

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“S e t u p . e x e”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 : ¥ I N S T A L L ¥ P C I ¥ D A ¥ H D A 7 7 0 を開く。

(2) “S e t u p . E X E”を実行（ダブルクリック）する。

当操作以下によりHDA-770PCI 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:¥WINDOVS>CD¥：【ENTER】

C:¥>CD□D: ¥ I N S T A L L ¥ P C I ¥ D A ¥ H D A 7 7 0 【ENTER】

C:¥>D: I N S T A L L □ D : □ C : 【ENTER】

各プログラムグループ（C, B A S I C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】= y e s , 【N】= n o , 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 = C O M M A N D . C O M が C ドライブの¥にある場合、
> S E T □ C O M S P E C = C : ¥ C O M M A N D . C O M 【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 —	WIN95 —	CF9050DP.COM : コンフィギュレーション (95用)
(本ボードの設定)		
	—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
		—REGSDRV.EXE : デバイスドライバ登録・削除用
—BOARDTST—	—770QB1.EXE :	本ボードの試運転・動作確認用プログラム
	—770QB1.COM :	英語モードに切り替えた後、EXEを実行する
—SMP770C—	—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
	—INT770.C :	外部割り込み入力時にDA出力動作
	—GPP770.C :	最も簡単な16ch汎用DA出力動作
	—MSV770.C :	複数ボードの同期DA出力動作
—SMP770B—	—770QB1.BAS :	Quick-Basic (4.5) 用サンプル
(BASICサンプル)		
—SMP770VB—	SMP770.VBP :	プロジェクト
(Visual)	—DRIVER.FRM :	ドライバ関連フォーム
(BASIC用)	—SMP770.FRM :	メインフォーム
(サンプル)	—MODE.FRM :	モード設定フォーム
	—DRVDDL.BAS :	汎用IO制御ドライバ定義
	—MS_PCI.BAS :	PCIリソース取得定義
	—HDA770.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。 WINDOWS9x・ME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。

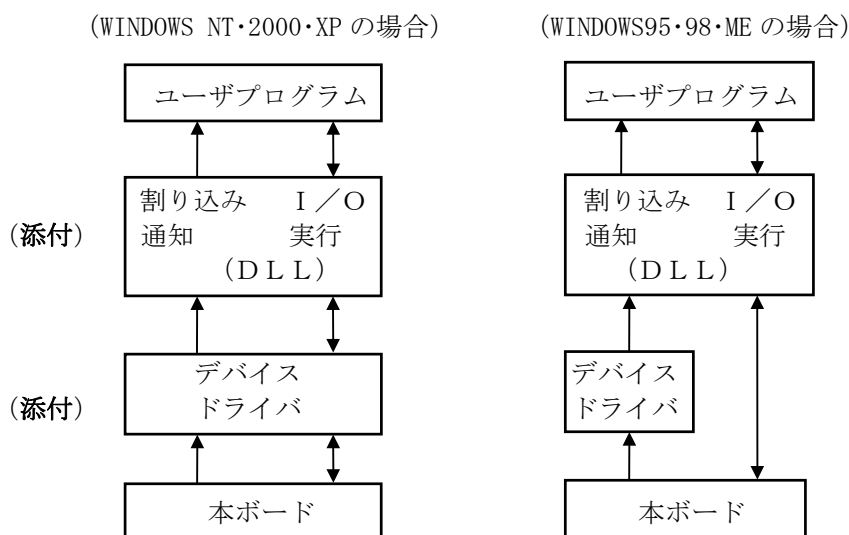
WINDOWS NT用はWinNTフォルダ中にあり、同フォルダ中の専用インストーラで導入してください。 また、WINDOWS 2000用ドライバ (WDM) はボード自体のインストール時に自動インストールされます。 使用法、サンプルなどはWin2Kフォルダ内にあります。

(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) で利用でき (4.0) 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) で利用でき (or XP) 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_P C I . D L L /WINDOWS9x・ME・NT 用 について

ユーザプログラムから本P C I 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D L L）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D L L & ドライバが必要になります。使用手順は、

【1】P C I 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

P C I 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 I D、デバイス I D、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 I D = 1 3 F D H (マイクロサイエンス社 P C I ボード共通)、
- ◆デバイス I D = ▲▲▲ H (HDA-770PCI)。

【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

P C I 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 P C I インターフェース素子 9 0 5 0 (P L X 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 1 0 H : P C I インターフェース素子 9 0 5 0 で使用。
- 1 4 H : 未使用
- 1 8 H : I / O マップレジスタのベースアドレス B A S E 1 (3-2 項)。
- 1 C H : 未使用、2 0 H : 未使用、2 4 H : 未使用

なお、

◆ I/Oマップでは得られた data の下位 2bit 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

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 VendeID, WORD DeviceID, WORD nNum, WORD Flag, WORD _far *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

【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。)

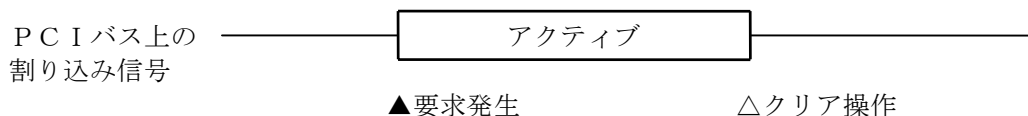
Int ReadPciByte (WORD magic, WORD reg, BYTE _far *data)	
引数	magic : 【1】GetPciDevice で得られた マジック番号、 *data : データ取得先ポインタ、 reg : PCI コンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0 : 成功、 -1 : 失敗

使用方法 : 前記WINDOWS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。

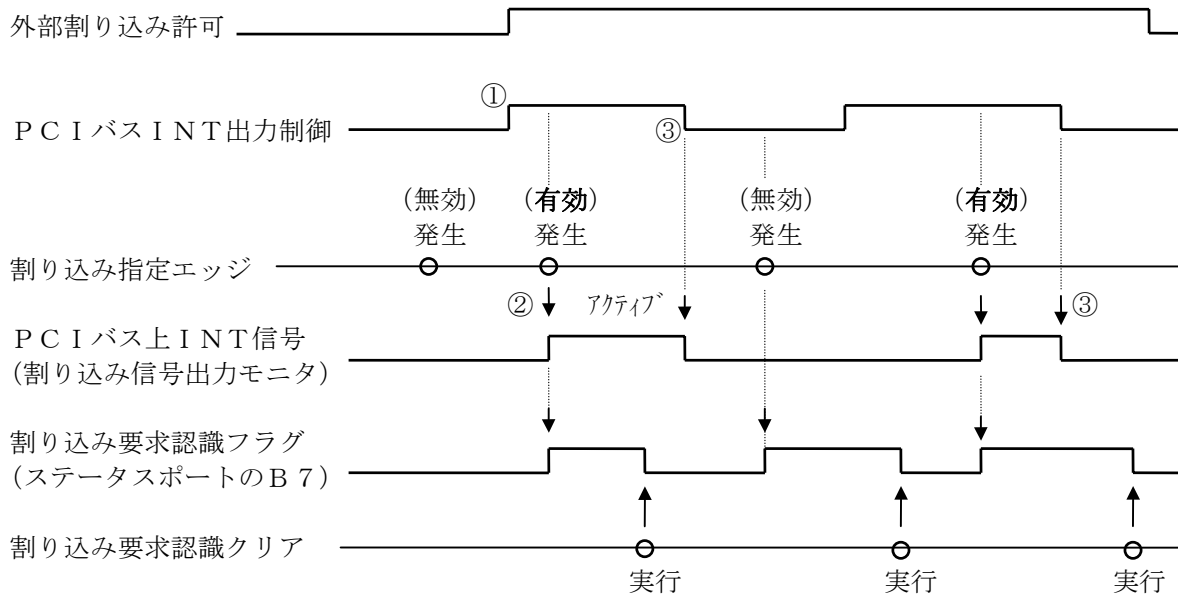
なお、本ボード上のROMに書き込まれているデフォルト（初期）のコンフィギュレーション情報では（プラグアンドプレイの動作時に）割り込みリソースを要求しません。もし要求したときに空きが無く拒否されるとI/Oアドレスの割り当ても受けられず認識不能状態になる恐れがあるからです。割り込みを利用するときはリソースに空きがあることを確認してから添付のコンフィギュレーション・ユーティリティで（割り込みリソースを要求するように）修正してください。【1-5項.参照】

（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

- ① 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【3-8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。
- ② 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。
- ③ 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／3-4項／割り込み信号モニタポートでPCIバス上の割り込み信号ステータスに反映されますが、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？）
- ④ 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4-4 B. 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

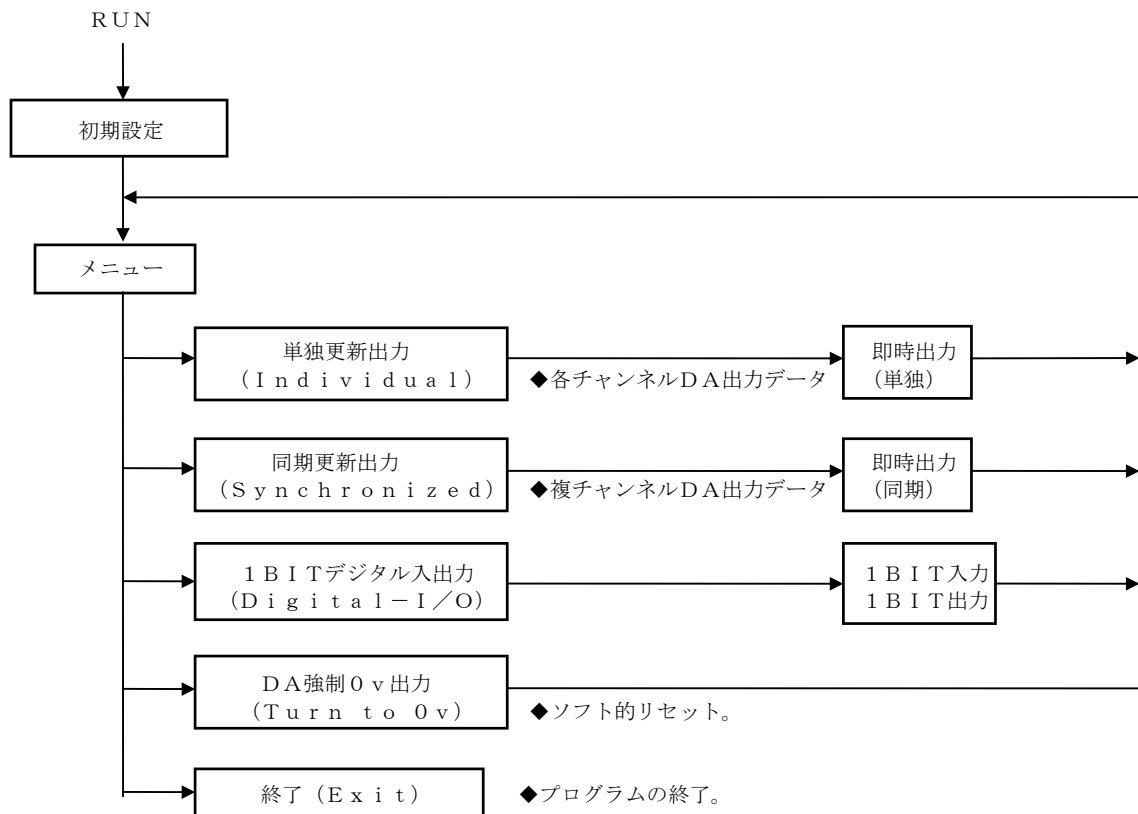
 (割り込み不使用)

前①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認識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) 用サンプルプログラム“770QB1. 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“770QB1. 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. “770QB1. BAS”のフロー概要



4-6. Cのサンプル (MS-DOS用/16ビット・コード)

①単純な16チャンネルDA即時更新出力、②外部割り込み入力によるDA更新出力、および③複数ボード（ここでは2枚=32チャンネル）のDA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4-6A. 汎用16チャンネルDA動作【GPP770.C】

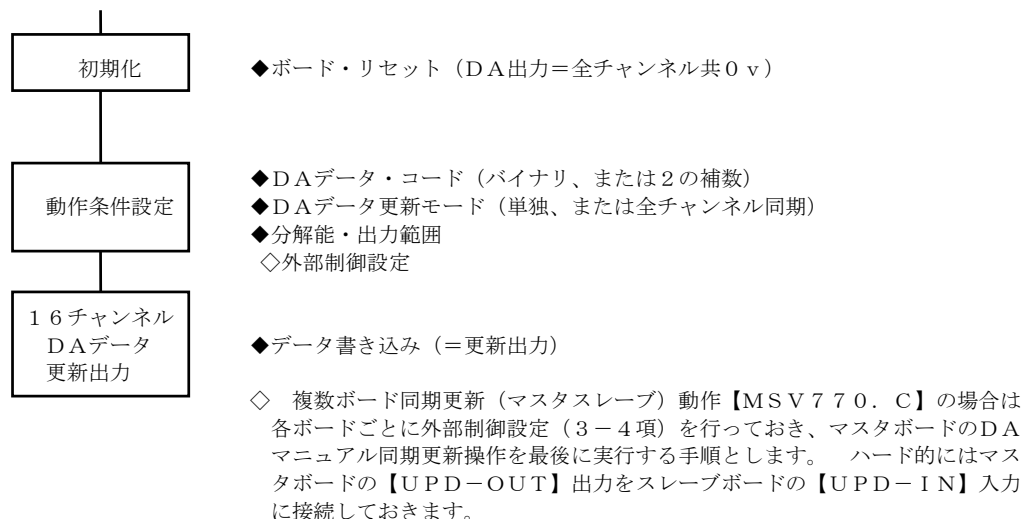
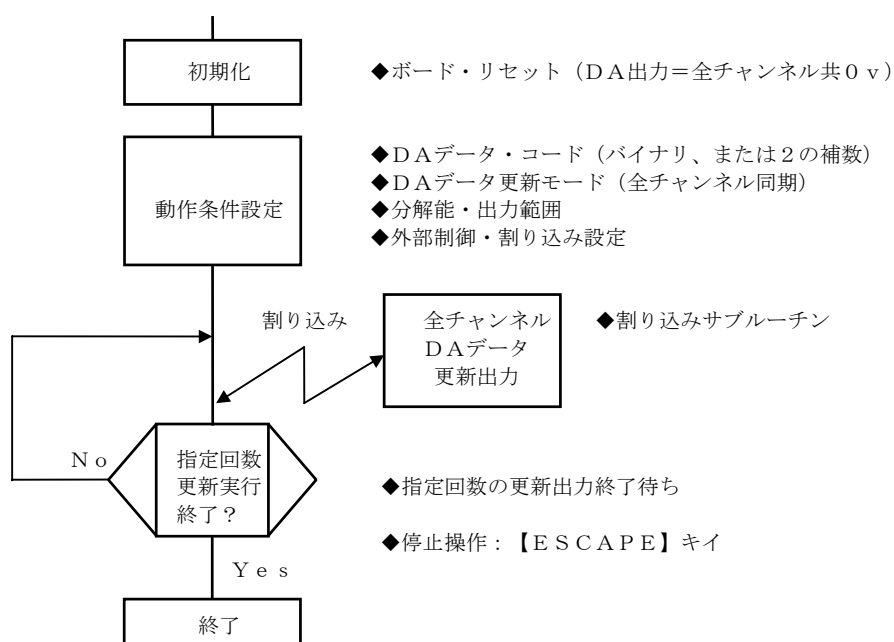


図4-6B. 外部割り込み入力によるDA更新出力【INT770.C】



第6章. WINDOWSハンドラ

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、さらにソフトの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。

【注】自動波形出力機能を外し複数の本ボード（最大16枚：256ch同期出力）に対応した新ハンドラを2006年6月から追加しました。波形出力スレッドが無い分だけソフトが軽くなるので（1枚のみ使用時も）お勧めです。／御利用方法は6-5項にジャンプ／

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/コンパイラ : 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 : HDA-770PCI

◇チャンネル数 : 最大16（本ボード1枚）／追加サポート：複数ボード対応は6-5項／

◇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。

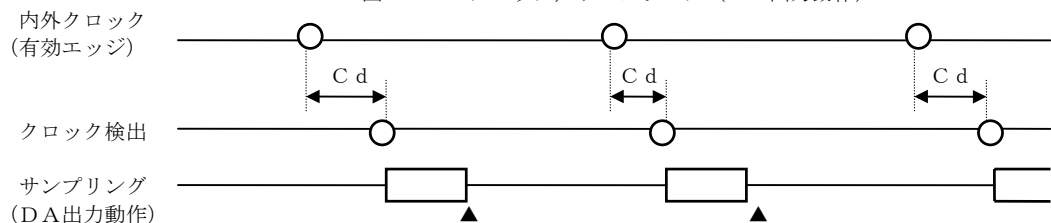
◇出力データ点数 : 拡張メモリ空容量（1語=2バイト）

◇割り込み : 不使用。

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Cdがあり、これは（pentium/400MHz のとき）最大+20μs程度です。
なお、この遅れ誤差は累積しません。

図6-1. クロック／サンプリング（DA出力動作）



【注1】Cd=0～20μs（at pentium/400MHz）

【注2】DA出力動作時間は数μs～10μs/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（HDA-770PCI）1枚／16チャンネルを制御できます。
ボード上の設定・接続は標準出荷状態（1-2項）のとおりです。

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。

◆WINDOWS 95・98・MEの場合

- (1) ¥WINDOWS フォルダ に“H770_95△.DLL”をコピーする。
- (2) ¥WINDOWS¥SYSTEM フォルダに“D770_95△.VXD”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“H770_NT△.DLL”をコピーする。
- (2) ¥WINDOWS¥SYSTEM32¥DRIVERS フォルダに “D770_NT△.SYS”
- (3) 任意のフォルダにユーティリティ “D770Reg.exe” 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（D770Reg）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PCIボード（複数可能）に共通使用できるWINDOWS 2000／XP用のWDMドライバ“DMS_PCI.SYS”はボードインストール時に（添付のCDROM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WDMドライバは当社製の全PCI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PCIボード専用の関数DLLを用意すれば、当WDMドライバ1本で当社製の全PCIボードを動作させることができます。

- (2) 本関数DLL（H770_2k）もボードインストール作業時に所定の ¥WINDOWS¥SYSTEM32 にコピーされているので、即サンプル（¥MSCIENCE¥HND_2K¥Hda770 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。
 - ◆ サンプルング・モード関連 **【DA_Set_SampMode（）】**
- (3) サンプルング（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) サンプルング開始。 **【DA_Start_Samp（）】**
- (5) 動作状態（ステータス）取得。 **【DA_Get_Status（）】**
- (6) サンプルング停止。 **【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	
【2】DA_Reset_Board	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、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】 WINDOWS2000／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：DA770_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 **HWND_Owner**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：DA770_Open_DASys

【1】本ハンドラのオープン、および初期化

int DA_Open_DASys (HWND Owner)	
HWND Owner	ウインドウ・ハンドル。
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	<p>プラグアンドプレイで設定された本ボードの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。</p> <p>◇ 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。</p> <p>◇ 波形出力するときは【3】～【6】の条件設定後、【7】でスタート。</p>

【2】本ボード（HDA-770PCI）のリセット

int DA_Reset_Board (void)	
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	<p>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）</p>

【3】DA出力モードの設定

int DA_Set_SampMode (int ch_num, int upd_mode, int buf_dll)	
ch_num	使用するチャンネル数（1～16）
upd_mode	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）
buf_dll	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

<code>int DA_Set_Range (int ch_no, int range, int range_mode, int data_code)</code>	
<code>ch_no</code>	設定対象DA出力チャンネル番号。／整数0～15
<code>range</code>	DA出力範囲。／ 0 : 0～10v、 1 : 0～5v、 2 : ±10v、 3 : ±5v
<code>data_code</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出力範囲を設定します。 DA出力データ・コードは全チャンネル同一コードを指定してください。 （異なるコードを指定した場合は最後に指定したコードが全チャンネルに適用される。）

【5】サンプリング・クロックの設定（波形出力に使用）

<code>int DA_Set_Clock (int clk_source, int time_unit, int clk_period)</code>	
<code>clk_source</code>	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）
<code>time_unit</code>	指定クロック周期値の単位。／ 0 : s、 1 : ms
<code>clk_period</code>	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 <code>time_unit</code> 、 <code>clk_period</code> 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

<code>int DA_Write_DLLBuf (int num_samp, int num_data, WORD *bufptr)</code>	
<code>num_samp</code>	各チャンネルの出力データ数（＝クロック動作数、全チャンネル共通）。／
<code>num_data</code>	今回転送するデータ数。（図6-4の構造で、領域の上から順に埋められる。）
<code>bufptr</code>	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	出力すべき波形データを用意します。

図6-4. 波形データ格納バッファ内のデータ配置（1出力分）

(チャンネル0) DAデータの下位バイト
(チャンネル0) DAデータの上位バイト
(チャンネル1) DAデータの下位バイト
(チャンネル1) DAデータの上位バイト
⋮
⋮
⋮
⋮
⋮
(チャンネル14) DAデータの下位バイト
(チャンネル14) DAデータの上位バイト
(チャンネル15) DAデータの下位バイト
(チャンネル15) DAデータの上位バイト

【7】 サンプリング（波形出力） スタート

int DA_Start_Samp (int loop_num, int trig)	
loop_num trig	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

int DA_Out_Prompt (int ch_no, int upd_prompt)	
ch_no upd_prompt	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 *busy, int *status)</code>	
<code>busy</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ワード生データ（3-8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

int DA_Out_Aux (int out_data)	
out_data	1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1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

int DA_Inp_Aux (void)	
戻り値	正常終了時： =1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1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 set)	
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	ボードの汎用デジタル入力ビット0を緊急入力とし、 High（or開放）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0x101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	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	チャンネル数指定エラー。	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
－ 14	DA出力範囲の指定が不適当。	range
－ 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

6-5. 複数ボード同期出力用ハンドラ (WINDOWS 2000/XP 専用)

前6-3項・6-4項で説明されたハンドラは本ボード1枚（最大16ch）のみに対応したものです。本項で説明するハンドラは複数の本ボード（最大16枚＝256ch）を同時に制御することができます。実用的には波形出力機能が無い分、CPUの使用効率が上るので軽いアプリケーションを製作できます。（本ボード1枚のみ使用のときもお勧めです。）

ハードウェアの準備

本ハンドラは（HAD-770PCI）最大16枚／256チャンネルを制御できます。

◆ ボード上の設定

ボード上のDIPスイッチSW-BNはマスタ機を＜初期値＝0＞1枚目のスレーブ機を＜1＞、以下＜2＞＜3＞…＜F＞に設定する。

◆ ボード間の接続：

マスタ機の同期更新出力UPD-OUTをスレーブ機の同期更新入力（UPD-IN）に接続しておきます。

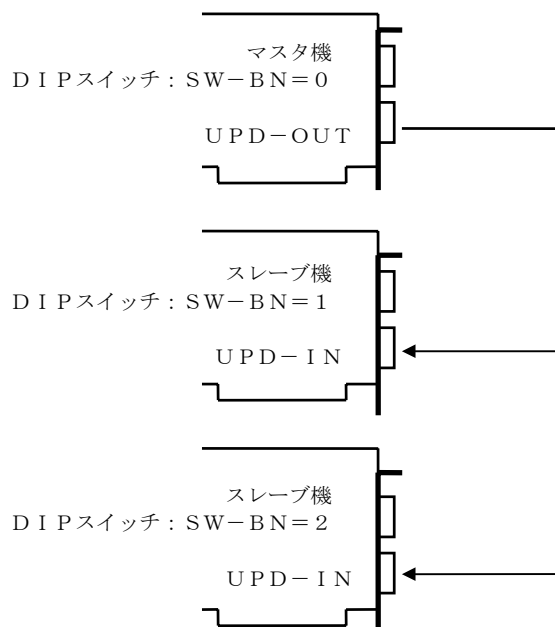


図6-5. マスタスレーブ接続
(3枚使用例)

【注】駆動能力：7枚以内。
これ以上のときは、
適当な順番のスレーブ機の
＜UPD-OUT＞を以降の
＜UPD-IN＞に接続する
ような構成で使用できます。

ソフトウェアの準備

／Administrator レベルで行う／

- (1) 当社製PCIボード（複数可能）に共通使用できるWINDOWS 2000/XP用のWDMドライバ“DMS_PCI.SYS”はボードインストール時に（添付のCDROM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WDMドライバは当社製の全PCI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PCIボード専用の関数DLLを用意すれば当WDMドライバ1本で当社製の全PCIボードを動作させることができます。

- (2) 本関数DLL（M770_2k）を所定の ¥WINDOWS¥SYSTEM32 にコピーします。
動作確認には各言語サンプルを御使用ください。
各ファイルは（¥MSCIENCE¥HND_2K フォルダ以下）に格納されています。

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。前頁のようにインストール、準備しておきます。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。 **【DA770_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) 各チャンネルごとの単独・即時更新動作か、全チャンネル同期更新動作かを選択する。 **【DA770_Set_SampMode（）】**
- (3) 出力範囲、データコードを設定する。 **【DA770_Set_Range（）】**
- (4) 動作状態（ステータス）取得。 **【DA770_Get_Status（）】**
- (5) 単独・即時更新モード時の更新出力 **【DA770_Out_Prompt（）】**
 - ◆同期更新モード時はバッファリング：出力準備。
- (6) 同期更新モード時の更新出力タイミング **【DA770_Sync_Update（）】**
- (7) 汎用デジタル／ラッチ／出力 **【DA770_Out_Aux（）】**
- (8) 汎用デジタル／現在値／入力 **【DA770_Inp_Aux（）】**
- (9) 本ハンドラの終了。 **【DA770_Close_DASys（）】**

表6-5A. 関数一覧

関数名	機能・動作	主なパラメータ等
【1】DA770_Open_DASys	ボード、本ハンドラの初期化	
【2】DA770_Reset_Board	ボードのリセット	
【3】DA770_Set_SampMode	DA出力モードの設定	単独更新／同期更新
【4】DA770_Set_Range	DA出力範囲、コードの設定	ch／範囲／コード
【5】DA770_Out_Prompt	DA即時更新出力	DAデータ
【6】DA770_Get_Status	ボード・ステータスの取得	ボード・ステータス
【7】DA770_Close_DASys	本ハンドラの終了	
【8】DA770_Out_Aux	汎用デジタル（ラッチ）出力	更新データ
【9】DA770_Inp_Aux	汎用デジタル（現在値）入力	
【10】DA770_Force_0v	デジタル入力による強制0v出力	指定ボード
【11】DA770_Sync_Update	全チャンネル同期更新命令	
【12】DA770_Get_Libver	本ハンドラのバージョン情報取得	

関数仕様・エラーコード

【1】本ハンドラのオープン、および初期化

int DA770_Open_DASys (int num_board)	
num_board	使用する本ボード（HDA-770PCI）の枚数：1～16 ◇ボード上のDIPスイッチSW-BNの設定／前々ページ図6-5参照／ 1枚目のボード（マスタボード）を0に、以下2枚目以降のボード（スレーブボード）を1、2、3、、、Fと順に設定する。
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6-5B参照）
機能・動作	プラグアンドプレイで設定された本ボードの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初期状態で、0v出力、即時更新モード【3】、また出力範囲は0～10v【4】となっている。

【2】本ボード（HDA-770PCI）のリセット

／全ボードについて／

int DA770_Reset_Board (void)	
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6-5B参照）
機能・動作	全ての制御レジスタは初期状態に戻る。（ただし汎用デジタル出力のみは変化しない） 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

／全ボードについて／

int DA770_Set_SampMode (int upd_mode)	
upd_mode	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）
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6-5B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、コードの設定

／各ボードの各チャンネルごとに／

<code>int DA770_Set_Range (int board_no, int ch_no, int range, int data_code)</code>	
<code>board_no</code>	設定対象ボード番号。／0～15
<code>ch_no</code>	設定対象DA出力チャンネル番号。／0～15
<code>range</code>	DA出力範囲。／ 0 : 0～10v、 1 : 0～5v、 2 : ±10v、 3 : ±5v
<code>data_code</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6-5B参照)
機能・動作	各ボードの各チャンネルごとにDA出力範囲を設定します。 DA出力データ・コードは全ボード全チャンネルに同一コードを指定してください。 (異なるコードを指定すると最後に指定したコードが全ボード全チャンネルに適用される。)

【5】DAデータ出力

／各ボードの各チャンネルごとに／

<code>int DA770_Out_Prompt (int board_no, int ch_no, int upd_prompt)</code>	
<code>board_no</code>	出力対象ボード番号。／0～15
<code>ch_no</code>	DA出力チャンネル番号。／0～15
<code>upd_prompt</code>	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6-5B参照)
機能・動作	DA出力時刻モード (【3】参照) が単独の場合は指定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) が同期の場合は各ボード各チャンネルについて当関数で更新データを書込んでおき、最後に【11】全ch同期更新命令を実行する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。

【6】ボードステータスの取得

／マスタボードのみ／

<code>int DA770_Get_Status (int *busy, int *status)</code>	
<code>busy</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ワード生データ (3-8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6-5B参照)
機能・動作	マスタボード (1枚目ボード) のステータス取得。 なお、前【5】DAデータ出力の実行時には同関数内部で各ボードの<busy=0>を確認しているため、当関数を使用するの事前チェックは無用です。

【7】本ハンドラの終了

int DA770_Close_DASys (void)	
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6-5B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。

【8】汎用デジタル（ラッチ）出力更新

／各ボードごとに／

int DA_Out_Aux (int out_data)	
board_no	出力対象ボード番号。／0～15
out_data	1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6-5B参照）
機能・動作	TTLレベルの1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、 【2】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。//

【9】汎用デジタル（現在値）入力

／各ボードごとに／

int DA_Inp_Aux (void)	
board_no	入力対象ボード番号。／0～15
戻り値	正常終了時： =1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6-5B参照）
機能・動作	TTLレベルの1ビット汎用デジタル入力（現在状態）を読み込む。

【10】デジタル入力による強制0v出力の設定

／各ボードごとに／

int DA_Force_0v (int set)	
board_no	設定対象ボード番号。／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6-5B参照）
機能・動作	ボードの汎用デジタル入力ビット0を緊急入力とし、 High（or開放）になるとDA出力を強制的に0vとする設定。

【11】全チャンネル同期更新命令

int DA770_Sync_Update (void)	
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6-5B参照）
機能・動作	同期更新モードのときは、 本関数の実行により全ボード全チャンネルのDA出力が同時更新される。

【12】本ハンドラのバージョン取得

int DA770_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6-5B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0x101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6-5B. 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	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	ボード番号、またはチャンネル数指定エラー。	num_board、board_no、ch_no
－ 11	DA出力時刻モードの指定が不適当。	upd_mode
－ 12	DA出力範囲の指定が不適当。	range
－ 13	データコードの指定が不適当。	data_code
－ 14	デジタル入力による強制0v出力設定エラー	set
－ 15	同期更新モードではない。	DA770_Sync_Update

第7章. 保守・その他

7-1. 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：5K Ω 以上、1000pF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1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7－1参照）

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。

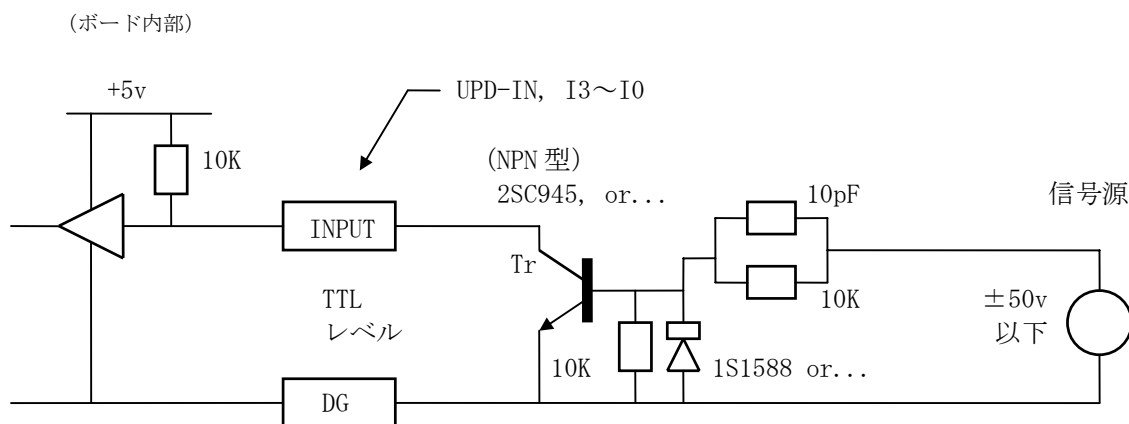
QAリクエスト時には当プログラムの実行結果をレポートしてください。

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：-0.6V】【正側：+7V】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【-0.6V】が低いことに注意してください。
- ◆ +5V以上に振れるロジック信号も接続できません。12V～24V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±15V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ウンド・レベルに差があるときも危険です。（テストで測定可能）

図7-1. 【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。

7-2. 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。 2006年6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。

7-3. 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 ==

- ① 本ボード上の諸設定は出荷時の状態（1-2項）とします。
- ② 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）
- ③ 図7-3のように、本ボードの基準電圧出力テスト端子（TP+/TPG）をデジタル電圧計に接続します。
- ④ 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7-3. 再調整用の測定機接続



- ⑤ 本ボード上の基準電圧出力テスト端子をデジタル電圧計で測定・モニタしながら、
Aモードの場合は10.24v、
Bモードの場合は10.00vとなるようにボード上のトリマTMGを調整します。
- ⑥ 調整は上記⑤の作業のみ、以下は確認作業です。
電源を投入し、MS-DOSシステムを立ち上げます。確認に使用するプログラムは試運転でも使用した“770QB1”です。>770QB1【ENTER】でプログラムが走り始めます。（WINDOWS95・98のDOS窓でも動作可能、NTのDOS窓は不可）
5分程度のウォームアップ後、以下の調整作業を行います。
- ⑦ メニューからDA出力（単独、または同期更新）を選択し、表2-2D、Eの目標値と比較します。確認はユーザシステムに都合のよい出力範囲で実施してください。なお、本機の製造時は**Aモード**で調整されています。

得られる正確度

通常、パソコン用のDAボードでは製造・調整環境と実機を含む現場環境が一致しませんから、現場での絶対正確度を保証することはできませんが、DAボード自体の性能を規定する相対正確度（＝校正可能限度）と製造・調整環境で使用する測定器で決まる絶対正確度（製造時・常温）は下表のとおりです。

表 7-3 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 (Aモード)	0.004	0.058	0.07
その他の出力範囲 (Bモード)		0.078	0.09

定義

- (a) 非直線性 : 使用されるDA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。（＝校正可能限度）
 (c) 絶対正確度 : 相対正確度に校正測定器の正確度を加算した値。（製造時・常温）

【注1】 当製品は正確度0.012%の測定器を使用し、常温で最終調整を行っています。
 当表に表示した相対正確度と絶対正確度の差はこれによるものです。
 なお、周囲温度の変化が大きいときは温度ドリフト (typ. 20 ppm/°C) による誤差も加算されます。 また、経年変化のデータや保証はありません。

【注2】 本機のアナログ調整は全チャンネル共通のゲイン（基準電圧）だけで、各チャンネル個別のゲイン、およびオフセット調整はありません。 ここで定義されている正確度はゲイン、オフセット誤差を含めたものです。 特定のチャンネルに最適化した再調整を行うと、他のチャンネルの正確度が仕様を超えて悪化（表7-3 Aに記す値の約2倍）することがありますので御注意ください。

【注3】 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) 程度が普通です。（数十KHz帯域において）

7-4. 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。

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	HDA-770PCI		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）8362

御使用者		(所属部・課)
団体名		
TEL		(所在地)
FAX		