USB の基本アーキテクチャ

【物理的なアーキテクチャ】

USB での物理的な構成は下図のようになっていて、1台のパソコンが親機となって、ポーリング（問い合わせ）によって子機側と通信をします。

すべての通信の制御を親機が制御し、子機側から勝手に通信を開始することは出来ないようになっています。

このような構成図をネットワークとして見て、全体構成を**トポロジー**（Topology）と呼び、親機を**ホスト**（Host）、子機を**ノード**（Node）と呼んでいます。

実際の使用状態では、ノードには「**デバイス**」と「**ハブデバイス**」（ハブと呼ぶ）とがあります。

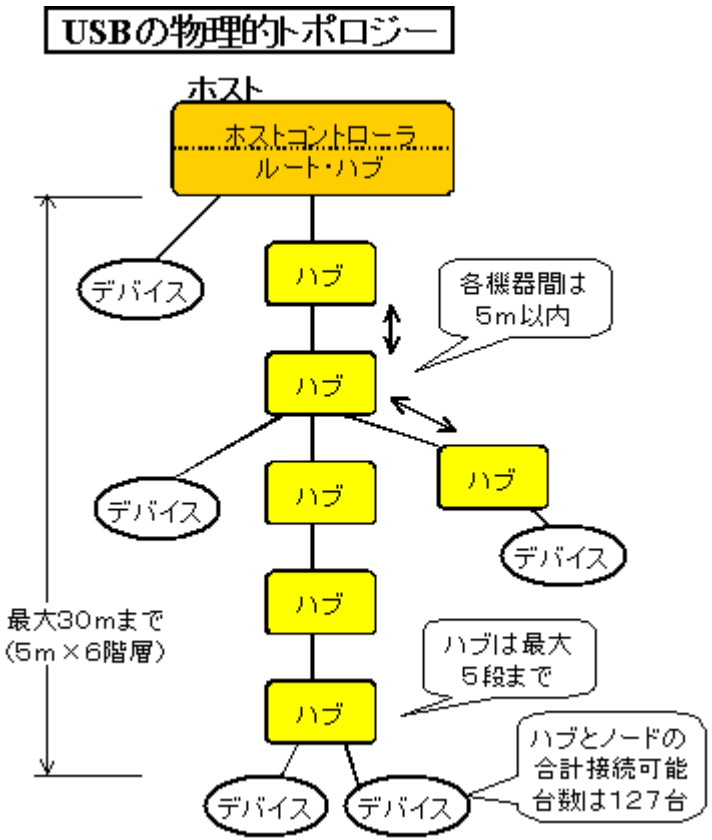
下図のように、ホストと呼ばれる中身は、処理装置となるホストコントローラと、最初の USB バスの接続部となるルートハブが一体化されたものとなっています。

そして実際のモノとしては、これがパソコン本体になります。

このルートハブに直接ノードとなるデバイスを接続するこも出来ますし、さらにデバイスの台数が多いときには、ハブを挿入して台数を増やすことが出来ます。

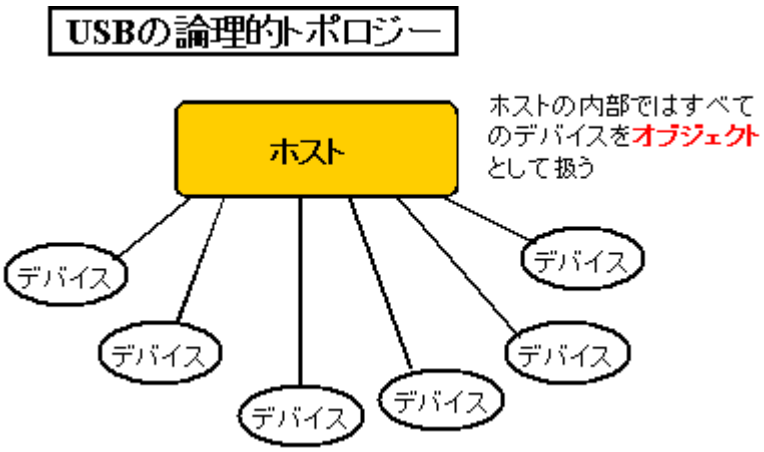
市販されているハブは、4台から8台程度のデバイスを接続することが可能になっています。しかし、ハブの従属接続は5段までと制限されています。

ホストの内部ではハブとデバイスすべてにアドレスが付与されて管理されます。このアドレスが最大127個なので、USB に接続可能なデバイスは最大127台ということになります。（ルートハブはアドレスが不要なので含まれません）



【論理構成とデバイスクラス】

上記は USB の物理的な構成ですが、ホストの内部でソフトウェアから見た論理的な構成は下図のような単純な構成になっています。



ホストの中では、各デバイス（ノードとハブ両方を含む）を、**オブジェクト**として扱っています。従って、オブジェクトとなる各デバイスは、「**クラスの概念**」を持っており、それぞれに「**デバイスクラス**」を持っています。 基本のデバイスクラスはひとつで、いろいろな種類のデバイスはその基本のデバイスクラスの派生クラスとなっています。

このオブジェクト指向の概念を使っているため、見かけ上、ホストからデバイスへのアクセスはオブジェクトへのアクセスという形となります。つまり、「**メッセージ**」のやり取りでデータの送受が実行されます。
具体的には、get･･･( ) と set･･･( ) コマンドだけでデータの送受を行うことになります。

そしてそのメッセージの処理を実行する「**メソッド**」の部分がデバイスの中のファームウェアの処理で実現されることになるわけです。
従って、デバイス側はメッセージに対応したメソッド部分だけを実装すれば、全体の制御はホストが実行してくれるので、デバイスは簡単な構成で処理を実現することが可能となります。

【通信のアーキテクチャ】

USB での通信の全体構成は下図で表すことが出来ます。

(1) 物理レベルの通信

まず、物理的な通信は Wire つまり実際のケーブル接続で行われます。この時に実際に物理レベルでの通信を実行するのは、USB コントローラの IC の役割となっていますので、使う側からはまったく物理的な通信については知らなくとも USB を使うことが可能になるようになっています。

この１本のケーブル上で通信されるので、瞬間的には１個のデータの送受信しか出来ない訳ですが、タイムシェアをして見かけ上、いくつかの通信線でつながっているような使い方をします。

(2) システムレベルの通信

次のレベルではホストのシステムソフトウェア（Windows のデバイスドライバレベル）での通信で、これは必須の通信となりますので、ホストとデバイス間には「**デフォルトパイプ**」という通信の論理接続を用意します。

そしてこのデフォルトパイプを使って行う通信を、「**コントロール転送**」と呼んでいます。

このコントロール転送により、リセットで初期化されたときに、システム設定を初期化したり、USB ケーブルが接続された時に、各種の設定制御をするために通信ができるようにします。

つまり、このコントロール転送を使って、「**コンフィギュレーション**」を行い、デバイスの使い方の設定情報をホストとデバイス間でやりとりします。

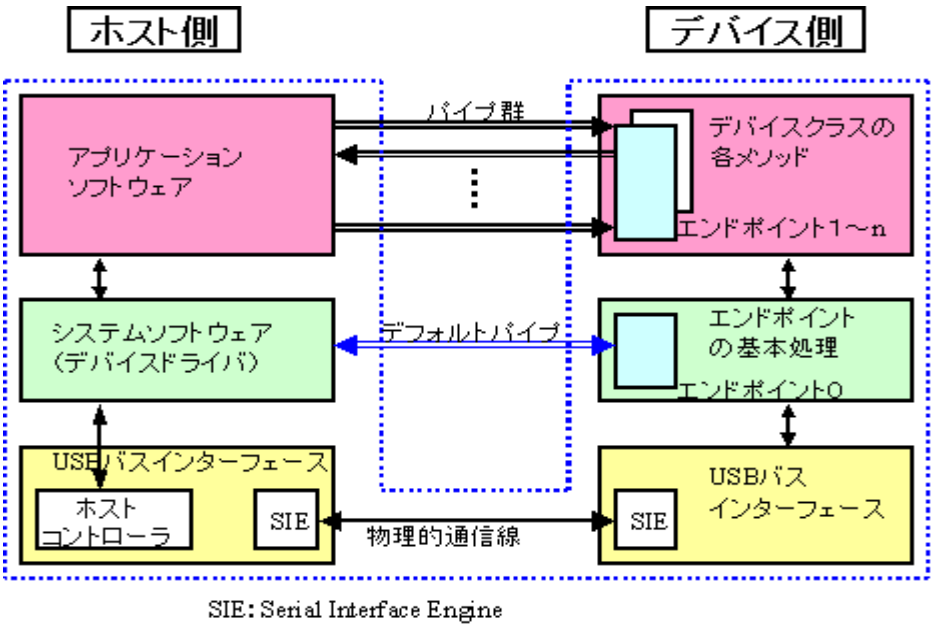
これで使用するパイプの本数や、転送モードなどの設定情報をホストがデバイスに要求し、それを元にしてホストがデバイスの使用条件を設定します。

(3) アプリケーションレベルの通信

次のレベルはアプリケーションのレベルで、この間の論理的な通信線を「**パイプ**」と呼び、必要に応じて複数のパイプを設定することが出来ます。

このパイプはあくまでも論理的な通信線ですから、実際の通信は１本の USB の線上で時分割で行われることになります。

このパイプを使った通信は、ホストから転送する「OUT」と、デバイス側から転送する「IN」のどちらか片方向だけの通信となっています。
これに対し、コントロール転送を行うデフォルトパイプは双方向通信となっています。



SIE: Serial Interface Engine

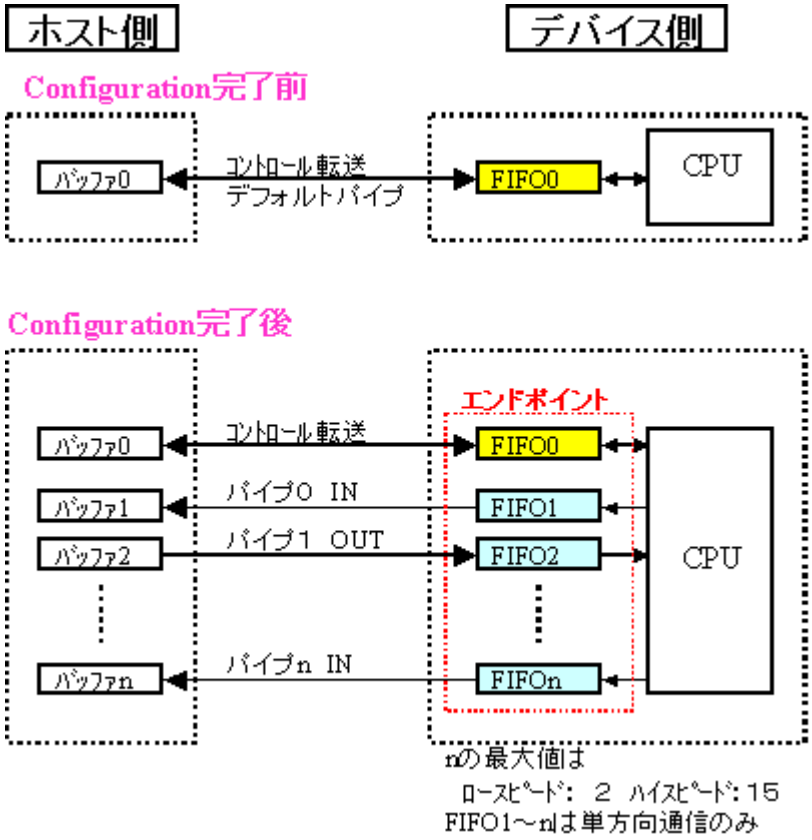
【エンドポイントとパイプ】

USB での論理的な通信はエンドポイントとパイプの概念で表現されます。
これを図で表すと下図のようになっています。
つまり、デバイス側には「エンドポイント」と呼ばれる「FIFO バッファ」が通信のための実体となり、ホスト側にも同様のバッファが用意されて、このバッファ同士が「パイプ」で接続されて、データを送受することになります。

まず、デバイスを USB に接続した直後の状態は、コンフィギュレーション前の状態となっていて、デフォルトパイプであるコントロール転送だけが通信のできる状態となっています。これに対応するのが「エンドポイント0（FIFO0）」となります

このコントロール転送を使って、コンフィギュレーションを実行し、デバイスの使い方が設定されたあとは、下側のように、コントロール転送以外に、あらたなエンドポイントが追加され、パイプが構成されます。このパイプは、片方向の通信しか出来ないようになっています。このエンドポイントの番号はホストが指定することが出来ます。
IN/OUT の方向はホストを中心に考え、ホストへの入力を IN、ホストからの出力を OUT

と定義しています。
またパイプの番号とエンドポイントの番号とは独立で、ホストが決めることが出来ます。



【USB の転送モード】

上記のパイプを使った USB の通信の仕方には、「バルク転送」、「インタラプト転送」
「アイソクロナス転送」があります。これにデフォルトパイプの「コントロール転送」を加えて
全部で4つの転送方法があり、それぞれの特徴は下記のようになっています。
（USB　Ver1.1 仕様）

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	少ないデータ量の半二重通信	大容量データの一括高信頼転送	小容量データの定周期転送	一定時間内のデータ量が保証された転送
用　途	セットアップ、設定パラメータ転送用	記憶装置。スキャナなどの大容量高速データの転送	計測やマンマシン機器のデータ転送	音声などのリアルタイムな転送

転送速度	1.5Mbps/12Mbps	12Mbs	1.5Mbps/12Mbps	12Mbps
転送周期	不定	不定	Nmsec(N=1～255)	1ms/フレーム
データ量/ﾊﾟｹｯﾄ	1～64 バイト(フル) 1～8 バイト(ロー)	8/16/32/64 バイト	1～64 バイト(フル) 1～8 バイト(ロー)	1～1023 バイト
信頼性	ー(再送あり)	◎(再送あり)	◎(再送あり)	△(再送なし)
転送速度	ー	◎	○	◎
遅延時間	ー	△	◎	◎

この転送モードのどれを使うかは、デバイス側がコンフィギュレーション用のデータとして持っていて、USB を接続した時にコントロール転送を使ってホストにその情報を渡し、ホストで確認したあと、ホスト側から設定されてからモードが確定します。
この時ホスト側から、デバイスのアドレス付けと、パイプ番号とエンドポイント番号の割り付けが行われます。