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。**

アプリケーション・ノート

V850ES/JG3-H, V850ES/JH3-H

32ビット・シングルチップ・マイクロコントローラ

USB プリンタ・クラス ドライバ編

V850ES/JG3-H
- μPD70F3760
- μPD70F3761
- μPD70F3762
- μPD70F3770

V850ES/JH3-H
- μPD70F3765
- μPD70F3766
- μPD70F3767
- μPD70F3771

資料番号　R01AN0002JJ0100 (U20303JJ1V0AN00)（第1版）
発行年月　April 2010

〔メ　モ〕

MINICUBEは，NECエレクトロニクス株式会社の登録商標です。

Windows, Windows XP，およびWindows Vistaは，米国Microsoft Corporationの米国およびその他の国における登録商標または商標です。

PC/ATは，米国IBM社の商標です。

CMOSデバイスの一般的注意事項

(1) 入力端子の印加波形 : 入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOSデバイスの入力がノイズなどに起因して，VIL（MAX.）からVIH（MIN.）までの領域にとどまるような場合は，誤動作を引き起こす恐れがあります。入力レベルが固定な場合はもちろん，VIL（MAX.）からVIH（MIN.）までの領域を通過する遷移期間中にチャタリングノイズ等が入らないようご使用ください。

(2) 未使用入力の処理 : CMOSデバイスの未使用端子の入力レベルは固定してください。未使用端子入力については，CMOSデバイスの入力に何も接続しない状態で動作させるのではなく，プルアップかプルダウンによって入力レベルを固定してください。また，未使用の入出力端子が出力となる可能性（タイミングは規定しません）を考慮すると，個別に抵抗を介してVDDまたはGNDに接続することが有効です。資料中に「未使用端子の処理」について記載のある製品については，その内容を守ってください。

(3) 静電気対策 : MOSデバイス取り扱いの際は静電気防止を心がけてください。MOSデバイスは強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には，当社が出荷梱包に使用している導電性のトレーやマガジン・ケース，または導電性の緩衝材，金属ケースなどを利用し，組み立て工程にはアースを施してください。プラスチック板上に放置したり，端子を触ったりしないでください。また，MOSデバイスを実装したボードについても同様の扱いをしてください。

(4) 初期化以前の状態　電源投入時，MOSデバイスの初期状態は不定です。電源投入時の端子の出力状態や入出力設定，レジスタ内容などは保証しておりません。ただし，リセット動作やモード設定で定義している項目については，これらの動作ののちに保証の対象となります。リセット機能を持つデバイスの電源投入後は，まずリセット動作を実行してください。

(5) 電源投入切断順序　内部動作および外部インタフェースで異なる電源を使用するデバイスの場合，原則として内部電源を投入した後に外部電源を投入してください。切断の際には，原則として外部電源を切断した後に内部電源を切断してください。逆の電源投入切断順により，内部素子に過電圧が印加され，誤動作を引き起こしたり，異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源投入切断シーケンス」についての記載のある製品については，その内容を守ってください。

(6) 電源OFF時における入力信号　当該デバイスの電源がOFF状態の時に，入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により，誤動作を引き起こしたり，異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源OFF時における入力信号」についての記載のある製品については，その内容を守ってください。

はじめに

対象者　このアプリケーション・ノートは，V850ES/Jx3-Hの機能を理解し，それを用いたアプリケーション・システムを開発しようとするユーザを対象とします。

目　的　このアプリケーション・ノートは，V850ES/Jx3-Hに内蔵のUSBファンクション・コントローラを使用するためのサンプル・ドライバの仕様をユーザに理解していただくことを目的とします。

構　成　このアプリケーション・ノートは，大きく分けて次の内容で構成しています。

- USB規格の概要
- サンプル・ドライバ、サンプル・アプリケーションの仕様
- 開発環境
- サンプル・ドライバ、サンプル・アプリケーションの応用

読み方　このアプリケーション・ノートの読者には，電気，論理回路，マイクロコンピュータの一般知識を必要とします。

V850ES/Jx3-Hのハードウエア機能，および電気的特性を知りたいとき

→ 別冊の対象となるV850ES/JG3-H, V850ES/JH3-H ユーザーズ・マニュアル　ハードウエア編を参照してください。

V850ES/Jx3-Hの命令機能を知りたいとき

→ 別冊のV850ES ユーザーズ・マニュアル アーキテクチャ編を参照してください。

凡　例　データ表記の重み：左側が上位桁，右側が下位桁

注：　本文中につけた注の説明

注意：　特に気をつけて読んでいただきたい内容

備考：　本文の補足説明

数の表記：　2進数または10進数 … XXXX

16進数 … 0xXXXX

2のべき数を示す接頭語（アドレス空間，メモリ容量）：

K（キロ） … $2^{10} = 1024$

M（メガ） … $2^{20} = 1024^2$

G（ギガ） … $2^{30} = 1024^3$

T（テラ） … $2^{40} = 1024^4$

P（ペタ） … $2^{50} = 1024^5$

E（エクサ） … $2^{60} = 1024^6$

関連資料 関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。予め、ご了承ください。

○V850ES/Jx3-Hに関する資料

資 料 名	資料番号
V850ES ユーザーズ・マニュアル アーキテクチャ編	U15943J
V850ES/JG3-H, V850ES/JH3-H ユーザーズ・マニュアル ハードウエア編	U19181J
V850ES/JG3-H, V850ES/JH3-H アプリケーション・ノート USBプリンタクラスドライバ編	このアプリケーション・ノート

○開発ツールに関する資料（ユーザーズ・マニュアル）

資 料 名		資料番号
QB-V850ESJX3H インサーキット・エミュレータ		U19170J
QB-V850MINI オンチップ・デバック・エミュレータ		U17638J
QB-MINI2 フラッシュ・プログラミング機能付きオンチップ・デバック・エミュレータ		U18371J
CA850 Ver.3.30 Cコンパイラ・パッケージ	操作編	U18512J
	C言語編	U18513J
	アセンブリ言語編	U18514J
	リンク・ディレクティブ編	U18515J
PM+ Ver.6.31 プロジェクト・マネージャ		U18416J
ID850QB Ver.3.60 統合デバッガ	操作編	U18604J
SM850 Ver.2.50 システム・シミュレータ	操作編	U16218J
SM850 Ver.2.00以上 システム・シミュレータ	外部部品ユーザ・オープン・インタフェース仕様編	U14873J
SM+ システム・シミュレータ	操作編	U18601J
	ユーザ・オープン・インタフェース編	U18212J
	基礎編	U13430J
	インストレーション編	U17419J
	テクニカル編	U13431J
RX850 Ver.3.20 リアルタイムOS	タスク・デバッガ編	U17420J
	基礎編	U18165J
	内部構造編	U18164J
RX850 Pro Ver.3.21 リアルタイムOS	タスク・デバッガ編	U17422J
AZ850 Ver.3.30 システム・パフォーマンス・アナライザ		U17423J
PG-FP5 フラッシュ・メモリ・プログラマ		U18865J

目　　次

第1章 概　　説

このアプリケーション・ノートは，マイクロコントローラ V850ES/Jx3-H(V850ES/JG3-H, V850ES/JH3-H)のUSBファンクション・コントローラ機能を用いて作成されたUSBプリンタ・クラス用サンプル・ドライバについて説明します。主に次に示す内容で構成されます。

- サンプル・ドライバの仕様
- サンプル・ドライバの利用、およびアプリケーション・プログラム開発のための環境

この章では，サンプル・ドライバの概要と，適用対象となるマイクロコントローラについて説明します。

1. 1　サンプル・ドライバの概要

1. 1. 1　　サンプル・ドライバの特徴

V850ES/Jx3-H向けUSBプリンタ・クラス用サンプル・ドライバには，次の様な特徴があります。動作の詳細は**第3章　サンプル・ドライバの仕様**を参照してください。

- USBプリンタ・クラス Ver.1.1に準拠
- 仮想USBプリンタ・デバイスとして、印刷データをRAM領域に保存
- 次に示すサイズのメモリを占有（ベクタ・テーブルを除く）

  ROM：約4.1 kバイト

  RAM：約17.1kバイト(印刷データ格納領域(約16.4kバイト)を含む)

1. 1. 2　　サンプル・ドライバの構成

このサンプル・ドライバは次のようなファイルで構成されています。

表 1-1　サンプル・ドライバのファイル構成

フォルダ	ファイル	概　　要
src	main.c	メイン・ルーチン，初期化，サンプル・アプリケーション
	usbf850.c	USB初期化，エンドポイント制御，バルク転送，コントロール転送
	usbf850_print.c	プリンタ・クラス固有処理
	set_spreg.s	特殊レジスタ設定処理
include	main.h	main.c関数プロトタイプ宣言
	usbf850.h	usbf850.c関数プロトタイプ宣言
	usbf850_print.h	usbf850_print.c関数プロトタイプ宣言
	usbf850_desc.h	ディスクリプタ定義
	usbf850_error.h	エラー・コード定義
	usbf850_types.h	ユーザ型定義

備考　このほか，PM+（NECエレクトロニクス製統合開発ツール）で開発環境を構築した場合に生成されるプロジェクト関連ファイル一式も同梱されています。詳細は**5. 2. 1　ホスト環境整備**を参照してください。

1. 2 V850ES/Jx3-H の概要

V850ES/Jx3-Hは，NECエレクトロニクスのリアルタイム制御向けシングルチップ・マイクロコントローラV850マイコンのロウ・パワー・シリーズの製品群です。V850ES CPUコアを使用し，ROM/RAM，タイマ／カウンタ，シリアル・インタフェース，A/Dコンバータ，D/Aコンバータ，DMAコントローラ，CAN，USBファンクション・コントローラなどの周辺機能を内蔵しています。詳細はV850ES/JG3-H，V850ES/JH3-H ユーザーズ・マニュアル ハードウエア編を参照してください。

1. 2. 1 USBファンクション・コントローラの特徴

V850ES/Jx3-HのUSBファンクション・コントローラには，次のような特徴があります。

- Universal Serial Bus Specification Rev.2.0に準拠
- フル・スピード（12 Mbps）デバイスとして動作
- エンドポイントを次のように構成

表 1-2 V850ES/Jx3-Hのエンドポイント構成

エンドポイント名	FIFOサイズ（バイト）	転送タイプ	備 考
Endpoint0 Read	64	コントロール転送（IN）	−
Endpoint0 Write	64	コントロール転送（OUT）	−
Endpoint1	64×2	バルク転送1（IN）	2バッファ構成
Endpoint2	64×2	バルク転送1（OUT）	2バッファ構成
Endpoint3	64×2	バルク転送2（IN）	2バッファ構成
Endpoint4	64×2	バルク転送2（OUT）	2バッファ構成
Endpoint7	8	インタラプト転送（IN）	−

- USB標準リクエストには自動応答（一部のリクエストを除く）
- バス・パワードとセルフ・パワードを選択可能 注1
- 内部クロックと外部クロックを選択可能 注2

  内部クロック：外部6 MHz×内部8逓倍（48 MHz）

  外部クロック：UCLK端子へ入力（f_{USB} = 48 MHz）

注1. サンプル・ドライバではセルフ・パワードを選択します。

2. サンプル・ドライバでは内部クロックを選択します。

1. 2. 2 適用製品

サンプル・ドライバは，次に示す製品に適用できます。

表 1-3 V850ES/Jx3-H製品一覧

愛 称	品 名	内蔵メモリ		内蔵USB機能	割り込み		UM
		フラッシュ・メモリ	RAM[注1]		内部[注2]	外部[注2]	
V850ES/JG3-H	μPD70F3760	256 KB	40 KB	Function	69	17	ユーザーズ・マニュアル
	μPD70F3761	384 KB	48 KB	Function	69	17	V850ES/JG3-H,V850ES/JH3-H
	μPD70F3762	512 KB	56 KB	Function	69	17	32ビット・シングルチップ・マイ
	μPD70F3770	256 KB	40 KB	Function	73	17	クロコントローラ ハードウエ
V850ES/JH3-H	μPD70F3765	256 KB	40 KB	Function	69	20	ア編(U19181J)
	μPD70F3766	384 KB	48 KB	Function	69	20	
	μPD70F3767	512 KB	56 KB	Function	69	20	
	μPD70F3771	256 KB	40 KB	Function	73	20	

注1. データ専用RAM領域8 KBを含みます。

2. ノンマスカブル割り込みを含みます。

第2章 USBの概要

この章では，サンプル・ドライバが準拠するUSB規格の概要を説明します。

USB（Universal Serial Bus）は共通のコネクタでさまざまな周辺機器をホスト・コンピュータに接続できるようにするためのインタフェース規格です。ハブと呼ばれる分岐点を追加することで最大127個の機器を接続でき，Plug&Playで機器を認識できるホットプラグに対応しているなど，従来のインタフェースより柔軟で使いやすいのが特徴です。現在ではPCのUSBインタフェース搭載率はほぼ100%になってきており，PCと周辺機器間の標準インタフェースとして定着したと言えます。

USB規格の策定と管理はUSB Implementers Forum（USB-IF）という団体が行っています。USB規格の詳細はUSB-IFの公式ウェブサイト（www.usb.org）を参照してください。

2. 1 転送方式

USB規格では，4種類の転送方式（コントロール，バルク，インタラプト，アイソクロナス）が定義されています。各転送方式には表 2-1に示す特徴があります。

表 2-1 USBの転送方式

転送方式 / 項目		コントロール転送	バルク転送	インタラプト転送	アイソクロナス転送
特徴		周辺機器の制御などに必要な情報のやりとりに使用される転送方式	非周期的に大量データを扱う転送方式	周期的でバンド幅が低いデータ転送方式	リアルタイム性が要求される転送方式
設定可能パケット・サイズ	ハイ・スピード 480 Mbps	64バイト	512バイト	1-1024バイト	1-1024バイト
	フル・スピード 12 Mbps	8, 16, 32, 64バイト	8, 16, 32, 64バイト	1-64バイト	1-1023バイト
	ロウ・スピード 1.5 Mbps	8バイト	−	1-8バイト	−
転送の優先順位		3	3	2	1

2.2　エンドポイント

エンドポイントはホスト・デバイスが通信相手を特定するための情報の1つで，0-15の番号と方向（IN/OUT）で指定されます。エンドポイントは周辺機器で使用するデータ通信経路ごとに用意しなければならず，複数の通信経路で共用できません[注]。たとえば，SDカードへの書き込み／読み出しとプリント出力の機能を持った機器の場合，SDカードへの書き込み用エンドポイント，読み出し用エンドポイント，プリント出力用エンドポイントを個別に持つ必要があります。どのような機器でも必ず使用するコントロール転送には，エンドポイント0を使用します。

データ通信を行うとき，ホスト・デバイスは機器を特定するUSBデバイス・アドレスとともにエンドポイント（番号と方向）を使用して，機器内部の通信先を特定します。

エンドポイントのための物理的な回路として周辺機器内にバッファ・メモリを装備し，USBと通信先（メモリ等）の速度差を吸収するFIFOの役割も果たします。

注　オルタナティブ設定という仕組みを使い，排他的に切り替える方法はあります。

2.3　デバイス・クラス

USBを介して接続する周辺機器（ファンクション・デバイス）には，その機能によりさまざまなデバイス・クラスが定義されています。代表的なクラスとしてマス・ストレージ・クラス（MSC），コミュニケーション・デバイス・クラス（CDC），ヒューマン・インタフェース・デバイス・クラス（HID）などがあります。各デバイス・クラスにはプロトコルなどで標準仕様が定められているため，これに準拠していれば共通のホスト・ドライバを使用できます。

プリンタ・クラスは，ホスト・コンピュータに接続して使用するプリンタ・デバイスの為のクラスです。通信形態に応じて、インタフェース・プロトコルが定義されており、サンプル・ドライバはその中の片方向通信インタフェースを使用しています。プリンタ･クラスの仕様に各プリンタ固有のコマンドに関する規定は一切ありませんので、各プリンタ固有のコマンド処理は各自、実装する必要があります。プリンタ・クラスの詳細は, Universal Serial Bus Device Class Definition for Printing Device Version1.1を参照して下さい。

2.4　リクエスト

USB規格では，ホスト・デバイスからファンクション・デバイスに対してリクエストと呼ばれるコマンドを発行することにより通信が開始されます。リクエストには処理の方向，種類，ファンクション・デバイスのアドレスなどのデータが含まれています。

2.4.1 種　　類

標準リクエスト，クラス・リクエスト，ベンダ・リクエストの3種類があります。

サンプル・ドライバでは次に示すリクエストに対応しています。

(1)　標準リクエスト

すべてのUSB対応機器で共通のリクエストです。

表 2-2　標準リクエスト一覧

リクエスト名	対象ディスクリプタ	概　　要
GET_STATUS	デバイス	電源（セルフ／バス）とリモート・ウエイクアップの設定の読み取り
	エンドポイント	Halt状態の読み取り
CLEAR_FEATURE	デバイス	リモート・ウエイクアップのクリア
	エンドポイント	Halt状態の解除（DATA PID = 0）
SET_FEATURE	デバイス	リモート・ウエイクアップまたはテスト・モードの設定
	エンドポイント	Halt状態の設定
GET_DESCRIPTOR	デバイス	対象ディスクリプタの読み取り
	コンフィギュレーション	
	ストリング	
SET_DESCRIPTOR	デバイス	対象ディスクリプタの変更（オプション）
	コンフィギュレーション	
	ストリング	
GET_CONFIGURATION	デバイス	現行設定のコンフィギュレーション値の読み取り
SET_CONFIGURATION	デバイス	コンフィギュレーション値の設定
GET_INTERFACE	インタフェース	対象インタフェースの現行設定のうちオルタナティブ設定値の読み取り
SET_INTERFACE	インタフェース	対象インタフェースのオルタナティブ設定値の設定
SET_ADDRESS	デバイス	USBアドレスの設定
SYNCH_FRAME	エンドポイント	フレーム同期のデータ読み取り

(2)　クラス・リクエスト

デバイス・クラス固有のリクエストです。サンプル・ドライバではPrinterクラスに対応したクラス・リクエストへの応答処理を実装しています。応答可能なリクエストは次のとおりです。

- Get Device ID
  プリンタ・デバイスのID(IEEE1284互換)をホストへ送信要求するリクエストです。
- Get Port Status
  プリンタ・デバイスのポート状態をホストへ送信要求するリクエストです。
- Soft Reset
  プリンタ・デバイスのリセットを要求するリクエストです。

2. 4. 2　フォーマット

USBリクエストは8バイト長で，次のようなフィールドで構成されています。

表 2-3　USBリクエストのフォーマット

オフセット	フィールド		説　　明
0	bmRequestType		リクエストの属性
		ビット7	データ転送方向
		ビット6, 5	リクエスト・タイプ
		ビット4-0	対象ディスクリプタ
1	bRequest		リクエスト・コード
2	wValue	下位	リクエストで使用する任意の数値
3		上位	
4	wIndex	下位	リクエストで使用するインデックスまたはオフセット
5		上位	
6	wLength	下位	データ・ステージでの転送バイト数
7		上位	(データ長)

2.5　ディスクリプタ

USB規格では，各ファンクション・デバイス固有の情報を定められた形式でコード化したものをディスクリプタと呼んでいます。ファンクション・デバイスは，ホスト・デバイスからのリクエストに応じてディスクリプタを送信します。

2.5.1 種　　類

次に示す5種類のディスクリプタが定義されています。

- デバイス・ディスクリプタ
  どのデバイスにも必ず存在するディスクリプタで，対応しているUSB仕様のバージョン，デバイス・クラス，プロトコル，Endpoint0に対する転送で利用可能な最大パケット長，ベンダID，プロダクトIDなどの基本情報が含まれています。
  GET_DESCRIPTOR_Deviceリクエストに応答して送信するディスクリプタです。
- コンフィギュレーション・ディスクリプタ
  すべてのデバイスに1つ以上存在するディスクリプタで，デバイスの属性（電源供給方法），消費電力などの情報を含みます。
  GET_DESCRIPTOR_Configurationリクエストに応答して送信するディスクリプタです。
- インタフェース・ディスクリプタ
  インタフェースごとに必要なディスクリプタで，インタフェース識別番号，インタフェース・クラス，サポートするエンドポイントの数などが含まれます。
  GET_DESCRIPTOR_Configurationリクエストに応答して送信するディスクリプタです。
- エンドポイント・ディスクリプタ
  インタフェース・ディスクリプタに指定されたエンドポイントごとに必要なディスクリプタで，転送タイプ（転送方向），転送で利用可能な最大パケット長，転送のインターバルを定義します。ただし，Endpoint0はこのディスクリプタを持ちません。
  GET_DESCRIPTOR_Configurationリクエストに応答して送信するディスクリプタです。
- ストリング・ディスクリプタ
  任意の文字列を含むディスクリプタです。
  GET_DESCRIPTOR_Stringリクエストに応答して送信するディスクリプタです。

2. 5. 2　フォーマット

ディスクリプタのサイズとフィールドは，次のように種類ごとに異なります。

備考　各フィールドのデータ並びはリトル･エンディアンです。

表 2-4　デバイス・ディスクリプタのフォーマット

フィールド	サイズ（バイト）	説　　明
bLength	1	ディスクリプタのサイズ
bDescriptorType	1	ディスクリプタの種類
bcdUSB	2	USB仕様リリース番号
bDeviceClass	1	クラス・コード
bDeviceSubClass	1	サブクラス・コード
bDeviceProtocol	1	プロトコル・コード
bMaxPacketSize0	1	Endpoint0の最大パケット・サイズ
idVendor	2	ベンダID
idProduct	2	プロダクトID
bcdDevice	2	デバイスのリリース番号
iManufacturer	1	製造者を表すストリング・ディスクリプタへのインデックス
iProduct	1	製品を表すストリング・ディスクリプタへのインデックス
iSerialNumber	1	デバイスの製造番号を表すストリング・ディスクリプタへのインデックス
bNumConfigurations	1	コンフィギュレーションの数

備考　ベンダID：USBデバイスを開発する各企業がUSB-IFから取得する識別番号
プロダクトID：ベンダIDを取得後，各企業が自社製品に割り振る識別番号

表 2-5　コンフィギュレーション・ディスクリプタのフォーマット

フィールド	サイズ（バイト）	説　　明
bLength	1	ディスクリプタのサイズ
bDescriptorType	1	ディスクリプタの種類
wTotalLength	2	コンフィギュレーション，インタフェース，エンドポイント・ディスクリプタの総バイト数
bNumInterfaces	1	このコンフィギュレーションが持つインタフェースの数
bConfigurationValue	1	このコンフィギュレーションの識別番号
iConfiguration	1	このコンフィギュレーションを記述するストリング・ディスクリプタへのインデックス
bmAttributes	1	このコンフィギュレーションの特徴
bMaxPower	1	このコンフィギュレーションの最大消費電流（2 *m* A単位）

表 2-6　インタフェース・ディスクリプタのフォーマット

フィールド	サイズ（バイト）	説　　明
bLength	1	ディスクリプタのサイズ
bDescriptorType	1	ディスクリプタの種類
bInterfaceNumber	1	このインタフェースの識別番号
bAlternateSetting	1	このインタフェースに対するオルタナティブ設定の有無
bNumEndpoints	1	このインタフェースが持つエンドポイントの数
bInterfaceClass	1	クラス・コード
bInterfaceSubClass	1	サブクラス・コード
bInterfaceProtocol	1	プロトコル・コード
iInterface	1	このインタフェースを記述するストリング･ディスクリプタへのインデックス

表 2-7　エンドポイント・ディスクリプタのフォーマット

フィールド	サイズ（バイト）	説　　明
bLength	1	ディスクリプタのサイズ
bDescriptorType	1	ディスクリプタの種類
bEndpointAddress	1	このエンドポイントの転送方向 このエンドポイントのアドレス
bmAttributes	1	このエンドポイントの転送タイプ
wMaxPacketSize	2	この転送の最大パケット・サイズ
bInterval	1	このエンドポイントのポーリング間隔

表 2-8　ストリング・ディスクリプタのフォーマット

フィールド	サイズ（バイト）	説　　明
bLength	1	ディスクリプタのサイズ
bDescriptorType	1	ディスクリプタの種類
bString	任意	任意のデータ列

第3章 サンプル・ドライバの仕様

この章では，V850ES/Jx3-H向けUSBプリンタ・クラス用サンプル・ドライバの機能と処理内容の詳細，および実装している関数の仕様について説明します。

3.1 概　　要

3.1.1 機　　能

サンプル・ドライバには次のような処理が実装されています。

(1) 初期化

USBファンクション・コントローラを使用できるようにするため，各種レジスタを操作して設定します。大きく分けて，CPUレジスタに対する設定とUSBファンクション・コントローラのレジスタに対する設定があります。詳細は**3. 2. 1 CPU初期化処理**，**3. 2. 2 USBファンクション・コントローラ初期化処理**を参照してください。

(2) エンドポイントに対する処理

USBファンクション・コントローラ内の転送用エンドポイントの状態はINTUSBF0割り込みにより通知されます。大きく分けて，コントロール転送用エンドポイント（Endpoint0）に対してサンプル・ドライバでデコードを行うリクエストを受信した時のCPUDEC割り込み，とバルク・アウト転送（受信）用エンドポイント（Endpoint2）に対してデータが正常受信されたことを示すBKO1DT割り込みがあります。Endpoint0の処理では，リクエスト応答も行います。詳細は**3. 2. 3 INTUSBF0割り込み処理**を参照してください。

(3) サンプル・アプリケーション

バルク・アウト転送（受信）用エンドポイントにあるデータを読み出し，印刷データ格納用RAM領域に書き込みます。詳細は**第4章 サンプル・アプリケーションの仕様**を参照してください。

3.1.2 リクエストへの対応

ここでは，サンプル・ドライバが対応するUSBリクエストについて説明します。

(4) 標準リクエスト

V850ES/Jx3-Hが自動的に応答しないリクエストに対し，サンプル・ドライバは次のような応答処理を行います。

(a) GET_DESCRIPTOR_string

ホストがファンクション・デバイスのストリング・ディスクリプタを取得するためのリクエストです。このリクエストを受信すると，サンプル・ドライバは要求されたストリング・ディスクリプタの送信処理（コントロール・リード転送）を行います。

(b) その他

サンプル・ドライバはSTALL応答します。

(5)　クラス・リクエスト

プリンタ・クラスの各クラス・リクエストに対し，サンプル・ドライバは次のような応答処理を行います。

(a)　Get Device ID

プリンタ・デバイスのID(IEEE1284互換)をホストへ送信要求するリクエストです。デバイス側はControl Endpointでデバイス IDを送信します。サンプル･ドライバでは、このリクエストを受信したら、Control Endpointで文字列 ”MFG:NECEL;CMD:none;MDL:V850EJG3HUSB;CLS:PRINTER”をホストへ送信します。

MFG：会社名

CMD：コマンドセット(例：PCL, NPAP, CLS, MLC, PS, PostScript, 1284.4等)

MDL：製品識別子

CLS：クラス情報(例：PRINTER, MODEM, NET, PCMCIA, PORTS, SCANNER, DIGCAM等)

(b)　Get Port Status

プリンタ・デバイスのポート状態をホストへ送信要求するリクエストです。デバイス側はControl EndpointでPrinter Port Status(1byte)を送信します。サンプル･ドライバでは、このリクエストを受信したら、Control Endpointで固定値”0x10(Select=1、Paper Empty=0、Not Error=1)”をホストに送信します。

表 3-1 ポート状態ビット定義

ビット	フィールド	概　　要
7~6	予約	0
5	Paper Empty	Paper Not Empty = 0, Paper Empty = 1
4	Select	Not Select = 0, Select = 1
3	Not Error	Error = 0, Not Error = 1
2~0	予約	0

(c)　Soft Reset

プリンタ・デバイスのリセットを要求するリクエストです。デバイス側はプリンタ・デバイスをリセットします。サンプル･ドライバでは、このリクエストを受信したら、受信バッファ(Bulk Outエンドポイント)のクリアを行います。

3. 1. 3　ディスクリプタの設定

サンプル・ドライバでの各ディスクリプタの設定を次に示します。各ディスクリプタの設定は，ヘッダ・ファイル "usbf850_desc.h" に記述されています。

(1)　デバイス・ディスクリプタ

GET_DESCRIPTOR_deviceリクエストに応答して送信されるディスクリプタです。

GET_DESCRIPTOR_deviceリクエストにはハードウエアが自動的に応答するため，設定内容はUSBファンクション・コントローラの初期化時にUF0DDnレジスタ（n = 0-17）に格納します。

表 3-2　デバイス・ディスクリプタの設定

フィールド	サイズ（バイト）	設定値	説　　明
bLength	1	0x12	ディスクリプタのサイズ：18バイト
bDescriptorType	1	0x01	ディスクリプタの種類：デバイス
bcdUSB	2	0x0200	USB仕様リリース番号：USB 2.0
bDeviceClass	1	0x00	インタフェース・ディスクリプタに記述
bDeviceSubClass	1	0x00	サブクラス・コード：なし
bDeviceProtocol	1	0x00	プロトコル・コード：固有プロトコル未使用
bMaxPacketSize0	1	0x40	Endpoint0の最大パケット・サイズ：64
idVendor	2	0x0409	ベンダID：NEC
idProduct	2	0x1D2	プロダクトID： 0x1D2
bcdDevice	2	0x0001	デバイスのリリース番号：第1版
iManufacturer	1	0x01	製造者を表すストリング・ディスクリプタへのインデックス：1
iProduct	1	0x02	製品を表すストリング・ディスクリプタへのインデックス：2
iSerialNumber	1	0x03	デバイスの製造番号を表すストリング・ディスクリプタへのインデックス：3
bNumConfigurations	1	0x01	コンフィギュレーションの数：1

(2)　コンフィギュレーション・ディスクリプタ

GET_DESCRIPTOR_configurationリクエストに応答して送信されるディスクリプタです。

GET_DESCRIPTOR_configurationリクエストにはハードウエアが自動的に応答するため，設定内容はUSBファンクション・コントローラの初期化時にUF0CIEnレジスタ（n = 0-255）に格納します。

表 3-3　コンフィギュレーション・ディスクリプタの設定

フィールド	サイズ（バイト）	設定値	説　　明
bLength	1	0x09	ディスクリプタのサイズ：9バイト
bDescriptorType	1	0x02	ディスクリプタの種類：コンフィギュレーション
wTotalLength	2	0x0020	コンフィギュレーション，インタフェース，エンドポイント・ディスクリプタの総バイト数：48バイト
bNumInterfaces	1	0x01	このコンフィギュレーションが持つインタフェースの数：1
bConfigurationValue	1	0x01	このコンフィギュレーションの識別番号：1
iConfiguration	1	0x00	このコンフィギュレーションを記述するストリング・ディスクリプタへのインデックス：0
bmAttributes	1	0xC0	このコンフィギュレーションの特徴：セルフ・パワード，リモート・ウエイクアップなし
bMaxPower	1	0x1B	このコンフィギュレーションの最大消費電流：54 mA

(3)　インタフェース・ディスクリプタ

GET_DESCRIPTOR_configurationリクエストに応答して送信されるディスクリプタです。

GET_DESCRIPTOR_configurationリクエストにはハードウエアが自動的に応答するため，設定内容はUSBファンクション・コントローラの初期化時にUF0CIEnレジスタ（n = 0-255）に格納します。

表 3-4　Interface0のインタフェース・ディスクリプタの設定

フィールド	サイズ（バイト）	設定値	説　　明
bLength	1	0x09	ディスクリプタのサイズ：9バイト
bDescriptorType	1	0x04	ディスクリプタの種類：インタフェース
bInterfaceNumber	1	0x00	このインタフェースの識別番号：0
bAlternateSetting	1	0x00	このインタフェースに対するオルタナティブ設定の有無：なし
bNumEndpoints	1	0x01	このインタフェースが持つエンドポイントの数：1
bInterfaceClass	1	0x07	クラス・コード：コミュニケーション・インタフェース・クラス：プリンタ・クラス
bInterfaceSubClass	1	0x01	サブクラス・コード：Printers
bInterfaceProtocol	1	0x01	プロトコル・コード：片方向インタフェース
iInterface	1	0x00	このインタフェースを記述するストリング・ディスクリプタへのインデックス：0

(4)　エンドポイント・ディスクリプタ

GET_DESCRIPTOR_configurationリクエストに応答して送信されるディスクリプタです。

GET_DESCRIPTOR_configurationリクエストにはハードウエアが自動的に応答するため，設定内容はUSBファンクション・コントローラの初期化時にUF0CIEnレジスタ（n = 0-255）に格納します。

表 3-5　Endpoint1のエンドポイント・ディスクリプタの設定

フィールド	サイズ（バイト）	設定値	説　　明
bLength	1	0x07	ディスクリプタのサイズ：7バイト
bDescriptorType	1	0x05	ディスクリプタの種類：エンドポイント
bEndpointAddress	1	0x81	このエンドポイントの転送方向：IN方向 このエンドポイントのアドレス：1
bmAttributes	1	0x02	このエンドポイントの転送タイプ：バルク
wMaxPacketSize	2	0x0040	この転送の最大パケット・サイズ：64バイト
bInterval	1	0x00	このエンドポイントのポーリング間隔：0 ms

表 3-6　Endpoint2のエンドポイント・ディスクリプタの設定

フィールド	サイズ（バイト）	設定値	説　　明
bLength	1	0x07	ディスクリプタのサイズ：7バイト
bDescriptorType	1	0x05	ディスクリプタの種類：エンドポイント
bEndpointAddress	1	0x02	このエンドポイントの転送方向：OUT方向 このエンドポイントのアドレス：2
bmAttributes	1	0x02	このエンドポイントの転送タイプ：バルク
wMaxPacketSize	2	0x0040	この転送の最大パケット・サイズ：64バイト
bInterval	1	0x00	このエンドポイントのポーリング間隔：0 ms

(5)　ストリング・ディスクリプタ

GET_DESCRIPTOR_stringリクエストに応答して送信されるディスクリプタです。

GET_DESCRIPTOR_stringリクエストを受信すると，サンプル・ドライバはストリング・ディスクリプタをUSBファンクション・コントローラのUF0E0Wレジスタに格納します。

表 3-7　ストリング・ディスクリプタの設定

（a）String 0

フィールド	サイズ（バイト）	設定値	説　　明
bLength	1	0x04	ディスクリプタのサイズ：4バイト
bDescriptorType	1	0x03	ディスクリプタの種類：ストリング
bString	2	0x09, 0x04	言語コード：英語（U.S.）

（b）String 1

フィールド	サイズ（バイト）	設定値	説　　明
bLength 注1	1	0x2A	ディスクリプタのサイズ：42バイト
bDescriptorType	1	0x03	ディスクリプタの種類：ストリング
bString 注2	40	-	ベンダ：NEC Electronics Co.

注1. bStringフィールドのサイズにより設定値が異なります。

2. ベンダにより任意に設定できる領域のため，サイズや設定値は一定ではありません。

（c）String 2

フィールド	サイズ（バイト）	設定値	説　　明
bLength 注1	1	0x0E	ディスクリプタのサイズ：14バイト
bDescriptorType	1	0x03	ディスクリプタの種類：ストリング
bString 注2	12	-	製品の種類：PrtDrv（プリンタドライバ）

注1. bStringフィールドのサイズにより設定値が異なります。

2. ベンダにより任意に設定できる領域のため，サイズや設定値は一定ではありません。

（d）String 3

フィールド	サイズ（バイト）	設定値	説　　明
bLength 注1	1	0x1a	ディスクリプタのサイズ：26バイト
bDescriptorType	1	0x03	ディスクリプタの種類：ストリング
bString 注2	24	-	シリアル番号： 01D207010110

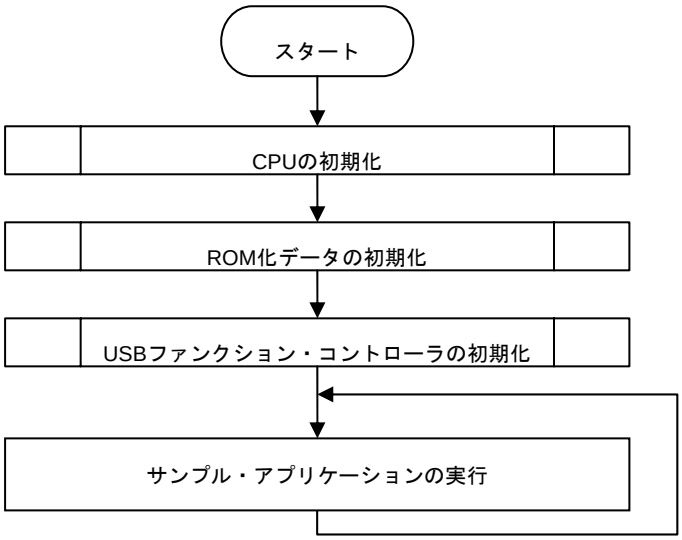
注1. bStringフィールドのサイズにより設定値が異なります。

2. ベンダにより任意に設定できる領域のため，サイズや設定値は一定ではありません。

3. 2　各部の動作

サンプル・ドライバを実行すると，次のような一連の処理を行います。ここでは，それぞれの処理について説明します。サンプル・アプリケーションの詳細は**第4章 サンプル・アプリケーションの仕様**を参照してください。

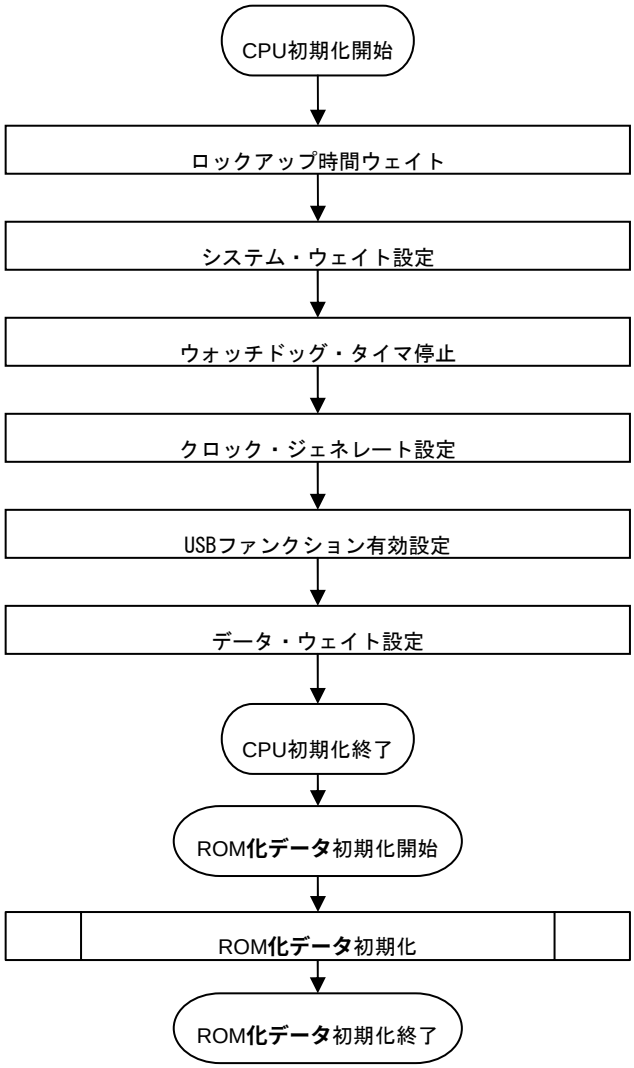
図 3-1　サンプル・ドライバの処理フロー



3. 2. 1　CPU初期化処理とROM化データ初期化処理

USBファンクション・コントローラを使用するために必要な項目を設定します。

図 3-2　CPU初期化とROM化データ初期化の処理フロー



(1)　ロックアップ時間ウェイト

CPUのクロック周波数が安定するまで（ロック状態）待機します。ここではLOCKRレジスタのLOCKビットが "0" であるかを監視します。

(2)　システム・ウェイト設定

内蔵周辺I/Oレジスタに対するバス・アクセスのウェイト数を設定します。

ここではVSWCレジスタに "0x12" を書き込みます。動作周波数に応じたウェイト数を設定して下さい。

(3)　ウォッチドッグ・タイマ停止

ウォッチドッグ・タイマの動作モードを切り替えます。

ここではWDTM2レジスタのWDM21，WDM20ビットに "00" を書き込みます。この設定により，ウォッチドッグ・タイマを停止させます。

(4)　クロック・ジェネレート設定

CPUの内部クロックの動作を設定します。ここでは次の3つのレジスタにアクセスします。

(a) CKCレジスタに "0x0B" を書き込みます。この設定により，内部発振回路で生成されたクロックの周波数がPLLで8逓倍されます。

(b) PLLCTLレジスタに "0x03" を書き込みます。この設定により，PLLモードが指定され，PLLの動作を開始させます。

(c) PCCレジスタに "0x00" を書き込みます。この設定により，内部クロック周波数としてf_{XX}が指定されます。サンプル・ドライバではメイン・クロックでの動作を想定して設定しています。

(5)　USBファンクション有効設定

UCKSELレジスタに"0x02"を書き込みます。この設定により，USBコントローラの動作クロックがメイン・クロックから供給されるようになります。次いで、UFCKMSKレジスタに" 0x00", UHCKMSKレジスタに" 0x03"を書き込み、USBファンクション･コントローラの動作を許可します。

(6)　データ・ウェイト設定

動作速度の異なる複数のI/Oそれぞれに対し，データ・アクセスのウェイト数を設定します。ここではDWC0レジスタに "0x1171" を書き込みます。この設定により，3つのI/Oのいずれも1ウェイトでアクセスされます。

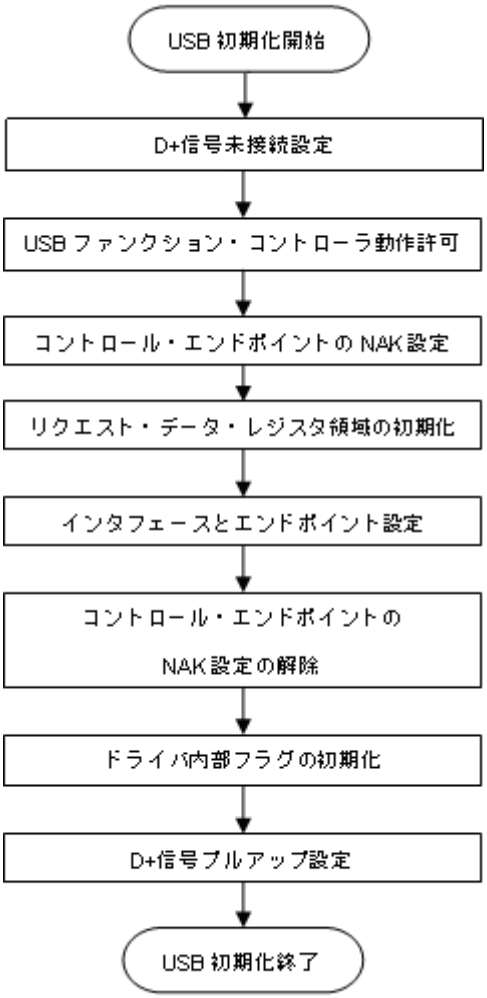
(7)　ROM化データ初期化

コピー関数(rcopy)を用い、ROM化データの初期化処理を行います。

3. 2. 2　USBファンクション・コントローラ初期化処理

USBファンクション・コントローラの使用を開始するために必要な項目を設定します。

図 3-3　USBファンクション・コントローラ初期化の処理フロー



(1)　D+信号プルダウン設定

CPUのPM4.1ビットに"0"を、PM4.2ビットに"1"を書き込み、P4.1を出力モードにして、P4.1に"0"を設定します。

(2)　コントロール・エンドポイントのNAK設定

ここではUF0E0NAレジスタのEP0NKAビットに "1" を書き込みます。この設定により，自動応答リクエストを含むすべてのリクエストに対してハードウエアがNAKで応答します。

このビットは，自動応答リクエストで使用するデータの登録が完了するまで，ハードウエアが自動応答リクエストに対して意図しないデータを返さないようにするために，ソフトウェアが使用します。

(3)　リクエスト・データ・レジスタ領域の初期化

GET_DESCRIPTORリクエストに自動応答するためのディスクリプタ・データなどを各種レジスタに登録します。

ここでは次に示すレジスタにアクセスします。

(a) UF0DSTLレジスタに "0x01" を書き込みます。この設定により，リモート・ウエイクアップ機能の使用が禁止され，USBファンクション・コントローラはセルフ・パワード・デバイスとして動作します。

(b) UF0EnSLレジスタ（n = 0-2）に "0x00" を書き込みます。この設定により，Endpoint nが正常に動作していることを示します。

(c) UF0DSCLレジスタに，必要なディスクリプタのデータ長の合計（バイト数）を書き込みます。この設定により，使用されるUF0CIEnレジスタ（n = 0-255）の範囲が決まります。

(d) UF0DDnレジスタ（n = 0-7）にデバイス・ディスクリプタのデータを書き込みます。

(e) UF0CIEnレジスタ（n = 0-255）にコンフィギュレーション・ディスクリプタ，インタフェース・ディスクリプタ，およびエンドポイント・ディスクリプタのデータを書き込みます。

(f) UF0MODCレジスタに "0x00" を書き込みます。この設定により，GET_DESCRIPTOR_configurationリクエストへの自動応答が許可されます。

(4)　インタフェースとエンドポイントの設定

サポートするインタフェースの数，オルタナティブ設定の状態，インタフェースとエンドポイントの関係などの情報を各種レジスタに設定します。

ここでは次に示すレジスタにアクセスします。

(a) UF0AIFNレジスタに "0x00" を書き込みます。この設定により，1つのインタフェースを有効にします。

(b) UF0AASレジスタに "0x00" を書き込みます。この設定により，オルタナティブ設定を無効にします。

(c) UF0E1IMレジスタに "0x20" を書き込みます。この設定により，Endpoint1がInterface0にリンクされます。

(d) UF0E2IMレジスタに "0x20" を書き込みます。この設定により，Endopoint2がIntreface0にリンクされます。

(5)　コントロール・エンドポイントのNAK設定の解除

ここではUF0E0NAレジスタのEP0NKAビットに "0" を書き込みます。この設定により，自動応答リクエストを含むすべてのリクエストに対して，それぞれに応じた応答が再開されます。

(6)　割り込みマスク・レジスタの設定

USBファンクション・コントローラの割り込み要因ごとのマスクを設定します。

ここでは次に示すレジスタにアクセスします。

(a) UF0ICnレジスタ（n = 0-7）に "0x00" を書き込みます。この設定により，すべての割り込み要因がクリアされます。

(b) UF0FICnレジスタ（n = 0, 1）に "0x00" を書き込みます。この設定により，すべての転送用FIFOがクリアされます。

(c) UF0IM0レジスタに "0x1D" を書き込みます。この設定により，UF0IS0レジスタに示される割り込み要因のうち，BUSRST割り込み、RSUSPD割り込み、SETRQ割り込み以外の要因がすべてマスクされます。

(d) UF0IM1レジスタに "0x7E" を書き込みます。この設定により，UF0IS1レジスタに示される割り込み要

因のうち，CPUDEC割り込み以外の要因がすべてマスクされます。

(e) UF0IM2レジスタに "0xF3" を書き込みます。この設定により，UF0IS2レジスタに示される割り込み要因がすべてマスクされます。

(f) UF0IM3レジスタに "0xFE" を書き込みます。この設定により，UF0IS3レジスタに示される割り込み要因のうち，BKO1DT割り込み以外の要因がすべてマスクされます。

(g) UF0IM4レジスタに "0x20" を書き込みます。この設定により，UF0IS4レジスタに示される割り込み要因がすべてマスクされます。

(i) BRGINTEレジスタに"0x0003"を書き込み、EPCINT0BEN、EPCINT1BENビットが立った時の割り込みを有効にします。

(j) UFIF0に"0"を、UFMK0に"0"を書き込み、INTUSBF0を有効にします。

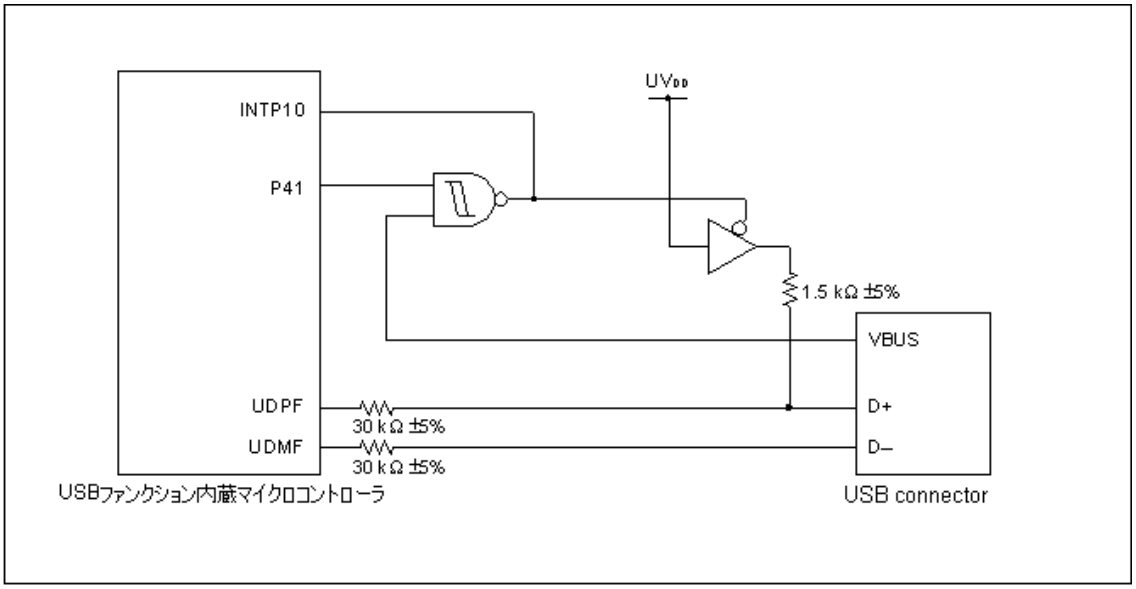
(7)　ドライバ内部フラグの初期化

ドライバ内部で使用するフラグ(usbf850_usbstate_flg，usbf850_rdata_flg)の初期化を行います。

(8)　D+信号プルアップ設定

V8CPUのP4レジスタに "0x02" を書き込みます。この設定により，P41から "1" が出力されます。これにより、D+信号からハイ・レベルを出力して，ホスト側にデバイスが接続されたことを通知します。サンプル・ドライバでは図 3-4に示すような接続を想定しています。

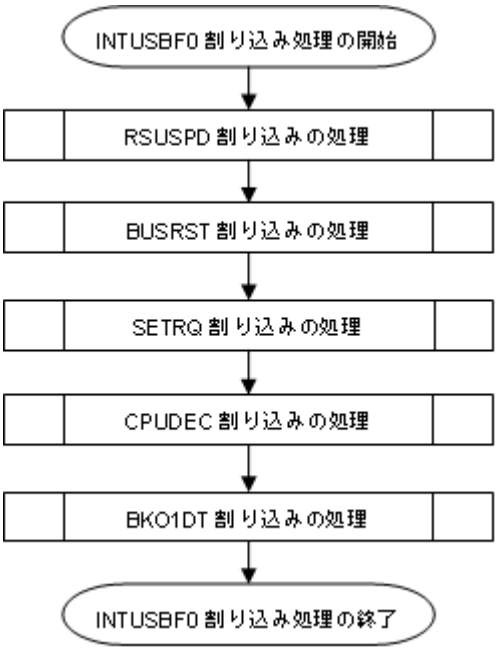
図 3-4　USBファンクション・コントローラ接続例



3. 2. 3　INTUSBF0割り込み処理

USBファンクション・コントローラからの割り込み要求（INTUSBF0）は、初期化時にマスク解除された割り込みについてのみ通知されます。必要な割り込みについては、初期化時に割り込みマスクを解除して下さい。通知された割り込みについて、それぞれ必要な処理を行います。

図 3-5　Endpoint0監視の処理フロー



(1)　RSUSPD割り込み処理

UF0IS0レジスタのRSUSPDビットが "1" であれば、UF0IC0のRSUSPDCビットをクリア(0)します。次に、UF0EPS1のRSUMビットが"1"であれば、サスペンド状態を示すフラグ(usbf850_rsuspd_flg)を更新(サスペンド発生)します。

(2)　BUSRST割り込み処理

UF0IS0レジスタのBUSRSTビットが "1"であれば、UF0IC0のBUSRSTCビットをクリア(0)します。次に、バスリセットが発生した事を示すフラグ(usbf850_usbstate_flg)を更新(バスリセット発生)し、バッファを初期化します。次にUF0E0NAレジスタのEP0NKAビットに "1" を書き込み、NAK応答設定します。UF0DSTLを再設定(“0x01”)し、設定後、NAK応答を解除します。

(3)　SETRQ割り込み処理

UF0IS0レジスタのSETRQビットが "1"であれば、UF0IC0のSETRQCビットをクリア(0)します。次に、UF0SETレジスタのSETCONビットとUF0MODSレジスタのCONFビットが"1"だった場合、Set Configurationリクエストが処理された事を示すフラグ(usbf850_ usbstate_flg)を更新(Configured状態)します。

(4)　CPUDEC割り込み処理

UF0IS1レジスタのCPUDECビットが "1"であれば、UF0IC1のCPUDECCビットをクリア(0)します。次に、UF0ESTレジスタを8回読み出し、リクエスト・データを取り込んでデコードします。リクエストが標準リク

エストならusbf850_standardreq()関数を呼び出し、クラス・リクエストならusbf850_classreq()関数を呼び出す。

(5)　BKODT割り込み処理

UF0IS3レジスタのBKODTビットが "1"であれば、UF0IC3のBKODTCビットをクリア(0)します。次に、データを受信した事を示すフラグ(usbf850_rdata_flg)を更新します。受信FIFOに有効なデータが格納された場合に発生します。

3. 3　関数の仕様

ここでは，サンプル・ドライバに実装されている各種関数について説明します。

3. 3. 1　関数一覧

サンプル・ドライバでは，ソース・ファイルそれぞれに次のような関数が実装されています。

表 3-8　サンプル・ドライバ内の関数

ソース・ファイル	関数名	説　　明
main.c	main	メイン・ルーチン
	cpu_init	CPUの初期化
	romp_init	ROM化データの初期化
	printerDev_init	プリンタ・データ格納領域の初期化
	rx_printing	受信データ読み出し、書き込み処理
usbf850.c	usbf850_init	USBファンクション・コントローラの初期化
	usbf850_intusbf0	Endpoint0の監視とリクエストへの応答制御
	usbf850_data_send	USBデータの送信
	usbf850_data_receive	USBデータの受信
	usbf850_rdata_length	USB受信データ長の取得
	usbf850_send_EP0	Endpoint0の送信
	usbf850_receive_EP0	Endpoint0の受信
	usbf850_send_null	Bulk/ Interrupt In EndpointへのNullパケット送信処理
	usbf850_sendnullEP0	Endpoint0用NULLパケットの送信
	usbf850_sendstallEP0	Endpoint0用STALL応答
	usbf850_ep_status	Bulk/ Interrupt In EndpointのFIFO状態通知処理
	usbf850_fifo_clear	Endpoint0以外のEndpointのFIFOクリア
	usbf850_standardreq	標準リクエストの処理
	usbf850_getdesc	GET_DESCRIPTORリクエストの処理
usbf850_print.c	usbf850_classreq	プリンタ・クラス・リクエスト受信処理
	usbf850_get_device_id	Get Device IDリクエストの処理
	usbf850_get_port_status	Get Port Statusリクエストの処理
	usbf850_soft_reset	Soft Resetリクエストの処理

3. 3. 2　関数の相関関係

関数によっては，処理の中で別の関数を呼び出しているものもあります。関数の呼び出し関係を次に示します。

図 3-6　メイン・ルーチンでの関数の呼び出し

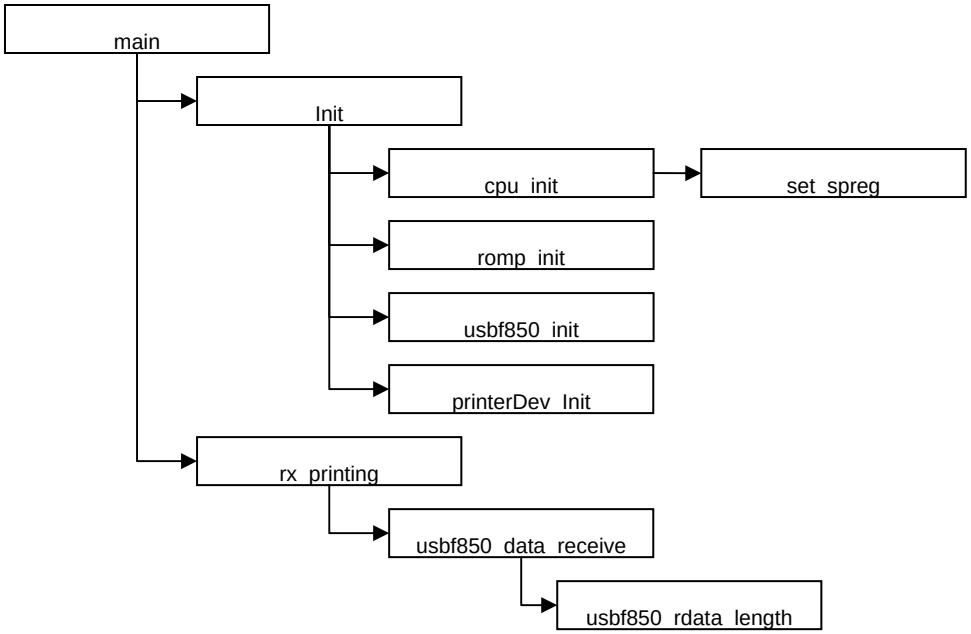
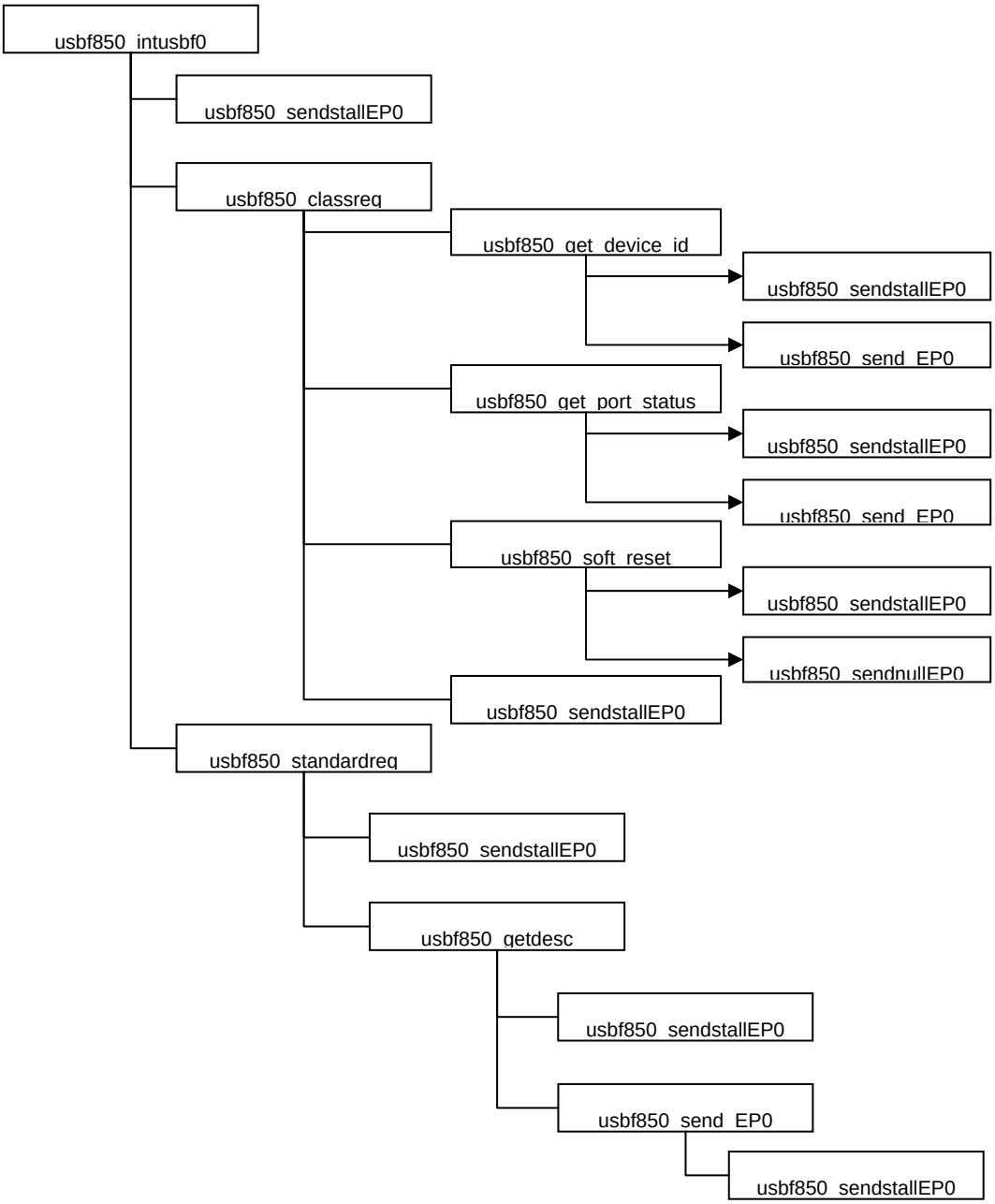


図 3-7　USBファンクション・コントローラ用処理での関数の呼び出し



3. 3. 3　関数の機能

ここでは，サンプル・ドライバに実装されている各種関数について解説します。

(1)　関数解説フォーマット

解説は，関数ごとに次の形式で記述されます。

関数名称

【概要】

概要説明

【C言語記述形式】

C言語上の記述形式

【パラメータ】

パラメータ（引数）の説明

パラメータ	説　　明
パラメータ型，名称	*パラメータ概要説明*

【戻り値】

戻り値の説明

シンボル	説　　明
戻り値型，名称	*戻り値概要説明*

【機能】

機能説明

(2)　メイン・ルーチンの関数

main

【概要】

メイン処理

【C言語記述形式】

```
void main(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

サンプル・ドライバを実行すると最初に呼び出される関数です。

CPU，USBファンクション・コントローラの初期化を順に実行します。次に，サンプル・アプリケーションの処理を行います。

cpu_init

【概要】

CPU初期化処理

【C言語記述形式】

```
void cpu_init(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

初期化処理で呼び出される関数です。

バス・アクセス時のウェイト数やクロック周波数，動作モードなど，USBファンクション・コントローラを使用するために必要な項目等を設定します。

romp_init

【概要】

ROM化パッケージ用初期化処理

【C言語記述形式】

```
void romp_init(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

初期化処理で呼び出される関数です。

コピー関数（_rcopy**）を呼び出し，指定されたアドレスに格納されている情報を**RAM**領域に**1**バイトずつコピーします。**

rx_printing

【概要】

印刷(サンプル・アプリケーション)処理

【C言語記述形式】

```
void rx_printing(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

メイン処理で受信割り込みがあった事を示すフラグ（usbf850_rdata_flg）を監視し、受信割り込みがあった場合に呼び出される関数です。要求サイズ分、印刷データを受信バッファより読み出し、印刷データ格納領域(printer_data)へと書き込むためusbf850_data_receive関数を呼び出します。終了後、読み出した分のポインタ操作を行います。

(3)　USBファンクション・コントローラ用処理の関数

usbf850_init

【概要】

USBファンクション・コントローラ初期化処理

【C言語記述形式】

```
void usbf850_init(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

初期化処理で呼び出される関数です。

データ領域の確保と設定，割り込み要求のマスクなど，USBファンクション・コントローラの使用を開始するために必要な項目を設定します。

usbf850_intusbf0

【概要】

Endpoint0**監視処理**

Endpoint2**監視処理**

【C言語記述形式】

```
void intusb0b(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

INTUSBF0**割り込みにより呼び出される割り込みサービスルーチンです。**

USB**ファンクション・コントローラのマスクされていない割り込みについて、割り込み内容を確認し、発生している割り込みについて処理を行います。**

usbf850_data_send

【概要】

Bulk/ Interrupt In Endpoint用USBデータ送信処理

【C言語記述形式】

```
INT32 usbf850_data_send(UINT8* data, INT32 len, INT8 ep)
```

【パラメータ】

パラメータ	説　　明
UINT8* data	送信データ・バッファ・ポインタ
INT32 len	送信データ長
INT8 ep	データ送信エンドポイント番号

【戻り値】

シンボル	説　　明
len（>=0）	正常に送信したデータサイズ
DEV_ERROR	異常終了

【機能】

送信データ・バッファに格納されているデータを，指定されたエンドポイント用のFIFOに1バイトずつ格納します。

usbf850_send_null

【概要】

Bulk/ Interrupt In Endpoint**用**Null**パケット送信処理**

【C言語記述形式】

```
INT32 usbf850_send_null (INT8 ep)
```

【パラメータ】

パラメータ	説　　明
INT8 ep	**データ送信エンドポイント番号**

【戻り値】

シンボル	説　　明
DEV_OK	**正常終了**
DEV_ERROR	**異常終了**

【機能】

指定されたEndpoint**（送信用）の**FIFO**をクリアし、データ終了を示すビットをセット**(1)**する事で、**USB**ファンクション・コントローラから**Null**パケットを送信します。**

usbf850_data_receive

【概要】

USB**データ受信処理**

【C言語記述形式】

```
INT32 usbf850_data_receive(UINT8* data, INT32 len, INT8 ep)
```

【パラメータ】

パラメータ	説　　明
UINT8* data	**受信データ・バッファ・ポインタ**
INT32 len	**受信データ長**
INT8 ep	**データ受信エンドポイント番号**

【戻り値】

シンボル	説　　明
len(>=0)	**正常に受信したデータサイズ**
DEV_ERROR	**異常終了**

【機能】

指定されたエンドポイント用のFIFO**からデータを**1**バイトずつ読み出し，受信データ・バッファに格納します。**

usbf850_rdata_length

【概要】

USB受信データ長取得

【C言語記述形式】

```
void usbf850_rdata_length(INT32* len , INT8 ep)
```

【パラメータ】

パラメータ	説　　明
INT32* len	受信データ長格納アドレス・ポインタ
INT8 ep	データ受信エンドポイント番号

【戻り値】

なし

【機能】

指定されたエンドポイントの受信データ長を読み出します。

usbf850_send_EP0

【概要】

Endpoint0用USBデータ送信処理

【C言語記述形式】

```
void usbf850_send_EP0(UINT* data, INT32 len)
```

【パラメータ】

パラメータ	説　明
UINT* data	送信データ・バッファ・ポインタ
INT32 len	送信データサイズ

【戻り値】

シンボル	説　明
DEV_OK	正常終了
DEV_ERROR	異常終了

【機能】

送信データ・バッファに格納されているデータをEndpoint0用送信FIFOに1バイトずつ格納します。

usbf850_sendnullEP0

【概要】

Endpoint0用NULLパケット送信処理

【C言語記述形式】

```
void usbf850_sendnullEP0(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

Endpoint0用のFIFOをクリアし，データ終了を示すビットをセット（1）することで，USBファンクション・コントローラからNULLパケットを送信させます。

usbf850_sendstallEP0

【概要】

Endpoint0用STALL応答処理

【C言語記述形式】

```
void usbf850_sendstallEP0(void)
```

【パラメータ】

なし

【戻り値】

なし

【機能】

STALLハンドシェーク使用を示すビットをセット（1）することで，USBファンクション・コントローラからSTALL応答させます。

usbf850_receive_EP0

【概要】

Endpoint0用USBデータ受信処理

【C言語記述形式】

```
INT32 usbf850_receive_EP0(UINT* data, INT32 len)
```

【パラメータ】

パラメータ	説　　明
UINT* data	受信データ・バッファ・ポインタ
INT32 len	受信データサイズ

【戻り値】

シンボル	説　　明
DEV_OK	正常終了
DEV_ERROR	異常終了

【機能】

Endpoint0用受信FIFOから1バイトずつ読み出し、受信データ・バッファに格納します。

usbf850_ep_status

【概要】

Bulk/ Interrupt In Endpoint用FIFO状態通知処理

【C言語記述形式】

```
INT32 usbf850_ep_status(INT8 ep)
```

【パラメータ】

パラメータ	説　明
INT8 ep	データ送信Endpoint番号

【戻り値】

シンボル	説　明
DEV_OK	正常終了
DEV_RESET	Bus Reset処理中
DEV_ERROR	異常終了

【機能】

指定されたEndpoint(送信用)のFIFO状態を通知します。

usbf850_standardreq

【概要】

USBファンクション・コントローラが自動応答しない標準リクエストの処理

【C言語記述形式】

```
void usbf850_standardreq(USB_SETUP *req_data)
```

【パラメータ】

パラメータ	説　明
USB_SETUP *req_data	受信データ長格納アドレス・ポインタ

【戻り値】

なし

【機能】

INTUSBF0割り込み処理のCPUDEC割り込み要因の場合に呼び出される関数です。

デコードされたリクエストがGET_DESCRIPTORの場合，GET_DESCRIPTORリクエスト処理関数（usbf850_getdesc）を呼び出します。それ以外のリクエストの場合はEndpoint0用STALL応答処理関数（usbf850_sendstallEP0）を呼び出します。

usbf850_getdesc

【概要】

GET_DESCRIPTOR**リクエスト処理**

【C言語記述形式】

```
void usbf850_getdesc(USB_SETUP *req_data)
```

【パラメータ】

パラメータ	説　　明
USB_SETUP *req_data	**リクエスト・データ格納アドレス・ポインタ**

【戻り値】

なし

【機能】

USB**ファンクション・コントローラが自動応答しない標準リクエストの処理で呼び出される関数です。デコードされたリクエストがストリング・ディスクリプタを要求している場合，**USB**データ送信処理関数（**usbf850_send_EP0**）を呼び出して，**Endpoint0**からストリング・ディスクリプタを送信させます。それ以外のディスクリプタを要求している場合は**Endpoint0**用**STALL**応答処理関数（**usbf850_sendstallEP0**）を呼び出します。**

usbf850_fifo_clear

【概要】

Bulk/ Interrupt Endpoint**用**FIFO**クリア処理**

【C言語記述形式】

```
void usbf850_fifo_clear(INT8 in_ep, INT8 out_ep)
```

【パラメータ】

パラメータ	説　　明
INT8 in_ep	**データ送信**Endpoint
INT8 out_ep	**データ受信**Endpoint

【戻り値】

なし

【機能】

指定されたEndpoint（Bulk/Interrupt）**の**FIFO**をクリアし、データ受信フラグ**(usbf850_rdata_flg)**をクリアします。**

(4)　USBプリンタ・クラス用処理の関数

usbf850_get_device_id

【概要】

Get Device ID **リクエスト処理**

【C言語記述形式】

```
void usbf850_get_device_id(USB_SETUP *req_data)
```

【パラメータ】

パラメータ	説　　明
USB_SETUP *req_data	**リクエスト・データ格納ポインタ・アドレス**

【戻り値】

なし

【機能】

データ受信処理関数（usbf850_send_EP0**）を呼び出して、**usbf850_print.h**内で定義しているデバイス**ID(**文字列** "MFG:NECEL;CMD:none;MDL:V850EJG3HUSB;CLS:PRINTER")**を**Endpoint0**で送信します。**

usbf850_get_port_status

【概要】

Get Port Status **リクエスト処理**

【C言語記述形式】

```
void usbf850_get_port_status(USB_SETUP *req_data)
```

【パラメータ】

パラメータ	説　　明
USB_SETUP *req_data	**リクエスト・データ格納ポインタ・アドレス**

【戻り値】

なし

【機能】

データ受信処理関数（usbf850_send_EP0**）を呼び出して、**usbf850_print.h**内で定義しているポート・ステイタス**”0x10(Select=1、Paper Empty=0、Not Error=1)”**を**Endpoint0**で送信します。**

usbf850_soft_reset

【概要】

Soft Reset **リクエスト処理**

【C言語記述形式】

```
void usbf850_soft_reset(USB_SETUP *req_data)
```

【パラメータ】

パラメータ	説　明
USB_SETUP *req_data	**リクエスト・データ格納ポインタ・アドレス**

【戻り値】

なし

【機能】

使用エンドポイントのバッファをクリアし、Halt**状態にあるものは解除します。また、印刷データ格納領域も初期化し、プリンタ・デバイスを初期化します。**

usbf850_classreq

【概要】

プリンタ・クラス・リクエスト受信処理

【C言語記述形式】

```
void usbf850_classreq(USB_SETUP *req_data)
```

【パラメータ】

パラメータ	説　　明
USB_SETUP *req_data	リクエスト・データ格納ポインタ・アドレス

【戻り値】

なし

【機能】

INTUSBF0割り込み処理のCPUDEC割り込み要因で呼び出される関数です。デコードされたリクエストがプリンタ・クラス固有のリクエストの場合、各リクエスト処理関数を呼び出します。それ以外の場合はEndpoint0にSTALL応答します。

第4章 サンプル・アプリケーションの仕様

この章では，サンプル・ドライバに含まれるサンプル・アプリケーションについて説明します。

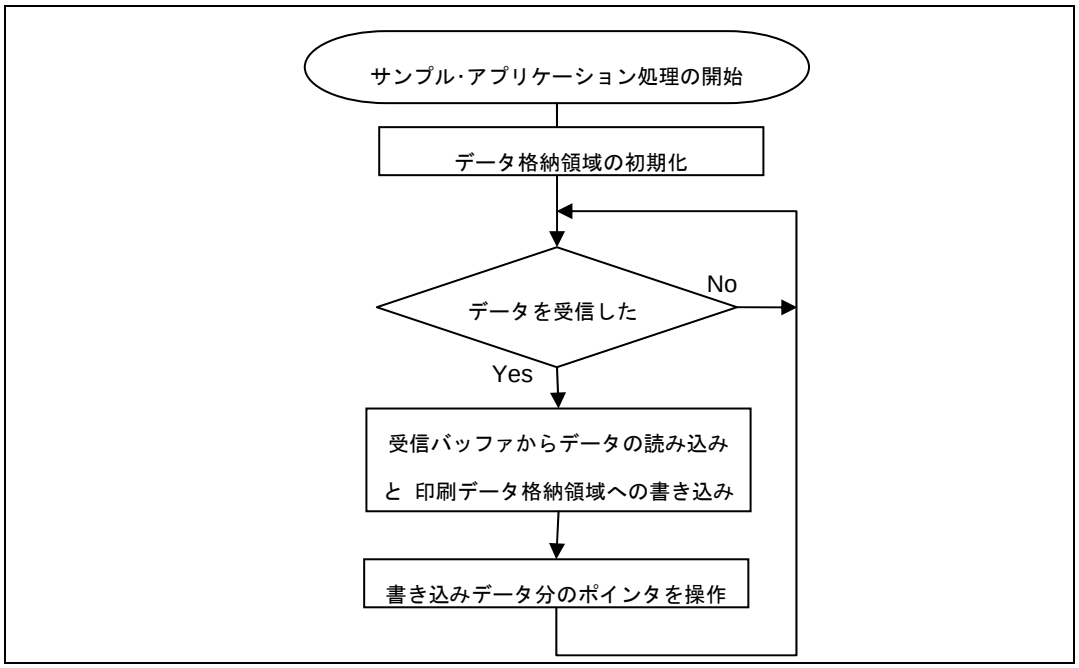
4.1 概　　要

サンプル・アプリケーションは，USBプリンタ・クラス用ドライバ利用の簡単な例として用意されたものです。USBファンクション・コントローラで受信したデータの読み出しと印刷データ格納領域への書き込みを処理します。

4.2 動　　作

サンプル・アプリケーションでは，次のようなフローで一連の処理が行われます。

図 4-1　サンプル・アプリケーションの処理フロー



(1)　データ格納領域の初期化

印刷データ格納領域(printer_data)を初期化します。

(2)　データ受信割り込み処理

データ受信割り込み発生通知フラグ（usbf850_rdata_flg）を監視し、割り込みが発生した場合は、受信処理関数(usbf850_data_receive)を呼び出し、受信バッファより読み出した受信データを印刷データ格納領域へ書き込みます。

(3)　印刷データ格納領域への書き込み用ポインタ操作

書き込んだ分だけ､印刷データ格納領域(printer_data)への書き込み用ポインタ(printer_data_ptr)を更新します。

第5章 開発環境

この章では，V850ES/Jx3-H向けUSBプリンタ・クラスのサンプル・ドライバを利用したアプリケーション・プログラムを開発する際の環境構築例と，デバッグの手順について説明します。

5. 1 開発環境

ここでは，ハードウエア・ツールとソフトウエア・ツールの製品構成例を示します。

5. 1. 1 プログラム開発

サンプル・ドライバを利用したシステムを開発する際は，次の様なハードウエアとソフトウェアが必要です。

表 5-1 プログラム開発環境構成例

構成品		製品例	備　考
ハードウエア	ホスト・マシン	－	PC/AT^{TM}互換機（OS：Windows XP）
ソフトウェア	統合開発ツール	PM+	V6.31
	コンパイラ	CA850	W3.30

5. 1. 2 デバッグ

サンプル・ドライバを利用したシステムをデバッグする際は，次の様なハードウエアとソフトウェアが必要です。

表 5-2 デバッグ環境構成例

構成品		製品例	備　考
ハードウエア	ホスト・マシン	－	PC/AT互換機（OS：Windows XP）
	ターゲット	TK-850/JG3H	テセラ・テクノロジー社製
	USBケーブル	－	miniBコネクタ - Aコネクタ
ソフトウェア	統合開発ツール	PM+	V6.31
	デバッガ	ID850QB	V3.60
ファイル	デバイス・ファイル	DF703771	V850ES/JG3-H用
	デバッグ・ポート用ホスト・ドライバ	－	注1
	プロジェクト関連ファイル	－	注2

注1. 製品や入手方法については弊社までお問い合わせください。

2. PM+で構築した場合のファイルがサンプル・ドライバに同梱されています。

5. 2　環境設定

ここでは，5. 1 開発環境に示した製品構成で開発やデバッグを行うための準備について、TK-850/JG3Hでの環境設定を例にして説明します。

5. 2. 1　ホスト環境整備

ホスト・マシン上に専用のワークスペースを作成します。

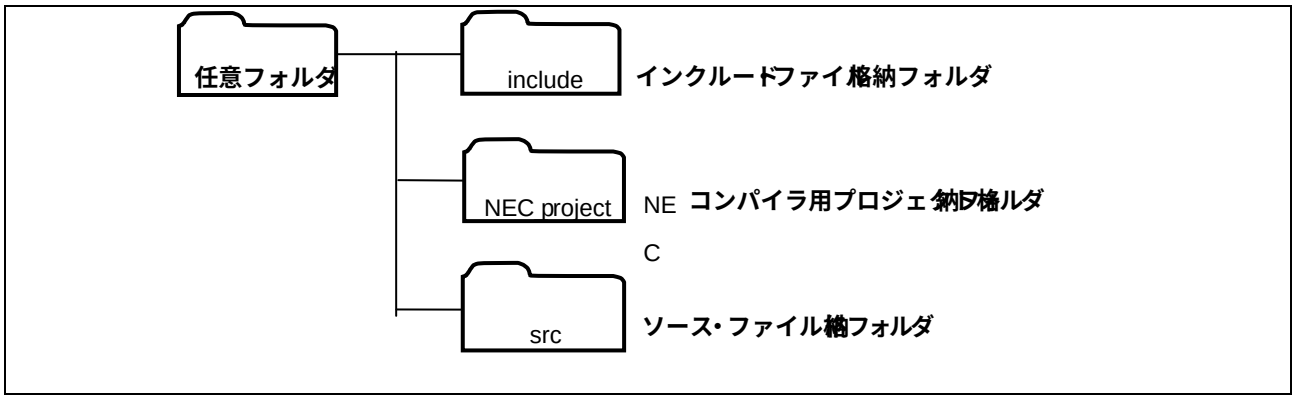
(1)　統合開発ツールのインストール

PM+をインストールします。詳細はPM+のユーザーズ・マニュアルを参照してください。

(2)　ドライバ類のダウンロード

サンプル・ドライバの提供ファイル一式を，フォルダ構成を変えずに任意のディレクトリに格納します。

図 5-1　サンプル・ドライバのフォルダ構成

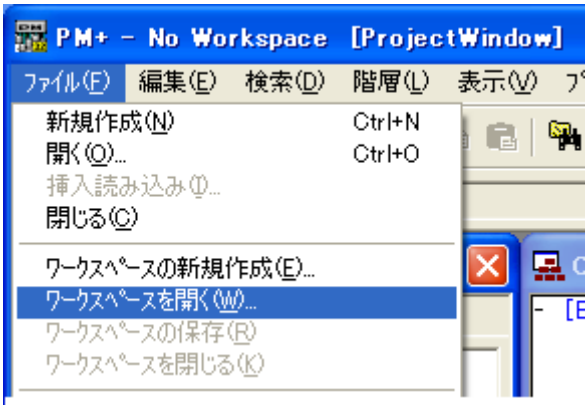


また，デバイス・ドライバを任意のディレクトリに格納します。

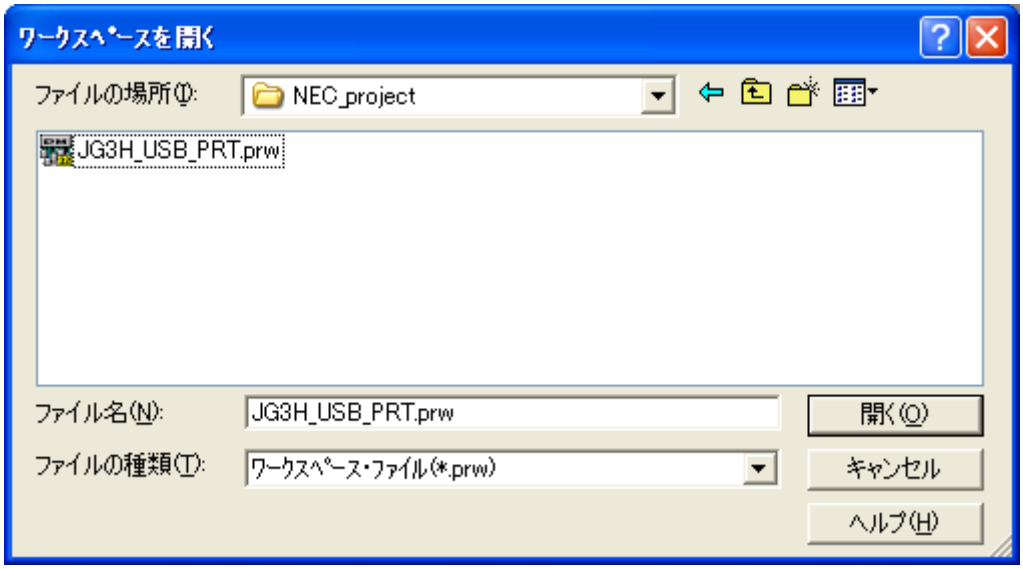
(3)　ワークスペースの設定

ここではサンプル・ドライバに同梱のプロジェクト関連ファイルを使用する場合の手順を示します。

<1> PM+を起動し，「ファイル」メニューから「ワークスペースを開く」を選択します。



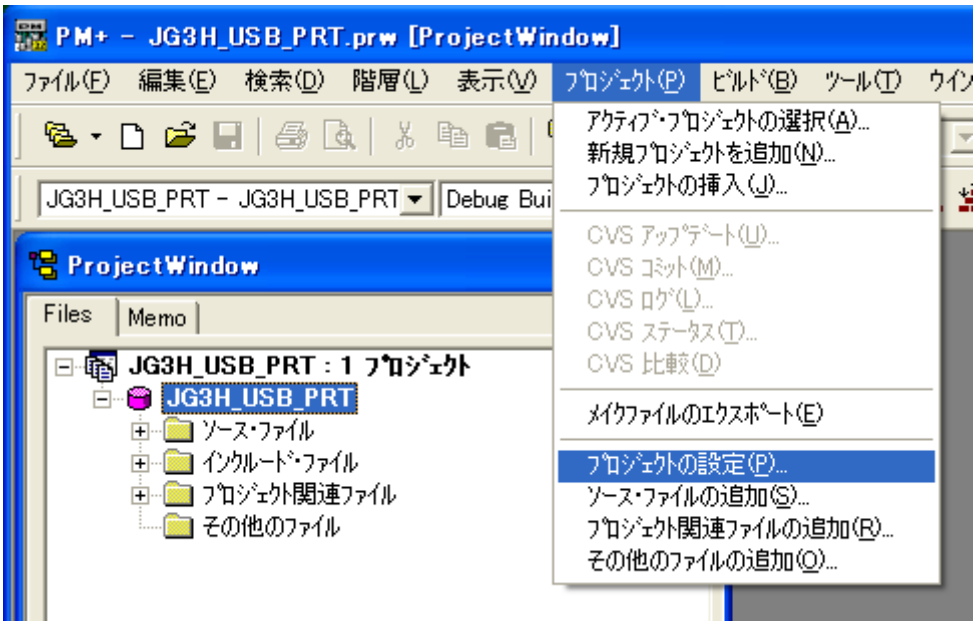
<2> 「ワークスペースを開く」ダイアログが開きます。サンプル・ドライバを格納したディレクトリの「NEC_project」フォルダにあるワークスペース・ファイルを指定します。



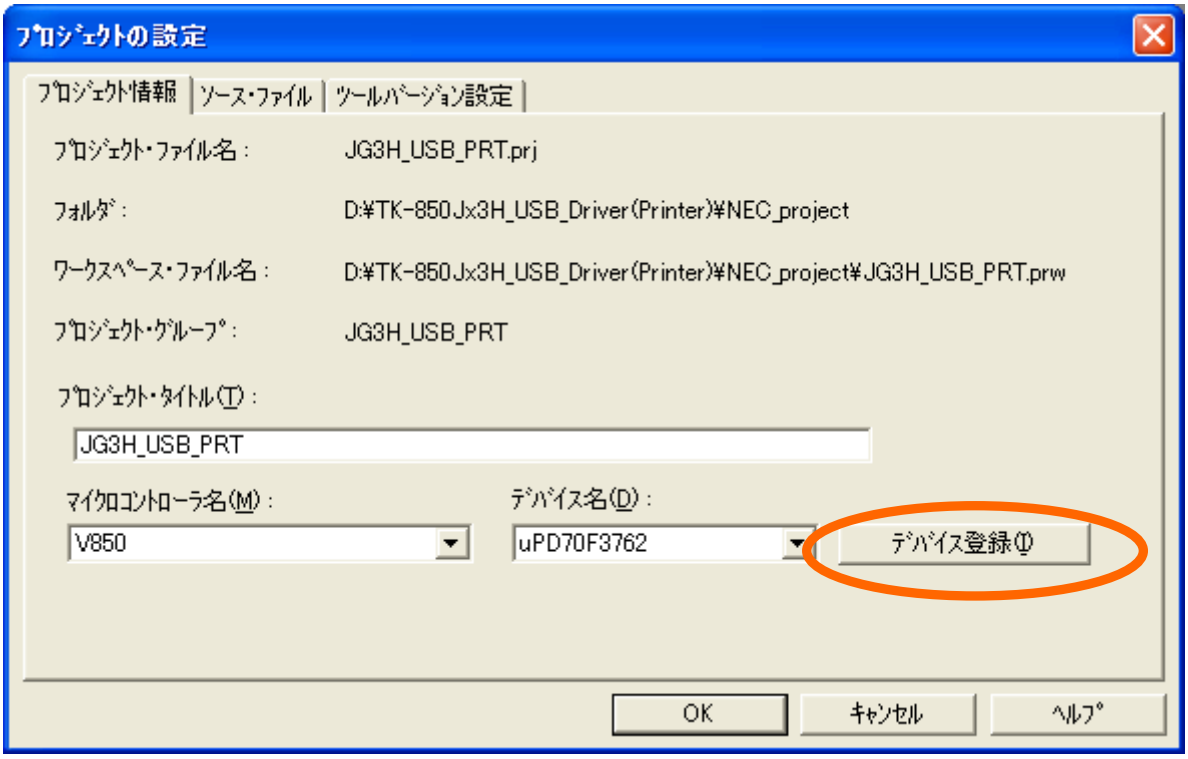
(4)　デバイス・ファイルのインストール

ここではV850ES/JG3-H用のデバイス・ファイルを使用する場合の手順を示します。

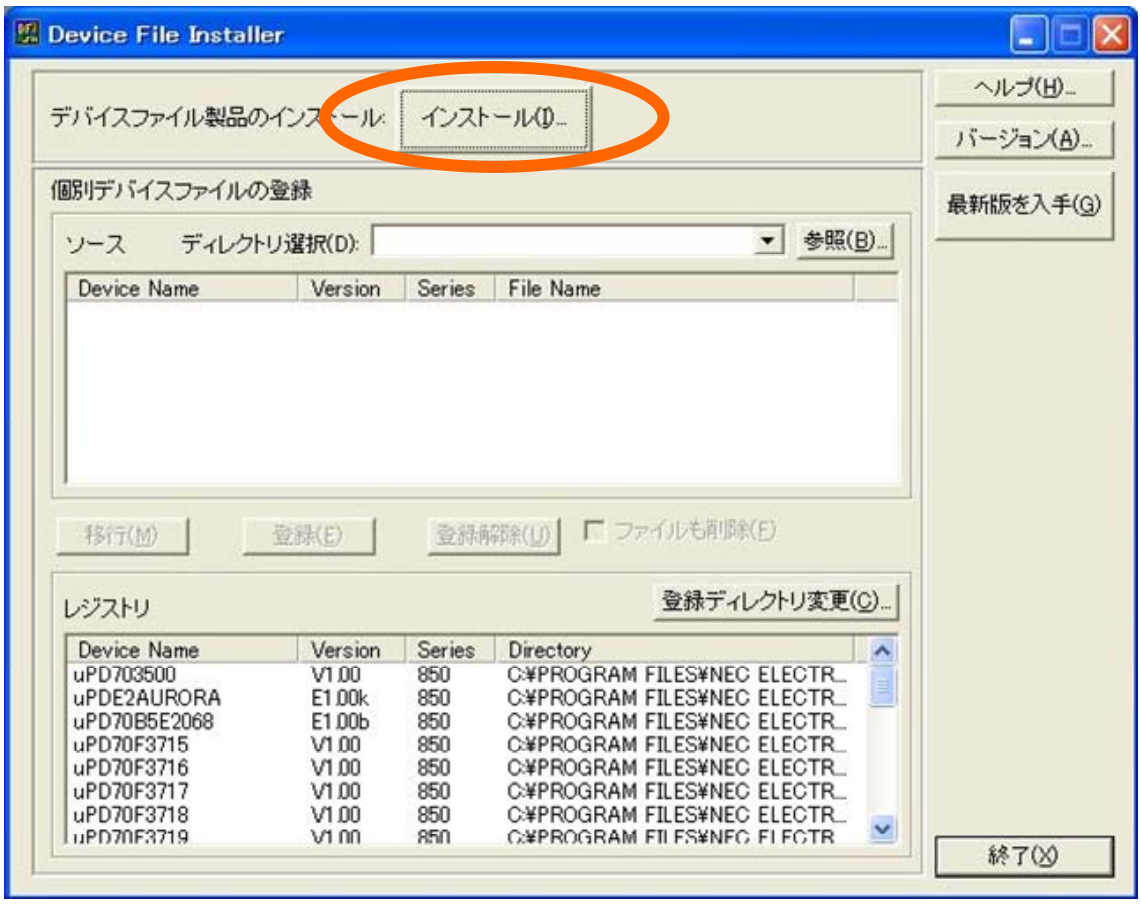
<1> PM+の「プロジェクト」メニューから「プロジェクトの設定」を選択します。



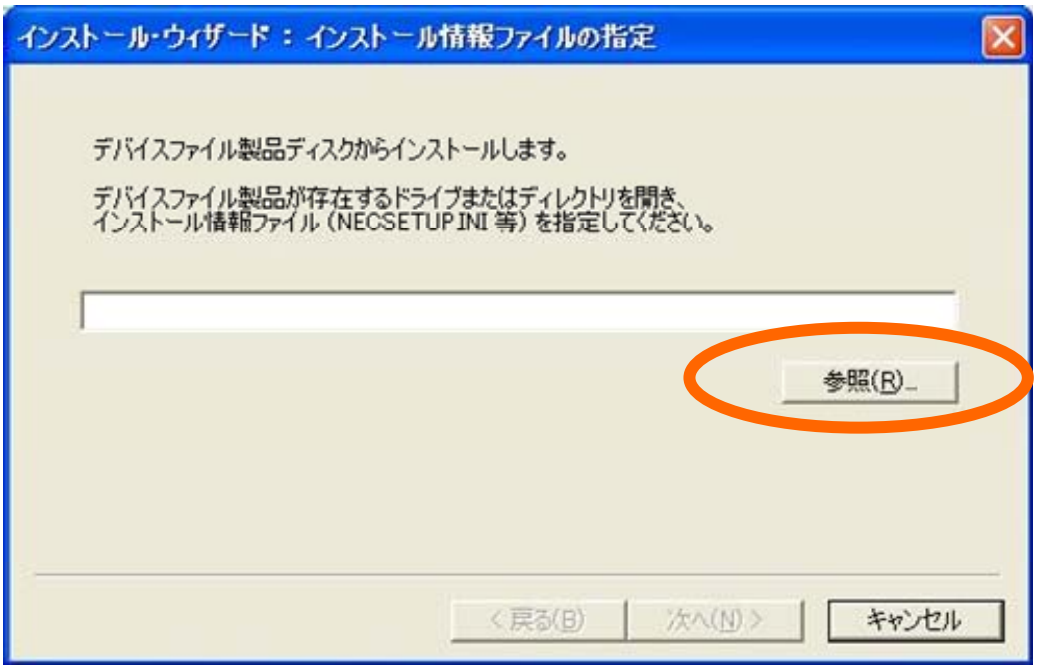
<2> 「プロジェクトの設定」ダイアログが開きます。「プロジェクト情報」タブの「デバイス登録」ボタンを押下してDevice File Installerを起動します。



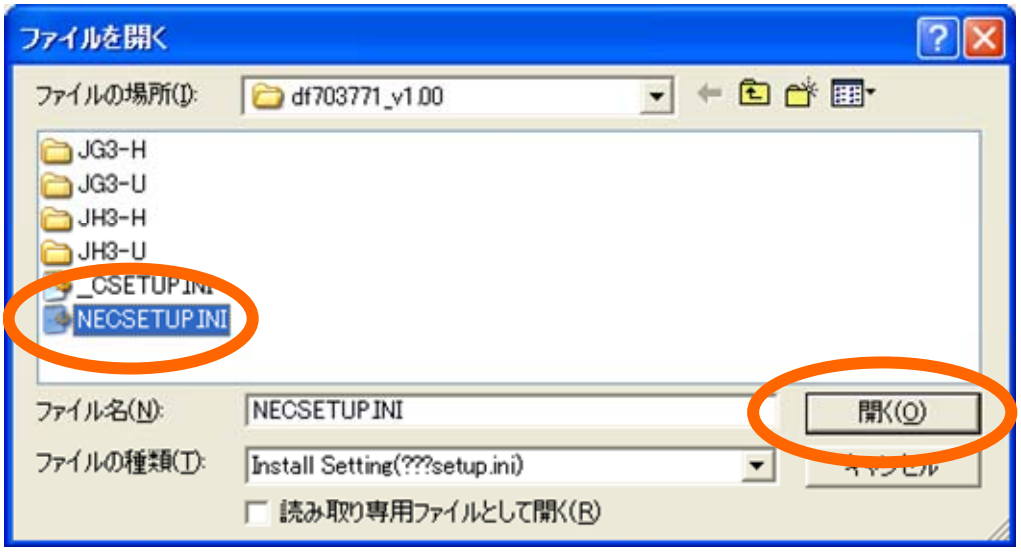
<3> 「Device File Installer」ダイアログが開きます。「インストール」ボタンを押下してインストール・ウィザードを起動します。



<4> 「インストール情報ファイルの指定」ダイアログが開きます。「参照」ボタンを押下します。

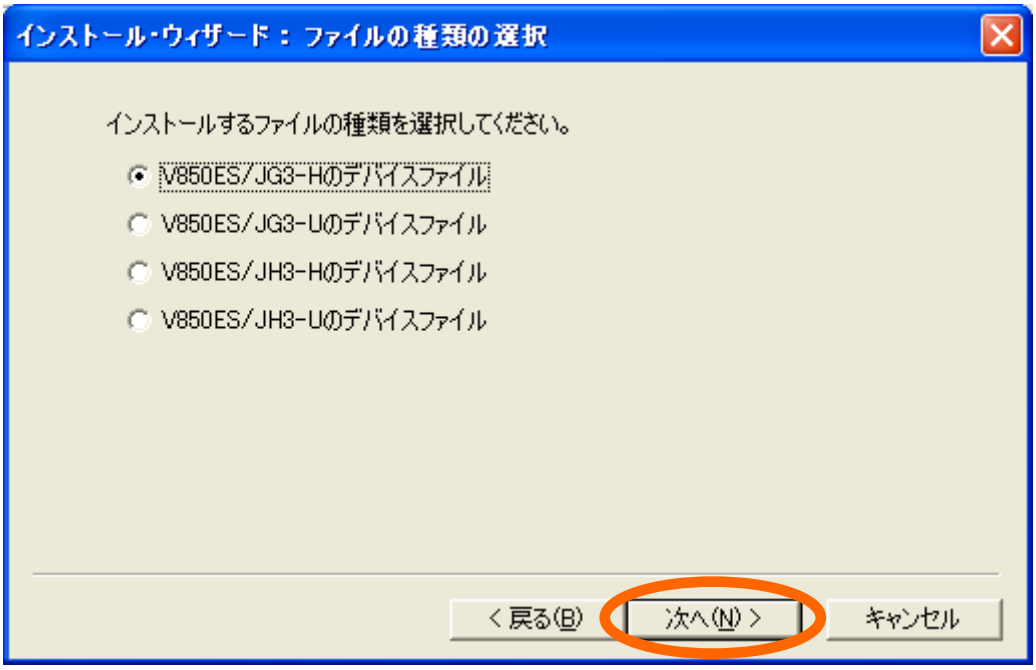


<5> 「ファイルを開く」ダイアログが表示されます。デバイス・ファイルを格納したディレクトリを開き，"NECSETUP.INI" ファイルを選択して「開く」ボタンを押下します。

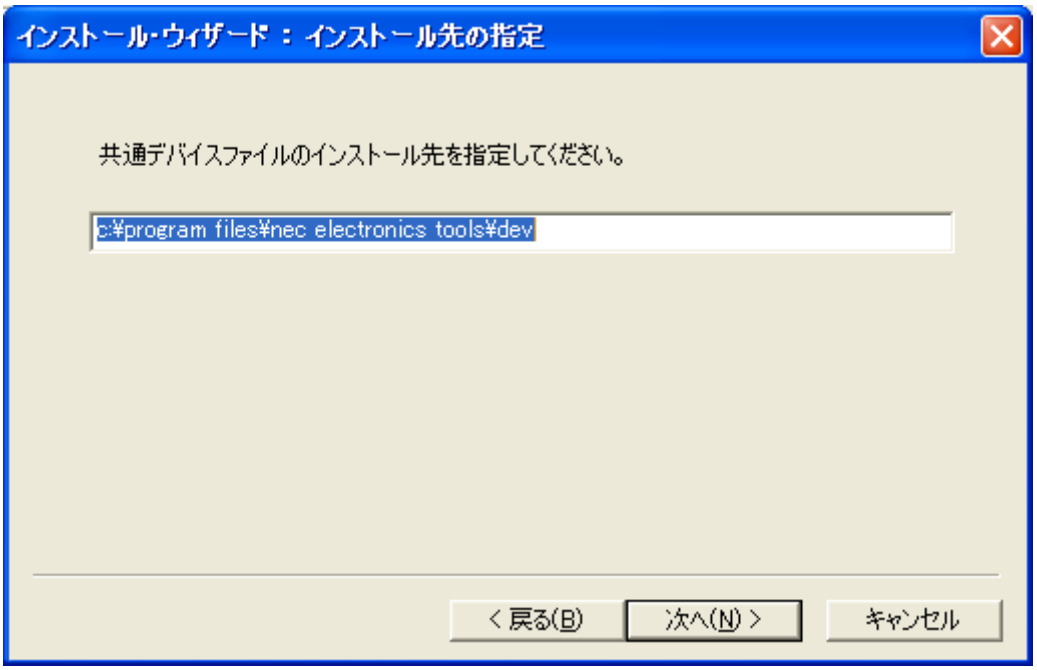


<6> 使用許諾に関するメッセージが表示されます。「次へ」ボタンを押下します。

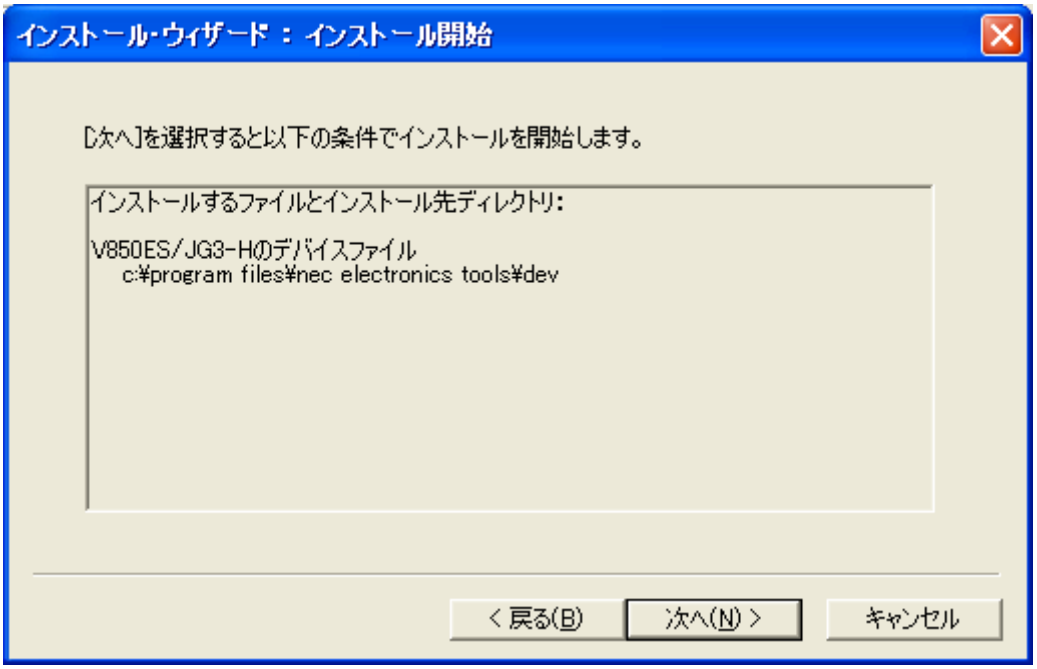
<7> 「ファイルの種類の選択」ダイアログが開きます。該当するデバイス・ファイルを選択して「次へ」ボタンを押下します。



<8> 「インストール先の指定」ダイアログが開きます。パスが表示されていますので，そのまま「次へ」ボタンを押下します。

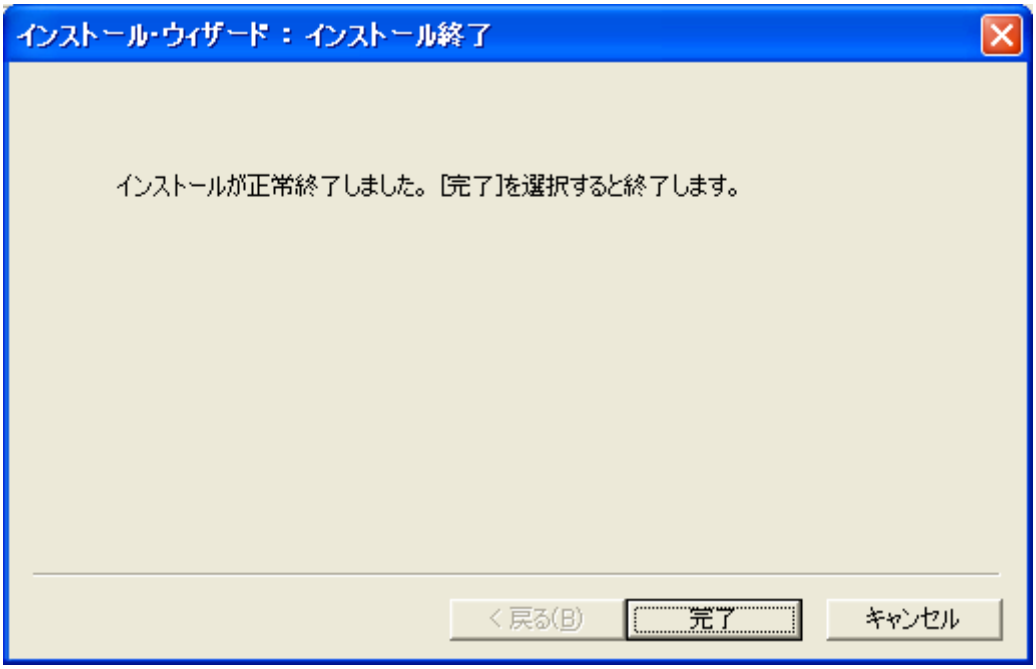


<9> 「インストール開始」ダイアログが開きます。「次へ」ボタンを押下します。



<10> デバイス・ファイルがプロジェクトにインストールされます。環境により，時間がかかる場合があります。

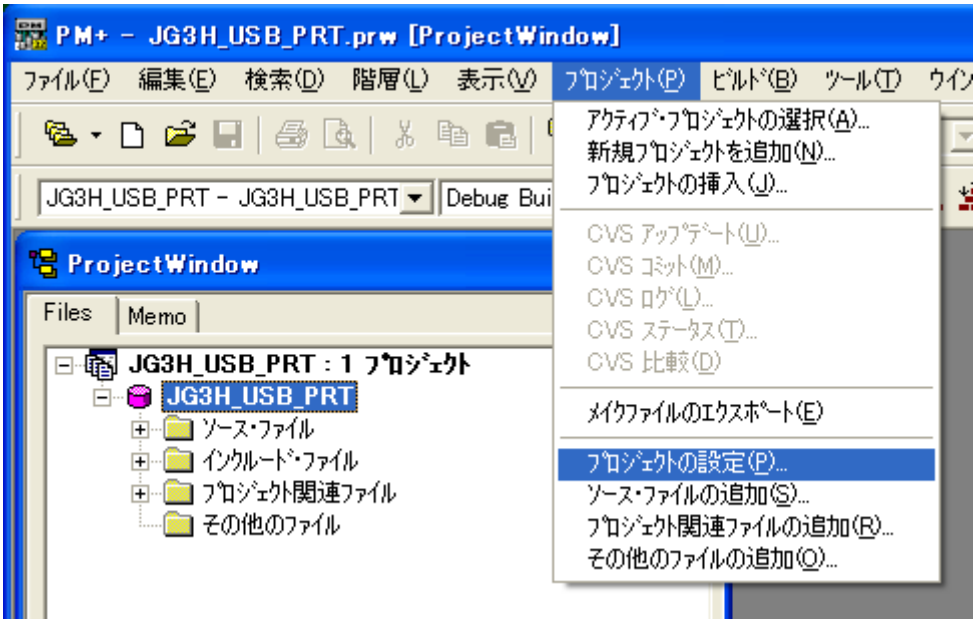
<11> インストールが終わると「インストール終了」ダイアログが開きます。「完了」ボタンを押下します。



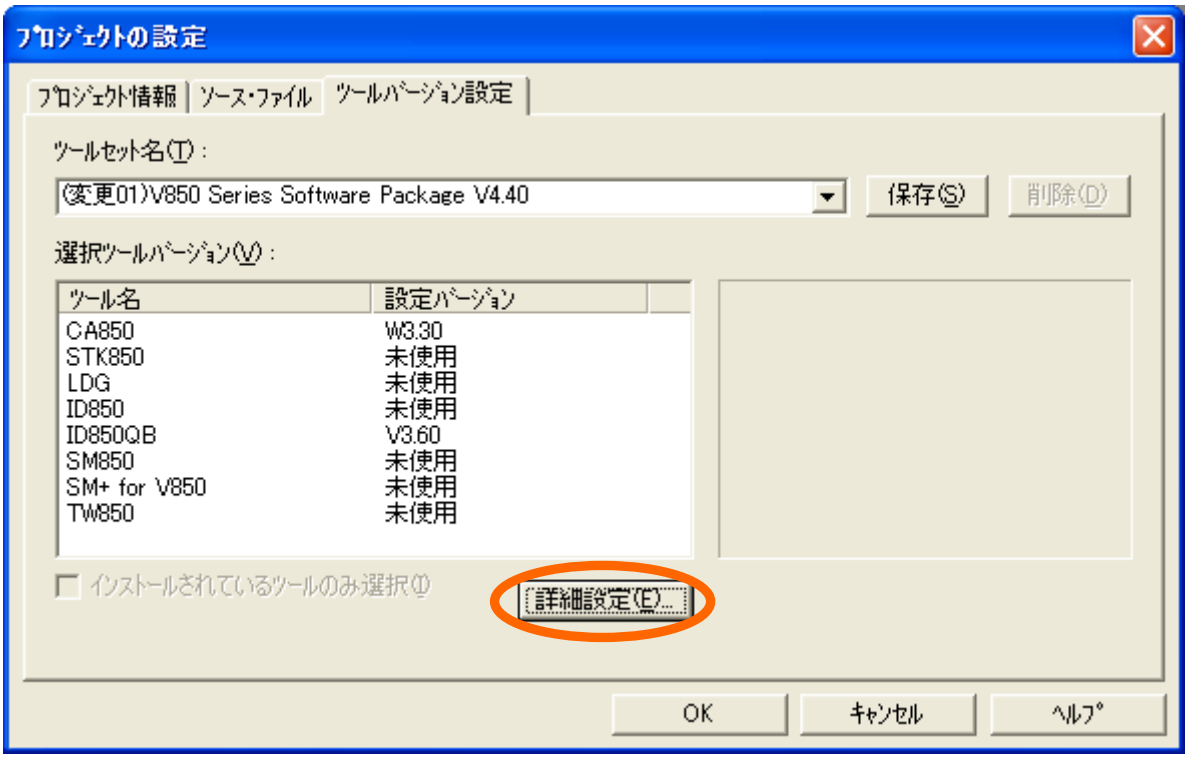
(5)　ビルド・ツールの設定

ここではビルド・ツールとしてCA850を，デバッグ・ツールとしてID850QBを使用する場合の手順を示します。

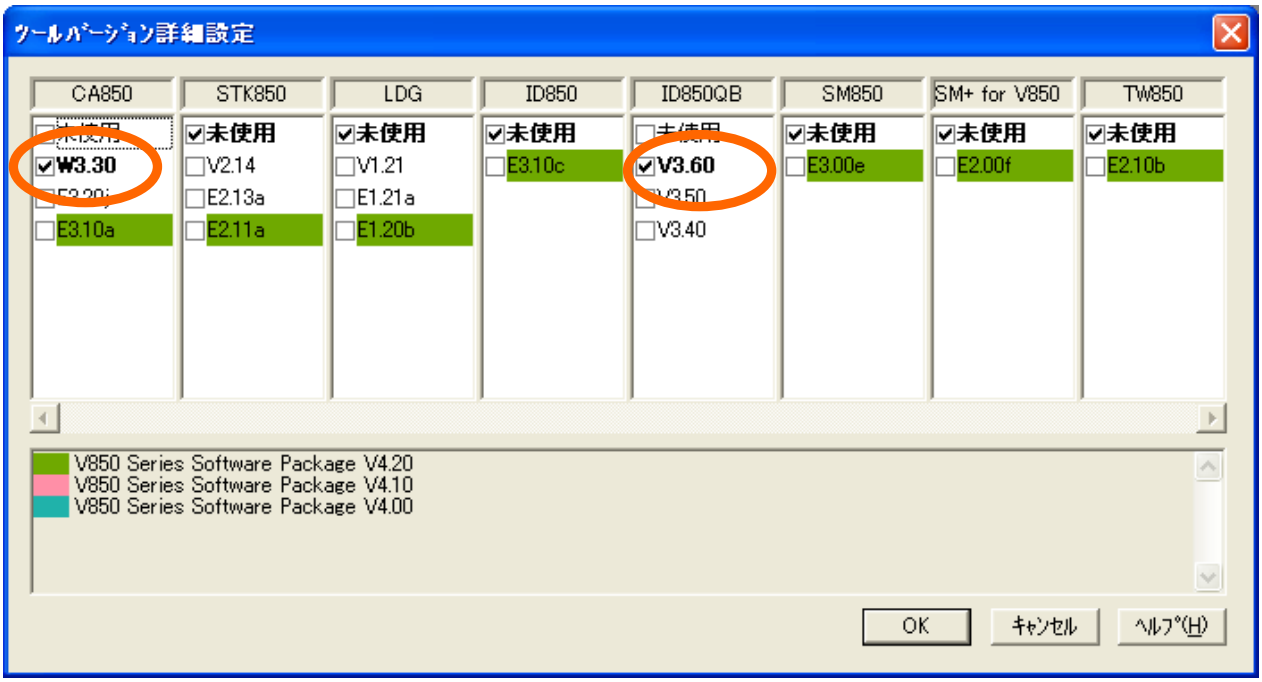
<1> PM+の「プロジェクト」メニューから「プロジェクトの設定」を選択します。



<2> 「プロジェクトの設定」ダイアログが開きます。「ツールバージョン設定」タブの「詳細設定」ボタンを押下します。



<3> 「ツールバージョン詳細設定」ダイアログが開きます。「CA850」の欄で使用するコンパイラのバージョンを，「ID850QB」の欄で使用するデバッガのバージョンを選択します。



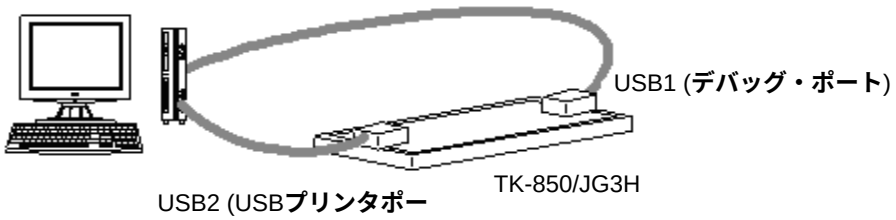
5. 2. 2　ターゲット環境整備

デバッグに使用するターゲット・デバイスを接続します。

(1)　ターゲット・デバイスの接続

TK-850/JG3Hの2つのUSBポートとホスト・マシンのUSBポートをUSBケーブルでそれぞれ接続します。

図 5-2　TK-850/JG3Hの接続



備考　TK-850/JG3Hの外形やポートについては，付録A　スタータ・キットを参照してください。

備考　TK-850/JG3Hの一方のUSBポートはデバッグ・ポートで，別途専用のホスト・ドライバ(スタータ・キットに付属)が必要です。

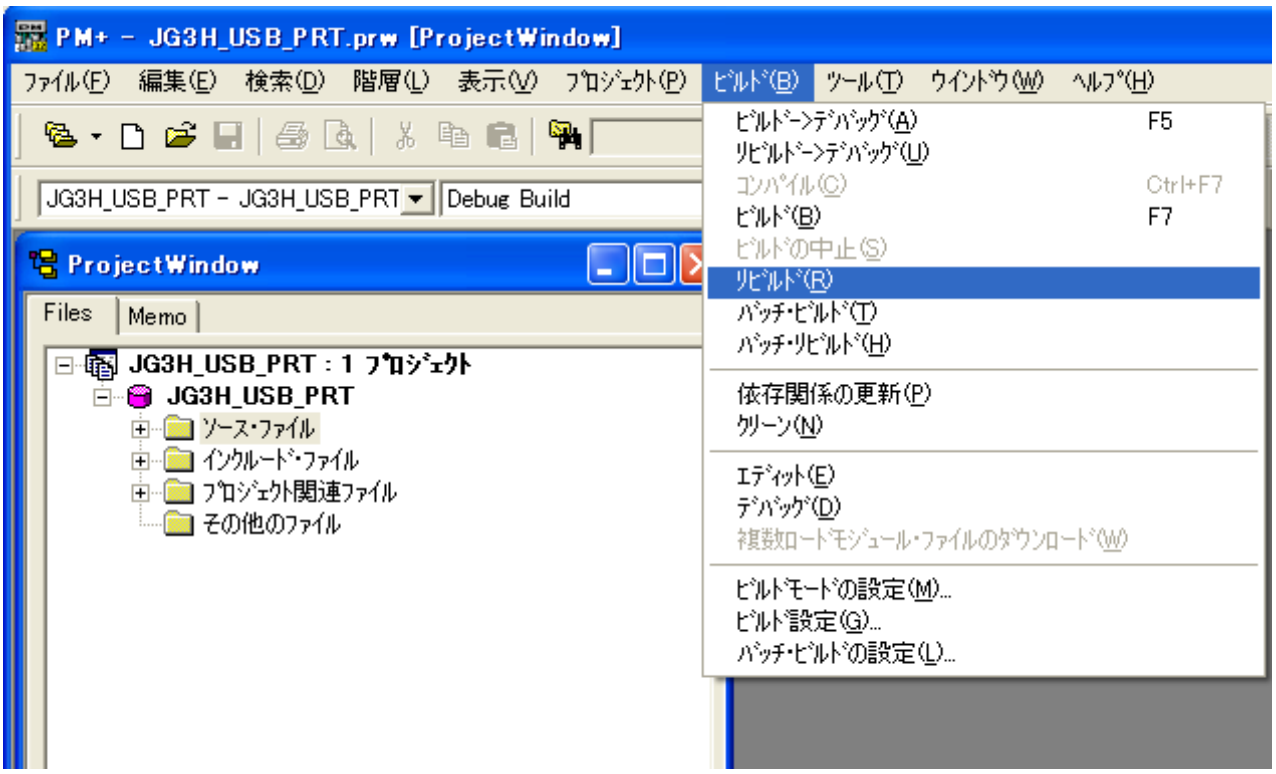
5. 3　オンチップ・デバッグ

ここでは，5. 2 環境設定に示したワークスペースでのデバッグ手順について説明します。V850ES/Jx3-Hでは，内蔵のフラッシュ・メモリにプログラムを書き込み，デバッガなどから実行させての動作検証（オンチップ・デバッグ）が可能です。

5. 3. 1　ロード・モジュール生成

ターゲット・デバイスにプログラムを書き込むには，C言語やアセンブリ言語で記述されたファイルをCコンパイラなどで変換してロード・モジュールを生成します。

PM+では，「ビルド」メニューから「リビルド」を選択すると，ロード・モジュールが生成されます。



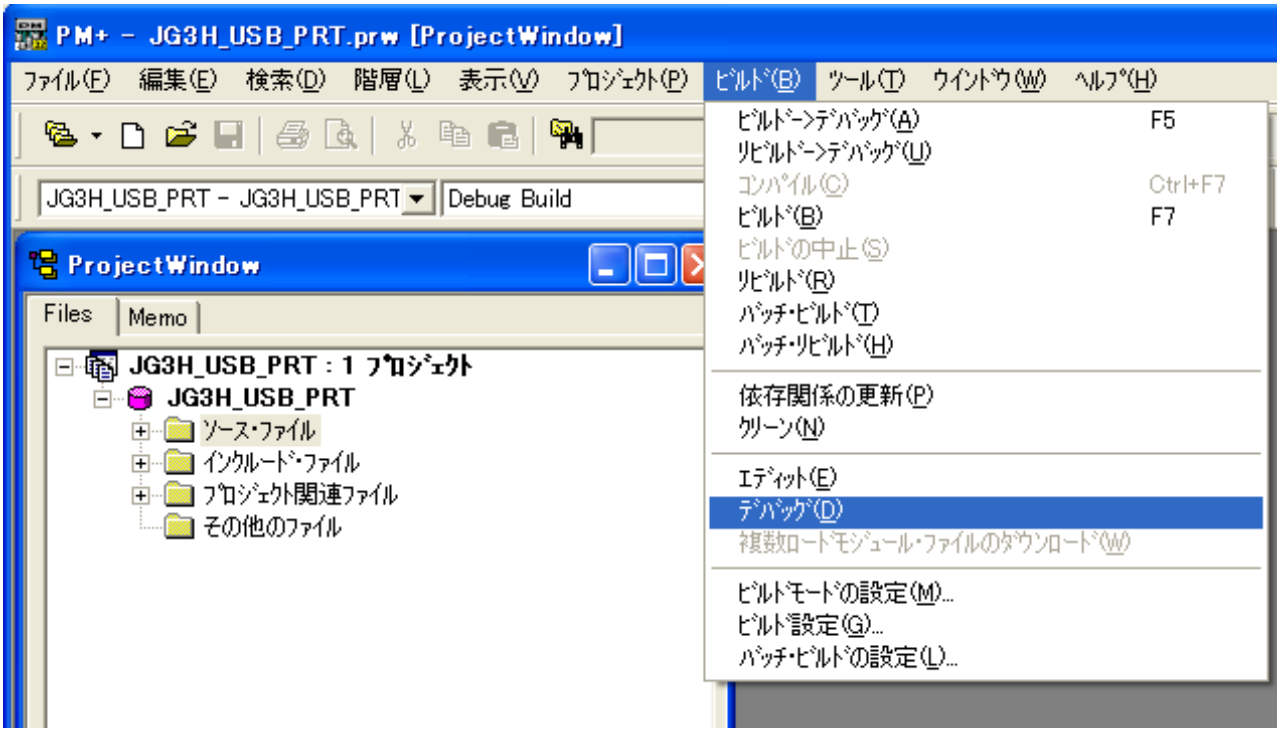
5. 3. 2　ロードと実行

生成したロード・モジュールをターゲットに書き込んで（ロード）実行させます。

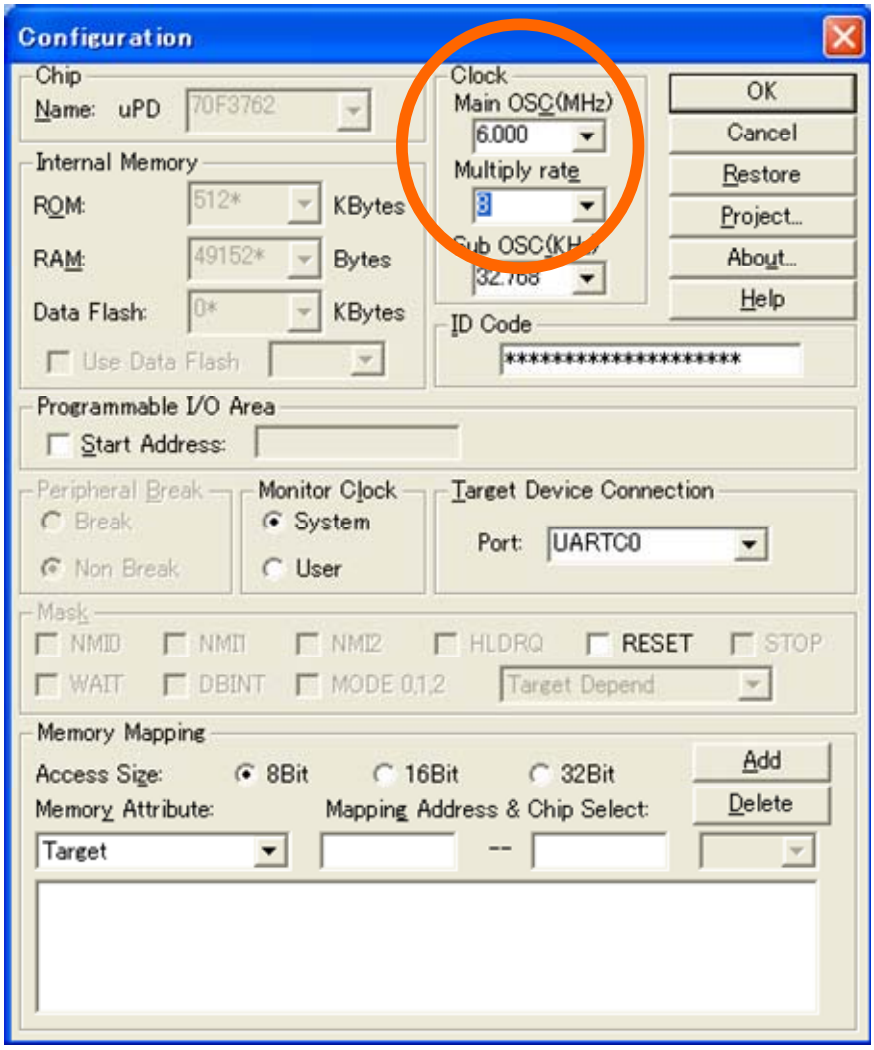
(1)　ロード・モジュールの書き込み

ここではPM+を介してTK-850/JG3Hにロード・モジュールを書き込む手順を示します。

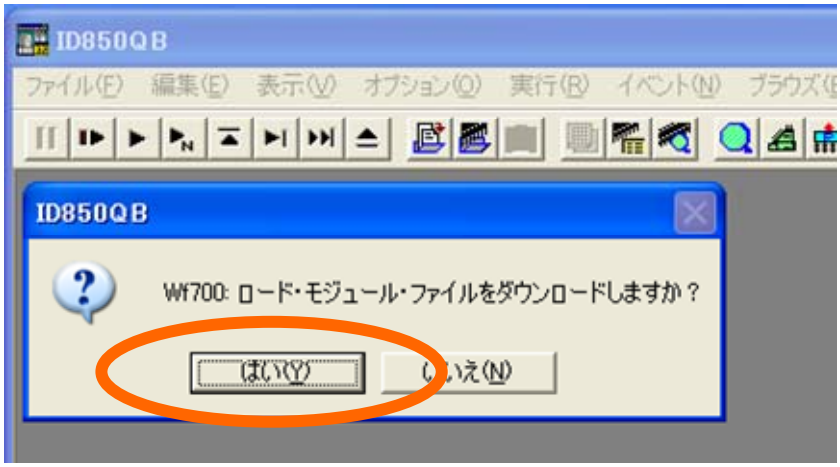
<1>　「ビルド」メニューから「デバッグ」を選択してID850QBを起動します。



<2> 「Configuration」ダイアログが開きます。「Main OSC」欄に "6.000" (MHz) を，「Multiply rate」欄に "8" を設定します。

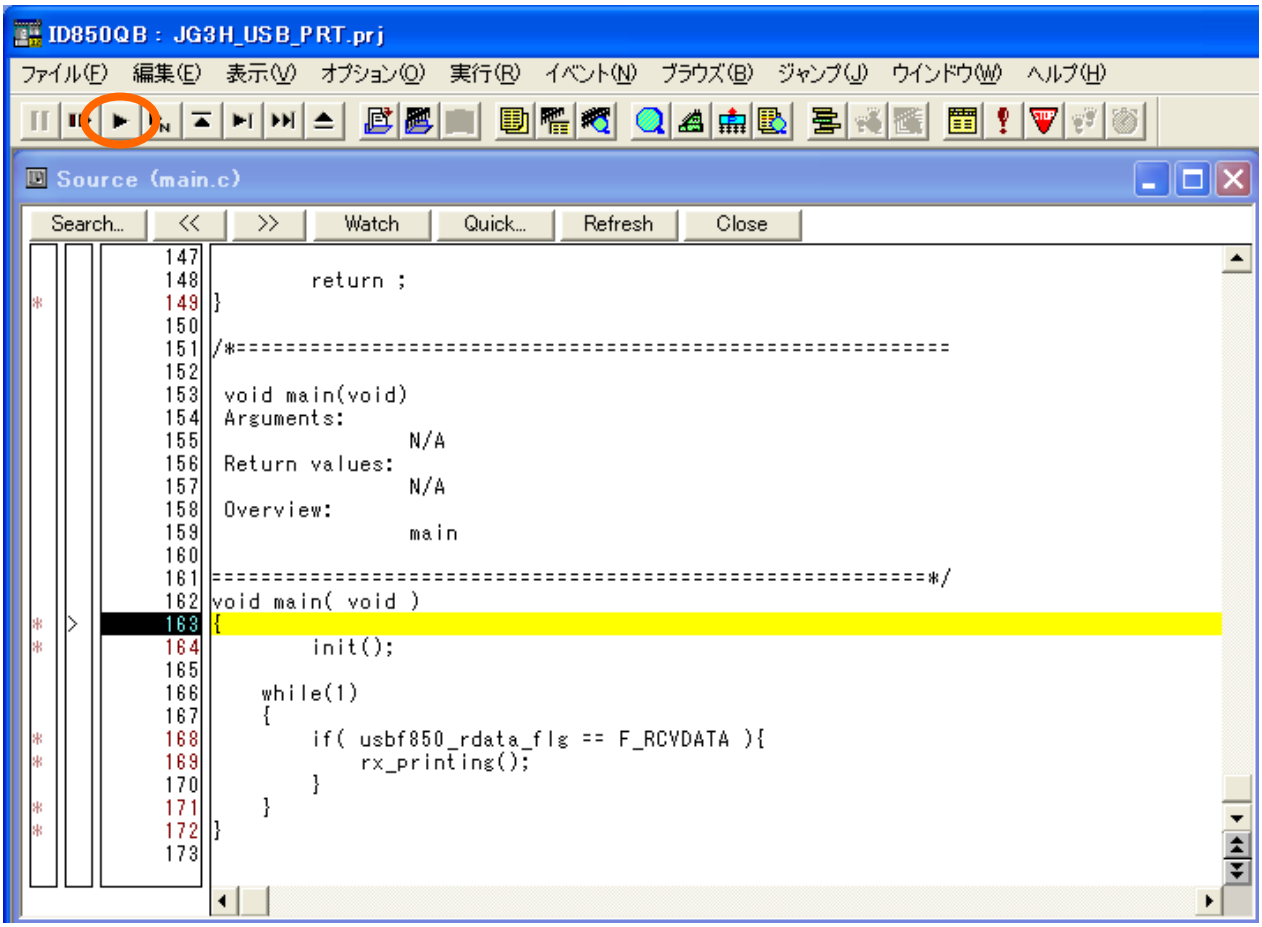


<3> サンプル・ドライバに同梱のプロジェクト・ファイルを使用している場合は，次の画面が表示されます。「はい」ボタンを押下してロード・モジュール・ファイルの書き込みを開始させます。



(2)　プログラムの実行

ID850QBの  ボタンを押下します。または「実行」メニューから「継続して実行」を選択します。



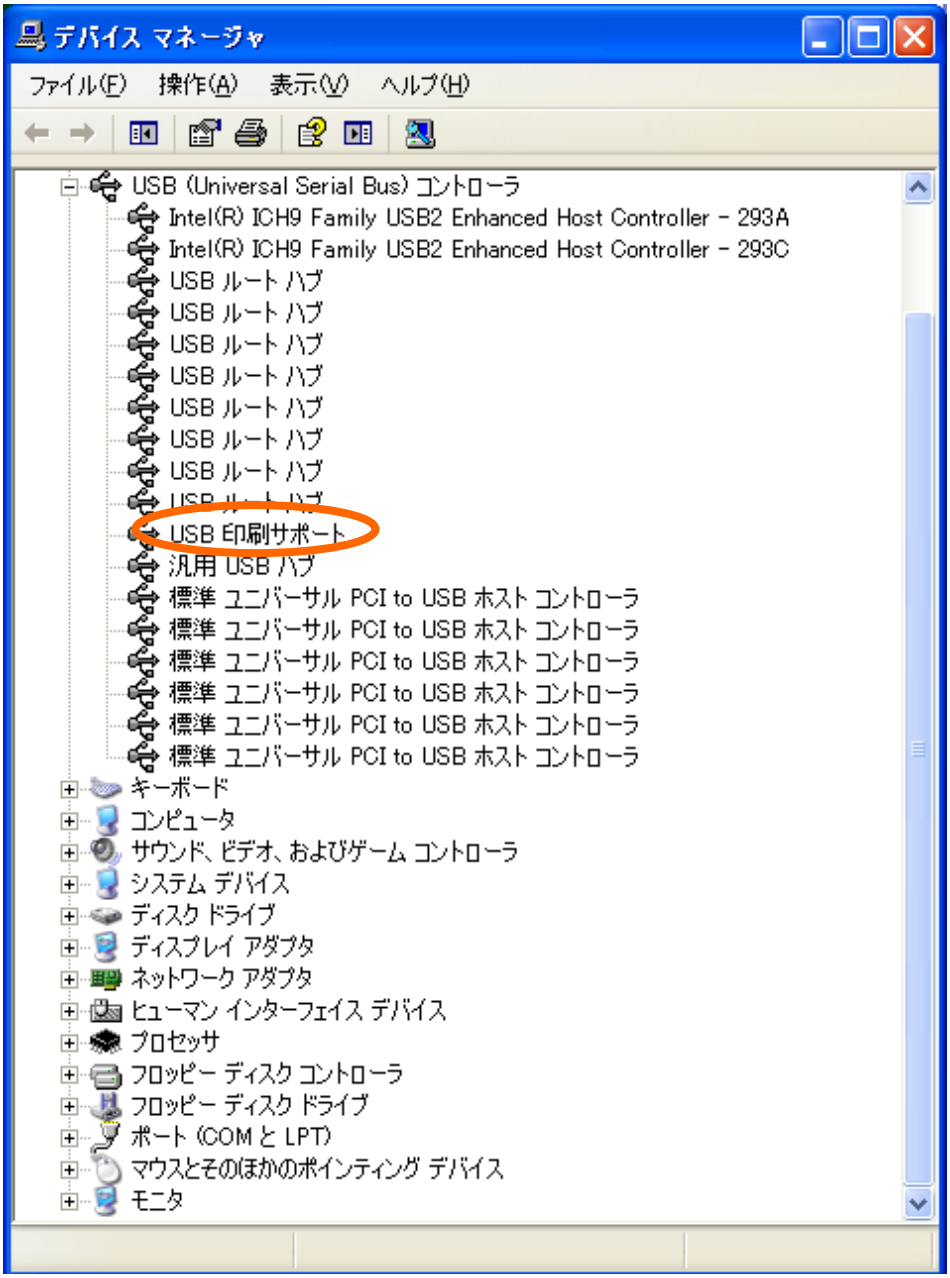
5. 4　動作確認

サンプル・ドライバをロードしたターゲット・デバイスとホスト・マシンがUSBで接続されていれば，ドライバ内のサンプル・アプリケーションの実行結果を確認できます。

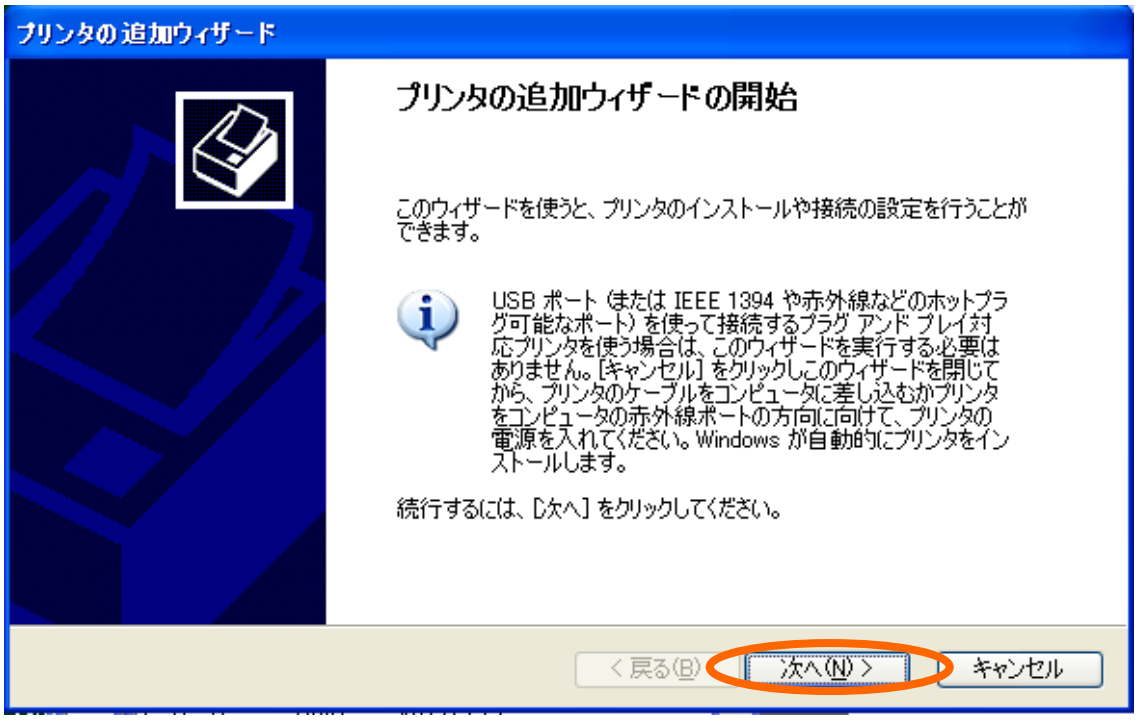
(1)　プリンタドライバのインストール

ここではサンプル･ドライバで印刷動作を確認する為に、プリンタドライバをインストールする手順を示します。

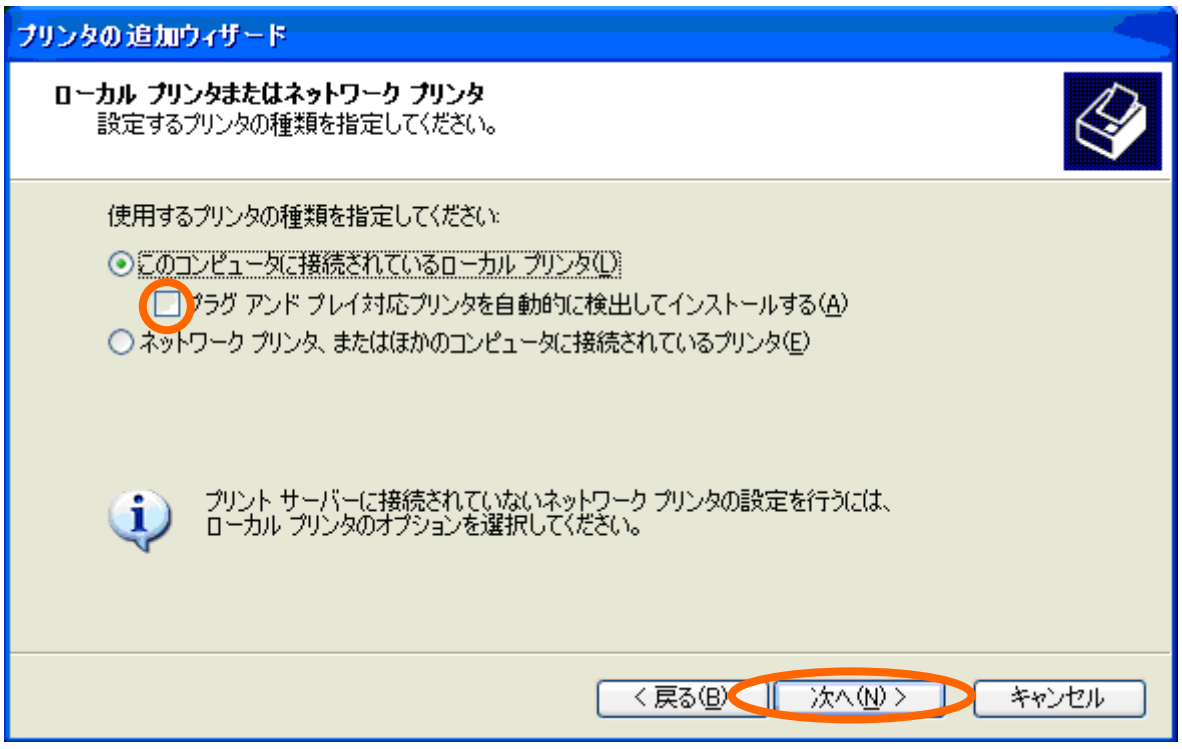
<1> TK-850/JG3Hの接続がホスト・マシンに認識されると，デバイスマネージャ上に「USB印刷サポート」というデバイスが表示されます。



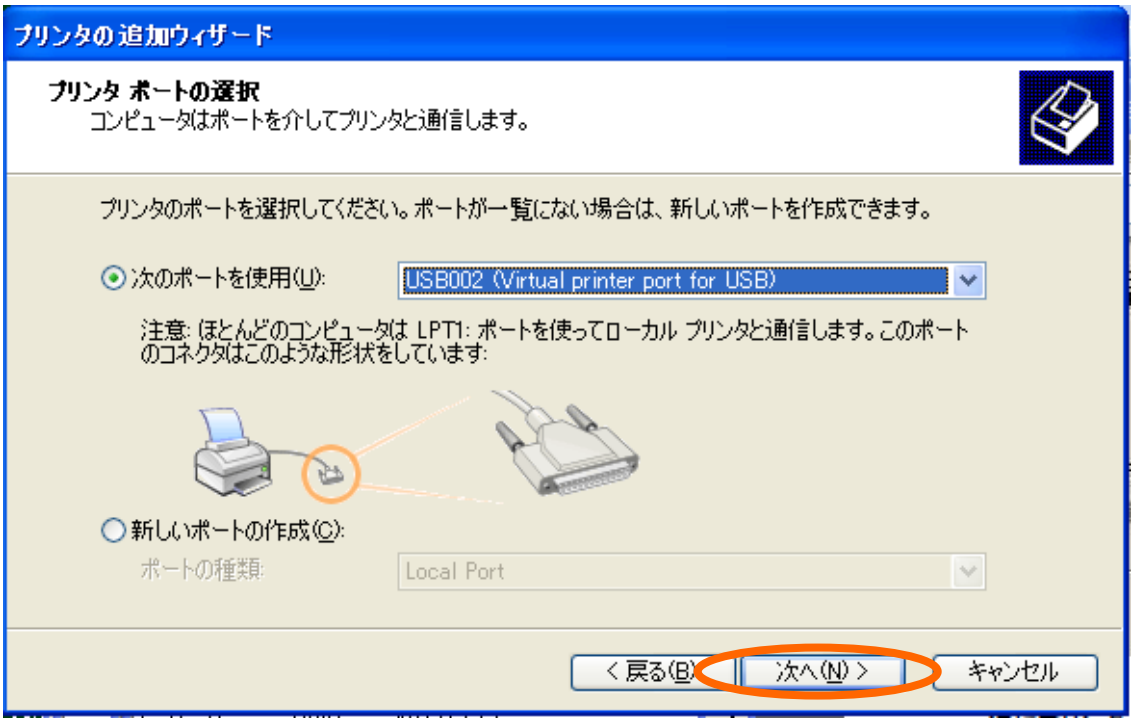
<2>コントロール・パネルの「プリンタとFAX」から「プリンタのタスク」の「プリンタのインストール」を選択してプリンタの追加ウィザードを起動します。



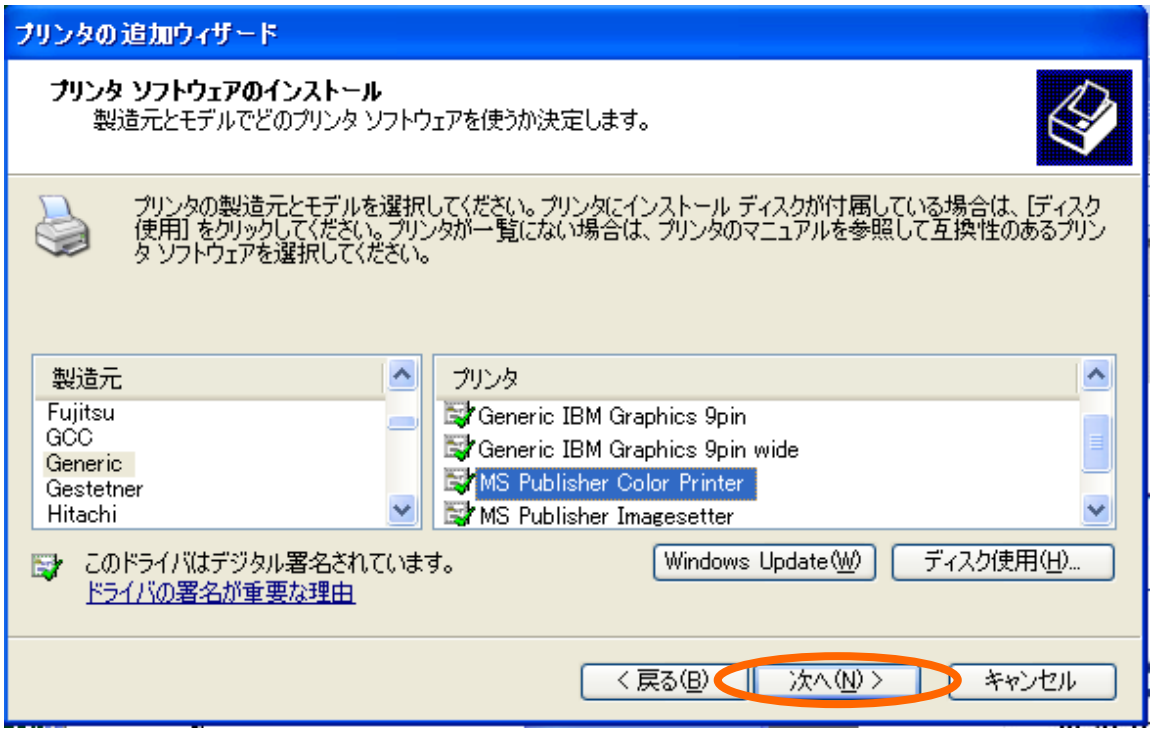
<3> ローカルプリンタまたはネットワークプリンタの選択で、「このコンピュータに接続されているローカルプリンタを」選択し、「プラグアンドプレイ対応プリンタを自動的に検出してインストールする」のチェックを外し、「次へ」のボタンを押して下さい。



<4> **プリンタポートの選択で、「次のポートを使用」を選択し、「**USB00X (Virtual printer port for USB)**」を指定する。**TK-850/JG3H**のUSBを接続して追加されたポートを指定して下さい。**

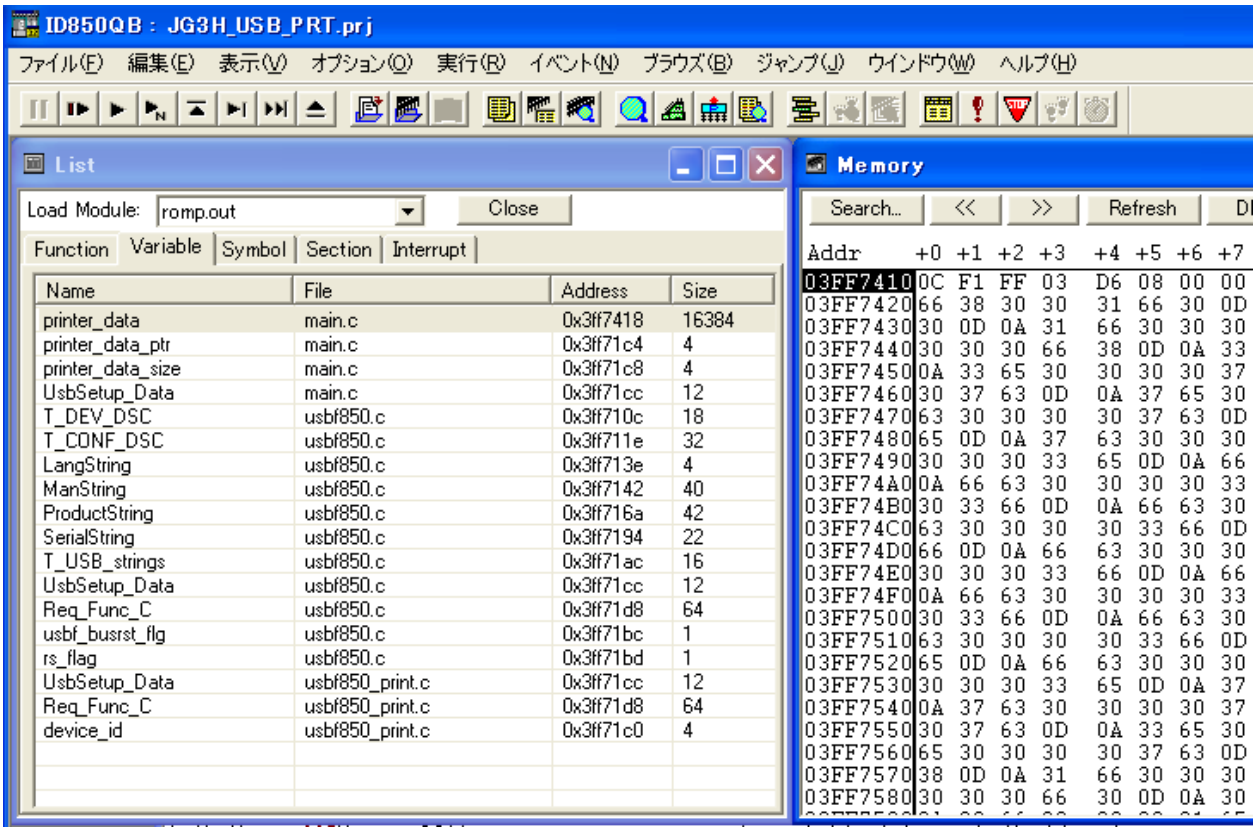


<5> **プリンタソフトウェアのインストールで、「**Generic**」の「**MS Publisher Color Printer**」を選択し、「次へ」ボタンを押下します。これが正常に終了すると、サンプル･ドライバの**USB**プリンタが使用可能になります。**



(2)　印刷データの確認

適当なファイルを、(1)でインストールしたMS Publisher Color Printerを指定して印刷。印刷終了後、ID850QBを停止させ、Memoryウィンドウを開き、Listウィンドウにある変数printer_dataのアドレスが指し示すメモリ値をMemoryウィンドウで参照、印刷データが書き込まれている事を確認します。



第6章 サンプル・ドライバの応用

この章では，V850ES/Jx3-H 向けUSBプリンタ・クラス用サンプル・ドライバを利用する際に，知っておいて頂きたい情報について説明します。

6.1 概　　要

サンプル・ドライバ利用の際に，通常、編集する必要が生じるのは以下の部分になります。

(1) カスタマイズ

次に示す部分を必要に応じて書き換えます。

- "main.c", "main.h" ファイル内のサンプル・アプリケーション部
- "usbf850.h" ファイル内の各種レジスタの設定値
- "usbf850_desc.h" ファイル内の各種ディスクリプタの内容

備考 サンプル・ドライバのファイル構成については1. 1. 3 **サンプル・ドライバの構成**を参照してください。

(2) 関数の利用

アプリケーション・プログラム内で必要に応じて呼び出します。実装されている関数の詳細は，3. 3 **関数の仕様**を参照してください。

6.2　カスタマイズ

ここでは，サンプル・ドライバの利用にあたり，必要に応じて書き換える部分について説明します。

6.2.1　アプリケーション部

"main.c" ファイルにあるアプリケーションは，サンプル・ドライバの利用例として簡単な処理を記述したものです。実際にアプリケーションとして必要な処理をここに適宜，記述して下さい。

サンプル・アプリケーションは、(印刷機能を有していないため)受信した印刷データをRAM上に確保した印刷データ格納領域(printer_data)に書き込むだけの機能を提供しています。

印刷データ格納領域(printer_data)とそのサイズ(PRINTER_BUFFER_SIZE)はmain.hの中で定義されています。初期化処理関数やアプリケーション内にあるデータ送受信関数はそのまま利用する事も出来ます。使用の際には **3.3 関数の仕様**， **6.3 関数の利用**を確認して使用して下さい。

6.2.2　レジスタの設定

サンプル・ドライバが使用する（書き込みを行う）レジスタとその設定値は，"usbf850.h" ファイルに定義されています。これらのファイル内の値を実際のアプリケーションでの使用法に合わせて書き換えることで，サンプル・ドライバを通してターゲット・デバイスの動作を設定できます。

6.2.3　ディスクリプタの内容

初期化処理時にサンプル・ドライバがUSBファンクション・コントローラに登録するディスクリプタ・データ（**3.1.3 ディスクリプタの設定**参照）が "usbf850_desc.h" ファイルに定義されています。このファイル内の値を実際のアプリケーションでの仕様に合わせて書き換えることで，ホストが必要なターゲット・デバイスに関する情報を設定します。

プロダクトIDやベンダID、ストリング・ディスクリプタ等はターゲット・デバイスに応じて正しい記述をする様にして下さい。ストリング・ディスクリプタには任意の情報を登録できます。サンプル・ドライバでは製造元や製品情報を定義していますので，適宜，書き換えてください。

6.3　関数の利用

使用頻度と汎用性の高い処理が定義済みの関数として用意されていますので，アプリケーションの記述を単純化でき，コード・サイズの節減にもつながります。各関数の詳細は**3.3 関数の仕様**を参照してください。

(1)　印刷データ受信処理

サンプル・ドライバでは，受信データの取り込みをデータサイズの取得とデータのコピーの2段階に分けて，それぞれの処理を定義した関数を用意しています。実際に受信したサイズを元に受信処理を実行する場合は、受信データサイズを確認することが出来ます。但し、1回の受信処理で処理出来る最大のデータサイズは、1パケットで受信されるデータサイズ以下である事に注意して下さい。サンプル・アプリケーションでは、バッファサイズが決まっている場合の使用例になっており、受信処理関数(usbf850_recv_buf)では、受信データがある場合、使用するエンドポイントから受信したデータを読み出します。

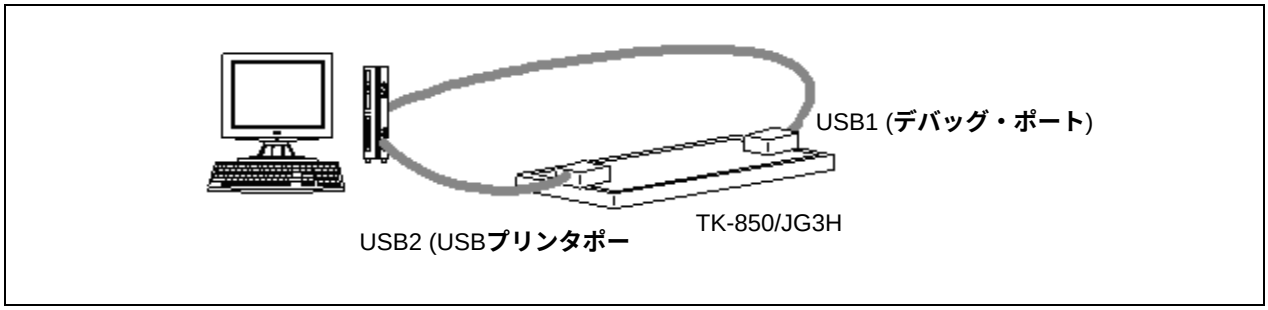
付録A スタータ・キット

この章では，テセラ・テクノロジー社製のV850ES/Jx3-H向けスタータ・キット TK-850/JG3Hについて説明します。

.1 概　　要

TK-850/JG3Hは，V850ES/Jx3-Hを使用したアプリケーション・システムの開発を体験できるキットです。ホスト・マシンに開発ツールやUSBドライバをインストールしてこのキットをUSB接続するだけで，プログラム作成からビルド，デバッグ，動作確認といった一連の開発フローに対応できます。このキットではモニタ・プログラムを使用しており，エミュレータを接続しない状態でのデバッグ（オンチップ・デバッグ）を実現します。

図 A-1　TK-850/JG3Hの接続イメージ



.1.1 特　　徴

TK-850/JG3Hには次のような特徴があります。

- 内蔵USBファンクション・コントローラ用USB miniBコネクタを装備
- 最大で84本のI/Oポートを装備
- コンパクトな名刺サイズ
- 統合開発環境（PM+）と組み合わせて効率的な開発を実現

.2　主な仕様

TK-850/JG3Hの主な仕様は次のとおりです。

○CPU	μPD70F3760（V850ES/JG3-H）
○動作周波数	48 MHz（サブシステム・クロック：32.768 kHz）
○インタフェース	USBコネクタ（miniB）×2基
	N-Wire用コネクタ（パットのみ）
	MINICUBE®2用コネクタ（SICA：パットのみ）
	周辺ボード・コネクタ 2基（パットのみ）
○対応機種	ホスト・マシン：USBインタフェース付きDOS／V機
	OS：Windows 2000，Windows XP
○動作電圧	5.0 V（内部動作3.3 V）
○本体寸法	W89×D52（mm）

〔メ　モ〕

【発行】NEC エレクトロニクス株式会社（http://www.necel.co.jp/）

【問い合わせ先】http://www.necel.com/contact/ja/