

## 第4章．ソフトウェア

《注1》 W I N D O W S 9 8 / M E 対応： 特に断りのない限りW I N D O W S 9 5 用ソフトがそのまま使用できます。

《注2》 W I N D O W S X P 対応： 特に断りのない限りW I N D O W S 2 0 0 0 用のソフトがそのまま使用できます。

### 4-1. インストール (ボードインストール後に行います。)

製品添付のソフトウェアは3.5インチ(1.44MB)フロッピーまたはCDに圧縮された形で格納されており、同メディア内のインストーラ“Setup.exe”実行により展開されます。

なお内容については充実・改良の目的で後日、追加・変更も有り得ます。

重要な変更があったときは同メディア内のドキュメントファイルに記すこととします。

#### 操作手順

インストール元：Dドライブ(CDROM)

インストール先：Cドライブ(HDD) の場合で例示。

(1) W I N D O W S 付属のエクスプローラで、  
D:¥INSTALL¥PCI¥DA¥TDA772 を開く。

(2) “Setup.EXE”を実行(ダブルクリック)する。

当操作以下によりTDA-772PCI関連プログラムが図4-1に示すロケーションに展開・インストールされます。

(2005年4月より以前のCDROMを使用する場合：DOS窓利用)

#### 操作手順

インストール元：Dドライブ(CDROM)

インストール先：Cドライブ(HDD) の場合で例示。

( はスペース)

C:¥WINDOWS>CD¥:【ENTER】

C:¥>CD D:¥INSTALL¥PCI¥DA¥TDA772【ENTER】

C:¥>D:INSTALL D: C:【ENTER】

各プログラムグループ(C, BASIC等)ごとにインストール実行の有無を問うてきますから、【Y】=yes, 【N】=no, で答えるだけで作業が進みます。

《注》 MS-DOSの環境変数“COMSPEC”が設定されていないか、または正常に設定されていないと本インストール・プログラムの作業が途中で停止してしまいます。 実行前に確認または設定しておきます。

= 設定例 = COMMAND.COMがCドライブの¥にある場合、  
>SET COMSPEC=C:¥COMMAND.COM【ENTER】

全ファイルをインストールした後のディレクトリ構造は図4-1のようになります。

図4-1. インストール後のディレクトリ

本図は原形です。 充実・改良の目的で後日、追加・変更も有り得ます。

¥	凡例	: サンプルプログラム番号等(1~)
MSCIENCE	T-C	: TURBO-C
	B-C	: BORLAND-C
UTILITY (本ボードの設定)	WIN95	CF9050DP.COM: コンフィギュレーション(95用)
	-WINNT	CF9050NT.EXE: コンフィギュレーション(NT用)
		-CF9050NT.sys: 上記NT用デバイスドライバ
		-REGSTDRV.EXE: デバイスドライバ登録・削除用
-BOARDTST-	-772QB1.EXE	: 本ボードの試運転・動作確認用プログラム
	-772QB1.COM	: 英語モードに切り替えた後、EXEを実行する
-SMP772C-	-MICROSOFT.H	: MS-C用ヘッダファイル
(各種Cサンプル)	-BORLAND.H	: TURBO-C, BORLAND用ヘッダファイル
	-INT772.C	: 外部割り込み入力時にDA出力動作
	-GPP772.C	: 最も簡単な2ch汎用DA出力動作
	-MSV772.C	: 複数ボードの同期DA出力動作
-SMP772B-	-772QB1.BAS	: Quick-Basic(4.5)用サンプル
(BASICサンプル)		
SMP772VB (Visual) (BASIC用) (サンプル)	SMP772.VBP	: プロジェクト
	DRIVER.FRM	: ドライバ関連フォーム
	SMP772.FRM	: メインフォーム
	-MODE.FRM	: モード設定フォーム
	DRVDLL.BAS	: 汎用IO制御ドライバ定義
	-MS_PCI.BAS	: PCIリソース取得定義
	TDA772.BAS	: ハードウェア定義
GL_DOS	MSPCID.H	: (DOS版)リソース取得ライブラリ・ヘッダ
	-MSPCIDM.LIB	: (MS-C用)ライブラリ/全モデル対応
	-MSPCIDT.LIB	: (T-C, B-C用)ライブラリ/全モデル対応
GL_W32	MS_PCI.H	: (Win32版)リソース取得ライブラリ・ヘッダ
	-MS_PCI.DLL	: リソース取得ライブラリDLL (Win95・98・NT兼用、要デバイスドライバ)
	-MS_PCI.LIB	: DLLインポートライブラリ
	-MS_PCI.BAS	: (VB/32bit版用)DLL関数定義モジュール
Hnd_95	専用ドライバ/関数DLL (WINDOWS95・98・ME用)	: 6-2項参照
Hnd_NT	専用ドライバ/関数DLL (WINDOWS NT4.0用)	: 6-2項参照
Hnd_2K	専用ドライバ/関数DLL (WINDOWS 2000・XP用)	: 6-2項参照

【注1】 本ボード専用のWINDOWS版ハンドラDLL/デバイスドライバは当作業の後、6-2項に従って必要なファイルを適合フォルダにコピーする必要があります。

【注2】 ボード依存性のない汎用のWINDOWS版I/O実行DLL/デバイスドライバは当作業ではインストールされません。WINDOWS95・98用はWin9xフォルダにありますので各ファイルを適合フォルダにコピーする必要があります。

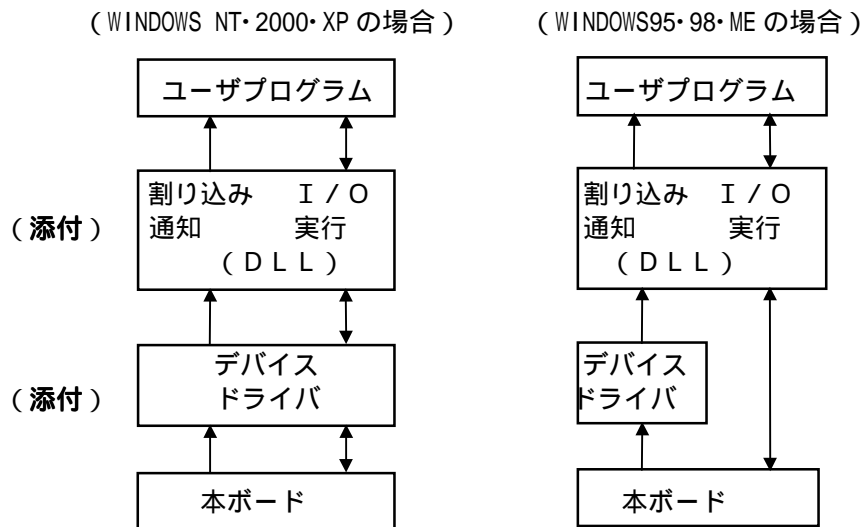
WINDOWS NT用はWinNTフォルダ中にあり、同フォルダ中の専用インストールで導入してください。またWINDOWS2000・XP用WDMドライバはボード自体のインストール時に自動インストールされます。使用法、サンプルなどはWin2Kフォルダ内にあります。

(追伸) CDROMの場合、Win9x、WinNT、およびWin2KフォルダはINSTALLフォルダ下のDriversフォルダ下にあります。

## 4-2. WINDOWSドライバについて

WINDOWS 9x / ME / NT / 2000 / XP 向に汎用 I / O 読み書き用 DLL が添付されています。

基本的には当 DLL (およびデバイスドライバ) を使用して本ボード上の各レジスタを読み書きすることでプログラミングが可能です。これらは御自身で (ボードにアクセスする部分の) ライブラリ等を製作する場合への便宜です。添付の本ボード専用 WINDOWS ドライバ / 関数ライブラリ (第6章) を利用する場合は不要です。



**WINDOWS 3.1 :** Win 3.1 フォルダ以下に格納されており、VB (2.0) で利用できます。C, C++ の場合は当 DLL を使用せずともインラインアセンブラで直接 I / O 命令を記述できます。割り込みを使用するときは、DOS 同様に直接制御で対応できます。

**WINDOWS 9x :** Win 9x フォルダ以下に格納されており、VB (4.0/5.0) で利用 (or ME) できます。ブロック I / O 命令もサポートされています。C++、C の場合は当 DLL を使用せずともインラインアセンブラで直接 I / O を記述できます。割り込みは DOS 同様に直接制御、またはデバイスドライバ (Pta95\_0.vxd) で対応します。

**WINDOWS NT :** Win NT フォルダ以下に格納されており、VB (5.0) で利用 (4.0) できます。ブロック I / O 命令、割り込みもサポートされています。

NT では I / O 制御、割り込み、共に必ずデバイスドライバが必要です。

本デバイスドライバは最大 16 枚のボードを (各単独に) 制御することができます。/ 当社製品でなくても可能 /

**WINDOWS 2000 :** Win 2K フォルダ以下に格納されており、VB (6.0) で利用 (or XP) できます。なおドライバ (WDM) は複数種類のボードで共用利用できるもので、第6章で説明する本ボード専用ハンドラ関数 DLL から利用します。(ボードインストール作業時にインストールされます。)

【注】WDM ドライバの性格から割り込みは使用できません。

詳細は %MSCIENCE%\WIN2K\DOC フォルダ内のテキスト参照。

### 4-3. ボードアクセス関連ライブラリ (WINDOWS2000/XP 以降では不要)

汎用リソース情報取得関数 `MS__PCI.DLL / WINDOWS9x・ME・NT 用` について

ユーザプログラムから本PCIボードにアクセス・制御するときはボードを検出し、リソース情報（ベースアドレス値、割り込み番号等）を取得する必要があります。第6章に記す本機の専用ハンドラ（専用ドライバ&関数DLL）を使用する場合は内部で処理されているため、必要ありませんが、ユーザプログラムから本ボードを直接制御するときは本DLL&ドライバが必要です。使用手順は、

#### 【1】PCIバス上のボードの検出

Int GetPciDevice (WORD VendeID, WORD DeviceID, WORD nNum, WORD Flag, WORD *magic)	
引数	VendeID: ベンダID、nNum: 検出対象ボード(0~) / 同ボード複数に対処・特定、DeviceID: デバイスID、Flag: 0 (固定)、*magic:マジック番号取得先ポインタ
戻り値	0:成功、-1:失敗

PCIボードの特定はロケーション（バス・デバイス・ファンクション）で行います。本ボードのベンダID、デバイスID、検出対象ボード番号（1枚目=0）を指定して実行すると同ボードのロケーションがmagicに得られます。同一ボードを複数インストールしているときは続いて検出対象ボード番号=1, 2, として実行すれば各ボードごとのロケーションが得られます。

ベンダID = 13FDH (マイクロサイエンス社PCIボード共通)、  
デバイスID = 105H (TDA-772PCI)。

#### 【2】指定ボード・指定レジスタのダブルワード読み込み（ベースアドレス値取得に使用できる。）

Int ReadPciDword (WORD magic, WORD reg, DWORD *data)	
引数	magic: 【1】GetPciDeviceで得られたマジック番号、*data:データ取得先ポインタ、reg: PCIコンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0:成功、-1:失敗

#### 【3】指定ボード・指定レジスタのワード読み込み（通常は不使用。）

Int ReadPciWord (WORD magic, WORD reg, WORD *data)	
引数	magic: 【1】GetPciDeviceで得られたマジック番号、*data:データ取得先ポインタ、reg: PCIコンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0:成功、-1:失敗

PCIリソース情報の取得は【1】で検出したロケーション（magic）とレジスタ番号を指定して行います。物理ベースアドレス値はダブルワードなので【2】の関数を使用します。

各レジスタは本ボード上のPCIインターフェース素子9050（PLX社製）内にあり、次のように割り付けられています。

レジスタ番号10H : PCIインターフェース素子9050で使用。  
14H : 未使用  
18H : I/OマップレジスタのベースアドレスBASE1 (3-2項)。  
1CH : 未使用  
20H : 未使用  
24H : 未使用

なお、

I/Oマップでは得られた data の下位 2 bit をマスクした値がベースアドレス値です。

bit31	bit 2 1 0
物理ベースアドレス	× 1

【4】指定ボード・指定レジスタのバイト読み込み（割り込み番号値取得に使用できる。）

Int ReadPciByte (WORD magic, WORD reg, BYTE *data)	
引数	magic: 【1】GetPciDevice で得られたマジック番号、 *data: データ取得先ポート、 reg: PCI コンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0: 成功、 -1: 失敗

割り込みリソース情報取得も【1】で検出したロケーション (magic) とレジスタ番号を指定して行います。 data 値はバイトなので【4】の関数を使用します。

本レジスタも本ボード上のPCIインターフェース素子9050 (PLX社製)内にあり、レジスタ番号 = 0EHです。

得られる data 値 (1 ~ 15) は割り込み番号です。 / 不使用時 = 0、または255 / プログラム内ではベクタに変換して御使用ください。

なお、

本ボードの標準設定では (プラグアンドプレイ実行時に) 割り込みリソースを要求しません。割り込みを使用するときは1 - 5項に記す手順で設定を変更してください。

## 汎用リソース情報取得関数ライブラリ / MS-DOS 版 について

## 関数一覧

## 【1】PCIバス上のボードの検出

Int_far GetPciDevice(WORD VendelD,WORD DeviceID,WORD nNum,WORD Flag,WORD _far *magic)	
引数	VendelD : ベンダ ID、 nNum : 検出対象ボード ( 0 ~ ) / 同ボード複数に対処・特定、 DeviceID : デバイス ID、 Flag : 0 ( 固定 )、 *magic :マジック番号取得先ポインタ
戻り値	0:成功、 -1 : 失敗

## 【2】指定ボード・指定レジスタのダブルワード読み込み ( I / O アドレス値取得に使用できる。 )

Int ReadPciDword (WORD magic , WORD reg , DWORD _far *data)	
引数	magic : 【1】GetPciDevice で得られた マジック番号、 *data : データ取得先ポインタ、 reg : PCI コンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0:成功、 -1 : 失敗

## 【3】指定ボード・指定レジスタのワード読み込み ( 通常は不使用。 )

Int ReadPciWord (WORD magic , WORD reg , WORD _far *data)	
引数	magic : 【1】GetPciDevice で得られた マジック番号、 *data : データ取得先ポインタ、 reg : PCI コンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0:成功、 -1 : 失敗

## 【4】指定ボード・指定レジスタのバイト読み込み ( 割り込みレベル値取得に使用できる。 )

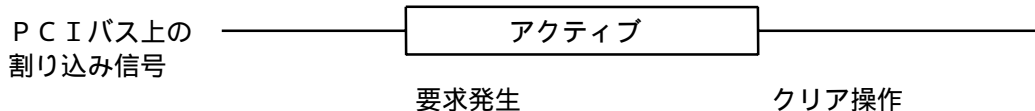
Int ReadPciByte (WORD magic , WORD reg , BYTE _far *data)	
引数	magic : 【1】GetPciDevice で得られた マジック番号、 *data : データ取得先ポインタ、 reg : PCI コンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0:成功、 -1 : 失敗

**使用方法 :** 前記W I N D O W S 版と同様です。

#### 4-4 . 割り込みについて

PCIバス上の割り込み信号は、これを検知したソフトウェアからクリア操作を行うまでアクティブ状態を（要求元側が）維持する“**レベル動作**”です。この仕組みでは複数のデバイスが1本の割り込みリソースを共有することもできます。

図 4 - 4 .



#### 要注意

当社製を始め、多くのISAバスボードの割り込みは要求元がパルス状の単発信号を発信する“**エッジ動作**”ですから割り込み要求のアクティブ状態は自動解消されるのですが、PCIバス上の“**レベル動作**”ではプログラム開発中などの事情で適切なクリア操作が行われなかった場合のハングアップ等、非常事態解消のためのハードウェアリセット（電源OFF）を余儀なくされることも考えられます。このような場合はハードディスクのクラッシュ等の大きな損害が発生する恐れがあります。

添付の（WINDOWS 95 / NT用）デバイスドライバでは汎用のため、アプリケーション側からクリア操作に必要なパラメータを（あらかじめ）受け取っておき、割り込みが発生したらクリア操作を実行します。またアプリケーション側からは割り込み発生（回数：Read Clear）を読み取る関数DLLをポーリングする形をサポートしていますが、このようなアルゴリズムは（割り込みを使用せず）ボードのステータスをポーリングする方法と等価ですから、無用なトラブルを回避するためにも後者をお勧めします。

なお、本ボード上のROMに書き込まれているデフォルト（初期）のコンフィギュレーション情報では（プラグアンドプレイの動作時に）割り込みリソースを要求しません。もし要求したときに空きが無く拒否されるとI/Oアドレスの割り当ても受けられず認識不能状態になる恐れがあるからです。割り込みを利用するときはリソースに空きがあることを確認してから添付のコンフィギュレーション・ユーティリティで（割り込みリソースを要求するように）修正してください。【1 - 5 項.参照】

（追伸） 一部のパソコンは標準状態で割り込みリソースに空きが無いものがあります。

### 割り込み制御の手順

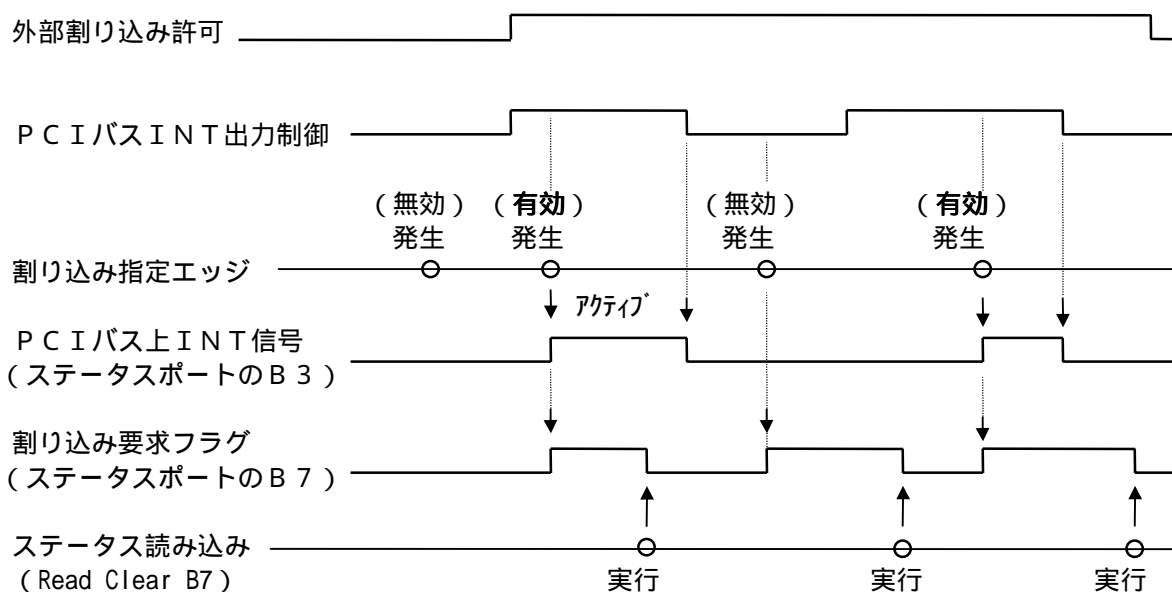
割り込み制御ポート【3-4項】で外部割り込み許可ビット（B7）をセット、また、エッジ（B6）を指定します。これだけの場合、外部割り込み【UPD-IN】に有効な信号エッジが入力されても《PCIバス上の割り込み要求信号》はアクティブになりません。ステータスポート【3-7項】の割り込み要求フラグ（B7）がセットされるだけです。さらにPCIバスINT出力制御ビットB0をセット（=1）することにより実際の割り込み要求が出力（アクティブ）可能になります。

外部割り込み【UPD-IN】に指定エッジが入力され、割り込み要求が発生しました。

割り込み要求が受け付けられた場合、通常はデバイスドライバ内でこれをクリアします。操作は割り込み制御ポート【3-4項】のPCIバスINT信号制御ビットB0をクリアします。但し同ビットを再びセットするまではクリア状態で、次の割り込み要求が発信できない状態なので通常は続けてセットします。この様子はステータスポートのPCIバス上INT信号ビット（B3）に反映されますが、通常はデバイスドライバ内ですぐクリアされるためユーザプログラムからの検出は困難です。（必要もないでしょう。）

PCIバス上の割り込み要求信号はボード・リセット【3-3項】でもクリアできます。

図4-4B. 割り込み制御・ステータス変化タイミング



上記タイミング図では各制御ビットの効果を示すため【UPD-IN】の有効エッジが必要以上に入力された場合で記してあります。

### ポーリングによるイベント制御 (割り込み不使用)

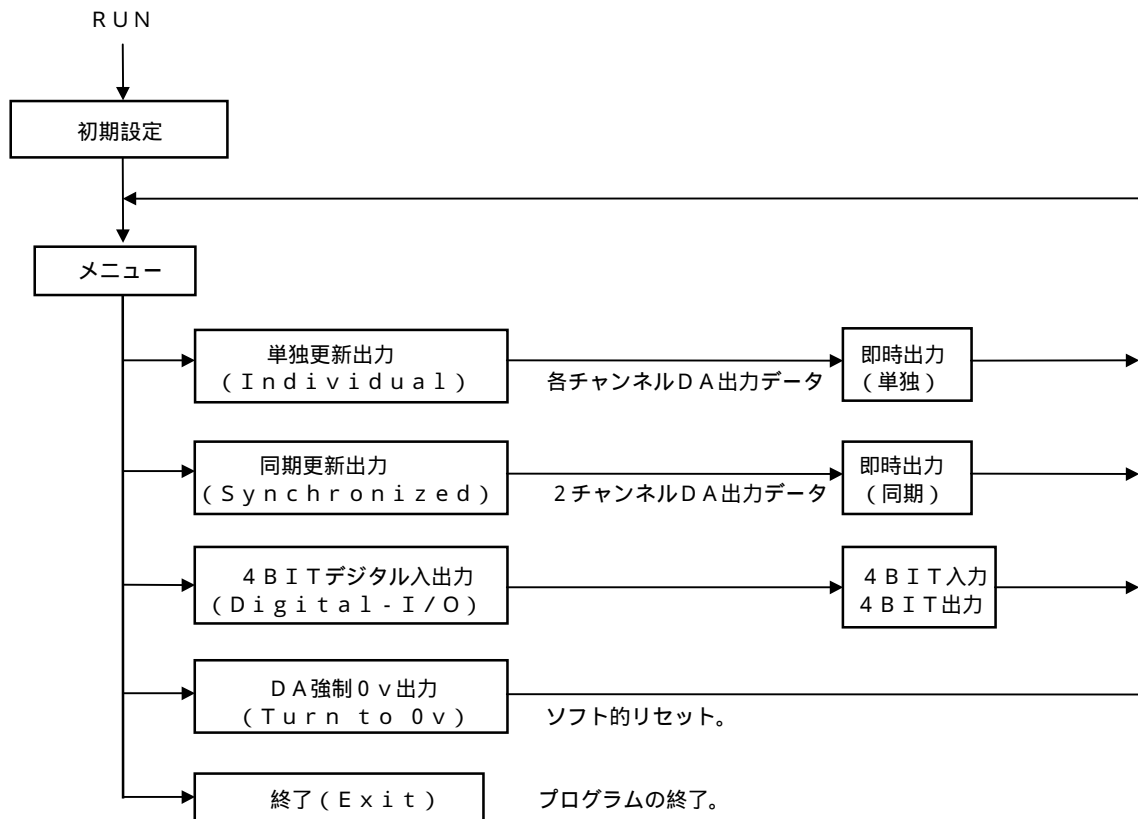
前 で説明したように割り込み制御ポートの外部割り込み許可ビット（B7）をセット、エッジ（B6）を指定しておき、PCIバス上INT出力制御ビット（B0）がクリア状態でも、外部割り込み【UPD-IN】に有効エッジが入力される毎にステータスポートの割り込み要求フラグがセットされます。これをポーリングして外部イベントの発生を検出する方法があります。（推奨します。）



## 4-5. Quick - Basicのサンプル

Quick - Basic (4.5)用のサンプルプログラム“772QB1.BAS”は基本的なBASIC文のみによる使用例です。なお本プログラムの実行形式“772QB1.EXE”は試運転・動作確認用にもなります。コーディングの詳細は同ソースのリストを御覧ください。

図4-5.“772QB1.BAS”のフロ-概要



## 4-6. Cのサンプル

単純な2チャンネルD A即時更新出力、 外部割り込み入力によるD A更新出力、および複数ボード（ここでは2枚＝4チャンネル）のD A同期更新出力例を示します。  
以下にアルゴリズムの概要を記します。 具体的にはソース【\* . C】を御参照ください。

図4 - 6 A . 汎用2チャンネルD A動作【GPP772 . C】

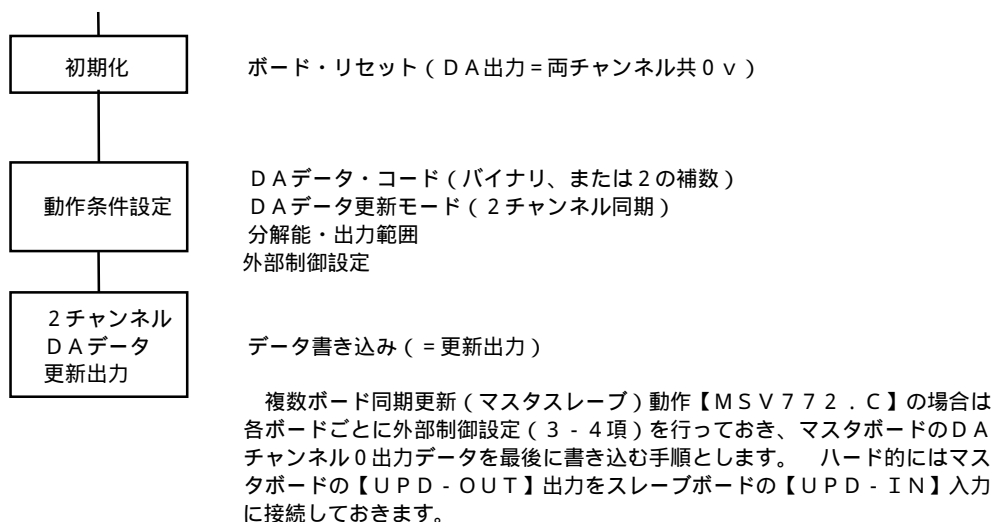
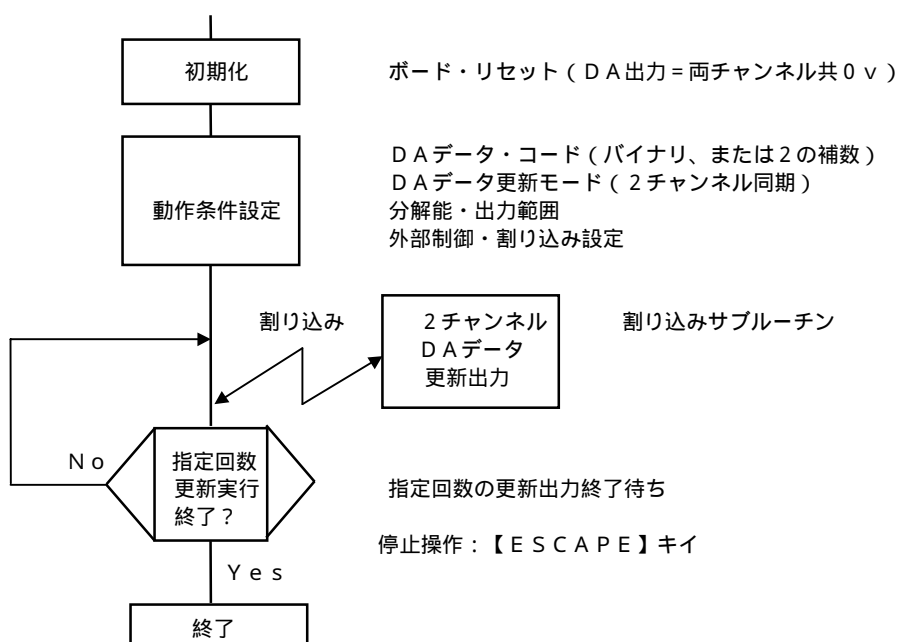


図4 - 6 B . 外部割り込み入力によるD A更新出力【INT772 . C】



## 4-7. Visual Basic (32BIT 版) のサンプル

WINDOWS 9x・ME、またはNT上で本ボードの機能を一通り動作させてみるものです。  
Visual Basic (4.0)で作成されており、VB (5.0/6.0)でも動作します。

当サンプルは当社提供のWINDOWS 9x・ME・NTの汎用I/Oドライバ/DLLを使用  
したものです。【1-5項参照】 第6章に記す専用ドライバ/関数DLL (推奨)にもVBサン  
プルがありますが、両者は無関係です。なお**本サンプルプログラム中の制御には割り込みを使用  
していません。** 割り込みリソースがあり、ドライバに登録した場合は外部割り込み入力要因  
に選択して (単に動作確認のため) 発生回数を表示するだけです。

またWINDOWS 9x・MEの場合で割り込みを使用しないときはドライバのインストールが  
不要です。(9x・MEでのI/O操作はDLLが直接ハードウェアにアクセスするため)

【注1】 本ボードの (プラグアンドプレイに対する) コンフィギュレーション初期設定は  
割り込みリソースを要求しません。 割り込みを使用するためには添付のユーティ  
リティを利用して設定変更を行う必要があります。(1-5項参照)

【注2】 当サンプルで使用するWINDOWS版I/O実行DLL/デバイスドライバは  
4-1項の作業だけではインストールされません。 WINDOWS 95・98用は  
Win9xフォルダにありますので各ファイルを適合フォルダにコピーする必要が  
あります。 NT4.0用はWinNTフォルダ中にあり、同フォルダ中の専用イン  
ストーラで導入してください。(1-5項参照)

(追伸) CDROMの場合、Win9xおよびWinNTフォルダはINSTALL  
フォルダ下のDriversフォルダ下にあります。

表 4 - 7 .

ソフトウェア要素	OS	使用するモジュール/ファイル
デバイスドライバ、および インタフェースDLL	9x (ME)	pta95_0.vxd accs_95.dll
	NT (4.0)	NtPta_?sys (? : 0 ~ 15 任意の整数) port_nt.dll
Visual Basic サンプルプログラム モジュール構成	9x・ME NT 共通	smp772.vbp (プロジェクト) smp772.frm (メインフォーム) mode.frm (モード設定フォーム) driver.frm (ドライバ関連フォーム) drvddl.bas (DLL定義) ms_pci.bas (PCIリソース取得定義) tda772.bas (ハードウェア定義)

本サンプルプログラムソースはWINDOWS 9x・ME・NT共通ですが、使用するDLLが  
異なるのでDLL関数定義用の標準モジュール (drvddl.bas) 先頭で、

```
#Const DRIVER = "Accs_95"      ' 9x・MEの場合 (デフォルト)
' #Const DRIVER = "Port_Nt"    ' NT4.0の場合
```

の定義により条件コンパイルで対応し、WINDOWS 9x・NTでは関数名が異なるOPEN/  
CLOSEをAlias機能を使用してプログラム本文中では同一名で扱えるようにしています。

さらにOPEN操作はWINDOWS 9x・ME/NTの各DLLではパラメータが異なります  
から、frmMainのLOADイベント中で対応したパラメータをcboDrvParamに  
セット、その値を元にOPEN関数のパラメータとしています。

《操作方法》 操作手順は ドライバのオープン、 以後はボードの各機能実行の順です。  
 終了時は必ずドライバのクローズ操作を行います。  
 テキストボックスに記入する値は全てHEX表記です。  
 (例) 12BITデータ800Hなら3文字“800”と入力します。

---

 スタートアップ・フォーム (frmDriver)
 

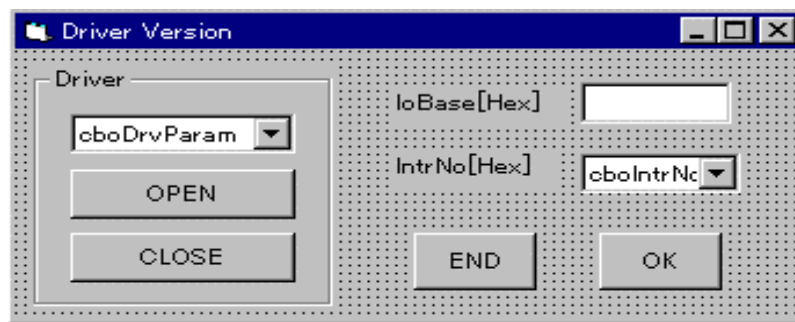
---

本ボードのサンプルプログラムはドライバを開閉する当フォームから実行を開始します。

cmdOk\_Click【OKボタン】の処理で当frmDriverを非表示し、frmMainをモーダルウィンドウで表示します。これがアプリケーションの本体となります。

frmMainのcmdEnd\_Click【ENDボタン】の処理はfrmMainをアンロードします。これによりfrmDriverに制御が戻ってドライバのクローズ処理が行われ、プログラムを終了します。

frmMainのcmdEndの処理が“END”ではなく“Unload frmMain”であることに注意してください。またfrmMainで使用する“iobase”“intr\_no”“devno”の各変数はtda772.basでグローバル宣言してあります。



OPEN

<cmdOpen> : デバイスドライバをオープンします。  
 WINDOWS 9x・MEの場合は当ボタンをクリックするだけ、  
 NTの場合は使用するデバイスドライバの枝番号(0~15)を指定して  
 から当ボタンをクリックします。

CLOSE

<cmdClose> : デバイスドライバをクローズします。

OK

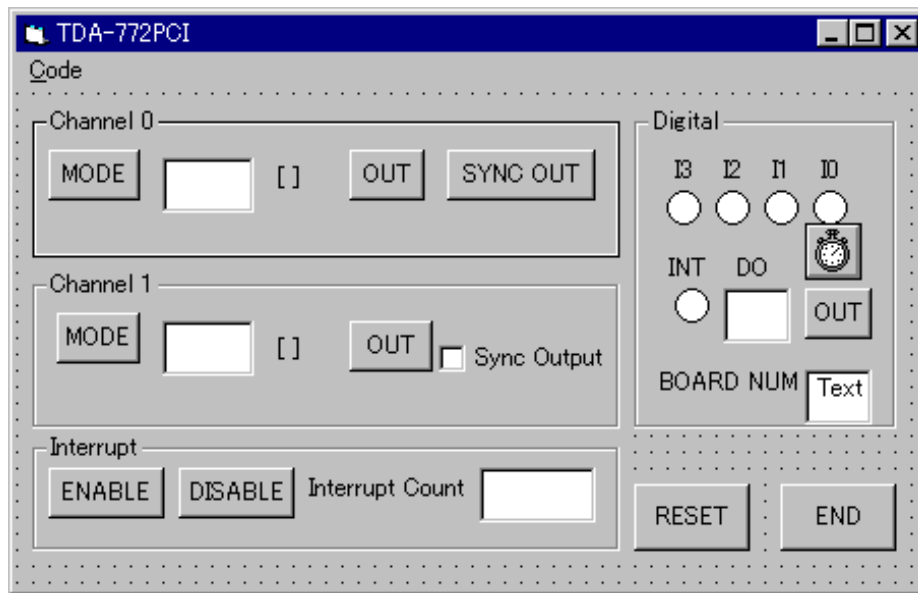
<cmdOk> : プラグアンドプレイで割り当てられているI/Oベース  
 アドレスと割り込み番号が(HEXで)テキストBOXに表示されます。  
 クリックするとボードの存在を確認し、存在していれば“frmMain”に  
 制御を渡します。

また、(“frmMain”のアンロードで)制御が戻って来たらドライバの  
 クローズを行い、終了します。

END

<cmdEnd> : 終了します。

## 《メインフォーム / 実行画面》



**R E S E T** < cmdReset > ボードをリセットします。

**E N D** < cmdEnd > 当 f r m M a i n を閉じ f r m D r i v e r に制御を戻します。

( channel 0 , channel 1 フレーム ) : D A 出力

**M O D E** < cmdCh0Mode > D A 出力モード設定フォーム f r m M o d e を開きます。  
< cmdCh1Mode >

**O U T** < cmdCh0Out > txtCh0Data , txtCh1Data に入力された値を D A 変換・更新出力  
< cmdCh1Out > します。 出力値は次回操作まで保持されます。  
B i t / R a n g e / M o d e / 2 ' s C o m p のいずれかが前回から変更されている場合は M o d e S e t D a プロシージャで各設定を更新したうえで行います。  
( 2 ' s C o m p : セットなら 2 の補数、リセットならバイナリ )

**S Y N C O U T** < CmdSyncOut > channel 1 フレームの SyncOutput チェックボックスがチェックされているときチャンネル 0、1 を同時に更新出力します。

( Digital フレーム )

**タイマー** < tmrGetDi > タイマのプロパティで設定した時間々隔で汎用デジタル入力と外部割り込み入力を調べ、シェイプコントロールのランプに表示します。  
また、デバイスドライバの G e t I n t r C o u n t ( ) 関数から得た割り込みの発生回数値を t x t I n t r C n t に表示します。

**O U T** < CmdDo > t x t D o D a t a に入力された汎用デジタル出力値を更新出力します。 出力値は次回操作まで保持されます。  
( 4 ビットなので 0 ~ F と記入 / 3 - 8 項参照 )

## ( Interrupt フレーム )

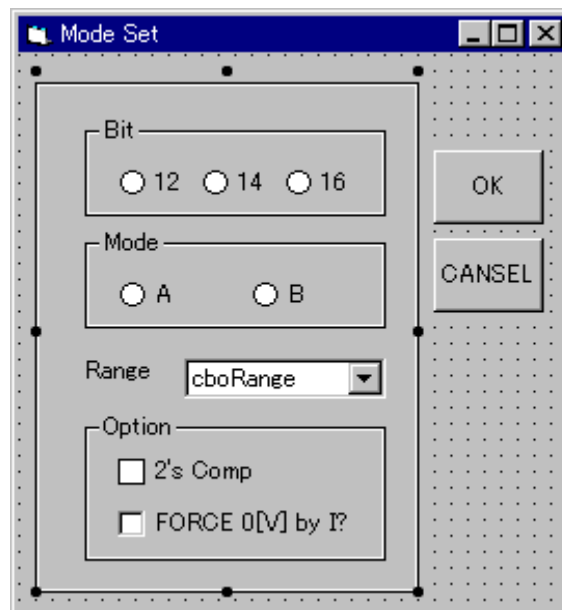
E N A B L E

<CmdIntrEn> 割り込みの発生回数を示す I n t r C o u n t 値をクリアし、本ボードからの割り込み信号出力を許可します。

D I S A B L E

<CmdIntrDis> 本ボードからの割り込み信号出力を禁止します。

《モード設定フォーム / 実行画面》



( Bit フレーム ) : < fraBit , optBit > D A データのビット長 ( 分解能 ) を選択します。

( Mode フレーム ) : < fraMode , optMode > D A 出力範囲の定義モードを選択します。  
/ 2 - 2 項参照。 /

( Range リストボックス ) : < cboRange > D A 出力範囲を選択します。

## ( Option フレーム )

2's Comp チェックボックス : Range リストボックスでバイポーラ ( ± ) 出力範囲が選択されているときに当チェックボックスをチェックすると、16 進数指定の D A データが 2 の補数コードとして扱われます。 / チェックされていないときはオフセットバイナリ / ユニポーラ ( 0 ~ + F s ) 出力範囲が選択されているときの 16 進数指定 D A データはストレートバイナリ扱いです。

FORCE 0[V] by I? チェックボックス : チェックされているとき、汎用デジタル入力による強制 0 v 制御が有効になります。  
( 3 - 5 項参照 )

## 第5章．保守・その他

### 5-1. 故障・トラブル等の原因と対処

本機は【DOS/V系パソコン】+【拡張ボックス】のシステム構成で全数検査のうえ出荷されています。お手元での動作確認方法は1 - 6項に記されています。動作に不具合があるときは以下の諸点を再点検してください。それでも不明なときは巻末の【Q & Aフォーム】にシステム構成（特に外部機器の接続回路）等の動作条件を御記入のうえ、技術部宛FAXしてください。

迅速に応答する体制となっています。なおTELいただく場合も、客観情報の整理・評価は問題解決のスピードアップにつながりますから、事前に【Q & Aフォーム】をFAXしてください。

#### 再点検・確認ポイント

- |             |  |
|-------------|--|
| (1) I/Oアドレス | ボードのインストール/認識は成功したか?(1 - 5項)   |
| (2) 割り込みレベル | リソースは取得できたか?(1 - 5項)   |
| (3) アナログ出力  | 負荷: 5 K 以上、1 0 0 0 p F 以下(2 - 1項)  |
| (4) デジタル入力  | 本ボードのTTL入力(外部制御、および汎用4ビット)に接続できる信号源はTTL(LS、CMOS等を含む5v電源動作素子)に限ります。現場で不適切な信号源を接続したために本ボード内のTTL入力素子を破損する事故が頻発していますので御注意ください。(次ページ/図5 - 1 参照) |

#### 動作確認方法

当社では原則として、ユーザ作成のソフトウェアについては評価しません。動作確認は本製品添付の当社製プログラム(1 - 6項)の実行結果について推測・適否・判定を行います。

QAリクエスト時には当プログラムの実行結果をレポートしてください。

ボード内TTL入力素子破損の主な原因

TTL入力素子の絶対最大定格は【負側：- 0.6 v】【正側：+ 7 v】です。このレベルを一瞬でも超えると入力素子破壊の原因になります。主な危険要素は、

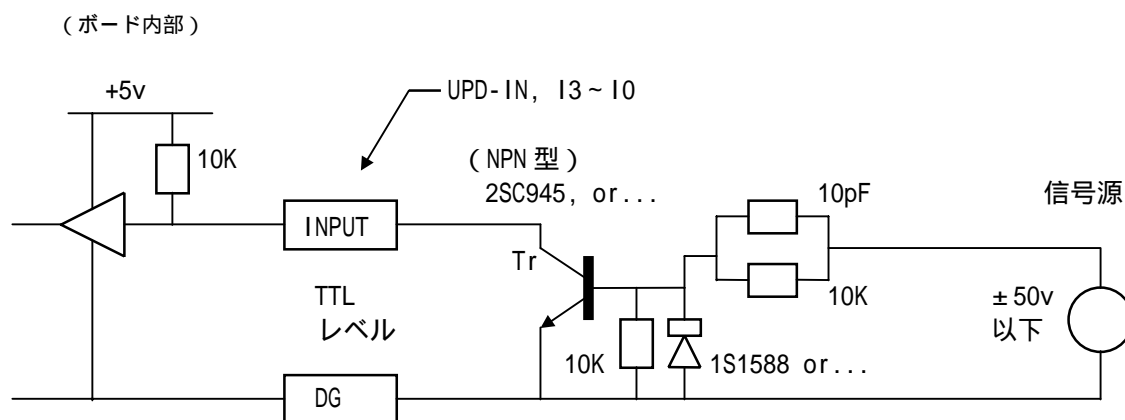
ファンクション・ジェネレータ等の交流信号出力を接続して破損させる例が多いようです。矩形波でも±に振れる信号は接続できません。特に、負側の許容レベル【- 0.6 v】が低いことに注意してください。

+ 5 v以上に振れるロジック信号も接続できません。12 v ~ 24 v電源を使用する機器からのデジタル信号は不可、信号レベルが不明なときは信号源の電源電圧が目安になります。

アナログ信号源は± 15 v電源によるオペアンプ出力が多く危険です。なお、TTL入力にアナログ信号を接続しても立上り / 立下り特性等が仕様を満足せず、正常な動作は期待できないでしょう。

信号源と本ボードのグランド・レベルに差があるときも危険です。（テストで測定可能）

図5 - 1 . 【高レベル信号】 【TTLレベル】変換回路例



《注》本回路はインバータ（極性反転）です。



## 5-2. 修理のときは

入手経路の如何にかかわらず当社宛に直接お申しつけください。 商社等を経由されますと時間がかかるだけでなく、情報交換の不便、費用の面でも不利になります。 なお当社では修理依頼を受けた製品が検査の結果、良品と判定された場合は（保証期間内でも）手数料を申し受けます。

特に最初からの不具合には誤解や情報不足によることが多いので、事前に御相談ください。

【Q & A フォーム】が便利です。

### 無償修理

納入後 1 年以内の自然故障、および当社製造上の問題に起因した故障に対しては無償修理を行います。 但し、故障・不具合の原因や無償修理の対象となるか否かは（過去の経験等に照らして）当社側で判定させていただきます。

なお当社では保証書を発行していませんが、社内では製造番号と出荷年月日の記録を基に判定しています。

### 有償修理

落雷等の自然現象、漏電・過電圧印加・機械的破損・その他、ユーザ側の責に帰する故障品、または納入後 1 年間を経過した製品の自然故障に対しては実費・有償にて修理をお請けします。 性格上、事前見積もりは不可能ですが、制限額を事前通知いただければ、作業過程で制限を超えそうな見通しがたった時点で連絡・相談させていただきます。

受け渡し : 宅配便によるセンドバックで行います。

修理期間 : 全んどの場合、当社内で 24 時間以内に完了・返送しています。時間を要する場合は御連絡いたします。

費用の目安 : 修理費用は事務管理手数料、技術者の所要時間（1 時間単位）手数料、および交換部品代の合計です。 2002 年 9 月現在（時勢により変動します）では、

事務管理手数料（1 件当り、返送運賃含）：＝ ¥ 4,000

修理時間手数料：＝（時間単価 ¥ 6,000）× 所要時間

交換部品代 :＝ ¥ 実費

故障経緯、システム客観情報の添付は時間の節約・コストダウンに有効です。典型的な事例では費用合計が ¥ 20,000 を超えることは希れです。

【注 2】 当社製品に対してユーザが改造を行った場合は、当社サポートの対象外になります。 改造とは製品に新たな部品を追加実装、または実装部品を削除したり、回路パターン・接続に変更を加えることです。 なお、当社がオプションとして供給、または指定した部品の追加実装・交換はこの限りではありません。

### 5-3. 再調整

動作テスト・確認の方法は【1 - 6 項】のとおりです。同テストから得られた値に出力範囲の変化やオフセットが認められるときは再調整が必要です。アナログ回路は経年・環境変化に対する保守を定期的に行うことが望ましく、夏冬の使用環境（周囲温度）に差がある場合は季節単位、通年安定した使用環境の場合は1～2年に1度は校正することが理想的です。

再調整の方法・手順を以下に記しますが、極細のドライバ、デジタル電圧計を必要とし、手順もやや複雑ですから御希望により当社でも（実費で）お請けします。

== 準備 ==

本ボード上の諸設定は出荷時の状態（1 - 2 項）とします。

本ボードをパソコン本体または拡張 I / O スロットに装着、インストールします。

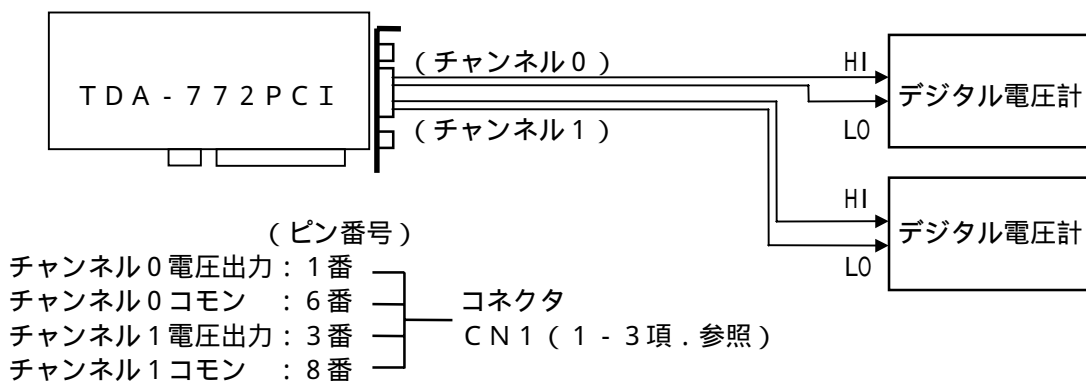
当作業の手順・詳細については1 - 5 項に記されています。

（以上、通常の使用状態）

図5 - 2のように、本ボードのDAチャンネル0、1出力をデジタル電圧計に接続します。

以上で準備完了です。電源投入順序は全機器同時、または外部機器を先にパソコン本体を最後に行います。電源切断は逆順序です。

図5 - 2 . 再調整用の測定機接続



電源を投入、MS - DOS システムを立上げます。再調整に使用するプログラムは試運転でも使用した“772QB1”です。>772QB1【ENTER】でプログラムが走り始めます。（WINDOWS 9x・MEのDOS窓で動作可、NT・2000・XPでは不可）  
10分程度のウォームアップ後、以下の調整作業を行います。

メニュー - からDA出力（単独、または同期更新）を選択し、表2 - 2 D ~ J の目標値を得よう、オフセット調整とゲイン調整を交互に2～3回繰り返して最適位置を求めます。

調整はユーザシステムに都合のよい出力範囲・分解能で実施してください。なお、本機の製造時は0～+10V（Aモード）・14BIT分解能で調整されています。

常温で製造調整時の正確度：0.04%FS（14ビット、0～+10V範囲/Aモード）  
その他の出力範囲：0.06%FS

### 電圧出力調整

ゲイン調整 : 指定D A出力値をフルスケール付近とし、出力電圧が指定値と一致するようにT M 1を調整する。

ゲイン調整 : 指定DA出力値を+1.0V付近とし、出力電流が整合値2.0mAとなるようにTMI1を調整する。

【注】電圧・電流出力値の換算式：
$$\begin{aligned} V &= (10/16) I - (10/4) \\ (V) \quad (mA) \quad I &= (16/10) V + 4 \end{aligned}$$

## 表 5 - 3 A . 正確度

アナログ出力範囲	非直線性 %FS	相対正確度 %FS	絶対正確度 %FS
最終調整範囲 (14BIT / 0 ~ 10V / A <sub>FE</sub> ・D <sup>+</sup> )	0.008	0.028	0.040
その他の出力範囲		0.048	0.060

(a) 非直線性：使用される D/A 変換素子に固有の性能。  
(b) 相対正確度：非直線性を含む、回路全体の性能。(＝較正可能限度)  
(c) 絶対正確度：相対正確度に較正測定器の正確度を加算した値。(製造時・常温)

【注2】 当表の値にはCPUを含むシステム全体から発生する雑音が含まれていません。  
この雑音は一定DAデータ値をDA変換出力したときの重畳ノイズとして認識され、  
12ビットのとき1LSB(0.025%FS)、16ビットのときは4LSB(0.0062%FS)程度が  
普通です。(数十kHz帯域において)

## 5-4. 付録 (WINDOWS 2000 / XPについて)

### WINDOWS 2000

ボードのインストール： WINDOWS 2000はNT4.0の上位バージョンですが、プラグアンドプレイ機能を持つため、本ボード装着直後のインストール作業時にWINDOWS 2000 対応のインストールディスク（当社製 /vr 2.00 以上）が必要です。作業手順は本書 1 - 5 項、または本ボードに同梱の作業説明書に従ってください。

ソフトウェアサポート： 汎用のI/Oドライバ、および本ボード専用の関数DLLが追加されています。 前者については4 - 2 項、後者については第6章をごらんください。

### WINDOWS - XP

ボードのインストールからドライバ、ハンドラ関数DLLまで、添付のWINDOWS 2000用ソフトウェアがそのまま御利用いただけます。

## 第6章. WINDOWSハンドラ

追加：1999年10月

本DAボード/HDA-770PCIをVC、VC++、VB等で簡単に使用することのできるWINDOWS 9x・ME・NT・2000・XP用ハンドラ関数DLL(+ドライバ)です。

本ボードの基本機能が関数化されており、さらにソフト的タイマによる自動出力：波形出力や外部イベントに同期した出力機能もサポートされています。

ユーザは自身の記述するメインルーチン中から呼び出して使用することができます。

## 6-1. システム構成・ソフトウェア構造

パソコン本体：IBM PC/AT互換機(含む98NX機)、メモリ16MB以上

OS/コパイラ：WINDOWS 9x、ME、NT、2000、XP / 32ビット専用。

添付サンプル：Visual-C、C++(5.0)、Visual-Basic(5.0)  
Borland-C(5.0)、Delphi(3.0)

対応DAボード：TDA-772PCI

チャンネル数：最大16(ボード数は8枚/マスタスレーブ接続)

内部クロック：1ms～3600s周期  
(パソコン内)

誤差：最大+20μs / 誤差は累積しない。  
WINDOWS 9x、Pentium 400MHz のとき。

外部クロック：最高追従速度200μs(WINDOWS 9x、Pentium 400MHz のとき。)

サンプリング速度：=クロック(上記)、CPU速度に多少依存。

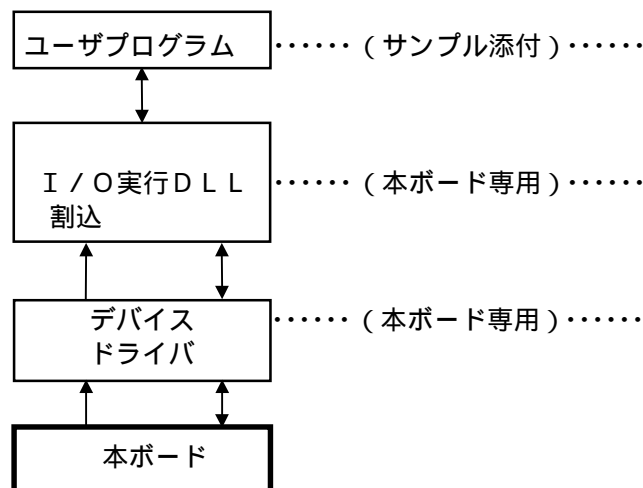
出力データ点数：拡張メモリ空容量(1語=2バイト)

割り込み：不使用。

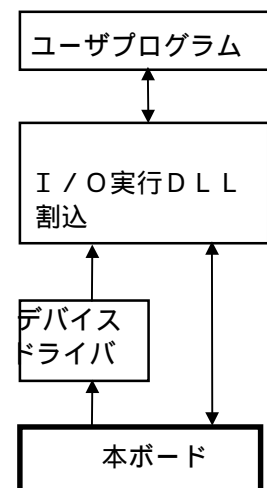
LabVIEW等で利用可能(サンプルv iソース参照)

図6-1. ソフトウェア構造

(WINDOWS NT/2000/XP の場合)



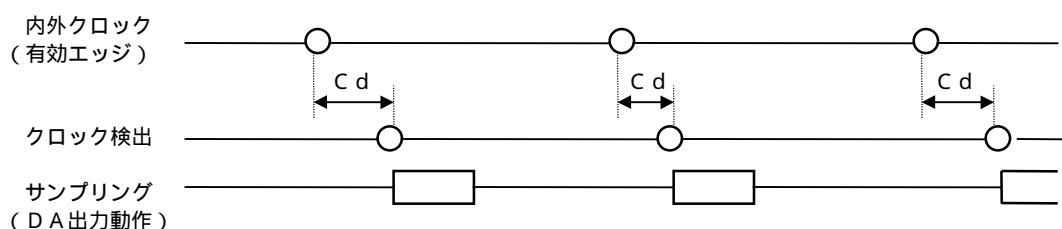
(WINDOWS9x・ME の場合)



## サンプリングの様子

内部クロック、外部クロック、共に本ハンドラはソフト的に検出します。そのため（WINDOWSシステムの性格上）特に排他的なソフトやデバイスが存在しない状態でも検出遅れ $C_d$ があり、これは（pentium/400MHz のとき）最大 + 20  $\mu$ s 程度です。  
なお、この遅れ誤差は累積しません。

図 6-1 C. クロック / サンプリング (DA 出力動作)



【注1】  $C_d = 0 \sim 20 \mu s$  (at pentium/400MHz)

【注2】 DA出力動作時間は数 $\mu s \sim 10 \mu s / ch$  (at pentium/400MHz)

【注3】 : 同期モード時の (全チャンネル同時) 更新タイミング

## 6-2. 使用準備

## ハードウェアの準備

本ハンドラは最大 8 枚の TDA-772PCI を運転することができます。各ボードの設定、接続は以下のとおりです。

1 枚だけ使用するときにはボード上のスイッチ SW-BN の設定を出荷時の値 (= 0) とする。

複数枚使用するときには、2 枚目以降 (スレーブ各機) ボード上のスイッチ SW-BN の設定を 1、2、3、…… 7 とし、1 枚目 (マスタボード) の UPD-OUT 出力を各スレーブボードの UPD-IN 入力に接続する。(3-8 項参照)

複数枚使用するときのチャンネル番号はマスタ (スイッチ SW-BN = 0) 機が 0、1、以下スレーブ (SW-BN = 1) 機が 2、3 の順となる。

同期更新モードの場合、全てのボードは TDA-772PCI-2 (2ch 機) として扱う。使用チャンネル数が奇数のときは最終スレーブ機の最終チャンネル (最大番号チャンネル) が遊びとなるが、このチャンネルにもダミーデータを書き込む操作が必要となる。

(この場合、最終スレーブ機に 1ch 機を使用することもできる。)

マスタ機の外部入力 (UPD-IN) は波形出力モード時の外部クロック、または外部トリガ (スタート) として使用することもできる。【注】いずれか一機能。

各ボードの出力範囲は任意で、一致している必要はない。

### ソフトウェアの準備

まず第4章(4-1項)にしたがって本ボード関連ファイルをインストールします。  
次に本関数DLL / 専用ドライバを所定のフォルダに移すインストール作業は以下のとおり。

¥	Hnd__95      Tda772 (WINDOWS 9x・ME用ハンドラ)	Dll: ハンドラDLL - Vxd: デバイスドライバ - Vb5: Visual Basic (5.0) 用サンプル - Vc5: Visual C (5.0) 用サンプル - Bc5: Borland C (5.0) 用サンプル - Vc5_cpp: Visual C++ (5.0) 用サンプル - Delphi3: Delphi (3.0) 用サンプル
	Hnd__NT      Tda772 (WINDOWS-NT用ハンドラ)	D57xreg: デバイスドライバ設定ユーティリティ Dll: ハンドラDLL - Sys: デバイスドライバ
	(WINDOWS 2000・XP用ハンドラ) Hnd__2K      Tda772	各サンプルは上記WINDOWS 9x・MEハンドラ同様

#### WINDOWS 95・98・MEの場合

- (1) ¥WINDOWS フォルダ に “ H 7 7 2 \_ 9 5 . D L L ” をコピーする。
- (2) ¥WINDOWS¥SYSTEM フォルダに “ D 7 7 2 \_ 9 5 . V X D ” をコピーする。

#### WINDOWS NT (4.0) の場合 / Administrator レベルで行う /

- (1) ¥WINDOWS¥SYSTEM32 フォルダに “ H 7 7 2 \_ N T . D L L ” をコピーする。
- (2) ¥WINDOWS¥SYSTEM32¥DRIVERS フォルダに “ D 7 7 2 \_ N T 0 . S Y S ”  
～ “ D 7 7 2 \_ N T 7 . S Y S ” をコピーする。
- (3) 任意のフォルダにユーティリティ “ D 7 7 2 R e g . e x e ” をコピーする。

【注】ファイル名中の末尾文字 はバージョン変更時に追加(初期は無し)されます。

#### 《デバイスドライバの設定 / リソースの確認》

デバイスドライバの設定 / リソースの確認ユーティリティ (D 7 7 2 R e g) を起動すると、本ボードの ( プラグアンドプレイで設定された ) I / O アドレス・割り込みレベル情報をレジストリに書き込み、( 確認のため ) 表示し、さらに本専用ドライバに同情報を設定します。

このとき同時に、以後の本専用ドライバ起動方法 ( 自動 / 手動 / 禁止 ) を選択設定します。

#### WINDOWS 2000・XP の場合 / Administrator レベルで行う /

- (1) 当社製 P C I ボード ( 複数可能 ) に共通使用できる W I N D O W S 2 0 0 0 用の W D M ドライバ “ DMS\_PCI.SYS ” はボードインストール時に ( ボードインストールディスクから ) 自動的にインストールされます。

インストール先: ¥WINDOWS¥SYSTEM32¥DRIVERS フォルダ

この W D M ドライバは当社製の全 P C I ボード ( 複数可 ) から共通に使用できる汎用品です。すなわち各個別 P C I ボード専用の関数 D L L を用意すれば、当 W D M ドライバ 1 本で当社製の全 P C I ボードを動作させることができます。

- (2) 本関数 D L L もボードインストール時に所定の ¥WINDOWS¥SYSTEM32 にコピーされているので、即サンプル ( ¥MSCIENCE¥HND\_2K¥Tda772 以下 ) を使用できます。

## 6-3. ユーザプログラム記述

御自身の記述したメインプログラムから本ハンドラDLL（+ドライバ）を使用します。

テストには付属のサンプルプログラムを御利用ください。前6-2項に従ってインストールしておきます。本ボードの操作は通常以下の手順となります。具体的なコーディングについてはサンプル・ソースを御覧ください。

- (1) 初期化を行う。 【DA\_Open\_DASys( )】  
 ここではボードの存在やPnPで設定されたリソース本ハンドラが認識すると同時にボードリセット、その他、ハンドラ内の参照テーブルやデータバッファを初期化する。
- (2) パソコン本体メモリ上にデータバッファを確保し、出力すべきDAデータを書き込む。  
 サンプリング・モード関連 【DA\_Set\_SampMode( )】
- (3) サンプリング(DA出力)条件を設定する。  
     トリガ関連                   【DA\_Set\_Trigger( )】  
     出力範囲の設定           【DA\_Set\_Clock( )】  
     クロックの設定           【DA\_Set\_Range( )】  
     波形データの設定       【DA\_Write\_DllBuf( )】
- (4) サンプリング開始。           【DA\_Start\_Samp( )】
- (5) 動作状態(ステータス)取得。   【DA\_Get\_Status( )】
- (6) サンプリング停止。           【DA\_Stop\_Samp( )】

表 6-3. 関数一覧

関数名	機能・動作	主なパラメータ等
【1】DA_Open_DASys	ボード、本ハンドラの初期化	使用ボード数
【2】DA_Reset_Board	全ボードのリセット	
【3】DA_Set_SampMode	DA出力モードの設定	波形バッファ利用の有無
【4】DA_Set_Range	DA出力範囲、分解能、コードの設定	範囲/分解能/コード
【5】DA_Set_Clock	クロック源、クロック値の設定	クロック源/周期/単位/
【6】DA_Write_DLLBuf	対DLL管理バッファへのデータ転送	波形出力データ
【7】DA_Start_Samp	波形出力スタート	スタート条件、繰り返し回数
【8】DA_Out_Prompt	DA即時更新出力	DAデータ
【9】DA_Get_Status	ボード・ステータスの取得	ボード・ステータス
【10】DA_Stop_Samp	波形出力の強制停止	
【11】DA_Close_DASys	本ハンドラの終了	
【12】DA_Out_Aux	汎用デジタル(ラッチ)出力	更新データ
【13】DA_Inp_Aux	汎用デジタル(現在値)入力	
【14】DA_Force_0v	デジタル入力による強制0v出力	指定DAチャンネル
【15】DA_Get_Libver	本ハンドラのバージョン情報取得	

【注】 WINDOWS 2000・XP用に限り、関数名が異なります。

すなわち関数名中、文字“DA”の直後に対象ボード名(3桁番号)が含まれます。

例: DA772\_Open\_DASys

### 重 要

本ハンドラDLLを(LabVIEWなどの)市販・完成アプリケーションから呼出し実行するときはオープン&初期化関数【1】でウインドウハンドルHWNDOwnerにNULLを渡します。この場合、アプリケーション側は本ハンドラからの波形出力終了メッセージを受け取ることができません。



## 6-4. 関数仕様、エラーコード

初期化、動作条件設定、スタート、ストップ、ステータス取得等、各関数は波形出力を実現する基本機能単位となっています。

また、各関数は自身の性格から適切な実行手順があります。（前6-3項・参照）

【注】 WINDOWS 2000・XP用に限り  
各関数名中“DA”の直後に3桁の  
ボード番号が入ります。  
例：DA772\_Open\_DASys

### 【1】本ハンドラのオープン、および初期化

<code>int DA_Open_DASys ( HWND Owner , int num_board )</code>	
<code>num_board</code>	TDA-772PCIボードの使用枚数。
戻り値	正常終了時： 0 エラー時： エラーコード（負の値 / エラーコード表6-4参照）
機能・動作	プラグアンドプレイで設定された（各ボードの）I/Oアドレスを認識すると同時に、ボードリセット、その他、ハンドラ内の参照テーブルやデータバッファ等を初期化する。  当関数実行直後の各ボードは0V出力、また即時更新モード【3】、出力範囲等を設定【4】すれば、いつでも1データ即時更新出力【8】が可能。  波形出力するときは【3】～【6】の条件設定後、【7】でスタート。

### 【2】全ボード（TDA-772PCI）のリセット

<code>int DA_Reset_Board ( void )</code>	
引数	なし。
戻り値	正常終了時： 0 エラー時： エラーコード（負の値 / エラーコード表6-4参照）
機能・動作	全ての制御レジスタは初期状態に戻る。（汎用デジタル出力は変化しない） アナログ出力は0Vに戻る。 なお【1】初期化実行直後には必要ない。（本関数の動作は【1】初期化の一部）

### 【3】DA出力モードの設定

<code>int DA_Set_SampMode ( int ch_num, int upd_mode, int buf_dll )</code>	
<code>ch_num</code>	使用するチャンネル数（1～16）
<code>upd_mode</code>	DA出力時刻モード指定 / 0：単独更新、 1：同期更新（複数チャンネルが一斉に更新）
<code>buf_dll</code>	DA出力更新モード指定 / 0：即時更新出力（マニュアル出力） 1：指定クロックによる波形出力（DLL管理のバッファ出力）
戻り値	正常終了時： 0 エラー時： エラーコード（負の値 / エラーコード表6-4参照）
機能・動作	DA出力タイミング条件を設定する。

## 【4】DA出力範囲、分解能、コードの設定

<pre>int DA_Set_Range(int ch_no, int resol,                  int range, int range_mode, int data_code)</pre>	
ch_no	設定対象DA出力チャンネル番号。/ 整数0～15
resol	DA出力の分解能。/ 0:12bit、1:14bit、2:16bit
range	DA出力範囲。/ 0:0～10V、1:0～5V、2:±10V、3:±5V
range_mode	DA出力範囲モード。/ 0:Aモード、1:Bモード (2-2項参照)
data_code	DA出力データ・コード。/ 0:バイナリ、1:2の補数
戻り値	正常終了時: 0 エラー時: エラーコード(負の値/エラーコード表6-4参照)
機能・動作	チャンネルごとにDA出力範囲、分解能、データコードを設定します。

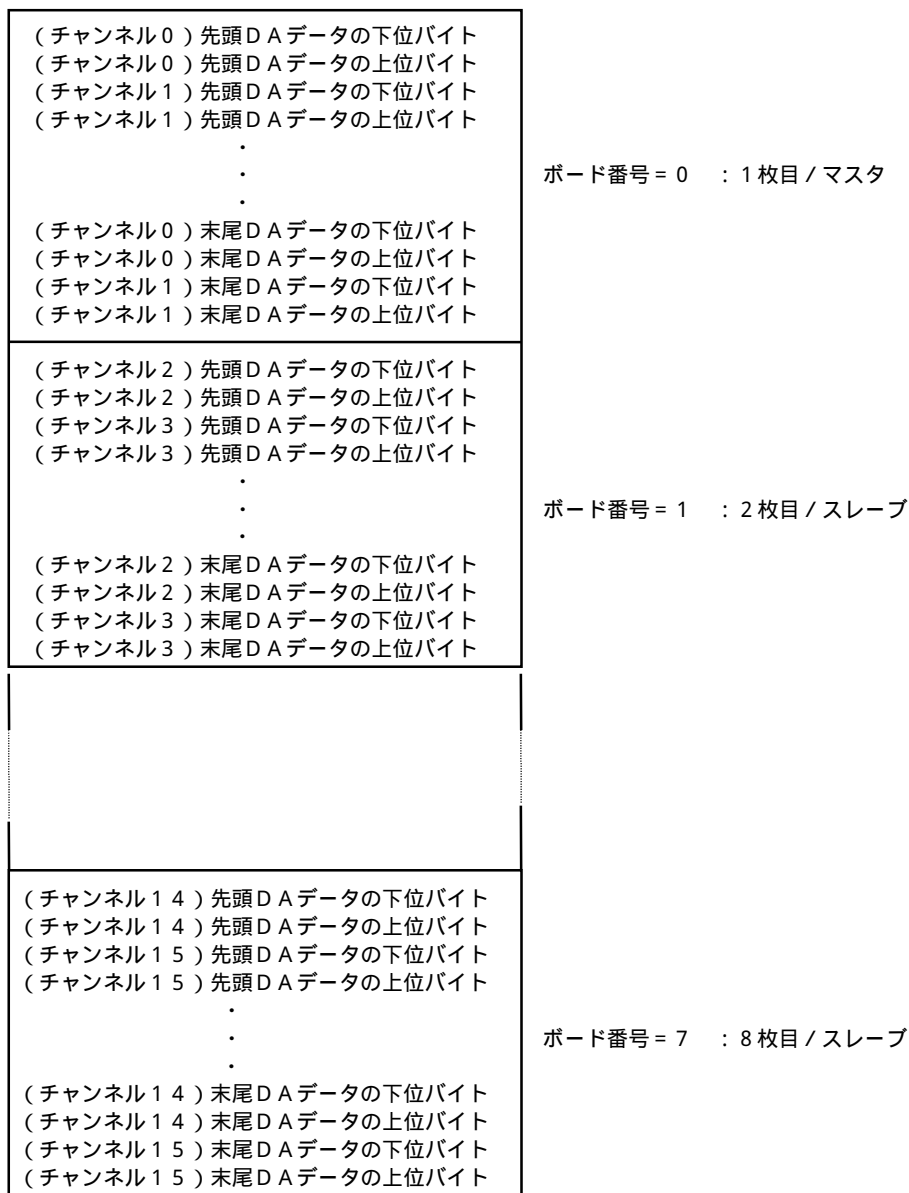
## 【5】サンプリング・クロックの設定(波形出力に使用)

<pre>int DA_Set_Clock(int clk_source,                  int time_unit, int clk_period)</pre>	
clk_source	クロック源。/ 0:ソフトタイマ、1:外部入力(負エッジ)、2:外部入力(正エッジ)
time_unit	指定クロック周期値の単位。/ 0:s、1:ms
clk_period	指定クロック周期値。/ int整数
戻り値	正常終了時: 0 エラー時: エラーコード(負の値/エラーコード表6-4参照)
機能・動作	波形出力のときに使用するクロックを指定します。 外部入力(UPD-IN)をクロック源に指定したときのtime_unit, clk_periodは無効、また外部トリガは使用できません。/ 参照: 【7】サンプリング・スタート

## 【6】(本DLLの管理するバッファへの)波形データ転送

<pre>int DA_Write_DLLBuf(int board_no, int num_samp,                     int num_data, WORD *bufptr)</pre>	
board_no	対象ボード番号。/ 整数0～7
num_samp	各チャンネルの出力データ数(全チャンネル共通)。/
num_data	今回転送するデータ数。(図6-4の構造で、各ボード用領域の上から順に埋められる。)
bufptr	転送先データバッファのポインタ
戻り値	正常終了時: 0 エラー時: エラーコード(負の値/エラーコード表6-4参照)
機能・動作	波形データ転送は対象ボード(2チャンネル分)単位で行います。 複数ボード使用のときで、奇数チャンネル数使用(最終チャンネルを不使用)のときも使用しない最終チャンネル分のダミーデータが必要です。 ボード1枚のみで1チャンネル(ch.0)のみ使用するときに関し、使用しないチャンネル(ch.1)用のダミーデータは不要です。(図6-4/【注1】【注2】参照)

図 6 - 4 . 波形データ格納バッファ内のデータ配置



【注1】 1チャンネルだけ使用する場合はボード番号0 / マスタのデータ配置（本図の最上段）のチャンネル0の部分だけで、（使用しない）チャンネル1のデータは転送しないこと。

【注2】 複数ボードを使用する場合は、全て2チャンネル機（TDA-772PCI-2）を使用するものとし、奇数チャンネル数使用（最終チャンネルを不使用）の場合は使用しない最終チャンネル用のダミーデータを転送すること。

## 【7】サンプリング（波形出力）スタート

<code>int DA_Start_Samp(int loop_num, int trig)</code>	
<code>loop_num</code> <code>trig</code>	繰り返し回数。 / 0 : 強制停止まで無限、 n : 指定回数（正の整数） トリガ条件。 / 0 : 即スタート、 1 : 外部入力（負エッジ）、 2 : 外部入力（正エッジ）
戻り値	正常終了時 : 0 エラー時 : エラーコード（負の値 / エラーコード表 6 - 4 参照）
機能・動作	本DLLの管理するデータバッファに格納されている各チャンネル波形データを1周期として、指定回数だけ連続出力します。 外部入力（UPD-IN）をトリガ（スタート）に指定する場合はソフトクロックだけで、外部クロックは使用できません。 / 参照 : 【5】サンプリング・クロックの設定。

## 【8】マニュアル（即時更新）DA出力

<code>int DA_Out_Prompt(int ch_no, int upd_prompt)</code>	
<code>ch_no</code> <code>upd_prompt</code>	DA出力チャンネル番号。（0 ~ 15） DA出力データ。 / 整数（digit）
戻り値	正常終了時 : 0 エラー時 : エラーコード（負の値 / エラーコード表 6 - 4 参照）
機能・動作	DA出力時刻モード（【3】参照）が単独の場合は当チャンネルのみが即時更新されます。 DA出力時刻モード（【3】参照）が同期の場合はチャンネル1 ~ 15について当関数で更新データを書き込んでおき、最後にチャンネル0（マスタ機）を書き込むと、全チャンネルが同一時刻に更新出力されます。 本関数は【1】初期化【3】出力モード設定【4】出力範囲設定の後、即実行可能です。

## 【9】ボードステータスの取得

<code>int DA_Get_Status(int board_no, int *busy2, int *status)</code>	
<code>board_no</code>	ステータス取得する対象ボードの番号 / 0 : 1枚目（マスタ）、 1 ~ 7 : スレーブ
<code>busy2</code>	ボード内部のDAデータ転送フラグ。 / 0 : 未転送・転送終了、 1 : 転送中
<code>*status</code>	ボードステータス1。 / 1バイト生データ（3 - 7項）
戻り値	正常終了時 : 0 エラー時 : エラーコード（負の値 / エラーコード表 6 - 4 参照）
機能・動作	本関数は当DLL / ドライバの各関数を使用するうえでは必要ありません。 各関数は内部でボード・ステータスを取得・評価のうえで動作するからです。

## 【10】サンプリング（波形出力）動作の強制停止

<code>int DA_Stop_Samp(void)</code>	
引 数	なし。
戻り値	正常終了時 : 0 エラー時 : エラーコード（負の値 / エラーコード表 6 - 4 参照）
機能・動作	全チャンネル（ボード）のサンプリング動作を強制的に停止します。

## 【11】本ハンドラの終了

<code>int DA_Close_DASys( void )</code>	
引 数	な し。
戻り値	正常終了時： 0 エラー時： エラーコード（負の値/エラーコード表6-4参照）
機能・動作	本ハンドラの終了。 確保したメモリ領域の開放等、開始前の状態に戻す。 【注】DA出力最終状態を維持するためにボードのリセットは行いません。

以上【1】～【11】が主要関数です。 以下の【12】～ は補助的な関数です。

## 【12】汎用デジタル（ラッチ）出力更新

<code>int DA_Out_Aux(int board_no, int out_data)</code>	
<code>board_no</code>	更新設定する対象ボードの番号/0:1枚目（マスタ）、 1～7:スレーブ
<code>out_data</code>	4ビット汎用デジタル（ラッチ）出力データ。
戻り値	正常終了時： 0 エラー時： エラーコード（負の値/エラーコード表6-4参照）
機能・動作	TTLレベルの4ビット汎用デジタル（ラッチ）出力を更新する。 当出力ポートは電源投入（ハードウェア・リセット）時=0Hとなるが、 【1】初期設定、【2】ボード・リセット、【11】ハンドラ終了などを実行しても 変化しない。 // 出力論理はボード上のスイッチSW-PLで切り替え可能。 //

## 【13】汎用デジタル（現在値）入力

<code>int DA_Inp_Aux(int board_no)</code>	
<code>board_no</code>	対象ボードの番号/0:1枚目（マスタ）、 1～7:スレーブ
戻り値	正常終了時： = 4ビット汎用デジタル（現在値）入力データ エラー時： エラーコード（負の値/エラーコード表6-4参照）
機能・動作	TTLレベルの4ビット汎用デジタル入力（現在状態）を読み込む。

## 【14】 デジタル入力による強制0v出力の設定

int DA_Force_0v( int ch_no, int set )	
ch_no	対象DA出力チャンネル( 0 ~ 15 )
set	設定の有無。 / 0 : 非設定、 1 : 設定
戻り値	正常終了時 : 0 エラー時 : エラーコード ( 負の値 / エラーコード表 6 - 4 参照 )
機能・動作	対象チャンネルの所属するボードの汎用デジタル入力ビット0を緊急入力とし、LOWになると強制的に0v出力とする設定。

## 【15】 本ハンドラのバージョン取得

int DA_Get_Libver( int ver )	
ver	0 : 戻り値は ( メジャー・バージョン番号 ) + ( マイナー・バージョン番号 ) 1 : 戻り値は ( メジャー・バージョン番号 ) 2 : 戻り値は ( マイナー・バージョン番号 )
戻り値	正常終了時 : 本ハンドラのバージョン番号 エラー時 : エラーコード ( 負の値 / エラーコード表 6 - 4 参照 )
機能・動作	本ハンドラのバージョン情報を得る 例えばバージョンが 1.01 の場合、本関数を ver = 0 として実行すると戻り値は 0 x 1 0 1 となります。

エラー                    本ハンドラの各関数は実行前後（または実行中）に不適当なパラメータや動作状態を検出するとエラーコードを返してきます。

表 6 - 4 . エラーコード一覧

戻り値	不具合の内容、または因果情報	適用関数、引数、等
- 1	指定枚数のボードを検出できない。（未装着）	DA_Open_DASys
- 2	IDの不一致。	DA_Open_DASys
- 3	ドライバファイルが見つからない。	DA_Open_DASys
- 4	ドライバファイルのバージョンが違う。	DA_Open_DASys
- 5	初期化（DA_Open_DASys）が未実行。	DA_Open_DASys
- 6	スタート関数【10】実行前の条件設定不足。	DA_Open_DASys
- 10	ボード数、またはチャンネル数指定エラー。	num_board, ch_num
- 11	DA出力時刻モードの指定が不適当。	upd_mode
- 12	DA出力更新モードの指定が不適当。	buf_dll
- 13	分解能の指定が不適当。	resol
- 14	DA出力範囲の指定が不適当。	range
- 15	DA出力範囲モードの指定が不適当。	range_mode
- 16	データコードの指定が不適当。	data_code
- 17	クロック源の指定が不適当。	clk_source
- 18	クロック周期単位の指定が不適当。	time_unit
- 19	クロック周期値の指定が不適当。	clk_period
- 20	DA出力データ数の指定値が不適当。	num_samp
- 21	転送データ点数の指定値が不適当。	num_data
- 22	繰り返し回数の指定が不適当。	loop_num
- 23	トリガ条件の指定が不適当。	trig
- 24	デジタル入力による強制0V出力の指定が不適当。	Set
- 30	DA出力（サンプリング）中。	
- 31	サンプリングは停止している。	
- 32	マニュアル出力に設定済み。	DA_Write_DllBuf
- 33	メモリ確保エラー	同上。
- 34	出力数を超えるデータの書き込み。	同上。
- 35	外部入力（UPD-IN）はクロックとして指定済み。	DA_Start_Samp

# マイクロサイエンス（株）行

FAX：03（3301）5593

## Q & A フォーム

発信： 年 月 日 / 時 分

製品名	TDA-772PCI（ ）-		購入時期	年 月	
ボード上の 設定、 使用状況					
その他					
I/O、 周辺状況	同時使用の 他ボード		I/Oアドレス 割り込み、等		
本体 システム	パソコン本体		拡張BOX		
	本体メモリ				
	OS	DOS（ ） WIN（ ）			
ソフト	言語		コンパイラ	（ v r ）	
	プログラム名				
（動作状況）					

《60分以内に応答のないときはお叱りください。》TEL：03（3396）8377

御使用者			（所属部・課）
団体名			
TEL			（所在地）
FAX			