ＩOG－PCＩボード

ＩOGドライバー関数 説明書

(株)ファード

履歴

第1版：２００５．８／７：初版
第2版：２００５．９／８：外部Ｉ／Ｏコントロール関数追加（IogEx....）
第3版：２００５．１１／１４：システムタイマ・リード変換関数追加。
パターン連続取り込み関数追加
ハンドルを構造体に変更(FCOM)

目 次

１．関数一覧

このドライバーは、ＰＣＩドライバーを実行し、IO－FPGA を制御する関数で、
DLL にて提供される。
また、アプリケーションが使用する関数である。

・一般関数

番号	関数名	概要
１	IogOpen	ボードドライバーと接続し、使用できるようにする。
２	IogClose	ボードドライバーと切り離し、使用を取り消す。
３	IogIntEntry	割り込み登録
４	IogIoRegWrite	IoReg ライト
５	IogIoRegRead	IoReg リード
６	IogIoRegReadEx	IoReg リード（ ヘッダー部付属）
７	IogIoMemWrite	SRAM ライト（IO-Memory）
８	IogIoMemRdReq	SRAM リード要求（ 〃 ）
９	IogUpLoadRead	UpLoad されたデータ のリード
１０	IogStop	転送を強制ストップさせる。
１１	IogGetUpTRCount	リード(UpLoad) 転送カウンタを取得
１２	IogINTRCnt	F-I/F 外部割込み(INTR)コントロール
１３	IogFPGAReset	IO－FPGA へのリセット

・プライベート関数

１４	IogPciRegWrite	PCI レジスタのライト
１５	IogPciRegRead	PCI レジスタのリード
１６	IogPciRegSet	PCI レジスタのセット
１７	IogPciRegReset	PCI レジスタのリセット
１８	IogPciBufRead	PCI バッファのリード（シングル）
１９	IogPciBufWrite	PCI バッファのライト（シングル）
２０	IogGetSysTimer	パソコンのシステムタイマを取得
２１	IogSysTimerToField	システムタイマをカレンダー時刻フィールドに変換

・外部Ｉ／Ｏコントロール関数（“IogEx”が接頭語）

Ex1	IogExMode	動作モードの設定
Ex2	IogExMode0DIR	モード０のデータ方向の設定
Ex3	IogExPortRead	ポートリード（外部からデータ入力）
Ex4	IogExPortReadEx	ポートリード（外部からデータ入力）ヘッダー部付属
Ex5	IogExPortWrite	ポートライト（外部へデータ出力）
Ex6	IogExMode1DIR	モード１のデータ方向の設定
Ex7	IogExTRGMode	トリガモードの設定
Ex8	IogExTRG1Out	トリガ１出力
Ex9	IogExTRG2Out	トリガ２出力
Ex10	IogExPMode	パターンモードのモード設定
Ex11	IogExPStart	パターンスタート
Ex12	IogExPStop	パターンストップ
Ex13	IogExPBusyRead	パターンビジーステータスリード
Ex14	IogExPStartADR	パターンスタートアドレス設定
Ex15	IogExPEndADR	パターンエンドアドレス設定
Ex16	IogExINTMaskSet	外部コントロール割り込みマスクセット（禁止）
Ex17	IogExINTMaskClear	外部コントロール割り込みマスククリア（許可）
Ex18	IogExINTStatusClear	外部コントロール割り込みステータスクリア
Ex19	IogExINTStatusRead	外部コントロール割り込みステータスリード

・パターン連続取り込み専用関数

Ex20	IogExPIStart	取り込みスタート
Ex21	IogExPIStop	〃　ストップ
Ex22	IogExPIRead	取り込みデータリード
Ex23	IogExPITimerRead	取り込みデータのタイマーリード

２．関数詳細

関数１	ボードのオープン
ﾌﾟﾛﾄﾀｲﾌﾟ	FCOM * IogOpen(BYTE sw)
引数	BYTE sw：ディップスイッチ値（１～15） ディップスイッチは、ボード上の“Ｓ１”(4Bit)　ON="1"

戻り値	FCOM * ：ボードのハンドル構造体ポインター
	以降の関数は、引数としてこのハンドル値を使用する。
	エラー
	=0xffffffff：ボードがないか、ドライバーがインストールされていない。
	＊＊構造体は、以下の構成になっていて、IogDriverIF.h に記述
	typedef struct Fcomm{
	HANDLE swno; // ハンドル
	void *PHeader; // リード時(戻り）のリードヘッダーポインター
	} FCOM;

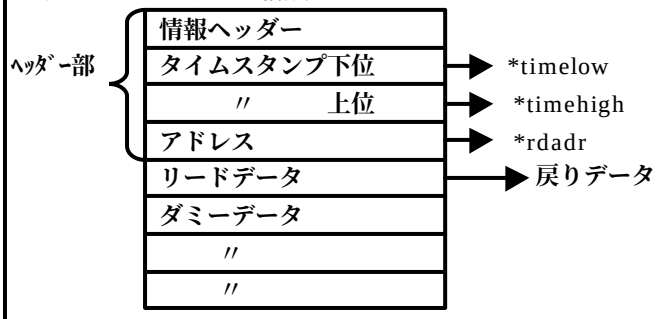
内容	ドライバーと接続し、ボードを使用できるようにする。
備考	

関数２	ボードのクローズ
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogClose（ FCOM *fcom ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値）
戻り値	なし
内容	ボードと切り離し、本アプリで使用を終了する。 終了時は、必ず実行しなければならない。
備考	

関数３	割り込みを登録する。
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogIntEntry(FCOM *fcom,DWORD hwnd,DWORD message)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD hwnd：メッセージを通知するウインドウハンドル DWORD message：メッセージ番号 ウインドウハンドルは、(DWORD)this->GetSafeHwnd()で得らる。 メッセージ番号は、WM_USER+xx にすること。 xx は、０～0x7fff です。
戻り値	なし
内容	割り込みメッセージを登録し、割り込み待ちスレッドを作成起動させる。 割り込みが発生する（イベント）と、登録メッセージを登録ウインドウに 通知する。
備考	

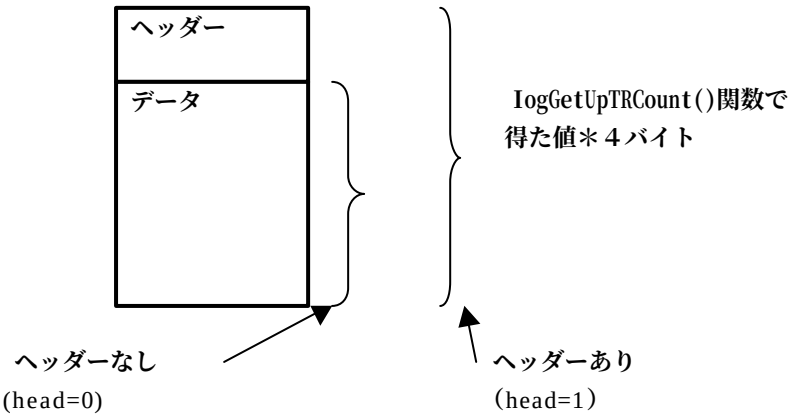
関数４	ＩｏＲｅｇにデータをライト
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogIoRegWrite(FCOM *fcom, DWORD regadr,DWORD dat,int *err)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD regadr：IO－FPGA レジスタアドレス（オフセット） DWORD dat：ライトデータ int *err：戻りステータス格納ポインター
戻り値	なし。 ・戻りステータス int *err： 0 ： 正常終了 エラー －1 ：F－I/F バスが使用できない。（他で使用中）
内容	IO-FPGA 内のレジスタにデータを書き込む。
備考	

関数５	ＩｏＲｅｇからデータをリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogIoRegRead(FCOM *fcom, DWORD regadr,int *err)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD regadr：IO－FPGA レジスタアドレス（オフセット） int *err：戻りステータス格納ポインター
戻り値	DWORD：正常終了時、レジスタのリードデータ。 ・戻りステータス int *err： 0 ： 正常終了 エラー －1 ：F－I/F バスが使用できない。（他で使用中） －２：転送個数エラー －３：リードデータのコマンドエラー －４： 〃 アドレスエラー
内容	IO-FPGA 内のレジスタからデータを読み出す。
備考	

関数６	ＩｏＲｅｇからデータをリード（ヘッダー部付属）
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogIoRegReadEx(FCOM *fcom, DWORD regadr, DWORD *rdadr,DWORD *timelow,DWORD *timehigh,int *err)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD regadr：IO－FPGA レジスタアドレス（オフセット） DWORD *rdadr ：ヘッダー部のアドレス読み出し格納ポインター DWORD *timelow： 〃 タイムスタンプ下位読み出し格納ポインタ DWORD *timehigh： 〃 〃 上位読み出し 〃 int *err：戻りステータス格納ポインタ
戻り値	DWORD：正常終了時、レジスタのリードデータ。 ・戻りステータス int *err： 0 ： 正常終了 エラー －1 ：F－I/F バスが使用できない。（他で使用中） －２：転送個数エラー －３：リードデータのコマンドエラー －４： 〃 アドレスエラー
内容	IO-FPGA 内のレジスタからデータを読み出す。 付属のヘッダー部も読み出す。 開発 FPGA からのデータ（リード）には、ヘッダー部がありその部分も 読み出す。 送られてくるデータ構成 
備考	

関数７	SRAM にデータをライト
ﾌﾟﾛﾄﾀｲﾌﾟ	int IogIoMemWrite(FCOM＊fcom,void ＊buf,DWORD len,DWORD memadr, int mode,int wait)
引数	FCOM ＊fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） void ＊buf：出力データバッファポインタ DWORD len：出力データバイト数（ただし、DWORD 単位） DWORD memadr：SRAM のアドレス int mode：転送モード TRG_MODE（=0x00）：ターゲットモード MST_MODE（=0x01）：マスタモード int wait：待ちモード INT_WAIT：終了割り込み待ちモード。（この関数は、すぐリターンされる） SYNC_WAIT：同期待ちモード。（出力終了まで、リターンされない）
戻り値	int： 0 ： 正常終了 エラー －1 ：ボードがオープンされていない。 －2 ：転送モードが使用できない。 －3 ：F－I/F バスが使用できない。
内容	SRAM に、データを書き込む。 ・転送モード ターゲットモード：CPU が、出力する。 マスタモード：ボードのマスタ転送機能により出力する。（ただしマスタ機能あり） ・終了待ちモード 割り込み待ち；終了割り込みをメッセージ受信にて待つ。 この関数は、終了しなくてもすぐ戻ってくる。 同期待ち；全て出力が終了すると、関数から戻ってくる。
備考	

関数８	SRAM からデータをリード要求
ﾌﾟﾛﾄﾀｲﾌﾟ	int IogIoMemRdReq(FCOM *fcom,DWORD len,DWORD memadr)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD len：入力データバイト数（ただし、４の倍数） DWORD memadr：SRAM のアドレス
戻り値	int： 0 ： 正常終了 エラー －1 ：ボードがオープンされていない。 －2 ：F－I/F バスが使用できない。
内容	SRAM から、データを読み出すコマンドを送信（DownLoad）。 この関数は、メモリリードコマンドを IO－FPGA にライトするのみ。 IO-FPGA がコマンドを解釈し、メモリをリードしてデータを UpLoad する。 F-F/F を通して、ドライバーのバッファに読み込まれると、 メッセージが発行され、アプリケーションは,ドライバーのバッファから IogUploadRead()関数で、リードする。
備考	

関数９	UpLoad されたデータをリード
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogUpLoadRead(FCOM *fcom,void *buf,DWORD len,int head)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） void *buf：リードデータ格納バッファのポインター DWORD len：リードデータバイト数（ただし、DWORD 単位） int head：ヘッダー付き 0：ヘッダーは、除く 1：ヘッダー付き リードデータバイト数は、IogGetUpTRCount()関数で実際 UpLOad された個数を取得したものを使用できる。（＊４が必要） ヘッダー付き
戻り値	なし
内容	UpLoad されたデータをドライバーのバッファからリードする。 ドライバーのバッファ 
備考	

関数１０	転送の強制終了
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogStop(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値）
戻り値	なし
内容	F－I/F 転送中（IoReg のリード／ライト、IoMem のリード／ライト） をストップさせる。 割り込みは、“ 正常終了 ” となる。
備考	

関数１１	リード（UpLoad）転送カウンタを取得
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogGetUpTRCount（ FCOM *fcom ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値）
戻り値	転送したカウント数 (注)バイト数ではない。（32bit 転送で＊４でバイト数）
内容	カウンタは、転送スタートで自動クリアされる。
備考	

関数１２	F-I/F 外部割込み（INTR）コントロール
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogINTRCnt（ FCOM *fcom,int intcnt ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） int intcnt：割り込みコントロール ０：割り込み禁止 １：割り込み許可
戻り値	なし。
内容	F-I/F の割り込み(INTR)をコントロールする。 (注)割り込みメッセージ（EVENT_INTR）を受信したときには 割り込み禁止状態になっているので、割り込み処理後 再び、割り込み許可にすること。（次の割り込みのため）
備考	

関数１３	I0－FPGA へのリセット
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogFPGAReset（ FCOM *fcom,int onoff ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） int onoff：リセットの ON/OFF ０：OFF １：ON（リセット状態）
戻り値	なし。
内容	IO－FPGA へのリセット ＯＮ／ＯＦＦはレベル出力なので、 ＯＮのままでは、リセット状態なので何も動作しない。 必ず、ＯＦＦに戻しておくこと。
備考	

関数１４	ＰＣＩレジスタのライト
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogPciRegWrite（ FCOM *fcom,DWORD offset,DWORD wrdat ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD offset：レジスタのオフセット DWORD wrdat：ライトデータ
戻り値	なし
内容	PCI-FPGA のレジスタにライトする。
備考	

関数１５	ＰＣＩレジスタのリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogPciRegRead（ FCOM *fcom,DWORD offset ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD offset：レジスタのオフセット
戻り値	リードしたレジスタ値
内容	PCI-FPGA のレジスタからリードする。
備考	

関数１６	ＰＣＩレジスタのセット（ビットのセット）
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogPciRegSet（ FCOM *fcom,DWORD offset,DWORD setdat ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD offset：レジスタのオフセット DWORD setdat：セットするデータ
戻り値	なし
内容	PCI-FPGA のレジスタをセットする。 setdat の”1”のビットのみセット（＝１）する。
備考	

関数１７	ＰＣＩレジスタのリセット（ビットのクリア）
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogPciRegReset（ FCOM *fcom,DWORD offset,DWORD clrdat ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD offset：レジスタのオフセット DWORD clrdat：リセット（クリア）するデータ
戻り値	なし
内容	PCI-FPGA のレジスタをリセット（クリア）する。 clrdat の”1”のビットのみリセット（＝０）する。
備考	

関数１８	ドライバーのバッファからデータをリード（シングル）
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogPciBufRead(FCOM *fcom,DWORD offset,DWORD udbuf)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD offset：バッファのオフセット（バイトオフセット） (注)バッファは、1MByte なので、それ以内。 DWORD udbuf：ドライバーバッファの種類 １：DownLoad 用バッファ ０：UpLoad 用 〃
戻り値	DWORD：読み出しデータ。
内容	Ｆ－Ｉ／Ｆ用のドライバーバッファから１DWORD データを読み出す。
備考	

関数１９	ドライバーのバッファにデータをライト（シングル）
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogPciBufWrite(FCOM *fcom,DWORD offset,DWORD wrdat,DWORD udbuf)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD offset：バッファのオフセット（バイトオフセット） (注)バッファは、1MByte なので、それ以内。 DWORD wrdat：ライトデータ。 DWORD udbuf：ドライバーバッファの種類 １：DownLoad 用バッファ ０：UpLoad 用 〃
戻り値	なし。
内容	Ｆ－Ｉ／Ｆ用のドライバーバッファに１DWORD データを書き込む。
備考	

関数２０	パソコンのシステムタイマを取得
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogGetSysTimer(FCOM *fcom,DWORD *timel,DWORD *timeh)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD *timel ：システムタイマ下位 32bit 格納ポインタ DWORD *timeh： 〃 上位 〃
戻り値	なし
内容	パソコンのシステムタイマを取得する。 このタイマは、64bit の 0.1uS カウンタである。 時刻は、グリニッジ時間の 1601 年１月１日からの積算カウンタである。
備考	

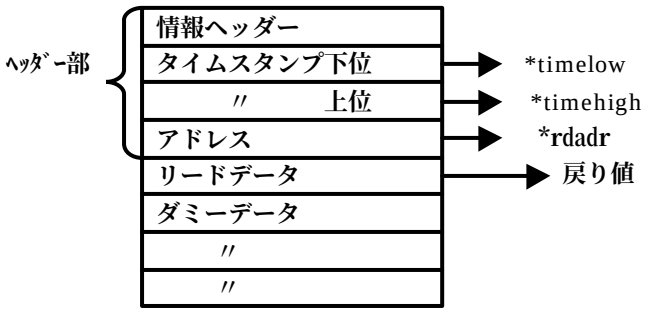
関数２１	システムタイマをカレンダー時刻フィールドに変換
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogSysTimerToField(FCOM *fcom,DWORD *timel,DWORD *timeh, WORD *time_field)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値） DWORD *timel ：システムタイマ下位 32bit 格納ポインタ DWORD *timeh： 〃 上位 〃 DWORD *time_field：カレンダー時刻格納配列ポインター 配列は、１０個必要
戻り値	なし
内容	システムタイマ（64bit）をカレンダー時刻フィールドに変換する。 WORD time_field[0]：ｎ秒 WORD time_field[1]：μ秒 WORD time_field[2]：m秒 WORD time_field[3]：秒 WORD time_field[4]：分 WORD time_field[5]：時 WORD time_field[6]：日 WORD time_field[7]：月 WORD time_field[8]：年 WORD time_field[9]：週（０~6：日曜日～土曜日） 値は、全て１６ビット・バイナリィである。 ｎ秒は、システムタイマが、100nS 分解能なので、０、１００.... ９００となる。
備考	

・外部Ｉ／Ｏコントロール関数（”IogEx“が関数名の接頭語）

関数 Ex1	動作モードの設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExMode(FCOM *fcom,int mode)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） int mode：動作モード QMODE0：0：モード０（単純入出力） QMODE1：1：モード１（ハンドシェークによる入出力） QMODEP：2：パターン入出力モード
戻り値	なし
内容	ポートの動作モードを設定する。
備考	パターンモードは、メモリ（ＳＲＡＭ）が、必要になります。

関数 Ex2	モード０のデータ方向の設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExMode0DIR(FCOM *fcom,int dir)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） int dir：バイト単位の入出力方向 各バイト単位に設定する。 位置 bit0：D0～7 bit1：D8～15 bit2：D16～23 bit3：D24～31 （設定内容） QDIRIN ：0：入力 QDIROUT：1：出力
戻り値	なし。
内容	バイト単位の方向を設定する。 ただし、バッファが”TTL”を実装しているときのみ有効である。
備考	

関数 Ex3	ポートリード（外部からのデータ入力）
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogExPortRead（FCOM *fcom ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	DWORD:ポートの 32 ビットデータ入力。 モード０：ポート直接入力データ。 モード１：ラッチされた入力データ。
内容	バッファＴＴＬポートのデータ入力を行う。 出力に指定されたポートについては、出力したデータが入力されます。 注１．モード１入力の場合、IBF=1 の場合ラッチされたデータをリードします。 IBF=0 の場合、IBF=1 になってリードして、リターンします。 注２．パターンモード時は、何もしません。
備考	

関数 Ex4	ポートリード（外部からのデータ入力）ﾍｯﾀﾞｰ部付属
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogExPortReadEx（FCOM *fcom,DWORD *rdadr, DWORD *timelow,DWORD *timehigh ）
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD *rdadr ：ヘッダー部のアドレス格納ポインタ DWORD *timelow： 〃 タイムスタンプ下位格納ポインタ DWORD *timehigh： 〃 〃 上位 〃
戻り値	DWORD:ポートの 32 ビットデータ入力。 モード０：ポート直接入力データ。 モード１：ラッチされた入力データ。
内容	外部コネクタからのデータ入力を行う。 出力に指定されたポートについては、出力したデータが入力されます。 また、ヘッダー部のデータも読み出します。 送られてくるデータ構成 
備考	注１．モード１入力の場合、IBF=1 の場合ラッチされたデータをリードします。 IBF=0 の場合、IBF=1 になってリードして、リターンします。 注２．パターンモード時は、何もしません。

関数 Ex5	ポートライト（外部へデータ出力）
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExPortWrite(FCOM *fcom,DOWRD dat)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD dat：出力データ
戻り値	なし
内容	ポートにデータを出力します。 入力に指定されたポートに付いては、無効です。 注１． モード０：出力ポートにデータをラッチします。 モード１；OBF=0 のときデータをラッチし、OBF=1 の場合 OBF=0 になってから データを出力にラッチし、リターンします。 注２．パターンモード時は、何もしません。
備考	

関数 Ex6	モード１ データ方向の設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExMode1DIR(FCOM *fcom,int dir)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） int dir：方向 QDIRIN ：０：入力 QDIROUT：１：出力 (注) ＴＴＬの場合：32 ビット全てのバイトになる。 差動バッファ時には、 入力バッファ；入力のみ 出力バッファ：出力のみ になり、それ以外だとデータは、保証されない。 （入力バッファ実装時に、出力モードを設定したとき。
戻り値	なし。
内容	モード１のデータの転送方向を設定する。 モード１使用時のコントロール信号 入力モード時 CTLO0：IBF（出力） CTLI0：STB（入力） 出力モード時 CTLO1：OBF（出力） CTLI1：DACK（入力）
備考	

関数 Ex7	トリガモードの設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExTRGMode(FCOM *fcom,int t1w,int t1pol,int t2w,int t2pol)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） int t1w：トリガ１のパルス幅の設定 設定値：０～１５ （設定値＋１）＊１０ｕＳ （Max160uS） int t1pol：トリガ１の極性 QTRGPLUS ：１：正 QTRGMINUS：０：負 int t2w：トリガ２のパルス幅の設定（内容は、トリガ１と同じ） int t2pol：トリガ２のパルス極性 （ 〃 〃 ）
戻り値	なし。
内容	トリガ１、２のモードを設定する。
備考	

関数 Ex8	トリガ１ 出力
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExTRG1Out(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	なし。
内容	トリガ１にトリガモード指定のパルスを１個出力する。 出力は、コントロール信号の“ CTLO2 ”である。
備考	

関数 Ex9	トリガ２ 出力
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExTRG2Out(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	なし。
内容	トリガ２にトリガモード指定のパルスを１個 出力する。 出力は、コントロール信号の“ CTLO３ ”である。
備考	

関数 Ex10	パターンモードのモード設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExPMode(FCOM *fcom,int rep,int dir,int div)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） int rep：パターン繰り返しモード QPREPT_ON ：１；繰り返しＯＮ QPREPT_OFF：０： 〃 ＯＦＦ（１回のみ行う） int dir:パターン入出力 QPDIRIN ：：０：パターン入力モード QPDIROUT:：１：パターン出力モード int div：出力クロックの分周 設定値：０～３ 分周値＝発振器の周波数／（２＾div） 1/1、1/2、1/4、1/8 分周出来る。 20MHz 発振器の場合 20MHz、10MHz、5MHz、2.5MHZ となる。
戻り値	なし
内容	パターンのモード設定を行う。 (注)繰り返しモードＯＮの場合の終了方法は、 パターン実行中、この関数を繰り返しＯＦＦ（QPREPT_OFF）で実行する。 パターンエンドアドレスまで、実行後終了します。
備考	

関数 Ex11	パターンスタート
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExPStart(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	なし
内容	パターンを開始する。
備考	

関数 Ex12	パターンストップ
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExPStop(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	なし。
内容	現在実行中のパターン動作を強制停止させる。
備考	

関数 Ex13	パターンビジーステータスをリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogExPBusyRead(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	DWORD ：ビジーステータス ０：停止 １：実行中（ビジー）
内容	パターン実行中フラグをリードする。
備考	

関数 Ex14	パターンスタートアドレスの設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExPStartADR(FCOM *fcom,DWORD sadr)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD sadr：スタートアドレスの設定 設定範囲：0x000000～0x3fffff（4M パターン） (注) DWORD アドレス 32bit(DWORD)＝１パターン
戻り値	なし
内容	パターンの実行スタートアドレスを設定する。
備考	

関数 Ex15	パターンエンドアドレスの設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExPEndADR(FCOM *fcom,DWORD eadr)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD eadr：パターンエンドアドレス 設定範囲：0x000000～0x3fffff（4M パターン）
戻り値	なし
内容	パターンの実行エンドアドレスを設定する。 繰り返しをＯＮにすると、エンドアドレス実行後、 スタートの実行に移る。
備考	

関数 Ex16	外部コントロール割り込みマスクセット
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExINTMaskSet(FCOM *fcom,DWORD mask)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD mask：マスク・セットデータ QINT1MSK：bit0：外部 INT1 マスク QINT2MSK：bit1： 〃 2 〃
戻り値	なし
内容	割り込みマスクをセットする。（割り込み禁止）
備考	

関数 Ex17	外部コントロール割り込みマスククリア
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogExINTMaskClear(FCOM *fcom,DWORD mask)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD mask：マスク・クリアデータ QINT1MSK：bit0：外部 INT1 マスク QINT2MSK：bit1： 〃 2 〃
戻り値	なし
内容	割り込みマスクをクリアする。（割り込み許可）
備考	

関数 Ex18	外部コントロール割り込みステータスをクリアする。
ﾌﾟﾛﾄﾀｲﾌﾟ	void IogINTStatusClear(FCOM *fcom,DWORD clrdat)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） DWORD clrdat：割り込みステータスクリアデータ QINT1CLR：bit0：外部 INT1 クリア QINT2CLR：bit1： 〃 2 〃
戻り値	なし
内容	割り込みステータスをクリアする。 マスクをクリア（割り込み許可）されていれば、本ドライバーにて クリアされます。 クリアされた割り込みは,メッセージの付加データにより知らせます。
備考	

関数 Ex19	割り込みステータスをリードする。
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD IogExINTStatusRead(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	DWORD ：割り込みステータス QINT1STS：bit0：外部 INT1 割り込みステータス QINT2STS：bit1： 〃 2 〃
内容	割り込みステータスをリードします。 割り込みマスクをクリア（割り込み許可）されていると 本ドライバーでクリアして、メッセージの付加データとして知らせます。 WindowProc()メッセージ待ち関数の引数 wParam EVENT_INT1：bit8：外部割込み INT1 EVENT_INT2：bit9： 〃 INT2
備考	

関数 Ex20	パターン連続取り込みスタート
ﾌﾟﾛﾄﾀｲﾌﾟ	int IogExPIStart(FCOM *fcom,int ptw,int enbpol)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） int ptw：入力データ幅　０：３２bit　１；１６bit int enbpol：入力イネーブル極性 ０：負　１：正
戻り値	int ０：正常終了 負：エラー終了
内容	パターン連続取り込みをスタートします。
備考	

関数 Ex21	パターンの取り込みを強制終了（ストップ）
ﾌﾟﾛﾄﾀｲﾌﾟ	int IogExPIStop(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。）
戻り値	０：正常終了 負：エラー終了
内容	現在取り込み中の動作をストップして終了させます
備考	

関数 Ex22	パターンデータをリード
ﾌﾟﾛﾄﾀｲﾌﾟ	int IogExPIRead(FCOM *fcom,void *buf,int timeout)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） void *buf：格納先バッファポインタ int timeout：待ちタイムアウト時間（単位ｍＳ）
戻り値	int 正：リードバイト数 負：－1=エラー終了 －2=タイムアウト
内容	パターンデータをリードします。 戻り値バイト数格納されています。 (注)バッファは、入力バイト数より大きめに確保してください。 ドライバー等は、ＭＡＸ４Ｍバイト確保しています。 （１イネーブルに対し最大４Ｍバイトです。）
備考	

関数 Ex23	パターンデータをリード時のタイマーリード
ﾌﾟﾛﾄﾀｲﾌﾟ	__int64　IogExPIRead(FCOM *fcom)
引数	FCOM *fcom：ハンドル構造体ポインター（ボードオープン時の戻り値。） （パターンデータ読み取り時のもの）
戻り値	__int64（６４ﾋﾞｯﾄ） パターン読み取り時のシステムタイマ値 システムタイマ 0.1μカウンタ （1601 年１月１日からの経過時間：グリニッジ時間）
内容	パターン読み取り時の FCOM 内に時刻データポインターが格納されていて そのポインターから時刻情報を取り出す。
備考	

３．メッセージ待ち

WindowProc()にて、登録メッセージで待つ

```
LRESULT CFTestDlg::WindowProc(UINT message, WPARAM wParam, LPARAM lParam)
```

message：IogIntEntry()関数で登録したメッセージ。

wParam：メッセージ要因

EVENT_WREND（0x01）：IogIoMemWrite（）関数の終了。
ただし、割り込み待ち（INT_WAIT）使用時。

EVENT_UPEND（0x02）：IogIoMemRdReq()後の UpLoad の終了。

EVENT_INTR（0x04）：F-I/F 外部割込み。

lParam：付加データ（DWORD）