KEYENCE
ILDLEP1 プロバイダ

Version 1.0.0

ユーザーズ ガイド

October 12, 2018

備考:

【改版履歴】

バージョン	日付	内容
1.0.0	2018-10-12	初版.

【動作確認機種】

機種	バージョン	注意事項
IL シリーズ		通信ユニット(DL-EP1) + アンプユニット(IL-1000 等) + センサユニット(IL-S100 等)の構成

目次

1. はじめに

KEYENCE ILDLEP1 プロバイダ(以下 ILDLEP1 プロバイダ)は，KEYENCE 製レーザ式変位センサ(IL シリーズ)に対し，通信ユニット(DL-EP1)を通じてアクセスを行う ORiN2 CAO プロバイダです.

本ドキュメントでは，ILDLEP1 プロバイダの概要と，実装されている CAO インターフェイス(関数仕様)について説明します.

2. プロバイダの概要

2.1. インストール

ILDLEP1 プロバイダモジュールは，下記の DLL で構成されています．ORiN2 SDK のインストーラでインストールした場合は，インストール作業は不要です．手動でインストールする場合は，表 2-1 のように実行してください．

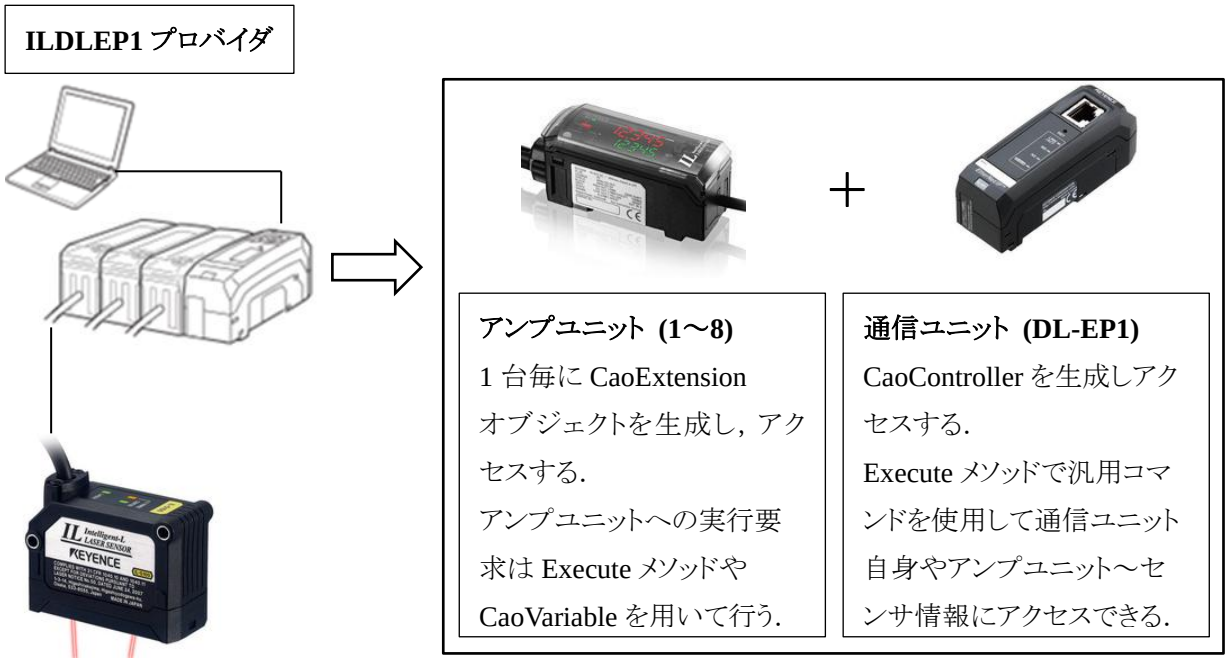
表 2-1 ILDLEP1 プロバイダ

ファイル名	CaoProvKEYENCEILDLEP1.dll
ProgID	CaoProv.KEYENCE.ILDLEP1
レジストリ登録	regsvr32 CaoProvKEYENCEILDLEP1.dll.dll
レジストリ登録の抹消	regsvr32 /u CaoProvKEYENCEILDLEP1.dll.dll

2.2. 概要

ILDLEP1 プロバイダは，CIP プロトコルを用いて DL-EP1 へのアクセスを行います．

DL-EP1 に接続されたアンプユニット毎に CaoExtension オブジェクトを生成し，Execute メソッド又は CaoVariable への読み書きを行いセンサ情報へアクセスします．



3. メソッド・プロパティ

3.1. CaoWorkspace::AddController メソッド

ILDLEP1 プロバイダは AddController 時に通信用の接続パラメータを参照し，通信の接続を行います.

書式 AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,<bstrPCName:BSTR>,
<bstrOption:BSTR>)

bstrCtrlName : [in] コントローラ名
bstrProvName : [in] プロバイダ名. 固定値 = "CaoProv.KEYENCE.ILDLEP1"
bstrPCName : [in] プロバイダの実行マシン名
bstrOption : [in] オプション文字列

以下にオプション文字列に指定するリストを示します.

表 3-1 CaoWorkspace::AddController のオプション文字列

オプション ([1])	説明
Conn=<接続パラメータ>	必須. 通信形態と接続パラメータ (参照 3.1.1)
MyIP[=<自 IP アドレス>]	自 IP アドレス. (複数 NIC 用途) (デフォルト:指定なし)
ConnTimeout[=<接続タイムアウト時間>]	接続時のタイムアウト時間. (ミリ秒) (デフォルト:3000)
TimeOut[=<タイムアウト時間>]	送受信時のタイムアウト時間. (ミリ秒) (デフォルト:3000)

3.1.1. Conn オプション

通信形態と接続パラメータを指定します.
以下に Conn オプションの接続パラメータ文字列を示します.

Ethernet/IP デバイス

"Conn=ETH:<Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"
"Conn=TCP:<Dest IP Address>[:<Dest Port No>[:<Src IP Address>[:<Src Port No>]]]"
<Dest IP Address> : TCP/IP 接続先 IP アドレス.
例:"127.0.0.1", "192.168.0.1"

[1] 角括弧("[]")内は省略可能を示します. また，各パラメータの解説中の下線部はオプションを指定しなかった時のデフォルト値になります.

<Dest Port No>	:	TCP/IP 接続先ポート番号 例:44818, 5006, 5007
<Src IP Address>	:	自 IP アドレス. (複数 NIC 用途) ([2])
<Src Port No>	:	自ポート番号. (複数 NIC 用途)

3.2. CaoController::Execute メソッド

CaoController クラスの Execute メソッドは, コマンドを実行するためのメソッドです. 第 1 引数にコマンド名, 第 2 引数にコマンドのパラメータを指定します. 各コマンドの詳細は 4.1 章を参照して下さい.

書式 [<vntRet:VARIANT> =]Execute(<bstrCmd:BSTR>[,<vntParam:VARIANT>])

<bstrCmd>	:	[in] コマンド名
<vntParam>	:	[in] パラメータ

3.3. CaoController::AddVariable メソッド

CaoController クラスの AddVariable メソッドは, 変数オブジェクトを作成するためのメソッドです. 使用できる変数一覧は 5.1 章を参照して下さい.

書式 AddVariable(<bstrVariableName:BSTR>[,<bstrOption:BSTR>])

<bstrVariableName>	:	[in] 変数名
<bstrOption>	:	[in] オプション文字列

3.4. CaoController::AddExtension メソッド

CaoController クラスの AddExtension メソッドは, DL-EP1 に接続されたアンプユニット毎の Extension オブジェクトを作成するためのメソッドです.

書式 AddExtension(<bstrChannelNo:BSTR>)

<bstrChannelNo>	:	[in] アンプユニットの ID 番号. "CH<ID 番号>"の書式で指定. (ID 番号:1～8)

3.5. CaoExtension::Execute メソッド

CaoExtension クラスの Execute メソッドは, コマンドを実行するためのメソッドです. 第 1 引数にコマンド名, 第 2 引数にパラメータを指定します. 各コマンドの詳細は 4.2 章を参照して下さい.

[2] Conn オプションと MyIP オプションの両方で自 IP アドレスを指定するとエラーになります. 利用する場合は必ずどちらか片方で指定するようにしてください.

書式 [<vntRet:VARIANT> =]Execute(<bstrCmd:BSTR>[,<vntParam:VARIANT>])

<bstrCmd> : [in] コマンド名
<vntParam> : [in] パラメータ

3.6. CaoExtension::AddVariable メソッド

CaoExtension クラスの AddVariable メソッドは，変数オブジェクトを作成するためのメソッドです．使用できる変数一覧は 5.2 章を参照して下さい．

書式 AddVariable(<bstrVariableName:BSTR>[,<bstrOption:BSTR>])

<bstrVariableName> : [in] 変数名
<bstrOption> : [in] オプション文字列

3.7. CaoController::get_VariableNames プロパティ

CaoController クラスの変数名リストを取得します．

3.8. CaoController::get_ExtensionNames プロパティ

CaoController クラスの Extension 名リストを取得します．

3.9. CaoExtension::get_VariableNames プロパティ

CaoExtension クラスの変数名リストを取得します．

3.10. CaoVariable::put_ID プロパティ

CaoVariable の ID を設定します．設定値は引数で指定した値を設定します．
一部の変数へアクセスする際の ID 指定に用います．

3.11. CaoVariable::get_ID プロパティ

CaoVariable の ID を取得します．

3.12. CaoVariable::put_Value プロパティ

変数名で指定した CaoVariable の変数値を設定します．設定値は引数で指定した値を設定します．

3.13. CaoVariable::get_Value プロパティ

変数名で指定した CaoVariable の変数値を取得します．

4. コマンド一覧

4.1. CaoController クラス

表 4-1 CaoController クラス コマンド一覧

コマンド名	機能	
Raw	汎用コマンドを送出します.	P10

4.1.1. CaoController::Execute("Raw") コマンド

汎用コマンドの送出を行う.

書式 Raw(<vntDataArray:VARIANT>)

<vntDataArray> : [in] コマンドデータ
(VT_ARRAY | VT_UI1)
[0] サービスコード
[1] クラス ID
[2] インスタンス ID
[3] アトリビュート ID(Low)
[4] アトリビュート ID(High)
[n] サービスデータ

- アトリビュート ID が 8Bit の場合は High 側を 0x00 で指定すること
- 要素 5 以降はサービスデータとして扱う
- サービスデータは Low 側から 8Bit ずつ指定すること
- サービスデータが奇数 Byte の場合は内部で NULL(0x00) パディングされる

戻り値 : [out] 応答データ
(VT_ARRAY | VT_UI1)

- 応答データが無いものは VT_EMPTY が返る
- 応答データは Low 側から 8Bit ずつ格納される

使用例

```
Byte param = new Byte[5];
```

```
param[0] = 0x0E;
param[1] = 0x67;
param[2] = 0x00;
param[3] = 0x64;
param[4] = 0x00;
Object result = m_CaoController.Execute("Raw", param);
```

4.2. CaoExtension クラス

表 4-2 CaoExtension クラス コマンド一覧

コマンド名	機能	
ZeroShift	ゼロシフト実行要求を行います.	P12
ZeroShiftReset	ゼロシフトリセット要求を行います.	P12
Reset	リセット要求を行います.	P12
InitReset	イニシャルリセット要求を行います.	P13
SystemParametersSet	システムパラメータセット要求を行います.	P13
ToleranceTuning	公差チューニング要求を行います.	P13
DecideHIGH1stPTOf2PTTuning	2 点チューニング HIGH 側 1 点目決定操作要求を行います.	P14
DecideHIGH2ndPTOf2PTTuning	2 点チューニング HIGH 側 2 点目決定操作要求(HIGH 側設定値決定)を行います.	P14
DecideLOW1stPTOf2PTTuning	2 点チューニング LOW 側 1 点目決定操作要求を行います.	P14
DecideLOW2ndPTOf2PTTuning	2 点チューニング LOW 側 2 点目決定操作要求(LOW 側設定値決定)を行います.	P15
DecideCALSET1	キャリブレーション SET1 決定操作要求を行います.	P15
DecideCALSET2	キャリブレーション SET2 決定操作要求(キャリブレーション実行)を行います.	P15
DecideCALSET1OfCALCV2PT	演算値 2 点キャリブレーション SET1 決定操作要求を行います.	P16
DecideCALSET2OfCALCV2PT	演算値 2 点キャリブレーション SET2 決定操作要求(演算値 2 点キャリブレーション実行)を行います.	P16
DecideCALSET1OfCALCV3PT	演算値 3 点キャリブレーション SET1 決定操作要求を行います.	P16
DecideCALSET2OfCALCV3PT	演算値 3 点キャリブレーション SET2 決定操作要求を行い	P17

	ます.	
DecideCALSET3OfCALCV3PT	演算値 3 点キャリブレーション SET3 決定操作要求(演算値 3 点キャリブレーション実行)を行います.	P17
Req1PTTuningForStepCountFilter	段差カウントフィルタ用 1 点チューニング要求を行います.	P17
Decide1stPTOf2PTTuningForStepCountFilter	段差カウントフィルタ用 2 点チューニング 1 点目決定操作要求を行います.	P18
Decide2ndPTOf2PTTuningForStepCountFilter	段差カウントフィルタ用 2 点チューニング 2 点目決定操作要求(HIGH 側/LOW 側設定値決定)を行います.	P18

4.2.1. CaoExtension::Execute("ZeroShift") コマンド

ゼロシフト実行要求を行います.

書式　ZeroShift ()

引数　:　無し
戻り値　:　無し

使用例

```
m_CaoExtension.Execute("ZeroShift");
```

4.2.2. CaoExtension::Execute("ZeroShiftReset") コマンド

ゼロシフトリセット要求を行います.

書式　ZeroShiftReset ()

引数　:　無し
戻り値　:　無し

使用例

```
m_CaoExtension.Execute("ZeroShiftReset");
```

4.2.3. CaoExtension::Execute("Reset") コマンド

リセット要求を行います.

書式　Reset ()

引数　：　無し
戻り値　：　無し

使用例

```
m_CaoExtension.Execute("Reset");
```

4.2.4. CaoExtension::Execute("InitReset") コマンド

イニシャルリセット要求を行います.

書式　InitReset ()

引数　：　無し
戻り値　：　無し

使用例

```
m_CaoExtension.Execute("InitReset");
```

4.2.5. CaoExtension::Execute("SystemParametersSet") コマンド

システムパラメータセット要求を行います.

書式　SystemParametersSet ()

引数　：　無し
戻り値　：　無し

使用例

```
m_CaoExtension.Execute("SystemParametersSet");
```

4.2.6. CaoExtension::Execute("ToleranceTuning") コマンド

公差チューニング要求を行います.

書式　ToleranceTuning ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("ToleranceTuning");
```

4.2.7. CaoExtension::Execute(″DecideHIGH1stPTOf2PTTuning″) コマンド

2 点チューニング HIGH 側 1 点目決定操作要求を行います.

書式 DecideHIGH1stPTOf2PTTuning ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideHIGH1stPTOf2PTTuning");
```

4.2.8. CaoExtension::Execute(″DecideHIGH2ndPTOf2PTTuning″) コマンド

2 点チューニング HIGH 側 2 点目決定操作要求(HIGH 側設定値決定)を行います.

書式 DecideHIGH2ndPTOf2PTTuning ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideHIGH2ndPTOf2PTTuning");
```

4.2.9. CaoExtension::Execute(″DecideLOW1stPTOf2PTTuning″) コマンド

2 点チューニング LOW 側 1 点目決定操作要求を行います.

書式 DecideLOW1stPTOf2PTTuning ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideLOW1stPTOf2PTTuning");
```

4.2.10. CaoExtension::Execute("DecideLOW2ndPTOf2PTTuning") コマンド

2 点チューニング LOW 側 2 点目決定操作要求(LOW 側設定値決定)を行います.

書式 DecideLOW2ndPTOf2PTTuning ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideLOW2ndPTOf2PTTuning");
```

4.2.11. CaoExtension::Execute("DecideCALSET1") コマンド

キャリブレーション SET1 決定操作要求を行います.

書式 DecideCALSET1 ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideCALSET1");
```

4.2.12. CaoExtension::Execute("DecideCALSET2") コマンド

演算値 2 点キャリブレーション SET2 決定操作要求(演算値 2 点キャリブレーション実行)を行います.

書式 DecideCALSET2 ()

引数 : 無し

戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideCALSET2");
```

4.2.13. CaoExtension::Execute(″DecideCALSET1OfCALCV2PT″) コマンド

演算値 2 点キャリブレーション SET1 決定操作要求を行います.

書式 DecideCALSET1OfCALCV2PT ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideCALSET1OfCALCV2PT");
```

4.2.14. CaoExtension::Execute(″DecideCALSET2OfCALCV2PT″) コマンド

演算値 2 点キャリブレーション SET2 決定操作要求（演算値 2 点キャリブレーション実行）を行います.

書式 DecideCALSET2OfCALCV2PT ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideCALSET2OfCALCV2PT");
```

4.2.15. CaoExtension::Execute(″DecideCALSET1OfCALCV3PT″) コマンド

演算値 3 点キャリブレーション SET1 決定操作要求を行います.

書式 DecideCALSET1OfCALCV3PT ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("DecideCALSET1OfCALCV3PT");
```

4.2.16. CaoExtension::Execute("DecideCALSET2OfCALCV3PT") コマンド

演算値 3 点キャリブレーション SET2 決定操作要求を行います.

書式　DecideCALSET2OfCALCV3PT ()

引数　:　無し
戻り値　:　無し

使用例

```
m_CaoExtension.Execute("DecideCALSET2OfCALCV3PT");
```

4.2.17. CaoExtension::Execute("DecideCALSET3OfCALCV3PT") コマンド

演算値 3 点キャリブレーション SET3 決定操作要求(演算値 3 点キャリブレーション実行)を行います.

書式　DecideCALSET3OfCALCV3PT ()

引数　:　無し
戻り値　:　無し

使用例

```
m_CaoExtension.Execute("DecideCALSET3OfCALCV3PT");
```

4.2.18. CaoExtension::Execute("Req1PTTuningForStepCountFilter") コマンド

段差カウントフィルタ用 1 点チューニング要求を行います.

書式　Req1PTTuningForStepCountFilter ()

引数　:　無し
戻り値　:　無し

使用例

```
m_CaoExtension.Execute("Req1PTTuningForStepCountFilter");
```

4.2.19. CaoExtension::Execute("Decide1stPTOf2PTTuningForStepCountFilter") コマンド

段差カウントフィルタ用 2 点チューニング 1 点目決定操作要求を行います.

書式 Decide1stPTOf2PTTuningForStepCountFilter ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("Decide1stPTOf2PTTuningForStepCountFilter");
```

4.2.20. CaoExtension::Execute("Decide2ndPTOf2PTTuningForStepCountFilter") コマンド

段差カウントフィルタ用 2 点チューニング 2 点目決定操作要求(HIGH 側/LOW 側設定値決定)を行います.

書式 Decide2ndPTOf2PTTuningForStepCountFilter ()

引数 : 無し
戻り値 : 無し

使用例

```
m_CaoExtension.Execute("Decide2ndPTOf2PTTuningForStepCountFilter");
```

5. 変数一覧

5.1. CaoController クラス

5.1.1. システム変数

表 5-1 CaoController クラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@MAKER_NAME	VT_BSTR	メーカ名="KEYENCE"を返す.	○	－
@VERSION	VT_BSTR	プロバイダバージョン情報を返す.	○	－
@LAST_RESPONSE_STATUS	VT_ARRAY | VT_I4	直前のエラー応答詳細情報を返す. (VT_ARRAY | VT_I4) [0] 一般ステータス [1] 追加ステータス(ない場合は 0 セット)	○	－

5.2. CaoExtension クラス

5.2.1. システム変数

表 5-2 CaoExtension クラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@ErrStatus	VT_I4	センサアンプエラー状態 エラーが発生している場合, 対応するビットが ON(1)となります. (初期値:---) bit0 : 過電流エラー(ErC) bit1 : EEPROM エラー(ErE) bit2 : ヘッドエラー(ErH) bit7 : 投光器レーザエラー (ErH LASEr) bit8 : 型式不一致エラー (ErH tyPE) bit11 : アンプ間通信エラー (Er.Com) bit12 : ユニット台数エラー	○	－

		(Er.Unt) bit13 : 演算エラー(Er.CAL) 上記以外 : 0 固定		
@ControlOutput	VT_I4	センサの出力状態 出力が ON となっている場合, 対応するビットが ON(1)となります. (初期値:---) bit0 : HIGH 判定出力 bit1 : LOW 判定出力 bit2 : GO 判定出力 bit3 : アラーム出力	○	－
@CriteriaValue	VT_R8	判定値(P.V.値) ([3]) -99.999 ～ +99.999(初期値:---)	○	－
@RawValue	VT_R8	内部判定値(R.V.値) ([3]) -99.999 ～ +99.999(初期値:---)	○	－
@PeakValue	VT_R8	ホールド時のピークホールド値 ([3]) ・ サンプルホールド以外のとき サンプリング期間中のピークホールド値 -99.999 ～ +99.999(初期値:---) ・ サンプルホールドのとき -99.998	○	－
@BottomValue	VT_R8	ホールド時のボトムホールド値 ([3]) ・ サンプルホールド以外のとき サンプリング期間中のピークホールド値 -99.999 ～ +99.999(初期値:---) ・ サンプルホールドのとき -99.998	○	－
@CalcValue	VT_R8	演算値(CALC 値) ([3]) -99.999 ～ +99.999	○	－
@AnalogValue	VT_R8	現在のアナログ出力値 (親機のみ) (初期値:---) 電圧 : -5.000 ～ +5.000	○	－

[3] IL-300/IL-600 使用時, -999.99 ~ +999.99, IL-2000 使用時, -9999.9 ~ +9999.9 になります.

		(エラー時 : +5.500) 4 – 20mA : +4.00 ～ +20.00 (エラー時 : +3.00) OFF : 0 固定		
@SettingError	VT_I4	設定異常状態 0 ～ 1(初期値:---) 0 : 設定正常 1 : 設定異常	○	－
@ExternalInputState	VT_I4	外部入力状態 センサアンプの外部入力線またはサイクリック通信の外部入力が ON の場合, 対応するビットが ON(1)となります. (初期値:---) bit0 : 外部入力 1 bit1 : 外部入力 2 bit2 : 外部入力 3 bit3 : 外部入力 4	○	－
@EEPROMWriteResult	VT_I4	EEPROM 書き込み結果 0 ～ 2(初期値:---) 0 : 書き込み中 1 : 正常終了 2 : 異常終了	○	－
@ZeroShiftResult	VT_I4	ゼロシフト・ゼロシフトリセットの実行結果 0 ～ 2(初期値:---) 0 : 実行中 1 : 正常終了 2 : 異常終了(SHiFt Err)	○	－
@ResetResult	VT_I4	リセットの実行結果 0 ～ 2(初期値:---) 0 : 実行中 1 : 正常終了 2 : 異常終了	○	－
@TuningResult	VT_I4	チューニングの実行結果 0 ～ 2(初期値:---) 0 : 実行中 1 : 正常終了	○	－

		2 : 異常終了		
@CalibrationResult	VT_I4	キャリブレーションの実行結果 0 ～ 2(初期値:---) 0 : 実行中 1 : 正常終了 2 : 異常終了	○	－
@Bank_HIGH	VT_R8	HIGH 側設定値(BANK0 ～ 3) ([3]) -99.999 ～ +99.999(初期値:+5.000) ※ バンク No は ID プロパティにて指定 (ID=0 ～ 3)	○	○
@Bank_LOW	VT_R8	LOW 側設定値(BANK0 ～ 3) ([3]) -99.999 ～ +99.999(初期値:-5.000) ※ バンク No は ID プロパティにて指定 (ID=0 ～ 3)	○	○
@Bank_ShiftTarget	VT_R8	シフト目標値(BANK0 ～ 3) ([3]) -99.999 ～ +99.999(初期値:0) ※ バンク No は ID プロパティにて指定 (ID=0 ～ 3)	○	○
@Bank_AnalogOutputUpperLimit	VT_R8	アナログ出力 上限値(BANK0 ～ 3) ([3]) (親機のみ) -99.999 ～ +99.999(初期値:+10.000) ※ バンク No は ID プロパティにて指定 (ID=0 ～ 3)	○	○
@Bank_AnalogOutputLowerLimit	VT_R8	アナログ出力 下限値(BANK0 ～ 3) ([3]) (親機のみ) -99.999 ～ +99.999(初期値:-10.000) ※ バンク No は ID プロパティにて指定 (ID=0 ～ 3)	○	○
@KeyLock	VT_I4	キーロック機能 0 ～ 1(初期値:0) 0 : アンロック 1 : キーロック	○	○
@Bank	VT_I4	バンク機能 Get 時:動作しているバンク番号 0 ～ 3(初期値:0) Put 時:バンク番号を変更	○	○

		0 : バンク 0 1 : バンク 1 2 : バンク 2 3 : バンク 3		
@Timing	VT_I4	Get 時:動作しているタイミング状態 0 ～ 1(初期値:0) 0 : サンプリング中 1 : 非サンプリング中 Put 時:タイミング入力を設定 0 : タイミング入力 OFF 1 : タイミング入力 ON	○	○
@ProjectionStop	VT_I4	Get 時:動作している投光停止状態 0 ～ 1(初期値:0) 0 : 投光中 1 : 投光停止中(投光停止入力 ON/レーザエラー/ヘッドエラー) Put 時:レーザ投光停止入力状態を設定 0 : 投光停止入力 OFF 1 : 投光停止入力 ON	○	○
@SubDisplay	VT_I4	サブ表示部の表示画面 0 ～ 5(初期値:0) 0 : R.V.値 1 : アナログ値 2 : HI 設定値 3 : LO 設定値 4 : ゼロシフト値 5 : CALC 値	○	○
@SystemParameters	VT_I4	Get 時:システムパラメータの現在状態 現在状態に応じて対応するビットが ON(1)となります. (初期値:---) Put 時: 設定の反映は, 設定書き込み後, 「SystemParametersSet」コマンド の実行が必要です.	○	○

		bit0 : 0 : NPN 1 : PNP bit1,2,3 : (親機のみ, 子機は 000 固定) 000 : アナログ出力 OFF 001 : 0 ～ +5V 010 : -5 ～ +5V 011 : +1 ～ +5V 100 : 4 ～ 20mA		
@ToleranceSettingWidth	VT_R8	公差チューニングの公差の設定幅 ([3]) 0.000 ～ 99.999(初期値:0.200)	○	○
@CAL	VT_I4	キャリブレーション機能 0 ～ 1(初期値:0) 0 : 初期状態 1 : ユーザ設定	○	○
@CALSET1	VT_R8	キャリブレーションの 1 点目の目標値 ([3]) -99.999 ～ +99.999(初期値:0.000)	○	○
@CALSET2	VT_R8	キャリブレーションの 2 点目の目標値 ([3]) -99.999 ～ +99.999(初期値:+5.000)	○	○
@CALCVCAL	VT_I4	演算値キャリブレーション機能 (親機のみ) 0 ～ 2(初期値:0) 0 : 初期状態 1 : 演算 2 点キャリブレーション 2 : 演算 3 点キャリブレーション	○	○
@CACLV2PTCALSET1	VT_R8	演算値 2 点キャリブレーションの 1 点目の目標値 ([3]) (親機のみ) -99.999 ～ +99.999(初期値:+5.000)	○	○
@CALCV2PTCALSET2	VT_R8	演算値 2 点キャリブレーションの 2 点目の目標値 ([3]) (親機のみ) -99.999 ～ +99.999(初期値:+10.000)	○	○
@CALCV3PTCALSET1	VT_R8	演算値 3 点キャリブレーションの 1 点目の目標値 ([3])	○	○

		(親機のみ) -99.999 ～ +99.999(初期値:+5.000)		
@CALCV3PTCALSET3	VT_R8	演算値 3 点キャリブレーションの 3 点目の目標値 (^{3}) (親機のみ) -99.999 ～ +99.999(初期値:+10.000)	○	○
@CalcMode	VT_I4	演算機能 (親機のみ) 0 ～ 2(初期値:0) 0 : OFF 1 : 加算 2 : 減算	○	○
@MeasureDirect	VT_I4	測定方向 0 ～ 1(初期値:0) 0 : 通常 1 : 反転	○	○
@SamplingCycle	VT_I4	サンプリング周期 0 ～ 4(初期値:0) 0 : デフォルト 1 : 0.33ms 2 : 1ms 3 : 2ms 4 : 5ms	○	○
@AverageTimes	VT_I4	平均回数・段差カウントフィルタ・ハイパスフィルタ 0 ～ 14(初期値:4) 0 : 1 回 1 : 2 回 2 : 4 回 3 : 8 回 4 : 16 回 5 : 32 回 6 : 64 回 7 : 128 回 8 : 256 回 9 : 512 回	○	○

		10 : 1024 回 11 : 2048 回 12 : 4096 回 13 : 段差カウントフィルタ 14 : ハイパスフィルタ		
@Output	VT_I4	出力様式 0 ～ 1(初期値:0) 0 : ノーマルオープン 1 : ノーマルクローズ	○	○
@Hold	VT_I4	ホールド機能の設定 0 ～ 5(初期値:0) 0 : サンプルホールド 1 : ピークホールド 2 : ボトムホールド 3 : ピーク to ピークホールド 4 : オートピークホールド 5 : オートボトムホールド	○	○
@HoldSettings	VT_R8	オートピークホールドまたはオートボトムホールド トリガレベル ([3]) -99.999 ～ +99.999(初期値:+1.000)	○	○
@TimingInput	VT_I4	タイミング入力設定 0 ～ 1(初期値:0) 0 : レベル 1 : エッジ	○	○
@DelayTimerSetting	VT_I4	ディレイタイマ 0 ～ 3(初期値:0) 0 : OFF 1 : オンディレイ 2 : オフディレイ 3 : ワンショット	○	○
@TimerTime	VT_I4	タイマ時間(単位:ms) 5 ～ 9999(初期値:60)	○	○
@Hysteresis	VT_R8	ヒステリシス ([3]) 0.000 ～ 99.999(初期値:0.000)	○	○
@AnalogSetting	VT_I4	アナログ出力スケーリング (親機のみ)	○	○

		0 ～ 2(初期値:0) 0 : 初期状態 1 : フリーレンジ 2 : バンク		
@AnalogOutput_Hi	VT_R8	アナログ出力 上限値 (^{3}) (親機のみ) -99.999 ～ +99.999(初期値:+10.000)	○	○
@AnalogOutput_Lo	VT_R8	アナログ出力 下限値 (^{3}) (親機のみ) -99.999 ～ +99.999(初期値:-10.000)	○	○
@ExternalInput	VT_I4	外部入力 1 ～ 4 の機能割り当て 0 ～ 1(初期値:0) 0 : 初期状態 1 : ユーザ設定	○	○
@ExternalInput1	VT_I4	外部入力 1 に割り当てる機能 0 ～ 4(初期値:0) 0 : ゼロシフト入力 1 : バンク A 入力 2 : バンク B 入力 3 : レーザ投光停止入力 4 : 使用しない	○	○
@ExternalInput2	VT_I4	外部入力 2 に割り当てる機能 0 ～ 4(初期値:0) 0 : リセット入力 1 : バンク A 入力 2 : バンク B 入力 3 : レーザ投光停止入力 4 : 使用しない	○	○
@ExternalInput3	VT_I4	外部入力 3 に割り当てる機能 0 ～ 4(初期値:0) 0 : タイミング入力 1 : バンク A 入力 2 : バンク B 入力 3 : レーザ投光停止入力 4 : 使用しない	○	○
@ExternalInput4	VT_I4	外部入力 4 に割り当てる機能	○	○

		0 ～ 3(初期値:0) 0 : ゼロシフト入力 1 : バンク A 入力 2 : バンク B 入力 3 : レーザ投光停止入力		
@BankSwitching	VT_I4	バンク切り替え方法 0 ～ 1(初期値:0) 0 : ボタン 1 : 外部入力	○	○
@ZeroShiftMem	VT_I4	ゼロシフト値記憶機能 0 ～ 1(初期値:0) 0 : OFF 1 : ON	○	○
@InterferencePrevention	VT_I4	相互干渉防止機能 0 ～ 1(初期値:0) 0 : 干渉防止 OFF 1 : 干渉防止 ON	○	○
@DisplayDigit	VT_I4	表示桁数 (初期値:0) 0 : 初期状態 2 : 0.001 3 : 0.01 4 : 0.1 5 : 1	○	○
@EcoMode	VT_I4	省電力モード 0 ～ 2(初期値:0) 0 : OFF 1 : ハーフ 2 : オール	○	○
@HeadDisp	VT_I4	ヘッド表示モード 0 ～ 2(初期値:0) 0 : 初期状態 1 : OK/NG 表示 2 : OFF	○	○
@DisplayColor	VT_I4	アンプの表示色 0 ～ 2(初期値:0)	○	○

		0 : GO 緑色 1 : GO 赤色 2 : 常時 赤色		
@OneShotTime	VT_I4	段差カウントフィルタのワンショット出力 ON 時間(単位:ms) 2 ～ 9999(初期値:10)	○	○
@CutoffFrequency	VT_I4	ハイパスフィルタのカットオフ周波数 0 ～ 9(初期値:0) 0 : 0.1Hz 1 : 0.2Hz 2 : 0.5Hz 3 : 1Hz 4 : 2Hz 5 : 5Hz 6 : 10Hz 7 : 20Hz 8 : 50Hz 9 : 100Hz	○	○
@AlarmSetting	VT_I4	アラーム設定 0 ～ 2(初期値:0) 0 : 初期状態 1 : クランプ 2 : ユーザ設定	○	○
@NumberOfAlarms	VT_I4	アラーム回数 2 ～ 1000(初期値:7)	○	○
@LAST_RESPONSE_STATUS	VT_ARRAY | VT_I4	直前のエラー応答詳細情報を返す. (VT_ARRAY | VT_I4) [0] 一般ステータス [1] 追加ステータス(ない場合は 0 セット)	○	－

6. エラーコード

ILDLEP1 プロバイダでは，以下の固有のエラーコードが定義されています．
ORiN2 共通エラーについては，「ORiN2 プログラミングガイド」のエラーコードの章を参照して下さい．

表 6-1 固有エラーコード

エラー名	エラー番号	説明
CIP 応答異常 (フォーマットエラー)	0x80100000	CIP 通信の応答パケットが想定外の異常なフォーマットであった場合に返ります．
CIP 応答異常 (Error Status Code)	0x80100001	CIP 通信の応答パケットがステータス異常であった場合に返ります．
CIP 応答異常 (Error General Status)	0x80100002	CIP 通信の応答パケットがエラー応答であった場合に返ります． 本エラーを受信した場合は，システム変数の @LAST_RESPONSE_STATUS を読み出すことで詳細エラーコードを得ることができます．

表 6-2 詳細エラーコード

一般ステータス	追加ステータス	説明
0x00	0x0000	正常終了．
0x05	0x0000	インスタンス ID が範囲外です．
0x08	0x0000	指定したインスタンス ID ではこのサービスは非対応であるか，実行できません．
0x09	0x0000	書き込まれたデータが範囲外です．
0x0C	0x0000	・ サイクリック通信で制御している機能にパラメータ書き込みを実行しました． ・ 動作指令の実行に失敗しました．センサアンプが，動作指令が実行できる状態か確認してください．
0x0E	0x0000	書き込みできないアトリビュート ID，または書き込みできない状態のアトリビュート ID に書き込もうとしています．
0x10	0x0000	読み出しできないアトリビュート ID，または読み出しできない状態のアトリビュート ID から読み出そうとしています．
0x13	0x0000	サービスデータのサイズが規定以下です． 追加ステータスには，規定されたデータサイズが格納されま

		す.
0x14	0x0000	アトリビュート ID が範囲外です.
0x16	0x0000	指定されたインスタンス ID に対応するセンサアンプが接続されていません.
0x1F	0xC350	指定されたインスタンス ID でこのサービスはサポートされていますが, 指定されたアトリビュート ID では使用できません.
	0xC351	現在のモードでは指定した設定値をパラメータに対して書き込みできません.
	0xC352	本機が通信初期化中です.
0xFE	0x0000	システムエラー.

7. サンプルプログラム

以下にサンプルコード(C#)を示します.

7.1. 接続とオブジェクトの生成

```
// 通信ユニットに接続し，アンプユニットやセンサヘッドにアクセスするオブジェクトを生成
private void Connect()
{
    try
    {
        // CAO エンジンの生成
        m_CaoEngine = new CCaoEngine();
        // ワークスペース取得
        m_CaoWorkspace = m_CaoEngine.Workspaces[0];
        // 通信ユニットへの接続
        m_CaoController = m_CaoWorkspace.AddController("Sample",
            "CaoProv.KEYENCE.ILDLEP1", "", "Conn=TCP:192.168.0.20");
        // アンプユニットオブジェクト生成
        m_CaoExtension = m_CaoController.AddExtension("CH1", "");
        // 変数オブジェクト生成
        m_BankHigh = m_CaoExtension.AddVariable("@Bank_HIGH", "");
        m_CriteriaValue = m_CaoExtension.AddVariable("@CriteriaValue", "");
    }
    catch (Exception e)
    {
        MessageBox.Show(this, e.Message);
    }
}
```

7.2. センサ ID=1 のリセット

```
// センサ ID=1 をリセット
private void Reset()
{
    try
    {
        // リセット実行要求
        m_CaoExtension.Execute("Reset");
    }
    catch (Exception e)
    {
        MessageBox.Show(this, e.Message);
    }
}
```

7.3. センサ ID=1 のバンク 2 HIGH 側設定値の設定

```
// センサ ID=1 のバンク 2 HIGH 側設定値の設定
private void PutBank_HIGH()
{
    try
    {
        // バンク No の設定
        m_BankHigh.ID = 2;
        // 値の設定
        Double val;
        if (Double.TryParse(txtPutBank_HIGH.Text, out val))
        {
            m_BankHigh.Value = val;
        }
    }
    catch (Exception e)
    {
        MessageBox.Show(this, e.Message);
    }
}
```

7.4. センサ ID=1 の判定値(P.V.値)の取得

```
// センサ ID=1 の判定値(P.V.値)を取得
private void GetCriteriaValue()
{
    try
    {
        // 値の取得
        Object val = m_CriteriaValue.Value;
        // 結果の反映
        txtGetCriteriaValue.Text = val.ToString();
    }
    catch (Exception e)
    {
        MessageBox.Show(this, e.Message);
    }
}
```

7.5. センサ ID=1 の判定値(P.V.値)の取得 (Raw コマンド)

```
// センサ ID=1 の判定値(P.V.値)を Raw コマンドを使用して取得
private void ExecuteRawGetCriteriaValue()
{
    try
    {
        txtRawResult.Text = "";
```

```
            // コマンドデータ作成
            Byte[] param = new Byte[5];
            param[0] = 0x0E;    // サービスコード          : パラメータ読み出し
            param[1] = 0x67;    // クラス ID               : DL Object クラス
            param[2] = 0x01;    // インスタンス ID         : センサ ID=1
            param[3] = 0x25;    // アトリビュート ID(Low)  : 判定値(P.V.値)(0x0325)
            param[4] = 0x03;    // アトリビュート ID(High) : ↑
            // Raw コマンド実行
            Object result = m_CaoController.Execute("Raw", param);
            // 結果反映
            String strResult = String.Empty;
            foreach (Byte b in (Byte[])result)
            {
                strResult += " ";
                strResult += b.ToString();
            }
            txtRawResult.Text = strResult;
        }
        catch (Exception e)
        {
            MessageBox.Show(this, e.Message);
        }
    }
```

7.6. 通信ユニットのステータスの取得（Raw コマンド）

```
    // 通信ユニットのステータスを Raw コマンドを使用して取得
    private void ExecuteRawGetStatus()
    {
        try
        {
            txtRawResult.Text = "";
            // コマンドデータ作成
            Byte[] param = new Byte[5];
            param[0] = 0x0E;    // サービスコード          : パラメータ読み出し
            param[1] = 0x67;    // クラス ID               : DL Object クラス
            param[2] = 0x00;    // インスタンス ID         : 通信ユニット=0
            param[3] = 0x64;    // アトリビュート ID(Low)  : ステータス(0x0064)
            param[4] = 0x00;    // アトリビュート ID(High) : ↑
            // Raw コマンド実行
            Object result = m_CaoController.Execute("Raw", param);
            // 結果反映
            String strResult = String.Empty;
            foreach (Byte b in (Byte[])result)
            {
                strResult += " ";
                strResult += b.ToString();
            }
            txtRawResult.Text = strResult;
        }
        catch (Exception e)
        {
```

```
                MessageBox.Show(this, e.Message);
            }
        }
```