KEYENCE
LK-G3000 プロバイダ
ユーザーズ ガイド

Version 1.0.1

December 21, 2021

備考：

【改版履歴】

バージョン	日付	内容
1.0.0	2020-04-27	初版.
1.0.1	2021-12-21	エラー誤検知を修正しました.

【動作確認機種】

機種	バージョン	注意事項
LK-G3000	--	必ず専用コード OP-96368（ストレートコード 2.5m）と OP-26401（D-sub 9 ピン）， または，OP- 96369（D-sub 25 ピン）を組み合わせて PC と機器と接続すること.

【対応機種】

機種
LK-G3000
LK-G3000P
LK-G3000V
LK-G3000PV

目次

1. はじめに 8
1.1. 参考となる情報源 8
2. アプリケーション開発のための環境セットアップ 10
2.1. LK-G3000 とクライアント PC との接続 10
3. コマンドリファレンス 11
3.1. メソッド/プロパティー覧 11
3.2. メソッド・プロパティ 11
3.2.1. CaoWorkspace クラス 11
3.2.1.1. AddController メソッド 11
3.2.2. CaoController クラス 13
3.2.2.1. VariableNames メソッド 13
3.2.2.2. Variables プロパティ 13
3.2.2.3. AddVariable メソッド 13
3.2.2.4. Execute メソッド 13
3.2.3. CaoVariable クラス 14
3.2.3.1. Value プロパティ 14
3.3. 拡張コマンドー覧 14
3.3.1. モード変更コマンド 16
3.3.1.1. SetMode コマンド 16
3.3.2. 測定・制御関連コマンド 17
3.3.2.1. GetCalcData コマンド 17
3.3.2.2. SetTiming コマンド 18
3.3.2.3. SetZero コマンド 19
3.3.2.4. SetReset コマンド 19
3.3.2.5. SetPanelLock コマンド 19
3.3.2.6. SetProgramNo コマンド 20
3.3.2.7. GetProgramNo コマンド 20
3.3.2.8. GetFigureData コマンド 20
3.3.2.9. ClearFigureData コマンド 22
3.3.2.10. StartDataStorage コマンド 23

3.3.2.11. StopDataStorage コマンド....23
3.3.2.12. ClearDataStorage コマンド....23
3.3.2.13. GetDataStorageData コマンド....23
3.3.2.14. GetDataStorageStatus コマンド....23
3.3.3. 設定内容関連コマンド....24
3.3.3.1. パネル表示関連コマンド....24
3.3.3.2. 公差設定関連コマンド....25
3.3.3.3. ヘッド設定関連コマンド....26
3.3.3.4. OUT 設定関連コマンド....34
3.3.3.5. 共通設定関連コマンド....48
3.4. 変数一覧....53
3.4.1. CaoController クラス変数....54
3.4.1.1. @MAKER_NAME....54
3.4.1.2. @VERSION....54
3.4.1.3. @CALCDATA....54
4. LK-G3000 プロバイダによるプログラミング....56
4.1. OUT1 と OUT2 の測定値を取得するサンプルプログラミング....56
4.1.1. サンプルプログラム....56
4.1.1.1. 接続....58
4.1.1.2. 本体の動作モード変更....59
4.1.1.3. OUT1 と OUT2 の測定値取得....59
4.1.1.4. 切断....60
4.2. データストレージの蓄積データを取得するサンプルプログラミング....60
4.2.1. サンプルプログラム....61
4.2.1.1. 接続....63
4.2.1.2. データストレージの設定....64
4.2.1.3. データストレージへのデータ蓄積開始....64
4.2.1.4. データストレージへのデータ蓄積終了....65
4.2.1.5. データストレージの蓄積状態を取得....65
4.2.1.6. データストレージの蓄積データを取得....65
4.2.1.7. データストレージの蓄積データをクリア....66
4.2.1.8. 切断....66
5. LK-G3000 プロバイダエラーコード....67
6. 付録....68

1. はじめに

本書は，KEYENCE 社製レーザー変位計 LK-G3000 シリーズからデータを取得するプロバイダのユーザーズガイドです．図 1-1 が本プロバイダとデバイスの全体構成図になります．以降本プロバイダを LK-G3000 プロバイダと呼称します．KEYENCE 社製レーザー変位計 LK-G3000 シリーズを LK-G3000 と呼称します．LK-G3000 プロバイダは LK-G3000 と，RS-232C を使用しシリアル通信を行います．

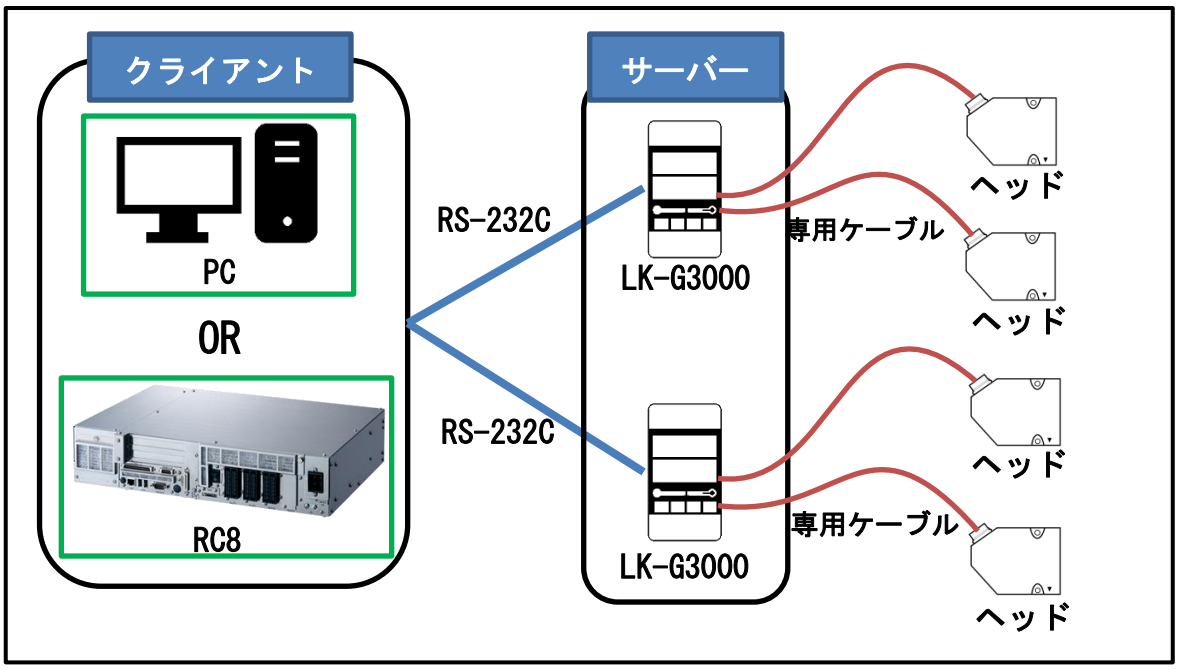


図 1-1 構成図

また，本プロバイダ及びデバイスそれぞれの対応を図 1-2に表します．
（※一例です．全てを表しているわけではありません．）

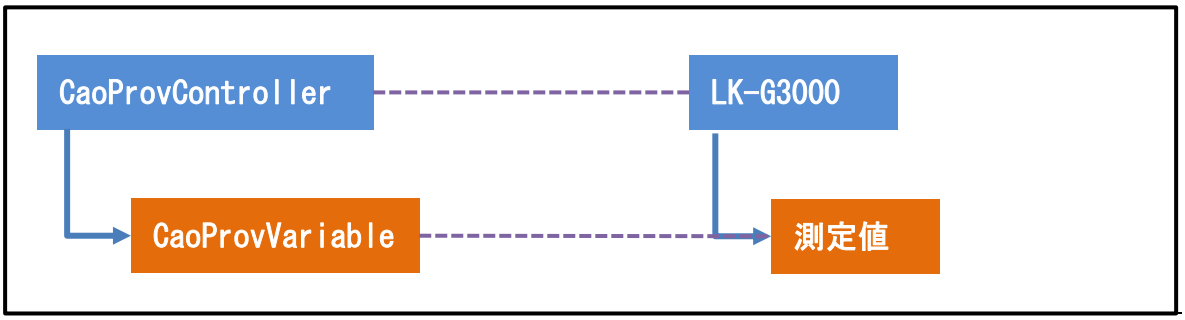


図 1-2 プロバイダの構成とデバイス情報との対応図

1.1. 参考となる情報源

LK-G3000LkIF プロバイダは，KEYENCE 社の「高速・高精度CCD レーザー変位計LK-G シリーズユーザ

ーズマニュアル 96M12274」を参考に開発しております. 以降このマニュアルを, LK-G3000 マニュアルと呼称します.

2. アプリケーション開発のための環境セットアップ

2.1. LK-G3000 とクライアント PC との接続

LK-G3000 プロバイダは LK-G3000 と RS-232C 通信を利用して接続を行います. クライアント PC と接続している専用ケーブル（OP-96368(ストレートコード 2.5m)と OP-26401(D-sub 9 ピン), または, OP- 96369(D-sub 25 ピン)）を LK-G3000 にも接続します. 図 2-1 のように赤枠内の差込口に対応の専用ケーブルを, カチッと音が鳴るまで差し込んでください.

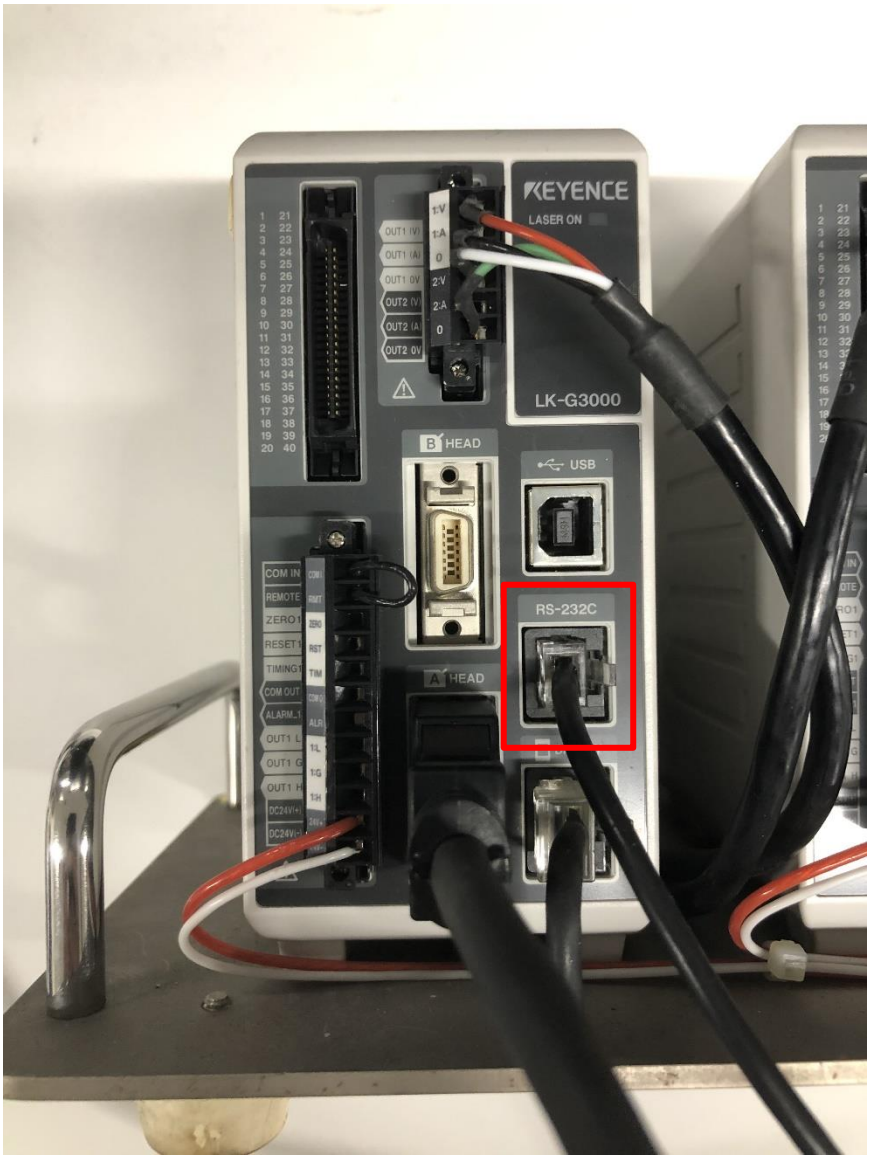


図 2-1 実際の LK-G3000 デバイス

3. コマンドリファレンス

3.1. メソッド/プロパティー覧

表 3-1 メソッド/プロパティー覧

カテゴリ	メソッド/プロパティ[1]		機能	参照
CaoWorkspace				
	AddController	M	コントローラに接続	P.11
CaoController				
	VariableNames	M	接続可能な変数名リストの取得	P.13
	Variables	P	コントローラが保持する変数コレクションの取得	P.13
	AddVariable	M	変数オブジェクトの追加	P.13
	Execute	M	拡張コマンドの実行	P.13
CaoVariable				
	Value	P	値の取得/設定	P.14

3.2. メソッド・プロパティ

3.2.1. CaoWorkspace クラス

3.2.1.1. AddController メソッド

CaoWorkspace に，コントローラオブジェクトを追加します．LK-G3000 プロバイダでは，AddController メソッド実行時に渡されたパラメータを参照し，該当する LK-G3000 と接続を行います．以下に，AddController メソッドの仕様を示します．

書式

```
AddController
(
        "<コントローラ名>",              // コントローラ名(任意)
        "CaoProv.KEYENCE.LK-G3000",     // プロバイダ名 (固定)
        "<マシン名>",                    // プロバイダ実行マシン名(未使用)
        "<オプション>"                   // オプション文字列(必須)
)
```

オプション

[1] M:メソッド，P:プロパティ，E:イベントをそれぞれ示します．

以下にオプション文字列に指定するオプションを示します．オプション文字列は下記に示す各オプションをカンマ(,)でつなげた文字列となります．

オプション	必須	説明	値範囲	デフォルト値
Conn=	○	LK-G3000 と接続するためのシリアル接続オプションを指定します．詳細は 3.2.1.1.1 を参照してください．	--	--
Timeout=	--	コマンド送信から LK-G3000 からデータ受信するまでの応答待機時間をミリ秒単位で指定します．	1 - 4294967265	500

使用例

```
Dim caoEng As CaoEngine          ' Engineオブジェクト
Dim caoWs As CaoWorkspace        ' Workspaceオブジェクト
Dim caoCtrl As CaoController     ' Controllerオブジェクト

' CaoEngine オブジェクトの生成
Set caoEng  = new CaoEngine
' CaoWorkspace オブジェクトの生成
Set caoWs   = caoEng.Workspaces.Item(0)
' CaoController オブジェクトの生成
Set caoCtrl = caoWs.AddController("LKG3000", _
                                  "CaoProv.KEYENCE.LK-G3000", _
                                  "", _
                                  "Conn=COM:1,Timeout=1000")
```

3.2.1.1.1. Conn オプション

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．ここで角括弧("[]")内は省略可能なことを，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をそれぞれ示します．

RS-232C

"Conn=COM:<COM Port>[:BaudRate>[:<Parity>:<DataBits>:<StopBits>[:Flow]]]"

<COM Port> ： COM ポート番号． '1' -COM1, '2' - COM2, ...

<BaudRate> ： 通信速度． 9600, 19200, 38400, 57600, 115200

<Parity> ： パリティ．'N'-NONE, 'E'-EVEN, 'O'-ODD

<DataBits> ： データビット数．'7'-7bit, '8'-8bit

<StopBits> ： ストップビット数．'1'-1bit, '2'-2bit

<Flow> ： フロー制御．'0'-None, '1'-Xon/Xoff, '2'-ハードウェア制御 OR をとって指定できます．

3.2.2. CaoController クラス

3.2.2.1. VariableNames メソッド

接続可能な変数名リストを取得します．本メソッドで取得した変数名は，後述する AddVariable メソッドの第一引数に使用することができます．

使用例

```
' 変数名リスト取得
Dim variableNames As Variant
variableNames = caoCtrl.variableNames
```

3.2.2.2. Variables プロパティ

コントローラが保持する，変数コレクションを取得します．

使用例

```
' 変数コレクション取得
Dim variables As CaoVariables
Set variables = caoCtrl.variables
' 変数取得
Dim variable As CaoVariable
Set variable = variables.Item(0)
```

3.2.2.3. AddVariable メソッド

CaoController に変数オブジェクトを追加します．変数名には 3.4.変数一覧に示すもののみ使用できます．

以下に，AddVariable の仕様を示します．

書式

```
AddVariable
(
        "<変数名>",             // 変数名
        "<オプション>"          // オプション文字列(省略可能)
)
```

3.2.2.4. Execute メソッド

CaoController の拡張コマンドを実行します．Execute で指定できる拡張コマンドについては 3.3.拡張コマンド一覧に示すもののみ使用可能です．以下に，Execute の仕様を示します．

書式

```
Execute
(
        "<拡張コマンド名>",             // 拡張コマンド名
```

```
        "<オプション文字列>"                // オプション文字列(省略可能)
)
```

3.2.3. CaoVariable クラス

3.2.3.1. Value プロパティ

接続した LK-G3000 からデータを取得/設定します. 変数名によって動作が異なります. 詳細は, 3.4. 変数一覧を参照してください.

3.3. 拡張コマンド一覧

使用可能な拡張コマンド一覧を定義します. 使用例は各コマンドの詳細で記述しています.

表 3-2 拡張コマンド一覧

コマンド	説明	参照
モード変更コマンド		
SetMode	本体の動作モードを設定します.	P.16
測定・制御関連コマンド		
GetCalcData	測定値を取得します.	P.17
SetTiming	タイミングの ON/OFF を設定します.	P.18
SetZero	オートゼロの ON/OFF を設定します.	P.19
SetReset	リセットを設定します.	P.19
SetPanelLock	パネルロックを設定します.	P.19
SetProgramNo	プログラム番号を切り替えます.	P.20
GetProgramNo	プログラム番号を取得します.	P.20
GetFigureData	統計結果を取得します.	P.20
ClearFigureData	統計値をクリアします.	P.22
StartDataStorage	データストレージの蓄積を開始します.	P.23
StopDataStorage	データストレージの蓄積を停止します.	P.23
ClearDataStorage	データストレージの蓄積データをクリアします.	P.23
GetDataStorageData	データストレージの蓄積データを取得します.	P.23
GetDataStorageStatus	データストレージの蓄積状態を取得します.	P.23
設定内容関連コマンド		
パネル表示関連コマンド		
SetPanel	パネル表示を切り替えます.	P.24

コマンド		説明	参照
	GetPanel	パネル表示を取得します.	P.24
公差設定関連コマンド			
	SetTolerance	公差を設定します.	P.25
	GetTolerance	公差を取得します.	P.25
ヘッド設定関連コマンド			
	SetAbleMode	ABLE チューニングモードを設定します.	P.26
	GetAbleMode	ABLE チューニングモードを取得します.	P.27
	SetAbleMinMax	ABLE 制御範囲を設定します.	P.27
	GetAbleMinMax	ABLE 制御範囲を取得します.	P.28
	SetMeasureMode	測定モードを設定します.	P.29
	GetMeasureMode	測定モードを取得します.	P.29
	SetNumAlarm	アラーム処理回数を設定します.	P.30
	SetAlarmLevel	アラームレベルを設定します.	P.30
	GetAlarm	アラーム情報を取得します.	P.31
	StartABLE	ABLE チューニングを開始します.	P.31
	StopABLE	ABLE チューニングを終了します.	P.32
	CancelABLE	ABLE チューニングを中止します.	P.32
	SetReflectionMode	設置モードを設定します.	P.33
	GetReflectionMode	設置モードを取得します.	P.33
OUT 設定関連コマンド			
	SetCalcMethod	演算方法を設定します.	P.34
	GetCalcMethod	演算方法を取得します.	P.37
	SetScaling	スケーリングを設定します.	P.38
	GetScaling	スケーリングを取得します.	P.39
	SetFilter	フィルタを設定します.	P.39
	GetFilter	フィルタを取得します.	P.41
	SetTriggerMode	トリガモードを設定します.	P.41
	GetTriggerMode	トリガモードを取得します.	P.42
	SetOffset	オフセットを設定します.	P.43
	GetOffset	オフセットを取得します.	P.43

コマンド		説明	参照
	SetAnalogScaling	アナログ出力スケーリングを設定します.	P.44
	GetAnalogScaling	アナログ出力スケーリングを取得します.	P.44
	SetCalcMode	計測モードを設定します.	P.45
	GetCalcMode	計測モードを取得します.	P.46
	SetDisplayUnit	最小表示単位を設定します.	P.46
	GetDisplayUnit	最小表示単位を取得します.	P.47
	SetAnalogThrough	アナログスルーを設定します.	P.47
	GetAnalogThrough	アナログスルーを取得します.	P.48
共通設定関連コマンド			
	SetDataStorage	データストレージの対象 OUT, 蓄積点数, 蓄積周期を設定します.	P.48
	GetDataStorage	データストレージの対象 OUT, 蓄積点数, 蓄積周期を取得します.	P.49
	SetSamplingCycle	サンプリング周期を設定します.	P.50
	GetSamplingCycle	サンプリング周期を取得します.	P.50
	SetMutualInterPrev	相互干渉防止を設定します.	P.50
	GetMutualInterPrev	相互干渉防止を取得します.	P.51
	SetTimingSync	タイミング同期を設定します.	P.51
	GetTimingSync	タイミング同期を取得します.	P.52
	SetToleCompOutputFormat	判定出力形態を設定します.	P.52
	GetToleCompOutputFormat	判定出力形態を取得します.	P.52
	SetStorobeTime	ストローブ時間を設定します.	P.53
	GetStorobeTime	ストローブ時間を取得します.	P.53

3.3.1. モード変更コマンド

3.3.1.1. SetMode コマンド

LK-G3000 本体の動作モードを変更します. なお, 3.3.2.測定・制御関連コマンドを実行する際には LK-G3000 本体の動作モードを「通常モード」にし, 3.3.3.設定内容関連コマンドを実行する際には LK-G3000 本体の動作モードを「通信モード」に切り替える必要があります. 以下に引数を示します.

項目	型説明

項目	型説明	
引数	VT_I4	動作モードを指定します．以下のいずれかを指定します． ・0 － 通常モード ・1 － 通信モード

使用例

```
' SetMode実行
Call caoCtrl.Execute("SetMode", 0)
```

3.3.2. 測定・制御関連コマンド

3.3.2.1. GetCalcData コマンド

測定値を取得します．以下に引数と戻り値を示します．また，戻り値は指定した OUT 番号によって変化します．

項目	型説明	
引数	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・0 － OUT1 + OUT2 ・1 － OUT1 ・2 － OUT2

・引数に OUT 番号「0」を指定した場合の戻り値：

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_R4 or VT_EMPTY	OUT1 の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
	1	VT_R4 or VT_EMPTY	OUT2 の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．

・引数に OUT 番号「1」又は「2」を指定した場合の戻り値：

項目	型説明	
戻り値	VT_R4 or VT_EMPTY	指定した OUT 番号の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．

表 3-3 測定値の詳細

データ型	取得値	測定値の状態
VT_R4	正常値	正常
VT_R4	1.#INF	+レンジオーバー
VT_R4	-1.#INF	-レンジオーバー
VT_EMPTY	--	判定待機状態

使用例

```
' GetCalcData実行
Dim values As Variant
values = caoCtrl.Execute("GetCalcData", 0)

If Not IsEmpty(values) Then
    If Not IsEmpty(values(0)) Then
        ' OUT1の測定値
        Dim value1 As Single
        value1 = values(0)
    End If
    If Not IsEmpty(values(1)) Then
        ' OUT2の測定値
        Dim value2 As Single
        value2 = values(1)
    End If
End If
```

3.3.2.2. SetTiming コマンド

タイミング入力を設定します. タイミング入力を設定すると設定時の測定値を保持します. 計測モードによってタイミング入力時の機能が異なります. タイミング入力の詳細は, LK-G3000 マニュアル内の「3 章 機能設定 - 測定値の出力条件を設定する - ホールド機能を使う(計測モード)」のタイミングチャートを参照してください. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_VARIANT		
	0	VT_I4	設定する OUT 番号を指定します. 以下のいずれかを指定してください. ・0 - OUT1 + OUT2 ・1 - OUT1 ・2 - OUT2
	1	VT_BOOL	タイミング入力を指定します. TRUE ならば ON, FALSE なら OFF を指定します.

使用例

```
' SetTiming実行
Dim param As Variant
param = Array(0, True)
Call caoCtrl.Execute("SetTiming", param)
```

3.3.2.3. SetZero コマンド

オートゼロを設定します. オートゼロを ON にすると測定中の測定値をゼロにします. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_VARIANT		
	0	VT_I4	設定する OUT 番号を指定します. 以下のいずれかを指定してください. ・0 - OUT1 + OUT2 ・1 - OUT1 ・2 - OUT2
	1	VT_BOOL	オートゼロを指定します. TRUE ならば ON, FALSE なら OFF を指定します.

使用例

```
' SetZero実行
Dim param As Variant
param = Array(0, True)
Call caoCtrl.Execute("SetZero ", param)
```

3.3.2.4. SetReset コマンド

リセット入力を設定し, 指定した OUT 番号の測定値をリセットします. 以下に引数を示します.

項目	型説明	
引数	VT_I4	設定する OUT 番号を指定します. 以下のいずれかを指定してください. ・0 - OUT1 + OUT2 ・1 - OUT1 ・2 - OUT2

使用例

```
' SetReset実行
Call caoCtrl.Execute("SetReset", 0)
```

3.3.2.5. SetPanelLock コマンド

表示パネルのキー操作をロックします. 表示パネルをロックすることで, 誤って操作キーに触れても誤操作を防止できます. 以下に引数を示します.

項目	型説明	
引数	VT_BOOL	パネルロックを指定します．TRUE ならば ON，FALSE なら OFF を指定します．

使用例

```
' SetPanelLock実行
Call caoCtrl.Execute("SetPanelLock", true)
```

3.3.2.6. SetProgramNo コマンド

プログラム番号を切り替えます．以下に引数を示します．

項目	型説明	
引数	VT_I4	設定するプログラム番号を指定します．0～7 までの値を指定できます．

使用例

```
' SetProgramNo実行
Call caoCtrl.Execute("SetProgramNo ", 0)
```

3.3.2.7. GetProgramNo コマンド

現在のプログラム番号を取得します．以下に戻り値を示します．

項目	型説明	
戻り値	VT_I4	プログラム番号を取得します．

使用例

```
' GetProgramNo実行
Dim value As Integer
value = caoCtrl.Execute("GetProgramNo")
```

3.3.2.8. GetFigureData コマンド

統計処理をした測定値の統計結果を取得します．統計処理の対象となるデータは，各測定モードにおいてホールドされたデータです．以下に引数と戻り値を示します．

項目	型説明	
引数	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
戻り値	VT_ARRAY | VT_VARIANT	

項目	型説明		
	0	VT_R4 or VT_EMPTY	平均の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
	1	VT_R4 or VT_EMPTY	最大の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
	2	VT_R4 or VT_EMPTY	最小の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
	3	VT_R4 or VT_EMPTY	最大値 － 最小値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
	4	VT_R4 or VT_EMPTY	標準偏差．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
	5	VT_I4	総データ数．
	6	VT_I4	High 判定データ数．High 判定になった測定値の総データ数です．
	7	VT_I4	Go 判定データ数．Go 判定になった測定値の総データ数です．
	8	VT_I4	Low 判定データ数．Low 判定になった測定値の総データ数です．

使用例

```
' GetFigureData実行
Dim values As Variant
values = caoCtrl.Execute("GetFigureData", 1)

If Not IsEmpty(values) Then
    If Not IsEmpty(values(0)) Then
        ' 平均値
        Dim average As Single
        average = values(0)
    End If
    If Not IsEmpty(values(1)) Then
        ' 最大値
        Dim maxValue As Single
        maxValue = values(1)
```

```
    End If
    If Not IsEmpty(values(2)) Then
        ' 最小値
        Dim minValue As Single
        minValue = values(2)
    End If
    If Not IsEmpty(values(3)) Then
        ' 最大値 - 最小値
        Dim difValue As Single
        difValue = values(3)
    End If
    If Not IsEmpty(values(4)) Then
        ' 標準偏差
        Dim stndDev As Single
        stndDev = values(4)
    End If

    ' 総データ数
    Dim totalDataCnt As Long
    totalDataCnt = values(5)

    ' High判定データ数
    Dim highDataCnt As Long
    highDataCnt = values(6)

    ' Go判定データ数
    Dim goDataCnt As Long
    goDataCnt = values(7)

    ' Low判定データ数
    Dim lowDataCnt As Long
    lowDataCnt = values(8)

End If
```

3.3.2.9. ClearFigureData コマンド

統計値をクリアします. 以下に引数を示します.

項目	型説明	
引数	VT_I4	クリアする OUT 番号を指定します. 以下のいずれかを指定してください. ・0 - OUT1 + OUT2 ・1 - OUT1 ・2 - OUT2

使用例

```
' ClearFigureData実行
Call caoCtrl.Execute("ClearFigureData", 0)
```

3.3.2.10. StartDataStorage コマンド

データストレージにデータの蓄積を開始します.

3.3.2.11. StopDataStorage コマンド

データストレージにデータの蓄積を停止します.

3.3.2.12. ClearDataStorage コマンド

データストレージの蓄積データをクリアします.

3.3.2.13. GetDataStorageData コマンド

データストレージの蓄積データを取得します. また本コマンドを実行する際には, 蓄積データ数に応じて, 3.2.1.1.AddContoller メソッドにて応答待機時間を設定しておく必要があります. 蓄積させるデータ数などの設定は 3.3.3.5.1.SetDataStorage コマンドを参照してください. 以下に引数と戻り値を示します.

項目	型説明		
引数	VT_I4		OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
戻り値	VT_EMPTY		蓄積データがない場合
	又は		
	VT_ARRAY | VT_VARIANT		
	i	VT_R4 or VT_EMPTY	測定値. 測定値の状態によって取得される値が変化します. 取得される測定値の詳細は表 3-3 を参照してください.

※ i - 取得した蓄積データ数分

使用例

4.2.1.6.データストレージの蓄積データを取得を参照してください.

3.3.2.14. GetDataStorageStatus コマンド

データストレージの蓄積状態を取得します. 以下に戻り値を示します.

項目	型説明
戻り値	VT_ARRAY | VT_VARIANT

項目	型説明		
	0	VT_BOOL	蓄積中かどうかを取得します. ・TRUE - 蓄積中 ・FALSE - 停止中
	1	VT_I4	OUT1 測定値の蓄積されているデータ件数
	2	VT_I4	OUT2 測定値の蓄積されているデータ件数

使用例

4.2.1.5. データストレージの蓄積状態を取得を参照してください.

3.3.3. 設定内容関連コマンド

3.3.3.1. パネル表示関連コマンド

3.3.3.1.1. SetPanel コマンド

パネル表示を切り替えます. 以下に引数を示します.

項目	型説明	
引数	VT_I4	表示させる OUT 番号を指定します. 以下のいずれかを指定してください. ・0 - OUT1+OUT2 ・1 - OUT1 ・2 - OUT2

使用例

```
' SetPanel実行
Call caoCtrl.Execute("SetPanel", 0)
```

3.3.3.1.2. GetPanel コマンド

表示中のパネルを取得します. 以下に戻り値を示します.

項目	型説明	
戻り値	VT_I4	表示中の OUT 番号を取得します. 値の詳細は 3.3.3.1.1 を参照してください.

使用例

```
' GetPanel実行
Dim outNo As Integer
outNo = caoCtrl.Execute("GetPanel")
```

3.3.3.2. 公差設定関連コマンド

3.3.3.2.1. SetTolerance コマンド

許容範囲の判定値(公差判定値)を設定します. それぞれ上限値を超えたとき(HI), 下限値を超えたとき(LO), 許容範囲(GO)の 3 段階に判定し, 表示と出力をすることができます. また, 測定値が公差判定値の付近で上下している場合は, 判定出力が ON/OFF を繰り返すことがあります. ヒステリシスを設定すると公差判定の検出値と復帰値に幅ができるので, このような状態を防ぐことができます. 表示内容及びヒステリシスに関しましては, LK-G3000 マニュアル内の「2 章 - 公差判定値を設定する」を参照してください. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
	1	VT_I4	公差上限値を指定します. -999999～999999 の値を指定します.
	2	VT_I4	公差下限値を指定します. -999999～999999 の値を指定します.
	3	VT_I4	公差ヒステリシスを指定します. 0～999999 の値を指定します.

※ 「公差上限値 -（公差下限値）> 公差ヒステリシス」を満たすように指定してください. 以下に例を記述します.

良い例 -> 公差上限値 = 1000, 公差下限値 = 100, 公差ヒステリシス = 0

悪い例 -> 公差上限値 = -1000, 公差下限値 = 100, 公差ヒステリシス = 0

使用例

```
' SetTolerance実行
Dim param As Variant
param = Array(1, 1000, 100, 0)
Call caoCtrl.Execute("SetTolerance", param)
```

3.3.3.2.2. GetTolerance コマンド

許容範囲の判定値(公差判定値)を取得します. 以下に引数と戻り値を示します.

項目	型説明	
引数	VT_I4	表示させる OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2

項目	型説明		
引数	VT_I4		表示させる OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	公差上限値を取得します．
	1	VT_I4	公差下限値を取得します．
	2	VT_I4	公差ヒステリシスを取得します．

使用例

```
' GetTolerance実行
Dim tolerance As Variant
tolerance = caoCtrl.Execute("GetTolerance", 1)
If Not IsEmpty(tolerance) Then
        ' 公差上限値
        Dim maxLimit As Integer
        maxLimit = tolerance(0)
        ' 公差下限値
        Dim minLimit As Integer
        minLimit = tolerance(1)
        ' 公差ヒステリシス
        Dim hysterisis As Integer
        hysterisis = tolerance(2)
End If
```

3.3.3.3. ヘッド設定関連コマンド

安定した検出を行うためのセンシングに関連する機能を設定/取得するコマンドです．

3.3.3.3.1. SetAbleMode コマンド

ABLE チューニングモードを設定します．ABLE 機能は，対象物の表面状態(色，光沢，材質)に適切な光量と感度に自動調整します．以下に引数を示します．

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	ヘッド番号を指定します．以下のいずれかを指定してください． ・1 - HEAD-A ・2 - HEAD-B

項目	型説明		
	1	VT_I4	ABLE チューニングモードを指定します. ABLE チューニングモードについては, 表 3-4 を参照してください. 以下のいずれかを指定してください. ・0 - 自動 ・1 - マニュアル

表 3-4 ABLE チューニングモード詳細

モード	機能
自動	自動で適切な光量に調節します. 通常はこちらを選択します.
マニュアル	光量と感度の調整範囲を 1 ～ 99 の任意の範囲に限定して調整します. 対象物の反射率が早い周期で大きく変化する場合や, 対象物のみを検出する場合に選択します.

使用例

```
' SetAbleMode実行
Dim param As Variant
param = Array(1, 0)
Call caoCtrl.Execute("SetAbleMode", param)
```

3.3.3.3.2. GetAbleMode コマンド

ABLE チューニングモードを取得します. 以下に引数と戻り値を示します.

項目	型説明	
引数	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
戻り値	VT_I4	ABLE チューニングモードを取得します. 値の詳細は 3.3.3.3.1 を参照してください.

使用例

```
' GetAbleMode実行
Dim mode As Integer
mode = caoCtrl.Execute("GetAbleMode", 1)
```

3.3.3.3.3. SetAbleMinMax コマンド

ABLE 制御範囲を設定します. 以下に引数を示します.

項目	型説明

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
	1	VT_I4	ABLE 上限値を指定します. 1～99 の値を指定します.
	2	VT_I4	ABLE 下限値を指定します. 1～99 の値を指定します.

※ 「ABLE 最大値 - ABLE 最小値 >= 0」を満たさない設定を行うと, デバイスから範囲外エラーが返ってきます.

使用例

```
' SetAbleMinMax実行
Dim param As Variant
param = Array(1, 10, 8)
Call caoCtrl.Execute("SetAbleMinMax", param)
```

3.3.3.3.4. GetAbleMinMax コマンド

ABLE 制御範囲を取得します. 以下に引数と戻り値を示します.

項目	型説明		
引数	VT_I4		ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	ABLE 最小値を取得します.
	1	VT_I4	ABLE 最大値を取得します.

使用例

```
' GetAbleMinMax実行
Dim value As Variant
value = caoCtrl.Execute("GetAbleMinMax", 1)
If Not IsEmpty(value) Then
        ' ABLE最大値
        Dim maxAble As Integer
        maxAble = value(0)
        ' ABLE最小値
        Dim minAble As Integer
        minAble = value(1)
```

```
End If
```

3.3.3.3.5. SetMeasureMode コマンド

測定対象物に合わせて測定モードを設定します．測定する対象物がどのようなものかを指定することで，安定した検出を行います．以下に引数を示します．

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	ヘッド番号を指定します．以下のいずれかを指定してください． ・1 - HEAD-A ・2 - HEAD-B
	1	VT_I4	測定モードを指定します．各測定モードの詳細については表 3-5 を参照してください．以下のいずれかを指定してください． ・0 - 標準 ・1 - 半透明体 ・2 - 透明体 ・3 - 透明体 2 ・4 - 多重反射体

表 3-5 測定モード詳細

モード	機能
標準	通常はこの設定を使用します．
半透明体	半透明樹脂など，光を吸収するような対象物に対応します．
透明体	透明体の変異測定や厚み測定に使用します． 透明体の複数面の反射率が同党の場合に使用します．
透明体 2	透明体の表面や裏面などの複数面(最大 4 面)の反射率が異なる場合などに使用します．
多重反射体	IC やコネクタの端子の曲がり測定などに使用します．

使用例

```
' SetMeasureMode実行
Dim param As Variant
param = Array(1, 0)
Call caoCtrl.Execute("SetMeasureMode", param)
```

3.3.3.3.6. GetMeasureMode コマンド

測定モードを取得します．以下に引数と戻り値を示します．

項目	型説明

項目	型説明	
引数	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
戻り値	VT_I4	測定モードを取得します. 値の詳細は 3.3.3.3.5 を参照してください.

使用例

```
' GetMeasureMode実行
Dim mode As Integer
mode = caoCtrl.Execute("GetMeasureMode", 1)
```

3.3.3.3.7. SetNumAlarm コマンド

アラーム処理回数を設定します. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
	1	VT_I4	アラーム処理回数を指定します. 0～999 の値を指定します.

使用例

```
' SetNumAlarm実行
Dim param As Variant
param = Array(1, 5)
Call caoCtrl.Execute("SetNumAlarm", param)
```

3.3.3.3.8. SetAlarmLevel コマンド

アラームレベルを設定します. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B

項目	型説明		
	1	VT_I4	アラームレベルを指定します. 0～9 の値を指定します. 大きな値になるほどアラームになりやすくなります.

使用例

```
' SetAlarmLevel実行
Dim param As Variant
param = Array(1, 3)
Call caoCtrl.Execute("SetAlarmLevel", param)
```

3.3.3.3.9. GetAlarm コマンド

アラーム情報を取得します. 以下に引数と戻り値を示します.

項目	型説明		
引数	VT_I4		ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	アラームレベルを取得します.
	1	VT_I4	アラーム処理回数を取得します.

使用例

```
' GetAlarm実行
Dim alarmvalue As Variant
alarmvalue = caoCtrl.Execute("GetAlarm", 1)
If Not IsEmpty(alarmvalue) Then
        ' アラームレベル
        Dim alarmLevel As Integer
        alarmLevel = alarmvalue(0)
        ' アラーム処理回数
        Dim numAlarm As Integer
        numAlarm = alarmvalue(1)
End If
```

3.3.3.3.10. StartABLE コマンド

ABLE チューニングを開始します. ABLE チューニング機能は, 対象のヘッドが実際に対象物を測定することで, ABLE の調整範囲を最適化します. 図 3-1 を参考に ABLE チューニングを行ってください. ABLE チューニングを実行すると, ABLE チューニングモードはマニュアルに, ABLE 上限値, ABLE 下限値は調整された値に設定されます. 以下に引数を示します.

項目	型説明	
引数	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B

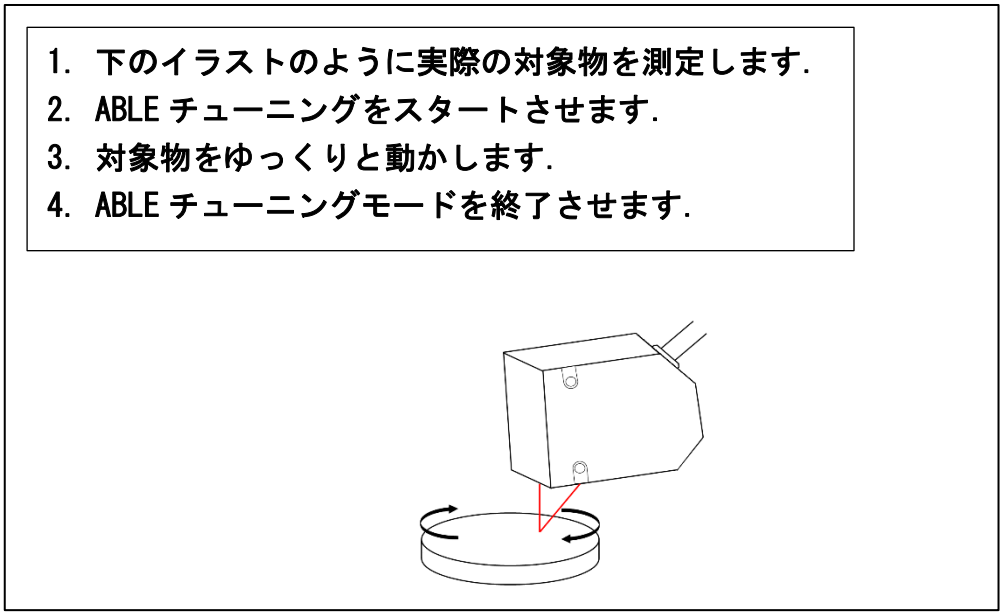


図 3-1 ABLE チューニングの流れ

使用例

```
' StartABLE実行
Call caoCtrl.Execute("StartABLE", 1)
```

3.3.3.3.11. StopABLE コマンド

ABLE チューニングを終了します.

使用例

```
' StopABLE実行
Call caoCtrl.Execute("StopABLE")
```

3.3.3.3.12. CancelABLE コマンド

ABLE チューニングを中止します.

使用例

```
' CancelABLE実行
Call caoCtrl.Execute("CancelABLE")
```

3.3.3.3.13. SetReflectionMode コマンド

設置モードを設定します. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
	1	VT_I4	設置モードを指定します. 設置モードの詳細については表 3-6 を参照してください. 以下のいずれかを指定してください. ・0 - 拡散反射 ・1 - 正反射

表 3-6 設置モード詳細

モード	機能
拡散反射	拡散反射を設定します. 通常はこの設定を選択します.
正反射	正反射を設定します. 測定対象物が鏡面やガラスなどのときに選択します.

使用例

```
' SetReflectionMode実行
Dim param As Variant
param = Array(1, 0)
Call caoCtrl.Execute("SetReflectionMode", param)
```

3.3.3.3.14. GetReflectionMode コマンド

設置モードを取得します. 以下に引数と戻り値を示します.

項目	型説明	
引数	VT_I4	ヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - HEAD-A ・2 - HEAD-B
戻り値	VT_I4	設置モードを取得します. 値の詳細は 3.3.3.3.13 を参照してください.

使用例

```
' GetReflectionMode実行
Dim mode As Integer
mode = caoCtrl.Execute("GetReflectionMode", 1)
```

3.3.3.4. OUT 設定関連コマンド

データ処理に関連する機能を設定/取得するコマンドです.

3.3.3.4.1. SetCalcMethod コマンド

ヘッド間の演算方法を設定します. 測定する対象物に応じて, ヘッド A, またはヘッド B のヘッド設定で得られたデータを演算することで表面変位, 厚み, 段差測定ができます. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
	1	VT_I4	演算方法を指定します. 演算方法の詳細については表 3-7 を参照してください. 以下のいずれかを指定してください. ・0 - ヘッド A ・1 - ヘッド B ・2 - ヘッド A+ヘッド B ・3 - ヘッド A-ヘッド B ・4 - ヘッド A 透明体 ・5 - ヘッド B 透明体
	2	VT_I4	測定対象を指定します. 測定対象の詳細については表 3-8 を参照してください. 以下のいずれかを指定してください. ・0 - ピーク 1 ・1 - ピーク 2 ・2 - ピーク 3 ・3 - ピーク 4 ・4 - ピーク 1 - ピーク 2 ・5 - ピーク 1 - ピーク 3 ・6 - ピーク 1 - ピーク 4 ・7 - ピーク 2 - ピーク 3 ・8 - ピーク 2 - ピーク 4 ・9 - ピーク 3 - ピーク 4

表 3-7 演算方法詳細

演算	機能
ヘッド A	ヘッド A, またはヘッド B の表面変位測定

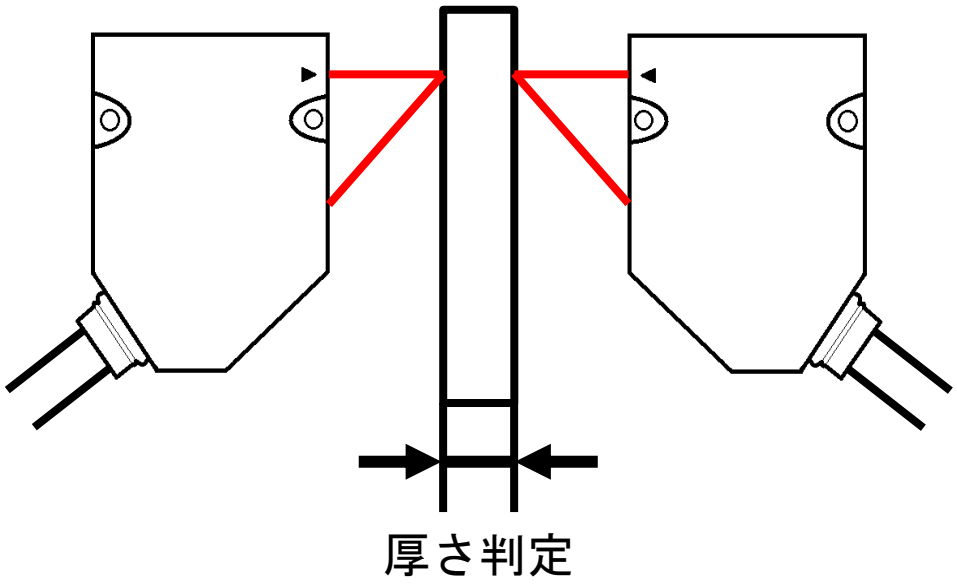
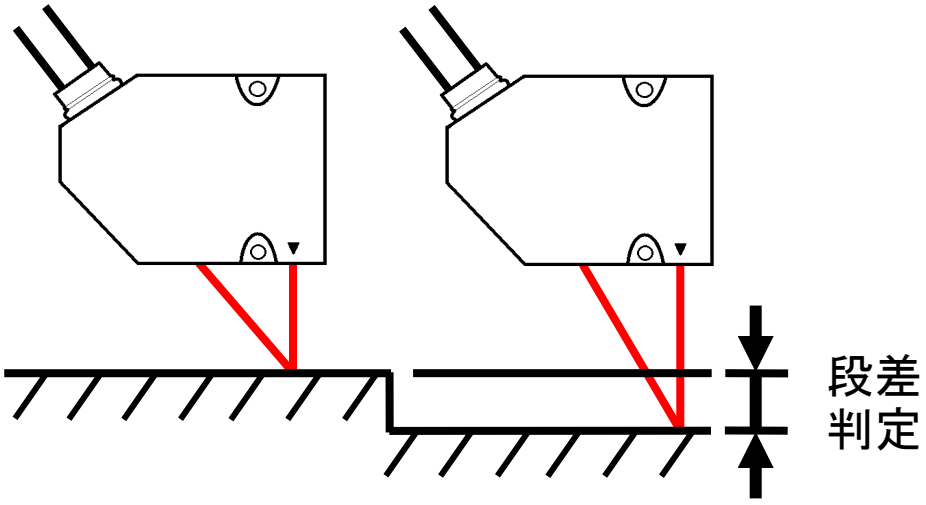
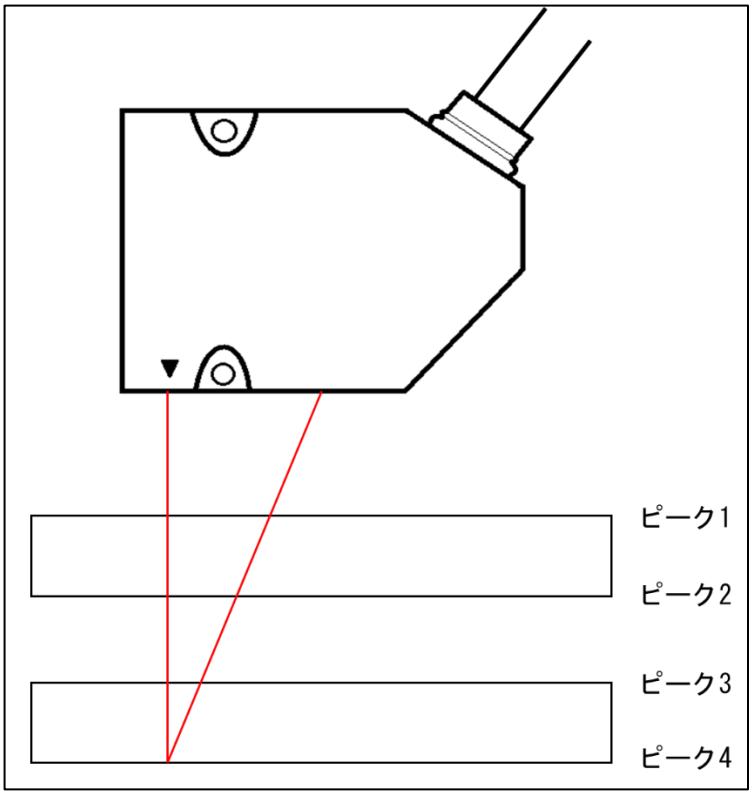
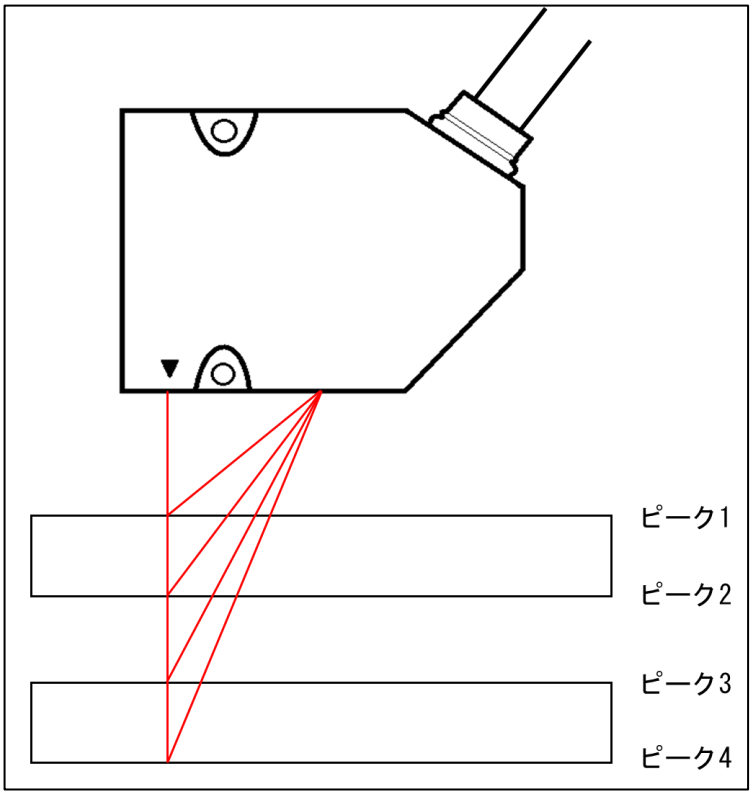
演算	機能
ヘッド B	
ヘッド A+ヘッド B	ヘッド A とヘッド B を使用した厚さ判定 
ヘッド A-ヘッド B	ヘッド A とヘッド B を使用した厚さ判定 
ヘッド A 透明体	透明体の変位測定や厚さ測定をします．対象面の選択は，測定対象にて指定します.
ヘッド B 透明体	

表 3-8 測定対象詳細

測定面	機能
ピーク 1	1 反射面の変位測定

<table>
<tr><th>測定面</th><th>機能</th></tr>
<tr><td>ピーク 2
ピーク 3
ピーク 4</td><td> 
</td></tr>
<tr><td>ピーク 1 － 2
ピーク 1 － 3
ピーク 1 － 4
ピーク 2 － 3
ピーク 2 － 4</td><td>2 反射面の測定と「ピーク 1 － 2」を選択すれば，1 枚目のガラスの厚みを測定できます．また，「ピーク 2 － 3」を選択すれば，1 枚目と 2 枚目のガラスの間隔を測定できます．</td></tr>
</table>

測定面	機能
ピーク 3 － 4	

使用例

```
' SetCalcMethod実行
Dim param As Variant
param = Array(1, 0, 0)
Call caoCtrl.Execute("SetCalcMethod", param)
```

3.3.3.4.2. GetCalcMethod コマンド

演算方法を取得します．以下に引数と戻り値を示します．

項目	型説明		
引数	VT_I4		OUT 番号を指定します．以下のいずれかを指定してください． ・1 － OUT1 ・2 － OUT2
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	演算方法を取得します．値の詳細は 3.3.3.4.1 を参照してください．
	1	VT_I4	測定対象を取得します．値の詳細は 3.3.3.4.2 を参照してください．

使用例

```
' GetCalcMethod実行
```

```
Dim value As Variant
value = caoCtrl.Execute("GetCalcMethod", 1)
If Not IsEmpty(value) Then
        ' 演算方法
        Dim method As Integer
        method = value(0)
        ' 測定対象
        Dim target As Integer
        target = value(1)
End If
```

3.3.3.4.3. SetScaling コマンド

スケーリングを設定します．スケーリングを設定することで，測定値に対する表示値を任意に校正することができます．校正は任意の 2 点のポイントに対してそれぞれ表示させる値を設定します．OUT1，OUT2 それぞれに対してヘッド A，ヘッド B の校正が出来ます．スケーリングの詳細につきましては，LK-G3000 マニュアルを参照してください．以下に引数を示します．

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号とヘッド番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 の HEAD-A ・2 - OUT1 の HEAD-B ・3 - OUT2 の HEAD-A ・4 - OUT2 の HEAD-B
	1	VT_I4	1 点目測定値を指定します． -999999～999999 の値を指定します．
	2	VT_I4	1 点目表示値を指定します． -999999～999999 の値を指定します．
	3	VT_I4	2 点目測定値を指定します． -999999～999999 の値を指定します．
	4	VT_I4	2 点目表示値を指定します． -999999～999999 の値を指定します．

※ 以下の条件を満たさない場合，設定に失敗します．

(1) 入力値 1 - 入力値 2 ≠ 0

(2) (2 点目表示値 - 1 点目表示値) / (2 点目測定値 - 1 点目測定値) < 10

使用例

```
' SetScaling実行
Dim param As Variant
```

```
param = Array(1, 500, 10, 1000, 1000)
Call caoCtrl.Execute("SetScaling", param)
```

3.3.3.4.4. GetScaling コマンド

スケーリングを取得します. 以下に引数と戻り値を示します.

項目	型説明		
引数	VT_I4		OUT 番号とヘッド番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 の HEAD-A ・2 - OUT1 の HEAD-B ・3 - OUT2 の HEAD-A ・4 - OUT2 の HEAD-B
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	1 点目測定値を取得します.
	1	VT_I4	1 点目表示値を取得します.
	2	VT_I4	2 点目測定値を取得します.
	3	VT_I4	2 点目表示値を取得します.

使用例

```
' GetScaling実行
Dim value As Variant
value = caoCtrl.Execute("GetScaling", 1)
If Not IsEmpty(value) Then
        ' 1点目測定値
        Dim inputValue1 As Integer
        inputValue1 = value(0)
        ' 1点目表示値
        Dim outputValue1 As Integer
        outputValue1 = value(1)
        ' 2点目測定値
        Dim inputValue2 As Integer
        inputValue2 = value(2)
        ' 2点目表示値
        Dim outputValue2 As Integer
        outputValue2 = value(3)
End If
```

3.3.3.4.5. SetFilter コマンド

フィルタを設定します. フィルタをかけることにより, 安定した測定を行うことができます. フィルタ機能の詳細に関しては, LK-G3000 マニュアル内の「3 章 -測定値の出力条件を設定する - フィ

ルタをかけて安定した測定をする」を参照してください. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
	1	VT_I4	フィルタモードを指定します. 以下のいずれかを指定してください. ・0 - 移動平均 ・1 - ローパスフィルタ ・2 - ハイパスフィルタ
	2	VT_I4	上記のフィルタモードによって指定する内容が変わります. 指定するフィルタモードから以下のいずれかを指定してください. [フィルタモードが移動平均(0)の場合] 平均回数を指定します. 以下のいずれかを指定してください. ・0 - 1 回 ・1 - 4 回 ・2 - 16 回 ・3 - 64 回 ・4 - 256 回 ・5 - 1024 回 ・6 - 4096 回 ・7 - 16384 回 ・8 - 65536 回 ・9 - 262144 回 [フィルタモードがローパスフィルタ(1)又はハイパスフィルタ(2)の場合] カットオフ周波数を指定します. 以下のいずれかを指定してください. ・0 - 1000Hz ・1 - 300Hz ・2 - 100Hz ・3 - 30Hz ・4 - 10Hz ・5 - 3Hz ・6 - 1Hz ・7 - 0.3Hz ・8 - 0.1Hz

使用例

```
' SetFilter実行
Dim param As Variant
param = Array(1, 0, 0)
Call caoCtrl.Execute("SetFilterMode", param)
```

3.3.3.4.6. GetFilter コマンド

フィルタ情報を取得します．以下に引数と戻り値を示します．

項目	型説明		
引数	VT_I4		OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	フィルタモードを取得します．値の詳細は 3.3.3.4.5 を参照してください．
	1	VT_I4	上記のフィルタモードによって取得する内容が変わります．値の詳細は 3.3.3.4.5 を参照してください．

使用例

```
' GetFilter実行
Dim value As Variant
value = caoCtrl.Execute("GetFilterMode", 1)
If Not IsEmpty(value) Then
        ' フィルターモード
        Dim filterMode As Integer
        filterMode = value(0)
        ' フィルターモードに準じたデータ
        Dim filterValue As Integer
        filterValue = value(1)
End If
```

3.3.3.4.7. SetTriggerMode コマンド

トリガモードを設定します．以下に引数を示します．

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2

項目	型説明		
	1	VT_I4	トリガモードを指定します．トリガモードの詳細については，表 3-9 を参照してください．以下のいずれかを指定してください． ・0 - 外部トリガ 1 ・1 - 外部トリガ 2

表 3-9 トリガモード詳細

モード	機能		
	標準	ピークホールド/ボトムホールド/ピーク to ピークホールド/アベレージホールド	サンプルホールド
トリガ 1	タイミング入力が立ち上がった(ON)ときの内部測定値を ON になっている期間ホールドします．	タイミング入力が立ち上がりエッジから次の立ち上がりエッジまでをサンプリング期間とします．	タイミング入力が立ち上がったときの内部測定値をホールドします．
トリガ 2		タイミング入力が立ち下がり(OFF)エッジから次の立ち上がりエッジまでをサンプリング期間とします．	タイミング入力が立ち上がると，そのときから設定された平均回数分のデータをサンプリングして確定した内部測定値をホールドします．

使用例

```
' SetTriggerMode実行
Dim param As Variant
param = Array(1, 0)
Call caoCtrl.Execute("SetTriggerMode", param)
```

3.3.3.4.8. GetTriggerMode コマンド

トリガモードを取得します．以下に引数と戻り値を示します．

項目	型説明	
引数	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
戻り値	VT_I4	トリガモードを取得します．値の詳細は 3.3.3.4.7 を参照してください．

使用例

```
' GetTriggerMode実行
Dim mode As Integer
mode = caoCtrl.Execute("GetTriggerMode", 1)
```

3.3.3.4.9. SetOffset コマンド

オフセットを設定します．オフセットを設定することで，表示値に任意の値を加算，減算することができます．また，オフセットを設定しておくと，オートゼロを実行したときに，オフセット値を表示できます．オフセット値は，計測モード処理，オートゼロ処理を行った後の測定値に対して設定されます．以下に引数を示します．

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
	1	VT_I4	オフセットを指定します． -999999～999999 の値を指定します．

使用例

```
' SetOffset実行
Dim param As Variant
param = Array(1, 10)
Call caoCtrl.Execute("SetOffset", param)
```

3.3.3.4.10. GetOffset コマンド

オフセットを取得します．以下に引数と戻り値を示します．

項目	型説明	
引数	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
戻り値	VT_I4	オフセットを取得します．

使用例

```
' GetOffset実行
Dim offset As Integer
offset = caoCtrl.Execute("GetOffset", 1)
```

3.3.3.4.11. SetAnalogScaling コマンド

アナログ出力スケーリングを設定します.アナログ出力スケーリグの詳細につきましては,LK-G3000 マニュアル内の「3 章 -測定値の出力条件を設定する - アナログ出力をスケーリングする」を参照してください. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
	1	VT_I4	1 点目の表示値を指定します. -999999～999999 の値を指定します.
	2	VT_I4	1 点目の出力電圧値をmV 単位で指定します. -10500～10500 の値を指定します.
	3	VT_I4	2 点目の表示値を指定します. -999999～999999 の値を指定します.
	4	VT_I4	2 点目の出力電圧値をmV 単位で指定します. -10500～10500 の値を指定します.

※ 以下の条件を満たさない場合, 設定に失敗します.

(1) 1 点目の表示値 - 2 点目の表示値 ≠ 0

(2) (2 点目出力電圧値 - 1 点目出力電圧値) / (2 点目表示値 - 1 点目表示値) ≦ 10

使用例

```
' SetAnalogScaling実行
Dim param As Variant
param = Array(1, 500, 10, 1000, 1000)
Call caoCtrl.Execute("SetAnalogScaling", param)
```

3.3.3.4.12. GetAnalogScaling コマンド

アナログ出力スケーリングを取得します. 以下に引数と戻り値を示します.

項目	型説明		
引数	VT_I4		OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	1 点目の表示値を取得します.
	1	VT_I4	1 点目の出力電圧値を取得します.
	2	VT_I4	2 点目の表示値を取得します.

項目	型説明		
	3	VT_I4	2 点目の出力電圧値を取得します.

使用例

```
' GetAnalogScaling実行
Dim value As Variant
value = caoCtrl.Execute("GetAnalogScaling", 1)
If Not IsEmpty(value) Then
        ' 1点目表示値
        Dim inputValue1 As Integer
        inputValue1 = value(0)
        ' 1点目出力電圧値
        Dim outputValue1 As Integer
        outputValue1 = value(1)
        ' 2点目表示値
        Dim inputValue2 As Integer
        inputValue2 = value(2)
        ' 2点目出力電圧値
        Dim outputValue2 As Integer
        outputValue2 = value(3)
End If
```

3.3.3.4.13. SetCalcMode コマンド

計測モードを設定します. 計測モードの詳細については, LK-G3000 マニュアル内の「3 章 -測定値の出力条件を設定する - ホールド機能を使う(計測モード)」を参照してください. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
	1	VT_I4	計測モードを指定します. 指定可能な計測モードの詳細については表 3-10 を参照してください. 以下のいずれかを指定してください. ・0 - 標準 ・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 - サンプルホールド ・5 - アベレージホールド

表 3-10 計測モード詳細

モード	機能
標準	測定した結果を随時表示/出力できます.
ピークホールド	サンプリング期間内の最大値を測定できます.
ボトムホールド	サンプリング期間内の最小値を測定できます.
ピーク to ピークホールド	サンプリング期間内の「最大値-最小値」を測定できます.
サンプルホールド	タイミング入力を ON にした瞬間の値を測定できます.
アベレージホールド	サンプリング期間内の平均値を測定できます.

使用例

```
' SetCalcMode実行
Dim param As Variant
param = Array(1, 0)
Call caoCtrl.Execute("SetCalcMode", param)
```

3.3.3.4.14. GetCalcMode コマンド

計測モードを取得します. 以下に引数と戻り値を示します.

項目	型説明	
引数	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
戻り値	VT_I4	計測モードを取得します. 値の詳細は 3.3.3.4.13 を参照してください.

使用例

```
' GetCalcMode実行
Dim mode As Integer
mode = caoCtrl.Execute("GetCalcMode", 1)
```

3.3.3.4.15. SetDisplayUnit コマンド

パネルに表示させる最小表示単位を設定します. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2

項目	型説明		
	1	VT_I4	最小表示単位を指定します．以下のいずれかを指定してください． ・0 - 0.01mm ・1 - 0.001mm ・2 - 0.0001mm ・3 - 0.00001mm ・4 - 0.01μm ・5 - 0.001μm

使用例

```
' SetDisplayUnit実行
Dim param As Variant
param = Array(1, 0)
Call caoCtrl.Execute("SetDisplayUnit", param)
```

3.3.3.4.16. GetDisplayUnit コマンド

最小表示単位を取得します．以下に引数と戻り値を示します．

項目	型説明	
引数	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2
戻り値	VT_I4	最小表示単位を取得します．値の詳細は 3.3.3.4.15 を参照してください．

使用例

```
' GetDisplayUnit実行
Dim unit As Integer
unit = caoCtrl.Execute("GetDisplayUnit", 1)
```

3.3.3.4.17. SetAnalogThrough コマンド

アナログスルーを設定します．計測モードで測定値をホールドしている場合に，アナログスルーを ON に設定すると，ホールドする前の内部測定値をアナログ出力します．以下に引数を示します．

項目	型説明		
引数	VT_ARRAY | VT_VARIANT		
	0	VT_I4	OUT 番号を指定します．以下のいずれかを指定してください． ・1 - OUT1 ・2 - OUT2

項目	型説明		
	1	VT_BOOL	アナログスルーを指定します. ・TRUE - ON ・FALSE - OFF

使用例

```
' SetAnalogThrough実行
Dim param As Variant
param = Array(1, false)
Call caoCtrl.Execute("SettAnalogThrough", param)
```

3.3.3.4.18. GetAnalogThrough コマンド

アナログスルーを取得します. 以下に引数と戻り値を示します.

項目	型説明	
引数	VT_I4	OUT 番号を指定します. 以下のいずれかを指定してください. ・1 - OUT1 ・2 - OUT2
戻り値	VT_BOOL	アナログスルーを取得します.

使用例

```
' GetAnalogThrough実行
Dim value As Variant
value = caoCtrl.Execute("GetAnalogThrough", 1)
```

3.3.3.5. 共通設定関連コマンド

ヘッド設定と OUT 設定に関連する共通機能を設定/取得するコマンドです.

3.3.3.5.1. SetDataStorage コマンド

データストレージの蓄積させる対象 OUT, 蓄積点数, 蓄積周期を設定します. 以下に引数を示します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	対象 OUT を指定します. 以下のいずれかを指定してください. ・0 - 対象 OUT なし ・1 - OUT1 ・2 - OUT2 ・3 - OUT1 と OUT2

項目	型説明		
	1	VT_I4	蓄積点数を指定します. 1～65536 の値を指定します.
	2	VT_I4	蓄積周期を指定します. 以下のいずれかを指定してください. ・0 - サンプリング周期×1 ・1 - サンプリング周期×2 ・2 - サンプリング周期×5 ・3 - サンプリング周期×10 ・4 - サンプリング周期×20 ・5 - サンプリング周期×50 ・6 - サンプリング周期×100 ・7 - サンプリング周期×200 ・8 - サンプリング周期×500 ・9 - サンプリング周期×1000

使用例

```
' SetDataStorage実行
Dim param As Variant
param = Array(1, 100, 0)
Call caoCtrl.Execute("SetDataStorage", param)
```

3.3.3.5.2. GetDataStorage コマンド

データストレージの対象 OUT, 蓄積点数, 蓄積周期を取得します. 以下に引数と戻り値を示します.

項目	型説明		
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	対象 OUT を取得します. 値の詳細は 3.3.3.5.1 を参照してください.
	1	VT_I4	蓄積点数を取得します.
	2	VT_I4	蓄積周期を取得します. 値の詳細は 3.3.3.5.1 を参照してください.

使用例

```
' GetDataStorage実行
Dim value As Variant
value = caoCtrl.Execute("GetDataStorage")
If Not IsEmpty(value) Then
        ' 対象OUT
        Dim out As Integer
        out = value(0)
        ' 蓄積点数
        Dim dataCnt As Integer
```

```
        dataCnt = value(1)
        ' 蓄積周期
        Dim cycle As Integer
        cycle = value(2)
End If
```

3.3.3.5.3. SetSamplingCycle コマンド

サンプリング周期を設定します. 以下に引数を示します.

項目	型説明	
引数	VT_I4	サンプリング周期を指定します. 以下のいずれかを指定してください. ・0 - 20μs ・1 - 50μs ・2 - 100μs ・3 - 200μs ・4 - 500μs ・5 - 1000μs

使用例

```
' SetSamplingCycle実行
Call caoCtrl.Execute("SetSamplingCycle", 0)
```

3.3.3.5.4. GetSamplingCycle コマンド

サンプリング周期を取得します. 以下に戻り値を示します.

項目	型説明	
戻り値	VT_I4	サンプリング周期を取得します. 値の詳細は 3.3.3.5.3 を参照してください.

使用例

```
' GetSamplingCycle実行
Dim value As Integer
value = caoCtrl.Execute("GetSamplingCycle")
```

3.3.3.5.5. SetMutualInterPrev コマンド

相互干渉防止を設定します. 相互干渉防止を ON にすると 2 台のヘッドが交互に発光して相手側ヘッドの干渉を受けなくなります. 以下に引数を示します.

項目	型説明

引数	VT_BOOL	相互干渉防止を指定します. ・TRUE - ON ・FALSE - OFF

使用例

```
' SetMutualInterPrev実行
Call caoCtrl.Execute("SetMutualInterPrev", true)
```

3.3.3.5.6. GetMutualInterPrev コマンド

相互干渉防止を取得します. 以下に戻り値を示します.

項目	型説明	
戻り値	VT_BOOL	相互干渉防止を取得します.

使用例

```
' GetMutualInterPrev実行
Dim value As Variant
value = caoCtrl.Execute("GetMutualInterPrev")
```

3.3.3.5.7. SetTimingSync コマンド

OUT1 と OUT2 のタイミング入力の制御方法を設定します. 以下に引数を示します.

項目	型説明	
引数	VT_I4	タイミング同期を指定します. タイミング同期の詳細については, 表 3-11 を参照してください. 以下のいずれかを指定してください. ・0 - 非同期 ・1 - 同期

表 3-11 タイミング同期詳細

タイミング同期	機能
非同期	OUT1 と OUT2 を非同期で制御します. OUT1, OUT2 それぞれに独立した入力端子を割り当てます. ・OUT1 : 12 極端子台の 8 番 ・OUT2 : 拡張コネクタの 8 番
同期	OUT1 と OUT2 を同期して制御します. 12 極端子台の 8 番が対応し, 拡張コネクタの 6 番は無効になります.

※ 入力端子に関しましては, LK-G3000 マニュアル内の「4 章 入力端子」を参照してください.

使用例

```
' SetTimingSync実行
Call caoCtrl.Execute("SetTimingSync", 0)
```

3.3.3.5.8. GetTimingSync コマンド

タイミング同期を取得します. 以下に戻り値を示します.

項目	型説明	
戻り値	VT_I4	タイミング同期を取得します. 値の詳細は 3.3.3.5.7 を参照してください.

使用例

```
' GetTimingSync実行
Dim value As Integer
value = caoCtrl.Execute("GetTimingSync")
```

3.3.3.5.9. SetToleCompOutputFormat コマンド

公差判定出力の出力形態を設定します. 以下に引数を示します.

項目	型説明	
引数	VT_I4	判定出力形態を指定します. 判定出力形態の詳細については, 表 3-12 を参照してください. 以下のいずれかを指定してください. ・0 - ノーマル ・1 - ホールド ・2 - オフディレイ

表 3-12 判定出力形態詳細

タイミング同期	機能
ノーマル	公差判定に合わせて出力します.
ホールド	ON になった出力をホールドします. 測定値リセットでホールドを解除します.
オフディレイ	ノーマル出力に 60ms のオフディレイがかかります. 測定値リセットでホールドを解除します.

使用例

```
' SetToleCompOutputFormat実行
Call caoCtrl.Execute("SetToleCompOutputFormat", 0)
```

3.3.3.5.10. GetToleCompOutputFormat コマンド

公差判定出力の出力形態を取得します. 以下に戻り値を示します.

項目	型説明	
戻り値	VT_I4	判定出力形態を取得します. 値の詳細は 3.3.3.5.9 を参照してください.

使用例

```
' GetToleCompOutputFormat実行
Dim value As Integer
value = caoCtrl.Execute("GetToleCompOutputFormat")
```

3.3.3.5.11. SetStorobeTime コマンド

ストローブ出力が ON する時間(ワンショット出力時間)を設定します. 以下に引数を示します.

項目	型説明	
引数	VT_I4	ストローブ時間を指定します. 以下のいずれかを指定してください. ・0 - 2ms ・1 - 5ms ・2 - 10ms ・3 - 20ms

使用例

```
' SetStorobeTime実行
Call caoCtrl.Execute("SetStorobeTime", 0)
```

3.3.3.5.12. GetStorobeTime コマンド

ストローブ出力が ON する時間(ワンショット出力時間)を取得します. 以下に戻り値を示します.

項目	型説明	
戻り値	VT_I4	ストローブ時間を取得します. 値の詳細は 3.3.3.5.11 を参照してください.

使用例

```
' GetStorobeTime実行
Dim value As Integer
value = caoCtrl.Execute("GetStorobeTime")
```

3.4. 変数一覧

各クラスで使用可能な変数一覧を定義します. なお変数は, CaoVariable クラスのオブジェクトを指します.

3.4.1. CaoController クラス変数

変数名	説明	Value		参照
		get	put	
@MAKER_NAME	メーカー名を取得します.	○	-	P.54
@VERSION	DLL バージョンを取得します.	○	-	P.54
@CALCDATA	測定中の OUT1 と OUT2 の測定値を取得します.	○	○	P.54

3.4.1.1. @MAKER_NAME

メーカー名の取得をします.

データ型

型説明	
VT_BSTR	メーカー名を取得します.

使用例

```
' 変数追加
Dim var As CaoVariable
var = caoCtrl.AddVariable("@MAKER_NAME")
' 値取得
Dim name As String
name = var.value
```

3.4.1.2. @VERSION

DLL のバージョンの取得をします.

データ型

型説明	
VT_BSTR	DLL のバージョンを取得します. *.*.*

使用例

```
' 変数追加
Dim var As CaoVariable
var = caoCtrl.AddVariable("@VERSION", "")
' 値取得
Dim version As String
version = var.value
```

3.4.1.3. @CALCDATA

測定中の OUT1 と OUT2 の測定値を取得します. この変数で値を取得する場合, LK-G3000 本体の動作モードを通常モードに切り替える必要があります.

データ型

型説明		
VT_ARRAY | VT_VARIANT		
0	VT_R4 or VT_EMPTY	OUT1 の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．
1	VT_R4 or VT_EMPTY	OUT2 の測定値．測定値の状態によって取得される値が変化します．取得される測定値の詳細は表 3-3 を参照してください．

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCtrl.AddVariable("@CALCDATA")

' 値の取得
Dim calcvalues As Variant
calcvalues = caoVrl.value

If Not IsEmpty(calcvalues) Then
        If Not IsEmpty(calcvalues(0)) Then
                ' OUT1 の測定値
                Dim value1 As Single
                value1 = calcvalues(0)
        End If
        If Not IsEmpty(calcvalues(1)) Then
                ' OUT2 の測定値
                Dim value2 As Single
                value2 = calcvalues(1)
        End If
End If
```

4. LK-G3000 プロバイダによるプログラミング

LK-G3000 プロバイダでは，以下の手順でクライアント PC と LK-G3000 を接続することができます.

- CaoEngine の作成
- CaoWorkspace の作成
- CaoController の作成

LK-G3000 に接続した後は，CaoController の Execute メソッドを使用する，もしくは，CaoVariabe オブジェクトを生成することで， LK-G3000 の情報にアクセスすることができます.

4.1. OUT1 と OUT2 の測定値を取得するサンプルプログラミング

ここでは例として OUT1 と OUT2 の測定値を読み込むサンプルプログラムを示します. 表 4-1 にサンプルプログラムの要件を，図 4-1 にサンプルプログラムの流れをそれぞれ記述しています.

表 4-1 測定値を取得するサンプルプログラムの要件

要件	説明
接続先	RS-232C 通信で接続する
	接続先ポート番号は 1
処理内容	LK-G3000 から OUT1 及び OUT2 の測定値を読込む.

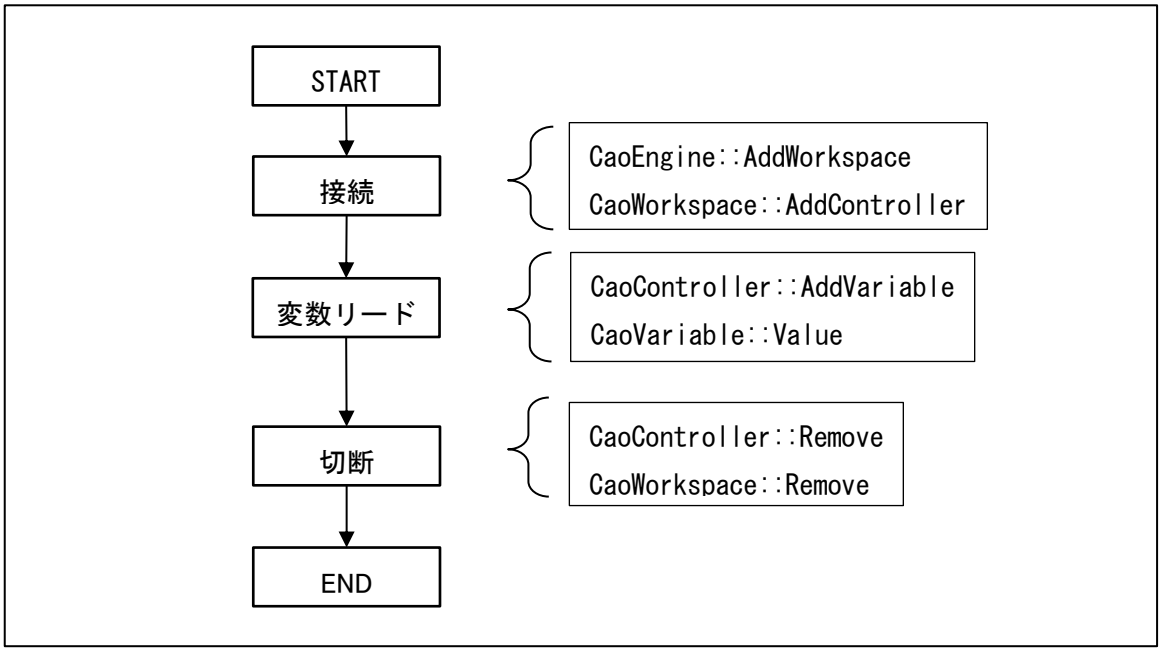


図 4-1 OUT1 及び OUT2 の測定値を読み込む流れ

以降の節から具体的なコードを示します.

4.1.1. サンプルプログラム

以下にサンプルプログラムの全体像を示します.

Sample GetCalcDataSample.cs

```
' オブジェクト
Dim caoEng As CaoEngine
Dim caoWs As CaoWorkspace
Dim caoCtrl As CaoController
Dim caoVrl As CaoVariable
Private Sub Main()
    ' 接続
    Call Connect

    ' 機器の動作モードが通常モードでないならこのコマンドを実行
    'Call caoCtrl.Execute("SetMode", 0)

    ' 値の取得
    Dim values As Variant
    values = caoVrl.value
    If Not IsEmpty(values) Then
        ' OUT1の測定値
        If Not IsEmpty(values(0)) Then
            Dim value1 As Single
            value1 = values(0)
        End If
        ' OUT2の測定値
        If Not IsEmpty(values(1)) Then
            Dim value2 As Single
            value2 = values(1)
        End If
    End If
    ' 後処理
    Call DisConnect
End Sub

' 準備メソッド
Private Sub Connect()
    ' CaoEngine オブジェクトの生成
    Set caoEng = New CaoEngine
    ' CaoWorkspace オブジェクトの生成
    Set caoWs = caoEng.AddWorkspace("Workspace", "")
    ' CaoController オブジェクトの生成
    Set caoCtrl = caoWs.AddController("LKG3000", _
                                      "CaoProv.KEYENCE.LK-G3000", _
                                      "", _
                                      "Conn=COM:1,Timeout=1000")
    ' CaoVariable オブジェクトの生成
    Set caoVrl = caoCtrl.AddVariable("@CALCDATA")
End Sub
' 後処理メソッド
Private Sub DisConnect()
```

```
    ' CaoController からCaoVariable を削除
    Call caoCtrl.variables.Remove(caoVrl.Index)
    ' CaoVariableの消去
    Set caoVrl = Nothing
    ' CaoWorkspace からCaoController を削除
    Call caoWs.Controllers.Remove(caoCtrl.Index)
    ' CaoController の消去
    Set caoCtrl = Nothing
    ' CaoEngine からCaoWorkspace を削除
    Call caoEng.Workspaces.Remove(caoWs.Index)
    ' CaoWorkspace の消去
    Set caoWs = Nothing
    ' CaoEngine の消去
    Set caoEng = Nothing
End Sub
```

4.1.1.1. 接続

LK-G3000 と接続するためには，以下の手順を取ります．

(1) オブジェクトを保持するための変数を用意します．コントローラ接続に必要なオブジェクトは，CaoEngineオブジェクトとCaoWorkspaceオブジェクトとCaoControllerオブジェクトです．CaoWorkpaceオブジェクトは，CaoControllerオブジェクトをCaoWorkspacesから取得する場合には変数を用意する必要はありません．また変数にアクセスするためのCaoVariableオブジェクトも必要になります．以下にVB6でのコード例を示します．

```
    ' CaoEngine オブジェクト用の変数
    Dim caoEng As CaoEngine
    ' CaoWorkspace オブジェクト用の変数
    Dim caoWs As CaoWorkspace
    ' CaoController オブジェクト用の変数
    Dim caoCtrl As CaoController
    ' CaoVariable オブジェクト用の変数
    Dim caoVrl As CaoVariable
```

(2) CaoEngineオブジェクトを生成します．CaoEngineオブジェクトはNewキーワードを使って生成します．

```
    ' CaoEngine オブジェクトの生成
    Set caoEng = New CaoEngine
```

(3) CaoWorkspaceオブジェクトを取得もしくは生成します．CaoEngineオブジェクトを生成すると，デフォルトでCaoWorkspacesオブジェクトとCaoWorkspaceオブジェクトを1つずつ生成しています．以下にCaoWorkspaceオブジェクトを新しく生成するコード例とデフォルトのCaoWorkspaceを示します．

```
    ' CaoWorkspace オブジェクトの生成
    Set caoWs = caoEng.AddWorkspace("Workspace", "")
```

(4) CaoControllerオブジェクトを生成します. CaoControllerオブジェクトを生成するには, 使用するプロバイダ名と使用するためのパラメータを設定します. LK-G3000プロバイダでは, 接続先のCOMポートと機器からの応答待機時間をオプションで指定します. 以下にコード例を示します.

```
' CaoController オブジェクトの生成
Set caoCtrl = caoWs.AddController("LKG3000", _
                                  "CaoProv.KEYENCE.LK-G3000", _
                                  "" _
                                  , _
                                  "Conn=COM:1,Timeout=1000")
```

(5) CaoVariableオブジェクトを生成します.取得したい変数のCaoVariableオブジェクトを生成します.以下にOUT1とOUT2の測定値にアクセスする変数オブジェクトを生成するコード例を示します.

```
' CaoVariable オブジェクトの生成
Set caoVrl = caoCtrl.AddVariable("@CALCDATA")
```

4.1.1.2. 本体の動作モード変更

LK-G3000 本体の動作モードが,「通信モード」の場合, 測定値を取得することができません. SetModeコマンドにて動作モードを変更する必要があります. 動作モードを変更するには, CaoController オブジェクトの Execute メソッドを実行します.

```
' 機器の動作モードが通常モードでないならこのコマンドを実行
Call caoCtrl.Execute("SetMode", 0)
```

4.1.1.3. OUT1 と OUT2 の測定値取得

OUT1 と OUT2 の測定値を取得するには,CaoVariable オブジェクトの Value プロパティを参照します. 測定値ごとに変数を用意する必要があります. 以下にコード例を示します.

```
' 値の取得
Dim values As Variant
values = caoVrl.value

If Not IsEmpty(values) Then
    ' OUT1の測定値
    If Not IsEmpty(values(0)) Then
            Dim value1 As Single
            value1 = values(0)
    End If
    ' OUT2の測定値
    If Not IsEmpty(values(1)) Then
            Dim value2 As Single
            value2 = values(1)
    End If
End If
```

4.1.1.4. 切断

コントローラと切断する場合には，生成したオブジェクトを消去すると共に，オブジェクトを管理するコレクションクラスから消去するオブジェクトを削除します．以下にコード例を示します．

```
' CaoWorkspace から CaoController を削除
Call caoWs.Controllers.Remove(caoCtrl.Index)
' CaoController の消去
Set caoCtrl = Nothing
' CaoEngine から CaoWorkspace を削除
Call caoEng.Workspaces.Remove(caoWs.Index)
' CaoWorkspace の消去
Set caoWs = Nothing
' CaoEngine の消去
Set caoEng = Nothing
```

4.2. データストレージの蓄積データを取得するサンプルプログラミング

ここでは例としてデータストレージにデータを蓄積させ，データストレージの蓄積データを読み込むサンプルプログラムを示します．表 4-2 にサンプルプログラムの要件を，図 4-1 にサンプルプログラムの流れをそれぞれ記述しています．

表 4-2 蓄積データを取得するサンプルプログラムの要件

要件	説明
接続先	RS-232C 通信で接続する
	接続先ポート番号は 1
処理内容	データストレージへのデータ蓄積を開始する．
	データストレージへのデータ蓄積を終了する．
	データストレージの蓄積状態を取得する．
	データストレージの蓄積データを取得する．
	データストレージの蓄積データをクリアする．

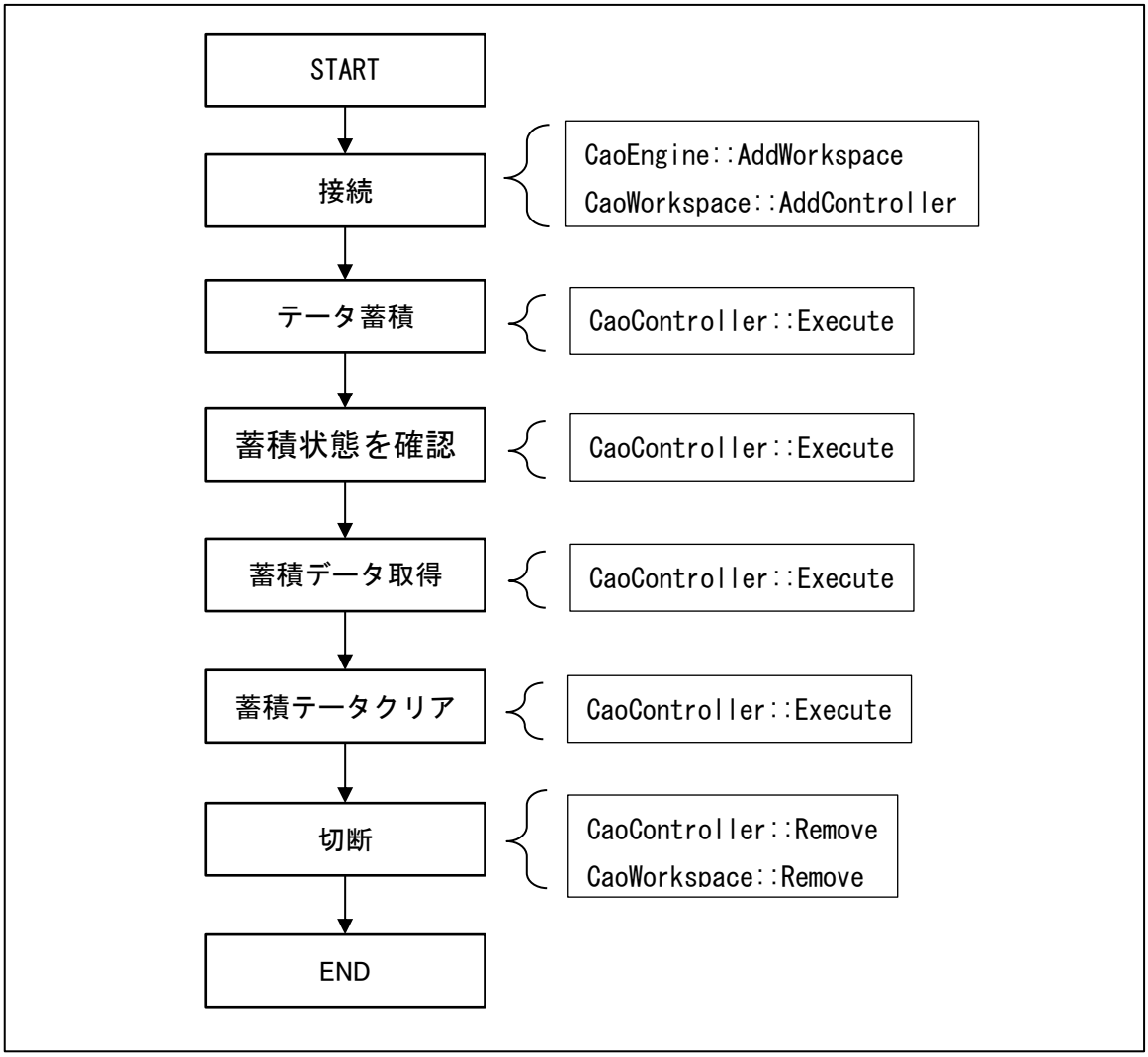


図 4-2 データストレージの蓄積データを取得する流れ

以降の節から具体的なコードを示します.

4.2.1. サンプルプログラム

以下にサンプルプログラムの全体像を示します.

Sample	DataStorageSample.cs

```
' Sleep関数を使用する場合
Private Declare Sub Sleep Lib "kernel32" (ByVal ms As Long)

' オブジェクト
Dim caoEng As CaoEngine
Dim caoWs As CaoWorkspace
Dim caoCtrl As CaoController
Dim caoVrl As CaoVariable
Private Sub Main()
        ' 接続
        Call Connect
```

```
' 機器の動作モードが通信モードでないならこのコマンドを実行
Call caoCtrl.Execute("SetMode", 1)

' データストレージの設定
Dim param As Variant
param = Array(1, 10, 1)
Call caoCtrl.Execute("SetDataStorage", param)

' 機器の動作モードが通常モードでないならこのコマンドを実行
Call caoCtrl.Execute("SetMode", 0)
' データストレージへのデータ蓄積を開始
Call caoCtrl.Execute("StartDataStorage")
' 0.5秒待機
Sleep 500

' データストレージへのデータ蓄積を終了
Call caoCtrl.Execute("StopDataStorage")
' データストレージの蓄積状態を取得
Dim statusvalue As Variant
statusvalue = caoCtrl.Execute("GetDataStorageStatus")
If Not IsEmpty(statusvalue) Then
        ' 蓄積中かどうか
        Dim isAccumulation As BooleanisAccumulation = statusvalue(0)
        ' OUT1に蓄積されているデータ件数
        Dim dataCnt1 As Integer
        dataCnt1 = statusvalue(1)
        ' OUT2に蓄積されているデータ件数
        Dim dataCnt2 As Integer
        dataCnt2 = statusvalue(2)
End If

' データストレージの蓄積データを取得
Dim datavalue As Variant
datavalue = caoCtrl.Execute("GetDataStorageData", 1)
If Not IsEmpty(datavalue) Then
        Dim i As Integer
        For i = LBound(datavalue) To (UBound(datavalue) - 1)
                If Not IsEmpty(datavalue(i)) Then
                        Dim accumulationData As Single
                        accumulationData = datavalue(i)
                End If
        Next i
End If

' データストレージの蓄積データをクリア
Call caoCtrl.Execute("ClearDataStorage")
```

```
        ' 切断
        Call DisConnect

End Sub

' 接続メソッド
Private Sub Connect()
        ' CaoEngine オブジェクトの生成
        Set caoEng = New CaoEngine
        ' CaoWorkspace オブジェクトの生成
        Set caoWs = caoEng.AddWorkspace("Workspace", "")
        ' CaoController オブジェクトの生成
        Set caoCtrl = caoWs.AddController("LKG3000", _
                                          "CaoProv.KEYENCE.LK-G3000", _
                                          "", _
                                          "Conn=COM:1,Timeout=1000")
End Sub

' 切断メソッド
Private Sub DisConnect()
        ' CaoWorkspace からCaoController を削除
        Call caoWs.Controllers.Remove(caoCtrl.Index)
        ' CaoController の消去
        Set caoCtrl = Nothing
        ' CaoEngine からCaoWorkspace を削除
        Call caoEng.Workspaces.Remove(caoWs.Index)
        ' CaoWorkspace の消去
        Set caoWs = Nothing
        ' CaoEngine の消去
        Set caoEng = Nothing
End Sub
```

4.2.1.1. 接続

LK-G3000 と接続するためには，以下の手順を取ります．

(1) オブジェクトを保持するための変数を用意します．コントローラ接続に必要なオブジェクトは，CaoEngineオブジェクトとCaoWorkspaceオブジェクトとCaoControllerオブジェクトです．CaoWorkpaceオブジェクトは，CaoControllerオブジェクトをCaoWorkspacesから取得する場合には変数を用意する必要はありません．また変数にアクセスするためのCaoVariableオブジェクトも必要になります．以下にVB6でのコード例を示します．

```
    ' CaoEngine オブジェクト用の変数
    Dim caoEng As CaoEngine
    ' CaoWorkspace オブジェクト用の変数
    Dim caoWs As CaoWorkspace
    ' CaoController オブジェクト用の変数
    Dim caoCtrl As CaoController
    ' CaoVariable オブジェクト用の変数
```

```
Dim caoVrl As CaoVariable
```

(2) CaoEngineオブジェクトを生成します．CaoEngineオブジェクトはNewキーワードを使って生成します．

```
' CaoEngine オブジェクトの生成
Set caoEng = New CaoEngine
```

(3) CaoWorkspaceオブジェクトを取得もしくは生成します．CaoEngineオブジェクトを生成すると，デフォルトでCaoWorkspacesオブジェクトとCaoWorkspaceオブジェクトを1つずつ生成しています．以下にCaoWorkspaceオブジェクトを新しく生成するコード例とデフォルトのCaoWorkspaceを示します．

```
' CaoWorkspace オブジェクトの生成
Set caoWs = caoEng.AddWorkspace("Workspace", "")
```

(4) CaoControllerオブジェクトを生成します．CaoControllerオブジェクトを生成するには，使用するプロバイダ名と使用するためのパラメータを設定します．LK-G3000プロバイダでは，接続先のCOMポートと機器からの応答待機時間をオプションで指定します．以下にコード例を示します．

```
' CaoController オブジェクトの生成
Set caoCtrl = caoWs.AddController("LKG3000", _
                                  "CaoProv.KEYENCE.LK-G3000", _
                                  "", _
                                  "Conn=COM:1,Timeout=1000")
```

4.2.1.2. データストレージの設定

必要であれば，LK-G3000 本体のデータストレージの設定をします．サンプルプログラムでは，対象 OUT を「OUT1」，データ蓄積数を「10」，蓄積周期を「サンプリング周期×2」として設定します．なお，サンプリング周期は LK-G3000 の初期値である 200μs とします．コマンドの詳細については，3.3.3.5.1.SetDataStorage コマンドを参照してください．以下にコード例を示します．

```
' 機器の動作モードが通信モードでないならこのコマンドを実行
Call caoCtrl.Execute("SetMode", 1)

' データストレージの設定
Dim param As Variant
param = Array(1, 10, 1)
Call caoCtrl.Execute("SetDataStorage", param)
```

4.2.1.3. データストレージへのデータ蓄積開始

データストレージへの蓄積を開始します．4.2.1.2. データストレージの設定にて指定した対象 OUT の測定値が指定したデータ蓄積数分，蓄積されます．対象 OUT が無しの場合，このコマンドは失敗します．以下にコード例を示します．

```
    ' 機器の動作モードが通常モードでないならこのコマンドを実行
    Call caoCtrl.Execute("SetMode", 0)

    ' データストレージへのデータ蓄積を開始
    Call caoCtrl.Execute("StartDataStorage")
```

4.2.1.4. データストレージへのデータ蓄積終了

データストレージへの蓄積を終了します．なお，4.2.1.2. データストレージの設定にて指定した蓄積数分蓄積された場合，蓄積は自動で終了します．以下にコード例を示します．

```
    ' データストレージへのデータ蓄積を終了
    Call caoCtrl.Execute("StopDataStorage")
```

4.2.1.5. データストレージの蓄積状態を取得

データストレージの蓄積状態を取得するには，CaoCaoController オブジェクトの Execute メソッドを参照します．以下にコード例を示します．

```
    ' データストレージの蓄積状態を取得
    Dim statusvalue As Variant
    statusvalue = caoCtrl.Execute("GetDataStorageStatus")
    If Not IsEmpty(statusvalue) Then
        ' 蓄積中かどうか
        Dim isAccumulation As Boolean
        isAccumulation = statusvalue(0)
        ' OUT1に蓄積されているデータ件数
        Dim dataCnt1 As Integer
        dataCnt1 = statusvalue(1)
        ' OUT2に蓄積されているデータ件数
        Dim dataCnt2 As Integer
        dataCnt2 = statusvalue(2)
    End If
```

4.2.1.6. データストレージの蓄積データを取得

データストレージの蓄積データを取得するには，CaoCaoController オブジェクトの Execute メソッドを参照します．以下にコード例を示します．

```
    ' データストレージの蓄積データを取得
    Dim datavalue As Variant
    datavalue = caoCtrl.Execute("GetDataStorageData", 1)
    If Not IsEmpty(datavalue) Then
        Dim i As Integer
        For i = LBound(datavalue) To (UBound(datavalue) - 1)
                If Not IsEmpty(datavalue(i)) Then
                        Dim accumulationData As Single
                        accumulationData = datavalue(i)
                End If
        Next i
```

```
End If
```

4.2.1.7. データストレージの蓄積データをクリア

データストレージの蓄積データをクリアするには，CaoCaoController オブジェクトの Execute メソッドを参照します．以下にコード例を示します．

```
' データストレージの蓄積データをクリア
Call caoCtrl.Execute("ClearDataStorage")
```

4.2.1.8. 切断

コントローラと切断する場合には，生成したオブジェクトを消去すると共に，オブジェクトを管理するコレクションクラスから消去するオブジェクトを削除します．以下にコード例を示します．

```
' CaoWorkspace から CaoController を削除
Call caoWs.Controllers.Remove(caoCtrl.Index)
' CaoController の消去
Set caoCtrl = Nothing
' CaoEngine から CaoWorkspace を削除
Call caoEng.Workspaces.Remove(caoWs.Index)
' CaoWorkspace の消去
Set caoWs = Nothing
' CaoEngine の消去
Set caoEng = Nothing
```

5. LK-G3000 プロバイダエラーコード

本プロバイダには，0x8011****でマスクした以下の独自エラーコードが存在します．（表 5-1 独自エラーコード表参照）

ORiN2 の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ください．

表 5-1 独自エラーコード表

エラー番号	説明
0x80110001	AddContoller時のConnオプション指定がありません.
0x80110002	AddContoller時のConnオプション指定方法が誤っています.
0x80110003	AddContoller時のConnオプションの値が有効範囲外です.
0x80110004	AddContoller時のTimeoutオプションの指定方法が誤っています.
0x80110005	AddContoller時のTimeoutオプションの値が有効範囲外です.
0x80110006	拡張コマンド実行時の引数がコマンドの仕様とあっていません.
0x80110007	拡張コマンド実行時の機器から想定外のデータを受信しました.

また，本プロバイダは，デバイスからのエラーコードを「0x8010****」でマスクして返します．以下のデバイスエラーコードが存在します．（表 5-2 デバイスエラーコード表参照）

表 5-2 デバイスエラーコード表

エラー番号	説明
0x80100000	コマンドエラー.
0x80100001	状態エラー.
0x80100020	コマンド長エラー.
0x80100021	パラメータ数エラー.
0x80100022	パラメータ範囲外エラー.
0x80100088	タイムアウトエラー.
0x80100099	その他.

6. 付録

付録A. 通信プロトコルコマンド対応表

CaoControllre::Execute メソッド	デバイス通信プロトコルコマンド
SetMode(通常モード設定時)	RO
SetMode(通信モード設定時)	QO
GetCalcData	Ma[2]
SetTiming	Tp[3]
SetZero(ON 設定時)	Va
SetZero(OFF 設定時)	Wa
SetReset	VR
SetPanelLock	KL
SetProgramNo	PW
GetProgramNo	PR
GetFigureData	DO
ClearFigureData	DQ
StartDataStorage	AS
StopDataStorage	AP
ClearDataStorage	AQ
GetDataStorageData	AO
GetDataStorageStatus	AN
SetPanel	DC
GetPanel	DR
SetTolerance	SW,LM
GetTolerance	SR,LM
SetAbleMode	SW,HA,M
GetAbleMode	SR,HA,M
SetAbleMinMax	SW,HA,R
GetAbleMinMax	SR,HA,R
SetMeasureMode	SW,HB
GetMeasureMode	SR,HB
SetNumAlarm	SW,HC,N

[2] a : OUT 指定
[3] p : ON/OFF 指定

SetAlarmLevel	SW,HC,L
GetAlarm	SR,HC
StartABLE	SW,HD,S
StopABLE	SW,HD,P
CancelABLE	SW,HD,C
SetReflectionMode	SW,HE
GetReflectionMode	SR,HE
SetCalcMethod	SW,OA
GetCalcMethod	SR,OA
SetScaling	SW,OB
GetScaling	SR,OB
SetFilter	SW,OC
GetFilter	SR,OC
SetTriggerMode	SW,OE,M
GetTriggerMode	SR,OE,M
SetOffset	SW,OF
GetOffset	SR,OF
SetAnalogScaling	SW,OH
GetAnalogScaling	SR,OH
SetCalcMode	SW,OD
GetCalcMode	SR,OD
SetDisplayUnit	SW,OG
GetDisplayUnit	SR,OG
SetAnalogThrough	SW,OI
GetAnalogThrough	SR,OI
SetDataStorage	SW,CI
GetDataStorage	SR,CI
SetSamplingCycle	SW,CA
GetSamplingCycle	SR,CA
SetMutualInterPrev	SW,CB
GetMutualInterPrev	SR,CB
SetTimingSync	SW,CC
GetTimingSync	SR,CC
SetToleCompOutputFormat	SW,CD
GetToleCompOutputFormat	SR,CD

SetStorobeTime	SW,CE
GetStorobeTime	SR,CE

CaoVariable::Value プロパティ	デバイス通信プロトコルコマンド
@CALCDATA::get	MO