

Rexroth IndraLogic XLC IndraMotion MLC 13VRS 汎用アプリケーション テンプレート

R911341767

第02版

アプリケーションマニュアル



表題 Rexroth IndraLogic XLC
IndraMotion MLC 13VRS
汎用アプリケーション テンプレート

ドキュメンテーション・タイプ アプリケーションマニュアル

ドキュメンテーション・タイプコード DOK-XLCMLC-TF*GAT**V13-AP02-JA-P

内部のファイルコード RS-cdff34e18f0001620a6846a000296caa-2-ja-JP-11

ドキュメンテーションの目的 本書は、IndraLogic PLC のプログラマーに構造化されたテンプレートを提供します。このテンプレートは PLC プログラム コードの追加および編集に使用できます。本書にはテンプレート、ウィザードおよびサンプル アプリケーションが含まれています。

変更の履歴

版コード	発行日	備考
第 01 版	04-2012	第 1 版
第 02 版	08-2014	修正実施済み

著作権 © Bosch Rexroth AG 2014

すべての権利は、知的所有権申請の場合も含めて、Bosch Rexroth AG に帰属します。複写権や配布権など、裁量権限 はすべて当社に帰属します。

免責条項 特性データは製品の説明のみを目的としており契約上明白に規定されていない限り特性を保証するものではありません。当文献の内容と製品の有効性に関連する無断転載を禁じます。

発行元 Bosch Rexroth AG
Bgm.-Dr.-Nebel-Str. 2 ■ 97816 Lohr a. Main, Germany
TEL +49 93 52 18 0 ■ FAX +49 9352 18 8400
<http://www.boschrexroth.com/>
Gac2Lo

注 本書は、無塩素漂白紙に印刷されています。

目次

1	この資料について	9
1.1	資料の有効性	9
1.2	資料の構成	9
1.3	必須資料および補足資料	10
1.4	情報の表記方法	15
1.4.1	安全上の指示	15
1.4.2	使用されている記号	16
1.4.3	用語および略語	16
2	使用上の重要な指示	19
2.1	使用目的	19
2.1.1	はじめに	19
2.1.2	使用範囲および用途	19
2.2	目的外の使用	20
3	電気ドライブおよび制御の安全の手引き	21
3.1	用語の定義	21
3.2	一般情報	22
3.2.1	安全上の指示の使用と、他者への引渡し	22
3.2.2	安全な使用のための要件	22
3.2.3	不適切な使用による危険	23
3.3	特定の危険に関する指示	24
3.3.1	電気部品および筐体への接触に対する保護	24
3.3.2	感電から保護する超低電圧	25
3.3.3	危険な動作に対する保護	25
3.3.4	運転および取り付け中の磁界および電磁界に対する保護	27
3.3.5	高温部品への接触に対する保護	27
3.3.6	取り扱いおよび取り付け中の保護	28
3.3.7	バッテリーの安全	28
3.3.8	加圧システムに対する保護	28
3.4	警告語および安全注意シンボルの説明	29
4	はじめに	31
5	アプリケーションの自動化	33
5.1	一般情報	33
5.2	初期化	33
5.3	運転モード	34
5.3.1	一般情報	34
5.3.2	自動	34
5.3.3	手動	34
5.4	周期的処理	34
5.4.1	一般情報	34

目次

ページ

5.4.2	運転モードから独立したロジック.....	34
5.4.3	診断メッセージの生成と表示.....	34
5.5	タスク管理.....	34
5.5.1	一般情報.....	34
5.5.2	タスク管理の原則.....	35
5.5.3	IndraWorks/IndraLogic 使用時の推奨タスク分割.....	35
5.5.4	サイクル時間の計測.....	37
5.5.5	アプリケーション固有のウォッチドッグ.....	39
5.6	エラー処理.....	40
5.6.1	一般情報.....	40
5.6.2	エラー処理の原則.....	40
5.6.3	エラー検出.....	40
5.6.4	エラーリアクション.....	41
5.6.5	エラー診断.....	42
5.6.6	エラー確認/エラーの後の再起動.....	43
5.7	表示.....	43
5.7.1	一般情報.....	43
5.7.2	表示の一般的な内容.....	43
5.7.3	PLC を使用したデータ提供.....	44
6	ML_TechInterface.library.....	49
6.1	導入と概要.....	49
6.2	使用開始時の TechInterface 設定.....	56
6.3	AxisInterface のファンクションブロック.....	59
6.3.1	MB_AxisInitType01.....	59
6.3.2	MB_AxisInterfaceType01.....	62
6.3.3	MB_AxisInterfaceType02.....	65
6.4	軸モード.....	67
6.4.1	ダイアログベースの軸コマンド.....	67
6.4.2	概要.....	69
6.4.3	Drive Ready (ドライブ準備完了).....	71
6.4.4	Drive Halt (ドライブ一時停止).....	72
6.4.5	Drive Stop (ドライブ停止).....	72
6.4.6	Drive Homing (ドライブ ホーミング).....	73
6.4.7	Absolute Positioning (絶対位置設定).....	73
6.4.8	Relative Positioning (相対位置設定).....	74
6.4.9	Additive Positioning (追加位置設定).....	75
6.4.10	Velocity Control (速度制御).....	75
6.4.11	運転モード "Phase Synchronization" (フェーズ同期).....	76
6.4.12	運転モード "Velocity Synchronization" (速度同期).....	77
6.4.13	運転モード "Electronic Cam" (電子カム).....	77
6.4.14	運転モード "Electronic MotionProfile".....	78
6.4.15	運転モード "Electronic FlexProfile".....	79
6.4.16	運転モード "Robot Control".....	80
6.4.17	運転モード "Drive-controlled Interruption" (ドライブ制御の中断).....	81
6.4.18	運転モード "External Function Block" (外部ファンクションブロック).....	82

目次

ページ

6.5	グローバル変数.....	82
6.6	最初の手順、AxisInterface.....	83
6.6.1	概要.....	83
6.6.2	AxisInterface 用プログラム.....	83
6.6.3	AxisInterface の表示.....	84
6.7	AxisInterface のユーザー拡張.....	97
6.7.1	概要.....	97
6.7.2	arAxisCtrl_gb[] 構造の拡張.....	99
6.7.3	arAxisStatus_gb[] 構造の拡張.....	100
6.7.4	ファンクションブロックの拡張.....	100
6.8	AxisInterface の構造.....	102
6.8.1	概要.....	102
6.8.2	MB_AXIS_STATUS_TYPE01 構造.....	103
6.8.3	MB_AXIS_CONTROL_TYPE01 構造.....	110
6.8.4	MB_AXIS_STATUS_TYPE02 構造.....	124
6.8.5	MB_AXIS_CONTROL_TYPE02 構造.....	125
6.9	IMC インターフェースの構造.....	126
6.9.1	概要.....	126
6.9.2	MB_IMC_Status_Type01.....	127
6.9.3	MB_IMC_Control_Type01.....	127
6.10	操作手順：一般的なユーザー アクティビティ.....	128
6.10.1	軸データへのアクセス.....	128
6.10.2	軸の追加.....	128
6.10.3	軸番号が 16 より大きい軸の最大数を大きくする.....	128
6.10.4	軸の削除/名前変更.....	128
6.10.5	タスクの作成.....	129
6.10.6	AxisInterface の拡張.....	129
6.10.7	Robot Control.....	130
7	キネマティクス インターフェース.....	133
7.1	はじめに.....	133
7.2	ML_KinTech.library.....	136
7.2.1	導入と概要.....	136
7.2.2	キネマティクス インターフェースの構造.....	138
7.2.3	キネマティクス インターフェースのファンクションブロック.....	162
7.2.4	運転モード.....	172
7.2.5	グローバル変数.....	183
7.2.6	最初の手順、キネマティクス インターフェース.....	184
7.2.7	キネマティクス インターフェースのユーザー拡張.....	195
7.3	ユース ケース.....	197
7.3.1	キネマティクス インターフェースによる逐次プログラミング.....	197
7.3.2	ユーザー定義変換の登録.....	199
8	GAT^{compact}.....	201
8.1	GAT ^{compact} の概要.....	201

目次

ページ

8.2	最初の手順 - GAT ^{compact}	201
8.2.1	単純なモーションアプリケーションの作成.....	201
8.3	初期化.....	208
8.4	運転モード.....	211
8.5	周期的処理.....	213
8.6	タスク処理.....	213
8.6.1	タスク設定、GAT ^{compact}	213
8.6.2	MotionTask の追加.....	214
8.6.3	BackgroundTask の追加.....	216
8.7	エラー処理.....	217
8.7.1	エラーの検出および診断.....	217
8.7.2	エラーリアクション.....	218
8.7.3	エラーのクリア.....	221
8.8	表示.....	221
8.8.1	起動中の支援のための内部表示.....	221
8.8.2	独立した HMI による外部表示 (WinStudio 表示).....	223
8.9	GAT ^{compact} のインターフェース.....	230
8.9.1	一般情報.....	230
8.9.2	機械状態のデータ.....	231
8.9.3	エラー処理用データ.....	231
8.9.4	軸データ.....	232
8.9.5	コントロール データ.....	232
8.10	操作手順：一般的なユーザー アクティビティ.....	232
8.10.1	軸データへのアクセス.....	232
8.10.2	軸の追加.....	233
8.10.3	軸番号が 16 より大きい軸の最大数を大きくする.....	234
8.10.4	軸の削除/名前変更.....	234
8.10.5	運転モードの追加.....	235
8.10.6	MotionTask での処理.....	236
8.10.7	AxisInterface の拡張.....	236
8.10.8	Robot Control.....	237
8.11	GAT ^{compact} の概略.....	240
8.11.1	構造の概要.....	240
8.11.2	機械の最も重要なグローバル変数の概要.....	240
8.11.3	軸の最も重要なグローバル変数の概要.....	241
8.11.4	機械のメソッドとアクションの概要.....	241
8.11.5	機械内の軸へのアクセス.....	241
9	sercos インターフェースなしの IndraLogic XLC 用 GAT ^{compact}	243
9.1	GAT ^{compact} の概要.....	243
9.2	GAT ^{compact} の最初の手順.....	243
9.3	初期化.....	244
9.4	運転モード.....	245
9.5	周期的処理.....	246
9.6	タスク処理.....	246

目次

ページ

9.6.1	GAT ^{compact} のタスク設定.....	246
9.6.2	BackgroundTask の追加.....	246
9.7	エラー処理.....	247
9.7.1	エラーの検出および診断.....	247
9.7.2	エラーリアクション.....	249
9.7.3	エラーのクリア	250
9.8	表示.....	250
9.8.1	起動中の支援のための内部表示.....	250
9.8.2	独立した HMI による外部表示 (WinStudio 表示).....	250
9.9	GAT ^{compact} のインターフェース	256
9.9.1	機械データ.....	256
9.9.2	エラー処理用データ.....	257
9.9.3	コントロール データ.....	258
9.10	操作手順：一般的なユーザー アクティビティ.....	258
9.10.1	運転モードの追加.....	258
9.11	GAT ^{compact} の概略.....	260
9.11.1	構造の概要.....	260
9.11.2	機械の最も重要なグローバル変数の概要.....	260
9.11.3	機械のメソッドとアクションの概要.....	260
10	集中状態の機械での GAT.....	263
10.1	一般情報.....	263
10.2	概要.....	263
10.2.1	基本概念.....	263
10.2.2	使用の要件.....	265
10.3	最初の手順 – 集中状態の機械での GAT.....	266
10.3.1	単純な動作アプリケーションの作成.....	266
10.4	機能原則.....	276
10.5	初期化.....	276
10.6	運転モード.....	277
10.6.1	一般情報.....	277
10.6.2	運転モードの状態.....	278
10.6.3	初期化後のターゲット運転モードの設定.....	279
10.6.4	運転モードの変更.....	280
10.6.5	運転モード終了時のユーザー操作.....	280
10.6.6	運転モード開始時のユーザー操作.....	281
10.6.7	集中運転モードのバイパス.....	281
10.7	モジュール.....	282
10.7.1	一般情報.....	282
10.7.2	作成規則.....	282
10.7.3	モジュール インターフェース.....	283
10.7.4	モジュールの状態.....	287
10.7.5	モジュールの連携.....	287
10.8	タスク処理.....	288
10.8.1	タスク設定、GAT.....	288

目次

ページ

10.8.2	GATMotionSyncTask.....	288
10.8.3	GATBackgroundProgTask.....	289
10.9	エラー処理.....	289
10.9.1	エラーの検出および診断.....	289
10.9.2	エラーリアクション.....	290
10.9.3	エラーのクリア.....	293
10.9.4	プロトコル エラー.....	294
10.10	表示.....	294
10.10.1	試運転サポートのための内部表示.....	294
10.10.2	独立した HMI による外部表示 (WinStudio 表示).....	295
10.11	GAT のインターフェース.....	304
10.11.1	一般情報.....	304
10.11.2	機械データ.....	304
10.11.3	エラー処理用データ.....	305
10.11.4	モジュール データ.....	305
10.11.5	軸データ.....	306
10.11.6	コントロール データ.....	307
10.12	操作手順： 一般的なユーザー アクティビティ.....	307
10.12.1	軸データへのアクセス	307
10.12.2	軸の追加.....	308
10.12.3	軸の削除/名前変更	309
10.12.4	運転モードの追加.....	311
10.12.5	エラー クリア後の必要な運転モードの選択.....	311
10.12.6	運転モード仕様の上位ロジック.....	312
10.12.7	状態と移行の追加.....	312
10.12.8	新しいモジュールの追加.....	313
10.12.9	モジュールのインスタンスの追加.....	314
10.12.10	モジュールの無効化.....	314
10.12.11	モジュールの再利用.....	315
10.12.12	モジュールでのグローバル変数とアドレス変数の使用.....	317
10.12.13	MotionTask での処理.....	319
10.12.14	AxisInterface の拡張.....	319
10.12.15	Robot Control.....	320
10.12.16	GAT に基づいたユーザー独自のプログラム フレームワークの生成.....	322
10.12.17	GAT に基づいた PackML V3.0 プログラム フレームワーク.....	323
10.12.18	時間依存でアプリケーション固有のエラーリアクション.....	325
10.12.19	F0 エラーへの GAT エラーリアクションのスイッチオフ	325
10.12.20	バージョン 10/11 から 12/13 への更新.....	325
10.13	GAT、概略.....	326
10.13.1	構造の概要.....	326
10.13.2	機械の最も重要なグローバル変数の概要.....	327
10.13.3	モジュールの最も重要なグローバル変数の概要.....	327
10.13.4	軸の最も重要なグローバル変数の概要.....	327
10.13.5	機械のメソッドとアクションの概要.....	328
10.13.6	機械のメソッドとアクションの概要.....	328
10.13.7	モジュール内の軸へのアクセス.....	330

10.13.8	モジュール内の I/O データへのアクセス.....	330
11	分散状態の機械での GAT.....	331
11.1	一般情報.....	331
11.2	概要.....	331
11.2.1	基本概念.....	331
11.2.2	使用の要件.....	333
11.3	最初の手順 – 分散状態の機械での GAT.....	333
11.3.1	単純な動作アプリケーションの作成.....	333
11.4	機能原則.....	344
11.5	初期化.....	344
11.6	運転モード.....	345
11.7	ユニット.....	346
11.7.1	一般情報.....	346
11.7.2	作成規則.....	347
11.7.3	ユニット インターフェース.....	347
11.7.4	ユニット内の運転モード.....	350
11.7.5	逐次プログラミング.....	351
11.7.6	ユニットの連携.....	352
11.8	タスク管理.....	352
11.8.1	タスク設定、GAT.....	352
11.8.2	BackgroundTask の追加.....	352
11.9	エラー処理.....	353
11.9.1	エラーの検出および診断.....	353
11.9.2	エラーリアクション.....	355
11.9.3	エラーのクリア.....	357
11.9.4	プロトコル エラー.....	358
11.10	製品と機械のデータ処理.....	358
11.10.1	はじめに.....	358
11.10.2	機械データの処理.....	358
11.10.3	製品データの処理.....	358
11.11	表示.....	359
11.11.1	表示の切替.....	359
11.11.2	試運転サポートのための内部表示.....	360
11.11.3	独立した HMI による外部表示 (WinStudio 表示).....	362
11.12	分散状態の機械での GAT のインターフェース.....	371
11.12.1	一般情報.....	371
11.12.2	機械のマシン ハンドラー データ (TE_MachineControl).....	371
11.12.3	集中エラー処理データ (TE_ErrorHandling).....	372
11.12.4	ユニット データ.....	373
11.12.5	軸データ.....	375
11.12.6	コントロール データ.....	375
11.13	操作手順：一般的なユーザー アクティビティ.....	376
11.13.1	軸データへのアクセス.....	376
11.13.2	軸の追加.....	376
11.13.3	軸の削除/名前変更.....	377

目次

	ページ
11.13.4 プライマリ運転モードの追加.....	377
11.13.5 新しいユニットの追加.....	378
11.13.6 ユニットのインスタンスの追加.....	379
11.13.7 ユニットの無効化.....	379
11.13.8 ユニットの再利用.....	380
11.13.9 ユニットでのグローバル変数とアドレス変数の使用.....	381
11.13.10 AxisInterface の拡張.....	381
11.13.11 Robot Control.....	382
11.13.12 GAT に基づいたユーザー独自のプログラム フレームワークの生成.....	385
11.14 GAT – 分散状態の機械、概略.....	387
11.14.1 構造の概要.....	387
11.14.2 機械の最も重要なグローバル変数の概要およびエラー処理の概要.....	387
11.14.3 ユニットの最も重要なグローバル変数の概要.....	388
11.14.4 軸の最も重要なグローバル変数の概要.....	388
11.14.5 TE_MachineControl のメソッドとアクションの概要.....	388
11.14.6 TE_ErrorHandling のメソッドとアクションの概要.....	389
11.14.7 ユニットのメソッドとアクションの概要.....	390
11.14.8 ユニット内の軸へのアクセス.....	390
11.14.9 ユニット内の I/O データへのアクセス.....	390
12 GAT プラグイン.....	391
12.1 はじめに.....	391
12.2 プラグインの追加.....	391
12.3 プラグインの削除.....	391
12.4 プラグインのインストール.....	391
12.5 プラグインの作成.....	392
13 サービスとサポート.....	397
索引.....	399

この資料について

1 この資料について

1.1 資料の有効性

対象グループおよび製品フェーズの概要

この資料により、明確に構造化されたロジック アプリケーションの簡単かつ迅速な開発が可能になります。

以下のテンプレートおよび *AxisInterface* が利用可能です。

- GAT compact
- GAT (集中状態の機械)
- GAT (分散状態の機械)
- ユーザー定義 GAT

下の図で、枠付きで表示されているアクティビティ、製品フェーズ、対象グループは現在の資料を参照します。

例: "Engineering" (エンジニアリング) の製品フェーズで、対象グループ "programmer" (プログラマー) はこの資料を使用して "programming" (プログラミング) のアクティビティを実行できます。

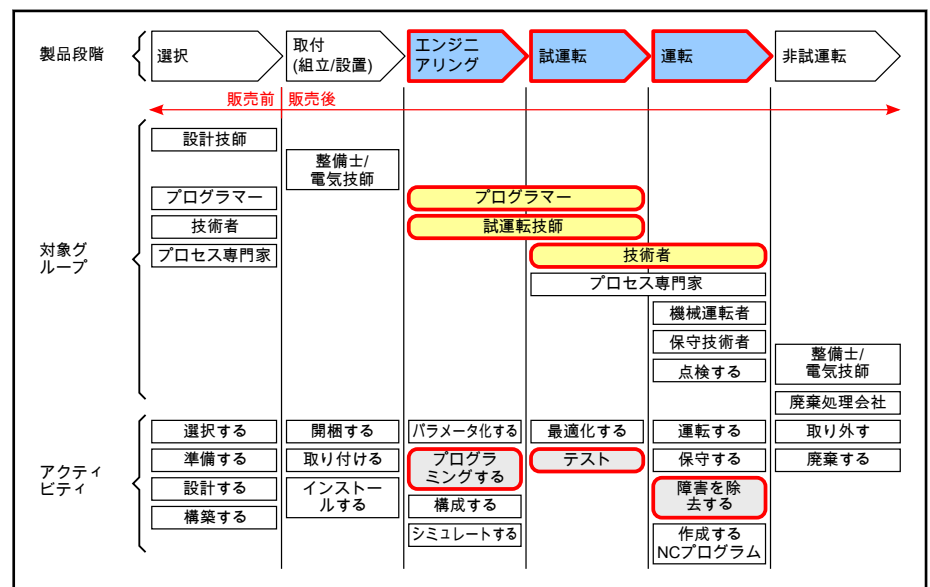


図 1-1: 対象グループ、製品フェーズおよび対象グループのアクティビティへの本資料の割当

1.2 資料の構成

資料の最初の部分には、使用および安全に関する重要な指示が記載されています (19 ページ "使用上の重要な指示" 2 章 および 21 ページ "電気ドライブおよび制御の安全の手引き" 3 章)。

31 ページ "はじめに" 4 章には IndraMotion MLC/IndraLogic XLC システム用 GAT 異形の概要が記載されています。

33 ページ "アプリケーションの自動化" 5 章ではアプリケーションの自動化におけるタスクについて説明しています。

49 ページ "ML_TechInterface.library" 6 章には GAT により使用されるこのライブラリの機能およびファンクションブロックの説明が記載されています。

201 ページ "GAT^{compact}" 8 章ではプログラミング フレーム、"GAT^{compact}" の用途について説明しています。

この資料について

243 ページ "sercos インターフェースなしの IndraLogic XLC 用 GATcompact"
9 章 には sercos インターフェースのない IndraLogic XLC システム用のプ
ログラミング フレーム、"GATcompact" の説明が記載されています。

263 ページ "集中状態の機械での GAT" 10 章 では IndraMotion MLC システ
ム用のプログラミング フレーム、"集中状態の機械での GAT" の用途について
説明しています。

331 ページ "分散状態の機械での GAT" 11 章 では IndraMotion MLC システ
ム用のプログラミング フレーム、"分散状態の機械での GAT" の用途について
説明しています。

391 ページ "GAT プラグイン" 12 章 ではプラグインの使用について説明し
ています。

397 ページ "サービスとサポート" 13 章 には Bosch Rexroth カスタマー サ
ービスのヘルプ デスクに関する情報が記載されています。

1.3 必須資料および補足資料

資料タイトルおよび種別コードと部品番号

IndraWorks		MLC	XLC
/36/	Rexroth IndraWorks 13VRS Software Installation (Rexroth IndraWorks 13VRS ソフトウェアのインス トール) DOK-IWORKS-SOFTINS*V13-CORS-EN-P, R911336880 この資料では IndraWorks のインストールについて説明されています。	X	X
/5/	Rexroth IndraWorks 13VRS Engineering (Rexroth IndraWorks 13VRS エンジニアリング) DOK-IWORKS-ENGINEE*V13-APRS-EN-P, R911336870 この資料では Rexroth Engineering ツールが組み込まれている IndraWorks の用途について説明して います。 IndraWorks での作業方法およびオシロスコープ機能の操作方法に関する説明も含まれてい ます。	X	X
/20/	Rexroth IndraMotion MLC 13VRS Functional Description(Rexroth IndraMotion MLC 13VRS 機能説明) DOK-MLC***-FUNC****V13-APRS-EN-P, R911336295 この資料では、ウィザード、コンテキスト メニュー、ダイアログ、コントロールの試運転、装置設定 および IndraMotion MLC の機能について説明しています。	X	
/20/	Rexroth IndraLogic XLC 13VRS Functional Description (Rexroth IndraLogic XLC 13VRS 機能説明) DOK-XLC***-FUNC****V13-APRS-EN-P, R911336352 この資料では、ウィザード、コンテキスト メニュー、ダイアログ、コントロールの試運転、装置設定 および IndraLogic XLC の機能について説明しています。		X
/7/	Rexroth IndraWorks 13VRS CamBuilder DOK-IWORKS-CAMBUIL*V13-APRS-EN-P, R911336291 この資料では、カム編集ツール、CamBuilder の基本原理と操作方法について説明しています。	X	X
/37/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Automation Interface (Rexroth IndraLogic XLC IndraMotion MLC 13VRS 自動化インターフェース) DOK-XLCMLC-AUT*INT*V13-APRS-EN-P, R911336356 この資料では、自動化インターフェースのインターフェースを介した IndraWorks プロジェクト デー タへのスクリプトベースのアクセスについて説明しています。	X	X
/38/	Rexroth IndraWorks 12VRS FDT Container (Rexroth IndraWorks 12VRS FDT コンテナ) DOK-IWORKS-FDT*CON*V12-APRS-EN-P, R911334398 この資料では、IndraWorks FDT コンテナの機能について説明しています。 プロジェクトでの機能の 起動および DTM での作業も取り上げています。	X	X

この資料について

IndraWorks		MLC	XLC
/29/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Project Conversion (Rexroth IndraLogic XLC IndraMotion MLC 13VRS プロジェクト変換) DOK-XLCMLC-PROCONV*V13-APRS-EN-P, R911336366 この資料では、IndraLogic 2G を用いた IndraWorks バージョン 12 の IndraLogic 04VRS および IndraMotion MLC04VRS のプロジェクト変換について説明しています。 モーションおよび PLC に関する変更について詳細に説明されています。	X	X
/28/	Rexroth IndraMotion MLC 13VRS Commissioning (Rexroth IndraMotion MLC 13VRS 試運転) DOK-MLC***-STARTUP*V13-CORS-EN-P, R911336308 この資料では、IndraMotion MLC システムの試運転および点検を行う手順について説明しています。 頻繁に行われるタスクのチェックリストおよび手順の詳細な説明も含まれています。	X	

表 1-1: XCL/MLC 資料の概要

モーション		MLC	XLC
/23/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS PLCopen Libraries (Rexroth IndraLogic XLC IndraMotion MLC 13VRS PLCopen ライブラリ) DOK-XLCMLC-FUNLIB**V13-LIRS-EN-P, R911336360 この資料では、IndraLogic XLC/IndraMotion MLC の RIL_CommonTypes、ML_Base および ML_PLCOpen ライブラリのファンクションブロック、機能、データ タイプについて説明しています。 また、ファンクションブロックのエラーリアクションも含まれています。	X	X
/27/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Generic Application Template (Rexroth IndraLogic XLC IndraMotion MLC 13VRS 汎用アプリケーション テンプレート) DOK-XLCMLC-TF*GAT**V13-APRS-EN-P, R911336370 この資料は、IndraLogic PLC のプログラマーに構造化されたテンプレートを提供します。 このテンプレートは PLC プログラミング コードの追加および編集に使用できます。 テンプレート、テンプレート ウィザードおよびアプリケーション例が含まれています。	X	X
/31/	Rexroth IndraMotion MLC 13VRS RCL Programming Instruction (Rexroth IndraMotion MLC 13VRS RCL プログラミングに関する説明) DOK-MLC***-RCL*PRO*V13-APRS-EN-P, R911336297 この資料には、RobotContro に関する情報が記載されています。 プログラミング言語、RCL (RobotControl Language) に焦点が当てられています。 プログラムの構造、変数、関数、動作ステートメント、必須システム パラメータについて説明しています。	X	
/21/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Parameters (Rexroth IndraLogic XLC IndraMotion MLC 13VRS パラメータ) DOK-XLCMLC-PARAM***V13-RERS-EN-P, R911336364 この資料では、XLC/MLC システムのパラメータのほか、パラメータ化とプログラミングとの間のインタラクションについて説明しています。 軸パラメータ、制御パラメータ、運動学的パラメータ、タッチプローブ パラメータ、プログラマブル リミット スイッチ パラメータが含まれています。	X	X
/10/	Rexroth IndraDrive Firmware for Drive Controllers MPH-, MPB-, MPD-, MPC-07 (ドライブ コントローラ MPH-、MPB-、MPD-、MPC-07 用 Rexroth IndraDrive ファームウェア) DOK-INDRV*-MP*-07VRS**-FKRS-EN-P, R911328670 この資料では、IndraDrive ファームウェアのタイプ MPH-07、MPB-07、MPD-07 および MPC-07 のすべての機能特性について説明しています。		
/11/	Rexroth IndraDrive MPx-17 Functions (Rexroth IndraDrive MPx-17 機能) DOK-INDRV*-MP*-17VRS**-APRS-EN-P, R911331236 この資料では、IndraDrive ファームウェアのタイプ MPB-17、MPM-17、MPC-17 および MPE-17 のすべての機能特性について説明しています。		

表 1-2: XCL/MLC 資料の概要

この資料について

フィールド バス		MLC	XLC
/4/	Rexroth IndraWorks 13VRS Field Buses (Rexroth IndraWorks 13VRS フィールド バス) DOK-IWORKS-FB*****V13-APRS-EN-P, R911336872 この資料では、サポートされているフィールド バスおよびそれらの診断ファンクションブロックについて説明しています。	X	X
/4a/	Rexroth IndraWorks 13VRS Field Buses Libraries (Rexroth IndraWorks 13VRS フィールド バス ライブラリ) DOK-IWORKS-FB*LIB**V13-LIRS-EN-P, R911337857 このマニュアルでは、以下のフィールド バス ライブラリについて説明しています。 RIL_ProfibusDP_02 、 RIL_ProfibusDP Slave 、 RIL_ProfinetIO 、 RIL_ProfinetIODevice 、 RIL_EtherNetIPAdapter、RIL_MappingList、RIL_SERCOSIII、RIL_Inline (それぞれのファンクション ブロックの診断およびエラーリアクションを含む)	X	X
/63/	SERCOS System Manual for I/O Devices (I/O 機器用 SERCOS システム マニュアル) DOK-CONTRL-ILS3*****-APRS-EN-P, R911333512 この資料では、SERCOS インターフェースを備えた I/O 機器の設定、パラメータ化、試運転および診 断について説明しています。	X	X

表 1-3: XCL/MLC 資料の概要

HMI		MLC	XLC
/8/	Rexroth IndraWorks 13VRS HMI DOK-IWORKS-HMI*****V13-APRS-EN-P, R911336874 この資料では、IndraWorks HMI Engineering および IndraWorks HMI Operation のユーザー インター フェースの機能、設定および操作について説明しています。	X	X
/6/	Rexroth IndraWorks 13VRS WinStudio DOK-IWORKS-WINSTUD*V13-APRS-EN-P, R911336882 この資料では、ソフトウェアのインストール、WinStudio での作業およびアプリケーションの作成と 操作について説明しています。	X	X
/50/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS HMI Connection (Rexroth IndraLogic XLC IndraMotion MLC 13VRS HMI 接続) DOK-XLCMLC-HMI*****V13-APRS-EN-P, R911336362 この資料では、IndraLogic XLC および IndraMotion MLC がサポートしている表示システムとその接続 について説明しています。	X	X

表 1-4: XCL/MLC 資料の概要

PLC		MLC	XLC
/3/	Rexroth IndraWorks 13VRS IndraLogic 2G Programming Instruction (Rexroth IndraWorks 13VRS IndraLogic 2G プログラミングに関する説明) DOK-IWORKS-IL2GPRO*V13-APRS-EN-P, R911336876 この資料では、PLC プログラミング ツール IndraLogic 2G およびその使用方法について説明していま す。基本的な使用方法、最初の手順、表示、メニュー項目およびエディタが含まれています。	X	X
/33/	Rexroth IndraWorks 13VRS Basic Libraries IndraLogic 2G (Rexroth IndraWorks 13VRS 基本ライブラ リ IndraLogic 2G) DOK-IL*2G*-BASLIB**V13-LIRS-EN-P, R911336285 この資料では、システムの包括的な PLC ライブラリについて説明しています。	X	X

表 1-5: XCL/MLC 資料の概要

この資料について

技術		MLC	XLC
/30/	Rexroth IndraMotion MLC 13VRS Technology Libraries(Rexroth IndraMotion MLC 13VRS 技術ライブラリ) DOK-MLC***-TF*LIB**V13-LIRS-EN-P, R911336324 この資料では、「ML_TechInterface.library」、「ML_TechMotion.library」、「RMB_TechCam.library」、「ML_TechBase.library」のファンクションブロック、機能およびデータタイプについて説明しています。ワインダー機能、レジスタコントローラ機能および CrossCutter 機能向けライブラリも含まれています。	X	
/60/	Rexroth IndraMotion MLC 13VRS RegisterControl (ライブラリ) DOK-MLC***-REGI*CO*V13-LIRS-EN-P, R911336306 この資料では、個々のファンクションブロックの入力および出力について説明しており、その使用に関する注意事項が記載されています。	X	
/62/	Rexroth IndraMotion MLC 13VRS RegisterControl (アプリケーション マニュアル) DOK-MLC***-REGI*CO*V13-APRS-EN-P, R911336304 この資料では、グラフィック印刷機向け統合レジスタコントロールの用途について説明しています。マークストリームセンサーのコンポーネント、HMI アプリケーションおよびエラー回復オプションが説明されています。この説明により、レジスタコントロールの操作方法、エラーの対処方法および診断をクエリする方法に関する情報が得られます。この資料は、機械の調整担当者および運転担当者向けに書かれています。	X	
/49/	Rexroth IndraMotion MLC 13VRS Winder Function Application(Rexroth IndraMotion MLC 13VRS ワインダー機能アプリケーション) DOK-MLC***-TF*WIND*V13-APRS-EN-P, R911336326 このアプリケーション関連のシステム資料では、ワインダー技術の機能の用途について説明しています。	X	

表 1-6: XCL/MLC 資料の概要

ハードウェア		MLC	XLC
/1/	Rexroth IndraControl L45/L65/L85 Control (Rexroth IndraControl L45/L65/L85 コントロール) DOK-CTRL-ICL45L65L85-PRRS-EN-P, R911332116 この資料では、IndraControl L45/L65/L85 コントロールについて説明しています。	X	X
/2/	Rexroth IndraControl L25 DOK-CTRL-IC*L25****-PRRS-EN-P, R911328474 この資料では、IndraControl L25 コントロールについて説明しています。	X	X
/24/	Rexroth IndraControl Lxx 13VRS Function Modules (Rexroth IndraControl Lxx 13VRS 機能モジュール) DOK-CTRL-FM*LXX**V13-APRS-EN-P, R911336283 この資料では、エンジニアリングおよび診断を含む、Lxx コントロールのすべての機能モジュールについて説明しています。	X	X
/12/	Rexroth IndraDrive Drive Controllers MPx-02 to MPx-07 (Rexroth IndraDrive ドライブ コントローラ MPx-02 ~ MPx-07) DOK-INDRV*-GEN-**VRS**-PARS-EN-P, R911297317 この資料では、IndraDrive シリーズのドライブ コントローラ用ファームウェアに実装されているすべてのパラメータについて説明しています。ドライブ コントローラのパラメータ化をサポートしています。		

この資料について

ハードウェア		MLC	XLC
/13/	Rexroth IndraDrive MPx-02 to MPx-07 and HMV (Rexroth IndraDrive MPx-02 ~ MPx-07 および HMV) DOK-INDRV*-GEN-**VRS**-WARS-EN-P, R911297319 この資料では、以下のファームウェアに実装されているすべての診断について説明しています。 - ドライブ コントローラのファームウェア MPx-02 ~ MPx-08 および - サプライ デバイスのファームウェアでタイプが「HVM」のもの。 トラブルシューティングの際に運転担当者ならびにプログラマーをサポートします。		
/35/	Rexroth IndraDrive Drive Controller Control Sections CSB01, CSH01, CDB01 (Rexroth IndraDrive ドライブ コントローラ コントロール セクション CSB01, CSH01, CDB01) DOK-INDRV*-CSH*****-PRRS-EN-P, R911295012 この資料は、Rexroth IndraDrive M および Rexroth IndraDrive C 製品シリーズのドライブ コントローラのコントロール セクション CSB01, CDB01 および CSH01 を選択して使用するためのプロジェクト計画策定マニュアルです。		

表 1-7: XCL/MLC 資料の概要

診断およびサービス		MLC	XLC
/26/	Rexroth IndraWorks 13VRS IndraMotion Service Tool (Rexroth IndraWorks 13VRS IndraMotion サービス ツール) DOK-IWORKS-IMST****V13-RERS-EN-P, R911337707 この資料では、IndraMotion サービス ツール (IMST) について説明しています。IMST は、イーサネット高速接続を介してコントロール システム IndraMotion MLC L25、L45 または 65 および MLP にアクセスするための Web ベースの診断ツールです。IMST を利用すると、OEM、エンド ユーザーおよびサービス エンジニアがシステムにアクセスしてリモートで診断を行うことができます。Internet Explorer バージョン 6 または 7 がインストールされた PC が必要です。	X	X
/22/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Diagnostics (Rexroth IndraLogic XLC IndraMotion MLC 13VRS 診断) DOK-XLCMLC-DIAG****V13-RERS-EN-P, R911336358 この資料には、コントロール システム IndraLogic XLC および IndraMotion MLC に実装されているすべての制御パラメータが含まれています。	X	X

表 1-8: XCL/MLC 資料の概要

システム概要		MLC	XLC
/48/	Rexroth IndraMotion for Printing 13VRS System Overview(Rexroth IndraMotion for Printing 13VRS システム概要) DOK-IM*PR*-SYSTEM**V13-PRRS-EN-P, R911336289 この資料では、「IndraMotion for Printing」という製品について説明しています。コントロール システム、ドライブ システム、I/O システムのほか、試運転およびプログラミングが含まれています。	X	
/48/	Rexroth IndraMotion for Packaging 13VRS System Overview (Rexroth IndraMotion for Packaging 13VRS システム概要) DOK-IM*PA*-SYSTEM**V13-PRRS-EN-P, R911336287 この資料では、「IndraMotion for Packaging」という製品について説明しています。コントロール システム、ドライブ システム、I/O システムのほか、試運転およびプログラミングが含まれています。	X	

この資料について

システム概要		MLC	XLC
/9/	Rexroth IndraMotion MLC 13VRS System Overview (Rexroth IndraMotion MLC 13VRS システム概要) DOK-MLC***-SYSTEM**V13-PRRS-EN-P, R911336310 この資料には、IndraMotion MLC 自動化システムの言及されているバージョンのハードウェア/ソフトウェア コンポーネントに関する概要が記載されています。システムのアセンブリに役立ちます。	X	
/9/	Rexroth IndraLogic XLC 13VRS System Overview (Rexroth IndraLogic XLC 13VRS システム概要) DOK-XLC***-SYSTEM**V13-PRRS-EN-P, R911336354 この資料には、IndraLogic XLC 自動化システムの各バージョンのハードウェア/ソフトウェア コンポーネントに関する概要が記載されています。システムのアセンブリに役立ちます。		X

表 1-9: XCL/MLC 資料の概要

最初の手順		MLC	XLC
/25/	Rexroth IndraMotion MLC 13VRS, First Steps (Rexroth IndraMotion MLC 13VRS, 最初の手順) DOK-MLC***-F*STEP**V13-CORS-EN-P, R911336293 この資料では、IndraMotion MLC および RobotControl の最初の手順について説明しています。ハードウェアとソフトウェアの前提条件のほか、プロジェクトの作成も含まれています。	X	
/25/	Rexroth IndraLogic XLC 13VRS, First Steps (Rexroth IndraLogic XLC 13VRS, 最初の手順) DOK-XLC***-F*STEP**V13-CORS-EN-P, R911336350 この資料では、IndraLogic XLC の最初の手順について説明しています。ハードウェアとソフトウェアの前提条件のほか、プロジェクトの作成も含まれています。		X

表 1-10: XCL/MLC 資料の概要

1.4 情報の表記方法

1.4.1 安全上の指示

資料に安全上の指示が記載されている場合、その指示には特定の信号ワード ("危険"、"警告"、"注意"、"通知") が含まれており、また時として安全上の警告マーク (ANSI Z535.6-2006 に準拠) が含まれていることがあります。

信号ワードは安全上の注意にユーザーの目を向けさせ、リスクの可能性を示します。

"危険"、"警告"、"注意"の信号ワードの前に記載されている安全上の警告マーク (感嘆符が 3 つ書かれた三角形の安全反射板) は、ユーザーにとっての危険を示します。

本書では、安全上の指示は以下のとおり表記されています。

危険

この安全上の指示を守らなかった場合、死または大きなけがを招きます。

警告

この安全上の指示を守らなかった場合、死または大きなけがを招く可能性があります。

注意

この安全上の指示を守らなかった場合、中程度または軽度のけがを招く可能性があります。

この資料について

注記

この安全上の指示を守らなかった場合、物品や所有物の損傷を招く可能性があります。

1.4.2 使用されている記号

注 注は次のように表記されています。



これはユーザー向けの注です。

ヒント

ヒントは次のように表記されています。



これはユーザー向けのヒントです。

1.4.3 用語および略語

用語	説明
GAT	汎用アプリケーション テンプレート (Generic Application Template)
AxisInterface	AxisInterface は PLCopen モーションファンクションブロックをプールして拡張し、操作しやすいインターフェースをドライブ機能に提供します。
Imc インターフェース	IMC インターフェース は操作しやすいインターフェースをコントロール機能に提供します。 診断のための制御信号とパラメータ、およびコントロールの管理が含まれています。
TechInterface	TechInterface は次のコンポーネントで構成されています。 - AxisInterface および - Imc インターフェース
ファンクションブロック	アプリケーション固有の機能を提供します
IndraMotion MLC	モーション、ロボットおよびロジック制御機能を備えたコンパクトなモーションロジック システム
IndraLogic XLC	モーション機能が組み込まれたコンパクトな PLC システム
IndraWorks エンジニアリング フレームワーク	Bosch Rexroth のプロジェクト計画策定・ 試運転ツール
IndraDrive	ドライブ コントロール ユニット
ウィザード	ウィザードは複数のダイアログを通じてユーザーが人間工学的にデータを入力できるよう導きます
IEC	国際電気標準会議 (International Electrotechnical Commission)
sercos	sercos (SErial REaltime COmmunication System) インターフェースは、コントロールとドライブ間の通信用の世界的な標準インターフェースです

この資料について

XML	拡張マークアップ言語 (Extensible Markup Language)。階層的に構造化されたデータをテキスト データとして表記するためのマークアップ言語です
CSV	このファイル形式は、単純な構造のデータを保存したり交換したりするためのテキスト ファイルの構造を記述します。ファイル拡張子 .csv は、"comma separated values" の略です。
IMC	IndraMotion Control
MLD-S	Motion Logic Drive (シングル ドライブ)
MLD-M	Motion Logic Drive (マルチ ドライブ)
i.e.	すなわち
e.g.	例
ms	ミリ秒
mm	ミリメートル
%	パーセント
rpm	回転数

表 1-11: 使用されている名称および略語

2 使用上の重要な指示

2.1 使用目的

2.1.1 はじめに

Bosch Rexroth の製品は最新の技術によって開発され、製造されています。製品は出荷前に検証されており、操作の安全性と信頼性が保証されています。

**警告**

製品を正しく使用しないと、人身の傷害および器物の破損につながります。

製品は、工業環境での使用を目的として開発されており、意図された用途でのみ使用してください。製品が意図された用途で使用されなかった場合、製品の誤った使用により器物の破損または人身の傷害につながる状況が発生します。



製造元である Bosch Rexroth は、製品が意図された用途で使用されなかったことにより損害が生じた場合、いかなる保証、責任または損害についての支払い義務も負わないものとし、製品の目的外の使用に起因するあらゆるリスクは、ユーザーが単独で負うものとしします。

Bosch Rexroth の製品を使用する前に、製品が意図された用途で使用されることを保証するために、以下の要件を満たす必要があります。

- Bosch Rexroth 製品を何らかの形で取り扱うすべての関係者は、関連する安全上の指示および使用目的を読み、承諾する義務を負います。
- ハードウェア製品は、変更を加えてはならず、初期の状態を保持する必要があります。すなわち、いかなる構造的な変更も許可されません。
- ソフトウェア製品の逆コンパイルまたはソースコードの変更は許可されていません。
- 損傷または不具合のある製品は、設置あるいは運転を行わないでください。
- 製品が関連する資料で説明されているとおりに設置されていることを確認する必要があります。

2.1.2 使用範囲および用途

製品 Bosch Rexroth の IndraMotion MLC/IndraLogic XLC はモーション/ロジック アプリケーションでの使用を目的としています。



IndraMotion MLC/IndraLogic XLC は、本書に記載されている付属部品および取付部品とともに使用する必要があります。明記されていないコンポーネントは、取り付けたり接続したりしてはなりません。ケーブルおよび配線についても同様です。

製品の運転は、明記されているコンポーネントの構成および組み合わせ、ならびに各機能の説明で指示および指定されているソフトウェアとファームウェアでのみ行う必要があります。

IndraMotion MLC/IndraLogic XLC は、ロジック アプリケーションを対象とし、また、単一軸および複数軸の駆動・制御タスクで使用するために開発されています。

使用上の重要な指示

IndraMotion MLC/IndraLogic XLC のアプリケーション固有の要件を満たすために、異なる性能とインターフェースを備えたさまざまな種類のデバイスが用意されています。

IndraMotion MLC/IndraLogic XLC の一般的な用途を以下に挙げます。

- ハンドリングシステムおよび取り付けシステム
- 梱包機械および食品機械
- 印刷機械および紙加工機械
- 工作機械
- プレス
- 鋳造・圧延技術
- 運搬工学
- ビルディング オートメーション
- 特殊用途機械
- 木工機械

IndraMotion MLC/IndraLogic XLC は、本書で指定されている取付・設置条件、位置・環境条件（温度、保護の程度、湿度、EMC など）でのみ運転できません。

2.2 目的外の使用

指定された用途範囲または資料および技術データで説明され、指定されている条件以外の運転条件での IndraMotion MLC/IndraLogic XLC の使用は、"目的外の"使用とみなされます。

以下の場合、IndraMotion MLC/IndraLogic XLC を使用しないでください。

- IndraMotion MLC/IndraLogic XLC が、指定された環境条件に合致しない運転条件にある場合。水中、気温の変化が極端に大きい環境、最高気温が極端に高い環境、または爆発を起こしやすい場所での運転は禁じられています。
- 意図された用途が Bosch Rexroth によって明示的に発表されていない場合。一般的な安全上の指示に記載されている情報をお読みください。
- 機械またはシステムに安全ではないインターネット接続が使用されている場合。IndraMotion MLC/IndraLogic XLC のイーサネット インターフェースは、不正なアクセスに対しては安全に保護されてはいません。このため、制御機器は安全なネットワークでのみ使用してください（セキュリティ）。安全なネットワークで使用しないと、不正なアクセスにより故障または制御機器の不具合が発生する可能性があります。

運転は、以下の標準およびガイドラインに基づいて禁じられています。

- エレベーター (Directive 95/16/EC)
- ATEX (94/9/EC)
- バーナー コントロール システム (EN 298)
- 圧力装置 (EN 764-7:2002/AC)

さらに、DIN EN ISO 13849 に基づいた制御機器の安全関連部品としての使用は、目的外の使用とみなされます。追加的な安全関連の認証済みコンポーネントによって機能的安全性が保証される必要があります。

3 電気ドライブおよび制御の安全の手引き

3.1 用語の定義

アプリケーション関連文書	アプリケーション関連文書は、製品の使用、および製品の設定、統合、設置、取り付け、試運転、運転、メンテナンス、修理、および廃止措置の安全に関連した機能についてユーザーに知らせるために使用される文書全体を意味します。以下の用語はユーザーガイド、運転マニュアル、試運転マニュアル、プロジェクト プランニング マニュアル、アプリケーション マニュアルなどの種類の文書でも使用されます。
コンポーネント	コンポーネントとは指定された機能を持つ要素の組み合わせで、装置、デバイス、またはシステムの一部です。たとえば、ドライブおよび制御システムのコンポーネントは、電源ユニット、ドライブ コントローラ、主電源チョーク、主電源フィルタ、モーター、ケーブルなどです。
制御システム	制御システムとは単一の機能ユニットとして市販されている、相互接続された複数の制御コンポーネントです。
デバイス	デバイスとは定義された機能を備えており、ユーザーを対象として個々の商品として市販される最終製品です。
電気備品	電気備品は機械、変圧器、切り替えデバイス、ケーブル、ライン、電力消費デバイス、回路基板アセンブリ、プラグイン ユニット、制御キャビネットなどに電気エネルギーを生成、変電、送電、配電、または印加するために使用される物を含みます。
電気ドライブシステム	主電源とモーター軸間のすべてのコンポーネントを含む電気ドライブシステム; たとえば、電気モーター、モーターエンコーダー、電源ユニットおよびドライブ コントローラ、また主電源フィルタ、主電源チョーク、対応するラインとケーブルのような補助、追加のコンポーネントを含みます。
設備	設備は定義された目的のため、および定義された場所で相互接続された複数のデバイスまたはシステムで、単一の機能ユニットとして市販されることを目的としていないものから成っています。
機械	機械とは相互接続された部品またはユニットで、少なくともその1つが可動する全体です。このように、機械は対応する機械ドライブ要素、制御および電力回路で構成され、特定の用途のために組み立てられています。たとえば機械は素材の加工、処理、移動、または包装を目的としています。「機械」という用語はまた、統合された全体として機能するような仕方では手配および管理された機械の組み合わせも意味しています。
メーカー	メーカーは、個人または法人名で市販される製品の設計および製造に責任を持っている個人または法人。または、最終製品、最終部品、最終要素を使用あるいは、請負業者に作業を外注することができます。しかし、メーカーは常に製品の全体的な管理を行い、製品に対して責任を負うために必要な権限を保持する必要があります。
製品	製品の例：デバイス、コンポーネント、部品、システム、ソフトウェア、ファームウェアなど。
プロジェクト プランニング マニュアル	プロジェクト プランニング マニュアルはシステム、機械、または設備のサイジングと計画を支援するアプリケーション関連文書の一部です。
資格のある作業員	このアプリケーション関連文書での、資格のある作業員とは、電気ドライブと制御システムのコンポーネントの設置、取り付け、試運転、操作、ならびにそれが意味する危険を熟知し、仕事に必要な資格を有している者を指します。これらの資格を満たすには、特に以下が必要です。 1)電気回路やデバイスの安全な電源オン/オフの切り替えや接地、さらにそれにマークを付けるためのトレーニングや説明を受ける、または承認を得ること。

電気ドライブおよび制御の安全の手引き

- 2) 十分に安全な装置を保守および使用するトレーニングまたは説明を受けること。
- 3) 応急措置の指導に出席すること。

ユーザー ユーザーとは、市販されている製品を接地、試運転、または使用している人物です。

3.2 一般情報

3.2.1 安全上の指示の使用と、他者への引渡し

製品に付属のすべてのドキュメントを最初に読んでから、電気ドライブおよび制御システムのコンポーネントを設置および操作してください。これらのコンポーネントで作業を行う前に、これらの安全上の指示およびすべてのユーザードキュメントを読み、理解してください。コンポーネントのユーザードキュメントをお持ちでない場合は、お客様担当の Bosch Rexroth 販売パートナーにお問い合わせください。コンポーネントの安全な操作に責任を負う担当者にこれらのドキュメントを速やかに送付するよう依頼してください。

コンポーネントをいかなる形で他者に転売、貸し出しおよび/または譲渡する場合も、これらの安全上の指示を、使用者の国の公用語でコンポーネントと一緒に引き渡す必要があります。

これらのコンポーネントの不適切な使用、本書にある安全上の指示の不履行、または安全装置の無効化を含む製品の改造は、物的損害、人的損害、感電、または死亡を招くことがあります。

3.2.2 安全な使用のための要件

人的損害および/または物的損害のリスクを排除するために、電気ドライブおよび制御システムの最初の試運転の前に以下の指示を読んでください。これらの安全上の指示に従う必要があります。

- Bosch Rexroth は、安全上の指示に従わなかったことに起因する損害に責任を負いません。
- 試運転前に、母国語で書かれた運転、メンテナンス、および安全上の指示をお読みください。利用可能な言語のアプリケーション関連文書でよく理解できない箇所があった場合には、サプライヤーにお問い合わせの上で確認してください。
- 適切で正しい輸送、保管、取り付け、設置、さらに運転やメンテナンスに注意を払うことが、このコンポーネントを最適かつ安全に運転するための前提条件です。
- 電気ドライブおよび制御システムのコンポーネントを取り扱う、またはその付近で作業できるのは資格のある作業員のみとなります。
- Bosch Rexroth によって承認されたアクセサリおよびスペアパーツのみを使用してください。
- 電気ドライブおよび制御システムのコンポーネントを運転している国の安全上の規制および要件に従ってください。
- 電気ドライブおよび制御システムのコンポーネントは、それぞれに定義された方法でのみ使用してください。"適切な使用"の章を参照してください。
- アプリケーション関連文書に記載された環境および動作条件に従う必要があります。
- 機能的安全性に関連する用途が許可されるのは、アプリケーション関連資料に「統合安全技術」と明確かつ明示的に記載されている場合のみとなります。許可されていない場合は、除外してください。機能的安全性

電気ドライブおよび制御の安全の手引き

とは電気、電子あるいはプログラム可能な制御システムが影響を与え得る人的損害のリスク縮小対策上の安全コンセプトです。

- 納品されたコンポーネントの使用に関してアプリケーション関連文書で提出されている情報には、用途および提案に関する例のみが記載されています。

機械および設備メーカーは以下を遵守する必要があります。

- 納品されたコンポーネントがユーザーの個別の用途に適していることを点検し、コンポーネントの使用に関し本書で提供されている情報を確認してください。
- ユーザーの個別の用途が該当する安全規定および基準に準じており、必要な措置、変更および補完が実施されていることを確認してください。
- 納品されたコンポーネントの試運転は、コンポーネントが設置された機械や設備が国の定める規定、安全仕様および仕様基準を順守している場合のみ許可されます。
- 用途に関して国の定める EMC 規制に適合している場合のみ運転することが許可されます。
- EMC 要件に適合した設置のための支持は、個別のアプリケーションで見出すことができます。

機械または装置メーカーは、国の規定によって定められている限界値を順守する責任があります。

- コンポーネントの技術データ、接続、および設置条件は、個々のアプリケーション関連文書に記載されており、常に順守する必要があります。

ユーザーが考慮する必要のある国の規定

- ヨーロッパ諸国：ヨーロッパ EN 規格に準ずる
- アメリカ合衆国 (USA) :
 - 米国電気規約(NEC)
 - 全国電気製造業者協会(NEMA)、および地域の技術規定
 - 全国防火協会(NFPA)の規定
- カナダ：カナダ規格協会(CSA)
- その他の国：
 - 国際標準化機構(ISO)
 - 国際電気標準会議(IEC)

3.2.3 不適切な使用による危険

- 高電圧および高動作電流の危険があります。電気ショックによる死亡または重傷を招く危険があります。
- 接続が正しくないため、高電圧が発生しています。電気ショックによる死亡または怪我を招く危険があります。
- 危険な動作です。思わぬモーターの作動による死亡、重傷または物的損害の危険があります。
- 電気ドライブ システムの近くにいる、心臓ペースメーカー、金属製のインプラント、および補聴器を使用する作業者に健康上の危険があります。
- 加熱された筐体の表面により火傷する危険があります。
- 不適切な取り扱いにより怪我をする危険があります。粉碎、せん断、切断、打撃によって怪我をすることがあります。

電気ドライブおよび制御の安全の手引き

- バッテリーを不適切に取り扱くと怪我をする危険があります。
- 加圧ラインの不適切な取り扱いにより怪我をする危険があります。

3.3 特定の危険に関する指示

3.3.1 電気部品および筐体への接触に対する保護



このセクションでは、50 V以上の電圧を使用する電気ドライブおよび制御システムのコンポーネントについて説明します。

50 V以上の電圧を伝導する部品に触ると、怪我や感電が発生することがあります。電気ドライブおよび制御システムのコンポーネントを操作するときに、これらのコンポーネントの部品が危険な電圧を伝導することは避けられません。

高電圧の危険があります。死亡、電気ショックによる怪我、または重傷を招く危険があります。

- 電気ドライブおよび制御システムの電気コンポーネントを運転、維持、および/または修理できるのは資格のある作業員だけです。
- 電力設備を取り扱うときは、一般的な設備および安全に関する規定に従ってください。
- 電源を入れる前に、接続図に従って装置の接地導体をすべての電気コンポーネントに固定接続する必要があります。
- この目的で提供されるコンポーネントのポイントに装置の接続導体が固定接続されていない場合は、短時間の測定やテストのためであっても運転しないでください。
- 50 V以上の電位になる電気部品を取り扱う前に、電気コンポーネントを主電源または電源ユニットから取り外す必要があります。電気コンポーネントが再接続されないようにしてください。
- 電気コンポーネントを使用するときは、以下を遵守してください。
電源を切った後に必ず 30 分待ち、電気コンポーネントに触れる前に充電したコンデンサを放電させます。作業を開始する前に、帯電している部品の電圧を測定し、装置に触っても安全であることを確認してください。
- 電源を入れる前に、安全のために提供されたカバーおよびガードを取り付けてください。
- 電源が入っている状態では、コンポーネントの電気接続点に絶対に触らないでください。
- コンポーネントの電源が入っているときは、コネクタを着脱しないでください。
- 特定の条件下で、電気ドライブシステムを交直両用の残留電流用遮断器で保護した電源で使うことができます (RCDs/RCMs)。
- 内蔵デバイスに制御キャビネットなどの外部筐体を取り付けて、異物や水が入ったり、直接接触しないようにする必要があります。

高筐体電圧および高漏えい電流の危険があります。電気ショックによる死亡または怪我を招く危険があります。

- 電源投入の前、および試運転の前に、電気ドライブおよび制御システムのコンポーネントを、接地位置で装置接地導体に接地、接続します。

電気ドライブおよび制御の安全の手引き

- ドライブおよび制御システムのコンポーネントの装置接地導体を常時主電源に接続します。漏えい電流は、3.5 mA 以上になります。
- 下記の表に従って最狭部断面の装置接地コネクタを設置してください。断面が 10 mm 以下の外部導体² (8 AWG)では、それぞれ外部導体と同様の断面を有する 2 つの装置接地導体を接続することができます。

外部導体の断面	最小断面の装置接地導体 漏えい電流 ≥ 3.5 mA	
	1 装置接地導体	2 装置接地導体
1.5 mm ² (AWG 16)	10 mm ² (AWG 8)	2 × 1.5 mm ² (AWG 16)
2.5 mm ² (AWG 14)		2 × 2.5 mm ² (AWG 14)
4 mm ² (AWG 12)		2 × 4 mm ² (AWG 12)
6 mm ² (AWG 10)		2 × 6 mm ² (AWG 10)
10 mm ² (AWG 8)		-
16 mm ² (AWG 6)	16 mm ² (AWG 6)	-
25 mm ² (AWG 4)		-
35 mm ² (AWG 2)		-
50 mm ² (AWG 1/0)	25 mm ² (AWG 4)	-
70 mm ² (AWG 2/0)	35 mm ² (AWG 2)	-
...	...	...

表 3-1: 装置接地導体の最小断面

3.3.2 感電から保護する超低電圧

保護超低電圧を使用することで、接続装置を超低電圧回路から基本的に絶縁することができます。

Bosch Rexroth による電気ドライブと制御システムのコンポーネントにおける、5 から 50 V の間のすべての接続と端子は PELV ("Protective Extra-Low Voltage")システムです。このため、基本絶縁のあるデバイス (プログラミング装置、PC、ノートパソコン、ディスプレイユニットなど) をこれらの接続部に接続できます。

電気ショックによる死亡または怪我を招く危険があります。接続が正しくないため、高電圧が発生しています。

50V 以上の電圧や回路が含まれているデバイスの超低電圧回路 (例: 本線接続) が Bosch Rexroth 製品に接続されている場合、接続されている超低電圧回路は PELV ("Protective Extra-Low Voltage")の要件に準拠している必要があります。

3.3.3 危険な動作に対する保護

接続されたモーターの制御に傷害がある場合に、危険な動作をすることがあります。以下は一般的な例です。

- 不適切な、または誤った配線 やケーブル接続
- オペレーターのエラー

電気ドライブおよび制御の安全の手引き

- 試運転前のパラメーターの誤った入力
- センサー およびエンコーダの誤作動
- 欠陥のあるコンポーネント
- ソフトウェアまたはファームウェアのエラー

これらのエラーは、装置の電源を入れた直後、あるいは一定期間問題なく運転した後も発生することがあります。

電気ドライブおよび制御システムのコンポーネントでの監視機能により、接続されたデバイスの不具合を通常は十分に防ぐことができます。作業者の安全、特に怪我や物的損害の危険については、これだけでは完全な安全性を確保することはできません。統合された監視機能が有効になるまで、ドライブ動作の傷害が発生する可能性があると思なす必要があります。ドライブ動作の傷害の程度は、制御タイプおよび運転状態によって異なります。

危険な動作です。死亡、怪我、重傷、または物的損傷の危険があります。

設備または機械に対して、電気ドライブおよび制御システムのコンポーネントが設置された特定の条件でリスク評価を準備する必要があります。

リスク評価の結果から、ユーザーは、作業員の安全のため、設備に関する監視機能およびより高いレベルの対策を提供しなければなりません。設備または機械に適用する安全規定を考慮する必要があります。安全装置の無効化、回避、無起動の際には、不測の機械動作やその他の不具合が発生する可能性があります。

事故、怪我、および/または物的損害を避けるため、以下に留意する必要があります。

- 機械の動作範囲および可動機械部品の範囲の空間を確保し、障害物がないようにします。作業者が誤って機械の可動範囲に入ることを防ぐため、以下のような対策をします。
 - － 安全フェンス
 - － 安全ガード
 - － 保護カバー
 - － 光バリア
- 安全フェンスと保護カバーは、あり得る最大の運動エネルギーに耐えられる強度があることを確認してください。
- 作業員の手が届く位置に、緊急停止スイッチを取り付けます。試運転を行う前に、緊急停止装置が動作することを確認します。緊急停止スイッチが動作しない場合は、機械を運転しないでください。
- 予期しない起動を防ぎます。オフスイッチ/オフボタンまたは安全起動ロックアウトによりドライブの電源接続を絶縁します。
- 危険ゾーンに接近または足を踏み入れる前に、ドライブが安全な停止状態になっていることを確認します。
- モーターの電源を切った後の落下や低下を防止するため垂直軸の固定を以下のように強化します。
 - － 機械的に垂直軸を固定する。
 - － 外部ブレーキ/拘束装置/クランプ機構を追加する
 - － 垂直軸の十分な平衡状態を確認する。
- 標準装置のモーター保持ブレーキまたはドライブコントローラによって制御される外部保持ブレーキでは作業者の安全を確実に保証できません。

電気ドライブおよび制御の安全の手引き

- 以下の場合には、マスタースイッチを使用して電気ドライブおよび制御システムのコンポーネントの電源を切り、再接続されないようにします（「ロックアウト」）。
 - メンテナンスおよび修理作業
 - 装置の清掃
 - 長期間装置を使用しないとき
- 電気ドライブや制御システムおよびそのリード線の近くで高周波数のリモートコントロール、無線装置を使用しないでください。これらのデバイスの使用が避けられない場合、電気ドライブや制御システムの最初の試運転の際に、このような高周波数のリモコンや無線装置を正常な使用時の位置で機械や設備に動作不良が生じないか確認します。特別な電波環境両立性(EMC)試験を行うことが必要になる可能性があります。

3.3.4 運転および取り付け中の磁界および電磁界に対する保護

電流が流れている導電体やモーターの永久磁石によって生じる磁界や電磁界は、心臓ペースメーカー、金属インプラント、および補聴器を使用する人に重大な危険となります。

心臓ペースメーカー、金属製インプラントおよび補聴器の使用者には電気コンポーネント付近は健康上危険です。

- 心臓ペースメーカーおよび金属インプラントの使用者は以下のエリアに立ち入らないでください。
 - 電気ドライブおよび制御システムのコンポーネントが取り付けられ、試運転および運転がされている領域。
 - 永久磁石を使用しているモーターの部品が保管、修理または取り付けられている領域。
- 心臓ペースメーカーの使用者がそれらの領域に立ち入る必要がある場合、事前に医師に相談してください。心臓ペースメーカーのノイズ耐性には大きな違いがあるため、一般原則を定めることはできません。
- 金属インプラントや金属部品、あるいは補聴器の使用者は、上記の領域に立ち入る前に医師に相談する必要があります。

3.3.5 高温部品への接触に対する保護

電気ドライブおよび制御システムのコンポーネントの高温面。火傷の危険があります。

- 制動抵抗器、ヒートシンク、電源ユニット、およびドライブ コントローラー、モーター、巻き線、およびラミネート加工をしたコアなどの高温面に触れないでください。
- 運転条件によっては、運転中または運転後に、表面温度が 60 °C 以上 (140 °F) に達することがあります。
- モーターの電源を切ってから、モーターに触れる前に、十分な時間放置して冷却させてください。冷却には**最長で 140 分**を要することがあります。技術データに指定されている熱時間定数のおよそ 5 倍の冷却時間が必要です。
- チョーク、電源ユニット、およびドライブ コントローラの電源を切った後、**15 分**待って冷却させてから触れるようにしてください。
- 安全グローブを装着するか、熱い表面に対する作業を行わないでください。
- 特定の用途では、機械または設備のメーカーが、各安全規定に従って、最終用途での火傷による傷害を避けるための対策を講じる必要があります。

電気ドライブおよび制御の安全の手引き

す。これらの対策には、機械または設備での警告、ガード（シールドまたはバリア）、またはアプリケーション関連文書での安全上の指示などがあります。

3.3.6 取り扱いおよび取り付け中の保護

不適切な取り扱いにより怪我をする危険があります。粉碎、せん断、切断、打撃によって怪我をすることがあります。

- 事故の防止に関連した法定の規則に従ってください。
- 取り付けおよび輸送には、適切な装置を使用してください。
- 適切な対策を講じて、故障が起こったり破砕しないようにしてください。
- 常に適切な工具を使用してください。指定されている場合は、特殊な工具を使用してください。
- つり上げ装置および工具を正しく使用してください。
- 適切な保護具（安全帽、安全ゴーグル、安全靴、安全グローブなど）を使用してください。
- つり上げられている荷物の下に立たないでください。
- 転倒を避けるため、床にこぼれた液体はすぐに拭き取ってください。

3.3.7 バッテリーの安全

バッテリーの堅い筐体の中に活性薬品が含まれています。そのため、不適切な扱いによって怪我や物的損害が生じることがあります。

不適切な取り扱いにより怪我をする危険があります。

- 電気残量が低くなったバッテリーを、加熱などによって再活性化しないでください（爆発や焼灼の危険があります）。
- 漏出や爆発の危険があるため、バッテリーを再充電しないでください。
- バッテリーを直火にかけないでください。
- バッテリーを分解しないでください。
- バッテリーを交換するときは、デバイスに取り付けられている電気部品を傷つけないようにしてください。
- 製品に指定されたバッテリータイプのみを使用してください。



環境保護および廃棄製品に含まれるバッテリーは、陸上輸送、空輸、および船舶輸送中には法的に危険な貨物（爆発の危険）と見なされます。使用済みのバッテリーは、他の廃棄物と分別して廃棄してください。自国の規制を遵守してください。

3.3.8 加圧システムに対する保護

『プロジェクト プランニング マニュアル』に記載されている情報に従って、液体および圧縮空気冷却されたモーターおよびコンポーネントに、圧縮空気、油圧オイル、冷却液、および冷却潤滑剤などの加圧媒体を部分的に外部から供給することも可能です。接続された供給システムや、供給ライン、接続を不適切に取り扱くと、怪我や物的損害を招くことがあります。

加圧ラインの不適切な取り扱いにより怪我をする危険があります。

- 加圧ラインを、分離、開封または切断しないでください（爆発の危険があります）。

電気ドライブおよび制御の安全の手引き

- 該当するメーカーの操作指示を順守してください。
- ラインを取り外す前に、圧力を抜いて、媒体を空にしてください。
- 適切な保護具 (安全ゴーグル、安全靴、安全グローブなど) を使用してください。
- 転倒防止のため、床にこぼれた液体はすぐに拭き取ってください。



環境保護および廃棄製品を運転するために使用される薬剤 (液体など) は、環境に有害である可能性があります。環境に有害な薬剤は、他の廃棄物と分別して廃棄してください。自国の規制を順守してください。

3.4 警告語および安全注意シンボルの説明

ご利用いただくアプリケーション関連文書内の安全の手引きには特定の警告語 (危険、警告、注意あるいは注記) および必要に応じて安全注意シンボルが使用されています (ANSI Z535.6-2011 に基づく)。

警告語には読み手の安全の手引きに対する注意を喚起し、危険度を特定する意味があります。

危険、警告、および注意といった警告語の前にある安全注意シンボル (感嘆符がついた三角マーク) は読み手に人身傷害の危険を警告します。

危険

この安全の手引きを遵守しない場合は、死亡や重度の傷害が発生します。

警告

この安全の手引きを遵守しない場合は、死亡や重度の障害が発生する可能性があります。

注意

この安全の手引きを遵守しない場合は軽度あるいは中度の傷害が発生する可能性があります。

注記

この安全の手引きを遵守しない場合は物的損害が発生する可能性があります。

はじめに

4 はじめに

GAT (Generic Application Template) (汎用アプリケーション テンプレート) を用いて、モーションおよびロジック アプリケーション向けに十分に構造化されたアプリケーション プログラムを短時間で開発してください。

GAT は、あらかじめ定義されたポイントにアプリケーション コードを入力して、すぐに実行することが可能なプログラミング フレームです。GAT ウィザードのダイアログベースのコード生成を使用して、準備された要素で十分に構造化された専門的なアプリケーション プログラムを作成してください。

アプリケーションのプログラミングおよび GAT の使用を始める前に、以下の章を読むことをお勧めします。

[Basic procedures for Automating Applications \(アプリケーション自動化の基本手順\)](#)、33 ページ、

アプリケーションの複雑さに応じて、以下のバージョンの GAT が選択可能です。

GAT バージョン	説明	利用可能なシステム
AxisInterface サンプル プロジェクト	Sample programm for the AxisInterface アプリケーション向けサンプル プログラム (簡易軸コマンド) 83 ページ "AxisInterface 用プログラム" 6.6.2 章	IndraLogic XLC IndraMotion MLC
GAT _{compact}	軸数の少ない単純なアプリケーション向けの非モジュール型プログラムの例 201 ページ "GAT _{compact} " 8 章 243 ページ "sercos インターフェースなしの IndraLogic XLC 用 GAT _{compact} " 9 章	IndraLogic XLC IndraMotion MLC
GAT (集中)	集中運転モード SFC を備えた柔軟なモジュール型機械コンセプト向けプログラム フレームワーク 263 ページ "集中状態の機械での GAT" 10 章	IndraMotion MLC
GAT (分散)	分散運転モード SFC を備えた柔軟なモジュール型機械コンセプト向けプログラム フレームワーク 331 ページ "分散状態の機械での GAT" 11 章	IndraMotion MLC

表 4-1: 利用可能な GAT バージョン

GAT および GAT_{compact} では、軸 (AxisInterface) およびコントロール (IndraMotion Control インターフェース) AxisInterface または IMC インターフェースの説明については、¹⁾ see 49 ページ "ML_TechInterface.library" 6 章。

GAT のすべてのバージョンは、IndraWorks の不可欠な構成要素です。

GAT ウィザード

GAT ウィザードで GAT 初期インポートを実行し、既存の GAT プロジェクトを編集します。GAT ウィザードのダイアログで、十分に構造化されたアプリケーション プログラムを短時間で作成します。

IndraWorks のツールバーにある GAT ボタン  を使用して GAT ウィザードを起動します。

¹⁾ *IndraMotionControl Interface* = 制御へのインターフェースを備えた IndraLogic ライブラリ。例: エラーをクリア

はじめに



GAT ウィザードは、常に IndraWorks プロジェクト内のアクティブなアプリケーションを参照します。

5 アプリケーションの自動化

5.1 一般情報

機械を自動化する際には、経験に基づいて以下のタスクを実行する必要があります。

- 初期化
- 運転モードの管理
- 周期的処理
- タスク処理/計算時間の割当
- 診断/エラー処理
- 表示

以下のセクションでは、これらのタスクを実行するための基本的な手順、およびモーションアプリケーションの詳細について説明します。コントロールとドライブ間の通信に関しては、sercos バスに特に焦点が当てられています。IndraWorks エンジニアリング フレームワークでは、タスクとエラー処理のほか、表示についても検討します。

5.2 初期化

機械が起動を始めると、運転の準備を整えるために、システム固有のコントロールおよび関連するドライブのほとんどのパラメータと変数が設定されます。この手順は「初期化」と呼ばれます。

初期化には異なる 2 つの種類があります。通信フェーズ 2 (P2、パラメータ化モード) および通信フェーズ 4 (BB、運転モード) です。

通信フェーズ 2 とは、コントロールとドライブがパラメータ化モードになっていることを意味します。このモードで書き込むことができる数多くのパラメータは、通信フェーズ 4 (運転モード) では書き込むことができなくなります。

通信フェーズ 2 では、以下が実行されます。

- 通常、通信パラメータがマスターからドライブに転送されます (フィールドバスおよび sercos デバイスの場合)
- 必要に応じて、"ドライブ パラメータの読み込み"および"ドライブ パラメータの保存" ("ファイル サービス") 機能が実行されます。

一般的に、機械を運転する準備を整えるために、初期化中に以下のアクティビティが実行されます。

- ドライブ ファームウェアをチェックする
- 既存のハードウェアをスキャンする
- 軸の設定を行う
- コントロールとドライブのパラメータのチェックおよび/または設定を行う
- PLC 変数/FB を設定/リセットする
- 初期化の最後に、軸にトルクを加える
- システム フェーズを動作モード (P4) に切り替える

アプリケーションの自動化

5.3 運転モード

5.3.1 一般情報

各アプリケーションの一般的な運転モードは、**Automatic** (自動) および **Manual** (手動) です。また、さらに別の運転モードがあることもあります (「クリーニング」など)。

通常、運転モードの変更は、モーションアプリケーションですべての軸が停止しているときにのみ行われます。

5.3.2 自動

モーションアプリケーションの「自動」運転モードでは、軸は一般に相互に連携して動きます。このように、1 つまたは複数のスレーブ軸がマスター軸に従って同じ速度と位置で動くことが少なくありません。また、スレーブ軸はアプリケーション固有の MotionProfile (カム) を旋回させることができます。軸はオペレータの関与なしに動作します。

複雑なアプリケーションで、より適切な管理または概要を実現するために、自動運転モードはしばしば複数の状態に細分化されます。

5.3.3 手動

「手動」運転モードは、機械のセットアップまたは調整を行うために使用します。一般的に、それぞれの軸は、特定の速度プロファイルまたは増分 (ジョグ) を使用して動かされます。また、軸を同期させて動かすこともできます。複雑さの程度に応じて、手動運転モードを複数の状態に細分化することもできます。

5.4 周期的処理

5.4.1 一般情報

運転モードに依存しないすべての自動化タスクは、それぞれのプログラム セクションで組み合わせる必要があります。その後、このセクションを周期的タスク内などで処理できます。

以下のセクションでは、周期的処理のための一般的な自動化タスクについて説明します。

5.4.2 運転モードから独立したロジック

通常、運転モードに依存しないロジックは別個のプログラム セクションで取り扱われます。一般に、このセクションは周期的に処理されます。運転モードに依存しないすべてのタスクは、この周期的タスク内にプログラムされます (自動モード中でのジョグを許可しないなど)。

5.4.3 診断メッセージの生成と表示

日常的なシステムの使用の際に、ユーザーはシステムのステータスに関する最新の情報を必要とします (現在の運転モード、温度、油圧など)。この必要な情報を提供するためには、情報を一元的に収集し、有意なステータス メッセージにする必要があります。これらのメッセージはシステム HMI などで表示するために提供されます。詳細については、[43 ページ "表示" 5.7 章](#) を参照してください。

5.5 タスク管理

5.5.1 一般情報

自動化のサブタスクの 1 つは、アプリケーションをタスクに実務的に分割することです。

同時に、システムへの負荷を評価するために、計算時間を計測する必要があります。

5.5.2 タスク管理の原則

以下の原則を常を守る必要があります。

- タスクは必要な数だけにし、できるだけ少なくする
 - マルチタスクによりシステムの複雑さが高まる
 - タスクの切替にはより多くの計算時間が必要となる
- PLC タスクは必ずウォッチドッグを用いて実行します (例外：自走タスク、バックグラウンドタスク)。そうしなければ、予期しないシステム挙動が発生する可能性があります。
 - オン：ウォッチドッグ時間=タスクのサイクル時間、感度=1
 - 通常：ウォッチドッグ時間=(タスクのサイクル時間+50~100%)、感度=2
 - アプリケーション プログラム内の早期警告としてのオプションのウォッチドッグ：時間計測機能を用いることにより、例えばシステムウォッチドッグの時間の90%でアプリケーション プログラムに"警告ウォッチドッグ"を実装できます。この方法により、警告を生成して、対策を実施することが可能になります。
- システムは以下の場合に最適な負荷がかかります。
 - 最小のコントロール変数をこのタスクに使用可能
 - コントロールのすべてのタスクが時間どおりに実行される
 - いずれの運転状態においても PLC タスク ウォッチドッグ (最悪のケース) が発生しない

5.5.3 IndraWorks/IndraLogic 使用時の推奨タスク分割

一般情報

プロセスにかかる時間が、設定されたウォッチドッグ時間を超え、このタスクのために IndraWorks/IndraLogic でウォッチドッグが起動した場合このタスクはエラー ステータス (例外エラー) で中止されます。

この場合、感度が以下のように計算されます。

- 複数回連続してウォッチドッグ時間を超過した場合
 - 感度が 1 または 2 であれば、サイクル 1 で例外エラー
 - 感度が 3 であれば、サイクル 2 で例外エラー
 - ...
 - 感度が n であれば、サイクル (n-1) で例外エラー
- ウォッチドッグ時間を 1 回超過した場合、現在のサイクル時間が (ウォッチドッグ時間×感度) より大きければ、例外エラーが発生します。

例：現在のサイクル時間 = 10 ms、感度 = 5 → タスクが 50 ms を 1 回超過すると例外エラー

アプリケーションの自動化

この情報に基づいて、以下のタスクおよびそれに関連する時間関係の設定をお勧めします。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
動作タスク (GAT ^{compact} で実行可能)	2	=動作のサイクル時間	T#2 ~ 10 ms	1..2
PlcTask	~5	T#10 ~ 50 ms	T#10 ~ 100 ms	2..3
BackgroundTask (GAT ^{compact} ではオプション)	~6	T#40 ~ 100 ms	-	-

表 5-1:

MotionTask の一般的なタスク

一般的に、動作タスクは特定のタスクのために予約されます。

- ガントリ軸および特別な要件のある軸の処理
- 特別な時間要件のある技術ファンクションブロックの処理 (ワインダー、レジスタ コントロール、SmartBelt、フライングシャー、...)
- 高速エラーリアクション
- 高速 I/O の処理

PlcTask の一般的なタスク

PlcTask はすべての標準タスクの中心的位置を占めます。

- 軸、運転モード切替の処理
- 機械のシーケンス
- 標準 I/O
- エラー処理
- 安全技術のサポート

BackgroundTask の一般的なタスク

オプションの BackgroundTask は、PlcTask の負荷を減らすために使用されます。

タイム クリティカルではないタスクはすべてここで処理できます。例えば、

- 診断メッセージの生成と表示
- 表示用データの準備
- 次の製品変更に備えるための複雑な計算 (カムなど)

5.5.4 サイクル時間の計測

タスク設定のサイクル時間ダイアログ

Cycle Times - IndraMotionMlc1

IndraMotionMlc1

Cycle time settings

Desired MC cycle time (T_{cyc}) us

SERCOS-Cycletime us

Sensitivity of the motion calculation

Motion-watchdog sensibility

Absolute error counter of the motion

Successive errors of the motion

Motion time display

	Minimum	Current	Maximum	
Actual MC cycle time (T _{cyc})		2000		us
Time for actual value acquisition	2	25	58	us
Pausing by motion-synchronous PLC tasks	0	29	88	us
Time for command value creation	0	7	18	us
Total time	2	62	125	us

図 5-1: モーションコントロールおよび PLC のためのサイクル時間の分割

アプリケーションの自動化

コントロールのタスク実行ビューワ

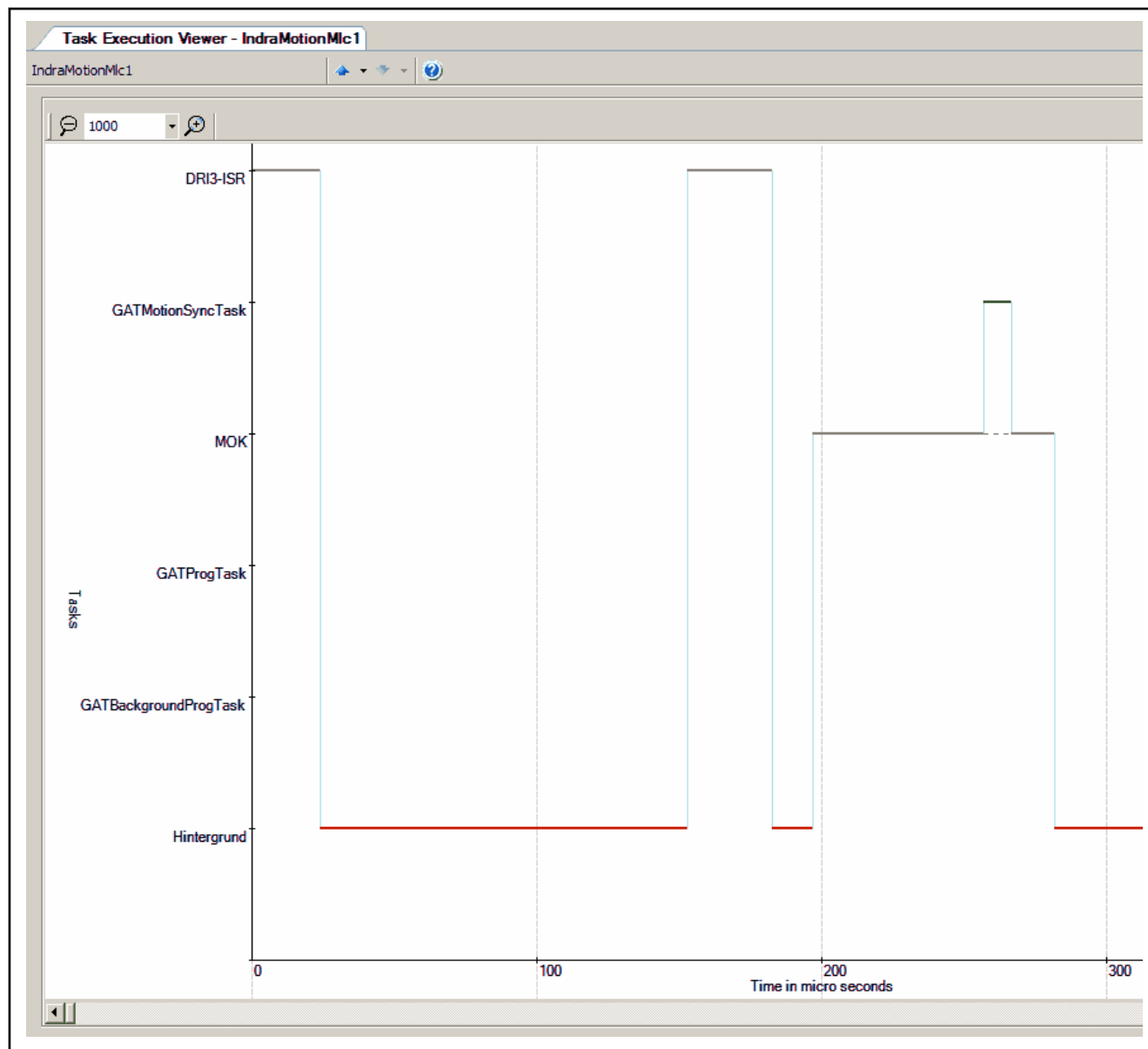


図 5-2: PLC および選択したシステム タスクの時間シーケンス

PLC プロジェクトにおける時間計測

個別のコード セクションを計測するために、コントロール システムは高分解能の時間計測機能を備えています。

- IL_HighResTimeTick および
- IL_HighResTimeDiff

RIL_UTILITIES ライブラリにて

PLC_load[IndraMotionMlc1: Logic: Application]

PLC task times (in uSec):

Minimum Time	Actual Time	MaximumTime
36	52	70

Clear values

TE_AxisInterfaceType01Prog[IndraMotionMlc1: Logic: Application]

IndraMotionMlc1.Application.TE_AxisInterfaceType01Prog

Expression	Comment	Type	Value
ClearError	clear internal errors of the program	BOOL	FALSE

```

90
91 udiStartTime[16#1D367869] := IL_HighResTimeTick(TRUE); // latch starting time
92
93 IF (InitDone[TRUE] AND bInitReferencesDone[TRUE]) THEN
94   FOR uiLoop[16#0011] := 1 TO uiMAX_AXIS_INDEX DO
95     fbAxis[uiLoop[16#0011]] ( ); // call FB, inputs are already set....
96   END_FOR
97 END_IF
98
99 udiStopTime[16#1D367891] := IL_HighResTimeTick(TRUE); // latch end time
100
101 // calculate actual runtime and store MIN and MAX
102 udiActCycleTime[16#00000028] := IL_HighResTimeDiff(udiStartTime[16#1D367869],udiStopTime[16#1D367891]);
103
104 stCyclesTime.udiActualTime[16#00000028] := udiActCycleTime[16#00000028];
105
106 IF (stCyclesTime.udiMaximumTime[16#00000046] < udiActCycleTime[16#00000028]) THEN
107   stCyclesTime.udiMaximumTime[16#00000046] := udiActCycleTime[16#00000028];
108 END_IF
109
110 IF (stCyclesTime.udiMinimumTime[16#00000024] > udiActCycleTime[16#00000028]) THEN
111   stCyclesTime.udiMinimumTime[16#00000024] := udiActCycleTime[16#00000028];
112 END_IF RETURN

```

図 5-3: PLC プロジェクトにおける時間計測

この時間計測機能を用いることにより、例えばシステム ウォッチドッグの時間の 90% でアプリケーション プログラムに "warning watchdog" (警告ウォッチドッグ) を実装できます。この方法により、機械の停止につながらないエラー/警告を実装できます。これとは対照的に、システム ウォッチドッグは PLC を停止させます。

5.5.5 アプリケーション固有のウォッチドッグ

IndraLogic タスク設定でウォッチドッグを入力した場合、そのウォッチドッグが応答すると、直ちに PLC プログラムが中止されます。この方法により、PLC プログラム内のアプリケーションで、このイベントに基づいた反応が発生しないようにすることができます。

アプリケーション固有のウォッチドッグ、すなわち内部サイクル時間計測を実装することにより、サイクル時間を超過した場合でも、PLC プログラム内で反応が起きるようにすることができます。このために使用する機能は 38 ページ "PLC プロジェクトにおける時間計測" 章 で説明されています。

アプリケーションの自動化

5.6 エラー処理

5.6.1 一般情報

エラー処理の分野では、工業用自動化システムのために特別な要件が設定されます。エラーが発生した場合の損傷が最小に抑えられることを確認する必要があります。さらに、ダウンタイムも最小に抑えなければなりません。これは、有意な診断および定義済みの再起動手順により実現できます。

以下のセクションでは、エラー処理についてより詳細に説明します。

5.6.2 エラー処理の原則

エラー処理は以下の4つの下位区分に分類されます。

1. エラー検出、ページ 40

考えられるすべてのエラーの発生元を調べる必要があります。エラー発生元の例としては、ドライブ (軸インターフェース)、コントロール (IMC インターフェース)、ファンクションブロックエラー、周辺機器のエラー (I/O、フィールドバス)、ユーザー定義エラー (材料の詰まりなど)、PLC エラー、その他があります。

2. エラーリアクション、ページ 41

エラーが発生した場合、損傷を最小に抑える必要があります。これは一般的に、定義済みのシャットダウンにより実現できます。いくつかのコンポーネント (ドライブなど) は予め定義済みのエラーリアクションがあります。また、ユーザー定義のエラーリアクションも必要なことが少なくありません (紙やフィルムが破れないようにするための同時シャットダウンなど)。

3. エラー診断、ページ 42

ダウンタイムを削減するために、できる限り早期にエラーの原因を判定する必要があります。分散システムの場合、診断は中心的な位置で提供される必要があります。さらに、メッセージにはできる限りプレーンテキストを使用する必要があります。場合によっては、表示されたエラーはエラーの原因を部分的にしか示さない余効です。この理由により、エラーの実際の原因を見つけ、エラーの経緯をよりの確に分析するための追加的な診断ツールとして、ログブックが非常に役立つことが少なくありません。

4. エラー確認/エラーの後の再起動、ページ 43

エラーの原因が解消されたら、できる限り迅速に生産モードを再確立する必要があります。エラー確認は、中心的な位置でのみ行ってください。また、アプリケーションではしばしば、エラーの発生しない生産モードを復元するために手順を踏む必要があります (ユーザー定義エラーのリセット、仮想マスター軸の始動など)。

5.6.3 エラー検出

一般的に、アプリケーション プログラムが生成される場合、以下のエラー発生元を調べ、中心的な位置にいる機械オペレータに結果を通知します。

アプリケーションの自動化

エラーの発生元	クエリ/検出
ドライブ	- AxisInterface なし : AxisData [AxisRef.AxisNo].Error Bit - AxisInterface あり : MB_AXIS_DIAGNOSIS.Error Bit
コントロール	- IMC インターフェースのない IndraLogic XLC/ IndraMotion MLC コントロール : ML_CONTROLDATA.C_0_0023.Bit26 - IMC インターフェースのない IndraMotion MLD コントロール : P-0-0115.Bit13 - IMC インターフェースのない IndraLogic XLC/ IndraMotion MLC および IndraMotion MLD コントロール : MB_IMC_DIAGNOSIS.Error
ファンクションブロック	呼び出された FB のエラー出力
周辺部	フィールドバス固有、例として library RIL_ProfibusDP_02.lib を参照
アプリケーション固有のウォッチドッグ	ユーザー定義

IMC インターフェース IndraMotionControl インターフェース = 制御へのインターフェースを備えた IndraLogic ライブラリ。
例：エラーをクリア

5.6.4 エラーリアクション

エラーの発生後、一般に機械および材料の損傷を防ぐためにエラーリアクションが起動します。エラーの発生元により、エラーリアクションが基本システムによって開始されます (下の表を参照)。また、補足的なユーザー固有のエラーリアクションをアプリケーション プログラムに実装できます (仮想マスター軸の定義済みシャットダウンなど)。

エラーの発生元	リアクション
ドライブ	- リアクションはエラーの深刻度によって異なります。 - F0 ~ F2 : 実現可能な最善のシャットダウン -> 設定されたパラメータ化に基づいて - F6 ~ F7 : 速度コマンド値、ゼロに設定 - F8 : トルク有効 - エラーの影響を受けた軸のみシャットダウン - アプリケーション プログラムは続行 注： 致命的でないドライブ エラーへの反応は設定可能です (『"Rexroth IndraMotion MLC xxVRS 機能説明"』の「"エラーリアクション"」の章も参照) xx - エディション

アプリケーションの自動化

エラーの発生元	リアクション
コントロール	- IndraLogic XLC/IndraMotion MLC : すべての軸をシャットダウン -> 減速勾配 A-0-0224 (緊急減速) によるフェイルセーフ シャットダウン - IndraMotion MLD : 反応は予めパラメータ化が可能 - あらゆるケースでアプリケーション プログラムは続行しない - sercos 中断 : アプリケーション プログラムは続行 - 動作カーネルのサイクル超過 : アプリケーション プログラムは続行しない
ファンクションブロック	- FB 資料に基づいたさまざまな反応 - アプリケーション プログラムは続行
周辺部	- 反応なし - アプリケーション プログラムは続行
アプリケーション固有のウォッチドッグ	- 反応はユーザー固有 - アプリケーション プログラムは続行

5.6.5 エラー診断

エラーの原因を判別するために、コントロール システムのコンポーネントは総合的な診断メッセージを提供します (下表を参照)。このため、ドライブおよびコントロールについてのエラー メッセージには、深刻度も含まれます。一般に、エラーは当初、最も高い深刻度で示されます。迅速な診断を確実にを行うために、診断メッセージは中心的な位置でまとめられ、機械オペレータに提供されます。

エラーの発生元	エラーの診断
ドライブ	- AxisInterface なし : A-0-0023 (XLC/MLC の場合のみ)、S-0-0095、S-0-0390 - AxisInterface あり : MB_AXIS_DIAGNOSIS.Message
コントロール	- IMC インターフェースのない IndraLogic XLC/IndraMotion MLC - C-0-0650、C-0-0627 - IMC インターフェースのない IndraMotion MLD - P-0-0115、P1351 - IMC インターフェースを備えた IndraMotion MLC および IndraMotion MLD - MB_IMC_DIAGNOSIS.Number - MB_IMC_DIAGNOSIS.Message
ファンクションブロック	ErrorID、ErrorIdent は FB 出力にて

アプリケーションの自動化

エラーの発生元	エラーの診断
周辺部	フィールド バス/インライン資料を参照
アプリケーション固有のウォッチドッグ	アプリケーション固有のウォッチドッグ

IMC インターフェース IndraMotionControl インターフェース = 制御へのインターフェースを備えた IndraLogic ライブラリ。
例：エラーをクリア

5.6.6 エラー確認/エラーの後の再起動

エラーの原因が解消された後で、エラーの発生元でエラーを確認する必要があります。ユーザーの視点からは、**中央エラーのクリア**が必要です。このため、エラー確認は中心的な位置からアプリケーション プログラムのエラーの原因に対して実行する必要があります。下表は、各サブコンポーネントでエラーをクリアする方法を示したものです。

エラーの発生元	エラー確認
ドライブ	- AxisInterface なし： ファンクションブロック MC_Reset - AxisInterface あり： MB_AXIS_ADMINISTRATION.ClearError
コントロール	- IMC インターフェースのない IndraLogic XLC/ IndraMotion MLC - 開始コマンド C-0-1030 (MB_Command を使用) - IMC インターフェースのない IndraMotion MLD - AxisRef = ローカル軸で MC_Reset ファンクションブロック - IMC インターフェースを備えた IndraMotion MLC および IndraMotion MLD - MB_IMC_ADMINISTRATION.ClearError
ファンクションブロック	実行/有効化 FB 入力で立下り
周辺部	フィールド バス/インライン資料を参照
アプリケーション固有のウォッチドッグ	アプリケーション固有のウォッチドッグ

IMC インターフェース IndraMotionControl インターフェース = 制御へのインターフェースを備えた IndraLogic ライブラリ。
例：エラーをクリア

5.7 表示

5.7.1 一般情報

表示では、機械の運転に必要なデータを提供するだけでなく、エラー診断のためのサポートも提供する必要があります。

5.7.2 表示の一般的な内容

機械ステータスの表示

- 有効な運転モード

アプリケーションの自動化

- 現在の診断/診断履歴 ([エラー検出](#)、[ページ 40](#) を参照)
- 試運転中のサポートのための詳細情報、例えば、軸の状態 (位置、速度、ステータスなど)
- アプリケーション固有の機械データ

機械の状態の制御

- 運転モード (自動、手動、...) の切替
- 既存エラーの確認
- 操作画面の切替
- アプリケーション固有の機械機能の操作

5.7.3 PLC を使用したデータ提供

一般情報

表示されるデータは、記号ファイルに入力する必要があります。

記号設定

変数は以下のアクセス権限を使用して設定できます。

- アクセス権なし
- 読み取り
- 書き込み
- 読み書き

記号設定は、"Symbol configuration" (記号設定) ウィンドウのダイアログで、POU の変数宣言エリアにある "Pragma directives" (プラグマ ディレクティブ) を使用して、または変数リストを用いて実行できます。

記号設定の前提条件は、PLC オブジェクトである "Symbol configuration" (記号設定) が IndraWorks プロジェクト ツリー内の PLC アプリケーションより下にあることです。これを行うには、ドラッグ & ドロップを使用して "PLC objects / General objects" (PLC オブジェクト/一般オブジェクト ライブラリ) ウィンドウからオブジェクトを移動し、"Logic / Application" (ロジック/アプリケーション) デバイス ノードの下に追加します。

アプリケーションの自動化

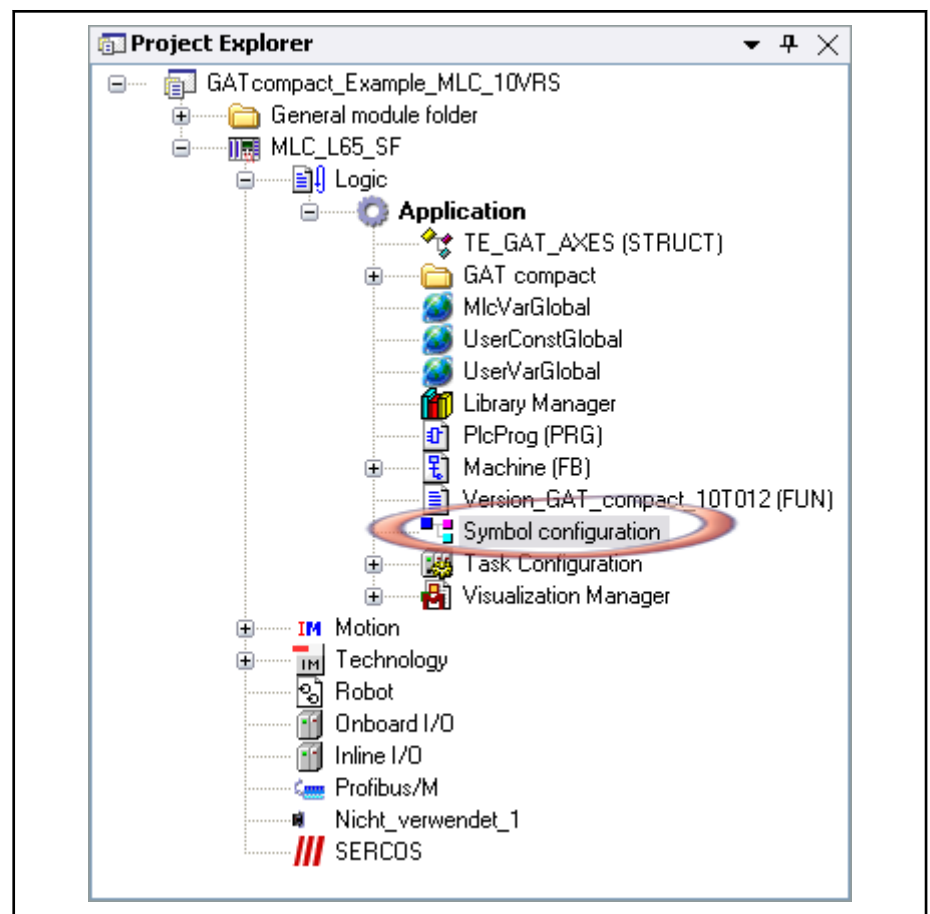


図 5-4: プロジェクト ツリー内の記号設定

ダイアログを使用した記号設定

記号設定のダイアログは、IndraWorks プロジェクト エクスプローラにある新しい "Symbol configuration" (記号設定) ノードを使用して開くことができます。

記号設定ウィンドウは 2 つのエリアに分かれています。

使用可能な変数	PLC プログラムのすべての変数と、それぞれの最大のアクセス権限レベルのリスト。リストに変数がない場合は、"Refresh" (更新) リンクを使用してリストを更新します。エラーなしでプロジェクトをコンパイルすることが可能でなければならないことをご注意ください。
選択された変数	記号アクセスのために設定されたすべての変数と、それぞれに設定されたアクセス権限のリスト。PLC プロジェクトにもはや存在しない変数をすべて強調表示するには、"Display invalid variables" (無効な変数を表示する) リンクを選択します。

変数を選択または削除するには、その要素をダブルクリックするか、または > および < ボタンを使用します。これにより、選択した要素およびすべての下位要素が選択または削除されます。

アプリケーションの自動化

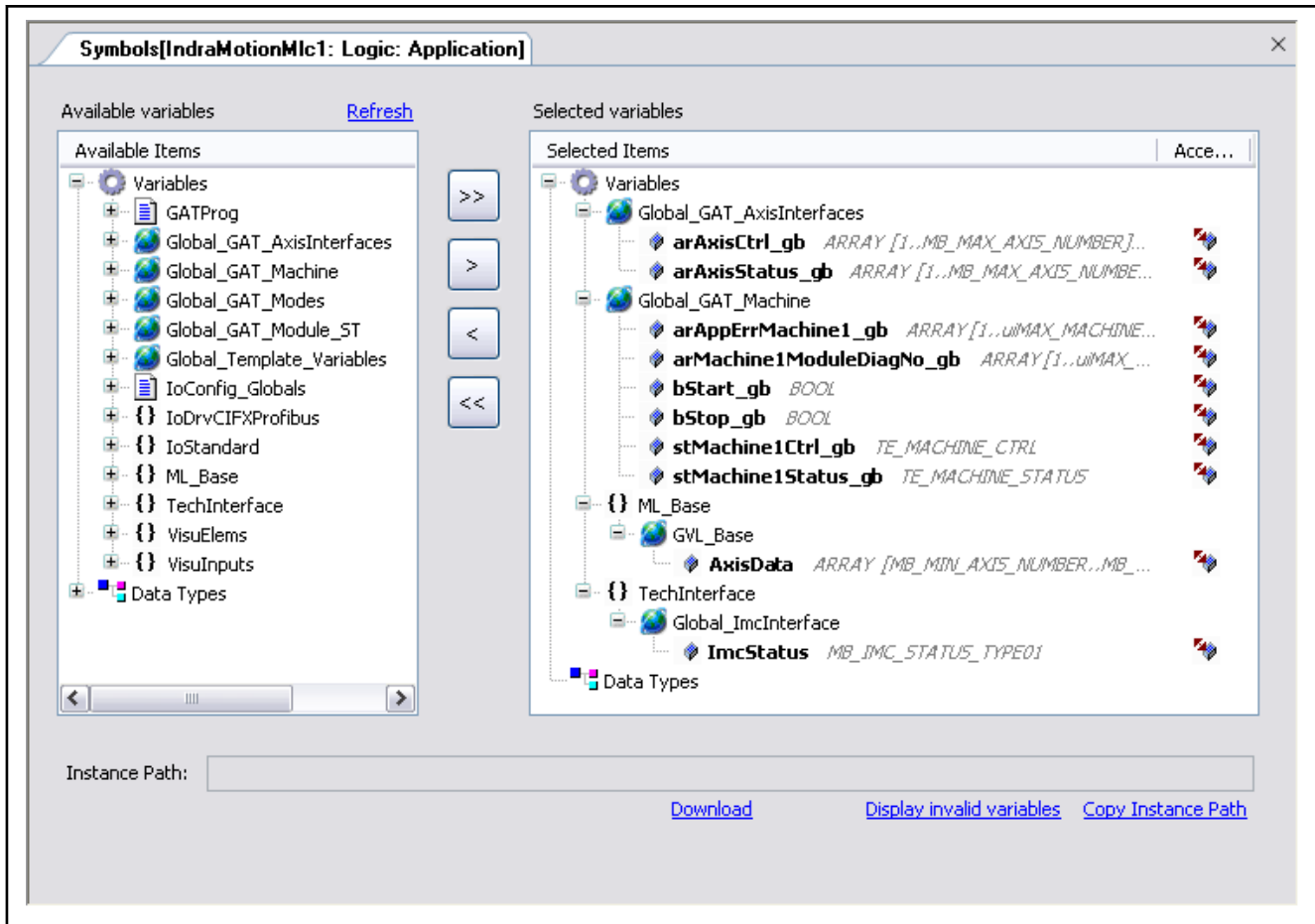


図 5-5: ダイアログを使用した記号設定

各変数のアクセス権は、"Selected variables" (選択された変数) セクションの "Access rights" (アクセス権) 列にある記号をクリックして変更できます。

アクセス権は以下の順序で切り替えられます。

-  読み書きアクセス権
-  書き込みアクセス権
-  読み取りアクセス権



ライブラリから取得したグローバル変数リストの変数の記号エントリは、各ライブラリのバージョンが変更された後で記号設定に挿入しなおす必要があります。

プラグマを使用した記号設定

"プラグマ ディレクティブ"を使用した記号設定は、変数リストや POU の中など、変数の宣言エリアで直接実行されます。

プラグマ設定では、アクセス権限のパラメータを伴う "{attribute 'symbol'}" 属性が使用できます。使用可能な設定は下の表に示されています。

アプリケーションの自動化

プラグマ ディレクティブ	説明
{attribute 'symbol' := 'none'}	変数へのアクセスはブロックされ、BR-WS ドライバの閲覧リストに表示されません。
{attribute 'symbol' := 'read'}	変数へのアクセスは読み取りアクセスのみが許可され、BR-WS ドライバの閲覧リストに表示されます。
{attribute 'symbol' := 'write'}	変数へのアクセスは書き込みアクセスのみが許可され、BR-WS ドライバの閲覧リストに表示されます。
{attribute 'symbol'}	変数へのアクセスは読み書きアクセスが許可され、BR-WS ドライバの閲覧リストに表示されます。
{attribute 'symbol' := 'readwrite'}	

BR-WS ドライバ Bosch Rexroth 製 WinStudio 通信ドライバ

プラグマが書き込まれる位置または行により、宣言エリア内のすべての変数について有効が、1つの変数についてのみ有効かが決まります。

- プラグマが1つの変数のみに影響を及ぼすためには、変数の直前の行にプラグマが書き込まれる必要があります。
- プラグマが宣言エリア内のすべての変数に影響を及ぼすためには、宣言リストの最初の行、または宣言エリアの直前の行 (VAR GLOBAL... END_VAR の直前など) にプラグマがなければなりません。この場合も、個別の変数に異なるアクセス権限を割り当てることは可能です。

PLC アプリケーションをダウンロードすると、記号情報が XML ファイルとして IndraWorks "IndraLogic" プロジェクトのサブディレクトリに作成され、記号アプリケーションとしてコントロールにロードされます。

通信設定

- WinStudio アプリケーションとコントロール間の通信は、BR-WS ドライバを使用して行われます。これは、記号ファイルから変数を表示するために使用できます。
- 通信性能を最適化するには、異なる間隔で更新されるさまざまなドライバシートに変数を分散させます。

6 ML_TechInterface.library

6.1 導入と概要

TechInterface は次のコンポーネントで構成されています。

- AxisInterface および
- IMC インターフェース (IMC = IndraMotion コントロール)

AxisInterface は PLCopen モーションファンクションブロックをプールして拡張し、操作しやすいインターフェースをドライブ機能に提供します。コードやコマンドが少なく、パフォーマンスが高ければ、アプリケーションプログラムの開発が加速されます。AxisInterface には、マスター軸とスレーブ軸のさまざまな運転モードの制御信号およびパラメータのほか、選択されたプロセスの値の設定オプションが含まれています。

IMC インターフェースは操作しやすいインターフェースをコントロール機能に提供します。IMC インターフェースには、コントロールの診断と管理のための制御信号およびパラメータが含まれています。

TechInterface は IndraMotion MLC コントロール システムと IndraMotion MLD ドライブ システムの両方によりサポートされています。

下の表は、TechInterface 向けのファンクションブロックおよびデータ構造を含む、サポートされているターゲット システムとライブラリをまとめたものです。

システム サポート	ドライブ ファームウェア サポート	ライブラリ
IndraMotion MLC 10VRS 以上	MPx04VRS 以上	ML_TechInterface.compiled-library
IndraMotion MLD ドライブ システム 07VRS 以上 資料 R911322053 "Rexroth IndraMotion AxisInterface for IndraLogic 1G" を参照	MPx07VRS 以上	MX_AxisInterface.lib MX_ImcInterface.lib

表 6-1: TechInterface ライブラリ

TechInterface ファンクションブロック

ファンクションブロック	説明
MB_AxisInitType01 (59 ページ "MB_AxisInitType01" 6.3.1 章 を参照)	単一軸について AxisInterface を初期化するために使用します。このファンクションブロックは、プログラムを起動するときに 1 回のみ、またはパラメータ化モードから運転モードにフェーズが切り替えられるたびに呼び出す必要があります。
MB_AxisInterfaceType01 (62 ページ "MB_AxisInterfaceType01" 6.3.2 章 を参照)	単一軸の AxisInterface を設定するために使用します。このファンクションブロックは運転モードが有効である間、周期的に呼び出す必要があります (モーションサイクルで、またはモーションサイクルより低速で)
MB_AxisInterfaceType02 (65 ページ "MB_AxisInterfaceType02" 6.3.3 章 を参照)	キネマティクスで協調運動を実行する軸について AxisInterface を設定するために使用します。 (Robot Control)
MB_GetAxisInterfaceIndex (機能)	転送された軸参照からの AxisInterface で使用されるマッチング指数を計算します
MB_ImcInterface (プログラム)	制御ステータス メッセージをトリガーして作成する機能

表 6-2: TechInterface



AxisInterface は IndraMotion プログラミング テンプレートとして、または軸機能向けの独立したインターフェースとして提供されます。

GAT プログラミング テンプレート (汎用アプリケーション テンプレート) または GAT *compact* と共に使用する場合、ユーザーはプロジェクト内の FB のインスタンス呼出を処理する必要がありません。この機能はテンプレートに完全に組み込まれています。ユーザーは数行のコードを書き込む必要があるだけです。

しかし、独立した機能として AxisInterface を使用する場合、ユーザーは各軸について両方の FB のインスタンスを作成する必要があります。このセクションには、独立した機能として AxisInterface を使用方法に関する説明と例が含まれています。

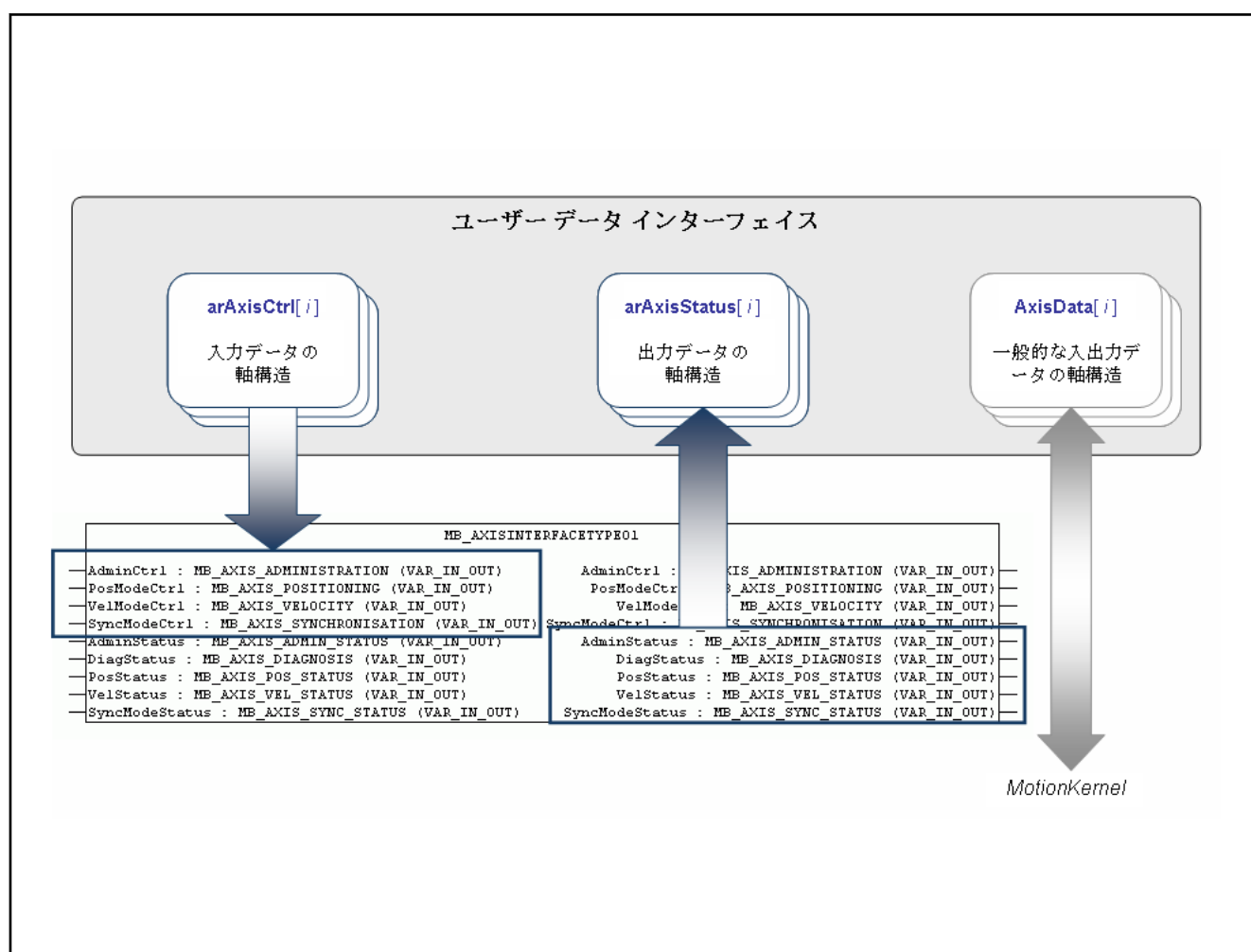


図 6-1: インターフェースの AxisInterface データ構造

下の表は、AxisInterface データ構造のユーザー インターフェースを示したものです。

ユーザー インターフェース	タイプ	説明
arAxisCtrl[]_gb	MB_AXIS_CONTROL_TYPE01 (MB_AXIS_CONTROL_TYPE02)	運転モードを有効にするためのコマンド値や変数などの制御構造
arAxisStatus[]_gb	MB_AXIS_STATUS_TYPE01 (MB_AXIS_STATUS_TYPE02)	運転モードの診断情報や確認などのステータス構造
AxisData[]		実際の値、ステータス ビットおよび周期的に設定可能なユーザー データ (ファームウェアで MotionControl により処理される) へのアクセスを含む、ステータス構造および制御構造
ImcCtrl[]	MB_IMC_CONTROL_TYPE01	通信モードの仕様および診断処理を含む、制御構造
ImcStatus[]	MB_IMC_STATUS_TYPE01	通信モードのステータスおよび診断情報を含む、ステータス構造

表 6-3: TechInterface – ユーザー インターフェースのデータ構造

FB の名称およびその構造での TYPE01 および TYPE02 の指定は、機能範囲の相違を示します。



さまざまな AxisInterface バージョン "02VRS" (MLC 04VRS) および TechInterface "10VRS" (MLC10VRS) ならびにより新しいバージョンの間で変更が実施され、その一部はユーザー プログラムでの適合を必要とします。必要となる適合は[56 ページ "使用開始時の TechInterface 設定" 6.2 章](#)で説明されています。

ML_TechInterface.library

Ausdruck	Datentyp	Wert	Ausdruck	Datentyp	Wert
IndraMotionMlc1.Application.arAxisCtrl_gb[4]	TE_AXIS_CONTROL...		IndraMotionMlc1.Application.AxisData[4]	ML_AXISDA...	
Admin	TE_AXIS_ADMINIST...		wUserCmdDataBitA_q	WORD	0
Axis	AXIS_REF		wUserCmdDataBitB_q	WORD	0
DiagMbrRefreshTime	TIME	T#2s	wUserCmdDataBitC_q	WORD	0
StopDeceleration	REAL	99	wUserCmdDataBitD_q	WORD	0
WaitTimePowerRestart	UINT	0	dwUserCmdDataA_q	SM_TYPES	
ClearError	BOOL	FALSE	dwUserCmdDataB_q	SM_TYPES	
SetAbsRef	BOOL	FALSE	dwUserCmdDataC_q	SM_TYPES	
EnableCyclicScanning	BOOL	TRUE	dwUserCmdDataD_q	SM_TYPES	
UpdateEveryInput	BOOL	FALSE	dwUserCmdDataE_q	SM_TYPES	
DriveInterruptReaction	BOOL	TRUE	dwUserCmdDataF_q	SM_TYPES	
PowerOn	BOOL	TRUE	dwMotionProfileStatusBits_j	DWORD	0
_OpMode	MB_AXIS_MODE_UN...		dwFlexProfileEventControlBits_q	DWORD	0
en	MB_AXIS_MODE	ModeAH	FlexProfil_Sync	BOOL	FALSE
b	MB_AXIS_MODE_BITS		FlexProfil_Active	BOOL	FALSE
RetriggerOpMode	BOOL	FALSE	FlexProfil_WaitForSwitching	BOOL	FALSE
PosMode	TE_AXIS_POSITIONI...		FlexProfil_ExeSingleFinished	BOOL	FALSE
VelMode	TE_AXIS_VELOCITY		FlexProfil_Set0_Checked	BOOL	FALSE
SyncMode	TE_AXIS_SYNCHRO...		FlexProfil_Set0_Ok	BOOL	FALSE
SetupMode	TE_AXIS_SETUP_MO...		FlexProfil_Set1_Checked	BOOL	FALSE
IndraMotionMlc1.Application.arAxisStatus_g...	TE_AXIS_STATUS_T...		FlexProfil_Set1_Ok	BOOL	FALSE
PtrAxisData	POINTER TO ML_AX...	16#0A9233F8	FlexProfil_Set2_Checked	BOOL	FALSE
Admin	TE_AXIS_ADMIN_ST...		FlexProfil_Set2_Ok	BOOL	FALSE
Active	BOOL	TRUE	FlexProfil_Set3_Checked	BOOL	FALSE
Name	STRING(40)	'MyRealAxis3'	FlexProfil_Set3_Ok	BOOL	FALSE
CmdDone	BOOL	FALSE	FlexProfil_EvtSet0_Checked	BOOL	FALSE
AxisType	MB_AXIS_CONFIG	RealAxisCfg	FlexProfil_EvtSet0_Ok	BOOL	FALSE
AbsRefCmdStatus	MB_SERCOS_DATA_...	MB_SERCOS_D...	FlexProfil_EvtSet1_Checked	BOOL	FALSE
InternalFlags	DWORD	5	FlexProfil_EvtSet1_Ok	BOOL	FALSE
_OpModeAck	MB_AXIS_MODE	ModeAH	FlexProfil_EvtSet2_Checked	BOOL	FALSE
MODE_AH	BOOL	TRUE	FlexProfil_EvtSet2_Ok	BOOL	FALSE
MODE_HOMING	BOOL	FALSE	FlexProfil_EvtSet3_Checked	BOOL	FALSE
MODE_POS_ABS	BOOL	FALSE	FlexProfil_EvtSet3_Ok	BOOL	FALSE
MODE_POS_REL	BOOL	FALSE	dwFlexProfileEventStatusBits_j	DWORD	0
MODE_POS_ADD	BOOL	FALSE	Axis_CamTab_0	BOOL	FALSE
MODE_VEL	BOOL	FALSE	Axis_CamTab_1	BOOL	FALSE
MODE_SYNC_PHASE	BOOL	FALSE	Axis_CamTab_2	BOOL	FALSE
MODE_SYNC_VEL	BOOL	FALSE	Axis_CamSwitching	BOOL	FALSE
MODE_SYNC_CAM	BOOL	FALSE	Axis_InVelocity	BOOL	TRUE
MODE_SYNC_PROFILE	BOOL	FALSE	Axis_Standstill	BOOL	TRUE
MODE_FLEX_PROFILE	BOOL	FALSE	Axis_InPosition	BOOL	FALSE
MODE_STOPPING	BOOL	FALSE	Axis_InSynchron	BOOL	FALSE
MODE_INTERRUPTED	BOOL	FALSE	Axis_Warning	BOOL	FALSE
MODE_EXTERNAL_FB	BOOL	FALSE	Axis_Error	BOOL	FALSE
Diag	TE_AXIS_DIAGNOSIS		Axis_Homed	BOOL	TRUE
PosMode	TE_AXIS_POS_STAT...		Axis_InTorque	BOOL	FALSE
VelMode	TE_AXIS_VEL_STATUS		Axis_OperationMode	BOOL	TRUE
SyncMode	TE_AXIS_SYNC_STA...		Axis_Inbb	BOOL	TRUE
SetupMode	TE_AXIS_SETUP_MO...		Axis_InAb	BOOL	TRUE
AxisData_HMI	TE_AXIS_DATA_STA...		Axis_Power	BOOL	TRUE
			Axis_CmdActive	BOOL	FALSE

図 6-2: AxisInterface データ構造の概要



aarAxisCtrl_gb[MyVirtualAxis.AxisNo].Admin などの列のインデックスとして、AxisNo of the AXIS_REF 構造を使用します。

EnableCyclicScanning

特定のコマンド値の内部処理は arAxisCtrl_gb[].Admin.EnableCyclicScanning 要素によって制御できます。"EnableCyclicScanning" が TRUE に設定されている場合、arAxisCtrl_gb[] 構造のコマンド値の一部は周期的にスキャンされ、値が変わると、直ちに適用されます。

Cyclic scanned	NOT Cyclic scanned
- IndraMotionMct1.Application.arAxisCtrl_obj[1] - Admin - Axis - DiagnoseRefreshTime - StopDeceleration REAL 99 - WaitTimePowerRestart UNIT 0 - ClearError BOOL FALSE - SetAbsRef BOOL FALSE - EnableCyclicScanning BOOL TRUE - UpdateEveryInput BOOL FALSE - DriveInterruptReaction BOOL TRUE - PowerOn BOOL TRUE - OptMode MB_AXIS_MODE_UN... ModeAt - en MB_AXIS_MODE - b MB_AXIS_MODE_BITS - MODE_AH BOOL FALSE - MODE_HOMING BOOL FALSE - MODE_POS_ABS BOOL FALSE - MODE_POS_REL BOOL FALSE - MODE_POS_ADD BOOL FALSE - MODE_VEL BOOL FALSE - MODE_SYNC_PHASE BOOL FALSE - MODE_SYNC_VEL BOOL FALSE - MODE_SYNC_CAM BOOL FALSE - MODE_SYNC_PROFILE BOOL FALSE - MODE_FLEX_PROFILE BOOL FALSE - MODE_STOPPING BOOL FALSE - MODE_INTERRUPTED BOOL FALSE - MODE_EXTERNAL_FB BOOL FALSE - RetriggerOptMode BOOL FALSE - PosMode TE_AXIS_POSITIONI... - Position REAL 0 - Distance REAL 10 - Velocity REAL 10 - Acceleration REAL 10 - Deceleration REAL 10 - Adjust MB_AXIS_ADJUST_D...	- VelMode TE_AXIS_VELOCITY - Velocity REAL 0 - Acceleration REAL 10 - Deceleration REAL 10 - Adjust MB_AXIS_ADJUST_D... - SyncMode TE_AXIS_SYNCRO... - InputRevolution UNIT 1 - OutputRevolution UNIT 1 - FineAdjust REAL 0 - CanShaftDistance REAL 360 - PhaseOffset REAL 0 - PhaseOffsetVel REAL 1 - PhaseOffsetAcc REAL 100 - SyncDirection MC_SYNC_DIRECTION SHORTCUTS - StartMode MC_START_MODE ABSOLUTE - CanTableID MC_CAM_ID CAM_TAB - Master AXIS_REF - Profile ML_AXIS_PROFILE - SetSelection UNIT 0 - RelativePositioning BOOL FALSE - UseSwitchingPosition BOOL FALSE - SwitchingPosition REAL 0 - ProfileEntry MB_PROFILE_START SLAVE_O - MasterOffset REAL 0 - SlaveOffset REAL 0 - SyncType MB_SYNC_TYPE SYNC_RA - ExecutionMode MB_EXECUTION_NO_EXECUTE - AdFineAdjust MB_AXIS_ADJUST_D... - AdCamShaftDistance MB_AXIS_ADJUST_D... - AdPhaseOffset MB_AXIS_ADJUST_D...

図 6-3: 周期的にスキャンされる `arAxisCtrl_gbf[]` の要素

UpdateEveryInput

上記のデータは、周期的にスキャンされてはいませんが、"UpdateEveryInput" 入力が設定され、周期的にスキャンされる要素に変更があれば、同様に受け入れられます。

例：運転モード "relative positioning" (相対的位置設定) が、上に示された値、速度=10 および加速=10 で起動されました。次の位置設定操作を異なる加速で開始するには、入力 "PosMode. Acceleration" を 10 から 20 に変更し、次の距離を適宜書き込みます。距離の変更と共に加速も受け入れられます。

- 
- 運転モード (.Admin_OpMode) が設定されている場合でも、"EnableCyclicScanning" 入力の設定にかかわらず、すべての入力データが読み取られます。
 - "EnableCyclicScanning" = TRUE である場合、グリーンでハイライトされたすべての入力データは周期的に読み取られます。これは、運転モードが有効化された後で、値に何らかの変更があれば、直ちに読み取られることを意味します。
 - 一方、ブルーでハイライトされたすべての入力データは、周期的には読み取られません。これは、運転モードが有効化された場合のみ、値が読み取られることを意味します。
 - "UpdateEveryInput" = TRUE である場合、ブルーでハイライトされたすべての入力データは、周期的にスキャンされる運転モードの要素のいずれかに変更があった場合、転送も行われます。
 - データの一貫性は、"EnableCyclicScanning" (FALSE → データを書き込む → TRUE) により実現されます。

このセクションで説明されている 5 つの構造要素に基づいたさまざまな IndraLogic 表示が、試運転の目的のためにサンプル プロジェクトで利用可能です (詳細については、[84 ページ "AxisInterface の表示" 6.6.3 章](#) を参照)。

バージョン 10VRS (MLC10VRS) の新機能

- TechInterface への AxisInterface と IMC interface の統合
IndraLogic 2G に加えられた変更の流れの中で、ライブラリ ML_AxisInterface.lib および ML_ImcInterface.lib が、1つのライブラリ ML_TechInterface.compiled-library にまとめられました。
- UNION による運転モードの選択：

ML_TechInterface.library

IndraLogic 2G では、列挙データ タイプ "ENUMERATION" を介して合併データ タイプ "UNION" を使用し、また新たなデータ タイプ "BIT" を介して バイナリ 形式で、運転モードの選択 `arAxisCtrl[MyRealAxis1.Axisno].Admin.OpMode` が実装されました。IndraLogic-1G では、"bitaccess" を使用して同様の機能が提供されていました。しかし、これは欠陥のある割当を発生させます。

- 新しい入力 `arAxisCtrl_gb[].Admin.UpdateEveryInput` : この入力は、運転モードが有効である間に周期的にスキャンされない入力の変更にも使用できます。
- 新しい運転モード "ModeStopping" : 運転モード "ModeStopping" は PLCopen 状態の機械を "Stopping" (停止) 状態に保ちます。すなわち、軸インターフェースの外部で呼び出された PLCopen ファンクションブロックはエラー "Command in state Stopping denied" (状態 Stopping のコマンドは拒否されました) によりキャンセルされます。運転モード "ModeAH" により、軸が PLCopen 状態 "StandStill" に転送されるようになります。すなわち、軸インターフェースの外部で呼び出された PLCopen ファンクションブロックが受け入れられ、軸インターフェースは "ModeExternalFB" を返します。
- 新しい出力 `arAxisStatus_gb[].Admin.AbsRefCmdStatus` : この出力は、`arAxisCtrl_gb[].Admin.SetAbsRef` を使用して開始されたコマンドのステータスを確認するために使用できます。
- 新しい出力 `ImcStatus.PassiveMode` : この出力は、例えば IndraWorks を使用してフェーズ切替が開始されたときに IMC インターフェースがバイパスされると、設定されます。切替が、`ImcCtrl.Admin.OpMode` で設定されているフェーズに再び戻ると、直ちに出力が再度削除されます。さらに、"PassiveMode" 出力が設定されている場合は、入力 `ImcCtrl.Admin.RetriggerOpMode` を使用してターゲットのフェーズに切り替えることができます。
- 新しい軸タイプ "controller axis" のサポート : MLC10VRS で導入された軸タイプ "controller axis" (コントローラ軸) は `arAxisStatus_gb[].Admin.AxisType = ControlAxisCfg (16#0080)` として認識され、可能な限りサポートされます。
- IndraLogic 2G のオブジェクト指向言語拡張により一段と使いやすくなったユーザー拡張 :

`arAxisCtrl_gb[]` および `arAxisStatus_gb[]` の下位構造も拡張が可能になりました (97 ページ "AxisInterface のユーザー拡張" 6.7 章も参照)

バージョン 11VRS (XLC11VRS/
MLC11VRS) の新機能

- 状態 "Axis_Interrupted" のモニタリング : `AxisInterface` はビット `"AxisData[].Axis_Interrupted"` をモニタリングし、`AxisStatus` が TRUE のとき、"ModeInterrupted" に切り替わります。これにより、軸インターフェースはドライブによって、例えばドライブの安全技術によってコントロールが引き継がれたことを報告します。

このモニタリングは、新しい入力 `"arAxisCtrl_gb[].Admin.DriveInterruptReaction"` をリセットすることにより無効にできます。デフォルト設定では、モニタリングは有効です。

また、`AxisInterrupted` のモニタリングは以前のバージョン (MLC10V08 以上) でも利用可能です。

- `arAxisCtrl_gb[].SyncMode.SyncDirection` (XLC/MLC11V04 以上) の周期的スキャン : `"SyncMode.PhaseOffset"` の調整を行うと、ドライブに補間がある場合にセット `"SyncMode.SyncDirection"` が有効になります。材料ウェブのある機械では、同期モードが有効になると、正の方向に同期化が起こります。その後の `"SyncMode.PhaseOffset"` の調整は、最も短い距離にしてください。これにより、運転モードが有効になったときに `"SyncMode.SyncDirection"` の変更が可能になります。

- AxisInterface は、軸のタイプとメッセージのタイプによって、各ドライブまたはコントロールのパラメータを読み込むようになりました。

ドライブに補間がある IndraDrive :

- 肯定応答：ドライブ パラメータが読み取られる。例："A0012 Ready for operation (運転準備完了)"
- ドライブのエラー：ドライブ パラメータが読み取られる。例："F2026 No power (電源なし)"
- コントロールでのエラー：コントロール パラメータが読み取られる。例："F2229233 Invalid master axis (無効なマスター軸)"

コントロールに補間がある IndraDrive :

- 肯定応答 A-0-0020/A-0-0023 が読み取られる。例："Synchronized Motion" (同期された動作)
- ドライブのエラー：ドライブ パラメータが読み取られる。例："F2026 No power (電源なし)"
- コントロールでのエラー：コントロール パラメータが読み取られる。例："F2229233 Invalid master axis (無効なマスター軸)"

SercosDrive と仮想軸 :

- コントロール パラメータが読み取られる。

バージョン 12VRS (XLC12VRS/ MLC12VRS) の新機能

- 新しい入力 "arAxisCtrl_gb[].Admin.PowerOn": このビットが設定されている場合 (デフォルト値は TRUE), AxisInterface で MC_Power が内部的に呼び出されます。入力が設定されていない場合、MC_Power はユーザープログラムによって操作する必要があります。
- サンプル プロジェクト AxisInterface: "GAT compact" に加えて、Template ウィザードにより特別な AxisInterface サンプル プロジェクトが提供されるようになりました。
- 診断番号と診断メッセージの作成: バージョン 12VRS 以上の場合、IndraDrive ドライブの A-0-0020/A-0-0023 は常にドライブ パラメータ S-0-0095/S-0-0390 で作成されます。AxisInterface はデフォルトにより必ずコントロール パラメータ A-0-0020/A-0-0023 を読み取るようになりました。この動作は、arAxisCtrl_gb[].Admin.DiagNbrRefreshTime を介して変更できます。
- IMC interface ターゲット モードの初期化: ターゲット モード "ImcCtrl.Admin._OpMode" が、ゼロによってではなく、パラメータ化に基づいて初期化されるようになりました。

IF 現在のモード (C-0-0452) >= スタートアップ ターゲット モード (C-0-0450) THEN

ImcCtrl.Admin._OpMode := 現在のモード (C-0-0452)

OTHERWISE

ImcCtrl.Admin._OpMode := スタートアップ ターゲット モード (C-0-0450)

これにより、少なくともスタートアップ ターゲット モード (C-0-0450) が実現されるか、または現在のモードが維持されます。

初期化は、PLC プログラムが起動されたときに 1 回のみライブラリで行われ、その後はいつでも上書きできます。

- 12V08 および 13V04 移行の新機能: 要素 "arAxisStatus_gb[].Admin.AxisFeatures" はバイナリ コードで軸プロパティを表示します。

要素 "arAxisStatus_gb[].Admin.AxisType" は置き換えられ、今後使用されません。

ML_TechInterface.library

**バージョン 13VRS (XLC13VRS/
MLC13VRS) の新機能**

- 協調運動のための軸の制御 (robot control)。新しいファンクションブロック MB_AxisInterfaceType02 (MB_AxisInterfaceType01 の代替) を使用することにより、運転モード "Robot Control" で軸を操作できます。
- 入力 "arAxisCtrl_gb[].Admin.ParamLevel" および出力 "arAxisStatus_gb[].Admin.ParamLevelAck" によるパラメータレベルのサポート
- 12V08 および 13V04 移行の新機能：要素 "arAxisStatus_gb[].Admin.AxisFeatures" はバイナリコードで軸プロパティを表示します。
要素 "arAxisStatus_gb[].Admin.AxisType" は置き換えられ、今後使用されません。



さまざまな AxisInterface バージョン "02VRS" (MLC 04VRS) および TechInterface "10VRS" (MLC10VRS) 以上により新しいバージョンの間で変更が実施され、その一部はユーザープログラムでの適合を必要とします。必要となる適合は [56 ページ "使用開始時の TechInterface 設定" 6.2 章](#) で説明されています。

6.2 使用開始時の TechInterface 設定

TechInterface の機能を使用するためには、まず初期化を行う必要があります。以下の手順により、IndraLogic で AxisInterface が有効になります。



このセクションで説明されている手順の多くは、すでにプログラム テンプレート "GAT または GAT^{compact}" に含まれています。これらのテンプレートは、IndraLogic にインポートし、新規プロジェクトの手引きとして使用することができます。GAT または GAT^{compact} テンプレートを使用する場合、ユーザーは AxisInterface が実行する拡張の可能性を含め、AxisInterface 処理の全体をプログラミングします。

サンプル プロジェクト GAT^{compact} の使用にあたっての最初の手順

1. 新規プロジェクトを作成し、例えば "MyControl" という名前の、対応するコントロールを挿入します。その後コントロールの IP アドレスを入力します。実在の軸がコントロールに接続されている場合、コントロールの下にそれを挿入します。
2. コントロールを右クリックし、コンテキストメニューで "GAT" を選択します。
3. ウィザードで "GAT compact" のバージョンを選択し、**Finish** (完了) をクリックしてインポートを開始します。GAT ウィザードは、指定されたプロジェクト構造が維持されていれば、軸の追加または削除を行うため、あるいは表示例を呼び出すために、いつでも呼び出すことができます。ユーザーによる使用開始を支援するために、GAT ウィザードにはすでに 2 本の仮想軸が挿入されています。実在の軸がある場合は、GAT ウィザードを使用してそれらを追加します。
4. モーションプロジェクトをコントロールに転送するには、"MyControl" をクリックし、その後 "MyControl->Go online" により転送を開始します。次に、"BB" アイコンを使用して運転モードに切り替えます。
5. **Debug ▶ Login to application...** (デバッグ -> アプリケーションにログイン) を介して PLC プロジェクトのロードを開始します。次に、矢印アイコンを使用して PLC を起動します。
6. これで、製品出荷時に組み込まれている表示例 (System_Overview_All_Axis) を使用して、仮想の軸を動かすことができます。

ユーザー自身のプログラムでの最初の手順

1. プロジェクト ツリーの "Logic->Application" (ロジック -> アプリケーション) の項目で Library Manager (ライブラリ マネージャ) を開き、ライブラリ **ML_TechInterface.compiled-library** を現在の IndraLogic プロジェクトに追加します。
2. "Application" (アプリケーション) ノードの下に、2 つの "global variable lists" (グローバル変数リスト) を追加し、例えば **"UserVarGlobal"** および **"UserConstGlobal"** などの名前を付けます (GAT compact で)。次に、以下のコードを生成します。

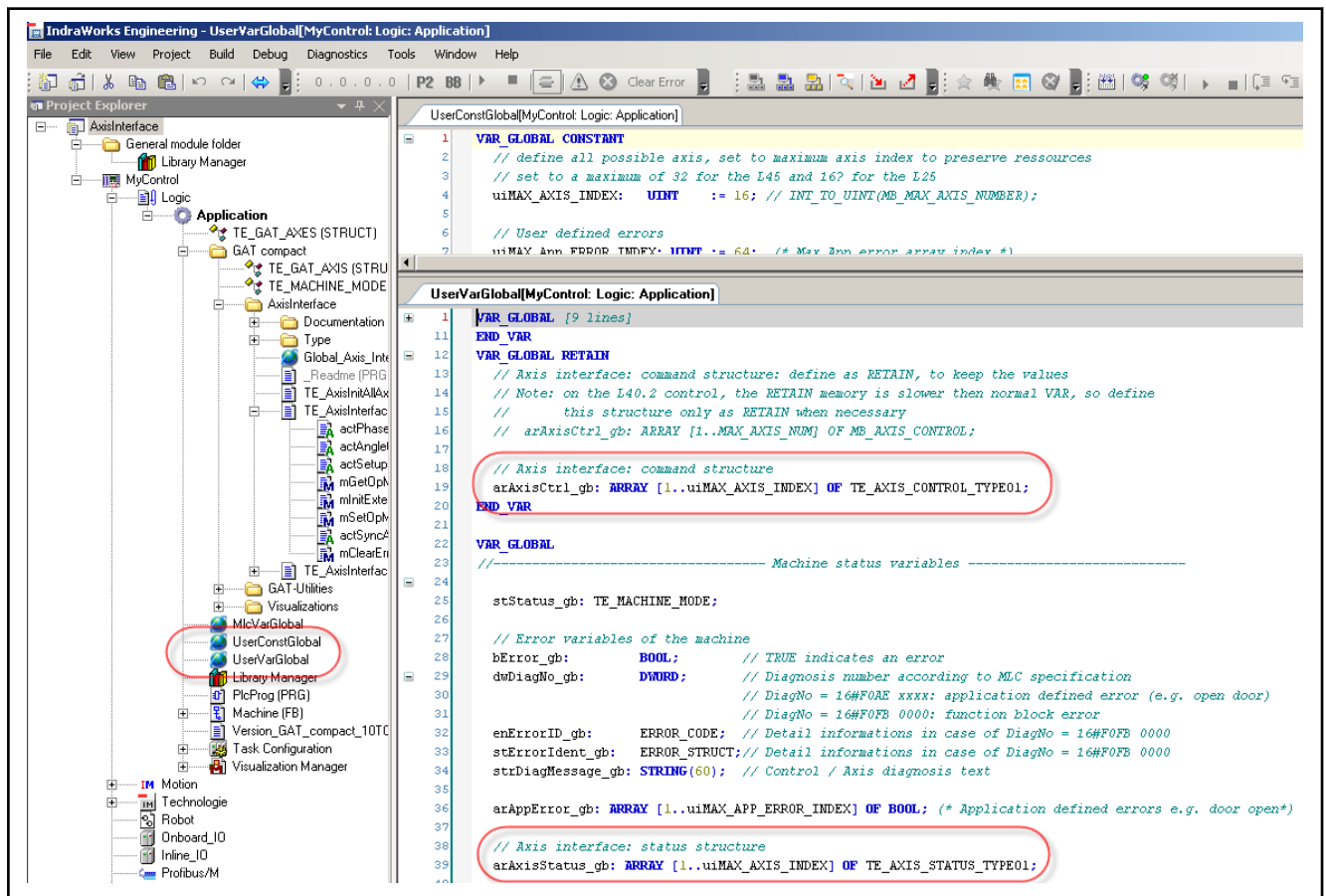


図 6-4: AxisInterface のグローバル変数

3. MAX_AXIS_NUM 軸 の ファンク シ ョ ン ブ ロ ッ ク MB_AxisInterfaceType01 を呼び出すプログラム (例: GAT compact の TE_AxisInterfaceType01Prog) を作成します。このファンクションブロックは "arAxisCtrl_gb" 内の値を解釈し、"arAxisStatus_gb" 内の値を更新します。また、プロジェクト構造のために必要な場合は、このファンクションブロックを別個のプログラムで呼び出すこともできます。
4. "Task Configuration" (タスク設定) をクリックし、周期的に呼び出される PlcTask として TE_AxisInterfaceType01Prog を指定します (間隔の例: T#20ms)。
5. "Task Configuration" で、ライブラリ TECHINTERFACE からプログラム "MB_ImcInterface" を呼び出します (間隔の例: T#20ms または自走タスクで)。
6. 起動時に 1 回、またはパラメータ化から運転モードへのフェーズ切替のたびに呼び出される初期化プログラムを作成し、すでに作成済みのプロ

ML_TechInterface.library

グラム (例 : GAT_{compact} の TE_AxisInterfaceType01Prog) にこのプロセスを組み込みます。

- 使用される各軸について、arAxisCtrl_gb および arAxisStatus_gb の基本的な初期化に MB_AxisInitType01 FB を使用します。

GAT_{compact} では、パラメータ化の結果次第で、これは TE_AxisInitAllAxisType01 FB によって実行されます。

- 構造 arAxisCtrl_gb[] は保持データとして定義できます。この場合、定義済みの値ですべての NOT RETAIN 値を初期化します (定義済みの値については、TE_AxisInterfaceType01Prog の "STATE_FIRST_INIT" を参照)。

構造が RETAIN として定義されない場合は、デフォルト値と異なるすべての要素を初期化する必要があります。

7. arAxisCtrl_gb[], arAxisStatus_gb[] および AxisData[] を使用してアプリケーションをプログラミングします。

AxisInterface プロジェクトの変換



以下の変換を実行する前に、IndraWorks プロジェクトアーカイブを作成します。プロジェクトプログラムの例の中の、変更すべきでない部分に適合が行われた場合、その部分は失われます。

AxisInterface 02VRS (MLC 04VRS) からプロジェクトを変換するには、AxisInterface に以下の調整を行う必要があります。

1. グローバル変数 :

構造の名前変更

- AXIS_CONTROL_TYPE01
(MB_AXIS_CONTROL_TYPE01 内) および
- AXIS_STATUS_TYPE01
(MB_AXIS_STATUS_TYPE01 内)

Global_Axis_Interface 内の定数 MAX_AXIS_IF_CYCLES は不要になりました。

2. タスク設定

呼出 "ImcInterface()" を "MB_ImcInterface()" に変更します。

3. 以下の構造要素の移動/名前変更が行われ、ユーザー プログラムで使用されるときに適合を行う必要があります。

- arAxisStatus[i].Active → arAxisStatus[i].Admin.Active
- arAxisCtrl[i].SyncMode.AdjCamShaftDist → arAxisCtrl[i].SyncMode.AdjCamShaftDistance
- arAxisCtrl[i].Comment → arAxisStatus[i].Admin.Name
- arAxisCtrl[i].SyncMode.InputRev → arAxisCtrl[i].SyncMode.InputRevolution
- arAxisCtrl[i].SyncMode.OutputRev → arAxisCtrl[i].SyncMode.OutputRevolution
- arAxisCtrl[i].SyncMode.CamShaftDist → arAxisCtrl[i].SyncMode.CamShaftDistance
- arAxisCtrl[i].SyncMode.AdjCamShaftDist → arAxisCtrl[i].SyncMode.AdjCamShaftDistance

4. 構造 arAxisCtrl/arAxisStatus が拡張された場合、MB_AXIS_CONTROL_TYPE01 または MB_AXIS_STATUS_TYPE01 からの変更をユーザー構造に組み込む必要があります。すべての構造の名前が AXIS_xxx から MB_AXIS_xxx に変更されました。

AxisInterface 02VRS (MLC 04VRS) から TechInterface 10VRS 以上 (MLC 09VRS 以上) にプロジェクトを変換するには、AxisInterface に以下の調整を

行う必要があります。IndraWorks プロジェクトの変換がすでに実行されていることを前提とします。

新しい BIT データ タイプにより、AxisCtrl 構造内の "_OpMode" へのアクセスが変化します。AxisStatus 構造では、"_OpMode" へのアクセスに変化はありません。

- 列挙としての使用例 (列挙データ タイプ) :

MLC 04VRS:

```
arAxisCtrl[MyRealAxis1.Axisno].Admin._OpMode:      =
ModeAH
```

MLC 10VRS 以上

```
arAxisCtrl[MyRealAxis1.Axisno].Admin._OpMode.en:    =
ModeAH
```

- BIT の使用例 :

02VRS:

```
arAxisCtrl[MyRealAxis1.Axisno].Admin.MODE_SYNC_PHASE:
= TRUE;
```

MLC 10VRS 以上

```
arAxisCtrl[MyRealAxis1.Axisno].Admin._OpMode.b.MODE_S
YNC_PHASE: = TRUE;
```

1. 列挙としての使用手順 (列挙データ タイプ) :

- "Admin._OpMode" を検索する
- "arAxisCtrl[]" が見つかった場所のみで "Admin._OpMode.en" に置き換える

2. BIT の使用手順 :

- "Admin.MODE_" を検索する
- "arAxisCtrl[]" が見つかった場所のみで "Admin._OpMode.b.MODE_" に置き換える

MLC リリース 10V04 以上については、AxisInterface サンプル プロジェクトは提供されなくなりました。代わって、"GAT compact" サンプル プロジェクトを導入用テンプレートとして使用できます。"GAT compact" への変換を進める中で、サンプル プロジェクトの多数の異なる部分の名前が変更されました。

• arAxisStatus[i]	→ arAxisStatus_gb[i]
• arAxisCtrl[i]	→ arAxisCtrl_gb[i]
• Global_Axis_Interface	→ UserVarGlobal
• Global_Axis_Interface_Const	→ UserConstGlobal
• MY_AxisInterface	→ TE_AxisInterface
• ML_AxisInitAllAxisType01	→ TE_AxisInitAllAxisType01
• AxisInterfaceType01Prog and InitProg	→ combined into TE_AxisInterfaceType01Prog
• Structures MY_	→ TE_...

6.3 AxisInterface のファンクションブロック

6.3.1 MB_AxisInitType01

概略 単一軸用の AxisInterface を初期化するために MB_AxisInitType01 FB が使用されます。このファンクションブロックは、プログラムを起動するときに 1

ML_TechInterface.library

割当: ターゲット システム/ライブラリ

回のみ、またはパラメータ化モードから運転モードにフェーズが切り替えられるたびに呼び出す必要があります。

ターゲット システム	ライブラリ
IndraMotion MLC 09VRS 以上	ML_TechInterface.compiled-library
IndraMotion MLD, MPx07 以上、追加パッケージ MA 付き	MX_AxisInterface.lib

表 6-4: 参照表 MB_AxisInitType01

インターフェースの説明

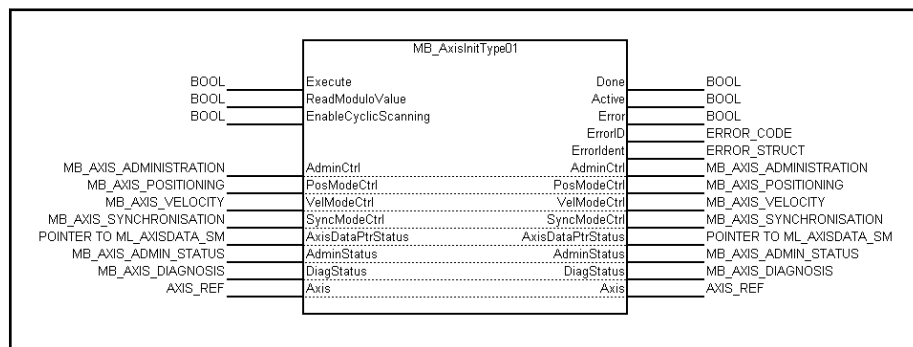


図 6-5: MB_AxisInitType01 ファンクションブロック

I/O タイプ	名称	データの種類	備考
VAR_INPUT	Execute	BOOL	初期化は "Execute" の立ち上がりエッジを介して開始されます。
	ReadModuloValue	BOOL	パラメータ設定により、回転軸のモジュール値が PosModeCtrl および SyncModeCtrl.AdjPhaseOffset の Adjustment 構造に書き込まれます。他のすべての設定値については、モジュール値はゼロに設定されます。デフォルト値は TRUE
	EnableCyclicScanning	BOOL	現在の運転モードの修正済み入力の周期的テストと転送を有効化します - 次のコメント付きの arAxisCtrl_gb[] に関連します: Cyclically tested AND transferred in case of change (変更の場合は周期的にテストされ、転送されます)。デフォルト値は TRUE
VAR_OUTPUT	Done	BOOL	ファンクションブロックが処理を完了したときに設定
	Active	BOOL	ファンクションブロックが有効な (アイドル モードではない) 場合に設定
	エラー	BOOL	ファンクションブロック インスタンスでエラーが発生したことを示します。
	ErrorID	ERROR_CODE	エラーの原因に関する短い注記
	ErrorIdent	ERROR_STRUCT	エラーに関する詳細な情報

I/O タイプ	名称	データの種類	備考
VAR_IN_OUT	Axis	AXIS_REF	AxisInterface で初期化される軸
	AdminCtrl	MB_AXIS_ADMINISTRATION	軸管理 arAxisCtrl_gb[i].Admin を接続
	PosModeCtrl	MB_AXIS_POSITIONING	位置設定運転モード arAxisCtrl_gb[i].PosMode を接続
	VelModeCtrl	MB_AXIS_VELOCITY	速度制御運転モード arAxisCtrl_gb[i].VelMode を接続
	SyncModeCtrl	MB_AXIS_SYNCHRONISATION	同期運転モード arAxisCtrl_gb[i].SyncMode を接続
	AxisDataPtrStatus	POINTER TO ML_AXISDATA_SM	この軸の AxisData へのポインタ arAxisStatus_gb[i].PtrAxisData を接続
	AdminStatus	MB_AXIS_ADMIN_STATUS	軸のステータス管理 arAxisStatus_gb[i].Admin を接続
	DiagStatus	MB_AXIS_DIAGNOSIS	軸の診断情報 arAxisStatus_gb[i].Diag を接続

表 6-5: I/O インターフェース、MB_AxisInitType01



1 つの入力を使用して MB_Axis_Control_Type01 および MB_Axis_Status_Type01 構造 (例: arAxisCtrl_gb[] / arAxisStatus_gb[]) のインスタンス全体をファンクションブロックに転送することは不可能です。これは、ユーザー アプリケーションの既存のコードへの特別な補足を可能にするために行われました。詳細は97 ページ "AxisInterface のユーザー拡張" 6.7 章に記載されています。

このため、MB_Axis_Control_Type01 および MB_Axis_Status_Type01 の必須要素は個別の入力として転送されます。

ファンクションブロック MB_AxisInitType01 は以下の構造要素をデフォルト値で初期化します。

構造要素	デフォルト
AdminCtrl._OpMode	ModeAB
AdminCtrl.ClearError	FALSE
AdminCtrl.SetAbsRef	FALSE
AdminCtrl.EnableCyclicScanning	EnableCyclicScanning 入力
AxisDataPtrStatus	ADR(AxisData[Axis.AxisNo])
AdminCtrl.Axis.CntrlNo	LOCAL_CNTRL
AdminCtrl.Axis.AxisNo	Axis.AxisNo 入力
PosModeCtrl.Adjust.Enable	FALSE
VelModeCtrl.Adjust.Enable	FALSE
SyncModeCtrl.AdjCamShaftDistance.Enable	FALSE

ML_TechInterface.library

構造要素	デフォルト
SyncModeCtrl.AdjFineadjust.Enable	FALSE
SyncModeCtrl.AdjPhaseOffset.Enable	FALSE
DiagStatus.Number	0
DiagStatus.Message	読み取りは無効
AdminStatus.Active	有効なドライブについて TRUE
AdminStatus.AxisFeatures	軸のタイプによっては MB_AXIS_FEATURES
AdminStatus.Name	パラメータ設定に依存する軸名

表 6-6: ファンクションブロックにより初期化される構造要素

上に挙げられた構造要素は、**ReadModulo = TRUE** 入力と以下の状態が存在する場合のみ初期化されます。

- VelModeCtrl.Adjust.ModuloValue = 0
- SyncModeCtrl.AdjCamShaftDistance.ModuloValue = 0
- SyncModeCtrl.AdjFineadjust.ModuloValue = 0
- 軸スケールが絶対フォーマットに設定されている場合
 - PosModeCtrl.Adjust.ModuloValue = 0
 - SyncModeCtrl.AdjPhaseOffset.ModuloValue = 0
- 軸スケールがモジュロ フォーマットに設定されている場合
 - PosModeCtrl.Adjust.ModuloValue = モジュロ値
 - SyncModeCtrl.AdjPhaseOffset.ModuloValue = モジュロ値

エラー処理 ファンクションブロックは特別なエラー コードを生成しません。パラメータを読み取るために内部的に使用される FB のエラー コードのみが転送されます。

6.3.2 MB_AxisInterfaceType01

概略 MB_AxisInterfaceType01 FB は、単一軸の AxisInterface を設定するために使用します。このファンクションブロックは運転モードが有効である間、周期的に呼び出す必要があります (モーションサイクルで、またはモーションサイクルより低速で)

割当: ターゲット システム/ライブラリ

ターゲット システム	ライブラリ
IndraMotion MLC 09VRS 以上	ML_TechInterface.compiled-library
IndraMotion MLD, MPx07 以上、追加パッケージ MA 付き	MX_AxisInterface.lib

表 6-7: 参照表、MB_AxisInterfaceType01

インターフェースの説明

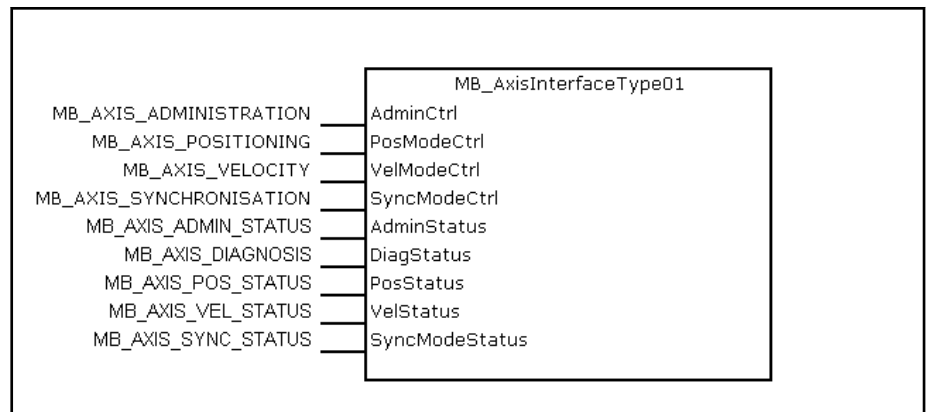


図 6-6: MB_AxisInterfaceType01 ファンクションブロック

I/O タイプ	名称	データの種類	備考
VAR_INPUT	AdminCtrl	REFERENCE TO MB_AXIS_ADMINISTRATION	軸管理、arAxisCtrl_gb[i].Admin を接続
	PosModeCtrl	REFERENCE TO MB_AXIS_POSITIONING	位置設定運転モード、arAxisCtrl_gb[i].PosMode を接続
	VelModeCtrl	REFERENCE TO MB_AXIS_VELOCITY	速度制御運転モード、arAxisCtrl_gb[i].VelMode を接続
	SyncModeCtrl	REFERENCE TO MB_AXIS_SYNCHRONISATION	同期運転モード、arAxisCtrl_gb[i].SyncMode を接続
	AdminStatus	REFERENCE TO MB_AXIS_ADMIN_STATUS	軸のステータス管理、arAxisStatus_gb[i].Admin を接続
	DiagStatus	REFERENCE TO MB_AXIS_DIAGNOSIS	軸のステータス診断、arAxisStatus_gb[i].Diag を接続
	PosStatus	REFERENCE TO MB_AXIS_POS_STATUS	位置設定運転モードのステータス、arAxisStatus_gb[i].PosMode を接続
	VelStatus	REFERENCE TO MB_AXIS_VEL_STATUS	速度運転モードのステータス、arAxisStatus_gb[i].VelMode を接続
	SyncModeStatus	REFERENCE TO MB_AXIS_SYNC_STATUS	同期運転モードのステータス、arAxisStatus_gb[i].SyncMode を接続

表 6-8: インターフェース変数、MB_AxisInterfaceType01

ML_TechInterface.library



1 つの入力を使用して MB_Axis_Control_Type01 および MB_Axis_Status_Type01 構造 (例: arAxisCtrl_gb[]/arAxisStatus_gb[]) のインスタンス全体をファンクションブロックに転送することは不可能です。これは、ユーザー アプリケーションの既存のコードへの特別な補足を可能にするために行われました。詳細は [97 ページ "AxisInterface のユーザー拡張" 6.7 章](#) に記載されています。

このため、MB_Axis_Control_Type01 および MB_Axis_Status_Type01 の必須要素は個別の入力として転送されます。

パフォーマンスを最適化するには、VAR_IN_OUT または VAR_INPUT としてではなく、REFERENCE TO により構造を接続します。これにより、入力の初期化は 1 回のみで済みます。この方法により、ファンクションブロックが周期的に呼び出されても、構造を転送する必要はなくなります。

ファンクションブロックは arAxisCtrl_gb[] の入力を確認し、軸のために要求されたコマンドを内部的に生成します。arAxisStatus_gb[] の出力は、このコマンドの結果次第で更新されます。

"例えば、ModeAb" から "ModePosAbs" への設定

"arAxisCtrl_gb[].Admin._OpMode" は、以下のシーケンスをもたらします。

- "Axis in Abo" などのモーションコマンドを有効にするために必要な状態を確認する
- PLCopen ファンクションブロック MC_Power (arAxisCtrl_gb.Admin.PowerOn = TRUE である場合) を有効化する
- 軸の電源が追加されたという応答を待つ (AH/AF)
- PosModeCtrl のコマンド値で PLCopen ファンクションブロック MC_MoveAbsolute を有効化する
- ModePosAbs に arAxisStatus_gb[].Admin._OpModeAck を応答する (Bit MODE_POS_ABS)
- PosModeCtrl.Position の値をスキャンし、PosModeCtrl.Velocity 変更があった場合は MC_MoveAbsolute を再度有効にする



OpMode "arAxisCtrl_gb[].Admin._OpMode" が "ModeCoordinated" およびファンクションブロック MB_AxisInterfaceType01 により使用されている場合、エラーメッセージ "OpMode" (運転モードはサポートされない) が出力されません。

エラー処理 ファンクションブロックは、表 "F_RELATED_TABLE"、16#0170 の DiagStatus.ErrorIdent.Additional1/Additional2 で以下のエラーメッセージを生成します。

ErrorID	Additional1	Additional2	説明
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	無効なファンクションブロックの状態
DEVICE_ERROR, (16#0008)	16#1500	16#0000	ドライブは AxisData[] でエラーを報告
INPUT_INVALID_ERROR, (16#0001)	16#1501	16#0000	無効な運転モードが選択された
RESOURCE_ERROR, (16#0003)	16#1502	16#0000	ドライブが AB がない

ErrorID	Additional1	Additional2	説明
INPUT_INVALID_ERROR, (16#0001)	16#1503	16#0000	無効な AdminCtrl.Axis.AxisNo
RESOURCE_ERROR, (16#0003)	16#150F	16#0000	AdminCtrl.PowerOn = FALSE およびドライブが AH/AF がない
INPUT_INVALID_ERROR, (16#0010)	16#1510	OpMode	運転モードはサポートされていない
INPUT_INVALID_ERROR, (16#0011)	16#1511	16#0000	1 つ以上の入力に参照がない

表 6-9: エラーコード、MB_AxisInterfaceType01

6.3.3 MB_AxisInterfaceType02

概略 Robot Control アプリケーションの一部として使用される単一軸用の AxisInterface を設定するために MB_AxisInterfaceType02 ファンクションブロックが使用されます。このファンクションブロックは運転モードが有効である間、周期的に呼び出す必要があります (モーションサイクルで、またはモーションサイクルより低速で)

割当: ターゲット システム/ライブラリ

ターゲット システム	ライブラリ
IndraMotion MLC 13VRS 以上	ML_TechInterface.compiled-library

表 6-10: 参照表、AxisInterfaceType02

インターフェースの説明

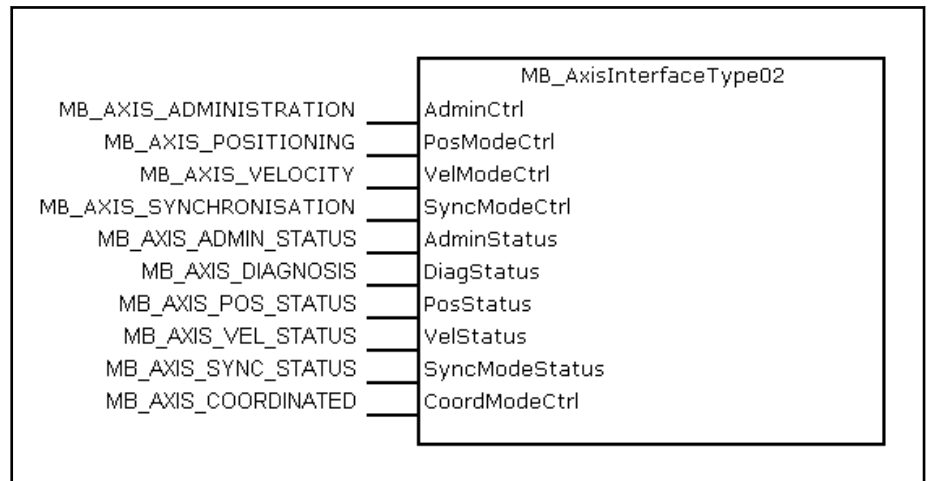


図 6-7: MB_AxisInterfaceType02 ファンクションブロック

ML_TechInterface.library

I/O タイプ	名称	データの種類	備考
VAR_INPUT	AdminCtrl	REFERENCE TO MB_AXIS_ADMINISTRATION	軸管理、arAxisCtrl_gb[i].Admin を接続
	PosModeCtrl	REFERENCE TO MB_AXIS_POSITIONING	位置設定運転モード、arAxisCtrl_gb[i].PosMode を接続
	VelModeCtrl	REFERENCE TO MB_AXIS_VELOCITY	速度制御運転モード、arAxisCtrl_gb[i].VelMode を接続
	SyncModeCtrl	REFERENCE TO MB_AXIS_SYNCHRONISATION	同期運転モード、arAxisCtrl_gb[i].SyncMode を接続
	AdminStatus	REFERENCE TO MB_AXIS_ADMIN_STATUS	軸のステータス管理、arAxisStatus_gb[i].Admin を接続
	DiagStatus	REFERENCE TO MB_AXIS_DIAGNOSIS	軸のステータス診断、arAxisStatus_gb[i].Diag を接続
	PosStatus	REFERENCE TO MB_AXIS_POS_STATUS	位置設定運転モードのステータス、arAxisStatus_gb[i].PosMode を接続
	VelStatus	REFERENCE TO MB_AXIS_VEL_STATUS	速度運転モードのステータス、arAxisStatus_gb[i].VelMode を接続
	SyncModeStatus	REFERENCE TO MB_AXIS_SYNC_STATUS	同期運転モードのステータス、arAxisStatus_gb[i].SyncMode を接続
	CoordModeCtrl	MB_AXIS_COORDINATED	CoordinatedMotion, arAxisCtrl_gb[i]. CoordMode を接続

表 6-11: インターフェース変数、MB_AxisInterfaceType02

-  ファンクションブロック MB_AxisInterfaceType02 はコントロール IndraMotion MLC のみで使用できます。
-  "ModeCoordinated" を伴う OpMode "arAxisCtrl_gb[i].Admin._OpMode" が ファンクションブロック MB_AxisInterfaceType01 と共に使用された場合、エラー メッセージ "OpMode" (運転モードはサポートされない) が返されます。

TechInterface による協調運動の一般的なシーケンス

- ファンクションブロックにはすべての入力および MB_AxisInterfaceType01 ファンクションブロックの機能のほか、軸の協調運動のための "CoordModeCtrl" 入力もあります。
- ファンクションブロックでの各キネマティクスに関わるすべての軸に運転モード "MODE_COORDINATED" を設定する
 - キネマティクス プログラムをコンパイルし、コントロール IndraMotion MLC にロードする
 - キネマティクス プログラムを選択する (MB_IRDProgSelect ファンクションブロックにより)
 - キネマティクス プログラムを起動する (MB_IRDProgStart ファンクションブロックにより)

- 協調運動が実行される
- ...
- キネマティクス プログラムを停止する (MB_IRDProgStop ファンクションブロックにより)
- ファンクションブロックでの各キネマティクスに関わるすべての軸について運転モード "MODE_COORDINATED" を再び解除する MB_AxisInterfaceType02 ファンクションブロック



キネマティクス プログラムは IEC または RCL (Robot Control 言語) でプログラミングできます。

エラー処理 ファンクションブロックは、表 "F_RELATED_TABLE"、16#0170 の DiagStatus.ErrorIdent.Additional1/Additional2 で以下のエラー メッセージを生成します。

ErrorID	Additional1	Additional2	説明
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	無効なファンクションブロックの状態
DEVICE_ERROR, (16#0008)	16#1500	16#0000	ドライブは AxisData[] でエラーを報告
INPUT_INVALID_ERROR, (16#0001)	16#1501	16#0000	無効な運転モードが選択された
RESOURCE_ERROR, (16#0003)	16#1502	16#0000	ドライブが AB がない
INPUT_INVALID_ERROR, (16#0001)	16#1503	16#0000	無効な AdminCtrl.Axis.AxisNo
RESOURCE_ERROR, (16#0003)	16#150F	16#0000	AdminCtrl.PowerOn = FALSE およびドライブが AH/AF がない
INPUT_INVALID_ERROR, (16#0011)	16#1511	16#0000	1 つ以上の入力に参照がない

表 6-12: エラー コード、MB_AxisInterfaceType02

6.4 軸モード

6.4.1 ダイアログベースの軸コマンド

- 利点** 快適な軸コマンドのために、ダイアログベースの軸コマンドがすべての Bosch Rexroth プログラミング テンプレートで利用できます。選択した軸に必要なすべてのコマンド値をダイアログで入力し、各 PLC コードを生成して挿入します。これにより、時間を省き、エラーを減らすことができ、しかも構文規則の知識も不要です。
- 呼出** コンテキスト メニュー "Command axis" (コマンド軸) を介してダイアログを呼び出します。この機能は、"Structured Text (ST) Editor" で軸コマンドに適した位置にカーソルがある場合に利用できます。

ML_TechInterface.library

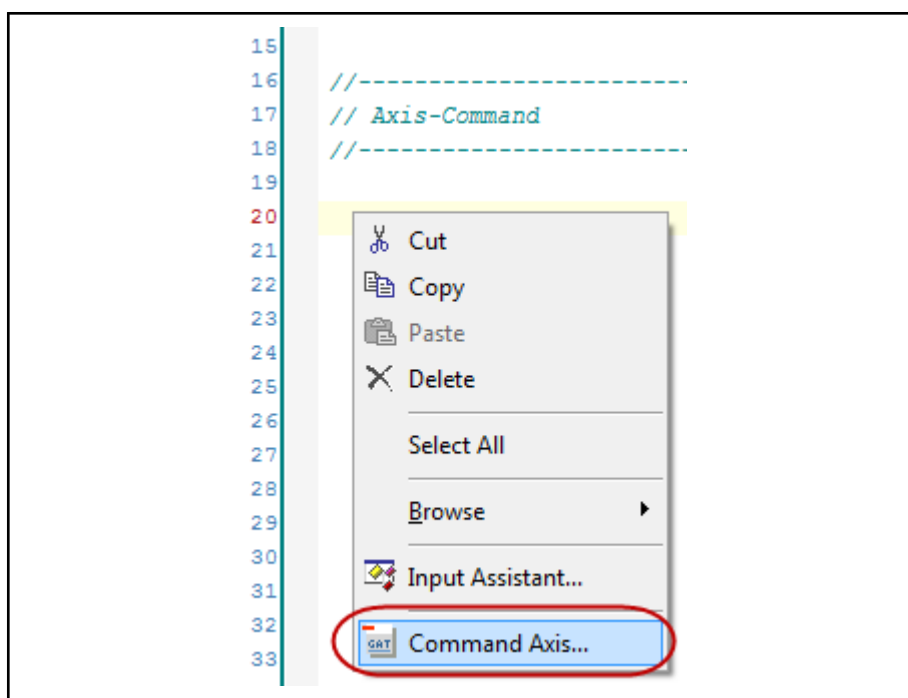


図 6-8: ダイアログベースの軸コマンドの呼出

以下の章で説明されている PLC コードの生成規則は自動的に順守されます。このため、以下の章は特殊なケースの参照として、または生成されたコードを解釈する上でのみ役立ちます。

ダイアログベースの軸コマンドは、以下のプログラム コード ポジションで利用可能です。

- サンプル プロジェクト AxisInterface : すべてのプログラム、ファンクションブロックおよび機能
- GATcompact: Machine ファンクションブロック
- GATcentral: すべてのモジュール
- GATdecentral: すべてのユニット

この機能には以下の特性があります。

- 特性**
- ターゲット運転モードと各コマンド値をダイアログで簡単に指定できます。
 - また、コマンド値の指定は PLC 変数を介して行うこともできます。
 - 入力支援は F2 キーを用いて入力フィールドで呼び出すことができます。
 - あるいは、コメント付きの応答コード例を追加し、運転モードが有効か否か、およびコマンド値に達したか否か確認します。
 - **Finish** (完了) をクリックした後で、PLC コードがカーソル位置に挿入されます。
 - デフォルト値 (初期値) がダイアログに表示されます。
 - 値の割当がなければコードの生成は行われません (この場合、デフォルト値または他の場所で指定された値が適用されます)

例 下の例では、軸 "MyVirtualAxis1" は指定された位置 "10" に正確に位置しています。

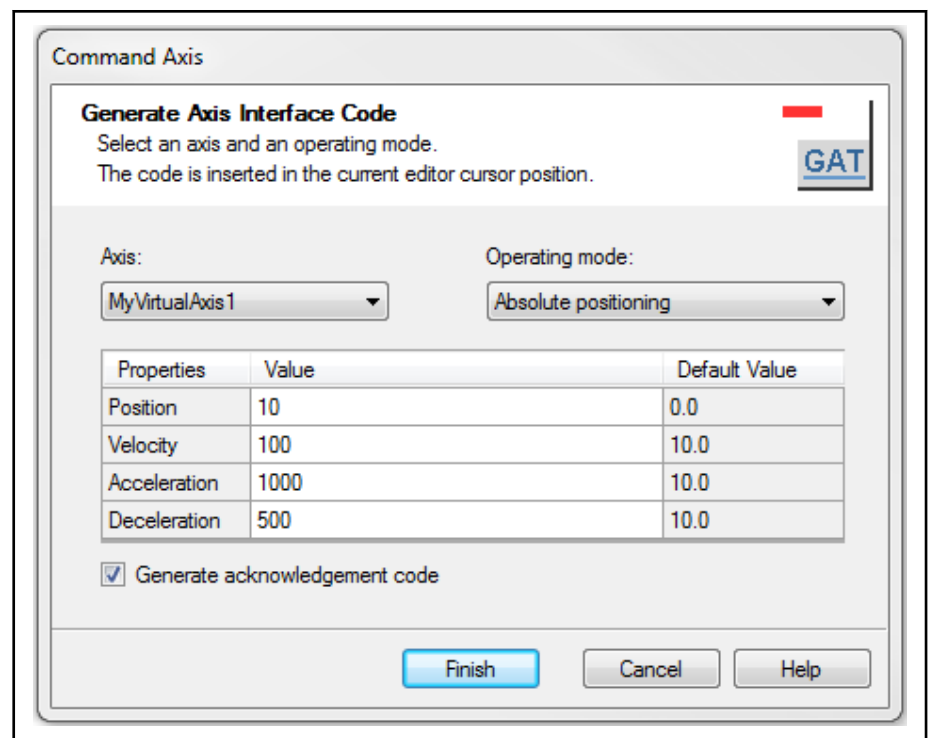


図 6-9: ダイアログベースの軸コマンドを介した軸 "MyVirtualAxis1" の絶対位置設定の例

Finish (完了) を使用して以下のプログラム コードを生成し、カーソル位置に挿入します。

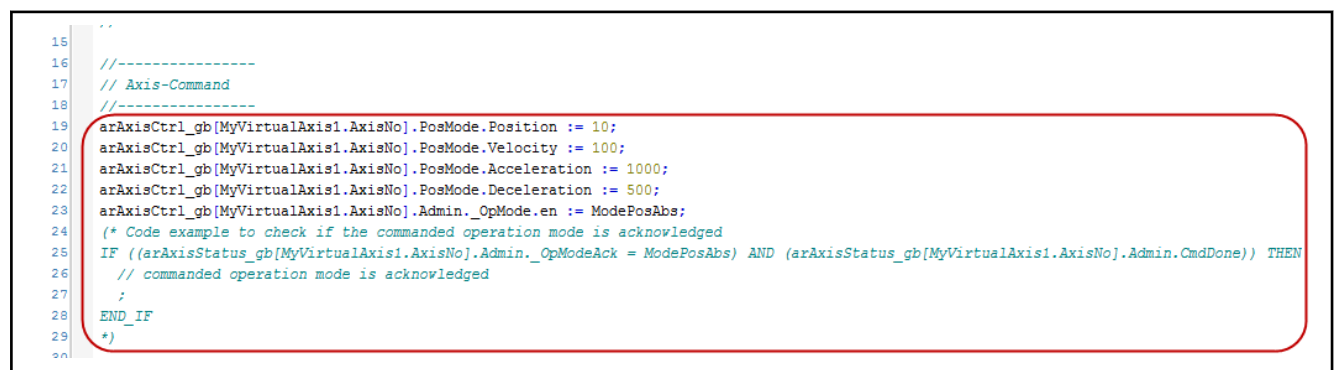


図 6-10: Finish (完了) の後で挿入されたコード

ここに示されているプログラム コードは、軸インターフェース サンプル プロジェクトの構文に対応します。他のすべての種類の GAT では軸コンテナ "Axes" が使用されているため、修正されたプログラム コードが使用されません。ダイアログベースの軸コマンドでは、すべての GAT 異形が自動的に検出され、常に適切なプログラム コードが生成されます。

6.4.2 概要

運転モードを有効にするには、以下の 2 つの方法があります。

1. Enum 値の選択

TYPE MB_AXIS_MODE の値を

arAxisCtrl_gb[].Admin._OpMode.en に割り当てる

arAxisCtrl_gb[].Admin._OpMode.en := ModePosAbs;

- または -

arAxisCtrl_gb[].Admin._OpMode.en := ModeAB;

ML_TechInterface.library

2. ビット アクセスの使用

まず、プログラミングで "_OpMode" を削除します。

```
arAxisCtrl_gb[].Admin._OpMode.en := ModeAB;
```

次に、ビット アクセス機能を使用してビットを設定します。

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_POS_ABS:= TRUE;
```



arAxisCtrl_gb[].Admin._OpMode に複数のビットを設定すると、エラーが発生し、ドライブが AB に切り替わります。

運転モードを有効にするには、まずすべての属性に値を割り当てる必要があります。すべての属性にはデフォルト値があります。一部はゼロ以外の値が設定されていますが、0 に定義されているものもあります。特別な要求により、これらに値を割り当てる必要があります。



現在の運転モードの切替が実行される前に、使用される属性 (位置、速度など) または変更されたデフォルト値を持つ属性のみを宣言する必要があります。

下の例は、運転モードを変更する場合の正しいプログラミング シーケンスを示したものです。

```

1 // The following code initializes position mode attributes
2 // to specific values
3
4 arAxisCtrl[MyRealAxis1.AxisNo].PosMode.Position := 250.0;
5 arAxisCtrl[MyRealAxis1.AxisNo].PosMode.Velocity := 500.0;
6 arAxisCtrl[MyRealAxis1.AxisNo].PosMode.Acceleration := 200.0;
7 arAxisCtrl[MyRealAxis1.AxisNo].PosMode.

```

maximum velocity (always positive) (not necessarily reached) [u/s]
 - start moving with "ModePosAbs" or "ModePosRel" or "ModePosAdd"
 (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive)
 Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)

 Positioniergeschwindigkeit, (Einheit: [u/s]) (positiv eingeben)
 - Freigabe mit "ModePosAbs" oder "ModePosRel" oder "ModePosAdd"
 (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive)
 Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE)

- Acceleration
- Adjust
- Deceleration
- Distance
- Position
- Velocity

```

15 // The following code commands the axis to switch to
16 // Absolute positioning mode
17 arAxisCtrl[MyRealAxis1.AxisNo].Admin._OpMode.en := ModePosAbs;
18
19 // The following code checks the status of the
20 // Absolute positioning mode
21 IF arAxisStatus[MyRealAxis1.AxisNo].Admin.MODE_POS_ABS THEN
22 // Customer specific code to handle status acknowledgement
23 // e.g. set a bit or output
24 ;
25 END_IF

```

図 6-11: 絶対位置設定モードの変更例



運転モードのステータス応答 (arAxisStatus_gb[].Admin.) は以下のとおり実装されます。

例：

1. 軸を絶対位置設定モードに切り替えるためのコマンドが実行された場合に限り、ステータス応答は TRUE を返します。

```
arAxisStatus_gb[].Admin.MODE_POS_ABS
```

2. ステータス応答は、ドライブ一時停止、ドライブ ホーミング および位置設定の運転モードについてのみサポートされます。

```
arAxisStatus_gb[].Admin.CmdDone
```

位置設定モードが有効になりました：

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_POS_ABS
```

ドライブが位置設定モードに切り替わり、回り始めると、出力は TRUE に設定されます：

```
arAxisStatus_gb[].Admin.MODE_POS_ABS
```

ドライブがターゲット位置に達すると、ステータス応答が設定されます：

```
arAxisStatus_gb[].Admin.CmdDone
```

入力 *arAxisCtrl_gb.Admin.PowerOn* は *AxisInterface* で *MC_Power* の内部的な使用を制御します。

- *PowerOn* = TRUE (デフォルト): *AxisInterface* が *MC_Power* の作動を支配します。運転モードが有効になると、最初の *MC_Power* が内部的に呼び出され、準備完了信号が出された後でのみ、実際に選択された運転モードが有効化されます。スイッチオフの場合 (ModeAb)、ドライブが一時停止された後で *MC_Power* が削除されます。
- *PowerOn* = FALSE: アプリケーション プログラムは *MC_Power* により操作する必要があります。運転モードが選択され、*MC_Power* が以前に作動していなかった場合、*AxisInterface* はエラーを出力します。

以下のセクションでは、*AxisInterface* Type01 によりサポートされている運転モードおよびそれらに割り当てることが可能な属性について説明します。以下においては、"PowerOn" 入力のデフォルト値が TRUE であることを前提とします。

6.4.3 Drive Ready (ドライブ準備完了)

この運転モードを有効にすると、ドライブが AB (Drive ready) に切り替わり、トルクがオフになります。この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en := ModeAB;
```



AxisInterface では *MC_Power* および *MC_Stop* PLCopen FBs を内部的に使用して切替を実行します。

ML_TechInterface.library

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	Admin.StopDeceleration	REAL	100.0	あり
	Admin.Axis	AXIS_REF		なし

表 6-13: "Drive ready" の属性

6.4.4 Drive Halt (ドライブ一時停止)

この運転モードを有効にすると、ドライブが AH (Drive Halt) に切り替わり、トルクが維持されます。運転モード "ModeAH" により、軸が PLCopen 状態 "StandStill" に転送されます。すなわち、軸インターフェースの外部で呼び出された PLCopen ファンクションブロックが受け入れられ、軸インターフェースは "ModeExternalFB" を返します。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en:= ModeAH;
```

または

```
arAxisCtrl_gb[ ].Admin.MODE_AH:= TRUE;
```



AxisInterface では MC_Power および MC_Stop PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	Admin.StopDeceleration	REAL	100.0	あり
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_AH	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 6-14: "Drive Halt" の属性

6.4.5 Drive Stop (ドライブ停止)

この運転モードを有効にすると、ドライブが AH (Drive Halt) に切り替わり、トルクが維持されます。運転モード "ModeStopping" は PLCopen 状態の機械を Stopping (停止) 状態に保ちます。すなわち、軸インターフェースの外部で呼び出された PLCopen ファンクションブロックはエラー "Command in state Stopping denied" (状態 Stopping でのコマンドは拒否されました) によりキャンセルされます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en:= ModeStopping;
```

または

```
arAxisCtrl_gb[ ].Admin.MODE_STOPPING:= TRUE;
```



AxisInterface では MC_Power および MC_Stop PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	Admin.StopDeceleration	REAL	100.0	あり
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_STOPPING	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 6-15: Drive Stop の属性

6.4.6 Drive Homing (ドライブ ホーミング)

この運転モードを有効にすることにより、ドライブのホーミングを行うコマンドが実行されます。ホーミング手順中、増分エンコーダにより軸を制御します。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl[ ].Admin._OpMode.en:= ModeHoming;
```

または

```
arAxisCtrl[ ].Admin._OpMode.b.MODE_HOMING:= TRUE;
```

運転モード コマンドの後には以下の応答が行われる必要があります。ドライブがホーミング位置に達すると、TRUE が返されます。

```
arAxisStatus[ ].Admin.CmdDone
```



AxisInterface では MC_Power および MC_Home PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	Admin.Axis	AXIS_REF		なし
arAxisStatus[]	Admin.MODE_HOMING	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 6-16: "Drive Homing" の属性

6.4.7 Absolute Positioning (絶対位置設定)

この運転モードを有効にすることにより、絶対動作が以前に指定されていたターゲット位置に戻ります。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en:= ModePosAbs;
```

または

```
arAxisCtrl_gb[ ].Admin._OpMode.b.MODE_POS_ABS:= TRUE;
```

運転モード コマンドの後には以下の応答が行われる必要があります。ドライブがターゲット位置に達すると、TRUE が返されます。

```
arAxisStatus_gb[ ].Admin.CmdDone
```

ML_TechInterface.library



AxisInterface では MC_Power および MC_MoveAbsolute PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	PosMode.Position	REAL	0.0	あり
	PosMode.Velocity	REAL	10.0	あり
	PosMode.Acceleration	REAL	10.0	なし
	PosMode.Deceleration	REAL	10.0	なし
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_POS_ABS	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 6-17: "Absolute Positioning" の属性

6.4.8 Relative Positioning (相対位置設定)

この運転モードを有効にすると、現在の実際の位置に PosMode.Distance を追加することによって、すでに指定済みのターゲット位置への相対動作が行われます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en := ModePosRel;
```

または

```
arAxisCtrl_gb[ ].Admin._OpMode.b.MODE_POS_REL := TRUE;
```

運転モード コマンドの後には以下の応答が行われる必要があります。ドライブがターゲット位置に達すると、TRUE が返されます。

```
arAxisStatus_gb[ ].Admin.CmdDone
```



AxisInterface では MC_Power および MC_MoveRelative PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	PosMode.Distance	REAL	0.0	あり
	PosMode.Velocity	REAL	10.0	あり
	PosMode.Acceleration	REAL	10.0	なし
	PosMode.Deceleration	REAL	10.0	なし
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_POS_REL	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 6-18: "Relative Positioning" の属性

6.4.9 Additive Positioning (追加位置設定)

この運転モードを有効にすると、現在のターゲット位置に PosMode.Distance を追加することによって、すでに指定済みのターゲット位置への相対動作が行われます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en:= ModePosAdd;
```

または

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_POS_ADD:= TRUE;
```

運転モード コマンドの後には以下の応答が行われる必要があります。ドライブがターゲット位置に達すると、TRUE が返されます。

```
arAxisStatus_gb[].Admin.CmdDone
```



AxisInterface では MC_Power および MC_MoveAdditive PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	PosMode.Distance	REAL	0.0	あり
	PosMode.Velocity	REAL	10.0	あり
	PosMode.Acceleration	REAL	10.0	なし
	PosMode.Deceleration	REAL	10.0	なし
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_POS_ADD	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 6-19: "Additive Positioning" の属性

6.4.10 Velocity Control (速度制御)

この運転モードを有効にすると、一時停止コマンドが発行されるまで、すでに指定済みの速度でモーションが実行されます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en:= ModeVel;
```

または

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_VEL:= TRUE;
```



AxisInterface では MC_Power および MC_MoveVelocity PLCopen FBs を内部的に使用して切替を実行します。

ML_TechInterface.library

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	VelMode.Velocity	REAL	10.0	あり
	VelMode.Acceleration	REAL	10.0	あり
	VelMode.Deceleration	REAL	10.0	あり
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_VEL	BOOL		該当せず

表 6-20: "Velocity Control" の属性

6.4.11 運転モード "Phase Synchronization" (フェーズ同期)

この運転モードが有効になると、ドライブは微調整を行う電子ギアを使用して操作されます。この機能により、マスター軸と選択されたスレーブ軸間のフェーズ同期が行われます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en:= ModeSyncPhase;
```

または

```
arAxisCtrl_gb[ ].Admin._OpMode.b.MODE_SYNC_PHASE:= TRUE;
```



AxisInterface では MC_Power、MB_GearInPos、MC_GearOut および MB_PhasingSlave PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	SyncMode.OutputRevolution	UINT	1	あり
	SyncMode.InputRevolution	UINT	1	あり
	SyncMode.Fineadjust	REAL	0.0	あり
	SyncMode.SyncDirection	MC_SYNC_DIRECTION	SHORTEST_WAY	あり
	SyncMode.StartMode	MC_START_MODE	ABSOLUTE	なし
	SyncMode.Master	AXIS_REF		あり
	Admin.Axis	AXIS_REF		なし
	SyncMode.PhaseOffset	REAL	0.0	あり
	SyncMode.PhaseOffsetVel	REAL	1.0	なし
	SyncMode.PhaseOffsetAcc	REAL	100.0	なし
arAxisStatus_gb[]	Admin.MODE_SYNC_PHASE	BOOL		該当せず
	SyncMode.PhasingSlaveDone	BOOL		該当せず

表 6-21: 属性、運転モード "phase synchronization" (フェーズ同期)

6.4.12 運転モード "Velocity Synchronization" (速度同期)

この運転モードが有効になると、ドライブは微調整を行う電子ギアを使用して操作されます。この機能により、マスター軸と選択されたスレーブ軸間の速度同期が行われます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en:= ModeSyncVel;
```

または

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_SYNC_VEL:= TRUE;
```



AxisInterface では MC_Power、MC_GearIn および MC_GearOut PLCopen_FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	SyncMode.OutputRevolution	UINT	1	あり
	SyncMode.InputRevolution	UINT	1	あり
	SyncMode.Fineadjust	REAL	0.0	あり
	SyncMode.Master	AXIS_REF		あり
	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_SYNC_VEL	BOOL		該当せず

表 6-22: 属性、運転モード "velocity synchronization"

6.4.13 運転モード "Electronic Cam" (電子カム)

この運転モードを有効にすると、ドライブは、ドライブに保存されているカム プロファイルを使用してマスターに従います。この運転モードは IndraDrive ドライブのみによりサポートされます。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en := ModeSyncCam;
```

または

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_SYNC_CAM := TRUE;
```



AxisInterface では MC_Power、MC_CamIn、MC_CamOut および MB_PhasingSlave PLCopen_FBs を内部的に使用して切替を実行します。

ML_TechInterface.library

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	SyncMode.OutputRevolution	UINT	1	あり
	SyncMode.InputRevolution	UINT	1	あり
	SyncMode.Fineadjust	REAL	0.0	あり
	SyncMode.CamShaftDistance	REAL	360.0	あり
	SyncMode.SyncDirection	MC_SYNC_DIRECTION	SHORTEST_WAY	あり
	SyncMode.StartMode	MC_START_MODE	ABSOLUTE	なし
	SyncMode.CamTableID	MC_CAM_ID	CAM_TABLE_1	あり
	SyncMode.Master	AXIS_REF		あり
	Admin.Axis	AXIS_REF		なし
	SyncMode.PhaseOffset	REAL	0.0	あり
	SyncMode.PhaseOffsetVel	REAL	1.0	なし
	SyncMode.PhaseOffsetAcc	REAL	100.0	なし
arAxisStatus_gb[]	SyncMode.PhasingSlaveDone	BOOL		
	SyncMode.CamActiveID	MC_CAM_ID		
	Admin.MODE_SYNC_CAM	BOOL		該当せず

表 6-23: 属性、運転モード "electronic cam"

6.4.14 運転モード "Electronic MotionProfile"

この運転モードを有効にすることにより、ドライブで電子カム機能のスイッチがオンに切り替わります。カム プロファイルは、電子的動作プロファイルのパラメータを使用して論理的に定義されます。マスター軸の位置により、スレーブ軸の同期位置コマンド値が決定されます。この運転モードは、仮想軸のため、または汎用ドライブのみに使用できます (プロファイルはコントロール内で計算)。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en := ModeSyncProfile;
```

または

```
arAxisCtrl_gb[ ].Admin._OpMode.b.MODE_SYNC_PROFILE := TRUE;
```



AxisInterface では MC_Power、MB_MotionProfile、MC_GearOut および MB_PhasingSlave PLCopen FBs を内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arAxisCtrl_gb[]	SyncMode.OutputRevolution	UINT	1	あり
	SyncMode.InputRevolution	UINT	1	あり
	SyncMode.Fineadjust	REAL	0.0	あり
	SyncMode.Profile.SetSelection	UINT	0	あり
	SyncMode.Profile.RelativePositioning	BOOL	FALSE	なし
	SyncMode.SyncDirection	MC_SYNC_DIRECTION	SHORTEST_WAY	あり
	SyncMode.StartMode	MC_START_MODE	ABSOLUTE	なし
	SyncMode.Master	AXIS_REF		あり
	Admin.Axis	AXIS_REF		なし
	SyncMode.PhaseOffset	REAL	0.0	あり
	SyncMode.PhaseOffsetVel	REAL	1.0	なし
	SyncMode.PhaseOffsetAcc	REAL	100.0	なし
arAxisStatus_gb[]	SyncMode.PhasingSlaveDone	BOOL		該当せず
	SyncMode.ProfileActiveSet	UINT		該当せず
	Admin.MODE_SYNC_CAM	BOOL		該当せず

表 6-24: 運転モード属性 "Electronic MotionProfile"

6.4.15 運転モード "Electronic FlexProfile"

この運転モードを有効にすることにより、ドライブで電子カム機能のスイッチがオンに切り替わります。カム プロファイルは、電子的 FlexProfile のパラメータを使用して論理的に定義されます。この運転モードは、仮想軸のため、または汎用ドライブのみに使用できます (プロファイルはコントロール内で計算)。この運転モードは、ドライブベースの IndraMotion MLD システムでは利用できません。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en:= ModeFlexProfile;
```

または

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_FLEX_PROFILE:= TRUE;
```



AxisInterface では MC_Power、ML_FlexProfile、MC_GearOut および MB_PhasingSlave PLCopen FBs を内部的に使用して切替を実行します。

ML_TechInterface.library

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arAxisCtrl_gb[]	SyncMode.OutputRevolution	UINT	1	あり
	SyncMode.InputRevolution	UINT	1	あり
	SyncMode.Fineadjust	REAL	0.0	あり
	SyncMode.Profile.SetSelection	UINT	0	あり
	SyncMode.Profile.UseSwitchingPosition	BOOL	FALSE	なし
	SyncMode.Profile.SwitchingPosition	REAL	0.0	なし
	SyncMode.Profile.ProfileEntry	MB_PROFILE_START_POINT	SLAVE_ORIGIN_MASTER_ORIGIN	なし
	SyncMode.Profile.MasterOffset	REAL	0.0	なし
	SyncMode.Profile.SlaveOffset	REAL	0.0	なし
	SyncMode.Profile.SyncType	MB_SYNC_TYPE	SYNC_RAMPIN_SHORTEST_WAY	なし
	SyncMode.Profile.ExecutionMode	MB_EXECUTION_MODE	EXECUTE_CYCLE	なし
	SyncMode.Master	AXIS_REF		あり
	Admin.Axis	AXIS_REF		なし
	SyncMode.PhaseOffset	REAL	0.0	あり
	SyncMode.PhaseOffsetVel	REAL	1.0	なし
	SyncMode.PhaseOffsetAcc	REAL	100.0	なし
arAxisStatus_gb[]	SyncMode.PhasingSlaveDone	BOOL		該当せず
	SyncMode.ProfileActiveSet	UINT		該当せず
	Admin.MODE_FLEX_PROFILE	BOOL		該当せず

表 6-25: 運転モード属性 "Electronic FlexProfile"

6.4.16 運転モード"Robot Control"

この運転モードを有効にすると、該当するドライブが周期的な位置制御 (ドライブ制御) に切り替わります。さらに、該当する軸が自動的にキネマティクスにグループ化されます。グループ化された軸は、IEC または RCL プログラム (Robot Control 言語) を使用して協調運動を実行できます。

この運転モードはドライブベースの IndraMotion MLD システムでは利用できません (この要件が、このシステムでは利用できない MB_AxisInterfaceType02 ファンクションブロックであるため)。

また、この運転モードは、コントロール IndraLogic XLC でも利用できません。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[ ].Admin._OpMode.en:= ModeCoordinated;
```


または

```
arAxisCtrl_gb[ ].Admin._OpMode.b.MODE_COORDINATED:= TRUE;
```

この運転モードは軸ステータス構造で TRUE を返します。

```
arAxisStatus_gb[ ].Admin.MODE_COORDINATED;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	CoordMode.AxisGroupRef.CtrlNo	CONTROLS	LOCAL_CNTRL	なし
	CoordMode.AxisGroupRef.GroupNo	KIN_OBJECTS	NO_KINEMATICS	なし

表 6-26: 運転モード属性 "Robot Control"

宣言：

```
例 : Kinematics1: AXIS_GROUP_REF:= (CtrlNo:=  
LOCAL_CNTRL,GroupNo:= KINEMATICS_1);
```

割当：

```
例 : arAxisCtrl_gb[ Antrieb1 ].Ctrl.CoordMode.AxisGroupRef:= Kinematics1;
```



軸インターフェースでは PLCopen ファンクションブロックである MC_Power、MC_AddAxisToGroup および MC_RemoveAxisFromGroup を内部的に使用して切替を実行します。

6.4.17 運転モード "Drive-controlled Interruption" (ドライブ制御の中断)

この運転モードは、ドライブ制御の中断を表示するために使用されます。

AxisInterface はビット "AxisData[].Axis_Interrupted" でのドライブの状態をモニタリングし、AxisStatus が TRUE のとき、ModelInterrupted に切り替わります。これにより、軸インターフェースはドライブによって、例えばドライブの安全技術によってコントロールが引き継がれたことを報告します。

このモニタリングは、入力 arAxisCtrl_gb[].Admin.DriveInterruptReaction をリセットすることにより無効にできます。デフォルト設定は "monitoring active" (モニタリング有効) です。

この運転モードを指定することはできません。"ModeExternalFb" においてのように、何らかの手段で "ModelInterrupted" が指定された場合、ドライブは AH (MC_Stop は無効) に切り替えられます。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arAxisCtrl_gb[]	Admin.DriveInterruptReaction	BOOL	TRUE	あり
arAxisStatus_gb[]	Admin.MODE_INTERRUPTED	BOOL		該当せず

表 6-27: 属性、運転モード "drive-controlled interruption"

Axis_Interrupted および入力 DriveInterruptReaction をモニタリングするためのアプリケーション：

- コントロールにブレーキと補間があるドライブによる接近：S-0-0206 "Drive on delay time" の値が大きい場合、コントロールの補間器がコマ

ンド値の指定を開始しますが、ドライブはそれに従いません。これにより、待機時間の終了時にコマンド値のジャンプが起こる可能性があります。 接近中のモニタリングは、入力 "DriveInterruptReaction" の如何にかかわらず、常に有効化されます。

- 安全技術またはドライブ内部コマンド (ブレーキ テストなど) : "DriveInterruptReaction" = TRUE. 運転モードで Axis_Interrupted が発生した場合、選択された運転モードの変更が行われるまで、arAxisStatus_gb[].Admin.MODE_INTERRUPTED が返されます。 "Axis_Interrupted" の立ち上がりエッジで、速度制御がゼロ速度に指定されます。 中断が終了する前、"Axis_Interrupted" の終了直後に新しい運転モードが指定された場合、シーケンスは新しい運転モードで続行されます。
- ドライブ機能を使用しているアプリケーション (例: 急停止タッチ プロープ) : AxisInterface によるモニタリングは望ましくないため、"DriveInterruptReaction" = FALSE によりスイッチがオフになります。
- 一時的なスイッチオフ: モニタリングを特定の時間にわたってオフにすることも可能です。 例: 固定された停止によるホーミングの場合、"DriveInterruptReaction" = FALSE のみにより固定された停止から離すことができます。

6.4.18 運転モード "External Function Block" (外部ファンクションブロック)

この運転モードを有効にすると、ドライブが AH (MC_Stop は無効) に切り替わり、その後、技術機能からのコマンドなどの外部モーションコマンドを待機します。

この運転モードは以下のコマンドにより有効化されます。

```
arAxisCtrl_gb[].Admin._OpMode.en:= ModeExternalFB;
```

または

```
arAxisCtrl_gb[].Admin._OpMode.b.MODE_EXTERNAL_FB:= TRUE;
```

 AxisInterface では MC_Power および PLCopen ファンクション ブロックを内部的に使用して切替を実行します。

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arAxisCtrl_gb[]	Admin.Axis	AXIS_REF		なし
arAxisStatus_gb[]	Admin.MODE_EXTERNAL_FB	BOOL		該当せず

表 6-28: 属性、運転モード "外部ファンクションブロック"

6.5 グローバル変数

ライブラリ ML_TechInterface (IndraMotion MLD MX_AxisInterface.lib および MX_ImcInterface.lib を伴う) には以下の 2 つのグローバル変数リストが含まれています。

- Global_MB_AxisInterfaceVars.
このリストは定数を定義します。
- Global_ImcInterface.
このリストは IMC インターフェースの制御構造とステータス構造を定義します。

- 制御構造 : ImcCtrl
- ステータス構造 : ImcStatus

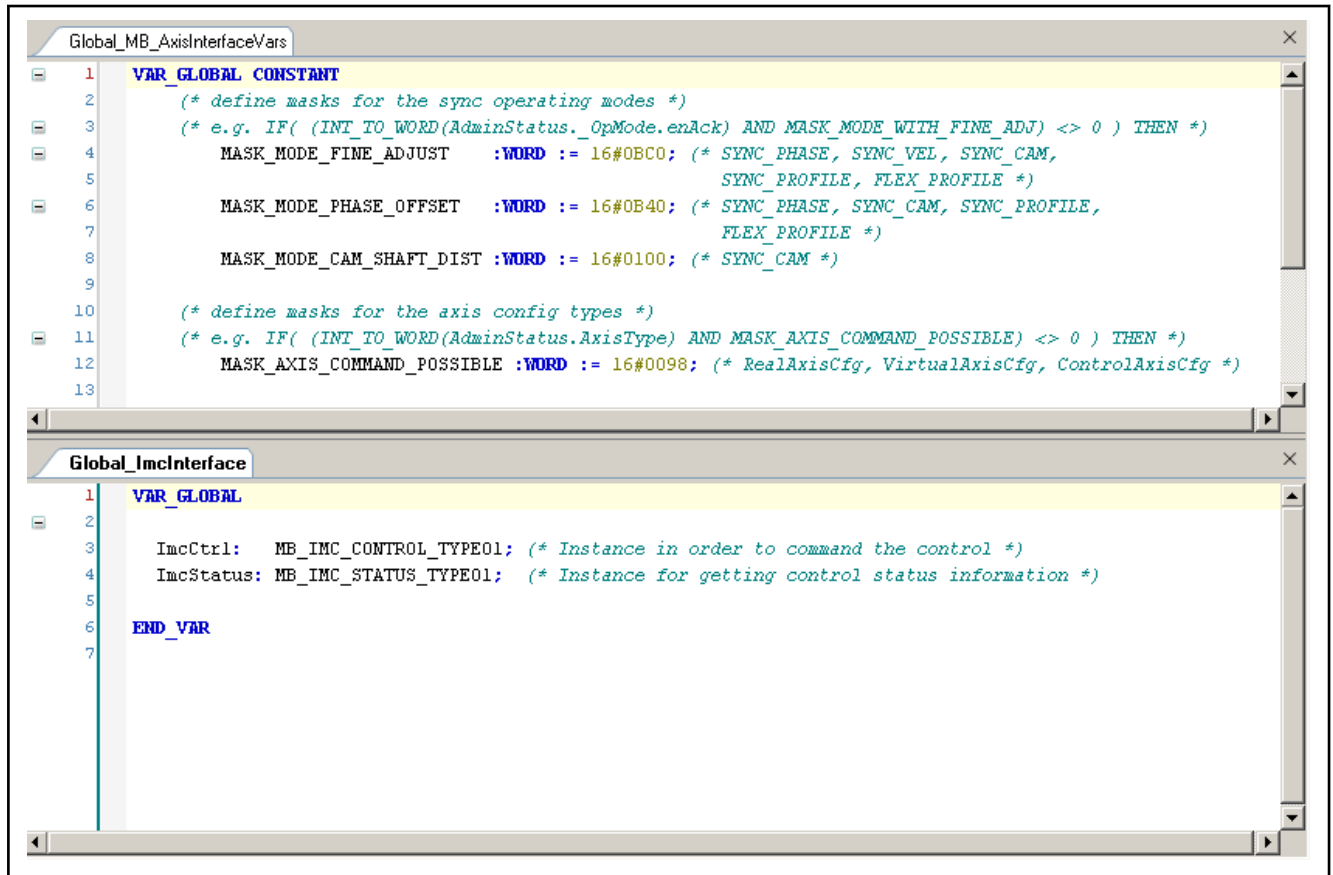


図 6-12: TechInterface のグローバル変数

6.6 最初の手順、AxisInterface

6.6.1 概要

プログラミング テンプレート "AxisInterface サンプル プロジェクト"は、AxisInterface のアプリケーション例として使用されます。

このサンプル プロジェクトは GAT ウィザードを使用していつでも生成できます。

1. 新しい IndraWorks プロジェクトを作成します。
2. コントロール IndraMotion MLC を追加します。
3. メニュー バーの左上にあるアイコンを使用して、GAT ウィザードを起動します。
4. AxisInterface サンプル プロジェクトを選択し、**Finish** (完了) をクリックしてインポートを開始します。

6.6.2 AxisInterface 用プログラム

サンプル プロジェクトのプログラム *TE_AxisInterfaceType01Prog* は以下のポイントを網羅します。

- **AxisInterface の初期化 :**
パラメータ化モード (フェーズ 2) に達すると、ファンクションブロック *TE_AxisInitAllAxisType01* を使用して設定済みのすべての軸について

AxisInterface の初期化が行われます。運転モード (フェーズ 4) に達すると、パラメータ化での変更を認識するために、初期化が再度実行されます。

- **AxisInterface の周期的な呼出：**

初期化が完了した後で、AxisInterface が周期的に呼び出されます。これを行うには、"TE_AxisInterface" ファンクションブロックを呼び出します。これは、ユーザー拡張を可能にするために、MB_AxisInterfaceType01 から派生します。

- **AxisInterface のためのユーザー拡張：**

"TE_AxisInterface" ファンクションブロックにはサンプルとしてプログラミングされたユーザー拡張が含まれています。ユーザーは、各アプリケーションをできる限りカバーするために、AxisInterface 構造に他の要素を追加できます。



プログラム TE_AxisInterfaceType01Prog は、AxisInterface を既存のプログラムに組み込むのに役立ちます。

新しいプロジェクトの場合は、軸インターフェースを作成するために GAT ウィザードのインポート機能を使用することを強くお勧めします。

AxisInterface を既存のプロジェクトに組み込むには、以下の操作が使用できます。

1. メニューバーの左上にあるアイコンを使用して、GAT ウィザードを起動します。
2. 表示された "Import template code" (インポート テンプレート コード) ダイアログで "AxisInterface" を選択します。
AxisInterface サンプル プロジェクトが作成されます。
3. これで、構造 "arAxisCtrl_gb"、"arAxisStatus_gb" および "AxisData" を使用して軸に接近できます。
4. コントロールを操作するには、"ImcCtrl" および "ImcStatus" が使用できます。

"PlcProg" プログラムで AxisInterface へのアクセスが示されます。このコードは、ユーザー自身のプロジェクトのために適宜適合を行う必要があります。



快適な軸コマンドのために、ダイアログベースの軸コマンドが利用可能です。このトピックに関する詳細は、[67 ページ "ダイアログベースの軸コマンド" 6.4.1 章](#) を参照してください。

6.6.3 AxisInterface の表示

概要

AxisInterface と IMC インターフェースは表示マスクを備えており、システム軸とコントロールの設定および制御のために既製の容易なインターフェースを提供します。

GAT および GAT compact には以下の表示が含まれています。

表示	説明
System_Overview_All_Axis	簡易診断、ステータスおよびジョグ操作要素を含むすべての定義済みの軸の完全な概要。
Axis_Overview	現在の軸の位置表示、速度表示、同期表示のためのナビゲーションと共に位置と速度の現在のコマンドおよび値を示します。軸をホームングしてスイッチをオフにすることも可能です。
Position_mode	位置設定モードのモニタリング
Velocity_mode	速度モードのモニタリング
Synchronization_mode	同期モードのモニタリング
Profile_mode	プロファイルによる同期化のモニタリング
ImcInterface_Status_Control	IndraMotion MLC コントロールおよび運転モード、応答、システム診断のステータス インターフェース。
PLC_load	AxisInterfaceType01Prog の最小、現在、および最大の PLC タスク サイクル時間を示します。

表 6-29: サンプル プロジェクトの表示

以下のグローバル変数は、表示内のシステム情報の制御およびアクセスに使用されます。

- arAxisCtrl_gb[]
- arAxisStatus_gb[]
- AxisData[] (MLC 標準)
- VisuAxisNo

システム概要表示

System_Overview_All_Axis 表示を用いると、ユーザーはプロジェクトで設定されている各軸を素早く接続できます。表示されているデフォルトの軸 (MyVirtualAxis1 および MyVirtualAxis2) は GAT compact サンプル プロジェクトの一部です。オフライン モードで表示するために、追加的な軸を追加できます。

この表示は全体的なシステム ステータスを示し、エラーをクリアすることを許可し、緊急停止の設定を可能にします。個別の軸操作要素は、運転モードステータス表示と共に軸の名前、診断、現在の位置と速度を提供します。SetupMode セクションでは、ユーザーは各軸のジョグと位置設定のほか、仮想軸の起動と停止を行うことができます。

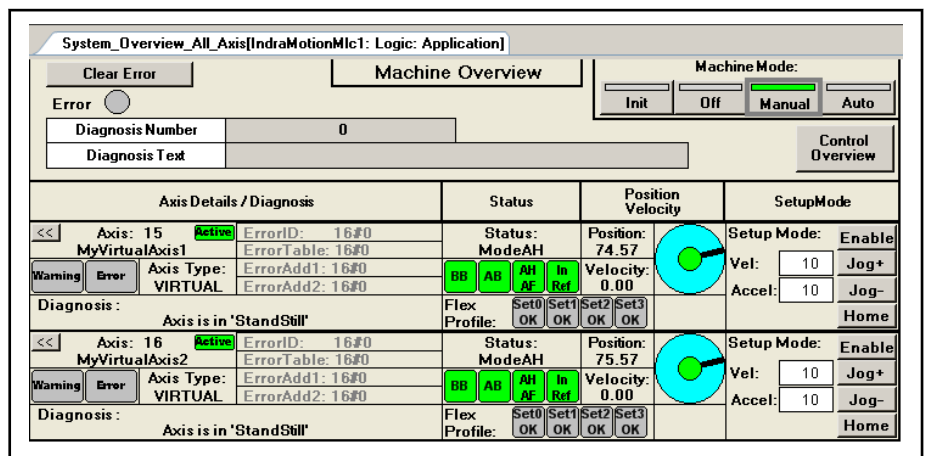


図 6-13: System_Overview_All_Axis 表示

- システム概要への軸の追加
- 新しい表示を追加するときシステム概要表示に軸が追加されます。これは各要素を選択して軸の名前と番号を指定することにより行われます。以下のステップにより、この手順を簡単に説明します。
1.

各 IndraWorks プロジェクトを開き、**Logic->Application->Visualizations->SystemOverview** タブを選択します。
2.

IndraLogic のオフライン モードで、System_Overview_All_Axis 表示をダブルクリックします。
3.

Toolbox (ツールボックス) ウィンドウで、**Frame (フレーム)** ボタンを選択し、軸の表の最終行の下に、適切な行の高さと幅でアウトラインを生成します。

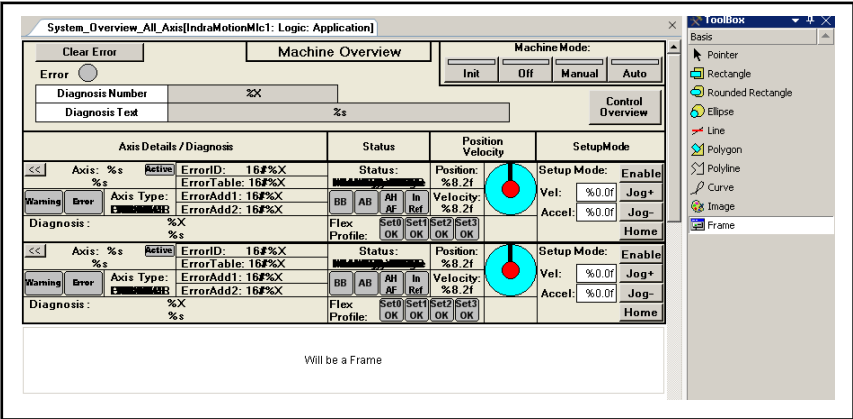


図 6-14: フレーム

4.
- フレームを右クリックし、"Frame selection" (フレーム選択) を有効にします。表示選択ウィンドウで **System_Overview** 要素を選択します。完全に新しいシステム概要 "Axis Line" が表示されます。
5.
- 新しい表示の内側をクリックし、"Properties" (プロパティ) ウィンドウで定義済みの軸の名前を **m_Input_AxisName** の値として入力します。
6.
- これで、システム概要の新しい AxisInterface のサイズ変更と位置設定を行うことができます。
7.
- IndraLogic プロジェクトを再コンパイルし、オンラインに接続します。



プロジェクトに追加されるすべての軸について、上述のステップを繰り返す必要があります。

- ナビゲーション システムの概要
- 軸の表の「Details」(詳細) 列にある **Axis Name** (軸の名前) ボタンをクリックし、各軸の概要表示を開きます。ユーザーは軸の概要表示から位置、速度および同期の運転モード表示に移動できます。

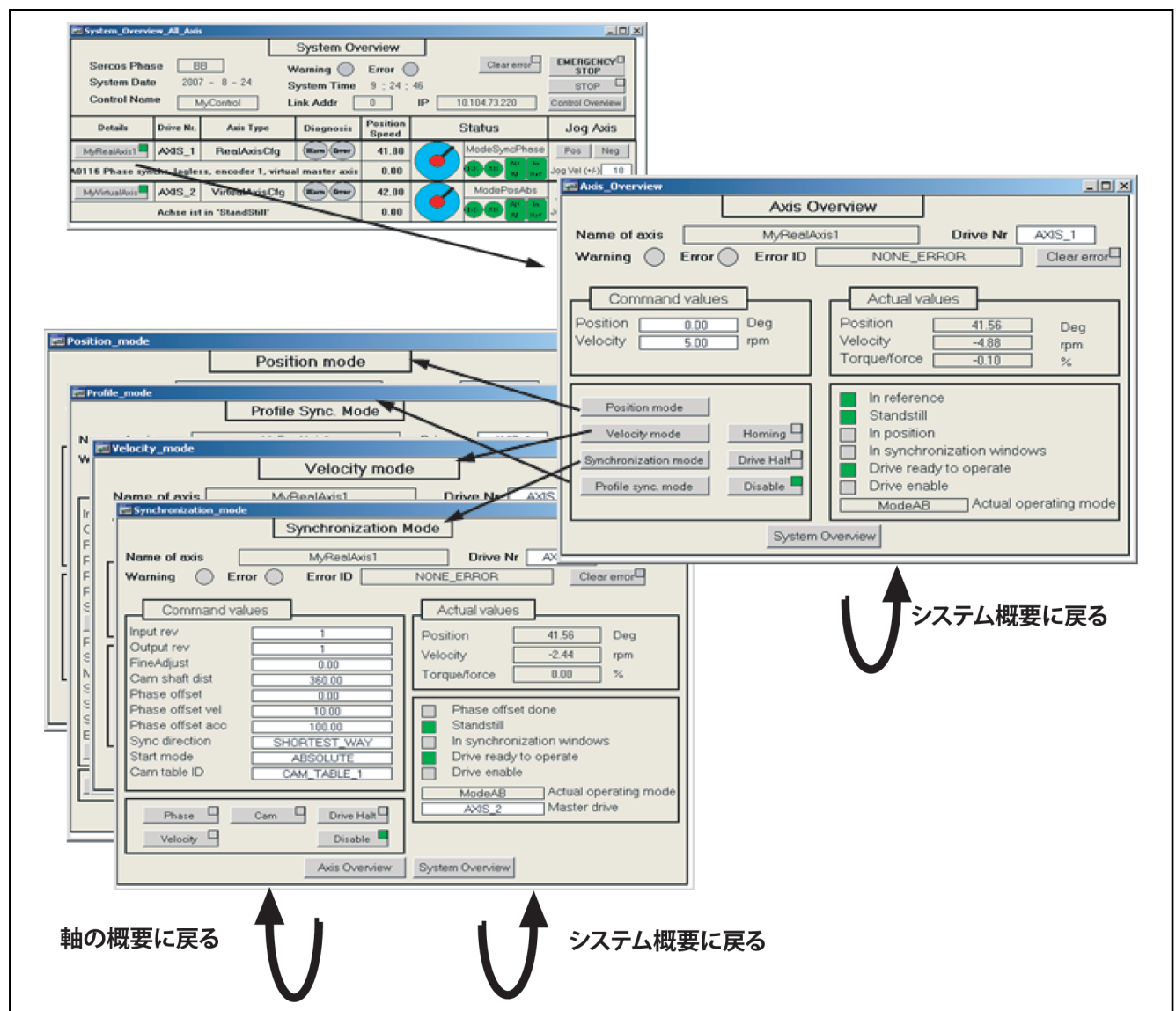


図 6-15: ナビゲーション システムの概要

Axis Overview (軸の概要) 表示

Axis Overview (軸の概要) 表示は、現在のコマンドおよび軸の実効値のためのシンプルなインターフェースを提供します。位置設定モードおよび速度モードのコマンド値が表示されます。ユーザーは **Drive No.** (ドライブ番号) フィールドをクリックしてそれぞれの軸番号を入力することにより、プロジェクト内の軸を切り替えることができます。

ML_TechInterface.library

The screenshot shows the 'Axis Overview' window. At the top, it displays 'Name of axis' as 'MyRealAxis1' and 'Drive Nr' as 'AXIS_1'. Below this are 'Warning' and 'Error' status indicators, both currently off, and an 'Error ID' field showing '0'. A 'Clear error' button is also present. The interface is divided into two main sections: 'Command values' and 'Actual values'. The 'Command values' section shows 'Position' at '0.00 Deg' and 'Velocity' at '5.00 rpm'. The 'Actual values' section shows 'Position' at '41.96 Deg', 'Velocity' at '-0.09 rpm', and 'Torque/force' at '0.00 %'. Below these are mode selection buttons: 'Position mode', 'Velocity mode', 'Synchronization mode', and 'Profile sync. mode'. To the right of these are 'Homing' and 'Drive Halt' buttons, and a 'Disable' button which is currently checked. A status panel on the right contains several green LEDs indicating 'In reference', 'Standstill', 'In position', 'In synchronization windows', 'Drive ready to operate', and 'Drive enable'. At the bottom, there is an 'Actual operating mode' field showing '0' and a 'System Overview' button.

図 6-16: Axis Overview (軸の概要) 表示

運転モード Positioning Mode (位置設定モード)、Velocity Control (速度制御)、Synchronization Mode (同期モード) および Synchronization mode with Profiles (プロファイルによる同期モード) のボタンを使用し、それぞれの運転モードに切り替えて、各運転モードのために特別な値を入力できます。現在の軸は、is can be homed, switched after トルクを維持 (AH) またはスイッチオフ (Ab) した状態で、Drive Halt (ドライバー時停止) の後でホーミングと切替ができます。

下の表は、表示の一部として使用される (一般的な) AxisInterface 構造要素をまとめたものです。

表示要素	AxisInterface の構造要素
Drive no. (ドライブ番号)	VisuAxisNo
Command (コマンド): Position (位置)	arAxisCtrl_gb[].PosMode.Position
Command (コマンド): Velocity (速度)	arAxisCtrl_gb[].PosMode.Velocity
Homing (ホーミング) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeHoming
Drive Halt (ドライバー時停止) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAh
Switch off (スイッチオフ) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAb
Current (現在): Position (位置)	AxisData[].rActualPosition_i
Current (現在): Velocity (速度)	AxisData[].rActualVelocity_i
Current (現在): Torque/force (トルク/力)	AxisData[].rActualTorqueForce_i
LED: Homed (ホーミング済み)	AxisData[].Axis_Homed
LED: At standstill (静止状態)	AxisData[].Axis_Standstill
LED: In-position (所定位置にあり)	AxisData[].Axis_InPosition
LED: In-synchronization window (同期ウィンドウにあり)	AxisData[].Axis_InSynchron
LED: Drive ready for operation (ドライブ運転準備完了)	AxisData[].Axis_InAb

表示要素	AxisInterface の構造要素
LED: Drive enabled (ドライブ有効)	AxisData[].Axis_Power
現在の運転モード	arAxisStatus_gb[].Admin._OpModeAck

表 6-30: Axis overview (軸の概要) の構造要素



arAxisCtrl_gb[] および arAxisStatus_gb[] 構造は ML_TechInterface ライブラリの一部です。AxisData [] 構造は ML_Base ライブラリの一部です。

Positioning Mode (位置設定モード) 表示

positioning mode (位置設定モード) 表示は以下のコマンド値を入力するために使用します。

- 位置
- 距離
- 速度
- 加速
- 減速

ユーザーは位置設定のタイプ (絶対、相対または追加) を選択できます。

図 6-17: Positioning Mode (位置設定モード) 表示

詳細については、以下の位置設定のタイプを参照してください。

絶対に関しては、73 ページ "Absolute Positioning (絶対位置設定)" 6.4.7 章を参照してください。

相対に関しては、74 ページ "Relative Positioning (相対位置設定)" 6.4.8 章を参照してください。

追加に関しては、75 ページ "Additive Positioning (追加位置設定)" 6.4.9 章を参照してください。

ML_TechInterface.library

下の表は、表示の一部として使用される (一般的な) AxisInterface 構造要素をまとめたものです。

表示要素	AxisInterface の構造要素
Drive no. (ドライブ番号)	VisuAxisNo
Command (コマンド): Position (位置)	arAxisCtrl_gb[].PosMode.Position
Command (コマンド): Distance (距離)	arAxisCtrl_gb[].PosMode.Distance
Command (コマンド): Velocity (速度)	arAxisCtrl_gb[].PosMode.Velocity
Command (コマンド): Acceleration (加速)	arAxisCtrl_gb[].PosMode.Acceleration
Command (コマンド): Deceleration (減速)	arAxisCtrl_gb[].PosMode.Deceleration
Drive Halt (ドライブ一時停止) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAh
Switch off (スイッチオフ) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAb
Current (現在): Position (位置)	AxisData[].rActualPosition_i
Current (現在): Velocity (速度)	AxisData[].rActualVelocity_i
Current (現在): Torque/force (トルク/力)	AxisData[].rActualTorqueForce_i
LED: At standstill (静止状態)	AxisData[].Axis_Standstill
LED: In-position (所定位置にあり)	AxisData[].Axis_InPosition
LED: Drive ready for operation (ドライブ運転準備完了)	AxisData[].Axis_InAb
LED: Drive enabled (ドライブ有効)	AxisData[].Axis_Power
Current operation mode (現在の運転モード)	arAxisStatus_gb[].Admin._OpModeAck

表 6-31: Positioning mode (位置設定モード) の構造要素

Velocity Mode (速度モード) 表示

velocity mode (速度モード) 表示は以下のコマンド値を入力するために使用します。

- 速度
- 加速
- 減速

図 6-18: 速度モードの表示

下の表は、表示の一部として使用される (一般的な) AxisInterface 構造要素をまとめたものです。

表示要素	AxisInterface の構造要素
Drive no. (ドライブ番号)	VisuAxisNo
Command (コマンド): Velocity (速度)	arAxisCtrl_gb[].VelMode.Velocity
Command (コマンド): Acceleration (加速)	arAxisCtrl_gb[].VelMode.Acceleration
Command (コマンド): Deceleration (減速)	arAxisCtrl_gb[].VelMode.Deceleration
Drive Halt (ドライブ一時停止) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAh
Switch off (スイッチオフ) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAb
Current (現在): Position (位置)	AxisData[].rActualPosition_i
Current (現在): Velocity (速度)	AxisData[].rActualVelocity_i
Current (現在): Torque/force (トルク/力)	AxisData[].rActualTorqueForce_i
LED: In velocity (速度)	AxisData[].Axis_InVelocity
LED: At standstill (静止状態)	AxisData[].Axis_Standstill
LED: In-position (所定位置にあり)	AxisData[].Axis_InPosition
LED: Drive ready for operation (ドライブ運転準備完了)	AxisData[].Axis_InAb
LED: Drive enabled (ドライブ有効)	AxisData[].Axis_Power
Current operation mode (現在の運転モード)	arAxisStatus_gb[].Admin._OpModeAck

表 6-32: Velocity mode (速度モード) の構造要素

Synchronization Mode (同期モード) 表示

Synchronization mode (同期モード) 表示は、ドライブの電子ギアの機能を制御するために使用されます。要件は、微調整とフェーズ オフセットの要素を使用することです。Drive No. (ドライブ番号) フィールドはスレーブ軸を定

ML_TechInterface.library

義し、**Master drive** (マスター ドライブ) フィールドはマスター軸を定義します。

ユーザーは同期モードのタイプ (フェーズ同期または速度同期、カムまたは MotionProfile) を選択できます。

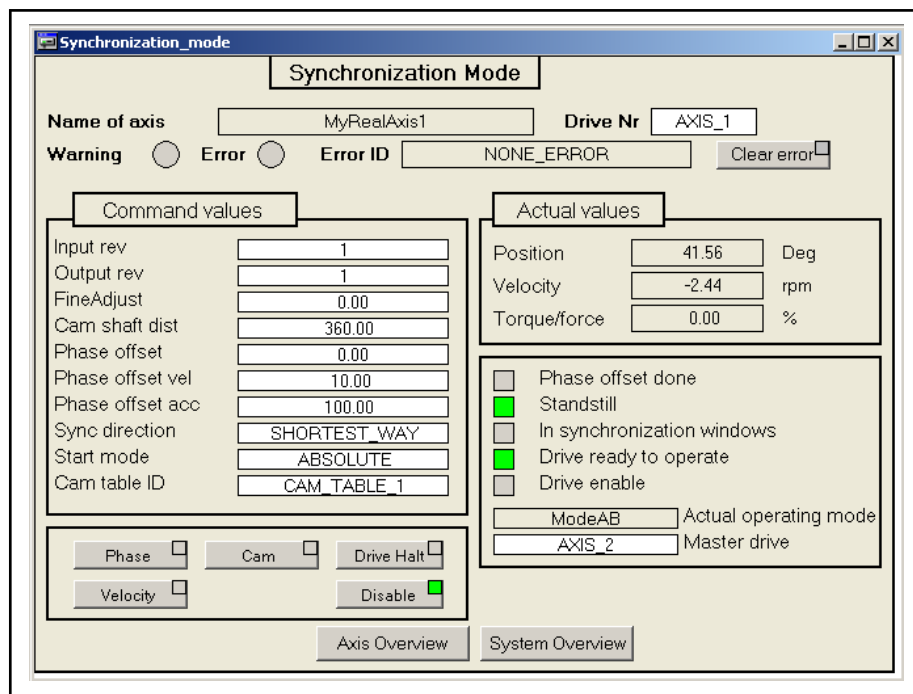


図 6-19: Synchronization mode (同期モード) 表示

詳細については、以下の同期のタイプを参照してください。

フェーズ同期に関しては、[76 ページ "運転モード Phase Synchronization \(フェーズ同期\)" 6.4.11 章](#) を参照してください。

速度同期に関しては、[77 ページ "運転モード Velocity Synchronization \(速度同期\)" 6.4.12 章](#) を参照してください。

カムに関しては、[77 ページ "運転モード Electronic Cam \(電子カム\)" 6.4.13 章](#) を参照してください。

下の表は、表示の一部として使用される (一般的な) AxisInterface 構造要素をまとめたものです。

表示要素	AxisInterface の構造要素
Drive no. (ドライブ番号)	VisuAxisNo
Command (コマンド): Input revolutions (入力回転数)	arAxisCtrl_gb[].SyncMode.InputRevolution
Command (コマンド): Output revolutions (出力回転数)	arAxisCtrl_gb[].SyncMode.InputRevolution
Command (コマンド): Fine adjustment (微調整)	arAxisCtrl_gb[].SyncMode.Fineadjust
Command (コマンド): Cam shaft stroke (カム シャフトのストローク)	arAxisCtrl_gb[].SyncMode.CamShaftDistance
Command (コマンド): Phase offset (フェーズ オフセット)	arAxisCtrl_gb[].SyncMode.PhaseOffset
Command (コマンド): Phase offset velocity (フェーズ オフセット速度)	arAxisCtrl_gb[].SyncMode.PhaseOffsetVel
Command (コマンド): Phase offset acceleration (フェーズ オフセット加速)	arAxisCtrl_gb[].SyncMode.PhaseOffsetAcc

表示要素	AxisInterface の構造要素
Command (コマンド): 同期方向	arAxisCtrl_gb[].SyncMode.SyncDirection
Command (コマンド): Synchronization startup mode (同期起動モード)	arAxisCtrl_gb[].SyncMode.StartMode
Command (コマンド): Cam ID (カム ID)	arAxisCtrl_gb[].SyncMode.CamTableID
Phase (フェーズ) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeSyncPhase
Velocity (速度) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeSyncVelocity
Cam (カム) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeSyncCam
Drive Halt (ドライブ一時停止) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAh
Switch off (スイッチオフ) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAb
Current (現在): Position (位置)	AxisData[].rActualPosition_i
Current (現在): Velocity (速度)	AxisData[].rActualVelocity_i
Current (現在): Torque/force (トルク/力)	AxisData[].rActualTorqueForce_i
LED: Phase offset executed (フェーズ オフセット実行済み)	arAxisStatus_gb[].SyncMode.PhasingSlaveDone
LED: At standstill (静止状態)	AxisData[].Axis_Standstill
LED: In-synchronization window (同期ウィンドウにあり)	AxisData[].Axis_InSynchron
LED: Drive ready for operation (ドライブ運転準備完了)	AxisData[].Axis_InAb
LED: Drive enabled (ドライブ有効)	AxisData[].Axis_Power
Current operation mode (現在の運転モード)	arAxisStatus_gb[].Admin._OpModeAck
Master axis drive (マスター軸ドライブ)	arAxisCtrl_gb[].SyncMode.Master.AxisNo

表 6-33: Synchronization mode (同期モード) の構造要素

表示、Synchronization Mode with Profiles

表示 "synchronization mode with profiles" (プロファイルによる同期モード) は、プロファイルによりドライブの電子ギアの機能を制御するために使用されます。要件は、微調整とフェーズ オフセットの要素を使用することです。**Drive No.** (ドライブ番号) フィールドはスレーブ軸を定義し、**Master drive** (マスター ドライブ) フィールドはマスター軸を定義します。

プロファイルによる同期モードのタイプ (FlexProfile または MotionProfile) 選択します。

ML_TechInterface.library

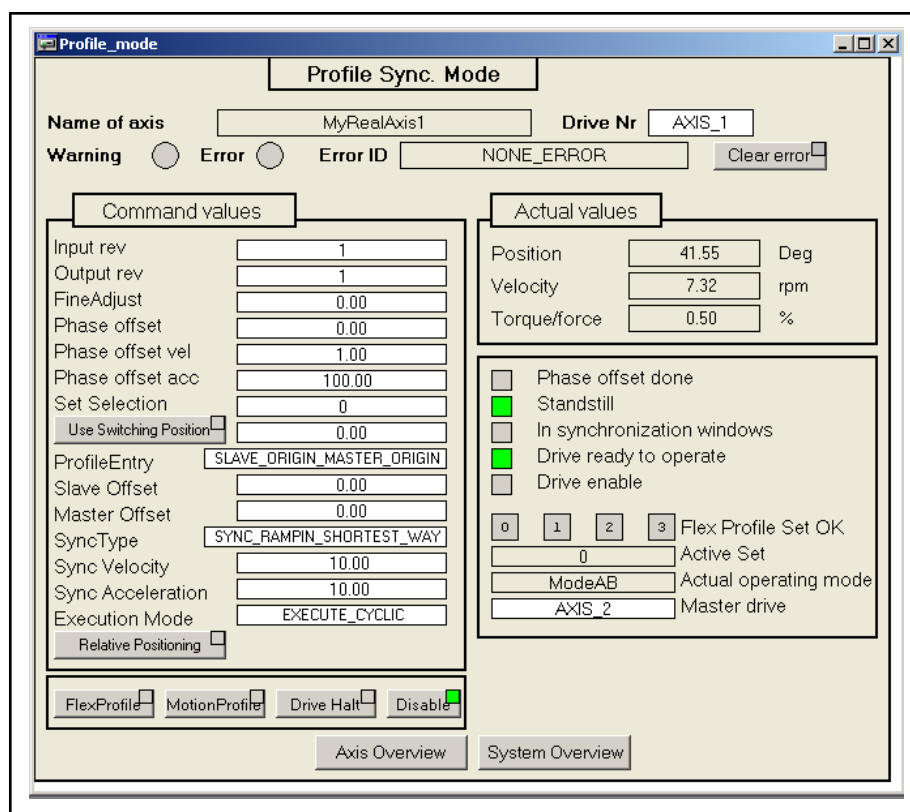


図 6-20: 表示 - synchronization mode with profiles

詳細については、以下の同期のタイプを参照してください。

MotionProfile に関しては、78 ページ "運転モード Electronic MotionProfile" 6.4.14 章を参照してください。

FlexProfile に関しては、79 ページ "運転モード Electronic FlexProfile" 6.4.15 章を参照してください。

下の表は、表示の一部として使用される (一般的な) AxisInterface 構造要素をまとめたものです。

表示要素	AxisInterface の構造要素
Drive no. (ドライブ番号)	VisuAxisNo
Command (コマンド): Input revolutions (入力回転数)	arAxisCtrl_gb[].SyncMode.InputRevolution
Command (コマンド): Output revolutions (出力回転数)	arAxisCtrl_gb[].SyncMode.InputRevolution
Command (コマンド): Fine adjustment (微調整)	arAxisCtrl_gb[].SyncMode.Fineadjust
Command (コマンド): Phase offset (フェーズ オフセット)	arAxisCtrl_gb[].SyncMode.PhaseOffset
Command (コマンド): Phase offset velocity (フェーズ オフセット速度)	arAxisCtrl_gb[].SyncMode.PhaseOffsetVel
Command (コマンド): Phase offset acceleration (フェーズ オフセット加速)	arAxisCtrl_gb[].SyncMode.PhaseOffsetAcc
Command (コマンド): Profile set selection (プロファイル セットの選択)	arAxisCtrl_gb[].SyncMode.Profile.SetSelection
Relative Positioning (相対位置設定) ボタン	arAxisCtrl_gb[].SyncMode.Profile.RelativePositioning
MotionProfile ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeSyncProfile

表示要素	AxisInterface の構造要素
FlexProfile ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeFlexProfile
Drive Halt (ドライバー時停止) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAh
Switch off (スイッチオフ) ボタン	arAxisCtrl_gb[].Admin._OpMode: = ModeAb
Current (現在): Position (位置)	AxisData[].rActualPosition_i
Current (現在): Velocity (速度)	AxisData[].rActualVelocity_i
Current (現在): Torque/force (トルク/力)	AxisData[].rActualTorqueForce_i
LED: Phase offset executed (フェーズ オフセット実行済み)	arAxisStatus_gb[].SyncMode.PhasingSlaveDone
LED: At standstill (静止状態)	AxisData[].Axis_Standstill
LED: In-synchronization window (同期ウィンドウにあり)	AxisData[].Axis_InSynchron
LED: Drive ready for operation (ドライブ運転準備完了)	AxisData[].Axis_InAb
LED: Drive enabled (ドライブ有効)	AxisData[].Axis_Power
Current operation mode (現在の運転モード)	arAxisStatus_gb[].Admin._OpModeAck
Master axis drive (マスター軸ドライブ)	arAxisCtrl_gb[].SyncMode.Master.AxisNo

表 6-34: プロファイルと構造要素による同期操作

IMC インターフェース コントロールとステータスの表示

ImcInterface_Status_Control 表示は、パラメータ モードと運転モードの切替、エラーのクリアおよびフェーズ切替の再起動などの制御機能へのアクセスを許可します。また、この表示はさまざまな sercos 運転モードと診断の制御ステータスも提供します。

ML_TechInterface.library

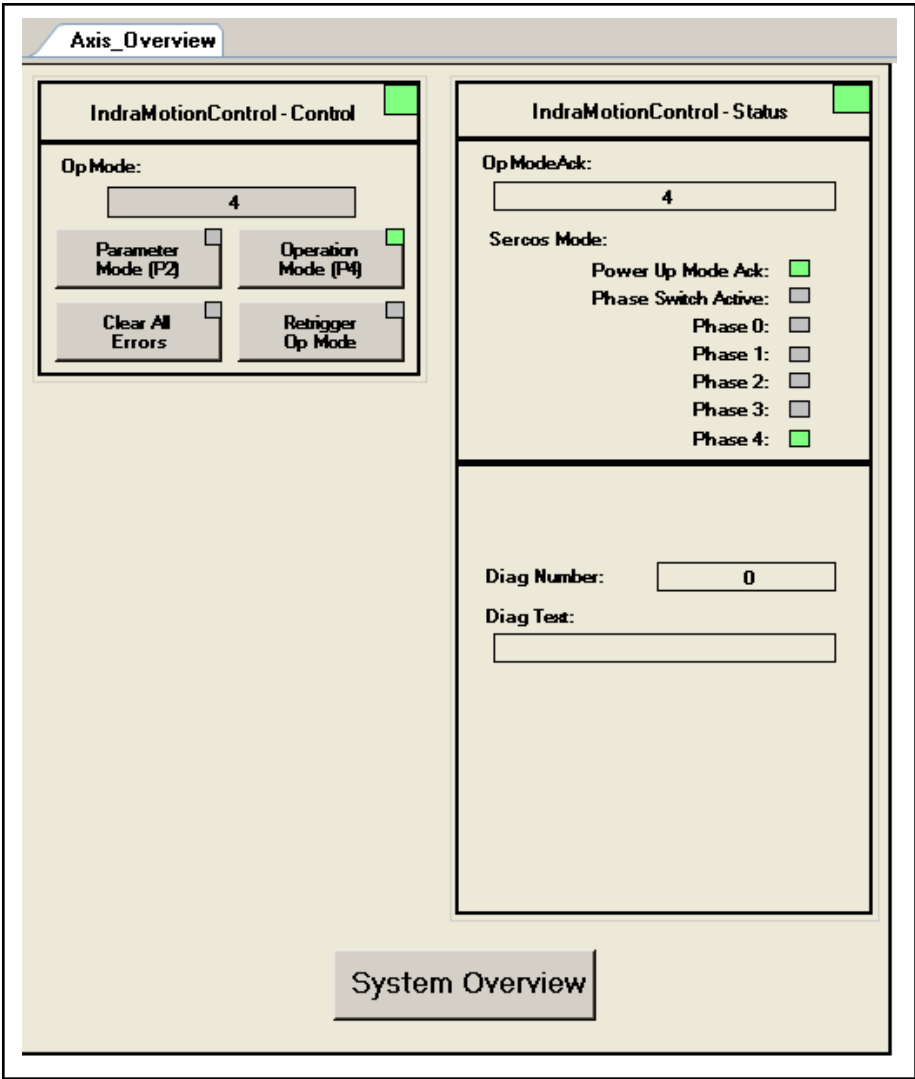


図 6-21: ImcInterface ステータスとコントロールの表示

下の表は、表示の一部として使用される (一般的な) AxisInterface 構造要素をまとめたものです。

表示要素	AxisInterface の構造要素
IndraMotion Control - コントロール	
Phase	ImcCtrl.Admin._OpMode
Parameterization mode (パラメータ化モード) (P2)	ImcCtrl.Admin._OpMode: = IMC_MODE_CTRL_P2
Operating mode (運転モード) (P4)	ImcCtrl.Admin._OpMode: = IMC_MODE_CTRL_BB
Clear all errors (すべてのエラーをクリア)	ImcCtrl.Admin.ClearError
Restart phase switching (フェーズ切替を再起動)	ImcCtrl.Admin.RetriggerOpMode
IndraMotion Control - ステータス	
Phase acknowledgement (フェーズ応答)	ImcStatus.Admin._OpModeAck
LED: Switch-on mode acknowledgement (モード応答のスイッチオン)	ImcStatus.Admin.PwrUpModeAck
LED: Phase switching active (フェーズ切替有効)	ImcStatus.Admin.PhaseSwitchActive

表示要素	AxisInterface の構造要素
LED: Phase (フェーズ) 0	ImcStatus.Admin.ModeStatus_P0
LED: Phase (フェーズ) 1	ImcStatus.Admin.ModeStatus_P1
LED: Phase (フェーズ) 2	ImcStatus.Admin.ModeStatus_P2
LED: Phase (フェーズ) 3	ImcStatus.Admin.ModeStatus_P3
LED: Phase (フェーズ) 4	ImcStatus.Admin.ModeStatus_BB
LED: Clear error acknowledgement (クリア エラー確認)	ImcStatus.Diag.ClearErrorAck
LED: エラー	ImcStatus.Diag.Error
LED: 警告	ImcStatus.Diag.Warning
ErrorID	ImcStatus.Diag.ErrorID
Error identification (エラーの識別): Table (表)	ImcStatus.Diag.ErrorIdent.Table
Error identification (エラーの識別): Additional 1	ImcStatus.Diag.ErrorIdent.Additional1
Error identification (エラーの識別): Additional2	ImcStatus.Diag.ErrorIdent.Additional2

表 6-35: IMC インターフェースの構造要素

PLC Load (PLC 負荷) 表示

PLC Load (PLC 負荷) 表示は PLC_Load をモニタリングするために使用されます。TE_AxisInterfaceType01Prog 内の AxisInterface 機能ブロックの周期的呼出のための最小、現在および最大の PLC サイクル時間が表示されます。

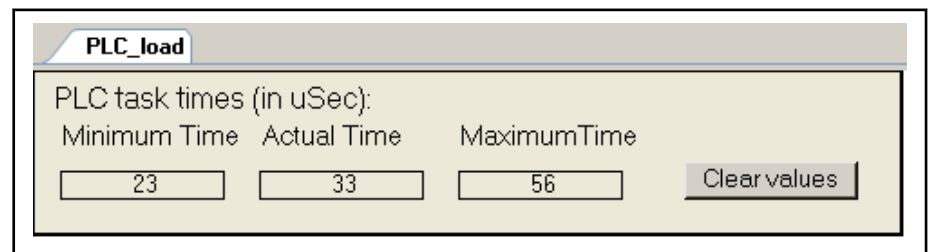


図 6-22: PLC load (PLC 負荷) 表示

6.7 AxisInterface のユーザー拡張

6.7.1 概要

ユーザーが `arAxisCtrl_gb[ ]` および `arAxisStatus_gb[ ]` 構造を拡張し、AxisInterface を特別なアプリケーションに適合させることができます。



ユーザー インターフェースの `arAxisCtrl_gb[ ]` および `arAxisStatus_gb[ ]` 構造は

MB_Axis_Control_Type01 および MB_Axis_Status_Type01 データ構造です (102 ページ "AxisInterface の構造" 6.8 章を参照)。

AxisInterface のデータ構造は ML_TechInterface ライブラリに統合されており、それゆえにユーザーはアクセスできないため、機能を追加できるようにするには、既存のデータ構造に対応する新しいデータ構造を、現在のプロジェクト内に作成する必要があります。

このセクションでは、MB_Phasing 機能を追加することにより AxisInterface を拡張する方法について説明します。MB_Phasing ファンクションブロック

ML_TechInterface.library

を利用すると、マスター軸にオフセットを追加できます。マスター軸は **MC_CamIn** または **MC_GearInPos** 同期ファンクションブロックへの入力として接続されます。

MB_PhasingSlave 機能は、

`arAxisCtrl_gb[ ].SyncMode.PhaseOffset`、`(.PhaseOffsetVel)`、`(.PhaseOffsetAcc)` 要素および

`arAxisStatus_gb[ ].SyncMode.PhasingSlaveDone` 出力を使用して、**AxisInterface** に統合されます。

以下の新しい入力と出力が定義されます。

- `arAxisCtrl_gb[ ].SyncMode.PhaseShift`
- `arAxisCtrl_gb[ ].SyncMode.PhaseShiftVel`
- `arAxisCtrl_gb[ ].SyncMode.PhaseShiftAcc`
- `arAxisStatus_gb[ ].SyncMode.PhasingMasterDone`



"`arAxisCtrl_gb[ ].SyncMode.PhaseShift`" 入力は周期的にスキャンされ、変更があった場合は直ちに有効化されます。

GAT および GAT ^{compact} には以下のユーザー固有の拡張が実装されています。

- ジョグ機能とホーミングが "SetupMode" として
- パラメータ A-0-2730 "表のフェーズ オフセット 開始" が "`SyncMode.AngleOffset`" として
- パラメータ A-0-2790 "同期速度" が "`SyncMode.SyncVelocity`" として
- パラメータ A-0-2791 "同期加速" が "`SyncMode.SyncAcceleration`" として
- ファンクションブロック **MB_Phasing()** が `SyncMode.PhaseShift`、`-.PhaseShiftVel` および `-.PhaseShiftAcc` として

以下においては、GAT または GAT ^{compact} で準備された構造が使用されることを前提とします。以下では、そこで指定されている POU で必要となる変更について説明します。

ユーザー拡張の実装に関する注意

ユーザー拡張は IndraLogic 2G からのオブジェクト指向の拡張を使用して実装されました。以下の特殊性について留意してください。

- ファンクションブロック (FB) **TE_AxisInterface** は **Basis FB MB_AxisInterfaceType01** から派生します。キーワード "SUPER" を使用して、**Basis FB** または **Basis FB** に関連するメソッド/アクションを呼び出すことができます。例えば、さまざまな場所で、診断システムでエラーを入力するために `SUPER^.mSetError(...)` を使用して **FB MB_AxisInterfaceType01** のメソッド `mSetError` が呼び出されます。
- **FB TE_AxisInterface** 内で、`arAxisCtrl_gb` のデータは入力 `AdminCtrlExt`、`PosModeCtrlExt` などを使用してアクセスでき、`arAxisStatus_gb` のデータは `AdminStatusExt`、`DiagStatusExt` などを使用してアクセスできます。入力 `AdminCtrl` (Ext なし) などは基本 FB に属するため、使用されません。
- **FB TE_AxisInterface** の入力は "REFERENCE TO" として定義されます。`mlnitExtension` メソッドで、参照は 1 回初期化され、FB の周期的な呼出中に再び転送する必要はありません。
- 基本構造 `TE_AXIS_CONTROL_TYPE01` および `TE_AXIS_STATUS_TYPE01` が追加的な下部構造により拡張される場合、下に挙げられている手順に以下の追加ステップを追加する必要があります (例として `SetupMode` および `SetupModeAck` を参照)。

- TE_AXIS_CONTROL_TYPE01 および TE_AXIS_STATUS_TYPE01 で新しい要素を入力する
- FB TE_AxisInterface に追加的な入力を REFERENCE TO として追加する
- FB TE_AxisInterface のメソッド mInitExtension の参照を初期化する

6.7.2 arAxisCtrl_gb[] 構造の拡張

MB_Axis_Control_Type01 構造の追加機能を追加するには、ユーザーは新しい構造を作成し、既存の構造から下位構造を派生させてから、新しい要素を追加する必要があります。 GAT/GAT_{compact} には、構造 TE_AXIS_CONTROL_TYPE01 および下位構造 TE_AXIS_ADMINISTRATION などがすでに用意されています。

必要に応じて以下のステップに進み、TE_AXIS_SYNCHRONISATION 構造の機能を拡張します。

1. IndraWorks で各プロジェクトを開きます。
2. TE_AXIS_SYNCHRONISATION 構造を開きます。

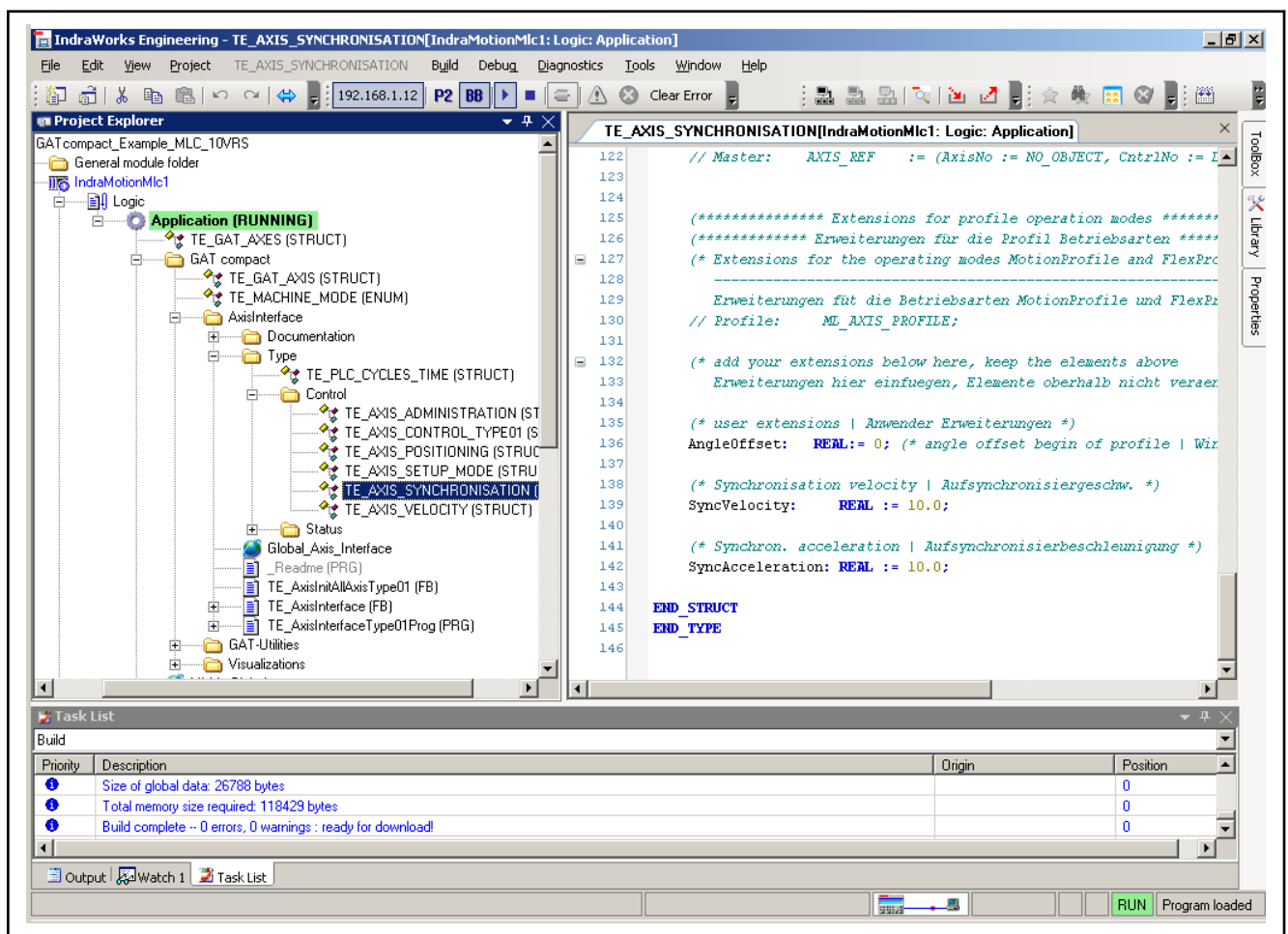


図 6-23: TE_AXIS_SYNCHRONISATION 構造を開く

3. 以下の 3 つの新しい要素割当

- PhaseShift
- PhaseShiftVel
- PhaseShiftAcc

ML_TechInterface.library

を、"TE_AXIS_SYNCHRONISATION" でコメントの下に REAL タイプとして追加します。

```

132      (* add your extensions below here, keep the elements above
133      Erweiterungen hier einfuegen, Elemente oberhalb nicht veraendern *)
134
135      PhaseShift:    REAL:=1.0;    // PhaseShift for MB_Phasing, cyclic scanned
136      PhaseShiftVel: REAL:=10.0;   // Velocity for MB_Phasing
137      PhaseShiftAcc: REAL:=100.0;  // PhaseShift for MB_Phasing
138

```

図 6-24: TE_AXIS_SYNCHRONISATION に追加された新しい要素

6.7.3 arAxisStatus_gb[] 構造の拡張

arAxisCtrl_gb[] 構造の拡張のステップを使用して、"arAxisStatus_gb[]" 構造を拡張します。以下のステップにより、この手順を簡単に説明します。

1. オブジェクト TE_AXIS_SYNC_STATUS を開きます。
2. 次の新しい要素割当 **PhasingMasterDone** を BOOL タイプとして TE_AXIS_SYNC_STATUS でコメントの下に追加します。

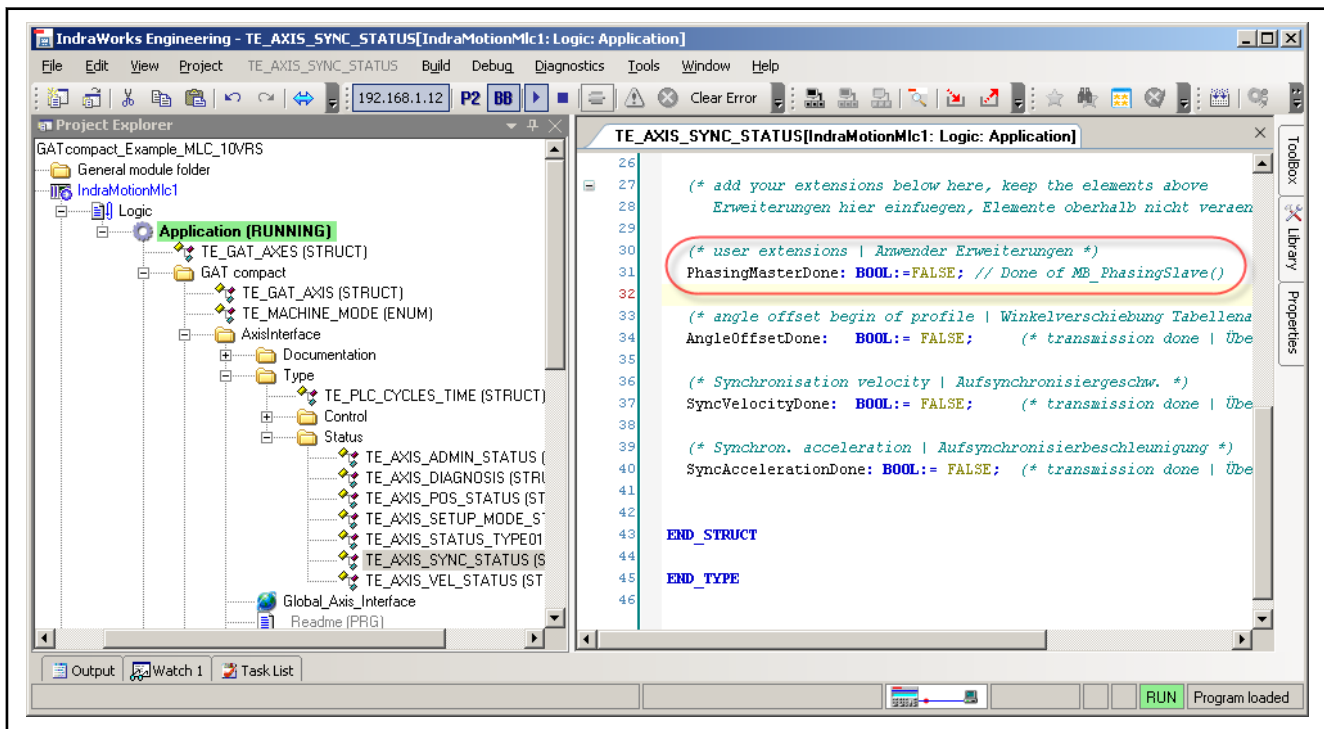


図 6-25: TE_AXIS_SYNC_STATUS に追加された新しい要素

6.7.4 ファンクションブロックの拡張

ユーザー拡張シーケンスの最後のステップは、新しい要素を使用できるようにファンクションブロックを拡張することです。

1. プロジェクト ツリーで、右クリックして **TE_AxisInterface** および **Add -> Action** を選択します。
2. 新しいアクション **actPhaseShift** に名前を付けます。
3. FB **TE_AxisInterface** で以下の変数を宣言します。
 - fbAxisPhasing: MB_Phasing;
 - bExecPhasing: BOOL := FALSE;

- rOldPhaseShift: REAL:= 0.01; // 転送を強制するために、古いデフォルト値は 0 に等しくない

4. アクションはこの方法でプログラミングできます。

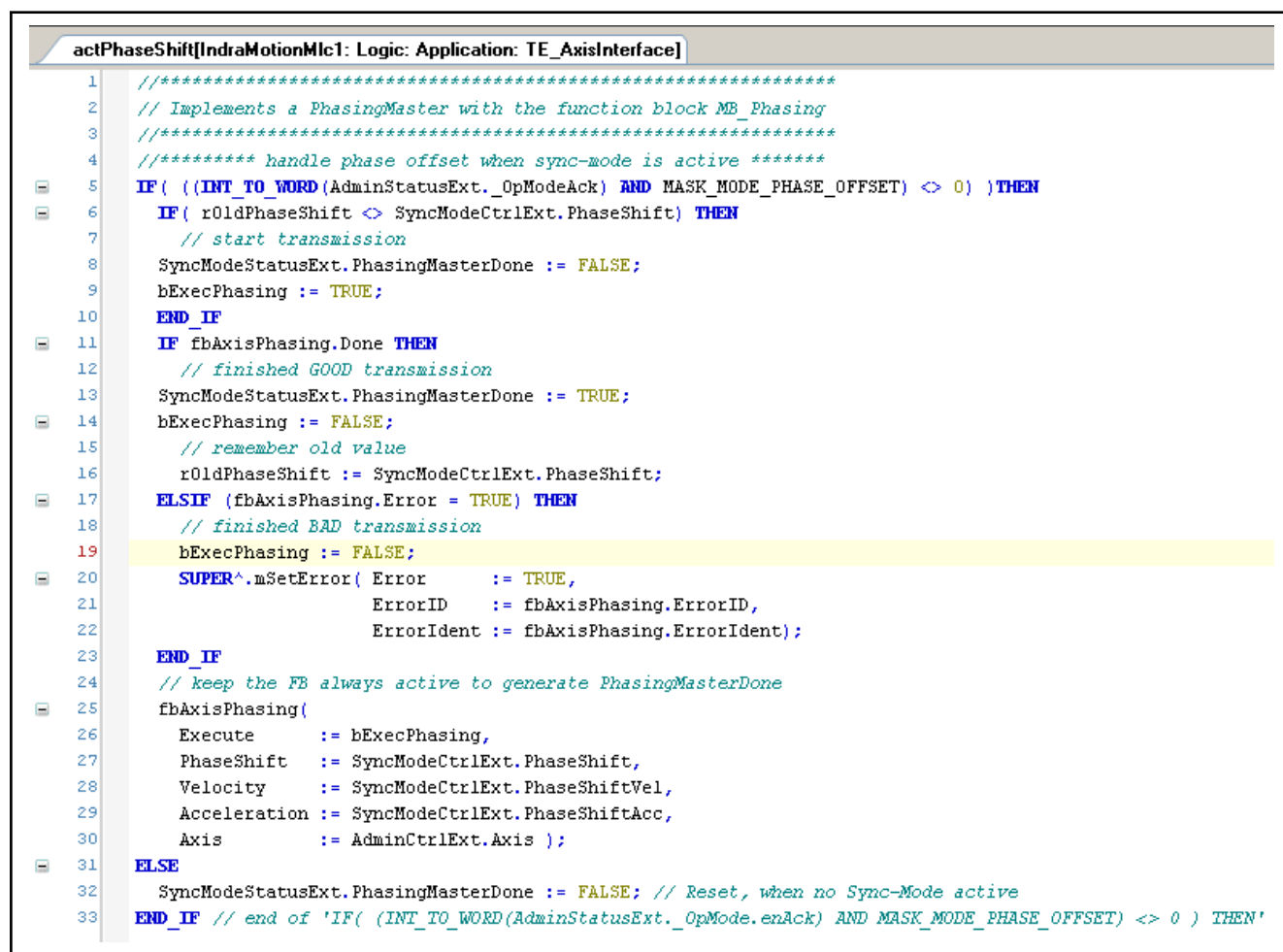


図 6-26: ユーザープログラムの拡張

5. FB TE_AxisInterface で新しいアクション "actPhaseShift()" を呼び出します。
6. プロジェクトを再度コンパイルし、プログラム エラーがないかチェックします。
7. コントロールにプロジェクトをロードします。

これで、新しい入力および出力要素が AxisInterface 構造の一部となり、UserVarGlobal 変数を使用して表示できます。

ML_TechInterface.library

◆ FineAdjust	REAL	0
◆ CamShaftDistance	REAL	360
◆ PhaseOffset	REAL	0
◆ PhaseOffsetVel	REAL	1
◆ PhaseOffsetAcc	REAL	100
◆ SyncDirection	MC_SYNC_DIRECTION	MC_SYNC_DIRECTI...
◆ StartMode	MC_START_MODE	MC_START_MODE.A...
◆ CamTableID	MC_CAM_ID	MC_CAM_ID,CAM_T...
⊕ ◆ Master	AXIS_REF	
⊕ ◆ Profile	ML_AXIS_PROFILE	
⊕ ◆ AdjFineAdjust	MB_AXIS_ADJUST_DATA	
⊕ ◆ AdjCamShaftDistance	MB_AXIS_ADJUST_DATA	
⊕ ◆ AdjPhaseOffset	MB_AXIS_ADJUST_DATA	
◆ AngleOffset	REAL	0
◆ PhaseShift	REAL	0
◆ PhaseShiftVel	REAL	10
◆ PhaseShiftAcc	REAL	100
◆ SyncVelocity	REAL	10
◆ SyncAcceleration	REAL	10
⇒ IndraMotionMlc1.Application.UserVarGlobal.arAxisStatus_gb[1]	TE_AXIS_STATUS_TYPE01	
⊕ ◆ Admin	TE_AXIS_ADMIN_STATUS	
⊕ ◆ Diag	TE_AXIS_DIAGNOSIS	
⊕ ◆ PosMode	TE_AXIS_POS_STATUS	
⊕ ◆ VelMode	TE_AXIS_VEL_STATUS	
⊕ ◆ SyncMode	TE_AXIS_SYNC_STATUS	
◆ CamActiveID	MC_CAM_ID	16#0000
◆ ProfileActiveSet	UINT	16#0000
◆ PhasingSlaveDone	BOOL	FALSE
⊕ ◆ AdjFineAdjust	MB_AXIS_ADJUST_STATUS	
⊕ ◆ AdjCamShaftDistance	MB_AXIS_ADJUST_STATUS	
⊕ ◆ AdjPhaseOffset	MB_AXIS_ADJUST_STATUS	
◆ AngleOffsetDone	BOOL	TRUE
◆ SyncVelocityDone	BOOL	TRUE
◆ SyncAccelerationDone	BOOL	TRUE
◆ PhasingMasterDone	BOOL	FALSE
⊕ ◆ PtrAxisData	POINTER TO ML_AXISDAT...	16#0A6B1B94
⊕ ◆ SetupMode	TE_AXIS_SETUP_MODE_S...	

図 6-27: ユーザー拡張を伴う arAxisCtrl_gb および arAxisStatus_gb 構造

6.8

AxisInterface の構造

6.8.1

概要

AxisInterface は、簡単な軸制御のためのデータ インターフェースを提供します。AxisInterface のデータ構造は、以下のセクションで説明されています。

プログラミング テンプレート GAT (Generic Application Template、汎用アプリケーション テンプレート) または GAT compact 内の AxisInterface は、ユーザーが拡張できるように準備されており、いくつかの拡張がすでに組み込まれています。これらの拡張を有効にするためには、ライブラリ構造を拡張するユーザー プロジェクトでユーザー自身の構造を定義する必要があります。拡張された構造は、プレフィックス "TE-" で識別します。例えば、GAT または GAT compact 内の構造が TE_AXIS_ADMIN_STATUS と呼ばれる場合、これは MB_AXIS_ADMIN_STATUS の拡張構造で、この構造に関する情報は、MB_AXIS_ADMIN_STATUS という名前の項目にあります。

下の表は、AxisInterface データ構造のユーザー インターフェースを示したものです。

ユーザー インターフェース	タイプ	説明
arAxisCtrl_gb[]	MB_AXIS_CONTROL_TYPE01 (GAT:TE_AXIS_CONTROL_TYPE01) MB_AXIS_CONTROL_TYPE02 (GAT:TE_AXIS_CONTROL_TYPE02)	運転モードを有効にするためのコマンド値や変数などの制御構造
arAxisStatus_gb[]	MB_AXIS_STATUS_TYPE01 (GAT:TE_AXIS_CONTROL_TYPE01) MB_AXIS_STATUS_TYPE02 (GAT:TE_AXIS_CONTROL_TYPE02)	運転モードの診断情報や確認などのステータス構造
AxisData[]		実際の値、ステータス ビットおよび周期的に設定可能なユーザー データ (ファームウェアで MotionControl により処理される) へのアクセスを含む、ステータス構造および制御構造

表 6-36: AxisInterface – ユーザー インターフェースのデータ構造

6.8.2 MB_AXIS_STATUS_TYPE01 構造

(MB_AXIS_STATUS_TYPE01 内)

概略 MB_AXIS_STATUS_TYPE01 データ構造は arAxisStatus_gb[] インスタンスを使用してアクセスされます。これには以下の要素が含まれています。

```

TYPE MB_AXIS_STATUS_TYPE01 :
STRUCT
  (***** Outputs of the following axis - from motion control to logic control *****)

  Admin:    MB_AXIS_ADMIN_STATUS; (* status of the axis | Achs-Status *)

  Diag:     MB_AXIS_DIAGNOSIS; (* axis diagnosis information, error, warning....
    Diagnose Informationen der Achse, Fehler, Warnung.... *)

  PosMode:  MB_AXIS_POS_STATUS; (* Status operating mode positioning (absolute, rel. and add.)
    Statusmeldungen Positionierbetrieb (absolut, relativ und additiv) *)

  VelMode:  MB_AXIS_VEL_STATUS; (* Status operating mode velocity control
    Statusmeldungen Geschwindigkeitsregelung (Freilauf) *)

  SyncMode: MB_AXIS_SYNC_STATUS; (* Status operating mode synchronisation (phase, vel, and cam)
    Statusmeldungen Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvensch.)*

  PtrAxisData: POINTER TO ML_AXISDATA_SM;
    (* Pointer to AxisData of this axis | Zeiger auf AxisData dieser Achse
    - Type .PtrAxisData^. for the intellisense menu | Eingabe .PtrAxisData^. fuer Auswahlmenue*)

  (* add your extensions below here, keep the elements above
    Erweiterungen hier einfuegen, Elemente oberhalb nicht veraendern *)

  (* user extensions | Anwender Erweiterungen *)

END_STRUCT
END_TYPE

```

図 6-28: (MB_AXIS_STATUS_TYPE01 内)

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

ML_TechInterface.library

- 依存関係
- IndraMotion MLD 用 MX_ImcInterface.lib
- System
- IndraMotion MLC/IndraMotion MLD

MB_AXIS_STATUS_TYPE01 の要素

MB_AXIS_ADMIN_STATUS 構造

- 概略
- MB_AXIS_ADMIN_STATUS 構造は (MB_AXIS_STATUS_TYPE01 内) の下位構造で、出力ステータスが含まれています。
下の図は、現在の MB_AXIS_ADMIN_STATUS 構造を示したものです。

```
MB_AXIS_ADMIN_STATUS
4  STRUCT
5  _OpModeAck: MB_AXIS_MODE:= ModeAb; (* acknowledge of the operating mode | Quittierung der Betriebsart
6  - check ... OpModeAck = ModeAB, ModeAH, ModePosAbs .... TO read the active operating mode
7  - OR use bits ....MODE_AH, ....MODE_POS_ABS ....
8  -----
9  - Lesen der aktiven Betriebsart: ... OpModeAck = ModeAB, ModeAH, ModePosAbs ....
10 - oder BIT-Auswertung ...MODE_AH, ...MODE_POS_ABS .... *)
11 (* bit access to the OpMode variables *)
12 {attribute 'relative_offset':='0'}
13 (* AH, under torque OpMode(Ack).0 *)
14 MODE_AH :BIT := FALSE;
15 MODE_HOMING :BIT := FALSE; (* AF, MB_Home OpMode(Ack).1 *)
16 MODE_POS_ABS :BIT := FALSE; (* AF, MC_MoveAbsolute OpMode(Ack).2 *)
17 MODE_POS_REL :BIT := FALSE; (* AF, MC_MoveRelative OpMode(Ack).3 *)
18 MODE_POS_ADD :BIT := FALSE; (* AF, MC_MoveAdditive OpMode(Ack).4 *)
19 MODE_VEL :BIT := FALSE; (* AF, MC_MoveVelocity OpMode(Ack).5 *)
20 MODE_SYNC_PHASE :BIT := FALSE; (* AF, MB_GearInPos OpMode(Ack).6 *)
21 MODE_SYNC_VEL :BIT := FALSE; (* AF, MC_GearIn OpMode(Ack).7 *)
22 MODE_SYNC_CAM :BIT := FALSE; (* AF, MC_CamIn OpMode(Ack).8 *)
23 MODE_SYNC_PROFILE :BIT := FALSE; (* AF, MB_MotionProfile OpMode(Ack).9 *)
24 { attribute 'hide' } MODE_TORQUE :BIT := FALSE; (* AF, MC_TorqueControl OpMode(Ack).10 not implemented *)
25 MODE_FLEX_PROFILE :BIT := FALSE; (* AF, MB_FlexProfile OpMode(Ack).11 *)
26 MODE_STOPPING :BIT := FALSE; (* AH, under torque stopping OpMode(Ack).12 *)
27 MODE_COORDINATED :BIT := FALSE; (* AF, AddAxisToGroup OpMode(Ack).13*)
28 MODE_INTERRUPTED :BIT := FALSE; (* only status: interrupted by a drive reaction e.g. safety OpMode(Ack).14 *)
29 MODE_EXTERNAL_FB :BIT := FALSE; (* ??, Switch to AH and wait for external move commands. [2 lines]
30 (* TRUE: drive is activated | TRUE: Achse aktiviert [1 lines]
31 Active: BOOL:=FALSE;
32
33 (* comment, name of axis | Kommentar, Name der Achse [1 lines]
34 Name: STRING(40);
35
36 (* Last motion command is finished | Letztes Kommando ist abgeschlossen. [1 lines]
37 CmdDone: BOOL:=FALSE;
38
39 (* Type of the axis | Achstyp [2 lines]
40 { attribute 'hide' } AxisType: MB_AXIS_CONFIG:=NoAxisConfig; // replaced by AxisFeatures from 13VRS on
41
42 (* Status of command SetAbsRef (P-0-0012) | Status des Kommandos SetAbsRef (P-0-0012)
43 - siehe auch MB_SERCOS_DATA_STATUS | - see also MB_SERCOS_DATA_STATUS *)
44 AbsRefCmdStatus: MB_SERCOS_DATA_STATUS:=MB_SERCOS_DATA_NOT_SET;
45
46 (* Internal flags for the handling of different drives, e.g use P-0-0012 or S-0-0447 as Cmd AbsRef *)
47 InternalFlags: DWORD:=0;
48
49 (* parameter level 1, parameter A-0-0024 "Achszustand" bit 3 active *)
50 (* TRUE: parameter level 1 active *)
51 (* FALSE: parameter level 1 not active *)
52 ParamLevelAck: BOOL:=FALSE;
53
54 (* Features of this axis (A-0-0004) | Funktionalität dieser Achse (A-0-0004) *)
55 AxisFeatures: MB_AXIS_FEATURES := (wData:=0);
56
57 END_STRUCT
```

図 6-29: MB_AXIS_ADMIN_STATUS

パラメータ化レベル

出力 "ParamLevelAck" は、入力 "ParamLevel" により軸がパラメータ化モードに設定されたか否かを表示します。
この入力をリセットすることにより、軸はデフォルトの状態にリセットされます。



パラメータ化モードが有効な場合、プロジェクト ツリーでは各軸に "PM" が表示され、AxisDatastructure でビット "OperationMode" が消えます。

AxisFeatures

要素 "AxisFeatures" はバイナリ コードで軸プロパティを表示します。
"AxisType" は置き換えられ、今後使用されません。

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_ADMIN_STATUS の要素

MB_AXIS_MODE

概略 この MB_AXIS_MODE は、[MB_AXIS_ADMIN_STATUS](#) に含まれている [_OpModeAck](#) 要素により使用される列挙です。

下の図は、現在の MB_Axis_Mode 列挙タイプを示したものです

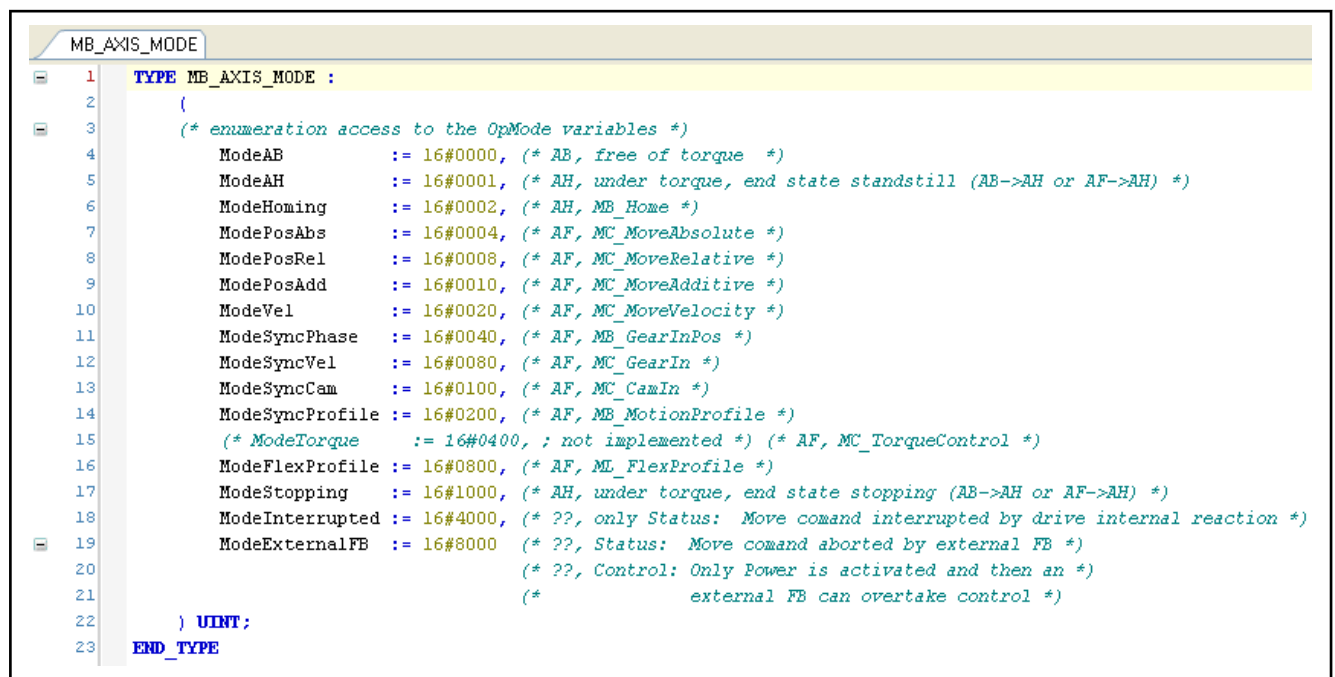


図 6-30: MB_Axis_Mode

MB_AXIS_FEATURES

概略 MB_AXIS_FEATURES は [MB_AXIS_ADMIN_STATUS](#) 構造により使用される構造です

ML_TechInterface.library

```

TYPE MB_AXIS_FEATURES :
STRUCT
{ attribute 'hide' } wData: WORD; // used only to set the info
{attribute 'relative_offset':='0'}
// bits from parameter A-0-0004 Axis type information is partly redundant to Status.Admin.AxisType
IPO_CONTROL:      BIT; // .0 FALSE: IPO Drive
PACK_PROFILE:     BIT; // .1 telegram konfiguration FALSE: device specific
HYDRAULIC:        BIT; // .2 FALSE: Electric drive
REAL_AXIS:        BIT; // .3 IndraDrive, HNC-axis, sercos drive
VIRTUAL_AXIS:     BIT; // .4 virtual axis
ENCODER_AXIS:     BIT; // .5 IndraDrive, HNC-axis, sercos drive
LINK_AXIS:        BIT; // .6 link axis
CONTROLLER_AXIS:  BIT; // .7 controller axis
{ attribute 'hide' } resbit8:      BIT;
{ attribute 'hide' } resbit9:      BIT;
{ attribute 'hide' } resbit10:     BIT;
OPMODE_VEL:       BIT; // .11 Operation mode velocity control supported
OPMODE_POS:       BIT; // .12 Operation mode positioning control supported
OPMODE_TORQUE:    BIT; // .13 Operation mode torque/force control supported
OPMODE_SYNC:      BIT; // .14 Operation mode synchronisation supported
PARAM_LEVEL:      BIT; // .15 param level supported (13VRS)
END_STRUCT
END_TYPE

```

図 6-31: MB_AXIS_FEATURES 構造

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_DIAGNOSIS 構造

概略 MB_AXIS_DIAGNOSIS 構造は (MB_AXIS_STATUS_TYPE01 内) の下位構造で、警告およびエラー出力が含まれています。下の図は、現在の MB_AXIS_DIAGNOSIS 構造を示したものです。

```

TYPE MB_AXIS_DIAGNOSIS :

STRUCT
  {***** Diagnosis outputs - from motion control to logic control *****}
  {***** Diagnose-Ausgaenge - Motion Control zur Logic Control *****}

  Warning:      BOOL := FALSE; (* Warning, copy of | Warnung, Kopie von
    AxisData[].Axis_Warning *)

  Error:        BOOL := FALSE;
  (* Error -> AxisData[].Axis_Error OR Error from an FB in the Axis-Interface
    Fehler -> AxisData[].Axis_Error ODER Fehler von einem FB im Axis-Interface *)

  ErrorID:      ERROR_CODE; (* Error type (internal used FB) | Fehlertyp (intern genutzter FB)
    - only valid when Error = TRUE | - gueltig wenn Error = TRUE *)

  ErrorIdent:   ERROR_STRUCT; (* Error id. (internal used FB) | Fehler-Id. (intern genutzter FB)
    - only valid when Error = TRUE | - gueltig wenn Error = TRUE *)

  Number:      DWORD; (* diagnosis number of the axis | Diagnosenummer der Achse
    - only valid in operating mode | nur gueltig im Betriebsmodus
    - refresh rate controlled with | Auffrischrate gesteuert mit
    arAxisCtrl[].Admin.DiagNbrRefreshTime *)

  Message:     STRING(60); (* diagnosis message of the axis | Diagnosemeldung der Achse
    - only valid in operating mode | nur gueltig im Betriebsmodus
    - refreshed when "Number" changes | Aufgefrischt bei Aenderung von "Number" *)

END_STRUCT

END_TYPE

```

図 6-32: MB_AXIS_DIAGNOSIS

診断表示

arAxisCtrl_gb[iColumn].Admin.DiagNbrRefreshTime は要素 "Number" の更新頻度を制御するために使用されます。要素 "Number" が変更されると直ちにテキスト "Message" が再度読み取られます。arAxisCtrl_gb[].Admin.DiagNbrRefreshTime を使用して診断情報の読み取りをオフにし、診断情報のデータソースを選択することも可能です。

- DiagNbrRefreshTime = T#0ms : 診断番号と診断メッセージ作成の作成がスイッチオフになりました。
- DiagNbrRefreshTime between T#1ms および T#199ms (デフォルトは T#100ms) : コントロール パラメータ A-0-0023/A-0-0020 が常に読み取られ、"Number" またはテキスト "Message" として出力されます。
- DiagNbrRefreshTime が T#200ms 以上 : AxisInterface は、軸のタイプとメッセージのタイプによって、対応する各ドライブまたはコントロールのパラメータを読み込みます。

ドライブに補間がある IndraDrive :

- 肯定応答 : ドライブ パラメータが読み取られます。例 : A0012 Ready for operation (運転準備完了)
- ドライブのエラー : ドライブ パラメータが読み取られます。例 : F2026 No power (電源なし)
- コントロールでのエラー : コントロール パラメータが読み取られます。例 : F2229233 Invalid master axis (無効なマスター軸)

コントロールに補間がある IndraDrive :

- 肯定応答 A-0-0020/A-0-0023 が読み取られる。例 : "Synchronized Motion" (同期動作)

ML_TechInterface.library

- ドライブのエラー：ドライブ パラメータが読み取られます。例：F2026 No power (電源なし)
 - コントロールでのエラー：コントロール パラメータが読み取られる。例：F2229233 Invalid master axis (無効なマスター軸)
- SercosDrive と仮想軸：
- コントロール パラメータが読み取られる。

ライブラリ	IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib
依存関係	IndraMotion MLC 用 ML_ImcInterface.lib/IndraMotion MLD 用 MX_ImcInterface.lib
システム	IndraMotion MLC/IndraMotion MLD

MB_AXIS_POS_STATUS 構造

- 概略 MB_AXIS_POS_STATUS 構造は (MB_AXIS_STATUS_TYPE01 内) の下位構造で、出力ステータスが含まれています。
- 下の図は、現在の MB_AXIS_POS_STATUS 構造を示したものです。

```
TYPE MB_AXIS_POS_STATUS :  
  
STRUCT  
  {***** Outputs of the following axis - from motion control to logic control *****}  
  {***** Achs-Ausgaenge - Motion Control zur Logic Control *****}  
  
  Adjust: MB_AXIS_ADJUST_STATUS; (* -> outputs of the adjusting functionality for the position  
    -> Ausgaenge fuer Tipfbetrieb der Position *)  
  
END_STRUCT  
  
END_TYPE
```

図 6-33: MB_AXIS_POS_STATUS

ライブラリ	IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib
依存関係	IndraMotion MLD 用 MX_ImcInterface.lib
システム	IndraMotion MLC/IndraMotion MLD

MB_AXIS_ADJUST_STATUS

- 概略 MB_AXIS_ADJUST_STATUS 構造は MB_AXIS_POS_STATUS の下位構造で、出力ステータスが含まれています。
- 下の図は、現在の MB_AXIS_POS_STATUS 構造を示したものです。

```

TYPE MB_AXIS_ADJUST_STATUS :
STRUCT
  {***** Outputs of the following axis - motion control to logic control *****}
  {***** Achs-Ausgaenge - Motion Control zur Logic Control *****}

  ModeAck:   BOOL := FALSE; (* Ackn. enable adjust mode | Quittierung Freigabe Tippbetrieb *)

  HighLimitAck: BOOL := FALSE; (* max. value reached | max. Wert erreicht *)

  LowLimitAck:  BOOL := FALSE; (* min. value reached | min. Wert erreicht *)

END_STRUCT
END_TYPE

```

図 6-34: MB_AXIS_ADJUST_STATUS



ライブラリ ML_TechBase からのファンクションブロック MB_ContinuousAdjustType01 および MB_IncrementalAdjustType01 は"調整"を行うために内部的に使用されます。

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImclInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_VEL_STATUS 構造

概略 MB_AXIS_VEL_STATUS 構造は (MB_AXIS_STATUS_TYPE01 内) の下位構造で、出力ステータスが含まれています。

下の図は、現在の MB_AXIS_VEL_STATUS 構造を示したものです。

```

TYPE MB_AXIS_VEL_STATUS :
STRUCT
  {***** Outputs of the following axis - from motion control to logic control *****}
  {***** Achs-Ausgaenge - Motion Control zur Logic Control *****}

  Adjust: MB_AXIS_ADJUST_STATUS; (* -> outputs of the adjusting functionality for the velocity
    -> Ausgaenge fuer Tippbetrieb der Geschwindigkeit *)

END_STRUCT
END_TYPE

```

図 6-35: MB_AXIS_VEL_STATUS

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImclInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_SYNC_STATUS 構造

概略 MB_AXIS_SYNC_STATUS 構造は (MB_AXIS_STATUS_TYPE01 内) の下位構造で、出力ステータスが含まれています。

下の図は、現在の MB_AXIS_SYNC_STATUS 構造を示したものです。

ML_TechInterface.library

```

TYPE MB_AXIS_SYNC_STATUS :
STRUCT
  (***** Outputs of the following axis - from motion control to logic control *****)
  (***** Achs-Ausgaenge - Motion Control zur Logic Control *****)

  CamActiveID:      MC_CAM_ID; (* - ID of the active CamTable | - Nr. aktive Stuetzpunkt-Tab.*)

  ProfileActiveSet: UINT:= 0; (* - Number of the active motion profile set 0 or 1
    - Nummer des aktiven Satzes 0 oder 1 des Bewegungsprofils *)

  PhasingSlaveDone: BOOL := FALSE; (* - Change of PhaseOffset is finished
    - Verstellung des PhaseOffset ist abgeschlossen *)

  (***** status for adjusting of synchronisation mode values *****)
  (***** Statusmeldungen, Tippen im Synchronbetrieb *****)

  AdjFineAdjust:      MB_AXIS_ADJUST_STATUS; (* adjusting | vertippen
    arAxisCntrl[].SyncMode.FineAdjust *)

  AdjCamShaftDistance: MB_AXIS_ADJUST_STATUS; (* adjusting | vertippen
    arAxisCntrl[].SyncMode.CamShaftDistance *)

  AdjPhaseOffset:      MB_AXIS_ADJUST_STATUS; (* adjusting | vertippen
    arAxisCntrl[].SyncMode.PhaseOffset *)

END_STRUCT
END_TYPE

```

図 6-36: MB_AXIS_SYNC_STATUS

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用
MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

6.8.3 MB_AXIS_CONTROL_TYPE01 構造

MB_AXIS_CONTROL_TYPE01

概略 MB_AXIS_CONTROL_TYPE01 データ構造は `arAxisCtrl_gb[]` インスタンス
を使用してアクセスされます。これには以下の要素が含まれています。

```

TYPE MB_AXIS_CONTROL_TYPE01 :
STRUCT
  Admin:      MB_AXIS_ADMINISTRATION; (* Administration of the axis | Verwaltung der Achse *)
  PosMode:    MB_AXIS_POSITIONING; (* operating mode positioning (absolute, rel. and additive)
    Positionierbetrieb (absolut, relativ und additiv) *)
  VelMode:    MB_AXIS_VELOCITY; (* operating mode velocity control | Geschwindigkeitsregelung *)
  SyncMode:   MB_AXIS_SYNCHRONISATION; (* operating mode synchronisation (phase, vel, and cam)
    Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvenscheibe) *)
  (* add your extensions below here, keep the elements above
    Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
  (* user extensions | Anwender Erweiterungen *)

END_STRUCT
END_TYPE

```

図 6-37: MB_AXIS_CONTROL_TYPE01

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_CONTROL_TYPE01 の要素

MB_AXIS_ADMINISTRATION 構造

概略 MB_AXIS_ADMINISTRATION 構造は [MB_AXIS_CONTROL_TYPE01](#) の下位構造で、制御変数とコマンド値が含まれています。

下の図は、現在の MB_AXIS_ADMINISTRATION 構造を示したものです。

ML_TechInterface.library

```

MB_AXIS_ADMINISTRATION
1  TYPE MB_AXIS_ADMINISTRATION :
2  STRUCT
3  (* selection of the operating mode | Betriebsarten-Auswahl
4  - level controlled
5  - set the corresponding variables for your operating mode before activating
6  - set ..._OpMode.en.en := ModeAB, ModeAH, ModePosAbs .... to activate an operating mode
7  - OR use bit variables ..._OpMode.en.b.MODE_AB, ..._OpMode.en.b.MODE_AH, _OpMode.en.b.MODE_POS_ABS.
8  Note: only one bit is allowed at any time!
9
10 - Pegel-sensitiv
11 - vor Auswahl der Betriebsart die Eingangsdaten fuer die Betriebsart beschreiben
12 - Betriebsarten-Auswahl: ..._OpMode.en.en := ModeAB, ModeAH, ModePosAbs .... (ENUMs)
13 - oder ..._OpMode.en.b.MODE_AB, ..._OpMode.en.b.MODE_AH, _OpMode.en.b.MODE_POS_ABS. (BIT-Variablen)
14 Hinweis: immer nur 1 Eingang setzen! *)
15 _OpMode: MD_AXIS_MODE_UNION := ( en := ModeAB);
16
17 (* the reference to this axis | Achse adressieren/auswählen *)
18 Axis:  AXIS_REF := (AxisNo := NO_OBJECT, CntrlNo := LOCAL_CNTRL);
19
20 (* Refresh time for arAxisStatus[].Diag.Number | Auffrischrate arAxisStatus[].Diag.Number
21 - switched off, if zero | - ausgeschaltet wenn Null
22 - read drive para when >= TW200ms | - liest Antriebsparameter wenn >= TW200ms *)
23 DiagNrRefreshTime: TIME:=T#100MS;
24
25 (* Deceleration when DRIVE_HALT is activated - always positive,
26 (Unit: e.g. in [rad/s2] depending on parametrization)
27 Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
28
29 Freigabe-Verzoegerung bei Aktiv. ANTRIEB_HALT- positiver Wert,
30 (Einheit: z.B. in [rad/s2] je nach Parametrierung)
31 Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)
32 StopDeceleration: REAL:=99.0;
33
34 (* WaitTime restart of the MC_Power in [aSec] | Wartezeit Neustart des MC_Power in [aSec]
35 - necessary for some drives e.g. Eco-Cs | - notwendig für einige Antriebe z.B. Eco-Cs *)
36 WaitTimePowerRestart: UDINT:=0;
37
38 (* clears axis errors with MC_Reset() | löscht Achsfehler mit MC_Reset()
39 - edge controlled | flanken-sensitiv *)
40 ClearError: BOOL:=FALSE;
41
42 (* Set absolute reference by command S-0-0447 (P-0-0012) | Absolutmass setzen mit Kommando S-0-0447 (P-0-0012)
43 - edge controlled | flanken-sensitiv *)
44 SetAbsRef: BOOL:=FALSE;
45
46 (* Enable the cyclic check and transmission of changed inputs for the actual operating mode
47 - relevant for the values in AxisCtrl with "Note: cyclic checked AND transmitted on change"
48 - level controlled
49
50 Zuschalten des bedarfsweisen Scannens von Eingangsdaten in der aktuellen Betriebsart
51 - betrifft Groessen in arAxisCtrl mit Ann. "Note: cyclic checked AND transmitted on change"
52 - Pegel-sensitiv *)
53 EnableCyclicScanning: BOOL:=TRUE;
54
55 (* Enable the update on inputs, that are not cyclic checked, without leaving the active opmode
56 - relevant for values in AxisCtrl without "Note: cyclic checked AND transmitted on change"
57 - level controlled
58
59 Zuschalten der Aktualisierung, von nicht zyklisch gescannten Eingangsdaten, ohne die
60 aktuelle Betriebsart zu verlassen
61 - betrifft Groessen in arAxisCtrl ohne "Note: cyclic checked AND transmitted on change"
62 - Pegel-sensitiv *)
63 UpdateEveryInput: BOOL:=FALSE;
64
65 (* TRUE: When the drive is in ModeInterrupted (e.g. drive internal command) the axis will be stopped
66
67 TRUE: Im Falle von ModeInterrupted (z.B. internes Antriebskommando) wird die Achse angehalten *)
68 DriveInterruptReaction: BOOL:=TRUE;
69
70 (* TRUE: MC_Power is internally activated when an _OpMode is selected
71 FALSE: the user application has to handle the MC_Power before an _OpMode is selected
72
73 TRUE: der MC_Power wird intern aktiviert wenn _OpMode ausgewählt wird
74 FALSE: die Anwenders Applikation muß den MC_Power freigeben, bevor eine Betriebsart angewählt wird *)
75 PowerOn: BOOL:=TRUE;
76
77 (* Switch to parameter level 1, parameter A-0-0024 "Achszustand" bit 3 *)
78 (* TRUE: activate parameter level 1 *)
79 (* FALSE: parameter level not activated *)
80 ParamLevel: BOOL:=FALSE;
81
82 END_STRUCT
83 END_TYPE
84

```

図 6-38: MB_AXIS_ADMINISTRATION

運転モードの選択

運転モードは以下のいずれかのメソッドを使用して選択できます。

- Enum 値の選択

タイプ MB_AXIS_MODE の値を

arAxisCtrl_gb[].Admin._OpMode に割り当てる

例 : arAxisCtrl_gb[].Admin._OpMode.en := ModePosAbs; または
arAxisCtrl_gb[].Admin._OpMode.en := ModeAB;

- ビット アクセスを介した選択

まず、arAxisCtrl_gb[].Admin._OpMode := ModeAB; を使用して
"_OpMode" を削除し、次にビット アクセス機能を使用して 1 つのビット
のみを設定します

例 : `arAxisCtrl_gb[ ].Admin._OpMode.b.MODE_POS_ABS := TRUE;`



`arAxisCtrl_gb[ ].Admin._OpMode` に複数のビットを設定すると、エラーが発生し、ドライブが AB に切り替わることがあります。

診断表示 "DiagNbrRefreshTime" 要素は

`arAxisStatus_gb[ ].Diag.Number` の更新頻度を制御するために使用されます。
`arAxisStatus_gb[ ].Diag.Number` が変更されると直ちに
`arAxisStatus_gb[ ].Diag.Message` が再度読み取られます。

パラメータ化レベル

"ParamLevel" 入力により、軸を個別にパラメータ化モードに切り替えることができます (パラメータ A-0-0024 "Axis status" ビット 3)。

これにより、これらの個別の軸のパラメータを変更できます。これによらない場合は、sercos フェーズ 2 で sercos リング全体をシャットダウンすることによってのみ、パラメータを変更できます。

出力 "ParamLevelAck" は、入力 "ParamLevel" により軸がパラメータ化モードに設定されたか否かを表示します。

この入力をリセットすることにより、軸はデフォルトの状態にリセットされます。



パラメータ化モードが有効な場合、プロジェクト ツリーでは各軸に "PM" が表示され、AxisDatastructure でビット "OperationMode" が消えます。

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImclInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_ADMINISTRATION の要素

MB_Axis_Mode

概略

この MB_AXIS_Mode_UNION は、[MB_AXIS_ADMINISTRATION](#) に含まれる **_OpMode** 要素によって使用される合併データ タイプ (UNION) です。UNION には列挙タイプ MB_AXIS_MODE およびビット構造 MB_AXIS_MODE_BITS が含まれています。_OpMode は、両方のデータタイプを使用して制御できます。両方のデータタイプは同じメモリにあります。下の図は、現在の MB_AXIS_MODE データタイプを示したものです。

ML_TechInterface.library

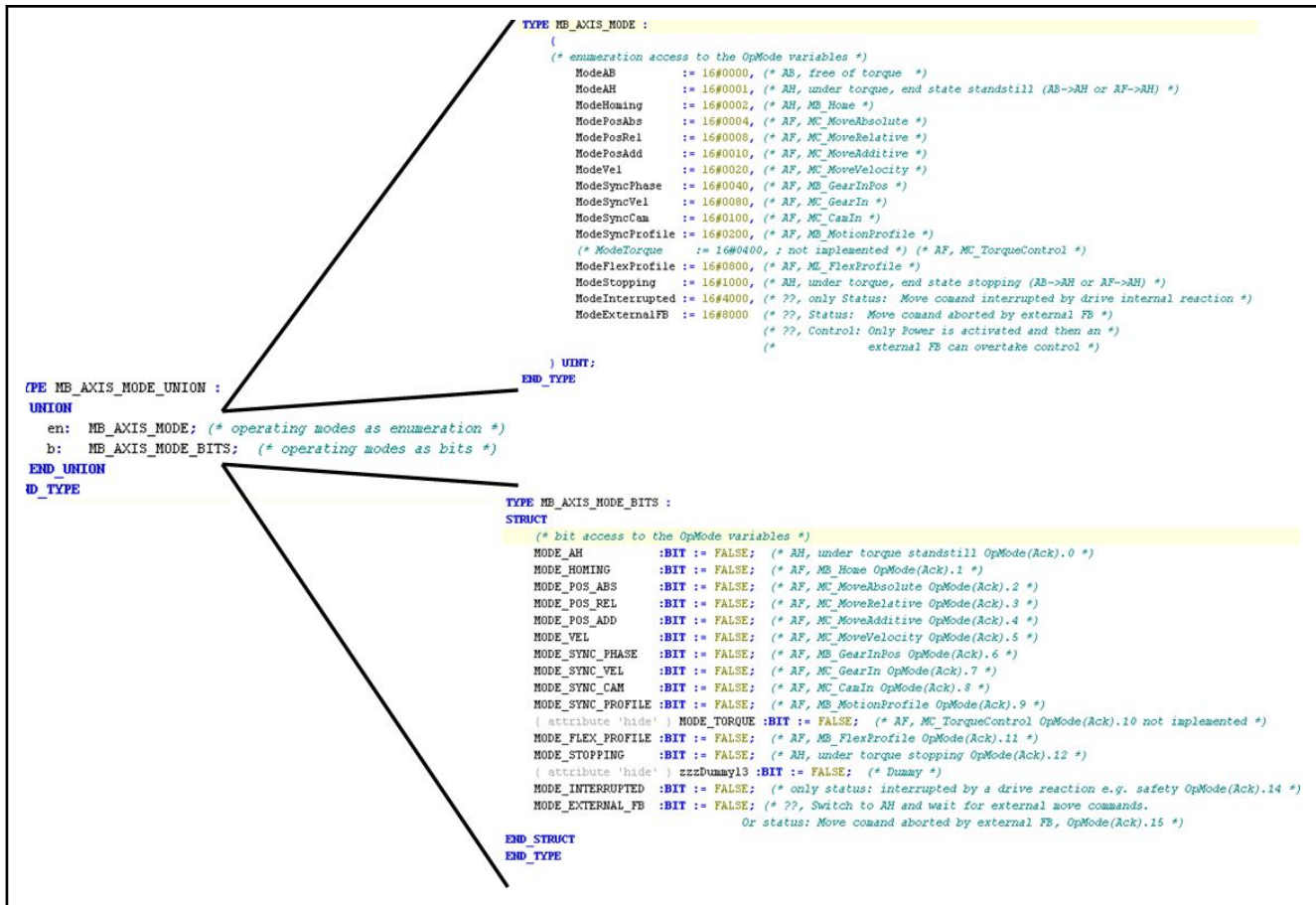


図 6-39: MB_AXIS_MODE データ タイプ

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

Axis_Ref

概略 この Axis_Ref 構造は "RIL_CommonTypes" ライブラリの一部で、各軸に関する情報が含まれています。これらの要素は列挙タイプ CONTROLS および OBJECTS のインスタンスを呼び出します。

下の図は現在の構造を示したものです。

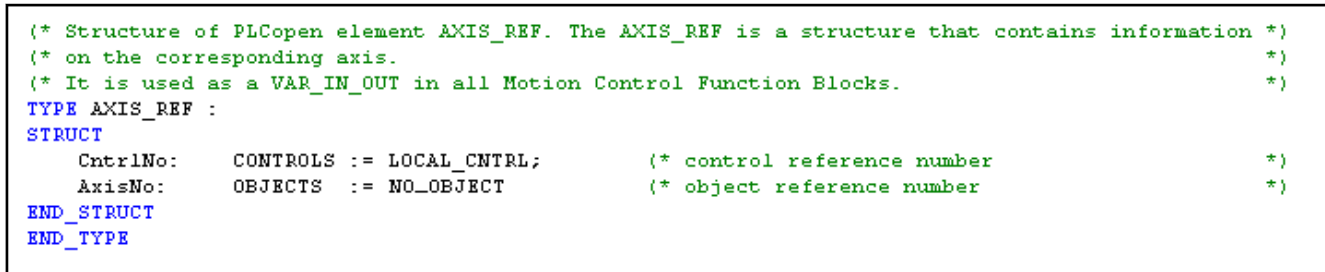


図 6-40: Axis_Ref 構造

MB_AXIS_POSITIONING 構造**MB_AXIS_ADJUST_DATA**

概略 構造 MB_AXIS_ADJUST_DATA は MB_AXIS_POSITIONING の下位構造で、バイナリ設定コマンドとコマンド値が含まれています。下の図は、現在の MB_AXIS_ADJUST_DATA 構造を示したものです。

```

TYPE MB_AXIS_ADJUST_DATA :
STRUCT
  (***** - binary adjust commands *****)
  (***** - Binaere Verstell-Eingaenge *****)

  Enable:          BOOL; (* Enable adjust mode | Freigabe Tippbetrieb
    - level controlled | - Pegel-sensitiv
    - set before activating: | - vor Freigabe folgende Groessen beschreiben:
      ModuloValue, HighLimit, LowLimit, StepWidth, LowStepWidth,
      StepsPerSecond, LowStepsPerSec *)

  Inc:             BOOL; (* Adjust + | Tippen +
    - level controlled when | - Pegel-sensitiv, wenn
      SelIncAdjust = FALSE (ML_ContinuousAdjustType01)
    - edge controlled when | - Flanken-sensitiv, wenn
      SelIncAdjust = TRUE (ML_IncrementalAdjustType01)
    - active, when Enable is TRUE | - aktiv, wenn Enable = TRUE *)

  Dec:             BOOL; (* Adjust - | Tippen -
    - level controlled when | - Pegel-sensitiv, wenn
      SelIncAdjust = FALSE (ML_ContinuousAdjustType01)
    - edge controlled when | - Flanken-sensitiv, wenn
      SelIncAdjust = TRUE (ML_IncrementalAdjustType01)
    - active, when Enable is TRUE | - aktiv, wenn Enable = TRUE *)

  SelIncAdjust:    BOOL; (* Incremental adjusting | Inkrementelle Verstellung
    - SelIncAdjust =
      TRUE: one step width is adjusted on every rising edge of Inc or Dec (edge triggered)
      FALSE: the adjusting is executed as long Inc or Dec are TRUE (level triggered)
    -----
    - SelIncAdjust =
      TRUE: Verstellung um 1 Schrittweite mit jeder steigenden Flanke an Inc oder Dec
      FALSE: kontinuierliche Verstellung, solange Inc oder Dec = TRUE *)

```

図 6-41: MB_AXIS_ADJUST_DATA 構造 (パート 1/2)

ML_TechInterface.library

```

SellLowLevel:    BOOL; (* Reduced speed of changing | Reduzierte Verstellgeschwindigkeit
- SellLowLevel = TRUE means:
  step width = StepWidth * LowStepWidth% / 100% and
  speed of changing = StepsPerSecond * LowStepsPerSec% / 100%
-----
- SellLowLevel = TRUE bedeutet:
  Schrittweite = StepWidth * LowStepWidth% / 100% und
  Verstellgeschwindigkeit = StepsPerSecond * LowStepsPerSec% / 100% *)

(***** - setpoints of adjust | - Verstell-Variablen *****)

AdjModuloValue:  REAL:=0.0; (* ModuloValue for the adjusting | Modulo Wert fuer das Vertippen
- choose zero for absolute scaling | - bei Absolut-Wichtung auf 0 setzen
- read from Parameter by ML_AxisInitType01() *)

HighLimit:       REAL; (* Upper limit of the adj. value | Max. Wert vertippte Zielgroesse *)

LowLimit:        REAL; (* Lower limit of the adj. value | Min. Wert vertippte Zielgroesse *)

StepWidth:       REAL; (* StepWidth | Schrittweite
- s.a.: used to calculate the changing vel. | - s.o.: Berechnung Verstellgeschwindigkeit *)

LowStepWidth:    INT; (* Reduced StepWidth in percent | Reduzierte Schrittweite in %
- see above: used to calculate the changing velocity
  SellLowLevel = TRUE: step width = StepWidth * LowStepWidth% / 100%
-----
- siehe oben: Berechnung der Verstellgeschwindigkeit
  SellLowLevel = TRUE: Schrittweite = StepWidth * LowStepWidth% / 100% *)

StepsPerSecond:  REAL; (* Adjust steps per second | Verstell-Schritte pro Sekunde
- s.a.: used to calculate the changing vel. | - s.o.: Berechnung Verstellgeschwindigkeit *)

LowStepsPerSec:  INT; (* Reduced changing speed in % | Reduzierte Verstellgeschw. in %
- see above: used to calculate the speed of changing
  when SellLowLevel = TRUE: speed of changing = StepsPerSecond * LowStepsPerSec% / 100%
-----
- siehe oben: Berechnung der Verstellgeschwindigkeit
  wenn SellLowLevel = TRUE: Verstellgeschw. = StepsPerSecond * LowStepsPerSec% / 100% *)

END_STRUCT
END_TYPE

```

図 6-42: MB_AXIS_ADJUST_DATA 構造 (パート 2/2)



ライブラリ ML_TechBase からのファンクションブロック MB_ContinuousAdjustType01 および MB_IncrementalAdjustType01 は"調整"を行うために内部的に使用されます。

ライブラリ IndraMotion MLC 用 ML_AxisInterface.lib/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_POSITIONING

概略 MB_AXIS_POSITIONING 構造は MB_AXIS_CONTROL_TYPE01 の下位構造で、位置設定モードのコマンド値が含まれています。下の図は、現在の MB_AXIS_POSITIONING 構造を示したものです。

```

TYPE MB_AXIS_POSITIONING :
STRUCT
  (***** - setpoints of positioning mode *****)
  (***** - Positions-Sollwerte *****)

  Position:      REAL;
  (* position for an absolute move (in technical unit[u]) (negative or positive)
  - start moving with "ModePosAbs" (MC_MoveAbsolute)
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Zielposition, absolut (Einheit: [u]) (negativ oder positiv)
  - Freigabe mit "ModePosAbs" (MC_MoveAbsolute)
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  Distance:      REAL;
  (* distance for a relative or additive move (in technical unit[u]) (negative or positive)
  - start moving with "ModePosRel" or "ModePosAdd" (MC_MoveRelative, MC_MoveAdditive)
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Relativer / additiver Sollwert (Einheit: [u]) (negativ oder positiv)
  - Freigabe mit "ModePosRel" oder "ModePosAdd" (MC_MoveRelative, MC_MoveAdditive)
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  Velocity:      REAL:=10.0;
  (* maximum velocity (always positive) (not necessarily reached) [u/s]
  - start moving with "ModePosAbs" or "ModePosRel" or "ModePosAdd"
    (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive)
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Positioniergeschwindigkeit, (Einheit: [u/s]) (positiv eingeben)
  - Freigabe mit "ModePosAbs" oder "ModePosRel" oder "ModePosAdd"
    (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive)
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  Acceleration:  REAL:=10.0;
  (* Value of the acceleration (always positive) (increasing energy of the motor) [u/s2]
  - start moving with "ModePosAbs" or "ModePosRel" or "ModePosAdd"
    (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive)
  -----
  Beschleunigung (Einheit: [u/s2]) (nur positive Werte)
  - Freigabe mit "ModePosAbs" oder "ModePosRel" oder "ModePosAdd"
    (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive) *)

  Deceleration:  REAL:=10.0;
  (* Value of the deceleration (always positive) (decreasing energy of the motor) [u/s2]
  - start moving with ModePosAbs or ModePosRel or ModePosAdd
    (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive)
  -----
  Verzögerung, (Einheit: [u/s2]) (nur positive Werte)
  - Freigabe mit "ModePosAbs" oder "ModePosRel" oder "ModePosAdd"
    (MC_MoveAbsolute, MC_MoveRelative, MC_MoveAdditive) *)

  Adjust:        MB_AXIS_ADJUST_DATA;
  (* -> adjusting of the element Position
  - only possible, when absolute positioning mode is active
  -----
  -> Manuelle Verstellung (Vertippen) der Position
  - wirksam bei freigegebenem Positionierbetrieb (absolut) *)

END_STRUCT
END_TYPE

```

図 6-43: MB_AXIS_POSITIONING

ML_TechInterface.library

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_VELOCITY 構造

概略 MB_AXIS_VELOCITY 構造は MB_AXIS_CONTROL_TYPE01 の下位構造で、Velocity (速度) モードのコマンド値が含まれています。

下の図は、現在の MB_AXIS_VELOCITY 構造を示したものです。

MB_AXIS_VELOCITY	
1	(***** - setpoints of velocity mode *****)
2	(***** - Sollwerte Geschwindigkeitsregelung (Freilauf) *****)
3	TYPE MB_AXIS_VELOCITY :
4	STRUCT
5	
6	(* Value of the maximum velocity (Unit: e.g. in [rpm] depending on parametrization)
7	- start moving with "ModeVel" (MC_MoveVelocity)
8	Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
9	-----
10	Geschwindigkeits-Sollwert, (Einheit: z.B. in [U/min] je nach Parametrierung)
11	- Freigabe mit "ModeVel" (MC_MoveVelocity)
12	Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)
13	Velocity: REAL:=0.0;
14	
15	(* Value of the acceleration (Unit: e.g. in [rad/s2] depending on
16	parametrization) (always positive)
17	- start moving with "ModeVel" (MC_MoveVelocity)
18	Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
19	-----
20	Beschleunigung, (Einheit: z.B. in [rad/s2] je nach Parametrierung) (immer positiv)
21	- Freigabe mit "ModeVel" (MC_MoveVelocity)
22	Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)
23	Acceleration: REAL:=10.0;
24	
25	(* Value of the deceleration (Unit: e.g. in [rad/s2] depending on
26	parametrization) (always positive)
27	- start moving with "ModeVel" (MC_MoveVelocity)
28	Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
29	-----
30	Verzögerung, (Einheit: z.B. in [rad/s2] je nach Parametrierung) (immer positiv)
31	- Freigabe mit "ModeVel" (MC_MoveVelocity)
32	Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)
33	Deceleration: REAL:=10.0;
34	
35	(* -> adjusting of the element Velocity
36	-> Manuelle Verstellung (Vertippen) der Geschwindigkeit *)
37	Adjust: MB_AXIS_ADJUST_DATA;
38	
39	END_STRUCT
40	END_TYPE

図 6-44: MB_AXIS_VELOCITY

ML_TechInterface.library

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_VELOCITY の要素

MB_AXIS_ADJUST_DATA

概略 構造 MB_AXIS_ADJUST_DATA は [MB_AXIS_POSITIONING](#) の下位構造で、バイナリ設定コマンドとコマンド値が含まれています (115 ページ ["MB_AXIS_ADJUST_DATA" 章](#) も参照)。

MB_AXIS_SYNCHRONISATION 構造

概略 MB_AXIS_SYNCHRONISATION 構造は [MB_AXIS_CONTROL_TYPE01](#) の下位構造で、同期モードのコマンド値と設定が含まれています。下の図は、現在の MB_AXIS_SYNCHRONISATION 構造を示したものです。

ML_TechInterface.library

```

TYPE MB_AXIS_SYNCHRONISATION :
STRUCT
  (***** - setpoint of synchronous motion *****)

  InputRevolution: UINT:=1;
  (* Electronic gear input revolution
  - start moving with "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam" or "ModeSyncProfile"
  - RatioNumerator for MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Elektronisches Getriebe, Eingangsumdrehungen
  - Freigabe mit "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam" oder "ModeSyncProfile"
  - RatioNumerator fuer MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  OutputRevolution: UINT:=1;
  (* Electronic gear output revolution
  - start moving with "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam" or "ModeSyncProfile"
  - RatioDenominator for MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Elektronisches Getriebe, Ausgangsumdrehungen
  - Freigabe mit "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam" oder "ModeSyncProfile"
  - RatioDenominator fuer MC_GearIn, MB_GearInPos, MC_CamIn
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  FineAdjust: REAL:=0.0;
  (* Master fine adjust
  - start moving with "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam" or "ModeSyncProfile"
  - MasterFineadjust for MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Getriebe-Feinabgleich
  - Freigabe mit "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam" oder "ModeSyncProfile"
  - MasterFineadjust fuer MC_GearIn, MB_GearInPos, MC_CamIn
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  CamShaftDistance: REAL:=360.0;
  (* Cam Shaft Distance
  - start moving with "ModeSyncCam"
  - CamShaftDist for MC_CamIn
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Kurvenscheibe, Hub
  - Freigabe mit "ModeSyncCam"
  - CamShaftDist fuer MC_CamIn
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

```

図 6-45: MB_AXIS_SYNCHRONISATION 構造 (パート 1/3)


```

PhaseOffset:    REAL:=0.0; (* Phase offset between master and slave axis
- start moving with "ModeSyncPhase", "ModeSyncCam" or "ModeSyncProfile"
- PhaseShift for MB_PhasingSlave
Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
-----
Winkelversatz (Lagesollwert additiv)
- Freigabe mit "ModeSyncPhase", "ModeSyncCam" oder "ModeSyncProfile"
- PhaseShift fuer MB_PhasingSlave
Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

PhaseOffsetVel: REAL:=1.0;
(* Velocity for the phase offset
- start moving with "ModeSyncPhase", "ModeSyncCam" or "ModeSyncProfile"
- Velocity for MB_PhasingSlave
-----
Aufsynchronisier-Geschwindigkeit
- Freigabe mit "ModeSyncPhase", "ModeSyncCam" oder "ModeSyncProfile"
- Velocity fuer MB_PhasingSlave *)

PhaseOffsetAcc: REAL:=100.0;
(* Acceleration for the phase offset
- start moving with "ModeSyncPhase", "ModeSyncCam" or "ModeSyncProfile"
- PhaseShift for MB_PhasingSlave
-----
Aufsynchronisier-Beschleunigung
- Freigabe mit "ModeSyncPhase", "ModeSyncCam" oder "ModeSyncProfile"
- PhaseShift fuer MB_PhasingSlave *)

SyncDirection: MC_SYNC_DIRECTION:= SHORTEST_WAY;
(* Synchronisation direction
-----
- values 'SHORTEST_WAY'; 'CATCH_UP' (always forward); 'SLOW_DOWN' (see RIL_CommonTypes.lib)
- start moving with "ModeSyncPhase", "ModeSyncCam" or "ModeSyncProfile"
- SyncMode for MB_GearInPos, MC_CamIn, MB_MotionProfile
-----
Aufsynchronisier-Richtung
- Werte 'SHORTEST_WAY'; 'CATCH_UP' (positiv); 'SLOW_DOWN' (siehe RIL_CommonTypes.lib)
- Freigabe mit "ModeSyncPhase", "ModeSyncCam" oder "ModeSyncProfile"
- SyncMode fuer MB_GearInPos, MC_CamIn, MB_MotionProfile *)

StartMode:      MC_START_MODE:= ABSOLUTE;
(* Start mode of the synchronisation
- values 'ABSOLUTE'; 'RELATIVE' (see RIL_CommonTypes.lib)
- start moving with "ModeSyncPhase", "ModeSyncCam" or "ModeSyncProfile"
- StartMode for MB_GearInPos, MC_CamIn, MB_MotionProfile
-----
Aufsynchronisier-Mode
- Werte: 'ABSOLUTE'; 'RELATIVE' (siehe RIL_CommonTypes.lib)
- Freigabe mit "ModeSyncPhase", "ModeSyncCam" oder "ModeSyncProfile"
- StartMode fuer MB_GearInPos, MC_CamIn, MB_MotionProfile *)

```

図 6-46: MB_AXIS_SYNCHRONISATION 構造 (パート 2/3)

ML_TechInterface.library

```

CamTableID:    MC_CAM_ID:= CAM_TABLE_1;
(* Number of the selected cam table
- values 'CAM_TABLE_1'; 'CAM_TABLE_2' ... 'CAM_TABLE_8' (see RIL_CommonTypes.lib)
- start moving with "ModeSyncCam"
- CamTableID for MC_CamIn
Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
-----
Mummer Stuetzpunkt-Tabelle
- Werte: 'CAM_TABLE_1'; 'CAM_TABLE_2' ... 'CAM_TABLE_8' (siehe RIL_CommonTypes.lib)
- Freigabe mit "ModeSyncCam"
- CamTableID fuer MC_CamIn
Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

Master:    AXIS_REF    := (AxisNo := NO_OBJECT, CntrlNo := LOCAL_CNTRL);
(* -> AXIS_REF of the master axis
- "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam", "ModeSyncProfile", "ModeFlexProfile"
- Master for MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile, ML_FlexProfile
Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
-----
-> AXIS_REF der Leitachse
- "ModeSyncPhase", "ModeSyncVel", "ModeSyncCam", "ModeSyncProfile", "ModeFlexProfile"
- Master fuer MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile, ML_FlexProfile
Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

(***** Extensions for profile operation modes *****)
(***** Erweiterungen für die Profil Betriebsarten *****)
Profile:    ML_AXIS_PROFILE;
(* Extensions for the operating modes MotionProfile and FlexProfile
-----
Erweiterungen für die Betriebsarten MotionProfile und FlexProfile *)

(***** adjusting of synchronisation mode values *****)
(***** Vertippen im Synchronbetrieb *****)

AdjFineAdjust:    MB_AXIS_ADJUST_DATA; (* adjusting | vertippen
arAxisCntrl[].SyncMode.FineAdjust *)

AdjCamShaftDistance: MB_AXIS_ADJUST_DATA; (* adjusting | vertippen
arAxisCntrl[].SyncMode.CamShaftDistance *)

AdjPhaseOffset:    MB_AXIS_ADJUST_DATA; (* adjusting | vertippen
arAxisCntrl[].SyncMode.PhaseOffset *)

END_STRUCT
END_TYPE

```

図 6-47: MB_AXIS_SYNCHRONISATION 構造 (パート 3/3)

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

MB_AXIS_SYNCHRONISATION の要素**ML_AXIS_PROFILE**

概略 MB_AXIS_PROFILE 構造は 120 ページ "MB_AXIS_SYNCHRONISATION 構造 (パート 1/3)" 6-45 図 の下位構造で、同期モードのコマンド値と設定が含まれています。下の図は、現在の ML_AXIS_PROFILE 構造を示したものです。

```

TYPE ML_AXIS_PROFILE :
STRUCT
  (* Extensions for the operating modes MotionProfile and FlexProfile
  -----
  Erweiterungen für die Betriebsarten MotionProfile und FlexProfile *)

  SetSelection:  UINT:= 0;
  (* Number of the selected motion profile set 0 or 1
  - SetSelection for MB_MotionProfile, ML_FlexProfile
  Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
  -----
  Auswahl des Satzes 0 oder 1 des Bewegungsprofils
  - SetSelection fuer MB_MotionProfile, ML_FlexProfile
  Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)

  RelativePositioning:  BOOL:= FALSE;
  (* position data processing
  - FALSE: absolute, TRUE: relative
  - RelativePositioning for MB_MotionProfile
  -----
  Verarbeitung der Positionierdaten
  - FALSE: absolut, TRUE: relativ
  - RelativePositioning fuer MB_MotionProfile *)

  (* All following: Extensions for the operating mode FlexProfile
  - all inputs have the same name as the inputs of ML_FlexProfile
  -----
  Erweiterungen für die Betriebsart FlexProfile
  - alle Eingängen haben die gleichen Namen wie die Eingänge von ML_FlexProfile *)

  UseSwitchingPosition:  BOOL:=FALSE;
  (* TRUE: the switching to a different profile is done, when the master axis reaches
  the position of the input "SwitchingPosition".
  -----
  TRUE: Umschaltung auf ein anderes Profil erst wenn die Leitachse die am Eingang
  "SwitchingPosition" angelegte Position erreicht. *)

  SwitchingPosition:  REAL:=0.0;
  (* SwitchingPosition, where the switching to a different profile should be done.
  -----
  Umschaltposition bzw. Anfangsposition, an der die Umschaltung auf ein anderes
  Profil bzw. der Profilstart erfolgen soll. *)

```

図 6-48: 構造 (パート 1/2) ML_AXIS_PROFILE

ML_TechInterface.library

```

(* Describes the interpretation of the profile beginning in case of a profile
   start or change of a profile
- Values: 'SLAVE_ORIGIN_MASTER_ORIGIN'; 'SLAVE_CURRENT_MASTER_CURRENT',
          'SLAVE_ORIGIN_MASTER_CURRENT', 'SLAVE_CURRENT_MASTER_ORIGIN' (see ML_PLCopen.lib)
-----
Beschreibt wie bei einem Profilstart oder Profilwechsel der Ursprung
des Profils interpretiert werden soll
- Werte: 'SLAVE_ORIGIN_MASTER_ORIGIN'; 'SLAVE_CURRENT_MASTER_CURRENT',
         'SLAVE_ORIGIN_MASTER_CURRENT', 'SLAVE_CURRENT_MASTER_ORIGIN' (siehe ML_PLCopen.lib) *)
ProfileEntry:      MB_PROFILE_START_POINT := SLAVE_ORIGIN_MASTER_ORIGIN;

(* Offset of the master axis | Offset der Leitachse *)
MasterOffset: REAL := 0.0;

(* Offset of the following axis | Offset der Folgeachse *)
SlaveOffset: REAL := 0.0;

(* Synchronisation type
- Values: 'SYNC_RAMPIN_SHORTEST_WAY'; 'SYNC_RAMPIN_CATCH_UP',
          'SYNC_RAMPIN_SLOW_DOWN', 'SYNC_DIRECT' (see ML_PLCopen.lib)
-----
Aufsynchronisier-Art
- Werte: 'SYNC_RAMPIN_SHORTEST_WAY'; 'SYNC_RAMPIN_CATCH_UP',
         'SYNC_RAMPIN_SLOW_DOWN', 'SYNC_DIRECT' (siehe ML_PLCopen.lib) *)
SyncType:      MB_SYNC_TYPE:= SYNC_RAMPIN_SHORTEST_WAY;

(* Execution type
- Values: 'EXECUTE_CYCLIC'; 'EXECUTE_SINGLE' (see ML_PLCopen.lib)
-----
Abarbeitungstyp
- Werte: 'EXECUTE_CYCLIC'; 'EXECUTE_SINGLE' (siehe ML_PLCopen.lib) *)
ExecutionMode:  MB_EXECUTION_MODE := EXECUTE_CYCLIC;

END_STRUCT
END_TYPE

```

図 6-49: 構造 (パート 2/2) ML_AXIS_PROFILE

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD 用 MX_AxisInterface.lib

依存関係 IndraMotion MLD 用 MX_ImcInterface.lib

システム IndraMotion MLC/IndraMotion MLD

6.8.4 MB_AXIS_STATUS_TYPE02 構造

MB_AXIS_STATUS_TYPE02

概略 MB_AXIS_STATUS_TYPE02 データ構造は arAxisCtr_gbl[] インスタンスを介してアクセスされ、MB_AXIS_STATUS_TYPE01 データ構造と同じ要素が含まれています。

```

TYPE MB_AXIS_STATUS_TYPE02 EXTENDS MB_AXIS_STATUS_TYPE01:

STRUCT
  (***** Outputs of the following axis - from motion control to logic control *****)

  (* Pointer to AxisData of this axis | Zeiger auf AxisData dieser Achse
  - Type .PtrAxisData^, for the intellisense menu | Eingabe .PtrAxisData^, fuer Auswahlmenue*)
  //PtrAxisData: POINTER TO ML_AXISDATA_SM:=0;

  (* status of the axis | Achs-Status *)
  //Admin: MB_AXIS_ADMIN_STATUS_TYPE02;

  (* axis diagnosis information, error, warning...
  Diagnose Informationen der Achse, Fehler, Warnung... *)
  //Diag: MB_AXIS_DIAGNOSIS;

  (* Status operating mode positioning (absolute, rel. and add.)
  Statusmeldungen Positionierbetrieb (absolut, relativ und additiv) *)
  //PosMode: MB_AXIS_POS_STATUS;

  (* Status operating mode velocity control
  Statusmeldungen Geschwindigkeitsregelung (Freilauf) *)
  //VelMode: MB_AXIS_VEL_STATUS;

  (* Status operating mode synchronisation (phase, vel, and cam)
  Statusmeldungen Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvensch.)*)
  //SyncMode: MB_AXIS_SYNC_STATUS;

  (* add your extensions below here, keep the elements above
  Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)

  (* user extensions | Anwender Erweiterungen *)
END_STRUCT

END_TYPE

```

図 6-50: MB_AXIS_STATUS_TYPE02

6.8.5 MB_AXIS_CONTROL_TYPE02 構造

MB_AXIS_CONTROL_TYPE02

概略 MB_AXIS_CONTROL_TYPE02 データ構造は `arAxisCtr_gbl[ ]` インスタンスを介してアクセスされ、以下の要素が含まれています。

```

TYPE MB_AXIS_CONTROL_TYPE02 EXTENDS MB_AXIS_CONTROL_TYPE01:

STRUCT
  //Admin: MB_AXIS_ADMINISTRATION; (* Administration of the axis | Verwaltung der Achse *)

  //PosMode: MB_AXIS_POSITIONING; (* operating mode positioning (absolute, rel. and additive)
  //Positionierbetrieb (absolut, relativ und additiv) *)

  //VelMode: MB_AXIS_VELOCITY; (* operating mode velocity control | Geschwindigkeitsregelung *)

  //SyncMode: MB_AXIS_SYNCHRONISATION; (* operating mode synchronisation (phase, vel, and cam)
  //Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvenscheibe) *)

  CoordMode: MB_AXIS_COORDINATED; (* Coordinated motion *)

END_STRUCT

END_TYPE

```

図 6-51: MB_AXIS_CONTROL_TYPE02 構造

ML_TechInterface.library



MB_AXIS_CONTROL_TYPE02 データ構造は MB_AXIS_CONTROL_TYPE01 データ構造から派生するため、以下の要素も含まれています。

- Admin
- PosMode
- VelMode
- SyncMode

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library

システム IndraMotion MLC

MB_AXIS_CONTROL_TYPE02 の要素

MB_AXIS_COORDINATED 構造

概略 MB_AXIS_COORDINATED 構造は MB_AXIS_CONTROL_TYPE02 の下位構造で、下位構造 AXIS_GROUP_REF が含まれています。

下の図は、現在の MB_AXIS_COORDINATED 構造を示したものです。

```
(* Command values multi-axis motions | Sollwerte Mehrachsbebewegungen *)
TYPE MB_AXIS_COORDINATED :
STRUCT

  AxisGroupRef:      AXIS_GROUP_REF; (* reference of kinematics | Referenz der Kinematik *)

END_STRUCT
END_TYPE
```

図 6-52: MB_AXIS_COORDINATED

6.9 IMC インターフェースの構造

6.9.1 概要

IndraMotion MLC コントロール用 ML_ImcInterface ライブラリ および IndraMotion MLD-M ドライブ用 MX_ImcInterface は、使いやすいインターフェースをあらゆるシステムに提供します。このインターフェースは、IMC インターフェース、すなわち (I)ndra (M)otion (C)ontrol インターフェースと呼ばれます。IMC インターフェースは、IndraMotion MLC または IndraMotion MLD-D システムのインターフェースとして使用された場合に以下の機能を提供します。

- MLC (IndraMotion MLD-M) 運転モード、P0 (フェーズ 0)、P2 (パラメータモード)、BB (運転モード) を選択する
- "Clear All Errors" (エラーをすべて削除) コマンドを実行する
- "RetriggerOpMode" を実行し、選択した運転モードをユーザーが再度実行できるようにする。sercos リングの接続が解除された場合は IMC インターフェースの運転モードのコントロールをリセットする
- 外部インスタンス (IndraWorks など) を介したフェーズ切替の場合、IMC インターフェースは出力 "PassiveMode" を設定するため、選択されたモードは作動しない。"RetriggerOpMode" を使用するか、または選択されたモードに再度切り替えて、"PassiveMode" を再度終了する



この章で使用されているライブラリの名前は、各システムによって異なります。

IndraMotion MLC コントロール システムはライブラリ ML_TechInterface.compiled-library を使用します。

IndraMotion MLD-M ドライブ システムはライブラリ MX_ImcInterface.lib を使用します。

下の表は、コントロール インターフェースのデータ構造のユーザー インターフェースを挙げたものです。

ユーザー インターフェース	タイプ	説明
ImcCtrl[]	MB_Imc_Control_Type01	運転モード、エラーのクリアおよび"運転モードの起動"のための入力を提供する制御構造
ImcStatus[]	MB_Imc_Status_Type01	運転モードの確認および診断情報を含むステータス構造

表 6-37: コントロール インターフェース - ユーザー インターフェースのデータ構造

IMC インターフェースを使用するためには、プログラム "MB_ImcInterface" のみを周期的に呼び出す必要があります。例: PlcTask 内で GAT または GAT compact を使用する場合、プログラムの呼出はすでに組み込まれています。

6.9.2 MB_IMC_Status_Type01

概略 MB_IMC_Status_Type01 データ構造は、インスタンス ImcStatus[] を使用してアクセスされます。これには以下の要素が含まれています。

```

TYPE MB_IMC_STATUS_TYPE01 :
  STRUCT
    InOperation: BOOL := FALSE;  (* TRUE: ImcInterface is operating *)

    Admin: MB_IMC_ADMIN_STATUS; (* -> Status of the administration of the control *)

    Diag: MB_IMC_DIAGNOSIS;  (* -> Control diagnosis information, error, warning..*)

    PassiveMode: BOOL := FALSE;  (* TRUE: External source controlling operation mode *)

  END_STRUCT
END_TYPE

```

図 6-53: MB_IMC_Status_Type01 構造

ライブラリ IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD-M 用 MX_ImcInterface.lib

依存関係 IndraMotion MLD-M 用 MX_AxisInterface.lib

システム IndraMotion MLC/IndraMotion MLD-M

6.9.3 MB_IMC_Control_Type01

概略 MB_IMC_Control_Type01 データ構造は、インスタンス ImcCtrl[] を使用してアクセスされます。これには以下の要素が含まれています。

ML_TechInterface.library

```

TYPE MB_IMC_CONTROL_TYPE01 :
  STRUCT

    Admin:      MB_IMC_ADMINISTRATION;  (* -> Administration of the control *)

    UserDiag: MB_IMC_USER_DIAGNOSTIC;  (* -> User controlled Diagnostics *)

  END_STRUCT
END_TYPE

```

図 6-54: MB_IMC_Control_Type01 構造

ライブラリ	IndraMotion MLC 用 ML_TechInterface.compiled-library/IndraMotion MLD-M 用 MX_ImcInterface.lib
依存関係	IndraMotion MLD-M 用 MX_AxisInterface.lib
システム	IndraMotion MLC/IndraMotion MLD-M

6.10 操作手順：一般的なユーザー アクティビティ

6.10.1 軸データへのアクセス

以下のデータが利用可能です (*name* = 該当する軸の名前)。

- Axes.<Name>.AxisData => 軸データ構造：ファームウェアのステータスデータおよび軸データ コンテナへのアクセス
- arAxisCtrl_gb[Name.AxisNo] => 軸インターフェースの制御構造
- arAxisStatus_gb[Name.AxisNo] => 軸インターフェースのステータス構造
- DV_Axis[Name.AxisNo] => 直接変数 (機能変数) へのアクセス

AxisData 構造はライブラリ ML_Base のグローバル変数内にあります。

6.10.2 軸の追加

ライブラリからドライブ (IndraDrive など) を選択して sercos ノードにドラッグすることにより、IndraWorks プロジェクトで実在軸を追加します。

必要な場合は、追加した軸のために軸インターフェースの特定の構造要素を初期化する必要があります。例：任意のマスター軸に割当を設定する：

```
arAxisCtrl_gb[Achsname.AxisNo].SyncMode.Master := vAxis1;
```

6.10.3 軸番号が 16 より大きい軸の最大数を大きくする

論理軸番号が「16」より大きい軸がプロジェクトに含まれている場合は、グローバル定数 "uiMAX_AXIS_INDEX" の値を適宜調整します。

"uiMAX_AXIS_INDEX" は、少なくともプロジェクト内の最も大きい論理軸番号の値でなければなりません。デフォルト設定 = '16'

定数 "uiMAX_AXIS_INDEX" を素早く見つけるには、IndraWorks 検索機能 (<Ctrl>+<F>) を使用するか、または以下のフォルダ構造を辿ります。

```
Application/UserConstGlobal
```

6.10.4 軸の削除/名前変更

利用可能な軸を IndraWorks プロジェクト ツリーの sercos ノードの下で削除し、または名前を変更することができます。

軸の名前を変更した場合は、PLC プログラム内の制御構造とステータス構造
 "arAxisCtrl_gb[geänderterAchsname.AxisNo]" および
 "arAxisStatus_gb[geänderterAchsname.AxisNo]" によってアクセスを適合させる必要があります

6.10.5 タスクの作成

一般に、PlcTask はプログラム "PlcProg" を使用してすでに作成されています。

プロジェクト ツリーで、フォルダ Task Configuration (タスク設定) で "Add" (追加) を使用して新しいタスクを作成することができます。

その後、以下を指定する必要があります。

- タスクの優先度 (数字が大きくなるほど優先度が低くなる)
- タスク タイプ (周期的、自由、イベント制御)
- 呼出間隔
- 関連付けられるプログラム組織ユニット "POU"
- 時間間隔によるウォッチドッグの使用

軸は通常、添付プログラム "PlcProg" 内の "PlcTask" で制御されます。このプログラムでは、AxisInterface ブロック "TE_AxisInterface" も呼び出されます。

特別なケースで、よりタイム クリティカルな軸コマンドが必要な場合は、別のタスク (イベント制御の MotionTask など) によりコマンドを実行することができます。この場合、AxisInterface ブロック "TE_AxisInterface" をこのタスクで呼び出す必要があります。

6.10.6 AxisInterface の拡張

AxisInterface では、軸インターフェースの構造のほぼすべての拡張が可能です。また、追加的な下位構造を追加したり、既存の下位構造を拡張したりすることも可能です。

例として、以下の構造要素がユーザー拡張として AxisCtrl 構造にプログラミングされています。

- SetupMode: AxisCtrl 構造の追加的な下位構造
 - Enable: Setup (セットアップ) モードを有効にする
 - JogPlus: ジョグ +
 - JogMinus: ジョグ -
 - Vel: ジョグ/参照移動速度
 - Accel: ジョグ/参照移動減速/加速
 - Homing: 軸のタイプ/設定によって ModeHoming|SetAbsRef|ModePosAbs を有効化
- SyncMode: AxisCtrl 構造内の既存の下位構造の拡張
 - AngleOffset: 表のフェーズ オフセット開始 -> この値によりパラメータ A-0-2730 を説明
 - PhaseShift: ギアの前のマスター軸に比例したオフセット (PhaseOffset: ギアの後マスター軸に比例したオフセット)
 - PhaseShiftVel: PhaseShift を調整するための速度
 - PhaseShiftAcc: PhaseShift を調整するための加速
 - SyncVelocity: フェーズ シフト速度 -> この値によりパラメータ A-0-2790 を説明
 - SyncAcceleration: フェーズ シフト加速 -> この速度はパラメータ A-0-2791 に書き込まれる

ML_TechInterface.library

例として、以下の構造要素がユーザー拡張として *AxisStatus* 構造にプログラミングされています。

- SetupMode: AxisStatus 構造内の追加的な下位構造
 - EnableAck: 設定モードが有効
 - Homingack: ホーミングが完了
- SyncMode: AxisStatus 構造内の既存の下位構造の拡張
 - AngleOffsetDone: 表のフェーズ オフセット開始の転送 -> この値によりパラメータ A-0-2730 を説明
 - PhasingMasterDone: PhaseShift が完了
 - SyncVelocityDone: 同期速度の転送 -> パラメータ A-0-2790 は転送済み
 - SyncAccelerationDone: Transfer of synchronization acceleration -> parameter A-0-2791 was transferred.

これらの拡張のコードは、"AxisInterface" フォルダ内の TE_AxisInterface() ブロックのアクションにあります。関連する構造は、"AxisInterface/Type/Control" および "AxisInterface/Type/Status" フォルダにあります。

ユーザーは、ユーザー独自の拡張を追加できます (これを行うには、[97 ページ "AxisInterface のユーザー拡張" 6.7 章](#) を参照)。

新しいエラー コードが導入された場合、フォルダ "AxisInterface" にある UserConstGlobal 変数リストの構造 "arAXIS_DIAG_MESSAGES" を拡張することにより、PLC 表示に診断メッセージを割り当てることができます。

6.10.7 Robot Control

協調運動のために軸を制御するには、TechInterface に MB_AxisInterfaceType02 FB を使用する必要があります。このため、GAT^{compact} で以下を調整する必要があります。

- 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によってこの構造を拡張する必要があります。

構造 TE_AXIS_CONTROL_Type01 の TE_AXIS_CONTROL_Type02 への名前変更:

```
//former type: TYPE TE_AXIS_CONTROL_TYPE01:
TYPE TE_AXIS_CONTROL_TYPE02: // rename for RobotControl
STRUCT
  Admin:      TE_AXIS_ADMINISTRATION; (* Administration of the axis | Verwaltung der Achse *)
  PosMode:    TE_AXIS_POSITIONING; (* operating mode positioning (absolute, rel. and additive)
    Positionierbetrieb (absolut, relativ und additiv) *)
  VelMode:    TE_AXIS_VELOCITY; (* operating mode velocity control | Geschwindigkeitsregelung *)
  SyncMode:   TE_AXIS_SYNCHRONISATION; (* operating mode synchronisation (phase, vel, and cam)
    Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvenscheibe) *)
  (* add your extensions below here, keep the elements above
    Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
  (* user extensions | Anwender Erweiterungen *)
  // Jog/Homing functionality | Tipp/Einrichtfunktionen
  SetupMode:  TE_AXIS_SETUP_MODE;
  CoordModeCtrl: MB_AXIS_COORDINATED; // inserted for RobotControl
END_STRUCT
END_TYPE
```

- "TE_AXIS_STATUS_Type01" 構造の名前を "TE_AXIS_STATUS_TYPE02" に変更します

構造 *TE_AXIS_STATUS_Type01* の *TE_AXIS_STATUS_TYPE02* への名前変更:

```

    (***** Outputs of the following axis - from motion control to logic control *****)
    // former type: TYPE TE_AXIS_STATUS_TYPE01:
TYPE TE_AXIS_STATUS_TYPE02:      // Rename for RobotControl
STRUCT
    (***** Outputs of the following axis - from motion control to logic control *****)

    PtrAxisData:  POINTER TO ML_AXISDATA_SM;
    (* Pointer to AxisData of this axis | Zeiger auf AxisData dieser Achse
       - Type .PtrAxisData^ for the intellisense menu | Eingabe .PtrAxisData^ fuer Auswahlmenue*)

    Admin:  TE_AXIS_ADMIN_STATUS; (* status of the axis | Achs-Status *)

    Diag:   TE_AXIS_DIAGNOSIS; (* axis diagnosis information, error, warning....
       Diagnose Informationen der Achse, Fehler, Warnung.... *)

    PosMode: TE_AXIS_POS_STATUS; (* Status operating mode positioning (absolute, rel. and add.)
       Statusmeldungen Positionierbetrieb (absolut, relativ und additiv) *)

    VelMode: TE_AXIS_VEL_STATUS; (* Status operating mode velocity control
       Statusmeldungen Geschwindigkeitsregelung (Freilauf) *)

    SyncMode: TE_AXIS_SYNC_STATUS; (* Status operating mode synchronisation (phase, vel, and cam)
       Statusmeldungen Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvensch.)*

    (* add your extensions below here, keep the elements above
       Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
    (* user extensions | Anwender Erweiterungen *)
    (* Jog/Homing functionality | Tipp/Einrichtfunktionen *)
    SetupMode:  TE_AXIS_SETUP_MODE_STATUS;

    // AxisData for HMI (without data type "BIT")
    AxisData_HMI:  TE_AXIS_DATA_STATUS;

END_STRUCT
END TYPE

```

- 新しい FB "MB_AxisInterfaceType02" により "TE_Axisinterface" ブロックを拡張する必要があります。 "TE_AxisInterface EXTENDS MB_AxisInterfaceType02"

TE Axisinterface の拡張:

```

//former: FUNCTION_BLOCK TE_AxisInterface EXTENDS MB_AxisInterfaceType01
FUNCTION_BLOCK TE_AxisInterface EXTENDS MB_AxisInterfaceType02
(
#####
#####

General Header
-----
Shortdescription      : AxisInterface Example

```

- メソッド `mInitExtensions()` : ファンクションブロック `"MB_AxisInterfaceType02"` の追加的な `"CoordModeCtrl"` 入力を初期化する必要があります

CoordModeCtrl の初期化 :

```
SUPERA^.mInitExtension(index);

IF NOT bFirstInit THEN
    // User: Assignment of user defined axis interface inputs
    SetupMode REF= arAxisCtrl_gb[index].SetupMode;
    // inserted for RobotControl
    CoordModeCtrl REF= arAxisCtrl_gb[index].CoordModeCtrl;
    // User: Assignment of user defined axis interface outputs
    SetupModeAck REF= arAxisStatus_gb[index].SetupMode;
    AxisData_HMI REF= arAxisStatus_gb[index].AxisData_HMI;

    // =====
    // Do not touch the following lines
    // =====

    // Set the references for the AxisInterface FB
    AdminCtrlExt REF= arAxisCtrl_gb[index].Admin;
```

ML_TechInterface.library

- "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります

構造変数の調整：

```

VAR_GLOBAL
// Axis interface: command structure: define as RETAIN, to keep the values
// Note: on the L40.2 control, the RETAIN memory is slower then normal VAR, so define
//       this structure only as RETAIN when necessary
// arAxisCtrl_gb: ARRAY [1..MAX_AXIS_NUM] OF MB_AXIS_CONTROL;

// Axis interface: command structure
//former: arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE01;
arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE02;
END_VAR

VAR_GLOBAL
//----- Machine status variables -----

// Axis interface: status structure
//former: arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE01;
arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE02;
bRemoteOn_gb : BOOL := TRUE; // TRUE: Internal IL visu is active
// FALSE: PlcProg is active
END_VAR

```



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 は "AxisInterface\Type" に含まれています。

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "UserVarGlobal" に含まれています。

キネマティクス インターフェース

軸インターフェースに基づいて、オプションのプラグインとしてキネマティクス インターフェースが利用可能です。キネマティクス インターフェースを使用すると、軸と同じ方法でキネマティクスを操作できます。この目的のために、GAT ウィザードを介してキネマティクス インターフェースをプラグインとして追加します。

キネマティクス インターフェースの資料は、プラグイン設定フォルダにあります。

以下の手順で、GAT ウィザードを介して設定フォルダへのリンクを参照します。

- GAT ウィザードでフォルダ "Plug-ins "を選択する
- ウィザード コントロール パネルに、リンク "Open plug.ins configuration folder" (プラグイン設定フォルダを開く) が表示される

7 キネマティクス インターフェース

7.1 はじめに

キネマティクス インターフェースは、Robot Control 機能の組み合わせと拡張を行います。これは軸インターフェースに対応するもので、キネマティクスを容易かつ快適に運用するために設計されています。これにより、Robot Control アプリケーション向けに十分に構造化されたアプリケーション プログラムを短時間で開発することが可能になります。



キネマティクス インターフェースは、Robot Control V1 および Robot Control V2 の両方に使用できます。ただし、特定の運転モードは 1 つの Robot Control バージョンに限って使用が可能です。詳細については、[172 ページ "運転モード" 7.2.4 章](#) を参照してください

システム要件

キネマティクス インターフェースには、以下のハードウェアとソフトウェアが必要です。

- ソフトウェア： IndraMotion MLC 13VRS
- ファームウェア： IndraMotion ML* 13VRS
- CML コントロール
 - IndraControl L25 または
 - IndraControl L45 または
 - IndraControl L65
- テンプレート： 軸インターフェースのサンプル プロジェクトまたは GAT

初期試運転

キネマティクス インターフェースの要件は、協調運動 (Robot Control) に軸インターフェースが組み込まれ、適合していることです。これがすでに行われている場合、以下の手順は省略できます。

軸インターフェースのサンプル プロジェクト、GAT^{compact}、集中状態の機械での GAT または分散状態の機械での GAT を使用してプロジェクトが作成された場合は、以下の手順を実行する必要があります。

軸インターフェースのサンプル プロジェクト

1. 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によって拡張します。
CoordModeCtrl: MB_AXIS_COORDINATED;
2. 構造の名前を "TE_AXIS_STATUS_Type01" から "TE_AXIS_STATUS_TYPE02" に変更します
TE_AXIS_STATUS_TYPE02
3. "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります
arAxisCtrl_gb: ARRAY [] OF TE_AXIS_CONTROL_TYPE02;
arAxisStatus_gb: ARRAY [] OF TE_AXIS_STATUS_TYPE02;
4. ファンクションブロック "TE_Axisinterface" の基本クラスを "MB_AxisInterfaceType02" に変更します

キネマティクス インターフェース

TE_AxisInterface EXTENDS MB_AxisInterfaceType02

5. メソッド "TE_Axisinterface.mInitExtensions()" で、ファンクションブロック "MB_AxisInterfaceType02" の追加的入力 "CoordModeCtrl" を初期化します。

CoordModeCtrl REF= arAxisCtrl_gb[index].CoordModeCtrl;



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 はフォルダ "AxisInterface\Type" に含まれています。

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "UserVarGlobal" に含まれています。

GATcompact:

1. 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によって拡張します。

CoordModeCtrl: MB_AXIS_COORDINATED;

2. 構造の名前を "TE_AXIS_STATUS_Type01" から "TE_AXIS_STATUS_TYPE02" に変更します
TE_AXIS_STATUS_TYPE02

3. "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります

arAxisCtrl_gb: ARRAY [] OF TE_AXIS_CONTROL_TYPE02;

arAxisStatus_gb: ARRAY [] OF TE_AXIS_STATUS_TYPE02;

4. ファンクションブロック "TE_Axisinterface" の基本クラスを "MB_AxisInterfaceType02" に変更します

TE_AxisInterface EXTENDS MB_AxisInterfaceType02

5. メソッド "TE_Axisinterface.mInitExtensions()" で、ファンクションブロック "MB_AxisInterfaceType02" の追加的入力 "CoordModeCtrl" を初期化します。

CoordModeCtrl REF= arAxisCtrl_gb[].CoordModeCtrl;

6. 新しい type02 構造への "TE_GAT_AXIS" 参照を調整します。

CTRL: REFERENCE TO TE_AXIS_CONTROL_TYPE02

Status: REFERENCE TO TE_AXIS_STATUS_TYPE02



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 はフォルダ "GATcompact\AxisInterface\Type".

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "UserVarGlobal" に含まれています。

構造 TE_GAT_AXIS はフォルダ "GATcompact" にあります

キネマティクス インターフェース

集中状態の機械での GAT

1. 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によって拡張します。

CoordModeCtrl: MB_AXIS_COORDINATED;

2. 構造の名前を "TE_AXIS_STATUS_Type01" から "TE_AXIS_STATUS_TYPE"02 に変更します

TE_AXIS_STATUS_TYPE02

3. 構造 "TE_AXISINTERFACE_CONTROL" および "TE_AXISINTERFACE_STATUS" の基本クラスを必要に応じて調整します (Type01 -> Type02)

TE_AXISINTERFACE_CONTROL は TE_AXIS_CONTROL_TYPE02 を拡張
TE_AXISINTERFACE_STATUS は TE_AXIS_STATUS_TYPE02 を拡張

4. ファンクションブロック "TE_Axisinterface" の基本クラスを "MB_AxisInterfaceType02" に変更します

TE_AxisInterface EXTENDS MB_AxisInterfaceType02

5. メソッド "TE_Axisinterface.mInitExtensions()" で、ファンクションブロック "MB_AxisInterfaceType02" の追加的入力 "CoordModeCtrl" を初期化します。

CoordModeCtrl REF= arAxisCtrl_gb[].CoordModeCtrl;



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 はフォルダ "GAT\AxisInterface\Type" に含まれています。

構造 TE_AXISINTERFACE_CONTROL および TE_AXIS_STATUS_TYPE02 はフォルダ "GAT\AxisInterface\Assignments" に含まれています"

分散状態の機械での GAT

1. 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によって拡張します。

CoordModeCtrl: MB_AXIS_COORDINATED;

2. 構造の名前を "TE_AXIS_STATUS_Type01" から "TE_AXIS_STATUS_TYPE"02 に変更します

TE_AXIS_STATUS_TYPE02

3. "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります

arAxisCtrl_gb: ARRAY [] OF TE_AXIS_CONTROL_TYPE02;
arAxisStatus_gb: ARRAY [] OF TE_AXIS_STATUS_TYPE02;

4. ファンクションブロック "TE_Axisinterface" の基本クラスを "MB_AxisInterfaceType02" に変更します

キネマティクス インターフェース

TE_AxisInterface EXTENDS MB_AxisInterfaceType02

5. メソッド "TE_Axisinterface.mInitExtensions()" で、ファンクションブロッ
 ック "MB_AxisInterfaceType02" の追加的入力 "CoordModeCtrl" を初期
 化します。
 CoordModeCtrl REF= arAxisCtrl_gb[].CoordModeCtrl;
6. "arAxisCtrl" および "arAxisStatus" のタイプを type02 構造に調整しま
 す。
 arAxisCtrl: TE_AXIS_CONTROL_TYPE02 の 列
 [1..uiMAX_AXIS_INDEX]
 arAxisStatus: TE_AXIS_STATUS_TYPE02 の 列
 [1..uiMAX_AXIS_INDEX]
7. 新しい type02 構造への "TE_GAT_AXIS" 参照を調整します。
 CTRL: REFERENCE TO TE_AXIS_CONTROL_TYPE02
 Status: REFERENCE TO TE_AXIS_STATUS_TYPE02



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 は "GAT\AxisInterface\Type" フォルダに含まれています。

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "Global_GAT_Axis_Interfaces" にあります。

タイプ arAxisCtrl および arAxisStatus は "GAT\AxisInterface\TE_AxisInitAllAxisType01" にあります

構造 TE_GAT_AXIS はフォルダ "GAT\MachineUnits" にあります

キネマティクス インターフェース
の組み込み

- キネマティクス インターフェースを組み込むには、以下の手順を実行します。
1. 次のボタンを押して GAT ウィザードを起動する : 
2. "Plug-ins" ノードを選択する
3. 機能 "Add Plug-in" (プラグインの追加) を選択する
4. "KinInterface" プラグインを選択し、**Apply** (適用) でインポートを確認する

7.2 ML_KinTech.library

7.2.1 導入と概要

ML_KinTech は以下のファンクションブロックを備えています。

ファンクションブロック	説明
ML_KinematicsInitType01 (162 ページ "ML_KinematicsInitType01" 章 を参照)	あらゆるキネマティクス向けにキネマティクス インターフェースを初期化するために使用します。このファンクションブロックは、プログラムを起動するときに 1 回、および/またはパラメータ化モードから運転モードにフェーズが切り替えられるたびに呼び出す必要があります。
ML_KinematicsInterfaceType01 (165 ページ "ML_KinematicsInterfaceType01" 章 を参照)	あらゆるキネマティクス向けにキネマティクス インターフェースを設定するために使用します。このファンクションブロックは、運転モードが有効である限り、周期的に呼び出す必要があります。

キネマティクス インターフェース

ファンクションブロック	説明
ML_KinJogV1 (167 ページ "ML_KinJogV1" 章 を参照)	Robot Control V1 でのキネマティクスの個別の軸または軸グループをジョグするために使用します。
ML_KinJog (169 ページ "ML_KinJog" 章 を参照)	Robot Control V2 でのキネマティクスの個別の軸または軸グループをジョグするために使用します。

表 7-1: ML_KinTech ファンクションブロック

キネマティクス インターフェースは、IndraMotion プログラミング テンプレートの軸インターフェース用のプラグインとして提供されます。このため、その構造および操作のモードは、軸インターフェースと類似しています。軸インターフェースでの軸は、キネマティクス インターフェースではキネマティクスにより置き換えられます。

下の図は、キネマティクス インターフェースのデータ構造を表したものです。

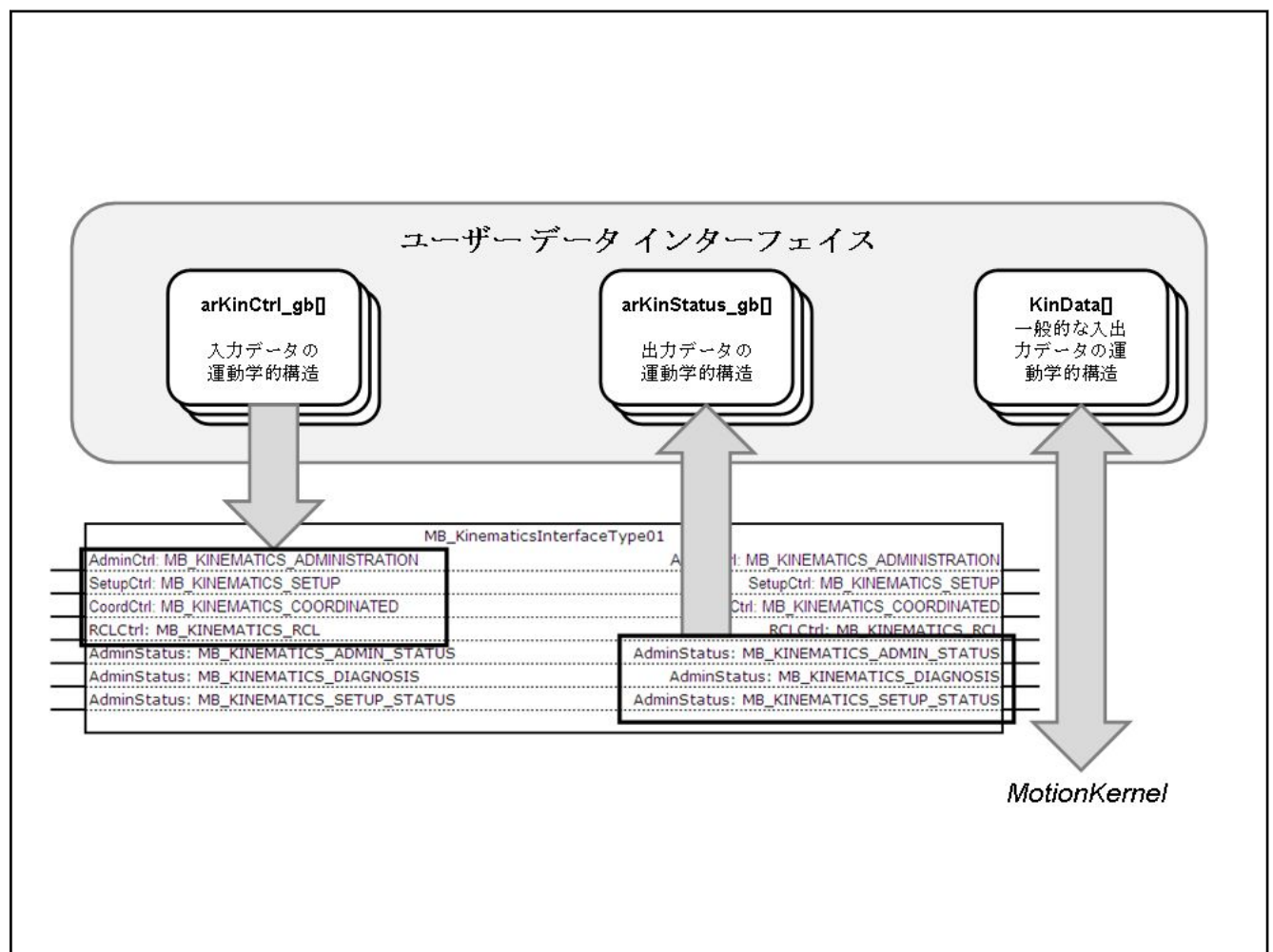


図 7-1: キネマティクス インターフェース - インターフェースのデータ構造

キネマティクス インターフェース

下の表は、軸インターフェースのデータ構造のユーザー インターフェースを示したものです。

ユーザー インターフェース	タイプ	説明
arKinCtrl[]_gb	MB_KINEMATICS_CONTROL_TYPE01 (ユーザー拡張： TE_KINEMATICS_CONTROL_TYPE01 MB_KINEMATICS_CONTROL_TYPE02 (ユーザー拡張： TE_KINEMATICS_CONTROL_TYPE02	運転モードを有効にするためのコマンド値や変数などの制御構造
arKinStatus[]_gb	MB_KINEMATICS_STATUS_TYPE01 (ユーザー拡張： TE_KINEMATICS_CONTROL_TYPE01) MB_KINEMATICS_STATUS_TYPE02 (ユーザー拡張： TE_KINEMATICS_CONTROL_TYPE02)	運転モードの診断情報や確認などのステータス構造
KinData[]		実際の値およびステータス ビットを含むステータスおよび制御構造 (ファームウェアの MotionControl によって処理)

表 7-2: キネマティクス インターフェース - ユーザー インターフェースのデータ構造

7.2.2 キネマティクス インターフェースの構造

概要

キネマティクス インターフェースは、簡単なキネマティクス操作のためのデータ インターフェースを提供します。キネマティクス インターフェースのデータ構造は、以下のセクションで説明されています。



キネマティクス インターフェースは、ユーザーによる拡張が可能です。これらの拡張を有効にするためには、ライブラリ構造を拡張するユーザー プロジェクトでユーザー構造を定義する必要があります。拡張された構造は、プレフィックス "TE_" で識別します。このため、例えば、TE_KINEMATICS_ADMIN_STATUS と呼ばれる構造は、MB_KINEMATICS_ADMIN_STATUS の拡張構造で、構造に関する情報は MB_KINEMATICS_ADMIN_STATUS という名称の項目にあります。

MB_KINEMATICS_CONTROL_TYPE01 構造

MB_KINEMATICS_CONTROL_TYPE01

概略 arKinCtrl_gb[] インスタンスは MB_KINEMATICS_CONTROL_TYPE01 データ構造にアクセスします。

これには以下の要素が含まれています。

```
MB_KINEMATICS_CONTROL_TYPE01
1  TYPE MB_KINEMATICS_CONTROL_TYPE01 :
2  STRUCT
3      Admin: MB_KINEMATICS_ADMINISTRATION;  (* Administration of the kinematics | Verwaltung der Kinematik *)
4
5      SetupMode: MB_KINEMATICS_SETUP;        (* Jog/Homing functionality | Tipp/Einrichtfunktionen *)
6
7      CoordMode: MB_KINEMATICS_COORDINATED;  (* operating mode coordinated motion (linear, circular, direct; absolute, relative) |
8                                              Koordinierbetriebsart (Linear, Zirkular, PTP) *)
9
10     RCLMode: MB_KINEMATICS_RCL;             (* operating mode robot control language (RCL) |
11                                             RCL-Betriebsart*)
12 END_STRUCT
13 END_TYPE
```

図 7-2: MB_KINEMATICS_CONTROL_TYPE01

- Admin キネマティクス管理属性へのアクセス
- SetupMode ジョグ/セットアップ機能の属性へのアクセス
- CoordMode 位置設定モードの属性へのアクセス
- RCLMode RCL 運転モードの属性へのアクセス
- Library ML_KinTech.compiled-library
- System IndraMotion MLC

MB_KINEMATICS_CONTROL_TYPE01 の要素

MB_KINEMATICS_ADMINISTRATION 構造

概略 MB_KINEMATICS_ADMINISTRATION 構造
は、"MB_KINEMATICS_CONTROL_TYPE01" の下位構造で、キネマティクス管理のための制御変数とコマンド値が含まれています。

キネマティクス インターフェース

下の図は、MB_KINEMATICS_ADMINISTRATION の構造を示したものです。

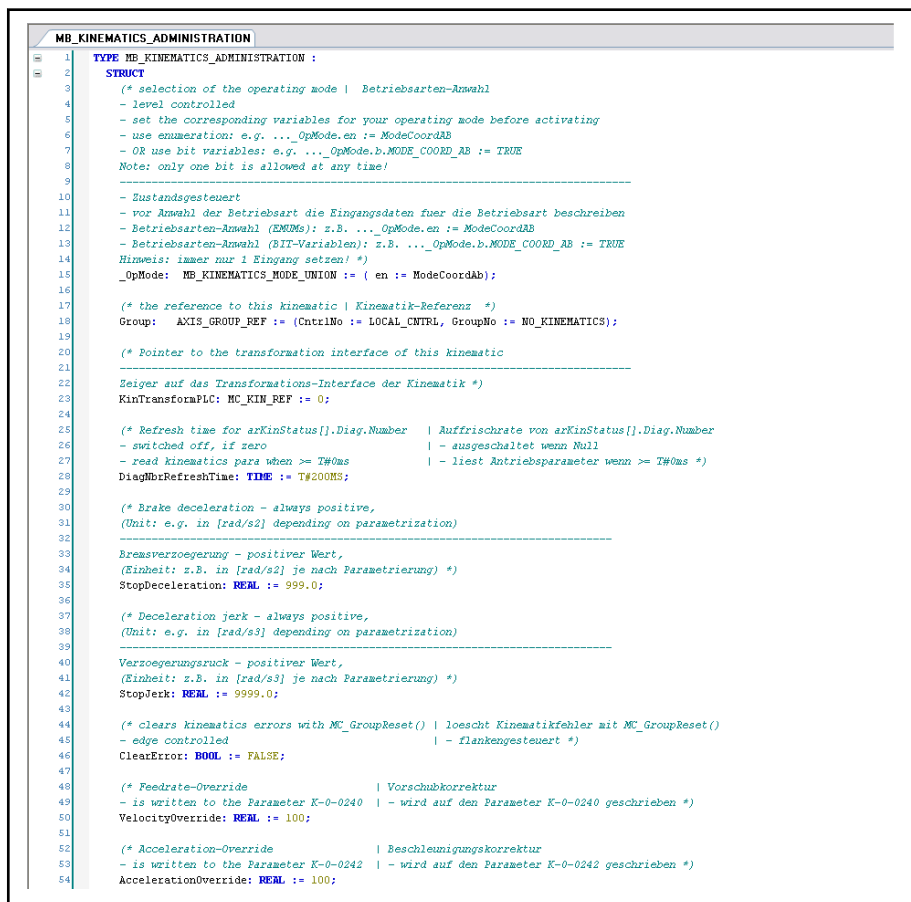


図 7-3: MB_KINEMATICS_ADMINISTRATION (ノート 1)

キネマティクス インターフェース

```

56      (* Deceleration-Override | Verzögerungskorrektur
57      - is written to the Parameter K-0-0243 | - wird auf den Parameter K-0-0243 geschrieben *)
58      DecelerationOverride: REAL := 100;
59
60      (* Enable the cyclic check and transmission of changed inputs for the actual operating mode
61      - relevant for the values in arKinCtrl_gb with note "Note: cyclic checked AND transmitted on change"
62      - level controlled
63
64      Umschalten des bedarfsweisen Scannens von Eingangsdaten in der aktuellen Betriebsart
65      - betrifft Groessen in arKinCtrl_gb mit Anm. "Hinweis: wird permanent zyklisch verarbeitet"
66      - Zustandsgesteuert *)
67      EnableCyclicScanning: BOOL := TRUE;
68
69      (* Switch to parameter level 1, parameter A-0-0024 bit 3
70      - TRUE: activate parameter level 1
71      - FALSE: deactivate parameter level 1
72
73      Umschalten der Achsen in den Parametriermodus 1, Parameter A-0-0024 Bit 3
74      - TRUE: aktiviert den Parametriermodus
75      - FALSE: deaktiviert den Parametriermodus *)
76      ParamLevel: BOOL := FALSE;
77
78      (* Restart the same OpMode again, e.g. ModeCoordPosLinAbs with same values
79      - edge controlled
80
81      Neustart der gleichen Betriebsart, z.B. ModeCoordPosLinAbs mit gleichen Werten
82      - flankengesteuert *)
83      RetrigggerOpMode: BOOL := FALSE;
84
85      (* Configuration of a conveyor belt for Robot-Control V1
86
87      Konfiguration eines Transportbandes für Robot-Control V1 *)
88      ConveyorBeltV1: ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF MB_KINEMATICS_CONVEYOR_BELT_V1;
89
90      (* Configuration of a conveyor belt for Robot-Control V2
91
92      Konfiguration eines Transportbandes für Robot-Control V2 *)
93      ConveyorBelt: ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF MB_KINEMATICS_CONVEYOR_BELT;
94
95      (* Handling of RCL points | Handhabung von RCL-Punkten
96      - set a single point | - Setzen eines einzelnen Punktes
97      - read a single point | - Lesen eines einzelnen Punktes
98      - read a points list | - Lesen einer Punkteliste
99      - read a program list | - Lesen einer Programmliste
100     - ONLY Robot-Control V1 | - NUR mit Robot-Control V1 zulässig *)
101     RCLPoint: MB_KINEMATICS_RCL_POINT;
102
103     (* Handling of RCL programs | Handhabung von RCL-Programmen
104     - load a program | - Laden eines Programms
105     - save a program | - Speichern eines Programms
106     - compile a program | - Kompilieren eines Programms
107     - ONLY Robot-Control V1 | - NUR mit Robot-Control V1 zulässig*)
108     RCLProg: MB_KINEMATICS_RCL_PROG;
109
110     (* Activities after changing operation mode
111     - NoAction: execute the latest block and command the next block
112     - StopMovement: stop the latest block and start the next block
113     - AbortMovement: abort the latest block and blend to the next block
114
115     Verhalten nach einem Betriebsartenwechsel
116     - NoAction: führe den aktuellen Satz aus und kommandiere den nächsten Satz
117     - StopMovement: stoppe den aktuellen Satz und starte den nächsten Satz
118     - AbortMovement: unterbreche den aktuellen Block und verschleife auf den nächsten Satz *)
119     ChangeOpMode: MB_KINEMATICS_ADMIN_CHANGE_OPMODE := NoAction;
120
121     END_STRUCT
122     END_TYPE

```

図 7-4: MB_KINEMATICS_ADMINISTRATION (パート 2)

OpMode 運転モードの選択

運転モードの詳細については、172 ページ "概要" 章 を参照してください



arKinCtrl_gb[].Admin._OpMode.b に複数のビットを設定すると、エラーが発生します。

Group

この要素には、コントロールに関する情報およびキネマティクス番号が含まれており、内部的に使用するファンクションブロックのための参照としての役割を果たします。

KinTransformPLC

この要素により、ユーザー定義の変換が可能になります。変換ファンクションブロックが作成された場合、そのファンクションブロックは PLC に登録されます。

キネマティクス インターフェース

		ユーザー定義の変換の登録は、パラメータ化モード P2 でのみ可能です。運転モード BB で変換の登録を試みても、結果は返されません。
		キネマティクス インターフェースが初回にユーザー定義の変換で起動された場合、コントロールはエラー メッセージ "Transformation not registered" (変換が登録されていません) を返します。これは、IMC インターフェースが運転モードへの切替を試みているためです。変換が登録されると、エラーを確認できません。
DiagNbrRefreshTime		この要素は arKinStatus_gb[].Diag の更新頻度を制御します。
StopDeceleration		この要素の値は、キネマティクスを停止するために適用される減速を示します。
		Robot Control V2 の場合のみ許可されます。
StopJerk		この要素の値は、キネマティクスの停止ジャークに適用される減速を示します。
		Robot Control V2 の場合のみ許可されます。
ClearError		このエッジ制御要素はキネマティクスおよび軸エラーを削除します。立ち上がりエッジはファンクションブロック "MC_GroupReset" の呼出も行います。
VelocityOverride		この要素はフィード補正を示します。値はパラメータ K-0-0240 に内部的に書き込まれます。
AccelerationOverride		この要素は加速補正を示します。値はパラメータ K-0-0242 に内部的に書き込まれます。
DecelerationOverride		この要素は減速補正を示します。値はパラメータ K-0-0243 に内部的に書き込まれます。
EnableCyclicScanning		この周期的要素は arKinCtrl_gb[].CoordMode.Point の周期的読み取りを制御します。この要素が設定されている場合、それぞれの新しい位置により新しい動作が開始され、それによって新しいブロックがブロック バッファに入力されます。
ParamLevel		このエッジ制御要素は、キネマティクスに属するすべての軸をパラメータ化モードに切り替えます。この要素をリセットすることにより、軸はそれぞれのデフォルトの状態にリセットされます。
		パラメータ化モードが有効な場合、プロジェクト ツリーで各軸と共に "PM" が表示されます。
RetriggerOpMode		このエッジ制御要素は、現在使用中の運転モードを再起動します。この要素を設定すると、キネマティクス インターフェースが内部的に "ModeCoordExternalFB" に切り替わり、続いて元の運転モードに戻ります。このため、位置値を変更せずに、相対的な移動を数回行うことが可能です。
ConveyorBeltV1		Robot Control V1 のベルトを設定するための属性へのアクセス 詳細については、 146 ページ "MB_KINEMATICS_CONVEYOR_BELT_V1" 章 を参照してください
		Robot Control V1 の場合のみ許可されます。各キネマティクスにつき最大 4 本のベルトが設定可能です。
ConveyorBelt		Robot Control V2 のベルトを設定するための属性へのアクセス

キネマティクス インターフェース

詳細については、を参照してください [144 ページ](#)
["MB_KINEMATICS_CONVEYOR_BELT" 章](#)



Robot Control V2 の場合のみ許可されます。各キネマティクスにつき最大 16 本のベルトが設定可能です。

RCLPoint

RCL ポイントを処理するための属性へのアクセス

詳細については、を参照してください [147 ページ](#)
["MB_KINEMATICS_RCL_POINT" 章](#)



Robot Control V1 の場合のみ許可されます。

RCLProg

RCL プログラムを処理するための属性へのアクセス

詳細については、を参照してください [149 ページ](#)
["MB_KINEMATICS_RCL_PROG" 章](#)



Robot Control V1 の場合のみ許可されます。

ChangeOpMode

この要素により、運転モード変更後の動作を定義できます。

以下の値が選択可能です。

- NoAction - 運転モードが変更された場合、すでに命令済みの動作が実行されます
- StopMovement - 運転モードが変更された場合、ファンクションブロック "MB_IRDPProgStop" (Robot Control V1) または "MC_GroupStop" (Robot-Control V2) により現在の動作が停止されます

Library

ML_KinTech.compiled-library

System

IndraMotion MLC

MB_KINEMATICS_ADMINISTRATION の要素

MB_KINEMATICS_MODE_UNION

概略

この MB_KINEMATICS_MODE_UNION 構造は、MB_KINEMATICS_ADMINISTRATION 構造に含まれる **_OpMode** 要素によって使用される合併データ タイプ (UNION) です。これには MB_KINEMATICS_MODE 列挙タイプおよび MB_KINEMATICS_MODE_BITS ビット構造が含まれています。**_OpMode** は両方のデータタイプによって制御できます。なぜなら、両方同じ保存場所にあるためです。

キネマティクス インターフェース

下の図は、現在の MB_KINEMATICS_MODE および MB_KINEMATICS_MODE_BITS のデータ タイプを示したものです。

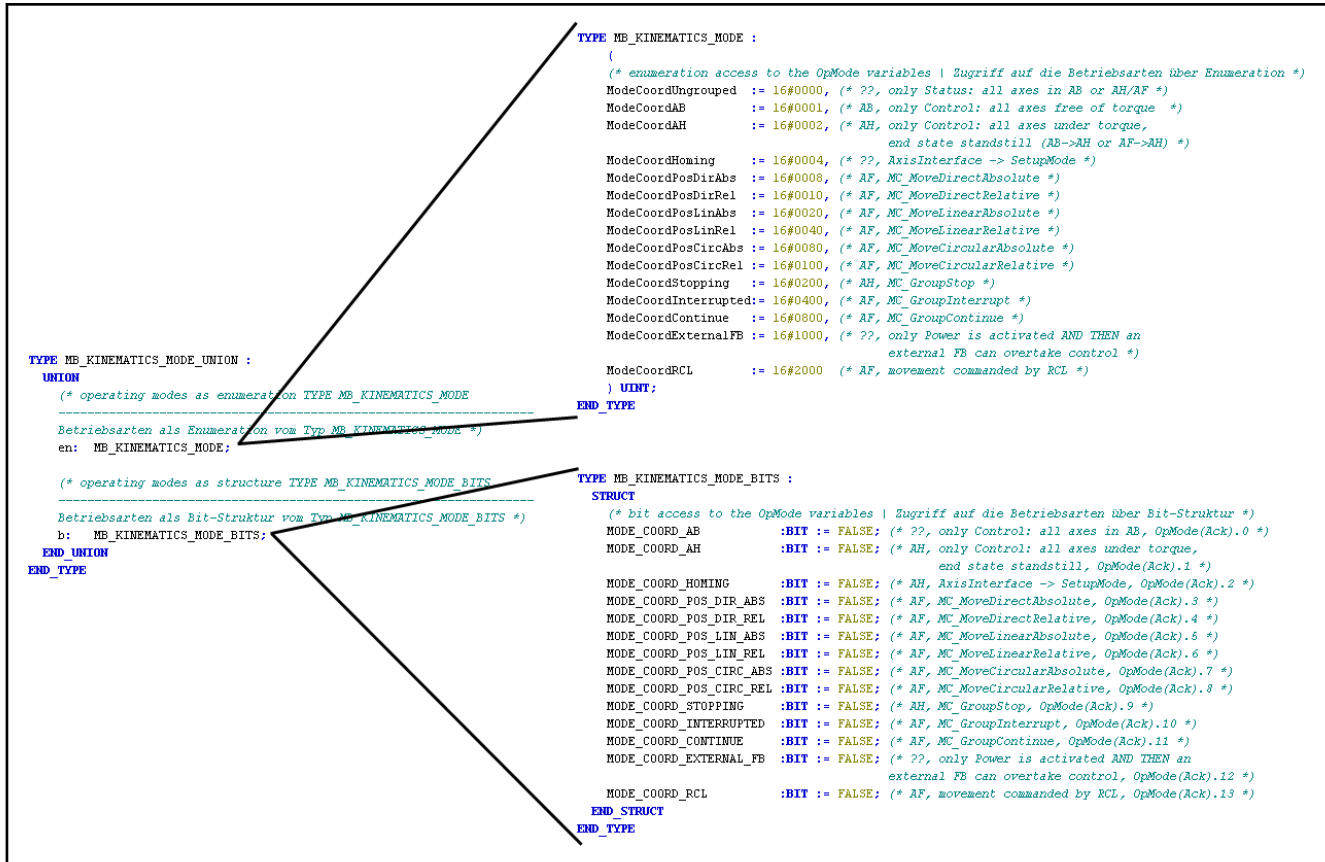


図 7-5: MB_KINEMATICS_MODE_UNION のデータ タイプ

- en 列挙値を介した運転モードへのアクセス
- b ビット構造を介した運転モードへのアクセス

Library ML_KinTech.compiled-library

System IndraMotion MLC

MB_KINEMATICS_CONVEYOR_BELT

概略 構造 MB_KINEMATICS_CONVEYOR_BELT は、Robot Control V2 でのベルトを設定するために使用されます。

この構造に含まれる要素は、ファンクションブロックの入力に対応します

- ML_SetBeltConfiguration
- ML_SetBeltErrorReaction

詳細については、各資料を参照してください。

下の図は、構造要素を示したものです。



Robot Control V2 の場合のみ許可されます。

MB_KINEMATICS_CONVEYOR_BELT	
1	TYPE MB_KINEMATICS_CONVEYOR_BELT :
2	STRUCT
3	<i>(* Execute belt configuration Ausführen der Bandkonfiguration</i>
4	<i>- edge controlled - flankengesteuert *)</i>
5	Execute: BOOL := FALSE ;
6	
7	<i>(* Defines the position and rotation of the belt relative to the WCS</i>
8	<i>-----</i>
9	<i>Definiert die Position und Verdrehung des Bandes relativ zum WCS *)</i>
10	ConveyorBeltOrigin: ARRAY [1..6] OF REAL := [6(0.0)];
11	
12	<i>(* Defines the length of the belt synchronous traverse from the starting</i>
13	<i>position</i>
14	<i>-----</i>
15	<i>Definiert die Länge des bandsynchronen Verfahrbereichs ab der Startposition *)</i>
16	Totallength: REAL := 0.0;
17	
18	<i>(* Defines the length of the latest starting for band synchronous movement</i>
19	<i>from the start position</i>
20	<i>-----</i>
21	<i>Definiert die Länge des spätesten Starten einer bandsynchronen Bewegung ab</i>
22	<i>der Startposition *)</i>
23	BeginLength: REAL := 0.0;
24	
25	<i>(* Referenz of the belt axis</i>
26	<i>-----</i>
27	<i>Referenz der Bandachse *)</i>
28	Axis: AXIS_REF ;
29	
30	<i>(* Defines the type of the error reaction</i>
31	<i>-----</i>
32	<i>Definiert den Typ der Fehlerreaktion *)</i>
33	ErrorReactionType: ML_BELT_ERROR_REACTION_MODE := ML_BELT_ERROR_NO_SPECIFIC_REACTION ;
34	
35	<i>(* Defines the time for the desynchronization</i>
36	<i>-----</i>
37	<i>Definiert die Zeit zum absynchronisieren *)</i>
38	ErrorReactionDesyncTime: REAL := 0.0;
39	
40	<i>(* Defines the deceleration for the desynchronization</i>
41	<i>-----</i>
42	<i>Definiert die Bremsverzögerung zum absynchronisieren *)</i>
43	ErrorReactionDeceleration: REAL := 0.0;
44	
45	<i>(* Defines the jerk for the desynchronization</i>
46	<i>-----</i>
47	<i>Definiert den Bremsruck zum absynchronisieren *)</i>
48	ErrorReactionJerk: REAL := 0.0;
49	END_STRUCT
50	END_TYPE

図 7-6: MB_KINEMATICS_CONVEYOR_BELT

Library ML_KinTech.compiled-library

System IndraMotion MLC

キネマティクス インターフェース

MB_KINEMATICS_CONVEYOR_BELT_V1

概略 構造 MB_KINEMATICS_CONVEYOR_BELT_V1 は Robot Control V1 でのベルトを設定するために使用されます。

この構造に含まれる要素は、ファンクションブロックの入力に対応します

- ML_SetBeltParameter

詳細については、各資料を参照してください。



Robot Control V1 の場合のみ許可されます。

下の図は、構造要素を示したものです。

```

MB_KINEMATICS_CONVEYOR_BELT_V1
1  TYPE MB_KINEMATICS_CONVEYOR_BELT_V1 :
2  STRUCT
3      (* Execute belt configuration      | Ausführen der Bandkonfiguration
4      - edge controlled                 | - flankengesteuert *)
5      Execute: BOOL := FALSE;
6
7      (* Defines the belt synchronization algorithm
8      -----
9      Definiert den Bandsynchronisationsalgorithmus *)
10     BeltKind: ML_BELT_KIND := BK_4;
11
12     (* Defines the starting position for belt synchronization
13     -----
14     Definiert die Startposition ab der bandsynchron Verfahren werden kann *)
15     Base: ARRAY [1..16] OF REAL := [16(0.0)];
16
17     (* Defines the length of the belt synchronous traverse from the starting
18     position
19     -----
20     Definiert die Länge des bandsynchronen Verfahrensbereichs ab der Startposition *)
21     FullLength: REAL := 0.0;
22
23     (* Defines the length of the latest starting for band synchronous movement
24     from the start position
25     -----
26     Definiert die Länge des spätesten Starten einer bandsynchronen Bewegung ab
27     der Startposition *)
28     BeginLength: REAL := 0.0;
29
30     (* Referenz of the belt axis
31     -----
32     Referenz der Bandachse *)
33     Axis: AXIS_REF;
34 END_STRUCT
35 END_TYPE

```

図 7-7: MB_KINEMATICS_CONVEYOR_BELT_V1

Library ML_KinTech.compiled-library

System IndraMotion MLC

キネマティクス インターフェース

MB_KINEMATICS_RCL_POINT

概略 構造 *MB_KINEMATICS_RCL_POINT* は以下の目的のために使用されます。

- 個別のポイント (ML_RoCoSetPoint) の書き込み
- 個別のポイント (ML_RoCoGetPoint) の読み取り
- ポイント リスト (ML_RoCoGetPointList) の読み取り
- プログラム リスト (ML_RoCoDir) の読み取り

詳細については、各資料を参照してください。



Robot Control V1 の場合のみ許可されます。

キネマティクス インターフェース

下の図は、構造要素を示したものです。

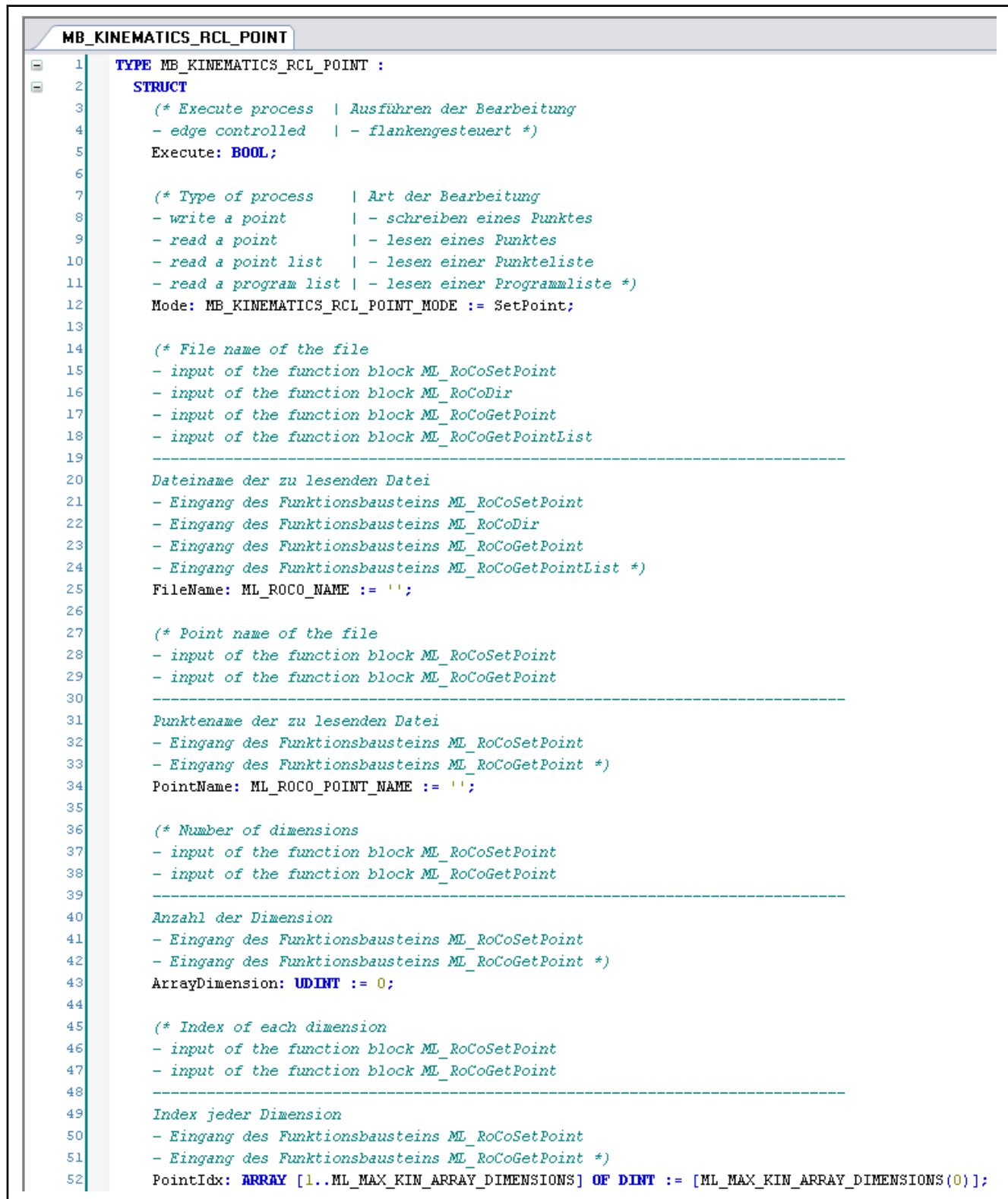


図 7-8: MB_KINEMATICS_RCL_POINT (パート 1)

キネマティクス インターフェース

```

54      (* Index of each dimension
55      - input of the function block ML_RoCoDir
56      - input of the function block ML_RoCoGetPointList
57      -----
58      Index jeder Dimension
59      - Eingang des Funktionsbausteins ML_RoCoDir
60      - Eingang des Funktionsbausteins ML_RoCoGetPointList *)
61      StartIdx: UDINT := 1;
62
63      (* Point structure with name and value
64      - input of the function block ML_RoCoSetPoint
65      -----
66      Struktur eines Punktes mit Namen und Werte
67      - Eingang des Funktionsbausteins ML_RoCoSetPoint *)
68      Point: ML_ROCO_POINT;
69
70      (* Direction of the read-sequence(low to high, high to low)
71      - input of the function block ML_RoCoDir
72      -----
73      Reihenfolge, in welcher gelesen wird (aufsteigend, absteigend)
74      - Eingang des Funktionsbausteins ML_RoCoDir *)
75      SearchDirection: BOOL := FALSE;
76  END_STRUCT
77 END_TYPE

```

図 7-9: MB_KINEMATICS_RCL_POINT (パート 2)

Mode この要素はアクセスの種類を定義します。

以下の列挙が選択可能です。

- SetPoint: 個別のポイントの書き込み
- GetPoint: 個別のポイントの読み取り
- GetPointList: ポイント リストの読み取り
- GetProgList: プログラム リストの読み取り

Library ML_KinTech.compiled-library

System IndraMotion MLC

MB_KINEMATICS_RCL_PROG

概略 構造 MB_KINEMATICS_RCL_PROG は以下の目的のために使用されます。

- Compact Flash カードからコントロールの内蔵メモリへの RCL プログラムのロード
- コントロールの内蔵メモリから Compact Flash カードへの RCL プログラムの保存
- コントロール内の RCL プログラムのコンパイル



Robot Control V1 の場合のみ許可されます。

キネマティクス インターフェース

パラメータ K-0-1060、K-0-1061 および K-0-1062 は以下のアクションに使用されます。

MB_KINEMATICS_RCL_PROG	
1	TYPE MB_KINEMATICS_RCL_PROG :
2	STRUCT
3	(* Name of the RCL program which will be loaded or saved
4	- the name is written to the parameter K-0-0220 and K-0-0221
5	-----
6	Name des RCL-Programms, welcher geladen oder gespeichert werden soll
7	- der Name wird auf den Parameter K-0-0220 und K-0-0221 geschrieben *)
8	ProgName: STRING [12] := '';
9	
10	(* copy a RCL program from the Compact Flash card to the control's
11	internal memory
12	- edge controlled
13	-----
14	Laden eines RCL-Programms von der Compact Flash Karte in den internen
15	Speicher der Steuerung
16	- flankengesteuert *)
17	ProgLoad: BOOL := FALSE ;
18	
19	(* save a RCL program from the control's internal memory to the
20	Compact Flash card
21	- edge controlled
22	-----
23	Speichern eines RCL-Programms vom internen Speicher der Steuerung auf die
24	Compact Flash Karte
25	- flankengesteuert *)
26	ProgSave: BOOL := FALSE ;
27	
28	(* compile a RCL program at the controller
29	- edge controlled
30	-----
31	Kompilieren eines RCL-Programms in der Steuerung
32	- flankengesteuert *)
33	ProgCompile: BOOL := FALSE ;
34	END_STRUCT
35	END_TYPE

図 7-10: MB_KINEMATICS_RCL_PROG

ProgName	この要素には、読み取られる、または保存される RCL プログラムの名前が含まれています。名前はパラメータ K-0-0220 および K-0-0221 に内部的に書き込まれます。
ProgLoad	この要素は、パラメータ K-0-0220 で指定されたファイルを Compact Flash カードからコントロールの内蔵メモリにコピーするコマンドを起動します。この目的のためにパラメータ K-0-1060 が内部的に使用されます。
ProgSave	この要素は、パラメータ K-0-0220 で指定されたファイルをコントロールの内蔵メモリから Compact Flash カードにコピーするコマンドを起動します。この目的のためにパラメータ K-0-1061 が内部的に使用されます。
ProgCompile	この要素は、パラメータ K-0-0221 で指定された QLL ファイルをコンパイルします。この目的のためにパラメータ K-0-1062 が内部的に使用されます。

キネマティクス インターフェース

Library ML_KinTech.compiled-library

System IndraMotion MLC

MB_KINEMATICS_SETUP 構造

概略 MB_KINEMATICS_SETUP 構造は、MB_KINEMATICS_CONTROL_TYPE01 の下位構造で、キネマティクスをジョグするための制御変数が含まれています。

下の図は、MB_KINEMATICS_SETUP 構造を示したものです。

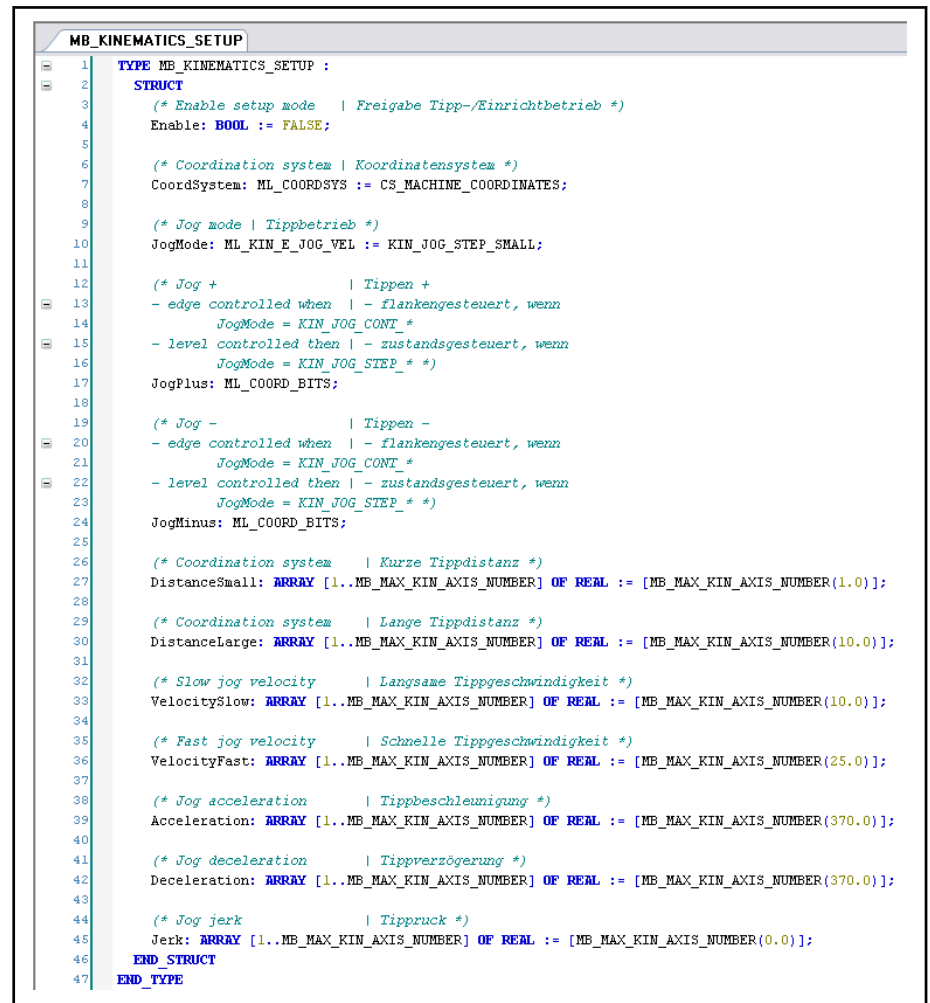


図 7-11: MB_KINEMATICS_SETUP

Enable この要素は、ジョグ/セットアップ モードを有効にします。

CoordSystem この要素により、ジョグの座標系が定義されます。

Robot Control V1 では以下の値が選択可能です。

- CS_MACHINE_COORDINATES
(KIN_CORD_SYS_ORIGIN_COORD に内部的に対応)
- CS_AXES_COORDINATES
(KIN_CORD_SYS_JOINT_COORD に内部的に対応)
- PRODUCT_COORDINATES_1
(KIN_CORD_SYS_WORKPIECE_COORD に内部的に対応)
- CS_GRIPPER_COORDINATES

キネマティクス インターフェース

(KIN_CORD_SYS_GRIPPER_COORD に内部的に対応)

Robot Control V2 では以下の値が選択可能です。

- CS_MACHINE_COORDINATES
- CS_AXES_COORDINATES
- CS_BASE_COORDINATES
- CS_PRODUCT_COORDINATES_1
- CS_PRODUCT_COORDINATES_2
- ...
- CS_PRODUCT_COORDINATES_16

JogMode	この要素はジョグ タイプ (小さな/大きな増分の動作、低速/高速の連続動作) を定義します。
JogPlus	この要素には、すべてのキネマティクス軸のビットが含まれています。それぞれのビットを設定することにより、指定に基づいた正の方向への横断動作がトリガーされます。
JogMinus	この要素には、すべてのキネマティクス軸のビットが含まれています。それぞれのビットを設定することにより、指定に基づいた負の方向への横断動作がトリガーされます。
DistanceSmall	この要素には、軸およびデカルト座標系での増分ジョグの短い距離が含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0080 および K-0-0080 に書き込まれます。
DistanceLarge	この要素には、軸およびデカルト座標系での増分ジョグの大きい距離が含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0081 および K-0-0081 に書き込まれます。
VelocitySlow	この要素には、軸およびデカルト座標系での連続的なジョグの遅い速度が含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0082 および K-0-0082 に書き込まれます。
VelocityFast	この要素には、軸およびデカルト座標系での連続的なジョグの速い速度が含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0083 および K-0-0083 に書き込まれます。
Acceleration	この要素には、軸およびデカルト座標系での連続的な増分ジョグの加速が含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0084 および K-0-0084 に書き込まれます。
Deceleration	この要素には、軸およびデカルト座標系での連続的な増分ジョグの減速が含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0085 および K-0-0085 に書き込まれます。
Jerk	この要素には、軸およびデカルト座標系での連続的な増分ジョグの最大の加速および減速ジャークが含まれています。Robot Control V1 を使用している場合、この値はパラメータ A-0-0086/87 および K-0-0086/87 に書き込まれます。
Library	ML_KinTech.compiled-library
System	IndraMotion MLC

MB_KINEMATICS_COORDINATED 構造

概略 MB_KINEMATICS_COORDINATED 構造は、MB_KINEMATICS_CONTROL_TYPE01 の下位構造で、位置設定モードのための制御変数とコマンド値が含まれています。

この構造に含まれる要素は、ファンクションブロックの入力に対応します。

- ML_MoveLinearAbsolute
- ML_MoveLinearRelative

キネマティクス インターフェース

- ML_MoveDirectAbsolute
- ML_MoveDirectRelative
- ML_MoveCircularAbsolute
- ML_MoveCircularRelative

詳細については、各資料を参照してください。

キネマティクス インターフェース

下の図は、MB_KINEMATICS_COORDINATED 構造を示したものです。

MB_KINEMATICS_COORDINATED	
1	TYPE MB_KINEMATICS_COORDINATED:
2	STRUCT
3	(* position for an absolute move (Unit: e.g. in [degree] depending on
4	parametrization) (negative or positive)
5	- start moving with positionings opmodes
6	Note: cyclic checked AND transmitted on change (EnableCyclicScanning = TRUE)
7	-----
8	Zielposition, absolut oder relativ (Einheit: z.B. in [Grad] je nach Parametrierung)
9	(negativ oder positiv)
10	- Freigabe mit den Positinierbetriebsarten
11	Hinweis: wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE) *)
12	Point: ARRAY [1..16] OF REAL := [16(0.0)];
13	-----
14	(* maximum velocity (Unit: e.g. in [rpm] depending on parametrization)
15	(always positive)
16	- start moving with positionings opmodes
17	-----
18	Positioniergeschwindigkeit, (Einheit: z.B. in [U/min] je nach Parametrierung)
19	(positiv eingeben)
20	- Freigabe mit Positinierbetriebsarten *)
21	Velocity: REAL := 10.0;
22	-----
23	(* Value of the acceleration (Unit: e.g. in [rad/s2] depending on
24	parametrization) (always positive)
25	- start moving with positionings opmodes
26	-----
27	Beschleunigung, (Einheit: z.B. in [rad/s2] je nach Parametrierung) (immer positiv)
28	- Freigabe mit Positinierbetriebsarten *)
29	Acceleration: REAL := 10.0;
30	-----
31	(* Value of the deceleration (Unit: e.g. in [rad/s2] depending on
32	parametrization) (always positive)
33	- start moving with Positinierbetriebsarten
34	-----
35	Verzögerung, (Einheit: z.B. in [rad/s2] je nach Parametrierung) (immer positiv)
36	- Freigabe mit Positinierbetriebsarten *)
37	Deceleration: REAL := 10.0;
38	-----
39	(* Value of the jerk (Unit: e.g. in [rad/s3] depending on parametrization)
40	(always positive)
41	- start moving with positionings opmodes
42	-----
43	Ruck, (Einheit: z.B. in [rad/s3] je nach Parametrierung) (immer positiv)
44	- Freigabe mit Positinierbetriebsarten *)
45	Jerk: REAL := 100.0;
46	-----
47	(* defines in which coordinate system the points have to be specified
48	(machine coordinates or joint coordinates)
49	- start moving with positionings opmodes
50	-----
51	Koordinatensystem (Maschinen- oder Achs-Koordinaten)
52	- Freigabe mit Positinierbetriebsarten *)
53	CoordSystem: ML_COORDSYS := CS_MACHINE_COORDINATES;
54	-----
55	(* Define the slope type of the motion (BlockSlope, ProgramSlope or ContinuousSlope)
56	- start moving with positionings opmodes
57	-----
58	Slope-Art der Bewegung (BlockSlope, ProgramSlope or ContinuousSlope)
59	- Freigabe mit Positinierbetriebsarten *)
60	SlopeType: ML_SLOPE_TYPE := ST_BLOCK_SLOPE;
61	-----
62	(* Defines whether the motion ends at the end point or is blended (to or via)
63	- start moving with positionings opmodes
64	-----
65	definiert ob die Bewegung am Endpunkt endet oder überschliffen wird (to oder via)
66	- Freigabe mit Positinierbetriebsarten *)
67	SetMode: ML_SET_MODE := SM_TO;
68	-----
69	(* Defines the blending radius (Unit: [%] from the parameter A-0-0230)
70	- start moving with positionings opmodes
71	-----
72	Überschleifradius (Unit: [%] vom Parameter A-0-0230)
73	- Freigabe mit Positinierbetriebsarten *)
74	BlendingRadius: REAL := 0.0;
75	-----
76	(* position for an circular move (Unit: e.g. in [degree] depending on
77	parametrization) (negative or positive)
78	- start moving with circular positionings opmodes
79	-----
80	Zielposition, zirkular (Einheit: z.B. in [Grad] je nach Parametrierung)
81	(negativ oder positiv)
82	- Freigabe mit Positinierbetriebsarten "Zircular" *)
83	CircularAuxPoint: ARRAY [1..16] OF REAL := [16(0.0)];
84	-----

図 7-12: MB_KINEMATICS_COORDINATED (1°-1)

キネマティクス インターフェース

```

85      (* Defines the interpretation of the "AuxPoint" input
86      - start moving with circular positionings opmodes
87
88      -----
89      Definiert die Interpretation des Eingangs "AuxPoint"
90      - Freigabe mit Positinerbetriebsart "Zircular" *)
91      CircularCircMode: ML_CIRCMODES := CM_BOARDER;
92
93  END_STRUCT
94  END_TYPE

```

図 7-13: MB_KINEMATICS_COORDINATED (パート 2)

Library ML_KinTech.compiled-library

System IndraMotion MLC

MB_KINEMATICS_RCL 構造

概略 MB_KINEMATICS_RCL 構造は、MB_KINEMATICS_CONTROL_TYPE01 の下位構造で、位置設定モードのための制御変数とコマンド値が含まれています。

この構造に含まれる要素は、ファンクションブロック "MB_IRDProgSelect" の入力に対応します。詳細については、各資料を参照してください。



Robot Control V1 の場合のみ許可されます。

下の図は、MB_KINEMATICS_RCL 構造を示したものです。

```

MB_KINEMATICS_RCL
1  TYPE MB_KINEMATICS_RCL :
2  STRUCT
3      (* Name of the IRD program without extension
4      -----
5      Name des IRD-Programms ohne Dateinamen-Erweiterung *)
6      ProgName: STRING[12] := '';
7  END_STRUCT
8  END_TYPE

```

図 7-14: MB_KINEMATICS_COORDINATED

Library ML_KinTech.compiled-library

System IndraMotion MLC

MB_KINEMATICS_STATUS_TYPE01 構造**MB_KINEMATICS_STATUS_TYPE01**

概略 arKinStatus_gb[] インスタンスは MB_KINEMATICS_STATUS_TYPE01 構造にアクセスします。

キネマティクス インターフェース

これには以下の要素が含まれています。

MB_KINEMATICS_STATUS_TYPE01

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

TYPE MB_KINEMATICS_STATUS_TYPE01 :

STRUCT

(* Pointer to KinData of this kinematic

- Type .PtrKinData^. for the intellisense menu

Zeiger auf KinData-Schnittstelle der Kinematik

- Eingabe .PtrKinData^. fuer Auswahlmenue*)

PtrKinData: POINTER TO ML_KINEMATICS_DATA_SM := 0;

(* Status of the kinematic | Kinematik-Status *)

Admin: MB_KINEMATICS_ADMIN_STATUS;

(* kinematic diagnosis information

- error, warning, etc.

Diagnose Informationen der Kinematik

- Fehler, Warnung, usw. *)

Diag: MB_KINEMATICS_DIAGNOSIS;

(* Jog functionality | Tipp/Einrichtungsfunktionen *)

SetupMode: MB_KINEMATICS_SETUP_STATUS;

END_STRUCT

END_TYPE

図 7-15: MB_KINEMATICS_STATUS_TYPE01

PtrKinData	キネマティクスの KinData インターフェースへのアクセス
Admin	キネマティクス管理のステータス属性へのアクセス
Diag	診断出力へのアクセス
SetupMode	ジョグ/セットアップ機能のステータス属性へのアクセス
Library	ML_KinTech.compiled-library
System	IndraMotion MLC

MB_KINEMATICS_STATUS_TYPE01 の要素

MB_KINEMATICS_ADMIN_STATUS 構造

概略 The MB_KINEMATICS_ADMIN_STATUS 構造は、MB_KINEMATICS_STATUS_TYPE01 の下位構造で、キネマティクス管理のための出力ステータスが含まれています。

キネマティクス インターフェース

下の図は、MB_KINEMATICS_ADMIN_STATUS 構造を示したものです。



図 7-16: MB_KINEMATICS_ADMIN_STATUS

_OpModeAck	この要素は、運転モードのステータスを返します。
MODE_COORD_AB ... AH	これらの要素は、プレースホルダーとしてのみの役割を果たし、制御構造内でのみ許可されます。
MODE_COORD_HOMING ... RCL	これらの要素は、運転モードのステータスをビットで返します。いずれのビットも設定されていない場合、これはステータス "ModeCoordUngrouped" を示します。
Active	この要素は、構造 arKinCtrl_gb[] および arKinStatus_gb[] が有効 (= TRUE) か無効 (= FALSE) かを示します。これはキネマティクス インターフェースの初期化中に設定されます。
Name	この要素はキネマティクスの名前を示します。これはキネマティクス インターフェースの初期化中に設定されます。
TrafoNb	この要素はキネマティクスの変換番号を示します。これはキネマティクス インターフェースの初期化中に設定されます。
CmdDone	この要素は、最後に発行されたコマンドが正常に完了したか否かを示します。

キネマティクス インターフェース

	 以下の運転モードの場合のみ有効です。 • ModeCoordContinue • ModeCoordInterrupted • ModeCoordPosDirAbs • ModeCoordPosDirRel • ModeCoordPosLinAbs • ModeCoordPosLinRel • ModeCoordPosCircAbs • ModeCoordPosCircRel • ModeCoordStopping • ModeCoordRCL
InBuffer	この要素は、最後に発行されたコマンドがブロック バッファに正常に入力されたか否かを示します。
	 以下の運転モードの場合のみ有効です。 • ModeCoordContinue • ModeCoordInterrupted • ModeCoordPosDirAbs • ModeCoordPosDirRel • ModeCoordPosLinAbs • ModeCoordPosLinRel • ModeCoordPosCircAbs • ModeCoordPosCircRel • ModeCoordStopping
ParamLevelAck	この要素は、キネマティクスに属するすべての軸がパラメータ化モード 1 であるか否かを示します。
SetConveyorBeltDone	この要素は、ベルトの設定が正常に完了したか否かを示します。
KinTransformPLCDone	この要素は、ユーザー定義の変換が正常に登録されたか否かを示します。
RCL	RCL ポイントおよびプログラムを処理するためのステータス属性へのアクセス 詳細については、158 ページ "MB_KINEMATICS_RCL_STATUS" 章 を参照してください
Library	ML_KinTech.compiled-library
System	IndraMotion MLC
	MB_KINEMATICS_ADMIN_STATUS の要素
	MB_KINEMATICS_RCL_STATUS
概略	MB_KINEMATICS_RCL_STATUS は、MB_KINEMATICS_RCL_POINT および MB_KINEMATICS_RCL_PROG のステータス構造を表します。

下の図は、MB_KINEMATICS_RCL_STATUS 構造を示したものです。

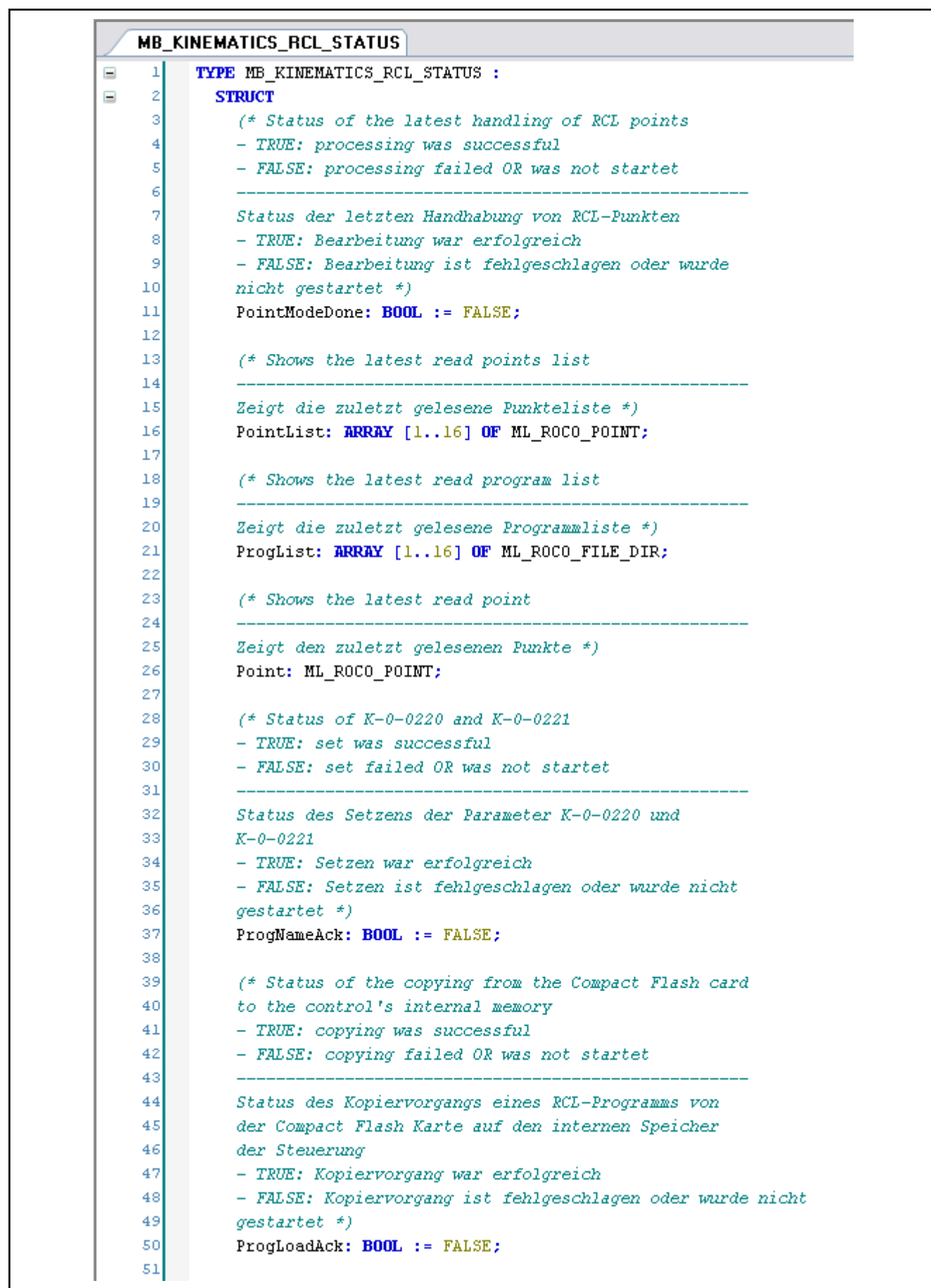


図 7-17: MB_KINEMATICS_RCL_STATUS のデータ タイプ (パート 1)

キネマティクス インターフェース

```

52      (* Status of the copying from the control's internal
53      memory to the Compact Flash card
54      - TRUE: copying was successful
55      - FALSE: copying failed OR was not startet
56      -----
57      Status des Kopiervorgangs eines RCL-Programms vom
58      internen Speicher der Steuerung auf der Compact Flash
59      Karte
60      - TRUE: Kopiervorgang war erfolgreich
61      - FALSE: Kopiervorgang ist fehlgeschlagen oder wurde nicht
62      gestartet *)
63      ProgSaveAck: BOOL := FALSE;
64
65      (* Status of the compile process of a RCL program
66      - TRUE: compile process was successful
67      - FALSE: compile process failed OR was not startet
68      -----
69      Status des Kompiliervorgangs eines RCL-Programms
70      - TRUE: Kompiliervorgangs war erfolgreich
71      - FALSE: Kompiliervorgangs ist fehlgeschlagen oder wurde nicht
72      gestartet *)
73      ProgCompileAck: BOOL := FALSE;
74  END_STRUCT
75  END_TYPE

```

図 7-18: MB_KINEMATICS_RCL_STATUS のデータタイプ (パート 2)

PointModeDone	この要素は、ポイントまたはポイント/プログラム リストを書き込むため、もしくは読み取るための最後のコマンドが正常に完了した (= TRUE) か、または正常に完了しなかった (= FALSE) かを示します。
PointList	この要素は最後に読み取られたポイント リストを表示します。
ProgList	この要素は最後に読み取られたプログラム リストを表示します。
Point	この要素は最後に読み取られたポイントを表示します。
ProgNameAck	この要素は、arKinCtrl_gb[].RCLMode.ProgName で指定された名前の書き込みが正常に行われた (= TRUE) か、または正常に行われなかった (= FALSE) かを示します。
ProgLoadAck	この要素は、arKinCtrl_gb[].RCLMode.ProgName で指定されたプログラムの、コントロールの内蔵メモリへの読み込みが正常に行われた (= TRUE) か、または正常に行われなかった (= FALSE) かを示します。
ProgSaveAck	この要素は、arKinCtrl_gb[].RCLMode.ProgName で指定されたプログラムの、Compact Flash カードへの保存が正常に行われた (= TRUE) か、または正常に行われなかった (= FALSE) かを示します。
ProgCompileAck	この要素は、arKinCtrl_gb[].RCLMode.ProgName で指定されたプログラムの、コントロールでのコンパイルが正常に行われた (= TRUE) か、または正常に行われなかった (= FALSE) かを示します。
Library	ML_KinTech.compiled-library
System	IndraMotion MLC

キネマティクス インターフェース

MB_KINEMATICS_DIAGNOSIS 構造

概略 MB_KINEMATICS_DIAGNOSIS 構造は、MB_KINEMATICS_STATUS_TYPE01 の下位構造で、警告およびエラー出力が含まれています。

下の図は、MB_KINEMATICS_DIAGNOSIS 構造を示したものです。

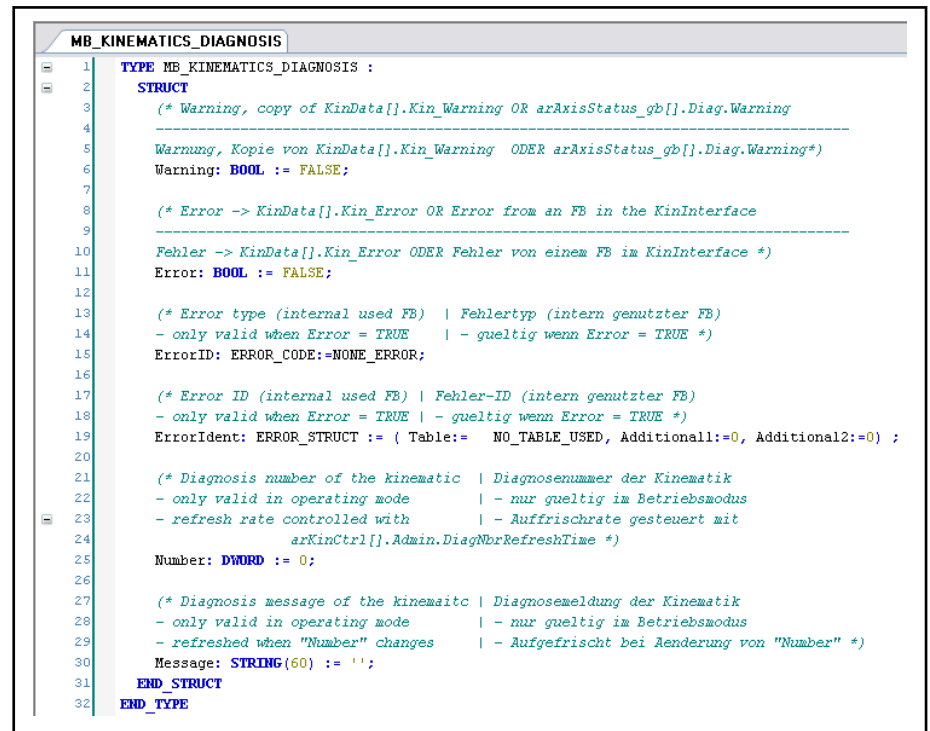


図 7-19: MB_KINEMATICS_DIAGNOSIS

arKinCtrl_gb[].Admin.DiagNbrRefreshTime は要素 "Number" の更新頻度を制御します。要素 "Number" が変更されると、直ちにテキスト "Message" が再度読み取られます。また、arKinCtrl_gb[].Admin.DiagNbrRefreshTime は読み取り診断情報 (DiagNbrRefreshTime := T#0ms) のスイッチオフも行います。

Warning	この要素は、KinData[].Kin_Warning または arAxisStatus_gb[].Diag.Warning の複製で、キネマティクスまたはその軸が警告を発していることを示します。
Error	この要素は、キネマティクス、その軸またはキネマティクス インターフェースの内部ファンクションブロックがエラーを発していることを示します。
ErrorID	この要素は、キネマティクス、その軸のいずれかまたは内部ファンクションブロックのエラー タイプを返します。
ErrorIdent	この要素は、キネマティクス、その軸のいずれかまたは内部ファンクションブロックのエラー ID を返します。
Number	この要素は、キネマティクス、その軸のいずれかまたは内部ファンクションブロックの診断番号を返します。
Message	この要素は、キネマティクス、その軸のいずれかまたは内部ファンクションブロックの診断メッセージを返します。
Library	ML_KinTech.compiled-library
System	IndraMotion MLC

キネマティクス インターフェース

MB_KINEMATICS_SETUP_STATUS 構造

概略

MB_KINEMATICS_SETUP_STATUS 構造は、MB_KINEMATICS_STATUS_TYPE01 の下位構造で、キネマティクスをジョグするための制御変数が含まれています。

下の図は、MB_KINEMATICS_SETUP_STATUS 構造を示したものです。

```
MB_KINEMATICS_SETUP_STATUS
1  TYPE MB_KINEMATICS_SETUP_STATUS :
2  STRUCT
3    (*Status enable jog mode | Status der Tipp-/Einrichtbetriebsfreigabe*)
4    EnableAck: BOOL := FALSE;
5  END_STRUCT
6  END_TYPE
```

図 7-20: MB_KINEMATICS_SETUP_STATUS

EnableAck

この要素は、ジョグ/セットアップ モードが有効 (= TRUE) か、または無効 (= FALSE) かを示します。

Library

ML_KinTech.compiled-library

System

IndraMotion MLC

7.2.3 キネマティクス インターフェースのファンクションブロック

ML_KinematicsInitType01

概略

ファンクションブロック ML_KinematicsInitType01 は、あらゆるキネマティクスのためにキネマティクス インターフェースを初期化するために使用します。このファンクションブロックは、プログラムを起動するときに 1 回、および/またはパラメータ化モードから運転モードにフェーズが切り替えられるたびに呼び出す必要があります。

インターフェースの説明

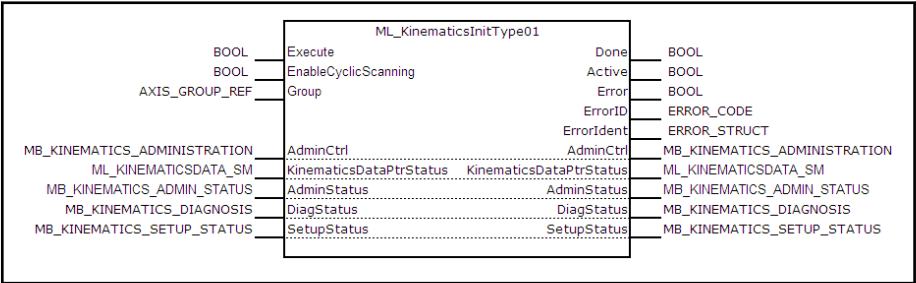


図 7-21: ファンクションブロック ML_KinematicsInitType01

I/O タイプ	名称	データの種類	備考
VAR_INPUT	Execute	BOOL	正のエッジによる初期化の開始
	EnableCyclicScanning	BOOL	現在の運転モードで変更された入力の周期的テストと送信
	Group	BOOL	キネマティクス インターフェースで初期化されるキネマティクス

キネマティクス インターフェース

I/O タイプ	名称	データの種類	備考
VAR_OUTPUT	Done	BOOL	ファンクションブロックが処理を完了したときに設定
	Active	BOOL	ファンクションブロックが有効な (アイドル モードではない) 場合に設定
	Error	BOOL	ファンクションブロック インスタンスでエラーが発生したことを示します。
	ErrorID	ERROR_CODE	エラーの原因に関する短い注記
	ErrorIdent	ERROR_STRUCT	エラーに関する詳細な情報
VAR_IN_OUT	AdminCtrl	MB_KINEMATICS_ADMINISTRATION	軸管理 arKinCtrl_gb[].Admin
	KinematicsDataPtrStatus	ML_KINEMATICS_DATA_POINTER	このキネマティクスの KinData へのポインタ arKinStatus_gb[].PtrAxisData
	AdminStatus	MB_KINEMATICS_ADMIN_STATUS	キネマティクスのステータス管理 arKinStatus_gb[].Admin
	DiagStatus	MB_KINEMATICS_DIAGNOSIS	キネマティクス診断情報 arKinStatus_gb[].Diag
	SetupStatus	MB_KINEMATICS_SETUP_STATUS	キネマティクスのジョグ/セットアップ機能のステータス arKinStatus_gb[].SetupMode

表 7-3: ML_KinematicsInitType01 のインターフェース変数



完全な構造 "MB_Kinematics_Control_Type01" および "MB_Kinematics_Status_Type01" を、個別の構造要素 "AdminCtrl"、"KinematicsDataPtrStatus" 等の代わりにファンクションブロックに転送することは不可能です。従ってユーザー アプリケーションで既存のコードに特定の拡張を追加することはできません。この結果、"MB_Kinematics_Control_Type01" および "MB_Kinematics_Status_Type01" の必須要素は、独立した入力として転送されます。

詳細については、195 ページ "キネマティクス インターフェースのユーザー拡張" 7.2.7 章を参照してください。



ファンクションブロック ML_KinematicsInitType01 は IndraMotion MLC コントロールのみで使用できます。

機能説明

ファンクションブロック ML_KinematicsInitType01 は以下の構造要素をデフォルト値で初期化します。

構造要素	デフォルト
AdminCtrl._OpMode	ModeCoordAB
AdminCtrl.ClearError	FALSE
AdminCtrl.EnableCyclicScanning	EnableCyclicScanning input
KinematicsDataPtrStatus	ADR(KinData[Group.GroupNo])
AdminCtrl.Group	Input Group
AdminStatus._OpModeAck	ModeCoordUngrouped

キネマティクス インターフェース

構造要素	デフォルト
AdminStatus.Active	FALSE
AdminStatus.Name	''
AdminStatus.TrafoNb	999
AdminStatus.CmdDone	FALSE
AdminStatus.InBuffer	FALSE
AdminStatus.ParamLevelAck	FALSE
AdminStatus.SetConveyorBeltDone[1]	FALSE
AdminStatus.SetConveyorBeltDone[2]	FALSE
AdminStatus.SetConveyorBeltDone[3]	FALSE
AdminStatus.SetConveyorBeltDone[4]	FALSE
AdminStatus.KinTransformPLCDone	FALSE
AdminStatus.RCL.PointModeDone	FALSE
AdminStatus.RCL.ProgNameAck	FALSE
AdminStatus.RCL.ProgLoadAck	FALSE
AdminStatus.RCL.ProgSaveAck	FALSE
AdminStatus.RCL.ProgCompileAck	FALSE
DiagStatus.Error	FALSE
DiagStatus.ErrorID	NONE_ERROR
DiagStatus.ErrorIdent.Table	NO_TABLE_USED
DiagStatus.ErrorIdent.Additional1	0
DiagStatus.ErrorIdent.Additional2	0
DiagStatus.Number	0
DiagStatus.Message	'reading not activated' (読み取りは有効化されていません)
SetupStatus.EnableAck	FALSE

表 7-4: *ML_KinematicsInitType01* により初期化される構造要素

ファンクションブロック *ML_KinematicsInitType01* は以下のパラメータを読み取り、以下の構造要素に値を書き込みます。

K-パラメータ (名称)	構造要素
K-0-0031 (変換スキーム)	AdminStatus.TrafoNb
K-0-0002 (キネマティクス名)	AdminStatus.Name
K-0-0008 (軸設定リスト)	内部目的の場合のみ必要

表 7-5: *ML_KinematicsInitType01* により読み取られるパラメータ**EnableCyclicScanning:**

"EnableCyclicScanning" 入力により、現在の運転モードの変更済み入力の周期的なテストおよび送信が可能になります。これは、`arkinCtrl_gb[]` の値で以下のコメントを伴うものに関連します: "Cyclically tested and transmitted in case of change (変更の場合は周期的にテストされ、送信されます)。" デフォルト値は TRUE です。

キネマティクス インターフェース

Group: "Group" 入力はこのキネマティクスを初期化するかを示します。

エラー処理 ファンクションブロックは、表 "F_RELATED_TABLE"、16#0170 の DiagStatus.ErrorIdent.Additional1/Additional2 で以下のエラー メッセージを生成します。

ErrorID	Additional1	Additional2	説明
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	無効なファンクションブロックの状態
INPUT_INVALID_ERROR, (16#0001)	16#2600	16#0000	無効なキネマティクス番号 "Group.GroupNo"
INPUT_INVALID_ERROR, (16#0001)	16#2600	16#0001	無効な制御番号 "Group.CntrlNo"。これは LOCAL_CNTRL に設定する必要あり

表 7-6: ML_KinematicsInitType01 のエラー コード

ML_KinematicsInterfaceType01

概略 ファンクションブロック ML_KinematicsInterfaceType01 は、あらゆるキネマティクスのためにキネマティクス インターフェースを設定するために使用します。このファンクションブロックは、ユーザーが運転モードにある限り、周期的に呼び出す必要があります。

インターフェースの説明

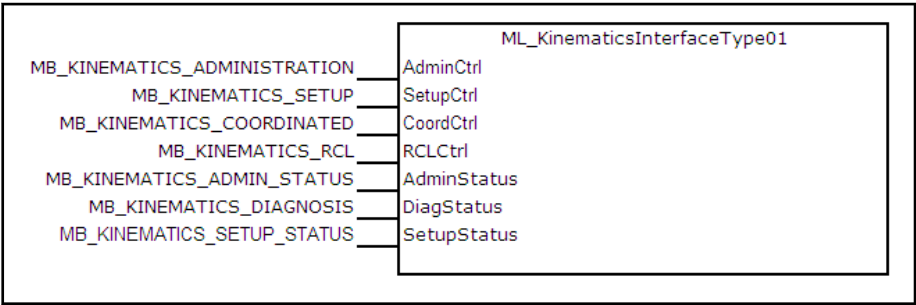


図 7-22: ファンクションブロック ML_KinematicsInterfaceType01

キネマティクス インターフェース

I/O タイプ	名称	データの種類	備考
VAR_INPUT	AdminCtrl	REFERENCE TO MB_KINEMATICS_ADMINISTRATION	軸管理 arKinCtrl_gb[].Admin
	SetupCtrl	REFERENCE TO MB_KINEMATICS_SETUP	ジョグ/セットアップ操作 arKinCtrl_gb[].SetupMode
	CoordCtrl	REFERENCE TO MB_KINEMATICS_COORDINATED	運転モードの協調運動 arKinCtrl_gb[].CoordMode
	RCLCtrl	REFERENCE TO MB_KINEMATICS_RCL	運転モード RCL arKinCtrl_gb[].RCLMode
	AdminStatus	REFERENCE TO MB_KINEMATICS_ADMIN_STATUS	軸のステータス管理 arKinStatus_gb[].Admin
	DiagStatus	REFERENCE TO MB_KINEMATICS_DIAGNOSIS	軸のステータス診断 arKinStatus_gb[].Diag
	SetupStatus	REFERENCE TO MB_KINEMATICS_SETUP_STATUS	ジョグ/セットアップ操作のステータス arKinStatus_gb[].SetupMode

表 7-7: ML_KinematicsInterfaceType01 のインターフェース変数



完全な構造 "MB_Kinematics_Control_Type01" および "MB_Kinematics_Status_Type01" を、個別の構造要素 "AdminCtrl"、"SetupCtrl" 等の代わりにファンクションブロックに転送することは不可能です。従ってユーザー アプリケーションで既存のコードに特定の拡張を追加することはできません。この結果、"MB_Kinematics_Control_Type01" および "MB_Kinematics_Status_Type01" の必須要素は、独立した入力として転送されます。

詳細については、[195 ページ "キネマティクス インターフェースのユーザー拡張" 7.2.7 章](#) を参照してください。

パフォーマンスを最適化するには、VAR_IN_OUT または VAR_INPUT としてではなく、REFERENCE TO により構造を接続します。これにより、入力の初期化は 1 回のみで済みます。この方法により、ファンクションブロックが周期的に呼び出されても、構造を転送する必要はなくなります。



ファンクションブロック ML_KinematicsInterfaceType01 は IndraMotion MLC コントロールのみで使用できます。

キネマティクス インターフェース

エラー処理 ファンクションブロックは、表 "F_RELATED_TABLE"、16#0170 の DiagStatus.ErrorIdent.Additional1/Additional2 で以下のエラー メッセージを生成します。

ErrorID	Additional1	Additional2	説明
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	無効な内部状態
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0001	無効なファンクションブロックの状態
RESOURCE_ERROR, (16#0003)	16#2600	16#0001	無効な制御番号 "Group.CntrlNo"。これは LOCAL_CNTRL に設定する必要あり
DEVICE_ERROR, (16#0008)	16#2601	16#0000	キネマティクスが KinData 構造でエラーを発している
INPUT_INVALID_ERROR, (16#0001)	16#2602	OpMode の数	無効な運転モードが選択された
INPUT_INVALID_ERROR, (16#0001)	16#2603	OpMode の数	無効でサポートされていない運転モードが有効化された
INPUT_RANGE_ERROR, (16#0006)	16#2604	16#0000	少なくとも1つの入力頻度 (AdminCtrl、AdminStatus など) が未設定
ACCESS_ERROR, (16#0004)	16#2605	16#0000	軸の参照が不可能かつ不要
ACCESS_ERROR, (16#0004)	16#2605	参照する軸の数	軸の参照が不可能
RESOURCE_ERROR, (16#0003)	16#2606	16#0000	セットアップ モード中に運転モードが変更された
OTHER_ERROR, (16#7FFE)	16#2607	16#0000	キネマティクスをグループ化できなかった
INPUT_INVALID_ERROR, (16#0001)	16#2609	16#0000	無効な座標系が選択された
RESOURCE_ERROR, (16#0003)	16#26FF	16#0000	無効な Robot Control バージョン

表 7-8: ML_KinematicsInterfaceType01 のエラー コード

ML_KinJogV1

概略 ファンクションブロック ML_KinJogV1 は、キネマティクスの個別の軸または軸グループをジョグするために使用します。このファンクションブロックは周期的に呼び出す必要があります。



Robot Control V1 の場合のみ許可されます。

インターフェースの説明

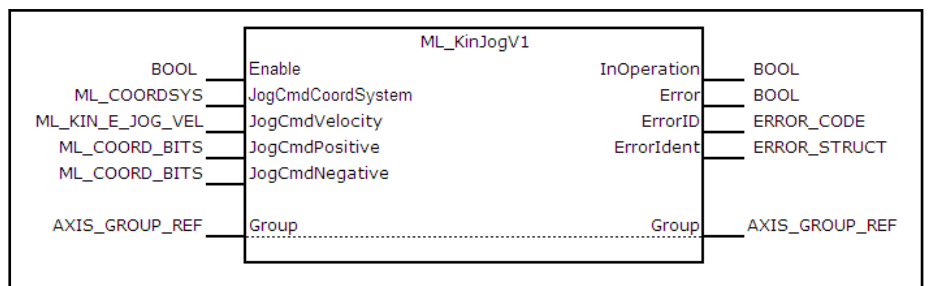


図 7-23: ML_KinJogV1 ファンクションブロック

キネマティクス インターフェース

I/O タイプ	名称	データの種類	備考
VAR_INPUT	Enable	BOOL	ファンクションブロックでの処理が有効になっている
	JogCmdCoordSystem	ML_COORDSYS	ジョグ座標系の指定
	JogCmdVelocity	ML_KIN_E_JOG_VEL	ジョグ速度の指定
	JogCmdPositive	ML_COORD_BITS	各座標につき 1 ビットを含み、これにより座標を正の方向にジョグ可能
	JogCmdNegative	ML_COORD_BITS	各座標につき 1 ビットを含み、これにより座標を負の方向にジョグ可能
VAR_OUTPUT	InOperation	BOOL	ファンクションブロックは実際のタスクでビジー状態
	エラー	BOOL	ファンクションブロック インスタンスでエラーが発生したことを示します。
	ErrorID	ERROR_CODE	エラーの原因に関する短い注記
	ErrorIdent	ERROR_STRUCT	エラーに関する詳細な情報
VAR_IN_OUT	Group	BOOL	ジョグされるキネマティクスへの参照

表 7-9: ML_KinJogV1 のインターフェース変数

名称	タイプ	最小値	最大値	デフォルト値	有効性
Enable	BOOL			FALSE	連続的
JogCmdCoordSystem	ML_COORDSYS			KIN_CORD_SYS_JOINT_COORD	連続的
JogCmdVelocity	ML_KIN_E_JOG_VEL			KIN_JOG_STOP	連続的
JogCmdPositive	ML_COORD_BITS			16(FALSE)	連続的
JogCmdNegative	ML_COORD_BITS			16(FALSE)	連続的

表 7-10: ML_KinJogV1 入力動作

 ファンクションブロック ML_KinJogV1 は IndraMotion MLC コントロールのみで使用できます。

機能説明 ML_KinJog ファンクションブロックを用いると、Robot Control V1 によりキネマティクスをジョグすることができます。ジョグは、KinData インターフェース KinData[].stJogCmd_q を介して実行されます。ファンクションブロックが正常に起動すると、キネマティクスに属する軸が自動的にグループ化されます。

- JogCmdCoordSystemこの入力は、ジョグが行われる座標系を示します。
入力は以下の値を持つことができます。
 - KIN_CORD_SYS_JOINT_COORD: 軸座標系
 - KIN_CORD_SYS_ORIGIN_COORD: 世界座標系
 - KIN_CORD_SYS_WORKPIECE_COORD: ワーク座標系
 - KIN_CORD_SYS_GRIPPER_COORD: グリッパー座標系
- JogCmdVelocityこの入力はジョグ タイプ (低速または高速の増分、低速または高速で連続的) を示します。

キネマティクス インターフェース

入力は以下の値を持つことができます。

- KIN_JOG_STOP
ジョグは無効
- KIN_JOG_STEP_SMALL
小刻みな増分ジョグ
- KIN_JOG_STEP_LARGE
大刻みな増分ジョグ
- KIN_JOG_CONT_SLOW
低速での連続的なジョグ
- KIN_JOG_CONT_LARGE
高速での連続的なジョグ

JogCmdPositive この入力により、各軸の正の方向への横断動作が有効になります。

JogCmdNegative この入力により、各軸の負の方向への横断動作が有効になります。

エラー処理 ファンクションブロックは、表 "F_RELATED_TABLE"、16#0170 を使用して ErrorIdent.Additional1/Additional2 で以下のエラー メッセージを生成します。

ErrorID	Additional1	Additional2	説明
RESOURCE_ERROR, (16#0003)	16#26FF	16#0000	無効な Robot Control バージョン

表 7-11: ML_KinJogV1 のエラー コード

ML_KinJog

概略 ファンクションブロック ML_KinJog は、キネマティクスの個別の軸または軸グループをジョグするために使用します。このファンクションブロックは周期的に呼び出す必要があります。



Robot Control V2 の場合のみ許可されます。

インターフェースの説明

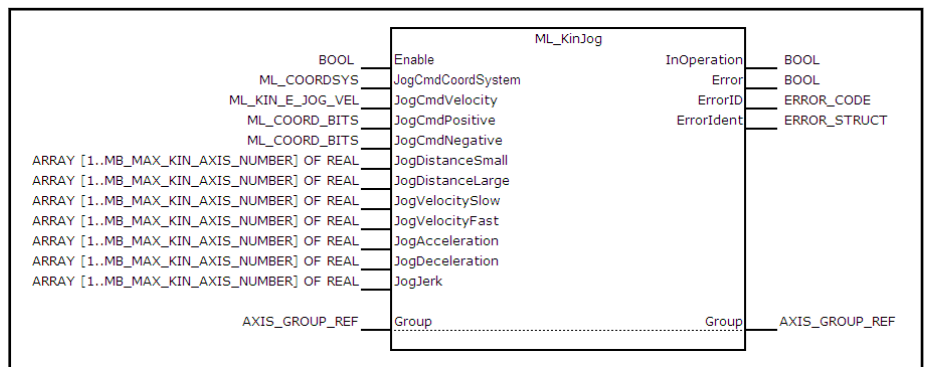


図 7-24: ML_KinJog ファンクションブロック

キネマティクス インターフェース

I/O タイプ	名称	データの種類	備考
VAR_INPUT	Enable	BOOL	ファンクションブロックでの処理が有効になっている
	JogCmdCoordSystem	ML_COORDSYS	ジョグ座標系の指定
	JogCmdVelocity	ML_KIN_E_JOG_VEL	ジョグ速度の指定
	JogCmdPositive	ML_COORD_BITS	各座標につき 1 ビットを含み、これにより座標を正の方向にジョグ可能
	JogCmdNegative	ML_COORD_BITS	各座標につき 1 ビットを含み、これにより座標を負の方向にジョグ可能
	JogDistanceSmall	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸の短いジョグ距離を指定
	JogDistanceLarge	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸の長いジョグ距離を指定
	JogVelocitySlow	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸の低速のジョグ速度を指定
	JogVelocityFast	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸の高速のジョグ速度を指定
	JogAcceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸のジョグ モードの加速を指定
	JogDeceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸のジョグ モードの減速を指定
	JogJerk	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	すべての軸のジョグ モードのジャークを指定
VAR_OUTPUT	InOperation	BOOL	ファンクションブロックは実際のタスクでビジー状態
	Error	BOOL	ファンクションブロック インスタンスでエラーが発生したことを示します。
	ErrorID	ERROR_CODE	エラーの原因に関する短い注記
	ErrorIdent	ERROR_STRUCT	エラーに関する詳細な情報
VAR_IN_OUT	Group	BOOL	ジョグされるキネマティクスへの参照

表 7-12: ML_KinJog のインターフェース変数

名称	タイプ	最小値	最大値	デフォルト値	有効性
Enable	BOOL			FALSE	連続的
JogCmdCoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	連続的
JogCmdVelocity	ML_KIN_E_JOG_VEL			KIN_JOG_STOP	連続的
JogCmdPositive	ML_COORD_BITS			16(FALSE)	連続的

キネマティクス インターフェース

名称	タイプ	最小値	最大値	デフォルト値	有効性
JogCmdNegative	ML_COORD_BITS			16(FALSE)	連続的
JogDistanceSmall	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		1.0	連続的
JogDistanceLarge	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		10.0	連続的
JogVelocitySlow	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		10.0	連続的
JogVelocityFast	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		25.0	連続的
JogAcceleration	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		370.0	連続的
JogDeceleration	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		370.0	連続的
JogJerk	ARRAY [1..MB_MAX_KIN_NUMBE R] OF REAL	>0.0		0.0	連続的

表 7-13: ML_KinJog input behavior



ファンクションブロック ML_KinJog は IndraMotion MLC コントロールのみで使用できます。

機能説明

ML_KinJog ファンクションブロックを用いると、Robot Control V2 によりキネマティクスをジョグできます。デカルト座標系でのジョグは直線的な横断モーションに対応しますが、軸座標系でのジョグは PTP 横断モーションに対応します。ファンクションブロックが正常に起動すると、キネマティクスに属する軸が自動的にグループ化されます。

JogCmdCoordSystem

この入力、ジョグが行われる座標系を示します。

入力は以下の値を持つことができます。

- CS_MACHINE_COORDINATES: 世界座標系
- CS_AXES_COORDINATES: 軸座標系
- CS_BASE_COORDINATES: 基本座標系
- CS_PRODUCT_COORDINATES_1: 製品座標系 1
- CS_PRODUCT_COORDINATES_2: 製品座標系 2
- ...
- CS_PRODUCT_COORDINATES_16: 製品座標系 16

JogCmdVelocity

この入力はジョグ タイプ (小刻みまたは大刻みな増分、低速または高速で連続的) を示します。

入力は以下の値を持つことができます。

- KIN_JOG_STOP

ジョグは無効

キネマティクス インターフェース

- KIN_JOG_STEP_SMALL
小刻みな増分ジョグ
 - KIN_JOG_STEP_LARGE
大刻みな増分ジョグ
 - KIN_JOG_CONT_SLOW
低速での連続的なジョグ
 - KIN_JOG_CONT_LARGE
高速での連続的なジョグ

JogCmdPositive

JogCmdNegative

JogDistanceSmall、
JogDistanceLarge、
JogVelocitySlow、JogVelocityFast、
JogAcceleration、JogDeceleration、
JogJerk

この入力により、各軸の正の方向への横断モーションが有効になります。

この入力により、各軸の負の方向への横断モーションが有効になります。

これらの入力は増分の連続的なジョグのジョグ距離、ジョグ速度、ジョグ加速およびジョグ ジャークを定義します。 これらの入力の単位は "JogCmdCoordSystem" に依存し、キネマティクス単位 (デカルト座標系の場合) または軸単位 (軸座標系の場合) に対応します。



連続的なジョグは、複数の横断コマンドの発行に基づきますが、入力 "JogCmdPositive" または "JogCmdNegative" の各ビットは設定されています。キネマティクスが作業領域の境界に移動した場合、早期エラー メッセージが表示されることがあり、命令されたポイントにはキネマティクスによって到達できないことを示します。この場合は、エラーを確認し、より低い速度で、または増分ジョグにより必要なポイントに移動します。

エラー処理

ファンクションブロックは、表 "F_RELATED_TABLE"、16#0170 を使用して ErrorIdent.Additional1/Additional2 で以下のエラー メッセージを生成します。

ErrorID	Additional1	Additional2	説明
RESOURCE_ERROR, (16#0003)	16#26FF	16#0000	無効な Robot Control バージョン

表 7-14: ML_KinJog のエラー コード

7.2.4

運転モード

概要

運転モードを有効にするには、以下の 2 つの方法があります。

1. Enum 値の選択
- タイプ MB_KINEMATICS_MODE の値の割当 :
- arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordAH;
2. ビット アクセスの使用
- ビット アクセス機能を使用したビットの設定
- arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_AH:= TRUE;



arKinCtrl_gb[].Admin._OpMode では一度に 1 ビットのみが設定可能です。複数のビットを設定すると、キネマティクス インターフェースでエラーが発生します。

運転モードを有効にするには、まずすべての属性に値を割り当てる必要があります。すべての属性にはデフォルト値があります。一部はゼロ以外の値が設定されていますが、0 に定義されているものもあります。特別な要求により、これらに値を割り当てる必要があります。

キネマティクス インターフェース



現在の運転モードの変更を実行する前に定義する必要があるのは、使用されている属性 (ポイント、速度など) またはデフォルト値が変更された属性のみです。

以下のセクションでは、キネマティクス インターフェースによりサポートされている運転モードおよびそれらに割り当てることが可能な属性について説明します。



キネマティクス インターフェースでは、特定の切替 (すべての軸のグループ化/グループ化解除、軸の参照など) を実行するために、内部的に軸インターフェースを使用します。

Kinematics Ready, Drives Ready (キネマティクス準備完了、ドライブ準備完了)

この運転モードが有効な場合、キネマティクスはグループ解除され、それに属しているすべての軸は AB (drive ready) に設定されます。この運転モードでは、キネマティクス インターフェースは無効になり、キネマティクスは PLCOpen 状態 "StandStill incomplete" となります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordAB;
```

または

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_AB := TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	Admin.StopDeceleration	REAL	999	あり
	Admin.StopJerk	REAL	9999	あり
arKinStatus_gb[]	Admin.MODE_COORD_AB	BOOL		該当せず

表 7-15: Kinematics ready, drives ready の属性



運転モード "ModeCoordAB" が選択された場合、キネマティクス インターフェースはステータス構造で運転モード "ModeCoordUngrouped" を返します。

Kinematics Ready, Drives Halt (キネマティクス準備完了、ドライブ停止)

この運転モードが有効な場合、キネマティクスはグループ解除され、それに属しているすべての軸は AH (drive halt) に設定されます。この運転モードでは、キネマティクス インターフェースは無効になり、キネマティクスは PLCOpen 状態 "StandStill incomplete" となります。軸は PLCOpen 状態 "StandStill" に切り替えられます。すなわち、軸インターフェースまたは別個の PLCOpen ファンクションブロックを使用して軸にコマンドを発することができます。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordAH;
```

または

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_AH:= TRUE;
```

キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	Admin.StopDeceleration	REAL	999.0	あり
	Admin.StopJerk	REAL	9999.0	あり
arKinStatus_gb[]	Admin.MODE_COORD_AH	BOOL		該当せず

表 7-16: Kinematics ready, drives halt の属性



運転モード "ModeCoordAH" が選択された場合、キネマティクス インターフェースはステータス構造で運転モード "ModeCoordUngrouped" を返します。

Kinematics Homing (キネマティクス ホーミング)

この運転モードが有効な場合、キネマティクスに属するすべての軸が参照されます。ホーミングは軸インターフェースの SetupMode を介して実行されます。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordHoming;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_HOMING := TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	Admin.HomingSequence	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	MB_MAX_KIN_AXIS_NUMBER(0.0)	あり
arKinStatus_gb[]	Admin.MODE_COORD_HOMING	BOOL		該当せず
	Admin.HomingAck	BOOL		該当せず

表 7-17: Kinematics homing の属性



運転モード "ModeCoordHoming" が選択されたときにキネマティクスがすでにグループ化されている場合、新たなホーミングは不可能なため、キネマティクス インターフェースはエラーを表示します。

Absolute PTP Positioning (絶対 PTP 位置設定)

この運転モードを有効にすることにより、絶対 PTP 動作が以前に指定されていたターゲット位置に戻ります。キネマティクスはグループ化され、PLCOpen 状態 "GroupMoving" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosDirAbs;
```

キネマティクス インターフェース

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_DIR_ABS := TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	あり
	CoordMode.Velocity	REAL	10.0	なし
	CoordMode.Acceleration	REAL	10.0	なし
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100	なし
	CoordCoordModeCtrl.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	なし
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	なし
	CoordMode.SetMode	ML_SET_MODE	SM_TO	なし
	CoordMode.BlendingRadius	REAL	0.0	なし
arKinStatus_gb[]	Admin.MODE_COORD_POS_DIR_ABS	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-18: Absolute PTP positioning の属性

Relative PTP Positioning (相対 PTP 位置設定)

この運転モードを有効にすることにより、すでに指定済みの位置の差分によって相対 PTP モーションが行われます。キネマティクスはグループ化され、PLCOpen 状態 "GroupMoving" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosDirRel;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_DIR_REL := TRUE;
```

キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_C HANGE_OPMODE	NoAction	あり
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	あり
	CoordMode.Velocity	REAL	10.0	なし
	CoordMode.Acceleration	REAL	10.0	なし
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100	なし
	CoordCoordModeCtrl.CoordSystem	ML_COORDSYS	CS_MACHINE_CO ORDINATES	なし
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOP E	なし
	CoordMode.SetMode	ML_SET_MODE	SM_TO	なし
	CoordMode.BlendingRadius	REAL	0.0	なし
arKinStatus_gb[]	Admin.MODE_COORD_POS_DIR_ REL	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-19: Relative PTP positioning の属性

Absolute Linear Positioning (絶対線形位置設定)

この運転モードを有効にすることにより、絶対線形モーションが以前に指定されていたターゲット位置に戻ります。キネマティクスはグループ化され、PLCOpen 状態 "GroupMoving" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosLinAbs;  
または  
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_LIN_ABS :=  
TRUE;
```


キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	あり
	CoordMode.Velocity	REAL	10.0	なし
	CoordMode.Acceleration	REAL	10.0	なし
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100.0	なし
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	なし
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	なし
	CoordMode.SetMode	ML_SET_MODE	SM_TO	なし
	CoordMode.BlendingRadius	REAL	0.0	なし
arKinStatus_gb[]	Admin.MODE_COORD_POS_LIN_ABS	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-20: Absolute linear positioning の属性

Relative Linear Positioning (相対線形位置設定)

この運転モードを有効にすることにより、すでに指定済みの位置の差分によって相対線形モーションが行われます。キネマティクスはグループ化され、PLCOpen 状態 "GroupMoving" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosLinRel;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_LIN_REL := TRUE;
```

キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	あり
	CoordMode.Velocity	REAL	10.0	なし
	CoordMode.Acceleration	REAL	10.0	なし
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100.0	なし
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	なし
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	なし
	CoordMode.SetMode	ML_SET_MODE	SM_TO	なし
	CoordMode.BlendingRadius	REAL	0.0	なし
arKinStatus_gb[]	Admin.MODE_COORD_POS_LIN_REL	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-21: Relative linear positioning の属性

Absolute Circular Positioning (絶対円形位置設定)

この運転モードを有効にすることにより、指定された中間点の周囲にあるすでに指定済みのターゲット位置に絶対円形モーションが戻ります。キネマティクスはグループ化され、PLCOpen 状態 "GroupMoving" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosCircAbs;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_CIRC_ABS := TRUE;
```

キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	CoordMode.AuxPoint	ARRAY [1..16] OF REAL	16(0.0)	なし
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	あり
	CoordMode.Velocity	REAL	10.0	なし
	CoordMode.Acceleration	REAL	10.0	なし
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100.0	なし
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	なし
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	なし
	CoordMode.SetMode	ML_SET_MODE	SM_TO	なし
	CoordMode.BlendingRadius	REAL	0.0	なし
	CoordMode.CircularCircMode	ML_CIRCMODES	CM_BOARDER	なし
arKinStatus_gb[]	Admin.MODE_COORD_POS_CIRC_ABS	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-22: Absolute circular positioning の属性

Relative Circular Positioning (相対円形位置設定)

この運転モードを有効にすることにより、すでに指定済みの位置の差分および指定された中間点の差分によって相対円形モーションが行われます。キネマティクスはグループ化され、PLCOpen 状態 "GroupMoving" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosCircRel;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_CIRC_REL := TRUE;
```

キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	CoordMode.CircularAuxPoint	ARRAY [1..16] OF REAL	16(0.0)	なし
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	あり
	CoordMode.Velocity	REAL	10.0	なし
	CoordMode.Acceleration	REAL	10.0	なし
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100.0	なし
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	なし
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	なし
	CoordMode.SetMode	ML_SET_MODE	SM_TO	なし
	CoordMode.BlendingRadius	REAL	0.0	なし
	CoordMode.CircularCircMode	ML_CIRCMODES	CM_BOARDER	なし
arKinStatus_gb[]	Admin.MODE_COORD_POS_CIRC_REL	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-23: Relative circular positioning の属性

Kinematics Stopping (キネマティクス停止)

この運転モードを有効にすることにより、グループ化を維持しつつ、キネマティクスが停止されます。キネマティクスは PLCOpen 状態 "Stopping" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordStopping;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_STOPPING := TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	AdminCtrl.StopDeceleration	REAL	999.0	あり
	AdminCtrl.StopJerk	REAL	9999.0	あり

キネマティクス インターフェース

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinStatus_gb[]	Admin.MODE_COORD_STOPPING	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-24: Kinematics stopping の属性



運転モード "ModeCoordStopping" は Robot Control V2 のみで利用可能です。このモードを Robot Control V1 で選択した場合、キネマティクス インターフェースはエラーを表示します。

"Interrupted" (中断) 運転モード

この運転モードを有効にすることにより、グループ化を維持しつつ、キネマティクスモーションが中断されます。キネマティクスは PLCOpen 状態 "GroupMoving" のままとなります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordInterrupted;
```

または

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_INTERRUPTED := TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	CoordMode.Deceleration	REAL	10.0	なし
	CoordMode.Jerk	REAL	100.0	なし
arKinStatus_gb[]	Admin.MODE_COORD_INTERRUPTED	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-25: 運転モード "Interrupted" の属性



運転モード "ModeCoordInterrupted" は Robot Control V2 のみで利用可能です。このモードを Robot Control V1 で選択した場合、キネマティクス インターフェースはエラーを表示します。

"Continue" (続行) 運転モード

この運転モードを有効にすることにより、以前に中断されたモーションが開始されます。キネマティクスは PLCOpen 状態 "GroupMoving" となります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordContinue;
```

または

キネマティクス インターフェース

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_CONTINUE      :=  
TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_C HANGE_OPMODE	NoAction	あり
arKinStatus_gb[]	Admin.MODE_COORD_CONTINU E	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず
	Admin.InBuffer	BOOL		該当せず

表 7-26: 運転モード "Continue" (続行) の属性

 運転モード "ModeCoordContinue" は Robot Control V2 のみで利用可能です。このモードを Robot Control V1 で選択した場合、キネマティクス インターフェースはエラーを表示します。

"外部ファンクションブロック" 運転モード

この運転モードを有効にすることにより、キネマティクスがグループ化され、外部モーションコマンドを待ちます。キネマティクスは PLCOpen 状態 "StandStill Complete" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordExternalFB;  
または  
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_EXTERNAL_FB  :=  
TRUE;
```

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的な スキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_C HANGE_OPMODE	NoAction	あり
arKinStatus_gb[]	Admin.MODE_COORD_EXTERNA L_FB	BOOL		該当せず

表 7-27: 属性、運転モード "外部ファンクションブロック"

"RCL" 運転モード

この運転モードを有効にすることにより、キネマティクスがグループ化され、arKinCtrl_gb[].RCLMode.ProgName で指定された RCL プログラムが起動します。キネマティクスは PLCOpen 状態 "ProgrammedMotion" に切り替わります。この運転モードは以下のコマンドにより有効化されます。

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordRCL;  
または  
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_RCL := TRUE;
```

キネマティクス インターフェース

下の表は、この運転モードによりサポートされている属性をまとめたものです。

要素	名称	タイプ	デフォルト	周期的なスキャン
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	あり
	RCLCtrl.ProgName	STRING	''	なし
arKinStatus_gb[]	Admin.MODE_COORD_RCL	BOOL		該当せず
	Admin.CmdDone	BOOL		該当せず

表 7-28: 運転モード "RCL" の属性



運転モード "ModeCoordRCL" は Robot Control V1 のみで利用可能です。このモードを Robot Control V2 で選択した場合、キネマティクス インターフェースはエラーを表示します。

7.2.5 グローバル変数

ML_KinTech ライブラリにはグローバル変数リストが含まれています。
Global_MB_KinInterfaceVars.

このライブラリは以下の定数を定義します。

1. arK8ConfigList_gb:

この列定数には、キネマティクスに割り当てられた軸に関する情報が含まれます。

- uiNumberOfKinAxis - 軸の合計数
- uiNumberOfMoveAxis - 位置設定軸および方向軸の数
- uiNumberOfBeltAxis - ベルト軸の数
- arAxisType - 軸のタイプ (位置設定軸、方向軸、ベルト軸または軸なし)
- arAxisNumber - 軸番号

2. arRocoKernel_gb:

この列定数には、キネマティクスの Robot Control バージョンに関する情報が含まれます。

- 1 - Robot Control V1
- 2 - Robot Control V2

3. arpAxisCtrlAdmin_gb:

このポインタ定数は、軸インターフェースの AdminCtrl 構造 ("MB_AXIS_ADMINISTRATION") をポイントします。これにより、キネマティクス インターフェースとユーザー アプリケーションの両方から軸インターフェースへの並行アクセスが可能になります。

4. arpAxisCtrlCoordMode_gb:

このポインタ定数は、軸インターフェースの CoordModeCtrl 構造 ("MB_AXIS_COORDINATED") をポイントします。これにより、キネマティクス インターフェースとユーザー アプリケーションの両方から軸インターフェースへの並行アクセスが可能になります。

5. arpAxisStatusAdmin_gb:

このポインタ定数は、軸インターフェースの AdminStatus 構造 ("MB_AXIS_ADMIN_STATUS") をポイントします。これにより、キネマ

キネマティクス インターフェース

ティクス インターフェースとユーザー アプリケーションの両方から軸インターフェースへの並行アクセスが可能になります。

6. arpAxisStatusDiag_gb:
- このポインタ定数は、軸インターフェースの DiagStatus 構造 ("MB_AXIS_DIAGNOSIS") をポイントします。これにより、キネマティクス インターフェースとユーザー アプリケーションの両方から軸インターフェースへの並行アクセスが可能になります。

7.2.6 最初の手順、キネマティクス インターフェース

概要

プログラミング テンプレート "軸インターフェース サンプル プロジェクト"と同様に、キネマティクス インターフェースはプラグインが組み込まれた場合に自動的に生成される同等のサンプル プロジェクトを提供します。このサンプル プロジェクトはタブ Logic ▶ Application ▶ AxisInterface ▶ Plugins ▶ KinInterface にあります。

キネマティクス インターフェースのプログラム

- プログラム TE_KinInterfaceType01Prog には以下のタスクがあります。
- キネマティクス インターフェースの初期化**
パラメータ化モード (フェーズ 2) に達すると、TE_KinematicsInitAllKinematicsType01 ファンクションブロックを使用しているすべての設定済みキネマティクスについて、キネマティクス インターフェースが初期化されます。運転モード (フェーズ 4) に達すると、パラメータ化での変更を認識するために、初期化が再度実行されます。
 - キネマティクス インターフェースの周期的な呼出**
初期化が正常に行われた後で ("TE_KinInterfaceType01Prog.InitDone"により示される) キネマティクス インターフェースが周期的に呼び出されます。この目的のために、ファンクションブロック "TE_KinematicsInterface" が呼び出されます。これはファンクションブロック "MB_KinInterfaceType01" に由来するため、ユーザー拡張が可能になります。
 - キネマティクス インターフェースのユーザー拡張**
ファンクションブロック "TE_KinematicsInterface" を用いると、ユーザーはキネマティクス インターフェースの構造に要素を追加し、それによってアプリケーションを最適な形で拡張できます。詳細については、[195 ページ "キネマティクス インターフェースのユーザー拡張" 7.2.7 章](#)を参照してください

キネマティクス インターフェースの表示

- 概要
- キネマティクス インターフェースには表示マスクが含まれており、キネマティクスを設定して制御するための既製の簡易インターフェースを提供します。
- これには以下の表示が含まれています。

表示	説明
SystemOverview	
• KinSystem_Overview_All	簡易診断、ステータスおよび座標を含むすべての定義済みキネマティクスの完全な概要

キネマティクス インターフェース

表示	説明
KinematicsModes	
• Kinematics_Overview	他のすべての表示へのナビゲーション ボタンに加えて、現在の運転モードおよびキネマティクス インターフェースのステータスを表示します。
• KinHoming_mode	ホーミング モードのコマンドとモニタリング
• KinPosition_mode	位置設定モードのコマンドとモニタリング
• KinRCL_mode	RCL 運転モードのコマンドとモニタリング
ユーティリティ	
• ConveyorBelt	1 本または複数のベルトの設定
• RCLPoint	RCL ポイントの設定および読み取り、RCL ポイント リストとプログラム リストの読み取り
• RCLProg	RCL プログラムの読み取り、保存およびコンパイル
• Setup	ジョグ モードのコマンドとモニタリング

表 7-29: キネマティクス インターフェースの表示

以下のグローバル変数は、表示内のシステム情報の制御およびアクセスに使用されます。

- arKinCtrl_gb[]
- arKinStatus_gb[]
- KinData[]
- VisuKinematicsNo

システム概要表示

表示 **KinSystem_Overview_All** により、プロジェクト内で設定されているすべてのキネマティクスを素早く組み込むことができます。表示されるデフォルトのキネマティクスに参照はありません。この参照をどのようにして設定する必要があるかについて以下に説明します。オフライン モードで表示するために、追加的なキネマティクスを追加できます。

表示は全体的なシステム ステータスを提供し、エラーのクリアを可能にします。個別のキネマティクス制御要素は、運転モード ステータス表示とともに、キネマティクス名とキネマティクス番号、特定の診断および世界座標としての現在の位置を返します。

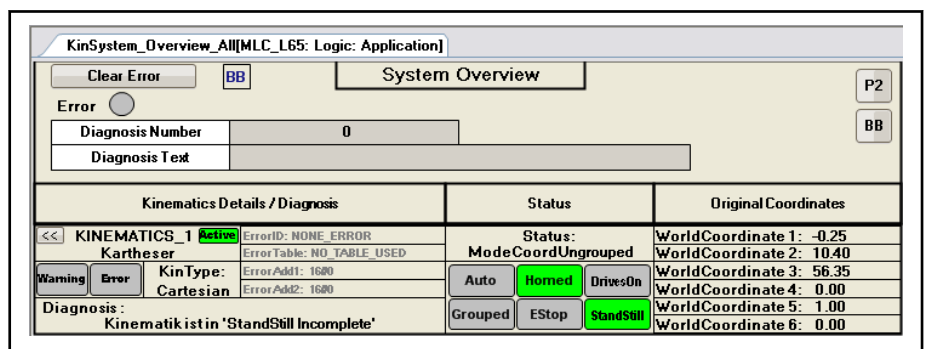


図 7-25: KinSystem_Overview_All 表示

エラーをクリア

このボタンは、IMC インターフェースを介して "Clear Error" (エラーのクリア) をトリガーします。

キネマティクス インターフェース

 キネマティクス エラーおよび軸エラーはクリアされません。これらのエラーは、各キネマティクスまたは軸の概要で個別にクリアする必要があります。

- P2

BB

システム概要へのキネマティクスの追加
- このボタンは、パラメータ化モード P2 に切り替えるために使用できます。

このボタンは、運転モード BB に切り替えるために使用できます。

新しい表示と同じ方法で、システム概要表示にキネマティクスを追加します。各要素を選択し、キネマティクス参照を指定します。システム概要に新しいキネマティクスを追加するための 2 種類のアプローチについて以下に説明します。

オプション 1：フレームによるキネマティクス表示の作成

1.

2.

3.
- 各 IndraWorks プロジェクトを開き、タブ **Logic** ▶ **Application** ▶ **AxisInterface** ▶ **Plugins** ▶ **KinInterface** ▶ **Visualizations** ▶ **SystemOverview** を選択します。

IndraLogic のオフライン モードで、表示 "KinSystem_Overview_All" をダブルクリックします。

ToolBox ウィンドウにある **Frame** (フレーム) ボタン  をクリックし、最後のキネマティクスの下に適切なサイズでアウトラインを生成します。

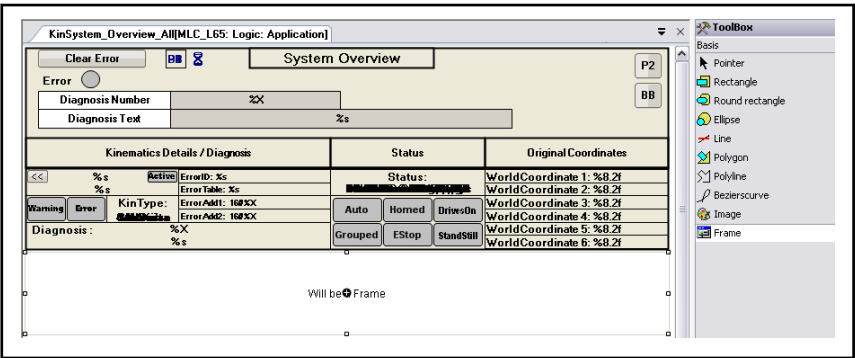


図 7-26: プロジェクトへのキネマティクスの追加

4.

5.
- フレームを右クリックし、"Frame selection" (フレーム選択) を有効にします。表示選択ウィンドウで **KinSystem_Overview** 要素を選択します。完全に新しいキネマティクスのシステム概要が表示されます。

新たに挿入したフレームのプロパティ ウィンドウを開き、**Referenced visualizations** ▶ **Referenced visualizations** ▶ **KinSystem_Overview** ▶ **m_Input_KinRef** の項目でキネマティクスの名前を入力します。同様の方法で 1 つ目のキネマティクスの参照を適合させます。

これで、作成したキネマティクスでシステム概要が利用できます。

オプション 2：既存のキネマティクス表示の複製

1.

2.

3.
- 各 IndraWorks プロジェクトを開き、タブ **Logic** ▶ **Application** ▶ **AxisInterface** ▶ **Plugins** ▶ **KinInterface** ▶ **Visualizations** ▶ **SystemOverview** を選択します。

IndraLogic のオフライン モードで、表示 "KinSystem_Overview_All" をダブルクリックします。

既存のキネマティクス表示をコピーし、必要に応じて何回でも挿入します。

キネマティクス インターフェース

4. 新たに挿入したキネマティクス表示のプロパティ ウィンドウを開き、**Referenced visualizations** ▶ **Referenced visualizations** ▶ **KinSystem_Overview** ▶ **m_Input_KinRef** の項目でキネマティクスの名前を入力します。同様の方法で1つ目のキネマティクスの参照を適合させます。

これで、指定したキネマティクスでシステム概要が利用できます。



プロジェクトに追加するすべてのキネマティクスについて、上述のオプションのいずれかを繰り返す必要があります。

ナビゲーション システムの概要

キネマティクスの概要から個別のキネマティクスの表示 "Kinematics_Overview" に移動することが可能です。それには、次のボタンを使います：<<<。このボタンは列 "Kinematics Details / Diagnosis" の左側にあります。他のすべての表示は、個別のキネマティクスの表示からアクセスできます。

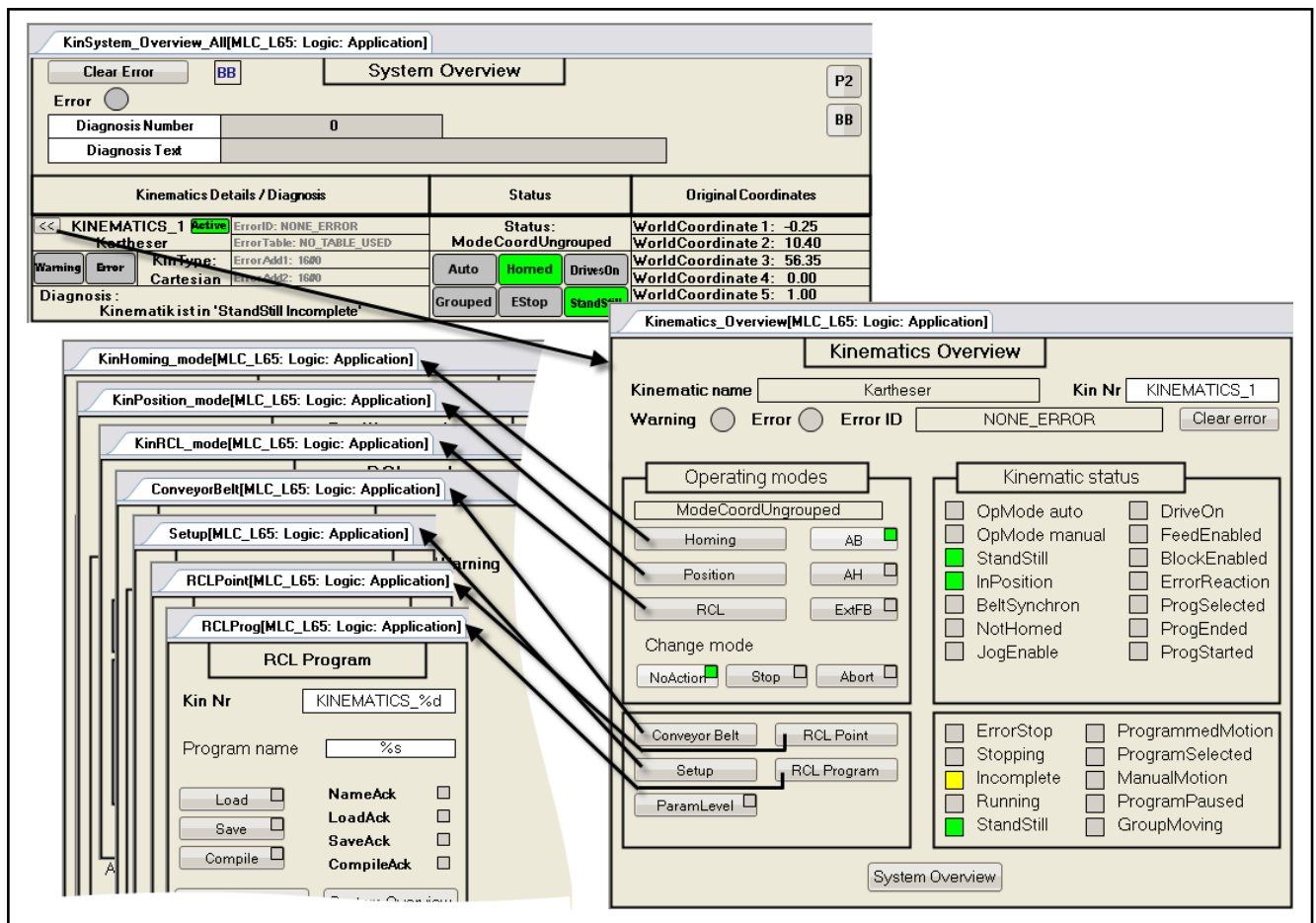


図 7-27: ナビゲーション システムの概要



フィールド Kin Nr にキネマティクス番号を入力することにより、すべての表示でさまざまなキネマティクス間の切替ができます。

キネマティクスの概要表示

Kinematics_Overview 表示は、キネマティクス インターフェースの最も重要な機能と KinData インターフェース診断を明確に表示します。

キネマティクス インターフェース

図 7-28: キネマティクスの概要表示

Homing、**Position** および **RCL** のボタンを使用して、各運転モードの表示を変更します。この表示では、各運転モードの特別な値を入力できます。各運転モードに直接切り替えるには、**AB**、**AH** および **ExtFB** のボタンを使用します。運転モードの変更後の反応を定義するには、**NoAction** および **Stop** のボタンを使用します。

Conveyor Belt (コンベヤー ベルト)、**Setup** (セットアップ)、**RCL Point** (RCL ポイント) および **RCL Program** (RCL プログラム) のボタンは、ベルト設定、キネマティクスのジョグ、RCL ポイント編集および RCL プログラム処理のために "Utilities" (ユーティリティ) に保存されている各表示を開くために使用します。メッセージ "SetupMode ACTIVE" は、ジョグ モードが有効であることを示します。

表示の右側の部分には、KinData インターフェースの診断データおよびキネマティクスの現在の PLCOpen 状態が表示されます。



使用されている Robot Control バージョンにより、特定の表示要素は非表示になります。このため、各 Robot Control バージョンで利用可能なキネマティクス インターフェース機能のみが選択できます。

参照の概要 (Homing) 表示

キネマティクス軸を参照するには、KinHoming_mode 表示を使用します。

キネマティクス インターフェース

図 7-29: 参照の概要表示

個別の軸の順序はランダムに選択できます。デフォルトにより、キネマティクスに割り当てられたすべての軸は、その設定順序に従って入力されます。軸番号 0 は、ホーミングシーケンスの中止条件です。

表示の右側の部分には、すべての指定された軸およびキネマティクスの参照ステータスが表示されます。軸またはキネマティクスが参照されている場合、表示要素は緑色に変わります。



"Kinematics homed" の表示要素は、すべての軸がすでに参照されていても、ホーミングシーケンスがエラーなしで処理された場合のみ緑色に変わります。

位置設定の概要 (Position) 表示

表示 KinPosition_mode キネマティクスの位置を設定します。

キネマティクス インターフェース

KinPosition_mode[MLC_L65: Logic: Application]

Position mode

Kinematic name Kin Nr

Warning ☐ Error ☐ Error ID

Command values

Point

[1]	%s	[2]	%s	[3]	%s
[4]	%s	[5]	%s	[6]	%s
[7]	%s	[8]	%s	[9]	%s
[10]	%s	[11]	%s	[12]	%s
[13]	%s	[14]	%s	[15]	%s
[16]	%s				

Velocity

Acceleration

Deceleration

Jerk

BlendRadius

AuxPoint

[1]	%s	[2]	%s	[3]	%s
[4]	%s	[5]	%s	[6]	%s
[7]	%s	[8]	%s	[9]	%s
[10]	%s	[11]	%s	[12]	%s
[13]	%s	[14]	%s	[15]	%s
[16]	%s				

Actual values

OrigCoord 1

OrigCoord 2

OrigCoord 3

OrigCoord 4

OrigCoord 5

OrigCoord 6

OrigCoord 7

OrigCoord 8

OrigCoord 9

OrigCoord 10

OrigCoord 11

OrigCoord 12

OrigCoord 13

OrigCoord 14

OrigCoord 15

OrigCoord 16

Velocity

Acceleration

Jerk

CoordSystem

SlopeType

SetMode

☐ InBuffer ☐ Stopping

☐ CmdDone ☐ ErrorStop

☐ InPosition ☐ GroupMoving

☐ BeltSynchron

図 7-30: 位置設定の概要表示

以下を入力または選択できます

- 位置
- 速度
- Acceleration
- Deceleration
- Jerk
- ブレンド半径
- 中間点
- 座標系
- モーションのスロープ タイプ
- ブレンドタイプ

各位置設定モードを選択するには、**DirectAbs**、**LinearAbs** などのボタンを使用します。詳細については、運転モードの説明を参照してください。

キネマティクス インターフェース

表示の右側の部分には、世界座標としての現在の位置、パラメータ化された境界条件、キネマティクス インターフェースと KinData インターフェースの重要なステータス要素、および現在の運転モードが表示されます。



使用されている Robot Control バージョンにより、特定の表示要素は非表示になります。このため、各 Robot Control バージョンで利用可能なキネマティクス インターフェース機能のみが選択できます。

Robot Control 言語 (RCL) 表示

RCL プログラムを選択して起動するには、表示 **KinRCL_mode** を使用します。

図 7-31: Robot Control 言語の表示

フィールド "ProgName" で RCL プログラム名を指定し、**Start** (開始) ボタンを使用してプログラムを起動します。RCL プログラムを停止するには、**AB** または **AH** ボタンを使用して運転モードを変更します。

表示の右側の部分には、キネマティクス インターフェースと KinData インターフェースの重要なステータス要素のほか、現在の運転モードが表示されます。



ファイル名の拡張子 (*.ird) を記述**せず**に、フィールド "ProgName" で RCL プログラム名を指定します。



Robot Control V1 の場合のみ許可されます。

ベルト設定 (ConveyorBelt) 表示

キネマティクスに属するベルトをすべて設定するには、**ConveyorBelt** 表示を使用します。

キネマティクス インターフェース

図 7-32: ベルト設定の表示

ベルト設定とベルト エラー反応を指定し、各ベルトをパラメータ化します。ベルトの数に応じてベルト設定概要を倍増させます。これは、システム概要へのキネマティクスの追加と同様の方法で行います (185 ページ "システム概要表示" 章 を参照)。

参照される表示として連続したベルト番号を指定します。このベルト番号を介して、キネマティクス インターフェース 構造 `arKinCtrl_gb[].Admin.ConveyorBelt` が内部的にアクセスされます。



ベルトはファンクションブロック "ML_SetBeltParameter" (Robot Control V1) または "ML_SetBeltConfiguration" (Robot Control V2) により内部的に設定されます。詳細については、各資料を参照してください。



使用されている Robot Control バージョンにより、特定の表示要素は非表示になります。このため、各 Robot Control バージョンで利用可能なキネマティクス インターフェース機能のみが選択できます。

ジョグ/セットアップ モード表示

キネマティクスは **Setup** (セットアップ) 表示によりジョグできます。

キネマティクス インターフェース

図 7-33: ジョグ/セットアップ操作表示

ジョグが行われる座標系を、"CoordSystem" にあるボタンで定義します。同様の方法で、ステップ幅または横断速度を "JogMode" で指定します。1+ から 6- のボタンを使用し、キネマティクスの最初の 6 本の軸を設定に従って正および負の方向に動かします。



ベルト軸は軸グループに属さないため、ジョグできません。



使用されている Robot Control バージョンにより、特定の表示要素は非表示になります。このため、各 Robot Control バージョンで利用可能なキネマティクス インターフェース機能のみが選択できます。

Robot Control 言語ポイント (RCLPoint) 表示

RCL ポイントの読み取りまたは書き込みおよび/または RCL ポイント リストまたはプログラム リストの読み取りを行うには、RCLPoint 表示を使用します。

キネマティクス インターフェース

図 7-34: Robot Control 言語ポイントの表示

この表示で指定する値を使用して、ポイント (**GetPoint**)、ポイント リスト (**GetPointList**) またはプログラム リスト (**GetProgList**) を読み出すことができます。ポイントを設定するためには (**SetPoint**)、ユーザー プログラムでキネマティクス インターフェース 構造要素 `arKinCtrl_gb[].Admin.RCLPoint.Point` を設定します。



ファイル名の拡張子 (*.pkt) を用いて、フィールド "FileName" でファイル名を指定する必要があります。



Robot Control V1 の場合のみ許可されます。

Robot Control 言語プログラム (RCLProg) 表示

RCLProg 表示を介して **RCL** ファイルをコピーできます。

- Compact Flash カードからコントロールの内蔵メモリに
 - コントロールの内蔵メモリから Compact Flash カードに
- または

RCLProg 表示を介して **RCL** ファイルをコンパイルできます。

- コントロール上

キネマティクス インターフェース

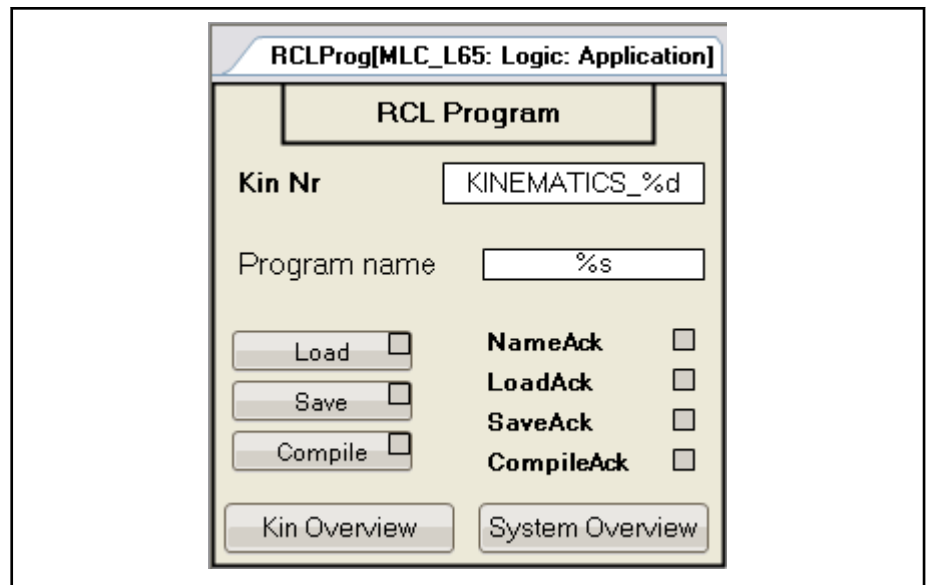


図 7-35: Robot Control 言語プログラムの表示



ファイル名の拡張子 (*.qll など) を用いて、フィールド "Program name" でファイル名を指定する必要があります。



Robot Control V1 の場合のみ許可されます。

7.2.7 キネマティクス インターフェースのユーザー拡張

概要

arKinCtrl_gb[] および arKinStatus_gb[] 構造を拡張し、キネマティクス インターフェースを特別なアプリケーションに適合させることができます。

このセクションでは、新しい機能を追加することによりキネマティクス インターフェースを拡張する方法について説明します。ここでは、キネマティクス インターフェースで準備された拡張構造が使用されることを前提とします。以下では、POU および構造で必要となる変更についてのみ説明しています。

ユーザー拡張の実装に関する注意

ユーザー拡張は IndraLogic 2G からのオブジェクト指向の拡張を使用して実装されました。

以下の特殊性について留意してください。

- ファンクションブロック **TE_KinematicsInterface** は基本ファンクションブロック "MB_KinInterfaceType01" に由来します。基本ファンクションブロックまたはそのメソッド/アクションは、キーワード "SUPER" を介して呼び出すことができます。複数の場所で、MB_KinInterfaceType01 ファンクションブロックの mSetError メソッドが呼び出され、例えば SUPER^.mSetError(...); を介して診断システムにエラーが入力されます。
- ファンクションブロック **TE_KinematicsInterface** 内で、arKinCtrl_gb のデータは入力 AdminCtrlExt、CoordModeCtrlExt などを使用してアクセスでき、arAxisStatus_gb のデータは AdminStatusExt、DiagStatusExt などを使用してアクセスできます。

キネマティクス インターフェース



入力 AdminCtrl (*Ext なし) などは基本ファンクションブロックに属し、ユーザー拡張は利用不可能なため、使用しません。

- ファンクションブロック TE_KinematicsInterface の入力は "REFERENCE TO" として定義されます。 "mInitExtension" メソッドで、参照は 1 回初期化され、ファンクションブロックが周期的に呼び出されたときに再び転送する必要はありません。

キネマティクス インターフェース構造の拡張

追加的な下位構造により、構造 TE_KINEMATICS_CONTROL_TYPE01 および TE_KINEMATICS_STATUS_TYPE01 を拡張するには、下記の手順に加えて以下のステップが必要です。

- TE_KINEMATICS_CONTROL_TYPE01 および / または TE_KINEMATICS_STATUS_TYPE01 で新しい要素を入力する
- ファンクションブロック TE_KinematicsInterface に追加的な入力を REFERENCE TO として追加する
- ファンクションブロック TE_KinematicsInterface のメソッド "mInitExtension" の参照を初期化する



下位構造 TE_KINEMATICS_ADMINISTRATION、TE_KINEMATICS_COORDINATEDなどを拡張するには、各構造について下記の手順のステップ 1 のみが必要です。この構造で、拡張を下に追加する必要があります (* 拡張をこの下に追加します | Erweiterungen hier einfuegen*). すると、新たに作成した入力または出力が、それぞれの *Ext 構造要素を介して利用可能になります。

ファンクションブロック TE_KinematicsInterface の拡張

ユーザー拡張手順の最後のステップとして、ユーザー拡張のプログラムコードが挿入できるような方法で TE_KinematicsInterface ファンクションブロックを拡張します。

- プロジェクト ツリーで、右クリックして TE_KinematicsInterface および Add ► Action を選択します。
- 拡張と一致する新しいアクションの名前を指定します。
- 新しいアクションにプログラムコードを挿入します。

サフィックス *Ext のある入力 (AdminCtrlExt、DiagStatusExt など) を使用して構造 arKinCtrl_gb[] および arKinStatus_gb[] にアクセスします。

- TE_KinematicsInterface ファンクションブロックで新しいアクションを呼び出します。
- プロジェクトを再度コンパイルし、プログラムエラーがないかチェックします。
- コントロールにプロジェクトをロードします。

これで、新しい入力および出力要素がキネマティクス インターフェース構造の一部となり、UserVarGlobal 変数を使用して表示できます。

7.3 ユース ケース

7.3.1 キネマティクス インターフェースによる逐次プログラミング

特定の境界条件を順守した場合、キネマティクス インターフェースによりモーションシーケンスを実装できます。これを行う方法について、例に基づいて以下に説明します。

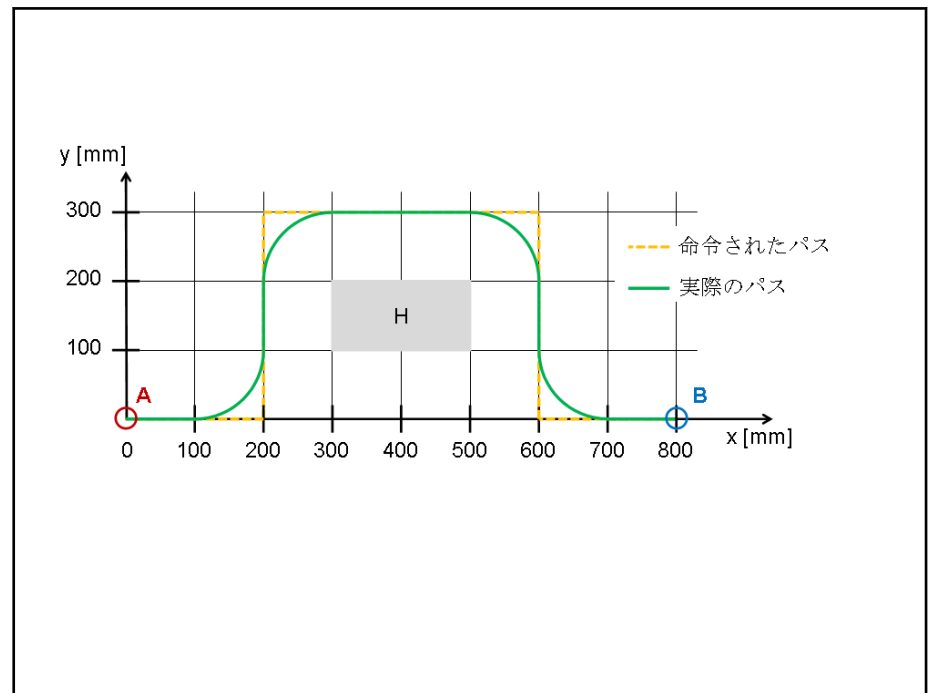


図 7-36: キネマティクス インターフェースによる逐次プログラミング - モーションパス

キネマティクス (この場合はデカルト) は、障害 H の周囲をポイント A からポイント B に移動します。この目的のために、上の図に示されているパスを指定します。

結果として得られる位置は以下のとおりです。

- A(0,0)
- P1(200,0)
- P2(200,300)
- P3(600,300)
- P4(600,0)
- B(800,0)

線形の絶対ブレンドモーションによりこれらの位置を移動します。キネマティクス インターフェースによりこれを実装するには、以下のプログラム ラインが必要です。速度、加速、減速およびジャークの制限値は任意に選択されました。

宣言

```
VAR
  uiState:          UINT := STATE_IDLE;
  bStartMove:       BOOL := FALSE;
  rVelocity:        REAL := 0.0;
  rAcceleration:    REAL := 0.0;
  rDeceleration:    REAL := 0.0;
  rJerk:            REAL := 0.0;
  arPoint1:         ARRAY[1..16] OF REAL := [200.0, 0.0, 14(0.0)];
  arPoint2:         ARRAY[1..16] OF REAL := [200.0, 300.0, 14(0.0)];
  arPoint3:         ARRAY[1..16] OF REAL := [600.0, 300.0, 14(0.0)];
```

キネマティクス インターフェース

```

arPoint4:    ARRAY[1..16] OF REAL := [600.0, 0.0, 14(0.0)];
arPointB:    ARRAY[1..16] OF REAL := [800.0, 0.0, 14(0.0)];
END_VAR

VAR CONSTANT
STATE_IDLE:  UINT := 1;
STATE_MOVE1: UINT := 2;
STATE_MOVE2: UINT := 3;
STATE_MOVE3: UINT := 4;
STATE_MOVE4: UINT := 5;
STATE_MOVE5: UINT := 6;
END_VAR

```

プログラム部分

```

CASE uiState OF
STATE_IDLE:
IF bStartMove THEN
bStartMove := FALSE;
uiState := STATE_MOVE1;
END_IF

STATE_MOVE1:
(* 運転モード変更後の動作を指定します。ブレンドモーションの場合は
必ず "NoAction" に設定する必要があります。 *)
arKinCtrl_gb[1].Admin.ChangeOpMode := NoAction;

(* arPoint1 = 位置 P1 を指定します *)
arKinCtrl_gb[1].CoordMode.Point := arPoint1;

(* 制限値を指定します *)
arKinCtrl_gb[1].CoordMode.Velocity := 100;
arKinCtrl_gb[1].CoordMode.Acceleration := 1000;
arKinCtrl_gb[1].CoordMode.Deceleration := 1000;
arKinCtrl_gb[1].CoordMode.Jerk := 10000;

(* 座標システムを指定します *)
arKinCtrl_gb[1].CoordMode.CoordSystem := CS_MACHINE_COORDINATES;

(* SetMode を指定します *)
arKinCtrl_gb[1].CoordMode.SetMode := SM_VIA;

(* SlopeType を指定します *)
arKinCtrl_gb[1].CoordMode.SlopeType := ST_CONTINUOUS_SLOPE;

(* ブレンド半径を指定します *)
arKinCtrl_gb[1].CoordMode.BlendingRadius := 100;

(* Send linear traversing command;
Block is set in block buffer *)
arKinCtrl_gb[1].Admin._OpMode.en := ModeCoordPosLinAbs;

(* 次の手順の前に: "InBuffer" 出力をクエリします *)
IF arKinStatus_gb[1].Admin.InBuffer THEN
uiState := STATE_MOVE2;
END_IF

STATE_MOVE2:
(* arPoint2 = 位置 P2 を指定します *)
arKinCtrl_gb[1].CoordMode.Point := arPoint2;

(* only input structures have to be described which
have changed *)
arKinCtrl_gb[1].CoordMode.Velocity := 20;

(* a linear traversing command does not have to be send
as this is automatically executed by changing
the position value *)

(* 次の手順の前に: "InBuffer" 出力をクエリします *)
IF arKinStatus_gb[1].Admin.InBuffer THEN
uiState := STATE_MOVE3;
END_IF

STATE_MOVE3:
(* arPoint3 = 位置 P3 を指定します *)
arKinCtrl_gb[1].CoordMode.Point := arPoint3;

(* 次の手順の前に: "InBuffer" 出力をクエリします *)
IF arKinStatus_gb[1].Admin.InBuffer THEN
uiState := STATE_MOVE4;
END_IF

```

キネマティクス インターフェース

```

STATE_MOVE4:
(* arPoint4 = 位置 P4 を指定します *)
  arKinCtrl_gb[1].CoordMode.Point := arPoint4;

(* 次の手順の前に: "InBuffer" 出力をクエリします *)
  IF arKinStatus_gb[1].Admin.InBuffer THEN
    uiState := STATE_MOVE5;
  END_IF

STATE_MOVE5:
(* arPointB = 位置 B を指定します *)
  arKinCtrl_gb[1].CoordMode.Point := arPointB;

(* 終了 *)
  uiState := STATE_IDLE;

END_CASE

```

ブロックの準備、すなわち SFC の処理は、モーションの実行より大幅に高速です。このため、キネマティクスがまだ動いている間に、SFC は STATE_IDLE になることがあります。



キネマティクス インターフェースのステータス出力は、必ず最後に発行されたモーションコマンドを参照します。この例では、これはポイント B に向かうモーションです。従って、前のモーションのステータスを評価する方法はありません。

7.3.2 ユーザー定義変換の登録

ユーザー定義変換を使用する場合、すなわち変換番号 1100 の場合、フェーズを BB に切り替える前に PLC で変換を登録する必要があります。登録はキネマティクス インターフェースを介して簡単に行うことができます。この目的のために、変換ファンクションブロックはキネマティクス インターフェースの "Admin.KinTransform" 入力に接続する必要があります。以下のコマンドは、変換登録を起動します ("fbTransformation" は変換ファンクションブロック インスタンスに対応)。

```

arKinCtrl_gb[1].Admin.KinTransformPLC := fbTransformation;

```



登録は、パラメータ化モード P2 でのみ可能です。変換が別のフェーズでキネマティクス インターフェース入力に接続された場合、登録はフェーズ 2 に達したときのみを開始されます。



キネマティクス インターフェースが初回にユーザー定義の変換で登録された場合、コントロールはエラー メッセージ "Transformation not registered" (変換が登録されていません) を返します。これは、IMC インターフェースが運転モードへの切替を試みているためです。変換が登録されると、エラーを確認できます。

ステータス出力 arKinStatus_gb[1].Admin.KinTransformPLCDone は、PLC での変換登録が正常に行われたことを示します。

8 GATcompact

8.1 GATcompact の概要

一般情報	compact 汎用アプリケーション テンプレートは、IEC 61131-3 に基づいたサンプル プロジェクトです。IndraLogic XLC および IndraMotion MLC 用 IndraWorks エンジニアリング フレームワークによって利用が可能で、ウィザードを使用してロードと起動を行うことができます。また、軸の処理など、モーションアプリケーションの特定の機能もサポートしています。
基本概念	GATcompact の使用は、簡素な機械は "Auto" (自動) と "Manual" (手動) の運転モードを備えている必要があるという考え方に基づいています。また、軸の処理は例に基づいて示されています。
使用の要件	ユーザーは、実装される機械のどのアクティビティが Automatic (自動) モードで実行され、どのアクティビティが Manual (手動) モードで実行されるかを把握する必要があります。

8.2 最初の手順 - GATcompact

8.2.1 単純なモーションアプリケーションの作成

準備

以下の説明に従って、GAT ウィザードを使用して 2 軸のモーションロジック アプリケーションを数分間で作成します。

これを行うために、IndraLogic XLC または IndraMotion MLC コントロールが必要ですが、実在のドライブは不要です。"GAT compact" のサンプル プロジェクトをベースとして使用します。

トピック

- 機械の自動モードでは、"マスター" 軸が一定の速度で動き続けます。速度は変数で変更可能です。
- 2 番目の "スレーブ" 軸は、1 本目の軸とフェーズ同期して動作します。
- 下の図は、タスクを図式によって示したものです。

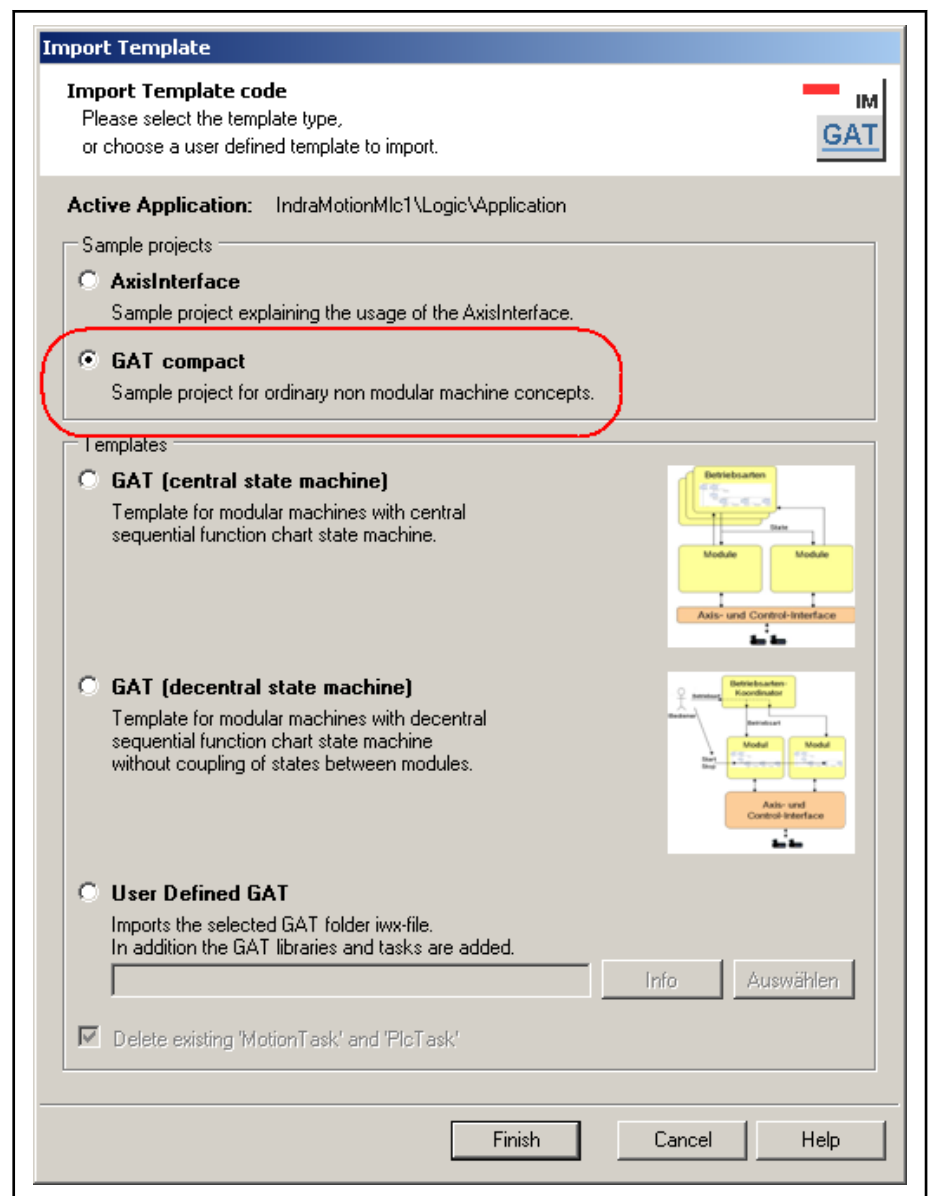


図 8-2: GAT ウィザードの起動ダイアログ

GATcompact を選択することにより、実行可能で予め定義済みのプロジェクトがインポートされます。このプロジェクトにはすでに 2 本の仮想軸が含まれています。

"AUTO" (自動) 運転モードは事前設定に含まれているため、運転モードの定義に関しては、タスクを実現するために他の設定を選択する必要はありません。当初のインポートにより、"AUTO" (自動) 運転モードのコード例はすでに以下のアクションで利用可能です。

- アクション "actModeAuto_entry" は軸を始動します。このアクションは、"AUTO" (自動) 運転モードが有効になると、初めて実行されます。コード例で、速度モード (ModeVel) は "MyAxis1" 軸について設定されています。さらに、"MyAxis2" 軸はフェーズ同期 (ModeSyncPhase) に切り替えられます。
- アクション "actModeAuto_active" は、運転モード "AUTO" (自動) が有効な場合に周期的かつ恒久的に処理されます。軸のコマンド速度と表示される速度はコピーされます。

GATcompact

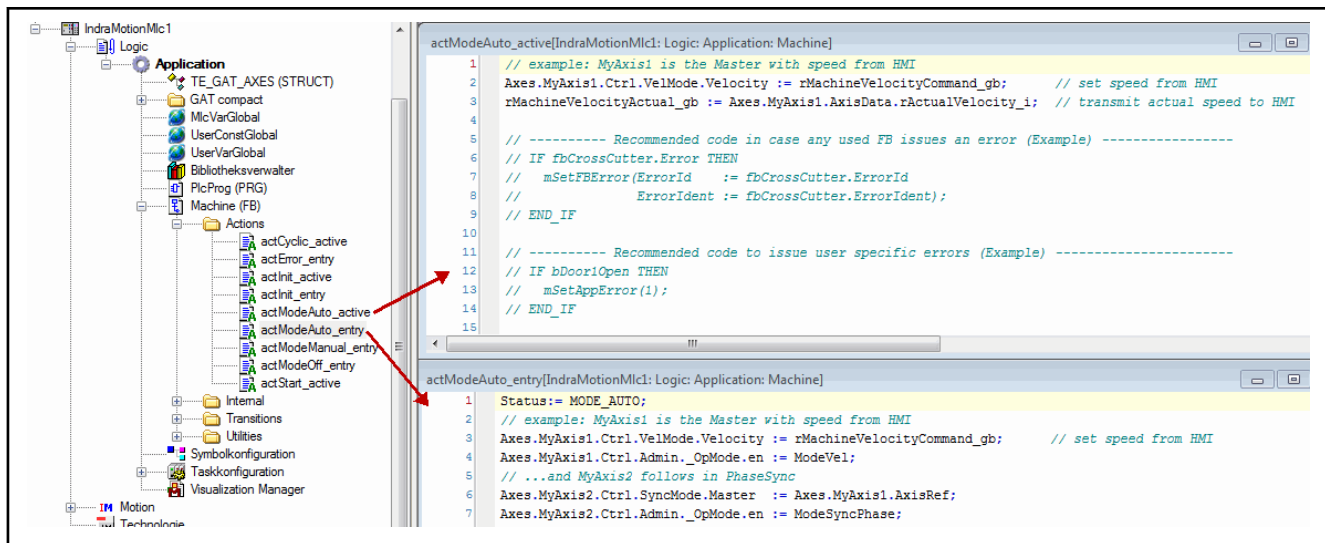


図 8-3: 軸の制御

調整を行うには、以下の手順を実行します。

- 軸 "MyAxis1" の名前を "Master" に変更します。これを行うには、以下のとおり GAT ウィザードを使用し、**Apply** (適用) をクリックします。

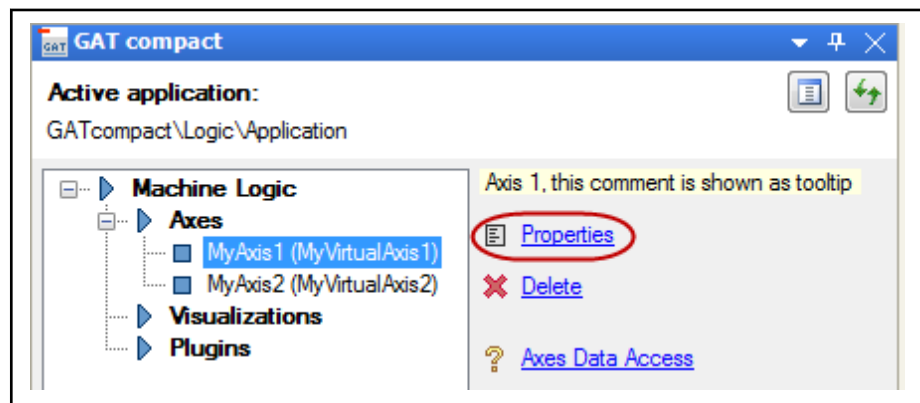


図 8-4: "Properties" (プロパティ) を経由した軸の名前の変更

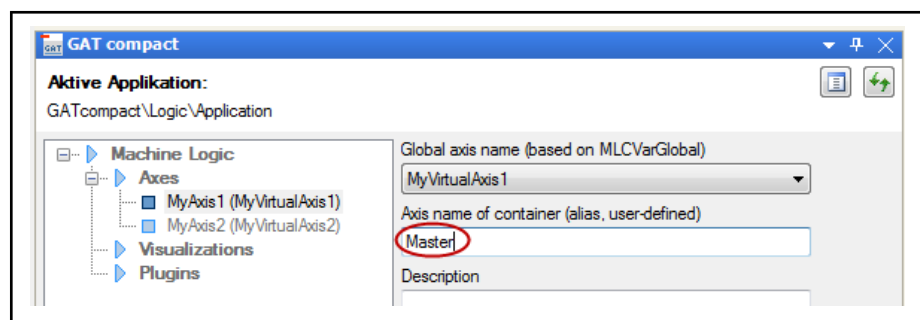


図 8-5: 軸の名前変更

- 同様にして、2 番目の GATcompact 軸の名前を "MyAxis2" から "Slave" に変更します。下の図は、調整を実行した後の表示を示したものです。

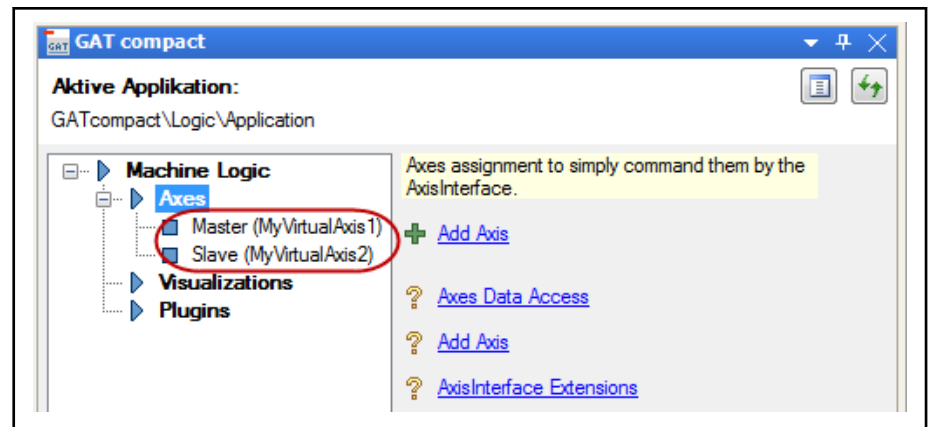


図 8-6: 調整後の軸の概要

- しかし、GATcompact ウィザードでは、GATcompact 軸の宣言に関するコード変更しか行うことができません。このため、アクション "actAutoMode_entry" および "actAutoMode_active" で変更した軸の名前を用いてプログラムコードを調整します。これは、"検索・置換" を用いて行うことができ、また、軸の名前をアクションに直ちに変更する場合は、IndraLogic Intellisense 機能によってアクションがサポートされます。これを行うには、識別子 "Axes" の後のドットを削除し、挿入します。このアクションを行うと、"Axes" 構造のすべての要素が表示されます。この方法により、"Slave" 軸が強調表示されて <Enter> が押されると、"MyAxis2" 識別子が "Slave" で置き換えられます (下の図を参照)。

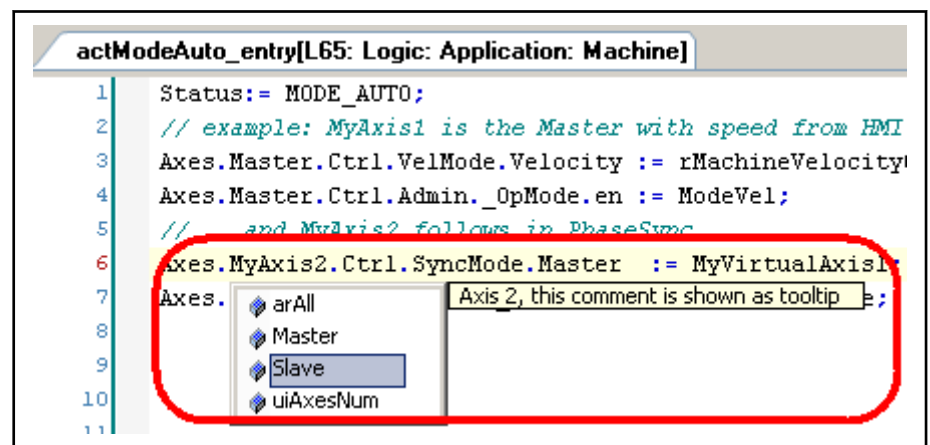


図 8-7: Intellisense 機能

- 調整の結果 :

GATcompact

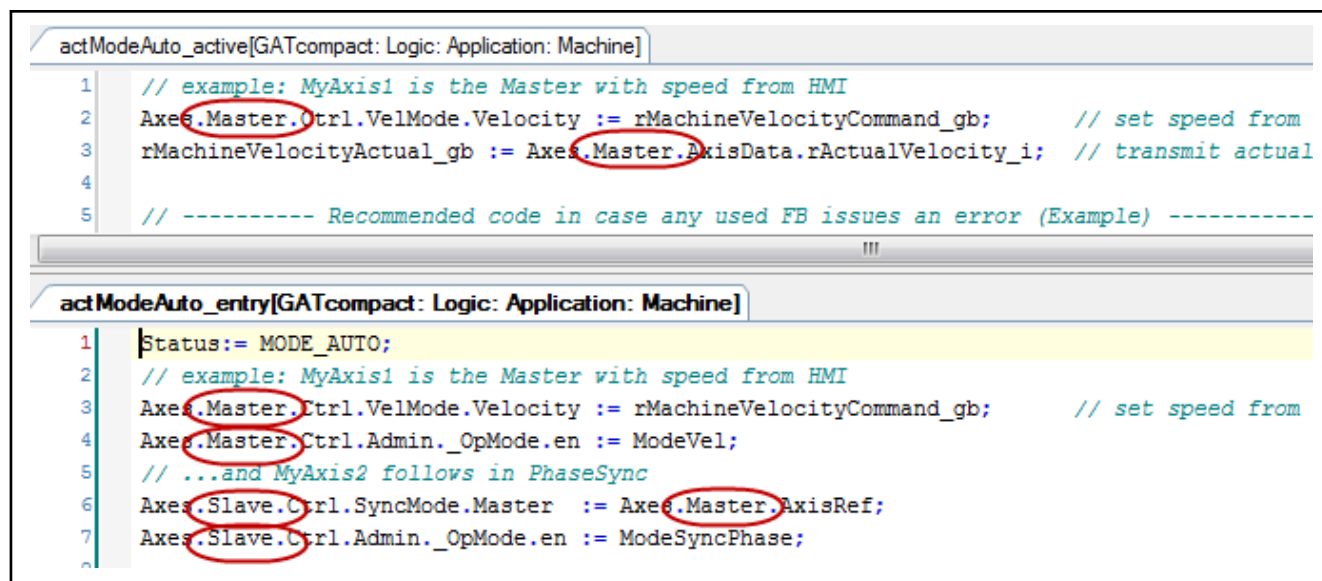


図 8-8: 軸制御の実行例

試運転表示の調整



最適な軸コマンドのために、ダイアログベースの軸コマンドも利用可能です。このトピックに関する詳細は、67 ページ "ダイアログベースの軸コマンド" 6.4.1 章を参照してください。

新しい軸の名前は、試運転表示で以下のとおり調整する必要があります。

- GAT ウィザードを介して試運転表示を開きます。これを行うには、GAT ウィザード ツリーでノード "Visualization" (表示) を選択し、"Open visualization VI Logic" (VI Logic 表示を開く) をクリックします。

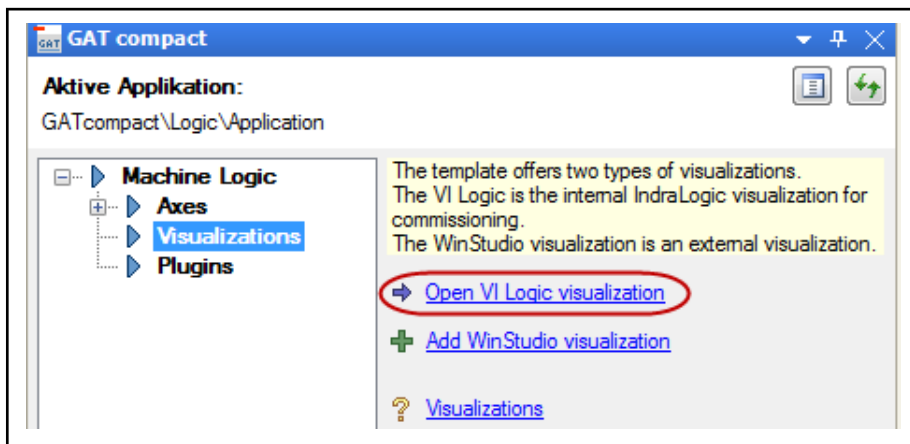


図 8-9: GAT ウィザードを介して表示を開く

- 下の図で強調表示されている表示オブジェクトをクリックし、入力フィールド "Referenced visualizations" で軸の名前を以下のとおり調整します。調整後、この軸表示は "Master" 軸に反映されます。

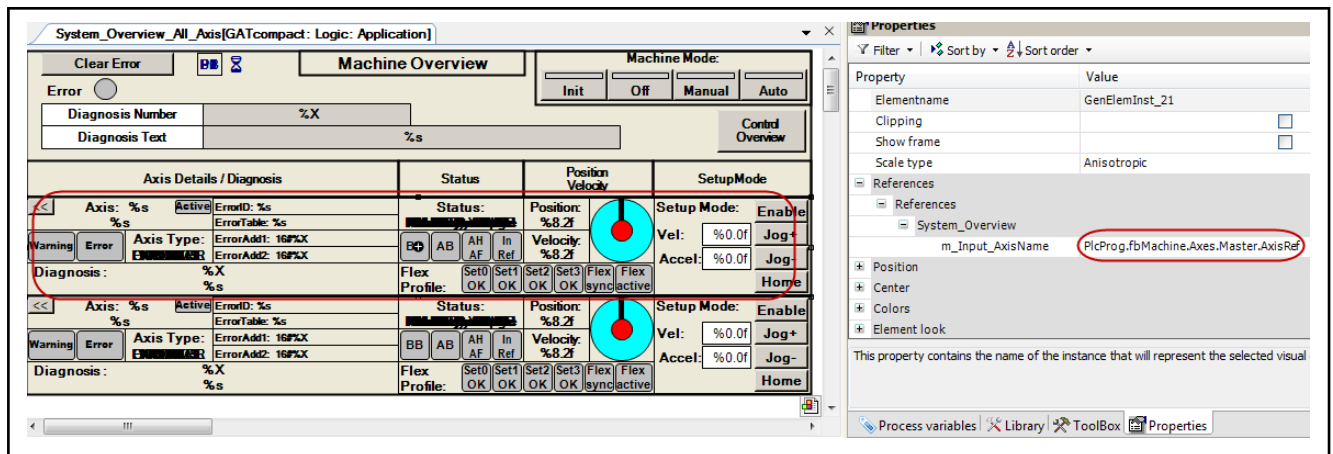


図 8-10: MasterAxis の軸表示の調整

入力フィールドに以下のテキストを入力します。

PlcProg.fbMachine.Axes.Master.AxisRef

- 下記の表示オブジェクトを、"Slave" 軸についても同様に調整します。

Slave 軸の入力フィールドに以下のテキストを入力します。

PlcProg.fbMachine.Axes.Slave.AxisRef

これによりプログラミングが完了し、プロジェクトのロードと起動を行うことができます。

PLC プロジェクトの設定、ロードと起動

すべての変更が正しく行われた場合、プロジェクトはエラーなくコンパイルされ、ロードされます。

設定 (1) と PLC プロジェクト (2) をダウンロードした後で、PLC を起動できます (3) (下の図を参照)。



図 8-11: PLC のダウンロード (設定/PLC オブジェクト) と起動

GAT ウィザード ツリーの "visualization" (表示) ノードで "Visualization VI Logic" (VI Logic 表示) ボタンを押して、試運転表示を開きます。

GATcompact

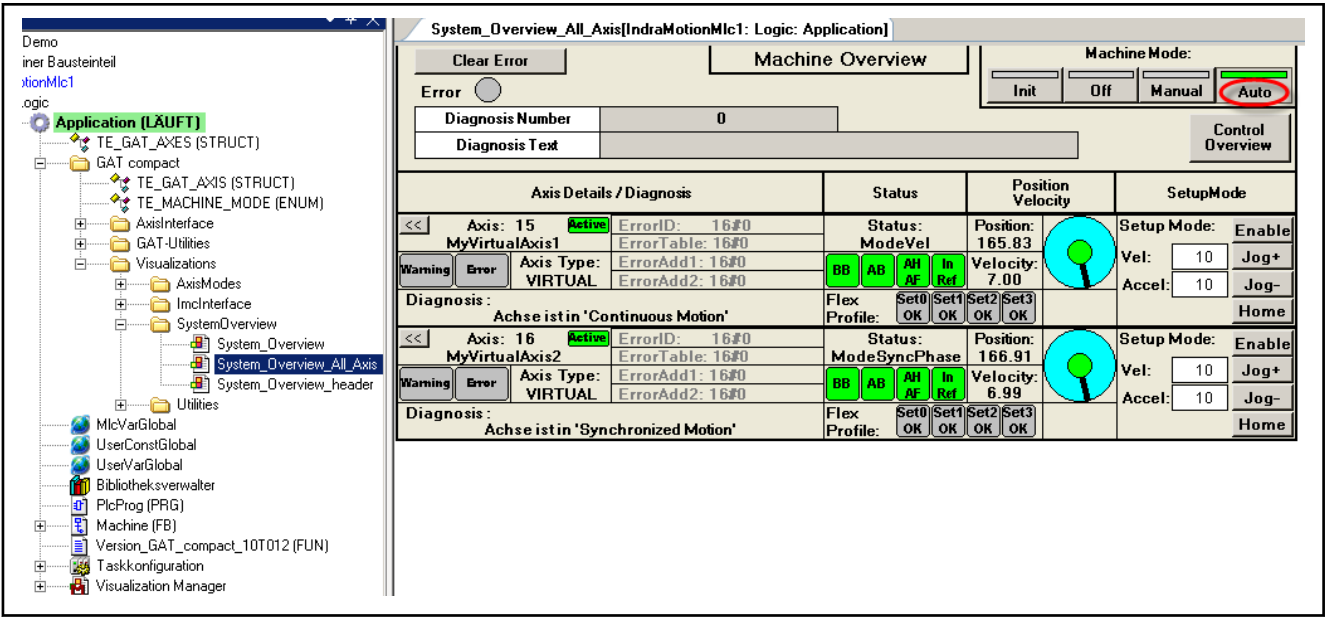


図 8-12: 試運転表示

Auto (自動) を用いて自動運転モードを選択します。 **Auto** (自動) 運転モードでは、GATcompact の軸コンテナで "Master" 軸が 7 rpm (7 rpm は変数 "rMachineVelocityCommand_gb" のデフォルト値) で連続的に回転します。それに従って、GATcompact 軸コンテナ内の "Slave" 軸がフェーズ同期します。手動モードに切り替えて軸を停止させるには、**Manual** (手動) を押します。手動モードでは、それぞれの有効化ボタンでセットアップモードを有効にします。セットアップモードでは、ジョグ ボタンで軸をジョグし、ホーム ボタンでそれを参照します。

8.3 初期化

初期化は起動時に実行されます。初期化が正常に完了した後、機械は運転モードに入ります。

初期化の開始時には、"コールド スタート" (パラメータ化モード/P2) に従ってシステムに指示が行われます。初期化の第 2 手順では、"ウォーム スタート" (BB) に従ってシステムに指示が行われます。

機械がすでに運転モードに入っている場合、初期化を選択することにより、"コールド スタート" に従って機械に指示を行うことができます。これは、運転モードの動作モードが選択されるまで維持されます。

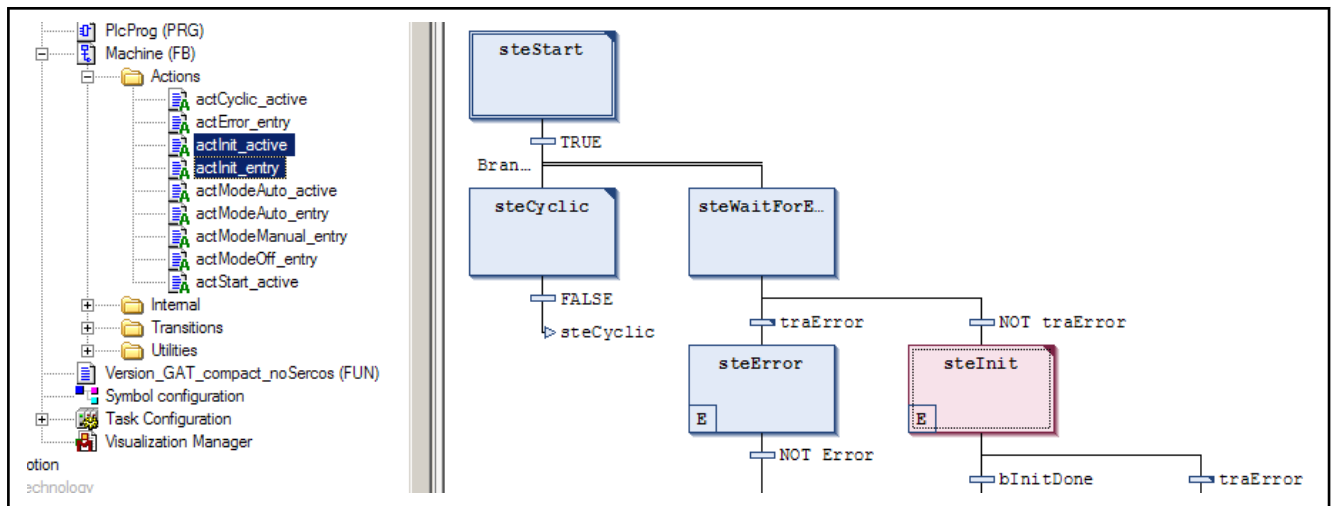


図 8-13: 機械 SFC の初期化手順

エラーが発生していない限り、"steInit" 手順で初期化が実行されます。この手順の開始時に入力アクション "steInit_entry" が 1 回処理されます。

初期化中にエラーが発生した場合、スイッチがエラー ステータスに変更されます。エラーをクリアした後、初期化は中断された位置から続行されます。

ビット "bInitDone" が設定されると、プロセスは初期化を終了し、次の手順に進みます。

GATcompact

```

actInit_active[IndraMotionMlc1: Logic: Application: Machine]
1  CASE uiStateInit OF
2      uiSTATE_INIT_PREPARE_COLDSTART:
3          bInitDone := FALSE; // Init is not done
4          bAllAxesOff := TRUE;
5
6          // switch all axes off to ModeAb
7  FOR uiLoopIndex := 1 TO uiMAX_AXIS_INDEX DO
8      // for all active axes, but only when added by the wizard (<>NO_OBJECT)
9      IF Axes.arAll[uiLoopIndex].Status.Admin.Active AND
10         Axes.arAll[uiLoopIndex].AxisRef.AxisNo <> NO_OBJECT AND
11         Axes.arAll[uiLoopIndex].Status.Admin._OpModeAck <> ModeAb
12  THEN
13      bAllAxesOff := FALSE; // at least one is under Power....
14  END_IF
15  END_FOR
16
17  IF (bAllAxesOff = TRUE)
18  THEN
19      uiStateInit := uiSTATE_INIT_SWITCH_TO_COLDSTART;
20  END_IF
21
22
23  uiSTATE_INIT_SWITCH_TO_COLDSTART:
24  ImcCtrl.Admin._OpMode := IMC_MODE_CTRL_P2; //Switch your MLC to P2 (Parameter mode)
25
26  IF (ImcStatus.Admin._OpModeAck = IMC_MODE_STATUS_P2)
27  THEN
28      uiStateInit := uiSTATE_INIT_COLDSTART;
29  END_IF
30
31
32  uiSTATE_INIT_COLDSTART:
33  IF (Ctrl <> MODE_INIT)
34  THEN
35      // *****
36      // Now we reached P2 (parameter mode)
37      // Add your code for writing the parameters (in P2) here
38      // *****
39      uiStateInit := uiSTATE_INIT_WARMSTART;
40  END_IF
41
42  uiSTATE_INIT_WARMSTART:
43  ImcCtrl.Admin._OpMode := IMC_MODE_CTRL_BB; //Switch your MLC to BB (Operation mode)
44
45  IF (ImcStatus.Admin._OpModeAck = IMC_MODE_STATUS_BB) AND
46     (TE_AxisInterfaceType01Prog.InitDone = TRUE)
47  THEN
48      // *****
49      // Now we reached BB (operating mode)
50      // Add your code for writing the parameters (in BB) here
51      // *****

```

図 8-14: "actInit_active" アクションでの初期化の実行

"actInit_active" 手順では以下が行われます。

1. はじめに、すべてのアクティブな軸の準備が整っており、トルク (Ab) がかかっていないことが確認されます (ライン 7~15)。これが確認でき

ないと、"コールド スタート" (P2) に従って機械に指示することはできません。

2. 機械は "コールド スタート" (P2) 状態になるよう命令されます (ライン 23~29)。
3. 機械が "コールド スタート" 状態になると、運転モードの運転モード (ModeOff、ModeAuto、ModeManual) が命令されるまで、その状態が維持されます。P2 のみで変更可能なパラメータをコントロールとドライブに書き込むことができるのは、この位置です (ライン 32~40)。
4. 機械は "ウォーム スタート" 状態になるよう命令されます (ライン 39)。P4 のみで変更可能なパラメータをコントロールとドライブに書き込むことができるのは、この位置です (ライン 42~51)。
5. 初期化が完了すると、"bInitDone" ビットが "TRUE" に設定され、プロセスは初期化を終了します。これで、機械は "Manual" (手動) または "Auto" (自動) 運転モードで運転する準備が整いました。

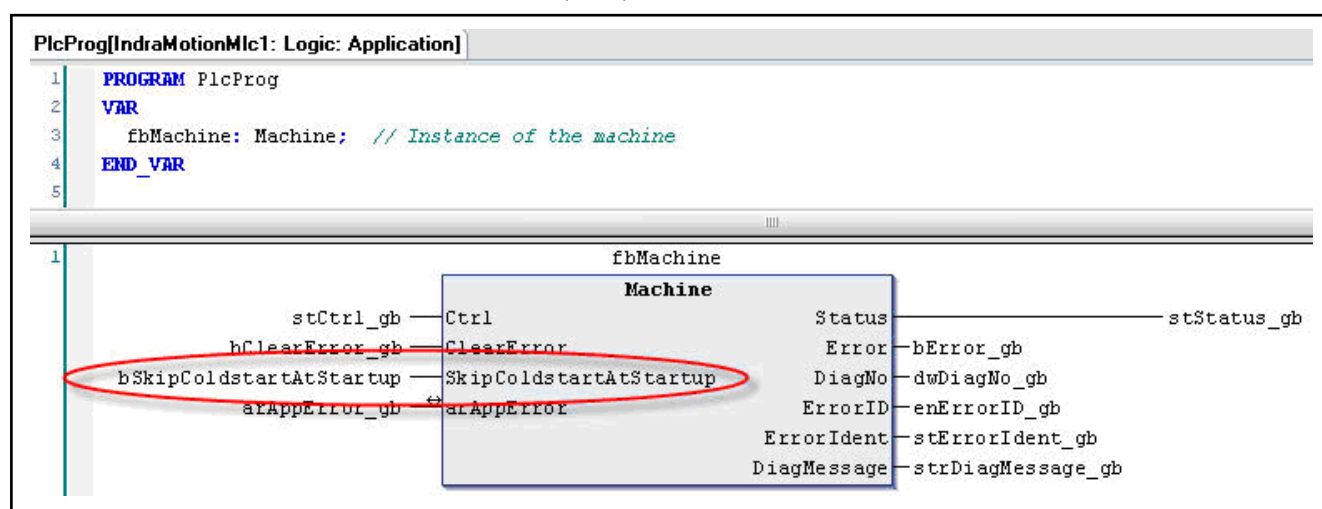


図 8-15: "Machine" FB での "SkipColdStartAtStartup" 入力

PLC を初めて起動したときに初期化が完了しなかった場合、この設定は "bSkipColdstartAtStartup" ビットを使用して行うことができます。プロセスは "ウォーム スタート" 状態に切り替わり、"コールド スタート" は処理されません。これにより、P2 への切替が排除され、BB に戻ります。この設定により、出力を上げる際に時間を節約できます。ただし、"コールド スタート" 状態で処理されるアクション (パラメータの書き込みなど) が不要であることを確認してください。

8.4 運転モード

初期化が正常に完了すると、シーケンスは以下の 3 つの運転モードのいずれかに入ります (下の図を参照)。

- ModeOff (すべてのドライブにトルクなし)
- ModeManual (機械のセットアップ、軸をジョグ可能)
- ModeAuto (周期的な自動運転)

GATcompact

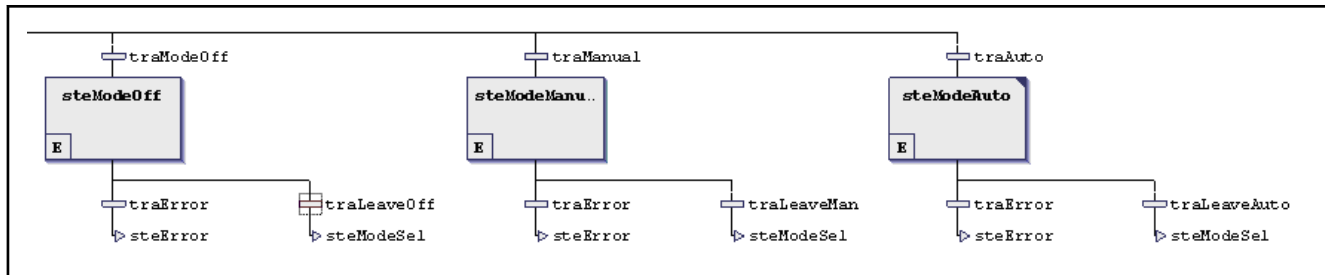


図 8-16: 運転モード

(208 ページ "初期化" 8.3 章 参照)

GATcompact では、初期の機械起動の後のターゲット運転モードとして "ModeManual" が予め設定されています。

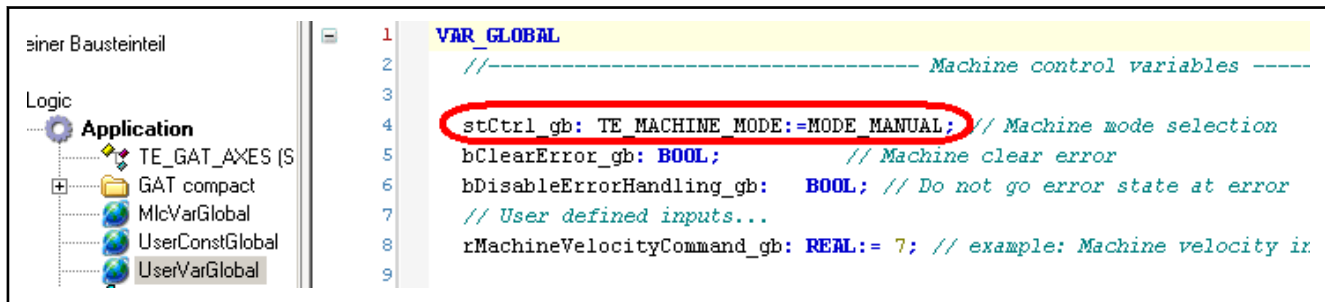


図 8-17: 初期の機械起動後のターゲット運転モードの設定

入力/アクティブ アクション

運転モードが有効になると、特別な"入力"アクションが 1 回のみ実行されます。下の表はこのアクションの概要をまとめたもので、各運転モードを有効化する方法および運転モードで行われる内容を示しています。

	起動の条件	アクション (当初、1 回のみ実行される)	周期的に実行されるアクション
ModeOff (ドライブへのトルクなし)	Ctrl=MODE_OFF クエリが traModeOff および traLeaveOff で実行される	actModeOff_entry - ステータスを "MODE_OFF" に設定 - すべてのアクティブな軸を "ModeAb" (軸へのトルクなし) に設定	予め定義済みのアクションなし
ModeManual (機械のセットアップ、軸のジョグ)	Ctrl=MODE_MANUAL クエリが traManual および traLeaveManual で実行される	actModeManual_entry - ステータスを "MODE_MANUAL" に設定 - すべてのアクティブな軸を "ModeAH" モード (軸にトルク負荷あり、v=0) に設定	予め定義済みのアクションなし 一般的なアクティビティ： 機械のセットアップおよび調整、例：選択した軸のジョグ
ModeAuto (周期的な自動運転)	Ctrl=MODE_AUTO クエリが traAuto および traLeaveAuto で実行される	actModeAuto_entry - ステータスを "MODE_AUTO" に設定 - 例の "MyVirtualAxis1" に対して軸速度モードに入るよう命令し、フェーズ オフセットなしで "MyVirtualAxis2" を同期させる	actModeAuto_active 例：プロセスコントローラの処理、生産プロセスのモニタリング (材料は使用可能か？、同期されたカプリング/カプリング解除)

有効な運転モードでの処理と同時に、継続的なエラーのスキanningが実行されます。エラーが発生した場合、プロセスはエラー処理を開始します (217 ページ "エラーの検出および診断" 8.7.1 章を参照)。

運転モードを追加するには、235 ページ "運転モードの追加" 8.10.5 章を参照してください。

8.5 周期的処理

GATcompact では、周期的プログラム セクションは現在のところ空です。一般情報については、34 ページ "周期的処理" 5.4 章を参照してください。

8.6 タスク処理

8.6.1 タスク設定、GATcompact

サンプル アプリケーション GATcompact は、意図的に PLC タスク (PlcTask) に限定されています。

以下のセクションでは、GATcompact の PlcTask のデフォルト設定を示します。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
PlcTask	5	T#10ms	T#20ms	3

表 8-1:

必要な場合は、迅速な MotionTask および優先度の低い BackgroundTask を GATcompact に追加できます。

GATcompact

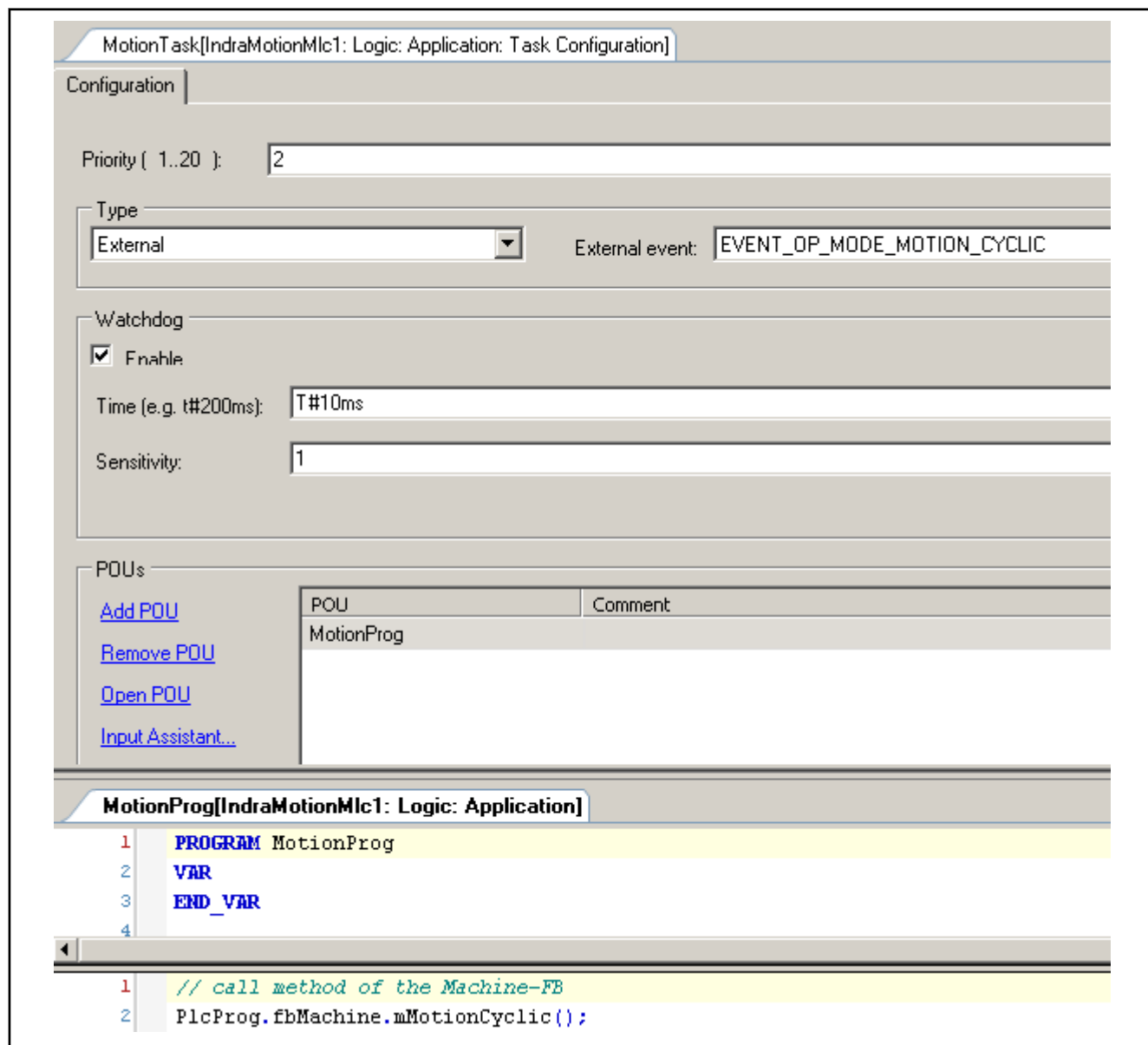
8.6.2 MotionTask の追加

36 ページ "MotionTask の一般的なタスク" 章 も参照してください。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
MotionTask	2	=sercos サイクル時間	T#2 ~ 10 ms	1..2

MOTION 同期タスクを追加するには、以下の手順が必要です。

- タスク設定、"MotionTask" の追加。設定については、214 ページ "MotionTask の設定" 8-18 図 を参照してください。
- POU "MotionProg" を追加し、MotionTask の設定で "MotionProg" を呼び出します
- "Machine" ファンクションブロックで "mMotionCyclic" メソッドを追加します。それを "MotionProg" で呼び出します。メソッド "mMotionCyclic" では、すべての変数 (一元的な軸入力ポイント Axes など) が利用できます。



MotionTask[IndraMotionMlc1: Logic: Application: Task Configuration]

Configuration

Priority { 1..20 } : 2

Type: External External event: EVENT_OP_MODE_MOTION_CYCLIC

Watchdog

☒ Enable

Time (e.g. t#200ms): T#10ms

Sensitivity: 1

POUs

[Add POU](#) [Remove POU](#) [Open POU](#) [Input Assistant...](#)

POU	Comment
MotionProg	

MotionProg[IndraMotionMlc1: Logic: Application]

```

1  PROGRAM MotionProg
2  VAR
3  END_VAR
4
5
6  // call method of the Machine-FB
7  PlcProg.fbMachine.mMotionCyclic();

```

図 8-18: MotionTask の設定

GATcompact

MotionTask から軸に継続的にアクセスするために、MotionTask で AxisInterface も呼び出す必要があります。すべての軸を MotionTask で処理すると、MotionTask に不要な負荷がかかるため、以下の手順をお勧めします。

以下の手順が必要です。

- 構造 "TE_AXIS_CONTROL_TYPE01" ("GATcompact/AxisInterface/Type/Control" フォルダ内) を、要素 "MotionSync" (タイプ BOOL) を含むことによって拡張します
- Init セクション ("actInit_entry" など) で、選択した軸の新しい要素 "MotionSync" を TRUE に設定します。
例 : ST Axes.MyVirtualAxis1.Ctrl.MotionSync := TRUE;
- TE_AxisInterfaceType01Prog():
 - フラグ クエリ MotionSync = FALSE でプログラムを補足します。

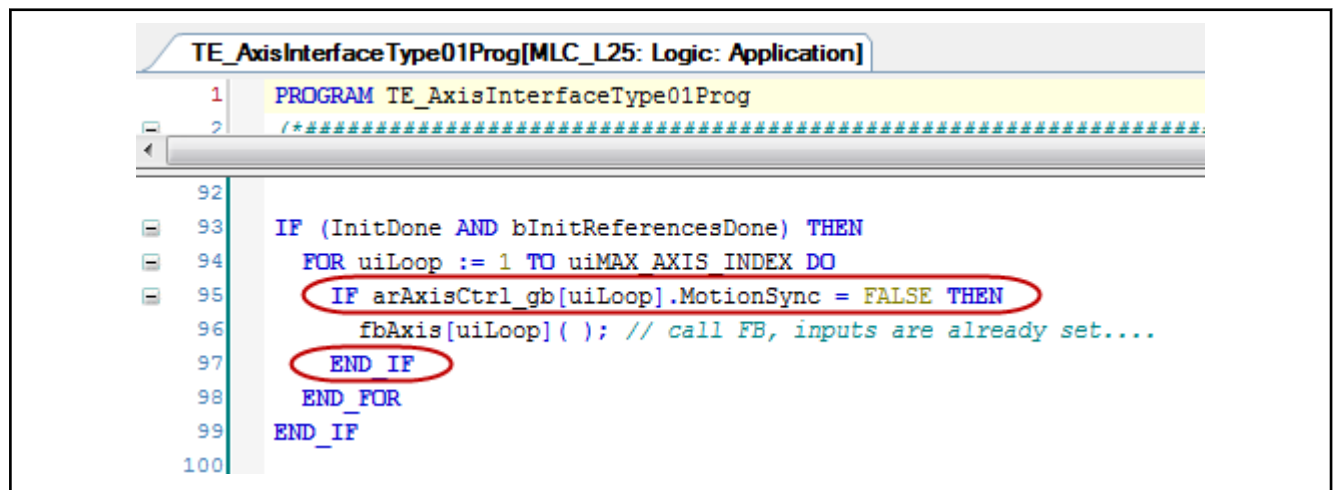


図 8-19: Flag query MotionSync = FALSE

- アクション "actMotionTask" を追加します。ファンクションブロック "fbAxis()" の呼出ループをコピーします (215 ページ "Flag query MotionSync = FALSE" 8-19 図を参照)。新しいループ変数 "uiLoopMotionSync" (uiLoop は使用しないでください) を作成し、フラグ クエリ "MotionSync" を TRUE に変更します。

結果 :

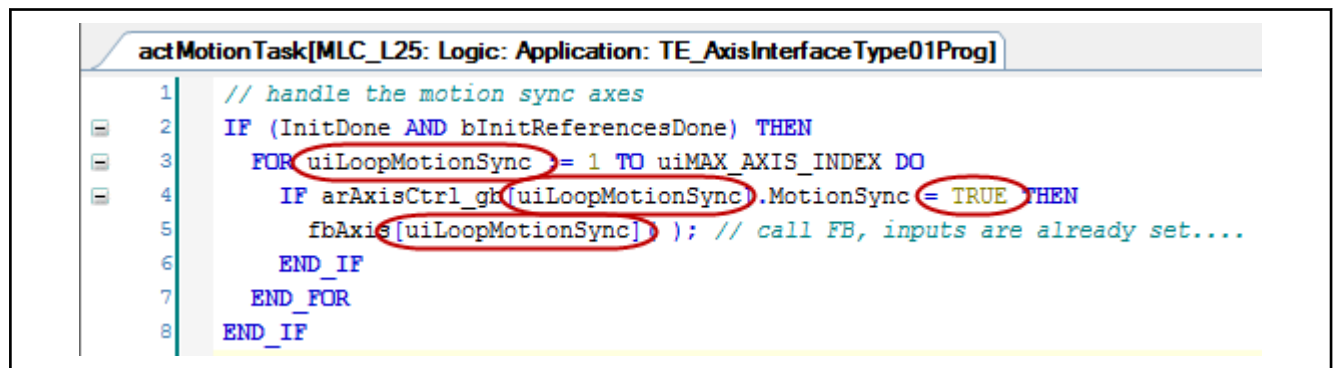


図 8-20: "actMotionTask" アクション

- MotionProg (TE_AxisInterfaceType01Prog.actMotionTask();) でアクション "actMotionTask" の呼出に追加します

GATcompact

8.6.3 BackgroundTask の追加

PlcTask のランタイムが長すぎる場合は、タイム クリティカルではないタスクをバックグラウンド タスクに移動して、PlcTask の負荷を減らすことができます。ただし、追加的なタスク Switch PlcTask <-> BackgroundTask によって、より大きな負荷がシステム全体にかかることに留意してください (36 ページ "BackgroundTask の一般的なタスク" 章 も参照)。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
BackgroundTask	~6	T#40ms	-	-

- BackgroundTask を追加するためには、以下の手順が必要となります。
- タスク設定、"BackgroundTask" の追加。設定については 216 ページ "BackgroundTask の設定" 8-21 図 を参照してください。
 - POU "BackgroundProg" を追加し、BackgroundTask の設定で "BackgroundProg" を呼び出します
 - "Machine" ファンクションブロックで "mBackground" メソッドを追加します。それを "BackgroundProg" で呼び出します。"mBackground" メソッドでは、メソッド内のすべての変数が利用可能です。

BackgroundTask[IndraMotionMlc1: Logic: Application: Task Configuration]

Configuration

Priority { 1..20 } : 6

TypeCyclicInterval (e.g. t#200ms): t#40ms

Watchdog☐ EnableTime (e.g. t#200ms):Sensitivity: 1

POUs

Add POURemove POUOpen POUInput Assistant...

POU	Comment
BackgroundProg	

BackgroundProg[IndraMotionMlc1: Logic: Application]

1PROGRAM BackgroundProg2VAR3END_VAR

1// call method of the Machine-FB2PlcProg.fbMachine.mBackground();

図 8-21: BackgroundTask の設定

8.7 エラー処理

8.7.1 エラーの検出および診断

プログラム テンプレート GAT^{compact} には、**中心的な位置でエラーを認識、処理、クリアするために予め定義済みの機能が含まれています**。さまざまな発生元からのエラー (コントロール エラー、アプリケーション エラー、ファンクションブロック エラーなど) が、機械全体のファンクションブロック "Machine" の "mErrorHandling" アクションによって自動的に集められ、ファンクションブロック "Machine" の診断出力を使用して利用可能になります。

複数のエラーが同時に発生した場合は、最も深刻なものが表示されます。同じ深刻度のエラーが複数、同時に発生した場合は、最後に検出されたものが出力されます。

FB "Machine" には、以下に説明されている診断目的の出力が含まれています。

要素	タイプ	説明
エラー	BOOL	TRUE はエラーを示します
DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの : 16#F0AE xxxx: アプリケーション エラー (例 : 安全扉が開いている) (xxxx = ユーザーが定義可能) 16#F0FB 0000: ファンクションブロック エラー (例 : TechFB でのエラー) -> 詳細については、ErrorID / ErrorIdent を参照 16#F0AA 0000: GAT 内部エラー -> 詳細については、ErrorID / ErrorIdent を参照
ErrorID	ERROR_CODE	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
ErrorIdent	ERROR_STRUCT	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
DiagMessage	STRING(60)	ドライブおよびコントロールのエラー診断テキストを含む

表 8-2:

DiagNo 出力は、HMI での使用のみを目的としています。エラー テキストには、任意の言語で数値情報が割り当てられます。

DiagMessage、**ErrorID** および **ErrorIdent** は、IndraLogic 表示の使用など、プログラマーによる試運転の際に主に表示される追加情報を提供します。

アプリケーション固有のエラーに関する注意

メソッド mSetAppError(Number) を呼び出すことにより、アプリケーション エラー (例 : 安全扉が開いている) を示すことができます。メソッド入力で伝達される"番号"は、アプリケーション エラー番号に対応します。

アプリケーション 番号が表示された場合、"Machine" FB の中央診断システムは、表示されたユーザー エラーの周期的なモニタリングを行い、アプリケーション エラー DiagNo = Error 16#F0AE xxxx (xxxx = アプリケーション エラー番号) を生成します。

グローバル変数リスト "UserConstGlobal" で、アプリケーション エラーに診断テキストを割り当てることができ、PLC 表示にそのテキストが表示されます。

uiMAX_APP_ERROR_INDEX の範囲 1 にある番号は、機械でのアプリケーション エラーについて有効です。定数 "uiMAX_APP_ERROR_INDEX" の値はデフォルトにより 64 に設定されています。必要な場合は、"Application" (ア

GATcompact

アプリケーション) フォルダにあるグローバル変数リスト "UserConstGlobal" でこの定数を調整できます。

アプリケーション エラーを表示するためのコード例:

```
// Recommended code to issue user specific errors (Example)
IF bDoor10Open THEN
    mSetAppError(1);
END_IF
```

深刻なエラーに関する注意

コントロール診断システムでは、10段階の"深刻度"が定義されています (「IndraLogic XLC-/IndraMotion MLC diagnostic system」 (IndraLogic XLC-/IndraMotion MLC 診断システム) を参照)。

深刻度はエラーリアクションに影響を及ぼします。

- DiagNo = 16#F0xx xxxx は最も低い深刻度です (例: アプリケーションエラー、ファンクションブロックエラー)
- DiagNo = 16#F9xx xxxx は最も高い深刻度です (致命的なシステムエラー)

複数のエラーが同時に発生した場合は、深刻度が最も高いエラーが表示されます (41 ページ "エラーリアクション" 5.6.4 章 を参照)。

ファンクションブロックエラーに関する注意

呼び出されたファンクションブロック (技術ファンクションブロックなど) でエラーが発生した場合は、メソッド "mSetFBError(ErrorID, ErrorIdent)" を呼び出して、エラーを引き起こしたファンクションブロックの "ErrorID" と "ErrorIdent" 出力をメソッドに転送することをお勧めします。これにより、エラーが "Machine" FB の診断システムに確実に転送され、"ErrorID" と "ErrorIdent" が "Machine" FB の診断出力で利用可能になります。

ファンクションブロックエラーを表示するためのコード例:

```
// Recommended code a used FB issues an error (Example)
IF fbCrossCutter.Error THEN
    mSetFBError(ErrorID:=fbCrossCutter.ErrorID,
                ErrorIdent:=fbCrossCutter.ErrorIdent);
END_IF
```

8.7.2 エラーリアクション

エラーリアクションは、コントロールおよびドライブシステムにおけるエラーによる損傷を可能な限り避けるように設計されています。このため、深刻度 2 以上のエラーリアクションは、制御ファームウェアと実在の軸によって自動的に実行されます (41 ページ "エラーリアクション" 5.6.4 章 を参照)。

深刻度が第 2 段階未満のその他のエラー (アプリケーションエラー、ファンクションブロックエラーなど) の場合、制御ファームウェアと実在の軸による自動エラーリアクションはありません。

このため、GATcompact には、ユーザー固有のエラーリアクションを開始するエラー検出機能が含まれています。

通常の GAT エラーリアクション

デフォルト設定では、GATcompact エラーリアクションは有効になっています。エラーが発生すると、選択されている運転モードが中断され (例外: 初期化)、エラーリアクションが実行されます。エラーリアクションは、ファンクションブロック "Machine" の "actError_entry" アクションでプログラミングされ、アプリケーションに適合させることができます。

標準的なケースの場合、すべての軸の減速は "ModeAH" (ModeAH = 完全な停止への減速、トルクは負荷状態が継続) によりプログラミングされます。

```

// ----- Error reaction example 1 -----
// Decelerate all module axes to standstill and hold power (e.g. hanging axes)
// --> switch all axes to ModeAH
// alternative: set to ModeAb to switch the axes OFF to Ab
// -----
FOR uiLoopIndex := 1 TO uiMAX_AXIS_INDEX DO
  // for all active axes, but only when added by the wizard (<>NO_OBJECT)
  IF Axes.arAll[uiLoopIndex].Status.Admin.Active      AND
     Axes.arAll[uiLoopIndex].AxisRef.AxisNo <> NO_OBJECT THEN
    Axes.arAll[uiLoopIndex].Ctrl.Admin.OpMode.en := ModeAH; // or set ModeAb
    // reset SetupMode Inputs to avoid hanging buttons in the
    // System_Overview_All_Axis visualisation
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.Enable := FALSE;
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.JogMinus := FALSE;
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.JogPlus := FALSE;
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.Homing := FALSE;
  END_IF
END_FOR

```

図 8-22: アクション "actError_entry" でのエラーリアクションの標準的な事例
上記の例で "ModeAH" が "ModeAb" で置き換えられた場合、減速で静止した後で実在の軸のトルク有効化がアクティブになります。



"ModeAb" を使用して油圧軸を静止させることをお勧めします。



"軸が一時停止している場合"、軸の降下を防ぐためにトルクが有効化されないことがあります。これらのアプリケーションでは、すべての軸を "ModeAH" に切り替えることをお勧めします。

軸を同期させて減速する必要があることも少なくありません (材料ウェブの断裂を防ぐためなど)。これらのアプリケーションでは、同期させた動きですべての軸を接続したままにすること、およびマスターのみを減速することをお勧めします。このユース ケースは下記の例に示されています。

モジュール アクション "actError_entry" (同期していない軸のみが "ModeAH" を使用して減速される) での軸の同期減速の例

GATcompact

```

// ----- Error reaction example 2 -----
// Keep all axes in synchronization with the master axis
// and decelerate only the master axis
// -----
FOR uiLoopIndex := 1 TO uiMAX_AXIS_INDEX DO
  // for all active axes, but only when added by the wizard (<>NO_OBJECT)
  IF Axes.arAll[uiLoopIndex].Status.Admin.Active      AND
     Axes.arAll[uiLoopIndex].AxisRef.AxisNo <> NO_OBJECT THEN
    IF (Axes.arAll[uiLoopIndex].AxisData.SynchronizedMotion) THEN
      ; // Axis is in synchronized motion --> no reaction as the master axis
      // decelerates to standstill and the slave axis follows in synchronization
    ELSE
      Axes.arAll[uiLoopIndex].Ctrl.Admin._OpMode.en := ModeAH;
      // Axis is not in synchronized motion --> stop axis
    END_IF
    // reset SetupMode Inputs to avoid hanging buttons in the
    // System_Overview_All_Axis visualisation
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.Enable := FALSE;
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.JogMinus := FALSE;
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.JogPlus := FALSE;
    Axes.arAll[uiLoopIndex].Ctrl.SetupMode.Homing := FALSE;
  END_IF
END_FOR

```

図 8-23: アクション "actError_entry" での軸の同期減速の例

実在の軸の同期減速には、以下の設定が必要です。

- モジュール バス反応をオフにする
- ドライブのエラーリアクションを "可能な最善のシャットダウン = 30 s NC 対応" により設定する



IndraLogic XLC/IndraMotion MLC コントロール システムの実在の軸の "可能な最善の減速 = 30s NC 対応" 設定で F2 エラーが発生した場合、エラーの影響を受ける軸は緊急停止ランプ A-0-0224 により減速されます。問題のない他のすべての軸は、影響を受けません。

通常の GAT エラーリアクションをオフにする

特別な場合、通常の GAT エラーリアクションをオフにすることができます。これを行うには、変数宣言を通じて変数 "bDisableErrorHandling_gb" を以下のとおり変更します。

- グローバル変数リスト: "UserVarGlobal"
- 宣言を以下のとおり変更します: "bDisableErrorHandling_gb: BOOL := TRUE;"

エラーリアクションがオフになっても、現在の運転モードは中断されません。しかし、エラーは継続して表示され、クリアすることも可能です。

初期化中の GAT エラー反応

デフォルト設定では、初期化中、GATcompact エラーリアクションは無効になっています。しかし、エラーは継続して表示され、クリアすることも可能です。

特別な場合、初期化中に GAT エラーリアクションをオンにすることができます。

これを行うには、変数宣言を通して変数 "bEnableErrorReactionInit_gb" を以下のとおり変更します。

- グローバル変数リスト : "UserVarGlobal"
- 宣言を以下のとおり変更します : "bEnableErrorReactionInit_gb: BOOL := TRUE;"

エラーリアクションが有効な場合、エラーが発生すると初期化が中断されます。その後、プログラミングされたエラーリアクションが実行されます。エラーリアクションは、"**Machine**" ファンクションブロックの "actError_entry" アクションでプログラミングされます。



初期化中のエラーリアクションは、通常の GAT エラーリアクションが有効な場合 (bDisableErrorHandling_gb = FALSE) のみ実行されます。

8.7.3 エラーのクリア

一元的なエラーのクリアは、"**Machine**" FB の "ClearError" 入力の正のエッジにより自動的に実行されます。システム全体を問題のない状態にするために、ファンクションブロックにより、すべての下位コンポーネント (例 : ImcInterface¹⁾、AxisInterface など...) にコマンドが転送されます。エラーがクリアされ、エラーリアクションのために運転モードを終了したプロセスがクリアされると、有効な、選択されている運転モードが処理を続行します。

8.8 表示

8.8.1 起動中の支援のための内部表示

一般情報

GATcompact には、試運転中の支援および拡張された診断機能を提供する、予め定義済みの IndraLogic 表示オブジェクトが含まれています。

System_Overview

表示要素 "System_Overview" は、個々の軸についてインスタンス化できます。これは "MyVirtualAxis1" 軸および "MyVirtualAxis2" 軸の "System_overview_All_Axis" 表示で例として実行されています。

¹⁾ *IndraMotionControl* インターフェース = 制御へのインターフェースを備えた *IndraLogic* ライブラリ。例 : エラーをクリア

GATcompact

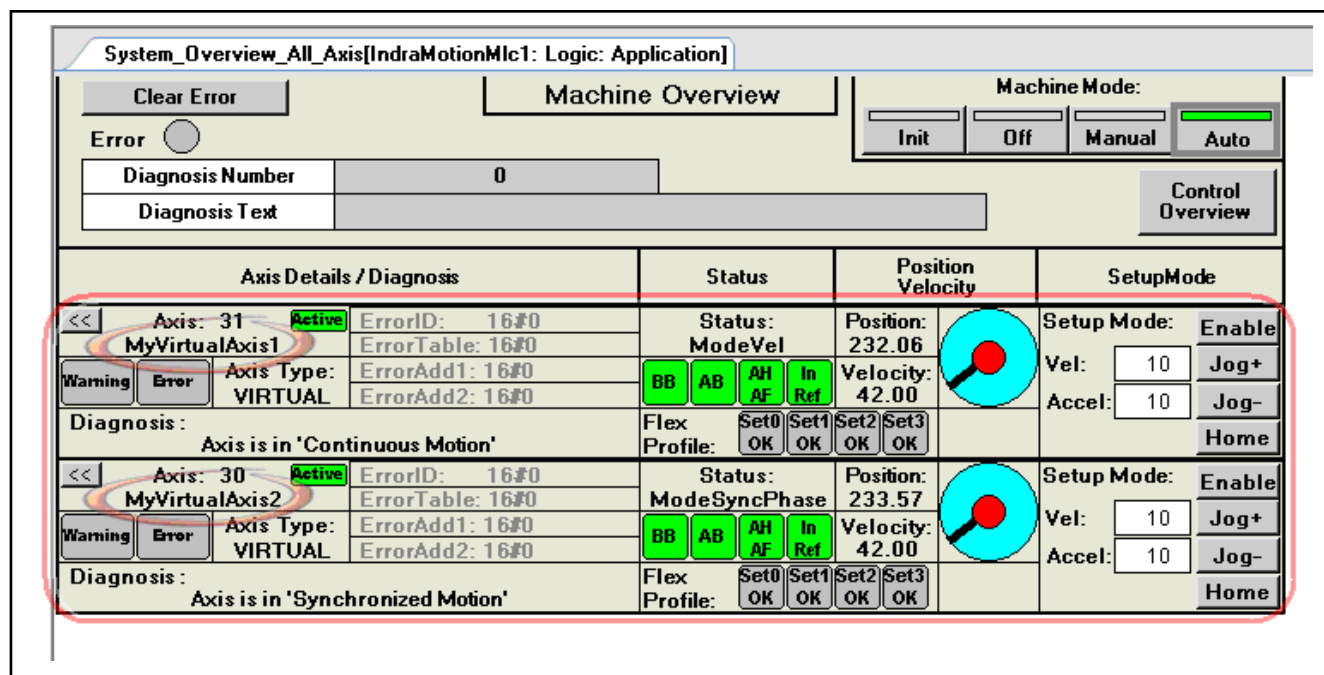


図 8-24: 起動中の支援のための内部表示

"Frame" オブジェクトを使用して、表示内で表示オブジェクト "System_Overview" のインスタンスを作成することができます。

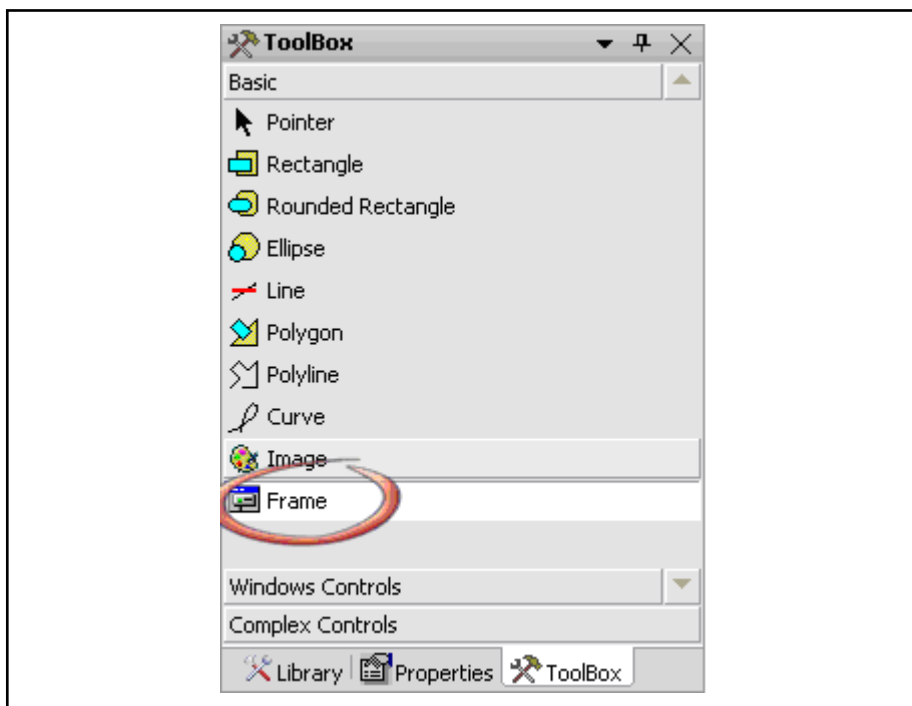


図 8-25: 表示オブジェクト "System_Overview" のインスタンスの作成

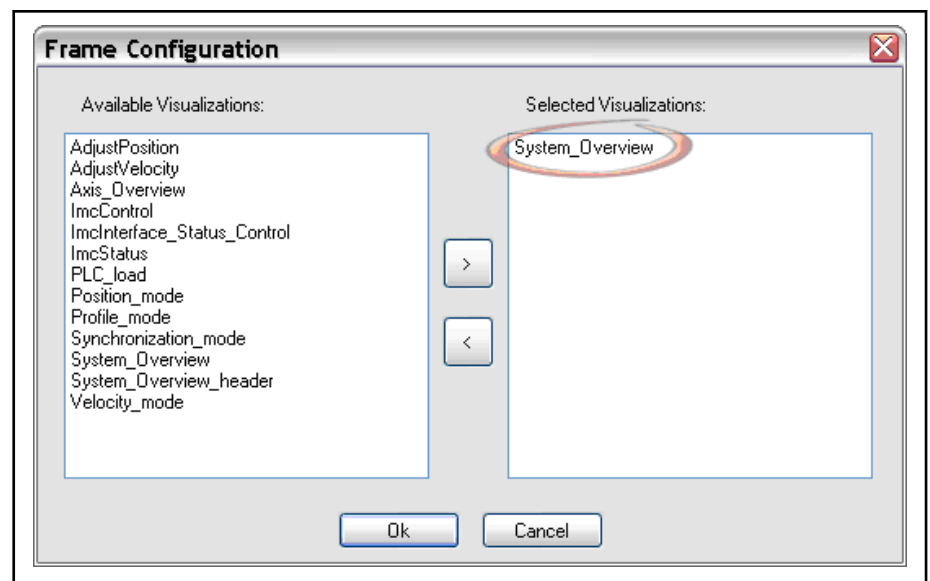


図 8-26: 表示オブジェクト "System_Overview" のインスタンスの作成
ここで、対応する軸参照を「Properties」の「References」の項目に入力する必要があります。

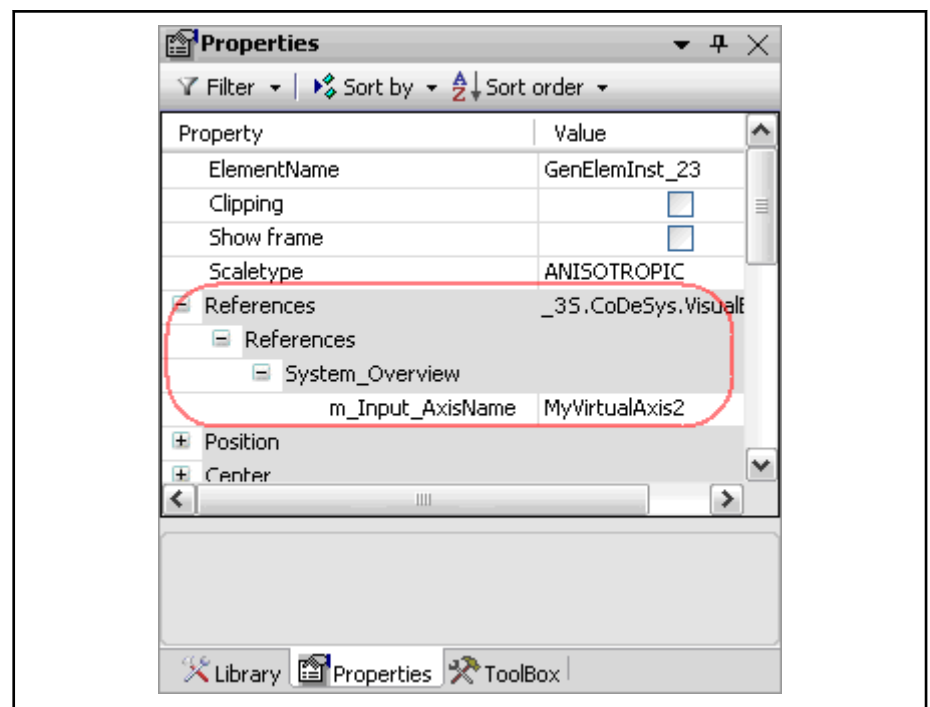


図 8-27: 表示オブジェクト "System_Overview" のインスタンスの作成

8.8.2 独立した HMI による外部表示 (WinStudio 表示)

一般情報

GATcompact では予め定義済みの WinStudio 表示が利用できます。ウィザードを使用して、この表示をプロジェクトに追加できます。

GATcompact



WinStudio を編集するには、少なくとも以下の開発環境のライセンス (またはそれ以上) が必要です。

WinStudio 開発ライセンス、変数最大 1,500 個 (Mat.No. R911323596)

VEP でプログラムを実行するには、以下のランタイム ライセンスが必要です。

オペレーティング システムとして Windows CE を使用しているデバイス用の WinStudio ランタイム ライセンス、変数最大 1,500 個 (Mat. No. R911323620)

WinStudio 表示の追加



WinStudio 表示をプロジェクトに追加するためには、対応する表示デバイスがプロジェクトになければなりません。

"ドラッグ & ドロップ"を使用して、デバイス ライブラリからプロジェクト ツリーに VEP を追加し、以下のとおり設定します。

- デバイス タイプ：VEPxx (CE)
- アプリケーション タイプ：WinStudio アプリケーション

図 8-28: 表示デバイスの追加

対応する表示デバイスがある場合、下記のリンクとウィザードを使用して WinStudio 表示を追加します。

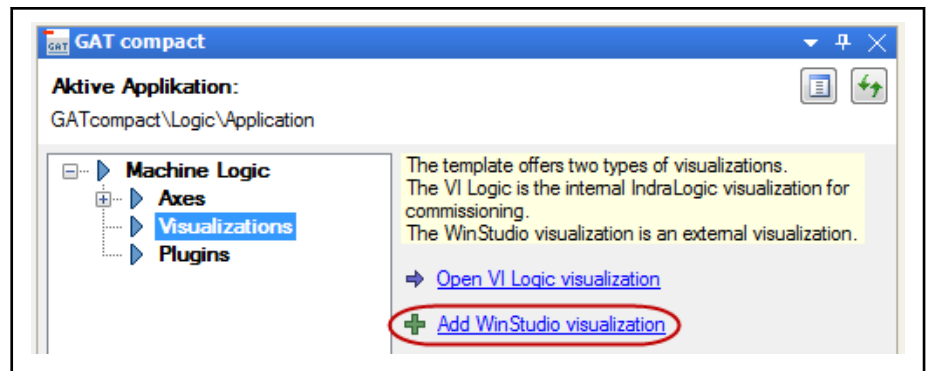


図 8-29: WinStudio 表示の追加

各設定に対応するプロジェクトの表示デバイスはすべて下記のウィザード ダイアログに含まれています。

対応する表示デバイスを選択した後で、**Apply** (適用) をクリックして WinStudio 表示を追加します。

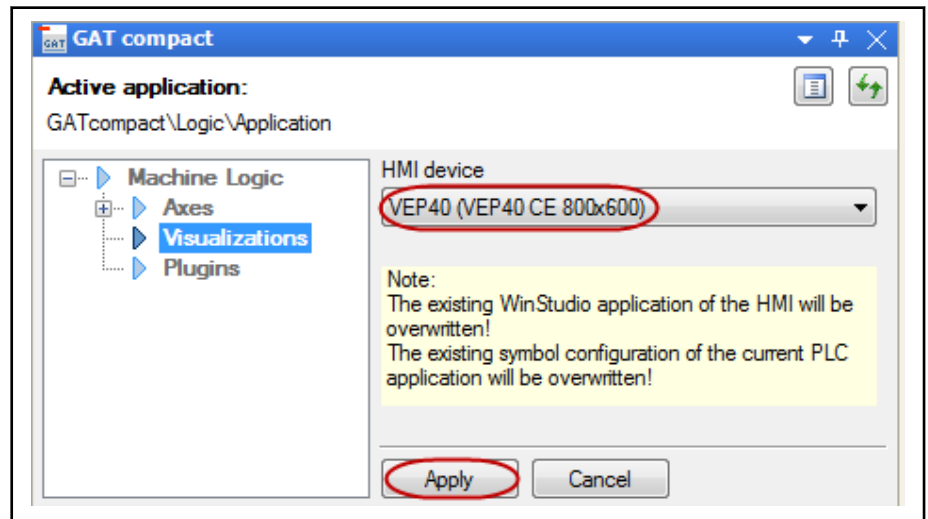


図 8-30: 表示デバイスの選択 – WinStudio 表示の追加



WinStudio 表示が追加されると、コントロールの記号設定とともに表示デバイスの既存の表示データが上書きされます。

表示が追加された後で、表示データを編集できます。

コントロール アドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

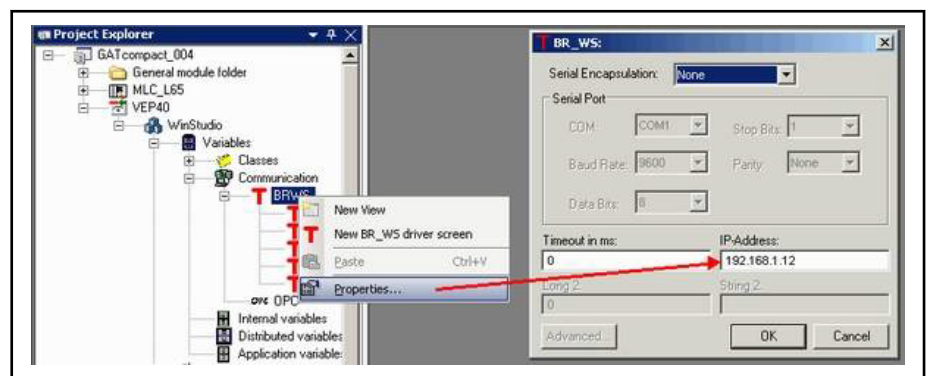


図 8-31: コントロールの IP アドレス設定

GATcompact

ここで、WinStudio ノードのコンテキスト メニューにある  "Start application" (アプリケーションを起動) を使用して WinStudio 表示のエミュレーションを開始します。

ダイアログ構造

操作画面は以下の 3 つのセクションに分かれています。

- ステータス情報があるヘッダー
- メニュー関係の表示と機能キーがあるメイン画面
- ユーザー マップの切替を行うためのキーがあるフッター

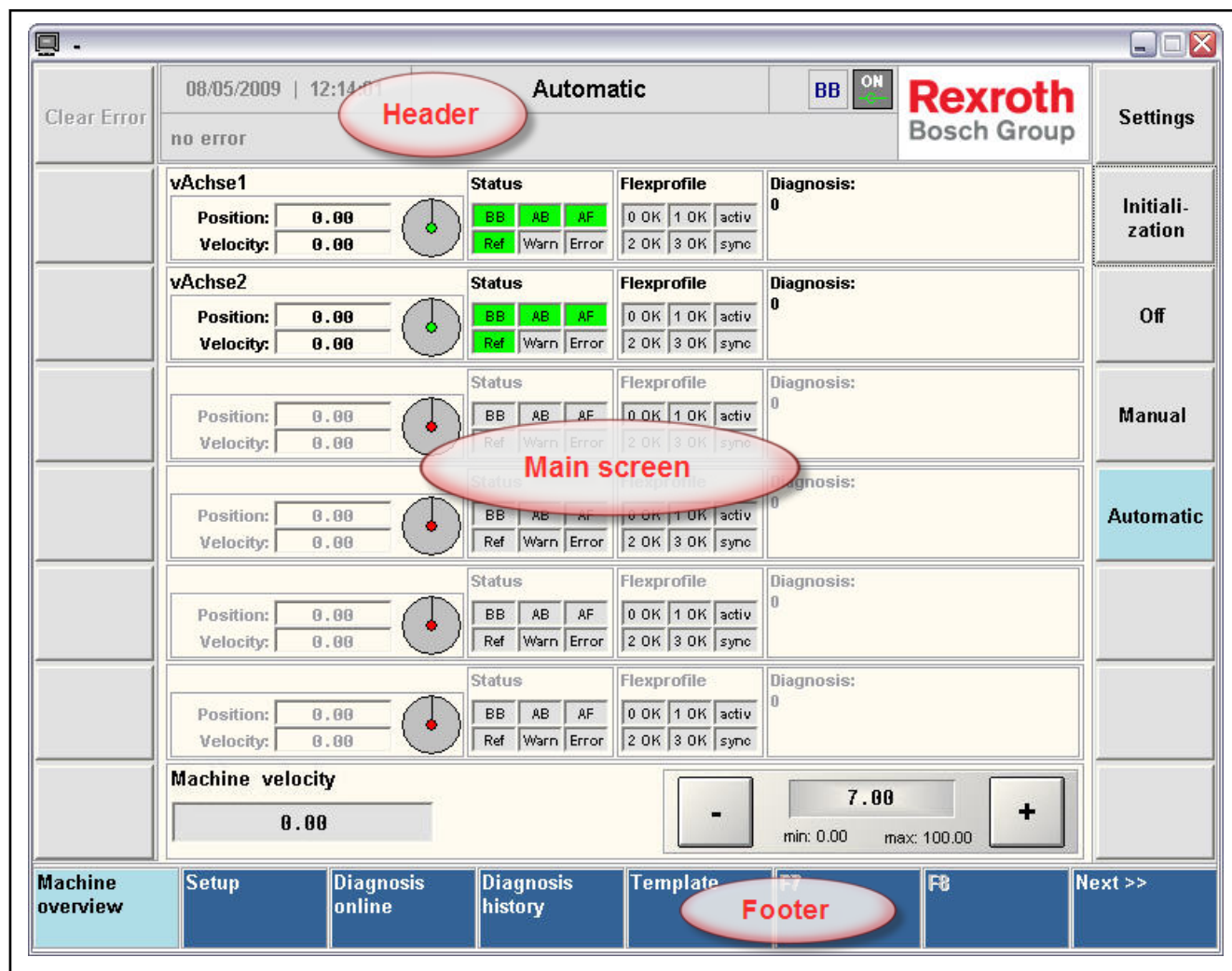


図 8-32: WinStudio 表示

ヘッダーには以下が含まれています：

- 運転モードの表示
- エラーをクリアしたり "Settings" (設定) ダイアログ (言語の切替、数値表示の小数点位置の設定などのため) を開いたりするためのグローバル機能キー
- 現在の機械診断の表示
- 現在の sercos フェーズの表示 (アクティブなフェーズ切替を含む)
- 通信ステータスの表示

メイン画面には、操作画面 (軸の情報、運転モードの切替など) に関連する表示と機能キーが含まれています。

フッターには、ユーザー マップを切り替えるためのキーが含まれています。

表示の拡張

操作画面の追加

表示を拡張するには、既存の "テンプレート" 画面をコピーして新しい名前を付けて挿入することにより、新しいユーザー マップを作成できます。

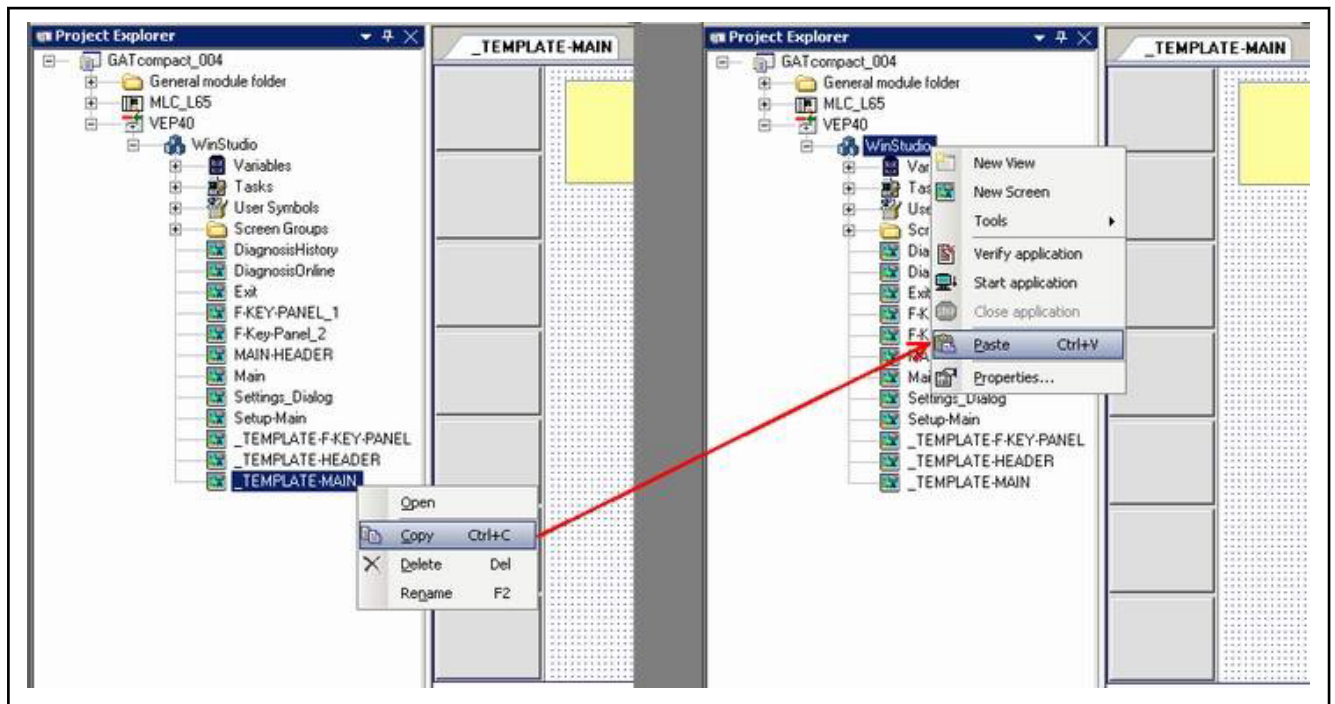


図 8-33: 操作画面の複製

軸データを表示するための要素は、"コピー"&"ペースト"で複製できる、グループ化された個別のオブジェクトです。異なる軸のために挿入されたオブジェクトを再設定するには、各オブジェクトに「置換」機能を使用して軸配列インデックスを適合させる必要があります (下の図を参照)。



図 8-34: 挿入されたオブジェクトの再設定

GATcompact

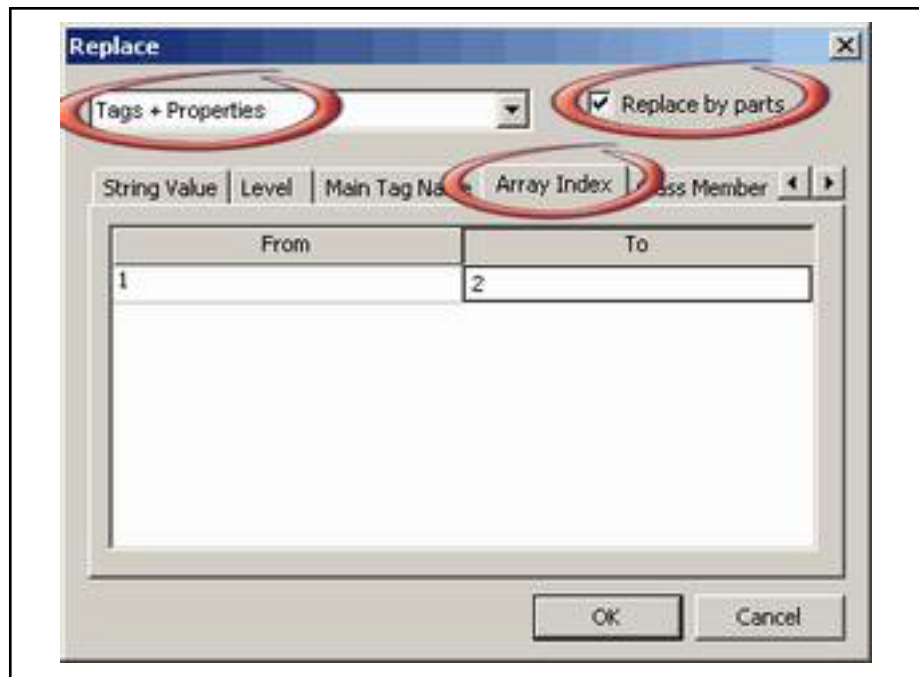


図 8-35: 挿入されたオブジェクトの再設定

フッターの拡張

ユーザー マップを切り替えるために使用されるキーは、パラメータ化が可能なグラフィック オブジェクト (リンクされた記号) として保存されます。キーには以下の転送パラメータがあります。

- Caption -> キー ラベル
- Disable -> キー機能をブロックするための入力 (グレー表示のラベル)
- ScreenName -> 操作画面を開くための画面名 (または画面グループ)
- ScreenNumber -> 画面に割り当てられる番号 (それぞれのキーを強調表示するため)

通信接続

表示とコントロール間の通信は、BR-WS ドライバを使用して記号的に行われます。これを行うには、表示する PLC 変数の記号エントリを生成します (47 ページ "通信設定" 章を参照)。

予め定義済みの WinStudio 表示に必要な記号設定には、以下のエントリが含まれます。

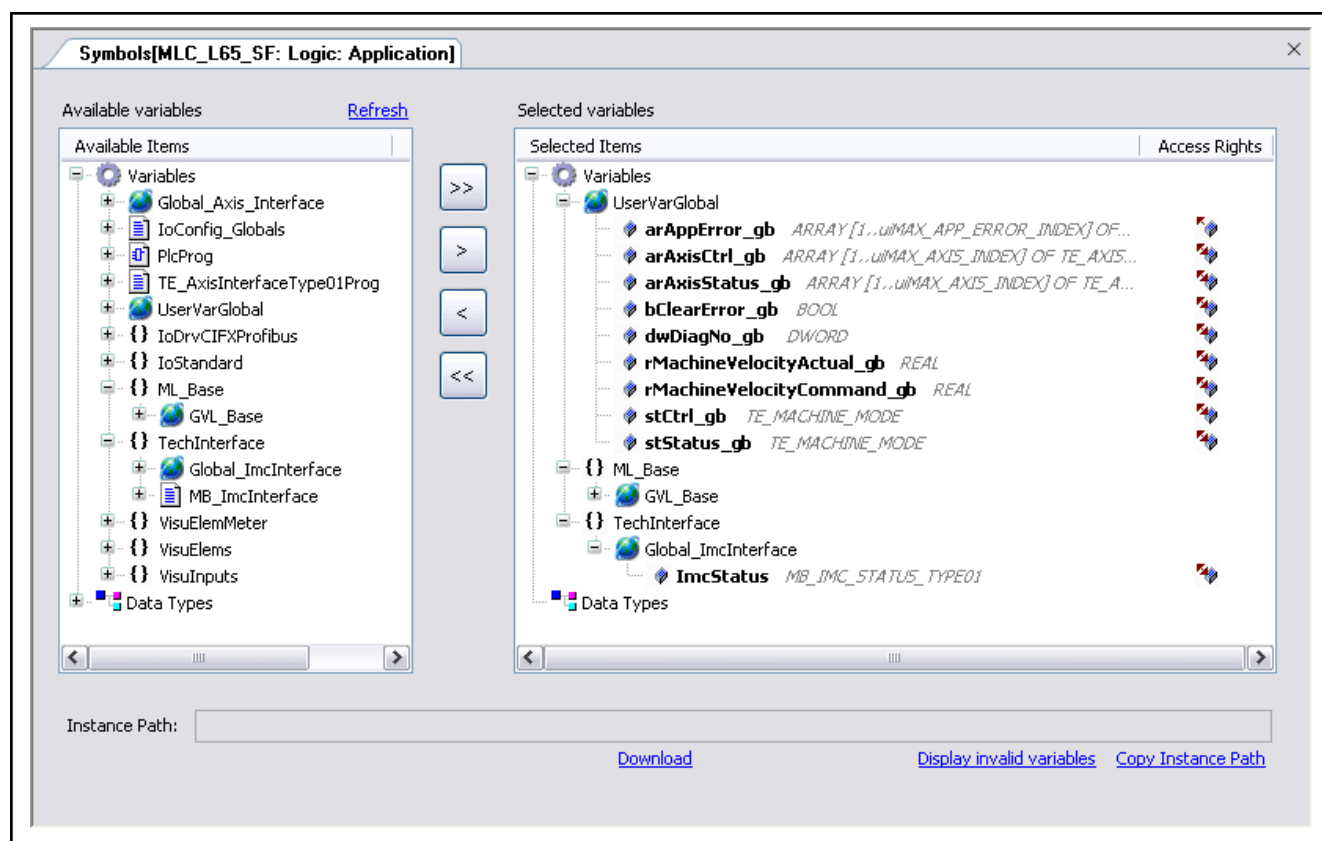


図 8-36: 予め定義済みの WinStudio 表示の記号設定



ライブラリから取得したグローバル変数リストの変数の記号エントリは、各ライブラリのバージョンが変更された後で記号設定に挿入しなおす必要があります。

コントロールの IP アドレス設定：

コントロール アドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

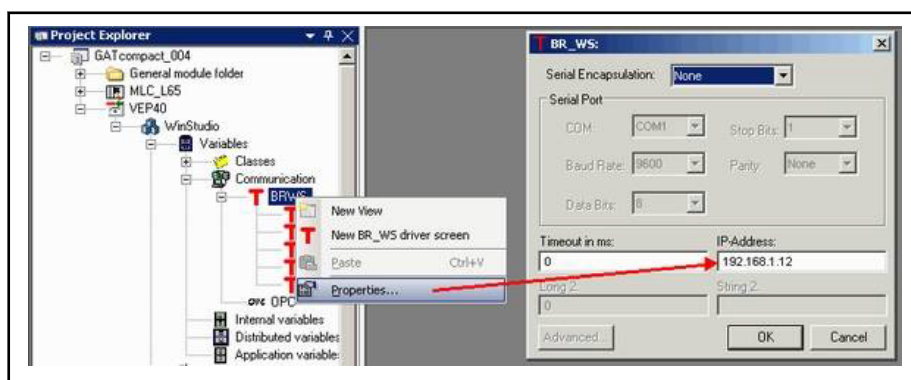


図 8-37: コントロールの IP アドレス設定

通信を構成するために、変数がさまざまなドライバシートに配置されています。

1. 高速で周期的に読み取る：これらの変数の値は短い間隔で周期的に読み取られます (診断番号、軸の位置など)
2. 1 回のみ読み取る：これらの変数は表示が開始された後で 1 回のみ読み取られます (軸の名前など)

GATcompact

3. **フェーズの切替後に読み取る**：これらの変数はフェーズの切替後に 1 回のみ読み取られます (軸がアクティブになった場合など)
4. **周期的に読み書き**：これらの変数の値は、表示により読み書きの両方が行われます (機械の速度、セットアップ モードの軸の速度など)
5. **書き込みのみ**：これらの変数は、表示を通じて読み取りのみが行われず (エラー のクリアの信号を発信する...)

ドライバシートを拡張するには、BR-WS ブラウザでオフライン記号ファイル (*.XML) を検索し、対応する変数を選択できます。

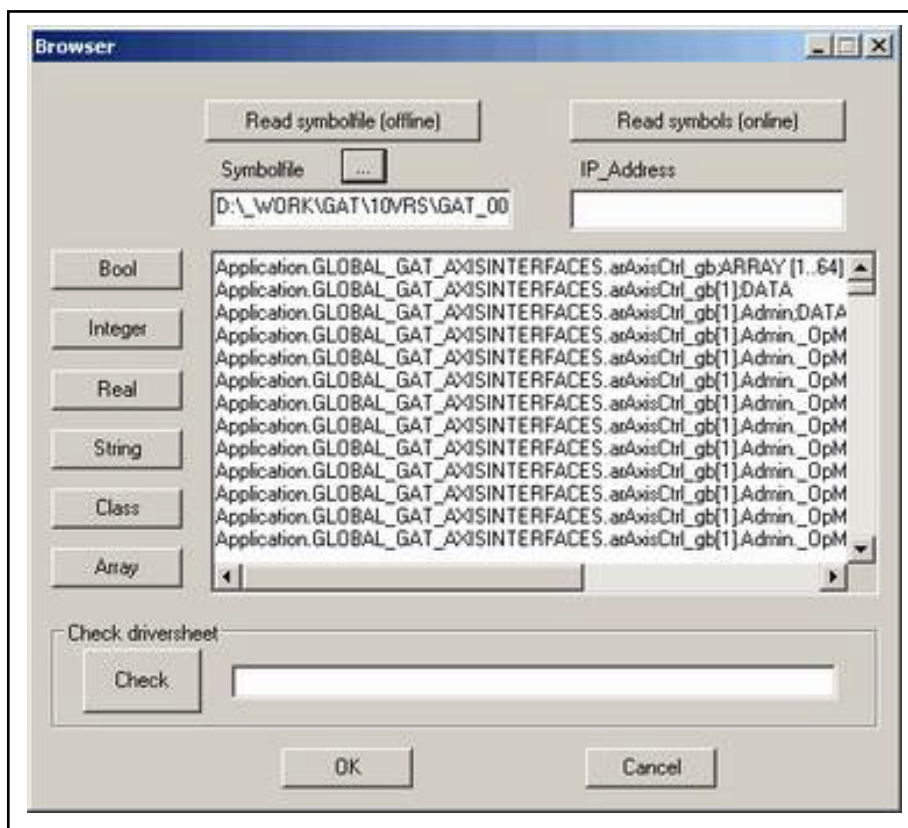


図 8-38: ドライバシートの拡張

複数言語機能

表示の複数言語機能は、IndraWorks への翻訳機能の組み込みにより実現されました。

コントロールおよびドライブの診断の翻訳データ、ならびに表示テキストは、すでに準備されています。翻訳データは IndraWorks で変更や修正が可能です。

ドライブ診断はドイツ語、英語、フランス語、スペイン語、トルコ語で含まれています。

他のすべてのテキストはドイツ語と英語です。

8.9 GATcompact のインターフェース

8.9.1 一般情報

GATcompact のほとんどのインターフェース データは、"UserVarGlobal" 変数リストにまとめられています。データの一部はライブラリで利用可能になっています。

GATcompact

以下のセクションでは、"Name of the variables" (変数の名前) 列のライブラリからのデータがそのように示されています。他のすべてのデータは "UserVarGlobal" 変数リストにあります。

8.9.2 機械状態のデータ

I/O タイプ	名称	データの種類	説明
I	stCtrl_gb	TE_MACHINE_MODE_CTRL	機械運転モードのコマンド
I	bSkipColdstartAtStartup_gb	BOOL	GAT ^{compact} を起動するとき、アプリケーションに必要なパラメータ化を実行するために、パラメータ化モード (P2) に切り替えられます。アプリケーションに P2 でのパラメータ化が不要な場合、このフェーズ切替は TRUE で省略できます。
O	stStatus_gb	TE_MACHINE_MODE_STAT US	現在の機械運転モードに関するフィードバック

I 書き込み用入力
O 読み取り用出力

8.9.3 エラー処理用データ

I/O タイプ	名称	データの種類	説明
I	bClearError_gb	BOOL	TRUE = グローバル クリア エラー
I	bDisableErrorHandling_gb	BOOL	TRUE = エラーが発生した場合、プロセスは "ERROR" 状態に切り替わりません。エラー表示は続行され、エラーのクリアは可能です。
O	bError_gb	BOOL	TRUE = グローバル エラーが発生しました
O	dwDiagNo_gb	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの： DiagNo = 16#F0AE xxxx: アプリケーション エラー (例：安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー
O	enErrorID_gb	ERROR_CODE	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	stErrorIdent_gb	ERROR_STRUCT	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	strDiagMessage_gb	STRING(60)	コントロール/軸の診断テキスト
O	arAppError_gb	ARRAY OF BOOL	アプリケーション固有のエラー (例：安全扉が開いている)

I 書き込み用入力
O 読み取り用出力

GATcompact

8.9.4 軸データ

I/O タイプ	名称	データの種類	説明
I	arAxisCtrl_gb	ARRAY OF TE_AXIS_CONTROL_TYPE 01	軸用のコマンド
O	arAxisStatus_gb	ARRAY OF TE_AXIS_STATUS_TYPE01	軸からのフィードバック
I/O	AxisData (ML_Base ライブラリからの GVL_Base)	ARRAY OF ML_AXISDATA_SM	軸の入力と出力のあるファームウェア構造

I 書き込み用入力
O 読み取り用出力
I/O 入力と出力のある構造

 arAxisCtrl_gb の要素は "Machine" FB に書かれているため、外部的に使用することはできません。

8.9.5 コントロール データ

I/O タイプ	名称	データの種類	説明
I	ImcCtrl (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_CONTROL_TYPE01	コントロール用のコマンド
O	ImcStatus (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_STATUS_TYPE01	コントロールからのフィードバック

I 書き込み用入力
O 読み取り用出力

 ImcCtrl_gb の要素は "Machine" FB に書かれているため、外部的に使用することはできません。

8.10 操作手順：一般的なユーザー アクティビティ

8.10.1 軸データへのアクセス

FB "Machine" では、構造 **Axes** は軸データにアクセスするための一元的なアクセス ポイントとして定義されています。軸のすべての関連データは、Axes (軸) 構造を使用してアドレスを指定できます。

この構造は、軸の名前を用いて軸データにアクセスできるように設計されています。これにより、読み取りが容易なプログラム コードが生成されます。

"Axes" (軸) の項目では以下のデータが利用可能です。

- Axes.Name.AxisData => 軸データ構造：ファームウェアのステータスデータおよび軸データ コンテナへのアクセス
- Axes.Name.Ctrl => AxisInterface の制御構造
- Axes.Name.Status => AxisInterface のステータス構造
- Axes.Name.DV => 直接変数 (機能変数) へのアクセス
- Axes.Name.AxisRef => ファンクションブロック呼出の軸参照構造 (MB_ReadParameter など)

Array Axes.arAll[] を使用して、上に挙げたデータにアクセスすることもできます。この方法は、ループプログラミング (例として、"actModeOff_entry" を参照) の場合にメリットがあります。

両方のバージョンで、同じストレージロケーションへの参照がアクセスされます。

例：以下のアクセスメソッドは完全に同等で、同じ結果が得られます。

- Axes.Name.AxisData.rActualPosition_i
- Axes.arAll[Name.AxisNo].AxisData.rActualPosition_i

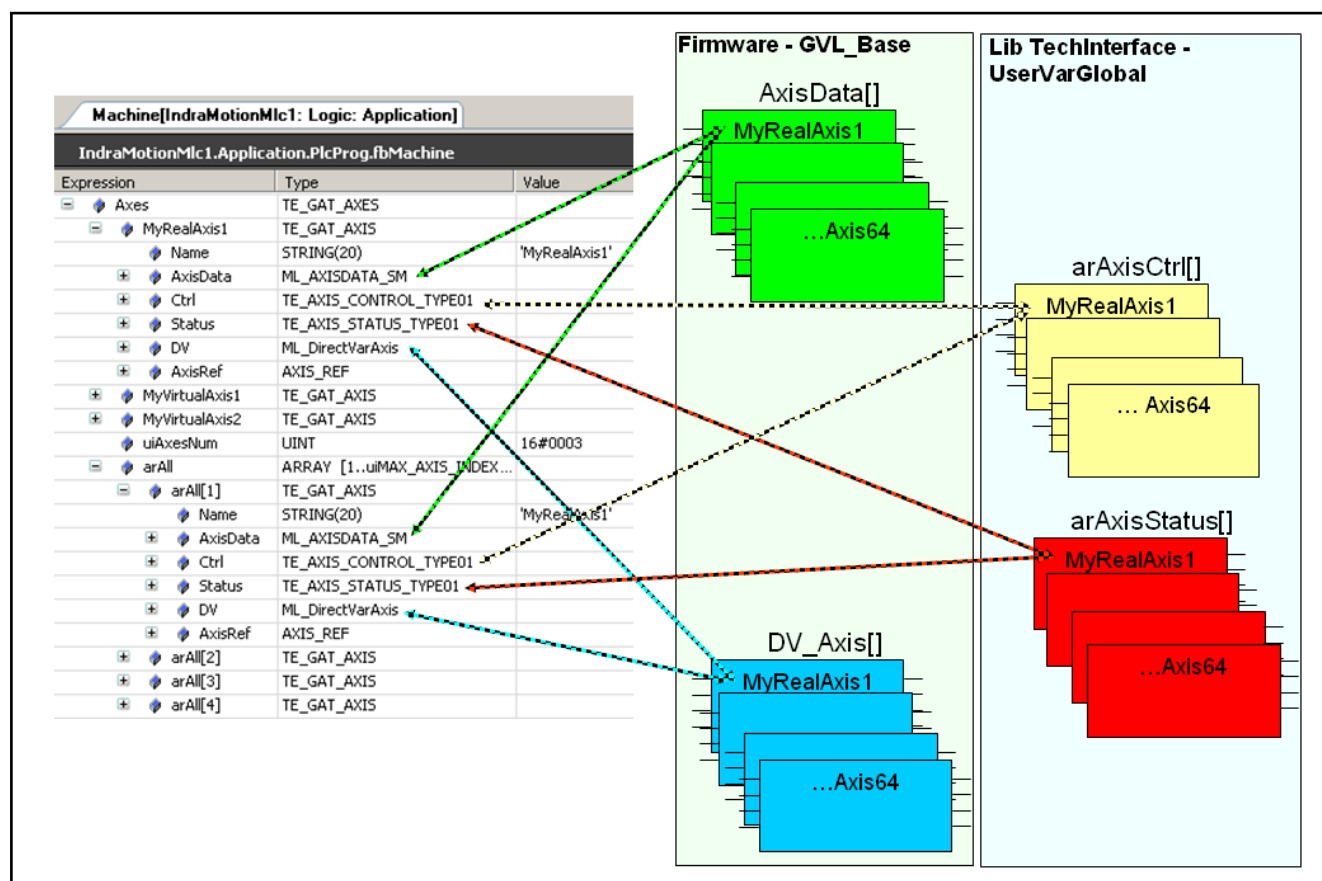


図 8-39: 軸データの概要

8.10.2 軸の追加

軸を追加するためには、以下の手順が必要です。

- IndraWorks プロジェクトに軸を追加する (sercos ノードにドラッグする)
- "SystemOverview_All_Axis" 表示を使用する場合：新しい軸のフレームを追加し、プロパティで新しい軸への参照を設定する (221 ページ "System_Overview" 章も参照)
- GAT ウィザードを使用した軸の追加

GATcompact

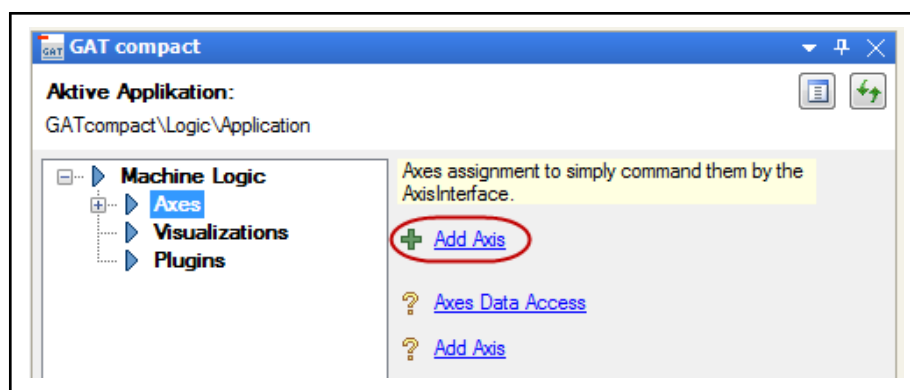


図 8-40: GAT ウィザードを使用した軸の追加

8.10.3 軸番号が 16 より大きい軸の最大数を大きくする

論理軸番号が「16」より大きい軸がプロジェクトに含まれている場合は、グローバル定数 "uiMAX_AXIS_INDEX" の値を適宜調整します。

"uiMAX_AXIS_INDEX" は、少なくともプロジェクト内の最も大きい論理軸番号の値でなければなりません。

デフォルト設定は "uiMAX_AXIS_INDEX := 16" です

定数 "uiMAX_AXIS_INDEX" を素早く見つけるには、IndraWorks 検索機能 (<Ctrl>+<F>) を使用するが、または以下のフォルダ構造を辿ります。

Application/UserConstGlobal

8.10.4 軸の削除/名前変更



ウィザードによって変更されるのは、軸宣言にある名前のみです。軸がすでに実装または内部表示で使用されている場合、新しい軸の名前は、"検索・置換"を使用して、これらの位置で変更する必要があります。

軸の削除/名前変更を行うためには、以下の手順が必要です。

- ndraWorks プロジェクトから軸を削除する/ndraWorks プロジェクトで名前を変更する
- "SystemOverview_All_Axis" 表示を使用する場合：削除した軸のフレームを削除し、プロパティで軸への参照を変更する
- GAT ウィザードを使用して軸を削除する/名前を変更する ("Properties" (プロパティ) を介した名前変更は、下の図を参照)

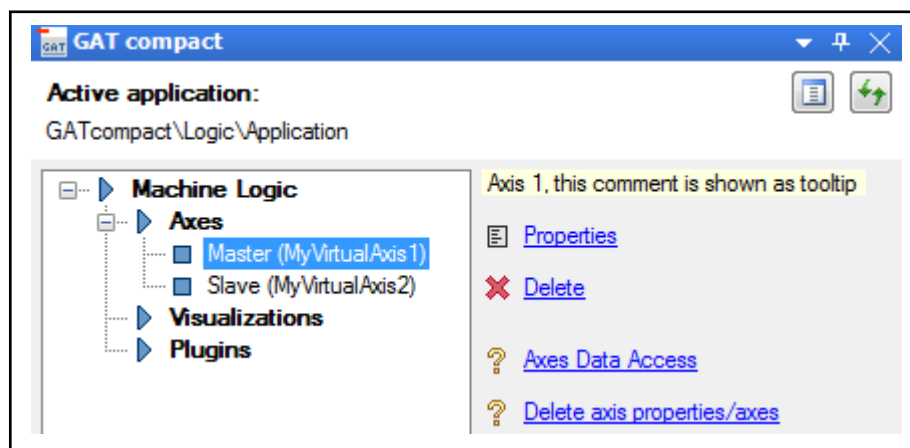


図 8-41: GAT ウィザードを使用した軸の削除/名前変更

8.10.5 運転モードの追加

GATcompact は以下の定義済み運転モードを備えています。

- オフ
- 手動
- 自動

例として、以下のセクションでは、"**CLEANING**" (クリーニング) 運転モード (機械のクリーニング) も実装します。

これを行うためには、以下の手順が必要です。

- "TE_MACHINE_MODE" と入力する (*GATcompact* フォルダ) : **MODE_CLEANING** 要素で拡張する
- *FB "Machine"* :
 1. 移行 "traAuto" をコピーし、名前を "traCleaning" に変更して、"traCleaning" 内で MODE_AUTO を MODE_CLEANING で置き換えます。結果 : (Ctrl= MODE_CLEANING;)
 2. 移行 "traLeaveAuto" をコピーし、名前を "traLeaveCleaning" を変更して、"traLeaveCleaning" 内で MODE_AUTO を MODE_CLEANING で置き換えます。結果 : NOT(Ctrl= MODE_CLEANING;)
 3. "ModeAuto" と平行して "ModeCleaning" 分岐を追加します
 4. 上に挙げられている移行を "ModeCleaning" で使用します
 5. 入力アクション "actModeCleaning_entry()" を "ModeCleaning" で使用します。アクションで、最初の行として以下のコードを追加します : Status:= MODE_CLEANING;
 6. 移行の後で、"ModeAuto" と同じジャンプ ターゲットを追加します。

GATcompact

拡張の結果

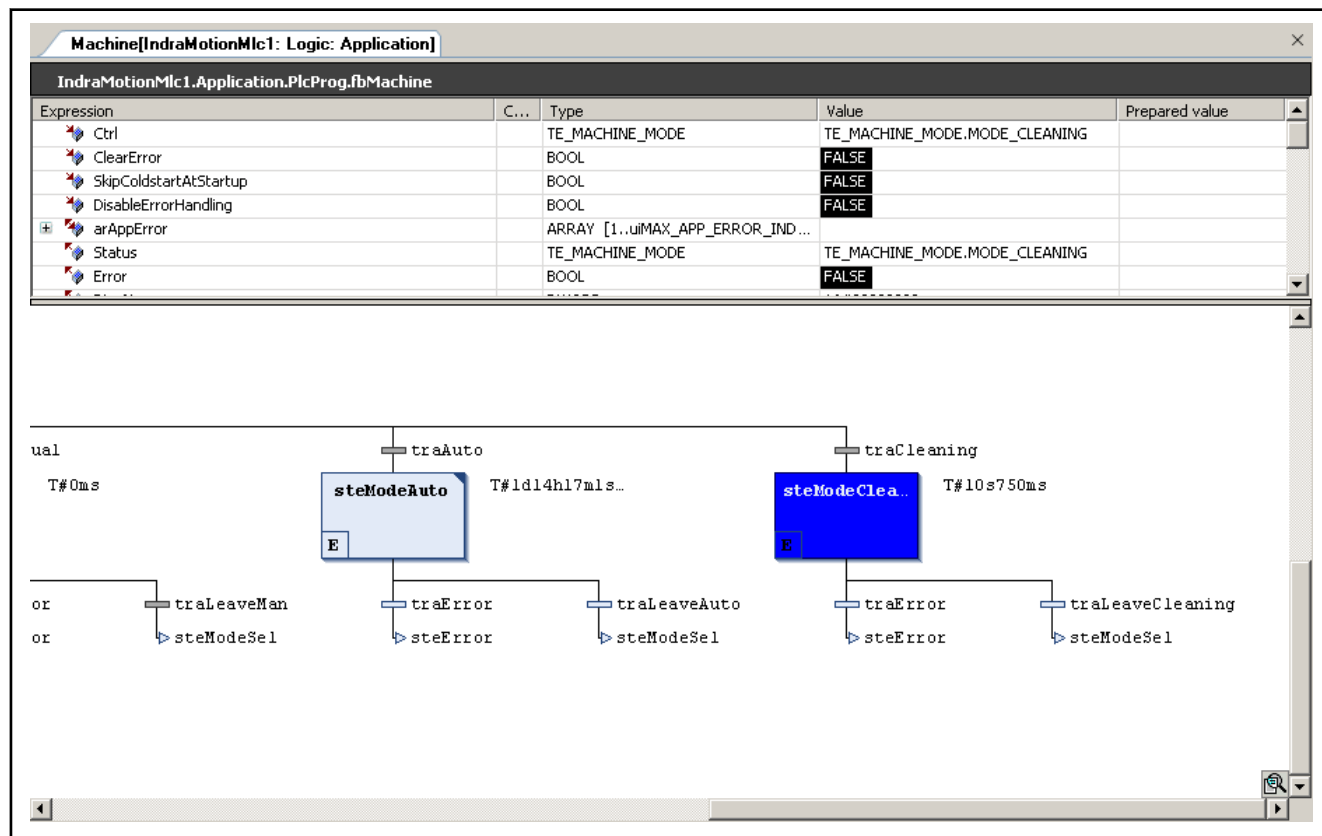


図 8-42: 運転モードを追加した後の結果

必要なモーションを、アクション "actModeCleaning_entry()" またはオプションのアクション "actModeCleaning_active()" に実装できます。

8.10.6 MotionTask での処理

MotionTask での処理の場合、FB "Machine" を拡張してメソッド "mMotionCyclic" を含めることができます。このメソッドより、すべてのファンクションブロック変数（一元的な軸入力ポイント Axes など）にアクセス可能です。

重要な点は、ここで制御する軸が MotionTask で更新されることです (214 ページ "MotionTask の追加" 8.6.2 章 を参照)。

8.10.7 AxisInterface の拡張

GATcompact では、AxisInterface 構造のほぼすべての拡張が可能です。また、追加的な下位構造を追加したり、既存の下位構造を拡張したりすることも可能です。

例として、以下の構造要素がユーザー拡張として AxisCtrl 構造にプログラミングされています。

- SetupMode: AxisCtrl 構造の追加的な下位構造
 - Enable: セットアップ モードを有効にする
 - JogPlus: ジョグ +
 - JogMinus: ジョグ -
 - Vel: ジョグ/参照移動速度
 - Accel: ジョグ/参照移動減速/加速

- Homing: 軸のタイプ/設定によって ModeHoming|SetAbsRef|ModesPosAbs を有効化
- SyncMode: AxisCtrl 構造内の既存の下位構造の拡張
 - AngleOffset: 表のフェーズ オフセット開始 -> この値によりパラメータ A-0-2730 を説明
 - PhaseShift: ギアの前のマスター軸に比例したオフセット (PhaseOffset: ギアの後のマスター軸に比例したオフセット)
 - PhaseShiftVel: PhaseShift を調整するための速度
 - PhaseShiftAcc: PhaseShift を調整するための加速
 - SyncVelocity: フェーズ シフト速度 -> この値によりパラメータ A-0-2790 を説明
 - SyncAcceleration: フェーズ シフト加速 -> この速度はパラメータ A-0-2791 に書き込まれる

例として、以下の構造要素がユーザー拡張として *AxisStatus* 構造にプログラミングされています。

- SetupMode: AxisStatus 構造内の追加的な下位構造
 - EnableAck: Setup (セットアップ) モードが有効
 - Homingack: ホーミングが完了
- SyncMode: AxisStatus 構造内の既存の下位構造の拡張
 - AngleOffsetDone: 表のフェーズ オフセット開始の転送 -> この値によりパラメータ A-0-2730 を説明
 - PhasingMasterDone: PhaseShift が完了
 - SyncVelocityDone: 同期速度の転送 -> パラメータ A-0-2790 は転送済み
 - SyncAccelerationDone: 同期加速の転送 -> パラメータ A-0-2791 は転送済み

これらの拡張のコードは、"GATcompact/AxisInterface" フォルダ内の TE_AxisInterface() ブロックのアクションにあります。関連する構造は、"GATcompact/AxisInterface/Type/Control" および "GATcompact/AxisInterface/Type/Status" フォルダにあります。

ユーザーは、ユーザー独自の拡張を追加できます (これを行うには、[97 ページ "AxisInterface のユーザー拡張" 6.7 章](#) を参照)。

新しいエラー コードが導入された場合、"GAT compact\GAT-Utilities" フォルダにある GatConstGLocal 変数リストの構造 "arAXIS_DIAG_MESSAGES" を拡張することにより、PLC 表示に診断メッセージを割り当てることができます。

8.10.8 Robot Control

協調運動のために軸を制御するには、TechInterface に MB_AxisInterfaceType02 FB を使用する必要があります。このため、GATcompact で以下を調整する必要があります。

- 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によってこの構造を拡張する必要があります。

構造 TE_AXIS_CONTROL_Type01 の TE_AXIS_CONTROL_Type02 への名前変更:

```
//former type: TYPE TE_AXIS_CONTROL_TYPE01:
TYPE TE_AXIS_CONTROL_TYPE02: // rename for RobotControl
STRUCT
```

GATcompact

```

Admin:      TE_AXIS_ADMINISTRATION; (* Administration of the axis | Verwaltung der Achse *)
PosMode:    TE_AXIS_POSITIONING; (* operating mode positioning (absolute, rel. and additive)
          Positionierbetrieb (absolut, relativ und additiv) *)
VelMode:    TE_AXIS_VELOCITY; (* operating mode velocity control | Geschwindigkeitsregelung *)
SyncMode:   TE_AXIS_SYNCHRONISATION; (* operating mode synchronisation (phase, vel, and cam)
          Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvenscheibe) *)

(* add your extensions below here, keep the elements above
  Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)

(* user extensions | Anwender Erweiterungen *)

// Jog/Homing functionality | Tipp/Einrichtfunktionen
SetupMode:  TE_AXIS_SETUP_MODE;

CoordModeCtrl: MB_AXIS_COORDINATED; // inserted for RobotControl

```

```

END_STRUCT
END_TYPE

```

-
- "TE_AXIS_STATUS_Type01" 構造の名前を
"TE_AXIS_STATUS_TYPE02" に変更します

構造 *TE_AXIS_STATUS_Type01* の *TE_AXIS_STATUS_TYPE02* への名前変更:

```

(***** Outputs of the following axis - from motion control to logic control *****)
// former type: TYPE TE_AXIS_STATUS_TYPE01:
TYPE TE_AXIS_STATUS_TYPE02: // Rename for RobotControl
STRUCT
(***** Outputs of the following axis - from motion control to logic control *****)

PtrAxisData: POINTER TO ML_AXISDATA_SM;
(* Pointer to AxisData of this axis | Zeiger auf AxisData dieser Achse
  - Type .PtrAxisData^. for the intellisense menu | Eingabe .PtrAxisData^. fuer Auswahlmenue*)

Admin:    TE_AXIS_ADMIN_STATUS; (* status of the axis | Achs-Status *)

Diag:     TE_AXIS_DIAGNOSIS; (* axis diagnosis information, error, warning....
          Diagnose Informationen der Achse, Fehler, Warnung.... *)

PosMode:  TE_AXIS_POS_STATUS; (* Status operating mode positioning (absolute, rel. and add.)
          Statusmeldungen Positionierbetrieb (absolut, relativ und additiv) *)

VelMode:  TE_AXIS_VEL_STATUS; (* Status operating mode velocity control
          Statusmeldungen Geschwindigkeitsregelung (Freilauf) *)

SyncMode: TE_AXIS_SYNC_STATUS; (* Status operating mode synchronisation (phase, vel, and cam)
          Statusmeldungen Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvensch.)*

(* add your extensions below here, keep the elements above
  Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
(* user extensions | Anwender Erweiterungen *)
(* Jog/Homing functionality | Tipp/Einrichtfunktionen *)
SetupMode: TE_AXIS_SETUP_MODE_STATUS;

// AxisData for HMI (without data type "BIT")
AxisData_HMI: TE_AXIS_DATA_STATUS;

```

```

END_STRUCT
END_TYPE

```

-
- 新しい FB "MB_AxisInterfaceType02" により "TE_Axisinterface" ブロックを拡張する必要があります。 "TE_AxisInterface EXTENDS MB_AxisInterfaceType02"

TE_Axisinterface の拡張:

```

//former: FUNCTION_BLOCK TE_AxisInterface EXTENDS MB_AxisInterfaceType01
FUNCTION_BLOCK TE_AxisInterface EXTENDS MB_AxisInterfaceType02
(*****
*****

```

General Header

Shortdescription : AxisInterface Example

- メソッド "mInitExtensions()" : ファンクションブロック "MB_AxisInterfaceType02" の追加的な "CoordModeCtrl" 入力を初期化する必要があります

CoordModeCtrl の初期化 :

```
SUPER^.mInitExtension(index);

IF NOT bFirstInit THEN
  // User: Assignment of user defined axis interface inputs
  SetupMode REF= arAxisCtrl_gb[index].SetupMode;
  // inserted for RobotControl
  CoordModeCtrl REF= arAxisCtrl_gb[index].CoordModeCtrl;
  // User: Assignment of user defined axis interface outputs
  SetupModeAck REF= arAxisStatus_gb[index].SetupMode;
  AxisData_HMI REF= arAxisStatus_gb[index].AxisData_HMI;

  // =====
  // Do not touch the following lines
  // =====

  // Set the references for the AxisInterface_FB
  AdminCtrlExt REF= arAxisCtrl_gb[index].Admin;
```

- "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります

構造変数の調整 :

```
VAR_GLOBAL
  // Axis interface: command structure: define as RETAIN, to keep the values
  // Note: on the L40.2 control, the RETAIN memory is slower then normal VAR, so define
  // this structure only as RETAIN when necessary
  // arAxisCtrl_gb: ARRAY [1..MAX_AXIS_NUM] OF MB_AXIS_CONTROL;

  // Axis interface: command structure
  //former: arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE01;
  arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE02;
END_VAR

VAR_GLOBAL
  //----- Machine status variables -----

  // Axis interface: status structure
  //former: arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE01;
  arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE02;
  bRemoteOn_gb : BOOL := TRUE; // TRUE: Internal IL visu is active
  // FALSE: PlcProg is active
END_VAR
```



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 はフォルダ "AxisInterface\Type" に含まれています。

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "UserVarGlobal" に含まれています。

キネマティクス インターフェース

軸インターフェースに基づいて、オプションのプラグインとしてキネマティクス インターフェースが利用可能です。キネマティクス インターフェースを使用すると、軸と同じ方法でキネマティクスを操作できます。この目的のために、GAT ウィザードを介してキネマティクス インターフェースをプラグインとして追加します。

キネマティクス インターフェースの資料は、プラグイン設定フォルダにあります。

以下の手順で、GAT ウィザードを介して設定フォルダへのリンクを参照します。

- GAT ウィザードでフォルダ "Plug-ins" を選択する

GATcompact

- ウィザード コントロール パネルに、リンク "Open plug.ins configuration folder" (プラグイン設定フォルダを開く) が表示される

8.11 GATcompact の概略

8.11.1 構造の概要

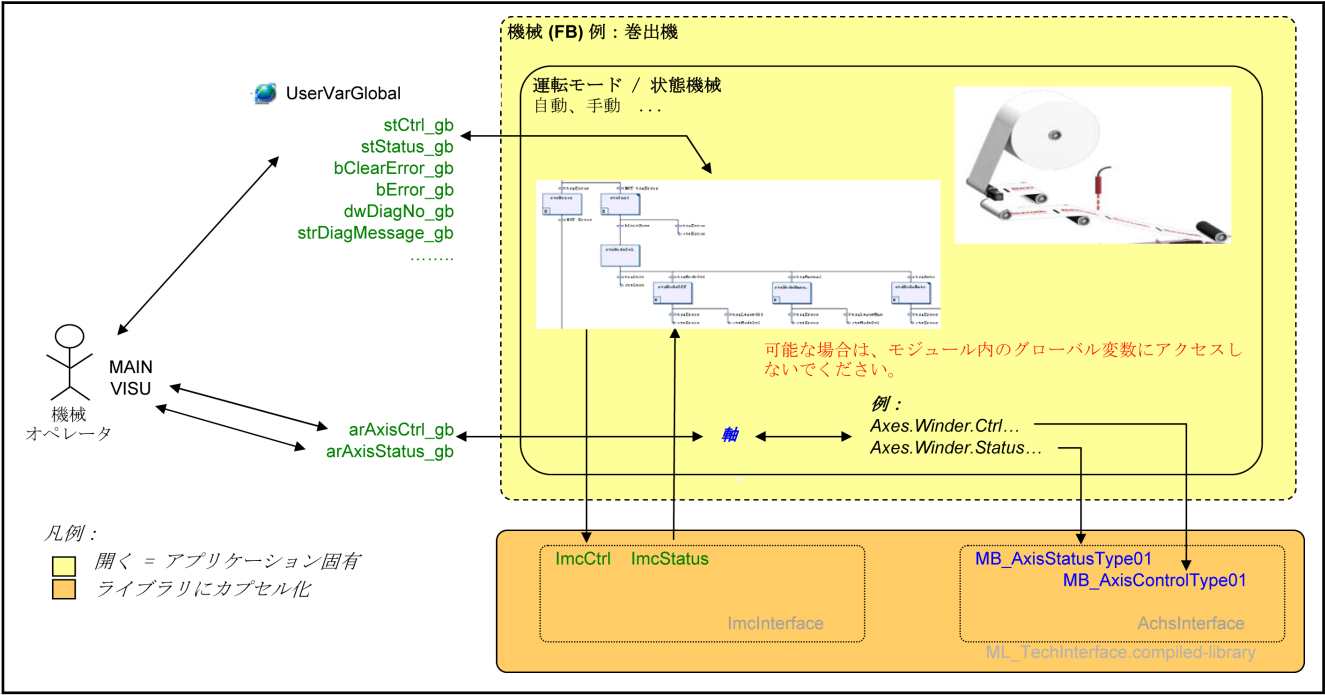


図 8-43: GATcompact の構造の概要

8.11.2 機械の最も重要なグローバル変数の概要

グローバル変数リスト "UserVarGlobal" を参照してください。

変数	説明	アクセス
stCtrl_gb	機械制御 (運転モードの選択、デフォルトの運転モード)	RD/WR
stStatus_gb	機械のステータス (現在の運転モード)	RD
bClearError_gb	エラーのクリア	RD/WR
bDisableErrorHandling_gb	エラーが発生した場合、エラー状態に切り替えない	RD/WR
bSkipColdstartAtStartup_gb	プログラムの起動時、"コールド スタート" (P2 への切替) が実行されない	RD/WR
bError_gb	TRUE はエラーを示します。	RD
dwDiagNo_gb	IndraLogic XLC-/IndraMotion MLC 資料に基づいた診断番号	RD
enErrorID_gb	dwDiagNo_gb = 16#F0FB 0000 (ファンクションブロック エラー) の場合の詳細情報	RD
stErrorIdent_gb	dwDiagNo_gb = 16#F0FB 0000 (ファンクションブロック エラー) の場合の詳細情報	RD

変数	説明	アクセス
strDiagMessage_gb	コントロールまたは軸の診断テキスト	RD
arAppError_gb	ユーザー固有の機械エラー フラグ付きの BOOL の列	RD

RD 読み取り
WR 書き込み

8.11.3 軸の最も重要なグローバル変数の概要

グローバル変数リスト "UserVarGlobal" を参照してください。

変数	説明	アクセス
arAxisCtrl_gb	AxisInterface コマンド データの列	RD/WR
arAxisStatus_gb	AxisInterface ステータス データの列	RD

RD 読み取り
WR 書き込み

8.11.4 機械のメソッドとアクションの概要

アクション/メソッド	呼出	関連タスク	説明/例
actCyclic_active	すべての状態で	PlcTask	安全技術などの総合的な動作
actError_entry	MODE_ERROR 状態への移行中のサイクル用	PlcTask	エラーリアクション、例：ModeAH (電源オンで完全停止) への切替
actInit_entry	MODE_INIT 状態への移行中のサイクル用	PlcTask	MODE_INIT 状態の準備、例：P2 へのフェーズ切替を有効にするためにすべての軸をスイッチオフにする
actInit_active	MODE_INIT 状態でのみ	PlcTask	パラメータ化モード P2 への切替、P2 のパラメータの説明である可能性あり。MODE_INIT を終了すると、運転モード (bb) に切り替わる、bb に達する際のパラメータの説明である可能性あり。
actModeOff_entry	MODE_OFF 状態への移行中のサイクル用	PlcTask	MODE_OFF 状態の準備、通常はすべての軸のスイッチオフ (ModeAb)。
actModeManual_entry	MODE_MANUAL 状態への移行中のサイクル用	PlcTask	MODE_MANUAL 状態の準備、通常はすべての軸の停止 (ModeAH)。
actModeAuto_entry	MODE_AUTO 状態への移行中のサイクル用	PlcTask	MODE_AUTO 状態の準備、例：すべてのスレーブ軸の同期モードの有効化 (ModeSyncVel など) およびマスター軸の有効化 (ModeVel など)
actModeAuto_active	MODE_AUTO 状態でのみ	PlcTask	自動モードでのモニタリング、プロセスコントローラ機能ブロックの呼出など
FB_init	PLC プログラムの起動/リセット前に 1 回	特定のタスク割当なし	内部システムの使用は、変更できません

8.11.5 機械内の軸へのアクセス



必ず Axes.<axisname> 経由でアクセスします。

GATcompact

例：軸へのアクセス

```
// 速度制御モードでのコマンド軸  
Axes.MyAxis1.Ctrl.Admin._OpMode.en := ModeVel;  
  
/ If axis has reached the velocity  
IF (Axes.MyAxis1.Status.PtrAxisData^.Axis_InVelocity = TRUE) THEN;
```

9 sercos インターフェースなしの IndraLogic XLC 用 GATcompact

9.1 GATcompact の概要

一般情報 sercos インターフェースなしの IndraLogic XLC 用汎用アプリケーション テンプレート (GATcompact は、IEC 61131-3 に基づいたオープン サンプル プロジェクトです。IndraWorks エンジニアリング フレームワークにより提供されており、ロードと起動にはウィザードが使用できます。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact には以下の部分が含まれています。

- SFC 運転モードを備えた Machine FB
- エラー処理
- 試運転表示
- 準備されている機械表示 (HMI)

9.2 GATcompact の最初の手順

初期インポート 以下の手順で GATcompact をコントロール プロジェクトに追加します。

- IndraLogic XLC コントロールをプロジェクトに追加します
- ツールバーにある GAT ボタン  をクリックします
- GATcompact を選択し、**Finish** (完了) をクリックします。初期インポートが実行されます。
- PLC アプリケーションにログインします (コントロール ハードウェアが利用可能な場合)
- PLC アプリケーションを起動します (コントロール ハードウェアが利用可能な場合)
- 表示 "System_Overview_header" を開きます 運転モードを選択し、エラーおよび表示コントロール診断を削除します
- PLC からログアウトし、下記の説明に従ってアプリケーション コードを追加します

アプリケーション コードの実装

初期インポート後、完全に準備の整った GATcompact プログラム フレームワークが利用可能です。

ファンクションブロック **Machine** の準備されたポイントでプログラム コードを入力し、ファンクションブロックの入力と出力を拡張します。

- 手順 **steCyclic** : 継続的かつ周期的に処理されるアプリケーション コードを入力します
- 手順 **steModeOff**、**steModeManual** および **steModeAuto** : 選択された運転モードでのみ周期的に処理されるアプリケーション コードを入力します
- 手順 **steError** : エラーが発生した場合に周期的に処理されるアプリケーション コードを入力します
- 手順 **stelnit** : 初期化にあたって周期的に処理されるアプリケーション コードを入力します
- 機械 (FB) の宣言に、その他の入力、出力およびローカル変数を入力します

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

9.3 初期化

初期化はコントロールのスイッチがオンになった後で実行されます。手順 **steInit** がアクティブになり、対応するコードが処理されます。初期化は以下の 2 つの手順で構成されます。

- コールド スタート (コントロールはパラメータ化モード)
- ウォーム スタート (コントロールは操作 bb の準備が整っている)

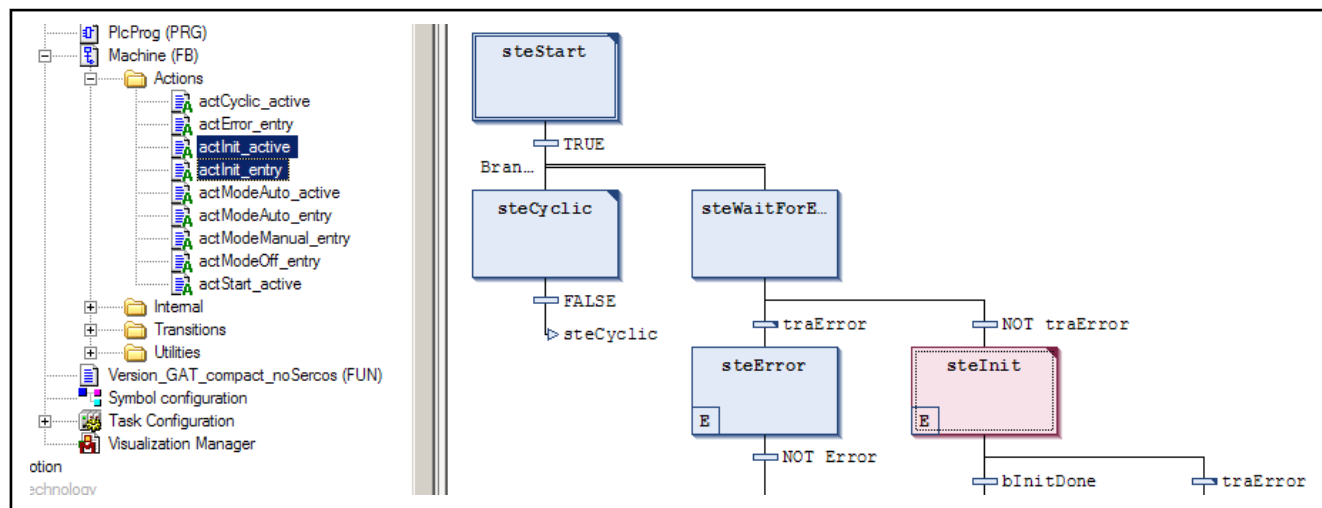


図 9-1: 機械 SFC の初期化手順

エラーが発生していない限り、"steInit" 手順で初期化が実行されます。この手順の開始時に入力アクション "steInit_entry" が 1 回処理されます。

初期化中にエラーが発生した場合、スイッチがエラー ステータスに変更されます。エラーをクリアした後、初期化は中断された位置から続行されます。ビット "bInitDone" が設定されると、プロセスは初期化を終了し、次の手順に進みます。

"actInit_active" 手順では以下のプロセスが実行されます。

1. コントロールがパラメータ化モードに切り替えられます (ライン 12)。
2. パラメータ化モードに切り替わった後、"コールド スタート" の状態に達します。この状態で、運転モードで保護されるパラメータを記述します (ライン 20~25)。
3. 運転モード <> "Init" が指定されるまで、"コールド スタート" の状態が有効なままとなります (ライン 26)。パラメータの書き込みに複数のサイクルが必要な場合は、IF クエリの続行を完了させます。
4. コントロールが運転モード (bb) に切り替えられます (ライン 33)。
5. 運転モードはコントロールによって応答されています。変数またはパラメータを初期化します (ライン 36~39)。
"bInitDone" が "TRUE" に設定され、選択された運転モードが有効になると、初期化が終了します。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

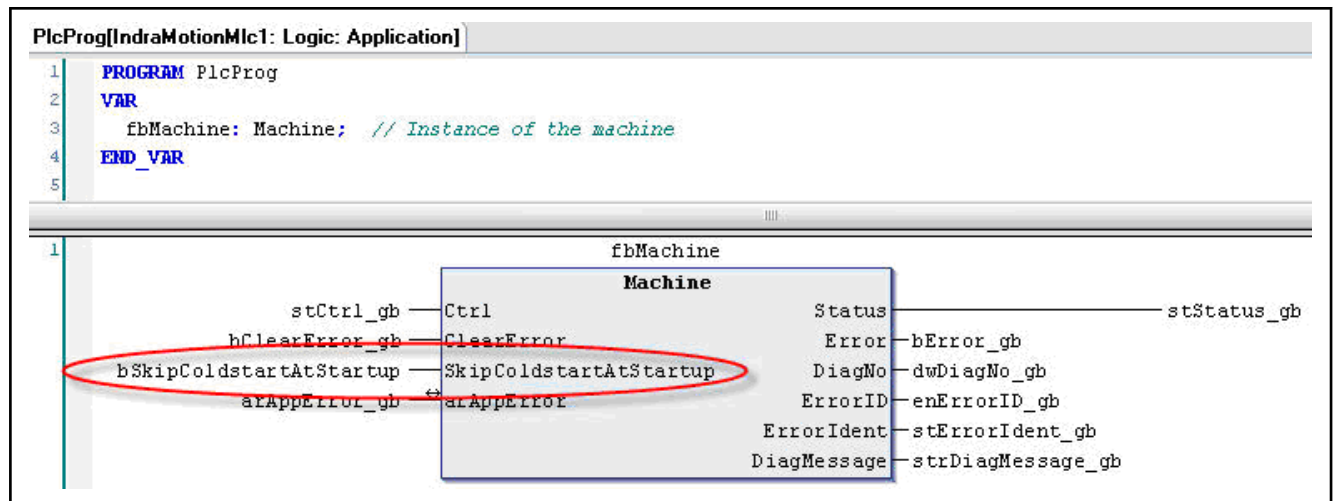


図 9-2: "Machine" FB での "SkipColdStartAtStartup" 入力

PLC を初めて起動したときに初期化が完了しなかった場合、この設定は "bSkipColdstartAtStartup" ビットを使用して行うことができます。プロセスは "ウォーム スタート" 状態に切り替わり、"コールド スタート" は処理されません。この設定により、出力を上げる際に時間を節約できます。ただし、"コールド スタート" 状態で処理されるアクション (パラメータの書き込みなど) が不要であることを確認してください。

9.4 運転モード

初期化が正常に完了すると、シーケンスは以下の 3 つの運転モードのいずれかに入ります (下の図を参照)。

- ModeOff
- ModeManual (機械のセットアップ)
- ModeAuto (周期的な自動運転)

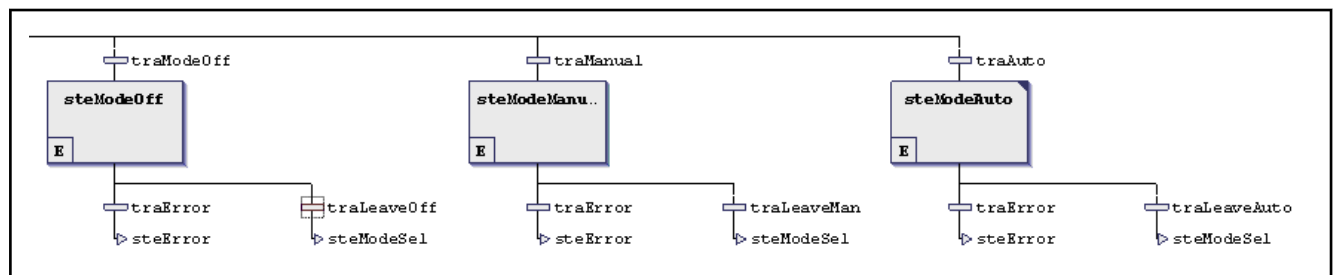


図 9-3: 運転モード

(244 ページ "初期化" 9.3 章 参照)

GATcompact では、初期の機械起動の後のターゲット運転モードとして "ModeManual" が予め設定されています。

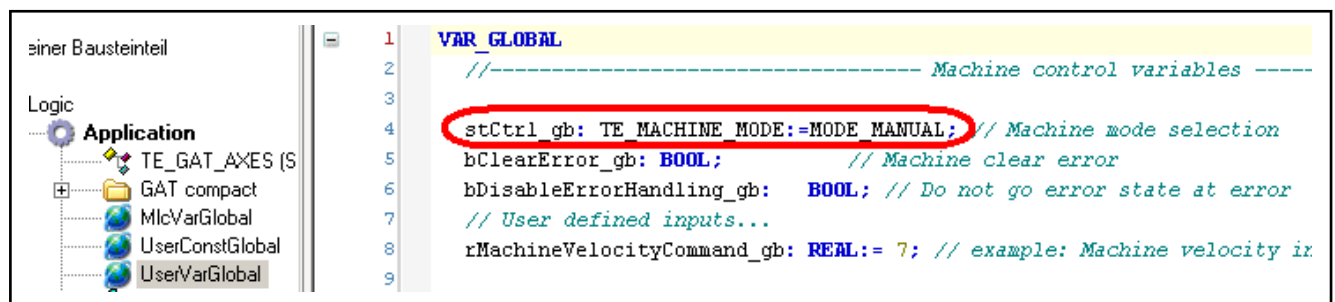


図 9-4: 初期の機械起動後のターゲット運転モードの設定

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

入力/アクティブ アクション

運転モードが有効になると、特別な"入力"アクションが 1 回のみ実行されます。下の表は、各運転モードを有効にする方法および運転モードで処理される内容など、このアクションの概要を示したものです。

	起動の条件	アクション (当初、1 回のみ実行される)	周期的に実行されるアクション
ModeOff	Ctrl=MODE_OFF クエリが traModeOff および traLeaveOff で実行される	actModeOff_entry ステータスを "MODE_OFF" に設定	予め定義済みのアクションなし
ModeManual (機械のセットアップ)	Ctrl=MODE_MANUAL クエリが traManual および traLeaveManual で実行される	actModeManual_entry ステータスを "MODE_MANUAL" に設定	予め定義済みのアクションなし 一般的なアクティビティ： 機械のセットアップおよび調整
ModeAuto (周期的な自動運転)	Ctrl=MODE_AUTO クエリが traAuto および traLeaveAuto で実行される	actModeAuto_entry ステータスを "MODE_AUTO" に設定	actModeAuto_active 自動プロセスの処理など

運転モードを追加するには、[258 ページ "運転モードの追加" 9.10.1 章](#) を参照してください。

9.5 周期的処理

GATcompact では、周期的なプログラム部分 actCyclic でコントロール インターフェースおよび中央エラー処理が呼び出されます。

9.6 タスク処理

9.6.1 GATcompact のタスク設定

サンプル アプリケーション GATcompact は、意図的に PLC タスク (PlcTask) に限定されています。

以下のセクションでは、GATcompact の PlcTask のデフォルト設定を示します。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
PlcTask	5	T#10ms	T#20ms	3

表 9-1:

必要に応じて、GATcompact に他のタスクを追加できます。

9.6.2 BackgroundTask の追加

PlcTask のランタイムが長すぎる場合は、タイム クリティカルではないタスクをバックグラウンド タスクに移動して、PlcTask の負荷を減らすことができます。ただし、追加的なタスク Switch PlcTask <> BackgroundTask によって、より大きな負荷がシステム全体にかかることに留意してください ([36 ページ "BackgroundTask の一般的なタスク" 章](#) も参照)。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
BackgroundTask	~6	T#40ms	-	-

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

BackgroundTask を追加するためには、以下の手順が必要となります。

- タスク設定、"BackgroundTask" の追加。設定については、[247 ページ "BackgroundTask の設定" 9-5 図](#) を参照してください。
- POU "BackgroundProg" を追加し、BackgroundTask の設定で "BackgroundProg" を呼び出します
- "Machine" ファンクションブロックで "mBackground" メソッドを追加します。それを "BackgroundProg" で呼び出します。"mBackground" メソッドでは、メソッド内のすべての変数が利用可能です。

図 9-5: *BackgroundTask* の設定

9.7 エラー処理

9.7.1 エラーの検出および診断

プログラム テンプレート GATcompact には、**中心的な位置**でエラーを認識、処理、クリアするために予め定義済みの機能が含まれています。さまざまな発生元からのエラー (コントロール エラー、アプリケーション エラー、ファンクションブロック エラーなど) が、機械全体のファンクションブロック "Machine" の "mErrorHandling" アクションによって自動的に集められ、ファンクションブロック "Machine" の診断出力を使用して利用可能になります。

複数のエラーが同時に発生した場合は、最も深刻なものが表示されます。同じ深刻度のエラーが複数、同時に発生した場合は、最後に検出されたものが出力されます。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

FB "Machine" には、以下に説明されている診断目的の出力が含まれていません。

要素	タイプ	説明
エラー	BOOL	TRUE はエラーを示します
DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの : 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) (xxxx = ユーザーが定義可能) 16#F0FB 0000: ファンクションブロック エラー (例: TechFB でのエラー) -> 詳細については、ErrorID / ErrorIdent を参照 16#F0AA 0000: GAT 内部エラー -> 詳細については、ErrorID / ErrorIdent を参照
ErrorID	ERROR_CODE	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
ErrorIdent	ERROR_STRUCT	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
DiagMessage	STRING(60)	コントロール エラーの診断テキストを含む

表 9-2:

DiagNo 出力は、HMI での使用のみを目的としています。エラー テキストには、任意の言語で数値情報が割り当てられます。

DiagMessage、**ErrorID** および **ErrorIdent** は、IndraLogic 表示の使用など、プログラマーによる試運転の際に主に表示される追加情報を提供します。

アプリケーション固有のエラーに関する注意

メソッド "mSetAppError(Number)" を呼び出すことにより、アプリケーション エラー (例: 安全扉が開いている) を表示できます。メソッド入力で伝達される"番号"は、アプリケーション エラー番号に対応します。

アプリケーション 番号が表示された場合、"Machine" FB の中央診断システムは、表示されたユーザー エラーの周期的なモニタリングを行い、アプリケーション エラー DiagNo = Error 16#F0AE xxxx (xxxx = アプリケーション エラー番号) を生成します。

グローバル変数リスト "UserConstGlobal" で、アプリケーション エラーに診断テキストを割り当てることができ、PLC 表示にそのテキストが表示されます。

uiMAX_APP_ERROR_INDEX の範囲 1 にある番号は、機械でのアプリケーション エラーについて有効です。定数 "uiMAX_APP_ERROR_INDEX" の値はデフォルトにより 64 に設定されています。必要な場合は、"Application" フォルダにあるグローバル変数リスト "UserConstGlobal" でこの定数を調整できます。

アプリケーション エラーを表示するためのコード例:

```
// ユーザー固有のエラーを発行するための推奨コード (例)
IF bDoor10open THEN
    mSetAppError(1);
END_IF
```

深刻なエラーに関する注意

コントロール診断システムでは、10 段階の"深刻度"が定義されています (「IndraLogic XLC-/IndraMotion MLC diagnostic system」 (IndraLogic XLC-/IndraMotion MLC 診断システム) を参照)。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

深刻度はエラーリアクションに影響を及ぼします。

- DiagNo = 16#F0xx xxxx は最も低い深刻度です (例：アプリケーションエラー、ファンクションブロックエラー)
- DiagNo = 16#F9xx xxxx は最も高い深刻度です (致命的なシステムエラー)

複数のエラーが同時に発生した場合は、深刻度が最も高いエラーが表示されます (41 ページ "エラーリアクション" 5.6.4 章を参照)。

ファンクションブロックエラーに関する注意

呼び出されたファンクションブロック (技術 FB など) でエラーが発生した場合は、メソッド "mSetFBError(ErrorID, ErrorIdent)" を呼び出して、エラーを引き起こしたファンクションブロックの "ErrorID" と "ErrorIdent" 出力をメソッドに転送することをお勧めします。これにより、エラーが "Machine" FB の診断システムに確実に転送され、"ErrorID" と "ErrorIdent" が "Machine" FB の診断出力で利用可能になります。

ファンクションブロックエラーを表示するためのコード例：

```
// 使用した FB がエラーを発行する場合の推奨コード (例)
IF fbCrossCutter.Error THEN
    mSetFBError(ErrorID:=fbCrossCutter.ErrorID,
                ErrorIdent:=fbCrossCutter.ErrorIdent);
END_IF
```

9.7.2 エラーリアクション

エラーリアクションは、コントロールシステムにおけるエラーによる損傷を可能な限り避けるように設計されています。深刻度によって、制御ファームウェアが自動的にエラーリアクションを実行します (41 ページ "エラーリアクション" 5.6.4 章を参照)。

深刻度が第2段階未満のエラー (アプリケーションエラー、ファンクションブロックエラーなど) の場合、制御ファームウェアによる自動エラーリアクションはありません。

このため、GATcompact には、ユーザー固有のエラーリアクションを開始するエラー検出機能が含まれています。

通常の GAT エラーリアクション

デフォルト設定では、GATcompact エラーリアクションは有効になっています。エラーが発生すると、選択されている運転モードが中断され (例外：初期化)、エラーリアクションが実行されます。エラーリアクションは、ファンクションブロック "Machine" の "actError_entry" アクションでプログラミングされ、アプリケーションに適合させることができます。

通常の GAT エラーリアクションをオフにする

特別な場合、通常の GAT エラーリアクションをオフにすることができます。これを行うには、変数宣言を通じて変数 "bDisableErrorHandling_gb" を以下のとおり変更します。

- グローバル変数リスト："UserVarGlobal"
- 宣言を以下のとおり変更します："bDisableErrorHandling_gb: BOOL := TRUE;"

エラーリアクションがオフになっても、現在の運転モードは中断されません。しかし、エラーは継続して表示され、クリアすることも可能です。

初期化中の GAT エラーリアクション

デフォルト設定では、初期化中、GATcompact エラーリアクションは無効になっています。しかし、エラーは継続して表示され、クリアすることも可能です。

特別な場合、初期化中に GAT エラーリアクションをオンにすることができます。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

これを行うには、変数宣言を通じて変数 `"bEnableErrorReactionInit_gb"` を以下のとおり変更します。

- グローバル変数リスト: `"UserVarGlobal"`
- 宣言を以下のとおり変更します: `"bEnableErrorReactionInit_gb: BOOL := TRUE;"`

エラーリアクションが有効な場合、エラーが発生すると初期化が中断されます。その後、プログラミングされたエラーリアクションが実行されます。エラーリアクションは、**"Machine"** ファンクションブロックの `"actError_entry"` アクションでプログラミングされます。



初期化中のエラーリアクションは、通常の GAT エラーリアクションが有効な場合 (`bDisableErrorHandling_gb = FALSE`) のみ実行されます。

9.7.3 エラーのクリア

一元的なエラーのクリアは、**"Machine"** FB の `"ClearError"` 入力の正のエッジにより自動的に実行されます。システム全体を問題のない状態にするために、ファンクションブロックにより、すべての下位コンポーネント (例: `ImcInterface1)` ...) にコマンドが転送されます。エラーがクリアされ、エラーリアクションのために運転モードを終了したプロセスがクリアされると、有効な、選択されている運転モードが処理を続行します。

9.8 表示

9.8.1 起動中の支援のための内部表示

一般情報

GATcompact には、試運転中の支援および拡張された診断機能を提供する、予め定義済みの IndraLogic 表示オブジェクトが含まれています。

System_Overview

表示要素 `"System_Overview"` は、運転モードの選択、診断およびエラーの確認のための試運転表示です。

9.8.2 独立した HMI による外部表示 (WinStudio 表示)

一般情報

GATcompact では予め定義済みの WinStudio 表示が利用できます。ウィザードを使用して、この表示をプロジェクトに追加できます。



WinStudio を編集するには、少なくとも以下の開発環境のライセンス (またはそれ以上) が必要です。

WinStudio 開発ライセンス、変数最大 1,500 個 (Mat.No. R911323596)

VEP でプログラムを実行するには、以下のランタイムライセンスが必要です。

オペレーティングシステムとして Windows CE を使用しているデバイス用の WinStudio ランタイムライセンス、変数最大 1,500 個 (Mat. No. R911323620)

WinStudio 表示の追加

¹⁾ *IndraMotionControl* インターフェース = 制御へのインターフェースを備えた IndraLogic ライブラリ。例: エラーをクリア

sercos インターフェースなしの IndraLogic XLC 用 GATcompact



WinStudio 表示をプロジェクトに追加するためには、対応する表示デバイスがプロジェクトになければなりません。

"ドラッグ&ドロップ"を使用して、デバイス ライブラリからプロジェクト ツリーに VEP を追加し、以下のとおり設定します。

- デバイス タイプ：VEPxx (CE)
- アプリケーション タイプ：WinStudio アプリケーション

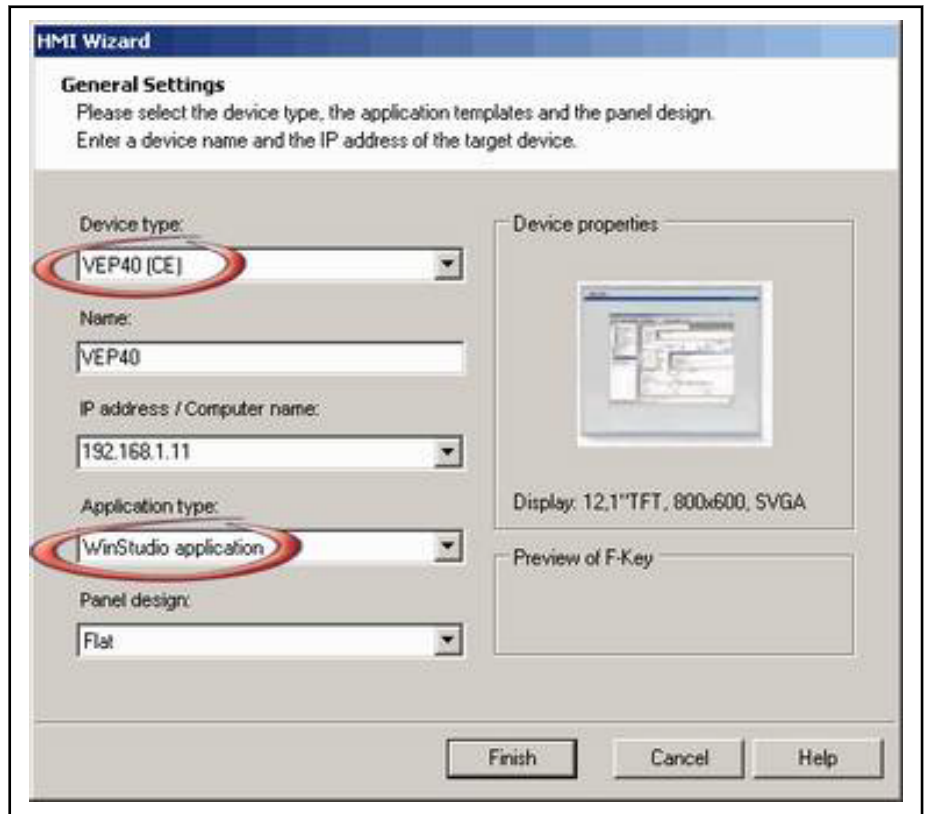


図 9-6: 表示デバイスの追加

対応する表示デバイスがある場合、下記のリンクとウィザードを使用して WinStudio 表示を追加します。

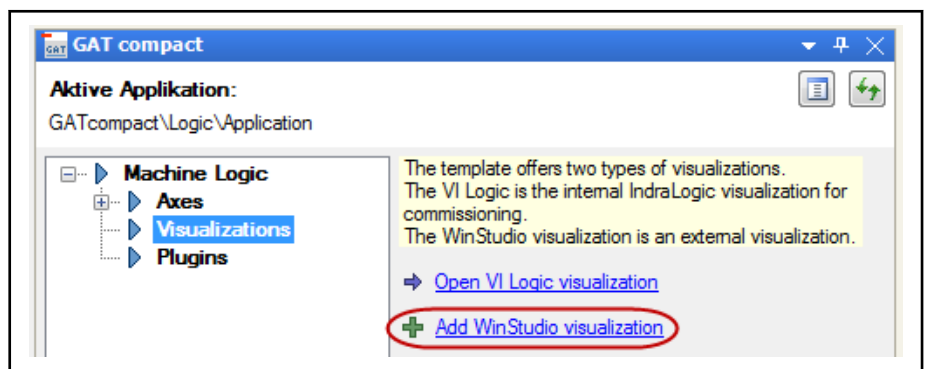


図 9-7: WinStudio 表示の追加

各設定に対応するプロジェクトの表示デバイスはすべて下記のウィザード ダイアログに含まれています。

対応する表示デバイスを選択した後で、**Apply** (適用) をクリックして WinStudio 表示を追加します。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

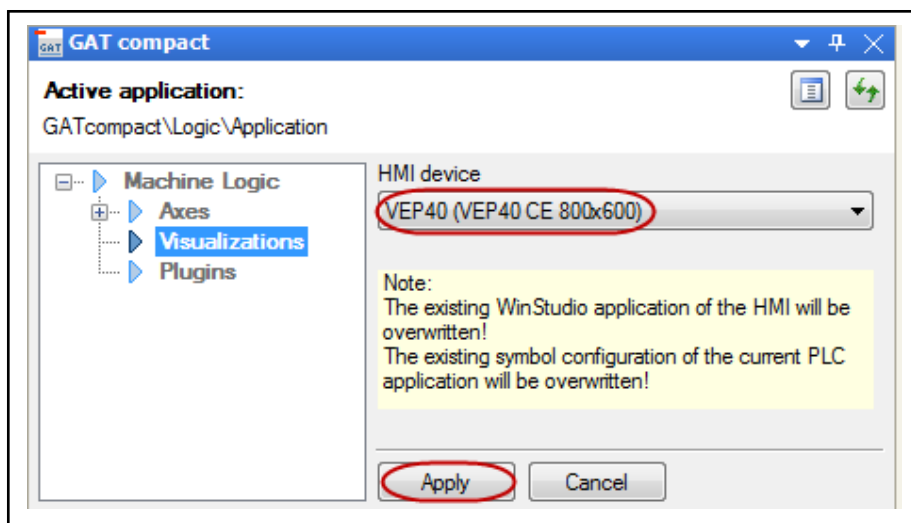


図 9-8: 表示デバイスの選択 – WinStudio 表示の挿入



WinStudio 表示が追加されると、コントロールの記号設定とともに表示デバイスの既存の表示データが上書きされます。

表示が追加された後で、表示データを編集できます。

コントロールのアドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

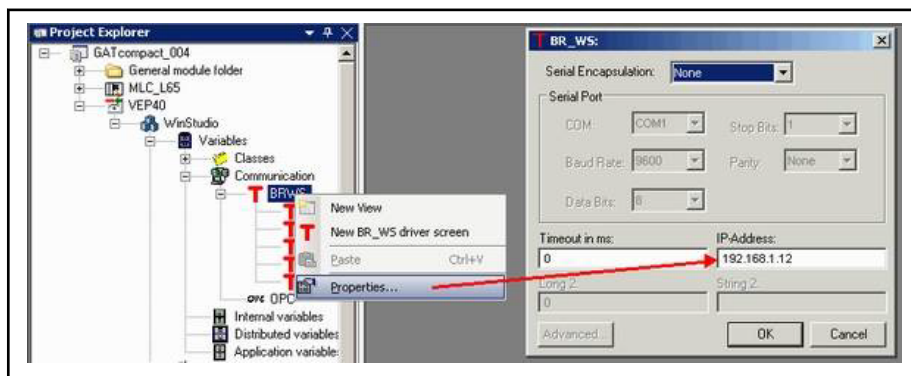


図 9-9: コントロールの IP アドレス設定

ここで、WinStudio ノードのコンテキスト メニューにある  "Start application" (アプリケーションを起動) を使用して WinStudio 表示のエミュレーションを開始します。

ダイアログ構造

操作画面は以下の 3 つのセクションに分かれています。

- ステータス情報があるヘッダー
- メニュー関係の表示と機能キーがあるメイン画面
- ユーザー マップの切替を行うためのキーがあるフッター

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

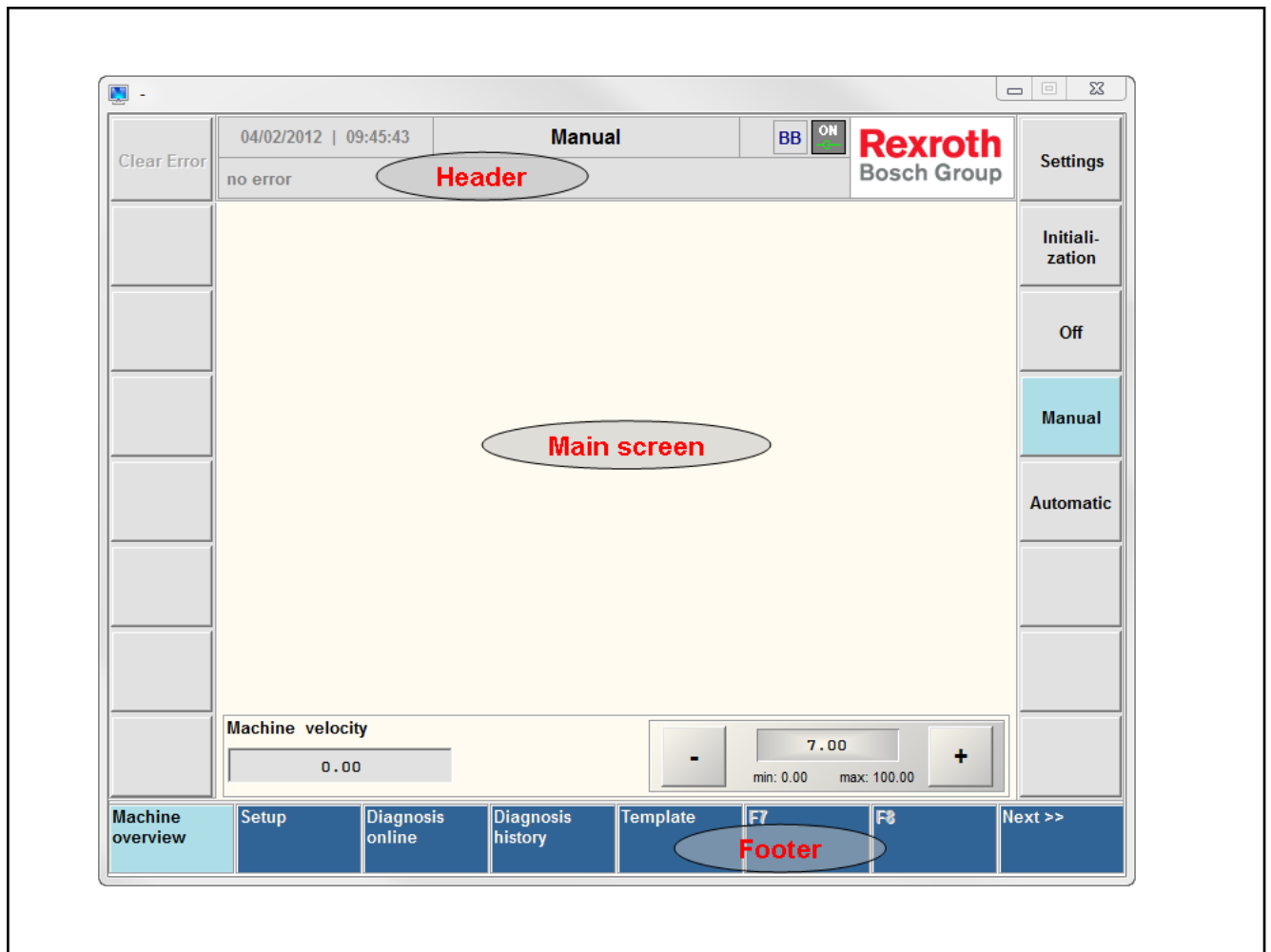


図 9-10: WinStudio 表示

ヘッダーには以下が含まれています：

- 運転モードの表示
- エラーをクリアしたり "Settings" (設定) ダイアログ (言語の切替、数値表示の小数点位置の設定などのため) を開いたりするためのグローバル機能キー
- 現在の機械診断の表示
- 通信ステータスの表示

メイン画面には、操作画面 (運転モードの切替など) に関連する表示と機能キーが含まれています。

フッターには、ユーザー マップを切り替えるためのキーが含まれています。

表示の拡張

操作画面の追加

表示を拡張するには、既存の "テンプレート" 画面をコピーして新しい名前を付けて挿入することにより、新しいユーザー マップを作成できます。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

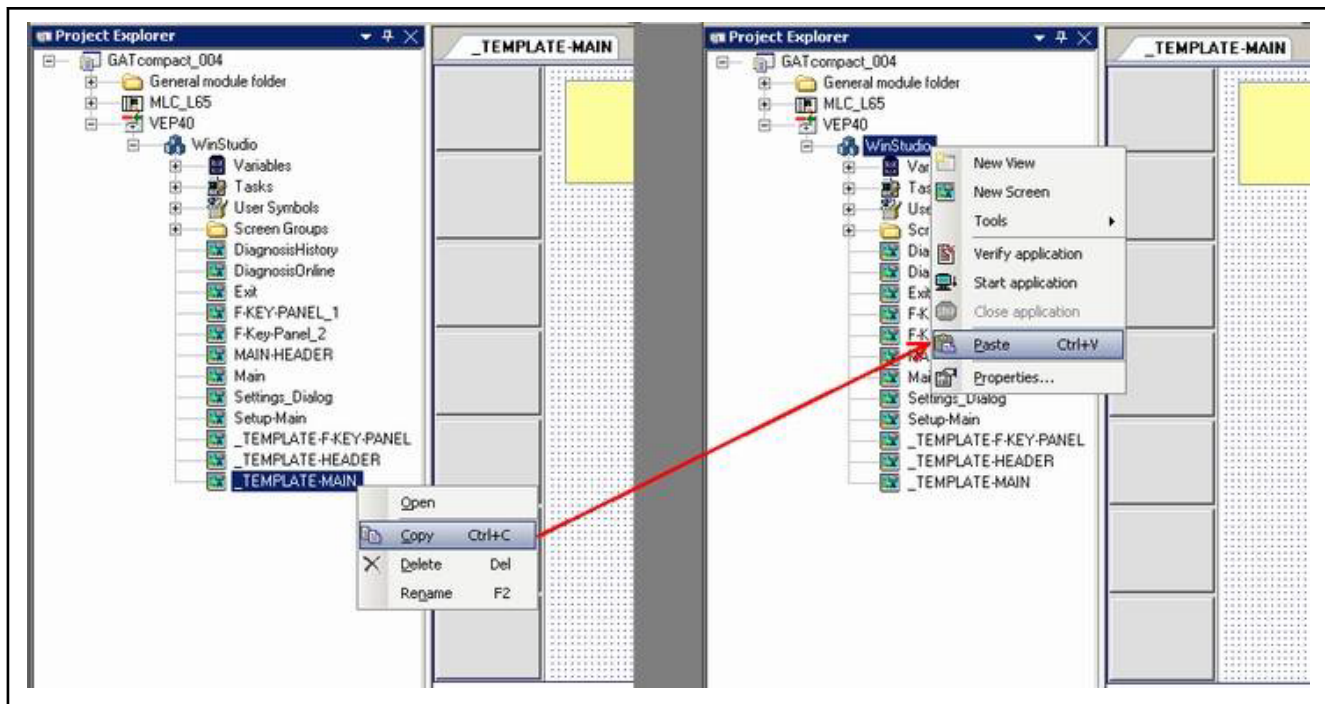


図 9-11: 操作画面の複製

フッターの拡張

ユーザー マップを切り替えるために使用されるキーは、パラメータ化が可能なグラフィック オブジェクト (リンクされた記号) として保存されます。キーには以下の転送パラメータがあります。

- Caption -> キー ラベル
- Disable -> キー機能をブロックするための入力 (グレー表示のラベル)
- ScreenName -> 操作画面を開くための画面名 (または画面グループ)
- ScreenNumber -> 画面に割り当てられる番号 (それぞれのキーを強調表示するため)

通信接続

表示とコントロール間の通信は、BR-WS ドライバを使用して記号的に行われます。これを行うには、表示する PLC 変数の記号エントリを生成します (47 ページ "通信設定" 章 を参照)

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

予め定義済みの WinStudio 表示に必要な記号設定には、以下のエントリが含まれます。

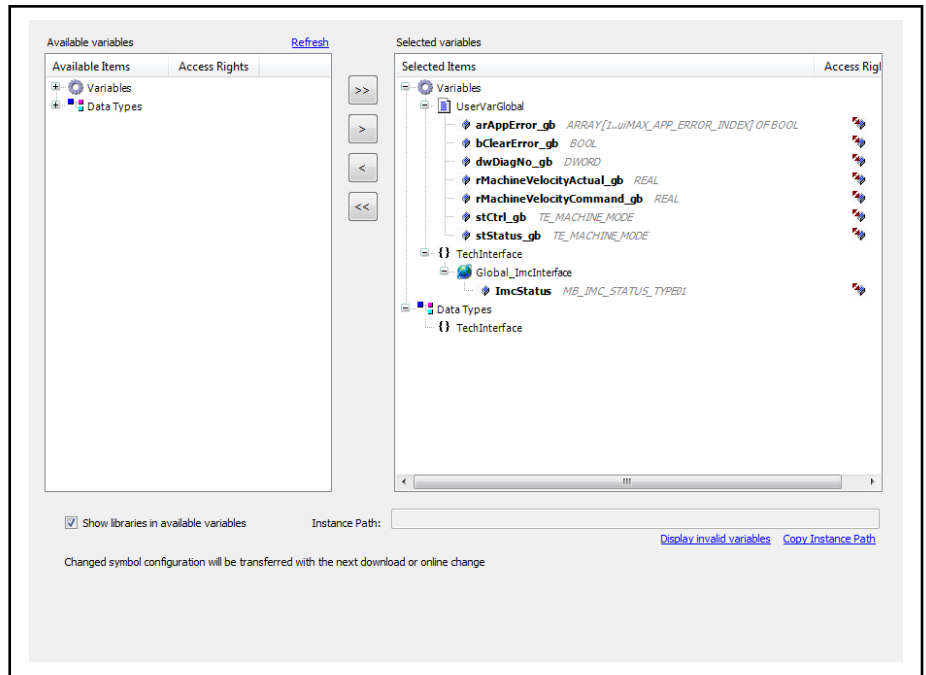


図 9-12: 予め定義済みの WinStudio 表示の記号設定



ライブラリから取得したグローバル変数リストの変数の記号エントリは、各ライブラリのバージョンが変更された後で記号設定に挿入しなおす必要があります。

コントロールの IP アドレス設定：

コントロール アドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

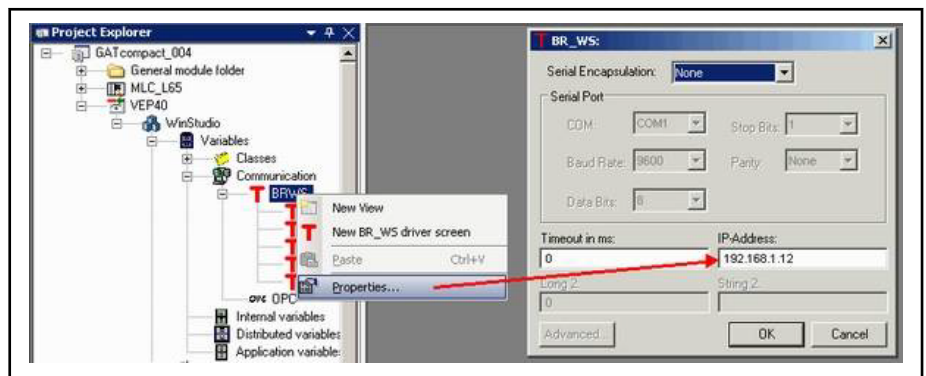


図 9-13: コントロールの IP アドレス設定

通信を構成するために、変数がさまざまなドライバシートに配置されています。

1. **高速で周期的に読み取る**：これらの変数の値は短い間隔で周期的に読み取られます (診断番号、軸の位置など)
2. **1 回のみ読み取る**：これらの変数は表示が開始された後で 1 回のみ読み取られます (軸の名前など)
3. **フェーズの切替後に読み取る**：これらの変数はフェーズの切替後に 1 回のみ読み取られます (軸がアクティブになった場合など)

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

4. **周期的に読み書き**：これらの変数の値は、表示により読み書きの両方が行われます (機械の速度、セットアップ モードの軸の速度など)
5. **書き込みのみ**：これらの変数は、表示を通じて読み取りのみが行われます (エラー のクリアの信号を発信する...)

ドライバ シートを拡張するには、BR-WS ブラウザでオフライン記号ファイル (*.XML) を検索し、対応する変数を選択できます。

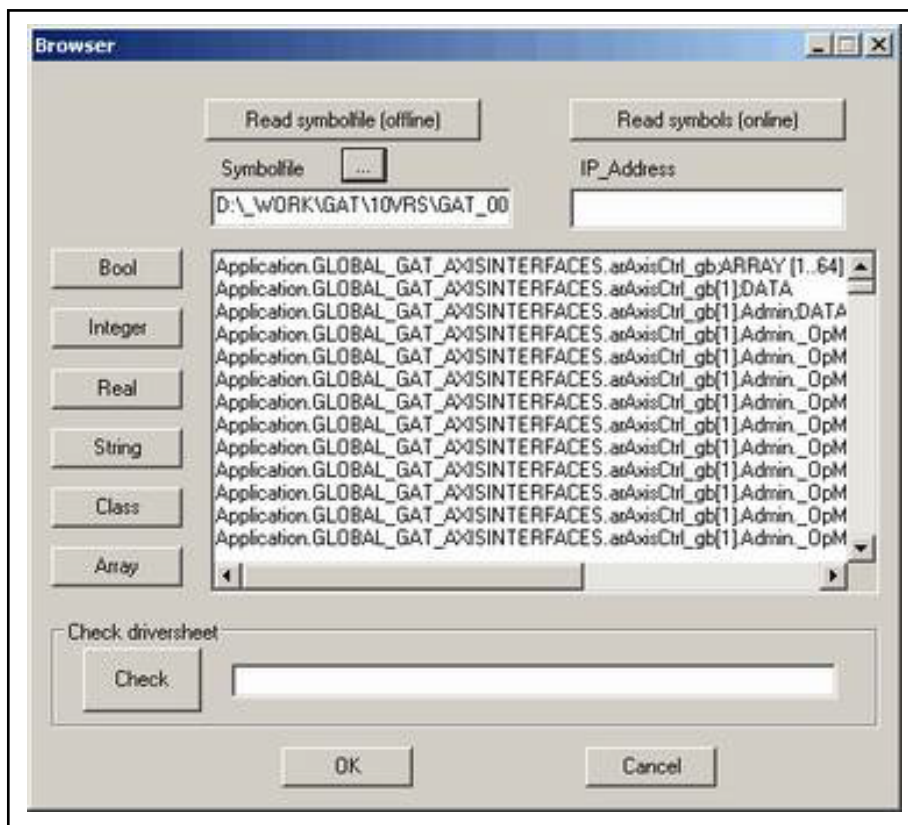


図 9-14: ドライバシートの拡張

複数言語機能

表示の複数言語機能は、IndraWorks への翻訳機能の組み込みにより実現されました。

コントロールおよびドライブの診断の翻訳データ、ならびに使用される表示テキストは、すでに準備されています。翻訳データは IndraWorks で変更や修正が可能です。

ドライブ診断はドイツ語、英語、フランス語、スペイン語、トルコ語で含まれています。

他のすべてのテキストはドイツ語と英語です。

9.9 GATcompact のインターフェース

9.9.1 機械データ

グローバル変数リスト "UserVarGlobal" を参照してください。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

I/O タイプ	名称	データの種類	説明
I	stCtrl_gb	TE_MACHINE_MODE_CTRL	機械運転モードのコマンド
I	bSkipColdstartAtStartup_gb	BOOL	GAT ^{compact} を起動するとき、アプリケーションに必要なパラメータ化を実行するために、パラメータ化モード (P2) に切り替えられます。アプリケーションに P2 でのパラメータ化が不要な場合、このフェーズ切替は TRUE で省略できます。
O	stStatus_gb	TE_MACHINE_MODE_STAT US	現在の機械運転モードに関するフィードバック

I
O書き込み用入力
読み取り用出力

9.9.2 エラー処理用データ

グローバル変数リスト "UserVarGlobal" を参照してください。

I/O タイプ	名称	データの種類	説明
I	bClearError_gb	BOOL	TRUE = グローバル クリア エラー
I	bDisableErrorHandling_gb	BOOL	TRUE = エラーが発生した場合、プロセスは "ERROR" 状態に切り替わりません。エラー表示は続行され、エラーのクリアは可能です。
I	bEnableErrorReactionInit_gb	BOOL	TRUE = 初期化中にエラーが発生した場合、bDisableErrorHandling_gb = FALSE であれば、プロセスは "エラー" 状態に切り替わります
I	bLogError_gb	BOOL	TRUE = GAT、ファンクションブロックおよびアプリケーション固有のエラーがコントロール ディスプレイに表示され、コントロールのエラーと診断のメモリに保存されます
O	bError_gb	BOOL	TRUE = グローバル エラーが発生しました
O	dwDiagNo_gb	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた診断番号: DiagNo = 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー
O	enErrorID_gb	ERROR_CODE	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	stErrorIdent_gb	ERROR_STRUCT	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	strDiagMessage_gb	STRING(60)	コントロール/軸の診断テキスト
O	arAppError_gb	ARRAY OF BOOL	アプリケーション固有のエラー (例: 安全扉が開いている)

I
O書き込み用入力
読み取り用出力

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

9.9.3 コントロール データ

I/O タイプ	名称	データの種類	説明
I	ImcCtrl (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_CONTROL_TYPE01	コントロール用のコマンド
O	ImcStatus (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_STATUS_TYPE01	コントロールからのフィードバック

I 書き込み用入力
O 読み取り用出力



ImcCtrl_gb の要素は "Machine" FB に書かれているため、外部的に使用することはできません。

9.10 操作手順：一般的なユーザー アクティビティ

9.10.1 運転モードの追加

GATcompact は以下の定義済み運転モードを備えています。

- オフ
- 手動
- 自動

例として、以下のセクションでは、"クリーニング" 運転モード (機械のクリーニング) も実装します。

これを行うためには、以下の手順が必要です。

- "TE_MACHINE_MODE" と入力する (*GATcompact* フォルダ) : **MODE_CLEANING** 要素で拡張する
- FB "Machine" :
 1. 移行 "traAuto" をコピーし、名前を "traCleaning" に変更して、"traCleaning" 内で MODE_AUTO を MODE_CLEANING で置き換えます。結果 : (Ctrl= MODE_CLEANING;)
 2. 移行 "traLeaveAuto" をコピーし、名前を "traLeaveCleaning" を変更して、"traLeaveCleaning" 内で MODE_AUTO を MODE_CLEANING で置き換えます。結果 : NOT(Ctrl= MODE_CLEANING;)
 3. "ModeAuto" と平行して "ModeCleaning" 分岐を追加します
 4. 上に挙げられている移行を "ModeCleaning" で使用します
 5. 入力アクション "actModeCleaning_entry()" を "ModeCleaning" で使用します。アクションで、最初の行として以下のコードを追加します : Status:= MODE_CLEANING;
 6. 移行の後で、"ModeAuto" と同じジャンプ ターゲットを追加します。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

拡張の結果

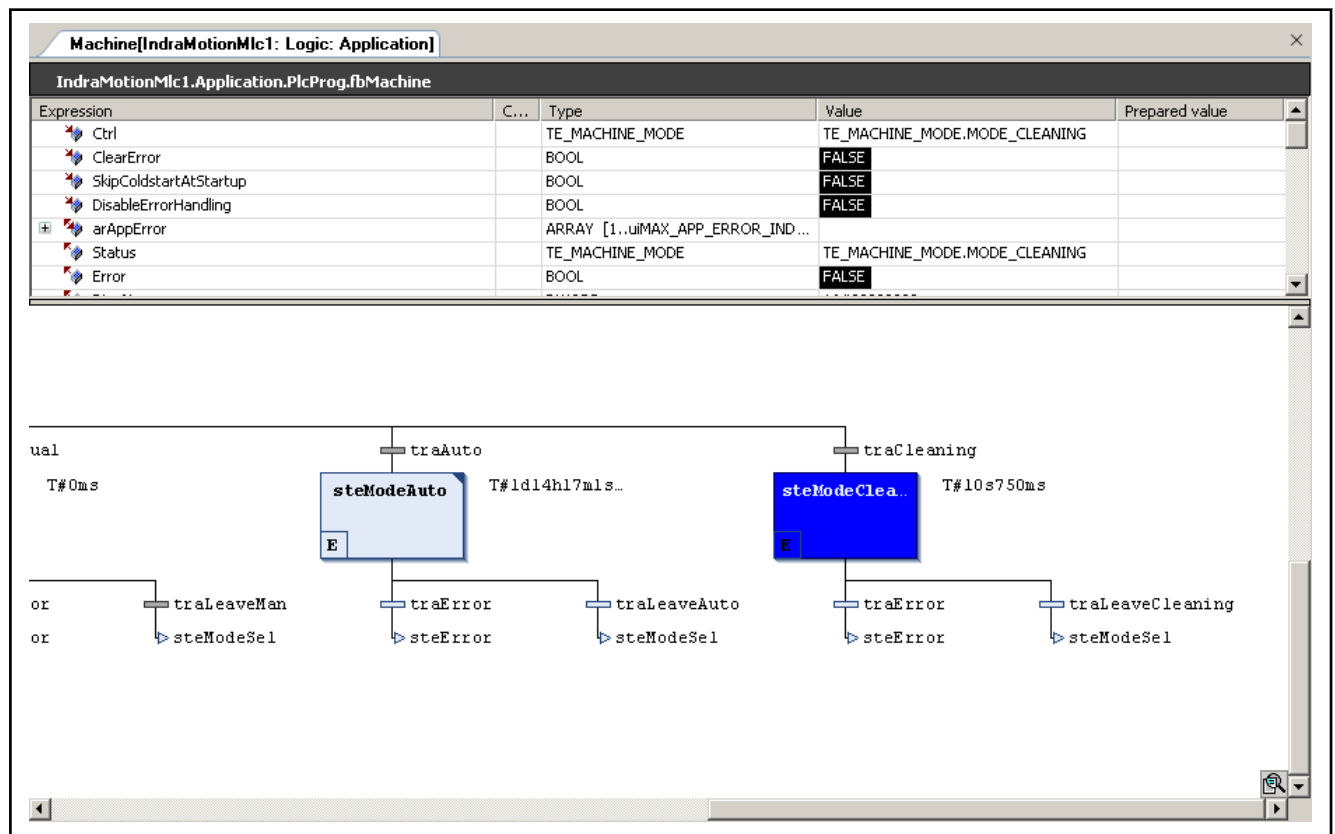


図 9-15: 運転モードを追加した後の結果

必要な動作を、アクション "actModeCleaning_entry()" またはオプションのアクション "actModeCleaning_active()" に実装できます。

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

9.11 GATcompact の概略

9.11.1 構造の概要

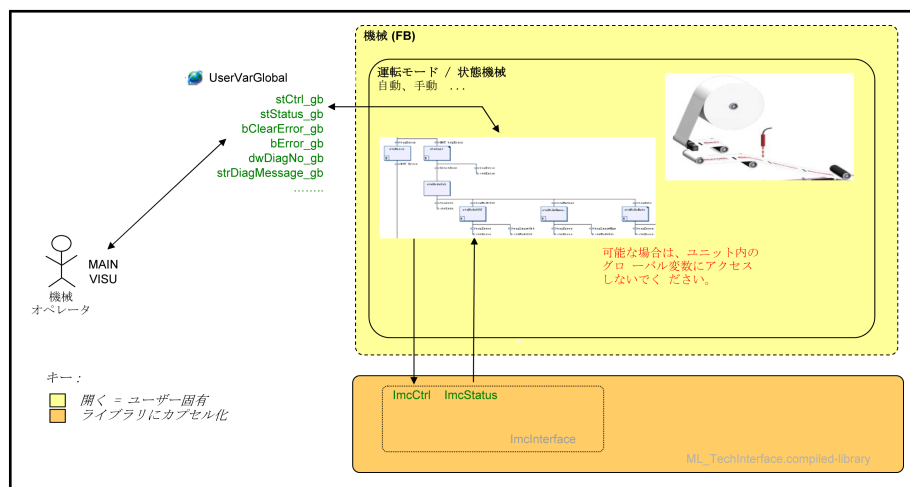


図 9-16: $GAT^{compact}$ の構造の概要

9.11.2 機械の最も重要なグローバル変数の概要

グローバル変数リスト "UserVarGlobal" を参照してください。

変数	説明	アクセス
stCtrl_gb	機械制御 (運転モードの選択、デフォルトの運転モード)	RD/WR
stStatus_gb	機械のステータス (現在の運転モード)	RD
bClearError_gb	エラーのクリア	RD/WR
bDisableErrorHandling_gb	エラーが発生した場合、エラー状態に切り替えない	RD/WR
bSkipColdstartAtStartup_gb	プログラムの起動時、"コールド スタート" (P2 への切替) が実行されない	RD/WR
bError_gb	TRUE はエラーを示します。	RD
dwDiagNo_gb	IndraLogic XLC-/IndraMotion MLC 資料に基づいた診断番号	RD
enErrorID_gb	dwDiagNo_gb = 16#F0FB 0000 (ファンクションブロック エラー) の場合の詳細情報	RD
stErrorIdent_gb	dwDiagNo_gb = 16#F0FB 0000 (ファンクションブロック エラー) の場合の詳細情報	RD
strDiagMessage_gb	コントロールまたは軸の診断テキスト	RD
arAppError_gb	ユーザー固有の機械エラー フラグ付きの BOOL の列	RD

RD	読み取り
WR	書き込み

9.11.3 機械のメソッドとアクションの概要

アクション/メソッド	呼出	関連タスク	説明/例
actCyclic_active	すべての状態で	PlcTask	安全エンジニアリングなどの総合的な動作
actError_entry	MODE_ERROR 状態 への移行中のサイクル 用	PlcTask	アプリケーション固有のエラーアクション (出力での停止)

sercos インターフェースなしの IndraLogic XLC 用 GATcompact

アクション/メソッド	呼出	関連タスク	説明/例
actInit_entry	MODE_INIT 状態への移行中のサイクル用	PlcTask	MODE_INIT 状態の準備
actInit_active	MODE_INIT 状態でのみ	PlcTask	パラメータ化モードへの切替、P2 のパラメータの説明である可能性あり。MODE_INIT を終了すると、運転モード (bb) に切り替わる、bb に達する際のパラメータの説明である可能性あり。
actModeOff_entry	MODE_OFF 状態への移行中のサイクル用	PlcTask	MODE_OFF 状態の準備
actModeManual_entry	MODE_MANUAL 状態への移行中のサイクル用	PlcTask	MODE_MANUAL 状態の準備
actModeAuto_entry	MODE_AUTO 状態への移行中のサイクル用	PlcTask	MODE_AUTO 状態の準備
actModeAuto_active	MODE_AUTO 状態でのみ	PlcTask	自動モードでのモニタリング、プロセスコントローラファンクションブロックの呼出など
FB_init	PLC プログラムの起動/リセット前に 1 回	特定のタスク割当なし	内部システムの使用は、変更できません

10 集中状態の機械での GAT

10.1 一般情報

Generic Application Template (GAT、汎用アプリケーション テンプレート) は、PLC アプリケーションを作成するために IndraWorks に組み込まれているソフトウェア ツールです。IEC61131-3 に準拠したプログラム フレームワークに基づいており、ダイアログベースのウィザードを備えているため、アプリケーションを迅速かつ容易に開発するために利用できます。また、軸の処理など、モーションアプリケーションの特定の機能もサポートしています。



集中運転モード SFC を備えた GAT は、IndraMotion MLC コントロールを必要とします。IndraLogic XLC コントロールは、現在のところ GAT によりサポートされていません。

10.2 概要

10.2.1 基本概念

GAT の使用は、アプリケーションはさまざまな運転モードで 1 つ以上の機能ユニットを協調させる機械で実行されるという考え方に基づいています。

以下の例を用いて、この考え方を説明します。

送り込みユニット、並べ替えユニットおよび包装ユニットからなる簡素な包装機械 (下の図を参照) のためにアプリケーションが作成されます。

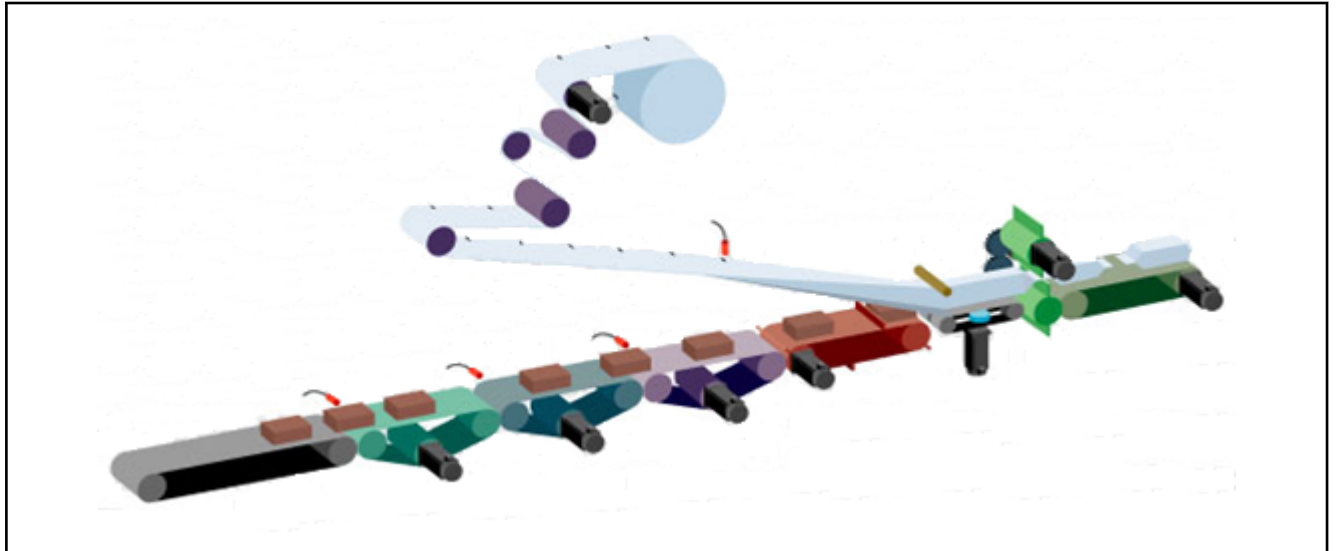


図 10-1: 例：水平製袋充填機

GAT を適用するために、この包装機械は 3 つの機能ユニット (モジュール) に分割されます。

集中状態の機械での GAT

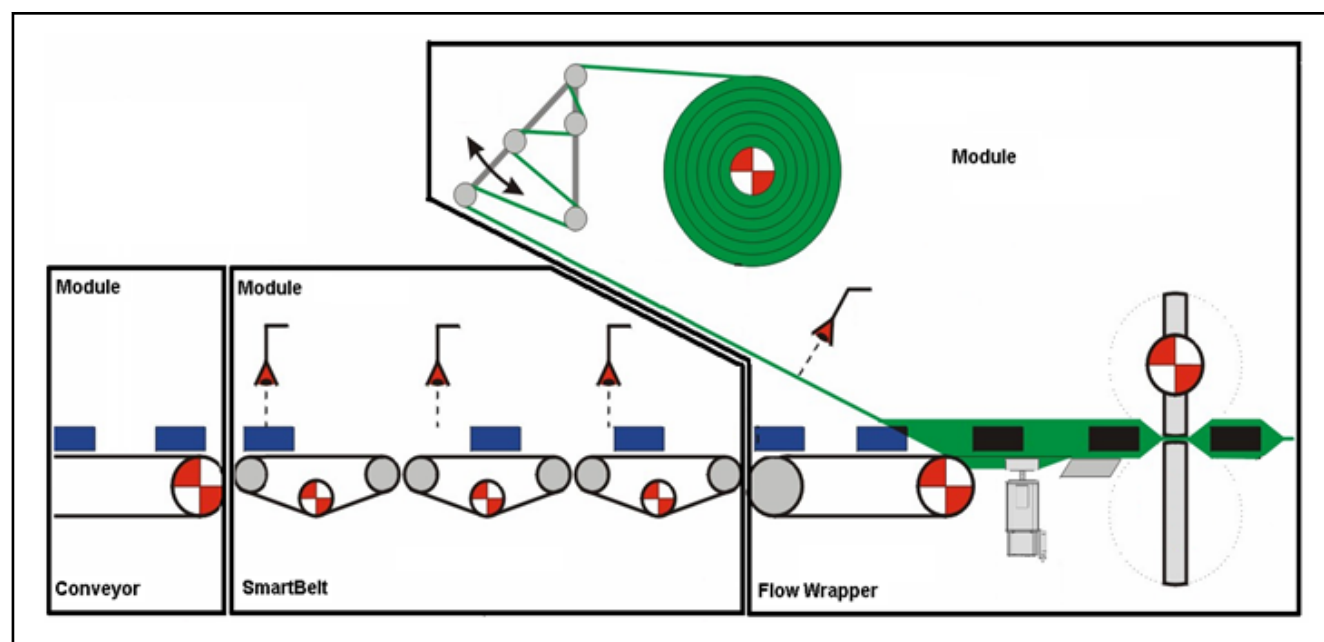


図 10-2: 水平製袋充填機の機能ユニット

その結果、以下のソフトウェア アーキテクチャが得られ、GAT を使用して実在の機械の対応ソフトウェアに直接実装できます。

集中状態の機械での GAT

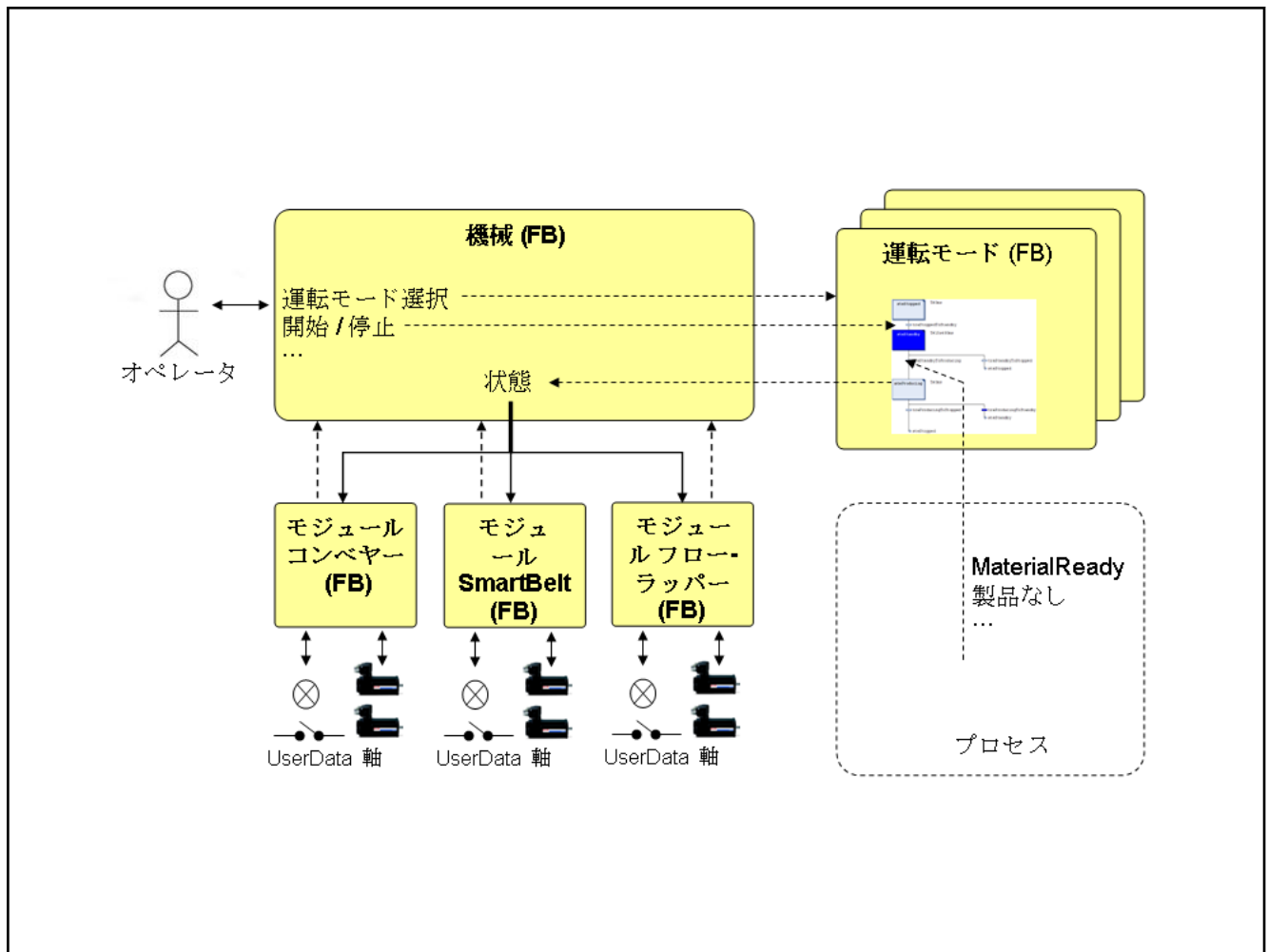


図 10-3: GAT を使用する場合のソフトウェア アーキテクチャ

10.2.2 使用の要件

前の章である程度提示しましたが、GAT を使用するための前提条件をここで再度まとめます。

GAT によりモーションアプリケーションを開発する前に、ユーザーはアプリケーションに関する以下の疑問について明らかにする必要があります。

- どの運転モードを実現するのか？ 各運転モードでどの状態が必要となるか？
- 機械はどの機能ユニット (モジュール) で構成されるか？ 各機能にどの軸および I/O が関連付けられるか？

この情報を用いて、ウィザードは実際のモーションアプリケーションのためのコード フレームワークを非常に短時間で生成できます。

以下のセクションでは、GAT の各パーツ、これらのパーツが相互にどのように動作するか、および実際のアプリケーションを実現するためにユーザーはどこで追加作業を行う必要があるかについて説明します。

集中状態の機械での GAT

10.3 最初の手順 – 集中状態の機械での GAT

10.3.1 単純な動作アプリケーションの作成

一般情報

以下の説明に従って、GAT ウィザードを使用して 2 つのモジュールと 2 本の軸を備えたモーションロジック アプリケーションを数分間で作成します。

これを行うために、IndraMotion MLC コントロールが必要ですが、実在のドライブは不要です。基盤は、集中運転モード SFC を備えた GAT です。

トピック

- 機械は 2 つの機能ユニット (モジュール) で構成されます。
- 1 つ目の機能ユニットで、軸は停止状態から始動して、生産状態では Automatic (自動) モードで毎分 100 回転で連続的に動作する必要があります。
- もう 1 つのユニットでは、2 番目の軸が Automatic (自動) 運転の生産中に 1 番目のモジュールの軸とフェーズ同期して動作します。下の図は、タスクを図式によって示したものです。

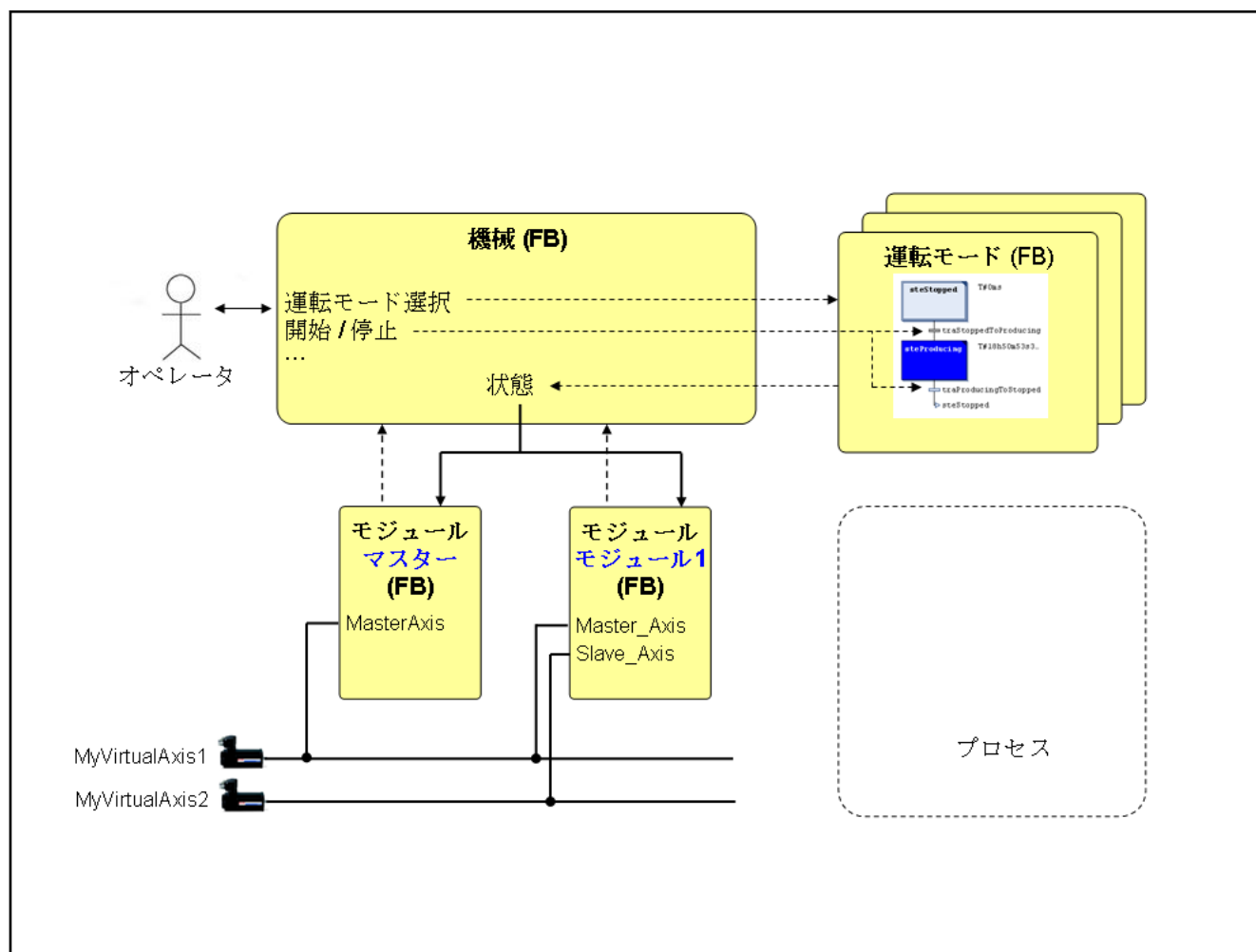


図 10-4: タスクの割当の図解

操作順序

- IndraWorks を開き、新規プロジェクトを作成します。

集中状態の機械での GAT

- MLC コントロールを、ライブラリからプロジェクト ツリーに追加します。
- GAT を MLC コントロールに追加します。これを行うには、コントロールの "GAT" コンテキスト メニューを使用するか、または IndraWorks ツールバーにある GAT ボタンをクリックします。
- 下のダイアログで "GAT (central state machine)" の選択肢を選び、"Finish" (完了) をクリックします。

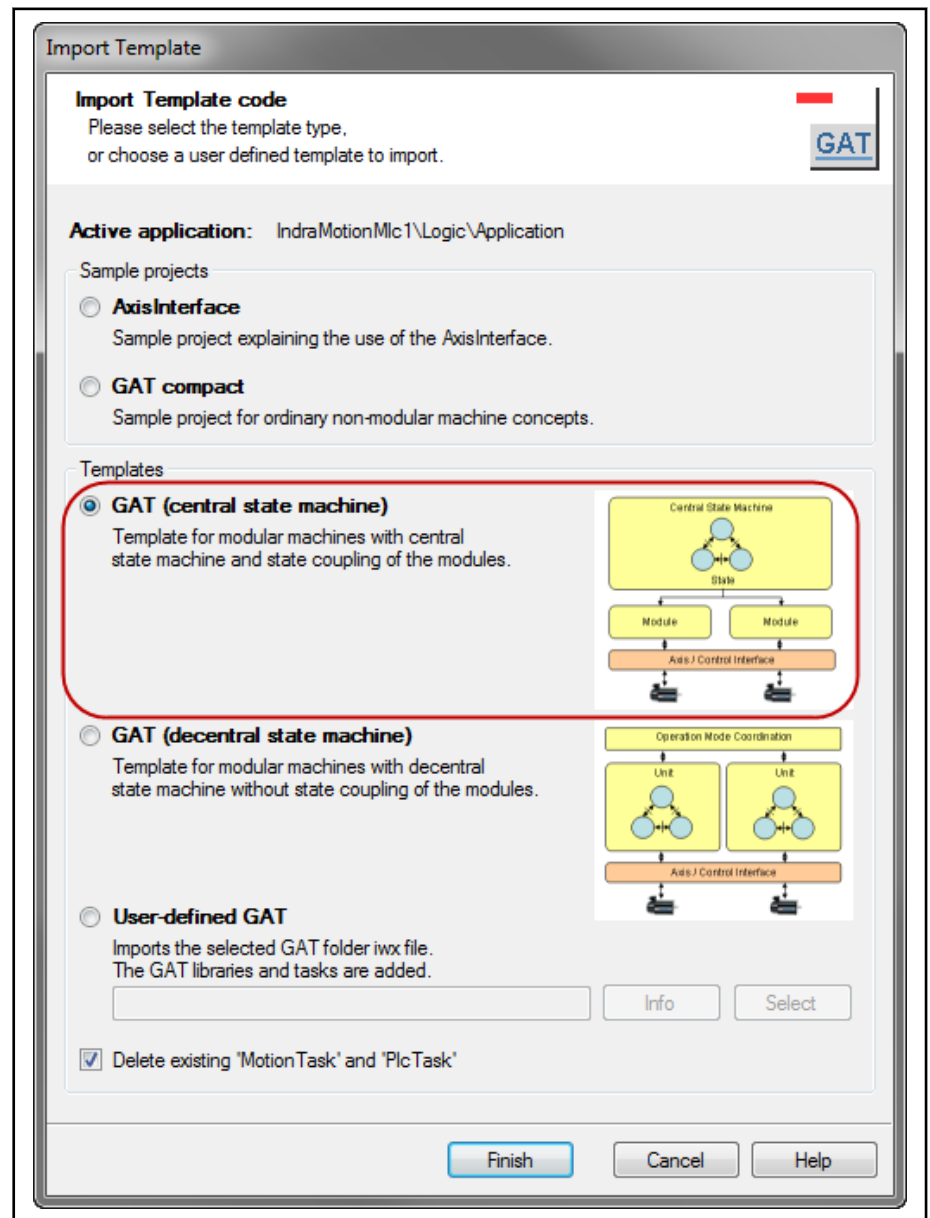


図 10-5: 集中状態の機械での GAT のインポート

"集中状態の機械での GAT" を選択することにより、実行可能で予め定義済みのプロジェクトがインポートされます。このプロジェクトにはすでに 2 本の仮想軸が含まれています。

AUTO (自動) 運転モード

GAT では、"Stopped" (停止) および "Producing" (生産中) の状態を備えた AUTO (自動) 運転モードが予め定義されています。グローバル変数 "bStart_gb" を設定すると、"Stopped" (停止) 状態から "Producing" (生産中) 状態への切替が行われます。

集中状態の機械での GAT

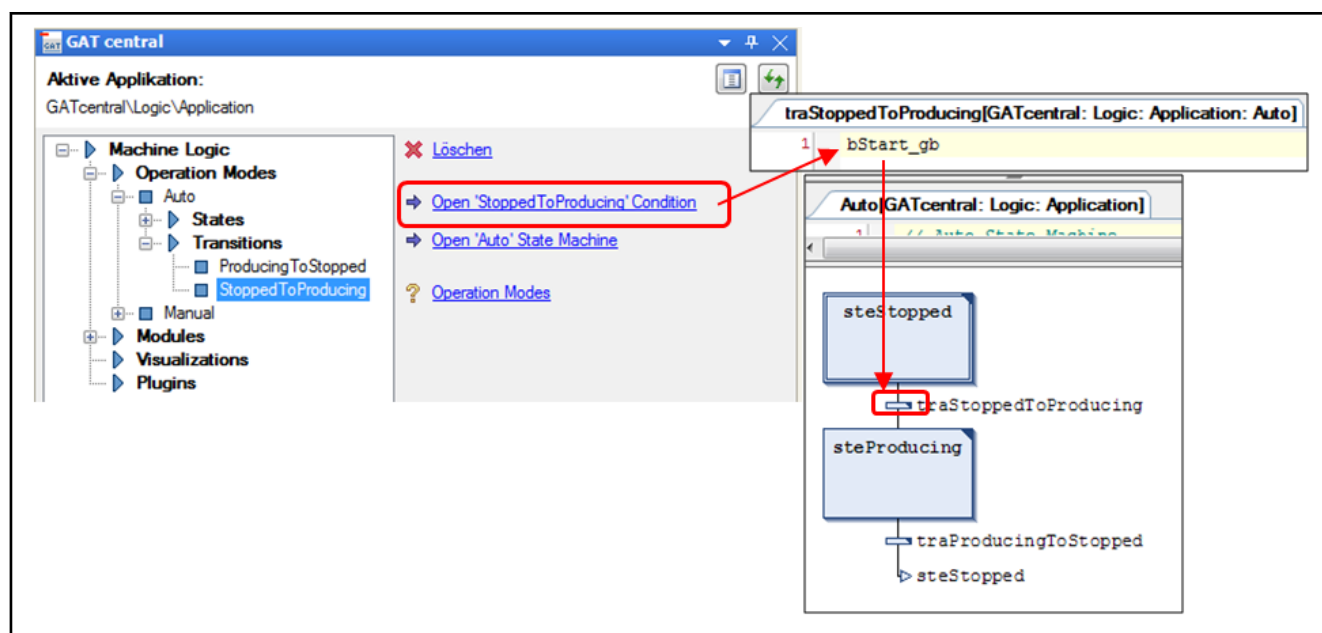


図 10-6: 移行の編集

運転モードまたは状態の定義に関するタスクを実現するために、他に必要となるものではありません。

Module_Master の作成

GAT 初期インポートにより、モジュール (Module1) はすでに利用可能です。タスクを完了するには、もう 1 つのモジュール "Module_Master" が必要です。このモジュールには、生産状態で動作し、非生産状態で停止する軸 "MasterAxis" が含まれています。

以下の手順でこのモジュールを作成します。

- GAT ディレクトリ ツリーで "Module" (モジュール) ノードをクリックし、"Add module" (モジュールを追加) を選択します。

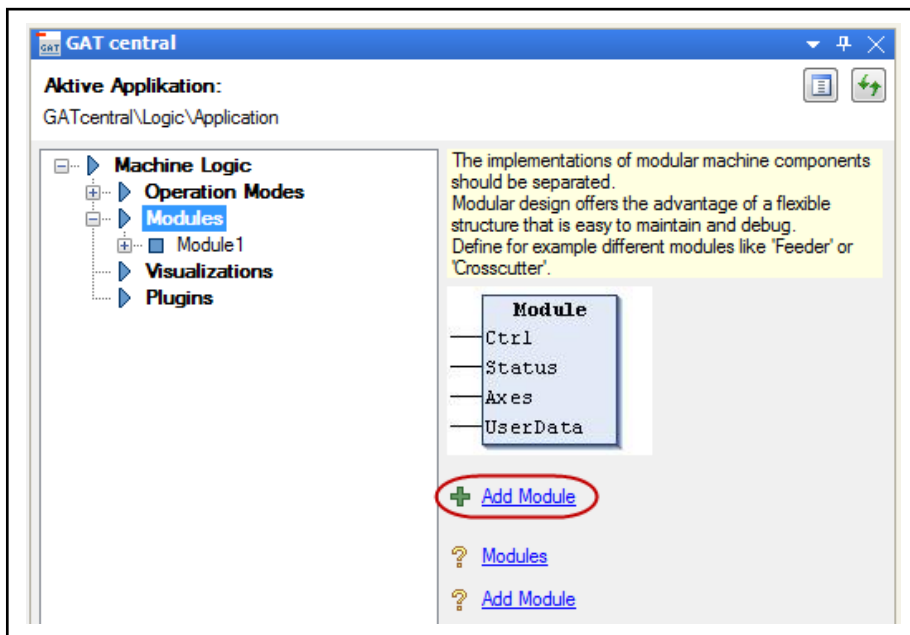


図 10-7: モジュールの追加

- ダイアログで名前 "Module_Master" を下記のとおり入力し、"Apply" (適用) をクリックします。

集中状態の機械での GAT

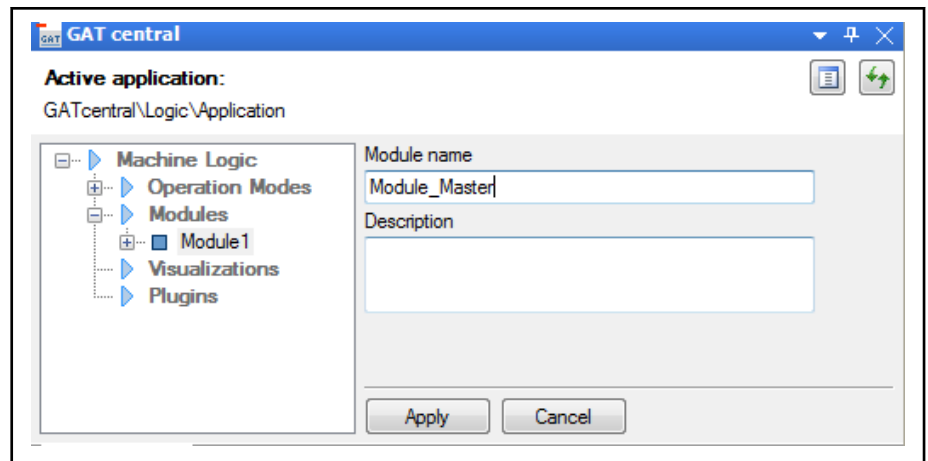


図 10-8: モジュール名の指定

- GAT ウィザードを介してモジュール "Module_Master" の軸 "MasterAxis" を以下のとおり追加し、"Apply" (適用) をクリックします。

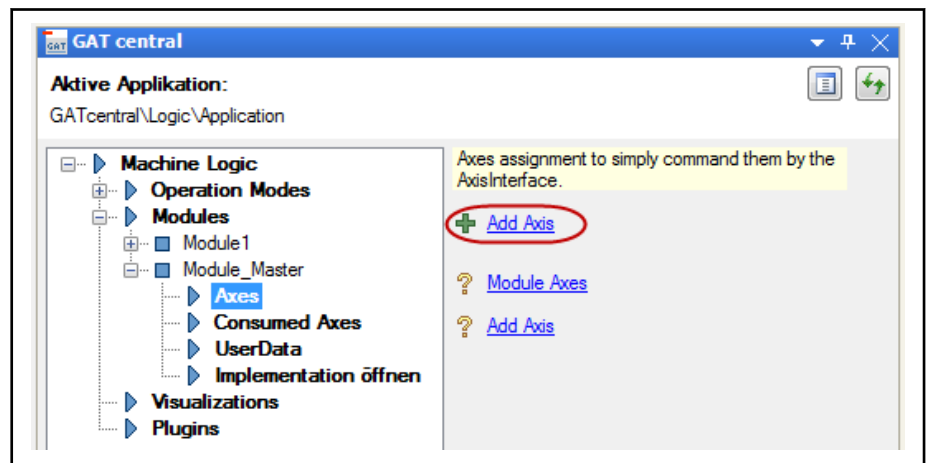


図 10-9: GAT ウィザードによる軸の追加

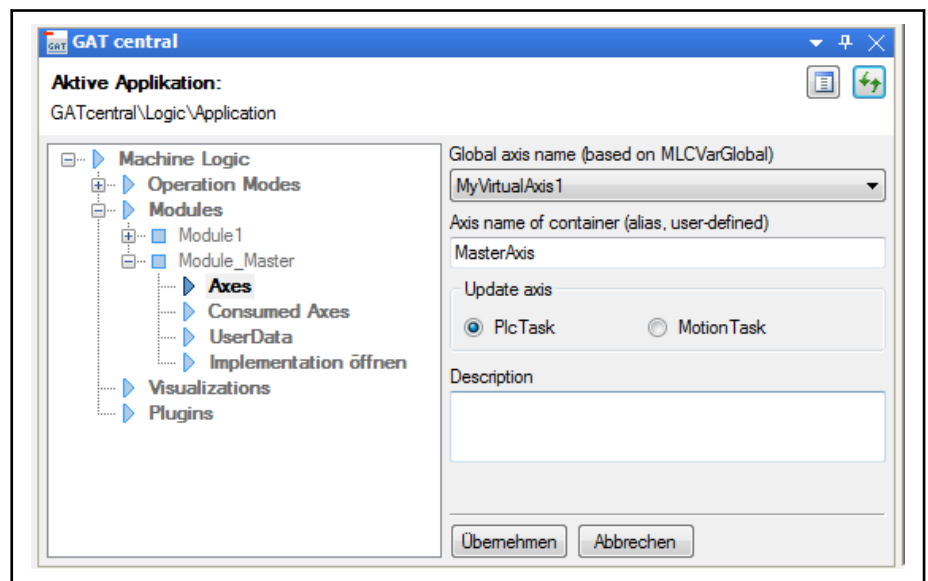


図 10-10: グローバル設定された軸へのモジュール軸の割当

集中状態の機械での GAT

- 状態 "Producing" (生産中) および "Stopped" (停止) で自動モードのプログラムコードを以下のとおり追加します。これを行うには、フォルダ "Open implementation" (実装を開く) を選択し、リンク "Auto_Producing" および "Auto_Stopped" を介してコード ポジションを開きます。そこで、下の図にあるプログラムコードを入力します。これにより、"Auto_Stopped" 状態で "MasterAxis" が停止するまで減速します (ModeAH により)。さらに、"Auto_Producing" 状態で "MasterAxis" は速度制御により 100 rpm で始動します (ModeVel により)。



快適な軸コマンドのために、ダイアログベースの軸コマンドも利用可能です。このトピックに関する詳細は、67 ページ "ダイアログベースの軸コマンド" 6.4.1 章を参照してください。

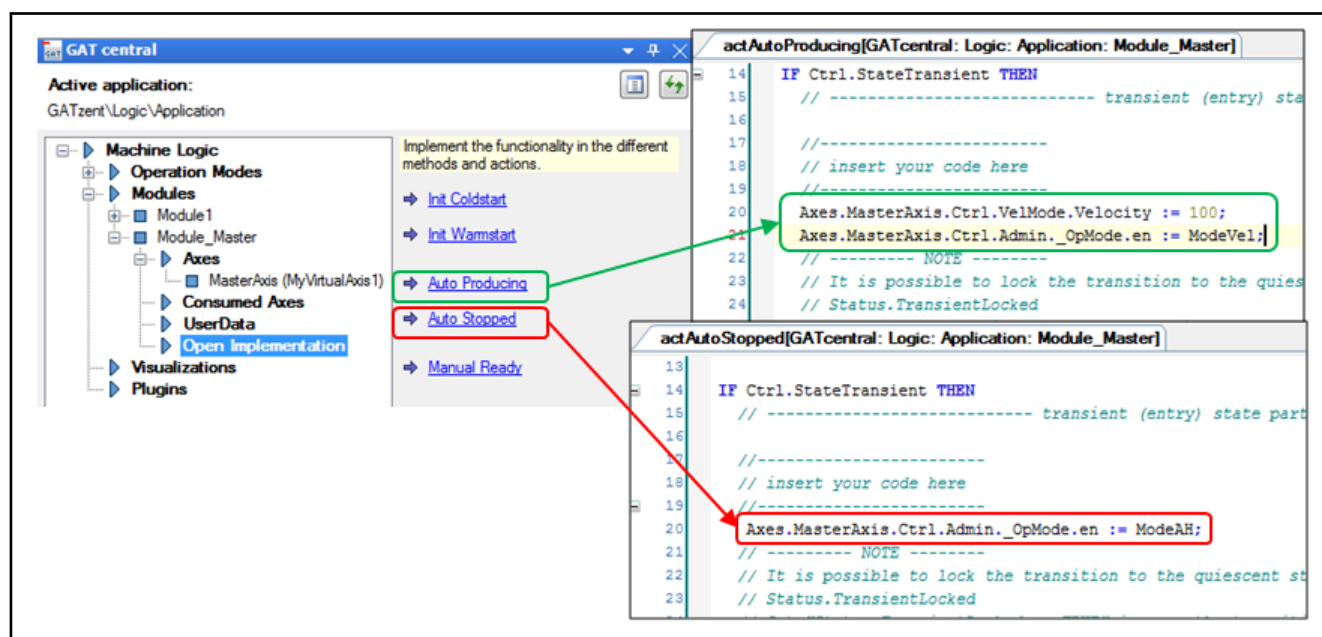


図 10-11: 状態のアクションでのコードの実装

軸コンテナ "Axes" に関する注意

モジュール内の軸の参照は、常に "Axes" を使用して行われます。すべてのモジュールで、軸コンテナの名前は "Axes" です。軸コンテナに含まれている軸をすべて表示するには、"." (ピリオド) を "Axes" の後に入力します (下の図を参照)。モジュールのすべての I/O データは、"UserData" を使用して同じ方法でアクセスできます。

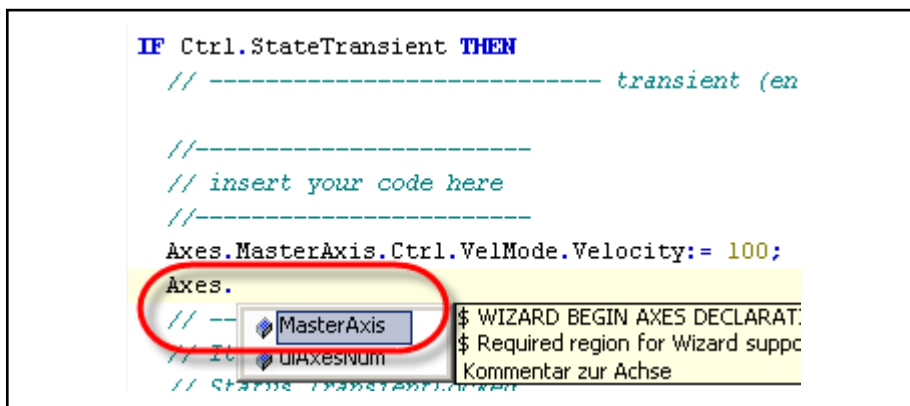


図 10-12: 軸のコマンドおよび Intellisense の使用

集中状態の機械での GAT

さらに、軸名の後にピリオドの演算子を入力することにより、その軸に関するデータを含む要素が表示されて選択できます (下の図を参照)。次のレベルに移動するには、前のレベルの名前の後にピリオドを入力します。

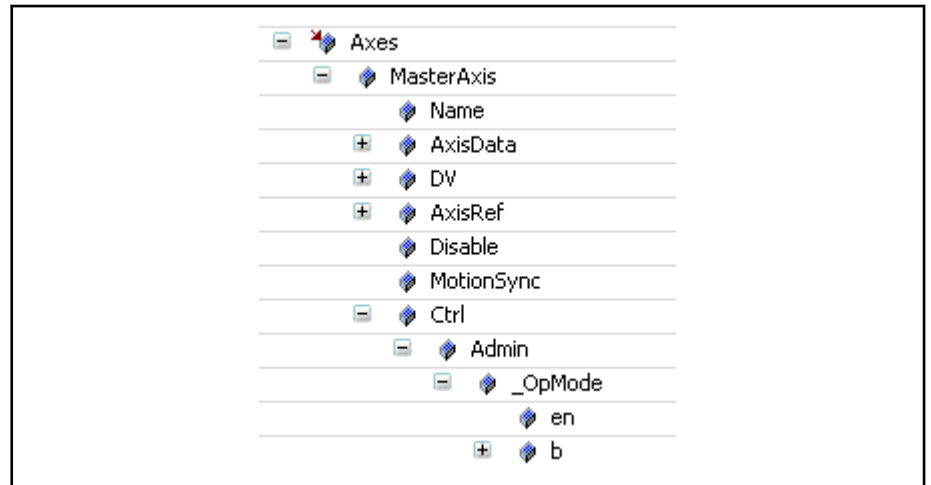


図 10-13: 利用可能な軸の構造要素

Module1 の調整

GAT 初期インポートにより、Module1 はすでに利用可能です。このモジュールの調整は以下の手順で行います。 "Automatic" (自動) 運転モードで、"Slave" 軸は "Master" 軸にフェーズ同期して動作します。 Automatic (自動) 運転モードでは、マスター軸はすでに "Module_Master" で始動または停止しています。

- GAT ウィザードを介して "Master_Axis" を消費軸として以下のとおり追加します。

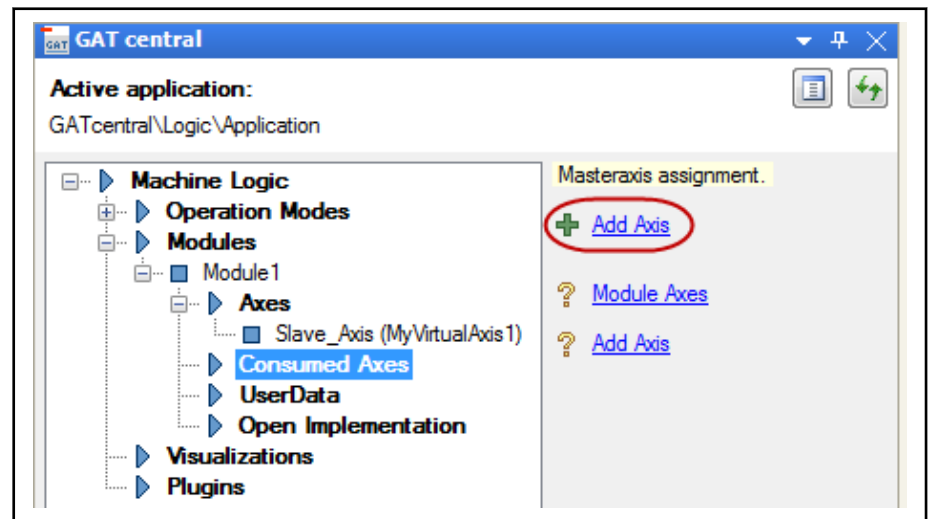


図 10-14: 消費軸の追加

集中状態の機械での GAT

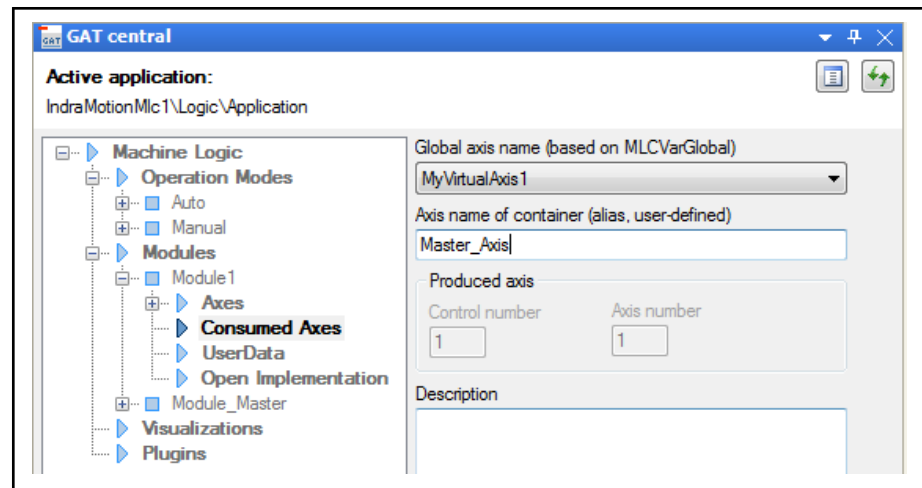


図 10-15: 消費マスター軸の追加



"Master_Axis" は、Module1 に"消費軸"として追加されます。これは、この軸がこのモジュールでマスター軸としてのみ使用されるためです。

- GAT ウィザードを介して軸 "MyAxis1" を削除します。

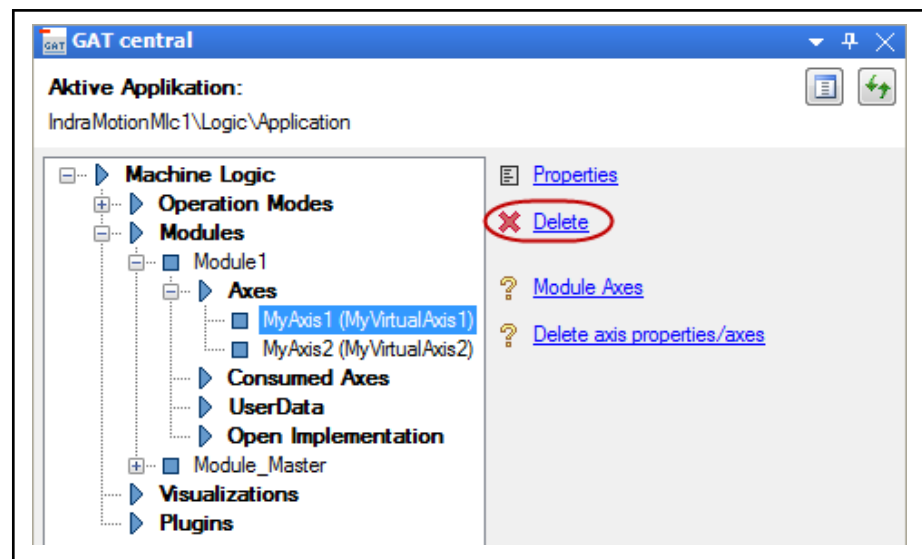


図 10-16: GAT ウィザードを介した軸 "MyAxis1" の削除

- 軸 "MyAxis2" の名前を "Slave_Axis" に変更します。

集中状態の機械での GAT

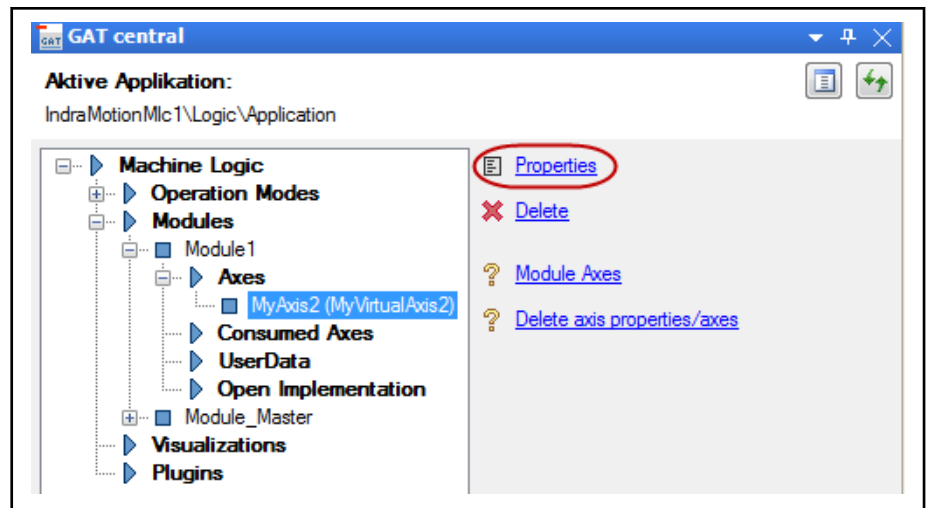


図 10-17: 設定済みの軸の編集

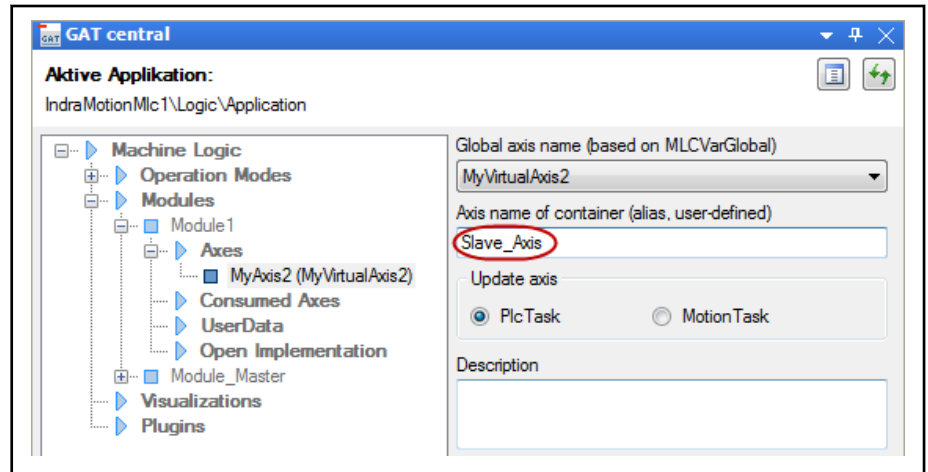
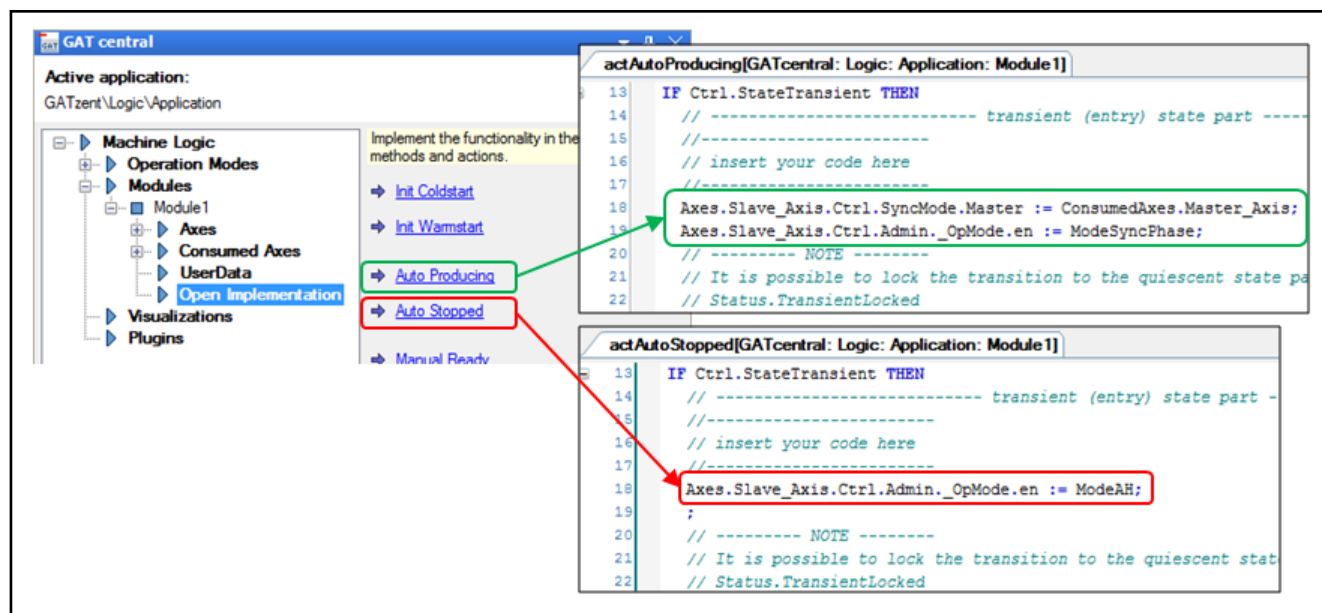


図 10-18: 軸の名前変更

- 状態 "Producing" (生産中) および "Stopped" (停止) で自動モードのプログラム コードを以下のとおり追加します。これを行うには、フォルダ "Open implementation" (実装を開く) を選択し、リンク "Auto_Producing" および "Auto_Stopped" を介してコード ポジションを開きます。そこで、下の図にあるプログラム コードを入力します。これにより、"Auto_Stopped" 状態で "Slave_Axis" が停止するまで減速します (ModeAH により)。さらに、"Auto_Producing" 状態で "Slave_Axis" はフェーズ同期運転モードにより "Master_Axis" と同期して始動します (ModeSyncPhase により)。

集中状態の機械での GAT



試運転表示の調整

図 10-19: "Auto_Producing" 状態のアクションでのコードの実装
新しい軸の名前は、試運転表示で以下のとおり調整する必要があります。

- GAT ウィザードを介して試運転表示を開きます。

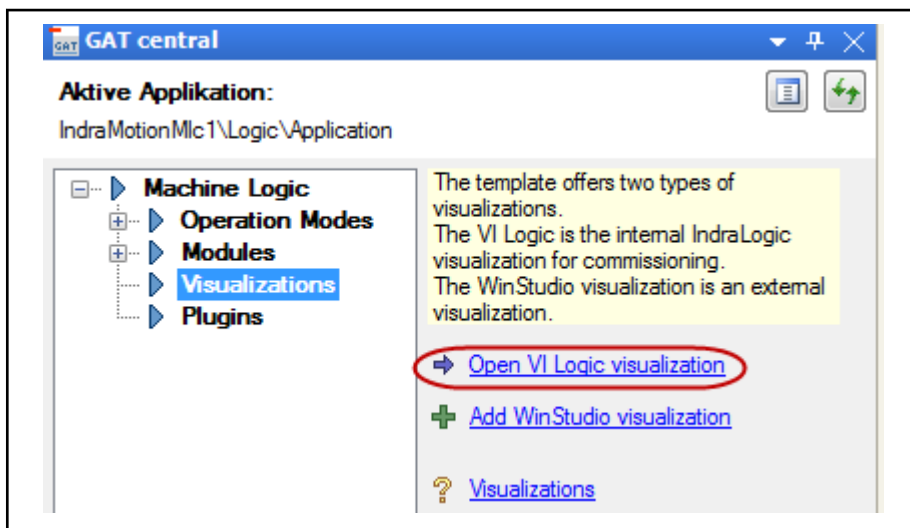


図 10-20: GAT ウィザードを介して試運転表示を開く

- 下の図で強調表示されている表示オブジェクトをクリックし、入力フィールド "Referenced visualizations" (参照される表示) でモジュールおよび軸の名前を以下のとおり調整します。すると、"Modul_Master" モジュールの "MasterAxis" に軸表示が適用されます。

集中状態の機械での GAT

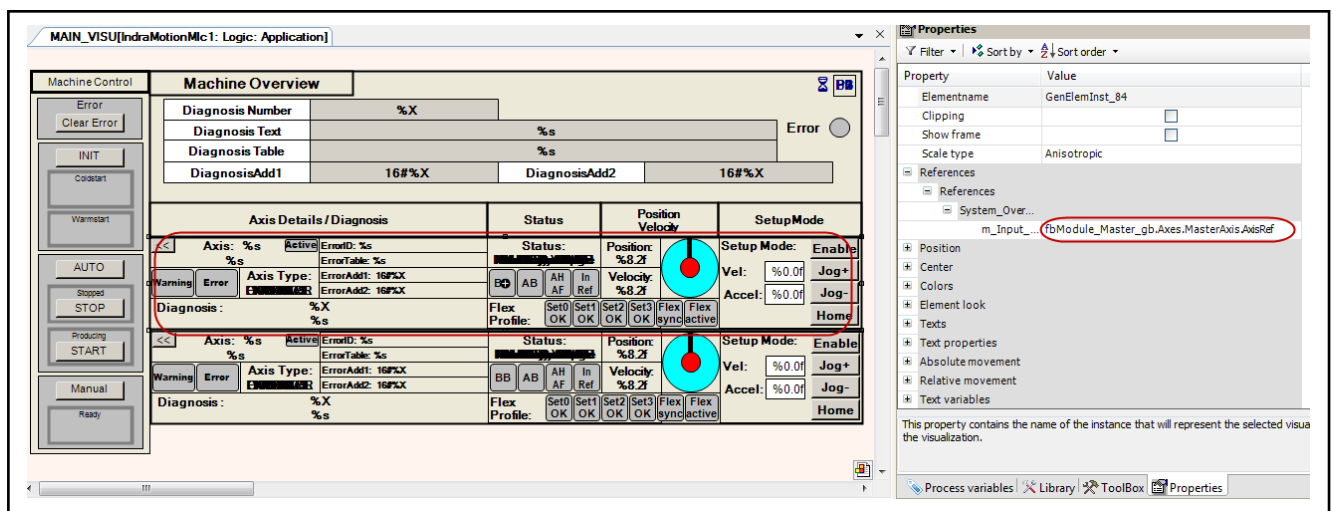


図 10-21: MasterAxis の軸表示の調整

入力フィールドに以下のテキストを入力します。

`fbModule_Master_gb.Axes.MasterAxis.AxisRef`

- 下の図で強調表示されている表示オブジェクトをクリックし、入力フィールド "Referenced visualizations" (参照される表示) でモジュールおよび軸の名前を以下のとおり調整します。すると、モジュール "Modul1" の "Slave_Axis" に軸表示が適用されます。

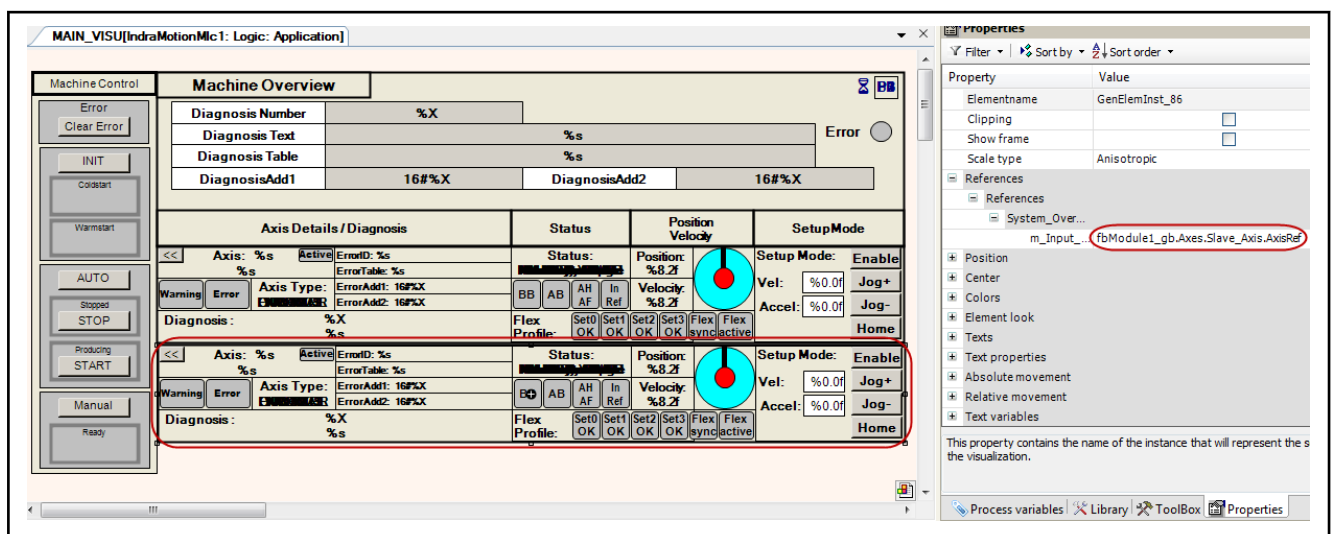


図 10-22: SlaveAxis の軸表示の調整

入力フィールドに以下のテキストを入力します。

`fbModule1_gb.Axes.Slave_Axis.AxisRef`

これによりプログラミングが完了します。

PLC プロジェクトの設定、ロードと起動

設定 (1) と PLC プロジェクト (2) をダウンロードした後で、PLC を起動します (3)。



図 10-23: PLC のダウンロード (設定/PLC オブジェクト) と起動

集中状態の機械での GAT

運転状態は、前の手順で作成した試運転表示 "MAIN_VISU" でモニタリングし、制御できます。

初期化が完了した後で、以下の手順を実行できます。

- デフォルト設定により、初期化後にコントロールは "AUTO" (自動) モードになります。 **START** (開始) ボタンをクリックします。これにより、状態が "Stopped" (停止) から "Producing" (生産中) に変わります。これにより、速度制御により Master 軸が始動し、Slave 軸が Master 軸とフェーズ同期して始動します。 **START** (開始) ボタンは、以前に書き込まれたグローバル変数 "bStart_gb" に書き込みます。
- **STOP** (停止) ボタンをクリックします。これにより、状態が "Producing" (停止) から "Stopped" (生産中) に変わります。これにより、マスター軸とスレーブ軸が停止します。 **Stop** はグローバル変数 "bStop_gb" に書き込みます。
- "Stopped" (停止) 状態では、運転モードを変更できます。これを行うには、 **Manual** (手動) ボタンをクリックします。この運転モードでは、SetupMode を介して軸のジョグまたは参照を行うことができます。表示の各 Enable (有効化) ボタンを使用して、SetupMode を有効にします。続いて、ジョグ ボタンを使用して軸をジョグし、ホームボタンでそれを参照します。

10.4 機能原則

モーションアプリケーションを簡単かつ効果的に生成するために、GAT のプログラム コードは 2 つのセクションに分かれています。1 つのセクションは、閉じたライブラリ (ML_TechTemplate.library) にありますが、もう 1 つのセクションへのアクセスは開かれています。

ウィザードを使用して、ユーザーは開いたセクションに特別なコードを配置します。ライブラリのコード セクションでは、運転モードと状態の呼出、モジュールおよび軸が整理されます。このライブラリは汎用です。すなわち、新しい運転モード、状態、モジュール、軸が追加されたり削除されたりした場合、またはその名前が変更された場合でも、要素の一貫性が維持されます。

この機能原則により、以下のメリットが得られます。

- コンパクトで分かりやすく整理されたユーザー コードの作成 :
-> 最適化されたプログラム テンプレートに基づいた簡素なインターフェースと明快な構造の使用を通じて
- より高い堅牢性 :
-> テストされ、最適化されたコード セクションの使用を通じて
- 保全性の向上
-> エラーが発生した場合のライブラリ交換が容易なため

10.5 初期化

初期化は機械の起動時に実行されます。初期化が正常に完了した後、機械は運転モードに入ります。

初期化は以下の 2 つの手順で実行されます。

1. コールド スタート : コントロールをパラメータ化モード (P2) に切り替えます。
2. ウォーム スタート : コントロールを運転モード (BB) に切り替えます。

集中状態の機械での GAT

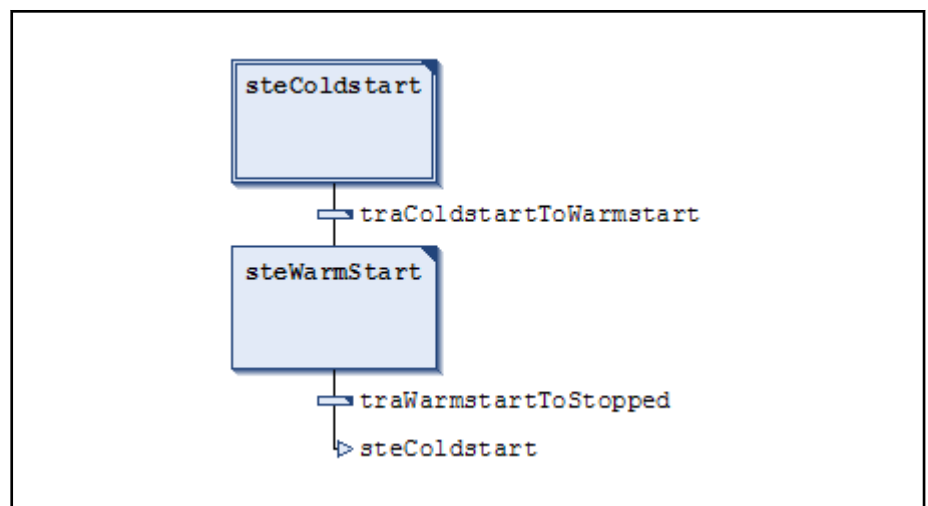


図 10-24: OpMode "_Init" 状態の機械

SkipColdstartAtStartup/
mOneTimeInit

PLC プロジェクトのダウンロードまたは PLC のリセットを行った後で、デフォルトにより "Init" 運転モードの "コールド スタート" 状態が有効になります。"コールド スタート = パラメータ化モード" への切替には時間がかかるため、特に当初の機械試運転の場合、この動作は障害を引き起こす可能性があります。このため、"SkipColdstartAtStartup = TRUE" 設定を使用して、この場合の "コールド スタート" への切替を防ぐことができます。この設定では、PLC のダウンロードまたはリセットの後、"コールド スタート" は省略され、"ウォーム スタート" のみが処理されます。ただし、"OpMode" 入力を使用して Init 機械運転モードが指定されている場合、"SkipColdStartAtStartup = TRUE" が設定されていても、"コールド スタート" が実行されます。

"SkipColdstartAtStartup" 設定の変更は、"Machine" FB にある "mOneTimeInit" メソッドの割当てで行うことができます (下の図を参照)。変更済みの PLC プロジェクトをダウンロードした後で、新しい設定が有効になります。

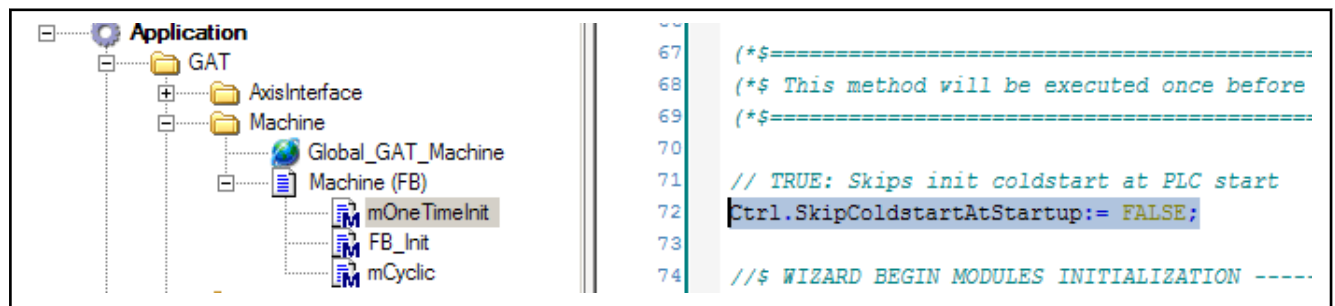


図 10-25: "SkipColdstartAtStartup" 構造要素を使用した初期化の設定



初期化のための状態機械 (SFC) ("GAT/OpModes/System/_Init" を参照) では、状態/移行を削除することはできません。

10.6 運転モード

10.6.1 一般情報

GAT には、予め設定済みの運転モード、"_Init"、"Auto" および "Manual" が含まれています。機械を初期化するための運転モード、"_Init" については、すでに前章で説明しました。

集中状態の機械での GAT

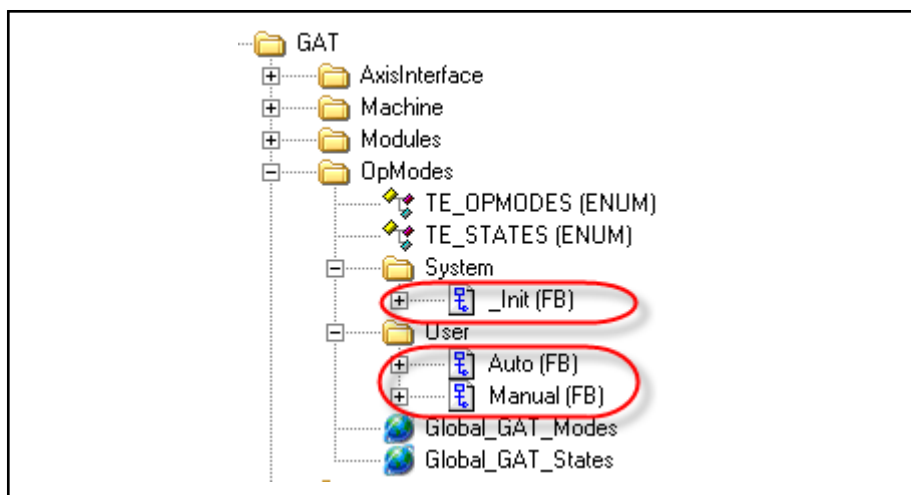


図 10-26: 設定済みの運転モード

運転モードは、一般に次のように区別されます。

- **システム運転モード**
システムにより提供され、ユーザーは変更できない "_Init" など
- **ユーザー運転モード**
ユーザーが変更可能な "Auto" および "Manual" など

通常、GAT アプリケーションの [282 ページ "モジュール" 10.7 章](#) はすべて同じ運転モードにあります。

実際のアプリケーションにより、ユーザーは最大 8 つのユーザー運転モードを定義できます ([311 ページ "運転モードの追加" 10.12.4 章](#) を参照)。

運転モードは、ML_TechTemplate.library にある機械構成ユニットにより処理されます。このため、ユーザーは GAT 内の運転モードを呼び出す必要はありません。

10.6.2 運転モードの状態

モーションアプリケーションを分かりやすく整理して構成するために、1 つの運転モードで複数の状態を定義できます ([311 ページ "エラー クリア後の必要な運転モードの選択" 10.12.5 章](#) を参照)。

これにより、例えば予め定義されたユーザー運転モード、"Auto" (自動) は、下の図に示されているように "Stopped" (停止) および "Producing" (生産中) の状態の組み合わせに基づいて識別されます。実際のアプリケーションで必要となる場合、ユーザーは新たな状態の追加、名前の変更や状態の消去を行うことができます。

集中状態の機械での GAT

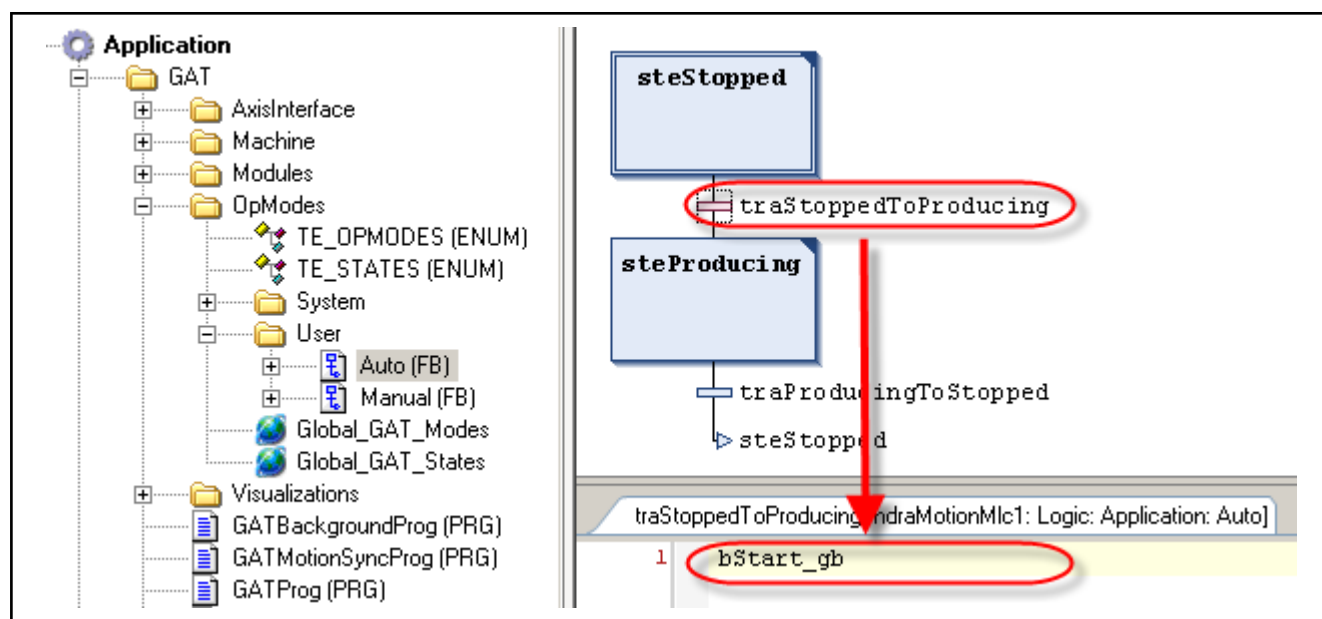


図 10-27: 運転モードの状態と移行

有効な状態の後の移行条件が TRUE である場合、運転モード内の状態変更が実行されます。ユーザーはすべての移行にアプリケーション固有の条件を設定できます。これにより、例えば、予め定義済みのユーザー運転モード "Auto" (自動) では、"Stopped" (停止) 状態が有効で、グローバル変数 "bStart_gb" の値が TRUE である場合、状態が "Stopped" (停止) から "Producing" (生産中) に切り替えられます (前の図を参照)。

10.6.3 初期化後のターゲット運転モードの設定

ターゲット運転モードは、初期化後に変数 **fbMachine_gb.Ctrl.Init_Successor** を使用して指定されます。デフォルト値は **OPMODE_AUTO** です。GAT は、別の運転モードのコマンドを出すために **fbMachine_gb.Ctrl.OpMode** が使用されるまで、この運転モードのままとなります。下の図は、これらの要素が GAT 内のどこにあるかを示したものです。新たな運転モードを作成するには、311 ページ "運転モードの追加" 10.12.4 章を参照してください。

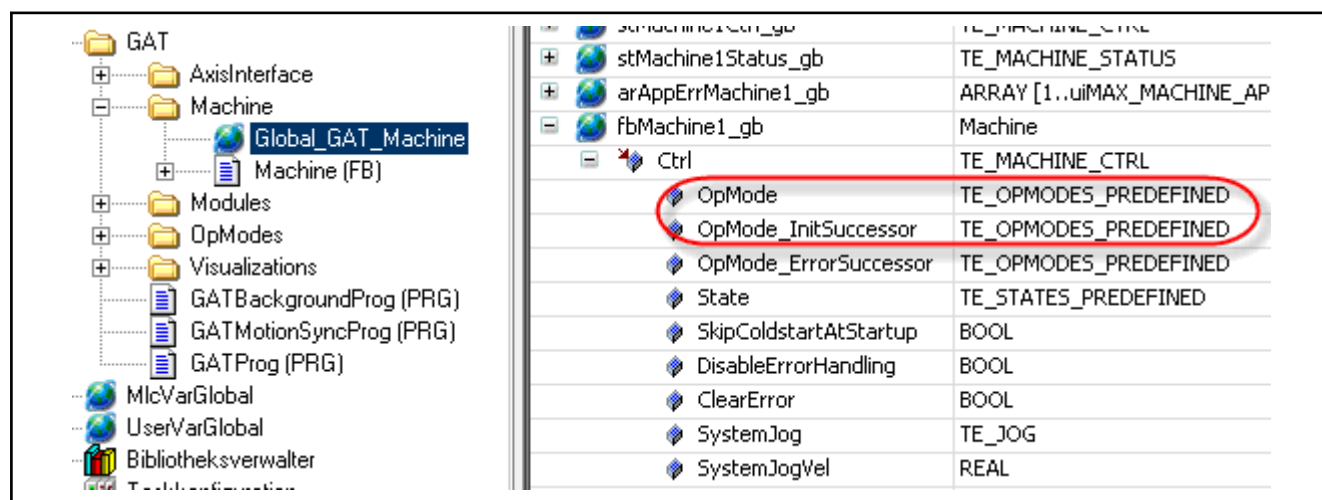


図 10-28: 初期化後のターゲット運転モードの設定

集中状態の機械での GAT

10.6.4 運転モードの変更

変数 `fbMachine_gb.Status.OpMode` とは異なる値（運転モード）が `fbMachine_gb.Ctrl.OpMode` に割り当てられると、運転モードの変更が行われます。しかし、これは現在の運転モードで初期状態 (STATE_<OPMODE>_STOPPED) が有効な場合のみ可能です。この場合、現在の運転モードの初期状態から新しい運転モードの初期状態への移行が行われます。

mOpModeChangeAllowed

オプションとして、"Machine" FB の "mOpModeChangeAllowed" メソッドにより運転モードの切替をブロックできます。このメソッドは、運転モードの切替が開始された場合に GAT によって自動的に呼び出されます。メソッドが FALSE の値を返す限り、運転モードの切替はブロックされます。メソッドが TRUE の値を返した場合、運転モードの切替はブロックされません（デフォルトの動作）。

プログラム

```
METHOD mOpModeChangeAllowed : BOOL
// 運転モード変更の場合はメソッドが呼び出されます
// mOpModeChangeAllowed := FALSE 運転モードの切替がブロックされます
// mOpModeChangeAllowed := TRUE 運転モードの切替が許可されます

// D 宣言
VAR_INPUT
    OpMode: TE_OPMODES_PREDEFINED; // 現在の運転モード
END_VAR

// 実行
(*
IF (OpMode=OPMODE_AUTO) THEN
    // 「自動」モードから他のモードへの切替は、「サービス キー」が有効な場合のみ許可されます
    mOpModeChangeAllowed := bServiceKeyEnabled;
END_IF
*)

// 実行終了
```



システムによっては、"Init" 運転モードへの切替（および、パラメータ化モード）は、すべての軸が Ab または AH 状態であることが確認された場合のみ許可されます。ユーザー運転モードから "Init" 運転モードに切り替える前に、Ab または AH 状態に変わるコマンドをすべての軸に発する必要があります。この理由のため、初期状態 "Stopped" の移行セクションでは対応する軸コマンドが推奨されます。

10.6.5 運転モード終了時のユーザー操作

mExitOpMode

特別なユースケースでは、運転モードを終了する際に特定のアクティビティを実行する必要があります（例：PLC 変数をリセットする）。このようなユースケースのために機能がすでに用意されており、以下のとおり使用できます。

メソッド "mExitOpMode" はすべてのモジュール FB/機械 FB に追加できます。このメソッドはサイクルのために運転モードを終了する際に GAT から呼び出されます。終了した運転モードは、メソッドの "OpMode" 入力を使用して GAT から転送されます。

この理由のため、以下のメソッドの宣言が必要です。

```
METHOD mExitOpMode
// Method is called for one cycle on operation mode exit

// Declaration
VAR_INPUT
    OpMode: TE_OPMODES_PREDEFINED; // Old operation mode
END_VAR

// Implementation
(*
CASE OpMode OF
```


集中状態の機械での GAT

```

OPMODE_AUTO:
; // Activities in order to reset all variables of operation mode AUTO
OPMODE_MANUAL:
; // Activities in order to reset all variables of operation mode MANUAL
ELSE
;
END_CASE
*)
// END Implementation

```

その後、メソッド内で運転モードはクエリされ、対応する PLC 変数がリセットされます。

10.6.6 運転モード開始時のユーザー操作

mEntryOpMode

特別なユース ケース によっては、運転モードを開始する際に特定のアクティビティを実行する必要があります (例: PLC 変数を設定する)。このようなユース ケースのために機能がすでに用意されており、以下のとおり使用できます。

メソッド "mEntryOpMode" は各モジュール FB/機械 FB に追加できます。このメソッドはサイクルのために運転モードを開始する際に GAT から呼び出されます。新たに開始した運転モードは、メソッドの "OpMode" 入力を使用して GAT から転送されます。

この理由のため、以下のメソッドの宣言が必要です。

```

METHOD mEntryOpMode
// Method is called for one cycle on operation mode entry

// Declaration
VAR_INPUT
OpMode: TE_OPMODES_PREDEFINED; // New operation mode
END_VAR

// Implementation
(
CASE OpMode OF
OPMODE_AUTO:
; // Activities in order to set all variables of operation mode AUTO
OPMODE_MANUAL:
; // Activities in order to set all variables of operation mode MANUAL
ELSE
;
END_CASE
*)
// END Implementation

```

その後、メソッド内で運転モードはクエリされ、対応する PLC 変数がリセットされます。

10.6.7 集中運転モードのバイパス

ある運転モードから別の運転モードに切り替える場合、一般にすべてのモジュールが新しい運転モードに切り替えられます。しかし、特定の状況においては、すべての機械モジュールが一元的に指定された運転モードに従うと、支障が生じることがあります。例えば、あるモジュールでメンテナンス作業を行う必要がある場合などです。

mForceOpMode(), mReleaseOpMode()

1 つ以上のモジュールを、FB "Machine" の fbMachine_gb.Ctrl.OpMode により指定された運転モードとは別の運転モードに切り替えることが可能です。この目的のために、モジュール メソッド "mForceOpMode()" および "mReleaseOpMode()" が利用できます。

両方の機能は、モジュールで直接呼び出されます。

```

fbModule1_gb.mForceOpMode(OPMODE_MANUAL); // OpMode "手動"に切り替わります
fbModule1_gb.mReleaseOpMode(); // グローバル OpMode に戻ります

```

個別のモジュールの手動切替は、ユーザー定義の運転モードでのみ可能です。運転モードを切り替える場合は、機械の集中運転モードを切り替える場

集中状態の機械での GAT

合と同じ規則が適用されます (280 ページ "運転モードの変更" 10.6.4 章を参照)。このため、指定された運転モードへの切替は、運転モードの切替を機械が許可する場合のみ実行されます。

機械がエラーを処理する必要がある場合、すべてのモジュールが影響を受けます。初期化が完了すると (OpMode _Init)、運転モードの手動バイパスはすべてのモジュールについて取り消されます。

 個別のモジュールが相互に依存している場合、集中運転モードをバイパスすることにより、機械全体の動作に意図しない変更をもたらす可能性があります。

名称	パラメータ	戻り値	説明
mForceOpMode	OpMode	BOOL	モジュールを、OpMode が指定する運転モードにします
mReleaseOpMode		BOOL	モジュールを、機械の運転モードにします

表 10-1: モジュールの運転モードを手動でバイパスするための機能

10.7モジュール

10.7.1一般情報

GAT にあるモジュールは、機械の機能ユニットを表しています。

ユーザーは、GAT 内のモジュールを呼び出す必要はありません。ML_TechTemplate ライブラリにある、ローカライズされた機械構成ユニットが呼出を自動的に行います。

313 ページ "新しいモジュールの追加" 10.12.8 章では、ウィザードを使用してモジュールの追加と削除を行う方法について説明しています。

提供されている GAT には、モジュール **Module1** がすでに含まれています (下の図を参照)。この名前は、GAT ウィザードを使用して簡単に変更できます (313 ページ "新しいモジュールの追加" 10.12.8 章を参照)。

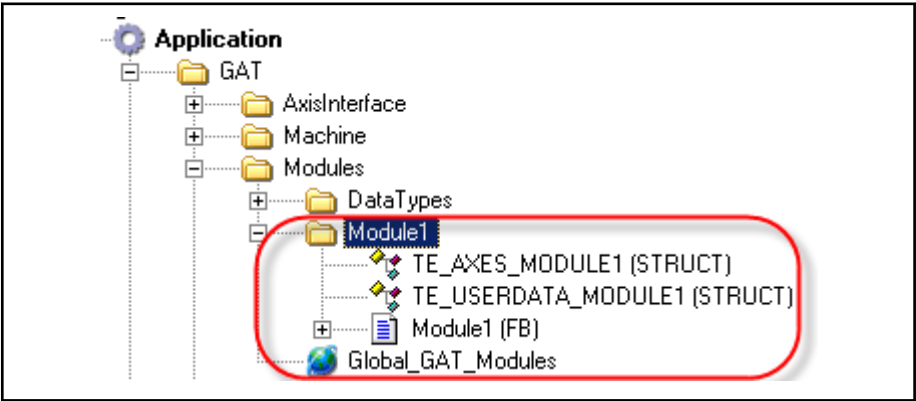


図 10-29: プロジェクト ディレクトリ内のモジュール ファイル

10.7.2作成規則

GAT モジュールの維持と再利用を容易にするために、モジュールは以下のよう

に設計する必要があります。

- モジュール間の依存関係を最低限度に抑える

- モジュール内でグローバル変数を使用しない

これを実現するために、各モジュールには以下のインターフェースがあります。

10.7.3 モジュール インターフェース

一般情報

下の図は、統一されたモジュール インターフェースを示したものです。

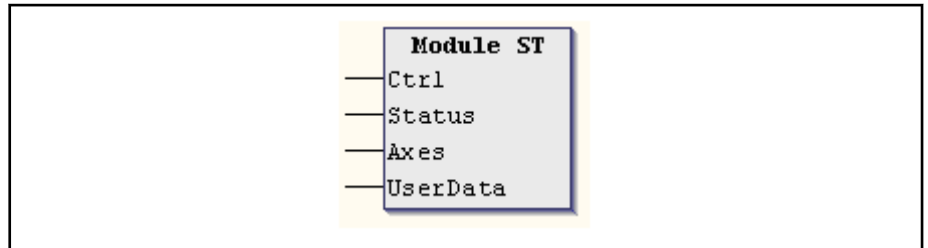


図 10-30: モジュール インターフェース

効果的で高性能なデータ処理のために、モジュール インターフェースのすべての要素は、参照として実行されます。すなわち、モジュールで必要とされるデータは、モジュールにコピーされるのではなく、参照によってモジュールに転送されます。

このインターフェースのコンポーネントについては、以下のセクションで説明します。

コントロールおよびステータスの構造

各モジュールには、制御のための変数 (Ctrl) と、ステータスを報告するための変数 (Status) があります (下の図を参照)。

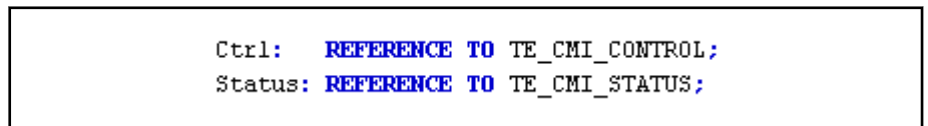


図 10-31: モジュールのコントロールおよびステータスの構造

これらの変数の名称と種類は、各モジュールで同じです。305 ページ "モジュール データ" 10.11.4 章に、どの要素がこれらの構造に属するか、ユーザーが使用できるのはどれか、および内部の機械構成ユニットでの使用に限定されているものはどれかについての概要が記載されています。

モジュール軸

各モジュールのインターフェースには、軸コンテナ (Axes) があります。このコンテナの軸はすべて、モジュール内で能動的に命令を受けます。



受動的に使用される軸 (マスター軸など) は、パフォーマンスを理由として軸コンテナ Axes で設定を行ってはいけません。

受動的な軸の設定は、Consumed Axes 軸コンテナで行います。

軸コンテナの名称は、各モジュールで同じですが、内容 (タイプ) はモジュールで使用される軸によって異なることがあります。

軸コンテナ内の各軸 (モジュール軸) はタイプ TE_GAT_AXIS で、これはタイプ TE_BASE_MODULE_AXIS に基づいています (ML_TechTemplate ライブラリを参照)。下の表は、構造タイプ TE_GAT_AXIS で利用可能なすべての要素を示したものです。

集中状態の機械での GAT

要素	タイプ	説明
名称	STRING(20)	文字列としての軸名 (システムが初期化された場合に初期化される。ユーザーによる変更が可能)
AxisData	REFERENCE TO ML_AXISDATA_SM	グローバル列 AxisData 内の対応する要素への参照 (ML_Base ライブラリを参照)
DV	REFERENCE TO ML_DirectVarAxis	ローカル軸がある場合： グローバル列 DV_Axis 内の対応する要素への参照 (ML_Base ライブラリを参照) リンク軸 (別のコントロールからの軸) がある場合： ダミー要素
GV	REFERENCE TO ML_GlobalVarAxis	グローバル列 GV_Axis 内の対応する要素への参照 (ML_Base ライブラリを参照)
AxisRef	AXIS_REF	軸の参照
Disable	BOOL	TRUE: 対応する Axis インターフェース データ セットは更新されない FALSE: デフォルト この変数の値に加えられた変更は、INIT モード (STATE_INIT_COLDSTART または STATE_INIT_WARMSTART) で初めて有効となる
MotionSync	BOOL	TRUE: この軸の軸データは、MotionTask で更新される FALSE: この軸の軸データは、PlcTask で更新される
Ctrl	REFERENCE TO TE_AXISINTERFACE_CONTROL	対応する軸の AxisInterface の制御構造への参照 (グローバル列 arAxisCtrl_gb の対応するエントリへの参照については、 AxisInterface フォルダにあるグローバル変数リスト Global_GAT_AxisInterfaces を参照)
Status	REFERENCE TO TE_AXISINTERFACE_STATUS	対応する軸の AxisInterface のステータス構造への参照 (グローバル列 arAxisStatus_gb の対応するエントリへの参照については、 AxisInterface フォルダにあるグローバル変数リスト Global_GAT_AxisInterfaces を参照)

表 10-2: タイプ **TE_GAT_AXIS** の利用可能なすべての要素。 モジュール内で、要素は **Axes.<AxisName>.<Element>** からアクセス可能

GAT プログラミング環境内の軸を、無効化するためのオプション (要素 **Disable**) または MotionTask に割り当てるためのオプション (要素 **MotionSync**) は、モジュール軸の定義または名前変更を行う際に GAT ウィザード内で選択できます (309 ページ "軸の削除/名前変更" 10.12.3 章 を参照)。



この軸を使用する MotionTask でアクティビティが実行中である場合、MotionTask に軸を割り当てる必要があります。

軸のすべての関連データは、**Axes** 軸コンテナを使用してアドレスを指定できます。

このコンテナは、軸の名前を用いて軸データにアクセスできるように設計されています。これにより、読み取りが容易なプログラム コードが生成されます。

下の表は、モジュール内およびモジュール外から軸コンテナにある軸にアクセスする方法を示したものです。

集中状態の機械での GAT

モジュール軸 ("Module1" モジュールの軸コンテナにある軸 "vAxis" など) へのアクセス	MotionTask への軸データの更新の割当	軸を無効にする
モジュール外からのアクセス	stAxesModule1_gb.vAxis.MotionSync := TRUE;	stAxesModule1_gb.vAxis.Disable := TRUE;
モジュール内からのアクセス	Axes.vAxis.MotionSync := TRUE;	Axes.vAxis.Disable := TRUE;

上位の機械のメソッド "mOneTimeInit" (このメソッドは GAT の初期化中に 1 回のみ処理される) で、ウィザードを使用して物理軸 (実在、仮想、...) を、モジュール内の対応する軸に割り当てます (下の図および 308 ページ "軸の追加" 10.12.2 章を参照)。

図 10-32: "mOneTimeInit" メソッドでの軸の割当

上の図からの例:

IndraWorks で生成されたグローバル軸 "MyVirtualAxis1" は、モジュール "Module1" の軸 "Master_Axis" を割り当てられます。

受動的な軸 (消費軸)

受動的に使用されるモジュールの軸 (マスター軸など) をすべて **ConsumedAxes** 軸コンテナで設定します。

GAT ウィザードを使用して以下の手順を実行します。

- GAT ウィザードを開く (IndraWorks ツールバーにある GAT ボタン



- モジュールの下に **Consumed Axes** (消費される軸) ノードを選択する
- "Add Axis" を選択してダイアログ クエリに従う

エンコーダ、リンクおよび C2C 消費軸も、受動的に使用される軸として使用できます。

集中状態の機械での GAT

Consumed Axes (消費される軸) 軸コンテナの使用は、以下のメリットに関連があります。

- パフォーマンス上のメリット AxisInterface は受動的な軸のために呼び出されないため
- よりユーザーフレンドリー どのモジュールでどの軸が能動的に命令を受けているかを素早く確認できるため

モジュールのプログラム コードでの消費軸の使用例

```
// 例: MySlaveAxis1 は、VirtualMaster 1 が使用する軸と同期されます
Axes.MySlaveAxis1.Ctrl.SyncMode.Master := ConsumedAxes.VirtualMaster1;
Axes.MySlaveAxis1.Ctrl.Admin._OpMode.en := ModeSyncPhase;
```

C2C 消費軸の場合、GV_LinkAxis[] 変数を用いてモジュール内で軸データ (位置、速度など) を読み取ることができます。

他のすべての軸のタイプの場合は、DV_Axis[] 変数を用いて消費軸の軸データを読み取ることができます。

消費軸は、要素 "CntrlNo" および "Axis No" を持つデータ タイプ AXIS_REF に対応します。

モジュール固有の I/O データ

モジュールを再利用する場合、そのモジュールでグローバル データにアクセスすることはできません。モジュールに必要なすべての I/O は、I/O コンテナ **UserData** を使用して利用可能になります。I/O コンテナの名前は各モジュールで同じですが、内容 (タイプ) は異なります。これは、すべてのモジュールが同じ I/O データを使用するわけではないためです。

変数 "UserData" は、ユーザーが要素を定義する必要のあるデータ構造です。

下の図は、GAT に含まれている "Module1" モジュールの I/O データのうち、定義する必要のある部分を示したものです。

```

1  TYPE TE_USERDATA_MODULE1 EXTENDS TE_DUMMY_MODULE:
2  STRUCT
3      (*$ Module specific interface =====
4      (*$ Name of the Module $*)
5      Name: STRING(20) := 'Module1';
6
7      (*$-----$*)
8      (*$ insert your module specific input variables here $*)
9      (*$-----$*)
10
11     (*$ END - insert your module specific input variables here $*)
12
13     (*$-----$*)
14     (*$ insert your module specific output variables here $*)
15     (*$-----$*)
16
17     (*$ END - insert your module specific output variables here $*)
18
19     (*$-----$*)
20     (*$ insert your module specific in-out variables here $*)
21     (*$-----$*)
22
23     (*$ END - insert your module specific in-out variables here $*)
24     (*$ END - Module specific interface =====
25
26  END_STRUCT
27  END_TYPE
  
```

図 10-33: モジュール固有の I/O データ

モジュール インスタンス

既存のモジュールから新しいインスタンスを作成できます。これらのインスタンスは、割り当てられる IndraWorks 軸および割り当てられる I/O のみが異

集中状態の機械での GAT

なります。モジュールのインスタンス化は GAT ウィザードによりサポートされており、セクション 314 ページ "モジュールのインスタンスの追加" 10.12.9 章 で説明されています。

10.7.4 モジュールの状態

運転モード内の状態を追加または削除する場合、ウィザードにより、その状態に対応するアクションが各モジュールで必ず作成されます。このアクションは、対応する状態が関連する運転モードで有効である場合のみ処理されます。このようにして、"AUTO" (自動) 運転モードの "Producing" (生産中) 状態が有効な場合、モジュール "Module1" でアクション "actAutoProducing" が処理されます。

状態を参照するモジュール内の各アクションは、移行セクションと静止セクションに分けられます。

状態を参照するアクションの移行セクションは、**Ctrl.StateTransient** 条件によって指定されます。ここで、この状態に入るために一定の期間のみ必要となるアクティビティ (AUTO_PRODUCING 状態での温度コントローラのスイッチオンなど) の連携を行います。PLC サイクルでこれが不可能な場合は、**Status.TransientLocked:= TRUE** の設定を行い、プロセスがこのセクションに残るようにします (例: AUTO_PRODUCING 状態で温度コントローラの命令温度に達するまで)。

Ctrl.StateTransient=FALSE の場合は、状態を参照するアクションの静止セクションが処理されます。この場合、この状態にすでに入った場合に実行されるすべてのアクティビティ (同期結合/製品が欠けている場合の結合解除など) をコード化します。**Status.StateLocked:=TRUE** を設定することにより、対応する移行条件が満たされた場合でも、この運転モードの別の状態への切替を防ぐことができます。

10.7.5 モジュールの連携

モジュールは、他のモジュールに関する情報は一切含まないため、モジュールの連携は上位の位置で行う必要があります。

ML_TechTemplate.library ライブラリの機械構成ユニットは、各 PLC サイクルで機械の "mCyclic" メソッドを呼び出します (下の図を参照)。周期的に呼び出されるこのメソッドで、I/O 情報がモジュール間で、およびモジュールから機械の I/O ポイントへと交換されます。

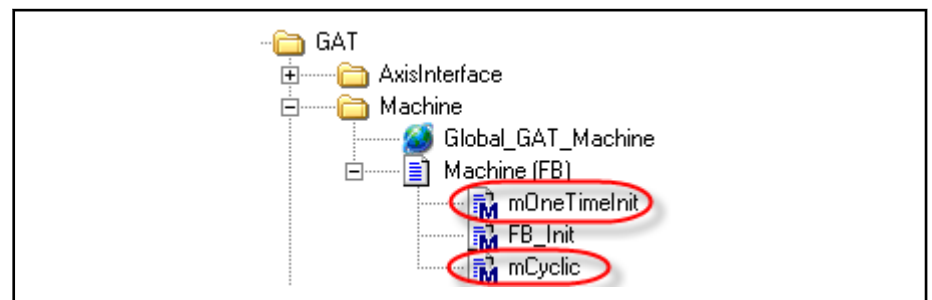


図 10-34: 機械構成ユニットのメソッド "mCyclic"

モジュール設定または対応するモジュール軸へのグローバル軸の割当はランタイム中に変化しないため、これらのアクティビティは機械の "mOneTimeInit" メソッドで実行されます。このメソッドは、機械構成ユニットを使用して GAT を初期化するために 1 回のみ呼び出されます (前の図を参照)。

"mOneTimeInit" メソッドに含まれているコードは、ウィザードにより管理されます。モジュールまたはモジュール軸を追加もしくは削除する場合、対応するエントリが追加または削除されます。

集中状態の機械での GAT

10.8 タスク処理

10.8.1 タスク設定、GAT

PLC ないでは複数のタスクが同時に実行されます。これらのタスクはそれぞれ独自の PLC コードを実行します。コントロール プロセッサによるタスクの処理は、タスクの優先度とサイクル時間に基づいて指定されます。

MotionTask、PlcTask および BackgroundTask はすべて標準の GAT に含まれています。

	優先度	サイクル時間	ウォッチドッグ 時間	ウォッチドッグ 感度	プログラム呼出
GATMotionSyncTask	2	=動作のサイクル時間	T#2ms	1	GATMotionSyncProg
GATProgTask	5	T#10ms	T#20ms	2	GATProg
GATBackgroundProgTask	6	T#40ms	-	-	GATBackgroundProg

10.8.2 GATMotionSyncTask

MotionTask から軸に継続的にアクセスするために、MotionTask で AxisInterface も呼び出す必要があります。すべての軸を MotionTask で処理すると、MotionTask に不要な負荷がかかります。このため、GAT ウィザードを使用して軸を追加し、AxisInterface の処理を PlcTask または MotionTask に割り当てることができます。

36 ページ "MotionTask の一般的なタスク" 章 も参照してください。

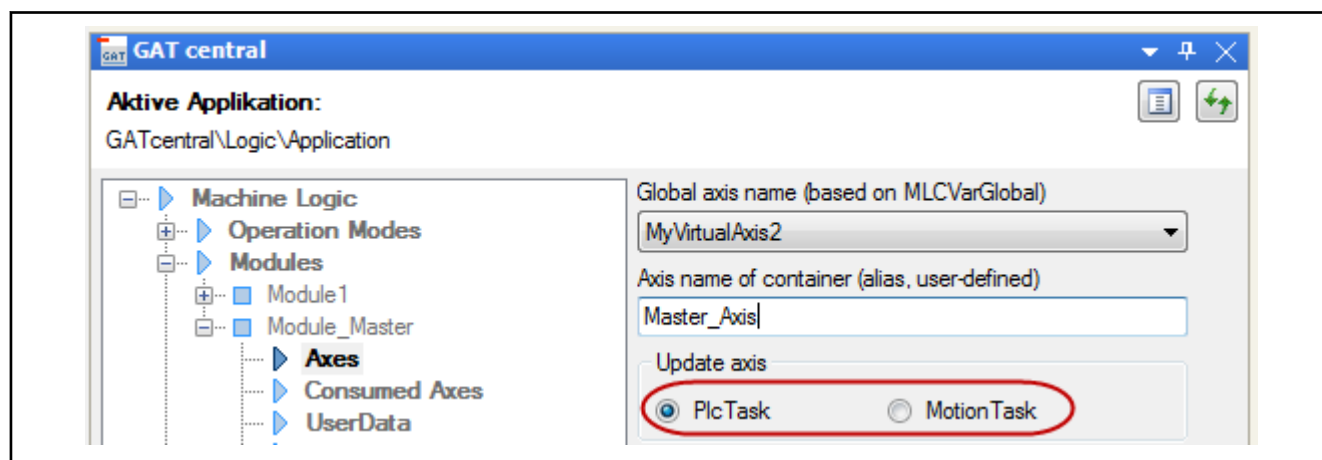


図 10-35: AxisInterface 処理の割り当て



原則として、軸データは PlcTask で更新する必要があります。

この軸を使用する MotionTask でアクティビティが実行中である場合、MotionTask に軸を割り当てる必要があります。(例：MotionTask でプロセス コントローラが処理されます。その場合、プロセス コントローラに割り当てられているすべての軸も、MotionTask で更新する必要があります。)

MotionTask での処理のために、各モジュール/機械は "mMotionCyclic" メソッドを備えています。このメソッドより、すべてのモジュール/機械変数 (一元的な軸入力ポイント Axes など) にアクセス可能です。

10.8.3 GATBackgroundProgTask

PlcTask のランタイムが長すぎる場合は、タイム クリティカルではないタスクをバックグラウンド タスクに移動して、PlcTask の負荷を減らすことができます。ただし、追加的なタスク Switch PlcTask <> BackgroundTask によって、より大きな負荷がシステム全体にかかることに留意してください (36 ページ "BackgroundTask の一般的なタスク" 章 も参照)。

BackgroundTask での処理のために、各モジュール/機械は "mBackground" メソッドを備えています。すべてのモジュール メソッドはこのメソッドからアクセス可能です。

10.9 エラー処理

10.9.1 エラーの検出および診断

プログラム テンプレート GAT には、**中心的な位置**でエラーを検出、処理、クリアするために予め定義済みの機能が含まれています。さまざまな発生元からのエラー (コントロール エラー、アプリケーション エラー、機能ブロック エラーなど) が、機械全体のファンクションブロック "Machine" によって自動的に集められ、ファンクションブロック "Machine" のステータス構造を使用して利用可能になります。

複数のエラーが同時に発生した場合は、最も深刻なものが表示されます。同じ深刻度のエラーが複数、同時に発生した場合は、最後に検出されたものが出力されます。

機械のステータス構造には、以下に説明されている診断目的の構造要素が含まれています。

要素	タイプ	説明
エラー	BOOL	TRUE はエラーを示します
DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの: 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) (xxxx = ユーザーが定義可能) 16#F0FB 0000: ファンクションブロック エラー (例: TechFB でのエラー) -> 詳細については、ErrorID / ErrorIdent を参照 16#F0AA 0000: GAT 内部エラー -> 詳細については、ErrorID / ErrorIdent を参照 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)
ErrorID	ERROR_CODE	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
ErrorIdent	ERROR_STRUCT	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
DiagMessage	STRING(60)	ドライブおよびコントロールのエラー診断テキストを含む

DiagNo 要素は、HMI での使用のみを目的としています。エラー テキストには、任意の言語で数値情報が割り当てられます。

DiagMessage、ErrorID および ErrorIdent は、IndraLogic 表示の使用など、プログラマーによる試運転の際に主に表示される追加情報を提供します。

アプリケーション固有のエラーに関する注意

集中状態の機械での GAT

mSetAppError

メソッド mSetAppError(Number) を呼び出すことにより、アプリケーション エラー (例: 安全扉が開いている) を示すことができます。メソッド入力で伝達される"番号"は、アプリケーション エラー番号に対応します。

アプリケーション エラーが表示された場合、モジュール/機械の中央診断システムは、表示されたアプリケーション エラーの周期的なモニタリングを行い、アプリケーション エラー DiagNo = Error 16#F0AE xxxx (xxxx = アプリケーション エラー番号) を生成します。

uiMAX_MODULE_APP_ERROR_INDEX の範囲 1 にある番号は、モジュールでのアプリケーション エラーについて有効です。定数 "uiMAX_MODULE_APP_ERROR_INDEX" の値はデフォルトにより 64 に設定されています。必要な場合は、"Machine" フォルダにあるグローバル変数リスト "Global_GAT_Machine" でこの定数を調整できます。

uiMAX_MACHINE_APP_ERROR_INDEX の範囲 1 にある番号は、機械でのアプリケーション エラーについて有効です。定数 "uiMAX_MACHINE_APP_ERROR_INDEX" の値はデフォルトにより 64 に設定されています。必要な場合は、"Machine" フォルダにあるグローバル変数リスト "Global_GAT_Machine" でこの定数を調整できます。

アプリケーション エラーを表示するためのコード例:

```
// ユーザー固有のエラーを発行するための推奨コード (例)
IF bDoor10open THEN
    mSetAppError(1);
END_IF
```

深刻なエラーに関する注意

コントロール診断システムでは、10 段階の"深刻度"が定義されています (「IndraMotion MLC diagnostic system」 (IndraLogic XLC-/IndraMotion MLC 診断システム) を参照)。深刻度はエラーリアクションに影響を及ぼします (41 ページ "エラーリアクション" 5.6.4 章 を参照)。

- DiagNo = 16#F0xx xxxx は最も低い深刻度です (例: アプリケーション エラー、ファンクションブロック エラー)
- DiagNo = 16#F9xx xxxx は最も高い深刻度です (致命的なシステム エラー)

複数のエラーが同時に発生した場合は、深刻度が最も高いエラーが表示されます。

ファンクションブロック エラーに関する注意**mSetFBError**

呼び出されたファンクションブロック (技術 FB など) でエラーが発生した場合は、メソッド "mSetFBError(ErrorID, ErrorIdent)" を呼び出して、エラーを引き起こしたファンクションブロックの "ErrorID" と "ErrorIdent" 出力をメソッドに転送することをお勧めします。これにより、エラーがモジュール/機械の診断システムに確実に転送され、"ErrorID" と "ErrorIdent" がモジュール/機械の診断出力で利用可能になります。

ファンクションブロック エラーを表示するためのコード例:

```
// 使用した FB がエラーを発行する場合の推奨コード (例)
IF fbCrossCutter.Error THEN
    mSetFBError(ErrorID:=fbCrossCutter.ErrorID,
                ErrorIdent:=fbCrossCutter.ErrorIdent);
END_IF
```

10.9.2 エラーリアクション

エラーリアクションは、コントロールおよびドライブ システムにおけるエラーによる損傷を可能な限り避けるように設計されています。このため、深刻度 2 以上のエラーリアクションは、制御ファームウェアと実在の軸によって自動的に実行されます (41 ページ "エラーリアクション" 5.6.4 章 を参照)。

集中状態の機械での GAT

深刻度が第2段階未満のその他のエラー (アプリケーション エラー、ファンクションブロック エラーなど) の場合、制御ファームウェアと実在の軸による自動エラー反応はありません。

このため、GAT には、ユーザー固有のエラーリアクションを開始するエラー検出機能が含まれています。

通常の GAT エラーリアクション

デフォルト設定では、GAT エラーリアクションは有効になっています。エラーが発生すると、選択されている運転モードが中断され (例外: 初期化)、エラーリアクションが実行されます。エラーリアクションは、すべての GAT モジュールの "actError" アクションに含まれています。各モジュールは対応するコンポーネント (軸など) のエラーリアクションを担当します。

モジュール アクション *actError* (すべてのモジュール軸が減速) でのエラーリアクションの標準的なケース

```
// ----- エラーリアクションの例 1 -----
// すべてのモジュールの軸を減速して停止させます --> すべての軸を ModeAH に切り替えます
// -----
FOR uiLoopIndex := 1 TO Axes.uiAxesNum DO
  IF mGetAxis(AxisNumber:=uiLoopIndex, PointerToAxis => pCurrentAxis) THEN
    IF pCurrentAxis^.AxisData.Axis_InAb THEN
      pCurrentAxis^.Ctrl.Admin._OpMode.en := ModeAH;
      // SetupMode 入力をリセットし、
      // System_Overview_All_Axis 表示にあるボタンのハングアップを防ぎます
      pCurrentAxis^.Ctrl.SetupMode.Enable := FALSE;
      pCurrentAxis^.Ctrl.SetupMode.JogMinus := FALSE;
      pCurrentAxis^.Ctrl.SetupMode.JogPlus := FALSE;
      pCurrentAxis^.Ctrl.SetupMode.Homing := FALSE;
    END_IF
  END_IF
END_FOR
```

上記の例で "ModeAH" が "ModeAb" で置き換えられた場合、減速で静止した後で実在の軸のトルク有効化がアクティブになります。



"ModeAb" を使用して油圧軸を静止させることをお勧めします。



"軸が一時停止している場合"、軸の降下を防ぐためにトルクが有効化されないことがあります。これらのアプリケーションでは、すべての軸を "ModeAH" に切り替えることをお勧めします。

軸を同期させて減速する必要があることも少なくありません (材料ウェブの断裂を防ぐためなど)。これらのアプリケーションでは、同期させた動きですべての軸を接続したままにすること、およびマスターのみを減速することをお勧めします。このユース ケースは下記の例に示されています。

モジュール アクション *"actError"* (同期していない軸のみが "ModeAH" を使用して減速される) での軸の同期減速の例:

```
// ----- エラーリアクションの例 2 -----
// すべての軸をマスター軸と同期した状態に保ち、
// and decelerate only the master axis
// -----
FOR uiLoopIndex := 1 TO Axes.uiAxesNum DO
  IF mGetAxis(AxisNumber:=uiLoopIndex, PointerToAxis => pCurrentAxis) THEN
    // SetupMode 入力をリセットし、
    // System_Overview_All_Axis 表示にあるボタンのハングアップを防ぎます
    pCurrentAxis^.Ctrl.SetupMode.Enable := FALSE;
    pCurrentAxis^.Ctrl.SetupMode.JogMinus := FALSE;
    pCurrentAxis^.Ctrl.SetupMode.JogPlus := FALSE;
    pCurrentAxis^.Ctrl.SetupMode.Homing := FALSE;
    IF (pCurrentAxis^.AxisData.SynchronizedMotion) THEN
      ; // 軸は同期モーション中です --> 応答はありません。なぜなら、マスター軸が
      // 減速して停止し、スレーブ軸もそれに同期して停止するためです
    ELSE
      IF pCurrentAxis^.AxisData.Axis_InAb THEN
        pCurrentAxis^.Ctrl.Admin._OpMode.en := ModeAH;
        ; // 軸は同期モーション中ではありません --> 命令して停止させます
      END_IF
    END_IF
  END_IF
END_FOR
```

集中状態の機械での GAT

```

END_IF
END_IF
END_IF
END_FOR

```

実在の軸の同期減速には、以下の設定が必要です。

- モジュール バス反応をオフにする
- ドライブのエラーリアクションを "可能な最善のシャットダウン = 30 s NC 対応" により設定する



IndraLogic XLC/IndraMotion MLC コントロール システムの実在の軸の "可能な最善の減速 = 30s NC 対応" 設定で F2 エラーが発生した場合、エラーの影響を受ける軸は緊急停止ランプ A-0-0224 により減速されます。問題のない他のすべての軸は、影響を受けません。

通常の GAT エラーリアクションをオフにする

特別な場合、通常の GAT エラーリアクションをオフにすることができます。これを行うには、変数宣言を通じて変数 **stMachine1Ctrl_gb** を以下のとおり変更します。

- グローバル変数リスト: "Global_GAT_Machine"
- 変数の既存の宣言 stMachine1Ctrl_gb を下記のコードで置き換えます。

```

stMachine1Ctrl_gb:          TE_MACHINE_CTRL          :=
(DisableErrorHandling := TRUE);

```

エラー反応がオフになっても、現在の運転モードは中断されません。しかし、エラーは継続して表示され、クリアすることも可能です。

初期化中の GAT エラーリアクション

デフォルト設定では、初期化中、GAT エラーリアクションは無効になっています。しかし、エラーは継続して表示され、クリアすることも可能です。

特別な場合、初期化中に GAT エラーリアクションをオンにすることができます。

これを行うには、変数宣言を通じて変数 **stMachine1Ctrl_gb** を以下のとおり変更します。

- グローバル変数リスト: "Global_GAT_Machine"
- 宣言を以下のとおり変更します:

```

stMachine1Ctrl_gb:          TE_MACHINE_CTRL          :=
(EnableErrorReactionInit := TRUE);

```

エラーリアクションが有効な場合、エラーが発生すると初期化が中断されます。その後、プログラミングされたエラーリアクションが実行されます。エラーリアクションは、各"モジュール"の "actError" アクションでプログラミングされます。



初期化中のエラーリアクションは、通常の GAT エラーリアクションが有効な場合 (DisableErrorHandling = FALSE) のみ実行されます。

エラーのフィルタリング

機械のすべてのエラーが必ずしもシステム全体のシャットダウンにつながるわけではありません。このため、エラーは機械によって一元的に処理することも、個別のモジュールによって分散された方法で処理することも可能です。この目的のために、"Machine" ファンクションブロックおよび各モジュールのファンクションブロックで "mlsCriticalError" メソッドが利用できます。

機械がエラーを検出した場合、以下の手順が実行されます。

1. 診断番号 (DiagNo) および、該当する場合はエラーを引き起こした軸が決定されます。

集中状態の機械での GAT

2. FB "Machine" の mIsCriticalError(DiagNo, AxisRef) が呼び出されます。
mIsCriticalError() が FALSE を返した場合、エラーは無視されます。
mIsCriticalError() が TRUE を返した場合、次の手順が実行されます。
3. mIsCriticalError(DiagNo, AxisRef) がすべてのアクティブなモジュールで呼び出されます。
この呼出に対してすべてのモジュールが FALSE を返した場合、エラーは無視されます。
少なくとも 1 つのモジュールが TRUE を返した場合、通常のエラー処理が継続されます。

デフォルトにより、"mIsCriticalError()" は TRUE を返します。すなわち、すべてのエラーが重大なエラーに分類されます。この動作は、FB "Machine" またはモジュール FB に "mIsCriticalError" メソッドを実装することにより実現できます。

例：ファンクションブロック "Machine" での "mIsCriticalError()" メソッドの実装

```

METHOD mIsCriticalError : BOOL
VAR_INPUT
    DiagNo: DWORD;      // 診断番号
    Axis: AXIS_REF;      // 軸
END_VAR

CASE DiagNo OF
    // 重大ではない私のエラー
    16#F0AE0001,
    16#F0AE0002: mIsCriticalError := FALSE;
ELSE
    // 他のすべてのエラー
    mIsCriticalError := TRUE;
END_CASE

```

例：モジュールファンクションブロックでの mIsCriticalError() メソッドの実装

```

METHOD mIsCriticalError : BOOL
VAR_INPUT
    DiagNo: DWORD;      // 診断番号
    Axis: AXIS_REF;      // 軸
END_VAR

mIsCriticalError := TRUE;

IF Axis.AxisNo = 0 THEN
    // 軸が不明なエラー
    mIsCriticalError := FALSE;
ELSIF mIsMyAxis(Axis) THEN
    // 私自身の軸でのエラー
    IF DiagNo=16#FFAA0002 THEN
        mIsCriticalError := FALSE;
    END_IF
END_IF

```

10.9.3 エラーのクリア

一元的なエラーのクリアは、機械制御構造の構造要素 "ClearError" の正のエッジにより自動的に実行されます。システム全体を問題のない状態にするために、"Machine" ファンクションブロックにより、すべての下位コンポーネント (例：IMC インターフェース¹⁾、モジュール、AxisInterface など...) にコマンドが転送されます。構造要素 "enOpModeCtrl_ErrorSuccessor" (機械制御構造を参照) により、エラーのクリア後に有効になる運転モードが決定されます。

¹⁾ *IndraMotionControl* インターフェース = 制御へのインターフェースを備えた *IndraLogic* ライブラリ。例：エラーをクリア

集中状態の機械での GAT

10.9.4 プロトコル エラー

- 機械制御構造で構造要素 **"LogError"** が設定されている場合 (*TRUE*)、以下の GAT 固有エラーは、コントロールのエラー/診断メモリにも入力されます。
- 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) (xxxx = ユーザーが定義可能)
 - 16#F0FB 0000: ファンクションブロック エラー (技術 FB でのエラーなど)
 - 16#F0AA 0000: GAT 内部エラー
 - 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)

 "LogError" が TRUE に設定されている場合、GAT 固有エラーはコントロール ディスプレイおよびコントロールの診断パラメータにも表示されます。

10.10 表示

10.10.1 試運転サポートのための内部表示

一般情報

GAT には、試運転中の支援および拡張された診断機能を提供する、予め定義済みの IndraLogic 表示オブジェクトが含まれています。

AXIS_GAT

表示要素 "AXIS_GAT" は、個々の軸についてインスタンス化できます。
これは "MyVirtualAxis1" 軸および "MyVirtualAxis2" 軸の "MAIN_VISU" 表示で例として実行されています。

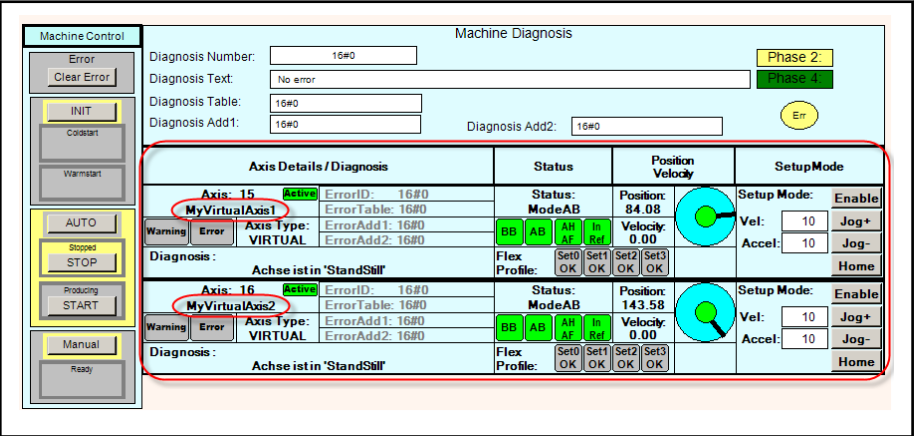


図 10-36: 起動中の支援のための内部表示

"Frame" オブジェクトを使用して、表示内で表示オブジェクト "AXIS_GAT" のインスタンスを作成することができます。

集中状態の機械での GAT

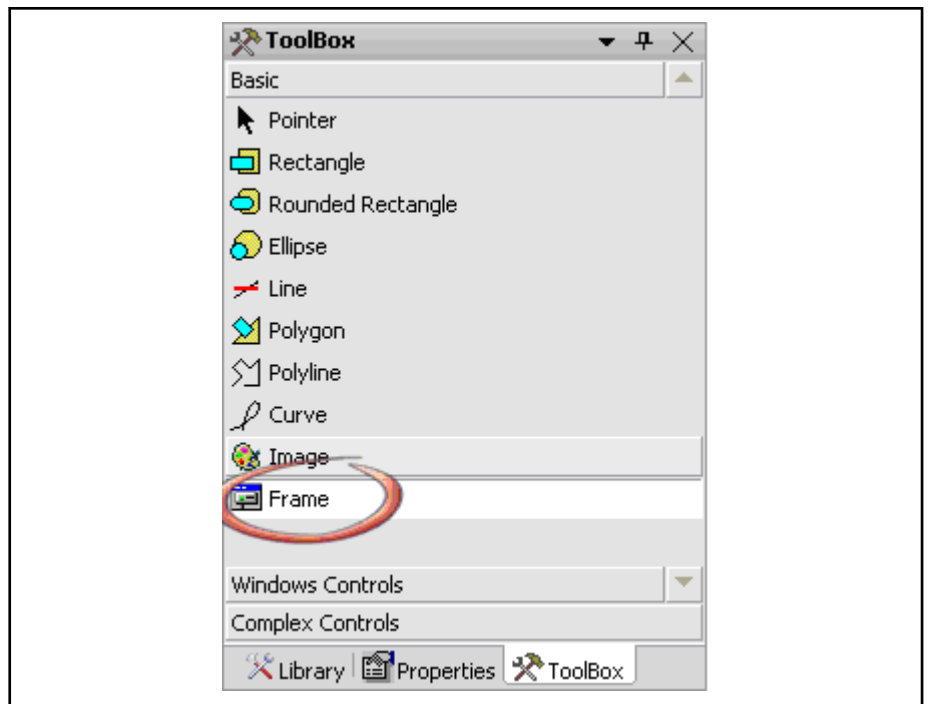


図 10-37: 表示オブジェクト "AXIS_GAT" のインスタンスの作成

ここで、対応する軸参照を「Properties」の「References」の項目に入力する必要があります。

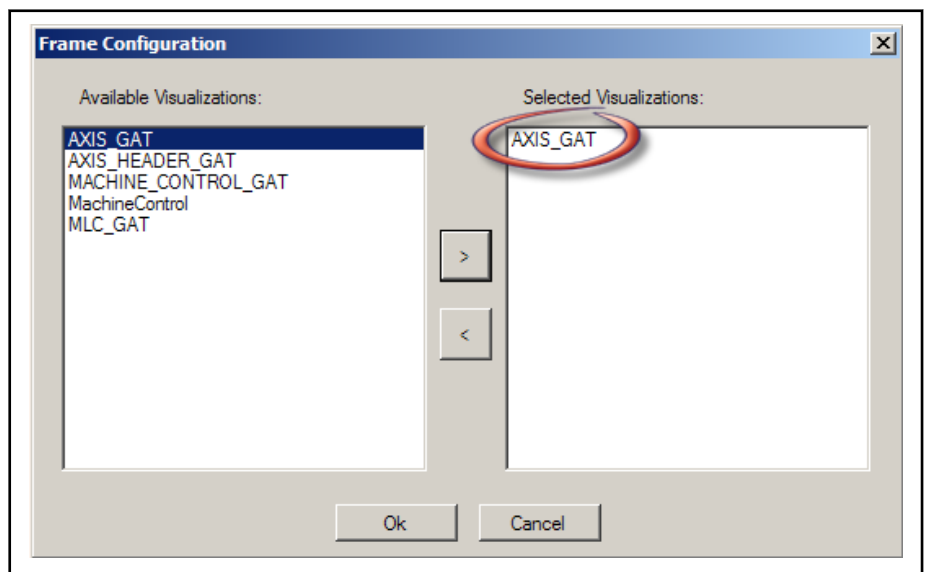


図 10-38: 表示オブジェクト "AXIS_GAT" のインスタンスの作成

10.10.2 独立した HMI による外部表示 (WinStudio 表示)

一般情報

GAT では予め定義済みの WinStudio 表示が利用できます。ウィザードを使用して、この表示をプロジェクトに追加できます。

集中状態の機械での GAT



WinStudio を編集するには、少なくとも以下の開発環境のライセンス (またはそれ以上) が必要です。

WinStudio 開発ライセンス、変数最大 1,500 個 (Mat.No. R911323596)

VEP でプログラムを実行するには、以下のランタイム ライセンスが必要です。

オペレーティング システムとして Windows CE を使用しているデバイス用の WinStudio ランタイム ライセンス、変数最大 1,500 個 (Mat. No. R911323620)

WinStudio 表示の追加



WinStudio 表示をプロジェクトに追加するためには、対応する表示デバイスがプロジェクトになければなりません。

"ドラッグ & ドロップ"を使用して、デバイス ライブラリからプロジェクト ツリーに VEP を追加し、以下のとおり設定します。

- デバイス タイプ：VEPxx (CE)
- アプリケーション タイプ：WinStudio アプリケーション

HMI Wizard

General Settings
Please select the device type, the application templates and the panel design.
Enter a device name and the IP address of the target device.

Device type:
VEP40 (CE)

Name:
VEP40

IP address / Computer name:
192.168.1.11

Application type:
WinStudio application

Panel design:
Flat

Device properties

Display: 12.1" TFT, 800x600, SVGA

Preview of F-Key

Finish Cancel Help

図 10-39: 表示デバイスの追加

集中状態の機械での GAT

対応する表示デバイスがある場合、下記のリンクとウィザードを使用して WinStudio 表示を追加します。

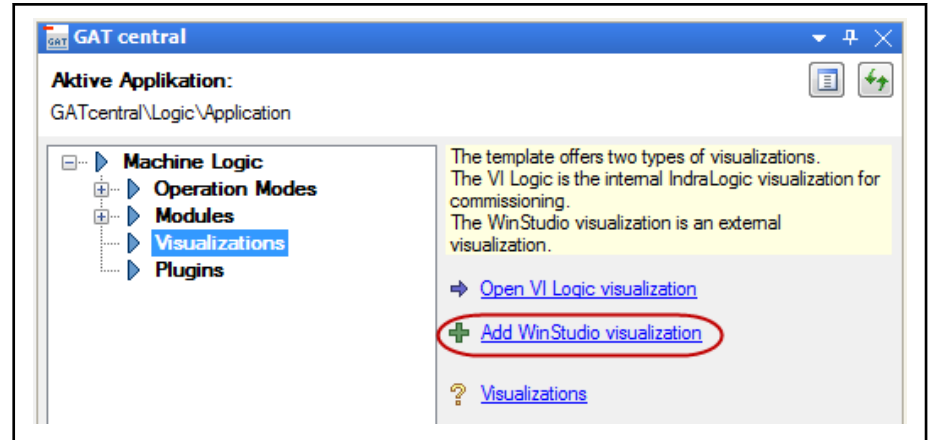


図 10-40: WinStudio 表示の追加

各設定に対応するプロジェクトの表示デバイスはすべて下記のウィザード ダイアログに含まれています。

対応する表示デバイスを選択した後で、**Apply** (適用) ボタンをクリックして WinStudio 表示を追加します。

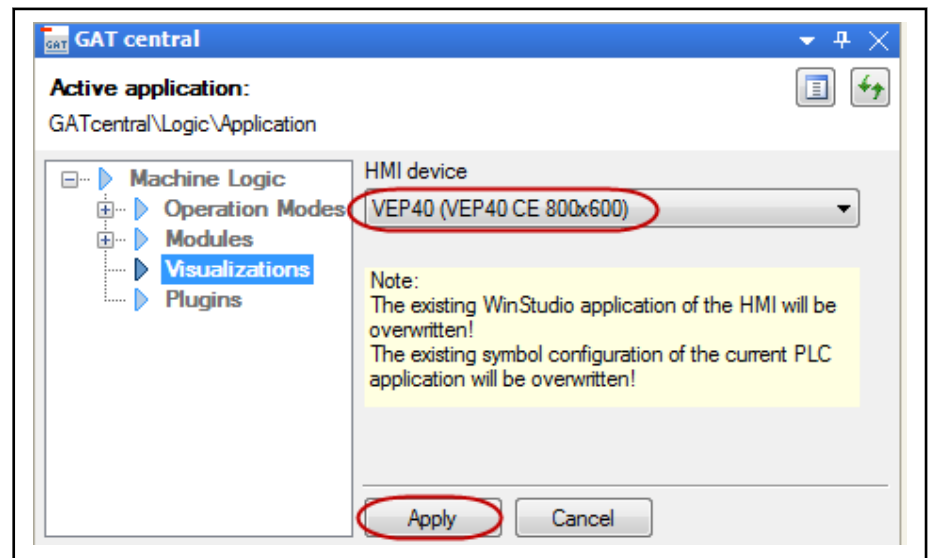


図 10-41: 表示デバイスの選択 – WinStudio 表示の追加



WinStudio 表示が追加されると、コントロールの記号設定とともに表示デバイスの既存の表示データが上書きされます。

表示が追加された後で、表示データを編集できます。

集中状態の機械での GAT

コントロール アドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

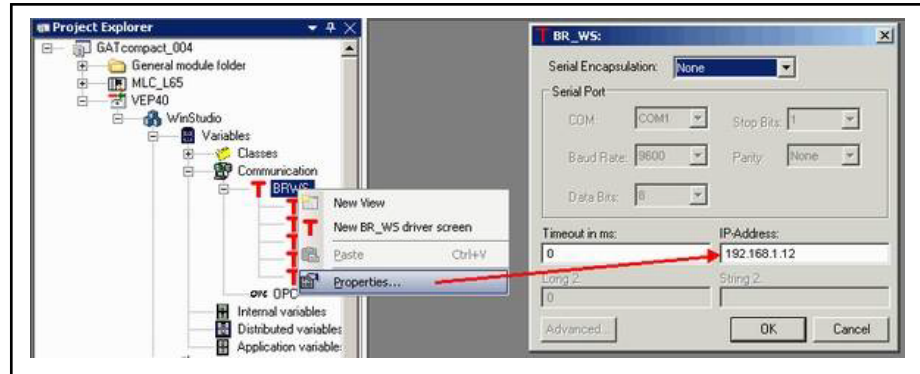


図 10-42: コントロールの IP アドレス設定

ここで、WinStudio ノードのコンテキスト メニューにある  "Start application" (アプリケーションを起動) を使用して WinStudio 表示のエミュレーションを開始します。

ダイアログ構造

操作画面は以下の 3 つのセクションに分かれています。

- ステータス情報があるヘッダー
- メニュー関係の表示と機能キーがあるメイン画面
- ユーザー マップの切替を行うためのキーがあるフッター

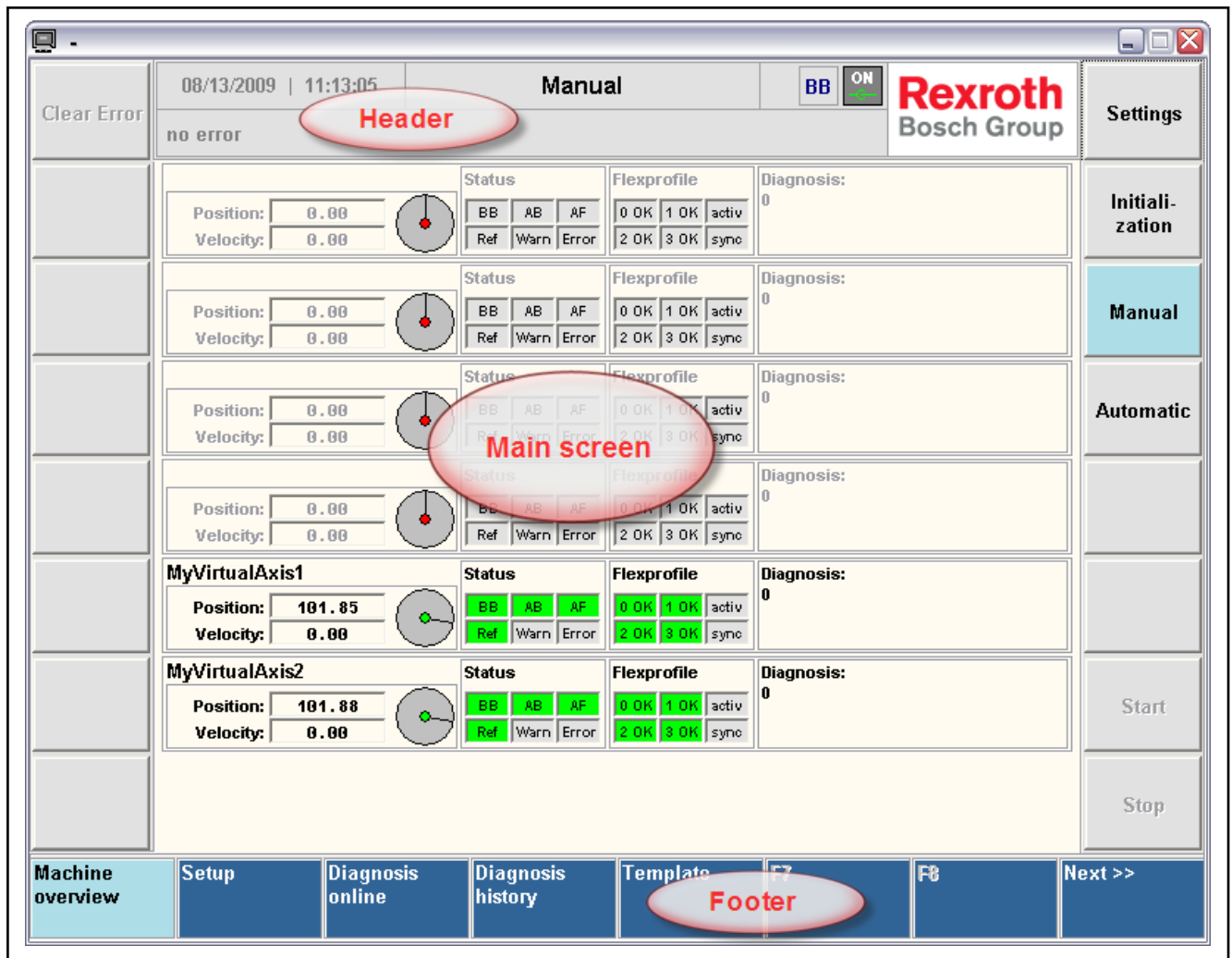


図 10-43: WinStudio 表示

ヘッダーには以下が含まれています：

- 運転モードの表示
- "エラーを削除"したり "Settings" (設定) ダイアログ (言語の切替、数値表示の小数点位置の設定などのため) を開いたりするためのグローバル機能キー
- 現在の機械診断の表示
- 現在の sercos フェーズの表示 (アクティブなフェーズ切替を含む)
- 通信ステータスの表示

メイン画面には、操作画面 (軸の情報、運転モードの切替など) に関連する表示と機能キーが含まれています。

フッターには、ユーザー マップを切り替えるためのキーが含まれています。

表示の拡張

操作画面の追加

集中状態の機械での GAT

表示を拡張するには、既存の"テンプレート"画面をコピーして新しい名前を付けて挿入することにより、新しい操作画面を作成できます。

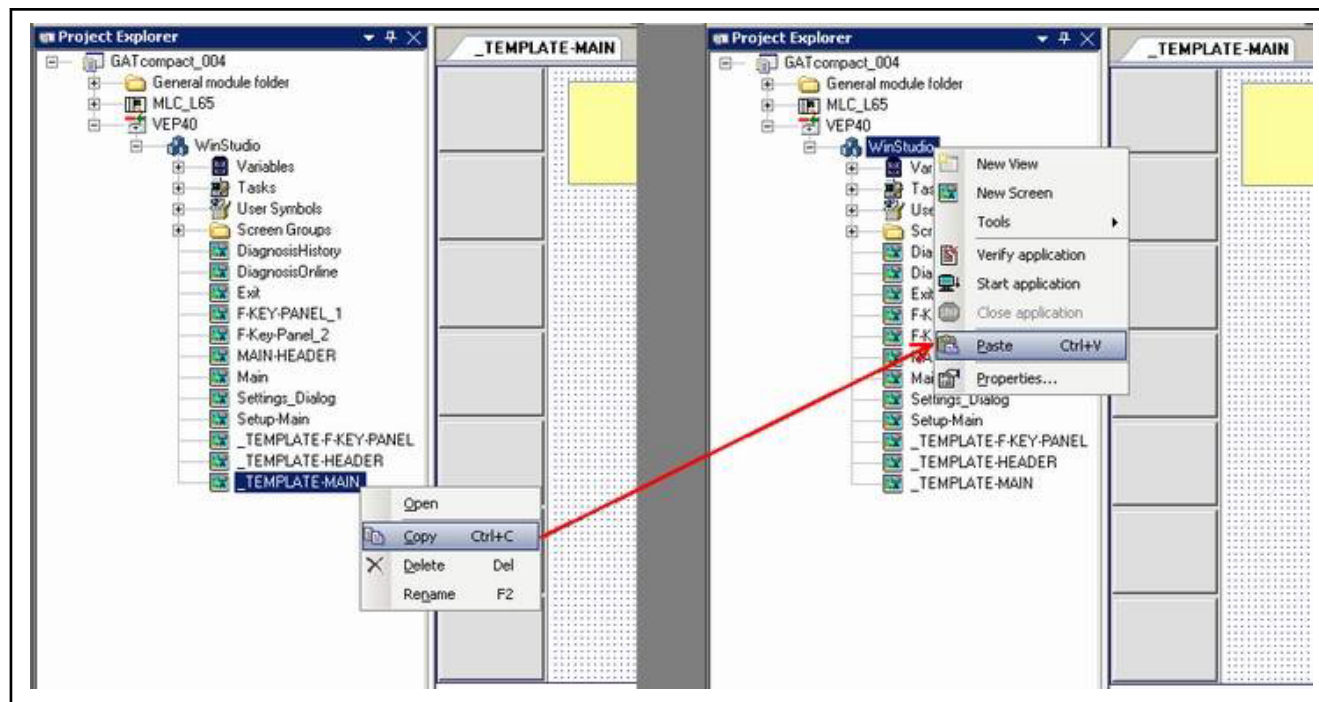


図 10-44: 操作画面の複製

軸データを表示するための要素は、"コピー" & "ペースト"で複製できる、グループ化された個別のオブジェクトです。

異なる軸のために挿入されたオブジェクトを再設定するには、各オブジェクトに「置換」機能を使用して軸配列インデックスを適合させる必要があります。



図 10-45: 挿入されたオブジェクトの再設定

集中状態の機械での GAT

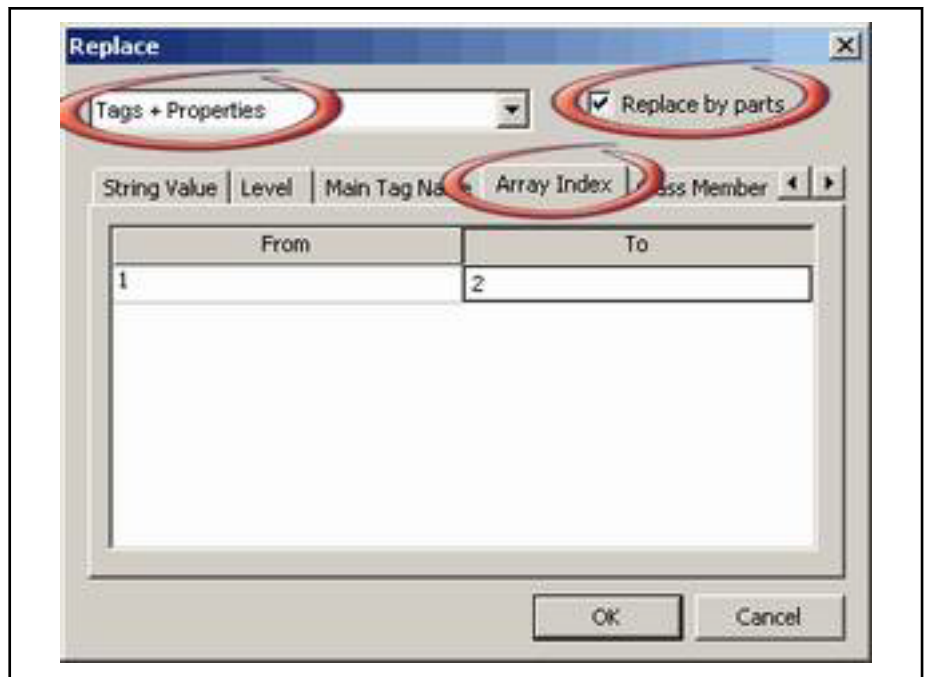


図 10-46: 挿入されたオブジェクトの再設定

フッターの拡張

ユーザー マップを切り替えるために使用されるキーは、パラメータ化が可能なグラフィック オブジェクト (リンクされた記号) として保存されます。キーには以下の転送パラメータがあります。

- Caption -> キー ラベル
- Disable -> キー機能をブロックするための入力 (グレー表示のラベル)
- ScreenName -> 操作画面を開くための画面名 (または画面グループ)
- ScreenNumber -> 画面に割り当てられる番号 (それぞれのキーを強調表示するため)

通信接続

表示とコントロール間の通信は、BR-WS ドライバを使用して行われます。これを行うには、表示する PLC 変数の記号エントリを生成します (47 ページ "通信設定" 章 を参照)

集中状態の機械での GAT

予め定義済みの WinStudio 表示に必要な記号設定には、以下のエントリが含まれます。

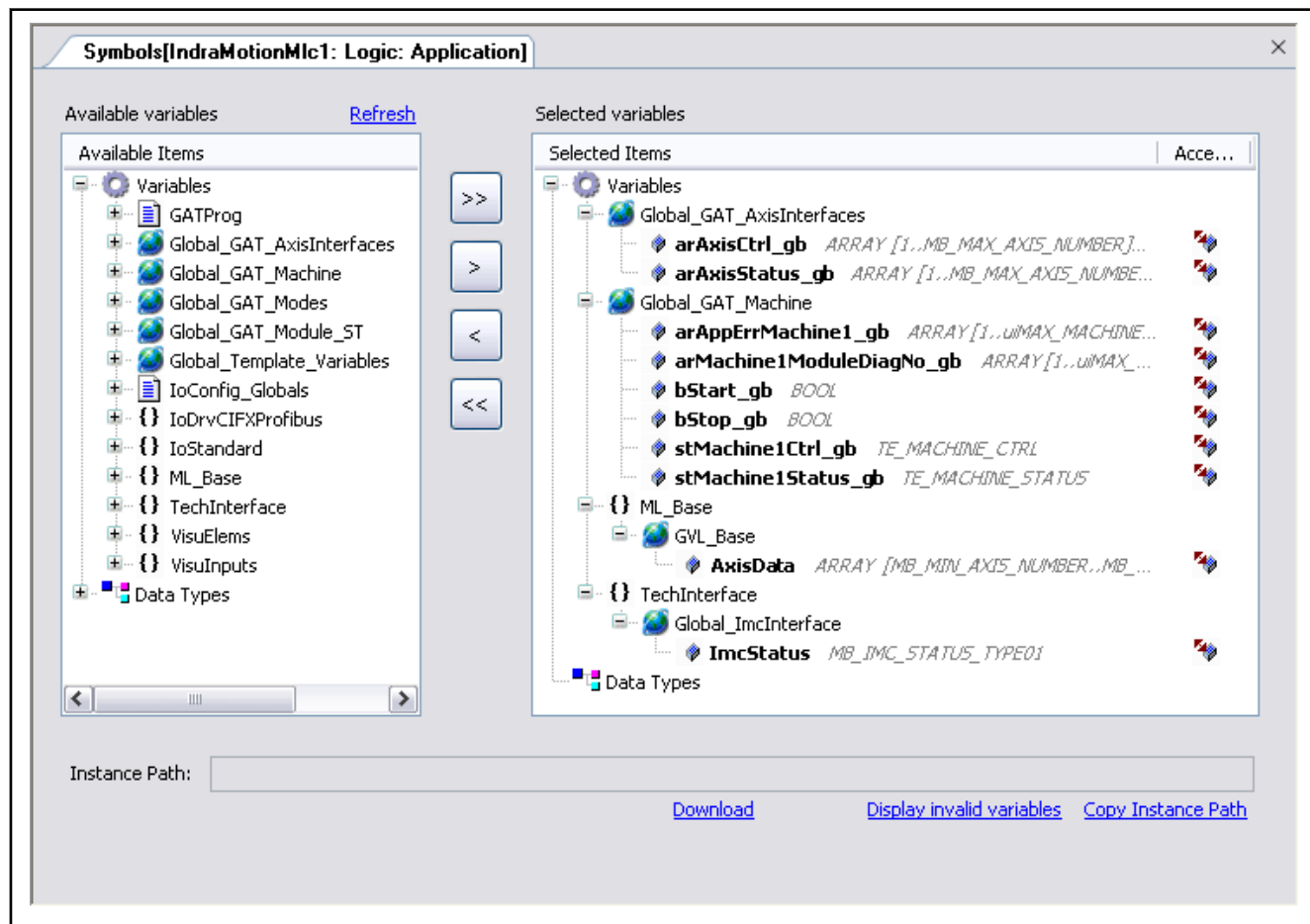


図 10-47: 予め定義済みの WinStudio 表示の記号設定



ライブラリから取得したグローバル変数リストの変数の記号エントリは、各ライブラリのバージョンが変更された後で記号設定に挿入しなおす必要があります。

コントロールの IP アドレス設定 :

コントロール アドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

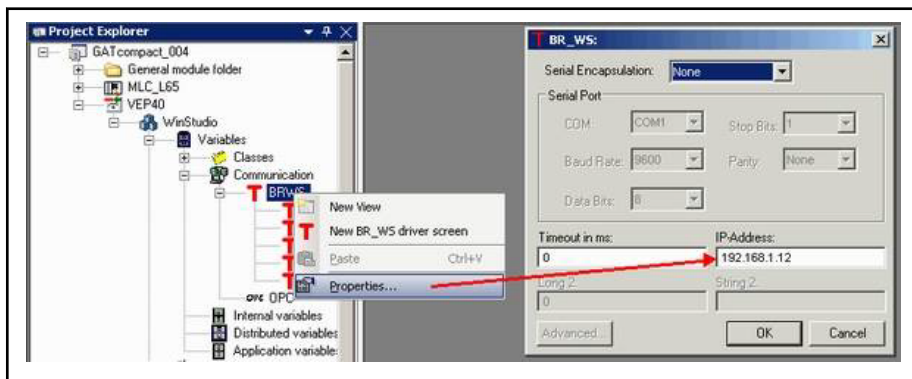


図 10-48: コントロールの IP アドレス設定

集中状態の機械での GAT

通信を構成するために、変数がさまざまなドライバシートに配置されています。

1. 高速で周期的に読み取る：-> これらの変数の値は短い間隔で周期的に読み取られます (診断番号、軸の位置など)
2. 1 回のみ読み取る：-> これらの変数は表示が開始された後で 1 回のみ読み取られます (軸の名前など)
3. フェーズの切替後に読み取る：これらの変数はフェーズの切替後に 1 回のみ読み取られます (軸がアクティブになった場合など)
4. 周期的に読み書き：-> これらの変数の値は、表示により読み書きの両方が行われます (機械の速度、セットアップ モードの軸の速度など)
5. 書き込みのみ：これらの変数は、表示を通じて読み取りのみが行われます (エラーの削除の信号を発信する...)

ドライバシートを拡張するために、BR-WS ブラウザでオフライン記号ファイル (*.XML) を検索できます。

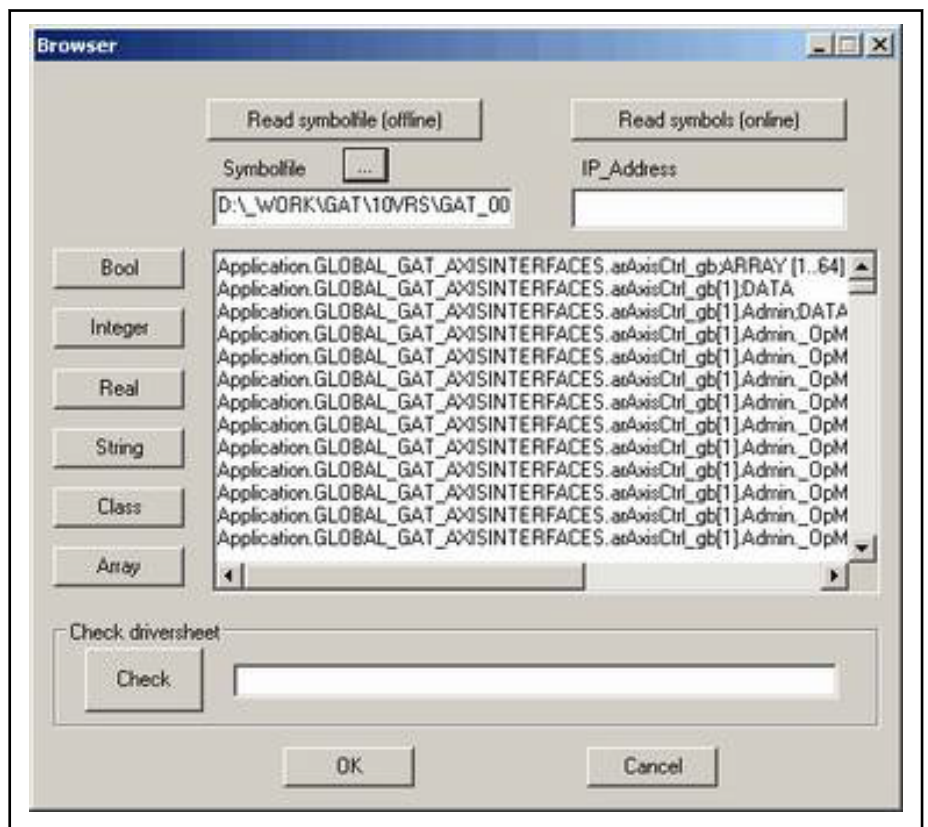


図 10-49: ドライバシートの拡張

複数言語機能

表示の複数言語機能は、IndraWorks への翻訳機能の組み込みにより実現されました。

コントロールおよびドライブの診断の翻訳データ、ならびに表示テキストは、すでに準備されています。翻訳データは IndraWorks で変更や修正が可能です。

ドライブ診断はドイツ語、英語、フランス語、スペイン語、トルコ語で含まれています。

他のすべてのテキストはドイツ語と英語です。

集中状態の機械での GAT

10.11 GAT のインターフェース

10.11.1 一般情報

GAT インターフェース データはさまざまな変数リストで利用可能です。データの一部はライブラリで利用可能になっています。

10.11.2 機械データ

機械の入力データは、タイプが TE_MACHINE_CTRL. の stMachine1Ctrl_gb (Global_GAT_MachineControl) 構造にまとめられています。

下の表は、stMachine1Ctrl_gb 構造の要素を説明したものです。

I/O タイプ	名称	データの種類	説明
I	OpMode	TE_OPMODES_PREDEFINED	機械運転モードの仕様
I	OpMode_InitSuccessor	TE_OPMODES_PREDEFINED	正常な初期化後の機械運転モード
I	OpMode_ErrorSuccessor	TE_OPMODES_PREDEFINED	エラーのクリアが正常に行われた後の機械運転モード
I ^{*)}	StateCtrl	TE_STATES_PREDEFINED	内部使用/機械の状態
I	SkipColdstartAtStartup	BOOL	GAT を起動するとき、アプリケーションに必要なパラメータ化を実行するために、パラメータ化モード (P2) への切替が行われます。アプリケーションに P2 でのパラメータ化が不要な場合、このフェーズ切替は TRUE で省略できます。
I	DisableErrorHandling	BOOL	TRUE = エラーが発生した場合、プロセスは "ERROR" 状態に切り替わりません。エラー表示は続行され、エラーのクリアは可能です。
I	ClearError	BOOL	正のエッジによるグローバル クリア エラー
I	LogError	BOOL	TRUE = GAT 固有エラー (アプリケーション、ファンクションブロック、GAT およびプラグインエラー) はコントロールのエラー/診断メモリにも保存されています

*) ユーザーによる記述不可。この入力の設定は、機械構成ユニットのみが可能。

I 書き込み用入力

機械の出力データは、タイプが TE_MACHINE_STATUS. の stMachine1Status_gb (Global_GAT_MachineControl) 構造にまとめられています。

下の表は、stMachine1Status_gb 構造の要素を説明したものです。

I/O タイプ	名称	データの種類	説明
O	OpMode	TE_OPMODES_PREDEFINED	現在の機械運転モードに関するフィードバック
O	State	TE_STATES_PREDEFINED	内部使用/現在の機械の状態に関するフィードバック
O	エラー	BOOL	TRUE = グローバル エラーが発生しました
O	ErrorID	ERROR_CODE	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報

集中状態の機械での GAT

I/O タイプ	名称	データの種類	説明
O	ErrorIdent	ERROR_STRUCT	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの: DiagNo = 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー DiagNo = 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)
O	DiagMessage	STRING(60)	コントロール/軸の診断テキスト

O 読み取り用出力

10.11.3 エラー処理用データ

エラー処理の入力および出力は、上述の機械構造に含まれています。

10.11.4 モジュール データ

モジュールは機械構成ユニットにより制御されます。モジュール データは、診断目的のために外部からアクセスできます。

すべてのモジュールの入力データは、ファンクションブロック bMachine1_gb (Global_GAT_MachineControl) 内にあるタイプ TE_CMI_CONTROL の arModuleCtrl 列にまとめられています。

下の表は、Global_GAT_MachineControl.fbMachine1_gb.arModuleCtrl 構造の要素を説明したものです。

I/O タイプ	名称	データの種類	説明
I	Enable	BOOL	TRUE = モジュールの処理を開始
I	ClearError	BOOL	TRUE = モジュール クリア エラー
I ^{*)}	State	TE_STATES_PREDEFINED	モジュールの状態用コマンド
I ^{*)}	StateTransient	BOOL	TRUE = モジュールは命令された状態への移行中
I ^{*)}	DisableTransient	BOOL	TRUE = モジュールの無効化が開始されました

I 書き込み用入力
*) ユーザーによる記述不可。この入力の設定は、機械構成ユニットのみが可能。



arModuleCtrl の要素は "Machine" FB に書かれているため、外部の位置からは読み取るのみが可能です。

すべてのモジュールの出力データは、ファンクションブロック bMachine1_gb (Global_GAT_MachineControl) 内にあるタイプ TE_CMI_STATUS の arModuleStatus 列にまとめられています。

下の表は、Global_GAT_MachineControl.fbMachine1_gb.arModuleStatus 構造の要素を説明したものです。

集中状態の機械での GAT

I/O タイプ	名称	データの種類	説明
O*)	InOperation	BOOL	TRUE = モジュールは処理されます
O*)	State	TE_STATES_PREDEFINED	モジュールの状態に関するフィードバック
O	StateLocked	BOOL	TRUE = 次の状態への切替がユーザーによってブロックされています
O	TransientLocked	BOOL	TRUE = 命令された状態の静止セクションへの移行がユーザーによってブロックされました
O*)	TransientState	TE_STATES_PREDEFINED	現在モジュールが切り替わろうとしている状態の表示 (命令された状態は未達成、関連するアクティビティは実行中)
O	DisableLocked	BOOL	TRUE = モジュール無効化のアクティビティはまだ実行中です
O*)	エラー	BOOL	TRUE = モジュール エラーが発生しました
O*)	ErrorID	ERROR_CODE	DiagNo = 16#F0FB 0000 の場合の詳細情報
O*)	ErrorIdent	ERROR_STRUCT	DiagNo = 16#F0FB 0000 の場合の詳細情報
O*)	DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの: DiagNo = 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)
O*)	DiagMessage	STRING(60)	コントロール/軸の診断テキスト

O*)

読み取り用出力
ユーザーによる記述不可。この出力の設定は、機械構成ユニットのみが可能。

10.11.5 軸データ

I/O タイプ	名称	データの種類	説明
I	arAxisCtrl_gb (Global_GAT_AxisInterfaces)	ARRAY OF TE_AXIS_CONTROL_TYPE 01	軸用のコマンド
O	arAxisStatus_gb (Global_GAT_AxisInterfaces)	ARRAY OF TE_AXIS_STATUS_TYPE01	軸からのフィードバック
I/O	AxisData (ML_Base ライブラリからの GVL_Base)	ARRAY OF ML_AXISDATA_SM	軸の入力と出力のあるファームウェア構造

I

O

I/O

書き込み用入力
読み取り用出力
入力と出力のある構造



arAxisCtrl_gb の要素は "Machine" FB に書かれているため、外部的に使用することはできません。

10.11.6 コントロール データ

I/O タイプ	名称	データの種類	説明
I	ImcCtrl (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_CONTROL_TYPE01	コントロール用のコマンド
O	ImcStatus (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_STATUS_TYPE01	コントロールからのフィードバック

I 書き込み用入力
O 読み取り用出力



ImcCtrl_gb の要素は "Machine" FB に書かれているため、外部的に使用することはできません。

10.12 操作手順：一般的なユーザー アクティビティ

10.12.1 軸データへのアクセス

軸データへのアクセスについて、すべてのモジュールで、モジュールに割り当てられたすべての軸の一元的なアクセス ポイントとして **Axes** 構造が定義されます。軸のすべての関連データは、Axes 構造を使用してアドレスを指定できます。

この構造は、軸の名前を用いて軸データにアクセスできるように設計されています。これにより、読み取りが容易なプログラム コードが生成されます。

"Axes" の項目では以下のデータが利用可能です。

- Axes.<Name>.AxisData => 軸データ構造：ファームウェアのステータス データおよび軸データ コンテナへのアクセス
- Axes.<Name>.Ctrl => AxisInterface の制御構造
- Axes.<Name>.Status => AxisInterface のステータス構造
- Axes.<Name>.DV => 直接変数 (機能変数) へのアクセス
- Axes.<Name>.AxisRef => ファンクションブロック呼出の軸参照構造 (MB_ReadParameter など)

集中状態の機械での GAT

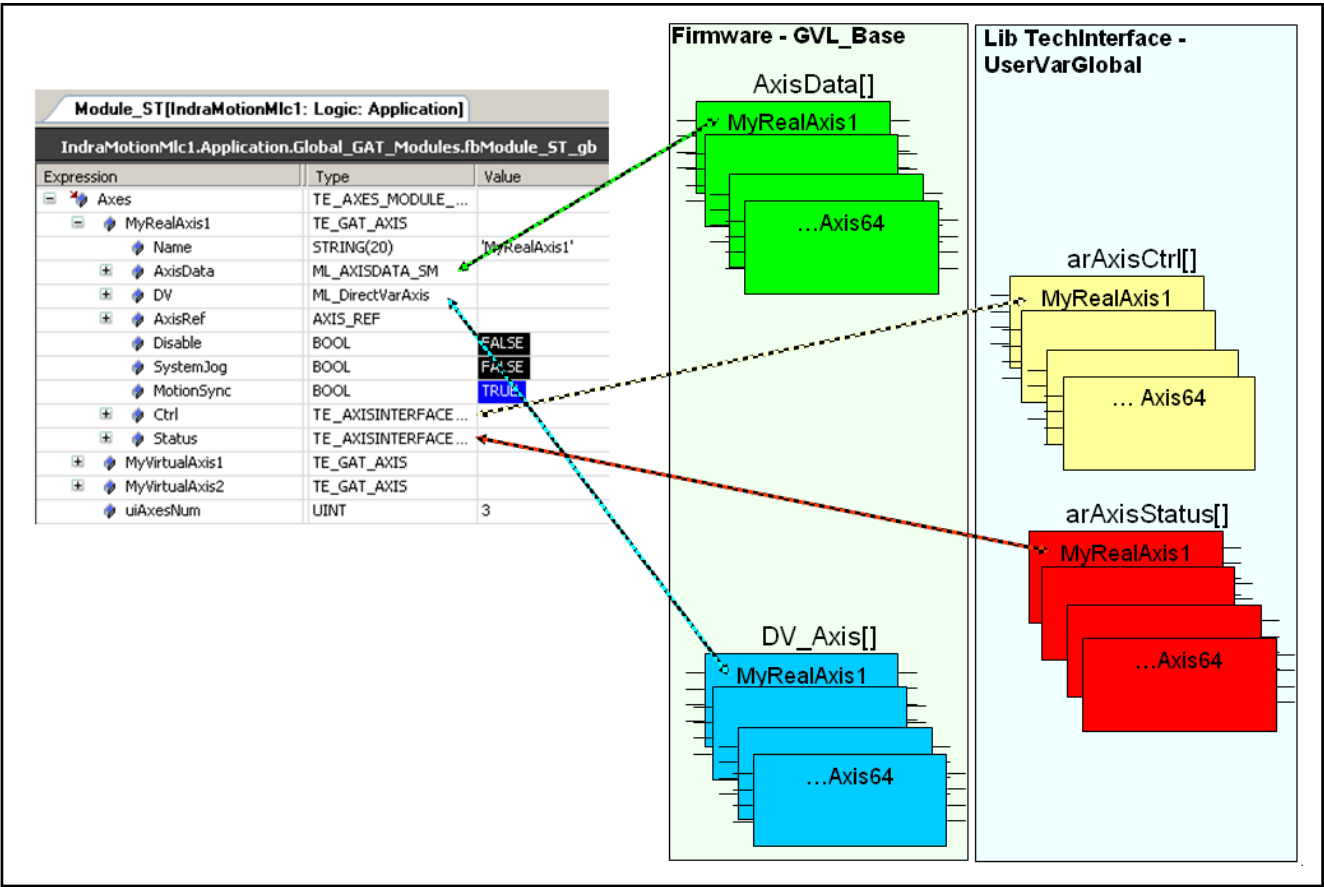


図 10-50: 概要/軸データへのアクセス

モジュールのすべての軸にわたるループをプログラミングするために、各モジュールにはメソッド **mGetAxis()** および要素 **Axes.uiAxesNum** があります。

例: "ModeAb" へのすべてのモジュール軸の切替

宣言

```
pCurrentAxis:    POINTER TO TE_GAT_AXIS;  
uiLoopIndex:     UINT; // For loop index
```

実装

```
// ----- Error reaction example 1 -----  
// Decelerate all module axes to standstill and turn power off  
// --> switch all axes off to ModeAb  
// -----  
FOR uiLoopIndex := 1 TO Axes.uiAxesNum DO  
  IF mGetAxis(AxisNumber:=uiLoopIndex, PointerToAxis => pCurrentAxis) THEN  
    pCurrentAxis^.Ctrl.Admin._OpMode.en := ModeAb;  
  END_IF  
END_FOR
```

10.12.2 軸の追加

- 軸を追加するためには、以下の手順が必要です。
- IndraWorks プロジェクトに軸を追加する (sercos ノードにドラッグする)
 - "SystemOverview_All_Axis" 表示を使用する場合: 新しい軸のフレームを追加し、プロパティで新しい軸への参照を設定する (233 ページ "軸の追加" 8.10.2 章 も参照)
 - GAT ウィザードを使用して軸を追加する (下の図を参照)

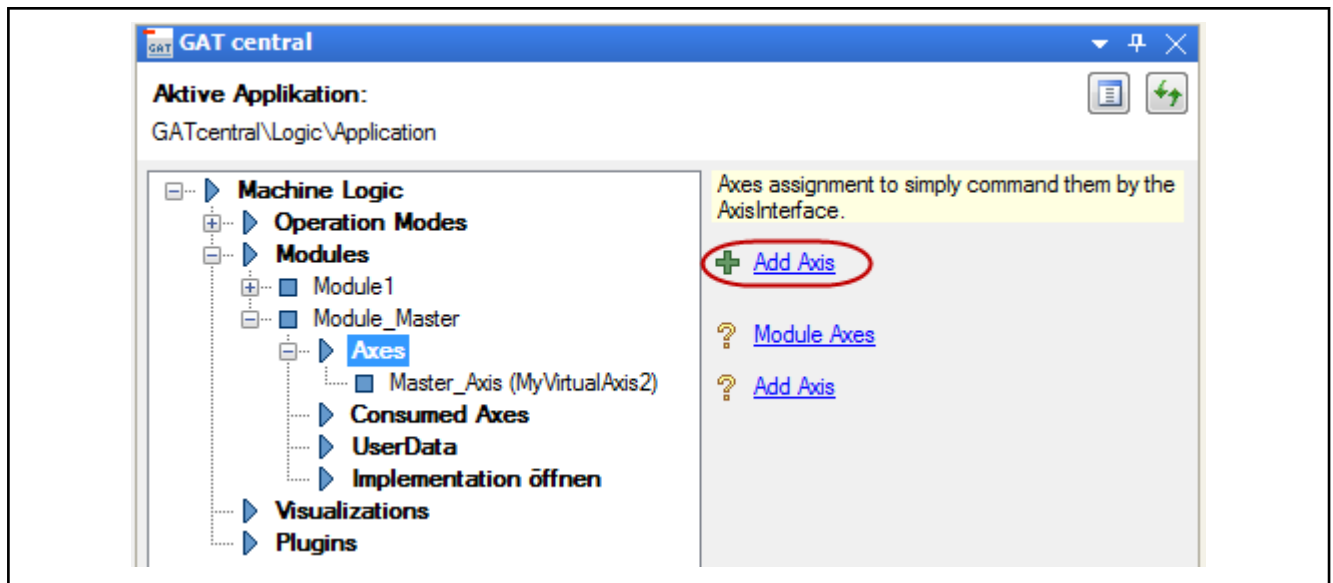


図 10-51: GAT ウィザードを使用した軸の追加

10.12.3 軸の削除/名前変更



ウィザードによって変更されるのは、軸宣言にある名前のみです。軸がすでに実装または内部表示で使用されている場合、新しい軸の名前は、"検索・置換"を使用して、これらの位置で変更する必要があります。

軸の削除/名前変更を行うためには、以下の手順が必要です。

- ndraWorks プロジェクトから軸を削除する/ndraWorks プロジェクトで名前を変更する
- "SystemOverview_All_Axis" 表示を使用する場合：削除した軸のフレームを削除し、プロパティで軸への参照を変更する
- GAT ウィザードを使用して軸を削除する/名前を変更する ("Properties" (プロパティ) を介した軸の名前変更は、下の図を参照)

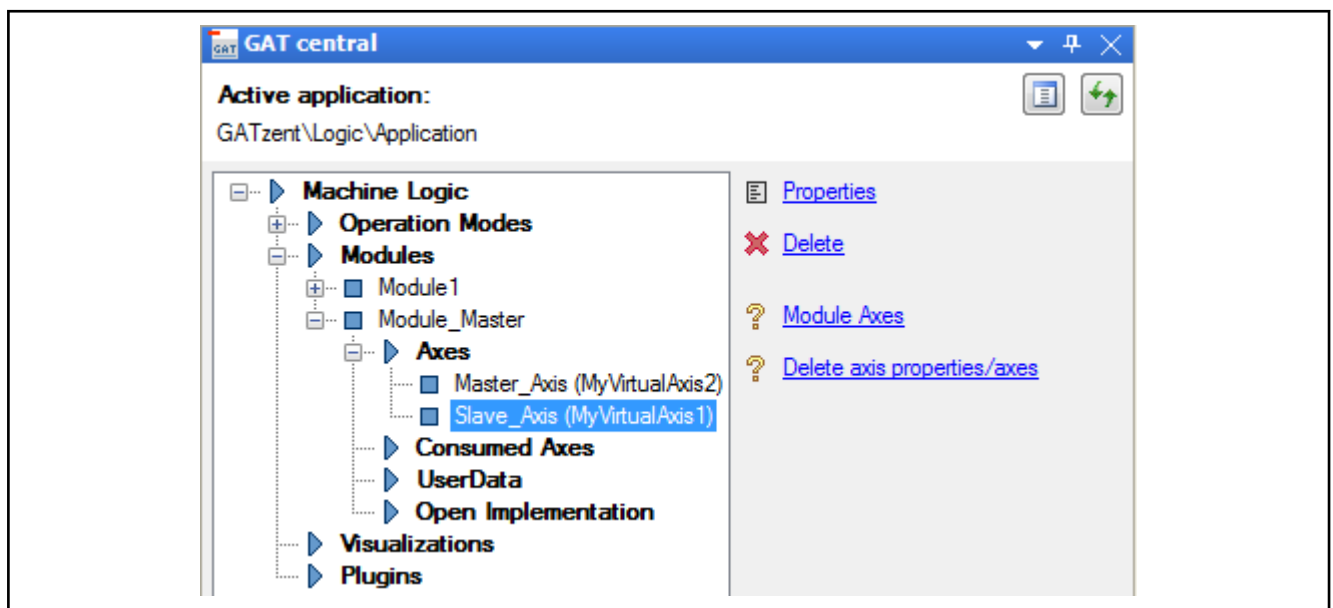


図 10-52: GAT ウィザードを使用した軸の削除/名前変更

集中状態の機械での GAT

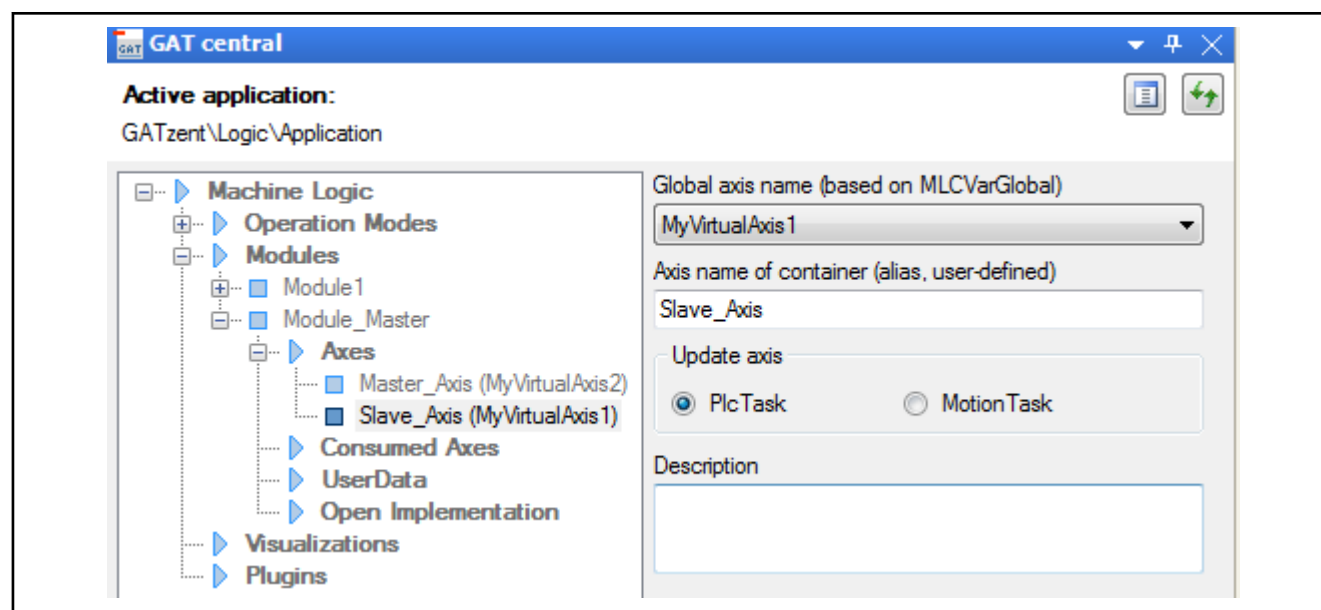


図 10-53: GAT ウィザードを使用した軸の削除/名前変更

10.12.4 運転モードの追加

GAT ウィザードを使用すると、機械への運転モードの追加が容易になります。また、ウィザードは運転モードの削除や名前変更にも使用できます。

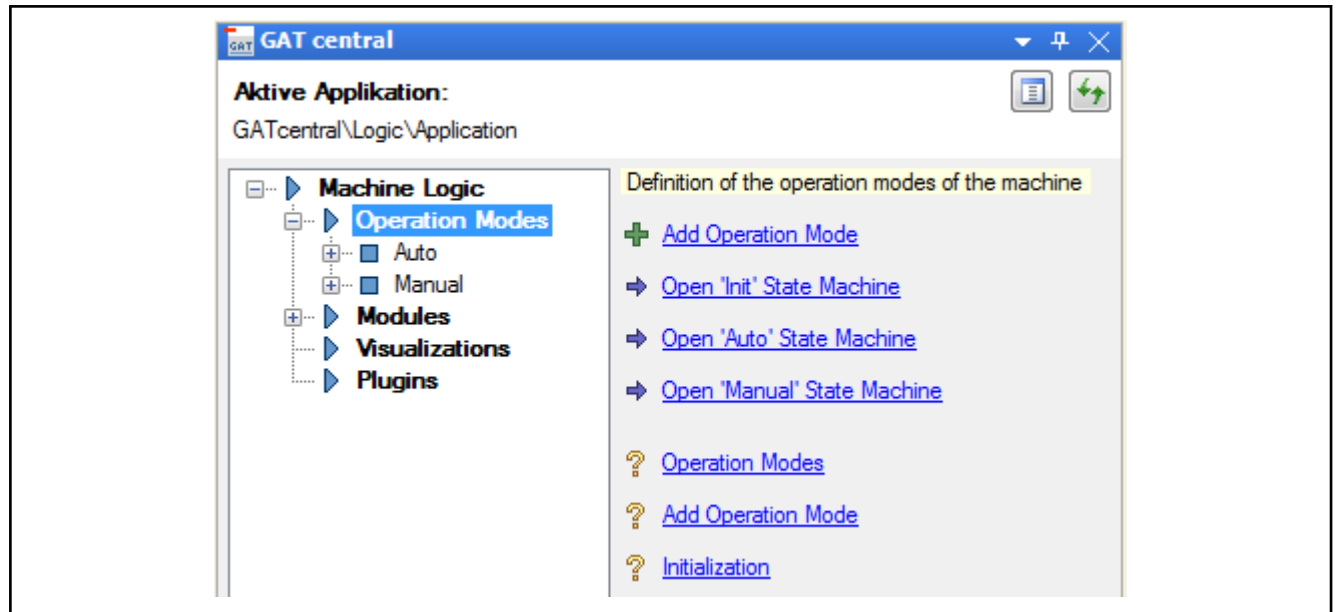


図 10-54: GAT ウィザードによる運転モードの追加

10.12.5 エラー クリア後の必要な運転モードの選択

変数 `fbMachine_gb.Ctrl. OpMode_ErrorSuccessor` は、エラーをクリアした後のアプリケーションの動作に影響を及ぼします。ユーザーはこの変数の値をいつでも変更できます。変数は、エラーをクリアした後に機械構成ユニットによって直ちに評価されます。

デフォルトにより、この変数の値は "STATE_UNDEFINED" です。この条件下でエラーがリセットされた場合、アプリケーションはエラー状態を終了し、STATE_INIT_WARMSTART 状態から、以前に有効だった STOPPED (停止) 運転モード状態に移行します。

例えば、エラーが発生する前に "AUTO" (自動) 運転モードの "PRODUCING" (生産中) 状態が有効でした。エラーが削除された場合、アプリケーションは STATE_INIT_WARMSTART 状態に切り替わり、続いて STATE_AUTO_STOPPED 状態に切り替わります。

変数 `fbMachine_gb.Ctrl. OpMode_ErrorSuccessor` の値が特定の運転モードである場合、エラーがクリアされた後でアプリケーションはこの運転モードの STOPPED (停止) 状態に入るコマンドを受けます。

例えば、エラーが発生する前は "AUTO" (自動) 運転モードの "PRODUCING" (生産中) 状態が有効でした。また、"`fbMachine_gb.Ctrl. OpMode_ErrorSuccessor=OPMODE_MANUAL`" が適用されます。エラーがクリアされた後で、アプリケーションは STATE_MANUAL_STOPPED 状態に直接切り替わります。

INIT の初期状態は STATE_INIT_COLDSTART です。

アプリケーションが STATE_INIT_COLDSTART 状態または STATE_INIT_WARSTART 状態にある間にエラーが発生した場合、エラーがクリアされた後で、アプリケーションはこの状態に再び戻ります。

集中状態の機械での GAT

10.12.6 運転モード仕様の上位ロジック

場合によっては、機械の運転モードの選択は追加的な条件に依存します。このロジックを実装するために推奨される場所は、"Machine" FB を呼び出す前のプログラム "GATProg" です。グローバル変数 `stMachine1Ctrl_gb.OpMode` を使用して、条件をクエリし、運転モードを指定することができます。

10.12.7 状態と移行の追加

運転モード内で実行されるアクティビティが非常に複雑な場合は、複数の状態への分割が推奨されます。これにより、アプリケーションが明確に構成され、理解しやすくなります。

GAT ウィザードを利用すると、状態および状態間の条件 (移行) の追加が容易になります。

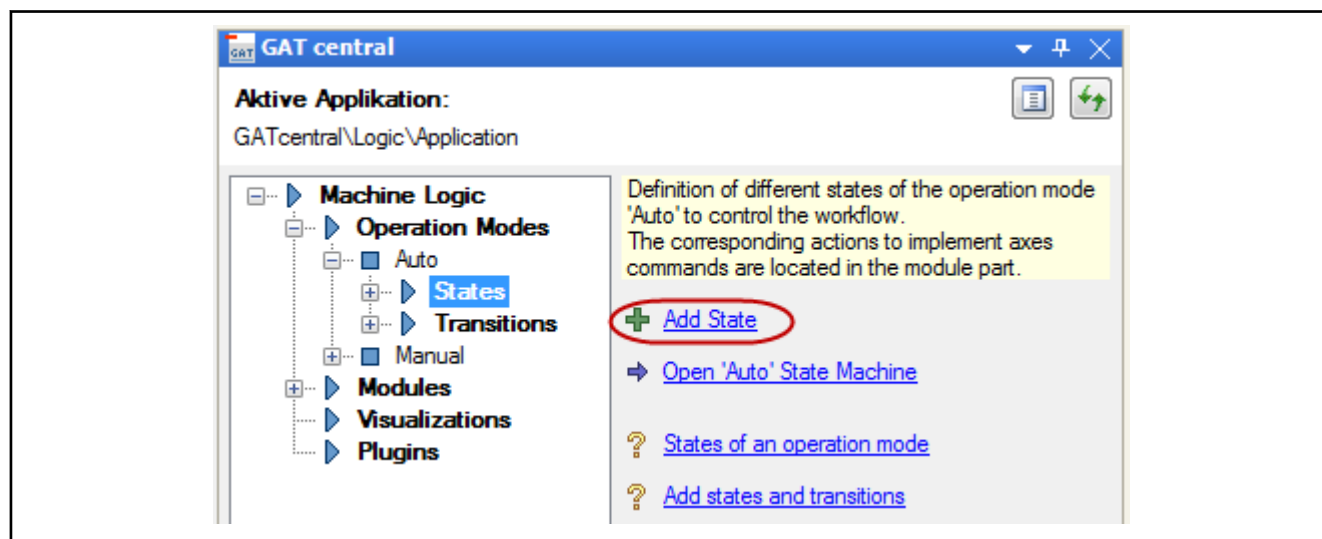


図 10-55: 状態の追加

運転モード内に状態を追加する場合、強調表示された状態の後に新しい状態を追加するのか、あるいは強調表示された条件 (移行) の後に追加するのか決定する必要があります。また、分岐を作成することもできます (前の図を参照)。

新しい状態を生成した場合、その状態に対応するアクションが、その時点で存在するすべてのモジュールで作成されます。

10.12.8 新しいモジュールの追加

GAT ウィザードを使用すると、機械へのモジュールの追加が容易になります。また、ウィザードはモジュールの削除や名前変更にも使用できます。

新しいモジュールでは、その時点で存在するすべての状態で、状態に対応するアクションが作成されます。

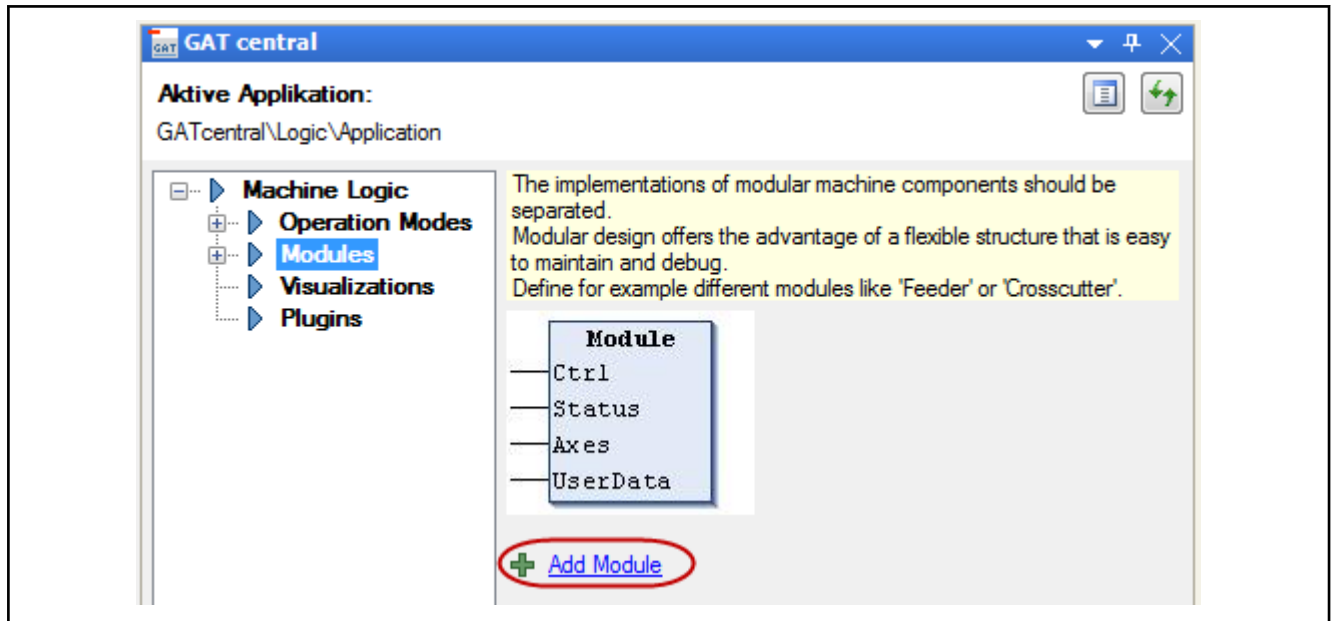


図 10-56: モジュールの追加

集中状態の機械での GAT

10.12.9 モジュールのインスタンスの追加

GAT ウィザードを使用すると、既存のモジュールの新しいインスタンスを機械に簡単に追加できます。

インターフェースは同じですが、他のグローバル軸 ("Axes" への) および I/O データ ("UserData" への) を割り当てる必要があります。

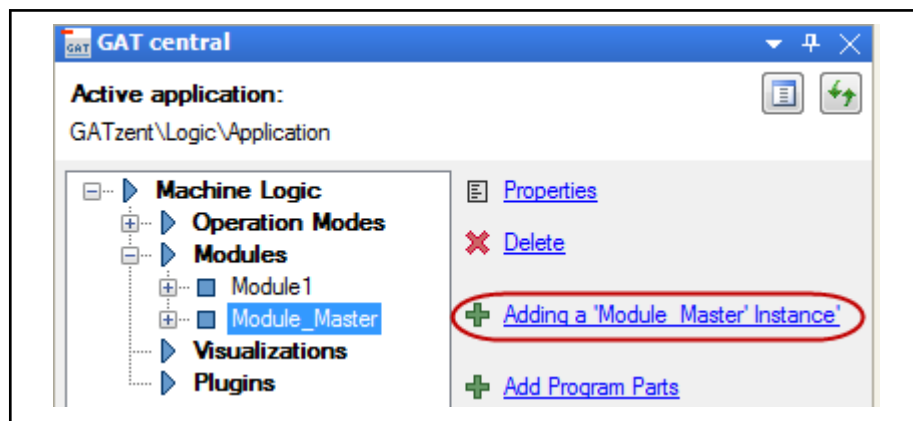


図 10-57: モジュールのインスタンスの追加

10.12.10 モジュールの無効化

モジュールを処理から除外するために、モジュールを無効にできます。モジュールが無効になると、そのモジュール コードは処理されなくなります。モジュールを有効または無効にするには、メソッド "mSetModuleEnable()" が使用できます。

デフォルトにより、GAT ウィザードを使用してモジュールを追加した場合、モジュールは "Machine" ファンクションブロックの "mOneTimeInit" メソッドで有効になります。

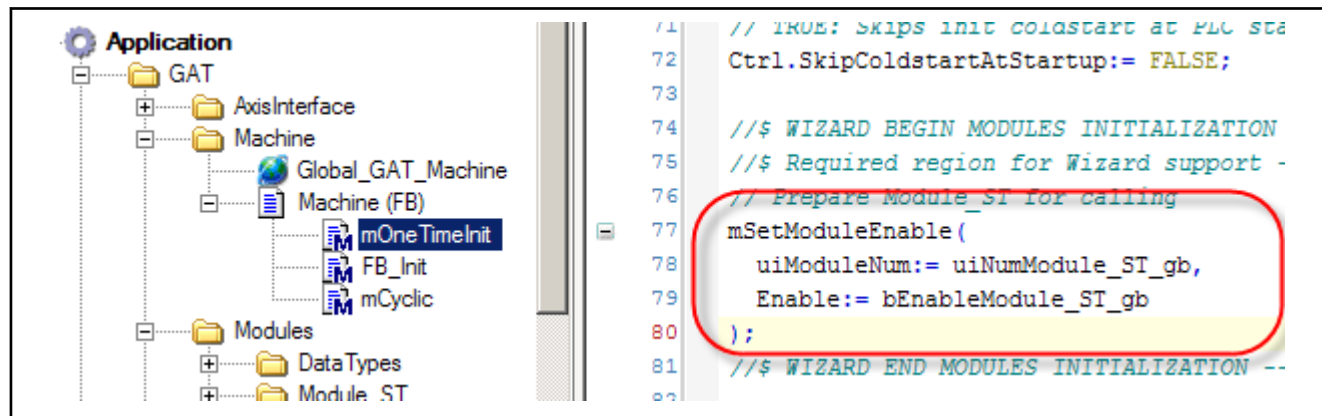


図 10-58: モジュールの有効化/無効化

モジュールは、PLC プロジェクトが再びダウンロードされた直後、またはランタイムの後で無効にできます。"Enable" 変数の初期値を FALSE に変更することにより (下の図を参照)、PLC ダウンロードの直後にモジュールが無効になります。この手順により、利用できないモジュールをスイッチオフにするだけで、さまざまな機械バージョンの場合でも、1 つのプロジェクトのみを維持できます。

集中状態の機械での GAT

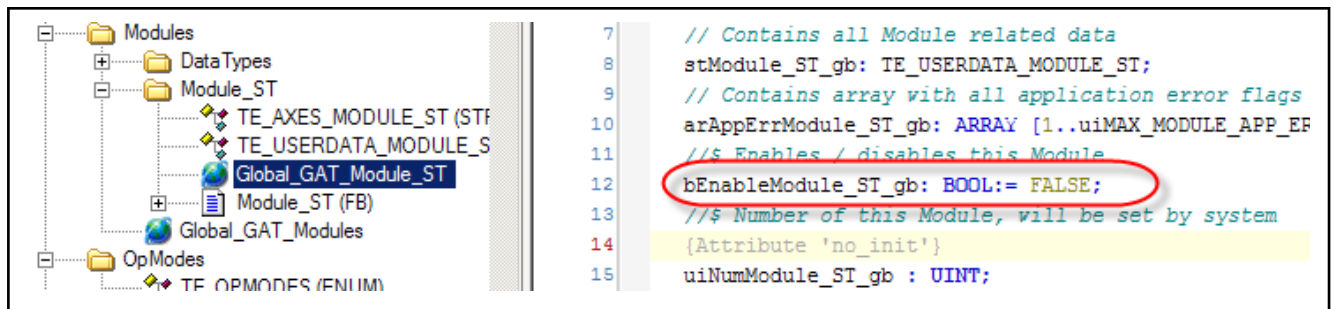


図 10-59: 初期値の変更によるモジュールの有効化/無効化

モジュールは、プログラムが実行中であっても、ランタイム中に無効にできません。

これを行うには、以下の手順を行う必要があります。

- まず、機械（および割り当てられているすべてのモジュール）をセキュアな初期状態（Automatic（自動）運転モードの "STOPPED"（停止）状態など）に設定します。安全関連の理由のため、セキュアな初期機械状態が実現されない場合、モジュールを無効にするためのコマンドは実行されません。
- 次に、"mSetModuleEnable(Enable:=FALSE...)" を使用してモジュール無効化の機能をアプリケーション プログラムから起動します。
- GAT は制御構造モジュールの "DisableTransient" 制御要素を、少なくとも 1 つのサイクルについて TRUE に設定します。
- 続いて、無効になったモジュールでアクション "actDisable" が処理されます。このアクションで、モジュールを無効にするために必要なコードを実装できます。アクション "actDisable" が少なくとも 1 サイクルについて処理されます。モジュールの無効化に複数のサイクルが必要な場合は、モジュール状態要素 "DisableLocked := TRUE" を設定することにより、処理を延長できます。アクション "actDisable" は、"DisableLocked" が再び FALSE にリセットされるまで処理されます。リセットされると、モジュール コードは処理されなくなります。

10.12.11 モジュールの再利用

GAT は、開発された機能ユニット（モジュール）を、GAT を使用して構築された他のプロジェクトでも再利用できるように設計されています。

これを行うには、任意のモジュールのフォルダを選択し、コンテキストメニューを使用して関連データをエクスポートします。

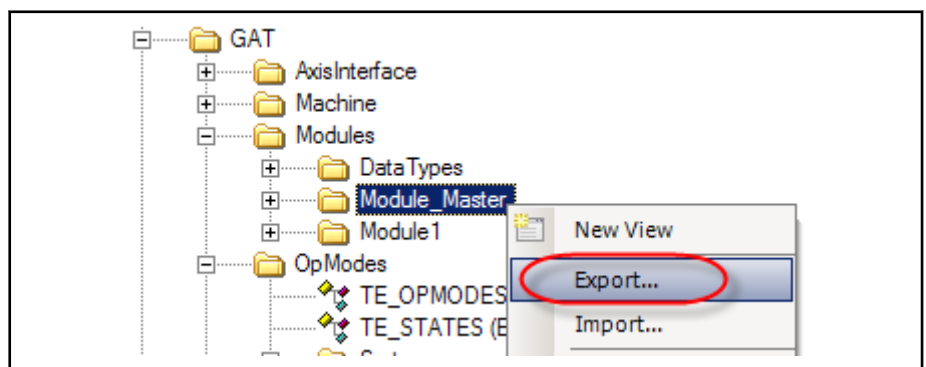


図 10-60: モジュールのエクスポート

その後のダイアログで要素をすべて選択します。

エクスポート中に生成されたファイルを新しいプロジェクトにインポートします。

集中状態の機械での GAT

これを行うには、"Modules" (モジュール) フォルダを強調表示し、関連するコンテキストメニューで "Import" (インポート) 機能を使用します。

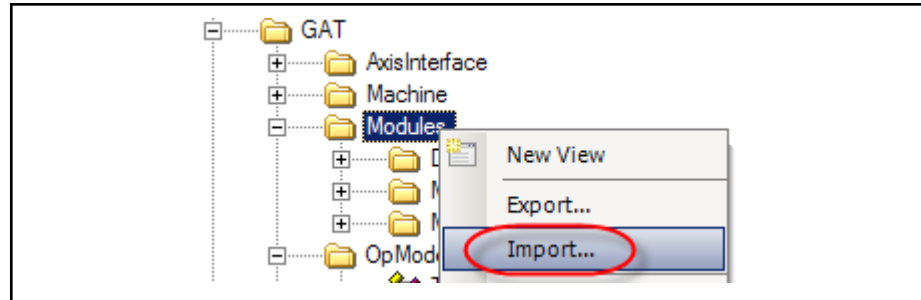


図 10-61: モジュールのインポート

別のプロジェクトでモジュールを使用する場合 (インポート)、そのモジュールに含まれているすべての状態が、新しい機械 (プロジェクト) でも定義されていることを確認します。

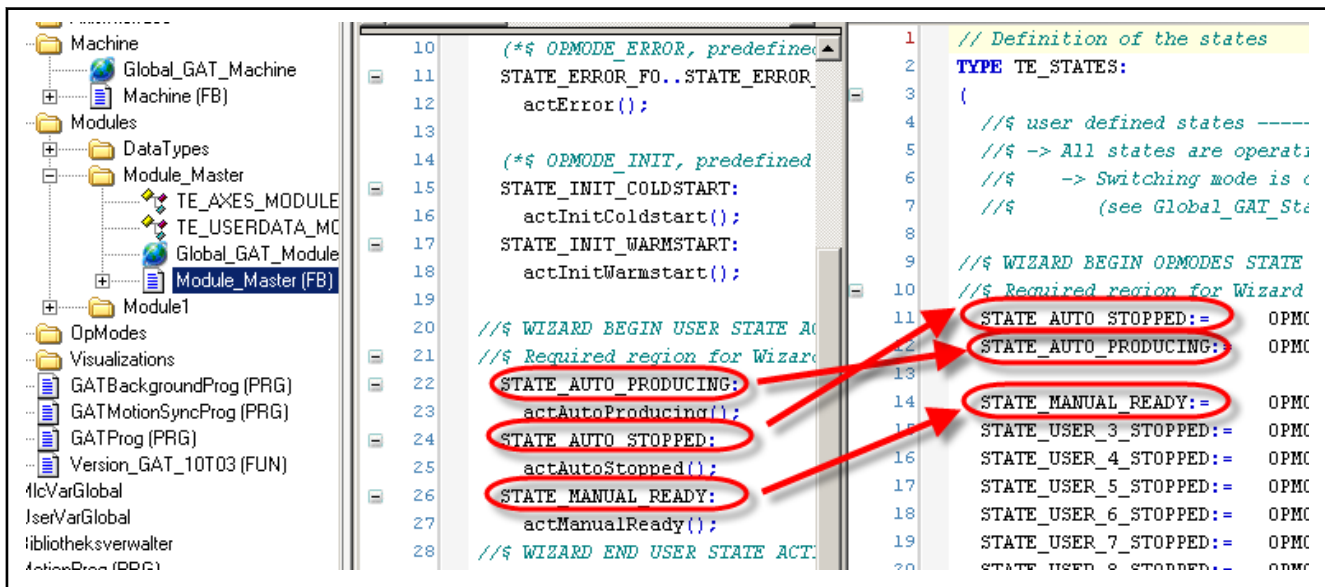


図 10-62: 状態の宣言

モジュールをインポートした後で、モジュールを有効/無効にするために、関連付けられているエントリをファンクションブロック "Machine" の "mOneTimeInit" メソッドに追加する必要があります。

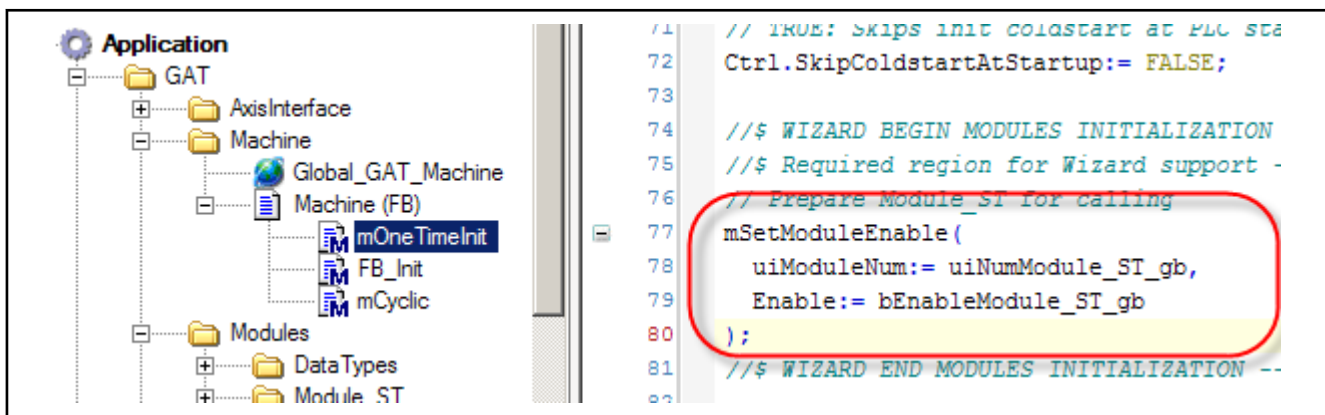


図 10-63: モジュールをインポートした後の FB "Machine" のメソッド "mOneTimeInit" の拡張

10.12.12 モジュールでのグローバル変数とアドレス変数の使用

モジュールはアプリケーションで複数回使用でき (インスタンス化)、また別のアプリケーションで再利用できるため、モジュール内ではグローバル変数またはアドレス変数への直接アクセスがない場合があります。

モジュールで使用されるすべてのグローバル変数とアドレス変数は、モジュール データ コンテナ "UserData" でインターフェース変数として宣言される必要があります。UserData コンテナ内の対応する要素へのグローバル変数またはアドレス変数の割当は、機械で 1 回のみ実行されます。モジュール内の要素は、**UserData.<Element>** を使用してアクセスされます。

以下の例は、アクセスがどのように処理されるかを示したものです。

グローバル変数リスト **MlcVarGlobal** で次の 2 つの変数が宣言されます：グローバル入力およびグローバル出力。

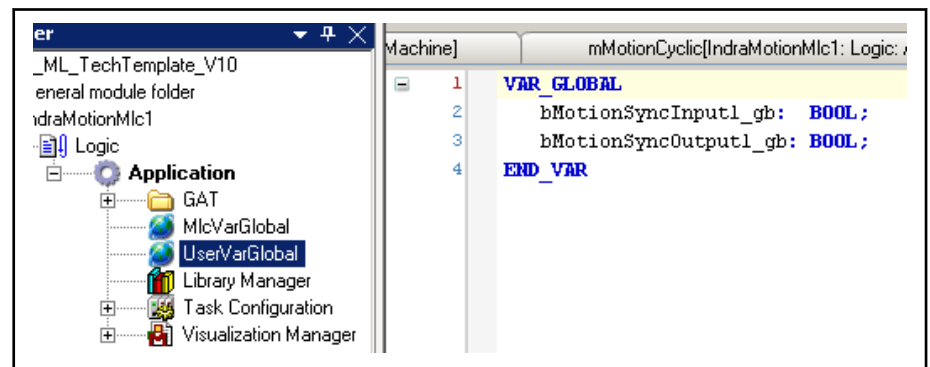


図 10-64: グローバル変数リスト **UserVarGlobal** の 1 つのブール値入力と 1 つのブール値出力変数の宣言

必要な場合は、これらの変数を物理アドレスにマッピングすることも可能です。

この具体的な事例では、グローバル入力変数がコントロールのオンボード入力の最初のビットにマッピングされ、グローバル出力変数がオンボード出力の最初のビットにマッピングされます。

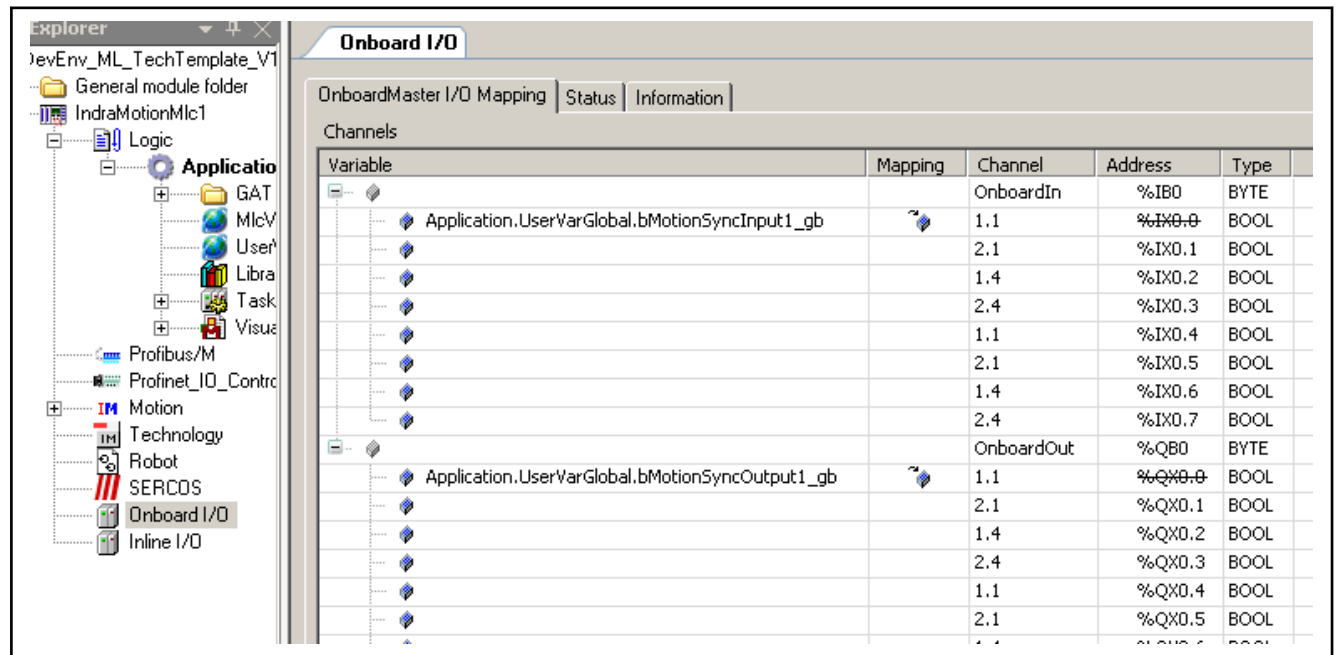


図 10-65: オンボード入力/出力上のブール値グローバル変数のマッピング

グローバル変数が使用されるモジュールで、参照変数が **UserData** データ構造で宣言される必要があります。

集中状態の機械での GAT

参照変数は、使用されるグローバル変数の種類に対応する必要があります。

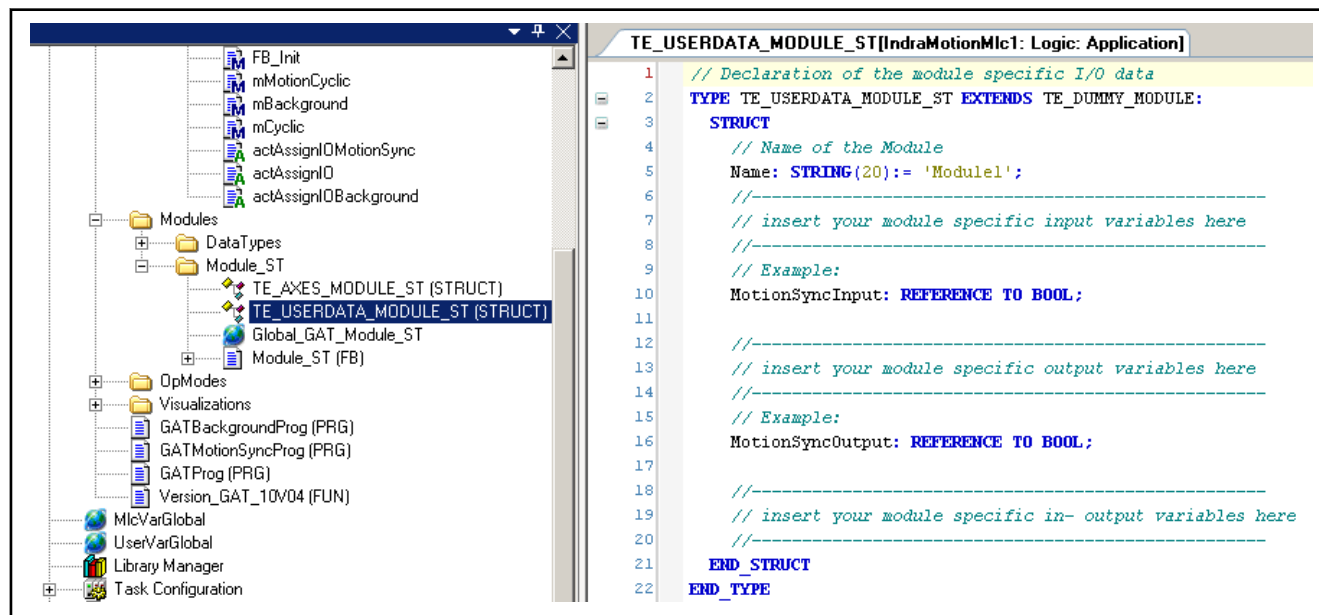


図 10-66: モジュールで使用されるグローバル変数のためのモジュール内の参照変数の宣言

最後に、グローバル変数と、モジュールの対応するインターフェース変数 (参照変数) との間の接続が確立する必要があります。原則として、グローバル変数とアドレス変数はタスクを参照して更新されます。このため、グローバル変数とアドレス変数が対応するモジュール変数とリンクされるファンクションブロック "Machine" には、タスク関連のアクションがあります。

タスク関連アクションは、以下のように相互に接続される必要があります。

- **actAssignIOMotionSync** アクションは **MotionSyncTask** 内
- **actAssignIO** アクションは **PlcTask** 内
- **actAssignIOBackground** アクションは **BackgroundTask** 内

下の図は、当初宣言されたグローバル入力と出力の変数を、対応するモジュール変数にリンクする方法を示したものです。

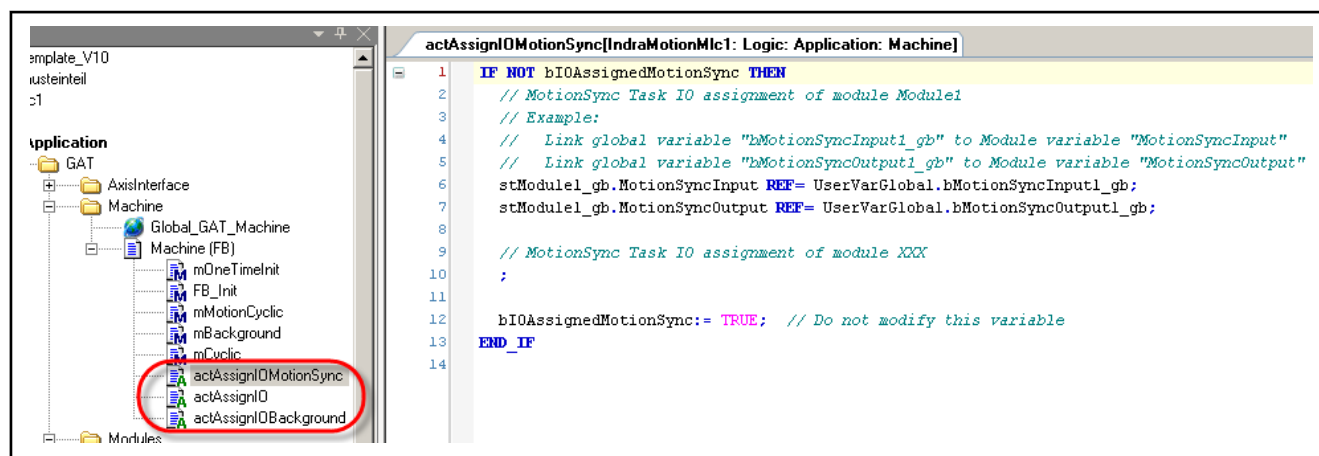


図 10-67: グローバル変数リスト **UserVarGlobal** からのグローバル入力および出力と、モジュール **Module1** のデータ コンテナにある対応する変数との接続

このリンクは **actAssignIOMotionSync** アクションで実行されるため、これらの変数は **MotionSyncTask** を使用して更新されます。

集中状態の機械での GAT

以前の宣言と割当に基づいて、モジュール内のグローバル変数またはアドレス変数へのアクセスは、データ コンテナ **UserData** を使用して実行されます。

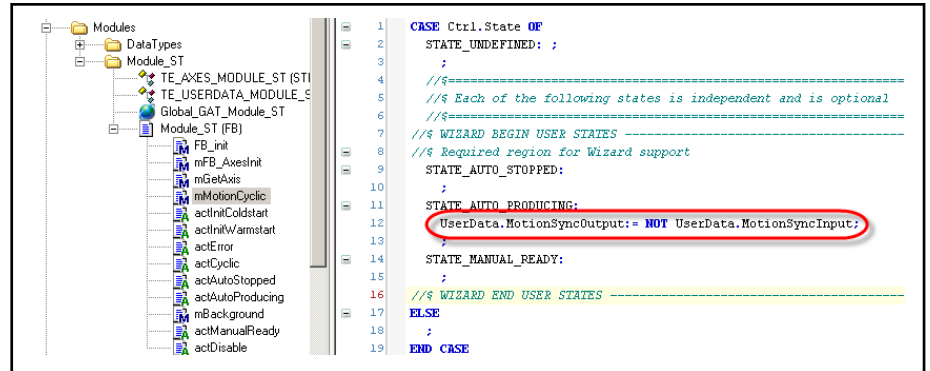


図 10-68: モジュールを参照するデータ コンテナ **UserData** を使用した、モジュール内のグローバル変数またはアドレス変数へのアクセス

10.12.13 MotionTask での処理

MotionTask での処理のために、各モジュール/機械は "mMotionCyclic" を備えています。このメソッドより、すべてのモジュール/機械変数 (一元的な軸入力ポイント Axes など) にアクセス可能です。

重要な点は、ここで制御する軸が MotionTask で更新されることです (34 ページ "タスク管理" 5.5 章 も参照)。

10.12.14 AxisInterface の拡張

GAT では、AxisInterface 構造のほぼすべての拡張が可能です。追加的な下位構造を追加したり、既存の下位構造を拡張したりできます。

例として、以下の構造要素がユーザー拡張として *AxisCtrl* 構造にプログラミングされています。

- SetupMode: AxisCtrl 構造の追加的な下位構造
 - Enable: Setup (セットアップ) モードを有効にする
 - JogPlus: ジョグ +
 - JogMinus: ジョグ -
 - Vel: ジョグ/参照移動速度
 - Accel: ジョグ/参照移動減速/加速
 - Homing: 軸のタイプ/設定によって ModeHoming|SetAbsRef|ModesPosAbs を有効化
- SyncMode: AxisCtrl 構造内の既存の下位構造の拡張
 - AngleOffset: 表のフェーズ オフセット開始 -> この値によりパラメータ A-0-2730 を説明
 - PhaseShift: ギアの前のマスター軸に比例したオフセット (PhaseOffset: ギアの後マスター軸に比例したオフセット)
 - PhaseShiftVel: PhaseShift を調整するための速度
 - PhaseShiftAcc: PhaseShift を調整するための加速
 - SyncVelocity: フェーズ シフト速度 -> この値によりパラメータ A-0-2790 を説明
 - SyncAcceleration: フェーズ シフト加速 -> この速度はパラメータ A-0-2791 に書き込まれる

集中状態の機械での GAT

例として、以下の構造要素がユーザー拡張として *AxisStatus* 構造にプログラミングされています。

- SetupMode: AxisStatus 構造内の追加的な下位構造
 - EnableAck: Setup (セットアップ) モードが有効
 - Homingack: ホーミングが完了
- SyncMode: AxisStatus 構造内の既存の下位構造の拡張
 - AngleOffsetDone: 表のフェーズ オフセット開始の転送 -> この値によりパラメータ A-0-2730 を説明
 - PhasingMasterDone: PhaseShift が完了
 - SyncVelocityDone: 同期速度の転送 -> パラメータ A-0-2790 は転送済み
 - SyncAccelerationDone: 同期加速の転送 -> パラメータ A-0-2791 は転送済み

これらの拡張のコードは、"GAT/AxisInterface" フォルダ内の TE_AxisInterface() ブロックのアクションにあります。関連する構造は、"GAT/AxisInterface/Type/Control" および "GAT/AxisInterface/Type/Status" フォルダにあります。

ユーザーは、ユーザー独自の拡張を追加できます (これを行うには [97 ページ "AxisInterface のユーザー拡張" 6.7 章](#) を参照)。

10.12.15 Robot Control

協調運動のために軸を制御するには、TechInterface に MB_AxisInterfaceType02 FB を使用する必要があります。

このため、GATcentral で以下を調整する必要があります。

- 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ "MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によってこの構造を拡張する必要があります。

構造 TE_AXIS_CONTROL_Type01 の TE_AXIS_CONTROL_Type02 への名前変更:

```
//former type: TYPE TE_AXIS_CONTROL_TYPE01:
TYPE TE_AXIS_CONTROL_TYPE02: // rename for RobotControl
STRUCT
  Admin:      TE_AXIS_ADMINISTRATION; (* Administration of the axis | Verwaltung der Achse *)
  PosMode:    TE_AXIS_POSITIONING; (* operating mode positioning (absolute, rel. and additive)
    Positionierbetrieb (absolut, relativ und additiv) *)
  VelMode:    TE_AXIS_VELOCITY; (* operating mode velocity control | Geschwindigkeitsregelung *)
  SyncMode:   TE_AXIS_SYNCHRONISATION; (* operating mode synchronisation (phase, vel, and cam)
    Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvenscheibe) *)

  (* add your extensions below here, keep the elements above
    Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)

  (* user extensions | Anwender Erweiterungen *)

  // Jog/Homing functionality | Tipp/Einrichtungsfunktionen
  SetupMode:  TE_AXIS_SETUP_MODE;

  CoordModeCtrl: MB_AXIS_COORDINATED; // inserted for RobotControl

END_STRUCT
END_TYPE
```

- "TE_AXIS_STATUS_Type01" 構造の名前を "TE_AXIS_STATUS_TYPE02" に変更します

集中状態の機械での GAT

構造 *TE_AXIS_STATUS_Type01* の *TE_AXIS_STATUS_TYPE02* への名前変更:

```

(***** Outputs of the following axis - from motion control to logic control *****)
// former type: TYPE TE_AXIS_STATUS_TYPE01:
TYPE TE_AXIS_STATUS_TYPE02:      // Rename for RobotControl
STRUCT
(***** Outputs of the following axis - from motion control to logic control *****)

PtrAxisData:  POINTER TO ML_AXISDATA_SM;
(* Pointer to AxisData of this axis | Zeiger auf AxisData dieser Achse
- Type .PtrAxisData^. for the intellisense menu | Eingabe .PtrAxisData^. fuer Auswahlmenue*)

Admin:  TE_AXIS_ADMIN_STATUS; (* status of the axis | Achs-Status *)

Diag:   TE_AXIS_DIAGNOSIS; (* axis diagnosis information, error, warning....
Diagnose Informationen der Achse, Fehler, Warnung.... *)

PosMode: TE_AXIS_POS_STATUS; (* Status operating mode positioning (absolute, rel. and add.)
Statusmeldungen Positionierbetrieb (absolut, relativ und additiv) *)

VelMode: TE_AXIS_VEL_STATUS; (* Status operating mode velocity control
Statusmeldungen Geschwindigkeitsregelung (Freilauf) *)

SyncMode: TE_AXIS_SYNC_STATUS; (* Status operating mode synchronisation (phase, vel, and cam)
Statusmeldungen Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvensch.)*

(* add your extensions below here, keep the elements above
Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
(* user extensions | Anwender Erweiterungen *)
(* Jog/Homing functionality | Tipp/Einrichtfunktionen *)
SetupMode:  TE_AXIS_SETUP_MODE_STATUS;

// AxisData for HMI (without data type "BIT")
AxisData_HMI:  TE_AXIS_DATA_STATUS;

END_STRUCT
END TYPE

```

- 新しい FB "MB_AxisInterfaceType02" により "TE_Axisinterface" ブロックを拡張する必要があります。 "TE_AxisInterface EXTENDS MB_AxisInterfaceType02"

TE_Axisinterface の拡張 :

```

//former: FUNCTION_BLOCK IE_AxisInterface EXTENDS MB_AxisInterfaceType01
FUNCTION_BLOCK IE_AxisInterface EXTENDS MB_AxisInterfaceType02
(
#####
#####

General Header
-----
Shortdescription      : AxisInterface Example

```

- メソッド `mInitExtensions()` : ファンクションブロック `"MB_AxisInterfaceType02"` の追加的な `"CoordModeCtrl"` 入力を初期化する必要があります

CoordModeCtrl の初期化 :

```

SUPER^.mInitExtension(index);

IF NOT bFirstInit THEN
    // User: Assignment of user defined axis interface inputs
    SetupMode      REF= arAxisCtrl_gb[index].SetupMode;
    // inserted for RobotControl
    CoordModeCtrl  REF= arAxisCtrl_gb[index].CoordModeCtrl;
    // User: Assignment of user defined axis interface outputs
    SetupModeAck   REF= arAxisStatus_gb[index].SetupMode;
    AxisData_HMI   REF= arAxisStatus_gb[index].AxisData_HMI;

    // =====
    // Do not touch the following lines
    // =====

    // Set the references for the AxisInterface FB
    AdminCtrlExt   REF= arAxisCtrl_gb[index].Admin;

```

集中状態の機械での GAT

- "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります

構造変数の調整：

```
VAR_GLOBAL
// Axis interface: command structure: define as RETAIN, to keep the values
// Note: on the L40.2 control, the RETAIN memory is slower then normal VAR, so define
//       this structure only as RETAIN when necessary
// arAxisCtrl_gb: ARRAY [1..MAX_AXIS_NUM] OF MB_AXIS_CONTROL;

// Axis interface: command structure
//former: arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE01;
arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE02;
END_VAR

VAR_GLOBAL
//----- Machine status variables -----

// Axis interface: status structure
//former: arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE01;
arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE02;
bRemoteOn_gb : BOOL := TRUE; // TRUE: Internal IL visu is active
// FALSE: PlcProg is active
END_VAR
```



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 はフォルダ "AxisInterface\Type" に含まれています。

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "UserVarGlobal" に含まれています。

キネマティクス インターフェース

軸インターフェースに基づいて、オプションのプラグインとしてキネマティクス インターフェースが利用可能です。キネマティクス インターフェースを使用すると、軸と同じ方法でキネマティクスを操作できます。この目的のために、GAT ウィザードを介してキネマティクス インターフェースをプラグインとして追加します。

キネマティクス インターフェースの資料は、プラグイン設定フォルダにあります。

以下の手順で、GAT ウィザードを介して設定フォルダへのリンクを参照します。

- GAT ウィザードでフォルダ "Plug-ins" を選択する
- ウィザード コントロール パネルに、リンク "Open plug.ins configuration folder" (プラグイン設定フォルダを開く) が表示される

10.12.16 GAT に基づいたユーザー独自のプログラム フレームワークの生成

GAT に基づいて開発されたアプリケーションは、さらに複雑な別のアプリケーションを開発するためのテンプレート (プログラム フレームワーク) としても使用できます。この方法により、ユーザーは、例えばさまざまな機種向けの独自のプログラム フレームワークを提供できます。しかし、このフレームワークは、より高度な開発 (追加的な運転モードやモジュールなど) のために、標準の GAT とは異なります。その後、GAT ウィザードを使用してさらに高度なプログラム フレームワークに取り組むことが可能です。

GAT により開発されたアプリケーションに基づいたユーザー独自のテンプレートを生成するには、以下の手順を使用します。

1. すべてのチェック ボックスを選択済みの状態にして、GAT フォルダ全体をエクスポートします (*.iwx ファイルを生成する)。Export (エクスポート) ダイアログに "Description" (説明) 入力フィールドがあります。ここで入力したテキストは、GAT ウィザードのインポート機能で表示できます。

集中状態の機械での GAT

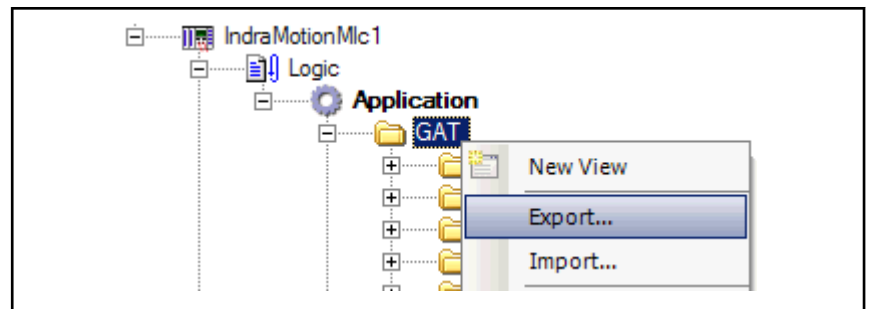


図 10-69: GAT フォルダのエクスポート

2. 手順 1 で生成した *.iwx ファイルには、適合/拡張テンプレートが含まれており、新規プロジェクトを作成するために、以下のとおり GAT ウィザードで使えます。

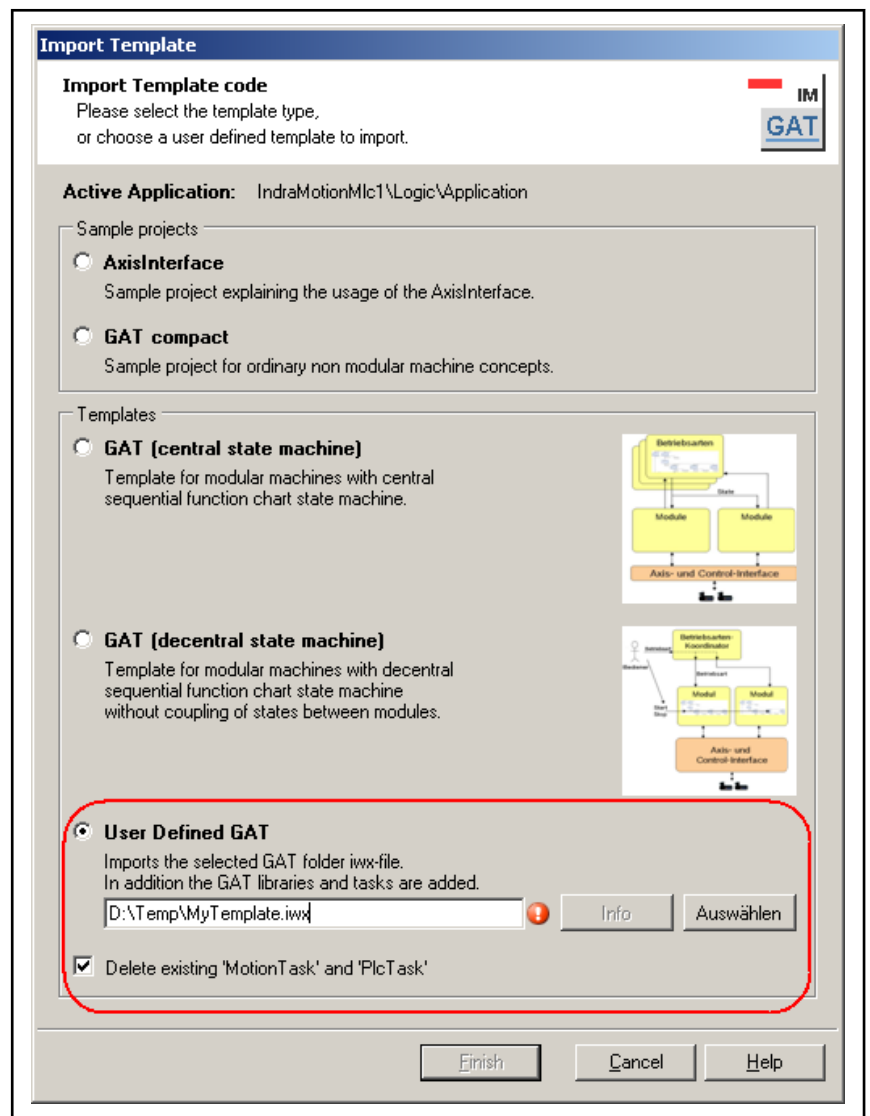


図 10-70: ユーザー定義テンプレートのインポート

10.12.17 GAT に基づいた PackML V3.0 プログラム フレームワーク

インストールには既製のプログラム フレームワークが備えられており、PackML V3.0 で定義されたモードおよび状態のマッピングを行います。

集中状態の機械での GAT

PackML V3.0 プログラム フレームワークは、インポートするファイルとして "GAT-PackML-V3.iwx" を選択し、GAT ウィザードを呼び出した後で、**user-defined GAT** 選択を使用してロードすることができます。"GAT-PackML-V3.iwx" ファイルは、IndraWorks のインストールにより提供され、対応する GAT ウィザード ディレクトリに保存されます。

すべての PackML V3.0 定義は、"PackML_V3" フォルダ内の GAT フォルダにあります。

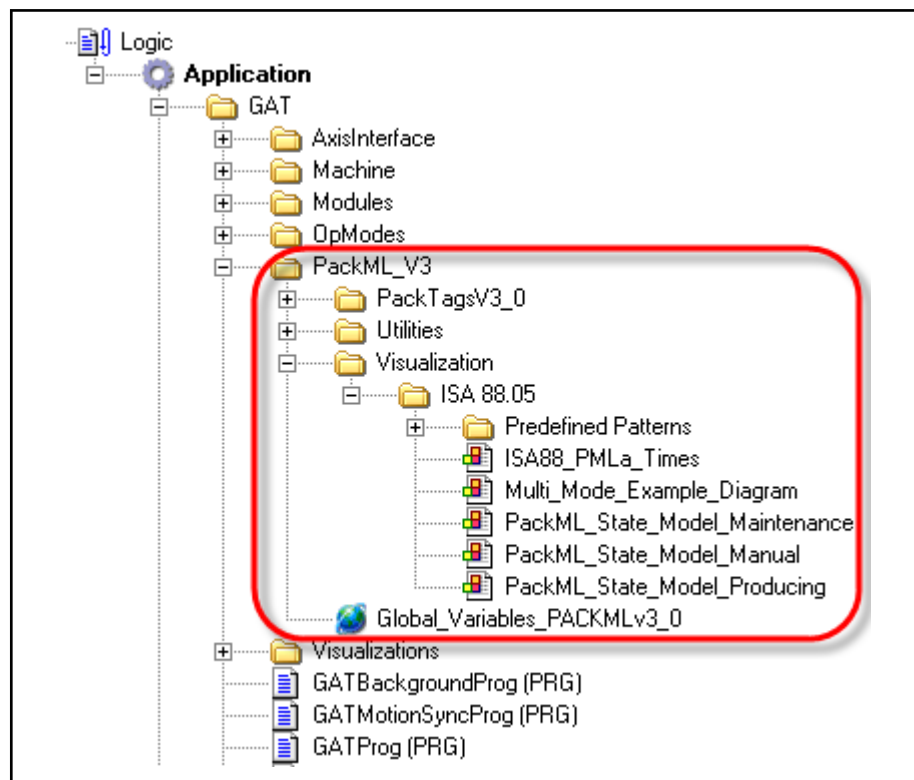


図 10-71: 予め定義済みのプログラム フレームワーク内の PackML V3.0 定義の位置

PackML V3.0 で定義されたモードと状態に加えて、このプログラム テンプレートには、標準テンプレートで定義された表示も含まれています。

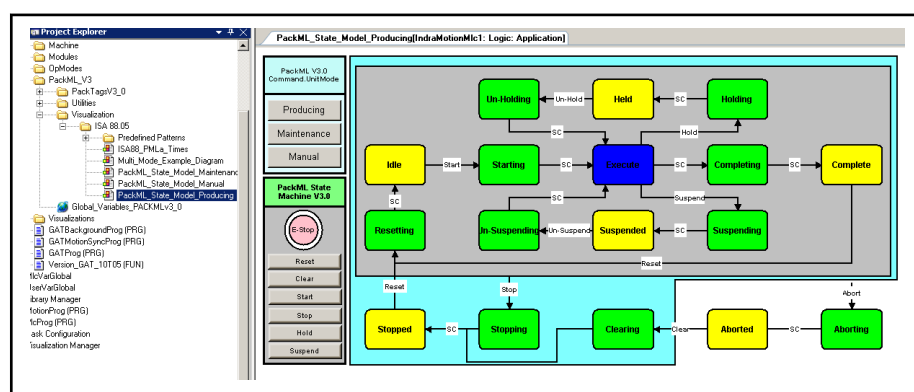


図 10-72: 既製のプログラム フレームワーク内の **Producing** (生産中) モード用の PackML V3.0 表示

このプログラム テンプレートは GAT に基づいているため、GAT に関して本書で言及されているすべての処理規則および可能性も適用されます。



移行 PackML 状態 (<statename>ing) を実装するには、状態に対応する GAT アクションの移行セクションに関連コードを配置する必要があります (287 ページ "モジュールの状態" 10.7.4 章を参照)。

静止 PackML 状態を実装するには、状態に対応する GAT アクションの静止セクションに関連コードを配置する必要があります (287 ページ "モジュールの状態" 10.7.4 章を参照)。

PackML V3.0 に関する詳細は、www.isa.org をご覧ください。

10.12.18 時間依存でアプリケーション固有のエラーリアクション

GAT の標準バージョンでは、運転モードと状態機械が "GATProgTask" (10 ms 周期的タスク) のコンテキストで処理されます。このため、エラーはこの比較的長い時間内で検出され、処理されます。

より迅速なエラーリアクションを必要とするアプリケーションの場合は、運転モードと状態機械を "GATMotionSyncTask" (モーションサイクルと同期した周期的処理) のコンテキストで処理することをお勧めします。エラー検出とエラーリアクションとの間の時間は、1 つのモーションサイクルです。

以下の手順が必要です。

1. タスク設定からの "GATProgTask" の消去
2. "GATMotionSyncTask" での追加的プログラム呼出 "GATProg"

10.12.19 F0 エラーへの GAT エラーリアクションのスイッチオフ

深刻度が "F0" のエラーの場合、コントロール側で制御ファームウェアは、エラーへの対応または軸の解除を自動的には行いません。しかし、GAT はエラーを検出し、現在選択されている運転モードを中断して、エラー処理に入ります。デフォルトの GAT 設定により、このエラー処理では "ModeAH" コマンドを使用してランプで軸が減速されます。

GAT が F0 コントロール エラーに対応しないようにする場合は、"Machine" FB の "mCyclic" メソッドに以下のコードを実装する必要があります。

```
Ctrl.DisableErrorHandling := TRUE; // エラーリアクションをオフにします
MB_IMCInterface(); // 制御診断番号を更新します
IF (TE_GetErrorSeverity(IMCStatus.Diag.Number) > 16#F0) THEN
  // F0 より大きいエラーが表示された場合、エラーリアクションをオンにします
  Ctrl.DisableErrorHandling := FALSE;
END_IF
```

10.12.20 バージョン 10/11 から 12/13 への更新

GAT は、アプリケーション プログラムを変更せずに、バージョン 10/11 から 12/13 に更新できます。これは、機能とインターフェースに互換性があるためです。

GAT ウィザードのバージョン 12/13 には、現在の制御ファームウェア バージョン 12VRS/13VRS が必要です。この理由のため、GAT ウィザードを使用する場合は、ファームウェアがバージョン 10/11 のコントロールの制御ファームウェアを更新する必要があります。

制御ファームウェアおよび制御設定を更新するには、以下の手順が必要です。

1. ファームウェア管理を使用して、バージョン 12/13 へのファームウェア更新を実行する
2. 制御設定でファームウェア バージョン 12VRS/13VRS を選択する
3. Library Manager で、ライブラリ ML_TechInterface および ML_TechTemplate, version 12/13 を選択する (プレースホルダー TECHINTERFACE または TECHTEMPLATE を使用)

10.13.2 機械の最も重要なグローバル変数の概要

グローバル変数リスト "GAT/Machine/Global_GAT_Machine" を参照してください。

変数	説明	アクセス
stMachine1Ctrl_gb	機械の制御 (運転モードの選択、一元的な"エラーのクリア"、初期化の後の運転モード/エラーのリセット、エラー処理のスイッチオフ、初期化設定など)	RD/WR
stMachine1Status_gb	機械のステータス (現在の運転モードと状態、エラーと診断の出力など)	RD
arAppErrMachine1_gb	ユーザー固有の機械エラー フラグ付きの BOOL の列	RD
arMachine1ModuleDiagNo_gb	すべてのモジュールの診断番号の列	RD
bStart_gb	自動運転モード/AUTO_STOPPED to AUTO_PRODUCING からの移行の開始信号	RD/WR
bStop_gb	自動運転モード/AUTO_PRODUCING to AUTO_STOPPED からの移行の停止信号	RD/WR

RD 読み取り
WR 書き込み

10.13.3 モジュールの最も重要なグローバル変数の概要

グローバル変数リスト "GAT/Module/Global_GAT_<Modulename>" を参照してください。

変数	説明	アクセス
st<Modulename>_gb	モジュール固有の I/O	RD/WR
arAppErr<Modulename>_gb	ユーザー固有のモジュール エラー フラグ付きの BOOL の列	RD
stAxes<Modulename>_gb	モジュール軸のあるコンテナ	RD

RD 読み取り
WR 書き込み

10.13.4 軸の最も重要なグローバル変数の概要

グローバル変数リスト "GAT/ AxisInterface/ Global_GAT_AxisInterfaces" を参照してください。

変数	説明	アクセス
arAxisCtrl_gb	AxisInterface コマンド データの列	RD/WR
arAxisStatus_gb	AxisInterface ステータス データの列	RD

RD 読み取り
WR 書き込み

集中状態の機械での GAT

10.13.5 機械のメソッドとアクションの概要

アクション/メソッド	呼出	関連タスク	説明/例
mMotionCyclic	すべての状態で	GATMotionSyncTask	GATMotionSyncTask のコンテキストでの時間依存のモジュール間口ジック (モジュール間の時間依存のモニタリングなど)
mCyclic	すべての状態で	GATProgTask	GATProgTask のコンテキストでの周期的モジュール間口ジック
mBackground	すべての状態で	GATBackgroundProgTask	GATBackgroundProgTask のコンテキストでの時間依存ではないモジュール間のバックグラウンド アクティビティ (HMI データ準備、統計など)
mOneTimeInit	GAT が初期化される際に 1 サイクル。"コールド スタート" 前の実行	GATProgTask	モジュールの有効化/無効化、"SkipColdStart" の設定の初期化、グローバル IndraWorks 軸へのモジュール軸の割当
mExitOpMode	運転モードの終了時に 1 サイクル	GATProgTask	メソッドは任意: 運転モード終了時のアクティビティ (変数のリセットなど)
mEntryOpMode	運転モードを有効にする際に 1 サイクル	GATProgTask	メソッドは任意: 運転モードを有効にする際のアクティビティ (変数の初期化など)
mOpModeChangeAllowed	運転モードを切り替える前	GATProgTask	メソッドは任意: メソッドは運転モードの切替の前に GAT により呼び出される。戻り値 = FALSE である限り、運転モードの切替はブロックされる
mIsCriticalError	エラーが発生した場合	GATProgTask	エラーが発生し、機械のエラーの排除が可能であれば呼び出される
actAssignIOMotionSync	PLC プログラムの起動/リセット後に 1 回	GATMotionSyncTask	"GATMotionSyncTask" での I/O 割当
actAssignIO	PLC プログラムの起動/リセット後に 1 回	GATProgTask	"GATProgTask" での I/O 割当
actAssignIOBackground	PLC プログラムの起動/リセット後に 1 回	GATBackgroundProgTask	"GATBackgroundProgTask" での I/O 割当
FB_init	PLC プログラムの起動/リセット前に 1 回	特定のタスク割当なし	内部システムの使用は、変更できません

10.13.6 機械のメソッドとアクションの概要

アクション/メソッド	呼出	関連タスク	説明/例
actInitColdstart	INIT_COLDSTART 状態でのみ	GATProgTask	初期化/パラメータ化モード (P2) でのパラメータの書き込み
actInitWarmstart	INIT_WARMSTART 状態でのみ	GATProgTask	初期化/パラメータの書き込み、軸の出力有効化

集中状態の機械での GAT

アクション/メソッド	呼出	関連タスク	説明/例
actAutoStopped	AUTO_STOPPED 状態でのみ	GATProgTask	軸の動作を停止
actAutoProducing	AUTO_PRODUCING 状態でのみ	GATProgTask	軸の動作を開始
actManualReady	MANUAL_READY 状態でのみ	GATProgTask	軸のジョグ中のモニタリング
actError	ERROR_F0.. ERROR_F9 状態でのみ	GATProgTask	追加的なユーザー固有のエラーリアクション (例: 軸の同期減速)
mMotionCyclic	すべての状態で	GATMotionSyncTask	GATMotionSyncTask のコンテキストでの周期的呼出。 時間依存のプロセス コントローラ ファンクションブロック (RegisterController など) の呼出
mCyclic	すべての状態で	GATProgTask	モジュール固有の周期的ロジック 用 GATProgTask のコンテキストでの周期的呼出
mBackground	すべての状態で	GATBackgroundProgTask	時間依存ではないバックグラウンド アクティビティ用 GATBackgroundProgTask (HMI データ準備、統計など) のコンテキストでの周期的呼出
mlsMyAxis	自動呼出なし	特定のタスク割当なし	軸がモジュールに属するか否か確認
mlsCriticalError	エラーが発生した場合	GATProgTask	エラーが発生し、このモジュールについてエラーの排除が可能であれば呼び出される
mForceOpMode	自動呼出なし	特定のタスク割当なし	- 機械により指定されたものとは別の運転モードにモジュールを設定。 - モジュールの運転モードを機械の運転モードにリセットする
mExitOpMode	運転モードの終了時に 1 サイクル	GATProgTask	メソッドは任意: 運転モード終了時のアクティビティ (変数のリセットなど)
mEntryOpMode	運転モードを有効にする際に 1 サイクル	GATProgTask	メソッドは任意: 運転モードを有効にする際のアクティビティ (変数の初期化など)
actDisable	モジュールを無効にする間、無効化が完了するまでのすべての状態で	GATProgTask	モジュールの無効化中の変数またはモジュールのリセット
mGetAxis	自動呼出なし	特定のタスク割当なし	内部システム使用、変更はできません。ポインターをモジュール軸に提供
mFB_AxesInit	軸の初期化中	GATProgTask	内部システムの使用は、変更できません
FB_init	PLC プログラムの起動/リセット前に 1 回	特定のタスク割当なし	内部システムの使用は、変更できません

集中状態の機械での GAT

10.13.7 モジュール内の軸へのアクセス



必ず **Axes.<axisname>** 経由でアクセスします。

例：

```
// 速度制御モードでのコマンド軸
Axes.MyAxis1.Ctrl.Admin._OpMode.en := ModeVel;

/ If axis has reached the velocity
IF (Axes.MyAxis1.Status.PtrAxisData^.Axis_InVelocity = TRUE) THEN;
```

10.13.8 モジュール内の I/O データへのアクセス



必ず **UserData.<Variable>** 経由でアクセスします。グローバル変数にアクセスしないでください！

例：

```
IF (UserData.bInput1=TRUE) THEN // "bInput1" が設定されている場合…
  UserData.bOutput1:=TRUE; // 出力を "bOutput1" に設定します
END_IF
```

11 分散状態の機械での GAT

11.1 一般情報

分散運転モード SFC を備えた Generic Application Template (GAT、汎用アプリケーション テンプレート) は、よりシンプルで拡張性が高く、すぐに利用可能な IndraWorks プログラム フレームワークで、IEC61131-3 に基づいています。機械とそのドライブを制御するための関連コンポーネントがすべて含まれています。

IndraWorks に組み込まれている GAT ウィザード を用いると、関連するすべてのコード ポジションにユーザーが直ちに移動できるため、トレーニングが迅速になります。

GAT には、運転モード管理、エラー処理や診断システムから、製品と機械のデータ処理、AxisInterface を使用した簡素な動作コマンドまで、機械のすべての重要な基本機能がすでに含まれています。GAT では表示も準備されているため、基本バージョンでも直感的に使用できます。

定義済みのインターフェースとウィザードは、調整や追加の作業を簡単に行う上で役立ちます。



分散運転モード SFC を備えた GAT は、IndraMotion MLC コントロールを必要とします。

11.2 概要

11.2.1 基本概念

GAT の使用は、アプリケーションはさまざまな運転モードで 1 つ以上の機能"ユニット"を協調させる機械で実行されるという考え方にに基づいています。最も単純なケースでは、選択された運転モードがユニットに転送されます。しかし、各ユニットに異なる運転モードを指定することも可能です。

GAT は以下の要素により構成されています。

分散状態の機械での GAT

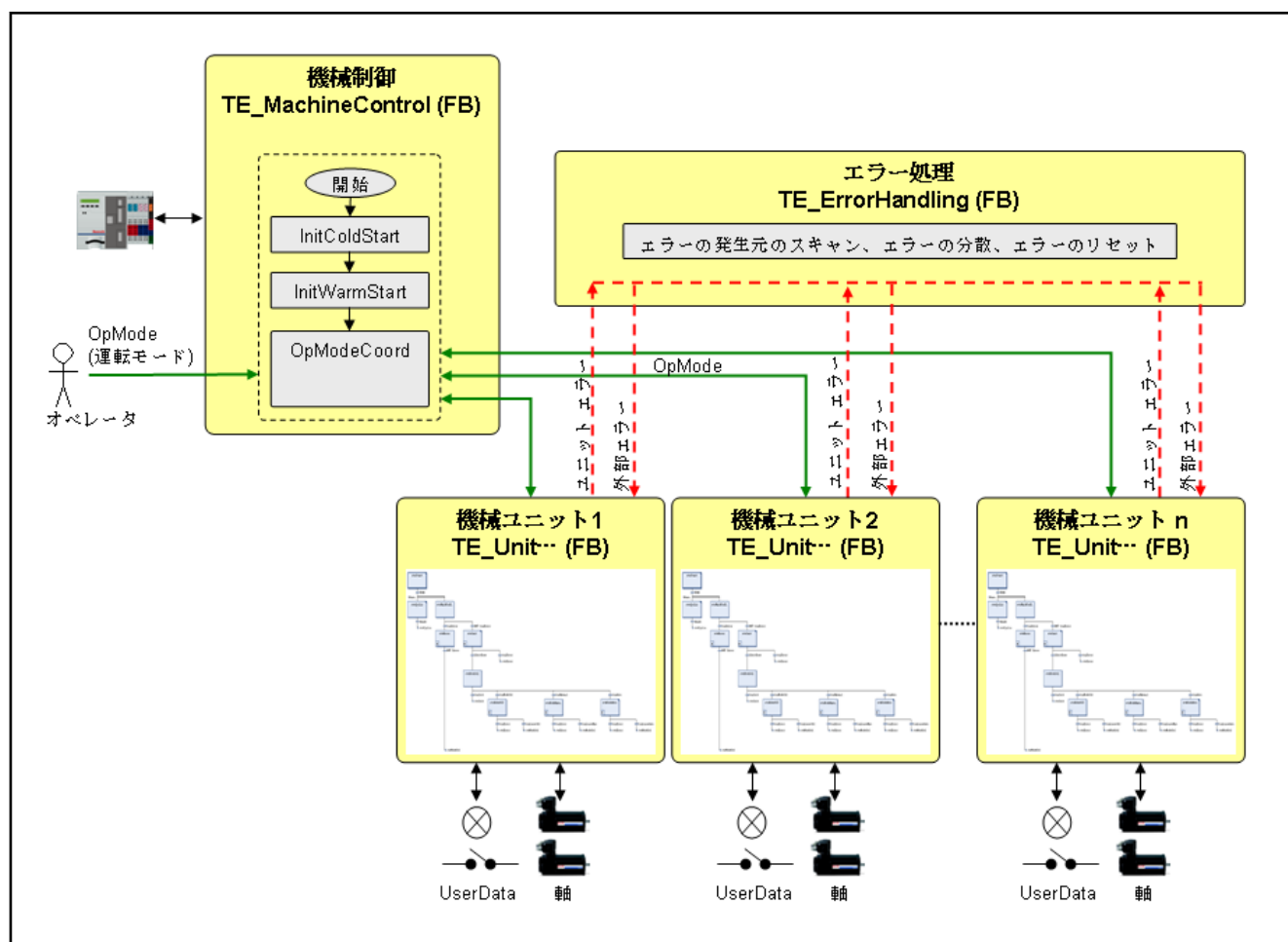


図 11-1: 分散状態の機械での GAT のソフトウェアアーキテクチャ

マシン ハンドラー

マシン ハンドラー (FB TE_MachineControl) は、オペレータが選択した運転モードを個別のユニットに転送します。システムの初期化中に、コントロールのフェーズ切替を命令し、これを行う中で、すべてのユニットの各初期化手順を進めます。初期化が完了した後で、最も単純なケースでは、オペレータが命令した運転モードがすべてのユニットに転送されます。

マシン ハンドラーは以下の部分に分けられます。

- 初期化のシーケンシング (アクション actOPModeSFC)
- 表示および機械と製品データ処理とのフィールド バス接続のためのデータ ハンドラー (メソッド mDataHandler)
- プライマリ運転モードの決定 (メソッド mOpModeSet)
- 各ユニットへの運転モードの配信 (メソッド mOpModeDistribution)
- 恒久的な周期的処理 (メソッド mCyclic)

エラー ハンドラー

エラー ハンドラー (FB TE_ErrorHandling) は、すべてのエラーが収集され、転送される中心的なポジションです。また、エラー処理によって一元的なエラーのクリアが実行されます。

エラー ハンドラーは以下の部分に分けられます。

- エラー検出セクションでは、検出されたエラーの発生元 (コントロール、ユニット、フィールド バス、インライン エラーなど) がすべてスキャンされます (メソッド mScanError)。

分散状態の機械での GAT

- エラー配信セクションでは、エラー情報がすべてのユニットに配信されます。また、エラー情報を選択的に特定のユニットに転送することもできます (メソッド `mDistributeError`)。
- エラー確認セクションでは、エラーをクリアするための命令がエラーのすべての下位発生元に転送されます (メソッド `mClearError`)。

ユニット

機械ユニット (以下、ユニットという) は、ソフトウェアをモジュール化するためにカプセル化された機械部分です (FB `TE_Unit...`)。すべてのユニットは、ユーザーフレンドリーな交換とメンテナンスを実現するために、統一されたインターフェースを備えています。

各ユニットは、関連付けられているコンポーネント (軸など) のエラー情報をクエリし、その情報をエラー ハンドラーに転送します。また、ユニットでエラーリアクションが実行されます。

11.2.2 使用の要件

前の章である程度提示しましたが、GAT を使用するための前提条件をここで再度まとめます。

GAT によりモーションアプリケーションを開発する前に、ユーザーはアプリケーションに関する以下の疑問について明らかにする必要があります。

- どの運転モードを実現するのか？
- 機械はどの機能ユニットで構成されるか？ 各機能にどの軸および I/O が関連付けられるか？

この情報を用いて、ウィザードは実際のモーションアプリケーションのためのコード フレームワークを非常に短時間で生成できます。

以下のセクションでは、GAT の各パーツ、これらのパーツが相互にどのように動作するか、および実際のアプリケーションを実現するためにユーザーはどこで追加作業を行う必要があるかについて説明します。

11.3 最初の手順 – 分散状態の機械での GAT

11.3.1 単純な動作アプリケーションの作成

準備

以下の説明に従って、GAT ウィザードを使用して 2 つのユニットと 2 本の軸を備えたモーションロジック アプリケーションを数分間で作成します。

これを行うために、IndraMotion MLC コントロールが必要ですが、実在のドライブは不要です。基盤は、分散運転モード SFC を備えた GAT です。

トピック

- 機械は 2 つの機能ユニットで構成されます。
- "UnitMaster" という名前のユニットで、ユニット軸 "Master_Axis" は "Automatic" (自動) 運転モードで毎分 100 回転で連続的に動作します。
- "Unit1" という名前のユニットで、ユニット軸 "Slave_Axis" は "Automatic" (自動) 運転モードで、連続的に動作する軸 "Master_Axis" とフェーズ同期して動作します。

下の図は、タスクを図式によって示したものです。

分散状態の機械での GAT

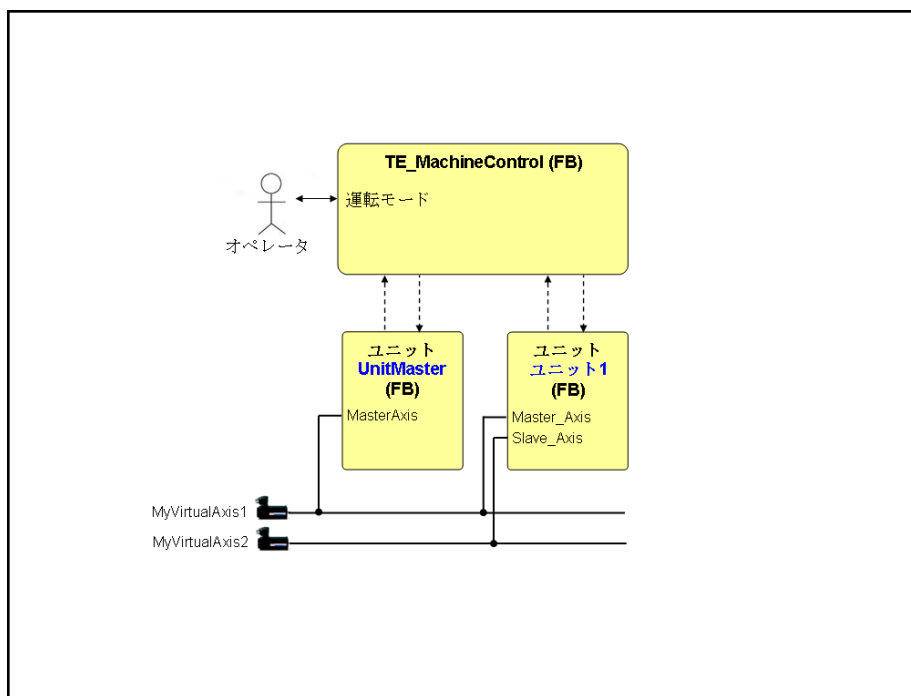


図 11-2: タスクの割当の図解

操作順序

- IndraWorks を開き、新規プロジェクトを作成します。
- MLC コントロールを、ライブラリからプロジェクト ツリーに追加します。
- GAT を MLC コントロールに追加します。これを行うには、コントロールの "GAT" コンテキスト メニューを使用するか、または IndraWorks ツールバーにある  をクリックします。
- 下のダイアログで "GAT (decentral state machine)" の選択肢を選び、"Finish" (完了) をクリックします。

分散状態の機械での GAT

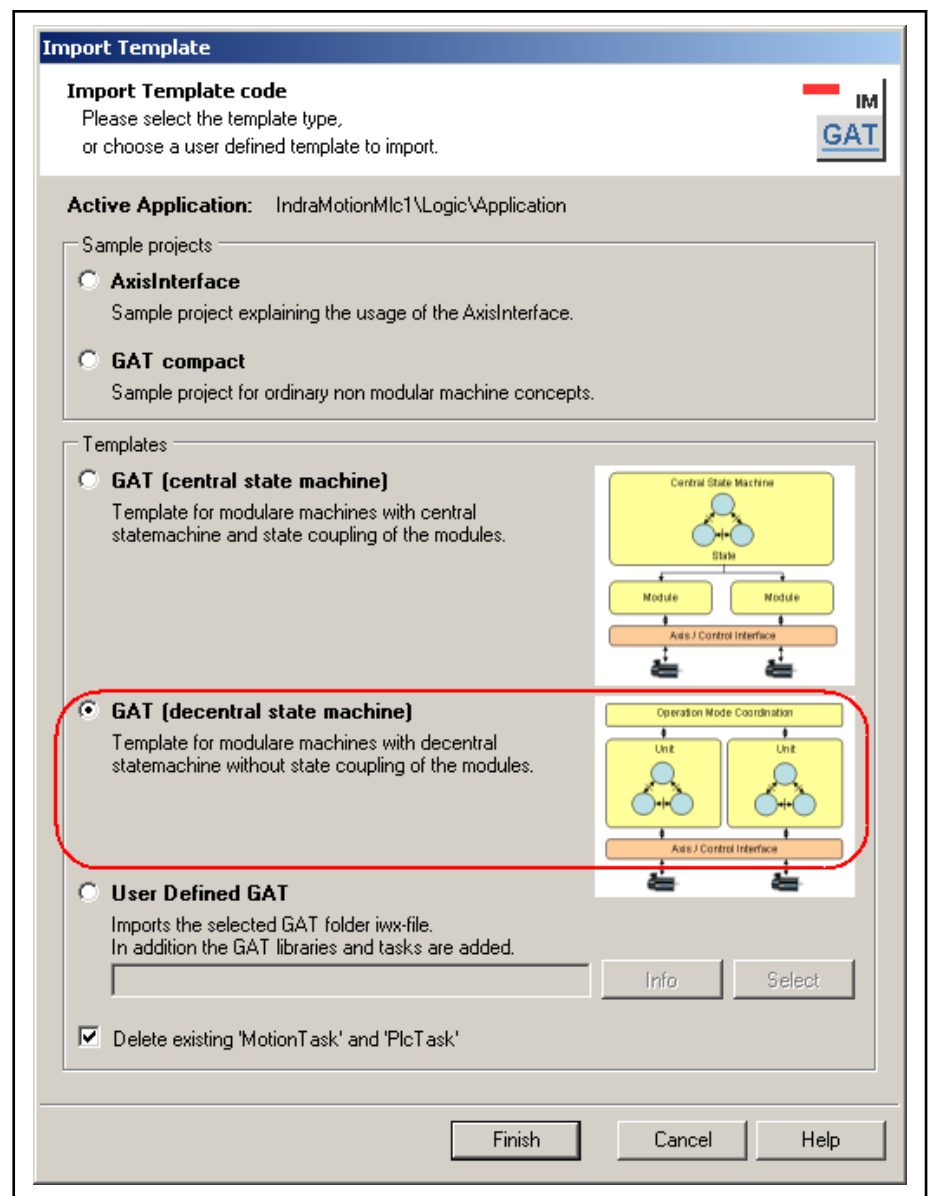


図 11-3: 分散状態の機械での GAT のインポート

"分散状態の機械での GAT" を選択することにより、実行可能で予め定義済みのプロジェクトがインポートされます。このプロジェクトにはすでに 2 本の仮想軸が含まれています。

"AUTO" (自動) 運転モードは事前設定に含まれているため、運転モードの定義に関しては、タスクを実現するために他の設定を選択する必要はありません。

UnitMaster の作成

まず、ユニット "Unit_Master" を作成します。このユニットには、"Automatic" (自動) モードで 100 rpm で動作する軸 ("MasterAxis") が含まれています。

以下のとおり手順を続行します。

- GAT ウィザードを介して新しいユニットを追加します。これを行うには、GAT ウィザードで "Units" ノードを選択します。次に、"Add unit" (ユニットを追加) ボタンをクリックします。

分散状態の機械での GAT

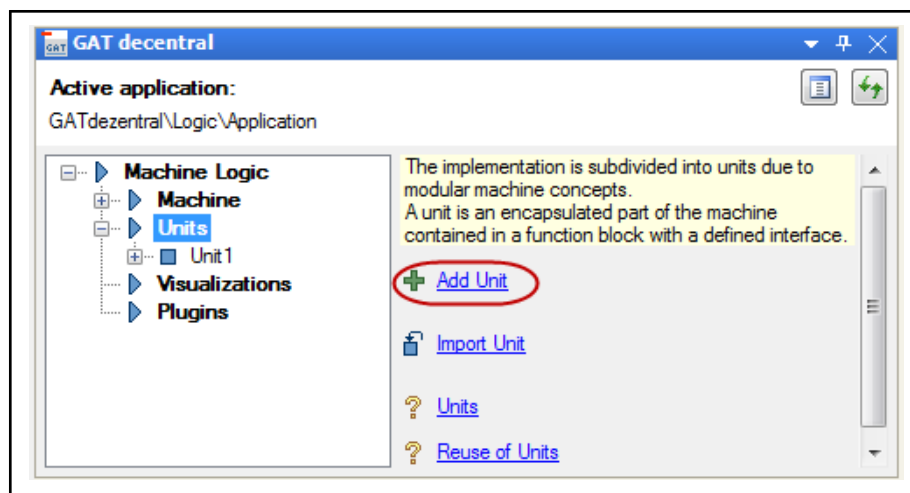


図 11-4: ユニットの追加

- ダイアログで名前 "Unit_Master" およびその他の設定を以下のとおり入力し、"Apply" (適用) ボタンをクリックします。

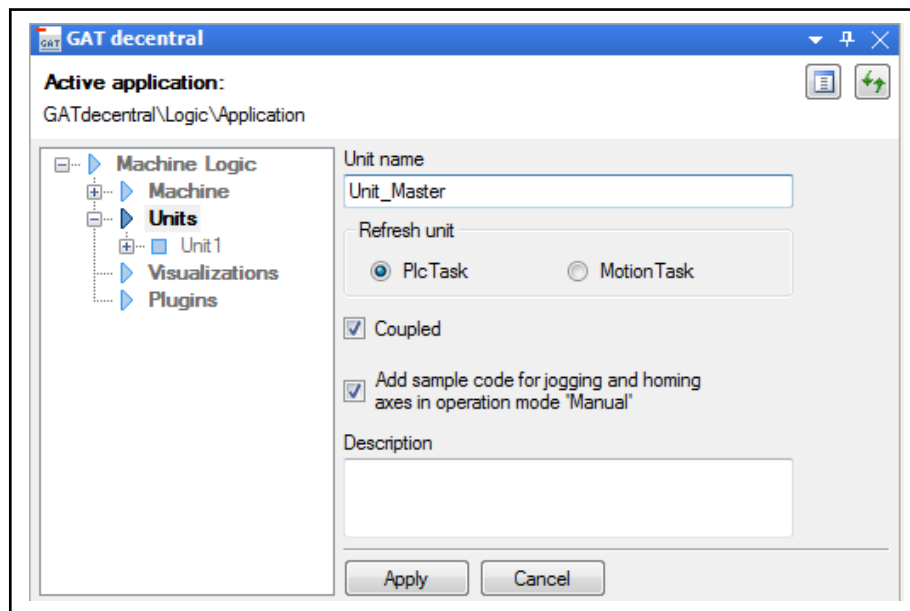


図 11-5: ユニット名の指定

- GAT ウィザードを介して "Unit_Master" ユニットに軸 "MasterAxis" を以下のとおり追加します。

分散状態の機械での GAT

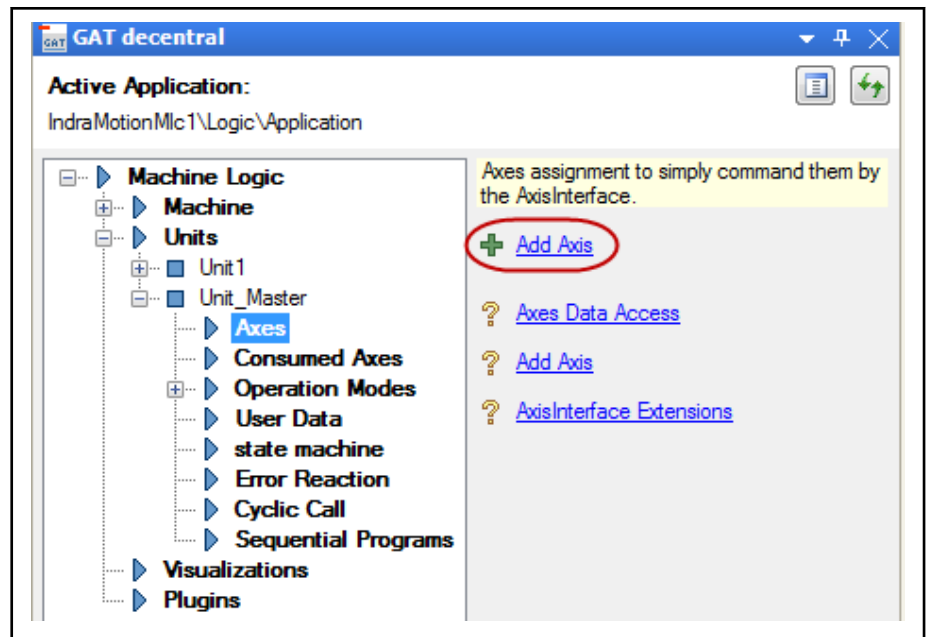


図 11-6: GAT ウィザードによるユニット軸の追加

- ダイアログで軸の名前 "MasterAxis" を入力し、グローバルに設定された軸 "MyVirtualAxis1" にその軸を割り当てます。

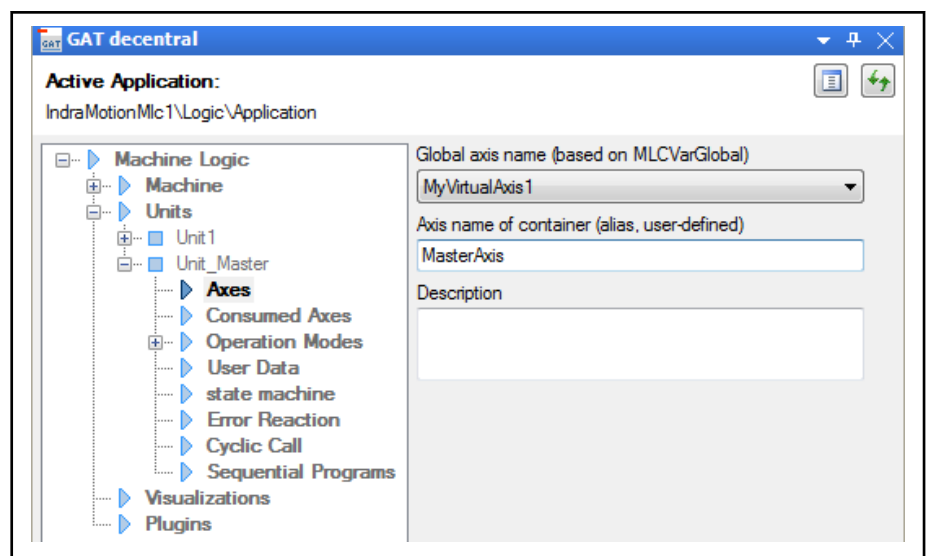


図 11-7: グローバル設定された軸へのユニット軸の割当

- プログラム コードを以下のとおり追加します。"Auto" (自動) フォルダを選択し、リンク "Open implementation" を経由してコード ポジションを開きます。

分散状態の機械での GAT

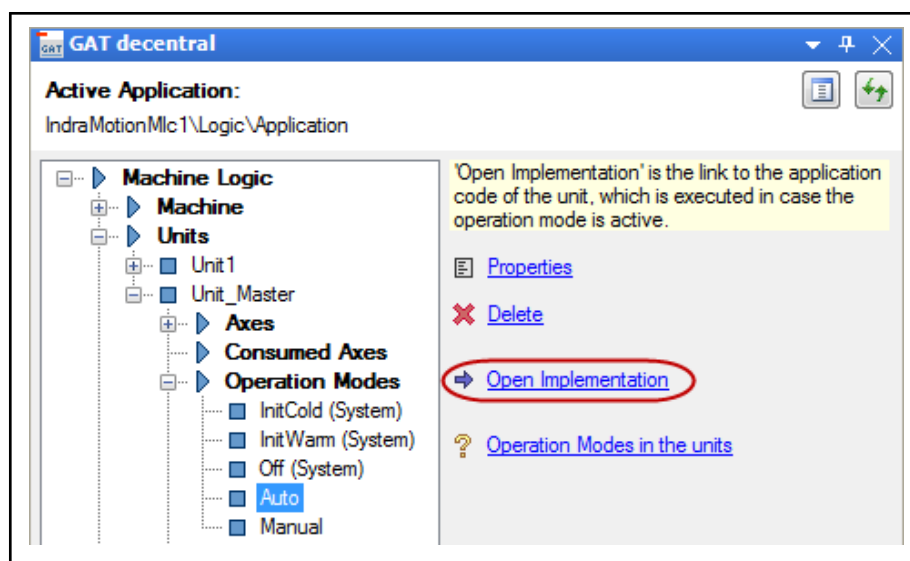


図 11-8: GAT ウィザードを使用した "Automatic" (自動) 運転モードのコードポジションの検索

- 以下のプログラムコードを挿入します。まずコマンド速度 100 rpm が決定され、次に運転モード "velocity control" (ModeVel) が設定されます。



快適な軸コマンドのために、ダイアログベースの軸コマンドも利用可能です。このトピックに関する詳細は、67 ページ "ダイアログベースの軸コマンド" 6.4.1 章を参照してください。

コマンド速度と運転モードの設定

```
//-----
// insert your code here
//-----
Axes.MasterAxis.Ctrl.VelMode.Velocity := 100;
Axes.MasterAxis.Ctrl.Admin._OpMode.en := ModeVel;
```

軸コンテナ "Axes" に関する注意

ユニット内の軸の参照は、常に "Axes" を使用して行われます。すべてのユニットで、軸コンテナの名前は "Axes" です。軸コンテナに含まれている軸をすべて表示するには、"." (ピリオド) を "Axes" の後に入力します (下の図を参照)。ユニットのすべての I/O データは、"UserData" を使用して同じ方法でアクセスできます。

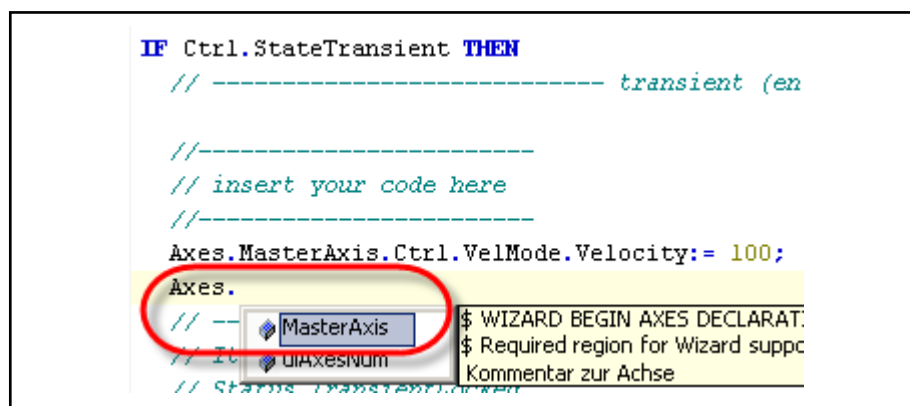


図 11-9: ユニット内の軸のコマンドと Intellisense

さらに、軸名の後にピリオドの演算子を入力することにより、その軸に関するデータを含む要素が表示されて選択できます (下の図を参照)。次のレベルに移動するには、前のレベルの名前の後にピリオドを入力します。

分散状態の機械での GAT

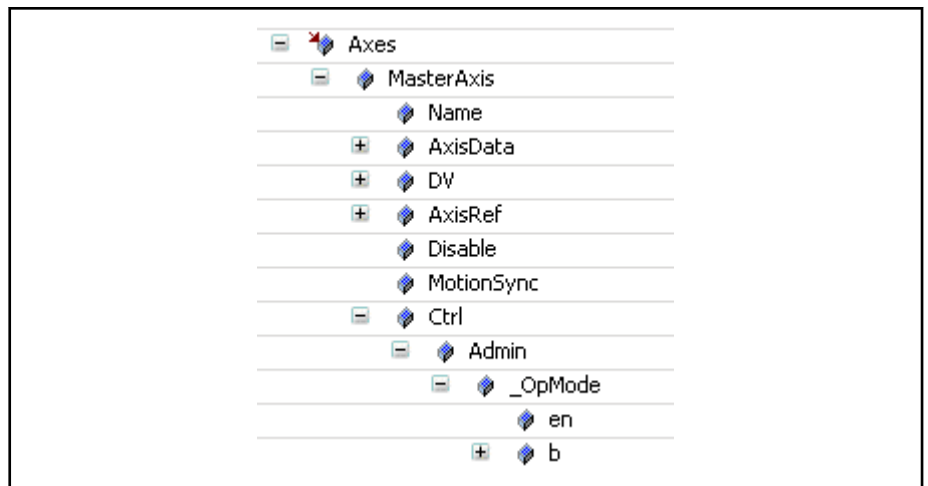


図 11-10: 利用可能なユニット軸の構造要素

Unit1 の調整

GAT 初期インポートにより、Unit1 はすでに利用可能です。このユニットの調整は以下の手順で行います。"Automatic" (自動) 運転モードで、"Slave" 軸は "Master" 軸にフェーズ同期して動作します。Automatic (自動) 運転モードでは、マスター軸は "Unit_Master" ユニットですすでに始動しています。

- GAT ウィザードを介して "MasterAxis" を消費軸として以下のとおり追加します。

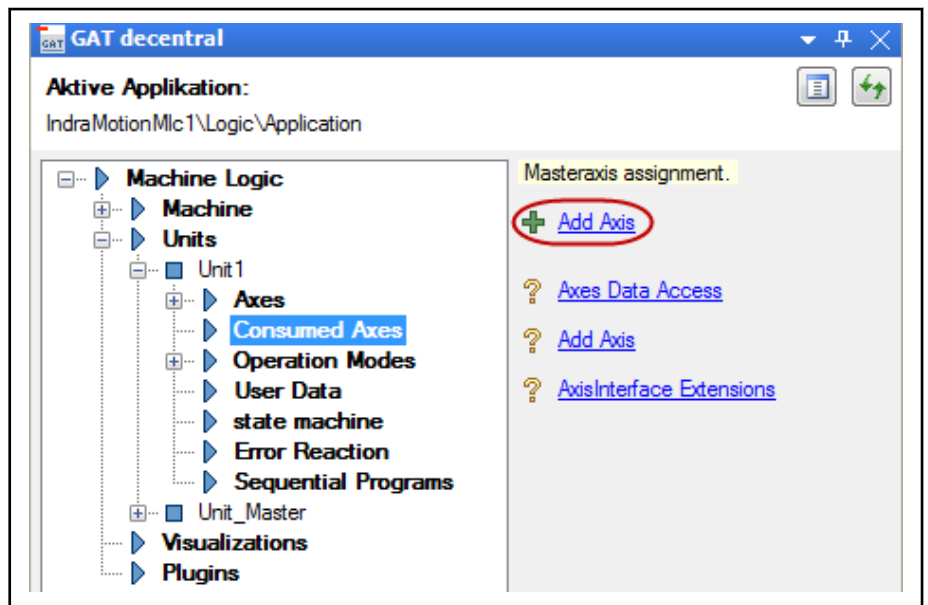


図 11-11: 消費軸としての "MasterAxis" の追加

分散状態の機械での GAT

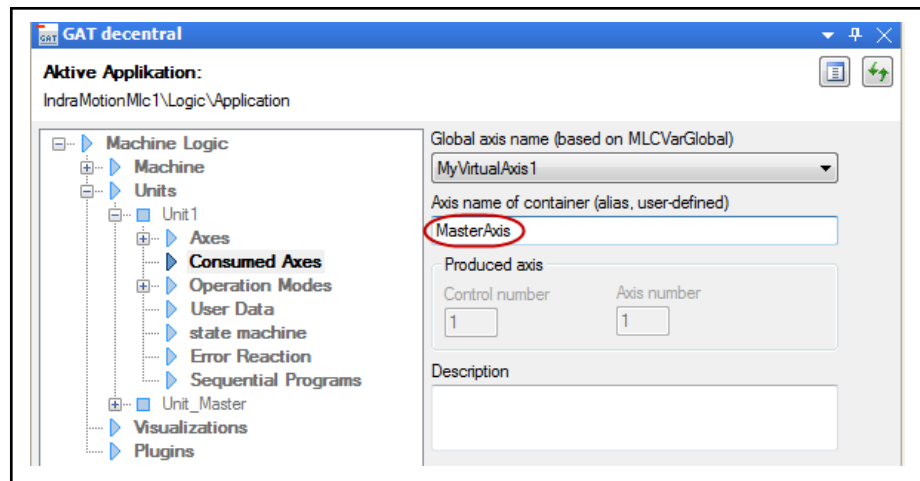


図 11-12: 消費軸としての "MasterAxis" の追加



"MasterAxis" は、Unit1 に"消費軸"として追加されます。これは、この軸がこのユニットでマスター軸としてのみ使用されるためです。

- GAT ウィザードを介して軸 "MyAxis1" を削除します。

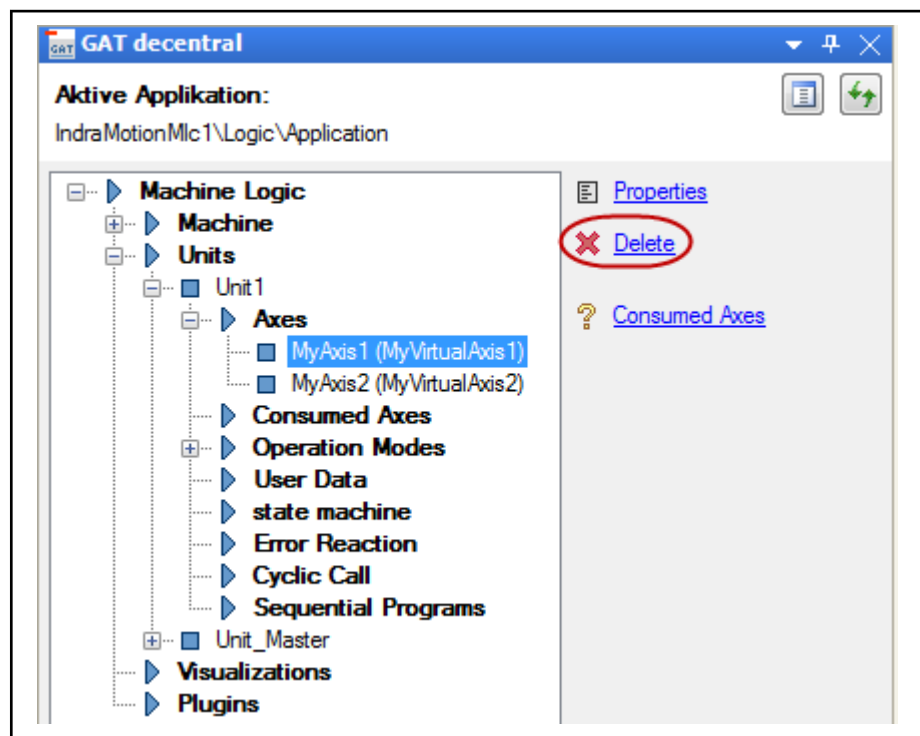


図 11-13: 不要な軸 "MyAxis1" の削除

- 軸 "MyAxis2" の名前を "SlaveAxis" に変更します。

分散状態の機械での GAT

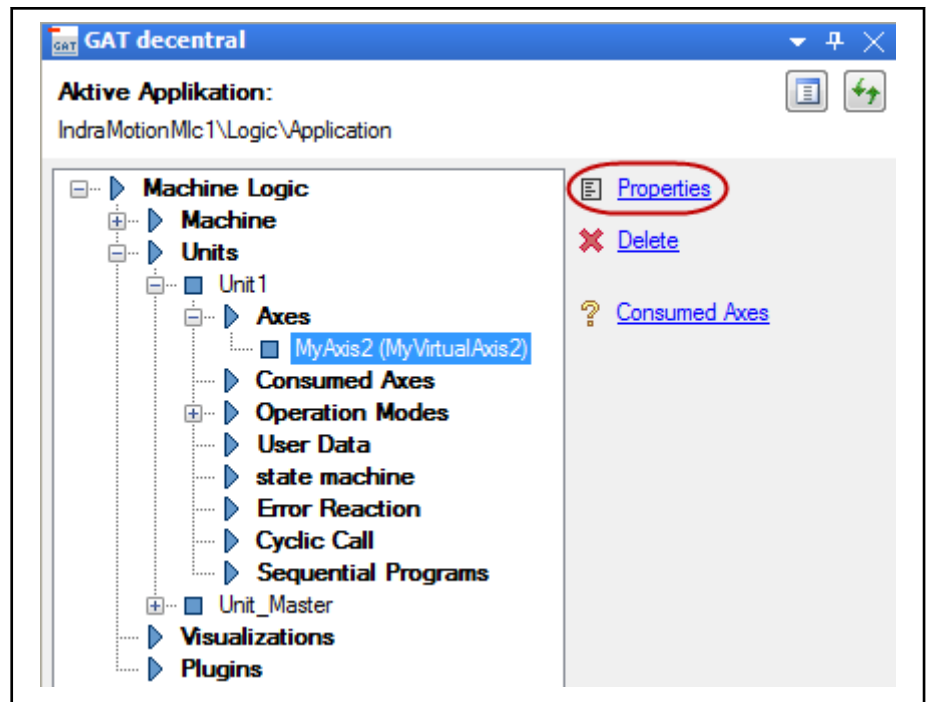


図 11-14: 軸 "MyAxis2" の "Slave_Axis" への名前変更

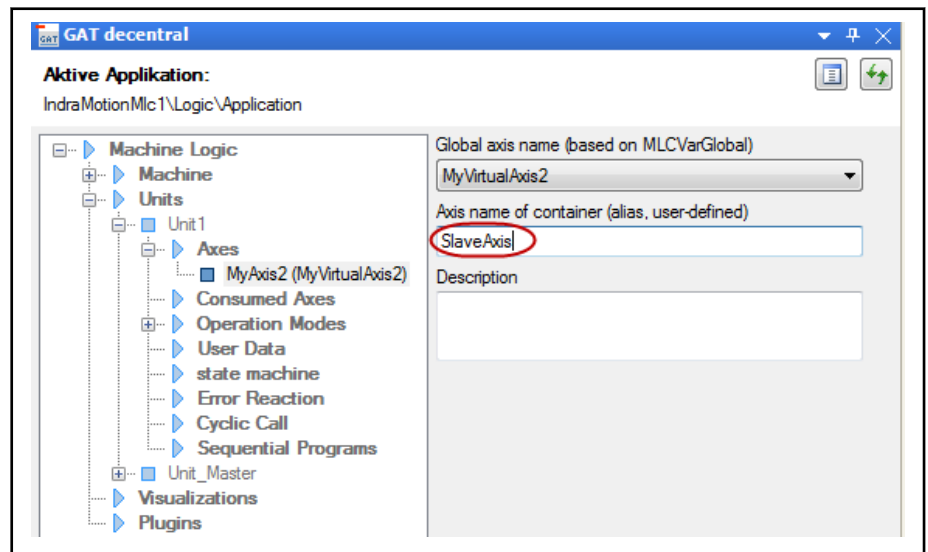


図 11-15: 軸 "MyAxis2" の "Slave_Axis" への名前変更

- Automatic (自動) モードで、初期インポートから既存のコード例を削除し、以下のプログラム コードに置き換えます。これにより、"MasterAxis" とフェーズ同期するコマンドが "SlaveAxis" に発せられます。

分散状態の機械での GAT

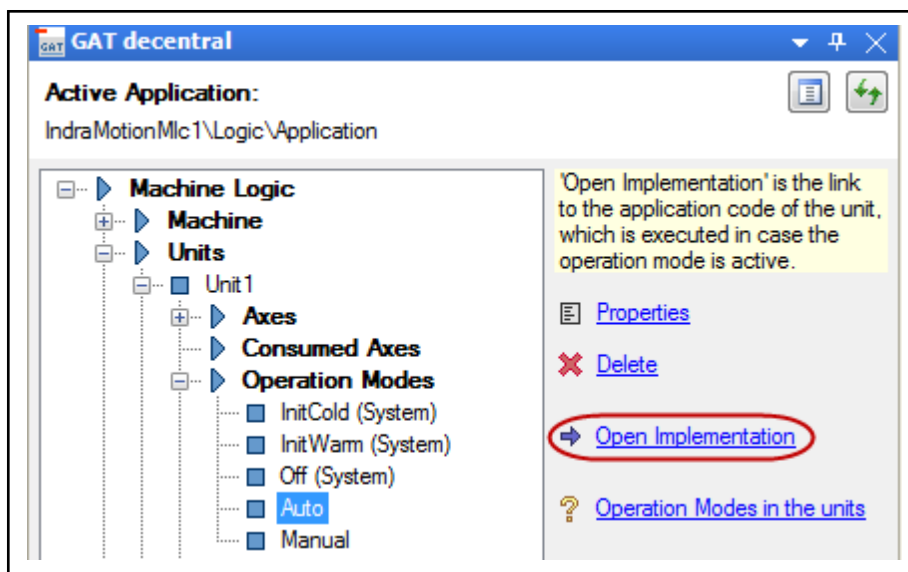


図 11-16: Unit1 の "Automatic" (自動) 運転モードを開く

MasterAxis と同期した SlaveAxis

```
//-----
// insert your code here
//-----
```

```
Axes.SlaveAxis.Ctrl.SyncMode.Master := ConsumedAxes.MasterAxis;
Axes.SlaveAxis.Ctrl.Admin._OpMode.en := ModeSyncPhase;
```

試運転表示の調整

新しい軸の名前は、試運転表示で以下のとおり調整する必要があります。

- GAT ウィザードを介して試運転表示を開きます。

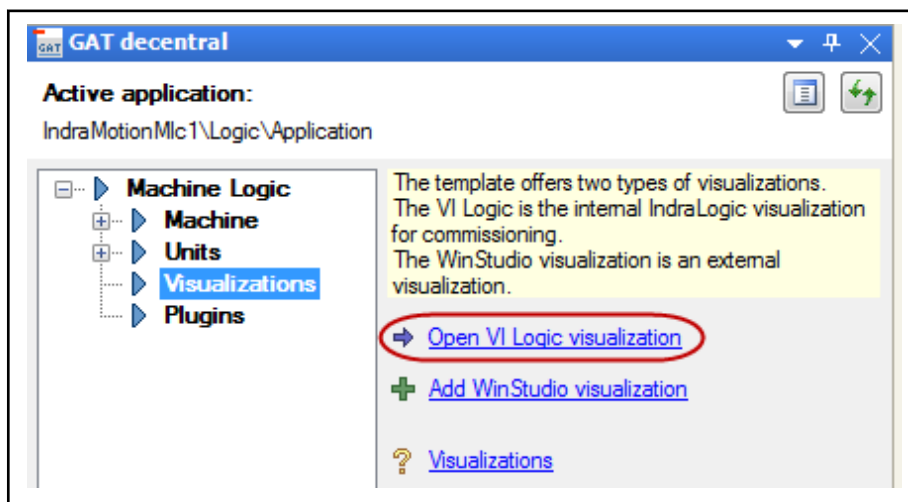


図 11-17: GAT ウィザードを介して試運転表示を開く

- 下の図で強調表示されている表示オブジェクトをクリックし、入力フィールド "Referenced visualizations" (参照される表示) でユニットおよび軸の名前を以下のとおり調整します。すると、"Unit_Master" ユニットの "MasterAxis" に軸表示が適用されます。

分散状態の機械での GAT

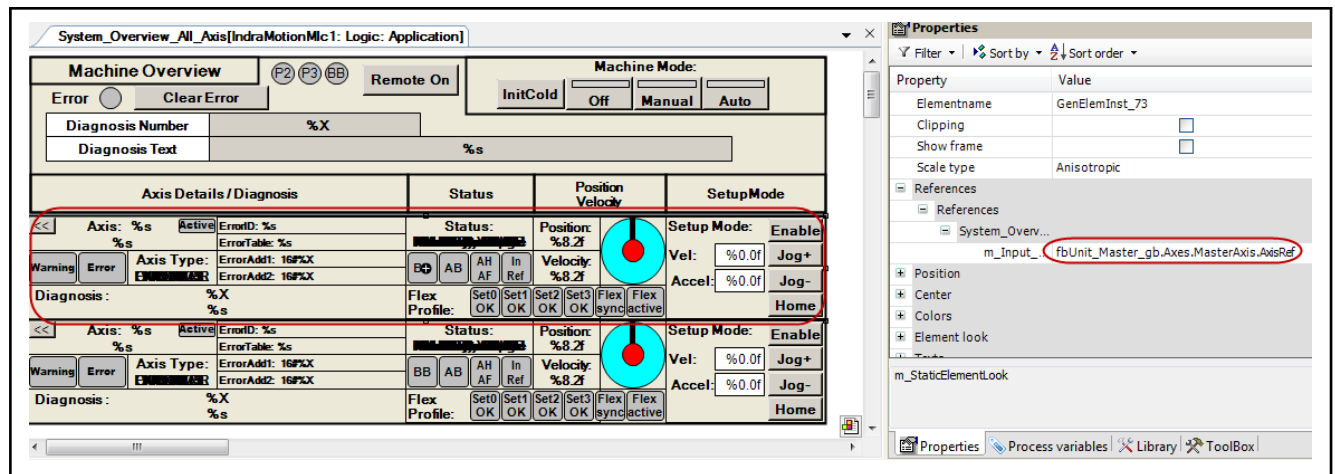


図 11-18: MasterAxis の軸表示の調整

fbUnit_Master_gb.Axes.MasterAxis.AxisRef

- 下の図で強調表示されている表示オブジェクトをクリックし、入力フィールド "Referenced visualizations" (参照される表示) でユニットおよび軸の名前を以下のとおり調整します。すると、"Unit1" ユニットの "SlaveAxis" に軸表示が適用されます。

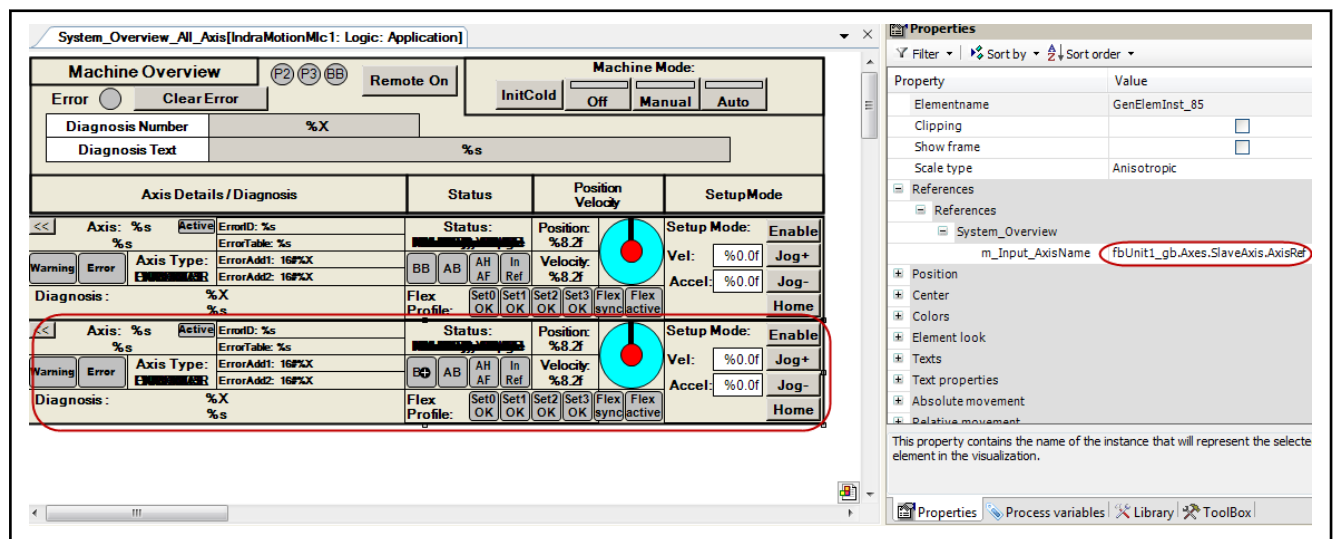


図 11-19: SlaveAxis の軸表示の調整

fbUnit1_gb.Axes.SlaveAxis.AxisRef

これによりプログラミングが完了します。

既定プロジェクトの設定エディタ(2)をダウンロードした後で、PLC を起動できます (3)。
と起動



図 11-20: PLC のダウンロード (設定/PLC オブジェクト) と起動

PLC が起動した後で、予め定義済みの内部表示 "System_Overview_All_Axis" を使用して GAT を操作できます。まず、表示の Remote On (リモート オン) ボタンをクリックし、GAT を制御できる状態にします。

分散状態の機械での GAT

次に、**Auto** (自動) ボタンをクリックし、運転モード "Automatic" (自動) を有効にします。これで、軸が適切に動作するようになります。

"Manual" (手動) 運転モードが選択されている場合は、軸が停止し、**Enable** (有効化) ボタンで各軸の **Setup** (セットアップ) モードを有効にできます。軸は、有効になった "Setup" (セットアップ) モードで、表示にある各ボタンを使用してジョグまたはホーミングが可能です。

11.4 機能原則

モーションアプリケーションを簡単かつ効果的に生成するために、GAT のプログラム コードは 2 つのセクションに分かれています。

さまざまなアプリケーションのために一般化が可能なプログラム コードは、GAT ライブラリ (ML_TechTemplate.compiled-library) の基本タイプおよび基本ブロックにあります。

ウィザードによるサポートを利用して、ユーザーはアプリケーション固有のプログラム コードをアプリケーション プログラム内の目的の位置に配置します。

この機能原則により、以下のメリットが得られます。

- コンパクトで分かりやすく整理されたユーザー コードの作成 :
 - 最適化されたプログラム テンプレートに基づいた簡素なインターフェースと明快な構造の使用を通じて
- より高い堅牢性 :
 - テストされ、最適化されたコード セクションの使用を通じて
- 保全性の向上
 - エラーを排除する際のライブラリ交換が容易なため

11.5 初期化

初期化は機械の起動時に実行され、2 つの手順で構成されます。

- 最初の手順では、"コールド スタート" (パラメータ化モード/P2) が有効化され、すべてのユニットに転送されます。初期化のためのアプリケーション固有のプログラム コードは、各ユニットのアクション "actInit" にあり、この状態で処理されます。コントロールのパラメータ化モード/P2 が有効になり、すべてのユニットで切替が有効になると、次の手順への切替が行われます (下の注意事項を参照)。
- 2 番目の手順では、"ウォーム スタート" (動作モード/BB) が有効化され、すべてのユニットに転送されます。コントロールの運転モード/BB が有効になり、すべてのユニットで切替が有効になると、初期化が完了します (下の注意事項を参照)。すると、機械のオペレータにより選択された運転モードが有効になります。初期化は、初期化運転モードを選択することにより、いつでも再度実行できます。



次の初期化手順への切替は、ユニット内の機能 "LockOpModeAcknowledge" を使用してブロックまたは有効化できます。

"LockOpModeAcknowledge := TRUE" を設定することにより、次の手順への切替は複数のサイクル (ドライブ パラメータの読み取りや書き込みなど) にわたってブロックできます。

SkipColdstartAtStartup

PLC プロジェクトのダウンロードまたは PLC のリセットを行った後で、デフォルトにより Init 運転モードの "コールド スタート" が有効になります。"コールド スタート" = パラメータ化モードへの切替には時間がかかるため、特に

分散状態の機械での GAT

当初の機械試運転の場合、この動作は障害を引き起こす可能性があります。このため、"**SkipColdstartAtStartup = TRUE**" 設定を使用して、この場合の "コールド スタート" への切替を防ぐことができます。この設定では、PLC のダウンロードまたはリセットの後、"コールド スタート" は省略され、"ウォーム スタート" のみが処理されます。

"SkipColdstartAtStartup" 設定の変更は、TE_MachineControl FB にある "mOneTimeInit" メソッドの割当てを行うことができます (下の図を参照)。変更済みの PLC プロジェクトをダウンロードした後で、新しい設定が有効になります。

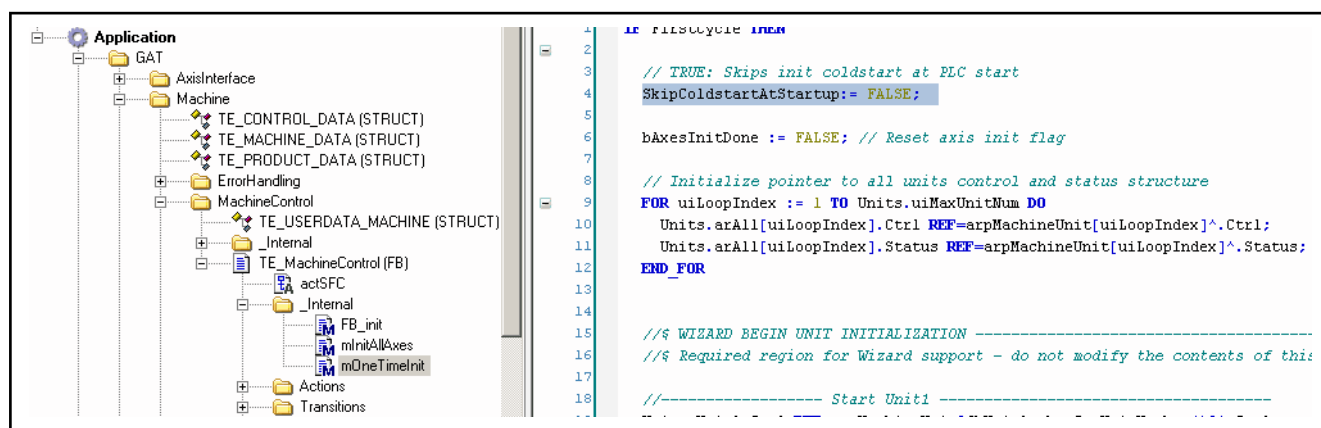


図 11-21: "SkipColdstartAtStartup" 構造要素を使用した初期化の設定

11.6 運転モード

一般情報

初期インポートで、GAT には予め定義済みのプライマリ運転モード "MODEOFF"、"AUTO" および "MANUAL" が含まれます。初期化は、すでに前章で取り上げました。

原則として、運転モードは以下のとおり区別されます。

- **プライマリ運転モード**は、すべての結合ユニットに転送され、運転モード SFC 内の分岐としてユニットに含まれます。
- **ユニット固有の運転モード**は、それぞれの非結合ユニットに固有の運転モードで、各運転モード SFC 内の分岐としてユニットに含まれます。

初期化/運転モード切替後のターゲット運転モード

初期化が完了した後で、オペレータが命令したプライマリ運転モードがすべての結合ユニットに転送され、有効になります。

すべての非結合ユニットの場合、ユニット固有の運転モードは FB "TE_MachineControl" のメソッド "mOpmodeDistribution" にあるプログラムコードを使用して各ユニットに指定されます。

運転モードの確認

選択されたプライマリ運転モードは、すべてのユニット (結合および非結合) が命令された運転モードを確認した後でのみ、機械のステータス出力 (変数 "stMachine_Status_gb") で確認されます。

機能 "LockOpModeAcknowledge" により、個別の各ユニットには運転モードの確認を延期するオプションがあります。

分散状態の機械での GAT

11.7 ユニット

11.7.1 一般情報

運転モード SFC

各ユニットには、周期的に呼び出される運転モード SFC (actOPModeSFC) が含まれています。設定されている運転モードにより、SFC 内の対応する手順がアクティブになり、関連する PLC コードが処理されます (自動モードのコードなど)。

ローカル エラー処理

エラーが発生した場合、運転モードが中断され、SFC はエラー処理に分岐します。各ユニットは割り当てられたコンポーネント (対応する軸など) のエラーリアクションを担当します。ユニット内でエラーが発生すると、情報が配信されるように、そのエラーが上位のエラー ハンドラーに転送されます。オプションとして、ユニット内の上位のエラー ハンドラーへのローカル エラーの転送を抑止できます ("[ユニットでのローカル エラー処理](#)" ページ 353 を参照)。

軸処理

ユニットに軸がある場合、軸は起動時に初期化され、ランタイム中に AxisInterface が呼び出されます。システムにより、各ユニット軸の AxisInterface が対応するユニットと同じタスクで自動的に呼び出されます。

アプリケーション プログラムでは、ウィザードを使用してユニットを簡単に追加、削除、変更できます。

原則として、ユニットは以下のとおり区別されます。

結合ユニット

オペレータが命令したプライマリ運転モードがすべての結合ユニットに自動的に転送されます。このため、結合ユニットで、すべてのプライマリ運転モードは運転モード SFC 内の分岐として含まれます。

ユニットへの運転モードの配信は、TE_MachineControl の メソッド **mOpmodeDistribution** で実行されます。ここで、プライマリ運転モードをすべての結合ユニットに転送するために、命令 "mSetUnitsOpModes(OpMode := OpMode, CoupledUnitsOnly := TRUE);" が使用されます。

結合ユニットにより、ランタイム中の運転モードの結合、およびその再確立が可能です。例えば、メンテナンス目的のためにこの機能を使用して運転モードリンクからユニットを削除し、他のすべてのユニットでは自動モードで生産を続行させることができます。

TE_MachineControl 内の命令 "Units.<UnitName>.Ctrl.OpModeCoupled := FALSE;" は、運転モードの結合をキャンセルするために使用します。この場合、機械で選択されたプライマリ運転モードはこのユニットに転送されません。この方法により、メソッド **mOpmodeDistribution** でユーザー プログラムを使用して、ユニットの任意のプライマリ運転モードを指定できます。

命令 "Units.<UnitName>.Ctrl.OpModeCoupled := TRUE;" は、運転モードの結合を再び確立するために使用され、ユニットは命令されたプライマリ運転モードを再び自動的に受信ようになります。

非結合ユニット

初期化が完了した後で、オペレータが命令したプライマリ運転モードは、非結合ユニットに自動的に転送されません。運転モードは、FB "TE_MachineControl" 内のメソッド "mOpmodeDistribution" を使用して、アプリケーション プログラムによって指定されます。

初期の GAT インポートの後で、1 つの結合ユニット "Unit1" がすでに利用可能です (下の図を参照)。ウィザードはユニット名の変更に使用できます。

分散状態の機械での GAT

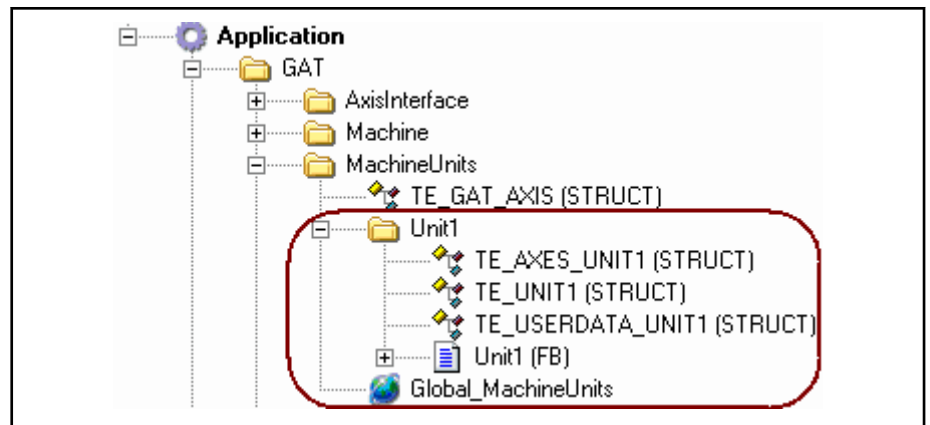


図 11-22: プロジェクトディレクトリ内のユニットフォルダ

11.7.2 作成規則

GAT ユニットの維持と再利用を容易にするために、モジュールは以下のように設計する必要があります。

- ユニット間の依存関係を最低限度に抑える
- ユニット内でグローバル変数を使用しない

これを実現するために、各ユニットには以下のインターフェースがあります。

11.7.3 ユニット インターフェース

一般情報

下の図は、統一されたユニット インターフェースを示したものです。

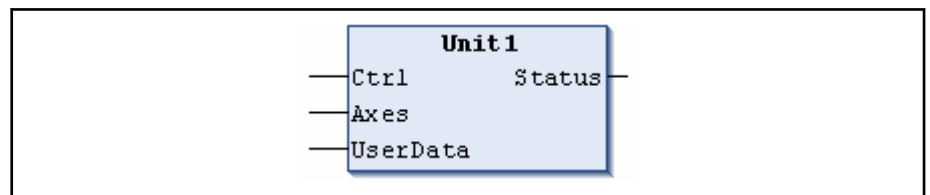


図 11-23: ユニット インターフェース

高性能なデータ処理のために、ユニット インターフェースの要素 "Axes" および "UserData" は、参照として実行されます。すなわち、ユニットで必要とされるデータは、ユニットにコピーされるのではなく、参照によってユニットに転送されます。

このインターフェースのコンポーネントについては、以下のセクションで説明します。

コントロールおよびステータスの構造

各ユニットには、制御のための変数 (Ctrl) と、ステータスを報告するための変数 (Status) があります。これらの要素は、GAT ライブラリの基本ブロックに含まれ、そこから管理されます。

ユニット軸

各ユニットのインターフェースには、軸コンテナ (Axes) があります。このコンテナの軸はすべて、ユニット内で能動的に命令を受けます。



受動的に使用される軸 (マスター軸など) は、パフォーマンスを理由として軸コンテナ **Axes** で設定を行ってはいけません。

受動的な軸の設定は、**Consumed Axes** 軸コンテナで行います。

分散状態の機械での GAT

軸コンテナの名称は、各モジュールで同じですが、内容 (タイプ) はユニットで使用される軸によって異なることがあります。

軸コンテナ内の各軸 (ユニット軸) はタイプ `TE_GAT_AXIS` で、これはタイプ `TE_BASE_MODULE_AXIS` に基づいています (ML_TechTemplate ライブラリを参照)。下の表は、構造タイプ `TE_GAT_AXIS` で利用可能なすべての要素を示したものです。

要素	タイプ	説明
名称	STRING(20)	文字列としての軸名 (システムが初期化された場合に初期化される、ユーザーによる変更が可能)
AxisData	REFERENCE TO ML_AXISDATA_SM	グローバル列 AxisData 内の対応する要素への参照 (ML_Base ライブラリを参照)
DV	REFERENCE TO ML_DirectVarAxis	ローカル軸がある場合：グローバル列 DV_Axis 内の対応する要素への参照 (ML_Base ライブラリを参照) リンク軸 (別のコントロールからの軸) がある場合：ダミー要素
GV	REFERENCE TO ML_GlobalVarAxis	グローバル列 GV_Axis 内の対応する要素への参照 (ML_Base ライブラリを参照)
AxisRef	AXIS_REF	軸の参照
Disable	BOOL	TRUE: 対応する AxisInterface データ セットは更新されない FALSE: デフォルト この変数の値に加えられた変更は、INIT モード (STATE_INIT_COLDSTART または STATE_INIT_WARMSTART) で初めて有効となる
Ctrl	REFERENCE TO TE_AXISINTERFACE_CONTROL	対応する軸の AxisInterface の制御構造への参照 (グローバル列 arAxisCtrl_gb の対応するエントリへの参照については、AxisInterface フォルダにあるグローバル変数リスト Global_GAT_AxisInterfaces を参照)
Status	REFERENCE TO TE_AXISINTERFACE_STATUS	対応する軸の AxisInterface のステータス構造への参照 (グローバル列 arAxisStatus_gb の対応するエントリへの参照については、AxisInterface フォルダにあるグローバル変数リスト Global_GAT_AxisInterfaces を参照)

表 11-1: タイプ `TE_GAT_AXIS` の利用可能なすべての要素。ユニット内で、要素は `Axes.<AxisName>.<Element>` からアクセス可能

軸のすべての関連データは、**Axes** 軸コンテナを使用してアドレスを指定できます。

このコンテナは、軸の名前を用いて軸データにアクセスできるように設計されています。これにより、読み取りが容易なプログラム コードが生成されます。

下の表は、ユニット内およびユニット外から軸コンテナにある軸にアクセスする方法を示したものです。

ユニット軸 (ユニット "Unit1" の軸コンテナにある軸 "vAxis" など) へのアクセス	軸の無効化
ユニット外のアクセス	<code>stAxesUnit1_gb.vAxis.Disable := TRUE;</code>
ユニット内のアクセス	<code>Axes.vAxis.Disable := TRUE;</code>

分散状態の機械での GAT

GAT ウィザードは、IndraWorks で設定されるユニット軸と軸 (実在、仮想...) の割当を確立します。これを行うには、FB TE_MachineControl のメソッド "mOneTimeInit" で、対応する物理軸をユニット軸に割り当てます (下の図および 376 ページ "軸の追加" 11.13.2 章を参照)。

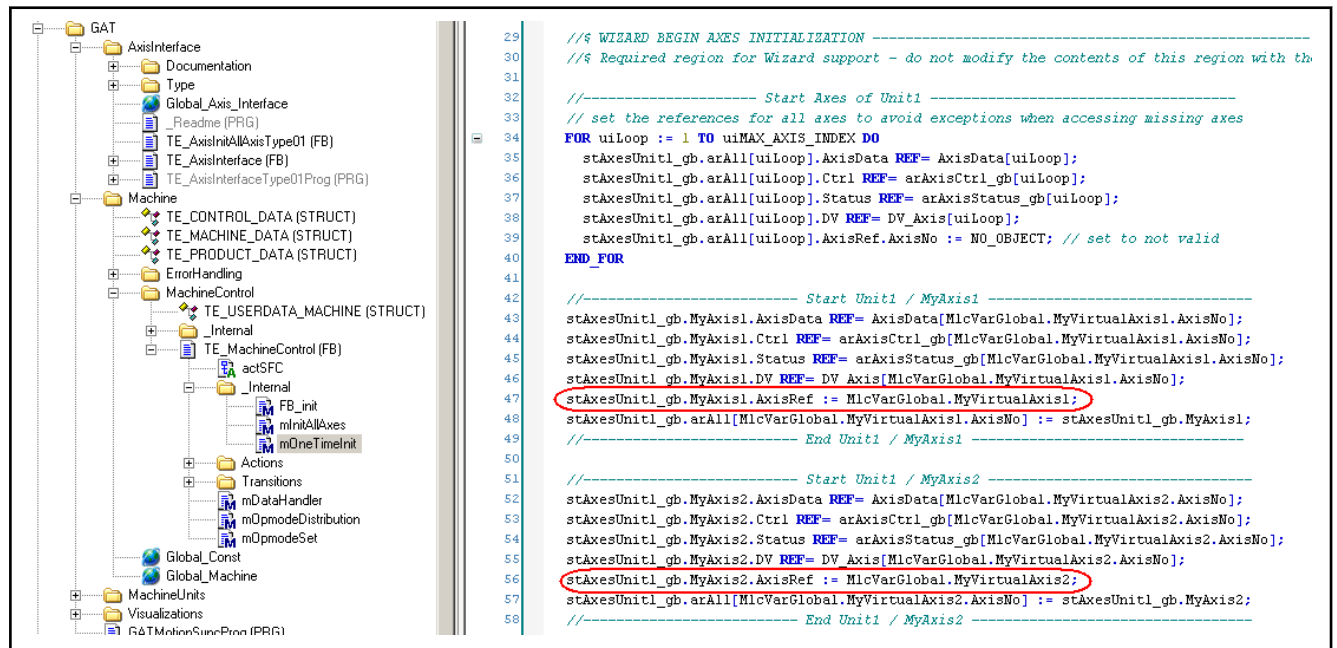


図 11-24: TE_MachineControl のメソッド "mOneTimeInit" での軸の割当

上の図からの例: IndraWorks で生成されたグローバル軸 "MyVirtualAxis1" は、ユニット "Unit1" の軸 "MyAxis1" を割り当てられます。

受動的な軸 (消費軸)

受動的に使用されるユニットの軸 (マスター軸など) をすべて **Consumed Axes** (消費される軸) 軸コンテナで設定します。

GAT ウィザードを使用して以下の手順を実行します。

- GAT ウィザードを開く (IndraWorks ツールバーにある GAT ボタン



- ユニットの下に **Consumed Axes** (消費される軸) ノードを選択する
- "Add Axis" (軸を追加) を選択してダイアログ クエリに従う

エンコーダ、リンクおよび C2C 消費軸も、受動的に使用される軸として使用できます。

Consumed Axes (消費される軸) 軸コンテナの使用は、以下のメリットに関連があります。

- パフォーマンス上のメリット AxisInterface は消費軸については呼び出されないため
- よりユーザーフレンドリー どのユニットでどの軸が能動的に命令を受けているかを素早く確認できるため

ユニットのプログラム コードでの消費軸の使用例

プログラム

```
// 例: MySlaveAxis1 は、VirtualMaster 1 が使用する軸と同期されます
Axes.MySlaveAxis1.Ctrl.SyncMode.Master := ConsumedAxes.VirtualMaster1;
Axes.MySlaveAxis1.Ctrl.Admin._OpMode.en := ModeSyncPhase;
```

C2C 消費軸の場合、GV_LinkAxis[] 変数を用いてユニット内で軸データ (位置、速度など) を読み取ることができます。

分散状態の機械での GAT

他のすべての軸のタイプの場合は、**DV_Axis[]** 変数を用いて消費軸の軸データを読み取ることができます。

消費軸は、要素 "CntlNo" および "Axis No" を持つデータ タイプ **AXIS_REF** に対応します。

ユニット固有の I/O データ

再利用可能性と複数のインスタンス化を実現するために、ユニット内のグローバル データへのアクセスは許可されません。ユニットに必要なすべての I/O は、I/O コンテナ **UserData** を使用して利用可能になります。I/O コンテナの名前は各ユニットで同じですが、内容 (タイプ) は異なります。これは、すべてのユニットが同じ I/O データを使用するわけではないためです。

変数 **UserData** は、ユーザーが要素を定義する必要のあるデータ構造です。

下の図は、GAT に含まれている Unit1 の I/O データのどの部分が定義されるかを示したものです。

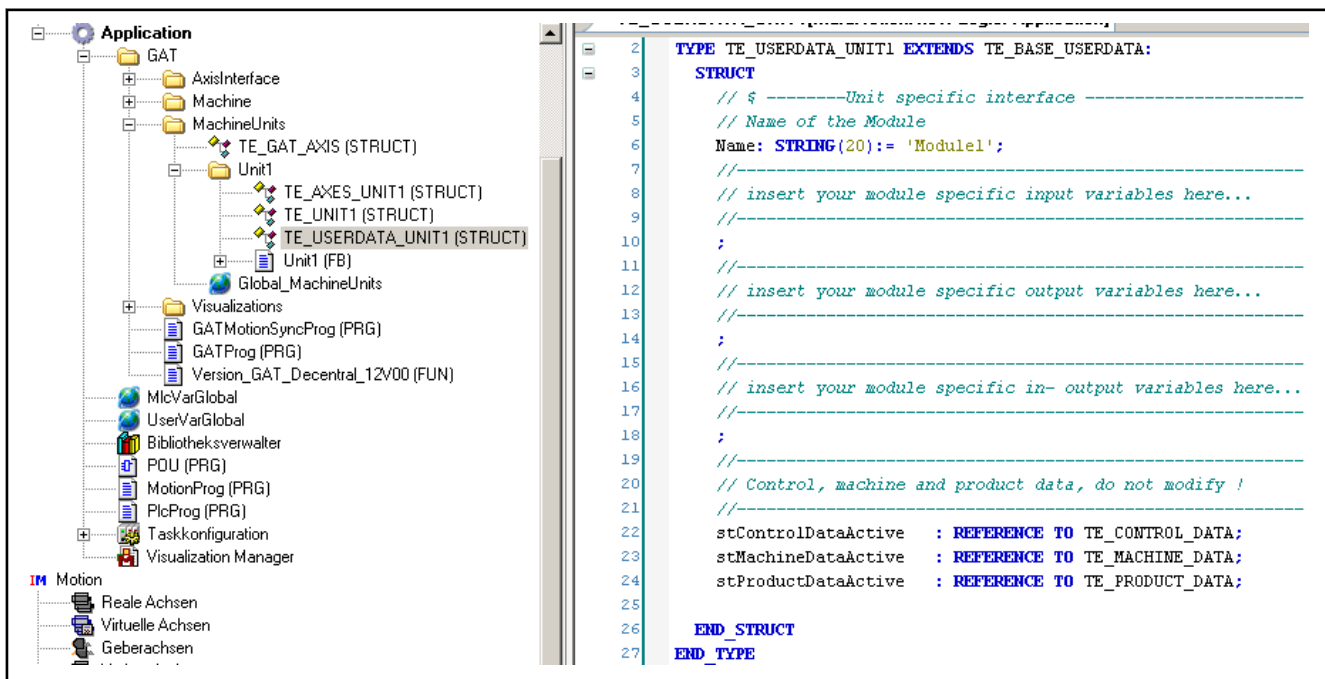


図 11-25: ユニット固有の I/O データ

ユニット インスタンス

既存のユニットから新しいインスタンスを作成できます。これらのインスタンスは、割り当てられる IndraWorks 軸および割り当てられる I/O のみが異なります。GAT ウィザードはユニットのインスタンス化をサポートしています。

11.7.4 ユニット内の運転モード

新しいプライマリ運転モードを追加する場合、ウィザードによってすべての結合ユニットで対応する分岐が運転モード SFC に作成されます。この手順は、対応するプライマリ運転モードが選択された後でのみ処理されます。

機能 "LockOpModeAcknowledge" により、個別の各ユニットには運転モードの確認を延期するオプションがあります。この方法により、次の初期化手順への切替を延期することもできます。例えば、関連するアクティビティに複数のサイクル (パラメータの書き込みなど) が必要な場合です。

運転モードの確認は、ユニットの メソッド "mOpModeAcknowledge" で実行されます。このメソッドはウィザードによって作成され、プライマリ運転モードが変更された場合に改変されます。

11.7.5 逐次プログラミング

逐次プログラミング入門

逐次プログラミングと GAT 間の インタラクション

逐次プログラミングは、IndraMotion MLC、特に自動プロセスでの逐次プログラミングを容易にすることを目的としています。シーケンシャル コマンドには IEC - 61131 (構造テキスト) プログラミング言語が使用されます。IndraWorks に組み込まれているエディタも逐次プログラミングに使用できるため、追加的なプログラミング環境は必要ありません。逐次プログラミングは GAT (分散状態の機械での) に完全に組み込まれています。GAT ウィザードは逐次プログラムの設定のために拡張されました。ユーザーはウィザードを使用して逐次プログラムを作成または削除できます。シーケンシャルコマンドを実行する、対応する軸は、逐次プログラムが作成されると、自動的に準備されます。

逐次プログラムは、ユニットの 1 つの運転モードのみに割り当てられます。試運転の前に、プログラムと軸を設定する必要があります。逐次プログラミングは GAT ウィザードにより、以下のとおり設定されます。

- 運転モードの逐次プログラムは、ユニット フォルダ <UnitName> の下のフォルダ (Mode <Operating mode>) に保存されます。プログラム中に逐次シーケンスがプログラミングされます。
- 各逐次プログラムは、個別のタスクに割り当てられます。タスクは、タスク設定に置かれています。
- 逐次プログラムを起動および停止するためには、それぞれの場合に "SeqTaskControl" ブロックのインスタンスが 1 つ必要です。このブロックのインスタンスは、対応するユニットのグローバル変数リスト Global <UnitName> に挿入されます。デフォルトにより、逐次プログラムの起動および停止のコマンドは、各運転モードの入力または出力の手順に挿入されます。

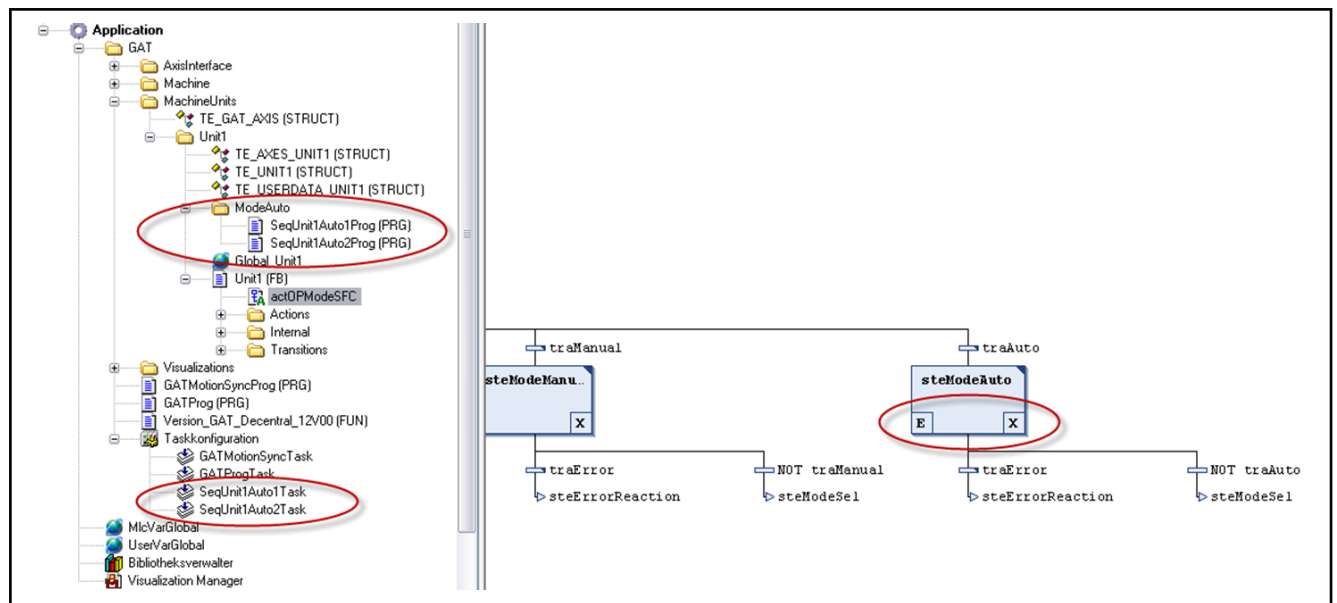


図 11-26: 逐次プログラミングによる入力および出力アクション

- GAT ウィザードは逐次プログラミングにユニット軸を割り当てます。これを行うには、TE_MachineControl ファンクションブロックの "mOneTimelnit" メソッドにある Seq[...] フィールドにユニット軸の軸参照を割り当てます。
- シーケンシャル軸ハンドラーは、"GATMotionSyncProg" で動作と同期して呼び出されます。

分散状態の機械での GAT

逐次プログラミングに関するその他の注意

- 運転モードの出力手順で、GAT ウィザードによりコマンド Seq[...].Disable() が各軸について入力されます。このコマンドにより、運転モードが閉じられたときに軸への逐次プログラミングのアクセスが終了します。

逐次プログラミングに関するその他の注意 (ファンクションブロック、コマンドおよびインターフェースなど) は、逐次プログラミングの機能説明に含まれています (資料「"Rexroth IndraLogic XLC IndraMotion MLC xxVRS Sequential Programming"」(Rexroth IndraLogic XLC IndraMotion MLC xxVRS 逐次プログラミング) を参照)。

xx - バージョン

11.7.6 ユニットの連携

ユニットは、他のユニットに関する情報は一切含まないため、ユニットの連携は上位の位置で行う必要があります。

上位ユニットの連携には以下のファンクションブロックが指定されます。

- **TE_ErrorHandling** はユニット間のエラー配信を調整します。基本バージョンでは、すべてのエラーがすべてのユニットに配信されます。
- **TE_MachineControl** は、エラーに関連しないすべてのユニット間アクティビティ (総合的な運転モード管理など) を処理します。

ファンクションブロック **TE_MachineControl** には、すべてのサイクルで呼び出されるメソッド **"mCyclic"** がすでに含まれています。周期的に呼び出されるこのメソッドで、I/O 情報がユニット間で、およびユニットから機械の I/O ポイントへと交換されます。

ユニット設定または対応するユニット軸へのグローバル軸の割当はランタイム中に変化しないため、これらのアクティビティは **TE_MachineControl** のメソッド **"mOneTimeInit"** で実行されます。このメソッドは、機械構成ユニットを使用して GAT を初期化するために呼び出されます。

"mOneTimeInit" メソッドに含まれているコードは、ウィザードにより管理されます。ユニットまたはユニット軸を追加もしくは削除する場合、対応するエントリが追加または削除されます。

11.8 タスク管理

11.8.1 タスク設定、GAT

デフォルトにより、GAT には以下のタスクが含まれています。

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度	プログラム呼出
GATMotionSyncTask	2	=モーションサイクル時間	T#2ms	1	GATMotionSyncProg
GATProgTask	5	T#10ms	T#20ms	2	GATProg

表 11-2:

11.8.2 BackgroundTask の追加

PlcTask のランタイムが長すぎる場合は、タイム クリティカルではないタスクをバックグラウンド タスクに移動して、PlcTask の負荷を減らすことができます。ただし、追加的なタスク Switch PlcTask <> BackgroundTask により、より大きな負荷がシステム全体にかかることに留意してください (34 ページ "タスク管理" 5.5 章 も参照)。

分散状態の機械での GAT

	優先度	サイクル時間	ウォッチドッグ時間	ウォッチドッグ感度
BackgroundTask	~6	T#40ms	-	-

BackgroundTask を追加するためには、以下の手順が必要となります。

- タスク設定、タスク "BackgroundTask" の追加。設定については上の表を参照してください。
- "BackgroundProg" を追加し、"BackgroundTask" の設定で "BackgroundProg" を呼び出します
- FB TE_MachineControl でメソッド "mBackground" を追加します。続いて、このメソッドを "BackgroundProg" で呼び出します。呼出によるプログラムライン：fbTE_MachineControl_gb.mBackground()。"FB TE_MachineControl" のすべての変数は、メソッド "mBackground" で利用可能です。

11.9 エラー処理

11.9.1 エラーの検出および診断

プログラム テンプレート GAT には、エラーを検出、処理、削除するために予め定義済みの機能が含まれています。基本的な手順について以下に説明します。

ユニットでのローカル エラー処理

エラー処理はユニット内に限定されます。各ユニットは割り当てられたコンポーネントのエラー反応を担当します。例えば、ユニットのドライブでエラーが発生した場合、そのユニットにもローカル エラーが発生します。デフォルトにより、他のユニットもエラーに対応できるように、このローカル エラーをエラー ハンドラーに転送する "error" 出力が設定されます。

エラー ハンドラーへのエラーの転送

エラーが発生した場合、ユニット ステータス出力の "error" ビットが自動的に設定されます。特定のケースでこれが望ましくない場合は、メソッド **mForwardLocalError** をユニットに追加することにより、エラー転送を抑制できます。エラーが発生した場合にはユニットのメソッド **mForwardLocalError** が GAT によって自動的に呼び出され、また "DiagNo" を介して検出されたローカル エラーの診断番号も受信します。アプリケーション コードで、"DiagNo" を評価し、その情報に基づいて、エラーを転送する (mForwardLocalError := TRUE) か、または転送しません (mForwardLocalError := FALSE)。

*メソッド **mForwardLocalError** を追加するとき、以下のインターフェースを宣言する必要があります。*

```
METHOD mForwardLocalError : BOOL
VAR_INPUT
    DiagNo:    DWORD;    // エラー番号
END_VAR
```

エラーが転送されない場合でも、通常の GAT エラーリアクションがオンになっていれば、同じサイクルで直ちにユニット内でローカル エラーリアクションが発生します。

ローカル エラーの発生元

ユニット内のエラーの発生元としては、以下を考慮に入れます。

- 軸エラー (軸ラグ エラーなど)
- アプリケーション固有のエラー (例：安全扉が開いている)
- ファンクションブロック エラー (例：呼び出されたファンクションブロックのエラーが原因)

ユニット間のエラー処理

FB TE_ErrorHandling は中心的な位置からユニット全体にわたり、エラーのすべての発生元をモニタリングします。エラーが検出された場合、エラーは外部エラー (ユニット入力 "Ctrl.ExtError" および "Ctrl.ExtErrDiagNo") としてす

分散状態の機械での GAT

すべてのユニットに転送されます。 オプションとして、アプリケーション プログラム内の各ユニットへのエラー配信を抑止することも可能です。 "TE_Errorhandling" のメソッド **mDistributeError** を適宜、適合させる必要があります。 その後、次のサイクルで対応するエラーリアクションが各ユニットで発生します。

ファンクションブロック **TE_ErrorHandling** はエラーの確認の調整を続行し、この情報を中心的な位置からすべてのユニットに転送します。 その後、ユニットは割り当てられたコンポーネントでのエラー確認を担当します。

複数のエラーが同時に発生した場合は、最も深刻なものが表示されます。 同じ深刻度のエラーが複数、同時に発生した場合は、最後に検出されたものが出力されます。

機械の診断は、変数 **stErr_Handling_Status_gb** を使用して **TE_ErrorHandling** の出力で出力されます。

要素	タイプ	説明
エラー	BOOL	TRUE はエラーを示します
DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの： 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) (xxxx = ユーザーが定義可能) 16#F0FB 0000: ファンクションブロック エラー (例: TechFB でのエラー) -> 詳細については、ErrorID / ErrorIdent を参照 16#F0AA 0000: GAT 内部エラー -> 詳細については、ErrorID/ErrorIdent を参照 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)
ErrorID	ERROR_CODE	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
ErrorIdent	ERROR_STRUCT	FB + GAT エラー (DiagNo = 16#F0FB/AA 0000) の場合の詳細な診断
DiagMessage	STRING(60)	ドライブおよびコントロールのエラー診断テキストを含む

DiagNo 要素は、HMI での使用のみを目的としています。 エラー テキストには、任意の言語で数値情報が割り当てられます。

DiagMessage、**ErrorID** および **ErrorIdent** は、IndraLogic 表示の使用など、プログラマーによる試運転の際に主に表示される追加情報を提供します。

アプリケーション固有のエラーに関する注意

mSetAppError メソッド "mSetAppError(Number)" を呼び出すことにより、アプリケーション エラー (例:安全扉が開いている) を表示できます。 メソッド入力で伝達される"番号"は、アプリケーション エラー番号に対応します。

アプリケーション エラーが表示された場合、ユニットの中央診断システムは、表示されたアプリケーション エラーの周期的なモニタリングを行い、アプリケーション エラー DiagNo = Error 16#F0AE xxxx (xxxx = アプリケーション エラー番号) を生成します。

uiMAX_MODULE_APP_ERROR_INDEX の範囲 1 にある番号は、ユニットでのアプリケーション エラーについて有効です。 定数 "uiMAX_MODULE_APP_ERROR_INDEX" の値はデフォルトにより 64 に設定されています。 必要な場合は、"Machine" (機械) フォルダにあるグローバル変数リスト "Global_Const" でこの定数を調整できます。

分散状態の機械での GAT

uiMAX_MACHINE_APP_ERROR_INDEX の範囲 1 にある番号は、機械でのアプリケーション エラーについて有効です。定数 "uiMAX_MACHINE_APP_ERROR_INDEX" の値はデフォルトにより 64 に設定されています。必要な場合は、"Machine" (機械) フォルダにあるグローバル変数リスト "Global_Const" でこの定数を調整できます。

アプリケーション エラーを表示するためのコード例：

```
// Recommended code to issue user specific errors (Example)
IF bDoor10open THEN
    mSetAppError(1);
END_IF
```

深刻なエラーに関する注意

コントロール診断システムでは、10 段階の"深刻度"が定義されています (「IndraMotion MLC diagnostics」 (IndraLogic XLC-/IndraMotion MLC 診断) を参照)。

深刻度はエラー反応に影響を及ぼします。

- DiagNo = 16#F0xx xxxx は最も低い深刻度です (例：アプリケーション エラー、ファンクションブロック エラー)
- DiagNo = 16#F9xx xxxx は最も高い深刻度です (致命的なシステム エラー)

複数のエラーが同時に発生した場合は、深刻度が最も高いエラーが表示されます (41 ページ "エラーリアクション" 5.6.4 章 を参照)。

ファンクションブロック エラーに関する注意

mSetFBError

呼び出されたファンクションブロック (技術 FB など) でエラーが発生した場合は、メソッド "mSetFBError(ErrorID, ErrorIdent)" を呼び出して、エラーを引き起こしたファンクションブロックの "ErrorID" と "ErrorIdent" 出力をメソッドに転送することをお勧めします。これにより、エラーがユニットの診断システムに確実に転送され、"ErrorID" と "ErrorIdent" がユニットの診断出力で利用可能になります。

ファンクションブロック エラーを表示するためのコード例：

```
// Recommended code a used FB issues an error (Example)
IF fbCrossCutter.Error THEN
    mSetFBError(ErrorID:=fbCrossCutter.ErrorID,
                ErrorIdent:=fbCrossCutter.ErrorIdent);
END_IF
```

11.9.2 エラーリアクション

エラーリアクションは、コントロールおよびドライブ システムにおけるエラーによる損傷を可能な限り避けるように設計されています。このため、深刻度 2 以上のエラーリアクションは、制御ファームウェアと実在の軸によって自動的に実行されます (41 ページ "エラーリアクション" 5.6.4 章 を参照)。

深刻度が第 2 段階未満のその他のエラー (アプリケーション エラー、ファンクションブロック エラーなど) の場合、制御ファームウェアと実在の軸による自動エラーリアクションはありません。

このため、GAT には、ユーザー固有のエラーリアクションを開始するエラー検出機能が含まれています。

通常の GAT エラーリアクション

デフォルト設定では、GAT エラーリアクションは有効になっています。エラーが発生すると、選択されている運転モードが中断され (例外：初期化)、エラーリアクションが実行されます。エラーリアクションは、すべての GAT ユニットのアクション "actErrorReaction_entry" に含まれています。各モジュールは対応するコンポーネント (軸など) のエラーリアクションを担当します。

分散状態の機械での GAT

ユニット アクション *actErrorReaction_entry* (すべての軸が減速) でのエラーリアクションの標準的なケース

```
// ----- Error reaction example 1 -----
// Decelerate all module axes to standstill --> switch all axes to ModeAH
// -----
FOR uiLoopIndex := 1 TO Axes.uiAxesNum DO
  IF mGetAxis(AxisNumber:=uiLoopIndex, PointerToAxis => pCurrentAxis) THEN
    IF pCurrentAxis^.AxisData.Axis_InAb THEN
      pCurrentAxis^.Ctrl.Admin._OpMode.en := ModeAH;
      // reset SetupMode Inputs to avoid hanging buttons in the
      // System_Overview_All_Axis visualisation
      pCurrentAxis^.Ctrl.SetupMode.Enable := FALSE;
      pCurrentAxis^.Ctrl.SetupMode.JogMinus := FALSE;
      pCurrentAxis^.Ctrl.SetupMode.JogPlus := FALSE;
      pCurrentAxis^.Ctrl.SetupMode.Homing := FALSE;
    END_IF
  END_IF
END_FOR
```



"ModeAb" を使用して油圧軸を静止させることをお勧めします。

上記の例で "ModeAH" が "ModeAb" で置き換えられた場合、減速で静止した後で実在の軸のトルク有効化がアクティブになります。



"軸が一時停止している場合"、軸の降下を防ぐためにトルクが有効化されないことがあります。これらのアプリケーションでは、すべての軸を "ModeAH" に切り替えることをお勧めします。

軸を同期させて減速する必要があることも少なくありません (材料ウェブの断裂を防ぐためなど)。これらのアプリケーションでは、同期させた動きですべての軸を接続したままにすること、およびマスターのみを減速することをお勧めします。このユースケースは下記の例に示されています。

ユニット アクション *"actErrorReaction_entry"* (同期していない軸のみが "ModeAH" を使用して減速される) での軸の同期減速の例

```
// ----- Error reaction example 2 -----
// Keep all axes in synchronization with the master axis
// and decelerate only the master axis
// -----
FOR uiLoopIndex := 1 TO Axes.uiAxesNum DO
  IF mGetAxis(AxisNumber:=uiLoopIndex, PointerToAxis => pCurrentAxis) THEN
    // reset SetupMode Inputs to avoid hanging buttons in the
    // System_Overview_All_Axis visualisation
    pCurrentAxis^.Ctrl.SetupMode.Enable := FALSE;
    pCurrentAxis^.Ctrl.SetupMode.JogMinus := FALSE;
    pCurrentAxis^.Ctrl.SetupMode.JogPlus := FALSE;
    pCurrentAxis^.Ctrl.SetupMode.Homing := FALSE;
    IF (pCurrentAxis^.AxisData.SynchronizedMotion) THEN
      ; // Axis is in sync motion --> no reaction as the master axis
      // decelerates to standstill and the slave axis follows in sync
    ELSE
      IF pCurrentAxis^.AxisData.Axis_InAb THEN
        pCurrentAxis^.Ctrl.Admin._OpMode.en := ModeAH;
        ; // Axis is not in sync motion --> command to stop
      END_IF
    END_IF
  END_IF
END_FOR
```

実在の軸の同期減速には、以下の設定が必要です。

- モジュール バス反応をオフにする
- ドライブのエラーリアクションを "可能な最善のシャットダウン = 30 s NC 対応" により設定する

分散状態の機械での GAT

通常の GAT エラーリアクション
をオフにする

IndraLogic XLC/IndraMotion MLC コントロール システム の実在の軸の "可能な最善の減速 = 30s NC 対応" 設定で F2 エラーが発生した場合、エラーの影響を受ける軸は緊急停止ランプ A-0-0224 により減速されます。問題のない他のすべての軸は、影響を受けません。

特別な場合は、各ユニットについて通常の GAT エラーリアクションを個別にオフにできます。エラーリアクションのスイッチオフは、例えば、エラー状態でも軸を動作させるために、点検運転モードで使用できます。ユニット制御ワードでビット "DisableErrorReaction" を設定すると、そのユニットでの通常の GAT エラーリアクションが無効になります。

必要なプログラム コードを以下のとおり実装します。

- 位置： ファンクションブロック "TE_MachineControl"、アクション "actInitColdStart"
- ここで、プログラム コードを以下のとおり修正します。

すべてのユニットについて通常の GAT エラーリアクションをオフにする

```
FOR uiLoopIndex := 1 TO Units.uiMaxUnitNum DO
    Units.arAll[uiLoopIndex].Ctrl.DisableErrorReaction := TRUE;
END_FOR
```

1 つのユニットについて通常の GAT エラーリアクションをオフにする
(<UnitName> をユニットの名前で置き換える)

```
Units.<UnitName>.Ctrl.DisableErrorReaction := TRUE;
```

初期化中の GAT エラーリアクシ
ョン

ユニットのエラーリアクションは、ファンクションブロック "TE_MachineControl" での初期化の後でオンとオフの切替ができます。

デフォルト設定では、初期化中、GAT エラーリアクションは無効になっています。しかし、エラーは継続して表示され、クリアすることも可能です。特別な場合、初期化中に GAT エラーリアクションをオンにすることができます。

必要なプログラム コードを以下のとおり実装します。

- 位置： ファンクションブロック "TE_MachineControl"、アクション "actInitColdStart"
- ここで、プログラム コードを以下のとおり修正します。

すべてのユニットの初期化中の通常の GAT エラーリアクションをオンにする

プログラム

```
FOR uiLoopIndex := 1 TO Units.uiMaxUnitNum DO
    Units.arAll[uiLoopIndex].Ctrl.EnableErrorReactionInit := TRUE;
END_FOR
```

1 つのユニットについて通常の GAT エラーリアクションをオンにする
(<UnitName> をユニットの名前で置き換える)

```
Units.<UnitName>.Ctrl.EnableErrorReactionInit := TRUE;
```



初期化中のエラーリアクションは、通常の GAT エラーリアクションが有効な場合 (DisableErrorHandling_bg = FALSE) のみ実行されます。

11.9.3 エラーのクリア

一元的なエラーのクリアは、変数 "stErr_Handling_Ctrl_gb" ("TE_ErrorHandling" の入力) の要素 "ClearAllError" の正のエッジによってト

分散状態の機械での GAT

リガーされ、すべての下位コンポーネント (IMC インターフェース¹⁾、ユニット、AxisInterface など) に自動的に転送されます。

エラーのない状態が達成された場合、オペレータによって選択された運転モードが再び有効になります。

11.9.4 プロトコル エラー

エラー処理制御構造 (変数 "stErr_Handling_Ctrl_gb") で構造要素 "LogError" が設定されている場合 (TRUE)、以下の GAT 固有エラーは、コントロールのエラー/診断メモリにも入力されます。

- 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) (xxxx = ユーザーが定義可能)
- 16#F0FB 0000: ファンクションブロック エラー (技術 FB でのエラーなど)
- 16#F0AA 0000: GAT 内部エラー
- 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)



"LogError" が TRUE に設定されている場合、GAT 固有エラーはコントロール ディスプレイおよびコントロールの診断パラメータにも表示されます。

11.10 製品と機械のデータ処理

11.10.1 はじめに

GAT は、精密に定義された時点で有効化されるデータ セットとして機械と製品のデータを準備するための機能を備えています。機械と製品のデータは、個別に表示され、管理されます。以下に説明するデータ処理は、"TE_MachineControl" のメソッド "mDataHandler" にオープン ソース コンセプトとして実装され、必要に応じて拡張または変更が可能です。

11.10.2 機械データの処理

機械データは機械を参照するデータで、運転中に変更が可能です。例としては、生産速度、加速および減速ランプなどが挙げられます。

これらの機械データは、変数リスト "Global_Machine" の変数 "stMachineDataEdit_gb" を使用して編集/準備されます。これらの変数の内容はアプリケーション固有で、データ構造 "TE_MACHINE_DATA" を使用して指定されます。

入力 "stControlDataActive_gb.bSetMachineData" の正のエッジにより、これらのデータが変数 "stMachineDataActive_gb" に書き込まれ、有効になります。確認は、"stControlDataActive_gb.bSetMachineDataDone" を使用して実行されます。

反対に、入力 "stControlDataActive_gb.bGetMachineData" の正のエッジにより、アクティブなデータ セットが、編集可能なデータ セット "stMachineDataEdit_gb" にコピーされ、その後、"stControlDataActive_gb.bGetMachineDataDone" を使用して確認されます。

11.10.3 製品データの処理

製品データは製品について説明するデータで、製造に必要となります。例としては、製品の長さ、レシピ データなどが挙げられます。

¹⁾ IndraMotionControl インターフェース = 制御へのインターフェースを備えた IndraLogic ライブラリ。例: エラーをクリア

分散状態の機械での GAT

これらの製品データは、変数リスト "Global_Machine" の変数 "stProductDataEdit_gb" を使用して編集/準備されます。これらの変数の内容はアプリケーション固有で、データ構造 "TE_PRODUCT_DATA" を使用して指定されます。

入力 "stControlDataActive_gb.bSetProductData" の正のエッジにより、これらのデータが変数 "stProductDataActive_gb" に書き込まれ、有効になります。確認は、"stControlDataActive_gb.bSetProductDataDone" を使用して実行されます。

反対に、入力 "stControlDataActive_gb.bGetMachineData" の正のエッジにより、アクティブなデータ セットが、編集可能なデータ セット "stProductDataEdit_gb" にコピーされ、その後、"stControlDataActive_gb.bGetProductDataDone" を使用して確認されます。

11.11 表示

11.11.1 表示の切替

GAT には、組み込みの試運転表示または外部表示により機械を制御するために準備された機能があります。ただし、一度に有効にできるのは 1 つの表示のみです。これを実現するために、組み込みの試運転表示と外部表示の両方が、それぞれ独自の変数を持っています。

組み込みの試運転表示は、グローバル変数リスト "GlobalMachine" からの変数 "stControlDataVisuIL_gb" を使用します。

外部表示 (フィールド バスまたはカスタマー HMI 経由など) は、グローバル変数リスト "GlobalMachine" からの変数 "stControlDataCustomer_gb" を使用します。

アプリケーション プログラムでは、組み込みの表示または外部表示のいずれかを参照する変数 "stControlDataActive_gb" が有効です。組み込みの試運転表示は、Remote On (リモート オン) ボタンを使用して参照の切替を行うことができ、これによりコントロールを得るか、またはコントロールを放棄します。

分散状態の機械での GAT

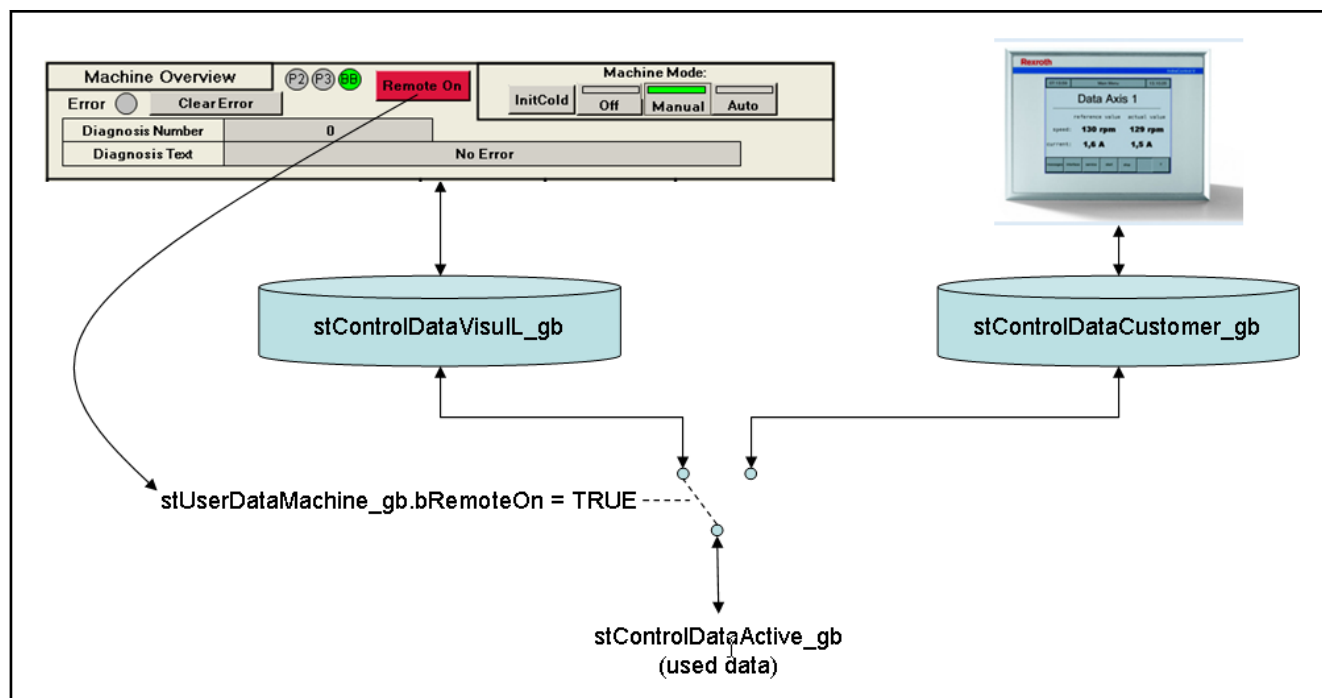


図 11-27: 試運転表示と外部表示の切替

11.11.2 試運転サポートのための内部表示

一般情報

GAT には、試運転中の支援および拡張された診断機能を提供する、予め定義済みの IndraLogic 表示オブジェクトが含まれています。

System_Overview

表示要素 "System_Overview" は、個々の軸についてインスタンス化できます。これは "MyVirtualAxis1" 軸 および "MyVirtualAxis2" 軸 の "System_overview_All_Axis" 表示で例として実行されています。

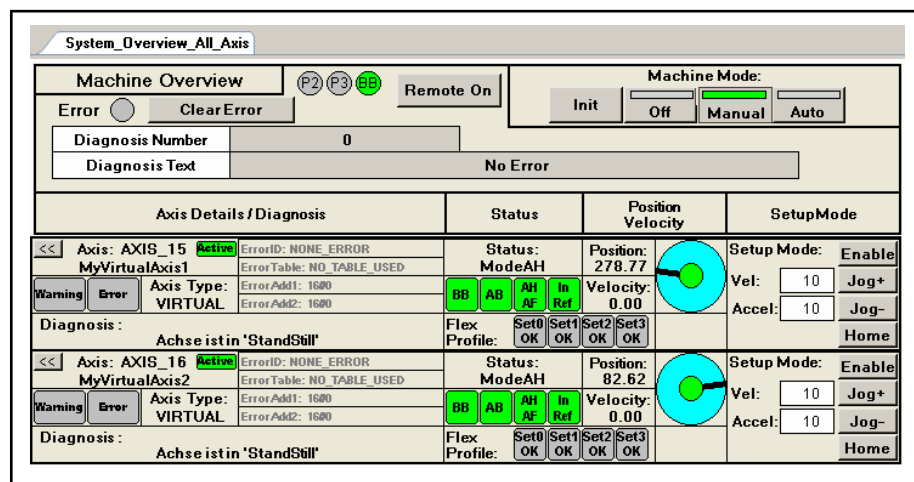


図 11-28: 起動中の支援のための内部表示



試運転表示は、**Remote On** (リモート オン) ボタンをクリックした後でのみ有効になります。

分散状態の機械での GAT

"Frame" オブジェクトを使用して、表示内で表示オブジェクト "System_Overview" のインスタンスを作成することができます。

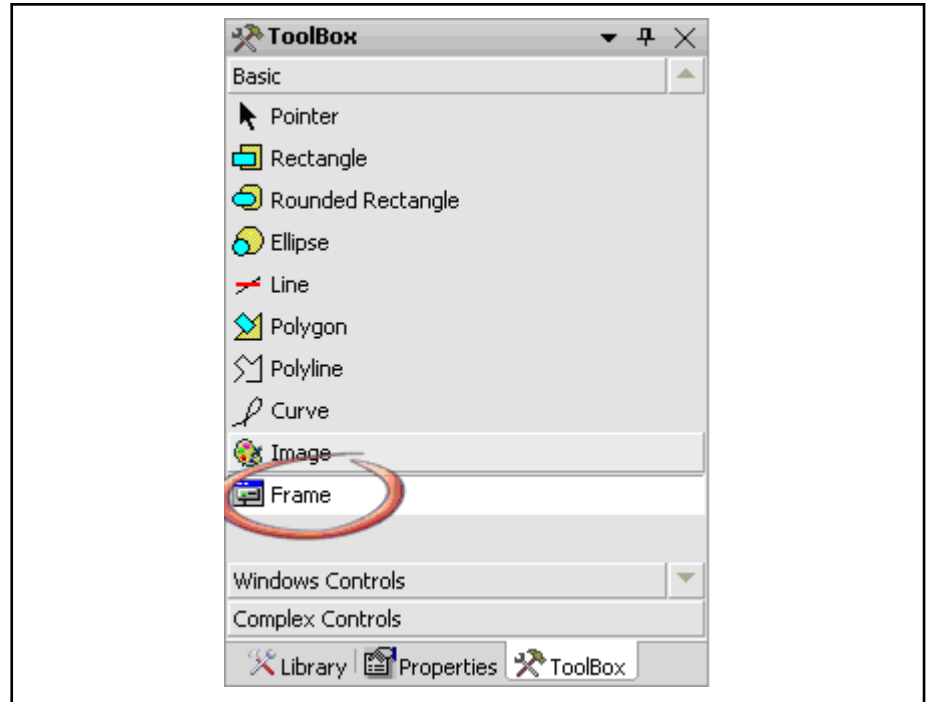


図 11-29: 表示オブジェクト "System_Overview" のインスタンスの作成

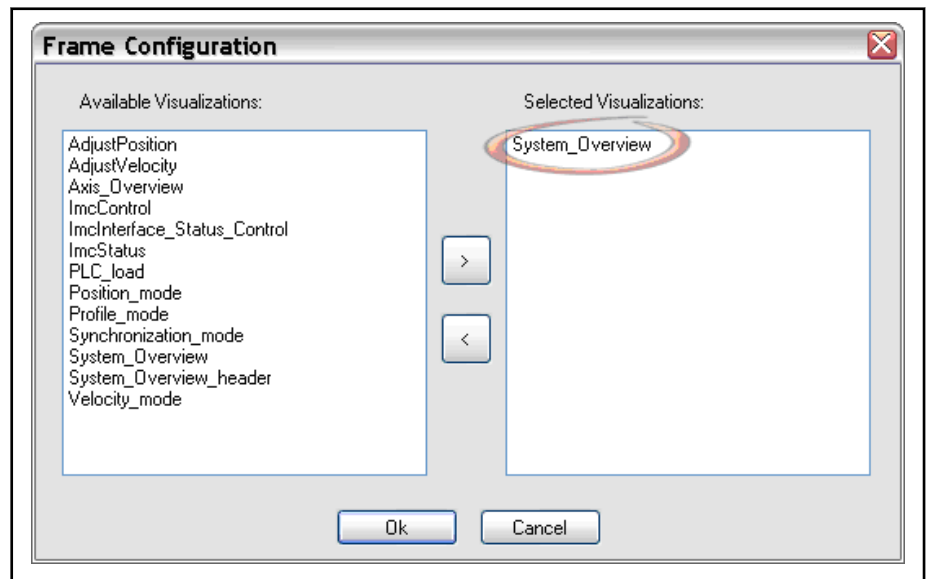


図 11-30: 表示オブジェクト "System_Overview" のインスタンスの作成

分散状態の機械での GAT

ここで、対応する軸参照を「Properties」の「References」の項目に入力します。

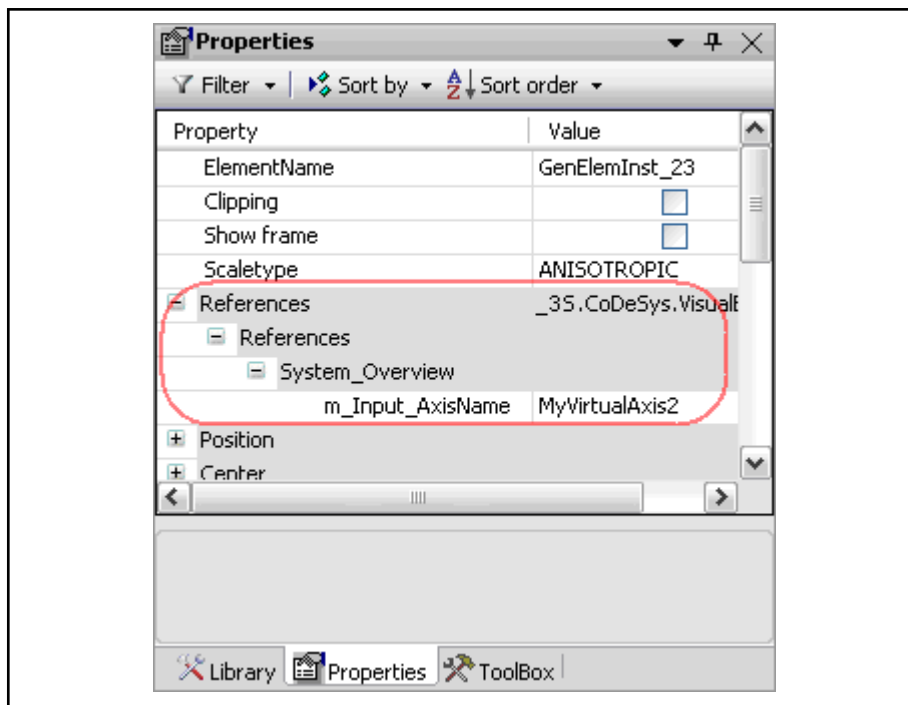


図 11-31: 表示オブジェクト "System_Overview" のインスタンスの作成

11.11.3 独立した HMI による外部表示 (WinStudio 表示)

一般情報

GAT では予め定義済みの WinStudio 表示が利用できます。ウィザードを使用して、この表示をプロジェクトに追加できます。



WinStudio を編集するには、少なくとも以下の開発環境のライセンス (またはそれ以上) が必要です。

WinStudio 開発ライセンス、変数最大 1,500 個 (Mat.No. R911323596)

VEP でプログラムを実行するには、以下のランタイム ライセンスが必要です。

オペレーティング システムとして Windows CE を使用しているデバイス用の WinStudio ランタイム ライセンス、変数最大 1,500 個 (Mat. No. R911323620)

WinStudio 表示の追加



WinStudio 表示をプロジェクトに追加するためには、対応する表示デバイスがプロジェクトになければなりません。

"ドラッグ & ドロップ"を使用して、デバイス ライブラリからプロジェクト ツリーに VEP を追加し、以下のとおり設定します。

- デバイス タイプ: VEPxx (CE)
- アプリケーション タイプ: WinStudio アプリケーション

分散状態の機械での GAT

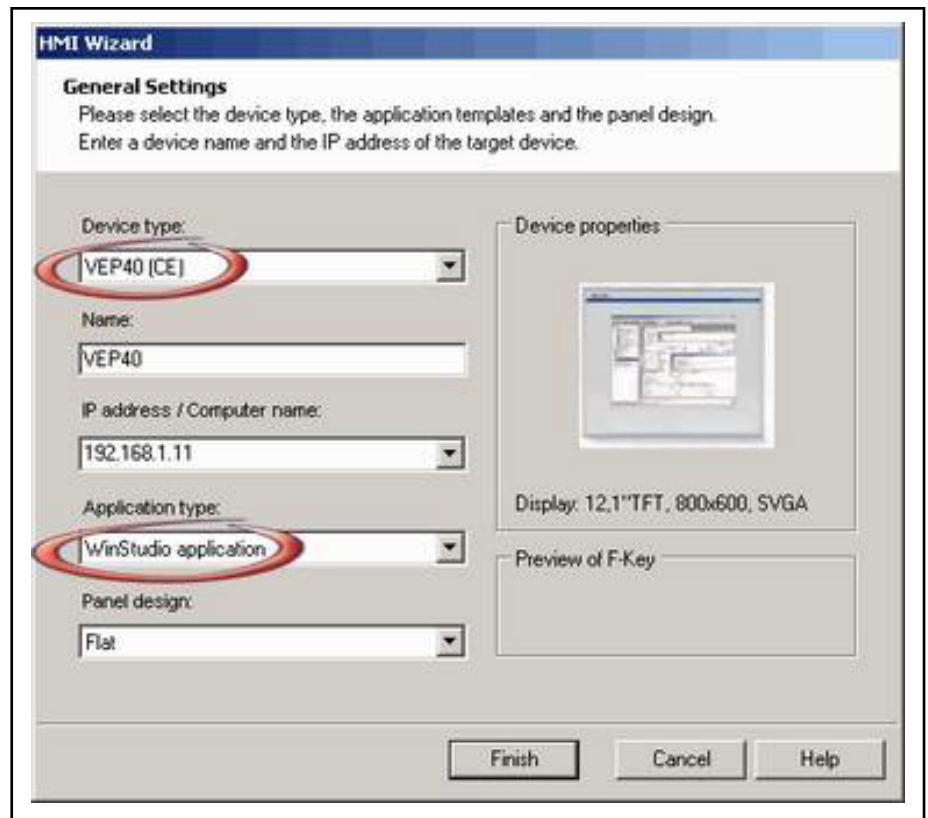


図 11-32: 表示デバイスの追加

対応する表示デバイスがある場合、下記のリンクとウィザードを使用して WinStudio 表示を追加します。

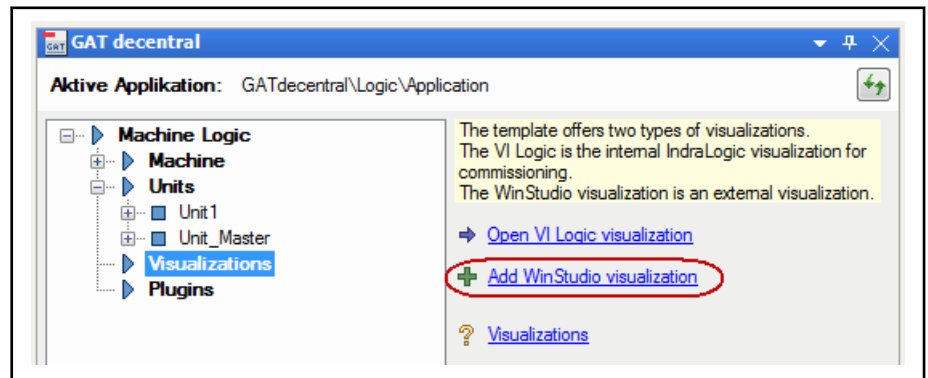


図 11-33: WinStudio 表示の追加

各設定に対応するプロジェクトの表示デバイスはすべて下記のウィザード ダイアログに含まれています。

分散状態の機械での GAT

対応する表示デバイスを選択した後で、**Apply** (適用) をクリックして WinStudio 表示を追加します。

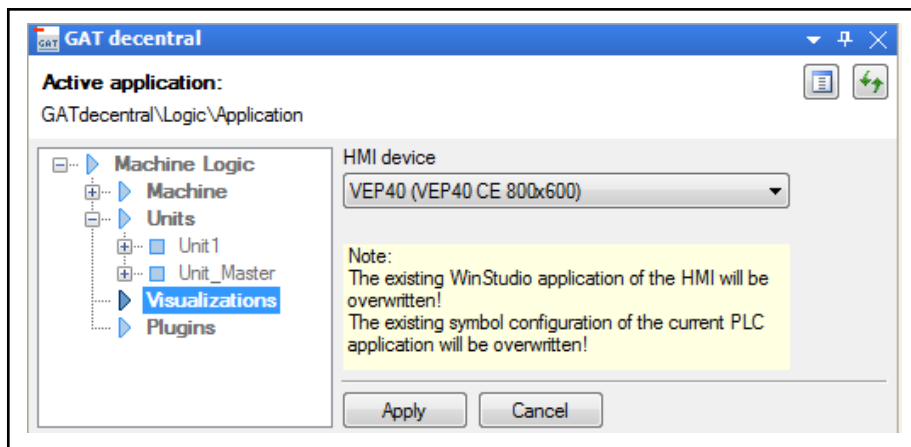


図 11-34: 表示デバイスの選択 – WinStudio 表示の挿入



WinStudio 表示が追加されると、コントロールの記号設定とともに表示デバイスの既存の表示データが上書きされます。

表示が追加された後で、表示データを編集できます。

コントロール アドレスは BR-WS ドライバの "Properties" (プロパティ) ダイアログで設定されます。

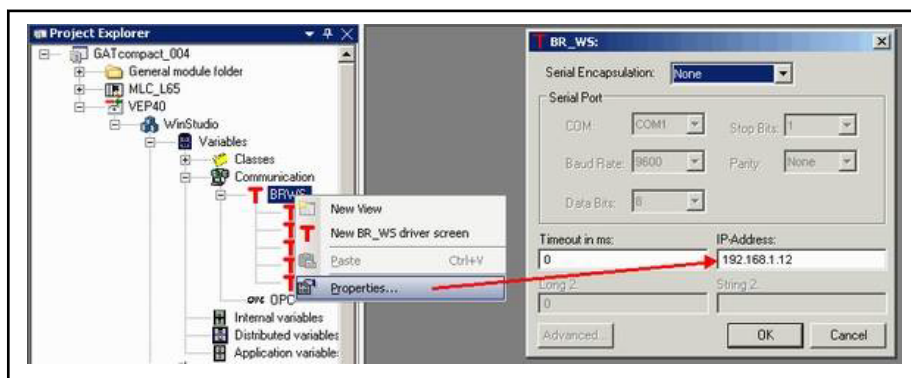


図 11-35: コントロールの IP アドレス設定

ここで、WinStudio ノードのコンテキスト メニューにある  "Start application" (アプリケーションを起動) を使用して WinStudio 表示のエミュレーションを開始します。

ダイアログ構造

操作画面は以下の 3 つのセクションに分かれています。

- ステータス情報があるヘッダー
- メニュー関係の表示と機能キーがあるメイン画面
- ユーザー マップの切替を行うためのキーがあるフッター

分散状態の機械での GAT

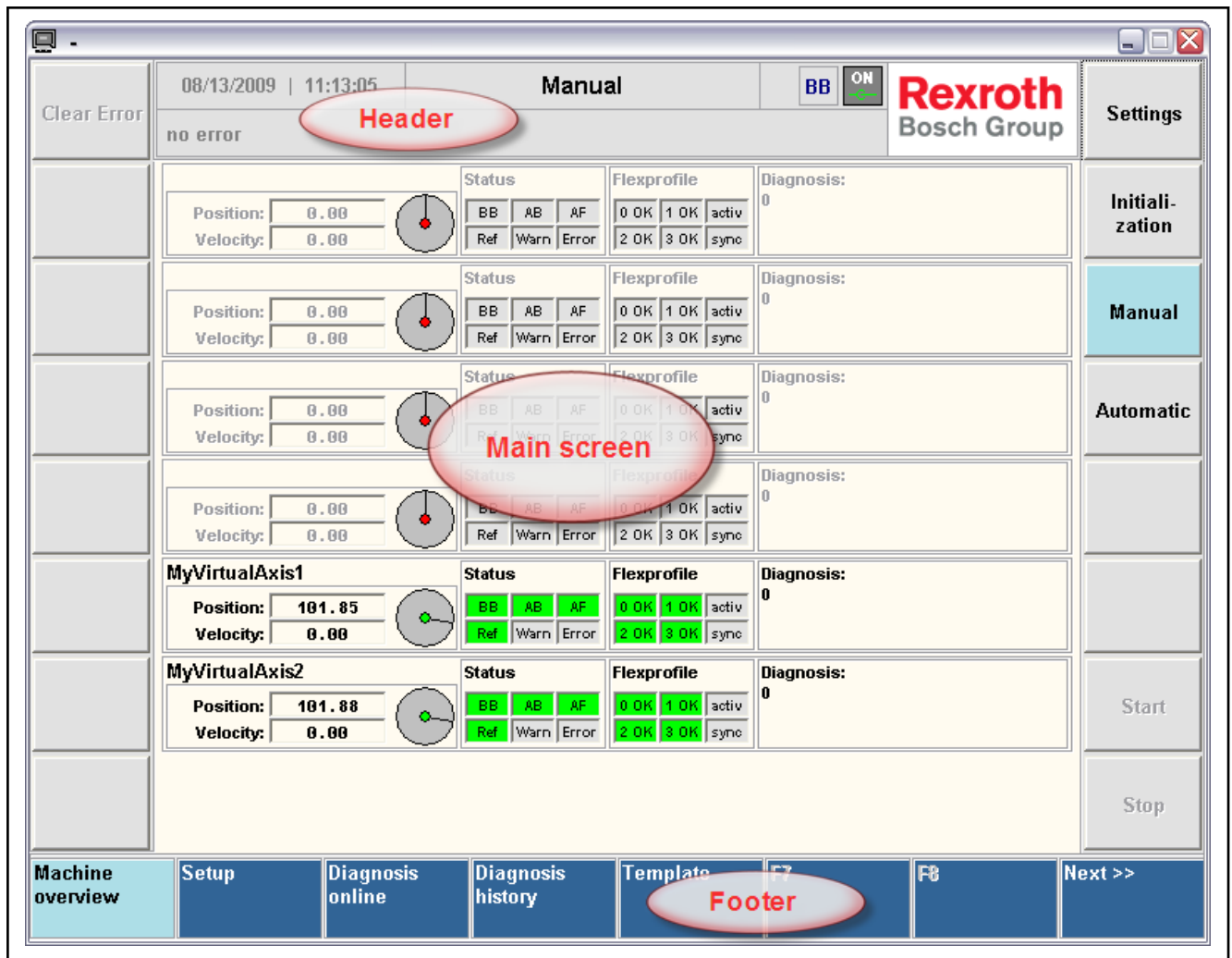


図 11-36: WinStudio 表示

ヘッダーには以下が含まれています：

- 運転モードの表示
- "エラーを削除"したり "Settings" (設定) ダイアログ (言語の切替、数値表示の小数点位置の設定などのため) を開いたりするためのグローバル機能キー
- 現在の機械診断の表示
- 現在の sercos フェーズの表示 (アクティブなフェーズ切替を含む)
- 通信ステータスの表示

メイン画面には、操作画面 (軸の情報、運転モードの切替など) に関連する表示と機能キーが含まれています。

フッターには、ユーザー マップを切り替えるためのキーが含まれています。

表示の拡張

操作画面の追加

分散状態の機械での GAT

表示を拡張するには、既存の"テンプレート"画面をコピーして新しい名前を付けて挿入することにより、新しい操作画面を作成できます。

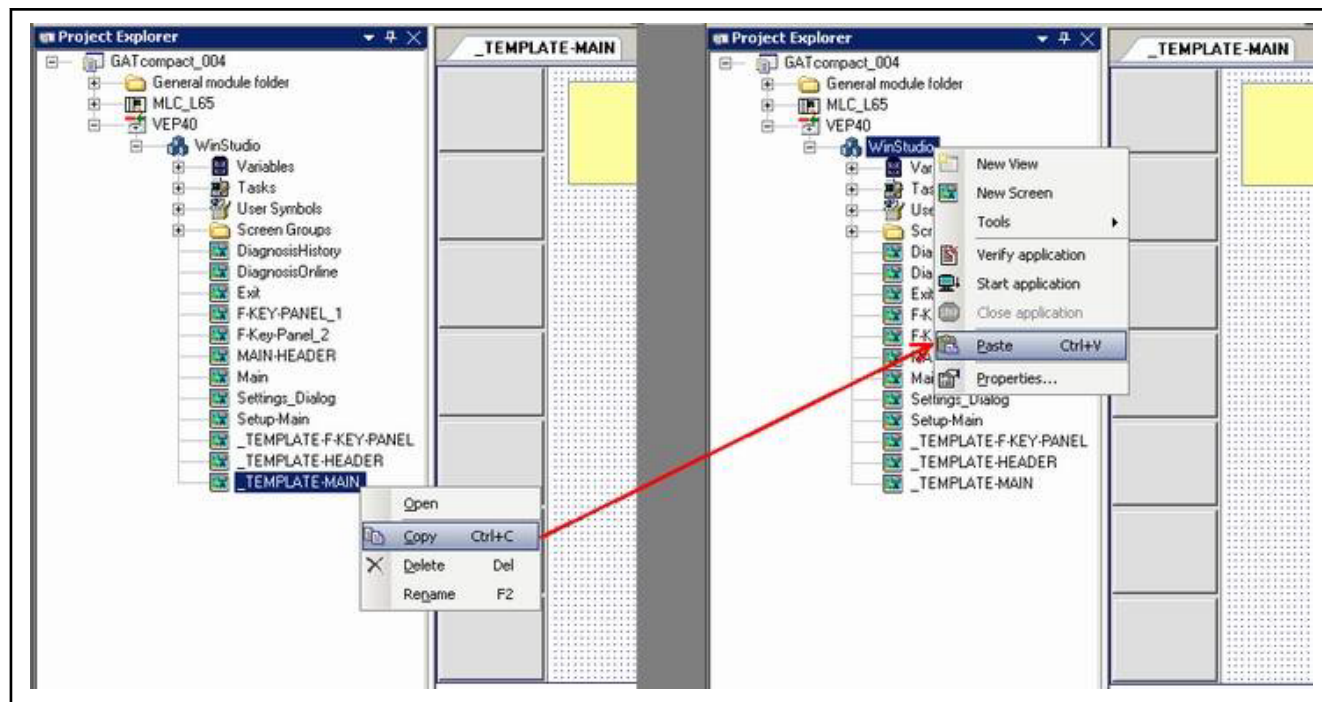


図 11-37: 操作画面の複製

軸データを表示するための要素は、"コピー" & "ペースト"で複製できる、グループ化された個別のオブジェクトです。異なる軸のために挿入されたオブジェクトを再設定するには、各オブジェクトの「置換」機能を使用して軸配列インデックスを適合させます (下の図を参照)。



図 11-38: 挿入されたオブジェクトの再設定

分散状態の機械での GAT

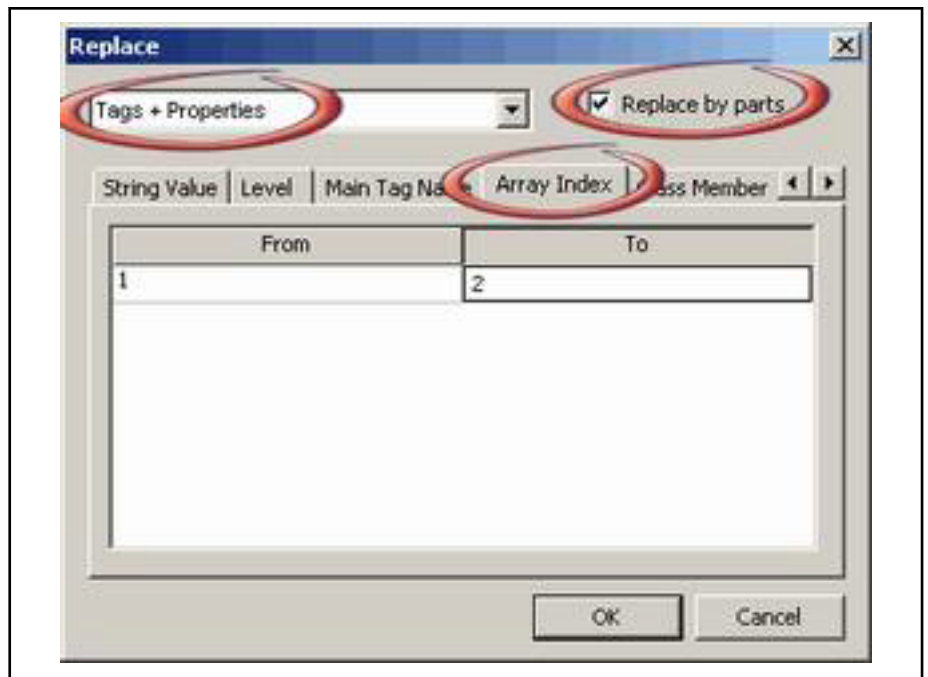


図 11-39: 挿入されたオブジェクトの再設定

フッターの拡張

ユーザー マップを切り替えるために使用されるキーは、パラメータ化が可能なグラフィック オブジェクト (リンクされた記号) として保存されます。キーには以下の転送パラメータがあります。

- Caption -> キー ラベル
- Disable -> キー機能をブロックするための入力 (グレー表示のラベル)
- ScreenName -> 操作画面を開くための画面名 (または画面グループ)
- ScreenNumber -> 画面に割り当てられる番号 (それぞれのキーを強調表示するため)

軸の最大数を大きくする

インポート後、表示は 16 軸用に設計されています。これは、IndraControl L25 の軸の最大数に対応しています。さらに軸が必要な場合は、表示の対応するドライバ シートを呼び出す必要があります。

分散状態の機械での GAT

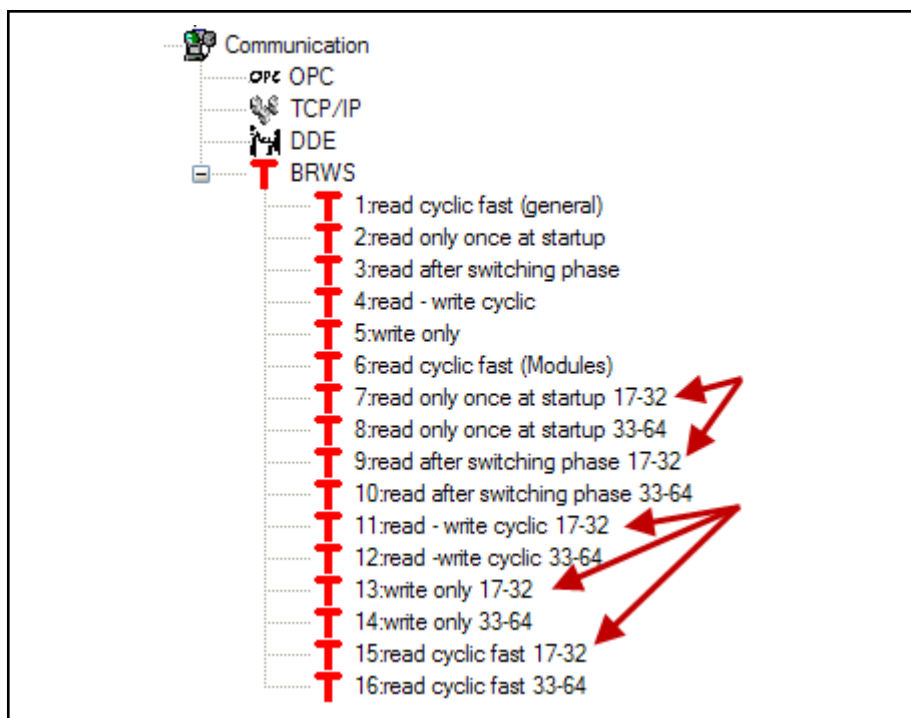


図 11-40: ドライバシートの概要

設定は、各ドライバシートをダブルクリックして開くことができます。アクティブではないドライバシートの場合、読み書きトリガー変数が空のため、アクティブなドライバシートからエントリを追加する必要があります。

ドライバシート "read only once at startup 17-32" に基づいた例：

ここにあるドライバシート "read only once at startup" からのエントリ "stCom.TrigOnce" をドライバシート "read only once at startup 17-32" にトリガー変数としてコピーする必要があります。

Description: read only once at startup ☐ Increase priority

Read Trigger: stCom.TrigOnce

Enable Read when Idle:

Read Completed:

Read Status:

Write Trigger:

Enable Write on Tag Change:

Write Completed:

Write Status:

Station:

Header: Sheet_2

☐ Mir:

☐ Max:

図 11-41: アクティブなドライバシートのドライバシート設定

Description: read only once at startup 17-32 ☐ Increase priority

Read Trigger:

Enable Read when Idle:

Read Completed:

Read Status:

Write Trigger:

Enable Write on Tag Change:

Write Completed:

Write Status:

Station:

Header: Sheet_7

☐ Mir:

☐ Max:

図 11-42: 32 軸に拡張するためのドライバシート設定

分散状態の機械での GAT

すべてのドライバシートで一致するように各エントリを調整する必要があります。ドライバシート後の番号は、これらのドライバシートが作成された対象の軸を示します。

通信接続

表示とコントロール間の通信は、BR-WS ドライバを使用して記号的に行われます。これを行うには、表示する PLC 変数の記号エントリを生成します (44 ページ "記号設定" 章を参照)

予め定義済みの WinStudio 表示に必要な記号設定には、以下のエントリが含まれます。

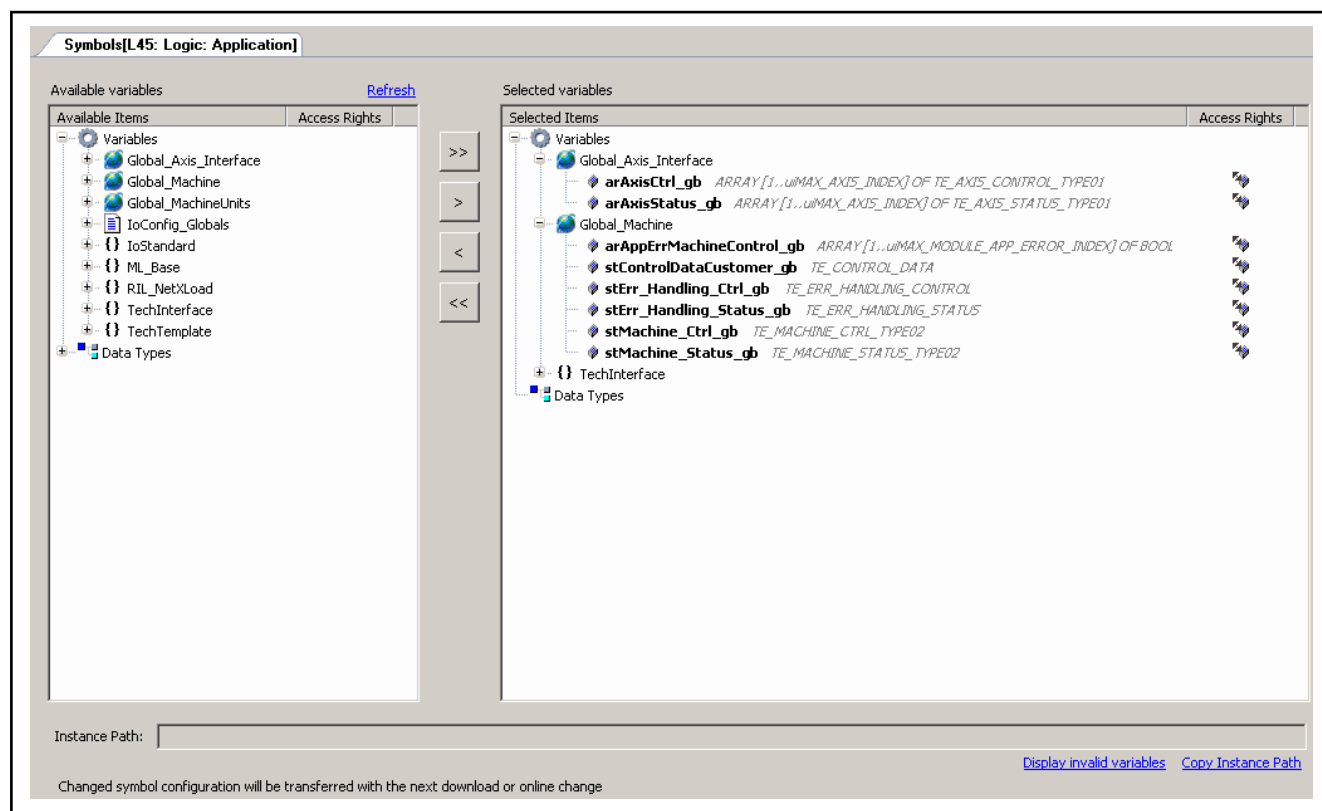


図 11-43: 予め定義済みの WinStudio 表示の記号設定



ライブラリから取得したグローバル変数リストの変数の記号エントリは、各ライブラリのバージョンが変更された後で記号設定に挿入しなおす必要があります。

コントロールの IP アドレス設定 :

コントロール アドレスは BR-WS ドライバの "プロパティ" ダイアログで設定されます。

分散状態の機械での GAT

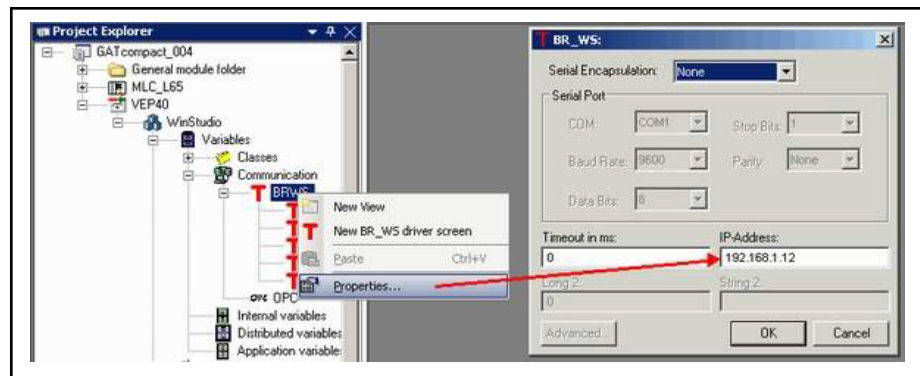


図 11-44: コントロールの IP アドレス設定

通信を構成するために、変数がさまざまなドライバシートに配置されています。

1. 高速で周期的に読み取る：-> これらの変数の値は短い間隔で周期的に読み取られます (診断番号、軸の位置など)
2. 1 回のみ読み取る：-> これらの変数は表示が開始された後で 1 回のみ読み取られます (軸の名前など)
3. フェーズの切替後に読み取る：これらの変数はフェーズの切替後に 1 回のみ読み取られます (軸がアクティブになった場合など)
4. 周期的に読み書き：-> これらの変数の値は、表示により読み書きの両方が行われます (機械の速度、Setup (セットアップ) モードの軸の速度など)
5. 書き込みのみ：これらの変数は、表示を通じて読み取りのみが行われます (エラーの削除の信号を発信する...)

ドライバシートを拡張するには、BR-WS ブラウザでオフライン記号ファイル (*.XML) を検索し、対応する変数を選択できます。

分散状態の機械での GAT

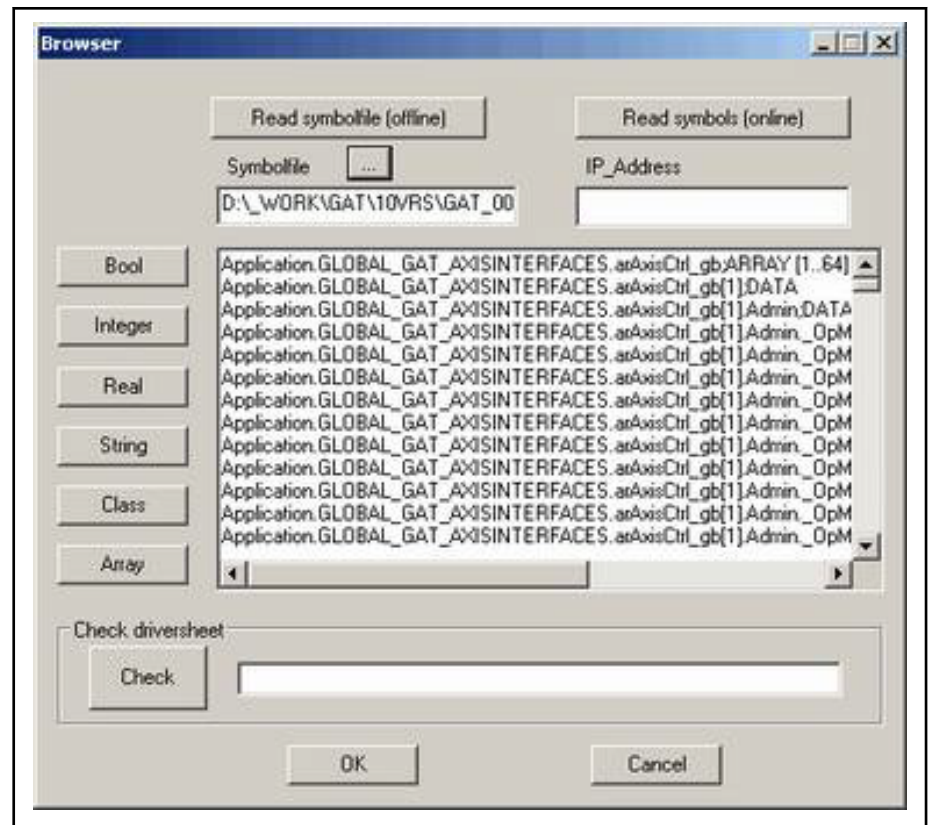


図 11-45: ドライバシートの拡張

複数言語機能

表示の複数言語機能は、IndraWorks への翻訳機能の組み込みにより実現されました。

コントロールおよびドライブの診断の翻訳データ、ならびに表示テキストは、すでに準備されています。翻訳データは IndraWorks で変更や修正が可能です。

ドライブ診断はドイツ語、英語、フランス語、スペイン語、トルコ語で含まれています。

他のすべてのテキストはドイツ語と英語です。

11.12 分散状態の機械での GAT のインターフェース

11.12.1 一般情報

GAT インターフェース データは変数リストで利用可能です。データの一部はライブラリで利用可能になっています。

11.12.2 機械のマシン ハンドラー データ (TE_MachineControl)

TE_MachineControl 入力データ

TE_MachineControl の入力データは、タイプ `TE_MACHINE_CTRL_TYPE02` の構造 `stMachineCtrl_gb` (変数リスト "Global_Machine") にまとめられています。

分散状態の機械での GAT

下の表は、入力構造の要素について説明したものです。

I/O タイプ	名称	データの種類	説明
I	Enable	BOOL	処理は有効
I	ClearError	BOOL	正のエッジがローカル エラー クリアを実行

I 入力 (読み書きアクセス)

表 11-3:

TE_MachineControl output data "TE_MachineControl" の出力データは、タイプ TE_MACHINE_STATUS_TYPE02 の構造 stMachineStatus_gb (変数リスト "Global_Machine") にまとめられています。

下の表は、出力構造の要素を説明したものです。

I/O タイプ	名称	データの種類	説明
O	InOperation	BOOL	処理はアクティブです
O	OpMode	TE_OPMODES_PREDEFIN ED_TYPE02	機械の現在の運転モードの確認
O	エラー	BOOL	ローカル エラーあり
O	ErrorID	ERROR_CODE	DiagNo = 16#F0FB 0000 または 16#F0AA xxxx の場合の詳細情報
O	ErrorIdent	ERROR_STRUCT	DiagNo = 16#F0FB 0000 または 16#F0AA xxxx の場合の詳細情報
O	DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/ IndraMotion MLC 13VRS 診断) に基づいた番号 に以下の拡張子が付いたもの: DiagNo = 16#F0AE xxxx: アプリケーション エ ラー (例: 安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー DiagNo = 16#F0B1...B9 xxxx: プラグイン追加 診断エラー (例: PROFIBUS 診断プラグインを 使用することによる)
O	DiagMessage	STRING(60)	コントロール/軸の診断テキスト

O 出力 (読み取りアクセス)

表 11-4:

11.12.3 集中エラー処理データ (TE_ErrorHandling)

TE_ErrorHandling 入力データ "TE_MachineControl" の入力データは、タイプ TE_ERR_HANDLING_CONTROL の構造 stErr_Handling_Ctrl_gb (変数リスト "Global_Machine") にまとめられています。

分散状態の機械での GAT

下の表は、入力構造の要素について説明したものです。

I/O タイプ	名称	データの種類	説明
I	Enable	BOOL	処理は有効
I	ClearAllError	BOOL	正のエッジが グローバル エラー クリアを実行
I	LogError	BOOL	TRUE = GAT 固有エラー (アプリケーション、機能ブロック、GAT およびプラグイン エラー) はコントロールのエラー/診断メモリにも保存されています。

I 入力 (読み書きアクセス)

表 11-5:

TE_ErrorHandling 出力データ "TE_MachineControl" の出力データは、タイプ **TE_ERR_HANDLING_CONTROL** の構造 "stErr_Handling_Status_gb" (変数リスト **Global_Machine**) にまとめられています。

下の表は、出力構造の要素を説明したものです。

I/O タイプ	名称	データの種類	説明
O	InOperation	BOOL	処理はアクティブです
O	エラー	BOOL	グローバル エラーあり
O	ErrorID	ERROR_CODE	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	ErrorIdent	ERROR_STRUCT	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの: DiagNo = 16#F0AE xxxx: アプリケーション エラー (例: 安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー DiagNo = 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例: PROFIBUS 診断プラグインを使用することによる)
O	DiagMessage	STRING(60)	コントロール/軸の診断テキスト

O 出力 (読み取りアクセス)

表 11-6:

11.12.4 ユニット データ

ユニット入力データ 各ユニットには、運転モード ハンドラー (TE_MachineControl) およびエラー ハンドラー (TE_ErrorHandling) により指定される入力としてコントロール データ構造 (Ctrl) があります。要素 ClearError、ExtError、および ExtErrDiagNo は、エラー ハンドラー (TE_ErrorHandling) により指定されます。他のすべての要素は、運転モード ハンドラー (TE_MachineControl) により指定されます。

分散状態の機械での GAT

ユニット TE_UNIT_CONTROL の入力データ構造の要素は、下の表に挙げられています。

I/O タイプ	名称	データの種類	説明
I	Enable	BOOL	処理は有効
I	ClearError	BOOL	正のエッジに基づいたエラー確認 (from TE_ErrorHandling より)
I	OpMode	TE_OPMODES_PREDEFIN ED_TYPE02	命令された運転モード
I	DisableErrorReaction	BOOL	エラーリアクションを無効化
I	EnableErrorReactionInit	BOOL	初期化中のエラーリアクションを有効化
I	OpModeCoupled	BOOL	TRUE = 結合ユニット FALSE = 非結合ユニット
I	ExtError	BOOL	外部エラー (TE_ErrorHandling より)
I	ExtErrDiagNo	DWORD	外部エラー番号 (TE_ErrorHandling より)

I 入力 (読み書きアクセス)

表 11-7:

ユニット出力データ

各ユニットには、運転モード ハンドラー (TE_MachineControl) およびエラー ハンドラー (TE_ErrorHandling) により評価される出力としてステータス データ構造 (Status) があります。すべての Error... 要素および Diag... 要素は、エラー ハンドラー (TE_ErrorHandling) により評価されます。他のすべての要素は、運転モード ハンドラー (TE_MachineControl) により評価されます。

ユニット TE_UNIT_STATUS の出力データ構造の要素は、下の表に挙げられています。

I/O タイプ	名称	データの種類	説明
O	InOperation	BOOL	処理はアクティブです
O	OpMode	TE_OPMODES_PREDEFIN ED_TYPE02	現在の運転モードの確認
O	エラー	BOOL	ローカル エラー
O	ErrorID	ERROR_CODE	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報
O	ErrorIdent	ERROR_STRUCT	DiagNo = 16#F0FB 0000 / 16#F0AA xxxx の場合の詳細情報

分散状態の機械での GAT

I/O タイプ	名称	データの種類	説明
O	DiagNo	DWORD	資料 "Rexroth IndraLogic XLC/IndraMotion MLC 13VRS Diagnostics" (Rexroth IndraLogic XLC/IndraMotion MLC 13VRS 診断) に基づいた番号に以下の拡張子が付いたもの : DiagNo = 16#F0AE xxxx: アプリケーション エラー (例 : 安全扉が開いている) DiagNo = 16#F0FB 0000: FB エラー DiagNo = 16#F0AA xxxx: GAT 内部エラー DiagNo = 16#F0B1...B9 xxxx: プラグイン追加診断エラー (例 : PROFIBUS 診断プラグインを使用することによる)
O	DiagMessage	STRING(60)	診断テキスト

O 出力 (読み取りアクセス)

表 11-8:

11.12.5 軸データ

I/O タイプ	名称	データの種類	説明
I	arAxisCtrl_gb (Global_Axis_Interfaces)	ARRAY OF TE_AXIS_CONTROL_TYPE 01	軸用のコマンド
O	arAxisStatus_gb (Global_Axis_Interfaces)	ARRAY OF TE_AXIS_STATUS_TYPE01	軸からのフィードバック
I/O	AxisData (ML_Base ライブラリからの GVL_Base)	ARRAY OF ML_AXISDATA_SM	軸の入力と出力のあるファームウェア構造

I 入力 (読み書きアクセス)

O 出力 (読み取りアクセス)

I/O 入力と出力のある構造



arAxisCtrl_gb の要素はユニットにより上書きされるため、外部的に使用することはできません。

11.12.6 コントロール データ

I/O タイプ	名称	データの種類	説明
I	ImcCtrl (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_CONTROL_TYPE01	コントロール用のコマンド
O	ImcStatus (Global_ImcInterface from the ML_TechInterface ライブラリ)	MB_IMC_STATUS_TYPE01	コントロールからのフィードバック

I 入力 (読み書きアクセス)

O 出力 (読み取りアクセス)



ImcCtrl_gb の要素は "TE_MachineControl" に記述されているため、外部的に使用することはできません。

分散状態の機械での GAT

11.13 操作手順：一般的なユーザー アクティビティ

11.13.1 軸データへのアクセス

軸データへのアクセスについて、すべてのユニットで、ユニットに割り当てられたすべての軸の一元的なアクセスポイントとして **Axes** (軸) 構造が定義されます。軸のすべての関連データは、Axes (軸) 構造を使用してアドレスを指定できます。

この構造は、軸の名前を用いて軸データにアクセスできるように設計されています。これにより、読み取りが容易なプログラムコードが生成されます。

"Axes" (軸) の項目では以下のデータが利用可能です。

- Axes.<Name>.AxisData => 軸データ構造：ファームウェアのステータスデータおよび軸データ コンテナへのアクセス
- Axes.<Name>.Ctrl => AxisInterface の制御構造
- Axes.<Name>.Status => AxisInterface のステータス構造
- Axes.<Name>.DV => 直接変数 (機能変数) へのアクセス
- Axes.<Name>.AxisRef => ファンクションブロック呼出の軸参照構造 (MB_ReadParameter など)

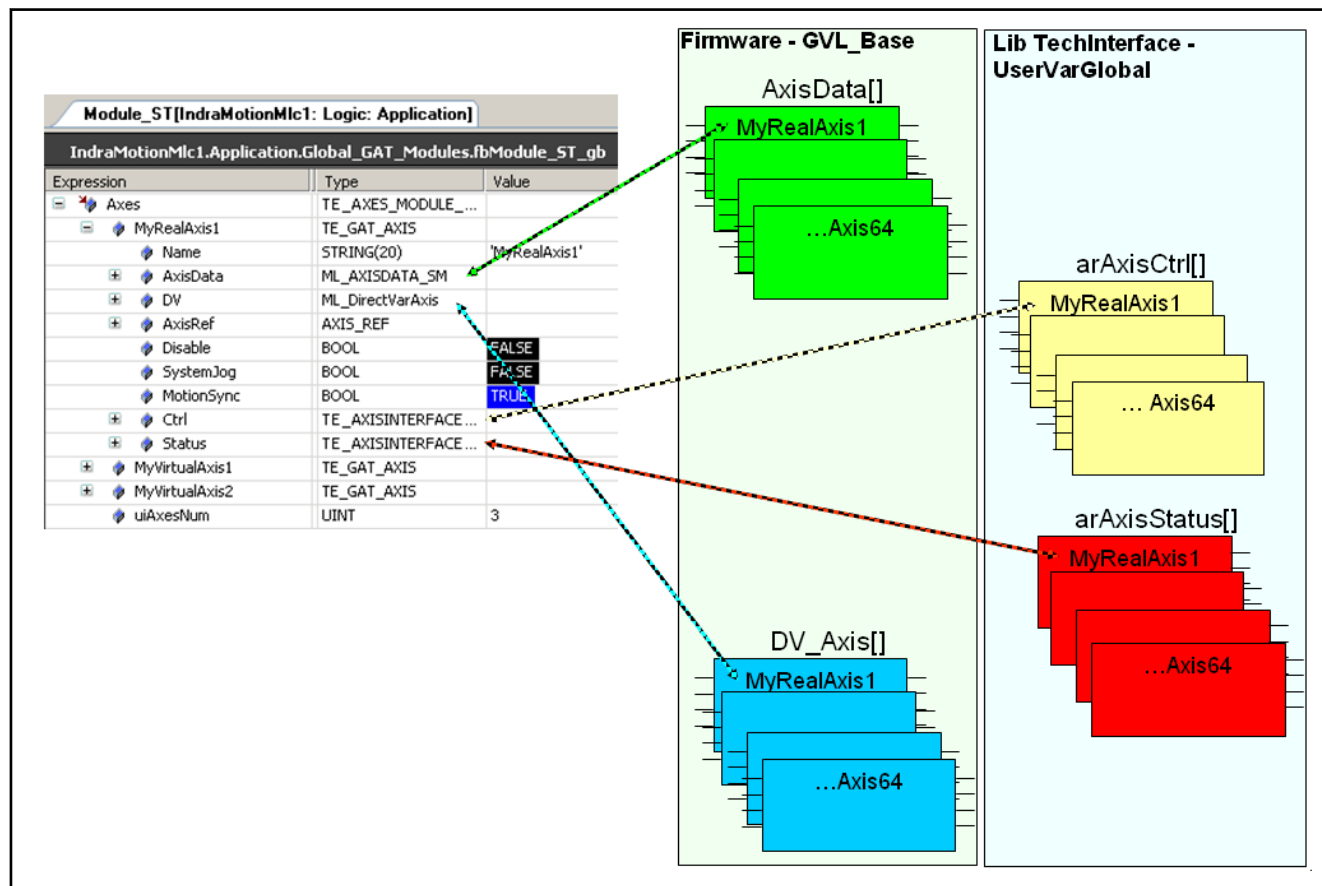


図 11-46: 概要/軸データへのアクセス

11.13.2 軸の追加

軸を追加するためには、以下の手順が必要です。

- IndraWorks プロジェクトに軸を追加する (sercos ノードにドラッグする)

分散状態の機械での GAT

- "SystemOverview_All_Axis" 表示を使用する場合：新しい軸のフレームを追加し、プロパティで新しい軸への参照を設定する (221 ページ "起動中の支援のための内部表示" 8.8.1 章 も参照)
- GAT ウィザードを使用してユニット軸を追加する (下の図を参照)

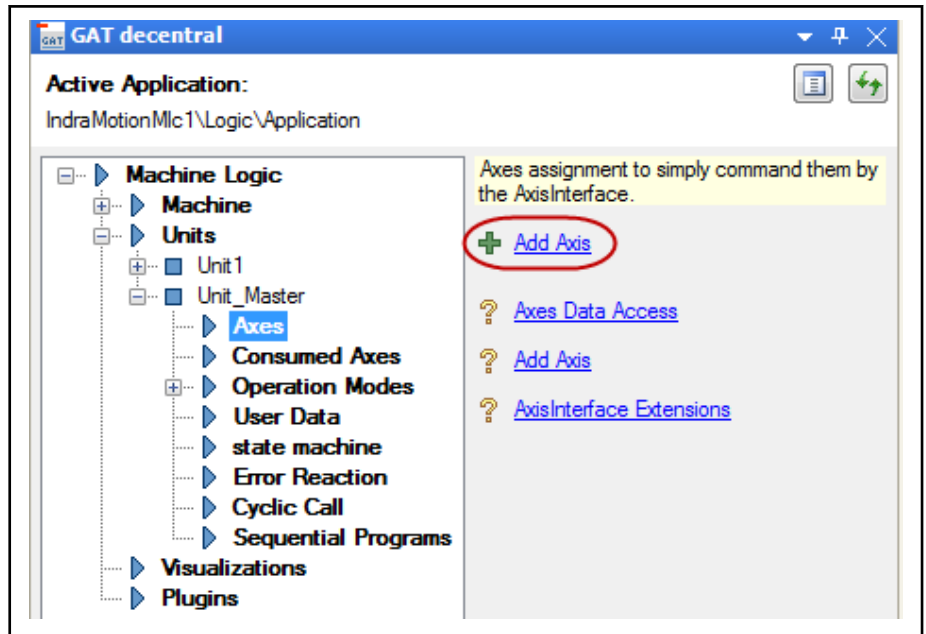


図 11-47: GAT ウィザードを使用した軸の追加

11.13.3 軸の削除/名前変更



ウィザードによって変更されるのは、軸宣言にある名前のみです。軸がすでに実装または内部表示で使用されている場合、新しい軸の名前は、"検索・置換"を使用して、これらの位置で変更する必要があります。

軸の削除/名前変更を行うためには、以下の手順が必要です。

- ndraWorks プロジェクトから軸を削除する/ndraWorks プロジェクトで名前を変更する
- "SystemOverview_All_Axis" 表示を使用する場合：削除した軸のフレームを削除し、プロパティで軸への参照を変更する
- GAT ウィザードを使用した軸の削除/名前変更

11.13.4 プライマリ運転モードの追加

GAT ウィザードを使用すると、機械へのプライマリ運転モードの追加が容易になります。また、ウィザードは運転モードの削除や名前変更にも使用できます。

分散状態の機械での GAT

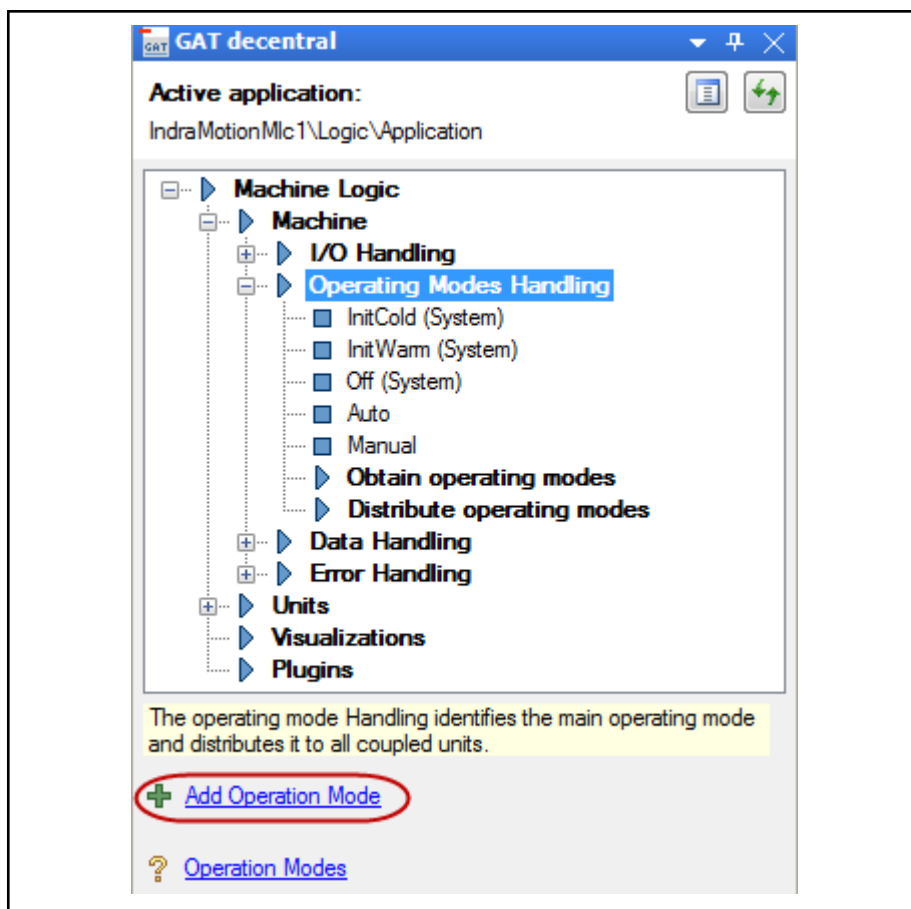


図 11-48: GAT ウィザードによる運転モードの追加

11.13.5 新しいユニットの追加

GAT ウィザードを使用すると、機械へのユニットの追加が容易になります。また、ウィザードはユニットの削除や名前変更にも使用できます。

新しい結合ユニットで、既存のすべてのプライマリ運転モードについて、運転モード SFC に対応する分岐が作成されます。

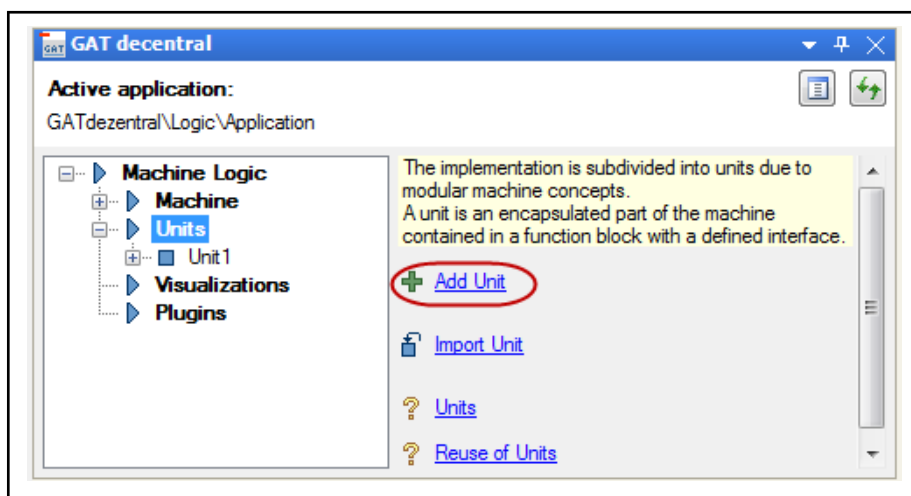


図 11-49: GAT ウィザードによるユニットの追加

11.13.6 ユニットのインスタンスの追加

GAT ウィザードを使用すると、既存のユニットの新しいインスタンスを機械に簡単に追加できます。

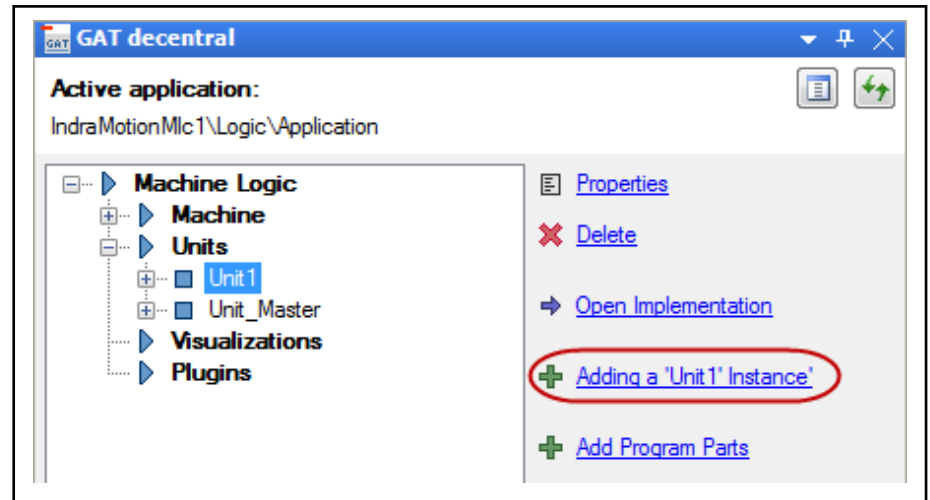


図 11-50: GAT ウィザードによるインスタンスの追加

インターフェースは同じですが、他のグローバル軸 ("Axes" への) および I/O データ ("UserData" への) を割り当てる必要があります。

11.13.7 ユニットの無効化

ファンクションブロック "TE_MachineControl" のアクション **actInitColdStart** で、命令 "mEnableUnits(Enable := TRUE);" は、Enable 入力を使用してすべてのユニットを有効にするために使用されます (下の図の強調表示 1 を参照)。

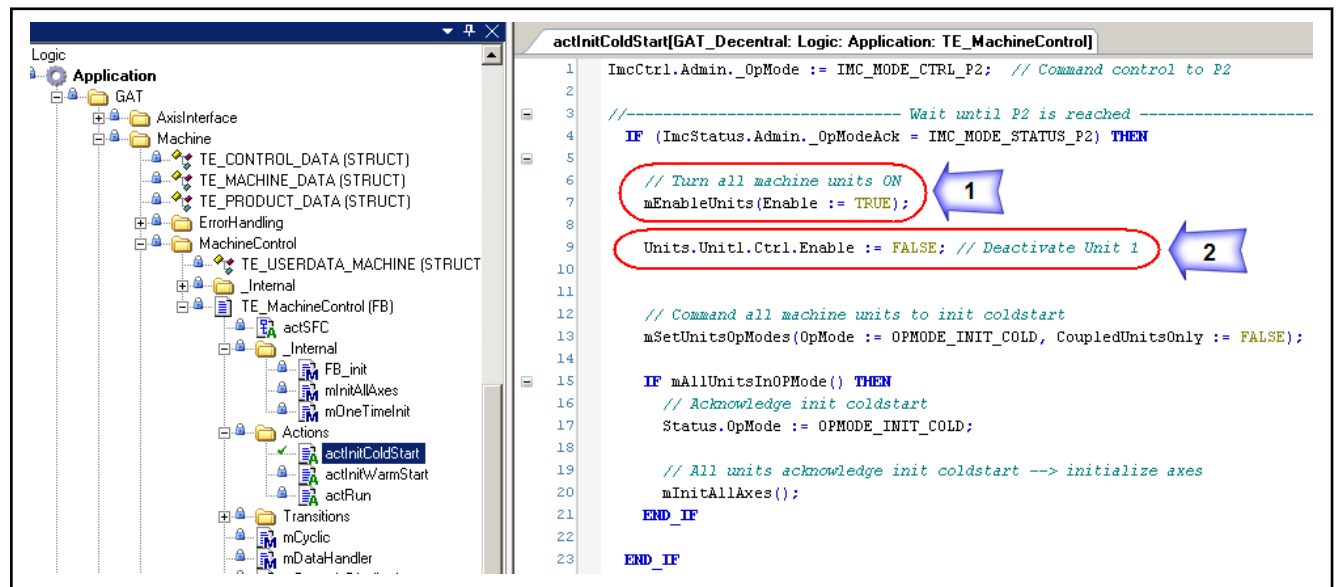


図 11-51: 例 "Unit1 を無効化する"

上記の説明に従って、命令 "Units.<UnitName>.Ctrl.Enable := FALSE" を使用し、目的の方法で個々のユニットを無効にすることが可能です(上の図の強調表示 2 を参照)。無効化されたユニットのプログラム コードは、処理から除外されます。ユニットはいつでも無効にできます。ステータス出力 "InOperation = FALSE" は、このユニットが無効になっていることを示します。

分散状態の機械での GAT

11.13.8 ユニットの再利用

GAT は、開発されたユニットを、GAT を使用して構築された他のプロジェクトでも再利用できるように設計されています。

これを行うには、任意のユニットのフォルダを選択し、コンテキストメニューを使用して関連データをエクスポートします (下の図を参照)。

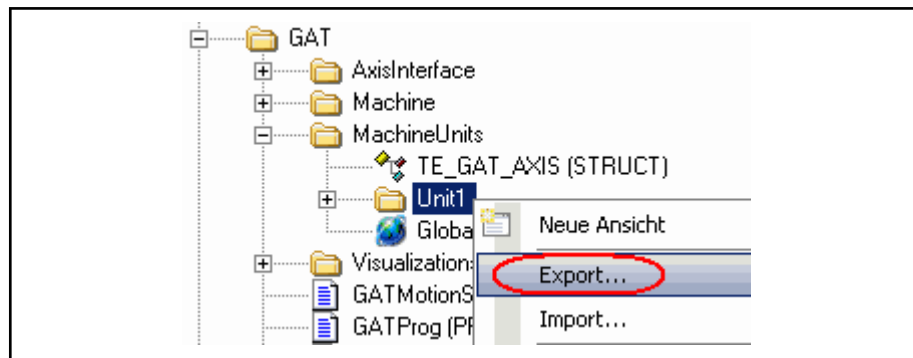


図 11-52: ユニットのエクスポート

その後のダイアログでエクスポートの要素をすべて選択します。

エクスポート中に生成されたファイルを新しいプロジェクトにインポートします。これを行うには、"MachineUnits" フォルダを強調表示し、関連するコンテキストメニューで "Import" (インポート) 機能を使用します (下の図を参照)。

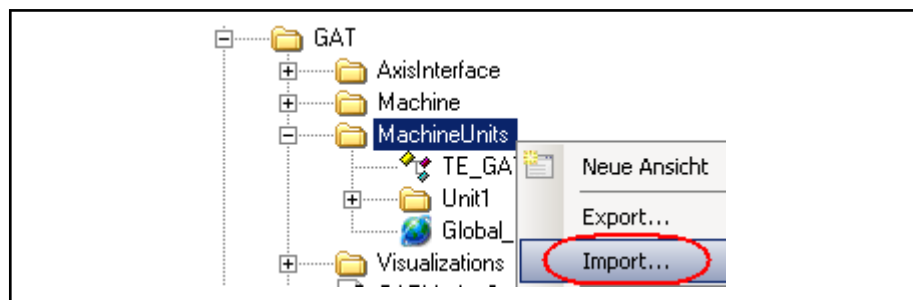


図 11-53: ユニットのインポート

データ構造 **TE_MACHINE_UNITS** で、インポートされたユニットは要素として受け入れられる必要があり、要素 **uiMaxUnitNum** の値を 1 大きくする必要があります。

分散状態の機械での GAT

下の図は、"MyImportUnit" という名前のユニットの例を示したものです。

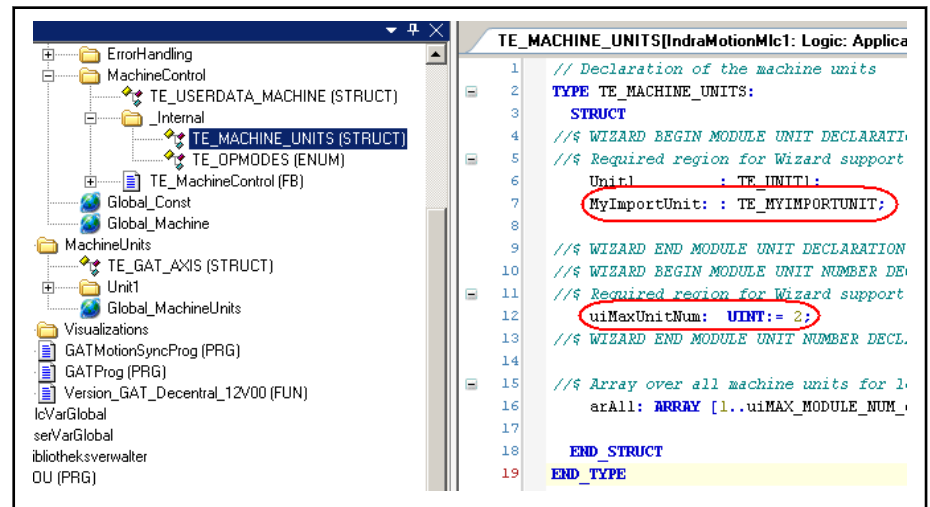


図 11-54: ユニット インポート後のデータ構造 "TE_MACHINE_UNITS" の調整例

インポートされたユニット インスタンスの呼出を、プログラム "GATProg" または "GATMotionSyncProg" に追加する必要があります。

新しいインスタンスのユニット参照は、FB TE_MachineControl のメソッド "mOneTimeInit" に追加します。結合ユニット (Units.Unit1.Ctrl.OpModeCoupled := TRUE; // Coupled unit) および非結合ユニット (Units.Unit1.Ctrl.OpModeCoupled := FALSE; // Uncoupled unit) の割当も、このエリアで入力する必要があります。

また、UserData 構造の要素の参照も、このメソッド (Units.<UnitName>.UserData.stControlDataActive REF=stControlDataActive_gb;) の ELSE 分岐に追加する必要があります。

新しいインスタンスのユニット参照は、FB "TE_ErrorHandling" のメソッド "mOneTimeInit" に追加します。

非結合ユニットの場合は、非結合ユニット用の部分をメソッド "mOpModeDistribution" に追加する必要があります。



インポートの後で、結合ユニットをチェックし、ユニットの SFC に機械のプライマリ運転モードもすべて含まれていることを確認します。含まれていない場合、欠けている運転モードを SFC 分岐に追加する必要があります。

11.13.9 ユニットでのグローバル変数とアドレス変数の使用

ユニットはアプリケーションで複数回使用でき (インスタンス化)、また別のアプリケーションで再利用できるため、ユニット内ではグローバル変数またはアドレス変数への直接アクセスがない場合があります。

ユニットで使用されるすべてのグローバル変数とアドレス変数は、ユニットデータ コンテナ "UserData" でインターフェース変数として宣言する必要があります。ユニット内の要素は、UserData.<Element> を使用してアクセスされます。

11.13.10 AxisInterface の拡張

GAT では、AxisInterface 構造の拡張が可能です。追加的な下位構造を追加したり、既存の下位構造を拡張したりできます。

分散状態の機械での GAT

例として、以下の構造要素がユーザー拡張として *AxisCtrl* 構造にプログラミングされています。

- SetupMode: *AxisCtrl* 構造の追加的な下位構造
 - Enable: Setup (セットアップ) モードを有効にする
 - JogPlus: ジョグ +
 - JogMinus: ジョグ -
 - Vel: ジョグ/参照移動速度
 - Accel: ジョグ/参照移動減速/加速
 - Homing: 軸のタイプ/設定によって ModeHoming|SetAbsRef|ModesPosAbs を有効化
- SyncMode: *AxisCtrl* 構造内の既存の下位構造の拡張
 - AngleOffset: 表のフェーズ オフセット開始 -> この値によりパラメータ A-0-2730 を説明
 - PhaseShift: ギアの前のマスター軸に比例したオフセット (PhaseOffset: ギアの後のマスター軸に比例したオフセット)
 - PhaseShiftVel: PhaseShift を調整するための速度
 - PhaseShiftAcc: PhaseShift を調整するための加速
 - SyncVelocity: フェーズ シフト速度 -> この値によりパラメータ A-0-2790 を説明
 - SyncAcceleration: フェーズ シフト加速 -> この速度はパラメータ A-0-2791 に書き込まれる

例として、以下の構造要素がユーザー拡張として *AxisStatus* 構造にプログラミングされています。

- SetupMode: *AxisStatus* 構造内の追加的な下位構造
 - EnableAck: Setup (セットアップ) モードが有効
 - Homingack: ホーミングが完了
- SyncMode: *AxisStatus* 構造内の既存の下位構造の拡張
 - AngleOffsetDone: 表のフェーズ オフセット開始の転送 -> この値によりパラメータ A-0-2730 を説明
 - PhasingMasterDone: PhaseShift が完了
 - SyncVelocityDone: 同期速度の転送 -> パラメータ A-0-2790 は転送済み
 - SyncAccelerationDone: 同期速度の転送 -> パラメータ A-0-2791 は転送済み

これらの拡張のコードは、"GAT/AxisInterface" フォルダ内の "TE_AxisInterface()" ブロックのアクションにあります。関連する構造は、"GAT/AxisInterface/Type/Control" および "GAT/AxisInterface/Type/Status" フォルダにあります。

ユーザーは、ユーザー独自の拡張を追加できます (これを行うには [97 ページ "AxisInterface のユーザー拡張" 6.7 章](#) を参照)。

11.13.11 Robot Control

協調運動のために軸を制御するには、TechInterface に MB_AxisInterfaceType02 FB を使用する必要があります。このため、GAT^{compact} で以下を調整する必要があります。

- 構造 "TE_AXIS_CONTROL_Type01" の名前を "TE_AXIS_CONTROL_TYPE02" に変更し、タイプ

分散状態の機械での GAT

"MB_AXIS_COORDINATED" の構造変数 "CoordModeCtrl" によってこの構造を拡張する必要があります。

構造 *TE_AXIS_CONTROL_Type01* の *TE_AXIS_CONTROL_Type02* への名前変更:

```
//former type: TYPE TE_AXIS_CONTROL_TYPE01:
TYPE TE_AXIS_CONTROL_TYPE02: // rename for RobotControl
STRUCT
  Admin:      TE_AXIS_ADMINISTRATION; (* Administration of the axis | Verwaltung der Achse *)
  PosMode:    TE_AXIS_POSITIONING; (* operating mode positioning (absolute, rel. and additive)
    Positionierbetrieb (absolut, relativ und additiv) *)
  VelMode:    TE_AXIS_VELOCITY; (* operating mode velocity control | Geschwindigkeitsregelung *)
  SyncMode:   TE_AXIS_SYNCHRONISATION; (* operating mode synchronisation (phase, vel, and cam)
    Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvenscheibe) *)
  (* add your extensions below here, keep the elements above
    Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
  (* user extensions | Anwender Erweiterungen *)
  // Jog/Homing functionality | Tipp/Einrichtfunktionen
  SetupMode:  TE_AXIS_SETUP_MODE;

  CoordModeCtrl: MB_AXIS_COORDINATED; // inserted for RobotControl

END_STRUCT
END_TYPE
```

- "TE_AXIS_STATUS_Type01" の構造の名前を "TE_AXIS_STATUS_TYPE02" に変更します

構造 *TE_AXIS_STATUS_Type01* の *TE_AXIS_STATUS_TYPE02* への名前変更:

```
(***** Outputs of the following axis - from motion control to logic control *****)
// former type: TYPE TE_AXIS_STATUS_TYPE01:
TYPE TE_AXIS_STATUS_TYPE02: // Rename for RobotControl
STRUCT
  (***** Outputs of the following axis - from motion control to logic control *****)

  PtrAxisData: POINTER TO ML_AXISDATA_SM;
  (* Pointer to AxisData of this axis | Zeiger auf AxisData dieser Achse
    - Type .PtrAxisData^. for the intellisense menu | Eingabe .PtrAxisData^. fuer Auswahlmenue*)

  Admin:      TE_AXIS_ADMIN_STATUS; (* status of the axis | Achs-Status *)

  Diag:       TE_AXIS_DIAGNOSIS; (* axis diagnosis information, error, warning....
    Diagnose Informationen der Achse, Fehler, Warnung.... *)

  PosMode:    TE_AXIS_POS_STATUS; (* Status operating mode positioning (absolute, rel. and add.)
    Statusmeldungen Positionierbetrieb (absolut, relativ und additiv) *)

  VelMode:    TE_AXIS_VEL_STATUS; (* Status operating mode velocity control
    Statusmeldungen Geschwindigkeitsregelung (Freilauf) *)

  SyncMode:   TE_AXIS_SYNC_STATUS; (* Status operating mode synchronisation (phase, vel, and cam)
    Statusmeldungen Synchronbetriebsarten (Winkelsync., Geschwindigkeitssync. und Kurvensch.)*

  (* add your extensions below here, keep the elements above
    Erweiterungen hier einfüegen, Elemente oberhalb nicht veraendern *)
  (* user extensions | Anwender Erweiterungen *)
  (* Jog/Homing functionality | Tipp/Einrichtfunktionen *)
  SetupMode:  TE_AXIS_SETUP_MODE_STATUS;

  // AxisData for HMI (without data type "BIT")
  AxisData_HMI: TE_AXIS_DATA_STATUS;

END_STRUCT
END_TYPE
```

- 新しい FB "MB_AxisInterfaceType02" により "TE_Axisinterface" ブロックを拡張する必要があります。 "TE_AxisInterface EXTENDS MB_AxisInterfaceType02"

分散状態の機械での GAT

TE_Axisinterface の拡張 :

```
//former: FUNCTION_BLOCK TE_AxisInterface EXTENDS MB_AxisInterfaceType01
FUNCTION_BLOCK TE_AxisInterface EXTENDS MB_AxisInterfaceType02
(*#####
#####
```

```
General Header
```

```
-----
Shortdescription      : AxisInterface Example
```

- メソッド "mInitExtensions()" : ファンクションブロック "MB_AxisInterfaceType02" の追加的な "CoordModeCtrl" 入力を初期化する必要があります

CoordModeCtrl の初期化 :

```
SUPER^.mInitExtension(index);
```

```
IF NOT bFirstInit THEN
    // User: Assignment of user defined axis interface inputs
    SetupMode REF= arAxisCtrl_gb[index].SetupMode;
    // inserted for RobotControl
    CoordModeCtrl REF= arAxisCtrl_gb[index].CoordModeCtrl;
    // User: Assignment of user defined axis interface outputs
    SetupModeAck REF= arAxisStatus_gb[index].SetupMode;
    AxisData_HMI REF= arAxisStatus_gb[index].AxisData_HMI;

    // =====
    // Do not touch the following lines
    // =====

    // Set the references for the AxisInterface FB
    AdminCtrlExt REF= arAxisCtrl_gb[index].Admin;
```

- "arAxisCtrl_gb" と "arAxisStatus_gb" の両方の構造変数の宣言を、対応する type02 構造に変更する必要があります

構造変数の調整 :

```
VAR_GLOBAL
    // Axis interface: command structure: define as RETAIN, to keep the values
    // Note: on the L40.2 control, the RETAIN memory is slower than normal VAR, so define
    // this structure only as RETAIN when necessary
    // arAxisCtrl_gb: ARRAY [1..MAX_AXIS_NUM] OF MB_AXIS_CONTROL;

    // Axis interface: command structure
    //former: arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE01;
    arAxisCtrl_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_CONTROL_TYPE02;
END_VAR

VAR_GLOBAL
    //----- Machine status variables -----

    // Axis interface: status structure
    //former: arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE01;
    arAxisStatus_gb: ARRAY [1..uiMAX_AXIS_INDEX] OF TE_AXIS_STATUS_TYPE02;
    bRemoteOn_gb : BOOL := TRUE; // TRUE: Internal IL visu is active
                                // FALSE: PlcProg is active
END_VAR
```



構造 TE_AXIS_CONTROL_TYPE02、TE_AXIS_STATUS_TYPE02 はフォルダ "AxisInterface\Type" に含まれています。

構造変数 arAxisCtrl_gb、arAxisStatus_gb はグローバル変数リスト "UserVarGlobal" に含まれています。

キネマティクス インターフェース

軸インターフェースに基づいて、オプションのプラグインとしてキネマティクス インターフェースが利用可能です。キネマティクス インターフェースを使用すると、軸と同じ方法でキネマティクスを操作できます。この目的のために、GAT ウィザードを介してキネマティクス インターフェースをプラグインとして追加します。

分散状態の機械での GAT

キネマティクス インターフェースの資料は、プラグイン設定フォルダにあります。

以下の手順で、GAT ウィザードを介して設定フォルダへのリンクを参照します。

- GAT ウィザードでフォルダ "Plug-ins" を選択する
- ウィザード コントロール パネルに、リンク "Open plug.ins configuration folder" (プラグイン設定フォルダを開く) が表示される

11.13.12 GAT に基づいたユーザー独自のプログラム フレームワークの生成

GAT に基づいて開発されたアプリケーションは、さらに複雑な別のアプリケーションを開発するためのテンプレート (プログラム フレームワーク) としても使用できます。この方法により、ユーザーは、例えばさまざまな機種向けの独自のプログラム フレームワークを提供できます。しかし、このフレームワークは、より高度な開発 (追加的な運転モードやユニットなど) のために、標準の GAT とは異なります。その後、GAT ウィザードを使用してさらに高度なプログラム フレームワークに取り組むことが可能です。

GAT により開発されたアプリケーションに基づいたユーザー独自のテンプレートを生成するには、以下の手順を使用します。

1. すべてのチェック ボックスを選択済みの状態にして、GAT フォルダ全体をエクスポートします (*.iwx ファイルを生成する)。エクスポート ダイアログには "Description" (説明) 入力フィールドがあります。ここで入力したテキストは、GAT ウィザードのインポート機能で表示できます。

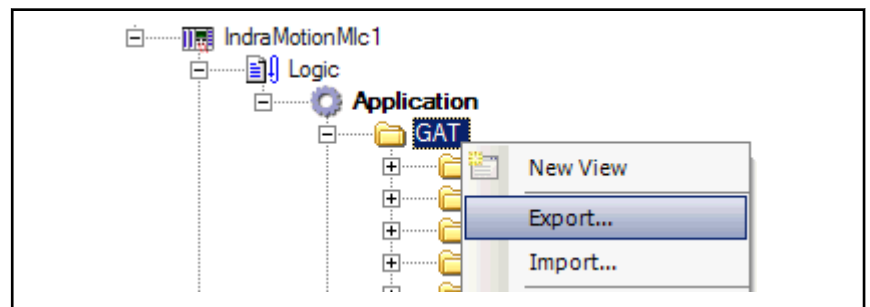


図 11-55: GAT フォルダのエクスポート

2. 手順 1 で生成した *.iwx ファイルには、適合/拡張テンプレートが含まれており、新規プロジェクトを作成するために、以下のとおり GAT ウィザードで使用できます。

分散状態の機械での GAT

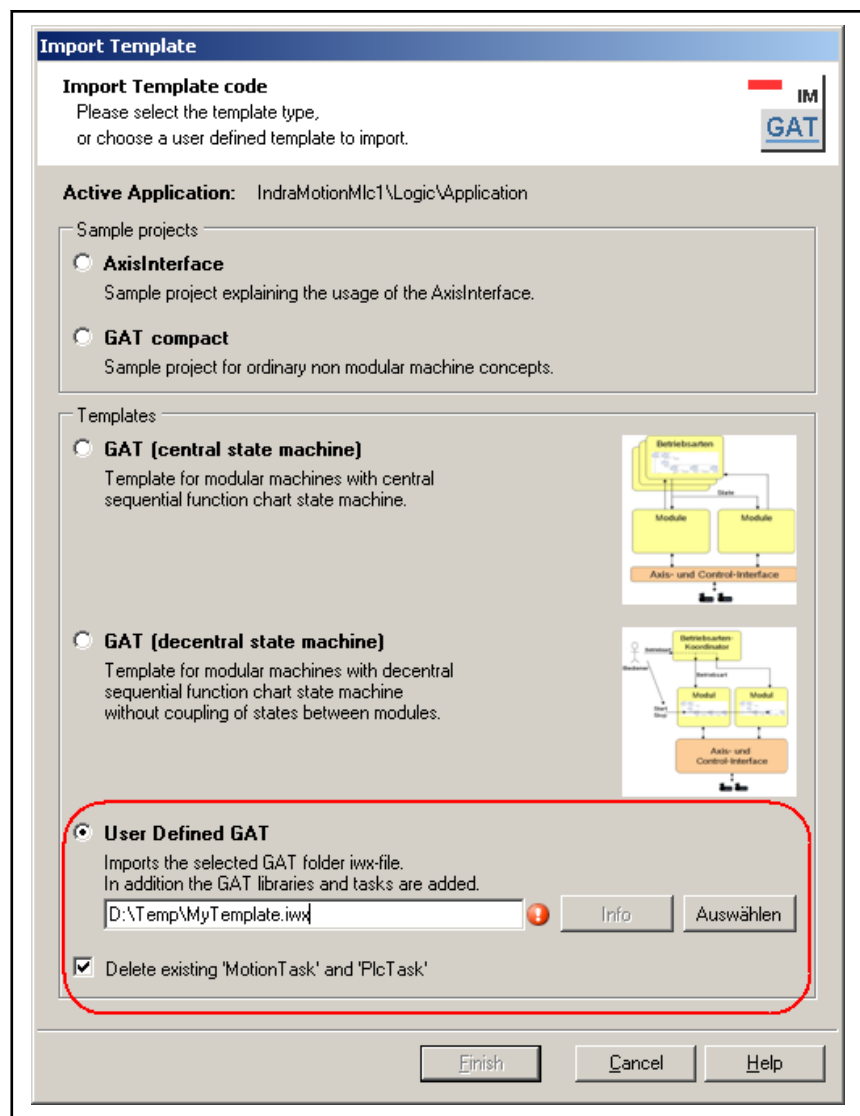


図 11-56: ユーザー定義テンプレートのインポート

11.14 GAT – 分散状態の機械、概略

11.14.1 構造の概要

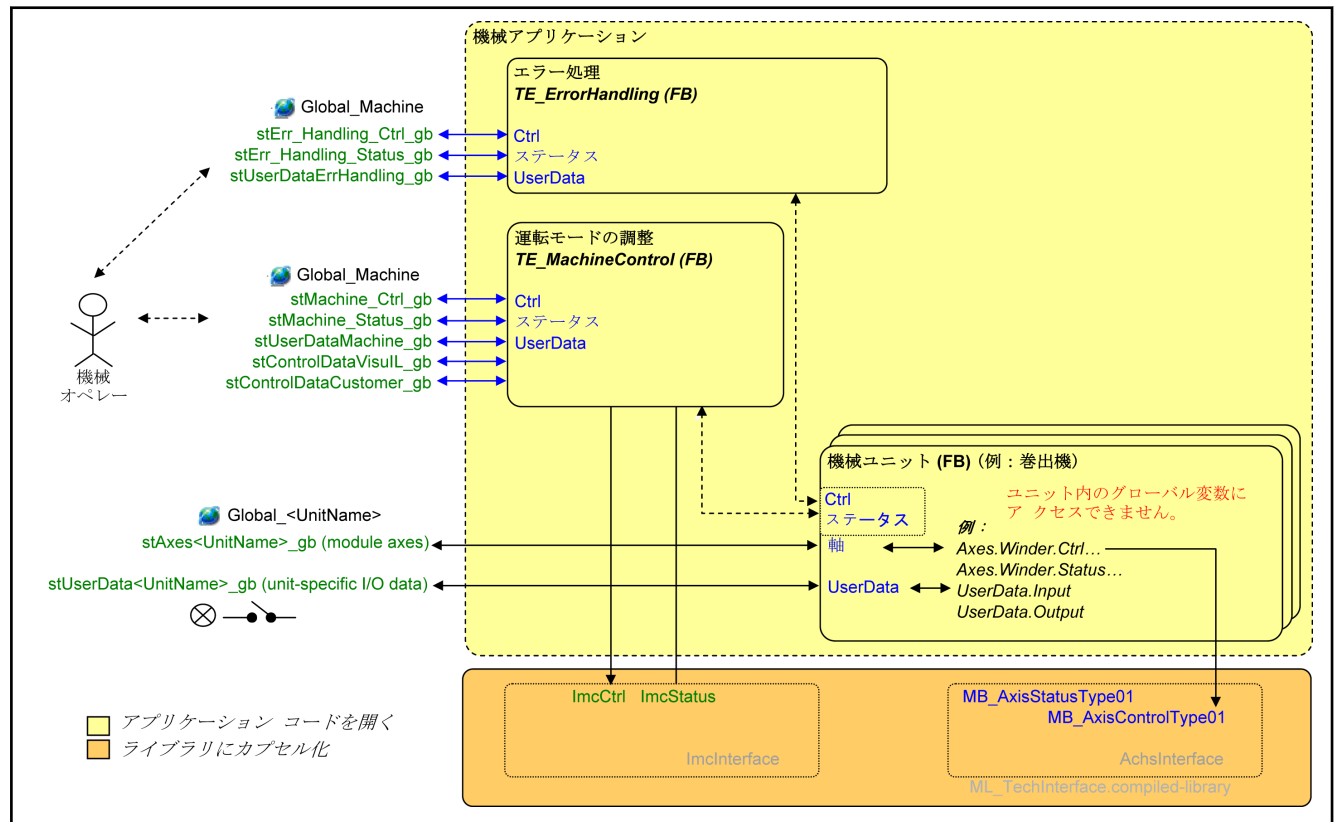


図 11-57: 構造の概要、分散状態の機械での GAT

11.14.2 機械の最も重要なグローバル変数の概要およびエラー処理の概要

下の表は、機械およびエラー処理の最も重要なグローバル変数を示したものです。

変数はグローバル変数リスト "GAT/Machine/Global_Machine" に含まれています。

変数	説明	アクセス
stControlDataVisuIL_gb	一元的機械制御 (運転モードの選択など)、組み込みの試運転表示を使用するコマンド	RD/WR
stControlDataCustomer_gb	一元的機械制御 (運転モードの選択など)、外部表示を使用するコマンド	RD/WR
stControlDataActive_gb	一元的機械制御の有効データ (運転モードの選択など)	RD/WR
stStatusData_gb	HMI 用の機械のステータス データ	RD/WR
stExchangeData_gb	ユニット間のデータ交換用データを含む	RD/WR
stMachineDataEdit_gb	準備されている機械固有のデータ	RD/WR
stMachineDataActive_gb	有効な機械固有のデータ	RD
stProductDataEdit_gb	準備されている製品固有のデータ	RD/WR

分散状態の機械での GAT

変数	説明	アクセス
stProductDataActive_gb_gb	有効な製品固有のデータ	RD
stMachine1Status_gb	機械のステータス (現在の運転モードの出力など)	RD
stUserDataMachine_gb	機械のユーザー固有の I/O データ	RD/WR
arAppErrMachineControl_gb	機械のユーザー固有のエラー フラグ	RD
stErr_Handling_Ctrl_gb	一元的なエラー処理、制御構造 (グローバル エラーの処理のためなど)	RD/WR
stErr_Handling_Status_gb	一元的なエラー処理、ステータス構造 (中央エラー ビットの出力など)	RD
stUserDataErrHandling_gb	エラー処理用のユーザー固有の I/O データ	RD/WR

RDWR

読み取り書き込み

11.14.3 ユニットの最も重要なグローバル変数の概要

下の表は、ユニットの最も重要なグローバル変数を示したものです。
変数はグローバル変数リスト "GAT/MachineUnits/Global_<Unitname>" に含まれています。

変数	説明	アクセス
fb<Unitname>_gb.Ctrl	ユニット制御構造 (運転モードのコマンドなど)	RD/WR
fb<Unitname>_gb.Status	ユニット ステータス構造 (運転モードの確認など)	RD
stUserData<Unitname>_gb	ユニット固有の I/O	RD/WR
arAppErr<Unitname>_gb	ユニットのユーザー固有のエラー フラグの列	RD
stAxes<Unitname>_gb	ユニット軸のあるコンテナ	RD

RDWR

読み取り書き込み

11.14.4 軸の最も重要なグローバル変数の概要

下の表は、軸の最も重要なグローバル変数を示したものです。
変数はグローバル変数リスト "GAT/AxisInterface/Global_GAT_AxisInterfaces" に含まれています。

変数	説明	アクセス
arAxisCtrl_gb	AxisInterface コマンド データの列	RD/WR
arAxisStatus_gb	AxisInterface ステータス データの列	RD

RDWR

読み取り書き込み

11.14.5 TE_MachineControl のメソッドとアクションの概要

下の表は、"TE_MachineControl" ファンクションブロックの最も重要なメソッドとアクションを示したものです。

分散状態の機械での GAT

アクション/メソッド	呼出	説明/例
actOPModeSFC	継続的かつ周期的	運転モード SFC
mCyclic	継続的かつ周期的	ユニット間のロジック
mDataHandler	継続的かつ周期的	内蔵の試運転表示と外部の表示、機械および製品データ処理間のデータ処理/切替
mOpModeDistrubution	継続的かつ周期的	ユニットへの運転モードの配信
mOpModeSet	継続的かつ周期的	プライマリ運転モードの決定
mOneTimeInit	GAT が初期化される際に 1 サイクル。"コールド スタート" 前の実行	"SkipColdstartAtStartup" の設定を初期化する グローバル IndraWorks 軸にユニット軸を割り当てる 参照を初期化する
mInitAllAxes	"InitColdStart" を終了したとき、"InitWarmStart" において	AxisInterface を初期化する
mErrorHandling	継続的かつ周期的	エラーのローカルの発生元を調べ、ローカル エラーをクリアする
FB_init	PLC プログラムの起動/リセット前に 1 回	内部システムの使用は、変更できません

11.14.6 TE_ErrorHandling のメソッドとアクションの概要

下の表は、機能ブロック "TE_ErrorHandling" の最も重要なメソッドとアクションを示したものです。

アクション/メソッド	呼出	説明/例
mScanError	継続的かつ周期的	エラーの発生元を調べる
mDistributeError	継続的かつ周期的	ユニットへのエラーの配信
mClearError	継続的かつ周期的	一元的なエラーのクリア
mOneTimeInit	GAT が初期化される際に 1 サイクル。コールド スタート 前の実行	"SkipColdstartAtStartup" の設定を初期化する グローバル IndraWorks 軸にユニット軸を割り当てる 参照を初期化する
FB_init	PLC プログラムの起動/リセット前に 1 回	内部システムの使用は、変更できません

分散状態の機械での GAT

11.14.7 ユニットのメソッドとアクションの概要

下の表は、ユニットの最も重要なメソッドとアクションを示したものです。

アクション/メソッド	呼出	説明/例
actOPModeSFC	継続的かつ周期的	運転モード SFC
mErrorHandling	継続的かつ周期的	エラーのローカルの発生元を調べ、ローカル エラーをクリアする
mRefreshAxes	継続的かつ周期的	AxisInterface データを更新
mOpModeAcknowledge	継続的かつ周期的	運転モードの確認
FB_init	PLC プログラムの起動/リセット前に 1 回	内部システムの使用は、変更できません

11.14.8 ユニット内の軸へのアクセス

必ず **Axes.<axisname>** 経由でアクセスします。

例 :

// 速度制御モードでのコマンド軸
Axes.MyAxis1.Ctrl.Admin._OpMode.en := ModeVel;

/ If axis has reached the velocity
IF (Axes.MyAxis1.Status.PtrAxisData^.Axis_InVelocity = TRUE) THEN;

11.14.9 ユニット内の I/O データへのアクセス

必ず **UserData.<Variable>** 経由でアクセスします。グローバル変数に直接アクセスしないでください！

例 :

IF (UserData.bInput1=TRUE) THEN // "bInput1" が設定されている場合…
 UserData.bOutput1:=TRUE; // 出力を "bOutput1" に設定します
END_IF

12 GAT プラグイン

12.1 はじめに

GAT プラグインは、すべての GAT バージョン向けの、再ロード可能なオプションの機能拡張です。GAT プラグインを用いると、プログラムのフレームワークを最重要機能のみに絞り、オプション機能は必要な場合のみ再ロードすることができます。

GAT プラグインのコンセプトは以下の特長を備えています。

- **シンプルさ**：プログラミングを必要としない、GAT ウィザードのダイアログによるプラグインの組み込み
- **柔軟性**：コードは"オープン ソース"のコンセプトで改変可能
- **拡張性**：プラグインはユーザーが作成、交換、再利用可能

プラグインは以下の要素により構成されています。

- プログラム、ファンクションブロック、データ構造、表示などの **PLC オブジェクト**
- 呼出ポイントおよび検索コードを判定する **設定データ**
- **ユーザー向け資料**

12.2 プラグインの追加

プラグインは、以下の手順で GAT プロジェクトに追加できます。

- GAT ウィザードで "plug-ins" (プラグイン) を選択する
- "Add plug-in" (プラグインの追加) 機能を選択する
- プラグインを選択し、**Apply** (適用) ボタンを使用して追加する

12.3 プラグインの削除

プラグインは、以下の手順で削除できます。

- GAT ウィザードで "plug-ins" (プラグイン) ノードを展開する
- プラグインを選択する
- "Delete plug-in" (プラグインを削除) を選択する

12.4 プラグインのインストール

プラグインは、後日いつでもエンジニアリング PC にインストールできます。プラグインは、GAT ウィザードを使用して以下の手順でインストールし、選択することができます。

- GAT ウィザードを開く
- "plug-ins" (プラグイン) ノードを選択する
- "open plug-ins configuration directory" (プラグイン設定ディレクトリを開く) リンクをクリックする
- 開いた設定ディレクトリに新しいプラグイン フォルダを追加します。プラグイン フォルダには、以下のファイルを含める必要があります。

GAT プラグイン

必要なプラグイン ファイル

説明	ファイル名
プラグインに必要なすべての PLC オブジェクトを含む IndraWorks Export ファイル	<Plug-inName>.iwx
設定ファイル	<Plug-inName>Config.txt
ユーザー向け資料	<Plug-inName>.pdf または <Plug-inName>.txt

表 12-1: 必要なプラグイン ファイル

12.5 プラグインの作成

ユーザーは以下の手順でプラグインを作成できます。

エクスポート ファイルを生成する

エクスポート ファイルには、プラグインのすべての PLC オブジェクト (プログラム、ファンクションブロック、構造、表示など) が含まれています。
エクスポート ファイルは以下の手順で生成します。

- IndraWorks で新しいプラグイン フォルダ (名前: <Plug-inName>) を作成する
- プラグインの PLC オブジェクトをすべてプラグイン フォルダに移動する
- IndraWorks を使用してプラグイン フォルダをエクスポートする (ファイル名: <Plug-inName>.iwx)

設定ファイルを生成する

GAT ウィザードは、プラグインの呼出コード、呼出位置、概略など、プラグイン固有の情報を必要とします。
設定は以下の部分 (領域) により構成されます。

- プラグインの概略 (GAT ウィザード内の黄色のフィールド)
- プラグイン関連資料へのリンク
- プラグインの呼出ポイントおよび呼出コード
- その他のライブラリ
- その他の PLC タスク

個別の領域は以下において説明されています。

概略

概略は、機能の概要をユーザーに伝えるために使用されます。プラグインの機能について簡単に説明します。プラグインを選択すると、GAT ウィザードのテキスト フィールドにテキストが表示されます。
領域は以下のテキスト マークで示されます。

- "WIZARD BEGIN PLUGIN SHORT DESCRIPTION" (ウィザードがプラグイン概略を開始) は領域の始まりを示します
- "WIZARD END PLUGIN SHORT DESCRIPTION" (ウィザードがプラグイン概略を終了) は領域の終わりを示します

資料へのリンク

概略

GAT ウィザードを経由してプラグイン資料が呼び出されると、この領域で指定されている資料ファイルがウィザードによって開かれます。資料はあらゆる標準形式 (*.pdf、*.txt など) で生成できます。

```
// $ WIZARD BEGIN PLUGIN SHORT DESCRIPTION -----  
My short description...  
// $ WIZARD END PLUGIN SHORT DESCRIPTION -----
```


GAT プラグイン

領域は以下のテキスト マークで示されます。

- "WIZARD BEGIN PLUGIN DOCUMENTATION LINK "(ウィザードがプラグイン資料のリンクを開始) は領域の始まりを示します
- "WIZARD END PLUGIN DOCUMENTATION LINK "(ウィザードがプラグイン資料のリンクを終了) は領域の終わりを示します

以下のオプションが選択可能です。

- ファイル名 (MyPlugin.pdf など) を入力する。ファイルがローカルのプラグイン ディレクトリに呼び出される
- リンク全体を入力し、リンクを経由してファイルを呼び出す。リンクは「http:\\」もしくは「\\」で始まる必要があります。

資料へのリンク

```
// $ WIZARD BEGIN PLUGIN DOCUMENTATION LINK -----
// $ File in the PLUGIn folder or a hyperlink (starting with 'http')
MyPlugInDocu.pdf
// $ WIZARD END PLUGIN DOCUMENTATION LINK -----
```

ファイルを作成するには、任意のテキスト処理プログラムまたはエディタが使用できます。

資料は、なるべくなら以下の構成で英語で作成します。

- 作成者、日付、バージョン、改訂履歴を含む表
- 説明
- インターフェースの説明
- 依存関係/必要なライブラリ

呼出コードおよび呼出位置

説明された領域で、呼出コードと呼出位置が指定されます。

プラグインを用いると、複数の呼出位置に異なる呼出コードを追加できます。このようにして、設定ファイルに任意の数の領域を含めることができます。

領域は以下のテキスト マークで示されます。

- "WIZARD BEGIN PLUGIN CALL UP" (ウィザードがプラグインの呼出を開始) は領域の始まりを示します
- "WIZARD END PLUGIN CALL UP" (ウィザードがプラグインの呼出を終了) は領域の終わりを示します

呼出コードおよび呼出位置

```
// $ WIZARD BEGIN PLUGIN CALL UP -----
// $ BEGIN PLUGIN PATH TO POU -----
// $ path to the ST pou where the PLUGIN call has to be inserted
// $ controls the availability of the PLUGIN in the GAT wizard
Enter the path to the POU where the PLUGIN call has to be inserted ...
// $ END PLUGIN PATH TO POU -----

// $ BEGIN PLUGIN CALL -----
// $ Code region to be inserted in the configured structured text pou
// $ This code is used if there is no other special code for a specific
// $ GAT type is available
Enter code to call and connect the PLUGIN ...
// $ END PLUGIN CALL -----
// $ WIZARD END PLUGIN CALL UP -----
```

領域は 2 つの部分に分かれています。

1 つ目の部分、"PLUGIN PATH TO POU" (POU へのプラグイン パス) は、呼出コードがどの位置に挿入されるかを説明します。以下の構文規則が使用されます。

GAT プラグイン

プログラム

```
/Gat-Typ = Path to code position | InsertLocation=Implementation/Declaration | InsertPosition=End/Top |
InsertMark="marked text"
```

以下の GAT タイプがサポートされています。

- GatCentral
- GatDecentral
- GatCompact
- AxisInterface



"Path to code position" (コード ポジションへのパス) で #AllUnits# および #AllModules# の構文を使用することにより、すべての GAT ユニットまたはすべての GAT モジュールに呼出コードを挿入できます。

例：

- GatCentral = #AllModules#\actCyclic: 呼出コードはすべてのモジュールの "actCyclic" アクションに挿入されます
- GatCentral = #AllModules#: 呼出コードはすべてのモジュールの機能ブロックに挿入されます
- GatDecentral = #AllUnits#\Actions\actOff: 呼出コードはすべてのユニットの "actOff" アクションに挿入されます
- GatDecentral = #AllUnits#: 呼出コードはすべてのユニットのファンクションブロックに挿入されます

呼出コードがどこに挿入されるかは、以下の要素により制御されます。

- "InsertLocation" は、呼出コードが PLC オブジェクトの実行部分または宣言部分のどちらに挿入されるかを説明します。
- "InsertPosition" は、呼出コードが最後 (End) または最初 (Top) のどちらに挿入されるかを説明します。
- "InsertMark" は任意で、テキスト マークを説明します。このテキストマークの前 (InsertPosition=Top) または後 (InsertPosition=End) に、呼出コードが挿入されます。

2 番目の部分、"BEGIN PLUGIN CALL" (プラグインの呼出開始) は、呼出コードを説明します。

以下の規則が適用されます。

- GAT タイプ固有の呼出コードが含まれていない場合は、一般的な呼出コードが追加されます
- 一般的な呼出コードは、構文 "BEGIN PLUGIN CALL" (プラグインの呼出開始) によって示されます
- GAT タイプ固有の呼出コードが設定ファイルに含まれている場合は、そのコードが挿入されます
- GAT タイプ固有の呼出コードは、以下の構文により示されます。
 - "BEGIN PLUGIN CALL GATCENTRAL" (GATcentral の場合)
 - "BEGIN PLUGIN CALL GATDECENTRAL" (GATdecentral の場合)
 - "BEGIN PLUGIN CALL GATCOMPACT" (GATcompact サンプルプロジェクトの場合)
 - "BEGIN PLUGIN CALL AXISINTERFACE" (AxisInterface サンプルプロジェクトの場合)

GAT プラグイン

次の例では、メソッド実行の最後に呼出コード "Test" が Machine ファンクションブロックの mCyclic に追加されます：

```
// $ WIZARD BEGIN PLUGIN CALL UP -----
// $ BEGIN PLUGIN PATH TO POU -----
// $ path to the ST pou where the PLUGIN call has to be inserted
// $ controls the availability of the PLUGIN in the GAT wizard
GatCentral = Gat\Machine\mCyclic | InsertLocation=Implementation | InsertPosition=End
// $ END PLUGIN PATH TO POU -----

// $ BEGIN PLUGIN CALL -----
// $ Code region to be inserted in the configured structured text pou
// $ This code is used if there is no other special code for a specific
// $ GAT type is available
Test
// $ END PLUGIN CALL -----
// $ WIZARD END PLUGIN CALL UP -----
```

その他のライブラリ

ライブラリ マネージャに含まれていない他の PLC ライブラリは、説明領域を経由して追加できます。

領域は以下のテキスト マークで示されます。

- "WIZARD BEGIN PLUGIN LIBRARIES" (ウィザードがプラグイン ライブラリを開始) は領域の始まりを示します
- "WIZARD END PLUGIN LIBRARIES" (ウィザードがプラグイン ライブラリを終了) は領域の終わりを示します

ライブラリはプレースホルダーを介して、または直接指定することができます。

その他のライブラリ

```
// $ WIZARD BEGIN PLUGIN LIBRARIES -----
// $ add required libraries by PlaceholderName or fixed
Optional: Enter additional libraries which are required by the PLUGIN...
// $ example PlaceholderName: RIL_PROFIBUSDP_02 = RIL_ProfibusDP_02, * (System)
// $ example fixed: RIL_ProfibusDP_02, 12.2.0.0 (System)
// $ WIZARD END PLUGIN LIBRARIES -----
```

その他の PLC タスク

タスク設定に含まれていない他の PLC タスクは、説明領域を経由して追加できます。

領域は以下のテキスト マークで示されます。

- "WIZARD BEGIN PLUGIN TASKS" (ウィザードがプラグイン タスクを開始) は領域の始まりを示します
- "WIZARD END PLUGIN TASKS" (ウィザードがプラグイン タスクを終了) は領域の終わりを示します

各タスクは、IndraWorks を介して個別にエクスポートする必要があります。このエクスポート ファイル (*.iwx) は、ローカルのプラグイン ディレクトリに保存する必要があります。続いて、設定ファイルでエクスポート ファイルの名前が指定されます。各タスクについて、個別のエクスポート ファイルを生成し、入力する必要があります。

その他の PLC タスク

```
// $ WIZARD BEGIN PLUGIN TASKS -----
// $ imports the additional defined tasks of the plugin
// $ taskname and filename are equal
Optional: Enter additional tasks which are required by the PLUGIN
Example:
PlugInTask1.iwx
// $ WIZARD END PLUGIN TASKS -----
```

設定ファイルの例

次の例は完全な設定ファイルを示しています。

テキスト エディタに以下のテキストを"コピー & ペーストすること"により、設定ファイルを作成できます。

GAT プラグイン

あるいは、既存のプラグインから設定ファイルをコピーし、必要に応じて変更を加えます。

設定ファイルの例

```
//$ WIZARD BEGIN PLUGIN SHORT DESCRIPTION -----
Enter PLUGIN short description here...
//$ WIZARD END PLUGIN SHORT DESCRIPTION -----

//$ WIZARD BEGIN PLUGIN DOCUMENTATION LINK -----
//$ File in the PLUGIN folder or a hyperlink (starting with 'http')
Enter documentation file name here...
//$ WIZARD END PLUGIN DOCUMENTATION LINK -----

//$ -----
//$ WIZARD BEGIN PLUGIN CALL UP -----
//$ -----

//$ BEGIN PLUGIN PATH TO POU -----
//$ path to the ST pou where the PLUGIN call has to be inserted
//$ controls the availability of the PLUGIN in the GAT wizard
Enter the path to the POU where the PLUGIN call has to be inserted ...
Examples:
GatCentral = Gat\Machine\Machine\mCyclic | InsertLocation=Implementation | InsertPosition=End
GatDecentral = ...
GatCompact = ...
AxisInterface = ...
//$ END PLUGIN PATH TO POU -----

//$ BEGIN PLUGIN CALL -----
//$ Code region to be inserted in the configured structured text pou
//$ This code is used if there is no other special code for a specific
//$ GAT type is available
Enter code to call and connect the PLUGIN ...
//$ END PLUGIN CALL -----

//$ BEGIN PLUGIN CALL GATCENTRAL -----
//$ This section is only required in case special code is required
//$ for GATcentral
Optional : Enter code to call and connect the PLUGIN with GATcentral...
//$ END PLUGIN CALL GATCENTRAL -----

//$ BEGIN PLUGIN CALL GATDECENTRAL -----
//$ This section is only required in case special code is required
//$ for GATdecentral
Optional : Enter code to call and connect the PLUGIN with GATdecentral...
//$ END PLUGIN CALL GATDECENTRAL -----

//$ BEGIN PLUGIN CALL GATCOMPACT -----
//$ This section is only required in case special code is required
//$ for GATcompact example project
Optional : Enter code to call and connect the PLUGIN with GATcompact...
//$ END PLUGIN CALL GATCOMPACT -----

//$ BEGIN PLUGIN CALL AXISINTERFACE -----
//$ This section is only required in case special code is required
//$ for the axis interface example project
Optional : Enter code to call and connect the PLUGIN with AxisInterface...
//$ END PLUGIN CALL AXISINTERFACE -----

//$ -----
//$ WIZARD END PLUGIN CALL UP -----
//$ -----

//$ WIZARD BEGIN PLUGIN LIBRARIES -----
//$ add required libraries by PlaceholderName or fixed
Optional: Enter additional libraries which are required by the PLUGIN...
//$ example PlaceholderName: RIL_PROFIBUSDP_02 = RIL_ProfibusDP_02, * (System)
//$ example fixed: RIL_ProfibusDP_02, 12.2.0.0 (System)
//$ WIZARD END PLUGIN LIBRARIES -----

//$ WIZARD BEGIN PLUGIN TASKS -----
//$ imports the additional defined tasks of the plugin
//$ taskname and filename are equal
Optional: Enter additional tasks which are required by the PLUGIN and
Example:
PlugInTask1.iwx
//$ WIZARD END PLUGIN TASKS -----
```

13 サービスとサポート

お客様のニーズにお答えして最適で素早いサポートのために世界にまたがるサービスネットワークを提供しております。そこでは弊社のエキスパートが皆様のご連絡をお待ちしております。お問い合わせは毎日 24 時間可能です。

サービス ドイツ

ドイツ、ロアにあるサービスセンターがサーボドライブシステムに関するサービスを一手に引き受けています。

ヘルプデスク & ホットラインのコンタクトデータは下記の通り：

電話： +81 -568-(35)-7701
Fax: +81 -568-(35)-7705
E-mail: servo.service@boschrexroth.co.jp
インター
ネット: <http://www.boschrexroth.co.jp>

サービス、修理（送付先住所など）およびトレーニングに関するその他の情報は弊社のホームページをご覧ください。

サービス 世界

ドイツ国外のお客様はまず販売店・サービスセンターへご連絡ください。ホットラインの電話番号はインターネットの販売店住所をご参照下さい。

必要な情報

より早く、効果的に対応できるように下記の情報をご用意ください：

- 異常と異常が発生する状況の詳細
- 該当製品のプレートネーム、タイプコードと製造番号
- お客様の連絡先（電話とファックス番号および電子メールアドレス）

索引

記号

(MB_AXIS_STATUS_TYPE01 内).....	103
安全上の指示.....	15
意図された	
最新技術.....	19
運転モード	
Drive ready.....	71
Drive-controlled interruption (ドライブ制御の中断).....	81
Phase synchronization (フェーズ同期).....	76
運転モード.....	172
Absolute Circular Positioning (絶対円形位置設定).....	178
Absolute Linear Positioning (絶対線形位置設定).....	176
Absolute Positioning (絶対位置設定).....	73
Absolute PTP Positioning (絶対 PTP 位置設定).....	174
Additive Positioning (追加位置設定).....	75
Continue.....	181
Drive halt.....	72
Drive Homing (ドライブ ホーミング).....	73
Drive stop.....	72
Electronic cam (電子カム).....	77
Electronic FlexProfile.....	79
Electronic MotionProfile.....	78
External function block (外部ファンクションブロック).....	82
Kinematics Homing (キネマティクス ホーミング).....	174
Kinematics ready (キネマティクス準備完了).....	173
Kinematics Stopping (キネマティクス停止).....	180
RCL.....	182
Relative Circular Positioning (相対円形位置設定).....	179
Relative Linear Positioning (相対線形位置設定).....	177
Relative Positioning (相対位置設定).....	74
Relative PTP Positioning (相対 PTP 位置設定).....	175
Robot Control.....	80
Velocity Control (速度制御).....	75
Velocity synchronization (速度同期).....	77
外部ファンクションブロック.....	182
中断.....	181
運転モード、一般情報.....	34
運転モードから独立したロジック.....	34
運転モードの状態.....	278
運転モードの変更、GAT.....	280
最初の手順、AxisInterface.....	83
最初の手順、キネマティクス インターフェース.....	184
使用されている記号.....	16
使用開始時の TechInterface 設定.....	56

使用上の重要な指示.....	19
使用目的.....	19
使用範囲および用途.....	19
資料の構成.....	9
時間依存でアプリケーション固有のエラーリアクション.....	325
自動運転モードに関する一般情報.....	34
軸モード.....	67
手動運転モードに関する一般情報.....	34
周期的処理、一般情報.....	34
集中状態の機械での GAT.....	263
AxisInterface の拡張.....	319
F0 エラーへの GAT エラーリアクションのスイッチオフ.....	325
GAT エラー処理.....	289
GAT に基づいた PackML V3.0 プログラムフレームワーク.....	323
GAT に基づいたユーザー独自のプログラムフレームワークの生成.....	322
GAT のインターフェース.....	304
GAT の構造の概要.....	326
GAT、概略、機械のメソッドとアクションの概要.....	328
MotionTask での処理.....	319
エラー クリア後の必要な運転モードの選択.....	311
エラー処理用データ.....	305
コントロール データ.....	307
タスク処理.....	288
バージョン 10/11 から 12/13 への更新.....	325
モジュール.....	282
モジュール データ.....	305
モジュールでのグローバル変数とアドレス変数の使用.....	317
モジュールのインスタンスの追加.....	314
モジュールの再利用.....	315
モジュールの最も重要なグローバル変数の概要.....	327
モジュールの無効化.....	314
モジュール内の I/O データへのアクセス.....	330
モジュール内の軸へのアクセス.....	330
運転モード.....	277
運転モードの追加.....	311
運転モード仕様の上位ロジック.....	312
概要.....	263
機械データ.....	304
機械のメソッドとアクションの概要.....	328
機械の最も重要なグローバル変数の概要.....	327
機能原則.....	276
最初の手順.....	266
最初の手順、単純なモーションアプリケーションの作成.....	266
試運転サポートのための内部表示.....	294
時間依存でアプリケーション固有のエラーリアクション.....	325
軸データ.....	306

索引

軸データへのアクセス.....	307	コントロール データ.....	375
軸の最も重要なグローバル変数の概要.....	327	タスク管理.....	352
軸の削除/名前変更.....	309	プライマリ運転モードの追加.....	377
軸の追加.....	308	ユニット.....	346
初期化.....	276	ユニット データ.....	373
状態と移行の追加.....	312	ユニットでのグローバル変数とアドレス	
新しいモジュールの追加.....	313	変数の使用.....	381
操作手順：一般的なユーザー アクティビ		ユニットのインスタンスの追加.....	379
ティ.....	307	ユニットのメソッドとアクションの概要.....	390
独立した HMI による外部表示		ユニットの再利用.....	380
(WinStudio 表示).....	295, 362	ユニットの最も重要なグローバル変数の	
表示.....	294, 359	概要.....	388
初期化.....	33	ユニットの無効化.....	379
診断メッセージの生成と表示.....	34	ユニット内の I/O データへのアクセス.....	390
設定.....	56	ユニット内の軸へのアクセス.....	390
逐次プログラミング.....	351	運転モード.....	345
電気ドライブおよび制御の安全の手引き.....	21	基本概念.....	331
電気ドライブシステム.....	21	機械データの処理.....	358
表示.....	43	機械のマシン ハンドラー データ	
IMC インターフェース コントロール.....	95	(TE_MachineControl).....	371
IMC インターフェース ステータス.....	95	機械の最も重要なグローバル変数の概要	
PLC load (PLC 負荷).....	97	およびエラー処理の概要.....	387
Positioning mode (位置設定モード).....	89	機能原則.....	344
Robot Control 言語.....	191	最初の手順.....	333
Robot Control 言語プログラム.....	194	最初の手順、準備.....	333
Robot Control 言語ポイント.....	193	最初の手順、単純なモーションアプリケ	
Synchronization mode (同期モード).....	91	ーションの作成.....	333
Synchronization mode with profiles (プロ		使用の要件.....	333
ファイルによる同期モード).....	93	試運転サポートのための内部表示.....	360
システム概要.....	85	軸データ.....	375
キネマティクスの概要.....	187	軸データへのアクセス.....	376
システム概要.....	185	軸の最も重要なグローバル変数の概要.....	388
ジョグ/セットアップ モード.....	192	軸の削除/名前変更.....	377
ベルト設定.....	191	軸の追加.....	376
位置設定の概要.....	189	集中エラー処理データ (TE_ErrorHandling).....	372
概要.....	84, 184	初期化.....	344
参照の概要.....	188	新しいユニットの追加.....	378
Axis overview (軸の概要).....	87	製品データの処理.....	358
Velocity mode (速度モード).....	90	製品と機械のデータ処理.....	358
表示、GAT、分散.....	359	操作手順：一般的なユーザー アクティビ	
表示、PLC を使用したデータ提供.....	44	ティ.....	376
記号設定.....	44	表示の切替.....	359
通信設定.....	47	分散状態の機械での GAT のインターフ	
表示の一般的な内容.....	43	ェース.....	371
機械ステータスの表示.....	43	保護超低電圧.....	25
機械の状態の制御.....	44	目的外の使用.....	20
分散状態の機械での GAT.....	331	結果、免責.....	19
AxisInterface の拡張.....	381	用語および略語.....	16
BackgroundTask の追加.....	352	略語.....	16
GAT に基づいたユーザー独自のプログラ			
ム フレームワークの生成.....	385	A	
GAT の構造の概要.....	387	actAssignIO.....	328
TE_MachineControl のメソッドとアクシ		actAssignIOBackground.....	328
ョンの概要.....	388	actAssignIOMotionSync.....	328
エラーリアクション.....	355	actAutoProducing.....	329
エラー処理.....	353	actAutoStopped.....	329
エラーのクリア.....	357	actDisable.....	315, 329
エラーの検出および診断.....	353		

actError.....	290, 329
actInitColdstart.....	328
actInitWarmstart.....	328
actManualReady.....	329
actOpModeSFC.....	345
arAppErrMachine1_gb.....	327
arAxisCtrl_gb.....	306
arAxisStatus_gb.....	306
arMachine1ModuleDiagNo_gb.....	327
AXIS_GAT.....	294
Axis_Ref.....	114
AxisInterface の表示.....	84
AxisInterface の運転モード.....	69
AxisInterface の拡張、GAT.....	319
AxisInterface の拡張、GAT、分散.....	381
AxisInterface のファンクションブロック.....	59
AxisInterface の構造.....	102
(MB_AXIS_STATUS_TYPE01 内).....	103
Axis_Mode.....	113
MB_AXIS_ADJUST_DATA.....	115, 119
MB_AXIS_ADMIN_STATUS.....	104
MB_AXIS_ADMINISTRATION.....	111
MB_AXIS_CONTROL_TYPE01 構造.....	110
MB_AXIS_CONTROL_TYPE02.....	125
MB_AXIS_CONTROL_TYPE02 構造.....	125
MB_AXIS_Coordinated - 構造.....	126
MB_AXIS_DIAGNOSIS.....	106
MB_AXIS_FEATURES.....	105
MB_AXIS_MODE.....	105
MB_AXIS_POS_STATUS.....	108
MB_AXIS_POSITIONING.....	115, 116
MB_AXIS_PROFILE.....	122
MB_AXIS_STATUS_TYPE02.....	124
MB_AXIS_STATUS_TYPE02 - 構造.....	124
MB_AXIS_SYNC_STATUS.....	109
MB_AXIS_SYNCHRONISATION.....	119
MB_AXIS_VEL_STATUS.....	109
MB_AXIS_VELOCITY.....	118
AxisInterface の構造要素.....	
ImcInterface コントロール.....	96
ImcInterface ステータス.....	96
Positioning Mode (位置設定モード) 表示.....	90
Synchronization mode (同期モード) 表示.....	92
Synchronization mode with profiles (プロ ファイルによる同期モード).....	94
速度モードの表示.....	91
AxisInterface の構造要素.....	
軸の概要表示.....	88
AxisInterface プロジェクトの変換.....	58
AxisInterface 用プログラム.....	83
B	
bStart_gb.....	327
bStop_gb.....	327
C	
ClearError.....	293

D

DisableErrorHandling GATcentral.....	292
DisableErrorReaction GATdecentral.....	357
DisableLocked.....	315
DisableTransient.....	315
DriveInterruptReaction.....	81

E

EnableCyclicScanning.....	52

G

GAT compact.....	
最初の手順、単純なモーションアプリケ ーションの作成.....	201
GAT の初期化.....	276, 344
分散状態の機械での GAT.....	
TE_ErrorHandling のメソッドとアクショ ンの概要.....	389
GAT のインターフェース.....	304
GAT のインターフェース、分散.....	371
GAT の運転モード.....	277
GAT の機能原則.....	276, 344
GAT プラグイン.....	391
はじめに.....	391
プラグインの作成.....	392
プラグインの削除.....	391
プラグインの追加.....	391
プラグインを後日インストールする.....	391
GAT 初期化後のターゲット モード.....	279
GATBackgroundProg.....	289
GATBackgroundProgTask.....	289
GATcompact.....	
AxisInterface の拡張.....	236
BackgroundTask の追加.....	216
GAT のインターフェース.....	230
MotionTask での処理.....	236
MotionTask の追加.....	214
エラーの検出および診断.....	217
エラーリアクション.....	218
エラー処理.....	217
エラーのクリア.....	221
エラー処理用データ.....	231
コントロール データ.....	232
タスク処理.....	213
タスク設定.....	213
運転モード.....	211
運転モードの追加.....	235
機械のメソッドとアクションの概要.....	241
機械の最も重要なグローバル変数の概要.....	240
機械状態のデータ.....	231
機械内の軸へのアクセス.....	241
構造の概要.....	240
最初の手順.....	201
軸データ.....	232
軸データへのアクセス.....	232
軸の最も重要なグローバル変数の概要.....	241
軸の削除/名前変更.....	234

索引

軸の追加.....	233	MB_KINEMATICS_ADMIN_STATUS.....	156
軸番号が 16 より大きい軸の最大数を大		MB_KINEMATICS_ADMINISTRATION.....	139
きくする	234	MB_KINEMATICS_COORDINATED.....	152
周期的処理.....	213	MB_KINEMATICS_DIAGNOSIS.....	161
初期化.....	208	MB_KINEMATICS_RCL.....	155
独立した HMI による外部表示		MB_KINEMATICS_SETUP.....	151
(WinStudio 表示).....	223	MB_KINEMATICS_SETUP_STATUS.....	162
表示.....	221	MB_Phasing.....	97
GATcompact のサンプル プロジェクト.....	201	MB_PhasingSlave.....	98
GATMotionSyncProg.....	288	mBackground.....	289, 328, 329
GATMotionSyncTask.....	288	mClearError.....	333
GATProg.....	312	mCyclic.....	328, 329
Global_GAT_MachineControl.....	305	mDataHandler.....	358
Global_ImcInterface.....	82	mDistributeError.....	333
Global_MB_AxisInterfaceVars.....	82	mEntryOpMode.....	281, 328, 329
Global_MB_KinInterfaceVars.....	183	mErrorHandling.....	353
I		mExitOpMode.....	280, 328, 329
IMC インターフェースの構造.....	126	mFB_AxesInit.....	329
ImcCtrl.....	307	mForceOpMode.....	281, 329
ImcStatus.....	307	mGetAxis.....	308, 329
IndraWorks/IndraLogic 使用時の推奨タスク		mlsCriticalError.....	292, 328, 329
分割	35	mlsMyAxis.....	329
MotionTask の一般的なタスク.....	36	ML_KinematicsInitType01.....	162, 163
BackgroundTask の一般的なタスク.....	36	ML_KinematicsInterfaceType01.....	165, 166
PlcTask の一般的なタスク.....	36	ML_KinJog.....	169, 171
Init.....	276	ML_KinJogV1.....	167, 168
K		ML_Kintech.library.....	136
KinSystem_Overview_All.....	185	ML_TechInterface.library.....	49
M		mMotionCyclic.....	288, 319, 328, 329
MB_AXIS_ADJUST_DATA.....	115, 119	ModeAB.....	71
MB_AXIS_ADJUST_STATUS.....	108	ModeAH.....	72
MB_AXIS_ADMIN_STATUS.....	104	ModeCoordAB.....	173
MB_AXIS_ADMINISTRATION.....	111	ModeCoordAH.....	173
MB_AXIS_CONTROL_TYPE01.....	110, 126	ModeCoordContinue.....	181
MB_AXIS_CONTROL_TYPE02.....	125, 126	ModeCoordExternalFB.....	182
MB_AXIS_Coordinated - 構造.....	126	ModeCoordHoming.....	174
MB_AXIS_DIAGNOSIS.....	106	ModeCoordInterrupt.....	181
MB_AXIS_FEATURES.....	105	ModeCoordPosCircAbs.....	178
MB_AXIS_MODE.....	105	ModeCoordPosCircRel.....	179
MB_AXIS_POS_STATUS.....	108	ModeCoordPosDirAbs.....	174
MB_AXIS_POSITIONING.....	115, 116	ModeCoordPosDirRel.....	175
MB_AXIS_PROFILE.....	122	ModeCoordPosLinAbs.....	176
MB_AXIS_STATUS_TYPE02.....	124	ModeCoordPosLinRel.....	177
MB_AXIS_SYNC_STATUS.....	109	ModeCoordRCL.....	182
MB_AXIS_SYNCHRONISATION.....	119	ModeCoordStopping.....	180
MB_AXIS_VEL_STATUS.....	109	ModeExternalFB.....	82
MB_AXIS_VELOCITY.....	118	ModeFlexProfile.....	79
MB_AxisInitType01.....	59	ModeHoming.....	73
MB_AxisInterfaceType01.....	62, 64, 65, 66	ModeInterrupted.....	81
MB_AxisInterfaceType02.....	65, 66, 67	ModePosAbs.....	73
MB_IMC_Control_Type01.....	127	ModePosAdd.....	75
MB_IMC_Status_Type01.....	127	ModePosRel.....	74
MB_IRDProgSelect.....	66	ModeStopping.....	72
MB_IRDProgStart.....	66	ModeSyncCam.....	77
		ModeSyncPhase.....	76
		ModeSyncProfile.....	78
		ModeSyncVel.....	77
		ModeVel.....	75

mOneTimeInit.....	277, 316, 328, 344
mOpModeChangeAllowed.....	280, 328
mReleaseOpMode.....	281
mScanError.....	332
mSetAppError.....	289, 354
mSetFBError.....	290, 355
mSetModuleEnable.....	314

O

OPMODE_AUTO.....	279
OpMode_ErrorSuccessor.....	311
OPMODE_INIT_COLD.....	344
OPMODE_INIT_WARM.....	344

P

PackML V3.0.....	323
PELV.....	25
PowerOn.....	71

R

Robot Control.....	67, 80, 130, 237, 320, 382

S

sercos なしの IndraLogic XLC 用 GATcompact	
GAT のインターフェース.....	256
エラーのクリア.....	250
エラーの検出および診断.....	247
エラーリアクション.....	249
エラー処理.....	247
エラー処理用データ.....	257
コントロール データ.....	258
タスク処理.....	246
タスク設定.....	246
運転モード.....	245
運転モードの追加.....	258
機械データ.....	256
機械のメソッドとアクションの概要.....	260
機械の最も重要なグローバル変数の概要.....	260
構造の概要.....	260
周期的処理.....	246
初期化.....	244
独立した HMI による外部表示 (WinStudio 表示).....	250
表示.....	250
sercos なしの IndraLogic XLC 用 GATcompact	
BackgroundTask の追加.....	246
sercos なしの IndraLogic XLC 用 GATcompact のサンプル プロジェクト.....	243
SkipColdstartAtStartup.....	277
stControlDataActive_gb.....	359
stControlDataCustomer_gb.....	359
stControlDataVisuIL_gb.....	359
stMachine1Ctrl_gb.....	327
stMachine1Status_gb.....	327
System_Overview_All_Axis.....	85

T

TE_AxisInterfaceType01Prog.....	83
TE_BASE_MODULE_AXIS.....	283
TE_BaseUnit.....	346
TE_CMI_CONTROL.....	305
TE_CMI_STATUS.....	305
TE_DummyMode.....	277
TE_DummyModule.....	282
TE_ErrorHandling.....	353
TE_GAT_AXIS.....	283
TE_KinInterfaceType01Prog.....	184
TE_Machine.....	318
TE_MACHINE_CTRL.....	304
TE_MACHINE_DATA.....	358
TE_MACHINE_STATUS.....	304
TE_MachineControl.....	352
TE_OPMODES.....	345
TE_UNIT_CONTROL.....	347
TE_UNIT_STATUS.....	347
TechInterface	
導入と概要.....	49

U

UpdateEveryInput.....	53
UserData.....	317, 330

ア

アプリケーションの自動化.....	33
アプリケーション固有のウォッチドッグ.....	39

エ

エラーリアクション.....	41
エラーリアクション、GAT.....	290
エラーリアクション、GAT.....	355
エラー確認/エラーの後の再起動.....	43
エラー検出.....	40
エラー処理、一般情報.....	40
エラー処理の原則.....	40
エラー診断.....	42

キ

キネマティクス インターフェースのファン クションブロック.....	162
キネマティクス インターフェースのプログ ラム.....	184
キネマティクス インターフェースの運転モード 概要.....	172
キネマティクス インターフェースの構造.....	138
Kinematics_Conveyor_Belt.....	144
Kinematics_Conveyor_Belt_V1.....	146
Kinematics_Mode_Union.....	143
Kinematics_RCL_Point.....	147
Kinematics_RCL_Prog.....	149
Kinematics_RCL_Status.....	158
MB_KINEMATICS_ADMIN_STATUS.....	156
MB_KINEMATICS_ADMINISTRATION.....	139
MB_KINEMATICS_CONTROL_TYPE01.....	138

索引

MB_KINEMATICS_CONTROL_TYPE01	
構造	138
MB_KINEMATICS_COORDINATED	152
MB_KINEMATICS_DIAGNOSIS	161
MB_KINEMATICS_RCL	155
MB_KINEMATICS_SETUP	151
MB_KINEMATICS_SETUP_STATUS	162
MB_KINEMATICS_STATUS_TYPE01	155
MB_KINEMATICS_STATUS_TYPE01 構造	155
キネマティクス インターフェースの表示	184

ク

グローバル変数	82, 183

こ

この資料について	9
資料の有効性	9
情報の表記方法	15

サ

サイクル時間の計測	37
PLC プロジェクトにおける時間計測	38
コントロールのタスク実行ビュー	38
タスク設定のサイクル時間ダイアログ	37
サポート	
サービスホットラインを参照	397

シ

システム概要へのキネマティクスの追加	186
システム概要への軸の追加	86

タ

ダイアログベースの軸コマンド	67
タスク管理、一般情報	34
タスク管理の原則	35
タスク設定、GAT	288
タスク設定、GAT、分散	352

ト

ドライブシステム	21

ナ

ナビゲーション システムの概要	86, 187

は

はじめに	31

フ

プロトコル エラー	294, 358

モ

モジュール	282
モジュール インスタンス	286
モジュール インターフェース	283
モジュール軸	283
モジュールの再利用	315

モジュールの状態	287
モジュールの連携	287
モジュール化規則	282, 347
モジュール固有の I/O データ	286

ユ

ユーザー拡張	97, 195
arAxisCtrl_gb[]	99
arStatusCtrl_gb[]	100
キネマティクス インターフェースの拡張	196
ファンクションブロック	100, 196
概要	195
ユーザー定義 GAT	322
ユーザー定義 GAT、分散	385
ユース ケース	197
キネマティクス インターフェースによる 逐次プログラミング	197
ユーザー定義変換の登録	199
ユニット	346
ユニット インスタンス	350
ユニット インターフェース	347
ユニットの再利用	380
ユニットの連携	352
ユニット固有の I/O データ	350
ユニット軸	347
ユニット内のアドレス変数	381
ユニット内のグローバル変数	381
ユニット内の運転モード	350

ラ

ライブラリ	
ML_KinTech.library	136
ML_TechInterface.library	49

メモ

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Tel. +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics



R911341767

DOK-XLCMLC-TF*GAT**V13-AP02-JA-P