ＩＯＧ－ＰＣＩボード

ＰＣＩ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：２００５．８・８：システムタイマ取得関数追加 PciGetSysTimer()
第３版：２００５．１１／１４：システムタイマ・リード変換関数追加。

目　　次

１．関数一覧.....3
２．詳細.....4
関数１　割り込み待ちを起動する。.....4
関数２　割り込み要因をリードする。.....5
関数３　Ｆ－Ｉ／ＦバスのＵＰＬｏａｄの禁止／解除.....6
関数４　Ｆ－Ｉ／Ｆバスの転送モードの設定.....7
関数５　F-I/F の転送開始.....8
関数６　F-I/F の転送を強制終了させる。.....9
関数７　DownLoad 転送バイト数の設定.....10
関数８　ドライバーのバッファにデータをライト（シングル）.....11
関数９　ドライバーのバッファからデータをリード（シングル）.....12
関数１０　ドライバーのバッファにデータをライト（ブロック）.....13
関数１１　ドライバーのバッファからデータをリード（ブロック）.....14
関数１２　FIF 転送カウンタを取得.....15
関数１３　ボード情報をリード.....16
関数１４　ＰＣＩ機能をリード.....17
関数１５　F-I/F コントロール信号をリード.....18
関数１６　割り込みマスクをセット（割り込み禁止）.....19
関数１７　割り込みマスクをクリア（割り込み許可）.....20
関数１８　割り込みステータスクリア.....21
関数１９　ＰＣＩコンフィグレジをリードする。.....22
関数２０　ＰＣＩ部のレジスタをリード.....23
関数２１　ＰＣＩ部レジスタをライト.....24
関数２２　ＰＣＩ部レジスタをセット.....25
関数２３　ＰＣＩ部レジスタをリセット.....26
関数２４　パソコンのシステムタイマを取得.....27
関数２５　システムタイマをカレンダー時刻フィールドに変換.....28

１．関数一覧

このドライバーは、ＰＣＩデバイスドライバーを実行し、PCI－FPGA を制御する関数で、DLL にて提供される。

また、上位 IO ドライバー（DLL）が使用する関数である。

・一般関数

番号	関数名	概要
１	PciWaitInt	イベント待ちを起動する。（ドライバーに対し）
２	PciEventRead	割り込み要因をリードする。
３	PciUPEnb	F-I/F バスの UPLoad 禁止／許可
４	PciModeSet	〃　　転送モード等設定
５	PciStart	〃　　転送開始
６	PciStop	〃　　転送強制終了
７	PciSetDWLen	DownLoad 転送バイト数の設定
８	PciBufWrite	ドライバのバッファにデータをライト（シングル）
９	PciBufRead	〃　　からデータを リード（シングル）
１０	PciBufBlockWrite	ドライバーのバッファにデータをライト（ブロック）
１１	PciBufBlockRead	〃　　からデータをリード（ブロック）
１２	PciGetTRCount	FIF バス転送カウンターを取得
１３	PciBoardTypeRead	ボード情報をリード
１４	PciPciTypeRead	Pci 機能情報をリード
１５	PciFIFCntRead	F-I/F ｺﾝﾄﾛｰﾙ信号をリード
１６	PciIntMaskSet	割り込みマスクをセット（禁止）
１７	PciIntMaskClear	〃　　クリア（許可）
１８	PciIntStatusClear	〃　　クリア

・プライベート関数

１９	PciConfRead	PCI コンフィグレジリード
２０	PciRegRead	PCI部レジスタリード
２１	PciRegWrite	〃　　ライト
２２	PciRegSet	〃　　セット（必用なビットのみセット）
２３	PciRegReset	〃　　リセット（必用なビットのみセット）
２４	PciGetSysTimer	パソコンのシステムタイマを取得
２５	PciSysTimerToField	システムタイマをカレンダー時刻フィールドに変換

２．詳細

関数１	割り込み待ちを起動する。
ﾌﾟﾛﾄﾀｲﾌﾟ	int PciWaitInt(HANDLE hnd,OVERLAPPED　*ovio)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） OVERLAPPED *ovio：OVERLAPPED 構造体のポインター 割り込み待ち用 Event ハンドルを含む。
戻り値	int：結果ステータス ０：正常終了 ０以下（負）：異常終了（割り込み待ちを起動できない）
内容	割り込み待ちスレッド内で使用し、 ドライバーに対して割り込み待ちを起動する。
備考	

関数２	割り込み要因をリードする。
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciEventRead(HANDLE hnd)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値）
戻り値	DWORD ：割り込み要因（イベント） bit0：F-I/F 転送終了 1：　〃 Over Run 終了　｝転送個数に対して 2：〃　Under Run 終了｝ 3：外部割込み（FIF） (注)bit3：外部割込みについては、割り込み処理（割り込み要求解除）後、 マスクをクリア（解除）すること。次の割り込みに対して。
内容	割り込み待ちを解除されたときの要因をリードする。
備考	

関数３	Ｆ－Ｉ／ＦバスのＵＰＬｏａｄの禁止／解除
ﾌﾟﾛﾄﾀｲﾌﾟ	int PciUPEnb(HANDLE hnd,int enb,int　retry)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） int enb：イネーブル ０：UPLoad 禁止 １：　〃　許可 int retry：異常終了時の再実行回数。
戻り値	int：結果 ０：正常終了 －１：異常終了 enb の指定状態にならなかった場合。
内容	F-I/F バスを**出力使用**するときに、入力を禁止する。 異常終了時は,時間を置いて指定回数リトライを行う。
備考	

関数４	Ｆ－Ｉ／Ｆバスの転送モードの設定
ﾌﾟﾛﾄﾀｲﾌﾟ	int　PciModeSet(HANDLE　hnd,DWORD　mode)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） DWORD mode：設定モード bit0：動作モード =0:ターゲットモード =1：マスタモード bit1：転送方向 =0：入力 =1：出力 （H/W レジスタに設定される。） bit8：終了モード =0(INT_WAIT)：割り込み待ちモード（PciStart()がすぐリターンされる） =1(SYNC_WAIT)：同期待ちモード（　〃　が終了までリターンされない）
戻り値	int：結果 ０：正常終了 －１：マスタ機能がないのに、マスタ指定をした。
内容	F-I/F バスの転送モード、方向、終了モードを設定する。
備考	

関数５	F-I/F の転送開始
ﾌﾟﾛﾄﾀｲﾌﾟ	int PciStart(HANDLE hnd)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値）
戻り値	int：結果 ０：正常終了 －１：転送バイト数がセットされていない（スタートしない）
内容	転送を開始する。 転送方向等は、これ以前にすでに設定されているものとする。 （入出力、マスタ／ターゲット共用） 転送の終了は、PciModeSet()の 終了モードに従う。 ・ INT_WAIT（すぐリターンされる。） ライト（DownLoad）：転送終了後、イベント発生。 リード（UpLoad）：バッファに格納後、 〃　　。 ・ SYNC_WAIT（終了するまでリターンしない。） (注)リードは、ドライバーバッファまでなので 終了後、PciBufBlockRead（）でアプリバッファにコピーする必要がある。
備考	

関数６	F-I/F の転送を強制終了させる。
ﾌﾟﾛﾄﾀｲﾌﾟ	void PciStop(HANDLE hnd)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値）
戻り値	なし
内容	現在転送中を強制終了させる。 （H/W より正常終了割り込みが起きるが、入力時データは保証されない）
備考	

関数７	DownLoad 転送バイト数の設定
ﾌﾟﾛﾄﾀｲﾌﾟ	void PciSetDWLen(HANDLE hnd,DWORD len)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） DWORD len：転送バイト数（４の倍数） FIF の転送は４DWORD 単位なので 足らないデータは、ダミーデータで転送される。
戻り値	なし
内容	DwonLoad 転送バイト数を設定する。 (注)UpLoad のバイト数の設定は、なく UpLoad 転送後、PciGetTRCount（）関数で、転送個数は得られる。
備考	

関数８	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 PciBufWrite(HANDLE hnd,DWORD offset,DWORD wrdat,DWORD udbuf)
引数	HANDLE hnd：ハンドル（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９	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 PciBufRead(HANDLE hnd,DWORD offset,DWORD udbuf)
引数	HANDLE hnd：ハンドル（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１０	ドライバーのバッファにデータをライト（ブロック）
ﾌﾟﾛﾄﾀｲﾌﾟ	int PciBufBlockWrite(HANDLE hnd,DWORD offset,void * wrbuf,DWORD len,DWORD udbuf ,OVERLAPPED *ovwr)
引数	HANDLE hnd：ハンドル（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。 void *wrbuf：ライトデータアプリバッファ。 DWORD len：ライトデータバイト数。（1MByte 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　〃 OVERLAPPED *ovwr：オーバーラップ構造体のポインター
戻り値	int：戻りステータス ０：正常終了 －１：バッファオーバーフロー offset＋len が、ドライバーのバッファを越えている時。 (注)バッファオーバーフローの場合なにもしない。 －２：ライトできない。
内容	Ｆ－Ｉ／Ｆ用のドライバーバッファに指定したバイト数分データを書き込む。
備考	

関数１１	ドライバーのバッファからデータをリード（ブロック）
ﾌﾟﾛﾄﾀｲﾌﾟ	int PciBufBlockWrite(HANDLE hnd,DWORD offset,void *rdbuf,DWORD len,DWORD udbuf ,OVERLAPPED *ovrd)
引数	HANDLE hnd：ハンドル（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。 void *rdbuf：リードデータ格納アプリバッファ。 DWORD len：リードデータバイト数。（1MByte 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　〃 OVERLAPPED *ovrd ：オーバーラップ構造体のポインター
戻り値	int：戻りステータス ０：正常終了 －１：バッファオーバーフロー offset＋len が、ドライバーのバッファを越えている時。 (注)バッファオーバーフローの場合なにもしない。 －２：リードできない。
内容	F－I／F用のドライバーバッファから指定したバイト数分データを読み出す。
備考	

関数１２	FIF 転送カウンタを取得
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciGetTRCount（ HANDLE hnd,int wrrd ）
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） int wrrd：転送方向 １：WRITE（DownLoad 転送） ０：READ （UpLoad 転送）
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	PciStart()関数での転送終了後有効となる。 カウンタは、転送スタートで自動クリアされる。
備考	UpLoad 時、必ず UnderRun 終了になり、転送個数がわからないので この関数により、リード個数を判別する。 DownLoad 時の OverRun 終了時にも転送個数がわからないので、 有効に使用できる。

関数１３	ボード情報をリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciBoardTypeRead(HANDLE hnd)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値）
戻り値	DWORD：ボード情報 bit0～3：BDNUM　ボード番号（ＤＩＰ－ＳＷ値） 同じボード使用時の識別。 bit4～31：BDTYPE ボード種類（０：IOG ～）　０～
内容	ボード情報をリードする。 ＰＣＩレジスタ（ＢＯＡＲＤＮ）をリードする。
備考	

関数１４	ＰＣＩ機能をリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciPciTypeRead(HANDLE hnd)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値）
戻り値	DWORD：ボード情報 bit0～15：ＰＣＩ機能（ＢＯＡＲＤＫレジスタ） bit0：MASTER　１：マスタ機能あり bit1：EN66MHZ　１：66MHz 動作可能 bit2：EN66BDT　１：64bit データモード動作可能 bit3：EN64BAD　１：64bit アドレスモード動作可能 bit4～5：PCIXCAP 00：PCI-X 機能なし。01：66MHz 10：133MHz bit6～7：(空き) bit8～15：(空き) bit16～31：予備
内容	ボード情報をリードする。 ＰＣＩレジスタ（ＢＯＡＲＤＫ）をリードする。
備考	

関数１５	F-I/F コントロール信号をリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciFIFCntRead(HANDLE hnd)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値）
戻り値	DWORD：コントロール信号 bit0～7：F-IF コントロール信号 bit0：VLD_DW 1：ENB_DW 2：REQ_DW 3：ACK_DW 4：VLD_UP 5：ENB_UP 6：REQ_UP 7：ACK_UP 1：アクティブ 0：インアクティブ
内容	F-I/F 信号をリードする。 FPGA の信号ピンのモニタ。
備考	

関数１６	割り込みマスクをセット（割り込み禁止）
ﾌﾟﾛﾄﾀｲﾌﾟ	void PciMaskSet(HANDLE hnd,DWORD msk)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） DWORD msk：セットする割り込みマスクビット bit0：転送終了 bit1：Over Run　｝ドライバーが制御するので bit2：Under Run　｝セットできない。 bit3：F-I/F 外部割込み（INTR） (注)ビット３以外は、ドライバーが制御する。
戻り値	なし
内容	割り込みのマスクをセットする。 セットされた割り込みは、禁止になる。 セットするマスクビットを“１”にしこの関数を実行する。
備考	

関数１７	割り込みマスクをクリア（割り込み許可）
ﾌﾟﾛﾄﾀｲﾌﾟ	void PciMaskClear(HANDLE hnd,DWORD msk)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） DWORD msk：クリアする割り込みマスクビット bit0:転送終了 bit1：Over Run　｝ドライバーが制御するので bit2：Under Run　｝クリアできない。 bit3：F-I/F 外部割込み（INTR） (注)外部割り込みが発生すると、このマスクは、ドライバーでセット(禁止)されるので **割り込み処理後、再びマスクをクリア**すること。
戻り値	なし
内容	割り込みのマスクをクリアする。 クリアされた割り込みは、許可になる。 クリアするマスクビットを“１”にしこの関数を実行する。
備考	

関数１８	割り込みステータスクリア
ﾌﾟﾛﾄﾀｲﾌﾟ	void PciIntStatusClear(HANDLE hnd,DWORD clr)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） DWORD clr：ステータスクリアデータ bit0：転送終了 bit1：Over Run bit2：Under Run　} ドライバーで制御する。 bit3：F-I/F 外部割込み(INTR) ＝１でクリアする。（０では、クリアしない） (注)外部割込みについては,マスクがクリアされていれば、自動的にクリアされる。 ステータスについては、直接 PCI レジスタからリードする。 PciRegRead（hnd,0x1c）；
戻り値	なし
内容	割り込みステータスをクリアする。
備考	

関数１９	ＰＣＩコンフィグレジをリードする。
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciConfRead(HANDLE hnd,DWORD offset)
引数	HANDLE hnd：ハンドル（ボードオープン時の戻り値） DWORD offset：コンフィグレジのオフセット（４の倍数）
戻り値	DWORD：指定オフセットのコンフィグレジデータ
内容	コンフィグレジスタを１DWORD分リードする。
備考	

関数２０	ＰＣＩ部のレジスタをリード
ﾌﾟﾛﾄﾀｲﾌﾟ	DWORD PciRegRead(HANDLE hnd,DWORD offset)
引数	HANDLE hnd：ハンドル（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	DWORD：リードレジスタ値
内容	指定オフセットのレジスタをリードする。
備考	

関数２１	ＰＣＩ部レジスタをライト
ﾌﾟﾛﾄﾀｲﾌﾟ	void　PciRegWrite(HANDLE hnd,DWORD offset,DWORD dat)
引数	HANDLE hnd：ハンドル（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 dat：ライトデータ
戻り値	なし
内容	指定オフセットのレジスタにデータをライトする。
備考	

関数２２	ＰＣＩ部レジスタをセット
ﾌﾟﾛﾄﾀｲﾌﾟ	void　PciRegSet(HANDLE hnd,DWORD offset,DWORD setdat)
引数	HANDLE hnd：ハンドル（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 dat：セットデータ
戻り値	なし
内容	指定オフセットのレジスタにデータをセットする。 ビットが“１”のビットのみセットする。
備考	

関数２３	ＰＣＩ部レジスタをリセット
ﾌﾟﾛﾄﾀｲﾌﾟ	void　PciRegReset(HANDLE hnd,DWORD offset,DWORD resetdat)
引数	HANDLE hnd：ハンドル（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 dat：リセットデータ
戻り値	なし
内容	指定オフセットのレジスタにデータをリセットする。 ビットが“１”のビットのみりセット（クリア）する。
備考	

関数２４	パソコンのシステムタイマを取得
ﾌﾟﾛﾄﾀｲﾌﾟ	void　PciGetSysTimer(HANDLE hnd,DWORD *timel,DWORD *timeh)
引数	HANDLE hnd：ハンドル（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２５	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　PciSysTimerToField(HANDLE hnd,DWORD *timel,DWORD *timeh, WORD *time_field)
引数	HANDLE hnd：ハンドル（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	