エスペック株式会社
espec TempHumi プロバイダ

Version 1.0.0

ユーザーズ ガイド

October 23, 2019

備考：

【改版履歴】

バージョン	日付	内容
1.0.0	2019-10-23	初版.

【動作確認機種】

機種	バージョン	注意事項
STH-120	-	-
SH-642	-	-
SU-642	-	-

目次

1. はじめに

本書は，エスペック株式会社の小型環境試験器シリーズ(SU-***, SH-***)，ミニサブゼロシリーズ(MC-***)，ライトスペック恒温（恒湿）器シリーズ(LH-***, LHL-***, LHU-***, LU-***)，熱風乾燥器(LC-***, LG-***)，真空乾燥器(LCV-***)，小型高温チャンバーシリーズ(ST-***, STH-***)，安定性試験器(CSH)，安定性試験室(CWH)，低温恒温恒湿器(CRH)のデバイスに対してデータの取得及び設定をするプロバイダのユーザーズガイドです．対応機種については付録にあります「表 6-3 RS-485 対応機種一覧」及び「表 6-4 Ethernet 対応機種一覧」を参照してください．以降エスペック株式会社の対応機種を TempHumi デバイスと呼称します．また，標準装備が Ethernet のデバイスを TempHumi Ethernet デバイス，標準装備が RS-485 のデバイスを TempHumi RS-485 デバイスと呼称します．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 espec TempHumi プロバイダと呼称します．

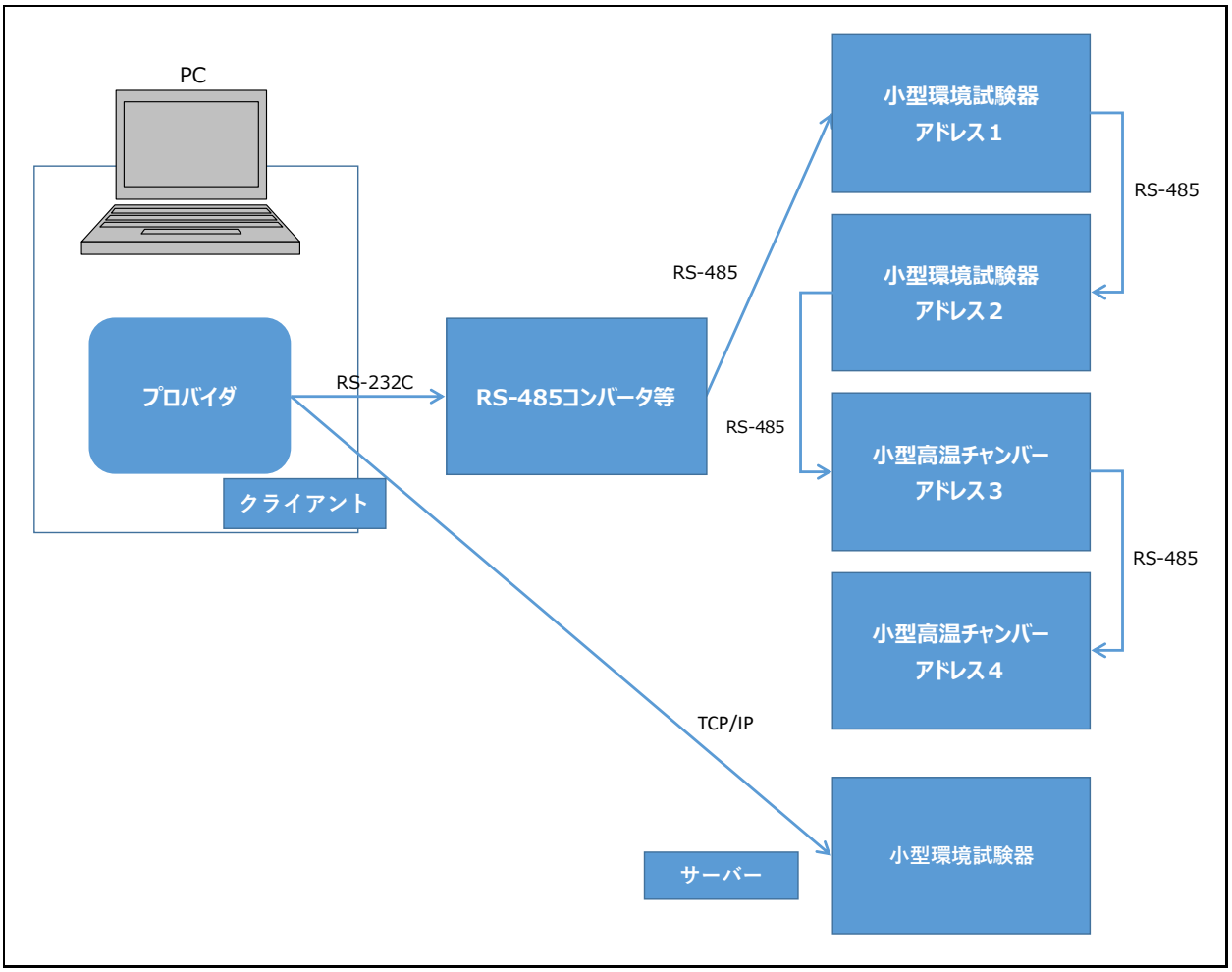


図 1-1 構成図

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．

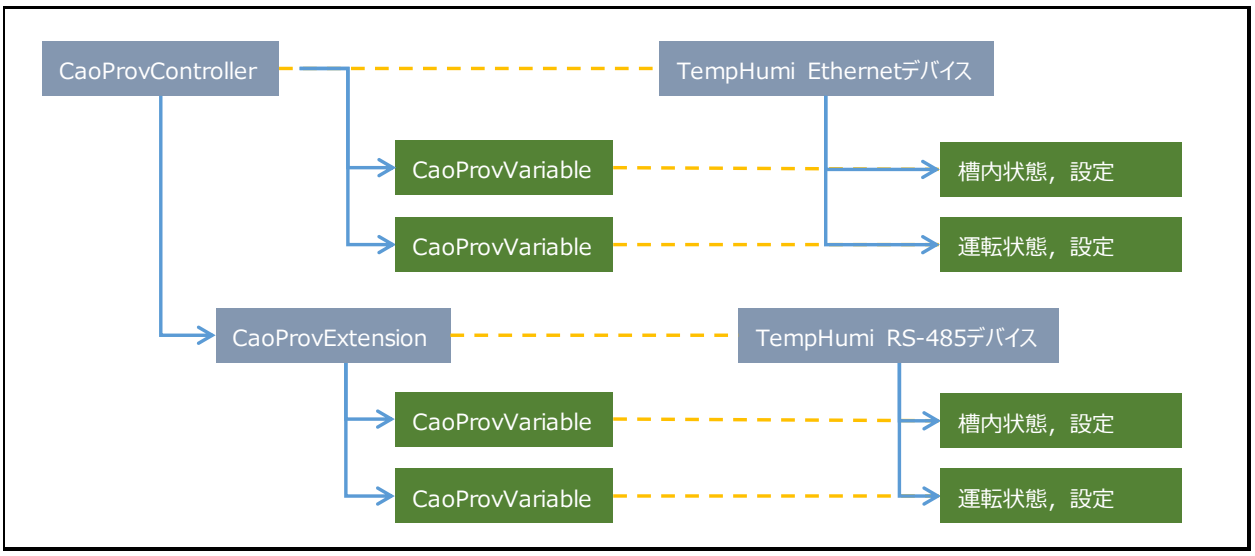


図 1-2 espec TempHumi プロバイダと TempHumi デバイスとの対応図

1.1. 参考となる情報源

espec TempHumi プロバイダは，エスペック株式会社から提供されたユーザーズマニュアルを参考に開発しています．以降このマニュアルを TempHumi RS-485 マニュアルならびに TempHumi Ethernet マニュアルと呼称します．

以下参考資料

■ユーザーズマニュアル 通信機能 RS-485 （資料番号 4009204000631）

■ユーザーズマニュアル 通信機能 Ethernet （資料番号 4000104005150）

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. TempHumi デバイスとクライアント PC との接続

2.1.1. シリアル通信で接続

TempHumi RS-485 デバイスとクライアント PC は RS-232C を介してシリアル通信にて接続されます. また, データ転送は標準モード及びデリミタは CR+LF で対応しています. 接続する TempHumi RS-485 デバイスでは TempHumi RS-485 マニュアルの「第２章　環境設定」を参照して以下の通りに設定して下さい.

設定項目	設定値
RS-485 転送モード設定	標準モード
RS-485 デリミタ設定	CR+LF

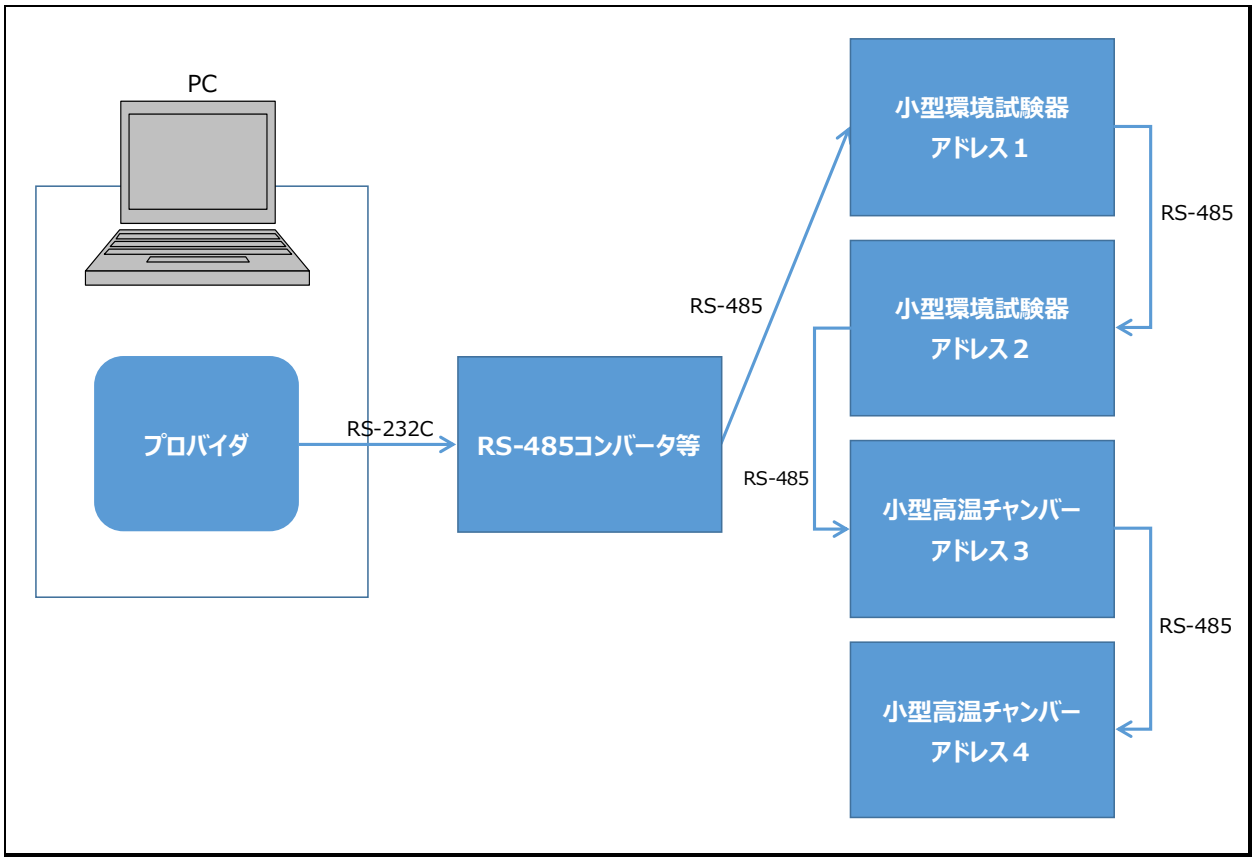


図 2-1 シリアル通信時の構成図

2.1.2. Ethernet 通信で接続

TempHumi Ethernet デバイスとクライアント PC は Ethernet を介して TCP/IP 通信にて接続されます. 尚, 接続する TempHumi Ethernet デバイスではデリミタは CR+LF 固定ですので, 環境設定で設定する

必要はありません.

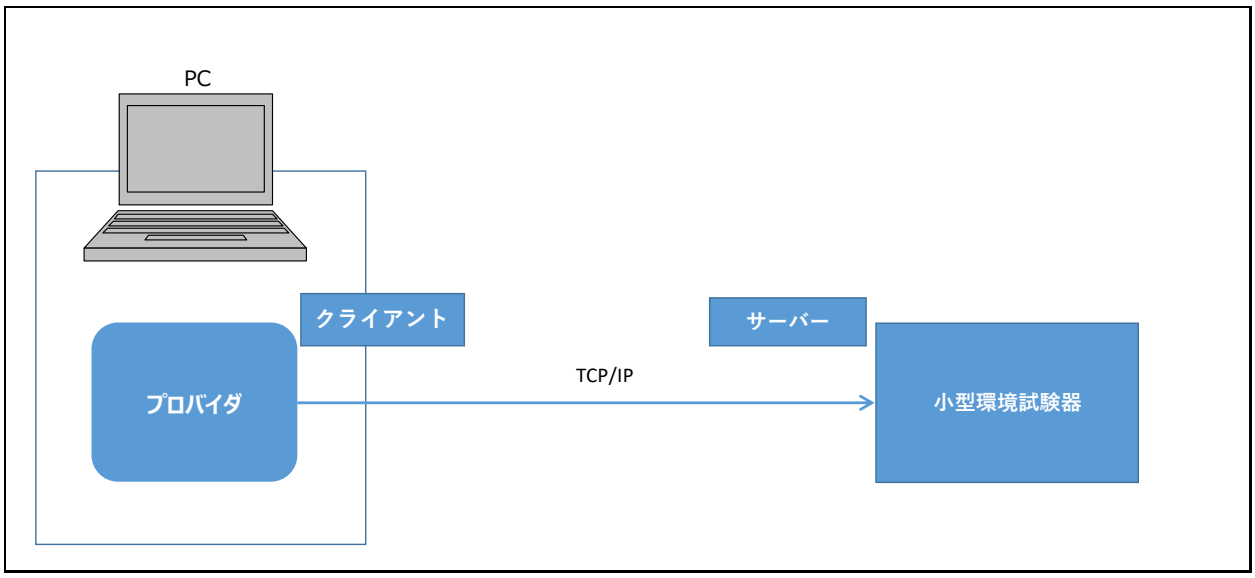


図 2-2 Ethernet 通信時の構成図

2.2. クライアント PC 開発環境のセットアップ

2.2.1. espec TempHumi プロバイダの手動インストール

espec TempHumi プロバイダを使用するために手動でインストールする場合は，下記レジストリ登録を行う必要があります．レジストリ登録を行う場合は，管理者権限でコマンドプロンプトを起動し，regsvr32 コマンドを実行してください.

表 2-1 espec TempHumi プロバイダ

ファイル名	CaoProvespecTempHumi.dll
ProgID	CaoProv.espec.TempHumi
レジストリ登録	regsvr32 CaoProvespecTempHumi.dll
レジストリ登録の抹消	regsvr32 /u CaoProvespecTempHumi.dll

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.9
CaoController				
	VariableNames	P	接続可能な変数名リストの取得	P.11
	AddExtension	M	拡張ボードオブジェクトの追加	P.11
	AddVariable	M	変数オブジェクトの追加	P.12
	Execute	M	拡張コマンドの実行	P.12
CaoExtenstion				
	ID	P	ID の取得/設定	P.13
	VariableNames	P	接続可能な変数名リストの取得	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.14
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

AddController メソッド実行時に渡されたパラメーターを参照し，該当する TempHumi デバイスと接続を行います．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.espec.TempHumi",       // プロバイダ名 (固定)
        "<マシン名>",                   // プロバイダ実行マシン名
        "<オプション>"                  // オプション文字列
)
```

[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．

オプション

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	○	TempHumi デバイスと接続するためのシリアルまたはイーサネット接続オプションを指定します. 詳細は 3.2.1.1.1 を参照して下さい.	--	--
Timeout	--	接続/送受信タイムアウトを ms 単位で指定します.	1 - 4294967295	500

使用例

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

Set caoEng = New CaoEngine
Set caoWor = caoEng.Workspaces.Item(0)
Set caoCon = caoWor.AddController("Controller", _
                                  "CaoProv.espec.TempHumi", _
                                  "", _
                                  "Conn=com:1")
```

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ー文字列を示します. ここで中括弧("[]")内は省略可能なことを, 各パラメーター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.

TempHumi RS-485 デバイスを接続する場合(シリアル通信)

TempHumi RS-485 デバイスを複数接続する場合(図 2-1 シリアル通信時の構成図を参照), 各デバイスの RS-485 通信速度設定, RS-485 ストップビット設定, RS-485 データビット設定, RS-485 パリティビット設定は同じにして下さい. 各デバイスの設定は, TempHumi RS-485 マニュアルの「第２章　環境設定」及び「第５章　仕様」を参照して下さい.

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

<COM Port>　　: COM ポート番号.

<BaudRate>　　: ボーレート. 4800, 9600, 19200

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

<DataBits> : データビット. '7'-7bit, '8'-8bit
<StopBits> : ストップビット. '1'-1bit, '2'-2bit

TempHumi Ethernet デバイスを接続する場合(TCP/IP 通信)

TempHumi Ethernet デバイスは 57732 のポート番号を利用しています.

"Conn=Eth:<IP>[:<Port>]"
<IP> : 接続先 IP アドレス.
<Port> : 接続先ポート. 57732

3.2.2. CaoController クラス

本クラスは, シリアル通信で TempHumi RS-485 デバイスと接続する際の, RS-485 コンバータに対応します. また, TCP/IP 通信で接続する際は, 各 TempHumi Ethernet デバイスに対応します.

3.2.2.1. VariableNames プロパティ

変数名リストを取得します. 本プロパティ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

```
' 接続
Call Connect
' 変数名リスト取得
Dim variables as Variant
variables = caoCon.VariableNames
' 切断
Call Disconnect
```

3.2.2.2. AddExtension メソッド

TempHumi RS-485 デバイスを接続する場合には, CaoController に各デバイスに対応する CaoExtension オブジェクトを追加する必要があります. また, デバイスにて設定されているアドレス番号をオプション文字列で指定して下さい. 以下に, AddExtension メソッドの仕様を示します.

書式

```
AddExtension
(
        "<拡張ボード名>",       // 拡張ボード名(任意)
        "<オプション>"          // オプション文字列
)
```

オプション

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
Addr	○	接続するデバイスのアドレスを指定します. アドレスは接続されている複数のデバイスを識別するために必要です. 各デバイスには一意のアドレスを設定して下さい. (図 2-1 シリアル通信時の構成図のアドレスを参照)	1 - 16	--

使用例

```
    ' 接続
    Call Connect
    ' 拡張ボードの追加
    Dim caoExt As CaoExtension
    Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
    Call caoCon.Extensions.Remove(caoExt.Index)
    Set caoExt = Nothing
    ' 切断
    Call Disconnect
```

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 3.4.1 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 3.3.1 に示すコマンドのみ実行可能です. 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. CaoExtension クラス

本プロバイダは，シリアル接続された CaoController クラスから，接続する各 TempHumi RS-485 デバイスにアクセスするクラスです．

3.2.3.1. ID プロパティ

拡張ボードのアドレスを取得します．

データ型

型説明	
VT_I4	アドレスが返却されます．

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' ID（アドレス）取得
Dim address Long
address = caoExt.ID
```

3.2.3.2. VariableNames プロパティ

変数名リストを取得します．本プロパティ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

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' 変数名リスト取得
Dim variables as Variant
variables = caoExt.VariableNames
```

3.2.3.3. AddVariable メソッド

CaoExtension に変数オブジェクトを追加します．変数名には 3.4.2 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)
 )
```

3.2.3.4. Execute メソッド

CaoExtension 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 3.3.2 に示すコマンドのみ実行可能です. 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)
)
```

3.2.4. CaoVariable クラス

3.2.4.1. Value プロパティ

接続したデバイスからデータを取得/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

各クラスで使用可能な拡張コマンド一覧を定義します.

3.3.1. CaoController クラスの拡張コマンド

CaoController クラスの拡張コマンドは, イーサネット接続されている場合のみ有効です. 使用例は各コマンドの詳細で記述しています.

表 3-2 CaoController クラスの拡張コマンド一覧

コマンド	説明	参照
機器の設定値関連コマンド		
GetROM	温調器の ROM バージョンを取得します.	P.16
GetROMDISP	表示器の ROM バージョンを取得します.	P.16
GetMASK	割り込みマスクを取得します.	P.17
SetMASK	割り込みマスクを設定します.	P.17

コマンド	説明	参照
GetTIMERMON	起動されているタイマーを取得します.	P. 18
GetTIMERSET	設定されているタイマーを取得します.	P. 19
GetTIMER0	クイックタイマーの設定を取得します.	P. 20
SetTIMER0	クイックタイマーの設定を設定します.	P. 20
GetTIMER1	運転開始タイマーの設定を取得します.	P. 21
SetTIMER1	運転開始タイマーの設定を設定します.	P. 21
GetTIMER2	運転停止タイマーの設定を取得します.	P. 22
SetTIMER2	運転停止タイマーの設定を設定します.	P. 22
GetKEYPROTECT	プロテクト設定を取得します.	P. 23
SetKEYPROTECT	プロテクト設定を設定します.	P. 23
GetTYPE	チャンバータイプを取得します.	P. 23
GetTEMP	温度パラメーターを取得します.	P. 24
SetTEMP	温度パラメーターを設定します.	P. 25
SetSTEMP	温度パラメーターの設定値を設定します.	P. 25
SetHTEMP	温度パラメーターの上限値を設定します.	P. 25
SetLTEMP	温度パラメーターの下限値を設定します.	P. 26
GetHUMI	湿度パラメーターを取得します.	P. 26
SetHUMI	湿度パラメーターを設定します.	P. 26
SetSHUMI	湿度パラメーターの設定値を設定します.	P. 27
SetHHUMI	湿度パラメーターの上限値を設定します.	P. 27
SetLHUMI	湿度パラメーターの下限値を設定します.	P. 28
GetSET	冷凍機の制御値を取得します.	P. 28
SetSET	冷凍機の制御値を設定します.	P. 28
GetCONSTANTSETTEMP	定値設定の温度設定値を取得します.	P. 29
GetCONSTANTSETHUMI	定値設定の湿度設定値を取得します.	P. 29
GetCONSTANTSETREF	定値設定の冷凍機設定値を取得します.	P. 30
GetSYSTEMSETPTS	試料温度モニター機能を取得します.	P. 30
GetSYSTEMSETPTC	試料温度制御機能を取得します.	P. 31
GetSYSTEMSETPTCOPT	試料温度制御機能オプションを取得します.	P. 31
SetTIMERMERASE	指定されたタイマーの設定を削除します.	P. 31

コマンド		説明	参照
	SetTIMERON	指定されたタイマーを起動します.	P.32
	SetTIMEROFF	指定されたタイマーを停止します.	P.32
機器の状態関連コマンド			
	GetDATE	内部カレンダーの日付を取得します.	P.33
	SetDATE	内部カレンダーの日付を設定します.	P.33
	GetTIME	内部カレンダーの現在時刻を取得します.	P.33
	SetTIME	内部カレンダーの現在時刻を設定します.	P.34
	GetSRQ	割り込みを取得します.	P.34
	SetSRQ	割り込みをリセットします.	P.35
	GetALARM	アラーム状態を取得します.	P.35
	GetMODE	運転モードを取得します.	P.35
	SetMODE	運転モードを設定します.	P.36
	GetMODEDETAIL	詳細な運転モードを取得します.	P.36
	GetMONDETAIL	詳細な槽内の状態を取得します.	P.37
	GetHEATER	ヒーターの状態を取得します.	P.38
プログラム関連コマンド			
	GetRUNPRGDATA	リモートプログラムデータを取得します.	P.39
	SetRUNPRGDATA	リモートプログラムデータを設定します.	P.40

3.3.1.1. 機器の設定値関連コマンド

機器に設定されている設定値を取得/設定するコマンドです.

3.3.1.1.1. GetROM コマンド

温調器の ROM バージョンを取得します.

項目	型説明	
戻り値	VT_BSTR	ROM タイプ　ROM バージョン

使用例

```
' GetROMコマンド実行
Dim val As String
val = caoCon.Execute("GetROM")
```

3.3.1.1.2. GetROMDISP コマンド

表示器の ROM バージョンを取得します.

項目	型説明	
戻り値	VT_BSTR	ROM タイプ　ROM バージョン

使用例

```
' GetROMDISPコマンド実行
Dim val As String
val = caoCon.Execute("GetROMDISP")
```

3.3.1.1.3. GetMASK コマンド

割り込みマスクを取得します.

項目	型説明	
戻り値	VT_UI1	割り込みマスク 以下はビット(最上位ビット順)の割付です. 1 ビット目　　：未使用 2 ビット目　　：装置でアラーム発生時 3 ビット目　　:リモートプログラム運転で 1 ステップの運転が終了時 4 ビット目　　：電源オフ/電源オンに変化した時 5 ビット目　　：未使用 6 ビット目　　：未使用 7 ビット目　　：GPIB 通信機能で予約 8 ビット目　　：未使用 各ビットに割り付けられた事象が発生した場合，対するビットが 1 ならば割り込みが発生します.

使用例

```
' GetMASKコマンド実行
Dim val As Byte
val = caoCon.Execute("GetMASK")
```

3.3.1.1.4. SetMASK コマンド

割り込みマスクを設定します.

項目	型説明

項目	型説明	
引数	VT_UI1	割り込みマスクを指定します. 以下はビット(最上位ビット順)の割付です. 1 ビット目 ：未使用 2 ビット目 ：装置でアラーム発生時 3 ビット目 :リモートプログラム運転で 1 ステップの運転が終了時 4 ビット目 ：電源オフ/電源オンに変化した時 5 ビット目 ：未使用 6 ビット目 ：未使用 7 ビット目 ：GPIB 通信機能で予約 8 ビット目 ：未使用 各ビットに割り付けられた事象が発生した場合, 対するビットが 1 ならば割り込みが発生します.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetMASKコマンド実行
Dim result As String
result = caoCon.Execute("SetMASK", 64)
```

3.3.1.1.5. GetTIMERMON コマンド

起動されているタイマーを取得します.

項目	型説明		
戻り値	VT_EMPTY		起動されているタイマーはありません
	または		
	VT_ARRAY | VT_I4		最大 3 個までの起動されているタイマー番号
	i	VT_I4	タイマー番号 タイマー番号は, 以下のようになります. ・0 ：クイックタイマー ・1 ：開始タイマー ・2 ：停止タイマー **※添字 i ： 起動されているタイマー数分(0 - 2)**

使用例

```
' GetTIMERMONコマンド実行
Dim values As Variant
values = caoCon.Execute("GetTIMERMON")

' 起動されているタイマーの数だけ取得
If Not IsEmpty(values) Then
    Dim i As Long
    For i = 0 To UBound(values)
        Dim val As Long
        val = values(i)
    Next i
End If
```

3.3.1.1.6. GetTIMERSET コマンド

設定されているタイマーを取得します.

項目	型説明		
戻り値	VT_EMPTY		設定されているタイマーはありません
	または		
	VT_ARRAY | VT_I4		最大 3 個までの設定されているタイマー番号
	i	VT_I4	タイマー番号 タイマー番号は，以下のようになります. ・0　　：クイックタイマー ・1　　：開始タイマー ・2　　：停止タイマー **※添字 i： 設定されているタイマー数分(0 - 2)**

使用例

```
' GetTIMERSETコマンド実行
Dim values As Variant
values = caoCon.Execute("GetTIMERSET")

' 設定されているタイマーの数だけ取得
If Not IsEmpty(values) Then
    Dim i As Long
    For i = 0 To UBound(values)
        Dim val As Long
        val = values(i)
    Next i
End If
```

3.3.1.1.7. GetTIMER0 コマンド

クイックタイマーの設定を取得します.

項目	型説明	
戻り値	VT_BSTR	運転モード,設定時間　または　停止モード,設定時間 運転モードは表 3-3 運転モードの詳細を参照して下さい. 停止モードは表 3-4 停止モードの詳細を参照して下さい.

表 3-3 運転モードの詳細

設定内容	応答データ表示	応答例
プログラム運転	"RUN,RAM:プログラム番号,STEPxx"	"RUN,RAM:1,STEP1"
定値運転	"CONSTANT"	"CONSTANT"

プログラム番号の範囲は, 1～8 になります.

表 3-4 停止モードの詳細

設定内容	応答データ表示	応答例
停止状態	"STANDBY"	"STANDBY"
パネル電源 OFF	"OFF"	"OFF"

使用例

```
' GetTIMER0コマンド実行
Dim val As String
val = caoCon.Execute("GetTIMER0")
```

3.3.1.1.8. SetTIMER0 コマンド

クイックタイマーの設定を設定します.

項目	型説明	
引数	VT_BSTR	起動時間,運転モード　または　起動時間,停止モードを指定します. 運転モードは表 3-3 運転モードの詳細を参照して下さい. 停止モードは表 3-4 停止モードの詳細を参照して下さい.
戻り値	VT_BSTR	コマンドが正常に実行された場合, 機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMER0コマンド実行
Dim result As String
result = caoCon.Execute("SetTIMER0", "10:00,CONSTANT")
```

3.3.1.1.9. GetTIMER1 コマンド

運転開始タイマーの設定を取得します.

項目	型説明	
戻り値	VT_BSTR	タイマー番号,起動モード,運転モード 起動モードは表 3-5 起動モードの詳細を参照して下さい. 運転モードは表 3-3 運転モードの詳細を参照して下さい.

表 3-5 起動モードの詳細

設定内容	応答データ表示	応答例
1 回実行モード	"MODE1,起動日,起動時間"	"MODE1,12.03/04,10:00"
毎週実行モード	"MODE2,起動曜日,起動時間" 起動曜日は表 3-6 起動曜日の詳細を参照して下さい.	"MODE2,SAT,23:00"
毎日実行モード	"MODE3,起動時間"	"MODE3,0:00"

表 3-6 起動曜日の詳細

曜日	応答データ表示
月曜日	"MON"
火曜日	"TUE"
水曜日	"WED"
木曜日	"THU"
金曜日	"FRI"
土曜日	"SAT"
日曜日	"SUN"

起動曜日は, 複数指定が可能です.
複数指定時は, "/"を使用します.
例："MODE2,MON/SAT,10:00"

使用例

```
' GetTIMER1コマンド実行
Dim val As String
val = caoCon.Execute("GetTIMER1")
```

3.3.1.1.10. SetTIMER1 コマンド

運転開始タイマーの設定を設定します.

項目	型説明	
引数	VT_BSTR	起動モード,運転モードを指定します. 起動モードは表 3-5 起動モードの詳細を参照して下さい. 運転モードは表 3-3 運転モードの詳細を参照して下さい.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMER1コマンド実行
Dim result As String
result = caoCon.Execute("SetTIMER1", "MODE1,12.03/04,10:00,CONSTANT")
```

3.3.1.1.11. GetTIMER2 コマンド

運転停止タイマーの設定を取得します.

項目	型説明	
戻り値	VT_BSTR	タイマー番号,起動モード,停止モード 起動モードは表 3-5 起動モードの詳細を参照して下さい. 停止モードは表 3-4 停止モードの詳細を参照して下さい.

使用例

```
' GetTIMER1コマンド実行
Dim val As String
val = caoCon.Execute("GetTIMER2")
```

3.3.1.1.12. SetTIMER2 コマンド

運転停止タイマーの設定を設定します.

項目	型説明	
引数	VT_BSTR	起動モード,停止モードを指定します. 起動モードは表 3-5 起動モードの詳細を参照して下さい. 停止モードは表 3-4 停止モードの詳細を参照して下さい.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMER2コマンド実行
Dim result As String
result = caoCon.Execute("SetTIMER2", "MODE2,SAT,10:00,OFF")
```

3.3.1.1.13. GetKEYPROTECT コマンド

プロテクト設定を取得します.

項目	型説明	
戻り値	VT_BSTR	キープロテクト状態 以下のいずれかが取得されます. ・ ON : キープロテクト ON ・ OFF : キープロテクト OFF キープロテクト ON とは, 設定変更プロテクト, 運転操作プロテクトのいずれかが ON の状態です

使用例

```
' GetKEYPROTECTコマンド実行
Dim val As String
val = caoCon.Execute("GetKEYPROTECT")
```

3.3.1.1.14. SetKEYPROTECT コマンド

プロテクト設定を設定します. 設定変更プロテクトと運転操作プロテクトを分けて設定することはできません.

項目	型説明	
引数	VT_BSTR	キープロテクトの状態を以下のいずれかにて指定します. ・ ON : キープロテクト ON ・ OFF : キープロテクト OFF
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetKEYPROTECTコマンド実行
Dim result As String
result = caoCon.Execute("SetKEYPROTECT", "OFF")
```

3.3.1.1.15. GetTYPE コマンド

チャンバータイプを取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_BSTR	乾球センサタイプ

項目	型説明		
	1	VT_BSTR	湿球センサタイプ ※温度のみタイプの装置の場合は，取得データが省略されているので VT_EMPTY となります
	2	VT_BSTR	温度調整器タイプ
	3	VT_R8	温度上限

使用例

```
' GetTYPEコマンド実行
Dim values As Variant
values = caoCon.Execute("GetTYPE")

' 乾球センサタイプ
Dim drySensor As String
drySensor = values(0)

' 湿球センサタイプ
If Not IsEmpty(values(1)) Then
    Dim wetSensor As String
    wetSensor = values(1)
End If

' 温度調整器タイプ
Dim tempCon As String
tempCon = values(2)

' 温度上限
Dim tempUplimit As Double
tempUplimit = values(3)
```

3.3.1.1.16. GetTEMP コマンド

温度パラメーターを取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_R8		
	0	VT_R8	測定値
	1	VT_R8	設定値
	2	VT_R8	上限値
	3	VT_R8	下限値

使用例

```
' GetTEMPコマンド実行
Dim values() As Double
values = caoCon.Execute("GetTEMP")
```

3.3.1.1.17. SetTEMP コマンド

温度パラメーターを設定します.

項目	型説明		
引数	VT_ARRAY | VT_R8		
	0	VT_R8	設定値を指定します.
	1	VT_R8	上限値を指定します.
	2	VT_R8	下限値を指定します.
戻り値	VT_BSTR		コマンドが正常に実行された場合, 機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTEMPコマンド実行
Dim result As String
result = caoCon.Execute("SetTEMP", Array(23.0, 100.0, -40.0))
```

3.3.1.1.18. SetSTEMP コマンド

温度パラメーターの設定値を設定します.

項目	型説明	
引数	VT_R8	設定値を指定します.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetSTEMPコマンド実行
Dim result As String
result = caoCon.Execute("SetSTEMP", 23.0)
```

3.3.1.1.19. SetHTEMP コマンド

温度パラメーターの上限値を設定します.

項目	型説明	
引数	VT_R8	上限値を指定します.

項目	型説明	
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetHTEMPコマンド実行
Dim result As String
result = caoCon.Execute("SetHTEMP", 100.0)
```

3.3.1.1.20. SetLTEMP コマンド

温度パラメーターの下限値を設定します.

項目	型説明	
引数	VT_R8	下限値を指定します.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetLTEMPコマンド実行
Dim result As String
result = caoCon.Execute("SetLTEMP", -40.0)
```

3.3.1.1.21. GetHUMI コマンド

湿度パラメーターを取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_I4		
	0	VT_I4	測定値
	1	VT_I4	設定値
	2	VT_I4	上限値
	3	VT_I4	下限値

使用例

```
' GetHUMIコマンド実行
Dim values() As Long
values = caoCon.Execute("GetHUMI")
```

3.3.1.1.22. SetHUMI コマンド

湿度パラメーターを設定します.

項目	型説明		
引数	VT_ARRAY | VT_I4		
	0	VT_I4	設定値を指定します.
	1	VT_I4	上限値を指定します.
	2	VT_I4	下限値を指定します.
戻り値	VT_BSTR		コマンドが正常に実行された場合，機器からの応答文字列がそのまま格納されます.

使用例

```
' SetHUMIコマンド実行
Dim result As String
result = caoCon.Execute("SetHUMI", Array(23, 100, 0))
```

3.3.1.1.23. SetSHUMI コマンド

湿度パラメーターの設定値を設定します.

項目	型説明	
引数	VT_I4	設定値を指定します.
戻り値	VT_BSTR	コマンドが正常に実行された場合，機器からの応答文字列がそのまま格納されます.

使用例

```
' SetSHUMIコマンド実行
Dim result As String
result = caoCon.Execute("SetSHUMI", 85)
```

3.3.1.1.24. SetHHUMI コマンド

湿度パラメーターの上限値を設定します.

項目	型説明	
引数	VT_I4	上限値を指定します.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetHHUMIコマンド実行
Dim result As String
result = caoCon.Execute("SetHHUMI", 100)
```

3.3.1.1.25. SetLHUMI コマンド

湿度パラメーターの下限値を設定します.

項目	型説明	
引数	VT_I4	下限値を指定します.
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetLHUMIコマンド実行
Dim result As String
result = caoCon.Execute("SetLHUMI", 0)
```

3.3.1.1.26. GetSET コマンド

冷凍機の制御値を取得します.

項目	型説明	
戻り値	VT_BSTR	制御値 以下のいずれかが取得されます. ・ REF0 ～ REF8 : 手動設定 ・ REF9 : 自動設定

使用例

```
' GetSETコマンド実行
Dim val As String
val = caoCon.Execute("GetSET")
```

3.3.1.1.27. SetSET コマンド

冷凍機の制御値を設定します.

項目	型説明	
引数	VT_BSTR	制御値を以下のいずれかにて指定します. ・ REF0 : 手動 OFF ・ REF1 : 手動 ON ・ REF2 ～ REF9 : 自動
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetSETコマンド実行
Dim result As String
result = caoCon.Execute("SetSET", "REF9")
```

3.3.1.1.28. GetCONSTANTSETTEMP コマンド

定値設定の温度設定値を取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_R8	温度設定値
	1	VT_BSTR	制御許可 以下のいずれかが取得されます. ・ ON : 温度制御有効 ・ OFF : 温度制御無効

使用例

```
' GetCONSTANTSETTEMPコマンド実行
Dim values As Variant
values = caoCon.Execute("GetCONSTANTSETTEMP")

' 温度設定値
Dim temp As Double
temp = values(0)

' 制御許可
Dim permission As String
permission = values(1)
```

3.3.1.1.29. GetCONSTANTSETHUMI コマンド

定値設定の湿度設定値を取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_I4	湿度設定値
	1	VT_BSTR	制御許可 以下のいずれかが取得されます. ・ ON : 湿度制御有効 ・ OFF : 湿度制御無効

使用例

```
' GetCONSTANTSETHUMIコマンド実行
Dim values As Variant
values = caoCon.Execute("GetCONSTANTSETHUMI")

' 湿度設定値
Dim humi As Long
humi = values(0)

' 制御許可
Dim permission As String
permission = values(1)
```

3.3.1.1.30. GetCONSTANTSETREF コマンド

定値設定の冷凍機設定値を取得します.

項目	型説明	
戻り値	VT_BSTR	設定値 以下のいずれかが取得されます. ・ AUTO : 自動設定 ・ OFF : 手動設定 OFF ・ 100 : 手動設定 ON

使用例

```
' GetCONSTANTSETREFコマンド実行
Dim val As String
val = caoCon.Execute("GetCONSTANTSETREF")
```

3.3.1.1.31. GetSYSTEMSETPTS コマンド

試料温度モニター機能を取得します.

項目	型説明	
戻り値	VT_BSTR	搭載情報 以下のいずれかが取得されます. ・ MON : 試料温度モニター機能搭載 ・ OFF : 試料温度モニター機能未搭載

使用例

```
' GetSYSTEMSETPTSコマンド実行
Dim val As String
val = caoCon.Execute("GetSYSTEMSETPTS")
```

3.3.1.1.32. GetSYSTEMSETPTC コマンド

試料温度制御機能を取得します.

項目	型説明	
戻り値	VT_BSTR	搭載情報 以下のいずれかが取得されます. ・ ON : 試料温度制御機能搭載 ・ OFF : 試料温度制御機能未搭載

使用例

```
' GetSYSTEMSETPTCコマンド実行
Dim val As String
val = caoCon.Execute("GetSYSTEMSETPTC")
```

3.3.1.1.33. GetSYSTEMSETPTCOPT コマンド

試料温度制御機能オプションを取得します.

項目	型説明	
戻り値	VT_BSTR	搭載情報 以下のいずれかが取得されます. ・ OFF : 試料温度未搭載 ・ M : 試料温度搭載(モニターのみ) ・ C : 試料温度搭載(試料温度制御のみ) ・ MC : 試料温度搭載(モニター&試料温度制御)

使用例

```
' GetSYSTEMSETPTCOPTコマンド実行
Dim val As String
val = caoCon.Execute("GetSYSTEMSETPTCOPT")
```

3.3.1.1.34. SetTIMERMERASE コマンド

指定されたタイマーの設定を削除します.

項目	型説明	
引数	VT_I4	削除したいタイマー番号を指定します. タイマー番号は, 以下のようになります. ・0 : クイックタイマー ・1 : 開始タイマー ・2 : 停止タイマー

項目	型説明	
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMERMERASEコマンド実行
Dim result As String
result = caoCon.Execute("SetTIMERMERASE", 1)
```

3.3.1.1.35. SetTIMERON コマンド

指定されたタイマーを起動します.

項目	型説明	
引数	VT_I4	起動したいタイマー番号を指定します. タイマー番号は, 以下のようになります. ・0 : クイックタイマー ・1 : 開始タイマー ・2 : 停止タイマー
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMERONコマンド実行
Dim result As String
result = caoCon.Execute("SetTIMERON", 1)
```

3.3.1.1.36. SetTIMEROFF コマンド

指定されたタイマーを停止します.

項目	型説明	
引数	VT_I4	停止したいタイマー番号を指定します. タイマー番号は, 以下のようになります. ・0 : クイックタイマー ・1 : 開始タイマー ・2 : 停止タイマー
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMEROFFコマンド実行
Dim result As String
result = caoCon.Execute("SetTIMEROFF", 2)
```

3.3.1.2. 機器の状態関連コマンド

機器の状態を取得/設定するコマンドです.

3.3.1.2.1. GetDATE コマンド

内部カレンダーの日付を取得します.

項目	型説明	
戻り値	VT_BSTR	日付 YY.MM/DD

使用例

```
' GetDATEコマンド実行
Dim val As String
val = caoCon.Execute("GetDATE")
```

3.3.1.2.2. SetDATE コマンド

内部カレンダーの日付を設定します.

項目	型説明	
引数	VT_BSTR	日付を指定します. YY.MM/DD
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetDATEコマンド実行
Dim result As String
result = caoCon.Execute("SetDATE", "12.03/04")
```

3.3.1.2.3. GetTIME コマンド

内部カレンダーの現在時刻を取得します.

項目	型説明	
戻り値	VT_BSTR	現在時刻 hh:mm:ss

使用例

```
' GetTIMEコマンド実行
Dim val As String
val = caoCon.Execute("GetTIME")
```

3.3.1.2.4. SetTIME コマンド

内部カレンダーの現在時刻を設定します.

項目	型説明	
引数	VT_BSTR	現在時刻を指定します. hh:mm:ss
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetTIMEコマンド実行
Dim result As String
result = caoCon.Execute("SetTIME", "18:00:00")
```

3.3.1.2.5. GetSRQ コマンド

割り込みを取得します.

項目	型説明	
戻り値	VT_UI1	割り込みの状態 以下はビット(最上位ビット順)の割付です. 1 ビット目 : 未使用 2 ビット目 : 装置でアラーム発生時 3 ビット目 :リモートプログラム運転で 1 ステップの運転が終了時 4 ビット目 : 電源オフ/電源オンに変化した時 5 ビット目 : 未使用 6 ビット目 : 未使用 7 ビット目 : GPIB 通信機能で予約 8 ビット目 : 未使用 各ビットに割り付けられた事象が発生した場合, 対するビットが 1 ならば割り込みが発生します.

使用例

```
' GetSRQコマンド実行
Dim val As Byte
```

```
val = caoCon.Execute("GetSRQ")
```

3.3.1.2.6. SetSRQ コマンド

割り込みをリセットします.

項目	型説明	
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetSRQコマンド実行
Dim result As String
result = caoCon.Execute("SetSRQ")
```

3.3.1.2.7. GetALARM コマンド

アラーム状態を取得します.

項目	型説明		
戻り値	VT_EMPTY		発生しているアラームがありません
	または		
	VT_ARRAY | VT_I4		最大 16 個までのアラーム情報
	i	VT_I4	アラーム番号 **※添字 i： 発生しているアラーム数分(0 - 15)**

使用例

```
' GetALARMコマンド実行
Dim values As Variant
values = caoCon.Execute("GetALARM")

' 発生しているアラームの数だけ取得
If Not IsEmpty(values) Then
    Dim i As Long
    For i = 0 To UBound(values)
        Dim val As Long
        val = values(i)
    Next i
End If
```

3.3.1.2.8. GetMODE コマンド

運転モードを取得します.

項目	型説明	
戻り値	VT_BSTR	運転モード 以下のいずれかが取得されます. ・ OFF : パネル電源 OFF 状態 ・ STANDBY : 停止状態 ・ CONSTANT : 定値運転状態 ・ RUN : プログラム/リモート運転状態

使用例

```
' GetMODEコマンド実行
Dim val As String
val = caoCon.Execute("GetMODE")
```

3.3.1.2.9. SetMODE コマンド

運転モードを設定します.

項目	型説明	
引数	VT_BSTR	運転モードを以下のいずれかにて指定します. ・ OFF : パネル電源 OFF ・ STANDBY : 停止中 ・ CONSTANT : 定値運転(No1) ・ RUN : プログラム運転
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetMODEコマンド実行
Dim result As String
result = caoCon.Execute("SetMODE", "OFF")
```

3.3.1.2.10. GetMODEDETAIL コマンド

詳細な運転モードを取得します.

項目	型説明

項目	型説明	
戻り値	VT_BSTR	運転モード 以下のいずれかが取得されます. ・ OFF : パネル電源 OFF 状態 ・ STANDBY : 停止状態 ・ CONSTANT : 定値運転状態 ・ RUN : プログラム運転中 ・ RUN PAUSE : プログラム一時停止中 ・ RUN END HOLD : プログラム最終保持 ・ RMT RUN : リモート運転中 ・ RMT RUN PAUSE : リモート一時停止中 ・ RMT RUN END HOLD : リモート最終保持

使用例

```
' GetMODEDETAILコマンド実行
Dim val As String
val = caoCon.Execute("GetMODEDETAIL")
```

3.3.1.2.11. GetMONDETAIL コマンド

詳細な槽内の状態を取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_R8	測定温度
	1	VT_I4	測定湿度 ※温度のみタイプの装置の場合は，取得データが省略されているので VT_EMPTY となります
	2	VT_BSTR	運転モード GetMODEDETAIL コマンドの戻り値と同様ですので参照して下さい.
	3	VT_I4	発生中のアラーム数

使用例

```
' GetMONDETAILコマンド実行
Dim values As Variant
values = caoCon.Execute("GetMONDETAIL")

' 測定温度
```

```
Dim temp As Double
temp = values(0)

' 測定湿度
If Not IsEmpty(values(1)) Then
    Dim humi As Long
    humi = values(1)
End If

' 運転モード
Dim mode As String
mode = values(2)

' 発生中のアラーム数
Dim numAlarm As Long
numAlarm = values(3)
```

3.3.1.2.12. GetHEATER コマンド

ヒーターの状態を取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_R8	加熱ヒーター出力
	1	VT_R8	加湿ヒーター出力 ※温度のみタイプの装置の場合は，取得データが省略されているので VT_EMPTY となります

使用例

```
' GetHEATERコマンド実行
Dim values As Variant
values = caoCon.Execute("GetHEATER")

' 加熱ヒーター出力
Dim heating As Double
heating = values(0)

' 加湿ヒーター出力
If Not IsEmpty(values(1)) Then
    Dim humidification As Double
    humidification = values(1)
End If
```

3.3.1.3. プログラム関連コマンド

プログラムの状態及び設定値を取得/設定するコマンドです.

3.3.1.3.1. GetRUNPRGDATA コマンド

リモートプログラムデータを取得します.

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_R8	開始温度
	1	VT_R8	到達温度
	2	VT_I4	開始湿度 ※温度のみタイプの装置の場合は，取得データが省略されているので VT_EMPTY となります
	3	VT_I4	到達湿度 ※温度のみタイプの装置の場合は，取得データが省略されているので VT_EMPTY となります
	4	VT_BSTR	時間設定 "時間:分"(可変長)
	5	VT_BSTR	冷凍機の制御値 制御値に関しては，GetSET コマンドの戻り値と同様ですのでそちらを参照して下さい.
	6	VT_ARRAY | VT_VARIANT	タイムシグナル設定 ※ON されたタイムシグナルがない場合は，取得データが省略されているので VT_EMPTY となります
		6.0 VT_BSTR	タイムシグナル状態 以下のいずれかが取得されます. ・ ON ： ON されたタイムシグナル ・ OFF ： OFF されたタイムシグナル
		6.i VT_I4	タイムシグナル番号 **※添字 i： タイムシグナル数分(1 - 11)**

使用例

```
' GetRUNPRGDATAコマンド実行
Dim values As Variant
values = caoCon.Execute("GetRUNPRGDATA")

' 開始温度
Dim temp As Double
temp = values(0)
```

```
    ' 到達温度
    Dim goTemp As Double
    goTemp = values(1)

    ' 開始湿度
    If Not IsEmpty(values(2)) Then
        Dim humi As Long
        humi = values(2)
    End If

    ' 到達湿度
    If Not IsEmpty(values(3)) Then
        Dim goHumi As Long
        goHumi = values(3)
    End If

    ' 時間設定
    Dim time As String
    time = values(4)

    ' 冷凍機の制御値
    Dim ref As String
    ref = values(5)

    ' タイムシグナル設定
    Dim relays As Variant
    relays= values(6)

    If Not IsEmpty(relays) Then
        ' タイムシグナル状態
        Dim relayStatus As String
        relayStatus = relays(0)

        ' タイムシグナル番号

        Dim i As Long
        For i = 1 To UBound(relays)
            Dim relayNo As Long
            relayNo = relays(i)
        Next i
    End If
```

3.3.1.3.2. SetRUNPRGDATA コマンド

リモートプログラムデータを設定します.

項目	型説明

項目	型説明			
引数	VT_ARRAY | VT_VARIANT			
	0	VT_R8		開始温度を指定します.
	1	VT_R8		到達温度を指定します. ※VT_EMPTY にすると未設定が可能です.
	2	VT_I4		開始湿度を指定します. ※VT_EMPTY にすると未設定が可能です.
	3	VT_I4		到達湿度を指定します. ※VT_EMPTY にすると未設定が可能です.
	4	VT_BSTR		時間設定を以下のように指定します. "時間:分"(可変長)
	5	VT_BSTR		冷凍機の制御値を指定します. 指定する制御値に関しては, SetSET コマンドの引数と同様ですのでそちらを参照して下さい. ※VT_EMPTY にすると未設定が可能です.
	6	VT_ARRAY | VT_VARIANT		タイムシグナル設定を指定します. ※VT_EMPTY にすると未設定が可能です.
		6.0	VT_BSTR	タイムシグナル状態を以下のいずれかにて指定します. ・ON : ON させるタイムシグナル ・OFF : OFF させるタイムシグナル
		6.i	VT_I4	タイムシグナル番号を指定します. **※添字 i : タイムシグナル数分(1 - 11)**
戻り値	VT_BSTR			コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' SetRUNPRGDATAコマンド実行
Dim result As String

result = caoCon.Execute("SetRUNPRGDATA", Array(10.0, 23.0, 85, 100, "1:00", _
                                                Empty, Empty))
```

3.3.2. CaoExtension クラスの拡張コマンド

CaoExtension クラスの拡張コマンドは，シリアル通信で接続されている場合のみ有効です．使用例は各コマンドの詳細で記述しています．

表 3-7 CaoExtension クラスの拡張コマンド一覧

コマンド	説明	参照
機器の設定値関連コマンド		
GetROM	ROM バージョンを取得します.	P.43
GetMASK	割り込みマスクを取得します.	P.43
SetMASK	割り込みマスクを設定します.	P.43
GetKEYLOCK	キーロックの状態を取得します.	P.43
SetKEYLOCK	キーロックの状態を設定します.	P.44
GetTYPE	チャンバータイプを取得します.	P.44
GetTEMP	温度パラメーターを取得します.	P.44
SetTEMP	温度パラメーターを設定します.	P.44
SetSTEMP	温度パラメーターの設定値を設定します.	P.44
SetHTEMP	温度パラメーターの上限値を設定します.	P.44
SetLTEMP	温度パラメーターの下限値を設定します.	P.45
GetHUMI	湿度パラメーターを取得します.	P.45
SetHUMI	湿度パラメーターを設定します.	P.45
SetSHUMI	湿度パラメーターの設定値を設定します.	P.45
SetHHUMI	湿度パラメーターの上限値を設定します.	P.45
SetLHUMI	湿度パラメーターの下限値を設定します.	P.45
GetSET	冷凍機の制御値を取得します.	P.45
SetSET	冷凍機の制御値を設定します.	P.46
機器の状態関連コマンド		
GetSRQ	割り込みを取得します.	P.46
SetSRQ	割り込みをリセットします.	P.46
GetALARM	アラーム状態を取得します.	P.46
GetMODE	運転モードを取得します.	P.46
SetMODE	運転モードを設定します.	P.47
GetMON	槽内の状態を取得します.	P.47
GetHEATER	ヒーターの状態を取得します.	P.48

コマンド		説明	参照
	SetPOWER	電源の状態を設定します.	P.49
プログラム関連コマンド			
	GetRUNPRGDATA	リモートプログラムデータを取得します.	P.49
	SetRUNPRGDATA	リモートプログラムデータを設定します.	P.49

3.3.2.1. 機器の設定値関連コマンド

機器に設定されている設定値を取得/設定するコマンドです.

3.3.2.1.1. GetROM コマンド

ROM バージョンを取得します. 詳細は, CaoController クラスの「GetROM コマンド」を参照して下さい.

3.3.2.1.2. GetMASK コマンド

割り込みマスクを取得します. 詳細は, CaoController クラスの「GetMASK コマンド」を参照して下さい.

3.3.2.1.3. SetMASK コマンド

割り込みマスクを設定します. 詳細は, CaoController クラスの「SetMASK コマンド」を参照して下さい.

3.3.2.1.4. GetKEYLOCK コマンド

キーロックの状態を取得します.

項目	型説明	
戻り値	VT_BSTR	キーロック状態 以下のいずれかが取得されます. ・ ON : キーロック ON 状態 ・ OFF : キーロック OFF 状態

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' GetKEYLOCKコマンド実行
Dim val As String
val = caoExt.Execute("GetKEYLOCK")
```

3.3.2.1.5. SetKEYLOCK コマンド

キーロックの状態を設定します.

項目	型説明	
引数	VT_BSTR	キーロックの状態を以下のいずれかにて指定します. ・ ON ： キーロック ON ・ OFF ： キーロック OFF
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' SetKEYLOCKコマンド実行
Dim result As String
result = caoExt.Execute("SetKEYLOCK", "OFF")
```

3.3.2.1.6. GetTYPE コマンド

チャンバータイプを取得します. 詳細は, CaoController クラスの「GetTYPE コマンド」を参照して下さい.

3.3.2.1.7. GetTEMP コマンド

温度パラメーターを取得します. 詳細は, CaoController クラスの「GetTEMP コマンド」を参照して下さい.

3.3.2.1.8. SetTEMP コマンド

温度パラメーターを設定します. 詳細は, CaoController クラスの「SetTEMP コマンド」を参照して下さい.

3.3.2.1.9. SetSTEMP コマンド

温度パラメーターの設定値を設定します. 詳細は, CaoController クラスの「SetSTEMP コマンド」を参照して下さい.

3.3.2.1.10. SetHTEMP コマンド

温度パラメーターの上限値を設定します. 詳細は, CaoController クラスの「SetHTEMP コマンド」を参照して下さい.

3.3.2.1.11. SetLTEMP コマンド

温度パラメーターの下限値を設定します. 詳細は, CaoController クラスの「SetLTEMP コマンド」を参照して下さい.

3.3.2.1.12. GetHUMI コマンド

湿度パラメーターを取得します. 詳細は, CaoController クラスの「GetHUMI コマンド」を参照して下さい.

3.3.2.1.13. SetHUMI コマンド

湿度パラメーターを設定します. 詳細は, CaoController クラスの「SetHUMI コマンド」を参照して下さい.

3.3.2.1.14. SetSHUMI コマンド

湿度パラメーターの設定値を設定します. 詳細は, CaoController クラスの「SetSHUMI コマンド」を参照して下さい.

3.3.2.1.15. SetHHUMI コマンド

湿度パラメーターの上限値を設定します. 詳細は, CaoController クラスの「SetHHUMI コマンド」を参照して下さい.

3.3.2.1.16. SetLHUMI コマンド

湿度パラメーターの下限値を設定します. 詳細は, CaoController クラスの「SetLHUMI コマンド」を参照して下さい.

3.3.2.1.17. GetSET コマンド

冷凍機の制御値を取得します.

項目	型説明	
戻り値	VT_BSTR	制御値 以下のいずれかが取得されます. ・ REF0 : 手動設定(停止) ・ REF1 : 手動設定(運転) ・ REF9 : 自動設定

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
```

```
' GetSETコマンド実行
Dim val As String
val = caoExt.Execute("GetSET")
```

3.3.2.1.18. SetSET コマンド

冷凍機の制御値を設定します.

項目	型説明	
引数	VT_BSTR	制御値を以下のいずれかにて指定します. ・ REF0 : 手動(停止要求) ・ REF1 : 手動(運転要求) ・ REF9 : 自動
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' SetSETコマンド実行
Dim result As String
result = caoExt.Execute("SetSET", "REF9")
```

3.3.2.2. 機器の状態関連コマンド

機器の状態を取得/設定するコマンドです.

3.3.2.2.1. GetSRQ コマンド

割り込みを取得します. 詳細は, CaoController クラスの「GetSRQ コマンド」を参照して下さい.

3.3.2.2.2. SetSRQ コマンド

割り込みをリセットします. 詳細は, CaoController クラスの「SetSRQ コマンド」を参照して下さい.

3.3.2.2.3. GetALARM コマンド

アラーム状態を取得します. 詳細は, CaoController クラスの「GetALARM コマンド」を参照して下さい.

3.3.2.2.4. GetMODE コマンド

運転モードを取得します.

項目	型説明

項目	型説明	
戻り値	VT_BSTR	運転モード 以下のいずれかが取得されます. ・ OFF : パネル電源 OFF 状態 ・ STANDBY : スタンバイ状態 ・ CONSTANT : 定値運転状態 ・ RUN : プログラム/リモート運転状態

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' GetMODEコマンド実行
Dim val As String
val = caoExt.Execute("GetMODE")
```

3.3.2.2.5. SetMODE コマンド

運転モードを設定します.

項目	型説明	
引数	VT_BSTR	運転モードを以下のいずれかにて指定します. ・ OFF : パネル電源 OFF にする ・ STANDBY : スタンバイ状態にする ・ CONSTANT : 定値運転にする ・ RUN プログラム番号 : プログラム運転を行う
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' SetMODEコマンド実行
Dim result As String
result = caoExt.Execute("SetMODE", "STANDBY")
```

3.3.2.2.6. GetMON コマンド

槽内の状態を取得します.

項目	型説明

項目	型説明		
戻り値	VT_ARRAY | VT_VARIANT		
	0	VT_R8	測定温度
	1	VT_I4	測定湿度 ※温度のみタイプの装置の場合は，取得データが省略されているので VT_EMPTY となります
	2	VT_BSTR	運転モード GetMODE コマンドの戻り値と同様ですので参照して下さい.
	3	VT_I4	発生中のアラーム数

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' GetMONコマンド実行
Dim values As Variant
values = caoExt.Execute("GetMON")

' 測定温度
Dim temp As Double
temp = values(0)

' 測定湿度
If Not IsEmpty(values(1)) Then
    Dim humi As Long
    humi = values(1)
End If

' 運転モード
Dim mode As String
mode = values(2)

' 発生中のアラーム数
Dim numAlarm As Long
numAlarm = values(3)
```

3.3.2.2.7. GetHEATER コマンド

ヒーターの状態を取得します. 詳細は，CaoController クラスの「GetHEATER コマンド」を参照して下さい.

3.3.2.2.8. SetPOWER コマンド

電源の状態を設定します.

項目	型説明	
引数	VT_BSTR	電源の状態を以下のいずれかにて指定します. ・ ON ： パネル電源を ON にし, 定値運転にします ・ OFF ： パネル電源を OFF にします
戻り値	VT_BSTR	コマンドが正常に実行された場合,機器からの応答文字列がそのまま格納されます.

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' SetPOWERコマンド実行
Dim result As String
result = caoExt.Execute("SetPOWER", "OFF")
```

3.3.2.3. プログラム関連コマンド

プログラムの状態及び設定値を取得/設定するコマンドです.

3.3.2.3.1. GetRUNPRGDATA コマンド

リモートプログラムデータを取得します. 詳細は, CaoController クラスの「GetRUNPRGDATA コマンド」を参照して下さい.

3.3.2.3.2. SetRUNPRGDATA コマンド

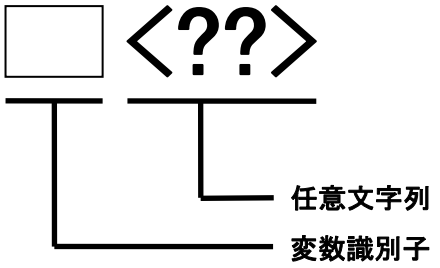
リモートプログラムデータを設定します. 詳細は, CaoController クラスの「SetRUNPRGDATA コマンド」を参照して下さい.

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. 複数変数を登録（オプションのみ変更したい場合等に有用）するために任意の文字列を付与することが可能です.

変数名に任意文字列を付与するための書式を以下に示します.

複数変数共通指定書式



3.4.1. CaoController クラスの変数

CaoController クラスの「@MAKER_NAME」,「@VERSION」を除いた変数は，イーサネット接続した場合のみ有効な変数です.

表 3-8 CaoController クラスの変数一覧

変数名	説明	Value		参照
		get	put	
プロバイダ情報関連変数				
@MAKER_NAME	メーカー名を取得します.	○	―	P.52
@VERSION	プロバイダバージョンを取得します.	○	―	P.52
@LASTDEVERR	最後のデバイスエラーを取得します.	○	―	P.52
機器の設定値関連変数				
@ROM	温調器の ROM バージョンを取得します.	○	―	P.53
@ROMDISP	表示器の ROM バージョンを取得します.	○	―	P.53
@MASK	割り込みマスクを取得/設定します.	○	○	P.54
@TIMERMON	起動されているタイマーを取得します.	○	―	P.54
@TIMERSET	設定されているタイマーを取得します.	○	―	P.55
@TIMER0	クイックタイマーの設定を取得/設定します.	○	○	P.55
@TIMER1	運転開始タイマーの設定を取得/設定します.	○	○	P.56
@TIMER2	運転停止タイマーの設定を取得/設定します.	○	○	P.56
@TIMERERASE	指定されたタイマーの設定を削除します.	―	○	P.57
@TIMERON	指定されたタイマーを起動します.	―	○	P.57
@TIMEROFF	指定されたタイマーを停止します.	―	○	P.57
@KEYPROTECT	プロテクト設定を取得/設定します.	○	○	P.58
@TYPE	チャンバータイプを取得します.	○	―	P.58

変数名	説明	Value		参照
		get	put	
@TEMPMON	温度パラメーターを取得します.	○	－	P.59
@TEMP	温度パラメーターを設定します.	－	○	P.59
@STEMP	温度設定値を設定します.	－	○	P.59
@HTEMP	温度上限値を設定します.	－	○	P.60
@LTEMP	温度下限値を設定します.	－	○	P.60
@HUMIMON	湿度パラメーターを取得します.	○	－	P.60
@HUMI	湿度パラメーターを設定します.	－	○	P.61
@SHUMI	湿度設定値を設定します.	－	○	P.61
@HHUMI	湿度上限値を設定します.	－	○	P.61
@LHUMI	湿度下限値を設定します.	－	○	P.62
@SET	冷凍機の制御値を取得/設定します.	○	○	P.62
@CONSTANTSETTEMP	定値設定の温度設定値を取得します.	○	－	P.62
@CONSTANTSETHUMI	定値設定の湿度設定値を取得します.	○	－	P.63
@CONSTANTSETREF	定値設定の冷凍機設定値を取得します.	○	－	P.63
@SYSTEMSETPTS	試料温度モニター機能を取得します.	○	－	P.63
@SYSTEMSETPTC	試料温度制御機能を取得します.	○	－	P.64
@SYSTEMSETPTCOPT	試料温度制御機能オプションを取得します.	○	－	P.64
機器の状態関連変数				
@DATE	内部カレンダーの日付を取得/設定します.	○	○	P.64
@TIME	内部カレンダーの現在時刻を取得/設定します.	○	○	P.65
@SRQ	割り込みを取得します.	○	－	P.65
@ALARM	アラーム状態を取得します.	○	－	P.66
@MODE	運転モードを取得/設定します.	○	○	P.66
@MODEDETAIL	詳細な運転モードを取得します.	○	－	P.67
@MONDETAIL	詳細な槽内の状態を取得します.	○	－	P.67
@HEATER	ヒーターの状態を取得します.	○	－	P.67
プログラム関連変数				
@RUNPRGDATA	リモートプログラムデータを取得/設定しま	○	○	P.68

変数名		説明	Value		参照
			get	put	
		す.			

3.4.1.1. プロバイダ情報関連変数

本プロバイダが保持している情報を取得する変数です.

3.4.1.1.1. @MAKER_NAME

メーカー名を取得します.

データ型

項目	型説明	
取得	VT_BSTR	メーカー名

使用例

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

3.4.1.1.2. @VERSION

プロバイダバージョンを取得します.

データ型

項目	型説明	
取得	VT_BSTR	プロバイダバージョン *.*.*

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@VERSION")
' 値取得
Dim val As String
val = var.value
```

3.4.1.1.3. @LASTDEVERR

最後に発生したデバイスエラーを取得します.

データ型

項目	型説明		
取得	VT_ARRAY | VT_VARIANT		
	0	VT_BSTR	エラーメッセージ
	1	VT_I4	エラーコード

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@LASTDEVERR")
' 値取得
Dim values As Variant
values = var.value

' エラーメッセージ
Dim errMsg As String
errMsg = values(0)

' エラーコード
Dim errCode As Long
errCode = values(1)
```

3.4.1.2. 機器の設定値関連変数

機器に設定されている設定値を取得/設定する変数です.

3.4.1.2.1. @ROM

温調器の ROM バージョンを取得します.

データ型

取得するデータについては, CaoController クラスの「GetROM コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@ROM")
' 値取得
Dim val As String
val = var.value
```

3.4.1.2.2. @ROMDISP

表示器の ROM バージョンを取得します.

データ型

取得するデータについては, CaoController クラスの「GetROMDISP コマンド」の戻り値を参照して

ください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@ROMDISP")
' 値取得
Dim val As String
val = var.value
```

3.4.1.2.3. @MASK

割り込みマスクを取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetMASK コマンド」の戻り値と「SetMASK コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_UI1	割り込みマスク

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@MASK")
' 値取得
Dim val As Byte
val = var.value
' 値設定
var.value = 64
```

3.4.1.2.4. @TIMERMON

起動されているタイマーを取得します.

データ型

取得するデータについては, CaoController クラスの「GetTIMERMON コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@TIMERMON")
' 値取得
Dim values As Variant
values = var.value
```

```
    ' 起動されているタイマー数分取得
    If Not IsEmpty(values) Then
        Dim i As Long
        For i = 0 To UBound(values)
                Dim timerNo As Long
                timerNo = values(i)
        Next i
    End If
```

3.4.1.2.5. @TIMERSET

設定されているタイマーを取得します.

データ型

取得するデータについては, CaoController クラスの「GetTIMERSET コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@TIMERSET")
    ' 値取得
    Dim values As Variant
    values = var.value

    ' 設定されているタイマー数分取得
    If Not IsEmpty(values) Then
        Dim i As Long
        For i = 0 To UBound(values)
                Dim timerNo As Long
                timerNo = values(i)
        Next i
    End If
```

3.4.1.2.6. @TIMER0

クイックタイマーの設定を取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetTIMER0 コマンド」の戻り値と「SetTIMER0 コマンド」の引数を参照してください.

データ型

項目	型説明	
取得	VT_BSTR	運転モード,設定時間　または　停止モード,設定時間
設定	VT_BSTR	起動時間,運転モード　または　起動時間,停止モード

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@TIMER0")
' 値取得
Dim val As String
val = var.value
' 値設定
var.value = "10:00,CONSTANT"
```

3.4.1.2.7. @TIMER1

運転開始タイマーの設定を取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetTIMER1 コマンド」の戻り値と「SetTIMER1 コマンド」の引数を参照してください.

データ型

項目	型説明	
取得	VT_BSTR	タイマー番号,起動モード,運転モード
設定	VT_BSTR	起動モード,運転モード

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@TIMER1")
' 値取得
Dim val As String
val = var.value
' 値設定
var.value = " MODE1,12.03/04,10:00,CONSTANT"
```

3.4.1.2.8. @TIMER2

運転停止タイマーの設定を取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetTIMER2 コマンド」の戻り値と「SetTIMER2 コマンド」の引数を参照してください.

データ型

項目	型説明	
取得	VT_BSTR	タイマー番号,起動モード,停止モード
設定	VT_BSTR	起動モード,停止モード

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@TIMER2")
    ' 値取得
    Dim val As String
    val = var.value
    ' 値設定
    var.value = "MODE2,SAT,10:00,OFF"
```

3.4.1.2.9. @TIMERERASE

指定されたタイマーの設定を削除します.

データ型

設定するデータについては, CaoController クラスの「SetTIMERMERASE コマンド」の引数を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@TIMERERASE")
    ' 値設定
    var.value = 1
```

3.4.1.2.10. @TIMERON

指定されたタイマーを起動します.

データ型

設定するデータについては, CaoController クラスの「SetTIMERON コマンド」の引数を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@TIMERON")
    ' 値設定
    var.value = 1
```

3.4.1.2.11. @TIMEROFF

指定されたタイマーを停止します.

データ型

設定するデータについては, CaoController クラスの「SetTIMEROFF コマンド」の引数を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@TIMEROFF")
    ' 値設定
    var.value = 2
```

3.4.1.2.12. @KEYPROTECT

キープロテクトの状態を取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetKEYPROTECT コマンド」の戻り値と「SetKEYPROTECT コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_BSTR	キープロテクト状態

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@KEYPROTECT")
    ' 値取得
    Dim val As String
    val = var.value
    ' 値設定
    var.value = "OFF"
```

3.4.1.2.13. @TYPE

チャンバータイプを取得します.

データ型

取得するデータについては, CaoController クラスの「GetTYPE コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@TYPE")
    ' 値取得
    Dim values As Variant
    values = var.value
```

3.4.1.2.14. @TEMPMON

温度パラメーターを取得します.

データ型

取得するデータについては, CaoController クラスの「GetTEMP コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@TEMPMON")
' 値取得
Dim values() As Double
values = var.value
```

3.4.1.2.15. @TEMP

温度パラメーターを設定します.

データ型

設定するデータについては, CaoController クラスの「SetTEMP コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@TEMP")
' 値設定
var.value = Array(23.0, 100.0, -40.0)
```

3.4.1.2.16. @STEMP

温度設定値を設定します.

データ型

設定するデータについては, CaoController クラスの「SetSTEMP コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@STEMP")
' 値設定
var.value = 23.0
```

3.4.1.2.17. @HTEMP

温度上限値を設定します.

データ型

設定するデータについては, CaoController クラスの「SetHTEMP コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@HTEMP")
' 値設定
var.value = 100.0
```

3.4.1.2.18. @LTEMP

温度下限値を設定します.

データ型

設定するデータについては, CaoController クラスの「SetLTEMP コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@LTEMP")
' 値設定
var.value = -40.0
```

3.4.1.2.19. @HUMIMON

湿度パラメーターを取得します.

データ型

取得するデータについては, CaoController クラスの「GetHUMI コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@HUMIMON")
' 値取得
Dim values() As Long
values = var.value
```

3.4.1.2.20. @HUMI

湿度パラメーターを設定します.

データ型

設定するデータについては, CaoController クラスの「SetHUMI コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@HUMI")
' 値設定
var.value = Array(23, 100, 0)
```

3.4.1.2.21. @SHUMI

湿度設定値を設定します.

データ型

設定するデータについては, CaoController クラスの「SetSHUMI コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@SHUMI")
' 値設定
var.value = 85
```

3.4.1.2.22. @HHUMI

湿度上限値を設定します.

データ型

設定するデータについては, CaoController クラスの「SetHHUMI コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@HHUMI")
' 値設定
var.value = 100
```

3.4.1.2.23. @LHUMI

湿度下限値を設定します.

データ型

設定するデータについては, CaoController クラスの「SetLHUMI コマンド」の引数を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@LHUMI")
' 値設定
var.value = 0
```

3.4.1.2.24. @SET

冷凍機の制御値を取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetSET コマンド」の戻り値と「SetSET コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_BSTR	制御値

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@SET")
' 値取得
Dim val As String
val = var.value
' 値設定
var.value = "REF9"
```

3.4.1.2.25. @CONSTANTSETTEMP

定値設定の温度設定値を取得します.

データ型

取得するデータについては, CaoController クラスの「GetCONSTANTSETTEMP コマンド」の戻り値を参照してください.

使用例

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

```
    ' 値取得
    Dim values As Variant
    values = var.value
```

3.4.1.2.26. @CONSTANTSETHUMI

定値設定の湿度設定値を取得します.

データ型

取得するデータについては, CaoController クラスの「GetCONSTANTSETHUMI コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@CONSTANTSETHUMI")
    ' 値取得
    Dim values As Variant
    values = var.value
```

3.4.1.2.27. @CONSTANTSETREF

定値設定の冷凍機設定値を取得します.

データ型

取得するデータについては, CaoController クラスの「GetCONSTANTSETREF コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@CONSTANTSETREF")
    ' 値取得
    Dim val As String
    val = var.value
```

3.4.1.2.28. @SYSTEMSETPTS

試料温度モニター機能を取得します.

データ型

取得するデータについては, CaoController クラスの「GetSYSTEMSETPTS コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
```

```
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@SYSTEMSETPTS")
    ' 値取得
    Dim val As String
    val = var.value
```

3.4.1.2.29. @SYSTEMSETPTC

試料温度制御機能を取得します.

データ型

取得するデータについては, CaoController クラスの「GetSYSTEMSETPTC コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@SYSTEMSETPTC")
    ' 値取得
    Dim val As String
    val = var.value
```

3.4.1.2.30. @SYSTEMSETPTCOPT

試料温度制御機能オプションを取得します.

データ型

取得するデータについては, CaoController クラスの「GetSYSTEMSETPTCOPT コマンド」の戻り値を参照してください.

使用例

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoCon.AddVariable("@SYSTEMSETPTCOPT")
    ' 値取得
    Dim val As String
    val = var.value
```

3.4.1.3. 機器の状態関連変数

機器の状態を取得/設定する変数です.

3.4.1.3.1. @DATE

内部カレンダーの日付を取得/設定します.

データ型

項目	型説明	
取得/設定	VT_BSTR	日付 YY.MM/DD

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@DATE")
' 値取得
Dim val As String
val = var.value
' 値設定
var.value = "19.10/03"
```

3.4.1.3.2. @TIME

内部カレンダーの現在時刻を取得/設定します.

データ型

項目	型説明	
取得/設定	VT_BSTR	現在時刻 hh:mm:ss

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@TIME")
' 値取得
Dim val As String
val = var.value
' 値設定
var.value = "18:00:00
```

3.4.1.3.3. @SRQ

割り込みを取得します.

データ型

取得するデータについては, CaoController クラスの「GetSRQ コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
```

```
Set var = caoCon.AddVariable("@SRQ")
' 値取得
Dim val As Byte
val = var.value
```

3.4.1.3.4. @ALARM

アラーム状態を取得します.

データ型

取得するデータについては, CaoController クラスの「GetALARM コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@ALARM")
' 値取得
Dim values As Variant
values = var.value

If Not IsEmpty(values) Then
    Dim i As Long
    For i = 0 To UBound(values)
        Dim val As Long
        val = values(i)
    Next i
End If
```

3.4.1.3.5. @MODE

運転モードを取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetMODE コマンド」の戻り値と「SetMODE コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_BSTR	運転モード

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@MODE")
' 値取得
Dim val As String
val = var.value
' 値設定
```

```
var.value = "OFF"
```

3.4.1.3.6. @MODEDETAIL

詳細な運転モードを取得します.

データ型

取得するデータについては, CaoController クラスの「GetMODEDETAIL コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@MODEDETAIL")
' 値取得
Dim val As String
val = var.value
```

3.4.1.3.7. @MONDETAIL

詳細な槽内の状態を取得します.

データ型

取得するデータについては, CaoController クラスの「GetMONDETAIL コマンド」の戻り値を参照してください.

使用例

```
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@MONDETAIL")
' 値取得
Dim values As Variant
values = var.value
```

3.4.1.3.8. @HEATER

ヒーターの状態を取得します.

データ型

取得するデータについては, CaoController クラスの「GetHEATER コマンド」の戻り値を参照してください.

使用例

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

```
' 値取得
Dim values As Variant
values = var.value
```

3.4.1.4. プログラム関連変数

プログラムの状態及び設定値を取得/設定する変数です.

3.4.1.4.1. @RUNPRGDATA

リモートプログラムデータを取得/設定します. 取得/設定するデータの詳細については, CaoController クラスの「GetRUNPRGDATA コマンド」の戻り値と「SetRUNPRGDATA コマンド」の引数を参照してください.

データ型

項目	型説明		
取得/設定	VT_ARRAY | VT_VARIANT		
	0	VT_R8	開始温度
	1	VT_R8	到達温度
	2	VT_I4	開始湿度
	3	VT_I4	到達湿度
	4	VT_BSTR	時間設定
	5	VT_BSTR	冷凍機の制御値
	6	VT_ARRAY | VT_VARIANT	タイムシグナル設定
		6.0 VT_BSTR	タイムシグナル状態
		6.i VT_I4	タイムシグナル番号

使用例

```
' コントローラの追加
Dim caoCon As CaoController
Set caoCon = caoWor.AddController("Controller", "CaoProv.espec.TempHumi", "", _
                                  conn=eth:192.168.0.2")
' 変数追加
Dim var As CaoVariable
Set var = caoCon.AddVariable("@RUNPRGDATA")
' 値取得
Dim values As Variant
values = var.value
' 値設定
var.value = Array(10.0, 23.0, 85, 100, "1:00")
```

3.4.2. CaoExtension クラスの変数

CaoExtension クラスの変数は, シリアル通信で接続されている場合のみ有効です.

表 3-9 CaoExtension クラスの変数一覧

変数名	説明	Value		参照
		get	put	
プロバイダ情報関連変数				
@LASTDEVERR	最後のデバイスエラーを取得します.	○	－	P.70
機器の設定値関連変数				
@ROM	ROM バージョンを取得します.	○	－	P.70
@MASK	割り込みマスクを取得/設定します.	○	○	P.70
@KEYLOCK	キーロックの状態を取得/設定します.	○	○	P.70
@TYPE	チャンバータイプを取得します.	○	－	P.70
@TEMPMON	温度パラメーターを取得します.	○	－	P.71
@TEMP	温度パラメーターを設定します.	－	○	P.71
@STEMP	温度設定値を設定します.	－	○	P.71
@HTEMP	温度上限値を設定します.	－	○	P.71
@LTEMP	温度下限値を設定します.	－	○	P.71
@HUMIMON	湿度パラメーターを取得します.	○	－	P.71
@HUMI	湿度パラメーターを設定します.	－	○	P.71
@SHUMI	湿度設定値を設定します.	－	○	P.71
@HHUMI	湿度上限値を設定します.	－	○	P.71
@LHUMI	湿度下限値を設定します.	－	○	P.71
@SET	冷凍機の制御値を取得/設定します.	○	○	P.72
機器の状態関連変数				
@SRQ	割り込みを取得します.	○	－	P.72
@ALARM	アラーム状態を取得します.	○	－	P.72
@MODE	運転モードを取得/設定します.	○	○	P.72
@MON	槽内の状態を取得します.	○	－	P.73
@HEATER	ヒーターの状態を取得します.	○	－	P.73
@POWER	電源の状態を設定します.	－	○	P.73
プログラム関連変数				
@RUNPRGDATA	リモートプログラムデータを取得/設定します.	○	○	P.74

3.4.2.1. プロバイダ情報関連変数

本プロバイダが保持している情報を取得する変数です.

3.4.2.1.1. @LASTDEVERR

最後に発生したデバイスエラーを取得します. 詳細は, CaoController クラスの変数「@LASTDEVERR」を参照してください.

3.4.2.2. 機器の設定値関連変数

機器に設定されている設定値を取得/設定する変数です.

3.4.2.2.1. @ROM

ROM バージョンを取得します. 詳細は, CaoController クラスの変数「@ROM」を参照してください.

3.4.2.2.2. @MASK

割り込みマスクを取得/設定します. 割り込みマスクを取得/設定します. 詳細は, CaoController クラスの変数「@MASK」を参照してください.

3.4.2.2.3. @KEYLOCK

キーロックの状態を取得/設定します. 取得/設定するデータの詳細については, CaoExtension クラスの「GetKEYLOCK コマンド」の戻り値と「SetKEYLOCK コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_BSTR	キーロック状態

使用例

```
    ' 拡張ボードの追加
    Dim caoExt As CaoExtension
    Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoExt.AddVariable("@KEYLOCK")
    ' 値取得
    Dim val As String
    val = var.value
    ' 値設定
    var.value = "ON"
```

3.4.2.2.4. @TYPE

チャンバータイプを取得します. 詳細は, CaoController クラスの変数「@TYPE」を参照してください.

3.4.2.2.5. @TEMPMON

温度パラメーターを取得します. 詳細は, CaoController クラスの変数「@TEMPMON」を参照してください.

3.4.2.2.6. @TEMP

温度パラメーターを設定します. 詳細は, CaoController クラスの変数「@TEMP」を参照してください.

3.4.2.2.7. @STEMP

温度設定値を設定します. 詳細は, CaoController クラスの変数「@STEMP」を参照してください.

3.4.2.2.8. @HTEMP

温度上限値を設定します. 詳細は, CaoController クラスの変数「@HTEMP」を参照してください.

3.4.2.2.9. @LTEMP

温度下限値を設定します. 詳細は, CaoController クラスの変数「@LTEMP」を参照してください.

3.4.2.2.10. @HUMIMON

湿度パラメーターを取得します. 詳細は, CaoController クラスの変数「@HUMIMON」を参照してください.

3.4.2.2.11. @HUMI

湿度パラメーターを設定します. 詳細は, CaoController クラスの変数「@HUMI」を参照してください.

3.4.2.2.12. @SHUMI

湿度設定値を設定します. 詳細は, CaoController クラスの変数「@SHUMI」を参照してください.

3.4.2.2.13. @HHUMI

湿度上限値を設定します. 詳細は, CaoController クラスの変数「@HHUMI」を参照してください.

3.4.2.2.14. @LHUMI

湿度下限値を設定します. 詳細は, CaoController クラスの変数「@LHUMI」を参照してください.

3.4.2.2.15. @SET

冷凍機の制御値を取得/設定します. 取得/設定するデータの詳細については, CaoExtension クラスの「GetSET コマンド」の戻り値と「SetSET コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_BSTR	制御値

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
' 変数追加
Dim var As CaoVariable
Set var = caoExt.AddVariable("@SET")
' 値取得
Dim val As String
val = var.value
' 値設定
var.value = "REF9"
```

3.4.2.3. 機器の状態関連変数

機器の状態を取得/設定する変数です.

3.4.2.3.1. @SRQ

割り込みを取得します. 詳細は, CaoController クラスの変数「@SRQ」を参照してください.

3.4.2.3.2. @ALARM

アラーム状態を取得します. 詳細は, CaoController クラスの変数「@ALARM」を参照してください.

3.4.2.3.3. @MODE

運転モードを取得/設定します. 取得/設定するデータの詳細については, CaoExtension クラスの「GetMODE コマンド」の戻り値と「SetMODE コマンド」の引数を参照してください.

データ型

項目	型説明	
取得/設定	VT_BSTR	運転モード

使用例

```
' 拡張ボードの追加
Dim caoExt As CaoExtension
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
```

```
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoExt.AddVariable("@MODE")
    ' 値取得
    Dim val As String
    val = var.value
    ' 値設定
    var.value = "STANDBY"
```

3.4.2.3.4. @MON

槽内の状態を取得します.

データ型

取得するデータについては, CaoExtension クラスの「GetMON コマンド」の戻り値を参照してください.

使用例

```
    ' 拡張ボードの追加
    Dim caoExt As CaoExtension
    Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
    ' 変数追加
    Dim var As CaoVariable
    Set var = caoExt.AddVariable("@MON")
    ' 値取得
    Dim values As Variant
    values = var.value
```

3.4.2.3.5. @HEATER

ヒーターの状態を取得します. 詳細は, CaoController クラスの変数「@HEATER」を参照してください.

3.4.2.3.6. @POWER

電源の状態を設定します.

データ型

設定するデータについては, CaoExtension クラスの「SetPOWER コマンド」の引数を参照してください.

使用例

```
    ' 拡張ボードの追加
    Dim caoExt As CaoExtension
    Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
    ' 変数追加
```

```
Dim var As CaoVariable
Set var = caoExt.AddVariable("@POWER")
' 値設定
var.value = "OFF"
```

3.4.2.4. プログラム関連変数

プログラムの状態及び設定値を取得/設定する変数です.

3.4.2.4.1. @RUNPRGDATA

リモートプログラムデータを取得/設定します. 詳細は, CaoController クラスの変数「@RUNPRGDATA」を参照してください.

4. espec TempHumi プロバイダによるプログラミング

espec TempHumi プロバイダでは,以下の手順でクライアント PC と TempHumi RS-485 デバイスまたは, TempHumi Ethernet デバイスを接続することができます.

初めに, 以下の手順でクライアント PC と TempHumi RS-485 デバイスをシリアル通信で接続することができます.

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

TempHumi RS-485 デバイスに接続した後は,CaoExtension の Execute メソッドを使用する,もしくは, CaoVariable オブジェクトを生成することで, TempHumi RS-485 デバイスの情報にアクセスすることができます.

次に, 以下の手順でクライアント PC と TempHumi Ethernet デバイスをイーサネット通信で接続することができます.

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

TempHumi Ethernet デバイスに接続した後は, CaoController の Execute メソッドを使用する, もしくは, CaoVariable オブジェクトを生成することで, TempHumi Ethernet デバイスの情報にアクセスすることができます.

4.1. TempHumi RS-485 デバイスの運転モードを変更するサンプルプログラミング

ここでは例として運転モードを変更する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 サンプルプログラムの要件

要件	説明
接続先	シリアルで接続する
	接続先 COM ポートは COM1
処理内容	現在の運転モードを取得する
	運転モードをスタンバイ状態に変更する

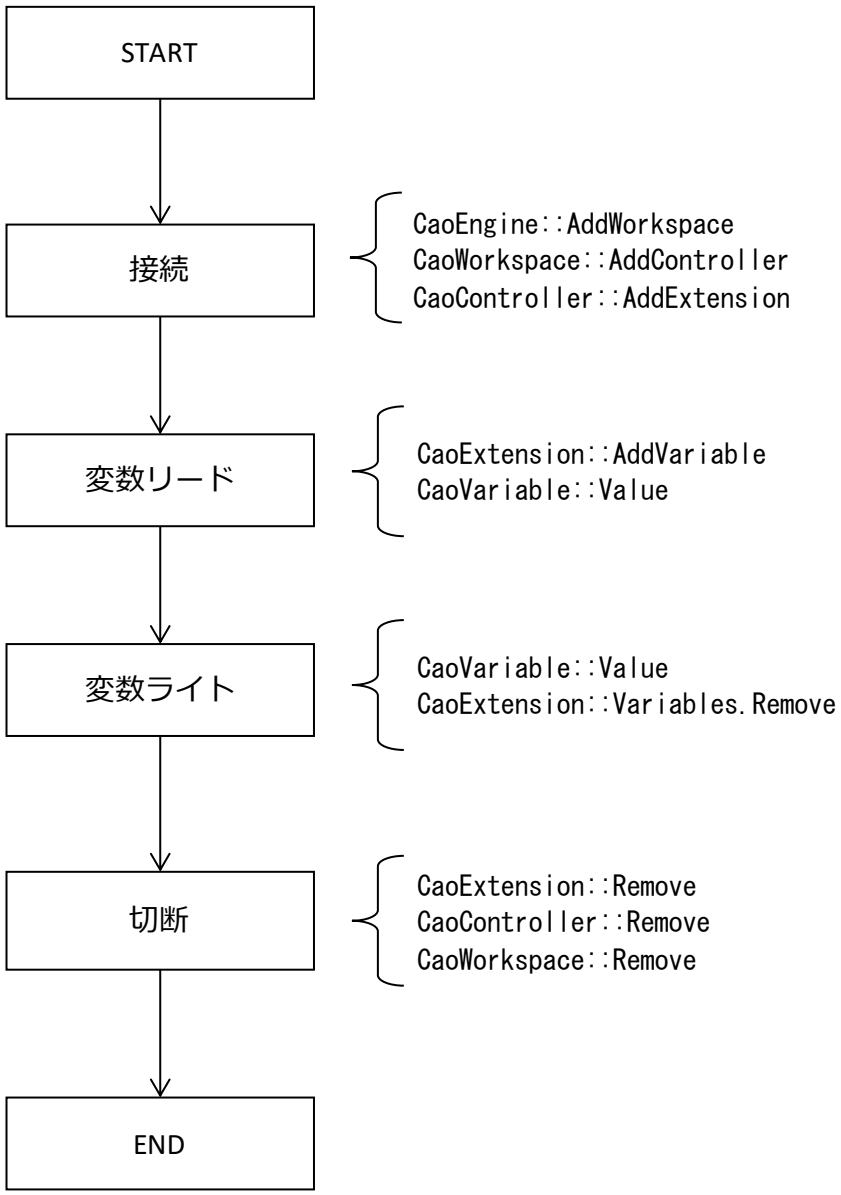


図 4-1 運転モード変更の流れ

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 ChangeOperationMode_STH-120.vb

```
' オブジェクト
Dim caoEng As CaoEngine
Dim caoWor As CaoWorkspace
Dim caoCon As CaoController
Dim caoExt As CaoExtension
```

```
Private Sub Main
    ' 接続
    Call Connect

    ' CaoVariable オブジェクトの生成
    Dim varMODE As CaoVariable
    Set varMODE = caoExt.AddVariable("@MODE")

    ' 運転モードを取得
    Dim value As String
    value = varMODE.value

    ' 運転モードを設定
    varMODE.value = "STANDBY"

    ' CaoExtension から CaoVariable を削除
    Call caoExt.Variables.Remove(varMODE.Index)
    ' CaoVariable の消去
    Set varMODE = Nothing

    ' 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 caoWor = caoEng.AddWorkspace("Workspace", "")
    ' CaoController オブジェクトの生成
    Set caoCon = caoWor.AddController("Controller", _
                                      "CaoProv.espec.TempHumi", _
                                      "", _
                                      "conn=com:1")
    ' CaoExtension オブジェクトの生成
    Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
End Sub

' 切断メソッド
Private Sub Disconnect()
    ' CaoController から CaoExtension を削除
    Call caoCon.Extensions.Remove(caoExt.Index)
    ' CaoExtension の消去
    Set caoExt = Nothing

    ' CaoWorkspace から CaoController を削除
```

```
    Call caoWor.Controllers.Remove(caoCon.Index)
    ' CaoController の消去
    Set caoCon = Nothing

    ' CaoEngine から CaoWorkspace を削除
    Call caoEng.Workspaces.Remove(caoWor.Index)
    ' CaoWorkspace の消去
    Set caoWor = Nothing

    ' CaoEngine の消去
    Set caoEng = Nothing
End Sub
```

4.1.1.1. TempHumi RS-485 デバイス接続

TempHumi RS-485 デバイス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．デバイス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オブジェクトとCaoExtension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．

```
    Dim caoEng As CaoEngine             ' CaoEngineオブジェクト用の変数
    Dim caoWor As CaoWorkspace          ' CaoWorkspaceオブジェクト用の変数
    Dim caoCon As CaoController         ' CaoControllerオブジェクト用の変数
    Dim caoExt As CaoExtension          ' CaoExtension オブジェクト用の変数
    Dim varMODE As CaoVariable          ' CaoVariable オブジェクト用の変数
```

(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 caoWor = 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ーを設定します．espec TempHumiプロバイダでは，

接続先のCOMポート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 caoCon = caoWor.AddController("Controller", _
                                  "CaoProv.espec.TempHumi", _
                                  "", _
                                  "conn=com:1")
```

(5) CaoExtensionオブジェクトを生成します．CaoExtensionオブジェクトを生成するには，使用するためのパラメーターを設定します．espec TempHumiプロバイダでは，接続先のアドレスをオプションで指定します．以下にコード例を示します．

```
' CaoExtensionオブジェクトの生成
Set caoExt = caoCon.AddExtension("Extension", "Addr = 1")
```

4.1.1.2. 変数の取得/設定

変数の値を取得/設定するには，アクセスしたい変数の CaoVariable オブジェクトを生成し，CaoVariable オブジェクトの Value プロパティを参照/設定します．Value プロパティを参照/設定する場合は，Value プロパティに合わせた型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．

```
' CaoVariableオブジェクトの生成
Dim varMODE As CaoVariable
Set varMODE = caoExt.AddVariable("@MODE")

' 運転モードを取得
Dim value As String
value = varMODE.value

' 運転モードを設定
varMODE.value = "STANDBY"

' CaoExtension から CaoVariable を削除
Call caoExt.Variables.Remove(varMODE.Index)
' CaoVariable の消去
Set varMODE = Nothing
```

4.1.1.3. TempHumi RS-485 デバイス切断

TempHumi RS-485 デバイス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．

```
' CaoController から CaoExtension を削除
Call caoCon.Extensions.Remove(caoExt.Index)
' CaoExtension の消去
Set caoExt = Nothing
' CaoWorkspace から CaoController を削除
```

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

4.2. TempHumi Ethernet デバイスの運転モードを変更するサンプルプログラミング

ここでは例として運転モードを変更する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 4-2 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 サンプルプログラムの要件

要件	説明
接続先	イーサネット(TCP/IP)で接続する
	接続先 IP アドレスは 192.168.0.2
処理内容	現在の運転モードを取得する
	運転モードをスタンバイ状態に変更する

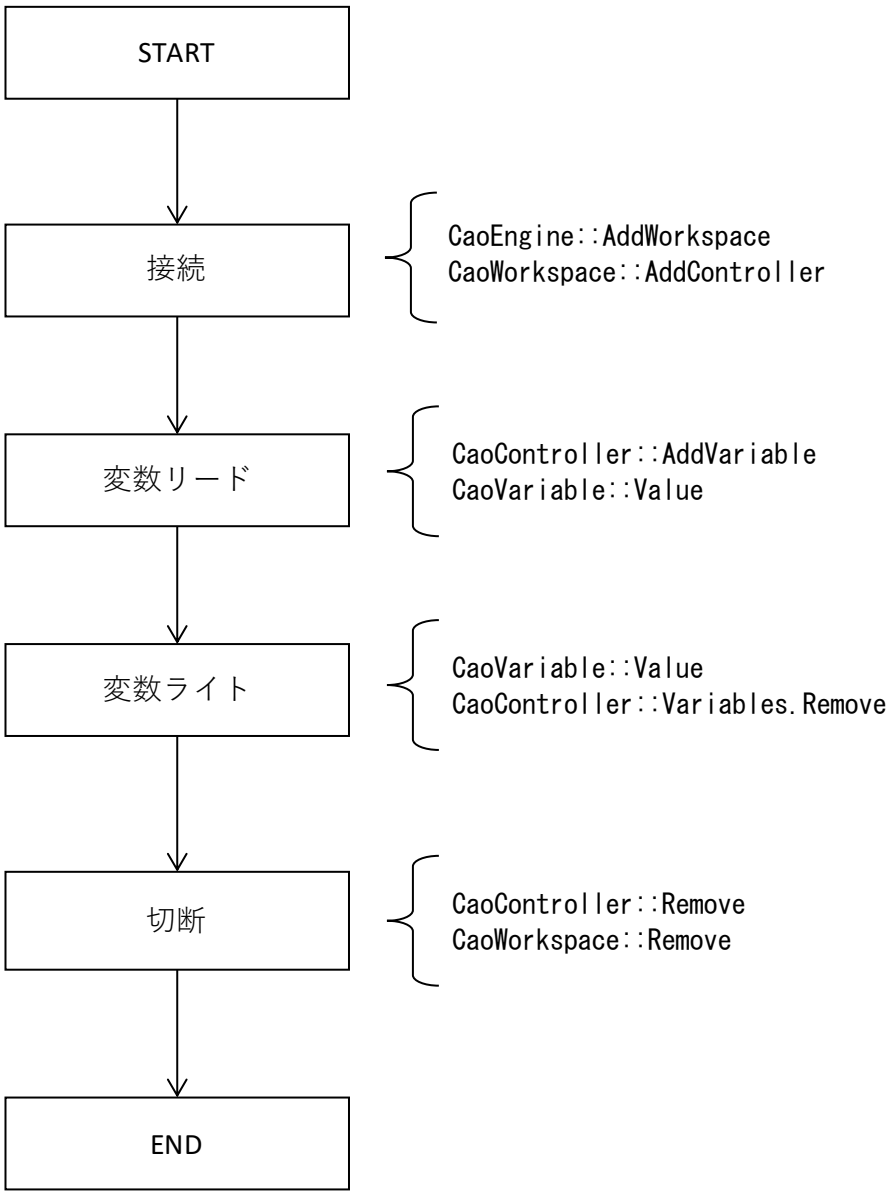


図 4-2 運転モード変更の流れ

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 ChangeOperationMode_SH-642.vb

```
' オブジェクト
Dim caoEng As CaoEngine
Dim caoWor As CaoWorkspace
Dim caoCon As CaoController
```

```
Private Sub Main
    ' 接続
    Call Connect

    ' CaoVariable オブジェクトの生成
    Dim varMODE As CaoVariable
    Set varMODE = caoCon.AddVariable("@MODE")

    ' 運転モードを取得
    Dim value As String
    value = varMODE.value

    ' 運転モードを設定
    varMODE.value = "STANDBY"

    ' CaoExtension から CaoVariable を削除
    Call caoCon.Variables.Remove(varMODE.Index)
    ' CaoVariable の消去
    Set varMODE = Nothing

    ' 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 caoWor = caoEng.AddWorkspace("Workspace", "")
    ' CaoController オブジェクトの生成
    Set caoCon = caoWor.AddController("Controller", _
                                      "CaoProv.espec.TempHumi", _
                                      "", _
                                      "conn=eth:192.168.0.2")
End Sub

' 切断メソッド
Private Sub Disconnect()
    ' CaoWorkspace から CaoController を削除
    Call caoWor.Controllers.Remove(caoCon.Index)
    ' CaoController の消去
    Set caoCon = Nothing

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

```
    ' CaoEngine の消去
    Set caoEng = Nothing
End Sub
```

4.2.1.1. TempHumi Ethernet デバイス接続

TempHumi RS-485 デバイス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．デバイス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オブジェクトとCaoExtension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．

```
    Dim caoEng As CaoEngine             ' CaoEngineオブジェクト用の変数
    Dim caoWor As CaoWorkspace          ' CaoWorkspaceオブジェクト用の変数
    Dim caoCon As CaoController         ' CaoControllerオブジェクト用の変数
    Dim varMODE As CaoVariable          ' CaoVariable オブジェクト用の変数
```

(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 caoWor = 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ーを設定します．espec TempHumiプロバイダでは，接続先のCOMポート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 caoCon = caoWor.AddController("Controller", _
                                      "CaoProv.espec.TempHumi", _
                                      "", _
                                      "conn=eth:192.168.0.2")
```

4.2.1.2. 変数の取得/設定

変数の値を取得/設定するには，アクセスしたい変数の CaoVariable オブジェクトを生成し，CaoVariable オブジェクトの Value プロパティを参照/設定します．Value プロパティを参照/設定する場合は，Value プロパティに合わせた型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．

```
    ' CaoVariableオブジェクトの生成
    Dim varMODE As CaoVariable
    Set varMODE = caoCon.AddVariable("@MODE")

    ' 運転モードを取得
    Dim value As String
    value = varMODE.value

    ' 運転モードを設定
    varMODE.value = "STANDBY"

    ' CaoController から CaoVariable を削除
    Call caoCon.Variables.Remove(varMODE.Index)
    ' CaoVariable の消去
    Set varMODE = Nothing
```

4.2.1.3. TempHumi Ethernet デバイス切断

TempHumi RS-485 デバイス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 caoWor.Controllers.Remove(caoCon.Index)
    ' CaoController の消去
    Set caoCon = Nothing
    ' CaoEngine から CaoWorkspace を削除
    Call caoEng.Workspaces.Remove(caoWor.Index)
    ' CaoWorkspace の消去
    Set caoWor = Nothing
    ' CaoEngine の消去
    Set caoEng = Nothing
```

5. espec TempHumi 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	アドレスオプションがありません．
0x80110002	アドレスが既に登録されています．
0x80110003	オブジェクトを追加できません．

また，本プロバイダは，デバイスのエラーコードを「0x8010****」でマスクして返します．

デバイスから送られてきたエラーメッセージを元に以下のエラーコードとなっています．（表 5-2 デバイスのエラーコード表参照）

表 5-2 デバイスのエラーコード表

エラー番号	エラーメッセージ	説明
0x80100001	COMMAND ERR	コマンドデータとして認識できない
0x80100002	CONTROLLER NOT READY-1	湿度制御をサポートしていない装置に湿度に関するコマンドデータを送った
0x80100003	CONTROLLER NOT READY-2	プログラム運転中以外の時に，プログラムに関するコマンドデータを送った
0x80100004	CONTROLLER NOT READY-3	電源オフ状態で，キーロック設定を行った
0x80100005	CONTROLLER NOT READY-4	設定変更不可のタイムシグナルを設定しようとした
0x80100006	CONTROLLER NOT READY-5	冷凍機未搭載のチャンバーに，冷凍機関連のコマンドを発行した
0x80100007	CONTROLLER NOT READY-6	ダンパ未搭載のチャンバーに，ダンパ関連のコマンドを発行した
0x80100008	DATA NOT READY	有効なデータがセットされていないプログラムパターンを実行しようとした
0x80100009	PARAMETER ERR	設定コマンドに必要なパラメーターが無い　もしくは　付加されたパラメーターが認識不可能
0x8010000A	DATA OUT OF RANGE	設定範囲を超える温(湿)度設定値をセットした
0x8010000B	PROTECT ON	キーロック設定有効時にデータの変更を行った
0x8010000C	PRGM WRITE ERR-1	編集／上書きモードを指定せずに，データの書き込みを行った

エラー番号	エラーメッセージ	説明
0x8010000D	PRGM WRITE ERR-2	編集中でないときに，編集に関するコマンドを発行した
0x8010000E	PRGM WRITE ERR-3	編集動作中に，上書き動作を行った
0x8010000F	PRGM WRITE ERR-4	上書き中に，編集動作を行った
0x80100010	PRGM WRITE ERR-5	上書き中でないときに，上書きに関するコマンドを送った
0x80100011	PRGM WRITE ERR-6	書き込み中と異なるプログラムパターンを指定した
0x80100012	PRGM WRITE ERR-7	ステップＮｏが，連続でない
0x80100013	PRGM WRITE ERR-8	繰り返し設定に誤りがある
0x80100014	PRGM WRITE ERR-9	運転中のプログラムを編集しようとした
0x80100015	PRGM WRITE ERR-10	有効なデータがセットされていないまま繰り返し設定や終了条件をセットした
0x80100016	PRGM WRITE ERR-11	無効なデータをセットしようとした
0x80100017	PRGM WRITE ERR-12	勾配設定ＯＮ時にさらし時間処理設定をセットした
0x80100018	PRGM WRITE ERR-13	湿度制御ＯＦＦ設定時に湿度勾配設定を行った
0x80100019	CMD_ERR	メインコマンドに誤りがある
0x8010001A	INVALID REQ	装置が対応できない機能を指定した
0x8010001B	CHB NOT READY	装置が受け付けられない状態の時にコマンドを指定した
0x8010001C	PARA ERR	オプションパラメーターに誤りがある
0x8010001D	TempHumi RS-485マニュアルの「1.4 通信機能で扱うデータ」の「表1.1 エラーメッセージ」に記載されていないエラーメッセージを受信した	

6. 付録

付録A. TempHumi プロバイダのコマンドとデバイスのコマンドの対応

本プロバイダのCaoControllerクラスのコマンドならびにCaoExtensionクラスのコマンドとデバイスのコマンドの対応は以下の通りです.

表 6-1 プロバイダのコマンドとデバイスのコマンド対応

プロバイダのコマンド		デバイスのコマンド
CaoController クラス	CaoExtension クラス	
GetROM		ROM?
GetROMDISP	―	ROM?,DISP
GetMASK		MASK?
SetMASK		MASK
GetTIMERMON	―	TIMER ON?
GetTIMERSET	―	TIMER USE?
GetTIMER0	―	TIMER LIST?,0
SetTIMER0	―	TIMER WRITE,NO0
GetTIMER1	―	TIMER LIST?,1
SetTIMER1	―	TIMER WRITE,NO1
GetTIMER2	―	TIMER LIST?,2
SetTIMER2	―	TIMER WRITE,NO2
GetKEYPROTECT	―	KEYPROTECT?
SetKEYPROTECT	―	KEYPROTECT
―	GetKEYLOCK	KEY PROTECT?
―	SetKEYLOCK	KEY PROTECT
GetTYPE		TYPE?
GetTEMP		TEMP?
SetTEMP		TEMP
SetSTEMP		TEMP
SetHTEMP		TEMP
SetLTEMP		TEMP
GetHUMI		HUMI?
SetHUMI		HUMI
SetSHUMI		HUMI
SetHHUMI		HUMI
SetLHUMI		HUMI
GetSET		SET?

プロパイダのコマンド		デバイスのコマンド
CaoController クラス	CaoExtension クラス	
SetSET		SET
GetCONSTANTSETTEMP	―	CONSTANT SET?TEMP
GetCONSTANTSETHUMI	―	CONSTANT SET?HUMI
GetCONSTANTSETREF	―	CONSTANT SET?REF
GetSYSTEMSETPTS	―	SYSTEM SET?PTS
GetSYSTEMSETPTC	―	SYSTEM SET?PTC
GetSYSTEMSETPTCOPT	―	SYSTEM SET?PTCOPT
SetTIMERMERASE	―	TIMER ERASE,NOタイマ―番号
SetTIMERON	―	TIMER,ON,タイマ―番号
SetTIMEROFF	―	TIMER,OFF,タイマ―番号
GetDATE	―	DATE?
SetDATE	―	DATE
GetTIME	―	TIME?
SetTIME	―	TIME
GetSRQ		SRQ?
SetSRQ		SRQ
GetALARM		ALARM?
GetMODE		MODE?
SetMODE		MODE
GetMODEDETAIL	―	MODE?,DETAIL
GetMONDETAIL	―	MON?,DETAIL
GetHEATER		%?
―	SetPOWER	POWER
GetRUNPRGDATA		RUN PRGM?
SetRUNPRGDATA		RUN PRGM

付録B. TempHumi プロパイダの変数とデバイスのコマンドの対応

本プロバイダのCaoControllerクラス変数ならびにCaoExtensionクラス変数とデバイスのコマンドの対応は以下の通りです.

表 6-2 プロバイダの変数とデバイスのコマンド対応

プロパイダの変数		デバイスのコマンド	
CaoController クラス	CaoExtension クラス	get_Value	put_Value
@ROM		ROM?	―

プロパイダの変数		デバイスのコマンド	
CaoController クラス	CaoExtension クラス	get_Value	put_Value
@ROMDISP	―	ROM?,DISP	―
@MASK		MASK?	MASK
@TIMERMON	―	TIMER ON?	―
@TIMERSET	―	TIMER USE?	―
@TIMER0	―	TIMER LIST?,0	TIMER WRITE,NO0
@TIMER1	―	TIMER LIST?,1	TIMER WRITE,NO1
@TIMER2	―	TIMER LIST?,2	TIMER WRITE,NO2
@TIMERERASE	―	―	TIMER ERASE,NOタイマー番号
@TIMERON	―	―	TIMER,ON,タイマー番号
@TIMEROFF	―	―	TIMER,OFF,タイマー番号
@KEYPROTECT	―	KEYPROTECT?	KEYPROTECT
―	@KEYLOCK	KEY PROTECT?	KEY PROTECT
@TYPE		TYPE?	―
@TEMPMON		TEMP?	―
@TEMP		―	TEMP
@STEMP		―	TEMP
@HTEMP		―	TEMP
@LTEMP		―	TEMP
@HUMIMON		HUMI?	―
@HUMI		―	HUMI
@SHUMI		―	HUMI
@HHUMI		―	HUMI
@LHUMI		―	HUMI
@SET		SET?	SET
@CONSTANTSETTEMP	―	CONSTANT SET?TEMP	―
@CONSTANTSETHUMI	―	CONSTANT SET?HUMI	―
@CONSTANTSETREF	―	CONSTANT SET?REF	―
@SYSTEMSETPTS	―	SYSTEM SET?PTS	―

プロパイダの変数		デバイスのコマンド	
CaoController クラス	CaoExtension クラス	get_Value	put_Value
@SYSTEMSETPTC	―	SYSTEM SET?PTC	―
@SYSTEMSETPTCOPT	―	SYSTEM SET?PTCOPT	―
@DATE	―	DATE?	DATE
@TIME	―	TIME?	TIME
@SRQ		SRQ?	―
@ALARM		ALARM?	―
@MODE		MODE?	MODE
@MODEDETAIL	―	MODE?,DETAIL	―
―	@MON	MON?	―
@MONDETAIL	―	MON?,DETAIL	―
@HEATER		%?	―
―	@POWER	―	POWER
@RUNPRGDATA		RUN PRGM?	RUN PRGM

付録C. 対応機種一覧

TempHumi RS-485 マニュアルならびに TempHumi Ethernet マニュアルに記載されている対応機種一覧を記載します.

表 6-3 RS-485 対応機種一覧

機種
SU-221
SU-241
SU-261
SU-641
SU-661
SH-221
SH-241
SH-261
SH-641
SH-661
MC-711T
MC-811T

LH-113
LHL-113
LHU-113
LU-113
LC-113
LC-123
LC-223
LG-113
LG-123
LCV-233
LCV-233P
LCV-243
LCV-243P
ST-110
ST-120
STH-120

表 6-4 Ethernet 対応機種一覧

機種
小型環境試験器(SU/SH)
安定性試験器(CSH)
安定性試験室(CWH)
ライトスペック恒温(恒湿)器(LU/LH)
小型超低温恒温器ミニサブゼロ(MC)
低温恒温恒湿器(CRH)