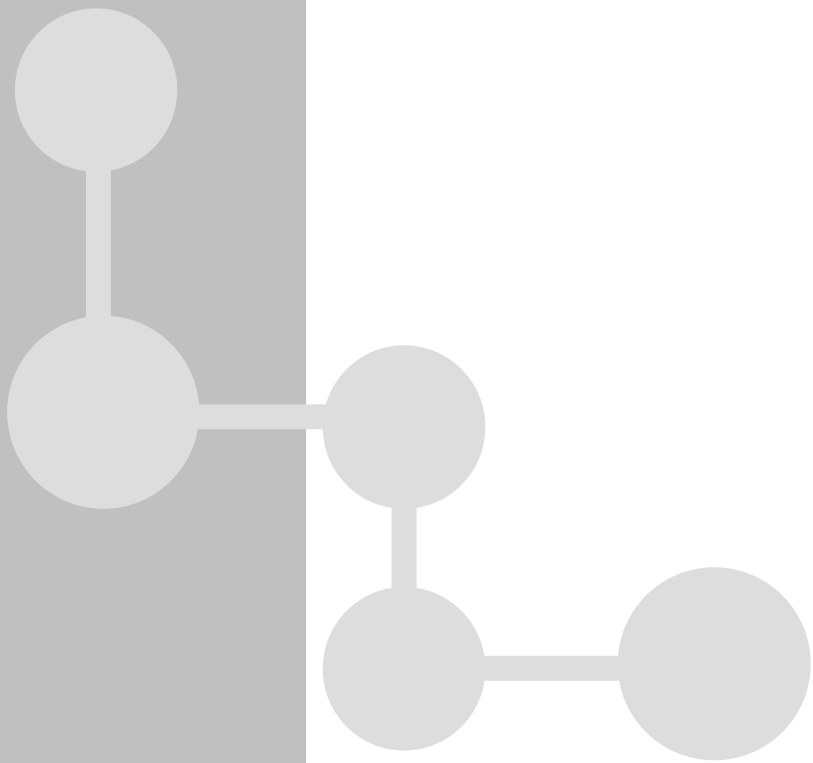
CASIO

CASSIOPEIA

DT-5100 シリーズ

.NET ライブラリマニュアル

【概要編】



Ver 3.00

変 更 履 歴

No	Revision	更新日	項	改訂内容
1	1.00	03/1/20	初版	初版発行
2	3.00	05/03/15		カシオライブラリマニュアル（.NET）開発マニュアルの§1～4をひとまとめにしました。
3				
4				
5				
6				
7				
8				
9				
１０				
１１				
１２				
１３				
１４				
１５				
１６				
１７				
１８				
１９				
２０				
２１				
２２				
２３				
２４				
２５				

目次

1 概要 1

1.1 カシオライブラリ 1
1.2 提供ファイルの構成 2

2 CE.NET 用クラスライブラリ 3

2.1 機能 3
2.2 開発環境 3
2.3 提供ファイル 3
2.3.1 C#.NET用クラスライブラリ 3
2.3.2 VB.NET用クラスライブラリ 3
2.4 使用方法 4

3 C#.NET用クラスライブラリ 5

3.1 C#.NET用クラスライブラリの機能 5
3.2 ラッパーにおけるDLL関数を呼び出し方の記述 5
3.3 エクスポート関数の宣言 5
3.4 カシオライブラリとC#での関数引数の対応表 6
3.5 カシオライブラリとC#での戻り値の対応 7
3.6 C#. NET アプリケーションでの使用例 8

4 VB.NET用クラスライブラリ開発 10

4.1 VB.NET用クラスライブラリの機能 10
4.2 ラッパーにおけるDLL関数を呼び出し方の記述 10
4.3 エクスポート関数の宣言 10
4.3.1 VB.NETとカシオライブラリの引数の対応 11
4.3.2 カシオライブラリとVBとの戻り値の対応 12
4.4 VB. NET アプリケーションでの使用例 14

1　概要

本マニュアルは DT-870/5100 用に、C#.NET および VB.NET 言語を使ってアプリケーションを開発する場合に、カシオライブラリを利用できるように、WindowsCE .NET コンパクトフレームワークに対応した C#.NET および VB.NET 用のクラスライブラリを提供する方法について述べたものです。

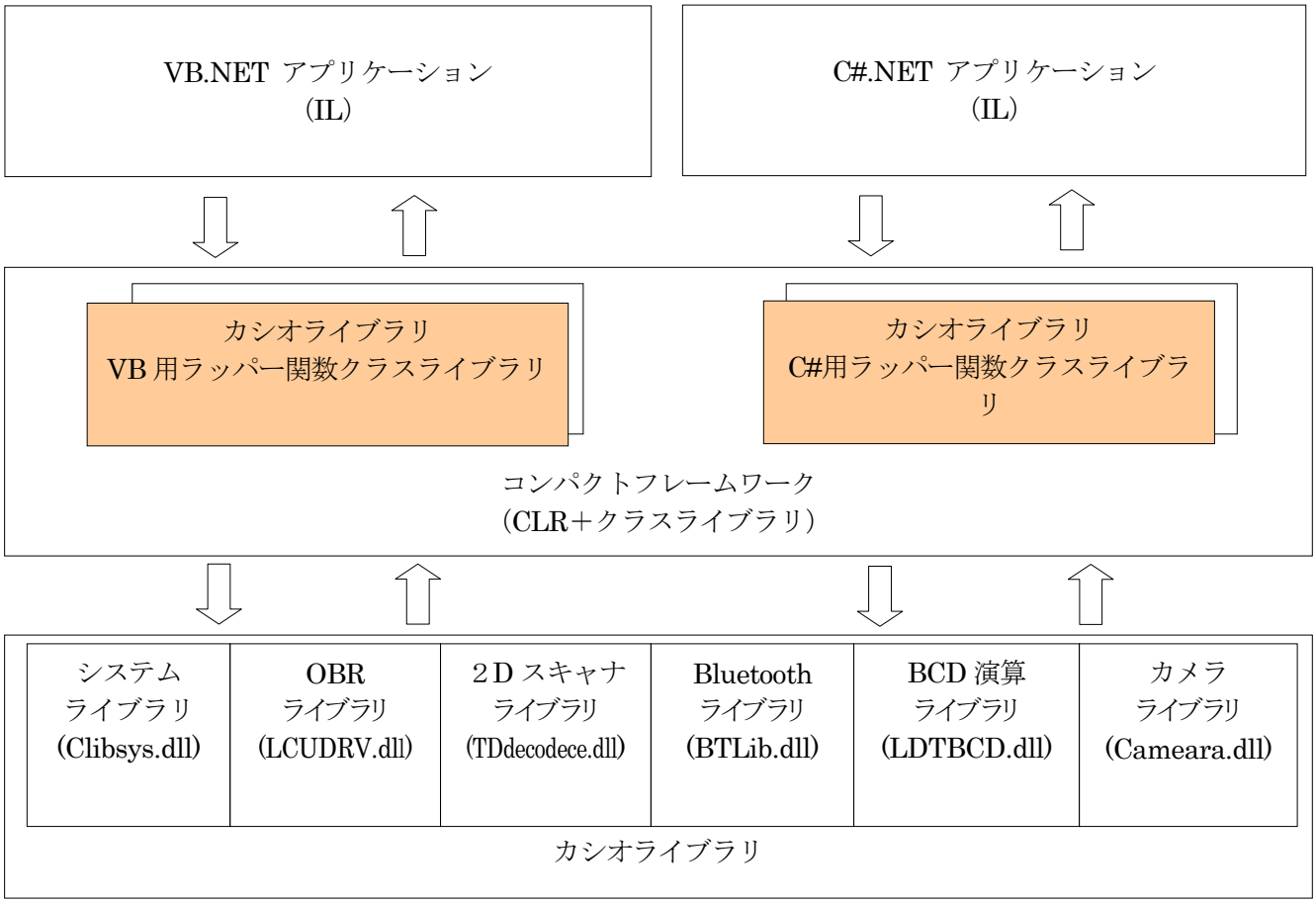
1.1　カシオライブラリ

今回対象となる DT-870/5100 カシオライブラリの構成と機種対応を以下に示します。

ライブラリ名	構成	関数の数	DT-870	DT-5100
システムライブラリ	Clibsys.lib Clibsys.dll Clibsys.h	34	○	○
OBR ライブラリ	LCUDRV.lib LCUDRV.dll OBRREQ.h	55		○
2D スキャナライブラリ	TDdecodece.dll TDdecodece.lib TDdecodece.h	92	○	
Bluetooth ライブラリ	BTLib.lib BTLib.dll BTLib.h	17	○	○
BCD 演算ライブラリ	LDTBCD.lib LDTBCD.dll LDTBCD.h	16		○
カメラライブラリ	Camera.dll CamHw.dll Camera.lib Camera.h Camera2.h	15		○
		合計　229		

1.2 提供ファイルの構成

各ソフトウェアの関連図を下図に示します。
カシオライブラリ（DLL 内の関数）の呼び出しは、VB 用、C#用のラッパー関数を介して「プラットフォーム呼び出しし（PInvoke：Platform Invocation Service）」で行います。プラットフォーム呼び出しは DLL からエクスポートされる関数を呼び出す汎用的なメカニズムです。
ラッパー関数はコンパクトフレームワークからカシオライブラリを呼び出すコードを記述し、アプリケーションがカシオライブラリDLLを利用できるように抽象化された関数インターフェースを提供し、コンパクトフレームのワーククラスライブラリコンポーネントとして位置づけられます。※



：提供ファイル（コンパクトフレーム用クラスライブラリ）※

2 CE.NET 用クラスライブラリ

2.1 機能

VisualStudio .NET 上で、C#.NET および VB.NET によるアプリケーション開発（ビルド）においてカシオライブラリを利用できるようにコンパクトフレームワーク用のクラスライブラリを提供します。
各関数の機能は

・DT-870 各ライブラリマニュアル
・DT-5100 各ライブラリマニュアル
を参照してください。

2.2 開発環境

・Microsoft Visual Studio .NET 2003

2.3 提供ファイル

2.3.1 C#.NET 用クラスライブラリ

C#.NET でアプリケーションを開発する場合に各ライブラリ関数呼び出しの引数と戻り値の調整を行います。

・ClbSysCs.dll　　システムライブラリ用クラスライブラリ
・LCUDRVCs　　OBR ライブラリ用クラスライブラリ
・TDDecodeceCs.dll　２D スキャナ用クラスライブラリ
・BTLibCs.dll　　Bluetooth ライブラリ用クラスライブラリ
・LDTBCDCs.dll　　BCD 演算ライブラリ用クラスライブラリ
・CameraC s .dll　　カメラライブラリ用クラスライブラリ

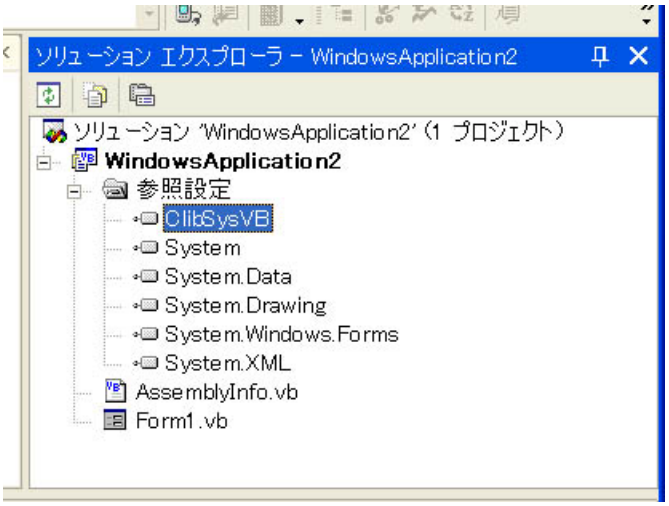
2.3.2 VB.NET 用クラスライブラリ

VB.NET でアプリケーションを開発する場合に各ライブラリ関数呼び出しの引数と戻り値の調整を行います。

・ClbSysVB.dll　　システムライブラリ用クラスライブラリ
・LCUDRVVB.dll　　OBR ライブラリ用クラスライブラリ
・TDDecodeceVB.dll　２D スキャナ用クラスライブラリ
・BTLibVB.dll　　Bluetooth ライブラリ用クラスライブラリ
・LDTBCDVB.dll　　BCD 演算ライブラリ用クラスライブラリ
・CameraVB.dll　　カメラライブラリ用クラスライブラリ

2.4 使用方法

各ライブラリ関数を C#.NET および VB.NET から利用するには、プロジェクトを作成し、使用したいライブラリのクラスライブラリをソリューションエクスプローラ内の「参照設定」に追加します。
下図は VB.NET でカシオライブラリを利用する例です。



C#.NET の場合も同様に C#.NET 用のクラスライブラリを参照設定に追加します。

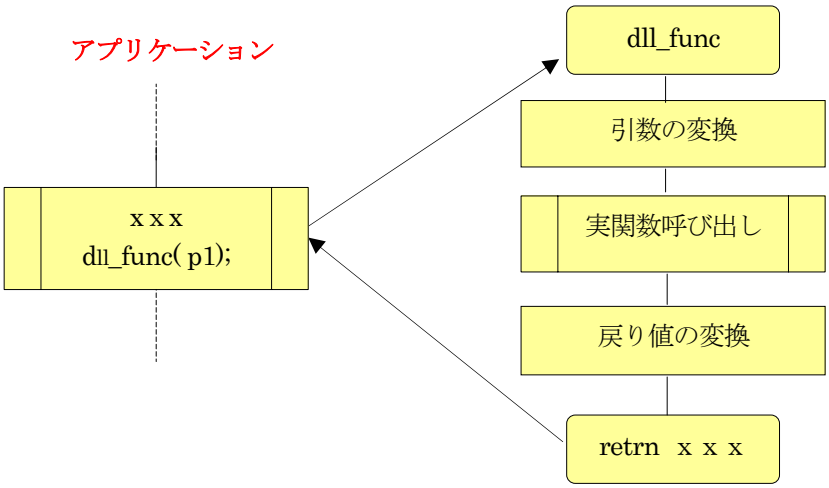
なお、プログラムコーディング例、各クラスライブラリインターフェイスは、（ライブラリマニュアル for C#.NET）および（ライブラリマニュアル for VB.NET）を参照してください。

3　C#.NET 用クラスライブラリ

3.1　C#. NET 用クラスライブラリの機能

C#.NET 用クラスライブラリが提供するラッパー関数は以下の処理を行います。

1) アプリケーションから受け取った引数をエクスポート関数用に変換します。
2) エクスポート関数を呼び出します。
3) エクスポート関数の戻り値をアプリケーション用に変換します。
4) 戻り値を設定してアプリケーションに制御を戻します。



3.2　ラッパーにおける DLL 関数を呼び出し方の記述

ソースコードの先頭に以下の記述を行います。

```
using System.Runtime.InteropServices;
```

3.3　エクスポート関数の宣言

DLL からエクスポートされる関数を利用するために以下の記述をします。

```
[DllImport( "user.dll" )]          ・・・・・　利用する関数をエクスポートする DLL 名
public static extern ＜戻り値のタイプ＞ dll_func( 引数 1, 引数 2, ・・・);
```

3.4 カシオライブラリと C#での関数引数の対応表

カシオライブラリ関数に渡す引数の対応を表にしたものを以下に示します。

	カシオライブラリの引数	C#呼び出し時の引数	ラッパー関数引数（アプリケーションに提示）
1	void	void	同左
2	char (8 ビット符号付き)	sbyte （8 ビット符号付き）	1 バイトデータとしての意味を持つ場合は sbyte、1 文字としての意味を持つ場合は char とする。
3	BYTE (8 ビット符号なし)	ｂyte （8 ビット符号なし）	ｂyte （8 ビット符号なし）
4	TCHAR (16 ビット)	string	ｃｈａｒ（16 ビット）
5	Short （16 ビット符号付）	short/Int16 （符号付 16 ビット）	同左
6	WORD （16 ビット符号なし）	ushort/UInt16 （16 ビット符号なし）	同左
7	Int （32 ビット符号付）	int/Int32	同左
8	Long （32 ビット符号付）	int/Int32	同左
9	DWORD （32 ビット符号なし）	uint/UInt32 （32 ビット符号なし）	同左
10	struct _bbb{ ... } bbb	struct bbb{...}	同左
11	char*	ref sbyte	同左
12	BYTE*	ref byte	同左
13	TCHAR*	ref string	同左
14	short*	ref short	同左
15	WORD*	ref ushort	同左
16	int*	ref Int32	同左
17	long*	ref Int32	同左
18	DWORD*	ref uint	同左
19	struct _bbb{ ... } *bbb （構造体のポインタ）	struct ＜名称＞ [...]; ＜名称＞ bbb; ref bbb;	同左（メンバの変換が必要な場合があります）

20	関数へのポインタ	public delegate bool CallBack1();
		CallBack1 x =
		new CallBack1(callback_sub1);
		Clib.dllfunc_callback1(5000, x);
			同左

※ [　　] 部分は CE ではできません。（VisualStudio.NET 2003＋Compact Framwork ）

3.5 カシオライブラリと C#での戻り値の対応

返却値の宣言記述の対応を表にしたものを以下に示します。

	カシオライブラリの記述	C#での関数リターン時の戻り値	ラッパー関数戻り値 (アプリケーションに提示)
1	void	void	同左
2	char (8 ビット符号付き)	sbyte （8 ビット符号付き）	同左
3	BYTE (8 ビット符号なし)	ｂyte （8 ビット符号なし）	同左
4	TCHAR (16 ビット)	string（16 ビット）	同左
5	Short （16 ビット符号付）	short/Int16 （16 ビット符号付）	同左
6	WORD （16 ビット符号なし）	ushort/UInt16 （16 ビット符号なし）	同左
7	Int （32 ビット符号付）	int/Int32	同左
8	Long （32 ビット符号付）	int/Int32	同左
9	DWORD （32 ビット符号なし）	uint/UInt32	同左
10	struct ＜構造体名＞	struct ＜構造体名＞	同左
11	char*	IntPtr で受け取って返す。	sbyte*
12	BYTE*	IntPtr で受け取って返す	byte*
13	TCHAR*	IntPtr で受け取って返す	char*
14	short*	IntPtr で受け取って返す	short*
15	WORD*	IntPtr で受け取って返す	ushort*
16	int*	IntPtr で受け取って返す	int*
17	long*	IntPtr で受け取って返す	int*
18	DWORD*	IntPtr で受け取って返す	uint*
19	構造体*	IntPtr で受け取って返す	構造体*

※ 　　部分は CE ではできません。（VisualStudio.NET 2003＋Compact Framwork ）

3.6 C#. NET アプリケーションでの使用例

カシオライブラリのシステムライブラリ関数を利用する場合のサンプルを以下に示します。
アプリケーション作成時に提供クラスライブラリファイル（CalibCs.dll）をプロジェクトに追加します。

アプリケーション側の記述例

```
using System;
using CalibCs;

namespace ClbsysSmp1
{
            ・・・・・・・・・・・・・・・・・・
    public class CeApiSample
    {
        public void ApiCe1( )
        {
            uint a1 = new uint();

            // Wakeup 要因による電源許可禁止の取得
            // BOOL CLBGetBootableButtons( DWORD * )
            bool x2 = ClbSysCs.CLBGetBootableButtons( ref a1 );

            // 左トリガーキーによる電源 ON は有効に
            uint a2 = (a1 | (uint)ClbSysCs.CLB_BUTTON_LEFTTRIGGER);

            // Wakeup 要因による電源 ON 許可禁止
            // BOOL CLBSetBootableButtons( DWORD )
            bool x1 = ClbSysCs.CLBSetBootableButtons( a2 );

    ・・・・・・・・・・・・・・・・・・・・
            uint b1 = new uint();
            uint b2 = new uint();

            // 電源 OFF を禁止する時間の取得
            // BOOL CLBGetOffMaskTime( DWORD *, DWORD * )
            bool x4 = ClbSysCs.CLBGetOffMaskTime( ref b1, ref b2 );

            // 電源 ON 後指定時間 OFF を禁止する
            // BOOL CLBSetOffMaskTime( DWORD, DWORD )
            bool x3 = ClbSysCs.CLBSetOffMaskTime( 10, 0 );

    ・・・・・・・・・・・・・・・・・・・・
        }

        public void ApiCe2( )
        {

    ・・・・・・・・・・・・・・・・・・・・
            // ｶｰﾄﾞのアクセス終了を待って電源 OFF
            // void CLBPowerOff( void )
            ClbSysCs.CLBPowerOff();

        }
    }
}
```

ClbsysCs.cs の内容

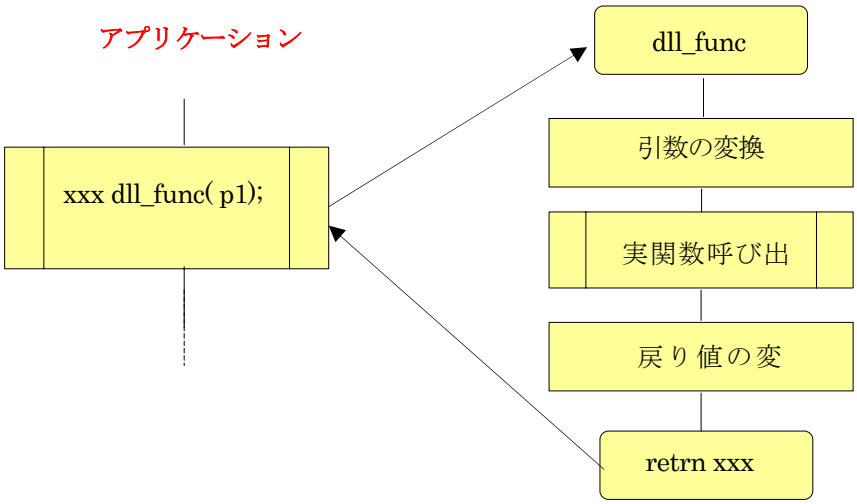
```
using System;
using System.Windows.Forms;
using System.Runtime.InteropServices;

namespace ClbsysCs
{
	/// <summary>
	/// Summary description for ClbsysCs.
	/// </summary>

	public class ClbsysCs
	{
		public Clbsys()
		{
			// TODO: Add constructor logic here
		}

	/*	Constant definition to import */
		// Bootable Buttons - DWORD dwBootableButtons
		public const uint CLB_BUTTON_LEFTTRIGGER	= 0x1000;
		public const uint CLB_BUTTON_RIGHTTRIGGER	= 0x0800;
～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～
～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～

	/*	Definition description of a Wrapper function */
		// Wakeup 要因による電源 ON 許可禁止
		public static bool	CLBSetBootableButtons( uint dwBootableButtons )
		{
			bool	ret = new bool();

			try
			{
				ret = ClbSys.CLBSetBootableButtons( dwBootableButtons );
			}
			catch(Exception exc)
			{
				MessageBox.Show( exc.ToString(), "Info CLBSetBootableButtons()" );
			}
			return ret;
		}

		// ｶｰﾄﾞ のアクセス終了を待って電源 OFF
		public static void	CLBPowerOff( )
		{

			try
			{
				ret = ClbSys.CLBPowerOff( );
			}
			catch(Exception exc)
			{
				MessageBox.Show( exc.ToString(), "Info CLBPowerOff()" );
			}
		}
～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～
～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～
	}

	public class ClbSys
	{
	/*	Description of the function to import */

		[DllImport("ClbSys.dll")]
		public static extern bool CLBSetBootableButtons( uint dwBootableButtons );

～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～
～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～

		[DllImport("ClbSys.dll")]
		public static extern void CLBPowerOff( );

～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～
～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～～
	}
}
```

4　VB.NET 用クラスライブラリ開発

4.1　VB. NET 用クラスライブラリの機能

VB.NET 用クラスライブラリが提供するラッパー関数は以下の処理を行います。

5）アプリケーションから受け取った引数をエクスポート関数用に変換します。
6）エクスポート関数を呼び出します。
7）エクスポート関数の戻り値をアプリケーション用に変換します。
8）戻り値を設定してアプリケーションに制御を戻します。



4.2　ラッパーにおける DLL 関数を呼び出し方の記述

ソースコードの先頭に以下の記述を行います。

```
import System.Runtime.InteropServices
```

4.3　エクスポート関数の宣言

DLL からエクスポートされる関数を利用するために以下の記述をします。
戻り値がない関数は Sub、戻り値がある関数は Function で宣言します。

1）　Declare Auto Sub dll_func lib　“user.dll”　(引数 1, ・・・)
　　・・・・・　戻り値がない場合

2）　Declare Auto Function dll_func lib　“user.dll”　(引数 1, ・・・)　As　＜戻り値タイプ＞
　　・・・・・　戻り値がある場合

4.3.1 VB.NET とカシオライブラリの引数の対応

カシオライブラリ関数に渡す引数の対応を表にしたものを以下に示します。

	カシオライブラリの引数	VB 呼び出し時の引数	ラッパー関数引数 (アプリケーションに提示)
1	void	無記述 sub()	同左
2	char xxx (8 ビット符号付き)	ByVal xxx As SByte （8 ビット符号あり）	1 バイトデータとして意味を持つ場合は、 ByVal xxx As SByte 1 文字としての意味を持つ場合は、 ByVal xxx As Char（16 ビット）
3	BYTE (8 ビット符号なし)	ByVal xxx As Byte （8 ビット符号なし）	1 バイトデータとして意味を持つ場合は、 ByVal xxx As Byte 1 文字としての意味を持つ場合は、 ByVal xxx As Char（16 ビット）
4	ＴCHAR (16 ビット)	ByVal xxx As Char （16 ビット）	同左
5	Short （16 ビット符号付）	ByVal xxx As Short/Int16 （符号付 16 ビット）	同左
6	WORD （16 ビット符号なし）	UInt16 （符号付 16 ビット）	ByVal xxx As UInt16
7	Int （32 ビット符号付）	ByVal xxx As Integer/Int32 （32 ビット符号付）	同左
8	Long （32 ビット符号付）	ByVal xxx As Integer/Int32 （32 ビット符号付）	同左
9	DWORD （32 ビット符号なし）	ByVal xxx As Integer/Int32 （32 ビット符号付）	ByVal xxx As UInt32
10	構造体	ByVal xxx As (Structure)	同左
11	char*	ByVal xxx As string ByRef xxx As SByte ByVal xxx As IntPtr ：	同左（オーバーロードにより適切なものを選択）
12	BYTE*	ByRef xxx As Byte	同左
13	TCHAR*	ByRef xxx As Char	同左
14	short*	ByRef ｘｘｘ As Short/Int16	同左
15	WORD*	ByRef ｘｘｘ As UInt16	ByRef ｘｘｘ As UInt16
16	int*	ByRef ｘｘｘ As Integer/Int32	同左
17	long*	ByRef ｘｘｘ As Integer/Int32	同左
18	DWORD*	ByRef ｘｘｘ As Integer/Int32	ByRef ｘｘｘ As UInt32
19	構造体*	ByRef xxx As (Structure)	同左
20	関数へのポインタ	ByVal xxx As CallBack	コールバック関数のデリゲートオブジェクトを作成し、引数として渡す Public Delegate Function _ CallBack(ByVal dat As Int32) As Int32

4.3.2 カシオライブラリと VB との戻り値の対応

カシオライブラリ関数からの戻り値の対応を表にしたものを以下に示します。

	カシオライブラリの記述	VB での関数リターン時の戻り値	ラッパー関数戻り値 (アプリケーションに提示)
1	void	無記述 sub()	同左
2	char (8 ビット符号付き)	Dim dat As Byte （8 ビット符号あり） dat = function()	1 バイトデータとして意味を持つ場合は、 Dim dat As SByte(8 ビット) 1 文字としての意味を持つ場合は、 Dim dat As Char（16 ビット）
3	BYTE (8 ビット符号なし)	Dim dat As SByte （8 ビット符号なし） dat = function()	1 バイトデータとして意味を持つ場合は、 Dim dat As Byte(8 ビット) 1 文字としての意味を持つ場合は、 Dim dat As Char（16 ビット）
4	ＴCHAR (16 ビット)	Dim dat As Char （16 ビット） dat = function()	同左
5	Short （16 ビット符号付）	Dim dat As Short/Int16 （符号付 16 ビット） dat = function()	ByVal xxx As UInt16
6	WORD （16 ビット符号なし）	Dim dat As Short/Int16 （符号付 16 ビット） dat = function()	同左
7	Int （32 ビット符号付）	Dim dat As Integer/Int32 dat = function()	同左
8	Long （32 ビット符号付）	Dim dat As Integer/Int32 dat = function()	同左
9	DWORD （32 ビット符号なし）	Dim dat As Integer/Int32 dat = function()	ByVal xxx As UInt32
10	構造体	Dim stc As struct1 stc = function()	同左
11	char*	Dim ptr As IntPtr Dim dat As Byte ptr = function() dat = Marshal.ReadByte(ptr)	Dim dat As Byte
12	BYTE*	Dim ptr As IntPtr Dim dat As Int16 ptr = function() dat=Marshal.ReadInt16(ptr)	Dim dat As Int16
13	TCHAR*	Dim ptr As IntPtr Dim dat As Char ptr = function() dat=ChrW(Marshal.ReadInt16(ptr))	Dim dat As Char
14	short*	Dim ptr As IntPtr Dim dat As Int16 ptr = function() dat=Marshal.ReadInt16(ptr)	Dim dat As Int16
15	WORD*	Dim ptr As IntPtr Dim dat As Int32 ptr = function() dat=Marshal.ReadInt32(ptr)	Dim dat As UInt16

16	int*	Dim ptr As IntPtr Dim dat As Int32 ptr = function() dat=Marshal.ReadInt32(ptr)	Dim dat As Int32
17	long*	Dim ptr As IntPtr Dim dat As Int32 ptr = function() dat=Marshal.ReadInt32(ptr)	Dim dat As Int32
18	DWORD*	Dim ptr As IntPtr Dim dat As UInt32 Dim dmy() as UInt32 ptr = function() Marshal.Copy(ptr,dat,0,1) dat = dmy(0)	Dim dat As UInt32 （Marshal.Copy により獲得する。 Copy()では配列指定しか出来ないため、 ダミー配列経由で獲得する）
19	構造体*	Dim ptr As IntPtr Dim stc As struct1 ptr = function() stc = Marshal.PtrToStructure(ptr, GetType(struct1))	Dim stc As struct1

※ 　　部分は CE ではできません。（VisualStudio.NET 2003＋Compact Framwork β2 で構造体の中に配列が定義されている場合）

4.4 VB.NET アプリケーションでの使用例

カシオライブラリのシステムライブラリ関数を利用する場合のサンプルを以下に示します。
アプリケーション作成時に提供ソースファイル（VBClibsys.vb）を組み込む。
VBClibsys.vb の内容

```
Imports System.Runtime.InteropServices

Public Class VBClibsys
    '-----------------------------------------
    ' 定数定義
    '-----------------------------------------
    'BUTTON
    Public Const CLB_BUTTON_LEFTTRIGGER As Int64 = 1
    Public Const CLB_BUTTON_RIGHTTRIGGER As Int64 = 2
                            :
                            :

    '-----------------------------------------
    ' システムライブラリ
    '-----------------------------------------
    Declare Auto Function CLBSetBootableButtons Lib "Clbsys" (ByVal dwBootableButtons As Int64) As Boolean
    Declare Auto Function CLBGetBootableButtons Lib "Clbsys" (ByRef pdwBootableButtons As Int64) As Boolean
    Declare Auto Function CLBSetOffMaskTime Lib "Clbsys" (ByVal pdwKey As Int64, ByVal dwCard As Int64) As Boolean
    Declare Auto Function CLBGetOffMaskTime Lib "Clbsys" (ByRef dwKey As Int64, ByRef dwCard As Int64) As Boolean
    Declare Auto Function CLBPowerOff Lib "Clbsys" ()
                            :
                            :

End Class
```

アプリケーションの記述

```
Imports System.Runtime.InteropServices
Imports (アプリケーション名).VBClibsys

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    ' ----- Wakeup要因による電源ONの有効／無効設定を行います。
    Dim bl As Boolean
    bl = VBClibsys.CLBSetBootableButtons(VBClibsys.CLB_BUTTON_LEFTTRIGGER)
    '       先頭で Imports により名前修飾しているため下記でも可
    bl = CLBSetBootableButtons(CLB_BUTTON_LEFTTRIGGER)

    ' ----- Wakeup要因による電源ONの有効／無効状態を読み出します。
    Dim dat As Int64
    bl = CLBGetBootableButtons(dat)
    If (dat And CLB_BUTTON_LEFTTRIGGER) <> 0 Then MsgBox("左ボタン有効")

    ' ----- 電源ON後、指定された時間電源OFFを禁止するための設定を行います。
    bl = CLBSetOffMaskTime(10, 0)       ' 10秒
    ' ----- 本体電源をOFFします。
    CLBPowerOff()

    ' ----- 本体とIOボックスとの接続状態を取得します。また、接続状態を監視する時間の設定も同時に行います。
    Dim ans As Int32
    ans = CLBCheckCharger(0)
End Sub
```

DT-5100

.NET ライブラリマニュアル

【概要編】

Ver3.00
発行元：カシオ計算機株式会社
〒162-8543
東京都渋谷区本町 1-6-2
ｼｽﾃﾑｿﾘｭｰｼｮﾝ営業統轄部
TEL:03-5334-4638