MITSUBISHI ELECTRIC

三菱電機 FA 統合コントローラ

MELSEC MX Controller

MELSEC MXコントローラ プログラミングテクニックガイド

-MXR300-16
-MXR300-32
-MXR300-64
-MXR500-128
-MXR500-256
-SW1DND-GXW3-J

安全上のご注意

(ご使用前に必ずお読みください)
本製品のご使用に際しては，本マニュアルおよび本マニュアルで紹介している関連マニュアルをよくお読みいただくとともに，安全に対して十分に注意を払って，正しい取扱いをしていただくようお願いいたします。
この「安全上のご注意」では，安全注意事項のランクを「⚠警告」，「⚠注意」として区分してあります。

⚠ 警告	取扱いを誤った場合に，危険な状況が起こりえて，死亡または重傷を受ける可能性が想定される場合。

⚠ 注意	取扱いを誤った場合に，危険な状況が起こりえて，中程度の傷害や軽傷を受ける可能性が想定される場合および物的損害だけの発生が想定される場合。

なお，⚠注意に記載した事項でも，状況によっては重大な結果に結びつく可能性があります。
いずれも重要な内容を記載していますので必ず守ってください。
本マニュアルは必要なときに読めるよう大切に保管すると共に，必ず最終ユーザまでお届けいただくようお願いいたします。

製品の適用について

(1) 当社シーケンサをご使用いただくにあたりましては，万一シーケンサに故障・不具合などが発生した場合でも重大な事故にいたらない用途であること，および故障・不具合発生時にはバックアップやフェールセーフ機能が機器外部でシステム的に実施されていることをご使用の条件とさせていただきます。

(2) 当社シーケンサは，一般工業などへの用途を対象とした汎用品として設計・製作されています。
したがいまして，以下のような機器・システムなどの特殊用途へのご使用については，当社シーケンサの適用を除外させていただきます。万一使用された場合は当社として当社シーケンサの品質，性能，安全に関る一切の責任（債務不履行責任，瑕疵担保責任，品質保証責任，不法行為責任，製造物責任を含むがそれらに限定されない）を負わないものとさせていただきます。
・各電力会社殿の原子力発電所およびその他発電所向けなどの公共への影響が大きい用途
・鉄道各社殿および官公庁殿など，特別な品質保証体制の構築を当社にご要求になる用途
・航空宇宙，医療，鉄道，燃焼・燃料装置，乗用移動体，有人搬送装置，娯楽機械，安全機械など
生命，身体，財産に大きな影響が予測される用途
ただし，上記の用途であっても，具体的に使途を限定すること，特別な品質（一般仕様を超えた品質等）をご要求されないこと等を条件に，当社の判断にて当社シーケンサの適用可とする場合もございますので，詳細につきましては当社窓口へご相談ください。

(3) DoS攻撃，不正アクセス，コンピュータウイルスその他のサイバー攻撃により発生するシーケンサ，およびシステムトラブル上の諸問題に対して，当社はその責任を負わないものとさせていただきます。

はじめに

このたびは三菱電機シーケンサをお買い上げいただき，まことにありがとうございました。
本マニュアルは，MELSEC MXコントローラ(MX-Rモデル)の下記の対象ユニットをご使用いただくときに有効なプログラミングテクニックについてご理解いただくためのマニュアルです。
ご使用前に本マニュアルや関連マニュアルをよくお読みいただき，コントローラの機能・性能を十分ご理解の上，正しくご使用くださるようお願いいたします。
また，本マニュアルで紹介するプログラム例を実際のシステムへ流用する場合は，対象システムにおける制御に問題がないことを十分検証ください。
本マニュアルにつきましては最終ユーザまでお届けいただきますよう，宜しくお願い申し上げます。

対象ユニット

MXR300-16，MXR300-32，MXR300-64，MXR500-128，MXR500-256

目次

安全上のご注意 1
製品の適用について 2
はじめに 2
関連マニュアル 6
用語 7
総称/略称 7

第1章　概要 8

第2章　プロジェクトの可読性の向上 10

2.1 FBや構造体による部品化 10
概要 10
実行手順 11
2.2 FBの配列化 16
概要 16
実行手順 16
2.3 モーション軸のグルーピング 20
概要 20
実行手順 20
2.4 グローバルラベルの分割 24
概要 24
実行手順 24

第3章　プログラム変換の高速化 27

3.1 プログラムファイルの分割 28
概要 28
実行手順 30
3.2 プログラムのFB化 37
概要 37
実行手順 38
3.3 制御構文を含むプログラムの短縮 42
概要 42
実行手順 43
3.4 グローバルラベル(構造体)の設定変更 45
概要 45
実行手順 46
3.5 配列同一要素への連続参照の削減 50
概要 50
実行手順 50

第4章　スキャンタイムの高速化 53

4.1 BMOV命令の複数命令への置き換え 54
概要 54
実行手順 54
4.2 立上り実行命令の削減 59
概要 59
実行手順 59

4.3 ユニットアクセス命令の削減 61
概要 61
実行手順 62
4.4 初期実行タイププログラムの活用 64
概要 64
実行手順 64
4.5 引数の省略と直接参照 70
概要 70
実行手順 70
4.6 命令数が多い回路ブロックの分割 74
概要 74
実行手順 74
4.7 ラベル初期値の活用 77
概要 77
実行手順 77

第5章 モーション制御の最適化 79
5.1 通信周期混在を使用した特定軸の高速化 79
概要 79
実行手順 79
5.2 ユーザプログラムとモーション演算の同期 84
概要 84
実行手順 84

第6章 ラベルの効率化 91
6.1 グローバルラベルの削減 91
概要 91
実行手順 91
6.2 未使用ラベルの削除 93
概要 93
実行手順 93

第7章 CC-Link IE TSN通信の高速化 95
7.1 リンクリフレッシュ時間の短縮 96
概要 96
実行手順 97
7.2 デバイス局の台数の均等化 103
概要 103
実行手順 103
7.3 送信上限サイズの設定 104
概要 104
実行手順 104
7.4 デバイス局の台数の削減 106
概要 106
実行手順 106
7.5 リンクデバイス設定の削除 110
概要 110
実行手順 110
7.6 リンクデバイス設定の点数の削減 115
概要 115

実行手順 115
7.7 ネットワーク同期通信設定の統一 120
概要 120
実行手順 120
7.8 サイクリック伝送分散の設定 123
概要 123
実行手順 123
改訂履歴 127
サービスのお問い合わせ 128
商標 128

CONTENTS

関連マニュアル

最新のe-ManualおよびマニュアルPDFは，三菱電機FAサイトからダウンロードできます。
www.MitsubishiElectric.co.jp/fa

マニュアル名称	内容
MELSEC MXコントローラ プログラミングテクニックガイド [BCN-89999-9684](本マニュアル)	MELSEC MXコントローラのプロジェクトの可読性の向上やプログラム変換の高速化，スキャンタイムの高速化，モーション制御の最適化，ラベルの効率化，CC-Link IE TSN通信の高速化に関するプログラミングテクニックについて記載しています。
MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル [SH-082640]	コントローラの運転までの手順，仕様，デバイス，メモリ，機能，パラメータ，トラブルシューティングなどについて記載しています。
MELSEC MXコントローラ(MX-Rモデル) プログラミングマニュアル [SH-082643]	プログラム言語の仕様，コントローラの命令，汎用ファンクション/ファンクションブロック，モーション制御ファンクションブロックの仕様について記載しています。
GX Works3 オペレーティングマニュアル [SH-081214]	GX Works3のシステム構成，パラメータ設定，オンライン機能の操作方法などについて説明しています。

用語

本マニュアルでは，特に明記する場合を除き，下記の用語を使用して説明します。

用語	内容
グローバルラベル	プロジェクト内で複数のプログラムデータを作成したときに，すべてのプログラムデータに対して有効となるラベルです。 グローバルラベルには，GX Works3が自動で生成するユニット固有のラベル(ユニットラベル)と任意で指定のデバイスに対して作成できるラベルがあります。
デバイス局	マスタ局以外の局(ローカル局，リモート局)です。
プログラムブロック	プログラムを構成するプログラム部品の集まりです。
マスタ局	ネットワーク全体を制御する局です。すべての局とサイクリック伝送およびトランジェント伝送ができます。 1ネットワークに1台のみ存在します。
モーション制御FB	モーション制御に関連するFBです。名称がMC_またはMCv_から始まります。
ユニットラベル	各ユニットが固有で定義しているメモリ(入出力信号やバッファメモリ)を，任意の文字列で表したものです。 使用するユニットからGX Works3が自動で生成し，グローバルラベルとして使用できます。
ラベル	入出力データや内部処理に任意の文字列を指定した変数です。
リモート局	ビット単位の入出力信号とワード単位の入出力データをサイクリック伝送する局です。トランジェント伝送もできます。
リンクデバイス	CC-Link IE TSN機能部またはネットワークユニットが内部に持っているデバイス(RX，RY，RWr，RWw，LB，LW)です。
ローカル局	マスタ局および他ローカル局と，サイクリック伝送およびトランジェント伝送を行う局です。
仮想ドライブ軸	仮想的に指令を生成できる仮想軸です。
軸	モーション制御を行う対象です。
軸変数	軸に関するパラメータやデータを含む，AXIS_*型変数のインスタンスです。
実エンコーダ軸	デバイス機器に接続したエンコーダの現在位置から指令位置を生成する軸です。単軸同期制御の主軸に使用します。
実ドライブ軸	CiA402ドライブプロファイルのcsp/csv/cstモード(逐次指令)に対応するCC-Link IE TSN対応デバイス機器に紐付いた軸です。
CC-Link IE TSN	標準Ethernet規格を拡張した「TSN(Time-Sensitive Networking)」を採用し，リアルタイム性を確保した制御と他オープンネットワークの情報を同時に扱えるオープンネットワークです。
CC-Link IE TSN Class	CC-Link協会による，CC-Link IE TSNに対応した機器およびスイッチングHUBの，機能・性能によるランク分けです。 CC-Link IE TSN Classについては，CC-Link協会から発行されているCC-Link IE TSN 敷設マニュアル(BAP-C3007-001)を参照してください。
FB	FBとはファンクションブロック(Function Block)の略称で，シーケンスプログラム内で繰り返し使用する回路ブロックを部品化して，シーケンスプログラムの中で流用できるようにしたものです。 これにより，プログラム開発を効率化するとともにプログラムミスを削減し，プログラムの品質が向上します。
FBインスタンス	シーケンスプログラムに貼り付けられたファンクションブロックです。
GX Works3	MELSECシーケンサソフトウェアパッケージの製品名です。
RWr	リンクデバイスのリモートレジスタです。デバイス局からマスタ局に16ビット(1ワード)単位で入力されます。(ローカル局では一部異なる。)
RWw	リンクデバイスのリモートレジスタです。マスタ局からデバイス局に16ビット(1ワード)単位で出力されます。(ローカル局では一部異なる。)
RX	リンクデバイスのリモート入力です。デバイス局からマスタ局にビット単位で入力されます。(ローカル局では一部異なる。)
RY	リンクデバイスのリモート出力です。マスタ局からデバイス局にビット単位で出力されます。(ローカル局では一部異なる。)

総称/略称

本リファレンスでは，特に明記する場合を除き，下記に示す総称および略称を使って説明します。

用語	内容
エンジニアリングツール	コントローラやサーボアンプの設定から，プログラミング，デバッグ，保守までを行うためのツールです。
コントローラ	MXR300-16，MXR300-32，MXR300-64，MXR500-128，MXR500-256の総称です。

1 概要

本書では，MELSEC MXコントローラ(以降コントローラと呼びます)のプロジェクト・プログラムの構造化や高速化，モーション制御の最適化，ラベルの効率化，CC-Link IE TSN通信の高速化などを行うためのテクニックについて説明します。
コントローラの基本的なパラメータ設定やプログラミングについては，下記を参照してください。
📖MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル
📖MELSEC MXコントローラ(MX-Rモデル) プログラミングマニュアル
また本書は，下記のバージョンのGX Works3で説明しています。
• GX Works3 Ver.1.115V

テクニックと効果

各テクニック対して想定される効果は以下のとおりです。

テクニック			効果				参照
			可読性/移植性向上	変換時間短縮	スキャンタイム短縮	同期性向上	
プログラムの構造化		FBや構造体による部品化	○	—	—	—	☞10ページ プロジェクトの可読性の向上
		FBの配列化	○	—	—	—	
		モーション軸のグルーピング	○	—	—	—	
		グローバルラベルの分割	○	○	—	—	
プログラム変換の高速化		プログラムファイルの分割	○	○	—	—	☞27ページ プログラム変換の高速化
		プログラムのFB化	○	○	—	—	
		制御構文を含むプログラムの短縮	○	○	—	—	
		グローバルラベル(構造体)の設定変更	—	○	—	—	
		配列同一要素への連続参照の削減	—	○	○	—	
スキャンタイムの高速化		BMOV命令の複数命令への置き換え	—	—	○	—	☞53ページ スキャンタイムの高速化
		立上り実行命令の削減	—	—	○	—	
		ユニットアクセス命令の削減	—	—	○	—	
		初期実行タイププログラムの活用	—	—	○	—	
		引数の省略と直接参照	—	—	○	—	
		命令数が多い回路ブロックの分割	○	—	○	—	
		ラベル初期値の活用	○	—	○	—	
モーション制御の最適化		通信周期混在を使用した特定軸の高速化	—	—	○	—	☞79ページ モーション制御の最適化
		ユーザプログラムとモーション演算の同期	—	—	—	○	
ラベルの効率化		グローバルラベルの削除	—	○	—	—	☞91ページ ラベルの効率化
		未使用ラベルの削除	—	○	—	—	
CC-Link IE TSN通信の高速化	リンクリフレッシュ時間の短縮	リンクリフレッシュ時間の短縮	—	—	○	—	☞95ページ CC-Link IE TSN通信の高速化
	通信周期間隔/サイクリック伝送時間の短縮	デバイス局の台数の均等化	—	—	○	—	
		送信上限サイズの設定					
		デバイス局の台数の削減					
		リンクデバイス設定の削除					
		リンクデバイス設定の点数の削減					
		ネットワーク同期通信設定の統一					
		サイクリック伝送分散の設定					

MEMO

2 プロジェクトの可読性の向上

プログラムを小さな処理単位(部品)に分けて階層的な構造にし，プロジェクトの基本的な構成やルールを統一すると，プロジェクトの可読性が向上します。プロジェクトの可読性の向上により，プロジェクトに変更がある場合の修正箇所がわかりやすくなり，また，プロジェクト作成者以外の人にも理解しやすくなります。
本章では，プロジェクトの可読性を向上させるためのテクニックについて示します。

テクニック一覧

本章で説明するテクニック一覧を下記に示します。

項目	内容	参照
FBや構造体による部品化	各プログラムブロックで共通で使用するプログラムをFBや構造体を使用して部品化します。	☞ 10ページ FBや構造体による部品化
FBの配列化	同じタイミングで実行するFBを配列化し，FOR文と組み合わせたFBに置き換えます。	☞ 16ページ FBの配列化
モーション軸のグルーピング	実ドライブ軸や実エンコーダ軸，仮想ドライブ軸などの軸種別ごとに軸番号をグループ分けします。	☞ 20ページ モーション軸のグルーピング
グローバルラベルの分割	グローバルラベルを意味のある単位でまとめ，異なるグローバルラベルエディタへ移動させます。	☞ 24ページ グローバルラベルの分割

2.1 FBや構造体による部品化

概要

各プログラムブロックで共通で使用するFBや構造体を部品化することで，プログラムの記述量が減少し，読みやすいプログラムを作成できます。
プログラムの部品化により，下記のようなメリットもあります。

プログラムの移植性・再利用性の向上

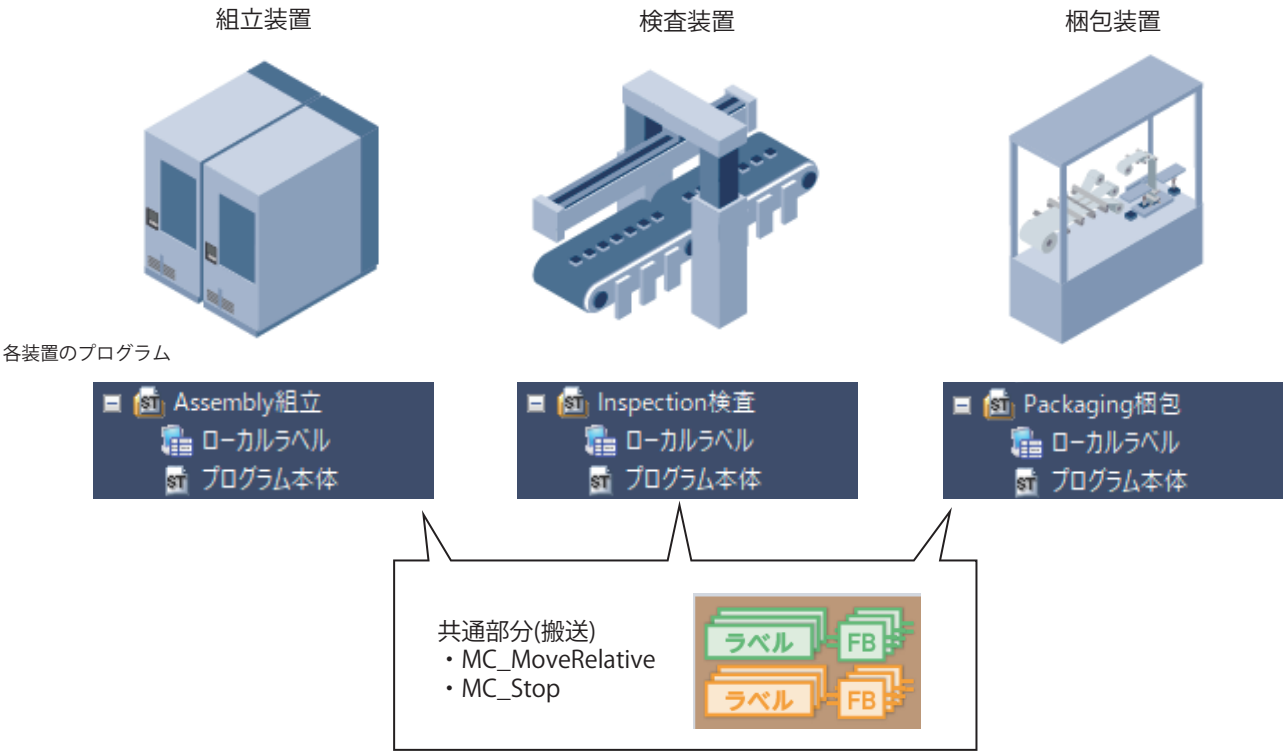
プログラムを部品化することで，処理や工程単位のプログラムの移植や再利用がしやすくなります。

プログラムの保守性の向上

プログラムを部品化し複数の処理や工程で利用すると，処理に追加や変更がある場合でも，部品化した構造体やFBのみを修正すれば複数のプログラムをそれぞれ修正する必要がなくなるため，プログラムの保守性が向上します。

実行手順

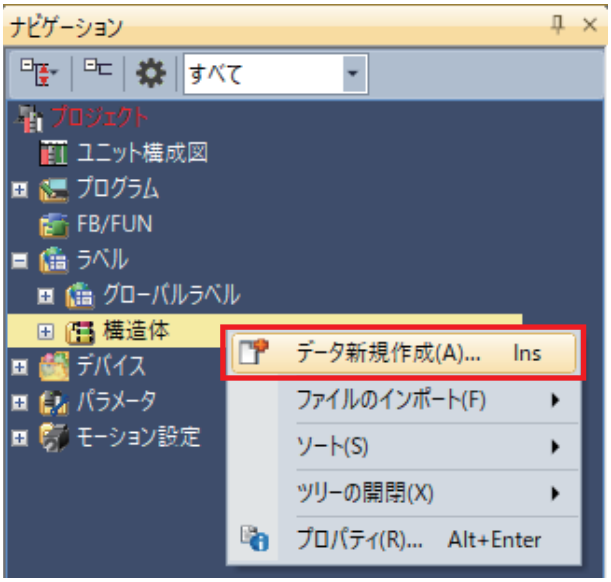
下記の組立装置，検査装置，梱包装置を例に，搬送の共通部品を作成して各装置のプログラムで使用する手順を示します。搬送の共通部品にはワークの搬送を行う"MC_MoveRelative"(相対値位置決め)，停止を行う"MC_Stop"(強制停止)のモーション制御FBを使用します。



操作手順

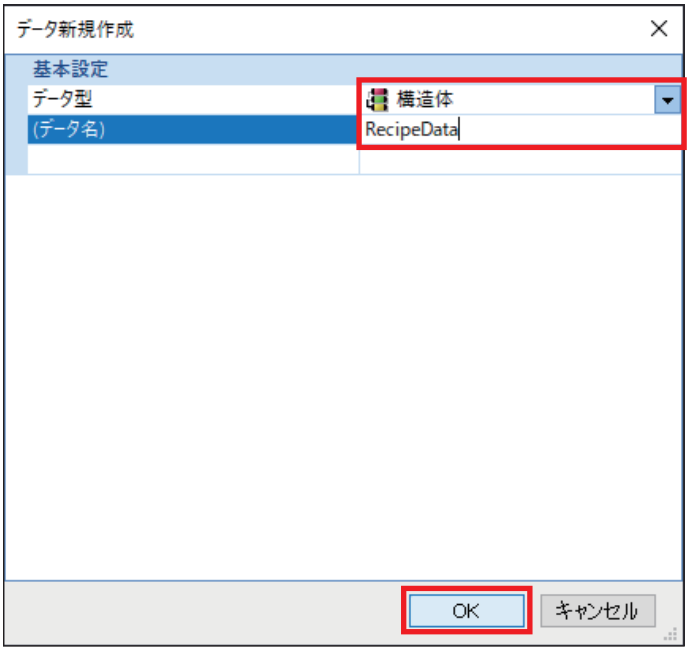
1. 各工程の搬送および停止のレシピデータを定義する構造体を新規作成します。

ナビゲーションウィンドウ⇨[ラベル]⇨[構造体]を右クリック⇨[データ新規作成]

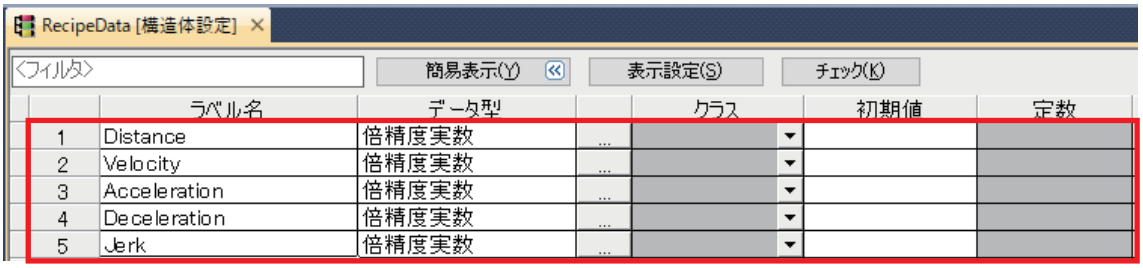


2. 各項目を設定し，[OK]ボタンをクリックします。

- データ型: 構造体
- (データ名): RecipeData



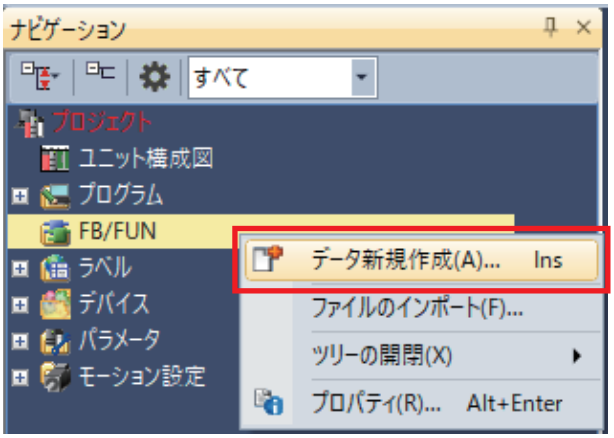
3. 構造体"RecipeData"のメンバに，以下の"MC_MoveRelative"および"MC_Stop"の共通のラベルを定義します。



No.	ラベル名	データ型
1	Distance	倍精度実数
2	Velocity	倍精度実数
3	Acceleration	倍精度実数
4	Deceleration	倍精度実数
5	Jerk	倍精度実数

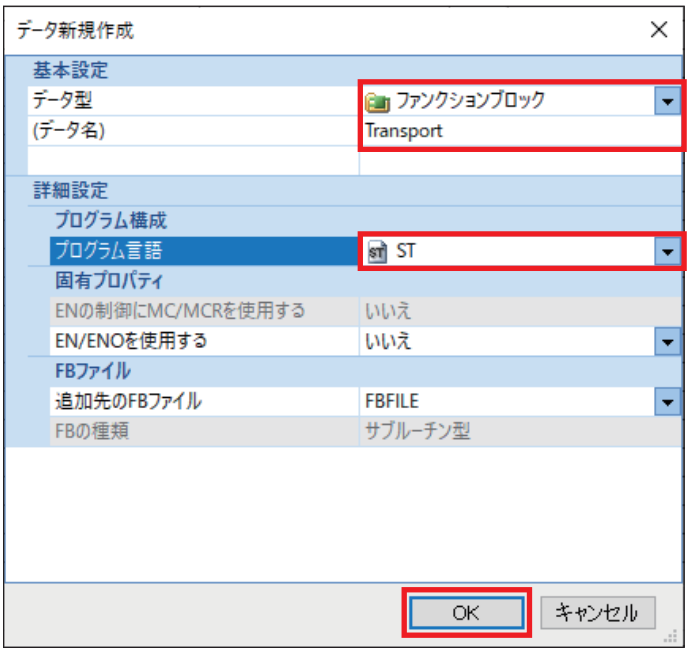
4. 共通部品のFBを新規作成します。

ナビゲーションウィンドウ⇨[FB/FUN]を右クリック⇨[データ新規作成]



5. 各項目を設定し，[OK]ボタンをクリックします。

- データ型: ファンクションブロック
- (データ名): Transport
- プログラム言語: ST



6. FB"Transport"のローカルラベルを開き，搬送に必要な以下のFBインスタンスおよびラベルを定義します。



No.	ラベル名	データ型	クラス	備考
1	MC_MoveRelative_1	MC_MoveRelative	VAR	モーション制御FB"MC_MoveRelative"のFBインスタンスです。
2	MC_Stop_1	MC_Stop	VAR	モーション制御FB"MC_Stop"のFBインスタンスです。
3	Axis_Transport	AXIS_REF	VAR_INPUT	構造体"AXIS_REF"をデータ型に使用します。
4	Start_Transport	ビット	VAR_INPUT	—
5	Stop_Transport	ビット	VAR_INPUT	—
6	RecipeData_Transport	RecipeData	VAR_PUBLIC	手順1～3で作成した構造体"RecipeData"をデータ型に使用します。
7	RecipeData_Stop	RecipeData	VAR_PUBLIC	

7. FB"Transport"のプログラム本体を開き，モーション制御FB"MC_MoveRelative"，"MC_Stop"を行う以下のプログラムを記述します。

Transport [FB] [ST] *

```
1  //ワークの搬送
2  MC_MoveRelative_1.Axis := Axis_Transport;
3  MC_MoveRelative_1(
4      Execute:= Start_Transport ,
5      Distance:= RecipeData_Transport.Distance ,
6      Velocity:= RecipeData_Transport.Velocity ,
7      Acceleration:= RecipeData_Transport.Acceleration ,
8      Deceleration:= RecipeData_Transport.Deceleration ,
9      Jerk:= RecipeData_Transport.Jerk
10     );
11
12 //停止
13 MC_Stop_1.Axis := Axis_Transport;
14 MC_Stop_1(
15     Execute:= Stop_Transport ,
16     Deceleration:= RecipeData_Stop.Deceleration ,
17     Jerk:= RecipeData_Stop.Jerk
18     );
```

行数	処理内容
1～3行目	モーション制御FB"MC_MoveRelative"，"MC_Stop"の引数"Axis"に，ラベル"Axis_Transport"を指定します。
5～13行目	モーション制御FB"MC_MoveRelative"を実行します。 引数"Distance"，"Velocity"，"Acceleration"，"Deceleration"，"Jerk"には，ラベル"RecipeData_Transport"の各メンバを指定します。
15～20行目	モーション制御FB"MC_Stop"を実行します。 引数"Deceleration"，"Jerk"には，ラベル"RecipeData_Stop"の各メンバを指定します。

8. 組立装置のプログラムブロック(本例では"Assembly 組立")のローカルラベルを開き，手順4～7で作成したFB"Transport"のFBインスタンスを追加します。

- ラベル名: Transport
- データ型: Transport(FB"Transport")
- クラス: VAR

Assembly組立 [PRG] [ローカルラベ...

<フィルタ> 簡易表示(Y) 表示設定(S) チェック(K)

	ラベル名	データ型		クラス	初期値	定数
1	Transport	Transport	...	VAR		

9. 各装置のプログラム本体を開き，共通部品のFB"Transport"を使用した任意の搬送のプログラムを作成します。

・組立装置のプログラム

Assembly組立 [PRG] [ST] 28バイト

```
//搬送処理
//初期設定
Transport.Axis_Transport := Axis0001.AxisRef;
Transport.RecipeData_Transport.Distance := 10000.0;
Transport.RecipeData_Transport.Acceleration := 2000.0;
Transport.RecipeData_Transport.Deceleration := 2000.0;
Transport.RecipeData_Transport.Jerk := 300.0;

//搬送開始
IF  G_Process = G_Process_Assembly  THEN
    IF  NOT G_Stop_Assembly  THEN
        Transport.Start_Transport := TRUE ;
    ELSE
        Transport.Start_Transport := FALSE ;
        Transport.Stop_Transport := TRUE ;
    END_IF;
END_IF;
Transport();

//以降組立の独自処理
```

・検査装置のプログラム

Inspection検査 [PRG] [ST] 28バイト

```
//搬送処理
//初期設定
Transport.Axis_Transport := Axis0001.AxisRef;
Transport.RecipeData_Transport.Distance := 20000.0;
Transport.RecipeData_Transport.Acceleration := 2000.0;
Transport.RecipeData_Transport.Deceleration := 2000.0;
Transport.RecipeData_Transport.Jerk := 300.0;

//搬送開始
IF  G_Process = G_Process_Inspection  THEN
    IF  NOT G_Stop_Inspection  THEN
        Transport.Start_Transport := TRUE ;
    ELSE
        Transport.Start_Transport := FALSE ;
        Transport.Stop_Transport := TRUE ;
    END_IF;
END_IF;
Transport();

//以降検査の独自処理
```

・梱包装置のプログラム

Packaging梱包 [PRG] [ST] 28バイト

```
//搬送処理
//初期設定
Transport.Axis_Transport := Axis0001.AxisRef;
Transport.RecipeData_Transport.Distance := 30000.0;
Transport.RecipeData_Transport.Acceleration := 2000.0;
Transport.RecipeData_Transport.Deceleration := 2000.0;
Transport.RecipeData_Transport.Jerk := 300.0;

//搬送開始
IF  G_Process = G_Process_Packaging  THEN
    IF  NOT G_Stop_Packaging  THEN
        Transport.Start_Transport := TRUE ;
    ELSE
        Transport.Start_Transport := FALSE ;
        Transport.Stop_Transport := TRUE ;
    END_IF;
END_IF;
Transport();

//以降梱包の独自処理
```

2.2 FBの配列化

概要

同じタイミングで複数回実行するFBを配列化しFOR文で繰り返し実行することにより，プログラムの記述量が減少するため，読みやすいプログラムを作成できます。
FBを配列化することで，下記のようなメリットもあります。

プログラムの移植性の向上

プログラムの記述量の減少によりプログラムを移植する際のプログラム量も削減できるため，プログラムの移植性が向上します。

実行手順

下記のモーション制御プログラムを例に，複数の軸に対して同じタイミングで実行するモーション制御FB"MC_Power"を配列化する手順を示します。

・STプログラム

ProgPou1 [PRG] [ST]

```
MC_Power_1(
    Axis:= Axis0001.AxisRef ,
    Enable:= TRUE ,
    ServoON:= TRUE
//  Status=> ?BOOL? ,
//  ReadyStatus=> ?BOOL? ,
//  Busy=> ?BOOL? ,
//  Error=> ?BOOL? ,
//  ErrorID=> ?WORD?
    );

MC_Power_2(
    Axis:= Axis0002.AxisRef ,
    Enable:= TRUE ,
    ServoON:= TRUE
//  Status=> ?BOOL? ,
//  ReadyStatus=> ?BOOL? ,
//  Busy=> ?BOOL? ,
//  Error=> ?BOOL? ,
//  ErrorID=> ?WORD?
    );
```

軸No.1～100に対して
モーション制御FB"MC_Power"を
同じタイミングで実行

・ローカルラベル



"MC_Power_1"～
"MC_Power_100"の
FBインスタンス

操作手順

1. 配列化するFBのインスタンスを1次元の配列にします。

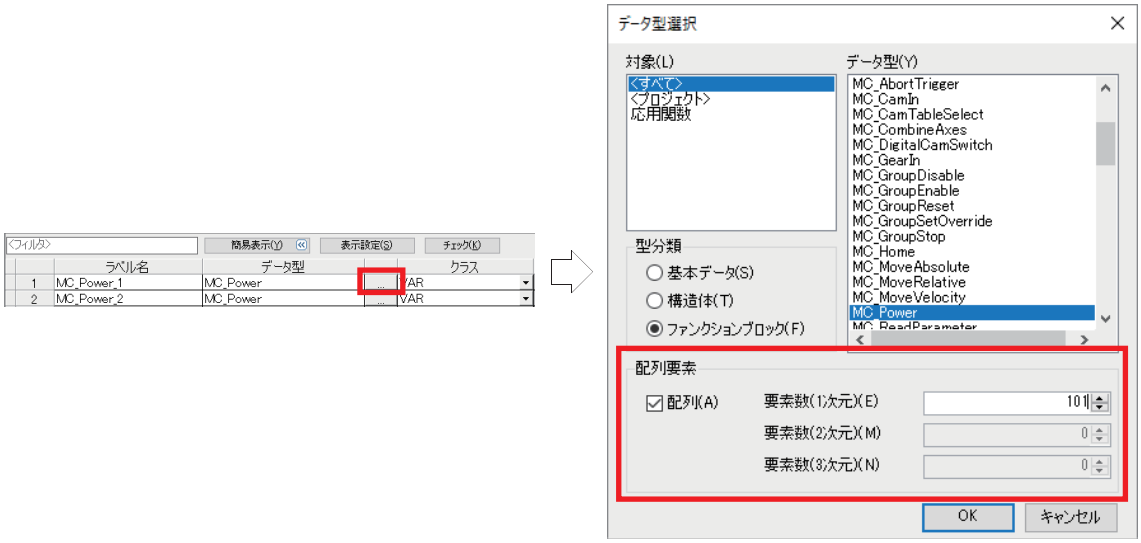
ローカルラベルを開き，FBインスタンス"MC_Power_1"のデータ型に"(1..100)"を追加します。また配列化に応じてラベル名を変更します。

- ラベル名: MC_Power_Axis1_100
- データ型: MC_Power(1..100)



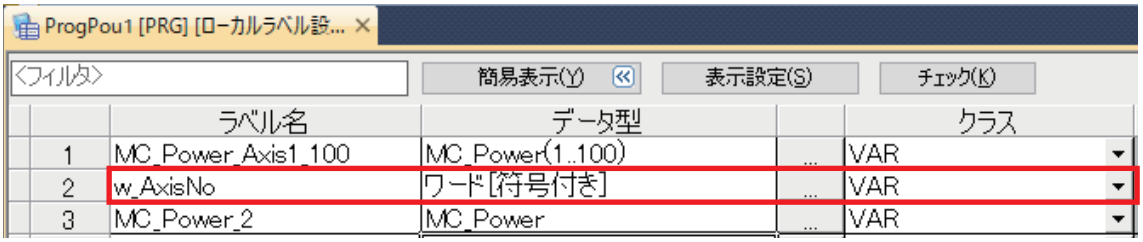
Point

- 配列要素数は，FBを実行する軸No.の先頭から後尾までを設定します。
- ラベルのデータ型は"データ型"列の[...]ボタンをクリックして表示される「データ型選択」画面からも変更できます。"配列要素"欄から"配列"にチェックを入れ，要素数を設定してください。



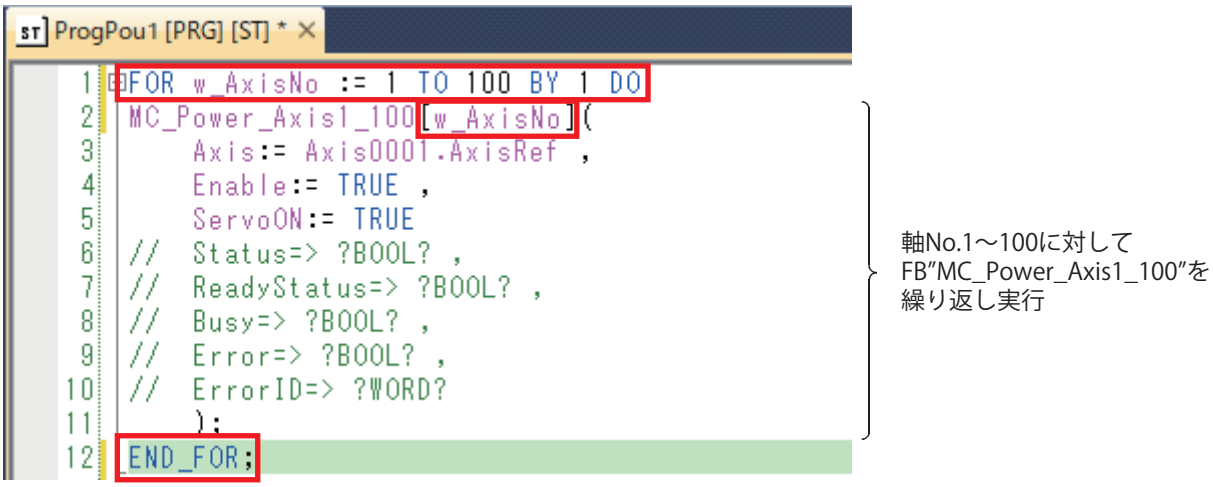
2. FOR文の反復変数に使用するローカルラベル(モーション制御FBを実行する軸No.を示すラベル)を新規作成します。

- ラベル名: w_AxisNo
- データ型: ワード[符号付き]
- クラス: VAR



3. プログラムを開き，FBインスタンス"MC_Power_Axis1_100"に配列要素"w_AxisNo"を追加します。
また配列要素"w_AxisNo"を1～100まで1つずつ増加し，軸No.1～100に対してFBを実行する以下のFOR文を追加します。

```
FOR w_AxisNo := 1 TO 100 BY 1 DO
MC_Power_Axis1_100[w_AxisNo](
実行文;
);
END_FOR;
```

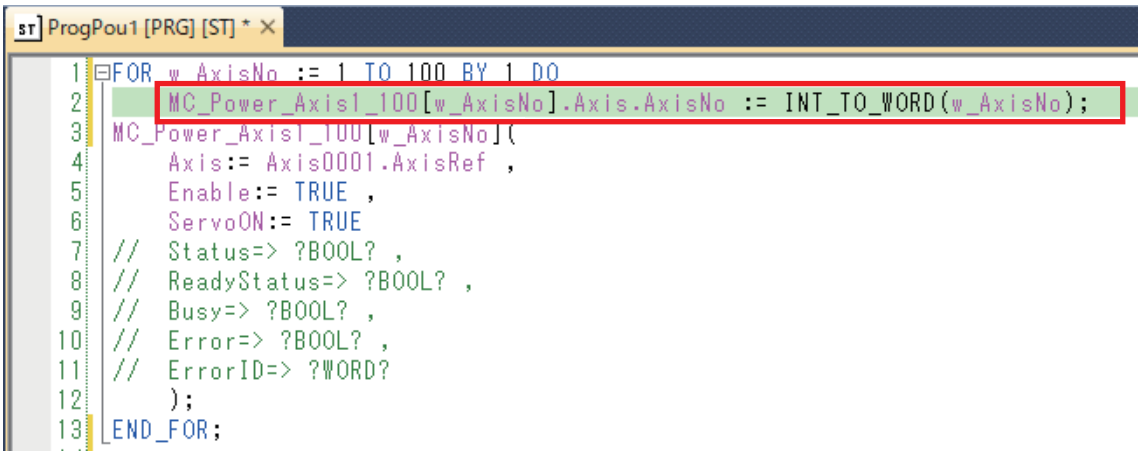


Point

FOR文の詳細については，下記のマニュアルを参照してください。
📖MELSEC MXコントローラ(MX-Rモデル) プログラミングマニュアル

4. 軸変数AxisName.AxisRef.AxisNo(軸No.)を使用して，FBインスタンス"MC_Power_Axis1_100"の入力変数"Axis"にラベル"w_AxisNo"(INT型)を直接指定する以下の文を追加します。

```
MC_Power_Axis1_100[w_AxisNo].Axis.AxisNo := INT_TO_WORD(w_AxisNo);
```

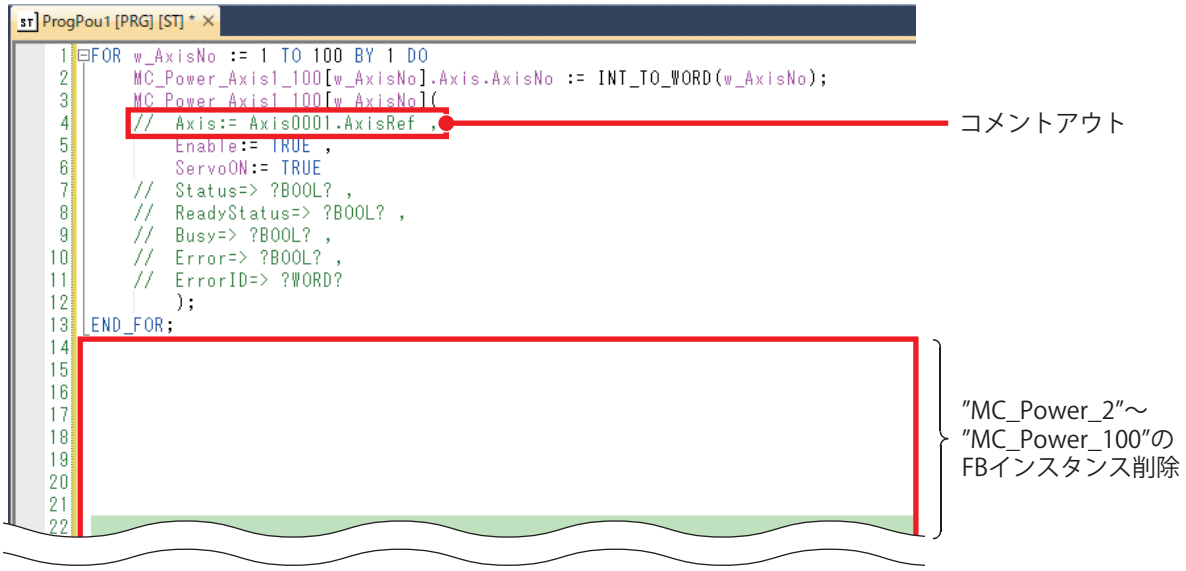


Point

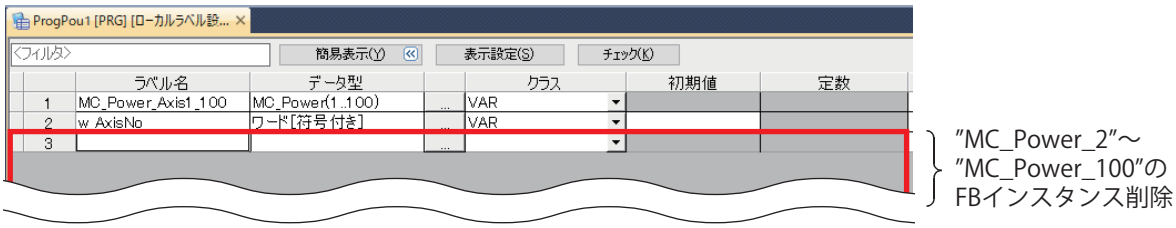
軸変数AxisName.AxisRef.AxisNo(軸No.)のデータ型はWORD(UINT)です。そのためAxisName.AxisRef.AxisNoには，WORD(UINT)型のデータを指定します。

5. FB実行文内の入力変数"Axis"を設定している文(例では4行目)をコメントアウトまたは削除します。
また不要なFBインスタンス("MC_Power_2"～"MC_Power_100")をプログラムとローカルラベルから削除します。

・STプログラム



・ローカルラベル



Point

FBを記述する際，使用しない，または初期値から変更しない入出力変数は省略できます。

2.3 モーション軸のグルーピング

概要

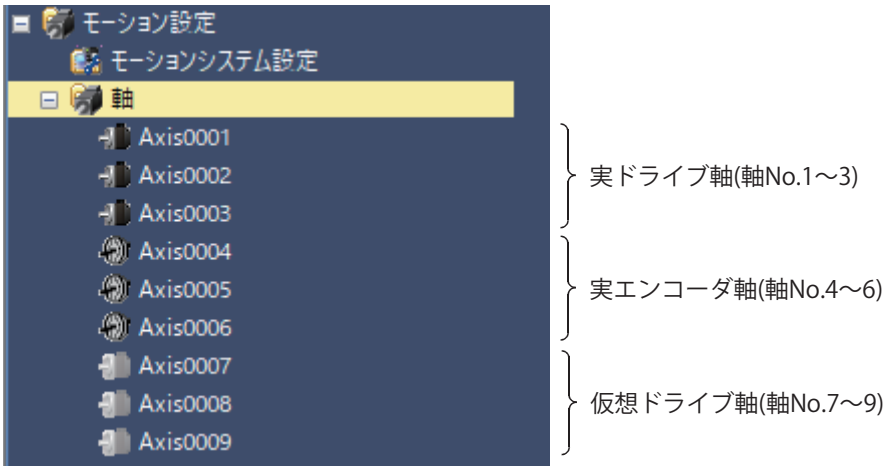
実ドライブ軸や実エンコーダ軸，仮想ドライブ軸など複数の軸種別を使用する場合，軸種別や装置ごとに軸をグループ分けしたり，軸種別ごとに軸番号の範囲や先頭軸番号を決めたりすることで，プロジェクト内で使用している軸を明確にできます。
モーション軸をグルーピングすることで，下記のようなメリットもあります。

プロジェクトの保守性の向上

軸種別や装置によって軸のフォルダを分けることで管理がしやすくなります。また軸種別ごとに軸番号の範囲や先頭軸番号を決めておくことで，設備改修によって軸の追加・削除がある場合にプログラムの修正が少なくなるため，保守性が向上します。

実行手順

下記のプロジェクトと軸番号のルールを例に，軸種別ごとにモーション軸をグルーピングする手順を示します。
・プロジェクト



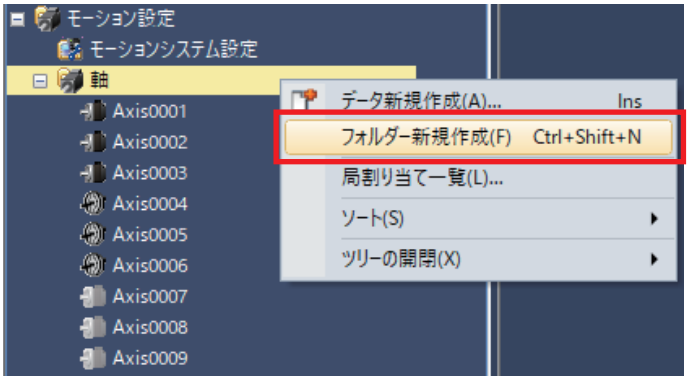
・軸番号のルール

軸種別	軸番号
実ドライブ軸	1～999
実エンコーダ軸	1000～1999
仮想ドライブ軸	2000～2999

操作手順

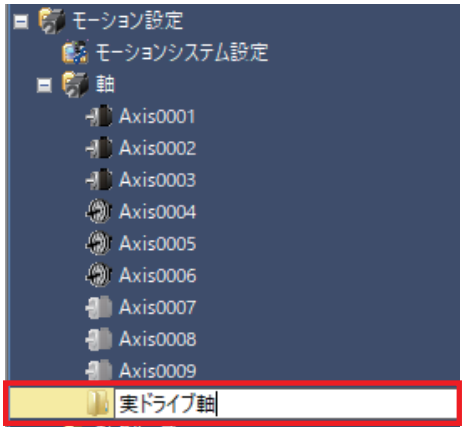
1. 軸をグループ分けするためのフォルダを新規作成します。

ナビゲーションウィンドウ⇨[モーション設定]⇨[軸]を右クリック⇨[フォルダー新規作成]

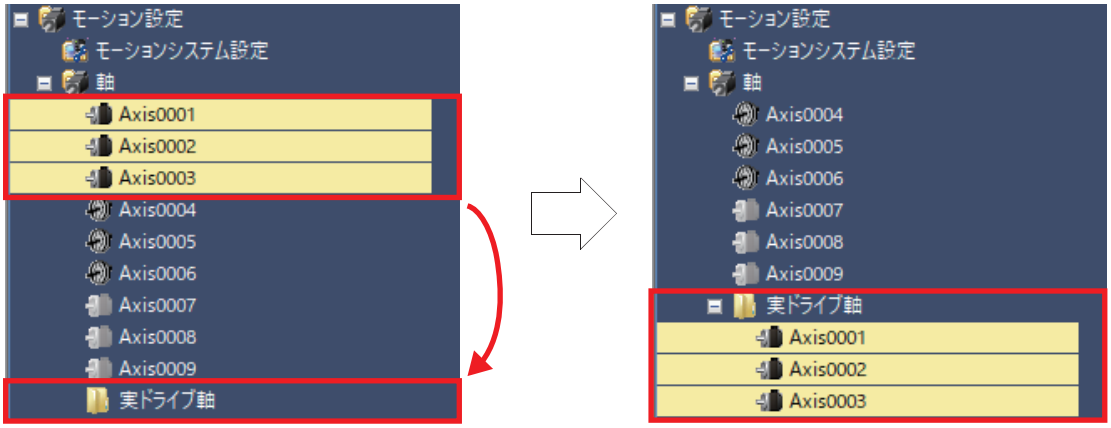


2. フォルダ名を任意の名前に設定します。

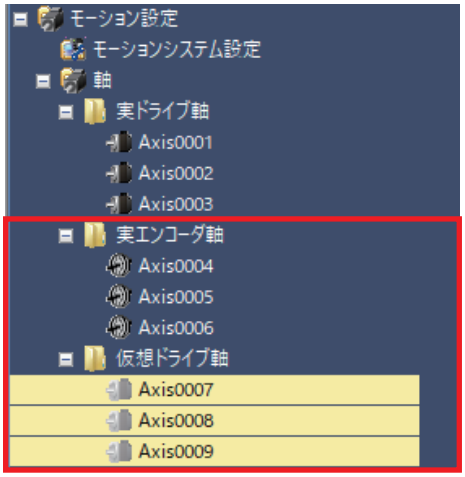
・フォルダ名: 実ドライブ軸



3. 格納する軸"Axis0001"～"Axis0003"を選択し，手順2で作成したフォルダにドラッグ&ドロップします。

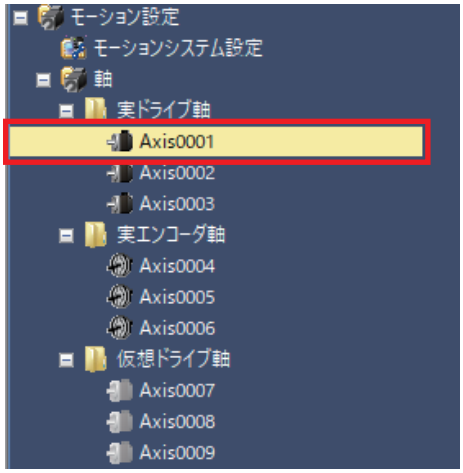


4. 手順1～3を繰り返し，実エンコーダ軸，仮想ドライブ軸も同様にグループ分けします。



5. 軸パラメータ設定を開きます。

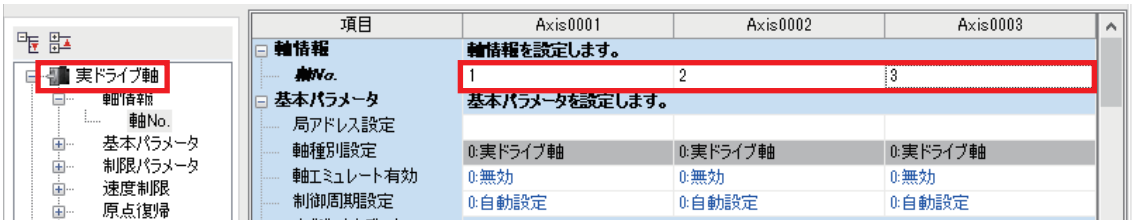
ナビゲーションウィンドウ⇨[モーション設定]⇨[軸]⇨任意の軸をダブルクリック



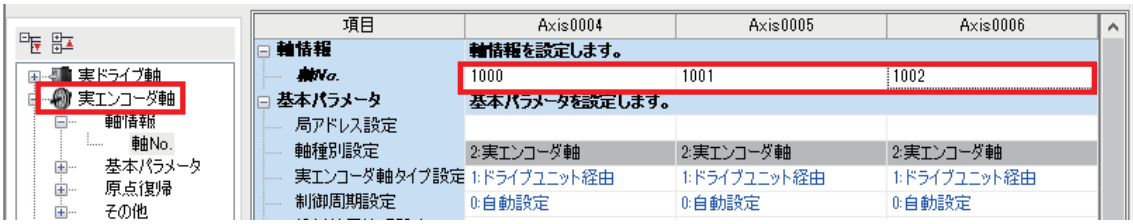
6. 軸種別ごとに軸番号を設定します。

設定する軸種別⇨[軸情報]⇨[軸No.]

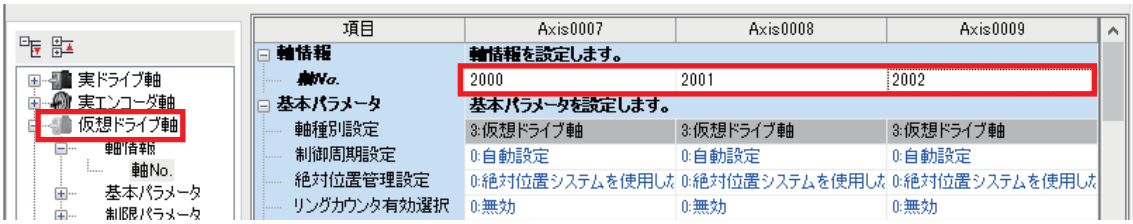
・実ドライブ軸: 軸No. 1～3



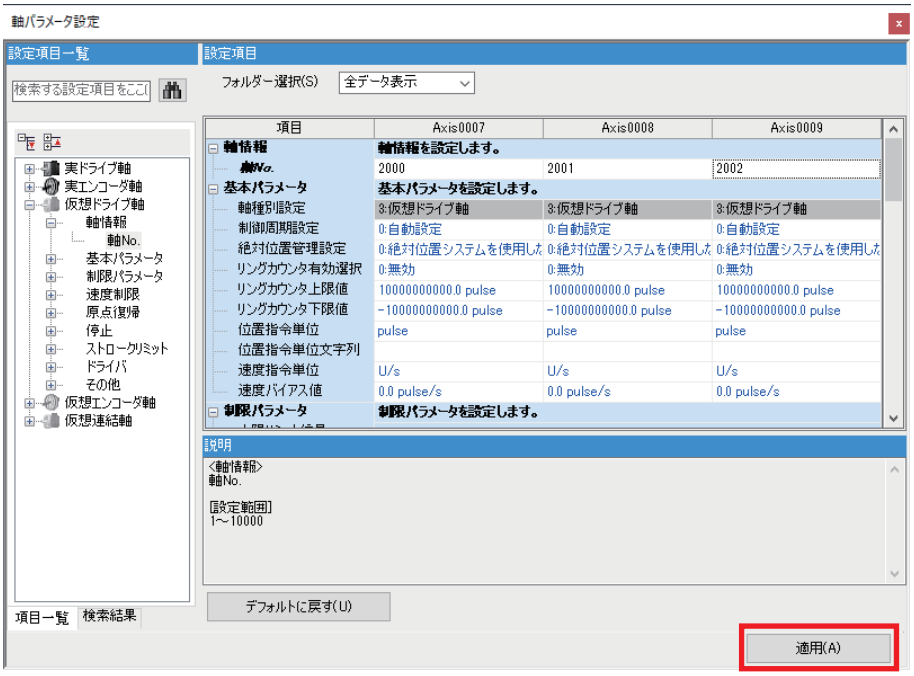
・実エンコーダ軸: 軸No. 1000～1002



・仮想ドライブ軸: 軸No. 2000～2002



7. [適用]ボタンをクリックして設定を適用します。



注意事項

各軸に局アドレス(IPアドレス)が設定されていない場合，適用時に警告が発生します。
警告が発生した場合は，各軸のパラメータ設定の[基本パラメータ]⇨[局アドレス設定]から局アドレスを設定してください。

2.4 グローバルラベルの分割

概要

グローバルラベルを工程や装置などの意味がある単位でまとめ，異なるグローバルラベルエディタへ移動させることで，単位ごとのラベルが明確になります。
グローバルラベルを分割することで，下記のようなメリットもあります。

グローバルラベルの移植性の向上

グローバルラベルを分割することで，装置や工程などの意味がある単位ごとにグローバルラベルを移植できます。

プログラム変換時間の短縮

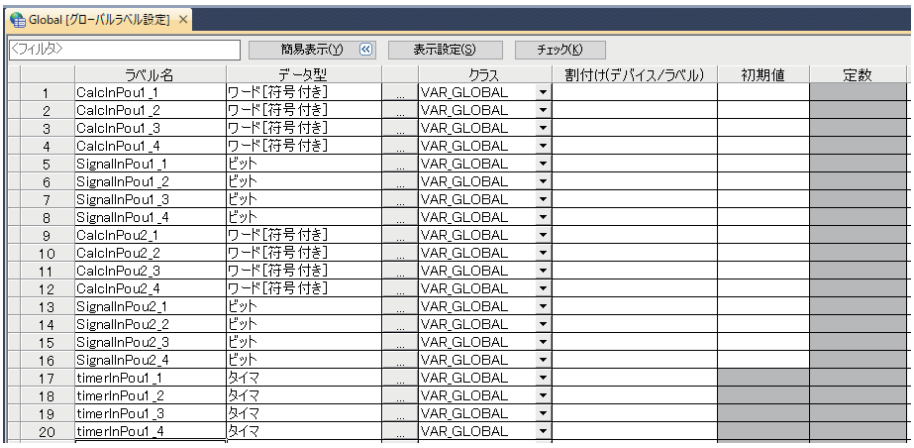
グローバルラベルを分割することでプログラムの変換範囲を局所化できるため，プログラム変換時の影響を抑えます。これにより，プログラム変換時間を短縮できます。

グローバルラベルの編集操作性の向上

装置や工程などの意味がある単位ごとにグローバルラベルを分割することで，編集時に該当のグローバルラベルを見つけやすくなり，修正もしやすくなります。

実行手順

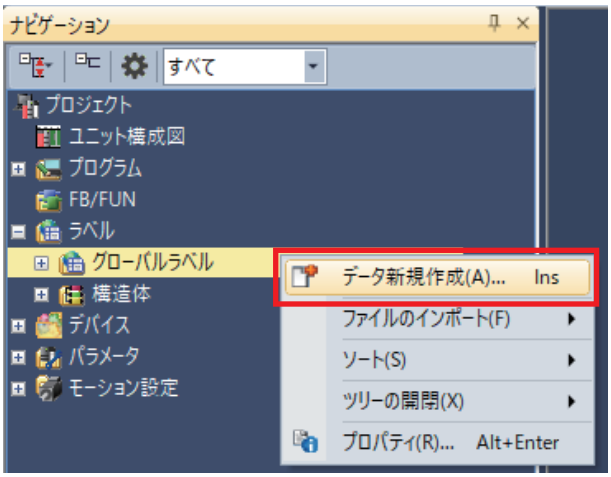
下記のグローバルラベルを例に，装置1で使用しているグローバルラベルと装置2で使用しているグローバルラベルに分割する手順を示します。



操作手順

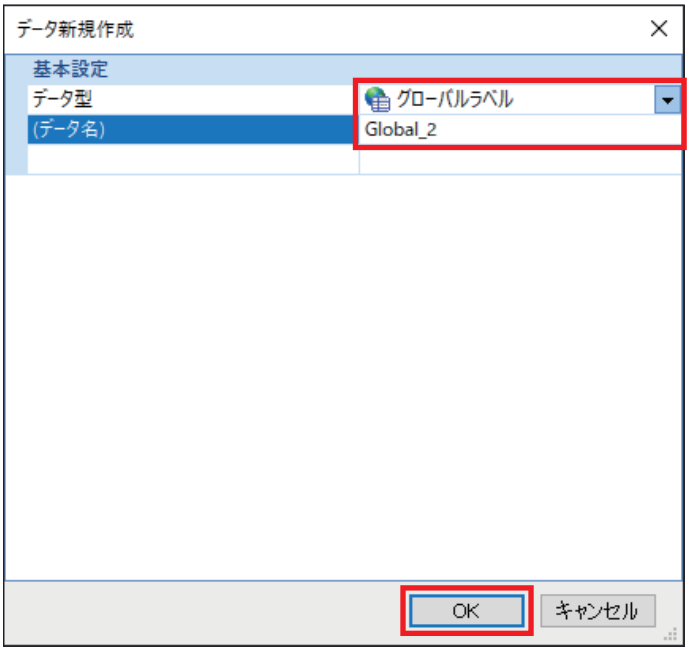
1. 分割した移動先となるグローバルラベルを新規作成します。

ナビゲーションウィンドウ⇨[ラベル]⇨[グローバルラベル]を右クリック⇨[データ新規作成]

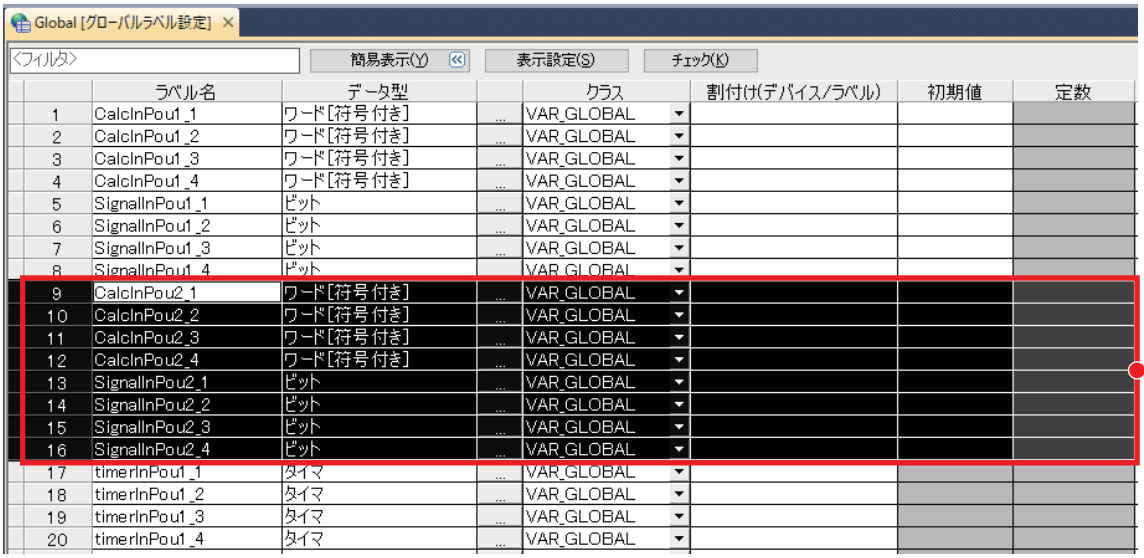


2. 各項目を設定し，[OK]ボタンをクリックします。

- データ型: グローバルラベル
- (データ名): Global_2



3. 分割前のグローバルラベルを開き，分割するグローバルラベル(本例ではNo.9～16)を選択して切り取ります。
エディタの表の行番号をクリックし，行全体を選択した状態で切り取ってください。



Point

- 行全体を選択していない場合，ラベルエディタ上で非表示になっている内容はコピーされません。
- 1つのグローバルラベルエディタ内のラベル数が，最大5000個となるよう分割することを推奨します。

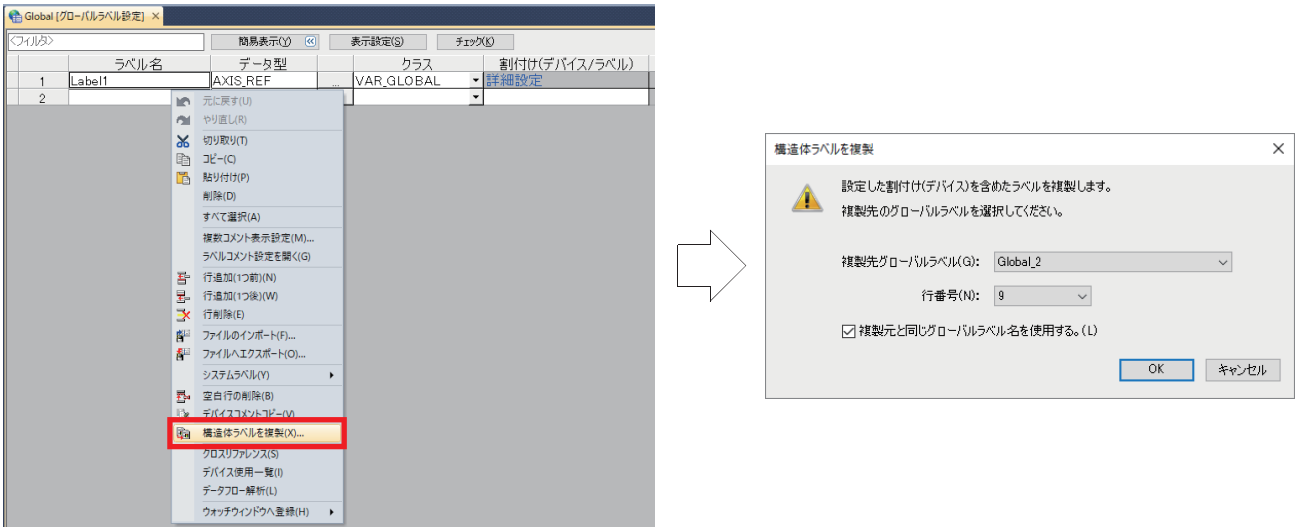
4. 手順1，2で作成したグローバルラベルを開き，切り取ったグローバルラベルを貼り付けます。



注意事項

- 分割先に移動させるグローバルラベルのデータ型が構造体の場合，本実行手順では割付けデバイスの内容がコピーされません。

該当のグローバルラベルを選択して右クリック⇨[構造体ラベルを複製]を選択し，ラベルを複製してグローバルラベルを分割してください。(複製後，分割前のグローバルラベルは削除してください)



3 プログラム変換の高速化

プログラムを分割することやFBで部品化することにより，プログラムの変換範囲を局所化できるため，プログラム変換を高速化できます。
本章では，プログラム変換の高速化を行うためのテクニックについて示します。

テクニック一覧

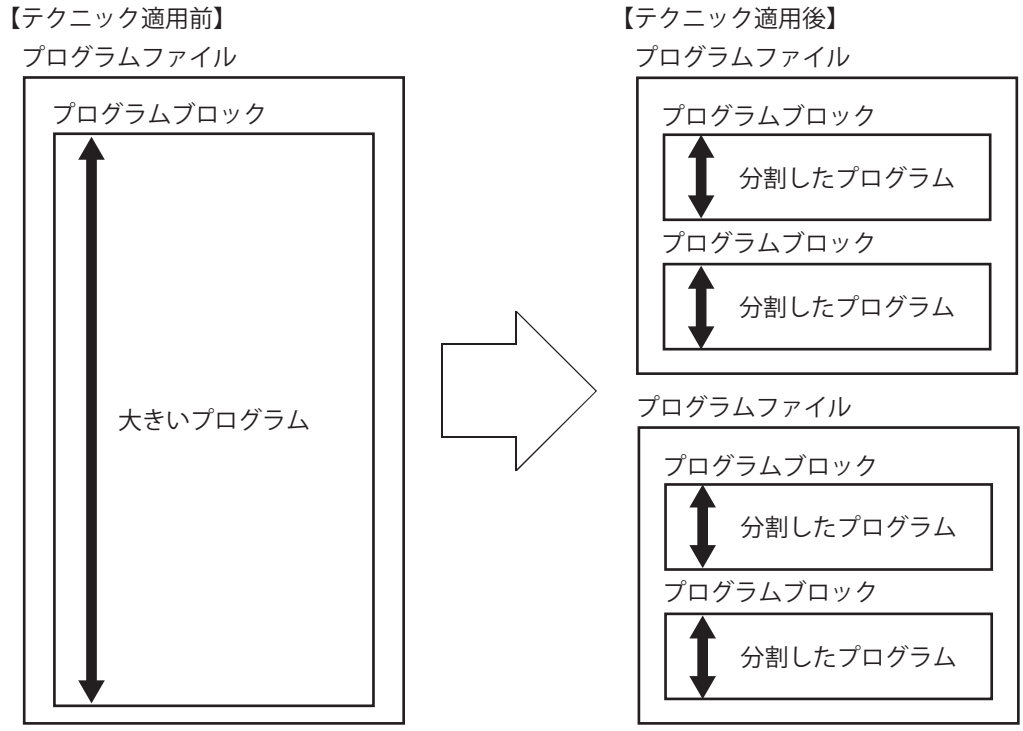
本章で説明するテクニック一覧を下記に示します。

項目	内容	参照
プログラムファイルの分割	プログラムファイル，プログラムブロックを意味のある単位で分割し，異なるプログラムファイルへ移動させます。	☞ 28ページ プログラムファイルの分割
プログラムのFB化	複数箇所にある類似プログラムやサブルーチンプログラムなど，ポインタや意味のある単位で区切られたプログラムをFB化します。	☞ 37ページ プログラムのFB化
制御構文を含むプログラムの短縮	サブルーチンや制御構文内にある大きなプログラムをFBに置き換えます。 または，構文外にプログラムを移動させます。	☞ 42ページ 制御構文を含むプログラムの短縮
グローバルラベル(構造体)の設定変更	構造体から不要なデバイスの割付を削除します。 またデバイスを割り付けた構造体をそのままFBの引数に渡さず，必要な構造体メンバのみを指定するように修正します。	☞ 45ページ グローバルラベル(構造体)の設定変更
配列同一要素への連続参照の削減	同じ配列要素へのアクセスを複数回行っている場合，一度だけアクセスするように修正します。	☞ 50ページ 配列同一要素への連続参照の削減

3.1 プログラムファイルの分割

概要

プログラムファイル，プログラムブロックを装置や工程，目的などの意味がある単位で分割することで，プログラムを構造化・標準化します。これによりプログラム変換の対象が局所化するため，プログラム変換時間を短縮できます。
分割したプログラムファイルはパラメータから実行順を設定して，分割前と同じ順番でプログラムを動作させることができます。



プログラムファイルを分割することで，下記のようなメリットもあります。

プログラムの可読性の向上

装置の処理や工程単位ごとの処理が明確になるため，プログラムの処理の流れや構造を見通し良くできます。

プログラムの移植性・再利用性の向上

装置の処理や工程単位ごとのプログラム本体とローカルラベルを持つプログラムファイルを作成できるため，処理や工程単位のプログラムの移植や再利用がしやすくなります。これにより，プログラムの開発効率も向上します。

プログラムのステップ数と変換速度の相関

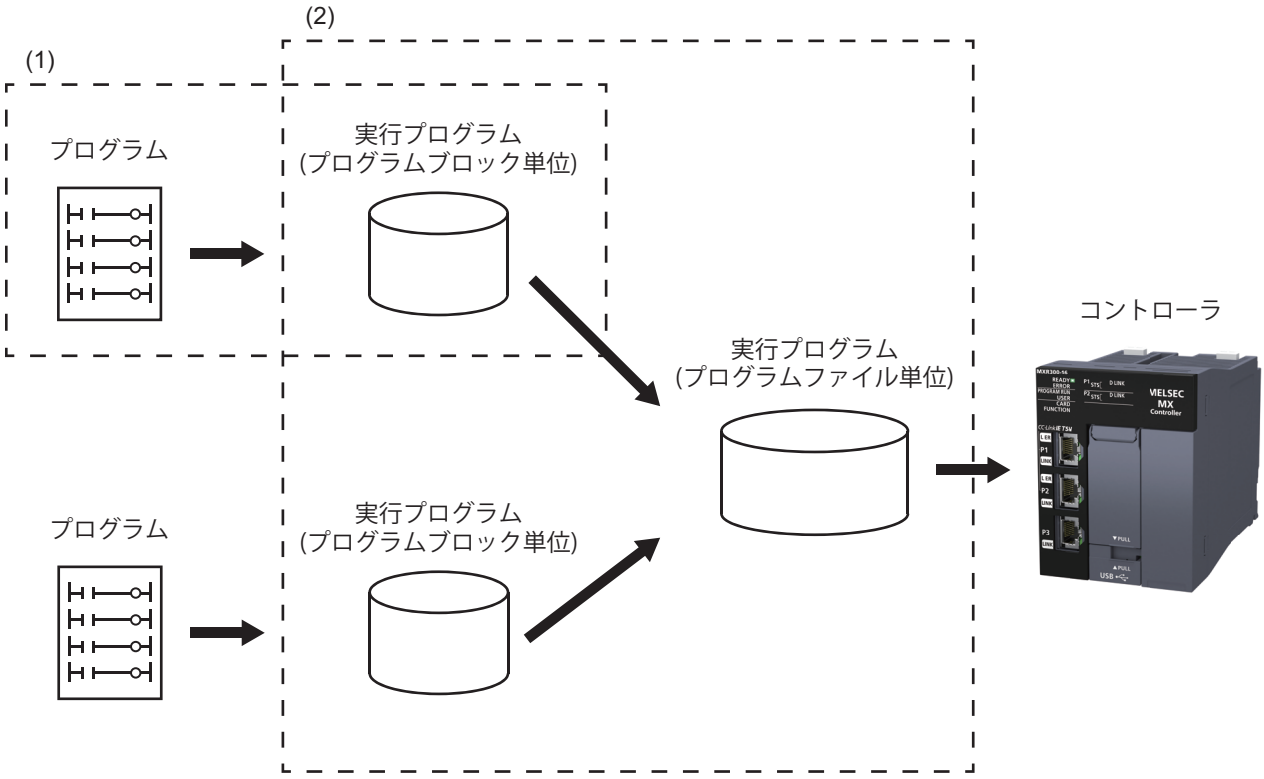
プログラムのステップ数と変換速度の相関例を下記に示します。
プログラムファイルやプログラムブロックを分割する際の目安としてください。

プログラムのステップ数 *1*2	変換速度(秒)*3*4		
	分割なし (1つのプログラムファイル内に1つのプログラムブロックが存在)	1つのプログラムファイル内でプログラムブロックを4つに分割	2つのプログラムファイル内でプログラムブロックをそれぞれ2つずつに分割
1000	4.2	4.2	4.2
10000	11.2	10.2	9.2
100000	125.2	83.1	79.7

*1 ステップ数分の接点とコイルのみが存在するシンプルなプログラムでの計測値です。
*2 プロジェクト構成やプログラム内容など様々な要因により変換速度が変動します。特にポインタを使用したプログラムの場合は，変換に時間を要する場合があるためご注意ください。
*3 Intel® Core™ i7-10610U(1.80GHz)搭載のパソコンでの計測値です。
*4 全変換を実行し，表示されるダイアログにて"変換後実行プログラムを作成"を選択した場合の変換速度です。

プログラム変換の流れと本テクニックの関連

プログラムを変換する流れと本テクニックの関連について下記に示します。



(1) 記述されたプログラムから，プログラムブロック単位の実行プログラムを生成する処理
(2) プログラムブロック単位の実行プログラムをプログラムファイル単位の実行プログラムに結合する処理

プログラムブロックを複数に分割した場合，(1)の工程で実行プログラムの生成が並列化されるため，「全変換・変換・変換+RUN中書込み」の時間を短縮します。
またプログラムファイルを複数に分割した場合，(2)の工程でシーケンサに書き込むファイル自体のサイズが小さくなるため，変更点のみを変換する「変換・変換+RUN中書込み」の時間を短縮します。

実行手順

下記のプログラムを例に，工程ごとにプログラムファイルを分割する手順を示します。

• プログラム



• ローカルラベル

モーション [PRG] [ローカルラベル設...

〈フィルタ〉　簡易表示(Y)　表示設定(S)　チェック(K)

	ラベル名	データ型		クラス	初期値	定数
1	Axis1_JogFw	ビット	...	VAR		
2	Axis1_JogBw	ビット	...	VAR		
3	Axis2_JogFw	ビット	...	VAR		
4	Axis2_JogBw	ビット	...	VAR		
5	Axis3_JogFw	ビット	...	VAR		
6	Axis3_JogBw	ビット	...	VAR		
7	Axis1_Homing	ビット	...	VAR		
8	Axis2_Homing	ビット	...	VAR		
9	Axis3_Homing	ビット	...	VAR		

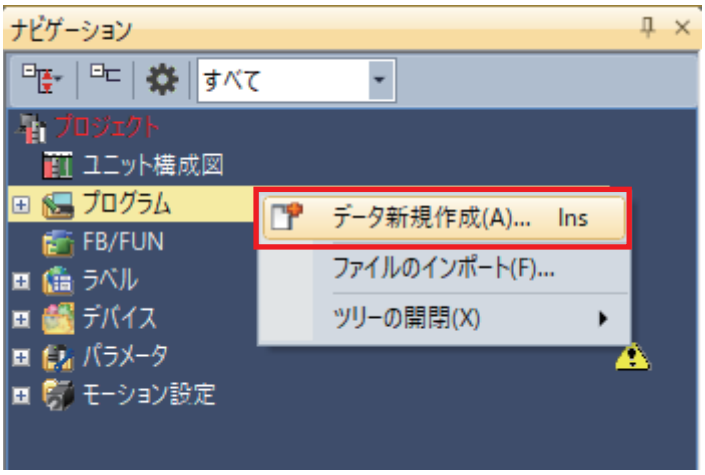
工程1で使用

工程2で使用

操作手順

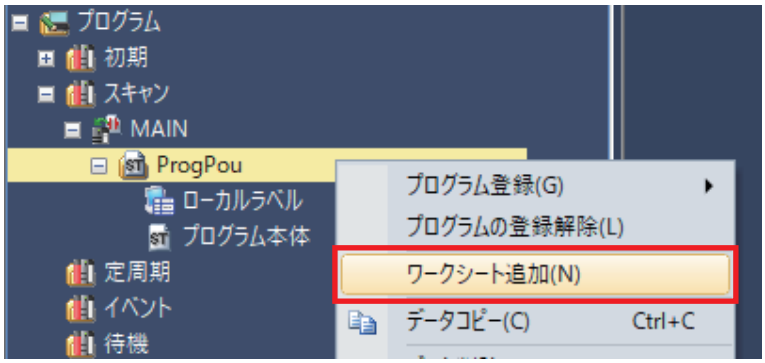
1. 分割したプログラムの移動先となるプログラムファイルを新規作成します。

ナビゲーションウィンドウ⇨[プログラム]を右クリック⇨[データ新規作成]



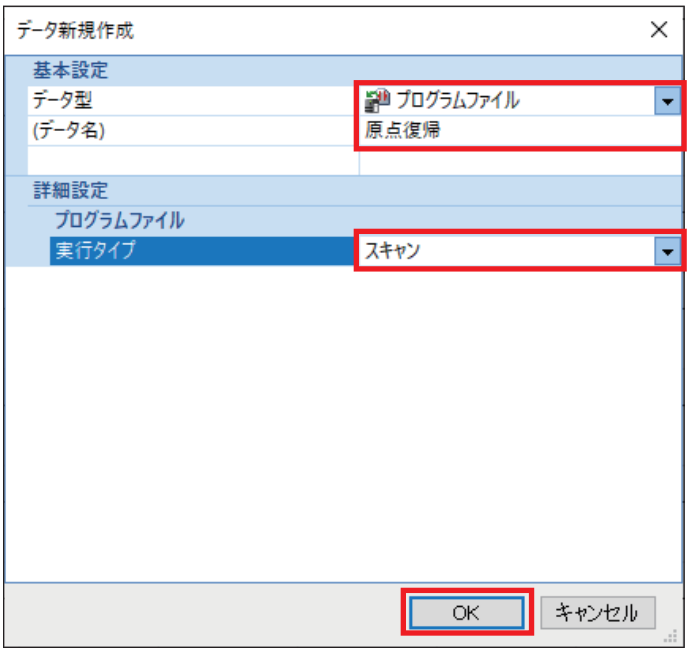
Point

- プログラム変換時間は，1プログラムファイルあたりのプログラム量が少ないほど短縮されます。そのため，分割したプログラムは新規のプログラムファイルに移動させることを推奨します。(既存のプログラムファイルへ移動させる場合でも，動作に問題はありません)
- STプログラムやFBDプログラムの場合は，同じPOU内に新規ワークシートを追加して分割することも可能です。これにより，プログラムの可読性/移植性が向上します。（プログラム変換時間の短縮の効果はありません）

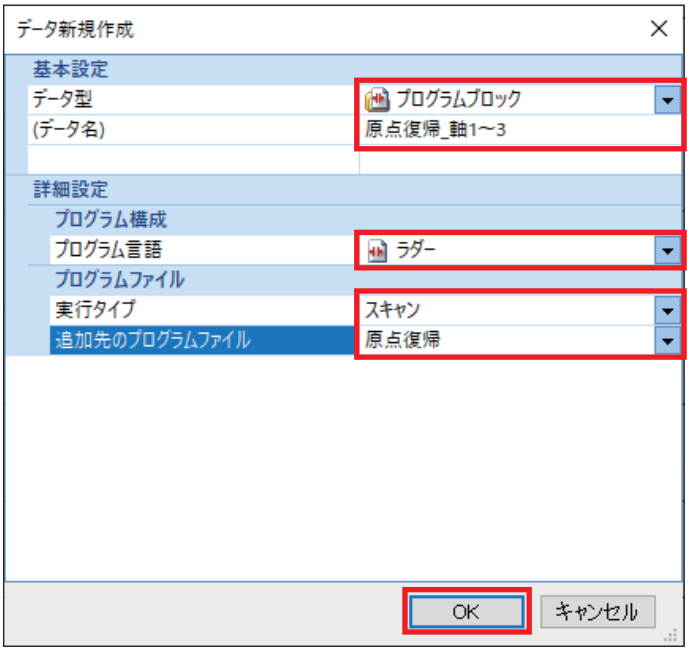


2. 各項目を設定し，[OK]ボタンをクリックします。

- データ型: プログラムファイル
- (データ名): 原点復帰
- 実行タイプ: スキャン

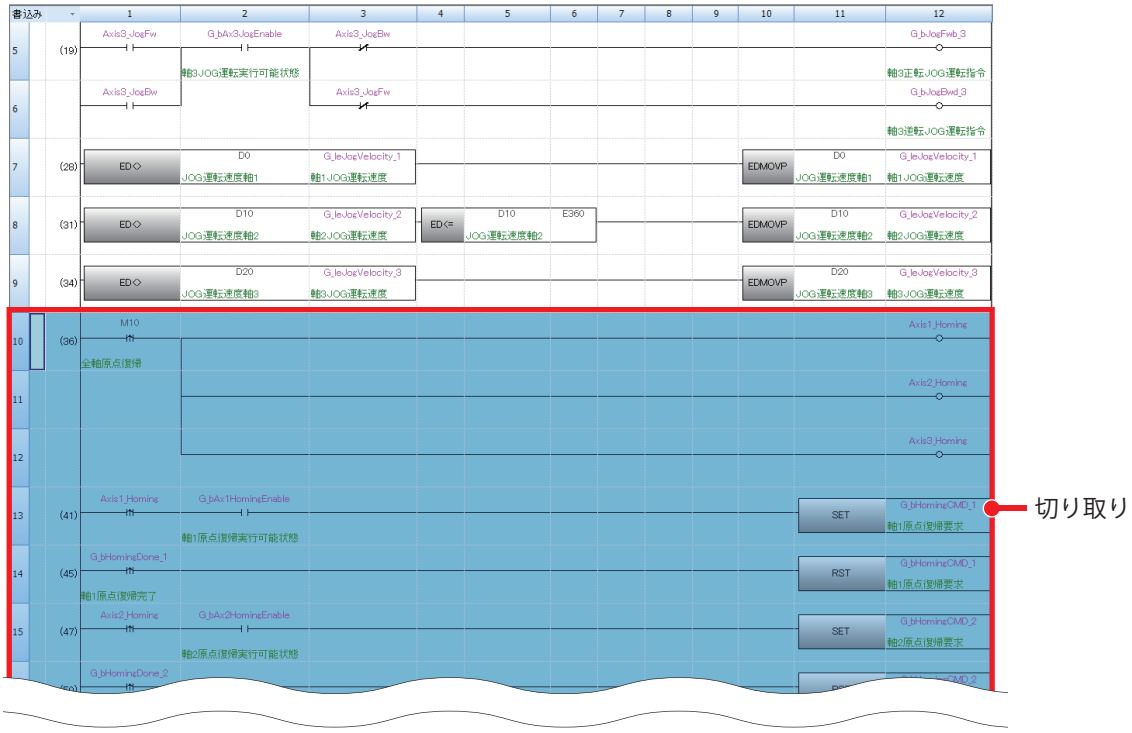


3. 手順1と同様の手順で「データ新規作成」画面を開き，プログラムブロックを新規作成します。
各項目を設定し，[OK]ボタンをクリックします。



項目	設定内容	備考
データ型	プログラムブロック	—
(データ名)	原点復帰_軸1～3	—
プログラム言語	ラダー	—
実行タイプ	スキャン	手順2で作成したプログラムファイルと同じ実行タイプを設定します。
追加先のプログラムファイル	原点復帰	手順2で作成したプログラムファイルを設定します。

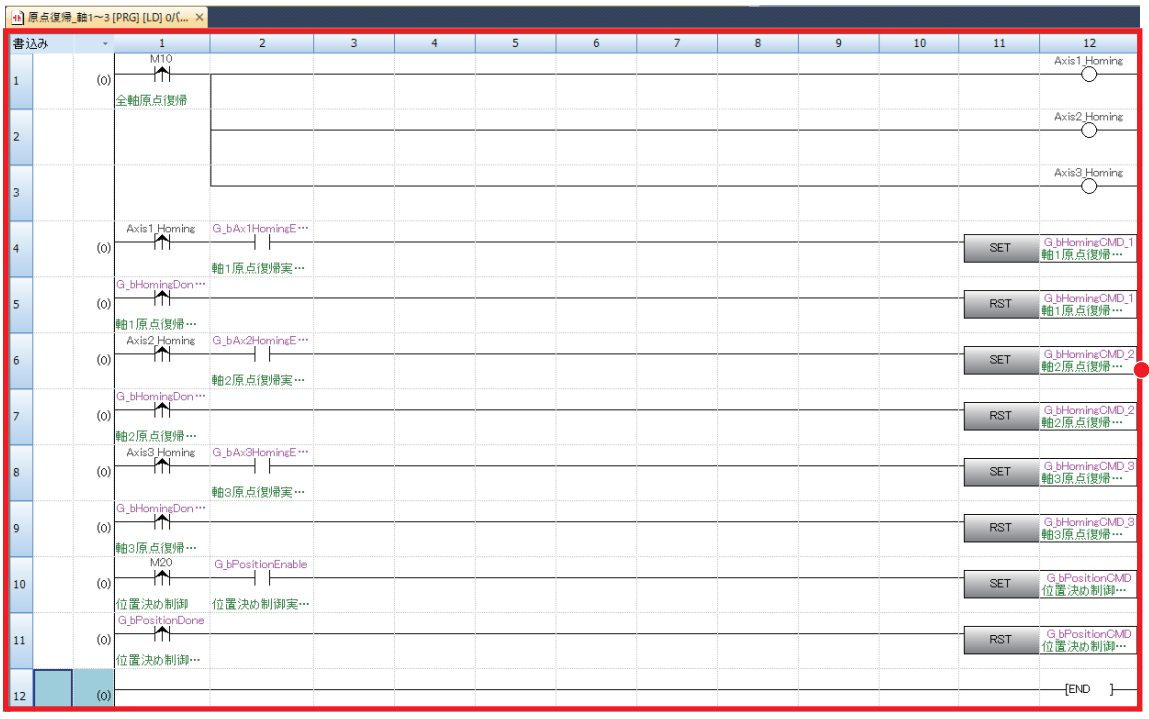
4. 分割するプログラムファイルのプログラム本体を開いて分割する位置(例では工程2の先頭)を決め，以降のプログラムをすべて選択して切り取ります。



Point

分割前のプログラム中に使用されているローカルラベルとローカルデバイスが，分割後のどちらかのプログラムでのみ使用されるように分割してください。このように分割することが難しい場合は，プログラムをFB化することを推奨します。
プログラムのFB化については，下記を参照してください。
☞ 37ページ プログラムのFB化

5. 手順3で作成したプログラムブロックのプログラム本体を開き，切り取ったプログラムを貼り付けます。



Point

分割したプログラムブロックより実行順が後ろのプログラムブロックがある場合は，新規プログラムファイルへ該当のプログラムブロックごと移動させてください。

6. 分割元のプログラムファイルのローカルラベルを開き，分割先のプログラムで使用しているローカルラベル(本例ではNo.7～9)を切り取ります。

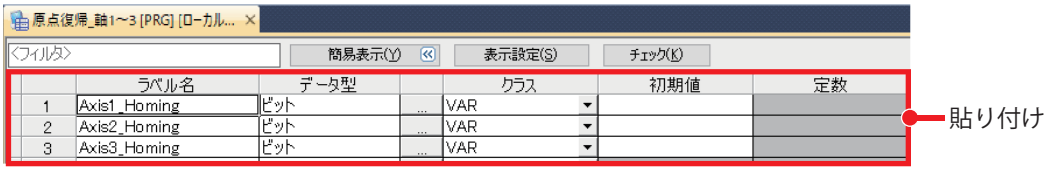
エディタの表の行番号をクリックし，行全体を選択した状態で切り取ってください。



Point

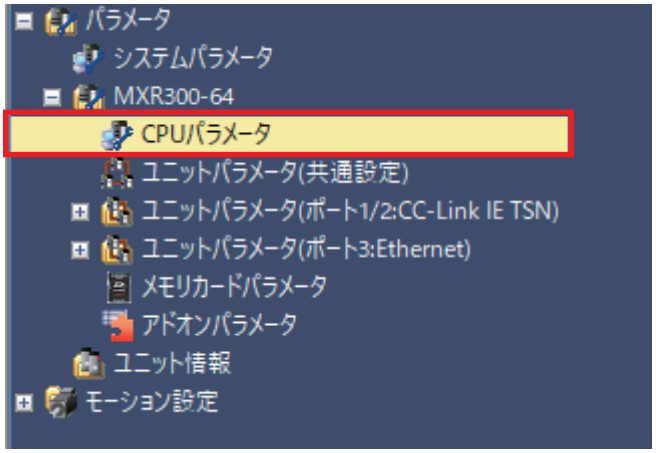
行全体を選択していない場合，ラベルエディタ上で非表示になっている内容はコピーされません。

7. 手順3で作成したプログラムブロックのローカルラベルを開き，切り取ったローカルラベルを貼り付けます。



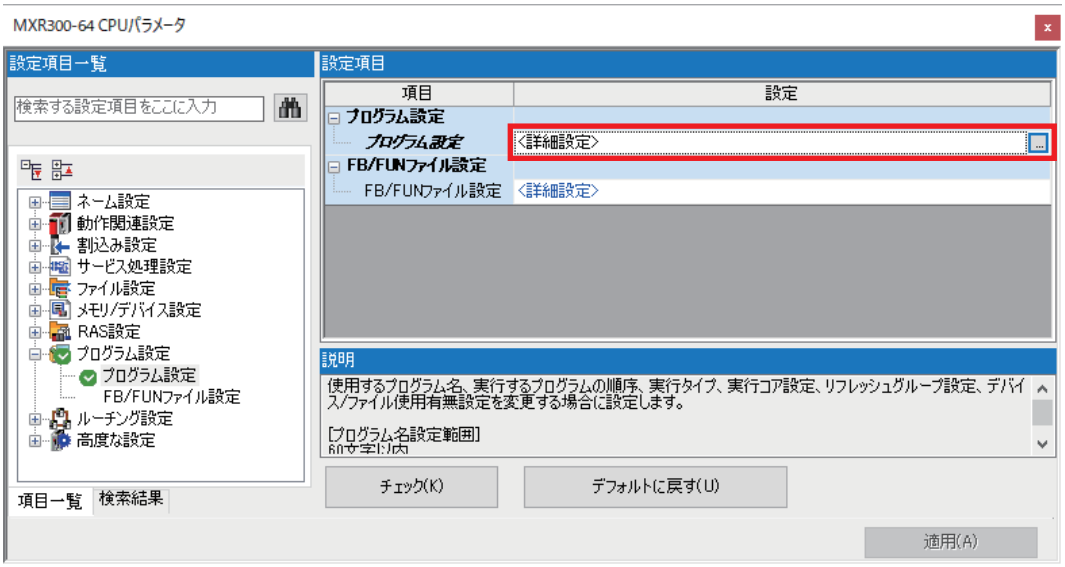
8. CPUパラメータを開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[CPUパラメータ]をダブルクリック



9. プログラムの実行順を設定するため，プログラム設定を開きます。

[プログラム設定]⇨[プログラム設定]の<詳細設定>をダブルクリック

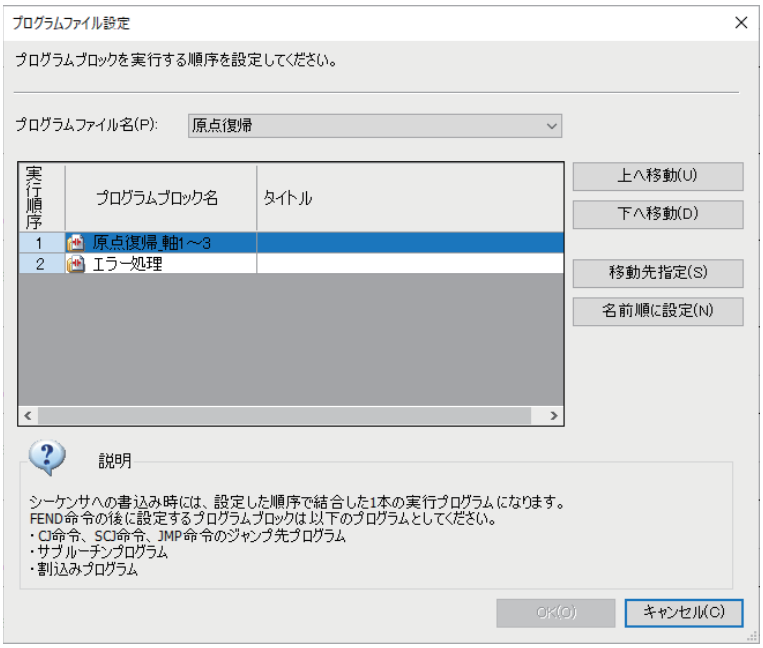


10. プログラムを実行順に並び替え，[適用]ボタンをクリックして設定を適用します。



Point

プログラムブロックの実行順を変更する場合は，「プログラムファイル設定」画面からプログラムブロックの実行順を設定できます。
詳細については，下記のマニュアルを参照してください。
📖GX Works3 オペレーティングマニュアル



注意事項

プログラムファイルの分割によって，ポインタを使用している命令のジャンプ元とジャンプ先が異なるプログラムファイルになる場合や，サブルーチンや制御構文などを構成する要素が同じプログラム内に存在しない場合，変換エラーが発生します。

3.2 プログラムのFB化

概要

複数個所にある類似プログラムやポインタで区切られたサブルーチンプログラムなど，意味のある単位で区切られたプログラムをFB化することでプログラム変換対象の範囲を局所化できるため，プログラム変換時の影響を抑えます。
またポインタを使用しているプログラムの場合は，ポインタをFBに置き換えることによりプログラムの変換対象を構造化できるため，プログラム変換時間を短縮できます。
ラダープログラムの場合は，既存の回路からFBを作成できます。生成したFBはFB化する前にプログラムがあった位置から呼び出し，必要なデバイス・ラベルを引数として与えることで，FB化前と同じプログラム動作を実現できます。
プログラムのFB化により，下記のようなメリットもあります。

プログラムの可読性の向上

プログラムをFB化することで，“箱”(FB)と入力，出力だけのシンプルなプログラムにすることができるため，読みやすいプログラムを作成できます。

プログラムの移植性・再利用性の向上

標準的なプログラムをFB化することで，プログラムの移植を行う際に，プログラムをコピーしてデバイスを修正する操作が不要になります。またFB化したプログラムをライブラリとして保存すると，各プロジェクトで共通のライブラリを使用できるため，プログラムの再利用性が向上します。

プログラム品質の向上

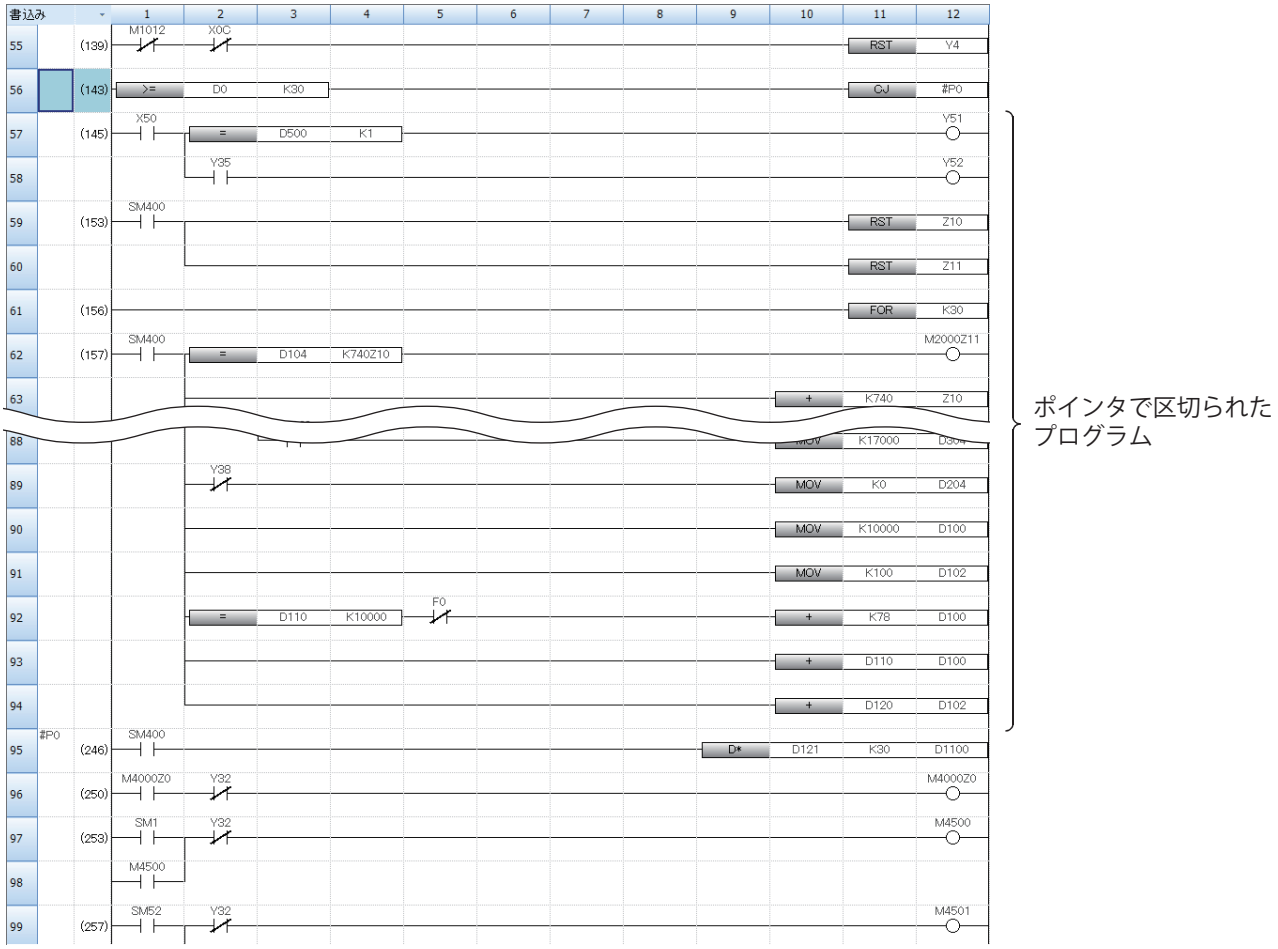
プログラムをFBとして部品化し再利用することで，開発者が異なる場合でも共通処理に同一のFBを使用できるため，プログラム開発者の技術レベルに依存しない均一な品質のプログラムを開発できます。

Point

ラダープログラム以外のプログラム(ST，FBD/LD)の場合は，FB化するプログラムとローカルラベルを切り取り，新規FBに貼り付ける方法でFBを作成できます。
FBの新規作成については，下記のマニュアルを参照してください。
GX Works3 オペレーティングマニュアル

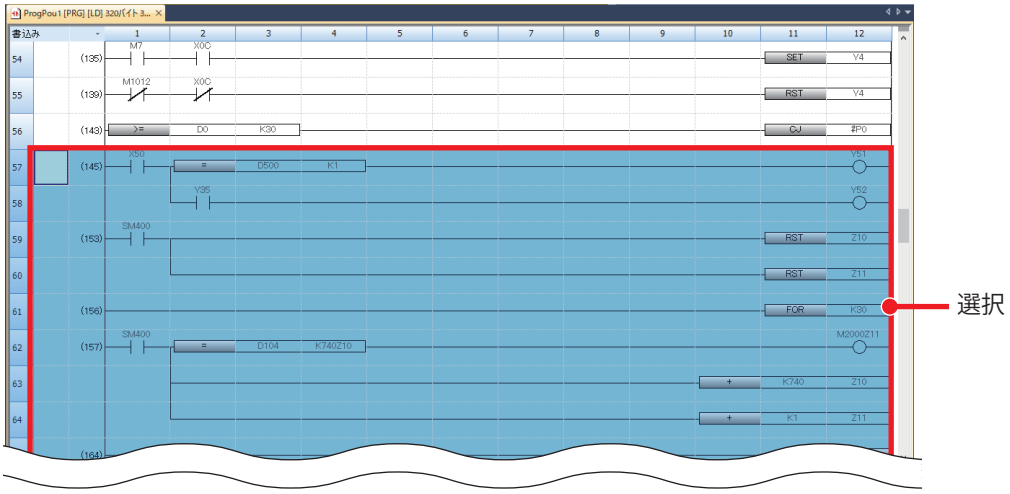
実行手順

下記のラダープログラムを例に，ポインタで区切られたプログラムをFB化する手順を示します。



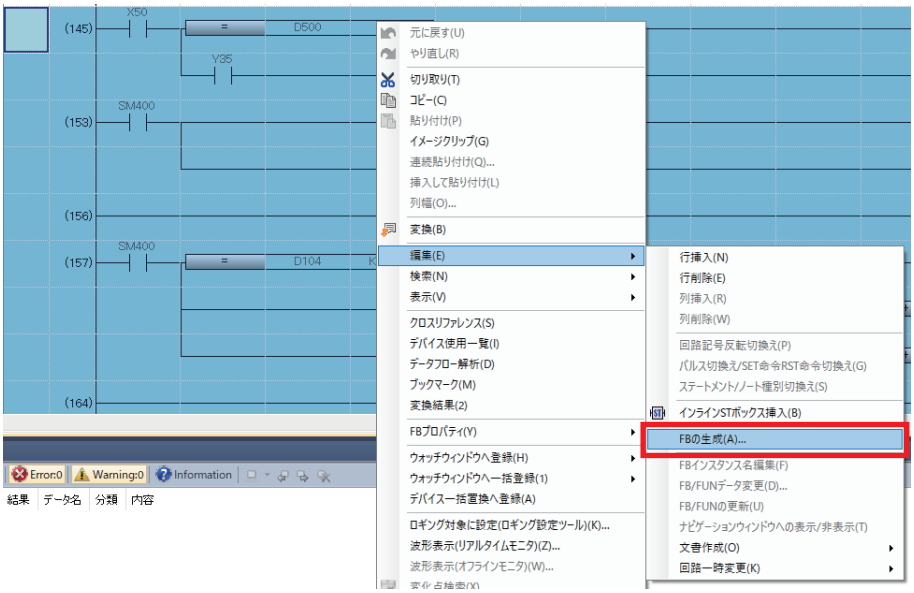
操作手順

1. FB化するプログラムを開き，FB化する範囲を選択します。

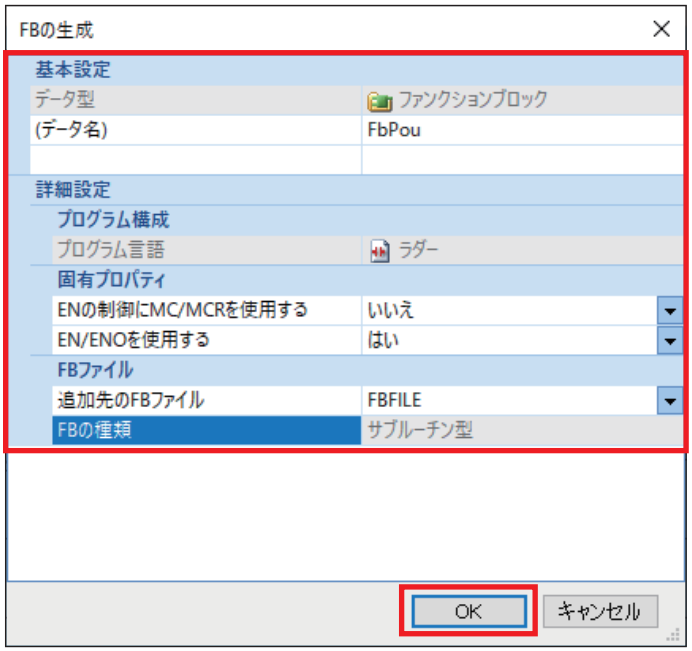


2. 選択した範囲からFBを生成します。

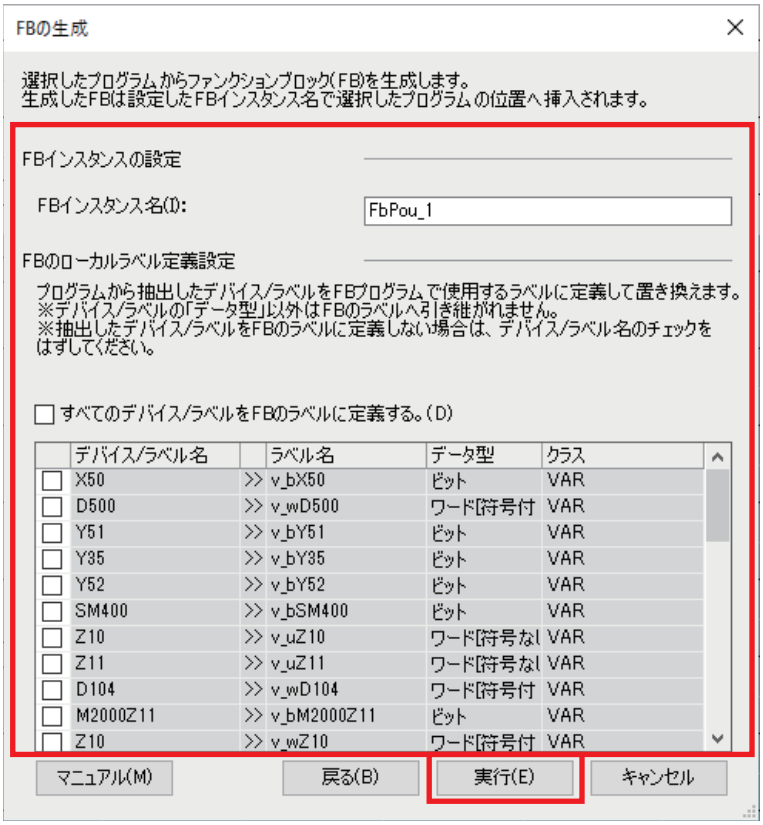
右クリック⇨[編集]⇨[FBの生成]



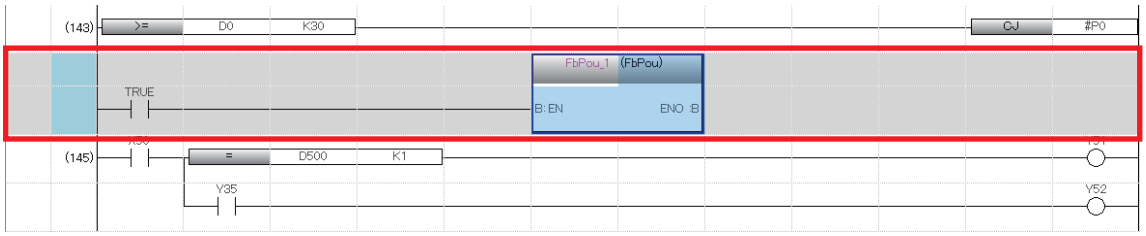
3. プログラムの内容に合わせて生成するFBの各項目を設定し，[OK]ボタンをクリックします。
例では，"EN/ENOを使用する"を"はい"に設定します。



4. FBのインスタンス名やローカルラベル定義を設定し，[実行]ボタンをクリックします。
例では"すべてのデバイス/ラベルをFBのラベルに定義する。"のチェックを外します。



FB化の範囲の先頭位置に作成したFBが挿入されます。



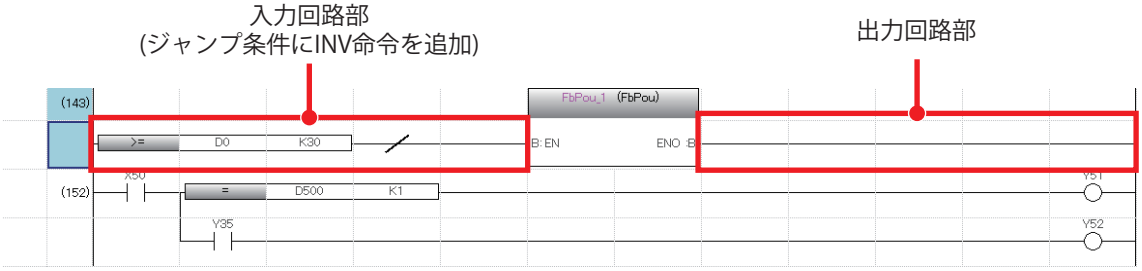
注意事項

FBのローカルラベル定義を設定する場合は，以下の点に注意してください。

- ローカルデバイスは，FBの引数に設定できません。"FBのローカルラベル定義設定"からデバイス/ラベル名のチェックを外してください。
- FBプログラム内と外で同じローカルデバイス/ローカルラベルを使用している場合は，ローカルデバイス/ローカルラベルをFBのラベル(引数)として定義してください。

FBのローカルラベル定義設定の詳細については，下記のマニュアルを参照してください。
📖GX Works3 オペレーティングマニュアル

5. FBの入力ラベルや出力ラベルに合わせて入力/出力回路部を作成し，FB化する前と等価なプログラムに修正します。
本例の場合，ポインタのジャンプ条件にINV命令を追加してFBの実行条件に修正します。



注意事項

- プログラムのFB化によって，ポインタを使用している命令のジャンプ元とジャンプ先が同じプログラム内に存在しない場合や，サブルーチンや制御構文などを構成する要素が同じプログラム内に存在しない場合，変換エラーが発生します。
- FBの引数で受け渡すデータ量が多くなると，スキャンタイムが遅くなる場合があります。

3.3 制御構文を含むプログラムの短縮

概要

ラダープログラムにおけるFOR～NEXT命令やサブルーチン，STプログラムにおけるIF文やFOR文などの制御構文内に大きなプログラムがある場合，プログラムをFB化することや，制御構文内に記載する必要がないプログラムを構文外へ移動させることで，プログラムを短縮します。これによりプログラム量が減少するため，プログラム変換時間を短縮できます。
プログラムの種類ごとの短縮可能な制御構文を下記に示します。

種類	制御構文
ラダープログラム	• FOR～NEXT命令 • サブルーチンプログラムなど，ポインタで区切られた回路ブロック群
STプログラム	• IF文 • CASE文 • FOR文 • WHILE文 • REPEAT文
FBD/LDプログラム	• FOR～NEXT命令 • ジャンプラベルで区切られた回路群

プログラムを短縮することで，下記のようなメリットもあります。

プログラムの可読性の向上

プログラム量の減少によってプログラムの可読性が向上します。

Point

プログラムのFB化の概要やメリットについては，下記を参照してください。
☞ 37ページ 概要

制御構文内のプログラムのステップ数と変換速度の相関

制御構文内のプログラムのステップ数と変換速度の相関の例を下記に示します。
プログラムを構文外へ移動させる際の目安としてください。

プログラムのステップ数[*1*2]	変換速度(秒)[*3*4]	
	制御構文内のプログラム[*5]	制御構文外のプログラム
1000	5.6	4.2
10000	19.7	11.2
100000	447.5	125.2

*1 ステップ数分の接点とコイルのみが存在するシンプルなプログラムでの計測値です。
*2 プロジェクト構成やプログラム内容など様々な要因により変換速度が変動します。特にポインタを使用したプログラムの場合は，変換に時間を要する場合があるためご注意ください。
*3 Intel® Core™ i7-10610U(1.80GHz)搭載のパソコンでの計測値です。
*4 全変換を実行し，表示されるダイアログにて"変換後実行プログラムを作成"を選択した場合の変換速度です。
*5 FOR～NEXT命令内にプログラムを記述したプログラムでの計測値です。

実行手順

プログラムを短縮する手順について示します。

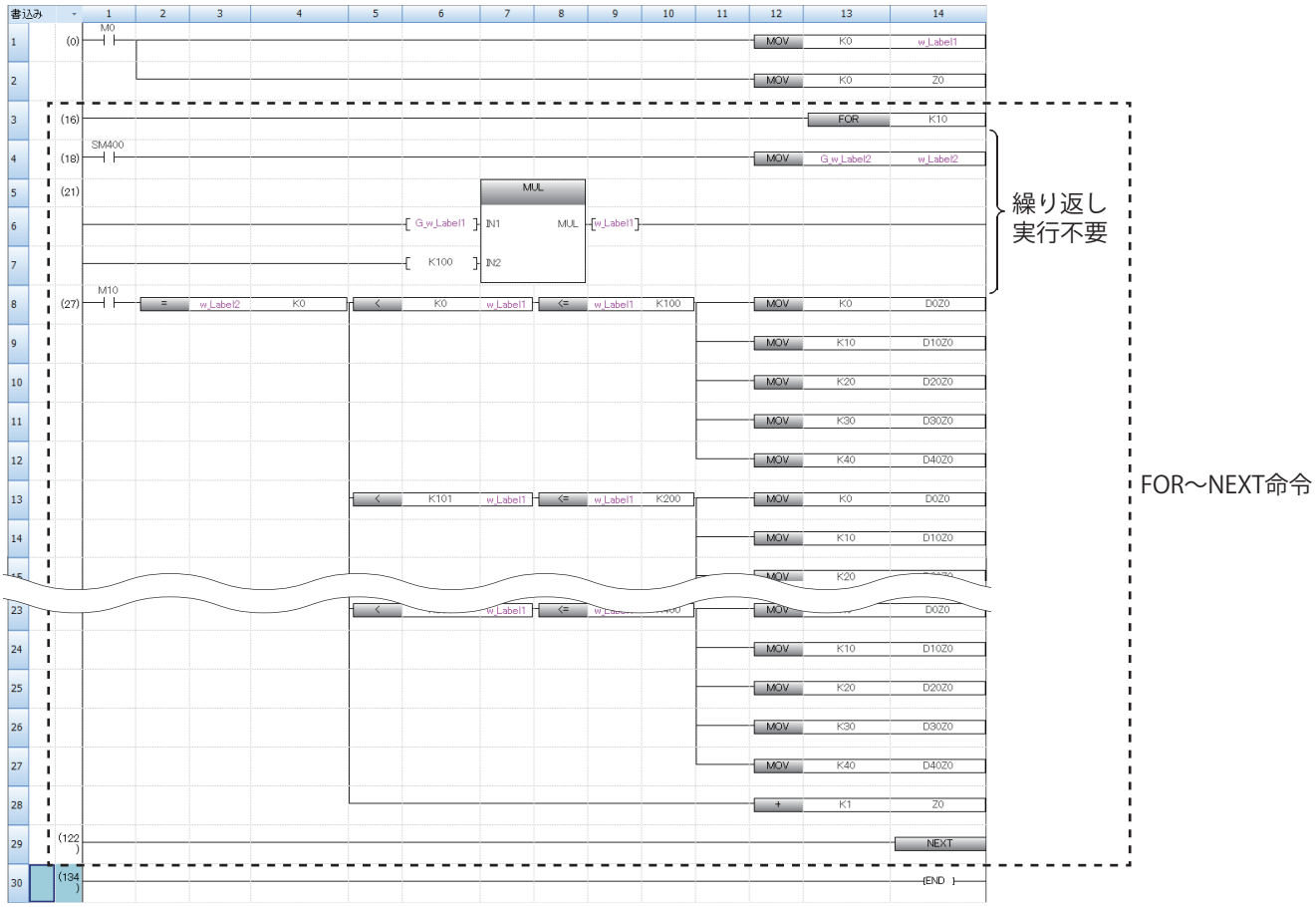
制御構文内のプログラムのFB化

制御構文内のプログラムのFB化については，下記の手順を参照してください。

☞ 38ページ 実行手順

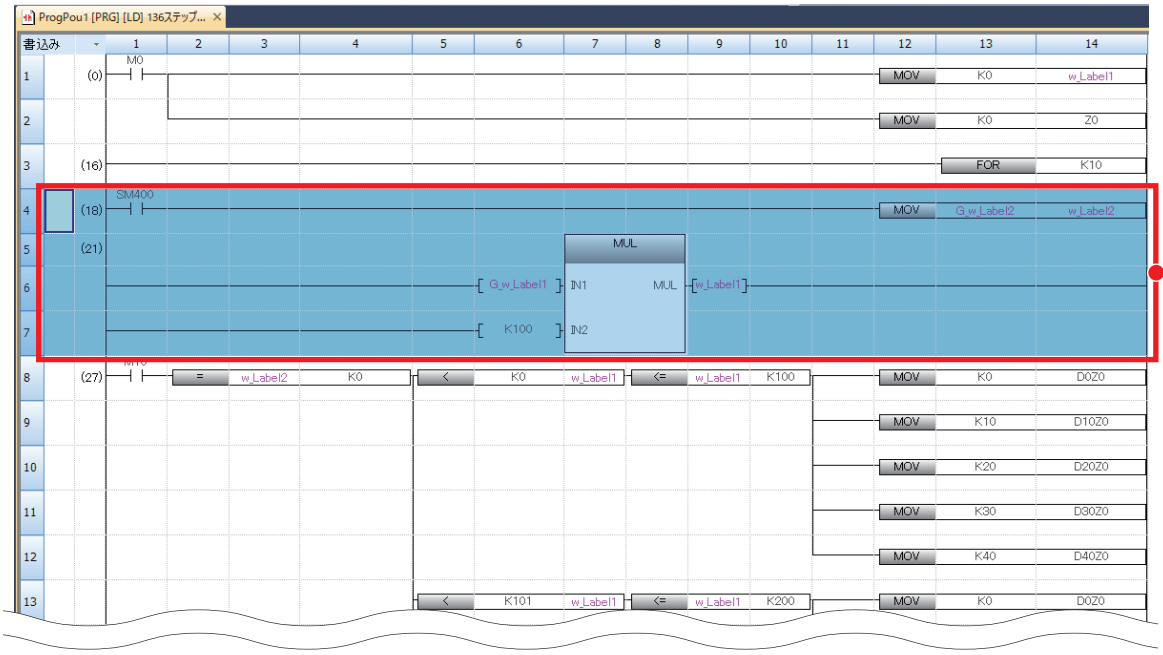
制御構文内の不要なプログラムの移動

下記のプログラムを例に，FOR～NEXT命令のプログラムで，繰り返し実行する必要がないプログラムをFOR～NEXT命令外へ移動させる手順を示します。

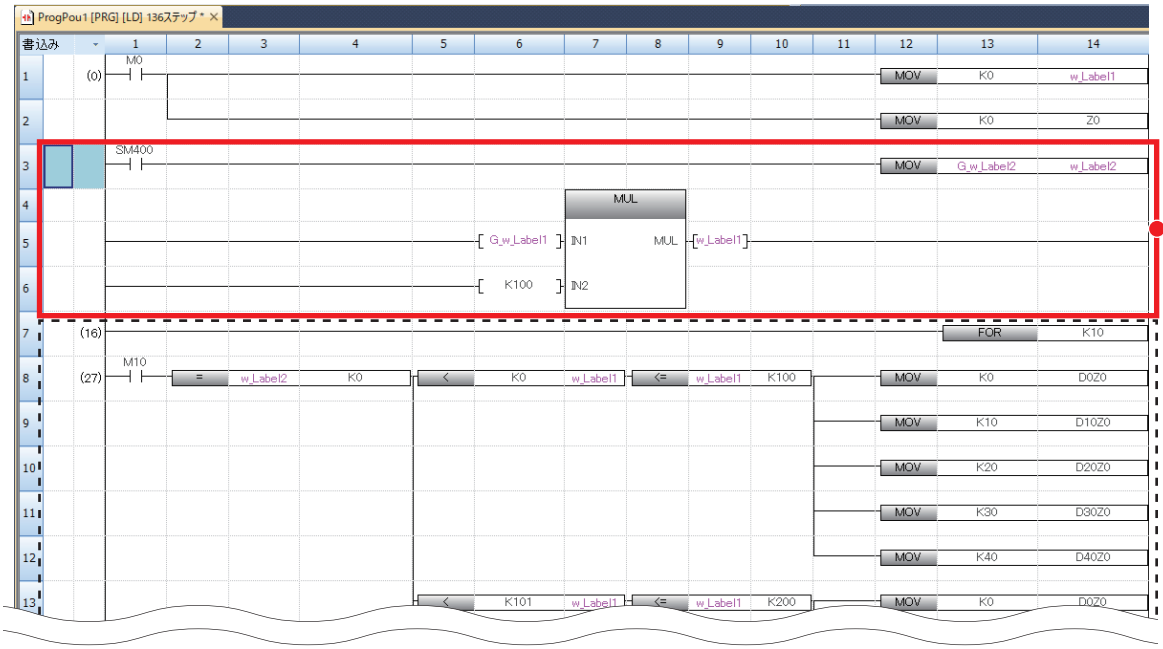


操作手順

1. プログラムを開き，FOR～NEXT命令内の繰り返し実行不要なプログラムを選択して切り取ります。



2. 切り取ったプログラムを制御構文外に貼り付けます。実行順に応じて正しい位置に貼り付けてください。



3.4 グローバルラベル(構造体)の設定変更

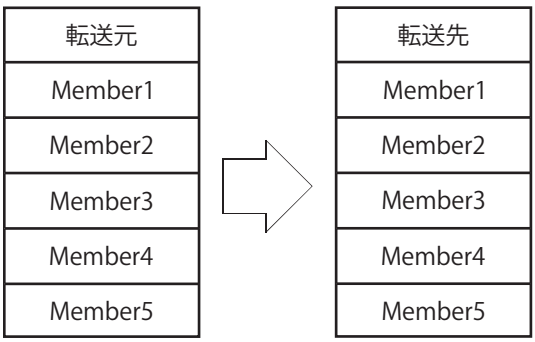
概要

デバイスを割り付けた構造体を以下の引数や代入文に使用する場合，構造体のメンバごとに値が転送されるため，内部的に多くの転送命令が生成されます。

- サブルーチン型FBの引数
- MOVE命令の引数
- STプログラムまたはFBDプログラムでの代入文

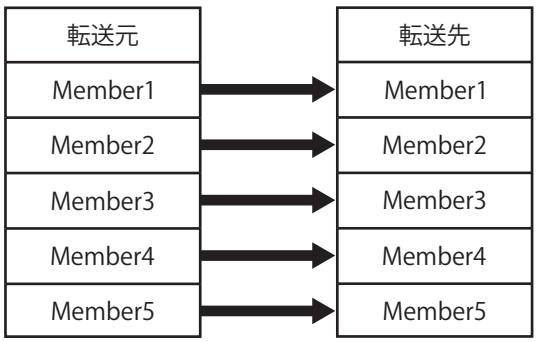
MOVE命令の引数に，デバイス割付なしの構造体とデバイス割付ありの構造体を使用した場合の転送例を以下に示します。

<デバイス割付なしの場合>



全メンバを一括転送
(ステップ数: 小)

<デバイス割付ありの場合>



メンバごとに転送
(ステップ数: 大)

転送命令は，構造体のインスタンスに対する不要なデバイス割付けを削除することで生成を抑制できます。またサブルーチン型FBの引数の場合は，必要な構造体メンバのみを引数に使用することで転送命令の生成を抑制します。内部的に生成される命令の削減により，プログラム変換時間を短縮できます。

構造体を修正することで，下記のようなメリットもあります。

プログラムサイズの縮小

構造体を修正することで内部的な転送命令の生成を抑制するため，プログラムサイズを縮小できます。

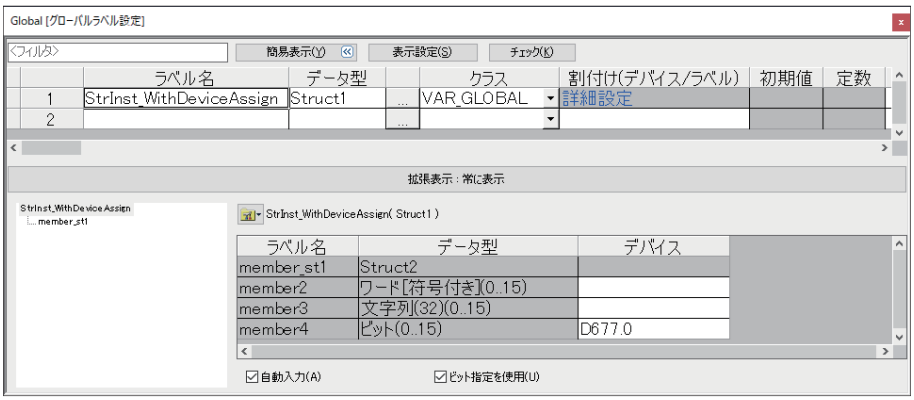
実行手順

下記のプログラムを例に，構造体"Struct1"のインスタンス"StrInst_WithDeviceAssign"に割り付いているデバイスを削除する手順を示します。また，入力引数が"StrInst_WithDeviceAssign"のサブルーチン型FB"FBPou_1"を使用しているプログラムを修正する手順を示します。

• 構造体"Struct1"

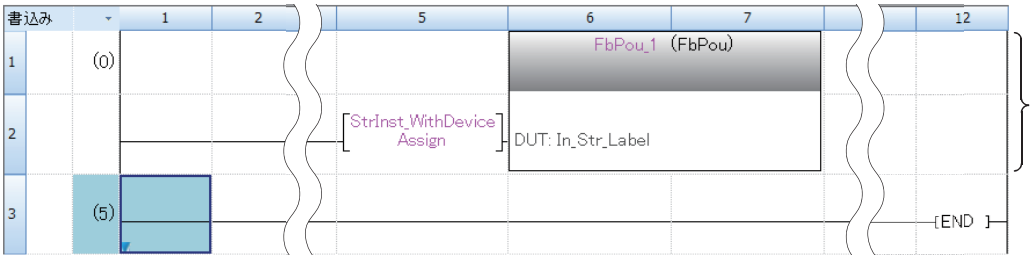


• 構造体のインスタンス"StrInst_WithDeviceAssign"(グローバルラベル)



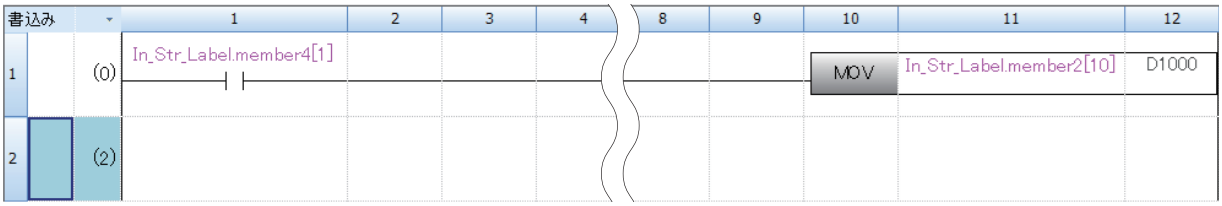
メンバ"member4"に
デバイス(D677.0)を割り付け

• サブルーチン型FB"FBPou_1"を使用しているプログラム



FBの入力引数に
構造体のインスタンスを
指定

• サブルーチン型FB"FBPou_1"のプログラム



• サブルーチン型FB"FBPou_1"のローカルラベル

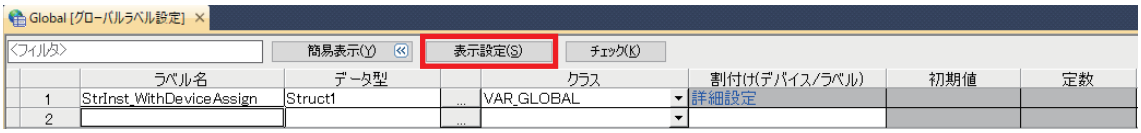


構造体のインスタンスに対する不要なデバイス割付の削除

構造体のインスタンス"StrInst_WithDeviceAssign"に対する不要なデバイス割付を削除する手順を示します。

操作手順

1. 構造体のインスタンス"StrInst_WithDeviceAssign"を登録しているグローバルラベルを開き，[表示設定]ボタンをクリックします。

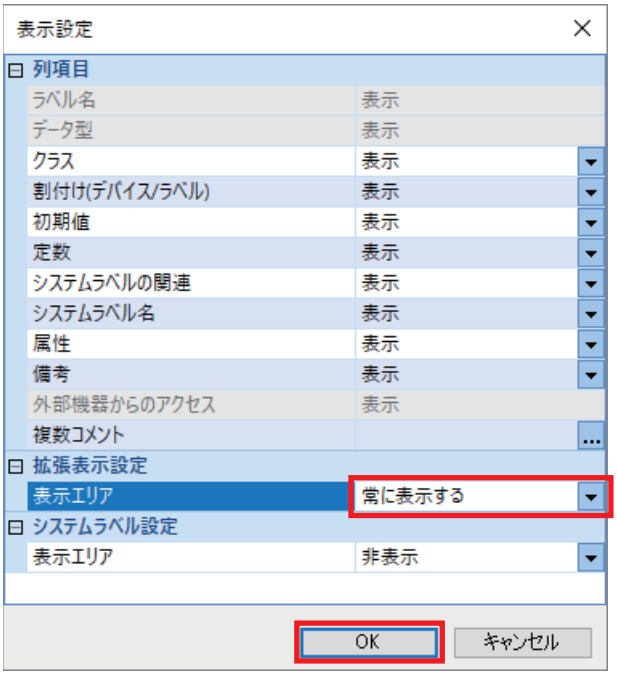


Point

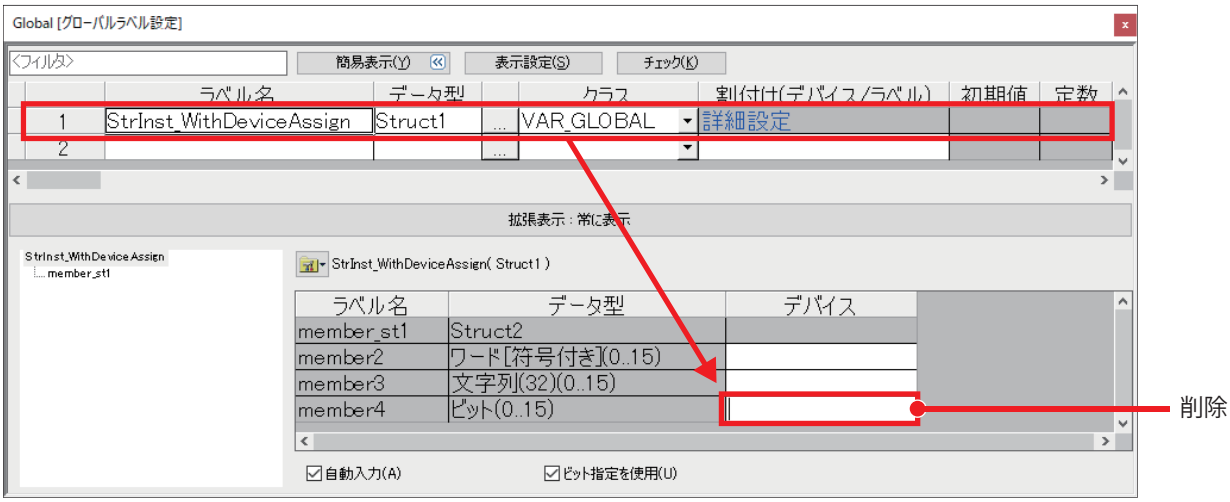
デバイスが割り当てられている構造体のインスタンスは，"割付け(デバイス/ラベル)"列の「詳細設定」が青字で表示されます。



2. [拡張表示設定]⇨[表示エリア]を"常に表示する"に変更して，[OK]ボタンをクリックします。



3. デバイスが割り当てられている構造体のインスタンスを選択して，拡張表示の”デバイス”列からmember4の割り付けデバイス”D677.0”を削除します。

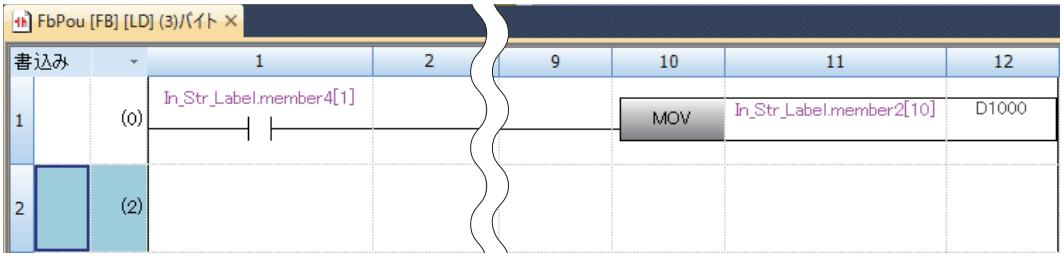


サブルーチン型FBの引数の修正

サブルーチン型FB”FBPou_1”の入力引数”StrInst_WithDeviceAssign”を必要な構造体メンバのみに修正する手順を示します。

操作手順

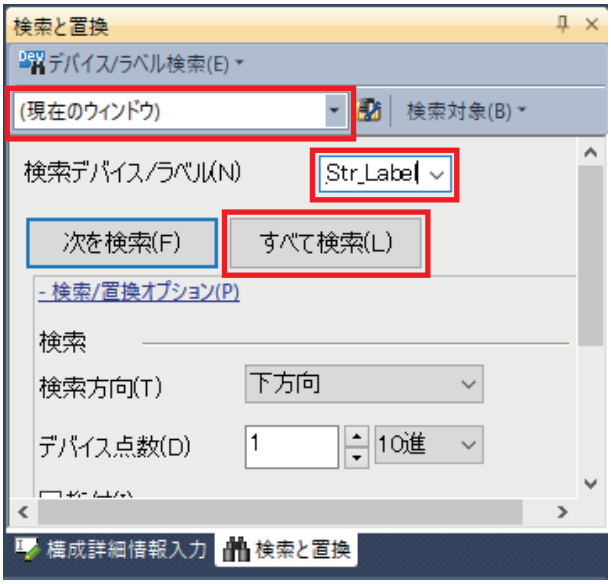
1. サブルーチン型FB”FBPou_1”のプログラムを開きます。



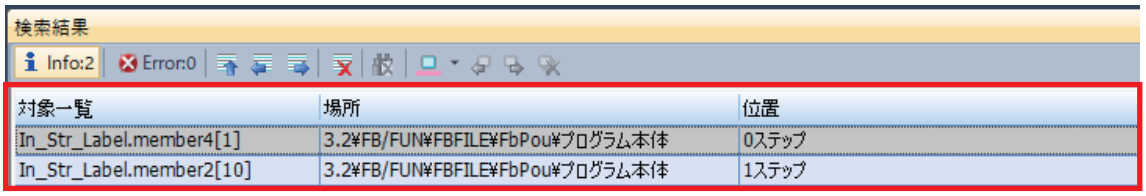
2. FB”FBPou_1”内で使用している構造体インスタンス”In_Str_Label”を検索します。

検索と置換ウィンドウから下記のように設定し，[すべて検索]ボタンをクリックします。

- 検索範囲: (現在のウィンドウ)
- 検索デバイス/ラベル: In_Str_Label



3. 検索結果ウィンドウに，構造体インスタンス"In_Str_Label"を使用している対象の一覧が表示されます。

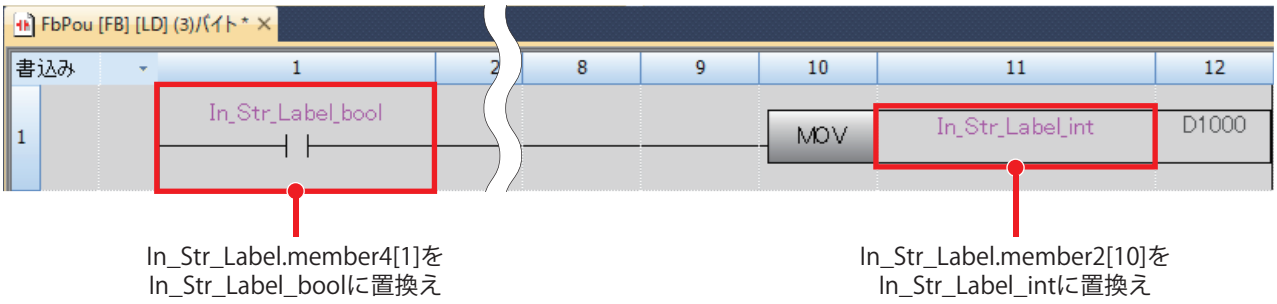


4. FB"FBPou_1"のローカルラベルを開き，手順3で検索した構造体メンバの代わりになる以下のラベルを作成します。データ型やクラスは構造体メンバや構造体インスタンスと合わせます。

- In_Str_Label.member4[1]の代わりのラベル: In_Str_Label_bool(データ型: ビット，クラス: VAR_INPUT)
- In_Str_Label.member2[10]の代わりのラベル: In_Str_Label_int(データ型: ワード[符号付き]，クラス: VAR_INPUT)

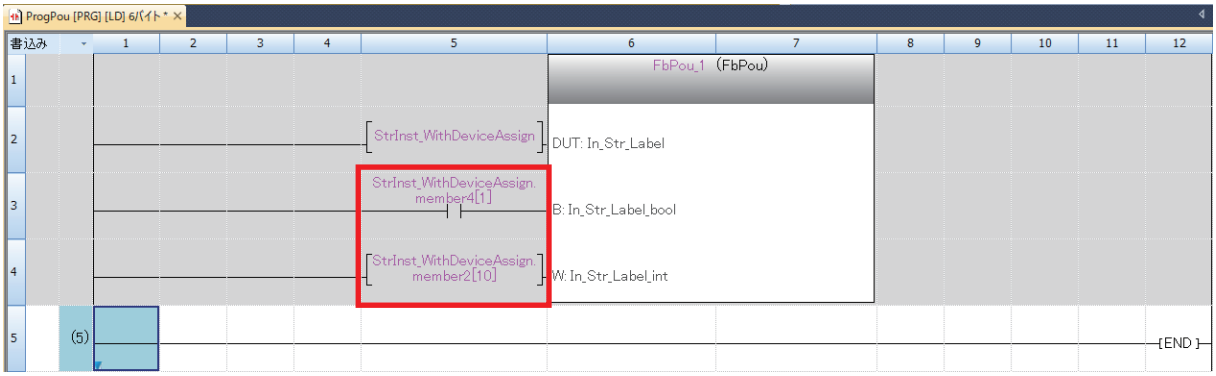


5. FB"FBPou_1"のプログラムを開き，手順3で検索した構造体メンバを手順4で作成したラベルに置き換えます。



6. FB"FBPou_1"を使用しているプログラムを開き，各引数に構造体インスタンス"StrInst_WithDeviceAssign"のメンバを設定します。

引数"In_Str_Label_bool"に"StrInst_WithDeviceAssign.member4[1]"，引数"In_Str_Label_int"に"StrInst_WithDeviceAssign.member2[10]"を設定します。



7. FB"FBPou_1"のローカルラベルを開き，不要な構造体インスタンス"In_Str_Label"を削除します。



3.5 配列同一要素への連続参照の削減

概要

配列の要素を動的に指定している(配列の添字にデバイスやラベルを指定している)場合，プログラム実行時にメモリにアクセスするための命令が複数生成されます。
そのため同じ要素へアクセスしている配列が複数ある場合，一度別のメモリにコピーすることで，プログラム実行時にメモリへアクセスするための命令の出力の回数を減らすことができます。これにより内部的に生成される命令が削減されるため，プログラム変換時間やスキャンタイムを短縮できます。

実行手順

下記のプログラムを例に，配列の同一要素への連続参照を削減する手順を示します。

・プログラム

ProgPou1 [PRG] [ST]

```
1  IF  (targetStLabel[D1, D2] > 5)  THEN
2        destLabel1 := targetStLabel[D1, D2] ;
3        destLabel2 := targetStLabel[D1, D2] + K10 ;
4        destLabel3 := targetStLabel[D1, D2] * K2 ;
5        destLabel4 := targetStLabel[D1, D2] + targetStLabel[D10, D20];
6  //  ELSE
7  //    ?実行文? ;
8  END_IF;
```

targetStLabel[D1, D2]に
複数回アクセスしている

・ローカルラベル

ProgPou1 [PRG] [ローカルラベル設...

<フィルタ>　簡易表示(Y)　表示設定(S)　チェック(K)

	ラベル名	データ型		クラス	初期値	定数
1	targetStLabel	ワード[符号付き](0..2,0..2)	...	VAR		
2	destLabel1	ワード[符号付き]	...	VAR		
3	destLabel2	ワード[符号付き]	...	VAR		
4	destLabel3	ワード[符号付き]	...	VAR		
5	destLabel4	ワード[符号付き]	...	VAR		

操作手順

1. 該当のプログラムを開き，置換対象(配列の同一要素を連続して参照している箇所)を定めます。
本例では，プログラム内で5回使用されている"targetStLabel[D1, D2]"を置換対象とします。

ProgPou1 [PRG] [ST]

```
1  IF  (targetStLabel[D1, D2] > 5)  THEN
2        destLabel1 := targetStLabel[D1, D2] ;
3        destLabel2 := targetStLabel[D1, D2] + K10 ;
4        destLabel3 := targetStLabel[D1, D2] * K2 ;
5        destLabel4 := targetStLabel[D1, D2] + targetStLabel[D10, D20];
6  //  ELSE
7  //    ?実行文? ;
8  END_IF;
```

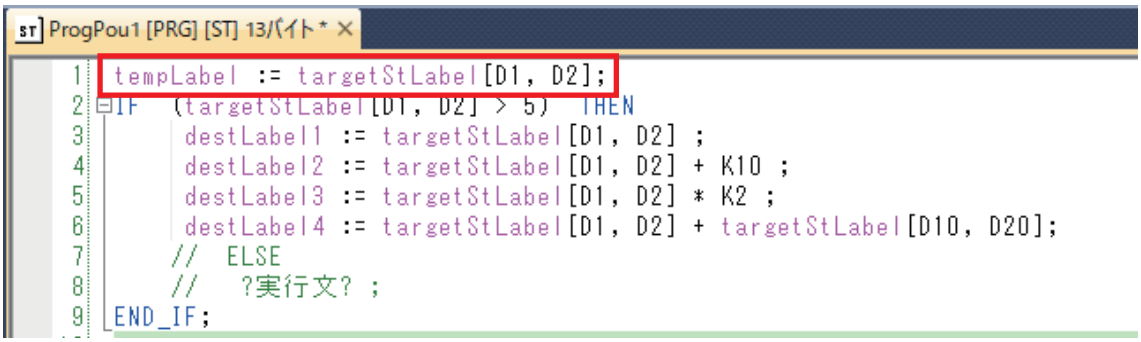
2. 該当のプログラムのローカルラベルを開き，配列要素へアクセスするラベルの代わりとなるローカルラベル"tempLabel"を追加します。

- ラベル名: tempLabel
- データ型: ワード[符号付き]
- クラス: VAR



3. 該当のプログラムを開き，プログラムの1行目に"targetStLabel[D1, D2]"を手順2で作成したローカルラベル"tempLabel"に退避する文を追加します。

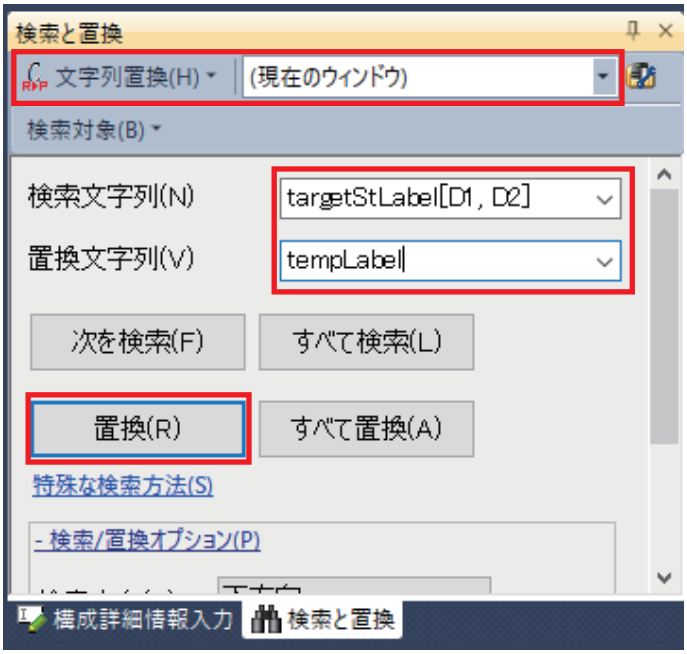
- 退避文: tempLabel := targetStLabel[D1, D2];



4. "targetStLabel[D1, D2]"を，手順2で作成したローカルラベル"tempLabel"に置換します。

検索と置換ウィンドウを下記のように設定し，[置換]ボタンをクリックします。(該当のラベルをすべて置換するまで繰り返します)

- 検索/置換方法: 文字列置換
- 検索範囲: (現在のウィンドウ)
- 検索文字列: targetStLabel[D1, D2]
- 置換文字列: tempLabel



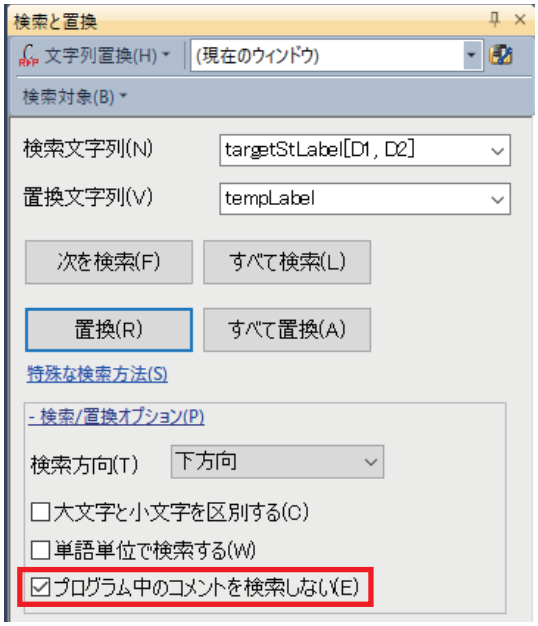
置換が完了すると，以下のようなプログラムになります。

ProgPou1 [PRG] [ST] 13バイト *

```
tempLabel := targetStLabel[D1, D2];
IF  (tempLabel > 5)  THEN
    destLabel1 := tempLabel ;
    destLabel2 := tempLabel + K10 ;
    destLabel3 := tempLabel * K2 ;
    destLabel4 := tempLabel + targetStLabel[D10, D20];
   //  ELSE
   //   ?実行文? ;
END_IF;
```

Point

- [すべて置換]ボタンから一括置換ができますが，1つずつ確認して置換することを推奨します。
- プログラム中のコメントを置換しない場合は，検索/置換オプションから"プログラム中のコメントを検索しない"にチェックを入れてください。



注意事項

- 本テクニックは，配列の添字が定数以外の場合に効果があります。
- 配列要素の動的指定に使われるデバイスやラベルがプログラムの途中で書き換わる場合など，プログラムの内容によっては配列要素へアクセスするラベルの置き換えができないことがあります。プログラムの内容を確認し，置き換えを行ってください。

"targetStLabel[D0, D1]"のD1の値が
プログラムの途中で書き換わる

ProgPou2 [PRG] [ST] *

```
IF  (targetStLabel[D1, D2] > 5)  THEN
    destLabel1 := targetStLabel[D1, D2] ;
    destLabel2 := targetStLabel[D1, D2] + K10 ;
    D1 := 0 ;
    destLabel3 := targetStLabel[D1, D2] * K2 ;
    destLabel4 := targetStLabel[D1, D2] + targetStLabel[D10, D20];
   //  ELSE
   //   ?実行文? ;
END_IF;
```

4 スキャンタイムの高速化

プログラムの記述やパラメータの設定を工夫して，スキャンタイムを高速化します。スキャンタイムを高速化することで，プログラムの処理時間を短縮できます。
本章では，スキャンタイムの高速化を行うためのテクニックについて示します。

テクニック一覧

本章で説明するテクニック一覧を下記に示します。

項目	内容	参照
BMOV命令の複数命令への置き換え	BMOV命令をFOR～NEXT命令を使用した複数のMOV命令に置換えます。	☞ 54ページ BMOV命令の複数命令への置き換え
立上り実行命令の削減	立上り実行命令(OFF→ONの立上り時のみ実行する命令)の使用を減らします。	☞ 59ページ 立上り実行命令の削減
ユニットアクセス命令の削減	インテリユニットのバッファメモリにダイレクトアクセスしている箇所を，インテリユニットの自動リフレッシュ，またはリンクリフレッシュに変更します。	☞ 61ページ ユニットアクセス命令の削減
初期実行タイププログラムの活用	STプログラムで，1度だけ実行する処理を初期実行タイププログラムに移動させます。	☞ 64ページ 初期実行タイププログラムの活用
引数の省略と直接参照	STプログラムで，モーション制御FBの入出力引数への代入処理や不要な割り当てを省略・削除します。	☞ 70ページ 引数の省略と直接参照
命令数が多い回路ブロックの分割	ラダープログラムで，1つの回路ブロックに含まれている命令数を100命令以下に減らします。	☞ 74ページ 命令数が多い回路ブロックの分割
ラベル初期値の活用	値を変更しないラベルやFBの初期値の設定にラベル初期値を使用します。	☞ 77ページ ラベル初期値の活用

スキャンタイムの確認方法

スキャンタイムの値は，下記の特殊レジスタから確認できます。
各テクニックを実行した際にスキャンタイムを確認したい場合は使用してください。

番号	名称	内容	内容詳細	セット
SD518	初期スキャンタイム	初期スキャンタイム(ms単位)	初期スキャンタイムがSD518，SD519に格納されます。(計測は1μs単位で実行します) SD518: msの位を格納(格納範囲: 0~65535) SD519: μsの位を格納(格納範囲: 0~999) STOP→RUN時に一度0クリアされます。	システム側で毎回END処理時にセット
SD519		初期スキャンタイム(μs単位)		
SD520	現在スキャンタイム	現在スキャンタイム(ms単位)	現在スキャンタイムがSD520，SD521に格納されます。(計測は1μs単位で実行します) SD520: msの位を格納(格納範囲: 0~65535) SD521: μsの位を格納(格納範囲: 0~999) STOP時に0クリアされます。	
SD521		現在スキャンタイム(μs単位)		
SD522	最小スキャンタイム	最小スキャンタイム(ms単位)	初期実行プログラムのスキャンタイムを除くスキャンタイムの最小値がSD522，SD523に格納されます。(計測は1μs単位で実行します) SD522: msの位を格納(格納範囲: 0~65535) SD523: μsの位を格納(格納範囲: 0~999) STOP→RUN時に一度0クリアされます。	
SD523		最小スキャンタイム(μs単位)		
SD524	最大スキャンタイム	最大スキャンタイム(ms単位)	初期実行プログラムのスキャンタイムを除くスキャンタイムの最大値がSD524，SD525に格納されます。(計測は1μs単位で実行します) SD524: msの位を格納(格納範囲: 0~65535) SD525: μsの位を格納(格納範囲: 0~999) STOP→RUN時に一度0クリアされます。	
SD525		最大スキャンタイム(μs単位)		

またラダープログラムの場合は，スキャンタイム測定機能を使用できます。
スキャンタイム測定機能の詳細については，下記のマニュアルを参照してください。
📖GX Works3 オペレーティングマニュアル

4.1 BMOV命令の複数命令への置き換え

概要

MOV命令はBMOV命令よりも実行速度が速いため，BMOV命令をMOV命令に置き換えることでスキャンタイムを短縮できます。
BMOV命令の置き換えには複数のMOV命令に置き換える方法と，FOR～NEXT命令でMOV命令を複数回実行させる方法があります。

MOV命令とBMOV命令の性能値

MXR500-256でのMOV命令(INC命令)とBMOV命令の性能値の例を下記に示します。
BMOV命令をMOV命令に置き換える際の目安としてください。

命令	実行速度(μ秒)
MOV命令	0.00050
INC命令	0.00361
BMOV命令(転送数1点の場合)	0.076
BMOV命令(転送数1024点の場合)	0.191

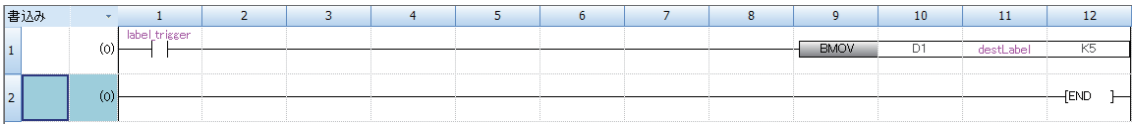
注意事項

機種によって性能値は異なります。

実行手順

下記のプログラムを例に，BMOV命令をMOV命令に置き換える手順を示します。

・プログラム



・ローカルラベル

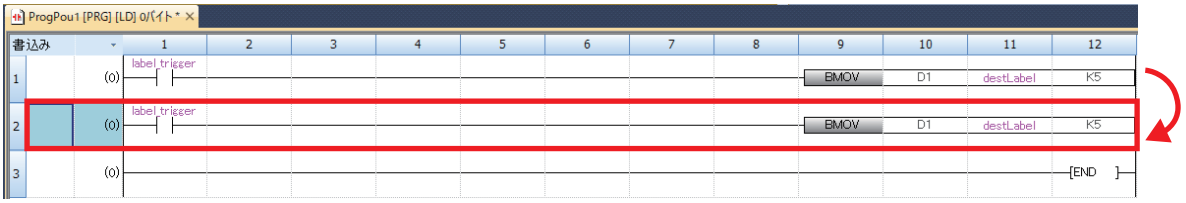


複数のMOV命令に置き換え

BMOV命令を複数のMOV命令に置き換える手順を示します。

操作手順

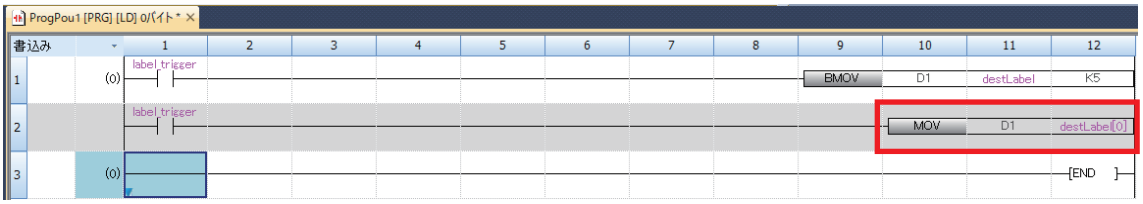
1. BMOV命令を使用しているプログラムを開き，BMOV命令の行をコピーして次の行に貼り付けます。



2. 手順1で貼り付けたBMOV命令をMOV命令に変更します。

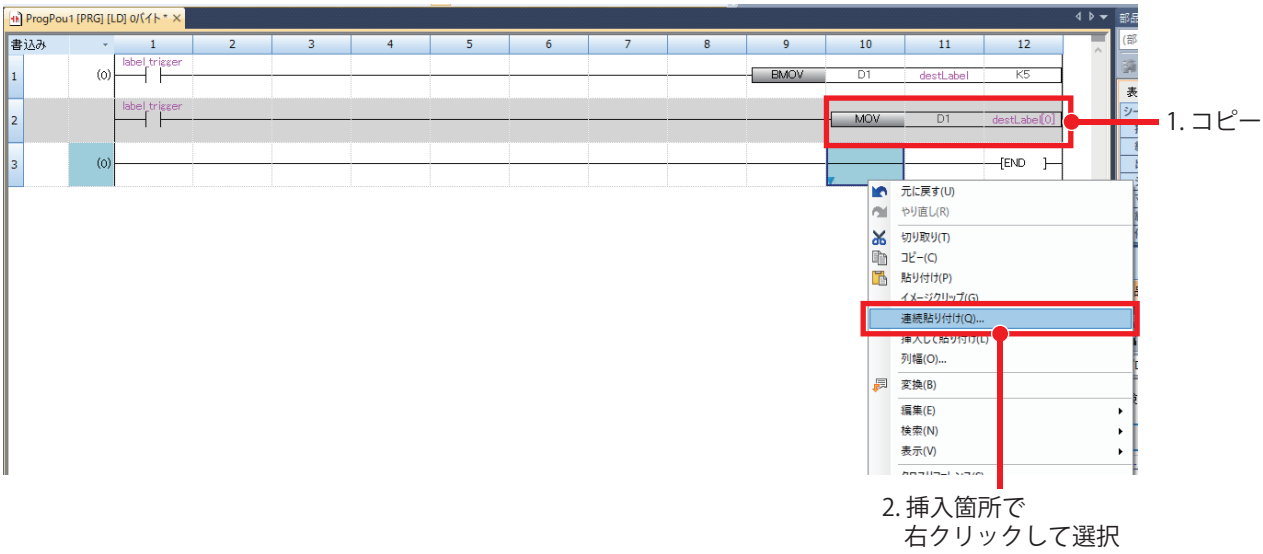
引数は，"D1"の値をローカルラベル"destLabel[0]"に転送するよう変更します。

- MOV D1 destLabel[0]



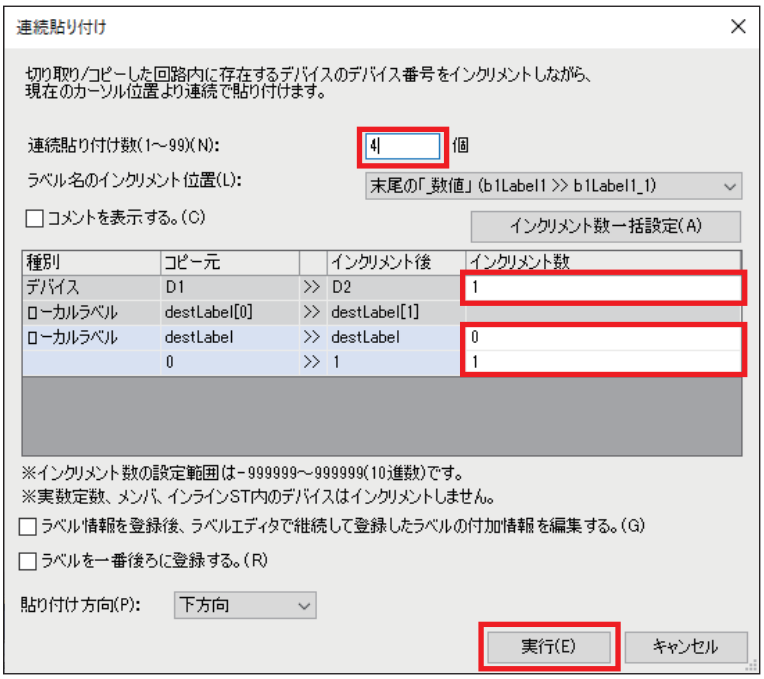
3. 手順2で修正したMOV命令をコピーして，次にMOV命令を挿入する位置で連続貼り付けを行います。

MOV命令を挿入する位置で右クリック⇨[連続貼り付け]

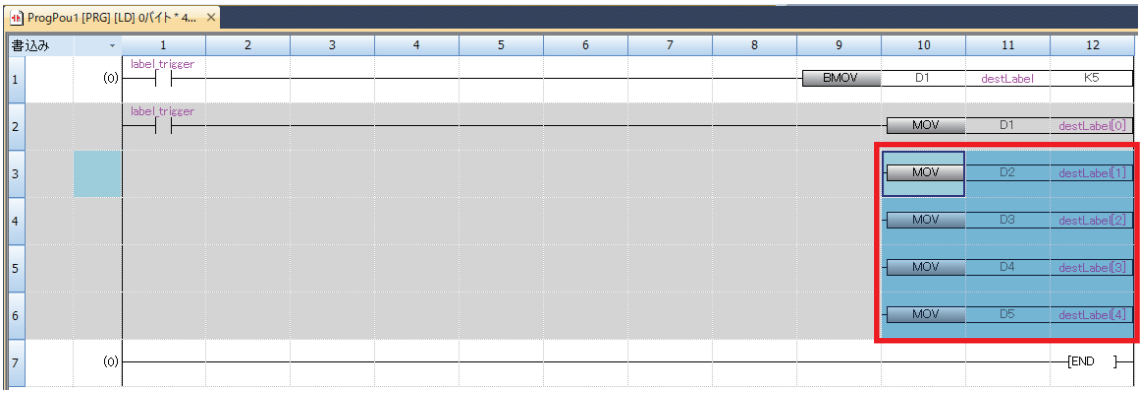


4. 連続貼り付け数とインクリメント数を設定して，[実行]ボタンをクリックします。

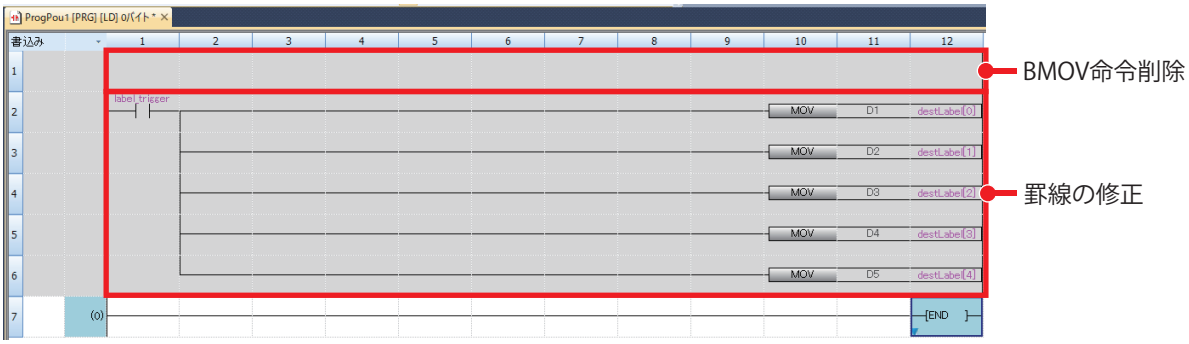
- 連続貼り付け数: 4個
- デバイス"D1"のインクリメント数: 1
- ローカルラベル"destLabel"の配列要素のインクリメント数: 1



連続貼り付けが行われます。



5. 罫線を修正して各MOV命令を手順1で追加した行の接点と接続し，不要なBMOV命令の行を削除します。



FOR～NEXT命令とMOV命令に置き換え

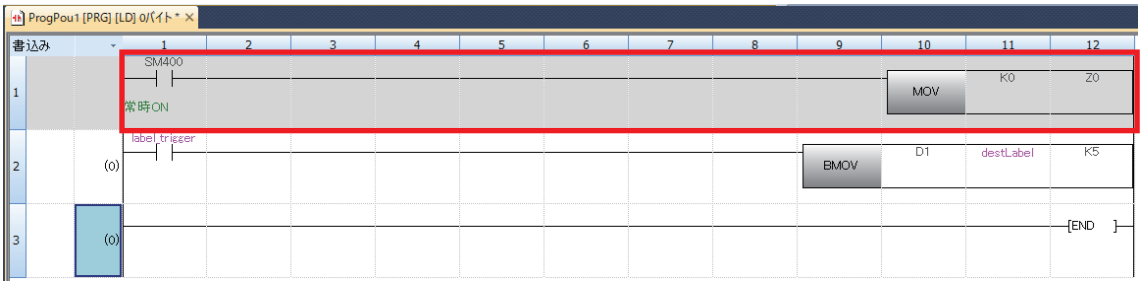
BMOV命令を，FOR～NEXT命令でMOV命令を複数回実行するプログラムに置き換える手順を示します。

操作手順

1. BMOV命令を使用しているプログラムを開き，1行目に新規行を挿入して，インデックスレジスタ"Z0"を初期化する回路を追加します。

接点には特殊リレー SM400(常時ON)を使用して，以下のMOV命令でインデックスレジスタ"Z0"にK0を転送します。

・MOV K0 Z0



注意事項

インデックスレジスタは，デバイス番号を間接的に指定する際に使用します。プログラムの他の箇所でインデックスレジスタを使用している場合は，インデックスレジスタ番号が重複しないように注意してください。

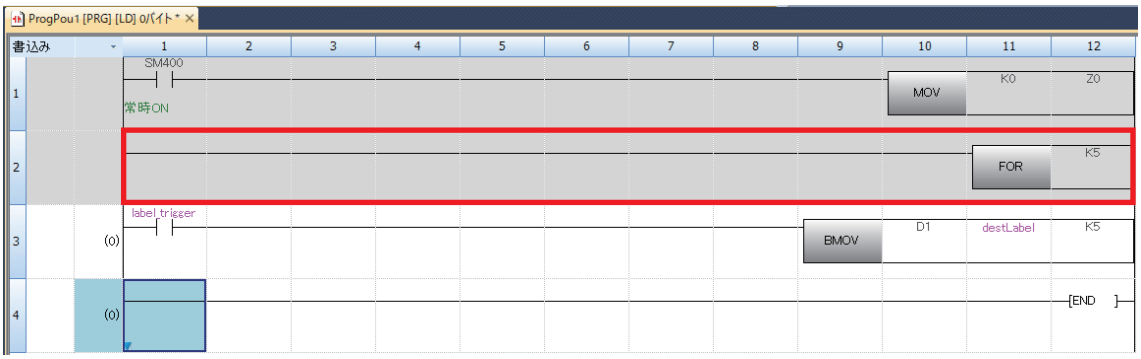
インデックスレジスタの詳細については，下記のマニュアルを参照してください。

📖MELSEC MXコントローラ(MX-Rモデル) プログラミングマニュアル

2. 2行目に新規行を挿入してFOR文を追加します。

MOV命令の実行回数分(転送数分)を繰り返すよう，引数に"K5"を設定します。

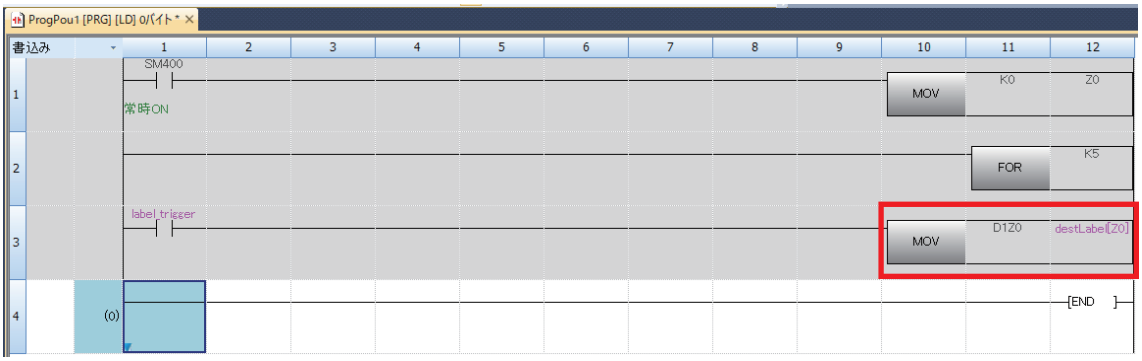
・FOR K5



3. BMOV命令をMOV命令に変更します。

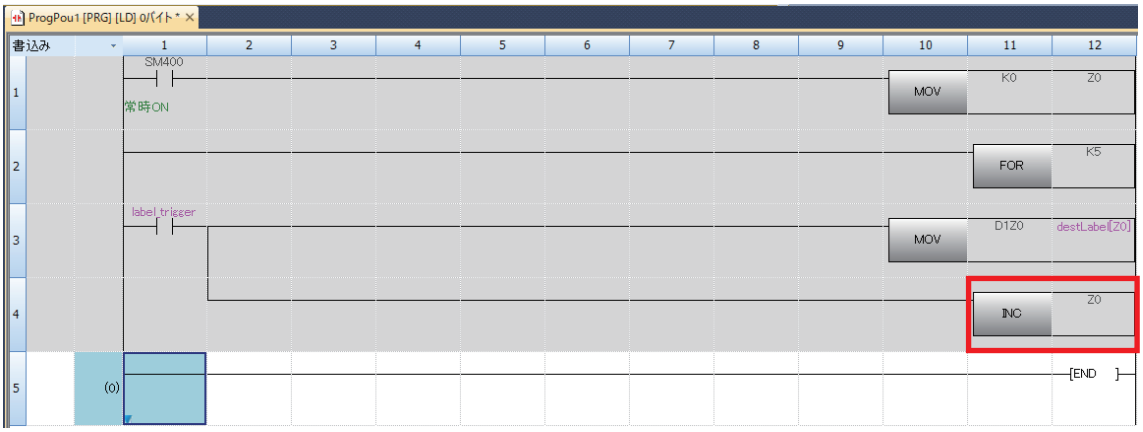
引数は，手順1で初期化したインデックスレジスタを使用して，"D1Z0"の値をローカルラベル"destLabel[Z0]"に転送するよう変更します。

・MOV D1Z0 destLabel[Z0]

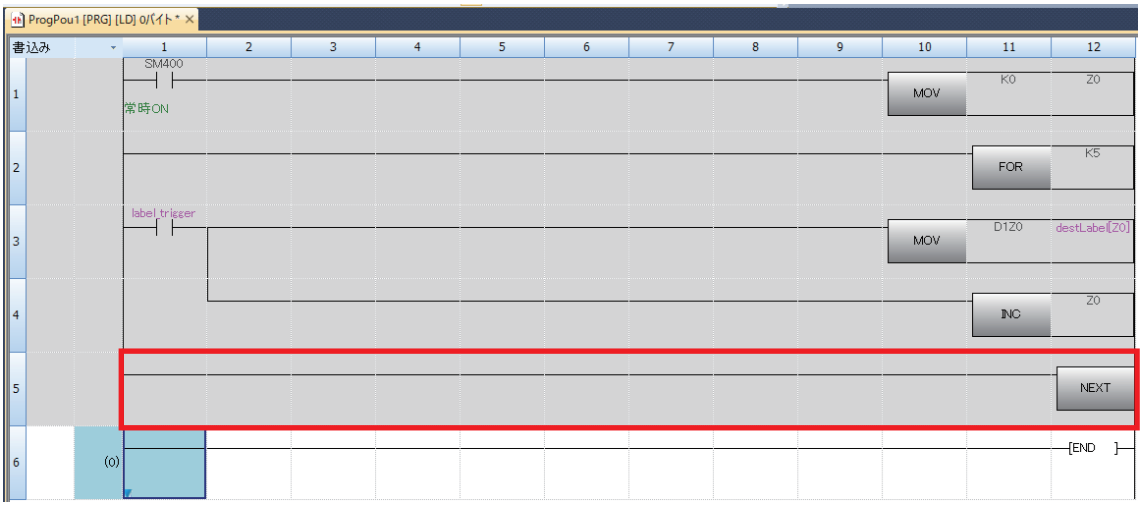


4. MOV命令の次の行に，インデックスレジスタ"Z0"を1つ増加する以下のINC命令を追加します。
追加したINC命令は，MOV命令の接点と接続します。

- INC Z0



5. INC命令の次の行にNEXT文を追加します。



注意事項

- BMOV命令を複数のMOV命令に置き換える場合，プログラムの可読性が低下することがあります。
- BMOV命令の転送数が多い場合，MOV命令に置き換えてもスキャンタイムが短縮しないことがあります。置き換える前にMOV命令とBMOV命令の実行速度を計算して確認してください。実行速度については，下記を参考にしてください。

☞ 54ページ MOV命令とBMOV命令の性能値

4.2 立上り実行命令の削減

概要

立上り実行命令(命令記号の語尾に"P"が付いており，OFF→ONの立上り時のみ実行する命令)は，ON時実行命令と比較して処理性能が遅いため，立上り実行命令を削減することでスキャンタイムを短縮できます。
立上り命令の削減には以下の方法があります。

- 不要な立上り実行命令は，ON時実行命令に置き換える
- 複数の立上り実行命令を使用している場合は，接点を立上り実行命令に変更し，その他の立上り実行命令をON時実行命令に変更する

立上り実行命令の性能値

実行条件が立上り命令の場合，ON時実行命令の処理時間に追加で時間が加算されます。(例: MOVPの処理時間 = MOVの処理時間 + 加算時間)
機種ごとの加算時間を以下に示します。
立上り命令を置き換える際や変更する際の目安としてください。

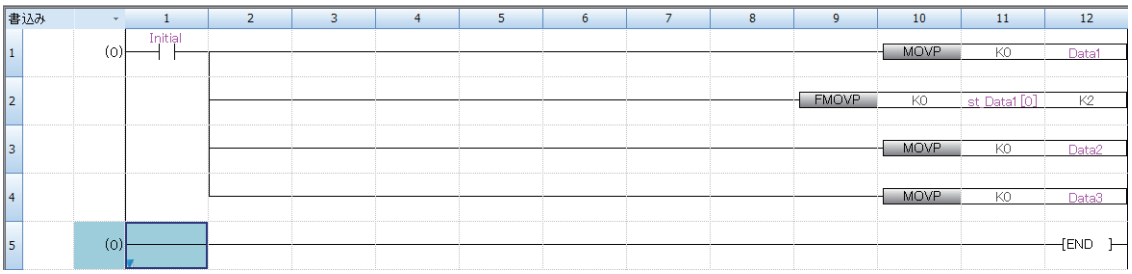
機種	加算時間(n秒)
MXR300-16，MXR300-32，MXR300-64	28.62
MXR500-128，MXR500-256	20.81

注意事項

機種によって性能値は異なります。

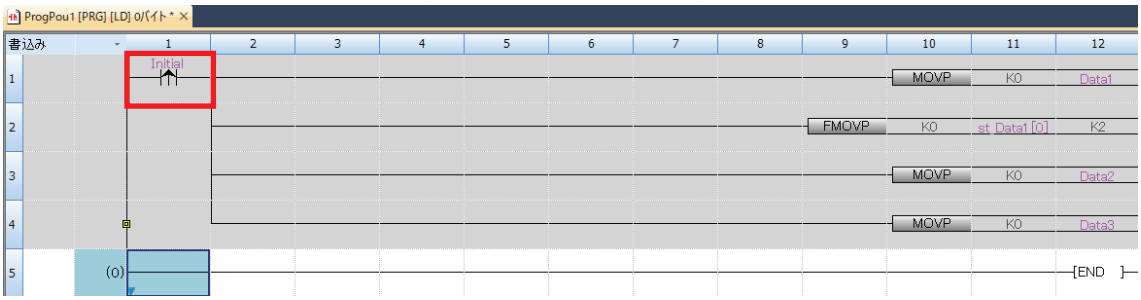
実行手順

下記のプログラムを例に，立上り実行命令"MOVP"，"FMOVP"を削減する手順を示します。



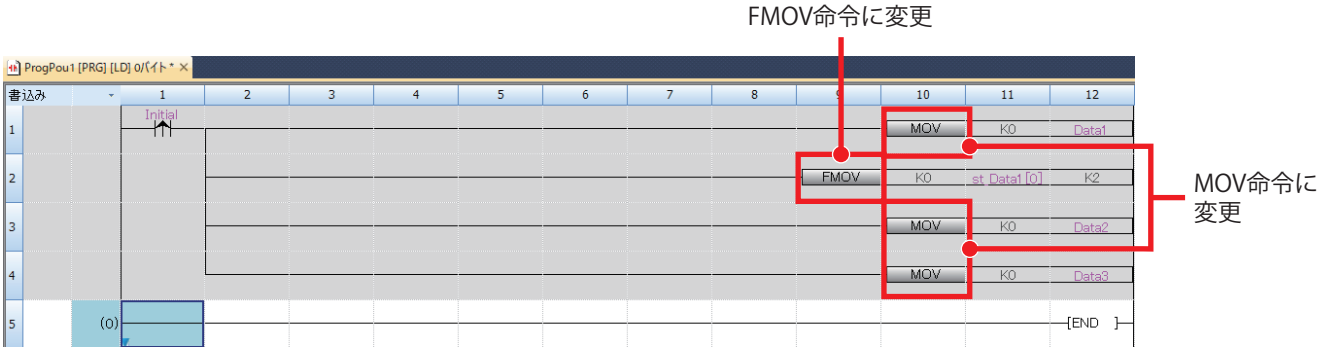
操作手順

1. プログラムを開き，複数の立上り実行命令が接続されている接点を立上り実行命令に変更します。
本例では，a接点演算開始命令(LD)を立上りパルス演算開始命令(LDP)に変更します。



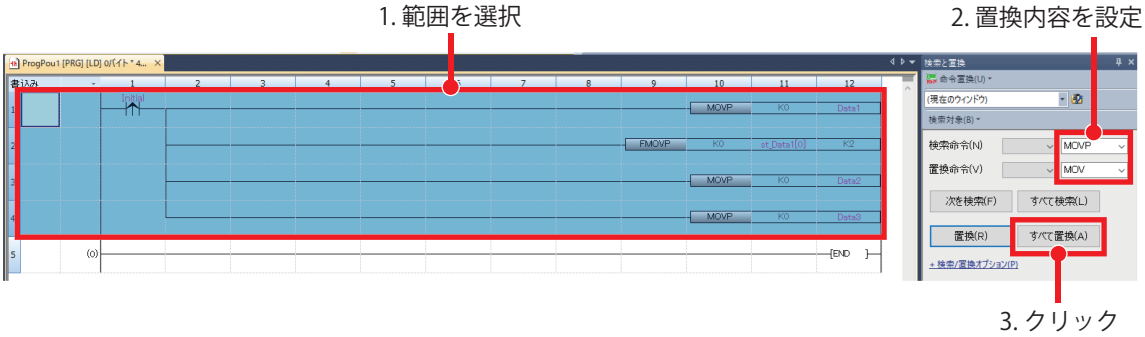
2. 手順1の接点に接続している立上り実行命令をON時実行命令に変更します。

- MOVP命令→MOV命令に変更
- FMOVP命令→FMOV命令に変更



Point

同じ命令が複数ある場合は，検索と置換ウィンドウの"命令置換"から一括置換ができます。
置換範囲を選択した状態で検索命令と置換命令を設定し，[すべて置換]ボタンをクリックしてください。



4.3 ユニットアクセス命令の削減

概要

ユニットバッファメモリへアクセスする場合，FROM命令やTO命令などのユニットアクセス命令を使用したバッファメモリへのダイレクトアクセスを集約することや，インテリユニットの自動リフレッシュ機能やリンクリフレッシュ機能を使用したアクセスに変更することで，スキャンタイムを短縮できます。

ユニットアクセス命令とリンクリフレッシュの性能値

MXR500-128，MXR500-256のユニットアクセス命令や自動リフレッシュ機能，リンクリフレッシュ機能の性能値の例を以下に示します。
ユニットアクセス命令を置き換える際の目安としてください。

項目	1ワードあたりの最小実行性能値		
	転送数: 1ワード	転送数: 1024ワード	転送数: 32768ワード
FROM命令	5470μs	40.42μs	—
TO命令	3740μs	23.43μs	—
自動リフレッシュ (読出し)*1	10730ns	30.45ns	20.32ns
自動リフレッシュ (書込み)*1	7360ns	17.17ns	10.22ns
リンクリフレッシュ (CC-Link IE TSNユニット)*1	52012ns	62.78ns	13.58ns

*1 コントローラのマニュアルに掲載の計算式にて算出しています。詳細については，下記を参照してください。
MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル

注意事項

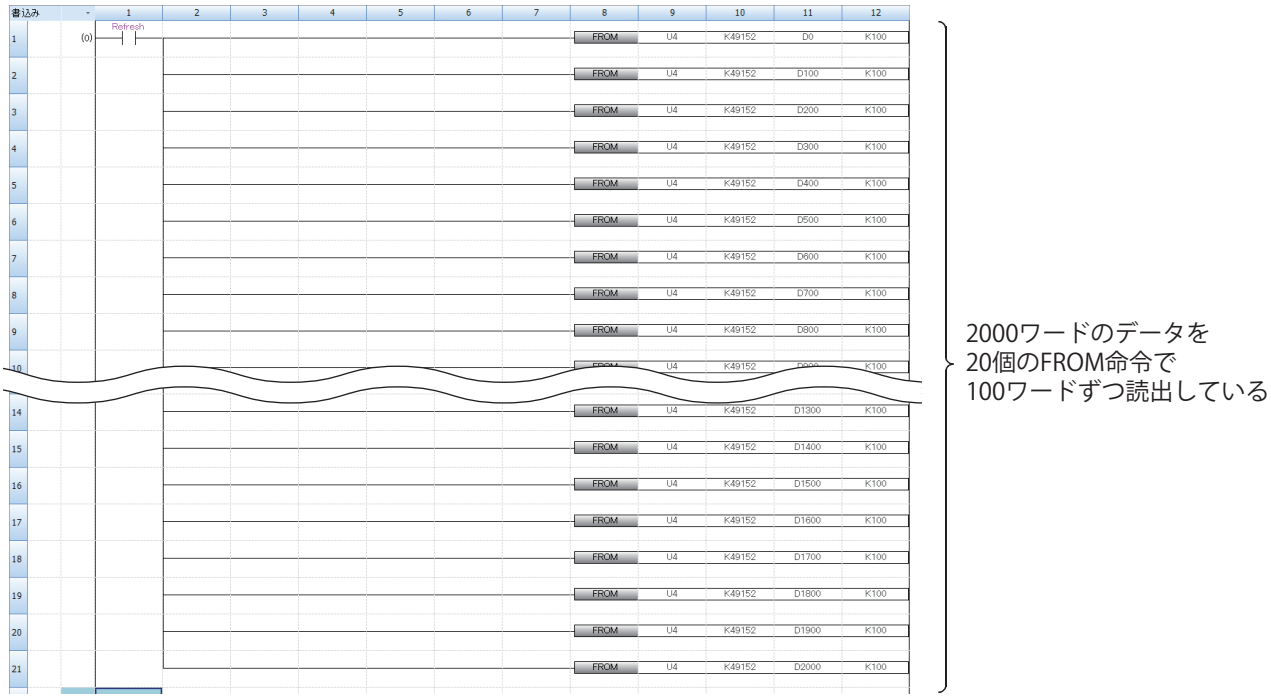
機種によって性能値は異なります。

実行手順

ユニットアクセス命令を削減する手順について示します。

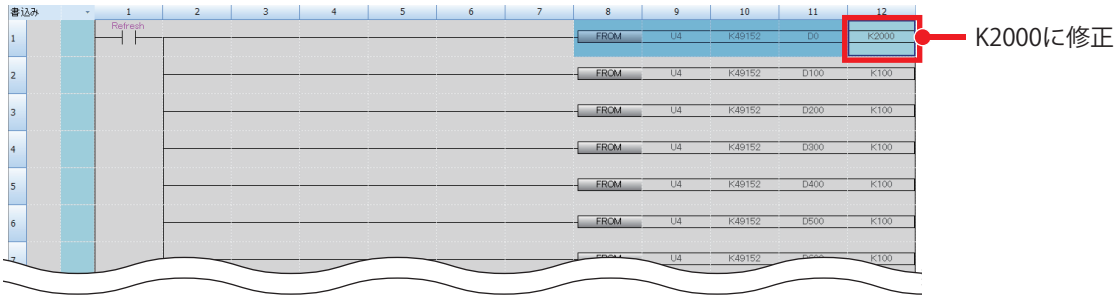
FROM命令やTO命令の集約

下記の先頭I/Oが0x0040のインテリジェント機能ユニットのバッファメモリからDデバイスへ2000点(2000ワード)分の値を読出すプログラムを例に，20個のFROM命令を1個のFROM命令に集約する手順を示します。



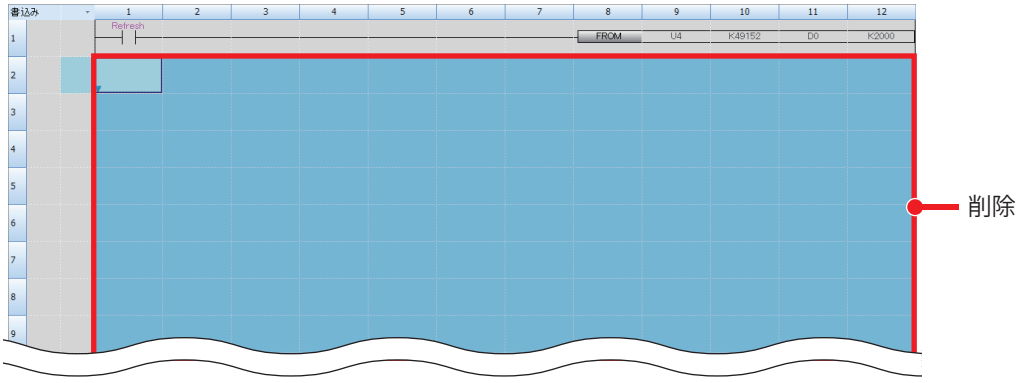
操作手順

1. 該当のプログラムファイルを開き，1行目のFROM命令の読出しデータ数をK2000(2000ワード)に変更します。



2. 2行目以降のFROM命令を削除します。

これにより，2000ワードのデータを100ワードずつ読み出す20個のFROM命令を，2000ワードのデータを一括で読出す1個のFROM命令に集約できます。



自動リフレッシュ機能やリンクリフレッシュ機能への置き換え

ユニットアクセス命令を自動リフレッシュ機能やリンクリフレッシュ機能へ置き換えます。
自動リフレッシュ機能やリンクリフレッシュ機能の使用方法は，各ユニットのマニュアルを参照してください。

4.4 初期実行タイププログラムの活用

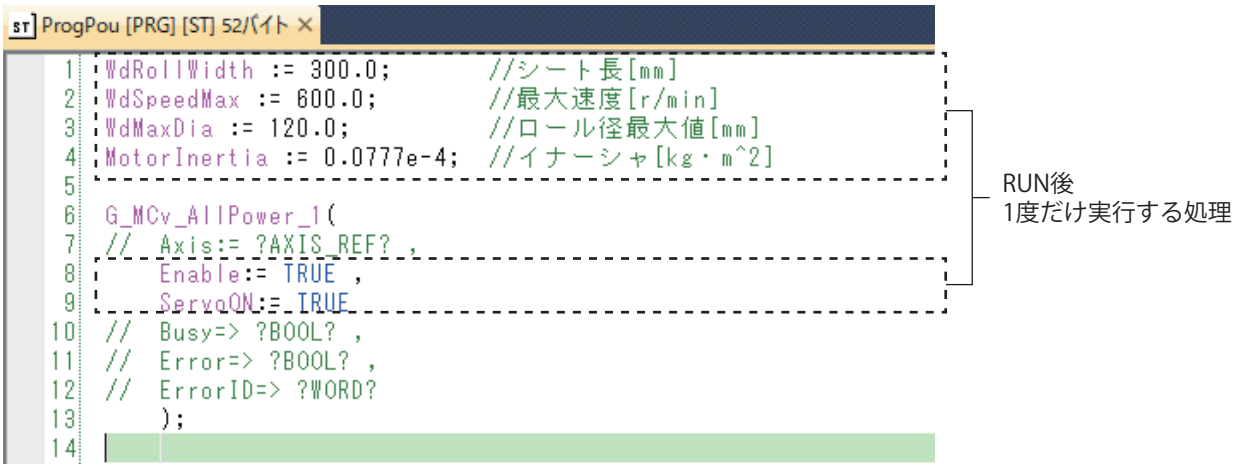
概要

スキャン実行タイププログラムは，1スキャンに1回プログラムが実行されます。
スキャン実行タイプのSTプログラム(モーション制御プログラム)の中にシーケンサRUN後に1度しか実行する必要がない処理がある場合，それらを初期実行タイプSTプログラムに移動させることで，スキャンタイムを短縮できます。

実行手順

下記のSTプログラムを例に，プログラム内でシーケンサRUN後に1度しか実行する必要がない処理を初期実行タイププログラムに移動させる手順を示します。

・STプログラム



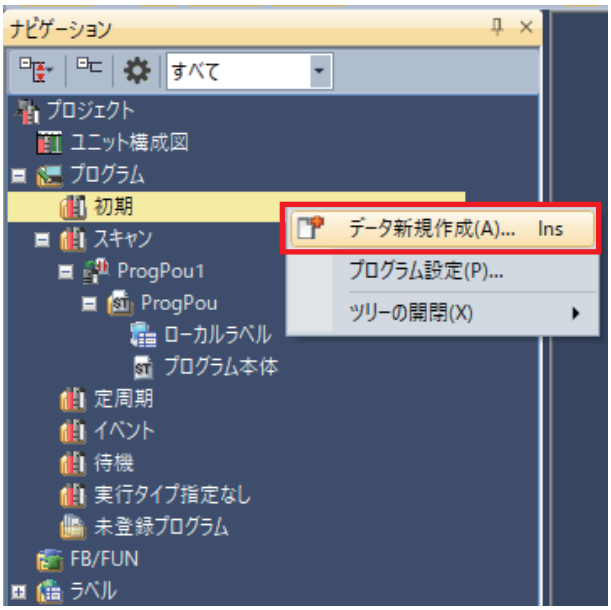
・ローカルラベル



操作手順

1. 初期実行タイプのプログラムファイルを新規作成します。

ナビゲーションウィンドウ⇨[プログラム]⇨[初期]を右クリック⇨[データ新規作成]

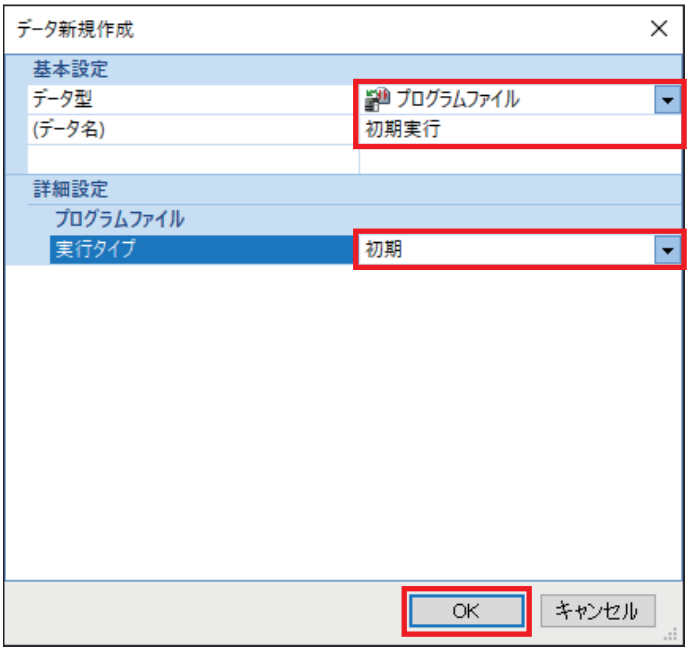


Point

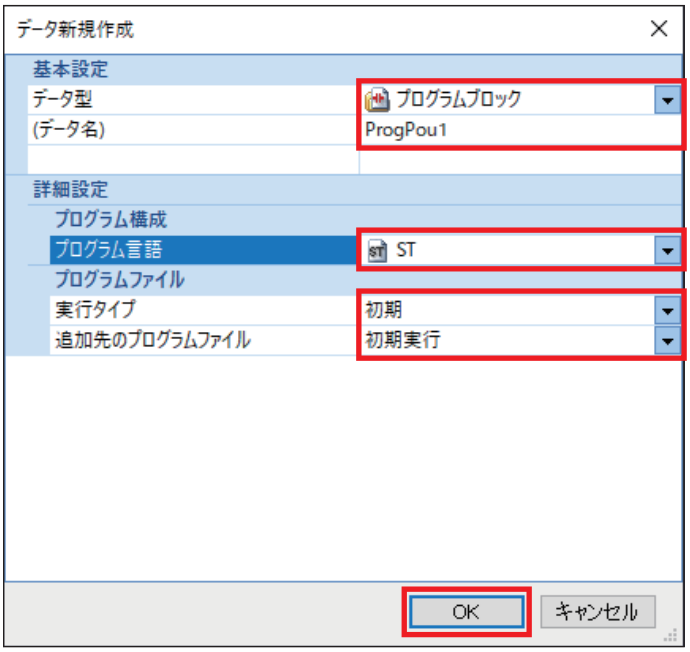
プログラムの変換時間は，1プログラムファイルあたりのプログラム量が少ないほど短縮されます。
そのためシーケンサRUN後1度しか実行しない処理は，新規の初期実行タイプのプログラムファイルに移動させることを推奨します。(既存の初期実行タイプのプログラムファイルへ移動させる場合でも，動作に問題はありません)

2. 各項目を設定し，[OK]ボタンをクリックします。

- データ型: プログラムファイル
- (データ名): 初期実行
- 実行タイプ: 初期

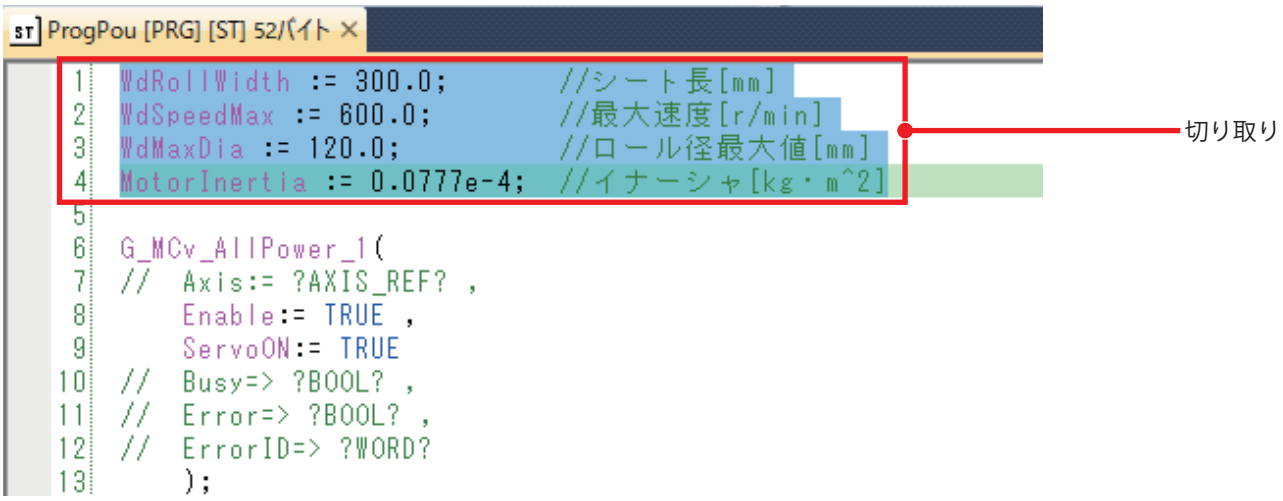


3. 手順1と同様の手順で「データ新規作成」画面を開き，プログラムブロックを新規作成します。
各項目を設定し，[OK]ボタンをクリックします。

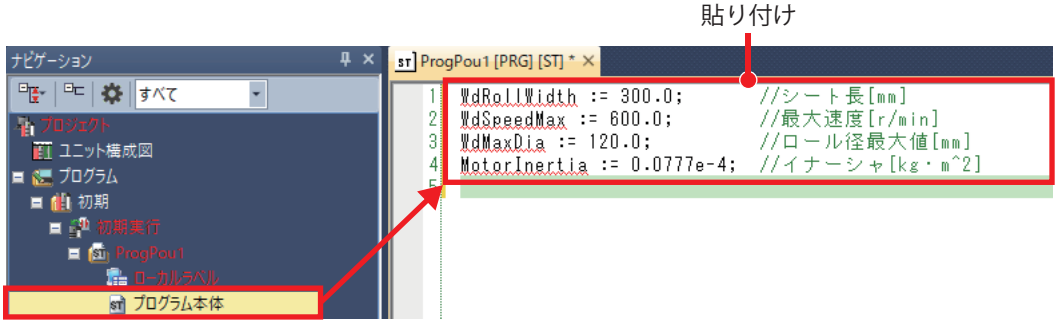


項目	設定内容	備考
データ型	プログラムブロック	—
(データ名)	ProgPou1	—
プログラム言語	ST	—
実行タイプ	初期	手順2で作成したプログラムファイルと同じ実行タイプを設定します。
追加先のプログラムファイル	初期実行	手順2で作成したプログラムファイルを設定します。

4. 移動前のSTプログラムを開き，1～4行目の各ラベルの初期値を設定している処理を切り取ります。



5. 手順3で作成した初期実行タイププログラムを開き，手順4で切り取った処理を貼り付けます。



6. 移動前のSTプログラムのローカルラベルを開き，初期実行タイププログラムで使用しているローカルラベル(本例ではNo.1～4)を切り取ります。

エディタの表の行番号をクリックし，行全体を選択した状態で切り取ってください。



Point

行全体を選択していない場合，ラベルエディタ上で非表示になっている内容はコピーされません。

7. グローバルラベルを開き，手順6で切り取ったローカルラベルを貼り付け，クラスを設定します。



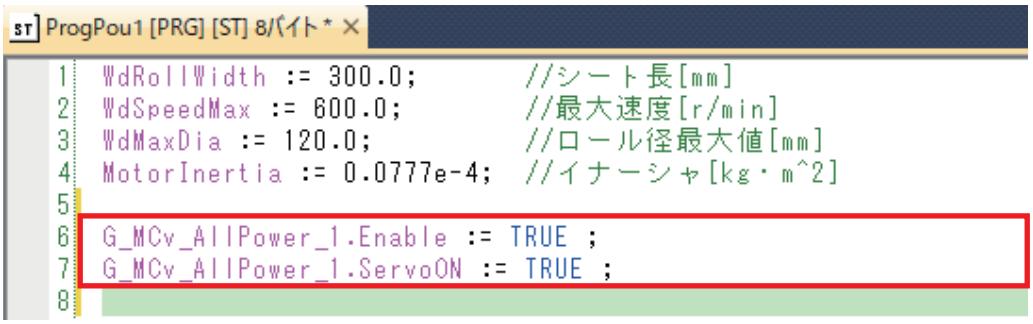
Point

本例ではローカルラベルの貼り付け先をグローバルラベルとしていますが，必ずしもグローバルラベルである必要はありません。

該当ラベルの参照状況を確認のうえ，貼り付け先をローカルラベルかグローバルラベルか選定してください。

8. 初期実行タイププログラムを開き，FBインスタンス"G_MCv_AllPower_1"の入力変数"Enable"，"ServoON"に"TRUE"を指定する処理を記述します。

- G_MCv_AllPower_1.Enable := TRUE;
- G_MCv_AllPower_1.ServoON := TRUE;

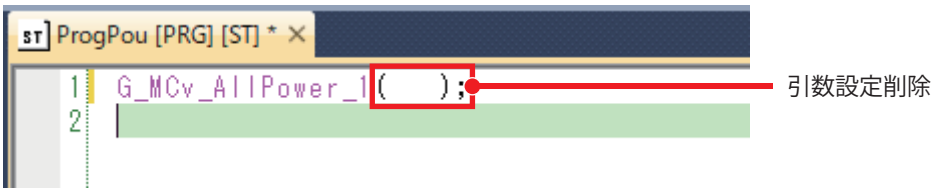


Point

- モーション制御FBのインスタンスは，移動前のプログラムと初期実行タイププログラムの両方で使用するためグローバルラベルである必要があります。ローカルラベルの場合は，グローバルラベルに変更してください。
- モーション制御FBのインスタンス名の後に"."(ピリオド)を入力するとラベル名の候補が表示され，候補を選択するとラベル名を自動で入力できます。候補が表示されない場合はメニューの[ツール]⇨[オプション]⇨"編集"⇨"命令/デバイス/ラベル候補表示"⇨"動作設定"⇨"命令/デバイス/ラベル名を予測して表示"から設定できます。詳細については，下記のマニュアルを参照してください。

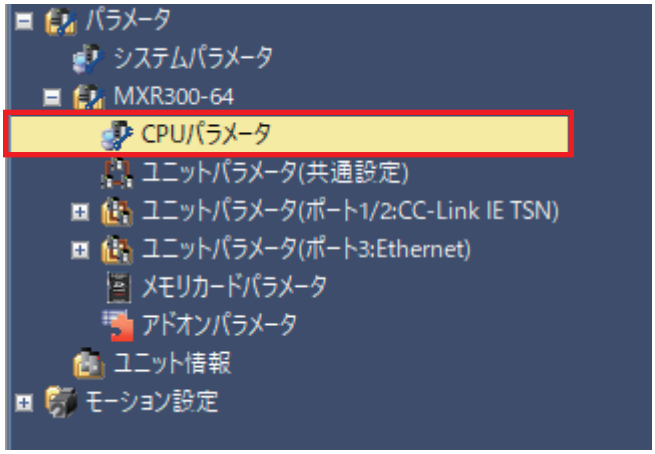
📖GX Works3 オペレーティングマニュアル

9. 移動前のSTプログラムを開き，FBインスタンス"G_MCv_AllPower_1"の引数設定を削除します。



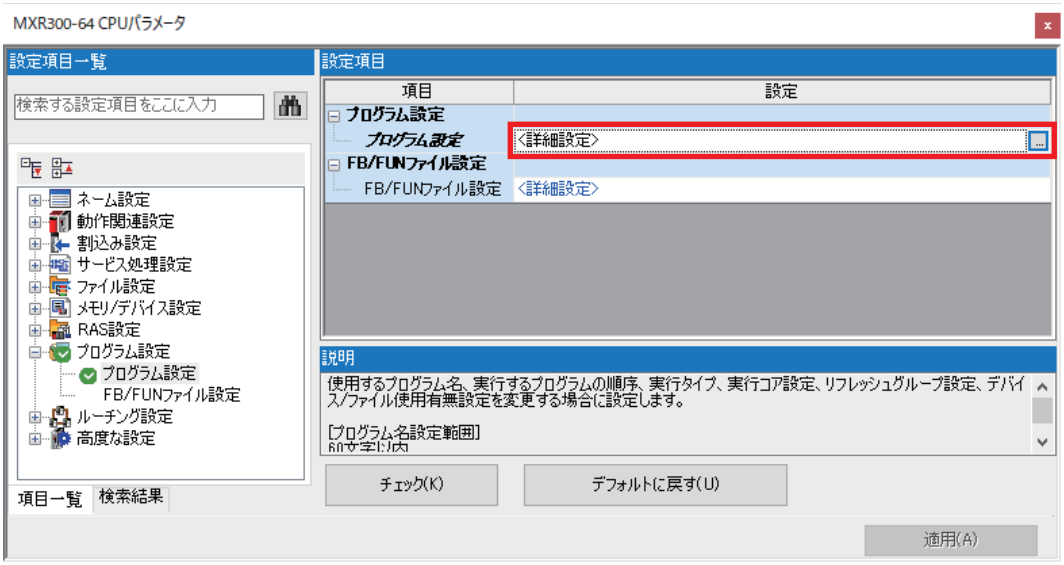
10. CPUパラメータを開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[CPUパラメータ]をダブルクリック

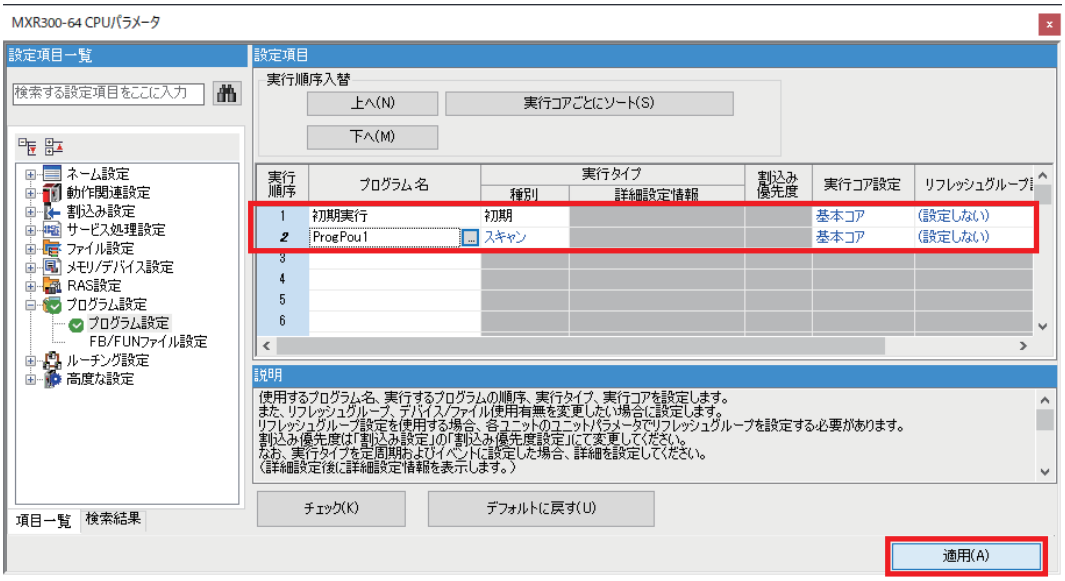


11. プログラムの実行順を設定するため，プログラム設定を開きます。

[プログラム設定]⇨[プログラム設定]の<詳細設定>をダブルクリック



12. 初期実行タイププログラムが最初に実行されるようプログラムを並び替え，[適用]ボタンをクリックして設定を適用します。



4.5 引数の省略と直接参照

概要

STプログラム(モーション制御プログラム)内のモーション制御FBは多くの引数を持つため，入出力引数に別のラベルを割り当てると代入/参照処理の負荷が発生しやすくなります。
そのため以下のように別のラベルを使用しないことで，代入/参照処理の生成を削減し，スキャンタイムを短縮します。

- ラベルを使用せずに入出力引数へ値を直接入力する
- 不要なFB引数の値/ラベルの割り当てを省略する
- IF文の条件判定ではFBのラベルを直接参照する
- 別ラベルへFBの値を代入する際はFB引数を直接代入する

実行手順

下記のプログラムを例に，モーション制御FBのインスタンス"MCv_Jog_1"に対する入出力引数の設定を修正する手順を示します。

ProgPou1 [PRG] [ST]

```
1  bLabel6[0] := TRUE;
2
3  //軸1：JOG運転実行
4  MCv_Jog_1(
5      Axis:= Axis0001.AxisRef ,
6      JogForward:= bLabel6[0] ,
7      JogBackward:= bLabel6[1] ,
8  //  Velocity:= ?LREAL? ,
9  //  Acceleration:= ?LREAL? ,
10 //  Deceleration:= ?LREAL? ,
11 //  Jerk:= ?LREAL? ,
12 //  Options:= ?DWORD? ,
13     Done=> bLabel7[0] ,
14     Busy=> bLabel7[1] ,
15     Active=> bLabel7[2] ,
16     CommandAborted=> bLabel7[3] ,
17     Error=> bLabel7[4] ,
18     ErrorID=> uLabel8
19     );
20
21 //軸1：エラー発生(bLabel7[4]がON)
22 IF  bLabel7[4]  THEN
23     //uLabel11にエラーID(uLabel8)を格納
24      uLabel11 := uLabel8;
25 END_IF;
26
```

引数"JogForward"にラベル"bLabel6[0]"を割り当てている(直接指定可能)

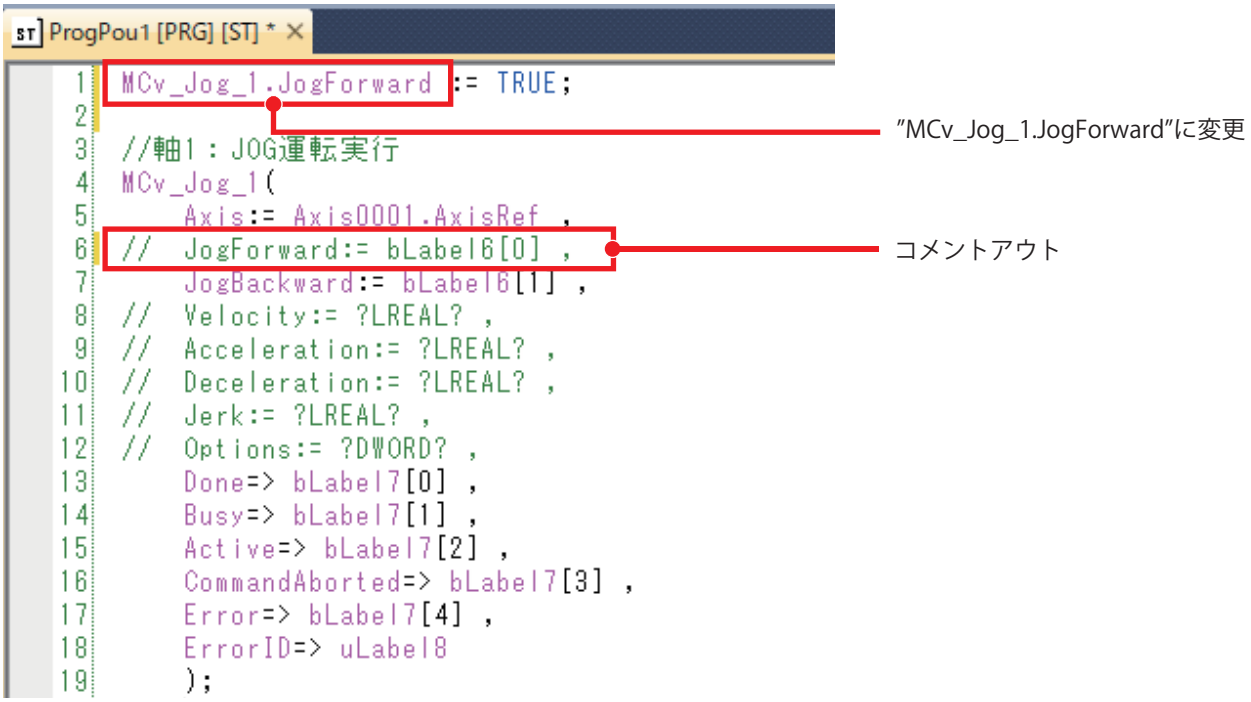
使用しない引数が残っている(省略可能)

使用しない引数のためラベルを指定する必要がない(省略可能)

引数"Error"，"ErrorID"にラベル"bLabel7[4]"や"uLabel8"を割り当てている(直接指定可能)

操作手順

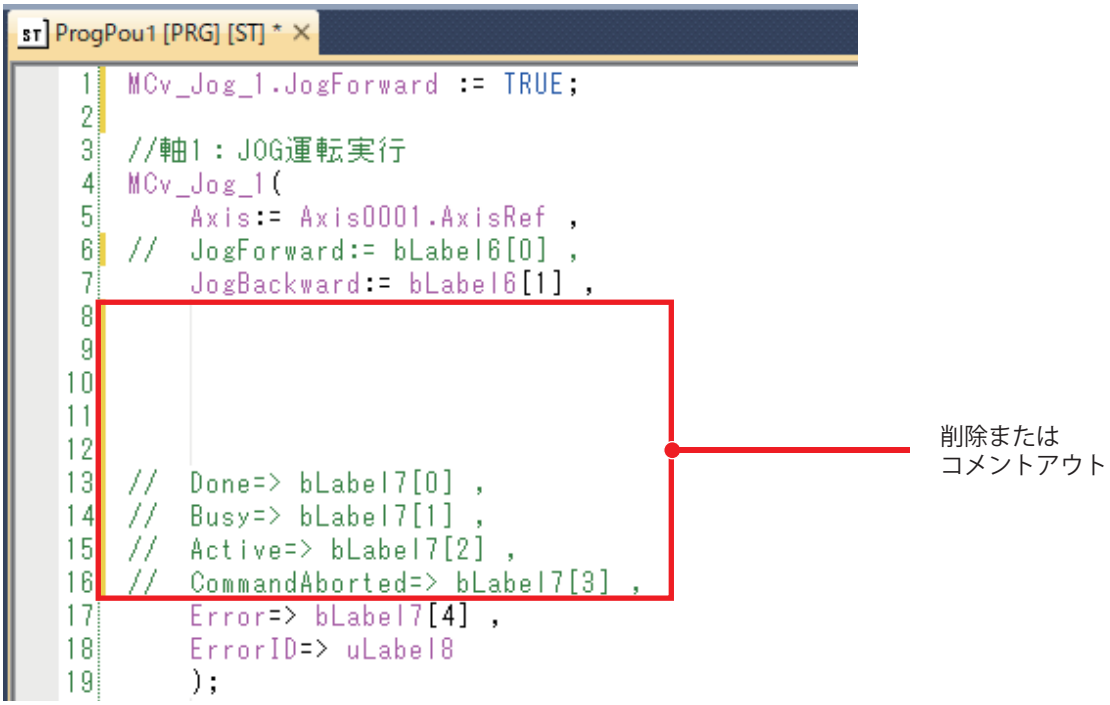
1. ラベル"bLabel6[0]"を使用せず，FBインスタンス"MCv_Jog_1"の入力変数"JogForward"に"TRUE"を直接指定します。1行目のラベル"bLabel6[0]"を"MCv_Jog_1.JogForward"に変更します。また，"MCv_Jog_1"の引数"JogForward"の設定を，コメントアウトまたは削除します。



Point

モーション制御FBのインスタンス名の後に"."(ピリオド)を入力するとラベル名の候補が表示され，候補を選択するとラベル名を自動で入力できます。
候補が表示されない場合はメニューの[ツール]⇨[オプション]⇨"編集"⇨"命令/デバイス/ラベル候補表示"⇨"動作設定"⇨"命令/デバイス/ラベル名を予測して表示"から設定できます。詳細については，下記のマニュアルを参照してください。
📖GX Works3 オペレーティングマニュアル

2. 8～16行目の未使用の引数設定をコメントアウトまたは削除します。

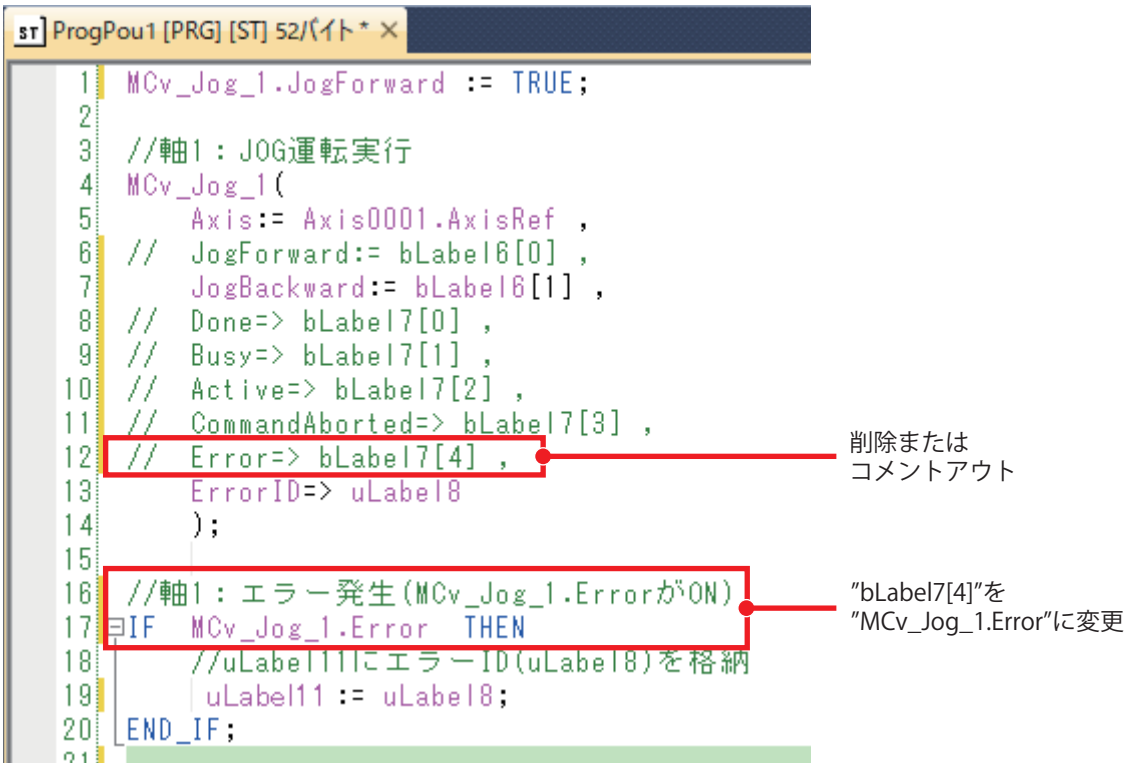


Point

使用しない，または初期値から変更しないFBの入出力変数は省略できます。

3. ラベル"bLabel7[4]"を使用せず，FBインスタンス"MCv_Jog_1"の入力変数"Error"を直接指定します。

16，17行目のラベル"bLabel7[4]"を"MCv_Jog_1.Error"に変更します。また，"MCv_Jog_1"の引数"Error"の設定を，コメントアウトまたは削除します。

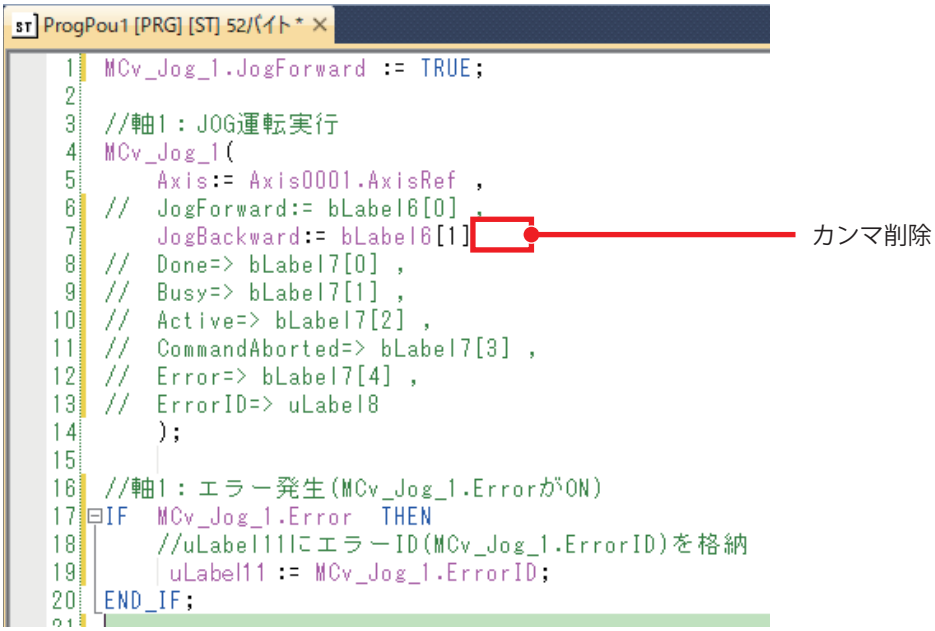


4. ラベル"uLabel8"を使用せず，FBインスタンス"MCv_Jog_1"の入力変数"ErrorID"を直接指定します。

18，19行目のラベル"uLabel8"を"MCv_Jog_1.ErrorID"に変更します。また，"MCv_Jog_1"の引数"ErrorID"の設定を，コメントアウトまたは削除します。



5. 手順1～4のコメントアウトや削除に応じて，構文にエラーがないようプログラムを修正します。
本例では，FBの引数設定の余分なカンマを削除します。



Point

入出力引数の直接指定や不要なラベルの割り当て削除によってプロジェクト全体で使用しないラベルができた場合は，ラベルエディタから不要なラベルを削除してください。

4.6 命令数が多い回路ブロックの分割

概要

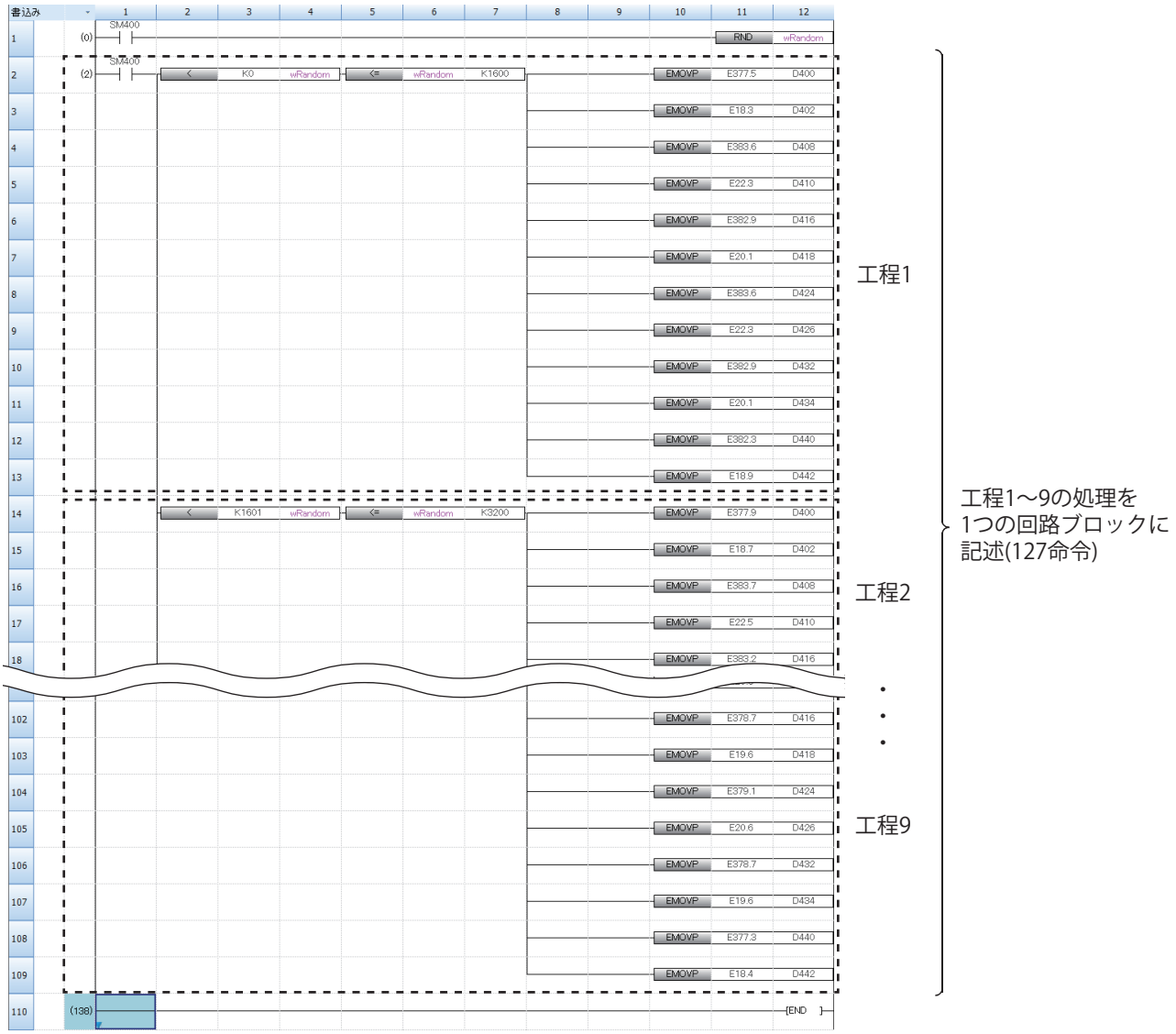
1つの回路ブロックに100個以上の命令がある場合，命令を複数の回路ブロックに分割して1つの回路ブロックに含まれる命令数を100個以下にすると，命令処理時間が短縮できます。これによりスキャンタイムを高速化できます。
回路ブロックを分割することで，下記のようなメリットもあります。

プログラムの可読性の向上

回路ブロックを複数に分割することで，プログラムの可読性が向上します。

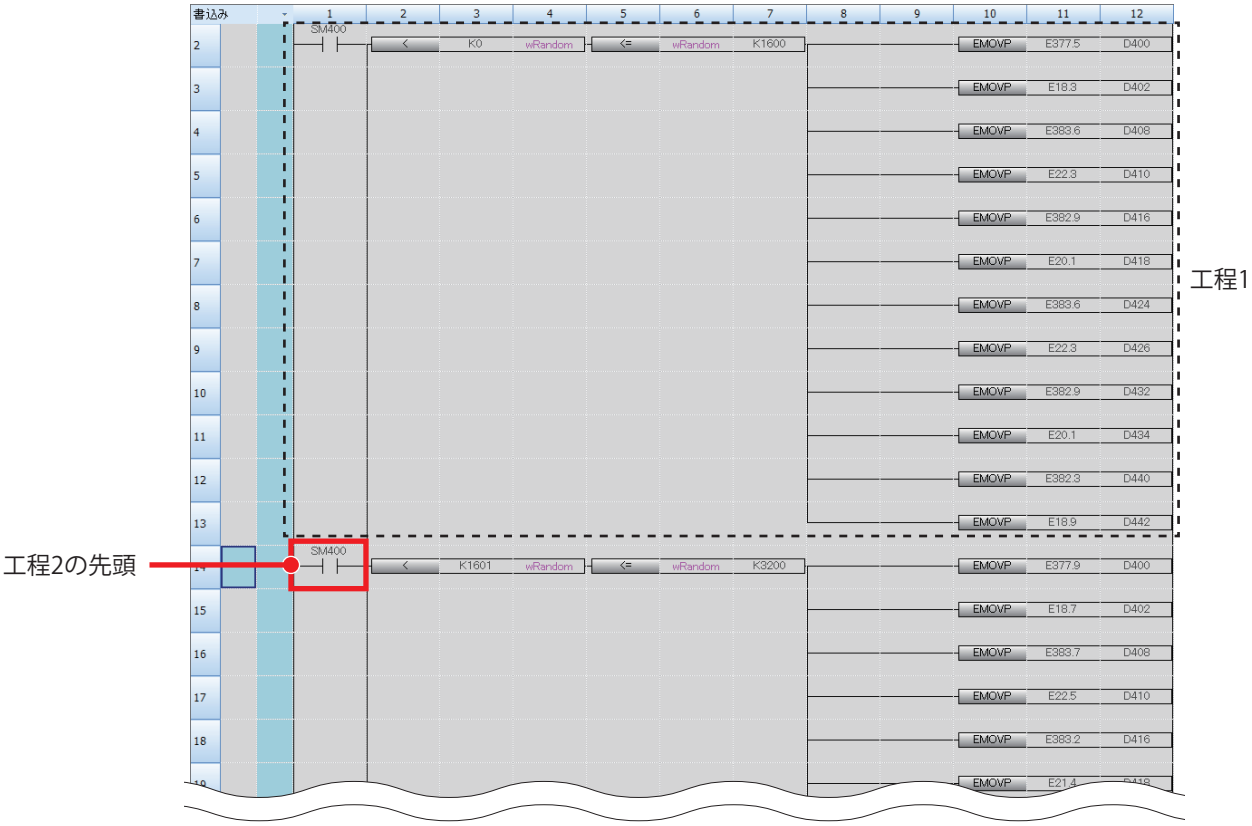
実行手順

下記のプログラムを例に，工程1～9までの命令(127命令)がある回路ブロックを工程ごとに分割する手順を示します。

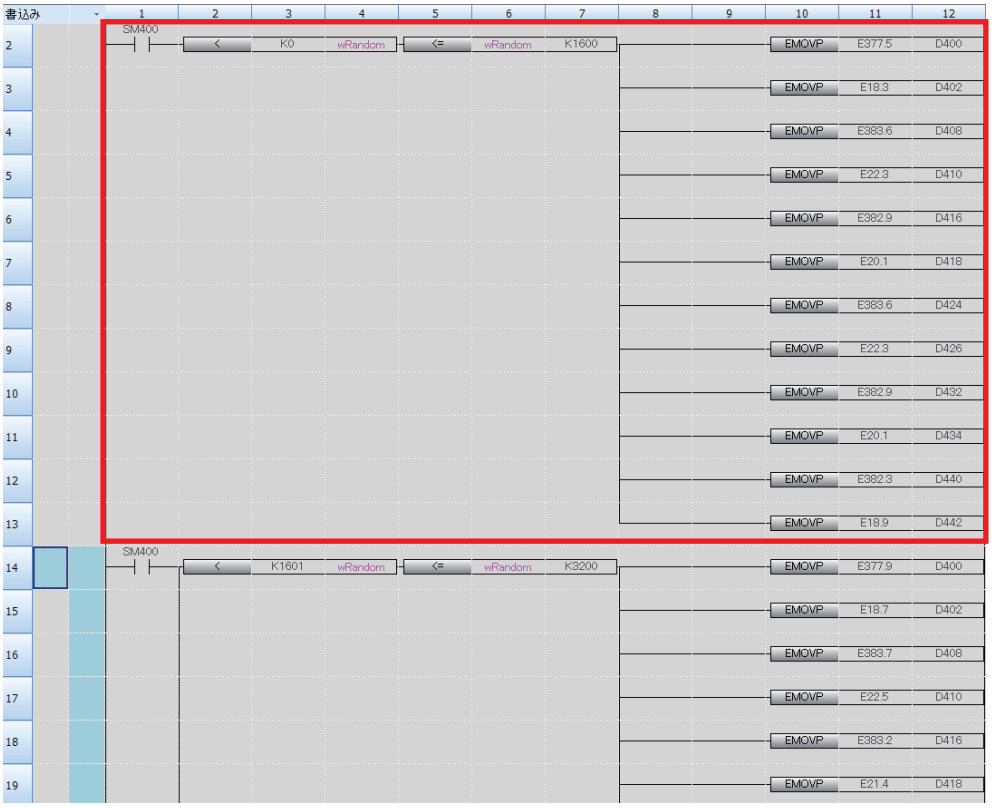


操作手順

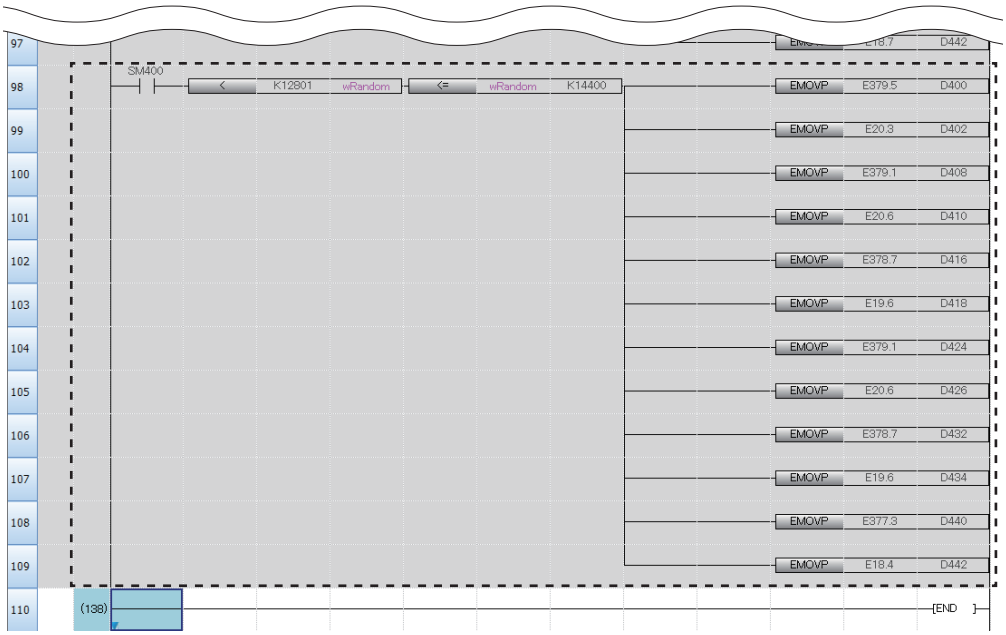
1. プログラムを開き，分割位置(工程2の先頭)に工程1の1行目と同じ接点"SM400"を追加します。



2. 工程1の接点の罫線を修正します。

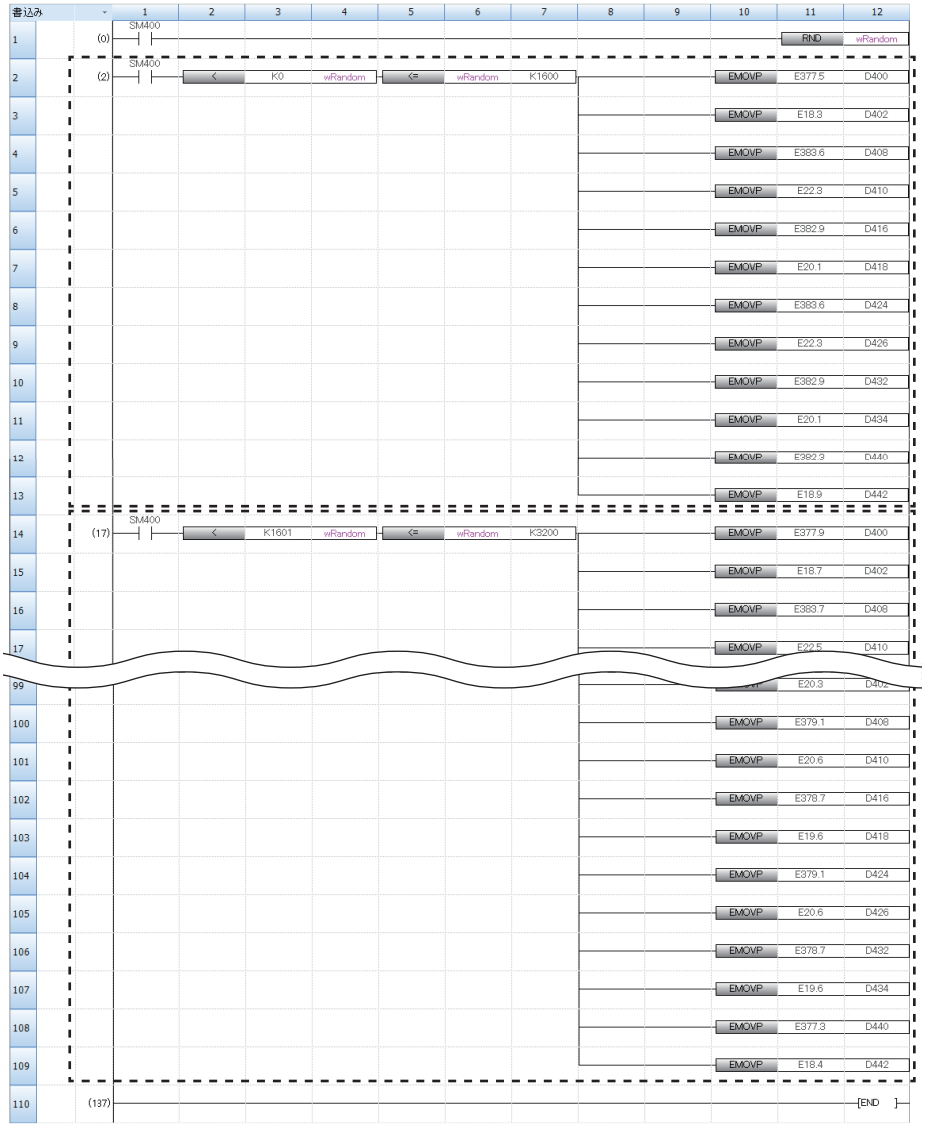


3. 手順1，2と同様の手順で工程9まで工程ごとに回路ブロックを分割します。



工程9

手順1～3によって，1工程(15命令)×9ブロックのプログラムになります。



工程1
(15命令)

工程2
(15命令)

・
・
・

工程9
(15命令)

工程ごとに
回路ブロックをわけて
記載
(15命令×9=135命令)

4.7 ラベル初期値の活用

概要

プログラムで常に値を変更する必要がないラベルやFBの初期値は，プログラム内で初期値を指定する代わりにラベルエディタから初期値を設定すると，スキャンタイムを短縮できます。
ラベル初期値を活用することで，下記のようなメリットもあります。

プログラムの可読性の向上

プログラム内の初期値を指定する処理を省略できるため，プログラムの可読性が向上します。

実行手順

下記の初期実行タイププログラムを例に，プログラム内で指定しているグローバルラベルやFBの初期値をラベルエディタから設定する手順を示します。

・初期実行タイププログラム

ProgPou1 [PRG] [ST]

```
1 //Label initialize
2 Label1 := 1;
3 Label2 := 1;
4
5 //FB initialize
6 Move1.Velocity := 1000;
7 Move1.Acceleration := 100;
8 Move1.Deceleration := 100;
```

Label1，Lable2の初期値を指定

FB"Move1"の初期値を指定

・グローバルラベル

Global [グローバルラベル設定]

<フィルタ> 簡易表示(Y) 表示設定(S) チェック(K)

	ラベル名	データ型		クラス	割付け(デバイス/ラベル)	初期値	定数
1	Label1	ワード[符号なし]/ビット列[16ビット]	...	VAR_GLOBAL			
2	Label2	ワード[符号なし]/ビット列[16ビット]	...	VAR_GLOBAL			
3	Move1	MC_MoveAbsolute	...	VAR_GLOBAL			

操作手順

1. グローバルラベルラベルエディタを開き，"Label1"，"Label2"に初期実行タイププログラムで指定している初期値と同じ内容を設定します。

・Label1: 1
・Label2: 1

Global [グローバルラベル設定]

<フィルタ> 簡易表示(Y) 表示設定(S) チェック(K)

	ラベル名	データ型		クラス	割付け(デバイス/ラベル)	初期値	定数
1	Label1	ワード[符号なし]/ビット列[16ビット]	...	VAR_GLOBAL		1	
2	Label2	ワード[符号なし]/ビット列[16ビット]	...	VAR_GLOBAL		1	
3	Move1	MC_MoveAbsolute	...	VAR_GLOBAL			

Point

"初期値"列が表示されていない場合は，[詳細表示]ボタンをクリックして表示できます。

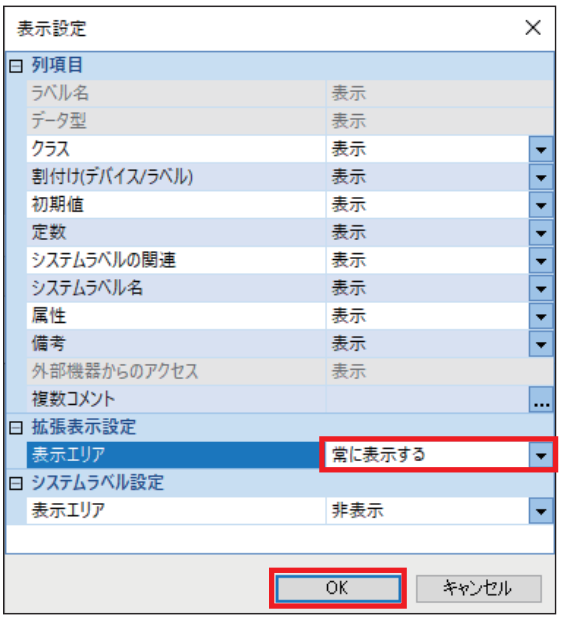
2. [表示設定]ボタンをクリックします。

Global [グローバルラベル設定]

<フィルタ> 簡易表示(Y) 表示設定(S) チェック(K)

	ラベル名	データ型		クラス	割付け(デバイス/ラベル)	初期値	定数
1	Label1	ワード[符号なし]/ビット列[16ビット]	...	VAR_GLOBAL		1	
2	Label2	ワード[符号なし]/ビット列[16ビット]	...	VAR_GLOBAL		1	
3	Move1	MC_MoveAbsolute	...	VAR_GLOBAL			

3. [拡張表示設定]⇨[表示エリア]を"常に表示する"に変更して，[OK]ボタンをクリックします。

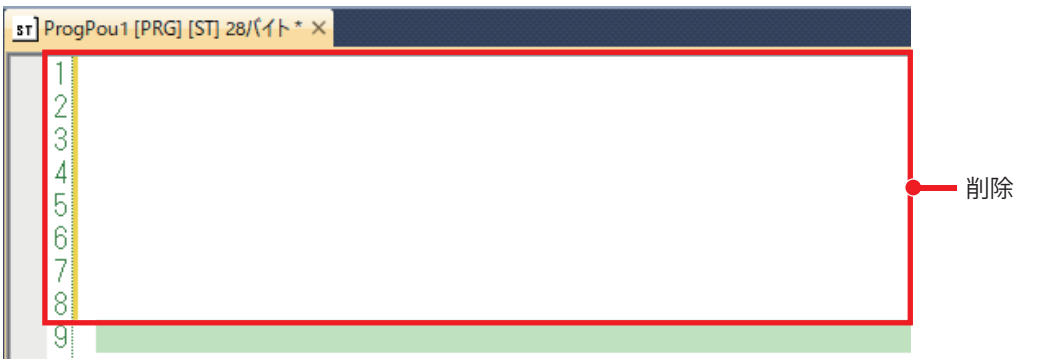


4. グローバルラベル"Move1"を選択して，拡張表示の"初期値"列に初期実行タイププログラムで指定している初期値と同じ内容を設定します。

- Velocity: 1000
- Acceleration: 100
- Deceleration: 100



5. 初期値を指定している初期実行タイププログラムを開き，初期値を指定している処理を削除します。



5 モーション制御の最適化

多軸制御の場合に特定の軸を高速化することや，ユーザプログラムとモーション制御を同期することで，モーション制御を最適化します。
本章では，モーション制御を最適化するためのテクニックについて示します。

テクニック一覧

本章で説明するテクニック一覧を以下に示します。

項目	内容	参照
通信周期混在を使用した特定軸の高速化	通信周期混在機能を使用して，多軸制御で高速な制御が必要な軸のみを高速化します。	☞ 79ページ 通信周期混在を使用した特定軸の高速化
ユーザプログラムとモーション演算の同期	マルチコアにおけるユーザプログラムとモーション演算の周期を同期し，変数の更新タイミングを同期します。	☞ 84ページ ユーザプログラムとモーション演算の同期

5.1 通信周期混在を使用した特定軸の高速化

概要

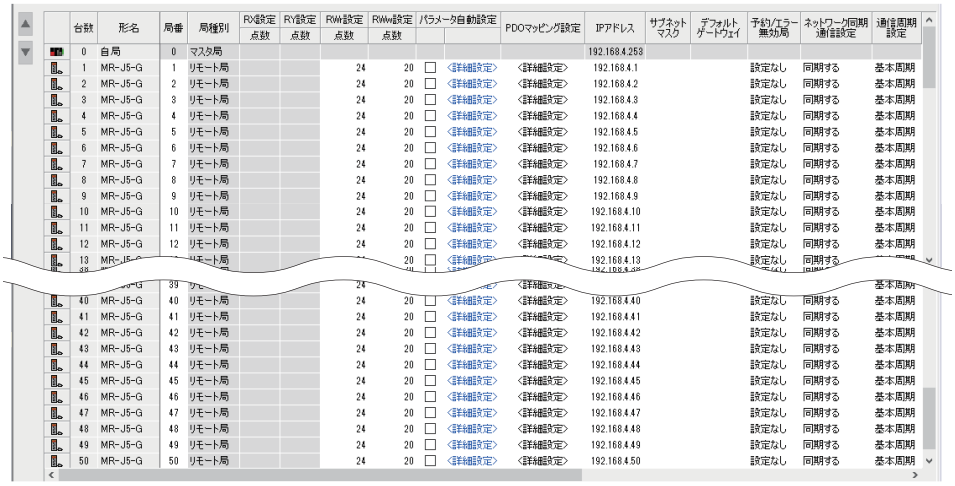
多軸制御を行うモーションプログラムで一部高速な制御が必要な軸がある場合は，通信周期混在機能を使用して軸ごとに周期を設定すると，高速な制御が必要な軸をより短い演算周期で制御することができます。また多軸構成において，全体の制御負荷を調整することができます。

Point

本テクニックは，CC-Link IE TSNの高速化のテクニックと合わせて適用するとより効果的です。CC-Link IE TSNの高速化については，下記を参照してください。
☞ 95ページ CC-Link IE TSN通信の高速化

実行手順

下記の軸構成を例に，通信周期混在機能を使用して特定軸のみ高速化する手順を示します。



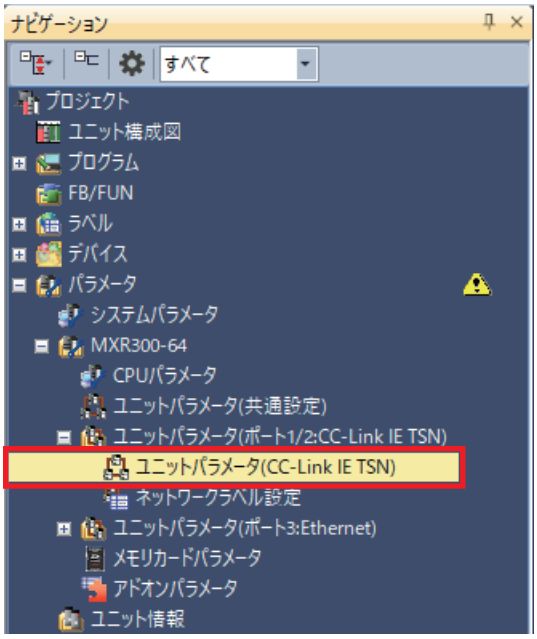
例では，以下のようにMR-J5-Gの通信周期を設定します。

局番	通信周期
1～5	基本周期
6～20	中速
21～50	低速

操作手順

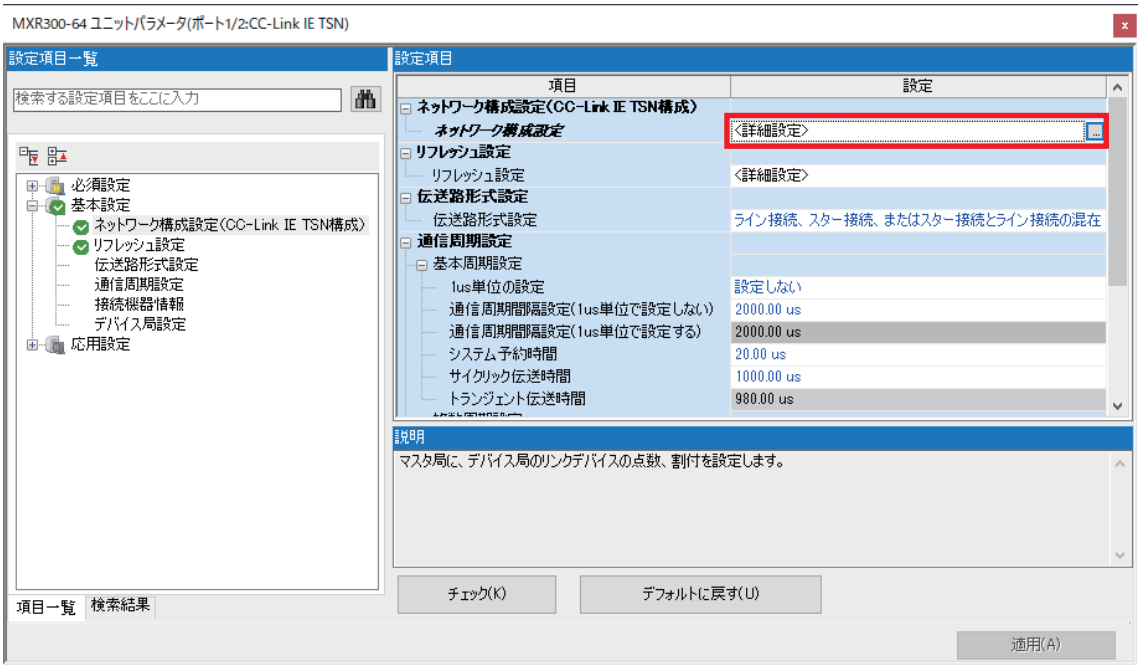
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック

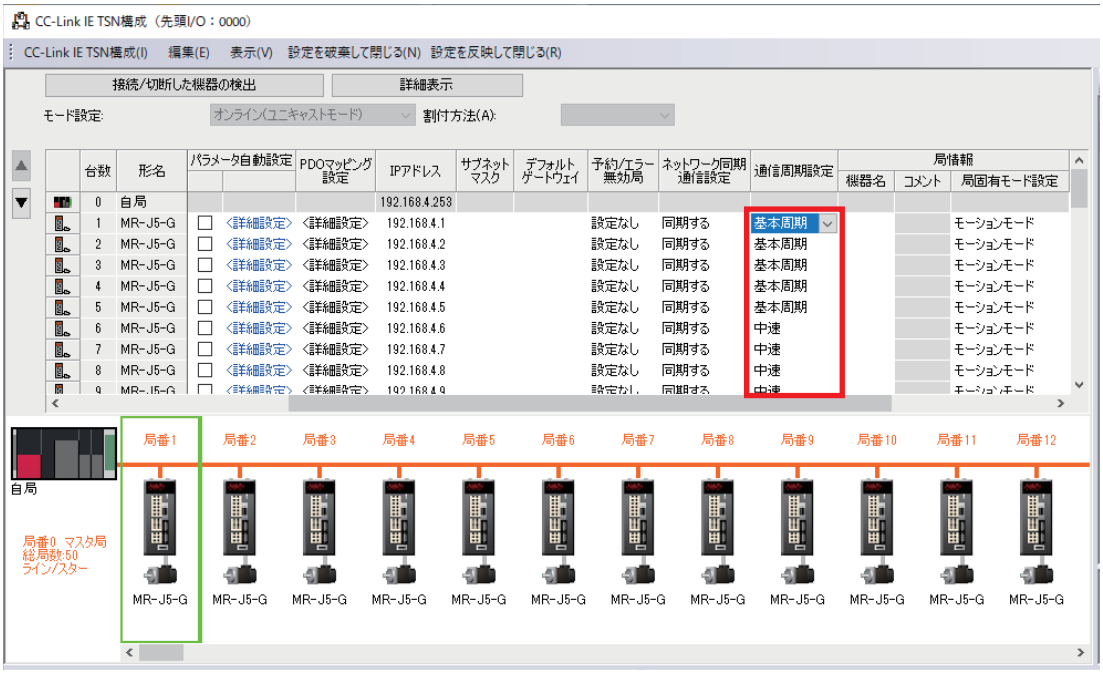


2. 基本設定から，ネットワーク構成設定を設定します。

[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



3. 「CC-Link IE TSN構成」画面の"通信周期設定"列から，各軸の通信周期を設定します。



局番	通信周期
1～5	基本周期
6～20	中速
21～50	低速

Point

基本周期の定周期間隔や中速・低速の倍率については，"ユニットパラメータ(CC-Link IE TSN)"の[基本設定]⇨[通信周期設定]から設定できます。

注意事項

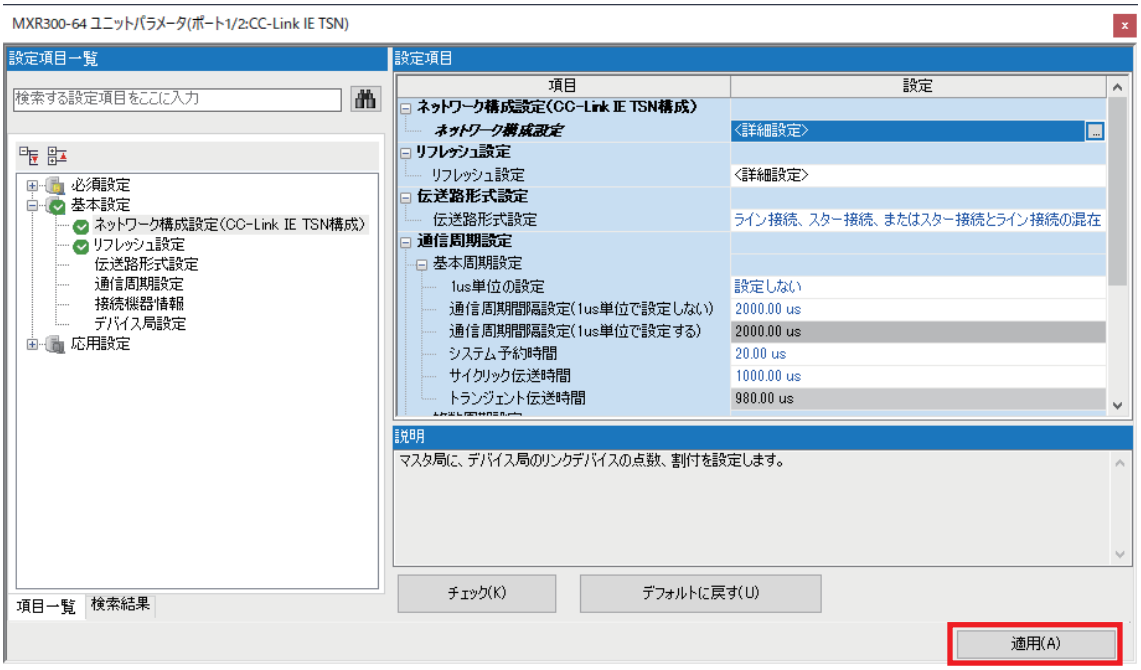
マスタ局やリモート局の通信速度によって，特定の通信周期が設定できない場合があります。マスタ局やリモート局の通信速度設定については，下記のマニュアルを参照してください。

📖MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル

4. [設定を反映して閉じる]ボタンをクリックします。

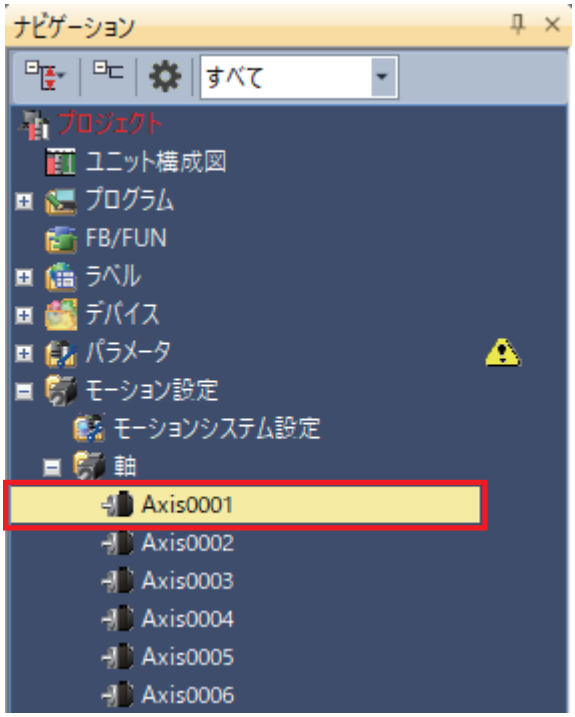


5. [適用]ボタンをクリックして設定を適用します。



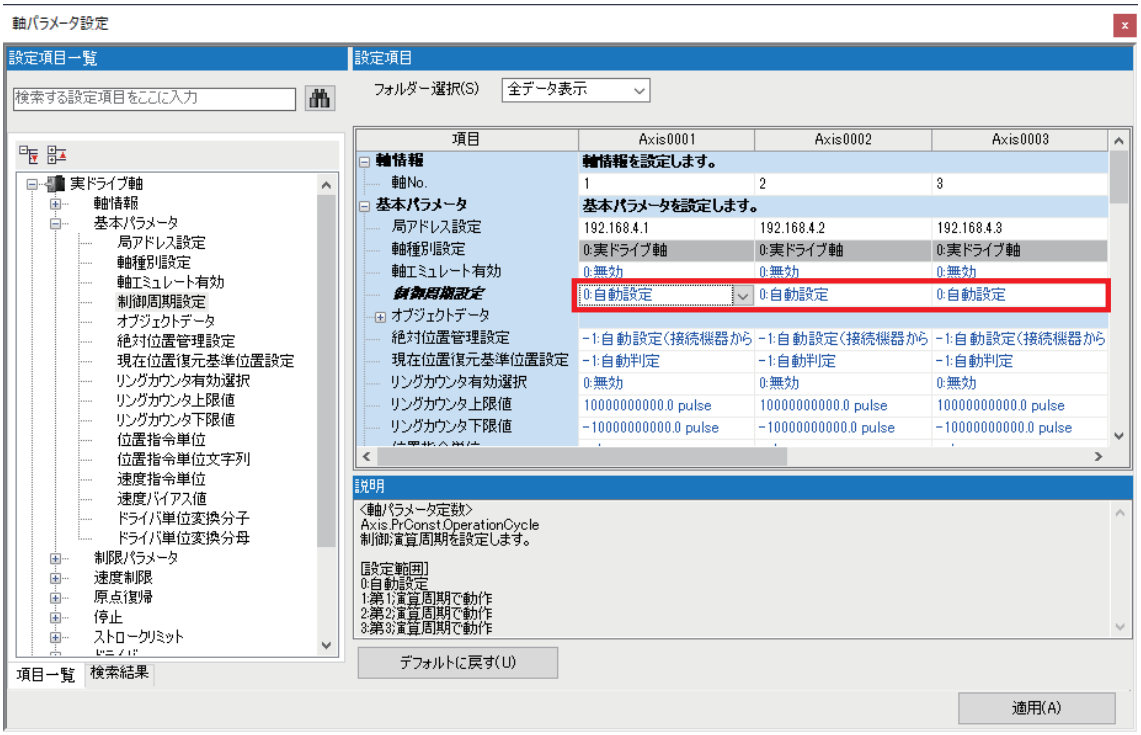
6. 各軸のパラメータを設定します。

ナビゲーションウィンドウ⇨[モーション設定]⇨[軸]⇨任意の軸をダブルクリック

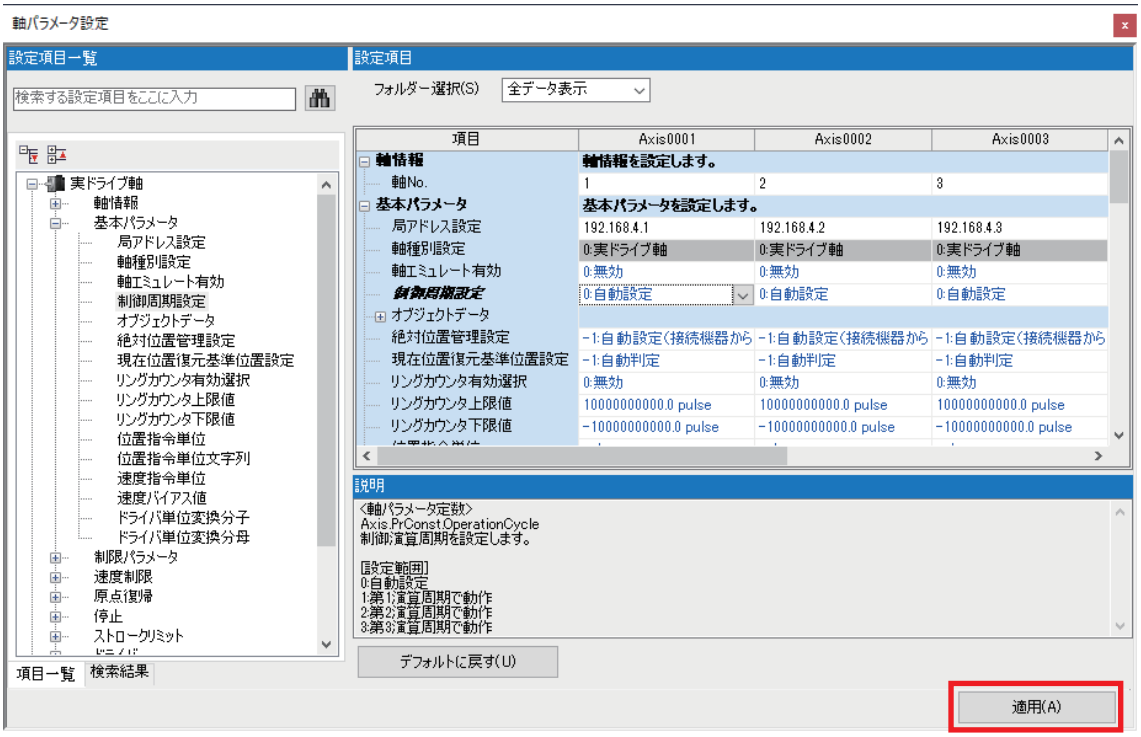


7. 各軸の制御周期設定を"0:自動設定"(デフォルト)に設定します。

設定する軸種別⇨[基本パラメータ]⇨[制御周期設定]



8. 手順7で設定を変更した場合は，[適用]ボタンをクリックして設定を適用します。



注意事項

異なる周期の軸同士を同期制御する場合は制約があります。詳細については，下記のマニュアルを参照してください。

📖MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル

5.2 ユーザプログラムとモーション演算の同期

概要

コントローラは，ユーザプログラムとモーション演算をマルチコア上で並列に実行します。そのため，モーション演算の周期と異なる実行周期のユーザプログラムからモーション演算で使用する変数(AxisName.Md.SetPositionなどのモーション機能部が更新する変数)にアクセスすると，変数の更新タイミングが不定になる可能性があります。
変数の更新タイミングが重要な場合は，モーション演算の周期に同期したプログラムを作成して，ユーザプログラムとモーション演算が使用する変数を同期します。

モーション演算の周期に同期するプログラム

プログラムの実行周期はネットワーク通信周期の先頭を起点として，常にプログラム→モーション演算という実行順序でモーション演算周期と同期できます。
モーション演算の周期に同期するネットワーク通信周期，プログラムの組み合わせを以下に示します。

モーション演算周期	同期するネットワーク通信周期	同期するプログラム		
		実行タイプ	トリガ種別	実行周期
第1演算周期	基本周期	イベント	ネットワーク通信周期同期	基本周期
第2演算周期	中速周期			中速周期
第3演算周期	低速周期			低速周期

実行手順

モーション演算で使用する変数"AxisName.Md.SetPosition"(指令現在位置)，"AxisName.Cd.VelocityOverride"(速度オーバーライド係数)を例に，これらの変数の更新タイミングとプログラムを同期する手順を示します。

操作手順

1. 任意のグローバルラベルエディタを開き，変数"AxisName.Md.SetPosition"，"AxisName.Cd.VelocityOverride"を退避するためのグローバルラベル(以降テンポラリラベル)を定義します。

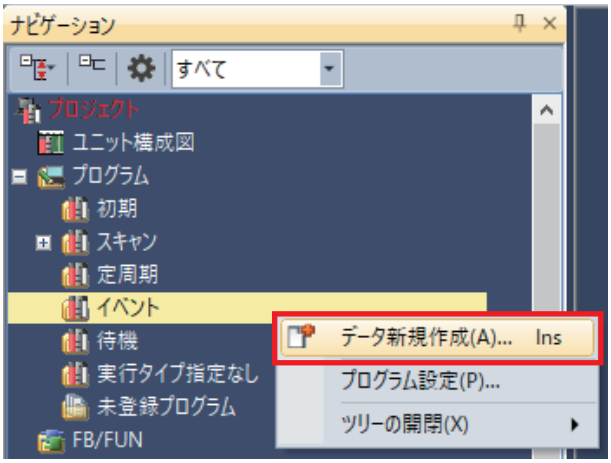
本手順では，以下のテンポラリラベルを定義します。



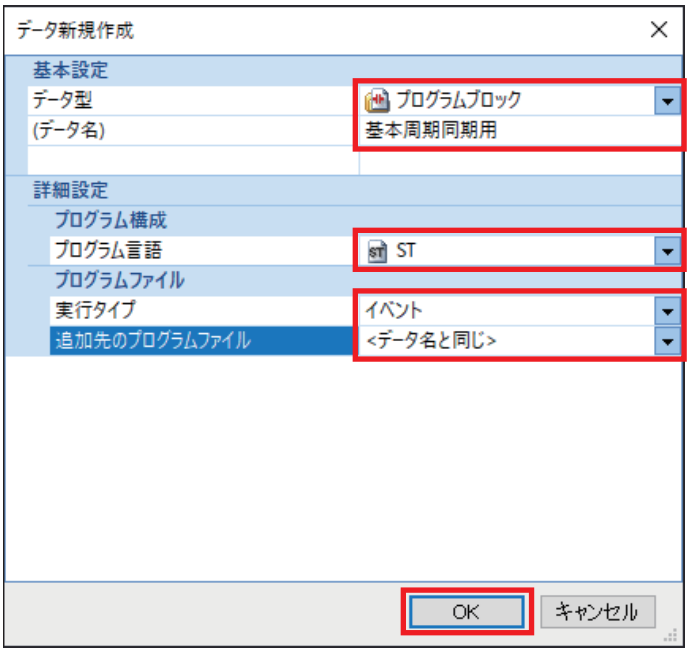
No.	ラベル名	データ型	クラス	備考
1	temp_NormalSpeed_Read_Axis0001_SetPosition	倍精度実数	VAR_GLOBAL	中速周期へのアクセス用の指令現在位置テンポラリラベルです。
2	temp_NormalSpeed_Write_Axis0001_VelocityOverride	倍精度実数	VAR_GLOBAL	中速周期へのアクセス用の速度オーバライドテンポラリラベルです。
3	temp_LowSpeed_Read_Axis0001_SetPosition	倍精度実数	VAR_GLOBAL	低速周期へのアクセス用の指令現在位置テンポラリラベルです。
4	temp_LowSpeed_Write_Axis0001_VelocityOverride	倍精度実数	VAR_GLOBAL	低速周期へのアクセス用の速度オーバライドテンポラリラベルです。

2. モーション演算と基本周期を同期させるイベント実行タイプのプログラムブロックを新規作成します。

ナビゲーションウィンドウ⇨[プログラム]⇨[イベント]を右クリック⇨[データ新規作成]



3. 各項目を設定し，[OK]ボタンをクリックします。

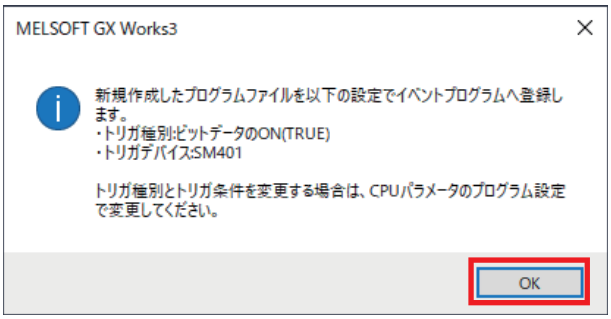


項目	設定内容
データ型	プログラムブロック
(データ名)	基本周期同期用
プログラム言語	ST
実行タイプ	イベント
追加先のプログラムファイル	<データ名と同じ>

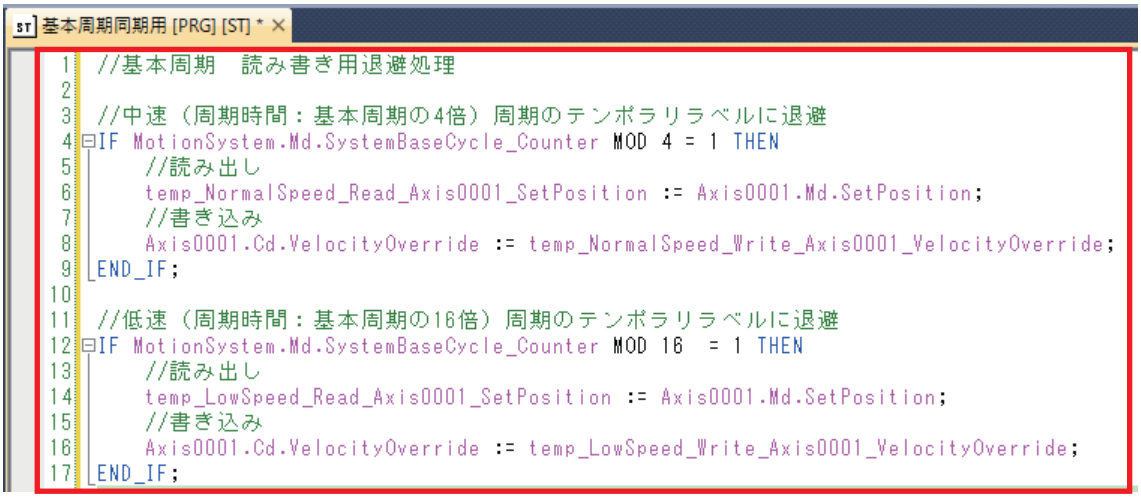
Point

新規にプログラムファイルとプログラムブロックを作成する際，プログラムファイルとプログラムブロック名を同じにする場合は，プログラムブロック作成時に"追加先のプログラムファイル"を「<データ名と同じ>」に設定すると，同じ名前のプログラムファイルが同時に作成されます。
これにより，プログラムファイルの新規作成を省略することができます。

4. [OK]ボタンをクリックします。



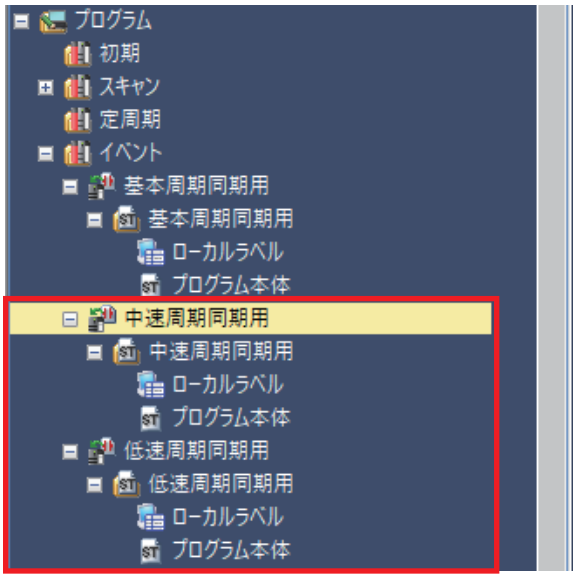
5. 手順3で作成したプログラムブロックのプログラム本体を開き，変数を基本周期に同期させる以下の処理を記述します。



行数	処理内容
3～9行目	第1演算周期で制御する軸の軸ラベルに中速周期(基本周期の4倍)のプログラムからアクセスする場合，システム基本周期カウンタ*1を4で割った余りが1となる周期で退避，復元処理を実行します。 ・読出し時: テンポラリラベルに変数"AxisName.Md.SetPosition"の値を退避します。 ・書込み時: テンポラリラベルの値を変数"AxisName.Cd.VelocityOverride"に復元します。
11～17行目	第1演算周期で制御する軸の軸ラベルに低速周期(基本周期の16倍)のプログラムからアクセスする場合，システム基本周期カウンタ*1を16で割った余りが1となる周期で退避，復元処理を実行します。 ・読出し時: テンポラリラベルに変数"AxisName.Md.SetPosition"の値を退避します。 ・書込み時: テンポラリラベルの値を変数"AxisName.Cd.VelocityOverride"に復元します。

*1　システム基本周期カウンタ(System.Md.SystemBaseCycle_Counter)は，ネットワークの基本周期間隔で1カウントアップを行い，各モーション演算周期の最小公倍数分の周期を超えると1へ戻るカウンタです。本カウンタにて各モーション演算周期処理の実行タイミングを確認することができます。
詳細については，下記のマニュアルを参照してください。
📖MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル

6. 手順2～4を参考に，中速周期と低速周期に同期するイベント実行タイププログラムブロックを同様に作成します。



7. 手順6で作成した中速周期と低速周期に同期するイベント実行タイププログラムブロックを開き，変数"AxisName.Md.SetPosition"の値に応じて変数"AxisName.Cd.VelocityOverride"に値を設定する以下の処理を記述します。

・中速周期に同期するイベント実行タイププログラム

中速周期同期用 [PRG] [ST] *

```
//第1演算周期で動作する軸Axis0001のMd.SetPositionが1000.0以下の場合
//第1演算周期で動作する軸Axis0001のCd.VelocityOverrideに10.0を設定する
IF temp_NormalSpeed_Read_Axis0001_SetPosition <= 1000.0 THEN
    temp_NormalSpeed_Write_Axis0001_VelocityOverride := 10.0;
END_IF;
```

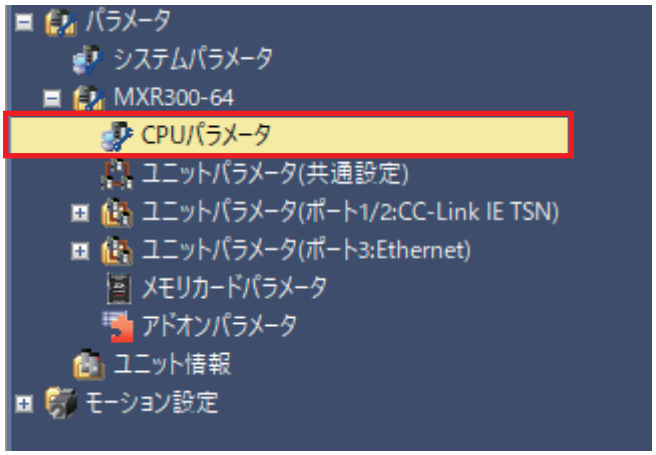
・低速周期に同期するイベント実行タイププログラム

低速周期同期用 [PRG] [ST] *

```
//第1演算周期で動作する軸Axis0001のMd.SetPositionが1000.0を超えたら
//第1演算周期で動作する軸Axis0001のCd.VelocityOverrideに5.0を設定する
IF temp_NormalSpeed_Read_Axis0001_SetPosition > 1000.0 THEN
    temp_NormalSpeed_Write_Axis0001_VelocityOverride := 5.0;
END_IF;
```

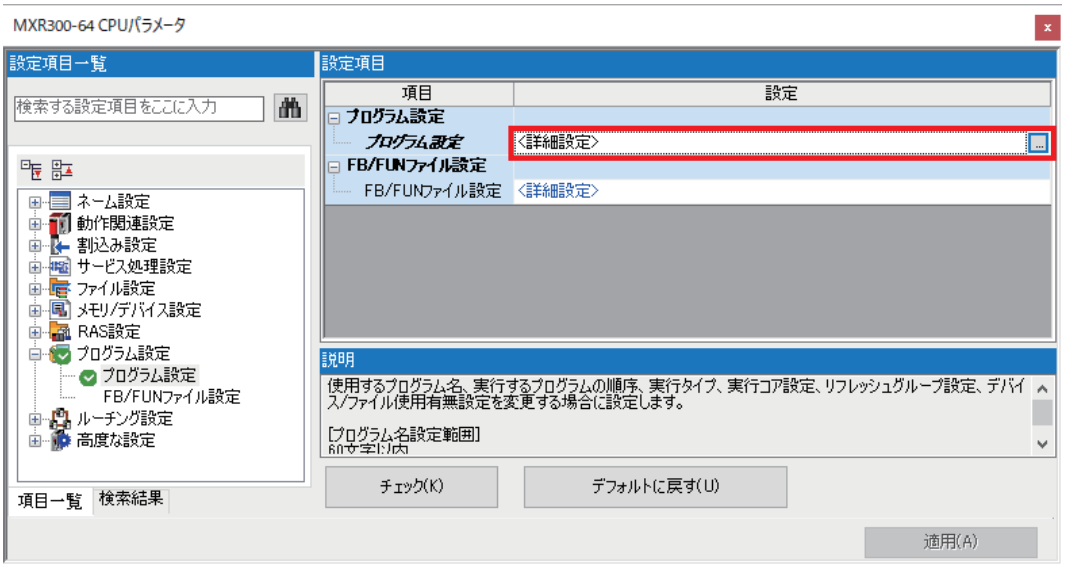
8. CPUパラメータを開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[CPUパラメータ]をダブルクリック

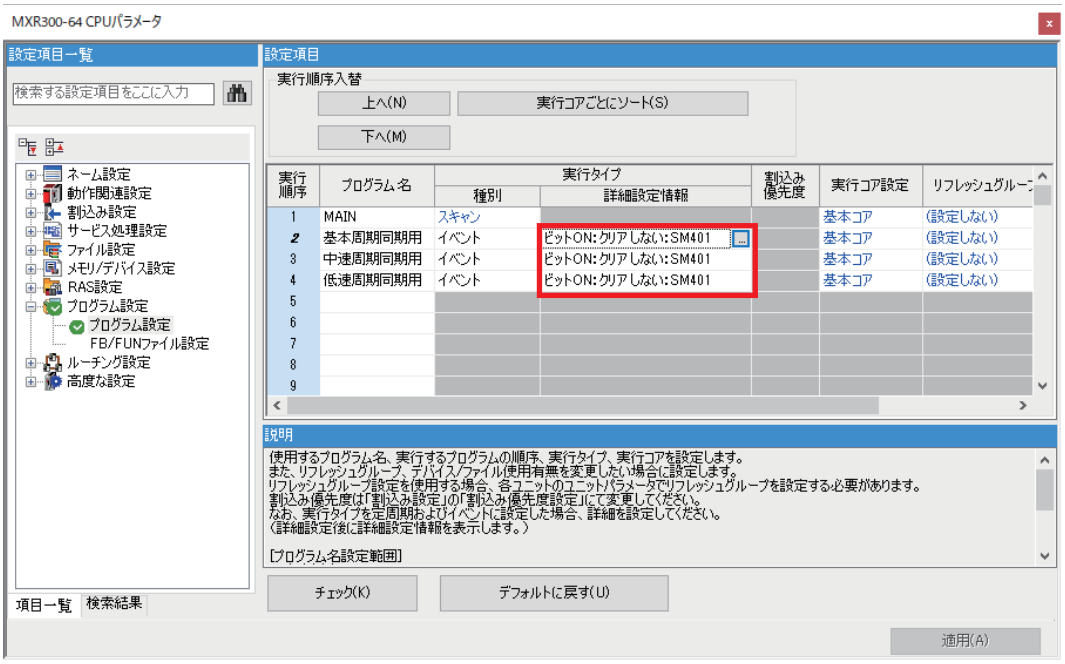


9. イベント実行タイププログラムのトリガ設定を変更するため，プログラム設定を開きます。

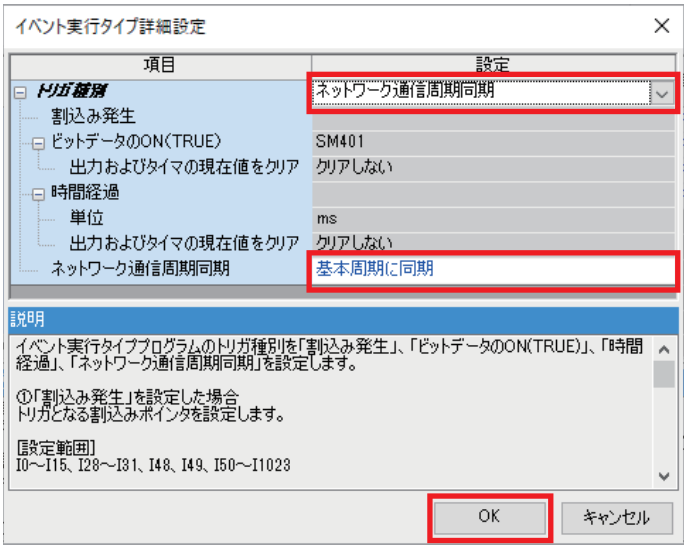
[プログラム設定]⇨[プログラム設定]の<詳細設定>をダブルクリック



10. 本手順で作成したイベント実行タイププログラムの"詳細設定情報"列をダブルクリックします。

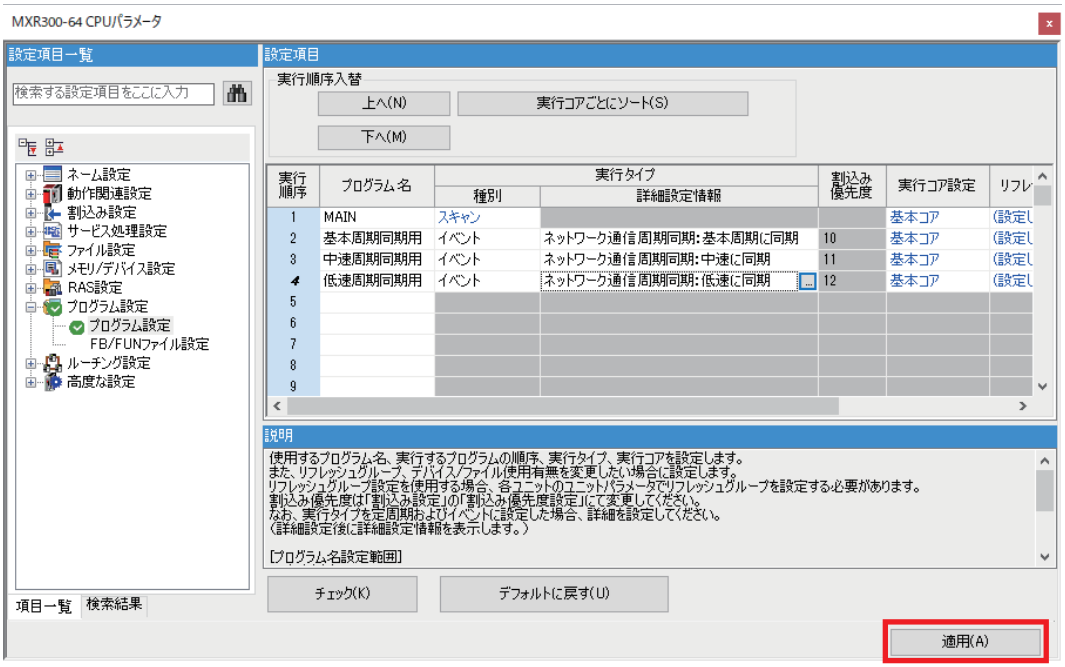


11. "トリガ種別"と"ネットワーク通信周期同期"をプログラムの内容に合わせて以下のように設定し，[OK]ボタンをクリックします。



プログラム	トリガ種別	ネットワーク通信周期同期
基本周期と同期するイベント実行タイププログラム (本手順では「基本周期同期用」)	ネットワーク通信周期同期	基本周期に同期
中速周期と同期するイベント実行タイププログラム (本手順では「中速周期同期用」)		中速に同期
低速周期と同期するイベント実行タイププログラム (本手順では「低速周期同期用」)		低速に同期

12. [適用]ボタンをクリックして設定を適用します。



注意事項

- テンポラリラベルへの退避，復元処理を行うプログラムよりも優先度の高いプログラムから退避先のテンポラリラベルにアクセスする場合は，DI命令/EI命令を使用し，割込み禁止状態で退避，復元処理を行ってください。

ProgPou1 [PRG] [ST] 24バイト

```
1  //読み書き用退避処理
2  //割込み禁止
3  DI(TRUE);
4  //低速（周期時間：基本周期の16倍）周期のテンポラリラベルに退避
5  IF MotionSystem.Md.SystemBaseCycle_Counter MOD 16  = 1 THEN
6      //読み出し
7      temp_LowSpeed_Read_Axis0001_SetPosition := Axis0001.Md.SetPosition;
8      //書込み
9      Axis0001.Cd.VelocityOverride := temp_LowSpeed_Write_Axis0001_VelocityOverride;
10 END_IF;
11 //割込み禁止状態解除
12 EI(TRUE);
13
```

- 本テクニックによりテンポラリラベルへの退避，復元処理が追加されるため，プログラムの複雑化やスキャンタイムの増大が発生する可能性があります。

6 ラベルの効率化

グローバルラベルである必要がないラベルをローカルラベルに変更することや，不要なラベルを削除することで，ラベルを効率化します。ラベルの効率化によって，プログラム全体の可読性も向上します。
本章では，ラベルの効率化を行うためのテクニックについて示します。

テクニック一覧

本章で説明するテクニック一覧を以下に示します。

項目	内容	参照
グローバルラベルの削減	グローバルラベルである必要がないラベルをローカルラベルに変更します。	☞91ページ グローバルラベルの削減
未使用ラベルの削除	プロジェクト内の未使用ラベルを検索して一括削除します。	☞93ページ 未使用ラベルの削除

6.1 グローバルラベルの削減

概要

グローバルラベルは，ラベルを複数のプログラムで使用する必要がある場合やラベルにデバイスを割り当てたい場合に使用するラベルです。それ以外の場合に使用しているグローバルラベルはローカルラベルに変更することで，グローバルラベルを削減できます。
グローバルラベルを削減することで，下記のようなメリットもあります。

プログラムの不具合発生リスクの減少

プログラムに対するラベルの影響範囲が局所化するため，不具合発生のリスクを抑えます。

プログラム変換時間の短縮

プログラムに対するラベルの影響範囲が局所化するため，プログラム変換時の影響を抑え，プログラム変換時間を短縮できます。

実行手順

不要なグローバルラベルを検索して削除し，ローカルラベルに変更する手順を示します。

操作手順

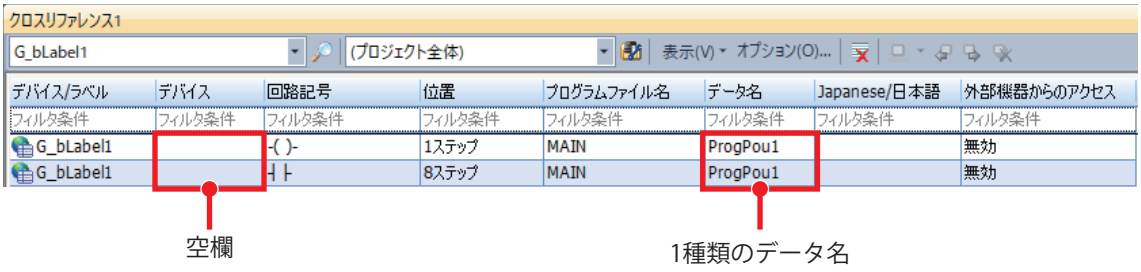
1. 各グローバルラベルの使用状況を，クロスリファレンスを使用して確認します。
グローバルラベルを開き，確認したいラベルを選択した状態でCtrl+Eキーを押します。



ラベルを選択した状態でCtrl+Eキーを押す

2. クロスリファレンス結果が表示されます。

"デバイス"列が空欄，かつ"データ名"列に1種類のデータ名しか表示されていないラベルは，グローバルラベルの必要がないと判断できます。



Point

クロスリファレンス結果からグローバルラベルをダブルクリックすると，ラベルが使用されているプログラムと位置を確認できます。グローバルラベル削除前に確認してください。

3. グローバルラベルエディタから，削除するグローバルラベルを切り取ります。

エディタの表の行番号をクリックし，行全体を選択した状態で切り取ってください。

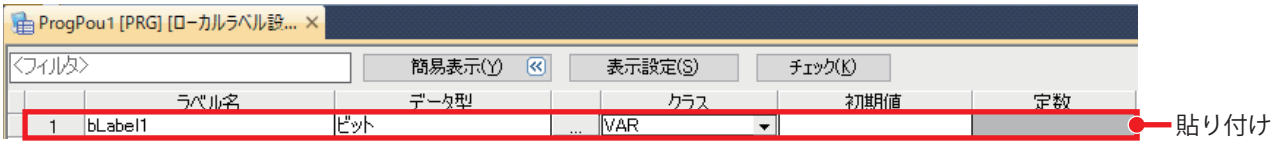


Point

行全体を選択していない場合，ラベルエディタ上で非表示になっている内容はコピーされません。

4. 削除するグローバルラベルが使用されているプログラムのローカルラベルを開き，手順3で切り取ったラベルを貼り付けます。

貼り付け後，任意のクラスを設定します。またローカルラベルやグローバルラベルで命名規則を決めている場合は，必要に応じてラベル名を変更します。



Point

- グローバルラベルからローカルラベルへコピーした内容を貼り付ける際，クラスの設定は貼り付けされません。任意のクラスを設定し直してください。
- プログラム上で使用しているラベルの名前は，手順1～4の変更に応じて自動で変更されます。そのため，プログラムを修正する必要はありません。変更されない場合はメニューの[ツール]⇨[オプション]⇨"その他のエディタ"⇨"ラベルエディタ共通"⇨"動作設定"⇨"ラベル名をプログラムエディタに自動追従させる"から設定できます。詳細については，下記のマニュアルを参照してください。

GX Works3 オペレーティングマニュアル

5. 他のグローバルラベルに対しても，手順1～4を同様に行います。

注意事項

削除するグローバルラベル(ローカルラベルに移動するラベル)は，外部機器からのアクセスや他プログラムでの使用ができません。グローバルラベルを削除する前にラベルが使用されているプログラム内容を確認して，削除の良否を判断してください。

6.2 未使用ラベルの削除

概要

GX Works3のクロスリファレンス機能では，プロジェクト内で使用していないラベルを検索して削除することができるためラベルを効率化できます。
未使用ラベルを削除することで，下記のようなメリットもあります。

ラベル使用可能量の増加

未使用ラベルの削除により，使用できるラベル容量が増えます。

プログラムの検索時間の短縮

未使用ラベルの削除により，プロジェクト内でラベルを検索する際に検索対象が減少するため，検索時間が短縮できます。

プログラム変換時間の短縮

未使用ラベルの削除によりラベルのデータ量が減少するため，プログラム変換時間を短縮できます。

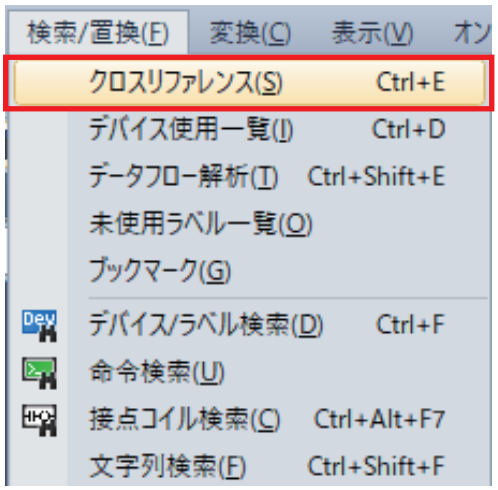
実行手順

未使用ラベルを削除する手順を示します。

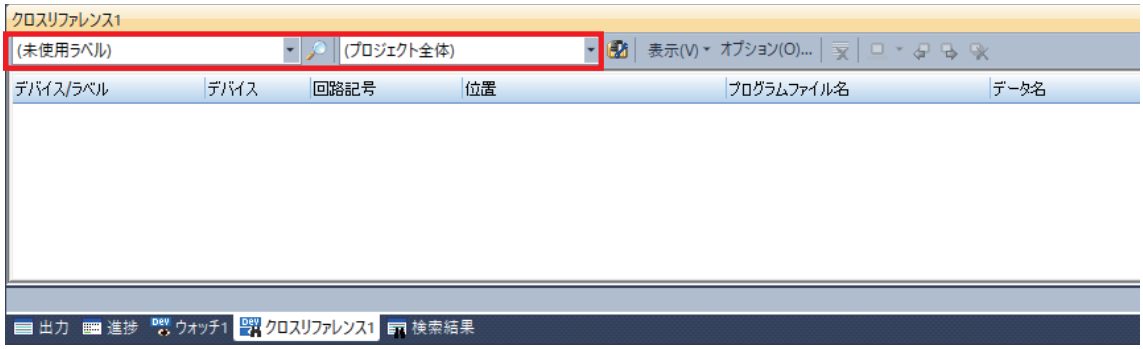
操作手順

1. 未使用ラベルを検索するクロスリファレンスを開きます。

メニューの[検索/置換]⇨[クロスリファレンス]

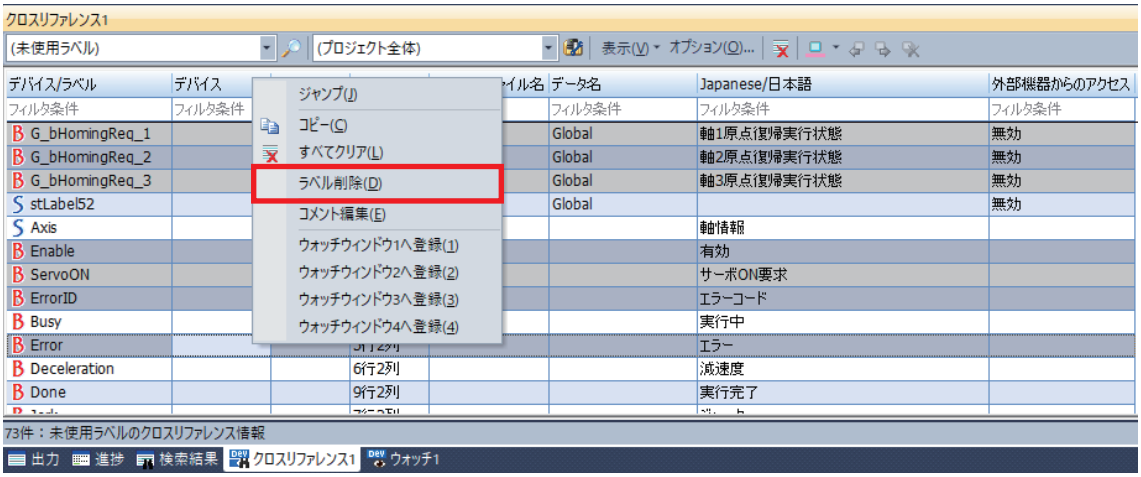


2. 検索範囲を"(未使用ラベル)"，"(プロジェクト全体)"に設定し，[ ]ボタンをクリックします。



3. プロジェクト内の未使用ラベル一覧が表示されます。

Ctrlキーを押下しながら削除するラベルを複数選択し，右クリックから[ラベル削除]を選択してラベルを削除します。



Point

- Ctrl+Aキーでラベルを全選択して一括削除することもできますが，1つずつ内容を確認して削除することを推奨します。
- ラベルを削除した後は再度未使用ラベルの検索を行うと，未使用ラベルが削除されていることを確認できます。

7 CC-Link IE TSN通信の高速化

デバイス局のリンクリフレッシュ設定や通信周期設定などを修正して，リンクリフレッシュ時間や通信周期間隔，サイクリック伝送時間を短縮することで，CC-Link IE TSN通信を高速化できます。CC-Link IE TSN通信の高速化によってより高速な制御の実現が可能になり，装置のタクトタイムが向上します。
本章では，CC-Link IE TSN通信の高速化を行うためのテクニックについて示します。

テクニック一覧

本章で説明するテクニック一覧を以下に示します。

項目		内容	参照
リンクリフレッシュ時間の短縮		リフレッシュ設定のデバイスの範囲が連続するようにリンクデバイス番号を再割付します。	☞ 96ページ リンクリフレッシュ時間の短縮
通信周期間隔/サイクリック伝送時間の短縮	デバイス局の台数の均等化	マスタ局の各ポートに接続するデバイス局の台数を均等にします。	☞ 103ページ デバイス局の台数の均等化
	送信上限サイズの設定	CC-Link IE TSN Classが「Class A」かつ，通信周期設定が「低速」のデバイス局の場合，CC-Link IE TSN Class Aグループの送信上限サイズを小さく設定します。	☞ 104ページ 送信上限サイズの設定
	デバイス局の台数の削減	不要なデバイス局がある場合，マスタ局に接続しているデバイス局の台数を削減します。	☞ 106ページ デバイス局の台数の削減
	リンクデバイス設定の削除	各デバイス局の不要なリンクデバイス設定を削除します。	☞ 110ページ リンクデバイス設定の削除
	リンクデバイス設定の点数の削減	各デバイス局のリンクデバイス設定の点数を削減します。	☞ 115ページ リンクデバイス設定の点数の削減
	ネットワーク同期通信設定の統一	CC-Link IE TSN Classが「Class B」のデバイス局の中で，ネットワーク同期通信設定が「同期しない」と「同期する」で混同している場合は，どちらかに統一します。	☞ 120ページ ネットワーク同期通信設定の統一
	サイクリック伝送分散の設定	CC-Link IE TSN Classが「Class B」かつ，通信周期設定が「中速」や「低速」のデバイス局の場合，サイクリック伝送分散を設定します。	☞ 123ページ サイクリック伝送分散の設定

通信周期間隔とサイクリック伝送時間の確認方法

通信周期間隔の実測値とサイクリック伝送時間の算出値は，以下のバッファメモリから確認できます。
各テクニックを実行した際の通信周期間隔やサイクリック伝送時間の確認に使用してください。

- 通信周期間隔(実測値): (U3E0￥G10200～U3E0￥G10201)
- サイクリック伝送時間(算出値): (U3E0￥G10272～U3E0￥G10273)

なお通信周期間隔は実測値のため，コントローラのリセットを行うと，表示される値が多少増減することがあります。

7.1 リンクリフレッシュ時間の短縮

概要

ネットワーク構成設定で設定されたデバイス局のRX設定/RY設定，RWr設定/RWw設定のデバイス番号を”通信周期間隔短縮割付”で再割付することで，リフレッシュ先のデバイス範囲が連続して設定されます。これによりリフレッシュ設定の設定数が少なくなるため，リフレッシュ設定を簡素化します。また，リンクリフレッシュ時間や通信周期間隔，サイクリック伝送時間の短縮ができるため，CC-Link IE TSN通信の高速化を実現します。

通信周期間隔短縮割付

ネットワーク構成設定のデバイス番号再割付機能では，指定した対象局のリンクデバイスにデバイス番号を連続して割り付けることができます。
デバイス番号再割付機能の割付方法で”通信周期間隔短縮割付”を選択すると，サイクリック受信処理時間を高速化するデバイス番号の割付けを全局に対して実行します。
通信周期間隔短縮割付ではネットワーク構成設定で設定されたデバイス局のRX設定，RY設定，RWr設定，RWw設定のデバイス番号を，以下の順で連続して割り付けます。(各設定内のデバイス番号は局番順に割り付けられます)

割り付け順番	ネットワーク同期通信設定	通信周期設定
1	同期しない	基本周期
2	同期しない	中速
3	同期しない	低速
4	同期する	基本周期
5	同期する	中速
6	同期する	低速

通信周期間隔短縮割付の詳細については，下記のマニュアルを参照してください。
📖MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル

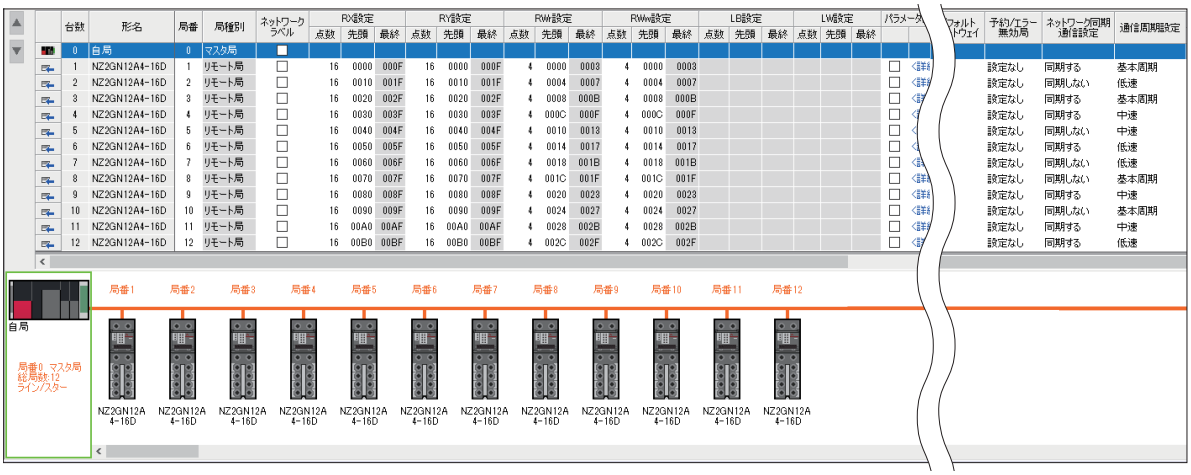
注意事項

- リモート局のリンクデバイスにすでにデバイス番号が割り付いている場合に通信周期間隔短縮割付を行うと，既存のリンクデバイス番号の順番が変更されます。
- 通信周期間隔短縮割付は，ネットワーク同期通信設定を「同期する」に設定しているデバイス局が存在する場合に有効です。

実行手順

下記のネットワーク構成とリフレッシュ設定を例に，リンクデバイス番号を再割付する手順を示します。

• ネットワーク構成



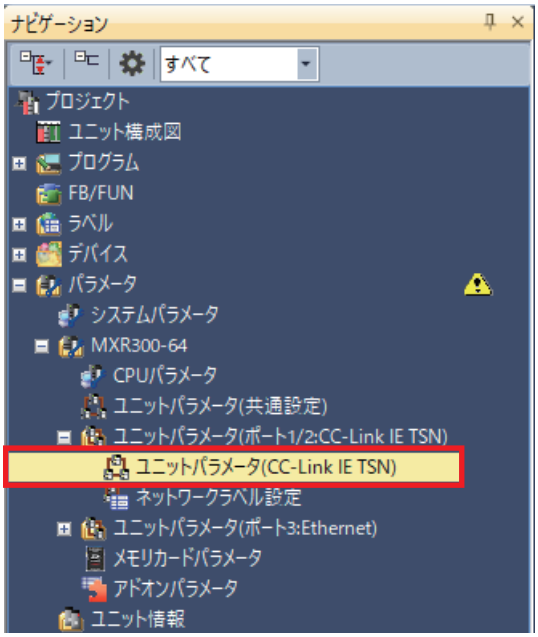
• リフレッシュ設定

No.	リンク側					CPU側				
	デバイス名	点数	先頭	最終		リフレッシュ先	デバイス名	点数	先頭	最終
-	SB									
-	SW									
1	RX	192	00000	000BF		指定デバイス	X	192	01000	010BF
2	RY	192	00000	000BF		指定デバイス	Y	192	01000	010BF
3	RWr	48	00000	0002F		指定デバイス	W	48	00000	0002F
4	RWw	48	00000	0002F		指定デバイス	W	48	01000	0102F

操作手順

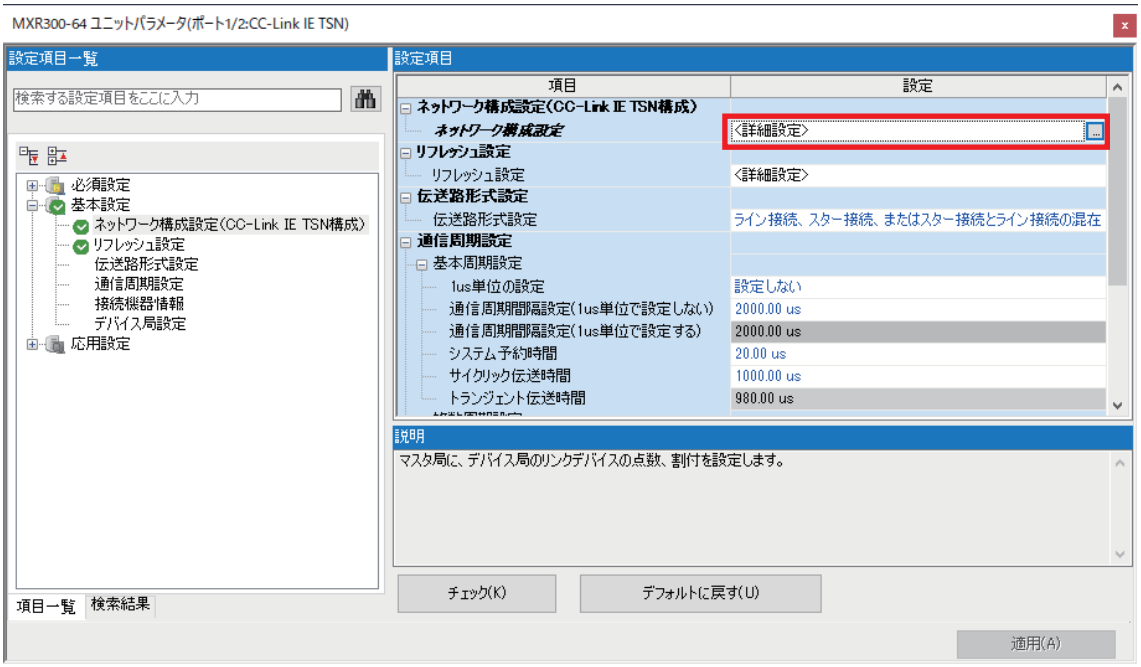
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック



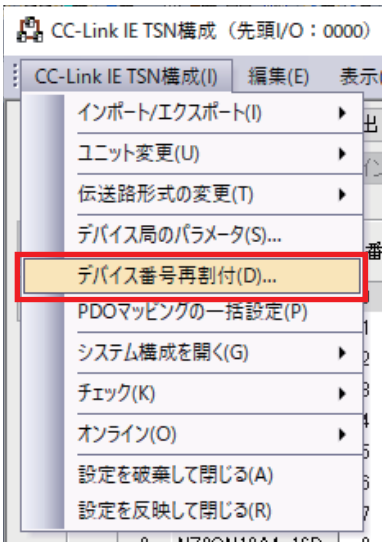
2. 基本設定から，ネットワーク構成設定を開きます。

[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



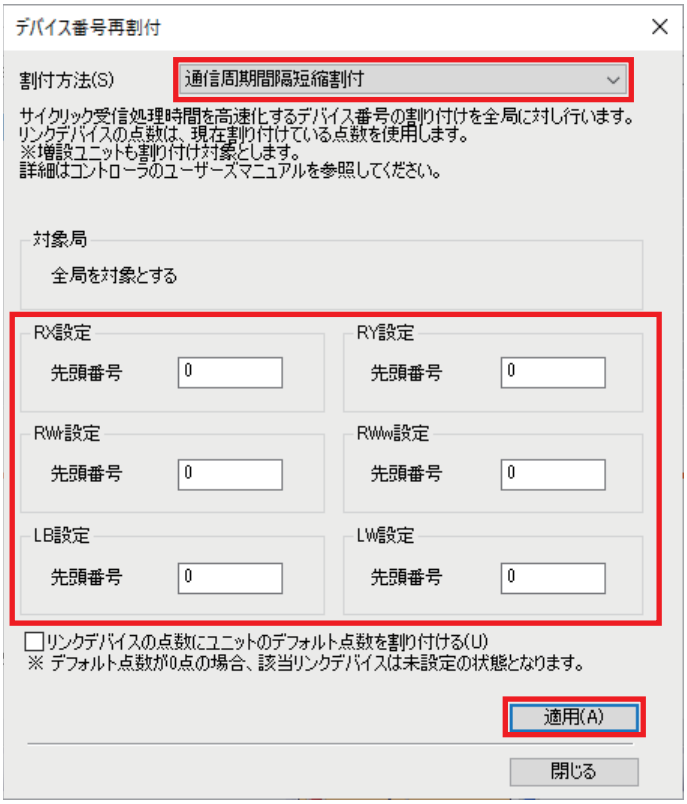
3. デバイス番号の再割付を行います。

メニューの[CC-Link IE TSN構成]⇨[デバイス番号再割付]をクリック



4. 割付方法に"通信周期間隔短縮割付"を選択し，各リンクデバイスの先頭番号を設定して[適用]ボタンをクリックします。

・各リンクデバイスの先頭番号: 0

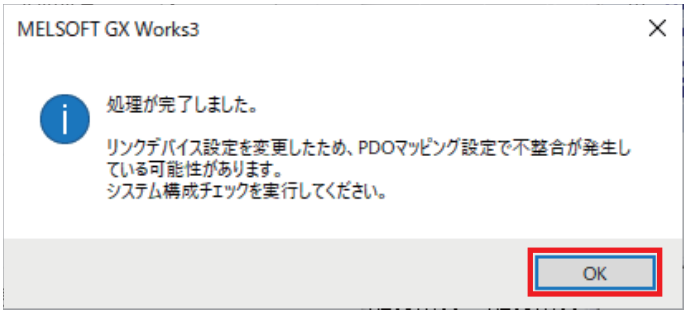


注意事項

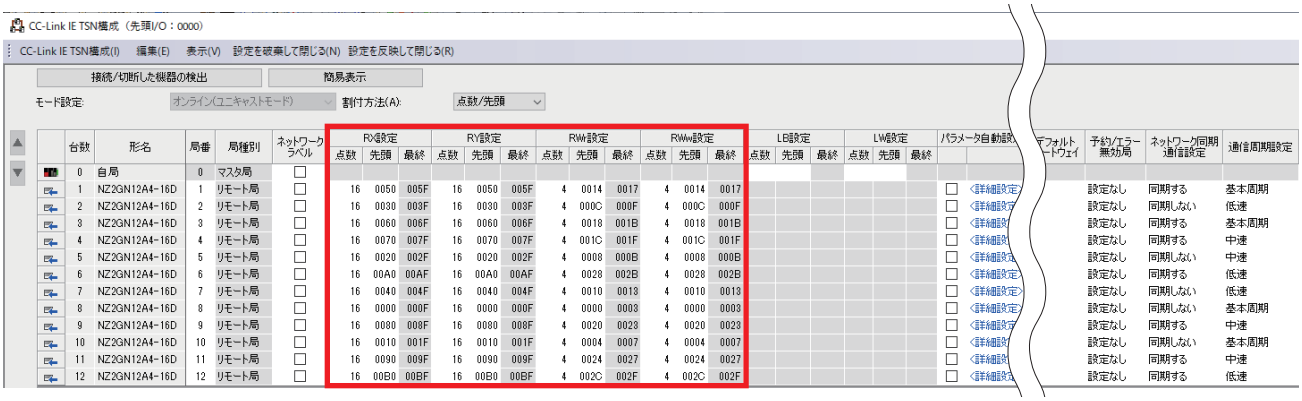
すでに各リンクデバイスに番号が割りついている場合，番号順が変更されます。デバイスの割り付け順については，下記を参照してください。

☞ 96ページ 通信周期間隔短縮割付

5. [OK]ボタンをクリックします。



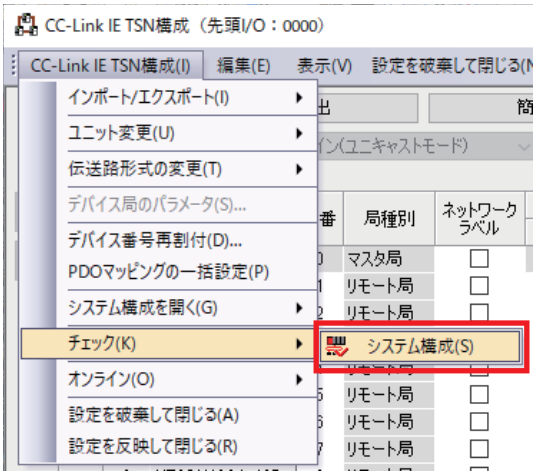
デバイス番号が再割付されます。



6. システム構成チェックを行います。

エラーや警告が発生した場合は，構成や設定を見直してください。

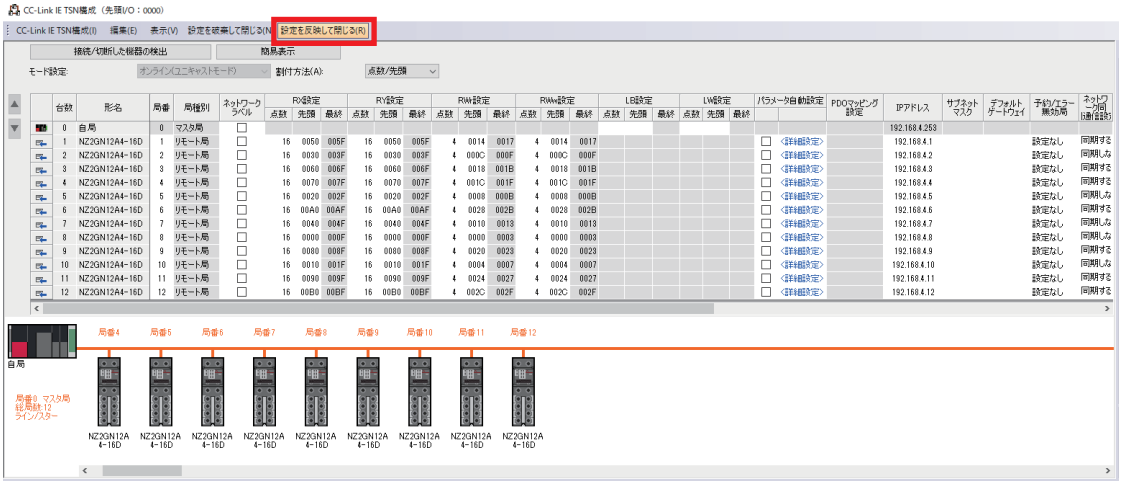
メニューの[CC-Link IE TSN構成]⇨[チェック]⇨[システム構成]をクリック



Point

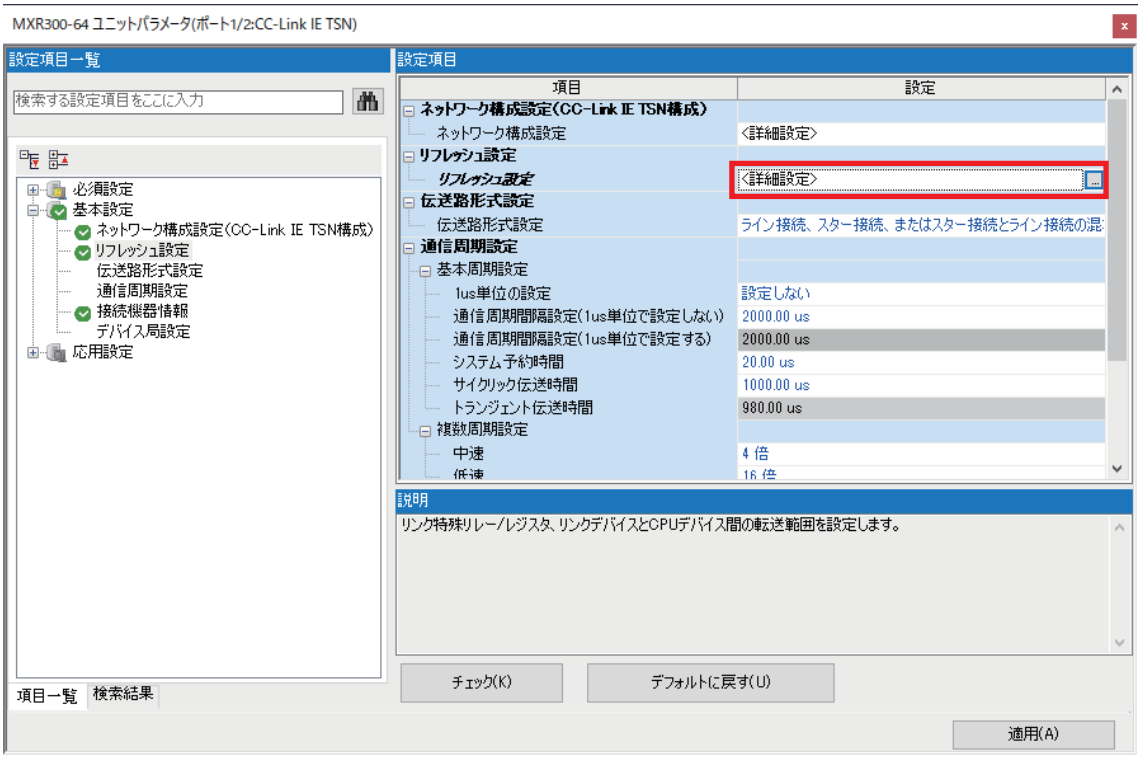
エラーや警告がある場合は，"出力"ドッキングウィンドウに内容が表示されます。

7. [設定を反映して閉じる]ボタンをクリックします。



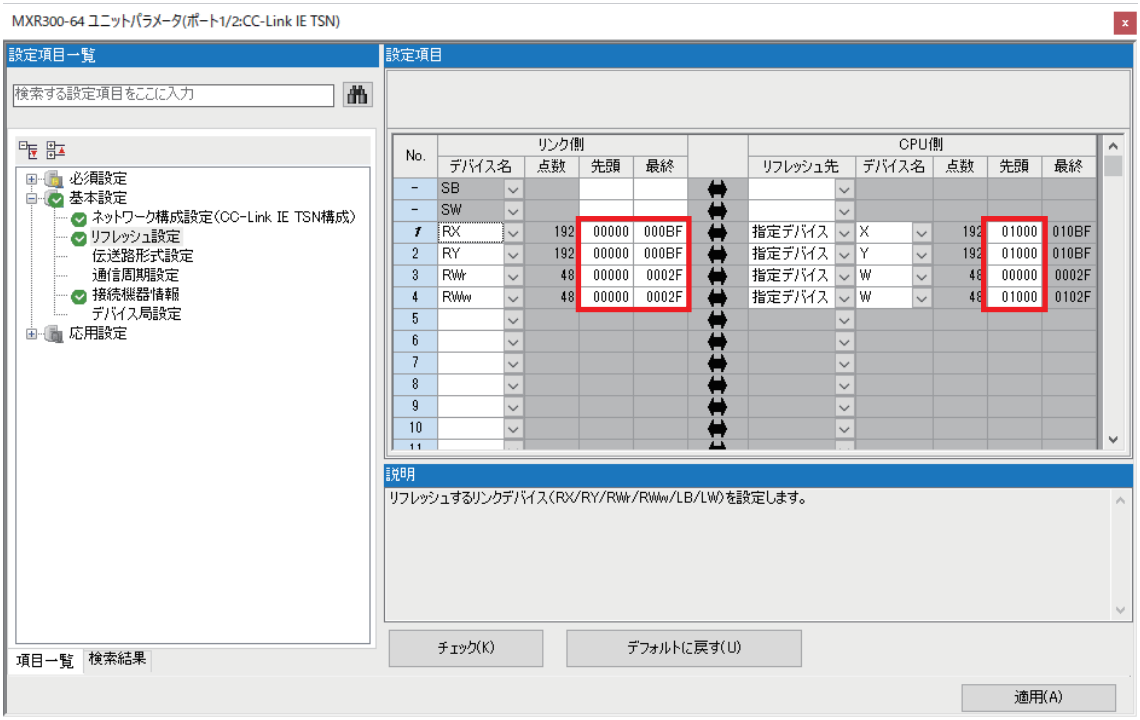
8. リンクリフレッシュ設定に合わせて，リフレッシュを設定します。

[基本設定]⇨[リフレッシュ設定]⇨[リフレッシュ設定]の<詳細設定>をダブルクリック

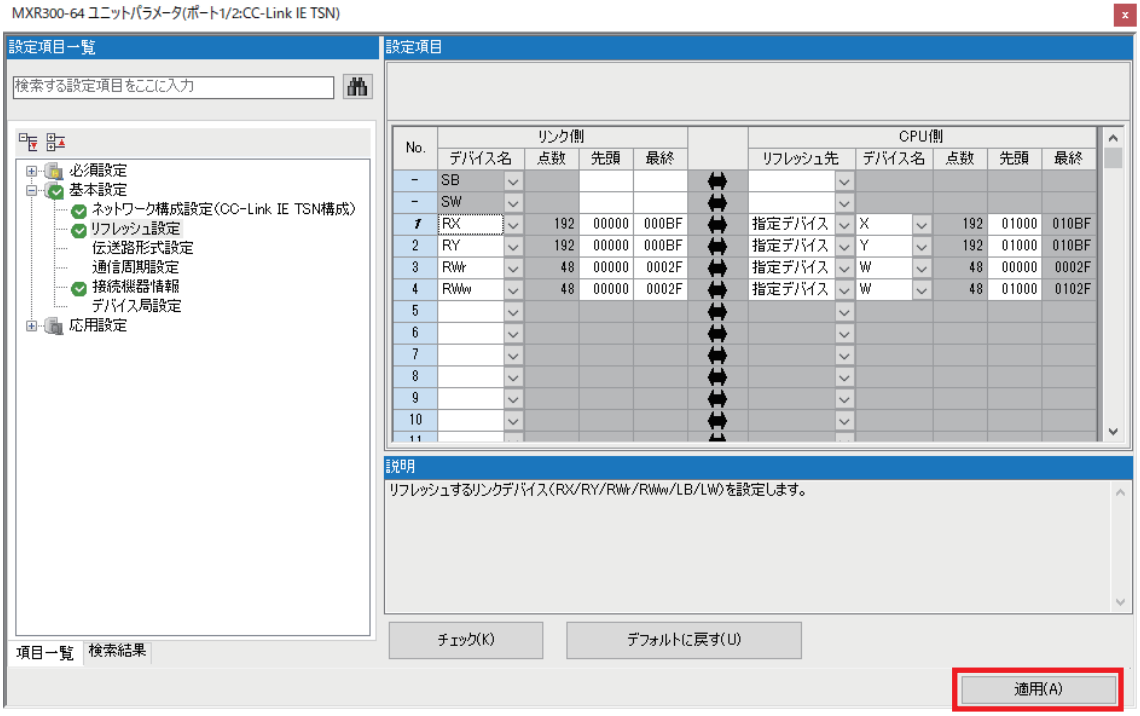


9. リンクデバイスの再割付により，リンク側のデバイスの先頭番号と最終番号に変更がないか確認し，必要に応じて変更します。

リンク側の設定を変更した場合は，CPU側の設定も見直してください。



10. [適用]ボタンをクリックして設定を適用します。



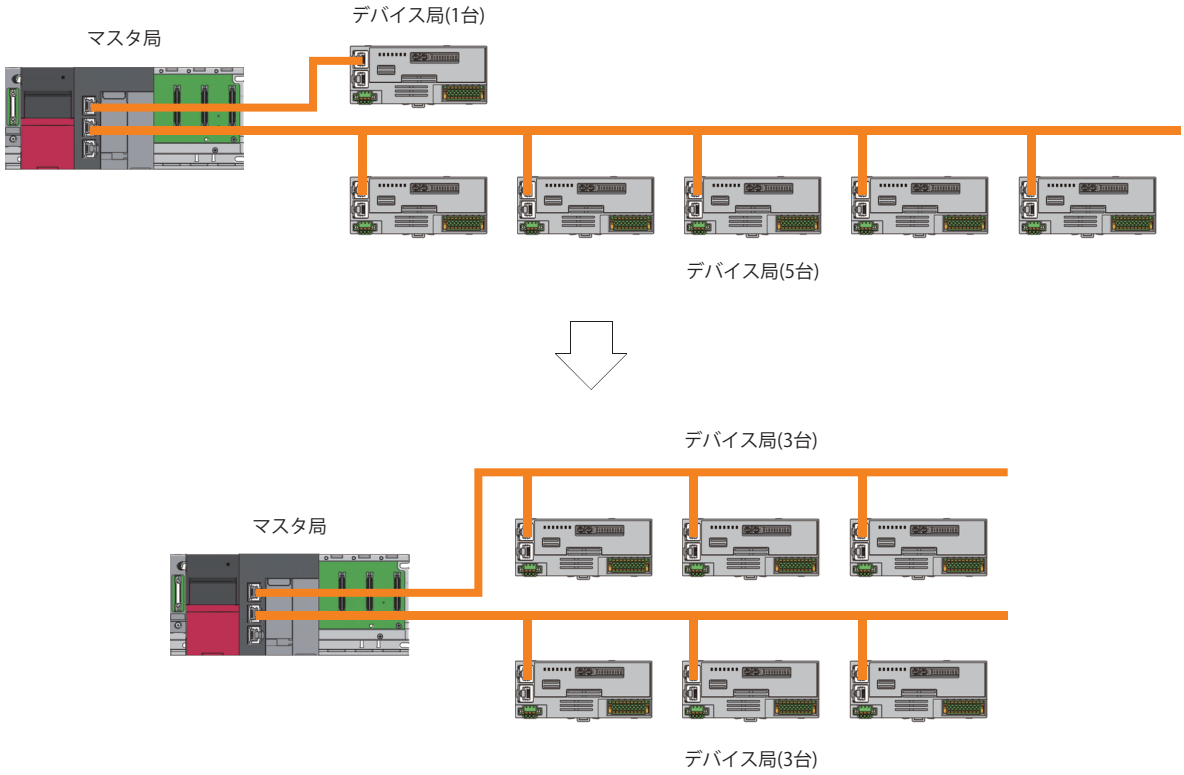
7.2 デバイス局の台数の均等化

概要

マスタ局の各ポートに接続しているデバイス局の台数を均等にすることで，通信周期間隔およびサイクリック伝送時間を短縮できます。

実行手順

下記のように各ポートで接続しているデバイス局の台数が異なる場合，均等になるよう接続を直してください。



注意事項

デバイス局の台数を均等化することで，デバイス局の配置や実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。

7.3 送信上限サイズの設定

概要

マスタ局に接続しているデバイス局でCC-Link IE TSN Classが「Class A」かつ，通信周期設定が「低速」の場合，CC-Link IE TSN Class Aグループの送信上限サイズを小さくすることで，通信周期間隔およびサイクリック伝送時間を短縮できます。

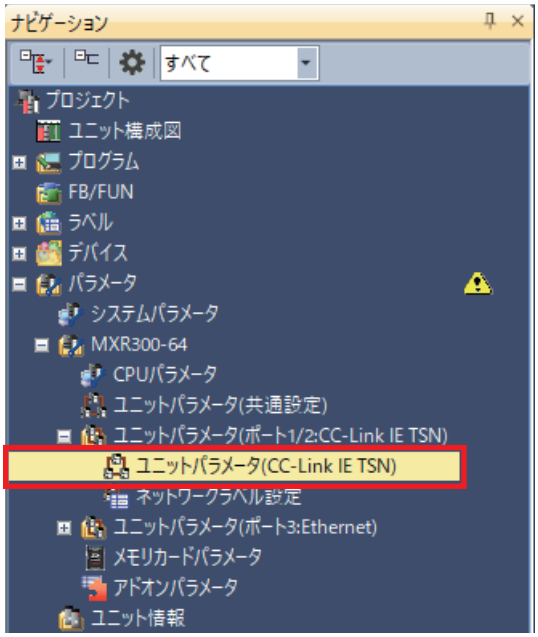
実行手順

マスタ局にCC-Link IE TSN Classが「Class A」かつ，通信周期が「低速」のデバイス局を接続している場合を例に，CC-Link IE TSN Class Aグループの送信上限サイズを設定する手順を示します。

操作手順

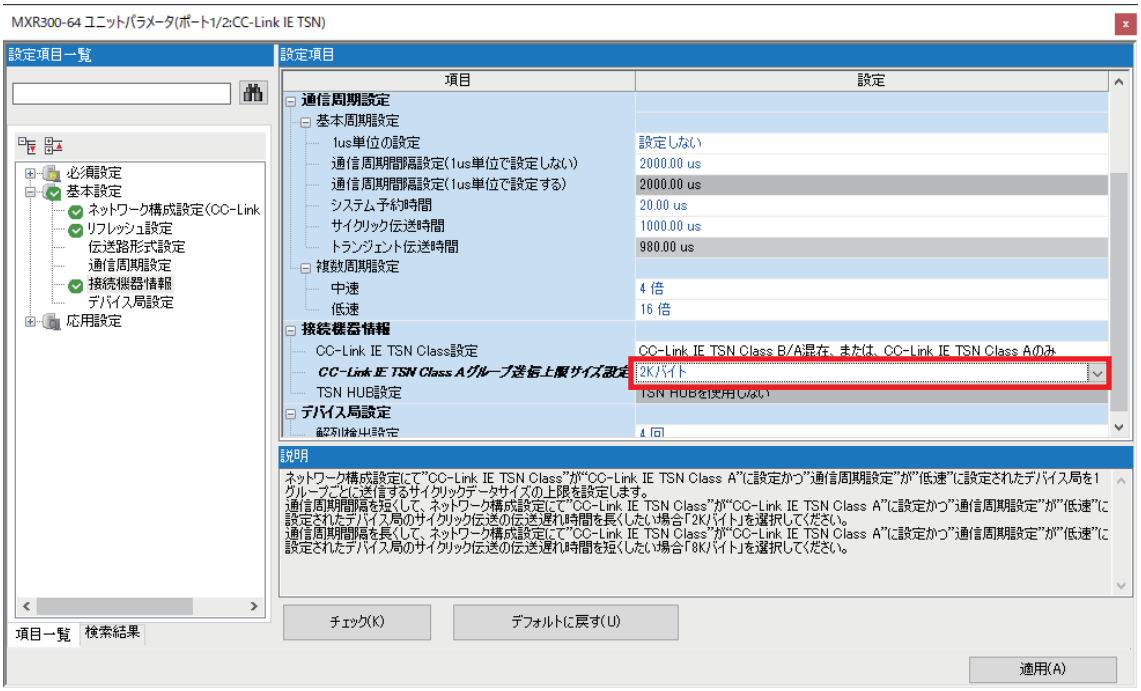
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック

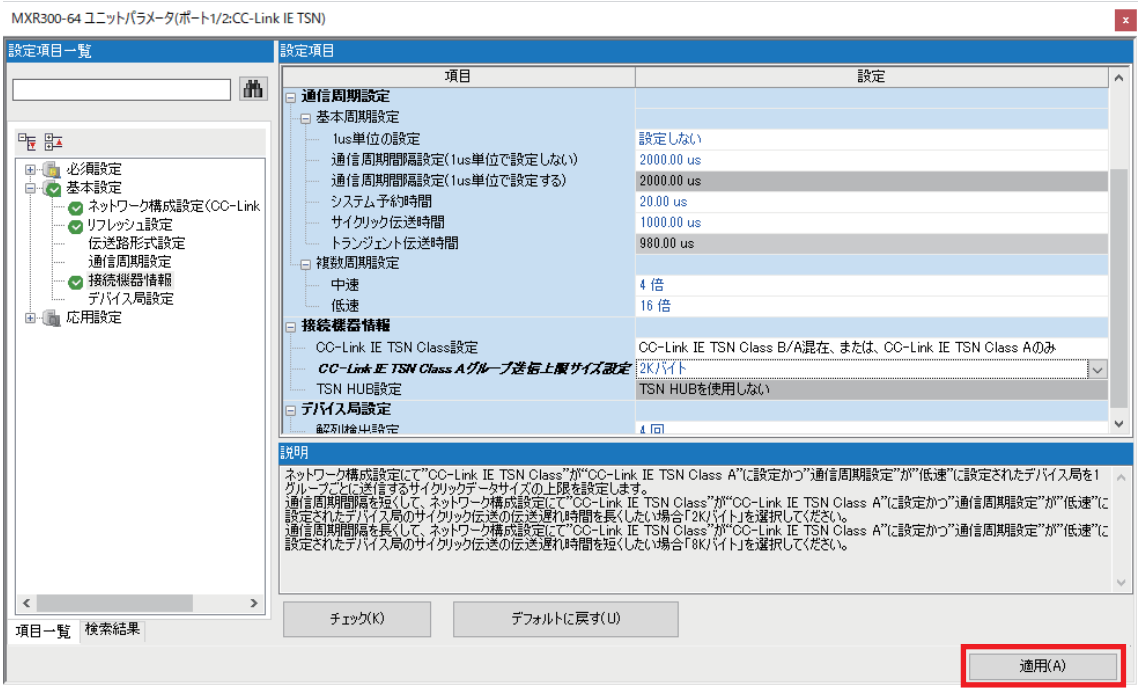


2. CC-Link IE TSN Class Aグループの送信上限サイズを"2Kバイト"(デフォルト)に設定します。

[基本設定]⇨[接続機器情報]⇨[CC-Link IE TSN Class Aグループの送信上限サイズ]



3. [適用]ボタンをクリックして設定を適用します。



注意事項

- 送信上限サイズを設定することで，実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。
- "CC-Link IE TSN Class Aグループ送信上限サイズ設定"を"2Kバイト"に設定した場合，1回の通信周期で1つのグループと通信を行うため，CC-Link IE TSN Classが「Class A」かつ通信周期が「低速」のデバイス局の伝送遅れ時間が長くなります。詳細については，下記のマニュアルを参照してください。

📖MELSEC MXコントローラ(MX-Rモデル) ユーザーズマニュアル

7.4 デバイス局の台数の削減

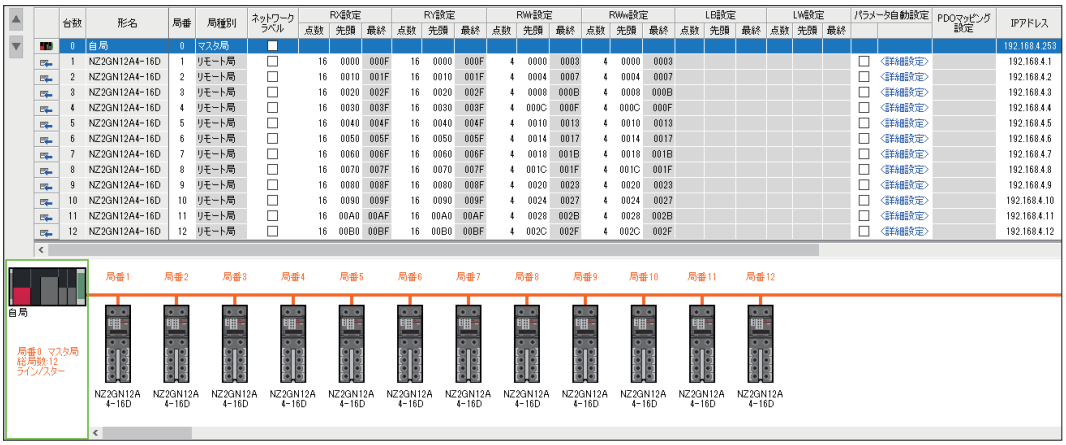
概要

マスタ局に接続しているデバイス局の台数を減らすことで，通信周期間隔およびサイクリック伝送時間を短縮できます。

実行手順

下記のネットワーク構成とリフレッシュ設定を例に，デバイス局の台数を削除する手順を示します。

・ネットワーク構成



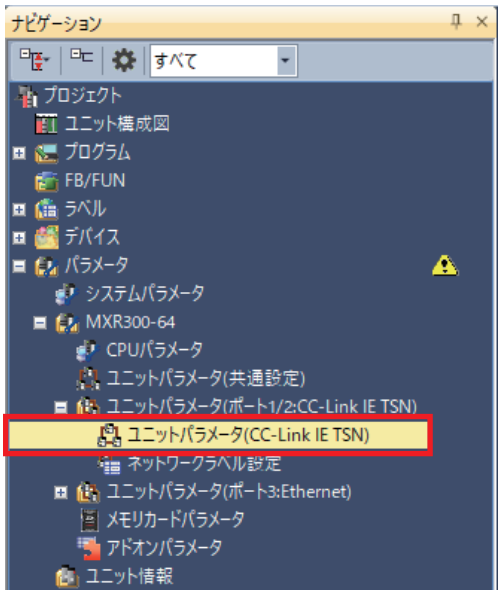
・リフレッシュ設定

No.	リンク側 デバイス名	点数	先頭	最終		CPU側 リフレッシュ先	デバイス名	点数	先頭	最終
-	SB									
-	SW									
1	RX	192	00000	000BF		指定デバイス	X	192	01000	010BF
2	RY	192	00000	000BF		指定デバイス	Y	192	01000	010BF
3	RWr	48	00000	0002F		指定デバイス	W	48	00000	0002F
4	RWw	48	00000	0002F		指定デバイス	W	48	01000	0102F

操作手順

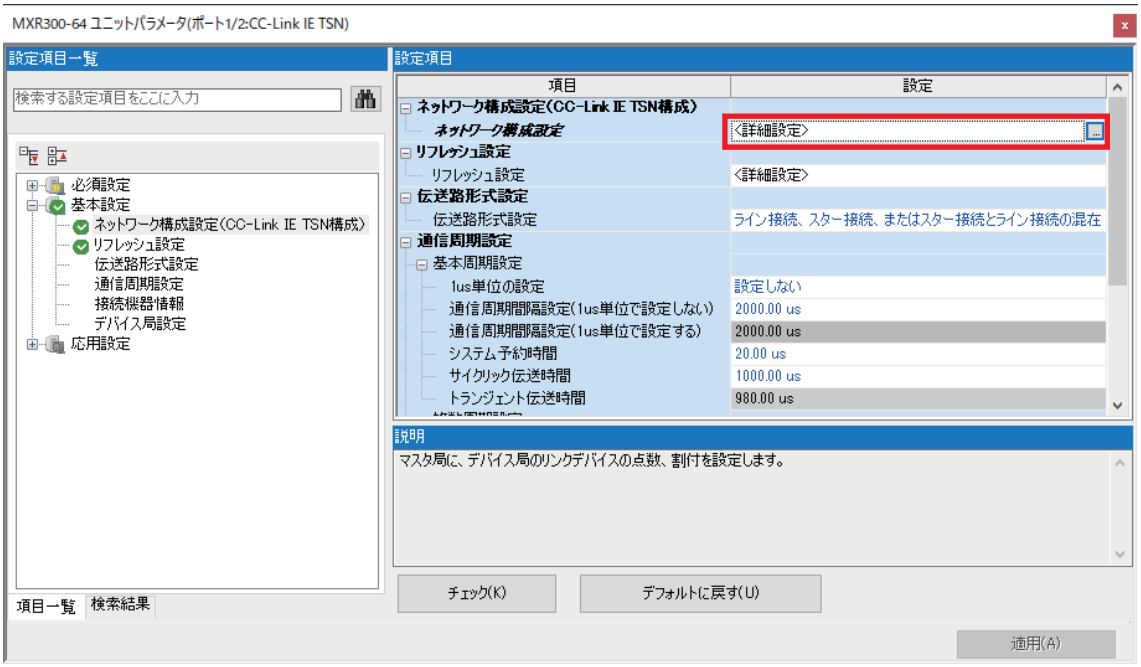
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック



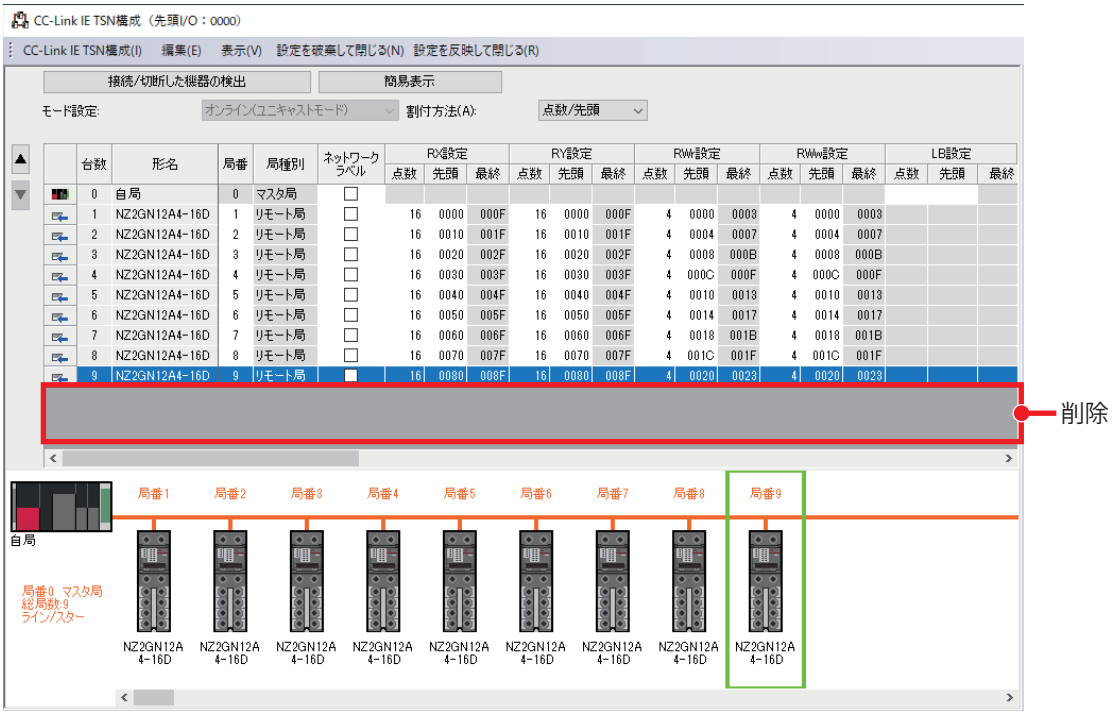
2. 基本設定から，ネットワーク構成設定を開きます。

[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



3. 使用しない不要なデバイス局を削除します。

・局番10～12のデバイス局: 削除



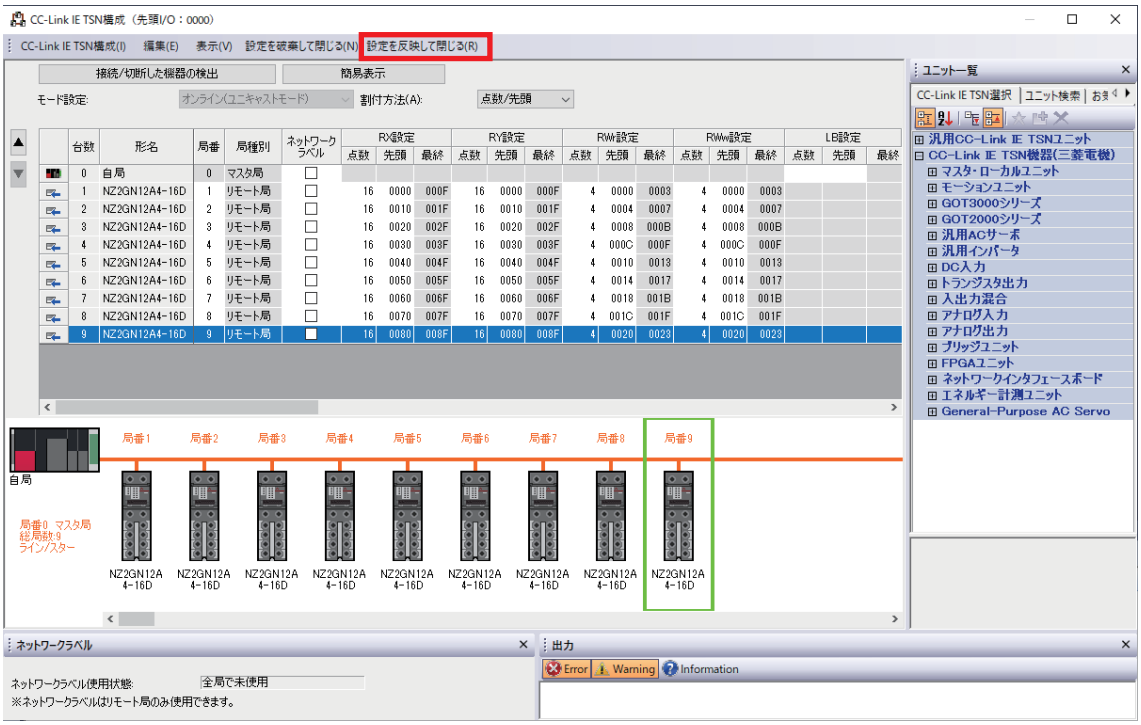
Point

不要なデバイス局の削除後，リンクデバイス番号を再割付することで，リンクリフレッシュ時間や通信周期間隔，サイクリック伝送時間をより短縮できる場合があります。

詳細については，下記を参照してください。

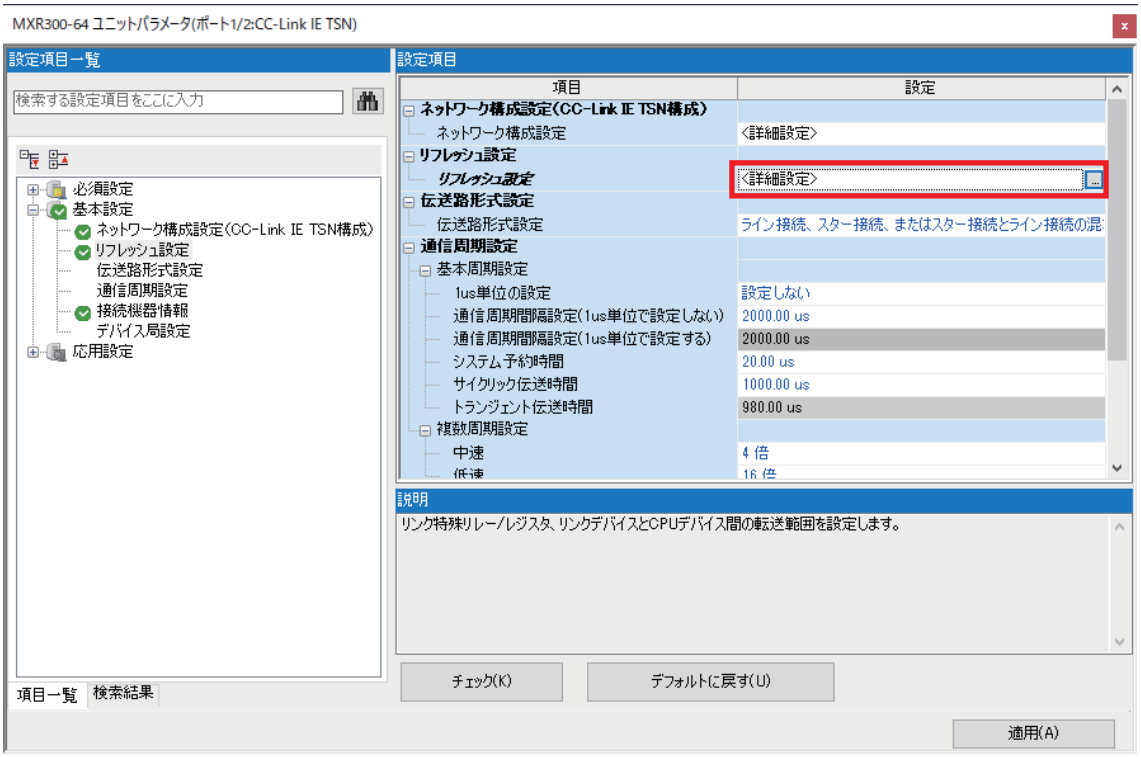
☞ 96ページ リンクリフレッシュ時間の短縮

4. [設定を反映して閉じる]ボタンをクリックします。



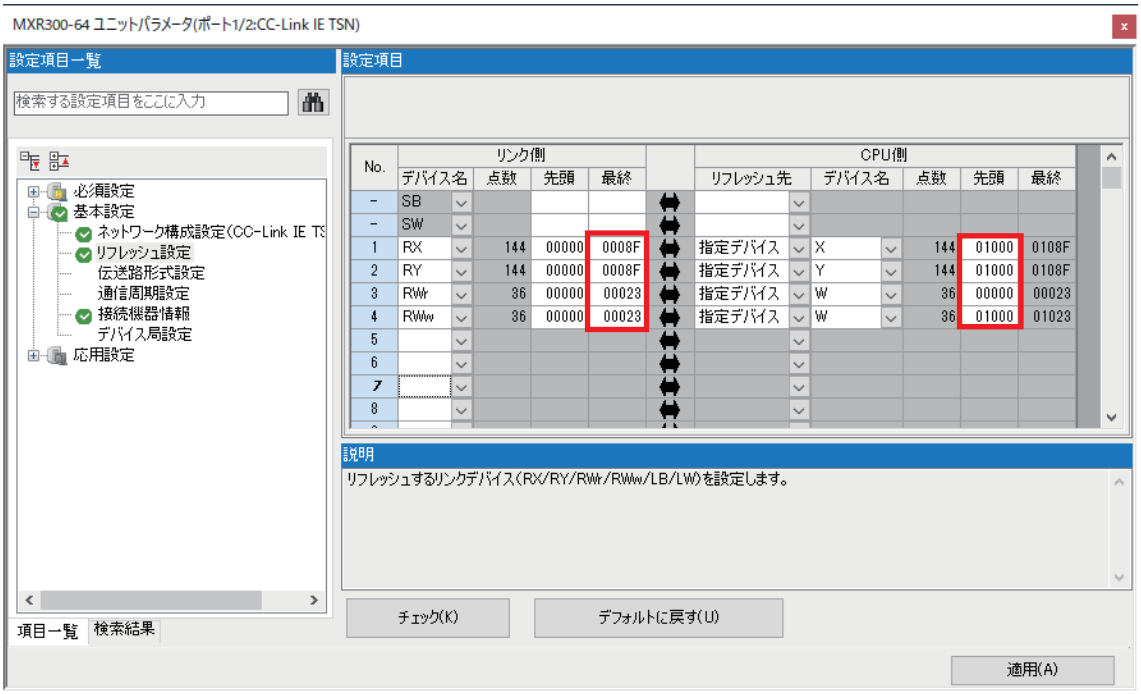
5. リンクリフレッシュ設定に合わせて，リフレッシュを設定します。

[基本設定]⇨[リフレッシュ設定]⇨[リフレッシュ設定]の<詳細設定>をダブルクリック

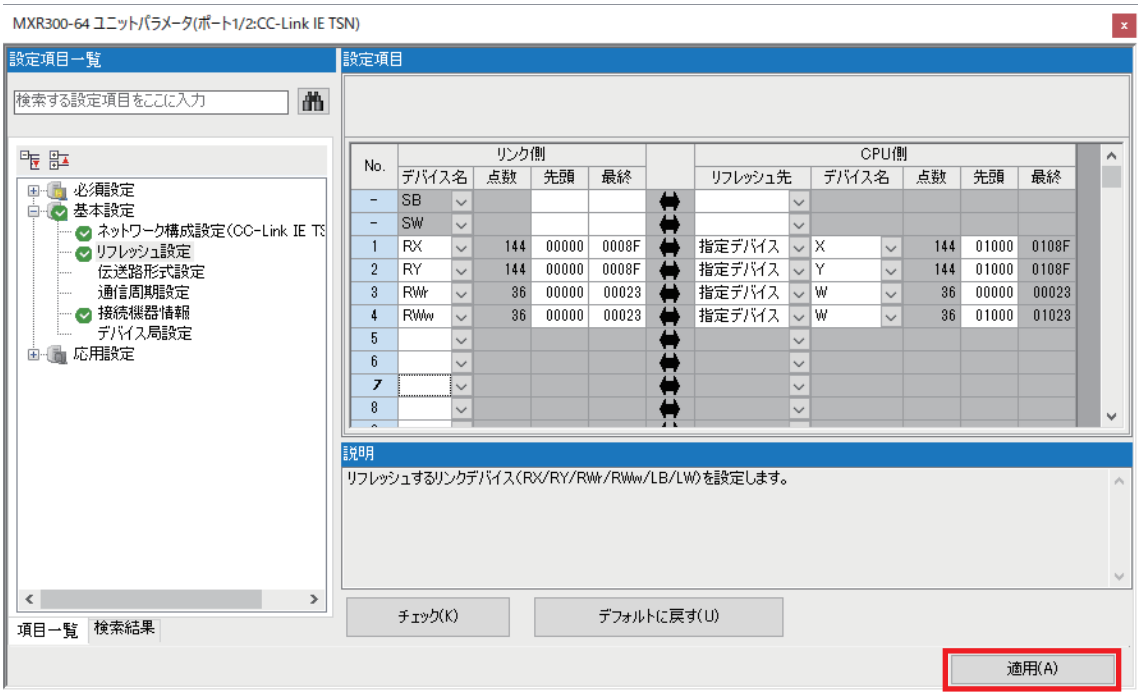


6. 手順3の不要なデバイス局の削除に合わせてリンク側のデバイスの最終番号を変更します。
またリンク側の設定の変更に応じて，CPU側の設定も見直してください。

- RX設定，RY設定の最終番号: 0008F
- RWr設定，RWw設定の最終番号: 00023



7. [適用]ボタンをクリックして設定を適用します。



注意事項

- デバイス局の台数を削除することで接続しているデバイス局の台数が減少し，実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。
- CC-Link IE TSN Classが「CC-Link IE TSN Class A」かつ通信周期設定が「低速」に設定されているデバイス局については，削除してもCC-Link IE TSNの高速化に効果はありません。各デバイス局の設定については，「ネットワーク構成設定」画面から確認してください。

7.5 リンクデバイス設定の削除

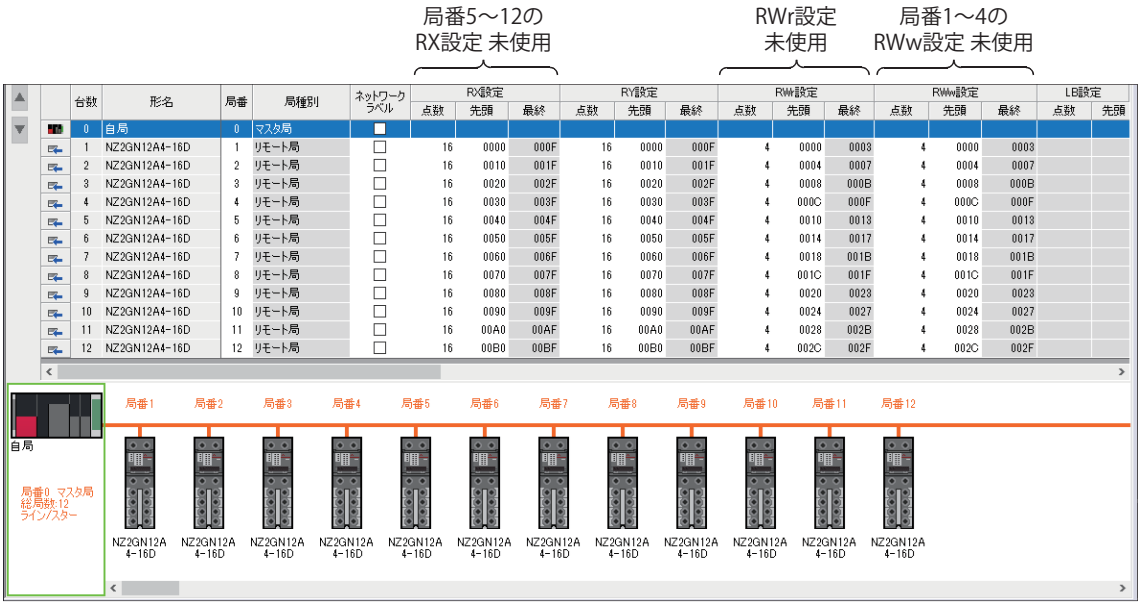
概要

各デバイス局の不要なリンクデバイス設定を削除することで，通信周期間隔およびサイクリック伝送時間を短縮できます。

実行手順

下記のネットワーク構成とリフレッシュ設定を例に，不要なリンクデバイス設定を削除する手順を示します。

・ネットワーク構成



台数	形名	局番	局種別	ネットワークラベル	RX設定 点数	RX設定 先頭	RX設定 最終	RY設定 点数	RY設定 先頭	RY設定 最終	RWr設定 点数	RWr設定 先頭	RWr設定 最終	RWw設定 点数	RWw設定 先頭	RWw設定 最終	LB設定 点数	LB設定 先頭
0	自局	0	マスタ局															
1	NZ2GN12A4-16D	1	リモート局		16	0000	000F	16	0000	000F	4	0000	0003	4	0000	0003		
2	NZ2GN12A4-16D	2	リモート局		16	0010	001F	16	0010	001F	4	0004	0007	4	0004	0007		
3	NZ2GN12A4-16D	3	リモート局		16	0020	002F	16	0020	002F	4	0008	000B	4	0008	000B		
4	NZ2GN12A4-16D	4	リモート局		16	0030	003F	16	0030	003F	4	000C	000F	4	000C	000F		
5	NZ2GN12A4-16D	5	リモート局		16	0040	004F	16	0040	004F	4	0010	0013	4	0010	0013		
6	NZ2GN12A4-16D	6	リモート局		16	0050	005F	16	0050	005F	4	0014	0017	4	0014	0017		
7	NZ2GN12A4-16D	7	リモート局		16	0060	006F	16	0060	006F	4	0018	001B	4	0018	001B		
8	NZ2GN12A4-16D	8	リモート局		16	0070	007F	16	0070	007F	4	001C	001F	4	001C	001F		
9	NZ2GN12A4-16D	9	リモート局		16	0080	008F	16	0080	008F	4	0020	0023	4	0020	0023		
10	NZ2GN12A4-16D	10	リモート局		16	0090	009F	16	0090	009F	4	0024	0027	4	0024	0027		
11	NZ2GN12A4-16D	11	リモート局		16	00A0	00AF	16	00A0	00AF	4	0028	002B	4	0028	002B		
12	NZ2GN12A4-16D	12	リモート局		16	00B0	00BF	16	00B0	00BF	4	002C	002F	4	002C	002F		

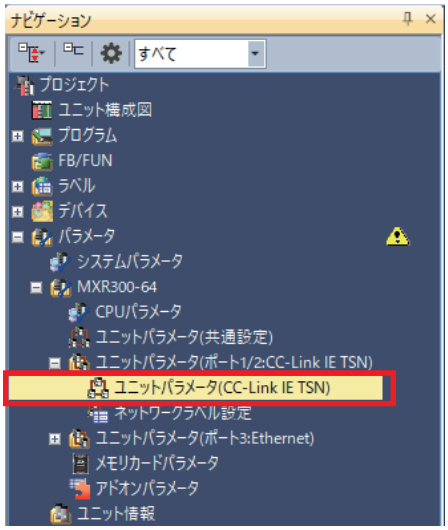
・リフレッシュ設定

No.	リンク側 デバイス名	リンク側 点数	リンク側 先頭	リンク側 最終		CPU側 リフレッシュ先	CPU側 デバイス名	CPU側 点数	CPU側 先頭	CPU側 最終
-	SB									
-	SW									
1	RX	192	00000	000BF		指定デバイス	X	192	01000	010BF
2	RY	192	00000	000BF		指定デバイス	Y	192	01000	010BF
3	RWr	48	00000	0002F		指定デバイス	W	48	00000	0002F
4	RWw	48	00000	0002F		指定デバイス	W	48	01000	0102F

操作手順

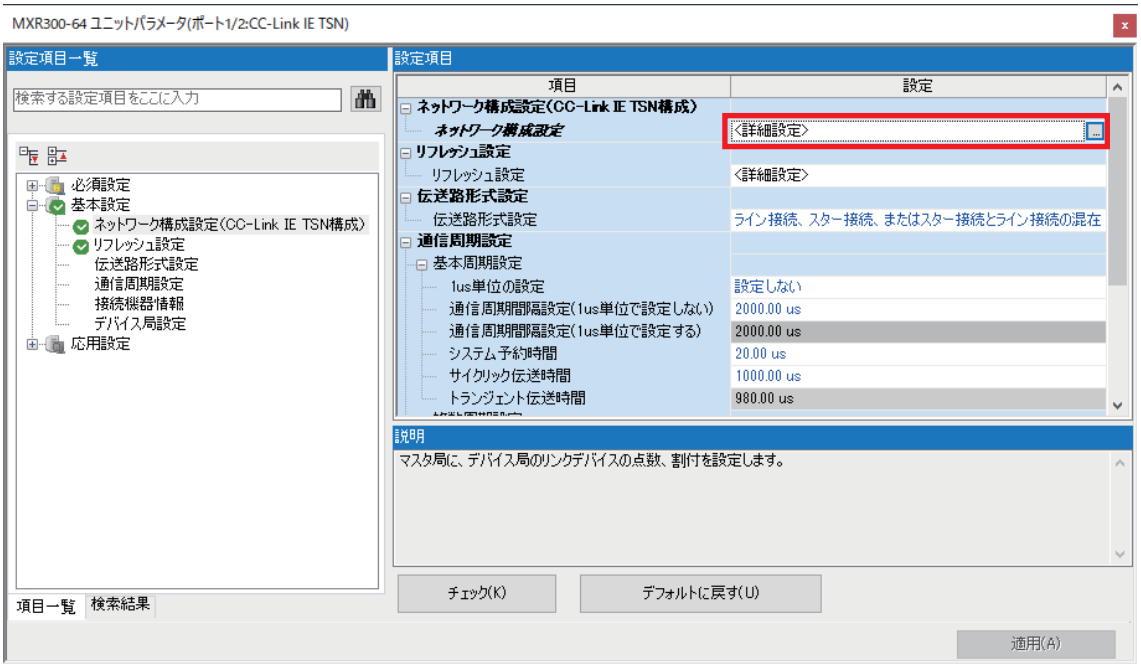
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック



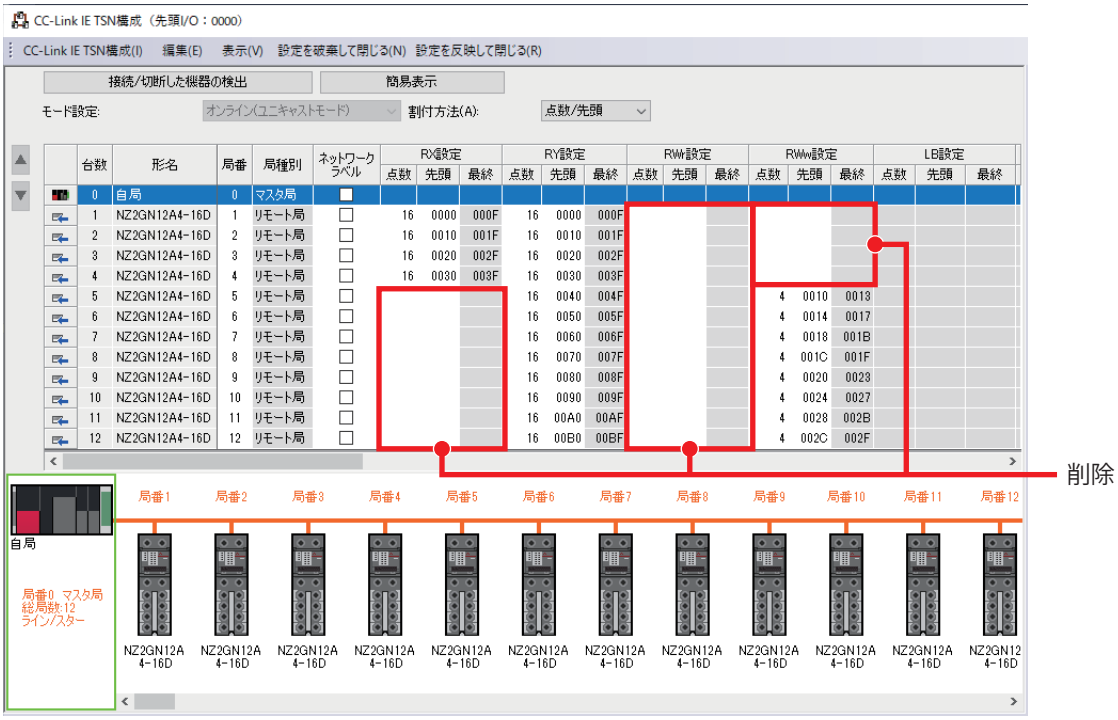
2. 基本設定から，ネットワーク構成設定を開きます。

[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



3. 各デバイス局の不要なRX設定，RY設定，RWr設定，RWw設定を削除(デバイス割付なしの状態に)します。

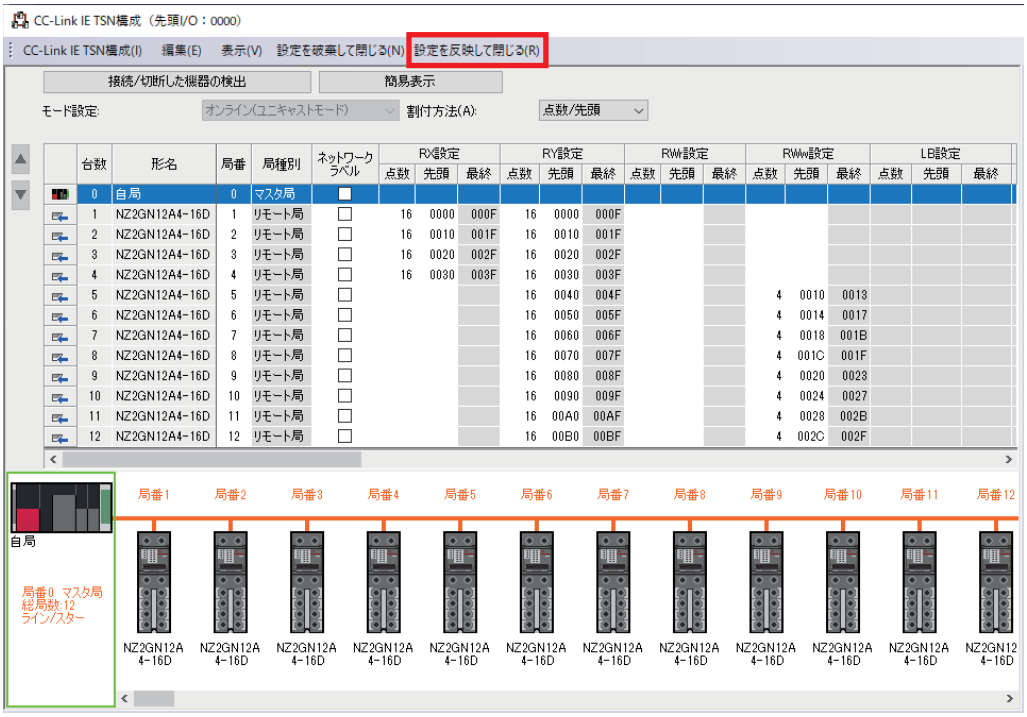
- 局番5～12のRX設定: 削除
- RWr設定の設定: 削除
- 局番1～4のRWw設定: 削除



Point

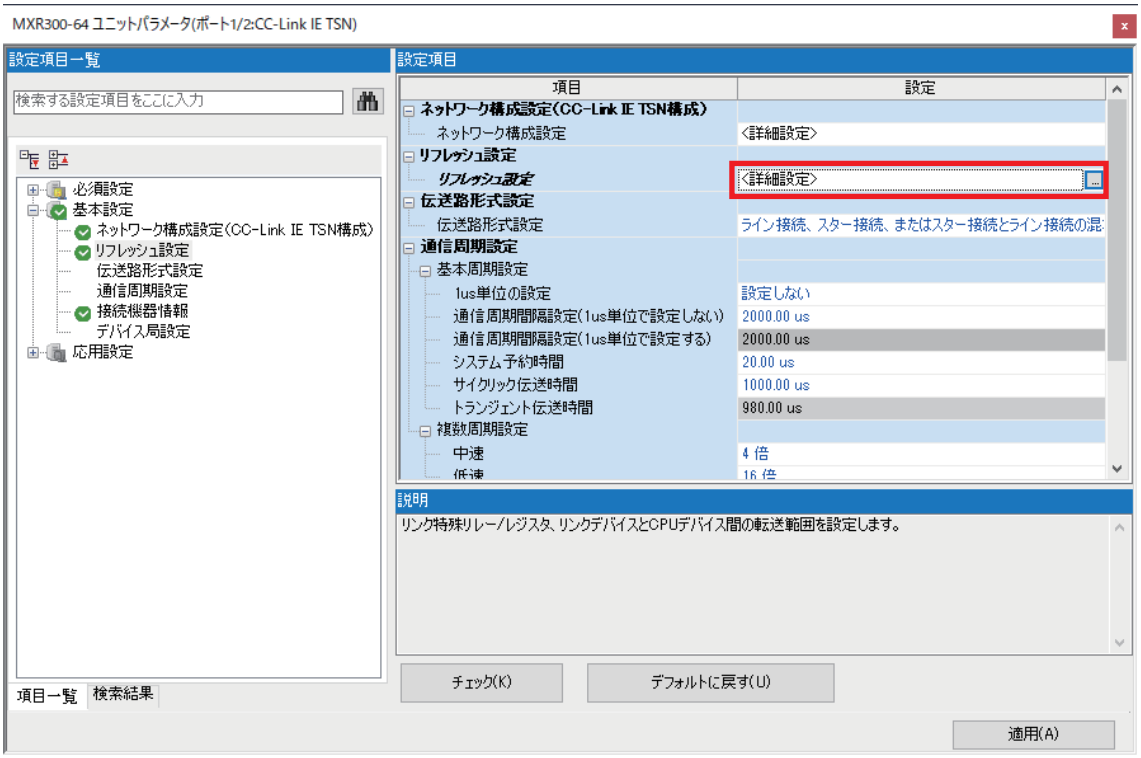
不要なリンクデバイス設定の削除後，リンクデバイス番号を再割付することで，リンクリフレッシュ時間や通信周期間隔，サイクリック伝送時間をより短縮できる場合があります。
詳細については，下記を参照してください。
☞ 96ページ リンクリフレッシュ時間の短縮

4. [設定を反映して閉じる]ボタンをクリックします。



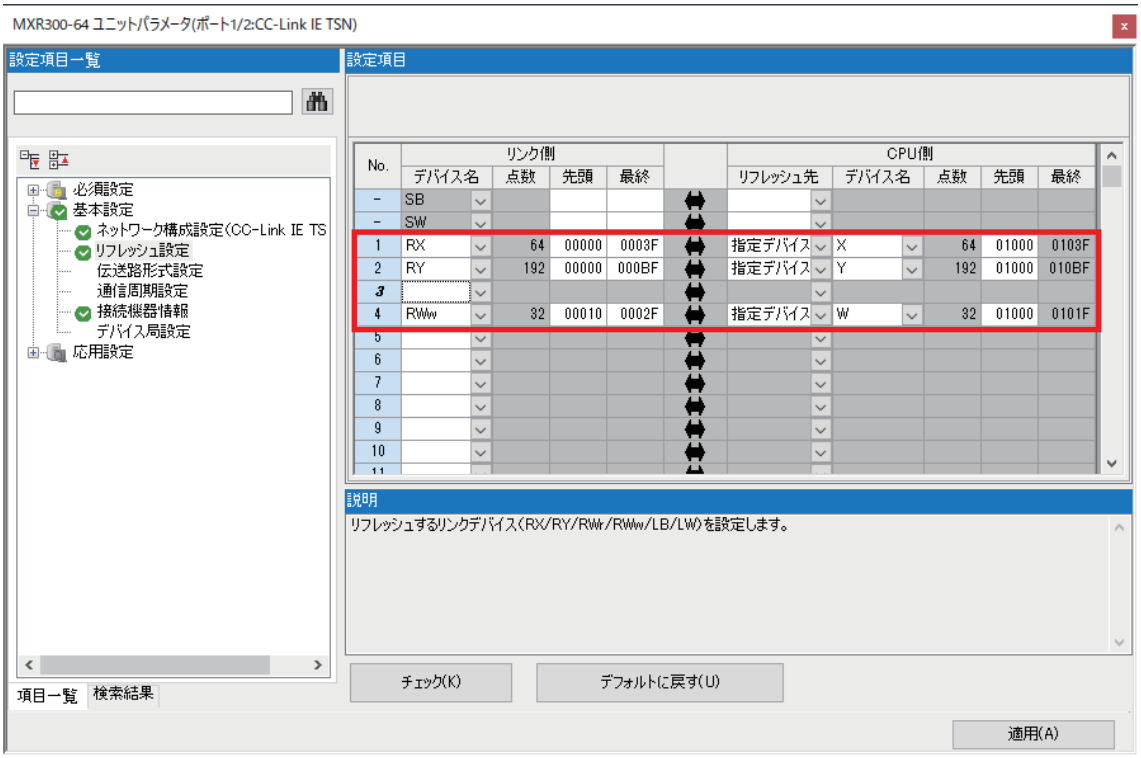
5. リンクリフレッシュ設定の変更に合わせて，リフレッシュ設定を変更します。

[基本設定]⇨[リフレッシュ設定]⇨[リフレッシュ設定]の<詳細設定>をダブルクリック

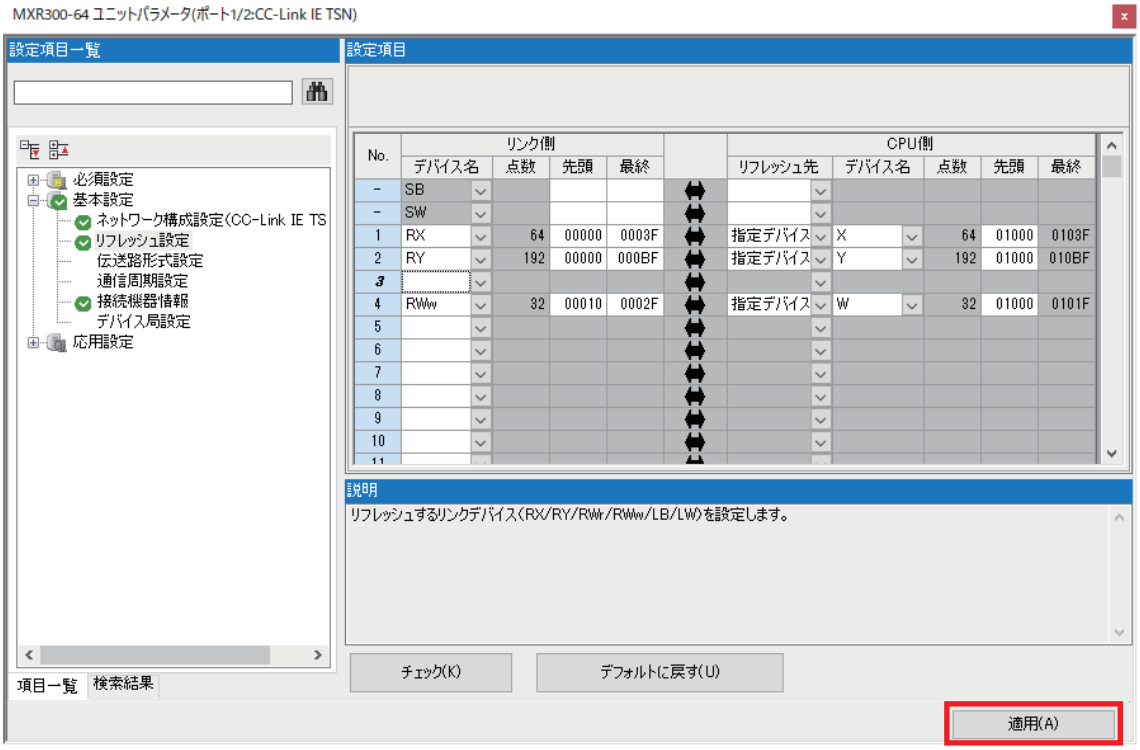


6. 手順3，4で設定したリンクデバイス番号に合わせて，リンク側のデバイスの先頭番号と最終番号を変更します。
またリンク側の設定の変更に応じて，CPU側の設定も見直してください。

- RX設定の番号範囲: 00000～0003F
- RWr設定の設定: 削除
- RWw設定の番号範囲: 00010～0002F



7. [適用]ボタンをクリックして設定を適用します。



注意事項

- リンクデバイス設定を削減することで，実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。
- CC-Link IE TSN Classが「CC-Link IE TSN Class A」かつ通信周期設定が「低速」に設定されているデバイス局については，削除してもCC-Link IE TSNの高速化に効果はありません。各デバイス局の設定については，「ネットワーク構成設定」画面から確認してください。

7.6 リンクデバイス設定の点数の削減

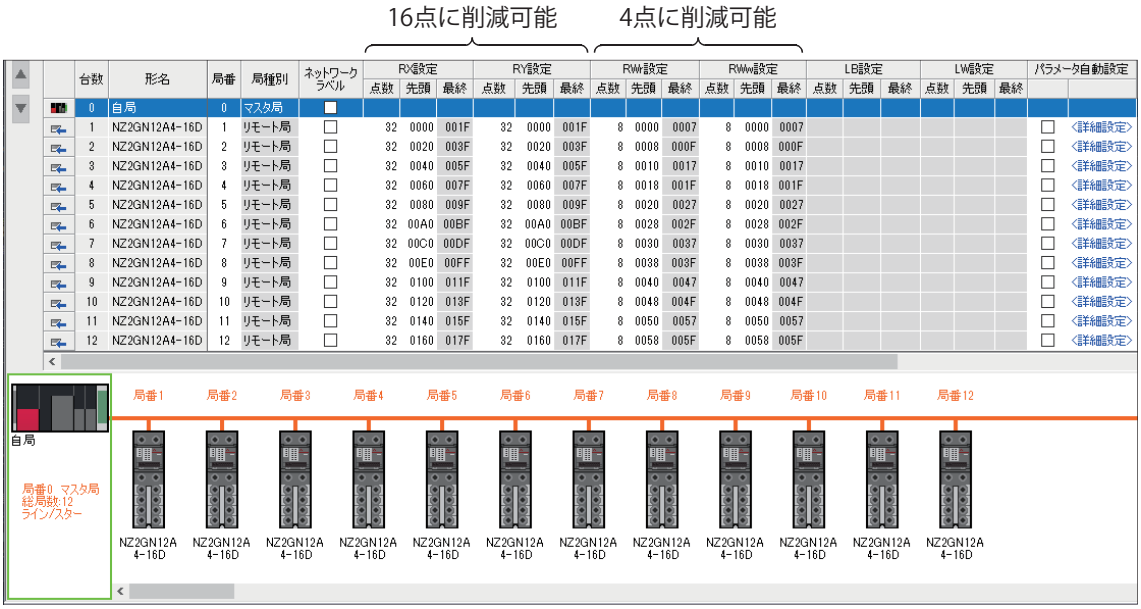
概要

各デバイス局のリンクデバイス設定の点数を大幅に削減することで，通信周期間隔およびサイクリック伝送時間を短縮できます。

実行手順

下記のネットワーク構成とリフレッシュ設定を例に，リンクデバイス設定の点数を削減する手順を示します。

・ネットワーク構成



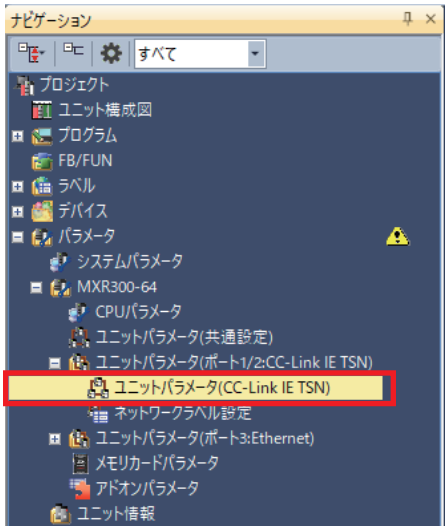
・リフレッシュ設定

No.	リンク側 デバイス名	点数	先頭	最終		CPU側 リフレッシュ先	デバイス名	点数	先頭	最終
-	SB									
-	SW									
1	RX	384	00000	0017F		指定デバイス	X	384	01000	0117F
2	RY	384	00000	0017F		指定デバイス	Y	384	01000	0117F
3	RWr	96	00000	0005F		指定デバイス	W	96	00000	0005F
4	RWw	96	00000	0005F		指定デバイス	W	96	01000	0105F

操作手順

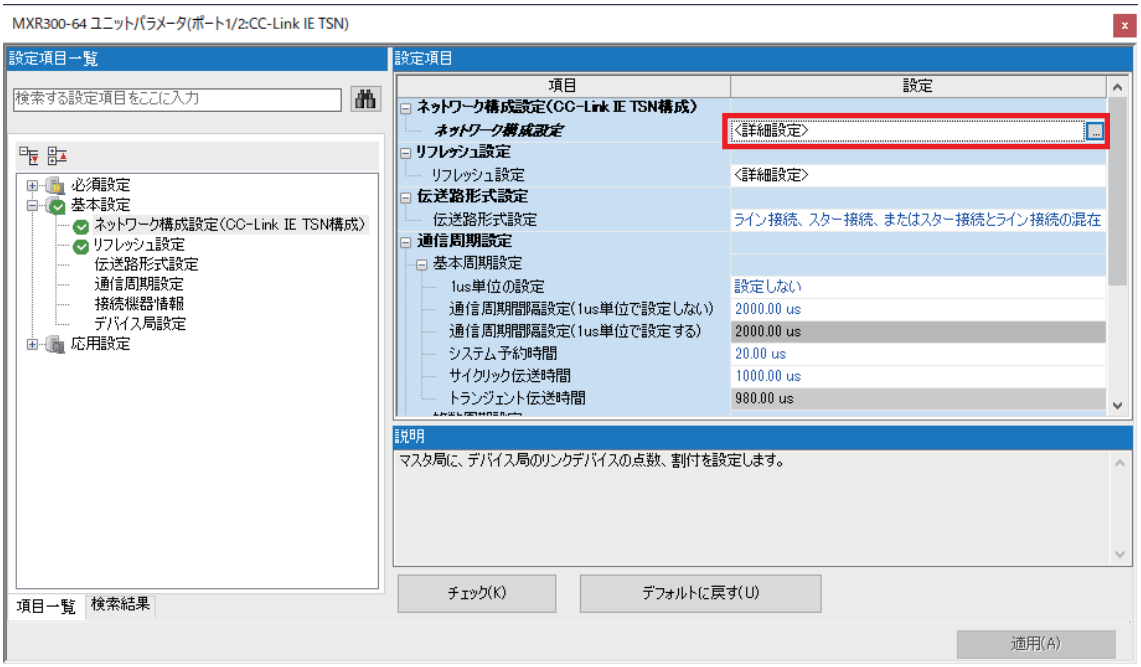
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック



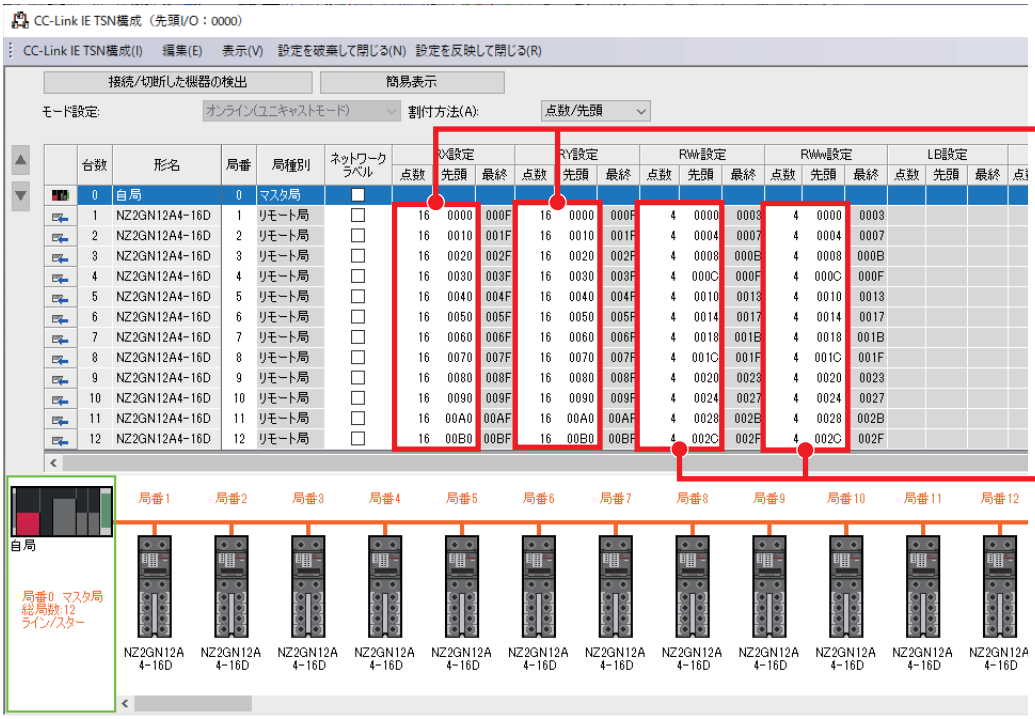
2. 基本設定から，ネットワーク構成設定を開きます。

[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



3. 各デバイス局のRX設定，RY設定，RWr設定，RWw設定の点数を削減し，削除点数に合わせて先頭アドレスを変更します。

- RX設定，RY設定の各点数: 16
- RWr設定，RWw設定の各点数: 4
- RX設定，RY設定，RWr設定，RWw設定の各先頭アドレス: 点数に合わせて変更



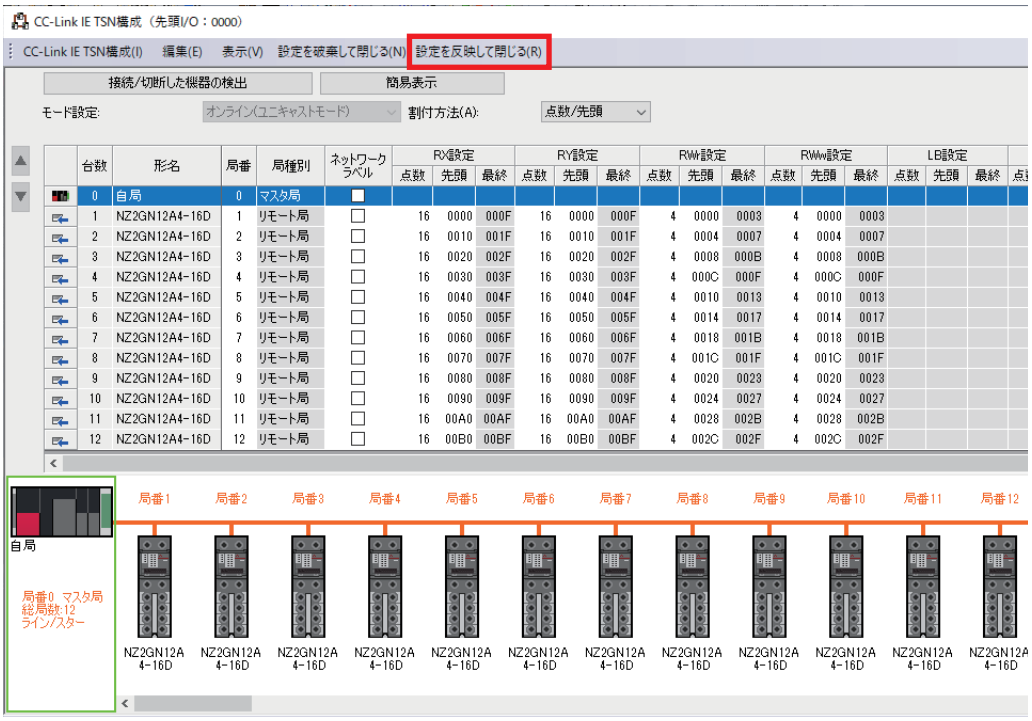
16点に変更
(先頭アドレスも修正)

4点に変更
(先頭アドレスも修正)

Point

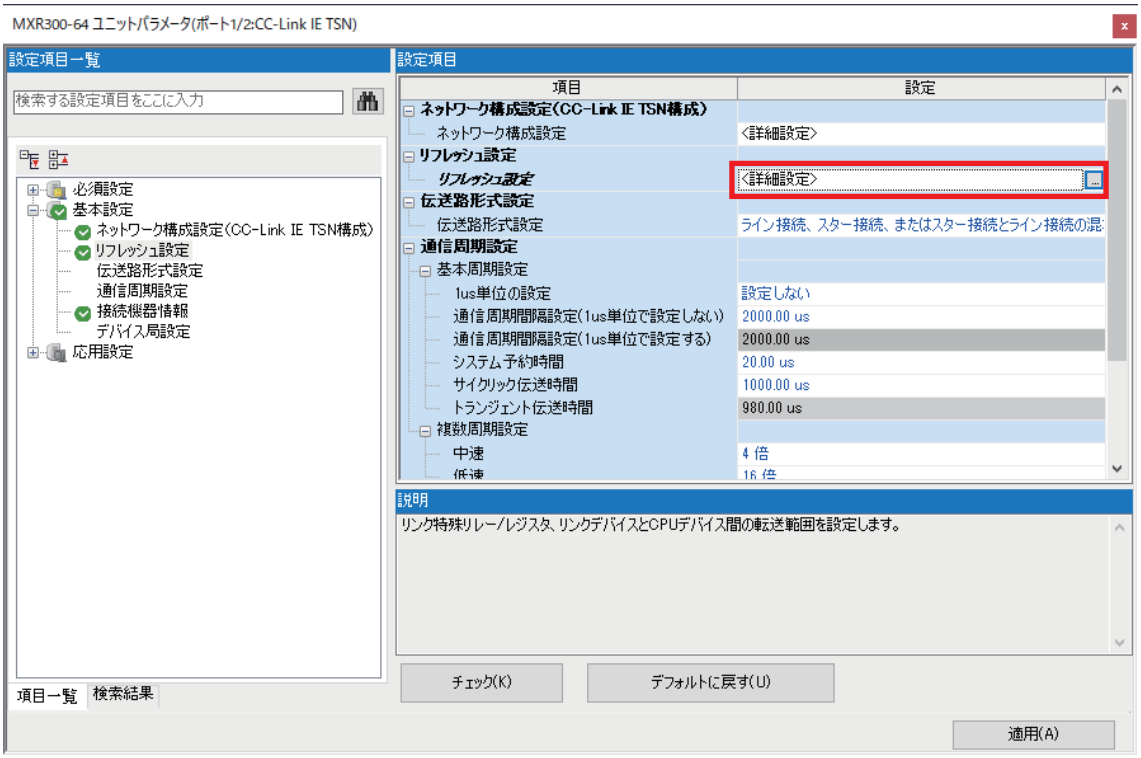
リンクデバイス設定の点数の削減後，リンクデバイス番号を再割付することで，リンクリフレッシュ時間や通信周期間隔，サイクリック伝送時間をより短縮できる場合があります。
詳細については，下記を参照してください。
☞ 96ページ リンクリフレッシュ時間の短縮

4. [設定を反映して閉じる]ボタンをクリックします。



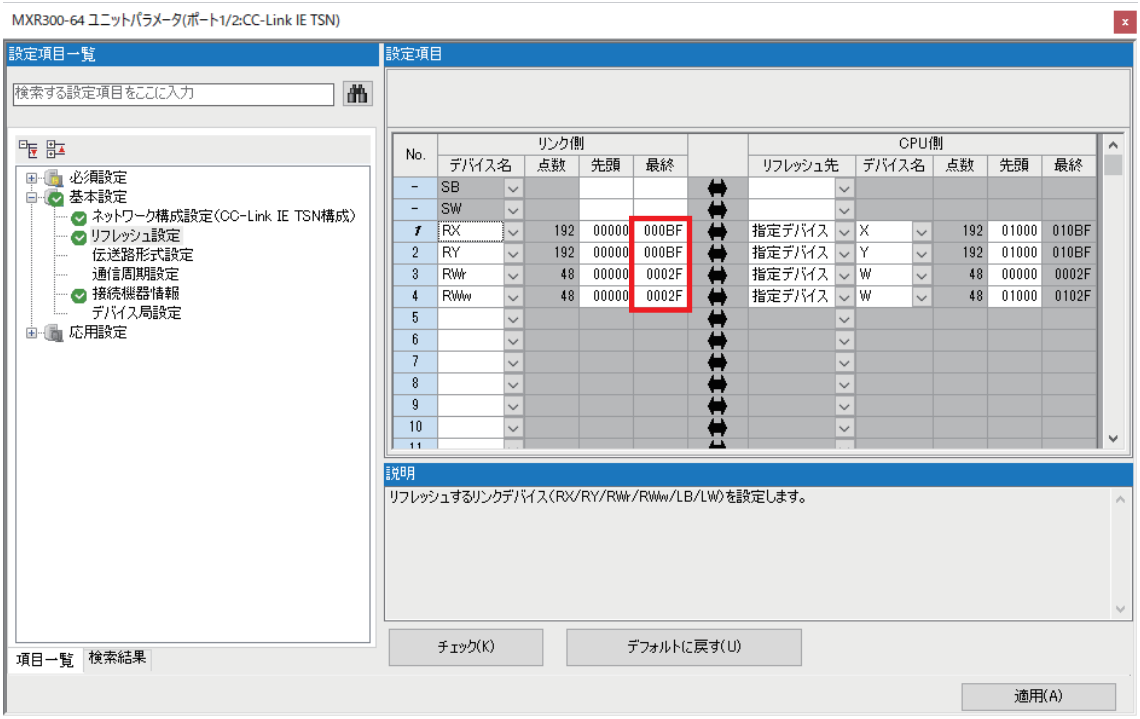
5. リンクリフレッシュ設定の変更に合わせて，リフレッシュ設定を変更します。

[基本設定]⇨[リフレッシュ設定]⇨[リフレッシュ設定]の<詳細設定>をダブルクリック

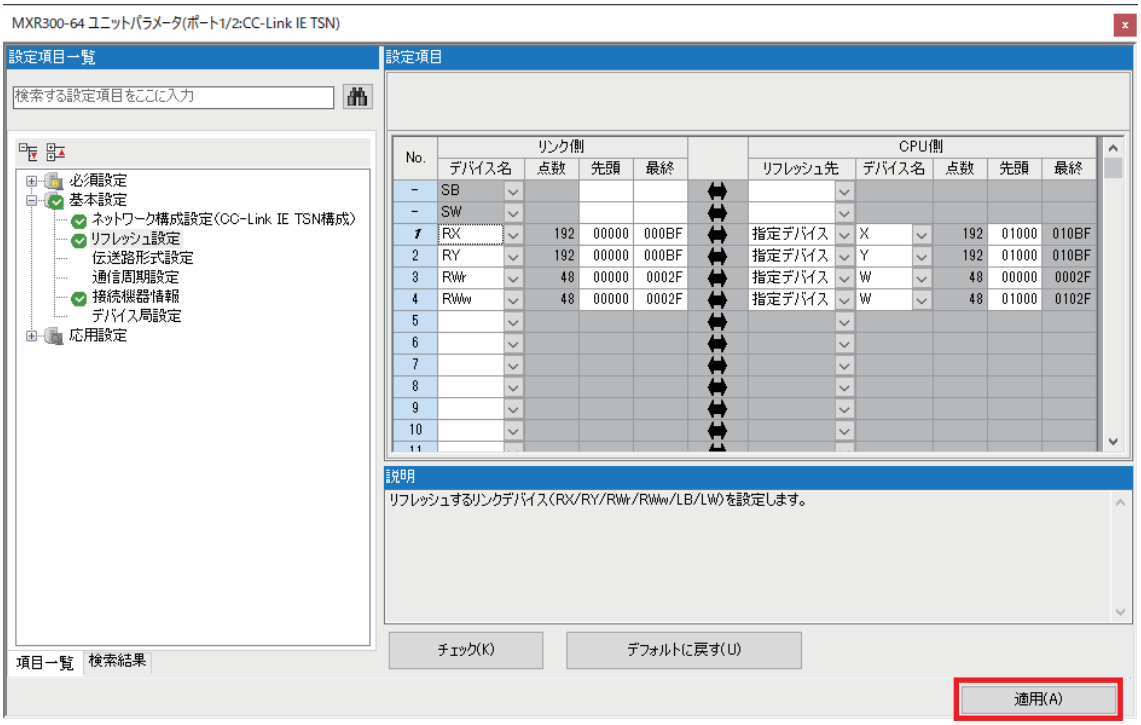


6. 手順3で設定したリンクデバイス番号に合わせて，リンク側のデバイスの最終番号を変更します。
またリンク側の設定の変更に応じて，CPU側の設定も見直してください。

- RX設定，RY設定の最終番号: 000BF
- RWr設定，RWw設定の最終番号: 0002F



7. [適用]ボタンをクリックして設定を適用します。



注意事項

- リンクデバイス設定を削減することで，実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。
- CC-Link IE TSN Classが「CC-Link IE TSN Class A」かつ通信周期設定が「低速」に設定されているデバイス局については，削除してもCC-Link IE TSNの高速化に効果はありません。各デバイス局の設定については，「ネットワーク構成設定」画面から確認してください。

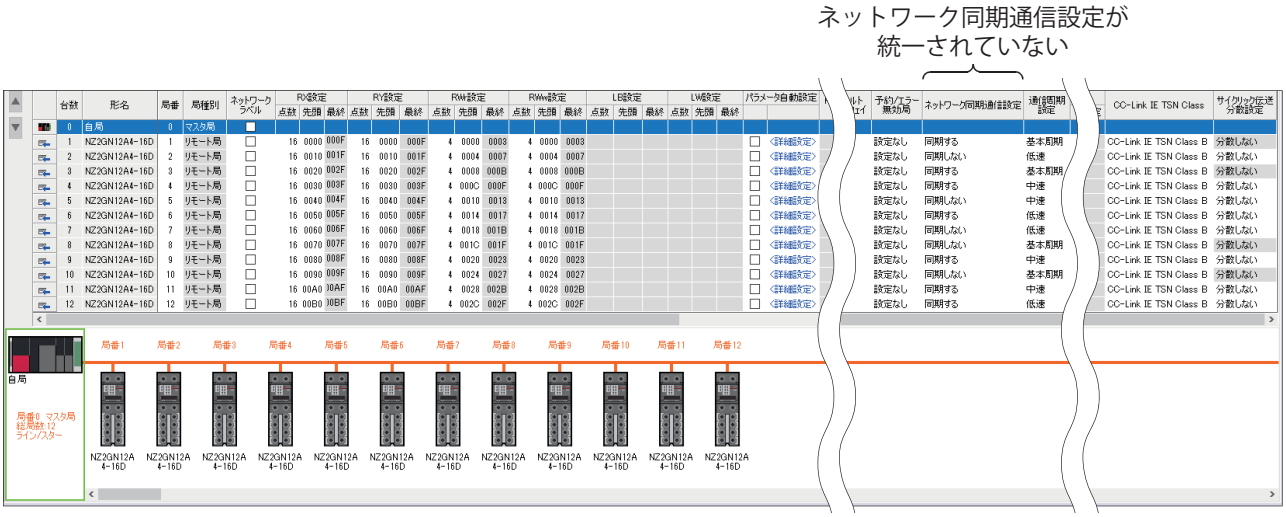
7.7 ネットワーク同期通信設定の統一

概要

CC-Link IE TSN Classが「Class B」のデバイス局の中で，ネットワーク同期通信設定が「同期する」と「同期しない」で混在している場合，どちらかの設定に統一することで，通信周期間隔およびサイクリック伝送時間を短縮できます。

実行手順

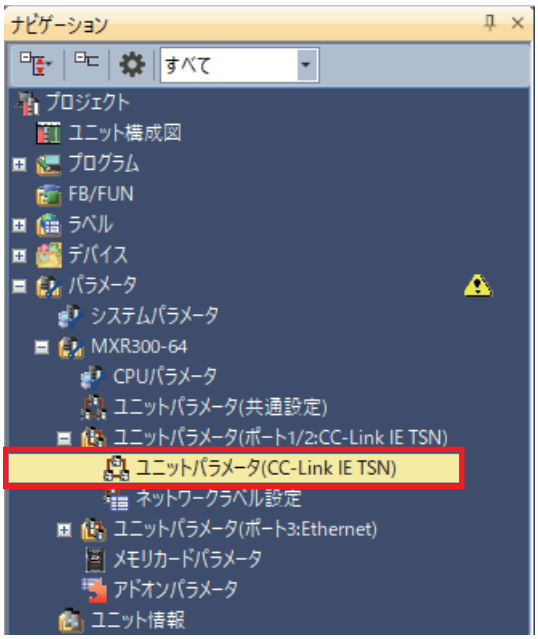
下記のネットワーク構成を例に，CC-Link IE TSN Classが「Class B」のデバイス局のネットワーク同期通信設定を統一する手順を示します。



操作手順

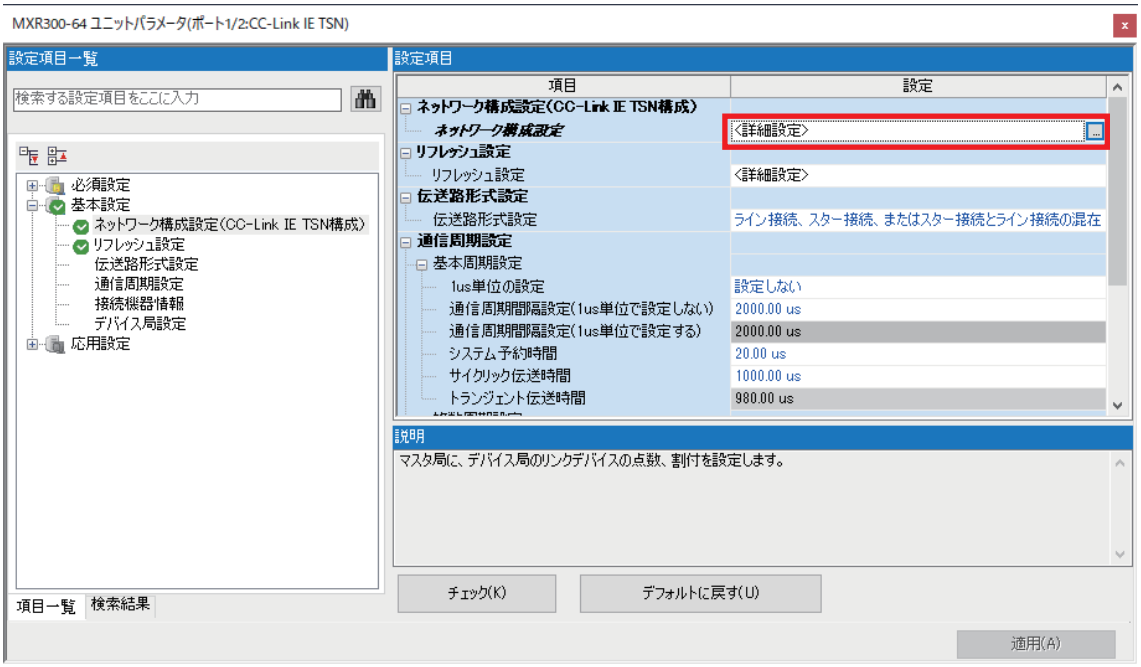
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック

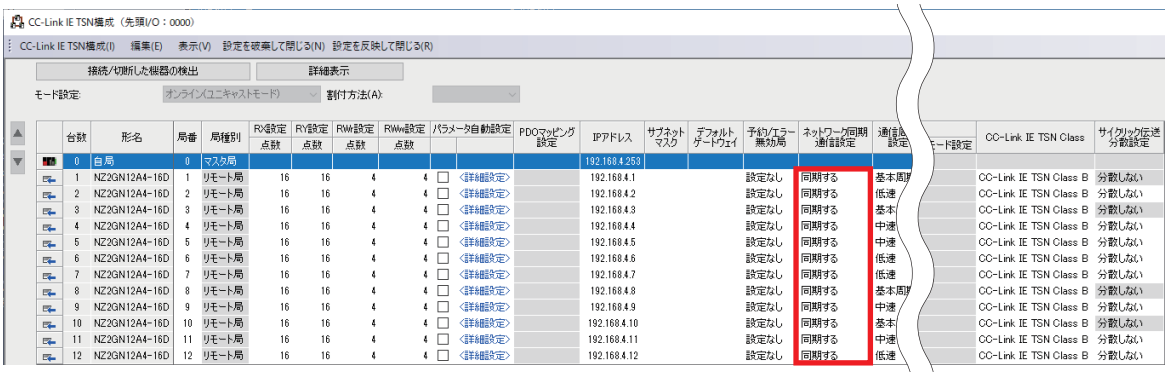


2. 基本設定から，ネットワーク構成設定を開きます。

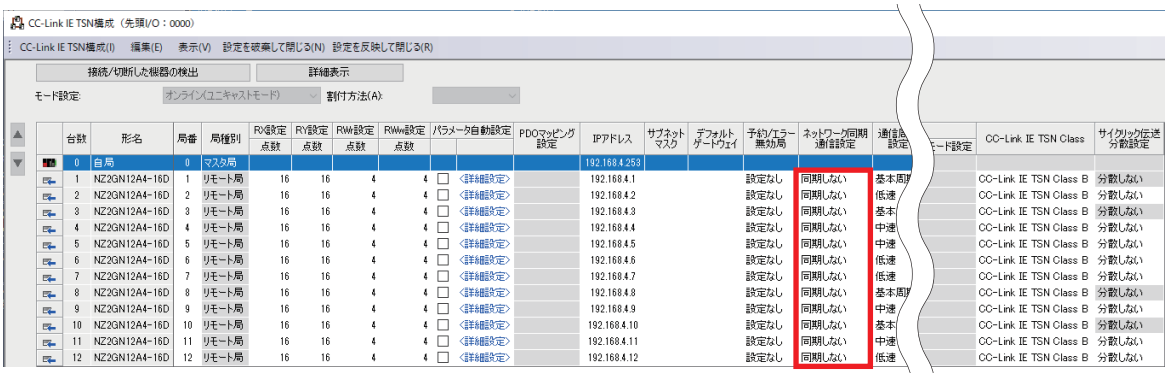
[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



3. CC-Link IE TSN Classが"Class B"に設定されたデバイス局の"ネットワーク同期通信設定"を，「同期する」または「同期しない」に設定します。



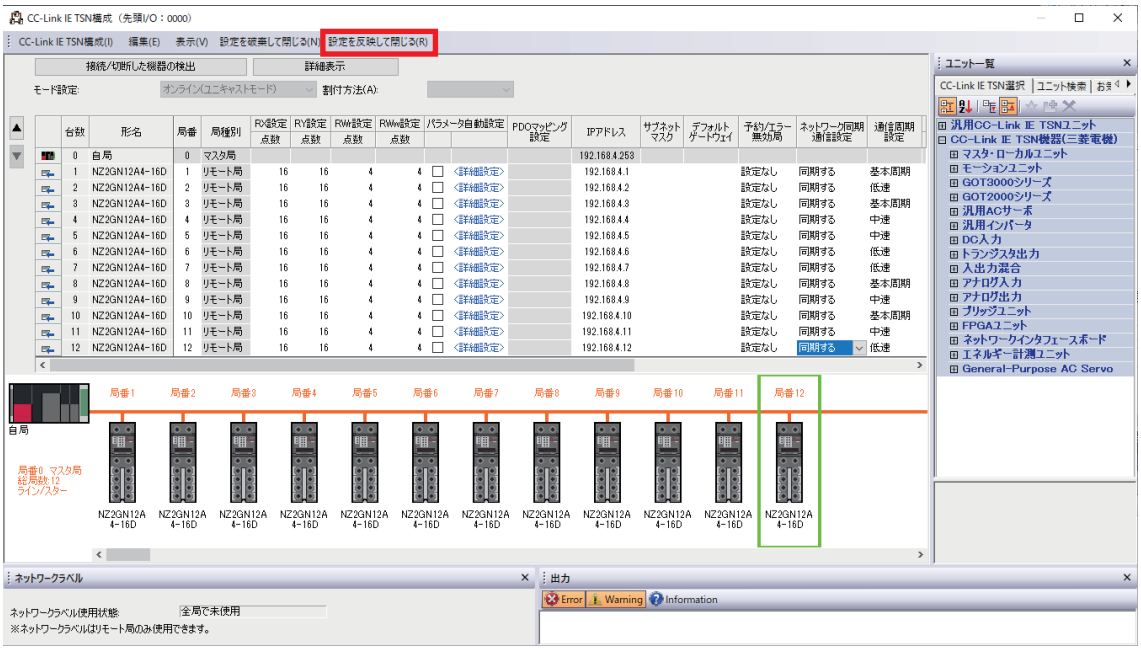
または



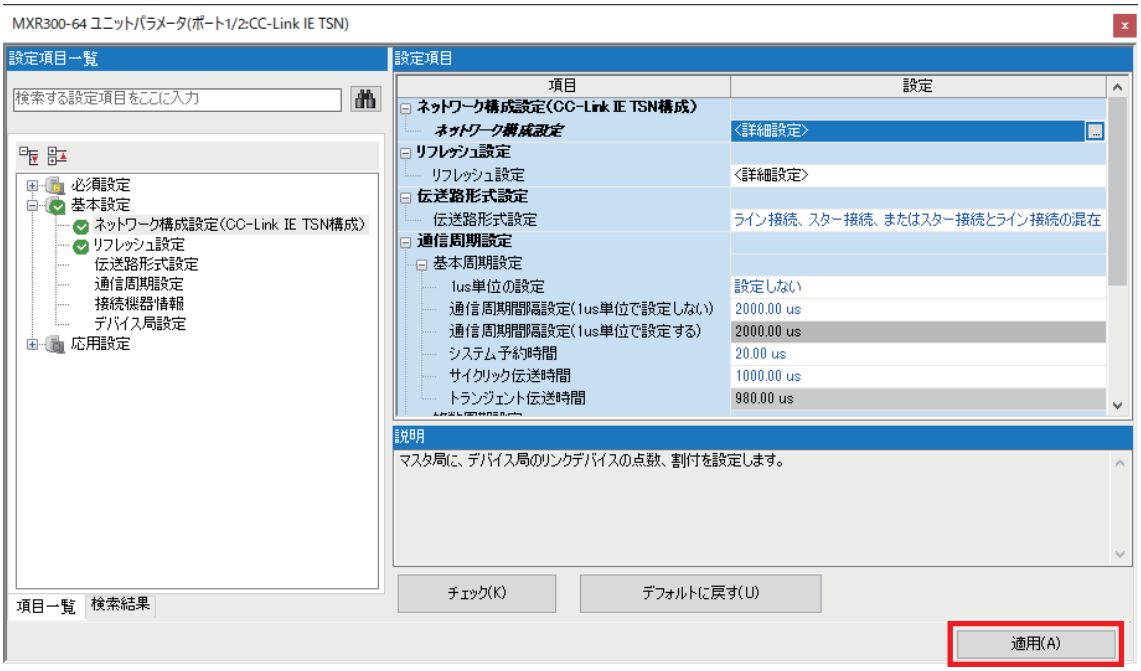
Point

ネットワーク同期通信設定の変更後，リンクデバイス番号を再割付することで，リンクリフレッシュ時間や通信周期間隔，サイクリック伝送時間をより短縮できる場合があります。
詳細については，下記を参照してください。
☞ 96ページ リンクリフレッシュ時間の短縮

4. [設定を反映して閉じる]ボタンをクリックします。



5. [適用]ボタンをクリックして設定を適用します。



注意事項

ネットワーク同期通信設定はCC-Link IE TSN Classが「Class B」のデバイス局のみ「同期する」の通信ができるため，この設定により使用するデバイス局や実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。

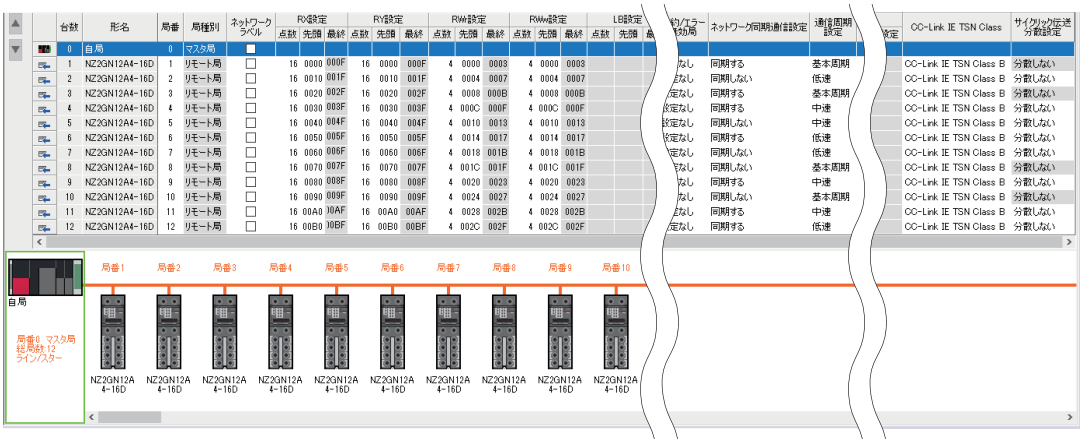
7.8 サイクリック伝送分散の設定

概要

CC-Link IE TSN Classが「Class B」かつ，通信周期設定が「中速」や「低速」のデバイス局の場合，通信周期内で局ごとにサイクリック伝送のタイミングを分散することで，基本周期の通信周期間隔およびサイクリック伝送時間を短縮できます。

実行手順

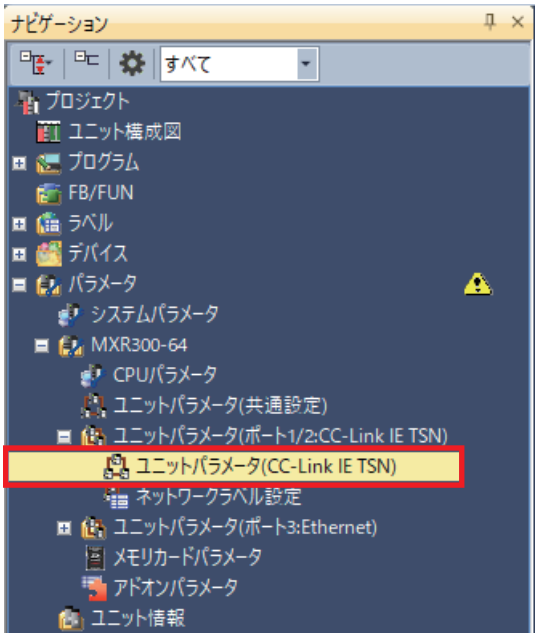
下記のネットワーク構成を例に，CC-Link IE TSN Classが「Class B」かつ，通信周期設定が「中速」や「低速」のデバイス局のサイクリック伝送のタイミングを分散する手順を示します。



操作手順

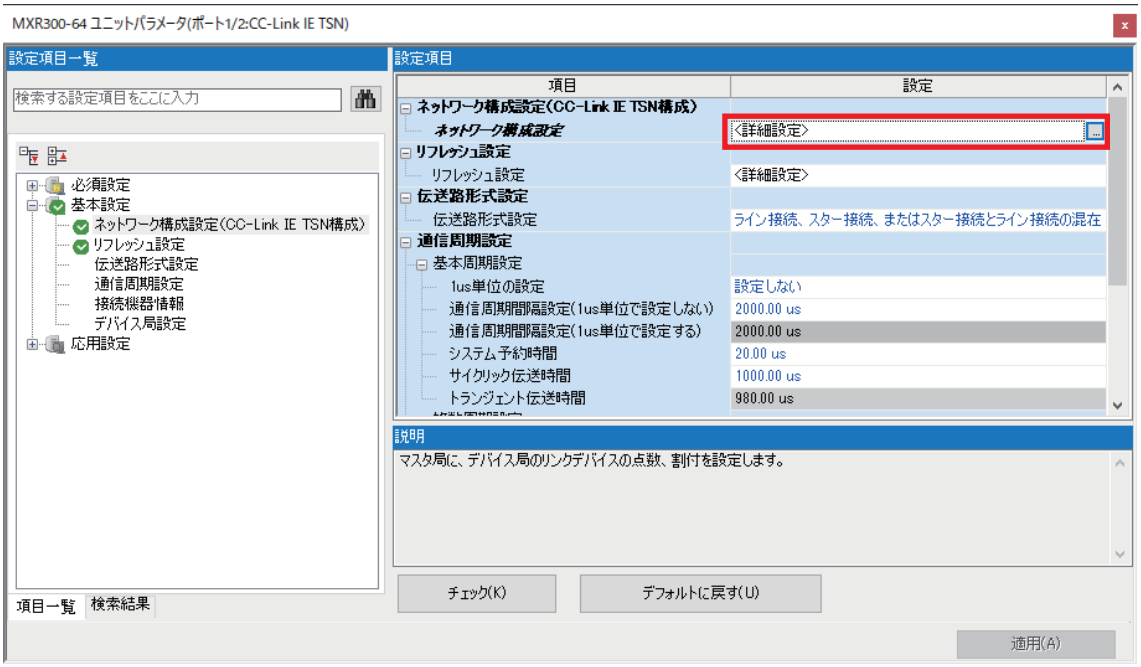
1. ユニットパラメータ(CC-Link IE TSN)を開きます。

ナビゲーションウィンドウ⇨[パラメータ]⇨[機種名]⇨[ユニットパラメータ(ポート1/2:CC-Link IE TSN)]⇨[ユニットパラメータ(CC-Link IE TSN)]をダブルクリック

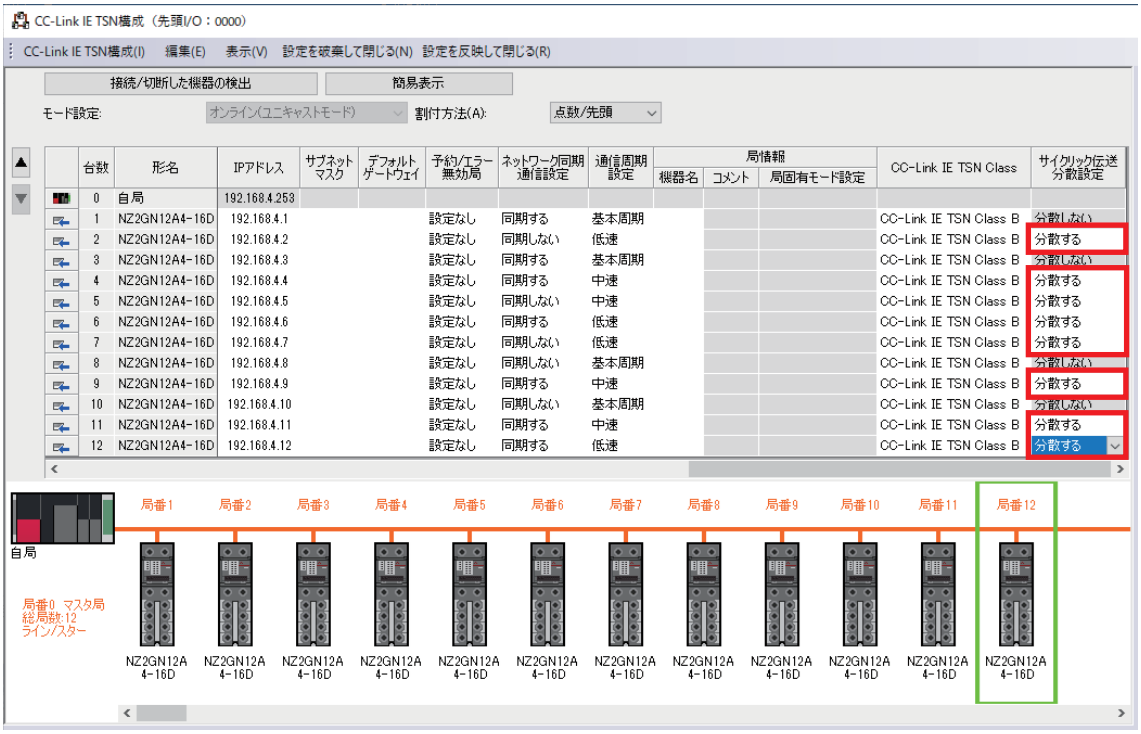


2. 基本設定から，ネットワーク構成設定を開きます。

[基本設定]⇨[ネットワーク構成設定(CC-Link IE TSN構成)]⇨[ネットワーク構成設定]の<詳細設定>をダブルクリック



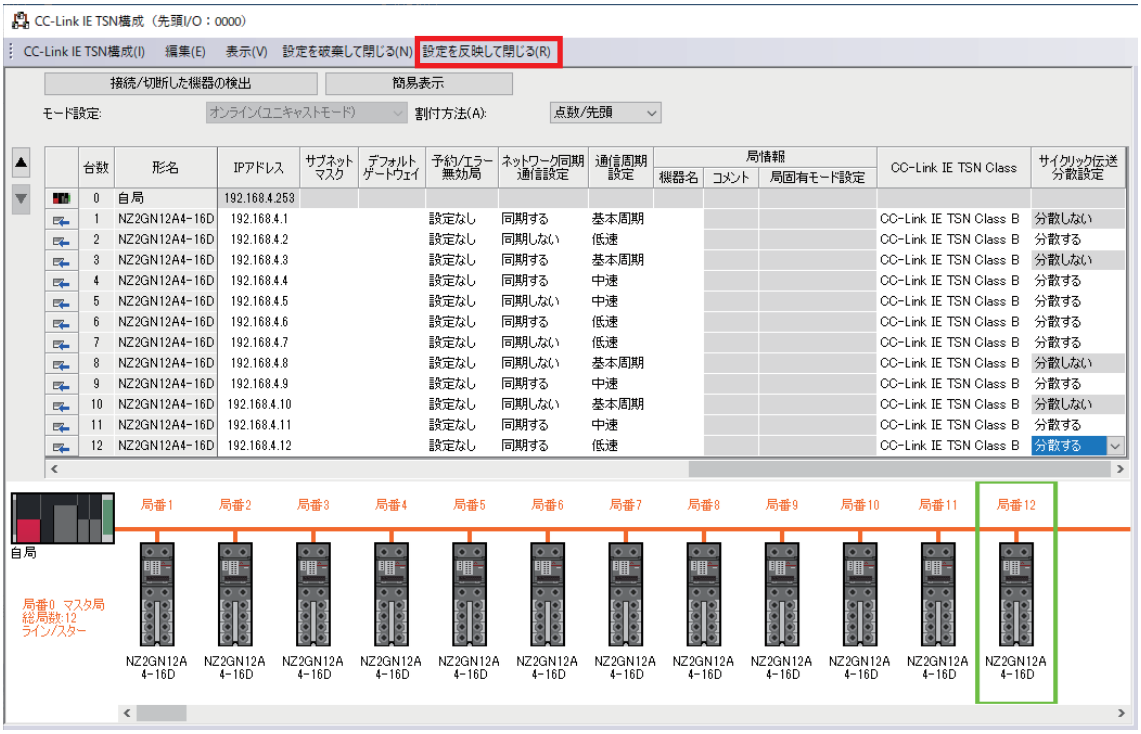
3. CC-Link IE TSN Classが"Class B"かつ通信周期が"中速"，または"低速"のデバイス局の"サイクリック伝送分散設定"を，「分散する」に設定します。



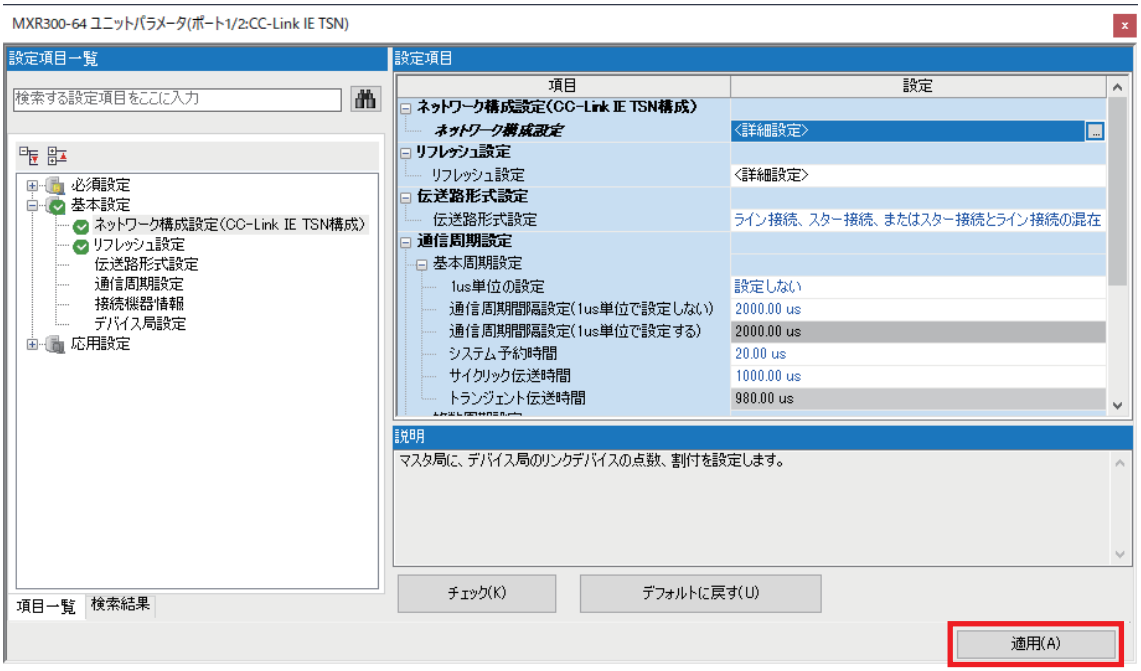
Point

"サイクリック伝送分散設定"の項目が表示されていない場合は，[詳細表示]ボタンをクリックして表示させてください。

4. [設定を反映して閉じる]ボタンをクリックします。



5. [適用]ボタンをクリックして設定を適用します。



注意事項

サイクリック伝送分散はCC-Link IE TSN Classが「Class B」かつ，通信周期設定が「中速」や「低速」のデバイス局のみ設定できるため，この設定により使用するデバイス局や実現可能なシステムに制限が生じる可能性があります。実現したいシステムの内容に応じて本テクニックの使用可否を判断してください。

MEMO

改訂履歴

*取扱説明書番号は，本説明書の裏表紙の左下に記載してあります。

改訂年月	*取扱説明書番号	改訂内容
2025年5月	BCN-89999-9684-A	初版

本書によって，工業所有権その他の権利の実施に対する保証，または実施権を許諾するものではありません。また本書の掲載内容の使用により起因する工業所有権上の諸問題については，当社は一切その責任を負うことができません。

© 2025 MITSUBISHI ELECTRIC CORPORATION

サービスのお問い合わせ

修理・サービスに関するお問い合わせはこちらにお問い合わせください。

三菱電機システムサービス株式会社

北日本支社	(022) 353-7814	北陸支店	(076) 252-9519
北海道支店	(011) 890-7515	関西支社	(06) 6458-9728
首都圏第２支社	(03) 3454-5521	京滋機器サービスステーション	(075) 874-3614
神奈川機器サービスステーション	(045) 938-5420	姫路機器サービスステーション	(079) 269-8845
関越機器サービスステーション	(048) 859-7521	中四国支社	(082) 285-2111
新潟機器サービスステーション	(025) 241-7261	岡山機器サービスステーション	(086) 242-1900
中部支社	(052) 722-7601	四国支店	(087) 831-3186
静岡機器サービスステーション	(054) 287-8866	九州支社	(092) 483-8208

商標

本文中における会社名，システム名，製品名などは，一般に各社の登録商標または商標です。
本文中で，商標記号(™，®)は明記していない場合があります。

BCN-89999-9684-A

三菱電機株式会社 〒100-8310 東京都千代田区丸の内2-7-3 (東京ビル)

お問い合わせは下記へどうぞ

本社機器営業部	〒110-0016	東京都台東区台東1-30-7 (秋葉原アイマークビル)	(03) 5812-1450
関越機器営業部	〒330-6034	さいたま市中央区新都心11-2 (明治安田生命さいたま新都心ビル)	(048) 600-5835
新潟支店	〒950-8504	新潟市中央区東大通2-4-10 (日本生命新潟ビル)	(025) 241-7227
神奈川機器営業部	〒220-8118	横浜市西区みなとみらい2-2-1 (横浜ランドマークタワー)	(045) 224-2624
北海道支社	〒060-0042	札幌市中央区大通西3-11 (北洋ビル)	(011) 212-3793
東北支社	〒980-0013	仙台市青葉区花京院1-1-20 (花京院スクエア)	(022) 216-4546
北陸支社	〒920-0031	金沢市広岡3-1-1 (金沢パークビル)	(076) 233-5502
中部支社	〒450-6423	名古屋市中村区名駅3-28-12 (大名古屋ビルヂング)	(052) 565-3314
豊田支店	〒471-0034	豊田市小坂本町1-5-10 (矢作豊田ビル)	(0565) 34-4112
関西支社	〒530-8206	大阪市北区大深町4-20 (グランフロント大阪タワーA)	(06) 6486-4122
中国支社	〒730-8657	広島市中区中町7-32 (ニッセイ広島ビル)	(082) 248-5348
四国支社	〒760-8654	高松市寿町1-1-8 (日本生命高松駅前ビル)	(087) 825-0055
九州支社	〒810-8686	福岡市中央区天神2-12-1 (天神ビル)	(092) 721-2247

三菱電機 FA 検索

www.MitsubishiElectric.co.jp/fa

メンバー登録無料!

インターネットによる情報サービス「三菱電機FAサイト」

三菱電機FAサイトでは、製品や事例などの技術情報に加え、トレーニングスクール情報や各種お問い合わせ窓口をご提供しています。また、メンバー登録いただくとマニュアルやCADデータ等のダウンロード、eラーニングなどの各種サービスをご利用いただけます。

仕様・機能に関するお問い合わせ

製品ごとにお問い合わせを受け付けております。
三菱電機FAサイト ― 仕様・機能に関するお問い合わせ
www.MitsubishiElectric.co.jp/fa/contact-us/spec/

BCN-89999-9684-A(2505)

2025年5月作成

本ガイドは，お断りなしに仕様を変更することがありますのでご了承ください。
本ガイドは，輸出する場合，経済産業省への役務取引許可申請は不要です。