PROLOGUE

USB コントローラの種類から USB ソフトウェアの動作まで

USB 機器のハードウェアとソフトウェアの構成

佐藤 陽二/桑野 雅彦

1 USB の特徴

● USB の特徴と組み込み機器におけるメリット

パソコンと周辺機器を接続することを目的とした USB（Universal Serial Bus）の特徴は，組み込み機器においてもメリットとなります．USB の特徴をあげると次のようになります．

（1）高速/高信頼性通信

1.5Mbps のロー・スピードから，12Mbps のフル・スピード，480Mbps のハイ・スピードまで対応しており，しかもデータ転送の信頼性を確保しています．

（2）プラグ＆プレイによる扱いやすさ

特に接続機器ごとの設定作業を必要とせず，電源投入状態でのケーブル挿抜ができます．また，一つのインターフェースで複数の機器と接続可能です．

（3）小型コネクタ，電力供給による省スペース，省電力化

小型コネクタのため組み込み機器における省スペース化に有効で，さらに電力供給も可能なため，小型のデバイスなら電源は不要です．

（4）パソコンとの互換性，親和性の高さ

パソコン用の安価な周辺機器やデバイスが利用可能で，標準クラス対応ならドライバを作成せずに接続が可能です．また，USB メモリなどで簡単にデータ交換ができます．

（5）ベンダ固有の拡張性が高い

ベンダ固有クラスが簡単に定義できるので応用範囲が広くなります．

（6）ホスト/デバイス構成による機能実装の容易さ

ホスト側はやや規模が大きくなりますが，その分，デバイス側は小さくなります．

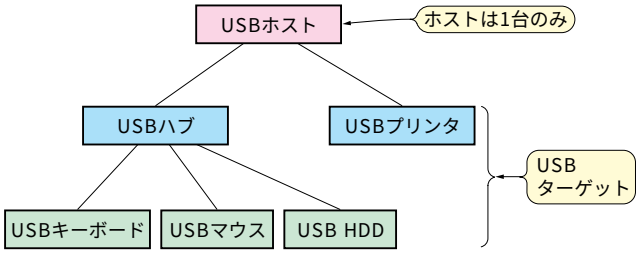

図1　USB のシステム構成

● USB のシステム構成

たとえば，Ethrenet や IEEE1394 などは，基本的にすべての機器が対等の立場でバスや通信路を共有しています（ハブなどの装置を除く）．よって転送を開始するにも，アービトレーション制御や衝突検出といった処理が必要になります．

USB では通信プロトコル上，ホストとターゲット（ファンクションやペリフェラル，単にデバイスなどとも呼ばれる）が明確に分けられています．しかも，一つの USB システム全体の中で，ホストはただ 1 台しか存在できず，このホストを頂点としたツリー構造でバスが構築されます（**図 1**）．

データ転送においても，ホストとターゲットという関係のみで転送が成立します．バス・アイドル状態から最初に転送をしようと行動を起こすことができるのは，ホストのみです．ターゲットは，ホストから自分宛ての転送データ（パケット）を受け取ったら，その内容に応じたデータをホストに返すという処理を行います．しかし，ターゲット間で直接データをやり取りすることはできません．また，ホストからの要求もないのに，ターゲットがいきなりホストにデータを送りつけることもできません．

このように，USB はホストの要求にターゲットが答えるというプロトコルのみを採用しているため，バスのスケジューリングはホストがすべて管理することになります．これにより，バスのアービトレーションや衝突検出といった処理が不要になり，ターゲットに要求されるリソースの負荷が軽くなるのです．

2 USB の活用範囲の広がり

USB 規格は，比較的低価格でパソコンと周辺機器とを簡単につなぐための規格として発表され，現在は 480Mbps の高速通信に対応した USB2.0 規格に進化し，ほぼすべてのパソコンに搭載されている標準的なインターフェースとなりました．こうした流れに乗り，組み込み機器においても USB の活用が進んできました．

▶第 1 段階：パソコンの周辺機器

パソコン側に USB インターフェースが搭載され，その周辺装置として USB インターフェースを搭載した各種組み込み機器（HID，プリンタ，マスストレージ，コミュニケーション，オーディオなど）が登場しました〔**図 2**（**a**）〕．

▶第 2 段階：パソコンと同等のホスト機器

パソコン用に USB デバイス機能を搭載した周辺機器が数多く市場に出回り，その安価かつ入手が容易な機器を利用するた

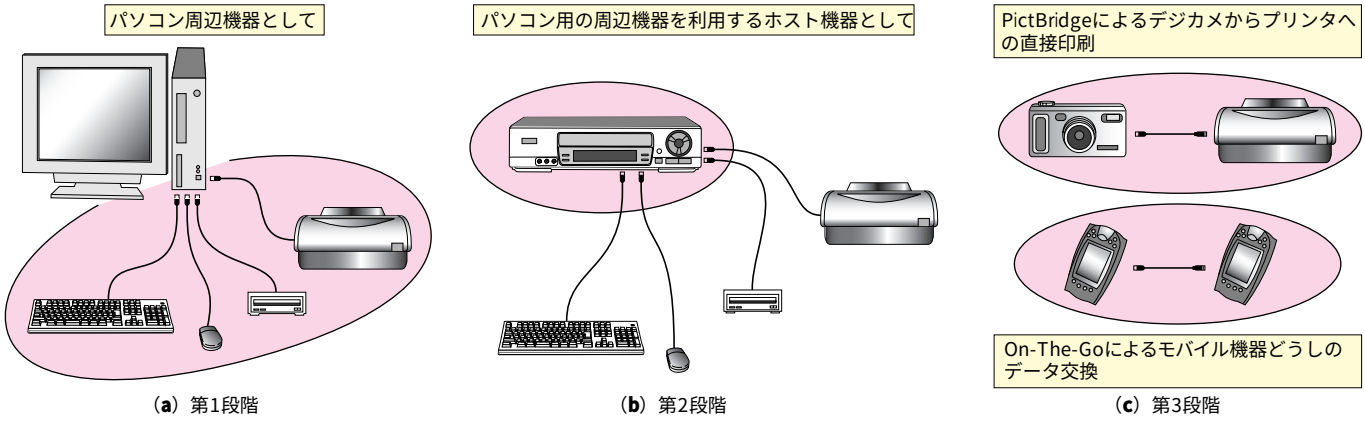

図2　USBの活用段階

め，パソコンと同等のUSBホスト機能を搭載した組み込み機器が登場しました〔**図2**(**b**)〕.

▶第3段階：PCレスの独自通信

これまであくまで周辺機器として存在してきた機器が，現在ではよりインテリジェントに進化してきています．周辺機器の高機能化が進むと，次のような要求も生まれます．

たとえば，PCとPDAをつなぐ場合，PCがホストでPDAがターゲットとなります．しかし，PDA同士を接続することを考えると，ターゲットどうしになるため，接続することができません．さらに，PCとデジカメやプリンタをつなぐ場合，PCがホストでデジカメやプリンタがターゲットとなります．しかし，デジカメとプリンタを直接接続することを考えると，ターゲットどうしになり，やはり接続することができません．

これらの場合，接続しようとするどちらかの機器がホストの役割を演じる必要があります〔**図2**(**c**)〕.

そこで登場してきたのが，On-The-GoやPictBridgeなどの仕様です．第2段階までは，あくまでもパソコン向けに用意された通信規格をPCや組み込み機器へ応用するにとどまっていましたが，On-The-GoやPictBridgeなどの登場により，これまで周辺機器としてしか接続できなかった機器どうしを接続できるようになりました．今後さらにディジタル家電やモバイル機器などへの応用が進むことが期待されます．

3 USB機器の基本構成

● USB機器のハードウェア構成

USB機器を実現するには，一般的にはUSBコントローラを採用し，ソフトウェアと組み合わせてUSB機器を開発します．

当然ながらUSBコントローラには，ホスト用のUSBコントローラ，ターゲット用のUSBコントローラ，そしてどちらの機能も内蔵したホスト/ターゲット両用のUSBコントローラがあります．

(1) USBターゲット・コントローラ

周辺機器としてだけ動作する機器であれば，採用するコントローラはターゲット専用のこのタイプで十分でしょう．このタイプのコントローラには，さらに次のような形態のものがあります．

●ワンチップ・マイコンUSBコントローラ

もっとも規模の小さいUSB機器なら，メモリも内蔵したワンチップ・マイコンといっしょになったUSBコントローラがあります．これ一つでUSB機器を実現できます．

キーボードやマウスなど，ロー・スピード対応のもっともコスト重視のコントローラは，このタイプが多いようです．

●CPU内蔵USBコントローラ(メモリは外付け)

ワンチップ・マイコンではメモリ容量が少ないという場合は，メモリは外付けするが，CPU内蔵周辺機能の一つとしてUSBターゲット・コントローラが内蔵されているCPUもあります．

本特集では，第2章で取り上げているSHやH8内蔵USBコントローラがこのタイプです(一部メモリ内蔵タイプもある).

●USBコントローラ単体

CPUは内蔵していないので，何らかのCPUのローカル・バスに接続して使います．CPUの種類やメモリ容量なども自由に選べるので，もっとも自由度の高いシステムを設計できます．

COLUMN

USB2.0 ≠ハイ・スピード

USB2.0仕様といえばハイ・スピード対応で，USB1.1仕様がフル/ロー・スピード対応と思っている人も多いようですが，そこにはやや誤解があります．

現在USBの最新仕様はUSB2.0であり，フル/ロー・スピードもすべて包含したものとなっています．いまでも便宜的にハイ・スピードをUSB2.0，フル/ロー・スピードをUSB1.1と呼ぶこともありますが，正しい表現としてはUSB2.0ハイ・スピードであり，USB2.0フル/ロー・スピードなのです．よってUSB2.0仕様であってもハイ・スピードに対応していない場合もあります．

New Products ――アイ・オー・データ，1Gバイトの DDR2 SDRAM「DX533-1G」を発売
(株)アイ・オー・データ機器は，1Gバイトの DDR2メモリ(DDR2 Type X)を発売する．アクセス速度は533MHz，CL＝3で240ピンの DIMM．価格は￥120,000．8月下旬から出荷を開始する予定．

本特集では，第1章で取り上げているISP1582が，このタイプです．

ターゲット・コントローラには標準仕様というものはなく，各デバイス・ベンダがそれぞれのコアをシリーズ展開しています．同じファミリであれば，上位互換になっているものが多いので，ファームウェアを共通にするといった設計も可能です．

(2) USBホスト・コントローラ

PCや，つねにホストとして動作する組み込み機器で採用するコントローラです．このタイプにもいくつか種類があります．

ホスト・コントローラには標準仕様がいくつか存在します．Intel製チップセットに内蔵されているUHCI，MicrosoftやPCベンダ主導で規格化されたOHCI，そしてハイ・スピード対応のEHCIがあります．

●PCIバス・ベースのフル仕様タイプ

PC向けチップセットやPCIバスに接続するタイプのUSBホスト・コントローラです．一般的に，UHCI/OHCI/EHCIのいずれかの仕様に準拠しています．

●高機能組み込み向けCPU内蔵タイプ

情報家電機器向けの高機能な32ビットRISCマイコンなどに，周辺機能の一つとしてUSBホスト・コントローラを内蔵したものがあります．コントローラ仕様としては，OHCIに準拠したものが多いようです．

●組み込み向け簡易タイプ/高機能タイプ

このタイプは，実際にはホストとターゲットの両方の機能を内蔵したものが多いようです．

第3章で取り上げているSL811は簡易タイプ，第4章で取り上げているM66596は高機能タイプといえるでしょうか．

(3) On-The-Go対応コントローラ

使用する場面により，ホストにもターゲットにもなりうる機器であれば，このタイプのコントローラを採用します．

第6章で取り上げているML60842や，第7章で取り上げているISP1362がこのタイプです．

前述のホストとターゲットの両機能を内蔵したタイプとの違いは，On-The-Go機能の有無です．前述のタイプのコントローラを採用する場合，機器にターゲット用のコネクタ(タイプB)とホスト用のコネクタ(タイプA)の両方を実装するのが普通です．On-The-Go対応コントローラの場合は，一つのコネクタでホストにもターゲットにも対応できるMini-ABと呼ばれるコネクタを採用できます．

図3
USBソフトウェアの基本構成

● USB機器のソフトウェア構成

ホストであれターゲットであれ，USBのソフトウェアは基本的に**図3**のような階層構造をとります．

(1) USBコントローラ・ドライバ

USBコントローラに依存したドライバ層です．コントローラに対するI/O処理，割り込み処理，コントローラに依存するスケジューリング処理などを行います．

(2) USBドライバ

コントローラやクラスに依存しないUSBの論理的な通信制御処理を受け持つドライバ層です．また，接続されたUSBデバイスの構成を管理する機能も一般的にこの層が管理します．

(3) クラス・ドライバ

接続するUSB機器のクラスに対応したエンドポイントの管理や通信プロトコル処理を行うドライバ層です．

(4) 上位プロトコル

USB転送上にマッピングされた上位のプロトコル層です．USB側から見ればアプリケーションとみなすことができるものですが，USBの応用システムを開発する立場からすれば純粋なアプリケーション・ソフトウェアとは区別することになるので，あえて別の階層として記述しました．

(5) アプリケーション

USB機器を利用したアプリケーション・ソフトウェア層です．

4 Windowsにおける USBシステムの構成

● Windowsにおける USBハードウェアの構成

Windowsの場合，そのほとんどがPCIバス・ベースのUHCI/OHCI/EHCIのホスト・コントローラを搭載しています．また，Windowsマシン自体をUSB周辺機器として使うことはないため，ターゲットとしての機能はありません(PCのUSBポートをつないでファイル転送を実現するような接続ケーブルは，ケーブル自体がUSBターゲットとして動作している)．

● Windowsにおける USBソフトウェアの構成

Windowsにおける USBソフトウェアの構成を**図4**に示します．

Windowsでは，プラグ&プレイ機能により，デバイスが接続されたときに必要なモジュールを検索して動的にロードするようなしくみを持っています．そのため，初めて接続した機器で，まだドライバがインストールされていない場合でも，自動的にドライバ・インストール用のダイアログが表示され，そこで必要なドライバを指定しインストールすることで，初めて接続した機器もそのまま利用できるようになります．また，2回目以降は自動的にそのドライバがロードされるようになり，特に設定は必要ありません．

● Windowsにおける「USBドライバの作成」とは

図4でわかるように，UHCI/OHCI/EHCIの各ホスト・コン

Information ―― Windows CE初のバックドア型トロイの木馬が発見される
(株)シマンテックによると，Windows CEのバックドア型トロイの木馬「Backdoor.Brador.A」が発見された．バックドアの影響を受けるのは，ARMコアCPU搭載のPocketPCのみ．

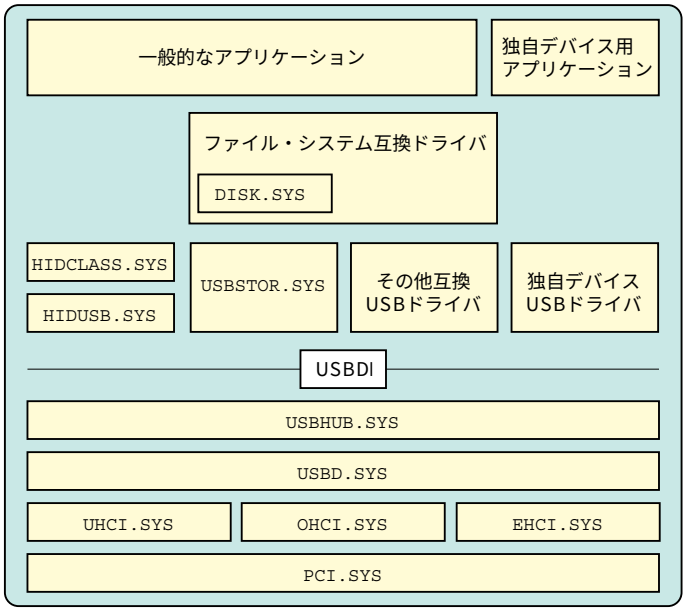

図4　Windows における USB ソフトウェア構成

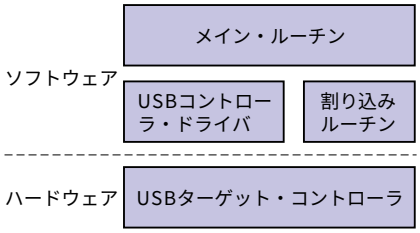

図5　もっとも簡単な USB ターゲットのソフトウェア構成例

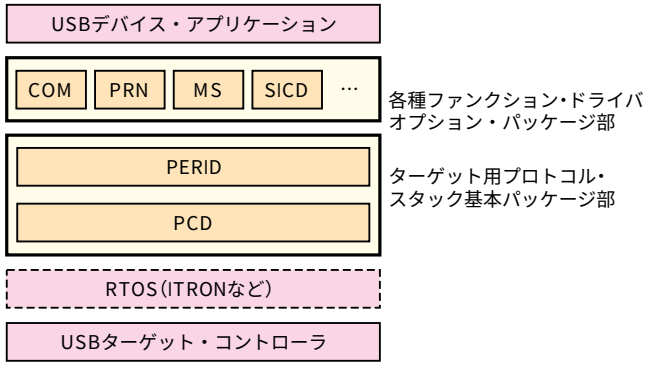

図6　開発効率/移植性を考慮した USB ターゲットのソフトウェア構成

トローラのドライバや USB ドライバ本体，USB ハブ用ドライバなどは，すでに Microsoft 社から OS 標準ドライバという形で提供されています．

よって，Windows 環境で「USB ドライバを作成する」といった場合は，**図4** にある「独自デバイス用 USB ドライバ」から上位階層部分を作成することを意味しています．

● Linux や BSD などの場合

Linux や BSD など規模の大きな OS も，基本的には Windows と同様な階層構造を取っています．これらの OS 上で USB ドライバを開発する場合も，ある部分から上の部分のみを作成すれば良いようになっています．

5 組み込み機器における USB システムの構成

● 組み込み機器における USB ハードウェアの構成

3 節で説明したように，規模や利用形態に合わせて，それに最適な USB コントローラを選択する必要があります．

● 組み込み機器における USB ソフトウェアの構成

組み込み機器はパソコンに比べてハードウェア資源の制約が非常に厳しいものになっています．また，組み込み機器は用途がある程度限定されているため，接続対象とする機器も限られたものとなります．そうしたことから，組み込み機器向けの USB ソフトウェアは，必要最小限の機能をあらかじめ組み込んで静的な構成にするのが一般的です．

▶ USB ターゲットの場合

もっとも簡単な USB ターゲットのソフトウェア構成例を**図5** に示します．USB コントローラに直接アクセスするルーチンと，割り込みルーチン，そしてメイン・ルーチンからなる簡単なもので，RTOS もありません．規模も小さく，実験/試験的な場合や趣味的な開発であれば，これでも十分でしょう．しかし，メイン・ルーチン部分をほかの USB コントローラに移植しようとすると，USB コントローラの構造の違いからくる仕様変更などで，たいへんな手間がかかることもあるでしょう．

開発効率や移植性を考慮した，より本格的な USB ターゲットのソフトウェア構成例を**図6** に示します．規模が大きくなれば，USB の処理だけでなく，それ以外の処理も増えてくるので，RTOS によるタスク切り替えなどは必須となってくるでしょう．

USB の場合，ホスト側に比べデバイス側の処理がそれほど多くなく，使用するコントローラによってはほとんどソフトウェアが介在する必要がない場合もあります．ただし，最近の USB デバイス機器では複数の機能を搭載したり，USB の上位アプリケーションの規模が大きくなる傾向があります．そうしたことから上位のソフトウェアの再利用性を高めるために汎用の USB デバイス・ドライバを利用するケースが増えてきています．

▶ USB ホストの場合

組み込み機器における USB ホストのソフトウェア構成例を**図7** に示します．

基本的には Windows と類似したソフトウェアの階層構造をとりますが，前述のとおり，あらかじめ必要なモジュールを組み込んだ静的な構造となり，サポート対象外の機器は接続できません．

▶ On-The-Go の場合

On-The-Go 対応の場合のソフトウェア構成例を**図8** に示しま

す．On-The-Goの場合，前述のホストおよびターゲット両方のスタックのほか，On-The-Go規格に準拠したSRP，HNPプロトコル処理およびホスト/デバイスの切り替え処理を行うためのOTGスタックが搭載されます．

● 組み込み機器における「USBドライバの作成」とは

組み込み機器でも，規模の大きなOSを採用した場合は，OSにあらかじめUSBのドライバが用意されています．この場合はWindowsなどと同様に，ユーザは独自のクラス・ドライバやその上位アプリケーション部分を作成することになります．

ITRONなど比較的軽い仕様のOSを採用した場合は，USB関連ドライバは含まれていない場合が多いので，ミドルウェア・メーカなどから販売されているUSB用プロトコル・スタックを購入し，その上に独自クラスのドライバやアプリケーションを作成します．仕様の軽いUSBターゲット機器では，足回りのドライバからすべてを自社開発する場合もあるでしょう．

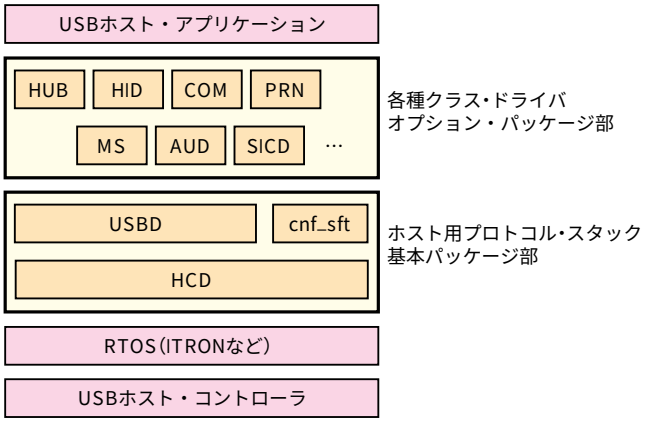

図7　組み込み機器によるUSBホストのソフトウェア構成例

6 各種用途におけるソフトウェア構成例

● キーボード/マウス（HIDクラス）

キーボード/マウスなどのHIDを使用するUSBホストの構成例を**図9**に示します．

キーボード/マウスを使用するためにはHIDクラス・ドライバが必要です．HIDクラス・ドライバでキーボード/マウスに関してどこまで処理しているのかは，ミドルウェア・ベンダにより実装が異なるので個別に確認が必要です．

また，キーボードとマウスを同時に接続して使用する場合にはハブが必要になり，一般にハブ・クラス・ドライバが必要となります．なお，USBコントローラによっては，複数のポートをサポートしており，そのポート数まではハブが必要ないものもあります．

● シリアル通信（Communicationクラス）

USBでシリアル通信をエミュレートするようなシステム構成例を**図10**に示します．

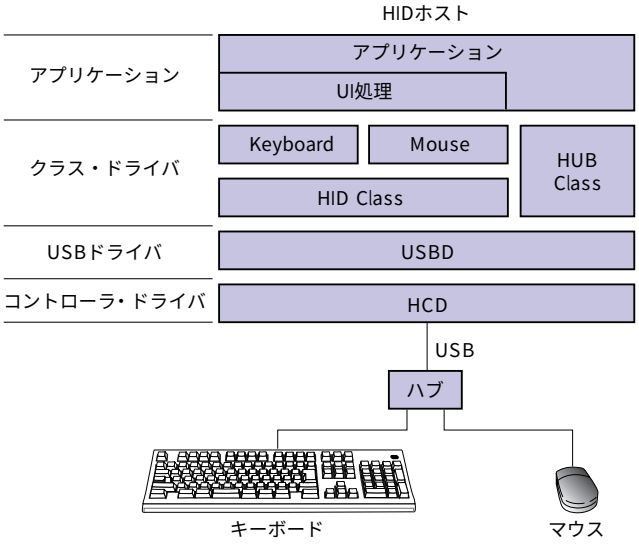

図9　HIDを使用するUSBホストの構成例

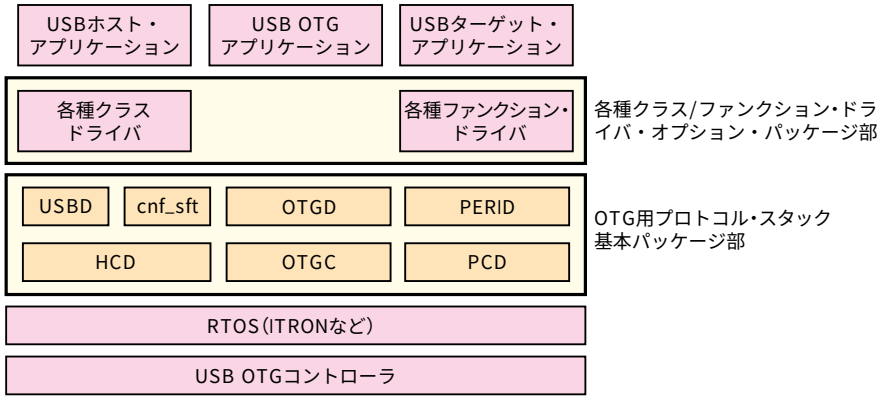

図8　On-The-Goのソフトウェア構成例

COLUMN

USBプロトコル・スタック

組み込み機器向けのUSBホスト・コントローラ・ドライバや各種クラス・ドライバは，USBプロトコル・スタックとして各社から販売されています．

実は**図6**～**図14**のソフトウェア構成例は，（株）グレープシステムより発売されているGR-USBシリーズのものです．他社のUSBプロトコル・スタックでも，基本的な考え方は同様と思われます．

（株）グレープシステム
http://www.grape.co.jp/

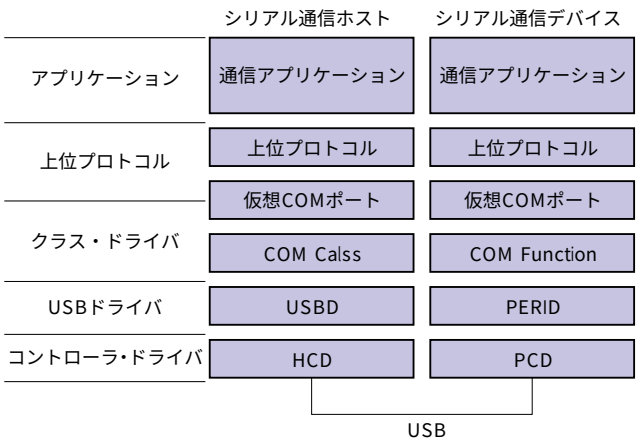

図 10　シリアル通信をエミュレートするシステムの構成例

一般にホスト，デバイス側ともコミュニケーション・クラスの上位に仮想 COM ポートのような API が提供されます．そのため，上位の通信プロトコルやアプリケーションは，基本的にシリアル通信と同等プログラミング・モデルを使用でき，それほど USB を意識することなく実装が可能です．

● プリンタ（Printer クラス）

USB 接続を使用したプリンタ・ホスト/デバイスの構成例を**図 11** に示します．

USB ではプリンタ・クラスが定義されていますが，このクラスは各プリンタ固有のコマンドに関して一切規定していないので，別途上位プロトコルとして各プリンタ固有のコマンド処理を実装する必要があります．

● ファイル・システム（MassStorage クラス）

ファイル・システムで USB のマスストレージ・デバイスを使用するホストおよびマスストレージ・デバイスのソフトウェア構成例を**図 12** に示します．

ホスト機器側では，ファイル・システムの下にマスストレージ・クラス・ドライバが必要です．マスストレージ・クラスには転送プロトコル種別として BOT，CBI，CB など，その上位に ATAPI，SCSI，UFI，SFF-8070i など各種コマンド体系に対応したサブクラスがあり，ホスト側では接続するデバイスがサポートするプロトコル，コマンド体系にあわせたサブクラスを実装する必要があります．

なお，USB のマスストレージ・クラスは各種コマンドを受け渡すための機能を提供しますが，そのコマンドをどのように使用するかについて規定しているものではありません．さらに，ファイル・システムと USB スタックは別々のプロダクトとして提供されることが多く，それらは直接インターフェースされていません．そうしたことから，ファイル・システムの下位ドライバとして USB のマスストレージ・クラスの機能を使用して，適切なコマンドを発行するドライバ・モジュールを用意する必要があります．

マスストレージ・デバイス側は，USB のマスストレージ・クラスのコマンドを，そのままメディア・ドライバのコマンドに変換して受け渡す程度の比較的簡単な処理の実装で実現可能です．

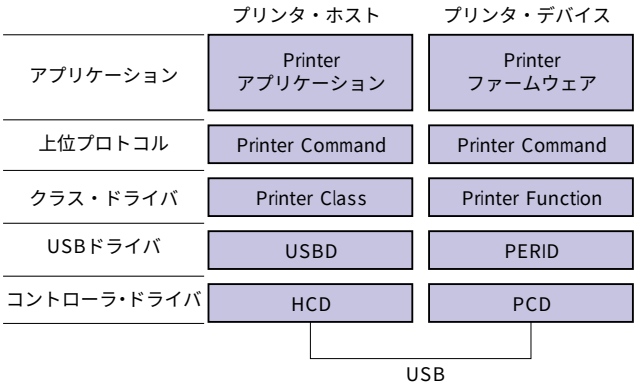

図 11　プリンタ・ホスト/デバイスの構成例

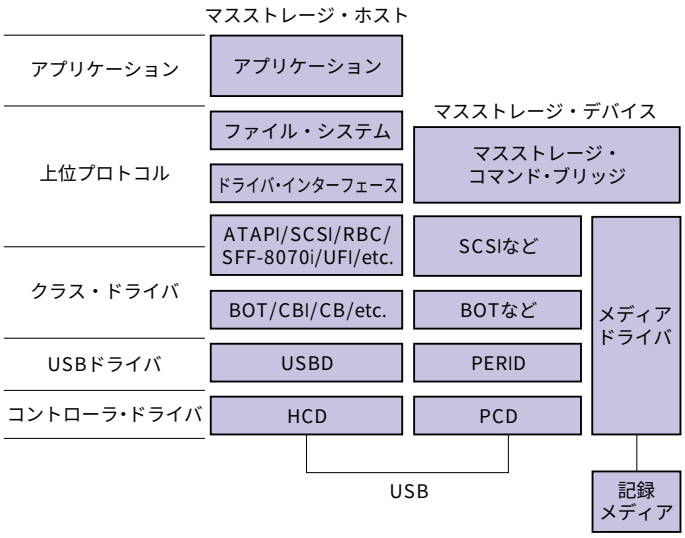

図 12　USB を利用したファイル・システムの構成例

● ネットワーク（ベンダ固有クラス）

USB ネットワーク・アダプタを利用したシステム構成例を**図 13** に示します．

ホスト側では TCP/IP などのネットワーク・プロトコルの下に，各ネットワーク・アダプタ用のクラス・ドライバが必要です．現在，Ethernet や Wireless LAN などの各種ネットワーク・アダプタが市場に出回っていますが，ほとんどの機種はベンダ固有のクラスが使用されおり，それぞれ固有のクラス・ドライバが必要となります．また，TCP/IP などのネットワーク・プロトコルによって下位のドライバ・インターフェースが異なるため，それぞれにあわせたドライバ・インターフェース・モジュールが必要になります．

TCP/IP などのネットワーク・プロトコルより上位のプロトコルやアプリケーションは特に USB を意識することはありません．

● PictBridge（SICD クラス）

PictBridge のシステム構成例を**図 14** に示します．

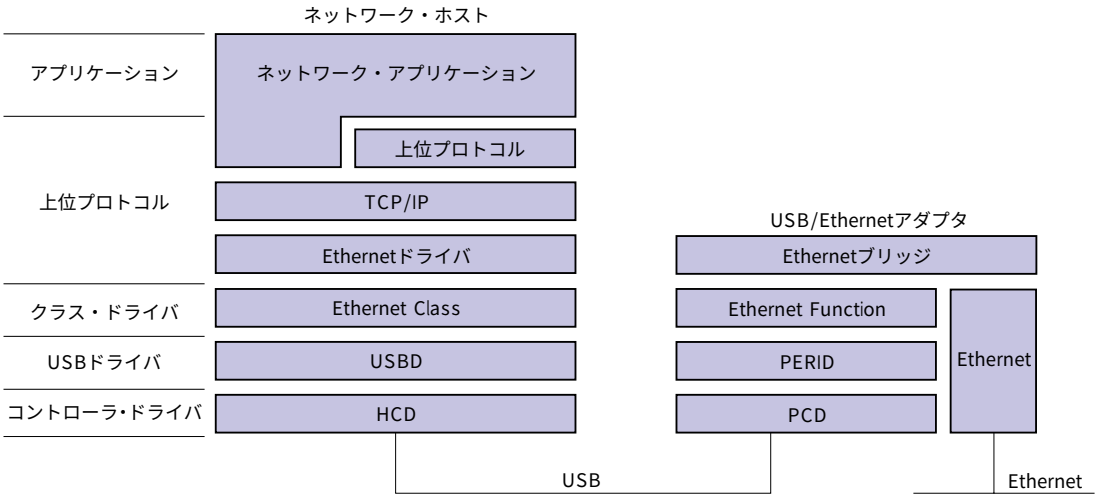

図13　USBネットワーク・アダプタを使用するシステム構成例

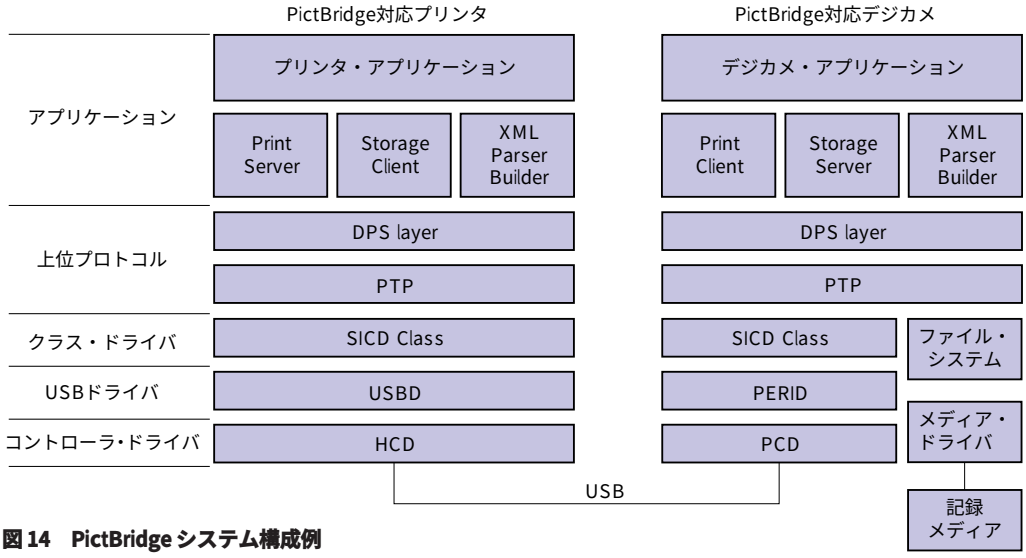

図14　PictBridgeシステム構成例

PictBridgeでは，デジカメなどの画像入力デバイス側がUSBデバイス，プリンタなどの画像出力デバイス側がUSBホストとなります．画像データや印刷制御コマンドをPTPプロトコルで受け渡すことによりPCレスの画像印刷処理を実現しています．

PictBridgeに関する説明はAppendix2で解説します．

7 組み込み機器におけるUSBソフトウェアの動作概要

組み込み機器におけるUSBソフトウェアの処理フローの全体を**図15**に示します．

● 初期化

(1) ドライバの初期化

各USBドライバ・モジュールの管理情報を初期化し，必要に応じて内部タスク，同期オブジェクトなどの生成および初期化を行います．

(2) コントローラの初期化

USBコントローラおよび関連ハードウェアの各種レジスタを初期化します．

(3) 割り込みの登録

USBコントローラおよび関連ハードウェアからの割り込み処理を登録します．

(4) コールバック・ルーチンの登録

上位アプリケーションに対する各種イベント通知用のコールバック・ルーチンなどを登録します．

(5) コントローラの起動

各種初期化が終了した後，USBコントローラおよび関連ハードウェアを動作可能な状態とします．

● 接続

(1) デバイス接続の認識

USBコントローラからの接続割り込み発生，あるいはハブからの接続通知により，ホスト側でデバイスの接続が認識され

ます．

(2) バス・エニュメレーション処理

接続されたデバイスに対するバス・エニュメレーション処理が起動され，各種ディスクリプタの取得，デバイス・アドレスの設定のほか，必要に応じてコンフィグレーションの設定などが行われます．

(3) パイプのオープン処理

取得したデバイス情報から該当するクラス・ドライバの接続処理が呼び出され，そのクラスのデータ転送に必要なパイプのオープン処理などが行われます．

(4) 上位アプリケーションへの接続通知

データ転送の準備が整った後，上位アプリケーションに対して接続通知のコールバックが行われます．上位アプリケーションでは接続されたデバイスと通信を開始する準備を行います．

● データ転送

デバイスとの転送用にオープンしたパイプを使用して各種データ転送(標準/ベンダ固有デバイス・リクエスト，バルク，インタラプト，アイソクロナス転送)を行います．

通常はクラス・ドライバが提供する API を使用してデータの送受信を行うため，直接パイプに対する転送要求を発行する必要はありません．

転送要求の完了は，コールバックなどで通知される非同期型と，転送要求 API が完了まで待ち状態となる同期型がありますが，組み込み向けの USB スタックではコードのサイズや必要資源を少なく抑えるため非同期型としているものが多いようです．

データ転送の要求方法は，USB ソフトウェアの構成によって異なります．基本的に次のような三つのケースがあります．

(1) USB ドライバ直接

クラス・ドライバが用意されていない場合や，独自にデバイス・リクエストを行うような場合には，USB ドライバの API を直接コールします．

USB ドライバの一般的な API としては，各パイプに対する通常の転送要求(バルク/インタラプト/アイソクロナス)や，各種デバイス・リクエスト転送要求(コントロール)などが用意されており，その API 仕様は各ドライバにより異なります．

(2) クラス・ドライバ経由

クラス・ドライバまで提供されており，その上位アプリケーションを開発するような場合は，クラス・ドライバの API を使用して USB データ転送を行います．

クラス・ドライバの API は一般的にそのクラスの提供する機能ごとに提供されます．たとえば，マスストレージ・クラスの場合，INQUIRY，READ(10)，WRITE(10)，SEEK，TEST UNIT READY などの各コマンドに対応した API が提供されます．

なお，クラス・ドライバの API が，どの程度 USB のデータ転送を意識するのかについては，それぞれのクラス・ドライバの実装により異なります．

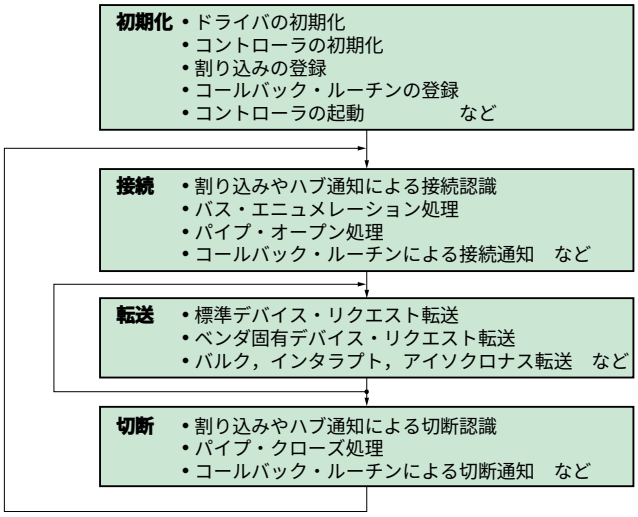

図 15　USB ソフトウェアの全体処理フロー

(3) 上位ミドルウェア経由

USB の上位にミドルウェアが搭載されている構成の場合には，そのミドルウェアの API をコールすることにより USB データ転送が実行されます．これらの API は基本的に USB のデータ転送を意識しません．

このような例としては，USB シリアル変換ケーブルなどの仮想 COM ポート，USB マスストレージ・デバイスをサポートしたファイル・システム，USB-LAN アダプタをサポートしたネットワーク・プロトコルなどがあります．

● 切断

(1) デバイス切断の認識

USB コントローラからの切断割り込み発生，あるいはハブからの切断通知により，ホスト側でデバイスの切断が認識されます．

(2) パイプのクローズ処理

切断されたデバイスとのデータ転送用パイプをクローズ状態にします．

(3) 上位アプリケーションへの切断通知

上位アプリケーションに対してコールバックなどで切断が通知されます．上位アプリケーションでは，現在処理中の転送要求の中断処理およびそれに伴う後処理などを行います．

まとめ

概略ですが，組み込み機器における一般的な USB システム構成と，USB のソフトウェアの動作概要を解説しました．

本稿が，これから USB インターフェースを搭載した組み込み機器を開発しようとする人にとって，USB ソフトウェア開発はどのようなものなのかをイメージするための一助となれば幸いです．

さとう・ようじ　(株)グレープシステム
くわの・まさひこ　パステルマジック

New Products ――デイテル，3 出力の絶縁型 DC-DC コンバータ「TPJ-5/300-15/50-D5」を発売
デイテル(株)は，5V/300mA と，± 15V/± 50mA の 3 出力絶縁型 DC-DC コンバータ「TPJ-5/300-15/50-D5」を発売した．入力電圧は 5V ± 5%，出力容量は 3W，外形寸法は 35mm × 27mm × 7mm．1 ～ 9 個購入時の単価は ¥2,860．