aPCI-011-050322

aPCI シリーズ A/D・D/A ボード

aPCI-A57

ソフトウェアマニュアル

株式会社アドテックシステムサイエンス

― 目 次 ―

１．はじめに

本マニュアルは、8ch12 ビット A/D・2ch12 ビット D/A・8 ビットパラレルボード「aPCI-A57」を Windows2000/WindowsXP でご利用いただくためのドライバのインストール手順およびソフトウェアの使用方法などについて記述しています。
ハードウェアに関する詳細は、ハードウェアマニュアルを参照してください。

２．動作環境

API は、Windows(R) 2000 および Windows(R) XP 上で、最大 16 枚までの aPCI-A57 (以下「デバイス」) を制御します。

３．ファイル一覧

```
¥
├ Win2000_XP                Windows2000/XP 用 32bit ドライバ
│  ├ apci57.inf             インストールファイル
│  ├ apci57.sys             ドライバ
│  └ apci57.dll             ドライバ制御 DLL
│
├ Sample                    サンプルソース
│  ├ VC                     Visual C++サンプルソース Ver.4/5/6 対応
│  │  ├ apci57.h            DLL 定義ヘッダ
│  │  ├ apci57w.h           ラッパー関数ヘッダ
│  │  ├ apci57w.c           ラッパー関数
│  │  ├ apci57s.c           ラッパー関数使用例
│  │  ├ buildvc.txt         サンプルソース構築例
│  │  └ Lib
│  │     └ Win2000_XP
│  │        └ apci57.lib    インポートライブラリ
│  │
│  ├ VB                     Visual Basic サンプルソース Ver.4(32bit)/5/6 対応
│  │  ├ apci57.bas          DLL 定義
│  │  ├ apci57w.bas         ラッパー関数
│  │  ├ apci57s.bas         ラッパー関数使用例
│  │  └ buildvb.txt         サンプルソース構築例
│  │
│  └ Delphi                 Delphi サンプルソース Ver.2/3/4/5/6/7 対応
│     ├ apci57w.pas         DLL 定義／ラッパー関数
│     ├ apci57s.pas         ラッパー関数使用例
│     ├ apci57s.dfm         フォームファイル
│     └ builddp.txt         サンプルソース構築例
│
├ apci57.hlp/cnt            ヘルプ
├ apci57.exe                動作チェックアプリケーション
├ apci57adj.exe             調整アプリケーション
├ apci57ap.txt              API リファレンス
└ readme.txt                リードミー
```

４．ドライバ

４－１．インストール

本製品をご使用になる前に、ソフトウェアの組み込み等の準備が必要です。

ソフトウェアは、サポートソフト（添付サポートディスクまたは弊社ホームページ http://www.adtek.co.jp/ からダウンロード）に収められています。
ここでは、サポートソフトを、フロッピーディスク（以下「サポートディスク」）にコピーして使用する場合について示しています。CD-R 等他のメディアをご使用の場合は、適宜読み替えて作業を進めてください。

４－１－１．Windows2000 へのインストール

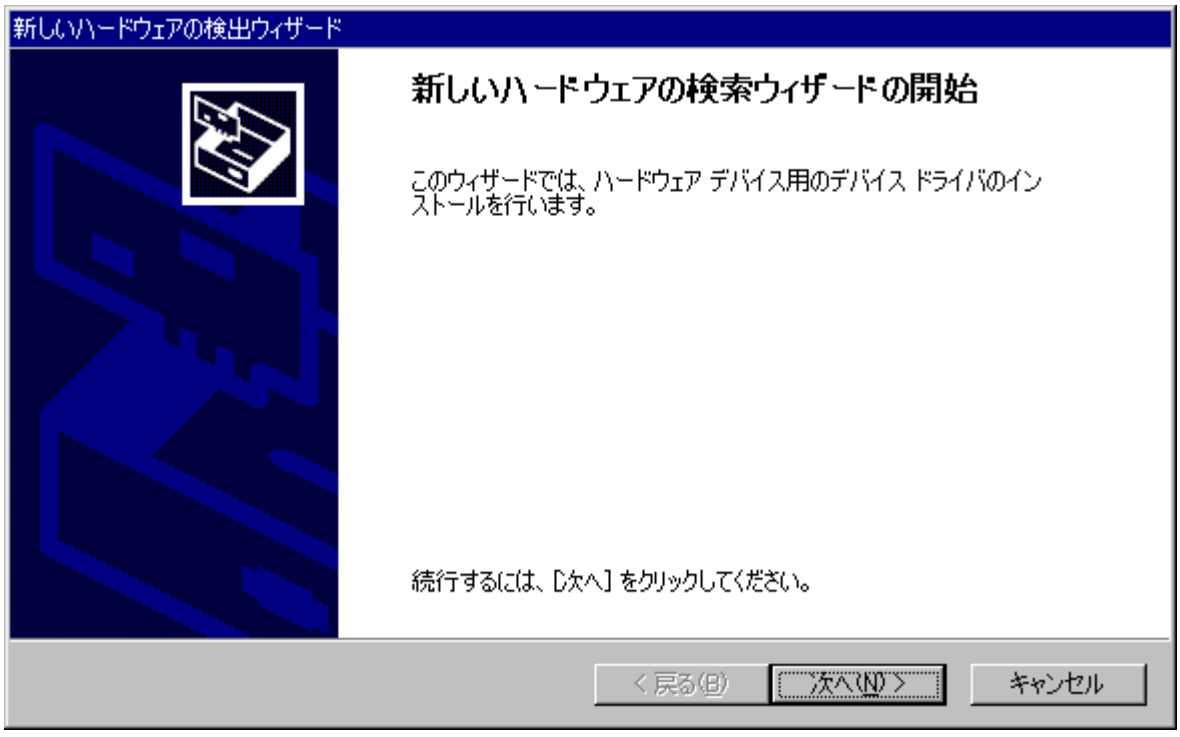
当ボードは、プラグアンドプレイに対応しておりますので、以下の手順に従って登録(インストール)を行ってください。

システムの電源が切れていることを確認し、ボードをスロットに挿入します。

システムの電源を入れ Windows2000 が起動したら、「Administrator」でログオンします。

[新しいハードウェアが見つかりました]とメッセージが出ます。その後"新しいハードウェアの検出ウィザード"が起動しますので、メッセージに従ってインストールを行います。

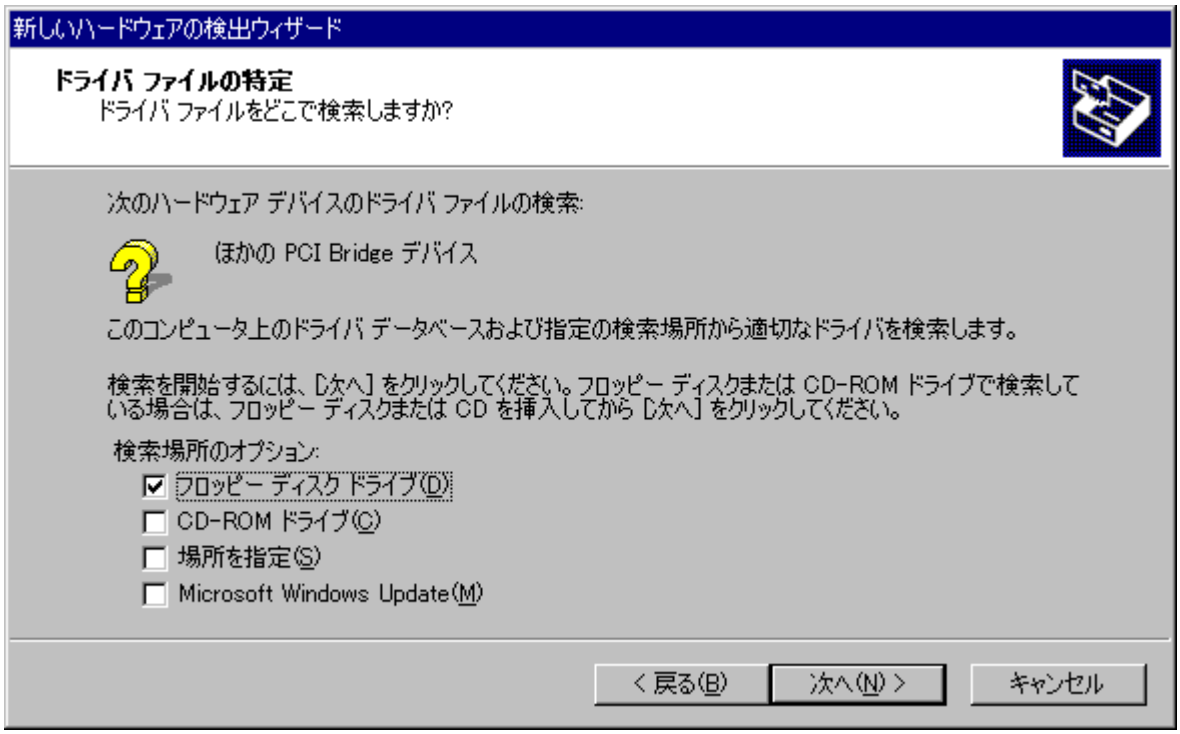
以下の画面が現れましたら、サポートディスクをドライブに挿入し、[次へ]ボタンをクリックします。



[デバイスに最適なドライバを検索する(推奨)]を選択し、[次へ]ボタンをクリックします。

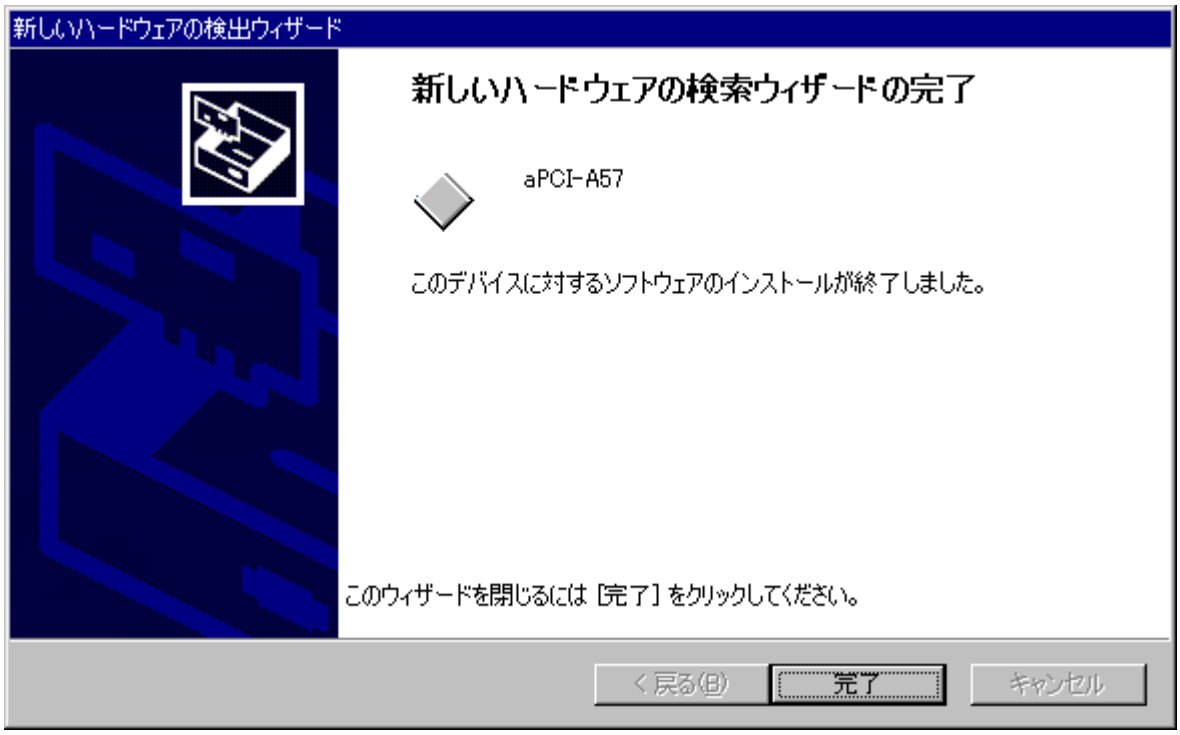
※ 画面中の「ほかの PCI Bridge デバイス」は「マルチメディア コントローラ」と表示される場合もあります。(以下の画面でも同様)

[フロッピーディスクドライブ]を選択し、[次へ]ボタンをクリックします。



[次へ]ボタンをクリックします。

[完了]ボタンをクリックしてインストールは終了です。



ドライバを制御する DLL ファイルは自動的にコピーされませんので、お使いの環境に合わせて、ファイルをコピーしてください。
一般的には「アプリケーションと同じディレクトリ」か、Windows2000 では
「%WinDir%¥System32 (例 C:¥Winnt¥System32)」となります。

４－１－２．WindowsXP へのインストール

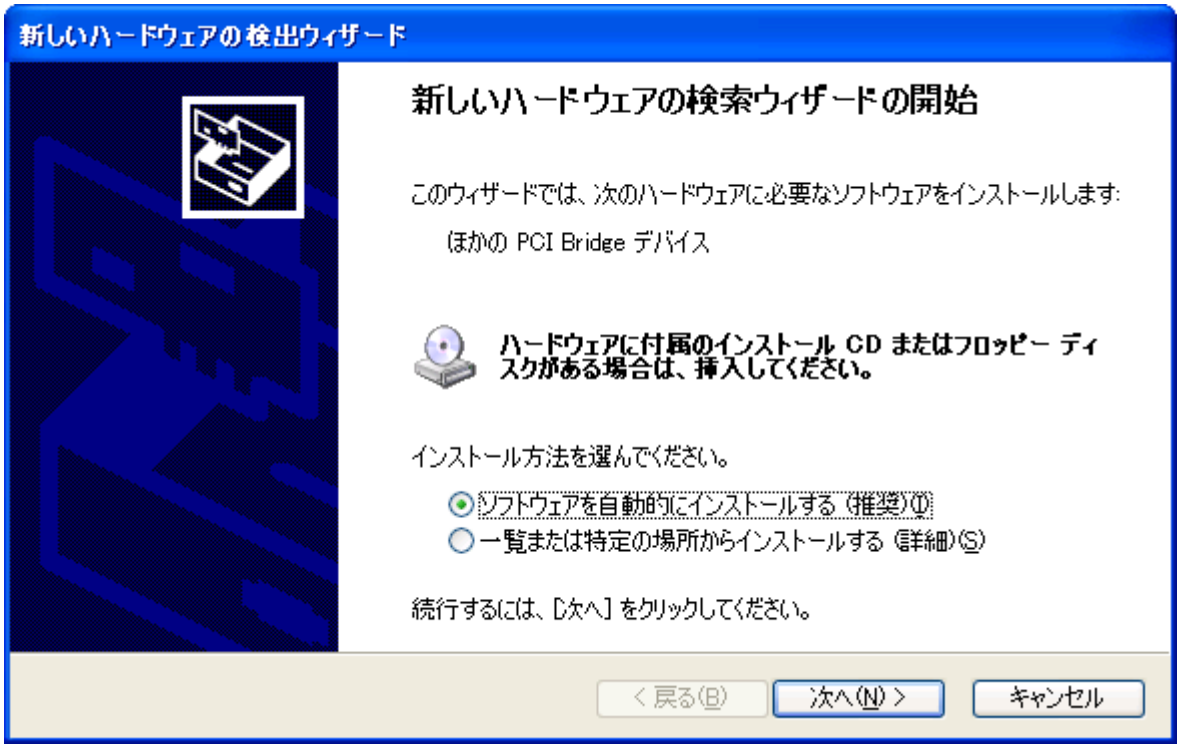
当ボードは、プラグアンドプレイに対応しておりますので、以下の手順に従って登録(インストール)を行ってください。

システムの電源が切れていることを確認し、ボードをスロットに挿入します。

システムの電源を入れ WindowsXP が起動したら、「Administrator」でログオンします。

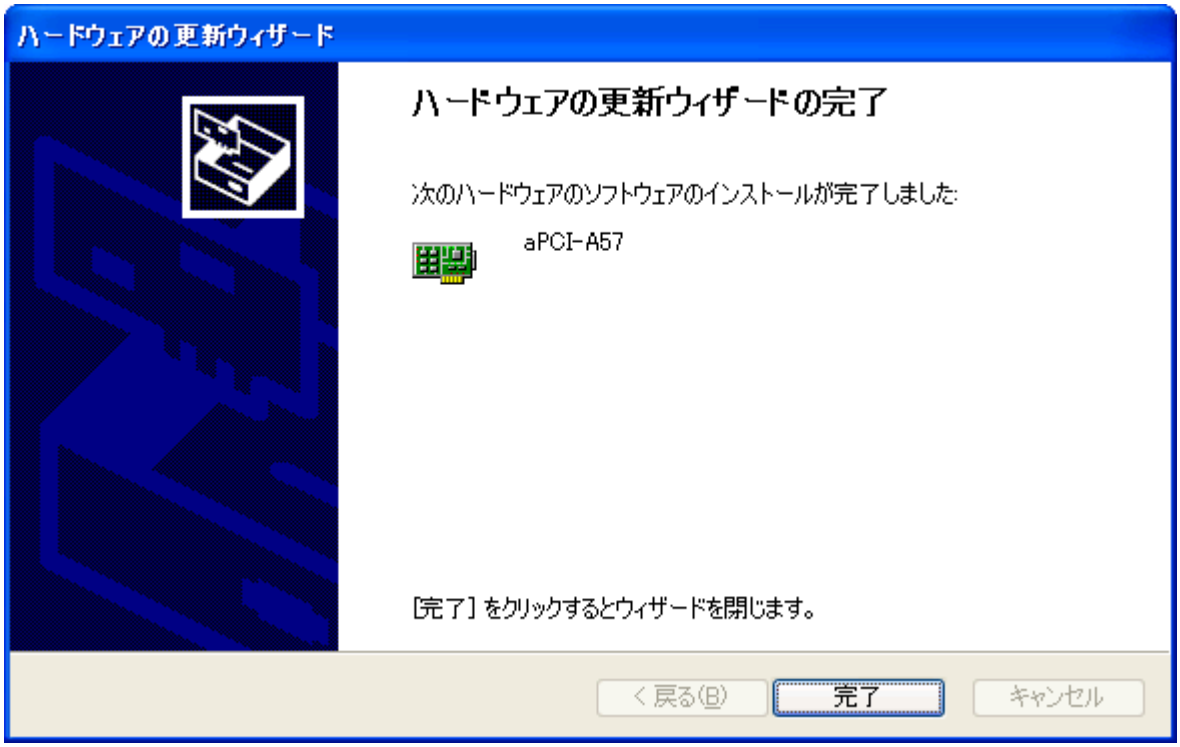
[新しいハードウェアが見つかりました]とメッセージが出ます。その後"新しいハードウェアの検出ウィザード"が起動しますので、メッセージに従ってインストールを行います。

以下の画面が現れましたら、サポートディスクをドライブに挿入し、[次へ]ボタンをクリックします。



※ 画面中の「ほかの PCI Bridge デバイス」は「マルチメディア コントローラ」と表示される場合もあります。

[完了]ボタンをクリックしてインストールは終了です。

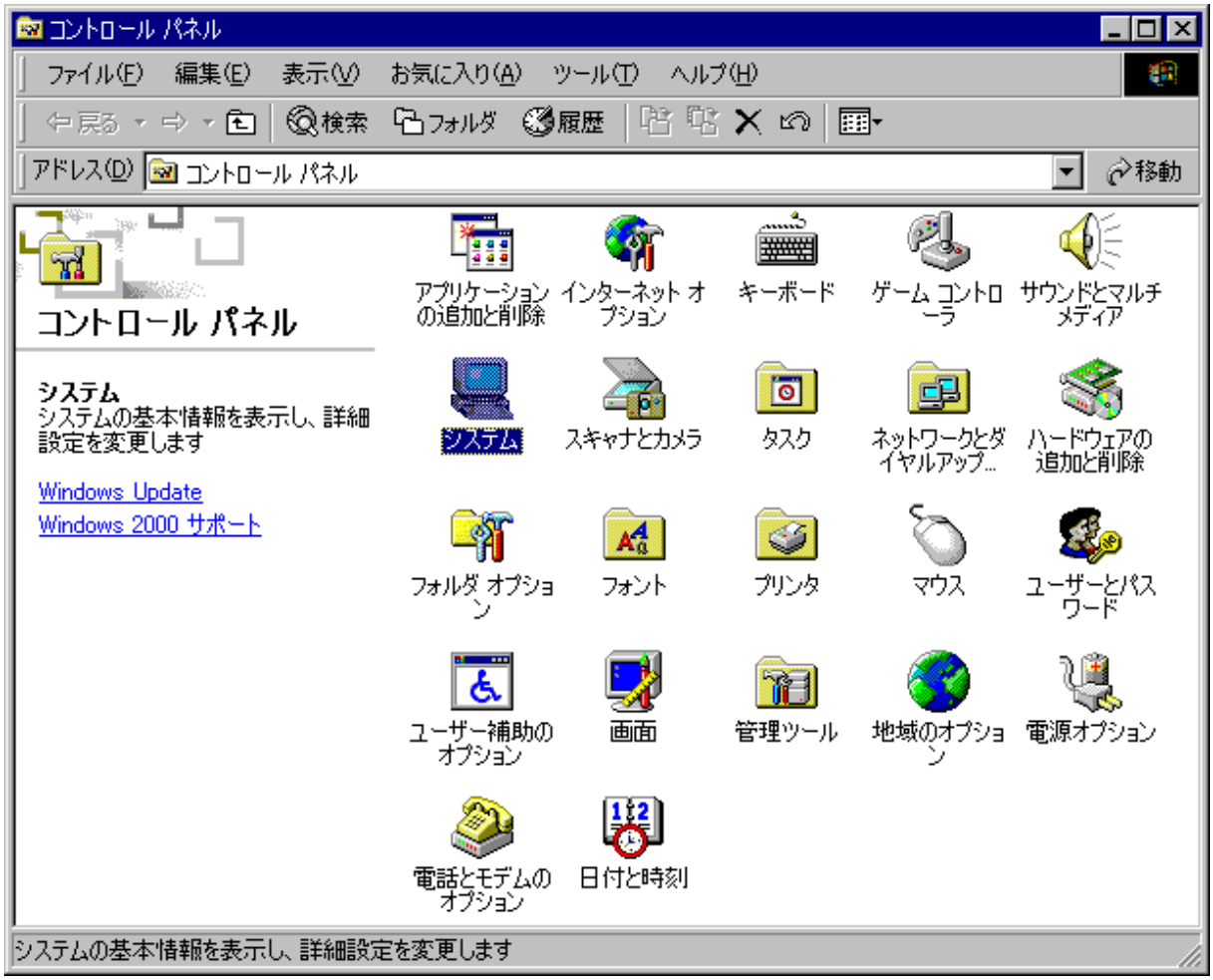


ドライバを制御する DLL ファイルは自動的にコピーされませんので、お使いの環境に合わせて、ファイルをコピーしてください。
一般的には「アプリケーションと同じディレクトリ」か、WindowsXP では
「%WinDir%¥System32 (例 C:¥Winnt¥System32)」となります。

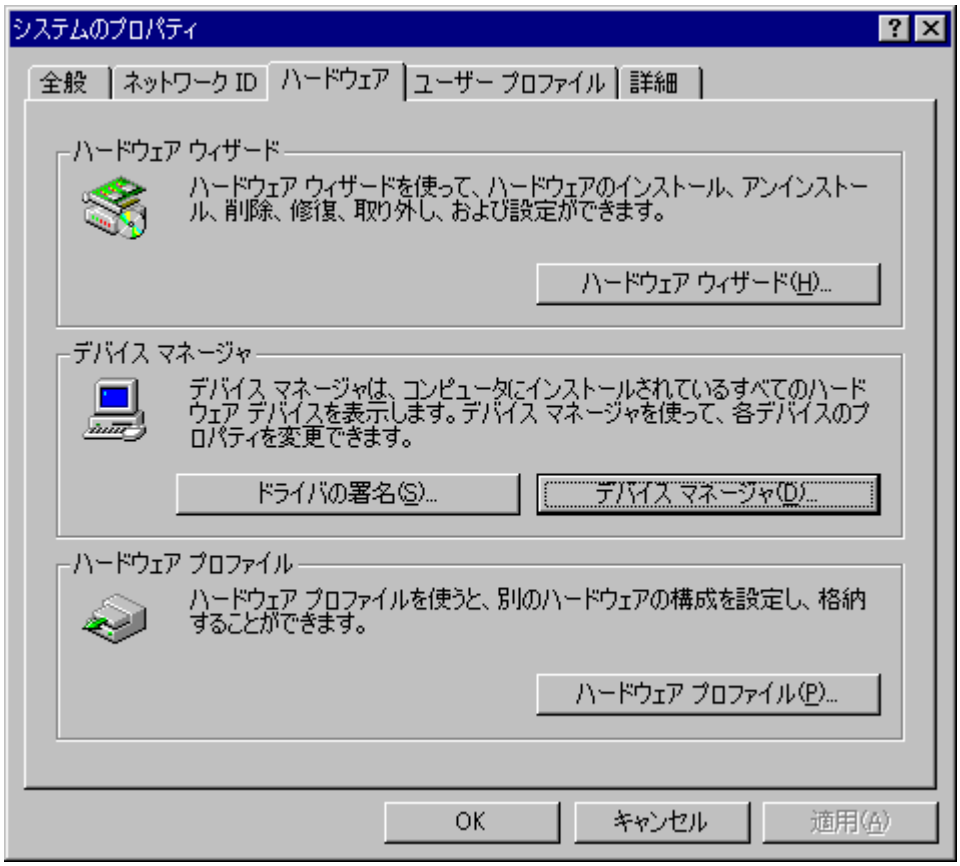
４－２．アンインストール

４－２－１．Windows2000 からのアンインストール

[スタートー設定ーコントロールパネル]を実行して、コントロールパネルを開きます。開いたら[システム]アイコンをダブルクリックします。

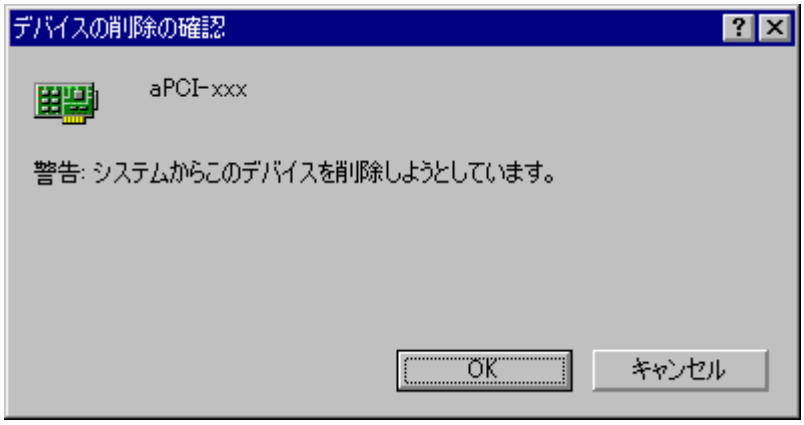


下図の欄で[デバイスマネージャ]ボタンをクリックします。



[aPCI-A57]上で右クリックして、[削除]を選択します。

[OK]ボタンをクリックします。

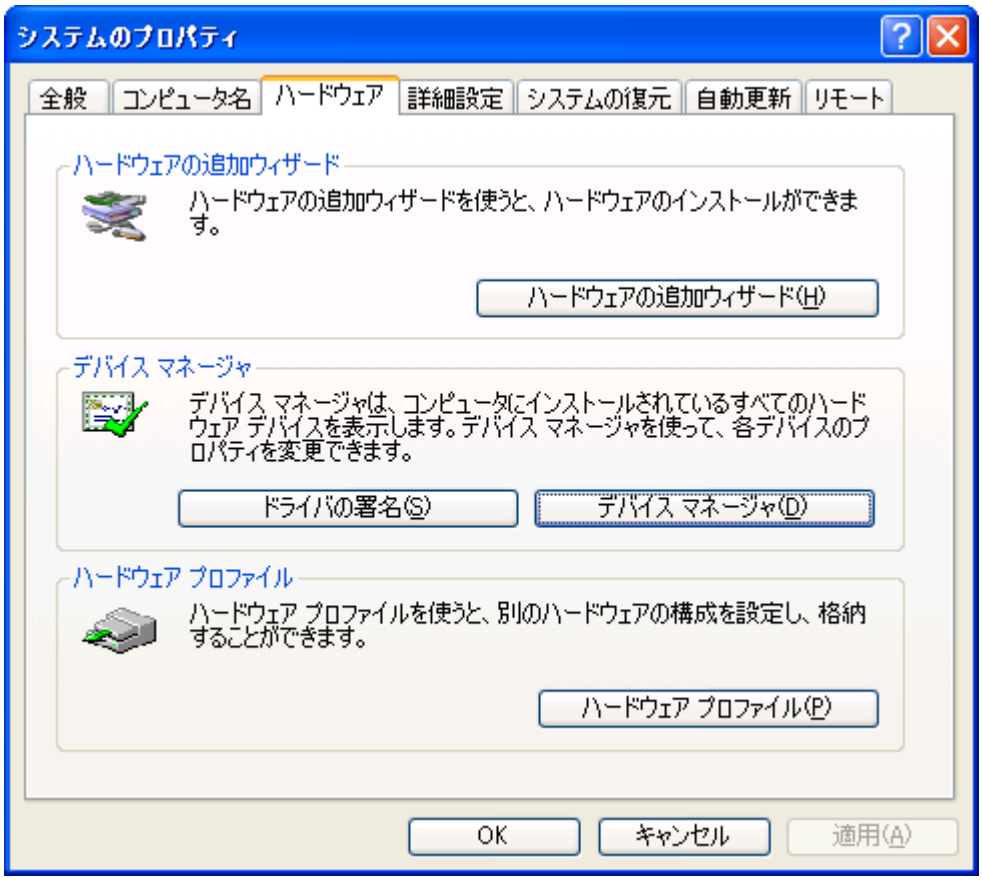


Windows2000 を終了してからパソコンの電源を切りボードを取り外します。

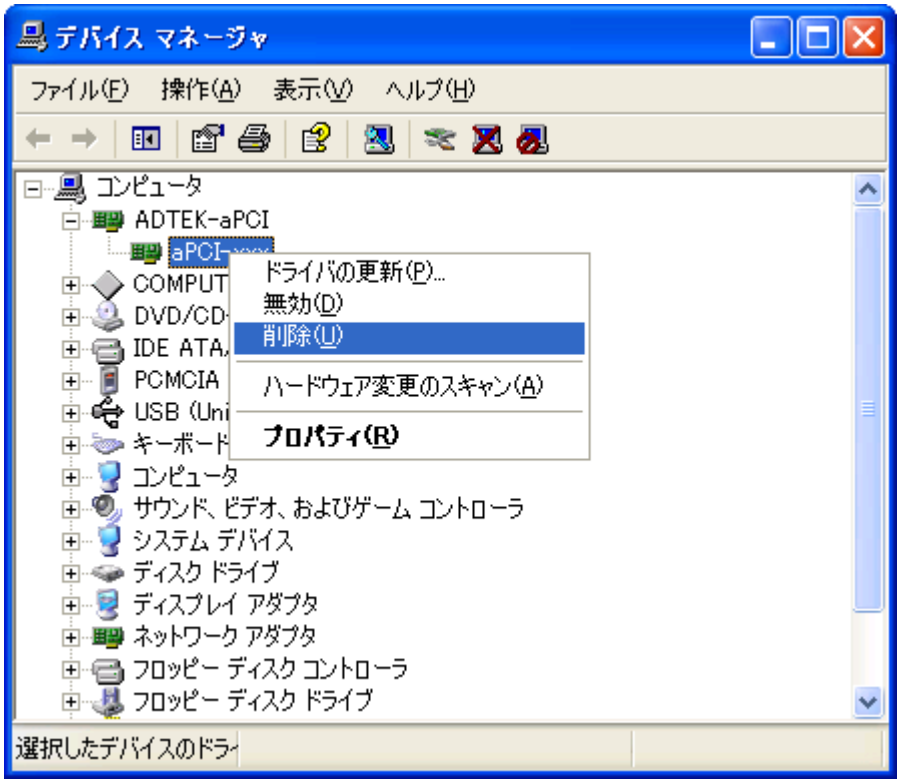
４－２－２．WindowsXP からのアンインストール

[スタートーコントロールパネル]を実行して、コントロールパネルを開きます。開いたら[システム]アイコンをダブルクリックします。

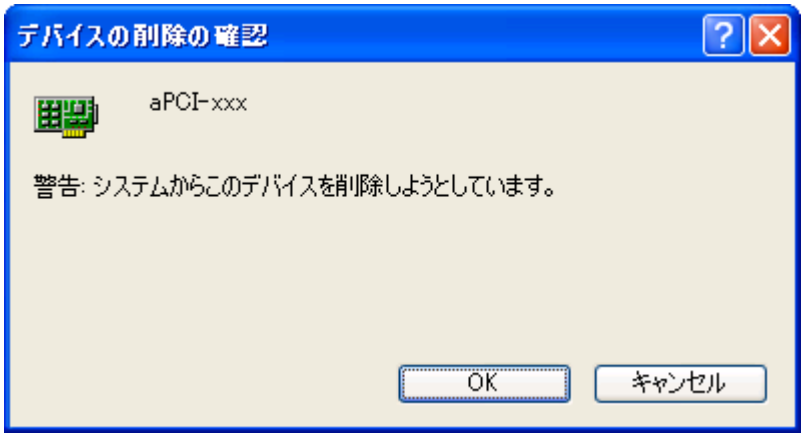
下図の欄で[デバイスマネージャ]ボタンをクリックします。



[aPCI-A57]上で右クリックして、[削除]を選択します。



[OK]ボタンをクリックします。



WindowsXP を終了してからパソコンの電源を切りボードを取り外します。

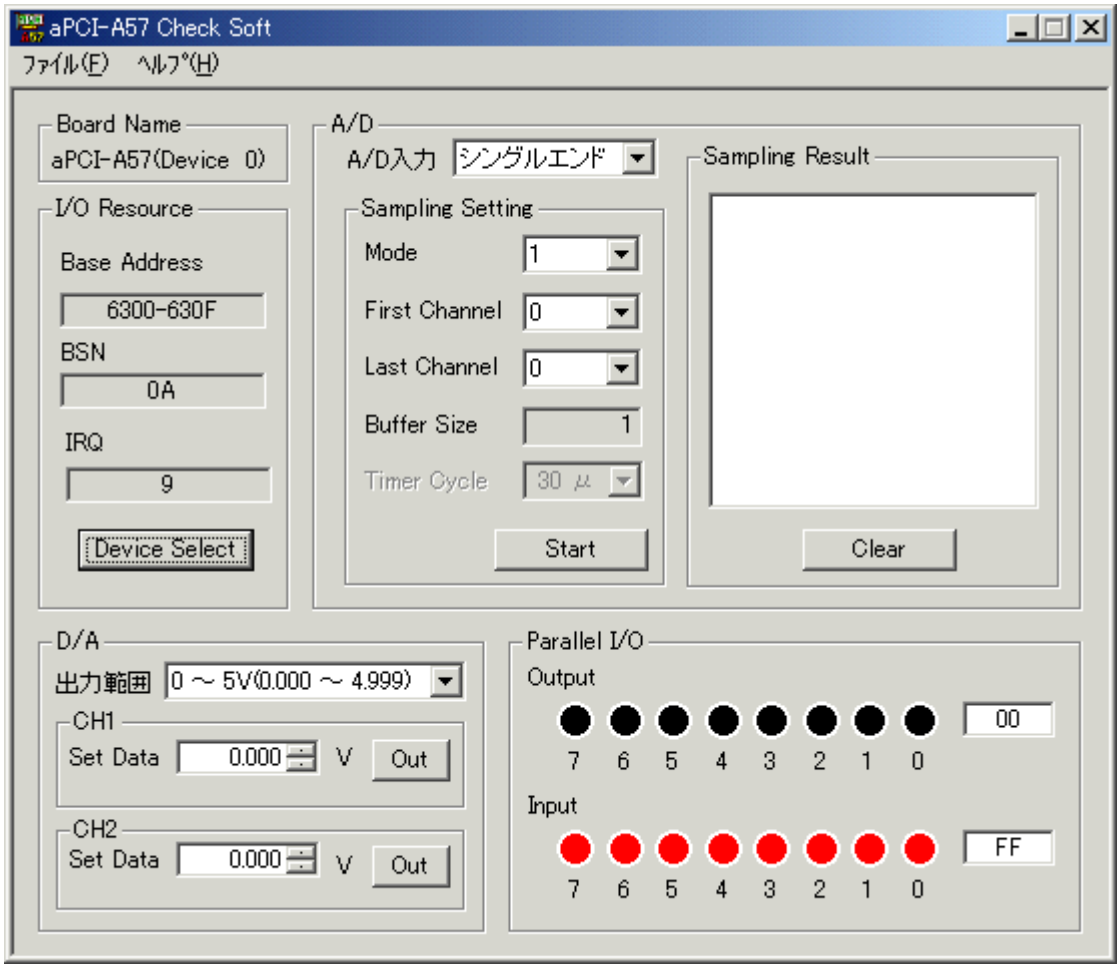
５．動作チェックソフト

５－１．概要

本アプリケーションソフト(apci57.exe)は、各機能の動作チェックを行うツールです。

５－２．起動画面

本アプリケーションを起動すると、下記の様な画面を表示します。



５－３.「ファイル」メニュー

［アプリケーションの終了］を選択することにより、本アプリケーションを終了します。

５－４.「ヘルプ」メニュー

［トピックの検索］を選択することにより、ヘルプファイル(apci57.hlp)を表示します。

［このプログラムについて］を選択することにより、ドライバおよび DLL のバージョン情報を表示します。

５－５. デバイス情報表示

[I/O Address]
このデバイスが使用している I/O アドレスが表示します。

[BSN]
BSN(Board Select Number)セレクトロータリースイッチで設定した値が表示します。

[IRQ]
このデバイスが使用している IRQ が表示します。

[Device Select]
デバイスを選択します。

５－６．A/D 入力

[A/D 入力]
A/D 入力方式「シングルエンド入力」「差動入力」を選択します。

[Mode]
サンプリングモードを設定します。

Mode	スタート方法	サンプリング方式	CLK ソース	バッファモード
1	ソフトスタート	シングル	内部	ドライバメモリ
2	ソフトスタート	バースト	内部（タイマ）	ドライバメモリ
3	外部トリガ	シングル	外部	ドライバメモリ
4	外部トリガ	バースト	外部（タイマ）	ドライバメモリ
5	ソフトスタート	バースト	内部（タイマ）	デバイスメモリ
6	外部トリガ	バースト	外部（タイマ）	デバイスメモリ

[First Channel]
サンプリング開始チャンネル。0～7 を選択します。
チャンネルインクリメントサンプリングを行う場合、First Channel と Last Channel を設定します。
First Channel と Last Channel に同じチャンネルを指定した場合は固定チャンネルでサンプリングします。

[Last Channel]
サンプリング終了チャンネル。0～7 を選択します。

[Buffer Size]
バッファサイズを設定します。

[Timer Cycle]
CLK ソースをタイマにした場合、タイマ値を設定します。
設定する値は、ハードウェアマニュアルを参照してください。

[Start]
上記の設定後、ボタンを押すとサンプリングを開始します。
スタート方法が外部トリガの場合は、ハード上の PI7 端子が 100nsec 以上の負論理パルスの立ち下りエッジを検出後、サンプリングを開始します。

[Sampling Result]
サンプリング結果を表示します。

５－７．D/A 出力

[出力範囲]
D/A 出力電圧レンジを選択します。

[Set Data]
出力電圧値を入力し[Out]ボタンを押して電圧を出力します。

５－８．ポート入出力

[Output]の場合、クリックすると H-L が反転し、出力データが 16 進数で表示します。

[Input]の場合、ビットの H-L の状態と入力データが 16 進数で表示します。

６．サンプルソース

サンプルソースには、Visual C++ (Ver.4/5/6) 版、Visual Basic (Ver.4(32bit)/5/6) 版、
Delphi (Ver.2/3/4/5/6/7) 版がございます。

DLL 内の関数のインターフェースは、各関数のヘッダ、注釈および「７．API 仕様」を参照してください。

＜ラッパー関数＞
apci57w.* のように「w」が付いているファイルには、DLL 内の関数を簡単にコールするためのラッパー (Wrapper) 関数が定義されています (DLL のロード/アンロード関数も含まれています)。
デバイスの制御を開始する前には DLL ロード関数を、デバイスの制御を終了する際には DLL アンロード関数をコールしてください。

サンプルソースは各開発環境にて実行してお試しいただけます。
詳しくは各ディレクトリ内の buildxx.txt の例をご覧ください。

※ サンプルソースのご利用については、各開発環境および OS・言語に対する充分な理解を前提としております。よって、これらそのものの使用方法に関するお問い合わせには一切お答えいたしかねますので、あらかじめご了承ください。

7．API 仕様

7－1．概要

本章では、aPCI-A57 用ユーザ公開 API を定義します。
デバイスドライバの詳細については触れていません。

7－2．プログラム構成

本章で定義される API は、apci57.sys (デバイスドライバ) および apci57.dll (ダイナミックリンクライブラリ) により実現されます。

７－３．API リファレンス

７－３－１．Apci57Create

■機　能　デバイスの使用を宣言

■形　式　Visual C++

```
BOOL   Apci57Create
(
  LPWORD lpwLogSlot,
  HWND hwnd,
  UINT uWMBase,
  LPVOID lpvInitArg
);
```

■形　式　Visual Basic

```
Function   Apci57Create
(
  lpwLogSlot As Integer,
  ByVal hwnd As Long,
  ByVal uWMBase As Long,
  LpvInitArg As Any
) As Long
```

■形　式　Delphi

```
function   Apci57Create
(
  var lpwLogSlot: WORD;
  hwnd: THandle;
  uWMBase: UINT;
  var lpvInitArg
 ): BOOL;
```

■入　力　lpwLogSlot

使用したいデバイスが挿入されている論理スロットを指定します。
論理スロット番号は 0 から始まります。
APCI57_SLOT_AUTO を指定した場合、使用可能な論理スロットを探します。
論理スロット 0 から検索し、見つかった時点でその論理スロット番号を返します。
すでにアプリケーションにより使用されているデバイスはスキップします。
以後、アプリケーションはこの値でデバイスを識別します。
NULL 不可。

hwnd

ドライバが送出するメッセージを受け取るウィンドウのハンドルを指定します。
メッセージを使用しない場合は NULL を指定します。

uWMBase
送出するメッセージ ID の値を指定します。
通常 WM_USER 以上の値を指定します。

lpvInitArg
使用しません (将来の為の予約です)。
Visual Basic の場合はダミーの変数を渡してください。

■出　力　*lpwLogSlot
使用可能なデバイスが見つかった場合は、その論理スロット番号を格納します。
見つからなかった場合、この値は未定義です。

■戻り値　API が正常終了したか、失敗したかを返します。
FALSE 失敗。
TRUE　正常終了。

■解　説　論理スロットに存在しているであろうデバイスを､アプリケーションが使用することをドライバに通知します。論理スロットにデバイスが存在していない場合は、エラーとなります。論理スロット内のデバイスがすでに他のアプリケーションで使用されている場合もエラーとなります。これにより、1 つのデバイスは単一のアプリケーションから排他的に使用されます。他の API を呼び出す前に必ずこの API を呼び出してください。

■エラー　APCI57_ERR_SLOT
論理スロット番号が不正です (または見つかりません)。
APCI57_ERR_NO_DEVICE
使用可能なデバイスがありません。
APCI57_ERR_IN_USE
デバイス使用中です。

７－３－２．Apci57Close

■機　能　デバイスの開放

■形　式　Visual C++

```
BOOL　Apci57Close
(
  WORD wLogSlot
);
```

■形　式　Visual Basic

```
Function　Apci57Close
(
  ByVal wLogSlot As Integer
) As Long
```

■形　式　Delphi

```
function　Apci57Close
(
  wLogSlot: WORD
): BOOL;
```

■入　力　wLogSlot

このデバイスを開放します。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。

TRUE　正常終了。

■解　説　アプリケーションがデバイスの使用を終了し、デバイスを他のアプリケーションに開放することをドライバに通知します。アプリケーションを終了する前に必ずこの API を呼び出してください。

■エラー　APCI57_ERR_SLOT

論理スロット番号が不正です。

７－３－３．Apci57GetVersion

■機　能　バージョン情報取得

■形　式　Visual C++

```
BOOL　Apci57GetVersion
(
  LPDWORD lpdwDllVer,
  LPDWORD lpdwDrvVer
);
```

■形　式　Visual Basic

```
Function　Apci57GetVersion
(
  lpdwDllVer As Long,
  lpdwDrvVer As Long
) As Long
```

■形　式　Delphi

```
function　Apci57GetVersion
(
  var lpdwDllVer: DWORD;
  var lpdwDrvVer: DWORD
): BOOL;
```

■入　力　lpdwDllVer
DLL のバージョン情報を格納する領域へのポインタ。
NULL 可。

lpdwDrvVer
DRIVER のバージョン情報を格納する領域へのポインタ。
NULL 可。

■出　力　*lpdwDllVer
DLL のバージョン情報。

*lpdwDrvVer
DRIVER のバージョン情報。

■戻り値　API が正常終了したか、失敗したかを返します。
FALSE 失敗。
TRUE　正常終了。

■解　説　DLL と DRIVER のバージョン情報を取得します。
それぞれのバージョンは 4 桁で表現されます。
初回リリースは 0.1.0.0 とし、0x00010000 となります。
最大値は 255.255.255.255 となり、0xffffffff となります。

7－3－4．Apci57GetResource

■機　能　リソース情報の取得

■形　式　Visual C++

```
#define     APCI57_MAX_MEM      9
#define     APCI57_MAX_IO      20
#define     APCI57_MAX_IRQ      7
#define     APCI57_MAX_DMA      7

typedef struct _APCI57RESOURCE
{
  DWORD   dwNumMemWindows;                      /* Not used */
  DWORD   dwMemBase      [APCI57_MAX_MEM];      /* Not used */
  DWORD   dwMemLength    [APCI57_MAX_MEM];      /* Not used */
  DWORD   dwMemAttrib    [APCI57_MAX_MEM];      /* Not used */
  DWORD   dwNumIOPorts;                         /* Num IO ports */
  DWORD   dwIOPortBase   [APCI57_MAX_IO];       /* I/O port base */
  DWORD   dwIOPortLength [APCI57_MAX_IO];       /* I/O port length */
  DWORD   dwNumIRQs;                            /* Num IRQ info */
  DWORD   dwIRQRegisters [APCI57_MAX_IRQ];      /* IRQ list */
  DWORD   dwIRQAttrib    [APCI57_MAX_IRQ];      /* IRQ Attrib list */
  DWORD   dwNumDMAs;                            /* Not used */
  DWORD   dwDMALst       [APCI57_MAX_DMA];      /* Not used */
  DWORD   dwDMAAttrib    [APCI57_MAX_DMA];      /* Not used */
  DWORD   dwReserved1    [3];                   /* Not used */
} APCI57RESOURCE;

typedef   APCI57RESOURCE      * PAPCI57R;

BOOL   Apci57GetResource
(
  WORD wLogSlot,
  PAPCI57R pres
);
```

■形　式　Visual Basic

```
Global Const APCI57_MAX_MEM = 9
Global Const APCI57_MAX_IO = 20
Global Const APCI57_MAX_IRQ = 7
Global Const APCI57_MAX_DMA = 7

Type APCI57RESOURCE
  dwNumMemWindows As Long                        ' Not used
  dwMemBase(1 To APCI57_MAX_MEM) As Long         ' Not used
  dwMemLength(1 To APCI57_MAX_MEM) As Long       ' Not used
  dwMemAttrib(1 To APCI57_MAX_MEM) As Long       ' Not used
```

```
  dwNumIOPorts As Long                                ' Num IO ports
  dwIOPortBase(1 To APCI57_MAX_IO) As Long            ' I/O port base
  dwIOPortLength(1 To APCI57_MAX_IO) As Long          ' I/O port length
  dwNumIRQs As Long                                   ' Num IRQ info
  dwIRQRegisters(1 To APCI57_MAX_IRQ) As Long         ' IRQ list
  dwIRQAttrib(1 To APCI57_MAX_IRQ) As Long            ' IRQ Attrib list
  dwNumDMAs As Long                                   ' Not used
  dwDMALst(1 To APCI57_MAX_DMA) As Long               ' Not used
  dwDMAAttrib(1 To APCI57_MAX_DMA) As Long            ' Not used
  dwReserved1(1 To 3) As Long                         ' Not used
End Type

Function  Apci57GetResource
(
  ByVal wLogSlot As Integer,
  pres As APCI57RESOURCE
) As Long
```

■形　式　Delphi

```
const
  APCI57_MAX_MEM = 9;
  APCI57_MAX_IO = 20;
  APCI57_MAX_IRQ = 7;
  APCI57_MAX_DMA = 7;

type
  TAPCI57RESOURCE = record
    dwNumMemWindows: DWORD;                                  { Not used }
    dwMemBase:       array [1 .. APCI57_MAX_MEM] of DWORD;   { Not used }
    dwMemLength:     array [1 .. APCI57_MAX_MEM] of DWORD;   { Not used }
    dwMemAttrib:     array [1 .. APCI57_MAX_MEM] of DWORD;   { Not used }
    dwNumIOPorts:    DWORD;                                  { Num IO ports }
    dwIOPortBase:    array [1 .. APCI57_MAX_IO] of DWORD;    { I/O port base }
    dwIOPortLength:  array [1 .. APCI57_MAX_IO] of DWORD;    { I/O port length }
    dwNumIRQs:       DWORD;                                  { Num IRQ info }
    dwIRQRegisters:  array [1 .. APCI57_MAX_IRQ] of DWORD;   { IRQ list }
    dwIRQAttrib:     array [1 .. APCI57_MAX_IRQ] of DWORD;   { IRQ Attrib list }
    dwNumDMAs:       DWORD;                                  { Not used }
    dwDMALst:        array [1 .. APCI57_MAX_DMA] of DWORD;   { Not used }
    dwDMAAttrib:     array [1 .. APCI57_MAX_DMA] of DWORD;   { Not used }
    dwReserved1:     array [1 .. 3] of DWORD;                { Not used }
  end;

PAPCI57RESOURCE = ^TAPCI57RESOURCE;
```

```
function  Apci57GetResource
(
  wLogSlot: WORD;
  pres: PAPCI57RESOURCE
): BOOL;
```

■入　力　wLogSlot
このデバイスのリソース情報を取得します。

pres
リソース情報を格納する領域へのポインタ。
NULL 不可。

■出　力　*pres
リソース情報。

■戻り値　API が正常終了したか、失敗したかを返します。
FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスに割り当てられているリソースを表示用に取得します。

■エラー　APCI57_ERR_SLOT
論理スロット番号が不正です。

７－３－５．Apci57GetSwitchValue

■機　能　ロータリスイッチの値を取得

■形　式　Visual C++

```
BOOL　Apci57GetSwitchValue
(
  WORD wLogSlot,
  LPDWORD lpdwSwitchValue
);
```

■形　式　Visual Basic

```
Function　Apci57GetSwitchValue
(
  ByVal wLogSlot As Integer,
  lpdwSwitchValue As Long
) As Long
```

■形　式　Delphi

```
function　Apci57GetSwitchValue
(
  wLogSlot: WORD;
  var lpdwSwitchValue: DWORD
): BOOL;
```

■入　力　wLogSlot

デバイスの論理スロット番号を指定。

lpdwSwitchValue

ロータリスイッチの値を格納する領域へのポインタ。
NULL は不可。

■出　力　*pdwSwitchValue

ロータリスイッチの値。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスに割り当てられているロータリスイッチの値を取得します。

■エラー APCI57_ERR_SYSTEM
Windows の GetLastError() をコールしてください。
APCI57_ERR_INVALID_SLOT
無効な論理スロット番号です。
APCI57_ERR_NO_CREATE
デバイスがクリエイトされていません。
APCI57_ERR_INVALID_ARGUMENT
lpdwSwitchValue が NULL です。

７－３－６．Apci57SetMode

■機　能　サンプリング動作モード設定

■形　式　Visual C++

```
typedef  struct  _APCI57MODE
{
  DWORD   dwChFirst;     // サンプリング開始チャネル (0～7)
  DWORD   dwChLast;      // サンプリング終了チャネル (0～7)
  DWORD   cwBuffer;      // バッファサイズ (ワード単位)
  DWORD   dwTimer;       // タイマにセットする値 (00H～3FH)
  DWORD   dwMode;        // 動作モード (1～6)
} APCI57MODE;

typedef CONST APCI57MODE       * PCAPCI57M;

BOOL  Apci57SetMode
(
  WORD wLogSlot,
  PCAPCI57M pcmode
);
```

■形　式　Visual Basic

```
Type APCI57MODE
  dwChFirst As Long          ' サンプリング開始チャネル (0～7)
  dwChLast As Long           ' サンプリング終了チャネル (0～7)
  cwBuffer As Long           ' バッファサイズ (ワード単位)
  dwTimer As Long            ' タイマにセットする値 (00H～3FH)
  dwMode As Long             ' 動作モード (1～6)
End Type

Function  Apci57SetMode
(
  ByVal wLogSlot As Integer,
  pcmode As APCI57MODE
) As Long
```

■形　式　Delphi

```
type
  TAPCI57MODE = record
    dwChFirst: DWORD;     { サンプリング開始チャネル (0～7) }
    dwChLast: DWORD;      { サンプリング終了チャネル (0～7) }
    cwBuffer: DWORD;      { バッファサイズ (ワード単位)}
    dwTimer: DWORD;       { タイマにセットする値 (00H～3FH) }
    dwMode:   DWORD;      { 動作モード (1～6) }
  end;

PAPCI57MODE = ^TAPCI57MODE;
```

```
function  Apci57SetMode
(
  wLogSlot: WORD;
  pcmode: PAPCI57MODE
): BOOL;
```

■入　力　wLogSlot
このデバイスのモードを設定します。

pcmode
モード設定データ。

■戻り値　API が正常終了したか、失敗したかを返します。
FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスのサンプリングモードを設定します。

pcmode
サンプリング情報

＜サンプリング情報について＞

dwChFirst、dwChLast
チャネルインクリメントサンプリングを行う場合は、サンプリング開始チャネルを dwChFirst に、サンプリング終了チャネルを dwChLast に設定します。dwChFirst と dwChLast に同じチャネルを指定した場合は固定チャネルでサンプリングされます。
この設定値はハード(aPCI-A57) にそのまま設定します。そのためインクリメント動作はハードの動作となります。

cwBuffer
cwBuffer で指定したサイズはそのままドライバ内部のバッファサイズとなります。(サンプリング方式＝バーストの場合)

dwTimer
タイマサンプリングを行う場合、サンプリングタイマを dwTimer で設定します。設定する値は、ハードウェアマニュアルを参照してください。

dwMode

dwMode	スタート方法	ｻﾝﾌﾟﾘﾝｸﾞ方式	CLKｿｰｽ	バッファモード	関連メッセージ
1	ソフトスタート	シングル	内部	ドライバメモリ	WM_EOC
2	ソフトスタート	バースト	内部(ﾀｲﾏ)	ドライバメモリ	WM_BUFF_HALF_FULL WM_BUFF_FULL WM_BUFF_OVERFLOW
3	外部ﾄﾘｶﾞｽﾀｰﾄ	シングル	外部	ドライバメモリ	WM_EOC
4	外部ﾄﾘｶﾞｽﾀｰﾄ	バースト	外部(ﾀｲﾏ)	ドライバメモリ	WM_BUFF_HALF_FULL WM_BUFF_FULL WM_BUFF_OVERFLOW
5	ソフトスタート	バースト	内部(ﾀｲﾏ)	デバイスメモリ	WM_BUFF_HALF_FULL WM_BUFF_FULL WM_BUFF_OVERFLOW
6	外部ﾄﾘｶﾞｽﾀｰﾄ	バースト	外部(ﾀｲﾏ)	デバイスメモリ	WM_BUFF_HALF_FULL WM_BUFF_FULL WM_BUFF_OVERFLOW

スタート方法

[ソフトスタート]は Apci57StartSampling を行った後すぐにサンプリングを開始します。

[外部トリガスタート]は Apci57StartSampling を行った後すぐにはサンプリングせず、ハード上の PI7 端子が 100nsec 以上の負論理パルスの立ち下りエッジを検出後、サンプリングを開始します。

サンプリング方式

[シングル]の場合は、1 度だけ変換を行います。dwTimer 及び cwBuffer の値は無視されます。
また、Apci57GetSamplingStatus で取得したデータ数はサンプリングが終了した時点で 1 となります。

[バースト]の場合は、Apci57StopSampling を行うまでサンプリングを行います。

CLK ソース

[CLK ソース＝内部、サンプリング方式＝シングル]の場合は、ボードの機能のシングルサンプリングを行います。タイマは使用せず、1 度だけサンプリングします。

[CLK ソース＝内部、サンプリング方式＝バースト]の場合は、ボードの機能のタイマサンプリングを行います。

[CLK ソース＝外部、サンプリング方式＝シングル]の場合は、外部トリガ検出後ボードの機能のシングルサンプリングを行います。タイマは使用せず、1 度だけサンプリングします。

[CLK ソース＝外部、サンプリング方式＝バースト]の場合は、外部トリガ検出後ボードの機能のタイマサンプリングを行います。

バッファモード

[ドライバメモリ]は、1 回のサンプリング毎に PC にデータを取り込みます。サンプリングレートが早い場合は PC に負担がかかり、最悪の場合データの取りこぼしが起こる可能性があります。
Apci57GetSamplingStatus のサンプリング済みデータ数は 1 毎に更新されます。

[デバイスメモリ]は、変換された A/D データは 4096 データ毎にまとめて取り込みます。
サンプリング中に Apci57GetSamplingStatus を実行した場合、サンプリング済みデータ数は 4096 毎に更新されます。
dwTimer を 1msec、cwBuffer を 5000 に設定した場合、Apci57GetSamplingStatus で取得したサンプリング済みデータ数が 5000 になるまでに、8194msec かかります。

■エラー APCI57_ERR_SLOT
論理スロット番号が不正です。
APCI57_ERR_CHANNEL
チャネル設定が不正です。
APCI57_ERR_TIMER
タイマ値が不正です。
APCI57_ERR_INVALID_ARG
不正な引数を指定しました。
APCI57_ERR_NOT_ENOUGH_MEM
十分なメモリが確保出来ませんでした。

７－３－７．Apci57StartSampling

■機　能　サンプリングを開始します。

■形　式　Visual C++

```
BOOL　Apci57StartSampling
(
  WORD wLogSlot
);
```

■形　式　Visual Basic

```
Function　Apci57StartSampling
(
  ByVal wLogSlot As Integer
) As Long
```

■形　式　Delphi

```
function　Apci57StartSampling
(
  wLogSlot: WORD
): BOOL;
```

■入　力　wLogSlot

このデバイスのサンプリングを開始します。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスのサンプリングを開始（できる状態に）します。
チャネルインクリメントとシングルサンプリングを組み合わせる場合は本 API を連続して呼ぶことによりチャネルが変化します。

■エラー　APCI57_ERR_SLOT

論理スロット番号が不正です。

APCI57_ERR_START

スタートに失敗しました。

７－３－８．Apci57StopSampling

■機　能　サンプリングを停止します。

■形　式　Visual C++

```
BOOL　Apci57StopSampling
(
  WORD wLogSlot
);
```

■形　式　Visual Basic

```
Function　Apci57StopSampling
(
  ByVal wLogSlot As Integer
) As Long
```

■形　式　Delphi

```
function　Apci57StopSampling
(
  wLogSlot: WORD
): BOOL;
```

■入　力　wLogSlot

このデバイスのサンプリングを停止します。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスのサンプリングを停止します。
サンプリング中以外の時に Apci57StopSampling を行ってもエラーにはなりません。

■エラー　APCI57_ERR_SLOT

論理スロット番号が不正です。

７－３－９．Apci57GetSamplingStatus

■機　能　サンプリングの状態を取得

■形　式　Visual C++

```
BOOL　Apci57GetSamplingStatus(
  WORD wLogSlot,
  LPDWORD lpcwNumData,
  LPDWORD lpdwStatus
);
```

■形　式　Visual Basic

```
Function　Apci57GetSamplingStatus
(
  ByVal wLogSlot As Integer,
  lpcwNumData As Long,
  lpdwStatus As Long
) As Long
```

■形　式　Delphi

```
function　Apci57GetSamplingStatus
(
  wLogSlot: WORD;
  var lpcwNumData: DWORD;
  var lpdwStatus: DWORD
): BOOL;
```

■入　力　wLogSlot

このデバイスのサンプリング状態を取得します。

lpcwNumData

サンプリング済みデータ数を格納する変数へのポインタ。

lpdwStatus

サンプリング状態のステータスを格納する変数へのポインタ。

■出　力　*lpcwNumData

サンプリング済みデータ数。

Apci57SetMode のバッファモードが[ドライバメモリ]の場合は、１毎に更新されます。

[デバイスメモリ]の場合は、4096 毎に更新されます。

*lpdwStatus

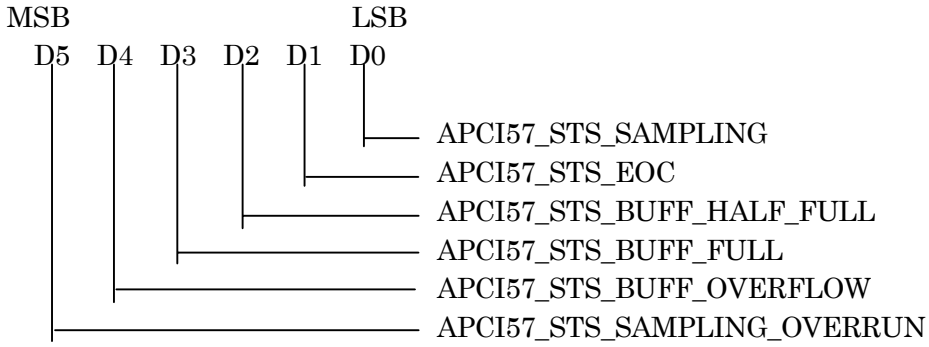
サンプリング状態のステータス。

■戻り値　API が正常終了したか、失敗したかを返します。
FALSE 失敗。
TRUE　正常終了。

■解　説　lpcwNumData は現在サンプリングされたデータ数が返ります。

サンプリング方式＝シングルの場合はデータがセットされた時点で 1 となります。
それ以外のモードでは、逐次サンプリングされていくごとにデータ数が更新されます。

lpdwStatus は、全てのサンプリング状態が取得されます。



APCI57_STS_SAMPLING
サンプリングを行っているときにこのビットが立ちます。
サンプリング方式＝シングルの場合は、スタートから EOC になるまでの間このビットが立ちます。それ以外のモードではスタートからストップまでの間このビットが立ちます。ただし、外部トリガスタートにした場合はトリガを受け付けた時点でこのビットが立ちます。

APCI57_STS_EOC
サンプリング方式＝シングルの時に、変換終了でセットされ、データを読み出した時点でリセットされます。
サンプリング中でないとこのビットは立ちません。

APCI57_STS_BUFF_HALF_FULL
リングバッファモードの時に、ドライバ内部のサンプリングバッファがハーフフルになった時点でこのビットが立ちます。
以降 HALF_FULL <= 状態 < FULL の間立ちっぱなしです。
サンプリング中でないとこのビットは立ちません。

APCI57_STS_BUFF_FULL
リングバッファモードの時に、ドライバ内部のサンプリングバッファがフルになった時点でこのビットが立ちます。
フルのときにのみこのビットが立ち、オーバーフローではビットはリセットされます。
サンプリング中でないとこのビットは立ちません。

APCI57_STS_BUFF_OVERFLOW
リングバッファモードの時に、ドライバ内部のサンプリングバッファがオーバーフローになった時点でこのビットが立ちます。
バッファの内容を読み出した時点でこのビットはリセットされます。

APCI57_STS_SAMPLING_OVERRUN
シングルサンプリングを除く全てのモードの時で、チャネルインクリメントモードの場合、チャネルに矛盾があった場合、このビットが立ちます。サンプリングは中止され、他の全てのビットはリセットされます。Apci57StopSampling を行った時点でこのビットはリセットされます。

```
#define APCI57_STS_SAMPLING          0x01
#define APCI57_STS_EOC               0x02
#define APCI57_STS_BUFF_HALF_FULL    0x04
#define APCI57_STS_BUFF_FULL         0x08
#define APCI57_STS_BUFF_OVERFLOW     0x10
#define APCI57_STS_SAMPLING_OVERRUN  0x20
```

■エラー　APCI57_ERR_SLOT
論理スロット番号が不正です。

７－３－１０．Apci57GetData

■機　能　サンプリングデータを取得

■形　式　Visual C++

```
BOOL　Apci57GetData
(
  WORD wLogSlot,
  WORD cwNumData,
  LPWORD lpwData
);
```

■形　式　Visual Basic

```
Function　Apci57GetData
(
  ByVal wLogSlot As Integer,
  ByVal cwNumData As Long,
  lpwData As Integer
) As Long
```

■形　式　Delphi

```
function　Apci57GetData
(
  wLogSlot: WORD;
  cwNumData: DWORD;
  var lpwData: WORD
): BOOL;
```

■入　力　wLogSlot

このデバイスのサンプリング済みデータを取得します。

cwNumData

読み出すデータ数。

lpwData

サンプリング済みデータを格納する領域へのポインタ。

■出　力　*lpwData

サンプリング済みデータ。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。

TRUE　正常終了。

■解　説　wLogSlot で指定された論理スロットのサンプリング済みデータを取得します。
cwNumData で指定したデータ数分取得します。

■エラー　APCI57_ERR_SLOT
　　論理スロット番号が不正です。
　　APCI57_ERR_NUMDATA
　　指定された数分のデータがバッファに存在しません。

７－３－１１．Apci57SetData

■機　能　D/A データを出力

■形　式　Visual C++

```
BOOL　Apci57SetData
(
  WORD wLogSlot,
  WORD wChData,
  WORD wDaData
);
```

■形　式　Visual Basic

```
Function　Apci57SetData
(
  ByVal wLogSlot As Integer,
  ByVal wChData As Integer,
  ByVal wDaData As Integer
 ) As Long
```

■形　式　Delphi

```
function　Apci57SetData
(
  wLogSlot: WORD;
  wChData: WORD;
  wDaData: WORD
): BOOL;
```

■入　力　wLogSlot
このデバイスにデータをセットします。

wChData
書き込み CH の指定 "0～1"

wDaData
D/A へのダイレクトデータ

■戻り値　API が正常終了したか、失敗したかを返します。
FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスに CH データと D/A の出力データを書き込みます。

■エラー　APCI57_ERR_SLOT
論理スロット番号が不正です。
APCI57_ERR_CHANNEL
チャネル指定が不正です。

７－３－１２．Apci57InPort

■機　能　入力実行

■形　式　Visual C++

```
BOOL　Apci57InPort
(
  WORD wLogSlot,
  LPBYTE lpbInValue
);
```

■形　式　Visual Basic

```
Function　Apci57InPort
(
  ByVal wLogSlot As Integer,
  lpbInValue As Byte
) As Long
```

■形　式　Delphi

```
function　Apci57InPort
(
  wLogSlot: WORD;
  var lpbInValue: BYTE
): BOOL;
```

■入　力　wLogSlot

デバイス指定。

lpbInValue

入力データを格納する領域へのポインタ。
NULL は不可。

■出　力　*lpbInValue

入力データ。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。
TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスのパラレル入力ポートから入力します。

■エラー　APCI57_ERR_SLOT

論理スロット番号が不正です。

７－３－１３．Apci57OutPort

■機　能　出力実行

■形　式　Visual C++

```
BOOL　Apci57OutPort
(
  WORD wLogSlot,
  BYTE bOutValue
);
```

■形　式　Visual Basic

```
Function　Apci57OutPort
(
  ByVal wLogSlot As Integer,
  ByVal bOutValue As Byte
) As Long
```

■形　式　Delphi

```
function　Apci57OutPort
(
  wLogSlot: WORD;
  bOutValue: BYTE
): BOOL;
```

■入　力　wLogSlot

デバイス指定。

bOutValue

出力データ。

■戻り値　API が正常終了したか、失敗したかを返します。

FALSE 失敗。

TRUE　正常終了。

■解　説　wLogSlot で指定されたデバイスのパラレル出力ポートに出力します。

■エラー　APCI57_ERR_SLOT

論理スロット番号が不正です。

７－３－１４．Apci57GetLastError

■機　能　エラーコード取得

■形　式　Visual C++

```
DWORD　Apci57GetLastError
(
  WORD wLogSlot
);
```

■形　式　Visual Basic

```
Function　Apci57GetLastError
(
  ByVal wLogSlot As Integer
) As Long
```

■形　式　Delphi

```
function　Apci57GetLastError
(
  wLogSlot: WORD
): DWORD;
```

■入　力　wLogSlot

このデバイスのもっとも最近起こったエラーのコードを取得します。
デバイスに依存しないエラーは wLogSlot の番号に関わらず取得されます。

７－４．定義

７－４－１．エラーコード

```
#define APCI57_SUCCESS                  0       // 正常終了
#define APCI57_ERR_NO_DEVICE            5000    // 使用可能なデバイスがありません
#define APCI57_ERR_RESOURCE             5001    // リソースエラー
#define APCI57_ERR_INVALID_ARG          5002    // 不正な引数を指定しました
#define APCI57_ERR_SLOT                 5003    // 論理スロット番号が不正です
                                                // (または見つかりません)
#define APCI57_ERR_CHANNEL              5004    // チャネル指定が不正です
#define APCI57_ERR_TIMER                5005    // タイマ値が不正です
#define APCI57_ERR_NOT_ENOUGH_MEM       5006    // 十分なメモリが確保できませんでした
#define APCI57_ERR_NUMDATA              5007    // 指定された数分のデータがバッファに
                                                   存在しません
#define APCI57_ERR_NOT_CREATED          5008    // デバイスがクリエイトされていません
#define APCI57_ERR_START                5009    // スタートに失敗しました
#define APCI57_ERR_INVALID_FUNC         5010    // 無効な関数呼び出しです
#define APCI57_ERR_IN_USE               5011    // デバイス使用中です
#define APCI57_ERR_INTERNAL             5100    // システムエラー
#define APCI57_ERR_UNKNOWN              5101    // システムエラー
```

APCI57_SUCCESS は、初期時に初期化される値です。
以降エラーが起こった場合はエラー内容が更新されるまでそのエラー値を保持します。

７－４－２．定数

```
#define APCI57_SLOT_AUTO                ((WORD) ~0U)
#define APCI57_MAX_SLOTS                16
#define APCI57_STS_SAMPLING             0x01
#define APCI57_STS_EOC                  0x02
#define APCI57_STS_BUFF_HALF_FULL       0x04
#define APCI57_STS_BUFF_FULL            0x08
#define APCI57_STS_BUFF_OVERFLOW        0x10
#define APCI57_STS_SAMPLING_OVERRUN     0x20
```

７－５．メッセージ

wParam には論理スロット番号が入ります。
lParam にはデバイス固有の ID が入ります。

WM_EOC

シングルサンプリング時の変換終了。サンプリング方式＝シングルでのみ発生。
タイマを使わず A/D 変換を行い、ハードの EOC が立った時点（EOC 割り込みのタイミング）で発生します。

WM_BUFF_HALF_FULL

ドライバ内部のサンプリングバッファのハーフフル。リングバッファとなった場合のみに発生。
指定されたバッファサイズ割る 2（端数切り下げ）の位置に書き込みを行った直後に発生します。
リングバッファを何周も走らせて連続して取りこぼしなくサンプリングしたい場合は WM_BUFF_HALF_FULL のタイミングで取り込みます。

WM_BUFF_FULL

ドライバ内部のサンプリングバッファのフル。リングバッファとなった場合のみに発生。
指定されたバッファサイズの最後の位置に書き込みを行った直後に発生します。
このメッセージは指定したサイズ分のデータを取り込んで終了したい場合に使用します(WM_BUFF_OVERFLOW, WM_BUFF_HALF_FULL のメッセージは無視します)。
終了は Apci57StopSampling で行います。

WM_BUFF_OVERFLOW

ドライバ内部のサンプリングバッファのオーバーフロー。リングバッファとなった場合のみに発生。
指定されたバッファサイズを超えて書き込みを行うタイミングで発生します。
実際には新たな書き込みは行われずアイドリング状態となります。また、読み込みを行って、バッファがフル未満となった時点で通常動作となります。

WM_SAMPLING_OVERRUN

サンプリング周期が速すぎる場合に割り込み処理が追従しない場合に発生します。
aPCI-A57 の場合はオーバーランをハード的に検出出来ないので、チャンルインクリメントの場合に限り、チャネルが歯抜けになったことを検出し、オーバーランとします。
オーバーランを検出した時点でサンプリングはストップされます。

パラレルポート(PI6)による割り込みは対応しておりません。

製品のお問い合わせについて

◆ お買い求めいただいた製品に対する次のようなお問い合わせは、お求めの販売店又は**株式会社アドテックシステムサイエンス**の各営業所にご連絡ください。

- お求めの製品にご不審な点や万一欠品があったとき
- 製品の修理
- 製品の補充品や関連商品について
- 本製品を使用した特注製品についてのご相談

◆ 技術サポート ――― 技術的な内容のお問い合わせは、「ファックス」「郵送」「E-mail」のいずれかにて、下記までお問い合わせください。また、お問い合わせの際は、内容をできるだけ詳しく具体的にお書きくださるようお願いいたします。

――――― 技術的な内容のお問い合わせ先 ―――――

株式会社 アドテック システム サイエンス　　テクニカルサポート
〒240-0005
神奈川県横浜市保土ヶ谷区神戸町 134　YBP ウエストタワー 8F
E-mail　support@adtek.co.jp
Fax　045-331-7770

改訂履歴

発行年月日　2003 年 12 月 18 日　第 1 版
2004 年 2 月 20 日　第 2 版
2005 年 3 月 22 日　第 3 版

aPCI-A57
ソフトウェアマニュアル

第 3 版発行　2005 年 3 月 22 日
発行所　株式会社 アドテック システム サイエンス
〒240-0005　神奈川県横浜市保土ヶ谷区神戸町 134
YBP ウエストタワー 8F
Tel　045-331-7575 ㈹　　Fax　045-331-7770

---

aPCI-011-050322