マシンオートメーションコントローラ　ＮＪシリーズ

汎用Ethernet接続ガイド (TCP/IP)
株式会社アイエイアイ

コントローラ編
（X-SELシリーズ）

Network
Connection
Guide

always in control

SBCZ-816A

目　次

1. 関連マニュアル

本資料に関連するマニュアルは以下のとおりです。

Man.No.	形式	マニュアル名称
SBCA-358	形 NJ501-□□□□	NJ シリーズ　CPU ユニット　ユーザーズマニュアル　ハードウェア編
SBCA-359	形 NJ501-□□□□	NJ シリーズ　CPU ユニット　ユーザーズマニュアル　ソフトウェア編
SBCD-359	形 NJ501-□□□□	NJ シリーズ　CPU ユニット内蔵　EtherNet/IP ポート　ユーザーズマニュアル
SBCA-362	形 SYSMAC-SE2□□□	Sysmac Studio Version 1　オペレーションマニュアル
SBCA-360	形 NJ501-□□□□	NJ シリーズ　コマンドリファレンスマニュアル　基本編
SBCA-360	形 NJ501-□□□□	NJ シリーズ　コマンドリファレンスマニュアル　基本編
MJ0116	形 X-SEL-J/K	株式会社アイエイアイ X-SEL コントローラ J/K タイプ 取扱説明書
MJ0134	形 X-SEL-KT	株式会社アイエイアイ グローバル仕様コントローラ X-SEL-KT 取扱説明書
MJ0148	形 X-SEL-P/Q	株式会社アイエイアイ X-SEL コントローラ P/Q タイプ 取扱説明書
MJ0119	形 X-SEL-JX/KX	株式会社アイエイアイ X-SEL コントローラ JX/KX タイプ 取扱説明書
MJ0152	形 X-SEL-PX/QX	株式会社アイエイアイ X-SEL コントローラ PX/QX タイプ 取扱説明書
MJ0154	形 IA-101-X-MW 形 IA-101-X-MW-J 形 IA-101-XA-MW 形 IA-101-X-USB 形 IA-101-X-USBMW	株式会社アイエイアイ X-SEL 用パソコン対応ソフト 取扱説明書
-	形 X-SEL-J/K(KE/KT/KET) 形 X-SEL-JX/KX(KTX) 形 TT 形 X-SEL-P/Q 形 X-SEL-PX/QX 形 SSEL 形 ASEL/PSEL	株式会社アイエイアイ X-SEL シリアル通信仕様書（フォーマット B）

2. 用語と定義

用語	説明・定義
IP アドレス	Ethernet では、IP アドレスを使用して通信を行います。 IP アドレス（インターネットプロトコルアドレス）は、Ethernet 上のノード（ホストコンピュータ、コントローラなど）を識別するためのアドレスです。 IP アドレスは、重複しないように設定や管理を行う必要があります。
ソケット	ソケットは、TCP または UDP の機能をユーザプログラムから直接利用するためのインタフェースです。 NJ シリーズマシンオートメーションコントローラでは、標準搭載のソケットサービス用命令を使用してソケット通信を行います。 ソケットサービスを使用するには、相手ノードとの間でコネクションの確立と切断が必要です。本資料では、確立処理を「ソケットオープン」または「TCP オープン」と、切断処理を「ソケットクローズ」または「クローズ」といいます。 ソケットサービスにより、相手ノードと任意のデータの送受信ができます。
Active と Passive	TCP ソケットのコネクション開設時、各ノードでオープン処理が実行されます。 このときノードがサーバになるか、クライアントになるかによって、オープンの方法が異なります。 本資料では、サーバとしてオープンする場合の処理を「Passive オープン」と、クライアントとしてオープンする場合の処理を「Active オープン」または「オープン処理(Active)」といいます。
keep-alive 機能	TCP/IP のソケットサービスにおいて相手ノード（サーバまたはクライアント）との間で、設定した時間以上に通信しない状態が継続すると、keep-alive の通信フレームを使用して相手ノードとの接続状態を確認します。 応答がなければ、一定間隔で確認を実施し、すべての確認に応答がなければ、コネクションを切断します。
linger 機能	TCP ソケットクローズ時に RST データを送信してポート No.開放までの時間を待たずに、即座に同じポート No.によるオープン処理を可能にする TCP ソケットのオプションです。 linger オプションを指定しない場合、TCP クローズ時に FIN データを発行し、その後の約 1 分間で相手ノードとの間で送達確認などの終了管理を行います。このため、同じポート No.の TCP ソケットを即座に使用できないことがあります。

3. 注意事項

(1) 実際のシステム構築に際しては､システムを構成する各機器･装置の仕様をご確認のうえ､定格･性能に対し余裕を持った使い方をし､万一故障があっても危険を最小にする安全回路などの安全対策を講じてください。
(2) システムを安全にご使用いただくため､システムを構成する各機器･装置のマニュアルや取扱説明書などを入手し､｢安全上のご注意｣｢安全上の要点｣など安全に関する注意事項を含め､内容を確認のうえ使用してください。
(3) システムが適合すべき規格・法規または規制に関しては、お客様自身でご確認ください。
(4) 本資料の一部または全部を、オムロン株式会社の許可なしに複写、複製、再配布することを禁じます。
(5) 本資料の記載内容は、2011 年 12 月時点のものです。
本資料の記載内容は、改良のため予告なく変更されることがあります。

本資料で使われているマークには、次のような意味があります。

安全上の要点

製品を安全に使用するために実施または回避すべきことを示します。

使用上の注意

製品が動作不能、誤動作、または性能や機能への悪影響を予防するために実施または回避すべきことを示します。

参考

必要に応じて読んでいただきたい項目です。
知っておくと便利な情報や、使用するうえで参考となる内容について説明しています。

4. 概要

本資料は、株式会社アイエイアイ（以下、アイエイアイ）製コントローラ（X-SEL シリーズ）を、オムロン株式会社（以下、オムロン）製マシンオートメーションコントローラ　NJ シリーズ（以下、コントローラ）と、Ethernet で接続する手順とその確認方法をまとめたものです。
あらかじめ準備されたプロジェクトファイルの Ethernet 通信設定を通して、設定手順と設定時のポイントを理解することにより、Ethernet 通信接続することができます。
本プロジェクトファイルでは、相手機器に対する「201H（バージョンコード照会）」メッセージの送受信により、Ethernet の接続確認を行います。

オムロンより「Sysmac Studio プロジェクトファイル」の最新ファイルを事前に準備してください。

名称	ファイル名	バージョン
Sysmac Studio プロジェクトファイル（拡張子：SMC）	IAI_X-SEL_ETN(TCP)_V100.SMC	Ver.1.00

5. 対象製品と対象ツール

5.1. 対象製品

接続を保証する対象機器は以下のとおりです。

メーカ	名称	形式	バージョン
オムロン	NJ シリーズ CPU ユニット	形 NJ501-□□□□	ー
アイエイアイ	コントローラ	形X-SEL-□	Ver. 0.28
アイエイアイ	単軸ロボット 直交ロボット スカラロボット	ー	
アイエイアイ	X-SEL 用パソコン対応ソフト	形 IA-101-X-MW	V7.07.10.00

参考

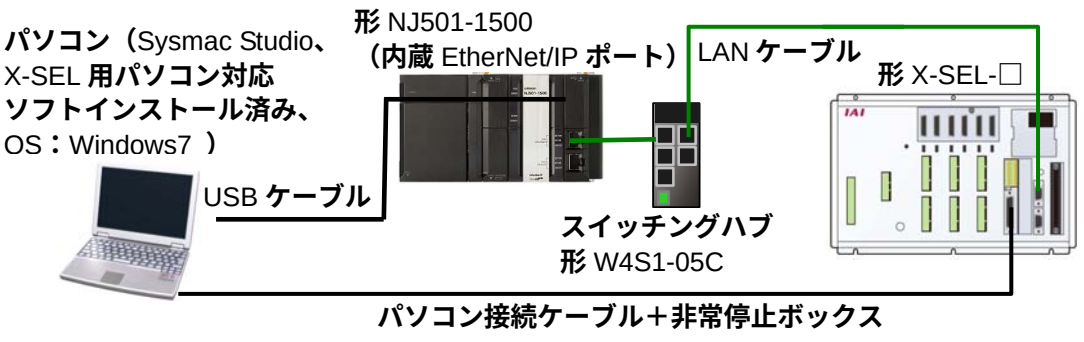
本資料は機器の通信接続確立までの手順について記載したものであって、機器個別の操作や設置および配線方法に関しては記載しておりません。
上記製品（通信接続手順以外）の詳細に関しましては、対象製品の取扱説明書を参照するか、機器メーカまでお問い合わせください。
（株式会社アイエイアイ　http://www.iai-robot.co.jp/）
上記連絡先は、本資料作成時点のものです。最新情報は各機器メーカにご確認ください。

参考

X-SEL コントローラに接続可能なアクチュエータに関しましては、機器メーカまでお問い合わせください。
（株式会社アイエイアイ http://www.iai-robot.co.jp）

5.2. デバイス構成

本資料の接続手順を再現するための構成機器は以下のとおりです。



メーカ	名称	形式	バージョン
オムロン	NJ シリーズ　CPU ユニット（内蔵 EtherNet/IP ポート）	形 NJ501-1500	
オムロン	電源ユニット	形 NJ-PA3001	
オムロン	スイッチングハブ	形 W4S1-05C	
オムロン	Sysmac Studio	形 SYSMAC-SE2□□□	Ver.1.00
オムロン	Sysmac Studio プロジェクトファイル	IAI_X-SEL_ETN(TCP)_V100.SMC	Ver.1.00
―	パソコン(OS：Windows7)		
―	USB ケーブル（USB2.0 準拠 B コネクタ）		
―	LAN ケーブル（Ethernet カテゴリ 5 以上の STP（シールドツイストペア）ケーブル）		
アイエイアイ	コントローラ	形 X-SEL-□	Ver.0.28
アイエイアイ	X-SEL 用パソコン対応ソフト	形 IA-101-X-MW	V7.07.10.00
アイエイアイ	パソコン接続ケーブル＋非常停止ボックス	形 CB-ST-E1MW050-EB	

使用上の注意

オムロン株式会社より「Sysmac Studio プロジェクトファイル」の最新ファイルを事前に準備してください。
（ファイルの入手については、オムロン株式会社までお問い合わせください）

参考

構成機器、バージョンが異なる場合再現できないことがあります。構成、形式、バージョンを確認のうえ、お客様の構成と異なる場合は、オムロンまでお問い合わせください。

参考

本資料ではコントローラとの接続に USB を使用します。USB ドライバのインストールについては、「Sysmac Studio Version1.0 オペレーションマニュアル」(SBCA-362)の「付録 A-1 USB ケーブルで直接接続する場合のドライバのインストール方法」を参照してください。

参考

パソコンとアイエイアイ製コントローラとの接続に使用するケーブルおよびパソコン対応ソフトは、コントローラの機種により異なります。詳細は各コントローラの取扱説明書を参照してください。

6. Ethernet の設定内容

本資料で設定する通信パラメータおよび変数名などの仕様を示します。

参考

本資料および本プロジェクトファイルでは、本章に記載されている設定およびコマンドのみの動作が可能です。本設定以外の通信を行うためには、改造が必要です。

6.1. Ethernet通信設定

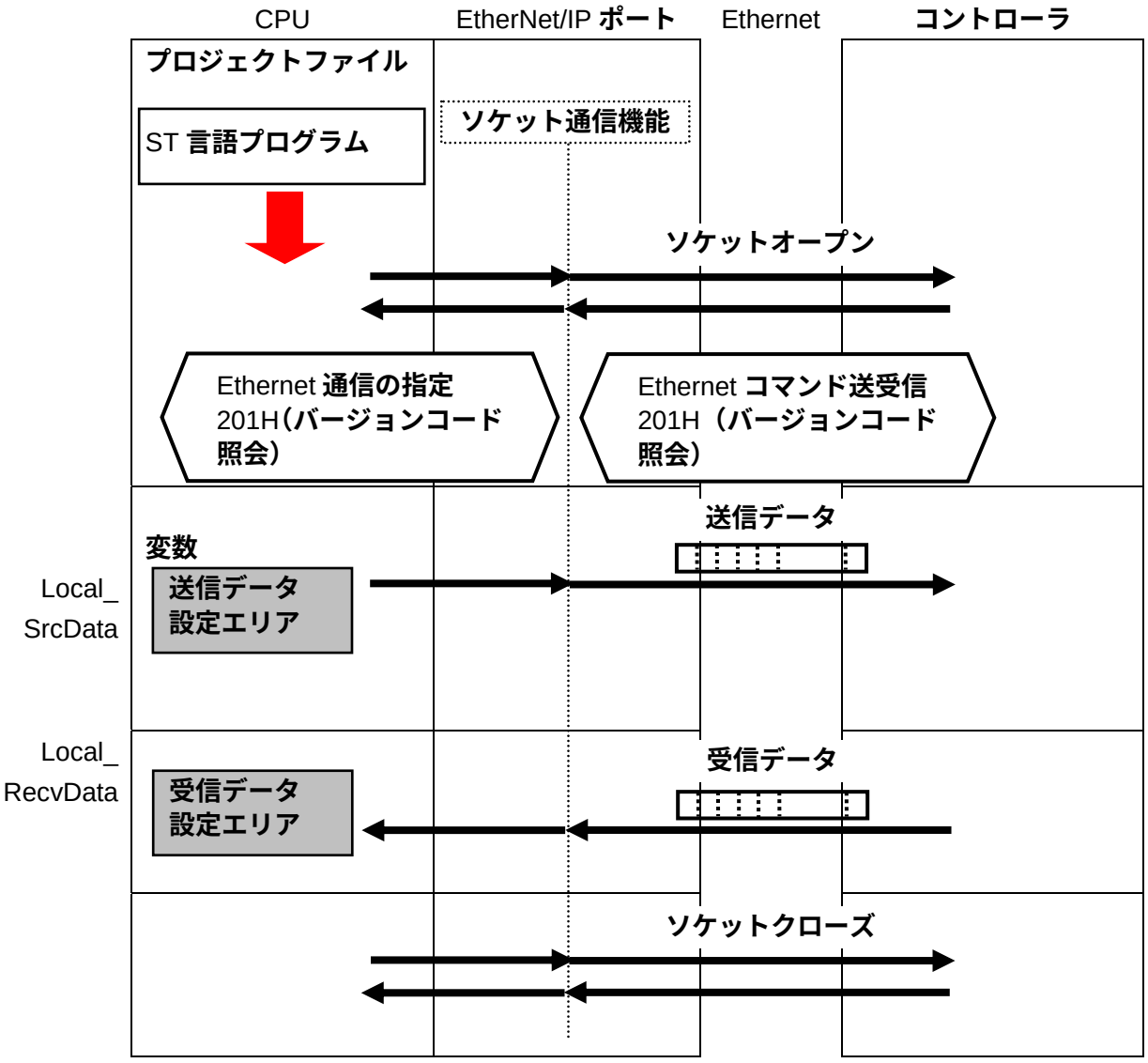
Ethernet 通信を行うための設定は、以下になります。

	形 NJ501-1500	形 X-SEL-□
IP アドレス	192.168.250.1	192.168.250.2
サブネットマスク	255.255.255.0	255.255.255.0
ゲートウェイ	-	（任意）
ポート No.	（プロジェクトファイルで設定）	2000
局番 （ユーザ開放 SIO チャンネル 0 局コード）	-	153（初期値）

※本資料では、同一セグメント内の接続のため、ゲートウェイの設定は不要です。

6.2. 通信接続確認例

本資料では、ストラクチャード・テキスト（以下、ST）言語によるプログラムで、コントローラからアイエイアイ製コントローラに対して「ソケットオープン」、「送受信」、「ソケットクローズ」を実行する場合を例とします。
コントローラとアイエイアイ製コントローラ間では、「201H（バージョンコード照会）」のメッセージを送受信します。動作概要を以下に示します。



7. 接続手順

本章では、コントローラを Ethernet 接続する手順について記載します。
本資料では、コントローラおよびアイエイアイ製コントローラが工場出荷時の初期設定状態であることを前提として説明します。各機器の初期化については「8.初期化方法」を参照してください。

7.1. 作業の流れ

コントローラを Ethernet 接続設定する手順は以下のとおりです。

7.2 アイエイアイ製コントローラの設定	アイエイアイ製コントローラの設定を行います。
▼	
7.2.1 パラメータ設定	コントローラのパラメータを設定します。
▽	
7.3 コントローラの設定	コントローラの設定を行います。
▼	
7.3.1 Sysmac Studio の起動とプロジェクトファイルの読み込み	オートメーションソフトウェア「Sysmac Studio」を起動し、「Sysmac Studio プロジェクトファイル」を読み込みます。
▼	
7.3.2 パラメータの確認とビルドの実行	設定パラメータを確認し、プロジェクトデータのプログラムチェックおよびビルドを実行します。
▼	
7.3.3 オンライン接続とプロジェクトデータの転送	「Sysmac Studio」をオンライン接続し、プロジェクトデータをコントローラに転送します。
▽	
7.4 接続状態確認	転送したプロジェクトファイルを実行し、Ethernet 通信が正しく行われていることを確認します。
▼	
7.4.1. プロジェクトファイルの実行と受信データの確認	プロジェクトファイルを実行し、コントローラの変数に正しいデータが書き込まれていることを確認します。

使用上の注意

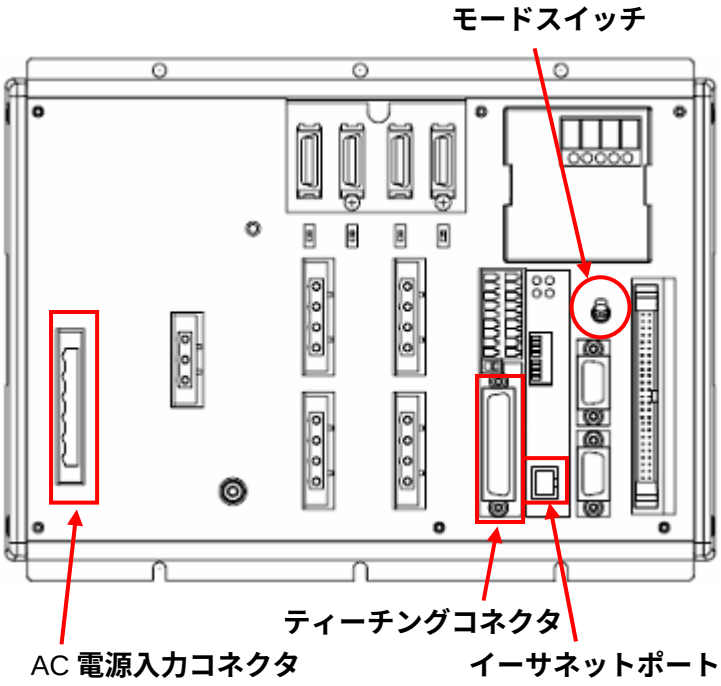
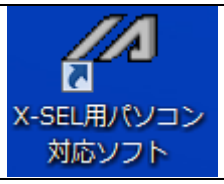
オムロン株式会社より「Sysmac Studio プロジェクトファイル」の最新ファイルを事前に準備してください。
（ファイルの入手については、オムロン株式会社までお問い合わせください）

7.2. アイエイアイ製コントローラの設定

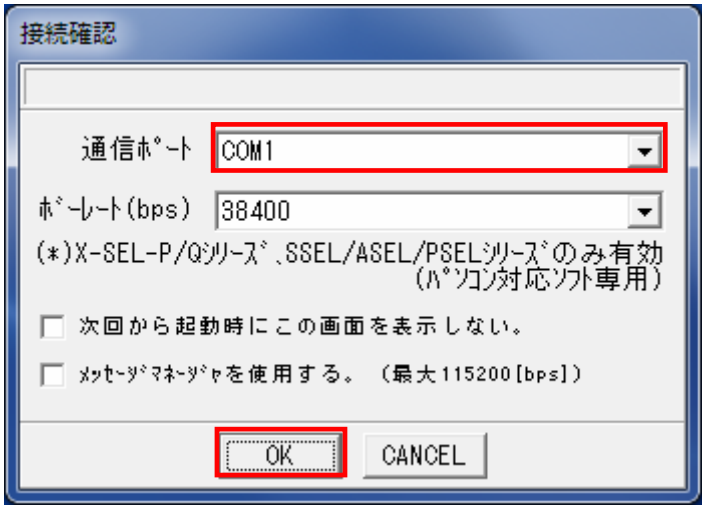
アイエイアイ製コントローラの設定を行います。

7.2.1. パラメータ設定

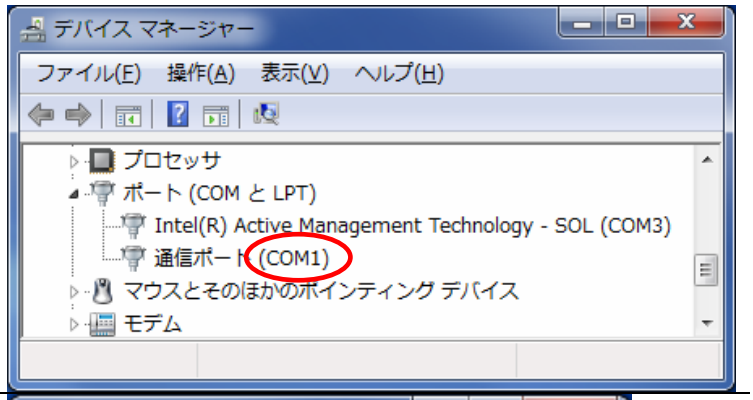
コントローラのパラメータを設定します。
パラメータの設定は「X-SEL 用パソコン対応ソフト」で行いますので、ソフトをあらかじめパソコンにインストールしてください。
なお、ソフトのインストール方法については「X-SEL 用パソコン対応ソフト取扱説明書」(MJ0154)を参照してください。

1	コントローラ前面のコネクタおよびスイッチの位置を確認します。 コントローラとパソコンを RS-232C ケーブルで接続します。RS-232C ケーブルはコントローラの［ティーチングコネクタ］に接続します。 コントローラの［イーサネットポート］とスイッチングハブ間を LAN ケーブルで接続します。 ［AC 電源入力コネクタ］に電源ケーブルを接続します。	
2	コントローラ前面のモードスイッチを[MANU]側に設定します。	
3	コントローラに電源を投入し、パソコンから［X-SEL 用パソコン対応ソフト］を起動します。	

4 **ソフトインストール後の初回起動時のみ、[アプリケーション設定] ダイアログが表示されます。「ポート」には「COM ポート番号」を選択し、[OK]をクリックします。**

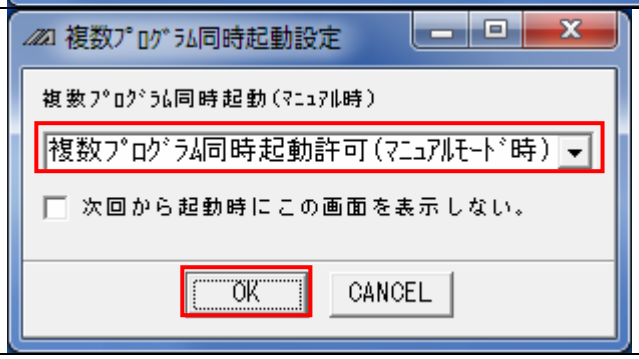


※「パソコンのシリアルポート」が複数存在する場合は、Windows のデバイスマネージャを表示し、「ポート（COM と LPT)」の下の「アイエイアイの機器が接続されている COM ポート番号（右図の例：COM1)」と同じポートを選択します。

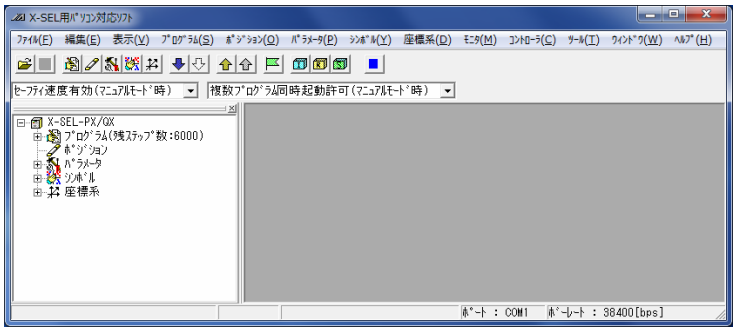


※デバイスマネージャは［コントロールパネル］から、[デバイスマネージャ］を選択してください。

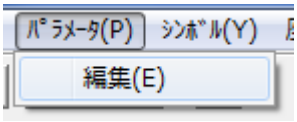
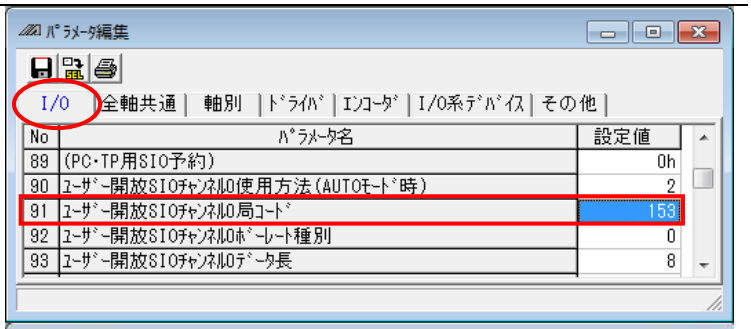
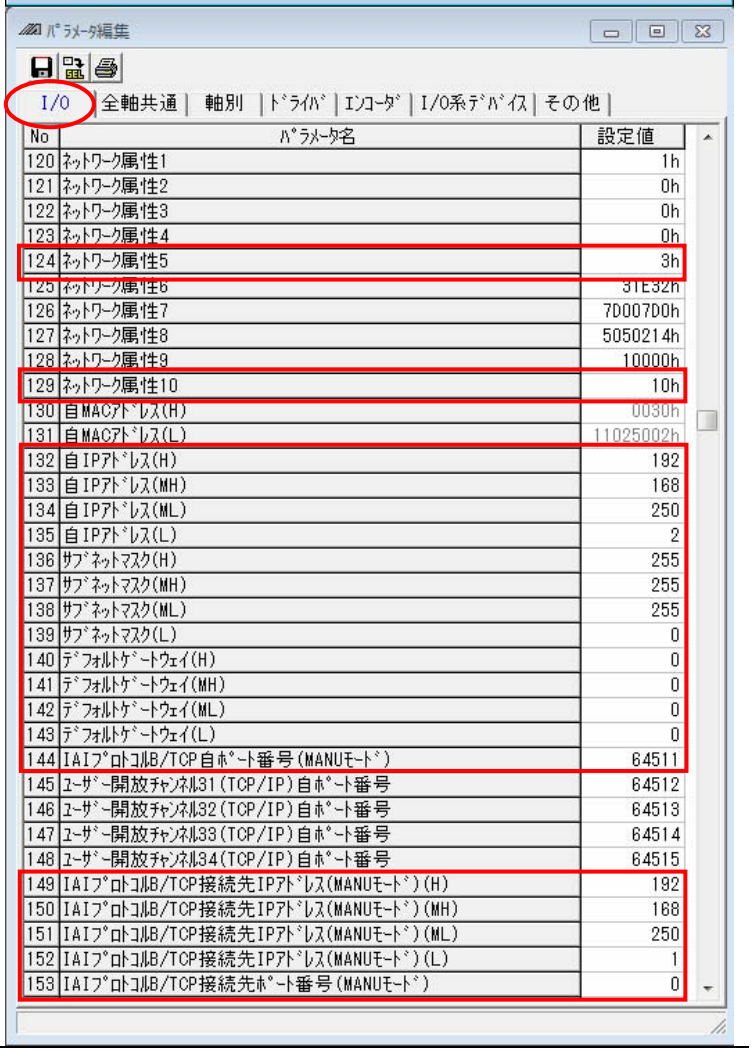
5 **［複数プログラム同時起動設定］ダイアログが表示されます。［複数プログラム同時起動許可（マニュアルモード時)］を選択し、[OK]をクリックします。**



6 **X-SEL 用パソコン対応ソフトが起動されます。**



※エラーダイアログが表示された場合は、[OK]をクリックします。

7	メニューバーから［パラメータ］－［編集］を選択します。	
8	［パラメータ編集］ダイアログの[I/O]タブが表示されますので、次の【変更パラメーター覧】のように変更します。 右図は、設定変更後の値になります。 ※各パラメータの詳細や工場出荷時の初期値は、「株式会社アイエイアイ X-SEL コントローラ PX/QX タイプ取扱説明書」(MJ0152)の「付録」－「パラメーター覧表」を参照してください。	 

【変更パラメーター一覧】

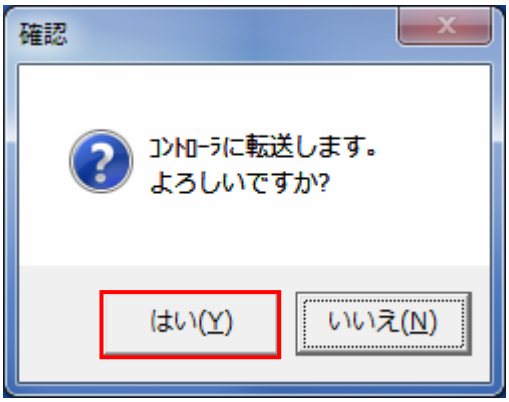
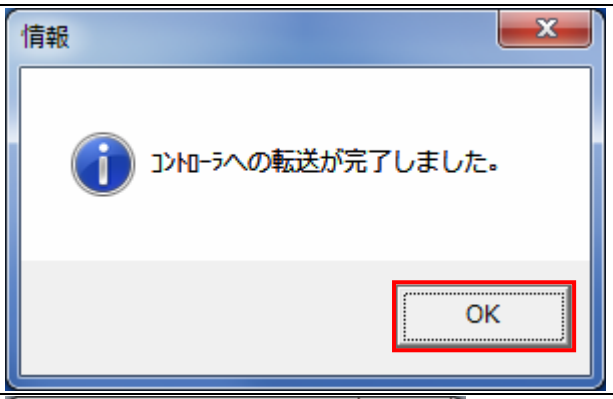
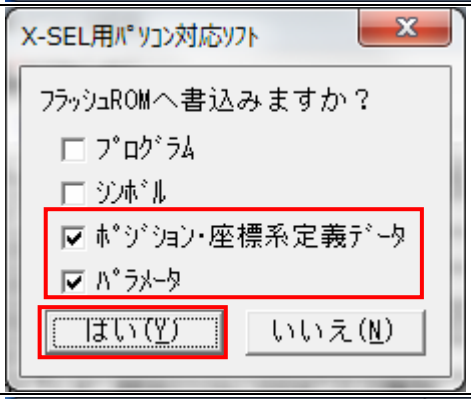
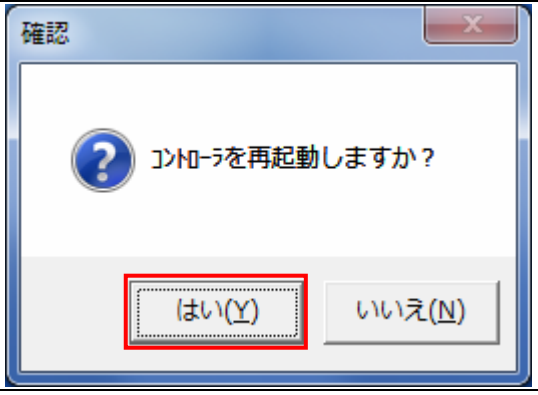
No.	パラメータ名	設定値	備考
91	ユーザ開放 SIO※1 チャンネル 0 局コード	153(99h)	（初期値）
124	ネットワーク属性 5（イーサネットクラサバ種別※2：サーバ）	3h	
129	ネットワーク属性 10 （イーサネット動作規定：TCP/IP メッセージ通信）	10h	
132	自 IP アドレス(H)※3	192	（初期値）
133	自 IP アドレス(MH)※3	168	（初期値）
134	自 IP アドレス(ML)※3	250	
135	自 IP アドレス(L)※3	2	
136	サブネットマスク(H)※3	255	（初期値）
137	サブネットマスク(MH)※3	255	（初期値）
138	サブネットマスク(ML)※3	255	（初期値）
139	サブネットマスク(L)※3	0	（初期値）
140	デフォルトゲートウェイ(H)※3	0	（初期値）
141	デフォルトゲートウェイ(MH)※3	0	（初期値）
142	デフォルトゲートウェイ(ML)※3	0	（初期値）
143	デフォルトゲートウェイ(L)※3	0	（初期値）
144	IAI プロトコル B※4/TCP 自ポート番号（MANU モード）	64511	（初期値）
149	IAI プロトコル B※4/TCP 接続先 IP アドレス（MANU モード）(H)※3	192	（初期値）
150	IAI プロトコル B※4/TCP 接続先 IP アドレス（MANU モード）(MH)※3	168	（初期値）
151	IAI プロトコル B※4/TCP 接続先 IP アドレス（MANU モード）(ML)※3	250	
152	IAI プロトコル B※4/TCP 接続先 IP アドレス（MANU モード）(L)※3	1	
153	IAI プロトコル B※4/TCP 接続先ポート番号（MANU モード）	0	

※1 相手機器の局番設定はユーザ開放 SIO（シリアル I/O）の設定と共用です。

※2 相手機器のクライアント／サーバ種別を設定します。

※3 H：第 1 オクテット、MH：第 2 オクテット、ML：第 3 オクテット、L：第 4 オクテット

※4「IAI プロトコル B」とは、相手機器 X-SEL シリーズが対応する通信仕様「フォーマット B」のことです。

9	パラメータ設定後は、[コントローラへ転送]をクリックします。	
	ダイアログが表示されますので、[はい]をクリックします。 ※パラメータの設定値に変更がない場合は、10～11 項の画面は表示されませんので、12 項へ進んでください。	
10	ダイアログが表示されますので、[OK]をクリックします。	
11	ダイアログが表示されますので、[ポジション・座標系定義データ]と[パラメータ]にチェックを入れて[はい]をクリックします。	
12	ダイアログが表示されますので、[はい]をクリックします。	

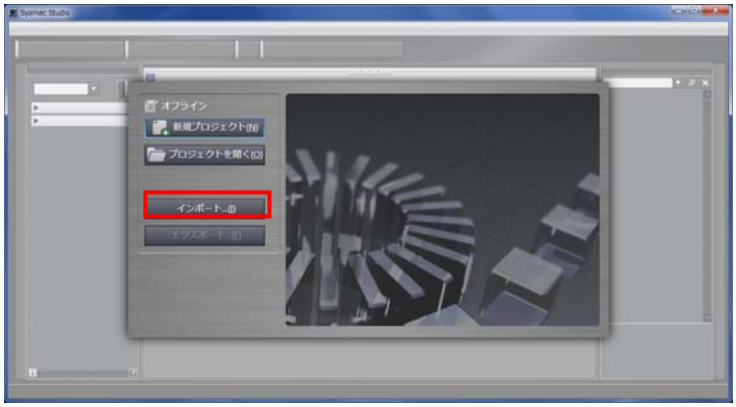
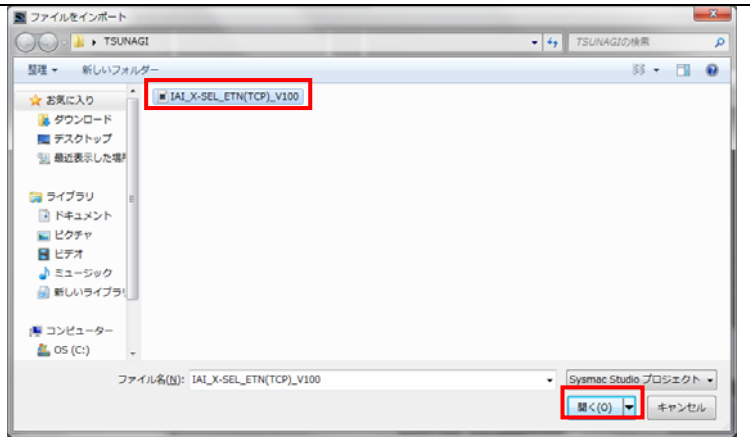
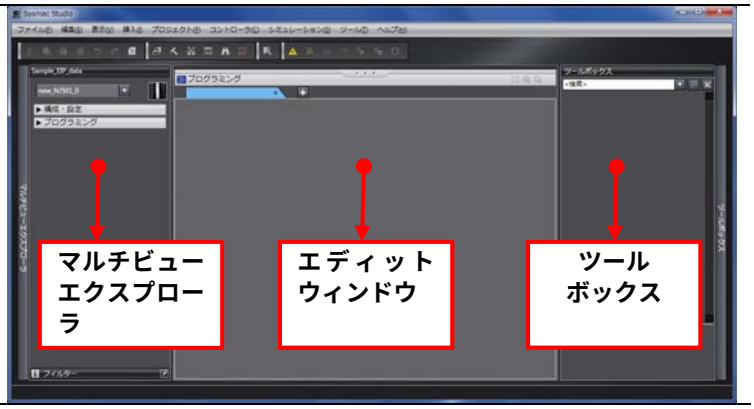
13	**再起動が実行され、コントローラと再接続できたことを確認し、**X-SEL **パソコン対応ソフトを終了します。** X-SEL **パソコン対応ソフトとコントローラが接続されていると、コントローラとのイーサネット通信を行うことができません。**

7.3. コントローラの設定

コントローラの設定を行います。

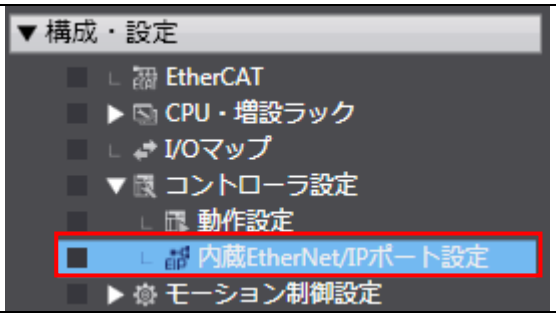
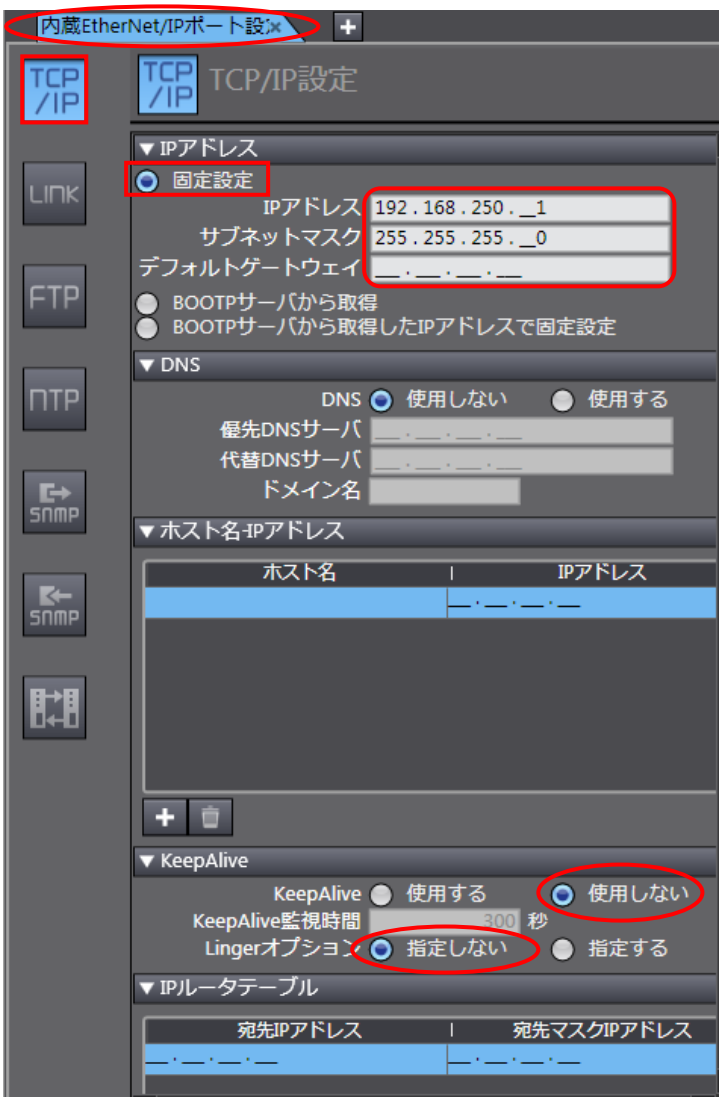
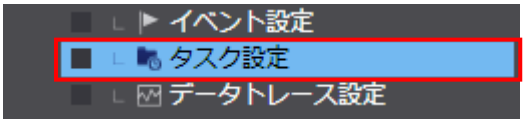
7.3.1. Sysmac Studioの起動とプロジェクトファイルの読み込み

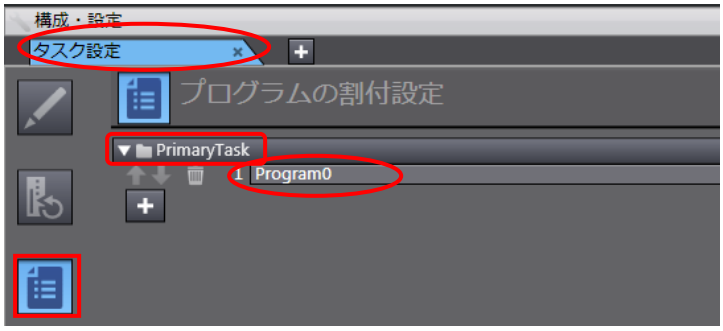
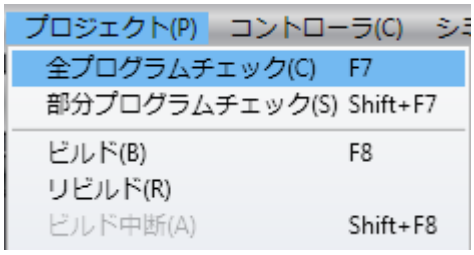
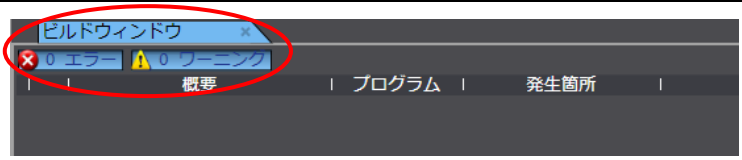
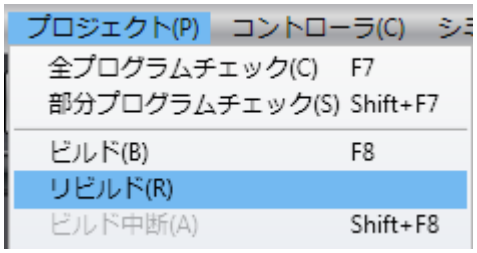
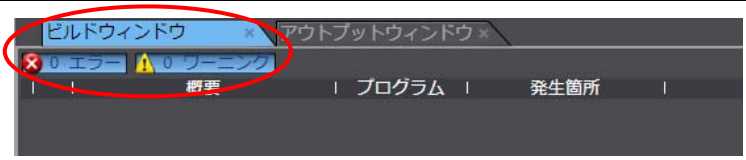
オートメーションソフトウェア「Sysmac Studio」を起動し、「Sysmac Studio プロジェクトファイル」を読み込みます。
ツールソフトと USB ドライバをあらかじめパソコンにインストールしてください。また、USB ケーブルをパソコンとコントローラに接続し、コントローラの電源を投入してください。

1	Sysmac Studio を起動します。 ［インポート］をクリックします。 ※起動時に、アクセス権確認用のダイアログが表示される場合、起動する選択を行ってください。	
2	［プロジェクトをインポート］ダイアログが表示されますので、[IAI_X-SEL_ETN(TCP)_V100.SMC]（Sysmac Studio プロジェクトファイル）を選択し、［開く］をクリックします。 ※使用する「Sysmac Studio プロジェクトファイル」は、オムロンより入手してください。	
3	[IAI_X-SEL_ETN(TCP)_V100]プロジェクト画面が表示されます。 画面左側を「マルチビューエクスプローラ」、右側を「ツールボックス」、中央を「エディットウィンドウ」といいます。	

7.3.2. パラメータの確認とビルドの実行

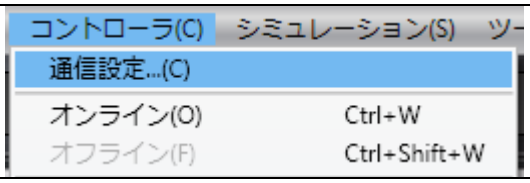
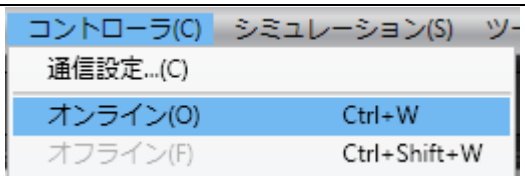
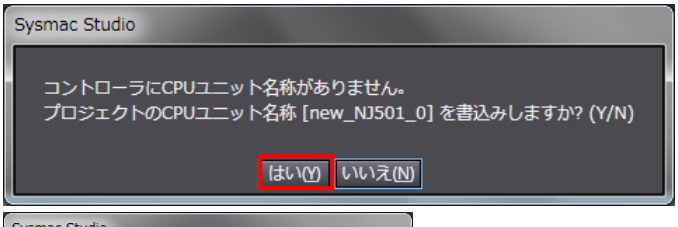
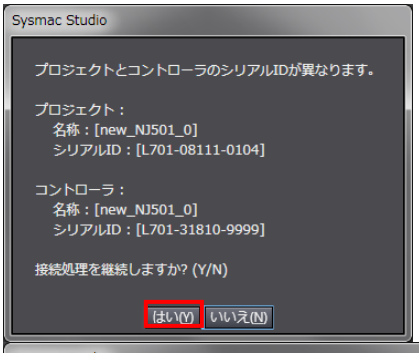
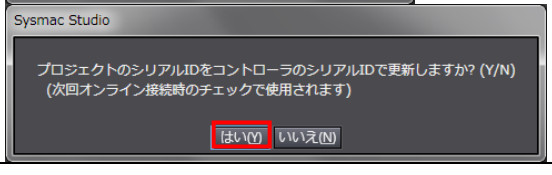
設定パラメータを確認し、プロジェクトデータのプログラムチェックおよびビルドを実行します。

1	［マルチビューエクスプローラ］から、［構成・設定］－［コントローラ設定］－［内蔵 EtherNet/IP ポート設定］をダブルクリックします。	
2	［エディットウィンドウ］に、［内蔵 EtherNet/IP ポート設定］タブが表示されます。 [TCP/IP]を選択し、[IP アドレス]の［固定設定］のチェックボックスを選択し、以下の設定であることを確認します。 IP アドレス：192.168.250.1 サブネットマスク： 255.255.255.0 デフォルトゲートウェイ： _._._._ [KeepAlive]の設定が以下の設定であることを確認します。 KeepAlive：使用しない Linger オプション： 指定しない	
3	［マルチビューエクスプローラ］から、［構成・設定］－［タスク設定］をダブルクリックします。	

4	［エディットウィンドウ］に、［タスク設定］タブが表示されます。 ［プログラムの割付設定］を選択し、[Primary Task]に[Program0]が設定されていることを確認します。	
5	メニューバーから、［プロジェクト］－［全プログラムチェック］を選択します。	
6	［エディットウィンドウ］下に、［ビルドウィンドウ］が表示されます。 エラーおよびワーニングが、ともに「0」であることを確認します。	
7	メニューバーから、［プロジェクト］－［リビルド］を選択します。 変換中の画面が表示されます。	 
8	［ビルドウィンドウ］内のエラーおよびワーニングが、ともに「0」であることを確認します。	

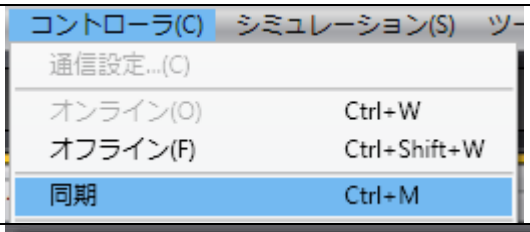
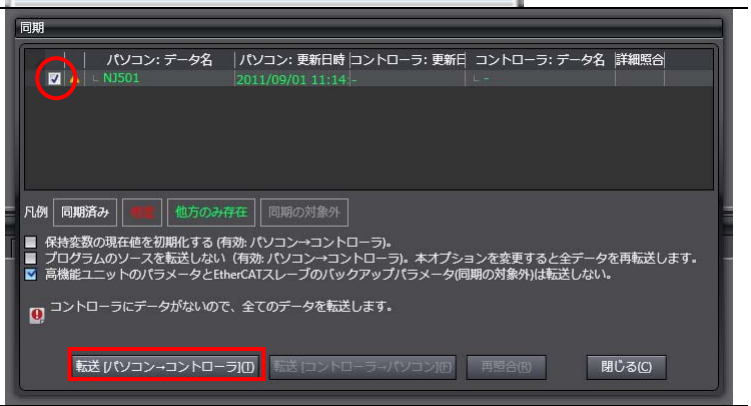
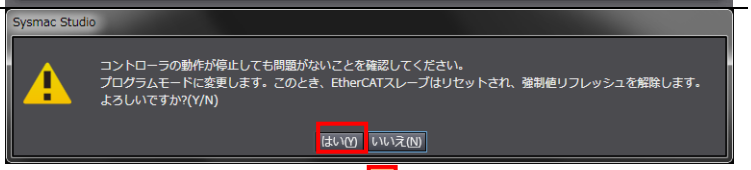
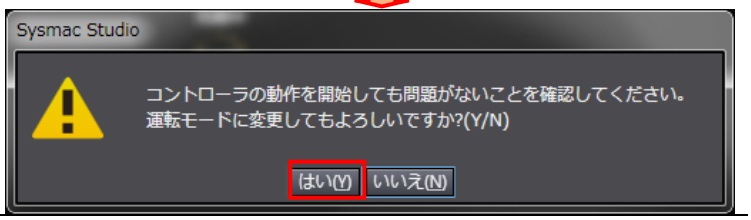
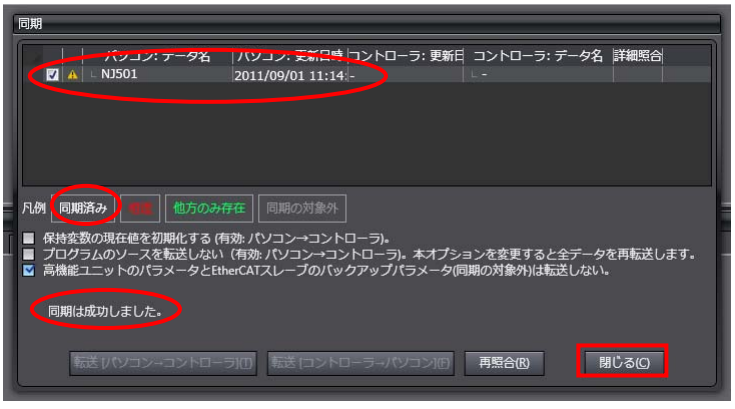
7.3.3. オンライン接続とプロジェクトデータの転送

「Sysmac Studio」をオンライン接続し、プロジェクトデータをコントローラに転送します。

1	メニューバーから、[コントローラ] － [通信設定] を選択します。	
2	[通信設定] ダイアログが表示されます。 [接続方法] から、[USB-直接接続] を選択します。 [OK]をクリックします。	
3	メニューバーから、[コントローラ] － [オンライン] を選択します。 確認のダイアログが表示されますので、[はい] をクリックします。 ※使用するコントローラの状態により、表示されるダイアログが異なりますが、[はい] や [Yes]など処理を進める選択を行ってください。 ※表示されるシリアル ID は機器により異なります。	   

参考

コントローラとのオンライン接続に関する詳細については、「Sysmac Studio Version1.0　オペレーションマニュアル」（SBCA-362）の「第 5 章 コントローラとの接続」を参照してください。

4	**オンライン状態になると、［エディットウィンドウ］の上段に、黄色い枠が表示されます。**	
5	**メニューバーから、［コントローラ］－［同期］を選択します。**	
6	**［同期］ダイアログが表示されます。** **転送したいデータ（右図では、[NJ501]）にチェックが付いていることを確認して、[転送[パソコン→コントローラ]］をクリックします。**	
7	**確認ダイアログが表示されますので、[はい] をクリックします。** **同期中の画面が表示されます。** **確認ダイアログが表示されますので、[はい] をクリックします。**	  
8	**同期したデータの文字色が［同期済み］色になり、「同期は成功しました。」と表示されていることを確認します。問題がなければ、［閉じる］をクリックします。** **※同期が失敗した場合は、配線を確認のうえ、本項の手順を再実行してください。**	

7.4. 接続状態確認

転送したプロジェクトファイルを実行し、Ethernet 通信が正しく行われていることを確認します。

使用上の注意

以降の手順を実施する前に、LAN ケーブルが接続されていることを確認ください。
接続されていない場合、各機器の電源を OFF にしてから LAN ケーブルを接続してください。

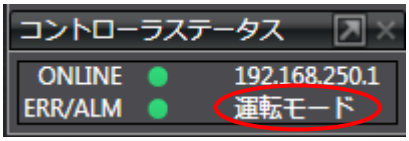
7.4.1. プロジェクトファイルの実行と受信データの確認

プロジェクトファイルを実行し、コントローラの変数に正しいデータが書き込まれていることを確認します。

安全上の要点

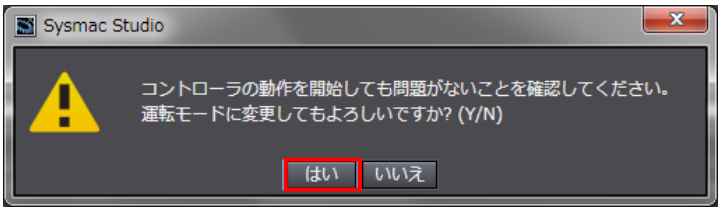
プロジェクトファイルを実行するときは、安全を十分に確認してから行ってください。
ユニットの動作モードにかかわらず、接続機器が誤動作し、けがをする恐れがあります。

1 Sysmac Studio の［コントローラステータス］ウィンドウに、［運転モード］が表示されていることを確認します。

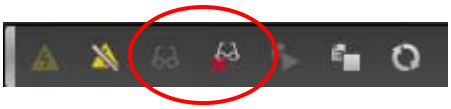


［プログラムモード］の場合は、メニューバーから、［コントローラ］－［動作モード］－［運転モード］を選択します。

確認用のダイアログが表示されますので、［はい］を選択します。

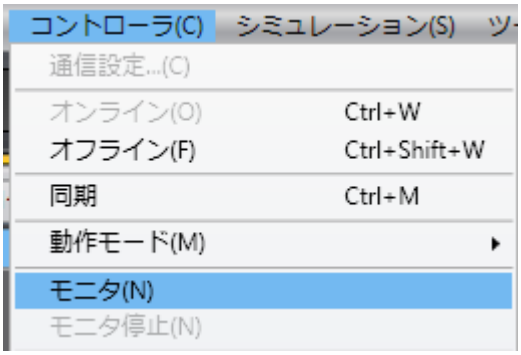


2 コントローラが、モニタ状態であることを、Sysmac Studio のツールバーの［モニタ］および［モニタ停止］ボタンで確認します。
右図のように［モニタ］ボタンが選択されて選択不可状態であり、［モニタ停止］ボタンが選択可能状態（モニタ状態）であることを確認します。
※モニタ停止状態の場合は、Sysmac Studio のメニューバーから、［コントローラ］－［モニタ］を選択します。

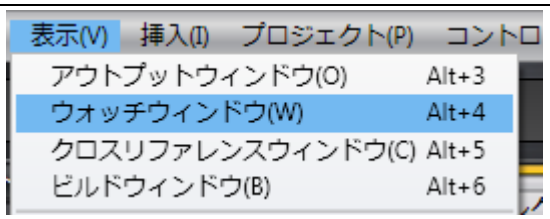


モニタ

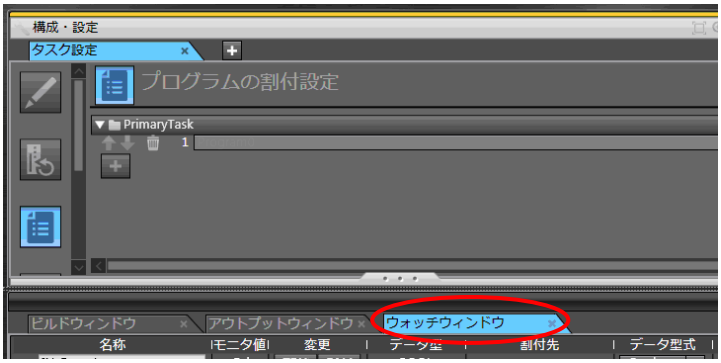
モニタ停止



3 メニューバーから、［表示］－［ウォッチウィンドウ］を選択します。



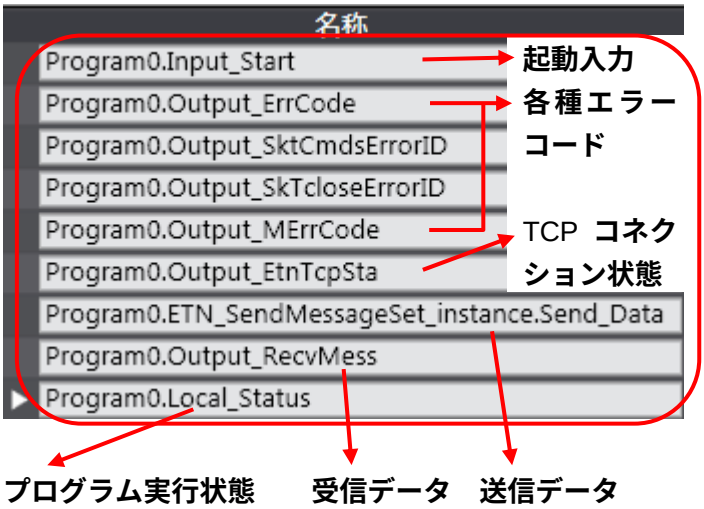
4 ［エディットウィンドウ］の下段に、［ウォッチウィンドウ］タブが表示されます。



5 ［名称］エリアに右図に示す変数が表示されていることを確認します。

※必要な変数が表示されていない場合は、［名前を入力...］をクリックして追加してください。

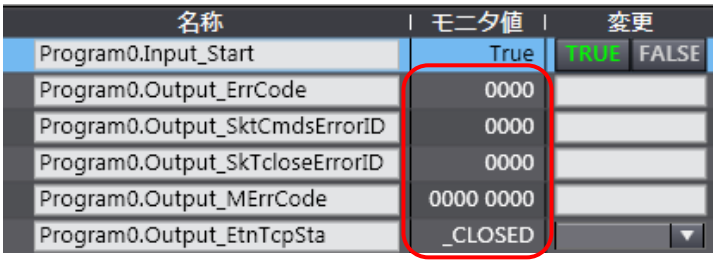
※以降の説明では、［名称］の「Program0」は省略して説明します。



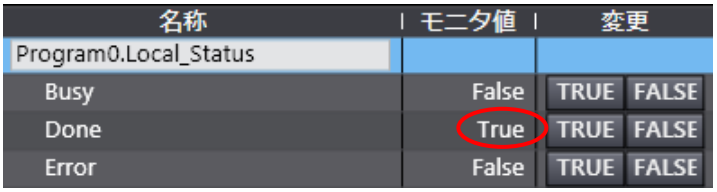
6 [Input_Start]の［変更］エリアの[TRUE]をクリックします。

[Input_Start]の［モニタ値］が[True]になります。
プログラムが動作し、相手機器とEthernet 通信が行われます。

7 通信が正常終了すると、各エラーコードが「0」になります。
TCP コネクション状態(Output_EtnTcpSta)は、[_CLOSED]になります。
※異常終了した場合は、発生した異常に応じて、エラーコードが格納されます。エラーコードの詳細は、「9.7.異常処理」を参照してください。



また、プログラムの実行状態を示す[Local_Status.Done]の［モニタ値］が[True]になります。異常終了時は、[Local_Status.Error]が[True]になります。
※[Input_Start]を[FALSE]にすると、[Local_Status]の各変数も[False]になります。詳しくは、「9.6.タイムチャート」を参照してください。



8

相手機器から受信したレスポンスデータは、Output_RecvMess**に格納されます。**
（ETN_SendMessageSet_instance.Send_Data **は送信コマンドです。）**
右図のように、参照したいエリアを［ウォッチウィンドウ］に指定して確認します。

※右図の受信内容は、ご使用の環境によって異なります。

※コマンドの詳細は「9.2.2. **コマンドの詳細説明」を参照してください。**

名称
Program0.ETN_SendMessageSet_instance.Send_Data
Program0.Output_RecvMess

モニタ値
!99201000B6RL
#99201000C271001C07D60C1B0E37006FRL

受信内容

- **ヘッダ：** “#”
- **局：** “99”
- **伝文** ID**：** “201”
- **ユニット種別：** “00”
- **デバイス** No.**：** “0”
- **機種コード：** “C2”
- **ユニットコード：** “71”
- **バージョン** No.**：** “001C” (V.0.28)
- **時刻（年）：** “07D6” (2006)
- **時刻（月）：** “0C” (12)
- **時刻（日）：** “1B” (27)
- **時刻（時）：** “0E” (14)
- **時刻（分）：** “37” (55)
- **時刻（秒）：** “00” (00)
- SC(**チェックサム**)**：** “6F”
- **フッタ：** “RL ”([CR][LF])

コントローラのバージョンは X-SEL **用パソコン対応ソフトのメニューバーで［コントローラ］－［**ROM **バージョン情報］を選択すると確認できます。**

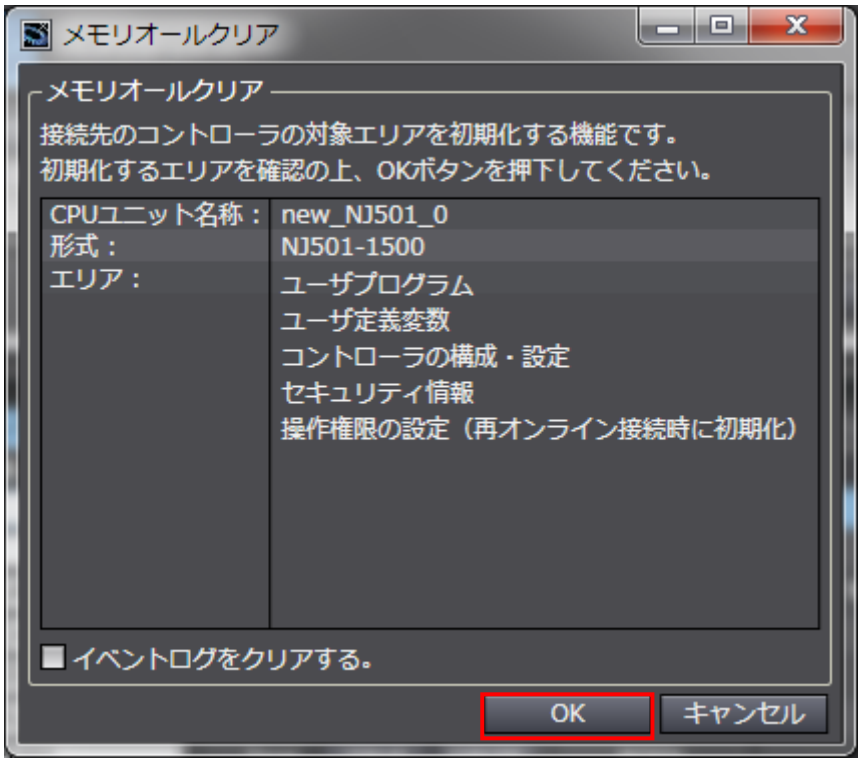
右図のように受信データと同じであることが確認できます。

8. 初期化方法

本資料では、各機器が工場出荷時の初期設定状態であることを前提としています。
初期設定状態から変更された機材を利用される場合には、各種設定が手順どおりに進めることができない場合があります。

8.1. コントローラ

コントローラの設定を初期設定状態に戻すためには、Sysmac Studio のメニューバーから［コントローラ］－［メモリオールクリア］を選択して処理を進めてください。



8.2. アイエイアイ製コントローラ

アイエイアイ製コントローラの初期化方法については、「X-SEL 用パソコン対応ソフト　取扱説明書」(MJ0154)の「付録：パラメータ（工場出荷時）初期化方法」を参照してください。

9. プロジェクトファイル

本資料で使用するプロジェクトファイルの詳細を示します。

9.1. 概要

本章では、アイエイアイ製コントローラ（形 X-SEL-□）（以下、「相手機器」と略す）をコントローラ（内蔵 EtherNet/IP ポート）（以下、「内蔵 EtherNet/IP ポート」と略す）に接続するためのプロジェクトファイルの仕様および機能について説明します。

プロジェクトファイルとは、「Sysmac Studio」プロジェクトファイルを指します。
本プロジェクトファイルでは、あらかじめ下記のデータが組み込まれています。
・内蔵 EtherNet/IP ポートの通信設定およびプログラムのタスク設定
・ソケット通信を行うためのプログラムとファンクションブロック
・ST 言語プログラムで使用する変数の「変数テーブル」およびデータ型の定義

本プロジェクトファイルでは、内蔵 EtherNet/IP ポートのソケットサービス機能を使用し、相手機器に対して「201H（バージョンコード照会）」を行い、正常／異常終了を判定します。
本プロジェクトファイルの正常終了は、TCP ソケット通信の正常終了とします。
また異常終了は、TCP ソケット通信の異常終了および相手機器の異常（相手機器からのレスポンスデータより判定）とします。

TCP ソケットオプションである keep-alive 機能および linger 機能は、本プロジェクトファイルでは使用していません。お客様のアプリケーション設計時に必要に応じてご検討ください。

参考

本プロジェクトファイルは、当社の実施した試験構成、各商品バージョン、評価に使用した商品ロットにおいて通信が可能であることを確認しております。
電気的ノイズ等の外乱下や機器自体の性能のばらつきにおいて、動作を保証するものではありません。

参考

Sysmac Studio では、10 進データと 16 進データの区別が必要な場合には、10 進データの先頭に変数型名＋'#'、16 進データの先頭に変数型名＋'#'＋'16'＋'#'を付け区別します。（10 進「INT#1000」→ 16 進「INT#16#03E8」など。DINT の場合変数型名＋'#'は不要。）

9.1.1. 通信データの流れ

内蔵 EtherNet/IP ポートから相手機器に対して TCP ソケット通信によりコマンドを発行し、相手機器からレスポンスデータを受信するまでの流れです。本プロジェクトファイルでは、TCP オープンからクローズまでの一連の処理を連続実行します。レスポンスデータが分割され複数の受信データとして到着する場合には、受信処理を繰り返し行います。

1.	TCP オープン処理	内蔵 EtherNet/IP ポートから相手機器に対して TCP オープン要求を発行し、TCP コネクションを確立します。
▼		
2.	コマンド送信処理	ST 言語によるプログラムで設定した送信メッセージを内蔵 EtherNet/IP ポートから相手機器に対して発行します。
▼		
3.	レスポンス受信処理	内蔵 EtherNet/IP ポートで取り込んだ相手機器からのレスポンスデータを、指定された CPU ユニットの内部メモリに格納します。
▼		
4.	クローズ処理	内蔵 EtherNet/IP ポートから相手機器に対してクローズ要求を発行し、TCP コネクションを切断します。

※相手機器や使用するコマンドにより、「コマンド受信後レスポンスデータを送信しない場合」や、「コネクション確立後すぐにレスポンスデータを送信する場合」があります。このため、本プロジェクトファイルでは、「汎用 Ethernet 通信送受信シーケンス設定」ファンクションブロックに、「送信／受信処理の要否設定」を行えるようにしています。

「送信のみ」を設定すると、「レスポンス受信処理」を実行しません。また、「受信のみ」を設定すると、「コマンド送信処理」を実行しません。

9.1.2. ソケットサービス用命令によるTCPソケット通信

TCP ソケットサービス用ファンクションブロック（以下、ソケットサービス用命令）による TCP ソケット通信と送受信メッセージの一般的な動きについての概要を説明します。

参考

詳しくは、「NJ シリーズ コマンドリファレンスマニュアル 基本編」(SBCA-360)の「第 2 章 各命令の説明 通信命令」を参照してください。

●ソケットサービス用命令による TCP ソケットサービス

本プロジェクトファイルでは、以下の 5 種類の標準搭載命令を使用して、ソケット通信を実現しています。

名称	ファンクションブロック	説明
TCP ソケットコネクト	SktTCPConnect	相手機器の TCP ポートに Active オープンで接続します。
TCP ソケット送信	SktTCPSend	指定した TCP ソケットからデータを送信します。
TCP ソケット受信	SktTCPRcv	指定した TCP ソケットから受信したデータを読み出します。
TCP/UDP ソケットクローズ	SktClose	指定した TCP ソケットをクローズします。
TCP ソケットのステータス読出	SktGetTCPStatus	指定した TCP ソケットのステータスを読み出します。 本プロジェクトファイルでは、受信処理時受信完了の確認に使用し、また、クローズ処理時クローズ状態の確認に使用しています。

※TCP ソケットコネクト命令(SktTCPConnect：SktTCPConnect_instance)で取得した Socket を他のソケットサービス用命令の入力パラメータとして使用します。「Socket」のデータ型の構造体_sSOCKET の仕様は、以下のとおりです。

変数	名称	内容	データ型	有効範囲	初期値
Socket	ソケット	ソケット	_sSOCKET	-	-
Handle	ハンドル	データ送受信のハンドル	UDINT	データ型に従う	-
SrcAdr	自アドレス	自身のアドレス※1	_sSOCKET_ADDRESS	-	-
PortNo	ポート番号	ポート番号	UINT	1～65535	
IpAdr	IP アドレス	IP アドレスまたはホスト名※2	STRING	データ型に従う	
DstAdr	相手アドレス	相手のアドレス※1	_sSOCKET_ADDRESS	-	-
PortNo	ポート番号	ポート番号	UINT	1～65535	
IpAdr	IP アドレス	IP アドレスまたはホスト名※2	STRING	データ型に従う	

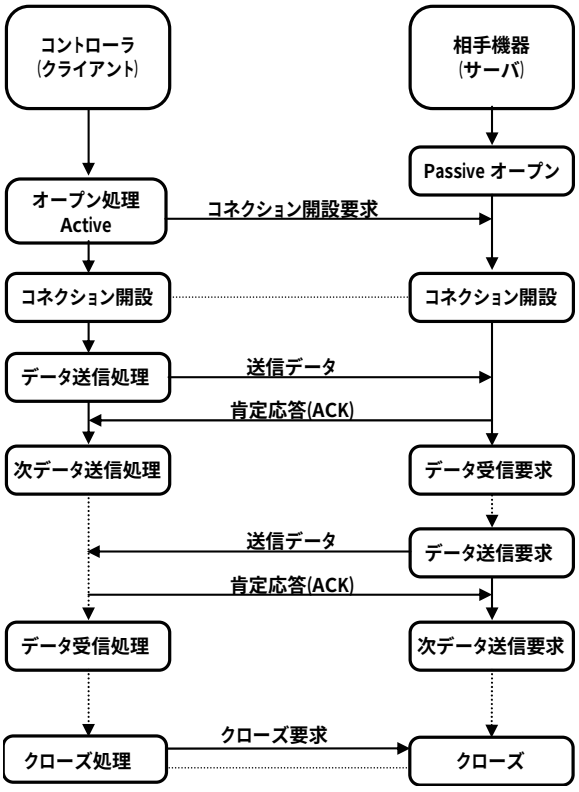
※1：アドレスは、IP アドレスとポート番号のことを指します。

※2：ホスト名の場合は、DNS 設定または Hosts 設定が必要。

●送受信メッセージ

●送受信シーケンス

相手機器（サーバ）とコントローラ（クライアント）間で TCP **による通信を行う場合は、次のような手順で処理が進行します。**



9.2. 相手機器コマンド

本プロジェクトファイルの相手機器コマンドについて説明します。

9.2.1. コマンドの概要

本プロジェクトファイルでは、「201H（バージョンコード照会）」コマンドを利用し、相手機器と Ethernet 通信を行います。

コマンド	内容
201H	バージョンコード照会

参考

詳しくは「株式会社アイエイアイ X-SEL シリアル通信仕様書（フォーマット B）」の「4.伝文ディテール」を参照してください。

9.2.2. コマンドの詳細説明

「201H（バージョンコード照会）」コマンドについて説明します。

●送信メッセージのコマンドフォーマット

「201H（バージョンコード照会）」コマンドの設定に従って、コントローラから相手機器に送信されるメッセージのコマンドフォーマットです。

・フッタ以外は ASCII コードを送信します。

・チェックサムの計算方法は、「9.2.3 コマンドの設定内容」を参照してください。

データ名称	バイト数	備考
ヘッダ	1	固定：”!”
局	2	”99” 相手機器の局番（初期値）
伝文 ID（コマンド）	3	固定：”201”
ユニット種別	2	“00”（0＝メイン CPU アプリ部／1＝メイン CPU コア部／2＝ドライバ CPU／3＝マウント SIO（シリアル I/O））
デバイス No.	1	“0” デバイスを指定する No.
SC（チェックサム）	2	“B6”
フッタ	2	固定：[CR][LF](#16#0D0A)

●受信メッセージのコマンドフォーマット

「201H（バージョンコード照会）」コマンドの設定に従って、コントローラで受信する相手機器からの「正常メッセージ」のレスポンスフォーマットです。

・フッタ以外は ASCII コードで受信します。

・チェックサムの計算方法は、「9.2.3 コマンドの設定内容」を参照してください。

データ名称	バイト数	備考
ヘッダ	1	固定：”#”
局	2	“99” 相手機器の局番（初期値）
伝文 ID（コマンド）	3	固定：“201”
ユニット種別	2	“00”（0＝メイン CPU アプリ部／1＝メイン CPU コア部／2＝ドライバ CPU／3＝マウント SIO（シリアル I/O））
デバイス No.	1	“0” デバイスを指定する No.
機種コード	2	“C2” (X-SEL-PX/QX)
ユニットコード	2	“71” (FROM16M 版)
バージョン No.	4	“001C” (V0.28)
時刻（年）	4	“07D6” （2006 年）
時刻（月）	2	“0C” （12 月）
時刻（日）	2	“1B” （27 日）
時刻（時）	2	“0E” （14 時）
時刻（分）	2	“37” （55 分）
時刻（秒）	2	“00” （00 秒）
SC（チェックサム）	2	“6F”
フッタ	2	固定：[CR][LF](#16#0D0A)

【機種コード・ユニットコード】

機種		機種コード	ユニットコード
X-SEL-J/K		B8	-
X-SEL-JX/KX		C0	-
X-SEL-P/Q	FROM16MB 版	BA	71
	FROM32MB 版		72
X-SEL-PX/QX	FROM16MB 版	C2	71
	FROM32MB 版		72

9.2.3. コマンドの設定内容

「201H（バージョンコード照会）」コマンドの設定内容の詳細について説明します。

●送信データ（コマンド）の設定内容

送信データは、ファンクションブロック：SendMessageSet_instance によって設定します。

変数	内容（データ形式）	設定値
Send_Header	送信ヘッダ(STRING[5])	'!'
Send_Addr	送信アドレス(STRING[5])	'99'
Send_Command	送信データ(STRING[256])	CONCAT('201','00','0')
Send_Check	送信チェックの付加(STRING[5])	'B6'(2 文字)
Send_Terminate	送信ターミネータ(STRING[5])	'RL' ([CR]＋[LF]：#16#0D0A)

変数	内容（データ形式）	データ	説明
Send_Data	送信メッセージ (STRING[256])	CONCAT(Send_Header, Send_Addr, Send_Command, Send_Check, Send_Terminate)	SktTCPSend 命令 (SktTCPSend_instance) の送信データとして使用します。

【SC（チェックサム）の計算方法】

チェックサムはヘッダからチェックサム手前までのオクテット値を全加算し、下位 1 バイトを ASCII コードに変換したものです。（本プロジェクトファイルでは StringSum 命令 (Function)を使用しています。）チェックサムとして"@@"を入力した場合は相手機器のチェックサムを無効にすることができます。

・計算方法

StringSum 命令にてヘッダから送信データまでの対象文字列のサム値（各文字の文字コードの総和）を算出して、2 バイト ASCII 文字出力します。

```
Send_Check := StringSum(CONCAT(Send_Header, Send_Addr, Send_Command),
USINT#2);
```

参考

StringSum 命令について、詳しくは「NJ シリーズ コマンドリファレンスマニュアル 基本編」(SBCD-360)の「第 2 章 各命令の説明 FCS 命令」を参照してください。

●受信データ（レスポンス）の格納内容

受信データは、ファンクションブロック：ReceiveCheck_instance によってデータチェック後に出力受信データとして格納されます。

変数	内容（データ形式）	格納エリア説明
Recv_Data	受信データ(STRING[256])	受信バッファ
Recv_Buff	受信データ(STRING[256])	受信データの格納エリア（受信バッファのデータをそのまま、格納します。）

【SC（チェックサム）の計算方法】

チェックサムはヘッダからチェックサム手前までのオクテット値を全加算し、下位 1 バイトを ASCII コードに変換したものです。（本プロジェクトファイルでは StringSum 命令(Function)を使用しています。）

・チェックサムの抜き出し

受信データの右端から 4 文字目を先頭に 2 文字抜き出します。

Receive_Check:= MID(Recv_Buff, UINT#2, (tLength-UINT#3));

・計算方法

受信データ（バッファ）からヘッダ「#」～「Check SUM＋CR＋LF」の直前のデータを抽出して、受信データの Check SUM を StringSum 命令で計算します。

Calc_Check:=StringSum(Left(Recv_Buff, (tLength - UINT#4)), USINT#2);;

●送受信メッセージ

※送信メッセージ

21	39	39	32	30	31	30	30	30	42	36	0D	0A	…
'!'	'9'	'9'	'2'	'0'	'1'	'0'	'0'	'0'	'B'	'6'	[CR]	[LF]	

※受信メッセージ 1（正常処理時）

23	39	39	32	30	31	30	30	30	xx	xx	xx	xx
'#'	'9'	'9'	'2'	'0 '	'1 '	'0'	'0'	'0'	機種コード		ユニットコード	

xx	xx	xx	xx	xx	xx	…	xx	xx	0D	0A	
バージョン No.				時刻（年）		…	SC		[CR]	[LF]	

※受信メッセージ 2（異常処理時）

26	39	39	xx	xx	xx	xx	xx	0D	0A
'&'	'9'	'9'	エラーコード			SC		[CR]	[LF]

参考

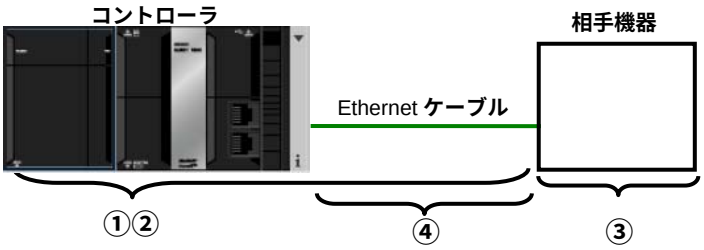
エラーコードの詳細は「株式会社アイエイアイ X-SEL コントローラ PX/QX タイプ取扱説明書」(MJ0152)の「付録」－「エラー表」を参照してください。

9.3. 異常判断処理

本プロジェクトファイルでの異常判断処理について説明します。

9.3.1. プロジェクトファイル内での異常判断

本プロジェクトファイルでは、以下に示す①～④の 4 つの内容について異常判断処理を行っています。エラーコードについては、「9.7.1.エラーコード一覧」を参照してください。



①ソケットサービス用命令による TCP ソケット通信の通信異常

通信ハード、コマンドフォーマットやパラメータの異常など、プログラム上で検出された TCP ソケット通信実行時の異常を「通信異常」として判定します。判定は、ソケットサービス用命令によるソケットサービス用命令の引数「ErrorID」により行います。

②相手機器との通信時のタイムアウト異常

オープン処理、送信処理、受信処理、クローズ処理が正常に行われず、監視時間内に各処理が完了しなかった場合を「タイムアウト異常」として判定します。判定はプロジェクトファイル内のタイマ監視により行います。プロジェクトファイル内タイマによる時間監視機能については、「9.3.2. 時間監視機能」を参照してください。

③相手機器の異常（相手機器異常）

相手機器でのコマンド異常、パラメータ異常、実行不可などの異常を「相手機器異常」として判定します。判定は、相手機器から返送されてくるレスポンスデータにより行います。本プロジェクトファイルでは、異常時に相手機器から返送されるエラーコードにより、相手機器異常を判定します。送受信メッセージについては、「9.2.2. コマンドの詳細説明」を参照してください。

正常メッセージ

‘#’	‘99’	‘201’	*…*	#0D0A
ヘッダ	局	伝文 ID（コマンド）	レスポンスデータ	フッタ

異常メッセージ

‘&’	‘99’	****	xx	#0D0A
ヘッダ	局	エラーコード	SC	フッタ

④処理終了時の TCP コネクション状態異常

本プロジェクトファイルでは、オープン処理から受信処理までの正常終了／異常終了にかかわらず、最後にクローズ処理を行ってから全処理を終了する手順としています。したがって、クローズ処理が正常に終了したかどうかを SktGetTCPStatus 命令の TCP コネクションステータス変数「TcpStatus」で判定します。クローズ処理に異常がある場合、次回のオープン処理を正しく行うことができない場合があります。TCP コネクション状態異常の対処については、「9.7.2. TCP コネクション状態異常の状況と対処方法」を参照してください。

9.3.2. 時間監視機能

本プロジェクトファイルでの時間監視機能について説明します。
監視時間の設定は、ファンクションブロック「ParameterSet」の変数により変更できます。

●プロジェクトファイル内タイマによる時間監視機能

本プロジェクトファイルでは、何らかの異常により処理が実行中のまま終了しない状態を想定し、プロジェクトファイル内のタイマにより処理の中断（タイムアウト）を可能にしています。タイムアウト値はオープンからクローズまでの各処理ともに 5 秒（初期値）としています。

【プロジェクトファイル内タイマによる時間監視機能】

処理内容	監視内容	変数名	タイムアウト値（初期値）
オープン処理	オープン処理開始から終了までの時間	TopenTime	5 秒後 (UINT#500)
送信処理	送信処理開始から終了までの時間	TfsTime	5 秒後 (UINT#500)
受信処理	受信処理開始から終了までの時間 ※受信処理が繰り返される場合は、受信処理ごとの監視となります。	TfrTime	5 秒後 (UINT#500)
クローズ処理	クローズ処理開始から終了までの時間 ※クローズ処理後の TCP コネクション状態が正常であることを確認し、処理終了と判断しています。	TcloseTime	5 秒後 (UINT#500)

●内蔵 EtherNet/IP ポート（ソケットサービス）による時間監視機能

内蔵 EtherNet/IP ポートには、ソケットサービスとして分割されて到着する受信データの時間監視機能があります。受信処理時にソケットサービス用命令[SktTCPRcv_instance]のパラメータ「TimeOut」に「TrTime=UINT#3(300ms)」（初期値）を格納します。また、本プロジェクトファイルでは、1 回受信を終了したあとの次レスポンスの受信待ち時間についても同じく変数「TrTime」を受信待ち時間監視タイマに設定します。この時間内に相手機器から次のレスポンスが到着しなかった場合、受信処理が終了したと判定しています。

参考

ソケットサービスによる時間監視機能については「NJ シリーズ コマンドリファレンスマニュアル 基本編」(SBCA-360) の「第 2 章 各命令の説明 通信命令」の「SktTCPRcv 命令」を参照してください。

●内蔵 EtherNet/IP ポート(TCP/IP)による再送／時間監視機能

通信障害が発生した場合、内蔵 EtherNet/IP ポートに異常がなければ TCP/IP が自動的にデータの再送および処理の時間監視を行います。本プロジェクトファイルでは処理の途中で異常終了した場合、クローズ処理により TCP/IP 再送／時間監視機能を停止します。しかし、クローズ処理が「TCP コネクション状態異常」を示した場合は、継続して内蔵 EtherNet/IP ポート内の TCP/IP 再送／時間監視機能が動作している場合があります。その状況および対処方法については、「9.7.2. TCP コネクション状態異常の状況と対処方法」を参照してください。

9.4. 使用変数

本プロジェクトファイルで使用している変数です。

9.4.1. 使用変数一覧

本プロジェクトファイルの実行にあたって必要な変数一覧です。

●入力変数

本プロジェクトファイルを操作する変数です。

変数名	データ型	説明
Input_Start	BOOL	OFF(FALSE)→ON(TRUE)で本プロジェクトファイルが起動します。正常または異常終了出力確認後に ON→OFF にします。

●出力変数

本プロジェクトファイルの実行結果が反映される変数です。

変数名	データ型	説明
Output_RecvMess	STRING[256]	受信データ（レスポンス）が格納されます。(256 バイト分のエリアを確保しています)
Output_ErrCode	WORD	オープン処理、送信処理、受信処理、クローズ処理時に検出した通信異常、タイムアウト異常のエラー結果（フラグ）が格納されます。 正常終了時には「#0000」が格納されます。
Output_SktCmdsErrorID	WORD	オープン処理、送信処理、受信処理に検出した通信異常、タイムアウト異常の各ソケットサービス用命令のエラーコードが格納されます。 正常終了時には「#0000」が格納されます。
Output_SkTcloseErrorID	WORD	オープン処理、送信処理、受信処理の異常とは別にクローズ処理時に検出した通信異常、タイムアウト異常の SktTcpClose 命令のエラーコードが格納されます。 正常終了時には「#0000」が格納されます。
Output_EtnTcpSta	_eCONNECTION_STATE	クローズ処理時に検出した通信異常、タイムアウト異常時の TCP コネクション状態が格納されます。 正常終了時には「_CLOSED」が格納されます。
Output_MErrCode	DWORD	受信処理時の結果、FCS 計算異常または相手機器異常を検出した場合の異常コードが格納されます。 正常終了時には「#00000000」が格納されます。

●内部変数

本プロジェクトファイルの演算のみに使用する変数です。

変数名	データ型	説明
Local_Status	sStatus (STRUCT)	プログラムの実行状態
Busy	BOOL	本プロジェクトファイル実行中に TRUE となり、非実行時に FALSE となります。
Done	BOOL	本プロジェクトファイル正常終了時に TRUE となり、[Input_Start]が TRUE→FALSE 時に FALSE となります。
Error	BOOL	本プロジェクトファイル異常終了時に TRUE となり、[Input_Start]が TRUE→FALSE 時に FALSE となります。
Local_State	DINT	状態処理番号
Local_ErrCode	uErrorFlgs (UNION)	エラーコードを設定します。
Local_ErrCode. WordData	WORD	エラーコードを WORD であらわします。
Local_ErrCode. BoolData	ARRAY[0..15] OF BOOL	・通信異常 BoolData[0]：送信処理　　：異常(TRUE)/正常(FALSE) BoolData[1]：受信処理　　：異常(TRUE)/正常(FALSE) BoolData[2]：オープン処理：異常(TRUE)/正常(FALSE) BoolData[3]：クローズ処理：異常(TRUE)/正常(FALSE) BoolData[4]：処理番号異常：異常(TRUE)/正常(FALSE) ・タイムアウト異常 BoolData[8]：送信処理　　：異常(TRUE)/正常(FALSE) BoolData[9]：受信処理　　：異常(TRUE)/正常(FALSE) BoolData[10]：オープン処理：異常(TRUE)/正常(FALSE) BoolData[11]：クローズ処理：異常(TRUE)/正常(FALSE) ・その他 BoolData[5]：送信・受信要否判定異常： 異常(TRUE)/正常(FALSE) BoolData[12]：相手機器異常エラー： 異常(TRUE)/正常(FALSE) BoolData[6..7],[13..14]：予約 BoolData[15]：エラー発生
Local_ExecFlgs	sControl (STRUCT)	ソケットサービス用命令起動フラグ
Send	BOOL	送信処理命令 実行(TRUE)/非実行(FALSE)
Recv	BOOL	受信処理命令 実行(TRUE)/非実行(FALSE)
Open	BOOL	オープン処理命令 実行(TRUE)/非実行(FALSE)
Close	BOOL	クローズ処理命令 実行(TRUE)/非実行(FALSE)
Status	BOOL	TCP ステータス命令 実行(TRUE)/非実行(FALSE)
Local_SrcDataByte	UINT	送信データのバイト数をセットします。
Local_SrcData	ARRAY[0..2000] OF BYTE	SktTCPSend 命令(SktTCPSend_instance)の送信データ格納エリアです。(256 バイト分のエリアを確保しています)

変数名	データ型	説明
Local_RecvData	ARRAY[0..2000] OF BOOL	SktTCPRcv 命令(SktTCPRcv_instance)受信データ（レスポンス）が格納されます。(256 バイト分のエリアを確保しています)
Local_ReceiveMessage	STRING[256]	Local_RecvData 受信 STRING データ（レスポンス）が格納されます。(256 文字のエリアを確保しています)
Local_RecvCheckFlg	BOOL	相手機器異常判定命令起動フラグ 実行(TRUE)/非実行(FALSE)
Local_InitialSettingOK	BOOL	初期処理正常設定フラグ
Local_TONFlgs	sTimerControl (STRUCT)	タイマ実行フラグ
Tfs	BOOL	送信処理時間監視タイマ命令 実行(TRUE)/非実行(FALSE)
Tfr	BOOL	受信処理時間監視タイマ命令 実行(TRUE)/非実行(FALSE)
Topen	BOOL	オープン処理時間監視タイマ命令 実行(TRUE)/非実行(FALSE)
Tclose	BOOL	クローズ処理時間監視タイマ命令 実行(TRUE)/非実行(FALSE)
Tr	BOOL	次レスポンス受信待ち時間監視タイマ命令 実行(TRUE)/非実行(FALSE)
Local_ComType	sControl (STRUCT)	送信・受信処理の要否を設定します。
Send	BOOL	送信処理 必要(TRUE)/不要(FALSE)。 ※送信処理 必要／受信処理が不要な場合： 送信処理時に受信データの到着を待たず、受信処理をスキップし、クローズ処理に遷移します。コマンド送信に対してレスポンスデータが送られてこない場合に指定します。
Recv	BOOL	受信処理 必要(TRUE)/不要(FALSE)。 ※送信処理 必要／受信処理が必要な場合： 送信処理後に受信データの到着を待ちます。受信データの到着確認後、受信処理に遷移します。コマンド送信に対してレスポンスデータが送られてくる場合に指定します。
Error	BOOL	送受信処理要否設定エラーフラグ（設定異常時にフラグをセットします。）

●ソケットサービス用命令の初期化に使用する変数

変数名	データ型	説明
NULL_SOCKET	_sSOCKET	内部_ソケットサービス用命令初期化データ（保持・定数：有効） 初期値(Handle := 0, SrcAdr := (PortNo := 0, IpAdr := ''), DstAdr := (PortNo := 0, IpAdr := '')) （全ソケット命令に使用します。）
NULL_ARRAYOFBYTE_1	ARRAY[0..0] OF BYTE	内部_送信ソケットサービス用命令初期化データ配列（保持・定数：有効） 初期値[0]（SktTCPSend 命令に使用します。）
NULL_ARRAYOFBYTE_2	ARRAY[0..0] OF BYTE	内部_受信ソケットサービス用命令初期化データ配列（保持・定数：無効） 初期値[0]（SktTCPRcv 命令に使用します。）

9.4.2. 使用ファンクションブロック/ファンクションの使用変数一覧

本プロジェクトファイルの実行にあたってプログラム中でユーザ定義となるファンクションブロックの一覧です。

下記ファンクションブロックの変数については「9.5.3. ファンクションブロックの詳細説明」を参照してください。

変数名	データ型	説明
ETN_ParameterSet_instance	ParameterSet	Ethernet 設定(相手先 IP アドレスなど) オープンからクローズまでの各処理の監視時間
ETN_SendMessageSet_instance	SendMessageSet	送信・受信処理の要否と送信メッセージの設定を行います。
ETN_ReceiveCheck_instance	ReceiveCheck	受信データの格納および正常・異常を判定します。

●タイマ

本プロジェクトファイルで使用するタイマです。

変数名	データ型	説明
Topen_TON_instance	TON	オープン処理の時間計測を行います。
Tfs_TON_instance	TON	送信処理の時間計測を行います。
Tfr_TON_instance	TON	受信処理の時間計測を行います。
Tclose_TON_instance	TON	クローズ処理の時間計測を行います。
Tr_TON_instance	TON	次レスポンス受信待ち処理時間計測を行います。

9.4.3. システム変数一覧

本プロジェクトファイルの実行にあたって必要な変数一覧です。

●システム変数（外部変数）

変数名	データ型	説明
_EIP_EtnOnlineSta	BOOL	内蔵 EtherNet/IP ポートの通信機能状態 TRUE：利用可能 /FALSE：利用不可

参考

システム変数および通信命令については、「NJ シリーズ コマンドリファレンスマニュアル 基本編」(SBCA-360)の「第 2 章 各命令の説明 通信命令」を参照してください。

9.5. プログラム（ST言語）

9.5.1. ST言語によるプログラムの機能構成

本プロジェクトファイルは ST 言語でプログラミングされています。その機能構成は、以下のとおりです。

大分類	小分類	内容
1.通信処理	1.1.通信処理開始 1.2.通信処理ステータスフラグ列クリア 1.3.通信処理実行中状態	通信処理を起動します。
2.初期処理	2.1.処理時間監視タイマの初期化 2.2.ソケットサービス用命令の初期化 2.3.ソケットサービス用命令起動フラグの初期化 2.4.処理時間監視タイマ実行フラグの初期化 2.5.エラーコード格納エリアの初期化 2.6.各処理監視時間設定および Ethernet 関連パラメータ設定 2.7.送受信処理要否の設定および送信データの設定 2.8.送信データを STRING 形式からバイト配列に変換 2.9.受信データ格納エリアの初期化 2.10.初期設定終了処理	Ethernet のパラメータ設定および異常コード格納エリアの初期設定を行います。 送信および受信の要否、送信データ、受信データの設定を行います。
3.オープン処理	3.1.オープン処理状況の判定と起動フラグセット 3.2.オープン処理時間監視タイマ実行 3.3.オープン命令起動(TCP.Active オープン処理)	TCP オープン(Active)処理を行います。 通信処理の起動、初期設定後に無条件に処理を開始します。
4.送信処理	4.1.送信処理状況の判定と起動フラグセット 4.2.送信処理時間監視タイマ実行 4.3.送信命令起動	送信処理要否が「必要」設定であり、かつオープン処理が正常に終了した場合に処理を開始します。
5.受信処理	5.1.受信処理状況の判定と起動フラグセット 5.2.受信待ち時間監視タイマ実行 5.3.受信処理時間監視タイマ実行 5.4.受信命令起動 5.5.TCP ステータス取得処理起動 5.6.相手機器異常判定命令起動	受信処理要否が「必要」設定であり、かつ送信処理が正常に終了した場合に処理を開始します。 受信データが複数到着する場合には、受信処理を繰り返します。 受信データの格納と受信データのチェックを行います。
6.クローズ処理	6.1.クローズ処理状況の判定と起動フラグセット 6.2.クローズ処理時間監視タイマ実行 6.3.クローズ命令起動 6.4.TCP ステータス取得処理起動	クローズ処理を行います。 以下の場合に処理を開始します。 ・受信処理要否が「不要」設定であり、かつ送信処理が正常に終了した場合 ・受信処理が正常に終了した場合 ・オープン処理、送信処理、受信処理のいずれかが異常終了した直後
7.処理異常処理	7.処理番号異常処理	存在しない処理番号を検出した場合の異常処理を実行します。

9.5.2. メインプログラムの詳細説明

以下に、本プロジェクトファイルを掲載します。
相手機器により変更が必要な通信設定や送信データ（コマンド）設定、受信データ（レスポンスデータ）確認は、ファンクションブロック(ETN_ParameterSet_instance, ETN_SendMessageSet_instance, ETN_ReceiveCheck_instance)**内で実施しています。これらの値を変更したい場合は、「**9.5.3 **ファンクションブロックの詳細説明」を参照してください。**

【メインプログラム:Program0】

1.**通信処理**

```
(* ================================================================================ *)
(* 名称：NJｼﾘｰｽﾞ汎用Ethernet通信ﾌﾟﾛｸﾞﾗﾑ                                             *)
(* 機能：汎用Ethernet通信ﾒｲﾝﾌﾟﾛｸﾞﾗﾑ                                                 *)
(* Ethernetﾕﾆｯﾄ：NJ501 (内蔵EtherNet/IPﾎﾟｰﾄ)                                         *)
(* 備考：                                                                           *)
(*                                                                                  *)
(* ﾊﾞｰｼﾞｮﾝ情報：2011/08/01 V1.00 新規                                               *)
(*                                                                                  *)
(* (C)Copyright OMRON Corporation 2011 All Rights Reserved.                         *)
(* ================================================================================ *)

(* 1. 通信処理 *)
(* 変数説明：通信処理 制御用 ========================================================  )
(                                                                                   )
(  入力起動ﾌﾗｸﾞ          : Input_Start                                             )
(                                                                                   )
(  通信処理ｽﾃｰﾀｽﾌﾗｸﾞ列   : Local_Status<STRUCT>                                      )
(                                  ↓                                              )
(    ┣通信処理実行中ﾌﾗｸﾞ    (Busy)   : Local_Status.Busy                            )
(    ┣通信処理正常終了ﾌﾗｸﾞ  (Done)   : Local_Status.Done                            )
(    ┗通信処理異常終了ﾌﾗｸﾞ  (Error)  : Local_Status.Error                           )
(                                                                                   )
(  状態処理番号          : Local_State                                              )
(                                10：初期処理                                       )
(                                11：ｵｰﾌﾟﾝ処理                                      )
(                                12：送信処理                                       )
(                                13：受信処理                                       )
(                                14：ｸﾛｰｽﾞ処理                                      )
(                                99：処理番号異常処理                               )
(                                                                                   )
( ================================================================================ *)

(* 1.1. 通信処理開始
    通信処理ｽﾃｰﾀｽﾌﾗｸﾞ列がｸﾘｱな状態で入力起動ﾌﾗｸﾞがONされた場合に通信処理を開始*)
IF Input_Start AND
       NOT(Local_Status.Busy OR Local_Status.Done OR Local_Status.Error) THEN
    Local_Status.Busy:=TRUE;
    Local_State:=10;                                        //10:初期処理へ
END_IF;

(* 1.2. 通信処理ｽﾃｰﾀｽﾌﾗｸﾞ列ｸﾘｱ
    通信処理非実行状態で入力起動ﾌﾗｸﾞOFFにより通信処理ｽﾃｰﾀｽﾌﾗｸﾞ列ｸﾘｱ *)
IF NOT(Local_Status.Busy) AND NOT(Input_Start) THEN
    Local_Status.Done:=FALSE;
    Local_Status.Error:=FALSE;
END_IF;

(* 1.3. 通信処理実行中状態
    状態処理番号(Local_State)に応じた処理を実行 *)
IF Local_Status.Busy THEN
    CASE Local_State OF
```

2.初期処理

```
10: (* ================================================================== *)
    (* 2. 初期処理                                                        *)
    (*   ・通信全体の各種初期化とﾊﾟﾗﾒｰﾀ設定                               *)
    (*   ・送信ﾃﾞｰﾀの設定と受信ﾃﾞｰﾀ格納ｴﾘｱの初期化                       *)
    (* ================================================================== *)

    (* 2.1. 処理時間監視ﾀｲﾏの初期化 *)
    Topen_TON_instance (In:=FALSE,PT:=TIME#0ms);
    Tfs_TON_instance   (In:=FALSE,PT:=TIME#0ms);
    Tr_TON_instance    (In:=FALSE,PT:=TIME#0ms);
    Tfr_TON_instance   (In:=FALSE,PT:=TIME#0ms);
    Tclose_TON_instance(In:=FALSE,PT:=TIME#0ms);

    (* 2.2. ｿｹｯﾄｻｰﾋﾞｽ用命令の初期化 *)
    SktTCPConnect_instance(
        Execute:=FALSE,SrcTcpPort:=UINT#0,DstTcpPort:=UINT#0,DstAdr:='');
    SktTCPSend_instance(
        Execute:=FALSE,Socket:=NULL_SOCKET,Size:=UINT#0,
        SendDat:=NULL_ARRAYOFBYTE_1[0]);
    SktTCPRcv_instance(
        Execute:=FALSE,Socket:=NULL_SOCKET,Size:=UINT#0,TimeOut:=UINT#0,
        RcvDat:=NULL_ARRAYOFBYTE_2[0]);
    SkTclose_instance(
        Execute:=FALSE,Socket:=NULL_SOCKET);
    SktGetTCPStatus_instance(
        Execute:=FALSE,Socket:=NULL_SOCKET);

    (* 2.3. ｿｹｯﾄｻｰﾋﾞｽ用命令起動ﾌﾗｸﾞの初期化 *)
    (* 変数説明：ｿｹｯﾄｻｰﾋﾞｽ用命令起動ﾌﾗｸﾞ(Executeﾊﾟﾗﾒｰﾀ用) ================ )
    (                                                                    )
    (  ｿｹｯﾄｻｰﾋﾞｽ用命令起動ﾌﾗｸﾞ列：Local_ExecFlgs<STRUCT>                  )
    (    |                                          ↓                   )
    (    ├送信命令起動ﾌﾗｸﾞ   (SktTCPSend)      :Local_ExecFlgs.Send       )
    (    ├受信命令起動ﾌﾗｸﾞ   (SktTCPRcv)       :Local_ExecFlgs.Recv       )
    (    ├ｵｰﾌﾟﾝ命令起動ﾌﾗｸﾞ  (SktTCPConnect)   :Local_ExecFlgs.Open       )
    (    ├ｸﾛｰｽﾞ命令起動ﾌﾗｸﾞ  (SkTclose)        :Local_ExecFlgs.Close      )
    (    └TCPｽﾃｰﾀｽ取得命令起動ﾌﾗｸﾞ                                      )
    (                        (SktGetTCPStatus):Local_ExecFlgs.Status     )
    ( ================================================================== *)
    Local_ExecFlgs.Send:=FALSE;
    Local_ExecFlgs.Recv:=FALSE;
    Local_ExecFlgs.Open:=FALSE;
    Local_ExecFlgs.Close:=FALSE;
    Local_ExecFlgs.Status:=FALSE;

    (* 2.4. 処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞの初期化 *)
    (* 変数説明：処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ(Inﾊﾟﾗﾒｰﾀ用) ================== )
    (                                                                    )
    (  処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ列：Local_TONFlgs<STRUCT>                  )
    (    |                                          ↓                   )
    (    ├送信処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ  (Tfs_TON) :Local_TONFlgs.Tfs      )
    (    ├受信処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ  (Tfr_TON) :Local_TONFlgs.Tfr      )
    (    ├ｵｰﾌﾟﾝ処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ(Topen_TON)                       )
    (    |                                        :Local_TONFlgs.Topen    )
    (    ├ｸﾛｰｽﾞ処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ(Tclose_TON)                      )
    (    |                                        :Local_TONFlgs.Tclose   )
    (    └受信待ち時間監視ﾀｲﾏ実行ﾌﾗｸﾞ  (Tr_TON)                          )
    (        (次ﾒｯｾｰｼﾞ待ち時間)                  :Local_TONFlgs.Tr       )
    (                                                                    )
    ( ================================================================== *)
    Local_TONflgs.Tfs:=FALSE;
    Local_TONflgs.Tfr:=FALSE;
    Local_TONflgs.Topen:=FALSE;
    Local_TONflgs.Tclose:=FALSE;
    Local_TONflgs.Tr:=FALSE;

   (* 2.5. ｴﾗｰｺｰﾄﾞ格納ｴﾘｱの初期化 *)
    Local_ErrCode.WordData:=WORD#16#0000;
    Output_ErrCode:=WORD#16#FFFF;
    Output_MErrCode:=DWORD#16#FFFFFFFF;
    Output_SktCmdsErrorID:=WORD#16#FFFF;
    Output_SkTcloseErrorID:=WORD#16#FFFF;
```

```
(* 2.6. 各処理監視時間設定およびEthernet関連ﾊﾟﾗﾒｰﾀ設定 *)
ETN_ParameterSet_instance(
    Execute:=TRUE);

(* 2.7. 送受信処理要否の設定および送信ﾃﾞｰﾀの設定 *)
ETN_SendMessageSet_instance(
    Execute:=TRUE);
    (* 送受信処理要否の設定異常判定 *)
(* ＜変数メモ＞
      ＞ Local_ComType.Send：送信処理要否ﾌﾗｸﾞ
      ＞ Local_ComType.Recv：受信処理要否ﾌﾗｸﾞ
      ＞ Local_ComType.Error：送受信処理要否設定ｴﾗｰ *)
Local_ComType.Send:=TestABit(ETN_SendMessageSet_instance.ComType,0);
Local_ComType.Recv:=TestABit(ETN_SendMessageSet_instance.ComType,1);
Local_ComType.Error:=NOT(Local_ComType.Send OR Local_ComType.Recv);
IF Local_ComType.Error THEN
    Output_ErrCode:=WORD#16#0020;
    Local_InitialSettingOK:=FALSE;
ELSE
    Local_InitialSettingOK:=TRUE;
END_IF;

(* 2.8. 送信ﾃﾞｰﾀをString形式からﾊﾞｲﾄ配列に変換 *)
Local_SrcDataByte:=
    StringToAry(ETN_SendMessageSet_instance.Send_Data,Local_SrcData[0]);
(* 2.9. 受信ﾃﾞｰﾀ格納ｴﾘｱの初期化 *)
ClearString(Local_ReceiveMessage);
ClearString(Output_RecvMess);
Local_RecvCHNo:=0;
Local_RecvDataLength:=0;
Local_ReceiveSize:=UINT#256;

(* 2.10. 初期設定終了処理 *)
IF Local_InitialSettingOK THEN
    Local_State:=11;                                    //11:ｵｰﾌﾟﾝ処理へ
ELSE
    Local_Status.Busy:=FALSE;
    Local_Status.Error:=TRUE;

    Local_State:=0;                                     //0:通信非実行状態へ
END_IF;
```

3.オープン処理

```
11: (* ================================================================== *)
    (* 3. ｵｰﾌﾟﾝ処理                                                       *)
    (*  ・相手TCPﾎﾟｰﾄにActive接続                                         *)
    (* ================================================================== *)
    (* ＜変数メモ＞
       ＞ Local_ExecFlgs.Open：ｵｰﾌﾟﾝ命令起動ﾌﾗｸﾞ
       ＞ Local_TONFlgs.Topen ：ｵｰﾌﾟﾝ処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ *)

    (* 3.1. ｵｰﾌﾟﾝ処理状況の判定と起動ﾌﾗｸﾞｾｯﾄ *)

        (* 3.1.1. ﾀｲﾑｱｳﾄ処理 *)
    IF Topen_TON_instance.Q THEN
        Local_ErrCode.BoolData[10]:=TRUE;
        Output_SktCmdsErrorID:=WORD#16#FFFF;
        Local_ExecFlgs.Open:=FALSE;
        Local_TONflgs.Topen:=FALSE;
        Local_State:=14;                                    //14:ｸﾛｰｽﾞ処理へ

        (* 3.1.2. 正常終了処理 *)
    ELSIF SktTCPConnect_instance.Done THEN
        Local_ErrCode.BoolData[2]:= FALSE;
        Output_SktCmdsErrorID:=WORD#16#0000;
        Local_ExecFlgs.Open:=FALSE;
        Local_TONflgs.Topen:=FALSE;
    (* ＜変数メモ＞
         ＞ Local_ComType.Send：送信処理要否ﾌﾗｸﾞ
         ＞ Local_ComType.Recv：受信処理要否ﾌﾗｸﾞ *)
        IF Local_ComType.Send THEN
            Local_State:=12;                                //12:送信処理へ
        ELSIF Local_ComType.Recv THEN
            Local_State:=13;                                //13:受信処理へ
        END_IF;
        (* 3.1.3. 異常終了処理 *)
    ELSIF SktTCPConnect_instance.Error THEN
        Local_ErrCode.BoolData[2]:=TRUE;
        Output_SktCmdsErrorID:=SktTCPConnect_instance.ErrorID;
        Local_ExecFlgs.Open:=FALSE;
        Local_TONflgs.Topen:=FALSE;
        Local_State:=14;                                    //14:ｸﾛｰｽﾞ処理へ

        (* 3.1.4. ｵｰﾌﾟﾝ命令起動ﾌﾗｸﾞｾｯﾄ／ﾀｲﾏ実行ﾌﾗｸﾞｾｯﾄ *)
    ELSE
        Local_ExecFlgs.Open:=TRUE;
        Local_TONflgs.Topen:=TRUE;
    END_IF;

    (* 3.2. ｵｰﾌﾟﾝ処理時間監視ﾀｲﾏ実行 *)
    Topen_TON_instance(
        IN:=Local_TONflgs.Topen,
        PT:=MULTIME(TIME#10ms,ETN_ParameterSet_instance.TopenTime));

    (* 3.3. ｵｰﾌﾟﾝ命令起動(TCP.Activeｵｰﾌﾟﾝ処理)
        内蔵ETN利用可能(_EIP_EtnOnlineStaがON)状態でｵｰﾌﾟﾝ命令起動 *)
    SktTCPConnect_instance(
        Execute:=Local_ExecFlgs.Open AND _EIP_EtnOnlineSta,
        SrcTcpPort:=ETN_ParameterSet_instance.SrcPort,
        DstTcpPort:=ETN_ParameterSet_instance.DstPort,
        DstAdr:=ETN_ParameterSet_instance.DstIPAddr);
```

4.送信処理

```
12: (* ================================================================== *)
    (* 4. 送信処理                                                        *)
    (*  ・指定したTCPﾎﾟｰﾄからﾃﾞｰﾀを送信                                   *)
    (* ================================================================== *)
    (* ＜変数メモ＞
       ＞ Local_ExecFlgs.Send：送信命令起動ﾌﾗｸﾞ
       ＞ Local_TONFlgs.Tfs  ：送信処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ *)

    (* 4.1. 送信処理状況の判定と起動ﾌﾗｸﾞｾｯﾄ *)

        (* 4.1.1. ﾀｲﾑｱｳﾄ処理 *)
    IF Tfs_TON_instance.Q THEN
        Local_ErrCode.BoolData[8]:=TRUE;
        Output_SktCmdsErrorID:=WORD#16#FFFF;
        Local_ExecFlgs.Send:=FALSE;
        Local_TONflgs.Tfs:=FALSE;
        Local_State:=14;                                    //14:ｸﾛｰｽﾞ処理へ

        (* 4.1.2. 正常終了処理 *)
    ELSIF SktTCPSend_instance.Done THEN
        Local_ErrCode.BoolData[0]:=FALSE;
        Output_SktCmdsErrorID:=WORD#16#0000;
        Local_ExecFlgs.Send:=FALSE;
        Local_TONflgs.Tfs:=FALSE;
        (* ＜変数メモ＞
           ＞ Local_ComType.Recv：受信処理要否ﾌﾗｸﾞ *)
        Local_State:=SEL(Local_ComType.Recv,14,13); //13:受信処理へ
                                                    //14:ｸﾛｰｽﾞ処理へ

        (* 4.1.3. 異常終了処理 *)
    ELSIF SktTCPSend_instance.Error THEN
        Local_ErrCode.BoolData[0]:=TRUE;
        Output_SktCmdsErrorID:=
            SktTCPSend_instance.ErrorID;
        Local_ExecFlgs.Send:=FALSE;
        Local_TONflgs.Tfs:=FALSE;
        Local_State:=14;                                    //14:ｸﾛｰｽﾞ処理へ

        (* 4.1.4. 送信命令起動ﾌﾗｸﾞｾｯﾄ／ﾀｲﾏ実行ﾌﾗｸﾞｾｯﾄ *)
    ELSE
        Local_ExecFlgs.Send:=TRUE;
        Local_TONflgs.Tfs:=TRUE;
    END_IF;

    (* 4.2. 送信処理時間監視ﾀｲﾏ実行 *)
    Tfs_TON_instance(
        In:=Local_TONflgs.Tfs,
        PT:=MULTIME(TIME#10ms, ETN_ParameterSet_instance.TfsTime));

    (* 4.3. 送信命令起動
        内蔵ETN利用可能(_EIP_EtnOnlineStaがON)状態で送信命令起動 *)
    SktTCPSend_instance(
        Execute:=Local_ExecFlgs.Send AND _EIP_EtnOnlineSta,
        Size:=Local_SrcDataByte,
        Socket:=SktTCPConnect_instance.Socket,
        SendDat:=Local_SrcData[0]);
```

5.受信処理

```
13: (* ========================================================================== *)
    (* 5. 受信処理                                                              *)
    (*  ・指定したTCPｿｹｯﾄの受信ﾊﾞｯﾌｧ内のﾃﾞｰﾀを読み出し                          *)
    (* ========================================================================== *)
    (* <変数メモ>
       > Local_ExecFlgs.Recv : 受信命令起動ﾌﾗｸﾞ
       > Local_ExecFlgs.Status : TCPｽﾃｰﾀｽ取得命令起動ﾌﾗｸﾞ
       > Local_TONFlgs.Tfr  : 受信処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ
       > Local_TONFlgs.Tr  : 受信待ち時間監視ﾀｲﾏ実行ﾌﾗｸﾞ
                                (次ﾒｯｾｰｼﾞ待ち時間)                              *)

    (* 5.1. 受信処理状況の判定と起動ﾌﾗｸﾞｾｯﾄ *)

        (* 5.1.1. 受信終了処理 *)
    IF Tr_TON_instance.Q THEN
        Local_ExecFlgs.Status:=FALSE;
        Local_TONflgs.Tfr:=FALSE;
        Local_TONflgs.Tr:=FALSE;

        (* 受信ﾃﾞｰﾀのBYTE配列->STRING型変換 *)
        Local_ReceiveMessage:=
                AryToString(Local_RecvData[0],Local_RecvDataLength);

        (* 相手機器異常判定命令起動ﾌﾗｸﾞｾｯﾄ *)
        Local_RecvCheckFlg:=TRUE;

        Local_State:=14;                                //14:ｸﾛｰｽﾞ処理へ

        (* 5.1.2. ﾀｲﾑｱｳﾄ処理 *)
    ELSIF Tfr_TON_instance.Q THEN
        Local_ErrCode.BoolData[9]:=TRUE;
        Output_SktCmdsErrorID:=WORD#16#FFFF;
        Local_ExecFlgs.Recv:=FALSE;
        Local_ExecFlgs.Status:=FALSE;
        Local_TONflgs.Tfr:=FALSE;
        Local_State:=14;                                //14:ｸﾛｰｽﾞ処理へ

        (* 5.1.3. 正常終了処理 *)
    ELSIF SktTCPRcv_instance.Done THEN
        Local_RecvDataLength
            :=Local_RecvDataLength+SktTCPRcv_instance.RcvSize;
        Local_RecvCHNo:=Local_RecvDataLength;

        Local_ExecFlgs.Recv:=FALSE;
        Local_TONflgs.Tfr:=FALSE;
        Local_TONflgs.Tr:=TRUE;             // 5.1.5. 受信ﾃﾞｰﾀ読み出し処理へ

        (* 5.1.4. 異常終了処理 *)
    ELSIF SktTCPRcv_instance.Error THEN;
            Local_ErrCode.BoolData[1]:=TRUE;
            Output_SktCmdsErrorID:=
                SktTCPRcv_instance.ErrorID;

        Local_ExecFlgs.Recv:=FALSE;
        Local_TONflgs.Tfr:=FALSE;

        Local_State:=14;                                //14:ｸﾛｰｽﾞ処理へ
```

```
        (* 5.1.5.  受信ﾃﾞｰﾀ読み出し処理 *)
    ELSIF SktGetTCPStatus_instance.Done
            OR SktGetTCPStatus_instance.Error THEN
        Local_ExecFlgs.Status:=FALSE;

            (* 読み出すﾃﾞｰﾀがある場合：受信処理を継続 *)
        IF SktGetTCPStatus_instance.DatRcvFlag THEN
            Local_ExecFlgs.Recv:=TRUE;
            Local_TONflgs.Tfr:=TRUE;
            Local_TONflgs.Tr:=FALSE;
        END_IF;
            (* 読み出すﾃﾞｰﾀがない場合：
                ・ﾃﾞｰﾀを全く受信していない場合は何も処理せず、
                  次のｻｲｸﾙでTCPｽﾃｰﾀｽ取得を再起動する
                ・既にﾃﾞｰﾀを受信している場合はﾚｽﾎﾟﾝｽ受信待ち時間を監視して
                  次のﾚｽﾎﾟﾝｽがなくﾀｲﾑｱｳﾄすれば、既に受信済みのﾃﾞｰﾀを
                  読み出して受信処理終了
                *)

        (* 5.1.6.  TCPｽﾃｰﾀｽ取得命令起動ﾌﾗｸﾞｾｯﾄ／ﾀｲﾏ実行ﾌﾗｸﾞｾｯﾄ *)
    ELSE
        Local_ExecFlgs.Status:=TRUE;
        Local_TONflgs.Tfr:=TRUE;

            (* 相手機器異常判定命令起動ﾌﾗｸﾞ初期化 *)
        Local_RecvCheckFlg:=FALSE;
    END_IF;

    (* 5.2.  受信待ち時間監視ﾀｲﾏ実行(次ﾚｽﾎﾟﾝｽ待ち時間) *)
    Tr_TON_instance(
        In:=Local_TONflgs.Tr,

        PT:=MULTIME(TIME#100ms,ETN_ParameterSet_instance.TrTime));

    (* 5.3.  受信処理時間監視ﾀｲﾏ実行 *)
    Tfr_TON_instance(
        In:=Local_TONflgs.Tfr,
        PT:=MULTIME(TIME#10ms,ETN_ParameterSet_instance.TfrTime));

    (* 5.4.  受信命令起動
        内蔵ETN利用可能(_EIP_EtnOnlineStaがON)状態で受信命令起動 *)
    SktTCPRcv_instance(
        Execute:=Local_ExecFlgs.Recv AND _EIP_EtnOnlineSta,
        Socket:=SktTCPConnect_instance.Socket,
        TimeOut:=ETN_ParameterSet_instance.TrTime,
        Size:=Local_ReceiveSize,
        RcvDat:=Local_RecvData[Local_RecvCHNo]);

    (* 5.5.  TCPｽﾃｰﾀｽ取得命令起動
        内蔵ETN利用可能(_EIP_EtnOnlineStaがON)状態でTCPｽﾃｰﾀｽ取得命令起動 *)
    SktGetTCPStatus_instance(
        Execute:=Local_ExecFlgs.Status AND _EIP_EtnOnlineSta,
        Socket:=SktTCPConnect_instance.Socket);

    (* 5.6.  相手機器異常判定命令起動 *)
    ETN_ReceiveCheck_instance(
        Execute:=Local_RecvCheckFlg,
        Recv_Buff:=Local_ReceiveMessage,
        Recv_Data:=Output_RecvMess,
        tLength:=Local_RecvDataLength,
        ErrorID:=Local_ErrCode.WordData,
        ErrorIDEx:=Output_MErrCode);
```

6.クローズ処理

```
14: (* ================================================================== *)
    (* 6.ｸﾛｰｽﾞ処理                                                        *)
    (*  ・指定したｿｹｯﾄをｸﾛｰｽﾞ                                            *)
    (* ================================================================== *)
    (* ＜変数メモ＞
       ＞ Local_ExecFlgs.Close : ｸﾛｰｽﾞ命令起動ﾌﾗｸﾞ
       ＞ Local_ExecFlgs.Staus : TCPｽﾃｰﾀｽ取得命令起動ﾌﾗｸﾞ
       ＞ Local_TONFlgs.Tclose  : ｸﾛｰｽﾞ処理時間監視ﾀｲﾏ実行ﾌﾗｸﾞ *)

    (* 6.1. ｸﾛｰｽﾞ処理状況の判定と起動ﾌﾗｸﾞｾｯﾄ *)

        (* 6.1.1. ﾀｲﾑｱｳﾄ処理 *)
    IF Tclose_TON_instance.Q THEN
        Local_ErrCode.BoolData[11]:=TRUE;
        Output_SkTcloseErrorID:=WORD#16#FFFF;
        Local_ExecFlgs.Close:=FALSE;
        Local_TONflgs.Tclose:=FALSE;
        Local_ExecFlgs.Status:=FALSE;
        Output_EtnTcpSta:=SktGetTCPStatus_instance.TcpStatus;
        Local_ErrCode.BoolData[15]:=TRUE;
        Output_ErrCode:=Local_ErrCode.WordData;
        Local_Status.Busy:=FALSE;
        Local_Status.Error:=TRUE;

        Local_State:=0;                                  //0:通信非実行状態へ

        (* 6.1.2. 正常終了処理 *)
    ELSIF SkTclose_instance.Done THEN
        Local_ExecFlgs.Status:=TRUE;
        IF  SktGetTCPStatus_instance.Done
            OR SktGetTCPStatus_instance.Error THEN
            Local_ExecFlgs.Status:=FALSE;

            IF SktGetTCPStatus_instance.TcpStatus = _CLOSED THEN
                Local_TONflgs.Tclose:=FALSE;
                Output_SkTcloseErrorID:=WORD#16#0000;
                Output_EtnTcpSta:=SktGetTCPStatus_instance.TcpStatus;
                Local_ExecFlgs.Close:=FALSE;

                (* 通信処理全体の処理結果判定 *)
                Local_Status.Busy:=FALSE;

                    (* 通信処理正常終了 *)
                IF Local_ErrCode.WordData = WORD#16#0000 THEN
                    Local_Status.Done:=TRUE;
                    Local_ErrCode.BoolData[15]:=FALSE;

                    (* 通信処理異常終了 *)
                ELSE
                    Local_Status.Error:=TRUE;
                    Local_ErrCode.BoolData[15]:=TRUE;
                END_IF;
                Output_ErrCode:=Local_ErrCode.WordData;

                Local_State:=0;                          //0:通信非実行状態へ

            END_IF;
        END_IF;

        (* 6.1.3. 異常終了処理 *)
    ELSIF SkTclose_instance.Error THEN
        Local_ErrCode.BoolData[3]:=TRUE;
        Output_SkTcloseErrorID:=SkTclose_instance.ErrorID;
        Local_ExecFlgs.Close:=FALSE;
        Local_TONflgs.Tclose:=FALSE;
        Local_ErrCode.BoolData[15]:=TRUE;
        Output_ErrCode:=Local_ErrCode.WordData;
        Local_Status.Busy:=FALSE;
        Local_Status.Error:=TRUE;

        Local_State:=0;                                  //0:通信非実行状態へ
```

```
            (* 6.1.4. ｸﾛｰｽﾞ命令起動ﾌﾗｸﾞｾｯﾄ／ﾀｲﾏ実行ﾌﾗｸﾞｾｯﾄ *)
        ELSE
            Local_ExecFlgs.Close:=TRUE;
            Local_TONflgs.Tclose:=TRUE;

        END_IF;

        (* 6.2. ｸﾛｰｽﾞ処理時間監視ﾀｲﾏ実行 *)
        Tclose_TON_instance(
            In:= Local_TONflgs.Tclose,
            PT:=MULTIME(TIME#10ms,ETN_ParameterSet_instance.TcloseTime));

        (* 6.3. ｸﾛｰｽﾞ命令起動
            内蔵ETN利用可能(_EIP_EtnOnlineStaがON)状態でｸﾛｰｽﾞ命令起動 *)
        SkTclose_instance(
            Execute:=Local_ExecFlgs.Close AND _EIP_EtnOnlineSta,
            Socket:=SktTCPConnect_instance.Socket);

        (* 6.4. TCPｽﾃｰﾀｽ取得命令起動
            内蔵ETN利用可能(_EIP_EtnOnlineStaがON)状態でTCPｽﾃｰﾀｽ取得命令起動 *)
        SktGetTCPStatus_instance(
            Execute:=Local_ExecFlgs.Status AND _EIP_EtnOnlineSta,
            Socket:=SktTCPConnect_instance.Socket);
```

7.処理番号異常処理

```
    99: (* ================================================================== *)
        (* 7. 処理番号異常処理                                                *)
        (*  ・存在しない状態処理番号が設定された場合の異常処理               *)
        (* ================================================================== *)

        Output_ErrCode:=WORD#16#0010;
        Local_Status.Busy:=FALSE;
        Local_Status.Error:=TRUE;
        Local_State:=0;                                  //0:通信非実行状態へ

    ELSE
        Local_State:=99;                                 //99:処理番号異常処理へ

    END_CASE;

END_IF;
```

9.5.3. ファンクションブロックの詳細説明

本プロジェクトファイルのファンクションブロックを以下に示します。
相手機器により可変なデータは、以下に示すファンクションブロックの赤枠内に設定しています。

●ETN_ParameterSet_instance：ファンクションブロック(ParameterSet)の内容

命令	名称	FB/FUN	グラフィック表現	ST 表現
ParameterSet	汎用 Ethernet 通信パラメータ設定	FB	なし	ETN_ParameterSet_instance (Execute, TfsTime, TrTime, TfrTime, , TopenTime, TcloseTime, SrcPort, DstIPAddr, DstPort);

・入出力変数テーブル（引数）

・入力

変数名	データ型	名称	内容	有効範囲	単位	初期値
Execute	BOOL	起動	OFF(FALSE)から ON(TRUE)にするとファンクションブロックが起動します。（常時：TRUE）	データ型に従う	ー	-

・出力

変数名	データ型	名称	内容	有効範囲	単位	初期値
TopenTime	UINT	オープン監視時間	オープン処理の監視時間を 10ms 単位で設定します。	データ型に従う	-	-
TfsTime	UINT	送信監視時間	送信処理の監視時間を 10ms 単位で設定します。	データ型に従う	-	-
TrTime	UINT	受信待ち監視時間	受信データの到着待機時間を 100ms 単位で設定します。	データ型に従う	-	-
TfrTime	UINT	受信処理時間	受信処理の監視時間を 10ms 単位で設定します。	データ型に従う	-	-
TcloseTime	UINT	クローズ監視時間	クローズ処理の監視時間を 10ms 単位で設定します。	データ型に従う	-	-
SrcPort	UINT	自 PortNo	自 port を設定します。	データ型に従う	ー	-
DstIPAddr	STRING [256]	相手 IP アドレス	相手 IP アドレスを設定します。	データ型に従う	-	-
DstPort	UINT	相手 PortNo	相手 PortNo を設定します。	相手機器に従う	ー	ー
Busy	BOOL	実行中	未使用（本プロジェクトでは使用しません。）	ー	ー	ー
Done	BOOL	正常終了				
Error	BOOL	異常終了				
ErrorID	WORD	異常情報				
ErrorIDEx	DWORD	異常情報				

・内部変数テーブル：なし

・プログラム

```
(* ================================================================================ *)
(* 名称：NJｼﾘｰｽﾞ汎用Ethernet通信ﾊﾟﾗﾒｰﾀ設定ﾌｧﾝｸｼｮﾝﾌﾞﾛｯｸ                              *)
(* 機能：各処理監視時間設定およびEthernet関連ﾊﾟﾗﾒｰﾀ設定                             *)
(*                                                                                  *)
(* 対象機器：                                                                       *)
(*    ﾒｰｶ名       ：株式会社ｱｲｴｲｱｲ                                                *)
(*    機器名称    ：ｺﾝﾄﾛｰﾗ                                                         *)
(*    ｼﾘｰｽﾞ/形式：X-SELｼﾘｰｽﾞ                                                       *)
(*    備考        ：ﾌｫｰﾏｯﾄB                                                        *)
(*                                                                                  *)
(* ﾊﾞｰｼﾞｮﾝ情報：2011/12/16 V1.00 新規                                               *)
(*                                                                                  *)
(* (C)Copyright OMRON Corporation 2011 All Rights Reserved.                         *)
(* ================================================================================ *)

(* 変数説明：引数 戻り値 ========================================================== )
(                                                                                  )
( 引数：        名称         ﾃﾞｰﾀ型      内容                                    )
(       ・入力：Execute      BOOL        起動ﾌﾗｸﾞ                                )
(                                                                                  )
(       ・出力：TopenTime    UINT        ｵｰﾌﾟﾝ処理監視時間                        )
(               TfsTime      UINT        送信処理監視時間                        )
(               TrTime       UINT        受信待ち処理監視時間                    )
(               TfrTime      UINT        受信処理監視時間                        )
(               TcloseTime   UINT        ｸﾛｰｽﾞ処理監視時間                        )
(               SrcPort      UINT        自PortNo                                )
(               DstIPAddr    UINT        相手機器IPｱﾄﾞﾚｽ                         )
(               DstPort      UINT        相手機器PortNo                          )
(               Busy         BOOL        未使用                                  )
(               Done         BOOL        未使用                                  )
(               Error        BOOL        未使用                                  )
(               ErrorID      WORD        未使用                                  )
(               ErrorIDEx    DWORD       未使用                                  )
(                                                                                  )
(       ・入出力：なし                                                           )
(                                                                                  )
( 戻り値：なし                                                                   )
(                                                                                  )
(  ================================================================================ *)

IF Execute THEN

    (* Ethernet関連ﾊﾟﾗﾒｰﾀ設定 *)
    SrcPort:= UINT#0;                   // 自ﾎﾟｰﾄNo
    DstIPAddr:= '192.168.250.2';        // 相手IPｱﾄﾞﾚｽ
    DstPort:= UINT#64511;               // 相手ﾎﾟｰﾄNo

    (* 処理監視時間設定：処理開始～終了までの最大時間 *)
    TopenTime := UINT#500;   // ｵｰﾌﾟﾝ処理監視時間設定：設定単位10ms＜500⇒5s＞
    TfsTime:= UINT#500;      // 送信処理監視時間設定：設定単位10ms＜500⇒5s＞
    TfrTime:= UINT#500;      // 受信処理監視時間：設定単位10ms＜500⇒5s＞
    TcloseTime:=UINT#500;    // ｸﾛｰｽﾞ処理監視時間：設定単位10ms＜500⇒5s＞

    (* ﾚｽﾎﾟﾝｽを複数ﾊﾟｹｯﾄで分割受信する場合のﾊﾟｹｯﾄ間隔の最大待ち時間（受信命令）
       および次のﾚｽﾎﾟﾝｽの最大待ち時間（受信待ち時間監視ﾀｲﾏ） *)
    TrTime:= UINT#3;         // 受信待ち監視時間：設定単位100ms＜3⇒300ms＞

END_IF;

RETURN;
```

●ETN_SendMessageSet_instance：ファンクションブロック(SendMessageSet)の内容

命令	名称	FB/FUN	グラフィック表現	ST 表現
SendMessageSet	汎用 Ethernet 通信送受信シーケンス設定	FB	なし	ETN_SendMessageSet_instance (Execute, Send_Data, ComType);

・入出力変数テーブル（引数）

・入力

変数名	データ型	名称	内容	有効範囲	単位	初期値
Execute	BOOL	起動	OFF(FALSE)から ON(TRUE)にするとファンクションブロックが起動します。(常時：TRUE)	データ型に従う	―	-

・出力

変数名	データ型	名称	内容	有効範囲	単位	初期値
Send_Data	STRING [256]	送信データ	相手機器への送信コマンドを設定します。	データ型に従う	-	-
ComType	BYTE	送受信タイプ	送信・受信の有無を設定します。1:送信のみ、2:受信のみ、3:送受信	1～3	-	-
Busy	BOOL	実行中	未使用（本プロジェクトでは使用しません。）	―	-	―
Done	BOOL	正常終了				
Error	BOOL	異常終了				
ErrorID	WORD	異常情報				
ErrorIDEx	DWORD	異常情報				

・内部変数テーブル

変数名	データ型	名称	内容	有効範囲	単位	初期値
Send_Header	STRING[5]	送信ヘッダ	送信メッセージのヘッダ	データ型に従う	―	―
Send_Addr	STRING[5]	相手機器アドレス	相手機器アドレス	データ型に従う	―	―
Send_Command	STRING[256]	送信データ	相手機器への送信コマンド	データ型に従う	―	―
Send_Check	STRING[5]	送信チェックコード	送信メッセージのチェックコード	データ型に従う	―	―
Send_Terminate	STRING[5]	送信ターミネータ	送信メッセージのターミネータ	データ型に従う	―	―

・プログラム

```
(* ================================================================================ *)
(* 名称：NJｼﾘｰｽﾞ汎用Ethernet通信送受信ｼｰｹﾝｽ設定ﾌｧﾝｸｼｮﾝﾌﾞﾛｯｸ                          *)
(* 機能：送信／受信処理の要否および送信ﾃﾞｰﾀ設定                                      *)
(*                                                                                  *)
(* 対象機器：                                                                       *)
(*    ﾒｰｶ名       ：株式会社ｱｲｴｲｱｲ                                                *)
(*    機器名称    ：ｺﾝﾄﾛｰﾗ                                                         *)
(*    ｼﾘｰｽﾞ/形式：X-SELｼﾘｰｽﾞ                                                       *)
(*    備考        ：ﾌｫｰﾏｯﾄB                                                        *)
(*                                                                                  *)
(* ﾊﾞｰｼﾞｮﾝ情報：2011/12/16 V1.00 新規                                               *)
(*                                                                                  *)
(* (C)Copyright OMRON Corporation 2011 All Rights Reserved.                         *)
(* ================================================================================ *)

(* 変数説明：引数 戻り値 ========================================================== )
(                                                                                  )
( 引数：          名称         ﾃﾞｰﾀ型       内容                                 )
(        ・入力：Execute      BOOL         起動ﾌﾗｸﾞ                             )
(                                                                                  )
(        ・出力：SendData     STRING[256] 送信ﾃﾞｰﾀ                              )
(                ComType      BYTE         送信／受信処理の要否設定             )
(                Busy         BOOL         未使用                               )
(                Done         BOOL         未使用                               )
(                Error        BOOL         未使用                               )
(                ErrorID      WORD         未使用                               )
(                ErrorIDEx    DWORD        未使用                               )
(                                                                                  )
(        ・入出力：なし                                                          )
(                                                                                  )
( 戻り値：なし                                                                   )
(                                                                                  )
(  ================================================================================ *)

IF Execute THEN

    (* 送信／受信処理の要否設定 *)
    ComType:= BYTE#16#03;  // 1:送信のみ、2:受信のみ、3:送信／受信の両方

    (* 送信ﾃﾞｰﾀ設定 *)
    Send_Header:='!';                          // ﾍｯﾀﾞ
    Send_Addr:='99';                           // ｱﾄﾞﾚｽ
    Send_Command:= CONCAT('201','00','0'); // 相手機器ｺﾏﾝﾄﾞ：ﾊﾞｰｼﾞｮﾝ読み出し
    Send_Check:=StringSum(CONCAT(Send_Header,Send_Addr,Send_Command),USINT#2);
                                               // SUM計算:ﾍｯﾀﾞからｺﾏﾝﾄﾞまで
    Send_Terminate:='$R$L';                    // ﾀｰﾐﾈｰﾀ：CR+LF(0x0D+0x0A)

    (* 送信ﾃﾞｰﾀの連結 *)
    Send_Data:=
        CONCAT(Send_Header,Send_Addr,Send_Command,Send_Check,Send_Terminate);

END_IF;

RETURN;
```

●ETN_ReceiveCheck_instance：ファンクションブロック(ReceiveCheck)の内容

命令	名称	FB/FUN	グラフィック表現	ST 表現
ReceiveCheck	汎用 Ethernet 通信受信処理	FB	なし	ETN_ReceiveCheck_instance (Execute, Recv_Data, Recv_Buff, Error, ErrorID, ErrorIDEx);

・入出力変数テーブル（引数）

・入力

変数名	データ型	名称	内容	有効範囲	単位	初期値
Execute	BOOL	起動	OFF(FALSE)から ON(TRUE)にするとファンクションブロックが起動します。	データ型に従う	―	-
tLength	UINT	受信データ長	受信バッファデータのバイト長	データ型に従う	―	-

・入出力

変数名	データ型	名称	内容	有効範囲	単位	初期値
Recv_Data	STRING[256]	受信データ	受信データ格納結果	データ型に従う	-	-
Recv_Buff	STRING[256]	受信バッファ	受信バッファデータ	データ型に従う	-	-
ErrorID	WORD	異常情報	エラーコード： 相手機器エラー＝#16#1000 FCS エラー＝#16#2000	-	-	-
ErrorIDEx	DWORD	異常情報	エラーコード： FCS 受信結果/相手機器異常コード	-	-	-

・出力

変数名	データ型	名称	内容	有効範囲	単位	初期値
Busy	BOOL	実行中	未使用 （本プロジェクトでは使用しません。）	-	-	-
Done	BOOL	正常終了				
Error	BOOL	異常終了	異常終了	-	-	-

・内部変数テーブル

変数名	データ型	名称	内容	有効範囲	単位	初期値
Receive_Check	STRING[5]	受信 FCS	受信データの FCS 受信結果	データ型に従う	―	―
Calc_Check	STRING[5]	受信 FCS 計算値	受信データの FCS 計算結果	データ型に従う	―	―

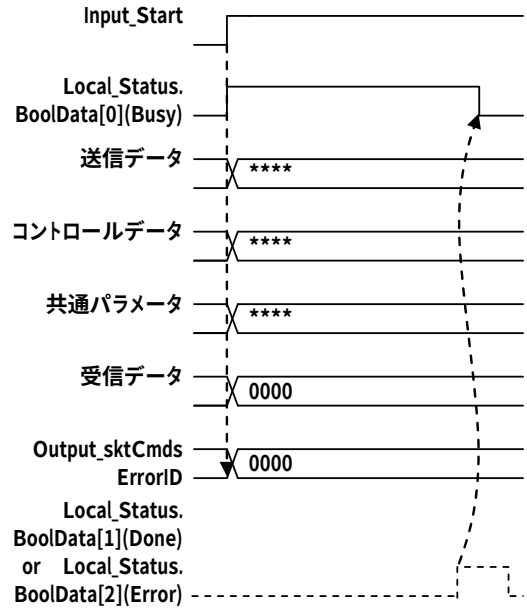
・プログラム

```
(* ================================================================================ *)
(* 名称：NJｼﾘｰｽﾞ汎用Ethernet通信受信処理ﾌｧﾝｸｼｮﾝﾌﾞﾛｯｸ                                *)
(* 機能：受信ﾃﾞｰﾀ格納および受信処理結果判定                                          *)
(*                                                                                  *)
(* 対象機器：                                                                       *)
(*    ﾒｰｶ名       ：株式会社ｱｲｴｲｱｲ                                                *)
(*    機器名称    ：ｺﾝﾄﾛｰﾗ                                                         *)
(*    ｼﾘｰｽﾞ/形式：X-SELｼﾘｰｽﾞ                                                       *)
(*    備考        ：ﾌｫｰﾏｯﾄB                                                        *)
(*                                                                                  *)
(* ﾊﾞｰｼﾞｮﾝ情報：2011/12/16 V1.00 新規                                               *)
(*                                                                                  *)
(* (C)Copyright OMRON Corporation 2011 All Rights Reserved.                         *)
(* ================================================================================ *)

(* 変数説明：引数 戻り値 ============================================================  )
(                                                                                    )
( 引数：          名称         ﾃﾞｰﾀ型       内容                                   )
(       ・入力：Execute      BOOL         起動ﾌﾗｸﾞ                               )
(               tLength      UINT         受信ﾃﾞｰﾀ長                             )
(                                                                                    )
(       ・出力：Busy         BOOL         未使用                                 )
(               Done         BOOL         未使用                                 )
(               Error        BOOL         ｴﾗｰﾌﾗｸﾞ                               )
(                                                                                    )
(       ・入出力：Recv_Data  STRING[256]  受信ﾃﾞｰﾀ格納ｴﾘｱ                         )
(               Recv_Buff    STRING[256]  受信ﾊﾞｯﾌｧ                               )
(               ErrorID      WORD         ｴﾗｰｺｰﾄﾞ                               )
(               ErrorIDEx    DWORD        FCS受信結果あるいは相手機器ｴﾗｰｺｰﾄﾞ      )
(                                                                                    )
( 戻り値：なし                                                                       )
(                                                                                    )
(  ================================================================================ *)
IF Execute THEN

    (* CheckSUMの判定 *)
    (* 受信ﾃﾞｰﾀ（ﾊﾞｯﾌｧ内）のCheckSUM取得 *)
    (* 右端から4文字目を先頭に2文字抜き出し *)
    Receive_Check:= MID(Recv_Buff,UINT#2,(tLength-UINT#3));

    (* 受信ﾃﾞｰﾀ（ﾊﾞｯﾌｧ内）のCheckSUM計算 *)
    (* ﾍｯﾀﾞ「#」～「CheckSUM+CR+LF」直前ﾃﾞｰﾀまでが対象 *)
    Calc_Check:= stringSUM(Left(Recv_Buff,(tLength - UINT#4)),USINT#2);

    (* 取得値と計算値CheckSumの比較 *)
    ErrorID:= SEL(STRING_TO_UDINT(Receive_Check) = STRING_TO_UDINT(Calc_Check),
                  WORD#16#2000,WORD#16#0000 );          // 結果をｴﾗｰｺｰﾄﾞに設定)
    Error:= ErrorID <> WORD#16#0000;                    // ｴﾗｰの場合ｴﾗｰﾌﾗｸﾞｾｯﾄ

    (* CheckSUMｴﾗｰ時の取得CheckSUMの格納 *)
    IF Error THEN
        ErrorIDEx:= STRING_TO_DWORD (Receive_Check);
        RETURN;                                         // Function Block終了
    END_IF;

    (* 受信ﾊﾞｯﾌｧのﾃﾞｰﾀを受信ﾃﾞｰﾀ格納ｴﾘｱに格納 *)
    Recv_Data:= Recv_Buff;

    (* 相手機器異常の判定 *)
    (* 正常：先頭（ﾍｯﾀﾞ）に'#'がある場合 *)
    IF FIND(LEFT(Recv_Buff,1),'#') = UINT#1 THEN
        Error:= FALSE;                                  // ｴﾗｰﾌﾗｸﾞﾘｾｯﾄ
        ErrorID:= WORD#16#0000;                         // ｴﾗｰｺｰﾄﾞｸﾘｱ
        ErrorIDEx:= DWORD#16#00000000;                  // 相手機器ｴﾗｰｺｰﾄﾞｸﾘｱ

    (* 異常：先頭（ﾍｯﾀﾞ）に'#'以外('&')がある場合 *)
    ELSE
        Error:= TRUE;                                   // ｴﾗｰﾌﾗｸﾞｾｯﾄ
        ErrorID:= WORD#16#1000;                         // ｴﾗｰｺｰﾄﾞ設定

        (* 相手機器ｴﾗｰｺｰﾄﾞの格納 *)
        (* 文字列の左から4文字目を先頭に3文字(ASCIIｺｰﾄﾞ)を16進数に変換 *)
        ErrorIDEx:= STRING_TO_DWORD ((MID(Recv_Buff,4,3)));
    END_IF;

END_IF;

RETURN;
```

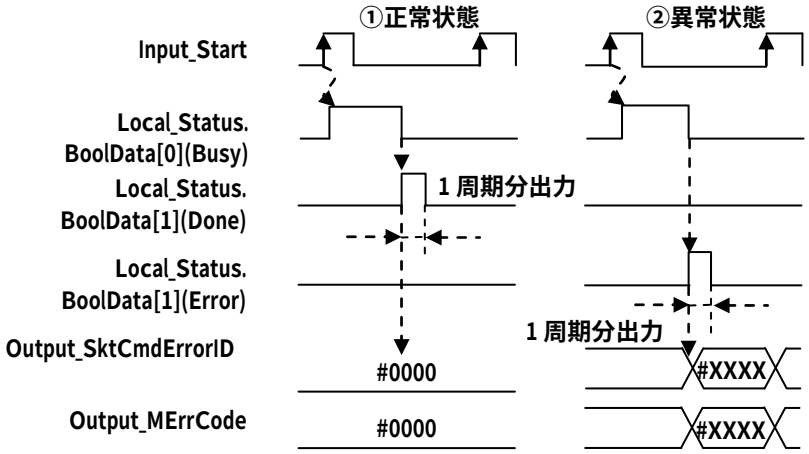
9.6. タイムチャート

ST 言語によるプログラムのタイムチャートです。

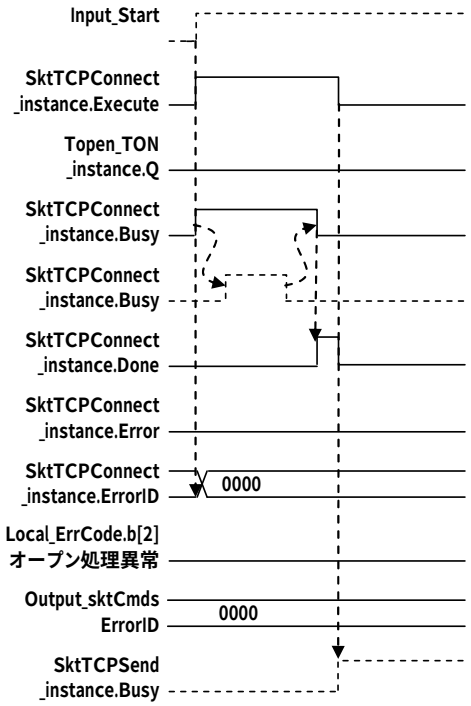
起動＆設定



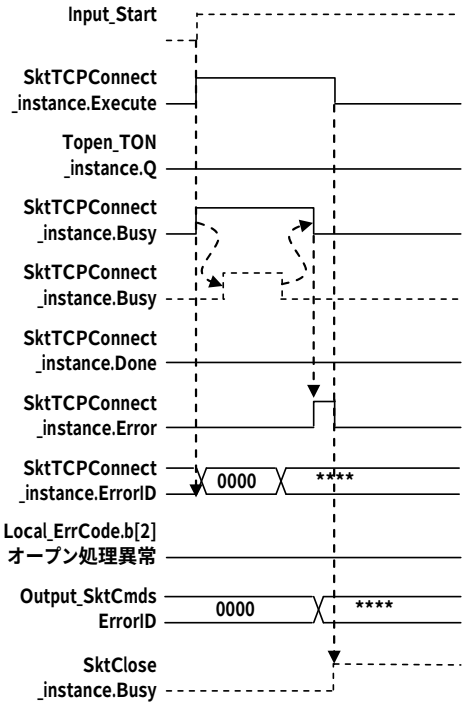
実行途中に［Input_Start］を「True(ON)」から「False(OFF)」に変更した場合、以下のように処理を完了してから正常または異常終了を 1 周期出力します。



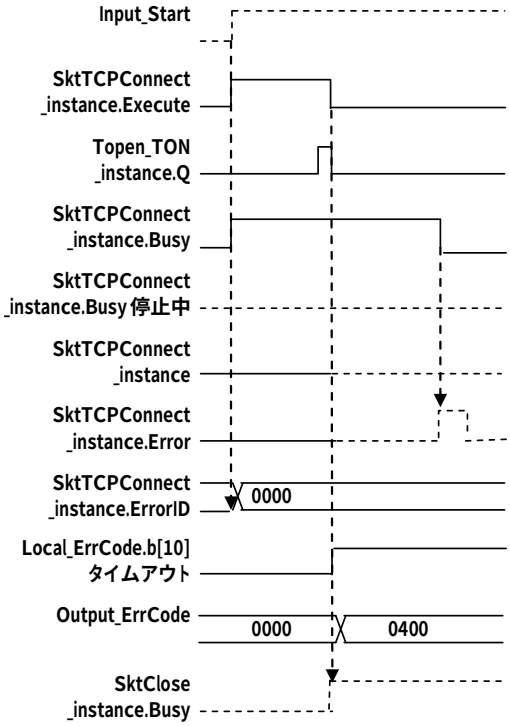
●オープン処理



（正常終了）

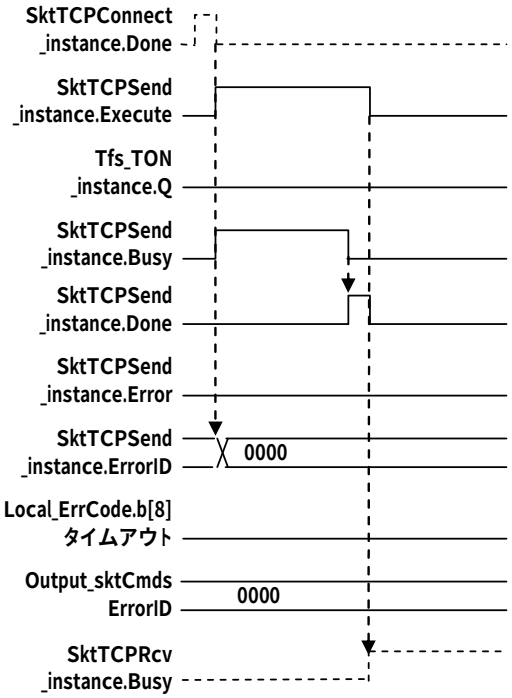


（異常終了）

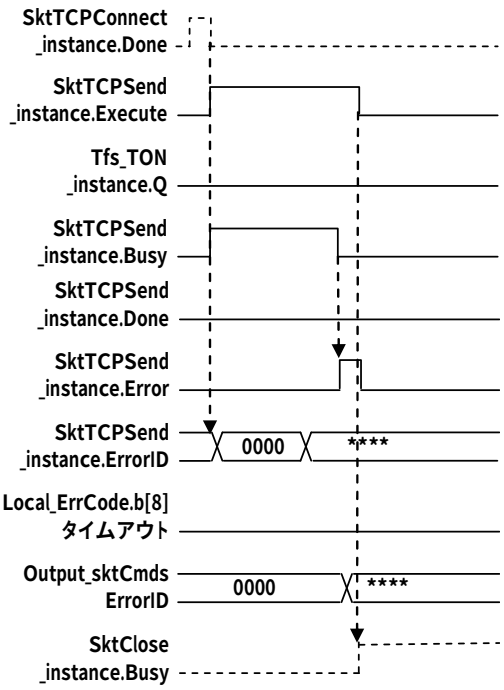


（タイムアウト）

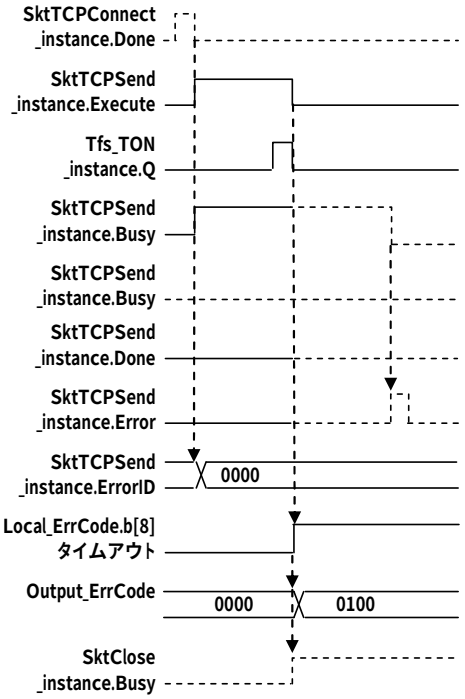
●送信処理



（正常終了）

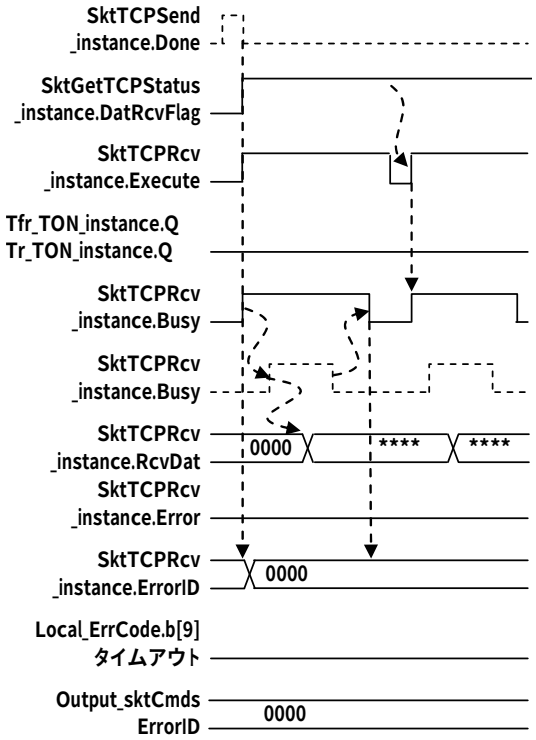


（異常終了）

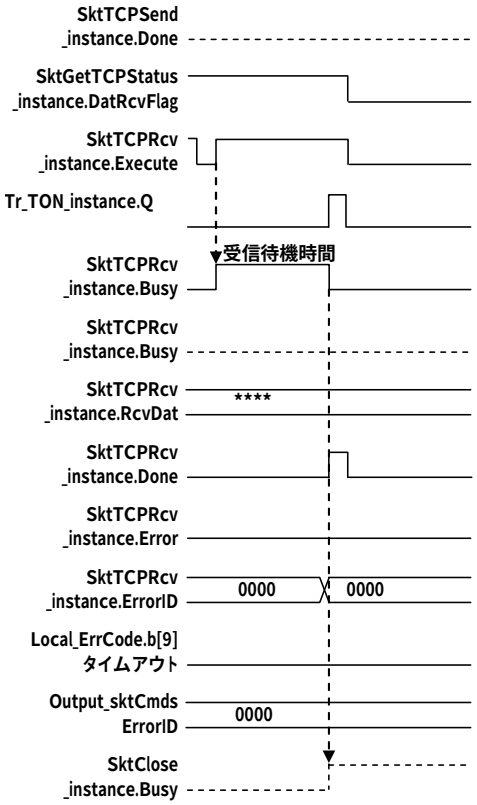


（タイムアウト）

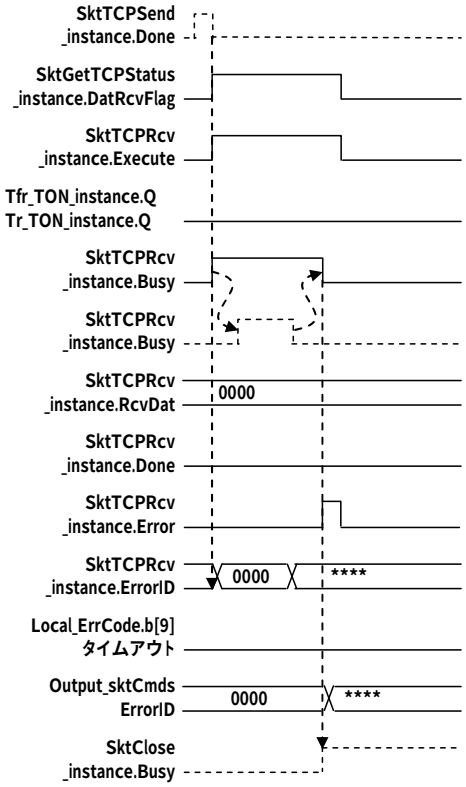
●受信処理



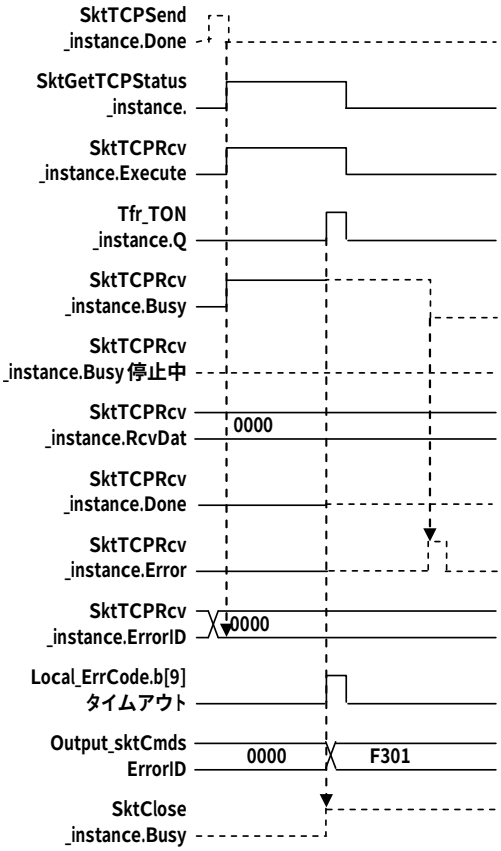
（繰り返し）



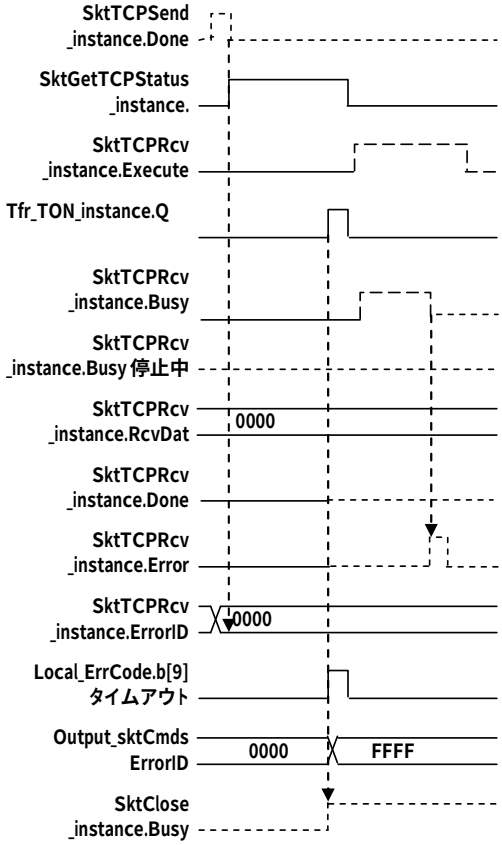
（正常終了）



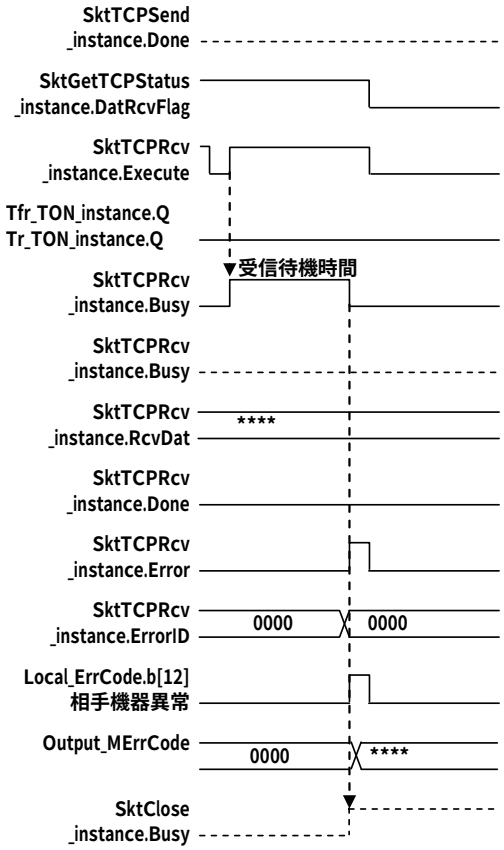
（異常終了）



（タイムアウト：受信異常）

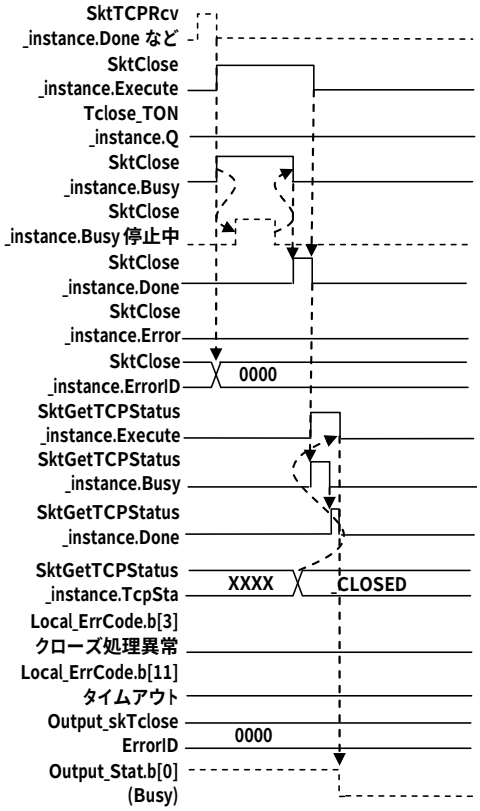


（タイムアウト：受信データなし）

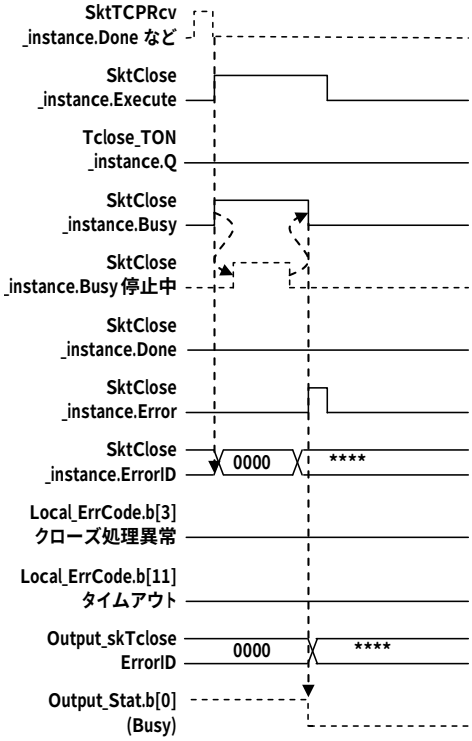


（相手機器異常）

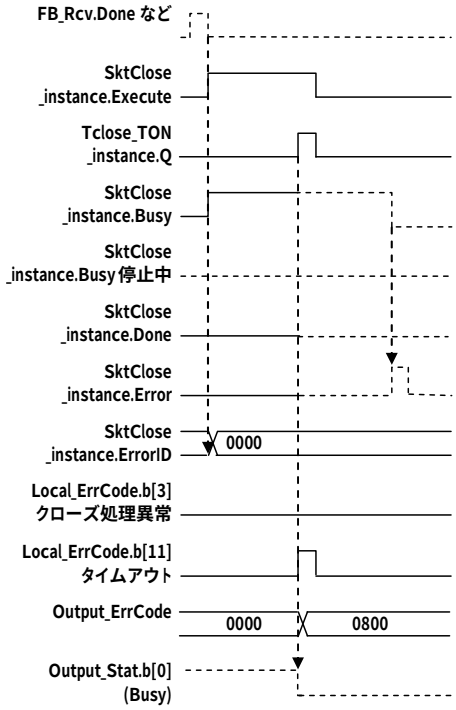
●クローズ処理



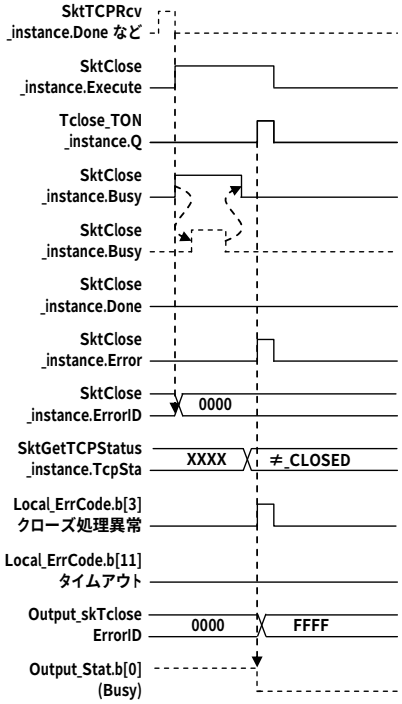
（正常終了）



（異常終了）



（タイムアウト）



（状態異常）

9.7. 異常処理

9.7.1. エラーコード一覧

本 ST 言語を実行して発生するエラーコード一覧を以下に示します。

●TCP コネクション状態異常[Output_EtnTcpSta]

クローズ処理後、TCP コネクション状態が時間内に正常状態(_CLOSED)に至らなかった場合、変数[Output_EtnTcpSta]に TCP コネクション状態コードがセットされます。（クローズ処理が異常終了した場合、いっしょに確認します。）

異常コード列挙子 [_eCONNECTION_STATE]	内容
_CLOSED	クローズされている。（正常状態）
_LISTEN	コネクションを待っている。
_SYN SENT	活性状態で SYN を送信した。
_SYN RECEIVED	SYN を送信し、受信した。
_ESTABLISHED	開設されている。
_CLOSE WAIT	FIN を受信したあと、終了待ちの状態。
_FIN WAIT1	終了して、FIN を送信した。
_CLOSING	終了して、FIN を交換した。FIN の送達確認(ACK)を待っている。
_LAST ACK	FIN を受信して終了した。FIN の送達確認(ACK)を待っている。
_FIN WAIT2	FIN の送達確認(ACK)を受信した。FIN を待っている。
_TIME WAIT	終了したあと、最大セグメント生存時間の 2 倍(2MSL)の沈黙待ちの状態。

●**エラーコード**[Output_SktCmdsErrorID]、[Output_SkTcloseErrorID]

オープン処理、送信処理、受信処理のいずれかでエラーが発生した場合、エラーコードを変数[Output_SktCmdsErrorID]**にセットしたあとクローズ処理を実行します。**
クローズ処理でエラーが発生した場合、エラーコードを変数[Output_SkTcloseErrorID]**にセットして終了します。おもなエラーコードは以下のとおりです。**

（O**：オープン処理**(SktTCPConnect **命令)、**S**：送信処理**(SktTCPSend **命令)、**R**：受信処理**(SktTCPRcv **命令)、**C**：クローズ処理**(SktClose **命令)をあらわし、「○」が対象となる処理です）**

エラーコード	O	S	R	C	内容
#16#0000	○	○	○	○	正常終了
#16#0400	○	○	○	-	命令の入力パラメータが入力変数の範囲を超えています。
#16#0407	-	○	○	-	命令の演算結果が、出力パラメータのデータ領域の範囲を超えています。
#16#2000	○	-	-	-	自 IP アドレスの設定エラーが発生している状態で、命令を実行しました。
#16#2002	○	-	-	-	命令でドメイン名指定した相手ノードのアドレス解決に失敗しました。
#16#2003	○	○	○	-	命令実行時の状態が適切ではありません。 ・SktTCPConnect 命令の場合 入力変数「SrcTcpPort」で指定した TCP ポートがすでにオープン済み 入力変数「DstAdr」で指定した相手ノードが存在しない 入力変数「DstAdr」、「DstTcpPort」で指定した相手ノードがコネクト待ちでない ・SktTCPRcv 命令の場合 指定したソケットが受信処理中 指定したソケットのコネクションが未確立 ・SktTCPSend 命令の場合 指定したソケットが送信処理中 指定したソケットのコネクションが未確立
#16#2006	-	-	○	-	ソケットサービス命令でタイムアウトが発生しました。
#16#2007	-	○	○	○	ソケットサービス命令で指定したハンドルが不正です。
#16#2008	○	○	○	○	同時に実行できるソケットサービス命令のリソースを超えて命令が実行されました。
#16#FFFF	○	○	○	○	命令を実行完了せず終了しました。

参考

詳しくは、「NJ **シリーズ コマンドリファレンスマニュアル 基本編」**(SBCA-360)**の「付録」の「**A-1 **命令に関連するエラーコード一覧」、「**A-2 **エラーコードの概要」「**A-3 **エラーコードの詳細」を参照してください。**

参考

内蔵 EtherNet/IP **ポート異常の詳細および処置については、「**NJ **シリーズ** CPU **ユニット内蔵** Ethernet/IP **ポート ユーザーズマニュアル」**(SBCD-359) **の「第** 9 **章 ソケットサービス機能」の「**9-7 **ソケットサービス使用上の留意事項」を参照してください。**

●エラーフラグ（異常終了／タイムアウト）[Output_ErrCode]

オープン処理、送信処理、受信処理、クローズ処理が異常終了またはタイムアウトした場合、変数[Output_ErrCode]にエラーフラグがセットされ、変数[Output_SktCmdsErrorID]または変数[Output_SkTcloseErrorID]にエラーコードが格納されます。

（クローズ処理が異常終了またはタイムアウトした場合、TCP コネクション状態異常の変数［Output_EtnTcpSta］もいっしょに確認します。）

（O：オープン処理(SktTCPConnect 命令)、S：送信処理(SktTCPSend 命令)、R：受信処理(SktTCPRcv 命令)、C：クローズ処理(SktClose 命令)をあらわし、「○」が対象となる処理です）

エラーフラグ	O	S	R	C	内容
#16#0000	○	○	○	○	正常終了
#16#0001		○			送信処理が異常終了した。
#16#0002			○		受信処理が異常終了した。
#16#0004	○				オープン処理が異常終了した。
#16#0008				○	クローズ処理が異常終了した。
#16#0100		○			送信処理が時間内に完了しなかった。
#16#0200			○		受信処理が時間内に完了しなかった。 （受信すべきレスポンスの到着を確認できなかった場合を含む）
#16#0400	○				オープン処理が時間内に完了しなかった。
#16#0800				○	クローズ処理が時間内に完了しなかった。
#16#0010					処理番号異常
#16#0020					送信・受信要否判定異常
#16#1000					相手機器異常エラー
#16#2000					相手機器 FCS（チェックサム）エラー
#16#8000	○	○	○	○	エラー発生

※エラーフラグには、各処理で検出したエラーフラグを加算した値が格納されます。

●相手機器異常コード

相手機器の受信メッセージにチェックサムエラーがある場合、変数[Output_MErrCode]に相手機器から受信したチェックサムデータが格納されます。

チェックサムエラーがない場合、相手機器の受信メッセージから、変数[Output_MErrCode]に検出されます相手機器異常コードが格納されます。

異常コード	内容
#16#00000000	正常終了
【応答コード】	下記【X-SEL コントローラ PX/QX タイプ取扱説明書】を参照
#16#FFFFFFFF	未実行

参考

エラーコード、相手機器異常の詳細および処置については「株式会社アイエイアイ X-SEL コントローラ PX/QX タイプ取扱説明書」(MJ0152)の「付録」－「エラー表」を参照してください。

9.7.2. TCPコネクション状態異常の状況と対処方法

「TCP コネクション状態異常」発生時の状況と対処方法について説明します。

●TCP コネクション状態異常の影響

「TCP コネクション状態異常」発生後、何も対処を行わずに、あるいは、「TCP コネクション状態異常」の発生に気づかずに本プロジェクトファイルを再度実行した場合、「入力変数「相手 IP アドレス(DstAdr)」、「相手ポート(DstTcpPort)」で指定した相手ノードがコネクト待ちでない」（以下、オープン処理異常）が発生することがあります。これは、前回の通信処理終了時の「TCP コネクション状態異常」が影響していると考えられます。（異常発生時のエラー内容は、「9.7.1 エラーコード一覧」を参照してください。）

●TCP コネクション状態異常発生時の状況

クローズ処理後の「TCP コネクション状態異常」とその影響による次通信処理時の「オープン処理異常」の原因は、いずれも「相手機器のクローズ処理が未完了の状態である」という可能性があります。これは、コントローラ内で本プロジェクトファイルの処理をすべて（クローズ処理まで）終了したにもかかわらず、相手機器からのクローズ完了通知を受け取っていない（相手機器のクローズ処理の完了が未確認である）という状況です。

●対処方法

相手機器のクローズ処理が未完了の可能性がありますので、相手機器の通信ポートがクローズされているかを確認します。その結果、クローズされていない場合や確認ができない場合には、相手機器の通信ポートのリセット処理が必要となります。相手機器の通信ポートのリセット方法には、ソフト的なリスタート処理や電源 OFF→ON による再起動処理などが考えられますが、詳しくは各相手機器の説明書を参照してください。

使用上の注意

相手機器の通信ポートのリセット処理は、相手機器が別の機器と接続状態にないことを確認してから実施してください。

●TCP コネクション状態異常時のコントローラ（内蔵 EtherNet/IP ポート）の状況

「TCP コネクション状態異常」が発生した場合、本プロジェクトファイルによる処理は終了していますが、「9.3.2. 時間監視機能」の「内蔵 EtherNet/IP ポート（TCP/IP 機能）による再送／時間監視」が動作している場合があります。ただし、この再送は以下のような状況で停止しますので、特に意識的に停止する必要はありません。

- プロジェクトファイルの起動により再度オープン処理要求が行われた場合
- 再送中に、ケーブル抜けなどの通信障害が解消された場合
- TCP/IP の時間監視（タイムアウト）機能で再送処理が終了した場合
- コントローラをリスタート、あるいは電源 OFF した場合

10. 改訂履歴

改訂記号	改訂年月日	改訂理由・改訂ページ
A	2011 年 12 月 15 日	初版

本誌には主に機種のご選定に必要な
内容を掲載し、ご使用上の注意事項等は
掲載しておりません。
ご使用上の注意事項等、
ご使用の際に必要な内容につきましては、
必ずユーザーズマニュアルをお読みください。

- 本誌に記載の標準価格はあくまで参考であり、確定されたユーザ購入価格を表示したものではありません。
  本誌に記載の標準価格には消費税が含まれておりません。
- 本誌に記載されているアプリケーション事例は参考用ですので、ご採用に際しては機器・装置の機能や安全性をご確認の上、ご使用ください。
- 本誌に記載のない条件や環境での使用、および原子力制御・鉄道・航空・車両・燃焼装置・医療機器・娯楽機械・安全機器、その他人命や財産に大きな影響が予測されるなど、特に安全性が要求される用途への使用をご検討の場合は、定格・性能に対し余裕を持った使い方やフェールセイフ等の安全対策へのご配慮をいただくとともに、当社営業担当者までご相談いただき仕様書等による確認をお願いします。
- 本製品の内、外国為替及び外国貿易法に定める輸出許可、承認対象貨物（又は技術）に該当するものを輸出（又は非居住者に提供）する場合は同法に基づく輸出許可、承認（又は役務取引許可）が必要です。

オムロン株式会社 インダストリアルオートメーションビジネスカンパニー

●お問い合わせ先

カスタマサポートセンタ

フリーコール **0120-919-066**（クイック / オムロン）

携帯電話・PHSなどではご利用いただけませんので、その場合は下記電話番号へおかけください。

電話 **055-982-5015**（通話料がかかります）

【技術のお問い合わせ時間】
■営業時間：8:00～21:00　■営　業　日：365日
■上記フリーコール以外のFAシステム機器の技術窓口：

電話 **055-977-6389**（通話料がかかります）

【営業のお問い合わせ時間】
■営業時間：9:00～12:00/13:00～17:30（土・日・祝祭日は休業）
■営　業　日：土・日・祝祭日/春期・夏期・年末年始休暇を除く

●FAXによるお問い合わせは下記をご利用ください。
カスタマサポートセンタ　お客様相談室　FAX 055-982-5051

●その他のお問い合わせ先
納期・価格・修理・サンプル・仕様書は貴社のお取引先、
または貴社担当オムロン営業員にご相談ください。

オムロン制御機器の最新情報をご覧いただけます。

www.fa.omron.co.jp

緊急時のご購入にもご利用ください。

オムロン商品のご用命は

カタログ番号　SBCZ-816A　2011年12月現在