MITSUBISHI ELECTRIC
Changes for the Better

家庭から宇宙まで、エコチェンジ。 eco Changes

# 三菱電機サーボシステム
# モーションユニットクイックスタートガイド

## さぁ、はじめよう！

# クイックスタートガイド

MELSEC iQ-Rシリーズ モーションユニット

MITSUBISHI ELECTRIC SERVO SYSTEM MELSERVO-J5

MELSEC iQ-R series

**対象機種**

- RD78G
- RD78GH
- MR-J5-G

# ● 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共に，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。

本マニュアルで示す注意事項は，本製品に関するもののみについて記載したものです。シーケンサシステムとしての安全上のご注意に関しては，MELSEC iQ-R ユニット構成マニュアルを参照してください。

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。

## 【設計上の注意事項】

### ⚠警告

- 外部電源の異常やシーケンサ本体の故障時でも，システム全体が安全側に働くようにシーケンサの外部で安全回路を設けてください。誤出力または誤動作により，事故の恐れがあります。
  (1) 非常停止回路，保護回路，正転／逆転などの相反する動作のインタロック回路，位置決めの上限／下限など機械の破損防止のインタロック回路は，シーケンサの外部で構成してください。
  (2) シーケンサは次の異常状態を検出すると，演算を停止し，出力は下記の状態になります。
  ・電源ユニットの過電流保護装置または過電圧保護装置が働いたときは全出力をOFFする。
  ・CPUユニットでウォッチドッグタイマエラーなどの自己診断機能で異常を検出したときは，パラメータ設定により，全出力を保持またはOFFする。
  (3) CPUユニットで検出できない入出力制御部分などの異常時は，全出力がONすることがあります。このとき，機械の動作が安全側に働くよう，シーケンサの外部でフェールセーフ回路を構成したり，安全機構を設けたりしてください。フェールセーフ回路例については，MELSEC iQ-Rユニット構成マニュアルの「フェールセーフ回路の考え方」を参照してください。
  (4) 出力回路のリレーやトランジスタなどの故障によっては，出力がONの状態やOFFの状態を保持することがあります。重大な事故につながるような出力信号については，外部で監視する回路を設けてください。
- 出力回路において，定格を超える負荷電流または負荷短絡などによる過電流が長時間継続して流れた場合，発煙や発火の恐れがありますので，外部にヒューズなどの安全回路を設けてください。
- シーケンサ本体の電源立上げ後に，外部供給電源を投入するように回路を構成してください。外部供給電源を先に立ち上げると，誤出力または誤動作により，事故の恐れがあります。
- ネットワークが交信異常になったときの各局の動作状態については，ご使用のネットワークのマニュアルを三菱電機FAサイトよりダウンロードして参照してください。誤出力または誤動作により，事故の恐れがあります。
- CPUユニットまたはインテリジェント機能ユニットに外部機器を接続して，運転中のシーケンサに対する制御(データ変更)を行うときは，常にシステム全体が安全側に働くように，プログラム上でインタロック回路を構成してください。 また，運転中のシーケンサに対するその他の制御(プログラム変更，パラメータ変更，強制出力，運転状態変更(状態制御))を行うときは，マニュアルを熟読し，十分に安全を確認してから行ってください。確認を怠ると，操作ミスにより機械の破損や事故の原因になります。なお，安全CPUの場合セーフティモードの運転中に，制御(データ変更)はできません。
- 外部機器から遠隔地のシーケンサに対する制御では，データ交信異常によりシーケンサ側のトラブルにすぐに対応できない場合があります。プログラム上でインタロック回路を構成すると共に，データ交信異常が発生したときのシステムとしての処置方法を外部機器とCPUユニット間で取り決めてください。
- ユニットのバッファメモリの中で，システムエリアまたは書込み不可のエリアにはデータを書き込まないでください。また，CPUユニットから各ユニットに対する出力信号の中で，使用禁止の信号を出力(ON)しないでください。システムエリアまたは書込み不可のエリアに対するデータの書込み，使用禁止の信号に対する出力を行うと，シーケンサシステムが誤動作する危険性があります。システムエリアまたは書込み不可のエリア，使用禁止の信号については，各ユニットのユーザーズマニュアルを参照してください。なお，安全通信で使用するエリアには，お客様による書き込みができないため，安全通信が誤動作することはありません。

## 【設計上の注意事項】

### ⚠警告

- 通信ケーブルが断線した場合は，回線が不安定になり，複数の局でネットワークが交信異常になる場合があります。交信異常が発生しても，システムが安全側に働くようにプログラム上でインタロック回路を構成してください。誤出力または誤動作により，事故の恐れがあります。なお，安全通信については，安全局インタロック機能によるインタロックが働きます。
- 外部電源の異常やシーケンサ本体の故障時でも，システム全体が安全側に働くようにシーケンサの外部で安全回路を設けてください。誤出力または誤動作により，事故の恐れがあります。
  (1) 機械原点復帰制御は，原点復帰方向と原点復帰速度の2つのデータによって制御され，近点ドグONにて減速を開始します。したがって，原点復帰方向を誤って設定すると減速せずに運転し続ける場合があるので，機械破損防止のインタロック回路をシーケンサの外部で構成してください。
  (2) ユニットがエラー検出時，パラメータの停止グループの設定により，通常の減速停止または急停止を行います。パラメータは，位置決めシステムの仕様に合わせてください。また原点復帰用パラメータおよび位置決めデータはパラメータの設定値以内にしてください。
  (3) ユニットで検出できない出力回路の絶縁素子やトランジスタなどの部品の故障によっては，出力がONの状態やOFFの状態を保持する，または不定になることがあります。重大な事故につながるようなシステムにおいては，出力信号を監視する回路を設けてください。
- ユニット，ドライブユニット，サーボモータを使用したシステムとしての安全基準(たとえばロボットなどの安全通則など)のあるものは安全基準を満足させてください。
- ユニット，ドライブユニットの異常時動作とシステムとしての安全方向動作が異なる場合はユニット・ドライブユニットの外部で対策回路を構成してください。

## 【設計上の注意事項】

### ⚠注意

- 制御線や通信ケーブルは，主回路や動力線と束線したり，近接させたりしないでください。
  100 mm以上を目安として離してください。ノイズにより，誤動作の原因になります。
- ランプ負荷，ヒータ，ソレノイドバルブなどの誘導性負荷を制御するときは，出力のOFF→ON時に大きな電流(通常の10倍程度)が流れる場合がありますので，定格電流に余裕のあるユニットをお使いください。
- CPUユニットの電源OFF→ONまたはリセット時，CPUユニットがRUN状態になるまでの時間が，システム構成，パラメータ設定，プログラム容量などにより変動します。RUN状態になるまでの時間が変動しても，システム全体が安全側に働くように設計してください。
- 各種設定を登録中に，ユニット装着局の電源OFFおよびCPUユニットのリセットを行わないでください。登録中にユニット装着局の電源OFFおよびCPUユニットのリセットを行うと，フラッシュROM内，SDメモリカードのデータ内容が不定となり，バッファメモリへの設定値の再設定，フラッシュROM，SDメモリカードへの再登録が必要です。また，ユニットの故障および誤動作の原因になります。

## 【セキュリティ上の注意事項】

| ⚠警告 |
| --- |
| ● ネットワーク経由による外部機器からの不正アクセス，DoS攻撃，コンピュータウイルスその他のサイバー攻撃に対して，シーケンサ，およびシステムのセキュリティ(可用性，完全性，機密性)を保つため，ファイアウォールやVPNの設置，コンピュータへのアンチウイルスソフト導入などの対策を盛り込んでください。 |

## 【取付け上の注意事項】

| ⚠警告 |
| --- |
| ● ユニットの着脱は，必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと，感電，ユニットの故障や誤動作の原因になります。 |

## 【取付け上の注意事項】

| ⚠注意 |
| --- |
| ● シーケンサは，安全にお使いいただくために(ベースユニットに同梱のマニュアル)記載の一般仕様の環境で使用してください。一般仕様の範囲以外の環境で使用すると，感電，火災，誤動作，製品の損傷または劣化の原因になります。<br>● ユニットを装着するときは，ユニット下部の凹部をベースユニットのガイドに挿入し，ガイドの先端を支点として押し，必ずネジで締め付けてください。ユニットが正しく装着されていないと，誤動作，故障または落下の原因になります。<br>● ユニット固定用フックのないユニットを装着するときは，ユニット下部の凹部をベースユニットのガイドに挿入し，ガイドの先端を支点として押し，必ずネジで締め付けてください。ユニットが正しく装着されていないと，誤動作，故障または落下の原因になります。<br>● ネジの締付けは，規定トルク範囲内で行ってください。ネジの締付けがゆるいと，落下，短絡または誤動作の原因になります。ネジを締め過ぎると，ネジやユニットの破損による落下，短絡または誤動作の原因になります。<br>● 増設ケーブルは，ベースユニットの増設ケーブル用コネクタに確実に装着してください。装着後に，浮上りがないか確認してください。接触不良により，誤動作の原因になります。<br>● SDメモリカードは，装着スロットに押し込んで確実に装着してください。装着後に，浮上りがないか確認してください。接触不良により，誤動作の原因になります。<br>● 拡張SRAMカセットまたはバッテリレスオプションカセットは，CPUユニットのカセット接続用コネクタに押し込んで確実に装着してください。装着後はカセットカバーを閉め，浮上りがないか確認してください。接触不良により，誤動作の原因になります。<br>● ユニット，SDメモリカード，拡張SRAMカセット，バッテリレスオプションカセットまたはコネクタの，導電部分に直接触らないでください。ユニットの故障や誤動作の原因になります。 |

## 【配線上の注意事項】

| ⚠警告 |
| --- |
| ● 取付けまたは配線作業は，必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと，感電，ユニットの故障や誤動作の原因になります。<br>● 取付けまたは配線作業後，通電または運転を行う場合は，空きスロットにブランクカバーユニット(RG60)を，増設ケーブル用コネクタに付属の増設コネクタ保護カバーを取り付けてください。ブランクカバーユニット(RG60)や増設コネクタ保護カバーを取り付けないと，感電の恐れがあります。 |

## 【配線上の注意事項】

### ⚠注意

- FG端子およびLG端子は，シーケンサ専用のD種接地(第三種接地)以上で必ず接地してください。感電または誤動作の恐れがあります。
- 圧着端子は適合圧着端子を使用し，規定のトルクで締め付けてください。先開形圧着端子を使用すると，端子ネジがゆるんだ場合に脱落し，故障の原因になります。
- ユニットへの配線は，製品の定格電圧および信号配列を確認後，正しく行ってください。定格と異なった電源を接続したり，誤配線したりすると，火災または故障の原因になります。
- 外部機器接続用コネクタは，メーカ指定の工具で圧着，圧接または正しくハンダ付けしてください。接続が不完全な場合，短絡，火災または誤動作の原因になります。
- コネクタは，確実にユニットに取り付けてください。接触不良により，誤動作の原因になります。
- 制御線や通信ケーブルは，主回路や動力線と束線したり，近接させたりしないでください。
  100 mm以上を目安として離してください。ノイズにより，誤動作の原因になります。
- ユニットに接続する電線やケーブルは，必ずダクトに納めるか，またはクランプによる固定処理を行ってください。電線やケーブルをダクトに納めなかったり，クランプによる固定処理をしていないと，ケーブルのふらつきや移動，不注意の引っ張りなどによる誤動作またはユニットやケーブルの破損の原因になります。
  特に振動，衝撃の大きい場所で使用する場合は，電線やケーブルの重量がユニットへの負荷となる場合があります。
  増設ケーブルには，外皮を取り除いたクランプ処理を行わないでください。ケーブルの特性変化により，誤動作の原因になります。
- ケーブル接続は，接続するインタフェースの種類を確認の上，正しく行ってください。異なったインタフェースに接続または誤配線すると，ユニットまたは外部機器の故障の原因になります。
- 端子ネジやコネクタ取付けネジの締付けは，規定トルク範囲内で行ってください。ネジの締付けがゆるいと，落下，短絡，火災または誤動作の原因になります。ネジを締め過ぎると，ネジやユニットの破損による落下，短絡，火災または誤動作の原因になります。
- ユニットに接続されたケーブルを取りはずすときは，ケーブル部分を引っ張らないでください。コネクタ付きのケーブルは，ユニットの接続部分のコネクタを持って取りはずしてください。端子台接続のケーブルは，端子台端子ネジを緩めてから取りはずしてください。ユニットに接続された状態でケーブルを引っ張ると，誤動作またはユニットやケーブルの破損の原因になります。
- ユニット内に，切粉や配線クズなどの異物が入らないように注意してください。火災，故障または誤動作の原因になります。
- 配線時にユニット内へ配線クズなどの異物混入を防止するため，ユニット上部に混入防止ラベルを貼り付けています。配線作業中は，本ラベルをはがさないでください。システム運転時は，放熱のために本ラベルを必ずはがしてください。
- シーケンサは，制御盤内に設置して使用してください。制御盤内に設置されたシーケンサ電源ユニットへの主電源配線に関しては，中継端子台を介して行ってください。また，電源ユニットの交換と配線作業は，感電保護に対して，十分に教育を受けたメンテナンス作業者が行ってください。配線方法は，MELSEC iQ-Rユニット構成マニュアルを参照してください。
- システムで使用するEthernetケーブルは，本マニュアル記載の仕様に従ってください。仕様外の配線では，正常なデータ伝送を保証できません。

## 【立上げ・保守時の注意事項】

### ⚠警告

- 通電中，端子に触れないでください。感電または誤動作の原因になります。
- バッテリコネクタは，正しく接続してください。バッテリに充電，分解，加熱，火中投入，ショート，ハンダ付け，液体を付着させる，強い衝撃を与えることは絶対に行わないでください。バッテリの取扱いを誤ると，発熱，破裂，発火，液漏れにより，ケガまたは火災の恐れがあります。
- 端子ネジ，コネクタ取付けネジまたはユニット固定ネジの増し締めや，ユニットの清掃は，必ずシステムで使用している外部供給電源を全相遮断してから行ってください。 全相遮断しないと，感電の恐れがあります。

## 【立上げ・保守時の注意事項】

### ⚠注意

- CPUユニットまたはインテリジェント機能ユニットに外部機器を接続して，運転中のシーケンサに対する制御(データ変更)を行うときは，常にシステム全体が安全側に働くように，プログラム上でインタロック回路を構成してください。また，運転中のシーケンサに対するその他の制御(プログラム変更，パラメータ変更，強制出力，運転状態変更(状態制御))を行うときは，マニュアルを熟読し，十分に安全を確認してから行ってください。確認を怠ると，操作ミスにより機械の破損や事故の原因になります。
- 外部機器から遠隔地のシーケンサに対する制御では，データ交信異常により，シーケンサ側のトラブルにすぐに対応できない場合があります。プログラム上でインタロック回路を構成すると共に，データ交信異常が発生したときのシステムとしての処置方法を外部機器とCPUユニット間で取り決めてください。
- ユニットの分解または改造はしないでください。故障，誤動作，ケガまたは火災の原因になります。
- 携帯電話やPHSなどの無線通信機器は，シーケンサ本体の全方向から25 cm以上離して使用してください。誤動作の原因になります。
- ユニットの着脱は，必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと，ユニットの故障や誤動作の原因になります。
- ネジの締付けは，規定トルク範囲内で行ってください。ネジの締付けがゆるいと，部品や配線の落下，短絡または誤動作の原因になります。ネジを締め過ぎると，ネジやユニットの破損による落下，短絡または誤動作の原因になります。
- 下記の着脱は，製品ご使用後，50回以内(JIS B 3502，IEC 61131-2に準拠)としてください。
  なお，50回を超えた場合は，誤動作の原因となる恐れがあります。
  - ・ユニットとベースユニット
  - ・CPUユニットと，拡張SRAMカセットまたはバッテリレスオプションカセット
  - ・ユニットと端子台
- SDメモリカードの取付け・取りはずしは，製品使用後，500回以内としてください。500回を超えた場合は，誤動作の原因となる恐れがあります。
- SDメモリカード取扱い時は，剥き出しになっているカード端子に触れないでください。故障や誤動作の原因になります。

## 【立上げ・保守時の注意事項】

### ⚠注意

- 拡張SRAMカセットまたはバッテリレスオプションカセット取扱い時は，基板上のICに触れないでください。故障や誤動作の原因になります。
- ユニットに装着するバッテリには，落下・衝撃を加えないでください。落下・衝撃により，バッテリが破損し，バッテリ液の液漏れがバッテリ内部で発生している恐れがあります。落下・衝撃を加えたバッテリは使用せずに廃棄してください。
- 制御盤内での立上げ・保守作業は，感電保護に対して，十分に教育を受けたメンテナンス作業者が行ってください。また，メンテナンス作業者以外が制御盤を操作できないよう，制御盤に鍵をかけてください。
- ユニットに触れる前には，必ず接地された金属などの導電物に触れて，人体などに帯電している静電気を放電させてください。静電気を放電させないと，ユニットの故障や誤動作の原因になります。
- 試運転は，パラメータの速度制限値を遅い速度に設定し，危険な状態が発生したとき即座に停止できる準備をしてから動作確認を行ってください。
- 運転前にプログラムおよび各パラメータの確認・調整を行ってください。機械によっては予期しない動きとなる場合があります。
- 絶対位置システム機能を使用している場合，新規立上げしたとき，またはユニット，絶対位置対応モータ等を交換したときは必ず原点復帰を行ってください。
- ブレーキ機能を確認してから運転を行ってください。
- 点検時にメガテスト(絶縁抵抗測定)を行わないでください。
- 保守・点検終了時，絶対位置検出機能の位置検出が正しいか確認してください。
- 電気設備に関する教育を受け，十分な知識を有する人のみ制御盤を開けることができるよう，制御盤に鍵をかけてください。

## 【運転時の注意事項】

| ⚠注意 |
| --- |
| ● インテリジェント機能ユニットにパソコンなどの外部機器を接続して運転中のシーケンサに対する制御(特にデータ変更，プログラム変更，運転状態変更(状態制御))を行うときはユーザーズマニュアルを熟読し，十分に安全を確認してから行ってください。データ変更，プログラム変更，状態制御を誤ると，システムの誤動作，機械の破損や事故の原因になります。<br>● ユニット内のフラッシュROMへバッファメモリの設定値を登録して使用する場合，登録中はユニット装着局の電源OFFおよびCPUユニットのリセットを行わないでください。登録中にユニット装着局の電源OFFおよびCPUユニットのリセットを行うと，フラッシュROM内，SDメモリカードのデータ内容が不定となり，バッファメモリへの設定値の再設定，フラッシュROM，SDメモリカードへの再登録が必要です。また，ユニットの故障や誤動作の原因になります。<br>● 補間運転の基準軸速度指定のときは，相手軸(2軸目，3軸目，4軸目)の速度が設定速度より大きく(速度制限値以上)なる場合がありますのでご注意ください。<br>● 試験運転やティーチングなどの運転中は機械に近寄らないでください。傷害の原因になります。 |

## 【廃棄時の注意事項】

| ⚠注意 |
| --- |
| ● 製品を廃棄するときは，産業廃棄物として扱ってください。<br>● バッテリを廃棄する際は，各地域にて定められている法令に従い分別を行ってください。EU加盟国内でのバッテリ規制の詳細については，MELSEC iQ-Rユニット構成マニュアルを参照してください。 |

## 【輸送時の注意事項】

| ⚠注意 |
| --- |
| ● リチウムを含有しているバッテリの輸送時は，輸送規制に従った取扱いが必要です。規制対象機種の詳細については，MELSEC iQ-Rユニット構成マニュアルを参照してください。<br>● 木製梱包材の消毒および除虫対策のくん蒸剤に含まれるハロゲン系物質(フッ素，塩素，臭素，ヨウ素など)が当社製品に侵入すると故障の原因になります。残留したくん蒸成分が当社製品に侵入しないようにご注意いただくか，くん蒸以外の方法(熱処理など)で処理してください。なお，消毒および除虫対策は梱包前の木材の段階で実施してください。 |

## 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 |
|---|---|---|
| 2020年 7月 | L(名)03190-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。

© 2020 MITSUBISHI ELECTRIC CORPORATION

# は　じ　め　に

ご使用前に本書をよくお読みいただき，サーボシステム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，本書で紹介するプログラム例を実際のシステムへ流用される場合は，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。

## 目　　次


安全上のご注意 …… A- 1
改訂履歴 …… A-10
目次 …… A-11

**1　概 要　1- 1～1-6**

1.1　サーボシステムとは …… 1- 1
1.2　サーボシステムの用途例 …… 1- 1
1.3　サーボシステムコントローラの役割 …… 1- 2
1.4　MELSEC iQ-RシリーズモーションユニットRD78G(H)の特長 …… 1- 2
1.5　モーションユニットのプログラミング …… 1- 3
1.6　本書の構成 …… 1- 4
1.7　関連マニュアル …… 1- 5

**2　システムの立ち上げ　2- 1～2-12**

2.1　装置の概要 …… 2- 1
2.2　システム構成 …… 2- 2
2.3　機器の準備 …… 2- 3
2.4　ファームウェア／ソフトウェアのインストール …… 2- 4
2.4.1　ファームウェア／ソフトウェアの準備 …… 2- 4
2.4.2　MELSOFT GX Works3のインストール …… 2- 5
2.4.3　プロファイルの登録 …… 2- 6
2.4.4　MELSEC iQ-Rシリーズ モーションユニットFBライブラリ …… 2- 7
2.5　ユニットの装着 …… 2- 8
2.6　配線，およびケーブルの接続 …… 2- 9

**3　パラメータの設定　3- 1～3-36**

3.1　パラメータの設定手順 …… 3- 1
3.2　プロジェクトの作成 …… 3- 2
3.3　パソコンとシーケンサCPUユニットの接続 …… 3- 3
3.4　初期設定 …… 3- 4
3.4.1　シーケンサCPUユニットの初期化 …… 3- 4
3.4.2　ファームウェアバージョンの確認 …… 3- 5
3.5　ユニット構成図の作成 …… 3- 6
3.6　ネットワーク構成 …… 3- 8
3.6.1　ネットワーク構成の設定 …… 3- 8
3.6.2　PDOマッピング …… 3-10
3.6.3　モーション管理局の設定 …… 3-10
3.7　サーボパラメータの設定 …… 3-11
3.8　ユニットパラメータの適用 …… 3-14

3.9 モーション制御設定 ····· 3-15
3.10 基本設定，システム設定 ····· 3-15
3.11 軸パラメータの設定 ····· 3-16
3.11.1 局アドレス，軸種別の設定 ····· 3-16
3.11.2 各項目の設定 ····· 3-19
3.11.3 ドライバ単位変換（電子ギア） ····· 3-22
3.11.4 軸パラメータの反映 ····· 3-23
3.12 軸グループの設定 ····· 3-24
3.13 ラベル ····· 3-26
3.13.1 ラベルの作成 ····· 3-26
3.14 公開ラベル ····· 3-28
3.14.1 公開ラベルの登録 ····· 3-28
3.14.2 公開ラベルの反映 ····· 3-29
3.14.3 公開ラベルの使用例 ····· 3-30
3.15 モーションユニットへの書き込み ····· 3-31
3.16 プロジェクトの保存 ····· 3-32
3.17 パラメータ一覧 ····· 3-33

**4 シーケンサCPUでのプログラミング 4- 1～4-56**

4.1 シーケンサCPUのプログラム作成手順 ····· 4- 2
4.1.1 プログラムブロックの作成 ····· 4- 2
4.1.2 プログラム実行タイプ ····· 4- 3
4.1.3 FBの入力方法 ····· 4- 4
4.2 ラベルの命名規則 ····· 4-11
4.3 プロジェクト構成 ····· 4-12
4.3.1 プログラム名 ····· 4-12
4.3.2 グローバルラベル，公開ラベル設定 ····· 4-13
4.4 シーケンサレディ （プログラム名：ServoON_Jog) ····· 4-17
4.5 サーボON（プログラム名：ServoON_Jog) ····· 4-18
4.6 JOG運転 （プログラム名：ServoON_Jog) ····· 4-19
4.7 原点復帰 （プログラム名：Homing) ····· 4-24
4.8 単軸位置決め制御（プログラム名：Positioning) ····· 4-27
4.9 単軸連続位置決め（プログラム名：ContinuousPositioning) ····· 4-29
4.10 補間制御 （プログラム名：LinearInterpolation) ····· 4-33
4.10.1 補間制御の手順 ····· 4-33
4.10.2 軸グループ有効化／無効化 ····· 4-33
4.10.3 補間制御 ····· 4-33
4.10.4 直線補間のプログラム例 ····· 4-34
4.11 同期制御（プログラム名：Synchronous) ····· 4-39
4.11.1 同期制御の手順 ····· 4-39
4.11.2 演算プロファイル ····· 4-40
4.11.3 単軸同期用FB ····· 4-42
4.11.4 軸構成 ····· 4-42
4.11.5 プログラム例 ····· 4-43
4.12 エラーリセット（プログラム名：ErrorReset) ····· 4-48
4.13 動作確認 ····· 4-49
4.13.1 プログラムの変換と書き込み ····· 4-49
4.13.2 軸モニタ ····· 4-50
4.13.3 プログラムモニタ ····· 4-52
4.13.4 ウォッチ ····· 4-53

4.13.5 イベント履歴 ····· 4-55

5 シーケンサCPUとモーションユニットに分散したプログラミング 5- 1～5-56

5.1 モーションユニットのプログラム作成手順 ····· 5- 2
5.1.1 プログラムブロックの作成 ····· 5- 2
5.1.2 プログラム実行タイプ ····· 5- 3
5.1.3 FBの入力方法 ····· 5- 4
5.1.4 ENUM列挙子 ····· 5-10
5.1.5 全変換，公開ラベルの反映 ····· 5-11
5.2 ラベルの命名規則 ····· 5-12
5.3 プロジェクト構成 ····· 5-13
5.3.1 プログラム名 ····· 5-13
5.3.2 グローバルラベル，公開ラベル設定 ····· 5-15
5.4 シーケンサレディ（プログラム名：ServoON_Jog) ····· 5-18
5.5 サーボON（プログラム名：ServoON_Jog) ····· 5-19
5.6 JOG運転（プログラム名：ServoON_Jog) ····· 5-20
5.7 原点復帰（プログラム名：Homing) ····· 5-25
5.8 単軸位置決め制御（プログラム名：Positioning) ····· 5-28
5.9 単軸連続位置決め （プログラム名：ContinuousPositioning) ····· 5-30
5.10 補間制御 （プログラム名：LinearInterpolation) ····· 5-34
5.10.1 補間制御の手順 ····· 5-34
5.10.2 軸グループ有効化／無効化 ····· 5-34
5.10.3 補間制御 ····· 5-34
5.10.4 直線補間のプログラム例 ····· 5-35
5.11 同期制御 （プログラム名：Synchronous) ····· 5-40
5.11.1 同期制御の手順 ····· 5-40
5.11.2 演算プロファイル ····· 5-41
5.11.3 単軸同期用FB ····· 5-43
5.11.4 軸構成 ····· 5-43
5.11.5 プログラム例 ····· 5-44
5.12 エラーリセット（プログラム名：ErrorReset) ····· 5-50
5.13 動作確認 ····· 5-51
5.13.1 プログラムの変換と書き込み ····· 5-51
5.13.2 軸モニタ ····· 5-52
5.13.3 プログラムモニタ ····· 5-53
5.13.4 ウォッチ ····· 5-54
5.13.5 イベント履歴 ····· 5-56

付 録 付- 1～付-34

付1 プログラムの流用 ····· 付- 1
付1.1 モーションユニットデータのエクスポート ····· 付- 1
付1.2 サーボパラメータの保存 ····· 付- 2
付1.3 流用先プログラムへの反映 ····· 付- 3
付1.4 全変換，公開ラベルの反映 ····· 付- 7
付2 外部入力信号を使用する場合 ····· 付- 8
付2.1 サーボアンプのDI信号を使用する場合 ····· 付- 8
付2.2 シーケンサCPUの入力信号を使用する場合 ····· 付-16
付2.3 リモート入力ユニットの入力信号を使用する場合 ····· 付-23
付3 絶対位置検出システムで使用する場合の注意事項 ····· 付-30

付4　シーケンスプログラムで制御する場合 ····· 付-31

# 第１章　概　要

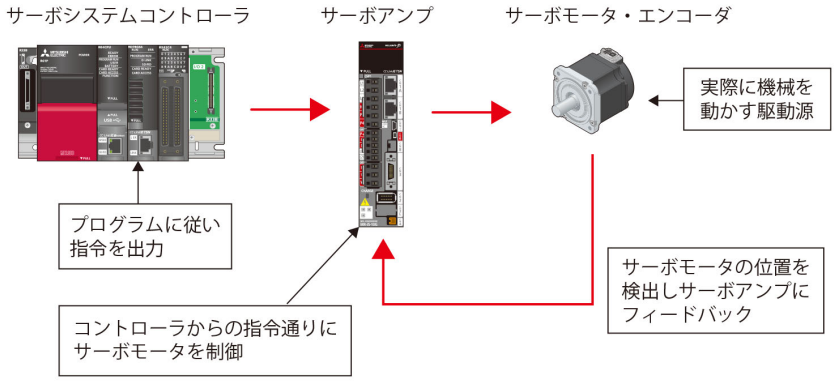
本書は，三菱電機モーションユニットRD78G(H)を初めて使用される方が，サーボシステムを構築するために必要なハードウェア，ソフトウェアについて学習することを目的としたガイダンスです。

## 1.1　サーボシステムとは

サーボシステムとは，「物体の位置，方位，姿勢などを制御量として，目標の任意の変化に追従されるように構成された制御系」のことです。

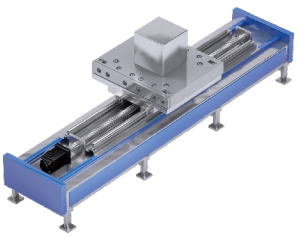
三菱電機サーボシステムは，下図に示すように，コントローラとなるシーケンサとサーボシステムコントローラから位置指令や速度指令をサーボアンプに伝達し，サーボモータを制御するシステムのことを指します。

サーボモータには位置や速度を検出するエンコーダが取り付けられており，それらの情報をフィードバックすることで，位置指令値や速度指令値と実際の位置や速度との偏差が小さくなるように制御します。

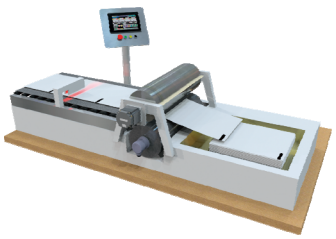



## 1.2　サーボシステムの用途例

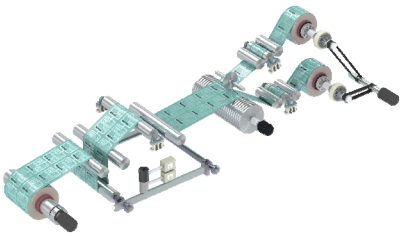
サーボシステムは，高い位置決め精度を必要とする機械，包装機や印刷機などの高度な同期を必要とする機械，また，巻取り／巻出し装置などの張力制御(トルク制御)や速度制御を必要とする機械など，幅広い用途で使用されています。



［位置制御が必要な装置］



［同期制御が必要な装置］



［トルク制御が必要な装置］

## 1.3　サーボシステムコントローラの役割

サーボシステムの指令を出力する役目のサーボシステムコントローラは，サーボアンプに対して位置指令や速度指令を出力するほか，複数の軸を使用した補間制御や同期制御を実行するための指令の生成を行ないます。

シーケンサで管理するセンサなどの値によって，位置指令や速度指令を変化させることもサーボシステムコントローラの役割です。

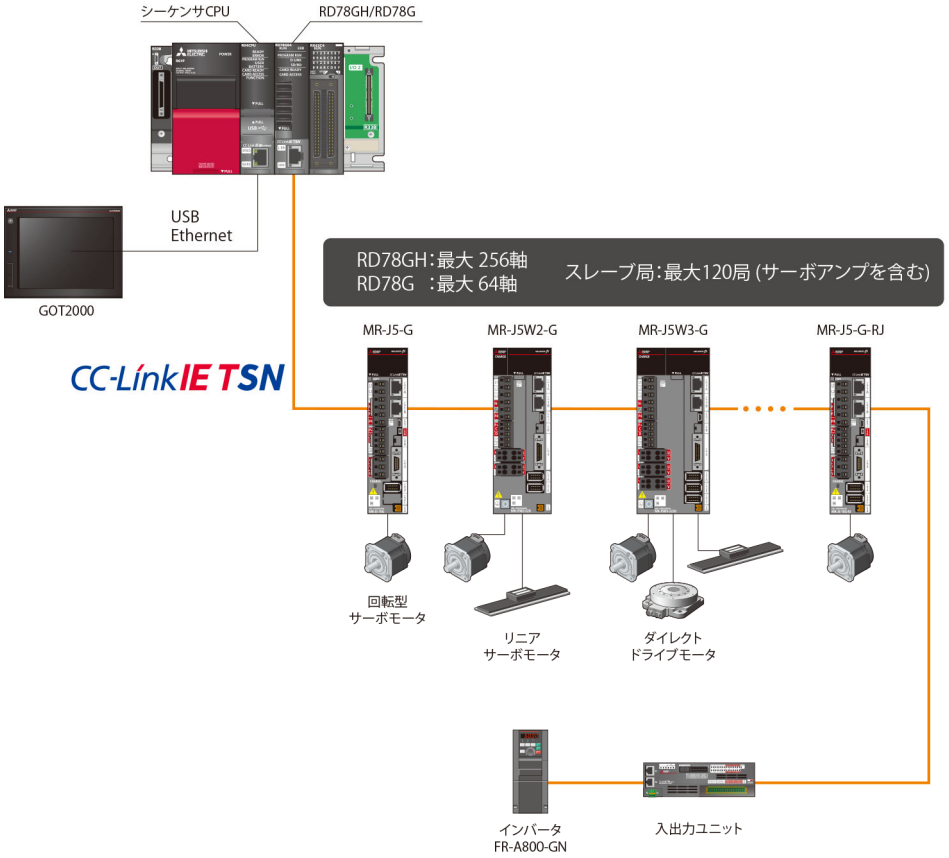
## 1.4　MELSEC iQ-RシリーズモーションユニットRD78G(H)の特長

MELSEC iQ-RシリーズモーションユニットRD78G(H)は，CC-Link IE TSNに対応したサーボシステムコントローラです。モーション制御に必要なリアルタイム性を保証した制御を実施しながら，ITシステムとの情報通信が混在可能なネットワークであるCC-Link IE TSNに対応することで，サーボアンプだけでなく，I/O，高速カウンタユニットなど様々な機器を自由に接続できます。

多様な機器を用いたフレキシブルなシステム構築が可能です。

RD78G(H)の主な特長を以下に示します。

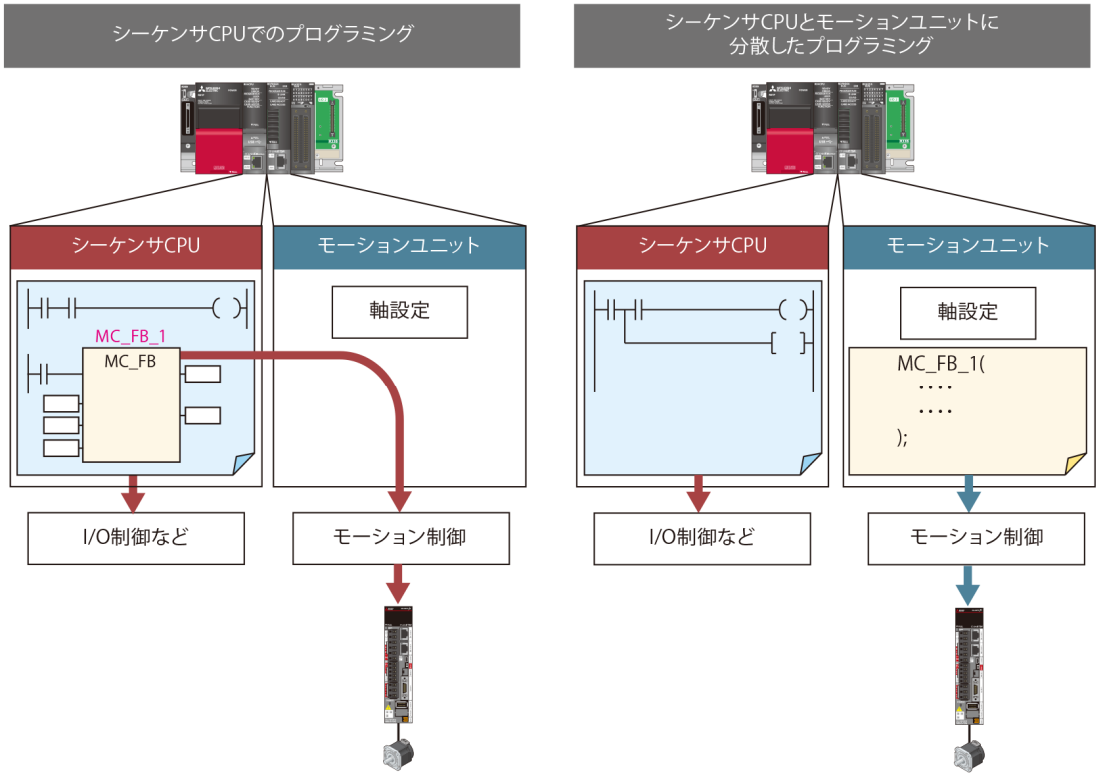
・マルチコアプロセッサを搭載し，高速処理を実現
・最大256軸のサーボモータ制御が可能
・PLCopen® Motion Control FBを使用して，位置決めなどのプログラミングが可能



## 1.5　モーションユニットのプログラミング

モーションユニットRD78G(H)のプログラムは，シーケンサCPUでプログラミングする方法と，シーケンサCPUとモーションユニットに分散してプログラミングする方法が選択できます。

シーケンサCPUは，IEC 61131-3で定められているラダー，FBD，SFC，ST言語でプログラミングします。また，モーションユニットは，ST言語でプログラミングします。




### (1) シーケンサCPUでのプログラミング

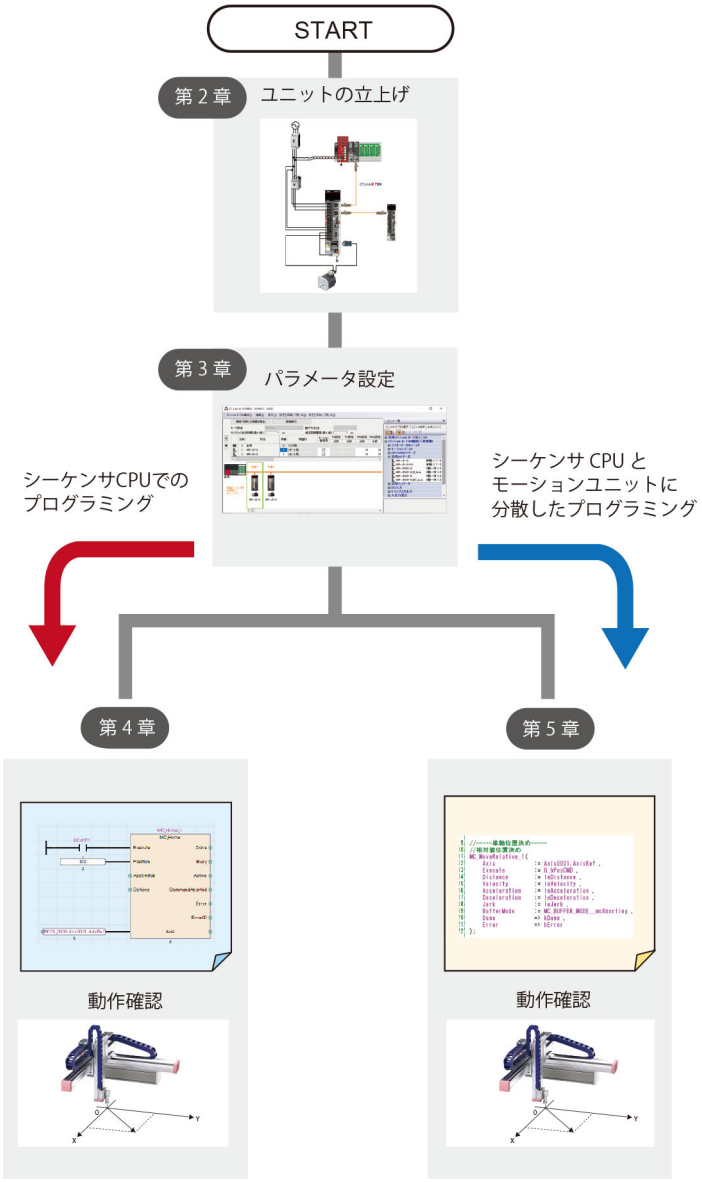
・メリット　　：1つのCPUだけでプログラミングでき，プログラムの管理が容易になります。
・デメリット　：1つのCPUで処理するため，スキャンタイムは長くなります。

### (2) シーケンサCPUとモーションユニットに分散したプログラミング

・メリット　　：プログラムを分散させることで，モーション制御が他のシーケンスプログラムの処理時間に影響せず，スキャンタイムが短縮できます。
・デメリット　：シーケンサCPUのプログラムとモーションユニットのプログラムの両方を管理する必要があります。

## 1.6　本書の構成

本書では，2軸のXYテーブルを想定して，立ち上げからデバッグまでの手順を順に記載しています。
プログラミングに関しては，シーケンサCPUでプログラミングする方法と，シーケンサCPUとモーションユニットに分散してプログラミングする方法に分けて記載しています。




**システムの立ち上げ**
- 機器の準備
- ファームウェア／ソフトウェアの準備
- プロファイルの登録
- モーションユニットFBライブラリの登録
- ユニットの装着
- 配線，およびケーブルの接続
- ロータリスイッチ設定

**パラメータの設定**
- プロジェクトの作成
- ネットワーク構成
- 各種パラメータの設定
- モーション制御設定
- 軸グループの設定
- ラベル
- 公開ラベル

**プログラミング**
- JOG，原点復帰
- 位置決め
- 直線補間制御
- 同期制御

**動作確認**
- 軸モニタ
- プログラムのウォッチ
- プログラムモニタ
- イベント履歴

## 1.7　関連マニュアル

### (1) モーションユニット

| 名　称 | 資料番号 |
|---|---|
| MELSEC iQ-R モーションユニット ユーザーズマニュアル(スタートアップ編) | IB(名)-0300405 |
| MELSEC iQ-R モーションユニット ユーザーズマニュアル(応用編) | IB(名)-0300410 |
| MELSEC iQ-R モーションユニット ユーザーズマニュアル(ネットワーク編) | IB(名)-0300425 |
| MELSEC iQ-R プログラミングマニュアル(モーション制御FB編) | IB(名)-0300532 |
| MELSEC iQ-R プログラミングマニュアル(モーションユニット用命令/汎用FUN/汎用FB編) | IB(名)-0300430 |

### (2) シーケンサ

| 名　称 | 資料番号 |
|---|---|
| MELSEC iQ-R CPUユニット ユーザーズマニュアル(スタートアップ編) | SH(名)-081223 |
| MELSEC iQ-R CPUユニット ユーザーズマニュアル(応用編) | SH(名)-081224 |
| MELSEC iQ-R Ethernet/CC-Link IE ユーザーズマニュアル(スタートアップ編) | SH(名)-081252 |
| MELSEC iQ-R Ethernet ユーザーズマニュアル(応用編) | SH(名)-081253 |
| MELSEC iQ-R プログラミングマニュアル(プログラム設計編) | SH(名)-081225 |
| MELSEC iQ-R ストラクチャードテキスト(ST)プログラミングガイドブック | SH(名)-081445 |
| MELSEC iQ-R プログラミングマニュアル(CPUユニット用命令/汎用FUN/汎用FB編) | SH(名)-081226 |

### (3) サーボアンプ

| 名　称 | 資料番号 |
|---|---|
| MR-J5 ユーザーズマニュアル(ハードウェア編) | SH(名)-030297 |
| MR-J5-G/MR-J5W-G ユーザーズマニュアル(導入編) | SH(名)-030293 |
| MR-J5-G/MR-J5W-G ユーザーズマニュアル(パラメータ編) | SH(名)-030307 |
| MR-J5 ユーザーズマニュアル(機能編) | SH(名)-030299 |
| MR-J5 ユーザーズマニュアル(通信機能編) | SH(名)-030301 |
| MR-J5 ユーザーズマニュアル(トラブルシューティング編) | SH(名)-030311 |
| MR-J5 ユーザーズマニュアル(オブジェクトディクショナリ編) | SH(名)-030303 |

# メ　モ

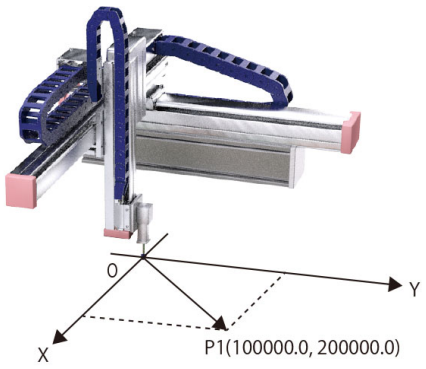
# 第２章　システムの立ち上げ

## 2.1　装置の概要

本書では，ボールネジを使用した2軸システムの装置を例に説明します。



### (1) 装置



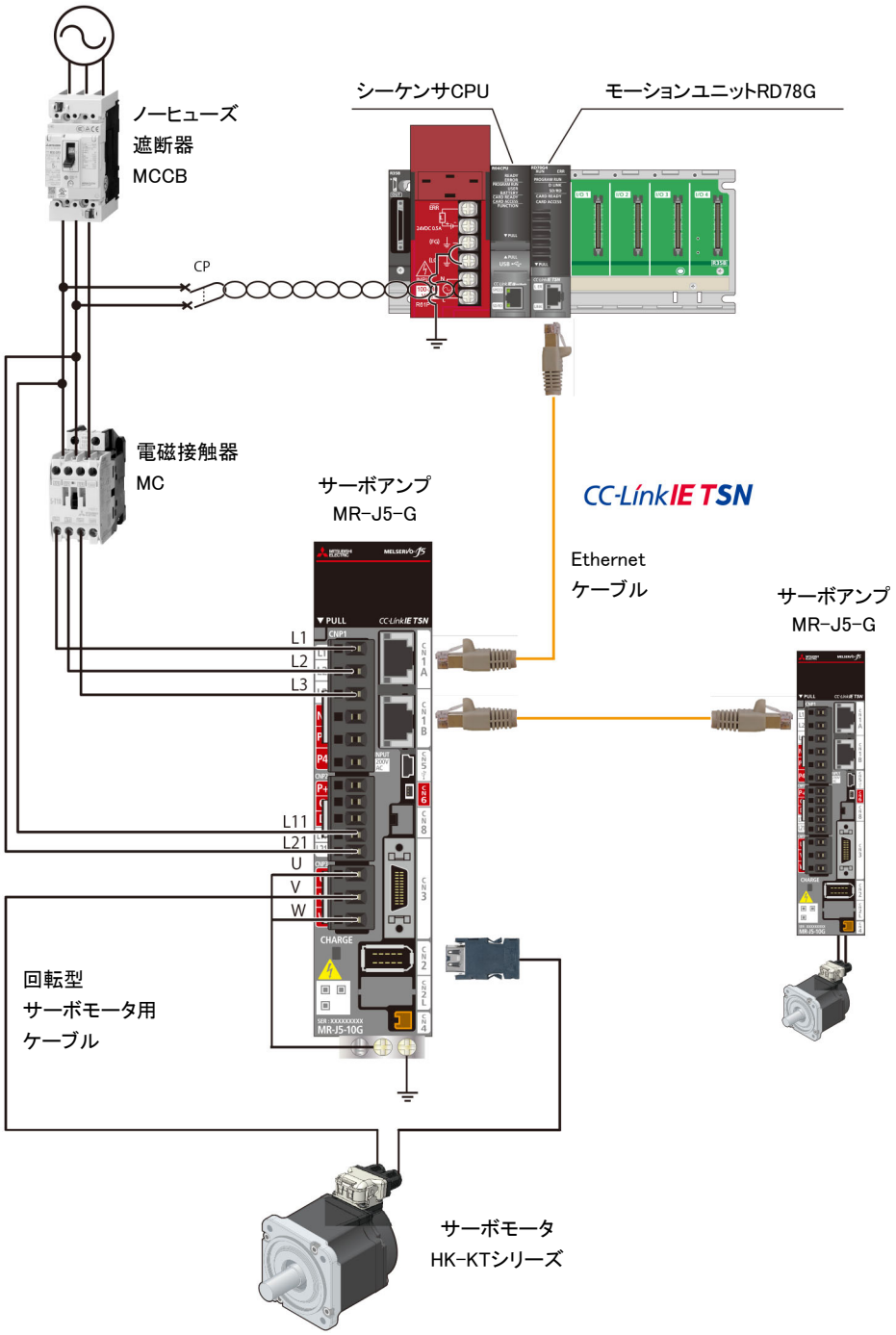

[単位：μm]

### (2) 仕様

ボールネジリード(PB)　：10.0［mm］
減速比(NL/NM)　　：1/2　（負荷側[NL]／モータ側[NM]）
　サーボモータが2回転すると，負荷側のボールネジが1回転します。
エンコーダ分解能　　：26ビット　（67108864[pulse]）

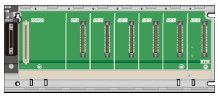
## 2.2　システム構成

モーションユニット(RD78G)，サーボアンプ(MR-J5-G)，およびサーボモータ(HK-KTシリーズ)を使用したシステム構成例を以下に示します。



## 2.3 機器の準備

本書では，以下の機器を使用したプロジェクトに基づいて説明しています。
お客様の装置にあわせて，ユニット，ケーブル，およびソフトウェアなどを準備してください。

| モーションユニット | エンジニアリング環境 | |
|---|---|---|
| RD78G4 | MELSOFT GX Works3 | |
| **サーボアンプ** | **サーボモータ** | |
| MR-J5-10G | HK-KT13W | |
| **基本ベースユニット** | **電源ユニット** | **CPUユニット** |
| R35B  | R61P | R04CPU |
| **回転型サーボモータ用ケーブル** | **Ethernetケーブル** | **USBケーブル** |
| MR-AEPB2CBL2M-A1-L | カテゴリ5e以上<br>（二重シールド付・STP）<br>ストレートケーブル | MR-J3USBCBL3M |
| **ノーヒューズ遮断器（MCCB）** | **電磁接触器（MC）** | **サーキットプロテクタ（CP）** |
| | | |

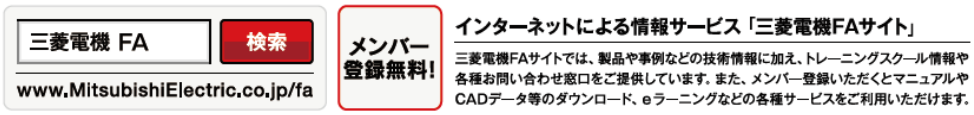
## 2.4　ファームウェア／ソフトウェアのインストール

### 2.4.1　ファームウェア／ソフトウェアの準備

本書の内容は，下表のファームウェア，ソフトウェア，およびエンジニアリング環境の仕様に基づいています。

使用するファームウェア，ソフトウェアのバージョンにより，画面が異なる場合がありますので，使用時には最新版を確認してください。

三菱電機FAサイトから最新版のアップデート専用のファイルをダウンロードできます。




［ファームウェア／ソフトウェア］

| ユニット | 内　容 | ユニット形名 | バージョン |
|---|---|---|---|
| シーケンサCPUユニット | ファームウェアアップデート情報ファイル | R00CPU/R01CPU/R02CPU | 14以降 |
| | | R04CPU/R08CPU/R16CPU/R32CPU/R120CPU*1 | 46以降 |
| | | R□ENCPU*1 | 46以降 |
| モーションユニット | ネットワークブートソフトウェア | RD78G/RD78GH | 04以降 |
| | ブートソフトウェア | | 04以降 |
| | 基本システムソフトウェア | | 09以降 |
| サーボアンプ | CC-Link IE TSN対応サーボアンプMR-J5-G用ファームウェア | | A7以降 |

＊1：Ver.23以降のユニットにインストール可能

［エンジニアリング環境］

| 項　目 | 内　容 | バージョン |
|---|---|---|
| MELSOFT GX Works3 | シーケンサエンジニアリングソフトウェア | 1.065T以降 |
| モーション制御設定 | モーションユニットの設定やモニタを行うためのソフトウェア | 1.011M以降 |
| MELSOFT MR Configurator2 | サーボアンプのセットアップ用ソフトウェア | 1.105K以降 |
| MELSOFT GX LogViewer | モーションユニット，CPUデータロギング機能などで収集した大容量のデータを，わかりやすい操作で表示・分析するツール | 1.106K以降 |

［プロファイル］

| 項　目 | 内　容 | バージョン |
|---|---|---|
| MR-J5-G用プロファイル | MELSERVO-J5シリーズ CC-Link IE TSN対応サーボアンプのプロファイル | 02以降 |

［FBライブラリ］　※シーケンサCPUでプログラミングする場合に使用

| 項　目 | 内　容 | バージョン |
|---|---|---|
| MELSEC iQ-R モーションユニットFBライブラリ | MELSEC iQ-Rシリーズのモーションユニットを使用したシステムで使用するためのFBライブラリ | 01B以降 |

### 2.4.2 MELSOFT GX Works3のインストール

(1) インストール前の確認

・「管理者」または「Administrator」のユーザとしてパソコンにログインしてください。
・インストール前に全てのアプリケーションを終了してください。他のアプリケーションが動作している状態でインストールすると，正常に動作しない場合があります。

(2) インストール手順

①MELSOFT GX Works3の製品DVD-ROMをDVD-ROMドライブに挿入し，Disk1フォルダの“setup.exe”をダブルクリックします。

②画面の指示に従って，必要事項を選択，または入力します。*1

＊1：プロダクトIDは，本製品に同梱されている「ライセンス許諾書」に記載されています。
12桁の数字を3桁-9桁で入力してください。

**[ポイント]**
**モーション制御設定，およびGX LogViewerは，MELSOFT GX Works3のDVDからインストール可能ですが，MELSOFT GX Works3のアップデートモジュールには含まれません。**
**アップデートモジュールを用いてMELSOFT GX Works3をアップデートする場合は，個別にモーション制御設定，GX LogViewerもアップデートしてください。**

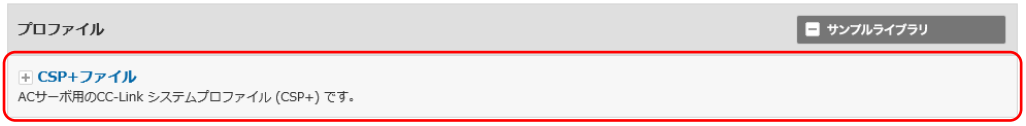
### 2.4.3　プロファイルの登録

サーボアンプ，リモートI/Oユニット等，使用する機器のプロファイルを登録する必要があります。（サーボアンプMR-J5-G／MR-J5W□-Gは，MELSOFT GX Works3をインストールすることにより，自動的に登録されます。）

#### (1) プロファイルのダウンロード

三菱電機FAサイトから最新版のアップデート専用のファイルをダウンロードできます。

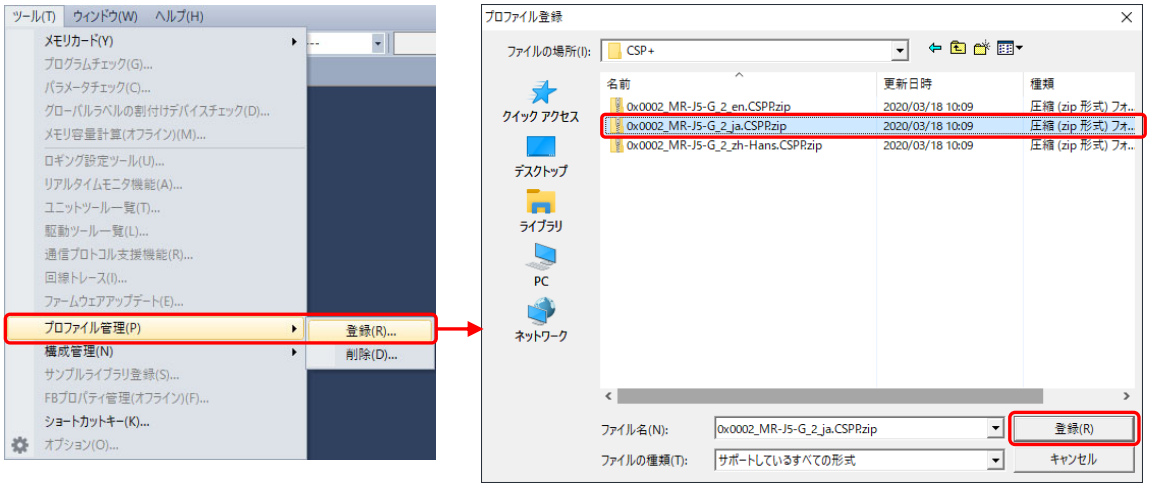
-プロファイル、ライブラリ




#### (2) プロファイルのインストール

①保存したダウンロードファイルを展開した後，任意のCSP＋ファイル(zipファイル)を登録してください。

②MELSOFT GX Works3を起動して，[ツール]⇒[プロファイル管理]⇒[登録]を選択し，“プロファイル登録”画面を表示します。
解凍したプロファイルから使用する言語のファイルを選択し，[登録]ボタンをクリックします。



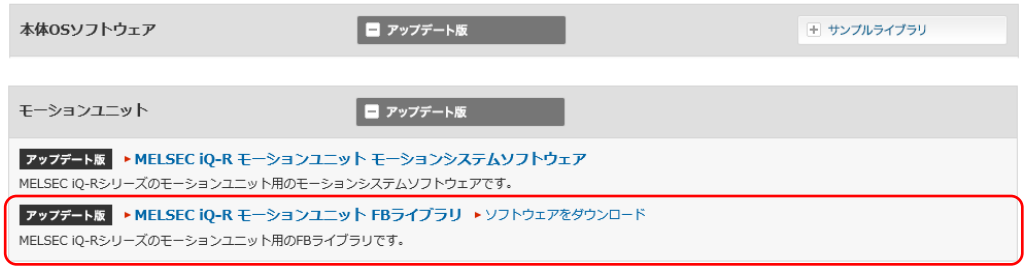
## 2.4.4 MELSEC iQ-Rシリーズ モーションユニットFBライブラリ

シーケンサCPUでプログラミングする場合は，以下の手順でFBライブラリを登録してください。

### (1) モーションユニットFBのダウンロード

三菱電機FAサイトから最新版のアップデート専用のファイルをダウンロードできます。

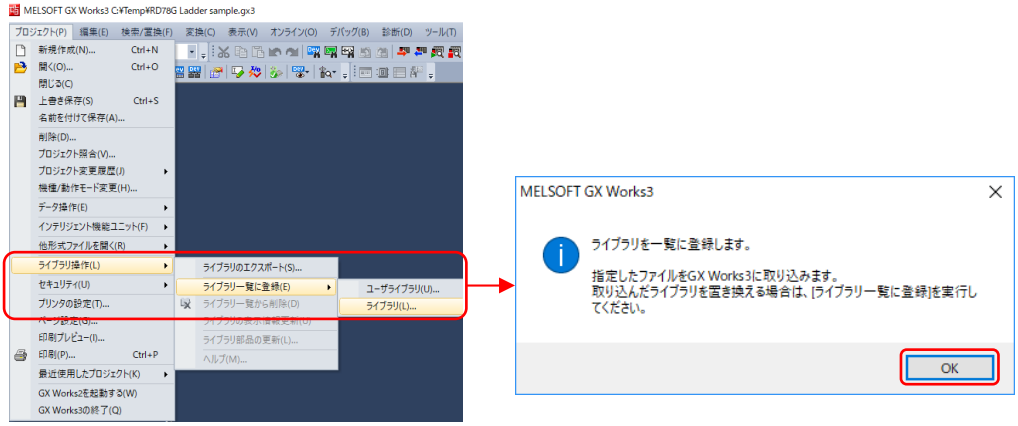
-ファームウェア




### (2) モーションユニットFBライブラリの登録手順

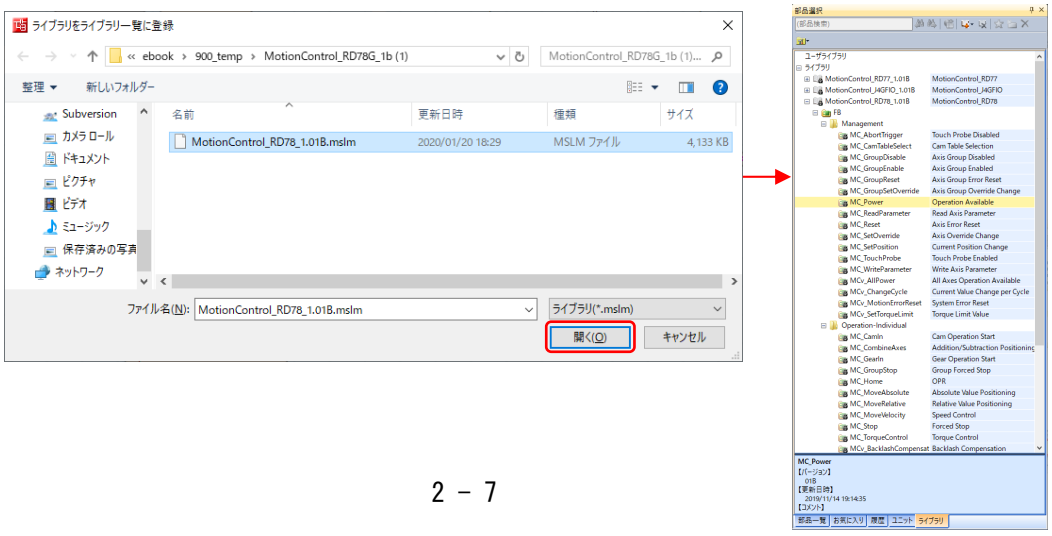
①MELSOFT GX Works3を起動します。

②任意のプロジェクトを開き，[プロジェクト]⇒[ライブラリ操作]⇒[ライブラリ一覧に登録]⇒[ライブラリ]を選択して，確認画面で[OK]をクリックします。




③“ライブラリをライブラリ一覧に登録”画面で[MotionControl_****.mslm]を選択して，[開く]ボタンをクリックします。

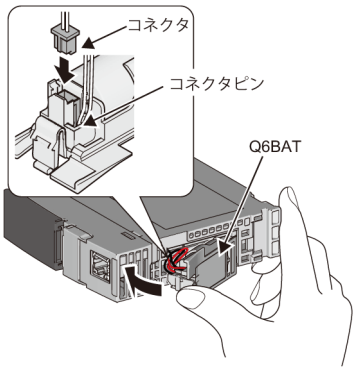
部品選択ウィンドウの[ライブラリ]タブにモーションユニットFBが登録されていることを確認してください。



## 2.5 ユニットの装着

### (1) バッテリの装着

CPUユニットのQ6BATは，コネクタをはずした状態で出荷されます。
下記の手順でコネクタを接続してください。




①CPUユニット底部のカバーを開けます。
②Q6BATが正しく装着されているか確認します。
③Q6BATに取り付けられているコネクタを，カバー側のコネクタピンに向きを確認して差し込みます。コネクタは奥までしっかりと押し込んでください。
④CPUユニット底部のカバーを閉じます。

### (2) ユニットの装着

各ユニットをベースユニットに装着してください。

## 2.6　配線，およびケーブルの接続

モーションユニットとサーボアンプの配線，および各ケーブルの接続例を示します。
本書では，サーボアンプMR-J5-10Gを使用した場合の電線サイズを記載しています。
サーボアンプの容量が異なる場合は，各ユーザーズマニュアルを参照してください。

### (1) 電源ユニットの配線

電源ユニットに電源線と接地線を配線する例です。
雷サージなどのノイズが多い場合は，絶縁トランスを接続してください。

| 項　目 | 適合電線サイズ |
|---|---|
| 電源線 | AWG18～AWG14 |
| 接地線 | AWG18～AWG14 |

### (2) サーボアンプの電源とサーボモータ動力線の配線

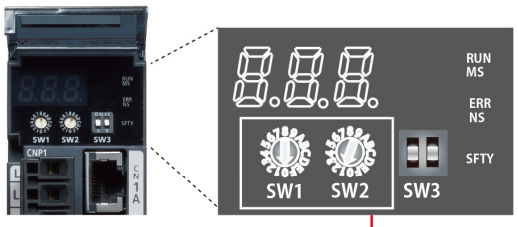
サーボアンプに制御回路電源（L11，L21），主回路電源（L1，L2，L3）とサーボモータ電源ケーブルを配線してください。

| 項　目 | 適合電線サイズ |
|---|---|
| 制御回路電源（L11，L21） | 1.25～2mm$^2$（AWG16～AWG14） |
| 主回路電源（L1，L2，L3） | 2mm$^2$（AWG14） |
| 接地線 | AWG18～AWG14 |

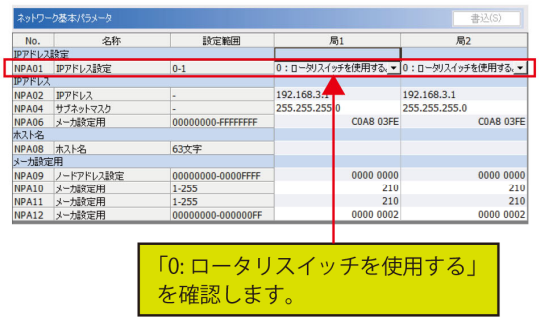
### (3) サーボアンプのロータリスイッチ設定

サーボアンプのロータリスイッチ“SW1，SW2”は，“CC-Link IE TSN構成”ウィンドウのIPアドレス第4オクテットに対応しています。

サーボアンプのロータリスイッチ




サーボアンプのネットワーク基本パラメータ画面




「0: ロータリスイッチを使用する」
を確認します。

192.168.3.1

第４オクテットを
ロータリスイッチで設定
(01 ～ FE)

### (4) 各種ケーブルの接続

Ethernetケーブル，および回転型サーボモータ用ケーブルを接続してください。
また，パソコンとCPUユニット間にUSBケーブルを接続してください。

### (5) システムの電源投入

①電源ユニットの配線を確認してください。
②シーケンサCPUユニットがSTOP状態になっていることを確認してください。
③電源をONします。

ユニットのLEDが点灯していることを確認してください。

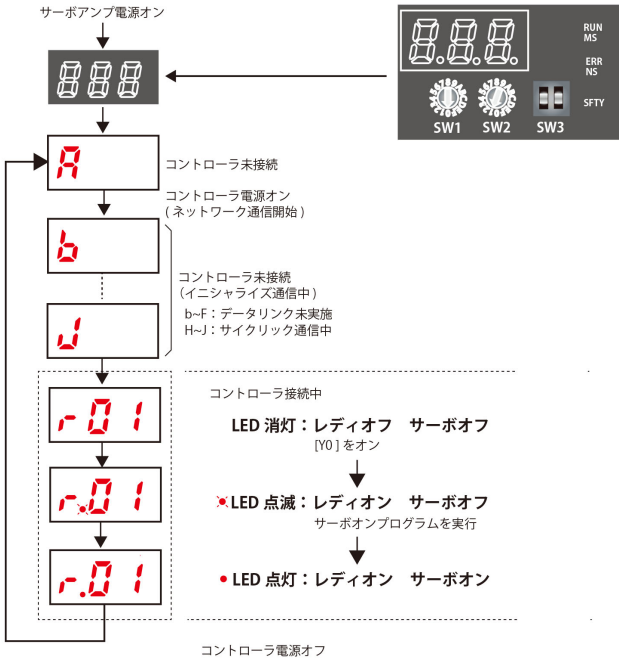
・電源ユニット：LED(緑色)点灯
・シーケンサCPUユニット：READY LED(緑色)点灯
パラメータ，プログラムがシーケンサCPUユニットに書き込まれていないと，シーケンサCPUユニットのERROR LEDが赤点滅しますが問題ありません。
パラメータ，プログラムを書き込み後，電源をOFF→ONすると，ERROR LEDは消灯します。
・モーションユニット：RUN LED(緑色)点灯
パラメータ，プログラムがモーションユニットに書き込まれていないと，ERR LEDが赤点滅しますが問題ありません。
パラメータ，プログラムを書き込み後，電源をOFF→ONし，スレーブ局との通信が正常に行なわれると，ERR LEDは消灯します。

### (6) サーボアンプの電源投入

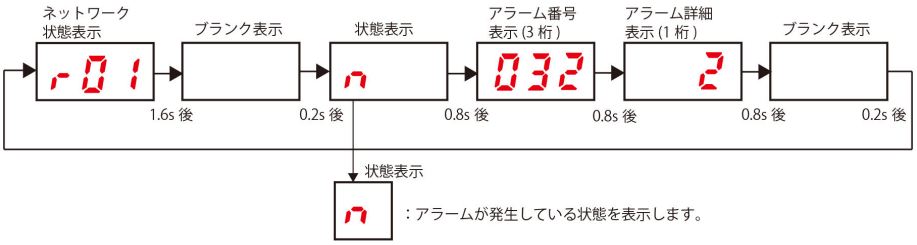
サーボアンプの配線を確認し，サーボアンプの制御回路電源をONします。

サーボアンプの表示部で，モーションユニットとの交信状態を確認することができます。

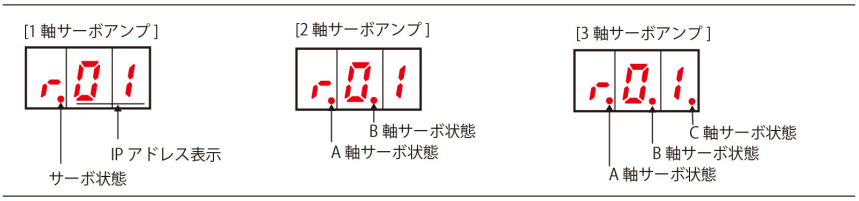
(a) サーボアンプの7セグメントLED表示（通常時）



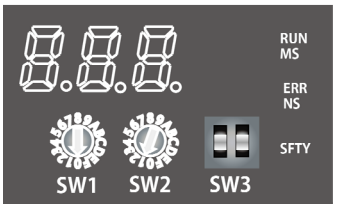

(b) 1軸サーボアンプのアラーム発生時




(c) ネットワーク接続中の7セグメントLEDの表示



(7) 表示部のLED

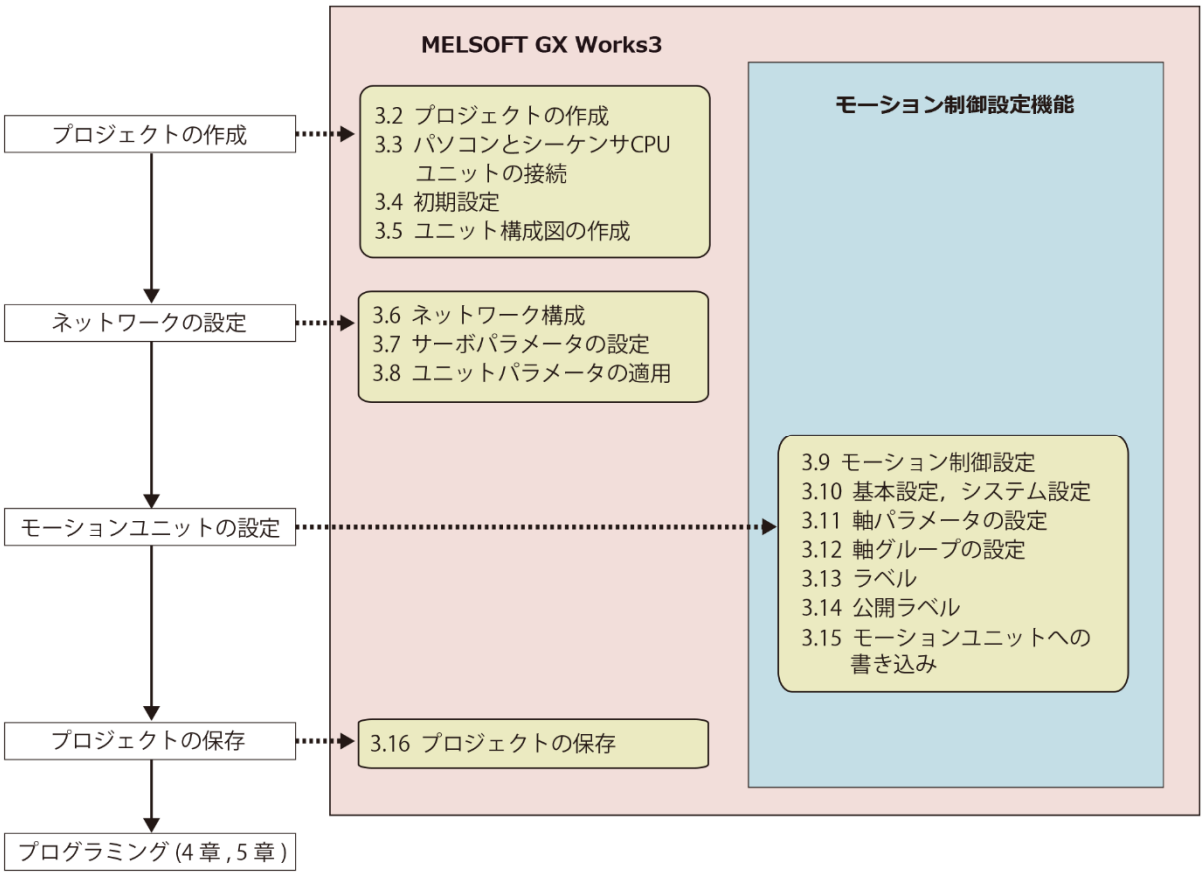



| 名　称 | LEDの色 | 内　容 |
|---|---|---|
| RUN<br>MS | 緑色 | 消灯：アラームが発生していることを示します。<br>点灯：電源が投入されていることを示します。 |
| ERR<br>NS | 赤色 | 消灯：アラームや警告が発生していないことを示します。<br>点滅：警告が発生していることを示します。<br>点灯：アラームが発生していることを示します。 |
| SFTY | 緑色 | 消灯：機能安全が作動できないことを示します。<br>点灯：機能安全が作動可能な状態を示します。 |

# 第３章　パラメータの設定

## 3.1　パラメータの設定手順

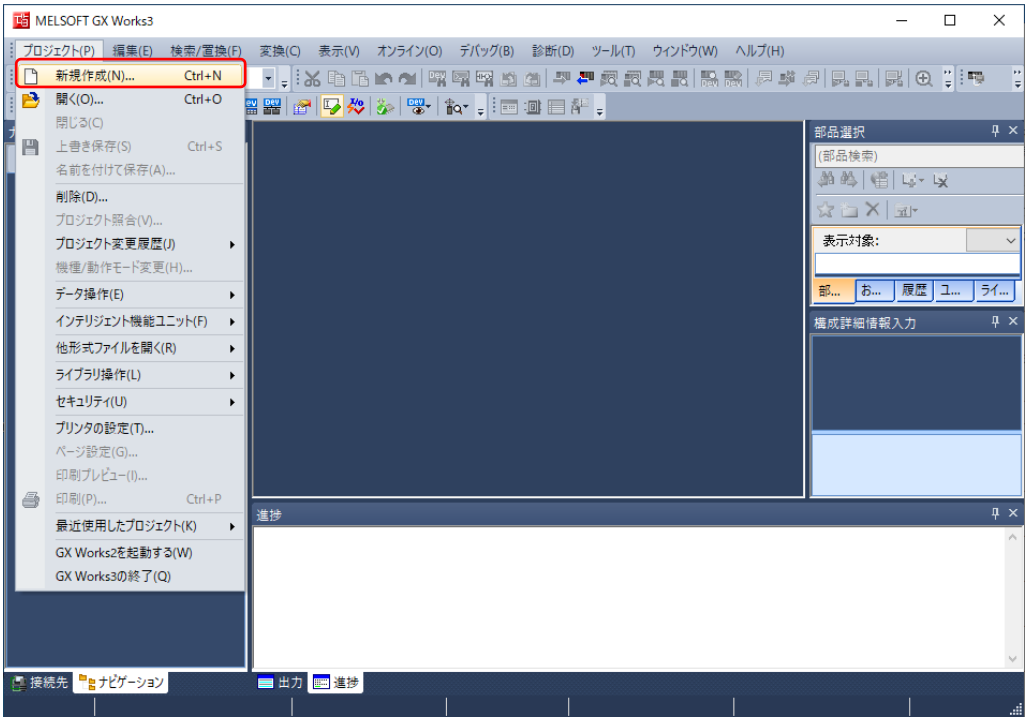
パラメータ設定手順を下図に示します。



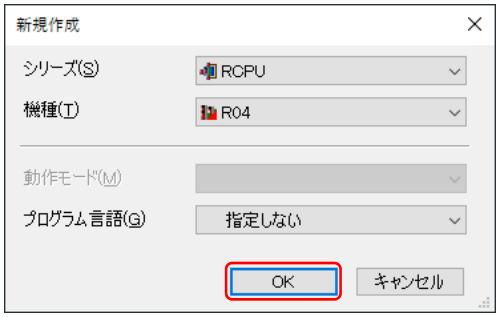
## 3.2　プロジェクトの作成

プロジェクトを新規作成します。

①MELSOFT GX Works3を起動して，[プロジェクト]⇒[新規作成]を選択します。

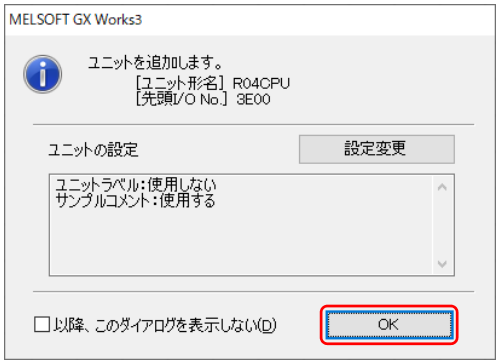



②シリーズ，機種，プログラム言語を選択し，[OK]ボタンをクリックします。




[設定例]
シリーズ　　：RCPU
機種　　：使用するシーケンサCPUを選択
プログラム言語　：使用する言語を選択

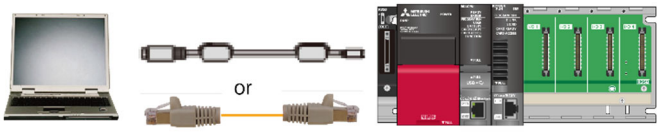
③表示されるメッセージを確認し，[OK]ボタンをクリックします。



## 3.3　パソコンとシーケンサCPUユニットの接続

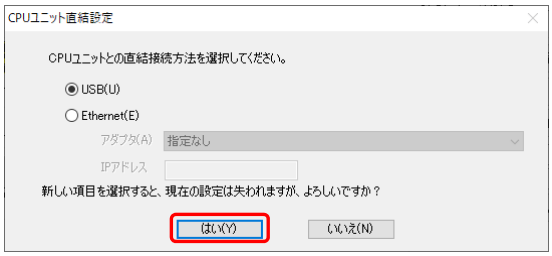
パソコンとシーケンサCPUユニット間の接続を確認します。

①パソコンとシーケンサCPUユニットを，USB，またはEthernetケーブルで接続します。




②[オンライン]⇒[現在の接続先]を選択して，“接続先指定”画面を表示します。
[CPUユニット直結設定]ボタンをクリックして，パソコンの接続先を設定します。

③使用する接続方法を選択し，[はい]ボタンをクリックして，“接続先指定”画面に戻ります。




“Ethernet”を選択する場合は，アダプタを指定する必要があります。

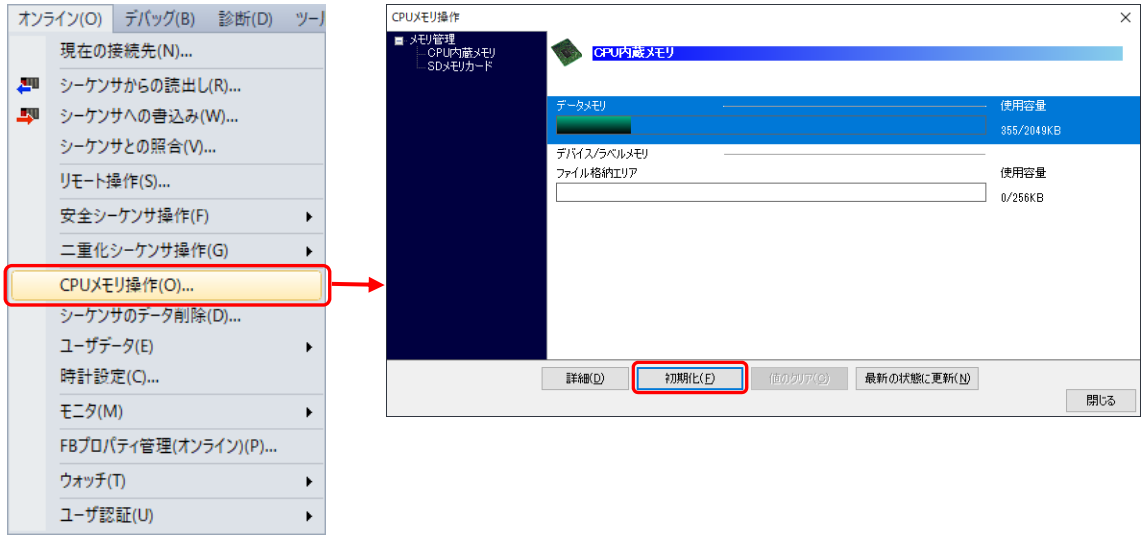
④“接続先指定”画面の[通信テスト]ボタンをクリックして，パソコンとの通信テストを行ないます。

## 3.4　初期設定

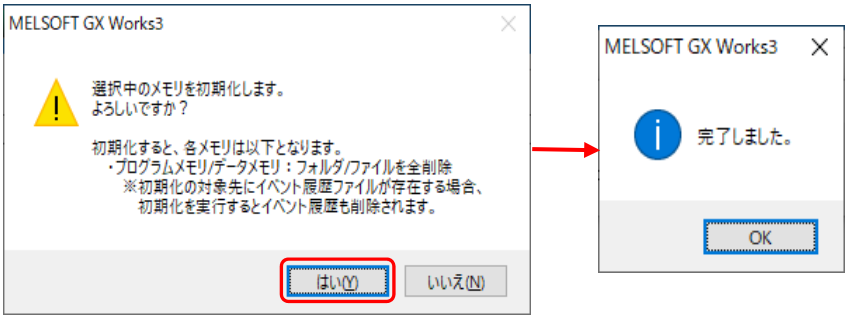
### 3.4.1　シーケンサCPUユニットの初期化

シーケンサCPUユニットは出荷時にバッテリを接続していないため，メモリのデータが不定となっています。メモリをクリアする必要があるため，シーケンサCPUユニットの初期化を行ないます。

①[オンライン]⇒[CPUメモリ操作]を選択します。“CPUメモリ操作”画面で「初期化」をクリックします。




②確認画面で，[はい]ボタンをクリックして初期化を始めます。

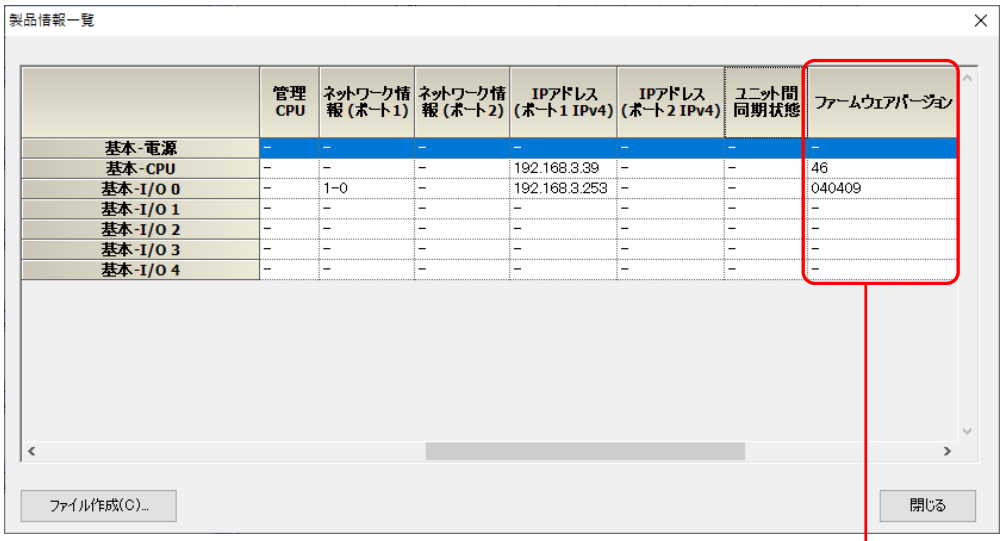


### 3.4.2　ファームウェアバージョンの確認

シーケンサCPUユニット，モーションユニットのファームウェアバージョンを確認します。

①[診断]⇒[システムモニタ] を選択して，“システムモニタ基本ベース(R35B)”画面を表示します。[製品情報一覧]ボタンをクリックして，“製品情報一覧”画面を表示します。

②ファームウェアバージョンを確認します。



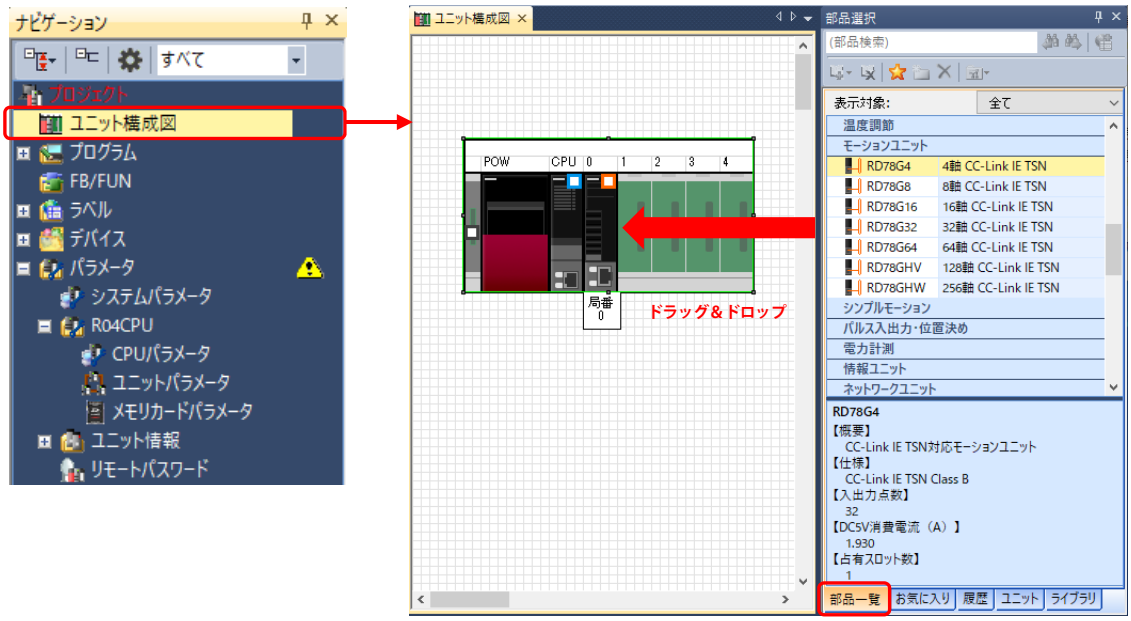

[モーションユニットバージョンの見方]

040409
- 基本システムソフトウェア
- ブートソフトウェア
- ネットワークブートソフトウェア

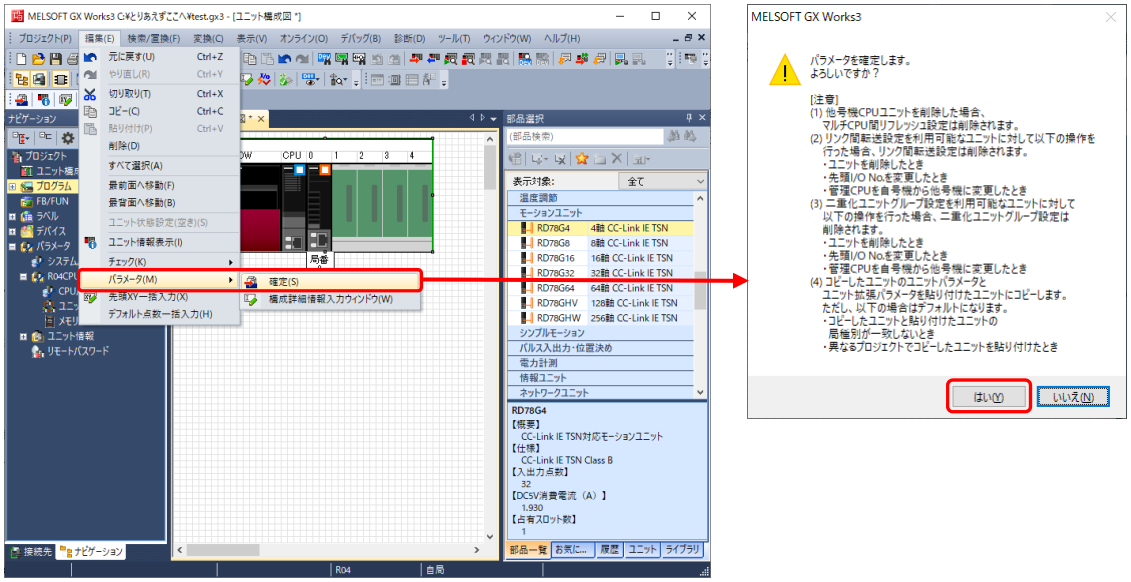
## 3.5　ユニット構成図の作成

ユニット構成図を作成します。

①ナビゲーションウィンドウの“ユニット構成図”をダブルクリックして，“ユニット構成図”ウィンドウを表示します。部品選択ウィンドウの[部品一覧]タブをクリックし，使用するユニット（基本ベースユニット，電源ユニット，シーケンサCPUユニット，モーションユニット）を選択し，ユニット構成図上にドラッグ＆ドロップします。




②[編集]⇒[パラメータ]⇒[確定] を選択して，確認画面で[はい]ボタンをクリックします。



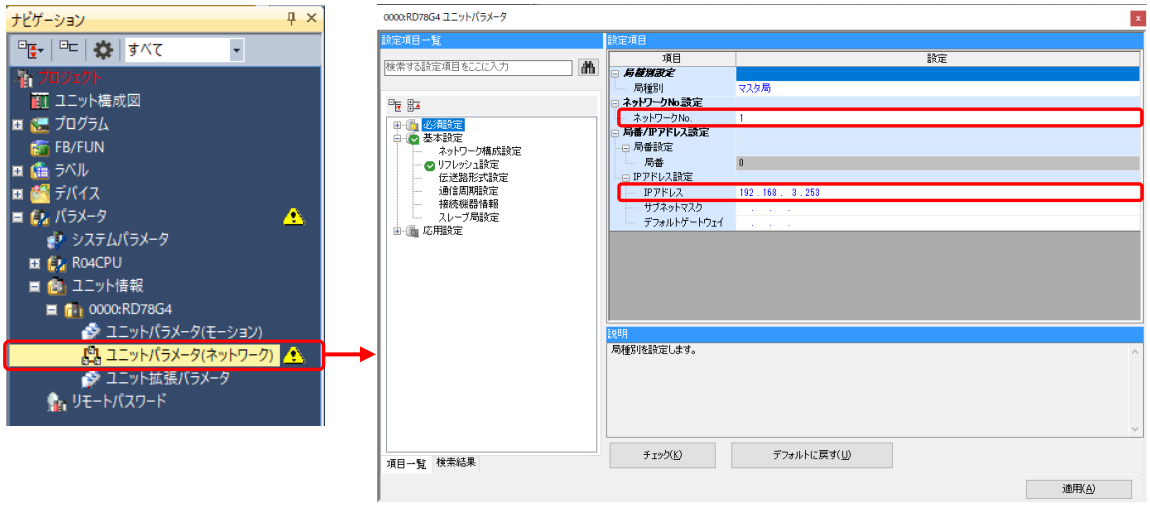
③確認画面で[はい]ボタンをクリックします。表示されるメッセージを確認し，[OK]ボタンをクリックして，各ユニットのパラメータ(初期値)を確定します。

## 3.6　ネットワーク構成

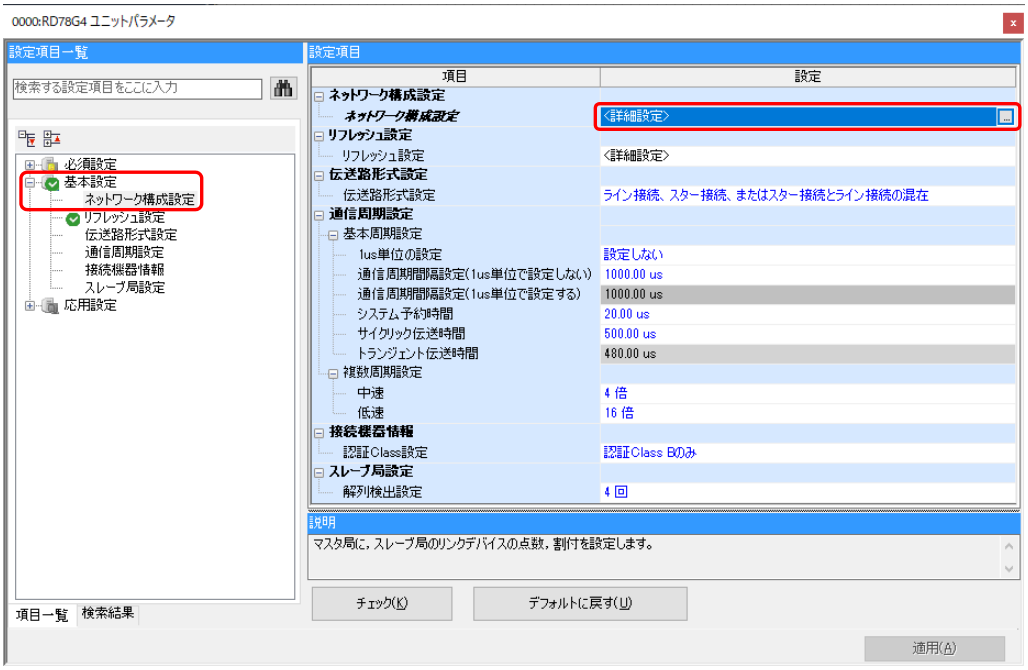
CC-Link IE TSNに接続するサーボアンプなどのスレーブ機器を設定します。

### 3.6.1　ネットワーク構成の設定

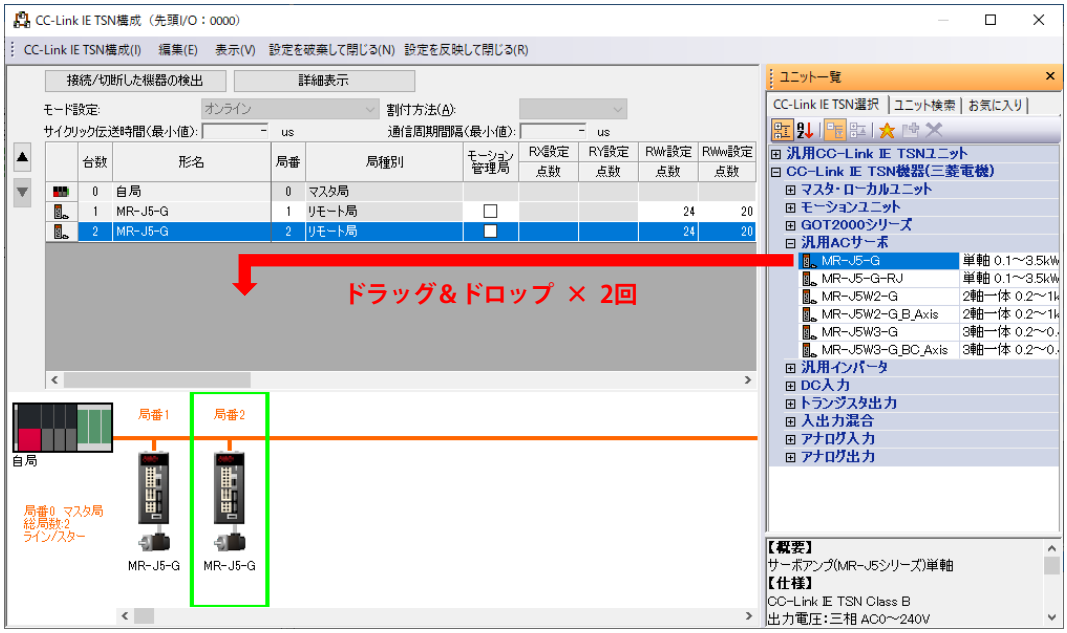
①ナビゲーションウィンドウで，使用するモーションユニットの“ユニットパラメータ(ネットワーク)”をダブルクリックします。パラメータエディタ(ユニットパラメータ)で“必須設定”⇒“ネットワークNo.”と“IPアドレス”を設定します。
本書では初期値から変更しません。(ネットワークNo. ：1，IPアドレス：192.168.3.253)




②パラメータエディタ(ユニットパラメータ)で“基本設定”⇒“ネットワーク構成設定”⇒“<詳細設定>”をダブルクリックして，“CC-Link IE TSN構成”ウィンドウを表示します。

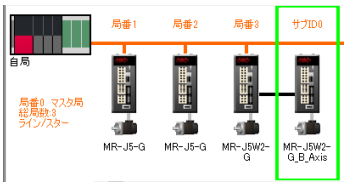


③ユニット一覧から使用する機器を選択し，ドラッグ＆ドロップします。
局番，入出力点数は，初期値が表示されます。
IPアドレスは，マスタ局の設定と局番により，ドロップした順番に自動で割り付けられます。
本書では，MR-J5-Gを2回ドラッグ＆ドロップし，2台のサーボアンプを登録してください。




※ユニット一覧に登録されていない機器を追加する場合は，プロファイルを登録してください。
(2.4.3項参照)

**[ポイント]**

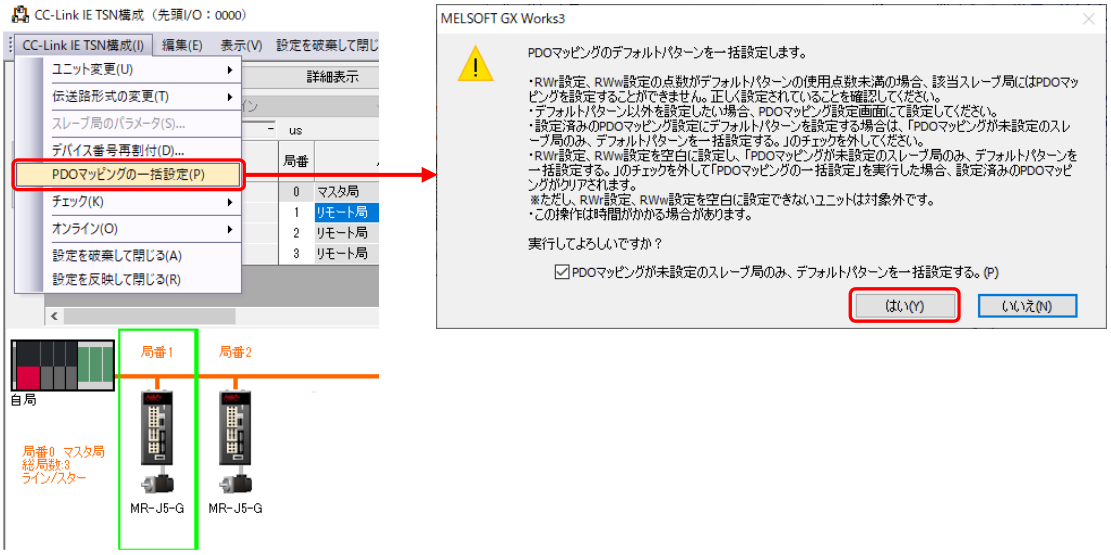



多軸一体サーボアンプを使用する場合は，サーボアンプ(MR-J5W□-G)をドラッグ＆ドロップすることにより，B軸以降もあわせて設定されます。

### 3.6.2 PDOマッピング

PDOマッピングのデフォルトデータを一括設定します。

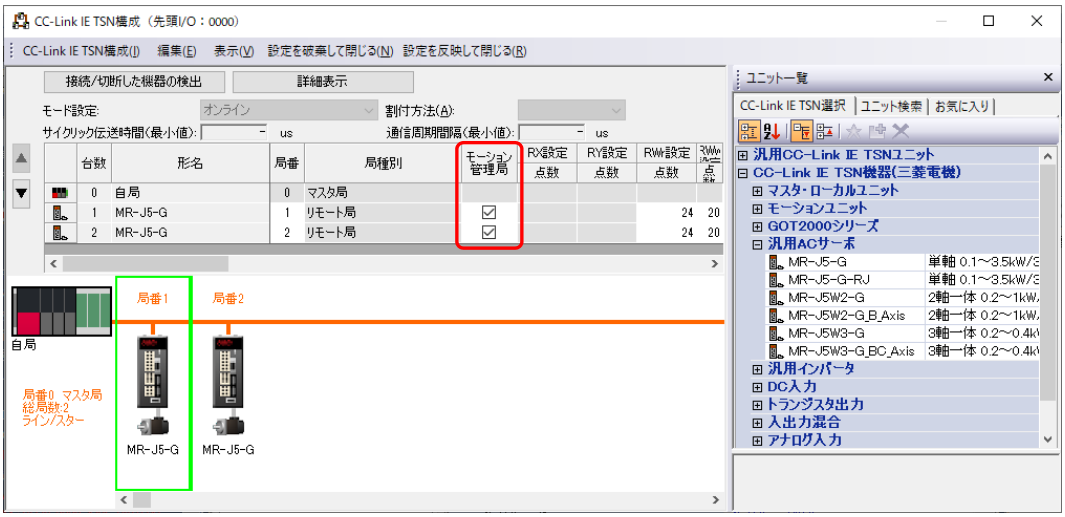
①[CC-Link IE TSN構成]⇒[PDOマッピングの一括設定]を選択して，確認画面を表示します。
[はい]ボタンをクリックして，PDOマッピングの一括設定を行ないます。




**[ポイント]**
PDOはProcess Data Objectの略です。
PDO通信は，従来のCC-Linkのサイクリック通信に相当します。
PDOマッピングとは，サイクリック通信（PDO通信）にて，コントローラとスレーブ間で授受するデータ（オブジェクト）をあらかじめマッピング（関連付け）することです。

### 3.6.3 モーション管理局の設定

モーションユニットで管理するユニット（サーボアンプ，入出力ユニットなど）の“モーション管理局”にチェックを入れます。



## 3.7　サーボパラメータの設定

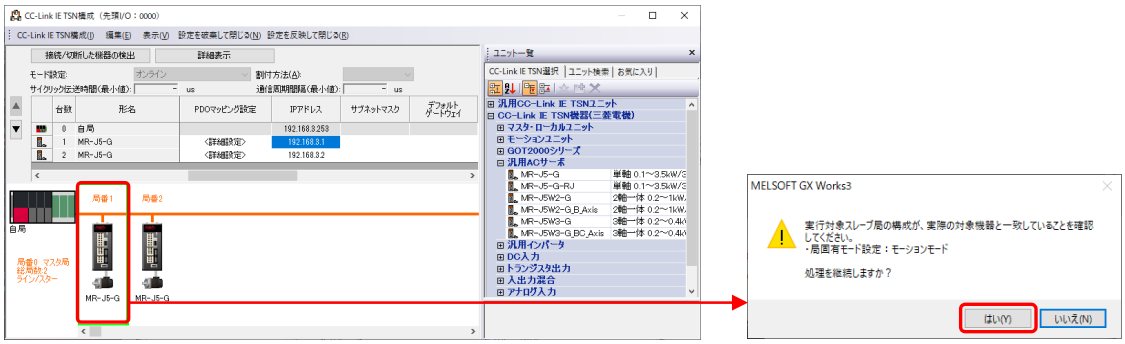
サーボアンプにパラメータを書き込む方法には，以下の2通りがあります。

| 書き込み方法 | メリット | デメリット |
|---|---|---|
| MELSOFT GX Works3 から<br>シーケンサ CPU に書き込む | ・パラメータの一括管理が可能<br>・機器の接続なしでパラメータ設定が可能 | ・初期通信に時間がかかる |
| MR Configurator2 から<br>サーボアンプに書き込む | ・電源 ON 時，パラメータ転送が不要なため，サーボアンプの立ち上がりが速い | ・サーボアンプ 1 台ごとにパラメータ書き込み作業が必要 |

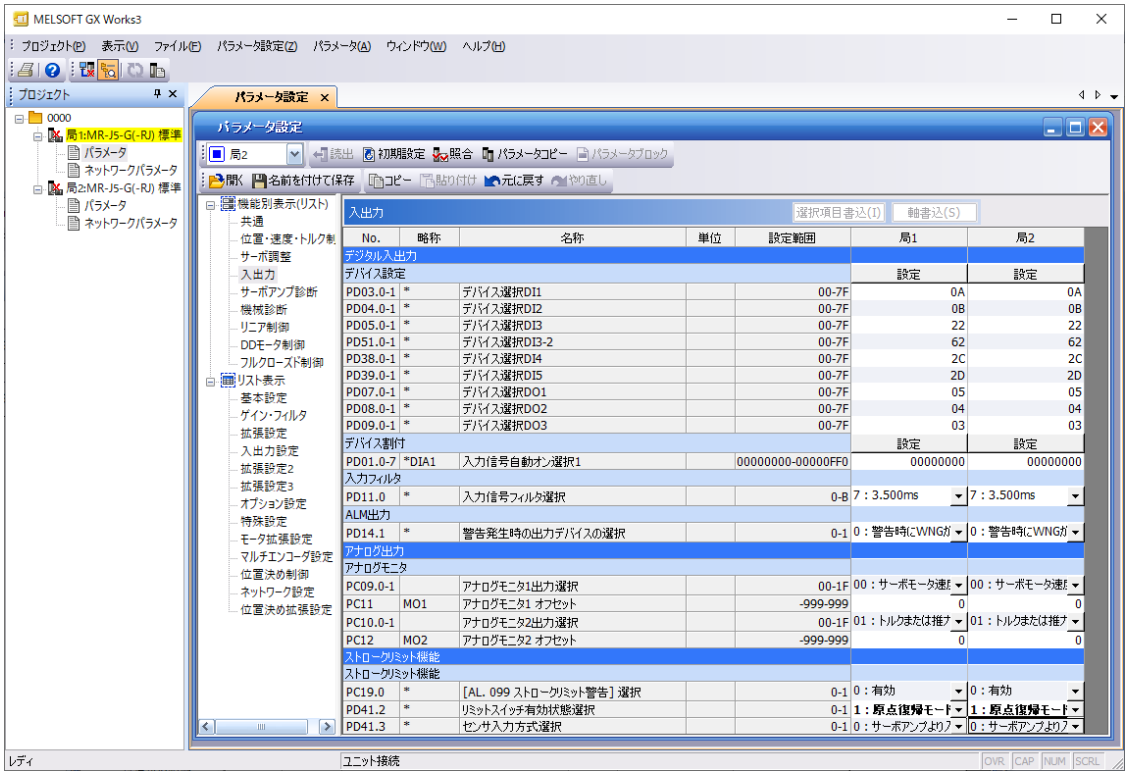
本書では，MELSOFT GX Works3からシーケンサCPUに書き込む場合の手順を説明します。

①“CC-Link IE TSN構成”ウィンドウで，“パラメータ自動設定”にチェックを入れることにより，イニシャル通信時にマスタ局からスレーブ局へパラメータを書き込みます。

②設定するサーボアンプのイラストをダブルクリックして，確認画面で[はい]ボタンをクリックします。



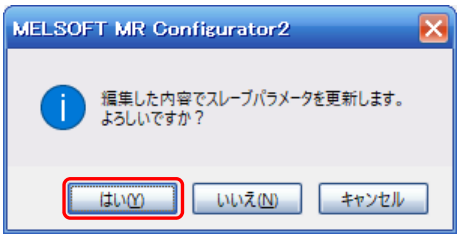
③「パラメータ設定」画面でサーボパラメータを設定します。



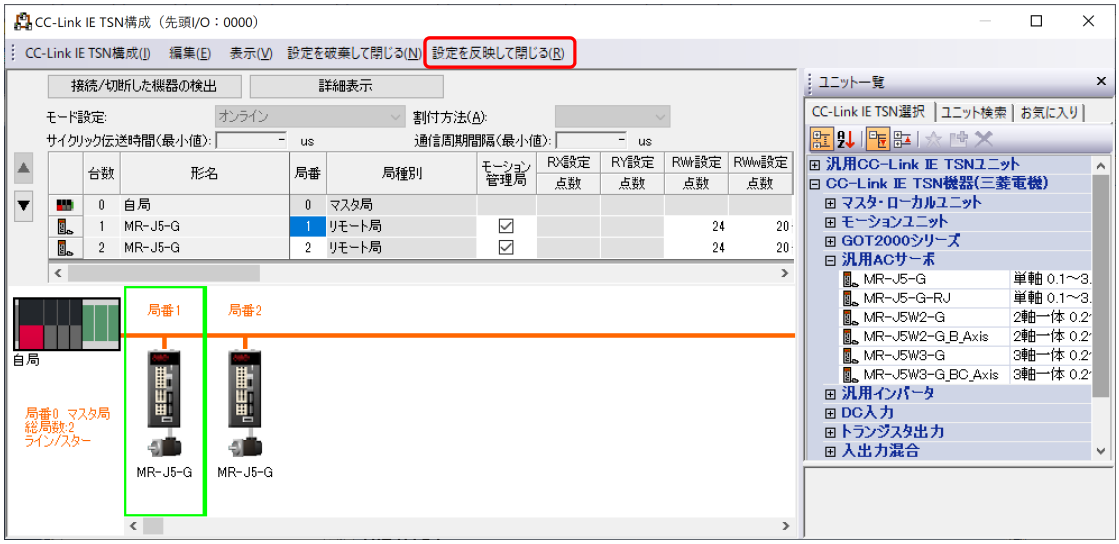

本書では，以下のパラメータを変更します。

| 表示位置 | 番号 | 項　目 | 設定値 |
|---|---|---|---|
| [共通]→[基本設定]→[強制停止] | PA04.2 | サーボ強制停止選択 | 0：有効 ⇒ 1：無効 |
| [入出力]→[ストロークリミット機能]→[ストロークリミット機能] | PC19.0 | [AL.099 ストロークリミット警告] 選択 | 0：有効 ⇒ 1：無効 |
| | PD41.2 | リミットスイッチ有効状態選択 | 0：常時有効<br>⇒ 1：原点復帰モード時のみ有効 |

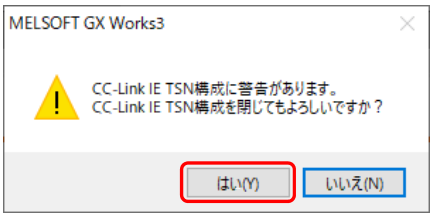
④ウィンドウ右上[×]ボタンをクリックして，確認画面を表示します。
[はい]ボタンをクリックして，パラメータを更新します。



⑤[設定を反映して閉じる]をクリックして，データを反映します。




バージョンによっては警告ウィンドウが出ますが，[はい]ボタンをクリックしてください。



## 3.8 ユニットパラメータの適用

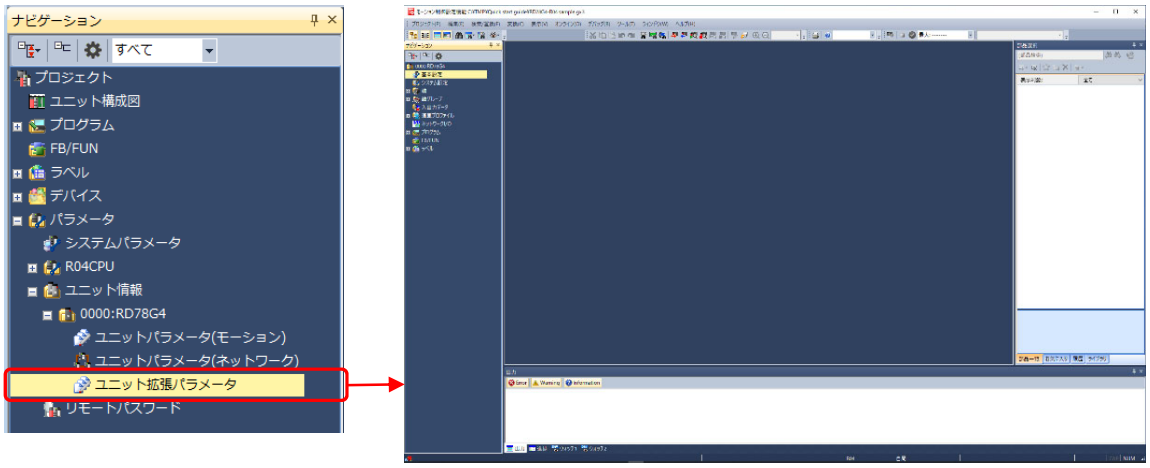
パラメータエディタ(ユニットパラメータ)で[適用]ボタンをクリックして，モーションユニットのパラメータを反映します。

**[ポイント]**
[適用]ボタンをクリックしないと，パラメータは反映されませんので注意してください。

## 3.9　モーション制御設定

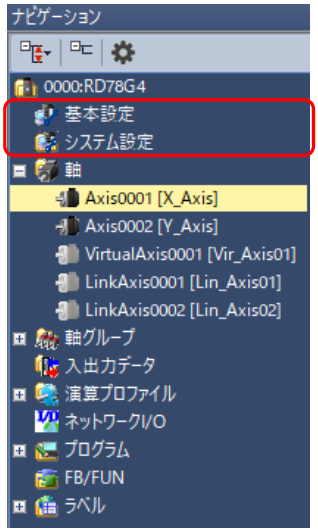
モーションユニットのパラメータ設定やプログラムは，モーション制御設定画面で行ないます。

ナビゲーションウィンドウで，使用するモーションユニットの“ユニット拡張パラメータ”をダブルクリックして，モーション制御設定機能を表示します。
設定した内容は，MELSOFT GX Works3のプロジェクトとして保存されます。




## 3.10　基本設定，システム設定

基本設定は，プログラムにおける各種パラメータを設定し，システム設定は，全軸緊急停止，アドオン時の各種パラメータを設定します。
必要に応じて変更してください。（本書では初期値のままとします。）

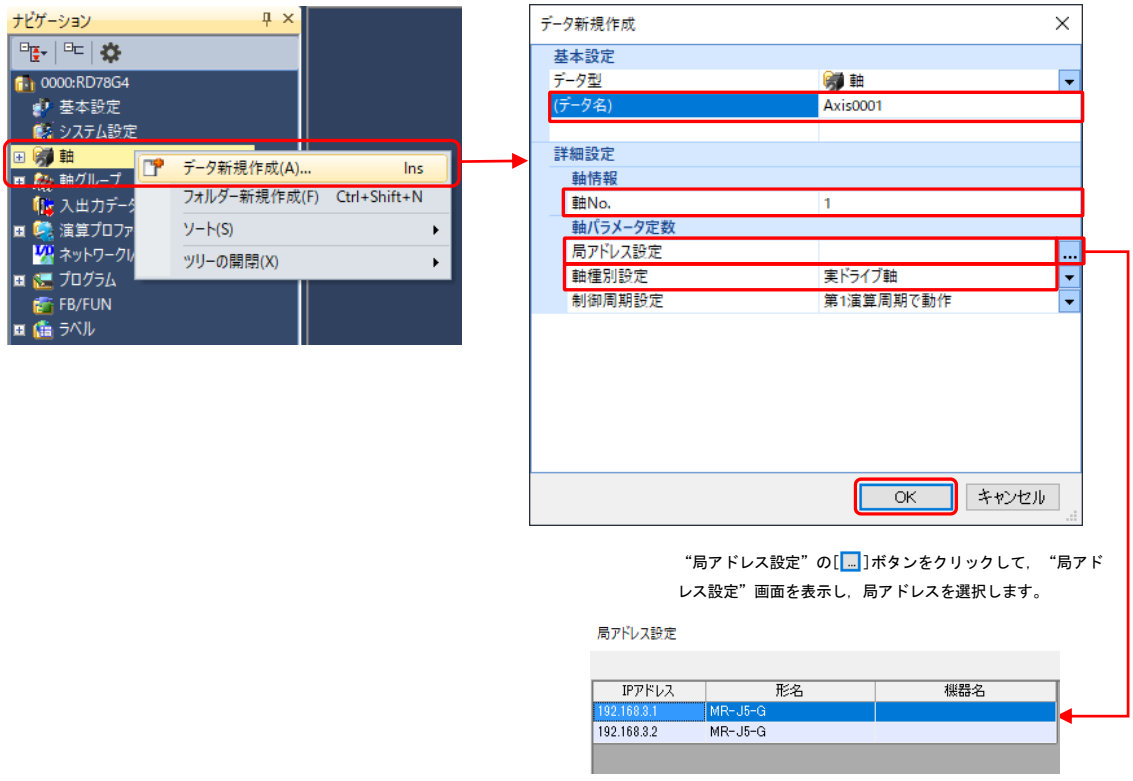


## 3.11　軸パラメータの設定

### 3.11.1　局アドレス，軸種別の設定

軸種別には，以下の5種類があります。
・実ドライブ軸　　：CC-Link IE TSN上に接続されたサーボアンプを使用して指令を出力する軸
・実エンコーダ軸　：CC-Link IE TSN上のサーボアンプに接続した同期エンコーダの出力パルスから現在位置を生成する軸
・仮想ドライブ軸　：仮想的に指令を生成できる軸
・仮想エンコーダ軸：仮想的に変数の値から現在位置を生成する軸
・仮想連結軸　　：仮想的に単軸同期制御の各FB間を接続するための軸

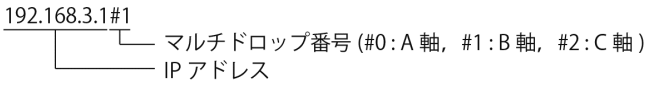
本書では，実ドライブ軸を2点，仮想ドライブ軸を1点，仮想連結軸を2点登録します。
①実ドライブ軸
ネットワーク構成で定義したサーボアンプと軸パラメータの軸情報を関連付けるため，局アドレスを設定します。
モーション制御設定機能で，ナビゲーションウィンドウの“軸”を右クリックし，[データ新規作成]を選択します。
データ名，軸No.，局アドレス設定，軸種別設定を下表のとおり入力して，[OK]ボタンをクリックします。




| 設定項目 | 1軸目 | 2軸目 |
|---|---|---|
| データ名 | Axis0001 | Axis0002 |
| 軸 No. | 1 | 2 |
| 局アドレス設定 | 192. 168. 3. 1 | 192. 168. 3. 2 |
| 軸種別設定 | 実ドライブ軸 | 実ドライブ軸 |

**[ポイント]**
局アドレス設定には，CC-Link IE TSN構成ウィンドウで設定したIPアドレスを使用します。
マルチドロップ番号は，多軸一体サーボアンプの1軸目以外の局アドレスを指定した場合に表示されます。

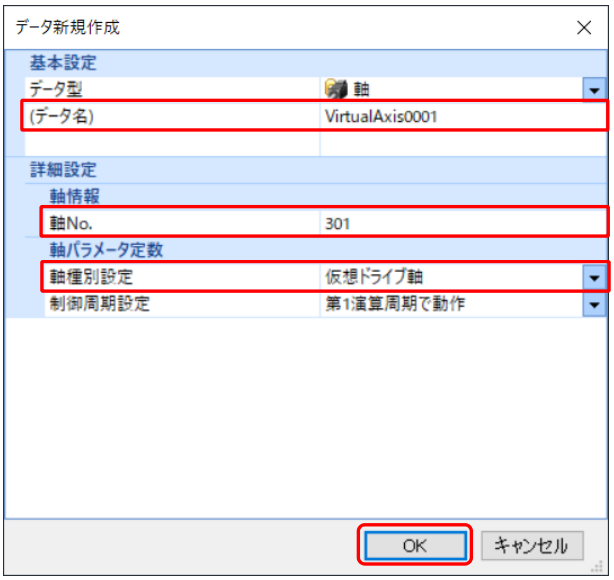
（例）MR-J5-W3でIPアドレス192.168.3.1，B軸の場合




②仮想ドライブ軸

モーション制御設定機能で，ナビゲーションウィンドウの“軸”を右クリックし，[データ新規作成]を選択します。

データ名，軸No.，軸種別設定を下表のとおり入力して，[OK]ボタンをクリックします。




| 設定項目 | 1軸目 |
|---|---|
| データ名 | VirtualAxis0001 |
| 軸 No. | 301 |
| 軸種別設定 | 仮想ドライブ軸 |

③仮想連結軸

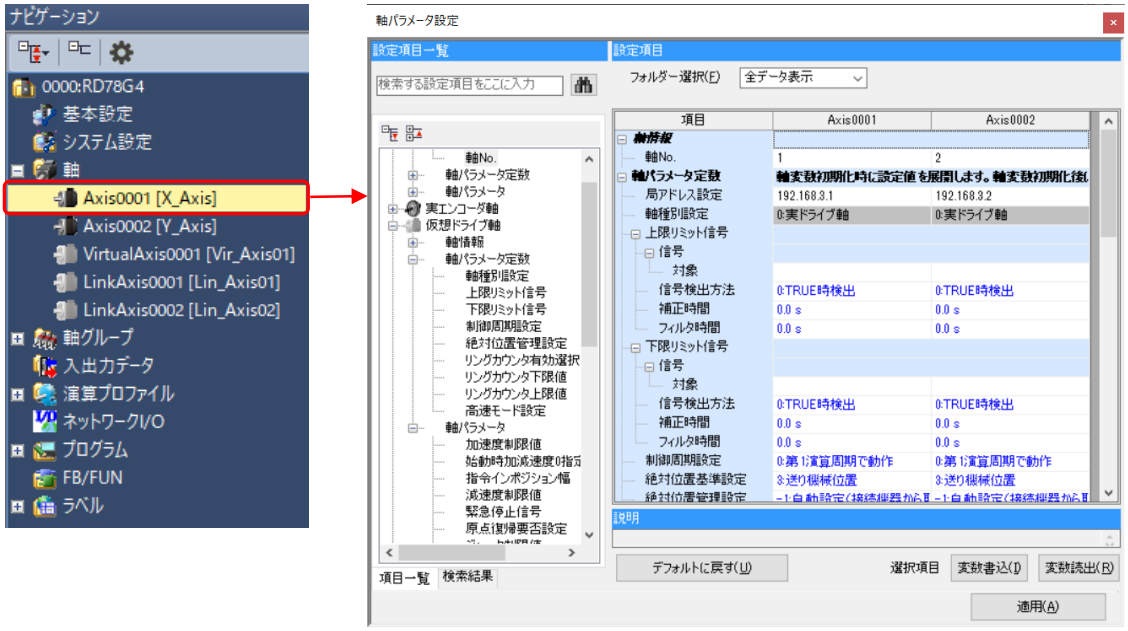
モーション制御設定機能で，ナビゲーションウィンドウの“軸”を右クリックし，[データ新規作成]を選択します。

データ名，軸No.，軸種別設定を下表のとおり入力して，[OK]ボタンをクリックします。

| 設定項目 | 1軸目 | 2軸目 |
|---|---|---|
| データ名 | LinkAxis0001 | LinkAxis0002 |
| 軸 No. | 401 | 402 |
| 軸種別設定 | 仮想連結軸 | 仮想連結軸 |

### 3.11.2　各項目の設定

①実ドライブ軸
ナビゲーションウィンドウの“軸”⇒“Axis0001”をダブルクリックして，“軸パラメータ設定”ウィンドウを表示します。



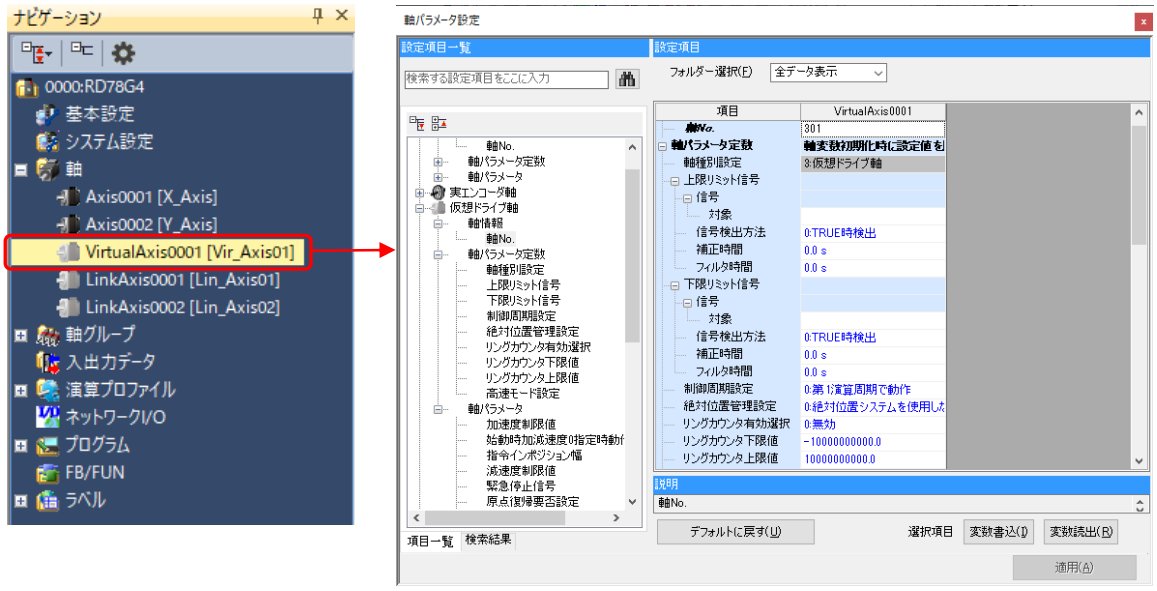

本書での設定は以下のとおりです。

| 項　目 | | 内　容 | Axis0001<br>設定値 | Axis0002<br>設定値 |
|---|---|---|---|---|
| 軸パラメータ<br>定数 | 局アドレス設定 | ドライバの IP アドレスを設定 | 192.168.3.1 | 192.168.3.2 |
| | 上限リミット信号<br>下限リミット信号 | 可動範囲の上限／下限に設置したハードウェアストロークリミットスイッチを外部信号割り付けで設定 | 初期値<br>（付録2 参照） | |
| 軸パラメータ | ドライバ単位変換分子 | コントローラとドライバ間の目標位置とフィードバック位置の単位変換<br>（設定方法は 3.11.3 項参照） | 67108864 | |
| | ドライバ単位変換分母 | | 5000 | |
| | 停止信号 → 信号 → 対象 | 停止信号を設定 | [VAR]G_bStopSignalX<br>※1 | [VAR]G_bStopSignalY<br>※1 |
| | 緊急停止信号，停止信号 | 緊急停止，停止信号を設定 | 初期値 | |
| | 原点復帰未完時始動許可 | 原点復帰要求がTRUEの場合における軸始動可否を設定 | 0：許可しない | |
| | ソフトウェアストローク<br>リミット上限値/下限値/対象 | ソフトウェアストロークリミット範囲と対象を設定。 | 初期値 | |
| | 位置指令単位 | モーション制御で使用する位置指令単位と速度指令単位を設定<br>本書では，位置指令単位[um]，速度指令単位[um/s]とします。 | um※2 | |
| | 速度指令単位 | | U/s | |

※1：グローバルラベルを停止信号に使用します。グローバルラベルの設定は，4.3.2 項，または 5.3.2 項を参照してください。
※2：um はマイクロメートルを示します。

②仮想ドライブ軸

ナビゲーションウィンドウの“軸”⇒“VirtualAxis0001”をダブルクリックして，“軸パラメータ設定”ウィンドウを表示します。

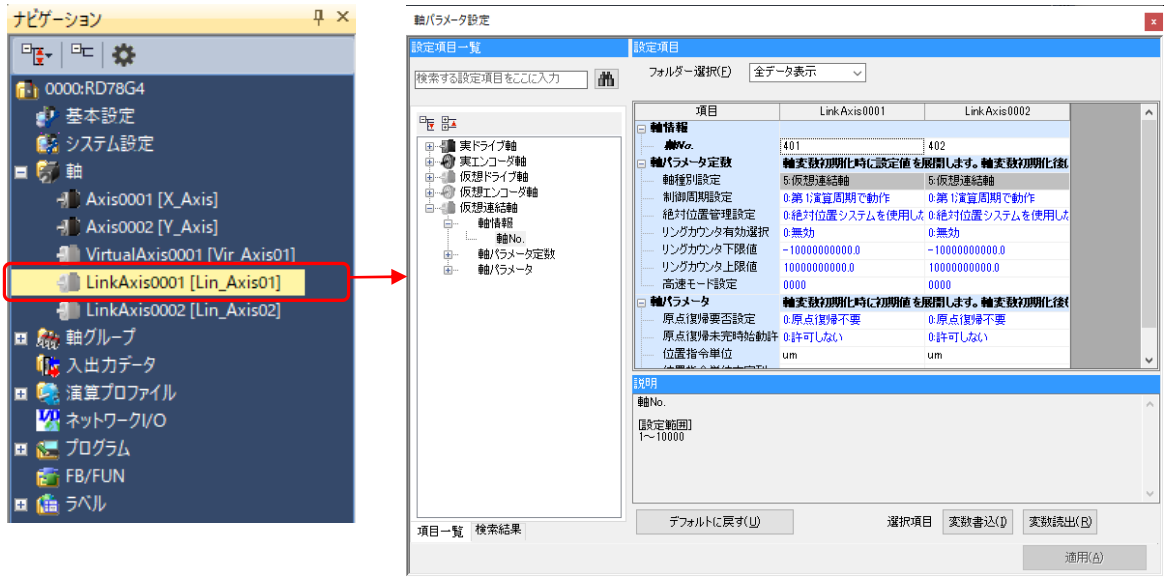



本書での設定は以下のとおりです。

| 項　目 | | 内　容 | 設定値 |
|---|---|---|---|
| 軸パラメータ | 位置指令単位 | モーション制御で使用する位置指令単位と速度指令単位を設定<br>本資料では，実ドライブ軸の設定にあわせて，位置指令単位[um]，速度指令単位[um/s]で説明します。 | um |
| | 速度指令単位 | | U/s |

③仮想連結軸

ナビゲーションウィンドウの“軸”⇒“LinkAxis0001”をダブルクリックして，“軸パラメータ設定”ウィンドウを表示します。




本書での設定は以下のとおりです。

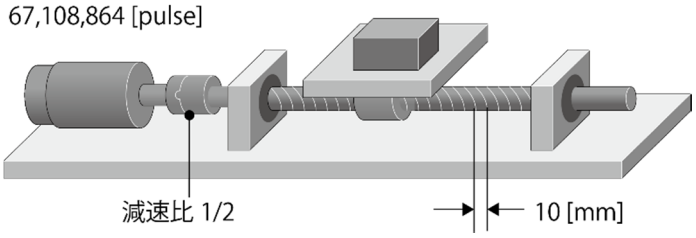
| 項　目 | | 内　容 | 設定値 |
|---|---|---|---|
| 軸パラメータ | 位置指令単位 | モーション制御で使用する位置指令単位と速度指令単位を設定<br>本資料では，実ドライブ軸の設定にあわせて，位置指令単位[um]，速度指令単位[um/s]で説明します。 | um |
| | 速度指令単位 | | U/s |

### 3.11.3 ドライバ単位変換（電子ギア）

ドライバ単位変換分子／ドライバ単位変換分母の設定例を以下に示します。

#### (1) 電子ギアの考え方

[ボールネジ]




サーボモータのエンコーダ分解能26ビット： 67,108,864 [pulse] (26 bit)
ボールネジリード： 10000[um]
減速比： 1/2（負荷側[NL]／モータ側[NM]）
モータが2回転すると，負荷側のボールネジが1回転します。

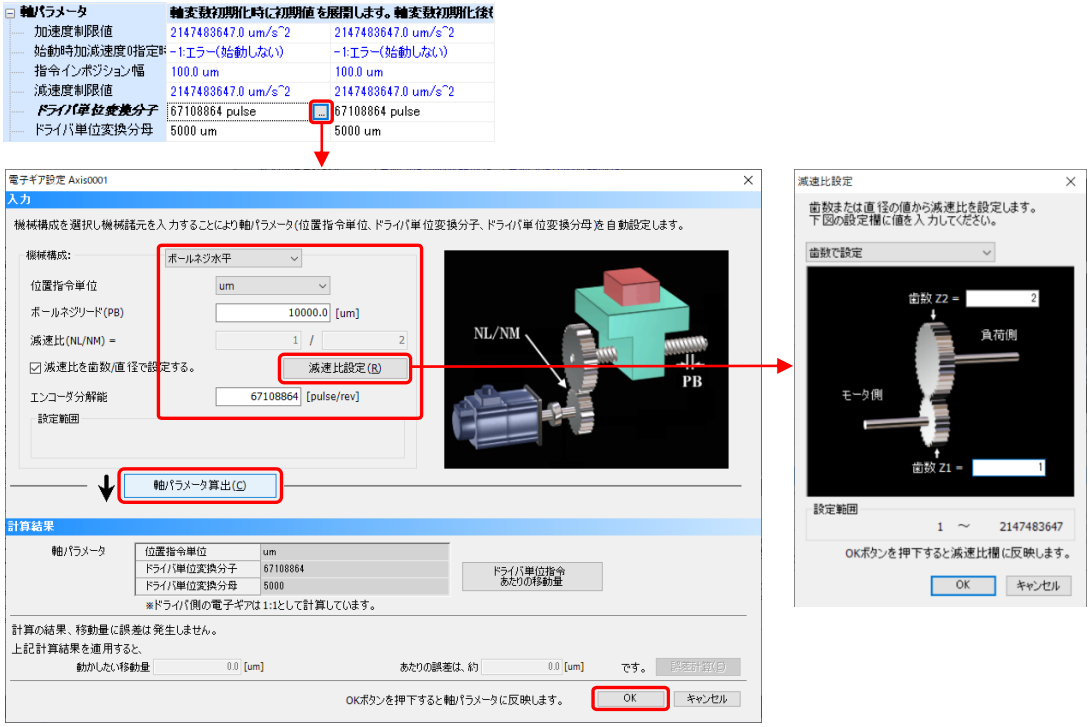
$$\frac{\textit{ドライバ単位変換分子}}{\textit{ドライバ単位変換分母}} = \frac{\textit{エンコーダパルス数}}{\textit{移動量} \times \textit{減速比}} = \frac{67108864}{10000 \times 1/2} = \frac{67108864}{5000}$$

ドライバ単位変換分子 = 1回転あたりパルス数 67,108,864
ドライバ単位変換分母 = 1回転あたり移動量 5000

### (2) 電子ギアの設定

電子ギアの設定手順を以下に示します。

①“軸パラメータ”⇒“ドライバ単位変換分子”，または“ドライバ単位変換分母”の[...]ボタンをクリックして，“電子ギア設定”画面を表示します。



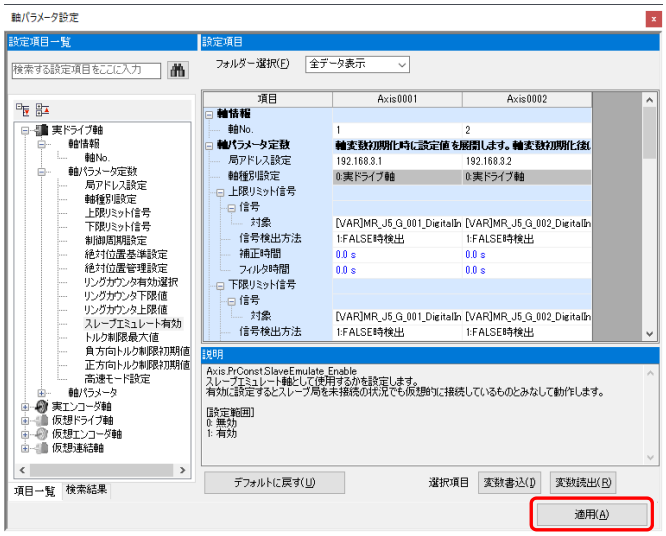

②機械構成の各諸元を入力します。[減速比設定]ボタンをクリックして，減速比設定の補助画面を表示します。

③[軸パラメータ算出]ボタンをクリックします。

④[OK]ボタンをクリックして，計算結果を軸パラメータに反映します。

## 3.11.4 軸パラメータの反映

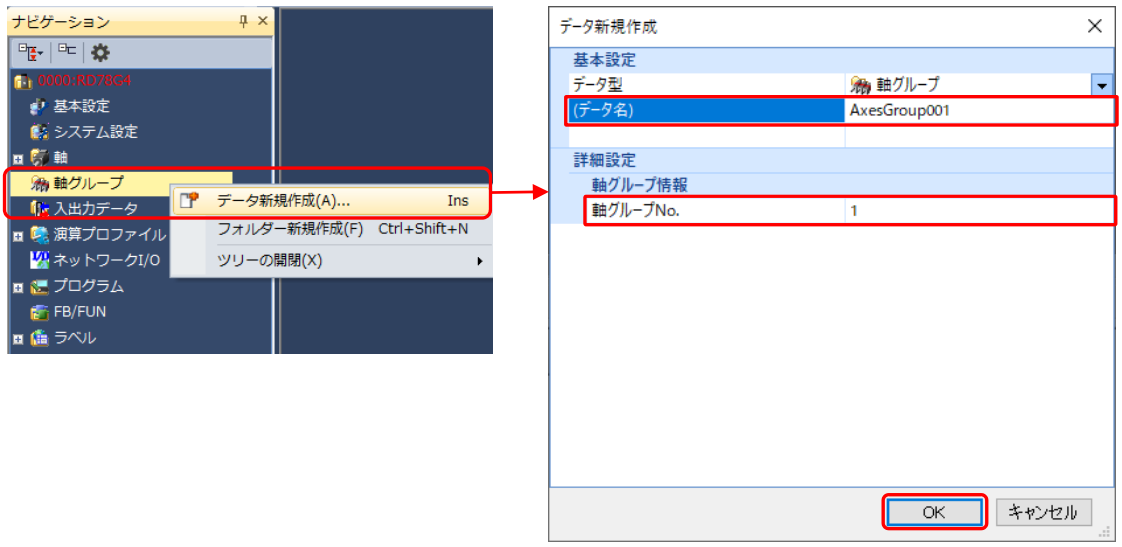
各項目を設定後，[適用]ボタンをクリックしてパラメータを確定します。



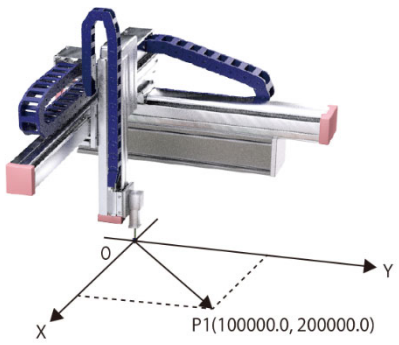
## 3.12　軸グループの設定

軸グループは，直線補間，円弧補間などの多軸制御を行う場合に使用します。
同期制御に関連する軸は，軸グループを作成する必要はありません。

モーション制御設定機能で，ナビゲーションウィンドウの“軸グループ”を右クリックして，[データ新規作成]を選択します。データ名と軸グループNo.を確認して[OK]ボタンをクリックします。



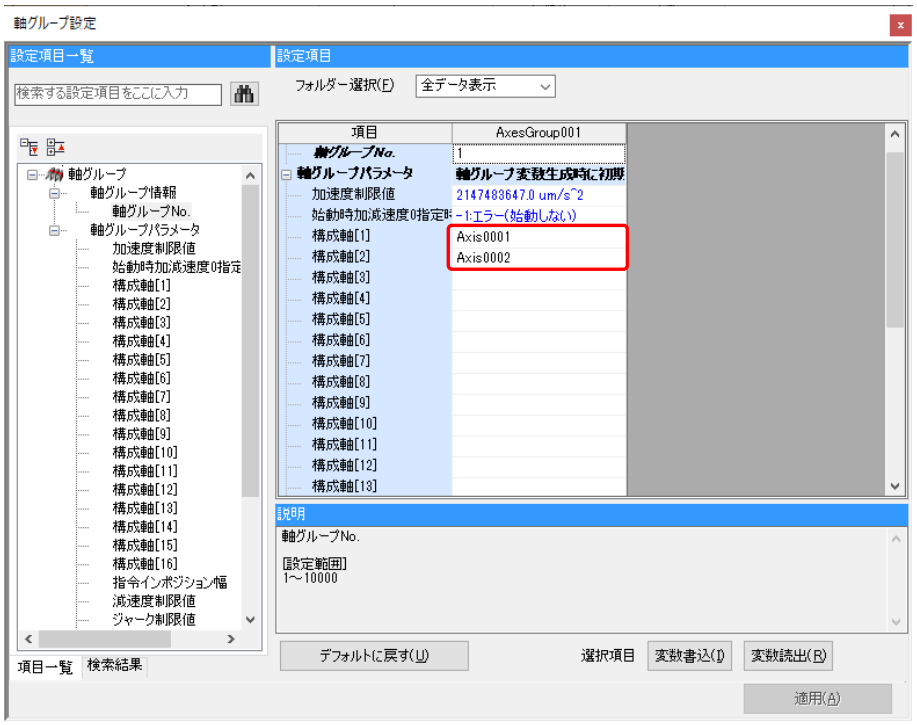

本書で扱う2軸システムの装置（2.1節参照）では，X軸(Axis0001)，Y軸(Axis0002)を軸グループに登録します。



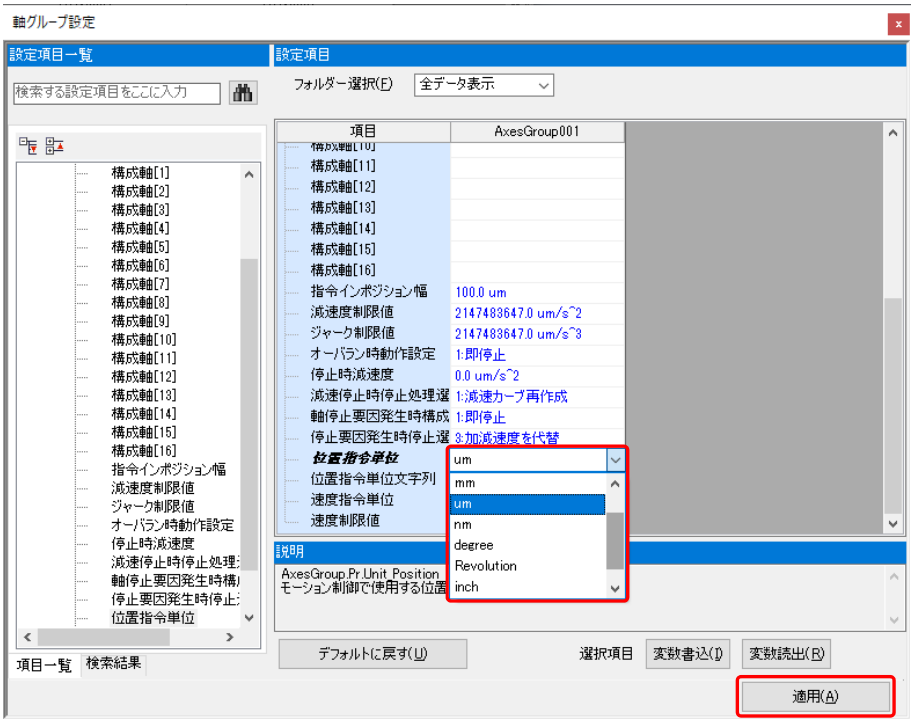

[単位：μm]

軸グループの設定例を以下に示します。

[軸グループ構成軸の設定例]




[軸グループ構単位の設定例]　（位置指令単位：[um]，速度指令単位：[U/s]）




各項目を設定後，[適用]ボタンをクリックして，軸グループの設定を確定します。

## 3.13　ラベル

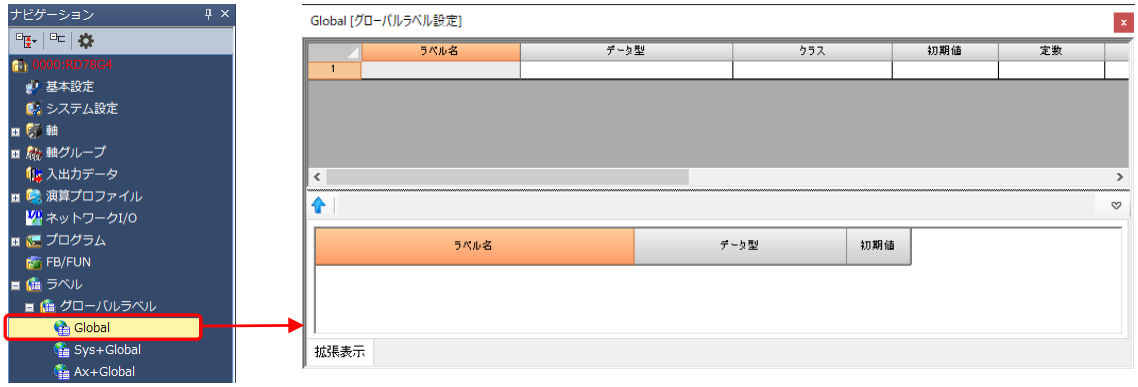
ラベルには，ローカルラベルとグローバルラベルがあります。
用途にあわせて使用してください。

| ラベル | 内　容 |
|---|---|
| ローカルラベル | ・1つのプログラム中のみで使用可能 |
| グローバルラベル | ・全てのプログラムで使用可能<br>・公開ラベルに設定可能（3.14節参照） |

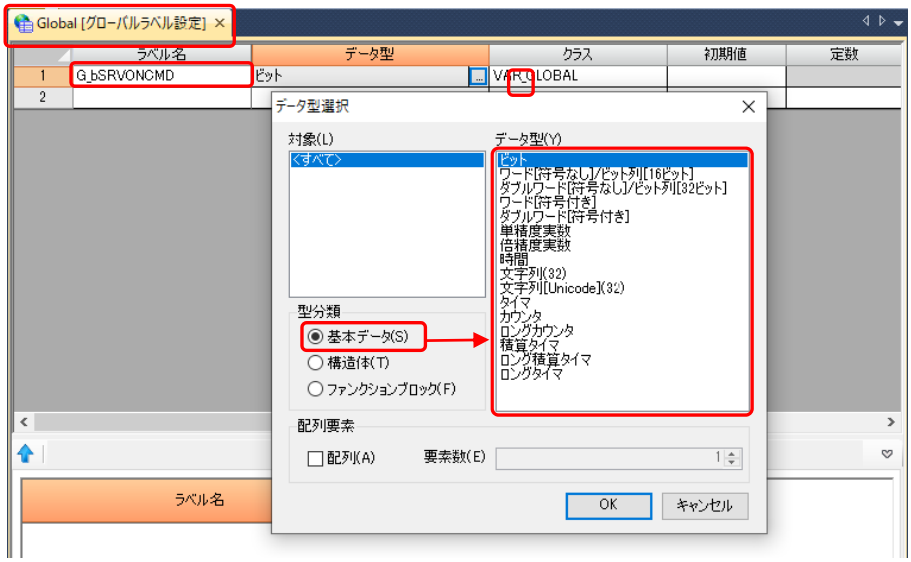
### 3.13.1　ラベルの作成

本項では，ラベルの作成方法について説明します。
実際のラベルの登録例については，4.3.2項，または5.3.2項を参照してください。

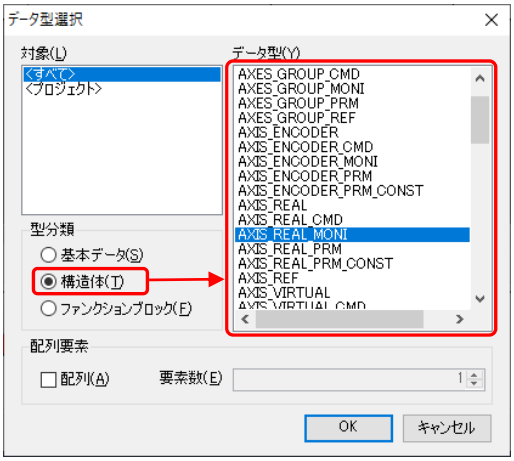
①モーション制御設定機能で，ナビゲーションウィンドウの“ラベル”⇒“グローバルラベル”
⇒“Global”をダブルクリックして，“Global[グローバルラベル設定]”画面を表示します。

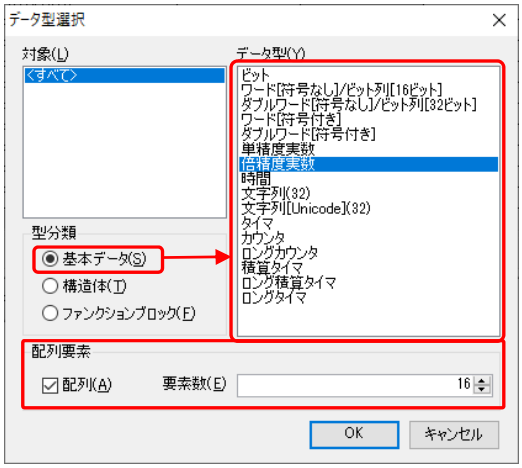



②グローバルラベルの設定を行ないます。
(a)ラベル名を入力します。（設定例：G_bSRVONCMD）
(b)データ型を選択します。（[...]ボタンをクリックして，“データ型選択”画面を表示。）
型分類で“基本データ”を選択し，データ型を選択します。



構造体変数，配列変数も基本データと同様に設定し，[OK]ボタンをクリックします。




**[グローバルラベルの登録例]**

第4章で使用するプログラムのグローバルラベルです。（詳細は第4章参照）

Global [グローバルラベル設定]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | G_bSRVONCMD | ビット | VAR_GLOBAL | | | サーボオンオフ | | 有効 | WRITE(⇒モーション) |
| 2 | G_bHoming1CMD | ビット | VAR_GLOBAL | | | 原点復帰指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 3 | G_bHoming2CMD | ビット | VAR_GLOBAL | | | 原点復帰指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 4 | G_bHoming3CMD | ビット | VAR_GLOBAL | | | 原点復帰指令 VirtualAxis0001 | | 有効 | WRITE(⇒モーション) |
| 5 | G_bPosCMD | ビット | VAR_GLOBAL | | | 単軸位置決め始動 | | 有効 | WRITE(⇒モーション) |
| 6 | G_bContPosCMD | ビット | VAR_GLOBAL | | | 単軸連続位置決め始動 | | 有効 | WRITE(⇒モーション) |
| 7 | G_bInterpolationCMD | ビット | VAR_GLOBAL | | | 2軸直線補間制御始動 | | 有効 | WRITE(⇒モーション) |
| 8 | G_bSyncCMD | ビット | VAR_GLOBAL | | | 同期制御始動 | | 有効 | WRITE(⇒モーション) |
| 9 | G_bResetCMD | ビット | VAR_GLOBAL | | | エラーリセット | | 有効 | WRITE(⇒モーション) |
| 10 | G_bMotionResetCMD | ビット | VAR_GLOBAL | | | システムエラーリセット | | 有効 | WRITE(⇒モーション) |
| 11 | G_bJogF1CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 12 | G_bJogR1CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 13 | G_bJogF2CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 14 | G_bJogR2CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 15 | G_leJogVelocity | 倍精度実数 | VAR_GLOBAL | | | JOG速度 | | 有効 | WRITE(⇒モーション) |
| 16 | G_bHoming1Done | ビット | VAR_GLOBAL | | | 原点復帰完了 Axis0001 | | 有効 | READ(モーション⇒) |
| 17 | G_bHoming2Done | ビット | VAR_GLOBAL | | | 原点復帰完了 Axis0002 | | 有効 | READ(モーション⇒) |
| 18 | G_bHoming3Done | ビット | VAR_GLOBAL | | | 原点復帰完了 VirtualAxis0001 | | 有効 | READ(モーション⇒) |
| 19 | G_bPosDone | ビット | VAR_GLOBAL | | | 単軸位置決め完了 | | 有効 | READ(モーション⇒) |
| 20 | G_bContPosDone | ビット | VAR_GLOBAL | | | 単軸連続位置決め完了 | | 有効 | READ(モーション⇒) |
| 21 | G_bInterpolationDone | ビット | VAR_GLOBAL | | | 2軸直線補間制御完了 | | 有効 | READ(モーション⇒) |
| 22 | G_bSyncDone | ビット | VAR_GLOBAL | | | 同期制御完了 | | 有効 | READ(モーション⇒) |
| 23 | G_bJog1Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0001 | | 有効 | READ(モーション⇒) |
| 24 | G_bJog2Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0002 | | 有効 | READ(モーション⇒) |
| 25 | G_bStopSignalX | ビット | VAR_GLOBAL | | | 停止指令 Axis0001 | | 有効 | READ(モーション⇒) |
| 26 | G_bStopSignalY | ビット | VAR_GLOBAL | | | 停止指令 Axis0002 | | 有効 | READ(モーション⇒) |
| 27 | | | | | | | | | |

※：公開ラベルの詳細は，3. 14節を参照してください。

## 3.14　公開ラベル

グローバルラベル，構造体のメンバを公開することにより，シーケンサCPUのプログラムより読み出し，または書き込みが可能になります。
また，位置や速度のモニタデータなどは，モーションユニット内で決まった名称の変数(ラベル)で定義されています。これらのラベルを公開すると，シーケンサCPUで使用可能となります。
本節では，ラベルの作成方法について説明します。実際のラベルの登録例については，4.3.2項，または5.3.2項を参照してください。

### 3.14.1　公開ラベルの登録

実ドライブ軸の軸情報AxisRef，原点復帰要求，および指令インポジションを公開ラベルに設定する方法を以下に示します。

①モーション制御設定機能で，ナビゲーションウィンドウの“ラベル”⇒“グローバルラベル”⇒“Ax+Global”をダブルクリックし，公開ラベルにする軸ラベルを有効にします。
②“構造体”⇒“AXIS_REAL”をダブルクリックし，軸情報AxisRef，軸モニタデータMdを有効にします。
③“構造体”⇒“AXIS_REAL_MONI”をダブルクリックし，軸状態AxisStatusを有効にします。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Axis0001 | AXIS_REAL | VAR_GLOBAL | <詳細設定> | | [X_Axis] | | 有効 | - |
| 2 | Axis0002 | AXIS_REAL | VAR_GLOBAL | <詳細設定> | | [Y_Axis] | | 有効 | - |
| 3 | VirtualAxis0001 | AXIS_VIRTUAL | VAR_GLOBAL | <詳細設定> | | [Vir_Axis01] | | 有効 | - |
| 4 | LinkAxis0001 | AXIS_VIRTUAL_LINK | VAR_GLOBAL | <詳細設定> | | [Lin_Axis01] | | 有効 | - |
| 5 | LinkAxis0002 | AXIS_VIRTUAL_LINK | VAR_GLOBAL | <詳細設定> | | [Lin_Axis02] | | 有効 | - |
| 6 | | | | | | | | | |

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 7 | AutoDeceleration | ビット | | 0 | | 自動減速中 | 無効 | - |
| 8 | AxisName | 文字列[Unicode](127) | | "" | | 軸名称 | 無効 | - |
| 9 | AxisStatus | ワード[符号付き] | | 0 | | 軸状態 | 有効 | - |
| 10 | BufferingFBs | ワード[符号付き] | | 0 | | バッファリングFB数 | 無効 | - |
| 11 | CmdInPos | ビット | | 0 | | 指令インポジション | 有効 | - |
| 12 | CmdInPos_Width | 倍精度実数 | | 100.0 | | 指令インポジション幅 | 無効 | - |

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AxisRef | AXIS_REF | | | | 軸情報 | 有効 | READ(モーション⇒):… |
| 2 | PrConst | AXIS_REAL_PRM_CONST | | | | 軸パラメータ定数 | 無効 | WRITE(⇒モーション)… |
| 3 | Pr | AXIS_REAL_PRM | | | | 軸パラメータ | 無効 | WRITE(⇒モーション)… |
| 4 | Md | AXIS_REAL_MONI | | | | 軸モニタデータ | 有効 | READ(モーション⇒):… |
| 5 | Cd | AXIS_REAL_CMD | | | | 軸制御データ | 無効 | WRITE(⇒モーション)… |
| 6 | | | | | | | | |

［ユーザ作成のグローバルラベルの公開］

ユーザ作成のグローバルラベルについても，公開ラベルに設定することで，シーケンサCPUより読み出し，または書き込みが可能になります。

ユーザ作成のグローバルラベルは，モーション制御属性の設定が必要です。

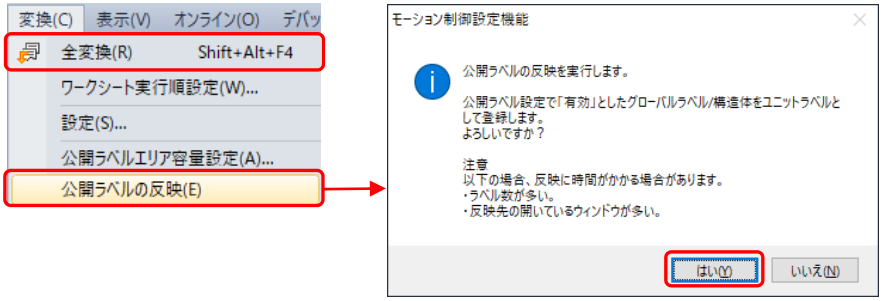
| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 8 | G_bSyncCMD | ビット | VAR_GLOBAL | | | 同期制御始動 | | 有効 | WRITE(⇒モーション) |
| 9 | G_bResetCMD | ビット | VAR_GLOBAL | | | エラーリセット | | 有効 | WRITE(⇒モーション) |
| 10 | G_bJogF1CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 11 | G_bJogR1CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 12 | G_bJogF2CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 13 | G_bJogR2CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 14 | G_leJogVelocity | 倍精度実数 | VAR_GLOBAL | | | JOG速度 | | 有効 | WRITE(⇒モーション) |
| 15 | G_bHoming1Done | ビット | VAR_GLOBAL | | | 原点復帰完了 Axis0001 | | 有効 | READ(モーション⇒) |
| 16 | G_bHoming2Done | ビット | VAR_GLOBAL | | | 原点復帰完了 Axis0002 | | 有効 | READ(モーション⇒) |
| 17 | G_bHoming3Done | ビット | VAR_GLOBAL | | | 原点復帰完了 VirtualAxis0001 | | 有効 | READ(モーション⇒) |
| 18 | G_bPosDone | ビット | VAR_GLOBAL | | | 単軸位置決め完了 | | 有効 | READ(モーション⇒) |
| 19 | G_bContPosDone | ビット | VAR_GLOBAL | | | 単軸連続位置決め完了 | | 有効 | READ(モーション⇒) |
| 20 | G_blnterpolationDone | ビット | VAR_GLOBAL | | | 補間制御完了 | | 有効 | READ(モーション⇒) |
| 21 | G_bSyncDone | ビット | VAR_GLOBAL | | | 同期制御完了 | | 有効 | READ(モーション⇒) |
| 22 | G_bJog1Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0001 | | 有効 | READ(モーション⇒) |
| 23 | G_bJog2Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0002 | | 有効 | READ(モーション⇒) |

・シーケンサCPUがモーションユニットから読み出し　：READ(モーション ⇒)
・シーケンサCPUからモーションユニットへ書き込み　：WRITE(⇒ モーション)

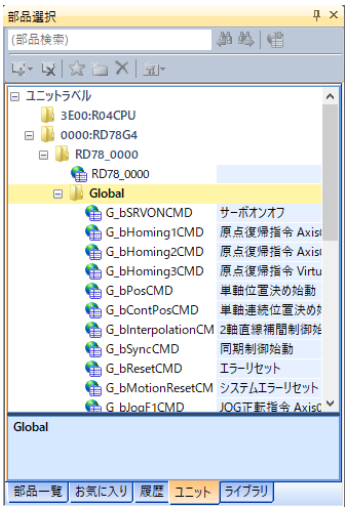
## 3.14.2　公開ラベルの反映

登録した公開ラベルを反映します。
公開ラベルを反映すると，有効にしたメンバをシーケンサCPUから参照できます。

①モーション制御設定機能で，［変換］⇒［全変換］を選択して，全ての公開ラベルを変換します。
②［変換］⇒［公開ラベルの反映］を選択して，確認画面で［はい］ボタンをクリックします。



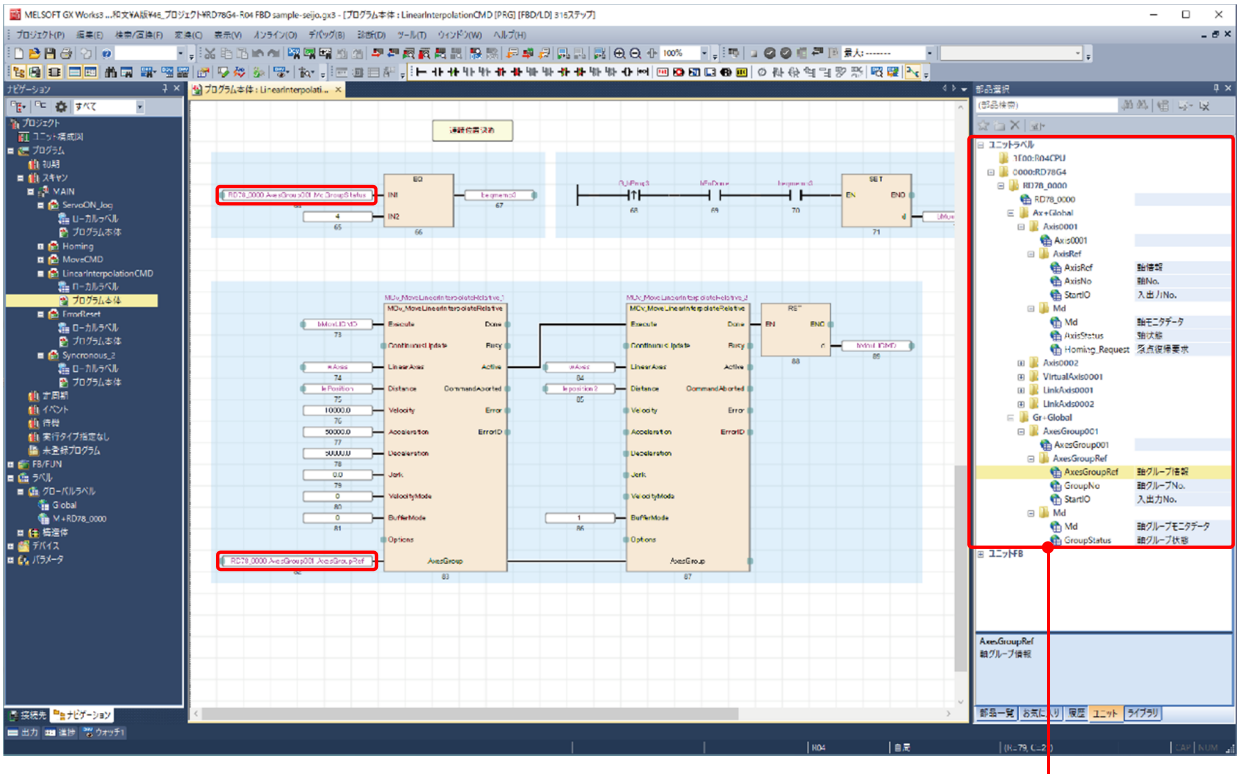

③シーケンサCPUで公開ラベルを使用するために，MELSOFT GX Works3側で全変換してください。
公開ラベルがMELSOFT GX Works3の部品選択ウィンドウのユニットラベルに登録されます。



### 3.14.3　公開ラベルの使用例

公開ラベルの使用例を以下に示します。

①部品選択ウィンドウの[ユニット]タブから“ユニットラベル”をダブルクリックして，公開ラベルを表示します。公開ラベルをドラッグ＆ドロップして使用します。




**[ポイント]**

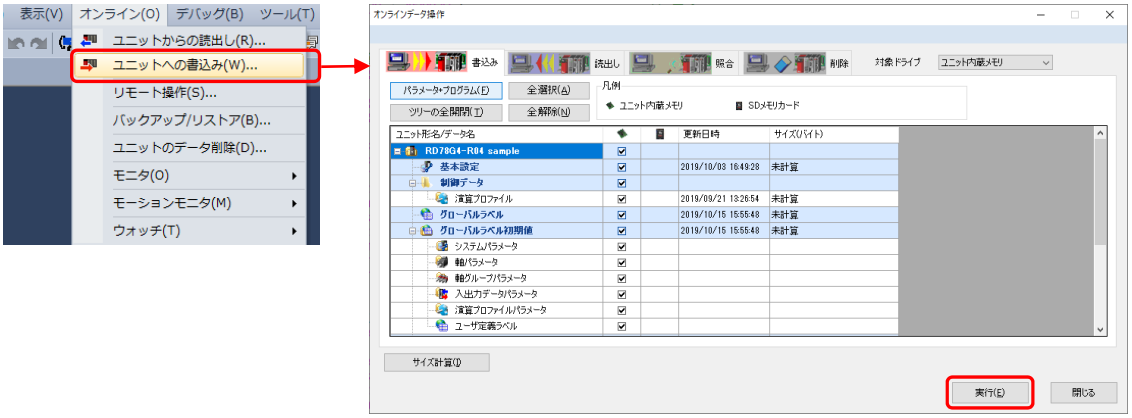
モーションユニットの公開ラベルをシーケンサCPUで参照する場合は，先頭に“ユニット＋ユニットNo.”が追加されます。

RD78_0000.Axis0001.AxisRef
- メンバ
- 軸ラベル名
- ユニット ＋ ユニット No.

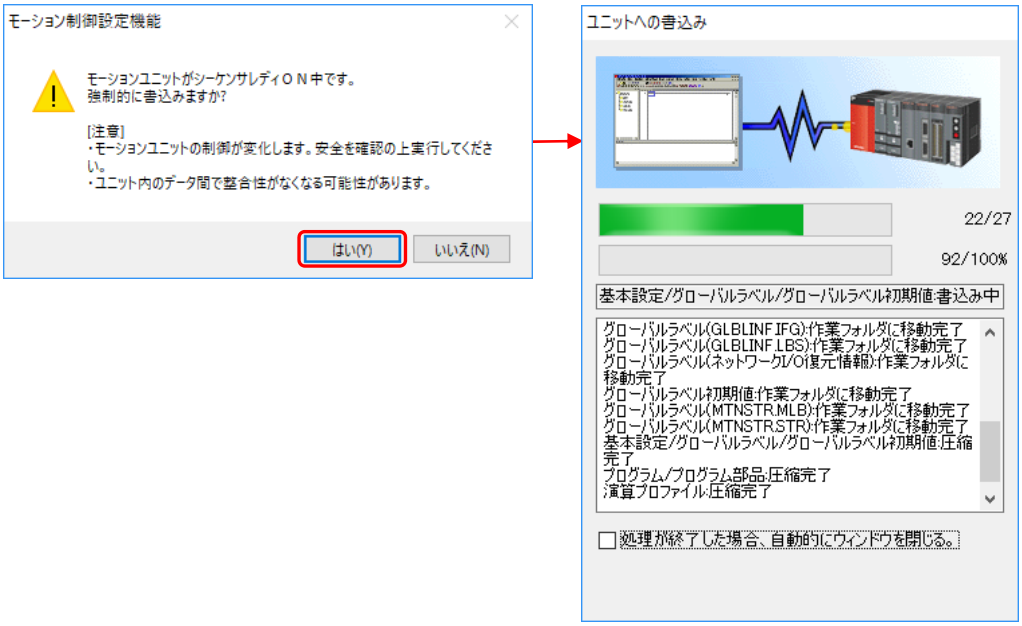
## 3.15　モーションユニットへの書き込み

シーケンサCPUにユニットパラメータを書き込み後，シーケンサCPUをリセットして再度電源を投入すると，モーションユニットと通信できるようになります。
次に，モーション制御設定で設定した内容をモーションユニットに書き込みます。

①[オンライン]⇒[ユニットへの書込み]を選択して，“オンラインデータ操作”画面を表示します。内容を確認して，[実行]ボタンをクリックします。




②パラメータをモーションユニットに書き込みます。

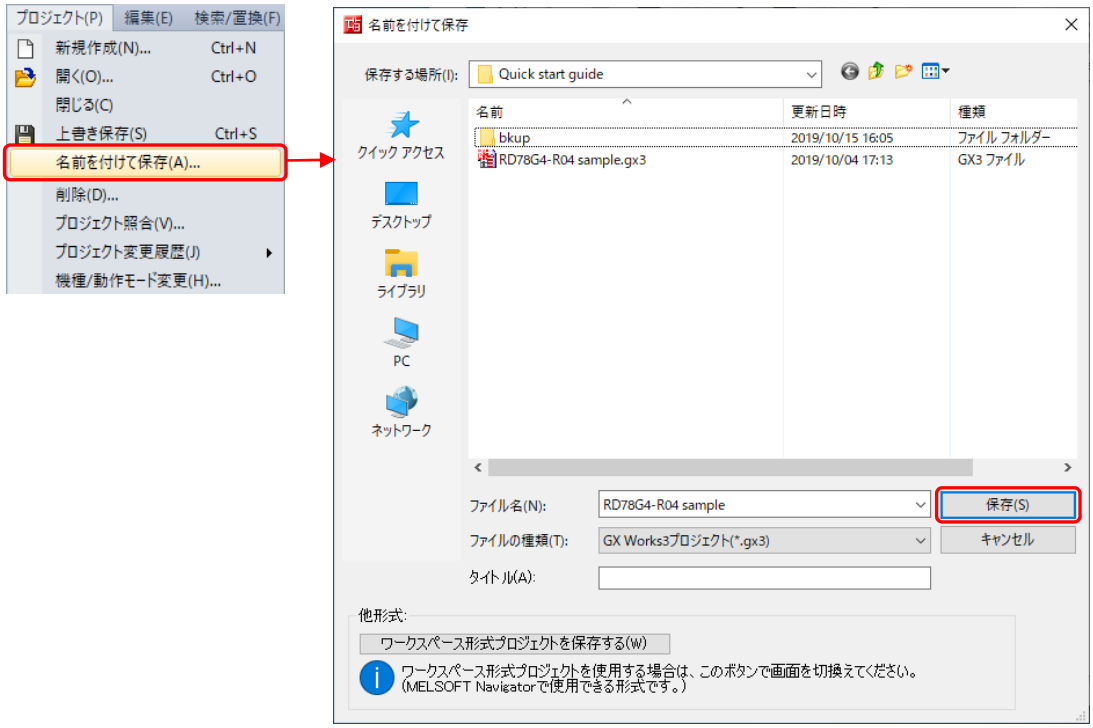



③再度，シーケンサCPUの電源をOFF→ONし，エラーが発生していないことを確認してください。
エラーは，シーケンサCPUやモーションユニットのLED表示，またはMELSOFT GX Works3のシステムモニタで確認できます。

## 3.16　プロジェクトの保存

作成したプロジェクトを保存します。

①MELSOFT GX Works3で[プロジェクト]⇒[名前を付けて保存]を選択して，“名前を付けて保存”画面を表示します。ファイル名を入力して，[保存]ボタンをクリックします。



## 3.17　パラメーター覧

本章で設定したパラメーター一覧を示します。
公開ラベルの設定は，第4章，または第5章を参照してください。

[軸パラメータ設定]
①実ドライブ軸

| 項目 | Axis0001 | Axis0002 |
|---|---|---|
| **軸情報** | | |
| 軸No. | 1 | 2 |
| **軸パラメータ定数** | 軸変数初期化時に設定値を展開します。軸変数初期化後に制御への | |
| 局アドレス設定 | 192.168.3.1 | 192.168.3.2 |
| 軸種別設定 | 0:実ドライブ軸 | 0:実ドライブ軸 |
| 上限リミット信号 | | |
| 信号 | | |
| 対象 | | |
| 信号検出方法 | 0:TRUE時検出 | 0:TRUE時検出 |
| 補正時間 | 0.0 s | 0.0 s |
| フィルタ時間 | 0.0 s | 0.0 s |
| 下限リミット信号 | | |
| 信号 | | |
| 対象 | | |
| 信号検出方法 | 0:TRUE時検出 | 0:TRUE時検出 |
| 補正時間 | 0.0 s | 0.0 s |
| フィルタ時間 | 0.0 s | 0.0 s |
| 制御周期設定 | 0:第1演算周期で動作 | 0:第1演算周期で動作 |
| 絶対位置基準設定 | 3:送り機械位置 | 3:送り機械位置 |
| 絶対位置管理設定 | -1:自動設定(接続機器から取得) | -1:自動設定(接続機器から取得) |
| リングカウンタ有効選択 | 0:無効 | 0:無効 |
| リングカウンタ下限値 | -100000000000.0 | -100000000000.0 |
| リングカウンタ上限値 | 100000000000.0 | 100000000000.0 |
| スレーブエミュレート有効 | 0:無効 | 0:無効 |
| トルク制限最大値 | 1000.0 % | 1000.0 % |
| 負方向トルク制限初期値 | 300.0 % | 300.0 % |
| 正方向トルク制限初期値 | 300.0 % | 300.0 % |
| 高速モード設定 | 0000 | 0000 |
| **軸パラメータ** | 軸変数初期化時に初期値を展開します。軸変数初期化後も制御への | |
| 加速度制限値 | 2147483647.0 um/s^2 | 2147483647.0 um/s^2 |
| 始動時加減速度0指定時動作選択 | -1:エラー(始動しない) | -1:エラー(始動しない) |
| 指令インポジション幅 | 100.0 um | 100.0 um |
| 減速度制限値 | 2147483647.0 um/s^2 | 2147483647.0 um/s^2 |
| ドライバ単位変換分子 | 67108864 pulse | 67108864 pulse |
| ドライバ単位変換分母 | 5000 um | 5000 um |
| 緊急停止信号 | | |
| 信号 | | |
| 対象 | | |
| 信号検出方法 | 0:TRUE時検出 | 0:TRUE時検出 |
| 補正時間 | 0.0 s | 0.0 s |
| フィルタ時間 | 0.0 s | 0.0 s |
| 原点復帰要否設定 | 1:原点復帰必要 | 1:原点復帰必要 |
| ジャーク制限値 | 2147483647.0 um/s^3 | 2147483647.0 um/s^3 |
| オーバラン時動作設定 | 1:即停止 | 1:即停止 |
| 原点復帰未完時始動許可 | 0:許可しない | 0:許可しない |
| 停止時減速度 | 0.0 um/s^2 | 0.0 um/s^2 |
| 減速停止時停止処理選択 | 1:減速カーブ再作成 | 1:減速カーブ再作成 |
| 停止要因発生時停止選択 | 3:加減速度を代替 | 3:加減速度を代替 |
| H/Wストロークリミットエラー発生時停止選択 | 1:即停止 | 1:即停止 |
| 運転中サーボOFF指令時処理選択 | 0:無視 | 0:無視 |
| S/Wストロークリミットエラー発生時停止選択 | 1:即停止 | 1:即停止 |
| ドライバ指令破棄検出設定 | 1:検出有効 | 1:検出有効 |
| 停止信号 | | |
| 信号 | | |
| 対象 | [VAR]G_bStopSignalX | [VAR]G_bStopSignalY |
| 信号検出方法 | 0:TRUE時検出 | 0:TRUE時検出 |
| 補正時間 | 0.0 s | 0.0 s |
| フィルタ時間 | 0.0 s | 0.0 s |
| ソフトウェアストロークリミット下限値 | -100000000000.0 um | -100000000000.0 um |
| ソフトウェアストロークリミット対象 | -1:無効 | -1:無効 |
| ソフトウェアストロークリミット上限値 | 100000000000.0 um | 100000000000.0 um |
| 位置指令単位 | um | um |
| 位置指令単位文字列 | | |
| 速度指令単位 | U/s | U/s |
| 負方向速度制限値 | 2500000000.0 um/s | 2500000000.0 um/s |
| 速度制限値オーバ時動作設定 | 0:無視 | 0:無視 |
| 正方向速度制限値 | 2500000000.0 um/s | 2500000000.0 um/s |

②仮想ドライブ軸

| 項目 | VirtualAxis0001 |
|---|---|
| **軸情報** | |
| 軸No. | 301 |
| **軸パラメータ定数** | **軸変数初期化時に設定値を展開し** |
| 軸種別設定 | 3:仮想ドライブ軸 |
| 上限リミット信号 | |
| 信号 | |
| 対象 | |
| 信号検出方法 | 0:TRUE時検出 |
| 補正時間 | 0.0 s |
| フィルタ時間 | 0.0 s |
| 下限リミット信号 | |
| 信号 | |
| 対象 | |
| 信号検出方法 | 0:TRUE時検出 |
| 補正時間 | 0.0 s |
| フィルタ時間 | 0.0 s |
| 制御周期設定 | 0:第1演算周期で動作 |
| 絶対位置管理設定 | 0:絶対位置システムを使用しない |
| リングカウンタ有効選択 | 0:無効 |
| リングカウンタ下限値 | -10000000000.0 |
| リングカウンタ上限値 | 10000000000.0 |
| 高速モード設定 | 0000 |
| **軸パラメータ** | **軸変数初期化時に初期値を展開し** |
| 加速度制限値 | 2147483647.0 um/s^2 |
| 始動時加減速度0指定時動作選択 | -1:エラー(始動しない) |
| 指令インポジション幅 | 100.0 um |
| 減速度制限値 | 2147483647.0 um/s^2 |
| 緊急停止信号 | |
| 信号 | |
| 対象 | |
| 信号検出方法 | 0:TRUE時検出 |
| 補正時間 | 0.0 s |
| フィルタ時間 | 0.0 s |
| 原点復帰要否設定 | 1:原点復帰必要 |
| ジャーク制限値 | 2147483647.0 um/s^3 |
| オーバラン時動作設定 | 1:即停止 |
| 原点復帰未完時始動許可 | 0:許可しない |
| 停止時減速度 | 0.0 um/s^2 |
| 減速停止時停止処理選択 | 1:減速カーブ再作成 |
| 停止要因発生時停止選択 | 3:加減速度を代替 |
| H/Wストロークリミットエラー発生時停止 | 1:即停止 |
| 運転中サーボOFF指令時処理選択 | 0:無視 |
| S/Wストロークリミットエラー発生時停止 | 1:即停止 |
| 停止信号 | |
| 信号 | |
| 対象 | |
| 信号検出方法 | 0:TRUE時検出 |
| 補正時間 | 0.0 s |
| フィルタ時間 | 0.0 s |
| ソフトウェアストロークリミット下限値 | -10000000000.0 um |
| ソフトウェアストロークリミット対象 | -1:無効 |
| ソフトウェアストロークリミット上限値 | 10000000000.0 um |
| 位置指令単位 | um |
| 位置指令単位文字列 | |
| 速度指令単位 | U/s |
| 負方向速度制限値 | 2500000000.0 um/s |
| 速度制限値オーバ時動作設定 | 0:無視 |
| 正方向速度制限値 | 2500000000.0 um/s |

③仮想連結軸

| 項目 | LinkAxis0001 | LinkAxis0002 |
|---|---|---|
| **軸情報** | | |
| 軸No. | 401 | 402 |
| **軸パラメータ定数** | **軸変数初期化時に設定値を展開します。軸変数初期化後に制御への** | |
| 軸種別設定 | 5:仮想連結軸 | 5:仮想連結軸 |
| 制御周期設定 | 0:第1演算周期で動作 | 0:第1演算周期で動作 |
| 絶対位置管理設定 | 0:絶対位置システムを使用しない | 0:絶対位置システムを使用しない |
| リングカウンタ有効選択 | 0:無効 | 0:無効 |
| リングカウンタ下限値 | -10000000000.0 | -10000000000.0 |
| リングカウンタ上限値 | 10000000000.0 | 10000000000.0 |
| 高速モード設定 | 0000 | 0000 |
| **軸パラメータ** | **軸変数初期化時に初期値を展開します。軸変数初期化後も制御への** | |
| 原点復帰要否設定 | 0:原点復帰不要 | 0:原点復帰不要 |
| 原点復帰未完時始動許可 | 0:許可しない | 0:許可しない |
| 位置指令単位 | um | um |
| 位置指令単位文字列 | | |
| 速度指令単位 | U/s | U/s |

[軸グループ設定]

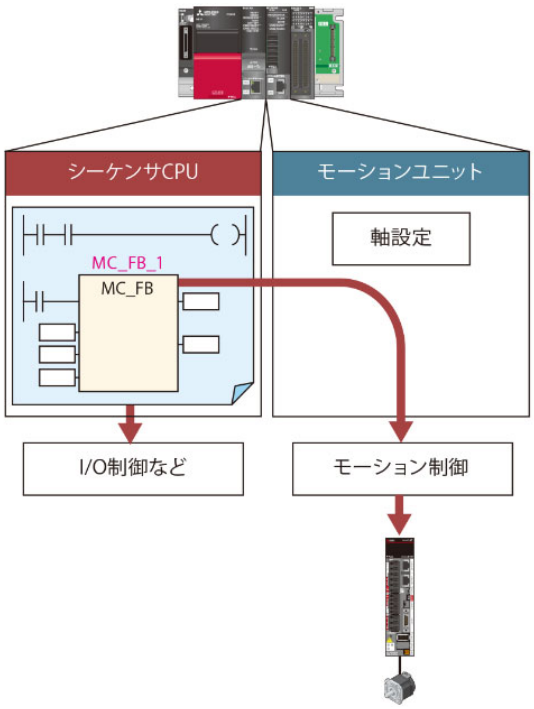
| 項目 | AxesGroup001 |
|---|---|
| **軸グループ情報** | |
| 軸グループNo. | 1 |
| **軸グループパラメータ** | **軸グループ変数生成時に初期** |
| 加速度制限値 | 2147483647.0 um/s^2 |
| 始動時加減速度0指定時動作選択 | -1:エラー(始動しない) |
| 構成軸[1] | Axis0001 |
| 構成軸[2] | Axis0002 |
| 構成軸[3] | |
| 構成軸[4] | |
| 構成軸[5] | |
| 構成軸[6] | |
| 構成軸[7] | |
| 構成軸[8] | |
| 構成軸[9] | |
| 構成軸[10] | |
| 構成軸[11] | |
| 構成軸[12] | |
| 構成軸[13] | |
| 構成軸[14] | |
| 構成軸[15] | |
| 構成軸[16] | |
| 指令インポジション幅 | 100.0 um |
| 減速度制限値 | 2147483647.0 um/s^2 |
| ジャーク制限値 | 2147483647.0 um/s^3 |
| オーバラン時動作設定 | 1:即停止 |
| 停止時減速度 | 0.0 um/s^2 |
| 減速停止時停止処理選択 | 1:減速カーブ再作成 |
| 軸停止要因発生時構成軸動作選択 | 1:即停止 |
| 停止要因発生時停止選択 | 3:加減速度を代替 |
| 位置指令単位 | um |
| 位置指令単位文字列 | |
| 速度指令単位 | U/s |
| 速度制限値 | 2500000000.0 um/s |

# メ　モ

# 第４章　シーケンサCPUでのプログラミング

シーケンサCPUは，FBD/LD，ラダー，ST言語でプログラミングできます。
本章では，FB間の関連がわかりやすいFBD/LD言語を使って説明します。
本章で説明するサンプルプログラムは，三菱電機FAサイトよりダウンロードできます。

■ファイル名：R04-78G4_FBDsample_*****.gx3　（*****にはバージョンが入ります。）




本サンプルプログラムを実際のシステムへ流用するときは，対象システムにおいて，制御に問題ないことを十分検証してください。
また，対象システムにおいて，インタロック条件が必要な箇所を検討し追加してください。

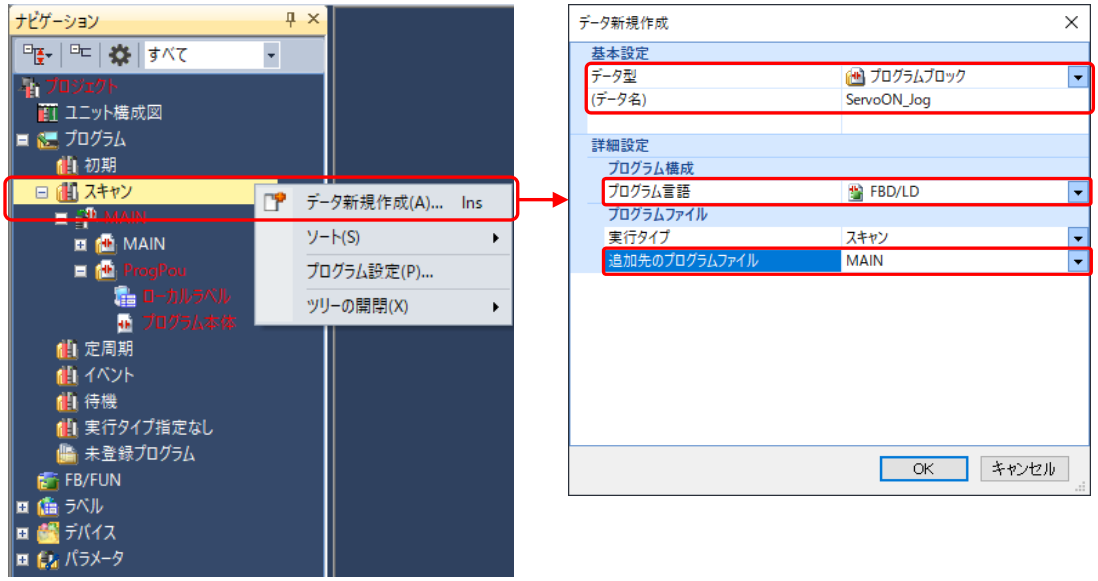
## 4.1　シーケンサCPUのプログラム作成手順

MELSOFT GX Works3で，シーケンサCPUのプログラミングを行ないます。

### 4.1.1　プログラムブロックの作成

①MELSOFT GX Works3のナビゲーションウィンドウで，“プログラム”⇒“プログラム実行タイプ（画面上：スキャン）”を右クリックして，“データ新規作成”をクリックします。

②“データ新規作成”画面で，データ名，プログラム言語，追加先のプログラムファイルを設定します。データ名には，半角英数字のみ使用可能です。




③ナビゲーションウィンドウにプログラムブロックが追加されます。

### 4.1.2 プログラム実行タイプ

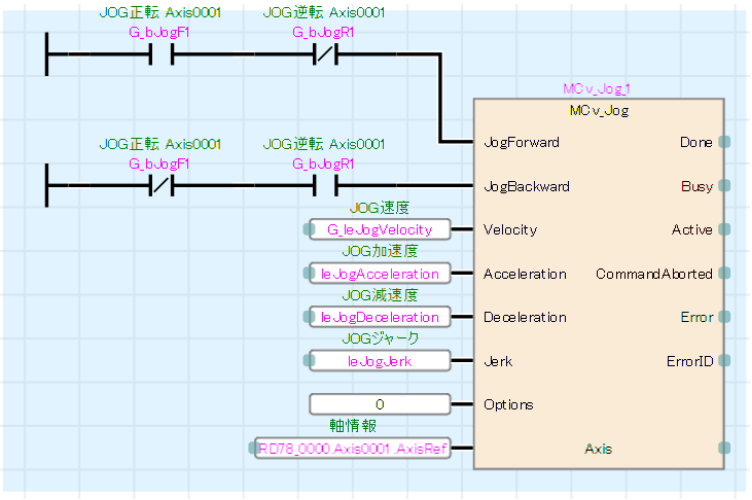
プログラムの実行タイプには，以下の5種類があります。

| 実行タイプ | 内　容 |
|---|---|
| 初期 | シーケンサ CPU ユニットの電源 OFF→ON，または STOP 状態から RUN 状態に切り換えたときに 1 回のみ実行 |
| スキャン | 初期実行タイププログラムを実行した次のスキャンから 1 スキャンに 1 回のみ実行 |
| 定周期 | 指定時間ごとに実行する割込みプログラム<br>ただし，通常の割込みプログラムとは異なり，割込みポインタ(I)や IRET 命令を記述する必要がなく，プログラムファイル単位で実行 |
| イベント | 指定した事象をトリガとして実行を開始 |
| 待機 | 実行要求があった場合のみ実行 |

ナビゲーションウィンドウで，実行タイプを設定するプログラムを右クリック⇒ショートカットメニュー[プログラム登録]から選択，またはドラッグ＆ドロップして，プログラムの実行タイプを設定します。
設定した実行タイプは，“CPUパラメータ”の“プログラム設定”に反映されます。

### 4.1.3 FBの入力方法

MELSOFT GX Works3にモーションユニットFBライブラリを登録します。（2.4.4項参照）
本項では，JOG運転用のFB(MCv_Jog)を例に，以下のプログラムを作成する手順を説明します。
サンプルプログラムは，このプログラムからインタロック条件などを追加します。




#### (1) ラベルの準備

FBの入出力信号に使用するラベルを準備します。本項では，MCv_Jogの入力信号の速度，加速度，減速度，ジャークにラベルを使用します。各ラベルを，グローバルラベルに登録するか，ローカルラベルに登録するかをあわせて検討します。

本項では例として，グローバルラベルには，JOG指令，JOG速度を登録します。JOG速度は，GOTなどの外部機器から設定するケースを想定してグローバルラベルに登録します。このプログラム内でのみ使用するJOG加速度，JOG減速度，JOGジャークなどは，4.1.1項で作成したプログラム“ServoON_Jog”のローカルラベルに登録します。
ラベルの命名規則は，4.2節を参照してください。

[グローバルラベル]

Global [グローバルラベル設定]

<フィルタ> 詳細表示(Y) 表示設定(S) チェック(K)

| | ラベル名 | データ型 | | Japanese/日本語(表示対象) | 外部機器からのアクセス |
|---|---|---|---|---|---|
| 1 | G_bJogF1 | ビット | ... | JOG正転 Axis0001 | □ |
| 2 | G_bJogR1 | ビット | ... | JOG逆転 Axis0001 | □ |
| 3 | G_leJogVelocity | 倍精度実数 | ... | JOG速度 | □ |
| 4 | | | ... | | □ |

[ローカルラベル]

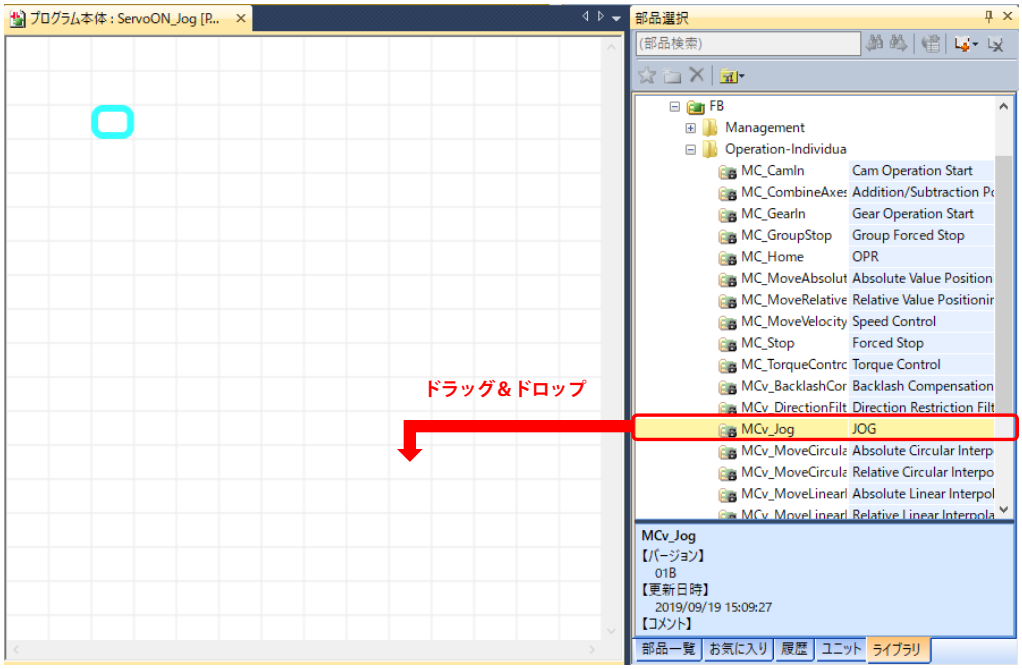
ServoON_Jog [PRG] [ローカルラベ...

<フィルタ> 詳細表示(Y) 表示設定(S) チェック(K)

| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | leJogAcceleration | 倍精度実数 | ... | JOG加速度 |
| 2 | leJogDeceleration | 倍精度実数 | ... | JOG減速度 |
| 3 | leJogJerk | 倍精度実数 | ... | JOGジャーク |
| 4 | | | ... | |

(2) FBの入力

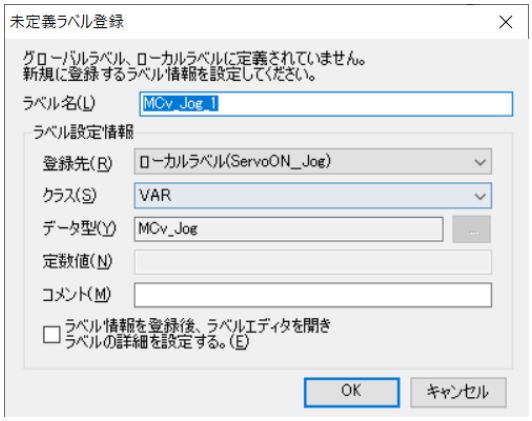
部品選択ウィンドウの[ライブラリ]タブで“ライブラリ”⇒“MotionControl_RD78G_****”⇒“FB”を選択すると，モーションユニットFBライブラリが登録されています。
使用するFBをプログラムエディタにドラッグ＆ドロップします。



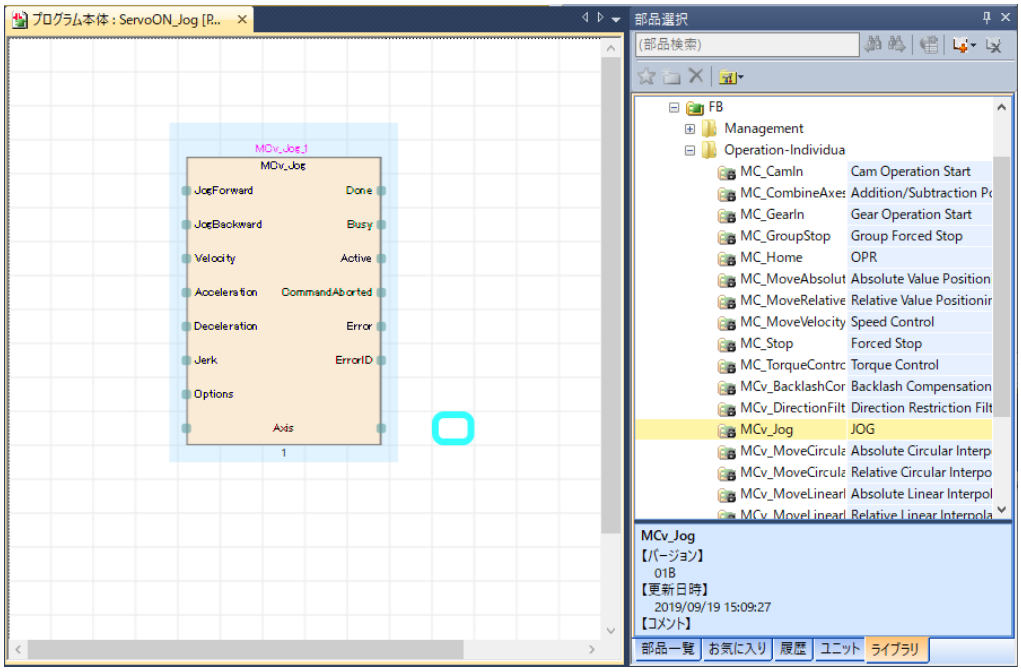

モーションユニットFBは，以下の3つのグループに分類されています。

| グループ | 内　容 |
|---|---|
| Management<br>(管理系) | 軸または軸グループを引数にとり，実行によって軸状態，または軸グループ状態が変化しないモーション制御 FB |
| Operation-Individual<br>(動作系・単一) | 軸または軸グループを引数にとり，実行によって軸状態，または軸グループ状態が変化するモーション制御 FB |
| StandardFB<br>(一般 FB) | 軸または軸グループを引数にとらないモーション制御 FB |

JOG運転用のMCv_Jogは，“Operation-Individual”のツリー下にあります。
MCv_Jogをドラッグ＆ドロップして，“未定義ラベル登録”画面を表示し，FBのラベル名と登録先，および必要に応じてコメントを入力します。ここでは，全て初期値とします。



[OK]ボタンをクリックして，プログラムエディタ上に[MCv_Jog]を表示します。




### (3) 入出力信号の入力

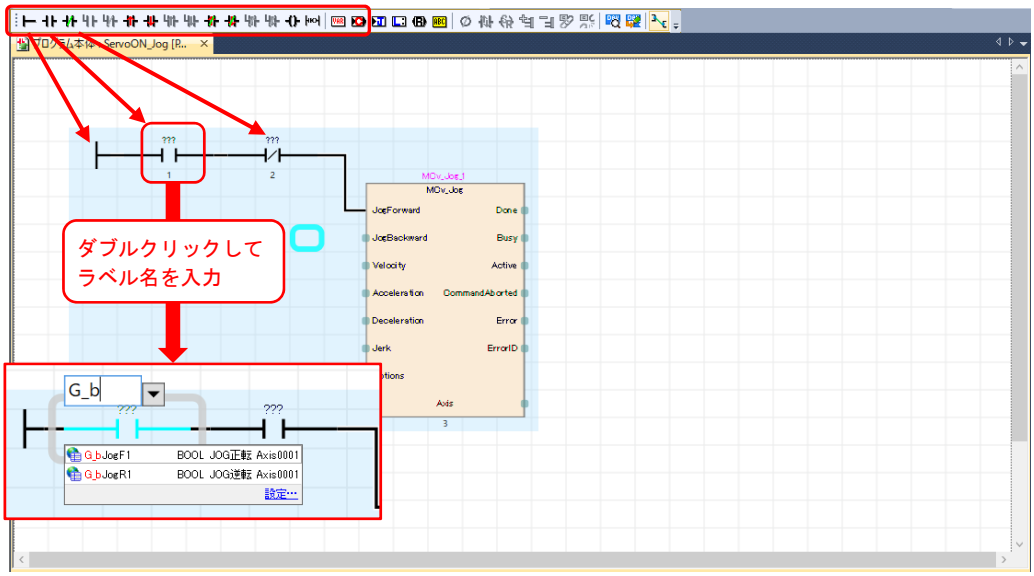
FBD部品やLD部品を，FBの入力接続ポイントに接続します。FBD部品やLD部品は，配置後にドラッグして移動させます。

(a) LD部品

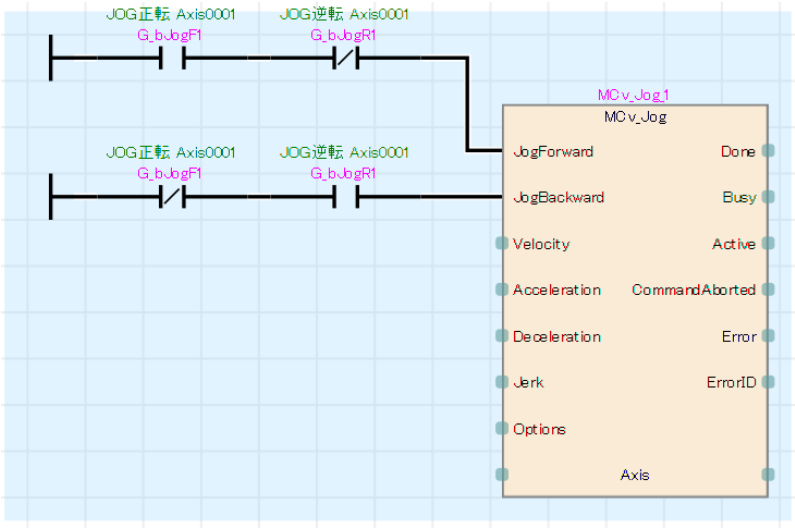
プログラムエディタ上のLD部品を配置する場所をクリックし，ツールバーのLD部品の各アイコンをクリックして，指定したLD部品を配置します。

接点部品は「???」となっているため，ダブルクリックして，ビットデバイスのラベル名，またはビットデバイス番号を入力します。グローバルラベルに登録済のラベルは，候補として表示されます。

入力後，LD部品をFBの入力接続ポイントに接続します。



JogForward入力とJogBackward入力の各接続ポイントにLD部品を接続すると，下図のようになります。

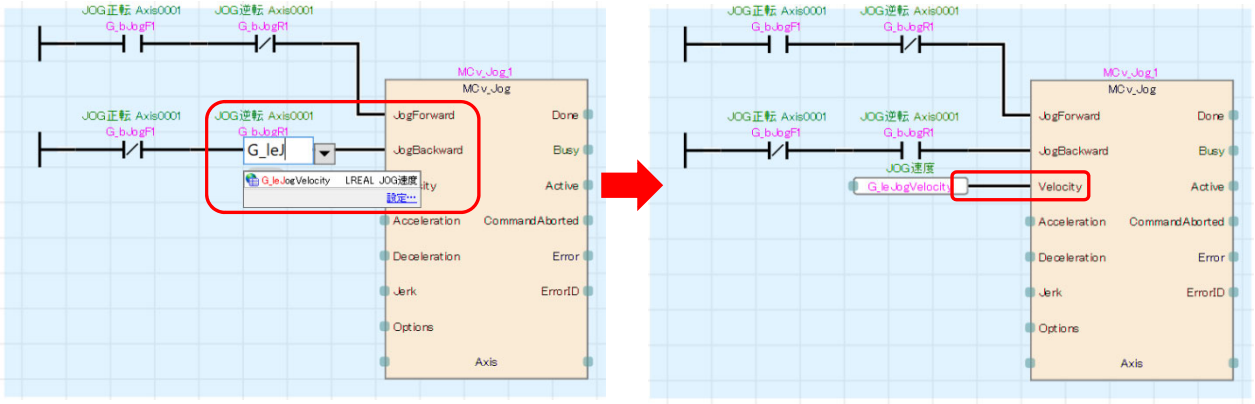



(b) FBD部品

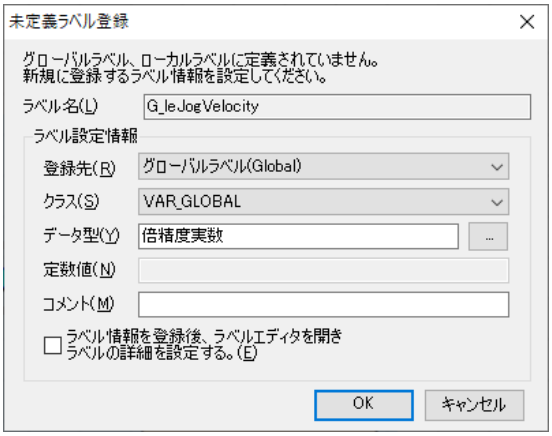
FBD部品は，プログラムエディタ上に配置する場所をクリックし，デバイス番号，またはラベル名を直接入力して配置します。

MCv_JogのVelocity入力に，(1)で登録したグローバルラベルの「G_leJogVelocity」を入力する場合は，MCv_JogのVelocityの左側をクリックし，「G_leJogVelocity」を入力します。グローバルラベルに登録済のラベルは，候補として表示されます。

入力後，FBのVelocity入力接続ポイントに接続します。

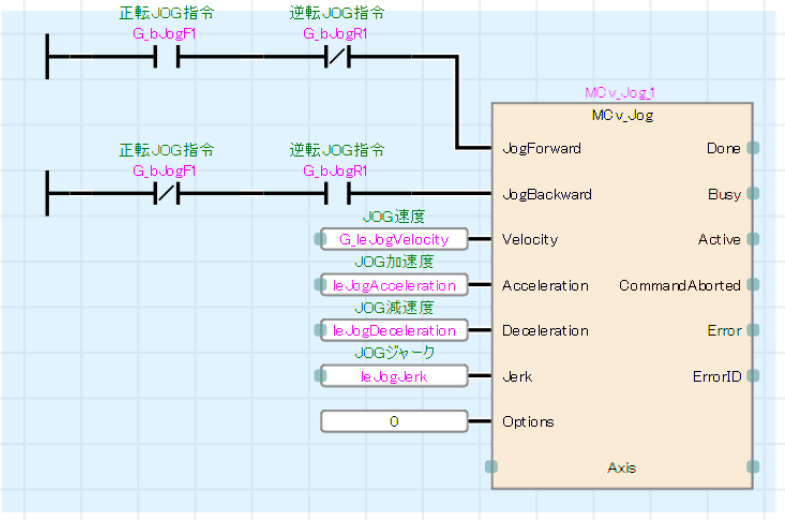


未定義のラベルを入力した場合は，“未定義ラベル登録”画面が表示されます。
データ型と登録先を設定してください。




同様に，(1)で登録したローカルラベルのleJogAcceleration，leJogDeceleration，leJogJerkを，MCv_JogのAcceleration入力，Deceleration入力，Jerk入力接続ポイントに接続します。Options入力接続ポイントには定数の「0」を接続します。

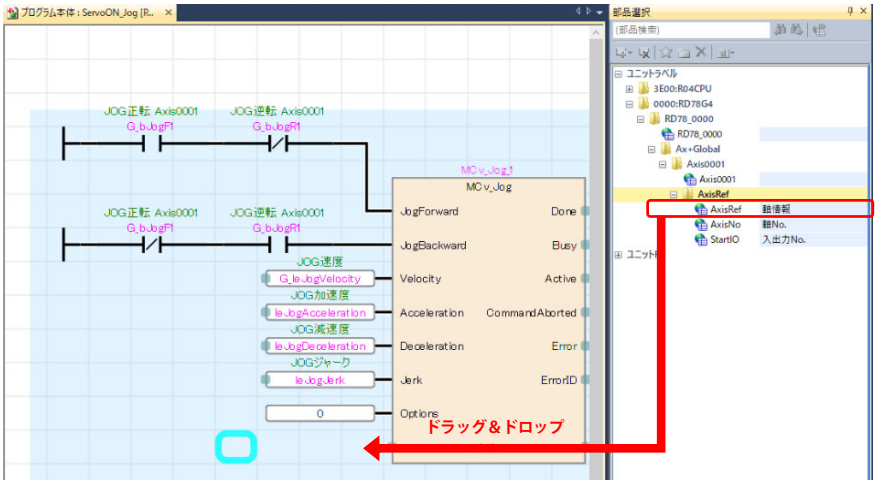
**[注意事項]**
G_leJogVelocity，leJogAccelertion，leJogDeceleration，leJogJerkには，別のプログラム(EDMOVP命令など)で数値を入力する必要があります。（設定例は，4.6節参照）



(c) Axis入力(公開ラベル)

Axis入力接続ポイントには，公開ラベルに登録したAxisRef型の構造体を接続します。(3.14.1項参照)

部品選択ウィンドウの[ユニット]タブをクリックします。“ユニットラベル”⇒“0000:RD78G4”⇒“RD78_0000”⇒“Ax+Global”⇒“Axis0001”⇒“AxisRef”を選択し，AxisRef(軸情報)をプログラムエディタ上にドラッグ＆ドロップします。



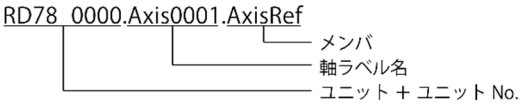

“RD78G_0000.Axis0001.AxisRef”がプログラム上に表示されたら，FBのAxis入力接続ポイントに接続します。

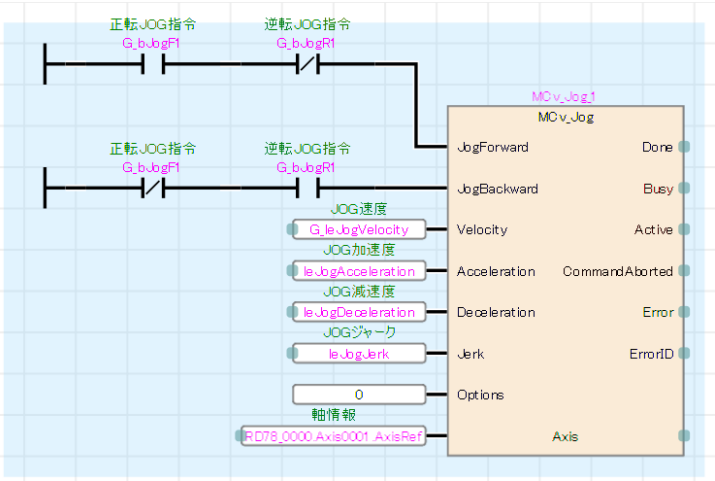
**[ポイント]**

公開ラベルの名称“RD78_0000.Axis0001.AxisRef”は，直接入力することもできます。

“RD78”と入力すると，“RD78_0000”が候補として表示されます。“RD78_0000”選択後に“.”を入力すると，構造体のメンバ“Axis0001”などが表示されます。

同様に，“Axis0001”を選択後に“.”を入力すると，構造体のメンバ“AxisRef”などが表示されます。“AxisRef”を選択すると入力が完了します。



以上で，プログラムの入力は完了です。

**[ポイント]**
FBの入力信号を初期値から変更しないで使用する場合は，入力を省略できます。
本項の例では，Options入力の初期値が“0”のため，省略可能です。

## 4.2 ラベルの命名規則

本書では，プログラム例に使用するラベルに接頭語を付けてデータ型を表現しています。

| データ型 | | 値の範囲 | 接頭語 | |
|---|---|---|---|---|
| | | | ローカル | グローバル |
| ビット | BOOL | 0 (FALSE), 1 (TRUE) | b | G_b |
| ワード［符号なし］<br>ビット列［16 ビット］ | WORD | 0～65535 | u | G_u |
| ダブルワード［符号なし］<br>ビット列［32 ビット］ | DWORD | 0～4294967295 | ud | G_ud |
| ワード［符号付き］ | INT | -32468～32767 | w | G_w |
| ダブルワード［符号付き］ | DINT | -2147483648～2147483647 | d | G_d |
| 単精度実数 | REAL | $-2^{128}$～$-2^{-126}$, 0, $2^{-126}$～$2^{128}$ | e | G_e |
| 倍精度実数 | LREAL | $-2^{1024}$～$-2^{-1022}$, 0, $2^{-1022}$～$2^{1024}$ | le | G_le |
| 時間 | TIME | T#-24d20h31m23s648ms～<br>T#24d20h31m23s647ms | tm | G_tm |
| タイマ | TIMER | TIMER の構造体<br>S：接点<br>C：コイル<br>N：現在値 | td | G_td |

［ローカルラベル例］

ビット　bMoveCMD
倍精度実数　lePosition
配列　wAxes[16]
タイマ　tdTimer1

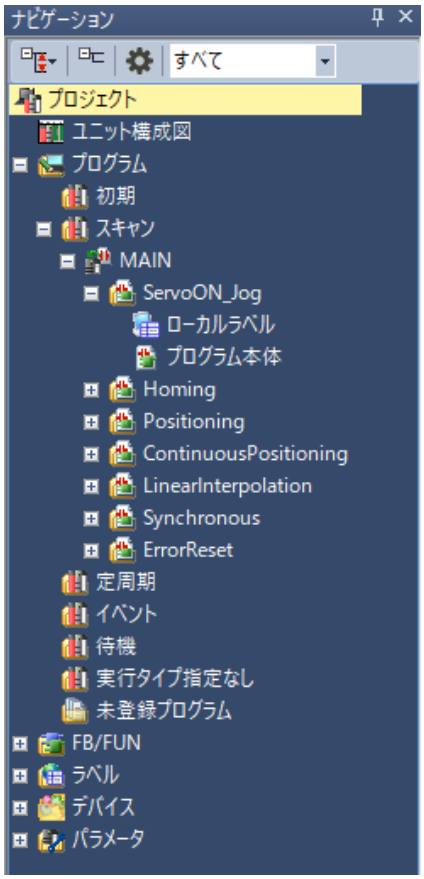
［グローバルラベル例］

ビット　G_bJogF1
倍精度実数　G_leVelocity

## 4.3　プロジェクト構成

### 4.3.1　プログラム名

本章で作成するプログラム例を以下に示します。




“スキャン”→“MAIN”のツリー下

プログラム名：ServoON_Jog
　全軸サーボON，JOG運転

プログラム名：Homing
　原点復帰

プログラム名：Positioning
　単軸位置決め制御

プログラム名：ContinuousPositioning
　単軸の連続位置決め制御

プログラム名：LinearInterpolation
　2軸の直線補間制御

プログラム名：Synchronous
　同期制御

プログラム名：ErrorReset
　エラーリセット

### 4.3.2　グローバルラベル，公開ラベル設定

サンプルプログラムのグローバルラベルの設定例を示します。
ローカルラベルについては各プログラムを参照してください。

(1) シーケンサCPU

| | ラベル名 | データ型 | | クラス | 割付け | 初期値 | 定数 | Japanese/日本語(表示対象) |
|---|---|---|---|---|---|---|---|---|
| 1 | G_bSRVOFF | ビット | … | VAR_GLOBAL | | | | サーボOFF |
| 2 | G_bJogF1 | ビット | … | VAR_GLOBAL | | | | JOG正転指令 Axis0001 |
| 3 | G_bJogR1 | ビット | … | VAR_GLOBAL | | | | JOG逆転指令 Axis0001 |
| 4 | G_bJogF2 | ビット | … | VAR_GLOBAL | | | | JOG正転指令 Axis0002 |
| 5 | G_bJogR2 | ビット | … | VAR_GLOBAL | | | | JOG逆転指令 Axis0002 |
| 6 | G_bHoming1CMD | ビット | … | VAR_GLOBAL | | | | 原点復帰指令 Axis0001 |
| 7 | G_bHoming2CMD | ビット | … | VAR_GLOBAL | | | | 原点復帰指令 Axis0002 |
| 8 | G_bHoming3CMD | ビット | … | VAR_GLOBAL | | | | 原点復帰指令 VirtualAxis0001 |
| 9 | G_bPosCMD | ビット | … | VAR_GLOBAL | | | | 単軸位置決め始動 |
| 10 | G_bContPosCMD | ビット | … | VAR_GLOBAL | | | | 単軸連続位置決め始動 |
| 11 | G_bInterpolationCMD | ビット | … | VAR_GLOBAL | | | | 2軸直線補間制御始動 |
| 12 | G_bSyncCMD | ビット | … | VAR_GLOBAL | | | | 同期制御始動 |
| 13 | G_bErrorReset | ビット | … | VAR_GLOBAL | | | | エラーリセット |
| 14 | G_bSysErrorReset | ビット | … | VAR_GLOBAL | | | | システムエラーリセット |
| 15 | G_leJogVelocity | 倍精度実数 | … | VAR_GLOBAL | | | | JOG速度 |
| 16 | G_bHoming1Req | ビット | … | VAR_GLOBAL | | | | 原点復帰始動要求 Axis0001 |
| 17 | G_bHoming2Req | ビット | … | VAR_GLOBAL | | | | 原点復帰始動要求 Axis0002 |
| 18 | G_bHoming3Req | ビット | … | VAR_GLOBAL | | | | 原点復帰始動要求 VirtualAxis0001 |
| 19 | G_bPosReq | ビット | … | VAR_GLOBAL | | | | 単軸位置決め始動要求 |
| 20 | G_bContPosReq | ビット | … | VAR_GLOBAL | | | | 単軸連続位置決め始動要求 |
| 21 | G_bInterpolationReq | ビット | … | VAR_GLOBAL | | | | 2軸直線補間制御始動要求 |
| 22 | G_bSyncReq | ビット | … | VAR_GLOBAL | | | | 同期制御始動要求 |
| 23 | G_bJog1Busy | ビット | … | VAR_GLOBAL | | | | JOG運転中 Axis0001 |
| 24 | G_bJog2Busy | ビット | … | VAR_GLOBAL | | | | JOG運転中 Axis0002 |
| 25 | | | … | | | | | |

①各プログラムの実行指令となるラベル
②JOG速度を格納する倍精度実数型のラベル
　グローバルラベルに登録することで，GOTや別のプログラムから変更することができます。
③各プログラムの動作中にONとなるラベル
　1つのプログラムの動作中に，別のプログラムが動作しないようにするためのインタロックとして使用します。

(2) モーションユニット

モーションユニット側のグローバルラベル設定，公開ラベル設定は以下のとおりです。
モーション制御設定機能で設定してください。設定後，公開ラベルの反映を行なってください。（3.14.2項参照）

[AX+Global]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Axis0001 | AXIS_REAL | VAR_GLOBAL | <詳細設… | | [X_Axis] | | 有効 | - |
| 2 | Axis0002 | AXIS_REAL | VAR_GLOBAL | <詳細設… | | [Y_Axis] | | 有効 | - |
| 3 | VirtualAxis0001 | AXIS_VIRTUAL | VAR_GLOBAL | <詳細設… | | [Vir_Axis01] | | 有効 | - |
| 4 | LinkAxis0001 | AXIS_VIRTUAL_LINK | VAR_GLOBAL | <詳細設… | | [Lin_Axis01] | | 有効 | - |
| 5 | LinkAxis0002 | AXIS_VIRTUAL_LINK | VAR_GLOBAL | <詳細設… | | [Lin_Axis02] | | 有効 | - |
| 6 | | | | | | | | | |

[Gr+Global]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | AxesGroup001 | AXES_GROUP | VAR_GLOBAL | <詳細設… | | X-Y Table | | 有効 | - |
| 2 | | | | | | | | | |

[Global]
システムに関するデータを格納する構造体です。公開ラベルを有効に設定します。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | G_bStopSignalX | ビット | VAR_GLOBAL | | | 停止指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 2 | G_bStopSignalY | ビット | VAR_GLOBAL | | | 停止指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 3 | | | | | | | | | |

[Sys+Global]
システムに関するデータを格納する構造体です。公開ラベルを有効に設定します。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | System | SYSTEM | VAR_GLOBAL | <詳細… | | | | 有効 | － |
| 2 | | | | | | | | | |

[AXIS_REAL構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AxisRef | AXIS_REF | | | | 軸情報 | 有効 | READ(モーション⇒): MdConst |
| 2 | PrConst | AXIS_REAL_PRM_CONST | | | | 軸パラメータ定数 | 無効 | WRITE(⇒モーション): PrConst |
| 3 | Pr | AXIS_REAL_PRM | | | | 軸パラメータ | 無効 | WRITE(⇒モーション): Pr |
| 4 | Md | AXIS_REAL_MONI | | | | 軸モニタデータ | 有効 | READ(モーション⇒): Md |
| 5 | Cd | AXIS_REAL_CMD | | | | 軸制御データ | 無効 | WRITE(⇒モーション): Cd |
| 6 | | | | | | | | |

[AXIS_REAL_MONI構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AccelerationLimit | 倍精度実数 | | 0.0 | | 加速度制限値 | 無効 | － |
| 2 | AccelerationOverride | 倍精度実数 | | 1.0 | | 加速度オーバライド係数 | 無効 | － |
| 3 | AccelerationZeroBehavior | ワード[符号付き] | | -1 | | 始動時加減速度0指定時動作選択 | 無効 | － |
| 4 | ActualPosition | 倍精度実数 | | 0.0 | | フィードバック位置 | 無効 | － |
| 5 | ActualVelocity | 倍精度実数 | | 0.0 | | フィードバック速度 | 無効 | － |
| 6 | Analyzing | ビット | | 0 | | 解析中 | 無効 | － |
| 7 | AutoDeceleration | ビット | | 0 | | 自動減速中 | 無効 | － |
| 8 | AxisName | 文字列[Unicode](127) | | "" | | 軸名称 | 無効 | － |
| 9 | AxisStatus | ワード[符号付き] | | 0 | | 軸状態 | 有効 | － |
| 10 | BufferingFBs | ワード[符号付き] | | 0 | | バッファリングFB数 | 無効 | － |
| 11 | CmdInPos | ビット | | 0 | | 指令インポジション | 有効 | － |
| 12 | CmdInPos_Width | 倍精度実数 | | 100.0 | | 指令インポジション幅 | 無効 | － |
| 13 | CommandedAcceleration | 倍精度実数 | | 0.0 | | 指定加速度 | 無効 | － |

≈ ≈

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 26 | Driver_Mode | ワード[符号付き] | | 0 | | ドライバ制御モード | 無効 | － |
| 27 | Driver_ReadyOn | ビット | | 0 | | ドライバレディON状態 | 無効 | － |
| 28 | Driver_ServoOn | ビット | | 0 | | ドライバサーボON状態 | 有効 | － |
| 29 | Driver_State | ワード[符号付き] | | 0 | | ドライバ状態 | 無効 | － |
| 30 | DriverError | ビット | | 0 | | ドライブユニットエラー検出 | 無効 | － |
| 31 | DriverErrorID | ワード[符号なし]/ビッ… | | 0 | | ドライブユニットエラーコード | 無効 | － |
| 32 | DriverErrorDetailID | ワード[符号なし]/ビッ… | | 0 | | ドライブユニットエラー詳細コード | 無効 | － |
| 33 | Error | ビット | | 0 | | 軸エラー検出 | 無効 | － |
| 34 | ErrorID | ワード[符号なし]/ビッ… | | 0 | | 軸エラーコード | 無効 | － |
| 35 | FeedMachinePosition | 倍精度実数 | | 0.0 | | 送り機械位置 | 無効 | － |
| 36 | FollowupDisable | ビット | | 0 | | フォローアップ無効中 | 無効 | － |
| 37 | ForcedStop_Released | ビット | | 0 | | 緊急停止解除中 | 無効 | － |
| 38 | ForcedStop_Signal | SIGNAL_SELECT | | | | 緊急停止信号 | 無効 | － |
| 39 | Homing_Complete | ビット | | 0 | | 原点復帰完了 | 無効 | － |
| 40 | Homing_Request | ビット | | 1 | | 原点復帰要求 | 有効 | － |
| 41 | Homing_Required | ビット | | 0 | | 原点復帰要否設定 | 無効 | － |
| 42 | Homing_Status | ワード[符号なし]/ビッ… | | 65535 | | 原点復帰動作状態 | 無効 | － |

≈ ≈

[AXES_GROUP構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AxesGroupRef | AXES_GROUP_REF | | | | 軸グループ情報 | 有効 | READ(モーション⇒):MdConst |
| 2 | Pr | AXES_GROUP_PRM | | | | 軸グループパラメータ | 無効 | WRITE(⇒モーション):Pr |
| 3 | Md | AXES_GROUP_MONI | | | | 軸グループモニタデータ | 有効 | READ(モーション⇒):Md |
| 4 | Cd | AXES_GROUP_CMD | | | | 軸グループ制御データ | 無効 | WRITE(⇒モーション):Cd |
| 5 | | | | | | | | |

[AXES_GROUP_MONI構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AccelerationLimit | 倍精度実数 | | 0.0 | | 加速度制限値 | 無効 | - |
| 2 | AccelerationOverride | 倍精度実数 | | 1.0 | | 加速度オーバライド係数 | 無効 | - |
| 3 | AccelerationZeroBehavior | ワード[符号付き] | | -1 | | 始動時加減速度0指定時動作選択 | 無効 | - |
| 4 | ActualVelocity | 倍精度実数 | | 0.0 | | フィードバック速度 | 無効 | - |
| 5 | Analyzing | ビット | | 0 | | 解析中 | 無効 | - |
| 6 | AutoDeceleration | ビット | | 0 | | 自動減速中 | 無効 | - |
| 7 | Axis | AXIS_REF(1..16) | | | | 構成軸 | 無効 | - |
| 8 | BufferingFBs | ワード[符号付き] | | 0 | | バッファリングFB数 | 無効 | - |
| 9 | CmdInPos | ビット | | 0 | | 指令インポジション | 有効 | - |
| 10 | CmdInPos_Width | 倍精度実数 | | 100.0 | | 指令インポジション幅 | 無効 | - |
| 11 | CommandedAcceleration | 倍精度実数 | | 0.0 | | 指定加速度 | 無効 | - |
| 12 | CommandedDeceleration | 倍精度実数 | | 0.0 | | 指定減速度 | 無効 | - |
| 13 | CommandedJerk | 倍精度実数 | | 0.0 | | 指定ジャーク | 無効 | - |
| 14 | CommandedVelocity | 倍精度実数 | | 0.0 | | 指定速度 | 無効 | - |
| 15 | DecelerationLimit | 倍精度実数 | | 0.0 | | 減速度制限値 | 無効 | - |
| 16 | Error | ビット | | 0 | | 軸グループエラー検出 | 無効 | - |
| 17 | ErrorID | ワード[符号なし]/ビ… | | 0 | | 軸グループエラーコード | 無効 | - |
| 18 | GroupName | 文字列[Unicode](1… | | "" | | 軸グループ名称 | 無効 | - |
| 19 | GroupStatus | ワード[符号付き] | | 0 | | 軸グループ状態 | 有効 | - |
| 20 | InterpolationAxes | ダブルワード[符号な… | | 0 | | 補間軸 | 無効 | - |

≈ ≈

[AXIS_VIRTUAL構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AxisRef | AXIS_REF | | | | 軸情報 | 有効 | READ(モーション⇒):MdConst |
| 2 | PrConst | AXIS_VIRTUAL_PRM_CONST | | | | 軸パラメータ定数 | 無効 | WRITE(⇒モーション):PrConst |
| 3 | Pr | AXIS_VIRTUAL_PRM | | | | 軸パラメータ | 無効 | WRITE(⇒モーション):Pr |
| 4 | Md | AXIS_VIRTUAL_MONI | | | | 軸モニタデータ | 有効 | READ(モーション⇒):Md |
| 5 | Cd | AXIS_VIRTUAL_CMD | | | | 軸制御データ | 無効 | WRITE(⇒モーション):Cd |
| 6 | | | | | | | | |

[AXIS_VIRTUAL_MONI構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AccelerationLimit | 倍精度実数 | | 0.0 | | 加速度制限値 | 無効 | - |
| 2 | AccelerationOverride | 倍精度実数 | | 1.0 | | 加速度オーバライド係数 | 無効 | - |
| 3 | AccelerationZeroBehavior | ワード[符号付き] | | -1 | | 始動時加減速度0指定時動作選択 | 無効 | - |
| 4 | Analyzing | ビット | | 0 | | 解析中 | 無効 | - |
| 5 | AutoDeceleration | ビット | | 0 | | 自動減速中 | 無効 | - |
| 6 | AxisName | 文字列[Unicode]… | | "" | | 軸名称 | 無効 | - |
| 7 | AxisStatus | ワード[符号付き] | | 0 | | 軸状態 | 有効 | - |
| 8 | BufferingFBs | ワード[符号付き] | | 0 | | バッファリングFB数 | 無効 | - |
| 9 | CmdInPos | ビット | | 0 | | 指令インポジション | 有効 | - |
| 10 | CmdInPos_Width | 倍精度実数 | | 100.0 | | 指令インポジション幅 | 無効 | - |
| 11 | CommandedAcceleration | 倍精度実数 | | 0.0 | | 指定加速度 | 無効 | - |

≈ ≈

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 21 | ForcedStop_Signal | SIGNAL_SELECT | | | | 緊急停止信号 | 無効 | - |
| 22 | Homing_Complete | ビット | | 0 | | 原点復帰完了 | 無効 | - |
| 23 | Homing_Request | ビット | | 1 | | 原点復帰要求 | 有効 | - |
| 24 | Homing_Required | ビット | | 0 | | 原点復帰要否設定 | 無効 | - |
| 25 | HwStrokeLimit_FlsStatus | ビット | | 0 | | 上限リミット信号状態 | 無効 | - |
| 26 | HwStrokeLimit_Override | 文字列(15) | | "" | | ハードウェアストロークリミットオーバ… | 無効 | - |

≈ ≈

[AXIS_VIRTUA_LINK構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | AxisRef | AXIS_REF | | | | 軸情報 | 有効 | READ(モーション⇒): MdConst |
| 2 | PrConst | AXIS_VIRTUAL_LINK_PRM_CONST | | | | 軸パラメータ定数 | 無効 | WRITE(⇒モーション): PrConst |
| 3 | Pr | AXIS_VIRTUAL_LINK_PRM | | | | 軸パラメータ | 無効 | WRITE(⇒モーション): Pr |
| 4 | Md | AXIS_VIRTUAL_LINK_MONI | | | | 軸モニタデータ | 有効 | READ(モーション⇒): Md |
| 5 | Cd | AXIS_VIRTUAL_LINK_CMD | | | | 軸制御データ | 無効 | WRITE(⇒モーション): Cd |
| 6 | | | | | | | | |

[SYS_MONI構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | Addon_AbsSystem | ADDON_MONI | | | | アドオンAbsSystemモニタ | 無効 | - |
| 2 | Addon_Axis | ADDON_MONI | | | | アドオンAxisモニタ | 無効 | - |
| 3 | Addon_ExternalSignal | ADDON_MONI | | | | アドオンExternalSignalモニタ | 無効 | - |
| 4 | Addon_FileTransfer | ADDON_MONI | | | | アドオンFileTransferモニタ | 無効 | - |

≈ ≈

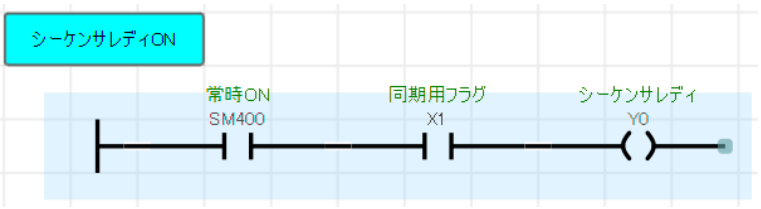
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 23 | BufferMemoryRefreshCycle | CYCLE_MONI | | | | バッファメモリリフレッシュ周期モニタ | 無効 | - |
| 24 | Environment_UserRootPath | 文字列(127) | | "" | | ユーザルートパス | 無効 | - |
| 25 | Error | ビット | | 0 | | モーション部システムエラー検出 | 有効 | - |
| 26 | ErrorHistory | ワード[符号なし]/ビット… | | 0 | | エラー履歴情報 | 無効 | - |
| 27 | ErrorHistory_Latest | ワード[符号付き] | | 0 | | 最新エラー履歴データ番号 | 無効 | - |

## 4.4　シーケンサレディ（プログラム名：ServoON_Jog）

モーションユニットRD78Gは，シーケンサCPUとのデータ授受に，入力32点，出力32点を使用します。プログラム言語にかかわらず，シーケンサレディ[Y0]をONする必要があります。
シーケンサCPUにてシーケンサレディをONして，モーションユニットを起動します。

### (1) プログラム例

シーケンサCPUの電源投入後，モーションユニットの同期用フラグ[X1]がONになったら，シーケンサレディ信号[Y0]をONします。



## 4.5　サーボON（プログラム名：ServoON_Jog）

サーボシステムに接続されている実ドライブ軸のサーボONを行ないます。
サーボONには，全軸サーボON命令と各軸のサーボON命令の2種類があります。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(管理系) | MC_Power | 指定した軸を運転可能状態に切り換えます。 |
| | MCv_AllPower | 全ての軸を運転可能状態に切り換えます。 |

### (2) ローカルラベル

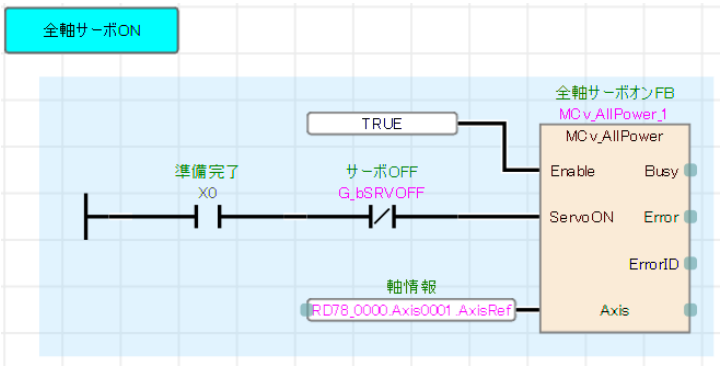
| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | MCv_AllPower_1 | MCv_AllPower | ... | 全軸サーボオンFB |
| 2 | MCv_Jog_1 | MCv_Jog | ... | JOG運転FB Axis0001 |
| 3 | MCv_Jog_2 | MCv_Jog | ... | JOG運転FB Axis0002 |
| 4 | leJogAcceleration | 倍精度実数 | ... | JOG加速度 |
| 5 | leJogDeceleration | 倍精度実数 | ... | JOG減速度 |
| 6 | leJogJerk | 倍精度実数 | ... | JOGジャーク |
| 7 | | | ... | |

JOG運転のプログラムで使用するラベルです。

プログラムエディタにMCv_AllPowerをドラッグ＆ドロップして，ローカルラベルにMCv_AllPower_1を追加します。（4.1.3項参照）

### (3) プログラム例

シーケンサレディ[Y0]をONすると，準備完了[X0]がONします。それを全軸サーボON信号として使用します。サーボOFFする場合は，G_bSRVOFFをONします。
MCv_AllPowerの場合，任意の軸のAxisRef型構造体をAxis入力接続ポイントに接続します。




**[ポイント]**
Axisに入力するAxisRef型構造体には，ユニットの入出力番号が格納されている必要があります。
AxisRef型構造体を手動で設定する場合，AxisRef.StartIOの値は，ユニットの入出力番号の下1桁を削除した数値になります。
例えば，入出力番号が“0010”の場合は，AxisRef.StartIOに“1”を入力してください。

## 4.6 JOG運転（プログラム名：ServoON_Jog）

JOG運転は，JOG正転指令／JOG逆転指令を入力している間，サーボシステムから軸に指令を出力し，指定方向へ軸が動作します。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MCv_Jog | 目標速度に従いJOG運転を実行します。 |

### (2) 加減速処理機能

モーション制御の加減速を，装置に適した加減速カーブに調整する機能です。

(a) 概要

加減速方式は，以下から選択します。

| 加減速方式 | 内　容 |
|---|---|
| 加減速度指定方式<br>(初期値) | FBで指定した加速度，減速度，ジャークを用いて，加速・減速します。 |
| 加減速時間一定方式 | 速度に関係なく，FBで指定した加減速時間を用いて，加速・減速します。 |

(b) 設定方法

MCv_Jogを含む，動作系MCFBのOptions入力によって設定します。

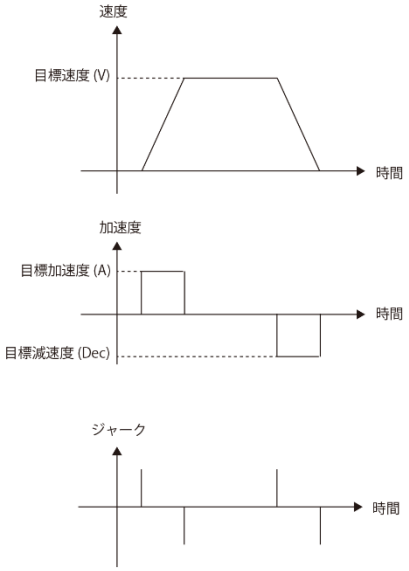
| ビット | 機能説明 |
|---|---|
| 0～2 | 加減速方式設定<br>0：mcAccDec 加減速度指定方式<br>1：mcFixedTime 加減速時間一定方式 |
| 3～15 | 空き |
| 16～31 | 各FBにより機能が異なります。 |

(c) 加減速度指定方式

FBの「加減速方式設定」(Options bit0～2)入力で「0：mcAccDec」を選択して，加速度・減速度・ジャークを設定します。

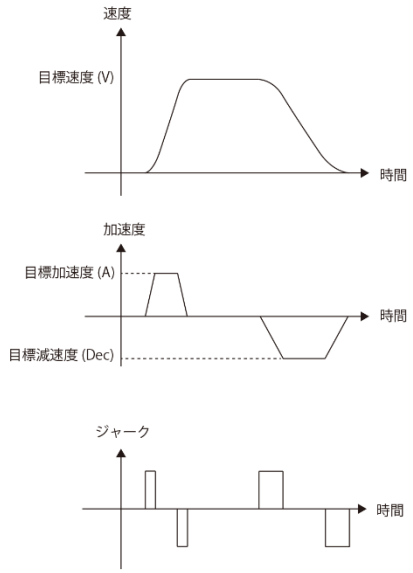
1) 台形加減速

ジャークに「0.0」を指定した場合を台形加減速と呼びます。
速度は台形の波形となります。



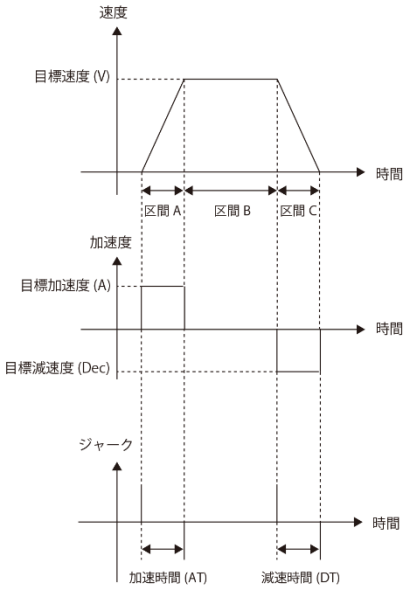

2) ジャーク加減速

ジャークに「0.0」以外を指定した場合をジャーク加減速と呼びます。
速度はS字の波形となります。



(d) 加減速時間一定方式

FBの「加減速方式設定」(Options bit0～2)で「1：mcFixedTime」を選択して，Accelerationに加減速時間を設定します。DecelerationとJerkは使用しません。




(e) FBの入力変数

(b)～(d)までの内容を，FBの入力値ごとに整理した内容を以下に示します。

| 入力変数 | 名　称 | 詳　細 |
|---|---|---|
| Velcotiy | 目標速度 | FB での目標速度を指定します。 |
| Acceleration | 加速度／加減速時間 | FBでの加速度／加減速時間を指定します。<br>・加減速方式が「0：mcAccDec」：[U/s$^2$]単位<br>・加減速方式が「1：mcFixedTime」：[s]単位 |
| Deceleration | 減速度 | FBでの減速度を指定します。<br>・加減速方式が「0：mcAccDec」：[U/s$^2$]単位<br>・加減速方式が「1：mcFixedTime」：使用しません |
| Jerk | ジャーク | FBでのジャークを指定します。<br>・加減速方式が「0：mcAccDec」：[U/s$^3$]単位<br>・加減速方式が「1：mcFixedTime」：使用しません |
| Options | オプション | bit0～2 で加減速方式を指定します。<br>0：mcAccDec 加減速度指定方式<br>1：mcFixedTime 加減速時間一定方式 |

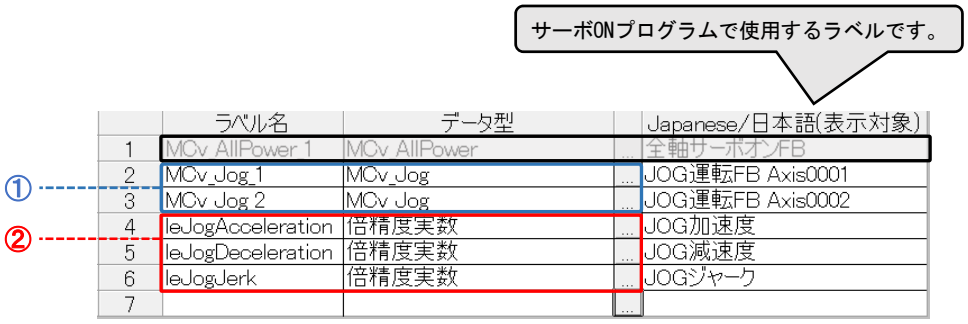
**[ポイント]**

**“0：加減速度指定方式”を選択した場合のAcceleration，Decelerationの値は，実行する加減速時間から，以下のように計算します。**

**[目標速度V[U/s]，加速時間[s]，減速時間[s]の場合]，**

```
Velocity := (目標速度V);
Acceleration := (目標速度V/加速時間);
Deceleration := (目標速度V/減速時間);
Options := (mcAccDec);
```

(3) ローカルラベル




| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | MCv_AllPower_1 | MCv_AllPower | ... | 全軸サーボオンFB |
| 2 | MCv_Jog_1 | MCv_Jog | ... | JOG運転FB Axis0001 |
| 3 | MCv_Jog_2 | MCv_Jog | ... | JOG運転FB Axis0002 |
| 4 | leJogAcceleration | 倍精度実数 | ... | JOG加速度 |
| 5 | leJogDeceleration | 倍精度実数 | ... | JOG減速度 |
| 6 | leJogJerk | 倍精度実数 | ... | JOGジャーク |
| 7 | | | ... | |

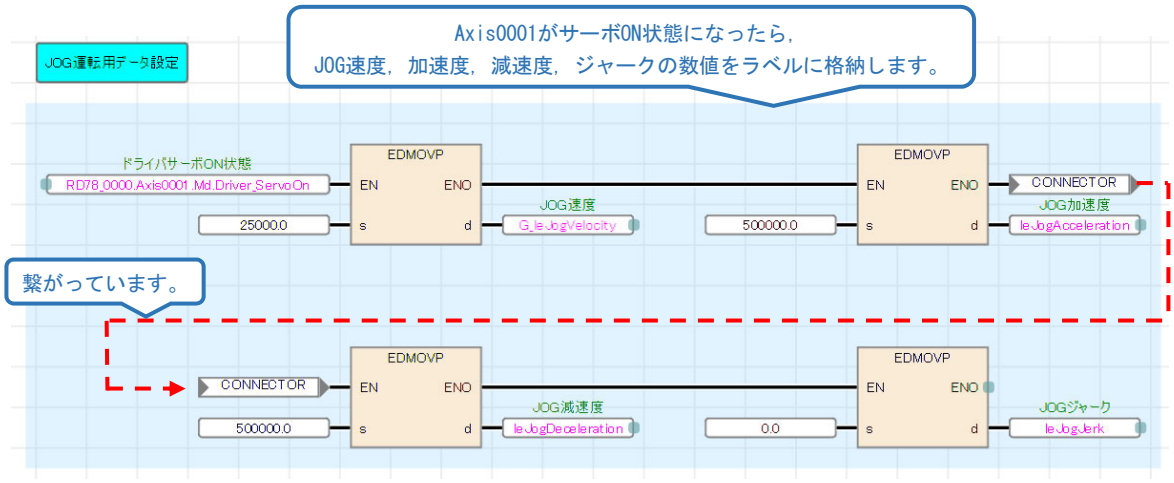
①プログラムエディタにFB(MCv_Jog)をドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

(4) プログラム例

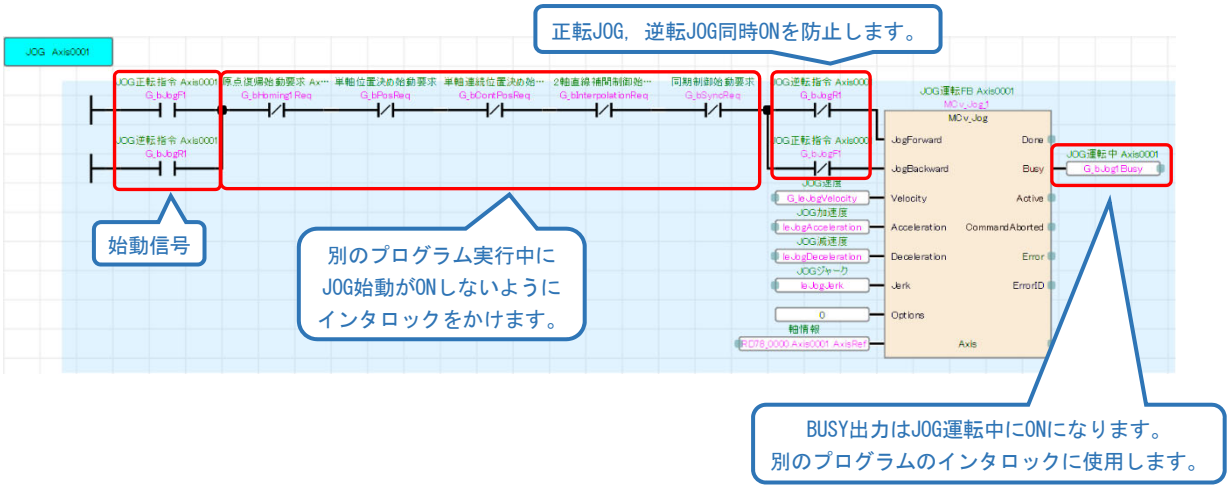
JOG速度や加速度，減速度，ジャークはEDMOVP命令で数値を格納します。
正転JOG指令，逆転JOG指令に割り付けたラベルを，FB(MCv_Jog)の指令入力(JogForward, JogBackward)接続ポイントに接続します。
別のプログラム実行中にJOG運転が実行されないようにインタロックをかけます。

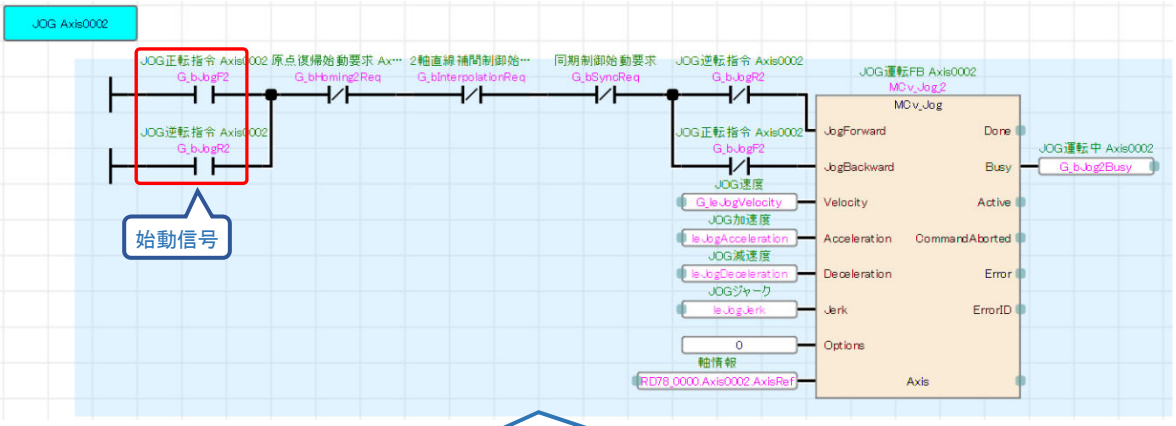
[始動信号]

| | Axis0001 | Axis0002 |
|---|---|---|
| JOG 正転指令 | G_bJogF1 | G_bJogF2 |
| JOG 逆転指令 | G_bJogR1 | G_bJogR2 |




**[ポイント]**
EDMOVP命令は，プログラムエディタに直接“EDMOVP”とタイプするか，部品選択ウィンドウの[部品一覧]タブから，“応用命令”⇒“浮動小数点命令”下にある“EDMOVP”をドラッグ＆ドロップして入力します。


正転JOG，逆転JOG同時ONを防止します。
JOG Axis0001
始動信号
別のプログラム実行中に
JOG始動がONしないように
インタロックをかけます。
JOG運転FB Axis0001
MCv_Jog
JogForward
JogBackward
Velocity
Acceleration
Deceleration
Jerk
Options
Axis
Done
Busy
Active
CommandAborted
Error
ErrorID
JOG運転中 Axis0001
BUSY出力はJOG運転中にONになります。
別のプログラムのインタロックに使用します。


JOG Axis0002
始動信号
JOG運転FB Axis0002
MCv_Jog
JogForward
JogBackward
Velocity
Acceleration
Deceleration
Jerk
Options
Axis
Done
Busy
Active
CommandAborted
Error
ErrorID
JOG運転中 Axis0002
構成は軸1のプログラムと共通です。

## 4.7 原点復帰（プログラム名：Homing）

原点復帰方式は，サーボアンプのパラメータで設定します。
本節では，データセット式原点復帰を例に説明します。
近点ドグ式原点復帰については付録を参照してください。

### (1) 概要

データセット式原点復帰は，原点復帰を実行した場所を原点とする方式です。

［データセット式原点復帰(Method37)のタイムチャート］

### (2) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MC_Home | 指定した軸の原点復帰を行ないます。 |

### (3) ローカルラベル

| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | MC_Home_1 | MC_Home | ... | 原点復帰FB1 Axis0001 |
| 2 | MC_Home_2 | MC_Home | ... | 原点復帰FB2 Axis0002 |
| 3 | MC_Home_3 | MC_Home | ... | 原点復帰FB3 VirtualAxis0001 |
| 4 | bHoming1Done | ビット | ... | 原点復帰FB1 Done出力 |
| 5 | bHoming1Error | ビット | ... | 原点復帰FB1 Error出力 |
| 6 | bHoming2Done | ビット | ... | 原点復帰FB2 Done出力 |
| 7 | bHoming2Error | ビット | ... | 原点復帰FB2 Error出力 |
| 8 | bHoming3Done | ビット | ... | 原点復帰FB3 Done出力 |
| 9 | bHoming3Error | ビット | ... | 原点復帰FB3 Error出力 |
| 10 | | | | |

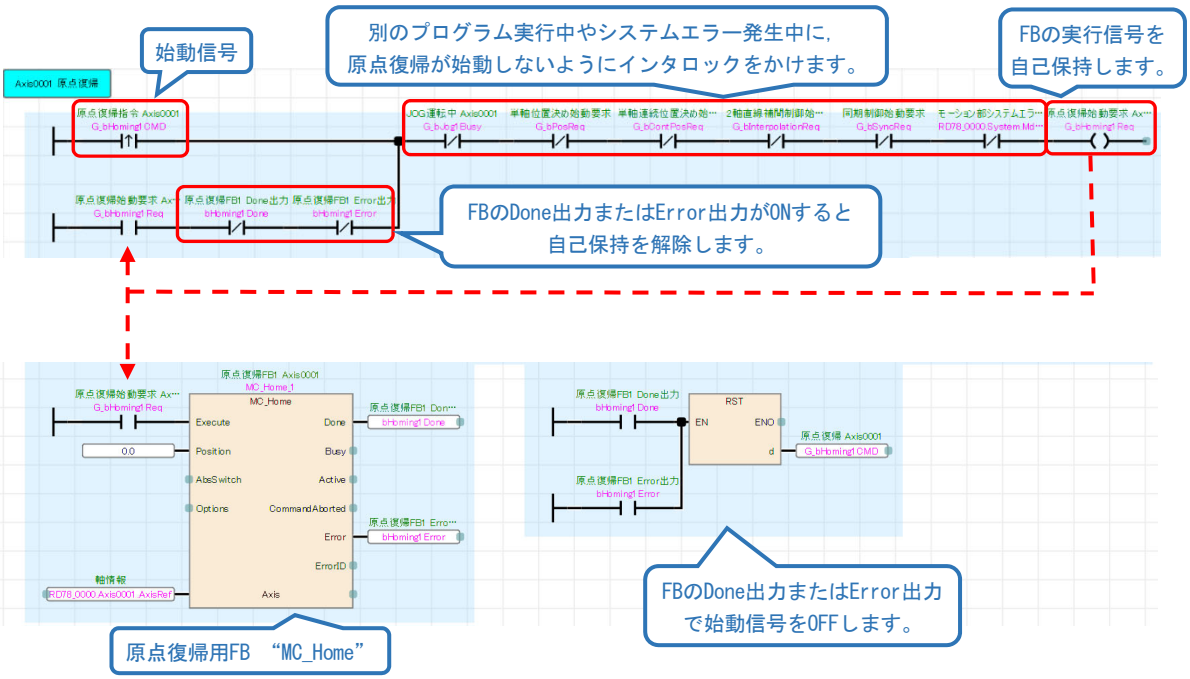
①プログラムエディタにFB(MC_Home)をドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

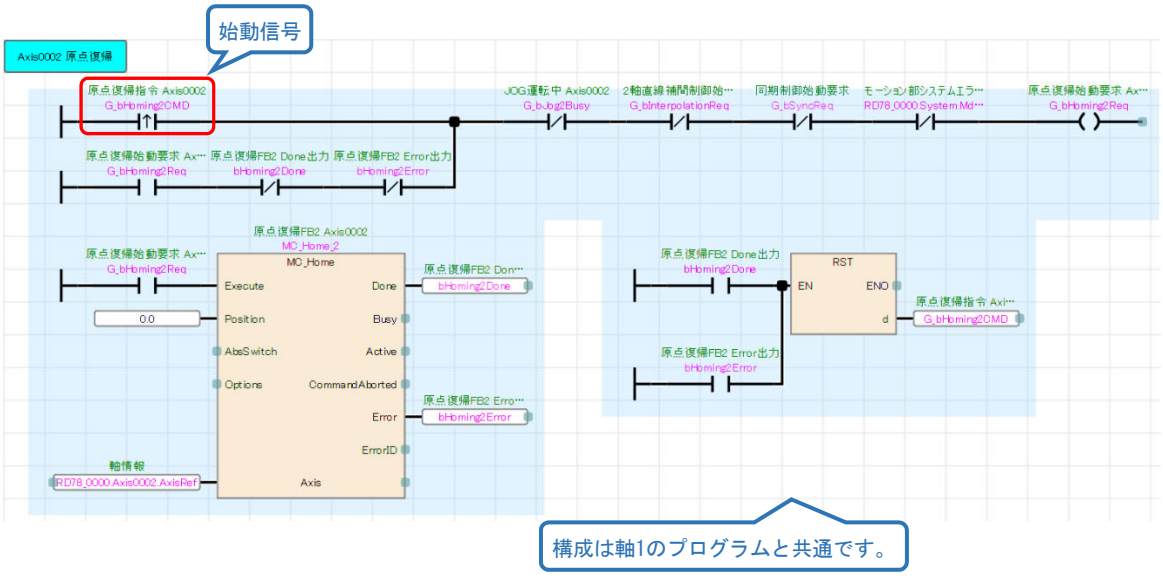
### (4) プログラム例

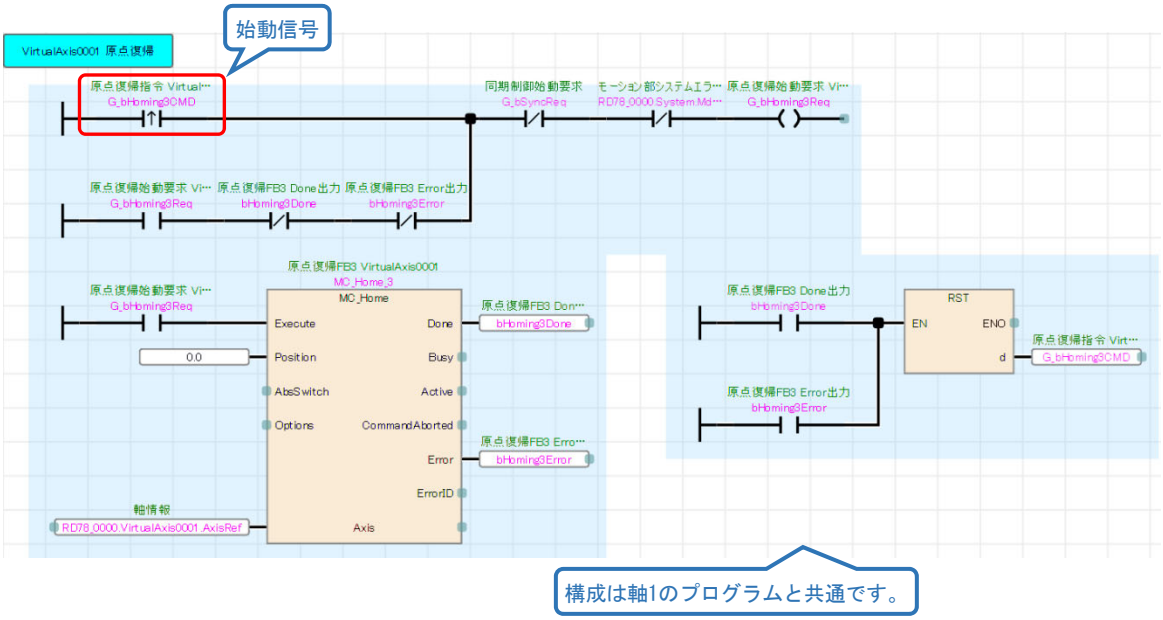
軸1(Axis0001)と軸2(Axis0002)，および仮想ドライブ軸1(VirtualAxis0001)の原点復帰を行ないます。原点復帰信号に割り付けたラベルの立ち上がりを保持し，FB(MC_Home)の実行入力(Execute)接続ポイントに接続します。原点復帰完了信号(MC_HomeのDone出力)がON，またはエラー発生時，始動信号をOFFします。
別のプログラム実行中やシステムエラー発生中に，原点復帰が始動しないようにインタロックをかけます。

[始動信号]

| | Axis0001 | Axis0002 | VirtualAxis0001 |
|---|---|---|---|
| 原点復帰指令 | G_bHoming1CMD | G_bHoming2CMD | G_bHoming3CMD |




始動信号
VirtualAxis0001 原点復帰
原点復帰指令 Virtual…
G_bHoming3CMD
同期制御始動要求
G_bSyncReq
モーション部システムエラ…
RD78_0000.SystemMd…
原点復帰始動要求 Vi…
G_bHoming3Req
原点復帰始動要求 Vi…
G_bHoming3Req
原点復帰FB3 Done出力
bHoming3Done
原点復帰FB3 Error出力
bHoming3Error
原点復帰FB3 VirtualAxis0001
MC_Home_3
MC_Home
原点復帰始動要求 Vi…
G_bHoming3Req
Execute
Done
原点復帰FB3 Don…
bHoming3Done
0.0
Position
Busy
AbsSwitch
Active
Options
CommandAborted
Error
原点復帰FB3 Erro…
bHoming3Error
ErrorID
軸情報
RD78_0000.VirtualAxis0001_AxisRef
Axis
原点復帰FB3 Done出力
bHoming3Done
RST
EN
ENO
d
原点復帰指令 Virt…
G_bHoming3CMD
原点復帰FB3 Error出力
bHoming3Error
構成は軸1のプログラムと共通です。

## 4.8　単軸位置決め制御（プログラム名：Positioning）

アドレス情報を使用して，指定の位置へ位置決めを行ないます。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MC_MoveAbsolute | 絶対位置の目標位置を指定し，位置決めを実行します。 |
| | MC_MoveRelative | 相対位置の移動量を指定し，位置決めを実行します。 |

### (2) ローカルラベル

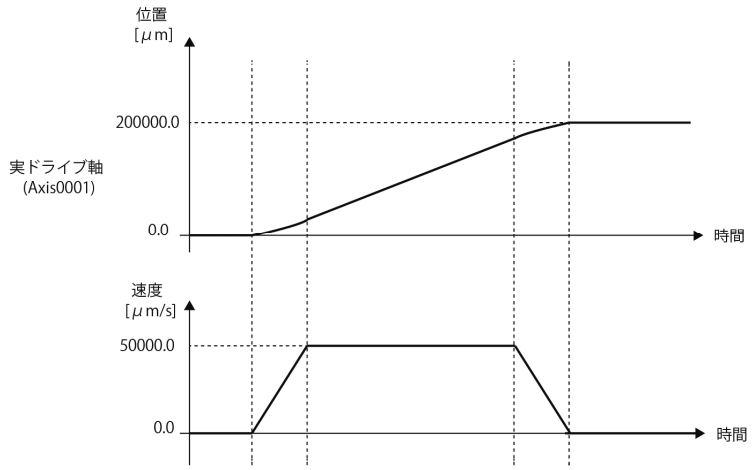
①
②

| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | MC_MoveRelative_1 | MC_MoveRelative | ... | 相対値位置決めFB |
| 2 | leDistance | 倍精度実数 | ... | 移動量 |
| 3 | leVelocity | 倍精度実数 | ... | 速度 |
| 4 | leAcceleration | 倍精度実数 | ... | 加速度 |
| 5 | leDeceleration | 倍精度実数 | ... | 減速度 |
| 6 | leJerk | 倍精度実数 | ... | ジャーク |
| 7 | bDone | ビット | ... | 相対値位置決めFB Done出力 |
| 8 | bError | ビット | ... | 相対値位置決めFB Error出力 |
| 9 | bValueSet | ビット | ... | 変数セット完了 |
| 10 | bCommandAborted | ビット | ... | 相対値位置決めFB 実行中断出力 |
| 11 | | | ... | |

①プログラムエディタにFB(MC_MoveRelative)をドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

### (3) プログラム例

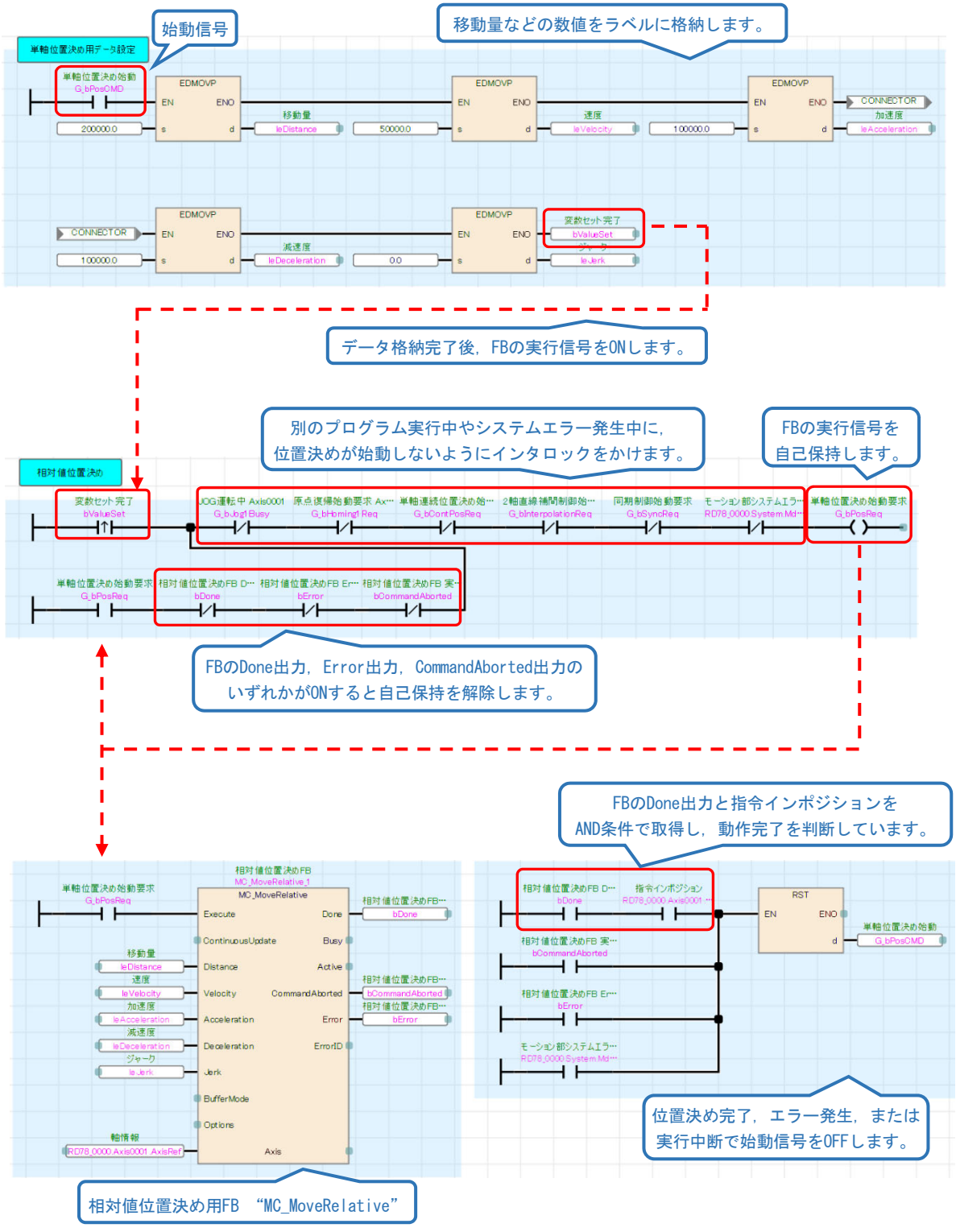
下図の動作パターンの相対位置指定の位置決めを実行するプログラム例を示します。




単軸位置決め指令がONになると，位置決め用の各データを各ラベルに格納します。全データ格納が完了したら，FB(MC_MoveRelative)の実行要求を自己保持します。位置決め完了信号(MC_MoveRelativeのDone出力と指令インポジションのAND条件)，エラー発生，または実行中断で，始動信号をOFFします。
別のプログラム実行中やシステムエラー発生中に，単軸位置決めが始動しないようにインタロックをかけます。

**[始動信号]**

| | |
|---|---|
| 単軸位置決め指令 | G_bPosCMD |

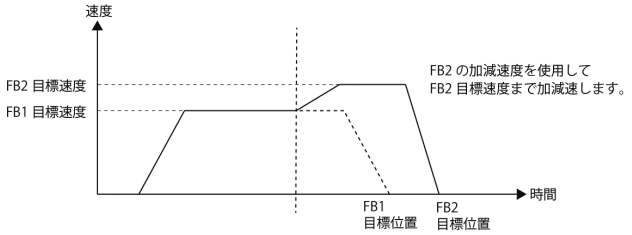
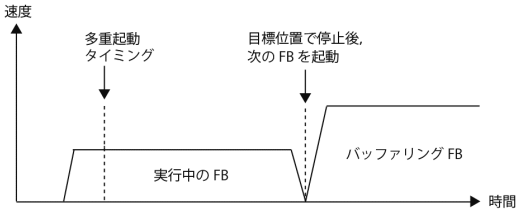
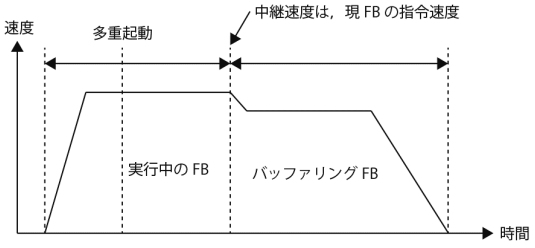
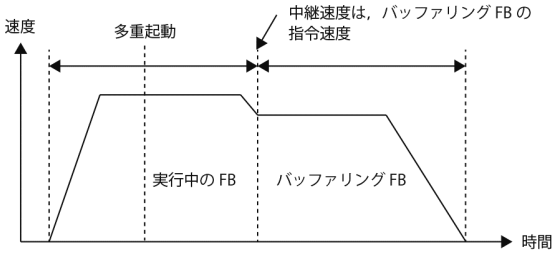

始動信号
移動量などの数値をラベルに格納します。
単軸位置決め用データ設定
単軸位置決め始動
G_bPosCMD
EDMOVP
EN
ENO
s
d
200000.0
移動量
leDistance
50000.0
速度
leVelocity
100000.0
CONNECTOR
加速度
leAcceleration
100000.0
減速度
leDeceleration
0.0
ジャーク
leJerk
変数セット完了
bValueSet
データ格納完了後，FBの実行信号をONします。
別のプログラム実行中やシステムエラー発生中に，
位置決めが始動しないようにインタロックをかけます。
FBの実行信号を
自己保持します。
相対値位置決め
変数セット完了
bValueSet
JOG運転中 Axis0001
G_b_log1Busy
原点復帰始動要求 Ax…
G_bHoming1Req
単軸連続位置決め始…
G_bContPosReq
2軸直線補間制御始…
G_bInterpolationReq
同期制御始動要求
G_bSyncReq
モーション部システムエラー…
RD78_0000.SystemMd…
単軸位置決め始動要求
G_bPosReq
単軸位置決め始動要求
G_bPosReq
相対値位置決めFB D…
bDone
相対値位置決めFB Er…
bError
相対値位置決めFB 実…
bCommandAborted
FBのDone出力，Error出力，CommandAborted出力の
いずれかがONすると自己保持を解除します。
FBのDone出力と指令インポジションを
AND条件で取得し，動作完了を判断しています。
相対値位置決めFB
MC_MoveRelative_1
MC_MoveRelative
Execute
Done
ContinuousUpdate
Busy
Distance
Active
Velocity
CommandAborted
Acceleration
Error
Deceleration
ErrorID
Jerk
BufferMode
Options
Axis
軸情報
RD78_0000.Axis0001.AxisRef
相対値位置決めFB…
bDone
bCommandAborted
bError
指令インポジション
RD78_0000.Axis0001…
RST
単軸位置決め始動
G_bPosCMD
モーション部システムエラ…
RD78_0000.SystemMd…
位置決め完了，エラー発生，または
実行中断で始動信号をOFFします。
相対値位置決め用FB “MC_MoveRelative”

## 4.9 単軸連続位置決め（プログラム名：ContinuousPositioning）

### (1) 概要

モーション制御FBを実行中の軸に，別インスタンスの動作系FBを実行することで，複数のモーション制御FBを停止することなく連続的に実行できます。
モーション制御FBのBufferMode入力で，バッファモードを指定します。
1つの軸でバッファリングできる動作系FBは最大2つです。

### (2) バッファモードの動作パターン

| バッファモード | 動　作 |
|---|---|
| 0：mcAborting | 実行中のFBを中断(キャンセル)して次のFBを即時実行します。<br>  |
| 1：mcBuffered | 実行中のFBが減速停止後，次のFBを実行します。<br>  |
| 2：mcBlendingLow | 実行中のFBとバッファリングFBの目標速度のうち，低い方の速度を切換え速度とします。 |
| 3：mcBlendingPrevious | 実行中のFBが目標位置へ到達後，次のFBを実行します。<br>目標速度は次のFB指令速度です。<br>  |
| 4：mcBlendingNext | 実行中のFBが目標位置へ到達後，次のFBを実行します。<br>目標位置到着時，次FBの指令速度になります。<br>  |
| 5：mcBlendingHigh | 実行中のFBとバッファリングFBの目標速度のうち，高い方の速度を切換え速度とします。 |

### (3) ローカルラベル

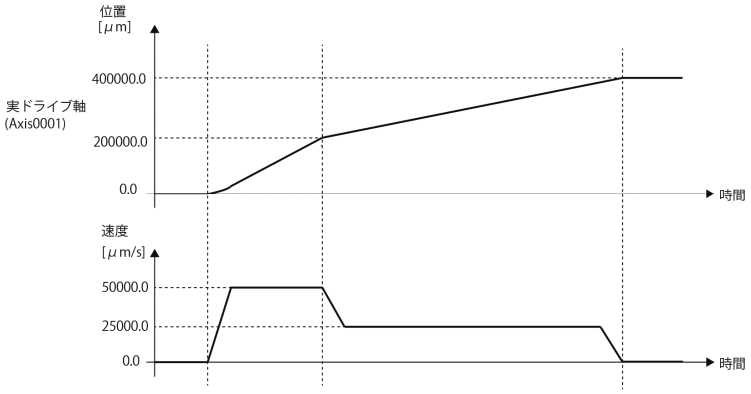
| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | leDistance1 | 倍精度実数 | ... | 移動量1 |
| 2 | leVelocity1 | 倍精度実数 | ... | 速度1 |
| 3 | leDistance2 | 倍精度実数 | ... | 移動量2 |
| 4 | leVelocity2 | 倍精度実数 | ... | 速度2 |
| 5 | leAcceleration1 | 倍精度実数 | ... | 加速度1 |
| 6 | leDeceleration1 | 倍精度実数 | ... | 減速度1 |
| 7 | leAcceleration2 | 倍精度実数 | ... | 加速度2 |
| 8 | leDeceleration2 | 倍精度実数 | ... | 減速度2 |
| 9 | leJerk | 倍精度実数 | ... | ジャーク |
| 10 | MC_MoveRelative_1 | MC_MoveRelative | ... | 相対値位置決めFB1 |
| 11 | MC_MoveRelative_2 | MC_MoveRelative | ... | 相対値位置決めFB2 |
| 12 | bError | ビット | ... | 相対値位置決めFB Error出力 |
| 13 | TON_1 | TON | ... | オンディレイタイマFB |
| 14 | bDwell_out | ビット | ... | タイマ出力 |
| 15 | bDwell_in | ビット | ... | タイマ入力 |
| 16 | bValueSet | ビット | ... | 変数セット完了 |
| 17 | bCommandAboted | ビット | ... | 相対値位置決めFB 実行中断出力 |
| 18 | | | ... | |

②
①

①プログラムエディタにFB(MC_MoveRelative, TON)をドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

### (4) プログラム例

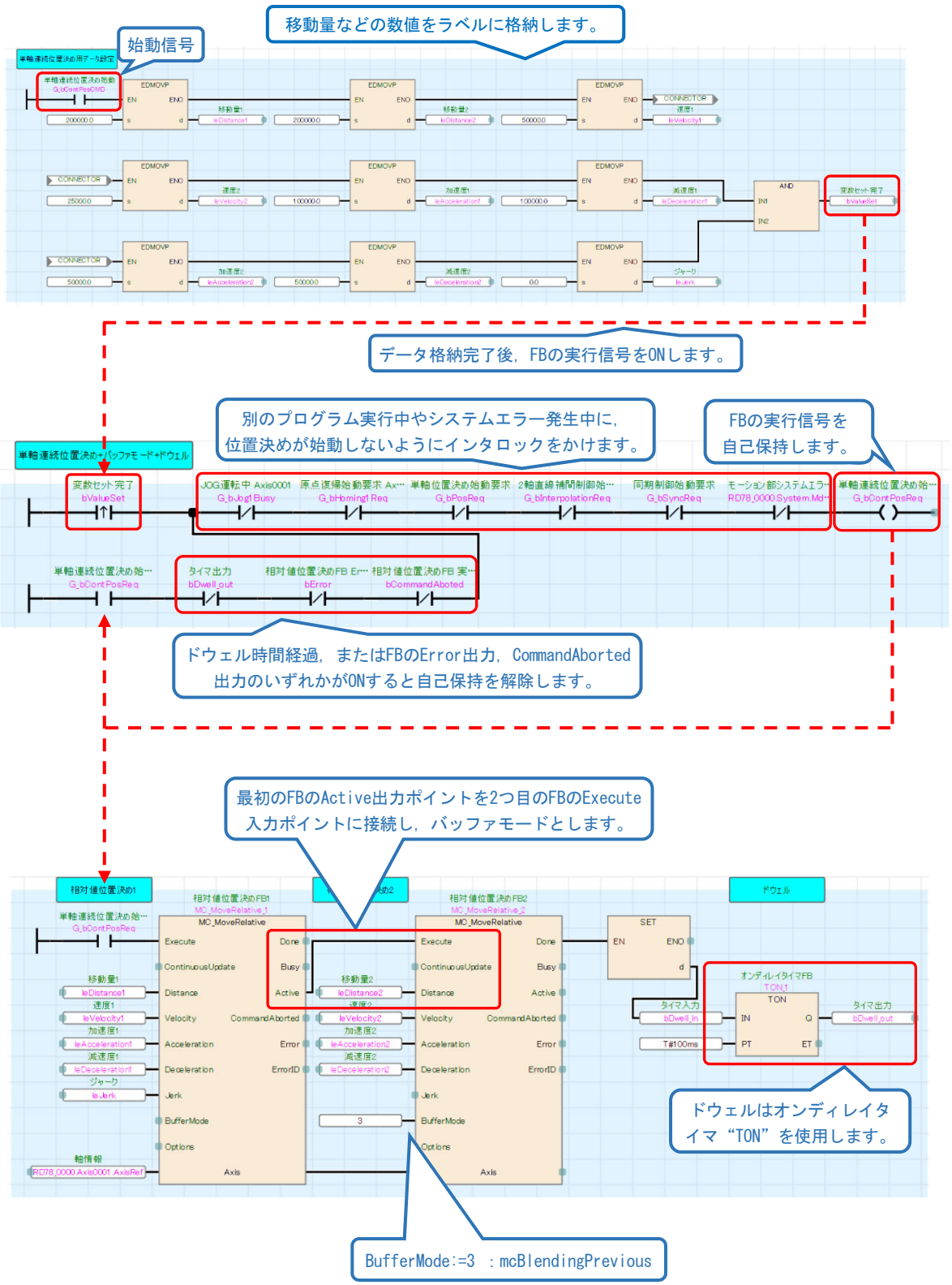
下図のような動作パターンの相対位置指定の位置決めを実行するプログラム例を示します。

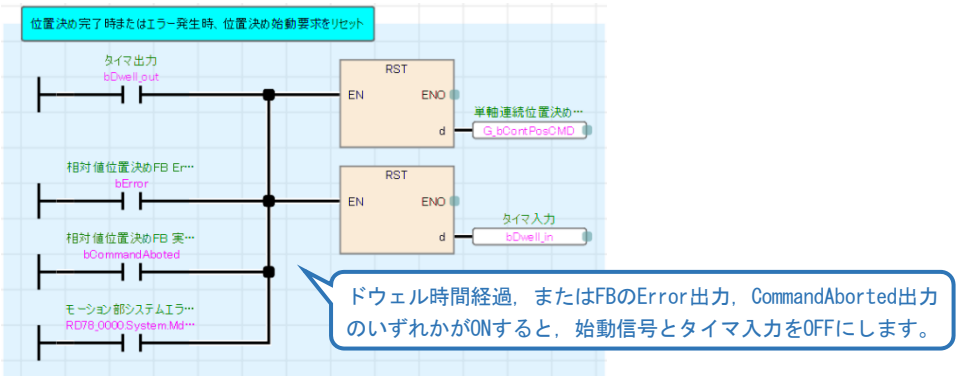


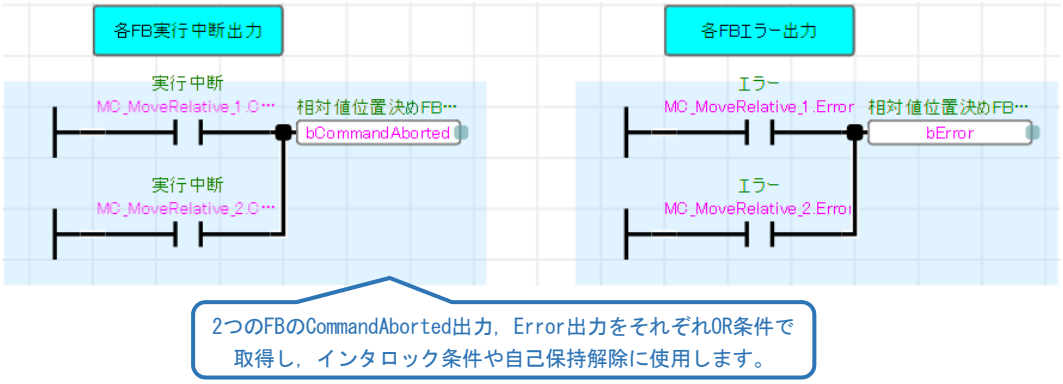

単軸連続位置決め指令がONになると，位置決め用の各データを各ラベルに格納します。
全データの格納が完了したら，FB(MC_MoveRelative_1)の実行要求を自己保持します。
連続位置決め始動信号に割り付けたデバイスの立ち上がりを保持し，1つ目の
FB(MC_MoveRelative)の実行指令(Execute)接続ポイントに接続します。
1つ目のMC_MoveRelativeのActive出力ポイントを2つ目のMC_MoveRelativeのExecute入力ポイントに接続することで，最初のFB実行中に2つ目のFBを実行し，バッファリングします。
1つ目のFBの位置決めが完了したら，バッファされている2つ目のFBが実行されます。ドウェルにはオンディレイタイマ(100[ms])を使用します。ドウェル時間経過後，またはエラー発生，または実行中断時，始動信号とオンディレイタイマの入力をリセットします。
別のプログラム実行中やシステムエラー発生中に，単軸連続位置決めが始動しないようにインタロックをかけます。

**[始動信号]**

| | |
|---|---|
| 単軸連続位置決め指令 | G_bContPosCMD |


移動量などの数値をラベルに格納します。
始動信号
データ格納完了後，FBの実行信号をONします。
別のプログラム実行中やシステムエラー発生中に，
位置決めが始動しないようにインタロックをかけます。
FBの実行信号を
自己保持します。
ドウェル時間経過，またはFBのError出力，CommandAborted
出力のいずれかがONすると自己保持を解除します。
最初のFBのActive出力ポイントを2つ目のFBのExecute
入力ポイントに接続し，バッファモードとします。
ドウェルはオンディレイタ
イマ“TON”を使用します。
BufferMode:=3 ：mcBlendingPrevious



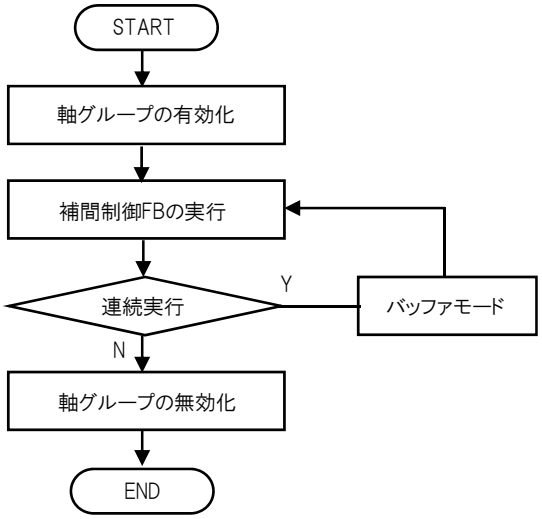




**[ポイント]**
FBの出力信号は，プログラム内に直接“(FB名).(出力信号名)”を直接入力すると，ラベルを接続しなくても使用することができます。

## 4.10 補間制御（プログラム名：LinearInterpolation）

### 4.10.1 補間制御の手順

2軸以上で補間制御を実行する手順を下図に示します。




### 4.10.2 軸グループ有効化／無効化

軸グループの設定は，3.12節を参照してください。
補間制御を実行するには，軸グループ状態を「4：GroupStandby」に設定する必要があります。

(1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(管理系) | MC_GroupEnable | 指定した軸グループの状態を「0：GroupDisabled」から「4：GroupStandby」に遷移します。 |
| | MC_GroupDisable | 指定した軸グループの状態を「0：GroupDisabled」に遷移します。 |

### 4.10.3 補間制御

直線補間用と円弧補間用を準備しています。軸グループ有効化後に，以下のFBを実行します。

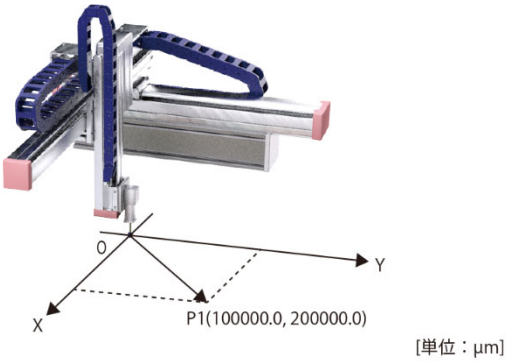
(1) 使用するFB

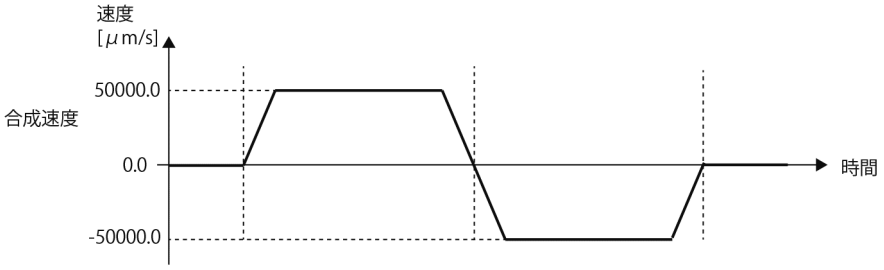
| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MCv_MoveLinearInterpolateAbsolute | 絶対値直線補間制御 |
| | MCv_MoveLinearInterpolateRelative | 相対値直線補間制御 |
| | MCv_MoveCircularInterpolateAbsolute | 絶対値円弧補間制御 |
| | MCv_MoveCircularInterpolateRelative | 相対値円弧補間制御 |

### 4.10.4　直線補間のプログラム例

(1) 動作パターン

原点(0.0, 0.0)[um]と点P1(100000.0, 200000.0)[um]の間を直線補間で往復します。




(2) 軸番号と移動量の設定方法

(a) MCv_MoveLinearInterpolateRelativeのLinearAxes入力

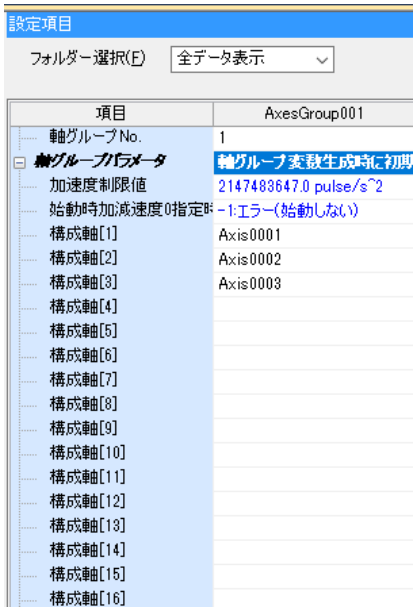
INT(符号付きワード)型の要素数16の配列を指定します。
サンプルプログラムでは，wAxes[0..15]というラベルを使用しています。
AxesGroup(3.12節参照)に登録した構成軸[1]～[16]のうち，直線補間に使用する軸の構成軸番号をwAxes[0]から順に格納します。

(b) Distance入力

LREAL(倍精度実数)型の要素数16の配列を指定します。
サンプルプログラムでは，lePosition[0..15]というラベルを使用しています。
AxesGroupに登録した構成軸[1]～[16]の移動量をlePosition[0]～lePosition[15]に格納します。

**[ポイント]**
使用する軸数にかかわらず，LinearAxes入力に設定するINT型の配列の要素数，およびDistance入力に設定するLREAL型の配列の要素数は，必ず16にする必要があります。

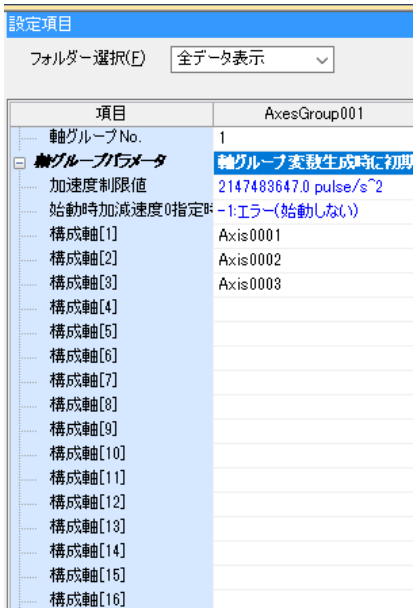
[設定例1]
・軸グループ …… 構成軸[1]：Axis0001，構成軸[2]：Axis0002，構成軸[3]：Axis0003
・直線補間 …… Axis0001，Axis0002の2軸を使用




```
wAxes[0] := 1;(←構成軸[1])
wAxes[1] := 2;(←構成軸[2])

lePosition[0] := (構成軸1(=Axis0001)の移動量);
lePosition[1] := (構成軸2(=Axis0002)の移動量);
```

[設定例2]
・軸グループ …… 構成軸[1]：Axis0001，構成軸[2]：Axis0002，構成軸[3]：Axis0003
・直線補間 …… Axis0002，Axis0003の2軸を使用




```
wAxes[0] := 2;(←構成軸[2])
wAxes[1] := 3;(←構成軸[3])

lePosition[0] := 0.0;
lePosition[1] := (構成軸2(=Axis0002)の移動量);
lePosition[2] := (構成軸3(=Axis0003)の移動量);
```

(3) ローカルラベル

| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | wAxes | ワード[符号付き](0..15) | ... | 補間軸 |
| 2 | lePosition1 | 倍精度実数(0..15) | ... | 位置データ1 |
| 3 | lePosition2 | 倍精度実数(0..15) | ... | 位置データ2 |
| 4 | leVelocity | 倍精度実数 | ... | 速度 |
| 5 | leAcceleration | 倍精度実数 | ... | 加速度 |
| 6 | leDeceleration | 倍精度実数 | ... | 減速度 |
| 7 | leJerk | 倍精度実数 | ... | ジャーク |
| 8 | MC_GroupEnable_1 | MC_GroupEnable | ... | 軸グループ有効FB |
| 9 | MCv_MoveLinearInterpolateRelative_1 | MCv_MoveLinearInterpolateRelative | ... | 相対値直線補間制御FB1 |
| 10 | MCv_MoveLinearInterpolateRelative_2 | MCv_MoveLinearInterpolateRelative | ... | 相対値直線補間制御FB2 |
| 11 | MC_GroupDisable_1 | MC_GroupDisable | ... | 軸グループ無効FB |
| 12 | bDone2 | ビット | ... | 相対値直線補間制御FB2 Done出力 |
| 13 | bGrpEnError | ビット | ... | 軸グループ有効FB Error出力 |
| 14 | bError | ビット | ... | 相対値直線補間制御FB Error出力 |
| 15 | bGrpDsblDone | ビット | ... | 軸グループ無効FB Done出力 |
| 16 | TON_1 | TON | ... | オンディレイタイマFB |
| 17 | bDwell_in | ビット | ... | タイマ入力 |
| 18 | bDwell_out | ビット | ... | タイマ出力 |
| 19 | bValueSet | ビット | ... | 変数セット完了 |
| 20 | bCommandAborted | ビット | ... | 相対値直線補間制御FB 実行中断出力 |
| 21 | bDone_Set | ビット | ... | 実行完了 |
| 22 | bCommandAborted_Set | ビット | ... | FB実行中断 |
| 23 | | | ... | |

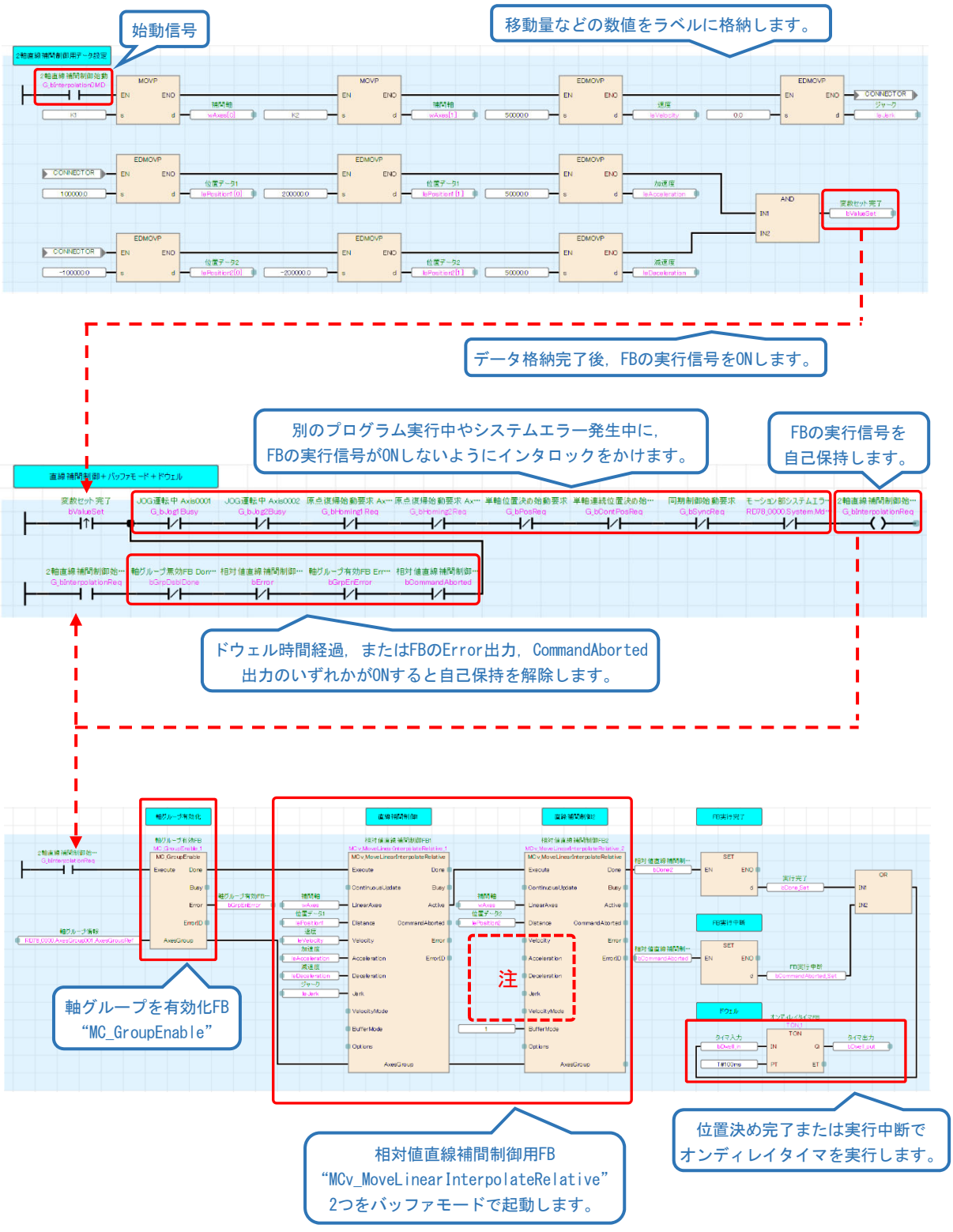
①プログラムエディタに各FBをドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

(4) プログラム例

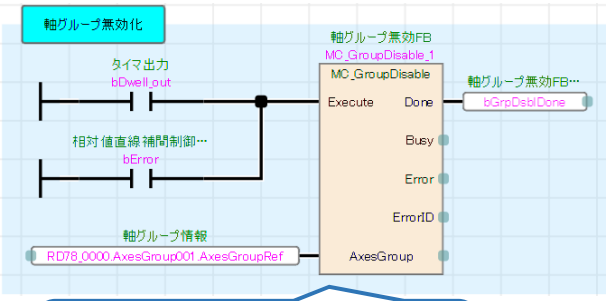
2軸直線補間制御始動がONになると，位置決め用の各データを各ラベルに格納します。
全データの格納が完了したら，FB(MC_GroupEnable_1)の実行要求を自己保持します。
MC_GroupEnableで軸グループを有効化した後，相対値直線補間制御FB
(MCv_MoveLinearInterpolateRelative)2つをバッファモードで起動します。
位置決めが完了し，ドウェル時間が経過するか，エラー発生または実行中断で，軸グループを無効化します。軸グループ無効化が完了したら，始動信号とオンディレイタイマの入力をリセットします。
別のプログラム実行中やシステムエラー発生中に，2軸直線補間制御が始動しないようにインタロックをかけます。

[始動信号]

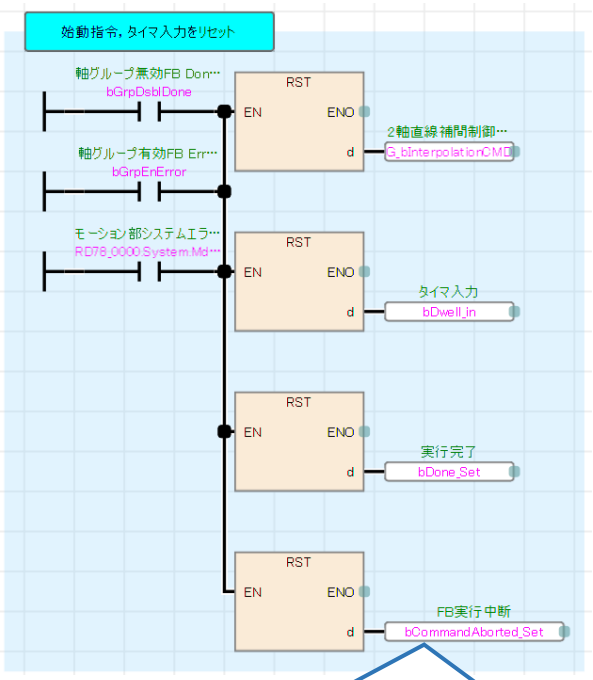
| | |
|---|---|
| 2 軸直線補間制御始動 | G_bInterpolationCMD |



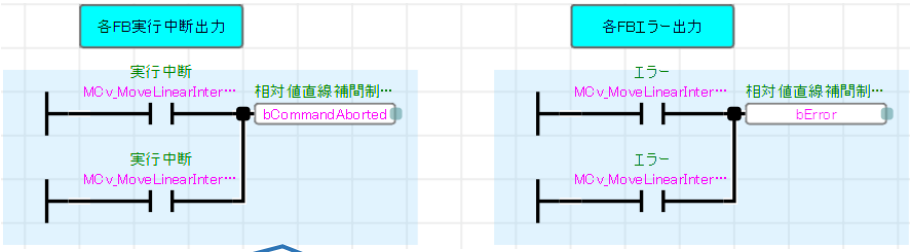

（注）バッファリング FB で指定速度，加速度指定，減速度指定を省略した場合，「バッファリング FB の直前の FB」の指定速度などが引き継がれます。




軸グループ無効化FB “MC_GroupDisable” ドウェル時間経過後，または直線補間制御エラー発生時にグループを無効化します。




以下の条件で，始動信号とタイマ入力などをOFFします。
- グループ無効化完了
- グループ有効化エラー発生時
- システムエラー発生時

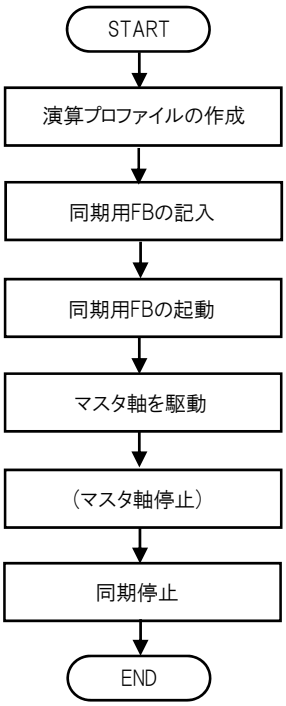



2つの直線補間用FBのCommandoAborted出力，Error出力をそれぞれOR条件で取得し，インタロック条件や自己保持解除に使用します。

## 4.11　同期制御（プログラム名：Synchronous）

### 4.11.1　同期制御の手順

同期制御を実行する手順を下図に示します。

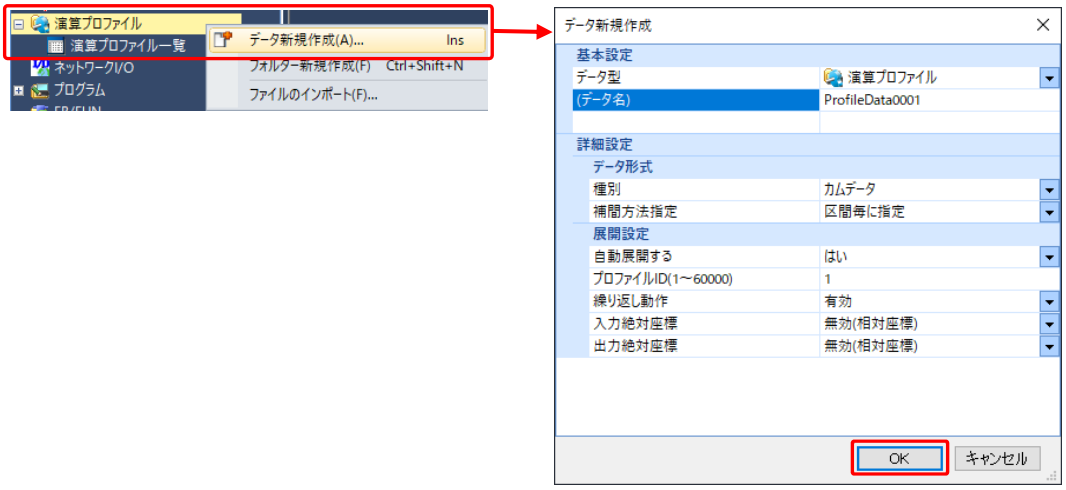


### 4.11.2　演算プロファイル

制御で使用する波形データを総称して，演算プロファイルと呼びます。
本項では，カムデータの作成方法について説明します。

#### (1) 演算プロファイルデータの新規作成

モーション制御設定機能で，ナビゲーションウィンドウの“演算プロファイル”を右クリックし，“データ新規作成”を選択して，“データ新規作成”画面を表示します。



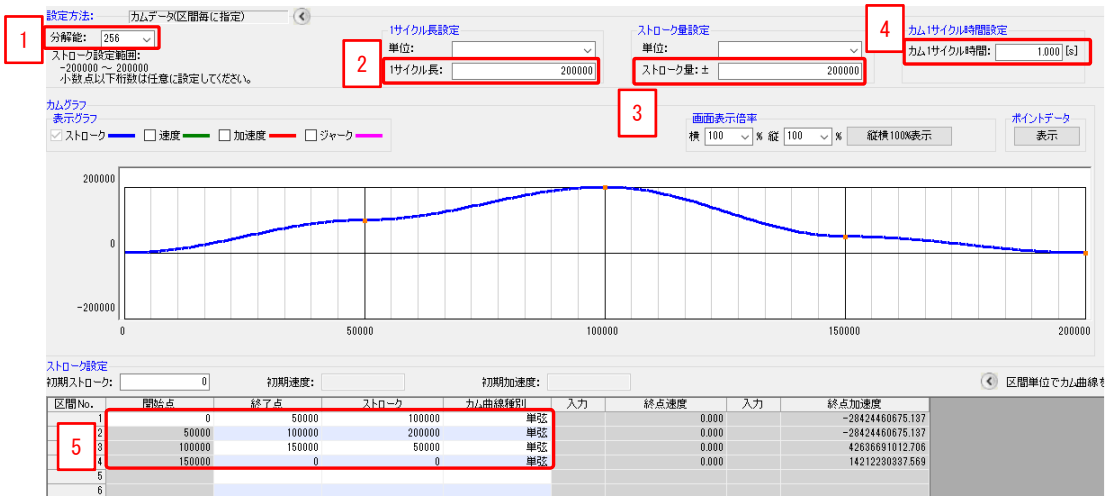

入力が必要な項目を以下に示します。

| 設定項目 | 説　明 |
|---|---|
| 自動展開する | はい：演算プロファイルを電源ON時に自動的に展開を行ないます。<br>いいえ：使用時に演算プロファイル展開 FB を実行する必要があります。 |
| 繰り返し動作 | 無効：演算プロファイルの終点まで実行すると，制御を終了します。<br>有効：設定した場合連続的に演算プロファイルの実行を繰り返します。 |
| 出力絶対座標 | 無効(相対座標)：<br>演算プロファイル(カム)実行開始時，現在の値を基準として出力値を計算します。送りカム動作を実施したい場合に選択します。<br>有効(絶対座標)：<br>演算プロファイル(カム)実行開始時の出力値が，演算プロファイルの1サイクル開始時に常に始点となるよう計算します。<br>繰り返し動作で演算プロファイルの始点と終点で異なる場合は，次の 1 サイクル開始時点で最初の出力値に戻るよう指令を 1 演算周期で出力します。 |

本書では，初期値のまま変更しません。[OK]ボタンをクリックします。

### (2) カムデータの作成

演算プロファイルの波形を設定します。




入力が必要な項目を以下に示します。

| No. | 項　目 | 内　容 |
|---|---|---|
| 1 | 分解能 | カムデータの分解能を選択します。 |
| 2 | 1サイクル長設定 | 1サイクル長の単位と1サイクル長を設定します。<br>(マスタ軸がどれだけ移動したらカムが一周するか) |
| 3 | ストローク量設定 | ストローク量の単位とストローク量を設定します。<br>(カムが一周する間に，スレーブ軸が最大どれだけ移動するか) |
| 4 | カム1サイクル時間設定 | カムが1サイクルする際にかかる時間を設定します。<br>速度，加速度，ジャークの数値の計算に使用されます。 |
| 5 | ストローク設定 | ストローク位置を設定します。 |

本項では，以下のように設定してください。

| No. | 項　目 | 設定値 |
|---|---|---|
| 1 | 分解能 | 256 |
| 2 | 1サイクル長設定 | 200000（単位はブランク） |
| 3 | ストローク量設定 | 200000（単位はブランク） |
| 4 | カム1サイクル時間設定 | 1.000 |
| 5 | ストローク設定 | 下表参照 |

| 区間No. | 開始点 | 終了点 | ストローク | カム曲線種別 |
|---|---|---|---|---|
| 1 | 0 | 50000 | 100000 | 単弦 |
| 2 | 50000 | 100000 | 200000 | 単弦 |
| 3 | 100000 | 150000 | 50000 | 単弦 |
| 4 | 150000 | 0 | 0 | 単弦 |

[ポイント]
直線カムの動作(マスタ軸と同じ動作，またはマスタ軸に対して一定の速度比を掛けた動作)をさせる場合，直線カムの演算プロファイルを作成するか，MC_GearInを使用してください。
直線カムの演算プロファイルデータはシステムでは準備していません。

### 4.11.3 単軸同期用FB

単軸同期用FBは，Masterに同期したSlaveの位置情報(指令)を伝達することで，ギア，変速機，カムなど機械的な仕組みをソフトウェアで制御できます。

(1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MC_CamIn | カム動作を実行します。 |
| | MC_GearIn | 主軸と従軸との間の速度比を設定してギア動作に入ります。 |
| | MC_CombineAxes | 選択可能な合成方法により，2つの軸の動作を合成して，第3の軸に出力します。 |
| | MCv_ChangeCycle | MC_CamIn制御中にカム1サイクル現在値を指定した値に変更します。<br>カム1サイクル現在値を任意の値に補正する目的で使用します。 |
| | MCv_******Filter | Masterの入力に対して特定のフィルタ処理を行ない，その結果をSlaveに出力します。 |
| | MC_Stop | 同期を終了するときに使用します。 |

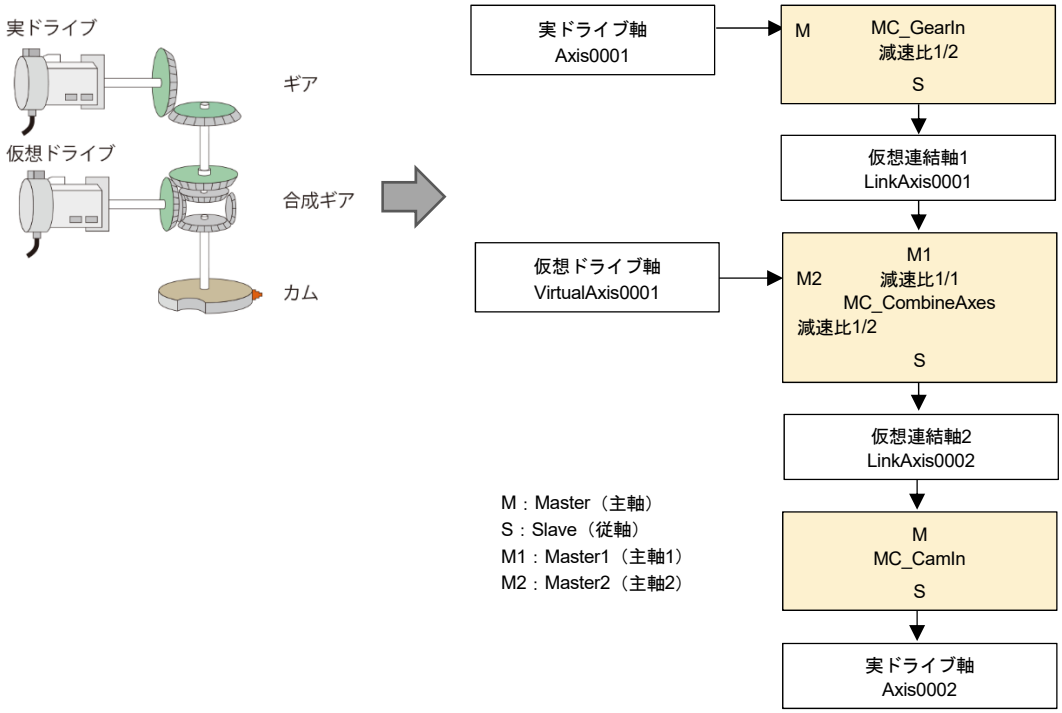
### 4.11.4 軸構成

単軸同期用FBにおいて，Master，およびSlaveに指定可能な軸を以下に示します。

| 軸種別 | 内　容 |
|---|---|
| 実ドライブ軸 | 複数の動作系 FB の制御を必要とする場合は，仮想連結軸ごとに FB を実行し，その結果(指令)を実ドライブ軸に伝達する構成にしてください。 |
| 仮想ドライブ軸 | 主に位置決め制御などにより指令を生成できます。 |
| 仮想連結軸 | 実ドライブ軸に指令を伝達するための中間軸として使用します。<br>ギアなど複数のFB(動作系)を使用する場合に，仮想連結軸の割り付けを実施してください。 |
| 実エンコーダ軸 | Masterとして使用します。<br>Slaveとした場合は，エラー「必須スレーブオブジェクト未設定」(エラーコード:1AA8H)となり始動しません。 |
| 仮想エンコーダ軸 | |

軸を単軸同期用FBに連結させることで，軸間で指令を伝達できます。また，複数のFBを実行してギアやカムなどの制御を同時に実行する場合は，軸を必要数分連結させて単軸同期用FBを実行してください。

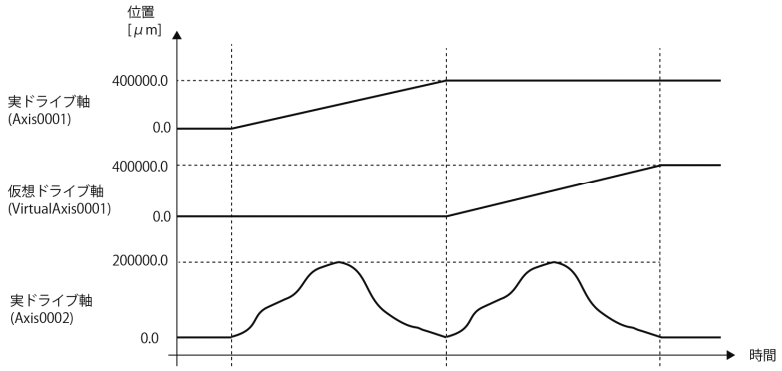
本章では，以下のカムのシステムを例に説明します。




## 4.11.5　プログラム例

### (1) 動作パターン

原点からX軸(Axis0001)が400000.0[um]移動すると同時に，Y軸(Axis0002)が4.11.2項で作成したカムパターンに従って動作します。X軸が400000.0[um]に到達後，仮想ドライブ軸が起動してY軸がカム動作します。このとき，X軸は停止しているため，Y軸のみ動作します。




### (2) 仮想ドライブ軸，仮想連結軸

実ドライブ軸(Axis0001，Axis0002)のほか，仮想ドライブ軸(VirtualAxis0001)と仮想連結軸(LinkAxis0001，LinkAxis0002)を使用します。

シーケンサCPUからVirtualAxisおよびLinkAxisを駆動させるプログラムに，AxisRef型の構造体を使用します。公開ラベル設定に，それぞれのAxisRefを追加します。

（4.3.2項参照）

(3) ローカルラベル

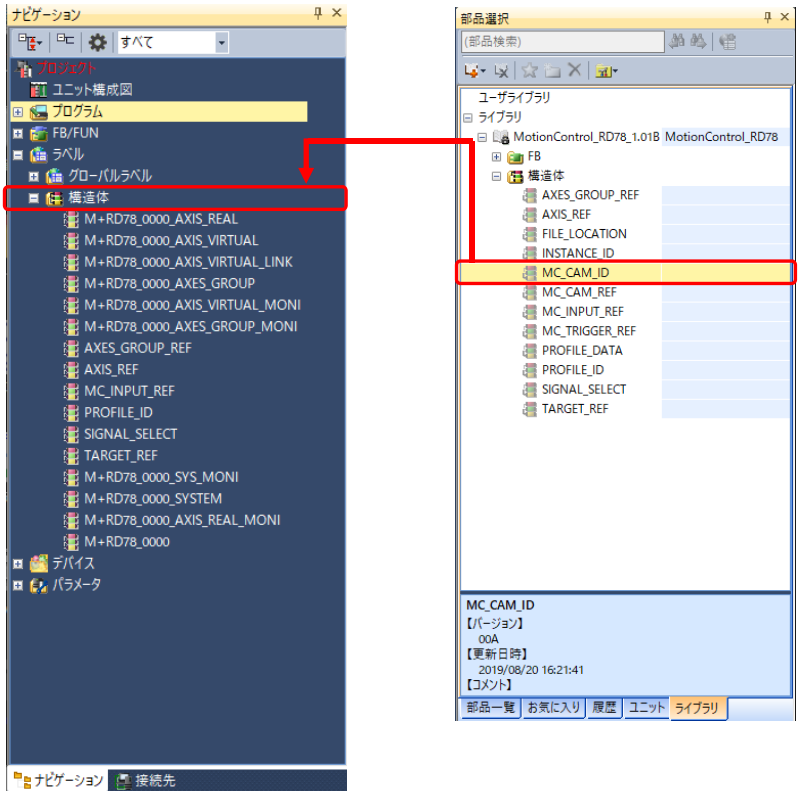
| | ラベル名 | データ型 | | Japanese/日本語(表示対象) |
|---|---|---|---|---|
| 1 | leVelocity | 倍精度実数 | ... | 速度 |
| 2 | leAcceleration | 倍精度実数 | ... | 加速度 |
| 3 | leDeceleration | 倍精度実数 | ... | 減速度 |
| 4 | leJerk | 倍精度実数 | ... | ジャーク |
| 5 | lePosition | 倍精度実数 | ... | 移動量 |
| 6 | MC_GearIn_1 | MC_GearIn | ... | ギア動作FB |
| 7 | MC_CombineAxes_1 | MC_CombineA | ... | 加減算位置決めFB |
| 8 | MC_CamIn_1 | MC_CamIn | ... | カム動作FB |
| 9 | CamID | MC_CAM_ID | ... | カムID |
| 10 | bInSync | ビット | ... | カム動作FB inSync出力 |
| 11 | MC_MoveRelative_1 | MC_MoveRelative | ... | 相対値位置決めFB |
| 12 | bError | ビット | ... | 相対値位置決めFB Error出力 |
| 13 | bDone1 | ビット | ... | 相対値位置決めFB1 Done出力 |
| 14 | MC_MoveRelative_2 | MC_MoveRelative | ... | 相対値位置決めFB |
| 15 | bDone2 | ビット | ... | 相対値位置決めFB2 Done出力 |
| 16 | bStopDone | ビット | ... | 軸停止完了 |
| 17 | bSyncMove | ビット | ... | 相対位置決め始動 |
| 18 | bValueSet | ビット | ... | 変数セット完了 |
| 19 | bCommandAborted | ビット | ... | 相対値位置決めFB 実行中断出力 |
| 20 | MC_Stop_1 | MC_Stop | ... | 軸停止FB1 |
| 21 | MC_Stop_2 | MC_Stop | ... | 軸停止FB2 |
| 22 | MC_Stop_3 | MC_Stop | ... | 軸停止FB3 |
| 23 | | | ... | |

※

②

①

①プログラムエディタに各FBをドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。
※：MC_CAM_ID型構造体の作成方法
部品選択ウィンドウの[ライブラリ]タブから，“ライブラリ”⇒“MotionControl_RD78_****”⇒“構造体”と進み，“MC_CAM_ID”を選択します。“MC_CAM_ID”をナビゲーションウィンドウの“ラベル”⇒“構造体”にドラッグ＆ドロップします。これにより，[構造体]のツリー下に“MC_CAM_ID”が登録され，ラベルエディタのデータ型でMC_CAM_IDが選択できるようになります。

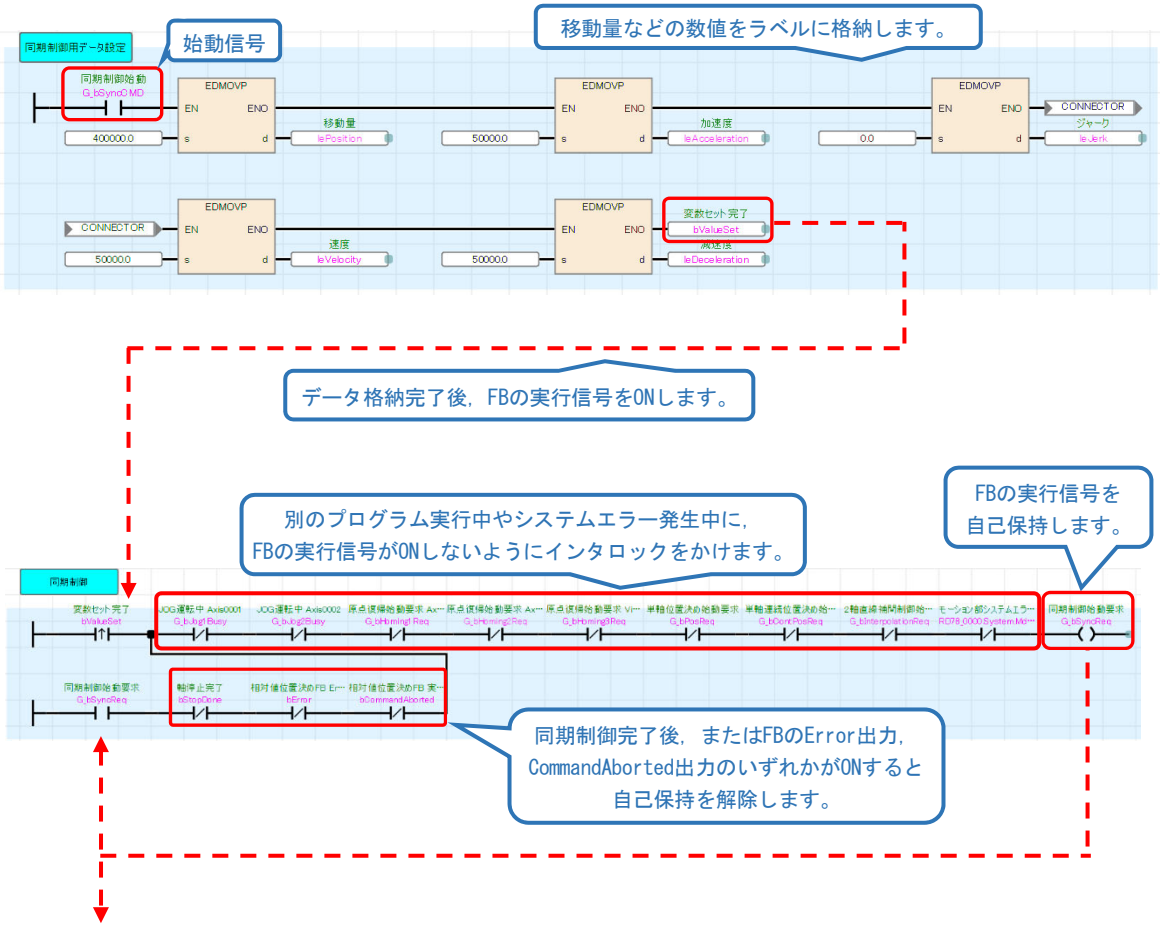


(4) プログラム例

同期制御始動がONになると，位置決め用の各データを各ラベルに格納します。
全データの格納が完了したら，FB(MC_GearIn,MC_CombineAxes,MC_CamIn)の実行要求を自己保持します。
Axis0002の軸状態が“7:同期運転中”になったことを確認し，Master(主軸)となるAxis0001を駆動させます。このとき，Axis0002は演算プロファイルに従って動作します。Axis0001の位置決め完了後，仮想ドライブ軸VirtualAxis0001を駆動させます。
このときAxis0001は停止していますが，Axis0002は演算プロファイルに従って動作します。
仮想ドライブ軸VirtualAxis0001の位置決めが完了するか，エラー発生または実行中断で，実ドライブ軸Axis0002と，仮想連結軸LinkAxis0001，LinkAxis0002のMC_Stopを実行し，同期運転を解除します。実ドライブ軸Axis0002の同期状態が解除されたら，始動信号をリセットします。
別のプログラム実行中やシステムエラー発生中に，同期制御が始動しないようにインタロックをかけます。

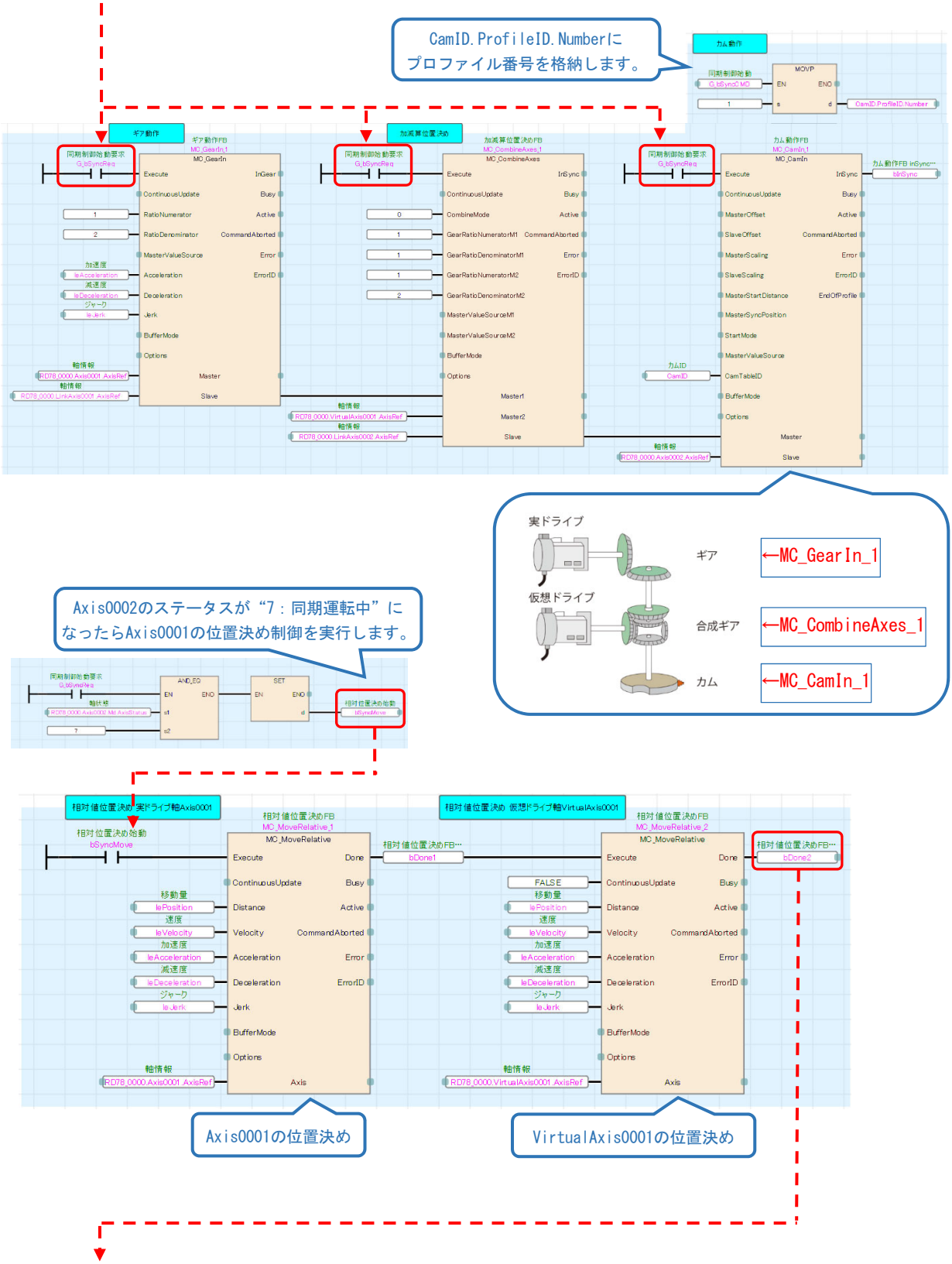
［始動信号］

| | |
|---|---|
| 同期制御始動 | G_bSyncCMD |



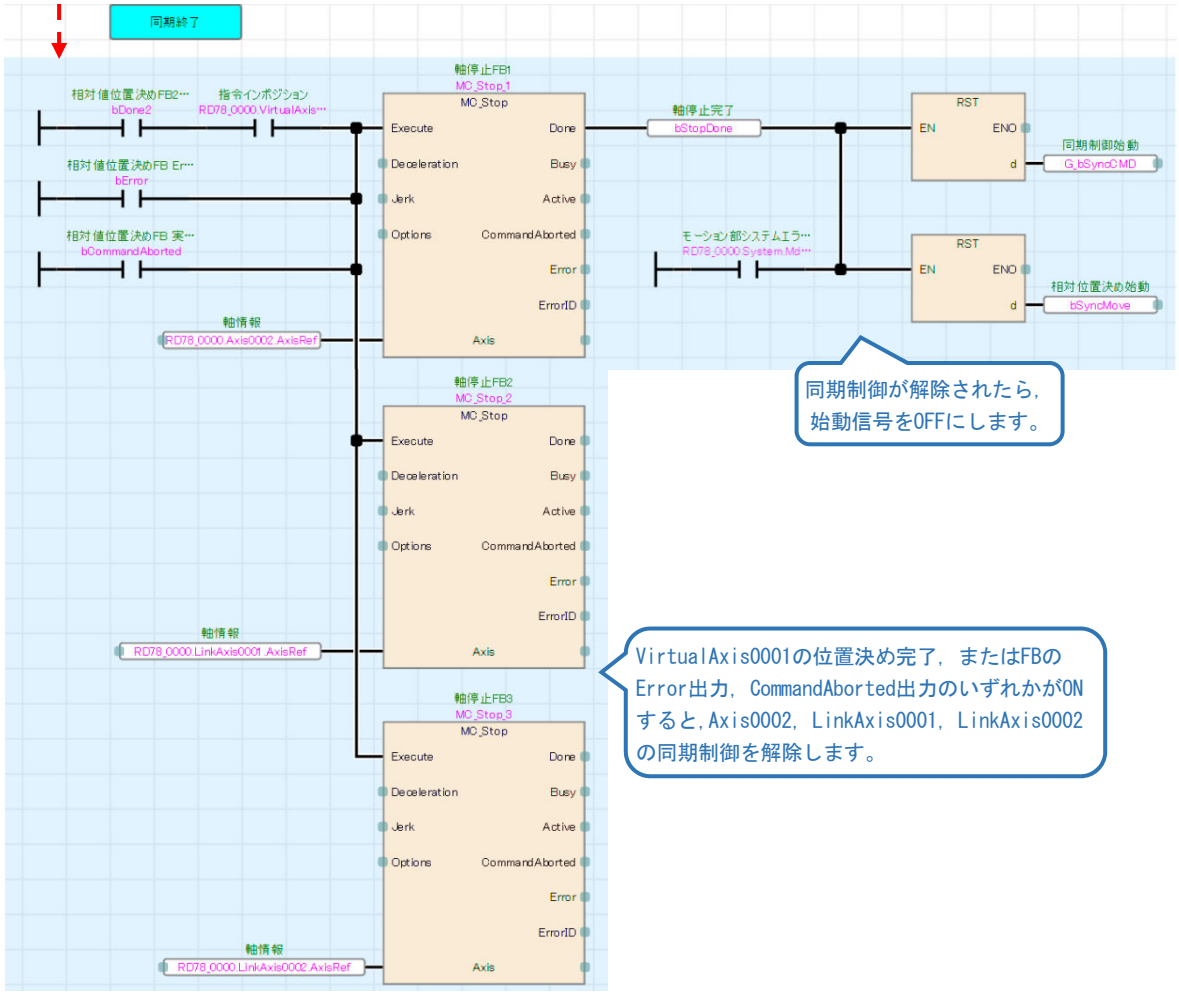

(次ページへ)

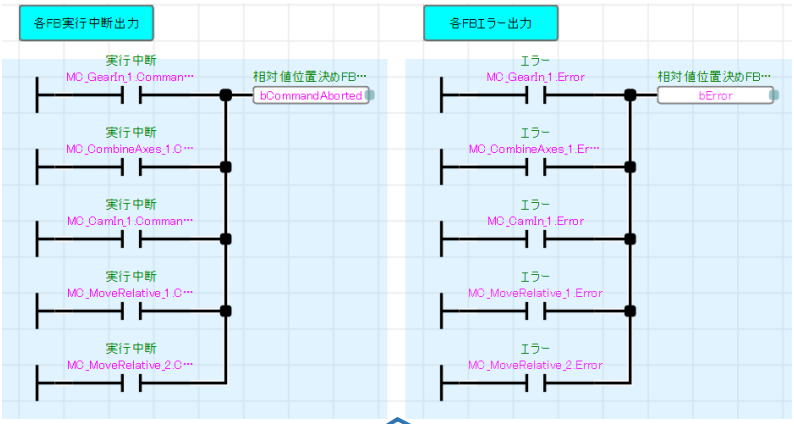
(前ページより)


CamID.ProfileID.Numberに
プロファイル番号を格納します。
ギア動作
加減算位置決め
カム動作
MC_GearIn
MC_CombineAxes
MC_CamIn
実ドライブ
仮想ドライブ
ギア
合成ギア
カム
←MC_GearIn_1
←MC_CombineAxes_1
←MC_CamIn_1
Axis0002のステータスが“7：同期運転中”に
なったらAxis0001の位置決め制御を実行します。
相対値位置決め 実ドライブ軸Axis0001
相対値位置決め 仮想ドライブ軸VirtualAxis0001
MC_MoveRelative
Axis0001の位置決め
VirtualAxis0001の位置決め

（次ページへ）

（前ページより）




同期制御用FBと位置決め制御用FBのCommandAborted出力，Error出力を
それぞれOR条件で取得し，インタロック条件や自己保持解除に使用します。

## 4.12　エラーリセット（プログラム名：ErrorReset）

各軸で発生したエラーをリセットします。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(管理系) | MC_Reset | 軸のエラー・警告をリセットします。 |
| | MC_GroupReset | 軸グループのエラー・警告をリセットします。 |
| | MCv_MotionErrorReset | モーションシステムの全エラー・警告をリセットします。 |

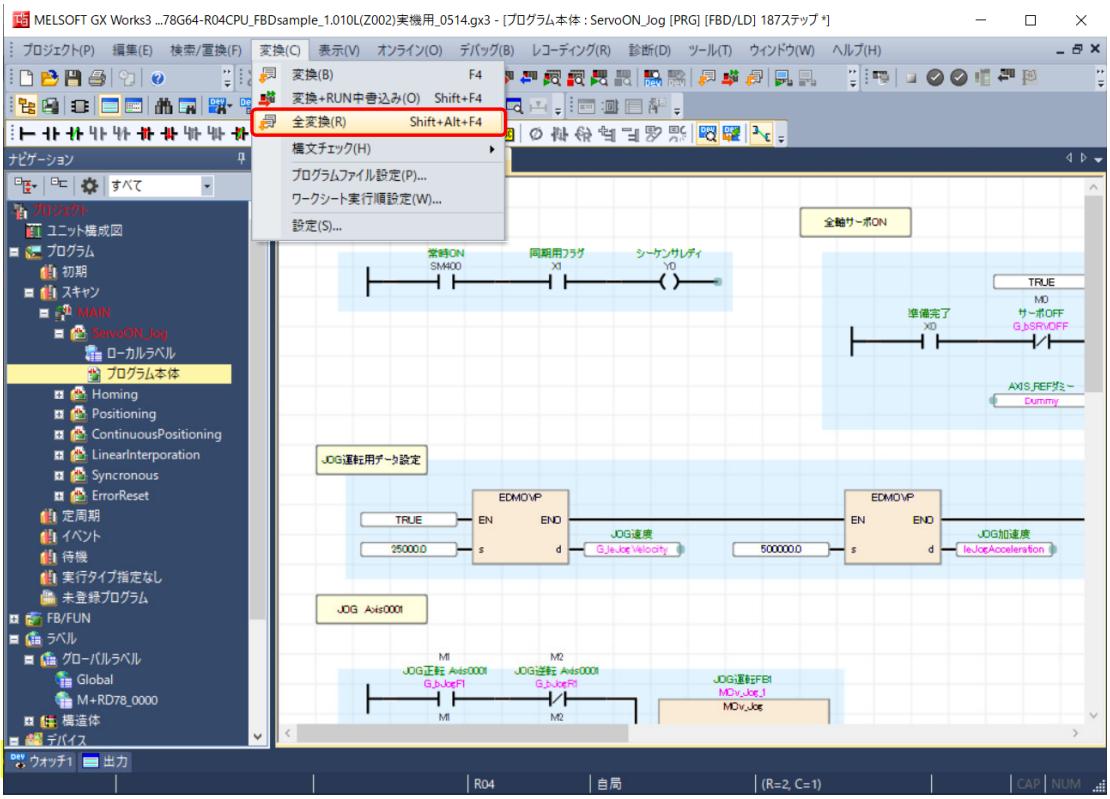
### (2) プログラム例

エラーリセットのプログラム例を示します。
エラーリセット信号のラベルをONすると，MC_Reset，MC_GroupResetを実行します。
また，システムエラーリセット信号のラベルをONすると，MCv_MotionErrorResetを実行します。

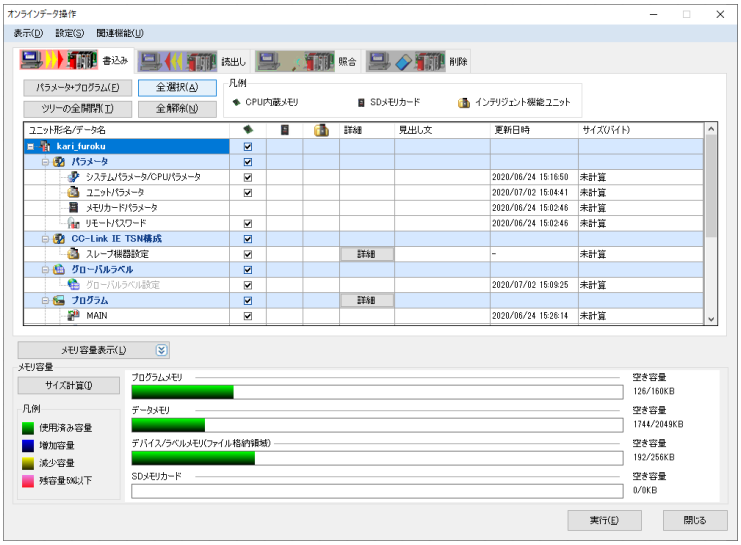
## 4.13　動作確認

### 4.13.1　プログラムの変換と書き込み

シーケンサCPUにプログラムを書き込みます。
プログラム作成後，［変換］⇒［全変換］を選択し，プログラムの全変換を行ないます。




全変換でエラーがないことを確認したら，［オンライン］⇒［シーケンサへの書込み］よりシーケンサへプログラムを書き込んでください。
モーションユニット側のパラメータや公開ラベルの設定を変更した場合は，モーションユニット側も全変換や公開ラベルの反映後，書き込みを行なってください。



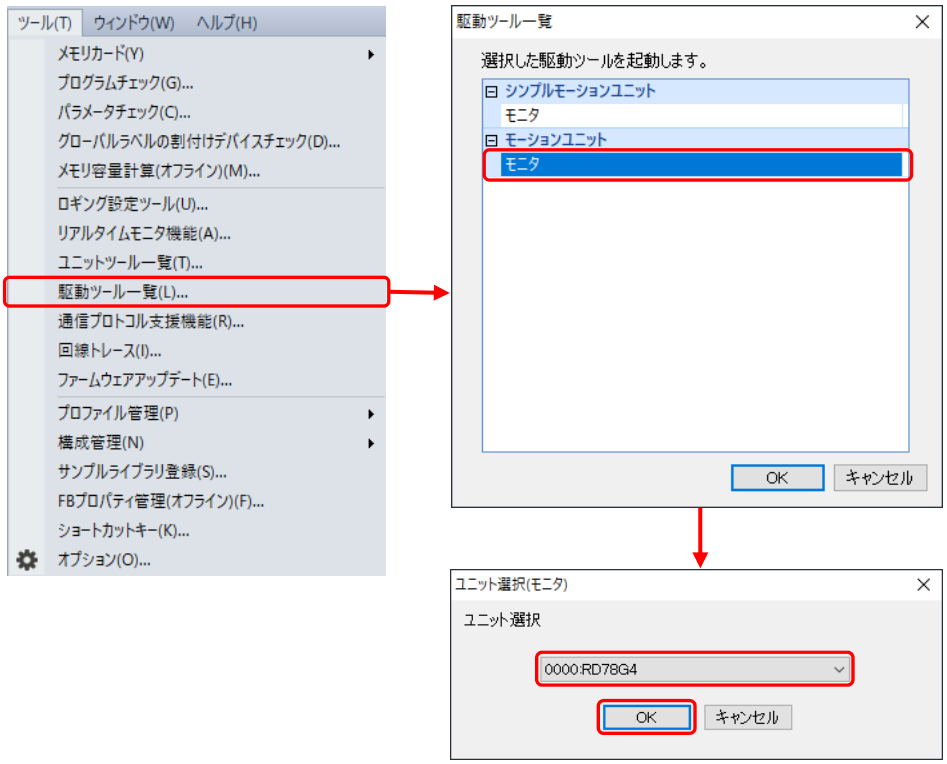
### 4.13.2　軸モニタ

全ての運転軸の現在値，エラーコードを一括してモニタ表示します。
システム稼働中に，現在値の確認，エラーの発生の有無を確認することができます。

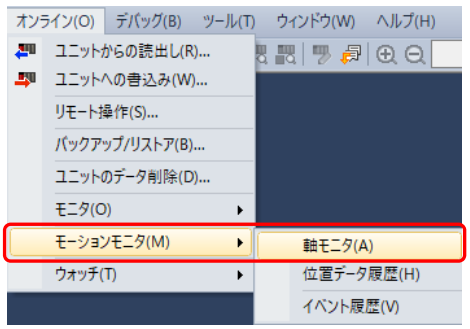
#### (1) 表示方法

以下の2通りの方法で，軸モニタ画面を表示させます。

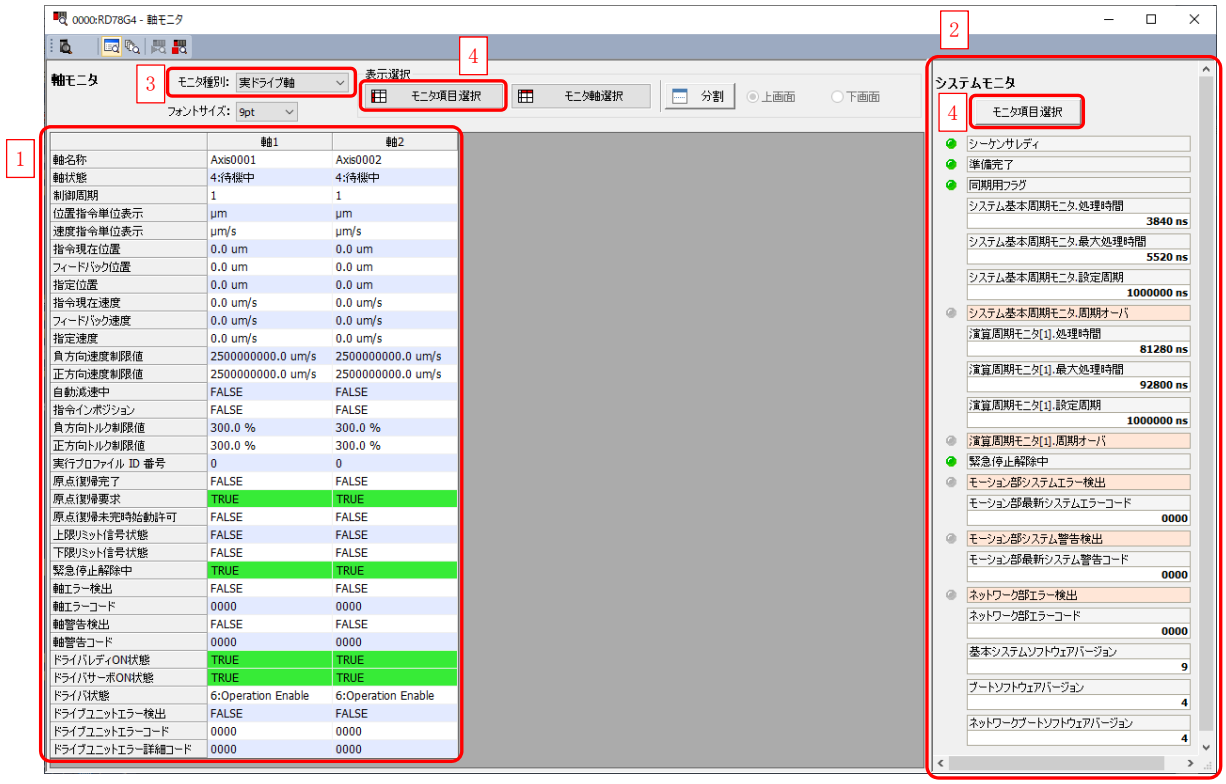
①MELSOFT GX Works3で[ツール]⇒[駆動ツール]を選択し，“駆動ツール一覧”画面からモーションユニットの[モニタ]をダブルクリックします。“ユニット選択(モニタ)”画面でモーションユニットを選択して，[OK]ボタンをクリックします。




②モーション制御設定機能で，[オンライン]⇒[モーションモニタ]⇒[軸モニタ]を選択します。



(2) 表示内容




| No. | 内　容 |
|---|---|
| 1 | 各軸のモニタ項目を表示します。 |
| 2 | システムのモニタ項目を表示します。 |
| 3 | モニタする軸の種別を切り換えます。 |
| 4 | モニタする内容を追加/削除することができます。 |

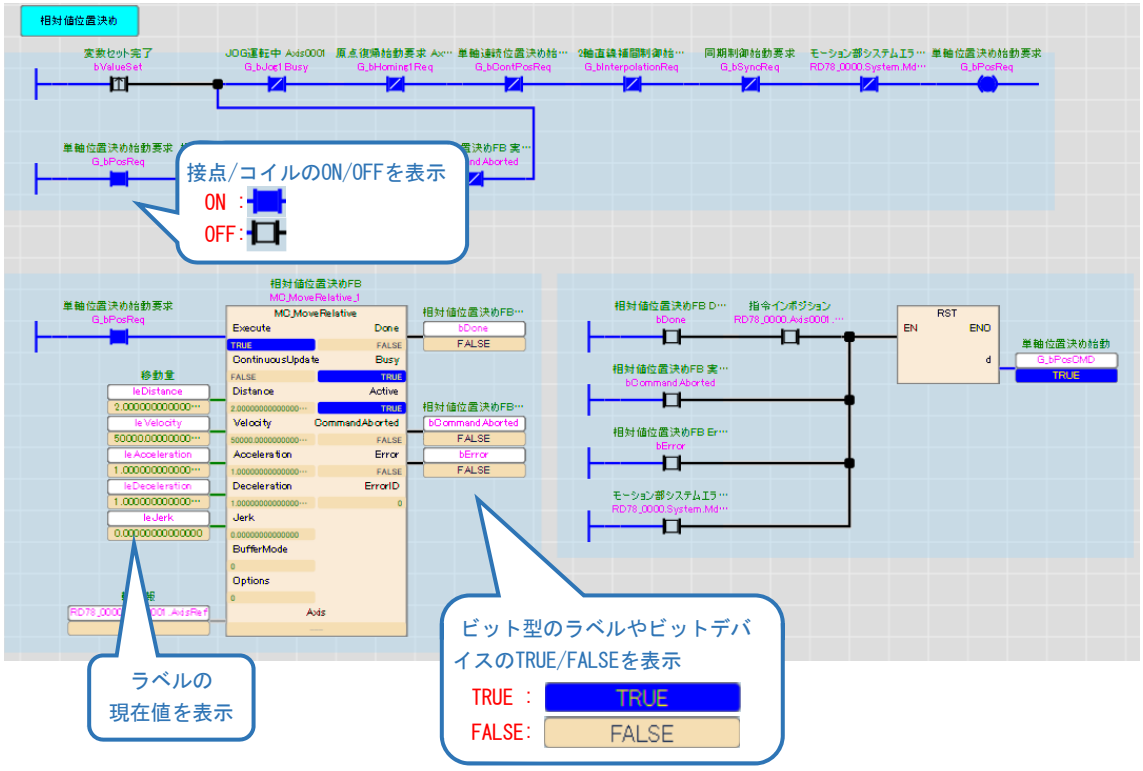
### 4.13.3　プログラムモニタ

モニタ機能を使用して，実行プログラムをプログラムエディタ上で確認します。

(1) 表示方法

MELSOFT GX Works3で，[オンライン]⇒[モニタ]⇒[モニタ開始(全ウィンドウ)]を選択するか，ツールバーの[ ]アイコンをクリックします。

(2) 表示内容

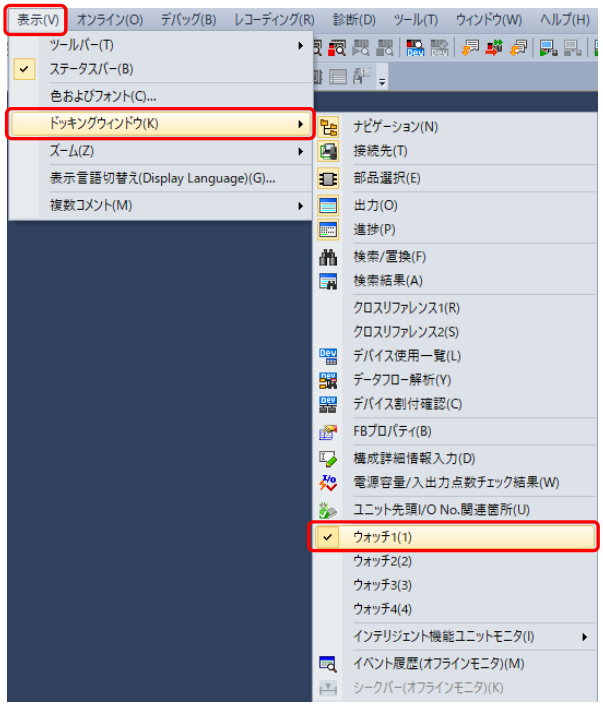


### 4.13.4　ウォッチ

ウォッチ機能を使用して，デバイス，ラベルを登録して現在値を確認します。
確認対象は，ウォッチウィンドウに登録します。

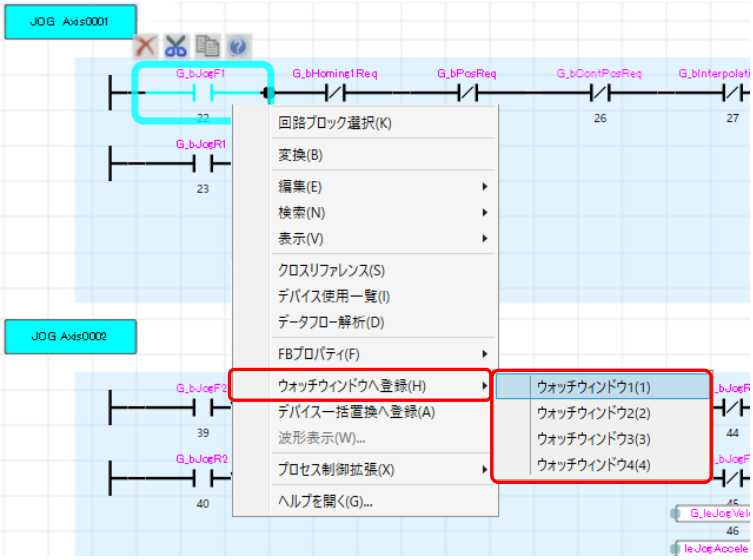
#### (1) 表示方法

MELSOFT GX Works3で，[表示]⇒[ドッキングウィンドウ]⇒[ウォッチ1(1)～ウォッチ4(4)]を選択します。




#### (2) ウォッチウィンドウへの登録

ウォッチウィンドウの名称欄に，デバイス番号，またはラベルや構造体の名前を入力するか，プログラムエディタ上で登録するラベルや構造体を選択し，右クリック⇒[ウォッチウィンドウへ登録]⇒[ウォッチウィンドウ1(1)～ウォッチ4(4)]を選択します。



### (3) ウォッチの開始

MELSOFT GX Works3の[オンライン]⇒[ウォッチ]⇒[ウォッチ開始]を選択します。

### (4) 現在値の変更

ウォッチ中は，“現在値”に変更する値を直接入力します。
ビットデバイスの場合は行を選択し，[Shift]+ダブルクリック，または[Shift]+[Enter]で，ON/OFFを変更することができます。

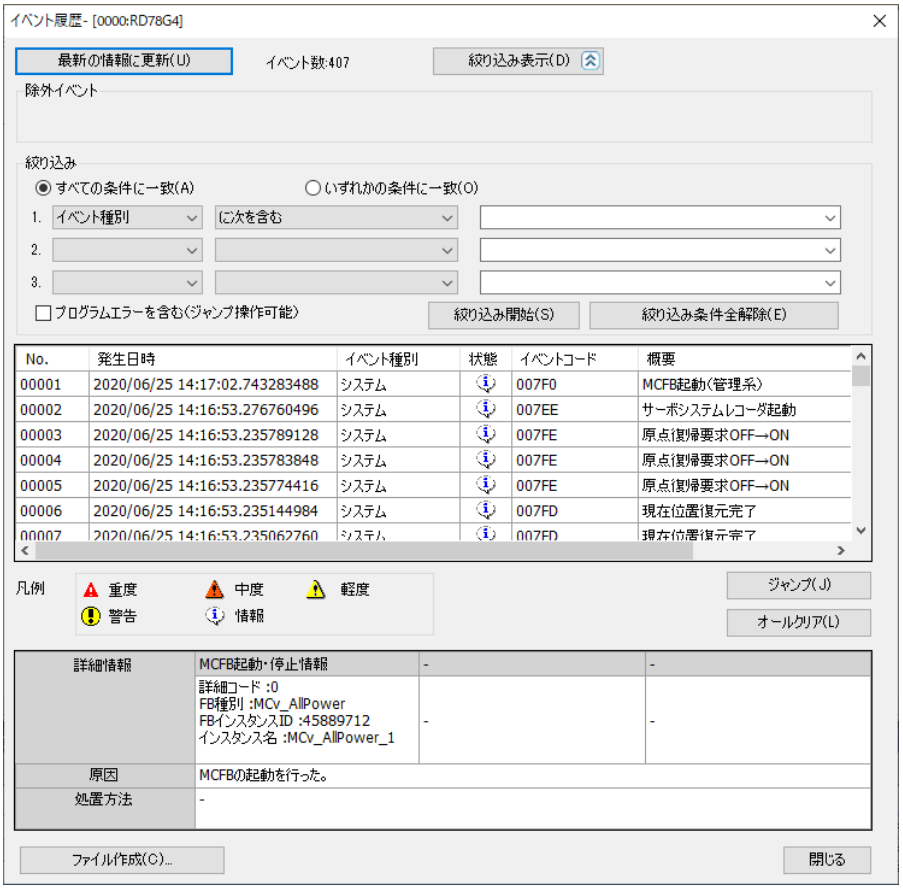
| 名称 | 現在値 | 表示形式 | データ型 | Japanese/日本語 | 強制入出力状態 | 実行条件付きデバイステスト |
|---|---|---|---|---|---|---|
| G_bSRVOFF | -- | 2進数 | ビット | サーボOFF | -- | -- |
| G_bJogF1 | -- | 2進数 | ビット | JOG正転指令 Axis0001 | -- | -- |
| G_bJogR1 | -- | 2進数 | ビット | JOG逆転指令 Axis0001 | -- | -- |
| G_bJogF2 | -- | 2進数 | ビット | JOG正転指令 Axis0002 | -- | -- |
| G_bJogR2 | -- | 2進数 | ビット | JOG逆転指令 Axis0002 | -- | -- |
| G_bHoming1CMD | -- | 2進数 | ビット | 原点復帰指令 Axis0001 | -- | -- |
| G_bHoming2CMD | -- | 2進数 | ビット | 原点復帰指令 Axis0002 | -- | -- |
| G_bHoming3CMD | -- | 2進数 | ビット | 原点復帰指令 VirtualAxis0001 | -- | -- |
| G_bPosCMD | -- | 2進数 | ビット | 単軸位置決め始動 | -- | -- |
| G_bContPosCMD | -- | 2進数 | ビット | 単軸連続位置決め始動 | -- | -- |
| G_bInterpolationCMD | -- | 2進数 | ビット | 2軸直線補間制御始動 | -- | -- |
| G_bSyncCMD | -- | 2進数 | ビット | 同期制御始動 | -- | -- |
| G_bErrorReset | -- | 2進数 | ビット | エラーリセット | -- | -- |

**[ポイント]**
サンプルプログラムには，ウォッチウィンドウ1に“RD78_0000.G_bStopSignalX”と“RD78_0000.G_bStopSignalY”が登録されています。プログラム動作中にこれらをONすることで，プログラムを中断することができます。

### 4.13.5　イベント履歴

モーションユニット側のイベント履歴は，モーション制御設定機能で，[オンライン]⇒[モーションモニタ]⇒[イベント履歴]を選択して表示される“イベント履歴”画面で確認できます。エラーが発生した場合，詳細情報が参照できます。

また，ここで記録される発生日時は，サーボアンプに記録されているアラーム発生日時と同期していますので，あわせて参照してください。



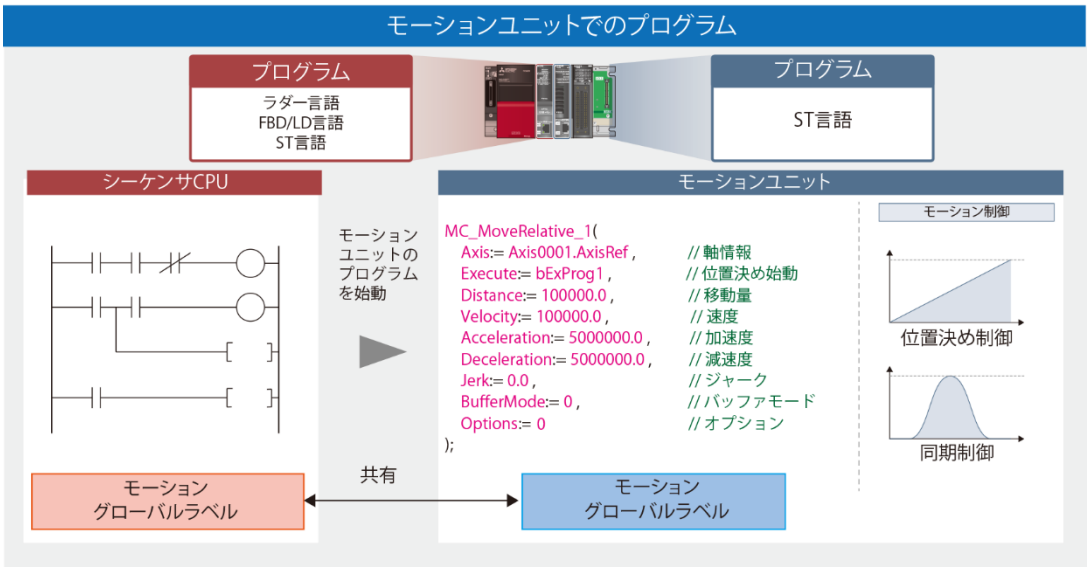

シーケンサCPU側のイベント履歴は，MELSOFT GX Works3の[診断]⇒[システムモニタ]で表示される“システムモニタ”画面の”イベント履歴”ボタンから確認できます。

# メ　モ

# 第５章　シーケンサCPUとモーションユニットに分散したプログラミング

シーケンサCPUとモーションユニット両方に分散してプログラミングする方法を説明します。
本章で説明するサンプルプログラムは，三菱電機FAサイトよりダウンロードできます。

■ファイル名：　R04-78G4_STsample_*****.gx3　　(*****にはバージョンが入ります。)




本サンプルプログラムを実際のシステムへ流用するときは，対象システムにおいて，制御に問題ないことを十分検証してください。
また，対象システムにおいて，インタロック条件が必要な箇所を検討し追加してください。

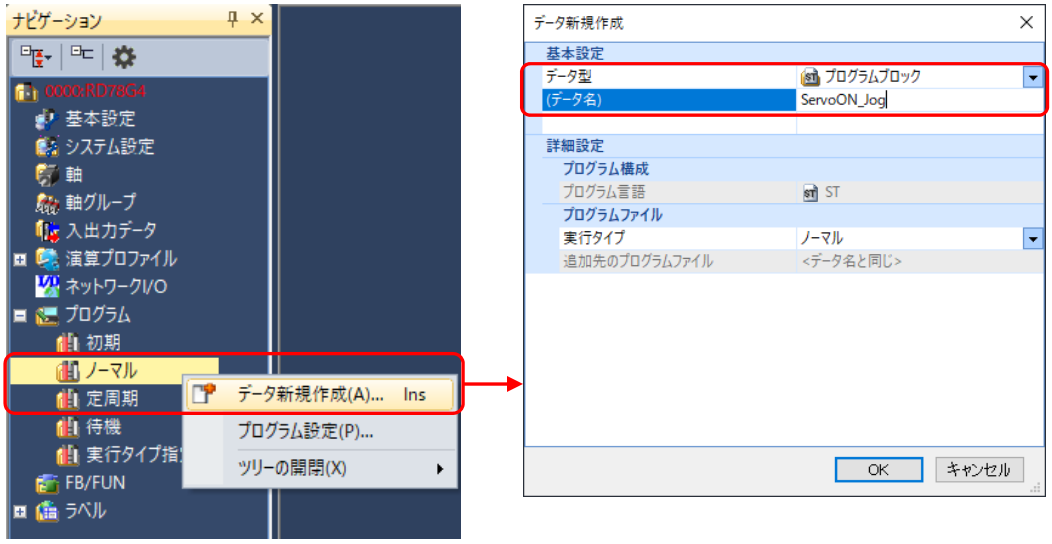
## 5.1　モーションユニットのプログラム作成手順

モーション制御設定機能で，モーションユニットのプログラミングを行ないます。

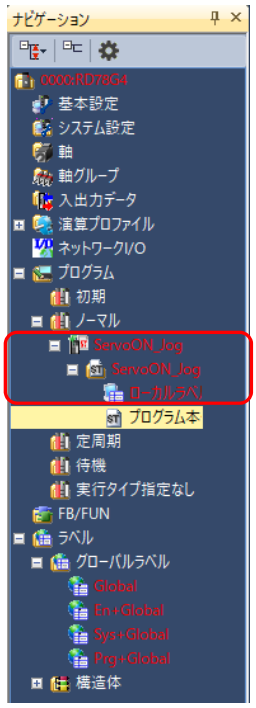
### 5.1.1　プログラムブロックの作成

①モーション制御設定機能のナビゲーションウィンドウで，“プログラム”⇒“プログラム実行タイプ(画面上：ノーマル)”を右クリックして，“データ新規作成”をクリックします。

②“データ新規作成”画面で，データ名を設定します。データ名には，半角英数字のみ使用可能です。




③ナビゲーションウィンドウにプログラムブロックが追加されます。



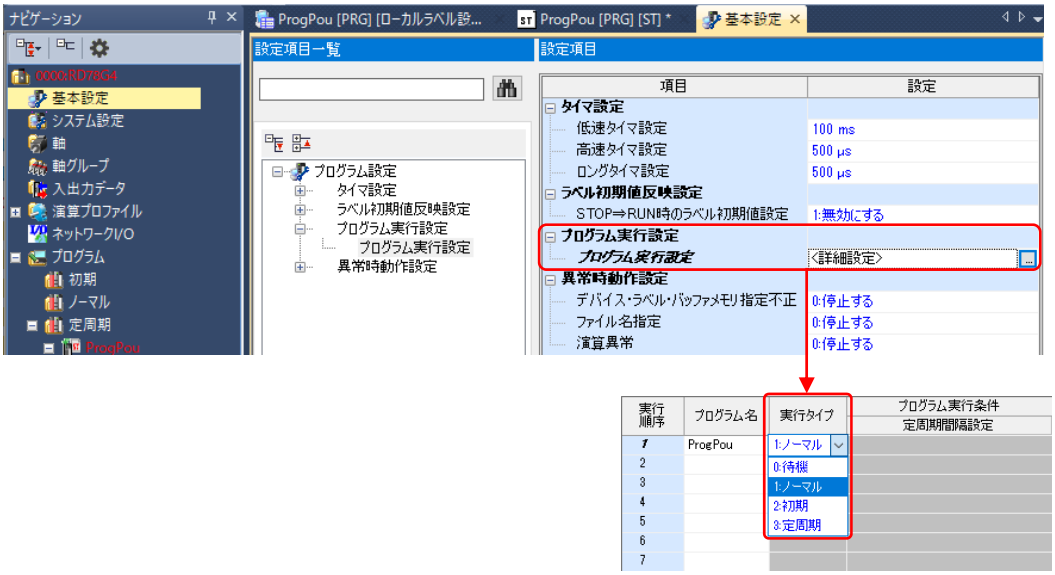
### 5.1.2　プログラム実行タイプ

プログラムの実行タイプには，以下の4種類があります。

| 実行タイプ | 内　容 |
|---|---|
| 0：待機 | 要求があったときにプログラムを実行<br>PSCAN 命令により，他の実行中プログラムからノーマル実行タイププログラムに変更して実行 |
| 1：ノーマル | モーションシステムのノーマルタスクにて，プログラムを実行 |
| 2：初期 | シーケンサレディ[Y0]OFF→ON 時に 1 回のみプログラムを実行 |
| 3：定周期 | 指定時間ごとに実行するプログラム<br>（第 1 演算周期 62.5[us]※1～6000[ms]）<br>ただし，演算周期より短い設定はできません。 |

※1：us はマイクロ秒を示します。

ナビゲーションウィンドウで，実行タイプを変更するプログラムブロック名をドラッグ＆ドロップして別の実行タイプに移動するか，“基本設定”⇒“プログラム実行設定”で変更します。



### 5.1.3 FBの入力方法

本項では，JOG運転用のFB(MCv_Jog)を例に，以下のプログラムを作成する手順を説明します。
サンプルプログラムは，このプログラムからインタロック条件や公開ラベル設定を追加します。

```
1  //Axis0001 JOG
2  MCv_Jog_1(
3      Axis         := Axis0001.AxisRef ,
4      JogForward   := G_bJogF1CMD ,
5      JogBackward  := G_bJogR1CMD ,
6      Velocity     := G_leJogVelocity ,
7      Acceleration:= leJogAcceleration ,
8      Deceleration:= leJogDeceleration ,
9      Jerk         := leJogJerk ,
10     Busy         => G_bJog1Busy
11 );
12
```

#### (1) ラベルの準備

FBの入出力信号に使用するラベルを準備します。本項では，MCv_Jogの入力信号の速度，加速度，減速度，ジャークにラベルを使用します。各ラベルを，グローバルラベルに登録するか，ローカルラベルに登録するかをあわせて検討します。

本項では例として，グローバルラベルには，JOG指令，JOG運転中信号，JOG速度を登録します。JOG速度は，GOTなどの外部機器から設定するケースを想定してグローバルラベルに登録します。このプログラム内でのみ使用するJOG加速度，JOG減速度，JOGジャークなどは，5.1.1項で作成したプログラム“ServoON_Jog”のローカルラベルに登録します。
ラベルの命名規則は，5.2節を参照してください。

［グローバルラベル］

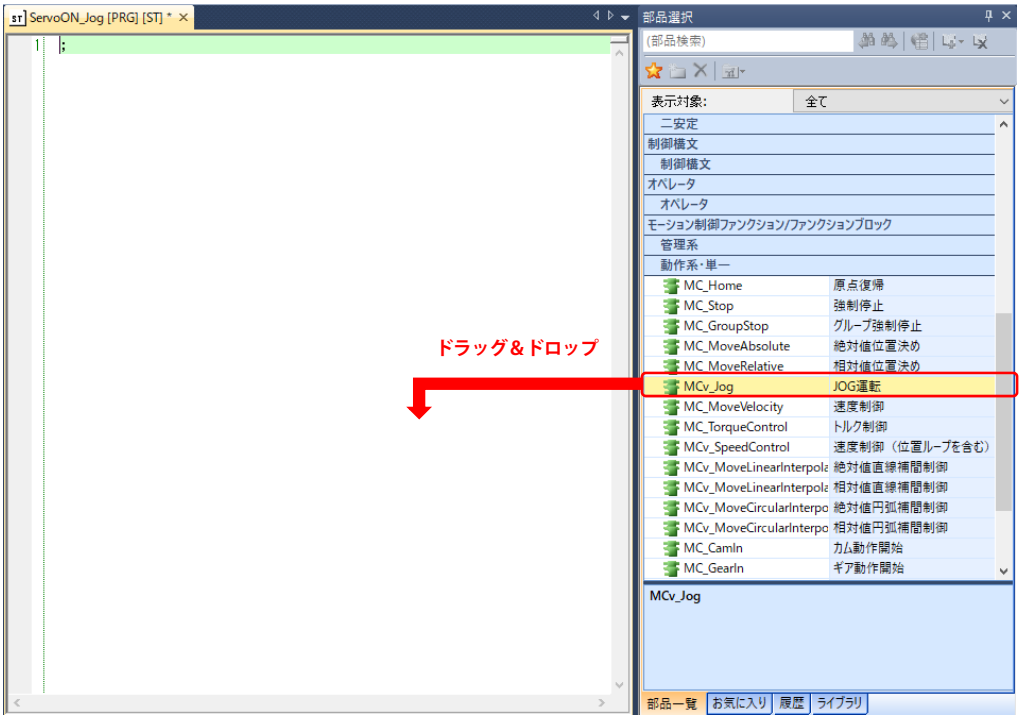
| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | G_bJogF1CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0001 | | 無効 | － |
| 2 | G_bJogR1CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0001 | | 無効 | － |
| 3 | G_bJog1Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0001 | | 無効 | － |
| 4 | G_leJogVelocity | 倍精度実数 | VAR_GLOBAL | | | JOG速度 | | 無効 | － |
| 5 | | | | | | | | | |

［ローカルラベル］

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | leJogAcceleration | 倍精度実数 | VAR | | | JOG加速度 |
| 2 | leJogDeceleration | 倍精度実数 | VAR | | | JOG減速度 |
| 3 | leJogJerk | 倍精度実数 | VAR | | | JOGジャーク |
| 4 | | | | | | |

### (2) FBの入力

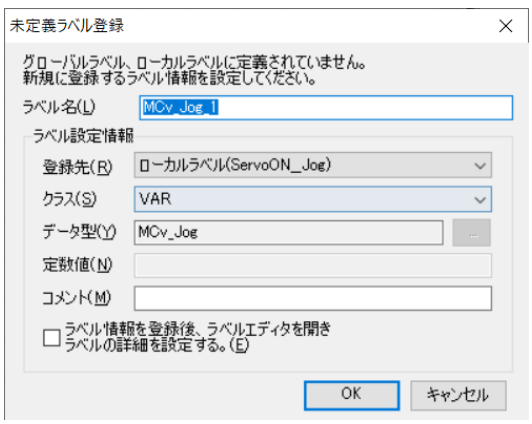
部品選択ウィンドウの[部品一覧]タブで“動作系・単一”⇒“MCv_Jog”をプログラムエディタにドラッグ＆ドロップします。



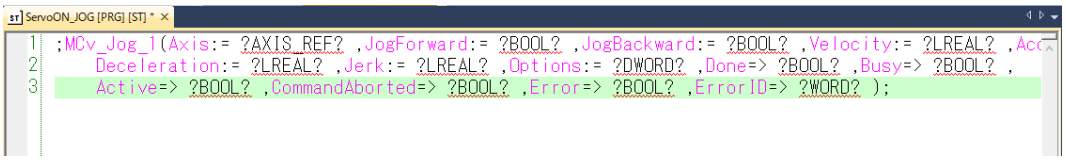

FBは，以下の3つのグループに分類されています。

| グループ | 内　容 |
|---|---|
| Management<br>(管理系) | 軸または軸グループを引数にとり，実行によって軸状態，または軸グループ状態が変化しないモーション制御 FB |
| Operation-Individual<br>(動作系・単一) | 軸または軸グループを引数にとり，実行によって軸状態，または軸グループ状態が変化するモーション制御 FB |
| StandardFB<br>(一般 FB) | 軸または軸グループを引数にとらないモーション制御 FB |

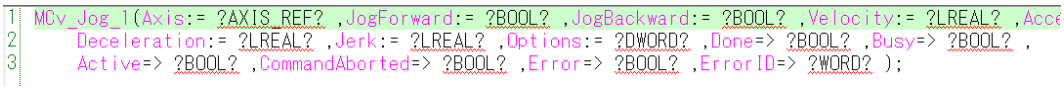
“未定義ラベル登録”画面を表示し，FBのラベル名と登録先，および必要に応じてコメントを入力します。ここでは，全て初期値とします。



[OK]ボタンをクリックして，プログラムエディタ上に[MCv_Jog]を表示します。




先頭のセミコロンは不要なため，削除します。（ドラッグ＆ドロップする位置によっては，FB末尾にセミコロンが2つになりますが，その場合は1つ削除します。）




FB内に改行やインデントは任意に入力できます。
ここでは，入出力信号1つごとに改行とインデントを下図のとおり追加します。

```
MCv_Jog_1(
    Axis:= ?AXIS_REF? ,
    JogForward:= ?BOOL? ,
    JogBackward:= ?BOOL? ,
    Velocity:= ?LREAL? ,
    Acceleration:= ?LREAL? ,
    Deceleration:= ?LREAL? ,
    Jerk:= ?LREAL? ,
    Options:= ?DWORD? ,
    Done=> ?BOOL? ,
    Busy=> ?BOOL? ,
    Active=> ?BOOL? ,
    CommandAborted=> ?BOOL? ,
    Error=> ?BOOL? ,
    ErrorID=> ?WORD?
);
```

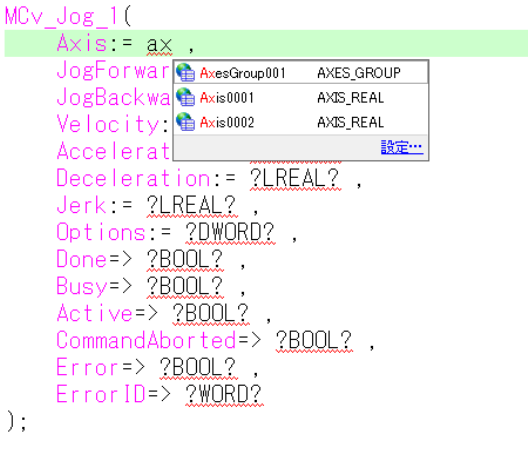
### (3) 入出力信号の入力

?AXIS_REF?，?BOOL? などの文字列をラベルに書き換えます。

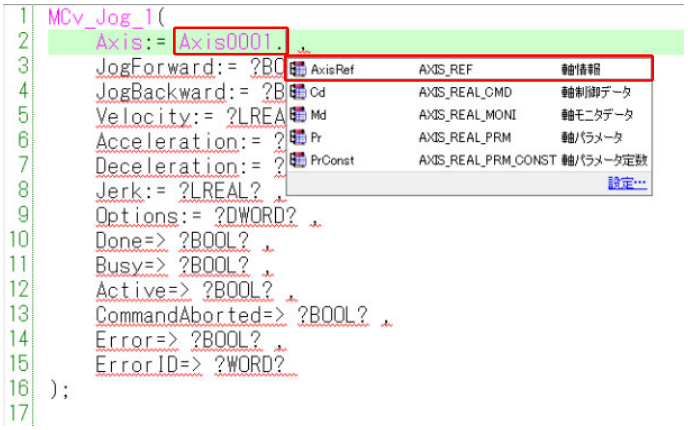
(a) AxisRef型構造体

入力変数“Axis”に，軸1の軸情報であるAxisRef型構造体を入力する方法を説明します。

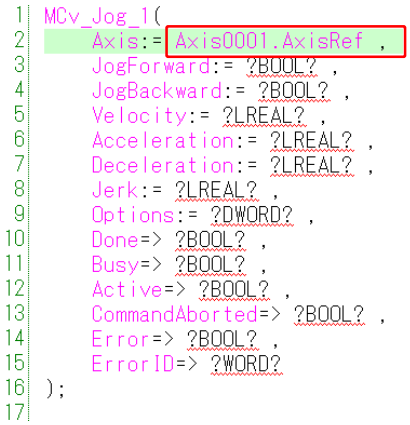
①?AXIS_REF?の代わりに“ax”を入力すると，登録されているラベルが表示されます。



②Axis0001を選択して“.”を入力すると，Axis0001の構造体のメンバが表示されます。



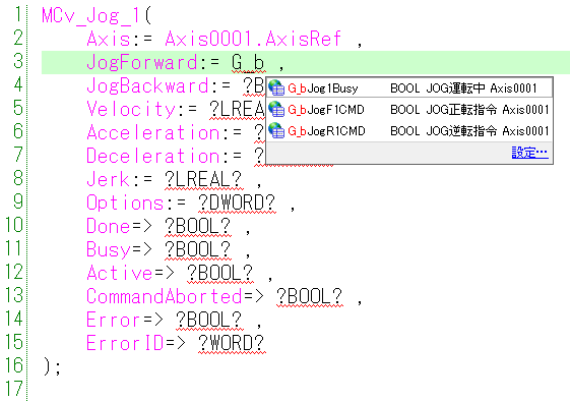

③表示された候補から「AxisRef」を選択します。




(b) グローバルラベル・ローカルラベル

①JogForward入力にグローバルラベルG_bJogF1CMDを入力します。

“JogForward:=”の次に?BOOL?の代わりに「G_b」と入力すると，登録されているラベルが表示されます。



②表示された候補から「G_bJogF1CMD」を選択します。

```
MCv_Jog_1(
    Axis:= Axis0001.AxisRef ,
    JogForward:= G_bJogF1CMD ,
    JogBackward:= ?BOOL? ,
    Velocity:= ?LREAL? ,
    Acceleration:= ?LREAL? ,
    Deceleration:= ?LREAL? ,
    Jerk:= ?LREAL? ,
    Options:= ?DWORD? ,
    Done=> ?BOOL? ,
    Busy=> ?BOOL? ,
    Active=> ?BOOL? ,
    CommandAborted=> ?BOOL? ,
    Error=> ?BOOL? ,
    ErrorID=> ?WORD?
);

```

③同様の手順で，JogBackward，Velocity，Acceleration，Deceleration，Jerk，Busyの各信号に，(1)で準備したラベルを入力します。

```
MCv_Jog_1(
    Axis:= Axis0001.AxisRef ,
    JogForward:= G_bJogF1CMD ,
    JogBackward:= G_bJogR1CMD ,
    Velocity:= G_leJogVelocity ,
    Acceleration:= leJogAcceleration ,
    Deceleration:= leJogDeceleration ,
    Jerk:= leJogJerk ,
    Options:= ?DWORD? ,
    Done=> ?BOOL? ,
    Busy=> G_bJog1Busy ,
    Active=> ?BOOL? ,
    CommandAborted=> ?BOOL? ,
    Error=> ?BOOL? ,
    ErrorID=> ?WORD?
);

```

(c) 入出力信号の省略

初期値から変更しない，または使用しないFBの入力信号は省略できます。
この例の場合，Options入力は初期値(0)から変更なし，Done，Active，CommandAborted，Error，ErrorIDの各出力信号は使用しないため，削除します。
FBの末尾に「,」は不要なので，削除します。

```
MCv_Jog_1(
    Axis:= Axis0001.AxisRef ,
    JogForward:= G_bJogF1CMD ,
    JogBackward:= G_bJogR1CMD ,
    Velocity:= G_leJogVelocity ,
    Acceleration:= leJogAcceleration ,
    Deceleration:= leJogDeceleration ,
    Jerk:= leJogJerk ,
    Busy=> G_bJog1Busy ,
);

```

FB末尾の「,」は削除

(d) 書式の整理

必要に応じてコメントの追加やインデントの追加を行ない，プログラムを整理します。

```
1  //Axis0001 JOG
2  MCv_Jog_1(
3      Axis         := Axis0001.AxisRef ,
4      JogForward   := G_bJogF1CMD ,
5      JogBackward  := G_bJogR1CMD ,
6      Velocity     := G_leJogVelocity ,
7      Acceleration:= leJogAcceleration ,
8      Deceleration:= leJogDeceleration ,
9      Jerk         := leJogJerk ,
10     Busy         => G_bJog1Busy
11 );
12
```

以上で，プログラムの入力は完了です。

### 5.1.4　ENUM列挙子

軸状態(Axis0001.Md.AxisStatus)やバッファモード(BufferMode)などの定数をENUM列挙子として定義しています。ENUM列挙子を使用したプログラミングが可能です。

#### (1) ENUM列挙子の入力

[FBの引数のENUM列挙子の例]

BufferModeは“MC_BUFFER_MODE__mc*****”というラベルで定義されています。
(BufferModeの詳細は，5.9節参照)

```
14  //相対値位置決め
15  MC_MoveRelative_1(
16      Axis            := Axis0001.AxisRef ,
17      Execute         := bExecute_P ,
18      Distance        := leDistance ,
19      Velocity        := leVelocity ,
20      Acceleration    := leAcceleration ,
21      Deceleration    := leDeceleration ,
22      Jerk            := leJerk ,
23      BufferMode      := mc_buf ,
24      Done            => bDone ,
25      CommandAborted  => bComma
26      Error           => bError
27  );
28
29  //シーケンサに送る <= 位置決め
30  G_bPosDone := (bDone & Axis0
31
32
```

MC_BUFFER_MODE__mcAborting　INT Aborting
MC_BUFFER_MODE__mcBlendingHigh　INT BlendingHigh
MC_BUFFER_MODE__mcBlendingLow　INT BlendingLow
MC_BUFFER_MODE__mcBlendingNext　INT BlendingNext
MC_BUFFER_MODE__mcBlendingPre...　INT BlendingPrevious
MC_BUFFER_MODE__mcBuffered　INT Buffered
設定...

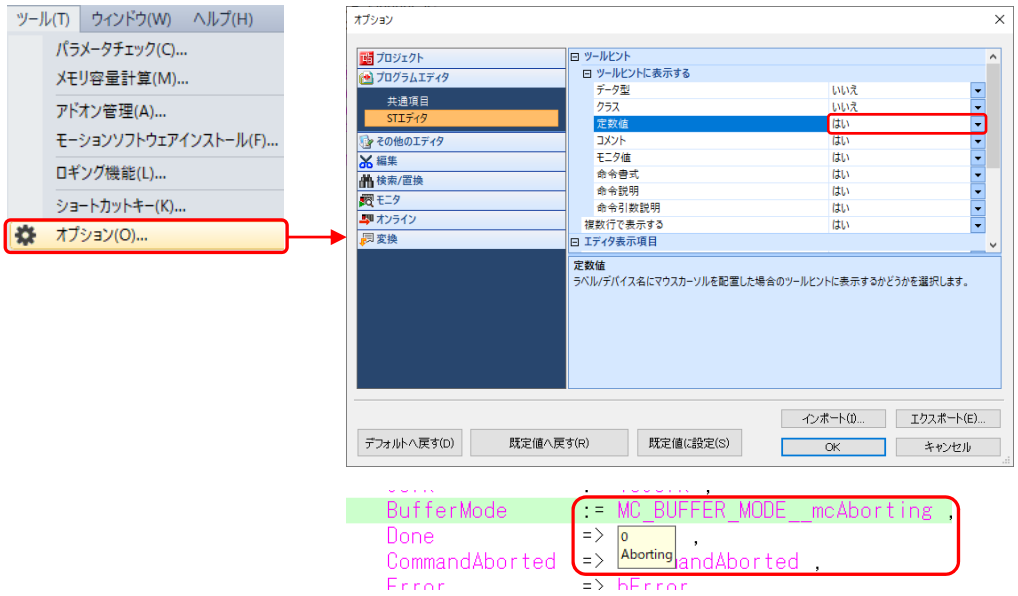
[軸変数のENUM列挙子の例]

軸ステータスは“MC_AXIS_STATUS__*****”というラベルで定義されています。

```
62
63  //入力軸始動
64  SET(G_bSyncCMD & (Axis0002.Md.AxisStatus = MC_AXIS_STATUS__SynchronizedMotion), bSyncMoveCMD);
65
```

#### (2) ENUM列挙子の数値確認

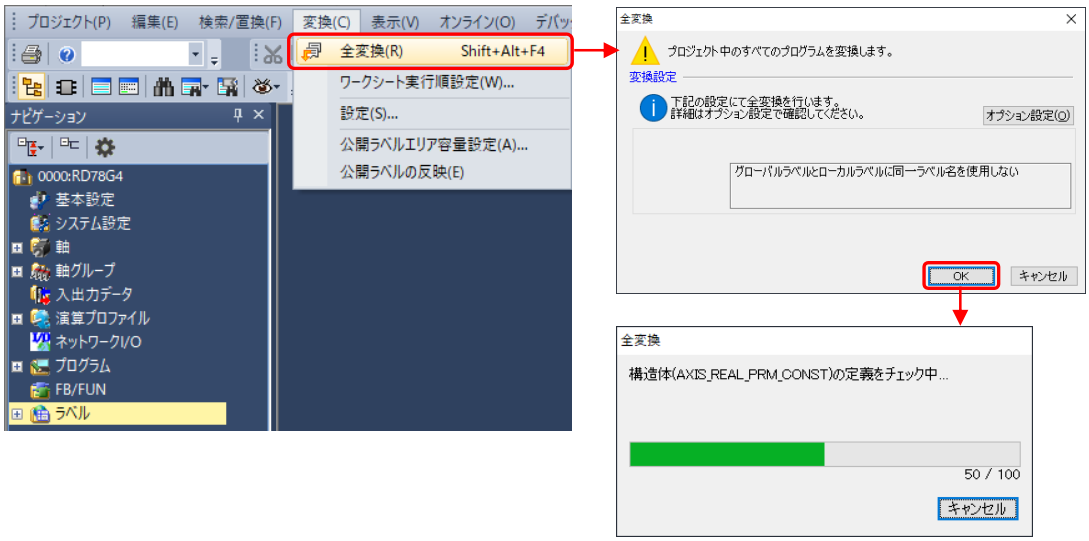
①[ツール]⇒[オプション]をクリックして，オプション画面を表示します。
②STエディタの定数値表示を“はい”にして，[OK]ボタンをクリックします。
③プログラムのENUM列挙子にマウスカーソルを重ねると，数値を確認できます。



## 5.1.5　全変換，公開ラベルの反映

### (1) 全変換

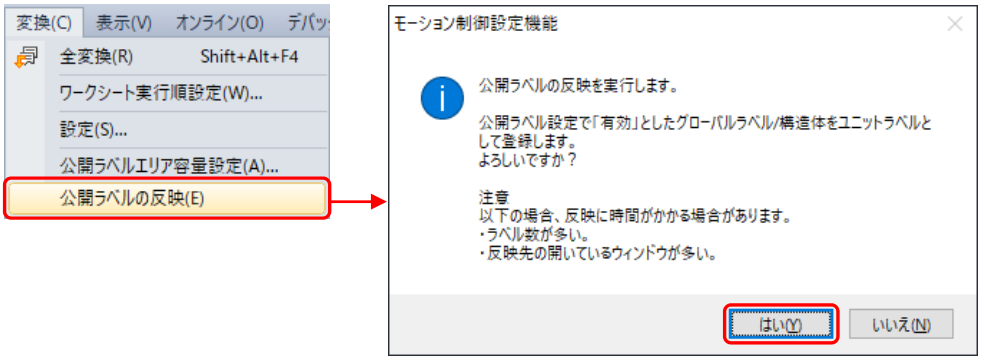
モーション制御設定機能で，[変換]⇒[全変換]をクリックして，確認メッセージを表示します。[OK]ボタンをクリックして，プログラム等を全変換します。




### (2) 公開ラベルの反映

公開ラベルを使用している場合は，[変換]⇒[公開ラベルの反映]をクリックして，確認メッセージを表示します。

[OK]ボタンをクリックして，シーケンサCPUに公開ラベルを反映します。（3.14.2項参照）




公開ラベルの反映後，シーケンサCPU側のプログラムを編集してください。

## 5.2 ラベルの命名規則

本書では，プログラム例に使用するラベルに接頭語を付けてデータ型を表現しています。

| データ型 | | 値の範囲 | 接頭語 | |
|---|---|---|---|---|
| | | | ローカル | グローバル |
| ビット | BOOL | 0 (FALSE), 1 (TRUE) | b | G_b |
| ワード［符号なし］<br>ビット列［16 ビット］ | WORD | 0～65535 | u | G_u |
| ダブルワード［符号なし］<br>ビット列［32 ビット］ | DWORD | 0～4294967295 | ud | G_ud |
| ワード［符号付き］ | INT | -32468～32767 | w | G_w |
| ダブルワード［符号付き］ | DINT | -2147483648～2147483647 | d | G_d |
| 単精度実数 | REAL | $-2^{128}$～$-2^{-126}$, 0, $2^{-126}$～$2^{128}$ | e | G_e |
| 倍精度実数 | LREAL | $-2^{1024}$～$-2^{-1022}$, 0, $2^{-1022}$～$2^{1024}$ | le | G_le |
| 時間 | TIME | T#-24d20h31m23s648ms～<br>T#24d20h31m23s647ms | tm | G_tm |
| タイマ | TIMER | TIMER の構造体<br>S：接点<br>C：コイル<br>N：現在値 | td | G_td |

［ローカルラベル例］

ビット　bMoveCMD
倍精度実数　lePosition
配列　wAxes[16]
タイマ　tdTimer1

［グローバルラベル例］

ビット　G_bJogF1
倍精度実数　G_leVelocity

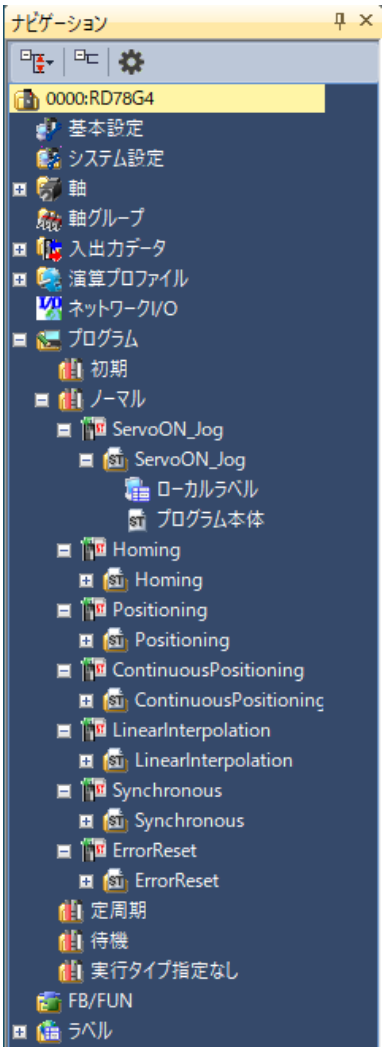
## 5.3　プロジェクト構成

### 5.3.1　プログラム名

本章で作成するプログラム例を以下に示します。

| | |
|---|---|
| [シーケンサCPU]<br> <br> | プログラム名：MAIN<br>シーケンサのデバイスでモーションユニットの始動信号をON/OFFする<br><br>接点処理が主な内容のため，ラダーで作成します。 |

［モーションユニット］




プログラム名：ServoON_Jog
　全軸サーボON，JOG運転

プログラム名：Homing
　原点復帰

プログラム名：Positioning
　単軸位置決め制御

プログラム名：ContinuousPositioning
　単軸の連続位置決め制御

プログラム名：LinearInterpolation
　2軸の直線補間制御

プログラム名：Synchronous
　同期制御

プログラム名：ErrorReset
　エラーリセット

### 5.3.2　グローバルラベル，公開ラベル設定

サンプルプログラムのグローバルラベルの設定例を示します。
ローカルラベルについては各プログラムを参照してください。

#### (1) シーケンサCPU

シーケンサCPU側は，モーションユニットの公開ラベルを使用します。

#### (2) モーションユニット

モーションユニット側のグローバルラベル設定，公開ラベル設定は以下のとおりです。

[Global]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | G_bSRVONCMD | ビット | VAR_GLOBAL | | | サーボオンオフ | | 有効 | WRITE(⇒モーション) |
| 2 | G_bHoming1CMD | ビット | VAR_GLOBAL | | | 原点復帰指令Axis0001 | | 有効 | WRITE(⇒モーション) |
| 3 | G_bHoming2CMD | ビット | VAR_GLOBAL | | | 原点復帰指令Axis0002 | | 有効 | WRITE(⇒モーション) |
| 4 | G_bHoming3CMD | ビット | VAR_GLOBAL | | | 原点復帰指令(仮想ドライブ軸 | | 有効 | WRITE(⇒モーション) |
| 5 | G_bPosCMD | ビット | VAR_GLOBAL | | | 単軸位置決め始動 | | 有効 | WRITE(⇒モーション) |
| 6 | G_bContPosCMD | ビット | VAR_GLOBAL | | | 単軸連続位置決め始動 | | 有効 | WRITE(⇒モーション) |
| 7 | G_bInterpolationCMD | ビット | VAR_GLOBAL | | | 補間制御始動 | | 有効 | WRITE(⇒モーション) |
| 8 | G_bSyncCMD | ビット | VAR_GLOBAL | | | 同期制御始動 | | 有効 | WRITE(⇒モーション) |
| 9 | G_bResetCMD | ビット | VAR_GLOBAL | | | エラーリセット | | 有効 | WRITE(⇒モーション) |
| 10 | G_bMotionResetCMD | ビット | VAR_GLOBAL | | | コントローラ エラーリセット | | 有効 | WRITE(⇒モーション) |
| 11 | G_bJogF1CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 12 | G_bJogR1CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0001 | | 有効 | WRITE(⇒モーション) |
| 13 | G_bJogF2CMD | ビット | VAR_GLOBAL | | | JOG正転指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 14 | G_bJogR2CMD | ビット | VAR_GLOBAL | | | JOG逆転指令 Axis0002 | | 有効 | WRITE(⇒モーション) |
| 15 | G_leJogVelocity | 倍精度実数 | VAR_GLOBAL | | | JOG速度 | | 有効 | WRITE(⇒モーション) |
| 16 | G_bHoming1Done | ビット | VAR_GLOBAL | | | 原点復帰完了 Axis0001 | | 有効 | READ(モーション⇒) |
| 17 | G_bHoming2Done | ビット | VAR_GLOBAL | | | 原点復帰完了 Axis0002 | | 有効 | READ(モーション⇒) |
| 18 | G_bHoming3Done | ビット | VAR_GLOBAL | | | 原点復帰完了 VirtualAxis0001 | | 有効 | READ(モーション⇒) |
| 19 | G_bPosDone | ビット | VAR_GLOBAL | | | 単軸位置決め完了 | | 有効 | READ(モーション⇒) |
| 20 | G_bContPosDone | ビット | VAR_GLOBAL | | | 単軸連続位置決め完了 | | 有効 | READ(モーション⇒) |
| 21 | G_bInterpolationDone | ビット | VAR_GLOBAL | | | 補間制御完了 | | 有効 | READ(モーション⇒) |
| 22 | G_bSyncDone | ビット | VAR_GLOBAL | | | 同期制御完了 | | 有効 | READ(モーション⇒) |
| 23 | G_bJog1Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0001 | | 有効 | READ(モーション⇒) |
| 24 | G_bJog2Busy | ビット | VAR_GLOBAL | | | JOG運転中 Axis0002 | | 有効 | READ(モーション⇒) |
| 25 | G_bStopSignalX | ビット | VAR_GLOBAL | | | 停止指令 Axis0001 | | 有効 | READ(モーション⇒) |
| 26 | G_bStopSignalY | ビット | VAR_GLOBAL | | | 停止指令 Axis0002 | | 有効 | READ(モーション⇒) |
| 27 | | | | | | | | | |

①各プログラムの実行指令となるラベル
②JOG速度を格納する倍精度実数型のラベル
　グローバルラベルに登録することで，GOTや別のプログラムから変更することができます。
③各プログラムの動作完了をシーケンサCPUに伝達するラベル
④JOG動作中にONするラベル
　JOG動作中に，別のプログラムが動作しないようにするためのインタロックとして使用します。
⑤軸停止信号となるラベル
　プログラム動作中にこのラベルをONすると，動作を中断します。（詳細は5.13.4項参照）

[Sys+Global]

システムに関するデータを格納する構造体です。公開ラベルを有効に設定します。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | System | SYSTEM | VAR_GLOBAL | <詳細… | | | | 有効 | － |
| 2 | | | | | | | | | |

[AX+Global]
モーション制御設定機能で軸を設定すると，自動的に登録されます。ラベルエディタ上での操作は必要ありません。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Axis0001 | AXIS_REAL | VAR_GLOBAL | <詳細設… | | [X_Axis] | | 無効 | − |
| 2 | Axis0002 | AXIS_REAL | VAR_GLOBAL | <詳細設… | | [Y_Axis] | | 無効 | − |
| 3 | VirtualAxis0001 | AXIS_VIRTUAL | VAR_GLOBAL | <詳細設… | | [Vir_Axis01] | | 無効 | − |
| 4 | LinkAxis0001 | AXIS_VIRTUAL_LINK | VAR_GLOBAL | <詳細設… | | [Lin_Axis01] | | 無効 | − |
| 5 | LinkAxis0002 | AXIS_VIRTUAL_LINK | VAR_GLOBAL | <詳細設… | | [Lin_Axis02] | | 無効 | − |
| 6 | | | | | | | | | |

[Gr+Global]
モーション制御設定機能で軸グループを設定すると，自動的に登録されます。ラベルエディタ上での操作は必要ありません。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | AxesGroup001 | AXES_GROUP | VAR_GLOBAL | <詳細… | | X-Y Table | | 無効 | − |
| 2 | | | | | | | | | |

[Prf+Global]
モーション制御設定機能でプロファイルデータを登録すると，自動的に登録されます。ラベルエディタ上での操作は必要ありません。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ProfileData0001 | MC_CAM_REF | VAR_GLOBAL | <詳細… | | Cam #1 | | 無効 | − |
| 2 | | | | | | | | | |

[Prg+Global]
モーション制御設定機能でプログラムを作成すると，自動的に登録されます。ラベルエディタ上での操作は必要ありません。

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 備考 | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Servo0N_Jog | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 2 | ErrorReset | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 3 | Homing | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 4 | Positioning | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 5 | Synchronous | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 6 | ContinuousPositioning | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 7 | LinearInterpolation | PROGRAM_INFO | VAR_GLOBAL | <詳細… | | | | 無効 | − |
| 8 | | | | | | | | | |

[En+Global]
ENUM列挙子の定義のためのグループです。このグループはシステムが自動で登録しますので，操作は不要です。

[SYS_MONI構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | Addon_AbsSystem | ADDON_MONI | | | | アドオンAbsSystemモニタ | 無効 | - |
| 2 | Addon_Axis | ADDON_MONI | | | | アドオンAxisモニタ | 無効 | - |
| 3 | Addon_ExternalSignal | ADDON_MONI | | | | アドオンExternalSignalモニタ | 無効 | - |
| 4 | Addon_FileTransfer | ADDON_MONI | | | | アドオンFileTransferモニタ | 無効 | - |
| ≈ | | | | | | | | |
| 24 | Environment_UserRootPath | 文字列(127) | | '' | | ユーザルートパス | 無効 | - |
| 25 | Error | ビット | | 0 | | モーション部システムエラー検出 | 有効 | - |
| 26 | ErrorHistory | ワード[符号なし]/ビット… | | 0 | | エラー履歴情報 | 無効 | - |

[SYSTEM構造体]

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント | 公開ラベル | モーション制御属性 |
|---|---|---|---|---|---|---|---|---|
| 1 | PrConst | SYS_PRM_CONST | | | | システムパラメータ定数 | 無効 | WRITE(⇒モーション): PrConst |
| 2 | Pr | SYS_PRM | | | | システムパラメータ | 無効 | WRITE(⇒モーション): Pr |
| 3 | Md | SYS_MONI | | | | システムモニタデータ | 有効 | READ(モーション⇒): Md |
| 4 | Cd | SYS_CMD | | | | システム制御データ | 無効 | WRITE(⇒モーション): Cd |
| 5 | LOGGING_REALTIME | LOGGING_REALTIME | | | | ロギングリアルタイムモニタ | 無効 | - |
| 6 | LoggingRef | LOGGING_REF(1..10) | | | | ロギング | 無効 | - |
| 7 | | | | | | | | |

**[ポイント]**
モーションユニットの公開ラベルの設定を変更した場合は，必ず公開ラベルの反映を行なってください。(3.14.2項参照)

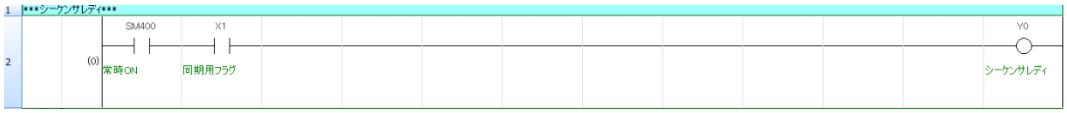
## 5.4　シーケンサレディ（プログラム名：ServoON_Jog）

モーションユニットRD78Gは，シーケンサCPUとのデータ授受に，入力32点，出力32点を使用します。プログラム言語にかかわらず，シーケンサレディ[Y0]をONする必要があります。
シーケンサCPUにてシーケンサレディをONして，モーションユニットを起動します。

### (1) プログラム例

シーケンサCPUの電源投入後，モーションユニットの同期用フラグ[X1]がONになったら，シーケンサレディ信号[Y0]をONします。

[シーケンサCPU]




[モーションユニット]

なし

## 5.5　サーボON（プログラム名：ServoON_Jog）

サーボシステムに接続されている実ドライブ軸のサーボONを行ないます。
サーボONには，全軸サーボON命令と各軸のサーボON命令の2種類があります。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(管理系) | MC_Power | 指定した軸を運転可能状態に切り換えます。 |
| | MCv_AllPower | 全ての軸を運転可能状態に切り換えます。 |

### (2) ローカルラベル

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MCv_AllPower_1 | MCv_AllPower | VAR | | | 全軸サーボオンFB |
| 2 | MCv_Jog_1 | MCv_Jog | VAR | | | JOG運転FB1 Axis0001 |
| 3 | MCv_Jog_2 | MCv_Jog | VAR | | | JOG運転FB2 Axis0002 |
| 4 | eJogAcceleration | 倍 | | | | |
| 5 | eJogDeceleration | 倍精度実数 | VAR | | | JOG減速度 |
| 6 | eJogJerk | 倍精度実数 | VAR | | | JOGジャーク |
| 7 | | | | | | |

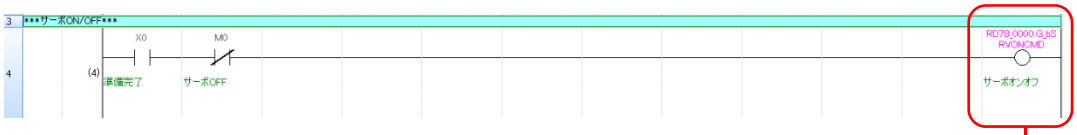
JOG運転のプログラムで使用するラベルです。

プログラムエディタにMCv_AllPowerをドラッグ＆ドロップして，ローカルラベルにMCv_AllPower_1を追加します。（5.1.3項参照）

### (3) プログラム例

シーケンサレディ[Y0]をONすると，準備完了[X0]がONします。それを全軸サーボON信号として使用します。サーボOFFする場合は，[M0]をONします。

[シーケンサCPU]




[モーションユニット]

```
1  //-----全軸サーボON-----
2  MCv_AllPower_1(
3      Enable  := TRUE ,
4      ServoON := G_bSRVONCMD
5  );
6
```

## 5.6 JOG運転（プログラム名：ServoON_Jog）

JOG運転は，JOG正転指令／JOG逆転指令を入力している間，サーボシステムから軸に指令を出力し，指定方向へ軸が動作します。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MCv_Jog | 目標速度に従いJOG運転を実行します。 |

### (2) 加減速処理機能

モーション制御の加減速を，装置に適した加減速カーブに調整する機能です。

(a) 概要

加減速方式は，以下から選択します。

| 加減速方式 | 内　容 |
|---|---|
| 加減速度指定方式<br>(初期値) | FBで指定した加速度，減速度，ジャークを用いて，加速・減速します。 |
| 加減速時間一定方式 | 速度に関係なく，FBで指定した加減速時間を用いて，加速・減速します。 |

(b) 設定方法

MCv_Jogを含む，動作系MCFBのOptions入力によって設定します。

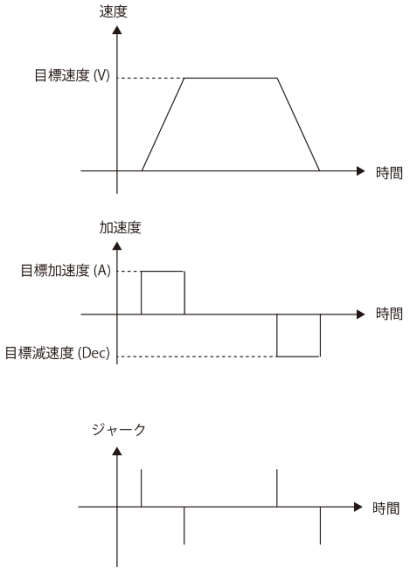
| ビット | 機能説明 |
|---|---|
| 0～2 | 加減速方式設定<br>0：mcAccDec 加減速度指定方式<br>1：mcFixedTime 加減速時間一定方式 |
| 3～15 | 空き |
| 16～31 | 各FBにより機能が異なります。 |

(c) 加減速度指定方式

FBの「加減速方式設定」(Options bit0～2)入力で「0：mcAccDec」を選択して，加速度・減速度・ジャークを設定します。

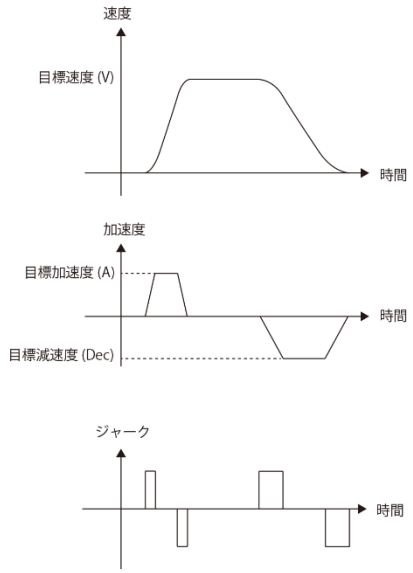
1) 台形加減速

ジャークに「0.0」を指定した場合を台形加減速と呼びます。
速度は台形の波形となります。



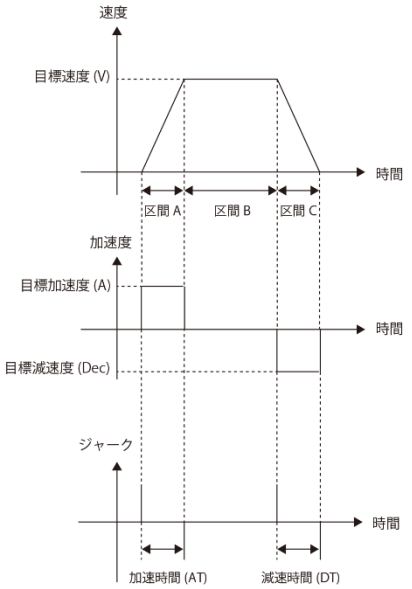

2) ジャーク加減速

ジャークに「0.0」以外を指定した場合をジャーク加減速と呼びます。
速度はS字の波形となります。



(d) 加減速時間一定方式

FBの「加減速方式設定」(Options bit0～2)で「1：mcFixedTime」を選択して，Accelerationに加減速時間を設定します。DecelerationとJerkは使用しません。




(e) FBの入力変数

(b)～(d)までの内容を，FBの入力値ごとに整理した内容を以下に示します。

| 入力変数 | 名　称 | 詳　細 |
|---|---|---|
| Velcotiy | 目標速度 | FB での目標速度を指定します。 |
| Acceleration | 加速度／加減速時間 | FBでの加速度／加減速時間を指定します。<br>・加減速方式が「0：mcAccDec」：[U/s$^2$]単位<br>・加減速方式が「1：mcFixedTime」：[s]単位 |
| Deceleration | 減速度 | FBでの減速度を指定します。<br>・加減速方式が「0：mcAccDec」：[U/s$^2$]単位<br>・加減速方式が「1：mcFixedTime」：使用しません |
| Jerk | ジャーク | FBでのジャークを指定します。<br>・加減速方式が「0：mcAccDec」：[U/s$^3$]単位<br>・加減速方式が「1：mcFixedTime」：使用しません |
| Options | オプション | bit0～2 で加減速方式を指定します。<br>0：mcAccDec 加減速度指定方式<br>1：mcFixedTime 加減速時間一定方式 |

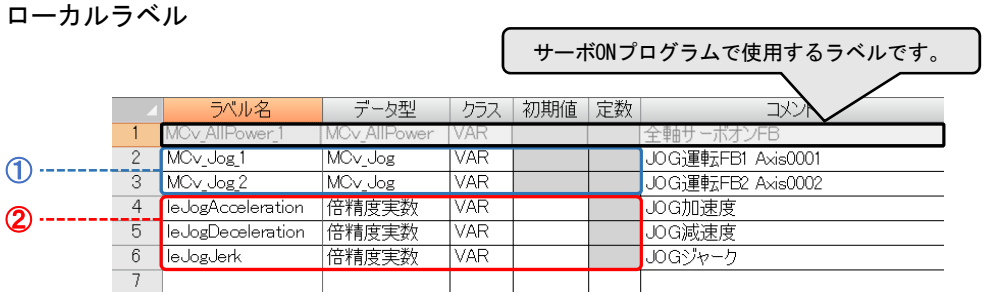
**[ポイント]**

**“0：加減速度指定方式”を選択した場合のAcceleration，Decelerationの値は，実行する加減速時間から，以下のように計算します。**

**[目標速度V[U/s]，加速時間[s]，減速時間[s]の場合]，**

```
Velocity := (目標速度V);
Acceleration := (目標速度V/加速時間);
Deceleration := (目標速度V/減速時間);
Options := (mcAccDec);
```

(3) ローカルラベル




| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MCv_AllPower_1 | MCv_AllPower | VAR | | | 全軸サーボオンFB |
| 2 | MCv_Jog_1 | MCv_Jog | VAR | | | JOG運転FB1 Axis0001 |
| 3 | MCv_Jog_2 | MCv_Jog | VAR | | | JOG運転FB2 Axis0002 |
| 4 | leJogAcceleration | 倍精度実数 | VAR | | | JOG加速度 |
| 5 | leJogDeceleration | 倍精度実数 | VAR | | | JOG減速度 |
| 6 | leJogJerk | 倍精度実数 | VAR | | | JOGジャーク |
| 7 | | | | | | |

①プログラムエディタにFB(MCv_Jog)をドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

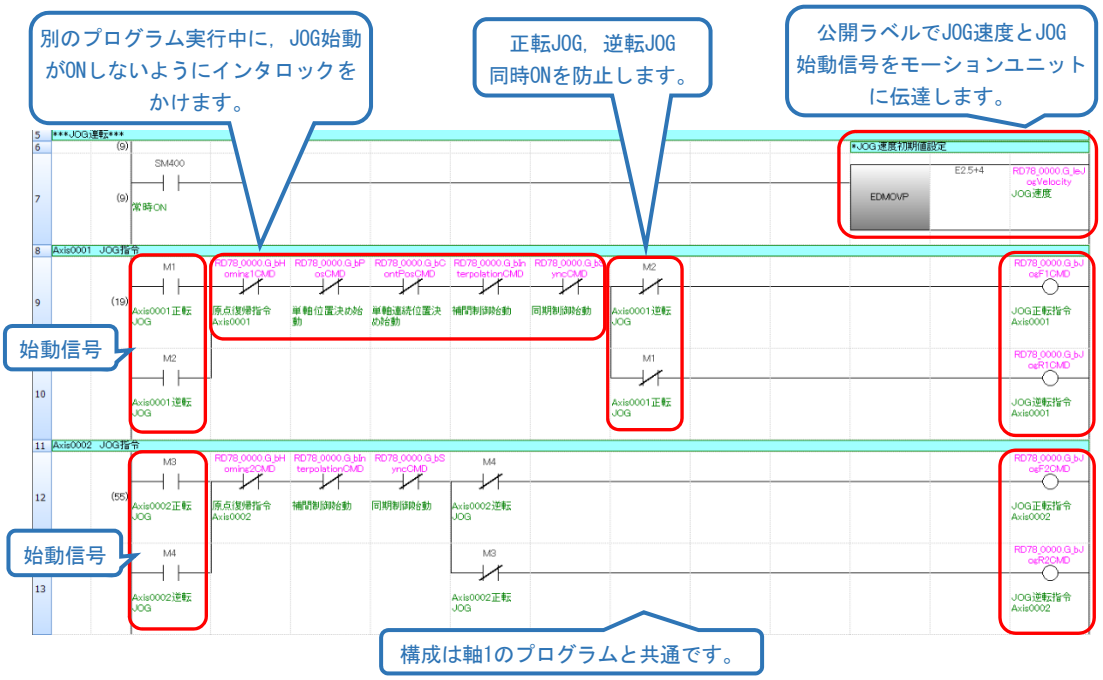
(4) プログラム例

JOG速度は，公開ラベルWrite(⇒ モーション)に設定しているため，シーケンサCPUから設定します。RUN後，シーケンサCPUのプログラムやGOTからJOG速度を変更することができます。
シーケンサCPU側の正転，逆転JOG信号に割り付けたデバイスのON/OFFを，公開ラベルを介してモーションユニットに伝達します。
別のプログラム実行中に，JOG運転が実行されないようにインタロックをかけます。

[始動信号]

| | Axis0001 | Axis0002 |
|---|---|---|
| JOG 正転指令 | M1 | M3 |
| JOG 逆転指令 | M2 | M4 |

[シーケンサCPU]



［モーションユニット］

```
7  //-----JOG運転用データ設定-----
8
9  IF Axis0001.Md.Driver_ServoOn THEN
10     leJogAcceleration := 50000.0;
11     leJogDeceleration := 50000.0;
12     leJogJerk         := 0.0;
13 END_IF;
14
15 //-----Axis0001 JOG運転-----
16
17 MCv_Jog_1(
18     Axis         := Axis0001.AxisRef ,
19     JogForward   := G_bJogF1CMD ,
20     JogBackward  := G_bJogR1CMD ,
21     Velocity     := G_leJogVelocity ,
22     Acceleration := leJogAcceleration ,
23     Deceleration := leJogDeceleration ,
24     Jerk         := leJogJerk ,
25     BUSY         => G_bJog1Busy
26 );
27
28
29 //-----Axis0002 JOG運転-----
30
31 MCv_Jog_2(
32     Axis         := Axis0002.AxisRef ,
33     JogForward   := G_bJogF2CMD ,
34     JogBackward  := G_bJogR2CMD ,
35     Velocity     := G_leJogVelocity ,
36     Acceleration := leJogAcceleration ,
37     Deceleration := leJogDeceleration ,
38     Jerk         := leJogJerk  ,
39     BUSY         => G_bJog2Busy
40 );
```

軸1がサーボON状態になったら，JOG加速度，JOG減速度，JOGジャークの値をラベルに格納します。

シーケンサCPU側から操作されます。

シーケンサCPUに伝達されます。別のプログラムのインタロックに使用します。

構成は軸1のプログラムと共通です。

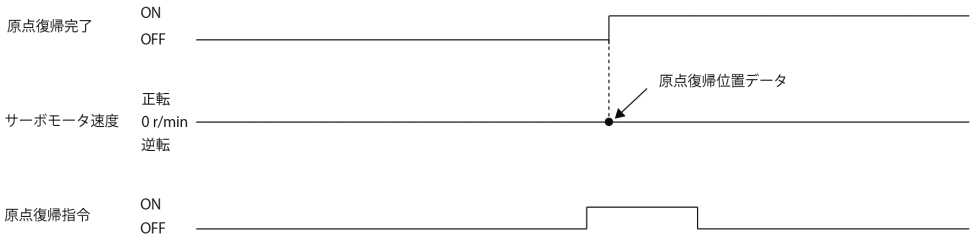
## 5.7 原点復帰（プログラム名：Homing）

原点復帰方式は，サーボアンプのパラメータで設定します。
本節では，データセット式原点復帰を例に説明します。
近点ドグ式原点復帰については付録を参照してください。

### (1) 概要

データセット式原点復帰は，原点復帰を実行した場所を原点とする方式です。

［データセット式原点復帰(Method37)のタイムチャート］




### (2) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MC_Home | 指定した軸の原点復帰を行ないます。 |

### (3) ローカルラベル

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MC_Home_1 | MC_Home | VAR | | | 原点復帰FB1 Axis0001 |
| 2 | MC_Home_2 | MC_Home | VAR | | | 原点復帰FB2 Axis0002 |
| 3 | MC_Home_3 | MC_Home | VAR | | | 原点復帰FB3 VirtualAxis0001 |
| 4 | bHoming1Done | ビット | VAR | | | 原点復帰FB1 Done出力 |
| 5 | bHoming1Error | ビット | VAR | | | 原点復帰FB1 Error出力 |
| 6 | bHoming2Done | ビット | VAR | | | 原点復帰FB2 Done出力 |
| 7 | bHoming2Error | ビット | VAR | | | 原点復帰FB2 Error出力 |
| 8 | bHoming3Done | ビット | VAR | | | 原点復帰FB3 Done出力 |
| 9 | bHoming3Error | ビット | VAR | | | 原点復帰FB3 Error出力 |
| 10 | | | | | | |

①
②

①プログラムエディタにFB(MC_Home)をドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

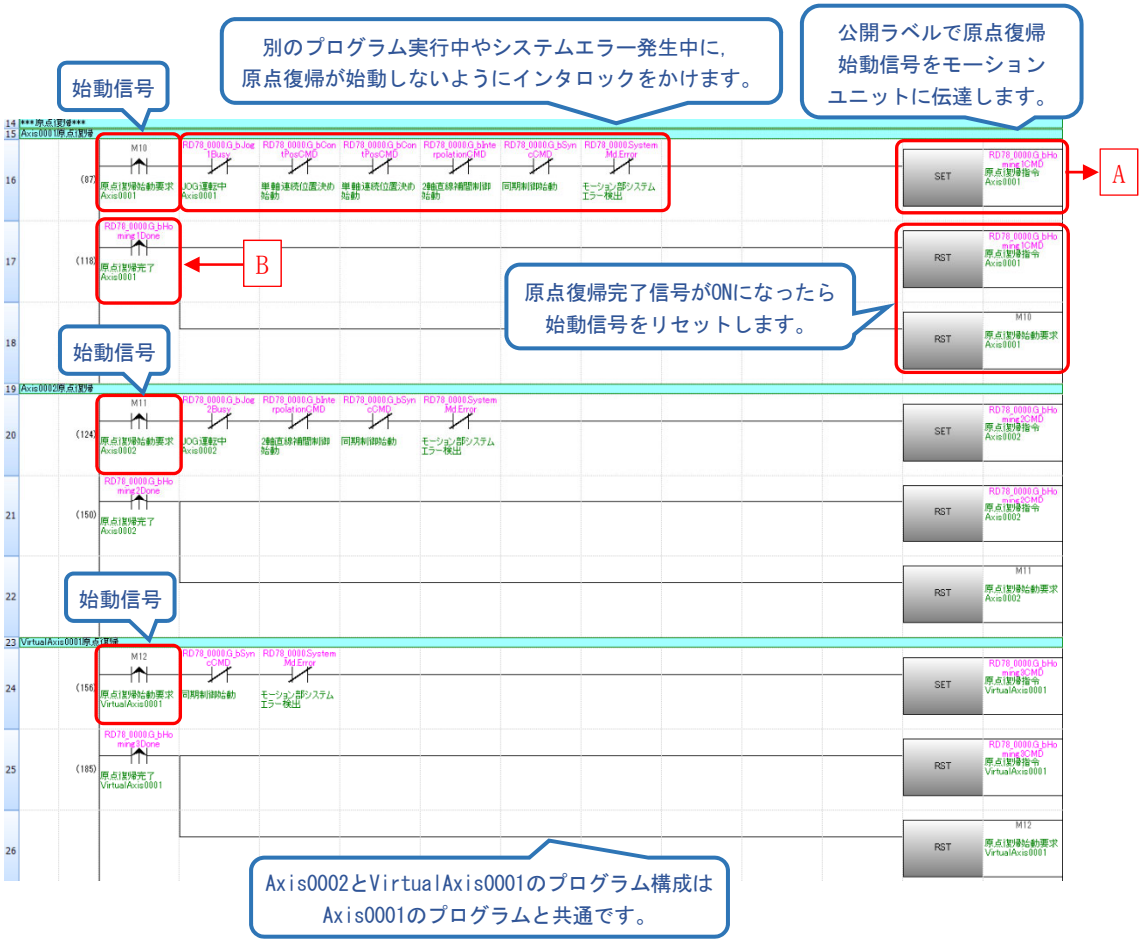
### (4) プログラム例

原点復帰信号に割り付けたデバイスの立ち上がりを，公開ラベルを介してモーションユニットに伝達します。モーションユニットの原点復帰完了信号をシーケンサCPUに返し，始動信号をOFFします。
別のプログラム実行中やシステムエラー発生中に，原点復帰が始動しないようにインタロックをかけます。

［始動信号］

| | Axis0001 | Axis0002 | VirtualAxis0001 |
|---|---|---|---|
| 原点復帰指令 | M10 | M11 | M12 |

［シーケンサCPU］


始動信号
別のプログラム実行中やシステムエラー発生中に，
原点復帰が始動しないようにインタロックをかけます。
公開ラベルで原点復帰
始動信号をモーション
ユニットに伝達します。
A
B
原点復帰完了信号がONになったら
始動信号をリセットします。
始動信号
始動信号
Axis0002とVirtualAxis0001のプログラム構成は
Axis0001のプログラムと共通です。

［モーションユニット］

```
1  //-----原点復帰 Axis0001-----
2  //原点復帰
3  MC_Home_1(
4      Axis     := Axis0001.AxisRef ,
5      Execute  := G_bHoming1CMD ,
6      Position := 0.0 ,
7      Done     => bHoming1Done,
8      Error    => bHoming1Error
9  );
10
11 //シーケンサに送る原点復帰完了信号をONする
12 G_bHoming1Done := bHoming1Done OR bHoming1Error;
13
14 //-----原点復帰 Axis0002-----
15 //原点復帰
16 MC_Home_2(
17     Axis     := Axis0002.AxisRef ,
18     Execute  := G_bHoming2CMD ,
19     Position := 0.0 ,
20     Done     => bHoming2Done,
21     Error    => bHoming2Error
22 );
23
24 //シーケンサに送る原点復帰完了信号をONする
25 G_bHoming2Done := bHoming2Done OR bHoming2Error;
26
27 //-----原点復帰 VirtualAxis0001-----
28 //原点復帰
29 MC_Home_3(
30     Axis     := VirtualAxis0001.AxisRef ,
31     Execute  := G_bHoming3CMD ,
32     Position := 0.0 ,
33     Done     => bHoming3Done,
34     Error    => bHoming3Error
35 );
36
37 //シーケンサに送る原点復帰完了信号をONする
38 G_bHoming3Done := bHoming3Done OR bHoming3Error;
39
```

A

B

Axis0002とVirtualAxis0001の
プログラム構成は
Axis0001のプログラムと共通です。

## 5.8　単軸位置決め制御（プログラム名：Positioning）

アドレス情報を使用して，指定の位置へ位置決めを行ないます。

### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MC_MoveAbsolute | 絶対位置の目標位置を指定し，位置決めを実行します。 |
| | MC_MoveRelative | 相対位置の移動量を指定し，位置決めを実行します。 |

### (2) ローカルラベル

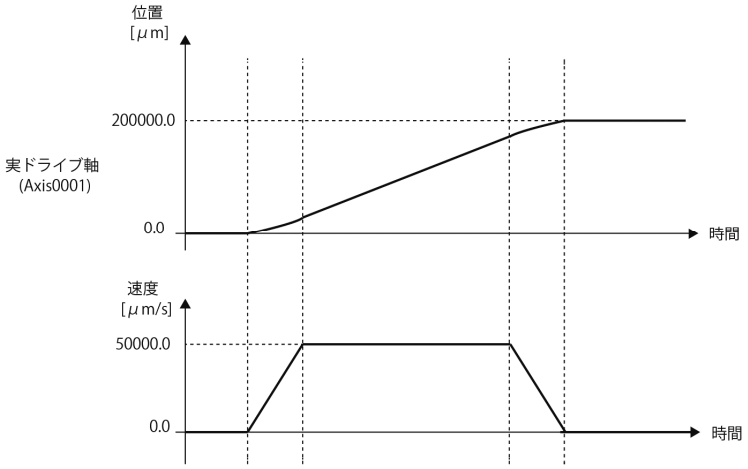
①
②

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MC_MoveRelative_1 | MC_MoveRelative | VAR | | | 相対値位置決めFB |
| 2 | leDistance | 倍精度実数 | VAR | | | 移動量 |
| 3 | leVelocity | 倍精度実数 | VAR | | | 速度 |
| 4 | leAcceleration | 倍精度実数 | VAR | | | 加速度 |
| 5 | leDeceleration | 倍精度実数 | VAR | | | 減速度 |
| 6 | leJerk | 倍精度実数 | VAR | | | ジャーク |
| 7 | bDone | ビット | VAR | | | 相対値位置決めFB Done出力 |
| 8 | bBusy | ビット | VAR | | | 相対値位置決めFB Busy出力 |
| 9 | bError | ビット | VAR | | | 相対値位置決めFB Error出力 |
| 10 | bExecute_P | ビット | VAR | | | 実行指令 |
| 11 | bCommandAborted | ビット | VAR | | | 相対値位置決めFB 実行中断出力 |
| 12 | | | | | | |

**①プログラムエディタにFB(MC_MoveRelative)をドラッグ＆ドロップすると自動で追加されます。**
**②手動で登録します。**

### (3) プログラム例

下図の動作パターンの相対位置指定の位置決めを実行するプログラム例を示します。

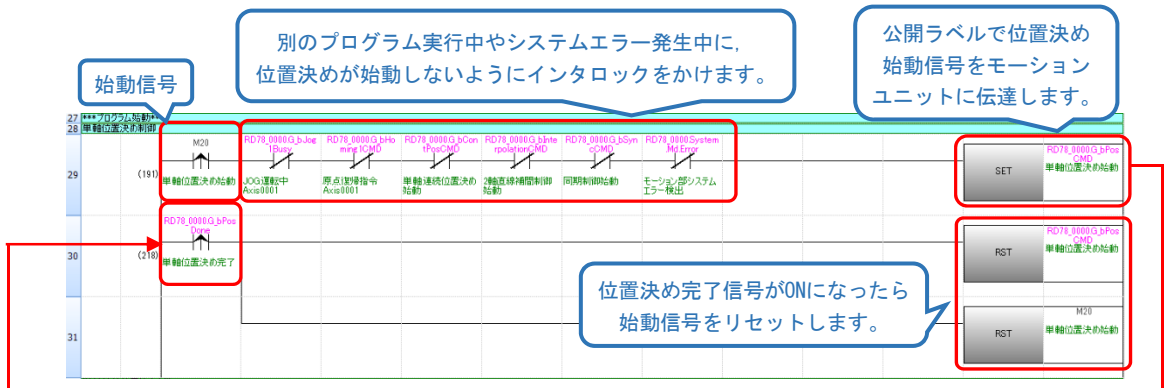



単軸位置決め始動信号に割り付けたデバイスの立ち上がりを，公開ラベルを介してモーションユニットに伝達します。モーションユニットは始動信号を受けて，位置決めに必要なデータをローカルラベルに格納し，位置決め用制御FBを実行します。位置決め完了信号をシーケンサCPUに返し，始動信号をOFFします。
別のプログラム実行中やシステムエラー発生中に，単軸位置決めが始動しないようにインタロックをかけます。

**[始動信号]**

| | |
|---|---|
| 単軸位置決め指令 | M20 |

［シーケンサCPU］




［モーションユニット］

```
1  //-----単軸位置決め制御用データ設定・実行指令&リセット  -----
2  IF G_bPosCMD THEN;
3      leDistance       := 200000.0;
4      leVelocity       := 50000.0;
5      leAcceleration   := 100000.0;
6      leDeceleration   := 100000.0;
7      leJerk           := 0.0;
8      bExecute_P := TRUE;
9  ELSE
10     bExecute_P := FALSE;
11 END_IF;
12
13 //-----単軸位置決め-----
14 //相対値位置決め
15 MC_MoveRelative_1(
16     Axis             := Axis0001.AxisRef ,
17     Execute          := bExecute_P ,
18     Distance         := leDistance ,
19     Velocity         := leVelocity ,
20     Acceleration     := leAcceleration ,
21     Deceleration     := leDeceleration ,
22     Jerk             := leJerk ,
23     Done             => bDone ,
24     Busy             => bBusy ,
25     CommandAborted   => bCommandAborted ,
26     Error            => bError
27 );
28
29 //シーケンサに送る <= 位置決め完了信号・FB実行中断・FBエラーのいずれかの出力がONする ＊制御終了信号
30 G_bPosDone := ( bDone & Axis0001.Md.CmdInPos ) OR bCommandAborted OR bError;
31
```

位置決め始動信号がONになったら
位置決め用データをラベルに格納します。

データの格納が完了したら
位置決め用FBを実行します。

FBのDone出力と指令インポジションをAND条件
で取得し，動作正常完了を判断しています。

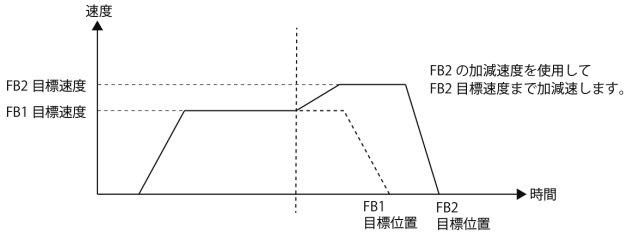
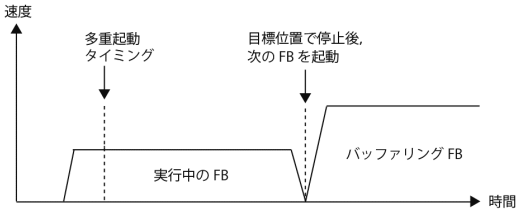
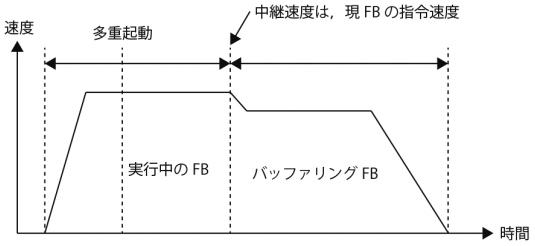
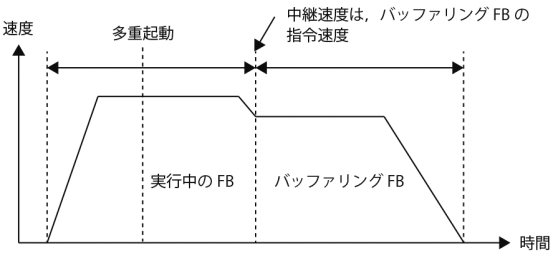
位置決め完了信号をシーケンサCPUに伝達します。

## 5.9 単軸連続位置決め（プログラム名：ContinuousPositioning）

### (1) 概要

モーション制御FBを実行中の軸に，別インスタンスの動作系FBを実行することで，複数のモーション制御FBを停止することなく連続的に実行できます。
モーション制御FBのBufferMode入力で，バッファモードを指定します。
1つの軸でバッファリングできる動作系FBは最大2つです。

### (2) バッファモードの動作パターン

| バッファモード | 動　作 |
|---|---|
| 0：mcAborting | 実行中のFBを中断(キャンセル)して次のFBを即時実行します。<br>  |
| 1：mcBuffered | 実行中のFBが減速停止後，次のFBを実行します。<br>  |
| 2：mcBlendingLow | 実行中のFBとバッファリングFBの目標速度のうち，低い方の速度を切換え速度とします。 |
| 3：mcBlendingPrevious | 実行中のFBが目標位置へ到達後，次のFBを実行します。<br>目標速度は次のFB指令速度です。<br>  |
| 4：mcBlendingNext | 実行中のFBが目標位置へ到達後，次のFBを実行します。<br>目標位置到着時，次FBの指令速度になります。<br>  |
| 5：mcBlendingHigh | 実行中のFBとバッファリングFBの目標速度のうち，高い方の速度を切換え速度とします。 |

### (3) ローカルラベル

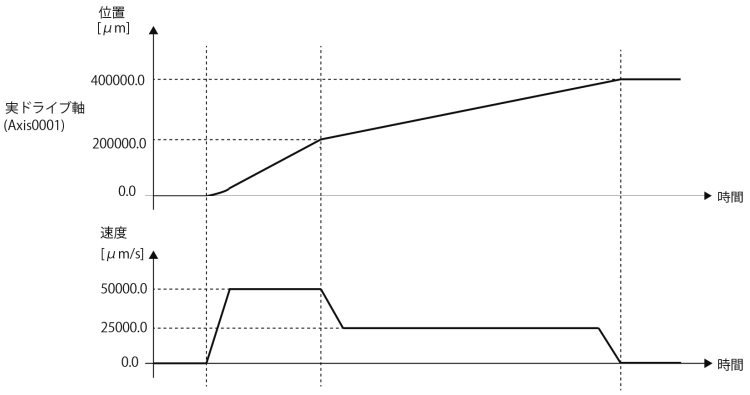
①
②

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MC_MoveRelative_1 | MC_MoveRelative | VAR | | | 相対値位置決めFB1 |
| 2 | MC_MoveRelative_2 | MC_MoveRelative | VAR | | | 相対値位置決めFB2 |
| 3 | leDistance1 | 倍精度実数 | VAR | | | 移動量1 |
| 4 | leDistance2 | 倍精度実数 | VAR | | | 移動量2 |
| 5 | leVelocity1 | 倍精度実数 | VAR | | | 速度1 |
| 6 | leVelocity2 | 倍精度実数 | VAR | | | 速度2 |
| 7 | leAcceleration1 | 倍精度実数 | VAR | | | 加速度1 |
| 8 | leDeceleration1 | 倍精度実数 | VAR | | | 減速度1 |
| 9 | leAcceleration2 | 倍精度実数 | VAR | | | 加速度2 |
| 10 | leDeceleration2 | 倍精度実数 | VAR | | | 減速度2 |
| 11 | leJerk | 倍精度実数 | VAR | | | ジャーク |
| 12 | bBusy1 | ビット | VAR | | | 相対値位置決めFB1 Busy出力 |
| 13 | bActive1 | ビット | VAR | | | 相対値位置決めFB1 Active出力 |
| 14 | bDone2 | ビット | VAR | | | 相対値位置決めFB2 Done出力 |
| 15 | bBusy2 | ビット | VAR | | | 相対値位置決めFB2 Busy出力 |
| 16 | TON_1 | TON | VAR | | | オンディレイタイマFB |
| 17 | bDwell_in | ビット | VAR | | | タイマ入力 |
| 18 | bDwell_out | ビット | VAR | | | タイマ出力 |
| 19 | bExecute_CP | ビット | VAR | | | 実行指令 |
| 20 | bCommandAborted | ビット | VAR | | | 相対値位置決めFB 実行中断出力 |
| 21 | bError | ビット | VAR | | | 相対値位置決めFB Error出力 |
| 22 | | | | | | |

**①プログラムエディタにFB(MC_MoveRelative, TON)をドラッグ＆ドロップすると自動で追加されます。**
**②手動で登録します。**

### (4) プログラム例

下図のような動作パターンの相対位置指定の位置決めを実行するプログラム例を示します。

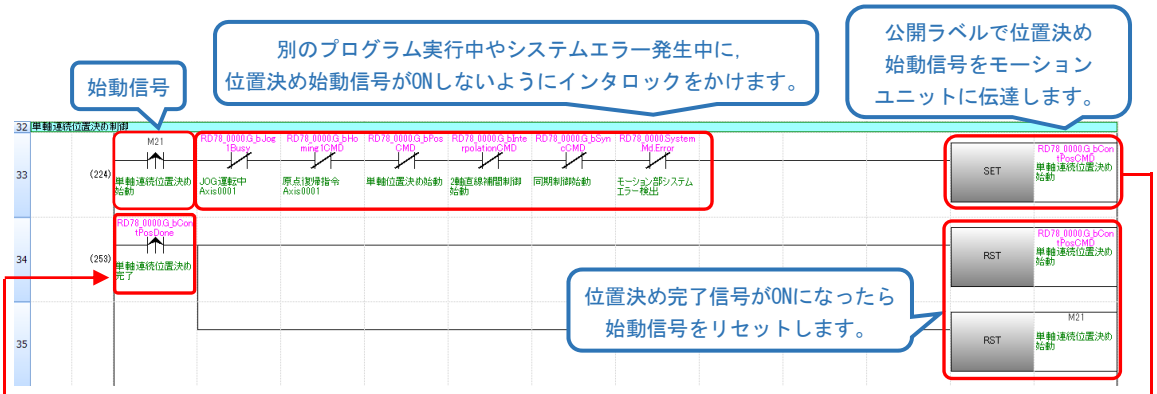



単軸連続位置決め始動信号に割り付けたデバイスの立ち上がりを，公開ラベルを介してモーションユニットに伝達します。モーションユニットは始動信号を受けて，位置決めに必要なデータをローカルラベルに格納し，位置決め用制御FBを実行します。
1つ目のMC_MoveRelativeのActive出力を2つ目のMC_MoveRelativeのExecute入力に接続することで，最初のFB実行中に2つ目のFBを実行し，バッファリングします。
1つ目のFBの位置決めが完了したら，バッファされている2つ目のFBが実行されます。ドウェルにはオンディレイタイマ(100[ms])を使用します。ドウェル時間経過後，またはエラー発生，または実行中断時，位置決め完了信号をシーケンサCPUに返し，始動信号をOFFします。
別のプログラム実行中やシステムエラー発生中に，単軸連続位置決めが始動しないようにインタロックをかけます。

**[始動信号]**

| | |
|---|---|
| 単軸連続位置決め指令 | M21 |

［シーケンサCPU］




［モーションユニット］

```
1  //-----単軸連続位置決め用データ設定・実行指令&リセット-----
2  IF G_bContPosCMD THEN;
3      leDistance1      :=  200000.0;
4      leDistance2      :=  200000.0;
5      leVelocity1      :=  50000.0;
6      leVelocity2      :=  25000.0;
7      leAcceleration1  :=  100000.0;
8      leDeceleration1  :=  100000.0;
9      leAcceleration2  :=  50000.0;
10     leDeceleration2  :=  50000.0;
11     leJerk           :=  0.0;
12     bExecute_CP := TRUE;
13 ELSE
14     bExecute_CP := FALSE;
15 END_IF;
16
17 //-----単軸連続位置決め+バッファモード-----
18 ///相対値位置決め1
19 MC_MoveRelative_1(
20     Axis          := Axis0001.AxisRef ,
21     Execute       := bExecute_CP ,
22     Distance      := leDistance1 ,
23     Velocity      := leVelocity1 ,
24     Acceleration  := leAcceleration1 ,
25     Deceleration  := leDeceleration1 ,
26     Jerk          := leJerk ,
27     Busy          => bBusy1 ,
28     Active        => bActive1
29 );
30
31 //相対値位置決め2
32 MC_MoveRelative_2(
33     Axis          := Axis0001.AxisRef ,
34     Execute       := bActive1,
35     Distance      := leDistance2 ,
36     Velocity      := leVelocity2 ,
37     Acceleration  := leAcceleration2 ,
38     Deceleration  := leDeceleration2 ,
39     BufferMode    := MC_BUFFER_MODE__mcBlendingPrevious ,
40     Done          => bDone2 ,
41     Busy          => bBusy2
42 );
43
44 //FB実行中断出力
45 bCommandAborted := MC_MoveRelative_1.CommandAborted OR MC_MoveRelative_2.CommandAborted;
46
47 //FBエラー出力
48 bError := MC_MoveRelative_1.Error OR MC_MoveRelative_2.Error;
49
50 //相対値位置決め2完了信号でタイマ入力をセット
51 SET( bDone2, bDwell_in );
52
53 //ドウェル
54 TON_1( IN := bDwell_in, PT := T#100ms, Q=> bDwell_out );
55
56 //シーケンサに送る <= ドウェル時間経過・FB実行中断・FBエラーのいずれかの出力がONする ＊制御終了信号
57 G_bContPosDone := bDwell_out OR bCommandAborted OR bError ;
58
59 //位置決め完了時、相対位置決め2完了信号保持(タイマ入力)をリセット
60 RST( bDwell_out, bDwell_in );
61
```

位置決め始動信号がONになったら
位置決め用データをラベルに格納します。

データの格納が完了したら
位置決め用FBを実行します。

最初のFBのActive出力を2つ目のFBのExecute
入力接続し，バッファモードとします。

ポイント参照

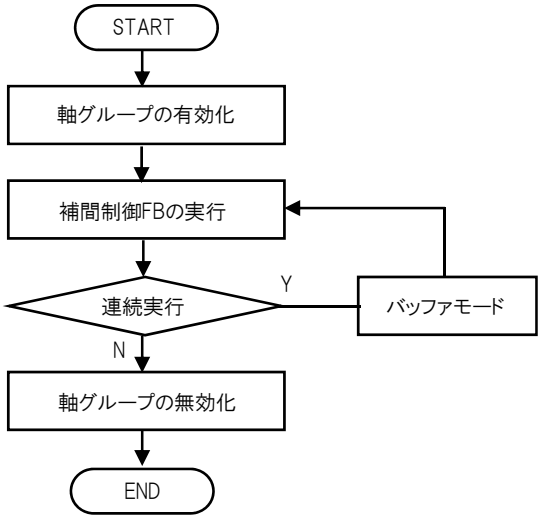
ドウェルはオンディレイタ
イマ“TON”を使用します。

**[ポイント]**
FBの出力信号は，プログラム内に直接“(FB名).(出力信号名)”を直接入力すると，ラベルを接続しなくても使用することができます。

## 5.10 補間制御（プログラム名：LinearInterpolation）

### 5.10.1 補間制御の手順

2軸以上で補間制御を実行する手順を下図に示します。




### 5.10.2 軸グループ有効化／無効化

軸グループの設定は，3.12節を参照してください。
補間制御を実行するには，軸グループ状態を「4：GroupStandby」に設定する必要があります。

(1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(管理系) | MC_GroupEnable | 指定した軸グループの状態を「0：GroupDisabled」から「4：GroupStandby」に遷移します。 |
| | MC_GroupDisable | 指定した軸グループの状態を「0：GroupDisabled」に遷移します。 |

### 5.10.3 補間制御

直線補間用と円弧補間用を準備しています。軸グループ有効化後に，以下のFBを実行します。

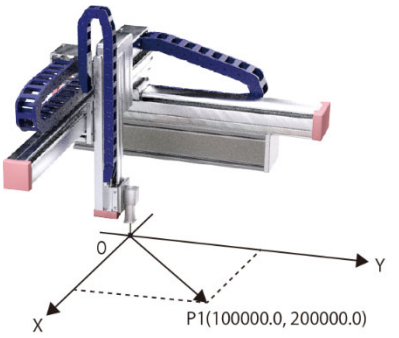
(1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MCv_MoveLinearInterpolateAbsolute | 絶対値直線補間制御 |
| | MCv_MoveLinearInterpolateRelative | 相対値直線補間制御 |
| | MCv_MoveCircularInterpolateAbsolute | 絶対値円弧補間制御 |
| | MCv_MoveCircularInterpolateRelative | 相対値円弧補間制御 |

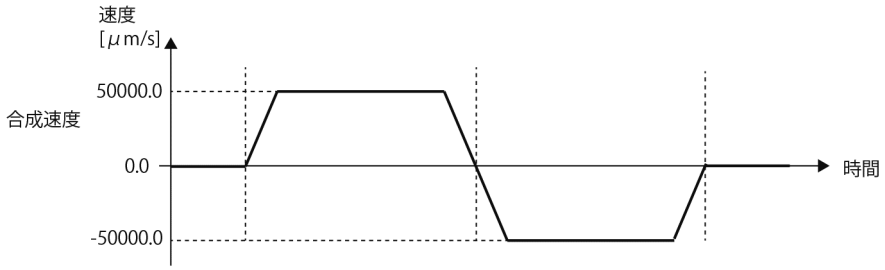
## 5.10.4　直線補間のプログラム例

### (1) 動作パターン

原点(0.0, 0.0)[um]と点P1(100000.0, 200000.0)[um]の間を直線補間で往復します。




[単位：μm]




### (2) 軸番号と移動量の設定方法

(a) MCv_MoveLinearInterpolateRelativeのLinearAxes入力

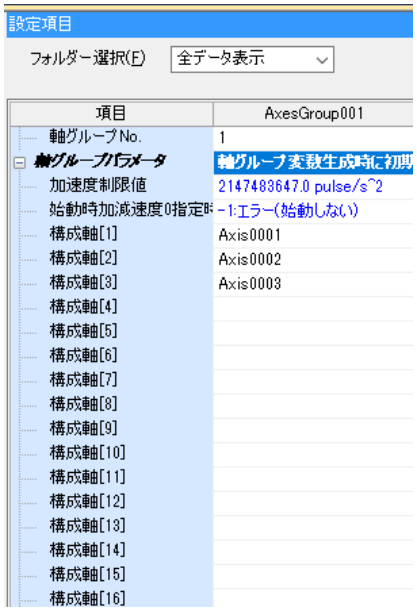
INT(符号付きワード)型の要素数16の配列を指定します。
サンプルプログラムでは，wAxes[0..15]というラベルを使用しています。
AxesGroup(3.12節参照)に登録した構成軸[1]～[16]のうち，直線補間に使用する軸の構成軸番号をwAxes[0]から順に格納します。

(b) Distance入力

LREAL(倍精度実数)型の要素数16の配列を指定します。
サンプルプログラムでは，lePosition[0..15]というラベルを使用しています。
AxesGroupに登録した構成軸[1]～[16]の移動量をlePosition[0]～lePosition[15]に格納します。

**[ポイント]**
使用する軸数にかかわらず，LinearAxes入力に設定するINT型の配列の要素数，およびDistance入力に設定するLREAL型の配列の要素数は，必ず16にする必要があります。

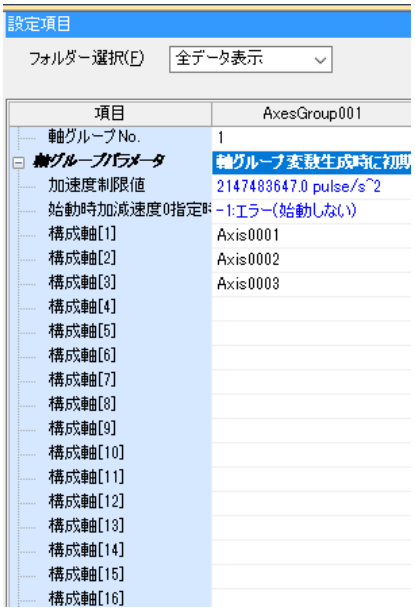
[設定例1]
・軸グループ …… 構成軸[1]：Axis0001，構成軸[2]：Axis0002，構成軸[3]：Axis0003
・直線補間 …… Axis0001，Axis0002の2軸を使用




```
wAxes[0] := 1;(←構成軸[1])
wAxes[1] := 2;(←構成軸[2])

lePosition[0] := (構成軸1(=Axis0001)の移動量);
lePosition[1] := (構成軸2(=Axis0002)の移動量);
```

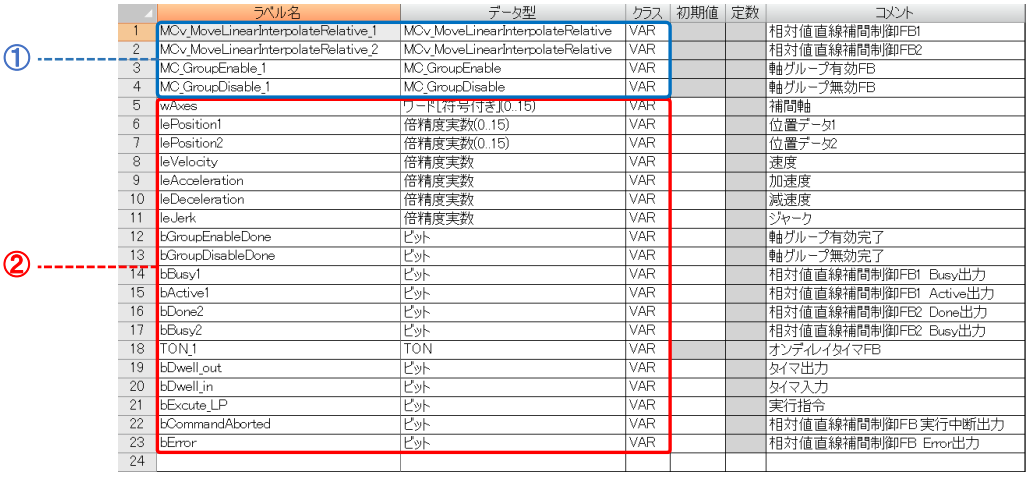
[設定例2]
・軸グループ …… 構成軸[1]：Axis0001，構成軸[2]：Axis0002，構成軸[3]：Axis0003
・直線補間 …… Axis0002，Axis0003の2軸を使用




```
wAxes[0] := 2;(←構成軸[2])
wAxes[1] := 3;(←構成軸[3])

lePosition[0] := 0.0;
lePosition[1] := (構成軸2(=Axis0002)の移動量);
lePosition[2] := (構成軸3(=Axis0003)の移動量);
```

### (3) ローカルラベル



| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MCv_MoveLinearInterpolateRelative_1 | MCv_MoveLinearInterpolateRelative | VAR | | | 相対値直線補間制御FB1 |
| 2 | MCv_MoveLinearInterpolateRelative_2 | MCv_MoveLinearInterpolateRelative | VAR | | | 相対値直線補間制御FB2 |
| 3 | MC_GroupEnable_1 | MC_GroupEnable | VAR | | | 軸グループ有効FB |
| 4 | MC_GroupDisable_1 | MC_GroupDisable | VAR | | | 軸グループ無効FB |
| 5 | wAxes | ワード[符号付き](0..15) | VAR | | | 補間軸 |
| 6 | lePosition1 | 倍精度実数(0..15) | VAR | | | 位置データ1 |
| 7 | lePosition2 | 倍精度実数(0..15) | VAR | | | 位置データ2 |
| 8 | leVelocity | 倍精度実数 | VAR | | | 速度 |
| 9 | leAcceleration | 倍精度実数 | VAR | | | 加速度 |
| 10 | leDeceleration | 倍精度実数 | VAR | | | 減速度 |
| 11 | leJerk | 倍精度実数 | VAR | | | ジャーク |
| 12 | bGroupEnableDone | ビット | VAR | | | 軸グループ有効完了 |
| 13 | bGroupDisableDone | ビット | VAR | | | 軸グループ無効完了 |
| 14 | bBusy1 | ビット | VAR | | | 相対値直線補間制御FB1 Busy出力 |
| 15 | bActive1 | ビット | VAR | | | 相対値直線補間制御FB1 Active出力 |
| 16 | bDone2 | ビット | VAR | | | 相対値直線補間制御FB2 Done出力 |
| 17 | bBusy2 | ビット | VAR | | | 相対値直線補間制御FB2 Busy出力 |
| 18 | TON_1 | TON | VAR | | | オンディレイタイマFB |
| 19 | bDwell_out | ビット | VAR | | | タイマ出力 |
| 20 | bDwell_in | ビット | VAR | | | タイマ入力 |
| 21 | bExcute_LP | ビット | VAR | | | 実行指令 |
| 22 | bCommandAborted | ビット | VAR | | | 相対値直線補間制御FB 実行中断出力 |
| 23 | bError | ビット | VAR | | | 相対値直線補間制御FB Error出力 |
| 24 | | | | | | |



①プログラムエディタに各FBをドラッグ＆ドロップすると自動で追加されます。

②手動で登録します。

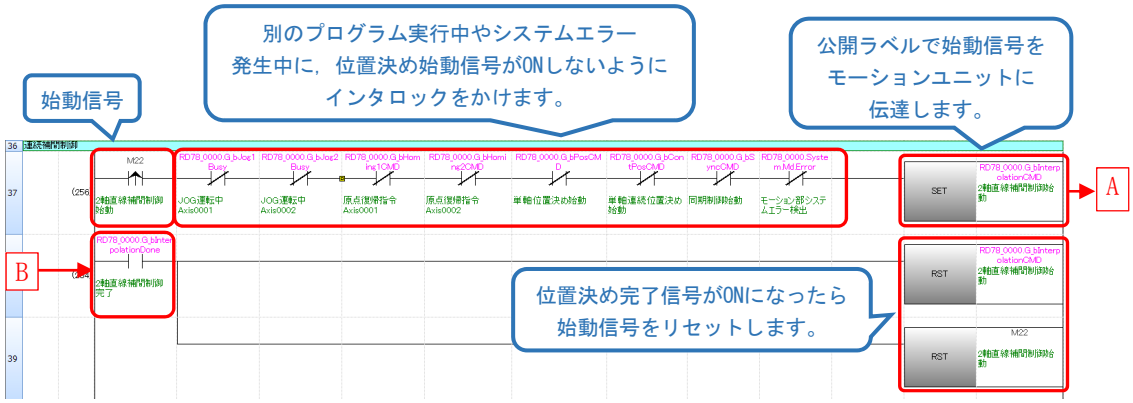
### (4) プログラム例

| | プログラム内容 | 説　明 |
|---|---|---|
| ① | 2 軸直線補間制御始動に割り付けたデバイスの立ち上がりを，公開ラベルを介してモーションユニットに伝達します。 | シーケンサ CPU |
| ② | ・始動信号を受けて，位置決めに必要なデータをローカルラベルに格納し，軸グループを有効化します。<br>・軸グループを有効化した後，相対値直線補間制御FB (MCv_MoveLinearInterpolateRelative)2つをバッファモードで起動します。<br>・ドウェル時間が経過するか，エラー発生または実行中断で，軸グループを無効化します。<br>・軸グループ無効化が完了したら，始動信号とオンディレイタイマの入力をリセットします。 | モーション<br>ユニット |
| ③ | 別のプログラム実行中やシステムエラー発生中に，2 軸直線補間制御が始動しないようにインタロックをかけます。 | シーケンサ CPU |

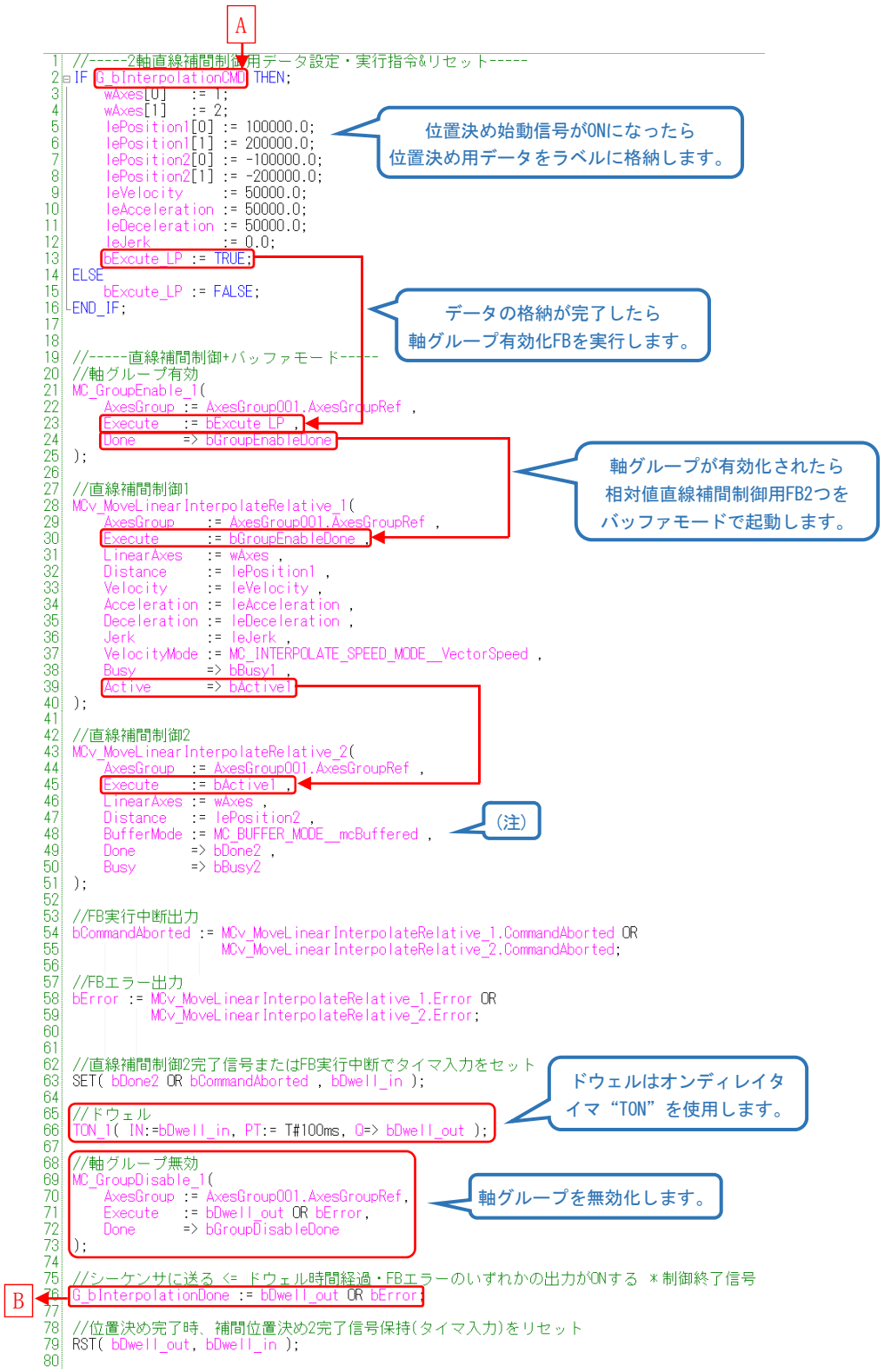
［始動信号］

| | |
|---|---|
| 2 軸直線補間制御始動 | M22 |

［シーケンサCPU］


始動信号
別のプログラム実行中やシステムエラー
発生中に，位置決め始動信号がONしないように
インタロックをかけます。
公開ラベルで始動信号を
モーションユニットに
伝達します。
A
B
位置決め完了信号がONになったら
始動信号をリセットします。
SET
RST
RST

［モーションユニット］



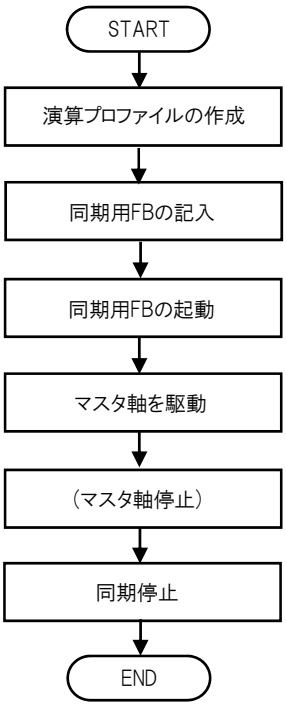

```
1  //-----2軸直線補間制御用データ設定・実行指令&リセット-----
2  IF G_bInterpolationCMD THEN;
3      wAxes[0]    := 1;
4      wAxes[1]    := 2;
5      lePosition1[0] := 100000.0;
6      lePosition1[1] := 200000.0;
7      lePosition2[0] := -100000.0;
8      lePosition2[1] := -200000.0;
9      leVelocity     := 50000.0;
10     leAcceleration := 50000.0;
11     leDeceleration := 50000.0;
12     leJerk         := 0.0;
13     bExcute_LP := TRUE;
14 ELSE
15     bExcute_LP := FALSE;
16 END_IF;
17
18
19 //-----直線補間制御+バッファモード-----
20 //軸グループ有効
21 MC_GroupEnable_1(
22     AxesGroup := AxesGroup001.AxesGroupRef ,
23     Execute   := bExcute_LP ,
24     Done      => bGroupEnableDone
25 );
26
27 //直線補間制御1
28 MCv_MoveLinearInterpolateRelative_1(
29     AxesGroup    := AxesGroup001.AxesGroupRef ,
30     Execute      := bGroupEnableDone ,
31     LinearAxes   := wAxes ,
32     Distance     := lePosition1 ,
33     Velocity     := leVelocity ,
34     Acceleration := leAcceleration ,
35     Deceleration := leDeceleration ,
36     Jerk         := leJerk ,
37     VelocityMode := MC_INTERPOLATE_SPEED_MODE__VectorSpeed ,
38     Busy         => bBusy1 ,
39     Active       => bActive1
40 );
41
42 //直線補間制御2
43 MCv_MoveLinearInterpolateRelative_2(
44     AxesGroup  := AxesGroup001.AxesGroupRef ,
45     Execute    := bActive1 ,
46     LinearAxes := wAxes ,
47     Distance   := lePosition2 ,
48     BufferMode := MC_BUFFER_MODE__mcBuffered ,
49     Done       => bDone2 ,
50     Busy       => bBusy2
51 );
52
53 //FB実行中断出力
54 bCommandAborted := MCv_MoveLinearInterpolateRelative_1.CommandAborted OR
55                    MCv_MoveLinearInterpolateRelative_2.CommandAborted;
56
57 //FBエラー出力
58 bError := MCv_MoveLinearInterpolateRelative_1.Error OR
59           MCv_MoveLinearInterpolateRelative_2.Error;
60
61
62 //直線補間制御2完了信号またはFB実行中断でタイマ入力をセット
63 SET( bDone2 OR bCommandAborted , bDwell_in );
64
65 //ドウェル
66 TON_1( IN:=bDwell_in, PT:= T#100ms, Q=> bDwell_out );
67
68 //軸グループ無効
69 MC_GroupDisable_1(
70     AxesGroup := AxesGroup001.AxesGroupRef,
71     Execute   := bDwell_out OR bError,
72     Done      => bGroupDisableDone
73 );
74
75 //シーケンサに送る <= ドウェル時間経過・FBエラーのいずれかの出力がONする ＊制御終了信号
76 G_bInterpolationDone := bDwell_out OR bError;
77
78 //位置決め完了時、補間位置決め2完了信号保持(タイマ入力)をリセット
79 RST( bDwell_out, bDwell_in );
80
```

（注）バッファリング FB で指定速度，加速度指定，減速度指定を省略した場合，「バッファリング FB の直前の FB」の指定速度などが引き継がれます。

## 5.11　同期制御（プログラム名：Synchronous）

### 5.11.1　同期制御の手順

同期制御を実行する手順を下図に示します。

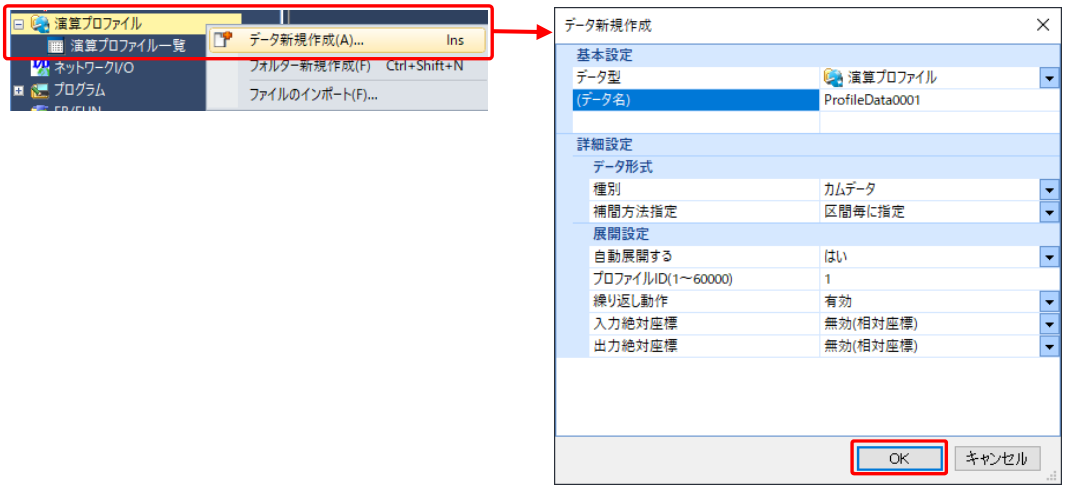


## 5.11.2　演算プロファイル

制御で使用する波形データを総称して，演算プロファイルと呼びます。
本項では，カムデータの作成方法について説明します。

### (1) 演算プロファイルデータの新規作成

モーション制御設定機能で，ナビゲーションウィンドウの“演算プロファイル”を右クリック⇒“データ新規作成”を選択して，“データ新規作成”画面を表示します。



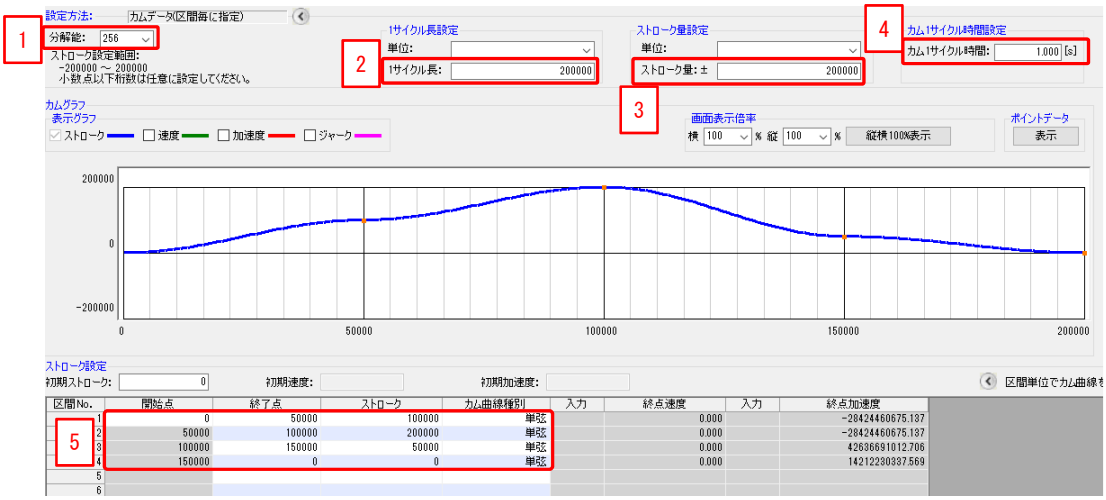

入力が必要な項目を以下に示します。

| 設定項目 | 説　明 |
|---|---|
| 自動展開する | はい：演算プロファイルを電源ON時に自動的に展開を行ないます。<br>いいえ：使用時に演算プロファイル展開 FB を実行する必要があります。 |
| 繰り返し動作 | 無効：演算プロファイルの終点まで実行すると，制御を終了します。<br>有効：設定した場合連続的に演算プロファイルの実行を繰り返します。 |
| 出力絶対座標 | 無効(相対座標)：<br>演算プロファイル(カム)実行開始時，現在の値を基準として出力値を計算します。送りカム動作を実施したい場合に選択します。<br>有効(絶対座標)：<br>演算プロファイル(カム)実行開始時の出力値が，演算プロファイルの1サイクル開始時に常に始点となるよう計算します。<br>繰り返し動作で演算プロファイルの始点と終点で異なる場合は，次の 1 サイクル開始時点で最初の出力値に戻るよう指令を 1 演算周期で出力します。 |

本書では，初期値のまま変更しません。[OK]ボタンをクリックします。

### (2) カムデータの作成

演算プロファイルの波形を設定します。




入力が必要な項目を以下に示します。

| No. | 項　目 | 内　容 |
|---|---|---|
| 1 | 分解能 | カムデータの分解能を選択します。 |
| 2 | 1サイクル長設定 | 1サイクル長の単位と1サイクル長を設定します。<br>(マスタ軸がどれだけ移動したらカムが一周するか) |
| 3 | ストローク量設定 | ストローク量の単位とストローク量を設定します。<br>(カムが一周する間に，スレーブ軸が最大どれだけ移動するか) |
| 4 | カム1サイクル時間設定 | カムが1サイクルする際にかかる時間を設定します。<br>速度，加速度，ジャークの数値の計算に使用されます。 |
| 5 | ストローク設定 | ストローク位置を設定します。 |

本項では，以下のように設定してください。

| No. | 項　目 | 設定値 |
|---|---|---|
| 1 | 分解能 | 256 |
| 2 | 1サイクル長設定 | 200000（単位はブランク） |
| 3 | ストローク量設定 | 200000（単位はブランク） |
| 4 | カム1サイクル時間設定 | 1.000 |
| 5 | ストローク設定 | 下表参照 |

| 区間No. | 開始点 | 終了点 | ストローク | カム曲線種別 |
|---|---|---|---|---|
| 1 | 0 | 50000 | 100000 | 単弦 |
| 2 | 50000 | 100000 | 200000 | 単弦 |
| 3 | 100000 | 150000 | 50000 | 単弦 |
| 4 | 150000 | 0 | 0 | 単弦 |

[ポイント]
直線カムの動作(マスタ軸と同じ動作，またはマスタ軸に対して一定の速度比を掛けた動作)をさせる場合，直線カムの演算プロファイルを作成するか，MC_GearInを使用してください。
直線カムの演算プロファイルデータはシステムでは準備していません。

### 5.11.3 単軸同期用FB

単軸同期用FBは，Masterに同期したSlaveの位置情報(指令)を伝達することで，ギア，変速機，カムなど機械的な仕組みをソフトウェアで制御できます。

(1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(動作系) | MC_CamIn | カム動作を実行します。 |
| | MC_GearIn | 主軸と従軸との間の速度比を設定してギア動作に入ります。 |
| | MC_CombineAxes | 選択可能な合成方法により，2つの軸の動作を合成して，第3の軸に出力します。 |
| | MCv_ChangeCycle | MC_CamIn制御中にカム1サイクル現在値を指定した値に変更します。<br>カム1サイクル現在値を任意の値に補正する目的で使用します。 |
| | MCv_*****Filter | Masterの入力に対して特定のフィルタ処理を行ない，その結果をSlaveに出力します。 |
| | MC_Stop | 同期を終了するときに使用します。 |

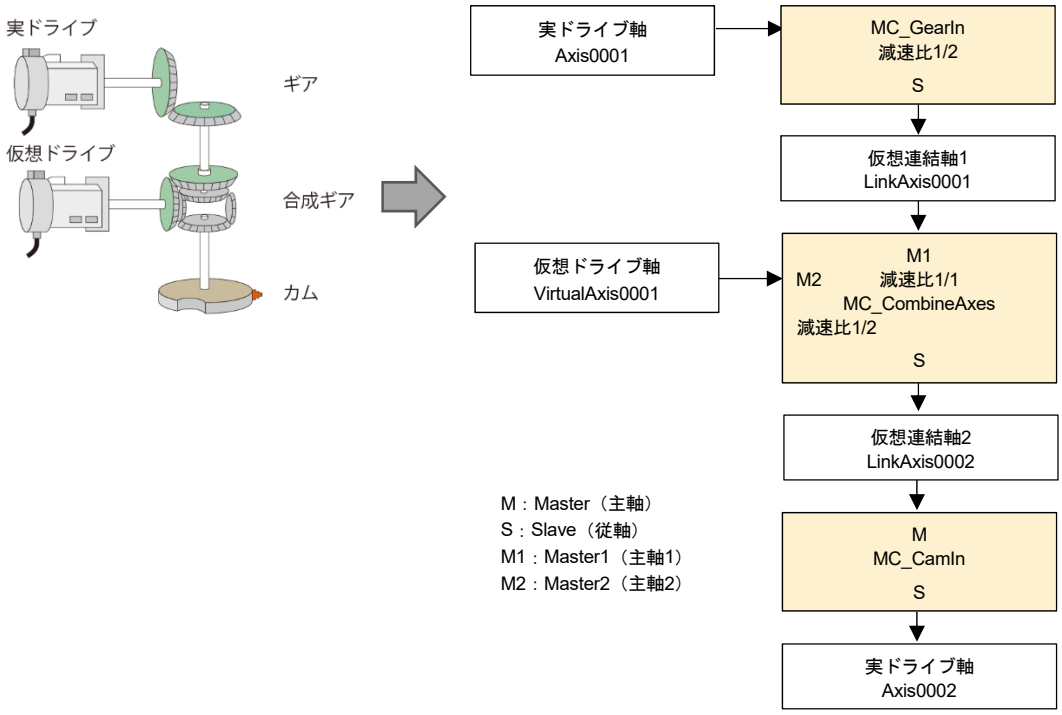
### 5.11.4 軸構成

単軸同期用FBにおいて，Master，およびSlaveに指定可能な軸を以下に示します。

| 軸種別 | 内　容 |
|---|---|
| 実ドライブ軸 | 複数の動作系 FB の制御を必要とする場合は，仮想連結軸ごとに FB を実行し，その結果(指令)を実ドライブ軸に伝達する構成にしてください。 |
| 仮想ドライブ軸 | 主に位置決め制御などにより指令を生成できます。 |
| 仮想連結軸 | 実ドライブ軸に指令を伝達するための中間軸として使用します。<br>ギアなど複数のFB(動作系)を使用する場合に，仮想連結軸の割り付けを実施してください。 |
| 実エンコーダ軸 | Masterとして使用します。<br>Slaveとした場合は，エラー「必須スレーブオブジェクト未設定」(エラーコード:1AA8H)となり始動しません。 |
| 仮想エンコーダ軸 | |

軸を単軸同期用FBに連結させることで，軸間で指令を伝達できます。また，複数のFBを実行してギアやカムなどの制御を同時に実行する場合は，軸を必要数分連結させて単軸同期用FBを実行してください。

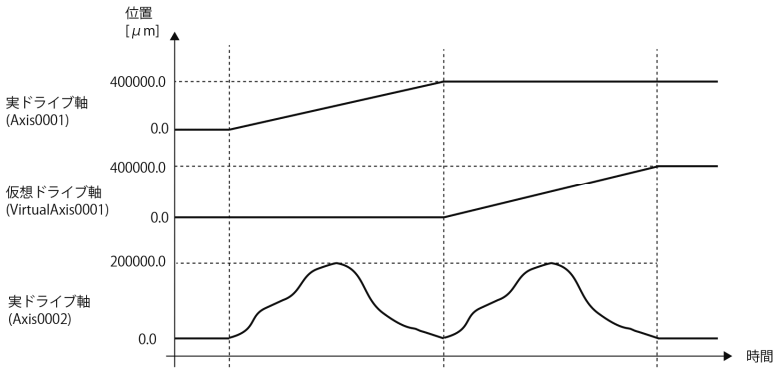
本章では，以下のカムのシステムを例に説明します。




## 5.11.5　プログラム例

### (1) 動作パターン

原点からX軸(Axis0001)が400000.0[um]移動すると同時に，Y軸(Axis0002)が5.11.2項で作成したカムパターンに従って動作します。X軸が400000.0[um]に到達後，仮想ドライブ軸が起動してY軸がカム動作します。このとき，X軸は停止しているため，Y軸のみ動作します。



(2) ローカルラベル

| | ラベル名 | データ型 | クラス | 初期値 | 定数 | コメント |
|---|---|---|---|---|---|---|
| 1 | MC_GearIn_1 | MC_GearIn | VAR | | | ギア動作FB |
| 2 | MC_CombineAxes_1 | MC_CombineAxes | VAR | | | 加減算位置決めFB |
| 3 | MC_CamIn_1 | MC_CamIn | VAR | | | カム動作FB |
| 4 | bGearInBusy | ビット | VAR | | | ギア動作FB Busy出力 |
| 5 | bCombineAxesBusy | ビット | VAR | | | 加減算位置決め動作FB Busy出力 |
| 6 | bCamInBusy | ビット | VAR | | | カム動作FB Busy出力 |
| 7 | MC_MoveRelative_1 | MC_MoveRelative | VAR | | | 相対値位置決めFB |
| 8 | MC_MoveRelative_2 | MC_MoveRelative | VAR | | | 相対値位置決めFB |
| 9 | MC_Stop_1 | MC_Stop | VAR | | | 軸停止FB1 |
| 10 | MC_Stop_2 | MC_Stop | VAR | | | 軸停止FB2 |
| 11 | MC_Stop_3 | MC_Stop | VAR | | | 軸停止FB3 |
| 12 | leAcceleration | 倍精度実数 | VAR | | | 加速度 |
| 13 | leDeceleration | 倍精度実数 | VAR | | | 減速度 |
| 14 | leJerk | 倍精度実数 | VAR | | | ジャーク |
| 15 | lePosition1 | 倍精度実数 | VAR | | | 移動量 |
| 16 | leVelocity | 倍精度実数 | VAR | | | 速度 |
| 17 | bInSync | ビット | VAR | | | カム動作FB inSync出力 |
| 18 | bDone1 | ビット | VAR | | | 相対値位置決めFB1 Done出力 |
| 19 | bBusy1 | ビット | VAR | | | 相対値位置決めFB1 Busy出力 |
| 20 | bDone2 | ビット | VAR | | | 相対値位置決めFB2 Done出力 |
| 21 | bBusy2 | ビット | VAR | | | 相対値位置決めFB2 Busy出力 |
| 22 | bStopReq1 | ビット | VAR | | | 軸停止要求1 |
| 23 | bStopDone1 | ビット | VAR | | | 軸停止完了1 |
| 24 | bStopDone2 | ビット | VAR | | | 軸停止完了2 |
| 25 | bStopDone3 | ビット | VAR | | | 軸停止完了3 |
| 26 | bSyncMoveCMD | ビット | VAR | | | 相対位置決め始動 |
| 27 | bExecute_S | ビット | VAR | | | 実行指令 |
| 28 | bCommandAborted | ビット | VAR | | | 相対値位置決めFB 実行中断出力 |
| 29 | bError | ビット | VAR | | | 相対値位置決めFB Error出力 |
| 30 | | | | | | |

①

②

①プログラムエディタに各FBをドラッグ＆ドロップすると自動で追加されます。
②手動で登録します。

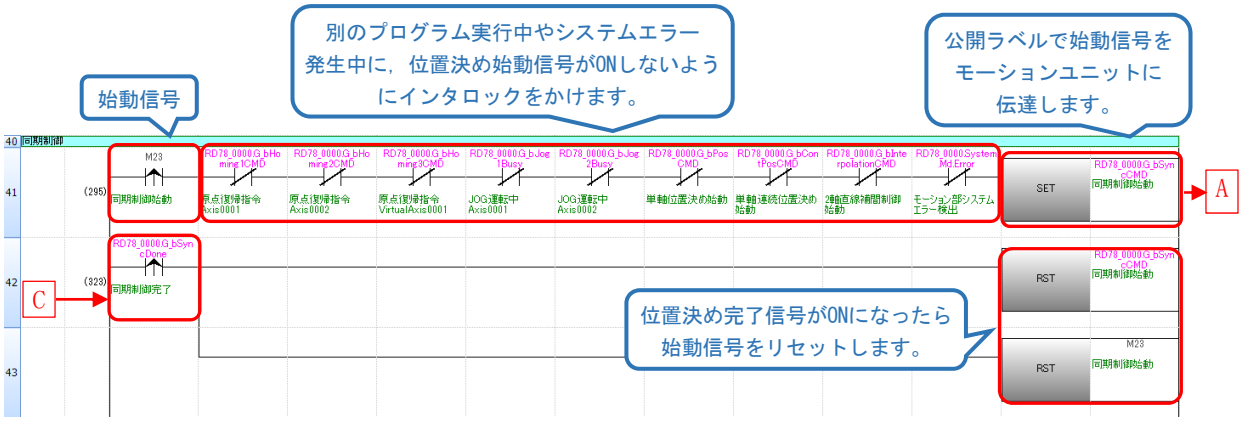
(3) プログラム例

| | プログラム内容 | 説　明 |
|---|---|---|
| ① | 2 軸直線補間制御始動に割り付けたデバイスの立ち上がりを，公開ラベルを介してモーションユニットに伝達します。 | シーケンサ CPU |
| ② | ・始動信号を受けて，位置決め用の各データを各ラベルに格納します。<br>全データ格納が完了したら，単軸同期用FB<br>(MC_GearIn,MC_CombineAxes，MC_CamIn)の実行要求をONします。<br>・Axis0002の軸状態が“7:同期運転中”になったことを確認し，Master(主軸)となるAxis0001を駆動させます。このとき，Axis0002は演算プロファイルに従って動作します。<br>・Axis0001の位置決め完了後，仮想ドライブ軸VirtualAxis0001を駆動させます。このときAxis0001は停止していますが，Axis0002は演算プロファイルに従って動作します。<br>・仮想ドライブ軸VirtualAxis0001の位置決めが完了するか，エラー発生または実行中断で，実ドライブ軸Axis0002と，仮想連結軸LinkAxis0001，LinkAxis0002のMC_Stopを実行し，同期運転を解除します。<br>・実ドライブ軸Axis0002の同期状態が解除されたら，始動信号をリセットします。 | モーション<br>ユニット |
| ③ | 別のプログラム実行中やシステムエラー発生中に，同期制御が始動しないようにインタロックをかけます。 | シーケンサ CPU |

[始動信号]

| | |
|---|---|
| 同期制御始動 | M23 |

[シーケンサCPU]



［モーションユニット］

```
//-----同期制御用データ設定・実行指令&リセット-----
IF G_bSyncCMD THEN;
    leePosition1    := 400000.0;
    leVelocity      := 50000.0;
    leAcceleration  := 100000.0;
    leDeceleration  := 100000.0;
    leJerk          := 0.0;
    bExecute_S := TRUE;
ELSE
    bExecute_S := FALSE;
END_IF;

//-----同期制御-----
//ギア動作
MC_GearIn_1(
    Master             := Axis0001.AxisRef ,
    Slave              := LinkAxis0001.AxisRef ,
    Execute            := bExecute_S ,
    RatioNumerator     := 1 ,
    RatioDenominator   := 2 ,
    MasterValueSource  := MC_SOURCE__mcSetValue ,
    Acceleration       := leAcceleration ,
    Deceleration       := leDeceleration ,
    Jerk               := leJerk ,
    Busy               => bGearInBusy
);

//加減算位置決め
MC_CombineAxes_1(
    Master1            := LinkAxis0001.AxisRef ,
    Master2            := VirtualAxis0001.AxisRef ,
    Slave              := LinkAxis0002.AxisRef ,
    Execute            := bExecute_S ,
    CombineMode        := MC_COMBINE_MODE__mcAddAxes ,
    GearRatioNumeratorM1   := 1 ,
    GearRatioDenominatorM1 := 1 ,
    GearRatioNumeratorM2   := 1 ,
    GearRatioDenominatorM2 := 2 ,
    MasterValueSourceM1    := MC_SOURCE__mcSetValue ,
    MasterValueSourceM2    := MC_SOURCE__mcSetValue ,
    Busy                   => bCombineAxesBusy
);

//カム動作
MC_CamIn_1.CamTableID.ProfileID := ProfileData0001.ProfileData.ID;
```

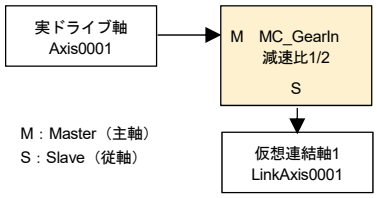
A

B

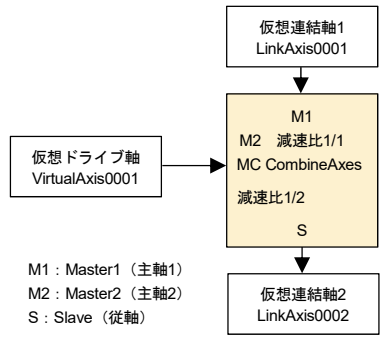
同期制御始動信号がONになったら
位置決め用データをラベルに格納します。

位置決め用データをラベルに格納したら
同期制御用FBを実行します。

Axis0001からの入力にギア比を乗じてLinkAxis0001に出力します。




LinkAxis0001からの入力とVirtualAxis0001からの入力を合成し，ギア比を乗じてLinkAxis0002に出力します。




MC_CamIn_1のCamTableID入力は，メンバのProfileIDに(ProfaileData名).ProfileData.IDを指定します。
これにより，どのプロファイルを使用するかを指定します。

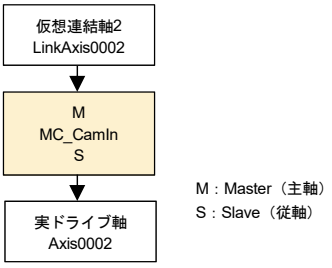
(次ページへ)

CamTableID入力は45行目で設定しているため省略します。

```
47  MC_CamIn_1(
48      Master           := LinkAxis0002.AxisRef ,
49      Slave            := Axis0002.AxisRef ,
50      Execute          := bExecute_S ,
51      MasterOffset     := 0.0 ,
52      SlaveOffset      := 0.0 ,
53      MasterScaling    := 1.0 ,
54      SlaveScaling     := 1.0 ,
55      MasterStartDistance := 0.0 ,
56      MasterSyncPosition  := 0.0 ,
57      StartMode           := MC_START_MODE__mcImmediate ,
58      MasterValueSource   := MC_SOURCE__mcSetValue ,
59      InSync              => bInSync ,
60      Busy                => bCamInBusy
61  );
62
```

B

LinkAxis0002を入力として，Axis0002を指定された演算プロフィルで駆動します。




Axis0002のステータスが“7:同期運転中”になったら，位置決めを始動します。

```
63  //入力軸始動
64  bSyncMoveCMD  := G_bSyncCMD & ( Axis0002.Md.AxisStatus = MC_AXIS_STATUS__SynchronizedMotion ) ;
65
66  //相対値位置決め 実ドライブ軸  Axis0001
67  MC_MoveRelative_1(
68      Axis         := Axis0001.AxisRef ,
69      Execute      := bSyncMoveCMD ,
70      Distance     := lePosition1 ,
71      Velocity     := leVelocity ,
72      Acceleration := leAcceleration ,
73      Deceleration := leDeceleration ,
74      Jerk         := leJerk ,
75      Done         => bDone1 ,
76      Busy         => bBusy1
77  );
78
79  //相対値位置決め  仮想ドライブ軸 VirtualAxis0001
80  MC_MoveRelative_2(
81      Axis         := VirtualAxis0001.AxisRef ,
82      Execute      := bDone1 & Axis0001.Md.CmdInPos ,
83      Distance     := lePosition1 ,
84      Velocity     := leVelocity ,
85      Acceleration := leAcceleration ,
86      Deceleration := leDeceleration ,
87      Jerk         := leJerk ,
88      Done         => bDone2 ,
89      Busy         => bBusy2
90  );
91
92  //FB実行中断出力
93  bCommandAborted := MC_GearIn_1.CommandAborted OR
94                     MC_CombineAxes_1.CommandAborted OR
95                     MC_CamIn_1.CommandAborted OR
96                     MC_MoveRelative_1.CommandAborted OR
97                     MC_MoveRelative_2.CommandAborted;
98
99  //FBエラー出力
100 bError := MC_GearIn_1.Error OR MC_CombineAxes_1.Error OR
101            MC_CamIn_1.Error OR MC_MoveRelative_1.Error OR
102            MC_MoveRelative_2.Error;
103
104 //位置決め完了時、または実行中断時、エラー発生時に軸停止信号をON
105 bStopReq1 := ( bDone2 & VirtualAxis0001.Md.CmdInPos ) OR bError OR bCommandAborted;
106
```

Axis0001位置決め

Axis0001の位置決めが完了したらVirtualAxus0001の位置決めを始動します。

VirtualAxis0001位置決め

VirtualAxis0001の位置決め完了，もしくはエラー，実行中断で軸停止信号(bStopReq1)をONします。

(次ページへ)

```
107 //同期終了
108 MC_Stop_1(
109     Axis    := Axis0002.AxisRef ,
110     Execute := bStopReq1 ,
111     Done    => bStopDone1
112 );
113
114 MC_Stop_2(
115     Axis    := LinkAxis0001.AxisRef ,
116     Execute := bStopReq1 ,
117     Done    => bStopDone2
118 );
119
120 MC_Stop_3(
121     Axis    := LinkAxis0002.AxisRef ,
122     Execute := bStopReq1 ,
123     Done    => bStopDone3
124 );
125
126 //同期終了後、始動要求信号をリセット
127 RST( bStopDone1 OR System.Md.Error, bSyncMoveCMD );
128 RST( bStopDone1 OR System.Md.Error, bStopReq1 );
129
130 //シーケンサに送る <= 位置決め完了信号・FB実行中断・FBエラーのいずれかの出力がONする ＊制御終了信号
131 G_bSyncDone := ( bStopDone2 & VirtualAxis0001.Md.CmdInPos ) OR bCommandAborted OR bError ;
132
```

Axis0002, LinkAxis0001.LinkAxis0002の
同期制御を解除します。

C

## 5.12　エラーリセット（プログラム名：ErrorReset）

各軸で発生したエラーをリセットします。

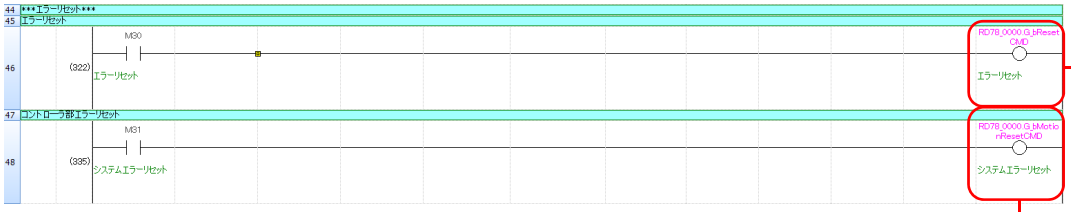
### (1) 使用するFB

| 種　別 | 命　令 | 内　容 |
|---|---|---|
| MCFB(管理系) | MC_Reset | 軸のエラー・警告をリセットします。 |
| | MC_GroupReset | 軸グループのエラー・警告をリセットします。 |
| | MCv_MotionErrorReset | モーションシステムの全エラー・警告をリセットします。 |

### (2) プログラム例

エラーリセットのプログラム例を示します。
エラーリセット始動信号に割り付けたデバイスのON/OFFを，公開ラベルを介してモーションユニットに伝達します。

［シーケンサCPU］




［モーションユニット］

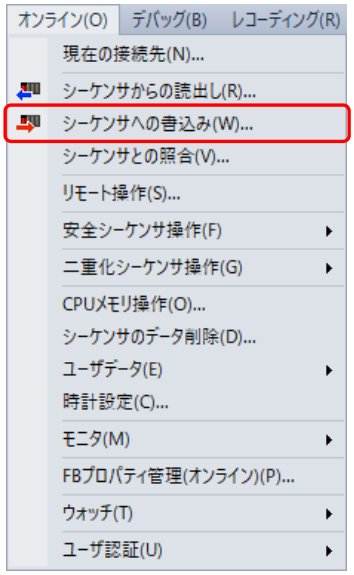
```
1
2  // エラーリセット
3  MC_Reset_1(Axis:= Axis0001.AxisRef          ,Execute:= G_bResetCMD );
4  MC_Reset_2(Axis:= Axis0002.AxisRef          ,Execute:= G_bResetCMD );
5  MC_Reset_3(Axis:= VirtualAxis0001.AxisRef   ,Execute:= G_bResetCMD );
6  MC_Reset_4(Axis:= LinkAxis0001.AxisRef      ,Execute:= G_bResetCMD );
7  MC_Reset_5(Axis:= LinkAxis0002.AxisRef      ,Execute:= G_bResetCMD );
8
9  // 軸グループエラーリセット
10 MC_GroupReset_1(AxesGroup:= AxesGroup001.AxesGroupRef ,Execute:= G_bResetCMD );
11
12 // コントローラエラーリセット
13 MCv_MotionErrorReset_1(Execute:= G_bMotionResetCMD );
14
```

## 5.13　動作確認

### 5.13.1　プログラムの変換と書き込み

シーケンサCPUとモーションユニットにプログラムを書き込みます。
MELSOFT GX Works3とモーション制御設定機能にて，それぞれプログラム作成後，[変換]⇒[全変換]を選択し，プログラムの全変換を行ないます。

全変換でエラーがないことを確認したら，シーケンサCPUとモーションユニットへプログラムを書き込んでください。
・MELSOFT GX Works3　　：[オンライン] ⇒ [シーケンサへの書込み]
・モーション制御設定機能　：[オンライン] ⇒ [ユニットへの書込み]




[MELSOFT GX Works3]

[モーション制御設定機能]

(注)モーションユニット側の公開ラベルの設定を変更した場合は，公開ラベルの反映後，シーケンサ CPU のプログラムの変換と書き込みを行なってください。

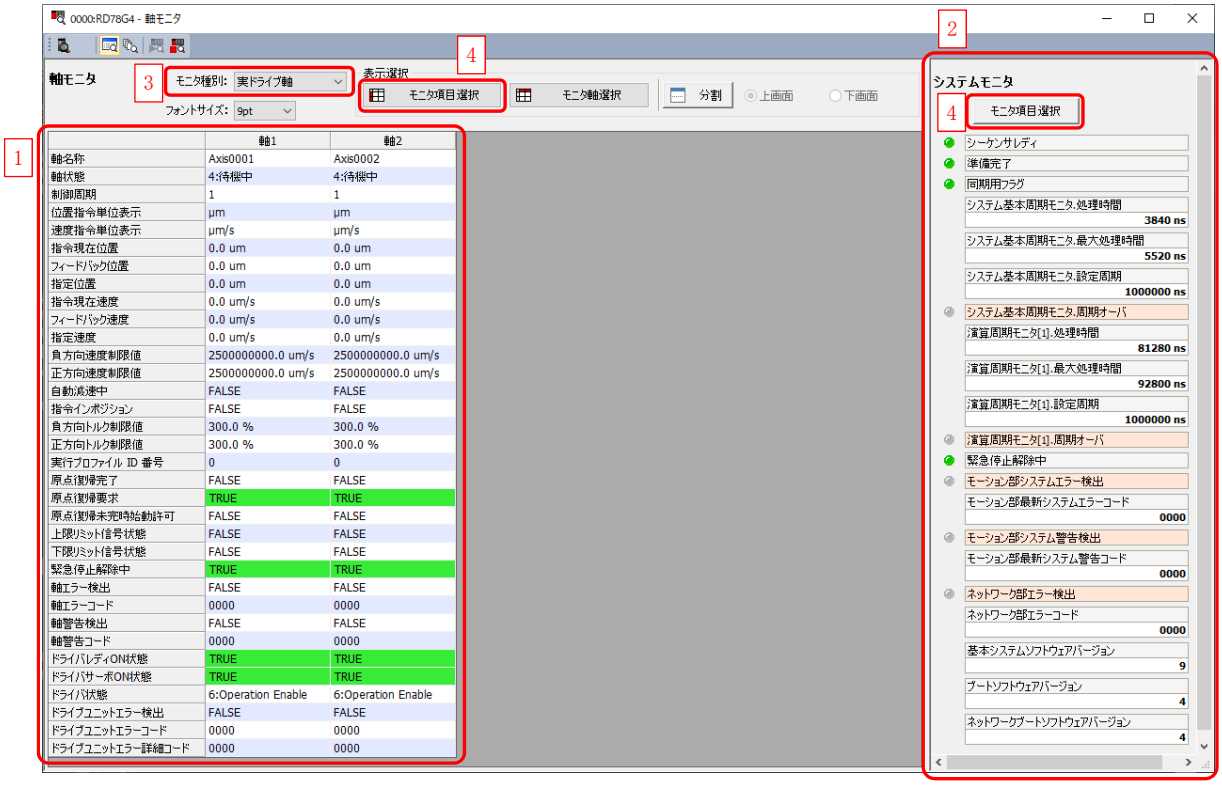
## 5.13.2　軸モニタ

全ての運転軸の現在値，エラーコードを一括してモニタ表示します。
システム稼働中に，現在値の確認，エラーの発生の有無を確認することができます。

### (1) 表示方法

モーション制御設定機能で，ナビゲーションウィンドウの“オンライン”⇒“モーションモニタ”⇒“軸モニタ”を選択します。

### (2) 表示内容




| No. | 内　容 |
|---|---|
| 1 | 各軸のモニタ項目を表示します。 |
| 2 | システムのモニタ項目を表示します。 |
| 3 | モニタする軸の種別を切り換えます。 |
| 4 | モニタする内容を追加/削除することができます。 |

### 5.13.3　プログラムモニタ

モニタ機能を使用して，実行プログラムをプログラムエディタ上で確認します。

(1) 表示方法

[シーケンサCPU]

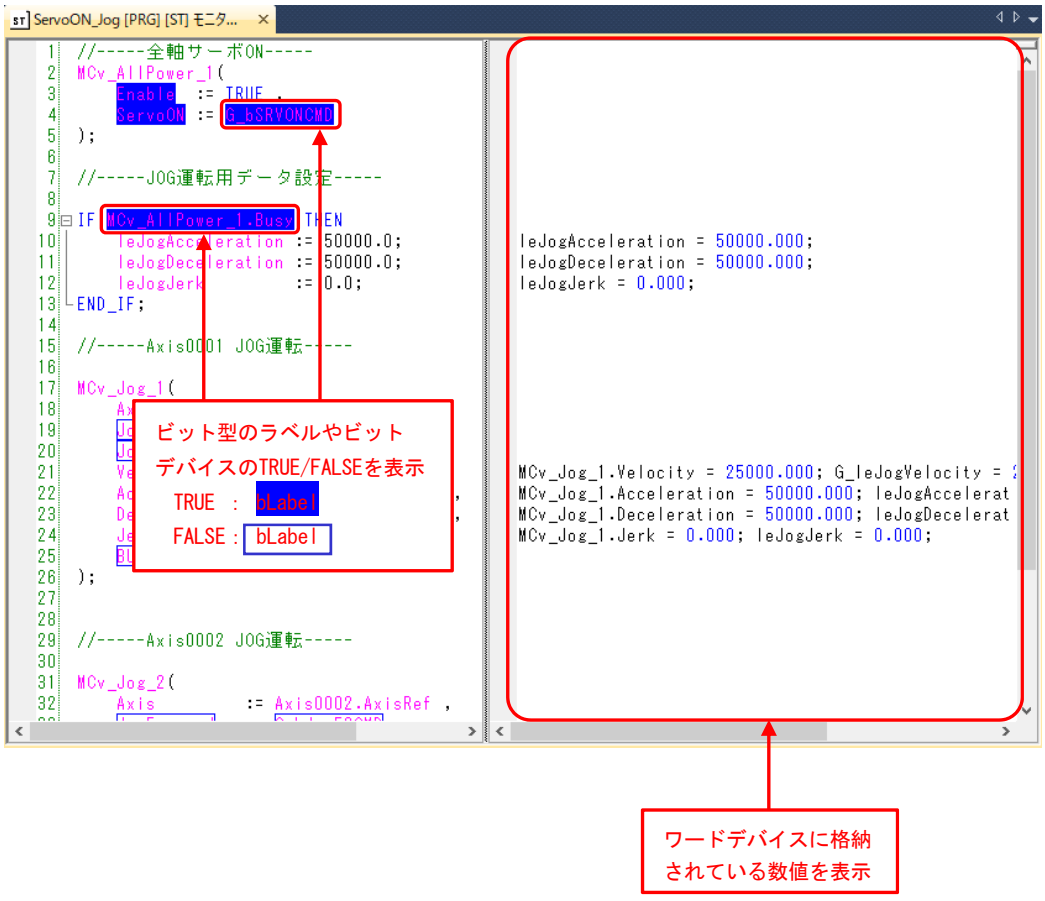
MELSOFT GX Works3で[オンライン]⇒[モニタ]⇒[モニタ開始(全ウィンドウ)]を選択するか，ツールバーの[ ]アイコンをクリックします。

[モーションユニット]

モーション制御設定機能で，[オンライン]⇒[モニタ]⇒[モニタ開始(全ウィンドウ)]を選択するか，ツールバーの[ ]アイコンをクリックします。

(2) 表示内容

“モーション制御設定機能”の画面は，モニタを開始すると以下のようになります。



### 5.13.4 ウォッチ

ウォッチ機能を使用して，デバイス，ラベルを登録して現在値を確認します。
確認対象は，ウォッチウィンドウに登録します。

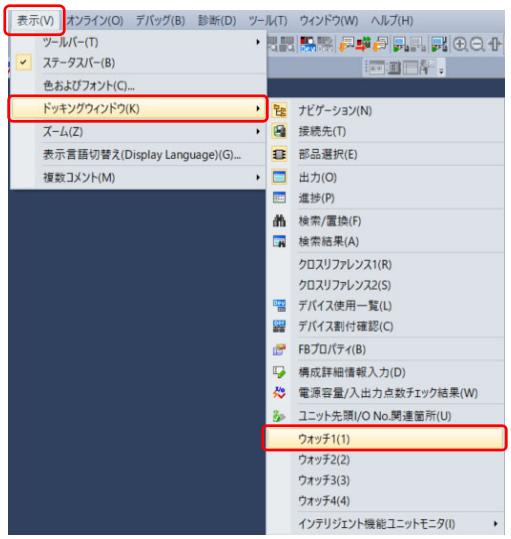
(1) 表示方法

[シーケンサCPU]

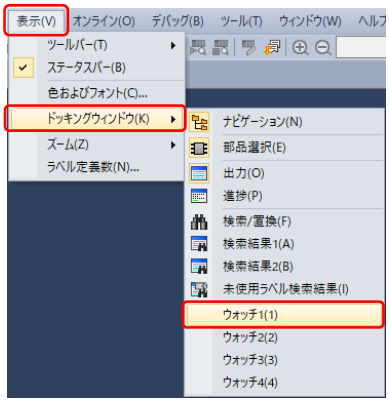
MELSOFT GX Works3で，[表示]⇒[ドッキングウィンドウ]⇒[ウォッチ1(1)～ウォッチ4(4)]を選択します。

[モーションユニット]

モーション制御設定機能で，[表示]⇒[ドッキングウィンドウ]⇒[ウォッチ1(1)～ウォッチ4(4)]を選択します。



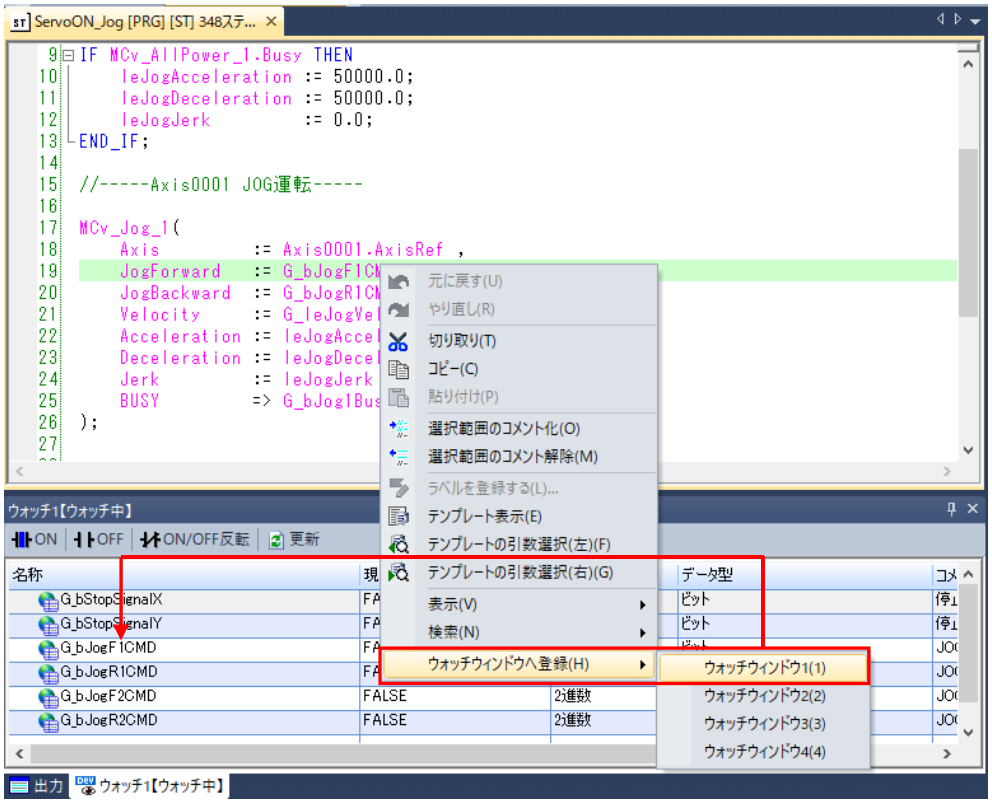

[MELSOFT GX Works3]




[モーション制御設定機能]

### (2) ウォッチウィンドウへの登録

ウォッチウィンドウの名称欄に，ラベルや構造体の名前を入力するか，プログラムエディタ上で登録するラベルや構造体を選択し，右クリック⇒[ウォッチウィンドウへ登録]⇒[ウォッチウィンドウ1(1)～ウォッチ4(4)]を選択します。




### (3) ウォッチの開始

MELSOFT GX Works3，またはモーション制御設定機能の[オンライン]⇒[ウォッチ]⇒[ウォッチ開始]を選択します。

### (4) 現在値の変更

ウォッチ中は，“現在値”に変更する値を直接入力します。
ビットデバイスの場合は行を選択し，[Shift]+ダブルクリック，または[Shift]+[Enter]で，ON/OFFを変更することができます。

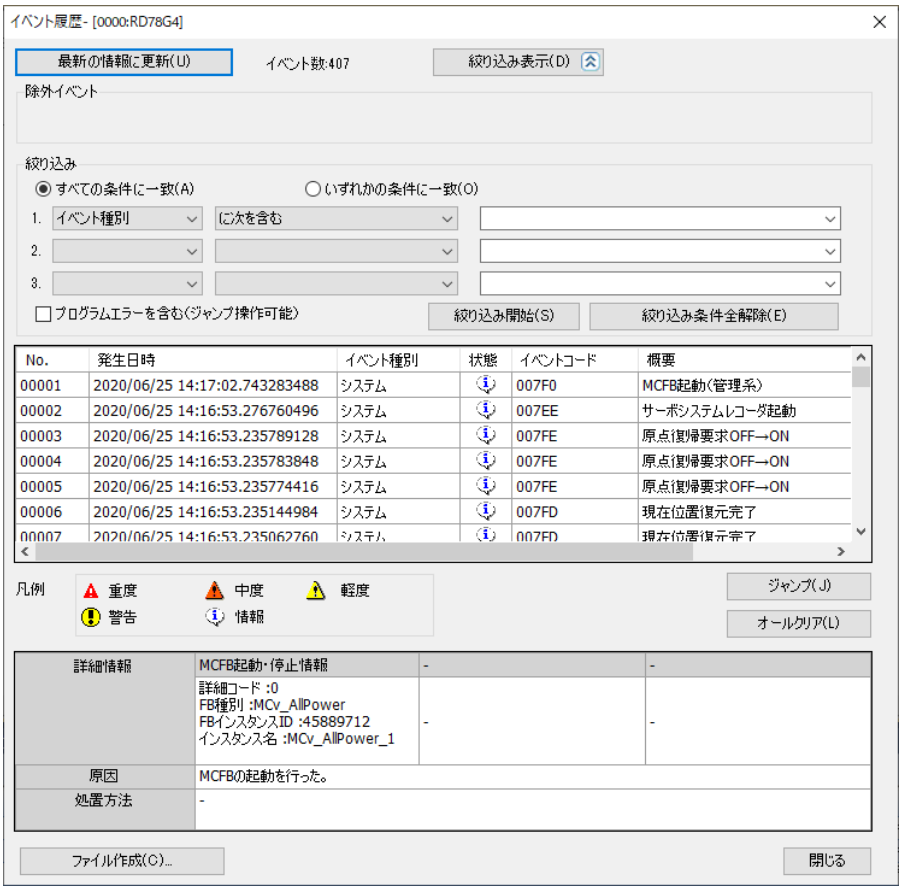
**[ポイント]**

サンプルプログラムには，ウォッチウィンドウ1に“G_bStopSignalX”と“G_bStopSignalY”が登録されています。プログラム動作中にこれらをONすることで，軸を停止させることができます。
“G_bStopSignalX”および“G_bStopSignalY”をシーケンサCPUから操作する場合は，公開ラベルのモーション制御属性を“READ(モーション⇒)”から“WRITE(⇒モーション)”に変更して，公開ラベルの反映を再度実施してください。

### 5.13.5　イベント履歴

モーションユニット側のイベント履歴は，モーション制御設定機能で，[オンライン]⇒[モーションモニタ]⇒[イベント履歴]を選択して表示される“イベント履歴”画面で確認できます。エラーが発生した場合，詳細情報が参照できます。

また，ここで記録される発生日時は，サーボアンプに記録されているアラーム発生日時と同期していますので，あわせて参照してください。




シーケンサCPU側のイベント履歴は，MELSOFT GX Works3の[診断]⇒[システムモニタ]で表示される“システムモニタ”画面の”イベント履歴”ボタンから確認できます。

# 付　録

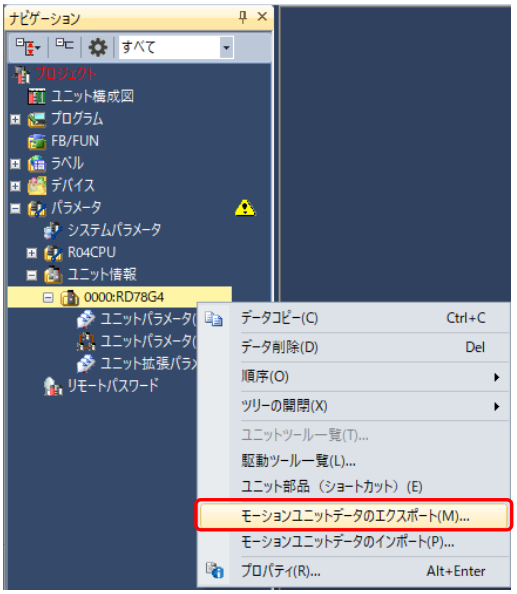
## 付1　プログラムの流用

サンプルプログラムのモーションユニットのパラメータやプログラムを，別のプログラムに流用する手順を説明します。使用するモーションユニットを変更する場合も，本手順にてデータの流用を行なってください。本章では，RD78G4からRD78GHVに変更する例で説明します。

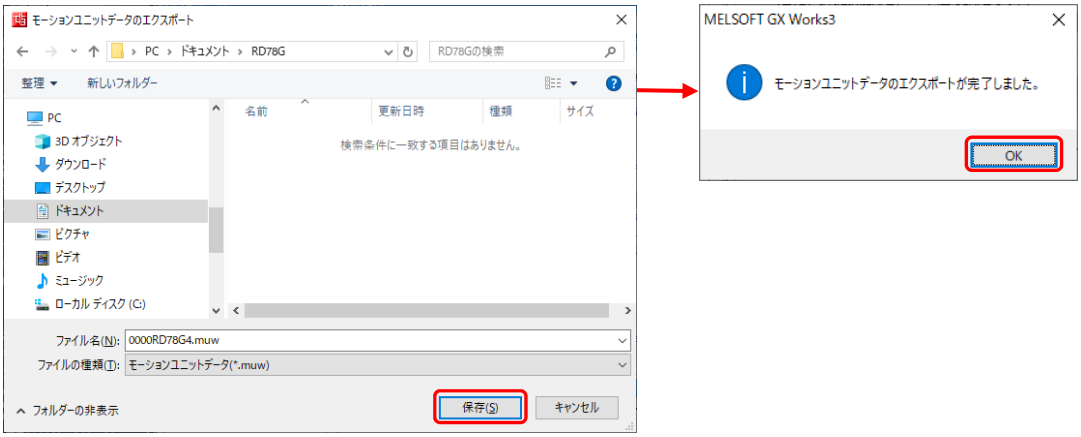
### 付1.1　モーションユニットデータのエクスポート

モーションユニットのパラメータとプログラムを1ファイルに出力します。

①MELSOFT GX Works3を起動して，流用元プログラムを開きます。ナビゲーションウィンドウで“パラメータ”⇒“ユニット情報”⇒“0000_RD78G4”を右クリックして，[モーションユニットデータのエクスポート]を選択します。



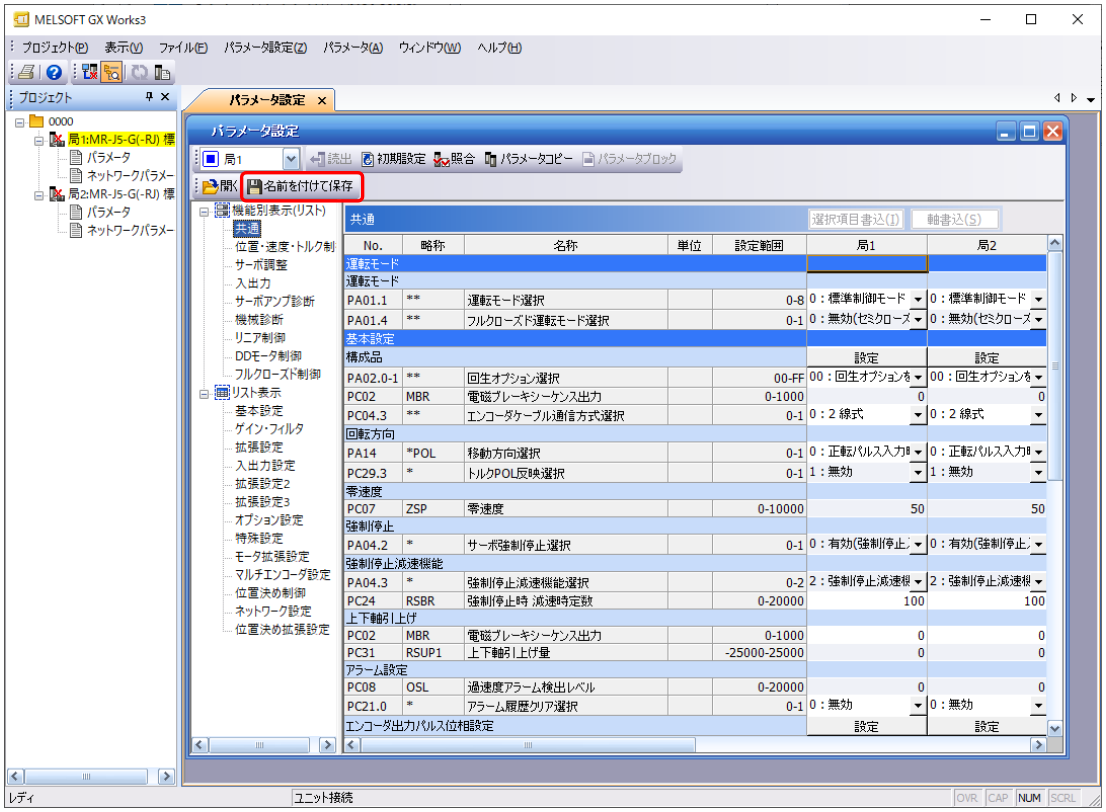

②“モーションユニットデータのエクスポート”画面で，エクスポートするフォルダを選択し，[保存]ボタンをクリックします。
確認画面で[OK]ボタンをクリックして，モーションユニットデータをmuwファイルに出力します。



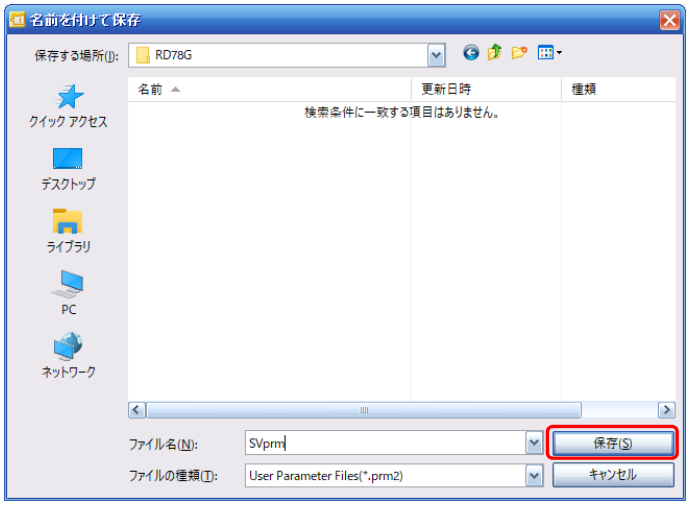
## 付1.2　サーボパラメータの保存

付1.1で出力したmuwファイルには，サーボパラメータが含まれていません。
モーションユニットからサーボアンプにサーボパラメータを書き込む場合は，流用元プログラムから別ファイルに保存します。

①“パラメータ設定”画面で，[名前を付けて保存]ボタンをクリックします。




②“名前を付けて保存”画面で，保存先のフォルダを選択し，[保存]ボタンをクリックします。
サーボパラメータをprm2ファイルに出力します。

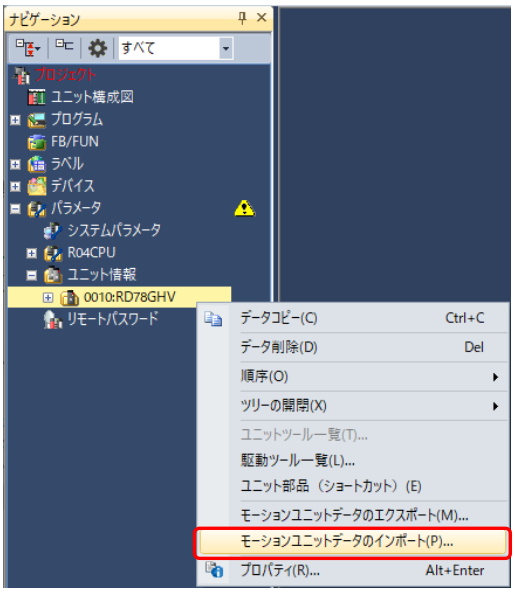


## 付1.3　流用先プログラムへの反映

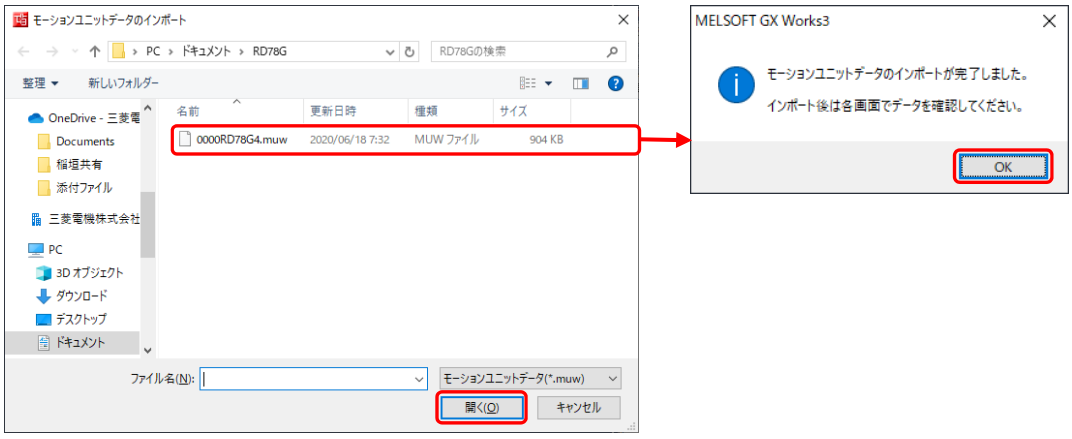
MELSOFT GX Works3を起動して，流用先のプログラムをユニット構成図まで作成します。

### (1) モーションユニットデータのインポート

①ナビゲーションウィンドウで，“パラメータ”⇒“ユニット情報”⇒“0010:RD78GHV”を右クリックして，[モーションユニットデータのインポート]を選択します。



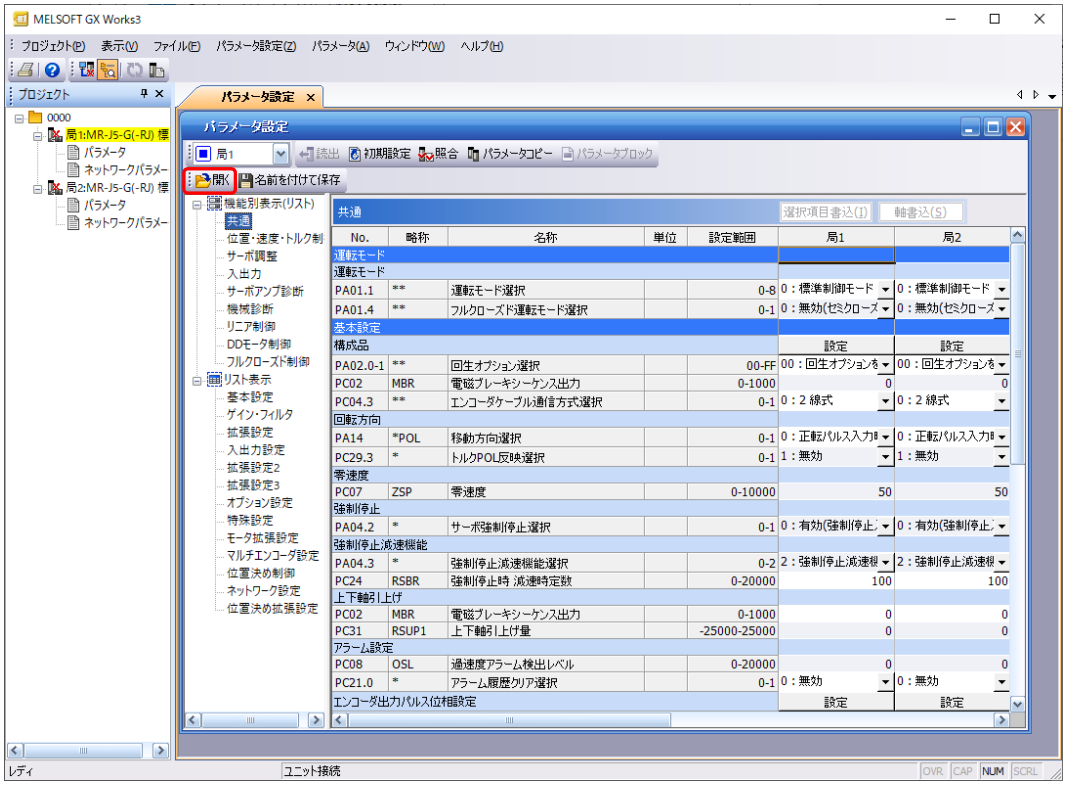

②“モーションユニットデータのインポート”画面で，付1.1で出力したmuwファイルを選択し，[開く]ボタンをクリックします。
確認画面で[OK]ボタンをクリックして，モーションユニットデータをインポートします。



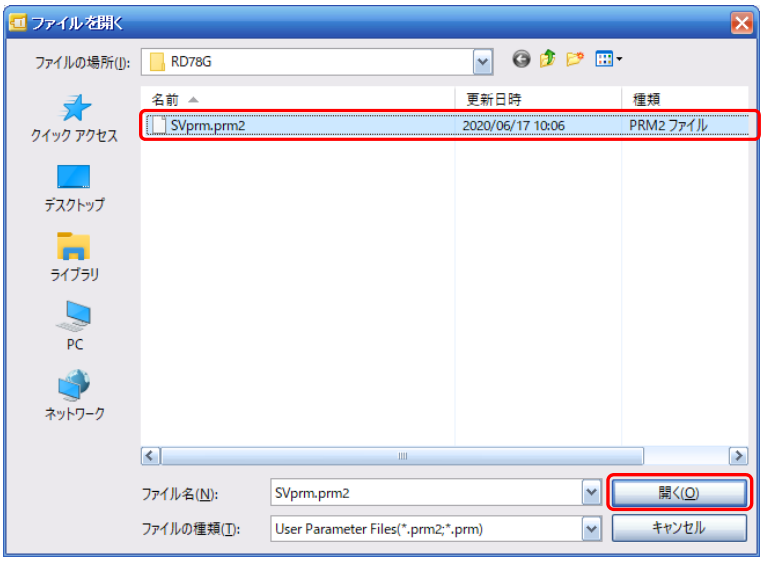
### (2) サーボパラメータの読み出し

モーションユニットからサーボパラメータを取り込む場合は，付1.2で出力したサーボパラメータのファイルを読み出します。

①流用先プログラムの“パラメータ設定”画面で，[開く]ボタンをクリックします。

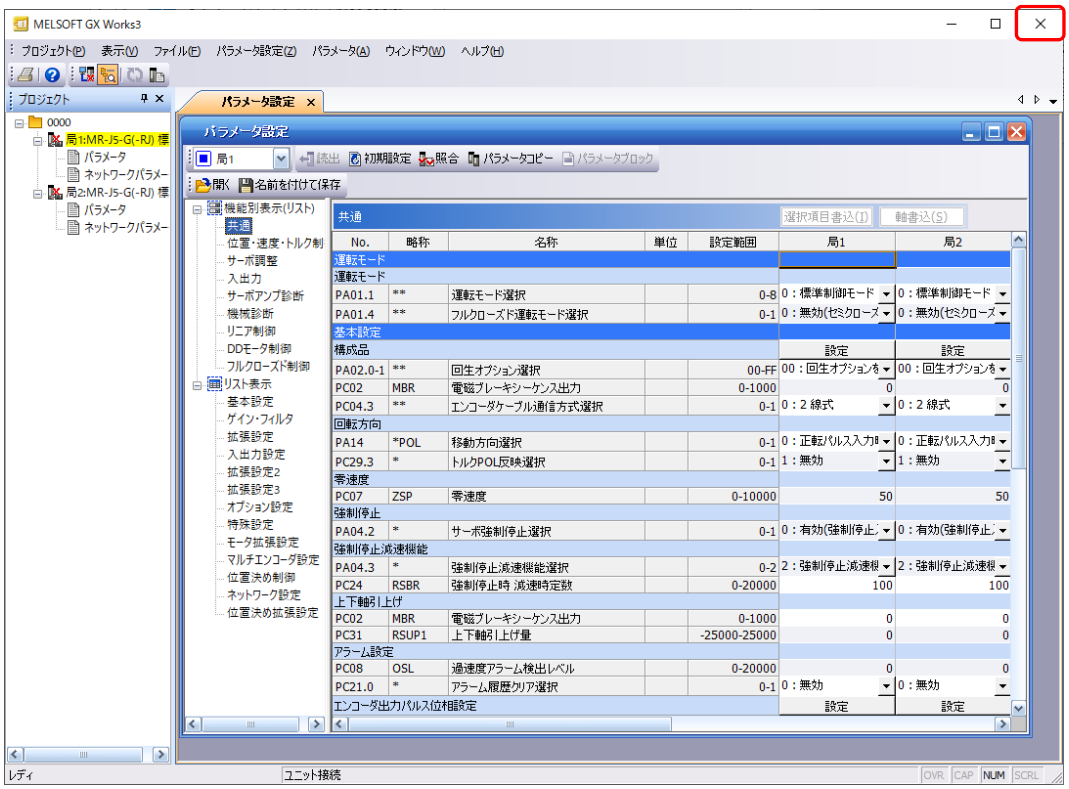



②“ファイルを開く”画面で，付1.2で出力したprm2ファイルを選択し，[開く]ボタンをクリックします。

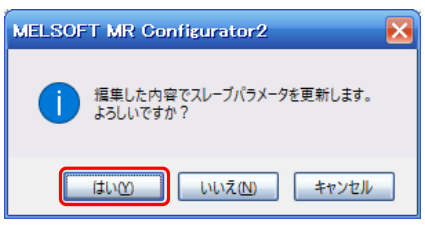


③サーボパラメータを取り込む局を選択し，[コピー]ボタンをクリックします。

④サーボパラメータが読み出されたことを確認し，“パラメータ設定”画面右上の[×]ボタンをクリックします。



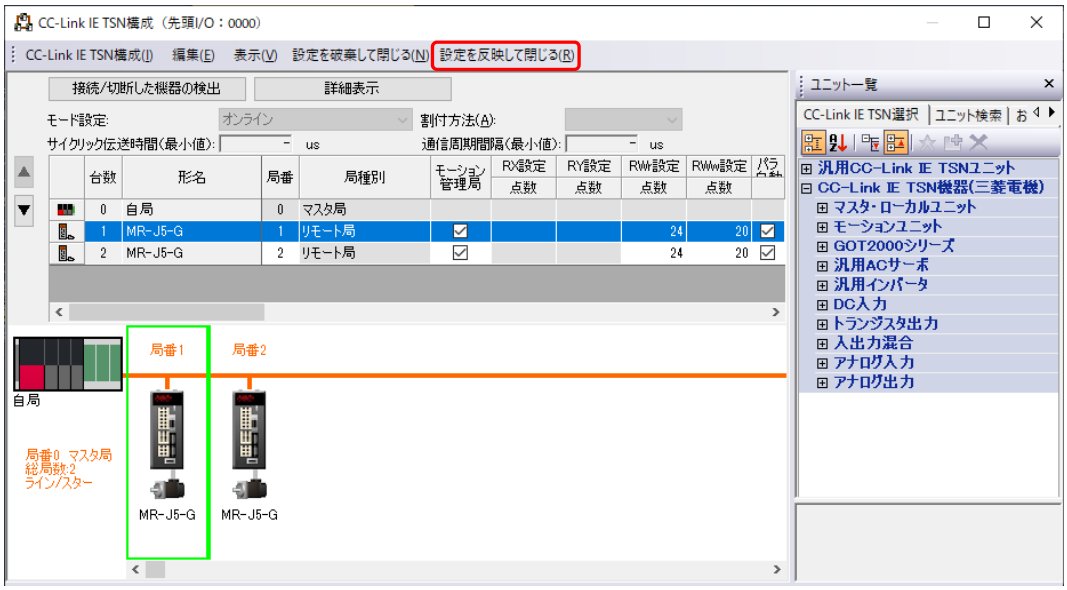

確認画面で[はい]ボタンをクリックして，パラメータを更新します。



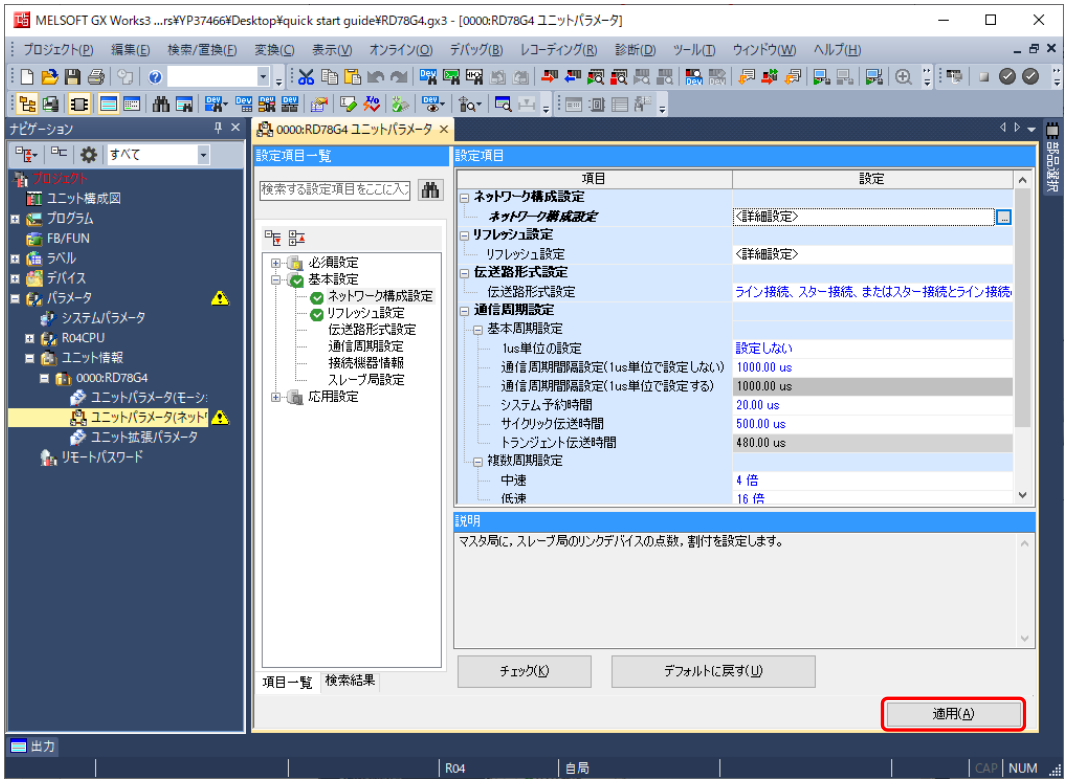
### (3) パラメータの確定

CC-Link IE TSN構成やユニットパラメータ(ネットワーク)を確定します。

①“CC-Link IE TSN構成”ウィンドウで，[設定を反映して閉じる]をクリックして，データを反映します。




②パラメータエディタ(ユニットパラメータ)で[適用]ボタンをクリックして，モーションユニットのパラメータを反映します。



### (4) プログラムの修正

先頭入出力番号が流用元のプログラムと流用先のプログラムで異なっている場合は，以下を変更してください。

・シーケンスプログラムのシーケンサレディ[Y0]
・準備完了[X0]
・同期用フラグ[X1]のデバイス番号
・バッファメモリアドレス(U**¥G～)のユニット番号
・公開ラベルの使用箇所

特に，RD78GHV，RD78GHWに変更した場合は，入出力点数が「空き16点＋32点」のため，スロットNo.0～1に設置したときのシーケンサレディは[Y10]になります。必ず確認してください。
また，公開ラベル名の先頭が[RD78_0000.****]から[RD78_0010.****]に変更されます。
付1.4の公開ラベルの反映を実行後，文字列置換でプログラム内のラベル名を全て変更してください。

## 付1.4 全変換，公開ラベルの反映

モーションユニット側を先に全変換し，その後，公開ラベルの反映を行ないます。公開ラベルが反映されたら，シーケンサCPU側の全変換を行ないます。

先にシーケンサCPU側の全変換を行なうと，公開ラベルが定義されていないため，エラーが発生します。

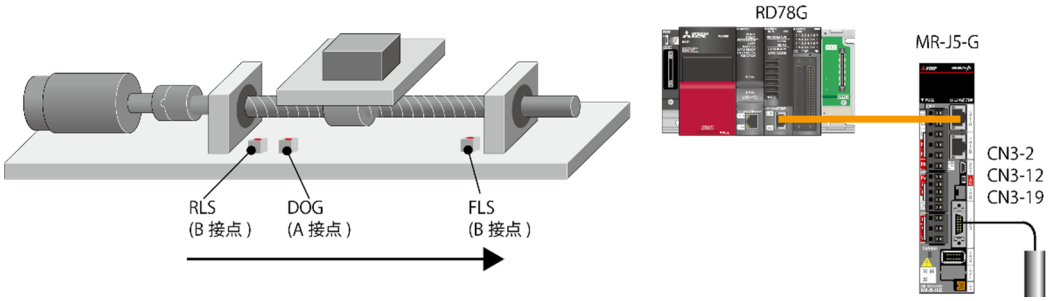
## 付2　外部入力信号を使用する場合

ハードウェアストロークリミット信号や近点ドグ信号に，外部からの入力信号を使用することができます。

| 使用可能な外部入力信号 | 本書参照先 |
|---|---|
| サーボアンプの DI 信号 | 付 2.1 |
| シーケンサ CPU の入力信号 | 付 2.2 |
| リモート入力ユニットの入力信号 | 付 2.3 |

## 付2.1　サーボアンプのDI信号を使用する場合

ハードウェアストロークリミット信号にサーボアンプのLSP/LSN信号を使用する場合，およびサーボアンプのDOG信号を使用して原点復帰を行なう場合について説明します。




### (1) ハードウェアストロークリミット

ハードウェアストロークリミットの配線は，負論理(b接点)を推奨します。

**[注　意]**
**ハードウェアストロークリミットの配線に正論理(a接点)を使用すると，断線やセンサ故障時に重大な事故が発生する可能性があります。**

(a) サーボアンプの信号の接続先とセンサの設置場所

サーボパラメータ[Pr.PA14(移動方向選択)]の設定により，センサの設置場所が変更になりますので，サーボアンプに接続するリミットスイッチを以下のように設置してください。

1) [Pr.PA14]：0（位置決めアドレス増加時CCWまたは正方向）の場合
- LSP信号：位置決めアドレス増加側の上限ストロークリミット(FLS)
- LSN信号：位置決めアドレス減少側の下限ストロークリミット(RLS)

2) [Pr.PA14]：1（位置決めアドレス増加時CWまたは負方向）の場合
- LSP信号：位置決めアドレス減少側の下限ストロークリミット(RLS)
- LSN信号：位置決めアドレス増加側の上限ストロークリミット(FLS)

(b) サーボパラメータ

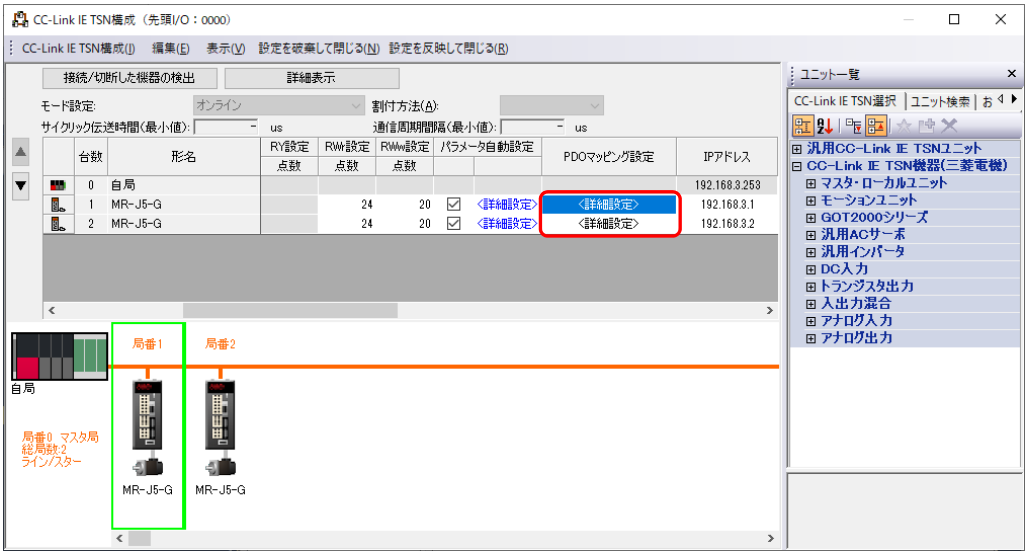
サーボパラメータが以下の設定になっていることを確認します。

| 番　号 | 項　目 | 設定値 |
|---|---|---|
| Pr. PD41. 2 | リミットスイッチ有効状態選択 | 1：原点復帰モードのみ有効 |
| Pr. PD41. 3 | センサ入力方式選択 | 0：サーボアンプより入力 |

| No. | 略称 | 名称 | 単位 | 設定範囲 | 局1 | 局2 |
|---|---|---|---|---|---|---|
| PD03.0-1 | * | デバイス選択DI1 | | 00-7F | 0A | 0A |
| PD04.0-1 | * | デバイス選択DI2 | | 00-7F | 0B | 0B |
| PD05.0-1 | * | デバイス選択DI3 | | 00-7F | 22 | 22 |
| PD51.0-1 | * | デバイス選択DI3-2 | | 00-7F | 62 | 62 |
| PD38.0-1 | * | デバイス選択DI4 | | 00-7F | 2C | 2C |
| PD39.0-1 | * | デバイス選択DI5 | | 00-7F | 2D | 2D |
| PD07.0-1 | * | デバイス選択DO1 | | 00-7F | 05 | 05 |
| PD08.0-1 | * | デバイス選択DO2 | | 00-7F | 04 | 04 |
| PD09.0-1 | * | デバイス選択DO3 | | 00-7F | 03 | 03 |
| デバイス割付 | | | | | 設定 | 設定 |
| PD01.0-7 | *DIA1 | 入力信号自動オン選択1 | | 00000000-00000FF0 | 00000000 | 00000000 |
| 入力フィルタ | | | | | | |
| PD11.0 | * | 入力信号フィルタ選択 | | 0-B | 7：3.500ms | 7：3.500ms |
| ALM出力 | | | | | | |
| PD14.1 | * | 警告発生時の出力デバイスの選択 | | 0-1 | 0：警告時にWNGがオン | 0：警告時にWNGがオン |
| アナログ出力 | | | | | | |
| アナログモニタ | | | | | | |
| PC09.0-1 | | アナログモニタ1出力選択 | | 00-1F | 00：サーボモータ速度(± | 00：サーボモータ速度(± |
| PC11 | MO1 | アナログモニタ1 オフセット | | -999-999 | 0 | 0 |
| PC10.0-1 | | アナログモニタ2出力選択 | | 00-1F | 01：トルクまたは推力(± | 01：トルクまたは推力(± |
| PC12 | MO2 | アナログモニタ2 オフセット | | -999-999 | 0 | 0 |
| ストロークリミット機能 | | | | | | |
| ストロークリミット機能 | | | | | | |
| PC19.0 | * | [AL. 099 ストロークリミット警告] 選択 | | 0-1 | 1：無効 | 1：無効 |
| PD41.2 | * | リミットスイッチ有効状態選択 | | 0-1 | 1：原点復帰モードの | 1：原点復帰モードのみ |
| PD41.3 | * | センサ入力方式選択 | | 0-1 | 0：サーボアンプより入力 | 0：サーボアンプより入力( |

(c) PDOマッピングの変更

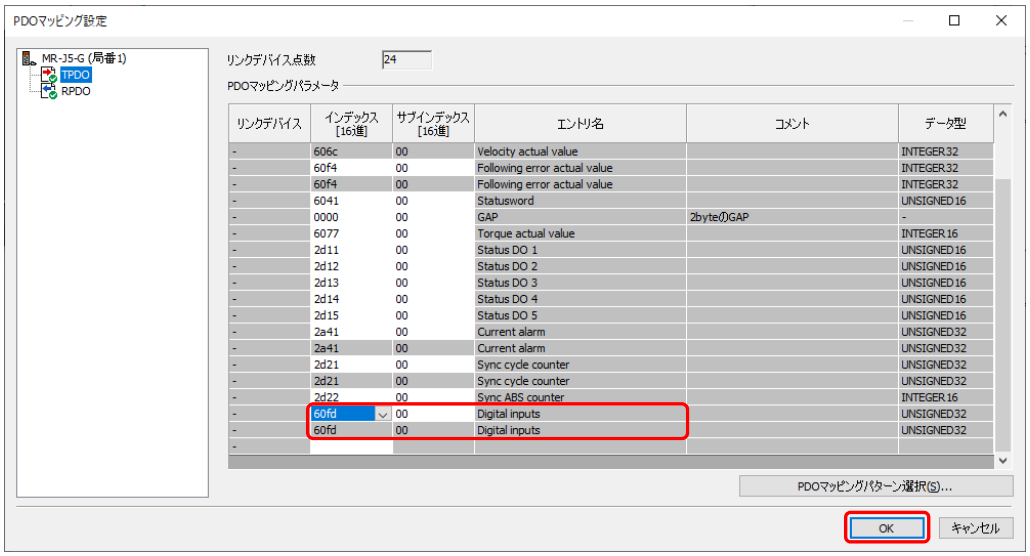
①“CC-Link IE TSN構成”ウィンドウで“PDOマッピング設定”の“<詳細設定>”をダブルクリックして，“PDOマッピング設定”画面を表示します。



②PDOマッピングパラメータの末尾に以下のとおり設定し，[OK]ボタンをクリックします。

| インデックス(16進) | サブインデックス(16進) |
|---|---|
| 60fd | 00 |

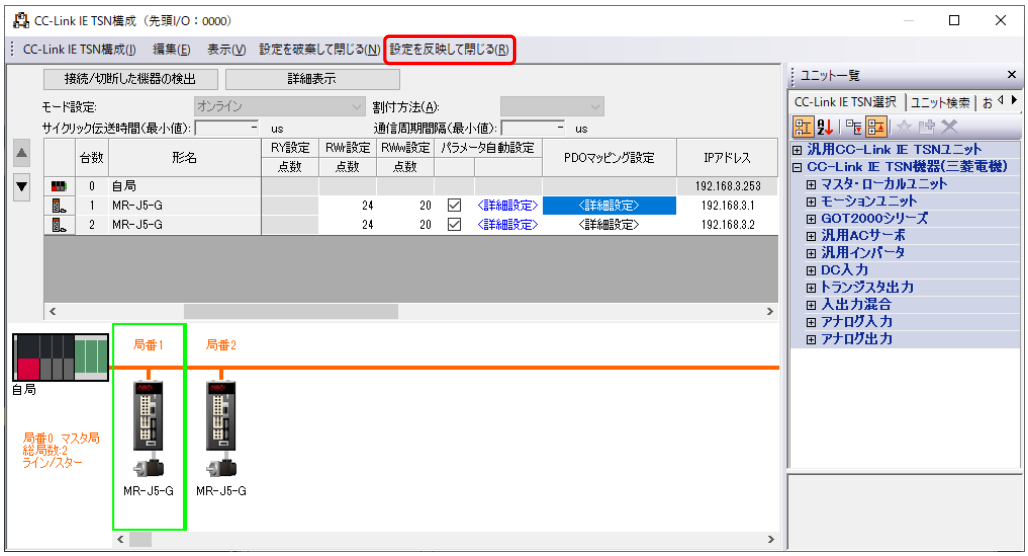
エントリ名にDigital inputsと表示されます。
(ダブルワード型のデータのため，2行表示)



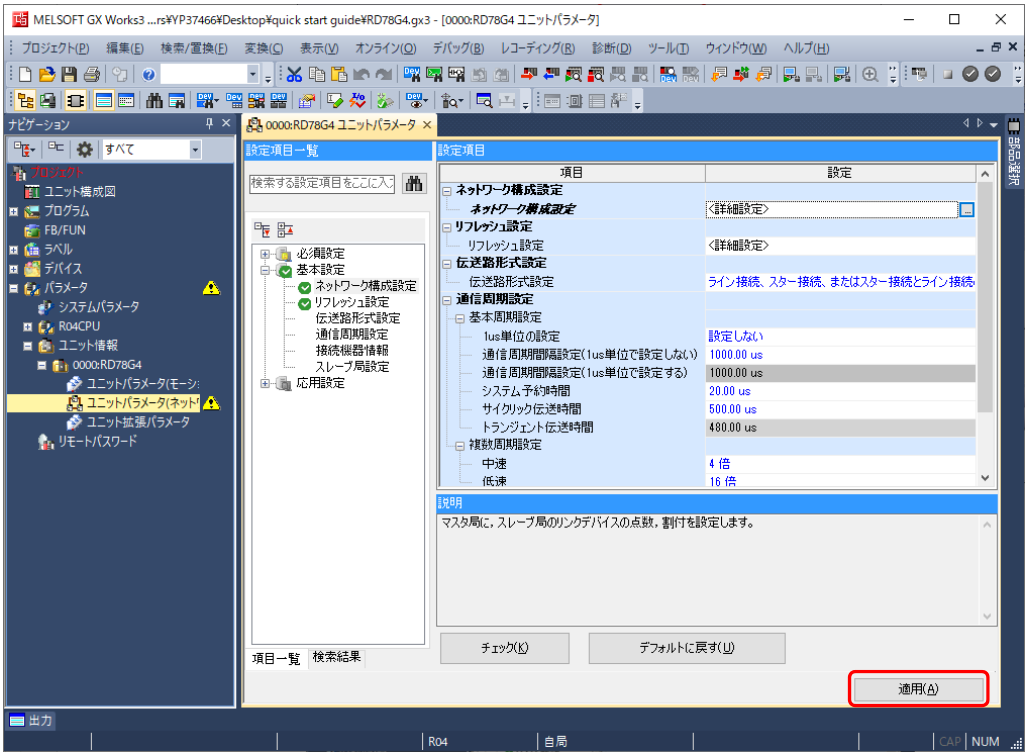

サーボアンプに接続しているハードウェアリミット信号を使用する全ての軸に対して，上記の設定を行ないます。

**[ポイント]**
“Digital inputs”は，サーボアンプの入力信号の状態が格納されるラベルです。
・bit0：Negative limit switch
・bit1：Positive limit switch
詳細は，「MR-J5-G/MR-J5W-G ユーザーズマニュアル(オブジェクトディクショナリ編)」の17.1節を参照してください。

③設定完了後，“CC-Link IE TSN構成”ウィンドウで，[設定を反映して閉じる]をクリックして，データを反映します。

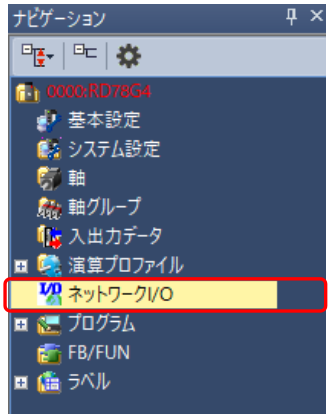



④パラメータエディタ(ユニットパラメータ)で[適用]ボタンをクリックして，モーションユニットのパラメータを反映します。

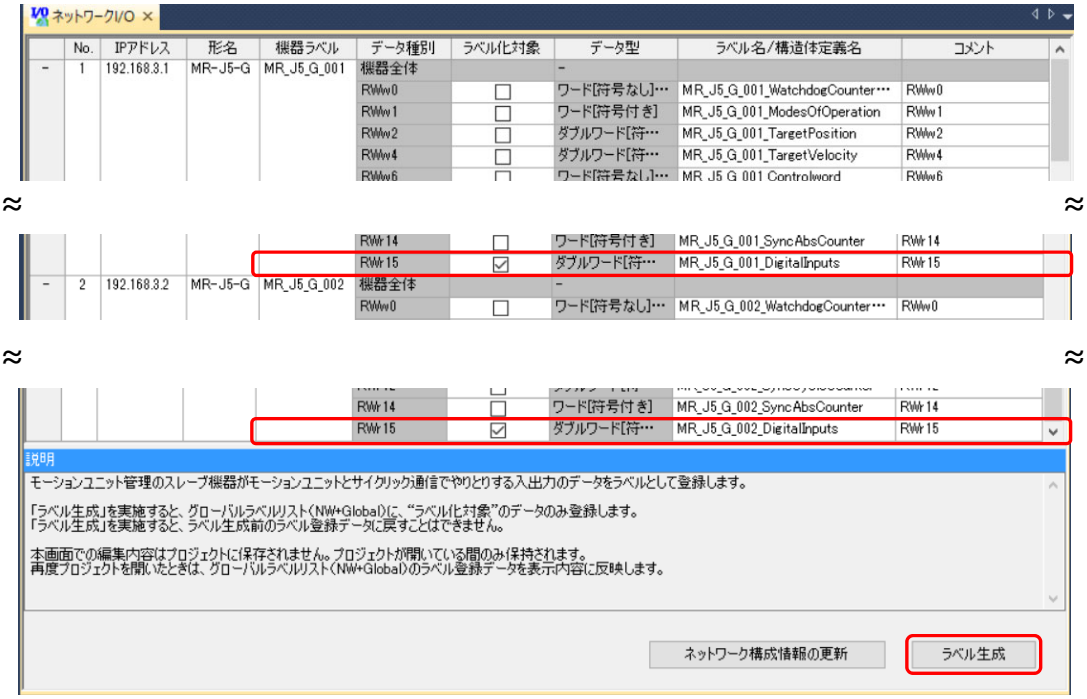


(d) モーションユニット側のパラメータの変更

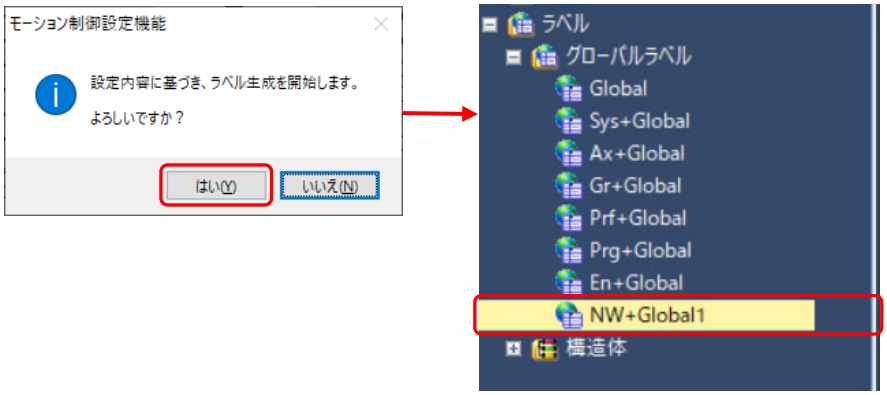
①モーション制御設定機能で，ナビゲーションウィンドウの“ネットワークI/0”をダブルクリックして，モーションユニット管理の機器を表示します。



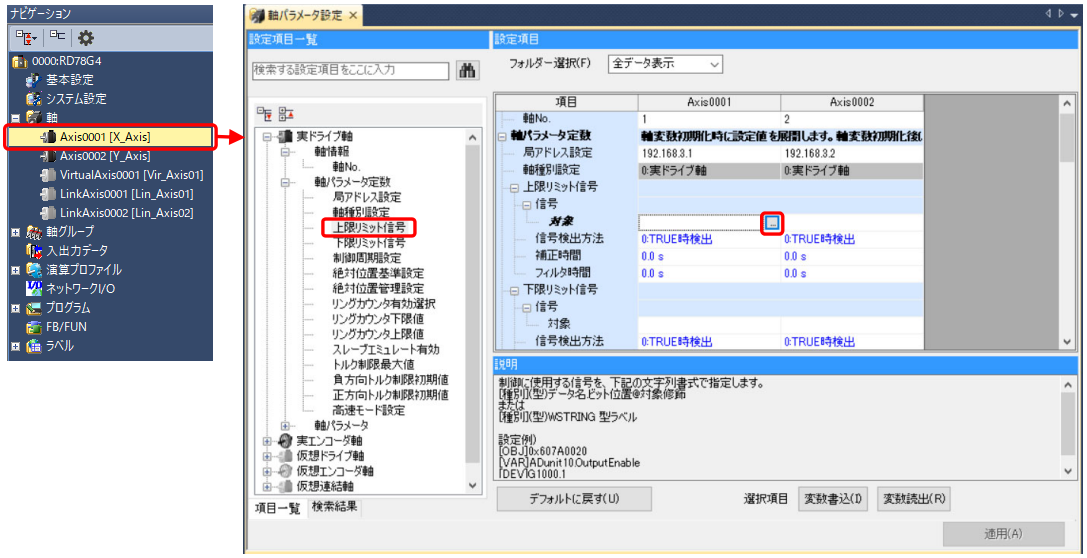

②ハードウェアストロークリミットを接続したサーボアンプのラベル[MR_J5_G_***_DigitalInputs]の“ラベル化対象”にチェックを入れ，[ラベル生成]ボタンをクリックします。



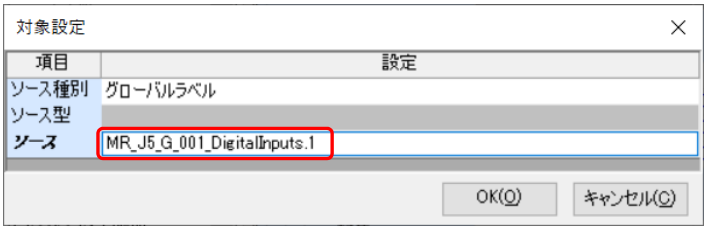
③確認画面で[はい]ボタンをクリックします。
ネットワークI/0ラベルが生成され，“グローバルラベル”の“NW+Global1”以下に格納されます。



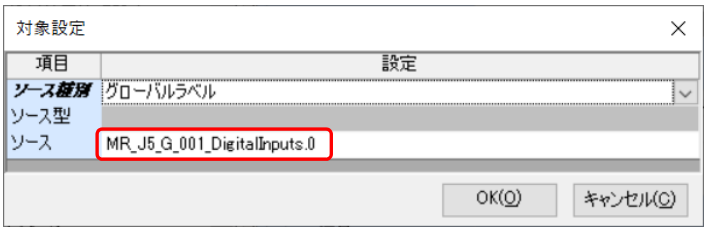

④実ドライブ軸の軸パラメータを開きます。
ナビゲーションウィンドウで，“軸”⇒“使用する軸名称(初期値：Axis0001)”をダブルクリックします。
“軸パラメータ設定”ウィンドウの“実ドライブ軸”⇒“上限リミット信号”⇒“対象”の[…]ボタンをクリックします。



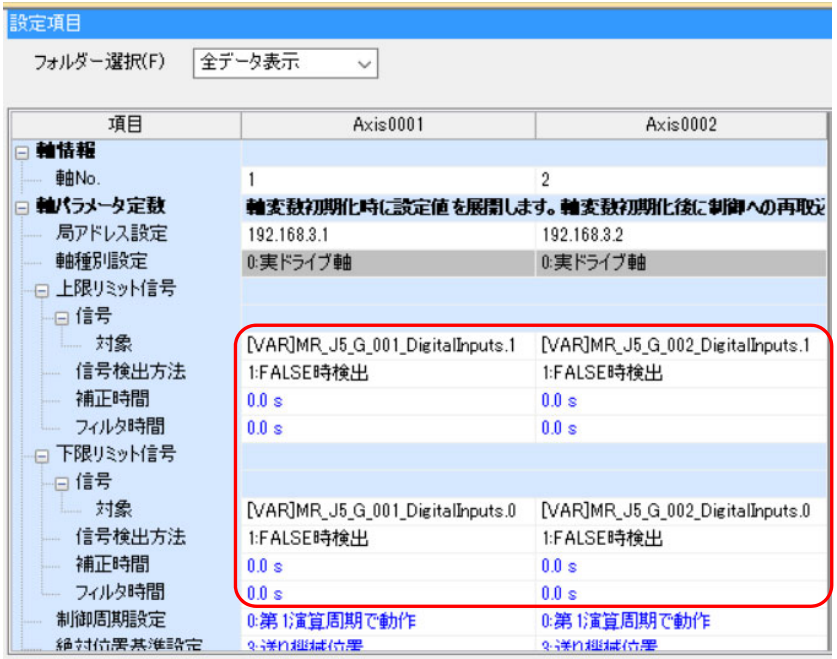
⑤“対象設定”画面で，“ソース”に“MR_J5_G_***_DigitalInputs.1”を入力します。




同様に，下限リミットの“対象設定”画面で，“ソース”に“MR_J5_G_***_DigitalInputs.0”を入力します。




サーボアンプに接続しているハードウェアリミット信号を使用する全ての軸に対して，上記の設定を行ないます。
軸ラベルと実際のサーボアンプの関係を間違えないように注意してください。
また，b接点を使用するため，信号検出方法を“1：FALSE時検出”に変更します。




以上で設定は完了です。全変換を行なってください。

(e) 動作確認

プログラムを書き込み，軸モニタの“上限リミット信号状態”，“下限リミット信号状態”で，それぞれの信号のON/OFFがモニタできるか確認してください。
リミット範囲内に復帰させる場合は，一度エラーリセットを実行してから，範囲内方向にJOG運転などで復帰させてください。

### (2) DOG信号を使用した原点復帰

近点ドグ式原点復帰などで，Homing SwitchにサーボアンプのDOG信号を使用する場合について説明します。

(a) サーボパラメータ

原点復帰方式と近点ドグ信号の極性を次のパラメータで設定します。

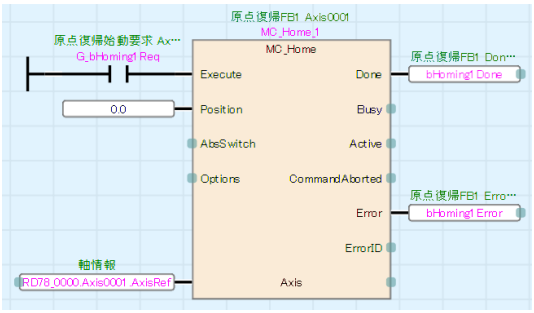
| 番　号 | 項　目 |
|---|---|
| Pr.PT29.0 | デバイス入力極性1 |
| Pr.PT45 | 原点復帰方式 |

原点復帰方式に応じて，以下から必要なパラメータを設定します。
（設定が必要なパラメータは，原点復帰方式によって変わります。）

| 番　号 | 項　目 |
|---|---|
| Pr.PT05 | 原点復帰速度 |
| Pr.PT06 | クリープ速度 |
| Pr.PT07 | 原点シフト量 |
| Pr.PT08 | 原点復帰位置データ |
| Pr.PT09 | 近点ドグ後移動量 |
| Pr.PT55.0 | 原点復帰時減速時定数選択 |
| Pr.PT56 | 原点復帰加速時定数 |
| Pr.PT57 | 原点復帰減速時定数 |

(b) プログラム

サーボアンプのDOG信号を使用する場合のプログラムでは，MC_HomeのAbsSwitch入力を省略します。




[FBDプログラム]

```
//原点復帰
MC_Home_1(
    Axis     := Axis0001.AxisRef ,
    Execute  := G_bHoming1CMD ,
    Position:= 0.0 ,
    Done     => bHoming1Done,
    Error    => bHoming1Error
);
```

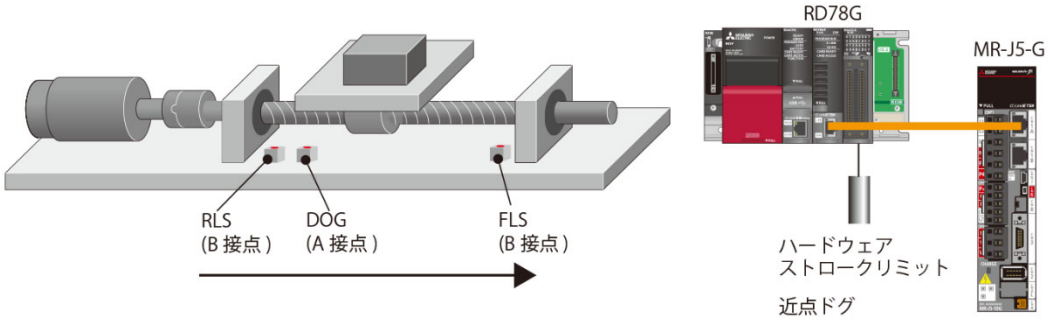
[STプログラム]

(c) 動作確認

プログラムを書き込み，指定した原点復帰方式で原点復帰が正しく行なわれるか確認してください。

## 付2.2　シーケンサCPUの入力信号を使用する場合

ハードウェアストロークリミット信号と近点ドグ信号に，シーケンサCPUの入力信号を使用する場合について説明します。




### (1) ハードウェアストロークリミット

ハードウェアストロークリミットの配線は，負論理(b接点)を推奨します。

**[注意事項]**
**ハードウェアストロークリミットの配線に正論理(a接点)を使用すると，断線やセンサ故障時に重大な事故が発生する可能性があります。**

ここでは，X20をFLS，X21をRLSとして使用する場合について説明します。

(a) センサの設置場所

位置決めアドレス増加側に上限ストロークリミット(FLS)，位置決めアドレス減少側に下限ストロークリミット(RLS)を設置してください。

(b) サーボパラメータ

以下のサーボパラメータを下表のとおりに設定してください。

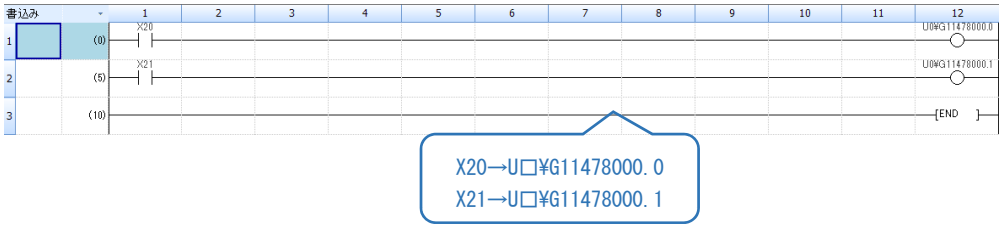
| 番　号 | 項　目 | 設定値 |
|---|---|---|
| Pr. PD41. 2 | リミットスイッチ有効状態選択 | 1：原点復帰モードのみ有効 |
| Pr. PD41. 3 | センサ入力方式選択 | 1：コントローラより入力 |

入出力　選択項目書込(I)　軸書込(S)

| No. | 略称 | 名称 | 単位 | 設定範囲 | 局1 |
|---|---|---|---|---|---|
| デジタル入出力 | | | | | |
| デバイス設定 | | | | | 設定 |
| PD03.0-1 | * | デバイス選択DI1 | | 00-7F | 0A |
| PD04.0-1 | * | デバイス選択DI2 | | 00-7F | 0B |
| PD05.0-1 | * | デバイス選択DI3 | | 00-7F | 22 |
| PD51.0-1 | * | デバイス選択DI3-2 | | 00-7F | 62 |
| PD38.0-1 | * | デバイス選択DI4 | | 00-7F | 2C |
| PD39.0-1 | * | デバイス選択DI5 | | 00-7F | 2D |
| PD07.0-1 | * | デバイス選択DO1 | | 00-7F | 05 |
| PD08.0-1 | * | デバイス選択DO2 | | 00-7F | 04 |
| PD09.0-1 | * | デバイス選択DO3 | | 00-7F | 03 |
| デバイス割付 | | | | | 設定 |
| PD01.0-7 | *DIA1 | 入力信号自動オン選択1 | | 00000000-00000FF0 | 00000000 |
| 入力フィルタ | | | | | |
| PD11.0 | * | 入力信号フィルタ選択 | | 0-B | 7：3.500ms |
| ALM出力 | | | | | |
| PD14.1 | * | 警告発生時の出力デバイスの選択 | | 0-1 | 0：警告時にWNGがオン |
| アナログ出力 | | | | | |
| アナログモニタ | | | | | |
| PC09.0-1 | | アナログモニタ1出力選択 | | 00-1F | 00：サーボモータ速度(± |
| PC11 | MO1 | アナログモニタ1 オフセット | | -999-999 | 0 |
| PC10.0-1 | | アナログモニタ2出力選択 | | 00-1F | 01：トルクまたは推力(±8 |
| PC12 | MO2 | アナログモニタ2 オフセット | | -999-999 | 0 |
| ストロークリミット機能 | | | | | |
| ストロークリミット機能 | | | | | |
| PC19.0 | * | [AL. 099 ストロークリミット警告] 選択 | | 0-1 | 0：有効 |
| PD41.2 | * | リミットスイッチ有効状態選択 | | 0-1 | 1：原点復帰モードのみ |
| PD41.3 | * | センサ入力方式選択 | | 0-1 | 1：コントローラより入力 |

(c) シーケンサCPUのプログラム

X20，X21のON/OFF状態を公開ラベル，またはバッファメモリにコピーしてモーションユニットに伝達します。以下は，バッファメモリにコピーする場合のプログラムです。

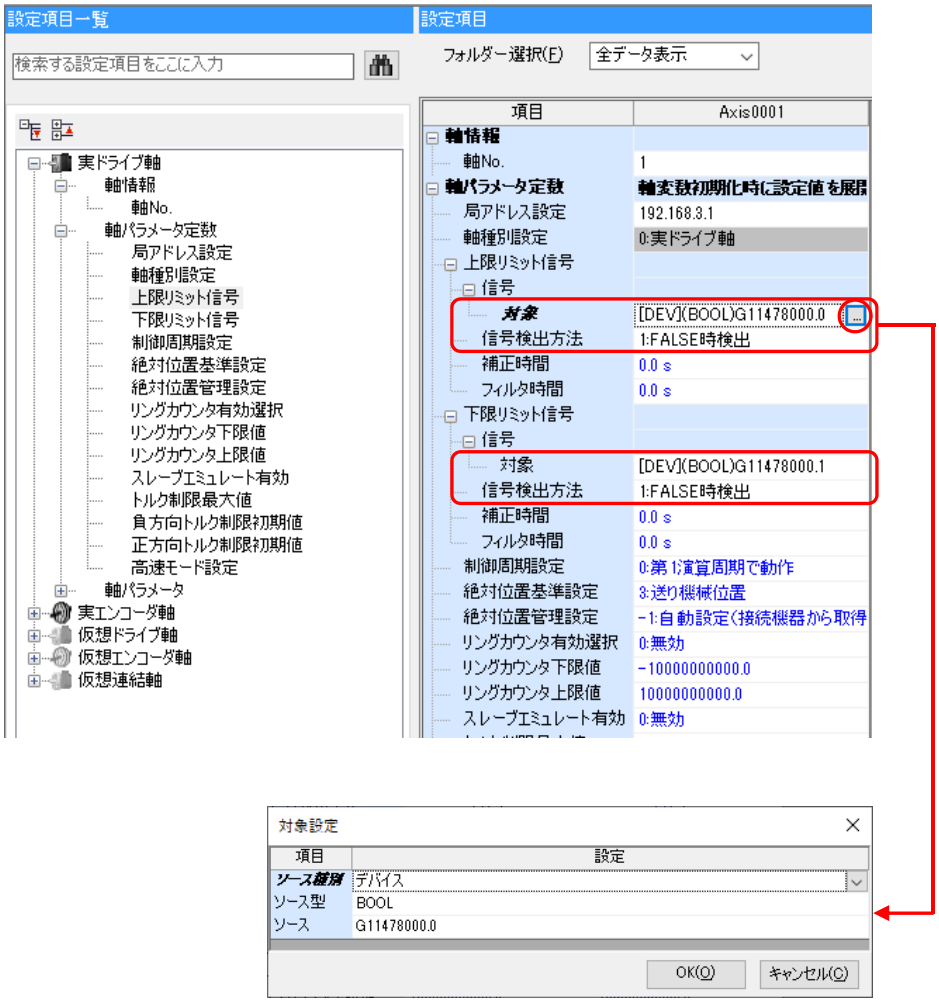



**[ポイント]**

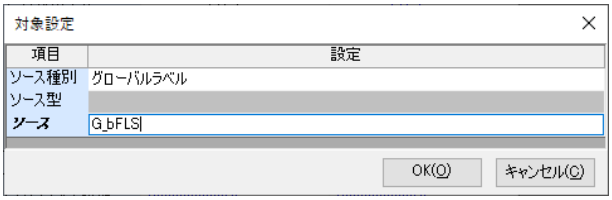
バッファメモリのユーザ自由エリアは，11478000～11997999です。

(d) モーションユニットのパラメータ

“軸パラメータ設定”ウィンドウで，“上限リミット信号”，および“下限リミット信号”の対象を以下のとおり設定します。




(注) 公開ラベルを使用する場合は，以下のとおり設定します。




上記と同様の設定を，ハードウェアストロークリミット信号を使用する全ての軸に対して行ないます。また，b接点を使用するため，信号検出方法を“1：FALSE時検出”に変更します。

(e) 動作確認

プログラムを書き込み，軸モニタの“上限リミット信号状態”，“下限リミット信号状態”で，それぞれの信号のON/OFFがモニタできるか確認してください。

リミット範囲内に復帰させる場合は，一度エラーリセットを実行してから，範囲内方向にJOG運転などで復帰させてください。

### (2) DOG信号を使用した原点復帰

ここでは，X22をDOG信号として使用します。

(a) サーボパラメータ

原点復帰方式と近点ドグ信号の極性を次のパラメータで設定します。

| 番　号 | 項　目 |
|---|---|
| Pr.PT29.0 | デバイス入力極性1 |
| Pr.PT45 | 原点復帰方式 |

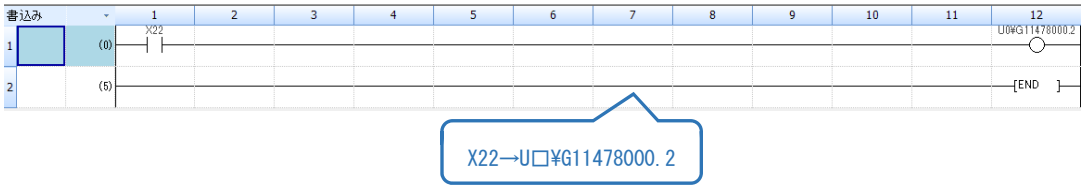
原点復帰方式に応じて，以下から必要なパラメータを設定します。
（設定が必要なパラメータは，原点復帰方式によって変わります。）

| 番　号 | 項　目 |
|---|---|
| Pr.PT05 | 原点復帰速度 |
| Pr.PT06 | クリープ速度 |
| Pr.PT07 | 原点シフト量 |
| Pr.PT08 | 原点復帰位置データ |
| Pr.PT09 | 近点ドグ後移動量 |
| Pr.PT55.0 | 原点復帰時減速時定数選択 |
| Pr.PT56 | 原点復帰加速時定数 |
| Pr.PT57 | 原点復帰減速時定数 |

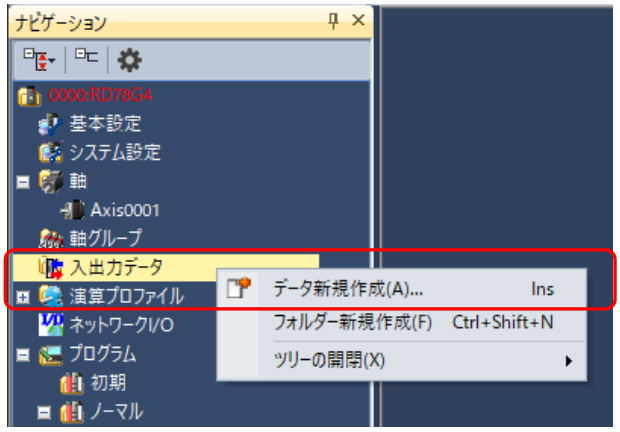
(b) モーションユニットでプログラムする場合

1) シーケンサCPUのプログラム

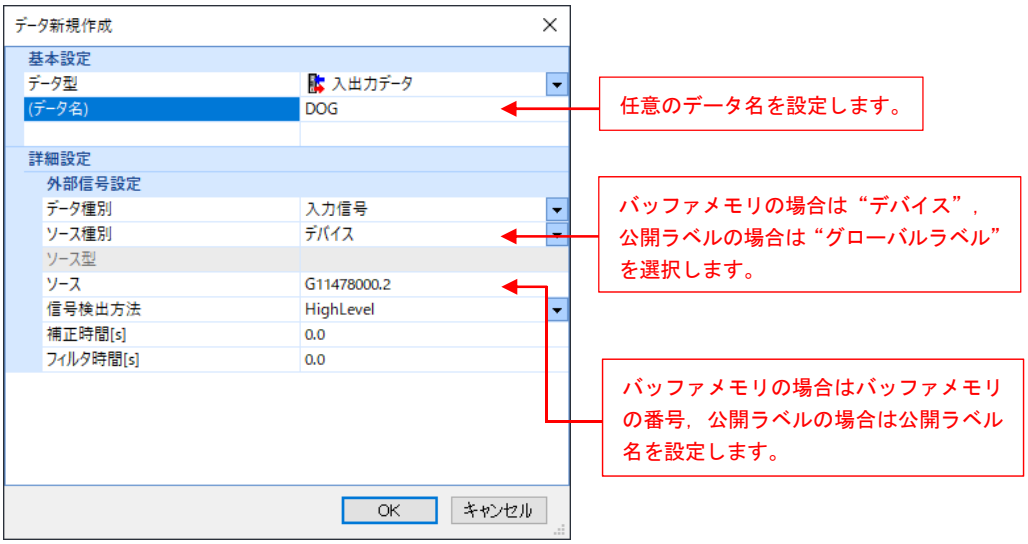
X22のON/OFF状態を公開ラベル，またはバッファメモリにコピーしてモーションユニットに伝達します。以下は，バッファメモリにコピーする場合のプログラムです。



2) モーションユニットのパラメータ
モーション制御設定機能で，ナビゲーションウィンドウの“入出力データ”を右クリックし，[データ新規作成]を選択します。




各項目を以下のとおり設定します。




3) モーションユニットのプログラム
MC_HomeのAbsSwitch入力に，2)で設定した入出力データ名を入力します。

```
//Homing
MC_Home_1(
    Axis:= Axis0001.AxisRef,
    Execute:= bHoming ,
    Position:= 0.0 ,
    AbsSwitch:= DOG ,
    Options:= 0 ,
    Done=> bHomeDone ,
    Error=> bHomeError
);
```

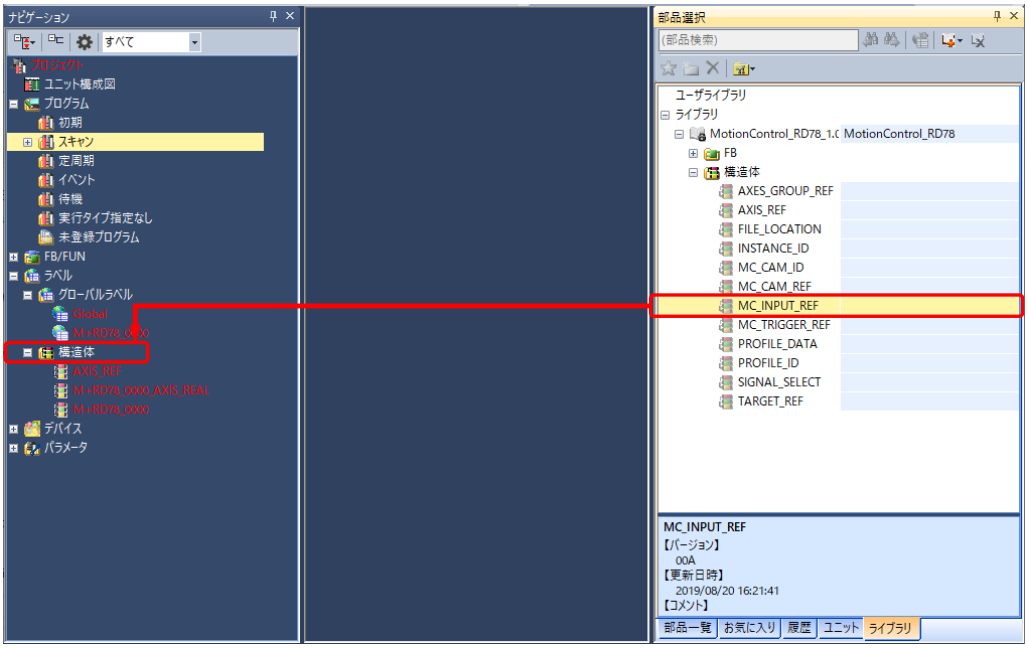
入出力データ名を入力します。

(c) シーケンサCPUでプログラムする場合

①シーケンサCPU側にMC_INPUT_REF型の構造体を定義する必要があります。

MELSOFT GX Works3の部品選択ウィンドウの[ライブラリ]タブから，“ライブラリ”⇒“MotionControl_RD78_****”⇒“構造体”をクリックします。

“MC_INPUT_REF”を選択し，ナビゲーションウィンドウの“ラベル”⇒“構造体”にドラッグ＆ドロップします。

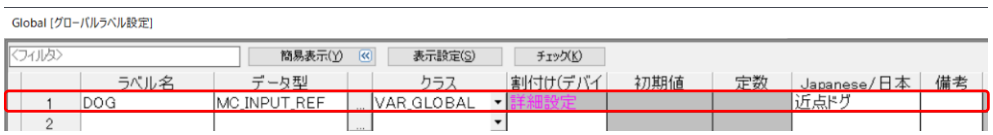



②“MC_INPUT_REF”とそのメンバの構造体“TARGET_REF”，“SIGNAL_SELECT”が登録されます。

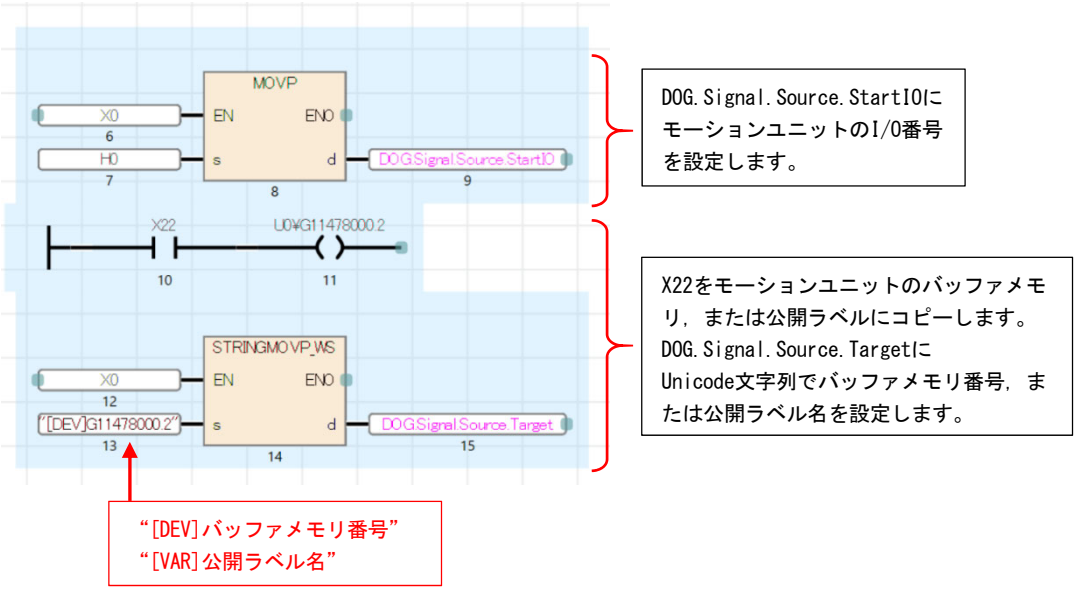
③シーケンサCPUのラベルにMC_INPUT_REF型の構造体を準備します。

ラベルの登録先（グローバルラベル または ローカルラベル）は任意です。

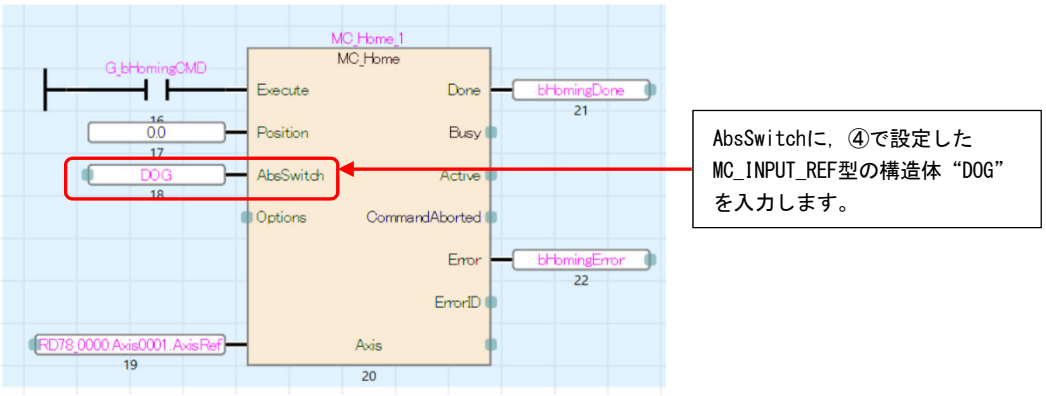
ここではラベル名をDOGとしています。



④MC_INPUT_REF型構造体のメンバをプログラムで設定します。




⑤MC_Homeをプログラムに記述します。

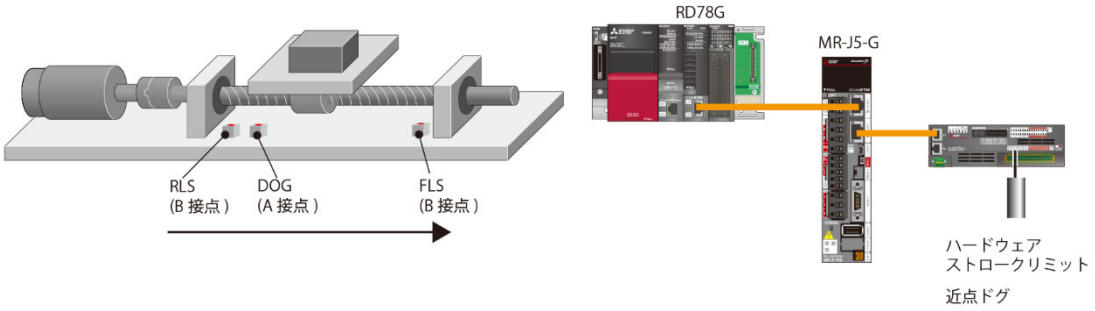



(d) 動作確認

プログラムを書き込み，指定した原点復帰方式で原点復帰が正しく行なわれるか確認してください。

## 付2.3　リモート入力ユニットの入力信号を使用する場合

ハードウェアストロークリミット信号と近点ドグ信号に，モーション管理局のリモート入力ユニットを使用する場合について説明します。




### (1) ハードウェアストロークリミット

ハードウェアストロークリミットの配線は，負論理(b接点)を推奨します。

**[注意事項]**
**ハードウェアストロークリミットの配線に正論理(a接点)を使用すると，断線やセンサ故障時に重大な事故が発生する可能性があります。**

ここでは，RX0をFLS，RX1をRLSとして使用する場合について説明します。

(a) センサの設置場所

位置決めアドレス増加側に上限ストロークリミット(FLS)，位置決めアドレス減少側に下限ストロークリミット(RLS)を設置してください。

(b) サーボパラメータ

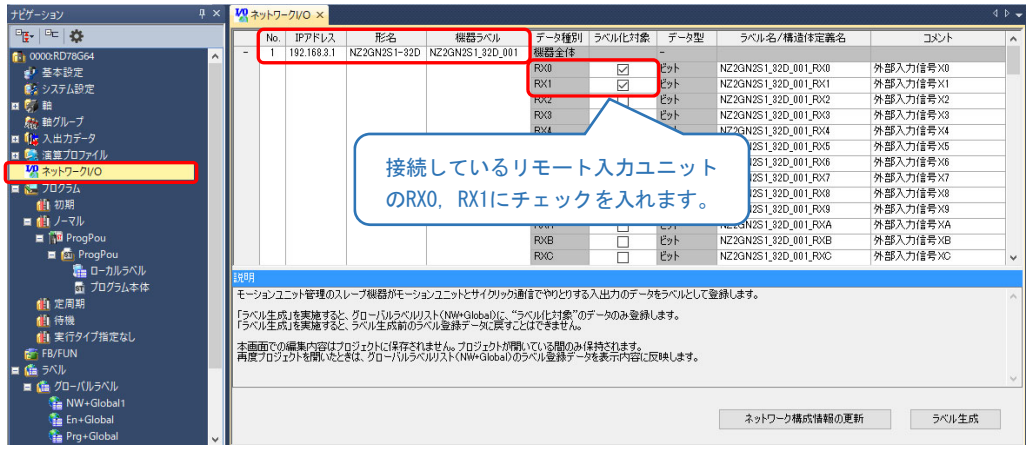
以下のサーボパラメータを下表のとおりに設定してください。

| 番　号 | 項　目 | 設定値 |
|---|---|---|
| Pr. PD41. 2 | リミットスイッチ有効状態選択 | 1：原点復帰モードのみ有効 |
| Pr. PD41. 3 | センサ入力方式選択 | 1：コントローラより入力 |

(c) モーションユニットのパラメータ

①ネットワークI/Oの設定

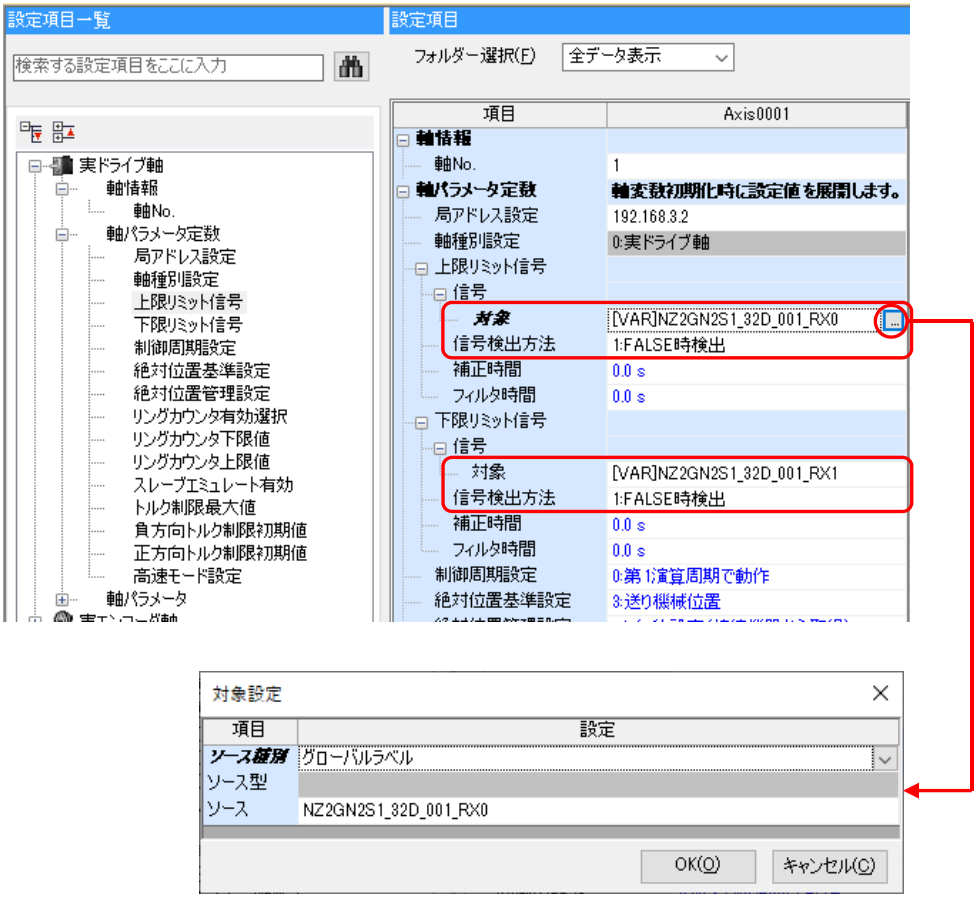
モーション制御設定機能で，ナビゲーションウィンドウの“ネットワークI/O”をダブルクリックします。この画面で，リモート入力ユニットの入力信号をラベル化します。RX0とRX1にチェックを入れ，[ラベル生成]ボタンをクリックします。




ラベル生成後は，ナビゲーションウィンドウの“グローバルラベル”に“NW+Global1”グループが追加されます。

②軸パラメータ

“軸パラメータ設定”ウィンドウで，“上限リミット信号”，および“下限リミット信号”の“対象”を以下のとおり設定します。




上記と同様の設定を，ハードウェアストロークリミット信号を使用する全ての軸に対して行ないます。また，b接点を使用するため，信号検出方法を“1：FALSE時検出”に変更します。

(d) 動作確認

プログラムを書き込み，軸モニタの“上限リミット信号状態”，“下限リミット信号状態”で，それぞれの信号のON/OFFがモニタできるか確認してください。

リミット範囲内に復帰させる場合は，一度エラーリセットを実行してから，範囲内方向にJOG運転などで復帰させてください。

### (2) DOG信号を使用した原点復帰

ここでは，RX2をDOG信号として使用します。

(a) サーボパラメータ

原点復帰方式と近点ドグ信号の極性を次のパラメータで設定します。

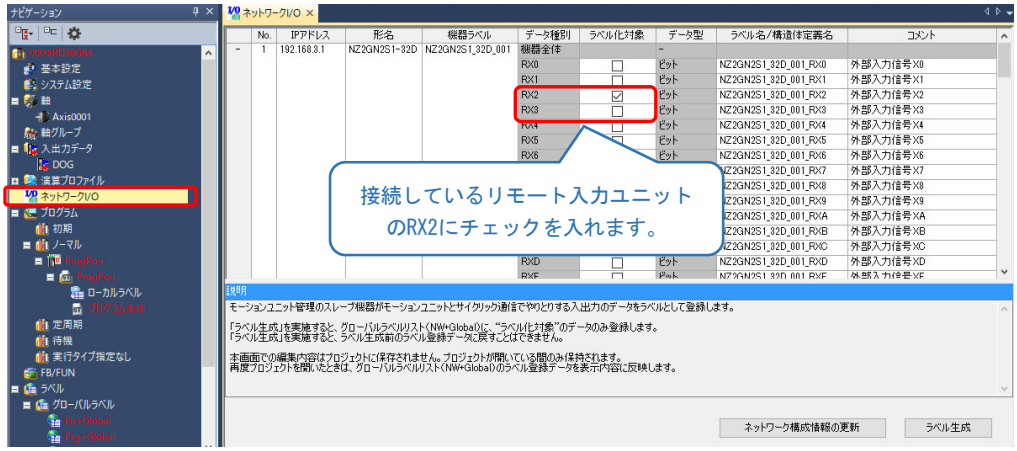
| 番　号 | 項　目 |
|---|---|
| Pr. PT29. 0 | デバイス入力極性 1 |
| Pr. PT45 | 原点復帰方式 |

原点復帰方式に応じて，以下から必要なパラメータを設定します。
（設定が必要なパラメータは，原点復帰方式によって変わります。）

| 番　号 | 項　目 |
|---|---|
| Pr. PT05 | 原点復帰速度 |
| Pr. PT06 | クリープ速度 |
| Pr. PT07 | 原点シフト量 |
| Pr. PT08 | 原点復帰位置データ |
| Pr. PT09 | 近点ドグ後移動量 |
| Pr. PT55. 0 | 原点復帰時減速時定数選択 |
| Pr. PT56 | 原点復帰加速時定数 |
| Pr. PT57 | 原点復帰減速時定数 |

(b) モーションユニットのパラメータ（ネットワークI/0の設定）

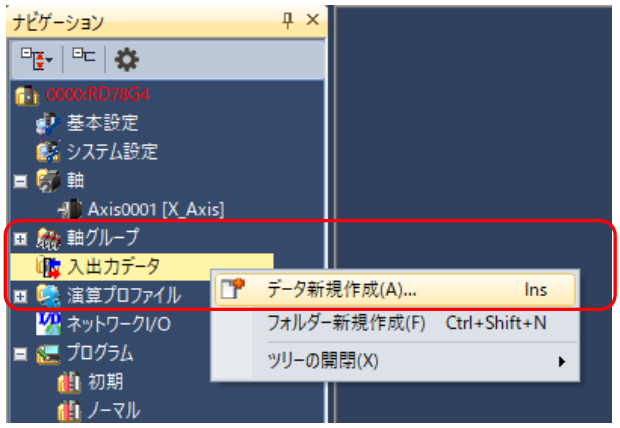
モーション制御設定機能で，ナビゲーションウィンドウの“ネットワークI/0”をダブルクリックします。この画面で，リモート入力ユニットの入力信号をラベル化します。
RX2にチェックを入れ，[ラベル生成]ボタンをクリックします。



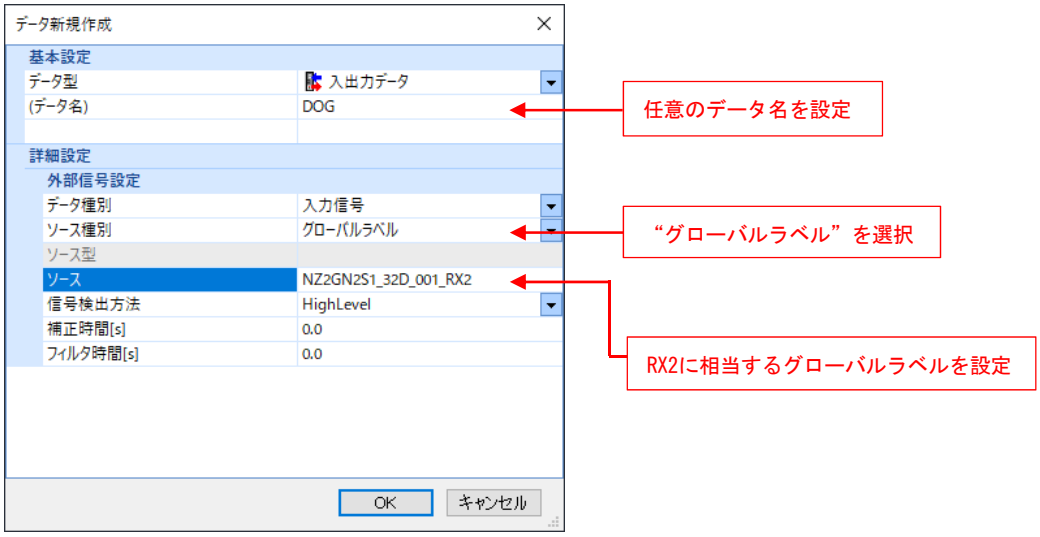

ラベル生成後は，ナビゲーションウィンドウの“グローバルラベル”に“NW+Global1”グループが追加されます。

(c) モーションユニットでプログラムする場合

①モーション制御設定機能で，ナビゲーションウィンドウの“入出力データ”を右クリックし，[データ新規作成]画面を選択します。




各項目を以下のとおり設定します。




②MC_HomeのAbsSwitch入力に，(2)で設定した入出力データ名を入力します。

```
//Homing
MC_Home_1(
    Axis:= Axis0001.AxisRef,
    Execute:= bHoming ,
    Position:= 0.0 ,
    AbsSwitch:= DOG ,
    Options:= 0 ,
    Done=> bHomeDone ,
    Error=> bHomeError
);
```

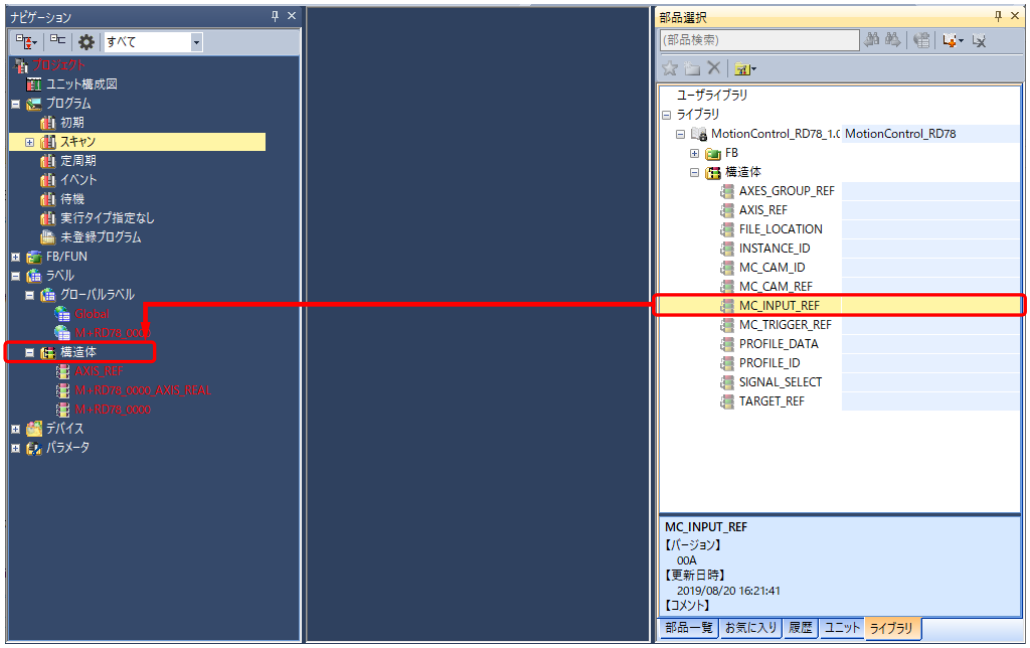
入出力データ名を入力します。

(d) シーケンサCPUでプログラムする場合

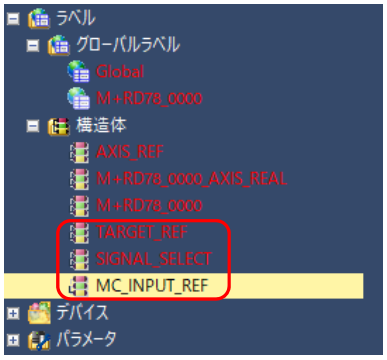
①シーケンサCPU側にMC_INPUT_REF型の構造体を定義する必要があります。

MELSOFT GX Works3の部品選択ウィンドウの[ライブラリ]タブから，“ライブラリ”⇒“MotionControl_RD78_****”⇒“構造体”を選択します。

“MC_INPUT_REF”を選択し，ナビゲーションウィンドウの“ラベル”⇒“構造体”にドラッグ＆ドロップします。



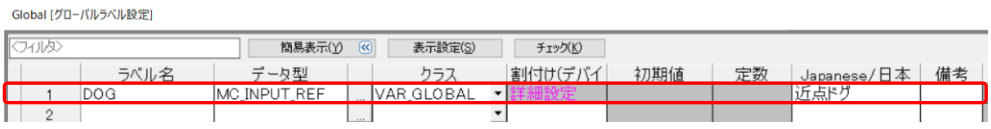

②“MC_INPUT_REF”とそのメンバの構造体“TARGET_REF”，“SIGNAL_SELECT”が登録されます。




③シーケンサCPUのラベルにMC_INPUT_REF型の構造体を準備します。

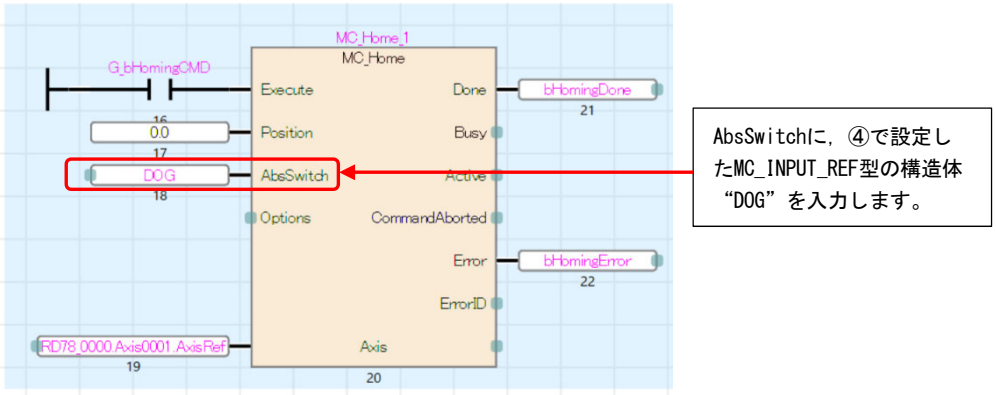
ラベルの登録先（グローバルラベル または ローカルラベル）は任意です。

ここではラベル名をDOGとしています。



④MC_INPUT_REF型構造体のメンバをプログラムで設定します。

⑤MC_Homeをプログラムに記述します。




(e) 動作確認

プログラムを書き込み，指定した原点復帰方式で原点復帰が正しく行なわれるか確認してください。

## 付3 絶対位置検出システムで使用する場合の注意事項

絶対位置検出システムで使用する場合，以下の項目に注意してください。

### (1) サーボパラメータの設定

サーボパラメータは，以下の2点を変更してください。

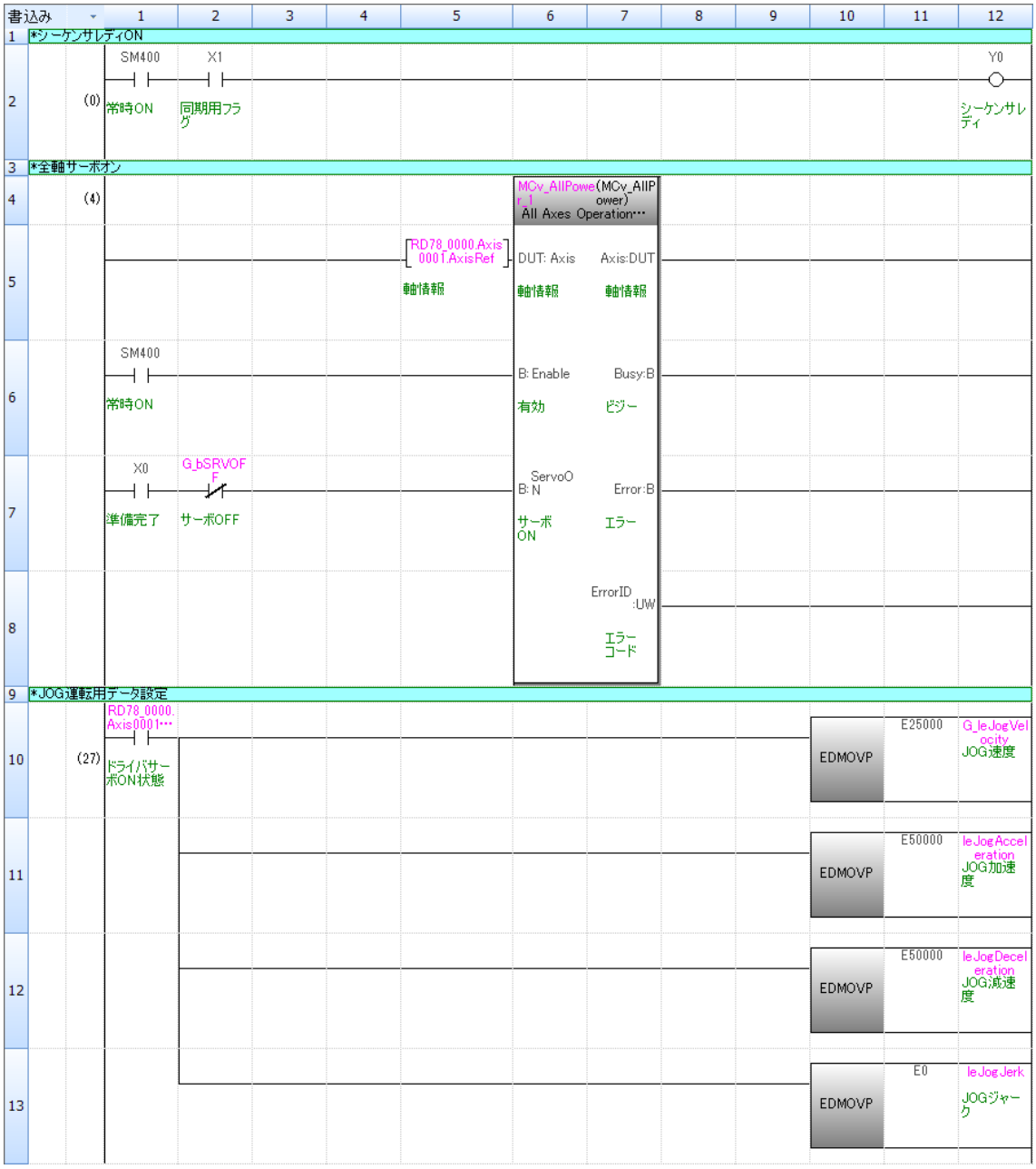
| 番 号 | 項 目 | 設定値 |
|---|---|---|
| PA03.0 | 絶対位置検出システム選択 | 0：無効 → 1：有効 |
| PC29.5 | [AL.0E3 絶対位置カウンタ警告] 選択 | 1：有効 → 0：無効 |

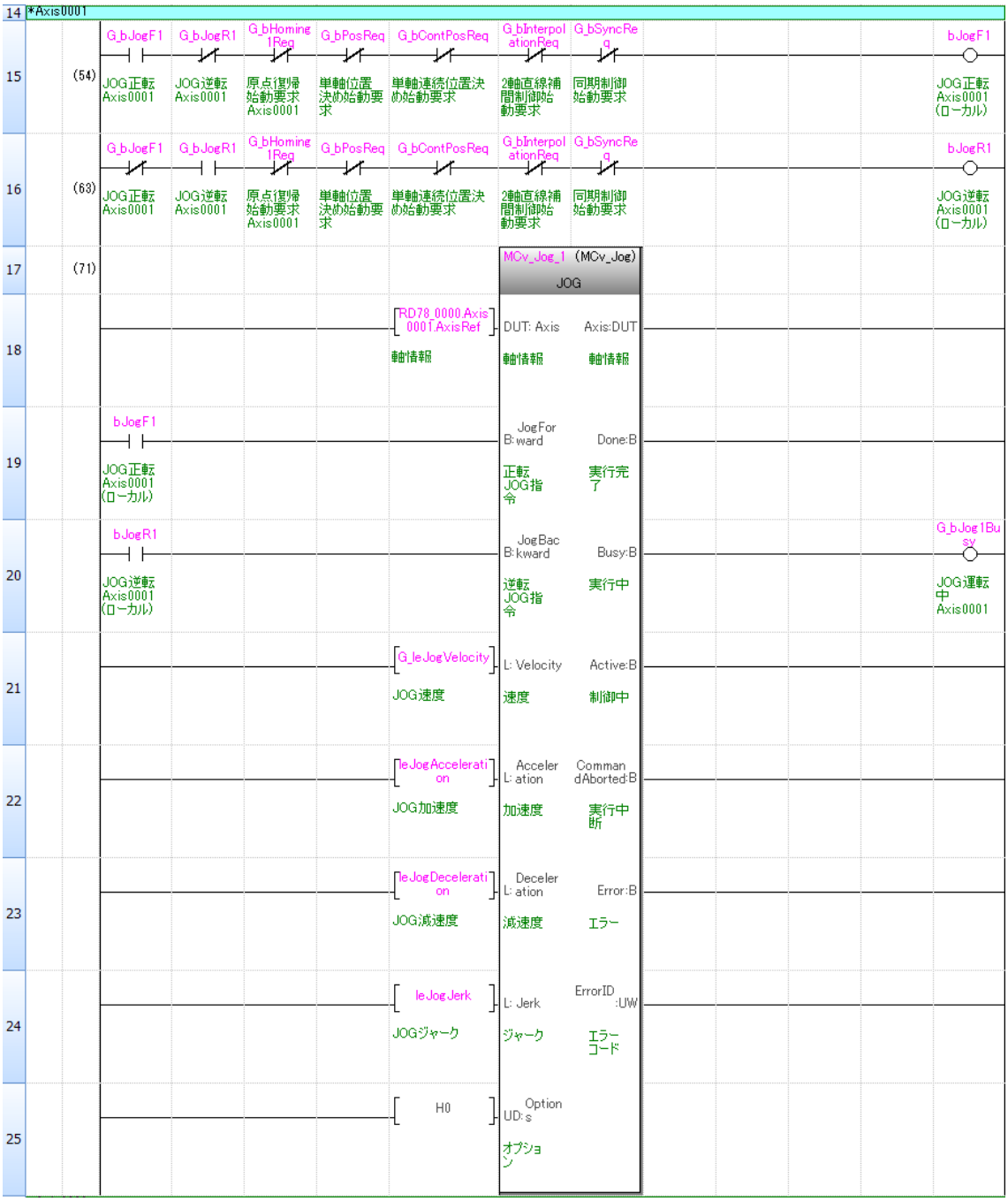
### (2) 初回起動時

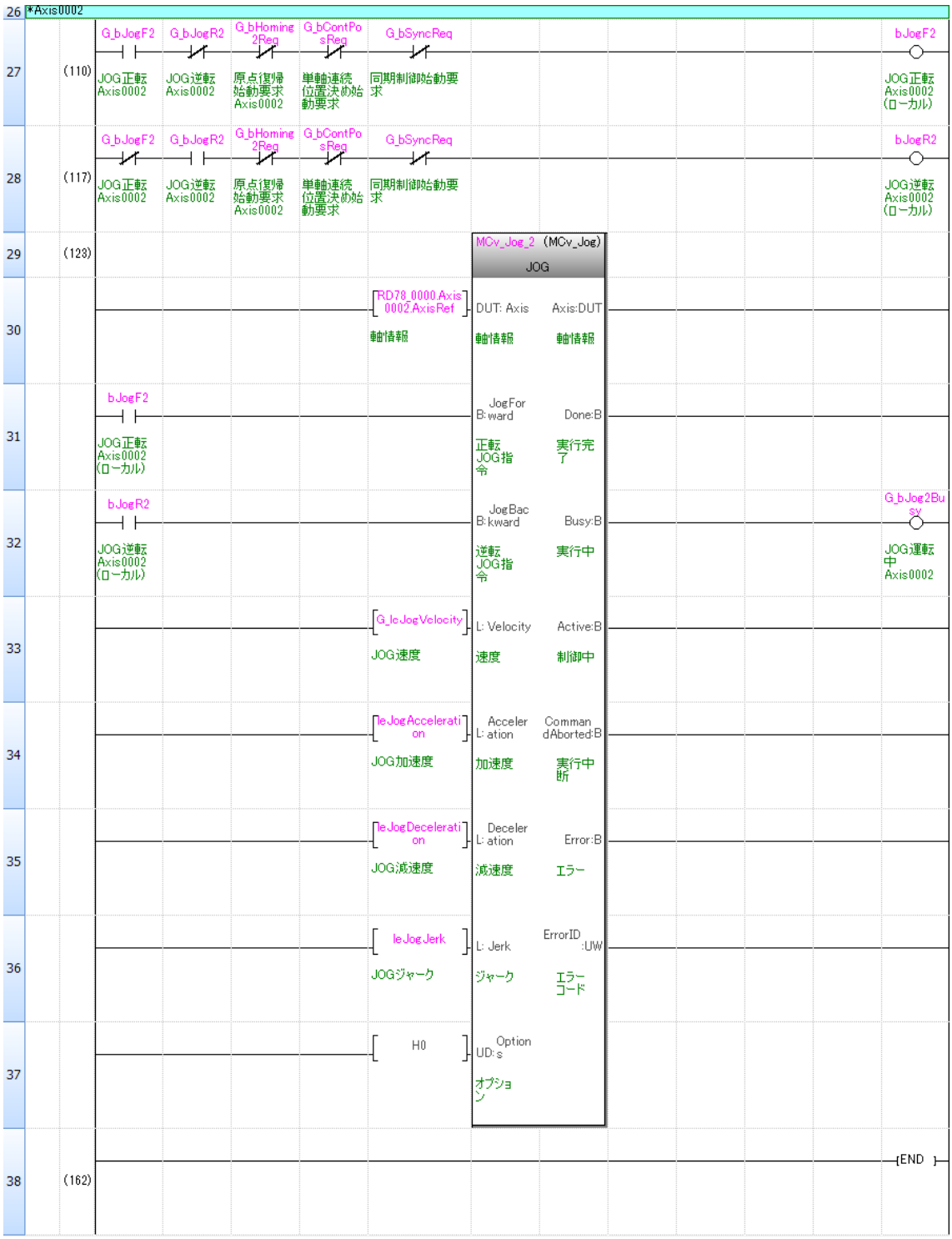
絶対位置検出システムに設定後，最初の電源投入時にサーボアンプの[AL.025.1 サーボモータエンコーダ 絶対位置消失]が発生します。
アラームが発生している状態で，5秒間放置してから電源を再投入してください。
その後，原点復帰を実施してください。

## 付4　シーケンスプログラムで制御する場合

第4章のサンプルプログラムを，FBDではなく，シーケンスプログラムで制御する場合，以下を[ServoON_Jog]のプログラム例として示します。




14 *Axis0001
15 (54)
G_bJogF1
JOG正転 Axis0001
G_bJogR1
JOG逆転 Axis0001
G_bHoming 1Req
原点復帰 始動要求 Axis0001
G_bPosReq
単軸位置 決め始動要求
G_bContPosReq
単軸連続位置決め始動要求
G_bInterpolationReq
2軸直線補間制御始動要求
G_bSyncReq
同期制御始動要求
bJogF1
JOG正転 Axis0001 (ローカル)
16 (63)
bJogR1
JOG逆転 Axis0001 (ローカル)
17 (71)
MCv_Jog_1 (MCv_Jog)
JOG
18
RD78_0000.Axis 0001.AxisRef
軸情報
DUT: Axis
Axis:DUT
19
JogFor B:ward
Done:B
正転 JOG指令
実行完了
20
JogBac B:kward
Busy:B
逆転 JOG指令
実行中
G_bJog1Busy
JOG運転中 Axis0001
21
G_leJogVelocity
JOG速度
L: Velocity
Active:B
速度
制御中
22
leJogAcceleration
JOG加速度
L: ation
Acceler
CommandAborted:B
加速度
実行中断
23
leJogDeceleration
JOG減速度
Deceler
Error:B
減速度
エラー
24
leJogJerk
JOGジャーク
L: Jerk
ErrorID :UW
ジャーク
エラーコード
25
H0
UD: s
Option
オプション


*Axis0002
G_bJogF2
JOG正転 Axis0002
G_bJogR2
JOG逆転 Axis0002
G_bHoming 2Req
原点復帰 始動要求 Axis0002
G_bContPo sReq
単軸連続 位置決め始 動要求
G_bSyncReq
同期制御始動要求
bJogF2
JOG正転 Axis0002 (ローカル)
bJogR2
JOG逆転 Axis0002 (ローカル)
MCv_Jog_2 (MCv_Jog)
JOG
RD78_0000.Axis 0002.AxisRef
軸情報
DUT: Axis
Axis:DUT
JogFor B:ward
Done:B
正転 JOG指令
実行完了
JogBac B:kward
Busy:B
逆転 JOG指令
実行中
G_bJog2Bu sy
JOG運転中 Axis0002
G_leJogVelocity
JOG速度
L: Velocity
Active:B
速度
制御中
leJogAccelerati on
JOG加速度
Acceler L: ation
Comman dAborted:B
加速度
実行中断
leJogDecelerati on
JOG減速度
Deceler L: ation
Error:B
減速度
エラー
leJogJerk
JOGジャーク
L: Jerk
ErrorID :UW
ジャーク
エラーコード
H0
Option UD: s
オプション
END

# メ　モ

# 三菱電機サーボシステム モーションユニットクイックスタートガイド


## 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)5812-1430 |
| 北海道支社 | 〒060-8693 | 札幌市中央区北二条西4-1(北海道ビル) (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 |
| 関越支社 | 〒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1-4-1(マルタケビル) (025)241-7227 |
| 神奈川支社 | 〒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)224-2623 |
| 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)565-3326 |
| 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)6486-4120 |
| 中国支社 | 〒730-8657 | 広島市中区中町7-32(ニッセイ広島ビル) (082)248-5445 |
| 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)721-2251 |


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。

### 電話技術相談窓口　受付時間※1　月曜～金曜 9:00～19:00、土曜・日曜・祝日 9:00～17:00

| 対象機種 | | | 電話番号 |
|---|---|---|---|
| 自動窓口案内 | | | 052-712-2444 |
| エッジコンピューティング製品 | | 産業用PC MELIPC | 052-712-2370※2 |
| | | Edgecross対応ソフトウェア(MTConnectデータコレクタを除く) | |
| シーケンサ | MELSEC iQ-R/Q/L/QnAS/AnSシーケンサ(CPU内蔵Ethernet機能などネットワークを除く) | | 052-711-5111 |
| | MELSEC iQ-F/FXシーケンサ全般 | | 052-725-2271※3 |
| | ネットワークユニット(CC-Linkファミリー/MELSECNET/Ethernet/シリアル通信) | | 052-712-2578 |
| | MELSOFTシーケンサエンジニアリングソフトウェア | MELSOFT GXシリーズ(MELSEC iQ-R/Q/L/QnAS/AnS) | 052-711-0037 |
| | MELSOFT統合エンジニアリング環境 | MELSOFT Navigator | 052-799-3591※2 |
| | iQ Sensor Solution | | |
| | MELSOFT通信支援ソフトウェアツール | MELSOFT MXシリーズ | 052-712-2370※2 |
| | MELSECパソコンボード | Q80BDシリーズなど | |
| | C言語コントローラ/C言語インテリジェント機能ユニット | | |
| | MESインタフェースユニット/高速データロガーユニット | | 052-799-3592※2 |
| | システムレコーダ | | |
| | MELSEC計装/iQ-R/Q二重化 | プロセスCPU/二重化機能SIL2プロセスCPU(MELSEC iQ-Rシリーズ) | 052-712-2830※2※3 |
| | | プロセスCPU/二重化CPU(MELSEC-Qシリーズ) | |
| | | MELSOFT PXシリーズ | |
| | MELSEC Safety | 安全シーケンサ(MELSEC iQ-R/QSシリーズ) | 052-712-3079※2※3 |
| | | 安全コントローラ(MELSEC-WSシリーズ) | |
| | 電力計測ユニット/絶縁監視ユニット | QEシリーズ/REシリーズ | 052-719-4557※2※3 |
| FAセンサ MELSENSOR | | レーザ変位センサ | 052-799-9495※2 |
| | | ビジョンセンサ | |
| | | コードリーダ | |
| 表示器 GOT | | GOT2000/1000シリーズなど | 052-712-2417 |
| | | MELSOFT GTシリーズ | |
| SCADA MC Works64 | | | 052-712-2962※2※6 |

| 対象機種 | | 電話番号 |
|---|---|---|
| サーボ/位置決めユニット/モーションユニット/シンプルモーションユニット/モーションコントローラ/センシングユニット/組込み型サーボシステムコントローラ | MELSERVOシリーズ | 052-712-6607 |
| | 位置決めユニット(MELSEC iQ-R/Q/L/AnSシリーズ) | |
| | モーションユニット(MELSEC iQ-Rシリーズ) | |
| | シンプルモーションユニット(MELSEC iQ-R/iQ-F/Q/Lシリーズ) | |
| | モーションCPU(MELSEC iQ-R/Q/AnSシリーズ) | |
| | センシングユニット(MR-MTシリーズ) | |
| | シンプルモーションボード/ポジションボード | |
| | MELSOFT MTシリーズ/MRシリーズ/EMシリーズ | |
| センサレスサーボ | FR-E700EX/MM-GKR | 052-722-2182 |
| インバータ | FREQROLシリーズ | 052-722-2182 |
| 三相モータ | 三相モータ225フレーム以下 | 0536-25-0900※2※4 |
| 産業用ロボット | MELFAシリーズ | 052-721-0100 |
| 電磁クラッチ・ブレーキ/テンションコントローラ | | 052-712-5430※5 |
| データ収集アナライザ | MELQIC IU1/IU2シリーズ | 052-712-5440※5 |
| 低圧開閉器 | MS-Tシリーズ/MS-Nシリーズ | 052-719-4170 |
| | US-Nシリーズ | |
| 低圧遮断器 | ノーヒューズ遮断器/漏電遮断器/MDUブレーカ/気中遮断器(ACB)など | 052-719-4559 |
| 電力管理用計器 | 電力量計/計器用変成器/指示電気計器/管理用計器/タイムスイッチ | 052-719-4556 |
| 省エネ支援機器 | EcoServer/E-Energy/検針システム/エネルギー計測ユニット/B/NETなど | 052-719-4557※2※3 |
| 小容量UPS(5kVA以下) | FW-Sシリーズ/FW-Vシリーズ/FW-Aシリーズ/FW-Fシリーズ | 052-799-9489※2※6 |

お問い合わせの際には、今一度電話番号をお確かめの上、お掛け間違いのないようお願い致します。
※1:春季・夏季・年末年始の休日を除く　※2:土曜・日曜・祝日を除く　※3:金曜は17:00まで　※4:月曜～木曜の9:00～17:00と金曜の9:00～16:30
※5:受付時間9:00～17:00　(土曜・日曜・祝日・当社休日を除く)　※6:月曜～金曜の9:00～17:00

### FAX技術相談窓口　受付時間　月曜～金曜　9:00～16:00(祝日・当社休日を除く)

| 対象機種 | FAX番号 |
|---|---|
| 電力計測ユニット/絶縁監視ユニット(QEシリーズ/REシリーズ) | 084-926-8340 |
| 三相モータ225フレーム以下 | 0536-25-1258※7 |
| 低圧開閉器 | 0574-61-1955 |
| 低圧遮断器 | 084-926-8280 |
| 電力管理用計器/省エネ支援機器/小容量UPS(5kVA以下) | 084-926-8340 |

三菱電機FAサイトの「仕様・機能に関するお問い合わせ」もご利用ください。
※7:月曜～木曜の9:00～17:00と金曜の9:00～16:30(祝日・当社休日を除く)


L(名)03190-A　2007〈IP〉

この印刷物は、2020年7月の発行です。なお、この印刷物に掲載した内容は、改善のために予告なく変更する場合がありますので、ご採用の節には、事前に弊社までお問い合わせください。

2020年7月作成