

様

仕様書
PComControlActiveX
PCCAx.ocx
シリアル通信制御用 ActiveX

NOTE:

Ver. 1.0.0.3

受領印欄

パナソニック システムネットワークス株式会社
ターミナルシステムビジネスユニット

福岡市博多区美野島 4 丁目 1 - 6 2
電話 : (092) 477-1592 (〒812-8531)

REF		REV
No.		D

発 行		
日付	2013. 04. 18	
確		
認		

・改定履歴

[illegible]

【目 次】

1.	適用範囲.....	1
2.	概要	1
2.1.	動作対象環境	1
2.2.	適用機種	1
3.	使い方.....	1
3.1.	開発環境	1
4.	情報	1
5.	APIリファレンス.....	2
6.	プロパティ.....	8
6.1.	GetPortNames プロパティ	8
6.2.	PortNo プロパティ	9
6.3.	BaudRate プロパティ.....	10
6.4.	DataBit プロパティ	11
6.5.	Parity プロパティ	12
6.6.	DSRCheck プロパティ	13
6.7.	CTSCheck プロパティ.....	14
6.8.	AutoHandshakeプロパティ	15
6.9.	TXBufferSizeプロパティ.....	16
6.10.	RXBufferSizeプロパティ.....	16
6.11.	SendDelayTime プロパティ	17
6.12.	WriteFileTimeOut プロパティ	17
6.13.	ACKTimer プロパティ	18
6.14.	ResponseTimer プロパティ	19
6.15.	DTRLineControl プロパティ	20
6.16.	RTSLineControl プロパティ	20
6.17.	BreakControl プロパティ.....	21
6.18.	GetCommStatusプロパティ.....	21
6.19.	Claimedプロパティ	22
6.20.	Versionプロパティ.....	22
6.21.	EventMaskプロパティ.....	23
6.22.	PrinterDeviceNameプロパティ	24
6.23.	LogFlag プロパティ	25
6.24.	LogFileFolder プロパティ.....	26
6.25.	SendData プロパティ.....	27
6.26.	ReceiveData プロパティ	27
6.27.	ReceiveDataLen プロパティ.....	28
6.28.	PollCmdComSts プロパティ.....	28
6.29.	PollCmdSen プロパティ	29
6.30.	PollCmdErr プロパティ	30
6.31.	PollCmdResData プロパティ	31
6.32.	HFRceiveReserve プロパティ	31
6.33.	HFRceiveResp プロパティ.....	32
6.34.	HFRceiveStat プロパティ	32
6.35.	HFRceiveData プロパティ	33
6.36.	HFRceiveDataLen プロパティ	33
6.37.	CancelFlag プロパティ	34

7. メソッド.....	35
7.1. ポートオープンメソッド pcxOpenPort.....	3 5
7.2. ポートクローズメソッド pcxClosePort.....	3 6
7.3. 送受信バッファークリアメソッド pcxComBufferClear	3 7
7.4. レスポンスプロパティクリアメソッド pcxResPropertyClear.....	3 8
7.5. ポーリングコマンド送受信メソッド pcxSendCommandRR.....	3 9
7.6. ポーリングコマンド送信メソッド pcxSendCommand.....	4 1
7.7. ポーリングレスポンス受信メソッド pcxReceiveResponse	4 2
7.8. 電文データ送受信メソッド pcxSendDataBCCRR	4 4
7.9. 電文データ送信メソッド pcxSendData.....	4 5
7.10. 電文データ受信メソッド pcxReceiveData	4 6
7.11. HF帯RFID電文データ送受信メソッド pcxSendDataHFRR.....	4 8
7.12. HF帯RFID電文データ受信メソッド pcxReceiveDataHF	4 9
7.13. UHF帯RFIDコマンド送受信メソッド pcxSendBinaryCommandRR	5 0
7.14. UHF帯RFIDコマンド送信メソッド pcxSendBinaryCommand	5 1
7.15. UHF帯RFIDコマンド受信メソッド pcxReceiveBinaryReport.....	5 2
7.16. バイナリデータ送信メソッド pcxSendBinaryData	5 3
7.17. バイナリデータ受信メソッド pcxReceiveBinaryData.....	5 4
7.18. プリントジョブクリアメソッド pcxPrintJobClear.....	5 5
7.19. SUM計算メソッド pcxDataSumCalc	5 5
7.20. ASCIIコード変換メソッド pcxCharAscHConv	5 6
7.21. JIS/Shift-JIS漢字コード変換メソッド pcxJISconvSJIS	5 7
8. イベント	58
8.1. ポーリングレスポンス受信イベント pcxReceiveResponseEvent	5 8
8.2. 電文データ受信イベント pcxReceiveDataBCCEvent.....	5 9
8.3. HF帯RFID電文データ受信イベント pcxReceiveDataHFEvent.....	6 0
8.4. UHF帯RFIDコマンド受信イベント pcxReceiveBinaryReportEvent.....	6 1
8.5. データ通信イベント pcxCommEvent	6 2
9. ログの構成	63
10. OCX使用上の注意事項.....	65
11. 補足資料.....	66
11.1. マルチスレッドの基本使用方法.....	6 6
11.2. マルチスレッドの動作説明.....	6 7
11.2.1. シングルスレッド（1ポート使用）の場合	6 7
11.2.2. マルチスレッド（同一ポート使用）の場合.....	6 8
11.2.3. マルチスレッド（別ポート使用）の場合.....	6 9
11.3. データ送受信処理について.....	7 0
11.3.1. 送信処理	7 0
11.3.2. 受信処理	7 1

1. 適用範囲

本仕様書はパナソニックシステムネットワークズ株式会社が提供する機器の上位アプリケーション（以降 APL と記す）のための RS-232C 通信制御を簡素化する ActiveX コントロール（PCCAx.ocx）のアプリケーションインターフェイス（API）仕様について記述します。

2. 概要

ドライバ PCCAx.ocx はパナソニックシステムネットワークズ株式会社が提供する機器を、Win32 から利用するための API を提供します。

2.1. 動作対象環境

本OCXの動作対象環境は以下のとおりです。

OS : Windows 2000 / Windows XP / Windows Server 2003 / Windows Vista / Windows Server 2008 / Windows7 / Windows8 / Windows Server 2012

※ すべて32ビット版のみ動作確認(Wow64上での動作確認含む)。

2.2. 適用機種

下記の機種でパナソニックシステムネットワークズ株式会社の標準通信プロトコルをサポートする機種

- ・ JT-KP41U / JT-KU4x シリーズ
- ・ KU-R3000 / KU-A3000 シリーズ
- ・ KU-R40000 シリーズ
- ・ KU-R10000 / KU-R18000 / KU-R13000 シリーズ
- ・ KU-R28000/A9000 シリーズ
- ・ KU-Z28000 / KU-Z9000 シリーズ
- ・ KU-A1400 シリーズ
- ・ KU-J5410 / J5510 / KU-J7100 シリーズ
- ・ KU-G5400 シリーズ

注記）コマンド仕様に関しては各機種のコマンド仕様書を参照してください。

3. 使い方

3.1. 開発環境

PCCAx.ocx は

- ・ Visual Studio 6.0 (C++,VB)
- ・ Visual Studio .net 2002(C#,VB)
- ・ Visual Studio .net 2003(C#,VB)
- ・ Visual Studio 2005(C#,VB)
- ・ Visual Studio 2008(C#,VB)
- ・ Visual Studio 2010(C#,VB)
- ・ Visual Studio 2012(C#,VB)
- ・ Internet Explorer 6,7,8,9,10 (Java Script, VBScript)

で利用可能（動作確認済）です。

4. 情報

ファイル名	PCCAx.ocx
Classid	8CD60D34-3E71-4CFC-A0D3-25F50B48073F
Codebase	PCCAx.ocx#VERSION=1,0,0,3

5. APIリファレンス

ここでは、PCCAx.ocxが提供するAPIについて説明する。

【設定関連プロパティの概要】

PCCAx.ocxが提供する設定関連のプロパティは以下の通り。

No	プロパティ名	型	アクセス	関連メソッド名
1	GetPortNames	String	Read	—
2	PortNo	Long	Read / Write	pcxOpenPort();
3	BaudRate	Long	Read / Write	
4	DataBit	String	Read / Write	
5	Parity	String	Read / Write	
6	DSRCheck	Boolean	Read / Write	
7	CTSCheck	Boolean	Read / Write	
8	AutoHandshake	Boolean	Read / Write	
9	TXBufferSize	Long	Read / Write	
10	RXBufferSize	Long	Read / Write	
11	SendDelayTime	Long	Read / Write	
12	WriteFileTimeOut	Long	Read / Write	
13	ACKTimer	Long	Read / Write	pcxSendCommandRR() pcxSendCommand() pcxReceiveResponse() pcxSendDataRR() pcxsendDataBCC() pcxReceiveDataBCC() pcxSendBinaryCommandRR() pcxSendBinaryCommand() pcxReciveBinaryReport() pcxSendBinaryData()
14	ResponseTimer	Long	Read / Write	pcxSendCommandRR() pcxReceiveResponse(), pcxSendDataRR() pcxReceiveDataBCC() pcxReceiveDataHF() pcxSendBinaryCommandRR() pcxReciveBinaryReport() pcxReciveBinaryData()
15	DTRLLineControl	Boolean	Read / Write	—
16	RTSLLineControl	Boolean	Read / Write	—
17	BreakControl	Boolean	Read / Write	—
18	GetCommStatus	String	Read	—
19	Claimed	Boolean	Read	—
20	Version	String	Read	—
21	EventMask	Long	Read / Write	pcxCommEvent()
22	PrinterDeviceName	String	Read / Write	pcxPrintJobClear()
23	LogFlag	Long	Read / Write	—
24	LogFileFolder	String	Read / Write	—

※ ☐ マルチスレッド環境使用時の初回のみコール対象

【設定関連プロパティの説明】

名前	説明
<i>GetPortNames</i>	使用可能なシリアルポートをCSV形式で列挙し示す (初期値:Null文字列) (例: COM1,COM2)
<i>PortNo</i>	オープンするシリアルポートの番号 (初期値: 1) (1~256)
<i>BaudRate</i>	シリアルポートの通信速度 (単位: bps) (初期値: 9600) (2400 / 4800 / 9600 / 19200 / 38400 / 57600 / 115200)
<i>DataBit</i>	シリアルポートのデータビット長 (初期値: 8) (7または8)
<i>Parity</i>	シリアルポートのパリティチェック (初期値: E) (N:なし/E:偶数/O:奇数)
<i>DSRCheck</i>	DSR信号線をチェックするかどうかのフラグ (初期値: True:する) (True:する/False:しない)
<i>CTSCheck</i>	CTS信号線をチェックするかどうかのフラグ (初期値: True:する) (True:する/False:しない)
<i>TXBufferSize</i>	COMポートの送信バッファサイズ (単位: Byte) (初期値: 2048 Byte)
<i>RXBufferSize</i>	COMポートの受信バッファサイズ (単位: Byte) (初期値: 2048 Byte)
<i>ACKTimer</i>	ACK待ちタイマー値 (単位: ms) (初期値: 1000 ms) (1~)
<i>SendDelayTime</i>	データ送信前に挿入する遅延時間 (単位: ms) (初期値: 0 ms)
<i>WriteFileTimeOut</i>	WriteFile APIの処理戻り時間 (単位: ms) (初期値: 5000 ms) (1~)
<i>ResponseTimer</i>	受信待ちタイマー値 (単位: ms) (初期値: 3000 ms) (1~)
<i>DTRLineControl</i>	DTR信号線のON/OFFを制御します (初期値: True) (True:ON/False:OFF)
<i>RTSLineControl</i>	RTS信号線のON/OFFを制御します (初期値: True) (True:ON/False:OFF)
<i>BreakControl</i>	Break信号のON/OFFを制御します (初期値: False) (True:ON/False:OFF)
<i>GetCommStatus</i>	DSR・CTS・RLSDの信号線状態を取得します (初期値: 000) (例: DSR_ON・CTS_ON・RLSD_OFF = 110)
<i>Claimed</i>	ポートがOPENされ、排他が行われている事を示す。 (True: 排他されている/True: 排他されていない)
<i>Version</i>	コントロールのバージョンを示す (例: 1, 0, 0, 0)
<i>EventMask</i>	pcxCommEventで監視を行うイベントのマスク値を設定する。
<i>PrinterDeviceName</i>	プリンターの名前を指定します。(Printer Device機能を持つ機器対象)
<i>LogFlag</i>	通信ログを取得するかどうかのフラグ (初期値: False:しない) (True:ON/False:OFF)
<i>LogFileFolder</i>	通信ログを保存するフォルダーを指定します。(初期値: User Desktop)

【データ入出力関連プロパティの概要】

PCCAx.ocxが提供するデータ入出力関連のプロパティは以下の通り。

No	プロパティ名	型	アクセス	桁	関連メソッド名
1	SendData	String	Read / Write	-	pcxSendCommand() pcxSendDataBCC() pcxSendBinaryCommand() pcxSendBinaryData() pcxSendCommandRR() pcxSendDataRR() pcxSendBinaryCommandRR()
2	ReceiveData	String	Read	可変長	pcxReceiveDataBCC() pcxReceiveBinaryReport() pcxReceiveBinaryData() pcxSendDataRR() pcxSendBinaryCommandRR()
3	ReceiveDataLen	Long	Read	-	
4	PollCmdComSts	String	Read	2 Byte	pcxReceiveResponse() pcxSendCommandRR()
5	PollCmdSen	String	Read	可変長	
6	PollCmdErr	String	Read	2 Byte	
7	PollCmdResData	String	Read	可変長	
8	HFRceiveReserve	String	Read	1 Byte	pcxSendDataHFRR() pcxReceiveDataHF()
9	HFRceiveResp	String	Read	2 Byte	
10	HFRceiveStat	String	Read	2 Byte	
11	HFRceiveData	String	Read	可変長	
12	HFRceiveDataLen	Long	Read	-	
13	CancelFlag	Boolean	Read / Write	-	pcxSendCommandRR() pcxReceiveResponse() pcxReceiveDataHF() pcxSendBinaryCommandRR() pcxReceiveBinaryReport() pcxSendDataBCCRR() pcxReceiveData() pcxSendDataHFRR()

※ ☐ マルチスレッド環境使用時の初回のみコール対象

【データ入出力関連プロパティの説明】

名前	説明
SendData	機器に送信する文字列データ（初期値：Null文字列）
ReceiveData	機器より受信したデータ部分を示す（初期値：Null文字列）
ReceiveDataLen	機器より受信したデータ部分（ReceiveData）のデータ長を示す （初期値：0）
PollCmdComSts	レスポンスのコマンドコード(=COM\$)・（初期値：Null文字列） レディステータスのステータス情報(=STS\$)を示す
PollCmdSen	レディステータスのセンサー情報(=SEN\$)を示す（初期値：Null文字列）
PollCmdErr	レスポンスのエラーコード(=ERR\$)・（初期値：Null文字列）
PollCmdResData	レスポンスのデータ部分(=DATA\$)を示す（初期値：Null文字列） (センサー情報以外の追加情報がある場合も、以降の情報は格納される)
HFRceiveReserve	1 レスポンス目のReserve部分を示す(1Byte)（初期値：Null文字列）
HFRceiveResp	1 レスポンス目のコード部分を示す(2Byte)（初期値：Null文字列）
HFRceiveStat	1 レスポンス目ステータス部分を示す(2Byte)（初期値：Null文字列）
HFRceiveData	1 又は 2 レスポンス目のデータ部分を示す（初期値：Null文字列）
HFRceiveDataLen	1 又は 2 レスポンス目のデータ部分のデータ長を示す（初期値：0）
CancelFlag	メソッド処理中もしくはレスポンス待ちを強制中止する。(初期値:False) (True:する/False:しない)

【メソッドの概要】

PCCAx.ocxが提供するAPIは以下のとおりです。

No	メソッド名	機能
1	pcxOpenPort()	ポートオープンメソッド
2	pcxClosePort()	ポートクローズメソッド
3	pcxComBufferClear()	送受信バッファークリアメソッド
4	pcxResPropertyClear()	レスポンスプロパティクリアメソッド
5	pcxSendCommandRR()	ポーリングコマンド送受信メソッド
6	pcxSendCommand()	ポーリングコマンド送信メソッド
7	pcxReceiveResponse()	ポーリングレスポンス受信メソッド
8	pcxSendDataBCCRR()	電文データ送受信メソッド
9	pcxSendData()	電文データ送信メソッド
10	pcxReceiveData()	電文データ受信メソッド
11	pcxSendDataHFRR()	HF 帯 RFID 電文データ送受信メソッド
12	pcxReceiveDataHF()	HF 帯 RFID 電文データ受信メソッド
13	pcxSendBinaryCommandRR()	UHF 帯 RFID コマンド送受信メソッド
14	pcxSendBinaryCommand()	UHF 帯 RFID コマンド送信メソッド
15	pcxReceiveBinaryReport()	UHF 帯 RFID コマンド受信メソッド
16	pcxSendBinaryData()	バイナリデータ送信メソッド
17	pcxReceiveBinaryData()	バイナリデータ受信メソッド
18	pcxPrintJobClear ()	プリントジョブクリアメソッド
19	pcxDataSumCalc()	PCC SUM 計算メソッド
20	pcxCharAschConv()	ASCII コード変換メソッド
21	pcxJISconvSJIS ()	JIS/Shift-JIS 漢字コード変換メソッド

※ ☐ マルチスレッド環境使用時の初回のみコール対象

また、各APIにおける戻り値の定数は以下のとおりです。

定数名	値
RET_PCX_SUCCESS	0
RET_PCX_EVENT_SUCCESS	1
RET_PCX_ACK	2
RET_PCX_NAK	3
RET_PCX_RESPONSE	10
RET_PCX_READY	11
RET_PCX_BUSY	12
RET_PCX_CANCELED	20
RET_PCX_FAILURE	30
RET_PCX_ERR_PARAM	100
RET_PCX_ERR_PORT	101
RET_PCX_ERR_RES_TIMEOUT	110
RET_PCX_ERR_CTS_TIMEOUT	111
RET_PCX_ERR_DSR_TIMEOUT	112
RET_PCX_ERR_BUSY_TIMEOUT	113
RET_PCX_ERR_BCC_RETRY	120
RET_PCX_ERR_BUFFEROVER	130
RET_PCX_ERR_BUSY	140
RET_PCX_ERR_COMM_WRITE	200
RET_PCX_ERR_COMM_READ	201

各定数の説明は、各APIの説明の部分を参照してください。

送信時のRET_PCX_ERR_COMM_WRITE (200)WIN32APIエラーとは、WriteFile APIの戻り値がFALSEだった場合を表しています。

受信時のRET_PCX_ERR_COMM_READ (201)WIN32APIエラーとは、ReadFile APIの戻り値がFALSEだった場合を表しています。

信号線のチェック：

DSRCheck / CTSCheckが”True”の場合はデータの送受信時とデータ送受信中には、シリアル信号線(DSR /CTS)のチェックを行います。

データ送受信時DSR線はこれはCTS線のチェックよりも先に、その方法は以下のとおりです。

- ・送信時 : 送信前にDSRをチェックし、オフであればタイムアウトまで待ちます。
- ・受信時 : 受信データなしであればDSRをチェックし、オフであればすぐエラーとします。

データの送受信メソッドについては信号線のチェックはこの方法で行います。

【パラメータの型と戻り値の型】

型	意味
Long	32 ビット長の符号付き整数
String	文字列
Boolean	True(0 以外) と False(0) を有効値とする整数

【イベントの概要】

PCCAx.ocx が提供するイベントは以下の通りです。

No	関数名	機能
1	pcxCommEvent()	通信イベント
2	pcxReceiveResponseEvent()	ポーリングレスポンス受信イベント
3	pcxReceiveDataBCCEvent()	電文データ受信イベント
4	pcxReceiveDataHFEvent()	HF 帯 RFID 電文データ受信イベント
5	pcxReceiveBainaryReportEvent()	UHF 帯 RFID コマンド受信イベント

※ ☐ マルチスレッド環境使用時の初回のみコール対象

機種別使用する PCCAx.ocx メソッドの一覧表

【機種別使用関数一覧】

機種	処理形式	関数名	内容	頁
全機種共通		pcxOpenPort	ポートオープンメソッド	35
		pcxClosePort	ポートクローズメソッド	36
		pcxComBufferClear	送受信バッファークリアメソッド	37
		pcxResPropertyClear	レスポンスプロパティクリアメソッド	38
	イベント受信	pcxCommEvent	データ通信イベント	62
KU-R3000/A3000 KU-Z28000/Z9000 KU-R28000/A9000 KU-A1400 シリーズ KU-J5410/J5510	ポーリング処理	pcxSendCommandRR	ポーリングコマンド送受信メソッド	39
		pcxSendCommand	ポーリングコマンド送信メソッド	41
		pcxReceiveResponse	ポーリングレスポンス受信メソッド	42
	イベント受信	pcxReceiveResponseEvent	ポーリングレスポンス受信イベント	58
KU-R40000 シリーズ KU-R10000 シリーズ KU-R13000 シリーズ KU-R18000 シリーズ (簡易 POS 接続仕様)	コマンド・レスポンス処理	pcxSendDataBCCRR	電文データ送受信メソッド	44
		pcxSendData	電文データ送信メソッド	45
		pcxReceiveData	電文データ受信メソッド	46
	イベント受信	pcxReceiveDataBCCEvent	電文データ受信イベント	59
KU-J7100(UHF) KU-J7100(HF) ポーリング インタフェース	ポーリング処理	pcxSendCommandRR	ポーリングコマンド送受信メソッド	39
		pcxSendCommand	ポーリングコマンド送信メソッド	41
		pcxReceiveResponse	ポーリングレスポンス受信メソッド	42
	イベント受信	pcxReceiveResponseEvent	ポーリングレスポンス受信イベント	58
UHF RFID インタフェース	UHF 帯 RFID 処理	pcxSendBinaryCommandRR	UHF 帯 RFID コマンド送受信メソッド	50
		pcxSendBinaryCommand	UHF 帯 RFID コマンド送信メソッド	51
		pcxReceiveBinaryReport	UHF 帯 RFID コマンド受信メソッド	52
	イベント受信	pcxReceiveBinaryReportEvent	UHF 帯 RFID コマンド受信イベント	61
HF RFID インタフェース	コマンド・レスポンス処理	pcxSendDataHFRR	HF 帯 RFID 電文データ送受信メソッド	48
		pcxSendData(0,0,0 以外)	電文データ送信メソッド	45
		pcxReceiveDataHF	HF 帯 RFID 電文データ受信メソッド	49
	イベント受信	pcxReceiveDataHFEvent	HF 帯 RFID 電文データ受信イベント	60
KU-U7200(UHF) シリーズ	UHF 帯 RFID 処理	pcxSendBinaryCommandRR	UHF 帯 RFID コマンド送受信メソッド	50
		pcxSendBinaryCommand	UHF 帯 RFID コマンド送信メソッド	51
		pcxReceiveBinaryReport	UHF 帯 RFID コマンド受信メソッド	52
	イベント受信	pcxReceiveBinaryReportEvent	UHF 帯 RFID コマンド受信イベント	61
KU-G5400 シリーズ	コマンド・レスポンス処理	pcxSendDataHFRR	電文データ送信メソッド	48
		pcxSendData	電文データ受信メソッド	45
		pcxReceiveDataHF	HF 帯 RFID 電文データ受信メソッド	49
	イベント受信	pcxReceiveDataHFEvent	HF 帯 RFID 電文データ受信イベント	60

【その他関数一覧】

機種	処理形式	関数名	内容	頁
自由度を持たせる為 の関数群	データ無加工 処理	pcxSendData(1,0 以外,0)	データ送信メソッド (ACK/NAK 応答待ちあり)	45
		pcxSendData(0,0,0)	データ送信メソッド (ACK/NAK 応答待ちなし)	45
		pcxReceiveData(0,0,0)	データ受信メソッド	46
	バイナリデータ 無加工処理	pcxSendBinaryData	バイナリデータ送信メソッド	53
		pcxReceiveBinaryData	バイナリデータ受信メソッド	54

6. プロパティ

6.1. GetPortNames プロパティ

【機能】

利用可能なシリアルポート名をレジストリから取得列挙し、CSV形式で格納し示す

【プロパティ】

GetPortNames

【データ型】

String

【アクセス】

Read

【初期値】

Null文字列

【値】

なし

【例】

"COM1,COM2,COM5"

【解説】

COM1,COM2 が実行対象 PC に存在する場合、"COM1,COM2"が **GetPortNames プロパティ**に、設定されます。

使用可能なシリアルポートが存在しない場合、Null 文字列が **GetPortNames プロパティ**に、設定されます。

※本プロパティは、常時取得可能です

6.2. PortNo プロパティ

【機能】

オープンするシリアルポートの番号を指定する。

【プロパティ】

PortNo

【データ型】

Long

【アクセス】

Read/Write

【初期値】

1

【値】

1から256までのポート番号

※値外の場合は、以前の設定

【例】

COM12 = "12"

【解説】

オープンメソッドを呼び出す前に本プロパティを設定しておく必要がある。

オープンメソッド発行後、クローズメソッドを呼び出す前に本プロパティを変更してはならない。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

BaudRate, DataBit, Parity, DSRCheck, CTSCheck, TXBufferSize, RXBufferSize

6.3. BaudRate プロパティ

【機能】

シリアルポートの通信速度を指定する。

【プロパティ】

BaudRate

【データ型】

Long

【アクセス】

Read/Write

【初期値】

9600

【値】

ボーレート値 2400 / 4800 / 9600 / 19200 / 38400 / 57600 / 115200

※値外の場合は、以前の設定。但し、115200以上の値は有効とする。

【例】

115200bps = “115200”

【解説】

オープンメソッドを呼び出す前に本プロパティを設定しておく必要がある。

オープンメソッド発行後、クローズメソッドを呼び出す前に本プロパティを変更してはならない。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,DataBit,Parity,DSRCheck,CTSCheck,TXBufferSize,RXBufferSize

6.4. DataBit プロパティ

【機能】

シリアルポートのデータビット長を指定する。

【プロパティ】

DataBit

【データ型】

String

【アクセス】

Read/Write

【初期値】

"8"

【値】

7または8

※値外の場合は、以前の設定

【例】

7bit = "7"

【解説】

オープンメソッドを呼び出す前に本プロパティを設定しておく必要がある。
オープンメソッド発行後、クローズメソッドを呼び出す前に本プロパティを変更してはならない。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,Parity,DSRCheck,CTSCheck,TXBufferSize,RXBufferSize,

6.5. Parity プロパティ

【機能】

シリアルポートのパリティチェックを指定する。

【プロパティ】

Parity

【データ型】

String

【アクセス】

Read/Write

【初期値】

"E"

【値】

N : なし / E : 偶数 / O : 奇数

※値外の場合は、以前の設定

【例】

奇数 = "O"

【解説】

オープンメソッドを呼び出す前に本プロパティを設定しておく必要がある。

オープンメソッド発行後、クローズメソッドを呼び出す前に本プロパティを変更してはならない。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,DSRCheck,CTSCheck,TXBufferSize,RXBufferSize

6.6. DSRCheck プロパティ

【機能】

データ送受信メソッド実行時またはメソッド実行中シリアルポートのDSR信号チェックの有無を指定する。

【プロパティ】

DSRCheck

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

True

【値】

True : チェックする／False : チェックしない

【解説】

データ送受信メソッド実行時と実行中にDSR信号線のチェックを行う。

オープンメソッドを呼び出す前に本プロパティを設定しておく必要がある。

オープンメソッド発行後、クローズメソッドを呼び出す前に本プロパティを変更してはならない。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,Parity,CTSCheck,TXBufferSize,RXBufferSize

6.7. CTSCheck プロパティ

【機能】

データ送受信メソッド実行時またはメソッド実行中シリアルポートのCTS信号チェックの有無を指定する。

【プロパティ】

CTSCheck

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

True

【値】

True : チェックする／False : チェックしない

【解説】

データ送受信メソッド実行時と実行中にCTS信号線のチェックを行う。

オープンメソッドを呼び出す前に本プロパティを設定しておく必要がある。

オープンメソッド発行後、クローズメソッドを呼び出す前に本プロパティを変更してはならない。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,Parity,DSRCheck,TXBufferSize,RXBufferSize

6.8. AutoHandshakeプロパティ

【機能】

デバイス間通信において、フロー制御を行います。（通常はTrue）

【プロパティ】

AutoHandshake

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

True

【値】

True : する / False : しない

【解説】

デバイス間通信において、送信／受信デバイス間で送信の停止／再開などをCOMの信号線を用いて通信制御を行います。（ハードウェアフロー制御）
（物理COMポートのIOポート制御が可能なポートで有効です）

【注記】

弊社の機器では全て“True”状態で問題ありません。（通常は変更しないでください）
pcxOpenPortメソッドを実行する前に必ず設定すること。
pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。
“True”状態時ではDTRLineControlプロパティ、RTSLineControlプロパティは使用できません。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo, BaundRate, DataBit, Parity, DSRCheck, CTSCheck,
TXBufferSize, RXBufferSize, DTRLineControl, RTSLineControl

6.9. TXBufferSizeプロパティ

【機能】

シリアルポートの送信バッファサイズを設定する

【プロパティ】

TXBufferSize

【データ型】

Long

【アクセス】

Read/Write

【初期値】

2048

【値】

1～ （単位：Byte）

【解説】

機器の最大送信電文によりサイズの変更をして下さい。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,Parity,DSRCheck,CTSCheck,RXBufferSize

6.10. RXBufferSizeプロパティ

【機能】

シリアルポートの受信バッファサイズを設定する

【プロパティ】

RXBufferSize

【データ型】

Long

【アクセス】

Read/Write

【初期値】

2048

【値】

1～ （単位：Byte）

【解説】

機器の最大送信電文によりサイズの変更をして下さい。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,Parity,DSRCheck,CTSCheck,TXBufferSize

6.11. SendDelayTime プロパティ

【機能】

データ送信前に挿入する遅延時間（単位はmsec）

【プロパティ】

SendDelayTime

【データ型】

Long

【アクセス】

Read/Write

【初期値】

0

【値】

0～ （単位：ms）

【解説】

通常は初期値のままお使いください。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,Parity,DSRCheck,CTSCheck,TXBufferSize,RXBufferSize
WriteFileTimeout

6.12. WriteFileTimeOut プロパティ

【機能】

WriteFile API を実行した際に処理が戻るまでの最大待ち時間（単位はmsec）

【プロパティ】

WriteFileTimeOut

【データ型】

Long

【アクセス】

Read/Write

【初期値】

5000

【値】

0～ （単位：ms）

【解説】

通常は初期値のままお使いください。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連メソッド】

pcxOpenPort()

【関連プロパティ】

PortNo,BaundRate,DataBit,Parity,DSRCheck,CTSCheck,TXBufferSize,RXBufferSize
SendDelayTime

6.13. ACKTimer プロパティ

【機能】

ACK/NAKレスポンス待ち及びCTS ON待ちを必要とするメソッドのタイマーを指定する。

(単位 : ms)

CTSCheckプロパティが (False : しない) の場合は、CTS ON待ちは無効。

【プロパティ】

ACKTimer

【データ型】

Long

【アクセス】

Read/Write

【初期値】

1000 // 1秒

【値】

1～

【例】

10秒 = 10000

【解説】

通常通信 : ACK(x06) / NAK(x15)

UHF帯通信 : ACK(x16x06) / NAK(x16x15)

【関連メソッド】

pcxSendCommandRR(),pcxSendCommand(),pcxReceiveResponse(),
pcxSendDataRR(),pcxsendDataBCC(),pcxReceiveDataBCC(),
pcxSendBinaryCommandRR(),pcxSendBinaryCommand(),
pcxReciveBinaryReport(),pcxSendBinaryData()

6.14. ResponseTimer プロパティ

【機能】

レスポンス待ちを必要とするメソッドのタイマーを指定する。(単位 : ms)

【プロパティ】

ResponseTimer

【データ型】

Long

【アクセス】

Read/Write

【初期値】

3000 // 3秒

【値】

0～

※ 本プロパティに0を設定し、各メソッド処理においてイベント処理を指定した場合イベント処理内でレスポンス無限待ちとなります。無限待ち状態を中断させる場合は、CancelFlagプロパティに"True"を設定してください。

【例】

60秒 = 60000

【解説】

関連メソッド対象のデータ電文を受信するタイマーを指定します。

【関連メソッド】

pcxSendCommandRR(),pcxReceiveResponse(),
pcxSendDataRR(),pcxReceiveDataBCC(),pcxReceiveDataHF(),
pcxSendBinaryCommandRR(),pcxReciveBinaryReport(),
pcxReciveBinaryData()

6.15. DTRLineControl プロパティ

【機能】

DTR信号線のON/OFFを制御します

【プロパティ】

DTRLineControl

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

True

【値】

True : ON / False : OFF

【解説】

ホスト（PC）側の電源信号ラインのON/OFF制御を行います。
AutoHandshakeプロパティが"False"状態のみ使用可能

【注意】

- ・ 物理COMポートのみ制御可能
（USBCOM等のIOポートが存在しないものは出来ません）
- ・ pcxOpenPortメソッド実行後のポートオープン中のみ有効

【関連プロパティ】

AutoHandshake, RTSLineControl

6.16. RTSLineControl プロパティ

【機能】

RTS信号線のON/OFFを制御します

【プロパティ】

RTSLineControl

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

True

【値】

True : ON / False : OFF

【解説】

ホスト（PC）側の送信要求信号ラインのON/OFF制御を行います。
AutoHandshakeプロパティが"False"状態のみ使用可能

【注意】

- ・ 物理COMポートのみ制御可能
（USBCOM等のIOポートが存在しないものは出来ません）
- ・ pcxOpenPortメソッド実行後のポートオープン中のみ有効

【関連プロパティ】

AutoHandshake, CTSLineControl

6.17. BreakControl プロパティ

【機能】

文字送信を強制的に中断し、送信回線を切断状態にします。

【プロパティ】

BreakControl

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

False

【値】

True : ON / False : OFF

【解説】

文字送信を強制的に中断し、送信回線を切断状態にしFalseが指定されるまで解除されません。

未送信データはクリアされません。

【注意】

- ・ 物理COMポートのみ制御可能
(USBCOM等のIOポートが存在しないものは制御出来ません)
- ・ pcxOpenPortメソッド実行後のポートオープン中のみ有効

6.18. GetCommStatus プロパティ

【機能】

ポートがOPENされている状態で、DSR・CTS・RLSDの信号線状態を示す。

【プロパティ】

GetCommStatus

【データ型】

String

【アクセス】

Read

【桁】

3 Byte

【初期値】

"000"

【値】

DSR・CTS・RLSDの順番で (0=OFF / 1=ON) の3 Bitで示す

例 : DSR_ON・CTS_OFF・RLSD_ON = "101"

【解説】

ポートがClose状態ではオール0 ("000") を示す

本プロパティは常時取得可能。

6.19. Claimedプロパティ

【機能】

ポートがOPENされ、排他が行われている事を示す。

【プロパティ】

Claimed

【データ型】

Boolean

【アクセス】

Read

【初期値】

False

【値】

True : 排他されている / False : 排他されていない

【解説】

本プロパティは**pcxOpenPort**メソッドによりTRUEになり、**pcxClosePort**メソッドによりFalseになる。

本プロパティは常時取得可能。

【関連メソッド】

pcxOpenPort(), **pcxClosePort()**

6.20. Versionプロパティ

【機能】

コントロールのバージョンを示す。

【プロパティ】

Version

【データ型】

String

【アクセス】

Read

【値】

"1, 0, 0, 0"

【解説】

本プロパティは常時取得可能。

バージョン情報が正常に取得出来なかった場合は“Faild”の値が返却されます

6.21. EventMaskプロパティ

【機能】

pcxCommEventメソッドで監視を行うイベントのマスク値を設定する。

【形式】

LONG EventMask;

【アクセス】

Read/Write

【値】

マスク値として以下の値のOR値を指定する。

値	説明
EVT_PCX_RXCHAR	入力バッファにデータを受信した。
EVT_PCX_TXEMPTY	出力バッファの最後の文字を送信した。
EVT_PCX_ERR_BUFFER_OVERFLOW	入力バッファサイズを越えて受信した。
EVT_PCX_CTS	CTS (送信可) 信号の状態が変わった。
EVT_PCX_DSR	DSR (データセットレディ) 信号の状態が変わった。
EVT_PCX_RLSD	RLSD (受信線信号検出)信号の状態が変わった。
EVT_PCX_BREAK	Break信号を検出した。
EVT_PCX_RING	呼び出し信号を検出した。
EVT_PCX_ERR_FRAME	フレーミングエラーが発生した。
EVT_PCX_ERR_OVERRUN	オーバーランエラーが発生した。
EVT_PCX_ERR_PARITY	パリティエラーが発生した。

実際の値は以下のように定義する。

定数名	値(16進数)
EVT_PCX_RXCHAR	0x00000001
EVT_PCX_TXEMPTY	0x00000002
EVT_PCX_ERR_BUFFER_OVERFLOW	0x00000004
EVT_PCX_CTS	0x00000008
EVT_PCX_DSR	0x00000010
EVT_PCX_RLSD	0x00000020
EVT_PCX_BREAK	0x00000040
EVT_PCX_RING	0x00000100
EVT_PCX_ERR_FRAME	0x00000200
EVT_PCX_ERR_OVERRUN	0x00000400
EVT_PCX_ERR_PARITY	0x00000800
EVT_PCX_ALL	0x0000FFFF

【解説】

マスク値として指定されたイベントのみが通知される。

本プロパティは常時設定・取得可能。

【注意】

弊社の“Virtual COM Driver”（USBによる仮想COM）と併用する場合、USBケーブルが突然切断された状態になった時に、“Virtual COM Driver”は“Break信号検出”の信号を返してきますので、EventMaskプロパティにて“EVT_PCX_BREAK”を取得許可するよう実装しこのときにポートクローズメソッド（pcxClosePort）にて、即座にポートをクローズするように実装してください。

（ポートがオープンのままでは、ドライバーがアンロード又はリロードできない場合があります）

処理手順：

USB ケーブル切断 ⇒ ブレーク信号発生（イベント取得） ⇒ ポートクローズ処理

6.22. PrinterDeviceNameプロパティ

【機能】

プリンターの名前を指定する。

【プロパティ】

PrinterDeviceName

【データ型】

String

【アクセス】

Read/WRITE

【値】

例 : “Rewritable Card Printer (40a)”

【解説】

pcxPrintJobClear()メソッドで使用する、プリンター名を指定する。

【関連メソッド】

pcxPrintJobClear()

6.23. LogFlag プロパティ

【機能】

本プロパティを設定することにより、本ActiveXの通信ログ出力を行います。

【プロパティ】

LogFlag

【データ型】

Long

【アクセス】

Read/Write

【初期値】

0

【値】

0 : ログを出力しない

1 : ログファイルを出力する (エラーと各種情報)

2 : ログファイルを出力する (エラーのみ)

3 : OutputDebugString() APIでログ出力する (エラーと各種情報)

4 : OutputDebugString() APIでログ出力する (エラーのみ)

※上記の値以外は初期値 : 0に設定されます

【解説】

各メソッドのデータの入出力通信ログを出力します。

デバッグ時のご使用ください。

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連プロパティ】

LogFileFolder

6.24. LogFileFolder プロパティ

【機能】

ログファイル出力設定時のパスとファイル名を設定します。

【プロパティ】

LogFileFolder

【データ型】

String

【アクセス】

Read/Write

【初期値】

PCCAx_%03d.log (パス無指定時 = User Desktop)

【値】

[Path]+[FileName]+[%03d]+[.]+[拡張子]

例 : "C:¥¥LOG¥¥PCCAx_%03d.log"

存在しないパス等が指定されたときは、ログは出力されません。

【解説】

注) LogFlagプロパティの値が1・2の時はReadのみ

出力するログファイル名のフォーマット

[Path]+[FileName]+[%03d]+[.]+[拡張子]

[%03d] = 通信ポート出力指定 (3桁出力の場合)

例 : LogFileName = C:¥¥Log¥¥pcxTest_%02d.logで指定した場合

ポートCOM8にて通信を行うと、ログがCドライブのLogフォルダのpcxTest_08.logに出力されます。

エスケープシーケンス'¥'は、プログラム内では文字として扱われないので、'¥¥'と指定します。

※ デフォルトは "PCCAx_%03d.log" (実行しているデスクトップに出力されます)

【注記】

pcxOpenPortメソッドを実行する前に必ず設定すること。

pcxOpenPortメソッド実行後、ポートオープン中はReadのみ可能。

【関連プロパティ】

LogFlag

6.25. SendData プロパティ

【機能】

機器に送信する文字列データを設定します。

【プロパティ】

SendData

【データ型】

String

【アクセス】

Read/Write

【初期値】

Null文字列

【例】

“MS:F1T0”

【解説】

関連メソッドの対象関数実行時に本プロパティの値を参照し機器に送信します。

【関連メソッド】

pcxSendCommand(),pcxSendDataBCC(),pcxSendBinaryCommand(),
pcxSendBinaryData(),pcxSendCommandRR(),
pcxSendDataRR(), pcxSendBinaryCommandRR()

6.26. ReceiveData プロパティ

【機能】

機器より受信したデータを文字列で示します。

【プロパティ】

ReceiveData

【データ型】

String

【アクセス】

Read

【桁】

可変長

【初期値】

Null文字列

【解説】

関連メソッドの対象関数の受信データの値を示します。

【関連メソッド】

pcxReceiveDataBCC(),pcxReceiveBinaryReport(),
pcxReceiveBinaryData(),
pcxSendDataRR(),pcxSendBinayCommandRR(),

【関連プロパティ】

ReceiveDataLen

6.27. ReceiveDataLen プロパティ

【機能】

機器より受信したデータ部分(ReceiveDataプロパティ)のデータ長を示します。

【プロパティ】

ReceiveDataLen

【データ型】

Long

【アクセス】

Read

【初期値】

0

【解説】

関連メソッドの対象関数のReceiveDataプロパティのデータ長を示します。

【関連メソッド】

pcxReceiveDataBCC(),pcxReceiveBinaryReport(),
pcxReceiveBinaryData(),
pcxSendDataRR(),pcxSendBinayCommandRR(),

【関連プロパティ】

ReceiveData

6.28. PollCmdComSts プロパティ

【機能】

ポーリング(x04x05)通信より受信したデータ部分のコマンドコード(=COM\$)・
またはレディーステータス情報(=STS\$)を示す。

【プロパティ】

PollCmdComSts

【データ型】

String

【アクセス】

Read

【桁】

2 Byte

【初期値】

Null文字列

【解説】

ポーリング受信レスポンスでのSTS\$の値2Byteのみ格納する。

・ [STX][“R”][STS\$][SEN\$][ETX][BCC]

【関連メソッド】

pcxReceiverresponse(),pcxSendCommandRR()

【関連プロパティ】

PollCmdSen, PollCmdErr, PollCmdResData

6.29. PollCmdSen プロパティ

【機能】

ポーリング(x04x05)通信より受信したデータ部分のレディーステータス情報(=SEN\$)を示す。

【プロパティ】

PollCmdSen

【データ型】

String

【アクセス】

Read

【桁】

可変長

【初期値】

Null文字列

【解説】

ポーリング受信レスポンスでの**SEN\$**の値のみ格納する。

- ・ [STX][“R”][STS\$][**SEN\$**][ETX][BCC]
- ・ [STX][“B”][COM\$][**SEN\$**][ETX][BCC]

レスポンスデータ(“R”)受信時 = 2 byte

レスポンスデータ(“B”)受信時 = 2 又は 3 byte

レスポンスデータ(“A”)受信時 = Null文字列

【関連メソッド】

pcxReceiverresponse(),pcxSendCommandRR()

【関連プロパティ】

PollCmdComSts, PollCmdResData

6.30. PollCmdErr プロパティ

【機能】

ポーリング(x04x05)通信より受信したデータ部分のエラーステータス情報(=ERR\$)を示す。

【プロパティ】

PollCmdErr

【データ型】

String

【アクセス】

Read

【桁】

2 Byte

【初期値】

Null文字列

【解説】

ポーリング受信レスポンスでのErr\$の値2byte格納する。

・ [STX][“R”][COM\$][Err\$][(Data\$)][ETX][BCC]

レスポンスデータ(“R”)受信時 = Null文字列

レスポンスデータ(“B”)受信時 = Null文字列

レスポンスデータ(“A”)受信時 = 2 byte

【関連メソッド】

pcxReceiverresponse(),pcxSendCommandRR()

【関連プロパティ】

PollCmdComSts, PollCmdResData

6.31. PollCmdResData プロパティ

【機能】

ポーリング(x04x05)通信より受信したデータ部分(=DATA\$)を示す。

【プロパティ】

PollCmdComSts

【データ型】

String

【アクセス】

Read

【桁】

可変長

【初期値】

Null文字列

【解説】

ポーリング受信レスポンスでの**Data\$**の値を格納する。

・ [STX][“R”][COM\$][Err\$][**Data\$**][ETX][BCC]

レスポンスデータ(“R”)受信時 = Null文字列

レスポンスデータ(“B”)受信時 = Null文字列

レスポンスデータ(“A”)受信時 = 可変長データ

【関連メソッド】

pcxReceiverresponse(),pcxSendCommandRR()

【関連プロパティ】

PollCmdComSts, PollCmdErr

6.32. HFReceiveReserve プロパティ

【機能】

PCC HF帯通信より受信した 1 又は 2 レスポンス目のデータ部分(=Reserved)を示す。

【プロパティ】

HFReceiveReserve

【データ型】

String

【アクセス】

Read

【桁】

1 Byte

【初期値】

Null文字列

【解説】

1 レスポンス目 [STX][**Reserve**][Resp][Stat][ETX][BCC]又は

2 レスポンス目 [STX][**Reserve**][Resp][Stat][DataField][ETX][BCC]の**Reserve**を示す。

レスポンスの取得指定はpcxReceiveDataBCCHF()メソッドの引数の指定による。

【関連メソッド】

pcxSendDataHFRR(),pcxReceiveDataHF()

【関連プロパティ】

HFReceiveResp, HFReceiveStat,
HFReceiveData, HFReceiveDataLen

6.33. HFReceiveResp プロパティ

【機能】

PCC HF帯通信より受信した 1 又は 2 レスポンス目のデータ部分(=Resp)を示す。

【プロパティ】

HFReceiveResp

【データ型】

String

【アクセス】

Read

【桁】

2 Byte

【初期値】

Null文字列

【解説】

1 レスポンス目[STX][Reserve][**Resp**][Stat][ETX][BCC]又は
2 レスポンス目[STX][Reserve][**Resp**][Stat][DataField][ETX][BCC]の**Resp**を示す。
レスポンスの取得指定はpcxReceiveDataBCCHF()メソッドの引数の指定による。

【関連メソッド】

pcxSendDataHFRR(),pcxReceiveDataHF()

【関連プロパティ】

HFReceiveReserve, HFReceiveStat
HFReceiveData, HFReceiveDataLen

6.34. HFReceiveStat プロパティ

【機能】

PCC HF帯通信より受信した 1 又は 2 レスポンス目のデータ部分(=Resp)を示す。

【プロパティ】

HFReceiveStat

【データ型】

String

【アクセス】

Read

【桁】

2 Byte

【初期値】

Null文字列

【解説】

1 レスポンス目[STX][Reserve][Resp][**Stat**][ETX][BCC]又は
2 レスポンス目[STX][Reserve][Resp][**Stat**][DataField][ETX][BCC]の**Stat**を示す。
レスポンスの取得指定はpcxReceiveDataBCCHF()メソッドの引数の指定による。

【関連メソッド】

pcxSendDataHFRR(),pcxReceiveDataHF()

【関連プロパティ】

HFReceiveReserve, HFReceiveResp,
HFReceiveData, HFReceiveDataLen

6.35. HFReceiveData プロパティ

【機能】

PCC HF帯通信より受信したレスポンスデータ部分(=DataField)を示す。

【プロパティ】

HFReceiveData

【データ型】

String

【アクセス】

Read

【桁】

可変長

【初期値】

Null文字列

【解説】

レスポンスデータ部分[STX][Reserve][Resp][Stat][**DataField**][ETX][BCC]
の**DataField**を示す。

【関連メソッド】

pcxSendDataHFRR(),pcxReceiveDataHF()

【関連プロパティ】

HFReceiveReserve, HFReceiveResp, HFReceiveStat,
HFReceiveDataLen

6.36. HFReceiveDataLen プロパティ

【機能】

PCC HF帯通信より受信したレスポンスデータ部分(=DataField)のデータ長を示す。

【プロパティ】

HFReceiveData

【データ型】

Long

【アクセス】

Read

【初期値】

0

【解説】

レスポンスデータ部分[STX][Reserve][Resp][Stat][**DataField**][ETX][BCC]
の**DataField**のデータ長を示す。

【関連メソッド】

pcxSendDataHFRR(),pcxReceiveDataHF()

【関連プロパティ】

HFReceiveReserve, HFReceiveResp, HFReceiveStat,
HFReceiveData

6.37. CancelFlag プロパティ

【機能】

各メソッドの通信中（Busy）に強制中断し、制御をアプリに返す。
必要な場合はキャンセルコマンドを機器に送信する。

【プロパティ】

CancelFlag

【データ型】

Boolean

【アクセス】

Read/Write

【初期値】

False

【値】

True：キャンセルする／False：キャンセルしない

【解説】

- ・ポーリングメソッドにてポーリング再送指定（IRec=1）を行った場合にBusyStatusが継続している状態で処理を中断し新たに処理を行いたい場合には本プロパティを“True”にする。
- ・pcxSendDataBCCRR、pcxReceiveData、pcxSendDataHFRR、pcxReceiveDataHF、pcxSendBinaryCommandRR、pcxReceiveBinaryReportメソッドにて、イベント受信（IResponseMode 0以外）処理を中断し、新たに処理を行いたい場合には本プロパティを“True”にする。
- ・処理が終了したらCancelFlagは“False”になります。
- ・CancelFlagプロパティを“True”を設定した時に、既にBusyStatusPolling再送処理が終了していた場合はCancelFlagを“False”に戻します。

【注記】

- ・CancelFlagにて処理中断後、新たに処理を行う場合はpcxComBufferClearメソッドを利用して、送受信バッファのクリアを行って下さい。

【関連メソッド】

pcxSendCommandRR(), pcxReceiveResponse(), pcxReceiveDataHF(),
pcxSendBinaryCommandRR(), pcxReceiveBinaryReport()
pcxSendDataBCCRR(), pcxReceiveData(), pcxSendDataHFRR()

7. メソッド

7.1. ポートオープンメソッド `pcxOpenPort`

【機能】

シリアルポートをオープンします。

【構文】

LONG `pcxOpenPort();`

【パラメータ】

なし

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	オープン完了
RET_PCX_FAILURE(30)	指定したシリアルポートは既にオープンされている
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	オープン不可（他のプロセスがポートを使用中）

【解説】

シリアルポートをオープンし、接続されている機器を使用可能な状態にします。
これ以降、このポート番号を経由した機器との送受信が可能となります。

本メソッド呼び出しの前に以下のプロパティを設定すること。

PortNo
BaudRate
DataBit
Parity
DSRCheck
CTSCheck
TXBufferSize
RXBufferSize

【関連プロパティ】

PortNo, BaudRate, DataBit, Parity, DSRCheck, CTSCheck, AutoHandshake
TXBufferSize, RXBufferSize

7.2. ポートクローズメソッド `pcxClosePort`

【機能】

シリアルポートをクローズします。

【形式】

`LONG pcxClosePort();`

【パラメータ】

なし

【戻り値】

値	説明
<code>RET_PCX_SUCCESS(0)</code>	クローズ完了
<code>RET_PCX_FAILURE(30)</code>	指定したシリアルポートは既にクローズされている
<code>RET_PCX_ERR_PARAM(100)</code>	パラメータエラー
<code>RET_PCX_ERR_BUSY(140)</code>	非同期処理を実行中

【解説】

シリアルポートをクローズし、再度オープンできる状態にします。

このAPIが成功した後は、`pcxOpenPort()`メソッドを実行しないと使用できません。

7.3. 送受信バッファークリアメソッド `pcxComBufferClear`

【機能】

シリアルポートバッファに格納されているすべてのデータを初期化（破棄）します。

【形式】

```
LONG pcxComBufferClear(  
    LONG ITxRxFlag  
);
```

【パラメータ】

名前	説明
<i>ITxRxFlag</i>	省略可能パラメータ 0：送受信バッファ（省略時適用） 1：送信バッファのみ 2：受信バッファのみ

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_FAILURE(30)	クリアー失敗
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー

【解説】

シリアルポートの送受信バッファのデータを、初期化（破棄）し次回の送受信に備えます。

7.4. レスポンスプロパティクリアメソッド `pcxResPropertyClear`

【機能】

アンサーレスポンス系プロパティの値をクリアする
ReceiveData, ReceiveDataLen,
PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResData,
HFReceiveReserve, HFReceiveResp, HFReceiveStat, HFReceiveData, HFReceiveDataLen

【形式】

`Void pcxResPropertyClear();`

【パラメータ】

なし

【戻り値】

なし

【解説】

全てのアンサーレスポンス系プロパティの値をクリアする。

【関連プロパティ】

ReceiveData, ReceiveDataLen,
PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResData,
HFReceiveReserve, HFReceiveResp, HFReceiveStat, HFReceiveData, HFReceiveDataLen

7.5. ポーリングコマンド送受信メソッド `pcxSendCommandRR`

【機能】

SendDataプロパティから、コマンド電文を生成し、シリアルポートに出力後、レスポンスデータの受信を行いSendData, PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResDataプロパティに格納します。

【形式】

```
LONG pcxSendCommandRR(  
    Long IRecMode,  
    Long IBusyTimeOut,  
);
```

【パラメータ】

名前	説明
<i>IRecMode</i>	ビジステータス受信の時にポーリング再送を行なうかどうかのフラグ (0: しない / 1: する(応答・レイ待ち処理) / 0,1以外: する(イベント処理))
<i>IBusyTimeOut</i>	ビジステータスの終了待ちタイマー値 (単位: msec) (0: 無限待ち[IRecMode=0,1以外]のみ有効)

【戻り値】

値	説明
RET_PCX_EVENT_SUCCESS(1)	イベント待ち登録完了
RET_PCX_NAK(3)	否定 (NAK) 応答
RET_PCX_RESPONSE(10)	レスポンスデータ("A")受信
RET_PCX_BUSY(12)	レスポンスデータ("B")受信
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	機器からの応答待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTSオン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSRオフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BUSY_TIMEOUT(113)	機器からのビジステータス終了待ちタイムアウト
RET_PCX_ERR_BCC_RETRY(120)	BCCエラーリトライオーバー
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時のWIN32APIエラー発生
RET_PCX_ERR_COMM_READ(201)	受信時のWIN32APIエラー発生

【解説】

SendDataプロパティに設定されたコマンドコード、コマンドパラメータから次の形式でコマンド電文を生成します。

電文形式 : [EOT] [STX] [SendDataプロパティ] [ETX] [BCC]

これをシリアルポートに出力し、機器からのACK応答を受信したら、次のポーリングコマンドを送信します。(ACKTimerプロパティ時間分、ACK応答を待ちます)

電文形式 : [EOT] [ENQ]

これにより機器からの**ビジステータス("B")**、**コマンドレスポンス("A")**のいずれかのデータを受信します。

応答がない場合、ResponseTimerプロパティ時間待ち、制御をアプリに返します。

但しIRecModeを0,1以外にした時は、ResponseTimerプロパティは無効になります。

ビジステータス("B")が受信された場合は、IRecModeを参照してポーリングするかどうか判断し、レディステータス("R")かコマンドレスポンス("A")が受信されるまでポーリングを繰り返します。

ビジステータス以外の状態に遷移するか、IBusyTimeOut時間経過した場合、アプリに通知します。

コマンドレスポンス("A")が受信された場合は、それぞれの受信電文形式から、対象とするデータ部分を抽出し、返却用プロパティにセットして、制御をアプリに返します。

返却用プロパティ : **PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResData**

また、機器からの応答電文のBCC(水平パリティ)を計算し、正しくない場合には、機器にNAKを送信する。

この動作を3回繰り返しても正常なBCCが得られないときは、エラーを返します。

IRecMode= (0,1以外 : する) に指定したときはpcxReceiveResponseEventで返却されます。

IRecMode= (1 : する) に指定したときはビジステータスの終了待ちタイムアウト、応答受信、レディレスポンス受信のいずれかにより本メソッドから戻ります。

※電文送信前に受信プロパティの値は、自動でクリア処理(pcxResPropetyClear)されます。

【関連プロパティ】

SendData, PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResData

【関連イベント】

pcxReceiveResponseEvent()

7.6. ポーリングコマンド送信メソッド `pcxSendCommand`

【機能】

SendDataプロパティから、コマンド電文を生成し、シリアルポートに出力しACK/NAKを待ちます。

【形式】

LONG `pcxSendCommand()`;

【パラメータ】

なし

【戻り値】

値	説明
RET_PCX_ACK(2)	肯定 (ACK) 応答
RET_PCX_NAK(3)	否定 (NAK) 応答
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	ACK/NAK待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTSオン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSRオフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時のWIN32APIエラー発生
RET_PCX_ERR_COMM_READ(201)	受信時のWIN32APIエラー発生

【解説】

SendDataプロパティに設定されたコマンドコード、コマンドパラメータから次の形式でコマンド電文を生成します。

電文形式 : [EOT] [STX] [SendDataプロパティ] [ETX] [BCC]

これをシリアルポートに出力し、機器からの応答をACKTimerプロパティ時間まで待ち、制御をAPLに返します。

※電文送信前に受信プロパティの値は、自動でクリアー処理(`pcxResPropetyClear`)されます。

【関連プロパティ】

SendData, PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResData

【関連メソッド】

`pcxReceiveResponse()`

【関連イベント】

`pcxReceiveResponseEvent()`

7.7. ポーリングレスポンス受信メソッド `pcxReceiveResponse`

【機能】

ポーリングを送信し、機器からのレスポンスデータを受け取ります。

【形式】

```
LONG pcxReceiveResponse(  
    LONG IRecMode,  
    LONG IBusyTimeOut,  
);
```

【パラメータ】

名前	説明
<i>IRecMode</i>	ビジステータス受信時にポーリング再送を行うかどうかのフラグ (0 : しない / 1 : する(応答・レディ待ち処理) / 0,1以外 : する(イベント処理))
<i>IBusyTimeOut</i>	ビジステータスの終了待ちタイマー値 (単位 : msec) (0 : 無限待ち [<i>IRecMode</i> = 0,1以外]のみ有効)

【戻り値】

値	説明
RET_PCX_EVENT_SUCCESS(1)	イベント待ち登録完了
RET_PCX_RESPONSE(10)	レスポンスデータ("A")受信
RET_PCX_READY(11)	レスポンスデータ("R")受信
RET_PCX_BUSY(12)	レスポンスデータ("B")受信
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	機器からの応答待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BUSY_TIMEOUT(113)	機器からのビジステータス終了待ちタイムアウト
RET_PCX_ERR_BCC_RETRY(120)	BCC エラーリトライオーバー
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー発生
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー発生

【解説】

シリアルポートに対して、次のポーリングコマンドを送信します。

形式 : [EOT] [ENQ]

これにより機器から以下のいずれかのデータを受信します。

レディステータス :

[STX][R][ステータス情報(2byte)][センサー情報(2byte)][ETX][BCC]

ビジステータス :

[STX][B][コマンドコード(2byte)][センサー情報(2byte又は3byte)][ETX][BCC]

コマンドレスポンス :

[STX][A][コマンドコード(2byte)][エラーコード(2byte)][受信データ(可変長)][ETX][BCC]

それぞれの受信形式から、対象とするデータ部分を抽出し、返却用プロパティにセットして、制御をAPLに戻します。

IRecMode = (0,1以外 : する) に指定したときは `pcxReceiveResponseEvent` で返却されます。

IRecMode = (1 : する) に指定したときはビジステータスの終了待ちタイムアウト、応答受信、レディレスポンス受信のいずれかにより本メソッドから戻ります。

データは下記の返却用プロパティに格納される

`PollCmdComSts` プロパティ = [ステータス情報(2byte)] or [コマンドコード(2Byte)]

`PollCmdSen` プロパティ = [センサー情報(2byte又は3byte)]

`PollCmdErr` プロパティ = [エラーコード(2byte)]

`PollCmdResData` プロパティ = [受信データ(可変長)]

機器からのビジステータス、コマンドレスポンスのいずれかのデータを受信します。

応答がない場合、ResponseTimerプロパティ時間待ち、制御をAPLに返します。

ビジステータスが受信された場合は、*IRecMode*を参照してポーリングするかどうか判断し、レディステータスかコマンドレスポンスが受信されるまでポーリングを繰り返します。ビジステータス以外の状態に遷移するか、*IBusyTimeOut*時間経過した場合、APLに通知します。

他のスレッドから同一ポートを指定して他のメソッドを呼び出した場合、pcxReceiveResponse()メソッドの終了を待ちます。

機器からの応答電文のBCC（水平パリティ）を計算し、正しくない場合には、機器にNAKを送信します。

この動作を3回繰り返しても正常なBCCが得られないときは、RET_PCX_ERR_BCC_RETRYを返します。

【関連プロパティ】

SendData, PollCmdComSts, PollCmdSen, PollCmdErr, PollCmdResData

【関連イベント】

pcxReceiveResponseEvent()

7.8. 電文データ送受信メソッド `pcxSendDataBCCRR`

【機能】

渡されたパラメータからコマンド電文を作成し、シリアルポートに出力後、レスポンスデータの受信を行う。
電文形式 : [STX]["Data"][ETX][BCC]

【形式】

```
LONG pcxSendDataRR(  
    LONG IResponseMode,  
    LONG IRetryCount  
);
```

【パラメータ】

名前	説明
<code>IReceiveMode</code>	データ受信をイベントで取得するかどうかのフラグ (0 以外 : する / 0 : しない)
<code>IRetryCount</code>	データ送信・応答データ受信タイムアウト時のリトライ回数

【戻り値】

値	説明
<code>RET_PCX_SUCCESS(0)</code>	データ受信あり
<code>RET_PCX_EVENT_SUCCESS(1)</code>	イベント待ち登録完了
<code>RET_PCX_NAK(3)</code>	否定 (NAK) 応答
<code>RET_PCX_BUSY(12)</code>	データ受信中／又は受信待ち中／他関数実行中
<code>RET_PCX_FAILURE(30)</code>	データ受信なし
<code>RET_PCX_ERR_PARAM(100)</code>	パラメータエラー
<code>RET_PCX_ERR_PORT(101)</code>	ポート未オープンエラー
<code>RET_PCX_ERR_RES_TIMEOUT(110)</code>	ACK/NAK/レスポンス待ちタイムアウト
<code>RET_PCX_ERR_CTS_TIMEOUT(111)</code>	CTS オン待ちタイムアウト
<code>RET_PCX_ERR_DSR(112)</code>	DSR オフエラー (端末未接続／電源オフ)
<code>RET_PCX_ERR_BCC_RETRY(120)</code>	BCC エラーリトライオーバー
<code>RET_PCX_ERR_COMM_WRITE(200)</code>	送信時の WIN32API エラー発生
<code>RET_PCX_ERR_COMM_READ(201)</code>	受信時の WIN32API エラー発生

【解説】

`SendData` プロパティの値を次の形式のデータ電文を作成します。

形式 : [STX][`SendData` プロパティ][ETX][BCC]

これをシリアルポートに出力し、機器からの ACK 応答を受信します。

`ResponseTimer` プロパティ時間だけ応答がない場合、及び NAK が受信された場合、電文再送リトライを `IRetryCount` 回数行います。

リトライ後もステータスが変わらない場合、最新のステータスを返却します。

ACK を受信したら、機器からの次の形式のデータを受信します。

形式 : [STX][`RecvData` プロパティ][`RecvDataLen` プロパティ][ETX][BCC]

機器からの応答電文の BCC (水平パリティ) を計算し、正しくない場合には、機器に NAK を送信します。

この動作を 3 回繰り返しても正常な BCC が得られない場合、`RET_PCX_ERR_BCC_RETRY(120)`を返します。

正常なデータが受信された場合、データ部分を抽出し、機器に ACK を送信します。
その後、データを返却用パラメータにセットして、制御を APL に返します。

`IReceiveMode` = (0 以外 : する) で指定したときは `pcxReceiveDataEvent` で受信します。

※電文送信前に受信プロパティの値は、自動でクリアー処理(`pcxResPropertyClear`)されます。

【関連プロパティ】

`SendData`, `RecvData`, `RecvDataLen`

【関連イベント】

`pcxReceiveDataBCCEvent()`

7.9. 電文データ送信メソッド `pcxSendData`

【機能】

SendDataプロパティのデータにSTX・ETXを付加し算出したBCCデータを付加し、
シリアルポートへ出力しACK/NAKを待ちます
電文形式 [STX]+["SendData プロパティ"]+[ETX]+[BCC]

【形式】

```
LONG pcxSendData(  
    LONG IACKCheck,  
    LONG IRetryCount,  
    LONG IStxEtxBcc  
);
```

【パラメータ】

名前	説明
<i>IACKCheck</i>	ACK/NAKを待つかどうかのフラグ (0以外: する / 0: しない)
<i>IRetryCount</i>	NAK受信・ACK/NAK受信タイムアウト時のリトライ回数 <i>IACKCheck</i> = (0: しない) 時は無効
<i>IStxEtxBcc</i>	参照したデータに[STX]+[SendDataプロパティ]+[ETX]+[BCC]形式の 電文にするかどうかのフラグ (0以外: する / 0: しない)

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_NAK(3)	否定 (NAK) 応答
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	ACK/NAK待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTSオン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT (112)	DSRオフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時のWIN32APIエラー発生
RET_PCX_ERR_COMM_READ(201)	受信時のWIN32APIエラー発生

【解説】

- ・ `pcxSendData(0, 0, 0)` のとき
SendDataプロパティのデータを加工せずにシリアルポート出力します。
- ・ `pcxSendData(0以外, 0, 0)` のとき
SendDataプロパティのデータを加工せずにシリアルポート出力し、機器からのACK/NAKを待ちます。
- ・ `pcxSendData(0以外, 3, 0)` のとき
SendDataプロパティのデータを加工せずにシリアルポート出力し、機器からのACK/NAKを
ACKTimerプロパティ値まで待ち、ACK/NAKタイムアウト及びNAK受信時は*IRetryCount*分
リトライします。
- ・ `pcxSendData(0以外, 3, 0以外)` のとき
SendDataプロパティのデータに[STX]+[SendDataプロパティ]+[ETX]+[BCC]形式に付加し
機器からのACK/NAKを待ちます。
ACK/NAKタイムアウト及びNAK受信時は*IRetryCount*分リトライします。
- ・ `pcxSendData(0, 0, 0以外)` のとき (例: HF帯送信プロトコル)
SendDataプロパティのデータに[STX]+[SendDataプロパティ]+[ETX]+[BCC]形式に付加し
シリアルポートに出力します。(ACK/NAKは待ちません)

※電文送信前に受信プロパティの値は、自動でクリアー処理(`pcxResPropertyClear`)されます。

【関連プロパティ】

SendData

【関連メソッド】

`pcxSendData()`

7.10. 電文データ受信メソッド `pcxReceiveData`

【機能】

シリアルポート受信バッファに格納されているデータ(STX~ETX+BCC の電文)を読み込みます。
ACK/NAK を返却します。

【形式】

```
LONG pcxReceiveData(  
    LONG IReceiveMode,  
    LONG IACKResponse,  
    LONG IStxEtxBcc,  
);
```

【パラメータ】

名前	説明
<i>IReceiveMode</i>	データ受信をイベントで取得するかどうかのフラグ (0 以外 : する / 0 : しない)
<i>IACKResponse</i>	ACK/NAK を返却するかどうかのフラグ (0 以外 : する / 0 : しない) <i>IStxEtxBcc</i> =0 : しないの時は無効
<i>IStxEtxBcc</i>	[STX]+[データ]+[ETX]+[BCC]電文形式で受け取り、データ部を ReceiveData プロパティに格納するかどうかのフラグ (0 以外 : する / 0 : しない)

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	データ正常受信あり
RET_PCX_EVENT_SUCCESS(1)	イベント待ち登録完了
RET_PCX_BUSY(12)	データ受信／又は受信待ち中／他関数実行中
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	レスポンス待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT (112)	DSR オフエラー (端末未接続／電源オフ)
RET_PCX_ERR_BCC_RETRY(120)	BCC エラー / BCC エラーリトライオーバー
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー発生
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー発生

【解説】

- **pcxReceiveData(0, 0, 0)**のとき
シリアルポートの受信バッファを読み込み ReceiveData,ReceiveDataLen プロパティに格納します。
- **pcxReceiveData(0, 0 以外, 0)**のとき
IStxEtxBcc= (0 : しない) の時は無効です。 **pcxReceiveData(0,0)**の時と同様です。
- **pcxReceiveData(0, 0, 0 以外)**のとき
[STX][“データ”][ETX][BCC]の電文形式で受け取りデータ部分のみを
ReceiveData,ReceiveDataLen プロパティに格納します。
- **pcxReceiveData(0, 0 以外, 0 以外)**のとき
[STX][“データ”][ETX][BCC]の電文形式で受け取りデータ部分のみを
ReceiveData,ReceiveDataLen プロパティに格納します。
機器からの応答電文のBCC（水平パリティ）を計算し、正しくない場合には、機器にNAKを送信します。この動作を3回繰り返しても正常なBCCが得られないときは、エラーを返します。
正常なデータが受信された場合は、データ部分をReceiveData, ReceiveDataLenプロパティに格納し、機器にACKを送信し、制御をAPLに返します。
- **pcxReceiveData(0以外, 0以外, 0以外)**のとき
[STX][“データ”][ETX][BCC]の電文形式をpcxReceiveDataBCCEvent()メソッドで受け取りACK/NAKを返却します。 (*IReceiveMode=0以外の時は、無条件でIStxEtxBcc=0以外になります*)
- **pcxReceiveData(0以外, 0, 0以外)**のとき
[STX][“データ”][ETX][BCC]の電文形式をpcxReceiveDataBCCEvent()メソッドで受け取りACK/NAKを返却しません。 (*IReceiveMode=0以外の時は、無条件でIStxEtxBcc=0以外になります*)
注-1) 通信バッファにデータが複数たまっていた場合は、
最後の電文のみを抽出返却し残りは破棄します。

注-2) **pcxReceiveData(0, 0, 0以外)**の時、RET_PCX_ERR_BCC_RETRY(120)
にて終了するとReceiveData,ReceiveDataLenプロパティにデータを格納します。

注-3) *IReceiveMode=0以外の時は、無条件でIStxEtxBcc=0以外になります*

【関連プロパティ】

ReceiveData,ReceiveDataLen

【関連イベント】

pcxReceiveDataBCCEvent ()

7.11. HF帯RFID電文データ送受信メソッド pcxSendDataHFRR

【機能】

ISO15693 対応非接触 IC タグリーダライターのコマンドを (STX+[SendData プロパティ]+ETX+BCC の電文)を形式でシリアルポートに送信しシリアルポート受信バッファからのデータを読み込み 1 電文目読み込み後エラー判定し、2 電文目を取得します。

【形式】

```
LONG pcxSendDataHFRR(  
                                LONG IResponseMode,  
                                LONG IACKJudge  
);
```

【パラメータ】

名前	説明
IResponseMode	データ受信をイベントで取得するかどうかのフラグ (0以外: する / 0: しない)
IACKJudge	1レスポンス目の電文で2レスポンス目を返却するかどうかを判断させる為のフラグ (0以外: する / 0: しないの場合は1レスポンスのみの返却)

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	データ正常受信あり
RET_PCX_EVENT_SUCCESS(1)	イベント待ち登録完了
RET_PCX_BUSY(12)	データ受信／又は受信待ち中／他関数実行中
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	レスポンス待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー (端末未接続／電源オフ)
RET_PCX_ERR_BCC_RETRY(120)	BCC エラー
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー発生

【解説】

SendDataプロパティより読み込んだISO15693対応非接触ICタグコマンドを、シリアルポートへ送信し受信へ備えます。

シリアルポート受信バッファより読み込んだ

1 電文目受信形式: [STX][Reserve(1byte) + Resp(2byte) + Stat(2byte)][ETX][BCC]

IACKJudge=(0 以外: する)の時は Resp の内容判定後、2 電文目を取得返却 (NG では返却しない)

2 電文目受信形式: [STX][Reserve(1byte) + Resp(2byte) + Stat(2byte)+DATA(可変長)][ETX][BCC]

取得した電文内の値は下記のプロパティに格納されます。

Reserve(1byte) = HFReceiveReserve プロパティ

Resp(2byte) = HFReceiveResp プロパティ

Stat(2byte) = HFReceiveStat プロパティ

DATA(可変長) = HFReceiveData プロパティ, HFReceiveDataLen プロパティ

- ・ **pcxSendDataHFRR(0 以外, 0 以外)**のときは pcxReceiveDataHFEvent()イベントで受信し 1 電文目を自動判定し 2 電文目を取得します。

注-1) 本メソッドはCOM受信バッファをの先頭から順番に取得していきますので、

取得しないかぎり受信バッファにデータは残ったままです。

通信バッファをクリアするにはpcxComRXBufferClearメソッドにてクリアしてください。

注-2) RET_PCX_ERR_BCC_RETRY(120)の場合でも各プロパティにデータを格納します。

※電文送信前に受信プロパティの値は、自動でクリア処理(pcxResPropertyClear)されます。

【関連プロパティ】

SendData

HFReceiveReserve, HFReceiveResp, HFReceiveStat, HFReceiveData, HFReceiveDataLen

【関連メソッド】

pcxComRXBufferClear()

【関連イベント】

pcxReceiveDataHFEvent()

7.12. HF帯RFID電文データ受信メソッド `pcxReceiveDataHF`

【機能】

シリアルポート受信バッファに格納されている、ISO15693 対応非接触 IC タグリーダーライターからのデータ(STX~ETX+BCC の電文)を 1 電文目読み込み後エラー判定し、2 電文目を取得します。

【形式】

```
LONG pcxReceiveDataHF(  
    LONG IResponseMode,  
    LONG IACKJudge  
);
```

【パラメータ】

名前	説明
<code>IResponseMode</code>	データ受信をイベントで取得するかどうかのフラグ (0以外: する / 0: しない)
<code>IACKJudge</code>	1レスポンス目の電文で 2 レスポンス目を返却するかどうかを判断させる 為のフラグ (0以外:する / 0:しないの場合は1レスポンスのみの返却)

【戻り値】

値	説明
<code>RET_PCX_SUCCESS(0)</code>	データ正常受信あり
<code>RET_PCX_EVENT_SUCCESS(1)</code>	イベント待ち登録完了
<code>RET_PCX_BUSY(12)</code>	データ受信／又は受信待ち中／他関数実行中
<code>RET_PCX_ERR_PARAM(100)</code>	パラメータエラー
<code>RET_PCX_ERR_PORT(101)</code>	ポート未オープンエラー
<code>RET_PCX_ERR_RES_TIMEOUT(110)</code>	レスポンス待ちタイムアウト
<code>RET_PCX_ERR_DSR_TIMEOUT(112)</code>	DSR オフエラー (端末未接続／電源オフ)
<code>RET_PCX_ERR_BCC_RETRY(120)</code>	BCC エラー
<code>RET_PCX_ERR_BUSY(140)</code>	非同期処理を実行中
<code>RET_PCX_ERR_COMM_READ(201)</code>	受信時の WIN32API エラー発生

【解説】

1 電文目受信形式: [STX][Reserve(1byte) + Resp(2byte) + Stat(2byte)][ETX][BCC]
`IACKJudge`=(0 以外:する)の時は Resp の内容判定後、2 電文目を取得返却 (NG では返却しない)
2 電文目受信形式: [STX][Reserve(1byte) + Resp(2byte) + Stat(2byte)+DATA(可変長)][ETX][BCC]

取得した電文内の値は下記のプロパティに格納されます。

`Reserve(1byte)` = `HFReceiveReserve` プロパティ
`Resp(2byte)` = `HFReceiveResp` プロパティ
`Stat(2byte)` = `HFReceiveStat` プロパティ
`DATA(可変長)` = `HFReceiveData` プロパティ, `HFReceiveDataLen` プロパティ

- ・ `pcxReceiveDataHF(0 以外, 0 以外)`のときは
`pcxReceiveDataHFEvent()` イベントで受信します。
(`IResponseMode`=0 以外のときは、無条件で `IACKJudge`=0 以外となります。)

注-1) 本メソッドはCOM受信バッファをの先頭から順番に取得していきますので、
取得しないかぎり受信バッファにデータは残ったままです。
通信バッファをクリアするには`pcxComRXBufferClear`メソッドにてクリア
してください。

注-2) `RET_PCX_ERR_BCC_RETRY(120)`の場合でも各プロパティにデータを格納します。

【関連プロパティ】

`HFReceiveReserve`, `HFReceiveResp`, `HFReceiveStat`, `HFReceiveData`, `HFReceiveDataLen`

【関連メソッド】

`pcxSendData(0, 0, 0以外)`, `pcxComRXBufferClear()`

【関連イベント】

`pcxReceiveDataHFEvent()`

7.13. UHF帯RFIDコマンド送受信メソッド `pcxSendBinaryCommandRR`

【機能】

SendDataプロパティのデータをバイナリデータに変換しUHF帯RFID用の電文に加工し、シリアルポートに出力後、シリアルポート受信バッファに格納されているデータを読み込み、制御コードを除くデータのみを文字列データに変換しReceiveDataプロパティに格納します。（文字列データで入出力を行ない、メソッド内部でバイナリデータに変換します）

送信電文形式：[SYN][STX][SendDataプロパティ → バイナリデータ変換][SYN][ETX][LRC]

受信電文形式：[SYN][STX][バイナリデータ → 文字列変換][SYN][ETX][LRC]

【形式】

```
LONG pcxSendBinaryCommandRR(  
                                LONG IResponseMode,  
                                LONG IRetryCount  
);
```

【パラメータ】

名前	説明
<i>IResponseMode</i>	イベント受信するかどうかのフラグ（0以外：する / 0：しない）
<i>IRetryCount</i>	データ送信、NAK受信、及びLRC受信時のリトライ回数

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	データ受信あり
RET_PCX_EVENT_SUCCESS(1)	イベント待ち登録完了
RET_PCX_FAILURE(30)	データ受信なし
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	リーダライタからの応答待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTSオン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSRオフエラー（端末未接続／電源オフ）
RET_PCX_ERR_BCC_RETRY(120)	BCCエラーリトライオーバー
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時のWIN32APIエラー
RET_PCX_ERR_COMM_READ(201)	受信時のWIN32APIエラー

【解説】

SendDataプロパティのデータから次の形式のデータ電文を生成します。

送信電文形式：[SYN][STX][SendDataプロパティ → バイナリデータ変換][SYN][ETX][LRC]

なお、SendDataプロパティに0x16(SYN)が含まれる場合、制御コードのSYNと区別するため、0x16の前にSYNを付加します。

これをシリアルポートに出力し、リーダライタ機からのACK応答を受信したら、リーダからのデータ受信を行います。

ACKTimerプロパティ及びResponseTimerプロパティ時間だけ応答がない場合、及びNAKが受信された場合、電文再送リトライを*iRetryCount*回数行います。リトライ後もステータスが変わらない場合は最新のステータスを返却します。

受信電文形式：[SYN][STX][バイナリデータ → 文字列変換][SYN][ETX][LRC]

指定されたシリアルポートの受信バッファを読み込み、受信電文からデータ部分を抽出（先頭の[SYN][STX]、及び末尾の[SYN][ETX][LRC]を除き、2つ続く0x16(SYN)を0x16にする）し、ReceiveDataプロパティに格納して制御をアプリケーションに戻します。

リーダライタ機からの応答伝聞のLRC（水平パリティ）を計算し、正しくない場合には、リーダライタ機にNAKを送信します。この動作を*iRetryCount*回数だけ繰り返しても正常なLRCが得られないときは、RET_PCX_ERR_BCC_RETRY(120)を返します。

正常なデータが受信された場合は、データ部分を抽出し、リーダライタ機にACKを送信します。その後、データを返却用パラメータにセットして、制御をAPLに戻します。

CrwSendBinaryCommandRRメソッドは、受信バッファを読み込む前にReceiveDataLenの指す変数を0に設定し、受信データのバッファへの格納が完了すると、実際に読み取ったサイズを格納します。

※電文送信前に受信プロパティの値は、自動でクリア処理(`pcxResPropertyClear`)されます。

【関連プロパティ】

SendData ,ReceiveData ,ReceiveDataLen

【関連イベント】

`pcxReceiveBinaryReportEvent()`

7.14. UHF帯RFIDコマンド送信メソッド `pcxSendBinaryCommand`

【機能】

SendDataプロパティのデータをバイナリデータに変換しUHF帯RFID用の電文に加工しシリアルポートに出力、ACK/NAKを受信します。

(文字列データで入力を行ない、メソッド内部でバイナリデータに変換します)

送信電文形式 : [SYN][STX][SendDataプロパティ → バイナリデータ変換][SYN][ETX][LRC]

【形式】

```
LONG pcxSendBinaryCommand (
    LONG IRetryCount,
);
```

【パラメータ】

名前	説明
<i>IRetryCount</i>	データ送信、NAK 受信、及び LRC 受信時のリトライ回数

【戻り値】

値	説明
RET_PCX_ACK(2)	肯定 (ACK) 応答
RET_PCX_NAK(3)	否定 (NAK) 応答
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	リーダライタからの ACK/NAK 待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー

【解説】

SendDataプロパティのデータをバイナリデータに変換し次の形式でコマンド電文を生成します。

電文形式 : [SYN][STX][SendDataプロパティ → バイナリデータ変換][SYN][ETX][LRC]

なお、SendDataプロパティに0x16(SYN)が含まれる場合、制御コードのSYNと区別するため、0x16の前にSYNを付加します。

これをシリアルポートに出力し、ACK/NAK応答をACKTimerプロパティ時間まで待ち、制御をAPLに返します。

ACKTimerプロパティ時間だけ応答がない場合、及びNAKが受信された場合、電文再送リトライを*IRetryCount*回数行います。

リトライ後もステータスが変わらない場合は最新のステータスを返却します。

※電文送信前に受信プロパティの値は、自動でクリアー処理(`pcxResPropertyClear`)されます。

【関連プロパティ】

SendData

【関連メソッド】

`pcxReceiveBinaryReport()`

【関連イベント】

`pcxReceiveBinaryReportEvent()`

7.15. UHF帯RFIDコマンド受信メソッド pcxReceiveBinaryReport

【機能】

シリアルポート受信バッファに格納されているデータを読み込み、UHF帯RFID用の電文を制御コードを除いたバイナリデータを文字列データに変換しReceiveDataプロパティに格納します。
(メソッド内部でバイナリデータから文字列データに変換して出力します)
受信電文形式: [SYN][STX][バイナリデータ → 文字列変換][SYN][ETX][LRC]

【形式】

```
LONG pcxReceiveBinaryReport (
    LONG IResponseMode,
    LONG IRetryCount,
);
```

【パラメータ】

名前	説明
<i>IResponseMode</i>	イベント受信するかどうかのフラグ (0 以外: する / 0: しない)
<i>iRetryCount</i>	LRC 受信時のリトライ回数

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_EVENT_SUCCESS(1)	イベント待ち登録完了
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	リーダライタからの応答待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BCC_RETRY(120)	LRC エラーリトライオーバー
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー

【解説】

受信電文形式: [SYN][STX][バイナリデータ → 文字列変換(*lpReceiveData*)][SYN][ETX][LRC]

指定されたシリアルポートの受信バッファを読み込み、受信電文からデータ部分を抽出
(先頭の[SYN][STX]、及び末尾の[SYN][ETX][LRC]を除き、2つ続く0x16(SYN)を0x16にする)
し、文字列データに変換後、ReceiveData・ReceiveDataLenプロパティに格納して
制御をアプリケーションに戻します。

リーダライタ機からの応答伝間のLRC (水平パリティ) を計算し、正しくない場合には、
リーダライタ機にNAKを送信します。この動作を*IRetryCount*回繰り返しても正常なLRCが
得られないときは、RET_PCX_ERR_BCC_RETRY(120)エラーを返します。

正常なデータが受信された場合は、データ部分を抽出し、リーダライタ機にACKを送信します。
その後、データを文字列データに変換し返却用プロパティにセットして、制御をAPLに戻します。

【関連プロパティ】

ReceiveData,ReceiveDataLen

【関連イベント】

pcxReceiveBinaryReportEvent()

7.16. バイナリデータ送信メソッド `pcxSendBinaryData`

【機能】

SendDataプロパティのデータをバイナリデータに変換し一切加工せずシリアルポートへ出力します。
(ACK/NAK応答は待たない。)

【形式】

LONG `pcxSendBinaryData ();`

【パラメータ】

なし

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー

【解説】

SendDataプロパティのデータをシリアルポートに出力し、制御をAPLに返します。

※電文送信前に受信プロパティの値は、自動でクリアー処理(`pcxResPropetyClear`)されます。

【関連プロパティ】

SendData

【関連メソッド】

`pcxReceiveBinaryData ()`

7.17. バイナリデータ受信メソッド `pcxReceiveBinaryData`

【機能】

シリアルポート受信バッファに格納されているバイナリデータを読み込み文字列データに変換後
ReceiveData・ReceiveDataプロパティに格納します。

【形式】

LONG `pcxReceiveBinaryData();`

【パラメータ】

なし

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_ERR_PARAM(100)	パラメータエラー
RET_PCX_ERR_PORT(101)	ポート未オープンエラー
RET_PCX_ERR_RES_TIMEOUT(110)	データの受信待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー（端末未接続／電源オフ）
RET_PCX_ERR_BUSY(140)	非同期処理を実行中
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー

【解説】

指定されたシリアルポートの受信バッファを読み込み、受信データを抽出して、文字列データに変換しReceiveDataプロパティに格納して、制御をアプリケーションに戻します。
`pcxReceiveBinaryData`は、最初にReceiveDataLenプロパティの指す変数を0に設定し、
受信データのReceiveDataプロパティへの格納が完了すると、実際に読み取ったサイズが設定されます。
ResponseTimerプロパティで指定した時間待つレスポンスを返却することができます。

【関連プロパティ】

ReceiveData, ReceiveDataLen

【関連メソッド】

`pcxSendBinaryData ()`

7.18. プリントジョブクリアメソッド `pcxPrintJobClear`

【機能】

指定プリンターのジョブを、すべて削除します。

【形式】

```
LONG pcxPrintJobClear ( );
```

【パラメータ】

なし

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_FAILURE(30)	WIN32API エラー
RET_PCX_ERR_PARAM(100)	PrinterDeviceName プロパティが未設定

【解説】

PrinterDeviceNameプロパティで指定されたプリンターのジョブを全て削除します。

プリンターデバイスインターフェイスを有する機器対象

弊社のUSBマルチインターフェイス等で、機器がトラブル等によりプリンターのジョブが、溜まってしまった時にご使用ください

【関連プロパティ】

PrinterDeviceName

7.19. SUM計算メソッド `pcxDataSumCalc`

【機能】

渡された文字列を計算し値(SUM値)を返します (2バイト)

【形式】

```
String pcxDataSumCalc(  
    String csQSumStr  
);
```

【パラメータ】

名前	説明
csQSumStr	計算するデータ

【戻り値】

計算したSUM値 (2バイト)

【解説】

SUM値の計算を行い、値を2バイトで返却します。

7.20. ASCIIコード変換メソッド pcxCharAscHConv

【機能】

文字列をASCIIコードの文字列に変換します。

【形式】

```
String pcxCharAscHConv(
    Long IConvMode,
    String csQConvStr,
);
```

【パラメータ】

名前	説明
IConvMode	0:Char ⇒ ASCII ("0" ⇒ "30") 0以外 : ASCII ⇒ Char ("30" ⇒ "0")
csQConvStr	変換するデータ

【戻り値】

変換したデータ

【解説】

IConvMode = 0 以外で変換できない文字は、"."で返却されます。

※ 文字コード

- ・漢字コード : J I S C 6 2 2 6 (1 9 8 3) 準拠字種
: J I S C 6 2 3 4 (1 9 8 3) 準拠字体
- ・ANKコード : ANKコード表参照

ANKコード表

上位 下位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			SP	0	@	P	`	p				一	タ	ミ		
1			!	1	A	Q	a	q				。	ア	チ	ム	
2	STX		"	2	B	R	b	r				「	イ	ツ	メ	
3	ETX		#	3	C	S	c	s				」	ウ	テ	モ	
4	EOT		\$	4	D	T	d	t				、	エ	ト	ヤ	
5	ENQ	NAK	%	5	E	U	e	u				・	オ	ナ	ユ	
6	ACK		&	6	F	V	f	v				ヲ	カ	ニ	ヨ	
7			'	7	G	W	g	w				ア	キ	ヌ	ラ	
8			(8	H	X	h	x				イ	ク	ネ	リ	
9)	9	I	Y	i	y				ウ	ケ	ノ	ル	
A	LF		*	:	J	Z	j	z				エ	コ	ハ	レ	
B		ESC	+	;	K	[k	[オ	サ	ヒ	ロ	
C			,	<	L	¥	l					ヤ	シ	フ	ワ	
D			—	=	M]	m	}				ユ	ス	ヘ	ン	
E	SO		.	>	N	^	n	~				ヨ	セ	ホ	°	
F	SI		/	?	O	_	o					ツ	ソ	マ		

7.21. JIS／Shift-JIS漢字コード変換メソッド pcxJISconvSJIS

【機能】

漢字コードJIS⇔Shift-JISの変換を行います。

【形式】

```
String pcxJISconvSJIS(  
    Long      IJConvMode,  
    String    csQJConvStr,  
);
```

【パラメータ】

名前	説明
<i>IJConvMode</i>	0:JIS ⇒ Shift-JIS (“4267” ⇒ “91E5”) = “大” 0以外 : Shift-JIS ⇒ JIS (“91E5” ⇒ “4267”) = “大”
<i>csQJConvStr</i>	変換するデータ

【戻り値】

変換したデータ

注意：判別不可能 or 変換不可能 or 規定バイト外(4の倍数)なコードはNullで返却

【解説】

例：“松下太郎”(Shift-JIS ⇒ JIS)

Work = pcxJISconvSJIS(1, “8FBC89BA91BE9859”)

Work = “3E3E323C42404F3A”

8. イベント

8.1. ポーリングレスポンス受信イベント `pcxReceiveResponseEvent`

【機能】

レスポンス受信メソッドで、ポーリング再送を行なった場合に、受信したレスポンスを通知する。

【形式】

```
void    pcxReceiveResponseEvent (
                                LONG IResponseCode
                                );
```

【パラメータ】

名前	説明
<code>IResponseCode</code>	受信したレスポンス

`IResponse`の値は以下の通り。

値	説明
<code>RET_PCX_RESPONSE(10)</code>	レスポンスデータ("A")受信
<code>RET_PCX_CANCELED(20)</code>	<code>CancelFlag</code> により処理中止
<code>RET_PCX_ERR_RES_TIMEOUT(110)</code>	機器からの応答待ちタイムアウト
<code>RET_PCX_ERR_CTS_TIMEOUT(111)</code>	CTSオン待ちタイムアウト
<code>RET_PCX_ERR_DSR_TIMEOUT(112)</code>	DSRオフエラー (端末未接続/電源オフ)
<code>RET_PCX_ERR_BUSY_TIMEOUT(113)</code>	機器からのビジーステータス終了待ちタイムアウト
<code>RET_PCX_ERR_BCC_RETRY(120)</code>	BCCエラーリトライオーバー
<code>RET_PCX_ERR_BUSY(140)</code>	非同期処理を実行中
<code>RET_PCX_ERR_COMM_WRITE(200)</code>	送信時のWIN32APIエラー発生
<code>RET_PCX_ERR_COMM_READ(201)</code>	受信時のWIN32APIエラー発生

【戻り値】

なし

【解説】

レスポンス受信メソッドで `IRecMode` を指定した場合にポーリング再送を行ない、**レディステータス**か**コマンドレスポンス**を受信した場合、**ビジーステータス**が継続した場合、またはエラー発生時に通知する。

シリアルポートに対して、次のポーリングコマンドを送信します。

形式: [EOT] [ENQ]

これにより機器から以下のいずれかのデータを受信します。

レディステータス:

[STX][‘R’][ステータス情報(2byte)][センサー情報(2byte)][ETX][BCC]

ビジーステータス:

[STX][‘B’][コマンドコード(2byte)][センサー情報(2byte又は3byte)][ETX][BCC]

コマンドレスポンス:

[STX][‘A’][コマンドコード(2byte)][エラーコード(2byte)][受信データ(可変長)][ETX][BCC]

それぞれの受信形式から、対象とするデータ部分を抽出し、返却用プロパティに格納します

※`CancelFlag` プロパティが、“True”に設定された場合は処理を中断しアプリに通知します。

データは下記の返却用プロパティに格納される

`PollCmdComSts` プロパティ = [ステータス情報(2byte)] or [コマンドコード(2Byte)]

`PollCmdSen` プロパティ = [センサー情報(2byte又は3byte)]

`PollCmdErr` プロパティ = [エラーコード(2byte)]

`PollCmdResData` プロパティ = [受信データ(可変長)]

【関連プロパティ】

`SendData` , `PollCmdComSts` , `PollCmdSen` , `PollCmdErr` , `PollCmdResData`

【関連メソッド】

`pcxSendCommandRR()` , `pcxSendCommand()` , `pcxReceiveResponse()`

8.2. 電文データ受信イベント pcxReceiveDataBCCEvent

【機能】

電文データ送受信又は受信メソッドでイベント受信指定時に(STX~ETX+BCCの電文)を通知します。
ACK/NAKを返却します。

【形式】

```
LONG pcxReceiveDataBCCEvent(  
                                LONG IResponseCode  
                                );
```

【パラメータ】

名前	説明
IResponseCode	受信したレスポンスコード

IResponseCodeの値は以下の通り。

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	データ正常受信あり
RET_PCX_CANCELED(20)	CancelFlagにより処理中止
RET_PCX_ERR_RES_TIMEOUT(110)	レスポンス待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー（端末未接続／電源オフ）
RET_PCX_ERR_BCC_RETRY(120)	BCC エラー／BCC エラーリトライオーバー
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー発生
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー発生

【解説】

[STX][“データ”][ETX][BCC]の電文形式で受け取りデータ部分のみを
ReceiveData,ReceiveDataLen プロパティに格納します。

pcxReceiveData()でIACKResponse=(0:しない)にすると、ACK/NAK返却はしません。

注-1) 通信バッファにデータが複数たまっていた場合は、
最後の電文のみを抽出返却し残りは破棄します。

※CancelFlagプロパティが、“True”に設定された場合は処理を中断しアプリに通知します。

【関連プロパティ】

ReceiveData,ReceiveDataLen

【関連メソッド】

pcxSendDataBCCRR(),pcxReceiveData()

8.3. HF帯RFID電文データ受信イベント pcxReceiveDataHFEvent

【機能】

シリアルポート受信バッファに格納されている、ISO15693 対応非接触 IC タグリーダーライターからのデータ(STX~ETX+BCC の電文)を 1 電文目読み込み後エラー判定し、2 電文目を取得します。

【形式】

```
LONG pcxReceiveDataHF(  
                                LONG IReceiveCode  
);
```

【パラメータ】

名前	説明
IReceiveCode	受信したレスポンスコード

IReceiveCodeの値は以下の通り。

【戻り値】

値	説明
RET_PCX_SUCCESS(0)	データ正常受信あり
RET_PCX_CANCELED(20)	CancelFlagにより処理中止
RET_PCX_ERR_RES_TIMEOUT(110)	レスポンス待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー（端末未接続／電源オフ）
RET_PCX_ERR_BCC_RETRY(120)	BCC エラー
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー発生

【解説】

pcxReceiveDataHF()又はpcxSendDataHFRR()にて**IACKJudge**のフラグにより下記の電文受信判定を行い専用プロパティに格納します。

1 電文目受信形式 : [STX][Reserve(1byte) + Resp(2byte) + Stat(2byte)][ETX][BCC]

IACKJudge=0 以外:する)の時は Resp の内容判定後、2 電文目を取得返却 (NG では返却しない)

2 電文目受信形式 : [STX][Reserve(1byte) + Resp(2byte) + Stat(2byte)+DATA(可変長)][ETX][BCC]

取得した電文内の値は下記のプロパティに格納されます。

Reserve(1byte) = HFReceiveReserve プロパティ

Resp(2byte) = HFReceiveResp プロパティ

Stat(2byte) = HFReceiveStat プロパティ

DATA(可変長) = HFReceiveData プロパティ, HFReceiveDataLen プロパティ

注-1) 本メソッドはCOM受信バッファをの先頭から順番に取得していきますので、

取得しないかぎり受信バッファにデータは残ったままです。

通信バッファをクリアするにはpcxComRXBufferClearメソッドにてクリアしてください。

注-2) RET_PCX_ERR_BCC_RETRY(120)の場合でも各プロパティにデータを格納します。

※CancelFlag プロパティが、"True"に設定された場合は処理を中断しアプリに通知します。

【関連プロパティ】

HFReceiveReserve, HFReceiveResp, HFReceiveStat, HFReceiveData, HFReceiveDataLen

【関連メソッド】

pcxSendDataHFRR (),pcxReceiveDataHF(),pcxComRXBufferClear()

8.4. UHF帯RFIDコマンド受信イベント pcxReceiveBinaryReportEvent

【機能】

UHF帯RFIDバイナリの受信したレポートを通知する。
シリアルポート受信バッファに格納されているデータを読み込み、UHF帯RFID用の電文を制御コードを除いたバイナリデータを文字列データに変換しReceiveDataプロパティに格納します。
(メソッド内部でバイナリデータから文字列データに変換して出力します)
受信電文形式: [SYN][STX][バイナリデータ → 文字列変換][SYN][ETX][LRC]

【形式】

```
void pcxReceiveBinaryReportEvent (
                                LONG IBinaryReport
);
```

【パラメータ】

名前	説明
IBinaryReport	受信したレスポンス

IBinaryReportの値は以下の通り。

値	説明
RET_PCX_SUCCESS(0)	正常終了
RET_PCX_CANCELED(20)	CancelFlagにより処理中止
RET_PCX_ERR_RES_TIMEOUT(110)	デバイスからの応答待ちタイムアウト
RET_PCX_ERR_CTS_TIMEOUT(111)	CTS オン待ちタイムアウト
RET_PCX_ERR_DSR_TIMEOUT(112)	DSR オフエラー (端末未接続/電源オフ)
RET_PCX_ERR_BCC_RETRY(120)	LRC エラーリトライオーバー
RET_PCX_ERR_COMM_WRITE(200)	送信時の WIN32API エラー
RET_PCX_ERR_COMM_READ(201)	受信時の WIN32API エラー

【解説】

受信電文形式: [SYN][STX][バイナリデータ → 文字列変換(lpReceiveData)][SYN][ETX][LRC]

指定されたシリアルポートの受信バッファを読み込み、受信電文からデータ部分を抽出
(先頭の[SYN][STX]、及び末尾の[SYN][ETX][LRC]を除き、2つ続く0x16(SYN)を0x16にする)
し、文字列データに変換後、ReceiveData・ReceiveDataLenプロパティに格納して
制御をアプリケーションに戻します。
リーダライタ機からの応答伝聞のLRC (水平パリティ) を計算し、正しくない場合には、
リーダライタ機にNAKを送信します。この動作をpcxReceiveBinaryReport()メソッドの
IRetryCount回繰り返しても正常なLRCが得られないときは、
RET_PCX_ERR_BCC_RETRY(120)エラーを返します。
正常なデータが受信された場合は、データ部分を抽出し、リーダライタ機にACKを送信します。
その後、データを文字列データに変換し返却用プロパティにセットして、制御をAPLに戻します。

※CancelFlagプロパティが、"True"に設定された場合は処理を中断しアプリに通知します。

【関連プロパティ】

SendData,ReceiveData,ReceiveDataLen

【関連メソッド】

pcxSendBinaryCommandRR(), pcxSendBinaryCommand(),
pcxReceiveBinaryReport()

8.5. データ通信イベント pcxCommEvent

【機能】

発生した通信イベントを取得する。

【形式】

```
void pcxCommEvent(  
    LONG IEvent  
);
```

【パラメータ】

名前	説明
IEvent	通信状態

IEventの値は以下の通り。

値	説明
EVT_PCX_RXCHAR	入力バッファにデータを受信した。
EVT_PCX_TXEMPTY	出力バッファの最後の文字を送信した。
EVT_PCX_ERR_BUFFER_OVERFLOW	入力バッファサイズを越えて受信した。
EVT_PCX_CTS	CTS (送信可) 信号の状態が変わった。
EVT_PCX_DSR	DSR (データセットレディ) 信号の状態が変わった。
EVT_PCX_RLSD	RLSD (受信線信号検出)信号の状態が 変わった。
EVT_PCX_BREAK	Break信号を検出した。
EVT_PCX_RING	呼び出し信号を検出した。
EVT_PCX_ERR_FRAME	フレーミングエラーが発生した。
EVT_PCX_ERR_OVERRUN	オーバーランエラーが発生した。
EVT_PCX_ERR_PARITY	パリティエラーが発生した。

実際の値は以下のように定義する。

定数名	値(16進数)
EVT_PCX_RXCHAR	0x00000001
EVT_PCX_TXEMPTY	0x00000002
EVT_PCX_ERR_BUFFER_OVERFLOW	0x00000004
EVT_PCX_CTS	0x00000008
EVT_PCX_DSR	0x00000010
EVT_PCX_RLSD	0x00000020
EVT_PCX_BREAK	0x00000040
EVT_PCX_RING	0x00000100
EVT_PCX_ERR_FRAME	0x00000200
EVT_PCX_ERR_OVERRUN	0x00000400
EVT_PCX_ERR_PARITY	0x00000800
EVT_PCX_ALL	0xFFFFFFFF

【戻り値】

なし

【解説】

シリアルポートの通信状態を監視し、何らかの状態の変化があれば、そのデータを返す。
EventMask プロパティによって指定されたイベント以外は通知されない。

9. ログの構成

出力されるログは以下ようになります。

```
2006:12:19:11:08:37: 90: 3232: 3292:Inf:Info      : OCX Version = 1, 0, 0, 3
2006:12:19:11:08:37:100: 3232: 3292:Inf:Enter     : pcxOpenPort
2006:12:19:11:08:37:100: 3232: 3292:Inf:CSLock    : pcxOpenPort
2006:12:19:11:08:37:141: 3232: 3292:Inf:Info      : Return Value is 0
2006:12:19:11:08:37:141: 3232: 3292:Inf:CSUnlock: pcxOpenPort
2006:12:19:11:08:37:141: 3232: 3292:Inf:Exit      : pcxOpenPort
2006:12:19:11:08:37:151: 3232: 3292:Inf:Enter     : pcxReceiveResponse
2006:12:19:11:08:37:161: 3232: 3292:Inf:CSLock    : pcxReceiveResponse
  Addr : +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
-----
0000 : 04 05                                     1|
  Addr : +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
-----
0000 : 02 52 30 30 31 30 03 50                   1R00101P

2007:08:22:14:13:36:421: 3520: 2452:Inf:Info      : Event Notification: 0x00000004(0, 0)
```

2006:12:19:11:08:37:100: 3232: 3292:Inf:Enter : pcxOpenPort
(1) (2) (3) (4) (5) (6) (7)

- (1) 日付 (yyyy:mm:dd)
ログに書き込まれた日付が記録されています。
- (2) 時刻 (hh:mm:ss:SSS)
ログに書き込まれた時刻が記録されています。
- (3) プロセスID
ログに書き込んだプロセスのIDが記録されています。
- (4) スレッドID
ログに書き込んだスレッドのIDが記録されています。
- (5) ログの種類1 (Inf: 情報 / Err: エラー)
書き込まれたログの種類が記録されています。
Inf 通常情報 (メソッド進入、メソッド脱出、パラメータ出力など)
Err エラー情報 (エラーコードの出力)
- (6) ログの種類2
書き込まれたログの詳細が記録されています。
Enter メソッド進入時
Exit メソッド脱出時
CSLock クリティカルセクションのロックを取得時
CSUnlock クリティカルセクションのロックを解放時
Param パラメータ出力
Info 各種情報
ErrInfo エラー情報
- (7) その他
進入 (脱出) したときのメソッド名、パラメータの値、エラーコードとエラー内容などがここに記述されます。

(7-1) イベントログ詳細

OCX内にて発生時のイベントのログについて説明します。

Event Notification: 0x00000004 (0,0)

① ② ③

① イベントコード詳細

イベントコード		説明
0x00000001	EV_RXCHAR	Any Character Received
0x00000002	RXFLAG	Received Certain Character
0x00000004	TXEMPTY	Transmitt Queue Empty
0x00000008	EV_CTS	CTS Changed State
0x00000010	EV_DSR	DSR Changed State
0x00000020	EV_RLSD	RLSD Changed State
0x00000040	EV_BREAK	BREAK Received
0x00000080	EV_ERR	Line Status Error Occurred
0x00000100	EV_RING	Ring Signal Detected
0x00000200	EV_PERR	Printer Error Occured
0x00000400	EV_RX80FULL	Receive Buffer Is 80 Percent full
0x00008000	EV_RXOVERFLOW	Receive Buffer Is Over Flow
0x00010000	EV_EVENT1	Provider specific event 1
0x00020000	EV_EVENT2	Provider specific event 2

② イベント発生前の受信バッファ中(Windows管理)残データ数

③ イベント発生後の受信バッファ中(Windows管理)残データ数

※注記

ログが書き込まれるタイミングとして、メソッドを呼び出した段階でEnterログが書き込まれます。そのため、マルチスレッド環境において、Enterログが続けて出力されることがあります。
また、不正なポート番号（1～256以外の番号）が指定された場合、ログファイルは出力されません。

10. OCX使用上の注意事項

本 OCX を使用するにあたり、特に注意が必要な点については以下のとおりです。

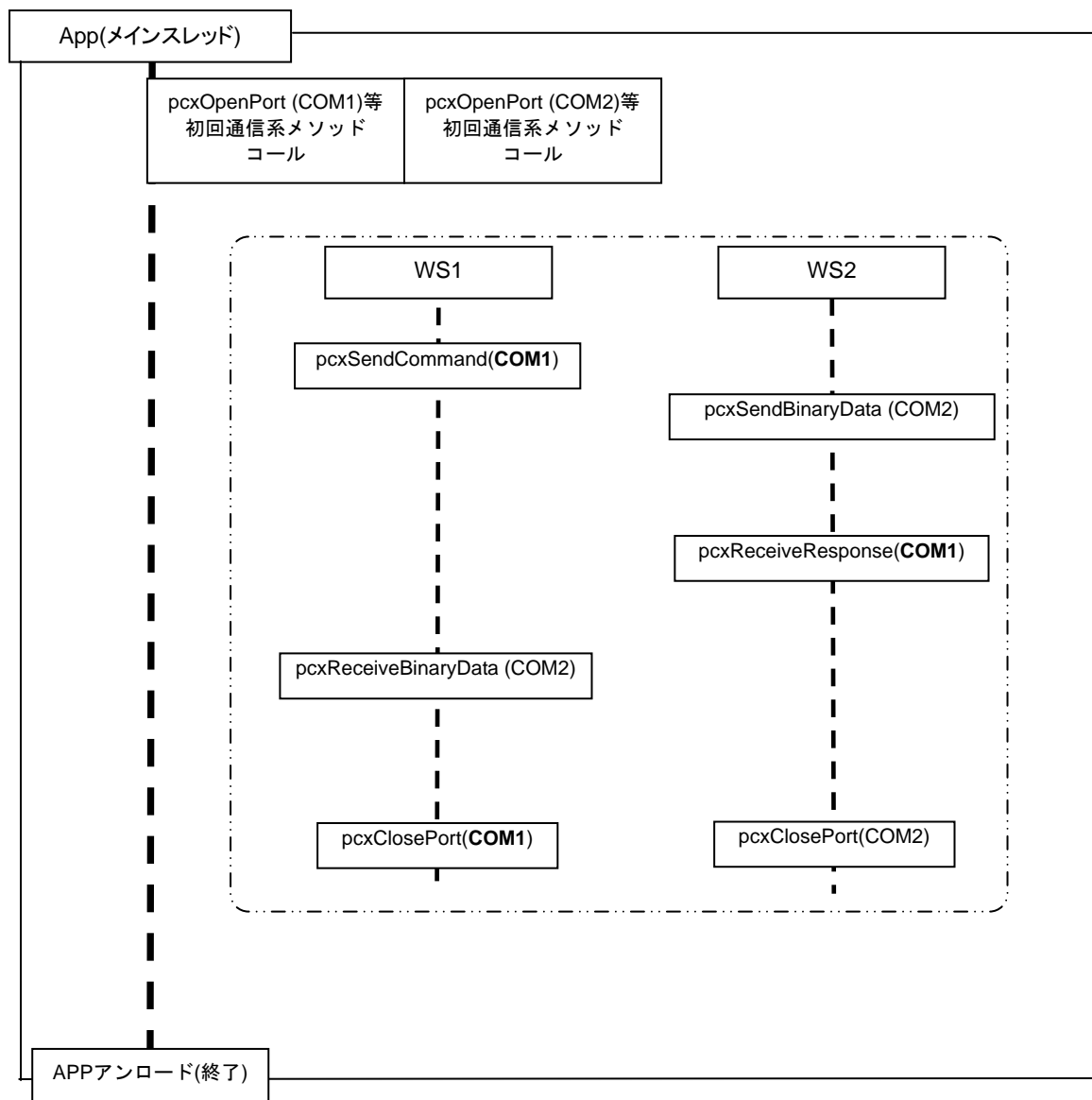
1. 同一のプロセスにおいて、複数のメソッドを非同期に(並行に)呼び出すことが可能です。同一ポートへのそれぞれの処理(ポートのオープン、データの送受信、ポートのクローズ)を異なるスレッドで行うことも可能です。ただし同一ポートへの同時メソッド呼び出しは、そのポートにおける実行中のメソッドがある場合はその処理が終わるまで待たされます。
2. 弊社、仮想COMドライバー(PCC Virtual COM)と併用する場合には、下記の点を踏まえ設計願います。
 - ・ ポートオープン中はUSBケーブルの抜き差しは基本的に行わないでください。
(通信が必要ないときは、pcxClosePort()メソッドにてクローズ処理を行うことをお勧めします)
 - ・ ポートオープン中USBケーブルが突然切断された状態になった時に、“PCC Virtual COM”は Break信号検出”の信号を返してきますので、データ通信イベント(pcxCommEvent)にて“EVT_PCX_BREAK”を取得許可するよう実装し、このときにポートクローズメソッド(pcxClosePort)にて、即座に対象ポートをクローズするように実装してください。
(ポートがオープンのままでは、ドライバーがアンロード又はリロードできない場合があります)

処理手順：

USBケーブル切断 ⇒ ブレーク信号発生（イベント取得） ⇒ ポートクローズ処理

11. 補足資料

11.1. マルチスレッドの基本使用方法



注意：各ポートに対して初回のみ PCCAx.ocx メソッド及び
プロパティ呼び出し時に MainThread にて、通信系メソッド(通常は pcxOpenPort)又はプ
ロパティをコールするように設計してください。

※対象の通信系メソッドを下記に記します。

【プロパティ】

- ・ SenDelayTime / WriteFileTimeOut / DTRLLineControl / RTSLineControl
- ・ BreakControl / GetCommStatus / Claimed / EventMask
- ・ LogFlag / LogFileFolder / CancelFlag

【メソッド】

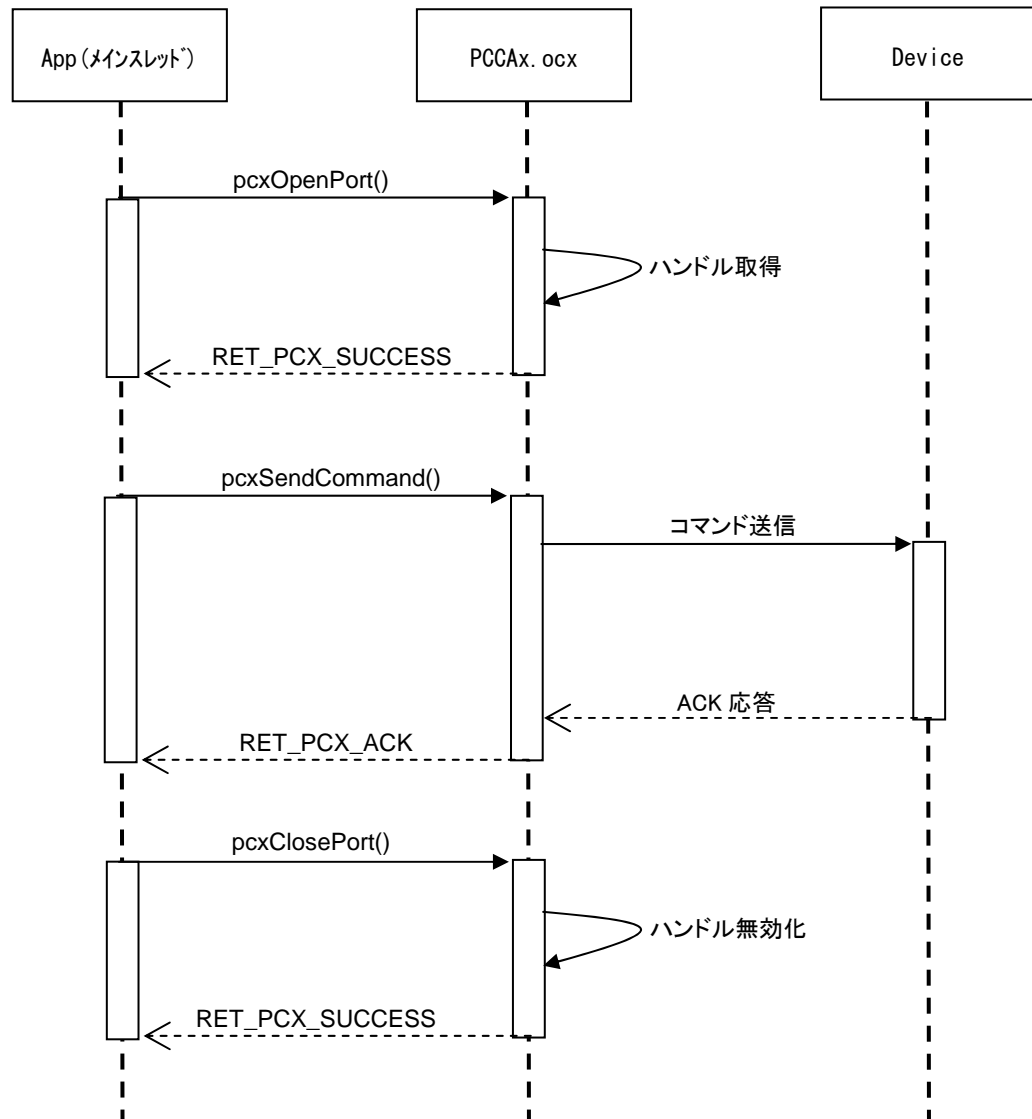
- ・ pcxOpenPort / pcxClosePort / pcxComBufferClear
- ・ pcxSendCommandRR / pcxSendCommand / pcxReceiveResponse
- ・ pcxSendDataBCCRR / pcxSendData / pcxReceiveData
- ・ pcxSendDataHFRR / pcxReceiveDataHF / pcxSendBinaryCommandRR
- ・ pcxSendBinaryCommand / pcxReceiveBinaryReport
- ・ pcxSendBinaryData / pcxReceiveBinaryData
- ・ pcxReceiveResponseEvent / pcxReceiveDataBCCEvent
- ・ pcxReceiveDataHFEvent / pcxReceiveBinaryReportEvent / pcxCommEvent

11.2. マルチスレッドの動作説明

マルチスレッドの前提として、以下の二つがあります。

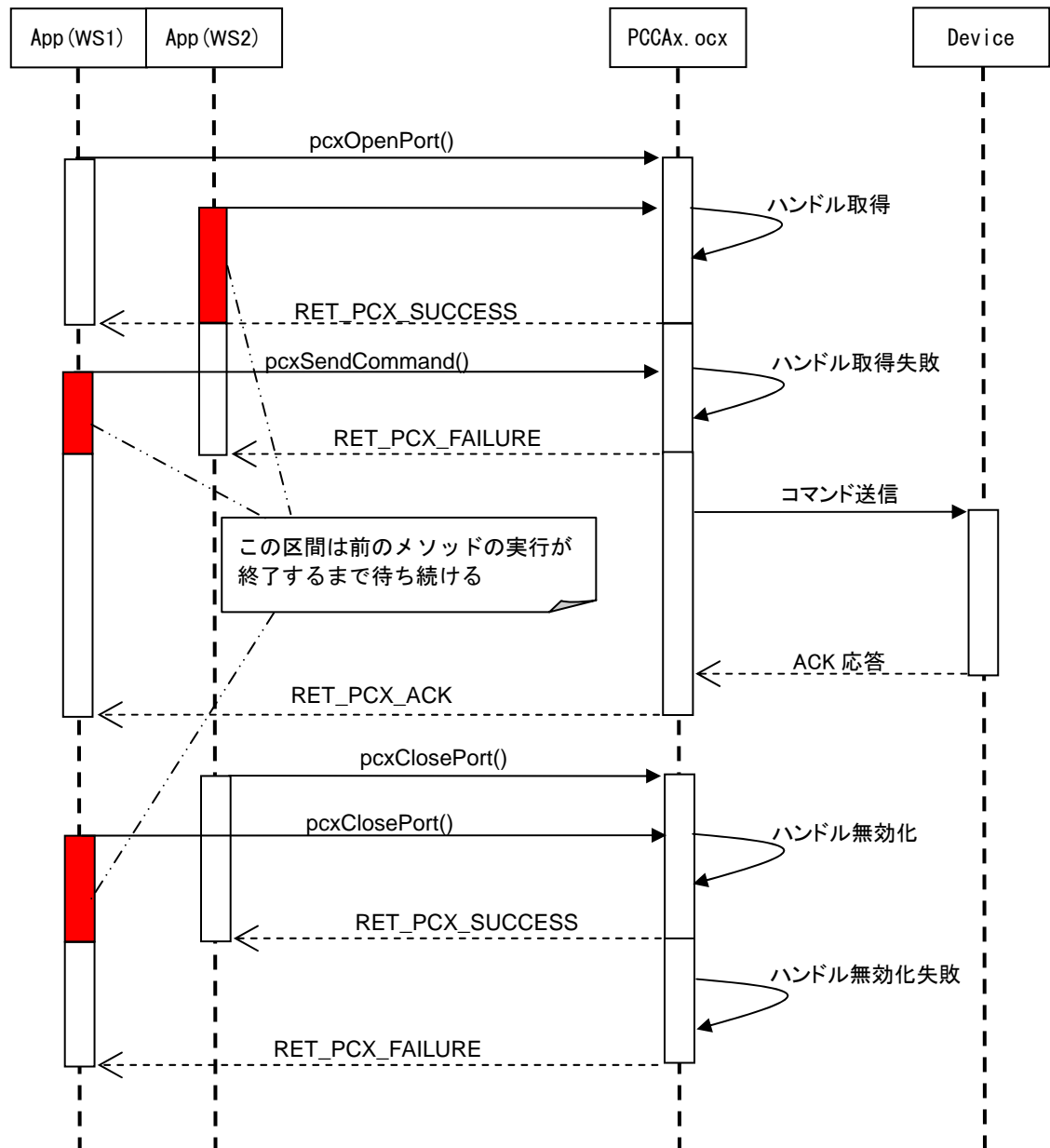
- (1) 同ポートに関する関数呼び出しは、一つの関数処理が終了するまで、別の関数の実行を待機します。
 - (2) 別ポートに関する関数呼び出しは、並列に動作します。
- 以下の項目では、それぞれの事例について説明します。

11.2.1. シングルスレッド（1ポート使用）の場合



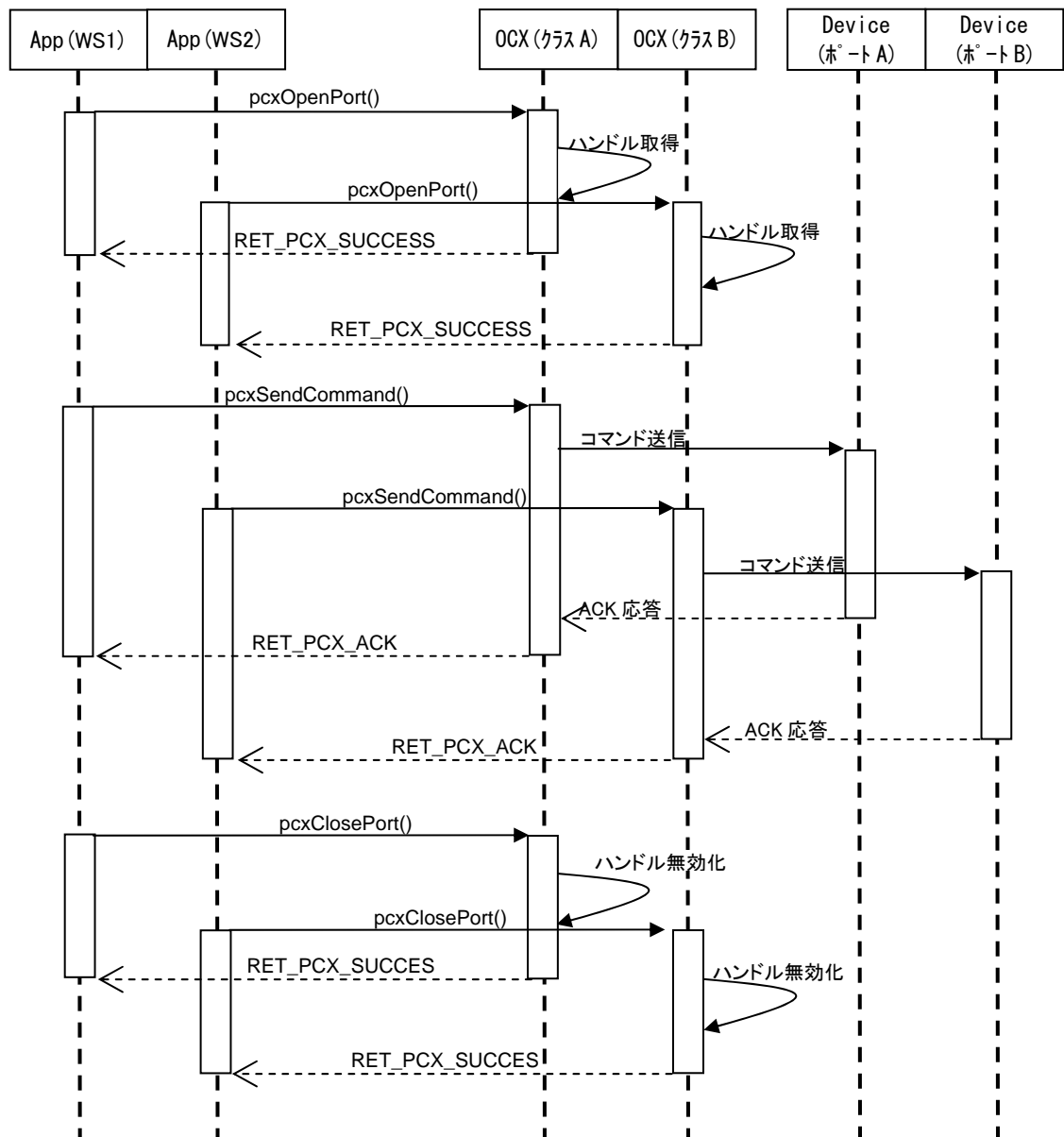
各メソッドは同期的に実行されます

11.2.2. マルチスレッド（同一ポート使用）の場合



複数のワーカースレッド（WS）からメソッドの呼び出しが行われますが、同じポートに対して、メソッドは同時実行することができないため、呼び出しから実行まで待ち時間が発生する場合があります。図の例では、WS1から`pcxOpenPort`メソッドが最初に呼ばれて実行されます。このとき、WS2から同一ポートに対して、`pcxOpenPort`メソッドを実行した場合、先に実行していたメソッドの終了を待ってからメソッドを実行するため、待ち時間が発生します。

11.2.3. マルチスレッド（別ポート使用）の場合



複数のワーカースレッド（WS）からメソッドの呼び出しが行われますが、それぞれが別のポートにアクセスしている場合は、それぞれのメソッドを並列に実行することができます。¹

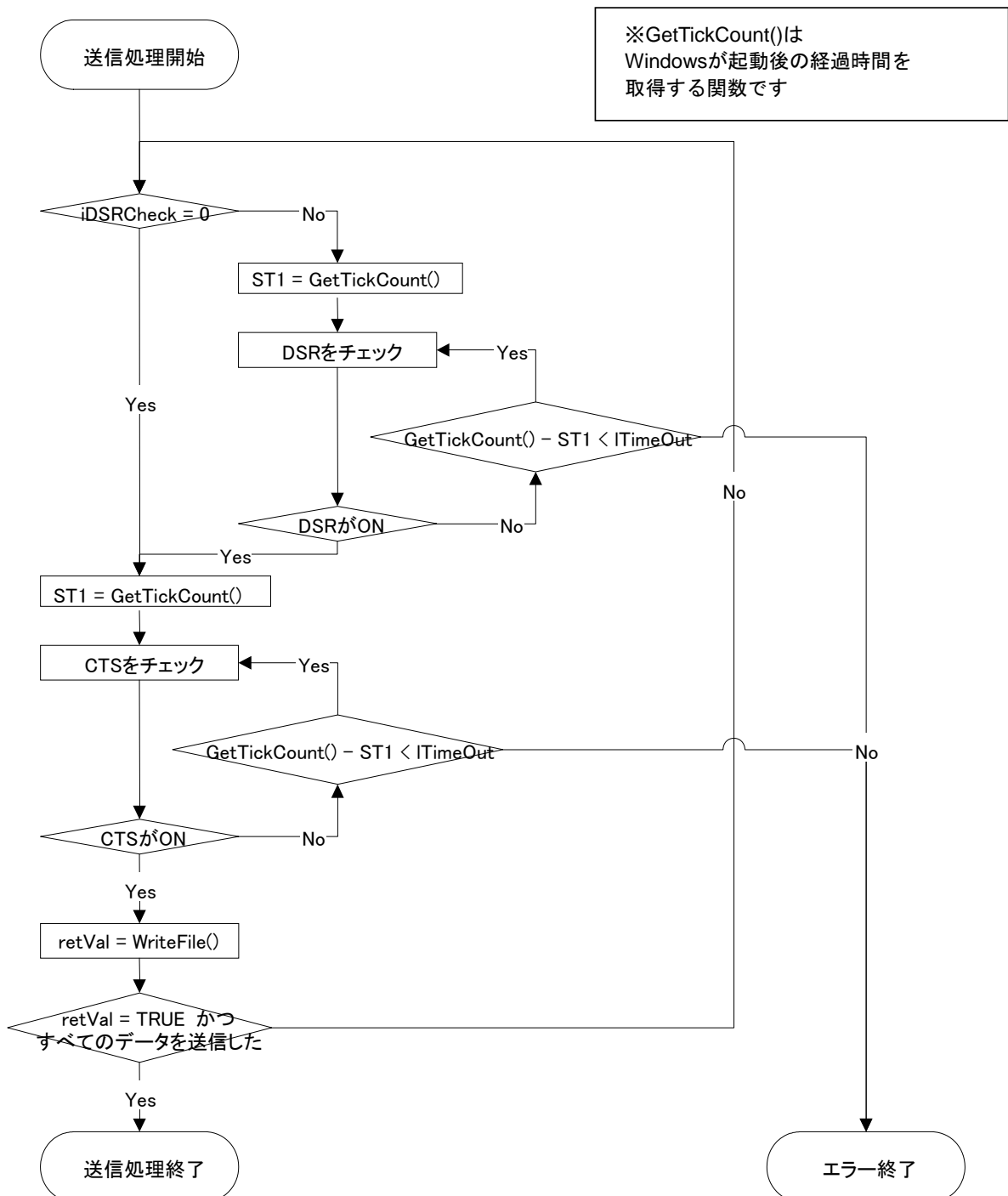
図の例では、WS1とWS2がそれぞれpcxOpenPort、pcxSendCommand、pcxClosePortメソッドを実行していますが、アクセスしているポートが異なるため、並列に実行することができます。

¹ OCX内部でポート別に並列処理が可能な実装となっております。図内のOCXクラスA,Bはあくまで内部の実装の記載ですので、OCX呼び出し側は通常のメソッド呼び出しを行うだけです。

11.3. データ送受信処理について

ここでは、pcxSendCommand()メソッドを例に送信処理・受信処理のフローチャートを次に示します。

11.3.1. 送信処理



11.3.2. 受信処理

