

第4章. ソフトウェア

《注1》WINDOWS 98/ME対応： 特に断りのない限り、WINDOWS 95用のソフトがそのまま使用できます。

《注2》WINDOWS XP/VISTA対応： 特に断りのない限り、WINDOWS 2000用のソフトがそのまま使用できます。

4-1. ソフトウェアのインストール (ボードのインストール後に行います。)

本製品用のソフトウェアは3.5インチ(1.44MB)FD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の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。

重要な変更について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。

- ◆ADデータ収集： (アプリケーション) WINDOWS 2000/XP/VISTA版LabDAQ-AQです。
ADサンプリング/リアルタイム波形表示/データファイル保存等、
すぐ実用になる基本機能を備えています。(信号処理機能なし版)
当社ホームページ (www.microscience.co.jp) から無償ダウンロード。
- ◆市販ソフト対応： 表計算 (EXCEL等) 向けのCSV形式、
および波形解析ソフトDADiSP向けの専用形式ファイルを上記のAD
データ収集ソフトで作成・保存できます。
- ◆ハンドラDLL： (WIN32ライブラリ) ADボード制御の基本機能を関数化、サンプリング条件設定から実行、
配列にデータを得るまで、ユーザ記述のアプリケーションから呼び出す
だけです。VB, C, C++, Delphi, C++Builder の使用例を添付。
【専用マニュアル：第6章、第8章】
- ◆サンプルソフト： (DOS版/ソース) MSDOS用のADボード制御プログラムを手造りするユーザ向けの
操作例です。C、およびQuick-Basicで例示しています。
ポーリング、割り込み等の使用方法を学習することができます。
- ◆その他： (for WINDOWS) WINDOWS 9x/ME/NT/2000/XP/VISTA で単純にI/O読み
書きを実行するための汎用ドライバ、およびDLLが添付されています。
基本的には当DLLを使用してボード上の各レジスタを読み書きする
ことでユーザ独自の関数プログラミングが可能です。
(当部分は圧縮されておらず、生のまま格納されています。)
- ◆LabVIEW： VIサンプルソースを添付ディスク内、または当社ホームページから
ダウンロードすることができます。
- ◆ActiveX： オプション (松山アドバンス社製：AXADM-68X ¥16,000)
- ◆Linux： オプション (ナリタ社製：Kuziraシリーズ) 各¥36,000

= ソフトウェアのインストール作業 = (1-4項: ボード自体のインストール後に行う。)

添付の(2005年2月版以降の)CDROMを使用します。

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) WINDOWS付属のエクスプローラで、
D:¥INSTALL¥PCI¥AD¥ADM681を開く。

(2) “Setup.EXE”を実行(ダブルクリック)する。

当操作以下により本ボード関連プログラムが
次頁: 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2月版以前の旧CDROMを利用する場合>

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】          (ADM688)
C:¥>CD□D:¥INSTALL¥PCI¥AD¥ADM681【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、または設定しておいてください。

= 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場合のディレクトリ構造は図4-1のようになります。

【注】 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、またWINDOWS2000/XP用はWin2Kフォルダにあります。

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

図4-1. インストール後のディレクトリ

★凡例 MS-C: MICROSOFT-Cの略

T-C : TURBO-Cの略

B-C : BORLAND-Cの略

¥
 |
 M S C I E N C E

■■■：対応ボード番号

```

— UTILITY — WIN95 — CF9050DP.COM : 設定変更ユーティリティ (95用)
(本ボードの設定) |
                    | -WINNT — CF9050NT.EXE : 設定変更ユーティリティ (NT用)
                    |   -CF9050NT.sys : 上記NT用デバイスドライバ
                    |   -REGSTDV.EXE : ドライバの登録ユーティリティ

```

— BOARDTST — ■■■ QB2. EXE : 本ボードの試運転・動作確認用プログラム
| ■■■ QB2. COM : 英語モードに切り替えた後、EXEを実行する

```

— HND■■■■C — INCLUDE — SMPEXT. H : ハンドラ用・共通ヘッダファイル
(DOS用ハンドラ) |

```

- LIB --- ■■■ TS. LIB : T-C、B-C用スモールモデル
- | --- ■■■ TL. LIB : T-C、B-C用ラージモデル
- | --- ■■■ MS. LIB : MS-C用スモールモデル
- | --- ■■■ ML. LIB : MS-C用ラージモデル

— SMP ■■■ — SAMPLE □. C

— SMP ■■■ C —	— MICROSOFT. H	: MS-C用ヘッダ
(Cの各種サンプル)	— BORLAND. H	: TURBO-C, BORLAND-C用ヘッダ
	— ADM ■■■. H	: 共通ヘッダ
	— POL ■■■ E 1. C	: ポーリング動作例 (エンプティ解消フラグ使用)
	— BLK ■■■ H 1. C	: ポーリング動作例 (ハーフフル・フラグ使用)
	— INT ■■■ E 1. C	: 割り込み動作例 (エンプティ解消フラグ使用)
	— P C I. C	: P C I 関連ルーチン

— SMP ■■■ B — — — ■■■ QB 1. BAS : 内部クロックによる連続サンプリング例
 (Q-B a s i c) | — ■■■ QB 2. BAS : マニュアル・サンプリング例
 (試運転・動作確認用プログラムのソース)

```
— Hnd_95 — Adm■■■ — Dll : ハンドラDLL  

(WINDOWS 9x・ME用ハンドラ) | - 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 — Adm■■■■ — D■■■■reg : デバイスドライバ設定ユーティリティ
 (WINDOWS-NT用ハンドラ)

- Dll : ハンドラDLL
- Sys : デバイスドライバ

— H n d _ 2 K — A d m ■ ■ ■ ————— D 1 1 : ハンドラDLL
(W I N D O W S 2 0 0 0 / X P / V I S T A 用 ハンドラ)

— GL_DOS —	MSPCID.H	:(DOS版)リソース取得ライブラリ・ヘッダ
(DOS用)	— 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
(9x・NT用)	— MS_PCI.DLL	:	リソース取得ライブラリDLL
			(Win9x・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) DLL関数定義モジュール

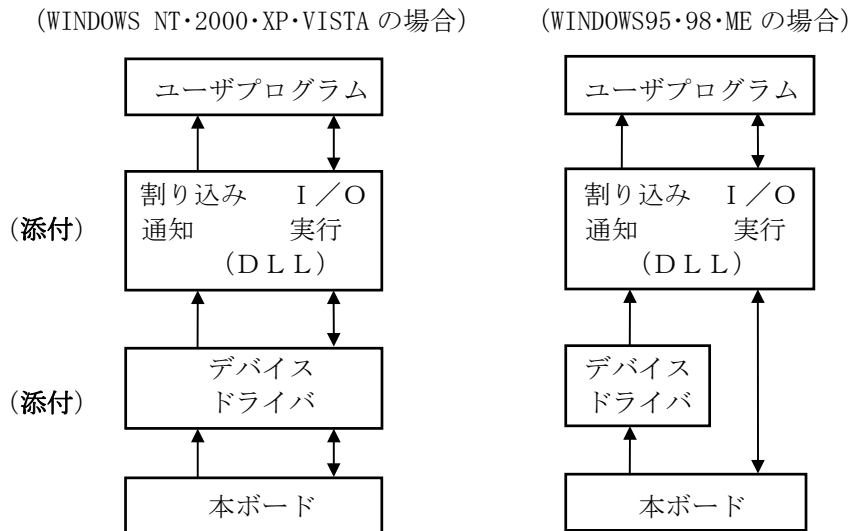
各サンプル等の内容は上記の
WINDOWS 9x・MEと
同様です。

【注】 WINDOWS (NT/9 x/2000/XP/VISTA) の各汎用 I/O 制御 DLL、デバイスドライバは別途。

4-2. WINDOWS (汎用) ドライバについて

WINDOWS 9 x / NT / 2000 / XP / VISTA 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、第 8 章) を利用する場合は不要です。



WINDOWS 3.1 : Win 31 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 9 x : Win 9 x 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 2 K フォルダ以下に格納されており、VB (6.0) で利用 (or XP or VISTA) 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. リソース情報取得／（WINDOWS 9x・ME・NT用）

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5章、6章に記す専用ハンドラ（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&ドライバが必要になります。（ドライバはボードインストール時にインストールされます。） 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=201H（ADM-681a PCI）

【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

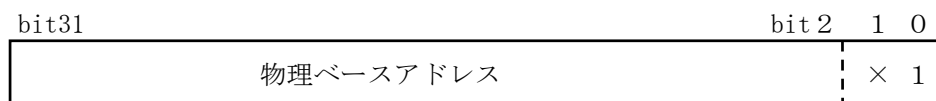
PCIリソース情報の取得は【1】で検出したロケーション（magic）とレジスタ番号を指定して行います。物理ベースアドレス値はダブルワードなので【2】の関数を使用します。

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 : 未使用
- 14H : 未使用
- 18H : I/OマップレジスタのベースアドレスBASE1（3-3項）
- 1CH : メモリマップレジスタの物理ベースアドレスBASE2（不使用）
- 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 2bit をマスクした値がベースアドレス値です。



【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。

4-4. リソース情報取得／(MS-DOS)

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。)

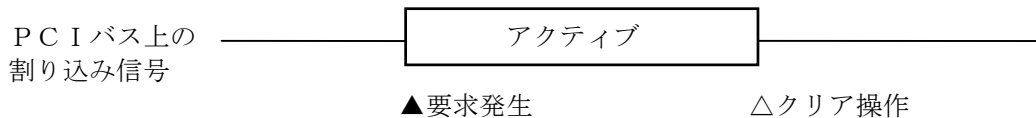
Int ReadPciByte (WORD magic, WORD reg, BYTE _far *data)	
引数	magic: 【1】GetPciDevice で得られた マジック番号、 *data:データ取得先ポインタ、 reg: PCI コンフィギュレーション空間ヘッダ領域のレジスタ番号。
戻り値	0:成功、 -1:失敗

使用方法 : 前4-3項と同様です。

4-5. 割り込み動作について

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4-5.



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。

添付の汎用デバイスドライバ（WINDOWS 9x/NT用）では、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。

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。

同じく添付の本ボード専用ハンドラ/デバイスドライバ（WINDOWS 9x/NT用）では割り込みによるADデータ転送、または外部イベント対応処理が標準でサポートされています。（割り込み不使用でもマルチスレッドでポーリング/ADデータ転送できます。）

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。

4-6. Quick Basicサンプル

BASIC文だけで記述したサンプルソフト68■QB1. BAS／68■QB2. BASがあります。

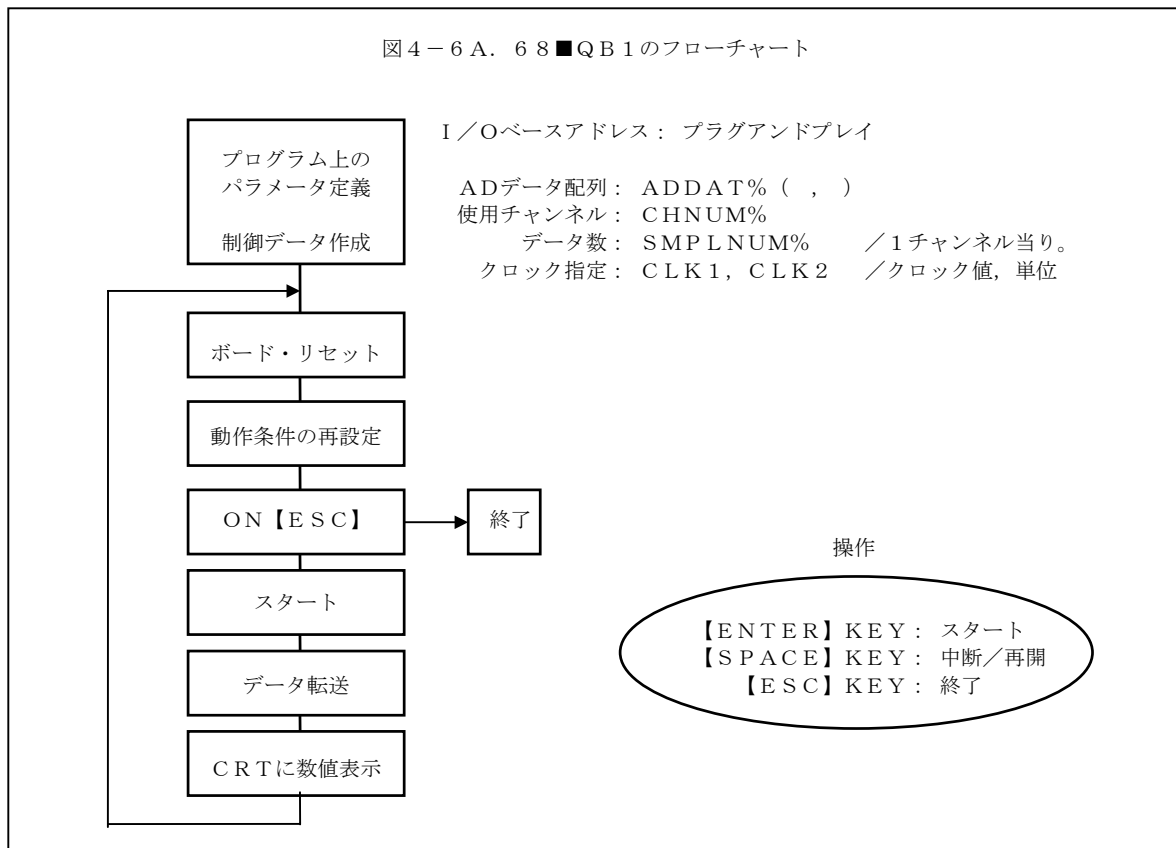
68■QB1. BAS

内部クロックを使用して指定点数だけ自動連続サンプリング、数値表示を繰り返し行います。データ転送はFIFOメモリのEMPTY解消【Not Empty】フラグを監視（ポーリング）して行います。FIFOメモリ容量までは本ボード自体の最高サンプリング速度が可能、それ以上はパソコン本体の実行速度（CPU／クロック）に依存します。

《注1》 例えばCPU=Pentium/166MHzのパソコンで本プログラムを使用するときの最高データ転送速度は、
 25KHz（at 1チャンネル）
 10KHz（at 4チャンネル）
 2.5KHz（at 16チャンネル）程度です。

《注2》 本ボード搭載のFIFOメモリ入力側速度（使用チャンネル数×サンプリング周波数）が出力側速度（パソコン側へのデータ転送速度）より速いときは同メモリの充満量が次第に増えてゆき、ついにはオーバーフローを起こしてエラー（ERR）フラグが立ちます。当時点以降にサンプリングされた新データは全て消失されますが、
 【ここでトリガ禁止操作によりサンプリングを止めれば】FIFOメモリ内のデータは全て有効に読み出すことができます。なおオーバーフロー発生がブロック転送実行タイミングと重なった場合はFIFOメモリ末尾側に最大1ブロック転送分の空領域を残すような形となります。（本サンプルではブロック転送は不使用）

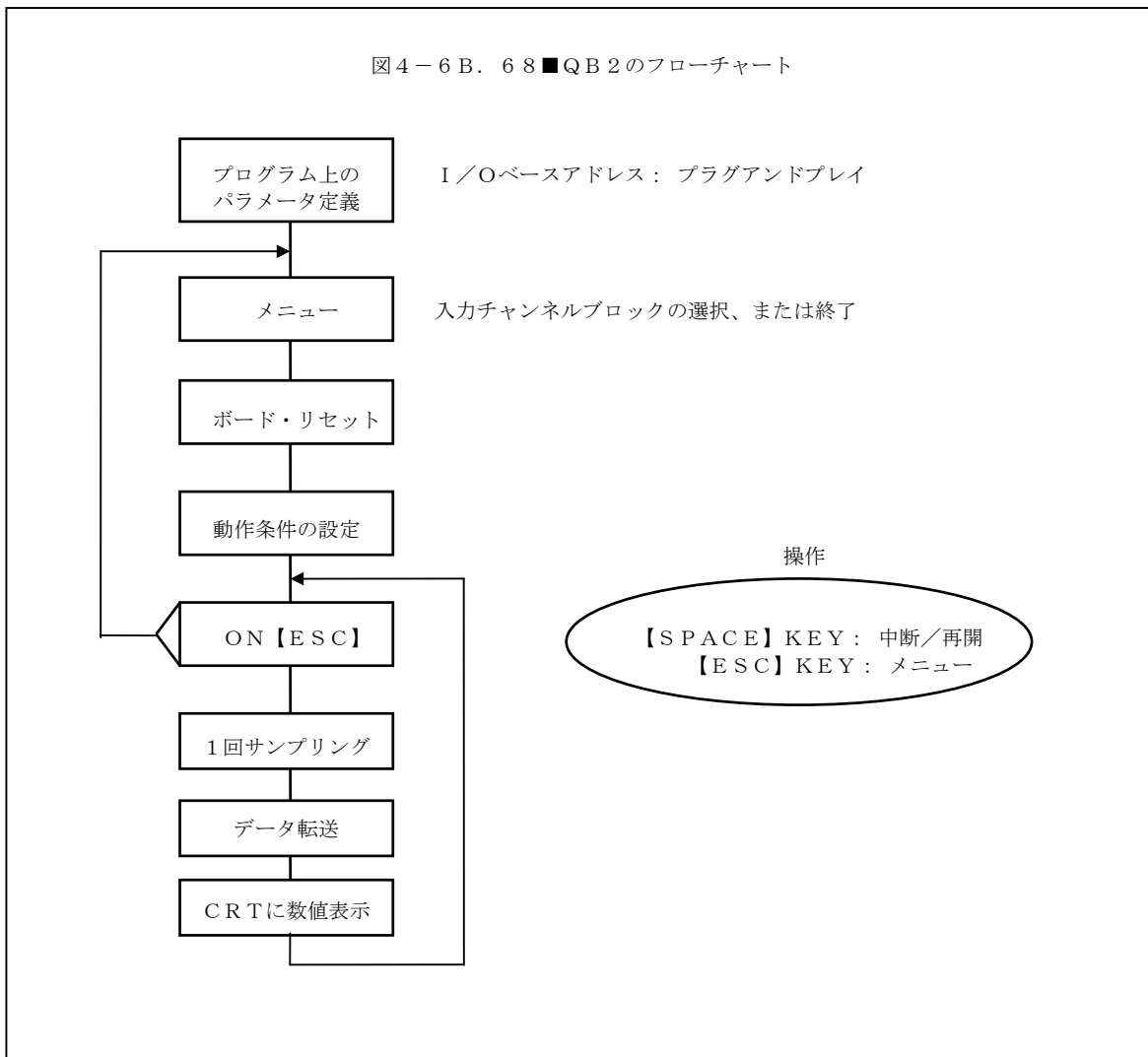
図4-6A. 68■QB1のフローチャート



68 ■ QB2. BAS

本例では1回ADサンプリング・スキャン／数値表示を繰り返し行います。
 実行形式68 ■ QB2. EXEも用意されており、動作確認・調整時のモニタに利用できます。
 (1-6項. 参照)

図4-6B. 68 ■ QB2のフローチャート



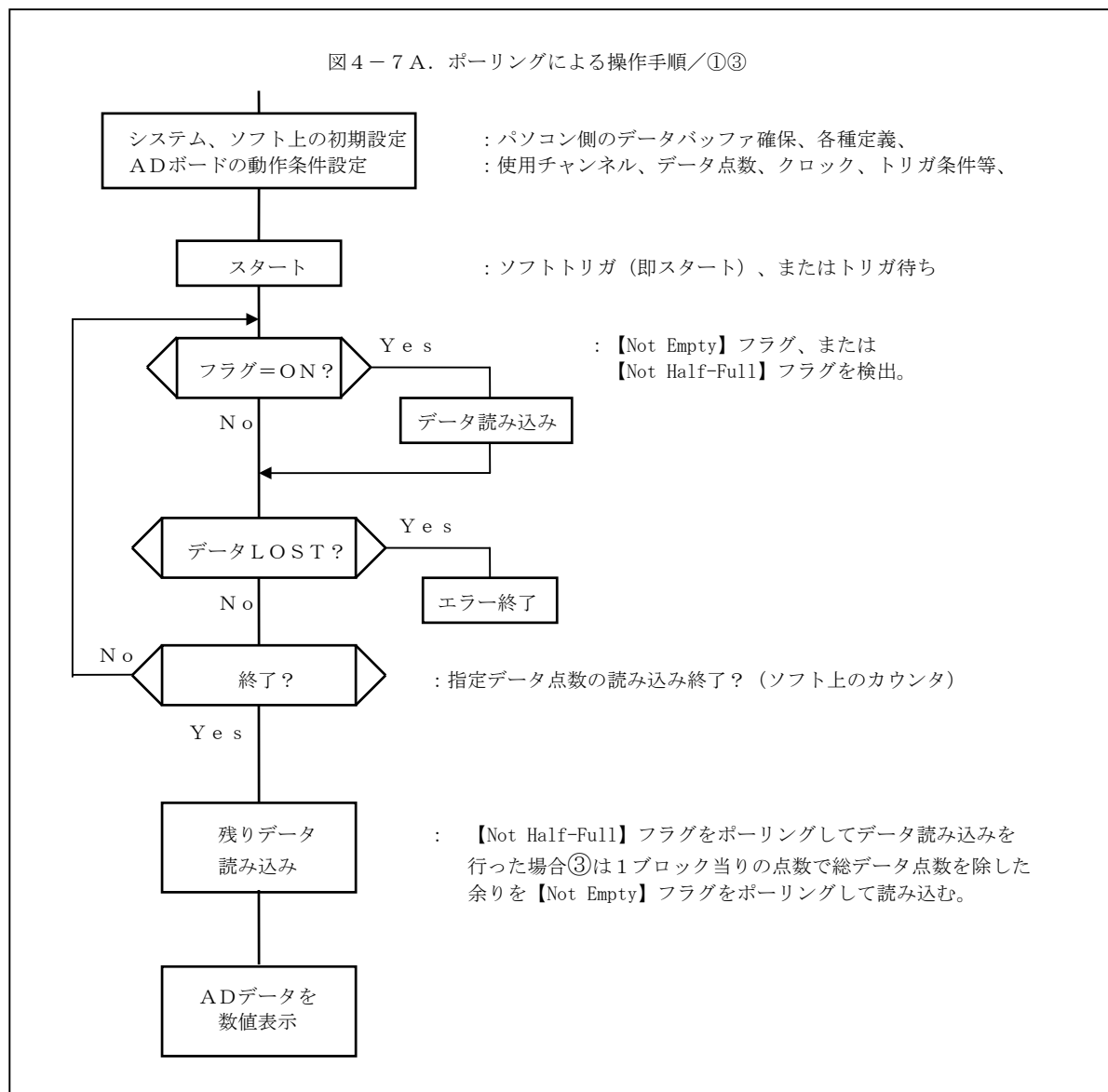
4-7. Cのサンプル

一般的な応用場面では基本機能・動作が関数化されたDOS版ハンドラ（第5章）の利用が便利です。本項ではADボードのドライバを自作するユーザ向けの操作手順学習用サンプルについて記します。

①POL68■E1. C
②INT68■E1. C
③BLK68■H1. C

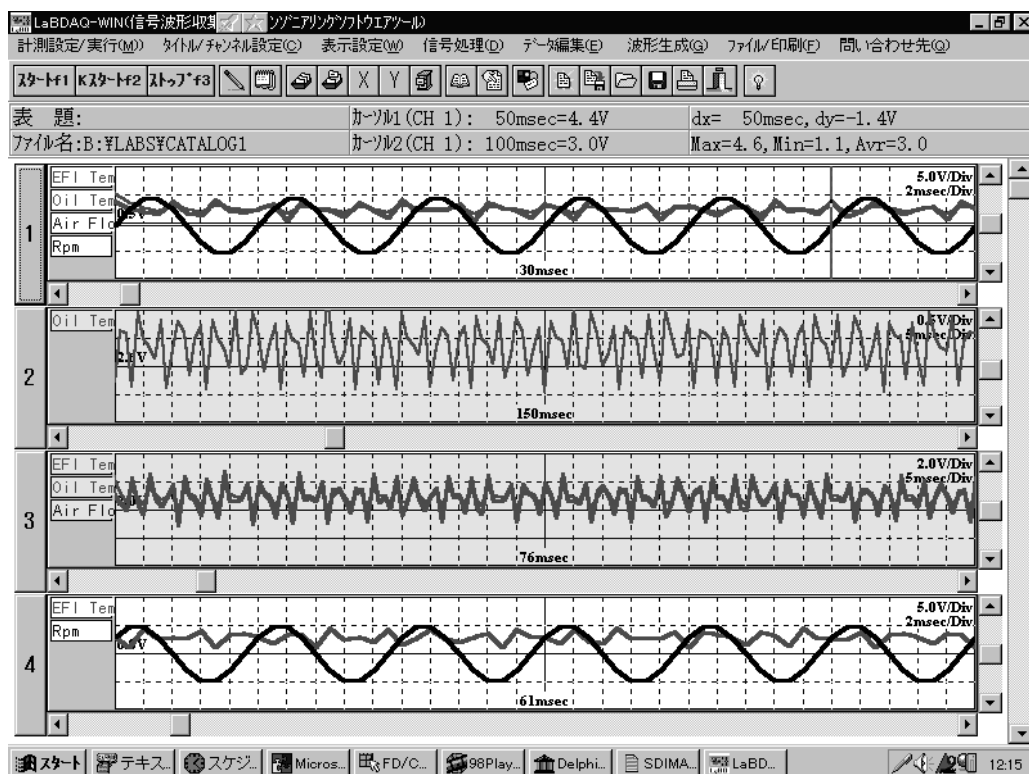
- ①は【Not Empty】フラグを監視して1語ずつ、
②は【Not Empty】状態変化による割り込みを使用して、同処理ルーチン内でデータを読み込む。
③は【Not Half-Full】フラグを監視してFIFOメモリ容量の半分（標準:512語）単位でブロック転送命令（INSW）を使用してデータを読み込む。／最高速の方法です。／

図4-7A. ボーリングによる操作手順／①③



《信号処理なし版：L a B D A Q - A Qは無償ダウンロード配布》
 W I N D O W S (2 0 0 0 ・ X P ・ V I S T A) データ収集・解析ソフトウェア

L a B D A Q - A Q / P R O



L a B D A Q - P R OはADボード、データファイルから読み込んだ時系列データ群のグラフ表示（リアルタイム可能）、ハードコピー、レポート印刷、市販ソフトにも読み込めるテキストファイル保存、さらにF F T ・ 相 関 ・ 移 動 平 均 ・ ピーク検出等のデータ処理まで行うことのできるパッケージソフトです。（無償版L a B D A Q - A Qは当社ホームページからダウンロード）

- ◆データ格納チャンネル数：最大512（ADデータ入力用、演算処理結果出力用の合計）
- ◆データ表示ウィンドウ数：最大32（同時表示は横4列×縦8列）
- ◆縦軸は任意の物理単位・スケールで表示可能、横軸は時間またはデータ順番号。
- ◆カーソルによる座標認識（時刻・値）、任意区間のズームング。
- ◆ADサンプリング実行中でもリアルタイム表示可能。
- ◆繰り返しADサンプリング実行機能。
- ◆F F T ・ 相 関 ・ 移 動 平 均 ・ ピーク検出等の汎用デジタル信号処理機能。
- ◆市販の表計算やデータ処理ソフトにインポート可能（C S Vテキストファイル保存機能）
- ◆多様なADボードに対応（マイクロサイエンス社製ADボード各機／別表）

動作環境

- ◇W I N D O W S 9 x ・ N T ・ 2 0 0 0 ・ X PまたはV I S T Aの動作する
I B M P C / A T互換機、9 8 N XまたはP C 9 8 0 1 ・ 9 8 2 1系機。
- ◇本体メモリ：16MB以上。
- ◇ハードディスク：空容量10MB以上
- ◇ディスプレイ：800×600以上。

ADサンプリング・デジタル信号処理仕様

- ◆AD入力チャンネル数 : 最大 512 (使用ボードのチャンネル数: 各 1 枚のみ対応/別表)
- ◆データ格納チャンネル数: 最大 512 (ADデータ入力用、演算処理結果出力用の合計)
- ◆最高サンプリング速度 : 各ADボードの最高値 (チャンネル数、データ数に依る/別表)
- ◆最大 (管理) データ量 : 搭載空RAM、および仮想メモリの範囲内。
- ◆デジタル信号処理機能 : チャンネル間の四則演算、直流成分の抽出または除去、
【PRO版のみ】単純移動平均、2次多項式適合平均、ピーク検出、
FFTパワースペクトル、位相スペクトル、振幅スペクトル、
自己相関、相互相関、伝達関数、AMDF、ヒストグラム、
IIRフィルタ (最大 30 次)、FIRフィルタ (最大 125 次)
(演算精度: IEEE規格32BIT浮動小数点/ANSI準拠)

データファイルの保存/読み込み

- ◆ADデータ保存: L a B D A Q専用バイナリ形式による高速保存/読み込み、または汎用CSVテキストファイル (表計算等、市販ソフトに読み込み可能)、波形解析ソフトD A D i S P専用形式。
- ◆動作条件保存 : ADサンプリング動作条件の保存/読み込み

サンプリング条件 (トリガ・クロック・モード) の設定

計測実行条件の設定

スタートモード

☒ 即時スタート
☐ トリガスタート

実行モード

☒ 1回のみ
☐ ストップまで連続

実行中状況表示

☐ 計測中状況表示しない(高速)
☐ 計測中カウントのみ表示する(中速)
☐ 計測中波形表示する(低速)

取込サンプリングクロック

1K: 1mS

クロック基準

☒ 1MHz(通常)
☐ 1.024MHz(FFT解析用)

トリガ条件設定

トリガ発生源
☒ 内部チャンネル0
☐ 外部コネクタトリガ端子

トリガデータ/電圧

128 0.0000 Volts

トリガモード

☒ レベル
☐ エッジ

トリガ極性

☒ - ↓
☐ + ↑

実行 中止(A)

最大512のデータ格納チャンネル（ADデータ入力用、演算処理結果出力用の合計）

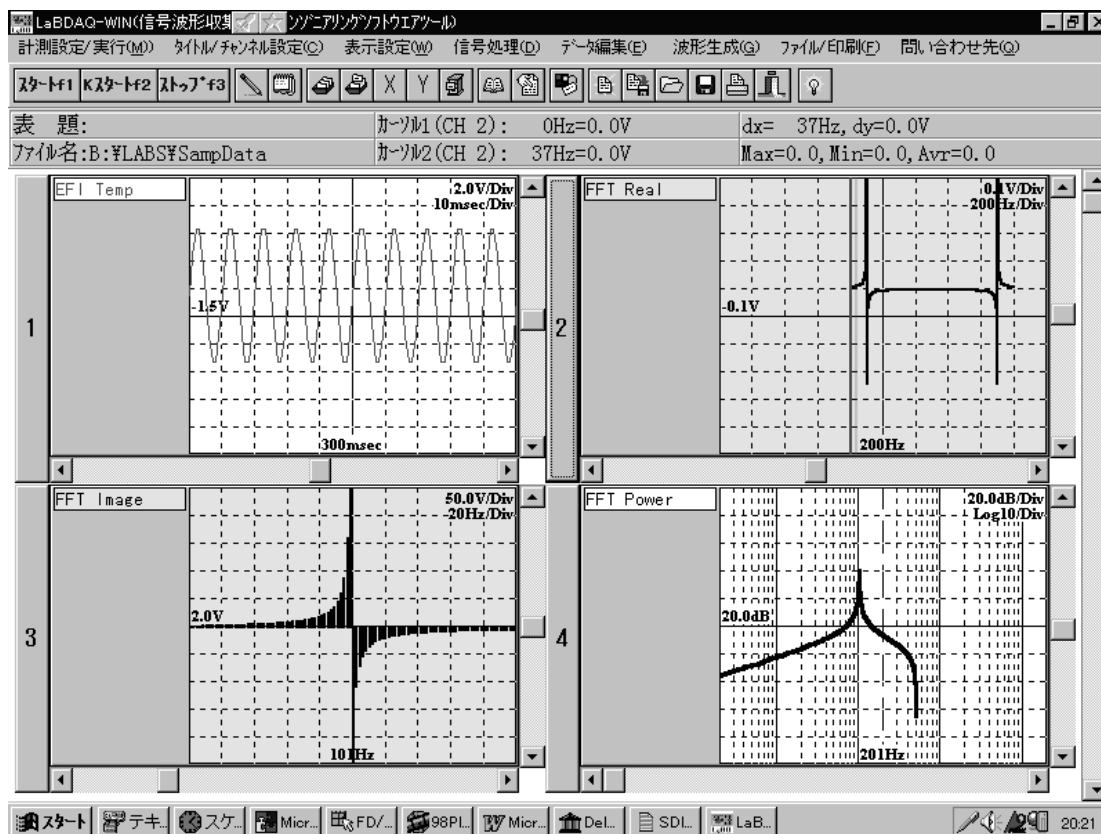
チャンネル設定

サンプリングチャンネル数: 4 サンプリングデータ点数: 512 (チャンネル×データ点数32M以下)
 演算チャンネル数: 4 演算データ点数: 512 (1チャンネル当たり1M以下)
 (両チャンネル計で32以下)
 データ間隔時間(サンプリングクロック): 1.0000 ☐ usec ☒ msec ☐ sec

チャンネルNo	信号名称	表示形式	表示色	変換w	変換係数a,b (Y=aX+b)	データ数
1 サンプリング用		実線2		<input type="checkbox"/>	1.000 0.000	512
2 サンプリング用		実線1		<input type="checkbox"/>	1.000 0.000	512
3 サンプリング用		実線3		<input type="checkbox"/>	1.000 0.000	512
4 サンプリング用		実線2		<input type="checkbox"/>	1.000 0.000	512

OK

入力データ（波形）に対するFFT等のデジタル信号処理



実行速度

下表は代表的な条件での最高サンプリング速度です。実際にはシステム構成や総データ点数等によって異なります。F I F Oメモリ付ADボードの場合、F I F Oメモリ容量までは（C P Uを含む）システム能力にかかわらず常にADボード自体の最高速度が得られます。

ADボードのF I F Oメモリ容量は現在、最大32M語まで増設可能です。
（ソケット実装により後日の増設も可能）

各ADボード/最高サンプリング速度：1Mデータの場合（複数チャンネルの場合は周期で示す）

ADボード名	P e n t i u m (400MHz)		486 (66MHz)	
	単チャンネル	複数チャンネル	単チャンネル	複数チャンネル
ADM-640AT	1MHz	(1×ch) μs	1MHz	(1×ch) μs
ADM-652AT	100KHz	(10×ch) μs	80KHz	4+(12×ch) μs
ADM-656AT	40KHz	(25×ch) μs	40KHz	(25×ch) μs
ADM-670PCI	2MHz	1MHz		
ADM-676PCI	200KHz	(5×ch) μs		
ADM-677PCI	100KHz	(10×ch) μs		
ADM-680xPCI	256KHz	(4×ch) μs		
ADM-681aPCI	500KHz	(2×ch) μs		
ADM-682PCI	256KHz	(8×ch) μs		
ADM-686zPCI	200KHz	(5×ch) μs		
ADM-687zPCI	200KHz	(5×ch) μs		
ADM-688PCI	500KHz	(2×ch) μs		
ADS-0128K	200KHz	(10×ch) μs		
ADM-5298BPC	100KHz	(10×ch) μs	35KHz	(32×ch) μs
ADM-5598BPC	100KHz	(10×ch) μs	35KHz	(32×ch) μs
ADM-5698BPC	40KHz	(25×ch) μs	40KHz	(25×ch) μs
ADM-5898BPC	83KHz	4+(8×ch) μs	35KHz	(32×ch) μs
ADM-8298BPC	200KHz	(5×ch) μs	100KHz	4+(11×ch) μs
ADM-8398BPC	200KHz	(5×ch) μs	100KHz	4+(11×ch) μs
ADM-8498BPC	200KHz	(5×ch) μs	100KHz	4+(11×ch) μs
ADM-8698BPC	166KHz	(6×ch) μs	100KHz	4+(11×ch) μs

【注】ISAボード：WINDOWS 9x・NTのみ対応、Cバスボード：WINDOWS 9xのみ対応。

◆合計データ点数：1データは2バイトですから、パソコンの空メモリ（RAM）に設定仮想メモリ量を加算した値を2で除したものになります。

但し仮想メモリ領域はディスクですから、サンプリングデータ数が空RAM領域を超えると急に遅くなります。上表の値はRAM領域に書き込む場合の速度です。

◆リアルタイム表示：サンプリングを実行しながら同時に経過をグラフ表示することもできます。実行・描画速度はパソコン側の能力にもよりますが、数ms／点程度です。

◆供給メディア：3.5インチFD（1.44MB）／IBM互換機・98系機共用。

◆価格
L a B D A Q - A Q ¥無償（ADデータ収集専用、2000/XP/VISTA版）
L a B D A Q - P R O ¥78,000（ADデータ収集・解析、フル機能版）

第5章. DOSハンドラ

当社製ADボード（IBM PC/AT互換機＝DOS/V機用）をMS-C、TURBO-C、およびBORLAND-Cで簡単に使用することのできるMS-DOS用ハンドラ（LIB）です。

ADボードの基本機能が関数化されており、ユーザは御自身の記述するメインルーチンの中から呼び出して使用することができます。すなわち、必要なパラメータ（動作条件）を変数に代入して本ハンドラを呼び出すだけでADサンプリングからパソコン本体メモリへのデータ転送まで、割り込み・ポーリング・ブロック転送等により高速実行されます。

5-1. 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
本体メモリ量 : 640KB以上（EMS、XMSメモリ使用可能）

OS/コンパイラ : MS-DOS（3.0以上）上で、
MS-C（7.0）/TURBO-C（4.0）/BORLAND-C（3.1）以上。

【注1】 本ハンドラ・ライブラリはWINDOWSのDOS窓では使用できません。

【注2】 本ハンドラ・ライブラリは16ビット・コード専用です。

供給メディア : 各ADボード添付のサンプルディスク内。

対応ADボード : ADM-681/681a PCI 【注】 FIFOメモリ32K語以内

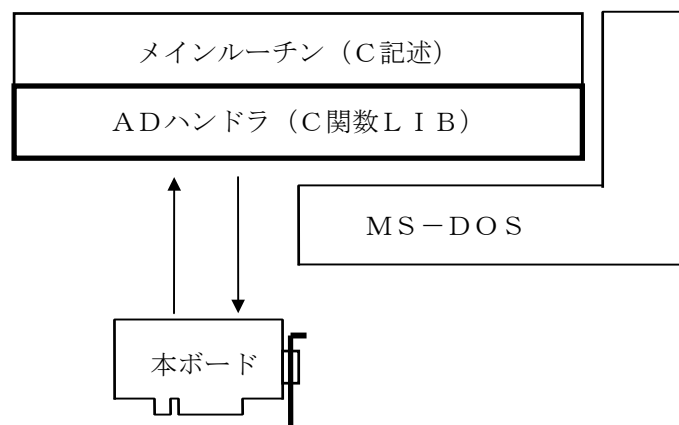
サンプリング : データ点数がFIFOメモリ容量までは各ボードの最高速度が可能、
それ以上の場合はCPU等に依存。（要実測）

データ点数 : 最大128K語、または拡張メモリ容量（1語＝2バイト）

割り込み : 使用は任意。（使用する場合は要リソース取得／1-5項参照。）

その他のボード : I/Oアドレス、割り込み番号が重複しない限り同時に使用可能。

図5-1. ソフトウェア構造



ハンドラの使用法はとても簡単です。具体的には5-4項以下で詳述しますが、要は使用するチャンネル数/サンプリング点数/クロック値/トリガ(スタート)条件等のパラメータをセットして各関数を呼ぶだけで、ADデータはバッファに格納されて戻ってきます。

本ハンドラでは大別して以下に記す2形態のサンプリング動作が可能です。

- 【注1】 指定チャンネル群： 各チャンネルを任意の順に並べた一群。
- 【注2】 内部（アナログ）トリガ： 指定レベル/極性と先頭チャンネル入力を比較して検出。
外部（デジタル）トリガ： 専用TTL入力（TRG-IN）の指定極性エッジ。
- 【注3】 外部クロック源入力： 専用TTL入力（CLK-IN）の指定極性エッジ。／最高10MHz

連続サンプリング
スタート関数の
呼び出しタイミング

外部トリガ
(負極性指定時)

トリガレベル

時刻

t_1

t_2

t_3

The diagram shows a horizontal axis for time (時刻) and a vertical axis for signal level. A box labeled '連続サンプリング スタート関数の 呼び出しタイミング' (Continuous Sampling Start Function Call Timing) points to the start of a curve. The curve starts at a low level and rises towards a 'トリガレベル' (Trigger Level) indicated by a dotted line. Three points on the curve are marked with circles. Horizontal arrows indicate time intervals: t_1 from the start to the first circle, t_2 between the first and second circles, and t_3 between the third and fourth circles. Above the curve, a square wave labeled '外部トリガ (負極性指定時)' (External Trigger (Negative Polarity Specified)) is shown, with its falling edge aligned with the start of the curve.

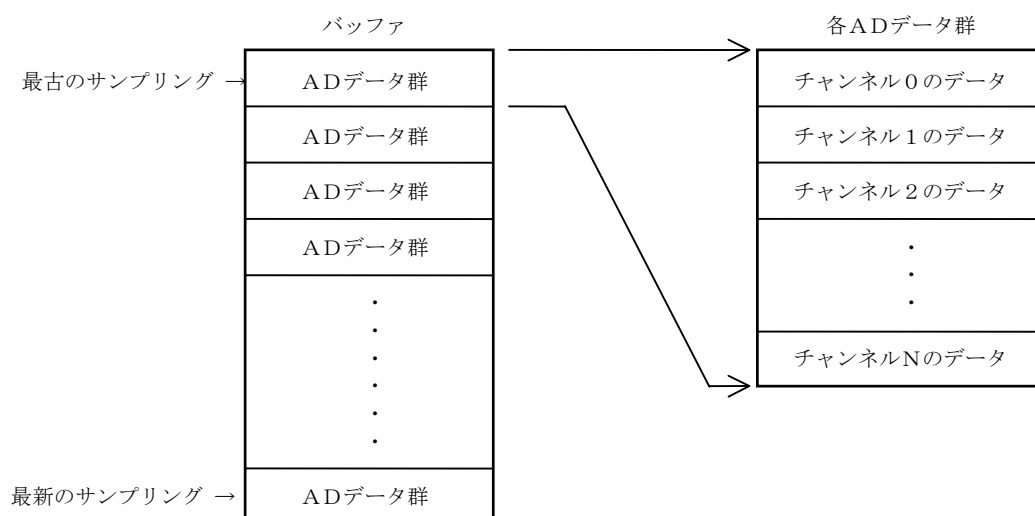
【注A】 ソフトトリガからの遅れ時間 t_1 はスタート関数の呼び出し、および同関数内の管理手続き実行時間を含むものです。 AD ボード自体のソフトトリガ・ビットをセットしてからの遅れは最大 125 ns です。

【注B】 内部（アナログ）トリガからの遅れ時間 t_2 、および外部（デジタル）トリガからの遅れ時間 t_3 は各スタート関数の呼び出しと同関数内の管理手続き実行してトリガ待ち状態となつてからの時間です。このトリガ待ち状態となるまでの時間は数 $100\text{ }\mu\text{s}$ を要します。

ADデータバッファと格納の様子

サンプルングされたADデータは指定バッファに転送されます。
 この（パソコン側メモリ上にある）バッファ内のデータ配置は（局所的には）ADボード上のFIFOメモリと同一イメージで全チャンネルのADデータが1個のバッファに格納されます。
 拡張メモリ使用時も同様です。

図5-2B. 全チャンネルのADデータが1個のバッファに入る



5-3. 使用準備（ボード上の設定、入力接続）：標準設定【1-2項】とします。

5-4. 関数仕様・エラーコード

御自身の記述したメインプログラムと本ハンドラ（LIB）をコンパイル／リンクして使用します。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。なおライブラリは4-1項に従ってインストールしておきます。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 : 【1】
- (2) ADデータバッファの確保 : 標準、または拡張領域（サンプルソース参照）。
- (3) サンプリング条件の設定 : 【2】～【6】 チャンネル／トリガ／クロック／動作モード。
- (4) スタート（or トリガ待ち） : 【7】 即スタート／トリガ待ち（内部 or 外部）
- (5) ステータス評価 : 【8】 連続サンプリング進捗状況
- (6) 以下、任意
- (7) ハンドラ終了 : 【9】 後処理

表 5-4 A. 制御関数一覧

関数名	機能・内容	引数（パラメータ）等
1】 ad_open_adsys	ボード、および本ハンドラの初期化	
2】 ad_set_sampch	サンプリング実行チャンネル関連設定	チャンネル数
3】 ad_set_sampmode	サンプリング動作モードの設定	ADデータ転送先、方法
4】 ad_set_trigger	トリガ関連設定（レンジトリガ以外）	トリガ源、レベル、モード
4】 ad_set_rangetrigger	トリガ関連設定（レンジトリガ）	
5】 ad_set_exclk	オプション、外部クロック源の設定	クロック源の周波数値
6】 ad_set_clock	サンプリング・クロックの設定	クロック源、周期値、単位
7】 ad_start_samp	サンプリング開始（トリガ待ち or 即）	ADデータ格納バッファ
8】 ad_get_status	ステータス取得	サンプリング進捗状況など
10】 ad_read_exmem	拡張メモリ内のADデータ読み込み	ADデータ格納バッファ
11】 ad_get_onescan	マニュアル（1回）サンプリング	
9】 ad_close_adsys	本ハンドラの終了	
	《以上が基本、以下は補助的》	
12】 ad_out_aux	汎用デジタル（ラッチ）出力	出力データ
13】 ad_inp_aux	汎用デジタル（現在値）入力	入力データ
14】 ad_set_samploop	データバッファをリング状に設定	
15】 ad_set_datacode	ADデータ・コードの指定	バイナリ、または2の補数
16】 ad_stop_samp	サンプリング動作の（強制）中止	
17】 ad_read_restdata	残りデータ読み込み	データロスエラー発生時
18】 ad_onkey_quit	key 押下でトリガ待ち停止の設定	BREAK(=CTRL+PAUSE)
19】 ad_onkey_trg	key 押下で強制トリガ動作の設定	ESC, SPACE, または ENTER
20】 ad_set_timeout	トリガ待ちタイムアウトの設定	秒単位
21】 ad_clear_flags	ボードのフラグクリア & 条件再設定	
22】 ad_onint_func	割り込み発生時に実行する関数の設定	ユーザ関数のポインタ
23】 ad_set_scanspeed	アナログ入力スキャン速度指定	
24】 ad_set_shc	外付・同時サンプルホールド制御	
25】 ad_get_libver	本ハンドラのバージョン番号を得る	

以下に各関数の仕様・詳細を記します。

【1】初期化

書式	int ad_open_adsys (void)
引数	なし
戻り値	正常終了時 : ボードのID / ADM-681aPCI : 14H エラー時 : エラーコード/負の値 (エラーコード表)
機能動作	プラグアンドプレイで設定したI/Oアドレス、割り込み番号を本ハンドラが自動認識すると共に、本ボードのリセット、ハンドラ内部の参照テーブルやデータバッファを初期化する。 また、ADボード上のFIFOメモリ容量をチェックする。

【注】 メガFIFOモジュール搭載のときは同モジュール上のDIPスイッチで<ハーフフル値>を設定しますが、本ハンドラのサポートする範囲は0 (512語) ~ 5 (16K語) までに限ります。
これは市販のFIFO素子で定義された元々の語意 (総容量の半分) を離れて、メモリの充満度を示すインジケータとなっています。

【2】サンプリング実行チャンネル関連の設定

書式	int ad_set_sampch (int no_ch, int scan_order[], int range[])
引数	no_ch : サンプリングを実行するチャンネル数。 scan_order [] : 各チャンネルのスキャン順。 (未使用: 本ボードでは固定) range [] : 各チャンネルの入力レンジ番号。 (未使用: 本ボードでは固定)
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)
機能動作	サンプリングを実行するチャンネル数、各チャンネルのスキャン順、入力レンジを設定する。 例えばno_ch : 5ならば、スキャン順番0~4のチャンネル群がサンプリング実行対象となる。 本ボードではスキャン順固定 (0, 1, 2, Max)、入力レンジはボード上のスイッチ設定。

【3】サンプリング動作モードの設定

書式	int ad_set_sampmode (int trs_trig, int buf_area, int intr_sw)
引数	trs_trig : 割り込み発信フラグ / 0 : EMPTY解消、1 : HALF-FULL 【注】 buf_area : ADデータ転送先 / 0 : 標準メモリ、1 : EMSメモリ、2 : XMSメモリ intr_sw : I/Oデータ転送で割り込み使用の有無 / 0 : 不使用、1 : 使用する
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)
機能動作	ADボードからパソコン本体メモリへのデータ転送方法、転送先メモリ、割り込み使用の有無を設定する。 割り込みを使用しないI/Oデータ転送のときはtrs_trigは無視され、ブロックI/O転送を含む最適化されたアルゴリズムでサンプリング&データ転送が実行される。

【注】 メガFIFOモジュール搭載のときは同モジュール上のDIPスイッチで<ハーフフル値>を設定しますが、本ハンドラのサポートする範囲は0 (512語) ~ 5 (16K語) までに限ります。
これは市販のFIFO素子で定義された元々の語意 (総容量の半分) を離れて、メモリの充満度を示すインジケータとなっています。 <この値をブロックとしたデータの一括転送命令を実行する>

【4】トリガ関連の設定A（レンジトリガ以外の場合）

書式	int ad_set_trigger(int trg_mode, int trg_source, int trg_pol, int trg_level)
引数	<trg_mode :="" トリガ動作モード／0="" ポストトリガ<br="" 即トリガ、1=""></trg_mode> trg_source : トリガ源 / 0 : ソフト、1 : 内部（アナログ）、2 : 外部 trg_pol : トリガ極性 / 0 : 負エッジ、1 : 正エッジ 2 : 負レベル、3 : 正レベル trg_level : トリガレベル（アナログ）／対応するADデータコード上位8ビットで指定。
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）
機能動作	トリガ動作モード、源、極性、レベル等の設定。 なお、トリガ動作モードをポストトリガ、トリガ源を外部（デジタル）で極性をレベルに指定したときは帯域サンプリングとなる。（3-1項参照） アナログトリガ・チャンネルはスキャン先頭チャンネル（ch0固定）。

【4】' トリガ関連の設定B（レンジトリガの場合）

書式	int ad_set_rangetrigger(int trg_sel, int trg_level_hi, int trg_level_lo)
引数	trg_sel : トリガ動作モード／0 : アウトレンジ、1 : インレンジ 2 : デュアルスロープ（+）、3 : デュアルスロープ（-） trg_level_hi : トリガレベル上限値 trg_level_lo : トリガレベル下限値
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）
機能動作	レンジトリガの極性を含む動作モード、レベルの設定。（レンジトリガ以外は前記【4】を使用） アナログトリガ・チャンネルはスキャン先頭チャンネル（ch0固定）。

【5】オプション、または外部クロック源周波数値の設定

書式	int ad_set_exclk (ULONG exclk_freq)
引数	exclk_freq : オプション、または外部クロック源の周波数値（Hz単位）
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）
機能動作	オプション、または外部クロック源を使用するとき、その周波数値を設定する。 後記【6】ad_set_clockで分周比によるサンプリングクロック指定、またはボードに標準搭載の内部クロック源を使用するときは必要ない。

【6】サンプリング・クロック値の設定

書式	int ad_set_clock(int clk_source, int set_mode, int *time_unit, ULONG *clk_period)
引数	<p>clock__source : クロック源／0 : 内部クロック源0 (40.00MHz) 1 : 内部クロック源1 (32.768MHz) 2 : 内部クロック源2 (オプション) 3 : 内部クロック源3 (未使用) 4 : 外部クロック源 (有効極性=↓) 5 : 外部クロック源 (有効極性=↑)</p> <p>set__mode : クロック値の指定方法／0 : クロック周期の値、 1 : 分周比 time__unit : クロック周期の単位 / 0 : s、 1 : ms、 2 : μs、 3 : ns clk__period : クロック周期の値、または分周比</p>
戻り値	<p>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)</p>
機能動作	サンプリングクロック値を設定する。なお、設定できないクロック周期の値を指定すると設定可能な長い方の近似値が設定される。また、クロック源に2, 4, 5を指定し、クロック周期の値で設定するときは前記【5】ad_set_exclckでクロック源周波数値を定義しておく。

【7】サンプリング・スタート (ADデータバッファが32K語以内のとき)

書式	int ad_start_samp (ULONG no_samp, WORD *bufptr, WORD bufsize)
引数	<p>no__samp : サンプリング・データ点数 (1チャンネル当りの点数) bufptr : ADデータ・バッファのポインタ bufsize : ADデータ・バッファの大きさ</p>
戻り値	<p>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)</p>
機能動作	<p>サンプリングを実行する。 前記【4】トリガ設定でトリガ動作モードを《即トリガ》としてあるときは即スタート、また《ポストトリガ》が選択されているときはトリガ待ちスタートとなる。前記【4】'で設定するレンジトリガのときは全てトリガ待ちスタートとなる。 データ転送先が拡張メモリEMS, XMSの場合、bufptrとbufsizeは無視される。 なお前回サンプリング実行時にセットされたフラグはここでクリアされるが、《エラー》フラグだけはクリアされない。《エラー》フラグをクリアするには 前記【1】初期化を実行する。</p>

【7】' サンプリング・スタート (ADデータバッファが32K語を超えるとき)

書式	int ad_start_samp_h (ULONG no_samp, WORD _huge *bufptr, ULONG bufsize)
引数	<p>no__samp : サンプリング・データ点数 (1チャンネル当りの点数) bufptr : ADデータ・バッファのポインタ bufsize : ADデータ・バッファの大きさ</p>
戻り値	<p>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)</p>
機能動作	<p>サンプリングを実行する。 前記【4】トリガ設定でトリガ動作モードを《即トリガ》としてあるときは即スタート、また《ポストトリガ》が選択されているときはトリガ待ちスタートとなる。前記【4】'で設定するレンジトリガのときは全てトリガ待ちスタートとなる。 データ転送先が拡張メモリEMS, XMSの場合、bufptrとbufsizeは無視される。 なお前回サンプリング実行時にセットされたフラグはここでクリアされるが、《エラー》フラグだけはクリアされない。《エラー》フラグをクリアするには 前記【1】初期化を実行する。</p>

【8】ステータス取得

書式	int ad_get_status (ULONG *sampled, int status[])
引数	sampled : サンプルング済みデータ点数 (1チャンネル当り) status [0] : ADボードのステータス1 / (3-12項) status [1] : ADボードのステータス2 / (未使用: 本ボードでは無い)
戻り値	正常終了時 : 連続サンプルング実行中=0、 停止中=1 エラー時 : エラーコード/負の値 (エラーコード表)
機能動作	当時点でのサンプルング済みデータ点数、およびADボードのステータス生データを得る。

【9】本ハンドラの終了

書式	int ad_close_adsys (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)
機能動作	本ハンドラの終了。 ADボードのリセット、確保したメモリ領域の開放等を行う。

【10】拡張メモリ内のサンプルング・データ読み込み

書式	int ad_read_exmem(ULONG no_data, ULONG data_pos, WORD *bufptr, WORD bufsize)
引数	no_data : 拡張メモリから読み出すデータ点数 (1チャンネル当り) data_pos : 拡張メモリから読み出すデータの先頭位置 (サンプルング順番号) bufptr : 転送先ADデータバッファのポインタ bufsize : 転送先ADデータバッファの大きさ
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)
機能動作	サンプルングされたADデータが拡張メモリ (EMS, XMS) に格納されているとき、任意の部分標準メモリ上のADデータバッファに転送する。

【11】マニュアル (1回) サンプルング・スキャン

書式	int ad_get_onescan (WORD *bufptr)
引数	bufptr : AD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)
機能動作	前記【2】で指定したチャンネル群に対して各1回だけサンプルングを実行する。

【12】汎用デジタル（ラッチ）出力

書式	void ad_out_aux (int out_data)
引数	o u t _ d a t a : 1ビット汎用デジタル（ラッチ）出力データ。
戻り値	なし
機能動作	汎用（ラッチ）出力データの更新。 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、ボードの制御部リセット（ソフトのリセット）では変化しない。

【13】汎用デジタル（現在値）入力

書式	int ad_inp_aux (void)
引数	なし
戻り値	汎用デジタル（現在値）入力データ。
機能動作	1ビット汎用デジタル（現在値）入力データの読み込み。

【14】データバッファをリング状／エンドレスに設定

書式	int ad_set_samploop (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）
機能動作	ADデータバッファをリング状・エンドレスで使用するか否かの設定。（トグル） リング状・エンドレスに設定した場合、ADデータバッファが満杯になると次は同バッファ先頭を上書きする形となる。この時、ステータス取得関数【8】で得るサンプリング済みデータ点数の値も先頭値（ゼロ）に戻る。 本設定によりADボードは無限サンプリング・モードとなる。 長時間の監視システム等に利用できる。

【15】アナログ入力モードの設定

書式	int ad_set_inpmode (int inp_mode, int ad_code, int ad_reso)
引数	i n p _ m o d e : 未使用／アナログ入力信号形式（ボード上のスイッチ設定／2-1項） a d _ c o d e : ADデータコード指定／0：バイナリ、1：2の補数 a d _ r e s o : 未使用／AD分解能（ボード上のスイッチ設定／2-2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）
機能動作	アナログ入力信号形式、変換出力データ・コード（符号化の型）、およびAD変換分解能を指定する。

【16】 サンプリング動作の強制停止

書式	int ad_stop_samp (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)
機能動作	サンプリングを強制的に中止する。 (ADボード上のFIFOバッファメモリに残りデータがあれば、それらは無効となる。)

【17】 残りADデータの読み込み

書式	long ad_read_restdata ()
引数	なし
戻り値	正常終了時 : =読み込んだ(残り)データ数 エラー時 : エラーコード／負の値 (エラーコード表)
機能動作	データロス・エラー発生(検出するとサンプリングを強制停止させる)したとき、 ADボード上のFIFOバッファメモリに残りデータがあれば、これを当関数で読み込むことができる。

【18】 (BREAKキー押下による) トリガ待ち、またはサンプリング中止の設定／解除

書式	int ad_onkey_quit (int set_mode)
引数	set_mode : 動作の指定／ 0 : 解除、 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)
機能動作	【BREAK=CTRL+PAUSE】キーの押下により トリガ待ち(サンプリング動作中のときはサンプリング)を強制的に中止する。 前記【7】ad_start_sampの実行によってトリガ待ち状態となったにもかかわらず 何かの理由でトリガが発生しない場合、人為的に待ちループから抜ける手段となる。

【19】 (キー押下による) 即トリガ動作の設定／解除

書式	int ad_onkey_trg (int act_key)
引数	act_key : 動作の指定(動作keyの選択)／ 0 : 解除、 1 : ESC、 2 : SPACE、 3 : ENTR
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)
機能動作	選択したキーの押下により強制的に即トリガ動作する。 前記【7】ad_start_sampの実行によってトリガ待ち状態となったにもかかわらず 何かの理由でトリガが発生しない場合、人為的に待ちループから抜ける手段となる。

【20】（トリガ待ち）タイムアウトの設定／解除

書式	int ad_set_timeout (WORD set_time)
引数	set_time : タイムアウトまでの時間（単位：秒）、0で解除。
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）
機能動作	前記【7】トリガ待ち開始後、ここで設定した時間を経過するとトリガ待ちを止める。

【21】（ADボード）各フラグのクリア、およびサンプリング条件の再設定

書式	int ad_clear_flags (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）
機能動作	ADボード上FIFOバッファメモリの各フラグをクリアし、直前に設定したサンプリング条件を再設定する。

【22】割り込み発生時に実行するユーザ関数の設定

書式	int ad_onint_func (int intr_source, void interrupt_far *func)
引数	intr_source : 割り込み要因／ 0 : 連続サンプリングの各回スキャン開始時 1 : 外部割り込み入力（↓負エッジ） 2 : 外部割り込み入力（↑正エッジ） 3 : トリガ発生 4 : 連続サンプリングの各回スキャン終了時 func : ユーザ作成関数のポインタ
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）
機能動作	ユーザが任意に作成した割り込み処理関数の指定。 但し、当指定は【3】サンプリング動作モードで《I/Oデータ転送で割り込み不使用》に設定されているときのみ有効である。

【23】アナログ入力スキャン速度指定

書式	int ad_set_scanspeed (int speed)
引数	speed : アナログ入力スキャン速度／ 0 : 高速モード、 1 : 低速モード、
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）
機能動作	スキャン速度により（複数チャンネル使用時に）得られるADデータの正確度が変わる。 詳しくは2-2項、および3-6項を参照。

【24】 外付・同時サンプルホールド制御

書式	int ad_set_shc (int shc)
引数	shc : 外付・同時サンプルホールド制御 / 0 : タイミング調整しない、1 : 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)
機能動作	複数チャンネル同時サンプルホールド回路を外付・接続する場合の各サンプリングスキャン開始タイミング制御。shc = 1 のときは1 μ s 遅らせることで先頭チャンネル (ch0) にかかるホールドステップ誤差を最少化するが、スキャン時間が1 μ s 増加するため最高サンプリング速度が遅くなる。(3-18項参照)

【25】 本ハンドラのバージョン取得

書式	int ad_get_libver (int ver)
引数	ver : = 0 のとき、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 (上位バイト) (下位バイト)
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)
機能動作	本ハンドラ (ライブラリ) のバージョン番号を得る。

□■ Cハンドラ使用例 ■□

本ハンドラ (LIB) とコンパイル / リンクして使用します。

表 5-4 C. ハンドラ使用例Cソース

ファイル名	動作モード	ADデータ・バッファ領域
SAMPLE1.C	ポーリング、または割り込み使用例。	標準メモリ (コンベンショナル)
SAMPLE2.C	32K語を超えるデータ数の使用例。	標準メモリ (コンベンショナル)
SAMPLE3.C	拡張メモリ使用例。	EMS、XMS
SAMPLE4.C	割り込みによるユーザ関数の使用例。	標準メモリ (コンベンショナル)
SAMPLE5.C	リング状バッファ動作例。	標準メモリ (コンベンショナル)
SAMPLE6.C	オプション、外部クロック源の使用例。	標準メモリ (コンベンショナル)

□■ 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5-4B. 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。（ボード不在等）	ad_open_adsys ()
－ 2	ハンドラ初期化の未実行。	
－ 3	スタート関数【7】【7】' 実行前の条件設定不足。	ad_start_samp ()
－ 4	サンプリング実行チャンネル数 指定パラメータの不適当	no_ch
－ 5	サンプリング・スキャン順 指定パラメータの不適当	scan_order []
－ 6	アナログ入力レンジ 指定パラメータの不適当	range []
－ 7	割り込み発信フラグ（データ転送トリガ） パラメータの不適当	trs_trig
－ 8	ADデータ転送先 指定パラメータの不適当	buf_area
－ 9	ADデータ転送に割り込み使用の有無 指定パラメータの不適当	intr_sw、
－ 10	【22】ad_onint_func で割り込みを使用している	intr_sw、
－ 11	トリガ動作モード 指定パラメータの不適当	trg_mode
－ 12	トリガ源 指定パラメータの不適当	trg_source
－ 13	トリガ極性 指定パラメータの不適当	trg_pol
－ 14	トリガ源選択 指定パラメータの不適当	trg_sel
－ 15	トリガレベル（レンジトリガ以外） 指定パラメータの不適当	trg_level
－ 16	トリガレベル（レンジトリガ） 指定パラメータの不適当	trg_level
－ 17	トリガレベル上限値・下限値の大小関係不適当	trg_level
－ 18	クロック源 指定パラメータの不適当	clk_source
－ 19	サンプリング・クロックの指定方法 パラメータの不適当	set_mode
－ 20	クロック周期の単位 指定パラメータの不適当	time_unit
－ 21	クロック周期の値、または分周比 指定パラメータの不適当	clk_period
－ 22	クロック源周波数が未設定なのにクロック周期で指定した。	exclk_freq
－ 23	クロック周期、または分周比の値が大きすぎる。	clk_period
－ 24	ADデータバッファが小さすぎる。	no_samp、bufsize
－ 25	マニュアル（1回）サンプリングスキャンのチャンネルが未設定	ad_get_onescan ()
－ 26	アナログ入力信号形式 指定パラメータの不適当	inp_mode
－ 27	ADデータ・コード 指定パラメータの不適当	data_code
－ 28	AD変換分解能 指定パラメータの不適当	ad_reso
－ 29	拡張メモリ～ADデータバッファ間、1回の転送は32KW以内	ad_read_exmem ()
－ 30	拡張メモリからサンプリング点数を超えて読み出そうとした。	ad_read_exmem ()
－ 31	サンプリング動作（ADデータ転送）に割り込みを使用している	ad_onint_func ()
－ 32	割り込み要因 指定パラメータの不適当	ad_onint_func ()
－ 33	ユーザ関数が存在しない。	ad_onint_func ()
－ 34	割り込みリソースが割り当てられていない。	ad_open_adsys ()
－ 40	（データロスト、オーバーラン）エラー発生	
－ 41	BREAKキイでトリガ待ち、またはサンプリングを中止した。	ad_onkey_quit ()
－ 42	タイムアウトによりトリガ待ちを中止した。	ad_set_timeout ()
－ 43	サンプリングは既に終了している。	ad_stop_samp ()
－ 44	現在、サンプリング中である。	ad_read_restdata ()
－ 50	トリガ待ち解除、サンプリング中止は既に設定済み。	ad_onkey_quit ()
－ 51	トリガ待ち解除、即トリガは既に設定済み。	ad_onkey_trg ()

表5-4B. 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
—60	EMS 関連：常駐していない。	
—61	EMS 関連：物理ページのエントリ取得エラー。	
—62	EMS 関連：物理ページ数が不足。（4ページ必要）	
—63	EMS 関連：ページ・フレームのアドレス取得エラー。	
—64	EMS 関連：未アロケート・ページ数取得エラー。	
—65	EMS 関連：メモリ不足エラー。	
—66	EMS 関連：ページの割り当てとハンドルの取得エラー。	
—67	EMS 関連：マッピングエラー、EMS データ転送エラー（割り込み時）。	
—68	EMS 関連：既に開放されている。	
—70	XMS 関連：常駐していない。	
—71	XMS 関連：ドライバ・エントリアドレス取得エラー。	
—72	XMS 関連：フリーメモリ取得エラー。	
—73	XMS 関連：メモリ不足エラー。	
—74	XMS 関連：メモリ確保エラー。	
—75	XMS 関連：転送バッファ確保エラー。	
—76	XMS 関連：データ転送エラー。	
—77	XMS 関連：マッピングエラー、XMS データ転送エラー（割り込み時）。	
—78	XMS 関連：既に開放されている。	

□■ Cハンドラ使用上の注意 ■□

- (1) 本ハンドラは不適切な使い方をするエラーを返してきますから、原因が除去されるようにデバッグしてください。特に問題を起こしやすい点は、

- ◆ハード（ADボード）とソフト（ハンドラ）設定の不整合。
- ◆ADボードと他のハード（非PnP対応ボード）の設定重複。 : I/Oアドレス
割り込み番号

特にハード同志の重複設定による不具合は本ハンドラで検出できませんから、システムの構築時に十分な確認が必要です。

- (2) 本ハンドラは【ad__open__adsys（）】で使用開始、
【ad__close__adsys（）】で使用終了とします。

特に終了手続きは本ハンドラ内で確保したメモリの開放等を含むので重要です。

- (3) 本ハンドラの管理する割り込みは1系統のみです。

サンプリング実行時の《ボード上のFIFOバッファ》～《パソコン側データバッファ》間データ転送の実行タイミングとして、またはユーザ関数実行タイミングとしてのいずれか一方にのみ利用できます。

- (4) データロスト・エラー

ADボードの自動サンプリング動作でFIFOバッファメモリにADデータが流入する速度よりパソコン側から読み出す速度が遅く、FIFOバッファメモリが満杯になったうえに次のデータが書き込まれようとする、そのデータはこぼれ（失われ）てしまいます。

これがデータロスト・エラーで、構築されたシステム全体の処理速度を超えたクロック値を指定して実行したときに起ります。なおデータロスト・エラーの発生を本ハンドラが検出すると連続サンプリングを強制停止させますが、それ以前のデータ（FIFOバッファメモリ中にある）は残りデータとして専用関数【17】で有効に読み出すことができます。

システムパフォーマンスはCPUを含むパソコン本体の実行速度や周辺機器の状態に左右されますが、（ユーザが記述する）本ハンドラの応用プログラムがサンプリング実行&データ読み込み（ブロック転送）に専念した場合は本ADボード自体の最高サンプリング速度が実現可能です。

- (5) 直接I/O操作を行うとき

本ハンドラを介さずADボードに直接I/O操作（OUT／INP）を行うと、本ハンドラの管理が行き届きませんから不本意な動作となることがあります。

第6章. WINDOWSハンドラ

《 追加の本ボード複数を同期動作させるハンドラは第8章 》

当社製ADボード（IBM PC/AT互換機＝DOS/V機用）をVC、VB等で簡単に使用することのできるWINDOWS95/98/ME/NT2000/XP/VISTA用ハンドラ関数（DLL）です。

ADボードの基本機能が関数化されており、ユーザは自身の記述するメインルーチンの中から呼び出して使用することができます。すなわち、必要なパラメータ（動作条件）を変数に代入して本ハンドラを呼び出すだけで、ADサンプリングからパソコン本体メモリへのデータ転送まで、ポーリング・ブロック転送等により高速実行されます。

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）

拡張メモリ量 : 16MB以上

OS/コパイラ : WINDOWS9x、ME、NT、2000、XP、VISTA/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)、C++Builder

供給メディア : 各ADボード添付のサンプルディスク内。

対応ADボード : ADM-681/681aPCI（1枚のみ）

サンプリング : データ点数がFIFOメモリ容量までは各ボードの最高速度が可能、それ以上の場合はCPU等に依存。（P3/1GHz機で800KHz程度）

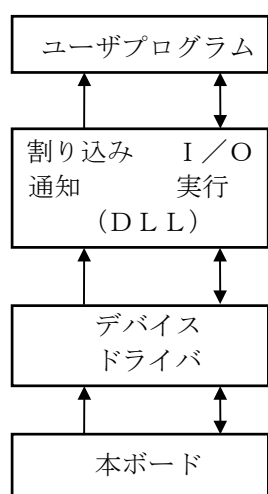
データ点数 : 拡張メモリ空容量（1語＝2バイト）

割り込み : 2000・XP・VISTAでは使用不可、9x・ME・NTでは**使用任意**。
（使用する場合は要リソース取得/1-5項参照。）

その他のボード : アドレス、割り込み番号が重複しない限り同時に使用可能。

図6-1. ソフトウェア構造

(WINDOWS2000・XP・VISTA の場合)

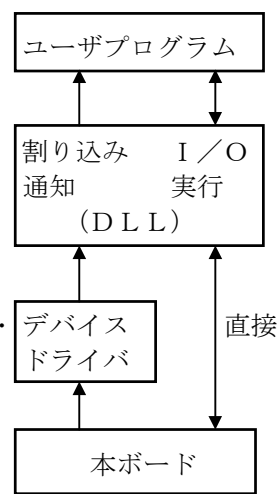


..... (サンプル添付)

..... (本ボード専用)

(本ボード専用)

(WINDOWS95・98・ME の場合)



6-2. サンプリグの様子とデータバッファ構造

ハンドラの使用法はとて簡単です。 具体的には6-4項以下で詳述しますが、要は使用するチャンネル数／サンプリグ点数／クロック値／トリガ（スタート）条件等のパラメータをセットして各関数を呼ぶだけで、ADデータはバッファに格納されて戻ってきます。

本ハンドラでは大別して以下に記す2形態のサンプリグ動作が可能です。

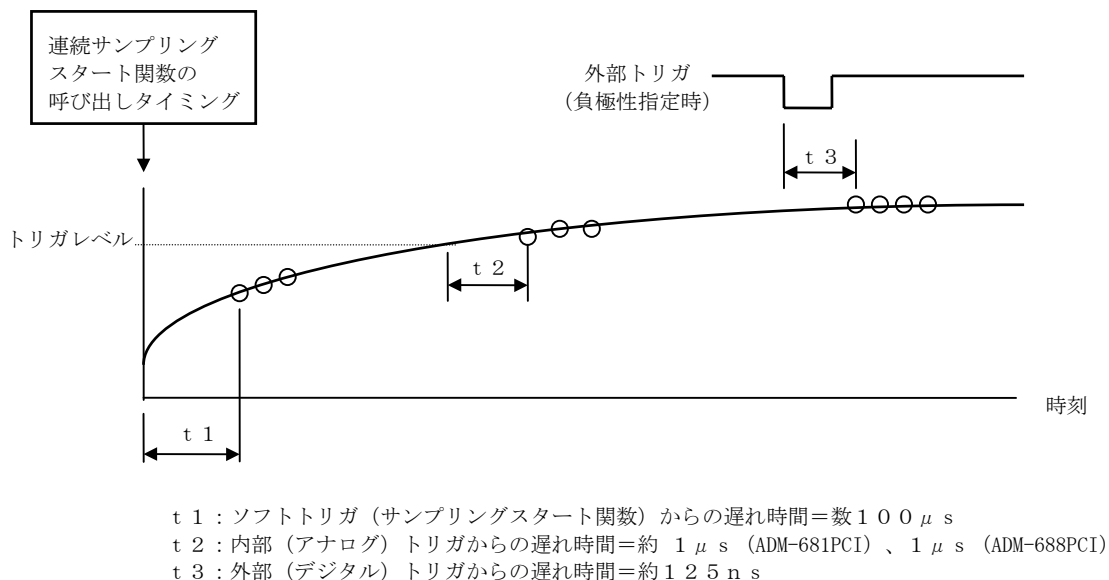
- (1) マニュアル（1回）サンプリグ・スキャン動作はメインルーチン上任意のプロセスで、指定チャンネル群入力を各1回だけAD変換し、結果をバッファに得ます。
- (2) 連続サンプリグ動作は指定チャンネル群に対し指定時間々隔（クロック周期）で指定回数だけAD変換を行い、結果をバッファに得ます。
動作開始となるトリガはソフト（即スタート）、内部（アナログ）、外部（デジタル）から選択できます。 クロック源も内蔵40MHz / 32.768MHz、外部TTL入力から選択できます。 いずれのクロック源も（ADボードの実行可能な速度以内で）任意の整数値で分周して利用します。 外部クロックを分周比＝1 / 1で使用すると、外部のイベントに同期したAD変換動作となります。

【注1】 指定チャンネル群： 各チャンネルを任意の順に並べた一群。

【注2】 内部（アナログ）トリガ： 指定レベル/極性とチャンネル0入力を比較して検出する。
外部（デジタル）トリガ： 専用TTL入力（TRG-IN）の指定極性エッジ。

【注3】 外部クロック源入力： 専用TTL入力（CLK-IN）の指定極性エッジ。 / 最高10MHz

図6-2A. 連続サンプリグの開始タイミング例



【注A】 ソフトトリガからの遅れ時間 t 1 はスタート関数の呼び出し、および同関数内の管理手続き実行時間を含むものです。 ADボード自体のソフトトリガ・ビットをセットしてからの遅れは最大125 ns です。

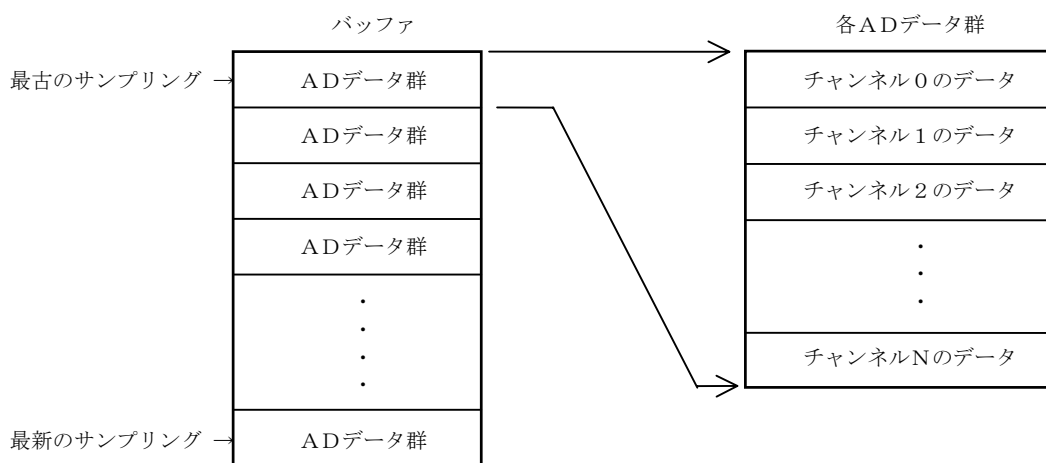
【注B】 内部（アナログ）トリガからの遅れ時間 t 2、および外部（デジタル）トリガからの遅れ時間 t 3 は各スタート関数の呼び出しと同関数内の管理手続き実行してトリガ待ち状態となつてからの時間です。 このトリガ待ち状態となるまでの時間は数100 μ s を要します。

ADデータバッファと格納の様子

サンプリングされたADデータは指定バッファに転送されます。

この（パソコン側メモリ上にある）バッファ内のデータ配置は（局所的には）ADボード上のFIFOメモリと同一イメージで、全チャンネルのADデータが1個のバッファに格納されます。

図6-2B. 全チャンネルのADデータが1個のバッファに入る



クロックによる連続サンプリング・ADデータ読み込み方法

自動モード

： 当ハンドラDLL側でバッファを確保し、これにADボード内で自動連続サンプリングされたADデータを自動転送する。 ユーザプログラムは別のタスクをマルチスレッドで併行することができる。

DLL管理バッファへのデータ読み込みは当ハンドラ自体がFIFOメモリの充満状態を示す【Half-Full】または【Not-Empty】フラグを監視して＜FIFOメモリ容量の半分＞または＜1 スキャン分＞単位で行う。 前者はブロック転送命令を使用するので速く、本ボードの最高速度を実現できる。 後者は1回サンプリングごとにADデータを評価して実行すべき処理があるような場合向きで、やや低速。

いずれの場合もユーザプログラム側からはステータス関数中のサンプリング済み点数を参照し、専用のデータ転送関数で受け取る。

《メッセージ》→ ADボードのサンプリング速度にプログラム側が追いつかなくなるか、ADボードが実行不可能な速いクロックを与えた場合に発生するサンプリングエラー（データロスト）、または指定数サンプリング終了がある。その他、ユーザプログラム中では割り込みイベントも利用できる。

マニュアルモード

： ユーザプログラム内にバッファを確保し、これにADボード内で自動連続サンプリングされたADデータを【連続サンプリングクロック】／【1回サンプリングスキャン終了】／【Half-Full】／【Not-Empty】／【トリガ発生】／【外部割り込み】／等による割り込みイベントまたはポーリングで＜FIFOメモリ容量の半分＞または＜1 スキャン分＞単位で行う。 サンプリング停止関数実行まで無限サンプリング動作となり、割り込みイベントを使用した場合はプログラム側の処理が追いつかないと例外発生によりユーザプログラムが停止する危険がある。

《メッセージ》→ 前記の各割り込みイベントによる。 サンプリングエラー、指定サンプリング終了などはユーザプログラム中でステータスをチェックする。

6-3. 使用準備

ボード上の設定・入力接続は標準設定【1-2項】とします。

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。

次に本機専用のハンドラ関数DLL、および汎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です。

■■■：ボード番号、△：バージョン番号

◆WINDOWS 95（98、ME）の場合

- (1) ¥MSCIENCERYHND_95¥Adm■■■¥DLL¥H■■■_95△.dll を ¥WINDOWS に、
- (2) ¥MSCIENCERYHND_95¥Adm■■■¥VXD¥D■■■_95△.vxd を ¥WINDOWS¥SYSTEM にコピーする。

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

- (1) ¥MSCIENCERYHND_NT¥Adm■■■¥DLL¥H■■■_NT△.dll を ¥WINDOWS¥SYSTEM32 に、
- (2) ¥MSCIENCERYHND_NT¥Adm■■■¥SYS¥D■■■_NT△.SYS を ¥WINDOWS¥SYSTEM32¥DRIVERS に、
- (3) ¥MSCIENCERYHND_NT¥Adm■■■¥D■■■Reg.exe を任意のフォルダにコピーする。

《NTにおける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■■■Reg）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・VISTA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用のWDM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。

- (1) ¥MSCIENCERYHND_2K¥Adm■■■¥DLL¥H■■■_2K△.dll を ¥WINDOWS¥SYSTEM32 にコピーする。

L a b V I E W等の完成アプリケーションから利用する場合は

本ハンドラ関数DLLを（L a b V I E W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HWN D O w n e rにN U L L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各種メッセージを受け取ることができませんから、この条件を踏まえた利用に制限されます。

■L a b V I E W用のサンプルV iを用意しました。（当社HPからダウンロードできます）

6-4. 関数仕様・エラーコード

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6-3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 : 【1】
- (2) サンプリング条件の設定 : 【2】～【6】チャンネル／トリガ／クロック／動作モード
- (3) スタートまたはトリガ待ち : 【7】即スタート／トリガ待ち（内部 or 外部）
- (4) ステータス評価 : 【8】連続サンプリング進捗状況
- (5) 以下・途中は任意
- (6) サンプリング中止 : 【17】データ転送に割り込み使用時はサンプリング終了後に必須。
- (7) ボードのフラグクリア : 【19】このあと、再度（2）or（3）に戻るときは必要。
- (8) ハンドラ終了 : 【9】後処理

表 6-4 A. 制御関数一覧

関数名	機能・内容	引数（パラメータ）等
1】AD_Open_ADSys	ボード、および本ハンドラの初期化	
2】AD_Set_SampCh	サンプリング実行チャンネル関連設定	実行チャンネル数
3】AD_Set_SampMode	サンプリングモード、イベントの設定	ADデータ転送先、方法
4】AD_Set_Trigger	トリガ関連設定（レンジトリガ以外）	トリガ源、レベル、モード
4】AD_Set_RangeTrigger	トリガ関連設定（レンジトリガ）	
5】AD_Set_Exclk	オプション、外部クロック源の設定	クロック源の周波数値
6】AD_Set_Clock	サンプリング・クロックの設定	クロック源、周期値、単位
7】AD_Start_Samp	サンプリング開始（トリガ待ち or 即）	サンプリング点数
8】AD_Get_Status	ステータス取得	サンプリング進捗状況など
10】AD_Read_DllData	DLL管理バッファからデータ読出し	ADデータ格納バッファ
11】AD_Read_DirectFifo	ADボードから直接にデータ読出し	ADデータ格納バッファ
12】AD_Get_OneScan	マニュアル（1回）サンプリング	
9】AD_Close_ADSys	本ハンドラの終了	
	《以上が基本、以下は補助的》	
13】AD_Out_Aux	汎用デジタル（ラッチ）出力	出力データ
14】AD_Inp_Aux	汎用デジタル（現在値）入力	入力データ
15】AD_Set_SampLoop	データバッファをリング状に設定	1 廻り後は上書き
16】AD_Set_Inpmode	データコード指定	
17】AD_Stop_Samp	サンプリング動作の（強制）中止	
18】AD_Read_RestData	ADボードの残りデータ読み込み	エラー停止の後
19】AD_Clear_Flags	ADボードのフラグクリア	ビット指定
20】AD_Set_Shc	外付サンプルホールド制御	オプション
21】AD_Set_ScanSpeed	アナログ入力スキャン速度指定	

【注】 WINDOWS 2000・XP・VISTA用に限り、関数名が異なります。
すなわち関数名中、文字“AD”の直後に対象ボード名（3桁番号）が含まれます。
例：AD681_Open_ADSys

重 要

本ハンドラを構成する関数は一定手順、または特定の組み合わせでのみ有効に動作します。 手順については本項先頭に記しましたが、組み合わせの点で注意すべきことは【3】で設定するサンプリングモードと【10】【11】のデータ転送関数です。 以下に連続サンプリング（波形取り込み）の典型的な例を示します。

- <例1> “自動モード”で“スタート操作”を実行すると、当ハンドラ自体が自動的にADボードのFIFOメモリからデータを“当ハンドラDLL管理バッファ”に読み込む。このとき、“ADデータ転送の参照フラグ”を《HALF-FULL》に設定してある場合は“FIFOメモリ容量”の半分単位でブロック転送、また《Not-Empty》に設定の場合は“1スキャン分”単位で転送する。 以上はバックグラウンドで自動実行され、指定データ点数に達すると自動停止する。
ユーザプログラムでは“サンプリング済みデータ点数”を参照して有効なデータを“当ハンドラDLL管理バッファ”から読み込む。
- <例2> 当ハンドラDLL管理バッファを“エンドレス・リング状”に設定し、例1と同様に操作する。 この場合、“当ハンドラDLL管理バッファ”の末尾と先頭が連結された構造となり、1廻り以後は上書きされる動作となる。
ユーザプログラム側からは“サンプリング済みデータ点数”の値を当バッファのポインタ（先頭＝0）として利用する。 連続サンプリングが“ストップ操作”まで無限に実行されるので長時間の監視システムや、低速のプリトリガ動作などに利用できる。
- <例3> “マニュアルモード”で“スタート操作”を実行すると、ADボードが連続サンプリングを開始しても当ハンドラは何もしない。 ユーザプログラムは自力でステータス関数をポーリングして“ボードの生ステータス”から《HALF-FULL》または《Not-Empty》を検出してデータ読み込みタイミングを知り、【11】AD_Read_DirectFifo()で読み込む。このとき、“ADデータ転送の参照フラグ”を《HALF-FULL》に設定してある場合は“FIFOメモリ容量”の半分単位でブロック転送、また《Not-Empty》に設定の場合は“1スキャン分”単位で読み込む。
当モードでは連続サンプリングが“ストップ操作”まで無限に実行されるので例2と同様なアプリケーションに適用できる。

補足説明

キーワード	関連関数など／【 】内の数字は関連する関数項目番号
“自動モード”	【3】連続サンプリング動作モード、 および割り込みイベントの設定関数で選択・指定する
“マニュアルモード”	
“ADデータ転送の参照フラグ”	
“スタート操作”	【7】連続サンプリング・スタート関数
“ハンドラDLL管理バッファ”	ユーザプログラムへは【10】AD_Read_DllData()で転送、 容量は【2】指定チャンネル数×【7】指定データ点数/ch
“FIFOメモリ容量”	【1】初期化のとき自動検出
“1スキャン分”	【2】指定チャンネル数を各1回サンプリング分
“サンプリング済みデータ点数”	【8】ステータスから得る
“エンドレス・リング状”	【15】DLL管理バッファをリング状・エンドレスに設定
“ストップ操作”	【17】連続サンプリング動作の強制停止関数の実行

《追伸》 ユーザプログラム上でADボード上のFIFOメモリ容量をソフト認識したいときは<例3>を“マニュアルモード” “ADデータ転送の参照フラグ”を《HALF-FULL》とし、ユーザプログラム上のバッファ容量を変化させて実行したとき、同容量がFIFOメモリの半分未満なら【11】AD_Read_DirectFifo()実行がエラーとなることから知ることができる

以下に各関数の仕様・詳細を記します。 【注】WINDOWS 2000・XP・VISTA用に限り、各関数名中“AD”の直後に3桁ボード番号が入ります。

【1】初期化

例：AD681_Open_ADSys

書式	int AD_Open_ADSys (HWND Owner)
引数	Owner : ウィンドウハンドル / 割り込み発生、サンプリング終了、データロス発生等の/ /メッセージングに使用。
戻り値	正常終了時 : ボードのID / 14H (ADM-681/681aPCI) エラー時 : エラーコード/負の値 (エラーコード表)
機能動作	プラグアンドプレイで設定したメモリアドレス、割り込み番号を本ハンドラが自動認識すると共に、本ボードのリセット、ハンドラ内部の参照テーブルやデータバッファを初期化する。 また、ADボード上のFIFOメモリ容量 (ハーフフル値) をチェックする。 【注1】 : ここで得るボードIDはPCIバスで定義されるSubsystemIDとは別物。 / 3-4項参照。

【注2】 メガFIFOモジュール搭載のときは同モジュール上のDIPスイッチで<ハーフフル値>を設定しますが、本ハンドラのサポートする範囲は0 (512語) ~ 5 (16K語) に限られます。
これは市販のFIFO素子で定義された元々の語意 (総容量の半分) を離れて、メモリの充満度を示すインジケータとなっています。

【2】サンプリング実行チャンネル関連の設定

書式	int AD_Set_SampCh (int no_ch, int scan_order[], int range[])
引数	no_ch : サンプリングを実行するチャンネル数。 scan_order [] : 各チャンネルのスキャン順。 (未使用 : 本ボードでは固定) range [] : 各チャンネルの入力レンジ番号。 (未使用 : 本ボードでは固定)
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)
機能動作	サンプリングを実行するチャンネル数、各チャンネルのスキャン順、入力レンジを設定する。 例えばno_ch : 5ならば、スキャン順番0~4のチャンネル群がサンプリング実行対象となる。 本ボードではスキャン順固定 (0, 1, 2, …… 15)、入力レンジはボード上のスイッチ設定。

【3】サンプリング動作モード、および割り込みイベントの設定

書式	int AD_Set_SampMode (int trs_trig, int buf_area, int intr_sw)
引数	trs_trig : ADデータ転送の参照フラグ / 0 : EMPTY解消、 1 : HALF-FULL buf_area : ADデータ転送先 / 0 : DLL管理バッファ、 1 : ユーザプログラム内バッファ (自動モード) (マニュアルモード) intr_sw : 割り込みイベント発生要因指定 / = 0 : 割り込み不使用、 = 1 : 連続サンプリング・クロック = 2 : 外部割り込み入力信号 (↑) = 3 : 外部割り込み入力信号 (↓) = 4 : トリガ発生 = 5 : 1回サンプリングスキャン終了 = 6 : Not-Empty = 7 : Half-Full
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)
機能動作	buf_areaでDLL管理バッファとした場合は当ハンドラ自体がDLLバッファ内にデータを読み込む自動モード、ユーザプログラム側からはステータス取得関数でサンプリング済みデータ点数を認識して専用の関数AD_Read_DLLDataで読み出す。 ユーザプログラム内バッファとした場合はユーザプログラム自体がステータス取得関数でボード上のFIFOメモリ状態を直接監視して、専用関数AD_Read_DirectFifoでユーザプログラム側が用意したバッファに読み込むマニュアルモード。 【注1】 : いずれのモードでも参照するフラグをtrs_trigで指定する。 【注2】 : いずれのモードでもADデータ転送自体に割り込みは使用されない。 また、intr_swは独立したイベント発生要因指定。 【注3】 : なおWINDOWS 2000用でのintr_swはダミー (割り込みサポートなし)

【4】トリガ関連の設定A（レンジトリガ以外の場合）

書式	int AD_Set_Trigger(int trg_mode, int trg_source, int trg_pol, int trg_level)
引数	<trg_mode :="" トリガ動作モード／0="" ポストトリガ<br="" 即トリガ、1=""></trg_mode> trg_source : トリガ源 / 0 : ソフト、1 : 内部（アナログ）、2 : 外部 trg_pol : トリガ極性 / 0 : 負エッジ、1 : 正エッジ 2 : 負レベル、3 : 正レベル trg_level : トリガレベル / 対応するADデータコード上位8ビットで指定する。
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）
機能動作	トリガ動作モード、源、極性、レベル等の設定。 なお、トリガ動作モードをポストトリガ、トリガ源を外部（デジタル）で極性をレベルに指定したときは帯域サンプリングとなる。（3-1項参照） アナログトリガ・チャンネルは【2】で設定されたスキャン先頭チャンネル（チャンネル0固定）

【4】' トリガ関連の設定B（レンジトリガの場合）

書式	int AD_Set_RangeTrigger(int trg_sel, int trg_level_hi, int trg_level_lo)
引数	trg_sel : トリガ動作モード／0 : アウトレンジ、1 : インレンジ 2 : デュアルスロープ（+）、3 : デュアルスロープ（-） trg_level_hi : トリガレベル上限値 trg_level_lo : トリガレベル下限値
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）
機能動作	レンジトリガの極性を含む動作モード、レベルの設定。（レンジトリガ以外は前記【4】を使用） アナログトリガ・チャンネルは【2】で設定されたスキャン先頭チャンネル。（チャンネル0固定）

【5】オプション、または外部クロック源周波数値の設定

書式	int AD_Set_Exclk (int exclk_freq)
引数	exclk_freq : オプション、または外部クロック源の周波数値（Hz 単位）
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）
機能動作	オプション、または外部クロック源を使用するとき、その周波数値を設定する。 後記【6】ad_set_clockで分周比によるサンプリングクロック指定、またはボードに標準搭載の内部クロック源を使用するときは必要ない。

【6】サンプリング・クロック値の設定

書式	int AD_Set_Clock(int clk_source, int set_mode, int *time_unit, int *clk_period)
引数	<p>clock_source : クロック源／0 : 内部クロック源0 (40.000MHz) 1 : 内部クロック源1 (32.768MHz) 2 : 内部クロック源2 (オプション) 3 : 内部クロック源3 (未使用) 4 : 外部クロック源 (有効極性=↓) 5 : 外部クロック源 (有効極性=↑)</p> <p>set_mode : クロック値の指定方法／0 : クロック周期の値、 1 : 分周比 time_unit : クロック周期の単位 / 0 : s、 1 : ms、 2 : μs、 3 : ns clk_period : クロック周期の値、または分周比</p>
戻り値	<p>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)</p>
機能動作	<p>サンプリングクロック値を設定する。 なお、設定できないクロック周期の値を指定すると設定可能な長い方の近似値が設定される。 また、クロック源に2, 4, 5を指定し、クロック周期の値で設定するときは前記【5】ad_set_exclkでクロック源周波数値を定義しておく。</p>

【7】サンプリング・スタート

書式	int AD_Start_Samp (int no_samp)
引数	no_samp : サンプリング・データ点数 (1チャンネル当りの点数)
戻り値	<p>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)</p>
機能動作	<p>サンプリングを実行する。 前記【4】トリガ設定でトリガ動作モードを《即トリガ》としてあるときは即スタート、また《ポストトリガ》が選択されているときはトリガ待ちスタートとなる。 前記【4】'で設定するレンジトリガのときは全てトリガ待ちスタートとなる。 なお前記【3】でサンプリングモードを自動に設定した場合、本DLLの管理するデータバッファサイズは当関数で指定する1チャンネル当りサンプリング・データ点数と前記【2】で指定されるサンプリング実行チャンネル数の積となる。(1データ=2バイト)</p>

【8】ステータス取得

書式	int AD_Get_Status (int *sampled, int status[])
引数	<p>sampled : サンプリング済みデータ点数 (1チャンネル当り) / 本DLL管理バッファ status[0] : ADボードのステータス1 / (3-12項) status[1] : ADボードのステータス2 / (未使用: 本ボードでは無い)</p>
戻り値	<p>正常終了時 : =0 : 連続サンプリング実行中、 =1 : 停止中。 エラー時 : エラーコード／負の値 (エラーコード表)</p>
機能動作	<p>当該時点でのサンプリング済みデータ点数 (本DLL管理バッファ内)、 およびADボードのステータス生データを得る。(FIFOメモリ状態を含む) / 3-12項参照。</p>

【9】本ハンドラの終了

書式	int AD_Close_ADSys (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)
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を行う。

【10】本DLL管理バッファからユーザプログラム内バッファにADデータを転送

書式	int AD_Read_DLLData(int no_data, data_pos, WORD *bufptr, int bufsize)
引数	no_data : 本DLL管理バッファから読み出すデータ点数 (1チャンネル当り) data_pos : 本DLL管理バッファから読み出すデータの先頭位置 (サンプリング順番号) bufptr : 転送先ADデータバッファのポインタ bufsize : 転送先ADデータバッファの大きさ (バイト指定=総データ数の2倍)
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)
機能・動作	サンプリングされたADデータが本DLL管理バッファに格納されているとき、任意の部分でユーザプログラム内のデータバッファに転送する。読み出すデータの末尾はステータス関数で得る<サンプリング済みデータ点数>の値以内でなくてはならない。 【注】当方法はサンプリング動作モード設定【3】で自動モードを指定したときだけ有効。

【11】ADボードからユーザプログラム内バッファに直接、ADデータを読み込む

書式	int AD_Read_DirectFifo (WORD *bufptr, int bufsize)
引数	bufptr : 転送先ADデータバッファのポインタ bufsize : 転送先ADデータバッファの大きさ (バイト指定=総データ数の2倍)
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)
機能・動作	ADボード上のFIFOメモリから直接、ユーザプログラム内のデータバッファに転送する。一般的に当関数は割り込みイベント (【3】参照)、またはFIFOメモリ状態をポーリングして実行タイミングを得る。サンプリング動作モード設定【3】でADデータ転送の参照フラグにNot-Emptyを指定したときは1回サンプリングスキャン分のADデータを転送する。また、Half-Fullを指定したときはADボード上のFIFOメモリ容量の半分を転送する。 【注】当方法はサンプリング動作モード設定【3】でマニュアルモードを指定したときだけ有効。

【12】マニュアル (1回) サンプリング・スキャン

書式	int AD_Get_OneScan (WORD *bufptr, int bufsize)
引数	bufptr : ADデータバッファのポインタ bufsize : ADデータバッファの大きさ (バイト指定=総データ数の2倍)
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)
機能・動作	前記【2】で指定したチャンネル群に対して各1回だけサンプリングを実行する。

【13】汎用デジタル（ラッチ）出力

書式	int AD_Out_Aux (int out_data)
引数	o u t _ d a t a : 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）
機能・動作	汎用（ラッチ）出力データの更新。 当出力ポートは電源投入（ハードウェア・リセット）時：00H となるが、ボードの制御部リセット（ソフト的リセット）では変化しない。

【14】汎用デジタル（現在値）入力

書式	int AD_Inp_Aux (void)
引数	なし
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）
機能・動作	汎用デジタル（現在値）入力データの読み込み。

【15】本DLL管理データバッファをリング状／エンドレスに設定

書式	int AD_Set_SampLoop (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）
機能・動作	本DLL管理のADデータバッファをリング状・エンドレスで使用するかどうかの設定。（トグル） リング状エンドレスに設定した場合、自動モードで本DLL管理データバッファが満杯になると次は同バッファ先頭を上書きする形となる。この時、ステータス取得関数【8】で得るサンプリング済みデータ点数の値も先頭値（ゼロ）に戻る。 本設定により当ハンドラは無限サンプリング・モードとなる。 長時間の監視システム等に利用できる。

【16】アナログ入力モードの設定

書式	int AD_Set_Inpmode (int inp_mode, int ad_code, int ad_reso)
引数	i n p _ m o d e : 未使用。／入力信号形式（ボード上のスイッチ設定） a d _ c o d e : ADデータコード指定／ 0 : バイナリ、 1 : 2の補数 a d _ r e s o : AD分解能指定／（未使用、ボード上のスイッチで12／14ビットを選択）
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）
機能・動作	アナログ入力信号形式、変換出力データコード（符号化の型）、およびAD変換分解能を指定する。

【17】 サンプリング動作の強制停止

書式	int AD_Stop_Samp (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)
機能・動作	サンプリングを強制的に中止する。 (ADボード上のFIFOバッファメモリに残りデータがあれば、それらは無効となる。) 本ハンドラではADデータ転送には割り込みを使用していないが、ユーザプログラム中で割り込みイベントを利用している場合は連続サンプリング終了後、必ず当関数を実行すること。

【18】 残りADデータの読み込み

書式	int AD_Read_RestData ()
引数	なし
戻り値	正常終了時 : =読み込んだ(残り)データ数 エラー時 : エラーコード／負の値 (エラーコード表)
機能・動作	データロス・エラー発生(当ハンドラが検出するとサンプリングを中止させる)したとき、ADボード上のFIFOバッファメモリに残データがあれば、これを当関数で読み込むことができる。 なお、前【17】サンプリングの強制停止を実行すると残りデータは読み出しできず、無効となる。

【19】 (ADボード) 各フラグのクリア

書式	int AD_Clear_Flags (int c__data)
引数	c__data : ビット0=1 なら FIFOメモリをクリア ビット2=1 なら ERR をクリア ビット5=1 なら TIM をクリア ビット6=1 なら INT をクリア ビット7=1 なら EOS をクリア / 3-12項参照 /
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)
機能・動作	ADボード上の各フラグをクリアする。

【20】 外付・同時サンプルホールド制御

書式	int Ad_Set_Shc (int shc)
引数	shc : 外付・同時サンプルホールド制御 / 0 : しない、 1 : 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)
機能・動作	複数チャンネル同時サンプルホールド回路を外付・接続する場合の各サンプリングスキャン開始タイミング制御。 shc=1のときは1 μ s遅らせることで先頭チャンネル(ch0)にかかるホールドステップ誤差を最小化するが、スキャン時間が1 μ s増加するため最高サンプリング速度が遅くなる。 shc=0のときは最高サンプリング速度で動作するが、スキャン先頭チャンネル(ch0)に数mvのホールドステップ誤差が発生する。(3-18項参照)

【21】アナログ入力スキャン速度指定

書式	int Ad_Set_ScanSpeed (int speed)
引数	speed : アナログ入力スキャン速度／ 0 : 高速モード、 1 : 低速モード、 2 : 予備モード2、 3 : 予備モード3。
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)
機能・動作	スキャン速度により (複数チャンネル使用時に) 得られるADデータの正確度が変わる。 詳しくは2-2項、および3-6項を参照。

表6-4B. 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。	AD_Open_AD Sys ()
- 2	ドライバファイルを検出できない。	AD_Open_AD Sys ()
- 3	ドライバファイルのバージョンが不適当。	AD_Open_AD Sys ()
- 4	初期化が未実行。	
- 5	連続サンプリング開始前の設定不足。	AD_Start_Samp ()
- 6	Ownerパラメータの不適当。	AD_Open_AD Sys ()
- 7	サンプリング実行チャンネル数 指定パラメータの不適当	no_ch
- 8	サンプリング・スキャン順 指定パラメータの不適当	scan_order []
- 9	アナログ入力レンジ 指定パラメータの不適当	range []
-10	ADデータ転送時の参照フラグ 指定パラメータの不適当	trs_trig
-11	ADデータ転送先 指定パラメータの不適当	buf_area
-12	割り込みイベント発生要因 指定パラメータの不適当	intr_sw、
-13	トリガ動作モード 指定パラメータの不適当	trg_mode
-14	トリガ源 指定パラメータの不適当	trg_source
-15	トリガ極性 指定パラメータの不適当	trg_pol
-16	トリガ源選択 指定パラメータの不適当	trg_sel
-17	トリガレベル (レンジトリガ以外) 指定パラメータの不適当	trg_level
-18	トリガレベル (レンジトリガ) 指定パラメータの不適当	trg_level_hi/_lo
-19	トリガレベル上限値・下限値の大小関係不適当	trg_level
-20	クロック源 指定パラメータの不適当	clk_source
-21	サンプリング・クロックの指定方法 パラメータの不適当	set_mode
-22	クロック周期の単位 指定パラメータの不適当	time_unit
-23	クロック周期の値、または分周比 指定パラメータの不適当	clk_period
-24	アナログ入力信号形式 指定パラメータの不適当	inp_mode
-25	ADデータ・コード 指定パラメータの不適当	data_code
-26	AD変換分解能 指定パラメータの不適当	ad_reso
-27	外部クロック源周波数を設定していない。	AD_Set_Clock ()
-28	クロック分周比 指定パラメータの不適当	clk_period
-29	ADデータバッファ・サイズが小さすぎる。	bufsize
-30	1回サンプリングスキャン実行時、チャンネル設定の未実行	AD_Get_OneScan ()
-31	bufptrエラー	bufptr
-32	サンプリングされたADデータ数を超えて読もうとした。	AD_Read_Data ()
-33	割り込みリソースが割り当てられていない。	AD_Set_SampMode ()
-34	割り込み設定エラー (重複など)	AD_Set_SampMode ()
-35	メモリ確保エラー	
-36	ユーザプログラム内のADデータバッファが指定されている。	AD_Read_Dlldata ()
-37	本ハンドラDLL管理ADデータバッファが指定されている。	AD_Read_DirectFifo ()
-38	連続サンプリング中なのに各種条件設定等をしようとした。	
-39	現在、サンプリング中である。	AD_Read_RestData ()

□■ WINDOWSハンドラ使用例 ■□

表 6-4 C. ハンドラ使用例

ファイル名	動作の概要	備考
Sample.DSW	チャンネル数、サンプリングクロック、サンプリング点数を指定し、即トリガ（即スタート）での動作例。 その他にADデータ転送先、割り込み要因の指定。	VC++ (5.0) 用
Sample.VBP		VB (5.0) 用

□■ Cハンドラ使用上の注意・制限 ■□

- (1) 本ハンドラは不適切な使い方をするとエラーを返してきますから、原因が除去されるようにデバッグしてください。特に問題を起こしやすい点は、

- ◆ハード（ADボード）とソフト（ハンドラ）設定の不整合。
- ◆ADボードと他のハード（非PnP対応ボード）の設定重複。 : メモリアドレス
割り込み番号

特にハード同志の重複設定による不具合は本ハンドラで検出できませんから、システムの構築時に充分な確認が必要です。

- (2) 本ハンドラは【AD_Open_ADSys ()】で使用開始、
【AD_Close_ADSys ()】で使用終了とします。

特に終了手続きは本ハンドラ内で確保したメモリの開放等を含むので重要です。

- (3) 本ハンドラの管理する割り込みは1系統のみです。

サンプリング実行時の《ボード上のFIFOバッファ》～《パソコン側データバッファ》間データ転送の実行タイミング検出には使用しておらず、単独の指定イベントによるメッセージ通知機能だけです。

- (4) データロスト・エラー

ADボードの自動サンプリング動作でFIFOバッファメモリにADデータが流入する速度よりパソコン側から読み出す速度が遅く、FIFOバッファメモリが満杯になったうえに次のデータが書き込まれようとする、そのデータはこぼれ（失われ）てしまいます。

これがデータロスト・エラーで、構築されたシステム全体の処理速度を超えたクロック値を指定して実行したときに起ります。なおデータロスト・エラーの発生を本ハンドラが検出すると連続サンプリングを強制停止させますが、それ以前のデータ（FIFOバッファメモリ中にある）は残りデータとして専用関数【18】で有効に読み出すことができます。

システムパフォーマンスはCPUを含むパソコン本体の実行速度や周辺機器の状態に左右されますが、WINDOWS95/98で（ユーザが記述する）本ハンドラの応用プログラムがサンプリング実行&データ読み込み（ブロック転送）に専念している時は本ADボード自体の最高サンプリング速度が実現可能です。WINDOWS2000/NTの場合は構造上、本ボードへのアクセス過程に時間をとられるため最高速度500KHzサンプリング程度です。

- (5) 直接の操作を行うとき

本ハンドラを介さずADボードに直接の操作（メモリ読み書き）を行うと、本ハンドラの管理が行き届きませんから不本意な動作となることがあります。

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) デジタル入出力 ◆ 本ボードのTTL入力（外部割り込み、および汎用ビット）に接続できる信号源はTTL（LS、CMOS等の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。（次ページ参照）
- (4) アナログ入力 ◆ 過電圧入力保護：±3.5V以内。
◆ 複数チャンネル使用時は各信号源のGND間電位差に注意。

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。

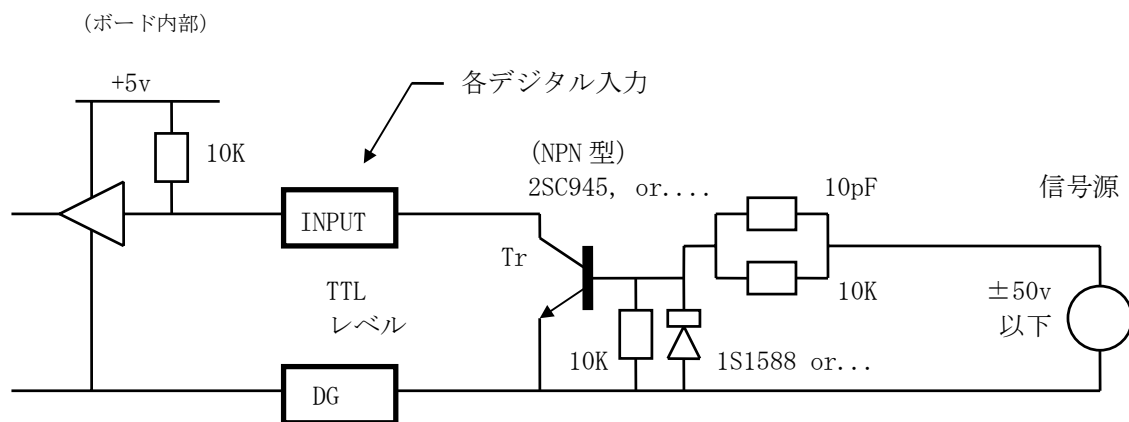
QAリクエスト時には当プログラムの実行結果をレポートしてください。

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3日間以内に完了・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。 2007年3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. アナログ入力範囲の再調整

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入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、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。
なお、実際に使用する入力範囲に設定しておきます。
- ② 本ボードをパソコン本体または拡張I/Oボックスに装着・インストールします。
当作業の詳細については1-5項に記されています。
（以上、通常の使用状態）
- ③ 図1-6のように、本ボード全チャンネルのアナログ入力を基準電圧源に接続します。
デジタル入出力の接続は不要です。
- 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。

== 操作 ==

- ⑤ 電源を投入、MS-DOSシステムを立ち上げます。再調整に使用するプログラムは試運転でも使用した“68■QB2”です。>68■QB2【ENTER】でプログラムが走り始めます。最初にI/Oベースアドレスの入力を要求されますから1-5項で調査した4桁のHex値を入力すると以下、操作メニューとなります。

■=1: ADM-681aPCI用

■=8: ADM-688PCI用

- ⑥ 外部基準電圧源を操作し、本ADボード設定入力範囲の下限付近でオフセット調整を、上限付近でゲイン調整を、目標値を得よう交互に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入力範囲で実施してください。なお、本機の標準出荷時はAモード±10V入力範囲で最適調整されています。

以後の操作は、

【SPACE】キー： 中断／再開

【ESC】キー： 終了

本プログラムではADデータを16進数（Hex）で表示します。

= ADM-681aPCIの再調整 =

調整はアナログ入力範囲の《最大値付近》と《0v》の2点間で行います。本機のアナログ回路/AD変換器は±0.013%FS以内の非直線性を持っており、2点間調整だけで設定入力範囲の全域に渡って±0.033%FSの精度(=相対正確度)を実現する能力を持っています。

この相対正確度(=較正可能限度)に較正に使用した測定器の正確度を積算した値が絶対正確度です。また高精度部品の使用により調整を実施した入力範囲以外に設定を変更しても、レンジ間誤差は±0.06%FS以内です。当社の製造調整は0.015%FSの基準電圧発生器を使用して常温で行っています。したがって当時点での絶対正確度は、

★最終調整±10v範囲/Aモード/14bitモードで 0.048%FS(低速モード)、
0.068%FS(高速モード)、

★その他の入力範囲では各々の値に0.06%FSを加算した値になります。

= (1) 12ビットでの調整手順 =

①AD変換部のオフセット調整 : ジャンパJPZをZ側に設定し、入力電圧が0vのとき、
(バイポーラ入力範囲のとき) AD変換値が800HとなるようTM0を調整します。

①AD変換部のオフセット調整 : ジャンパJPZをZ側に設定し、入力電圧が0vのとき、
(ユニポーラ入力範囲のとき) AD変換値が000HとなるようTM0を調整します。

②入力モジュールを含む : ジャンパJPZをM側に設定し、入力電圧が0vのとき、
総合的なオフセット調整 AD変換値が①の結果と同値になるよう、搭載されている
入力モジュール上のオフセットトリマを調整します。

③ゲイン調整 : 入力電圧が最大値付近のとき、AD変換値が整合した値
となるようにTM1を調整します。

[②オフセット調整 ③ゲイン調整] を2～3回繰り返して所定の精度に追い込みます。

【注】電源ONから調整実施までに1分以上のウォームアップ時間をとって下さい。

表7-3 A. 12ビット/Aモードの調整対象・目標

設定入力範囲 →→		±10v	±5v	±2.5v	0～+5v	0～+10v
オフセット調整	基準入力	0v	0v	0v	0v	0v
	設定目標	800H	800H	800H	000H	000H
	トリマ	TM0	TM0	TM0	TM0	TM0
ゲイン調整 (スケール)	基準入力	+10v	+5v	+2.5v	+5v	+10v
	設定目標	FD0H	FD0H	FD0H	FA0H	FA0H
	トリマ	TM1	TM1	TM1	TM1	TM1

表7-3 B. 12ビット/Bモードの調整対象・目標

設定入力範囲 →→		±10v	±5v	±2.5v	0～+5v	0～+10v
オフセット調整	基準入力	0v	0v	0v	0v	0v
	設定目標	800H	800H	800H	000H	000H
	トリマ	TM0	TM0	TM0	TM0	TM0
ゲイン調整 (スケール)	基準入力	+9.99512v	+4.99756v	+2.49878v	+4.99878v	+9.99756v
	設定目標	FFFH	FFFH	FFFH	FFFH	FFFH
	トリマ	TM1	TM1	TM1	TM1	TM1

＝ (2) 14ビットでの調整手順 ＝

- ①AD変換部のオフセット調整 : ジャンパJPZをZ側に設定し、入力電圧が0vのとき、
(バイポーラ入力範囲のとき) AD変換値が2000HとなるようTM0を調整します。
- ①AD変換部のオフセット調整 : ジャンパJPZをZ側に設定し、入力電圧が0vのとき、
(ユニポーラ入力範囲のとき) AD変換値が0000HとなるようTM0を調整します。
- ②入力モジュールを含む : ジャンパJPZをM側に設定し、入力電圧が0vのとき、
総合的なオフセット調整 AD変換値が①の結果と同値になるよう、搭載されている
入力モジュール上のオフセットトリマを調整します。
- ③ゲイン調整 : 入力電圧が最大値付近のとき、AD変換値が整合した値
となるようにTM1を調整します。

[②オフセット調整 ③ゲイン調整] を2～3回繰り返して所定の精度に追い込みます。

【注】電源ONから調整実施までに10分以上のウォームアップ時間をとって下さい。

表7-3C. 14ビット/Aモードの調整対象・目標

設定入力範囲 →→		±10v	±5v	±2.5v	0～+5v	0～+10v
オフセット調整	基準入力	0v	0v	0v		
	設定目標	2000H	2000H	2000H		
	トリマ	TM0	TM0	TM0		
ゲイン調整 (スケール)	基準入力	+10v	+5v	+2.5v		
	設定目標	3388H	3388H	3388H		
	トリマ	TM1	TM1	TM1		

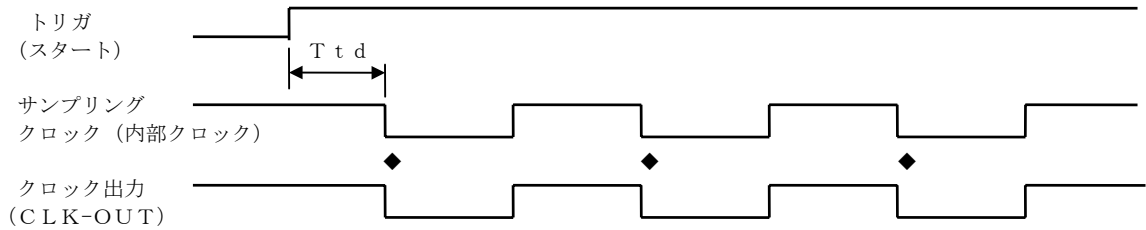
表7-3D. 14ビット/Bモードの調整対象・目標

設定入力範囲 →→		±10v	±5v	±2.5v	0～+5v	0～+10v
オフセット調整	基準入力	0v	0v	0v		
	設定目標	2000H	2000H	2000H		
	トリマ	TM0	TM0	TM0		
ゲイン調整 (スケール)	基準入力	+9.99878v	+4.99939v	+2.49969v		
	設定目標	3FFFH	3FFFH	3FFFH		
	トリマ	TM1	TM1	TM1		

7-4. 外部制御信号・タイミング等

クロック入出力

図7-4 A. 内部クロック源を使用する場合

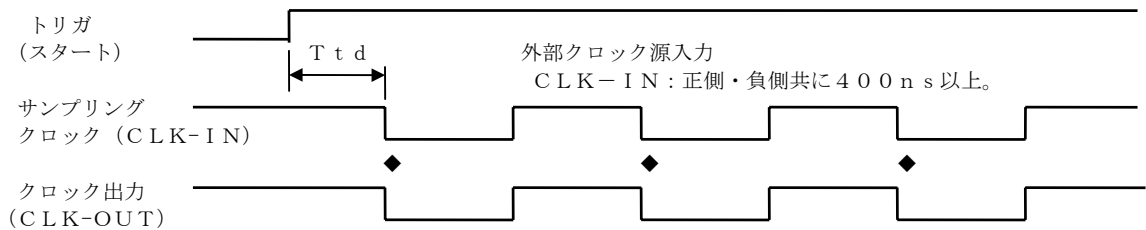


◆：クロック出力の有効エッジは常に立下り。

T t d：トリガ認識から最初のクロック有効エッジまでの最大遅れ時間（最大125ns）

実際のサンプリング実行は内部クロックの各有効エッジから50ns以内に開始。

図7-4 B. 外部クロック源を非分周（1／1）で使用する場合

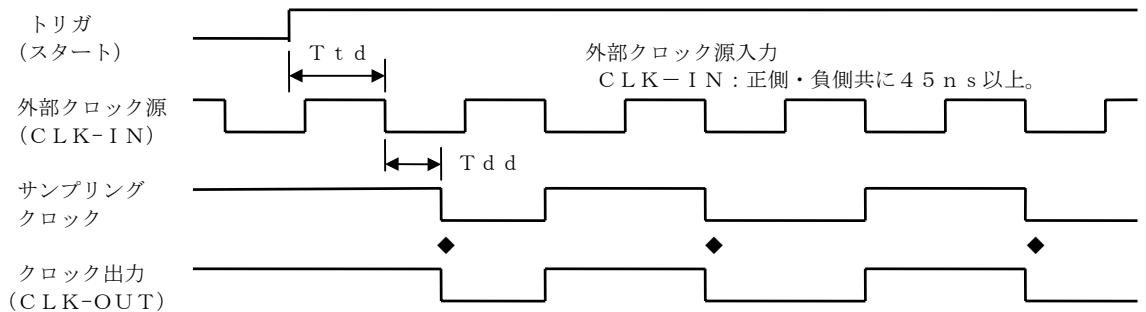


◆：クロック出力の有効エッジは常に立下り。

T t d：トリガ認識から最初のクロック有効エッジまでの最大遅れ時間（外部クロック源1周期）

実際のサンプリング実行は外部クロックの各有効エッジから100ns以内に開始。

図7-4 C. 外部クロック源を（任意に）分周して使用する場合



◆：クロック出力の有効エッジは常に立下り。

T t d：トリガ認識から最初のクロック有効エッジまでの最大遅れ時間（外部クロック源1周期）

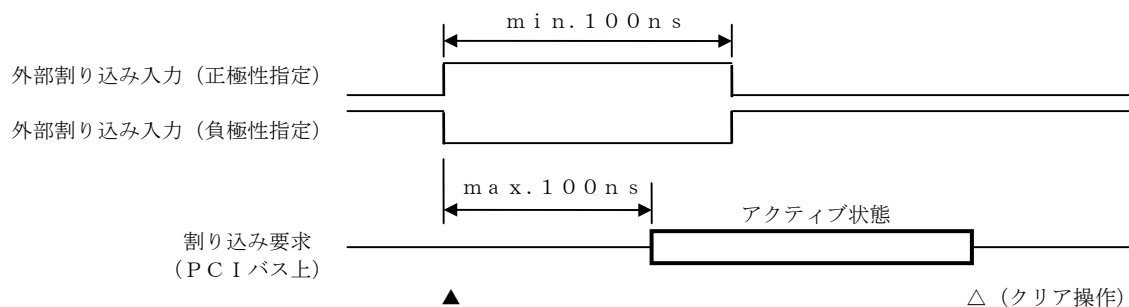
実際のサンプリング実行は外部クロックの各有効エッジから100ns以内に開始。

T d d：分周開始の遅れ時間（最大100ns）

外部割り込み入力

 (許可されている場合／3-14項)

図7-4D. 外部割り込み入力 ～ 割り込み受け付け



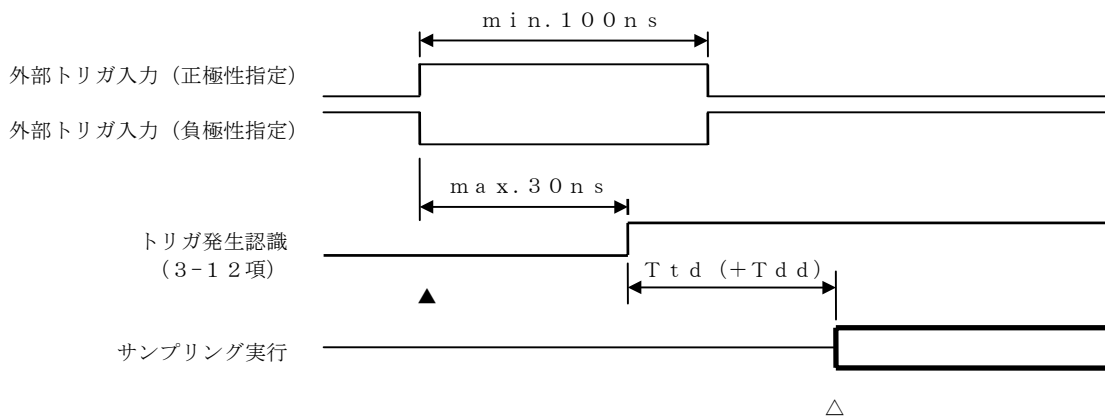
▲: 外部割り込み信号入力 (INT-IN) の有効エッジ。

△: 割り込みクリア操作タイミング。(通常、デバイスドライバ内で実行する)

外部トリガ入力

 (許可されている場合／3-10項)

図7-4E. 外部トリガ ～ 連続サンプリング開始



▲: 外部トリガ信号入力 (TRG-IN) の有効エッジ。

△: 連続サンプリングの開始タイミング。

Ttd: 図7-4A, B, C参照。

Tdd: 図7-4C参照。

LS型TTL素子の基本規格

 : 本機のデジタル入出力素子
入力: $I_{(IL)} = -0.4 \text{ mA}$ …… Lレベル (+0.4V ~ 0V) 印加時の流れ出し電流 $I_{(IH)} = 20 \mu\text{A}$ …… Hレベル (+2.7V ~ +5V) 印加時の流れ込み電流出力: $I_{(OL)} = 8 \text{ mA}$ …… Lレベル (+0.4V ~ 0V) 出力時の流れ込み可能電流 $I_{(OH)} = -0.4 \text{ mA}$ …… Hレベル (+2.7V ~ +5V) 出力時の流れ出し可能電流

7-5. 付録 (WINDOWS 2000・XP・VISTA 対応について)

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、VISTA

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	ADM-681aPCI		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	(ver)	
	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			

第8章. 複数ボード（マスタスレーブ動作）対応 WINDOWSハンドラ

当社製ADボード（IBM PC／AT互換機=DOS／V機用）をVC、VB等で簡単に使用することのできる、複数ボード（マスタスレーブ動作）対応のWINDOWS2000・XP、VISTA、98、ME用ハンドラDLL（+ドライバ）です。

ADボードの基本機能が関数化されており、ユーザは御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。すなわち必要なパラメータ（動作条件）を変数に代入して本ハンドラを呼び出すだけでADサンプリングからパソコン本体メモリへのデータ転送まで、ポーリング・ブロック転送等により高速実行されます。

8-1. システム構成・ソフトウェア構成

パソコン本体 : IBM PC／AT互換機（含む98NX機）

拡張メモリ量 : 16MB以上

OS／コンパイラ : WINDOWS9x、ME、または2000、XP、VISTA／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), C++Builder

供給メディア : 各ADボード添付のサンプルディスク内。

対応ADボード : ADM-681／681aPCI（最大8枚まで制御可能）

サンプリング : FIFOバッファメモリ容量まではボードの最高速度が可能、
ボード側からの実用的な最高データ転送速度は800KHz程度
(Pentium 1GHz)。

上記速度は当ハンドラDLLがボードからのADデータ読み込みに専念したときに期待できるPCIバス上の実用的な最高データ流速です。したがって本ボード1枚のとき $1.25\mu\text{s}/\text{ch}$ 、さらにボードの枚数に比例してリアルタイムに追いつく最高サンプリング速度は低下します。

但し読み込み速度が低下しても、各ボードの搭載するFIFOメモリ容量まではボード自体の最高サンプリング速度 $1\mu\text{s}/\text{ch}$ で取得したADデータを（遅れながらも）有効に読み込むことができます。

■ FIFOメモリは最大32M語まで増設可能です。 ■

データ点数 : 拡張メモリ空容量（1語＝2バイト）

割り込み : 使用不可。

その他のボード : I／Oアドレスが重複しない限り同時に使用可能。

8-2. サンプリングの様子とデータバッファ構造

単一ボード用ハンドラと同様です。【6-3項を御覧ください】

但し本ハンドラDLLの管理するデータバッファは使用するボードと同数作成されます。

各データバッファは単一ボード用ハンドラの場合と同一イメージ【6-3項／図6-2B】で、各ボード分のADデータが各1個の専用バッファに格納されます。

8-3. 使用準備

ボード上の設定・入力接続は標準設定【1-2項】【3-16項】とします。

このとき、ボード番号設定スイッチSW-BNはマスタを【0】に、以下、スレーブボードを各【1】～【7】に順次設定します。

続いて第4章（4-1項）に従い、本ボード関連ソフトをインストールします。

次に本機専用のハンドラ関数DLL、および汎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です。

◆WINDOWS 2000・XP・VISTA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用のWDMドライバ“DMS_PCL.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) 本ハンドラ関数DLL（M681_2K.dll）を所定のフォルダ

¥WINDOWS¥SYSTEM32 にコピーして御使用ください。

／CDROM：¥Mscience¥Hnd_2k¥Adm681¥Mst_Slv¥D11 から／

◆WINDOWS 98、MEの場合 --- 【注】2001-8月版以降のCDROMでサポート。

当社提供のボードインストール環境は通常、WINDOWS 95との互換性をとるため、単一ボード用ハンドラをVXD型ドライバで使用する前提で設定されています。

本ハンドラDLLはWINDOWS 2000と同一のWDM型ドライバを使用するため、WINDOWS 2000と同一のインストール作業【1-5項】が必要です。もし既にWINDOWS 95・98の方法でインストール済みの場合は以下の要領で一旦削除してから、あらためてWINDOWS 2000と同一の方法でインストールしてください。

- (1) WINDOWSのデバイスマネージャで表示される本ボードの関連付けを削除する。
- (2) ¥Windows¥Inf¥Others フォルダに以下の定義ファイルがあれば、これらを削除する
Micro Science ms_pci.inf、Micro Science Co.,Ltd.DMS_PCI.INF

なお ¥Windows¥Inf フォルダは隠しフォルダとなっているので、エクスプローラの
<表示>→<フォルダオプション>→<表示>にある“詳細設定”内の
“ファイルとフォルダの設定”で“全てのファイルを表示する”に変更して作業する。

- (3) WINDOWS 2000と同一の方法・手順で本ボードをインストールする。

- (4) 念のため、デバイスマネージャで本ボードの登録を確認します。

- (5) 本ハンドラ関数DLL（M681_2K.dll）を所定のフォルダ¥WINDOWS¥SYSTEM32 にコピーして御使用ください。

／CDROM：¥Mscience¥Hnd_2k¥Adm681¥Mst_Slv¥D11 から／

8-4. 関数仕様、エラーコード

以下に各関数の仕様・詳細を記します。

ADM-681aPCI用：■＝1

ADM-688 PCI用：■＝8

【1】初期化

書式	int AD68■_Open_ADSys (HWND Owner, int num_board)
引数	Owner : ウィンドウハンドル／割り込み発生、サンプリング終了、／ データロス発生等のメッセージングに使用。／ num_board : 使用する本ボード数。(1～8)
戻り値	正常終了時 : ボードのID＝14H (ADM-681／681aPCI) エラー時 : エラーコード／負の値 (エラーコード表8-4参照)
機能動作	プラグアンドプレイで設定したI/Oアドレス、割り込み番号を本ハンドラが自動認識すると共に、本ボードのリセット、ハンドラ内部の参照テーブルやデータバッファを初期化、またFIFOメモリ素子の＜ハーフフル値＞を認識する。なお、メガFIFOモジュール使用時の＜ハーフフル値＞は容量の半分ではなく、同モジュール上のDIPスイッチ設定値0（512語）～5（16K語）。 【注】：ここで得るボードIDはPCIバスで定義されるSubsystem IDとは別物。／3-4項参照。

【2】サンプリング実行チャンネル関連の設定／（ボードごとに実行）

書式	int AD68■_Set_SampCh (int board_num, int no_ch, int scan_order[], int range[])
引数	board_no : 対象ボード番号。(0～7)／本関数は各ボードごとに行う。 no_ch : サンプリングを実行するチャンネル数(1～16)／必ず全ボード同一値。 scan_order[] : 各チャンネルのスキャン順。(未使用、本ボードでは固定) range[] : 各チャンネルの入力レンジ番号。(未使用、本ボードでは固定)
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8-4参照)
機能動作	サンプリングを実行するチャンネル数、入力レンジを設定する。 スキャン順番（バッファメモリ中のADデータ並び）は若いチャンネル番号からの順に固定されている。 入力レンジはボード上のスイッチ設定。

★ボード番号： 1枚だけのボードを使用する場合はボード上の番号設定スイッチSW-BNを0に設定すること。
複数ボードを使用する場合はマスタ機の番号設定スイッチSW-BNを0とし、以下、スレーブ機の
同スイッチを順次1～7に設定する。

【3】FIFOからのサンプリングADデータ転送モード、および割り込みイベントの設定

書式	int AD68■_Set_SampMode (int trs_trig, int buf_area, int intr_sw)
引数	trs_trig : ADデータ転送の参照フラグ／0 : EMPTY解消、 1 : HALF-FULL buf_area : ADデータ転送先 / 0 : DLL管理バッファ、 1 : ユーザプログラム内バッファ (自動モード) (マニュアルモード) intr_sw : 割り込みイベント発生要因指定／（当ハンドラではダミー）
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8-4参照)
機能動作	buf_areaでDLL管理バッファとした場合は当ハンドラが管理するDLLバッファにデータを読み込む自動モード、ユーザプログラム側からはステータス取得関数でサンプリング済みデータ点数を認識して専用の関数AD681_Read_DLLDataで読み出す。 ユーザプログラム内バッファとした場合はユーザプログラム自体がステータス取得関数でボード上のFIFOメモリ状態を直接監視して、専用関数AD_Read_DirectFifoでユーザプログラム側バッファに読み込むマニュアルモード。 【注1】：いずれのモードでも参照するフラグをtrs_trigで指定する。 【注2】：メガFIFOモジュール使用時の＜HALF-FULL＞は半分ではなく、スイッチ設定値。 【注3】：当ハンドラは割り込みをサポートしていないのでintr_swはダミー。

【４】トリガ関連の設定A（レンジトリガ以外の場合）

書式	int AD68■_Set_Trigger(int trg_mode, int trg_source, int trg_pol, int trg_level)
引数	trg_mode : トリガ動作モード／0 : 即トリガ、1 : ポストトリガ trg_source : トリガ源 / 0 : ソフト、1 : 内部（アナログ）、2 : 外部デジタル trg_pol : トリガ極性 / 0 : 負エッジ、1 : 正エッジ 2 : 負レベル、3 : 正レベル trg_level : トリガレベル（アナログ）／対応するADデータコード上位8ビットで指定する
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8-4参照）
機能動作	トリガ動作モード、源、極性、レベル等の設定。 なお、トリガ動作モードをポストトリガ、トリガ源を外部（デジタル）で極性をレベルに指定したときは帯域サンプリングとなる。／（3-1項参照） トリガ検出対象は（複数ボード使用のときはマスタボードの）チャンネル0。

【４】' トリガ関連の設定B（アナログ・レンジトリガ使用の場合）

書式	int AD68■_Set_RangeTrigger(int trg_sel, int trg_level_hi, int trg_level_lo)
引数	trg_sel : アナログトリガ・モード／0 : アウトレンジ、1 : インレンジ 2 : デュアルスロープ（+）、3 : デュアルスロープ（-） trg_level_hi : トリガレベル上限値 trg_level_lo : トリガレベル下限値
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8-4参照）
機能動作	レンジトリガの極性を含む動作モード、レベルの設定。（レンジトリガ以外は前記【4】を使用） アナログトリガ検出対象は（複数ボード使用のときはマスタボードの）チャンネル0。

【５】オプション、または外部クロック源周波数値の設定

書式	int AD68■_Set_Exclk (int exclk_freq)
引数	exclk_freq : オプション、または外部クロック源の周波数値（Hz 単位）
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8-4参照）
機能動作	オプション、または外部クロック源を使用するとき、その周波数値を設定する。 後記【6】ad_set_clockで分周比によるサンプリングクロック指定、またはボードに標準搭載の内部クロック源を使用するときは必要ない。

【6】サンプリング・クロック値の設定

書式	int AD68■_Set_Clock(int clk_source, int set_mode, int *time_unit, int *clk_period)
引数	<p>clock_source : クロック源／0 : 内部クロック源0 (40.00MHz) 1 : 内部クロック源1 (32.768MHz) 2 : 内部クロック源2 (オプション) 3 : 内部クロック源3 (未使用) 4 : 外部クロック源 (有効極性=↓) 5 : 外部クロック源 (有効極性=↑)</p> <p>set_mode : クロック値の指定方法／0 : クロック周期の値、 1 : 分周比 time_unit : クロック周期の単位 / 0 : s、 1 : ms、 2 : μs、 3 : ns clk_period : クロック周期の値、または分周比</p>
戻り値	<p>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8-4参照)</p>
機能動作	サンプリングクロック値を設定する。なお、設定できないクロック周期の値を指定すると設定可能な長い方の近似値が設定される。また、クロック源に2, 4, 5を指定し、クロック周期の値で設定するときは前記【5】ad_set_exclkでクロック源周波数値を定義しておく。

【7】サンプリング・スタート

書式	int AD68■_Start_Samp (int no_samp)
引数	no_samp : トリガ後のサンプリング・データ点数 (1チャンネル当りの点数)
戻り値	<p>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8-4参照)</p>
機能動作	<p>サンプリングを実行する。</p> <p>前記【4】トリガ設定でトリガ動作モードを《即トリガ》としてあるときは即サンプリングスタート、また《ポストトリガ》が選択されているときはトリガ待ちスタートとなる。</p> <p>前記【4】で設定するレンジトリガのときは全てトリガ待ちスタートとなる。</p> <p>なお前記【3】でサンプリングモードを自動に設定した場合、本DLLで確保されるデータバッファサイズ (1ボード分) は当関数で指定する1チャンネル当りサンプリング・データ点数と前記【2】で指定されるサンプリング実行チャンネル数の積となる。 (1データ=2バイト)</p>

【8】ステータス取得／ (マスタボード)

書式	int AD68■_Get_Status (int *sampled, int status[])
引数	<p>sampled : サンプリング済みデータ点数 (1チャンネル当り) / DLLバッファ status[0] : ADボードのステータス1 / (3-12項) status[1] : ADボードのステータス2 / (未使用: 本ボードでは無い)</p>
戻り値	<p>正常終了時 : =0 : 連続サンプリング実行中、 =1 : 停止中。 エラー時 : エラーコード／負の値 (エラーコード表8-4参照)</p>
機能動作	<p>当該時点でのサンプリング済みデータ点数 (DLL管理バッファ内／ソフト認識)、およびマスタボードのステータス生データ (FIFOメモリ状態等) を得る。</p> <p>マニュアルモードのときはステータス生データのみ有効。</p>

【9】本ハンドラの終了

書式	int AD68■_Close_ADSys (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8-4参照）
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を行う。

【10】DLL管理バッファからユーザプログラム内バッファにADデータを転送（ボードごと）

書式	int AD68■_Read_DLLData(int board_no, int no_data, int data_pos, WORD *bufptr, int bufsize)
引数	board_no : 対象ボード番号。／0～7（複数ボード使用のときはボードごとに行う） no_data : DLL管理バッファから読み出すデータ点数（1チャンネル当り） data_pos : DLL管理バッファから読み出すデータの先頭位置（サンプリング順番号） bufptr : 転送先ADデータバッファのポインタ bufsize : 転送先ADデータバッファの大きさ（バイト指定＝総データ数の2倍）
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8-4参照）
機能動作	サンプリングされたADデータがDLL管理バッファに格納されているとき、任意の部分をユーザプログラム内のデータバッファに転送する。読み出すデータの末尾はステータス関数で得る<サンプリング済みデータ点数>の値以内でなくてはならない。 【注】当方法はサンプリング動作モード設定【3】で自動モードを指定したときだけ有効。

【11】FIFOからユーザプログラム内バッファに直接、ADデータを読み込む／（ボードごと）

書式	int AD68■_Read_DirectFifo (int board_no, WORD *bufptr, int bufsize)
引数	board_no : 対象ボード番号。／0～7（複数ボード使用のときはボードごとに行う） bufptr : 転送先ADデータバッファのポインタ bufsize : 転送先ADデータバッファの大きさ（バイト指定＝総データ数の2倍）
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8-4参照）
機能動作	ADボード上のFIFOメモリから直接、ユーザプログラム内のデータバッファに転送する。一般的に当関数は割り込みイベント（【3】参照）、またはFIFOメモリ状態をポーリングして実行タイミングを得る。サンプリング動作モード設定【3】でADデータ転送の参照フラグにNot-Emplyを指定したときは1回サンプリングスキャン分のADデータを転送する。また、Half-Fullを指定したときはADボード上のFIFOメモリ容量の半分を転送する。 【注】当方法はサンプリング動作モード設定【3】でマニュアルモードを指定したときだけ有効。

【12】マニュアル（1回）サンプリング・スキャン／（全ボードについて実行）

書式	int AD68■_Get_OneScan (WORD *bufptr, int bufsize)
引数	bufptr : ADデータバッファのポインタ bufsize : ADデータバッファの大きさ（バイト指定＝総データ数の2倍）
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8-4参照）
機能動作	前記【2】で指定した全ボード、全チャンネル群に対して各1回だけサンプリングを実行する。得られたADデータは全て、ここで指定した1個の専用バッファに格納される。複数ボード使用時の並び順は先頭がボード番号0（マスタ機）のチャンネル0, 1, 2, ～15で、以下、ボード番号1～7（スレーブ各機）のADデータが同チャンネル順に続く。

【13】汎用デジタル（ラッチ）出力／（ボードごとに実行）

書式	int AD68■_Out_Aux (int board_no, int out_data)
引数	board_no : 対象ボード番号 out_data : 汎用1ビット・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8-4参照）
機能動作	各ボードごとの汎用1ビット（ラッチ）出力データの更新。 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、ボードの制御部リセット（ソフト的リセット）では変化しない。

【14】汎用デジタル（現在値）入力／（ボードごとに実行）

書式	int AD68■_Inp_Aux (int board_no)
引数	board_no : 対象ボード番号
戻り値	正常終了時 : =汎用1ビット・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8-4参照）
機能動作	各ボードごとの汎用1ビット・デジタル（現在値）入力データ読み込み。

【15】DLL管理データバッファをリング状／エンドレスに設定

書式	int AD68■_Set_SampLoop (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8-4参照）
機能動作	DLL管理のADデータバッファをリング状・エンドレスで使用するか否かの設定。（トグル） リング状エンドレスに設定した場合、自動モードでDLL管理データバッファが満杯になると、次は同バッファ先頭を上書きする形となる。この時、ステータス取得関数【8】で得るサンプリング済みデータ点数の値も先頭値（ゼロ）に戻る。 本設定により当ハンドラは無限サンプリングモードになる。 長時間の監視システム等に利用できる。

【16】アナログ入力モードの設定／（全ボード共通）

書式	int AD68■_Set_Inpmode (int inp_mode, int ad_code, int ad_reso, int ad_rangemode)
引数	inp_mode : アナログ入力信号形式（本ハンドラではダミー）／ボード上のスイッチ設定 ad_code : ADデータコード指定 / 0: バイナリ、1: 2の補数 ad_reso : AD変換分解能（本ハンドラではダミー）／ボード上のスイッチ設定
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8-4参照）
機能動作	アナログ入力信号形式、変換出力データコード（符号化の型）、およびAD変換分解能を指定する

【17】 サンプリング動作の強制停止

書式	int AD68■_Stop_Samp (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 8-4 参照）
機能動作	サンプリングを強制的に中止する。 （ADボード上のFIFOバッファメモリに残りデータがあれば、それらは無効となる。）

【18】 残りADデータの読み込み／（全ボードについて共通に実行する）

書式	int AD68■_Read_RestData (void)
引数	な し
戻り値	正常終了時 : =読み込んだ（残り）データ数 エラー時 : エラーコード／負の値（エラーコード表 8-4 参照）
機能動作	データロス・エラー発生（当ハンドラが検出するとサンプリングを中止させる）したとき、 ADボード上FIFOバッファメモリに残りデータがあれば、これを当関数で読み込むことができる。 なお、前【17】サンプリングの強制停止を実行すると残りデータは読み出しできず、無効となる。

【19】 （ADボード）各フラグのクリア／（全ボードについて共通に実行する）

書式	int AD68■_Clear_Flags (int c__data)
引数	c__data : クリアビット指定データ。／3-12項参照／
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 8-4 参照）
機能動作	ADボード上の各フラグをクリアする。

【20】 外付・同時サンプルホールド制御／（各ボードごとに指定）

書式	int AD68■_Set_Shc (int board_no, int shc)
引数	board_no : 対象ボード番号。（0～7）／本関数は各ボードごとに行う。 shc : 外付・同時サンプルホールド制御／ 0 : しない、 1 : 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 8-4 参照）
機能動作	複数チャンネル同時サンプルホールド回路を外付・接続する場合の各サンプリングスキャン開始 タイミング制御。 shc = 1 のときは $1\mu s$ 遅らせることで先頭チャンネル（ch0）にかかる ホールドステップ誤差を最小化するが、スキャン時間が $1\mu s$ 増加するため最高サンプリング速度 が遅くなる。 shc = 0 のときは最高サンプリング速度で動作するが、スキャン先頭チャンネル （ch0）に数mvのホールドステップ誤差が発生する。（3-18項参照）

【21】 アナログ入力スキャン速度指定／（全ボード共通）

書式	int AD68■_Set_ScanSpeed (int speed)
引数	speed : アナログ入力スキャン速度／ 0 : 高速モード、 1 : 低速モード、 2 : 予備モード2、 3 : 予備モード3。
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8-4参照）
機能動作	スキャン速度により（複数チャンネル使用時に）得られるADデータの正確度が変わる。 詳しくは2-2項、および3-6項を参照。

【22】（オプション的な）メッセージ発信の設定

書式	int AD68■_Plus_Message (int submit)
引数	submit : オプション・メッセージ発信の有無指定。／0 : 発信しない、 1 : 発信する
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8-4参照）
機能動作	（マニュアルモードの場合） 関数【3】のtrs_trigで指定する（ADボードからデータを読み込むための）参照 フラグを検出した時点でメッセージを発信する。 （自動モードの場合） 関数【3】のtrs_trigで指定する（ADボードからデータを読み込むための）参照 フラグを検出し、所定の読み込み動作を行った直後にメッセージを発信する。

【23】 本ハンドラのバージョン取得

書式	int AD68■_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8-4参照）
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。

表8-4. エラーコード一覧（複数ボード対応ハンドラ用）

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？）	AD_Open_AD Sys ()
- 2	ボードIDが違う。	AD_Open_AD Sys ()
- 3	FIFOバッファにアクセスできない。	AD_Open_AD Sys ()
- 4	ドライバファイルを検出できない。	AD_Open_AD Sys ()
- 5	ドライバファイルのバージョンが不適切。	AD_Open_AD Sys ()
- 6	初期化が未来実行	
- 7	連続サンプリング開始前の設定不足。	AD_Start_Samp ()
- 10	指定ボード枚数（1～8が適切） 指定パラメータの不適当	num_board
- 11	指定ボード番号（0～7が適切） 指定パラメータの不適当	board_no
- 12	サンプリング実行チャンネル数 指定パラメータの不適当	no_ch
- 15	ADデータ転送時の参照フラグ 指定パラメータの不適当	trs_trig
- 16	ADデータ転送先 指定パラメータの不適当	buf_area
- 17	割り込みイベント発生要因 指定パラメータの不適当	intr_sw、
- 18	トリガ動作モード 指定パラメータの不適当	trg_mode
- 19	トリガ源 指定パラメータの不適当	trg_source
- 20	トリガ極性 指定パラメータの不適当	trg_pol
- 21	トリガ源選択 指定パラメータの不適当	trg_sel
- 22	トリガレベル（レンジトリガ以外） 指定パラメータの不適当	trg_level
- 23	トリガレベル（レンジトリガ） 指定パラメータの不適当	trg_level_hi/_lo
- 24	トリガレベル上限値・下限値の大小関係不適当	trg_level
- 25	クロック源 指定パラメータの不適当	clk_source
- 26	サンプリング・クロックの指定方法 パラメータの不適当	set_mode
- 27	クロック周期の単位 指定パラメータの不適当	time_unit
- 28	クロック周期の値、または分周比 指定パラメータの不適当	clk_period
- 29	トリガ後のサンプリング点数 指定パラメータの不適当	
- 32	ADデータ・コード 指定パラメータの不適当	data_code
- 35	外付・同時サンプルホールド 指定パラメータの不適当	shc
- 36	アナログ入力スキャン速度 指定パラメータの不適当	speed
- 41	外部クロック源周波数を設定していない。	AD_Set_Clock ()
- 42	ADデータバッファ・サイズが小さすぎる。	bufsize
- 43	1回サンプリングスキャン実行時、チャンネル数設定未実行	AD_Get_OneScan ()
- 44	bufptrエラー	bufptr
- 45	サンプリングされたADデータ数を超えて読もうとした。	AD_Read_Data ()
- 46	割り込みリソースが割り当てられていない。	AD_Set_SampMode ()
- 47	割り込み設定エラー（重複など）	AD_Set_SampMode ()
- 49	メモリ確保エラー	
- 50	ユーザプログラム内のADデータバッファが指定されている	AD_Read_DllData ()
- 51	本ハンドラDLL管理ADデータバッファが指定されている	AD_Read_DirectFifo ()
- 52	連続サンプリング中なのに各種条件設定等をしようとした。	
- 53	現在、サンプリング中である。	AD_Read_RestData ()

8-5. サンプルプログラム、使用上の注意

重要

本ハンドラを構成する関数は一定手順、または特定の組み合わせでのみ有効に動作します。手順については前項先頭にも記しましたが、組み合わせの点で注意すべきことは前項の関数【3】で設定するサンプリングモードと【10】【11】のデータ転送関数です。以下に連続サンプリングの典型例を示します。

<例1>

利用形態	プログラム・ファイル名	対応言語 (version)
即トリガ、またはポストトリガ動作で 当ハンドラDLLの管理バッファに 有限サンプリング 。	S a m p l e 1 m. v b p	Visual Basic (5,6)
	S a m p l e 1 m. c	Visual C (5,6)
	S a m p l e 1 m. c	Borland C (5)
	S a m p l e 1 m. d s p	Visual C++ (5,6)
	S a m p l e 1 m. d p r	Delphi (3,4,5)
<p>“自動モード”で“スタート操作”を実行すると、当ハンドラ自体が自動的にADボードのF I F Oメモリからデータを“当ハンドラDLLの管理するバッファ”に読み込む。</p> <p>このとき、“ADデータ転送の参照フラグ”を《Not-HALF-FULL》に設定してある場合は、“F I F Oメモリ容量”の半分単位でブロック転送、また、《Not-Empty》に設定の場合は、“1 スキャン分”単位で転送する。以上はバックグラウンドで自動実行され、指定トリガ後サンプリング点数に達すると自動停止する。</p> <p>ユーザプログラムでは“サンプリング済みデータ点数”を参照して有効なデータを当ハンドラDLLの管理するバッファから読み込む。</p>		

<例2>

利用形態	プログラムファイル名	対応言語 (version)
即トリガ、またはポストトリガ動作で 当ハンドラDLLの管理バッファに 無限サンプリング 。	S a m p l e 2 m. v b p	Visual Basic (5,6)
	S a m p l e 2 m. c	Visual C (5,6)
	S a m p l e 2 m. c	Borland C (5)
<p>当ハンドラDLLの管理するバッファを“エンドレス・リング状”に設定し、前<例1>と同様に操作する。この場合、“当ハンドラDLL内のバッファ”の末尾と先頭が連結された構造となり、1 廻り以後は上書きされる動作となる。</p> <p>ユーザプログラム側からは、“サンプリング済みデータ点数”の値を当バッファのポインタ（関数【8】ステータスから得る）として利用する。連続サンプリングが“ストップ操作”まで無限に実行されるので長時間の監視システムや、低速のプリトリガ動作などに利用できる。</p>		

＜例３＞

利用形態	プログラムファイル名	対応言語 (version)
即トリガ、またはポストトリガ動作で ユーザプログラムの管理バッファに 有限／無限サンプリング。	S a m p l e 3 m. v b p	Visual Basic (5,6)
	S a m p l e 3 m. c	Visual C (5,6)
	S a m p l e 3 m. c	Borland C (5)
<p>“マニユアルモード”で“スタート操作”を実行すると、ADボードが連続サンプリングを開始しても当ハンドラは何もしない。ユーザプログラムは自力でステータス関数をポーリングして“ボードの生ステータス”から《Not-HALF-FULL》または《Not-Empty》を検出してデータ読み込みタイミングを知り、【11】AD_Read_DirectFifo()で読み込む。</p> <p>このとき、“ADデータ転送の参照フラグ”を《Not-HALF-FULL》に設定してある場合は、“F I F Oメモリ容量”の半分単位でブロック転送、また、《Not-Empty》に設定の場合は、“1 スキャン分”単位で読み込む。当モードで指定できるトリガ後サンプリング点数は、【7】スタート時に指定する有限値であるが、【15】無限サンプリングモード設定操作を行うと、“ストップ操作”まで無限にサンプリング動作を続けることもできる。</p>		

補足説明

キーワード	関連関数など / 【 】内の数字は関連する関数項目番号
“自動モード”	【3】連続サンプリング動作モード、および 割り込みイベントの設定関数で選択・指定する
“マニユアルモード”	
“ADデータ転送の参照フラグ”	
“スタート操作”	【7】連続サンプリング・スタート関数
“ハンドラDLL管理バッファ”	ユーザプログラム側へは【10】AD_Read_DllData()で転送、 容量は【2】指定チャンネル数×【7】指定データ点数/ch
“F I F Oメモリ容量”	【1】初期化のとき自動検出
“1 スキャン分”	【2】指定チャンネル数を各1回サンプリング分
“サンプリング済みデータ点数”	【8】ステータスから得る
“エンドレス・リング状”	【15】DLL管理バッファをリング状・エンドレスに設定
“ストップ操作”	【17】連続サンプリング動作の強制停止関数