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。

H8S ファミリ

ASSP M66592 USB Sample Firmware

要旨

本資料は，制御用 MCU (H8S/2218) と M66592 を使用した USB インタフェース制御用サンプルプログラムである「ルネサス汎用 ASSP M66592 USB Sample Firmware for H8S/2218」 (以降 USB-FW と記載) の取扱説明書です。

動作確認デバイス

H8S/2218

目次

H8S/2218 と M66592 の接続編

M66592 の Sample Fimware 編

H8S/2218 と M66592 の接続編

1. 概要

- 参考文献

1. H8S, H8/300 シリーズ　High-performance Embedded workshop3 ユーザーズマニュアル
2. H8S, H8/300 シリーズ　High-performance Embedded workshop3 チュートリアル
3. H8S, H8/300 シリーズ　C/C++コンパイラパッケージ アプリケーションノート
4. Solution Engine® H8S/2218 CPU ボード概説書
5. Universal Serial Bus Revision 2.0 specification
   http://www.usb.org/developers/docs/

1.1 USB-FW for H8S/2218 の特長

USB-FW for H8S/2218 は以下のような特長を所持しています。

- ペリフェラルを特定しない構成 (ユーザが個別に定義可能)
- USBCommandVerifier.exe (以降 USBCV と記載) にて接続確認可能 (USBCV は, http://www.usb.org/developers/developers/tools/ よりダウンロードできます)
- Bulk (IN/OUT) / Interrupt (IN/OUT) データ通信のサンプルプログラムを提供
- ファイルは機能ごとに分割 (表 1 ファイル構成一覧参照)
- H8S/2218 上で動作するよう調整しており, H8S/2218CPU ボード (Solution Engine®: MS2218CP01 (株) 日立超 LSI システムズ製), 信号中継基板 (お客様にて作成が必要です), H8S/2218 E10A-USB の組み合わせによる評価が可能です。

【注】 Solution Engine®**は (株) 日立超** LSI **システムズの登録商標です。**

1.2 開発目的

USB-FW for H8S/2218 は以下の目的で開発しました。

- M66592 を使用した USB 通信プログラムの開発を容易にする。
- M66592 制御に関する具体例による補足説明を行なう。

1.3 サービス概要

USB-FW for H8S/2218 が上位層 (ユーザプログラム層) に提供するサービスは以下のとおりです。

- M66592 初期化 (リセット, 発振制御, パイプ初期化, etc)
- リクエスト応答 (スタンダードリクエスト [USB Revision 2.0 specification])
- データ転送 (Bulk, Interrupt 転送: CPU アクセス)
- ステータス通知 (状態通知関数)
- リクエスト通知 (リクエスト通知関数)

1.4 ファイル構成一覧

USB-FW for H8S/2218 のファイル構成は，ルネサス汎用 ASSP M66592 USB Sample Firmware 本体ファイル (一部変更) ・追加ファイルと H8S/2218 用のワークスペースファイル (High-performance Embedded Workshop 3 にて生成) の 3 種類に分けられます。

表 1 ファイル構成一覧

ファイル構成		ファイル名	概要
USB-サンプル FW 本体 ファイル		changeep.c	ユーザアプリケーション処理
		classvender.c	クラス/ベンダリクエスト処理
		controlrw.c	コントロールリード/ライト処理
		dataio.c	データリード/ライト処理
		datatble.h	送受信用ユーザバッファ定義
		def592.h	M66592 レジスタアドレス/ビット定義
		def_ep.h	パイプ設定用データ定義
		descrip.h	ディスクリプタデータ定義
		extern.h	外部参照定義
		global.c	グローバル変数処理
		intrn.c	INTR, INTN, BEMP 割り込み処理
		usbsig.c	USB 信号処理
		libassp.c	USB ASSP レジスタ操作処理
		lib592.c	USB ASSP レジスタ操作処理
		libassp.h	USB ASSP レジスタ操作用定義
		macusr.h	ユーザマクロ定義
		status.c	内部ステータス操作処理
		stdreqget.c	スタンダードリクエスト処理
		stdreqset.c	スタンダードリクエスト処理
		typedef.h	変数型定義
		version.h	バージョン情報定義
	変更 ファイル	usbint.c	USB 割り込み処理
		main.c	擬似ユーザアプリケーション
		defusr.h	ユーザ設定定義
	追加 ファイル	2218S.H	H8S/2218 レジスタ定義ファイル
H8S/2218 用 HEW3 ワークスペース ファイル		dbsct.c	Setting of B, R Section
		resetprg.c	Reset Program
		sbrk.c	Program of sbrk
		sbrk.h	Header file of sbrk file
		stacksct.h	Setting of Stack area
		Fw592_H8S2218.hws (.hbp/.tws)	ワークスペースファイル
		Fw592_H8S2218.hwp (.pgs/.tps)	HEW プロジェクトファイル
		defaultSession.hsf	セッションファイル
		¥debug,¥relase	アブソリュートファイル 生成フォルダ

2. USB-FW for H8S/2218 開発環境

USB-FW for H8S/2218 は，下記開発環境にて開発・評価しました。

表 2　USB-FW for H8S/2218 開発・評価環境

区分	型番	名称	備考
ハードウェア	M3A-0038G01	M66592 評価ボード	
	MS2218CP01	H8S/2218 Solution Engine® (CPU ボード)	
		信号中継基板	お客様にて作成が必要です
	HS0005KCU01H	E10A-USB	エミュレータ
ソフトウェア		High-performance Embedded Workshop ver.3.0.06 (relase2)	開発環境
		H8S, H8/300H Standard Toolchain (V.6.0.3.0)	ツールチェイン
	HS0005KCU01SR	HDI	エミュレータソフト

2.1 ハードウェア構成

2.1.1 使用 CPU ボード

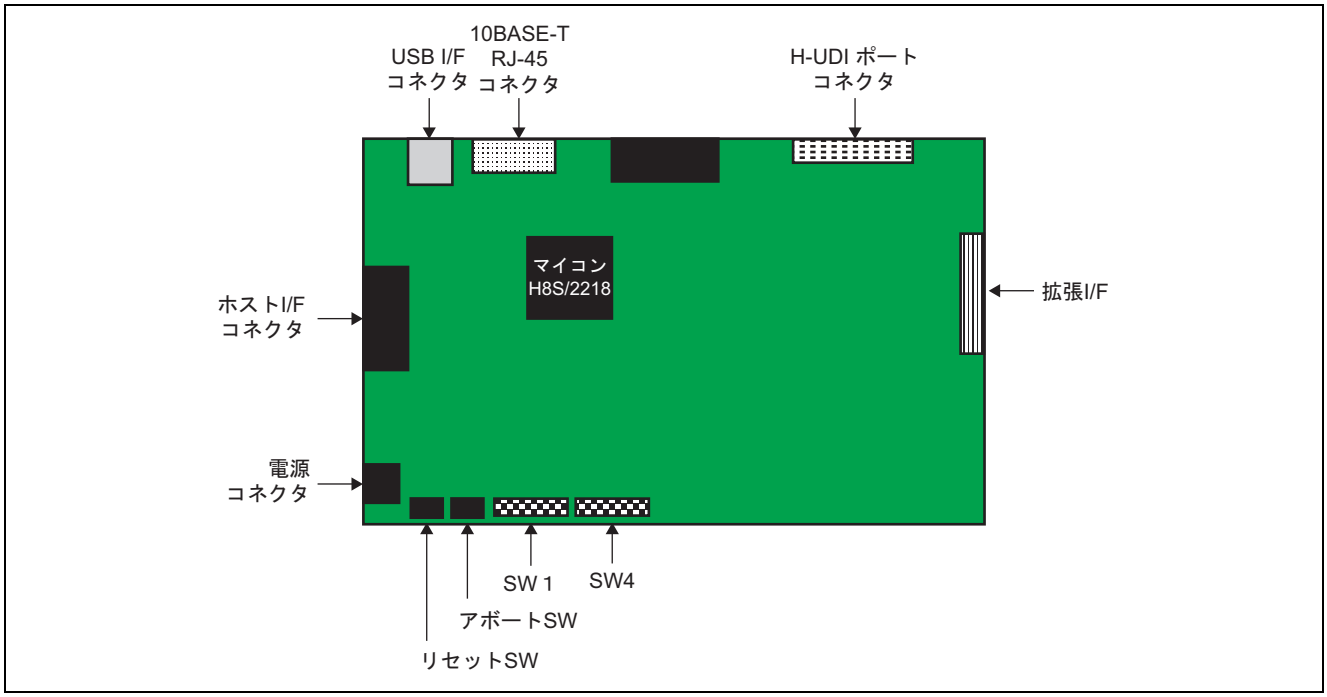


図 1 CPU ボード MS2218CP01

表 3 MS2218CP01 SW1 設定内容

SW	名称	機能	設定内容
SW1-1	MD0	MCU モード切り換え (MCU モード: 6)	ON
SW1-2	MD1		OFF
SW1-3	MD2		OFF
SW1-4	FEW	Flash 書き込み有効	OFF
SW1-5	EMLE	H-UDI 機能有効	OFF
SW1-6	—	未使用	OFF
SW1-7	—	未使用	OFF
SW1-8	—	未使用	OFF

【注】 Solution Engine® H8S/2218 CPU ボード取扱説明書をご参照ください。
SW の設定は，電源 OFF の状態で行なってください。

表 4　MS2218CP01 拡張 I/F (CN12) ピン配置図

H8S/2218CPU ボード　CN12 (拡張スロット)					
Pin	信号名	端子 No.	Pin	信号名	端子 No.
1	GND	—	2	ϕ	89
3	GND	—	4	D0	64
5	D1	65	6	D2	66
7	D3	67	8	GND	—
9	D4	68	10	D5	69
11	D6	70	12	D7	71
13	GND	—	14	D8	72
15	D9	73	16	D10	74
17	D11	75	18	GND	—
19	D12	76	20	D13	77
21	D14	78	22	D15	79
23	GND	—	24	3.3V	—
25	3.3V	—	26	GND	—
27	A0	10	28	A1	11
29	A2	12	30	A3	13
31	GND	—	32	A4	17
33	A5	18	34	A6	19
35	A7	20	36	GND	—
37	A8	37	38	A9	38
39	A10	39	40	A11	40
41	GND	—	42	A12	49
43	A13	50	44	A14	51
45	A15	52	46	GND	—
47	A16	1	48	A17	100
49	nCS1	27	50	nCS3	25
51	GND	—	52	nWAIT	95
53	3.3V	—	54	nRD	92
55	nIRQ0	6	56	nIRQ1	8
57	nIRQ2	97	58	nRES	58
59	GND	—	60	nHWR	93
61	nLWR	94	62	nAS	91
63	GND	—	64	3.3V	—

【注】 型名:　14-5015-064-102-861 (メスタイプ)
メーカ:　京セラエルコ

2.1.2 使用ユーティリティボード

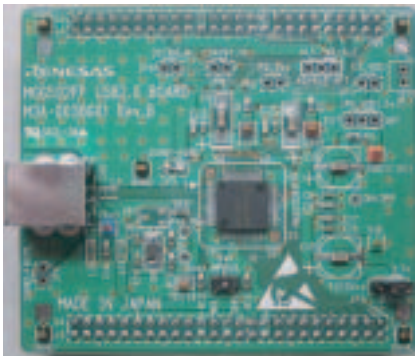


図 2　ユーティリティボード M3A-0038G01

表 5　M3A-0038G01 CN2 ピン配置図

M3A-0038G01 CN2					
Pin	16bit-sepa *1	16bit-mult *2	Pin	16bit-sepa *1	16bit-mult *2
1	GND	GND	2	D15	D15
3	D14	D14	4	D13	D13
5	D12	D12	6	D11	D11
7	D10	D10	8	D9	D9
9	D8	D8	10	GND	GND
11	D7	D7	12	D6	D6/AD6
13	D5	D5/AD5	14	D4	D4/AD4
15	D3	D3/AD3	16	D2	D2/AD2
17	D1	D1/AD1	18	D0	D0
19	GND	GND	20	GND	GND
21	**使用不可**	**使用不可**	22	**使用不可**	**使用不可**
23	WR1_N	WR1_N	24	VBUS	VBUS
25	EXIOVcc	EXIOVcc	26	EXIOVcc	EXIOVcc
27			28		
29	GND	GND	30	GND	GND
31			32		
33			34		
35			36		
37			38		
39			40		
41	SD7	SD7	42	SD6	SD6
43	SD5	SD5	44	SD4	SD4
45	SD3	SD3	46	SD2	SD2
47	SD1	SD1	48	SD0	SD0
49	GND	GND	50	GND	GND

【注】 1. 16bit-sepa: 16bit-SeparateBus use
2. 16bit-mult: 16bit-MultplexBus use

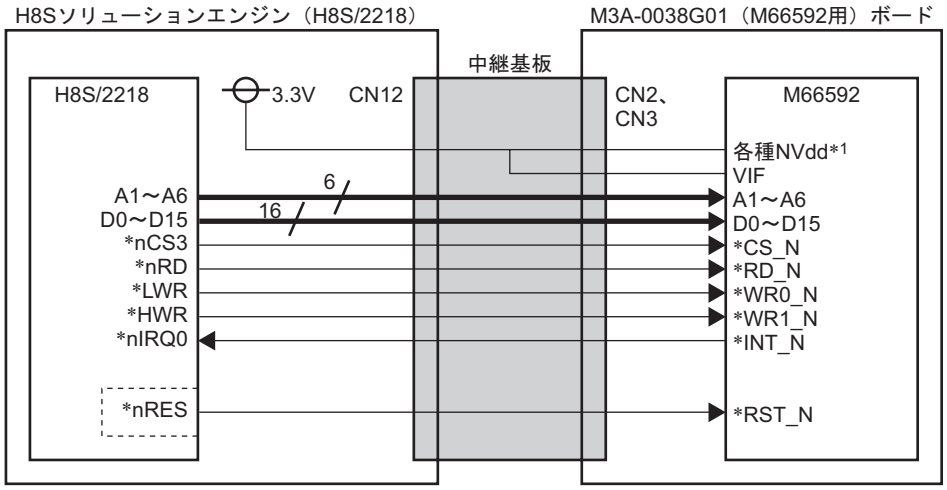
表 6　M3A-0038G01 CN3 ピン配置図

M3A-0038G01 CN3					
Pin	16bit-sepa *1	16bit-mult *2	Pin	16bit-sepa *1	16bit-mult *2
1	WR0_N	WR0_N	2	GND	GND
3	RD_N	RD_N	4	GND	GND
5	CS_N	CS_N	6	RST_N	RST_N
7	DREQ0_N	DREQ0_N	8	DACK0_N	DACK0_N
9	INT_N	INT_N	10	GND	GND
11	GND	GND	12	A1	**未使用**
13	A2	**未使用**	14	A3	**未使用**
15	A4	**未使用**	16	A5	**未使用**
17	A6	ALE	18	GND	GND
19	EXVcc	EXVcc	20	EXVcc	EXVcc
21	**未使用**	(JP7-ALE)	22		
23			24	SOF_N	SOF_N
25	DACK1_N/DSTB0_N	DACK1_N/DSTB0_N	26	DREQ1_N	DREQ1_N
27			28		
29	GND	GND	30	GND	GND
31	JP6-EXT (**外部** 1.5V **入力**)	JP6-EXT (**外部** 1.5V **入力**)	32		
33			34		
35	DACK1_N/DSTB0_N	DACK1_N/DSTB0_N	36	DEND0_N	DEND0_N
37			38		
39			40	DEND1_N	DEND1_N
41			42		
43			44		
45			46		
47			48		
49	GND	GND	50	GND	GND

【注】 1. 16bit-sepa: 16bit-SeparateBus use
2. 16bit-mult: 16bit-MultplexBus use

2.1.3 CPU ボードとユーティリティボードの接続

CPU ボードとユーティリティボードと中継基板により接続し，H8S/2218 対応 USB-FW の開発・評価を実施しています。



【注】*1 一部のVdd に関してはM3A-0038 にて1.5V に降圧して供給しています。

図 3 配線イメージ

2.1.4 システム構成

ユーザシステム (ターゲットシステム) は E10A-USB エミュレータを用いてホストコンピュータへ接続します。

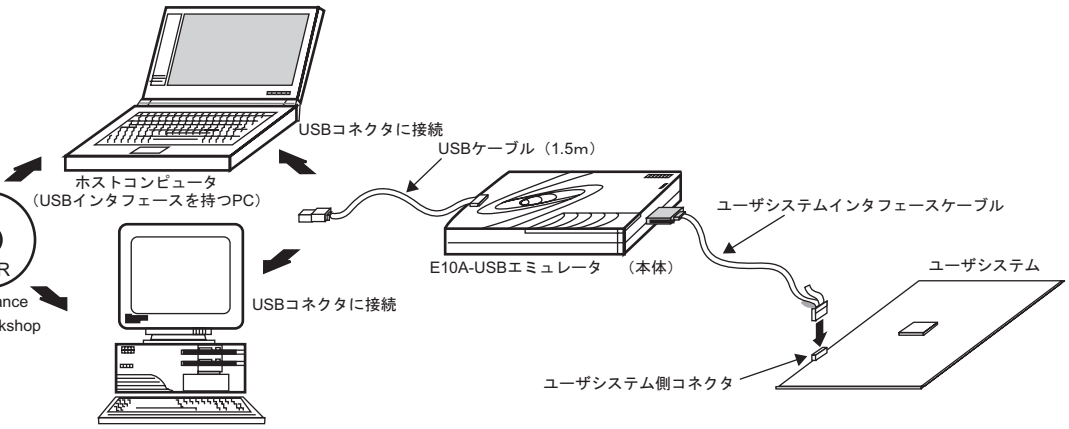


図 4 システム構成

2.2 ソフトウェア構成

USB-FW for H8S/2218 は High-performance Embedded Workshop 3 にてプロジェクトを作成し，以下の設定を施し開発・評価しています。

2.2.1 High-performance Embedded Workshop 3 に関する設定

(1) H8S/2218 用ワークスペースファイルの設定

H8S/2218 用ワークスペースファイルは High-performance Embedded Workshop 3 にて「新規プロジェクトワークスペースの作成」を指定し，表示される各入力画面の項目を設定し生成しています。

表 7　High-performance Embedded Workshop3 生成内容

入力画面	項目	選択内容	備考
	Application 選択	Application	
	ワークスペース名	Fw592_H8S2218	入力
	プロジェクト名	Fw592_H8S2218	入力
	CPU 種別	H8S, H8/300	
	ツールチェイン	Hitachi H8S, H8/300 standard	
1/9	Tool chain version	6.0.3.0	H8S, H8/300 C/C++ Library Generator (V.2.00.01)
			H8S, H8/300 C/C++ Compiler (V.6.00.03)
			H8S, H8/300 Assembler (V.6.01.00)
			Optimizing Linkage Editor (V.8.00.07)
	CPU Series	2000	
	CPU Type	Other	2218 が存在しません (CPU Type が Other の場合: CPU 情報と I/O レジスタ定義ファイルの作成が必要です)
2/9	動作モード	Advanced	
	アドレス空間	16Mbyte	
	ライブラリ生成方針	コードサイズ優先	
	スタック計算サイズ	ミディアム (2byte)	
3/9	□Use I/O Library	チェック無し	
	■USE HeapMemory	チェック	
	[Heap Size]	H'420	
	[generate main()Function]	None	CPU Type が Other 以外の場合: チェックしてください
	□I/O Register Dlfinition Files	チェック無し	
4/9	Library	[Disable all]をチェック	入力
5/9	StackPointer Address	H'00FFE800	
	Stack Size	H'200	
6/9	■Vector Defintionfiles	チェック	
7/9	□Target	すべてチェック無し	
8/9	ターゲット名	H8S/2218F E10A-USB SYSTEM (CPU2000)	
	コンフィギュレーション名	Debug_H8S_2218F_ E10A-USB SYSTEM (CPU2000)	
9/9	以下のソースファイルを生成します		完了ボタンを押して終了

(2) セクション設定

USB-FW for H8S/2218 の開発・評価にあたり必要となるセクション設定は以下となります。
(HEW - [Tools] - [オプション] - [H8S, H8/300 Standard Toolchain] - [最適化リンカ] - カテゴリ: セクションで設定)

表 8　セクション

アドレス	セクション名	説明
0x00000400	PResetPRG	**リセット関数**
	PIntPRG	**例外処理関数**
0x00001000	P	**プログラム領域**
	C	**定数領域**
	C$BSEC	_INITSCT **関数用**
	C$DSEC	_INITSCT **関数用**
	CIntPRG	**例外処理用**
	D	**初期化データ領域** (ROM)
0x00FFC000	B	**未初期化データ領域**
	R	**初期化データ領域** (RAM)
0x00FFE800	S	**スタック領域**

(3) CPU 情報作成

デバッグ用に CPU 情報を作成します。
(HEW - [Tools] - [オプション] - [H8S, H8/300 Standard Toolchain] - [最適化リンカ] - カテゴリ: ベリファイで作成)

表 9　CPU 情報

Device	Start	End
ROM	0x00000000	0x001FFFFF
RAM	0x00FFC000	0x00FFFFFF

なお，詳細は「H8S, H8/300 シリーズ　C/C++コンパイラパッケージ　アプリケーションノート」をご参照ください(USB-FW for H8S/2218 は H8S/2218 SolutionEngine を使用しているため上記の設定となっております)。

http://documentation.renesas.com/jpn/products/mpumcu/apn/rjj05b0558_h8s.pdf

2.2.2 H8S/2218 に関する設定

USB-FW for H8S/2218 は，ルネサス汎用 ASSP M66592 USB Sample Firmware 本体ファイル (Ver.1.00) に以下のような変更・追加を施し，H8S/2218 SolutionEngine 用に調整しています。

表 10 ルネサス汎用 ASSP M66592 USB Sample Firmware 追加・変更内容

NO	ファイル名	変更・追加内容	備考
1	main.c	delay_1ms(), delay_10us()を変更 CPU_init()を H8S/2218 用に設定	時間調整 バス・s/w 割り込み設定
2	Defusr.h	内容に合わせ変更	2.2.3 参照
3	Usbint.c	関数宣言方法変更 #pragma section IntPRG_interrupt (vect = 16) void usbint (void)	割り込み関数をベクタと共に宣言 (コンパイラ機能による設定)
4	2218S.H	H8S/2218 レジスタ定義ファイル追加	

(1) アドレスマップ

USB-FW for H8S/2218 にて使用したアドレスマップは下図のうち，H'000000～H'1FFFFF・H'780000～H'78007F・H'C00000～H'FFFFFF となります。

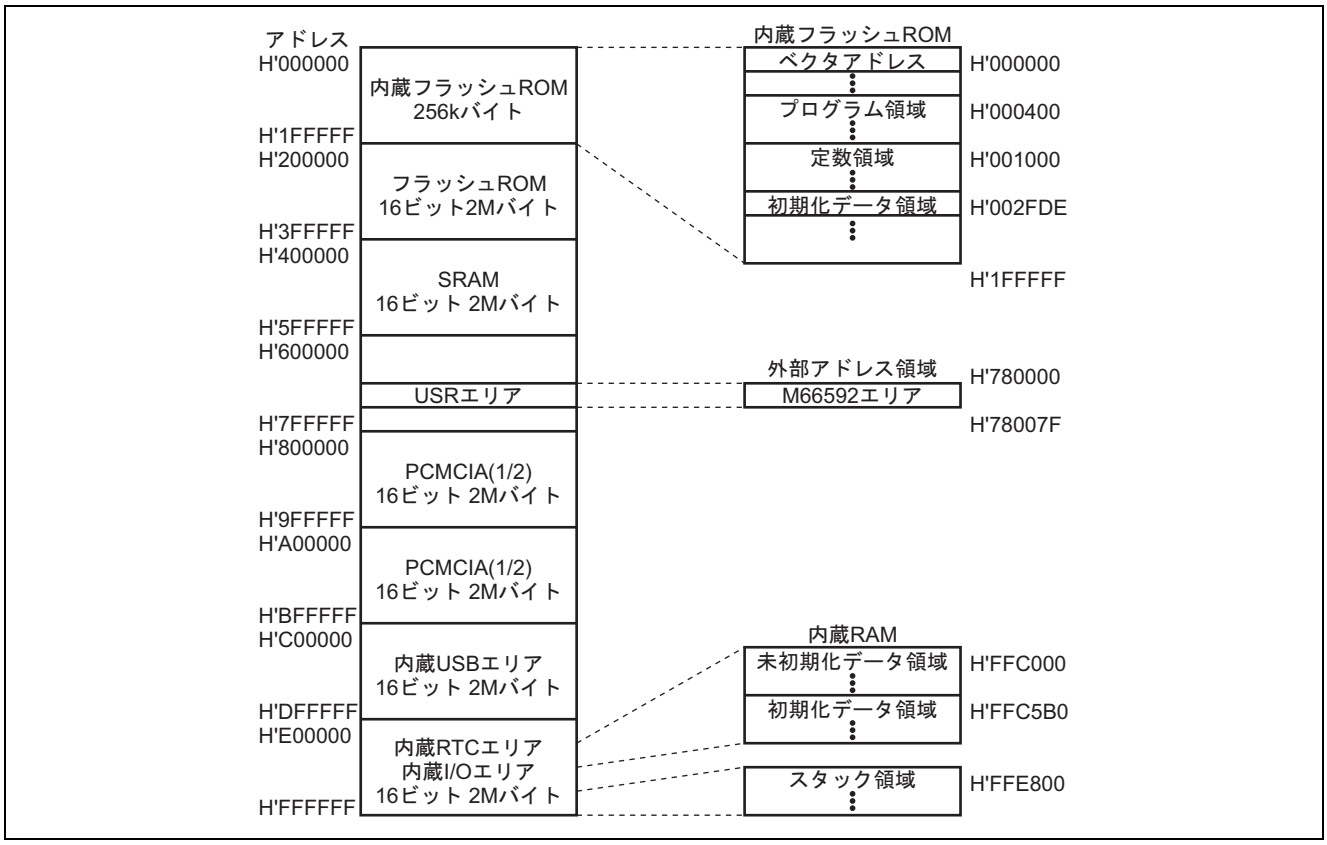


図 5 H8S/2218 SolutionEngine アドレスマップ

(2) バス幅・挿入ウェイト設定

USB-FW for H8S/2218 にて制御用 MCU に設定しているバス幅・挿入ウェイトは以下の設定となります。
(main.c: CPU_init()にて設定)

表 11　バス幅・ウェイト数

エリア No.	エリア名	バス幅	アクセスステート	ウェイトステート
CS3 (H'780000～H'78007F)	M66592 エリア	16Bit	3State	1State

2.2.3　USB-FW for H8S/2218 のユーザ定義内容

USB-FW for H8S/2218 は，H8S/2218 SolutionEngine 用に以下の設定を行なっています。

表 12　USB-FW for H8S/2218 ユーザ定義内容

No.	項目	設定内容	備考
1	低電力スリープ機能指定	#define　PCUT_MODE　PCUT_USE	低電力スリープ機能使用
2	自動クロック供給機能指定	#define　ATCKM_MODE　ATCKM_USE	-
3	FIFO のエンディアン指定	#define　FIFO_ENDIAN　BIG_ENDIAN	ビッグエンディアン設定 (H8S/2218 と M66592 の接続による設定事項)
4	I/O 電源指定	#define　VIF_SET　VIF3	3.3V 使用
5	M66592 アドレス	#define　USB_BASE　(0x780000)	H8S/2218 エリア 3 指定 (ソリューションエンジン使用による設定事項)
6	M66592 アドレスに対するポインタの型宣言	typedef　volatile U16　REGP;	near/far 宣言は H8S/2218 では不要
7	接続発振子の発振周波数	#define XIN XTAL24	24MHz 設定
8	リモートウェイクアップ指定	#define　RWUP_MODE　RWUP_NOT_USE	リモートウェイクアップ非使用設定

2.3　ご注意

USB-FW for H8S/2218 のリクエストは標準リクエストのみ対応しており，データ通信も USB-FW for H8S/2218 と擬似ユーザアプリケーション間で仮のインタフェースを用いて行なっています。

このため，クラス規定やベンダ固有リクエストへの応答が必要な場合をはじめ，通信速度，プログラム容量などを考慮する場合，さらにユーザインタフェースを個別に設定する場合には，お客様にてカスタマイズしてください。

USBCV に pass するためには, descrip.h にてベンダ ID･プロダクト ID を設定していただく必要があります。

【注】　USB-FW は，USB 通信動作を保証するものではありません。システムに適用される場合は，お客様における動作検証はもとより，多種多様なホストコントローラにおける接続確認を実施してください。

3. 内蔵 Flash 書き込み方法

H8S/2218 内蔵フラッシュメモリにユーザプログラムを書き込む方法について記述します。

3.1 SCI を用いた Flash 書き込み

CPU ボードの SW・JP を以下のように設定し，添付圧縮ファイルを使用します。

表 13 MS2218CP01 SW・JP 設定

SW・JP	名称	機能	設定内容	備考
SW1-1	MD0	動作モード切り換え (SCI ブートモード)	ON	
SW1-2	MD1		ON	
SW1-3	MD2		OFF	
SW1-4	FEW	Flash 書き込み有効	OFF	
SW1-5	EMLE	ポート機能有効	ON	
JP1	JP1	JP4 に TxD2 を出力	1-2 ピンショート	
JP2	JP2	JP5 に RxD2 を出力	1-2 ピンショート	
JP4	JP4	CN3 に TxD2 を出力	1-2 ピンショート	
JP5	JP5	CN3 に RxD2 を出力	1-2 ピンショート	

【注】 Solution Engine® H8S/2218 CPU ボード取扱説明書をご参照ください。
SW・JP の設定は，電源 OFF の状態で行なってください。

3.2 E10A-USB を用いた Flash 書き込み

CPU ボードの SW を以下のように設定し, High-performance Embedded Workshop 3 を立ち上げます。プロジェクトを開いた後，接続時に"Writing Flash memory"を選択します。

ダウンロードモジュールをダウンロードするとこにより，Flash への書き込みが実行されます。

表 14 MS2218CP01 SW 設定

SW・JP	名称	機能	設定内容	備考
SW1-1	MD0	MCU モード切り換え (MCU モード: 6)	ON	
SW1-2	MD1		OFF	
SW1-3	MD2		OFF	
SW1-4	FEW	Flash 書き込み有効	OFF	
SW1-5	EMLE	H-UDI 機能有効	OFF	

【注】 Solution Engine® H8S/2218 CPU ボード取扱説明書をご参照ください。
SW の設定は，電源 OFF の状態で行なってください。

ユーザプログラムを実行させるためには, 接続を解除し High-performance Embedded Workshop 3 を終了します。CPU ボードの SW を以下のように設定し，CPU ボードの電源を入れユーザプログラムを実行させます。

表 15　MS2218CP01 SW 設定

SW・JP	名称	機能	設定内容	備考
SW1-1	MD0	MCU モード切り換え (MCU モード: 6)	ON	
SW1-2	MD1		OFF	
SW1-3	MD2		OFF	
SW1-4	FEW	Flash 書き込み有効	OFF	
SW1-5	EMLE	ポート機能有効	ON	

【注】 Solution Engine® H8S/2218 CPU ボード取扱説明書をご参照ください。
SW の設定は，電源 OFF の状態で行なってください。

4. 制限事項

USB-FW for H8S/2218 には，以下の制限事項があります。

1. アイソクロナス転送は，正しく動作しない場合があります。
(動作状況によっては，CPU の処理が追いつかずパケットが抜けてしまうことがあります)
2. スプリットバスでの動作は，現在サポートしていません。
3. 同一エンドポイント番号を複数同時使用することは，現在できません。
例) パイプ 1 を BULK IN でエンドポイント 1 に指定している状態で，
パイプ 2 を BULK OUT でエンドポイント 1 に指定すると正しく動作しません。
4. DMA 転送処理は，対応していません。

1. 概要

- 参考文献

1. Universal Serial Bus Revision 2.0 specification
   http://www.usb.org/developers/docs/

1.1 USB-FW の特長

USB-FW は以下のような特長を所持しています。

1. 制御用 MCU およびペリフェラルを特定しない構成 (ユーザが個別に定義)

- USBCommandVerifier.exe (以降 USBCV と記載) にて接続確認可能
  (USBCV は，http://www.usb.org/developers/developers/tools/ よりダウンロードできます)
- Bulk (IN/OUT) / Interrupt (IN/OUT) データ通信のサンプルプログラムを提供
- ファイルは機能ごとに分割 (表 1 ファイル構成一覧参照)
- ユーザプログラムから M66592 レジスタの直接アクセスは不要

1.2 レイヤ

USB-FW は下図のレイヤで構成されます。

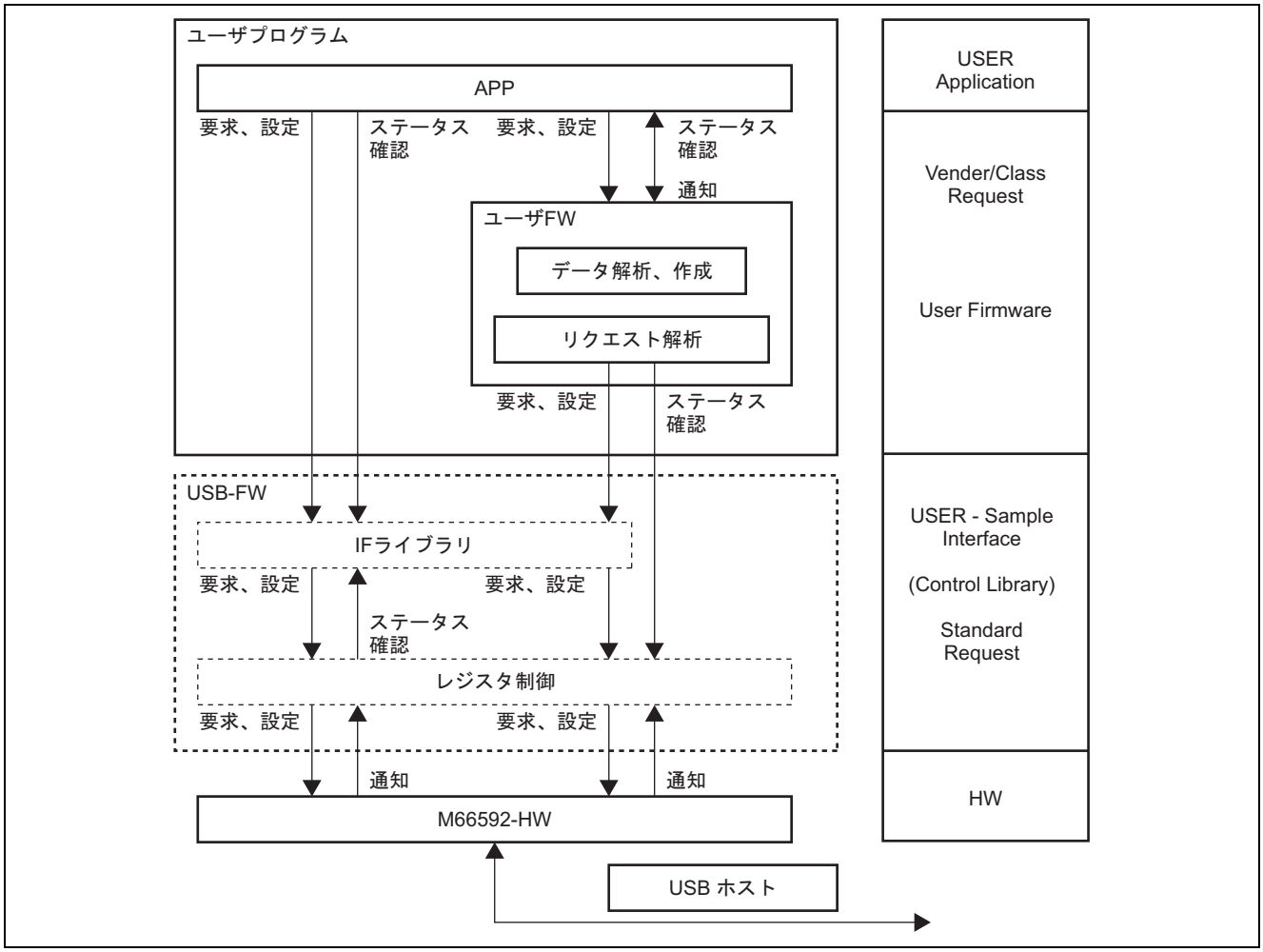


図 1 構成図

1.3 ファイル構成一覧

表 1 ファイル構成一覧

ファイル名	概要
changeep.c	ユーザアプリケーション処理
classvender.c	クラス/ベンダリクエスト処理
controlrw.c	コントロールリード/ライト処理
dataio.c	データリード/ライト処理
datatbl.h	送受信用ユーザバッファ定義
def592.h	M66592 用レジスタアドレス/ビット定義
def_ep.h	パイプ設定用データ定義
defusr.h	ユーザ設定定義
defusr.h	ディスクリプタデータ定義
extern.h	外部参照定義
global.c	グローバル変数処理
intrn.c	INTR, INTN, BEMP 割り込み処理
Usbsig.c	USB 信号処理
Libassp.c	USB ASSP レジスタ操作処理
lib592.c	USB ASSP レジスタ操作処理
libassp.h	USB ASSP レジスタ操作用定義
macusr.h	ユーザマクロ定義
main.c	擬似ユーザアプリケーション
status.c	内部ステータス操作処理
stdreqget.c	スタンダードリクエスト処理
stdreqset.c	スタンダードリクエスト処理
typedef.h	変数型定義
usbint.c	USB 割り込み処理
version.h	バージョン情報定義

1.4 開発目的

USB-FW は以下の目的で開発を行ないました。

- M66592 を使用した USB 通信プログラムの開発が容易になる。
- M66592 制御に関する具体例による補足説明を行なう。

1.5 サービス概要

USB-FW が上位層 (ユーザプログラム層) に提供するサービスは以下のとおりです。

- M66592 初期化 (リセット，発振制御，パイプ初期化，etc)
- リクエスト応答 (スタンダードリクエスト [USB Revision 2.0 specification])
- データ転送 (Bulk，Interrupt 転送: CPU アクセス)
- ステータス通知 (状態通知関数)
- リクエスト通知 (リクエスト通知関数)

1.6 概略フロー

USB-FW は，USB データ送受信を行なう制御関数を割り込みプログラムで構成しています。割り込みイベントは M66592 から制御用 MCU への外部割り込みによって発生します。

外部割り込みプログラムにて，割り込み要因を判別し，該当する処理を実行します。

(1) 特殊信号処理

Vbus 割り込み，レジューム割り込み，SOF 検出割り込み*，デバイスステート遷移割り込み

【注】 *: USB-FW **では，処理を実装していません。必要に応じて処理を作成してください。**

(2) コントロール転送処理

コントロール転送ステージ遷移割り込み, デバイスステート遷移割り込みをトリガにデータ転送を行ないます。

(3) パイプ転送処理

バッファエンプティ/サイズエラー割り込み，バッファノットレディ割り込み，バッファレディ割り込みをトリガにデータ転送を行ないます。

USB-FW では，USB 制御処理を USB 割り込みルーチン内で行なっているため，main 関数では，制御用 MCU および M66592 関連レジスタの初期設定を行なった後は, main 関数内の永久ループとなっています。

USB-FW の概略フロー図は次のとおりです。

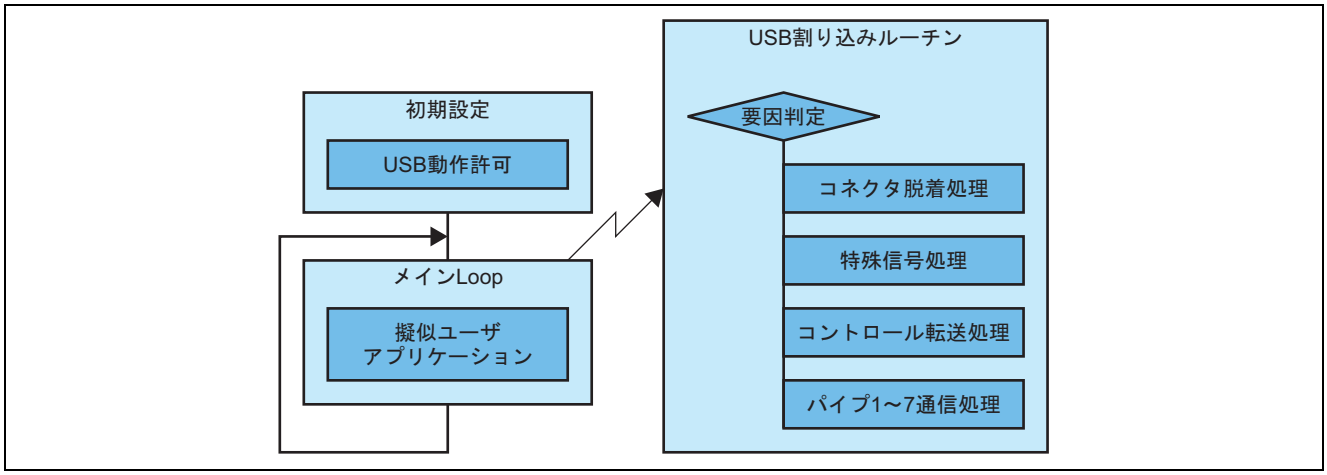


図 2 概略フロー図

2. USB-FW を動作させるには

USB-FW は，制御用 MCU およびペリフェラル初期化 ("main.c")，ユーザ定義情報 ("defusr.h") およびユーザ定義マクロ ("macusr.h") の各ファイルを変更することにより，通信が可能となり USBCV に pass することができます。

2.1 USB-FW の変更

USB-FW を動作させ，USBCV プログラムを pass するには，以下のプログラムファイルおよびヘッダファイルの改変が必要です。

(1) main.c 内の下記関数を改変する必要があります。
- 制御用 MCU の初期化 (CPUInit 関数)
- ペリフェラルの初期化 (PeripheralInit 関数)
- 制御用 MCU 割り込み許可 (enableINT 関数)
- 指定時間待ち関数の時間調整 (delay_1ms 関数，delay_10us 関数)
  ループ処理で指定時間のウェイトを行なっていますので，ご使用のシステムに合わせてループ回数を変更するなどの改変をして指定時間になるように調整してください。

(2) ユーザカスタマイズが必要なファイルがあります。
- descrip.h にて，ベンダ ID，プロダクト ID
  VenderID，ProductID
  (詳細は，「5. ユーザ定義情報」を参照してください)
- defusr.h にて，エンディアン指定，I/O 電圧，レジスタベースアドレス，far 領域，発振定数，割り込み関数指定，FIFO_ENDIAN，VIF_SET，USB_BASE，REGP，XIN，INTERRUPT 指定
  (詳細は，「6. ユーザ定義情報」を参照してください)

(3) ビルド時に必要
- セクション領域指定
- スタートアップルーチン作成

(4) その他
- 特殊信号処理 (FW 動作確認 (USBCV に pass) レベルでは不要)

2.2 ご注意

USB-FW は，制御用 MCU およびペリフェラルを特定しない汎用 FW です。リクエストは標準リクエストのみ対応しており，データ通信も USB-FW と擬似ユーザアプリケーション間で仮のインタフェースを用いて行なっています。このため，クラス規定やベンダ固有リクエストへの応答が必要な場合をはじめ，通信速度，プログラム容量などを考慮する場合，さらにユーザインタフェースを個別に設定する場合には，お客様にてカスタマイズしてください。

【注】 USB-FW は，USB 通信動作を保証するものではありません。システムに摘要される場合は，お客様における動作検証はもとより，多種多様なホストコントローラにおける接続確認を実施してください。

3. データ転送

USB-FW は，ホスト PC の USB 用ドライバとデータ転送用アプリケーションをお客様でご用意いただくことにより，ホストコンピューターデバイス間の簡易データ通信が行なえます (USB-FW のデバイス構成はディスクリプタ定義の項を参照してください)。

また，デバイス構成に必要な情報 (ディスクリプタ定義 ("descrip.h") ，パイプ定義 ("def_ep.h")) と擬似ユーザアプリケーション ("main.c") を変更することにより，お客様個別のシステム (ホスト PC 側システム) に対応した簡易データ通信を行なうことも可能です。

なお，データ通信はあくまでもお客様固有の機能仕様であり，転送方法，通信開始，終了などのリクエストおよびバッファ構成などは，お客様で個別改変していただく必要があります。

3.1 USB-FW の基本仕様

- USB-FW は，ユーザバッファと FIFO ポートレジスタ間のデータ転送を CPU アクセスにて実現しています (データの流れは，下図のようになっています)。
- ユーザバッファは，すべてのパイプに対して，512byte の領域を確保しています。
  (ユーザバッファ領域は，FIFO バッファ領域以上の領域を確保してください)
- ユーザバッファのサイズを変更することにより，パイプ FIFO バッファより大きいサイズのデータ転送が可能です。
- ホストへの送信データは，パイプ別で固定データを使用します。
- ユーザバッファアドレス通知関数 (DI_Start/DO_Start) は CPU/DMA アクセス共通関数となっています。
- データ送信/受信には，バッファレディ割り込みを使用し，データの送信/受信を始める際に各パイプのバッファレディ割り込みを許可します。

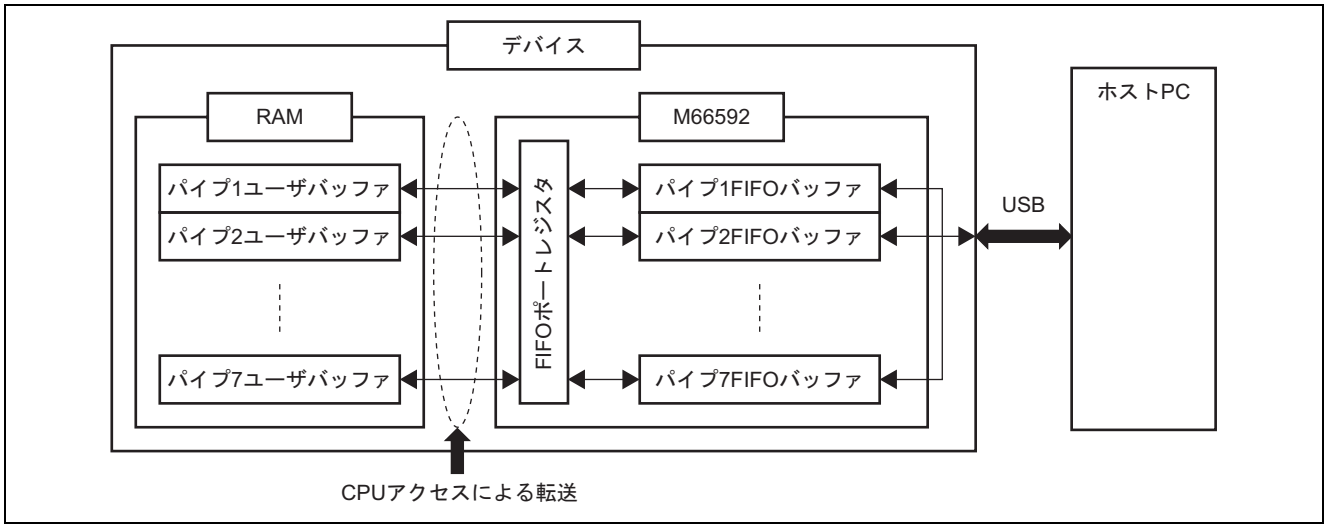


図 3 データフロー図

3.2 データ送信 (IN トークン対応) 動作

USB-FW は，ホスト PC へのデータ送信 (IN トークン対応) を以下の手順で行なっています。

1. ユーザバッファを使用可能かどうかチェックします (Buffer_Write_Data_Flag が DATA_NONE の場合使用可能)。
   ユーザバッファ使用禁止の場合は，何もせずに他のパイプの処理を行ないます。
2. 送信データがあるかどうかチェックします (Create_In_Data 関数の返値が DATA_NONE 以外なら送信データが有ります)。
   送信データが無い場合は，何もせずに他のパイプの処理を行ないます。
3. ユーザバッファアドレス設定，送信バイト数設定 (dtcnt) ，ユーザバッファ使用禁止設定 (Buffer_Write_Data_Flag に DATA_WAIT をセット) ，バッファレディ割り込み許可を行ないます。 (DI_Start 関数)
4. バッファレディ割り込みが発生すると，ユーザバッファ内のデータを FIFO ポートレジスタに書き込みます。
   — 送信バイト数 (dtcnt) ＞ FIFO バッファサイズ の場合
   FIFO バッファサイズ分データを書き込み, 送信バイト数から FIFO バッファサイズを減算 (dtcnt = dtcnt - FIFO バッファサイズ) します。
   — 送信バイト数 (dtcnt) ≦ FIFO バッファサイズ の場合
   送信バイト数分データを書き込み，バッファレディ割り込みの禁止，ユーザバッファ使用許可設定 (Buffer_Write_Data_Flag に DATA_NONE をセット) します。

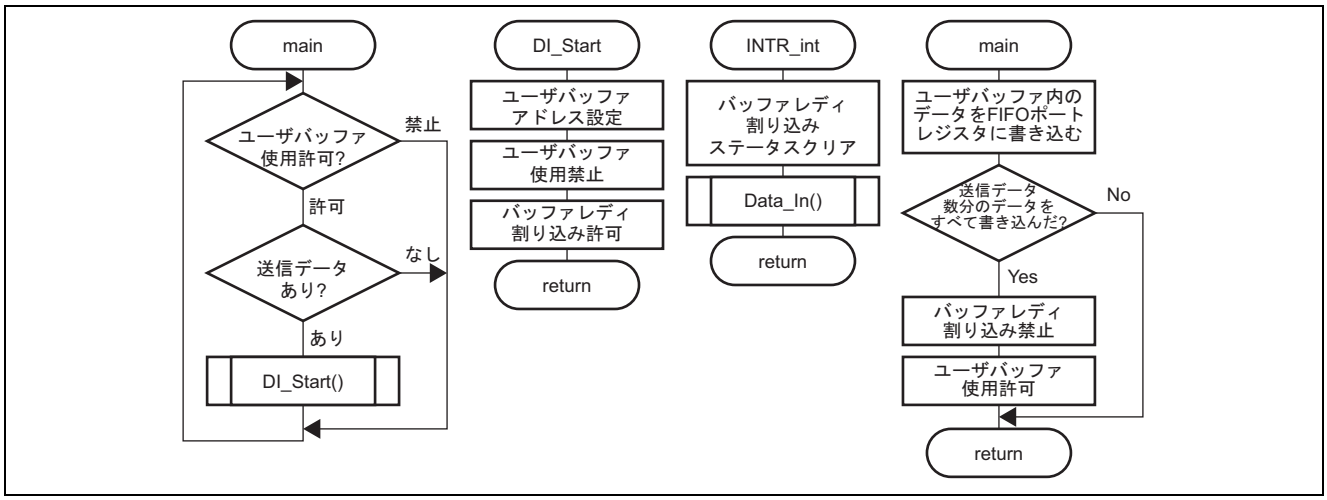


図 4　制御フロー

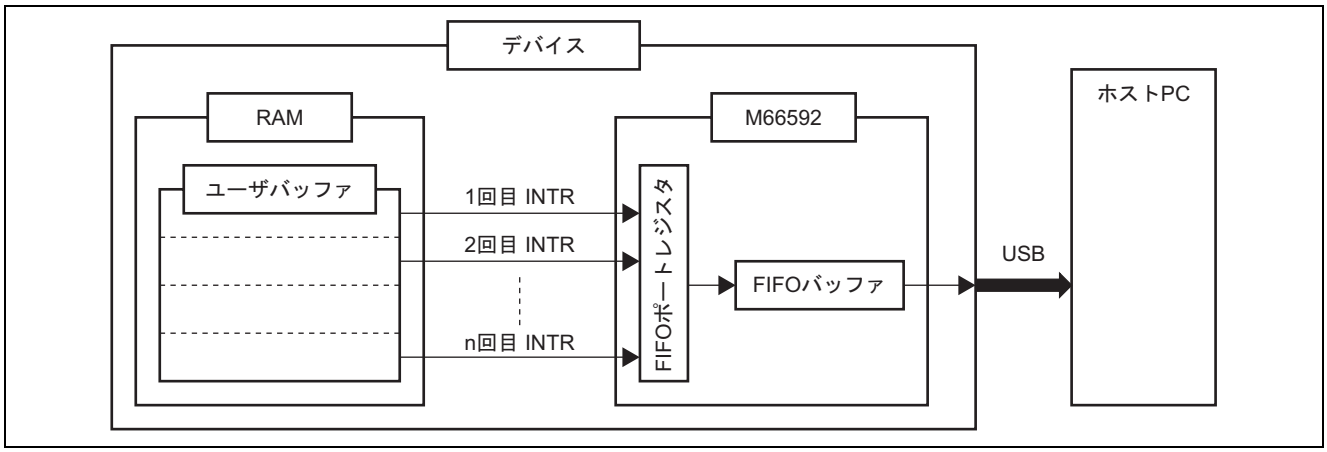


図 5　データフロー図

3.3 データ受信 (OUT トークン対応) 動作

USB-FW は，ホスト PC からデータ受信 (OUT トークン対応) を以下の手順で行なっています。

1. ユーザバッファを使用可能かどうかチェックします (Buffer_Read_Data_Flag が DATA_NONE の場合使用可能)。
   — ユーザバッファ使用禁止の場合，ユーザバッファ読み出し許可かチェックします。
   (Buffer_Read_Data_Flag が DATA_OK の場合，読み出し可能)
   ユーザバッファ読み出し許可の場合，ユーザバッファ使用許可設定 (Buffer_Read_Data_Flag に DATA_NONE をセット) します。
   ユーザバッファ読み出し禁止の場合，何もせずに他のパイプの処理を行ないます。
   — ユーザバッファ使用許可の場合，ユーザバッファアドレス設定，受信バイト数設定 (dtcnt)，ユーザバッファ使用禁止設定 (Buffer_Read_Data_Flag に DATA_WAIT をセット)，バッファレディ割り込み許可を行ないます (DO_Start 関数)。
2. バッファレディ割り込みが発生すると，FIFO ポートレジスタからデータを読み出しユーザバッファに書き込みます。
   — 受信バイト数 (dtcnt) ＞ FIFO バッファサイズの場合
   FIFO バッファサイズ分データを読み出し，受信バイト数から FIFO バッファサイズを減算 (dtcnt = dtcnt - FIFO バッファサイズ) します。
   — 受信バイト数 (dtcnt) ≦ FIFO バッファサイズの場合
   受信バイト数分データを読み出し，バッファレディ割り込みの禁止，ユーザバッファ使用許可設定 (Buffer_Read_Data_Flag に DATA_OK をセット) します。

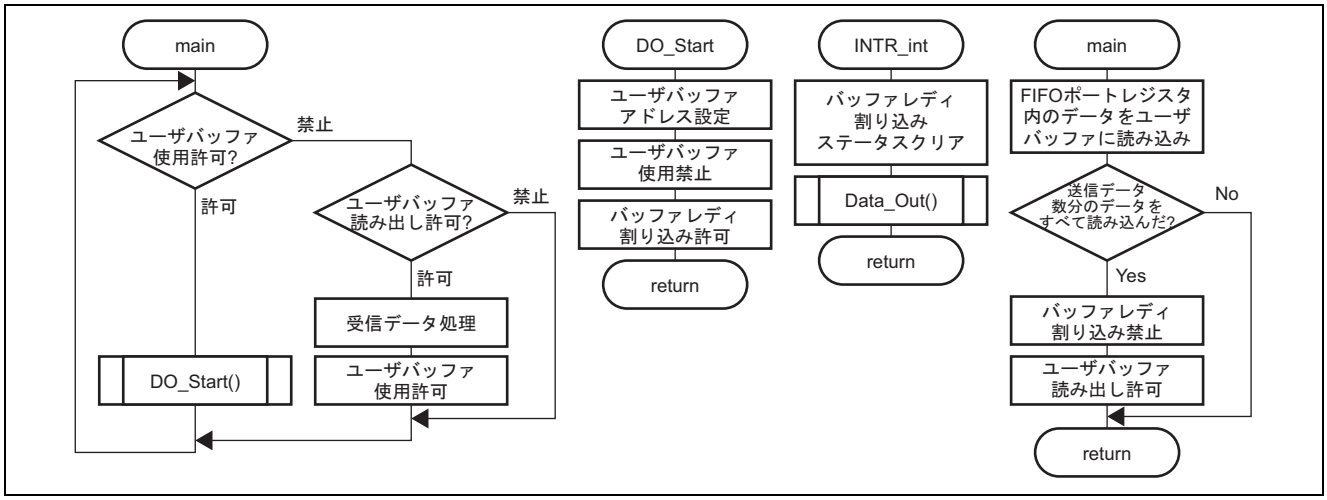


図 6　制御フロー

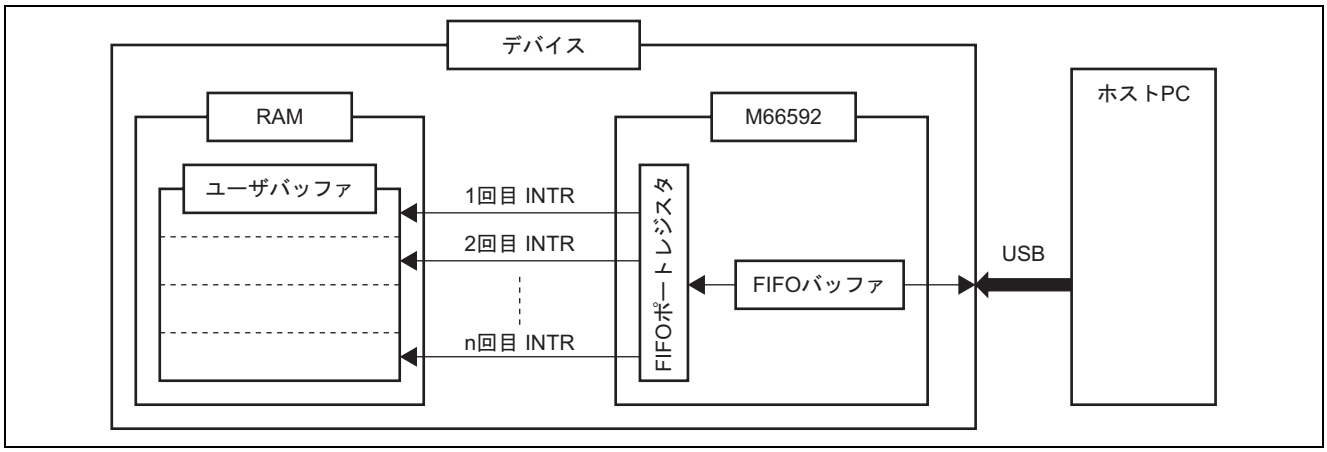


図 7　データフロー

3.4 データ転送用パイプ設定

USB 通信では，Set_Configuration / Set_Interface などのリクエストにより，データ通信用パイプ設定を変更する必要があります。

USB-FW は上記リクエスト受信時に，変更する構成もしくはインタフェースにて使用するパイプテーブルインデックス番号をディスクリプタテーブルより検索し，パイプ定義に従ってパイプコンフィギュレーションレジスタの自動設定を行ないます。

(1) パイプ設定

構成番号，インタフェース番号，代替設定値を引数に"Esrch"関数を呼び出します。
続けて，構成番号を引数に"resetEP"関数を呼び出します。

void Esrch (構成番号，インタフェース番号，代替設定値) ;
void resetEP (構成番号);

"Esrch"関数は, 指定された構成のインタフェースで使用する全パイプについてパイプ定義のインデックス番号を検索し設定します。

"resetEP"は, "Esrch"で見つかったすべてのパイプに対するパイプコンフィギュレーションレジスタの再設定を行ないます。パイプ仕様を変更する場合は，変更中の不正データ入力もしくは入力エラーに対応するため，既存のパイプ通信を禁止する必要があります。このため，resetEP では該当するパイプ割り込みを禁止しています。

(2) ディスクリプタテーブル

USB-FW では仮のディスクリプタ定義を使用しております。ホスト側 Windows ドライバおよびアプリケーションにあわせ，お客様固有のディスクリプタ定義 ("descrip.h") を作成してください。
なお, ディスクリプタ定義を変更する場合は, 同時にパイプ定義 ("def_ep.h") も変更してください。また，必要に応じて擬似ユーザアプリケーション (Change_Config / Change_Interface 関数) を変更してください。

【注】 ディスクリプタ定義とパイプ定義には，同じ順番でパイプ情報を定義してください。(詳細は，パイプ定義の項を参照してください。)

(3) Change_Config / Change_Interface 関数

Set_Configuration / Set_Interface リクエスト受信時に USB-FW より呼び出されます。ユーザアプリケーションに応じて必要な処理がある場合は，処理を追加してください。

3.5 改変時の注意事項

- ユーザバッファはマックスパケット (連続送受信時は FIFO バッファ) サイズ以上としてください。
- 通信確認を容易に行なうため，各ユーザバッファにダミー領域を設けています。
- ディスクリプタ定義 ("descrip.h") 変更時は，パイプ定義 ("def_ep.h") の変更も合わせて実施してください。
- ユーザバッファの使用方法 (もしくは構成) を変更した場合は，ライブラリ関数の変更を必要とする場合があります。

3.6 ユーザバッファ仕様

- ユーザバッファアクセス回数をカウントできるよう領域確保しています。
- ユーザバッファは，パイプごとに異なったデータを初期定義 (値設定) しています。

```
typedef struct {
   U16 size;                                    /* バッファサイズ */
   U16 count;                                   /* バッファアクセスカウンタ */
   U8 Dummy[12];                                /* データ領域位置調整 */
   U8 buff[EP1_DATA_SIZE];                      /* データバッファ領域 */
} ep_buff1;

ep_buff1 EP_Buff1 = {
  EP1_DATA_SIZE, 0,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,
  0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0A,0x0B,0x0C,0x0D,0x0E,
  0x0F,0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1A,0x1B,0x1C,0x1D,
  0x1E,0x1F,0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2A,0x2B,0x2C,
  0x2D,0x2E,0x2F,
                          :
                          :
ep_buff2 EP_Buff2 = {
  EP2_DATA_SIZE, 0,0x02,0x02,0x02,0x02,0x02,0x02,0x02,0x02,0x02,0x02,0x02,0x02,
0x0F,0x0E,0x0D,0x0C,0x0B,0x0A,0x09,0x08,0x07,0x06,0x05,0x04,0x03,0x02,0x01,
0xFF,0x1F,0x1E,0x1D,0x1C,0x1B,0x1A,0x19,0x18,0x17,0x16,0x15,0x14,0x13,0x12,
0x11,0x10,0x2F,0x2E,0x2D,0x2C,0x2B,0x2A,0x29,0x28,0x27,0x26,0x25,0x24,0x23,
0x22,0x21,0x20,
                          :
                          :
                          :
                          :
                          :
ep_buff7 EP_Buff7 = {
  EP7_DATA_SIZE, 0,0x07,0x07,0x07,0x07,0x07,0x07,0x07,0x07,0x07,0x07,0x07,0x07,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,0x10,
0x10,0x10,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,
0x20,0x20,0x20,
```

4. クラス/ベンダリクエスト

USB-FW は，ホスト PC の Windows ドライバ，アプリケーションおよびデバイス側ユーザ FW をお客様でご用意頂くことにより，クラス/ベンダリクエスト応答が行なえます。USB-FW では，データステージにおけるデータ通信の基本構造が，各パイプにおけるデータ通信と同じ構造で設計された関数の入り口のみ用意しています。

4.1 基本仕様

- ユーザバッファアドレス通知関数として以下の関数を用意しています。
  CR_Start (コントロールリード ユーザバッファアドレス通知関数)
  CW_Start (コントロールライト ユーザバッファアドレス通知関数)
- データリード/ライト関数として以下の関数を用意しています。
  Control_Read (コントロールリード データライト関数)
  Control_Write (コントロールライト データリード関数)
- ユーザバッファは，データステージで送受信するデータ容量以上の領域を確保してください。
  (1 回のコントロール転送で転送するデータ容量以上のユーザバッファサイズが必要)

4.2 詳細仕様

- USB-FW とユーザ FW 間でユーザバッファを設け，USB-FW は，ユーザバッファと FIFO ポートレジスタ間のデータ転送を CPU アクセスにて実現します。
  →ユーザアプリケーションでは CR_Start または CW_Start 関数を使用してください。
- データステージの転送方向は，INTR_int，BEMP_int 関数で判定しています。
- 送受信時は，割り込みを使用してデータの継続データ転送を行ないます。
  →Control_Read/Control_Write 関数にて，継続データ転送を行ないます。
- ホストへの送信データ (Control_Read 対応) 作成はユーザ FW で行なってください。
  作成案 1: 割り込み内のリクエスト判定で作成 (標準リクエスト同様)
  作成案 2: アプリケーションで作成 (割り込みからユーザアプリケーションに受信通知)
- ホストからの受信データ (Control_Write 対応) 解析はユーザ FW で行なってください。
  作成案 1: 割り込み内のリクエスト判定で作成 (標準リクエスト同様)
  作成案 2: アプリケーションで作成 (割り込みからアプリケーションに受信通知)

4.3 コントロールリード (IN 方向) ユーザ FW インタフェース例

クラス/ベンダリクエストのユーザ FW 作成例は，以下のとおりです。

1. コントロールリードデータステージ (ClassTrans1 関数) で CR_Start 関数を呼ぶ
2. バッファエンプティ/サイズオーバ割り込みが発生 (BEMP_int 関数)
   Control_Read 関数で送信データを FIFO バッファに転送する。

【注】 これは作成例です。実際は，お客様の仕様に合わせて作成してください。

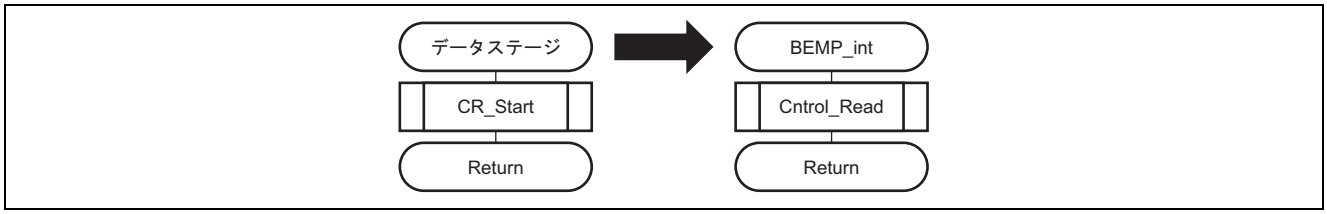


図 8 送信フロー

【特記事項】

- 送信データの作成終了後に，送信開始通知 (CR_Start 関数) を呼んでください。
- 送信開始通知 (CR_Start 関数) では，ユーザバッファアドレス，データサイズを通知してください。
- ユーザバッファはコントロールリードデータステージで送信するデータサイズ (wLength) 以上の容量を確保してください。
- USB 割り込み関数のコントロールリードステータスステージ遷移にてコントロール転送完了を通知 (Control_End 関数) してください。
- ユーザ FW は送信データ作成を示すため，リクエスト種別 (コントロールリードセットアップステージ情報) を記憶している必要があります。

4.4 コントロールライト (OUT 方向) ユーザ FW インタフェース例

クラス/ベンダリクエストのユーザ FW 作成例は，以下のとおりです。

1. コントロールライトデータステージ (ClassTrans2 関数) で CW_Start 関数を呼ぶ
2. ホスト PC よりデータを受信するとバッファレディ割り込みが発生 (INTR_int 関数) Control_Write 関数で受信データをユーザバッファに転送する。
3. コントロールライトステータスステージ (ClassTrans5 関数) でユーザバッファ内の受信データを処理する。

【注】 これは作成例です。実際は，お客様の仕様に合わせて作成してください。

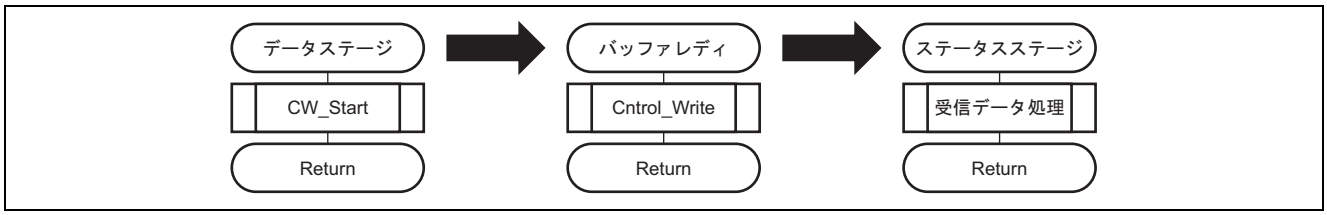


図 9 受信フロー

【特記事項】

- 受信開始通知 (CW_Start 関数) では，バッファアドレス，データサイズを通知してください。
- ユーザバッファはコントロールライトデータステージで受信するデータサイズ (wLength) 以上の容量を確保してください。
- 受信データの解析終了後は，USB-FW にコントロール転送完了を通知 (Control_End 関数を使用) してください。

5. ユーザ定義情報

USB-FW は, 汎用 FW として記述されておりますので, ユーザ定義情報ファイルを書き換えることにより, お客様固有の実行ファイルを作成することが可能です。

お客様のシステムに合わせて，以下の 12 項目を変更してください。

【注】 制御 MCU によっては，他の定義が必要な場合もあります。

1. ベンダ ID
2. プロダクト ID
3. 低電力スリープ機能 (PCUT) 指定
4. 自動クロック供給機能 (ATCKM) 指定
5. FIFO のエンディアン指定
6. I/O 電源指定
7. M66592 のアドレス
8. M66592 のアドレスに対するポインタの型宣言
9. M66592 に接続する発振子の発振周波数
10. 割り込みベクタの宣言
11. 多重割り込み許可
12. リモートウェイクアップ指定

5.1 ベンダ ID (descrip.h)

お客様のシステムに合わせて，ベンダ ID を指定してください。

例) 0x1234 に設定する場合

```
#define VendorID 0x1234
```

5.2 プロダクト ID (descrip.h)

お客様のシステムに合わせて，プロダクト ID を指定してください。

例) 0x5678 に設定する場合

```
#define ProductID 0x5678
```

5.3 低電力スリープ機能 (PCUT) 指定 (defusr.h)

M66592 の低電力スリープ機能を使用する/しないを指定してください。

PCUT_USE を指定した場合，下記の ATCKM_MODE の指定に関係無く自動クロック供給機能は使用する (ATCKM_USE) になります。

例) 使用する場合

```
#define PCUT_MODE PCUT_USE
```

5.4 自動クロック供給機能 (ATCKM) 指定 (defusr.h)

低電力スリープ機能を使用しない場合に，M66592 の自動クロック供給機能を使用する/しないを指定してください。

例) 使用する場合

```
#define ATCKM_MODE ATCKM_USE
```

5.5 FIFO のエンディアン指定 (defusr.h)

M66592 の FIFO ポートにアクセスするためのバイトエンディアンを指定してください。

例) リトルエンディアンに設定する場合

```
#define FIFO_ENDIAN LITTLE_ENDIAN
```

5.6 I/O 電源指定 (defusr.h)

M66592 の I/O 端子の電圧を指定してください。

例) 3.3V の場合

```
#define VIF_SET VIF3
```

5.7 M66592 のアドレス (defusr.h)

M66592 にアクセスするための基準アドレスを指定してください。

すべてのレジスタは，基準アドレスからのオフセットでアドレス指定を行なっています。

例）0x8000 番地に設定する場合

```
#define USB_BASE (0x8000)
```

5.8 M66592 のアドレスに対するポインタの型宣言 (defusr.h)

M66592 にアクセスするためのポインタの型を指定してください。

near / far の宣言が必要な MCU に対応するために，M66592 上の各レジスタは，"REGP"型でキャストしています。制御用 MCU のタイプに合わせて，near/far のどちらか片方をコメントアウトしてください。

例) near/far 宣言が必要な MCU で M66592 が near 領域に割り付けられた場合

```
typedef volatile U16 REGP;
/* typedef volatile far U16 REGP; */
```

5.9 接続発振子の発振周波数 (defusr.h)

M66592 に接続する発振子の発振周波数を 3 種類の発振タイプから選択してください。

例) 24MHz の発振子を使用する場合

```
#define XIN XTAL24
```

5.10 割り込みベクタの宣言 (defusr.h)

割り込み処理関数の宣言が必要な MCU には，USB-FW が提供する USB 通信用割り込みの宣言をする必要があります。必要に応じてコメントを解除してください。また，宣言が"#pragma"で無い場合は，宣言も変更してください。

例) 割り込み処理関数 (usbint) に pragma 宣言が必要な場合

```
#pragma INTERRUPT usbint
```

5.11 多重割り込み許可 (defusr.h)

USB 割り込み処理中に多重割り込みを許可する場合には割り込み許可をするための命令を設定してください。

M3A-0033 と KD308 をご使用の場合，USB 割り込み処理が長いために KD308 が不正終了する場合があります。この場合には多重割り込みを許可してください。

例) M16C/80 で多重割り込みを許可する場合

```
#define MULTI_INT_ENABLE() asm ("fset i")
```

5.12 リモートウェイクアップ指定 (defusr.h)

リモートウェイクアップを使用する/しないを，指定してください。

例) リモートウェイクアップを使用しない場合

```
#define RWUP_MODE RWUP_NOT_USE
```

6. ユーザ定義マクロ ("macusr.h")

M66592 は USB Revision 2.0 specification 準拠で設計されており, 各レジスタへのアクセスはリトルエンディアンで行なう必要があります。USB-FW は，制御用 MCU のエンディアンが M66592 と異なっている場合をも想定した汎用 FW として記述されています。レジスタアクセスおよび FIFO レジスタアクセスをマクロ記述しているため，ユーザ定義マクロヘッダファイルを換えることにより，お客様固有の実行ファイルを作成することが可能です。お客様のシステムに合わせて，レジスタおよび FIFO レジスタへのアクセスマクロを変更してください。レジスタアクセスマクロは以下の 4 種 9 マクロで構成されます。

1. レジスタおよび FIFO データレジスタ読み出し/書き込みマクロ
2. レジスタビットセット/クリア/修正マクロ
3. ステータスレジスタビットクリアマクロ
4. ステータスレジスタビットセットマクロ

6.1 レジスタおよび FIFO データレジスタ読み出し/書き込みマクロ

レジスタおよび FIFO ポートレジスタのデータを読み書きするマクロです。

制御 MCU と M66592 の結線は，メモリ－FIFO ポートレジスタ間の DMA 転送が正常に行なえるように配線する必要があります。

制御 MCU のエンディアン指定方法は，「5.5 FIFO のエンディアン指定 (defusr.h) 」を参照してください。

```
#define USBRD_FF( r, v )     do{ ( v ) = ( r ); } while(0)
#define USBWR_FF( r, v )     do{ ( r ) = ( v ); } while(0)
#define USBRD( r, v )        do{ ( v ) = ( r ); } while(0)
#define USBWR( r, v )        do{ ( r ) = ( v ); } while(0)
```

【注】制御 MCU－メモリ－M66592 の配線はシステム構成を十分に考慮の上設定してください。

6.2 レジスタビットセット/クリア/修正マクロ

レジスタのビットセット，クリア，修正を行なうマクロです。

各マクロは RMW (Read Modify Write) 命令に従った記述で，先のレジスタおよび FIFO ポートレジスタ読み出し/書き込みマクロを使用しています。

【注】 本マクロは，お客様固有に変更する必要はありません。
本ビットクリアマクロにてステータスレジスタのビットクリアは行なわないでください。ステータスレジスタのビットクリアには，ステータスレジスタビットクリアマクロを使用してください。

```
/* set bit(s) of USB register       */
/* r : USB register                 */
/* v : value to set                 */
#define USB_SET_PAT( r, v )         do{ register U16 tmp; ¥
                                            USBRD( r, tmp ); ¥
                                            tmp |= (v); ¥
                                            USBWR( r, tmp ); }while(0)
/* reset bit(s) of USB register    */
/* r : USB register                */
/* m : bit pattern to reset       */
#define USB_CLR_PAT( r, m )        do{ register U16 tmp; ¥
                                           USBRD( r, tmp ); ¥
                                           tmp &= (~(m)); ¥
                                           USBWR( r, tmp ); }while(0)
/* modify bit(s) of USB register   */
/* r : USB register                */
/* v : value to set                */
/* m : bit pattern to modify       */
#define USB_MDF_PAT( r, v, m )     do{ register U16 tmp; ¥
                                            USBRD( r, tmp ); ¥
                                            tmp &= (~(m)); ¥
                                            tmp |= v; ¥
                                            USBWR( r, tmp ); }while(0)
```

6.3 ステータスレジスタビットクリアマクロ

ステータスレジスタのビットクリアを行なうマクロです。本マクロは RMW (Read Modify Write) 命令に従った記述で, 先のレジスタおよび FIFO ポートレジスタ読み出し/書き込みマクロを使用しています。また, 本マクロは RMW 命令により，命令実行中に変化したステータス消去を回避するためのマクロです。

【注】 本マクロは，お客様固有に変更する必要はありません。
"1"書き込み無効のステータスレジスタにのみ使用してください。

```
/* reset bit(s) of USB status     */
/* r : USB register               */
/* m : bit pattern to reset      */
#define USB_CLR_STS( r, m )    USBWR( r, (~(m)) )
```

6.4 ステータスレジスタビットセットマクロ

ステータスレジスタのビットセットを行なうマクロです。

本マクロは，先のレジスタ書き込みマクロを使用しています。また，本マクロはすべてのビットに"1"を書き込むことにより，命令実行中に変化したステータス消去を回避するためのマクロです (内部クロック停止状態での VBUSINT クリアなどで"1"を書き込む場合にご使用ください)。

【注】 本マクロは，お客様固有に変更する必要はありません。
"1"書き込み無効のステータスレジスタにのみ使用してください。

```
/* set bit(s) of USB status    */
/* r : USB register            */
/* m : dummy                   */
#define USB_SET_STS( r, m )   USBWR( r, 0xffff )
```

7. パイプ定義 ("def_ep.h")

M66592 は，パイプに対する各種設定を，コンフィギュレーションレジスタによって行ないます。USB-FW は，Set_Configuration / Set_Interface などの標準リクエストによる，構成および代替変更に伴うパイプ設定変更を想定した汎用 FW として記述されています。各状態におけるパイプ情報 (使用方法) をデータテーブルとしてヘッダファイル上に構成し，パイプ定義ヘッダファイル ("def_ep.h") を書き換えることにより，お客様固有の実行ファイルを作成することが可能です。

パイプに対する各種設定は，FULL スピード用 (EPtbl_Full_n) と HIGH スピード用 (EPtbl_Hi_n) の 2 つが必要です。

お客様のシステムに合わせて，パイプ定義を変更してください。

【注】 パイプ定義を変更する場合，パイプ定義に合わせてディスクリプタ定義 (descrip.h) も変更してください。

デフォルトコントロールパイプ定義は，以下の 2 項目 (U16×2) で構成されます。

1. CFIFO ポート選択 (0x1E 番地)
2. DCP コンフィギュレーションレジスタ (0x5C 番地)

パイプ 1～7 定義は，以下の 5 項目 (U16×5) で構成されます。

1. パイプウインドウ選択レジスタ (0x64 番地)
2. パイプコンフィギュレーションレジスタ (0x66 番地)
3. パイプバッファ指定レジスタ (0x68 番地)
4. パイプマックスパケットサイズレジスタ (0x6A 番地)
5. パイプ周期制御レジスタ (0x6C 番地)

7.1 デフォルトコントロールパイプ定義

デフォルトコントロールパイプ定義は，テーブルで構成されています。なお，USB-FW でサンプルとして記述されているデフォルトコントロールパイプ情報テーブルは以下のようになっています。

例)

```
const U16 DCPtbl[] = {
   /* CFIFOSEL (0x1E) */
      MBW_16,                                   ←定義項目１
   /* DCPCFG (0x5C) */ CNTMD                    ←定義項目２
};
```

7.1.1 デフォルトコントロールパイプ定義項目 1

CFIFO ポート選択レジスタに設定する値を指定します。

以下の項目を設定してください。

- FIFO ポートアクセスビット幅: 8 ビット幅のと "MBW_8", 16 ビット幅のとき"MBW_16"を指定してください。
  例) 16 ビット幅のとき
  `MBW_16,`

7.1.2 デフォルトコントロールパイプ定義項目 2

DCP コンフィギュレーションレジスタに設定する値を指定します。

以下の項目を設定してください。

- 連続送受信モード: 連続受信の時は"CNTMD"を指定してください。
  例) 連続受信のとき
  CNTMD

【注】 **デフォルトコントロールパイプのマックスパケットサイズの指定方法は，「**8. **ディスクリプタ定義** ("descrip.h") **」を参照ください。**

7.2 パイプ 1～7 定義

各パイプ定義はディスクリプタ定義と同様に構成ごとにテーブルが構成されており，関連付けられているインタフェース，代替設定順に記載します。なお，USB-FW でサンプルとして記述されているパイプ定義は以下の構成になっています。各パイプ定義項目は，次項以降で説明します。

例)

```
/* Configuration 1 */
const U16 EPtbl_Full_1[] = {                                    ←フルスピード用
    /* Interface 1-0-0 */
        /* Endpoint 1-0-0-0 */

        /* Pipe Window Select Register (0x64) */
            PIPE1,                                              ←パイプ定義項目１
        /* Pipe Configuration Register (0x66) */
            BULK | DBLB | CNTMD | DIR_IN | EP1,                 ←パイプ定義項目２
        /* Pipe Buffer Configuration Register (0x68) */
            BUF_SIZE(512) | 6,                                  ←パイプ定義項目３
        /* Pipe Maxpacket Size Register (0x6A) */
            64,                                                 ←パイプ定義項目４
        /* Pipe Cycle Configuration Register (0x6C) */
            OFF | 0,                                            ←パイプ定義項目５
                            :
                            :
                            :
};

/* Configuration 1 */
const U16 EPtbl_Hi_1[] = {                                      ←ハイスピード用
    /* Interface 1-0-0 */
        /* Endpoint 1-0-0-0 */

        /* Pipe Window Select Register (0x64) */
            PIPE1,
        /* Pipe Configuration Register (0x66) */
            BULK | DBLB | CNTMD | DIR_IN | EP1,
        /* Pipe Buffer Configuration Register (0x68) */
            BUF_SIZE(512) | 6,
        /* Pipe Maxpacket Size Register (0x6A) */
            512,
        /* Pipe Cycle Configuration Register (0x6C) */
            OFF | 0,
                            :
                            :
                            :
};
```

7.2.1 パイプ定義項目 1

パイプウインドウ選択レジスタに設定する値を指定します。

以下の項目を設定してください。

パイプ選択: 選択パイプ (PIPE1～PIPE7) を指定してください。

例) パイプ 1 のとき

```
PIPE1,
```

7.2.2 パイプ定義項目 2

パイプコンフィギュレーションレジスタの設定を指定します。

以下の項目を設定してください。

- 転送タイプ: BULK，INT，ISO のいずれか一つを指定してください。
- ダブルバッファモード: ダブルバッファにする場合"DBLB"，シングルバッファにする場合"OFF"を指定してください。
- 連続送受信モード: 単送受信モードは"OFF"，連続送受信モードは"CNTMD"を指定してください。
- 転送方向: IN 方向のときは"DIR_IN"，OUT 方向のときは"DIR_OUT"を指定してください。
- エンドポイント番号: エンドポイント番号 (EP1～EP15) を指定してください。
  例) バルク転送，ダブルバッファ，連続送受信，IN 方向，EP1 のとき

```
BULK | DBLB | CNTMD | DIR_IN | EP1,
```

7.2.3 パイプ定義項目 3

パイプバッファ指定レジスタの設定を指定します。

以下の項目を設定してください。

- バッファサイズ: 64 バイト単位で PIPE バッファサイズを指定してください。
- バッファ先頭番号: バッファの先頭番号を指定してください。
  例) バッファサイズ 512 バイト，バッファ先頭番号 6 のとき

```
BUF_SIZE (512) | 6,
```

7.2.4 パイプ定義項目 4

パイプマックスパケットサイズレジスタの設定を指定します。

以下の項目を設定してください。

- マックスパケットサイズ: パイプのマックスパケットサイズを指定してください。
  例) マックスパケットサイズ 64 のとき

```
64,
```

7.2.5 パイプ定義項目 5

パイプ周期制御レジスタの設定を指定します。

以下の項目を設定してください。

- アイソクロナス IN 転送バッファフラッシュ: フラッシュする場合"IFIS"，フラッシュしない場合"OFF"を指定してください。
- インターバルエラー検出間隔: インターバル値を指定してください。
  例) バッファフラッシュしない，インターバル設定値 0 のとき

```
OFF | 0,
```

8. ディスクリプタ定義 ("descrip.h")

USB-FW は，汎用 FW として記述されていますので，ディスクリプタ定義ヘッダファイルを書き換えることにより，お客様固有のディスクリプタ定義でデバイス構成を行なうことが可能です。

ディスクリプタ定義は以下の 4 種類で構成されます。

【注】 各ディスクリプタの詳細は USB Revision 2.0 specification **の** Chapter9 **を参照ください。**
ディスクリプタ定義を変更する場合，ディスクリプタ定義に合わせてパイプ定義 (def_ep.h) **も変更してください。**

1. Standard Device Descriptor
   USB-FW では，下記テーブルで定義しています。
   U8 DeviceDescriptor[]
2. Device Qualifier Descriptor
   USB-FW では，下記テーブルで定義しています。
   U8 QualifierDescriptor[]
3. Configuration/Other_Speed_Configuration/Interface/Endpoint
   USB-FW では，下記テーブルで定義しています。
   U8 Configuration_Full_1[]
   U8 Configuration_Hi_1[]
4. String Descriptor
   USB-FW では，下記テーブルで定義しています。
   U8 StringDescriptor_tbl0[]
   U8 StringDescriptor_tbl1[]
   U8 StringDescriptor_tbl2[]
   U8 StringDescriptor_tbl3[]
   U8 StringDescriptor_tbl4[]
   U8 StringDescriptor_tbl5[]

8.1 ディスクリプタの作成

USB-FW は，前頁の 1～3 のテーブルを使用し，現在の M66592 の動作モードによりプログラム中で "Configuration_Full_n[1]", "Configuration_Hi_n[1]"に"DT_CONFIGURATION"または "DT_OTHER_SPEED_CONFIGURATION" をコピーし，ディスクリプタを RAM 上に作成します。

プログラム中で変更する部分は，"SOFTWARE_CHANGE"と書いてあります。

作成の流れは，以下のとおりです。

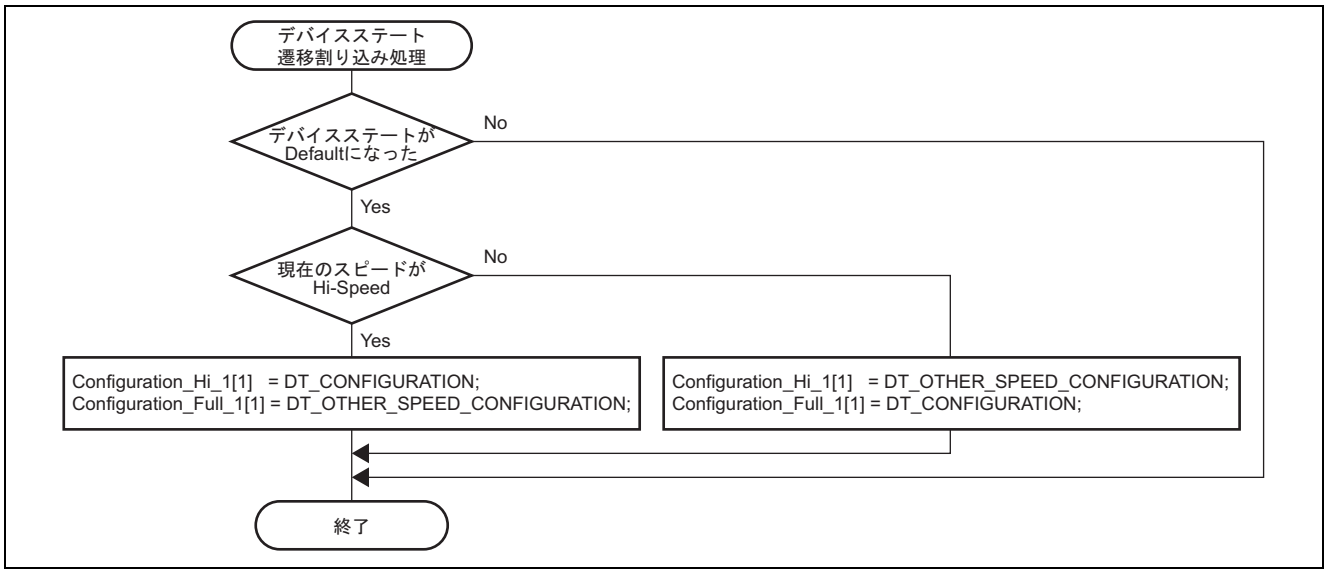


図 10 ディスクリプタの作成

8.2 デフォルトコントロールパイプの設定

USB-FW は，デスクリプタを RAM 上に作成した後 M66592 のデフォルトのコントロールパイプのためレジスタ設定を行ないます。

M66592 の以下のレジスタを設定する必要があります。

1. CFIFO ポート選択レジスタ (0x1E 番地)
2. DCP コンフィギュレーションレジスタ (0x5C 番地)
3. DCP マックスパケットサイズレジスタ (0x5E 番地)

上記のうち，1～2 は，定数テーブル"DCPtbl[]"のデータをレジスタに設定します。詳しくは「7. パイプ定義」を参照ください。

上記 3 は，RAM 上に作成されたテーブル"DeviceDescriptor[7]"のデータをレジスタに設定しますので，使用するマックスパケットサイズを設定してください。

8.3 USB-FW サンプルディスクリプタ構成

USB-FW のディスクリプタ構成は，以下のように定義されています。

```
/*
 * |--- Configuration 1
 * |      |--- Interface 1-0-0
 * |      |      |--- Endpoint 1-0-0-0
 * |      |      |--- Endpoint 1-0-0-1
 * |      |      |--- Endpoint 1-0-0-2
 * |      |      |--- Endpoint 1-0-0-3
 * |      |      |--- Endpoint 1-0-0-4
 * |      |      |--- Endpoint 1-0-0-5
 * |      |      |--- Endpoint 1-0-0-6
 */
```

9. 低電力スリープ機能 (PCUT)

USB-FW は，下記の 3 種類の処理サンプルが記述されています。

1. 低電力スリープ機能を使用する場合 (自動クロック供給機能を使用する)
2. 低電力スリープ機能を使用しない場合 (自動クロック供給機能を使用する)
3. 低電力スリープ機能を使用しない場合 (自動クロック供給機能を使用しない)

低電力スリープ機能を使用する/しないは，defusr.h 内の PCUT_MODE の宣言を変更してください。

9.1 低電力スリープ機能を使用する場合

defusr.h 内の PCUT_MODE の宣言を PCUT_USE にしてください。

この場合，サスペンド時，デタッチ時にはクロックを停止し，低電力スリープ機能を使用します。

クロックの起動は，ハードウェアで行ないます。

9.2 低電力スリープ機能を使用しない場合 (自動クロック供給機能を使用する)

defusr.h 内の PCUT_MODE の宣言を PCUT_NOT_USE にしてください。

defusr.h 内の ATCKM_MODE の宣言を ATCKM_USE にしてください。

この場合，サスペンド機能またはデタッチ機能を使用するときにはクロックを停止します。

クロックの起動は，ハードウェアで行ないます。

9.3 低電力スリープ機能を使用しない場合 (自動クロック供給機能を使用しない)

defusr.h 内の PCUT_MODE の宣言を PCUT_NOT_USE にしてください。

defusr.h 内の ATCKM_MODE の宣言を ATCKM_NOT_MODE にしてください。

この場合，サスペンド機能またはデタッチ機能を使用するときにはクロックを停止します。

クロックの起動は，ソフトウェアで行なってください。

10. 制限事項

USB-FW には，以下の制限事項があります。

1. アイソクロナス転送は，正しく動作しない場合があります。
   (動作状況によっては，CPU の処理が追いつかずパケットが抜けてしまうことがあります)
2. スプリットバスでの動作は，現在サポートしていません。
3. 同一エンドポイント番号を複数同時に使用することは，現在できません。
   例）パイプ 1 を BULK IN でエンドポイント１に指定している状態で，
   パイプ 2 を BULK OUT でエンドポイント１に指定すると正しく動作しません。
4. ＤＭＡ転送処理は，現在サポートしていません。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2005.08.12	—	初版発行

安全設計に関するお願い

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

本資料ご利用に際しての留意事項

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