Digital
Human Machine Interface

Pro-face®

PSシリーズGタイプ DIOドライバ

ライブラリインターフェイス リファレンス

1.概要

DIOドライバはPSシリーズGタイプアプリケーション開発キット(以下ADKと称します)で開発したアプリケーションからPSG-XY08の機能を容易に使用するためのアプリケーションインターフェイス(API)をダイナミックリンクライブラリ(DLL)形式で提供します。

DIOドライバはOpen/Close処理、DOUTへの出力、DIOからの現在状態の読み取り、DIOからのラッチ入力の読み取り、ラッチ状態のクリア、チャタリングキャンセルパルス幅の設定、ワンショットカウンタイベント処理、サイクリックカウンタイベント処理を提供します。

2.開発環境および動作環境

DIOドライバを用いたアプリケーションの開発環境および動作環境はADKと同様です。

詳細につきましては**「PSシリーズGタイプアプリケーション開発キット(ADK)デベロッパーズマニュアル　第１章　2 ハードウェア環境 / 3 ソフトウェア環境」**を参照してください。

3.フロッピーディスクの構成

フロッピーディスク内のフォルダ構成を以下に示します。

```
\Windows CE Tools ─ \wce300 ─ \PSGWCE30 ┬ \Include ─── \DIO_API.H
                                        ├ \Lib\SH4 ┬─ \DIO_API.LIB
                                        │          └─ \DIO_API.EXP
                                        └ \VBSDK\SH4 ─ \DIO_API.BAS

\Manual ─ \PSGXY08_J.pdf

\PSG ─── \Windows ─ \DIO_API.DLL
```

DIO_API.DLL	DIOユニットをアプリケーションから使用するためのドライバです。PS-Gの[\Windows]フォルダへコピーして使用します。
DIO_API.LIB	DIO_API.DLLを使用するアプリケーションを開発する時にリンクするライブラリファイルです。
DIO_API.H	DIO_API.DLLをeMbedded Visual C++から使用するためのヘッダーファイルです。
DIO_API.BAS	DIO_API.DLLをeMbedded Visual Basicから使用するためのヘッダーファイルです。

4.組み込み方法

DIOドライバの組み込み

① CFカード、ActiveSync、またはイーサネット経由でPS-Gの[\Windows]フォルダへDIO_API.DLLをコピーします。

② PS-Gのコントロールパネルにあるバックアップ機能によりDIO_API.DLLをCFカードに保存します。**参照→「各PSシリーズGタイプ ユーザーズマニュアル 第3章 2.2 コントロールパネルの設定」**

③ PS-Gを再起動します。
再起動後、DIO_API.DLLが使用可能な状態になります。

インターフェイスライブラリおよびヘッダーファイルの組み込み

ADKがインストールされているパソコン(アプリケーション開発用パソコン)に、フロッピーディスク内のフォルダ構成にしたがって、各ファイル(DIO_API.DLLを除く)をコピーします。

参照→ 3.フロッピーディスクの構成

5. API仕様

API関数を使用するには使用前にドライバのOpen、使用後にドライバのCloseを行う必要があります。

例)
```
BOOL bRet;
BYTE byData;
bRet = DioDriverOpen();
bRet = GetDinData(&byData);
bRet = DioDriverClose();
```

5.1 DIOドライバAPI一覧

API名	内容
DioDriverOpen	DIOドライバのオープン
DioDriverClose	DIOドライバのクローズ
SetDinFilter	チャタリングキャンセルパルス幅の設定
GetDinData	DINポート入力データの取得
GetDinLatchData	DINポートラッチ入力データの取得
ClearDinLatchData	DINポートのラッチ状態のクリア
SetDoutData	DOUTポートへのデータの出力
GetDoutData	DOUTポート出力データの取得
SetCounterCycle	DINポートの監視周期の指定
SetCounterEvent	イベントをシグナル状態にする機能をビットごとに登録
StartCounterEvent	DINビット監視カウンタの開始
StopCounterEvent	DINビット監視カウンタの停止
ClearCounterEvent	DINビット監視登録の削除
GetCounterStat	DINビット監視登録状態の取得

5.2 関数仕様詳細

5.2.1 eMbedded Visual C++用　関数仕様詳細

DioDriverOpen

呼び出し形式

BOOL WINAPI DioDriverOpen(void)

戻り値

TRUE:正常

FALSE:エラー

引数

なし

処理概要

本DLLの機能を使用可能にします。

多重Openはできません。

使用例

```
BOOL bRet = DioDriverOpen();
```

DioDriverClose

呼び出し形式

BOOL WINAPI DioDriverClose(void)

戻り値

TRUE:正常

FALSE:エラー

引数

なし

処理概要

本DLLの機能の使用を終了します。

使用例

```
BOOL bRet = DioDriverClose();
```

SetDinFilter

呼び出し形式

BOOL WINAPI SetDinFilter(BYTE Filter)

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Filter	チャタリングキャンセルパルス幅
DIN_FILTER_1MS	1.3msのフィルタ
DIN_FILTER_3MS	2.8msのフィルタ
DIN_FILTER_6MS	5.7msのフィルタ
DIN_FILTER_12MS	11.7msのフィルタ
DIN_FILTER_24MS	23.5msのフィルタ
DIN_FILTER_46MS	46.0msのフィルタ
DIN_FILTER_94MS	94.0msのフィルタ
DIN_FILTER_188MS	188.0msのフィルタ

処理概要

DIN入力のチャタリングキャンセルパルス幅を設定します。

チャタリングキャンセルパルス幅を設定した後は必ずラッチ状態をクリアしてください。

使用例

```
BOOL bRet = SetDinFilter(DIN_FILTER_1MS);
ClearDinLatchData(0x00);                    //全ビットラッチ状態クリア
```

GetDinData

呼び出し形式

BOOL WINAPI GetDinData(BYTE *pData)

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE *pData　　DIN入力データを取得する変数へのポインタ

D7bit　DIN7　1:ON / 0:OFF

D6bit　DIN6　1:ON / 0:OFF

D5bit　DIN5　1:ON / 0:OFF

D4bit　DIN4　1:ON / 0:OFF

D3bit　DIN3　1:ON / 0:OFF

D2bit　DIN2　1:ON / 0:OFF

D1bit　DIN1　1:ON / 0:OFF

D0bit　DIN0　1:ON / 0:OFF

処理概要

現在のDINポートの入力データを取得します。

使用例

```
BYTE byData;
BOOL bRet = GetDinData(&byData);
```

GetDinLatchData

呼び出し形式

BOOL WINAPI GetDinLatchData(BYTE *pData)

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE *pData　　DINラッチ入力データを取得する変数へのポインタ

D7bit　DIN7　1:ON / 0:OFF

D6bit　DIN6　1:ON / 0:OFF

D5bit　DIN5　1:ON / 0:OFF

D4bit　DIN4　1:ON / 0:OFF

D3bit　DIN3　1:ON / 0:OFF

D2bit　DIN2　1:ON / 0:OFF

D1bit　DIN1　1:ON / 0:OFF

D0bit　DIN0　1:ON / 0:OFF

処理概要

DINポートのラッチ入力データを取得します。

使用例

```
BYTE byData;
BOOL bRet = GetDinLatchData(&byData);
```

ClearDinLatchData

呼び出し形式

```
BOOL WINAPI ClearDinLatchData(BYTE Data)
```

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Data　　クリアするDINラッチの指定

D7bit　DIN7　1:保持 / 0:クリア

D6bit　DIN6　1:保持 / 0:クリア

D5bit　DIN5　1:保持 / 0:クリア

D4bit　DIN4　1:保持 / 0:クリア

D3bit　DIN3　1:保持 / 0:クリア

D2bit　DIN2　1:保持 / 0:クリア

D1bit　DIN1　1:保持 / 0:クリア

D0bit　DIN0　1:保持 / 0:クリア

処理概要

DINポートのラッチ状態をクリアします。

使用例

```
BOOL bRet = ClearDinLatchData(0x00);     // 全ビットクリア
```

SetDoutData

呼び出し形式

```
BOOL WINAPI SetDoutData(BYTE Data, BYTE Mask)
```

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Data　　DOUTに出力するデータ

D7bit　DOUT7　1:ON / 0:OFF

D6bit　DOUT6　1:ON / 0:OFF

D5bit　DOUT5　1:ON / 0:OFF

D4bit　DOUT4　1:ON / 0:OFF

D3bit　DOUT3　1:ON / 0:OFF

D2bit　DOUT2　1:ON / 0:OFF

D1bit　DOUT1　1:ON / 0:OFF

D0bit　DOUT0　1:ON / 0:OFF

BYTE Mask　　DOUTにデータを出力するビット指定

DOUT_MASK_7　DOUT7に出力する

DOUT_MASK_6　DOUT6に出力する

DOUT_MASK_5　DOUT5に出力する

DOUT_MASK_4　DOUT4に出力する

DOUT_MASK_3　DOUT3に出力する

DOUT_MASK_2　DOUT2に出力する

DOUT_MASK_1　DOUT1に出力する

DOUT_MASK_0　DOUT0に出力する

処理概要

DOUTポートへデータを出力します。

使用例

```
BOOL bRet = SetDoutData(0x00, DOUT_MASK_1 | DOUT_MASK_3 );
```

GetDoutData

呼び出し形式

```
BOOL WINAPI GetDoutData(BYTE *pData)
```

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE *pData　　DOUTに出力したデータを取得する変数へのポインタ

D7bit　DOUT7　1:ON / 0:OFF

D6bit　DOUT6　1:ON / 0:OFF

D5bit　DOUT5　1:ON / 0:OFF

D4bit　DOUT4　1:ON / 0:OFF

D3bit　DOUT3　1:ON / 0:OFF

D2bit　DOUT2　1:ON / 0:OFF

D1bit　DOUT1　1:ON / 0:OFF

D0bit　DOUT0　1:ON / 0:OFF

処理概要

DOUTポートへ出力したデータを取得します。

使用例

```
BYTE byData;
BOOL bRet = GetDoutData(&byData);
```

SetCounterCycle

呼び出し形式

```
BOOL WINAPI SetCounterCycle(BYTE Cycle)
```

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Cycle　　カウンタ機能でDINポートを監視する周期(1～99ms)

処理概要

カウンタでDINポートを監視する周期を指定します。

指定可能な周期は1～99msまでです。

周期を変更するにはカウンタイベントがすべて停止している必要があります。

カウンタイベントが動作中に周期が変更された場合にはすべてのカウンタイベントが停止した時に新しい周期に変更されます。

使用例

```
BOOL bRet = SetCounterCycle(50);
```

SetCounterEvent

呼び出し形式

```
int WINAPI SetCounterEvent(HANDLE Object, BYTE Trigger, WORD Count, BOOL Cyclic)
```

戻り値

0:正常

ERR_DIO_TRIGGER	既にトリガビットとして使用されている。
ERR_DIO_COUNT	カウント値が無効
ERR_DIO_OTHER	その他のエラー

引数

HANDLE Object	イベント発生時にアプリケーションへの通知に使用する オブジェクトハンドル
BYTE Trigger	カウンタとして監視するDINビット
DIN_TRIGGER_7	DIN7を監視する
DIN_TRIGGER_6	DIN6を監視する
DIN_TRIGGER_5	DIN5を監視する
DIN_TRIGGER_4	DIN4を監視する
DIN_TRIGGER_3	DIN3を監視する
DIN_TRIGGER_2	DIN2を監視する
DIN_TRIGGER_1	DIN1を監視する
DIN_TRIGGER_0	DIN0を監視する
WORD Count	イベントを発生させるまでのカウント値(1～65535)
BOOL Cyclic	TRUE:サイクリックイベント / FALSE:ワンショットイベント

処理概要

DINの各ビットが指定回数ONになったことをカウントしイベントをシグナル状態にする機能をビットごとに登録します。Triggerで指定したDINビットが指定回数ONになった時にObjectで指定されたイベントをシグナル状態にします。CyclicがTRUEに指定されていれば、指定回数ONになった時に再びイベントがシグナル状態となります。（サイクリックカウンタイベント機能）
CyclicがFALSEに指定されていれば、指定回数ONになった時にイベントがシグナル状態となり自動的にイベント設定登録が削除されます。（ワンショットカウンタイベント機能）
DINのそれぞれのビットは多重にTriggerに指定する事はできません。（DIN0～DIN7までなので、最大同時8個までのイベント制御が可能です。）
Triggerは同時に複数Bit指定する事はできません。カウンタイベント機能はDINラッチ入力を使用するので、カウンタイベント機能動作中はClearDinLatchDataを使用しないでください。
登録されたカウンタイベント機能は、StartCounterEventを使用してカウントを開始させなければなりません。

使用例

```
HANDLE  hEvent = CreateEvent(NULL, FALSE, FALSE, _T("DinEvent"));
int iRet = SetCounterEvent(hEvent, DIN_TRIGGER_0, 10, FALSE);
StartCounterEvent(DIN_TRIGGER_0);
WaitForSingleObject(hEvent, INFINITE);
```

StartCounterEvent

呼び出し形式

BOOL WINAPI StartCounterEvent(BYTE Trigger)

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Trigger	開始するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の監視を開始する
DIN_TRIGGER_6	DIN6の監視を開始する
DIN_TRIGGER_5	DIN5の監視を開始する
DIN_TRIGGER_4	DIN4の監視を開始する
DIN_TRIGGER_3	DIN3の監視を開始する
DIN_TRIGGER_2	DIN2の監視を開始する
DIN_TRIGGER_1	DIN1の監視を開始する
DIN_TRIGGER_0	DIN0の監視を開始する

処理概要

Triggerで指定したDINビットを監視するカウンタイベントを開始します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

使用例

```
BOOL bRet = StartCounterEvent(DIN_TRIGGER_0 | DIN_TRIGGER_1);
```

StopCounterEvent

呼び出し形式

BOOL WINAPI StopCounterEvent(BYTE Trigger)

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Trigger	停止するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の監視を停止する
DIN_TRIGGER_6	DIN6の監視を停止する
DIN_TRIGGER_5	DIN5の監視を停止する
DIN_TRIGGER_4	DIN4の監視を停止する
DIN_TRIGGER_3	DIN3の監視を停止する

DIN_TRIGGER_2	DIN2の監視を停止する
DIN_TRIGGER_1	DIN1の監視を停止する
DIN_TRIGGER_0	DIN0の監視を停止する

処理概要

Triggerで指定したDINビットを監視するカウンタイベントを停止します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

一度StopCounterEventで停止したカウンタイベントを、再びStartCounterEventで開始した場合、停止するまでのカウント回数は保持されずにクリアされ、新規にカウントが開始されます。

使用例

```
BOOL bRet = StopCounterEvent(DIN_TRIGGER_0 | DIN_TRIGGER_1);
```

ClearCounterEvent

呼び出し形式

```
BOOL WINAPI ClearCounterEvent(BYTE Trigger)
```

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Trigger	削除するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の監視登録を削除
DIN_TRIGGER_6	DIN6の監視登録を削除
DIN_TRIGGER_5	DIN5の監視登録を削除
DIN_TRIGGER_4	DIN4の監視登録を削除
DIN_TRIGGER_3	DIN3の監視登録を削除
DIN_TRIGGER_2	DIN2の監視登録を削除
DIN_TRIGGER_1	DIN1の監視登録を削除
DIN_TRIGGER_0	DIN0の監視登録を削除

処理概要

Triggerで指定したDINビットの監視登録を削除します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

使用例

```
BOOL bRet = ClearCounterEvent(DIN_TRIGGER_0 | DIN_TRIGGER_1);
```

GetCounterStat

呼び出し形式

BOOL WINAPI GetCounterStat(BYTE Trigger, BYTE *pStat, WORD *pCount)

戻り値

TRUE:正常

FALSE:エラー

引数

BYTE Trigger	状態を取得するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の状態を取得
DIN_TRIGGER_6	DIN6の状態を取得
DIN_TRIGGER_5	DIN5の状態を取得
DIN_TRIGGER_4	DIN4の状態を取得
DIN_TRIGGER_3	DIN3の状態を取得
DIN_TRIGGER_2	DIN2の状態を取得
DIN_TRIGGER_1	DIN1の状態を取得
DIN_TRIGGER_0	DIN0の状態を取得
BYTE *pStat	カウンタイベントの状態を取得する変数へのポインタ
EVENT_NOREGIST	カウンタイベントが登録されていない
EVENT_ONESHOT_STOP	ワンショットカウンタイベントが停止中
EVENT_ONESHOT_RUN	ワンショットカウンタイベントがカウント中
EVENT_CYCLIC_STOP	サイクリックカウンタイベントが停止中
EVENT_CYCLIC_RUN	サイクリックカウンタイベントがカウント中
BYTE *pCount	カウンタの現在値を取得する変数へのポインタ

処理概要

Triggerで指定したDINビットの監視登録状態を取得します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

同時に複数のDINビットを指定することはできません。

使用例

```
BYTE byStat;
WORD wCount;
BOOL bRet = GetCounterStat(DIN_TRIGGER_0, &byStat, &wCount);
```

5.2.2 eMbedded Visual Basic用　関数詳細仕様

DioDriverOpen

呼び出し形式

Declare Function DioDriverOpen Lib "Dio_Api.DLL" () As Long

戻り値

TRUE:正常

FALSE:エラー

引数

なし

処理概要

本DLLの機能を使用可能にします。

多重Openはできません。

使用例

```
Dim ret As Long
ret = DioDriverOpen()
```

DioDriverClose

呼び出し形式

Declare Function DioDriverClose Lib "Dio_Api.DLL" () As Long

戻り値

TRUE:正常

FALSE:エラー

引数

なし

処理概要

本DLLの機能の使用を終了します。

使用例

```
Dim ret As Long
ret = DioDriverClose()
```

SetDinFilter

呼び出し形式

Declare Function SetDinFilter Lib "Dio_Api.DLL" (ByVal byFilter As Long) As Long

戻り値

TRUE:正常

FALSE:エラー

引数

byFilter As Long	チャタリングキャンセルパルス幅
DIN_FILTER_1MS	1.3msのフィルタ
DIN_FILTER_3MS	2.8msのフィルタ
DIN_FILTER_6MS	5.7msのフィルタ
DIN_FILTER_12MS	11.7msのフィルタ
DIN_FILTER_24MS	23.5msのフィルタ
DIN_FILTER_46MS	46.0msのフィルタ
DIN_FILTER_94MS	94.0msのフィルタ
DIN_FILTER_188MS	188.0msのフィルタ

処理概要

DIN入力のチャタリングキャンセルパルス幅を設定します。

チャタリングキャンセルパルス幅を設定した後は、必ずラッチ状態をクリアしてください。

使用例

```
Dim ret As Long
ret = SetDinFilter(DIN_FILTER_1MS)
ClearDinLatchData(0)
```

GetDinData

呼び出し形式

```
Declare Function GetDinData Lib "Dio_Api.DLL" (ByRef pbyData As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

pbyData As Long	DIN入力データを取得する変数

D7bit　DIN7　1:ON / 0:OFF

D6bit　DIN6　1:ON / 0:OFF

D5bit　DIN5　1:ON / 0:OFF

D4bit　DIN4　1:ON / 0:OFF

D3bit　DIN3　1:ON / 0:OFF

D2bit　DIN2　1:ON / 0:OFF

D1bit　DIN1　1:ON / 0:OFF

D0bit　DIN0　1:ON / 0:OFF

処理概要

現在のDINポートの入力データを取得します。

使用例

```
Dim ret As Long
Dim pbyData As Long
ret = GetDinData(pbyData)
```

GetDinLatchData

呼び出し形式

```
Declare Function GetDinLatchData Lib "Dio_Api.DLL" (ByRef pbyData As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

pbyData As Long　　DINラッチ入力データを取得する変数

D7bit　DIN7　1:ON / 0:OFF

D6bit　DIN6　1:ON / 0:OFF

D5bit　DIN5　1:ON / 0:OFF

D4bit　DIN4　1:ON / 0:OFF

D3bit　DIN3　1:ON / 0:OFF

D2bit　DIN2　1:ON / 0:OFF

D1bit　DIN1　1:ON / 0:OFF

D0bit　DIN0　1:ON / 0:OFF

処理概要

DINポートのラッチ入力データを取得します。

使用例

```
Dim ret As Long
Dim pbyData As Long
ret = GetDinLatchData(pbyData);
```

ClearDinLatchData

呼び出し形式

```
Declare Function ClearDinLatchData Lib "Dio_Api.DLL" (ByVal byData As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

byData As Long　クリアするDINラッチの指定

D7bit　DIN7　　1:保持 / 0:クリア

D6bit　DIN6　　1:保持 / 0:クリア

D5bit　DIN5　　1:保持 / 0:クリア

D4bit　DIN4　　1:保持 / 0:クリア

D3bit　DIN3　　1:保持 / 0:クリア

D2bit　DIN2　　1:保持 / 0:クリア

D1bit　DIN1　　1:保持 / 0:クリア

D0bit　DIN0　　1:保持 / 0:クリア

処理概要

DINポートのラッチ状態をクリアします。

使用例

```
Dim ret As Long
Rem 全ビットクリア
ret = ClearDinLatchData(0)
```

SetDoutData

呼び出し形式

```
Declare Function SetDoutData Lib "Dio_Api.DLL"
(ByVal byData As Long, ByVal byMask As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

byData As Long　DOUTに出力するデータ

D7bit　DOUT7　　1:ON / 0:OFF

D6bit　DOUT6　　1:ON / 0:OFF

D5bit　DOUT5　　1:ON / 0:OFF

D4bit　DOUT4　　1:ON / 0:OFF

D3bit　DOUT3　　1:ON / 0:OFF

D2bit　DOUT2　　1:ON / 0:OFF

D1bit　DOUT1　　1:ON / 0:OFF

D0bit　DOUT0　1:ON / 0:OFF

byMask As Long　DOUTにデータを出力するビット指定

DOUT_MASK_7　DOUT7に出力する

DOUT_MASK_6　DOUT6に出力する

DOUT_MASK_5　DOUT5に出力する

DOUT_MASK_4　DOUT4に出力する

DOUT_MASK_3　DOUT3に出力する

DOUT_MASK_2　DOUT2に出力する

DOUT_MASK_1　DOUT1に出力する

DOUT_MASK_0　DOUT0に出力する

処理概要

DOUTポートへデータを出力します。

使用例

```
Dim ret As Long
ret = SetDoutData(0, DOUT_MASK_1 Or DOUT_MASK_3 )
```

GetDoutData

呼び出し形式

```
Declare Function GetDoutData Lib "Dio_Api.DLL" (ByRef pbyData As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

pbyData As Long　　DOUTに出力したデータを取得する変数

D7bit　DOUT7　1:ON / 0:OFF

D6bit　DOUT6　1:ON / 0:OFF

D5bit　DOUT5　1:ON / 0:OFF

D4bit　DOUT4　1:ON / 0:OFF

D3bit　DOUT3　1:ON / 0:OFF

D2bit　DOUT2　1:ON / 0:OFF

D1bit　DOUT1　1:ON / 0:OFF

D0bit　DOUT0　1:ON / 0:OFF

処理概要

DOUTポートへ出力したデータを取得します。

使用例

```
Dim ret As Long
Dim pbyData As Long
ret = GetDoutData(pbyData)
```

SetCounterCycle

呼び出し形式

```
Declare Function SetCounterCycle Lib "Dio_Api.DLL" (ByVal byCylie As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

bicycle As Long　　カウンタ機能でDINポートを監視する周期(1～99ms)

処理概要

カウンタでDINポートを監視する周期を指定します。

指定可能な周期は1～99msまでです。

周期を変更するにはカウンタイベントがすべて停止している必要があります。

カウンタイベントが動作中に周期が変更された場合にはすべてのカウンタイベントが停止した時に新しい周期に変更されます。

使用例

```
Dim ret As Long
ret = SetCounterCycle(50)
```

SetCounterEvent

呼び出し形式

```
Declare Function SetCounterEvent Lib "Dio_Api.DLL"
(ByVal hEvent As Long, ByVal byTrigger As Long, ByVal wCount As Long,
ByVal bCyclic As Long) As Long
```

戻り値

0:正常

ERR_DIO_TRIGGER　　既にトリガビットとして使用されている。

ERR_DIO_COUNT　　カウント値が無効

ERR_DIO_OTHER　　その他のエラー

引数

hEvent As Long　　イベント発生時にアプリケーションへの通知に使用する

オブジェクトハンドル

byTrigger As Long　　カウンタとして監視するDINビット

DIN_TRIGGER_7　　DIN7を監視する

DIN_TRIGGER_6　　DIN6を監視する

DIN_TRIGGER_5　　DIN5を監視する

DIN_TRIGGER_4　　DIN4を監視する

DIN_TRIGGER_3　　DIN3を監視する

DIN_TRIGGER_2　　DIN2を監視する

DIN_TRIGGER_1　　DIN1を監視する

DIN_TRIGGER_0　　DIN0を監視する

wCount As Long　　イベントを発生させるまでのカウント値(1～65535)

bCyclic As Long TRUE:サイクリックイベント / FALSE:ワンショットイベント

処理概要

DINの各ビットが指定回数Onになったことをカウントしイベントをシグナル状態にする機能をビットごとに登録します。Triggerで指定したDINビットが指定回数ONになった時にObjectで指定されたイベントをシグナル状態にします。CyclicがTRUEに指定されていれば、指定回数ONになった時に再びイベントがシグナル状態となります。（サイクリックカウンタイベント機能）

CyclicがFALSEに指定されていれば指定回数ONになった時にイベントがシグナル状態となり自動的にイベント設定登録が削除されます。（ワンショットカウンタイベント機能）

DINのそれぞれのビットは多重にTriggerに指定することはできません。（DIN0～DIN7までなので最大同時8個までのイベント制御が可能です。）

Triggerは同時に複数ビット指定することはできません。カウンタイベント機能はDINラッチ入力を使用するのでカウンタイベント機能動作中はClearDinLatchDataを使用しないでください。

登録されたカウンタイベント機能はStartCounterEventを使用してカウントを開始させなければなりません。

使用例

```
Dim ret As Long
Dim hEvent As Long
hEvent = CreateEvent(0, False, False, "DinEvent")
ret = SetCounterEvent(hEvent, DIN_TRIGGER_0, 10, FALSE)
StartCounterEvent(DIN_TRIGGER_0)
WaitForSingleObject(hEvent, INFINITE)
```

StartCounterEvent

呼び出し形式

```
Declare Function StartCounterEvent Lib "Dio_Api.DLL" (ByVal byTrigger As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

byTrigger As Long	開始するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の監視を開始する
DIN_TRIGGER_6	DIN6の監視を開始する
DIN_TRIGGER_5	DIN5の監視を開始する
DIN_TRIGGER_4	DIN4の監視を開始する
DIN_TRIGGER_3	DIN3の監視を開始する
DIN_TRIGGER_2	DIN2の監視を開始する
DIN_TRIGGER_1	DIN1の監視を開始する
DIN_TRIGGER_0	DIN0の監視を開始する

処理概要

Triggerで指定したDINビットを監視するカウンタイベントを開始します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

使用例

```
Dim ret As Long
ret = StartCounterEvent(DIN_TRIGGER_0 Or DIN_TRIGGER_1)
```

StopCounterEvent

呼び出し形式

```
Declare Function StopCounterEvent Lib "Dio_Api.DLL" (ByVal byTrigger As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

byTrigger As Long	停止するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の監視を停止する
DIN_TRIGGER_6	DIN6の監視を停止する
DIN_TRIGGER_5	DIN5の監視を停止する
DIN_TRIGGER_4	DIN4の監視を停止する
DIN_TRIGGER_3	DIN3の監視を停止する
DIN_TRIGGER_2	DIN2の監視を停止する
DIN_TRIGGER_1	DIN1の監視を停止する
DIN_TRIGGER_0	DIN0の監視を停止する

処理概要

Triggerで指定したDINビットを監視するカウンタイベントを停止します。

カウンターイベントはSetCounterEventで登録されていなければなりません。

一度StopCounterEventで停止したカウンタイベントを、再びStartCounterEventで開始した場合、停止するまでのカウント回数は保持されずにクリアされ、新規にカウントが開始されます。

使用例

```
Dim ret As Long
ret = StopCounterEvent(DIN_TRIGGER_0 Or DIN_TRIGGER_1)
```

ClearCounterEvent

呼び出し形式

```
Declare Function ClearCounterEvent Lib "Dio_Api.DLL" (ByVal byTrigger As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

byTrigger As Long	削除するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の監視登録を削除
DIN_TRIGGER_6	DIN6の監視登録を削除
DIN_TRIGGER_5	DIN5の監視登録を削除
DIN_TRIGGER_4	DIN4の監視登録を削除
DIN_TRIGGER_3	DIN3の監視登録を削除
DIN_TRIGGER_2	DIN2の監視登録を削除
DIN_TRIGGER_1	DIN1の監視登録を削除
DIN_TRIGGER_0	DIN0の監視登録を削除

処理概要

Triggerで指定したDINビットの監視登録を削除します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

使用例

```
Dim ret As Long
ret = ClearCounterEvent(DIN_TRIGGER_0 Or DIN_TRIGGER_1)
```

GetCounterStat

呼び出し形式

```
Declare Function GetCounterStat Lib "Dio_Api.DLL"
(ByVal byTrigger As Long, ByRef pbyStat As Long, ByRef pwCount As Long) As Long
```

戻り値

TRUE:正常

FALSE:エラー

引数

byTrigger As Long	状態を取得するカウンタイベントのDINビット
DIN_TRIGGER_7	DIN7の状態を取得
DIN_TRIGGER_6	DIN6の状態を取得
DIN_TRIGGER_5	DIN5の状態を取得
DIN_TRIGGER_4	DIN4の状態を取得
DIN_TRIGGER_3	DIN3の状態を取得
DIN_TRIGGER_2	DIN2の状態を取得
DIN_TRIGGER_1	DIN1の状態を取得
DIN_TRIGGER_0	DIN0の状態を取得
pbyStat As Long	カウンタイベントの状態を取得する変数
EVENT_NOREGIST	カウンタイベントが登録されていない
EVENT_ONESHOT_STOP	ワンショットカウンタイベントが停止中
EVENT_ONESHOT_RUN	ワンショットカウンタイベントがカウント中
EVENT_CYCLIC_STOP	サイクリックカウンタイベントが停止中
EVENT_CYCLIC_RUN	サイクリックカウンタイベントがカウント中
pwCount As Long	カウンタの現在値を取得する変数

処理概要

Triggerで指定したDINビットの監視登録状態を取得します。

カウンタイベントはSetCounterEventで登録されていなければなりません。

同時に複数のDINビットを指定することはできません。

使用例

```
Dim ret As Long
Dim pbyStat As Long
Dim pwCount As Long
ret = GetCounterStat(DIN_TRIGGER_0, pbyStat, pwCount)
```