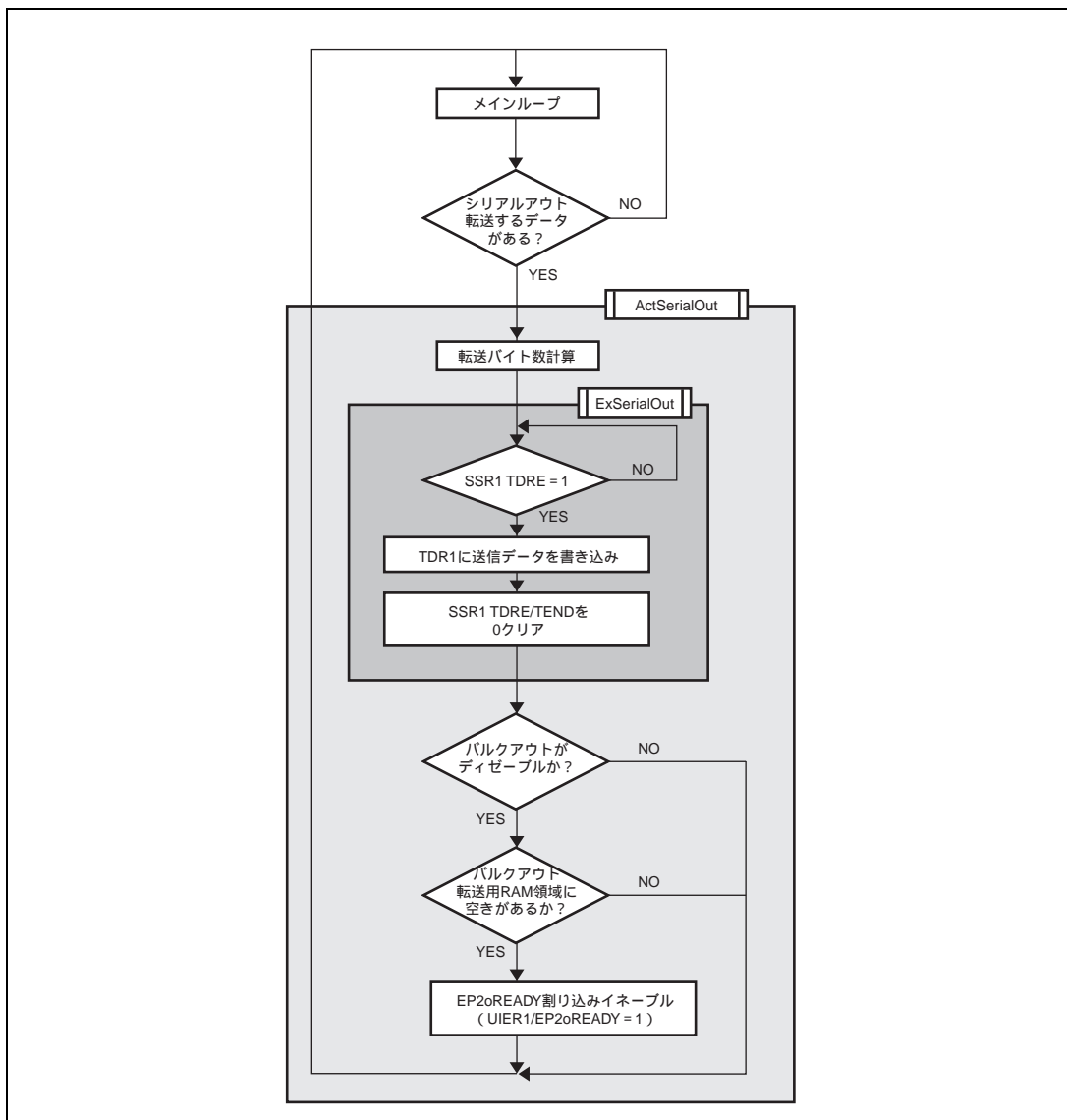


図 4.17 シリアルアウト転送

4.7.2 シリアルイン転送

図 4.18、図 4.19 にシリアルイン転送におけるサンプルプログラムの動作を示します。ERI1、RXI1 の受信割り込みが発生した場合、ActSerialIn 関数が呼びべられます。

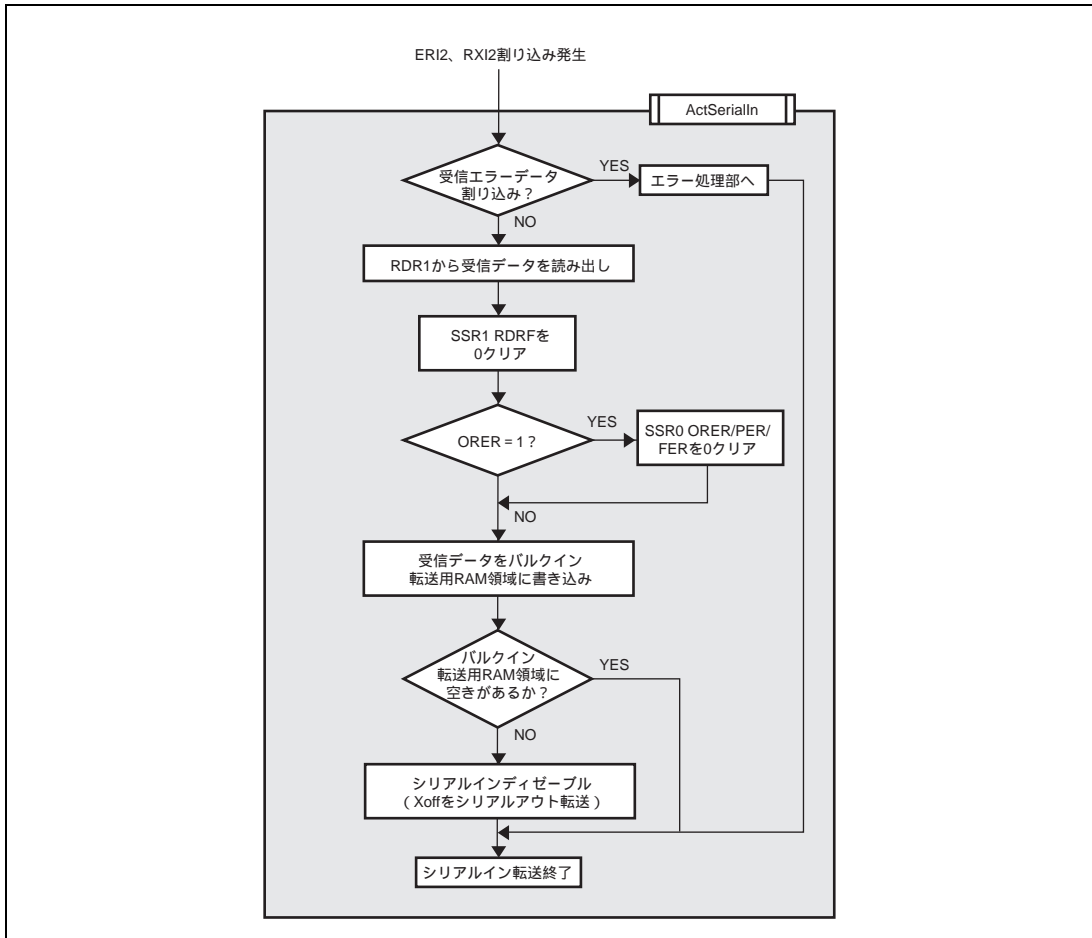


図 4.18 シリアルイン転送 (受信データ処理部)

4. サンプルプログラムの動作

ERI1 受信では、ORER が発生した場合は、RXI1 受信時と同様に RDR1 からデータを読み出します。ORER が伴わない受信エラーの場合は、RDR1 の値をダミーリードして捨てて、エラーフラグをクリアします。この時、ブ레이크信号も受信している場合は、シリアル受信をディゼーブルにし、FER フラグをクリアせずに関数を抜けます。この場合、FER は 1 が立ったままなので、ブ레이크信号を受信しなくなるまで、連続して割り込みが発生し、ActSerialIn 関数が呼ばれ続けます。また、この期間中に USB 割り込みを受け付けるために、USB と SCI1 の割り込み優先順位を逆転させます。

ORER エラーもしくは、正常受信の場合は、RDR1 からデータを受信し、バルクイン転送用 RAM 領域に書き込みます。その後、この RAM 領域の空き容量をチェックし、空き容量がない場合はデータの欠落を防止するために、シリアル接続 PC へ Xoff を出力し、シリアルイン転送をディゼーブルにします。

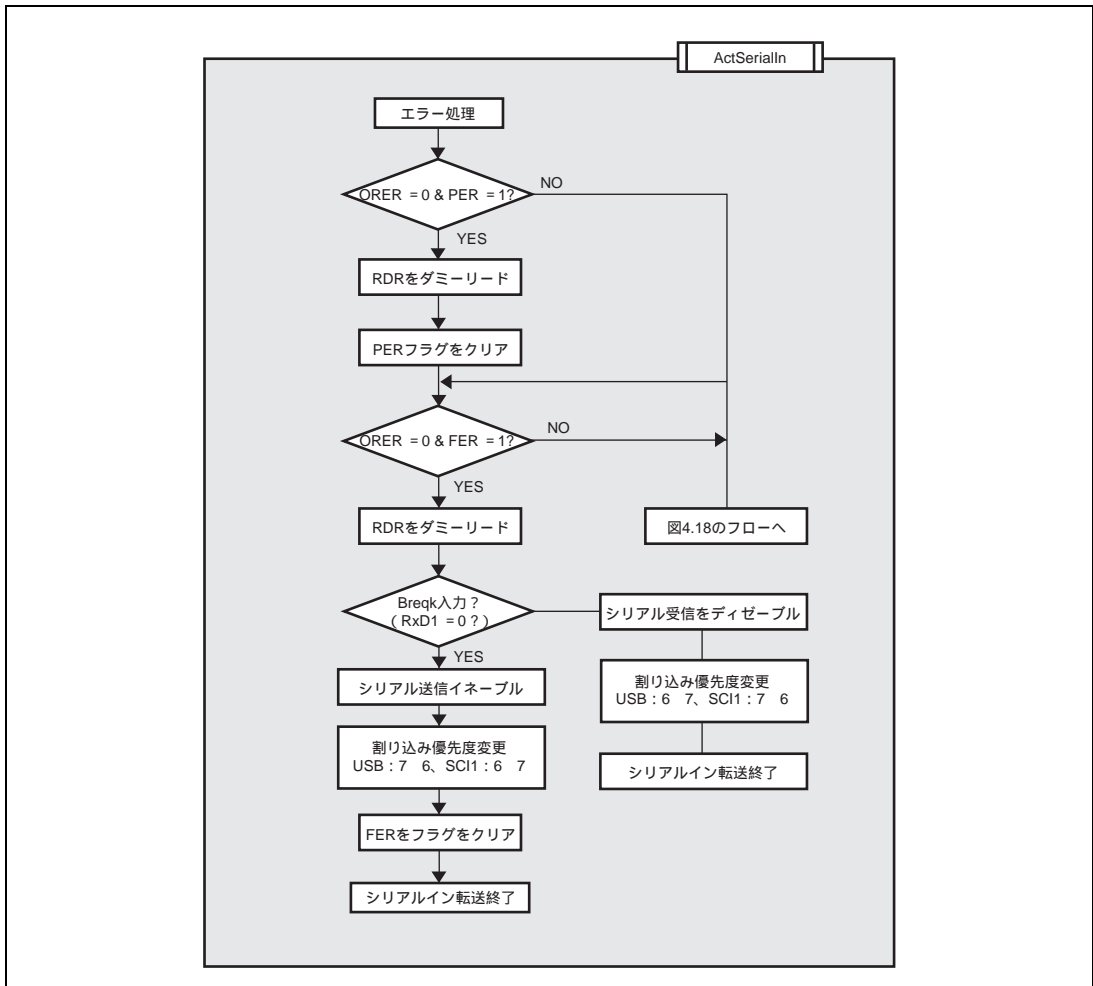


図 4.19 シリアルイン転送（エラー処理部）

4.8 ベンダコマンド

本サンプルプログラムでは、日立超 LSI システムズ社製の USB シリアル変換ドライバがサポートする 4 つのベンダコマンドをデコードします。

表 4.6 に USB シリアル変換ドライバが規定する 4 つのベンダコマンドを示します。

表 4.6 (a) ベンダリクエスト

bmRequestType	bRequest	wValue	wIndex	wLength	Data
01000001b	SET_LINE_CODING	Zero	Interface	8	Line Coding Structure
11000001b	GET_LINE_CODING	Zero	Interface	8	Line Coding Structure
01000001b	SET_CONTROL_LINE_STATE	Control Signal Bitmap	Interface	Zero	None
01000001b	SEND_BREAK	Duration of Break	Interface	Zero	None

表 4.6 (b) ベンダリクエストコード

bRequest	Value
SET_LINE_CODING	0
GET_LINE_CODING	1
SET_CONTROL_LINE_STATE	2
SEND_BREAK	3

次節以降に、各コマンドの詳細を示します。

4.8.1 SetLineCoding

本リクエストは、調歩同期式のデータ伝送で使用されるパラメータを設定します。

bmRequestType	bRequest	wValue	wIndex	wLength	Data
01000001b	SET_LINE_CODING	Zero	Interface	8	Line Coding Structure

Line Coding Structure の定義を表 4.7 に示します。

本サンプルプログラムでは、このコマンド受信により、受信した Line Coding Structure の設定で SCI1 を再起動します。

4. サンプルプログラムの動作

表 4.7 Line Coding Structure

Offset	Field	Size	Value	Description
0	DwDTERate	4	Number	データ端末の速度 (bps)
4	BcharFormat	1	Number	ストップビット 0 - 1 Stop bits 1 - 1.5 Stop bits 2 - 2 Stop bits
5	BparityType	1	Number	パリティ 0 - None 1 - Odd 2 - Even 3 - Mask 4 - Space
6	BdataBits	1	Number	データビット (5, 6, 7, 8)
7	BflowType	1	Number	フローコントロール 0 - ソフトウェア or なし 1 - ハードウェア

4.8.2 GetLineCoding

本リクエストは、ホストがデバイスの現在のパラメータについて確認するコマンドです。

本サンプルプログラムは、このコマンドを受信すると、表 4.8 で示すデフォルト値を返します。

bmRequestType	bRequest	wValue	wIndex	wLength	Data
11000001b	GET_LINE_CODING	Zero	Interface	8	Line Coding Structure

表 4.8 Line Coding Structure のデフォルト値

Offset	Field	Size	Value	Description
0	DwDTERate	4	0x1C200	データ端末の速度 (38400bps)
4	BcharFormat	1	0x0	ストップビット (1 Stop bit)
5	BparityType	1	0x0	パリティ (None)
6	BdataBits	1	0x8	データビット (8)
7	BflowType	1	0x0	フローコントロール (ソフトウェア or なし)

4.8.3 SetControlLineState

本リクエストは、制御信号を設定します。

bmRequestType	bRequest	wValue	wIndex	wLength	Data
01000001b	SET_CONTROL_LINE_STATE	Control Signal Bitmap	Interface	Zero	None

表 4.9 Control Signal Bitmap

Bit Position	Description
D15...D2	予約 (0 にリセット)
D1	DCE の送信機能を制御 0 - RTS OFF 1 - RTS ON
D0	DTE がレディ状態かの通知 0 - DTR OFF 1 - DTR ON

H8S/2215 が RTS、DTR の信号線を持たないため、本サンプルプログラムでは、このリクエストについてはデコードのみを行い、DCE の制御は行いません。

なお、本サンプルプログラムでは、D1=1 & D0=1 を検知することで、USB ホスト PC 側ハイパーターミナルの通信準備が完了したと認識し、このタイミングで、バルクイン、バルクアウト転送データ領域を示すポインタ、および本サンプルプログラムの持つ内部フラグの初期化を行います。

4.8.4 SendBreak

本リクエストは、デバイスにブレイク信号を発生させます。

bmRequestType	bRequest	wValue	wIndex	wLength	Data
01000001b	SEND_BREAK	Duration of Break	Interface	Zero	None

wValue フィールドは、ブレイク信号送出時間 (msec) を記入。wValue が 0xFFFF の場合、デバイスは、wValue が 0x0000 の SendBreak リクエストを受信するまで、ブレイク信号を送出し続けます。

本サンプルプログラムでは、このリクエストのデコードは行いますが、ブレイク送信は行いません。

5. アナライザのデータ

この章では、H8S/2215 内蔵 USB ファンクションモジュールを使用して、CATC 社製 USB プロトコルアナライザ「USB Advisor」(国内：(株)東陽テクニカ(<http://www.toyo.co.jp/>))を用いた測定を行い、実際にバスを流れているデータについて「デバイス接続時のコントロール転送」および「ベンダコマンド送信時のバルクアウト転送」を例に説明します。

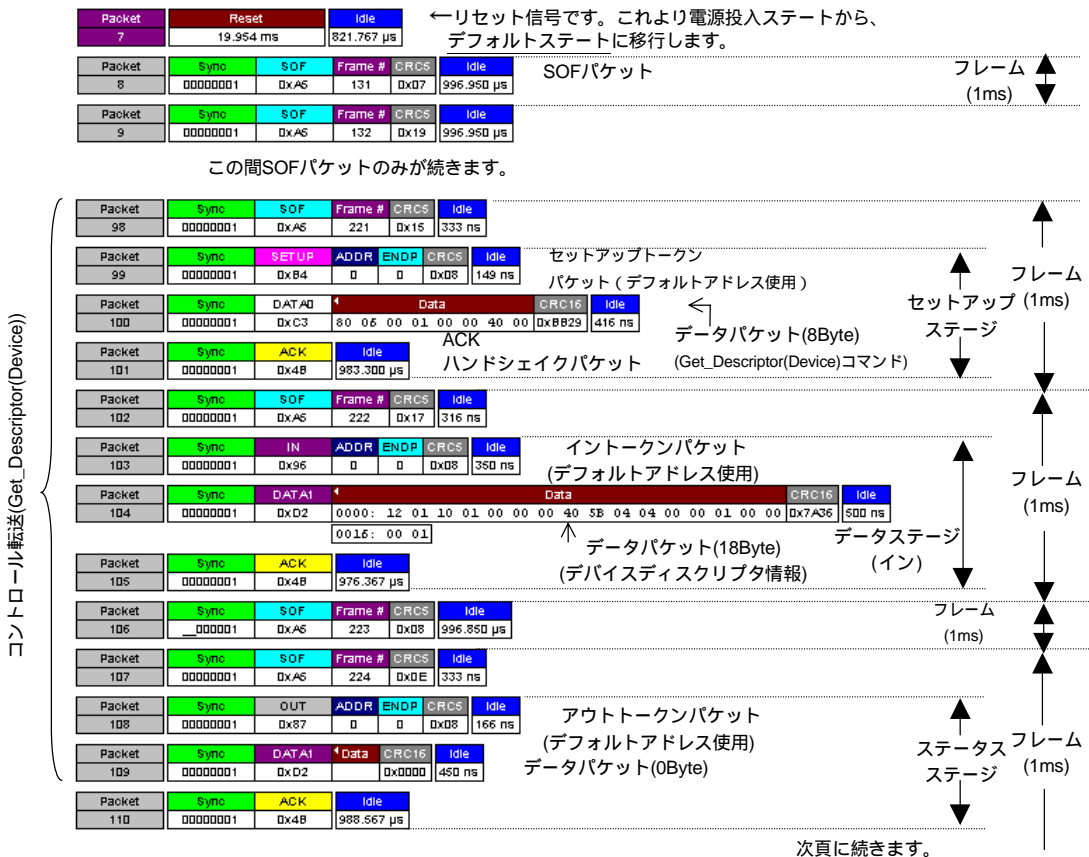
【注】 各パケットの前面にある「Packet#」は測定時のパケット通し番号です。

後面にある「Idle」はパケット間のアイドルとなります。

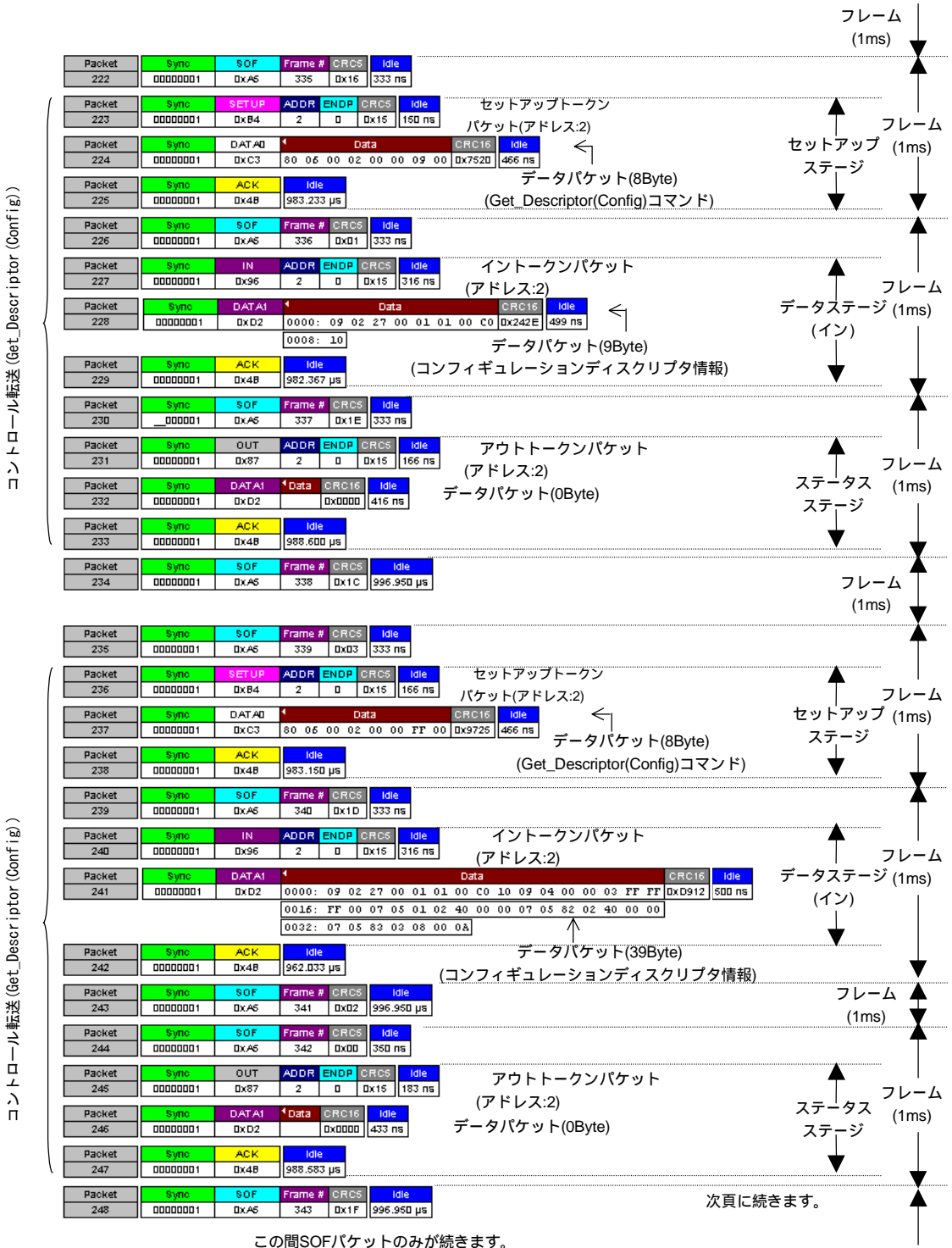
5.1 デバイス接続時のコントロール転送

図 5.1 は本デバイスをホストコントローラに接続し、Vbus に電源が供給されている(電源投入状態)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。

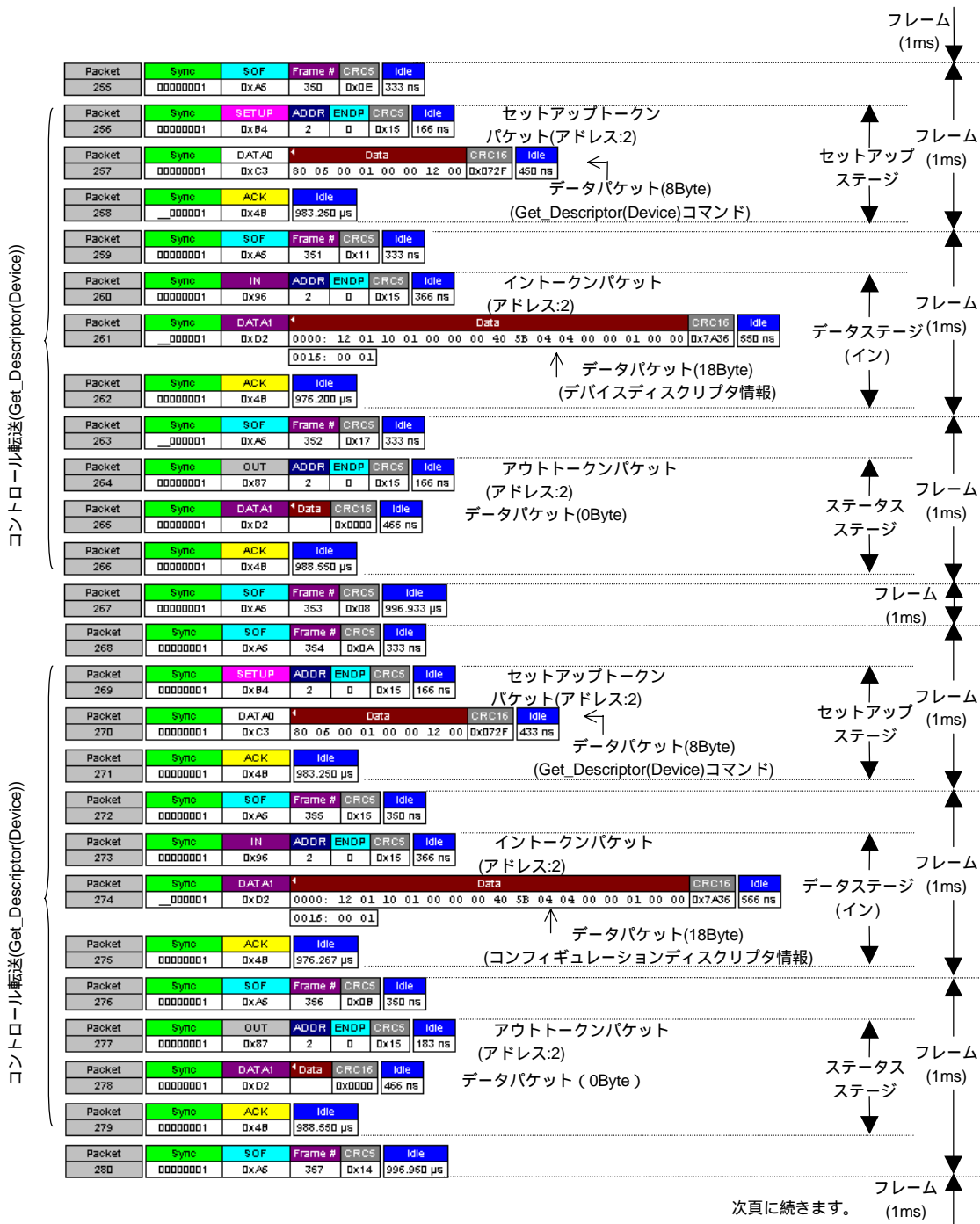
なお、ホストコントローラによりパケットのスケジューリングが本図と異なる場合がありますが構成ステートに至るまでのコマンドの流れは同一です。



5. アナライザのデータ



5. アナライザのデータ



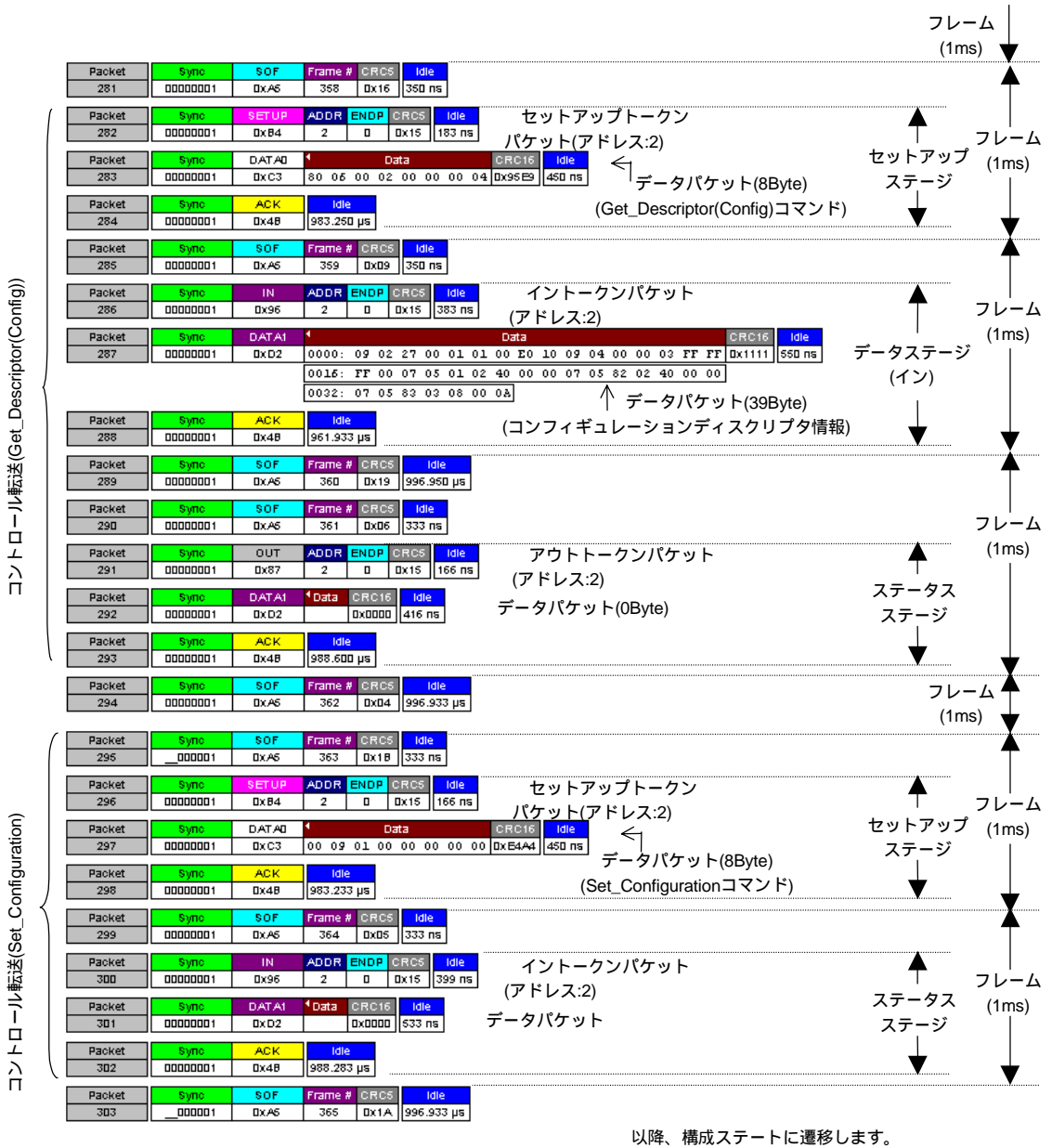
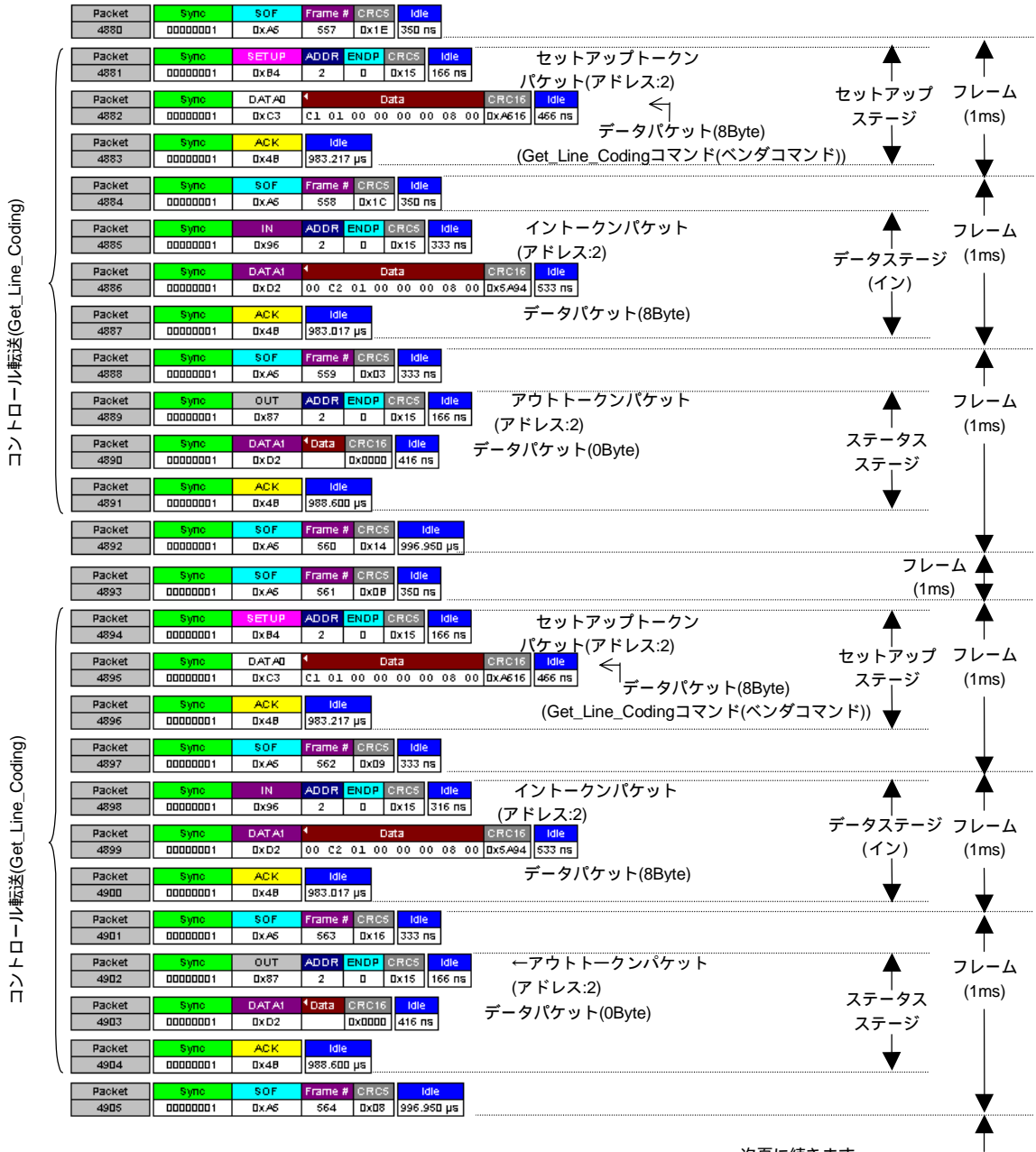


図 5.1 デバイス接続時のコントロール転送

5. アナライザのデータ

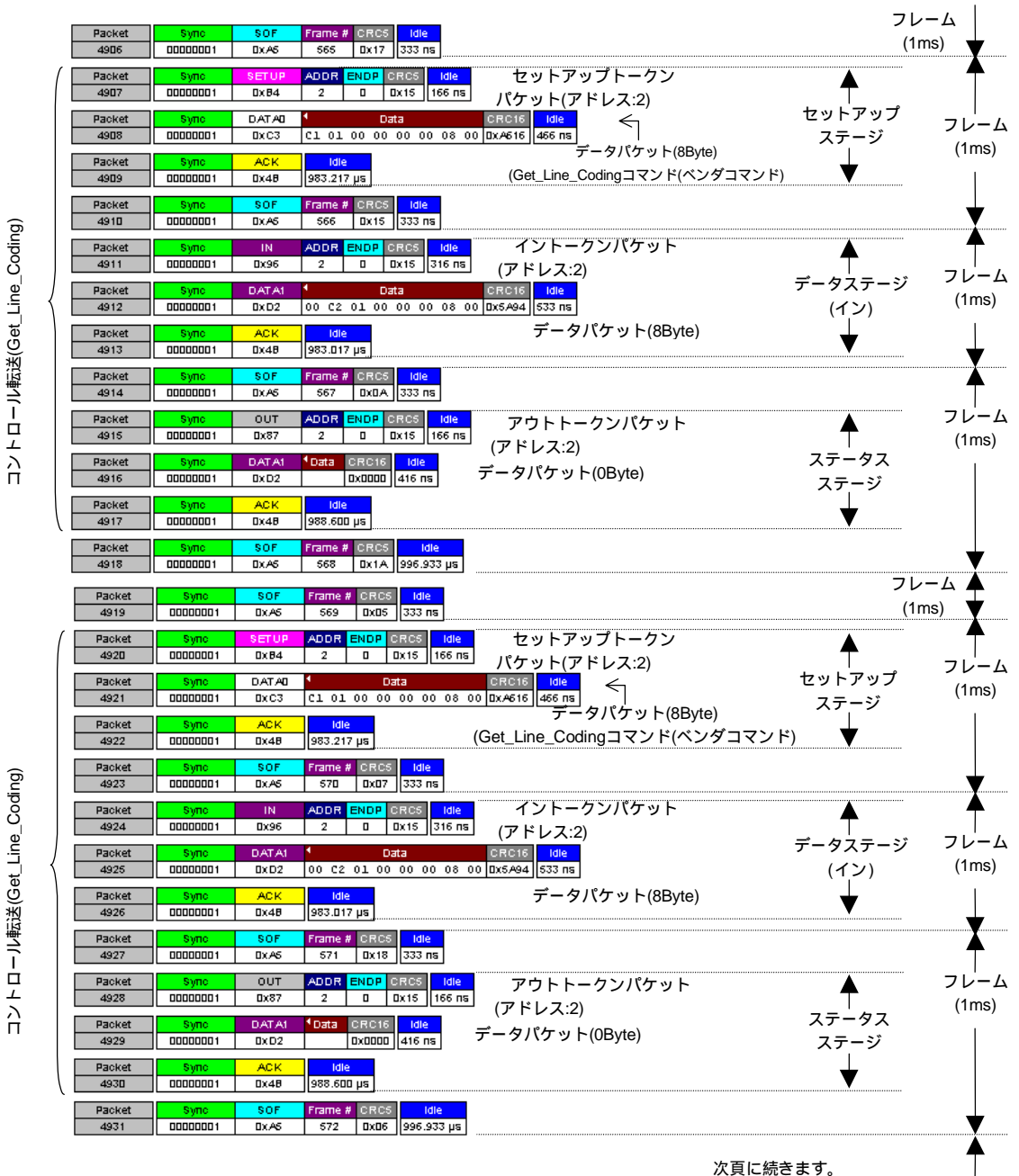
5.2 ベンダコマンド転送時のコントロール転送

図 5.2 は、ホストコントローラとデバイス間でベンダコマンドをコントロール転送で送信している際の測定結果です。(ベンダコマンドについては 4.7 項をご参照下さい)

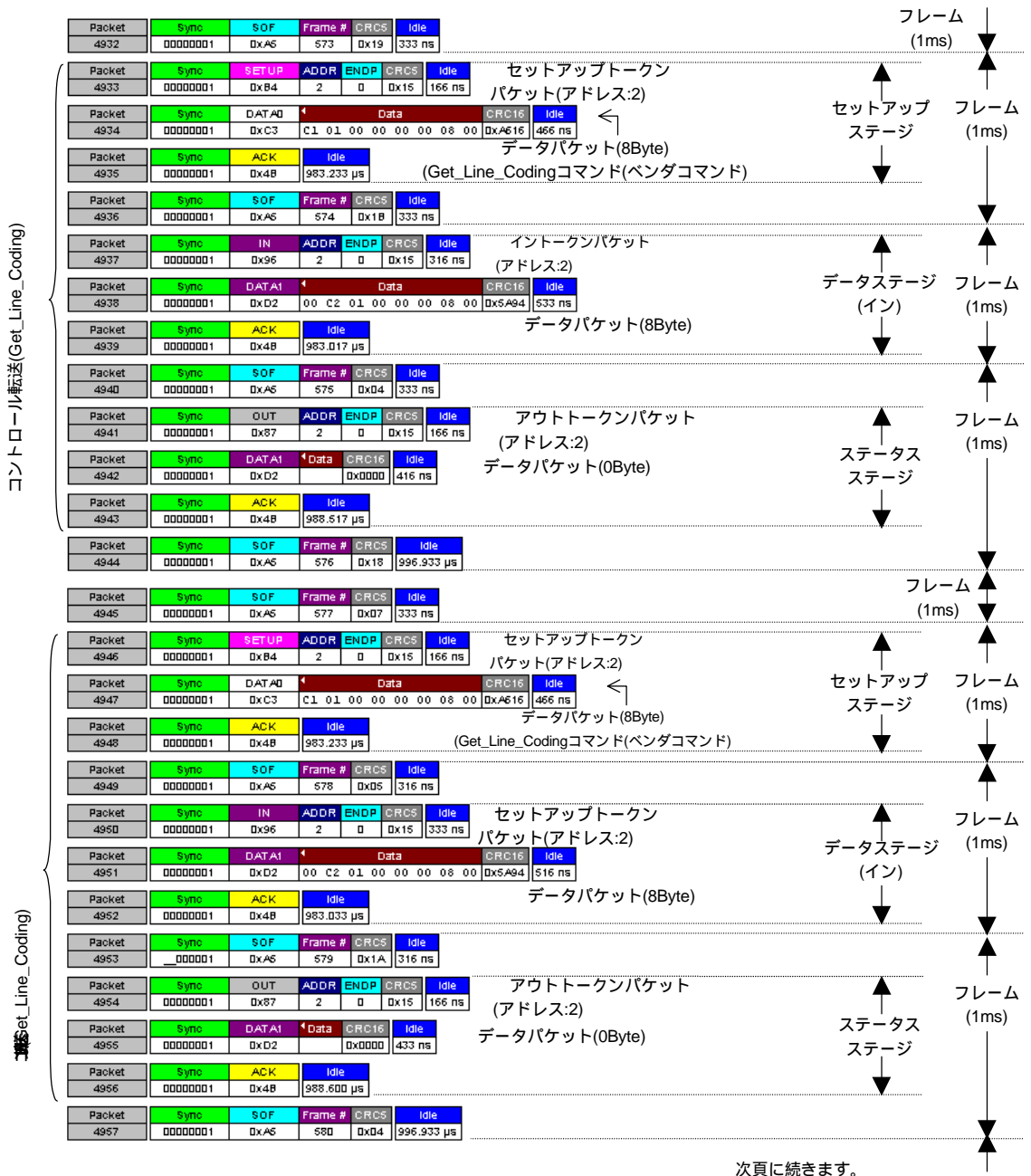


次頁に続きます。

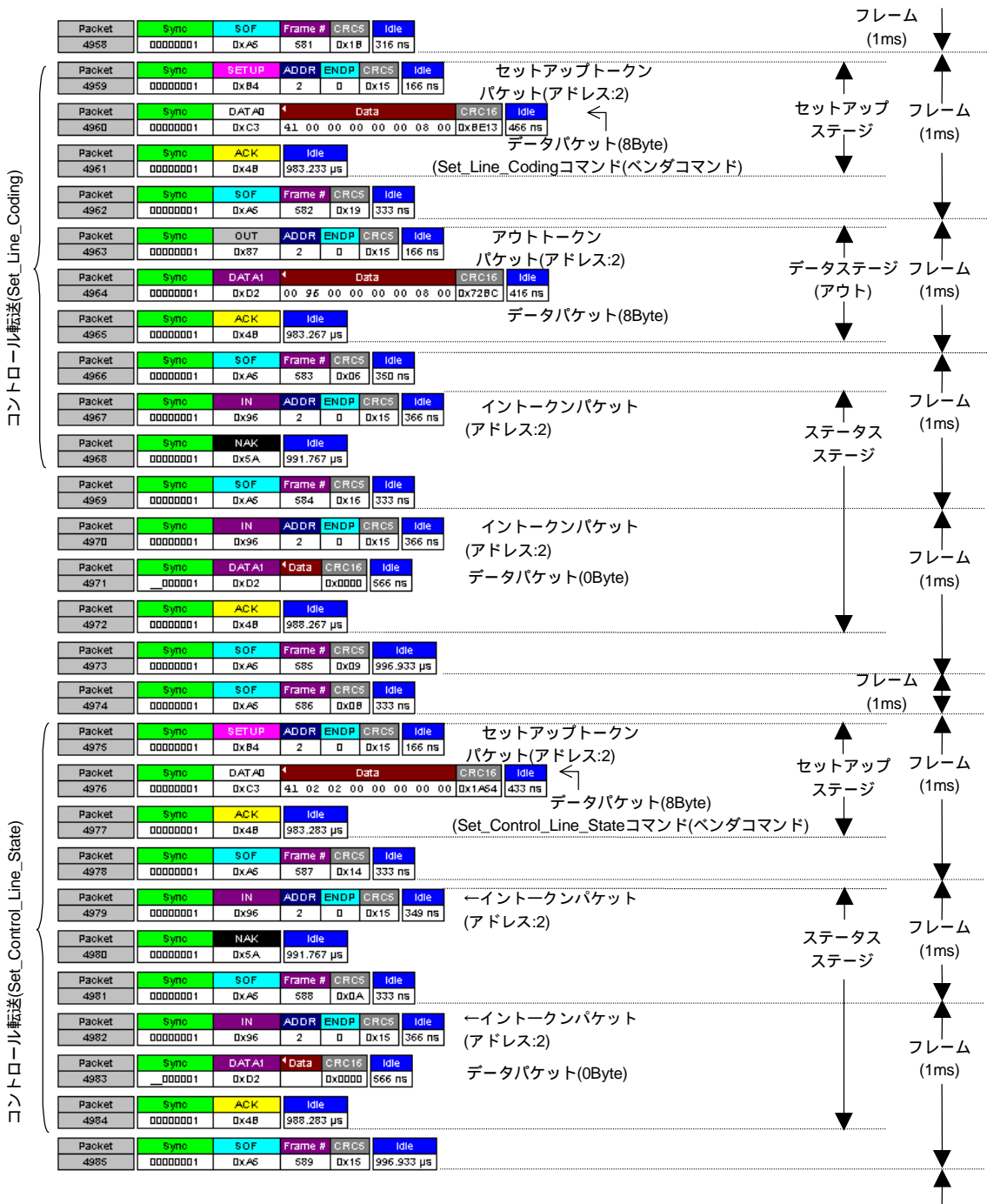
5. アナライザのデータ



5. アナライザのデータ

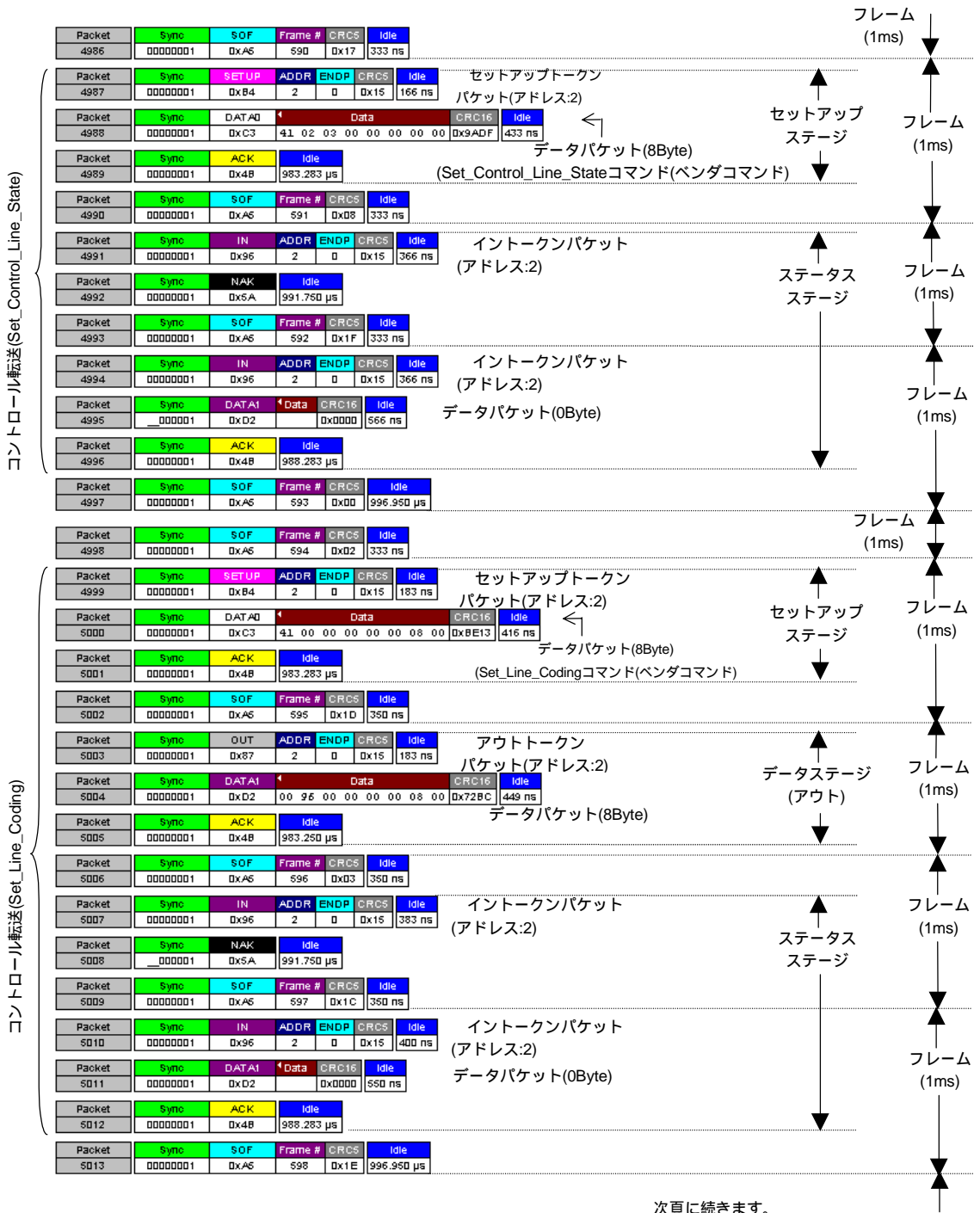


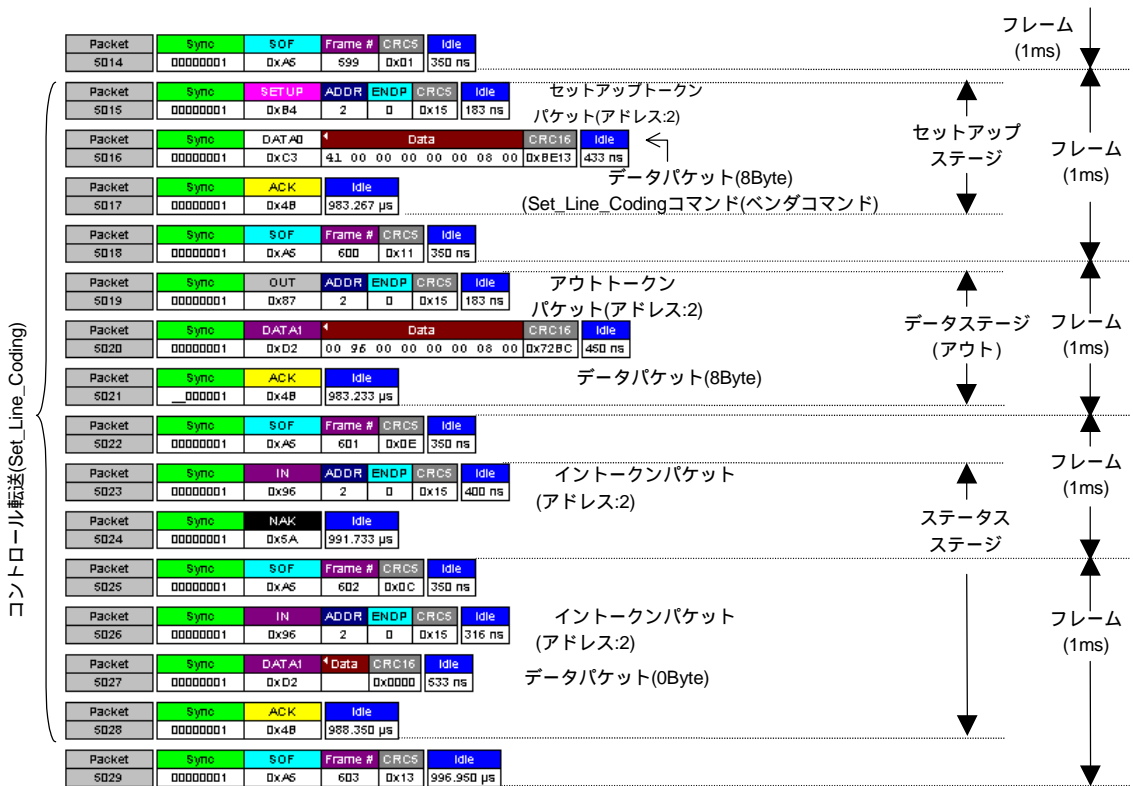
5. アナライザのデータ



次ラに続きます。

5. アナライザのデータ





以降、バルク転送があるまで定常状態となります。

図 5.2 バンダコマンド送信時のバルクアウト転送

H8S/2215 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

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

©株式会社 日立製作所 2002

H8S/2215 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



ルネサスエレクトロニクス株式会社
神奈川県川崎市中原区下沼部1753 〒211-8668

ADJ-502-086