T-Engine Forum Specification

TEF040-S212-01.00.00/ja
2004.1.14

標準 MIDI デバイスドライバ仕様書

Number: TEF040-S212-01.00.00/ja
Title: 標準 MIDI デバイスドライバ仕様書
Status: [ ] Working Draft, [ ] Final Draft for Voting, [*] Standard
Date: 2003/9/29 First Edited （作成者 ヤマハ株式会社 石川 克己）
2003/12/25 v0.92 属性データに送受信タイムアウト設定・取得を追加。
2004/1/14 Voted.

目次

1　はじめに

この文書は T-Engine プラットホームで MIDI ストリームを扱うためのデバイスドライバインターフェイスを規定するものである。

2　本ドライバの位置付け

T-Engine 本体は MIDI コネクタ(DIN-5PIN)を持たないため、MIDI 機器を接続するためには何らかのプロトコル変換デバイスが必要である。MIDI デバイスドライバは、これら変換デバイスを制御して論理的な MIDI ストリームを送受信するためのドライバである。個々の MIDI 機器はアプリケーションが送受信する MIDI メッセージによって制御されるものであって、MIDI デバイスドライバはその動作に関与しない。

MIDI インターフェイスデバイスは、物理層として USB を利用するものとシリアル(RS-232C/422 等)を利用するものが一般的であるが、本書では物理的な伝送路については特に制限しない。また、MIDI インターフェイス機能を内蔵した音源モジュール等、複合的な機能を持つデバイスが存在するが、本書では MIDI ストリームの入出力のみに注目し、デバイス内部のモジュール構成については言及しない。

MIDI 信号は 31.25kbps のビットレートを持つシリアル伝送路を通じて送受信される。互換性の維持とチャネル数※の確保のため、MIDI の伝送能力の広帯域化はストリームごとのビットレートの直接的な向上ではなく、複数のストリームの多重化によって実現されている。多重化された際、MIDI ケーブル一本分に相当する伝送路(または接続点)を「ポート」と呼び、多重化に対応した MIDI インターフェイスをマルチポート(MIDI)インターフェイスと呼ぶことがある。明示的に区別する必要がある場合は、旧来のインターフェイスをシングルポート(MIDI)インターフェイスと呼ぶ。

※ MIDI プロトコルではストリームごとに 16 の論理チャネルが使用できる。演奏パートを指定するために
メッセージパケットに含める情報で、一般の多重化プロトコルで用られるチャネルとは概念が異なる

USB-MIDI の標準仕様は USB Audio Class の Sub-Class として定義されており、入出力それぞれ最大 16 ポート分の MIDI ストリームを扱うことができる。シリアル MIDI においても事実上の業界標準となる多重化プロトコルが存在している。

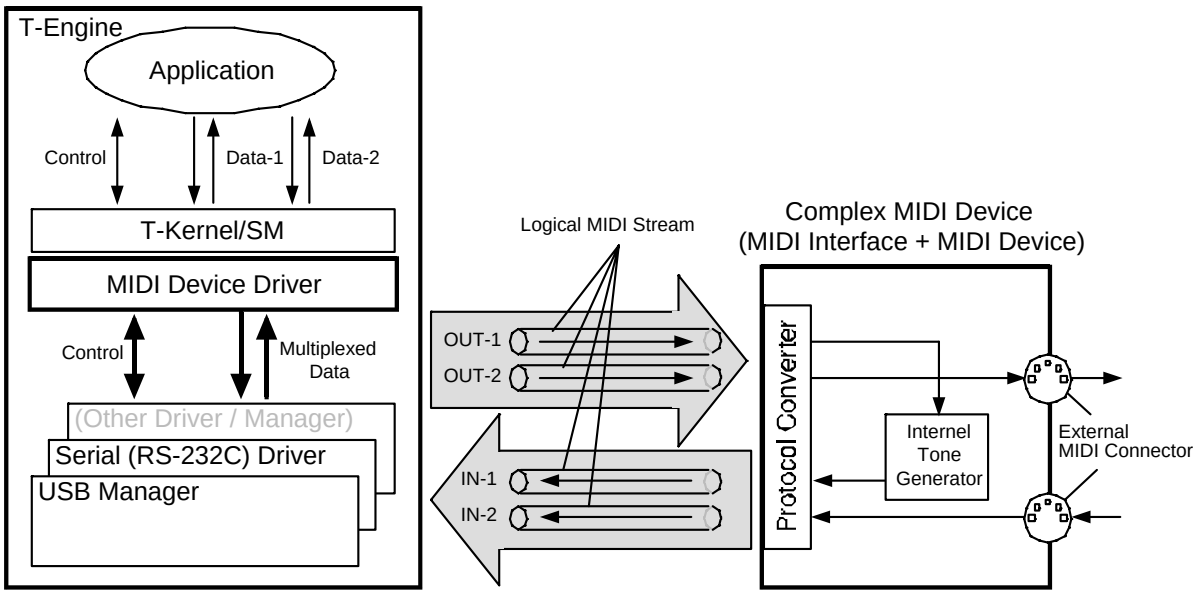

図 1:機能ブロック構成例 (2 ポートの複合デバイス)

T-Engine プラットホームで MIDI ストリームを扱う際の機能ブロックの構成例を図 1 に示す。この例では、T-Engine に接続されたデバイスが、MIDI IN/OUT それぞれ 2 ポートを有するマルチポートインターフェイスとして機能することを想定しており、ポート 1 が内蔵音源、ポート 2 は外部の MIDI コネクタに接続されている(MIDI では一部の例外を除いて 1-Origin の呼称を用いる)。このデバイスは MIDI インターフェイス機能と音源機能を併せ持った複合デバイスと見做すことができる。アプリケーションは T-Kernel/SM のデバイス管理機構を利用して MIDI ストリームを送受信し、MIDI 機器の制御を行う。

接続された MIDI インターフェイスがマルチポート機能を持つ場合、アプリケーションはポートの存在を意識し、MIDI デバイスドライバに対して入出力の対象となるポートを明示しなければならない。また、MIDI では IN/OUT の各コネクタが物理的に独立しているため、入出力のストリームを互いに独立して扱えることが重要である。例えば、MIDI INコネクタに接続されたMIDI キーボードデバイスの演奏情報を MIDI OUT コネクタに出力して、音源機能を持つ別のデバイスを発音させることができる。原則として IN/OUT のポート間に関連は無く、オープン制御のプロトコルによって通信が行われる。一部、IN/OUT の双方向通信によって単一デバイスとのハンドシェイクを要求するケースが存在するが、それらは MIDI 規格によるものではなく、上位プロトコルとして独自に定められたものである。

MIDI デバイスドライバは、制御対象となる MIDI インターフェイスデバイスとの間でプロトコル変換を行い、アプリケーションに対して MIDI ストリームの入出力機能及びユーティリティ機能を提供する。デバイスがマルチポートタイプであれば、制御可能なポート数に応じて論理ポートを設けて多重化処理を行う。また、利用する物理層の特性に応じたデバイス管理機構を持たなければならない。

3 実装に関して

ここでは MIDI デバイスドライバのモジュール構成例及び実装時の注意事項について解説する。

3.1 モジュール構成例

図 2 にデバイスドライバのモジュール構成例を示す。ここでは MIDI インターフェイスデバイスが出力 n ポート、入力 m ポートを持つマルチポートタイプであることを想定している。シングルポートタイプのインターフェイスをドライブする場合は n=1, m=1 となり、図中の多重化モジュール(Multiplexer)は一対一のプロトコル変換のみを行う。

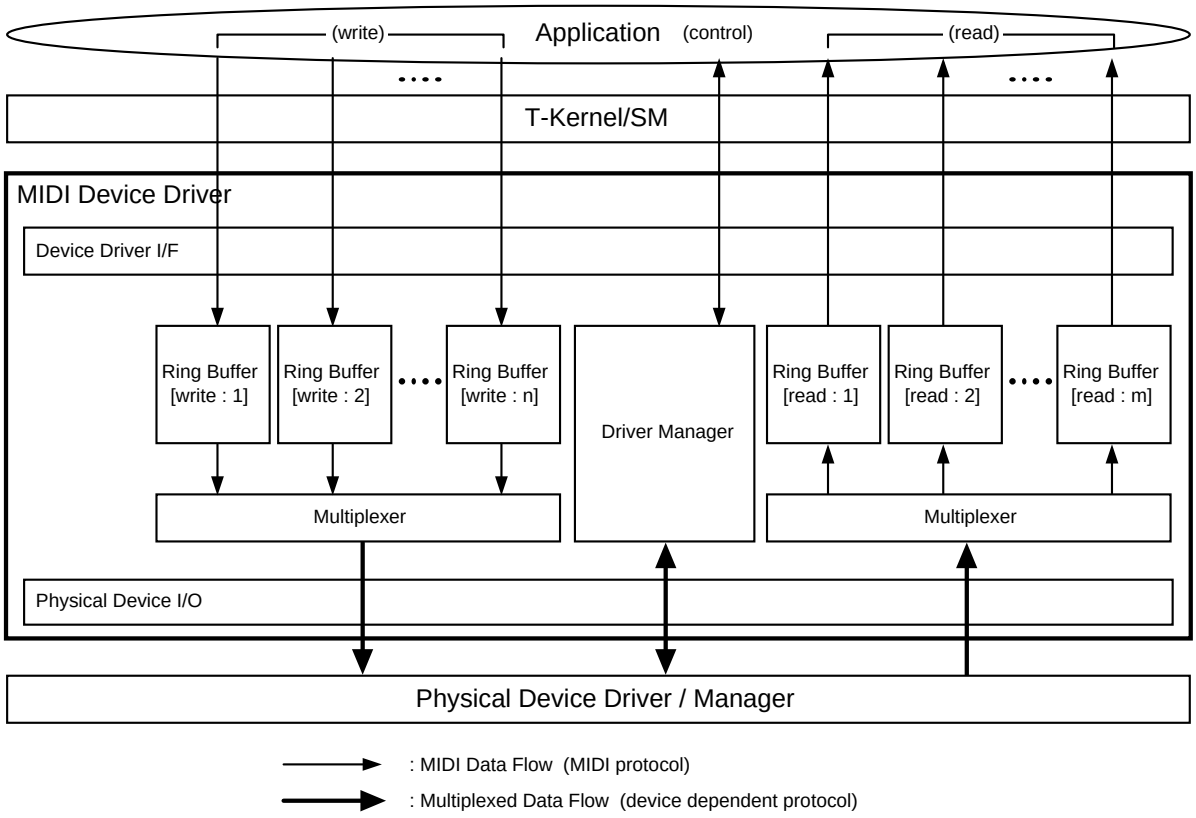

図 2： デバイスドライバモジュール構成

アプリケーション － デバイスドライバ間は MIDI プロトコルに準じたデータ入出力を行う(実装を容易にするため、一部の仕様を制限している)。デバイスドライバ － 物理ドライバ/マネージャ間の入出力には、デバイスに応じた多重化プロトコルを用いる。多重化プロトコルには USB-MIDI 標準クラス仕様の他、ベンダーやデバイスに依存した各方式が存在するが、本仕様ではこの部分の実装方式を制限しない。

デバイスドライバは主として、1)ドライバインターフェイス、2)ドライバ管理モジュール、3)メモリ管理モジュール及びデータバッファ、4)多重化処理を含むプロトコル変換モジュール、5)物理ドライバインターフェイス、の各モジュールから構成される。入出力のタイミング調整や処理コンテキストの分離を容易にするために、入出力の各ポートごとにリングバッファ(256～512byte 程度)を設ける実装方法が一般的である。ドライバ管理モジュールは、本仕様が定めたユーティリティ API に関連する処理の他、ドライバが利用する物理層の特性に応じた制御を行う。また、T-Kernel/SM が要求するデバイス管理機能をサポートしなければならない。

MIDI デバイスドライバはアプリケーションに対して MIDI ストリーム単位の入出力機能を提供する。MIDI プロトコルは、チャネル

の概念によってストリームあたり 16 パートの演奏情報を送受信可能であるが、チャネルの管理及び演奏パート間の情報のマージはアプリケーションの責任で処理する必要がある。

3.2 実装上の注意事項

3.2.1 デバイス名とサブユニットの割り当て

デバイス名は種別を “midi” とし、この後ろにユニットを示すアルファベット “a”～ と、サブユニット “0”～ を続ける。複数のドライバを同時に組み込む場合はユニット名によって互いを区別する。MIDI デバイスドライバではサブユニットを必須とし、デバイス登録時はサブユニット数の属性に 1 以上を指定する。ドライバのオープン時に物理デバイス名(“midi” + “a”～)を指定することはできない。

MIDI デバイスドライバでは、MIDI インターフェイスデバイスのポート番号をサブユニットに対応させる。サブユニットが 0-Origin であるのに対して、ポート番号は 1-Origin であることに注意が必要である。MIDI デバイスドライバでは論理デバイスを read 専用 / write 専用に分けて取り扱う。サブユニット 0～15 は read 専用デバイス、16～31 は write 専用デバイスである。ひとつのサブユニットを多重オープンすることはできない。

これらのルールに従い、1-IN/4-OUT の MIDI インターフェイスデバイスをサブユニットに割り当てた例を図 3 に示す。

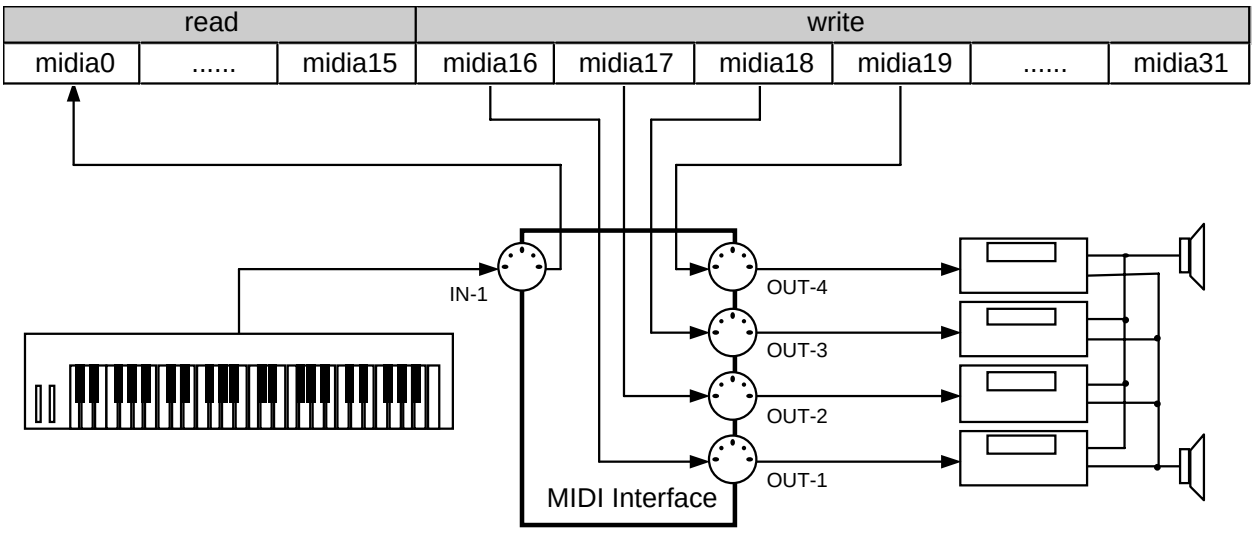

図 3:サブユニットと MIDI ポートの関係 (1-IN/4-OUT のデバイスの場合)

デバイスドライバは一つのデバイスのみを管理・制御する。複数デバイスの接続をサポートする場合は、ユニット名を変更して必要とされる数だけデバイスドライバを登録しなければならない。各デバイスの制御プロトコルが同一であれば、ドライバの実処理部分のコードは共通化が可能である。

それぞれの物理デバイスで使用可能な論理デバイス(MIDI ポート)の数は属性データの read によって取得可能である。また、デバイス名やポート名等の固有情報の参照もサポートされる。USB に代表されるホットプラグ対応の伝送路を利用する場合は、デバイスとポートの関係が動的に変化する場合があるので、アプリケーションはポートの固有情報を利用して論理デバイスを特定しなければならない。

3.2.2 MIDI 転送のリアルタイム性と非同期転送

MIDI プロトコルは即時発行・即時処理を原則としている。各メッセージはタイムスタンプを内包しておらず、送信されたタイミングがそのままイベントの発火期待時刻となる。主要なメッセージは 3 バイト以下で表現されており、リアルタイム性を考慮したプロトコルになっている。一般に MIDI 送信処理は同期転送を前提とした完了型 API で提供され、API ユーザは関数から戻った時点で送信処理が完了したものと見做して処理を継続する。受信処理においても、同期転送、すなわち、要求したサイズのデータを受信するまでドライバ内で待つ形の実装形式を取ることが多い。また、MIDI プロトコルは過去の履歴によって挙動を変えるメッセージを多々含むため、ひとつのストリーム処理系において、送信側・受信側のいずれにおいてもイベントの処理順序を保証しなければならない。

T-Kernel/SM のデバイス管理機構は同期/非同期転送の拡張サービスコールを提供しているが、MIDI デバイスドライバではこれら MIDI プロトコルの特性に配慮した実装が要求されるため、同期/非同期に関わらず処理要求を可能な限り迅速に受け付けなければならない。また、少量・高頻度のアクセス要求によるオーバーヘッドを回避するため、ドライバ内部のリングバッファが ready、即ち、要求サイズ分の read/write が可能な状態であれば、そのまま呼び出しコンテキストで処理を完了するのが望ましい。待ち合わせが発生する場合は T-Kernel/SM の要求仕様に従って非同期アクセスの手順を踏む。

write 要求の場合、リングバッファにデータを書き込んだ後に多重化/プロトコル変換処理を経て実際の転送が行われるが、後段の処理が正常に動作している限りはバッファへの書き込み完了をもって転送完了と見做すことができる。また、デバイスドライバは MIDI 受信時にタイムスタンプを付加しないため、read 要求によって得られたデータのタイミングはアプリケーションがケアする必要がある。

3.2.3 処理要求のタイムアウト

MIDI デバイスドライバでは処理要求のタイムアウトを 2 つのステージに分けて取り扱う。

一方は T-Kernel/SM のデバイス管理機能によって規定されるもので、要求そのものの受け付けが遅延した場合に発生するタイムアウトである。詳細については T-Kernel 仕様書を参照のこと。

他方は MIDI デバイスドライバが独自に実現するもので、受け付けた一つの要求に関する入出力処理が指定時間内に完了しなかった場合にその要求をキャンセルする。タイムアウトの発生状況は `tk_wai_dev` の `ioer` の内容や `tk_srea_dev`, `tk_swri_dev` の戻り値によって確認することができる。処理がタイムアウトした場合であってもそれまでに送受信したデータは有効である。タイムアウト時間は属性データ `DN_MIDI_SETTMO` によって設定可能で、初期状態では `0`(タイムアウトなし)となっている。

MIDI プロトコルはオープンループ制御を原則とするためタイムアウトを規定していないが、上位アプリケーションが特定の MIDI 機器とのハンドシェイクを必要とする場合には後者のタイムアウトを利用することができる。

3.2.4 ポート間の処理の独立性

MIDI ではあらゆるポート間で独立して送受信を行えなければならない。マルチポートデバイスの異なるポート間はもちろんのこと、シングルポートデバイスであっても read/write は互いに干渉しないように実装される必要がある。

MIDI インターフェイスのポートを T-Kernel/SM のサブユニットに対応させる場合、デバイスドライバは異なるサブユニットへのアクセス要求を独立して受け付ける。仮に、あるサブユニットに関する処理が待ち状態に入っても、それによって他のサブユニットに対する要求がブロックされることはない。非同期アクセス要求のキューイングは、各サブユニットごとに独立してカウントされなければならない。

3.2.5 属性データの扱い

属性データへのアクセス要求は、固有データへのアクセス要求のキューイングとは無関係に実行されなければならない。ある

ポートに対して固有データの非同期アクセス要求がキューイングされていても、属性データの read/write 要求は即座に受け付けられること。

3.2.6 その他の留意点

MIDI デバイスドライバに関して、本仕様書が特に規定しない項目は T-Kernel の標準仕様に準ずるものとする。

4 API 仕様

4.1 ID tk_opn_dev(UB *devnm, UINT omode)

MIDI ポートを確保し、待機状態にする。同じサブユニットを多重オープンすることはできない。

引数

devnm

論理デバイス名の文字列へのポインタ。物理デバイス名は指定不可。

ex) MIDI IN : ”midi” + “a”～”z” + サブユニット(ポート番号 - 1)
MIDI OUT : ”midi” + “a”～”z” + サブユニット(ポート番号 + 15)

omode

オプション指定

TD_NOLOCK 固有データの送受信バッファをデバイスドライバでロックしない。

4.2 ER tk_cls_dev(ID dd, UINT option)

送受信処理を停止し、MIDI ポートを解放する。キューイングされている非同期アクセス要求はキャンセルされる。

4.3 ID tk_wai_dev(ID dd, ID reqid, INT *asize, ER *ioer, TMO tmout)

reqid で指定した非同期アクセス要求の完了を待ち合わせる。

引数

ioer

入出力エラーを返す。

(E_IO | E_MIDI_TMO) DN_MIDI_SETTMO の設定によって送受信処理がタイムアウトした

4.4 ID tk_wri_dev(ID dd, INT start, VP buf, INT size, TMO tmout)
ER tk_swri_dev(ID dd, INT start, VP buf, INT size, INT *asize)

4.4.1 MIDI データ送信

バッファに格納された MIDI データを指定サイズ分送信する。要求が完了または中断されるまでバッファを解放してはならない。

送信データに MIDI ランニングステータスを用いてはならない。必ず上位でステータスバイトを補完すること。要求受け付け時、デバイスドライバはバッファ内のデータの MIDI メッセージとしての正当性をチェックしない(エラーを返さない)。実際の送信時(プロトコル変換時)に不正なメッセージを検知した場合は、有効なメッセージが出現するまでバッファの内容を読み捨てる。ひとつのバッファ内に複数の MIDI メッセージを連続して記述し、まとめて送信することができる。バッファ終端で MIDI メッセージが完結して

いる必要はない。

引数

start

DN_MIDI_SNDDATA(= 0)

buf

送信データを格納したバッファの先頭アドレス

size

バッファサイズ(バイト単位)

0 を指定した場合は書き込みを行わず、現時点で書き込み可能なサイズを返す(T-Kernel/SM 仕様)

戻り値 (tk_swri_dev のみ)

(E_IO | E_MIDI_TMO)　　DN_MIDI_SETTMO の設定によって送信処理がタイムアウトした

その他のエラーについては T-Kernel 仕様書を参照のこと。

4.4.2 送受信タイムアウト設定

当該 MIDI ポートに対する送受信処理のタイムアウトをミリ秒単位で設定する。read/write 要求の送受信サイズに関わらず、時間内に 1 回の要求が完了しなかった場合にタイムアウトする。それまでに処理したデータは有効である。タイムアウトせずに送受信完了を待ち続ける場合は 0 を設定する(初期状態)。

引数

start

DN_MIDI_SETTMO

buf

UW 型変数のポインタ。タイムアウト時間を格納する。

size

必ず sizeof(UW)とする。

4.4.3 内部バッファクリア

ドライバ内部の送受信用リングバッファをクリアする。MIDI ポート単位で実行される。

引数

start

DN_MIDI_CLRBUF

buf

(参照しない)

size

(参照しない)

4.5 ID tk_rea_dev(ID dd, INT start, VP buf, INT size, TMO tmout) ER tk_srea_dev(ID dd, INT start, VP buf, INT size, INT *asize)

4.5.1 MIDI データ受信

バッファを与え、指定サイズ分の MIDI データを受信する。要求が完了または中断されるまでバッファを解放してはならない。

デバイスドライバはデバイスからデータを受信する際にランニングステータスを補完する。また、受信データが MIDI メッセージとして不正であった場合はドライバ内部の受信用リングバッファへの書き込みを行わず、有効なメッセージが出現するまで読み捨てる。処理完了後、要求時に指定したサイズによっては受信データの終端で MIDI メッセージが完結しない場合があるが、次回の受信要求で続きのデータを読み出すことができる。一回の受信要求で MIDI メッセージの途中まで読み出された場合であっても、残されたデータバイトをドライバが削除することはない。

送信元の MIDI デバイスのメッセージ送信量に対して本関数が十分な頻度で実行されず、ドライバ内部の受信用リングバッファが溢れたときはドライバによってバッファがクリアされる。その際、上位に対して通知を行わないので注意すること。

引数

start

DN_MIDI_RCVDATA(= 0)

buf

受信データを格納するバッファの先頭アドレス

size

受信するデータサイズ (バイト単位)

0 を指定した場合は読み込みを行わず、現時点で読み込み可能なサイズを返す(T-Kernel/SM 仕様)。

戻り値 (tk_srea_dev のみ)

(E_IO | E_MIDI_TMO)　　DN_MIDI_SETTMO の設定によって受信処理がタイムアウトした

その他のエラーについては T-Kernel 仕様書を参照のこと。

4.5.2 送受信タイムアウト取得

当該 MIDI ポートに対して現在設定されている送受信タイムアウト時間(ミリ秒単位)を取得する。タイムアウトが設定されていない(処理完了を待ち続ける)場合は 0 となる。

引数

start

DN_MIDI_GETTMO

buf

UW 型変数のポインタ。タイムアウト時間が格納される。

size

必ず sizeof(UW)とする。

4.5.3 デバイス情報取得

このデバイスディスクリプタが利用している送受信ポートを含むデバイスの情報を取得する。同一デバイス内のどのサブユニットのデバイスディスクリプタから参照しても同じ値となる。

引数

start

DN_MIDI_GETDEVINFO

buf

ポートマップ情報を格納する構造体の先頭ポインタ

```
struct {
    W    nOutPortNum;
    W    nInPortNum;
} MidiDriverDevInfo;
```

nOutPortNum

このデバイスの出力ポート数

nInPortNum

このデバイスの入力ポート数

size

必ず sizeof(MidiDriverDevInfo)を与える

4.5.4 ポート名取得 (オプション)

このデバイスディスクリプタが利用している送受信ポートのポート名を取得する。ポート名は USB String Descriptor 等から取得しても良いし、ドライバ内部で固定値として生成しても良い。

文字列はドライバによって '¥0' でターミネートされる。指定されたバイト数よりもポート名が長い場合は (size - 1) バイト分を書き込んでターミネートする。

引数

start

DN_MIDI_GETPORTNAME

buf

ポート名を格納するバッファの先頭アドレス

size

取得するサイズ (バイト単位)

0 を指定した場合は書き込みを行わず、このポート名のサイズを返す

4.5.5 デバイス名取得 (オプション)

このデバイスディスクリプタが利用している送受信ポートを含むデバイスのデバイス名を取得する。同一デバイス内のどのサブユニットのデバイスディスクリプタから参照しても同じ値となる。

デバイス名は USB String Descriptor 等から取得しても良いし、ドライバ内部で固定値として生成しても良い。USB デバイス等では動的に機器構成が変わる場合や同一デバイスが複数台接続される場合があるため、デバイス名にシリアルナンバー等を含めて個体の特定ができるようにするのが望ましい。

文字列はドライバによって '¥0' でターミネートされる。指定されたバイト数よりもポート名が長い場合は (size - 1) バイト分を書き込んでターミネートする。

引数

start

DN_MIDI_GETDEVNAME

buf

デバイス名を格納するバッファの先頭アドレス

size

取得するサイズ (バイト単位)

0 を指定した場合は書き込みを行わず、このデバイス名のサイズを返す

5 参考資料

・ MIDI 1.0 規格書 / MIDI Standard & Recommended Practices (Japanese Edition)
著作・発行 : 社団法人 音楽電子事業協会(AMEI)
発売: 株式会社リットーミュージック
ISBN4-8456-0348-9

・ MIDI バイブル I 基礎編、MIDI バイブル II 応用編
リットーミュージック出版編集部 編
ISBN4-8456-0267-9 / ISBN4-8456-0303-9

・ Universal Serial Bus Device Class Definition for MIDI Devices
http://www.usb.org/developers/devclass_docs#approved