イメージセンサ用駆動回路ライブラリ

DCamUSB　　：バージョン 2.1.1.0
DCamTmpCtrl：バージョン 2.1.0.0

関数仕様書

文　書　番　号　　：K46-B60010
ドキュメントリビジョン：2.02　2014年7月29日

浜松ホトニクス株式会社

ソフトウェアの使用許諾条件

以下の条件（以下「本許諾条件」といいます。）をよくお読みください。

浜松ホトニクス株式会社（以下、「弊社」といいます。）は、本許諾条件を承諾する弊社 MCD ヘッド製品ならびに駆動回路製品（以下、「弊社製品」といいます。）のユーザーに対してのみ「アプリケーションソフトウェア及び DLL ファイル等のソフトウェア類」（以下、「本ソフト」といいます）の使用を許諾します。本ソフトをインストール又はこれらファイルをコピーすることによって、ユーザーは本許諾条件に拘束されることに同意したものとみなされます。本許諾条件の全部または一部に同意されない場合、ユーザーは本ソフトをインストールならびに使用することはできません。

1. 本ソフトの目的

本ソフトは、弊社製品を簡易にお使いになりたいユーザーの便宜を考え、無償かつ無保証で使用許諾される制御用ソフトウエアです。本ソフトはユーザーの責任と判断でご使用下さい。

2. 使用許諾

弊社は、本許諾条件に同意したユーザーに対してのみ、本ソフトをインストールし弊社製品の制御ならびに弊社製品を使用した計測を実施する目的でのみ使用する権利を許諾します。

3. 著作権その他の権利の帰属

本ソフトおよび付属文書に関する所有権、知的財産権その他一切の権利は弊社に帰属します。本ソフトは、著作権法等の知的財産権に関する法令ならびに国際条約により保護されています。ユーザーは、本ソフトあるいは付属文書に付された権利表示を改変あるいは除去してはいけません。

本許諾条件により明示的に許諾された事項を除き、弊社はユーザーに対していかなる権利も譲渡または付与するものではなく、本ソフトおよび付属文書に関する全ての権利は弊社に留保されます。

4. 複製

ユーザーは、本許諾条件のあらゆる条項を遵守することを条件に、本ソフトをバックアップの目的で複製することができます。

5. 禁止条項

ユーザーは、以下のことを行うことはできません。ただし、ユーザーが弊社製品を第三者に譲渡またはリースもしくは貸与する場合に、弊社製品とともに本ソフトを当該第三者に引き渡す場合において、当該第三者が本許諾条件に同意する場合には、弊社は当該第三者に対して引き続き本許諾条件のもとで本ソフトの使用を許諾します。

① 第三者に対し、本ソフトを販売その他頒布し、または販売その他頒布を目的とした宣伝、展示、使用、複製、営業等を行うこと。
② 第三者に対し、本ソフトの使用権を譲渡あるいは再許諾すること。
③ 第三者に対し、本ソフトを貸与、リースもしくは担保設定すること。
④ 本許諾書その他の付属文書を含め、本ソフトの一部または全部を改変あるいは除去すること。
改変にはファイル名の変更も含まれます。
⑤ 本ソフトウエアの全部若しくは一部を複製したり、翻案、翻訳、リバースエンジニアリング、逆アセンブル又は逆コンパイルまたはその他の方法でソースコードを解明しようと試みること。

6. 責任の制限

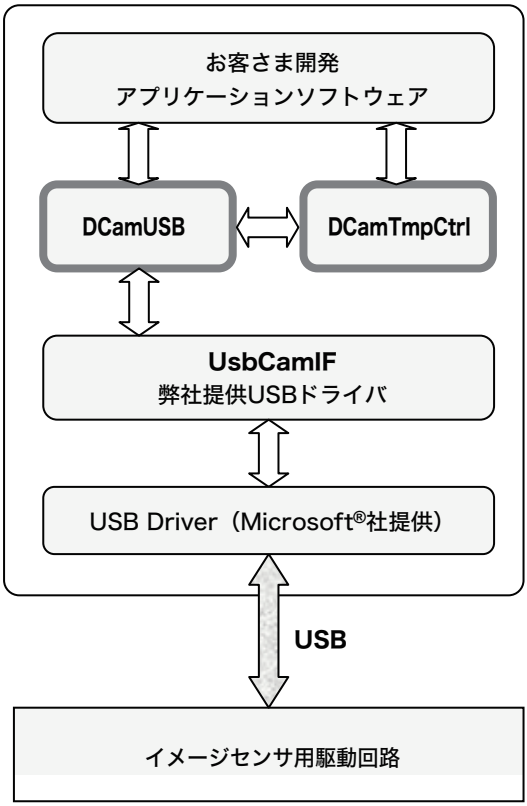
弊社は、本ソフトおよび付属文書について、その品質、性能または特定目的に対する適合性を含め、一切保証はいたしません。いかなる場合においても、本ソフトおよび付属文書の使用または使用不能から生じるコンピュータの故障または損傷、情報の消失、その他あらゆる直接的および間接的損害に関し、弊社は一切責任を負いません。また、本ソフトウエアについてメンテナンスやサポートをするものではなく、不具合や障害等が生じた場合においても、改修・修復等を含め何らの責任も負うものではありません。

弊社は、改良の為に本ソフトの変更を予告なしに行う事があります。

目　　次

1 概要

イメージセンサ用駆動回路ライブラリ (以後 DCamUSB と略す) ならびに温度制御ライブラリ（以後、DCamTmpCtrl と略す）は、弊社のイメージセンサ用駆動回路を制御するためのライブラリです。このライブラリを使用することによりソフトウェア開発を容易に行うことが可能になります。



2 動作環境

■ 対応 OS

Microsoft® Windows 7®（Service Pack 1 以上）

■ CPU

OS の推奨システム環境に準ずる

■ メモリ

OS の推奨システム環境に準ずる

3 開発環境の構築

開発環境の所定のフォルダに DCamUSB.dll、DCamTmpCtrl.dll をコピーしてください。
開発の際に DCamUSB.h、DCamTmpCtrl.h、DCamUSB.lib、DCamTmpCtrl.lib が必要な場合は同様にコピーが必要です。その場合、DCamStatusCode.h もコピーしてください。
また、インストーラを使用してドライバをインストールする際は、"USB カメラモジュール 制御用ドライバソフトウェア インストール説明書 " を参照してください。

NOTE

温度制御機能を搭載しない駆動回路では、
DcomTmpCtrl.dll
DcomTmpCtrl.h
DcomTmpCtrl.lib
は必要ありません。

4 開発に必要なファイル

本ライブラリは、以下のファイルで構成されます。

ライブラリファイル	：	DCamUSB.dll、DCamTmpCtrl.dll
ヘッダーファイル	：	DCamUSB.h、DCamTmpCtrl.h DCamStatusCode.h
インポートライブラリ	：	DCamUSB.lib、DCamTmpCtrl.lib
ドライバ	：	UsbCamIF.sys
ドライバ情報ファイル	：	UsbCamIF.inf

5 関数一覧

本ライブラリでは、以下の機能を提供します

■ DCamUSB.dll

1.	DcamInitialize	ライブラリを初期化します。
2.	DcamUninitialize	ライブラリの終了処理をします。
3.	DcamOpen	デバイスをオープンします。
4.	DcamClose	デバイスをクローズします。
5.	DcamGetDeviceState	デバイスの接続状態を取得します。
6.	DcamGetImageSize	画像サイズを取得します。
7.	DcamGetBitPerPixel	1 ピクセルあたりのビット数を取得します。
8.	DcamSetCCDType	センサのタイプを設定します。
9.	DcamGetCCDType	センサのタイプを取得します。
10.	DcamSetMeasureDataCount	計測データのライン数を設定します。
11.	DcamGetMeasureDataCount	計測データのライン数を取得します。
12.	DcamGetCaptureBytes	1 フレームのバイト数を取得します。
13.	DcamGetTotalCaptureBytes	1 回のデータ取得あたりの総バイト数を取得します。
14.	DcamCapture	画像取込みを実行します。
15.	DcamCaptureReverseX	画像取込みを実行します。データは X 軸反転されます。
16.	DcamStop	画像取込みを中断します。
17.	DcamStopEx	画像取込みを中断します。
18.	DcamWait	画像取得の完了を待ちます。
19.	DcamSetGain	ゲインを設定します。
20.	DcamGetGain	ゲインを取得します。
21.	DcamSetOffset	オフセットを設定します。
22.	DcamGetOffset	オフセットを取得します。
23.	DcamSetBinning	ビニングモードを設定します。
24.	DcamGetBinning	ビニングモードを取得します。
25.	DcamSetTriggerMode	トリガモードを設定します。
26.	DcamGetTriggerMode	トリガモードを取得します。
27.	DcamSetTriggerPolarity	トリガ極性を設定します。
28.	DcamGetTriggertPolarity	トリガ極性を取得します。
29.	DcamSetExposureTime	露光時間を設定します。
30.	DcamGetExposureTime	露光時間を取得します。
31.	DcamSetOperatingMode	CCD 動作モードを設定します。
32.	DcamGetOperatingMode	CCD 動作モードを取得します。
33.	DcamSetLEDOperatingMode	LED 発光制御モードを設定します。
34.	DcamGetLEDOperatingMode	LED 発光制御モードを取得します。
35.	DcamSetStandardTimeUnit	基準時間単位を設定します。
36.	DcamGetStandardTimeUnit	基準時間単位を取得します。
37.	DcamSetOutPulse	パルス信号出力条件を設定します。
38.	DcamGetOutPulse	パルス信号出力条件を取得します。
39.	DcamLoadParameters	デバイスに保存されている設定を読み込みます。
40.	DcamStoreParameters	デバイスに現在の設定を保存します。
41.	DcamGetVersion	ライブラリのバージョン情報を取得します。
42.	DcamGetDriverVersion	ドライバのバージョン情報を取得します。

43.	DcamGetFirmwareVersion	ファームのバージョン情報を取得します。
44.	DcamGetDeviceInformation	デバイス情報を取得します。
45.	DcamGetTransferRateType	USB 転送速度タイプを取得します。
46.	DcamGetLastError	最新のエラーコードを取得します。
47.	DcamSetOverClock	オーバークロックを設定します。
48.	DcamGetOverClock	オーバークロックを取得します。
49.	DcamSetMPPMode	MPP モードを設定します。
50.	DcamGetMPPMode	MPP モードを取得します。
51.	DcamSetLineTime	LineTime を設定します。
52.	DcamGetLineTime	LineTime を取得します。
53.	DcamSetIntegralCapacity	静電容量を設定します。
54.	DcamGetIntegralCapacity	静電容量を取得します。
55.	DcamSetDriveMode	CCD 駆動モードを設定します。
56.	DcamGetDriveMode	CCD 駆動モードを取得します。
57.	DcamSetElectronicShutter	Erectronic シャッターモードを設定します。
58	DcamGetElectronicShutter	Erectronic シャッターモードを取得します。
59	DcamSetSensorSignalPulseWidth	TG のパルス幅を設定します。
60	DcamGetSensorSignalPulseWidth	TG のパルス幅を取得します。

■ DCamTmpCtrl.dll

1.	DcamTmpCtrlInitialize	温度制御ライブラリを初期化します。
2.	DcamTmpCtrlUninitialize	温度制御ライブラリの終了処理をします。
3.	DcamTmpCtrlGetLastError	最新のエラーコードを取得します。
4.	DcamTmpCtrlSetCoolingControl	冷却制御の状態を設定します。
5.	DcamTmpCtrlGetCoolingControl	冷却制御の状態を取得します。
6.	DcamTmpCtrlLoadCoolingTemperature	冷却温度 (設定値) を読み込みます。
7.	DcamTmpCtrlSaveCoolingTemperature	冷却温度 (設定値) を保存します。
8.	DcamTmpCtrlGetCoolingTemperature	冷却温度 (設定値) を取得します。
9.	DcamTmpCtrlSetCoolingTemperature	冷却温度 (設定値) を設定します。
10.	DcamTmpCtrlGetCoolingTemperatureCurrent	冷却温度 (現在値) を取得します。
11.	DcamTmpCtrlGetCoolingStatus	冷却状態を取得します。
12.	DcamTmpCtrlGetThermistorStatus	サーミスターの状態を取得します。

5.1 パラメータ定義

5.1.1 DCamUSB.h

■ デバイス接続状態

DCAM_DEVSTATE_NON	未接続、デバイスなし
DCAM_DEVSTATE_DEVICE	未接続、デバイスあり
DCAM_DEVSTATE_NODEVICE	接続済み、デバイスなし
DCAM_DEVSTATE_CONNECT	接続済み、デバイスあり
DCAM_DEVSTATE_BOOT	接続済み、デバイスあり（起動中）

■ ピクセルあたりのビット数

DCAM_BITPIXEL_8	8 ビット
DCAM_BITPIXEL_10	10 ビット
DCAM_BITPIXEL_12	12 ビット
DCAM_BITPIXEL_14	14 ビット
DCAM_BITPIXEL_16	16 ビット

■ 画像取得

DCAM_WAITSTATUS_COMPLETED	画像取得完了
DCAM_WAITSTATUS_UNCOMPLETED	画像取得未完了
DCAM_WAIT_INFINITE	画像取得完了まで待ちます。

■ ビニング

DCAM_BINNING_AREA	エリアスキャニング
DCAM_BINNING_FULL	フルラインビニング

■ トリガモード

DCAM_TRIGMODE_INT	内部同期モード（「INT」モード）
DCAM_TRIGMODE_EXT_EDGE	外部同期モード 1 （「EXT.EDGE」モード）
DCAM_TRIGMODE_EXT_LEVEL	外部同期モード 2 （「EXT.LEVEL」モード）
DCAM_TRIGMODE_GS_INT	GS 内部同期モード（「INT」モード）
DCAM_TRIGMODE_GS_EXT_EDGE	GS 外部同期モード 1 （「EXT.EDGE1」モード）
DCAM_TRIGMODE_GS_EXT_ONE_SHOT	GS 外部同期モード 2 （「EXT.EDGE2」モード）
DCAM_TRIGMODE_GS_EXT_GATED	GS 外部同期ゲートモード （「EXT. GATED」モード）
DCAM_TRIGMODE_RS_INT	RS 内部同期モード（「INT」モード）
DCAM_TRIGMODE_RS_EXT_EDGE	RS 外部同期モード 1 （「EXT.EDGE1」モード）

DCAM_TRIGMODE_RS_EXT_ONE_SHOT	RS 外部同期モード 2 (「EXT. EDGE2」モード)
DCAM_TRIGMODE_RS_EXT_GATED	RS 外部同期ゲートモード (「EXT. GATED」モード)

※ その他トリガモードは、「DCamUSB.h」を参照してください。

■ トリガ極性

DCAM_TRIGPOL_POSITIVE	正
DCAM_TRIGPOL_NEGATIVE	負

■ センサータイプ

DCAM_CCD_TYPE0	2068x1(S10420-1106), 2068x70(S10420-1106-01, S11071-1106)
DCAM_CCD_TYPE2	2068x22(S10420-1104-01, S11071-1104)
DCAM_CCD_TYPE3	1044x1(S10420-1006), 1044x70(S10420-1006-01, S11071-1006)
DCAM_CCD_TYPE5	1044x22(S10420-1004-01, S11071-1004)
DCAM_CCD_TYPE10	64x64(G11097)

※ その他センサタイプは、「DCamUSB.h」を参照してください。

■ CCD 駆動モード

DCAM_CCDDRVMODE_SUSPEND	Suspend モード
DCAM_CCDDRVMODE_STANDBY	Stanadby モード

■ CCD 動作モード

DCAM_OPMODE_DARKCURRENT	低暗電流動作モード
DCAM_OPMODE_SATURATION	大飽和動作モード

■ LED 制御モード

DCAM_LEDOPMODE_OFF	LED 発光無し (常にオフ)
DCAM_LEDOPMODE_ON	LED 発光有り

■基準時間単位タイプ

DCAM_TIME_UNIT_TYPE1	トリガ [msec]、パルス出力 [msec]
DCAM_TIME_UNIT_TYPE2	トリガ [usec]、パルス出力 [usec]
DCAM_TIME_UNIT_TYPE3	トリガ [msec]、パルス出力 [usec]
DCAM_TIME_UNIT_TYPE4	トリガ [Clock]、パルス出力 [Clock]

■パルス出力モード

DCAM_OUTMODE_NOTOUTPUT	パルス出力 OFF
DCAM_OUTMODE_PLS_DT_PW	パルス出力 ON(ディレイ時間 + パルス幅)
DCAM_OUTMODE_PLS_ACCUM	パルス出力 ON(蓄積時間より計算)

■パルス出力極性

DCAM_OUTPOL_POSITIVE	正
DCAM_OUTPOL_NEGATIVE	負

■ デバイス情報の種類

DCAM_DEVINF_TYPE	型番
DCAM_DEVINF_SERIALNO	シリアル番号
DCAM_DEVINF_VERSION	バージョン

■ USB 転送速度タイプ

DCAM_TRANSRATE_USB11	USB 1.1 規格
DCAM_TRANSRATE_USB20	USB 2.0 規格

■ MPP モード

DCAM_CCDMPPMODE_OFF	MPP モード OFF
DCAM_CCDMPPMODEON	MPP モード ON

■ Electronic シャッターモード

DCAM_CCDESHUTTER_OFF	Electronic シャッターモード OFF
DCAM_CCDESHUTTER_ON	Electronic シャッターモード ON

5.1.2 DCamTmpCtrl.h

■冷却制御状態

DCAM_COOLING_CONTROL_OFF	冷却装置 OFF
DCAM_COOLING_CONTROL_ON	冷却装置 ON

■冷却温度状態

DCAM_COOLING_STATUS_NORMAL	冷却温度が許容範囲内である
DCAM_COOLING_STATUS_LOWER	冷却温度が許容範囲より低い
DCAM_COOLING_STATUS_HIGHER	冷却温度が許容範囲より高い

■サーミスター状態

DCAM_THERMISTOR_STATUS_NOERROR	サーミスター正常
DCAM_THERMISTOR_STATUS_ERROR	サーミスター異常
DCAM_THERMISTOR_STATUS_OVER	サーミスター温度異常

5.2 エラーコード表（実行ステータス）

5.2.1 DCamStatusCode.h

0	dcCode_Success	正常終了
1	dcCode_Unknown	未知のエラーが発生しました
2	dcCode_NoInit	ライブラリが初期化されていません
3	dcCode_AlreadyInit	他で使用されています
4	dcCode_NoDriver	ドライバが存在しません
5	dcCode_NoMemory	メモリが不足しています
6	dcCode_NotConnected	接続が行われていません
9	dcCode_InvalidParam	パラメータが不正です
100	dcCode_DeviceDefect	デバイスの状態が正常ではありません
111	dcCode_Timeout	タイムアウトが発生しました
120	dcCode_AlreadyStarted	既に開始状態です
200	dcCode_CoolingOn	冷却制御をしています
201	dcCode_CoolingOff	冷却制御をしていません

6 関数詳細

6.1 DCamUSB.dll

6.1.1 初期化関数（DcamInitialize）

BOOL DcamInitialize(VOID)

【概要】
本ライブラリを初期化します。

【引数】
なし

【戻り値】
正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】
この関数は必ず最初に実行します。
既に初期化されている場合はエラーになります。
OS 内で 1 つのプロセスのみ使用することができます。
尚、本関数は、「DcamUnitialize」とセットで使用します。

【参照】
DcamUninitialize, DcamOpen, DcamClose

【例】
以下が関数の呼び出し例です。

```
DWORD  dwErrCode;
if(DcamInitialize() != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.2　終了処理関数（DcamUninitialize）

BOOL DcamUninitialize(VOID)

【概要】

本ライブラリのリソースの開放、およびデバイスドライバのクローズをします。

【引数】

なし

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

本関数は、「DcamInitialize」とセットで使用します。
また、プログラムの終了時または本ライブラリが不要になった時に必ず呼び出してください。

【参照】

DcamInitialize, DcamOpen, DcamClose

【例】

以下が関数の呼び出し例です。

```
DWORD  dwErrCode;
if(DcamUninitialize() != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.3　オープン処理関数（DcamOpen）

BOOL DcamOpen(VOID)

【概要】
デバイスのオープン処理を行います。

【引数】
なし

【戻り値】
正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】
なし

【参照】
DcamClose, DcamInitialize, DcamUninitialize

【例】
以下が関数の呼び出し例です。

```
DWORD  dwErrCode;

// Initialize Library
if(DcamInitialize() != TRUE){
    dwErrCode = DcamGetLastError();
    return;
}

// Open Device
if(DcamOpen() != TRUE){
    dwErrCode = DcamGetLastError();
    return;
}
```

6.1.4　クローズ処理関数（DcamClose）

BOOL DcamClose(VOID)

【概要】
デバイスのクローズ処理を行います。

【引数】
なし

【戻り値】
正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】
なし

【参照】
DcamOpen, DcamInitialize, DcamUninitialize

【例】
以下が関数の呼び出し例です。

```
DWORD  dwErrCode;

// Close Device
if(DcamClose() != TRUE){
    dwErrCode = DcamGetLastError();
    return;
}

// Uninitialize Library
if(DcamUninitialize() != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.5 デバイス接続状態確認関数（DcamGetDeviceState）

BOOL DcamGetDeviceState(INT* pState)

【概要】

デバイス接続状況を取得します。

【引数】

pState　現在のデバイス接続状態を格納する変数のアドレスを指定します。
以下のいずれかになります。

DCAM_DEVSTATE_NON	：未接続、デバイスなし
DCAM_DEVSTATE_DEVICE	：未接続、デバイスあり
DCAM_DEVSTATE_NODEVICE	：接続済み、デバイスなし
DCAM_DEVSTATE_CONNECT	：接続済み、デバイスあり
DCAM_DEVSTATE_BOOT	：接続済み、デバイスあり（起動中）

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamOpen, DcamClose, DcamInitialize, DcamUninitialize

【例】

以下が関数の呼び出し例です。

```
INT         nState;
DWORD   dwErrCode;

// Get device state
if(DcamGetDeviceState(&nState) != TRUE){
    dwErrCode = DcamGetLastError();
}

// Remove the device
if(nState == DCAM_DEVSTATE_NODEVICE){
    DcamClose();
}
```

6.1.6 画像サイズ取得関数（DcamGetImageSize）

BOOL DcamGetImageSize (INT* pWidth, INT* pHeight)

【概要】

デバイスから取得する画像データ 1 フレームの幅と高さを取得します。

【引数】

pWidth　　画像の幅を格納する変数のアドレスを指定します。

pHeight　　画像の高さを格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

DcamSetBinning または DcamSetMeasureDataCount を実行したあとは、この関数を使って取得画像サイズを確認してください。

【参照】

DcamGetBitPerPixel, DcamGetCaptureBytes, DcamSetBinning, DcamSetMeasureDataCount

【例】

以下が関数の呼び出し例です。

```
INT         nWidth = 0;
INT         nHeight = 0;
DWORD       dwErrCode;

// Get size of image
if(DcamGetImageSize(&nWidth, &nHeight) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.7　ビット数取得関数（DcamGetBitPerPixel）

BOOL DcamGetBitPerPixel(INT* pBit)

【概要】

画素あたりのビット数を取得します。

【引数】

pBit	画素あたりのビット数を格納する変数のアドレスを指定します。 以下のいずれかになります。

DCAM_BITPIXEL_8	：８ビット
DCAM_BITPIXEL_10	：10 ビット
DCAM_BITPIXEL_12	：12 ビット
DCAM_BITPIXEL_14	：14 ビット
DCAM_BITPIXEL_16	：16 ビット

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetImageSize, DcamGetCaptureBytes

【例】

以下が関数の呼び出し例です。

```
INT         nBit   = 0;
DWORD   dwErrCode;

// Get size of image
if(DcamGetBitPerPixel(&nBit) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.8　センサタイプ設定関数 （DcamSetCCDType）

BOOL DcamSetCCDType(INT nType)

【概要】

センサのタイプを設定します。

【引数】

nType　　センサのタイプを指定します。
センサタイプは、DCamUSB.h に規定されたタイプを指定します。
例)
DCAM_CCD_TYPE0 ： 2068x1(S10420-1106),
2068x70(S10420-1106-01,
S11071-1106)
DCAM_CCD_TYPE2 ： 2068x22(S10420-1104-01,
S11071-1104)
DCAM_CCD_TYPE3 ： 1044x1(S10420-1006),
1044x70(S10420-1006-01,
S11071-1006)
DCAM_CCD_TYPE5 ： 1044x22(S10420-1004-01,
S11071-1004)
DCAM_CCD_TYPE10: 64x64(G11097)

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

【参照】

DcamGetCCDType, DcamGetImageSize, DcamGetCaptureBytes

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetCCDType(DCAM_CCD_TYPE0) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.9　センサタイプ取得関数 （DcamGetCCDType）

BOOL DcamGetCCDType(INT* pType)

【概要】

センサのタイプを取得します。

【引数】

pType　　センサのタイプを格納する変数のアドレスを指定します。
取得されるセンサタイプは、DCamUSB.h に規定されています。
例）
DCAM_CCD_TYPE0 ： 2068x1(S10420-1106),
2068x70(S10420-1106-01,
S11071-1106)
DCAM_CCD_TYPE2 ： 2068x22(S10420-1104-01,
S11071-1104)
DCAM_CCD_TYPE3 ： 1044x1(S10420-1006),
1044x70(S10420-1006-01,
S11071-1006)
DCAM_CCD_TYPE5 ： 1044x22(S10420-1004-01,
S11071-1004)
DCAM_CCD_TYPE10: 64x64(G11097)

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetCCDType

【例】

以下が関数の呼び出し例です。

```
INT         nType;
DWORD   dwErrCode;

if(DcamGetCCDType(&nType) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.10 計測データ数設定関数（DcamSetMeasureDataCount:）

BOOL DcamSetMeasureDataCount(INT nCount)

【概要】

取得する計測データのフレーム数を設定します。

【引数】

nCount	計測データのフレーム数を指定します。 指定範囲は、パラメータリスト （DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

【参照】

DcamGetBinning, DcamGetImageSize, DcamGetBitPerPixel,
DcamDcamGetCaptureBytes,
DcamGetMeasureDataCount

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetMeasureDataCount (10) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.11 計測データ数取得関数（DcamGetMeasureDataCount）

BOOL DcamGetMeasureDataCount(INT* pCount)

【概要】

計測データのフレーム数の取得を行います。

【引数】

pCount　　計測データのフレーム数を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

【参照】

DcamGetBinning, DcamGetImageSize, DcamGetBitPerPixel,
DcamGetCaptureBytes,
DcamSetMeasureDataCount

【例】

以下が関数の呼び出し例です。

```
INT         nCount = 0;
DWORD   dwErrCode;

// Get the measurement line count
if(DcamGetMeasureDataCount (&nCount) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.12 取得総バイト数取得関数（DcamGetCaptureBytes）

BOOL DcamGetCaptureBytes(INT* pBytes)

【概要】

1 フレームのバイト数を取得します。

【引数】

pBytes　　データ計測における1フレームの取得バイト数を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

【参照】

DcamGetImageSize, DcamGetBitPerPixel, DcamGetMeasureDataCount, DcamGetTotalCaptureBytes

【例】

以下が関数の呼び出し例です。

```
INT         nBytes = 0;
DWORD   dwErrCode;

// Get total bytes of capture size.
if(DcamGetFrameBytes(&nBytes) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.13 取得総バイト数取得関数（DcamGetTotalCaptureBytes）

BOOL DcamGetTotalCaptureBytes(INT* pBytes)

【概要】

データ計測 1 回の取得総バイト数を取得します。

【引数】

pBytes	データ計測1回の取得総バイト数を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

【参照】

DcamGetImageSize, DcamGetBitPerPixel, DcamGetMeasureDataCount, DcamGetCaptureBytes

【例】

以下が関数の呼び出し例です。

```
INT         nBytes = 0;
DWORD   dwErrCode;

// Get total bytes of capture size.
if(DcamGetFrameBytes(&nBytes) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.14 画像取得開始関数（DcamCapture）

BOOL DcamCapture(LPVOID pImageBuff, INT nBuffSize)

【概要】

デバイスから 1 画像の取得を開始します。

【引数】

pImageBuff　画像データを格納するバッファの先頭アドレスを指定します。

nBuffSize　画像データバッファのバイト数を指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

1. この関数は画像取得の開始命令を発行します。関数が終了しても画像取得は完了していません。画像取得完了は DcamWait で確認します。
2. 必要のバッファサイズは DcamGetCaptureBytes で取得できます。

【参照】

DcamWait, DcamStop, DcamStopEx, DcamGetCaptureBytes

【例】

「7.1 データ計測手順について」を参照してください。

6.1.15 画像取得開始関数［X 軸反転］(DcamCaptureReverseX)

BOOL DcamCaptureReverseX(LPVOID pImageBuff, INT nBuffSize)

【概要】

デバイスから 1 画像の取得を開始します。取得した画像データは X 軸反転されます。

【引数】

pImageBuff　画像データを格納するバッファの先頭アドレスを指定します。

nBuffSize　画像データバッファのバイト数を指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

1. この関数は画像取得の開始命令を発行します。関数が終了しても画像取得は完了していません。画像取得完了は DcamWait で確認します。
2. 必要のバッファサイズは DcamGetCaptureBytes で取得できます。

【参照】

DcamWait, DcamStop, DcamStopEx, DcamGetCaptureBytes

【例】

「7.1 データ計測手順について」を参照してください。

6.1.16 画像取得中止関数（DcamStop）

BOOL DcamStop(VOID)

【概要】
画像取得を中断します。

【引数】
なし

【戻り値】
正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】
なし

【参照】
DcamCapture, DcamCaptureReverseX, DcamWait, DcamStopEx

【例】
「7.1 データ計測手順について」を参照してください。

6.1.17 画像取得中止関数（DcamStopEx）

BOOL DcamStopEx(VOID)

【概要】

画像取得を中断します。
基本機能は、"DcamStop" と同じですが、下記の機種をご使用の場合は、
"DcamStop" ではなく、本関数をご使用ください。

機種	バージョン
C11287	“V1.01C_V1.01F” 以前
C11288	“V1.01C_V1.01F” 以前
C11860	“V0.02C_V0.03F” 以前
C11861	“V0.02C_V0.03F” 以前

【引数】

なし

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamCapture, DcamCaptureReverseX, DcamWait, DcamStop

【例】

「7.1 データ計測手順について」を参照してください。

6.1.18 画像取得完了待機関数（DcamWait）

BOOL DcamWait(DWORD* pStatus, INT nTimeout)

【概要】

画像取得の完了を待ちます。

【引数】

pStatus	画像取得完了ステータスを格納する変数のアドレスを指定します。 完了状態はこの変数内の値で判断します。値は以下のいずれかになります。 DCAM_WAITSTATUS_COMPLETED　：画像取得完了 DCAM_WAITSTATUS_UNCOMPLETED　：画像取得未完了
nTimeout	タイムアウトをミリ秒で指定します。 DCAM_WAIT_INFINITE を指定した場合は画像取得が完了するまで待ちます。 0 を指定した場合、ステータスを確認後すぐに制御を戻します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamCapture, DcamCaptureReverseX, DcamStop, DcamStopEx

【例】

「7.1 データ計測手順について」を参照してください。

6.1.19 ゲイン設定関数（DcamSetGain）

BOOL DcamSetGain(INT nGain)

【概要】

ゲインの設定を行います。

【引数】

nGain	ゲインを指定します。 指定範囲は、パラメータリスト （DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetGain

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetGain(5) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.20 ゲイン取得関数（DcamGetGain）

BOOL DcamGetGain(INT* pGain)

【概要】

ゲインの取得を行います。

【引数】

pGain　　ゲインを格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetGain

【例】

以下が関数の呼び出し例です。

```
INT         nGain = -1;
DWORD   dwErrCode;

if(DcamGetGain(&nGain) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.21 オフセット設定関数（DcamSetOffset）

BOOL DcamSetOffset(INT nOffset)

【概要】

オフセットの設定を行います。

【引数】

nOffset　　オフセットを指定します。
指定範囲は、パラメータリスト
（DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetOffset

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetOffset(10) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.22 オフセット取得関数（DcamGetOffset）

BOOL DcamGetOffset(INT* pOffset)

【概要】

オフセットの取得を行います。

【引数】

pOffset　　オフセットを格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetOffset

【例】

以下が関数の呼び出し例です。

```
INT         nOffset = -1;
DWORD   dwErrCode;

if(DcamGetOffset(&nOffset) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.23 ビニング設定関数（DcamSetBinning）

BOOL DcamSetBinning(INT nBinning)

【概要】

ビニングを設定します。

【引数】

nBinning　　ビニングを指定します。以下のいずれかになります。

DCAM_BINNING_AREA　：エリアスキャニング
DCAM_BINNING_FULL　：フルラインビニング

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

この関数を実行すると 1 回の計測データ取得サイズあたりのバイト数が変わることがあります。
DcamGetCaptureBytes などでサイズを確認してください。

【参照】

DcamGetBinning, DcamGetCaptureBytes

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetBinning(DCAM_BINNING_FULL) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.24 ビニング取得関数（DcamGetBinning）

BOOL DcamGetBinning(INT* pBinning)

【概要】

ビニングを取得します。

【引数】

pBinning　現在設定されているビニングを格納する変数のアドレスを指定します。
取得した値は以下のいずれかになります。

DCAM_BINNING_AREA　：エリアスキャニング
DCAM_BINNING_FULL　：フルラインビニング

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetBinning

【例】

以下が関数の呼び出し例です。

```
INT         nBinning;
DWORD   dwErrCode;

if(DcamGetBinning(&nBinning) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.25 トリガモード設定関数（DcamSetTriggerMode）

BOOL DcamSetTriggerMode(INT nMode)

【概要】

トリガモードを設定します。

【引数】

nMode　　トリガモードを指定します。以下のいずれかになります。

DCAM_TRIGMODE_INT	：内部同期モード（「INT」モード）
DCAM_TRIGMODE_EXT_EDGE	：外部同期モード1(「EXT.EDGE」モード)
DCAM_TRIGMODE_EXT_LEVEL	：外部同期モード2(「EXT.LEVEL」モード)
DCAM_TRIGMODE_GS_INT	：GS 内部同期モード（「INT」モード）
DCAM_TRIGMODE_GS_EXT_EDGE	：GS 外部同期モード 1（「EXT.EDGE1 」モード）
DCAM_TRIGMODE_GS_EXT_ONE_SHOT	：GS 外部同期モード 2（「EXT.EDGE2」モード）
DCAM_TRIGMODE_GS_EXT_GATED	：GS 外部同期ゲートモード（「EXT. GATED」モード）
DCAM_TRIGMODE_RS_INT	：RS 内部同期モード（「INT」モード）
DCAM_TRIGMODE_RS_EXT_EDGE	：RS 外部同期モード 1（「EXT.EDGE1」モード）
DCAM_TRIGMODE_RS_EXT_ONE_SHOT	：RS 外部同期モード 2（「EXT. EDGE2」モード）
DCAM_TRIGMODE_RS_EXT_GATED	：RS 外部同期ゲートモード（「EXT. GATED」モード）

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetTriggerMode, DcamSetTriggerPolarity, DcamGetTriggerPolarity, DcamSetExposureTime, DcamGetExposureTime

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetTriggerMode(DCAM_TRIGMODE_INT) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.26 トリガモード取得関数（DcamGetTriggerMode）

BOOL DcamGetTriggerMode(INT* pMode)

【概要】

トリガモードを取得します。

【引数】

pMode　現在設定されているトリガモードを格納する変数のアドレスを指定します。
取得した値は以下のいずれかになります。

DCAM_TRIGMODE_INT	：内部同期モード（「INT」モード）
DCAM_TRIGMODE_EXT_EDGE	：外部同期モード1(「EXT.EDGE」モード)
DCAM_TRIGMODE_EXT_LEVEL	：外部同期モード2(「EXT.LEVEL」モード)
DCAM_TRIGMODE_GS_INT	：GS 内部同期モード（「INT」モード）
DCAM_TRIGMODE_GS_EXT_EDGE	：GS 外部同期モード 1（「EXT.EDGE1 」モード）
DCAM_TRIGMODE_GS_EXT_ONE_SHOT	：GS 外部同期モード 2（「EXT.EDGE2」モード）
DCAM_TRIGMODE_GS_EXT_GATED	：GS 外部同期ゲートモード（「EXT. GATED」モード）
DCAM_TRIGMODE_RS_INT	：RS 内部同期モード（「INT」モード）
DCAM_TRIGMODE_RS_EXT_EDGE	：RS 外部同期モード 1（「EXT.EDGE1」モード）
DCAM_TRIGMODE_RS_EXT_ONE_SHOT	：RS 外部同期モード 2（「EXT. EDGE2」モード）
DCAM_TRIGMODE_RS_EXT_GATED	：RS 外部同期ゲートモード（「EXT. GATED」モード）

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetTriggerMode, DcamSetTriggerPolarity, DcamGetTriggerPolarity, DcamSetExposureTime, DcamGetExposureTime

【例】

以下が関数の呼び出し例です。

```
INT         nMode;
DWORD   dwErrCode;
 if(DcamGetTriggerMode(&nMode) != TRUE){
     dwErrCode = DcamGetLastError();
}
```

6.1.27 トリガ極性設定関数（DcamSetTriggerPolarity）

BOOL DcamSetTriggerPolarity(INT nPolarity)

【概要】

トリガ極性を設定します。

【引数】

nPolarity　　トリガ極性を指定します。以下のいずれかになります。

DCAM_TRIGPOL_POSITIVE　：正
DCAM_TRIGPOL_NEGATIVE　：負

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetTriggerPolarity, DcamSetTriggerMode, DcamGetTriggerMode, DcamSetExposureTime, DcamGetExposureTime

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetTriggerPolarity(DCAM_TRIGPOL_POSITIVE) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.28 トリガ極性取得関数（DcamGetTriggerPolarity）

BOOL DcamGetTriggerPolarity(INT* pPolarity)

【概要】

トリガ極性を取得します。

【引数】

pPolarity　現在設定されているトリガ極性格納する変数のアドレスを指定します。
取得した値は以下のいずれかになります。

DCAM_TRIGPOL_POSITIVE　：正
DCAM_TRIGPOL_NEGATIVE　：負

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetTriggerPolarity, DcamSetTriggerMode, DcamGetTriggerMode, DcamSetExposureTime, DcamGetExposureTime

【例】

以下が関数の呼び出し例です。

```
INT         nPolarity;
DWORD   dwErrCode;

if(DcamGetTriggerPolarity(&nPolarity) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.29 露光時間設定関数（DcamSetExposureTime）

BOOL DcamSetExposureTime(INT nTime)

【概要】

露光時間を設定します。

【引数】

nTime　　露光時間を基準時間単位で指定します。
指定範囲は、パラメータリスト
（DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetExposureTime, DcamSetTriggerMode, DcamGetTriggerMode,
DcamSetTriggerPolarity, DcamGetTriggerPolarity

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetExposureTime(120 /* 120 msec */) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.30 露光時間取得関数（DcamGetExposureTime）

BOOL DcamGetExposureTime(INT* pTime)

【概要】

露光時間を取得します。

【引数】

pTime　　現在設定されている露光時間を取得します。単位は基準時間単位です。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetExposureTime, DcamSetTriggerMode, DcamGetTriggerMode, DcamSetTriggerPolarity, DcamGetTriggerPolarity

【例】

以下が関数の呼び出し例です。

```
INT         nTime;// msec
DWORD   dwErrCode;

if(DcamGetExposureTime(&nTime) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.31 CCD 動作モード設定関数（DcamSetOperatingMode）

BOOL DcamSetOperatingMode(INT nMode)

【概要】

CCD の動作モードを設定します。

【引数】

nMode　　CCD の動作モードを指定します。以下のいずれかになります。

DCAM_OPMODE_DARKCURRENT　：低暗電流動作モード
DCAM_OPMODE_SATURATION　　：大飽和動作モード

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetOperatingMode

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetOperatingMode(DCAM_OPMODE_DARKCURRENT) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.32 CCD 動作モード取得関数（DcamGetOperatingMode）

BOOL DcamGetOperatingMode(INT* pMode)

【概要】

CCD の動作モードを取得します。

【引数】

pMode	現在設定されている CCD の動作モードを格納する変数のアドレスを指定します。 取得した値は以下のいずれかになります。 DCAM_OPMODE_DARKCURRENT ：低暗電流動作モード DCAM_OPMODE_SATURATION ：大飽和動作モード

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetOperatingMode

【例】

以下が関数の呼び出し例です。

```
INT         nMode;
DWORD       dwErrCode;

if(DcamGetOperatingMode(&nMode) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.33 LED 制御モード設定関数 （DcamSetLEDOperatingMode）

BOOL DcamSetLEDOperatingMode(INT nMode)

【概要】

LED の発光制御モードを設定します。

【引数】

nMode　　LED の発光制御モードです。以下のいずれかになります。

DCAM_LEDOPMODE_OFF : LED を発光させません（常にオフ）
DCAM_LEDOPMODE_ON : LED を発光させます。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetLEDOperatingMode

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamSetLEDOperatingMode(DCAM_LEDOPMODE_OFF) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.34 LED 制御モード取得関数 （DcamGetLEDOperatingMode）

BOOL DcamGetLEDOperatingMode(INT* pMode)

【概要】

LED の発光制御モードを取得します。

【引数】

pMode　　LED の発光制御モードを格納する変数のアドレスを指定します。
LED の発光制御モードは、以下のいずれかになります。

DCAM_LEDOPMODE_OFF : LED を発光させません（常にオフ）
DCAM_LEDOPMODE_ON : LED を発光させます。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetLEDOperatingMode

【例】

以下が関数の呼び出し例です。

```
INT         nMode;
DWORD       dwErrCode;

if(DcamGetLEDOperatingMode(&nMode) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.35 基準時間単位設定関数 (DcamSetStandardTimeUnit)

BOOL DcamSetStandardTimeUnit(INT nType)

【概要】

基準時間単位の設定をします。

【引数】

nType　基準時間単位のタイプです。以下のいずれかになります。
DCAM_TIME_UNIT_TYPE1：
トリガ時間単位 [msec]、パルス出力時間単位 [msec]
DCAM_TIME_UNIT_TYPE2：
トリガ時間単位 [usec]、パルス出力時間単位 [usec]
DCAM_TIME_UNIT_TYPE3：
トリガ時間単位 [msec]、パルス出力時間単位 [usec]
DCAM_TIME_UNIT_TYPE4：
トリガ [Clock]、パルス出力 [Clock]

【戻り値】

正常に終了した場合は TRUE(1)、それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetStandardTimeUnit、DcamSetExposureTime、
DcamGetExposureTime、DcamSetOutPulse、DcamGetOutPulse

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;

if(DcamSetStandardTimeUnit(DCAM_TIME_UNIT_TYPE1) !=TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.36 基準時間単位取得関数 (DcamGetStandardTimeUnit)

BOOL DcamGetStandardTimeUnit(INT *pType)

【概要】

基準時間単位の取得をします。

【引数】

pType　基準時間単位のタイプを格納する変数のアドレスを指定します。
基準時間単位のタイプは、以下のいずれかになります。
DCAM_TIME_UNIT_TYPE1：
トリガ時間単位 [msec]、パルス出力時間単位 [msec]
DCAM_TIME_UNIT_TYPE2：
トリガ時間単位 [usec]、パルス出力時間単位 [usec]
DCAM_TIME_UNIT_TYPE3：
トリガ時間単位 [msec]、パルス出力時間単位 [usec]
DCAM_TIME_UNIT_TYPE4：
トリガ [Clock]、パルス出力 [Clock]

【戻り値】

正常に終了した場合は TRUE(1)、それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetStandardTimeUnit、DcamSetExposureTime、
DcamGetExposureTime、DcamSetOutPulse、DcamGetOutPulse

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT   nType;

if(DcamGetStandardTimeUnit(&nType) !=TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.37 パルス出力条件設定関数 （DcamSetOutPulse）

BOOL DcamSetOutPulse(INT nMode, INT nPolarity, INT nDelayTime, INT nPulseWidth)

【概要】

パルス信号出力条件を設定します。

【引数】

nMode　パルス信号出力モードを設定します。以下のいずれかになります。

DCAM_OUTMODE_NOTOUTPUT	：パルス出力 OFF
DCAM_OUTMODE_PLS_DT_PW	：パルス出力 ON（ディレイ時間 + パルス幅）
DCAM_OUTMODE_PLS_ACCUM	：パルス出力 ON（蓄積時間より計算）

nPolarity　極性を設定します。以下のいずれかになります。

DCAM_OUTPOL_POSITIVE	：正
DCAM_OUTPOL_NEGATIVE	：負

nDelayTime　パルス出力ディレイ時間を基準時間単位で指定します。
指定範囲は、パラメータリスト（DCamUSB_FunctionParameterList_J.pdf）を参照ください。

nPulseWidth　パルス幅を基準時間単位で指定します。
指定範囲は、パラメータリスト（DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetOperatingMode

【例】

以下が関数の呼び出し例です。

```
DWORD   dwErrCode;

if(DcamSetOutPulse(DCAM_OUTMODE_PLS_DT_PW,DCAM_OUTPOL_NEGATIVE,10,100) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.38 パルス出力条件取得関数 （DcamGetOutPulse）

BOOL DcamGetOutPulse (INT* pMode, INT* pPolarity, INT* pDelayTime, INT* pPulseWidth)

【概要】

パルス信号出力条件を取得します。

【引数】

pMode　パルス信号出力モードを格納する変数のアドレスを指定します。
パルス信号出力モードは、以下のいずれかになります。

DCAM_OUTMODE_NOTOUTPUT	: パルス出力 OFF
DCAM_OUTMODE_PLS_DT_PW	: パルス出力 ON（ディレイ時間 + パルス幅）
DCAM_OUTMODE_PLS_ACCUM	: パルス出力 ON（蓄積時間より計算）

pPolarity　極性を格納する変数のアドレスを指定します。
極性は、以下のいずれかになります。

DCAM_OUTPOL_POSITIVE	: 正
DCAM_OUTPOL_NEGATIVE	: 負

pDelayTime　パルス出力ディレイ時間を格納する変数のアドレスを指定します。
パルス出力ディレイ時間は基準時間単位で、範囲は 0 ～ 255 です。

pPulseWidth　パルス幅を格納する変数のアドレスを指定します。
パルス幅は基準時間単位で、範囲は 0 ～ 1023 です。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetOutPulse

【例】

以下が関数の呼び出し例です。

```
INT         nMode;
INT         nPolarity;
INT         nDelayTime = 0;
INT         nPulseWidth = 0;
DWORD       dwErrCode;
if(DcamGetOutPulse(&nMode, &nPolarity, &nDelayTime, &nPulseWidth)
!= TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.39 デバイスパラメータ読込関数（DcamLoadParameters）

BOOL DcamLoadParamters(INT nTimeout)

【概要】

内蔵のEEPROMからデバイスの設定を読み込み、最後の動作状態に復帰させます。

【引数】

nTimeout　　タイムアウトをミリ秒で指定します。

【戻り値】

正常に終了した場合はTRUE (1), それ以外はFALSE (0) です。
詳細エラー情報は、DcamGetLastError関数で取得することができます。

【備考】

なし

【参照】

DcamStoreParameters

【例】

以下が関数の呼び出し例です。

```
DWORD   dwErrCode;

if(DcamLoadParamters() != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.40 デバイスパラメータ保存関数（DcamStoreParameters）

BOOL DcamStoreParamters(VOID)

【概要】

デバイスの現設定を内蔵の EEPROM に書き込みます。

【引数】

なし

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamLoadParameters

【例】

以下が関数の呼び出し例です。

```
DWORD    dwErrCode;

if(DcamStoreParamters() != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.41 バージョン情報取得関数（DcamGetVersion）

BOOL DcamGetVersion(char* szVersion, INT nBufSize)

【概要】

ライブラリのバージョンを文字列で取得します。

【引数】

szVersion	ライブラリのバージョンを格納する文字列バッファの先頭アドレスを指定します。
nBufSize	バッファのサイズをバイト数で指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamFirmwareVersion, DcamGetDriverVersion,
DcamGetDeviceInformation

【例】

以下が関数の呼び出し例です。

```
char        szVersion[256];
DWORD   dwErrCode;

if(DcamGetVersion(szVersion, sizeof(szVersin)) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.42 ドライバ情報取得関数（DcamGetDriverVersion）

BOOL DcamGetDriverVersion(char* szVersion, INT nBufSize)

【概要】

ドライバのバージョン番号を文字列で取得します。

【引数】

szVersion	ドライバのバージョンを格納する文字列バッファの先頭アドレスを指定します。
nBufSize	バッファのサイズをバイト数で指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamVersion, DcamGetFirmwareVersion, DcamGetDeviceInformation

【例】

以下が関数の呼び出し例です。

```
char        szVersion[256];
DWORD    dwErrCode;

if(DcamGetDriverVersion(szVersion, sizeof(szVersin)) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.43 ファームウェア情報取得関数（DcamGetFirmwareVersion）

BOOL DcamGetFirmwareVersion(char* szFirmVersion, INT nBufSize)

【概要】

ファームウェアのバージョン番号を文字列で取得します。

【引数】

szFirmVersion	ファームウェアのバージョンを格納する文字列バッファの先頭アドレスを指定します。
nBufSize	バッファのサイズをバイト数で指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamVersion, DcamGetDriverVersion, DcamGetDeviceInformation

【例】

以下が関数の呼び出し例です。

```
char        szVersion[256];
DWORD    dwErrCode;

if(DcamGetFirmwareVersion(szVersion, sizeof(szVersin)) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.44 デバイス情報取得関数（DcamGetDeviceInformation）

BOOL DcamGetDeviceInformation(INT nType, char* pszBuff, INT nBufSize)

【概要】

デバイスの情報を取得します。

【引数】

nType　取得する情報を指定します。以下のいずれかになります。

DCAM_DEVINF_TYPE	：型番
DCAM_DEVINF_SERIALNO	：シリアル番号
DCAM_DEVINF_VERSION	：バージョン

pszBuff　情報を格納するバッファの先頭アドレスを指定します。

nBufSize　バッファのサイズをバイト数で指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamVersion, DcamGetDriverVersion, DcamGetFirmwareVersion,

【例】

以下が関数の呼び出し例です。

```
char        szInfo[256];
DWORD    dwErrCode;

if(DcamGetDeviceInformation(DCAM_DEVINF_TYPE, szInfo,
sizeof(szInfo)) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.45 USB 転送速度タイプ取得関数（DcamGetTransferRateType）

BOOL DcamGetTransferRateType (INT* pType)

【概要】

接続している USB 転送速度タイプを取得します。

【引数】

pType	現在接続されている USB 転送速度タイプを格納する変数のアドレスを指定します。 取得した値は以下のいずれかになります。 DCAM_TRANSRATE_USB11 : USB 1.1 規格 DCAM_TRANSRATE_USB20 : USB 2.0 規格

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

なし

【例】

以下が関数の呼び出し例です。

```
INT         nType;
DWORD   dwErrCode;

if(DcamGetTransferRateType(&nType) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.46 最新エラーコード取得関数（DcamGetLastError ）

DWORD　DcamGetLastError(VOID)

【概要】

最新のエラーコードを取得します。

【引数】

なし

【戻り値】

最新のエラーコードです。エラーコードについては、「 5.2. エラーコード表（実行ステータス)」をご覧ください。

【備考】

なし

6.1.47 オーバークロック設定関数（DcamSetOverClock）

BOOL DcamSetOverClock (INT nClock)

【概要】

オーバークロックを設定します。

【引数】

nClock　　オーバークロックのクロック数を指定します。
指定範囲は、パラメータリスト
(DCamUSB_FunctionParameterList_J.pdf) を参照ください。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamCtrlGetLastError 関数で取得することができます。

【備考】

トリガモードが RS 内部同期モード、RS 外部同期モード 1、RS 外部同期モード 2、RS 外部同期ゲートモード時のみ実行できます。

【参照】

DcamGetOverClock , DcamSetTriggerMode, DcamGetTriggerMode

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamSetOverClock (10) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.1.48 オーバークロック取得関数（DcamGetOverClock）

BOOL DcamGetOverClock (INT *pClock)

【概要】

オーバークロックを取得します。

【引数】

pClock　　オーバークロックのクロック数を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetOverClock

【例】

以下が関数の呼び出し例です。

```
INT nClock;
DWORD dwErrCode;
if(DcamGetOverClock (&nClock) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.1.49 MPP モード設定関数（DcamSetMPPMode）

BOOL DcamSetMPPMode(INT nMode)

【概要】

MPP モードを設定します。

【引数】

nMode	MPP モードを指定してください。以下の何れかになります。 DCAM_CCDMPPMODE_OFF ：MPP モード OFF DCAM_CCDMPPMODE_ON ：MPP モード ON

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

C11165-01 使用時、本関数を使用し、MPP モードを変更すると、ハードウェア内部で　蓄積時間および LineTime が再計算されますので、本関数呼出し後は、蓄積時間と LineTime を取得してください。

【参照】

DcamGetMPPMode

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
If( DcamSetMPPMode(DCAMM_CCDMPMODE_ON) != TRUE ){
    dwErrCode = DcamGetLastError();
}
```

6.1.50 MPP モード取得関数（DcamGetMPPMode）

BOOL DcamGetMPPMode(INT* pMode)

【概要】

MPP モードを取得します。

【引数】

pMode　　MPP モードを格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetMPPMode

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nMode ;
If( DcamGetMPPMode( &nMode )!=TRUR){
    dwErrCode = DcamGetLastError();
}
```

6.1.51 Line Time 設定関数（DcamSetLineTime）

BOOL DcamSetLineTime(INT nTime)

【概要】

Line Time を設定します。

【引数】

nTime	LineTime を設定します。 指定範囲は、パラメータリスト （DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetLineTime

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
If( DcamSetLineTime( nTime )!=TRUR){
    dwErrCode = DcamGetLastError();
}
```

6.1.52 Line Time 取得関数（DcamGetLineTime）

BOOL DcamGetLineTime(INT* pTime)

【概要】

Line Time を取得します。

【引数】

pTime　　LineTime を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetLineTime

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nTime
If( DcamGetLineTime( &nTime )!=TRUR){
    dwErrCode = DcamGetLastError();
}
```

6.1.53 積分容量設定関数（DcamSetIntegralCapacity）

BOOL DcamSetIntegralCapacity (INT nType)

【概要】

積分容量を示すタイプを設定します。

【引数】

nType	設定する積分容量を示すタイプを設定します。 指定範囲は、パラメータリスト （DCamUSB_FunctionParameterList_J.pdf）を参照ください。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetIntegralCapacit

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
If(DcamSetIntegralCapacity ( 0 )!=TRUR){
    dwErrCode = DcamGetLastError();
}
```

6.1.54 積分容量取得関数（DcamGetIntegralCapacity）

BOOL DcamGetIntegralCapacity(INT* pType)

【概要】

積分容量を示すタイプを取得します。

【引数】

pType　　積分容量を示すタイプを格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetIntegralCapacity

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nType;
If(DcamGetIntegralCapacity (&nType )!=TRUR){
    dwErrCode = DcamGetLastError();
}
```

6.1.55 CCD 駆動モードの設定関数 (DcamSetDriveMode)

BOOL DcamSetDriveMode(INT nMode, INT nTimeout)

【概要】

CCD 駆動モードを設定します。

【引数】

nMode	CCD 駆動モードを指定します。 以下のいずれかになります。 DCAM_CCDDRVMODE_SUSPEND ：Suspend DCAM_CCDDRVMODE_STANDBY ：Standby
nTimeout	CCD 駆動モード設定のタイムアウト時間 (ms) を指定します。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetDriveMode

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamSetDriveMode(DCAM_CCDDRVMODE_SUSPEND,0) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.56 CCD 駆動モードの取得関数 (DcamGetDriveMode)

BOOL DcamGetDriveMode(INT* pMode)

【概要】

CCD 駆動モードを取得します。

【引数】

pMode	CCD 駆動モードを格納する変数のアドレスを指定します。 取得した値は、以下のいずれかになります。 DCAM_CCDDRVMODE_SUSPEND ：Suspend DCAM_CCDDRVMODE_STANDBY ：Standby

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetDriveMode

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nMode ;
if(DcamGetDriveMode(&nMode) != TRUE){
    dwErrCode = DcamGetLastError();
}
```

6.1.57 Erectronic シャッターモード設定関数 (DcamSetElectronicShutter)

BOOL DcamSetElectronicShutter (INT nMode)

【概要】

Erectronic シャッターモードを設定します。

【引数】

nMode	Erectronic シャッターモードを指定してください。 以下の何れかになります。 DCAM_CCDESHUTTER_OFF : Erectronic シャッターモードOFF DCAM_CCDESHUTTER_ON : Erectronic シャッターモード ON

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamGetElectronicShutter

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
If(DcamSetElectronicShutter (DCAM_CCDESHUTTER_ON) != TRUE ){
    dwErrCode = DcamGetLastError();
}
```

6.1.58 Erectronic シャッターモード取得関数 (DcamGetElectronicShutter)

BOOL DcamGetElectronicShutter (INT* pMode)

【概要】

Erectronic シャッターモードを取得します。

【引数】

pMode	Erectronic シャッターモードを格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamSetElectronicShutter

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nMode ;
If(DcamGetElectronicShutter ( &nMode )!=TRUR){
    dwErrCode = DcamGetLastError();
}
```

6.1.59 TG パルス幅設定関数 (DcamSetSensorSignalPulseWidth)

BOOL DcamSetSensorSignalPulseWidth(INT nSignalSensor, INT nWidth) ;

【概要】

TG のパルス幅を設定します。
この値が蓄積時間の下限値になります。蓄積時間よりこの値が大きくなった場合は、必ず蓄積時間を変更してください。

【引数】

nSignalSecsor	0 固定です。
nWidth	TG パルス幅を設定します。 範囲は 2 ～ 500[usec] です。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

DcamGetSensorSignalPulseWidth

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
If( DcamSetSensorSignalPulseWidth( 0, 40 ){
        dwErrCode = DcamGetLastError();
}
```

6.1.60 TG パルス幅設定関数 (DcamGetSensorSignalPulseWidth)

BOOL DcamGetSensorSignalPulseWidth(INT nSignalSensor, INT* pWidth) ;

【概要】

TG のパルス幅を取得します。

【引数】

nSignalSecsor　0 固定です。
pWidth　TG パルス幅を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE(1), それ以外は FALSE(0) です。
詳細エラー情報は、DcamGetLastError 関数で取得することができます。

【備考】

DcamSetSensorSignalPulseWidth

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT     nWidth
If( DcamGetSensorSignalPulseWidth( 0, &nWidth ){
        dwErrCode = DcamGetLastError();
}
```

6.2 DcamTmpCtrl

6.2.1 初期化関数（DcamTmpCtrlInitialize）

BOOL DcamTmpCtrlInitialize (VOID)

【概要】

本ライブラリを初期化します。
この関数は C11512 専用です。

【引数】

なし

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

この関数は必ず最初に実行します。
既に初期化されている場合はエラーになります。
OS 内で 1 つのプロセスのみ使用することができます。

【参照】

DcamTmpCtrlUninitialize

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlInitialize () != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3 DcamTmpCtrl

6.3.1 初期化関数（DcamTmpCtrlInitialize）

BOOL DcamTmpCtrlInitialize (VOID)

【概要】
本ライブラリを初期化します。

【引数】
なし

【戻り値】
正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】
この関数は必ず最初に実行します。
既に初期化されている場合はエラーになります。
OS 内で 1 つのプロセスのみ使用することができます。

【参照】
DcamTmpCtrlUninitialize

【例】
以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlInitialize () != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.2　終了処理関数（DcamTmpCtrlUninitialize）

BOOL DcamTmpCtrlUninitialize (VOID)

【概要】

本ライブラリのリソースの開放、およびデバイスドライバのクローズをします。

【引数】

なし

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

プログラムの終了時、または本ライブラリが不要になったときに呼び出してください。

【参照】

DcamTmpCtrlInitialize

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlUninitialize () != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.3 最新エラーコード取得関数（DcamTmpCtrlGetLastError ）

DWORD DcamTmpCtrlGetLastError (VOID)

【概要】

最新のエラーコードを取得します。

【引数】

なし

【戻り値】

最新のエラーコードです。エラーコードについては、「 5.2. エラーコード表（実行ステータス)」をご覧ください。

【備考】

なし

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
dwErrCode = DcamTmpCtrlGetLastError ();
```

6.3.4 冷却制御状態設定関数（DcamTmpCtrlSetCoolingControl）

BOOL DcamTmpCtrlSetCoolingControl (BOOL bOnOff)

【概要】

冷却制御の状態を設定します。

【引数】

bOnOff	冷却制御状態を設定します。以下のいずれかになります。 DCAM_COOLING_CONTROL_OFF : 冷却装置 OFF DCAM_COOLING_CONTROL_ON ：冷却装置 ON

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamTmpCtrlGetCoolingControl

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlSetCoolingControl (DCAM_COOLING_CONTROL_ON) !=
TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.5 冷却制御状態取得関数（DcamTmpCtrlGetCoolingControl）

BOOL DcamTmpCtrlGetCoolingControl (BOOL *pbOnOff)

【概要】

冷却制御の状態を取得します。

【引数】

pbOnOff　　冷却制御状態を格納する変数のアドレスを指定します。
以下のいずれかになります。
DCAM_COOLING_CONTROL_OFF ：冷却装置 OFF
DCAM_COOLING_CONTROL_ON ：冷却装置 ON

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamTmpCtrlSetCoolingControl

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
BOOL bOnOff;
if(DcamTmpCtrlGetCoolingControl (&bOnOff) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.6 冷却温度 (設定値) 読み込み関数（DcamTmpCtrlLoadCoolingTemperature）

BOOL　DcamTmpCtrlLoadCoolingTemperature ()

【概要】

冷却温度 (設定値) を読み込みます。

【引数】

なし。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamTmpCtrlSaveCoolingTemperature

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlLoadCoolingTemperature () != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.7　冷却温度 (設定値) 保存関数（DcamTmpCtrlSaveCoolingTemperature）

BOOL　DcamTmpCtrlSaveCoolingTemperature ()

【概要】

冷却温度 (設定値) を保存します。

【引数】

なし。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamTmpCtrlLoadCoolingTemperature

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlSaveCoolingTemperature () != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.8 冷却温度 (設定値) 取得関数（DcamTmpCtrlGetCoolingTemperature）

BOOL DcamTmpCtrlGetCoolingTemperature (INT *pValue)

【概要】

冷却温度 (設定値) を取得します。

【引数】

pValue 冷却温度 (設定値) を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamTmpCtrlSetCoolingTemperature

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nValue;
if(DcamTmpCtrlGetCoolingTemperature (&nValue) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.9 冷却温度 (設定値) 設定関数（DcamTmpCtrlSetCoolingTemperature）

BOOL DcamTmpCtrlSetCoolingTemperature (INT nValue)

【概要】

冷却温度 (設定値) を設定します。

【引数】

nValue 冷却温度 (設定値) を指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

DcamTmpCtrlGetCoolingTemperature

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
if(DcamTmpCtrlSetCoolingTemperature (10) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.10 冷却温度 (現在値) 取得関数 (DcamTmpCtrlGetCoolingTemperatureCurrent)

BOOL　DcamTmpCtrlGetCoolingTemperatureCurrent (INT *pValue)

【概要】

冷却温度 (現在値) を取得します。

【引数】

pValue　　冷却温度 (現在値) を格納する変数のアドレスを指定します。

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

なし

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nValue;
if(DcamTmpCtrlGetCoolingTemperatureCurrent (&nValue) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.11 冷却状態取得関数（DcamTmpCtrlGetCoolingStatus）

BOOL　DcamTmpCtrlGetCoolingStatus (INT *pValue)

【概要】

冷却状態を取得します。

【引数】

pValue	冷却状態を格納する変数のアドレスを指定します。 以下のいずれかになります。 DCAM_COOLING_STATUS_NORMAL ：冷却温度が許容範囲内である DCAM_COOLING_STATUS_LOWER ：冷却温度が許容範囲より低い DCAM_COOLING_STATUS_HIGHER ：冷却温度が許容範囲より高い

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

なし

【例】

以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nValue;
if(DcamTmpCtrlGetCoolingStatus (&nValue) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

6.3.12 サーミスター状態取得関数（DcamTmpCtrlGetThermistorStatus）

BOOL　DcamTmpCtrlGetThermistorStatus (INT *pValue)

【概要】

サーミスターの状態を取得します。

【引数】

pValue	サーミスターの状態を格納する変数のアドレスを指定します。 以下のいずれかになります。 DCAM_THERMISTOR_STATUS_NOERROR　：サーミスター正常 DCAM_THERMISTOR_STATUS_ERROR　：サーミスター異常 DCAM_THERMISTOR_STATUS_OVER　：サーミスター温度異常

【戻り値】

正常に終了した場合は TRUE (1), それ以外は FALSE (0) です。
詳細エラー情報は、DcamTmpCtrlGetLastError 関数で取得することができます。

【備考】

なし

【参照】

なし

【例】

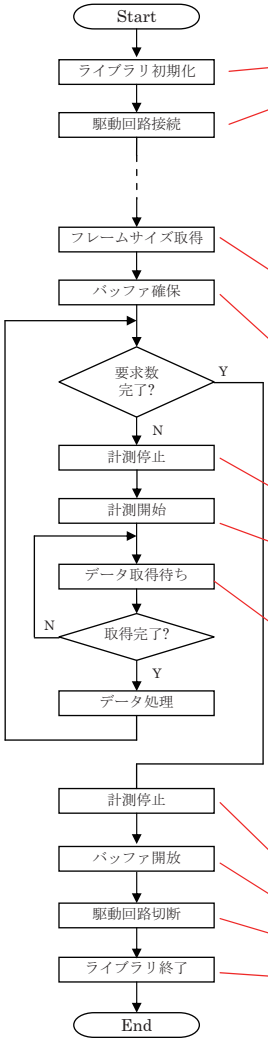
以下が関数の呼び出し例です。

```
DWORD dwErrCode;
INT nValue;
if(DcamTmpCtrlGetThermistorStatus (&nValue) != TRUE){
    dwErrCode = DcamTmpCtrlGetLastError ();
}
```

7 補足説明

7.1 データ計測手順について

DCamUSB.dll を使用し、データ計測を行う手順を記載します。




```
DcamInitialize();// Initialize library(DLL).
DcamOpen();// Open camera
DcamStop();// Stop Acquisition

// Set camera parameters
DcamSetLEDOperatingMode(DCAM_LEDOPMODE_ON); // LED operation
DcamSetStandardTimeUnit(DCAM_TIME_UNIT_TYPE1); // Standard time unit
DcamSetBinning(DCAM_BINNING_FULL);   // Full line binning
DcamSetTriggerMode(DCAM_TRIGMODE_INT); // Trigger mode to INTERNAL
DcamSetTriggerPolarity(DCAM_TRIGPOL_NEGATIVE); // Trigger polarity
DcamSetExposureTime(100);
DcamSetCCDType(DCAM_CCD_TYPE0); // Sensor type to "2068x70"
DcamSetGain(1); // Gain 1
DcamSetOffset(10); // Offset 10
DcamSetMeasureDataCount(nFrameCount); // Acquisition frame count

// Get camera parameters
DcamGetBitPerPixel(&nBitSize);   // Bit per pixel
DcamGetImageSize(&nWidth, &nHeight); // Image size
char szVersion[256] = {0};
DcamGetVersion(szVersion, 256); // DLL Version number string

// Calculation Image size and allocate buffer
nImageSize = nWidth * nHeight;
pDataBuff = new WORD[ nImageSize * nFrameCount];

// Acquisition execute
if( pDataBuff != NULL )
{
  // 256 frames capture sample
  for(int nDataIdx=0; nDataIdx<256; nDataIdx++)
  {
    // Stop capture (clear camera buffer)
    DcamStop();
    // Start image acquisition
    if (DcamCapture(pDataBuff,   nImageSize * nFrameCount * 2) == FALSE)
    {
      // Fail the start acquisition.
      break;
    }
    // Wait to complete the capture image
    DWORD dwRetStatus = DCAM_WAITSTATUS_COMPLETED;
    while( TRUE )
    {
      DcamWait(&dwRetStatus, 0);
      if( dwRetStatus == DCAM_WAITSTATUS_COMPLETED )
      {
        // Data process
        //      ・
        //      ・
        //      ・

        break;
      }
    }
  }
}

// Process for exit.
DcamStop(); // Stop acquisition
delete [] pDataBuff; // Release acquisition buffer
DcamClose(); // Close the device
DcamUninitialize(); // Uninitialize DLL
```

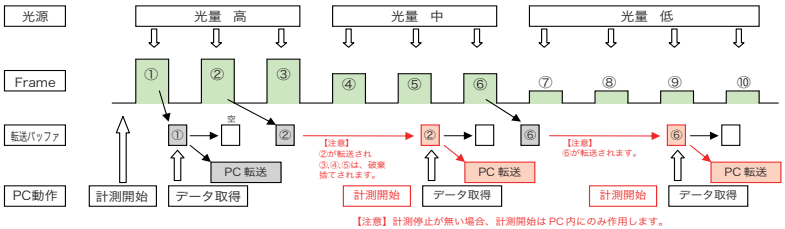
7.2 "DcamStop()" の使用について

"DcamStop()" は、PC ソフトウェアの計測と駆動回路の計測動作を停止させ、転送バッファをクリアする機能を持ちます。
従って、PC ソフトウェア起動時には、駆動回路が計測動作にある際の停止操作を行い、また、PC ソフトウェア終了時には、やはり計測動作を停止させ終了する為に使用することが可能です。

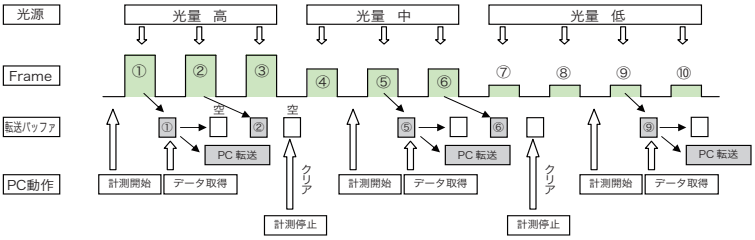
サンプルプログラムでは、更に計測動作中に
【計測停止：DcamStop()】→【計測開始：DcamCapture()】→
【取得待ち：DcamWait()】

の組み合わせで使用していますが、ここでの "DcamStop()" は使用しなくても計測動作は可能です。
ただし、"DcamStop()" を使用した場合と使用しない場合で、PC が取り込めるデータ (フレーム) が　異なります。
これは、駆動回路と PC 間のデータ転送方式によるものです。ご注意ください。

■ 繰り返し計測で "DcamStop()" を使用しない場合



■ 繰り返し計測で "DcamStop()" を使用する場合



7.3 デバイス接続及び取り外しについて

DCamUSB ライブラリは、デバイスの接続状態を確認する関数（DcamGetDeviceState）があります。
ライブラリを使用してアプリケーションがデバイスへ接続している時にデバイスやUSB ケーブル等が外された場合、アプリケーションがデバイスの接続状態を確認するためです。

但し、ライブラリ自身はデバイスが外されたことを自動では認識はしません。
デバイスが外されたことを認識するのは、アプリケーションが行ないます。なぜなら、デバイスが外されたとき、OS はデバイスが外されたことを認識してアプリケーションのトップウィンドウに、デバイスに変化があったことを通知するメッセージ（WM_DEVICECHANGE）を送信するからです。

アプリケーションは、メッセージを受信した時、DcamGetDeviceState 関数でデバイスの接続状態を確認して下さい。関数から取得したステータスタイプによって、デバイスが存在するか確認ができます。

デバイスに接続している時にデバイスの接続状態を確認した結果が “ デバイスなし ”(DCAM_DEVSTATE_NODEVICE) の場合は、デバイスが外されているので、ライブラリで切断処理 (DcamClose, DcamUninitialize) を行って下さい。
また、デバイスが接続された場合も、OS はデバイスの変化を認識し、同じメッセージをアプリケーションへ通知するので、デバイスの接続状態を確認して、デバイスが見つかれば接続が可能です。

上記の操作において、いくつか注意することがあります。

1) デバイスを外したり、接続したりする操作は、頻繁に行うとデバイスによっては故障の原因になるかもしれません。使用するデバイスの仕様をよく理解した上で利用してください。

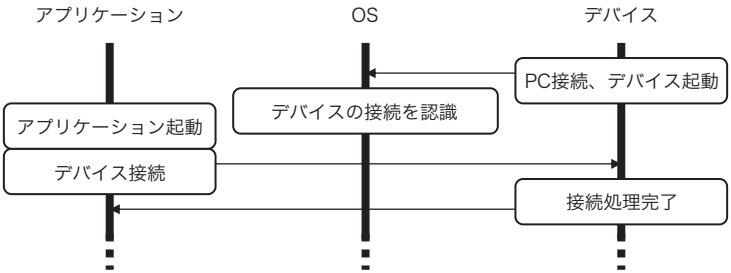
2) デバイスを外したり、接続したりする操作において、OS から送信されるデバイスの状態変化を通知するメッセージは、操作する時に必ず 1 回だけ送信されるとは限りません。デバイスによって複数回送信する場合もあります。また、他のデバイスの操作でもメッセージが送信されます。（例えば、CD ドライブに CD が挿入された場合など）。プログラミングの際はその点を考慮して作成して下さい。

3) デバイスを外した場合、アプリケーションがデバイスに対して行った設定が、再接続を行った際、デバイスでは初期状態に戻ってしまいます。再接続する際、デバイスを外す前の設定に戻す場合は、アプリケーションが設定した情報を保持し、再接続した後に設定を行う必要があります。

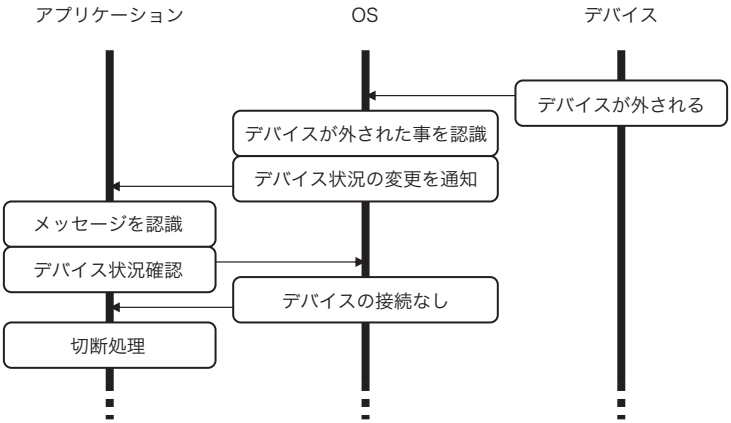
但し、仕様によってデバイスを PC へ接続したとき、起動に時間が掛かるデバイスもあります。再接続後、すぐ設定を行っても起動中のため設定できない場合があります。デバイスの仕様をよくご理解の上ご使用ください。

以下は、デバイスの接続及び取り外しを行う際の、基本的な流れです。

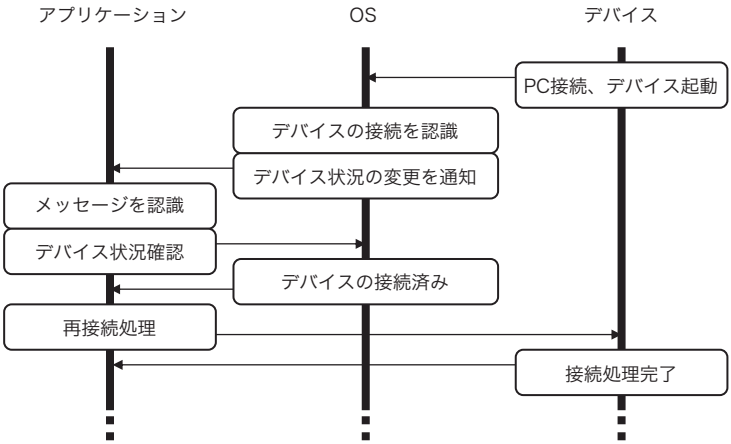
・ デバイスに接続



・ デバイスを取り外す



・ デバイスを再接続



[記述例]

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message,
                                    WPARAM wParam, LPARAM lParam)
{
    INT     nState;    // Device State
     :
     :
    switch( message ) {
          :
          :
        case WM_DEVICECHANGE:
            // Get Device State
            if( DcamGetDeviceState(&nState) ) {
                if(nState == DCAM_DEVSTATE_NODEVICE) {    // No device found
                    DisconnectionDevice();
                } else if(nState == DCAM_DEVSTATE_DEVICE) { // Device found
                    ConnectionDevice();
            }
        }
break;
:
```

改定履歴

改定日付	リビジョン	改定内容
2009 年 6 月 1 日	1.00	初版
2009 年 7 月 23 日	1.10	下記関数の記述を追加。 ・ DcamSetStandardTimeUnit ・ DcamGetStandardTimeUnit
2009 年 10 月 8 日	1.20	誤記訂正 DCamTmpCtrl.dll 情報の追加
2011 年 5 月 30 日	1.30	下記関数の記述を追加。 ・ DcamSetMPPMode ・ DcamGetMPPMode ・ DcamSetLineTime ・ DcamGetLineTime ・ DcamSetIntegralCapacity ・ DcamGetIntegralCapacity ・ DcamSetSensorSignalPulseWidth ・ DcamGetSensorSignalPulseWidth
2011 年 7 月 15 日	1.40	下記関数の記述を削除。 ・ DcamSetSensorSignalPulseWidth ・ DcamGetSensorSignalPulseWidth 下記関数の記述を追加。 ・ DcamSetDriveMode ・ DcamGetDriveMode 下記定数 (MPP モード) の記述を追加。 ・ DCAM_CCDMPPMODE_OFF ・ DCAM_CCDMPPMODE_ON
2011 年 9 月 28 日	1.41	DCamUSB のバージョン番号更新 DCamTmpCtrl のバージョン番号更新
2011 年 11 月 11 日	1.42	下記関数の記述を追加。 ・ DcamSetElectronicShutter ・ DcamGetElectronicShutter

改定日付	リビジョン	改定内容
2012 年 5 月 1 日	1.43	下記関数の記述を更新 ・ DcamGetImageSize ・ DcamGetCCDType ・ DcamSetCCDType ・ DcamSetMeasDataCount ・ DcamGetMeasDataCount ・ DcamGetCaptureByte ・ DcamCapture ・ DcamCaptureReverseX ・ DcamStop ・ DcamWait 下記関数の記述を追加 ・ DcamStopEx ・ DcamGetTotalCaptureBytes 下記補足説明追加 ・ 7.1 データ計測手順について ・ 7.2 "DcamStop()" の使用について
2012 年 6 月 28 日	2.00	ソフトウェアの Windows7 対応による
2014 年 2 月 25 日	2.01	以下の関数を追記 ・ DcamSetSensorSignalPulseWidth ・ DcamGetSensorSignalPulseWidth
2014 年 7 月 29 日	2.02	WindowsXP のサポート終了によるマニュアルの改訂 裏表紙の拠点一覧を作成

イメージセンサ用駆動回路ライブラリ DCamUSB/DCamTmpCtrl 関数仕様書

製造者

浜松ホトニクス株式会社 WEB SITE：http://www.hamamatsu.com/

固体事業部
〒 435-8558 静岡県浜松市東区市野町 1126-1 TEL(053)434-3311 FAX(053)434-5184

仙台営業所	〒 980-0011	宮城県仙台市青葉区上杉 1-6-11（日本生命仙台勾当台ビル 2 階）	TEL(022)267-0121	FAX(022)267-0135
筑波営業所	〒 305-0817	茨城県つくば市研究学園 D6 街区 8 画地（研究学園スクウェアビル 7 階）	TEL(029)848-5080	FAX(029)855-1135
東京営業所	〒 105-0001	東京都港区虎ノ門 3-8-21（虎ノ門 33 森ビル 5 階）	TEL(03)3436-0491	FAX(03)3433-6997
中部営業所	〒 430-8587	静岡県浜松市中区砂山町 325-6（日本生命浜松駅前ビル 4 階）	TEL(053)459-1112	FAX(053)459-1114
大阪営業所	〒 541-0052	大阪府大阪市中央区安土町 2-3-13（大阪国際ビル 10 階）	TEL(06)6271-0441	FAX(06)6271-0450
西日本営業所	〒 812-0013	福岡県福岡市博多区博多駅東 1-13-6（竹山博多ビル 5 階）	TEL(092)482-0390	FAX(092)482-0550
固体営業推進部	〒 435-8558	静岡県浜松市東区市野町 1126-1	TEL(053)434-3311	FAX(053)434-5184

文書番号：K46-B60010
ドキュメントリビジョン 2.02　2014 年 7 月 29 日