

RC8 Command-Slave活用ガイド

株式会社デンソーウェーブ

RC8 Command-Slave 活用ガイド

著作権・商標について

スクリーンショットは、オムロン株式会社のSysmac Studioの画面を使用しています。
本資料に記載されている会社名・製品名は、それぞれ各社の商標または登録商標です



RC8 Command-Slave 活用ガイド

目次

1.	準備	4
1.1.	NJ 用ファンクションブロックライブラリのダウンロード	4
1.2.	Sysmac Studio への FB ライブラリインポート	8
1.2.1.	Sysmac Studio 上での FB ライブラリインポート	8
1.3.	RC8Command-Slave 機能の設定	12
1.3.1.	Command-Slave 機能オプションの有効化	12
1.3.2.	Command-Slave 起動フラグの有効化	19
1.3.3.	Command-Slave ポーリング周期の設定	23
1.3.4.	RC8 起動権の設定	30
1.3.5.	Command-Slave 外部入力信号(Mini I/O)の設定	33
1.3.6.	ティーチングペンドントの設定	34
1.4.	Command-Slave 機能の使い方	35
1.4.1.	複数のプログラムで構成する場合	35
1.4.2.	1 つのプログラムで構成する場合	37
1.4.3.	構造型体データ	39
1.4.4.	列挙型データ	41
1.4.5.	Command-Slave 変数の変換、ロボット Enable、モータ処理	43
1.4.6.	変数書込	45
1.4.7.	変数読込	48
1.4.8.	ツール設定	50
1.4.9.	ツール変更とツール番号の取得	53
1.4.10.	ワーク設定	55
1.4.11.	ワーク変更とワーク番号の取得	58
1.5.	エラー表示の見方	60
2.	プログラミング例	61
2.1.	動作ステップ	62
2.2.	PLC プログラム構成	68
2.2.1.	Section0(設定、有効処理、初期化)の説明	69
2.2.2.	Section1(ロボット動作)の説明 (Next 処理を使用する場合)	72
2.2.3.	Section1(ロボット動作)の説明 (Next 処理を使用しない場合)	77
3.	アプリケーション	82
3.1.	エリア設定	82
3.1.1.	エリアの検出時の出力の設定	85
3.1.2.	エリアの検知時のロボット位置を位置変数に取り込む設定	89
3.1.3.	エリア検知時エラー検出の設定	92
3.2.	パレタイジング	95
3.2.1.	作業内容	95
3.2.2.	プログラミング例	97
3.2.3.	Command-Slave で実施例を実行する場合	99
3.3.	力制御	103
3.3.1.	作業内容	103
3.3.2.	プログラミング例	106
3.3.3.	パラメータの設定	111
3.3.4.	Command-Slave で実施例を実行する場合	118
4.	付録	123
4.1.	「ロボット Enable 中」=OFF のときに起動できる非動作系の FB	123
4.2.	ツールの説明	125
4.3.	ワークの説明	129

1. 準備

1.1. NJ用ファンクションブロックライブラリのダウンロード

弊社HPより、DENSWAVE_RC8_EIP_CmdSlv_Ver1.0.1.slrファイルをダウンロードします。

STEP 1:

下記URLから、デンソーロボットHPへアクセスします。

<https://www.denso-wave.com/ja/robot/index.html>

The screenshot shows the Denso Wave website homepage. At the top, there's a navigation bar with links for Global, 企業情報 (Corporate Information), 採用情報 (Recruitment Information), お問い合わせ (Contact Us), 自動認識 (Automatic Recognition), ロボット (Robot), and コントローラ (Controller). Below the navigation bar is a search bar with a '検索' (Search) button. The main content area features a banner for the International Robot Exhibition 2013, which took place from November 6 to 9, 2013, at Tokyo Big Sight. The banner includes the text 'meets your needs' and '[2013 国際ロボット展]'. To the right of the banner is a red-bordered box labeled 'DENSO ROBOT MEMBER SITE' containing fields for ID (メールアドレス) and PW (Password), a 'ログイン' (Login) button, and links for 'パスワードを忘れた方' (Forgot Password) and '新規会員登録' (New Member Registration). On the left side of the main content area, there are two boxes: 'お知らせ' (Announcements) and '更新情報' (Update Information). The 'お知らせ' box lists three items: 2014.10.17 (Denso Wave and Denso co-developed a surgical robot arm), 2014.10.16 (Participated in the 16th China International Industrial Expo), and 2014.10.03 (Participated in the 2014 Monozukuri Fair). The '更新情報' box lists one item: 2014.10.09.

STEP 2:

画面右側のメンバーサイトへログインします。

パスワードをお持ちでない方は、新規会員登録をしてください。

The screenshot shows the Denso Robot Member Site login page. The page title is 'DENSO ROBOT MEMBER SITE'. It contains a message: '取扱説明書、ソフトウェアのダウンロードなどの技術的なお問い合わせははこちら'. Below this is a form with two input fields: 'ID: メールアドレス' (ID: Email Address) and 'PW:' (Password). Both fields are highlighted with a red border. Below the fields is a 'ログイン' (Login) button. At the bottom of the page are links for 'パスワードを忘れた方' (Forgot Password) and '新規会員登録' (New Member Registration).

STEP 3: メンバーサイトログイン後、画面右側メニューの「ダウンロード」をクリックします。



STEP 4:

ダウンロードページの一番下にある「その他」をクリックします。

○ ダウンロード

カタログ、CADデータ・取扱説明書・アプリケーションソフトなどをダウンロードできます。

▶ カタログ

デンソーロボットのカタログ(PDFデータ)をご提供します。

▶ CADデータ

デンソーロボットの製品外形CADデータをご提供します。
(ダウンロードには会員登録が必要です。会員様のご登録内容により、ご覧いただけるコンテンツが異なります。)

▶ 取扱説明書

デンソーロボットの取扱説明書をご提供します。
(ダウンロードには会員登録が必要です。会員様のご登録内容により、ご覧いただけるコンテンツが異なります。)

▶ アプリケーションソフト

デンソーロボットのアプリケーションソフトをご提供します。
(ダウンロードには会員登録が必要です。会員様のご登録内容により、ご覧いただけるコンテンツが異なります。)

▶ D-ROMOシステム

デンソーロボットのD-ROMOシステムをご提供します。
(ダウンロードには会員登録が必要です。会員様のご登録内容により、ご覧いただけるコンテンツが異なります。)

▶ パーツリスト

デンソーロボットのパーツリストをご提供します。
(ダウンロードには会員登録が必要です。会員様のご登録内容により、ご覧いただけるコンテンツが異なります。)

▶ その他

その他のダウンロードはこちら。
(ダウンロードには会員登録が必要です。会員様のご登録内容により、ご覧いただけるコンテンツが異なります。)

STEP 5:

「Command-Slave」をクリックします。



STEP 6:

>OMRON >>NJ の「活用ガイド」をダウンロードします。

また、必要に応じて、ファンクションブロック説明書やEtherNet/IP接続ガイド(RC8)もダウンロードします。



- OMRON
- >NJ
 - ・ファンクションブロックライブラリ
 - ・EtherNet/IP接続ガイド(RC8)
 - ・**活用ガイド**
 - ・Command-Slaveファンクションブロック説明書

STEP 7:

使用許諾内容をお読み頂き、内容についてご理解頂けた場合、下記確認事項の各項目にチェックし、「同意する」ボタンを押します。



STEP 8:

ダウンロードが開始されます。



■ダウンロード中...
ダウンロードが始まらない場合は、[こちら](#)をクリックしてください。



STEP 9:

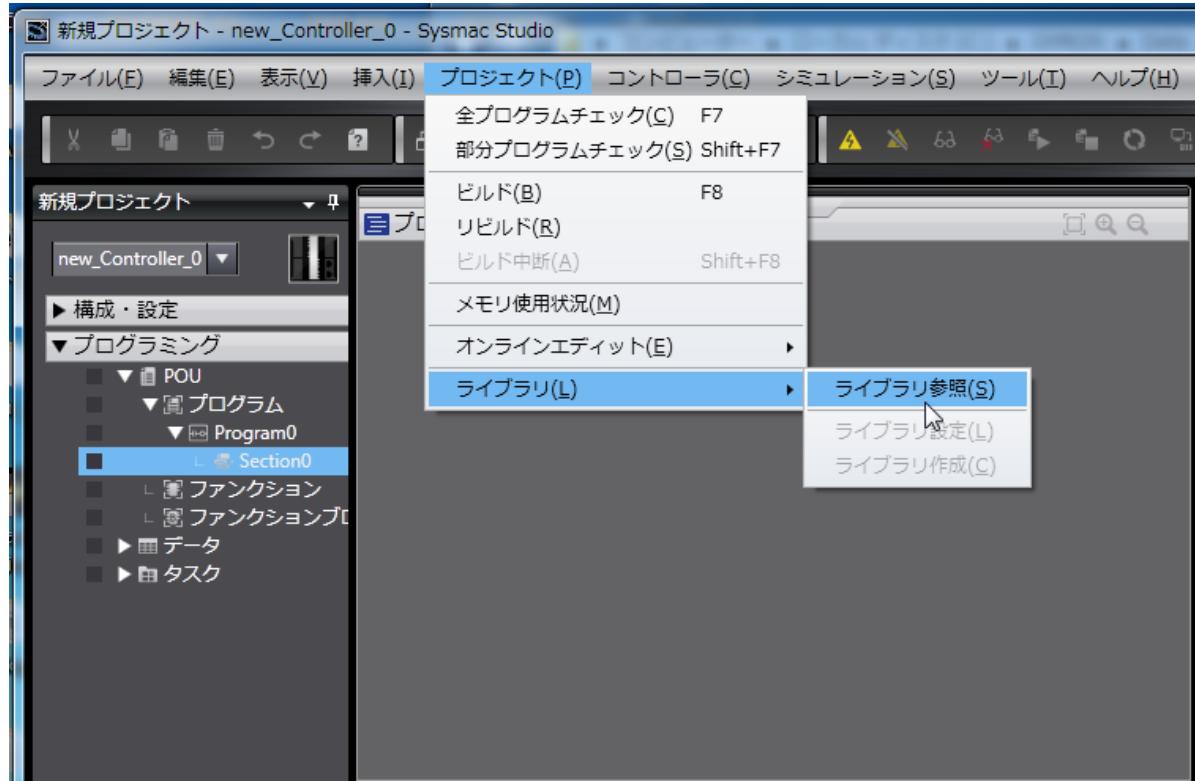
ダウンロードしたDENSOWAVE_RC8_EIP_CmdSlv_Ver1.0.1.slr ファイルは、Sysmac StudioがインストールされたPCの C:\OMRON\pData\lib に格納されます。



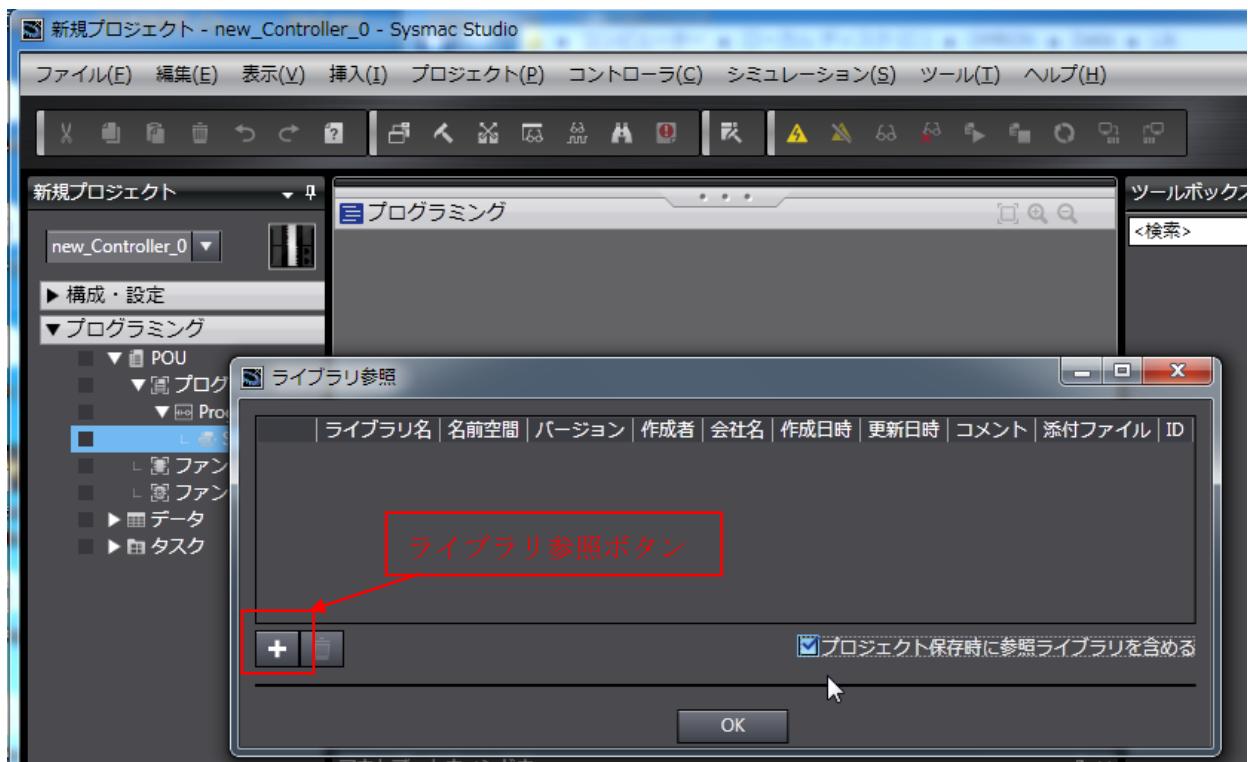
1.2. Sysmac Studio への FB ライブラリインポート

1.2.1. Sysmac Studio 上での FB ライブラリインポート

Sysmac Studio を起動後、挿入したいプロジェクトを選択し、メインメニューから「プロジェクト」 - 「ライブラリ」 - 「ライブラリ参照」を選択します。

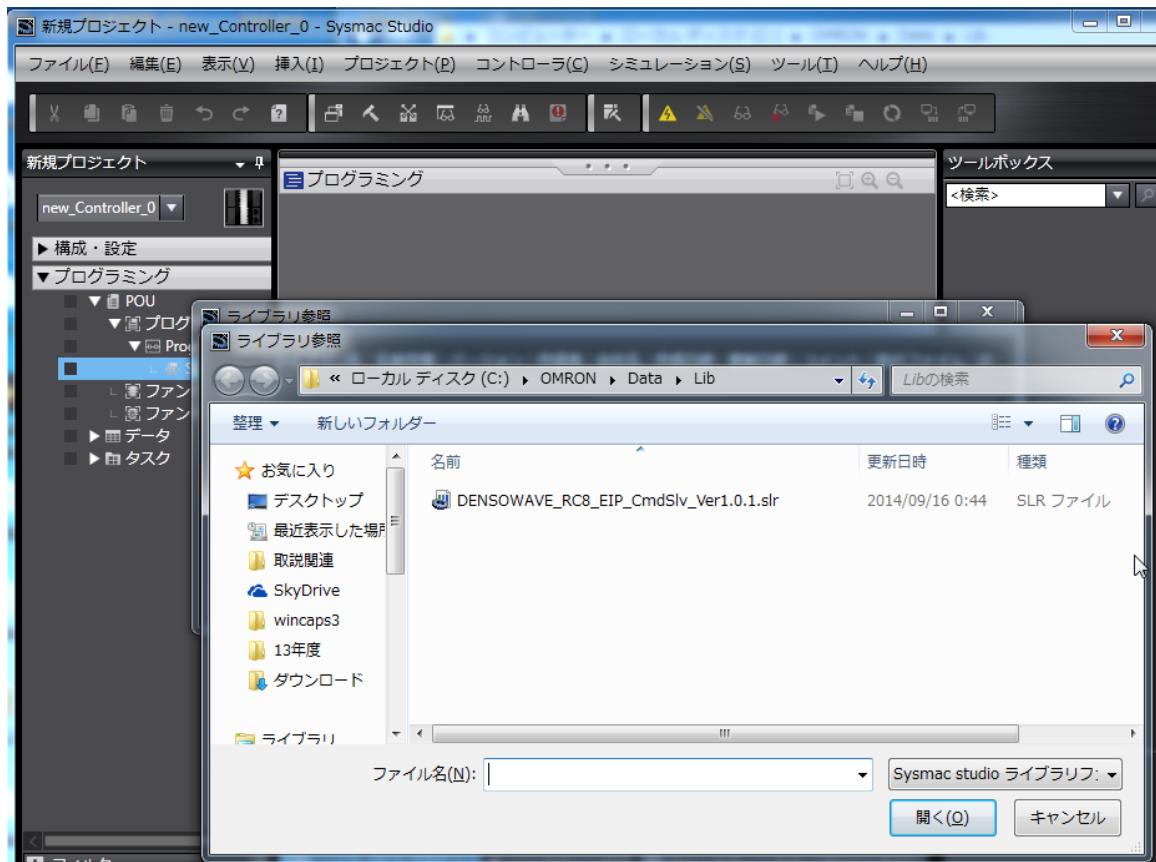


「プロジェクト保存時に参照ライブラリを含める」のチェックボックスを有効にし、「ライブラリ参照ボタン」を押します。

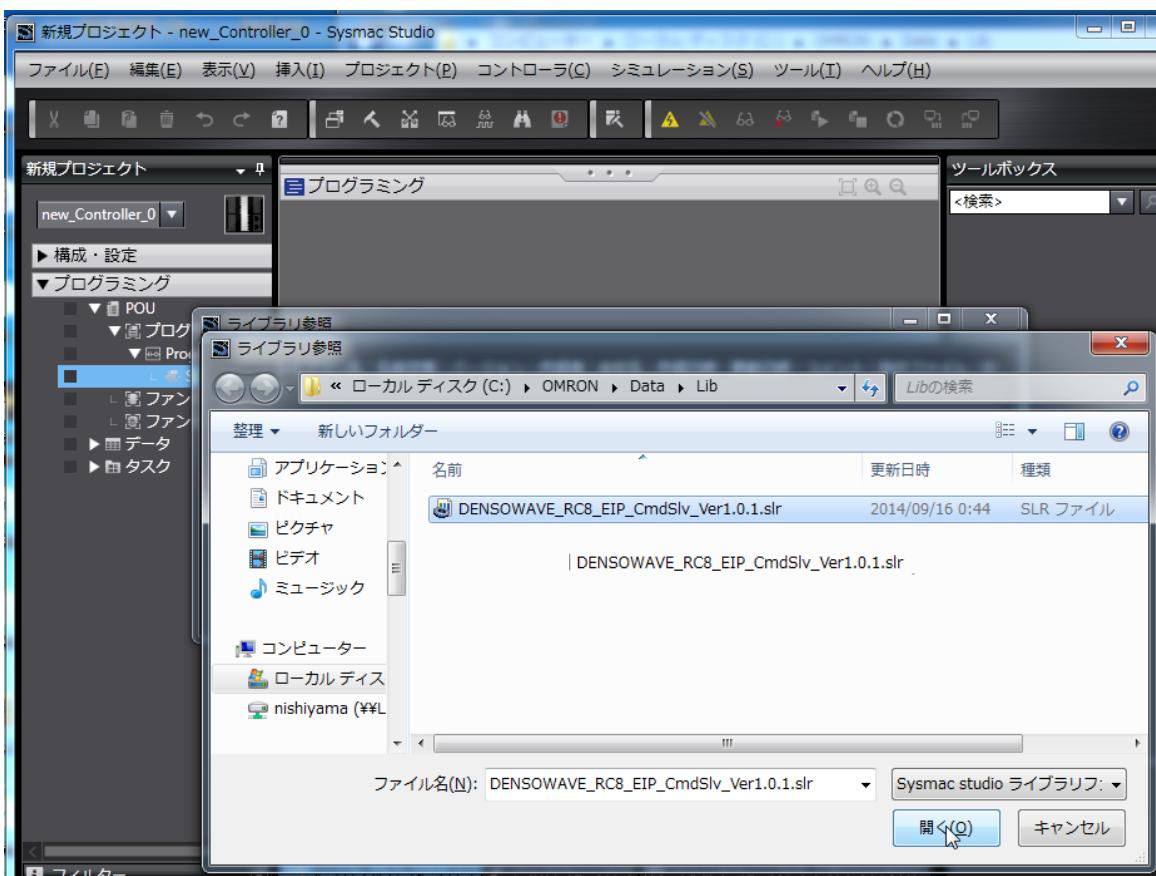


RC8Command-Slave 機能ガイド

ライブラリ参照ダイアログボックスが表示されます。

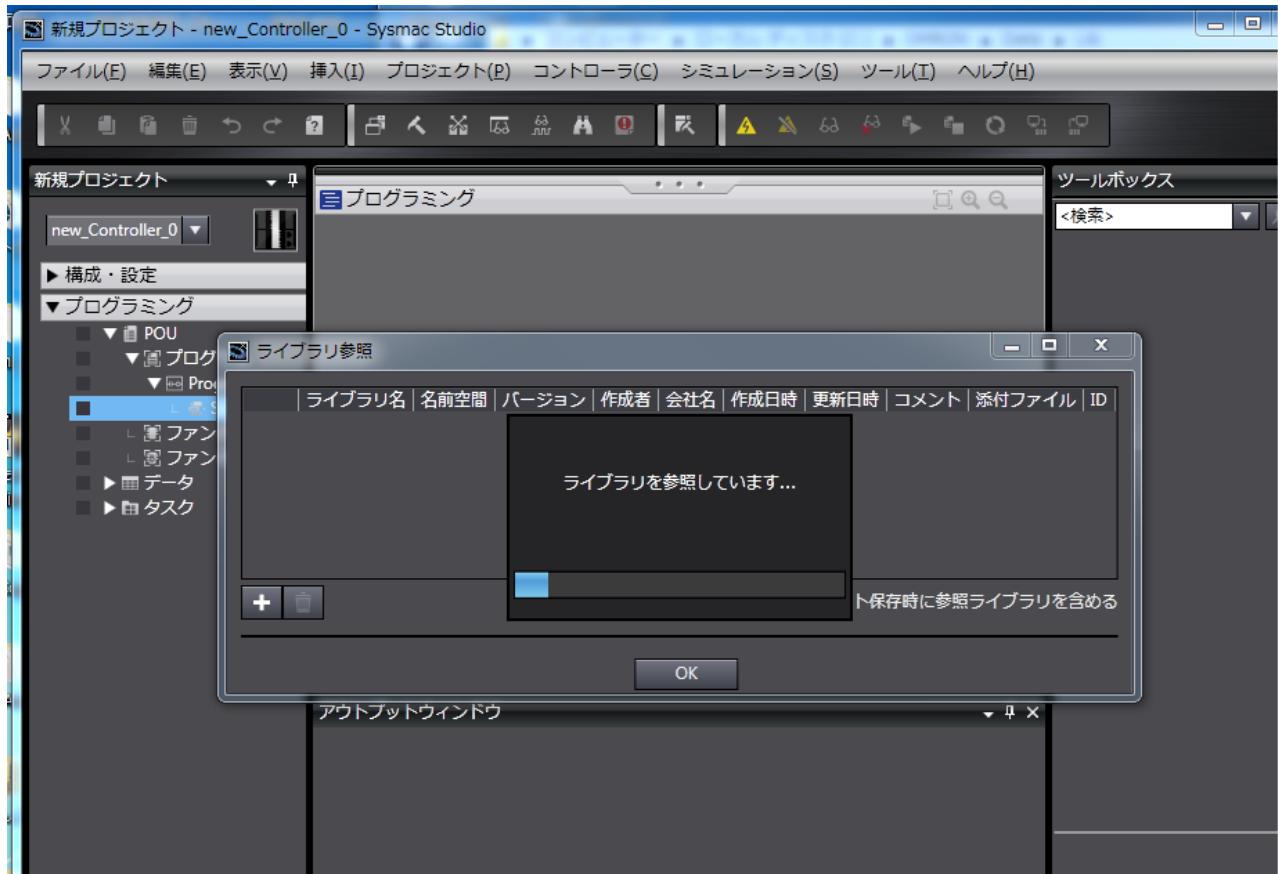


ダウンロードした DENSWAVE_RC8_EIP_CmdSlv_Ver1.0.1.slr ファイルを選択し、「開く」 ボタンをクリックします。

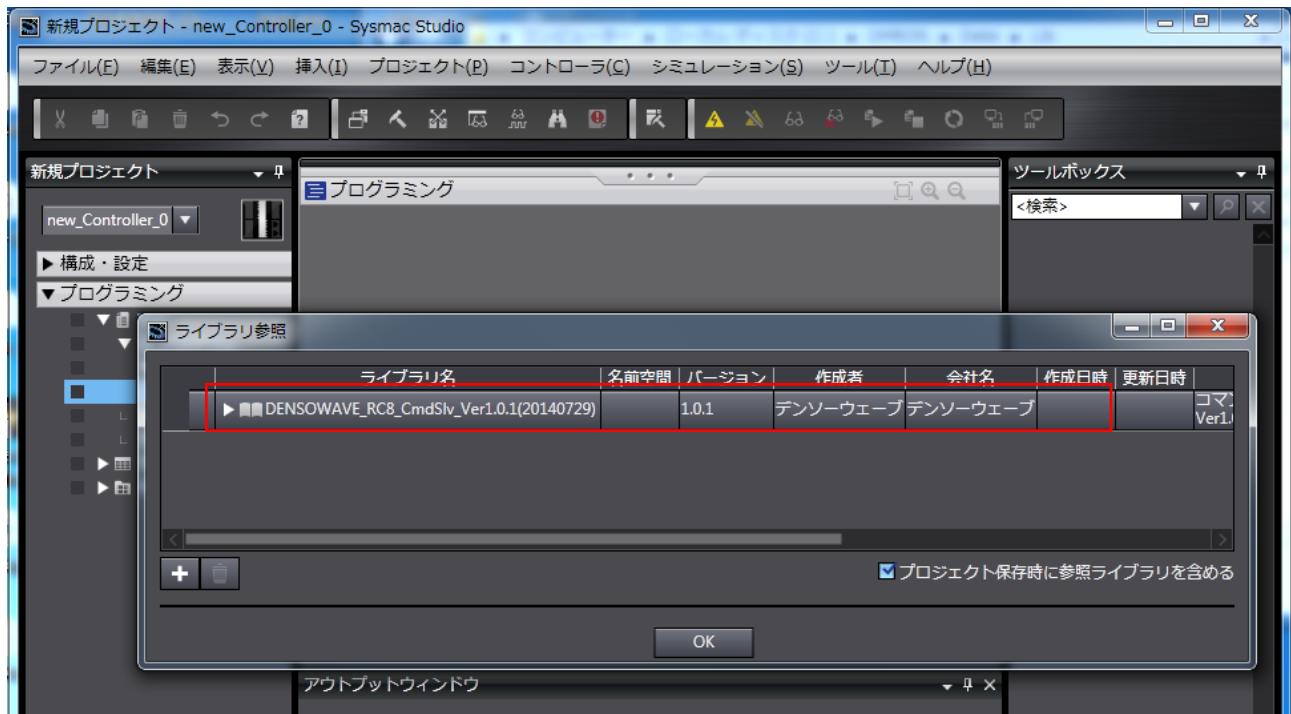


RC8Command-Slave 機能ガイド

ライブラリファイルの読み込みが開始されます。

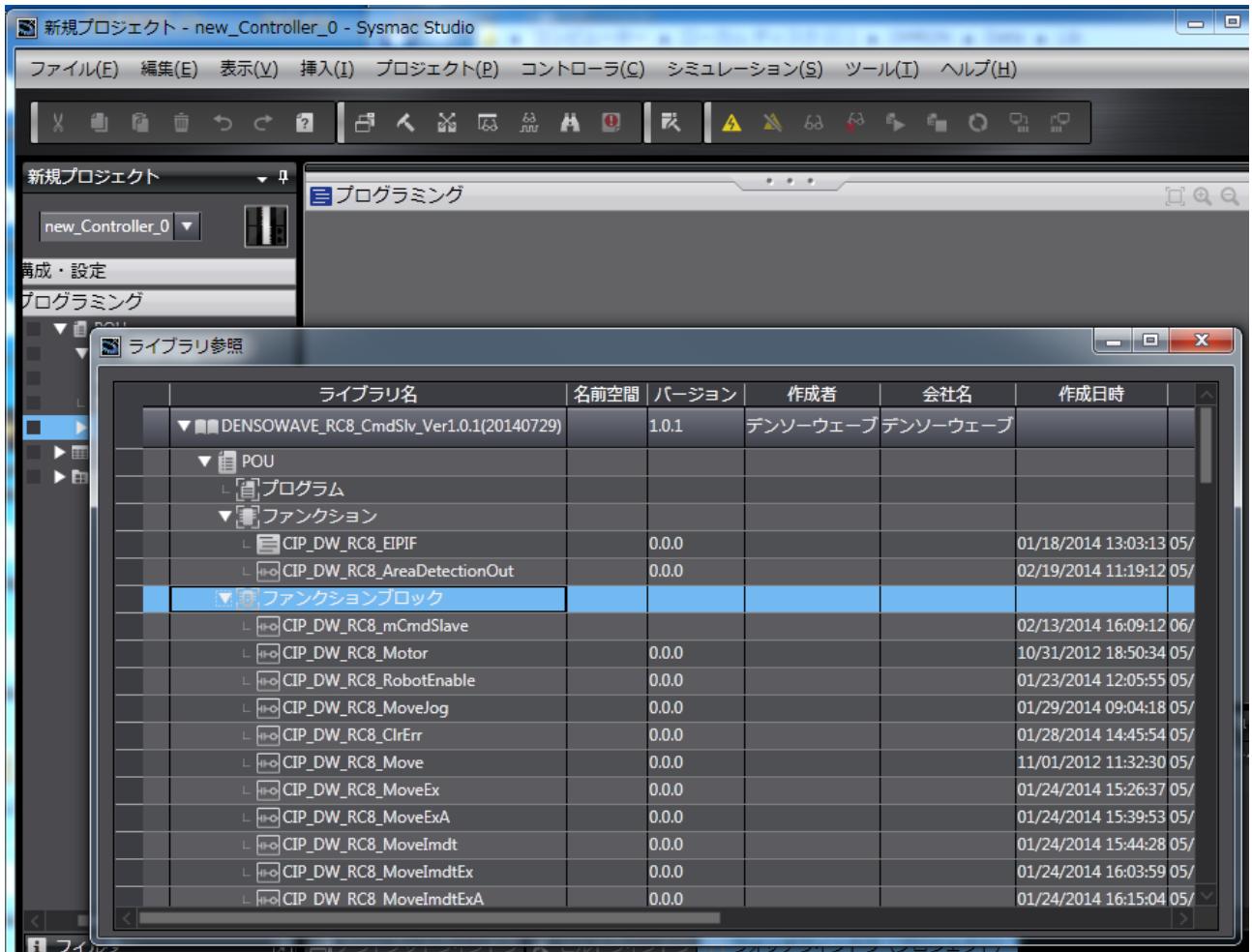


ライブラリファイルの読み込みが完了すると DENSOWAVE_RC8_EIP_CmdSlv_Ver1.0.1.slr が表示されます。



RC8Command-Slave 機能ガイド

「DENSWAVE_RC8_EIP_CmdSlv_Ver1.0.1.slr」 – 「POU」 – 「ファンクション」・「ファンクションブロック」をクリックすれば、プロジェクトに読み込まれたデンソーロボット用ファンクションブロックの一覧を見ることができます。これで、ライブラリ参照は完了です。



1.3. RC8Command-Slave 機能の設定

Command-Slave 機能の設定方法について説明します。

1.3.1. Command-Slave 機能オプションの有効化

Command-Slave機能は、RC8ソフトウェアオプションの一つです。Command-Slave機能を利用するには、RC8ロボットコントローラの無償オプションライセンスを有効化し、かつCommand-Slave起動フラグをONにする必要があります。また、ロボットの起動権設定は工場出荷時の「I/O」にしておく必要があります。Command-Slaveオプションライセンスの有効化は、ティーチングペンドントにて行います。ライセンスを有効化するには、弊社HPより無償オプションライセンスのパスワードを入手する必要があります。以下に、その設定手順を示します。

STEP 1 :

ティーチングペンドントのトップ画面から[F6 設定] → [F1 ログイン] でユーザレベル設定画面へ移動します。



STEP 2 :

ユーザレベルが「プログラマ」以上でないと、オプションライセンス追加が出来ません。もしユーザレベルがオペレータの場合、画面上で「プログラマ」を選択し、暗証番号[5596045]を入力し、ユーザレベルを変更します。



RC8Command-Slave 機能ガイド



STEP 3 :

「設定」画面から「F8 : オプション」 – 「F1 : 機能拡張」で「オプション管理」画面へ移行します。



RC8Command-Slave 機能ガイド



STEP 4 :

[RC8 Command-Slave]が“有効”化されている場合、ライセンス入力をする必要はありません。

その場合、“1.2.2 Command-Slave起動フラグの設定”へ移行して下さい。

“無効”となっている場合、下図のSTEP5～7の手順で弊社HPより“無償ライセンスキーキー”を入手して下さい。

<https://www.denso-wave.com/ja/robot/index.html>



STEP 5 :

お手持ちのPCにて、弊社HPの“DENSO ROBOT MEMBER SITE”にログインし、“RC8無償ライセンス確認”を選択します。

**STEP 6 :**

RC8 無償ライセンス確認画面へ移行し、「製品名」 - 「シリアルNo」を入力します。

製品名：ドロップダウンリストから“機能拡張_Command-Slave”を選択します。

シリアルNo：ロボットコントローラのシリアル番号を入力します。シリアル番号はコントローラ上面に記載されています。

両方のパラメータ入力が完了した後 “>送信する”を押します。

<input checked="" type="radio"/> RC8 無償ライセンス確認	
<small>*は必須項目です。必ず入力(全て半角英数字)してください。</small>	
製品名*	機能拡張_Command-Slave
シリアルNo(RC8)*	*****
> 送信する	

【RC8 無償ライセンスとは】

RC8に搭載されているプロバイダや機能拡張の中で、無償で使用できるライセンスです。
ライセンスはティーチングペンドントから入力いただきます。

STEP 7 :

登録が完了すると、下記画面にてライセンスキーが表示されます。

RC8 無償ライセンス確認

製品名	機能拡張コマンドスレーブ
シリアルNO(RC8)	99X ****-****-*
ライセンスキー	81DC-****-****-****

STEP 8 :

STEP4の画面で「F6：追加」を押し、追加ライセンス入力画面に入り、取得したライセンスキーを入力します。キー入力後、“OK”キーを押します。



RC8 Command-Slave 活用ガイド

STEP 9 :

ライセンス入力が完了すると、「RC8 Command-Slave」の状態が“有効”となります。



これで、Command-Slaveオプションライセンスの有効化は完了です。

RC8 Command-Slave 活用ガイド

1.3.2. Command-Slave 起動フラグの有効化

Command-Slave起動フラグの有効化は、ティーチングペンダントにて行います。以下に、その設定手順を示します。

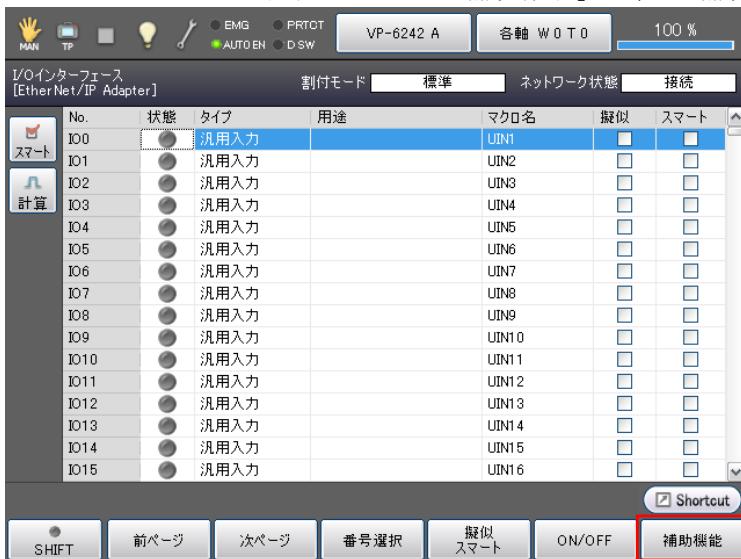
STEP 1 :

ティーチングペンダントのトップ画面から「F4： I/O」でI/Oインターフェース画面へ移行します。



STEP 2 :

I/Oインターフェース画面から「F6：補助機能」で、I/O補助機能画面へ移行します。



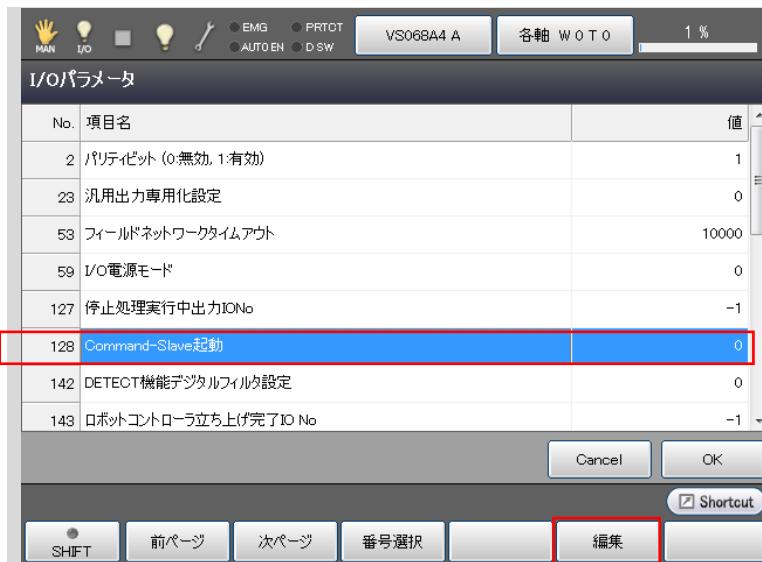
STEP 3 :

I/O補助機能画面から「F5 : I/Oパラメータ」で、I/Oパラメータ設定画面へ移行します。



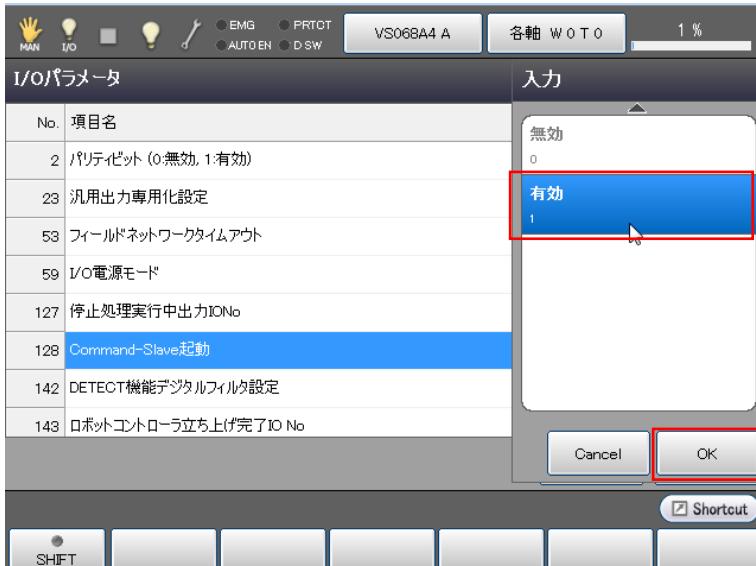
STEP 4 :

I/Oパラメータ設定画面にて「No.128 : Command-Slave起動」を選択し、「F5 : 編集」を押します。

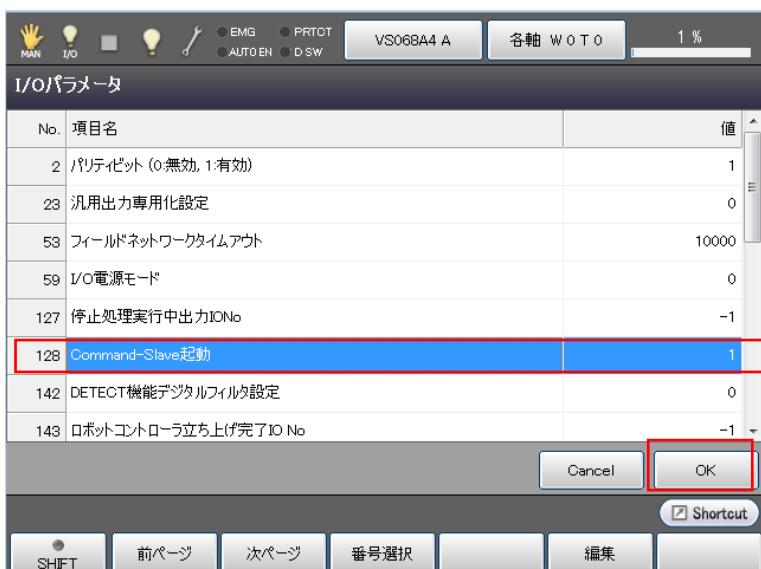


STEP 5 :

「Command-Slave起動」欄に 値 “1” を選択し、 “OK” を押します。

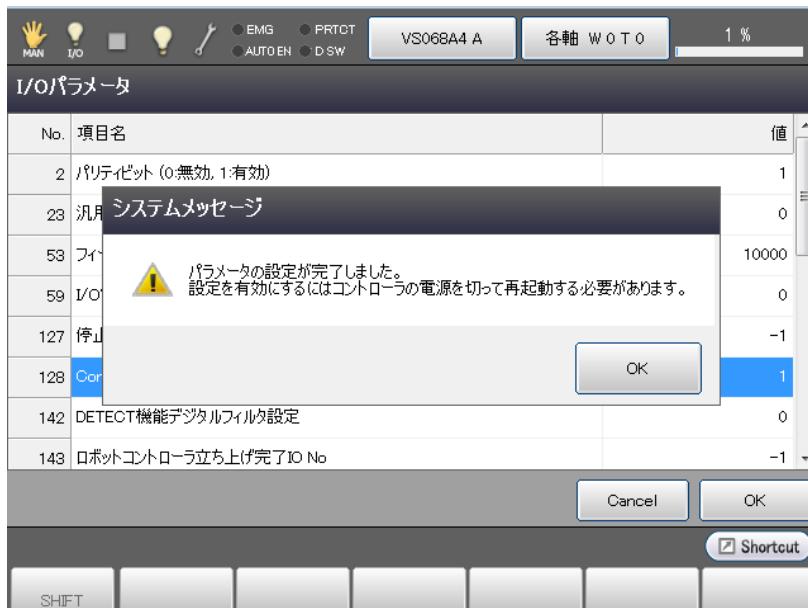
**STEP 6 :**

値 “1” が入力された事を確認し、 “OK” を押します。



STEP 7 :

パラメータ設定が完了すると、下記システムメッセージが表示されるので“OK”を押します。



これで、Command-Slave起動フラグの設定は完了です。

1.3.3. Command-Slave ポーリング周期の設定

Command-Slave機能は、一定周期毎にPLCからのコマンドと引数を監視しています。この監視周期をポーリング周期と呼び、工場出荷時は8msに設定されています。このポーリング周期は、設定パラメータとなっており、TP画面にて変更することができます。（工場出荷時は、8msに設定されています）

ポーリング周期を短くすることで、RC8のコマンドレスポンス時間の短縮が可能です。ただし、システム全体のレスポンスを短縮するためには、同時にEtherNet/IPのネットワークRPI（Request Packet Interval）やPLCのスキャン周期も考慮する必要があります。

以下に、ポーリング周期を8msから4msに変更する手順を示します。

STEP 1:

最初にユーザレベルを変更します。ティーチングペンドントのトップ画面から[F6：設定]を押します。ユーザレベルが「オペレータ」の場合、このパラメータはリードオンリーです。



STEP 2:

「F1:ログイン」を押します。



STEP 3:

ログイン画面にて、「プログラマ」を選択します。



STEP 4:

暗証番号「5596045」を入力します。



STEP 5:

「OK」を押すとプログラマになります

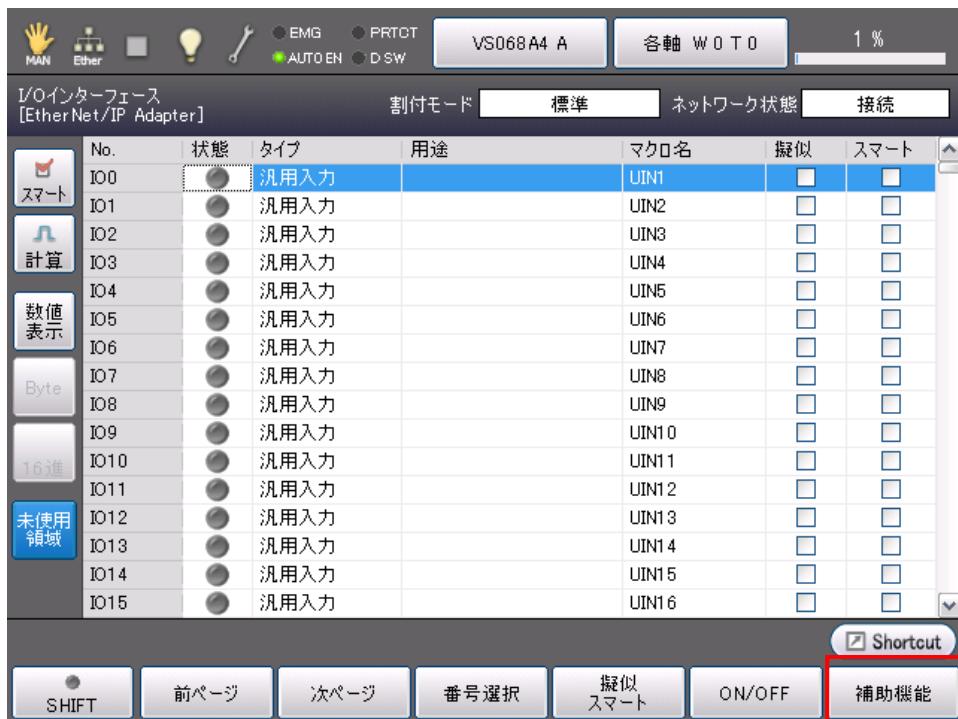
**STEP 6:**

Command-Slaveのポーリング周期を変更します。 「F4: I/O」を押します。



STEP 7:

「F6：補助機能」を押します。

**STEP 8:**

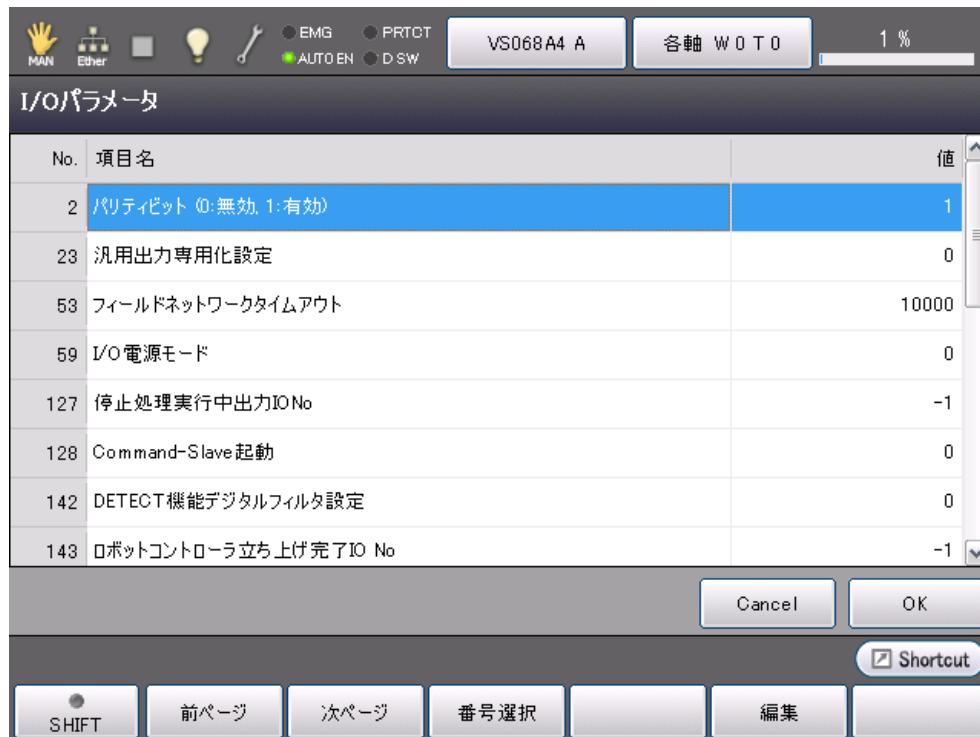
「F5：I/Oパラメータ」を押します。



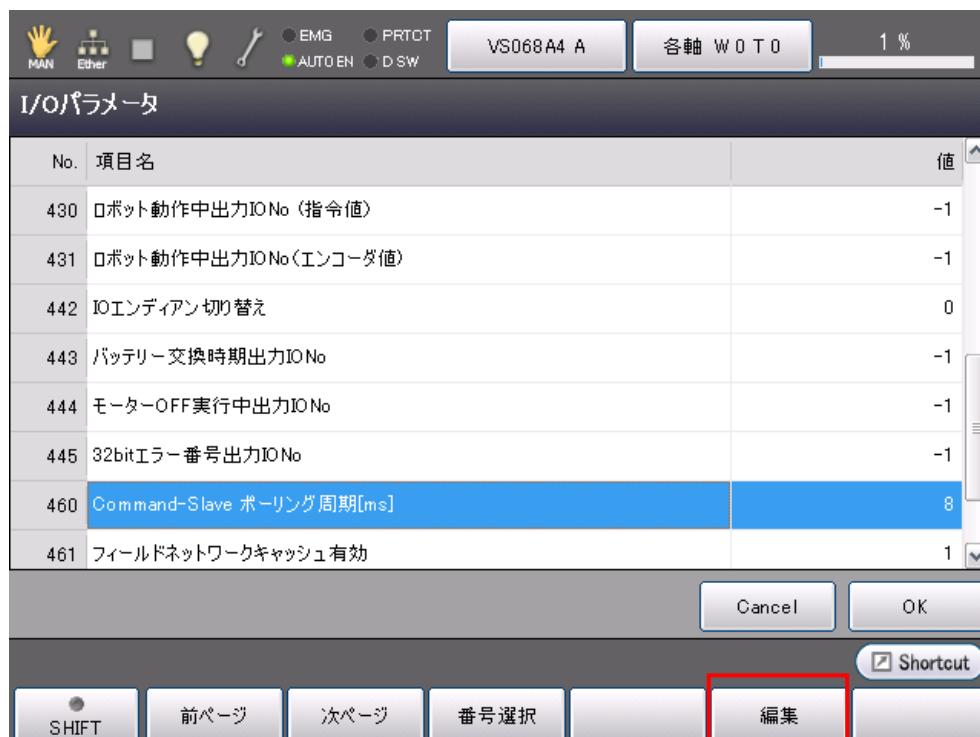
STEP 9:

IOパラメータ変更画面が出ます。

ジョグダイヤルにて「460:Command-Slaveポーリング周期[ms]」を選択します。

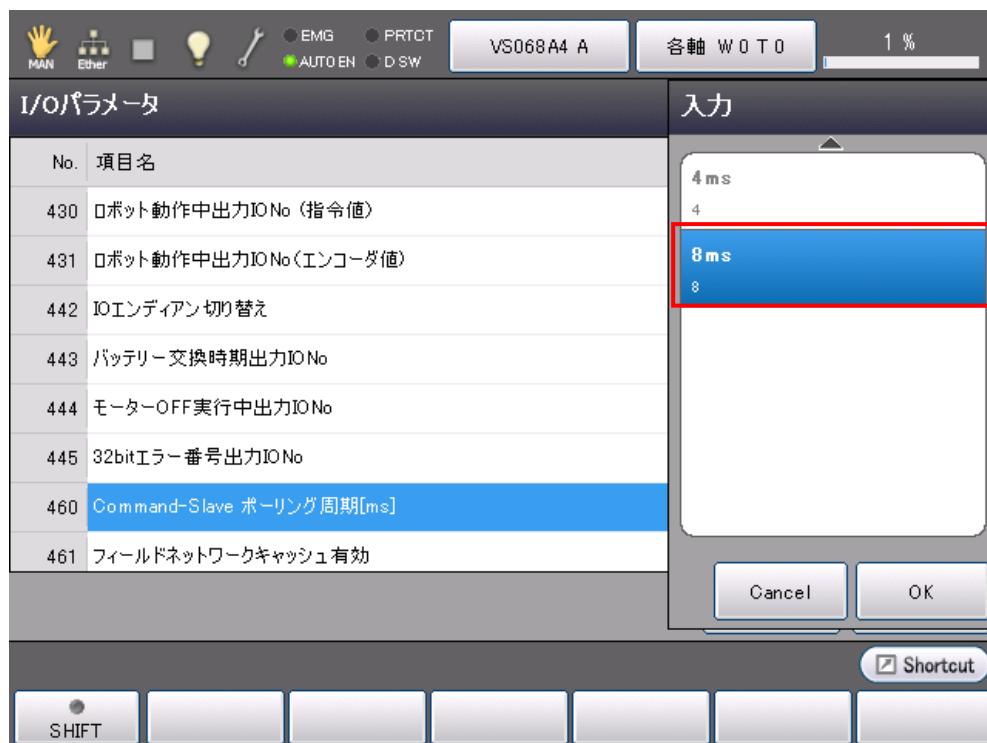
**STEP 10:**

「F5:編集」を押します。

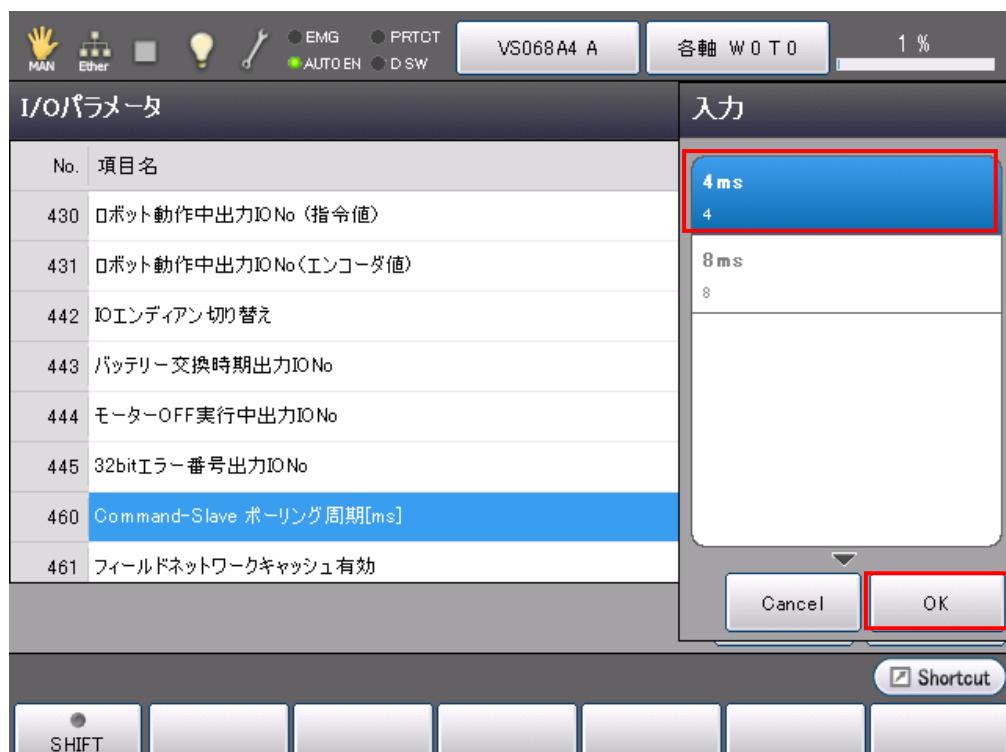


STEP 11:

工場出荷時は8msになっています。

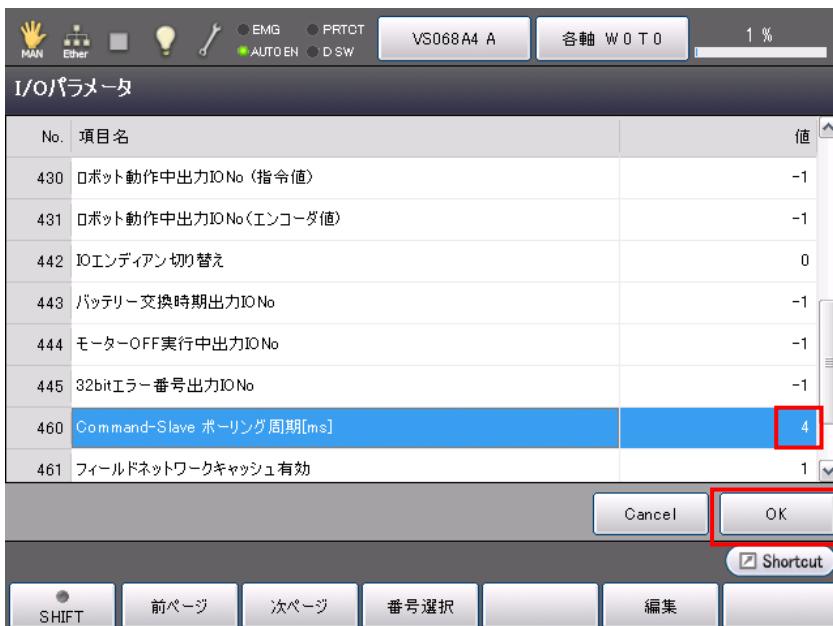
**STEP 12:**

4msを選択し、「OK」を押します。



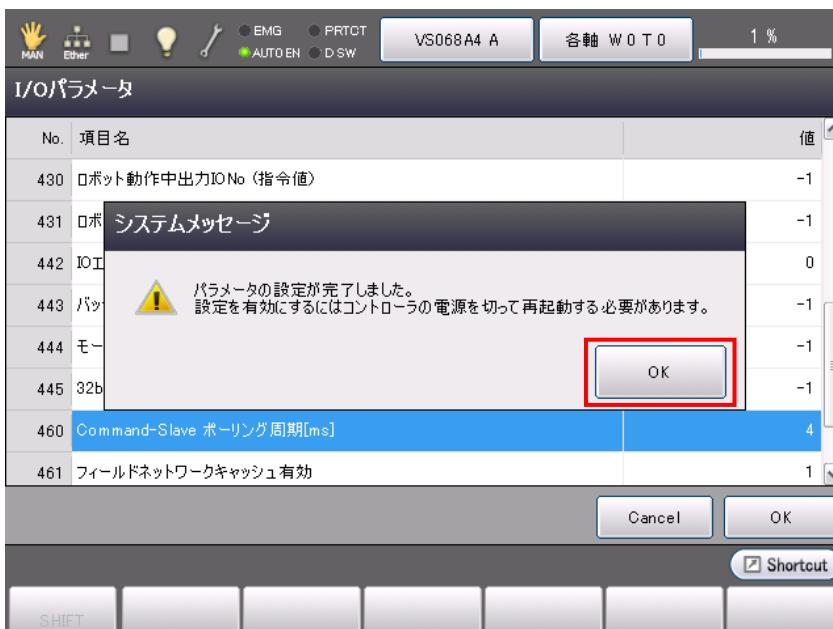
STEP 13:

4msになっていることを確認し、「OK」を押します。

**STEP 14:**

システムメッセージの「OK」を押します。

コントローラの電源を切って、再起動してください。



以上で、RC8 Command-Slaveポーリング周期の設定は完了です。

1.3.4. RC8 起動権の設定

起動権の設定は、ティーチングペンダントにて行います。Command-Slave機能を有効化する場合、起動権を「I/O」にします。以下に、その設定手順を示します。

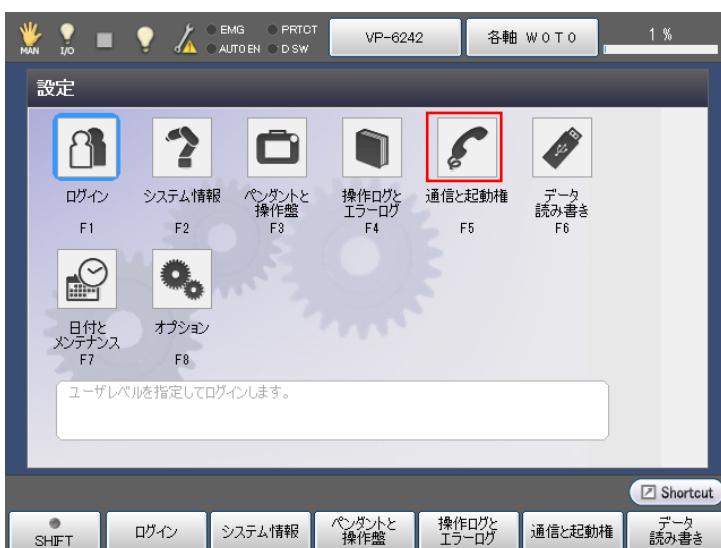
STEP 1 :

ティーチングペンダントのトップ画面から「F6：設定」を押します。



STEP 2 :

「F5：通信と起動権」を押します。

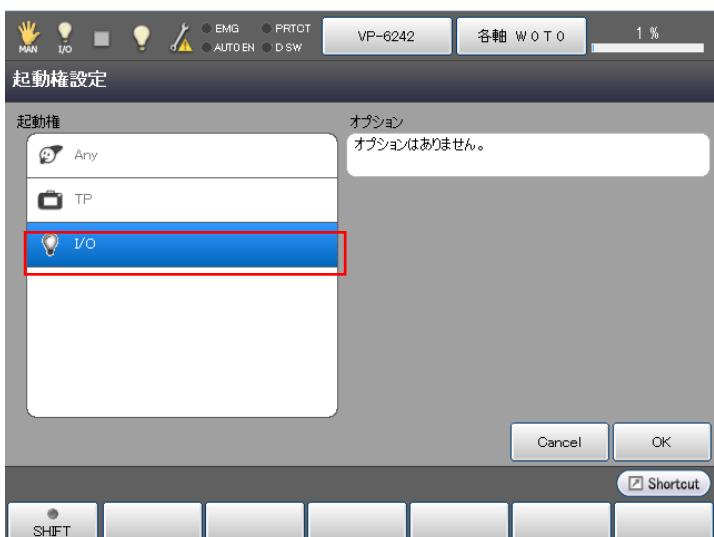


STEP 3 :

「F1：起動権」を押します。

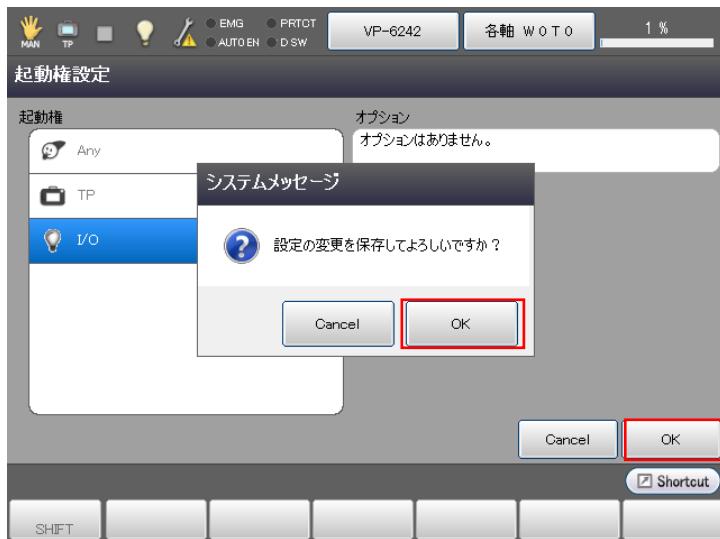
**STEP 4 :**

起動権設定画面にて、起動権が「I/O」であることを確認します。



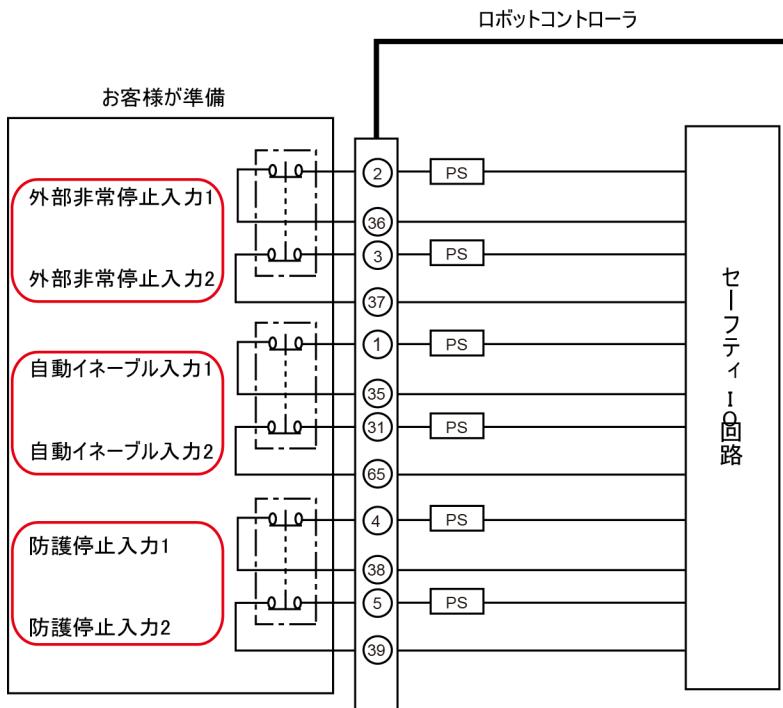
STEP 5 :

そうでなければ「I/O」に設定し “OK” キー、 “OK” キーを押します。



1.3.5. Command-Slave 外部入力信号(Mini I/O)の設定

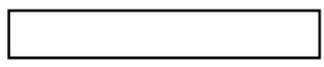
「外部非常停止入力」、「自動イネーブル入力」、「防護停止入力」を短絡します。



ケーブル側結合面より見た図

34

1



88

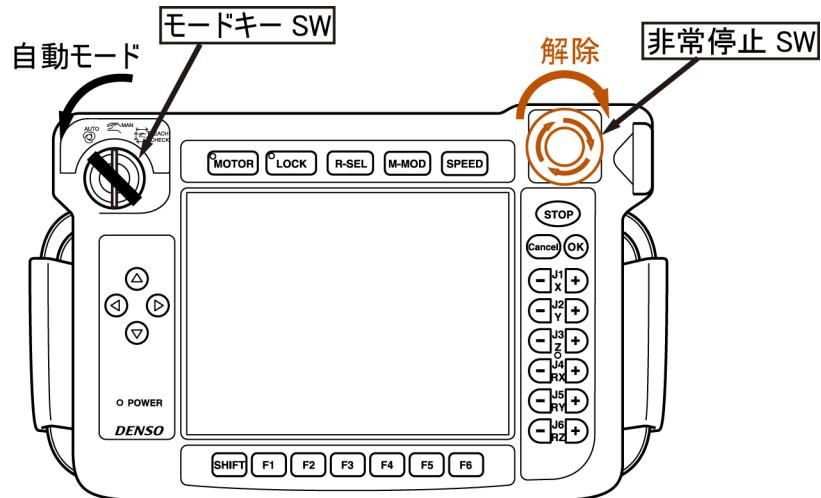
35

端子 No.	名称	ポート番号	線色	端子 No.	名称	ポート番号	線色
1	自動イネーブル入力 1-1	—	黒	35	自動イネーブル入力 1-2	—	桃
2	外部非常停止入力 1b-1	—	茶	36	外部非常停止入力 1b-2	—	桃
3	外部非常停止入力 2b-1	—	赤	37	外部非常停止入力 2b-2	—	桃
4	防護停止入力 1-1	—	橙	38	防護停止入力 1-2	—	桃
5	防護停止入力 2-1	—	黄	39	防護停止入力 2-2	—	桃
31	自動イネーブル入力 2-1	—	青	65	自動イネーブル入力 2-2	—	灰

1.3.6. ティーチングペンダントの設定

ティーチングペンダントは以下の状態で使用します。

モードキーSW	非常停止 SW
自動モード	解除状態



1.4. Command-Slave 機能の使い方

Command-Slave 機能の使い方について説明します

1.4.1. 複数のプログラムで構成する場合

■ グローバル変数

RC8 と NJ 間のタグデータリンク(EtherNet/IP)のデータをグローバル変数で扱います。

名称	データ型	ネットワーク公開	相手機器の割り当て
EIP_IN	ARRAY[0..125] OF DWORD	入力	RC8 入力データ (504Byte)
EIP_OUT	ARRAY[0..125] OF DWORD	出力	RC8 出力データ (504Byte)

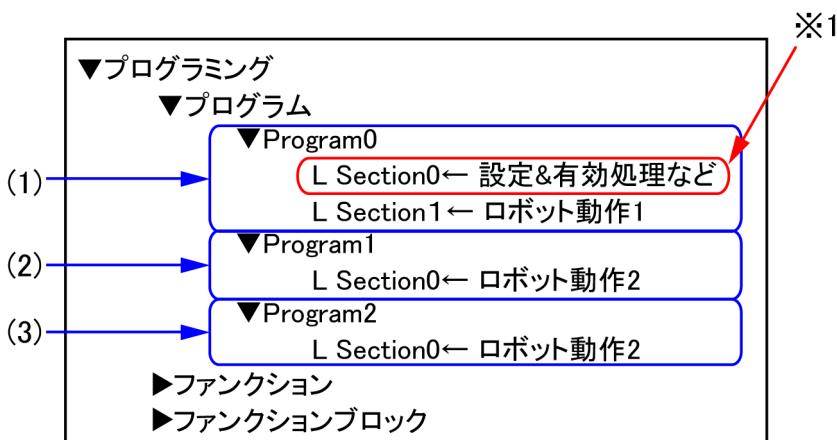
RC8Command-Slave 変数

- 複数のプログラム構成で Command-SlaveFB を使う場合は、グローバル変数を使用します。
- 一つのプログラムのみで Command-SlaveFB を使う場合は、ローカル変数も使用可能です。

名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

■ プログラム構成

プログラム構成の例を以下に示します。



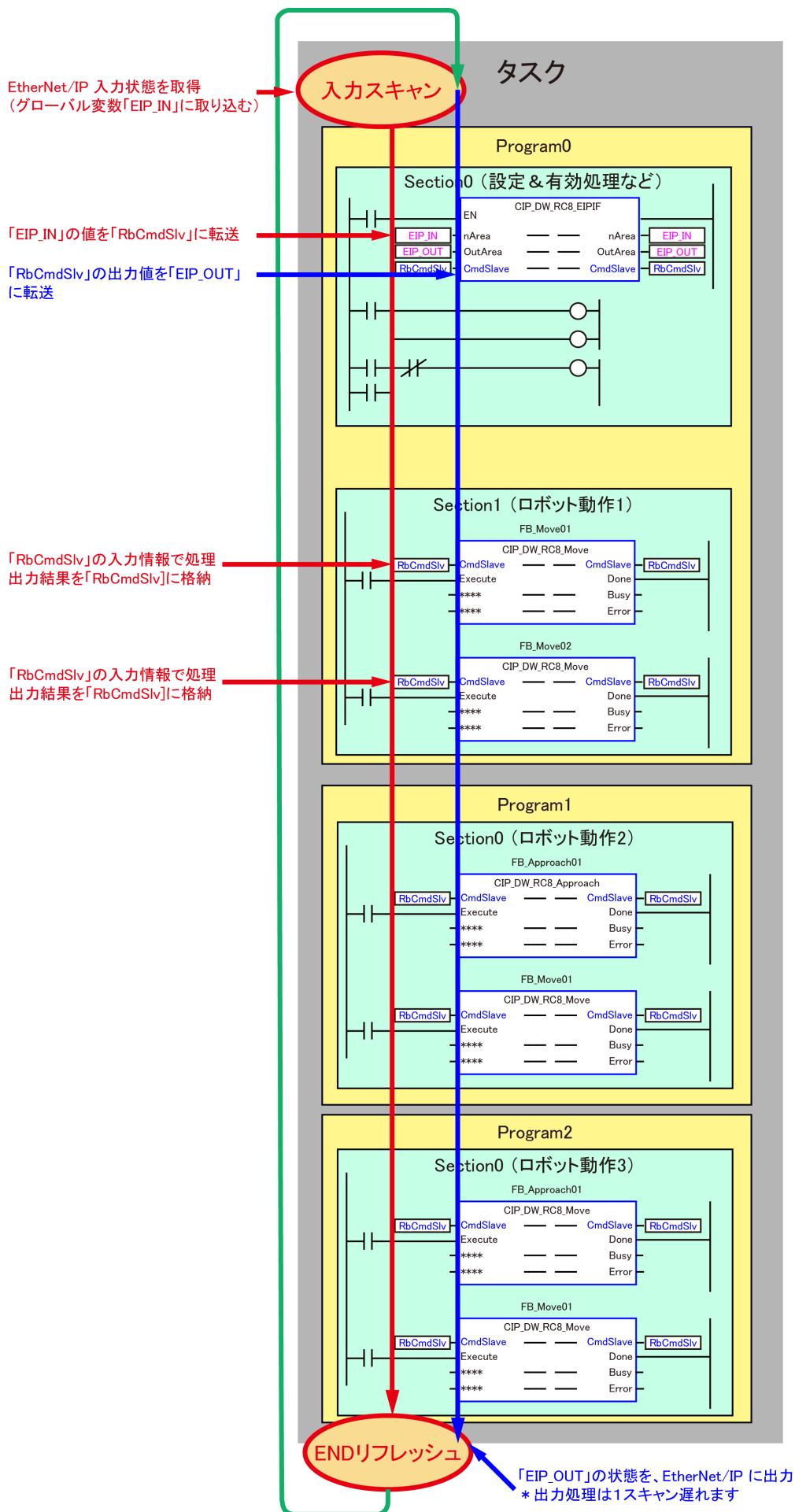
このプログラムは「Program0」、「Program1」、「Program2」で構成しています。

- 「Program0」は「Section0」(設定・有効処理など)と「Section1」(ロボット動作 1)で構成しています。
- 「Program1」は「Section0」(ロボット動作 2)で構成しています。
- 「Program2」は「Section0」(ロボット動作 3)で構成しています。

※1 「Section0」(設定&有効処理など)について

- 「設定&有効処理など」のセクションは他のセクション、他のプログラムより上で構成します。
- 設定&有効処理などのプログラム先頭に、CIP_DW_RC8_EIPIF(Command-Slave 変数変換ファンクション)をプログラムします。(フレッシュの関係で、他の Command-Slave 用 FB と FUN を使う前に RC8 からの入力状態を取得取り込んでおく必要があるため)

RC8Command-Slave 活用ガイド



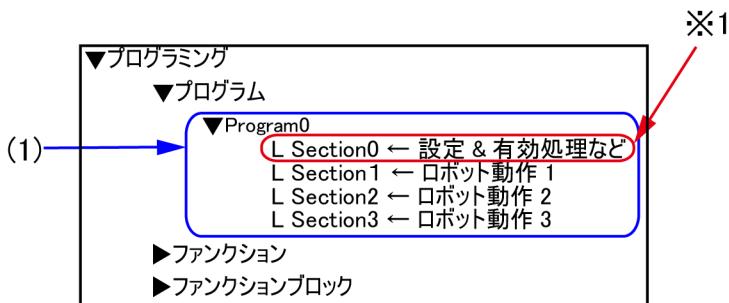
1.4.2. 1つのプログラムで構成する場合

■ グローバル変数

名称	データ型	ネットワーク公開	相手機器の割り当て
EIP_IN	ARRAY[0..125] OF DWORD	入力	RC8 入力データ (504Byte)
EIP_OUT	ARRAY[0..125] OF DWORD	出力	RC8 出力データ (504Byte)

■ プログラム構成

プログラム構成の例を以下に示します。



このプログラムは「Program0」で構成しています。

(1)セクションは「Section0」(設定&有効処理など)、「Section1」(ロボット動作1)、「Section2」(ロボット動作2)、「Section3」(ロボット動作3)で構成しています。

※1 「Section0」(設定&有効処理など)について

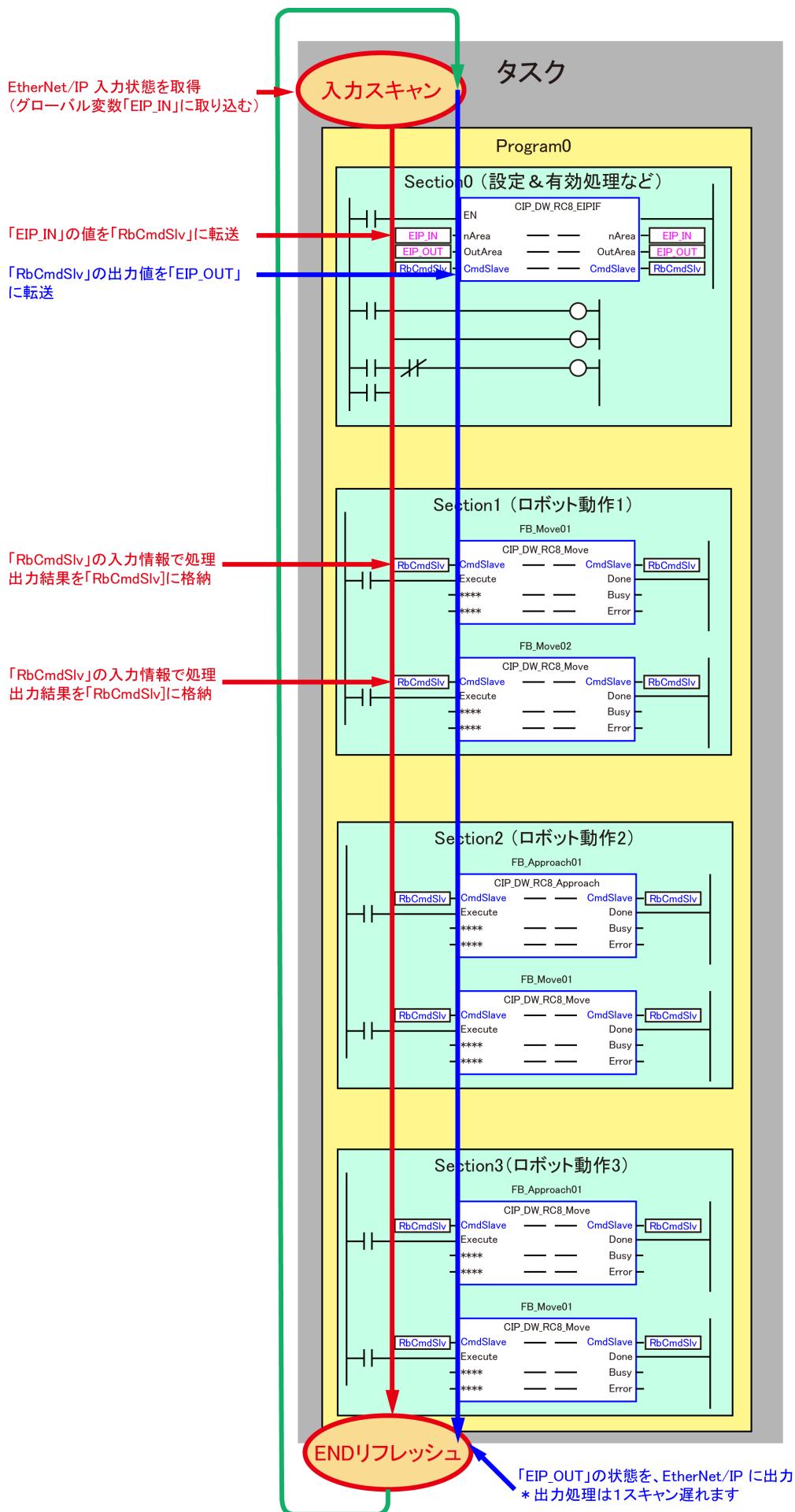
- 「設定&有効処理など」のセクションは他のセクション、他のプログラムより上で構成します。
- 設定&有効処理などのプログラム先頭に、CIP_DW_RC8_EIPIF(Command-Slave 変数変換アンクション)をプログラムします。(フレッシュの関係で、他の Command-Slave 用 FB と FUN を使う前に RC8 からの入力状態を取得取り込んでおく必要があるため)

■ ローカル変数

1つのプログラムの場合、RC8Command-Slave 変数はローカル変数で対応可能です。(グローバル変数扱いです。)

名称	データ型	初期値	割付先	保持
RbCmdSlv	sRC8_COMMAND_SLAVE	-----	-----	-----

RC8Command-Slave 活用ガイド



1.4.3. 構造体型データ

■RC8Command-Slave 変数

変数の名称はユーザが任意に決定します。下表では「RbCmdSlv」として説明しています。

複数のロボット制御の場合は、1台目「RbCmdSlv_01」、2台目「RbCmdSlv_02」などのロボットを区別しやすい名称にすると便利です。

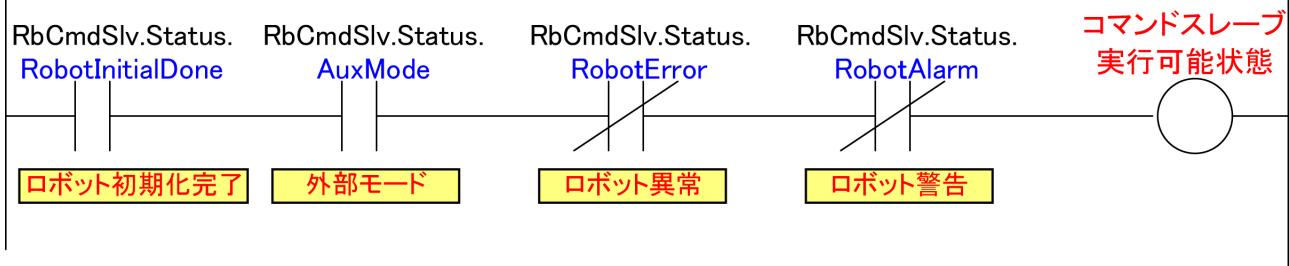
名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

名称	データ型	説明
RbCmdSlv	sRC8_COMMAND_SLAVE	RC8Command-Slave 変数
Status	sRC8_STATUS	RC8Command-Slave_ステータス関連
CmdBusy	BOOL	コマンド実行中
CmdDone	BOOL	コマンド実行完了
CmdPermit	BOOL	コマンド実行許可
MotorProcessBusy	BOOL	モータ処理実行中
MotorProcessPermit	BOOL	モータ処理実行許可
RobotEnableProcessBusy	BOOL	ロボット Enable 処理実行中
RobotEnableProcessPermit	BOOL	ロボット Enable 処理許可
CmdSlaveEnable	BOOL	Command-Slave 機能有効
RobotOnMoveCmd	BOOL	ロボット動作中信号出力(指令値)
ServoOn	BOOL	サーボ ON 中
RobotError	BOOL	ロボット異常
RobotAlarm	BOOL	ロボット警告
RobotInitialDone	BOOL	ロボット初期化完了
AuxMode	BOOL	外部モード
RobotEnable	BOOL	ロボット Enable 有効中(TakeArm)
MachineLock	BOOL	マシンロック状態
Result	ARRAY[1..18] OF DWORD	戻り値
RobotMoveSID	DWORD	ロボット動作の変化
ErrorNum	DWORD	エラー番号
AreaOut	ARRAY[0..31] OF BOOL	エリア検知出力
Cmd	sRC8_COMMAND	RC8Command-Slave_コマンド関連
CmdStrobe	BOOL	Command-Slave_ストローブ状態
MotorEnable	BOOL	「FB_Motor」の Enable 入力状態
RobotEnable	BOOL	「FB_RobotEnable」の Enable 入力状態
Cmd	Cmd ARRAY[0..37] OF DWORD	Command-Slave コマンド状態
TransNextOption	BOOL	前 FB_Next 処理判定

■変数のプログラム例

Command-Slave 実行可能状態を「RC8Command-Slave 変数」でプログラムした例を以下に示します。

//コマンドスレーブ実行可能状態



1.4.4. 列挙型データ

■変数の型

データ型			
Rb_POSETYPE			
	列挙子	値	変数の型
	P	257	ポジション型
	J	258	ジョイント型
	T	259	同次変換型

ロボット動作関連 FB の中で、座標指定が必要な FB を下表に示します。

列挙型対象の FB	
変数指定の FB (入力変数「PoseVarType」)	直接値指定の FB (入力変数「PoseType」)
Move	MoveImdt
MoveEx	MoveImdtEx
MoveExA	MoveImdtExA
MoveC	MoveCIImdt
MoveCE x	MoveCIImdtEx
MoveCExA	MoveCIImdtExA
Approach	ApproachImdt
ApproachEx	ApproachImdtEx
ApproachExA	ApproachImdtExA

変数指定の FB に変数 P10 を指定する場合

列挙子を指定する場合	列挙型データに続いて列挙子を指定する場合
<pre>***Instance ***_DW_RC8_Move -CmdSlave --- CmdSlave -Execute Done -Interpolation NextActionPermit -PassStartDisplacement Busy [P] PoseVarType Active [10] PoseVarNo Error -NextOption ErrorID -MotionOption ErrorIDEx -TimeOption</pre>	<pre>***Instance ***_DW_RC8_Move -CmdSlave --- CmdSlave -Execute Done -Interpolation NextActionPermit -PassStartDisplacement Busy [PoseVarType] PoseVarType Active [10] PoseVarNo Error -NextOption ErrorID -MotionOption ErrorIDEx -TimeOption</pre>

直接値指定の FB にポジション型(P)を指定する場合

列挙子を指定する場合	列挙型データに続いて列挙子を指定する場合
<pre>***Instance ***_DW_RC8_MoveImdt -CmdSlave --- CmdSlave -Execute Done -Interpolation NextActionPermit -PassStartDisplacement Busy [P] PoseType Active -MoveDistance Error -NextOption ErrorID -MotionOption ErrorIDEx -TimeOption</pre>	<pre>***Instance ***_DW_RC8_MoveImdt -CmdSlave --- CmdSlave -Execute Done -Interpolation NextActionPermit -PassStartDisplacement Busy [PoseType] PoseType Active -MoveDistance Error -NextOption ErrorID -MotionOption ErrorIDEx -TimeOption</pre>

RC8Command-Slave 活用ガイド

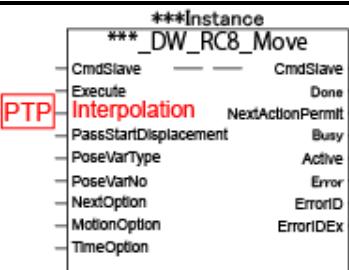
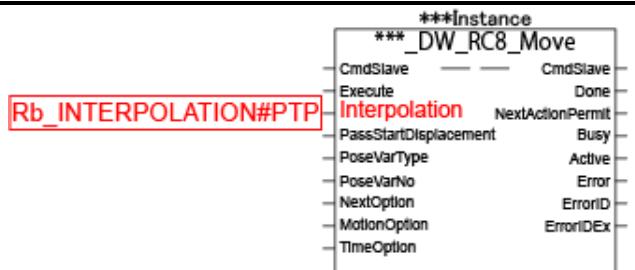
■補間方法

データ型			
Rb_INTERPOLATION			
	列挙子	値	補間方法
	PTP	0	動作時間が最短となるように補間
	CP	1	経路が直線上になるように補間

全てのロボット動作 FB が対象になります。

入力変数は「Interpolation」です。

補間方法として PTP 補間を指定する場合

列挙子を指定する場合	列挙型データに続いて列挙子を指定する場合
	

1.4.5. Command-Slave 変数の変換、ロボット Enable、モータ処理

■ グローバル変数

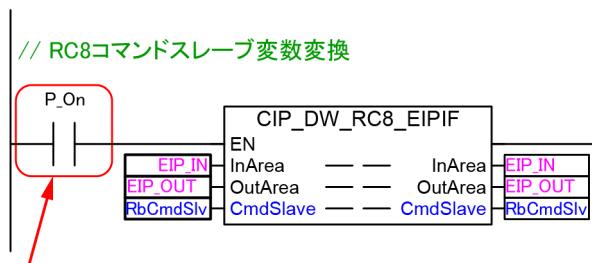
名称	データ型	ネットワーク公開	相手機器の割り当て
EIP_IN	ARRAY[0..125] OF DWORD	入力	RC8入力データ(504Byte)
EIP_OUT	ARRAY[0..125] OF DWORD	出力	RC8出力データ(504Byte)
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

■ ラダー図

(1) RC8Command-Slave 変数の変換

プログラムスキャンの先頭に配置します。

PLC 起動中は常に「CIP_DW_RC8_EIPIF」を実行します。※常時、EtherNet/IP の入出力とデータ交換が必要です。



「常時ON(P_On)」で常にファンクション「CIP_DW_RC8_EIPIF」を起動します。

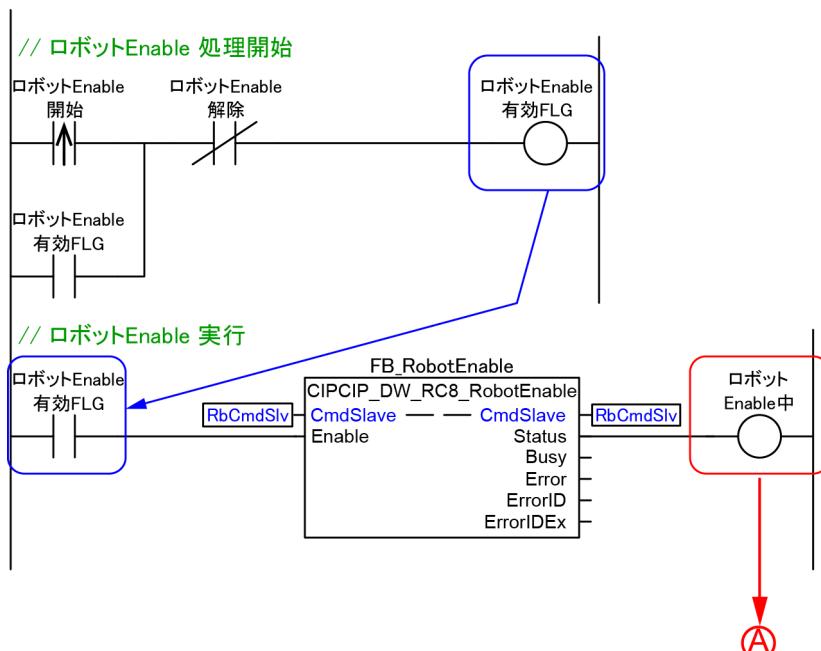
(2) ロボット Enable

RC8 に対してロボット制御を有効状態にします。

ロボットの非動作系の FB には、「ロボット Enable 中」= OFF のときに使用できないものがあります。

(「3.1. 「ロボット Enable 中」=OFF の状態で起動できる非動作系の FB」を参照)

全ての非動作系の FB は、「ロボット Enable 中」=ON での使用を推奨します。



RC8Command-Slave 活用ガイド

(3)モータ処理

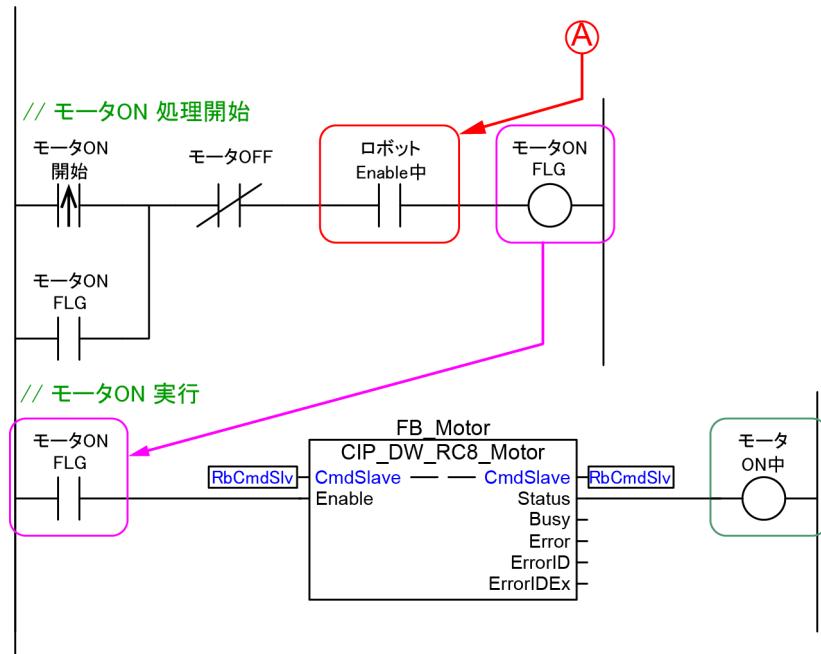
RC8に対してモータON/OFFを実行します。(モータONの条件に「ロボットEnable中」を入れます。)

「ロボットEnable中」=OFFの状態で、モータON処理を実行した場合、エラーになります。

モータON処理の条件に「ロボットEnable中」=ONを入れる事を推奨します。

ロボット動作系のFBは、「モータON中」=ONの状態で使用可能です。

「モータON中」を起動条件に入れたプログラムにします。



1.4.6. 変数書込

RC8の変数書込のしかたを下記の例で説明します。

例として変数(P型、J型、T型、V型、I型、F型)に値を書き込みます。

変数名	X	Y	Z	RX	RY	RZ	FIG
P11	400	310.5	200	180	0	180	5

変数名	1軸	2軸	3軸	4軸	5軸	6軸	7軸	8軸
J21	0	45	90	0	45	0	---	---

変数名	X	Y	Z	OX	OY	OZ	AX	AY	AZ	FIG
T11	400	200	300	0	1	0	0	0	-1	5

変数名	X	Y	Z
V1	100	200	300

変数名	値
I5	1234

変数名	値
F7	3.14

■ グローバル変数

名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdS1v	sRC8_COMMAND_SLAVE	---	無し

■ ローカル変数

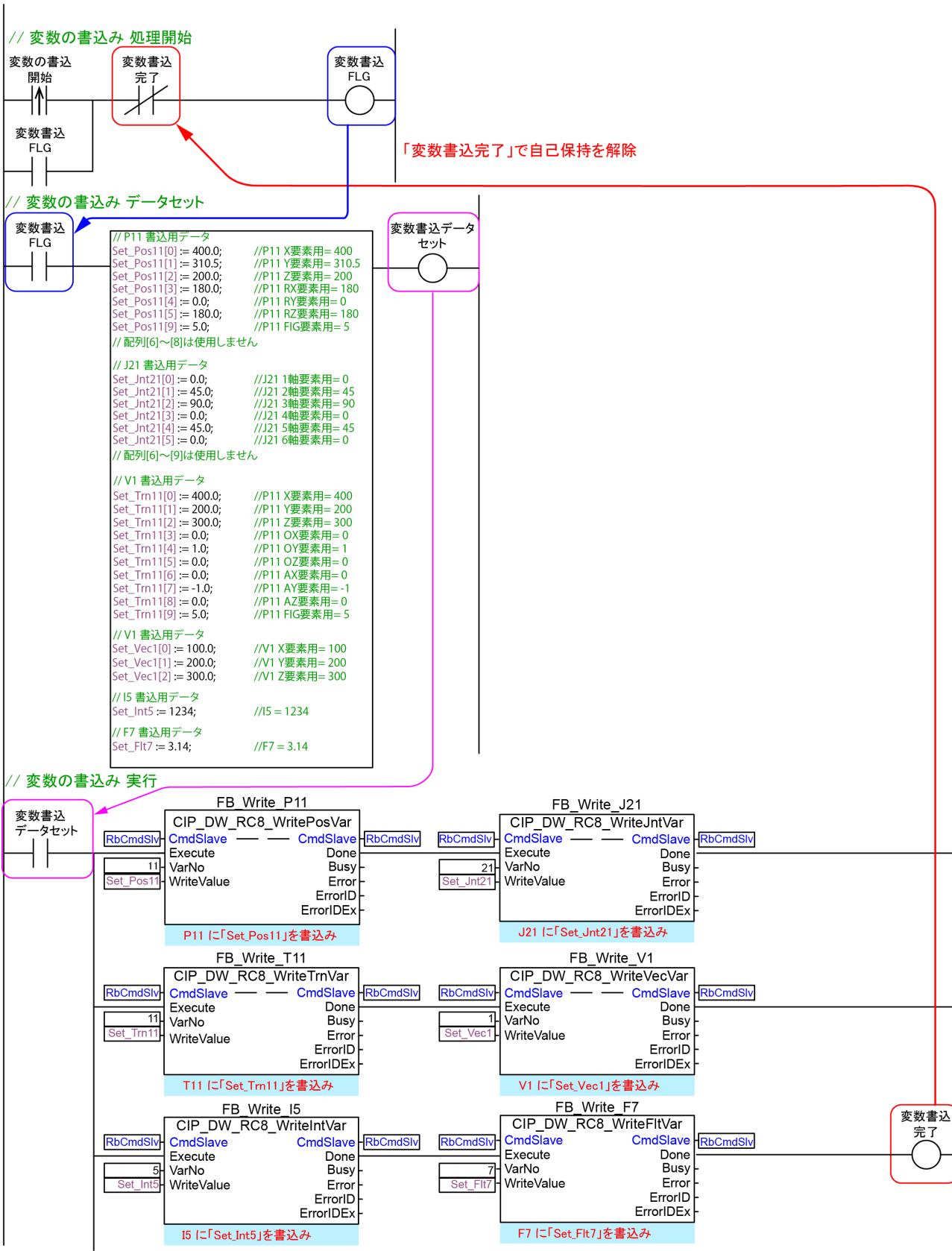
名称	データ型	備考
Set_Pos11	ARRAY[0..9] OF REAL	P11 書込用
Set_Jnt21	ARRAY[0..9] OF REAL	J21 書込用
Set_Trn11	ARRAY[0..9] OF REAL	T11 書込用
Set_Vec1	ARRAY[0..2] OF REAL	V1 書込用
Set_Int5	DINT	I5 書込用
Set_Flt7	REAL	F7 書込用

PLC で RC8 の変数を扱う場合は以下の点に注意してください。

- P型、J型、T型は REAL 型の 10 配列のデータで扱います。
- V型は REAL 型の 3 配列データで扱います。
- I型は DINT 型で扱います。
- F型は REAL 型で扱います。

RC8Command-Slave 活用ガイド

■ ラダーツ



RC8Command-Slave 活用ガイド

NOTE:P型変数の配列の構成は 6軸ロボットと4軸ロボットで異なります。下表を参照して下さい。

上図の例では6軸ロボットを想定して説明しています。

配列	要素	6軸ロボット位置データ	4軸ロボット位置データ
0	X	X 座標値[mm]	X 座標値[mm]
1	Y	Y 座標値[mm]	Y 座標値[mm]
2	Z	Z 座標値[mm]	Z 座標値[mm]
3	RX	X 軸回転量[deg]	180 ※固定値
4	RY	Y 軸回転量[deg]	0 ※固定値
5	RZ	Z 軸回転量[deg]	Z 軸回転量[deg]
6	未使用	-	-
7	未使用	-	-
8	未使用	-	-
9	FUG	形態(0 ~ 31)の32形態	FIG 形態(0、1、16、17)の4形態

1.4.7. 変数読込

RC8の変数読込のしかたを下記の例で説明します。

例として変数(P型、J型、T型、V型、I型、F型)に値を書き込みます。

変数名	X	Y	Z	RX	RY	RZ	FIG
P11	400	310.5	200	180	0	180	5

変数名	1軸	2軸	3軸	4軸	5軸	6軸	7軸	8軸
J21	0	45	90	0	45	0	---	---

変数名	X	Y	Z	OX	OY	OZ	AX	AY	AZ	FIG
T11	400	200	300	0	1	0	0	0	-1	5

変数名	X	Y	Z
V1	100	200	300

変数名	値
I5	1234

変数名	値
F7	3.14

■ グローバル変数

名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdS1v	sRC8_COMMAND_SLAVE	---	無し

■ ローカル変数

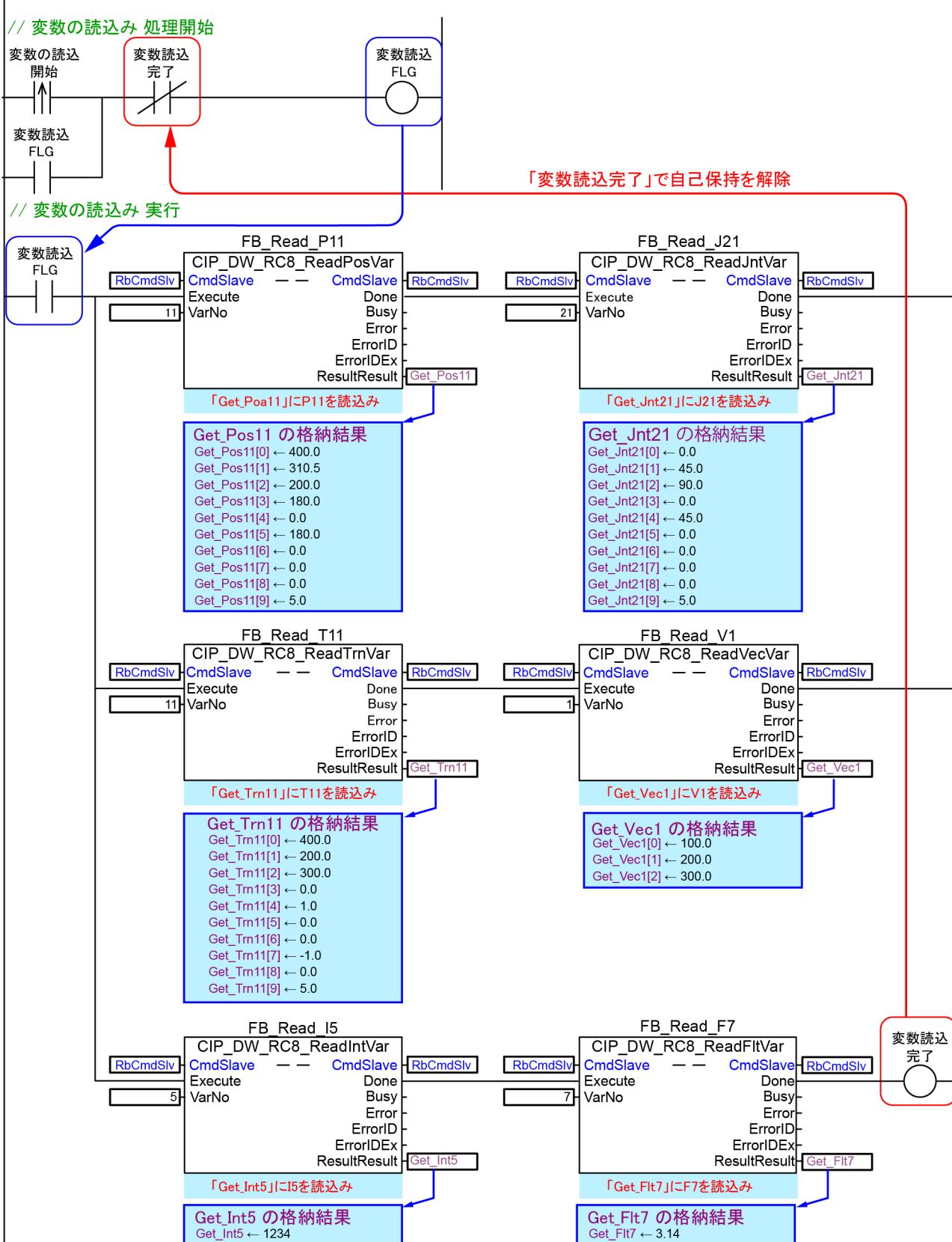
名称	データ型	備考
Get_Pos11	ARRAY[0..9] OF REAL	P11 読込用
Get_Jnt21	ARRAY[0..9] OF REAL	J21 読込用
Get_Trn11	ARRAY[0..9] OF REAL	T11 読込用
Get_Vec1	ARRAY[0..2] OF REAL	V1 読込用
Get_Int5	DINT	I5 読込用
Get_Flt7	REAL	F7 読込用

PLC で RC8 の変数を扱う場合は以下の点に注意してください。

- P型、J型、T型は REAL型の 10配列のデータで扱います。
- V型は REAL型の 3配列データで扱います。
- I型は DINT型で扱います。
- F型は REAL型で扱います。

RC8Command-Slave 活用ガイド

■ ラダー図



1.4.8. ツール設定

Tool1～Tool7 に以下の値を設定します。(ツール座標については「3.2.ツールの説明」参照)

「ロボット Enable」が実行中である事が必要です。※ラダー図(1)を参照。

設定値は一度 RC8 の P 型変数に書き込み、書込んだ P 型変数の番号を指定してツール設定を行います。

ツール番号	X	Y	Z	RX	RY	RZ	格納 P 型変数
1	-50	0	0	0	0	0	P21
2	0	50	0	0	0	0	P22
3	0	0	50	0	0	0	P23
4	-50	50	50	0	0	0	P24
5	0	0	50	45	0	0	P25
6	0	0	50	0	-45	0	P26
7	0	0	50	0	0	45	P27

■ グローバル変数

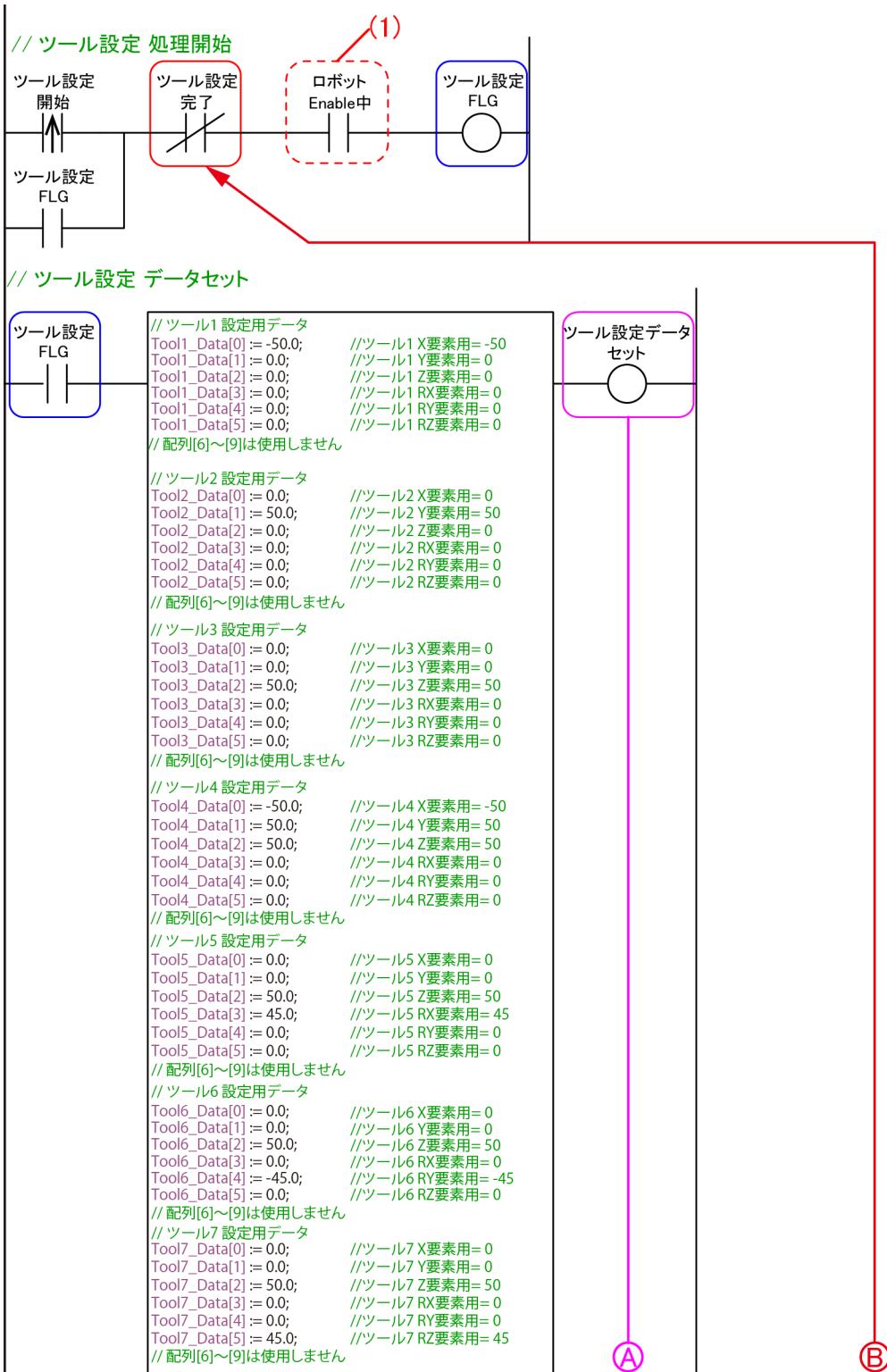
名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdS1v	sRC8_COMMAND_SLAVE	---	無し

■ ローカル変数

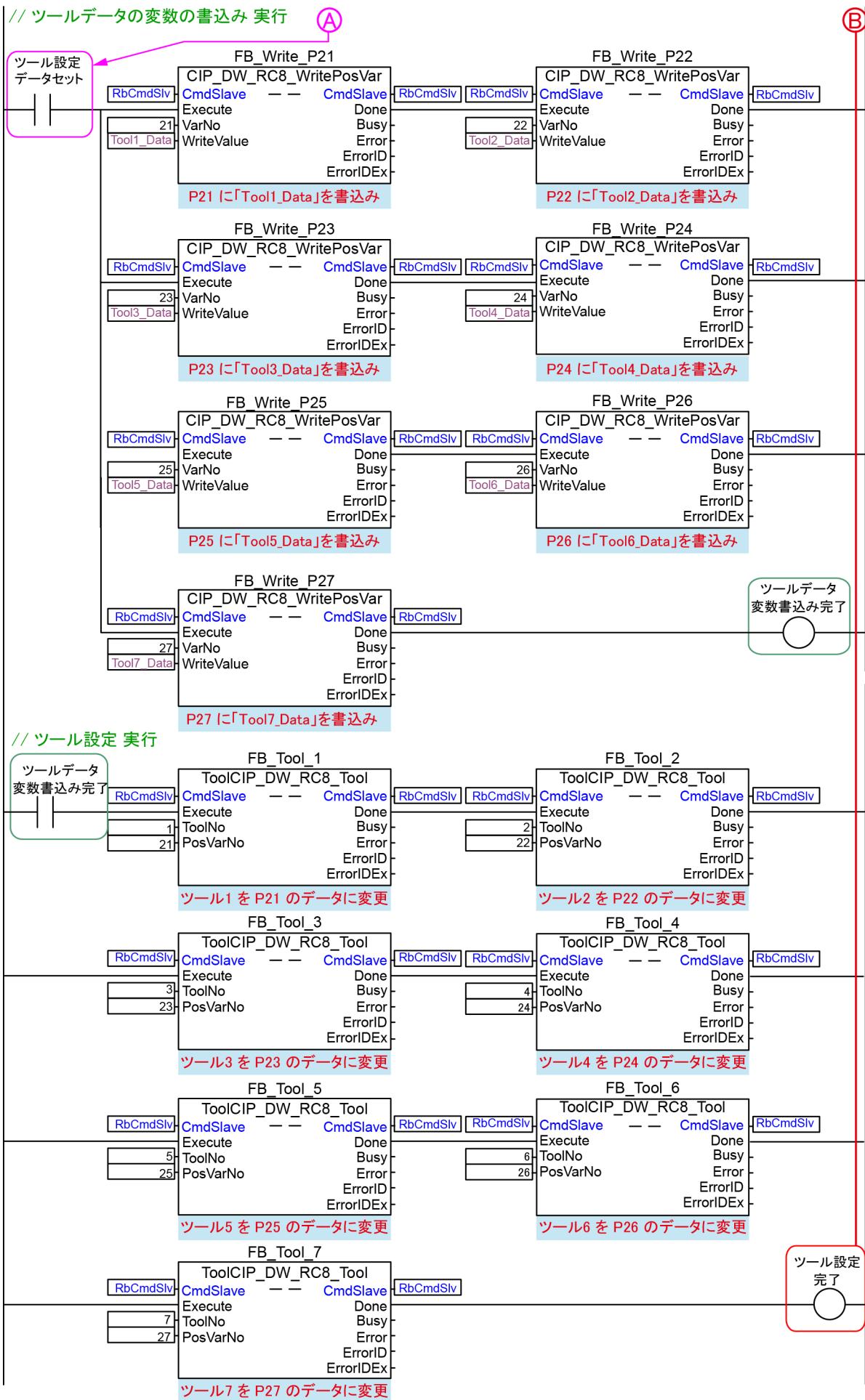
名称	データ型	備考
Tool1_Data	ARRAY[0..9] OF REAL	Tool1 設定値
Tool2_Data	ARRAY[0..9] OF REAL	Tool2 設定値
Tool3_Data	ARRAY[0..9] OF REAL	Tool3 設定値
Tool4_Data	ARRAY[0..9] OF REAL	Tool4 設定値
Tool5_Data	ARRAY[0..9] OF REAL	Tool5 設定値
Tool6_Data	ARRAY[0..9] OF REAL	Tool6 設定値
Tool7_Data	ARRAY[0..9] OF REAL	Tool7 設定値

RC8Command-Slave 活用ガイド

■ ラダー図



RC8Command-Slave 活用ガイド



1.4.9. ツール変更とツール番号の取得

Tool1 に変更して、現在のツール番号を取得します。(ツール座標については「3.2.ツールの説明」参照)

「ロボット Enable」が実行中である事が必要です。※ラダー図(1)を参照。
標準ツール(Tool 0)への変更や番号取得も可能です。

ツール番号	X	Y	Z	RX	RY	RZ
1	-50	0	0	0	0	0
2	0	50	0	0	0	0
3	0	0	50	0	0	0
4	-50	50	50	0	0	0
5	0	0	50	45	0	0
6	0	0	50	0	-45	0
7	0	0	50	0	0	45
0 ※	0	0	0	0	0	0

※標準ツール(ツール 0)の座標値編集はできません。

■ グローバル変数

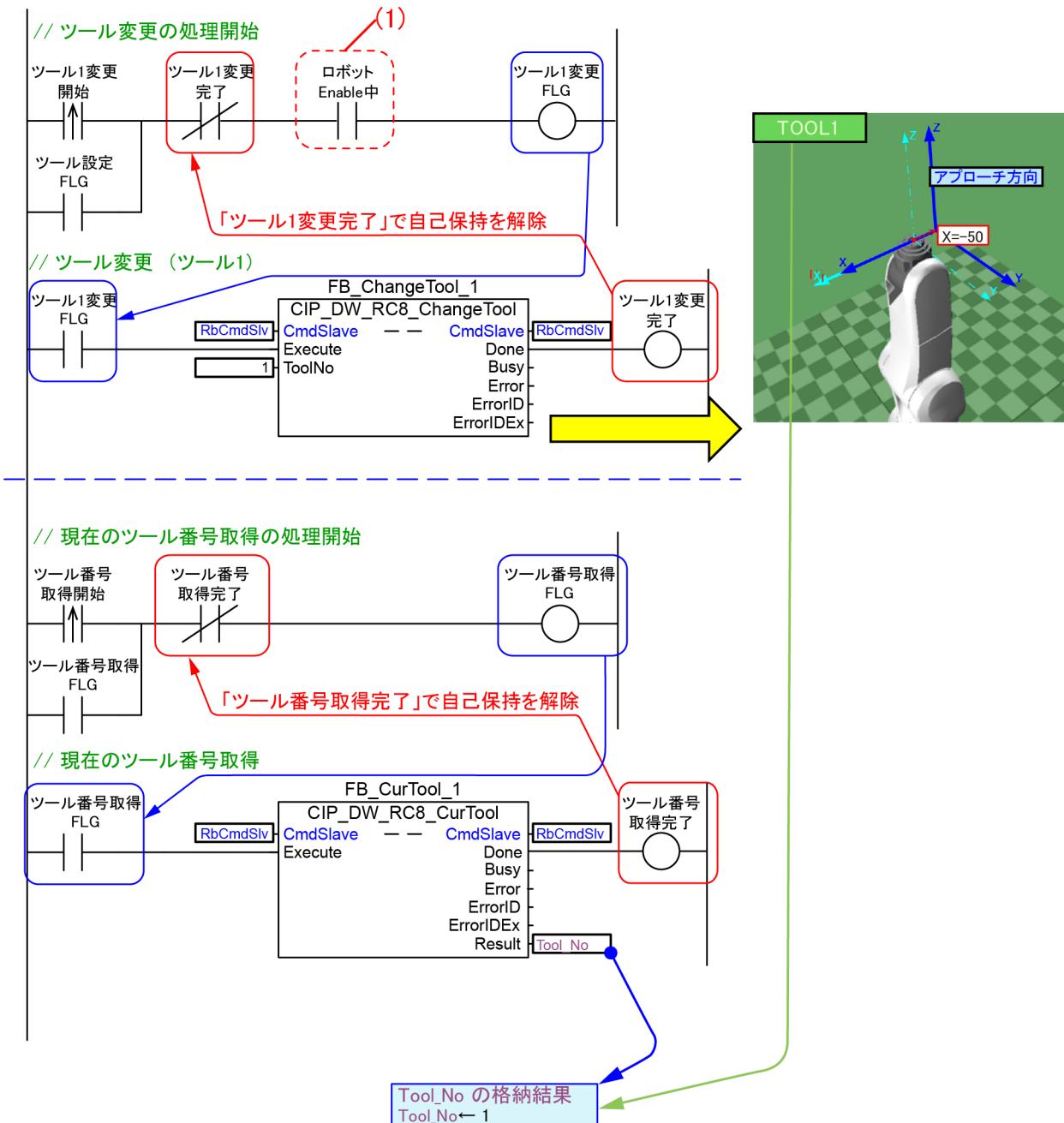
名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

■ ローカル変数

名称	データ型	備考
Tool_No	DINT	ツール番号の取得値保存用

RC8Command-Slave 活用ガイド

■ ラダー図



1.4.10. ワーク設定

WORK 1 ~ WORK 6 に以下の値を設定します。(ワークについては「3.3.ワークの説明」参照)

「ロボット Enable」が実行中である事が必要です。※ラダー図(1)を参照。

設定値は一度 RC8 の P 型変数に書き込み、書込んだ P 型変数の番号を指定してワーク設定を行います。

ワーク番号	X	Y	Z	RX	RY	RZ	格納 P 型変数
1	500	0	0	0	0	0	P31
2	350	350	100	0	0	-45	P32
3	0	500	0	0	0	80	P33
4	700	0	200	0	-45	0	P34
5	500	-500	300	0	-45	-45	P35
6	0	-700	200	0	-45	-90	P36

■ グローバル変数

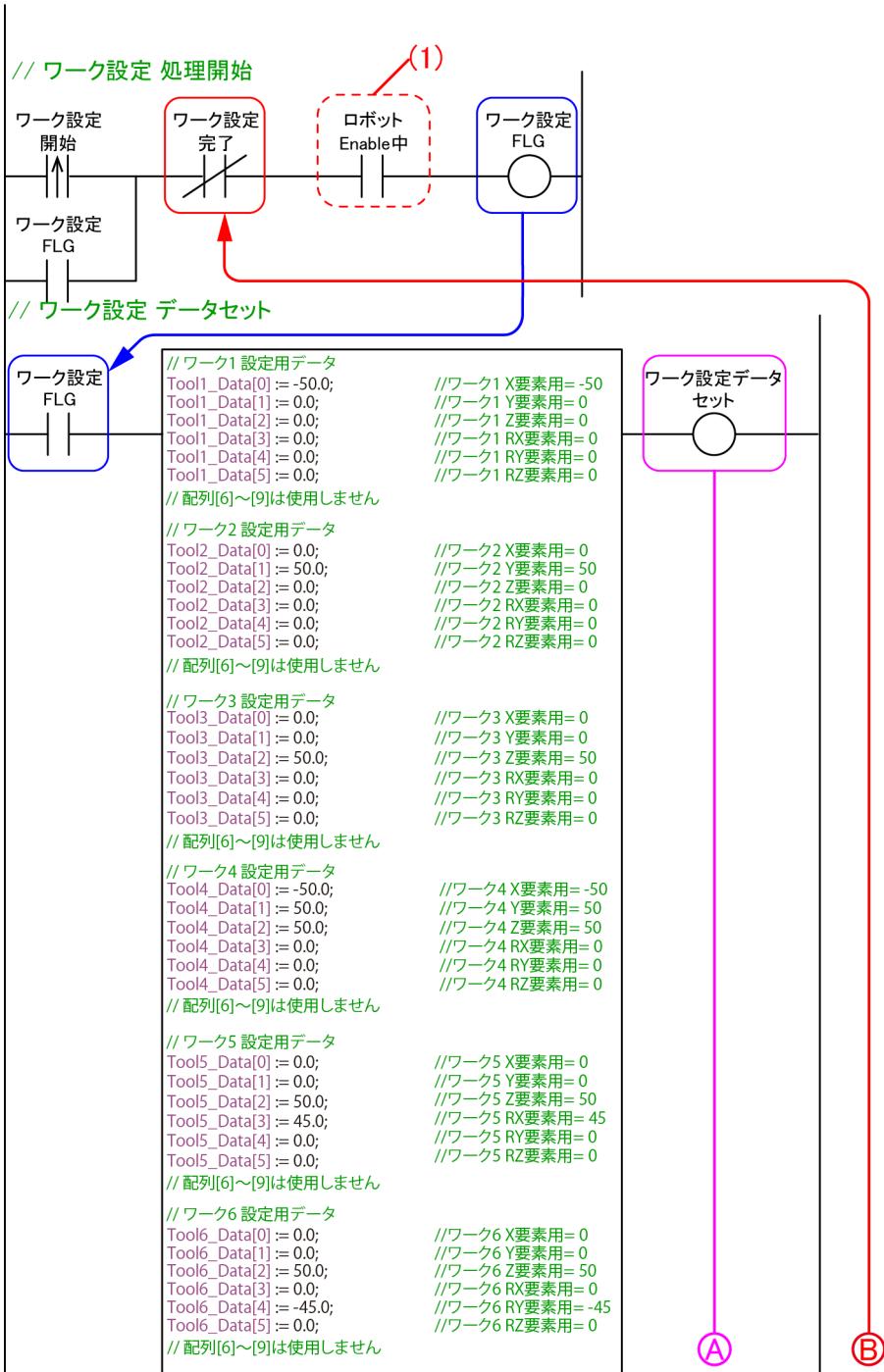
名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

■ ローカル変数

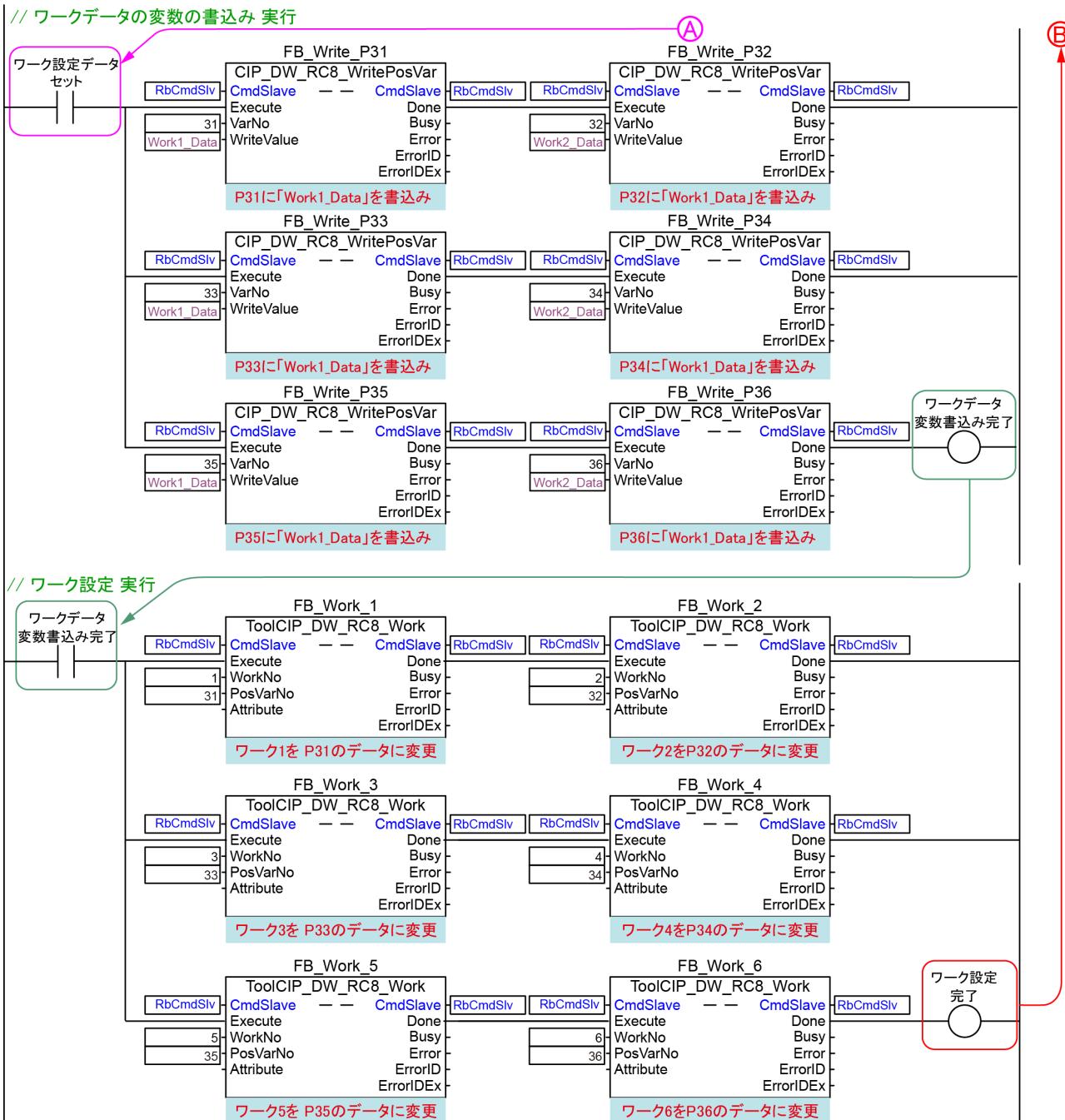
名称	データ型	備考
Work1_Data	ARRAY[0..9] OF REAL	ワーク 1 設定値
Work2_Data	ARRAY[0..9] OF REAL	ワーク 2 設定値
Work3_Data	ARRAY[0..9] OF REAL	ワーク 3 設定値
Work4_Data	ARRAY[0..9] OF REAL	ワーク 4 設定値
Work5_Data	ARRAY[0..9] OF REAL	ワーク 5 設定値
Work6_Data	ARRAY[0..9] OF REAL	ワーク 6 設定値
Work7_Data	ARRAY[0..9] OF REAL	ワーク 7 設定値

RC8Command-Slave 活用ガイド

■ ラダー図



RC8Command-Slave 活用ガイド



1.4.11. ワーク変更とワーク番号の取得

WORK 1 に変更して、現在のワーク番号を取得します。(ワークについては「3.3.ワークの説明」参照)

「ロボット Enable」が実行中である事が必要です。ラダー図を参照。

WORK 0(ベース座標)への変更&番号取得も可能です。

ワーク番号	X	Y	Z	RX	RY	RZ
1	500	0	0	0	0	0
2	350	350	100	0	0	-45
3	0	500	0	0	0	80
4	700	0	200	0	-45	0
5	500	-500	300	0	-45	-45
6	0	-700	200	0	-45	-90
0	0	0	0	0	0	0

※ベース座標(ワーク番号 0)の座標値編集はできません。

■ グローバル変数

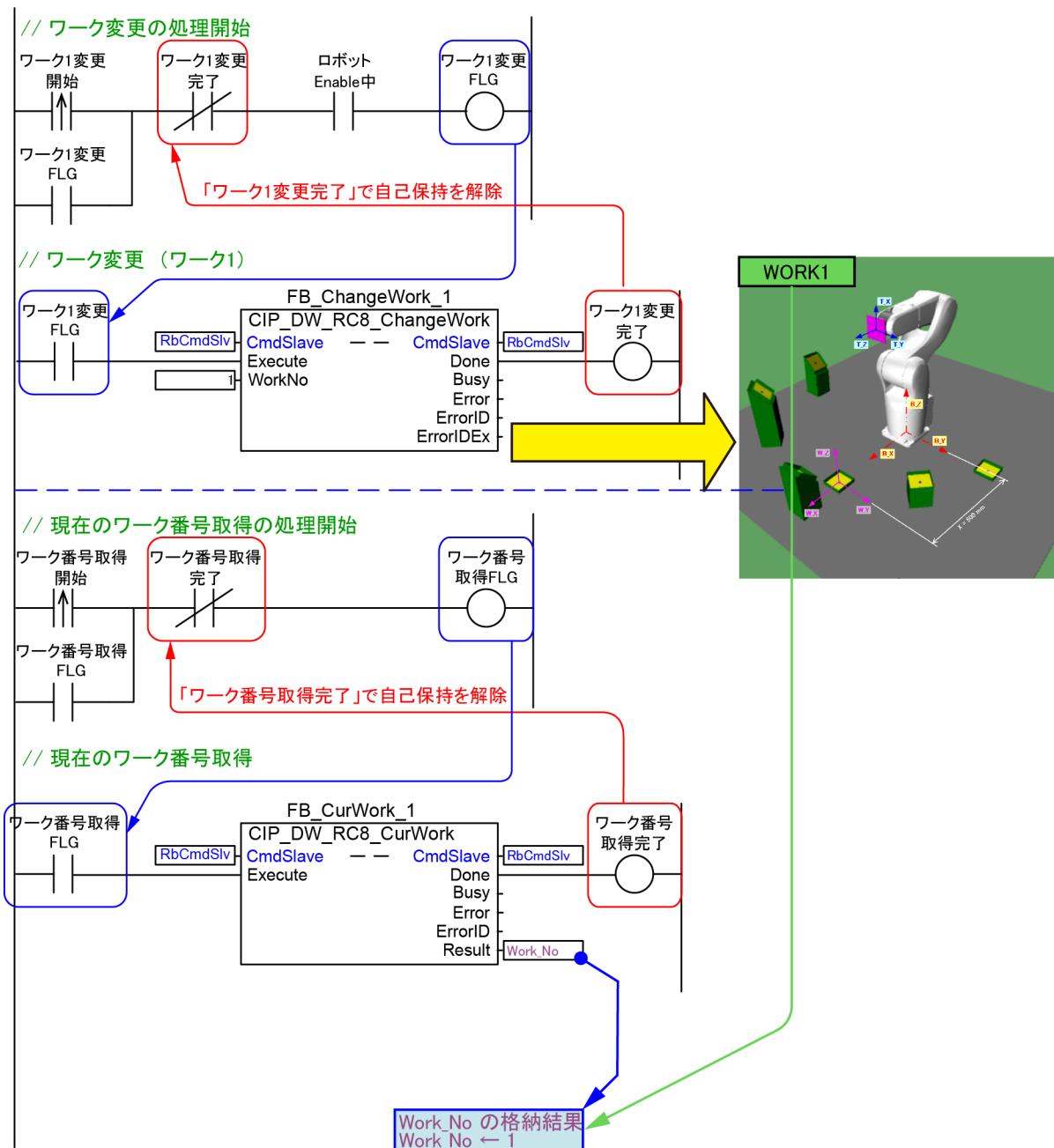
名称	データ型	ネットワーク公開	相手機器の割り当て
RbCmdS1v	sRC8_COMMAND_SLAVE	---	無し

■ ローカル変数

名称	データ型	備考
Work_No	DINT	ワーク番号の取得値保存用

RC8Command-Slave 活用ガイド

■ ラダー図



1.5. エラー表示の見方

ファンクションブロック起動時に、何らかの要因でファンクションブロックが異常終了すると、出力変数「Error」（データ型：BOOL）が TRUE となります。

その場合、「ErrorID」・「ErrorIDEx」には、以下の内容が表示されます。

【ErrorID:異常状態に応じたエラーコードを出力します】

エラーコード	名称	内容
2800	FB 下請処理異常	FB 内で使用する下請 FB や外部機器のエラーを検出した
2801	FB 処理異常	FB 自身でエラーを検出した

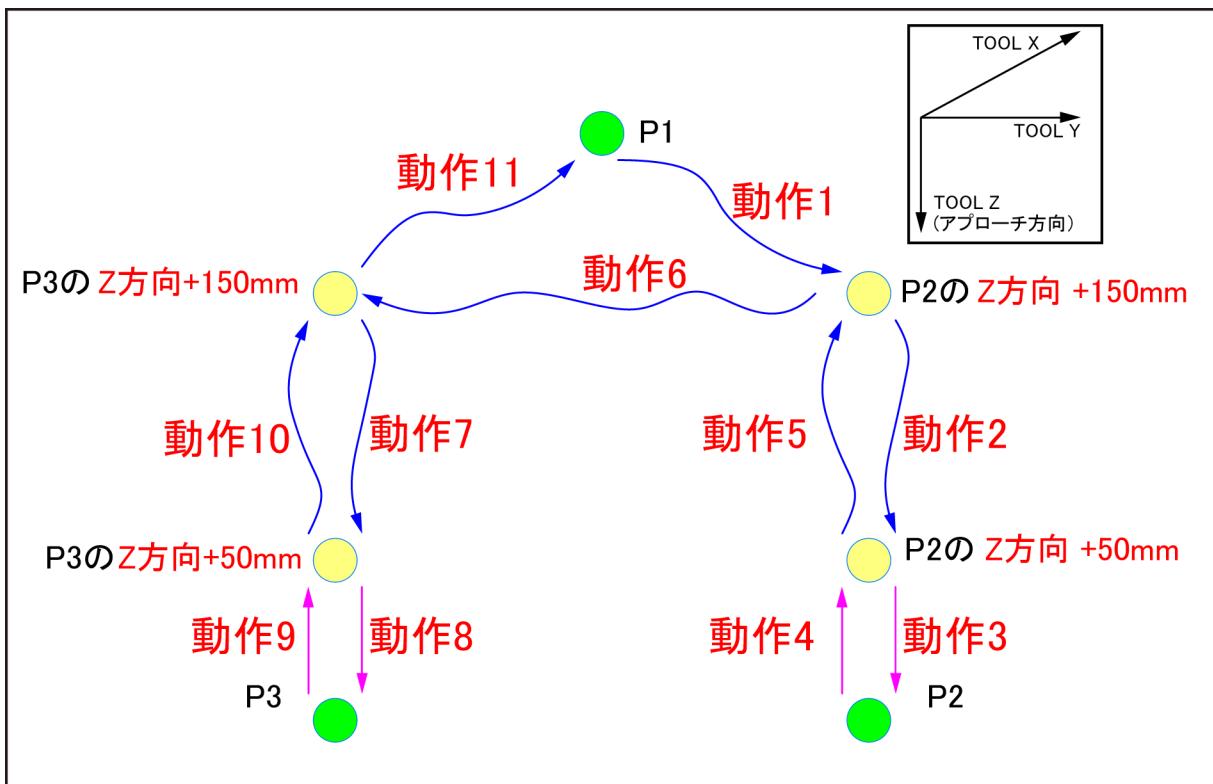
【ErrorIDEx:異常状態に応じた詳細エラーコードを出力します】

エラーコード	名称	内容
FF01	タイムアウト	RC8 のコマンドレスポンスが 2 秒を超えた (EN 入力 ON 後、Busy 出力が ON するまでの時間)
FF02	FB 多重起動	FB を同時に複数起動した
FF03	ロボット 実行不可	ロボットが実行不可の状態で FB を起動した
FF04	Command-Slave ライセンス無効	Command-Slave 機能有効が OFF のときに FB を起動した
FF05	ロボット制御権 未取得	ロボット Enable 処理許可が OFF のときに RobotEnableFB を実行した。またはモータ処理許可が OFF のときに MotorFB を実行した。
8*****	ロボットエラーコード	詳細は、RC8 取扱説明書のエラーコード表を参照ください

2. プログラミング例

P1 からスタートして P2 でワークを取り P3 へ運ぶ、一連の動きを例として説明します。

(1) プログラミング例の動作図



(2) TOOL 座標(TOOL1 を使用)

TOOL 番号	X	Y	Z	RX	RY	RZ
1	0	0	90	0	0	0

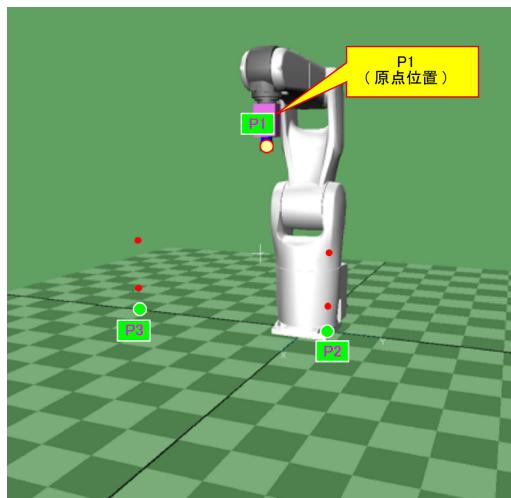
(3) ロボット動作位置(TOOL1 での値)

位置	変数	X	Y	Z	RX	RY	RZ	FIG
原点位置	P1	300	0	500	180	0	180	5
作業位置 1	P2	500	200	160	180	0	180	5
作業位置 2	P3	500	-200	160	180	0	180	5

2.1. 動作ステップ

動作ステップを以下に示します。

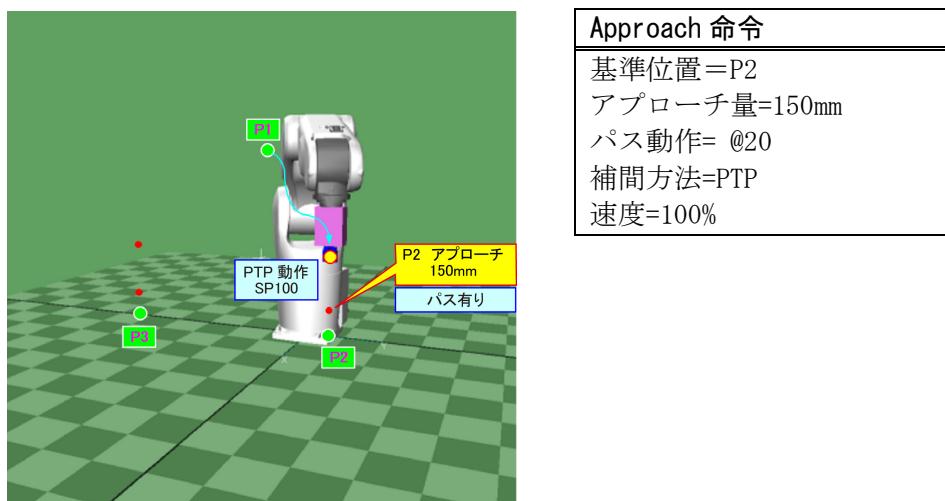
ここでは P1 を原点位置として説明します。



動作 1

原点位置(P1)から P2 のアプローチ 150mm の位置へ PTP 動作で移動します。

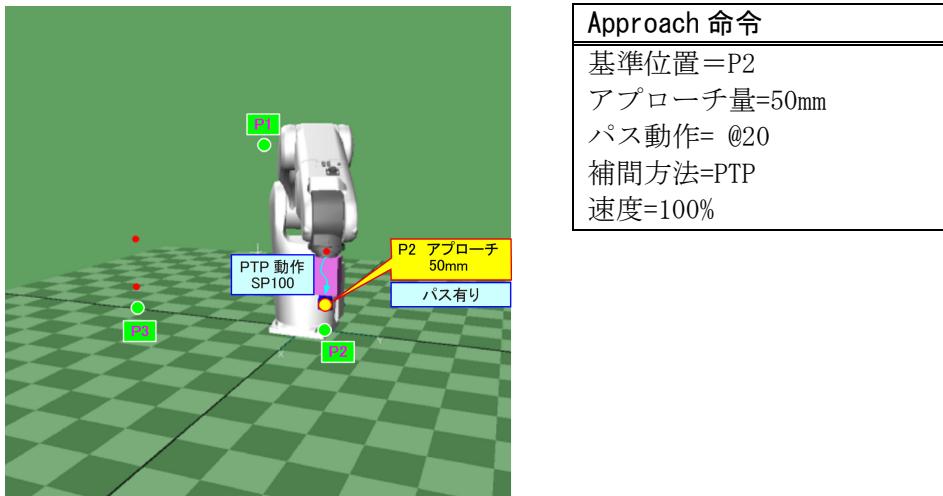
(パス : @20、速度 : SP100)



動作 2

P2 のアプローチ 50mm の位置へ PTP 動作で移動します。

(パス : @20、速度 : SP100)

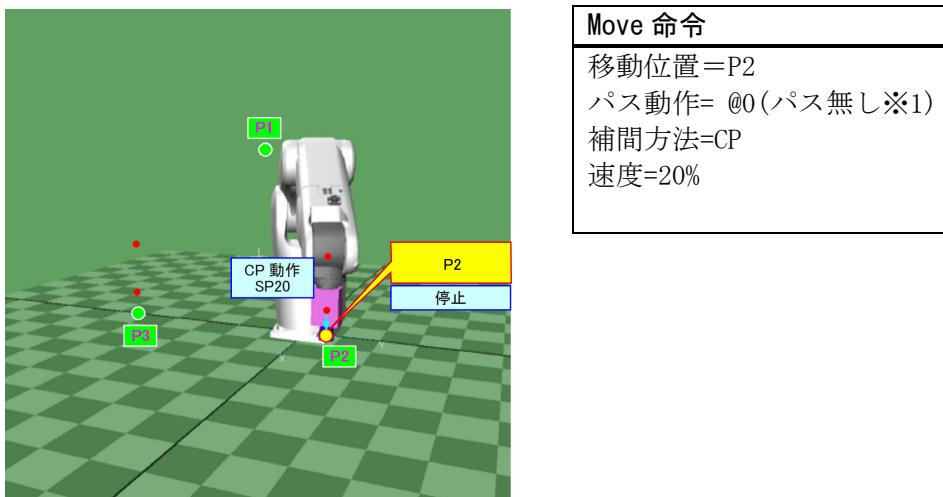


動作 3

P2 の位置へ CP 動作で移動し、動作停止後にワークをチャックします。

(パス : @20、速度 : SP20)

※ワークと周辺機器との干渉を考慮して CP 動作(直線補間)と低速度に設定



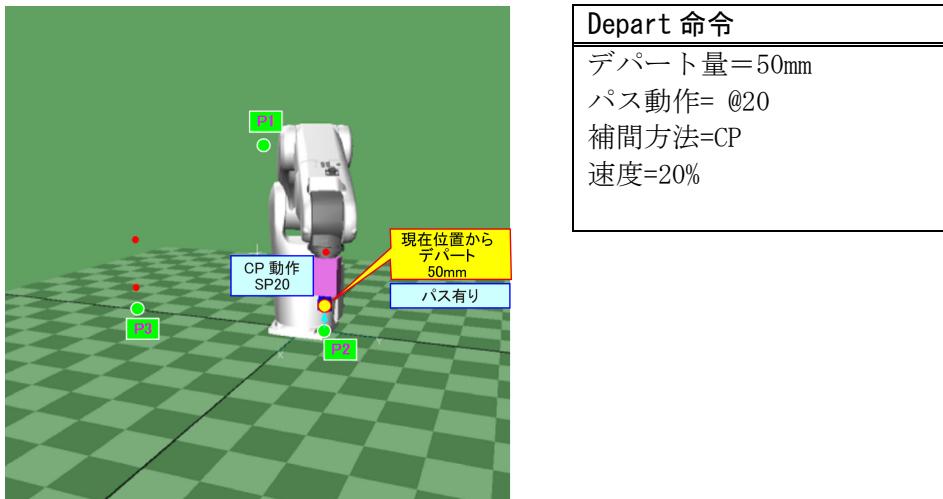
※1 パス動作 =@0 はパス無しで、その場に停止します。

動作 4

現在位置からデパート 50mm の位置へ CP 動作で移動します。

(パス : @20、速度 : SP20)

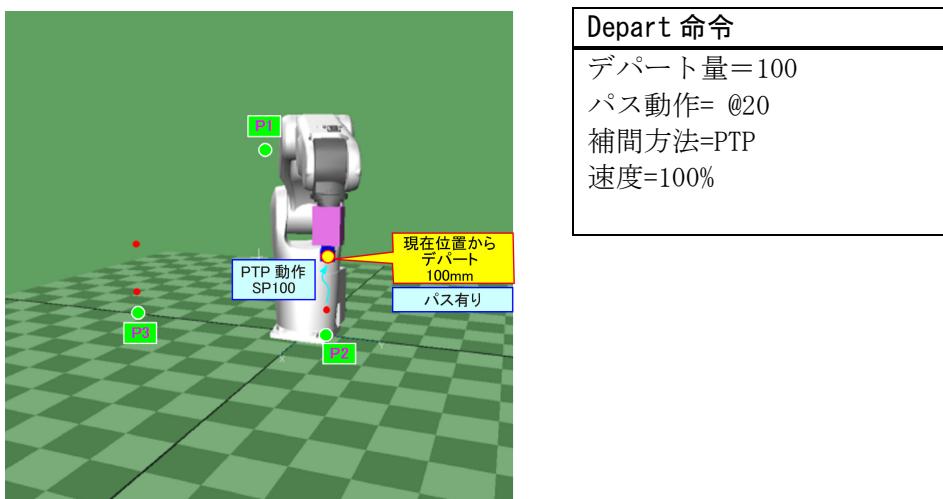
※ワークと周辺機器との干渉を考慮して CP 動作(直線補間)と低速度に設定



動作 5

現在位置からデパート 100mm の位置へ PTP 動作で移動します。

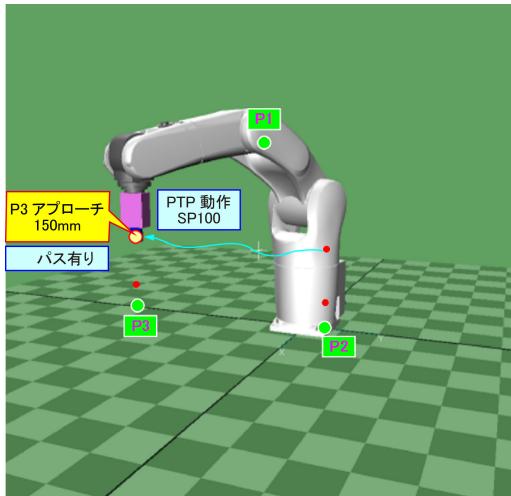
(パス : @20、速度 : SP100)



動作 6

P3 のアプローチ 150mm の位置へ PTP 動作で移動します。

(パス : @20、速度 : SP100)

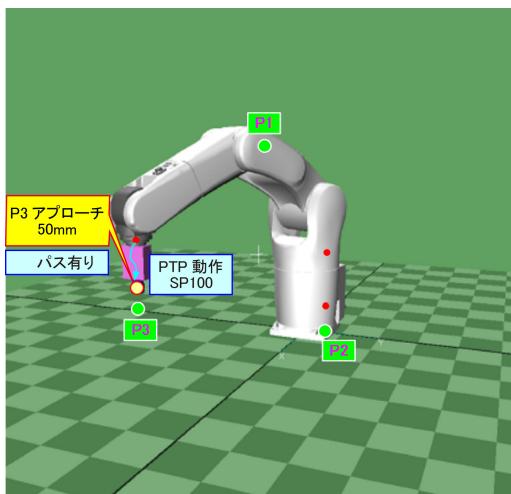


Approach 命令
基準位置=P3
アプローチ量=150mm
パス動作=@20
補間方法=PTP
速度=100%

動作 7

P3 のアプローチ 50mm の位置へ PTP 動作で移動します。

(パス : @20、速度 : SP100)

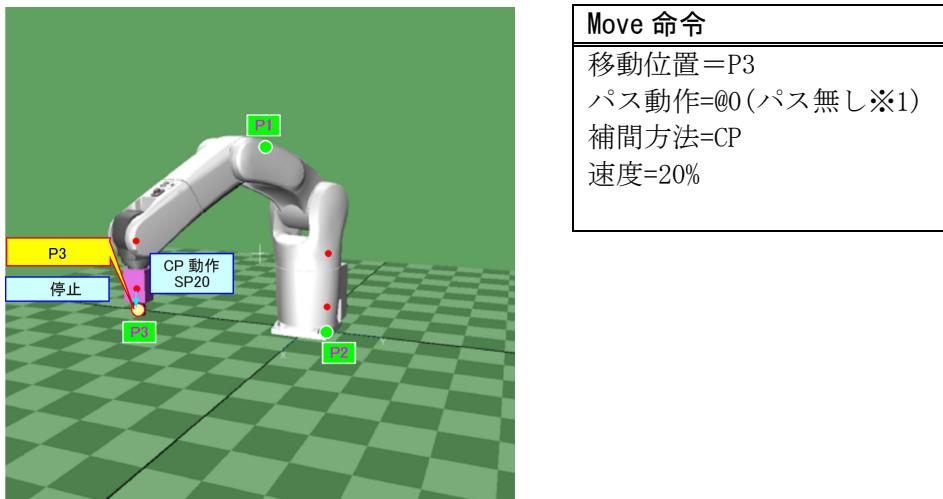


Approach 命令
基準位置=P3
アプローチ量=50mm
パス動作=@20
補間方法=PTP
速度=100%

動作 8

P3 の位置へ CP 動作で移動し、動作停止後にワークをアンチャックします。
(パス:@20、速度：SP20)

※ワークと周辺機器との干渉を考慮して CP 動作(直線補間)と低速度に設定

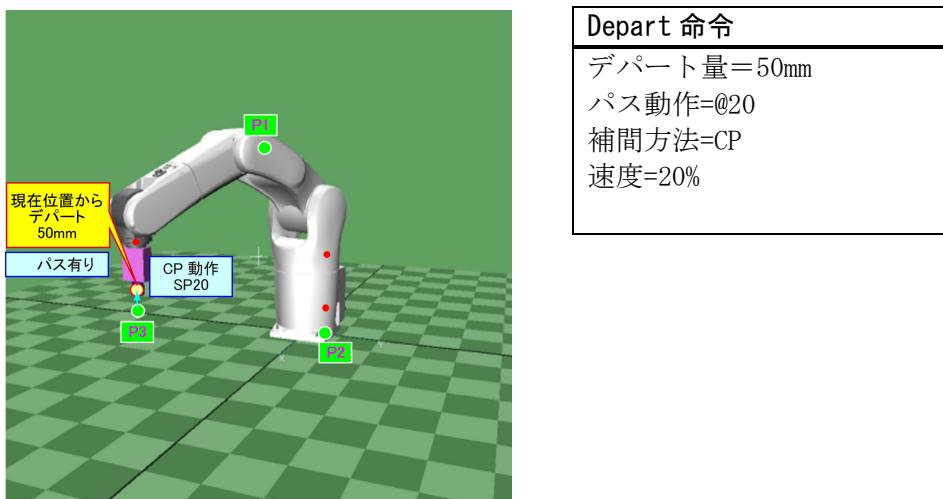


動作 9

現在位置からデパート 50mm の位置へ CP 動作で移動します。

(パス:@20、速度：SP20)

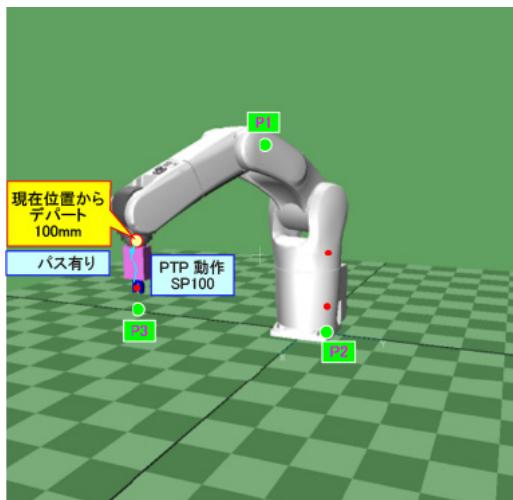
※ワークと周辺機器との干渉を考慮して CP 動作(直線補間)と低速度に設定



動作 10

現在位置からデパート 100mm の位置へ PTP 動作で移動します。

(パス : @20、速度 : SP100)

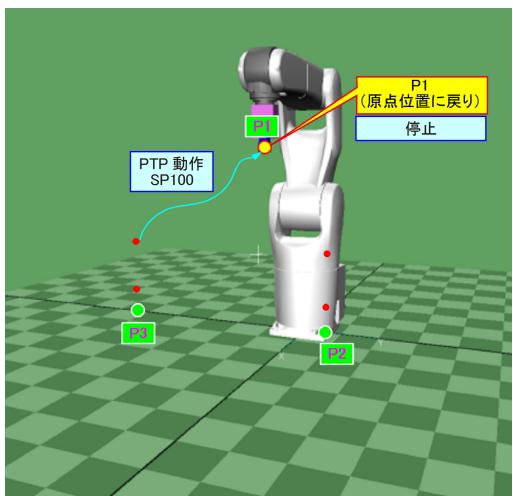


Depart 命令	
デパート量	=100mm
パス動作	=@20
補間方法	=PTP
速度	=100%

動作 11

P1 の位置へ PTP 動作で移動し、動作停止後に次のサイクル待ち状態になります。

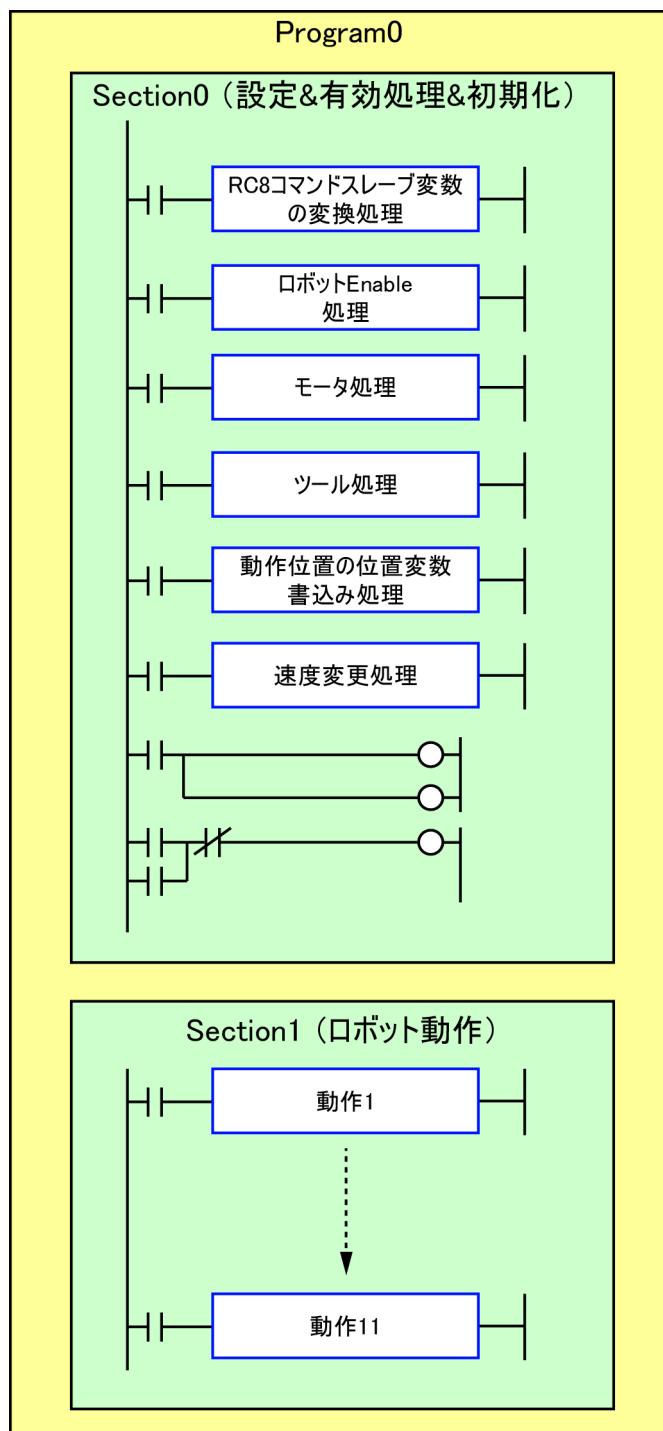
(パス : @20、速度 : SP100)



Move 命令	
移動位置	=P1
パス動作	=@0 (パス無し※1)
補間方法	=PTP
速度	=100%

2.2. PLC プログラム構成

例では「Program0」で構成しています。セクションは「Section0(設定、有効処理、初期化)」と「Section1(ロボット動作)」で構成しています。

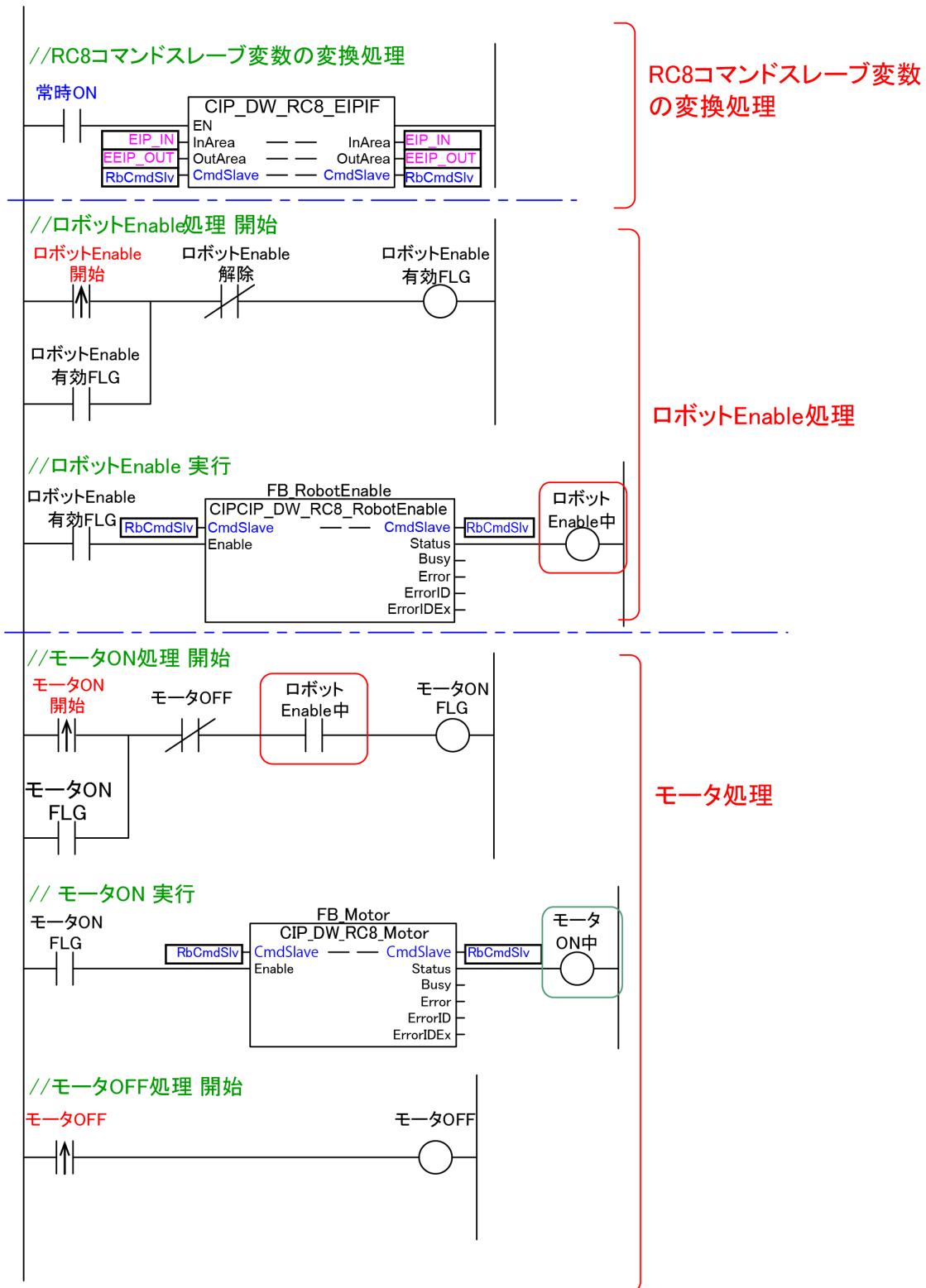


2.2.1. Section0 (設定、有効処理、初期化)の説明

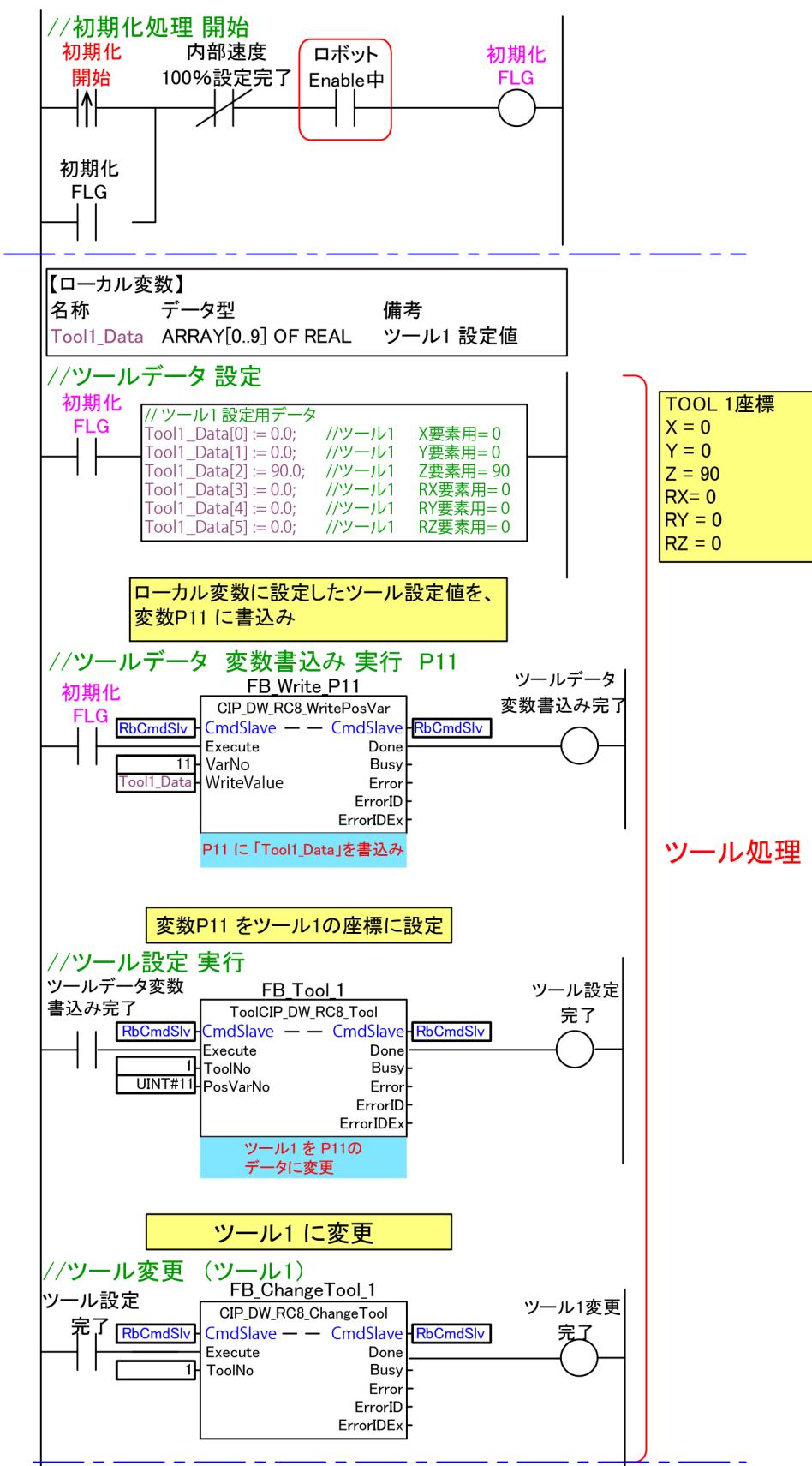
■ グローバル変数

名称	データ型	ネットワーク公開	相手機器の割り当て
EIP_IN	ARRAY[0..125]OF DWORD	入力	RC8 入力データ(504Byte)
EIP_OUT	ARRAY[0..125]OF DWORD	出力	RC8 出力データ(504Byte)
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

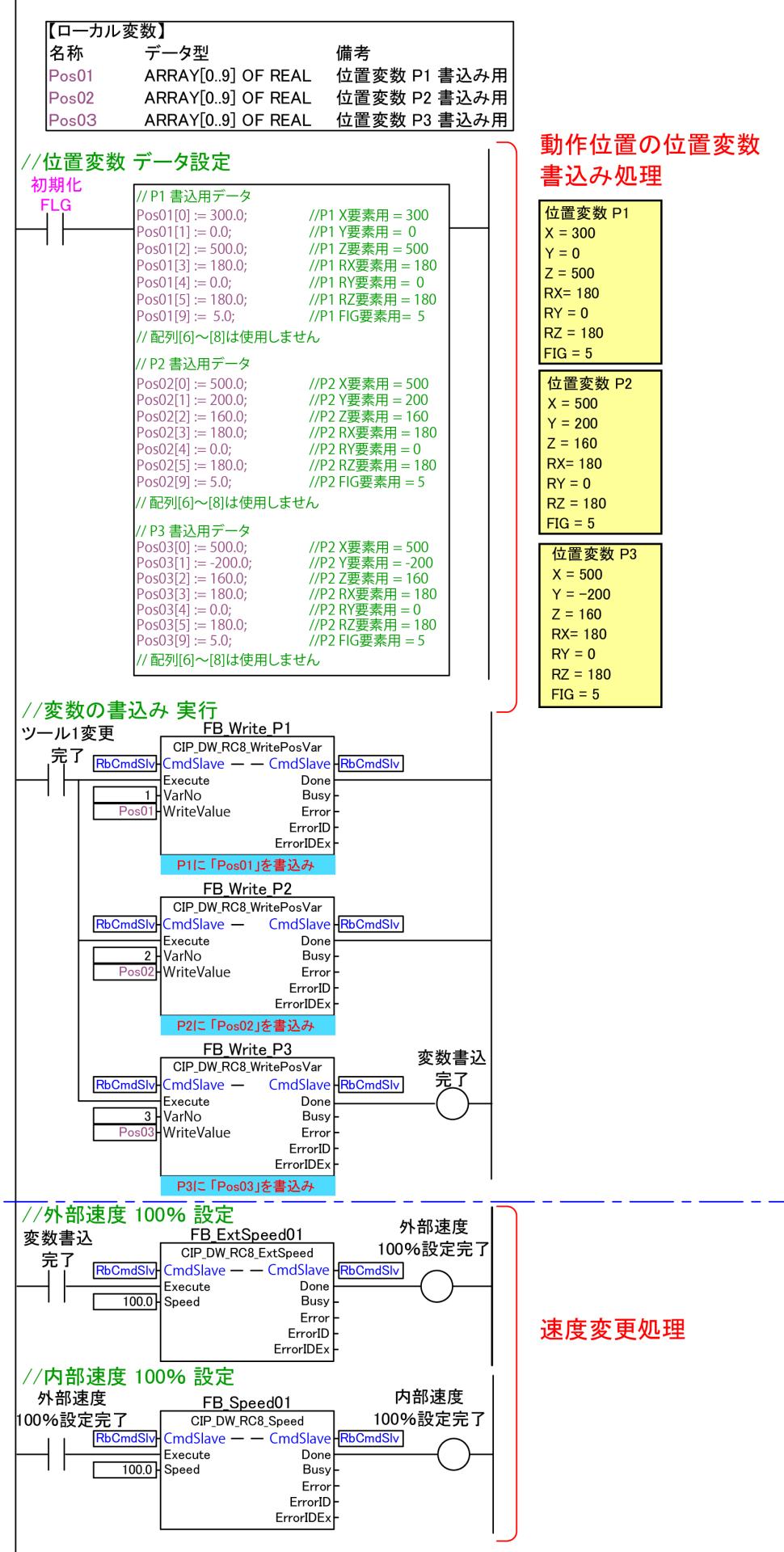
■ ラダー図



RC8Command-Slave 活用ガイド



RC8Command-Slave 活用ガイド



2.2.2. Section1(ロボット動作)の説明 (Next 処理を使用する場合)

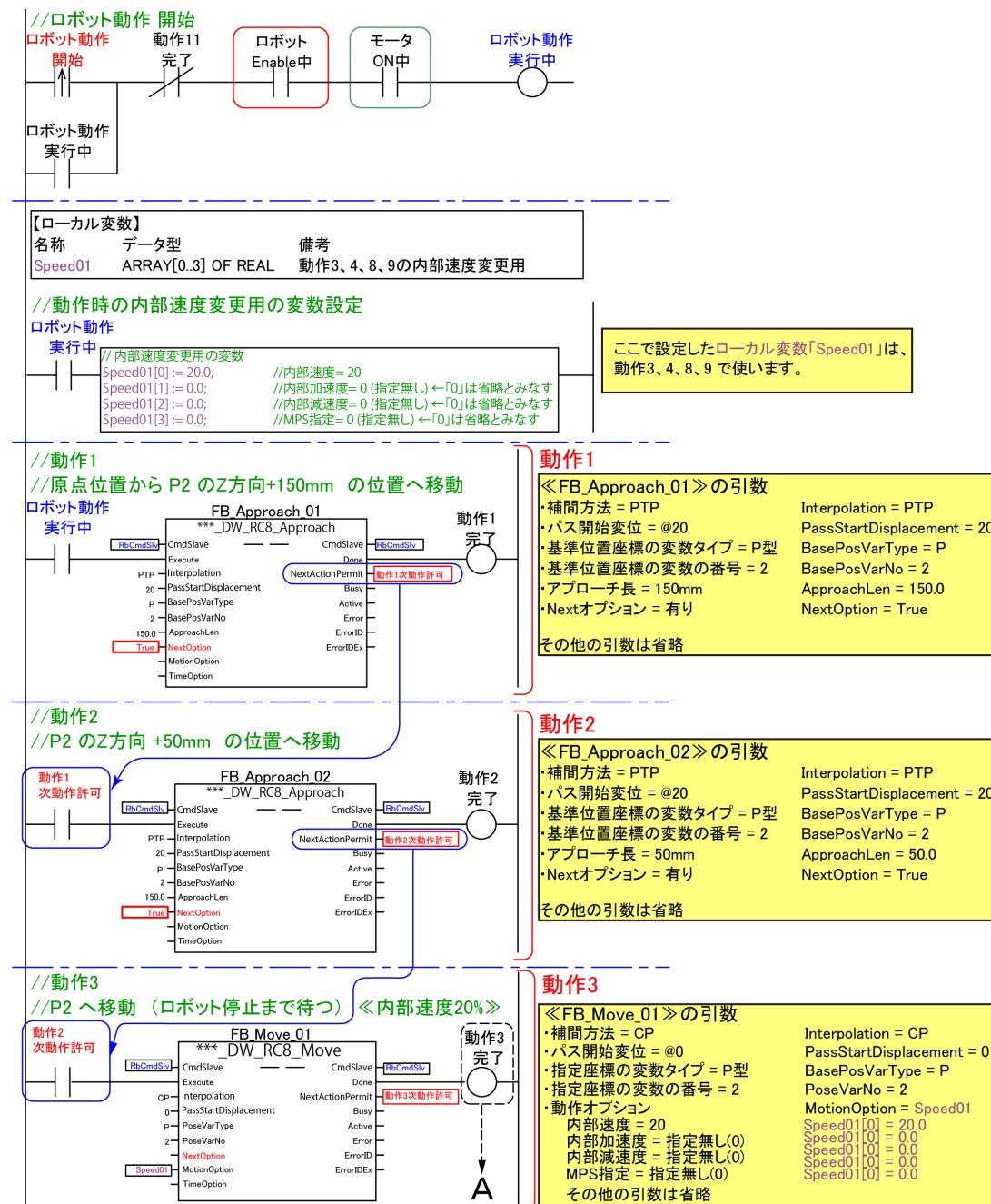
Next オプション有りの場合について説明します。

Next オプション有りの場合は、動作コマンドの完了を待たずに次のコマンドを実行するため、多くの場合、全体の処理時間を短くすることができます。

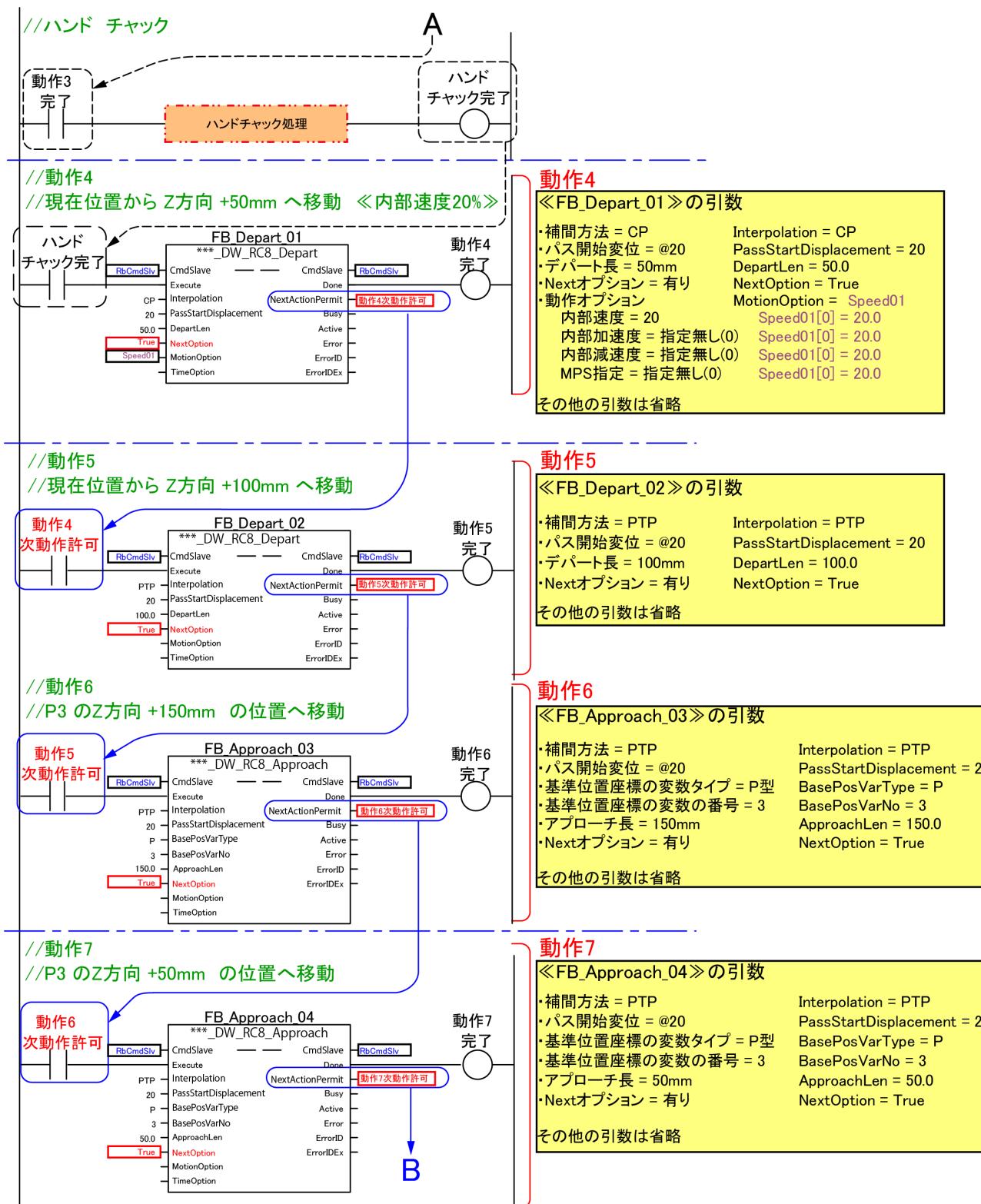
■ グローバル変数

名称	データ型	ネットワーク公開	相手機器の割り当て
EIP_IN	ARRAY[0..125]OF DWORD	入力	RC8 入力データ(504Byte)
EIP_OUT	ARRAY[0..125]OF DWORD	出力	RC8 出力データ(504Byte)
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

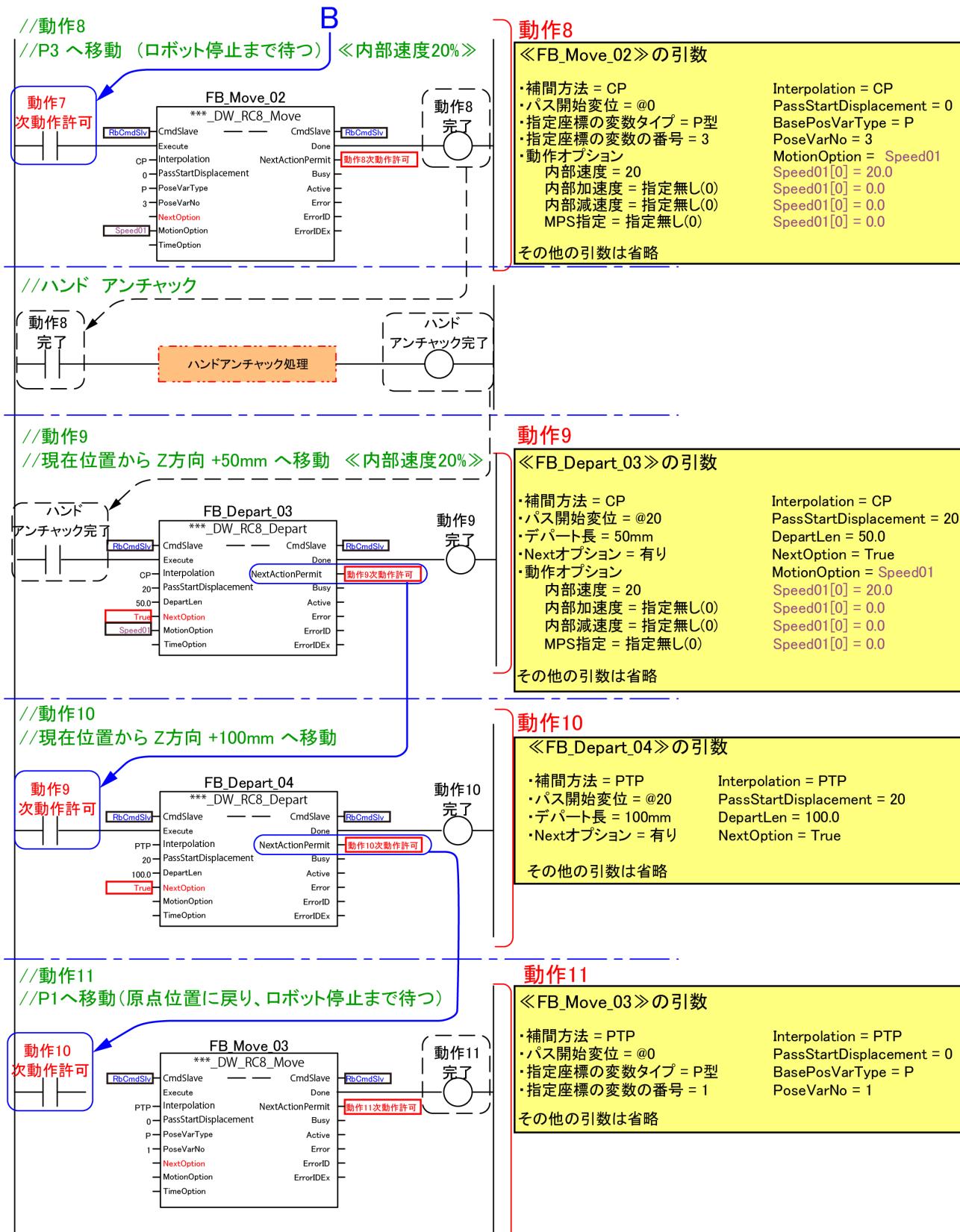
■ ラダー図



RC8Command-Slave 活用ガイド

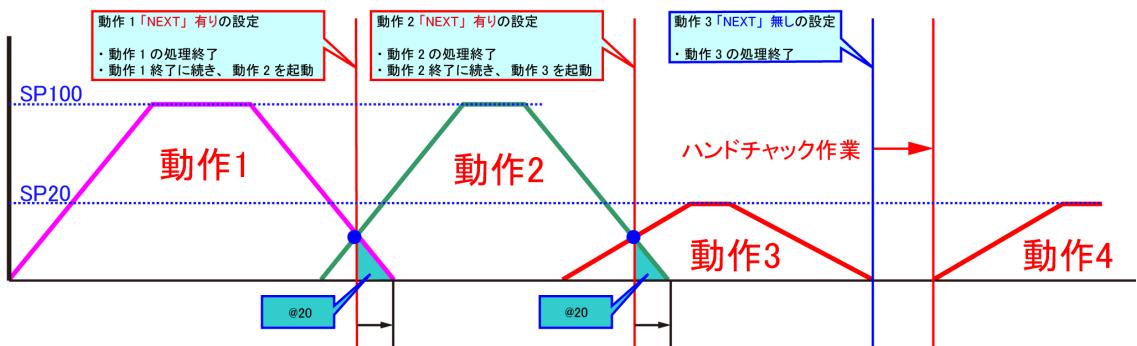


RC8Command-Slave 活用ガイド

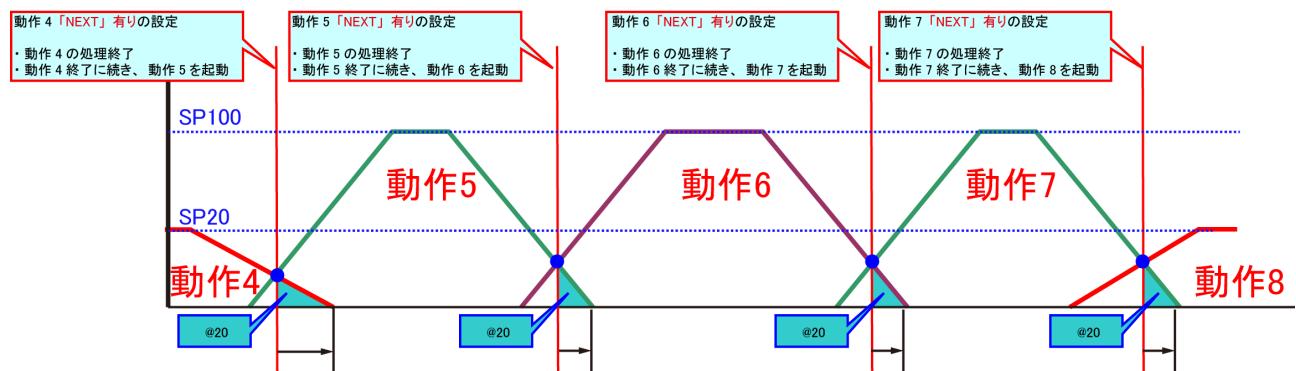


■動作のチャート

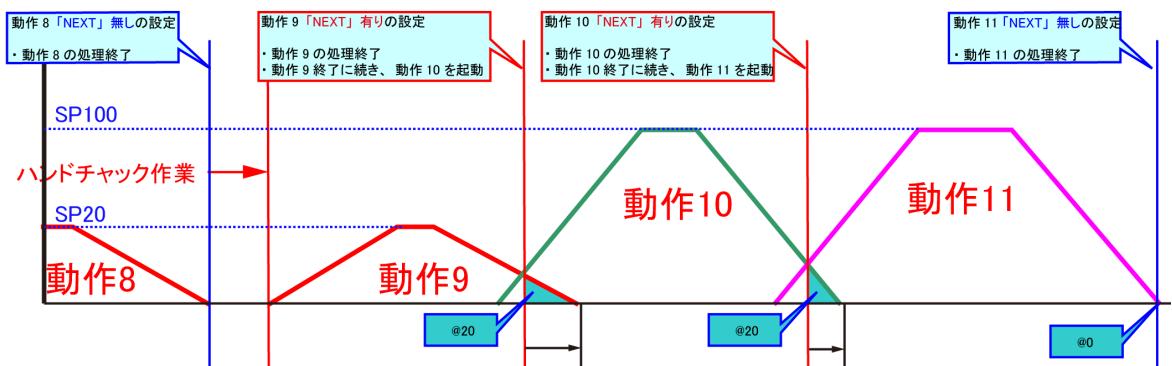
動作 1 ~ 4



動作 4 ~ 8

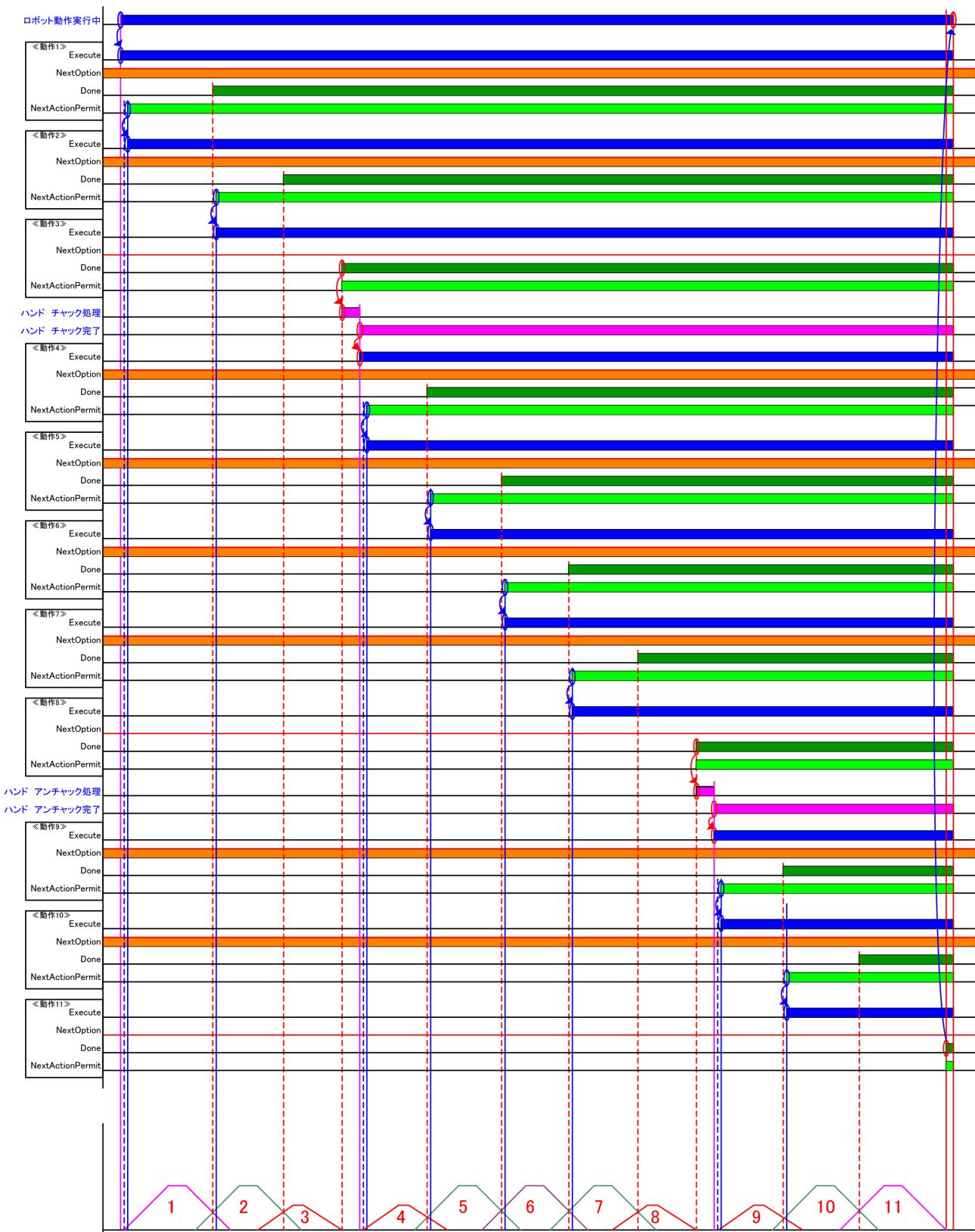


動作 8 ~ 11



RC8Command-Slave 活用ガイド

■動作 1～11 の各 FB の起動と動作完了、次動作許可のタイムチャート

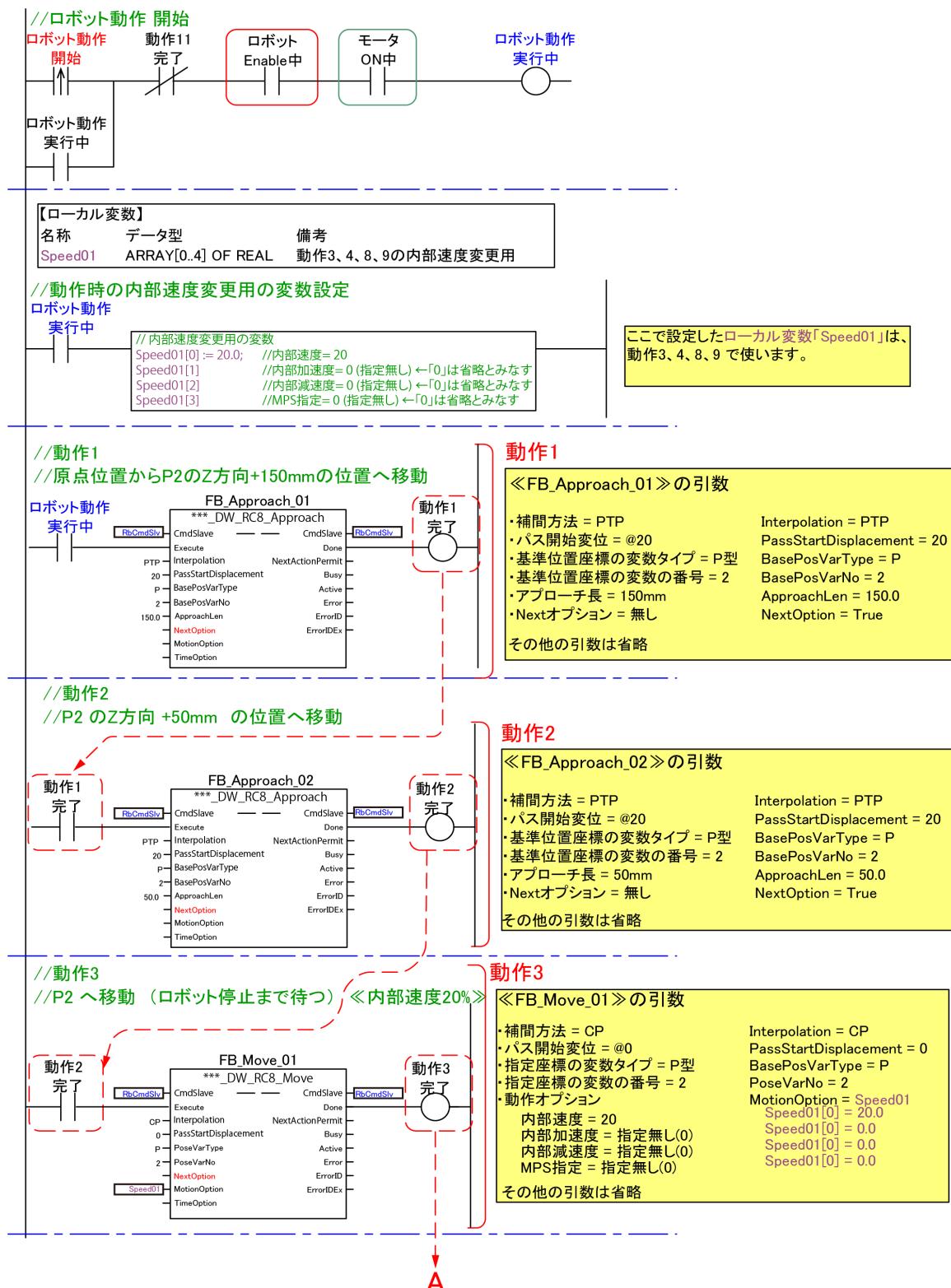


2.2.3. Section1(ロボット動作)の説明 (Next 処理を使用しない場合)

■ グローバル変数

名称	データ型	ネットワーク公開	相手機器の割り当て
EIP_IN	ARRAY[0..125]OF DWORD	入力	RC8 入力データ(504Byte)
EIP_OUT	ARRAY[0..125]OF DWORD	出力	RC8 出力データ(504Byte)
RbCmdSlv	sRC8_COMMAND_SLAVE	---	無し

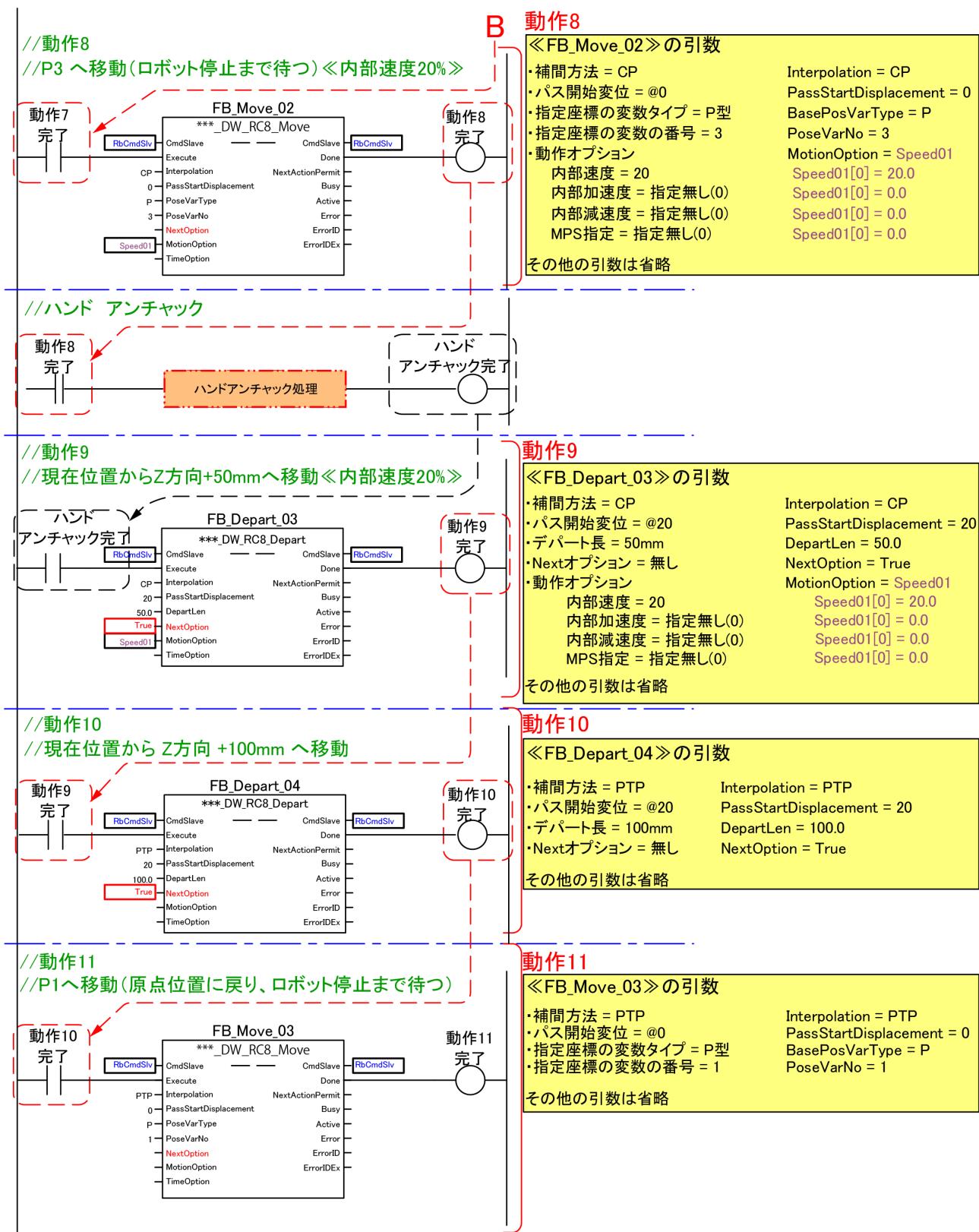
■ ラダー図



RC8Command-Slave 活用ガイド

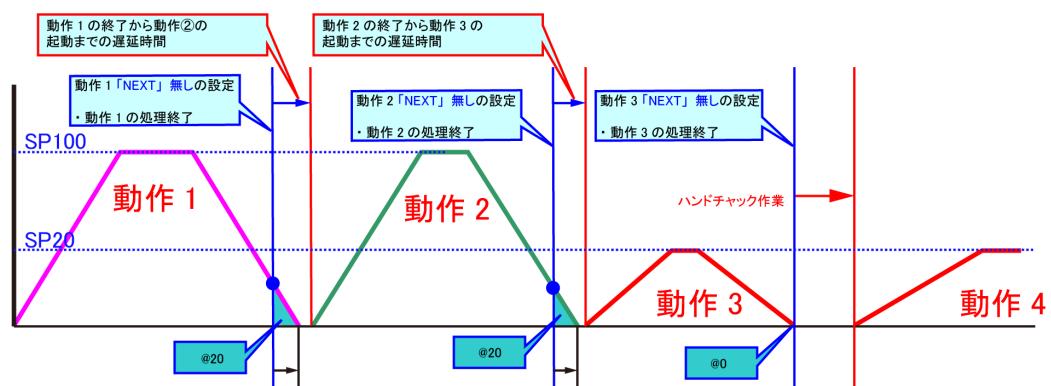


RC8Command-Slave 活用ガイド

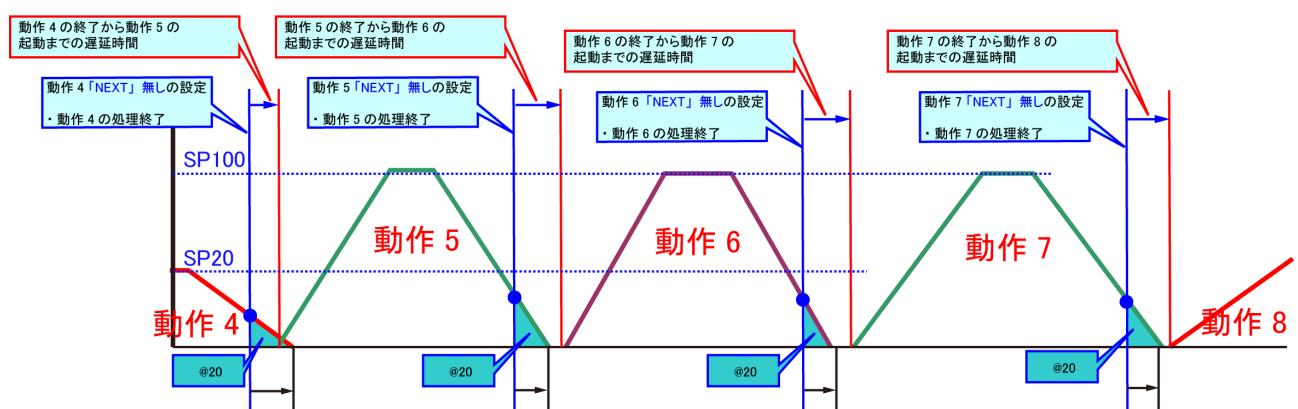


■動作のチャート

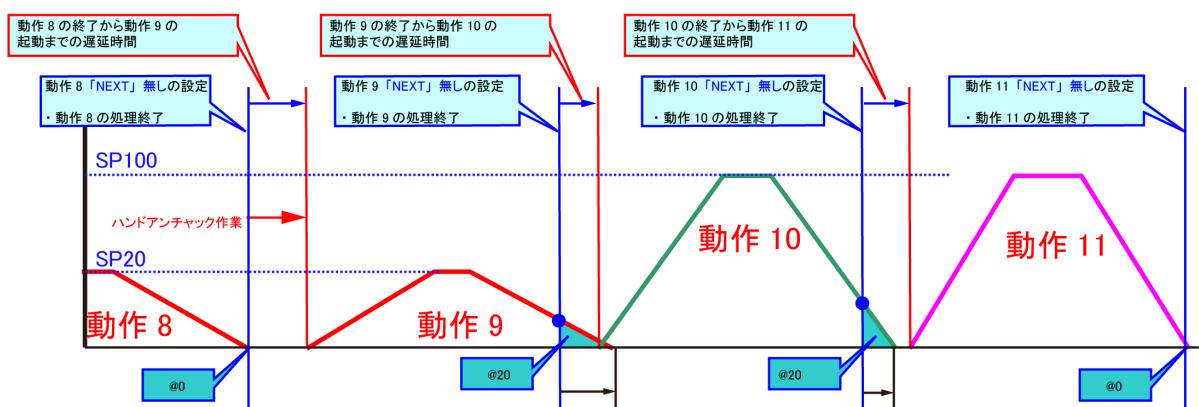
動作 1 ~ 4



動作 4 ~ 8

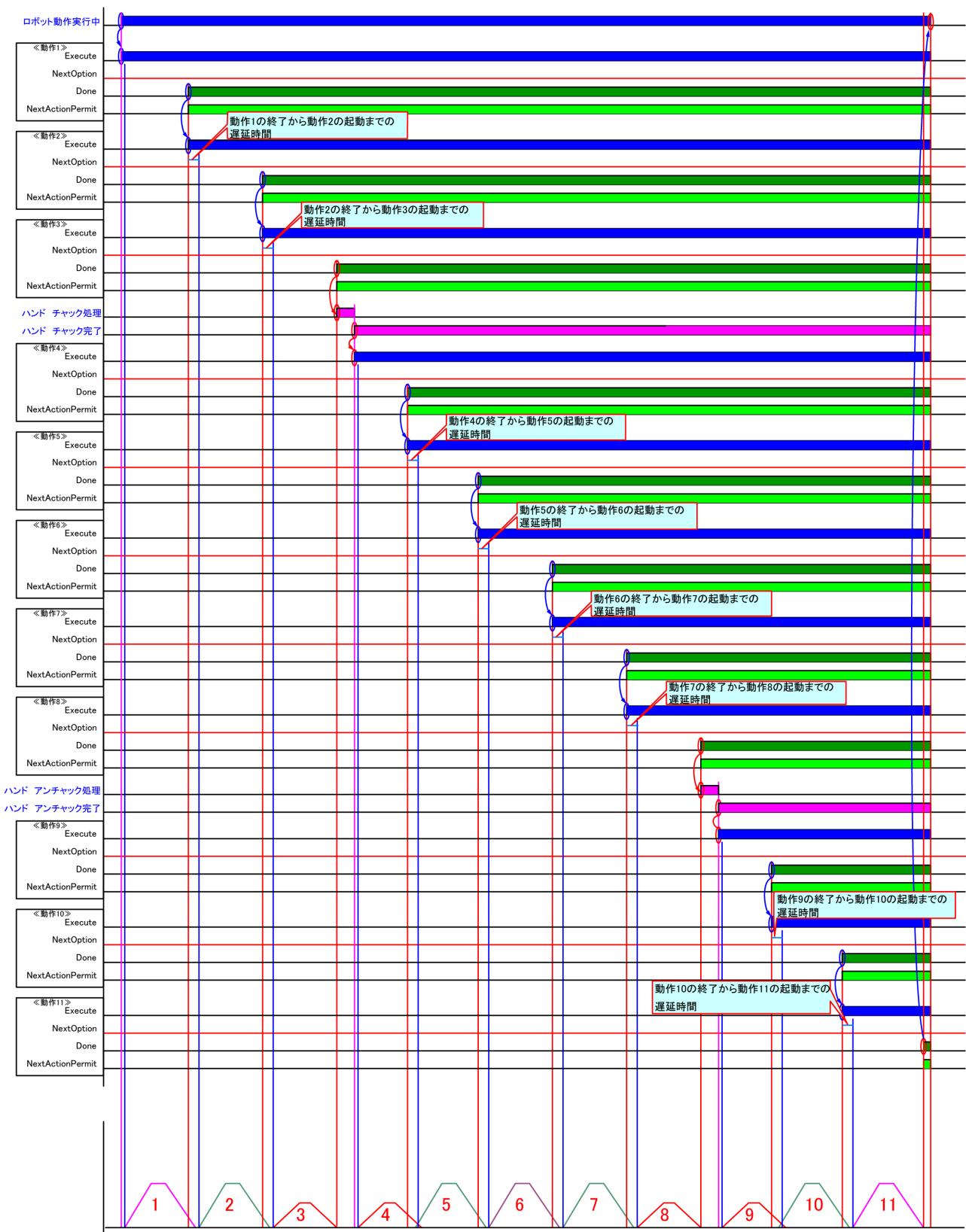


動作 8 ~ 11



RC8Command-Slave 活用ガイド

■動作 1～11 の各 FB の起動と動作完了、次動作許可のタイムチャート



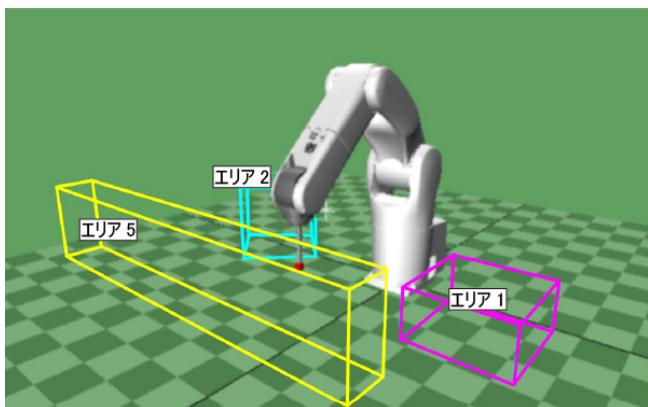
3. アプリケーション

3.1. エリア設定

エリアの設定方法を説明します。

■エリアの中心位置と大きさ(範囲)の設定

「エリア1」、「エリア2」、「エリア3」の中心位置と大きさ(範囲)を Command-Slave で設定します。



■エリアの中心位置、中心座標傾きと大きさ(範囲)の設定値

エリア名	中心座標			中心座標傾き			大きさ(範囲)		
	X	Y	Z	RX	RY	RZ	DX	DY	DZ
エリア1	400	400	200	200	-300	200	600	-200	200
エリア2	0	0	0	0	0	45	0	0	0
エリア5	100	100	50	100	100	100	50	500	100

■設定内容

エリアの中心位置と大きさ(範囲)をそれぞれ位置変数、ベクトル変数で指定します。

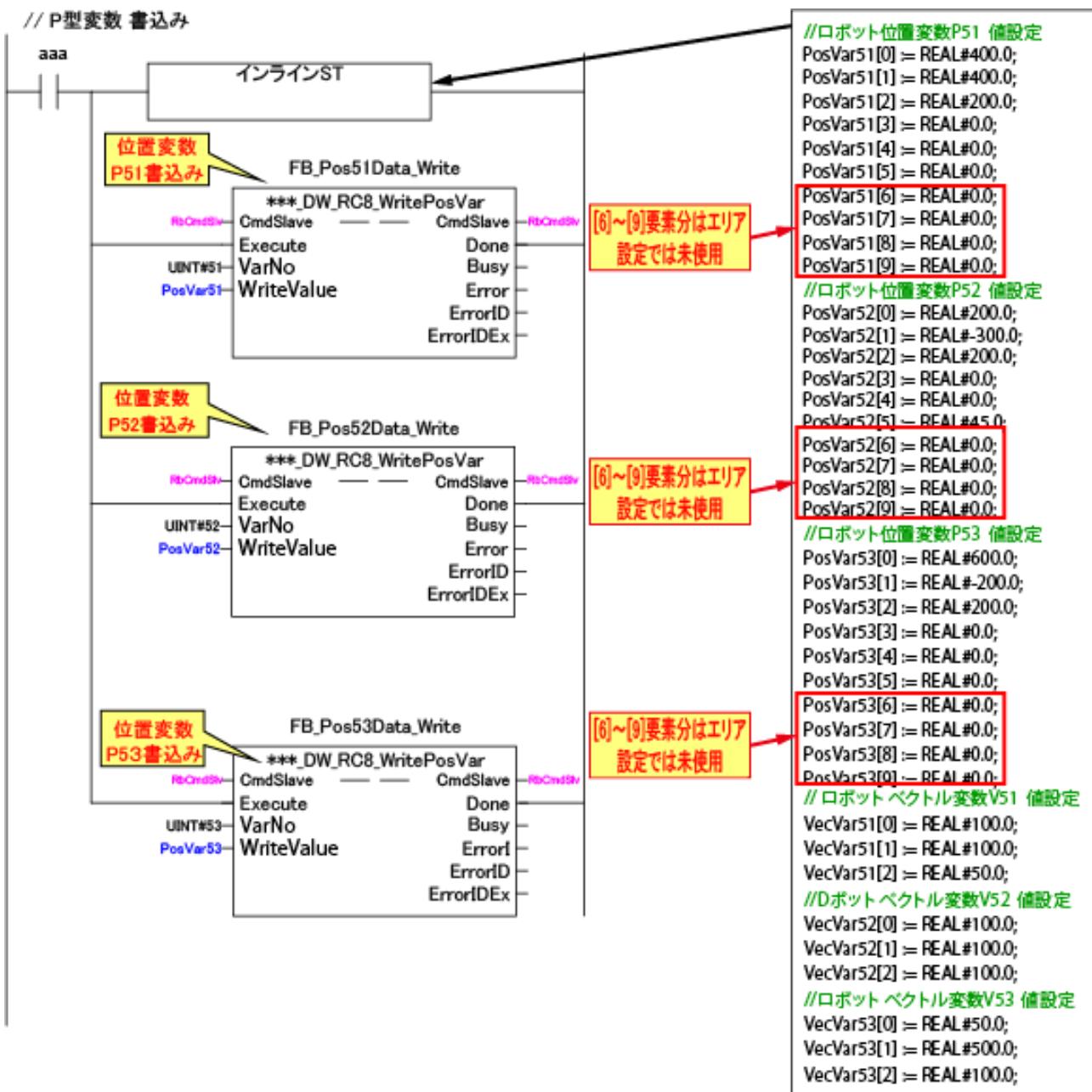
エリア名	中心位置	大きさ(範囲)
エリア1	P51	V51
エリア2	P52	V52
エリア3	P53	V53

■ローカル変数

名称	データ型	備考
RbCmdSlv	sRobotCmdSlave	RC8Command-Slave変数
PosVar51	ARRAY[0..9]OF REAL	RC8位置変数 P51に書込む値を定義した変数
PosVar52	ARRAY[0..9]OF REAL	RC8位置変数 P52に書込む値を定義した変数
PosVar53	ARRAY[0..9]OF REAL	RC8位置変数 P53に書込む値を定義した変数
VerVar51	ARRAY[0..2]OF REAL	RC8位置変数 V51に書込む値を定義した変数
VerVar52	ARRAY[0..2]OF REAL	RC8位置変数 V52に書込む値を定義した変数
VerVar53	ARRAY[0..2]OF REAL	RC8位置変数 V53に書込む値を定義した変数

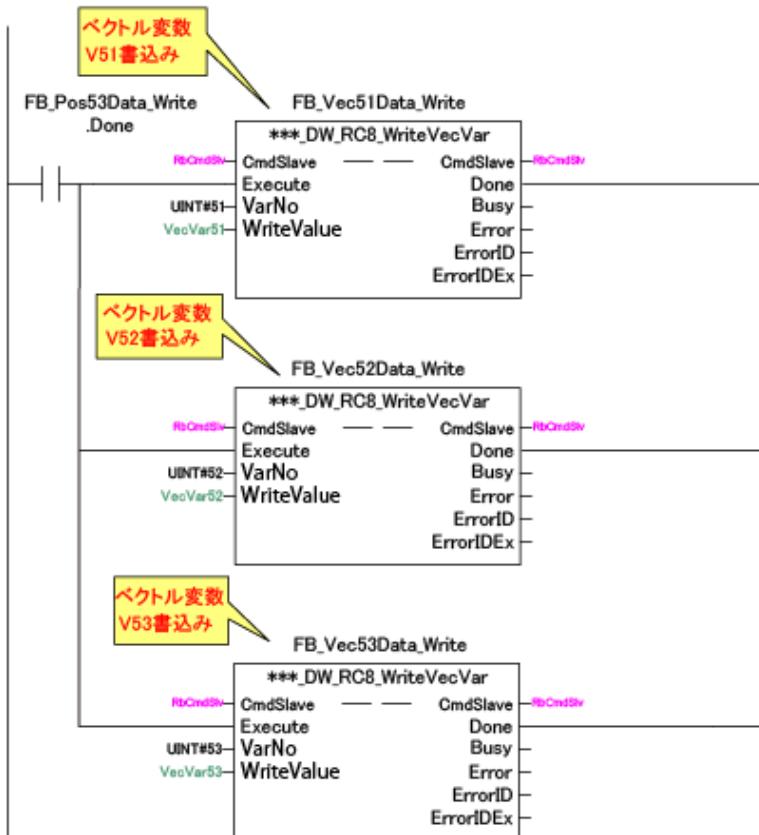
RC8Command-Slave 活用ガイド

■ ラダー図

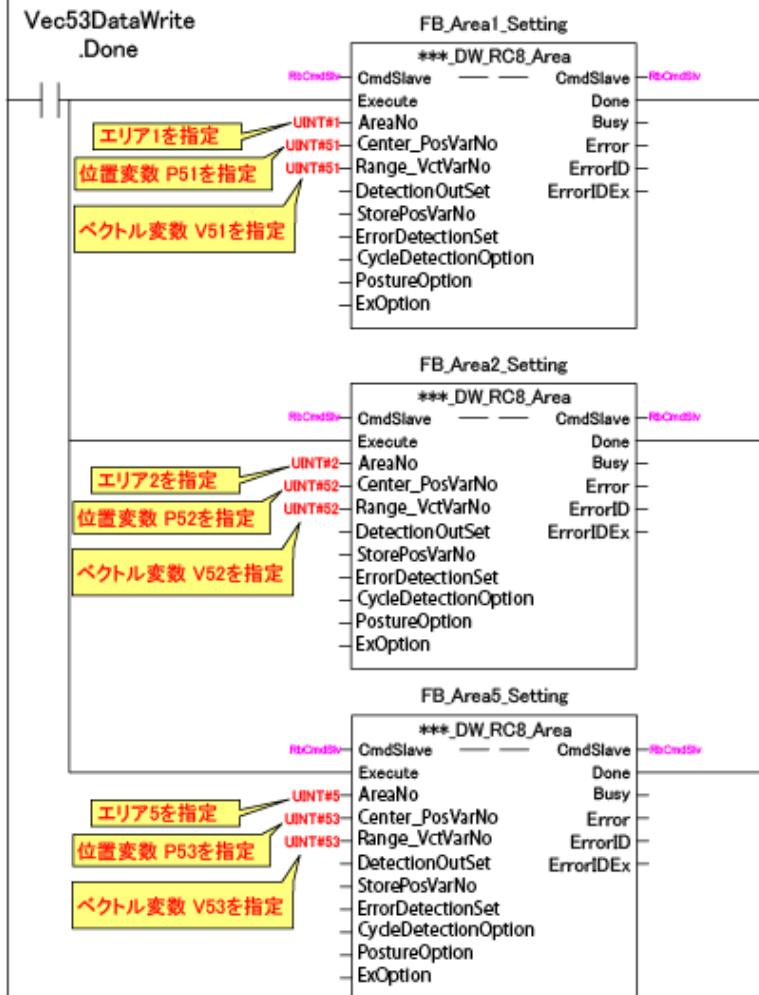


RC8Command-Slave 活用ガイド

// V型変数 書込み



// エリア設定



3.1.1. エリアの検出時の出力の設定

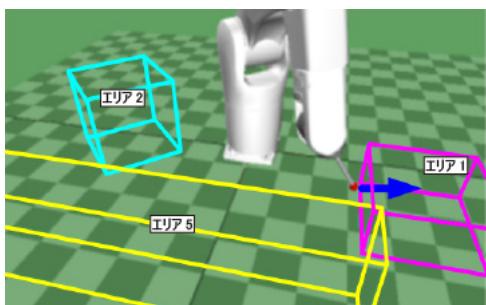
「エリア 1」、「エリア 2」、「エリア 3」の検知時の出力を Command-Slave で設定します。

■ エリア検知出力の設定値

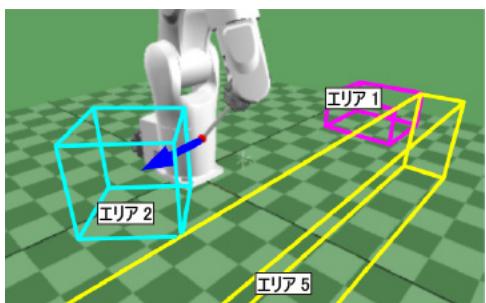
エリア名	エリア検知出力設定
エリア1	有効
エリア2	有効
エリア3	無効

■ 設定内容

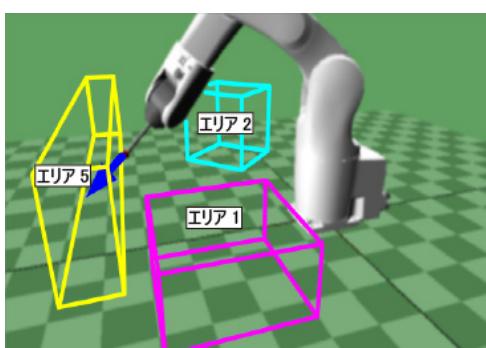
- (1) ロボットツール座標の原点がエリア 1 の範囲内に入った場合は、「エリア 1 出力」を出力します。



- (2) ロボットツール座標の原点がエリア 2 の範囲内に入った場合は、「エリア 2 出力」を出力します。



- (3) ロボットツール座標の原点がエリア 5 の範囲内に入った場合は、「エリア 5 出力」を出力しません。



RC8Command-Slave 活用ガイド

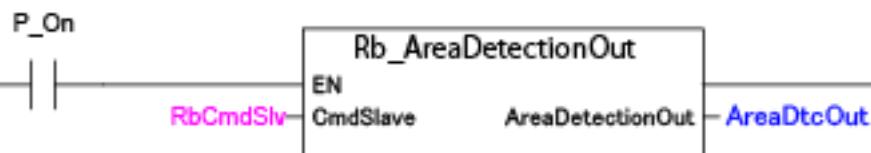
■ ローカル変数

名称	データ型	備考
RbCmdSlv	sRobotCmdSlave	RC8Command-Slave 変数
AreaDtcOut	ARRAY[0..31]OF BOOL	エリア検知出力の格納変数
Area1Pos	ARRAY[0..9]OF REAL	エリア 1 で検知した位置を取込む変数
Area2Pos	ARRAY[0..9]OF REAL	エリア 2 で検知した位置を取込む変数
Area5Pos	ARRAY[0..9]OF REAL	エリア 5 で検知した位置を取込む変数

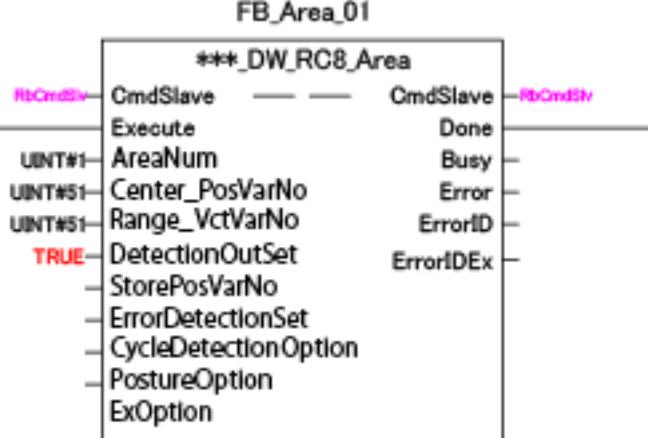
RC8Command-Slave 活用ガイド

■ ラダー図

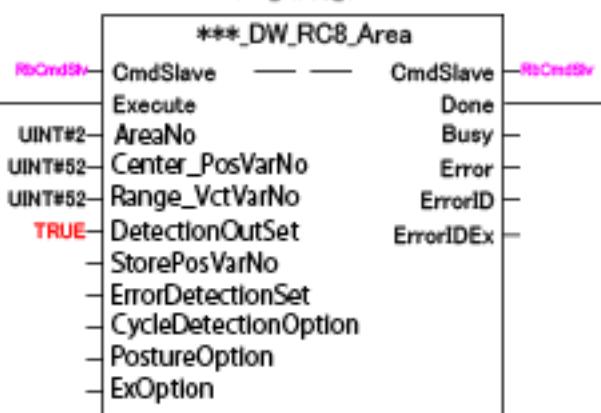
//エリア検知出力 有効



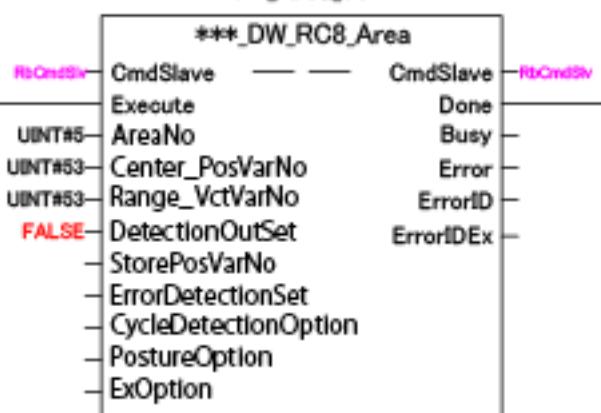
//エリア設定

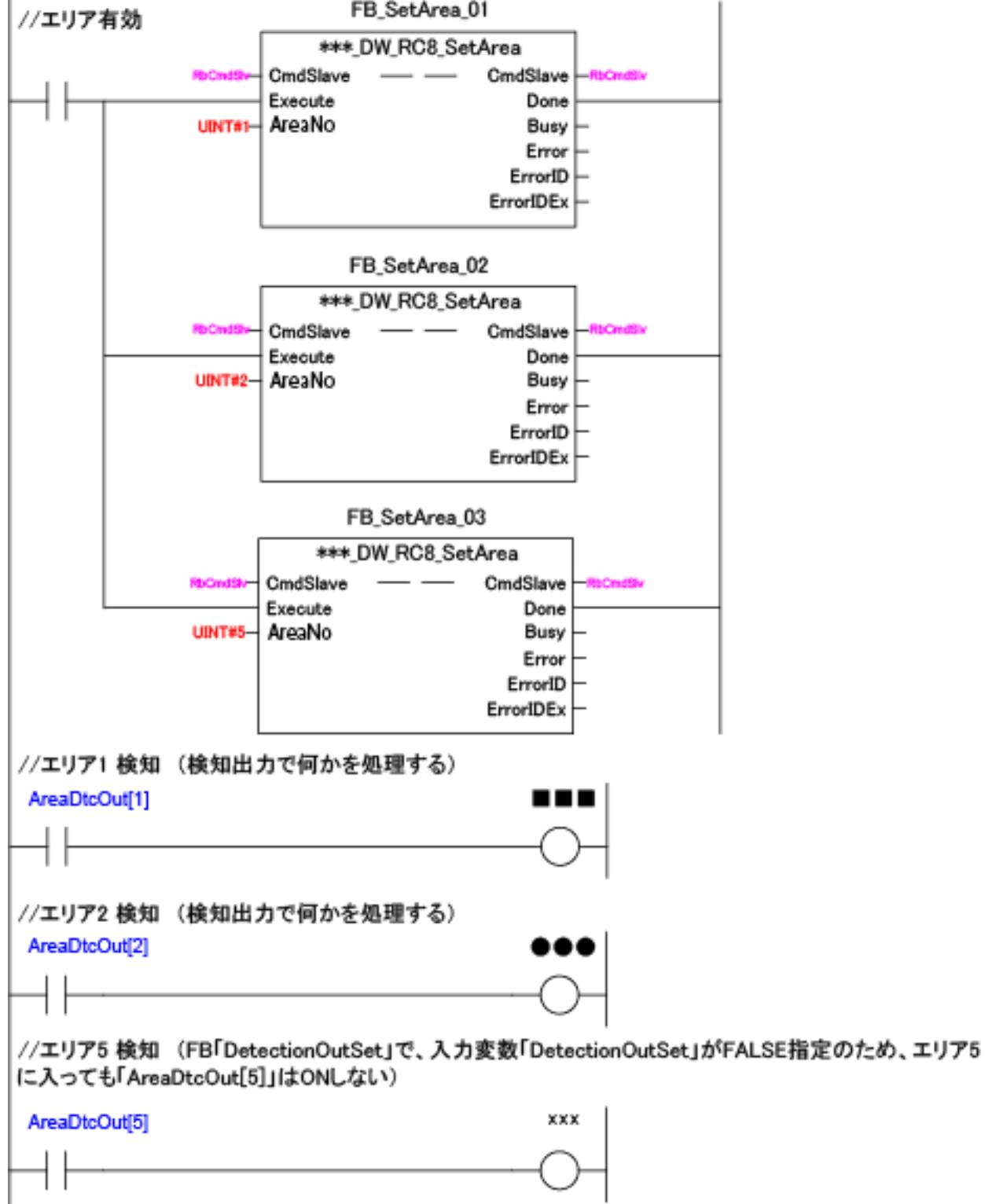


FB_Area_02



FB_Area_03





3.1.2. エリアの検知時のロボット位置を位置変数に取り込む設定

「エリア 1」、「エリア 2」、「エリア 3」の検知時のロボット位置を位置変数に取り込むように Command-Slave で設定します。

■検知時位置に格納する位置変数

エリア名	検知時位置格納位置変数
エリア1	P61
エリア2	P62
エリア3	P63

■設定内容

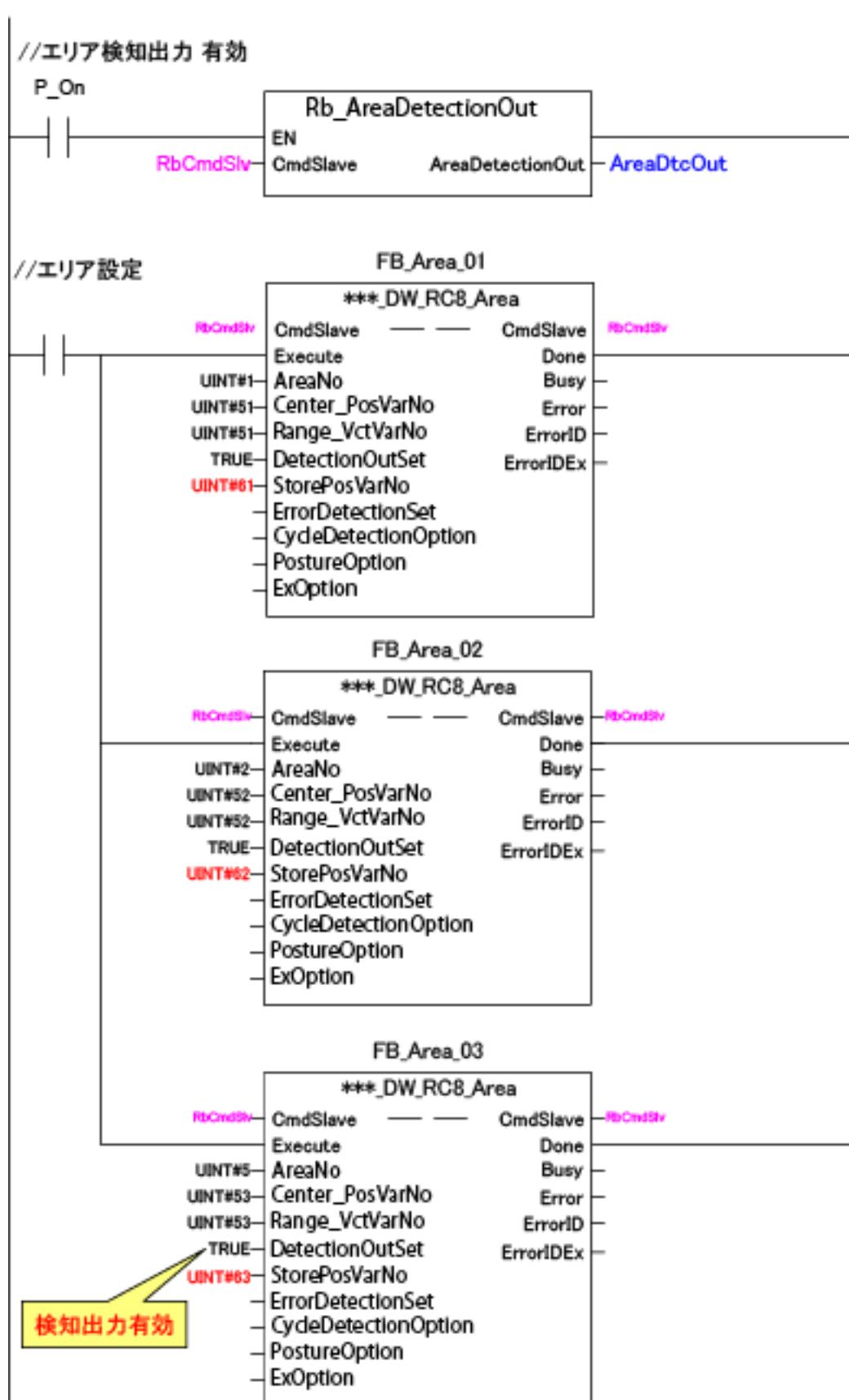
- ロボットツール座標の原点がエリア 1 の範囲に入った場合は、位置変数 P61 にロボット座標を格納します。
- ロボットツール座標の原点がエリア 2 の範囲に入った場合は、位置変数 P62 にロボット座標を格納します。
- ロボットツール座標の原点がエリア 5 の範囲に入った場合は、位置変数 P63 にロボット座標を格納します。

■ローカル変数

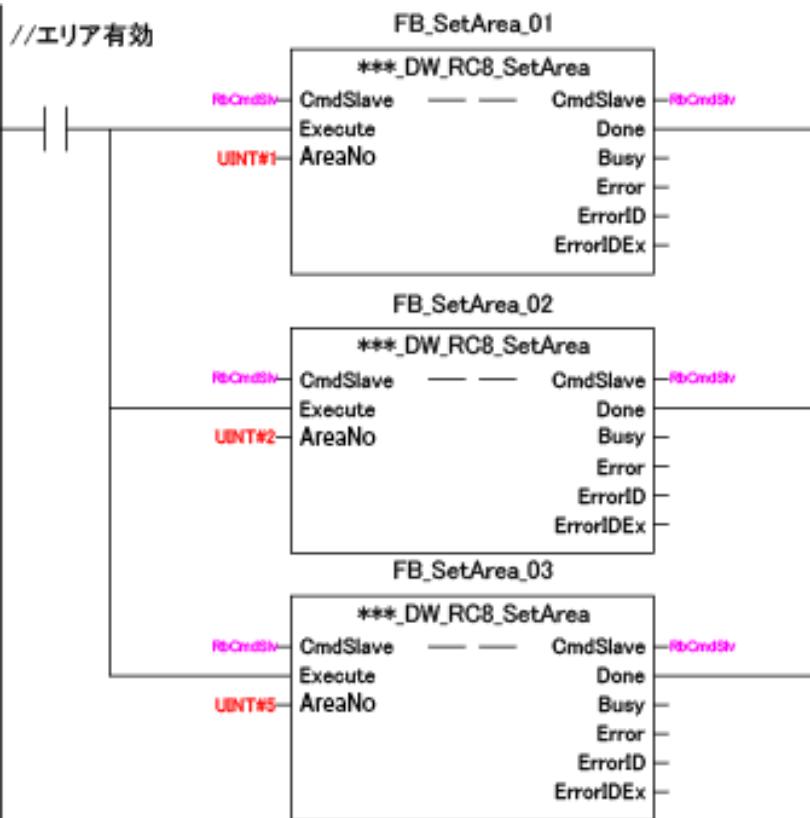
名称	データ型	備考
RbCmdSlv	sRobotCmdSlave	RC8Command-Slave 変数
AreaDtcOut	ARRAY[0..31]OF BOOL	エリア検知出力の格納変数
Area1Pos	ARRAY[0..9]OF REAL	エリア 1 で検知した位置を取込む変数
Area2Pos	ARRAY[0..9]OF REAL	エリア 2 で検知した位置を取込む変数
Area5Pos	ARRAY[0..9]OF REAL	エリア 5 で検知した位置を取込む変数

RC8Command-Slave 活用ガイド

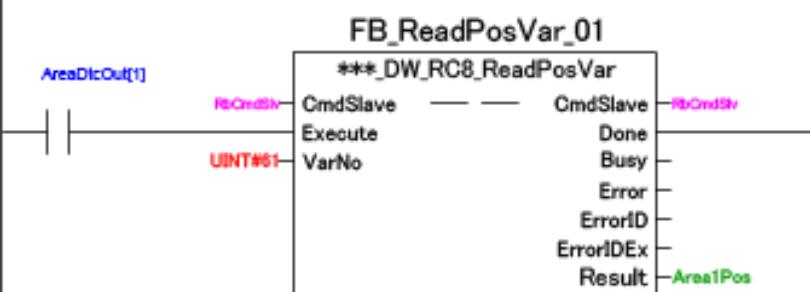
■ ラダー図



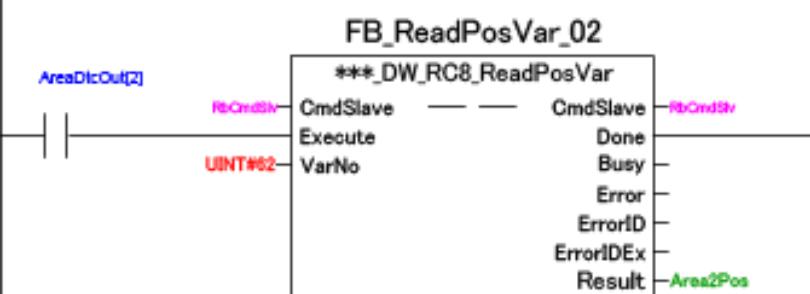
RC8Command-Slave 活用ガイド



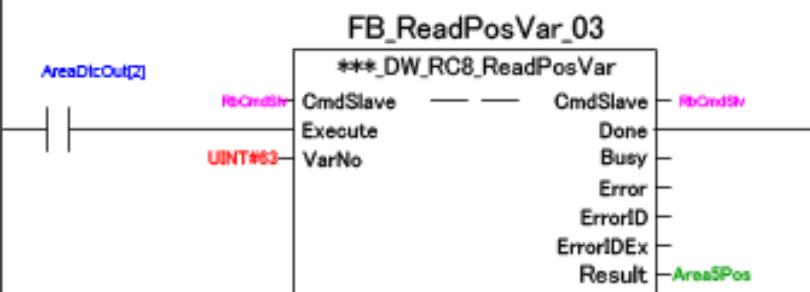
//エリア1検知（検知した位置「P61」を内部変数「Area1Pos」に読み込む）



//エリア2検知（検知した位置「P62」を内部変数「Area2Pos」に読み込む）



//エリア5検知（検知した位置「P63」を内部変数「Area5Pos」に読み込む）



3.1.3. エリア検知時エラー検出の設定

「エリア 1」、「エリア 2」、「エリア 3」のエリア検知時のエラー検出を Command-Slave で設定します。

■エラー検出の設定

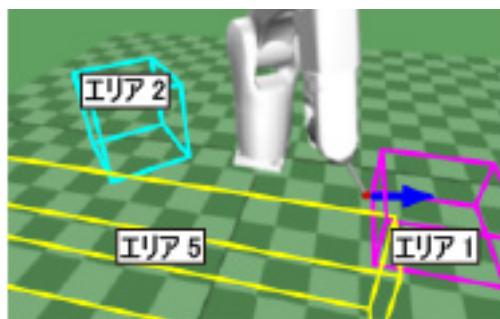
エリア名	設定値
エリア1	0
エリア2	2
エリア3	5

■設定値と内容

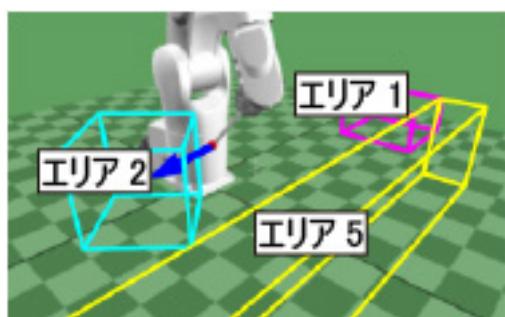
設定値	内容
0	エリア領域内部に干渉時、エラー出力無し
1	エリア領域内部に干渉時、エラー出力(干渉時モータON不可)
2	エリア領域内部に干渉時、エラー出力(干渉時、手動モードでのみモータON可)
3	エリア領域外部に干渉時、エラー出力無し
4	エリア領域外部に干渉時、エラー出力(干渉時モータON不可)
5	エリア領域外部に干渉時、エラー出力(干渉時、手動モードでのみモータON可)

■設定内容

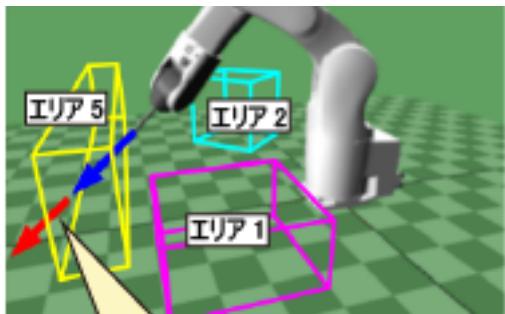
(1)ロボットツール座標の原点がエリア 1 の範囲内に入った場合は、エラー出力をしません。



(2)ロボットツール座標の原点がエリア 2 の範囲内に入った場合は、エラー出力をします。



(3) ロボットツール座標の原点がエリア 5 の範囲外に出た場合は、エラー出力をします。



■ エリア 5 の範囲外に出た場合は、
エラー出力します。

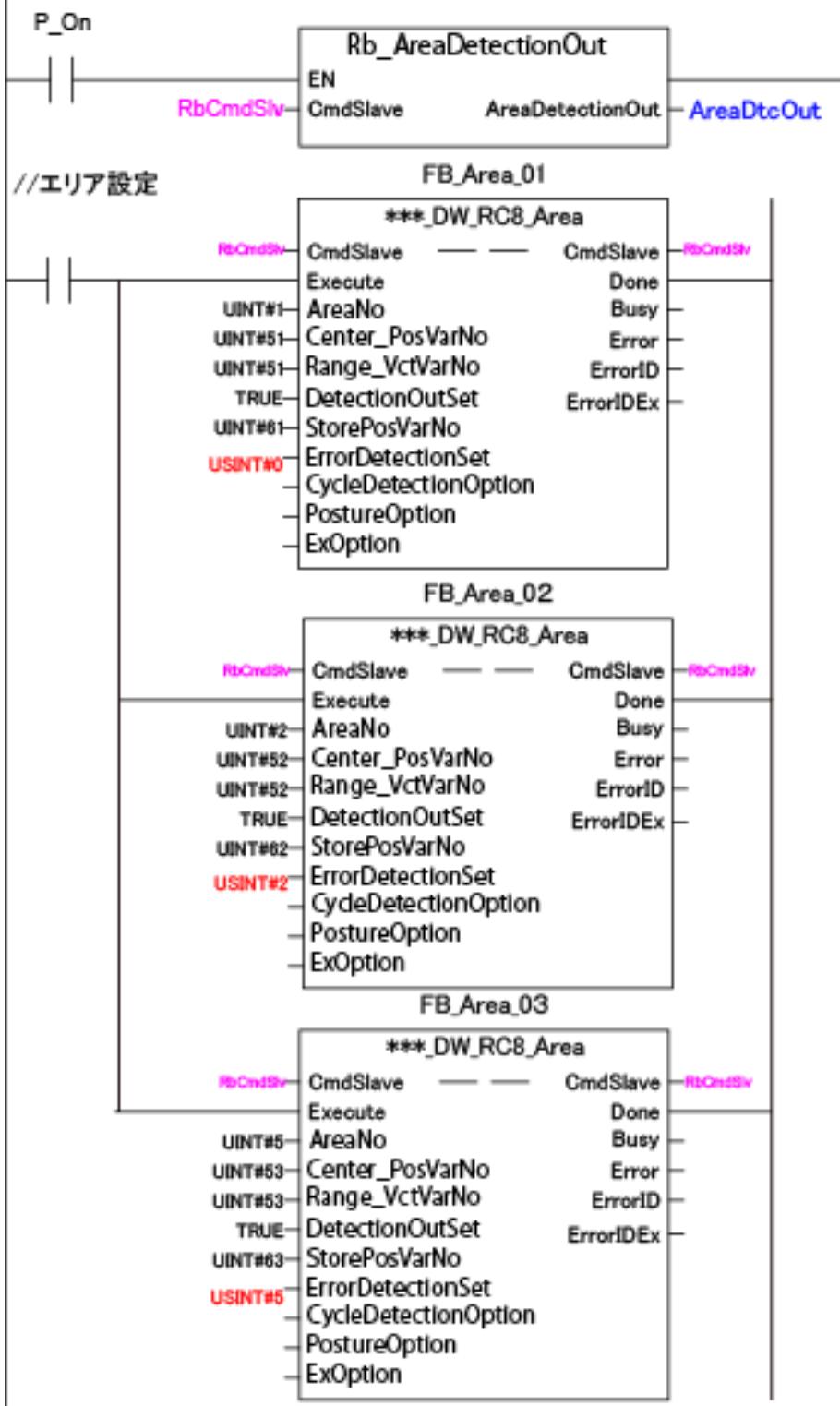
■ ローカル変数

名称	データ型	備考
AreaDtcOut	ARRAY[0..31]OF BOOL	エリア検知出力の格納変数

RC8Command-Slave 活用ガイド

■ ラダー図

//エリア検知出力 有効



3.2. パレタイジング

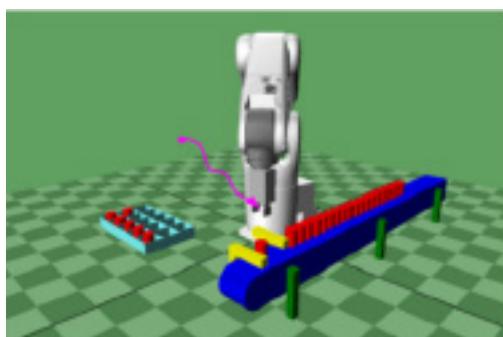
3.2.1. 作業内容

ロボットは以下のように、パレットの箱詰め位置を取得して、コンベアからワークを取り出して、パレットに箱詰めをします。

パレットの条件		
横	縦	段数
3分割	5分割	1段

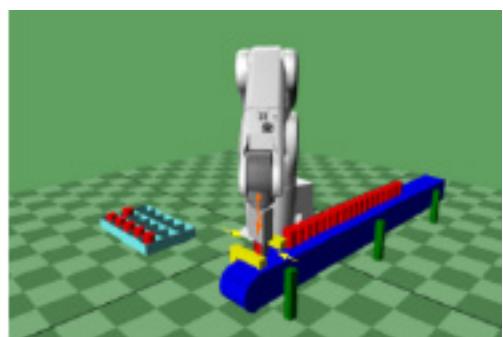
STEP1:

パレットの箱詰め位置を取得して、ワーク上空へ移動します。



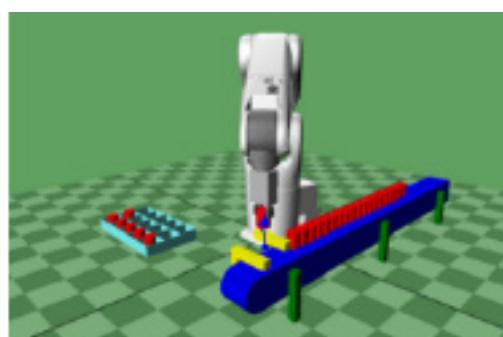
STEP2:

ワーク位置へ移動して、ワークをチャックします。



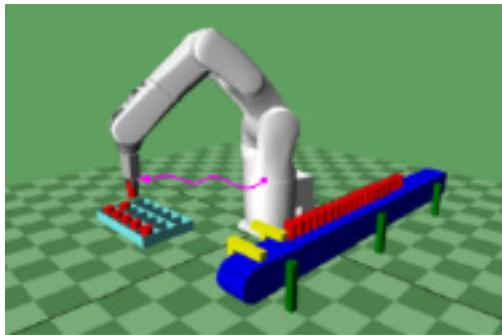
STEP3:

ワーク位置から上空へ i 移動します。



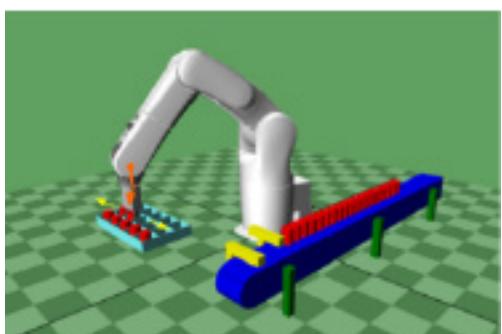
STEP4:

パレット箱詰め位置から上空へ移動します。



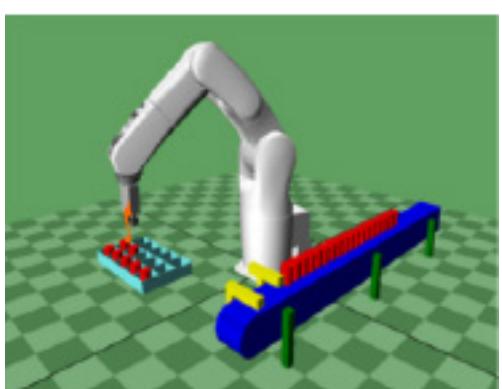
STEP5:

パレット箱詰め位置へ移動して、ワークをアンチャックします。



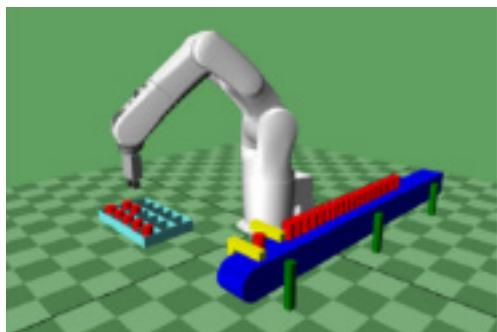
STEP6:

パレット箱詰め位置から上空へ移動します。



STEP7:

次のワークがコンベア取り出し位置にあるかを確認します。

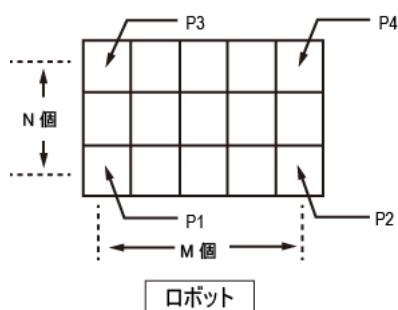


3.2.2. プログラミング例

RC8 でのパラメータの決め方とプログラミングのしかたを説明します。

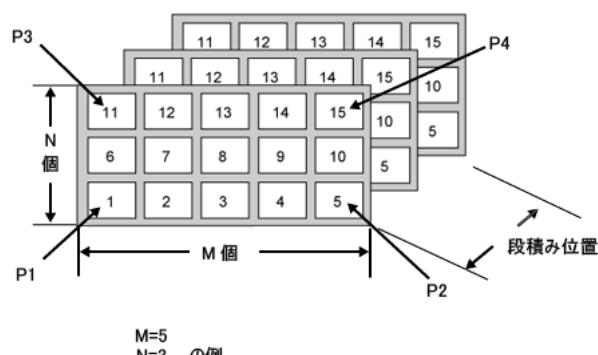
■パレタイジングのパラメータ

パレットの条件を指定し、その条件で何段目のパレットの先頭から何番目かを指定すると、その位置を取得できます。



パレットの条件は、縦分割数(N)、横分割数(M)、パレットの高さ、パレットの四隅位置(P1～P4)で指定しています。

パレットの位置番号は下図のように P1 から P2 に向かって振られ、P2 に達すると、P1 から P3 へ向かって 1 プラスした位置から横に振られています。



RC8Command-Slave 活用ガイド

■実施例を RC8 でプログラミングした場合

```
' ロボットプログラミング (Pcs 言語) でパレタイジング実行する場合のサンプル
,
' ≪パレット条件≫ //////
' 横分割 3
' 縦分割 5
' パレット高さ 30mm
' パレット四隅位置変数 P11、P12 、P13 、P14 を使用
' パレット目標位置 (3×5=15 位置) I 型変数 I11 でインクリメント (パレタイジングカウンタ)
' パレット段積 1 段
' ≪コンベア位置≫ //////
,
' ワーク取り出し位置 P10

Sub Main

Dim PalletPos As Position

TakeArm

' --- パレット目標位置カウント確認-----
If I11 > 15 Then I11 = 1

' --- パレット箱詰め位置の抽出-----
PalletPos = Pallet.CalcPos( 3, 5, 30, P11, P12, P13, P14, I11, 1 )

' --- コンベアからワーク取り出し-----
Approach P, P10, @P 100 ,
Move L, @0 P10

' ハンドチャック

Depart L, 100
,
' --- パレット箱詰め処理-----
Approach P, PalrPos, @P 100 ,

Move L, @0 PalrPos

' ハンドアンチャック
Depart L, 100

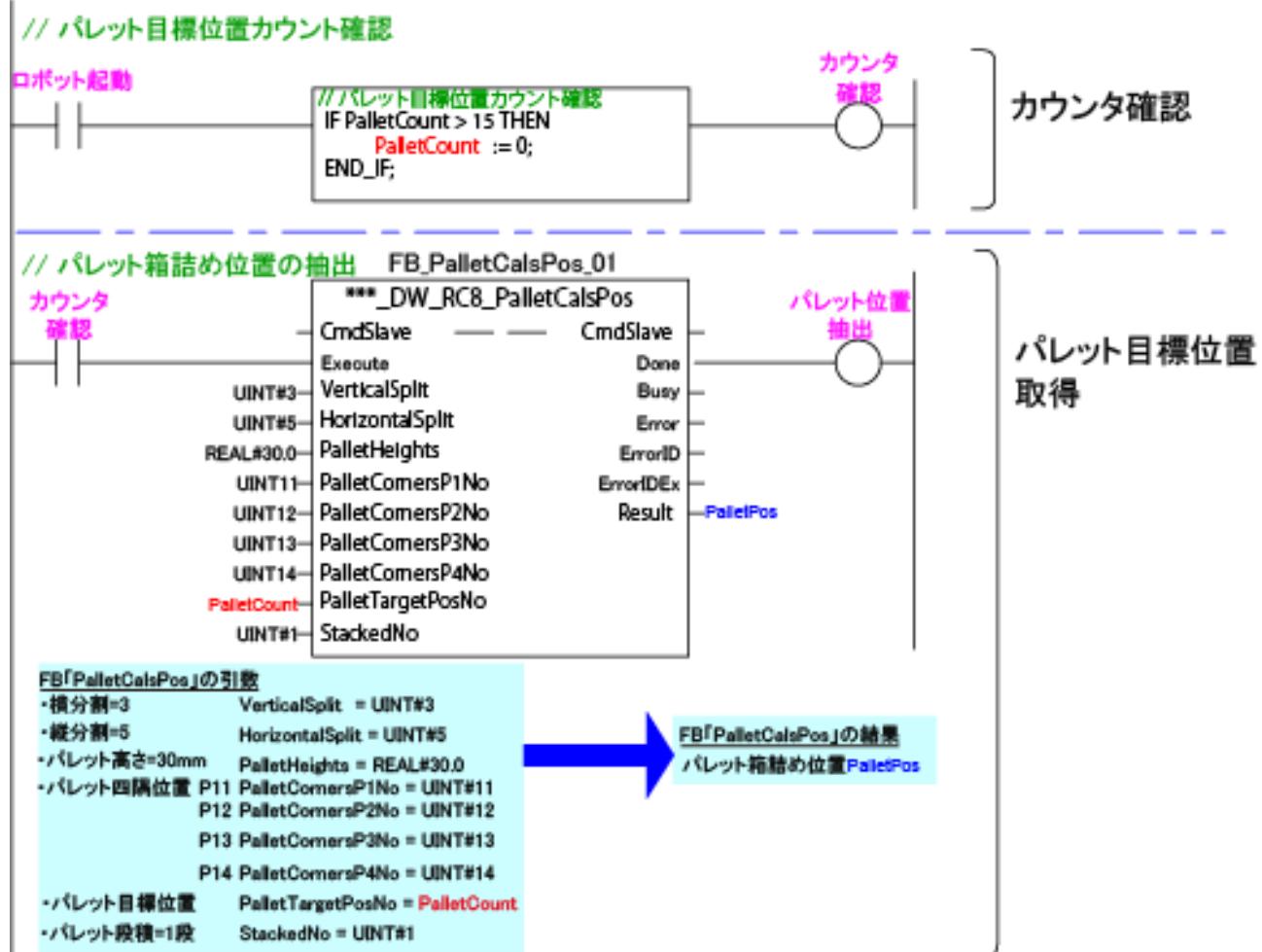
' --- パレタイジングカウンタのインクリメント-----
I11 = I11 + 1
End Sub
```

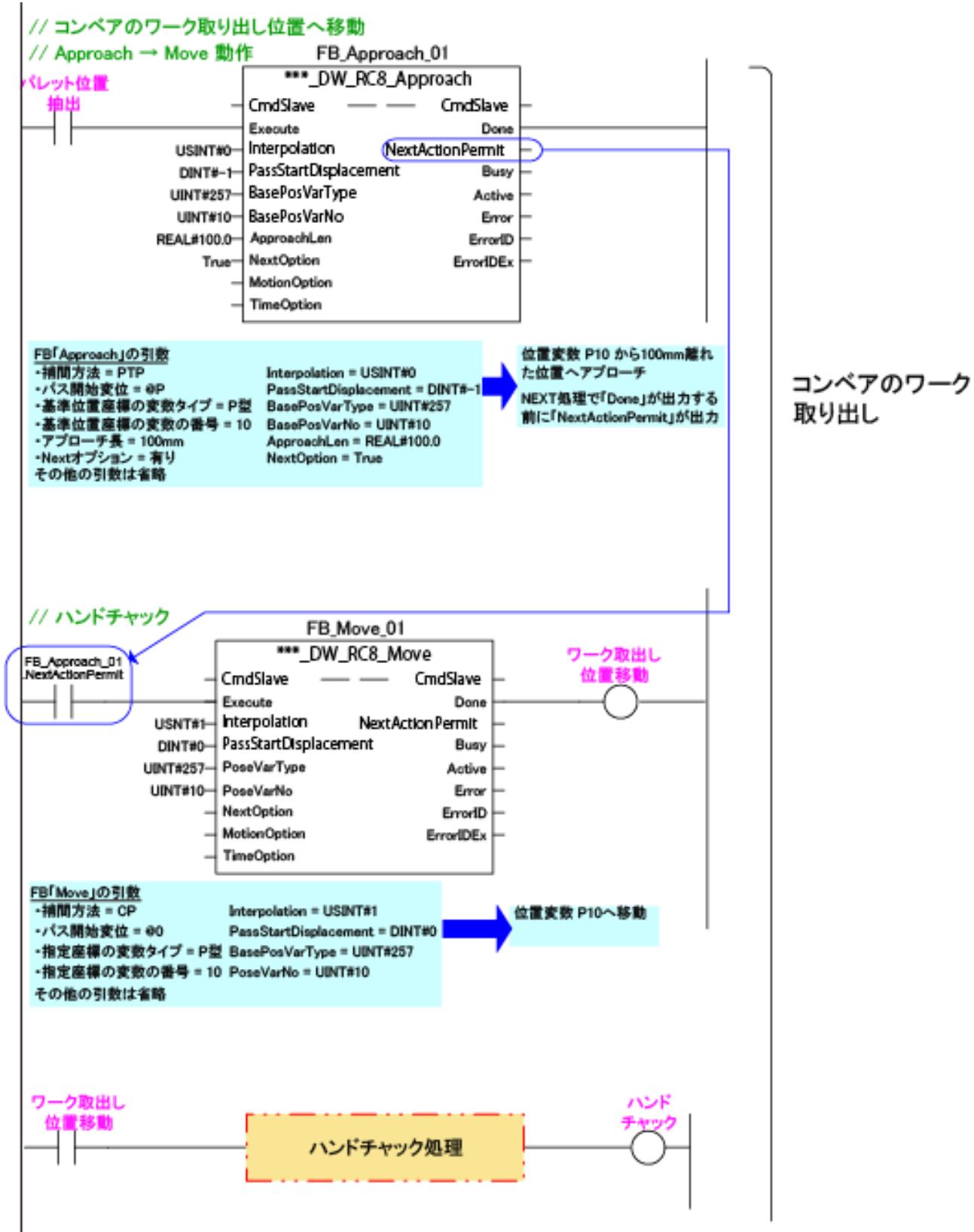
3.2.3. Command-Slave で実施例を実行する場合

■ ローカル変数

名称	データ型	備考
PalletPos	ARRAY[0..9] OF REAL	パレット位置格納用 (パレット目標位置)
PalletCount	UINT	パレット目標位置カウンタ
FB_PalletCalsPos_01	***_DW_RC8_PalletCalsPos	パレット位置取得 FB インスタンス
FB_Approach_01	***_DW_RC8_Approach	アプローチ FB(変数指定) インスタンス
FB_Move_01	***_DW_RC8_Move	MOVE_Fb(変数指定) インスタンス
FB_ApproachImdt01	***_DW_RC8_ApproachImdt	アプローチ FB(直値指定) インスタンス
FB_MoveImdt_01	***_DW_RC8_MoveImdt	MOVE_Fb(直値指定) インスタンス
FB_Depart_01	***_DW_RC8_Depart	デパート FB インスタンス
FB_Depart_02	***_DW_RC8_Depart	デパート FB インスタンス

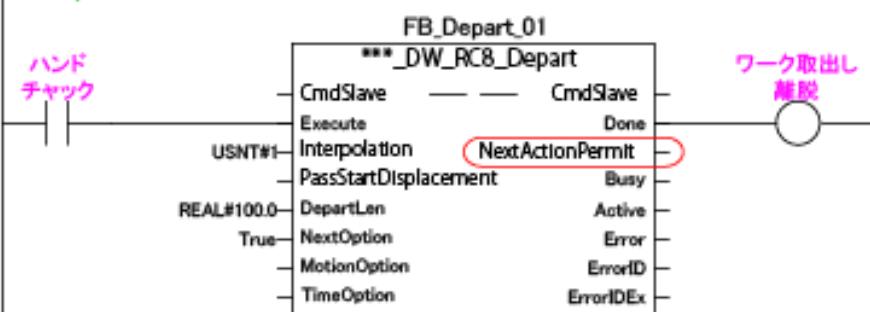
■ ラダー図





RC8Command-Slave 活用ガイド

// コンベアのワーク取り出し位置から離れる
// Depart動作



FB「Depart」の引数

- ・補間方法 = CP
- ・パス開始変位 = USINT#1
- ・デパート長 = 100mm
- ・Interpolation = USINT#1
- ・DepartLen = REAL#100.0
- ・Nextオプション = 有り
- ・MotionOption = True
- その他の引数は省略

位置変数 P10 から

100mm -Z方向へ移動

NEXT処理で「Done」が outputする前に
「NextActionPermit」が出力

// パレット目標位置へ移動
// Approach → Move 動作



FB「Approach2」の引数

- ・補間方法 = PTP
- ・パス開始変位 = @P
- ・基準位置座標のタイプ = P型
- ・基準位置 = PalletPos
- ・アプローチ長 = 100mm
- ・Nextオプション = 有り
- その他の引数は省略

Interpolation = USINT#0

PassStartDisplacement = DINT#-1

BasePosType = UINT#257

BasePos =

PalletPos

ApproachLen = REAL#100.0

NextOption = True

パレット目標位置 PalletPos から

100mm 離れた位置へアプローチ

NEXT処理で「Done」が outputする前に
「NextActionPermit」が出力

パレットへワーク
を供給

FB「Move2」の引数

- ・補間方法 = CP
- ・パス開始変位 = @0
- ・指定座標(ポーズ)タイプ = P型
- ・移動量 = PalletPos
- その他の引数は省略

FB「Move2」の引数

- Interpolation = USINT#1
- PassStartDisplacement = DINT#0
- PoseType = UINT#257
- MoveDistance = PalletPos

パレット位置 移動

パレット目標位置 PalletPos へ移動

// ハンドアンチャック

パレット位置
移動

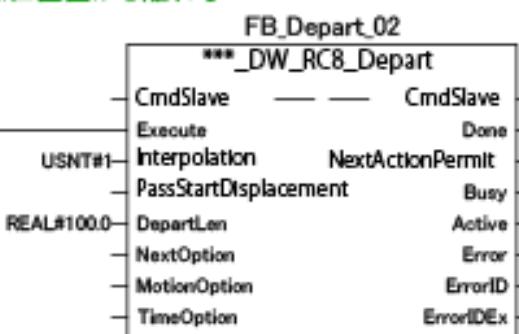
ハンドアンチャック処理

ハンド
アンチャック

// パレット目標位置から離れる

// Depart動作

ハンド
アンチャック



パレット位置
離脱

パレットへワーク
を供給

FB「Depart」の引数

- ・補間方法 = CP
- ・デパート長 = 100mm
- その他の引数は省略

Interpolation = USINT#1
DepartLen = REAL#100.0

パレット目標位置 PalletPos から
100mm -Z 方向へ移動

// パレタイジングカウンタのインクリメント

パレット位置
離脱

// パレタイジングカウンタのインクリメント
PalletCount := PalletCount + 1;

カウンタ
インクリ

カウンタ
の
インクリメント

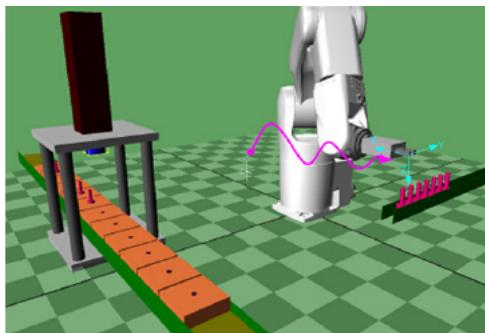
3.3. 力制御

3.3.1. 作業内容

ロボットは以下のように、ワークを把持し圧入を補佐します。圧入時にコンプライアンス機能を有効にすることで、プレス方向に倣います。

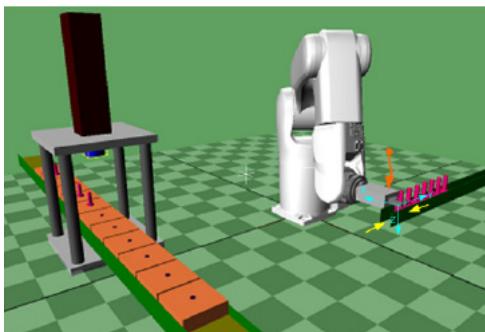
STEP1

ツール座標系を TOOL1 に変更して、ワーク上空位置(P1 のアプローチ位置)へ移動します。



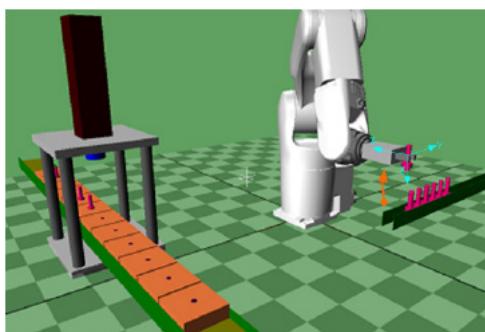
STEP2

ワーク位置(P1)へ移動して、ワークをチャックします。



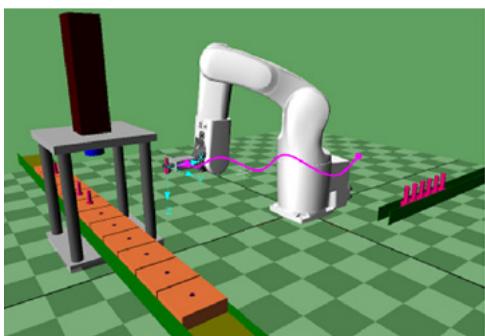
STEP3

ワーク位置から上空(P1 からツール座標-Z 方向)へ移動します。



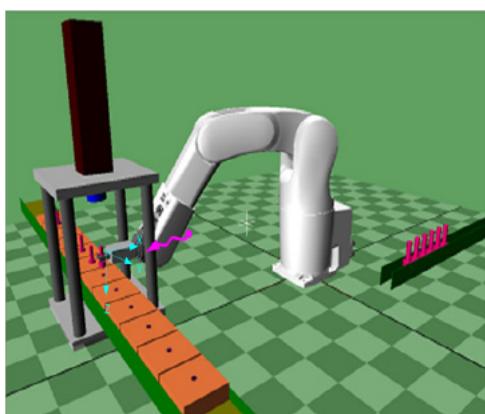
STEP4

プレス前位置(P2)へ移動します。



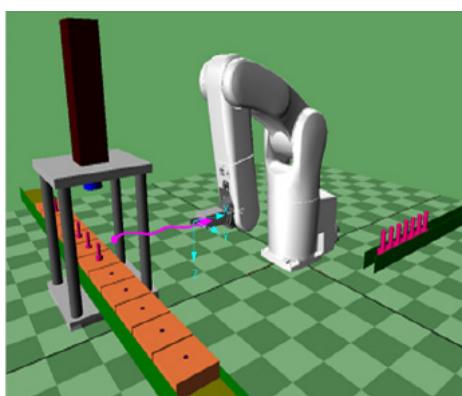
STEP5

圧入上空位置(P3 のアプローチ位置)へ移動します。



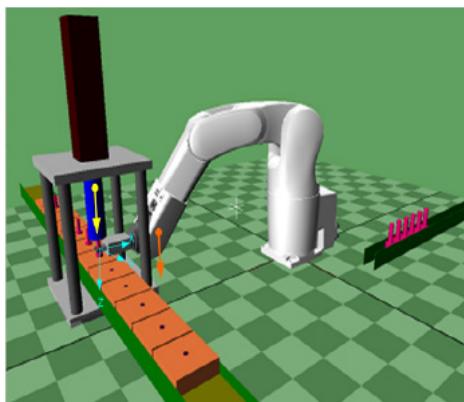
STEP6

圧入位置(P3)へ移動し、コンプライアンス機能を有効にします。プレスを起動します。



STEP7

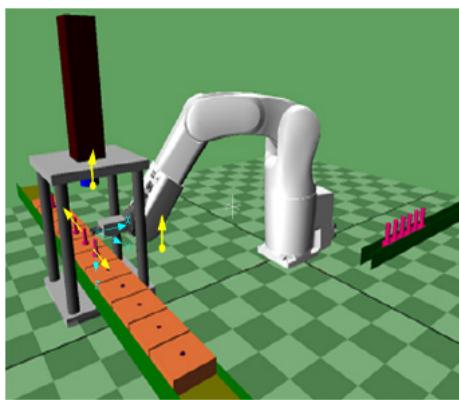
圧入が開始されます。この時コンプライアンス機能が有効であるため、ロボットはプレス方向に倣います。



STEP8

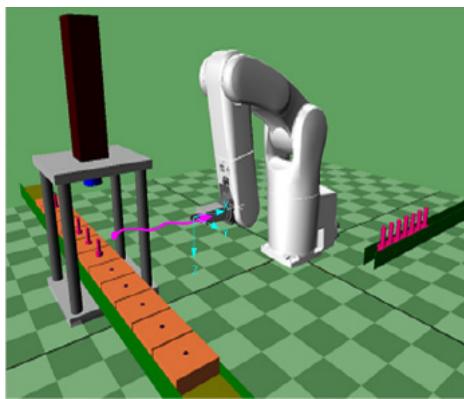
圧入完了後、プレスを上昇させ、コンプライアンス機能を無効にします。

ワークをアンチャックし、圧入位置から上空(P3 からツール座標-Z 方向)へ戻ります。



STEP9

プレス前位置(P2)へ戻ります。



3.3.2. プログラミング例

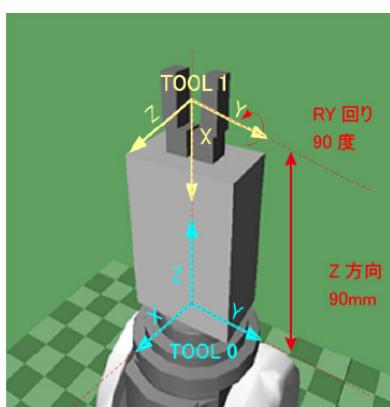
各パラメータの決め方とプログラミングのしかたを説明します。

■TOOL 座標

実施例のハンドは TOOL 座標の Z 方向に 90mm オフセットしています。

また、ワーク取り出しや圧入位置の状態で TOOL 座標の RY 回りに 90deg 回転しています。

TOOL1 座標 : X=0 , Y=0 , Z=90 , RX=0 , RY=90 , RZ=0

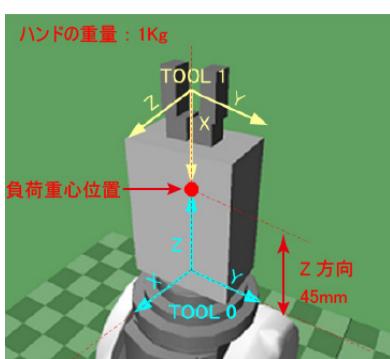


■内部負荷条件

コンプライアンス機能は実際のハンドとワークの質量と重心位置が必要です。

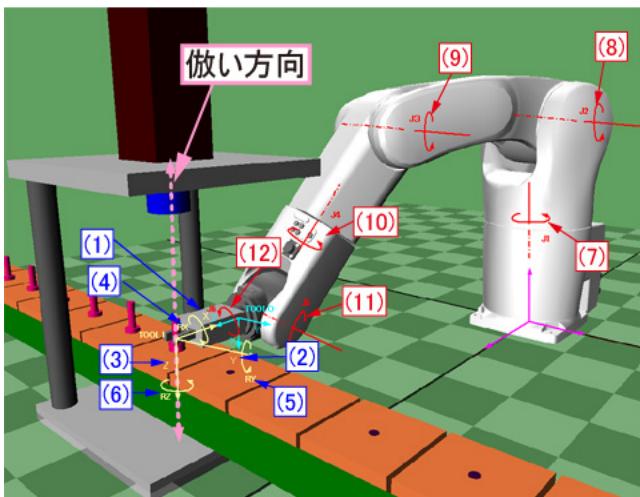
先端負荷質量 : 1kg (ハンドの重量)

負荷重心位置 : X = 0 mm, Y = 0 mm, Z = 45 mm



■ パラメータ詳細

下図のロボット姿勢で、「TOOL1」が Z 方向にコンプライアンス(倣い方向)を設定する場合の各パラメータを説明します。



座標系パラメータ

No.	TOOL 座標系	目標制御力	偏差許容値
(1)	X	100[N]	10[mm]
(2)	Y	100[N]	10[mm]
(3)	Z	0[N]	100[mm]
(4)	RX	10[Nm]	10[deg]
(5)	RY	10[Nm]	10[deg]
(6)	RZ	10[Nm]	10[deg]

各軸パラメータ

No.	軸	軸電流制限値	軸偏差許容値
(7)	1軸	50[%]	50[deg]
(8)	2軸	30[%]	100[deg]
(9)	3軸	30[%]	100[deg]
(10)	4軸	50[%]	50[deg]
(11)	5軸	30[%]	100[deg]
(12)	6軸	50[%]	50[deg]

力、電流制限の値はあくまで参考値です。

姿勢、倣い方向、ハンドの重量などにより上記表の値では柔らかくなりすぎる場合があります。実機で現状による調整が必要です。

他に現状で調整が必要なパラメータとして粘性、柔らかさがあります。

RC8Command-Slave 活用ガイド

● 座標系パラメータ

目標制御力(力) 偏 差 許 容 値 (PosEralw=位置偏差値)	ロボットを制御するための力で、設定した力以上は出力しません。(範囲は0~) この値を小さくすると、小さな外力に倣って動作するようになります。 実施例ではZ方向を0[N]に設定することで、Z方向の外力に倣いやすくなります。X方向、Y方向は100[N]、RX、RY、RZは10[Nm]に設定することで、他の方向と回転系はサーボ保持が効いている状態になります。 手先のサーボ許容偏差値を設定します。(範囲は0~) この値を大きくする事で、倣い動作によって偏差が増加しても偏差関係のエラーは発生しなくなります。 実施例ではZ方向を100[mm]に設定することで、100[mm]まで倣い動作で倣うことが出来ます。X方向、Y方向は10[mm]、RX、RY、RZは10[deg]の設定は、外力による多少の偏差許容を考慮しています。
---	---

● 各軸パラメータ

軸電流制限値 (CurLmt=電流制限値)	各軸モータのトルク値(電流値)を設定します。(範囲は0~100[%]) 100[%]が定格で、値を小さくすることでトルク値(電流値)を下げ柔らかさを実現します。 実施例ではTOOL座標系のZ方向に倣わす場合、2軸、3軸、5軸に負荷が掛るため30[%]に設定します。1軸、4軸、6軸は多少の負荷が掛るため50[%]に設定します。
軸偏差許容値 (Eralw=各軸偏差許容)	各軸のサーボ許容偏差値を設定します。(範囲は0~) 電流を制限した軸に対し、外力を受けると各関節が回転し偏差が増加しても偏差関係のエラーは発生しなくなります。 実施例ではTOOL座標系のZ方向に倣わす場合、2軸、3軸、5軸に負荷が掛るため100[deg]に設定します。1軸、4軸、6軸は多少の負荷が掛るため50[deg]に設定します。

■ ティーチング位置

実施例はTOOL1の時のティーチング位置です。

P1	ワーク取出し位置	X=200、Y=460、Z=220、RX=180、RY=0、RZ=-90、FIG=1
P2	プレス前位置	X=390、Y=-200、Z=180、RX=180、RY=0、RZ=-180、FIG=1
P3	圧入位置	X=600、Y=-200、Z=105、RX=180、RY=0、RZ=-180、FIG=1

RC8Command-Slave 活用ガイド

■実施例を RC8 でプログラミングした場合

メインプログラム(Samp.pcs)

```
#Include "CompOn.pcs"
#include "CompOff.pcs"

Sub Main
    TakeArm

    ' --- 内部負荷 ---
    V1 = V(0, 0, 45)      ' V1 に重心位置を書き込み
    PayLoad 1000 , V1     ' 質量 1000g、重心位置を V1 で設定

    ' --- TOOL1 ---
    P0 = P(0, 0, 90, 0, 90, 0)  ' P0 に TOOL1 に設定する値を書き込み
    Tool 1, P0              ' TOOL1 を P0 の値で設定
    ChangeTool 1             ' TOOL1 変更

    ' --- ワーク取り出し ---
    Approach P, P1, @P 80
    Move L, @P P1
    ' ハンドチャック
    Depart L, @P 80

    ' --- プレス前位置へ移動 ---
    Move P, @P P2

    ' --- 圧入位置 ---
    Approach P, P3, @P 70
    Move L, @P P3

    Call CompOn      ' コンプライアンス有効
    ' プレス起動
    ' 圧入中
    ' 圧入完了
    ' プレス上昇

    Call CompOFF     ' コンプライアンス無効
    ' ハンドアンチャック
    Depart L, @P 70

    ' --- プレス前位置へ移動 ---
    Move P, @P P2

End Sub
```

RC8Command-Slave 活用ガイド

コンプライアンス有効(CompOn.pcs)

```
Sub Main

    ' --- コンプライアンスパラメータの設定 ---
    P11 = P(100, 100, 0, 10, 10, 10)      ' 座標系：力値の設定値
    P12 = P(10, 10, 100, 10, 10, 10)      ' 座標系：位置偏差許容の設定値
    J11 = J(50, 30, 30, 50, 30, 50)       ' 各軸：電流制限値の設定値
    J12 = J(50, 100, 100, 50, 100, 50)     ' 各軸：偏差許容の設定値

    ForceParam 1, 1, P11, PosEralw = P12, CurLmt = J11, Eralw = J12

    ' --- コンプライアンス機能を有効にする。 ---
    ForceCtrl On, 1

End Sub
```

コンプライアンス無効(CompOff.pcs)

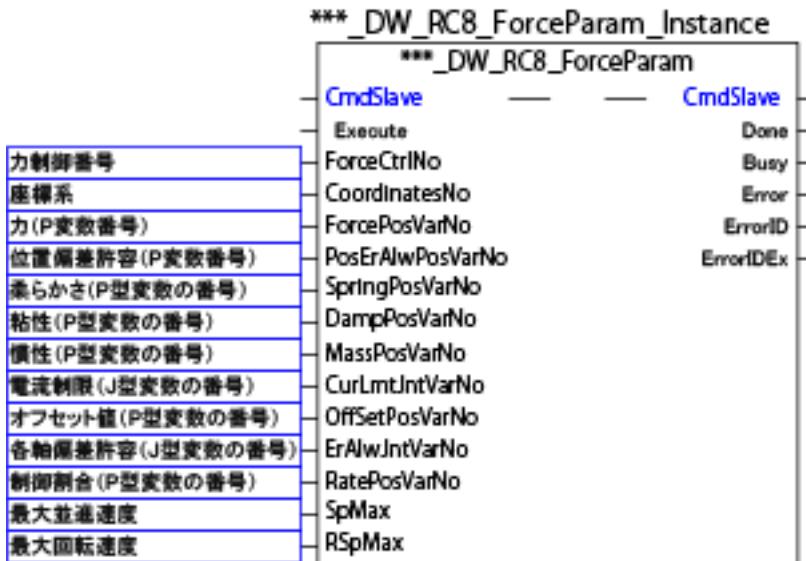
```
Sub Main

    ' --- コンプライアンス機能を無効にする。 ---
    ForceCtrl Off

End Sub
```

3.3.3. パラメータの設定

■Command-Slave の「ForceParm」パラメータ



項目名	変数指定	引数省略	単位	範囲	力センサ		適用	詳細
					有	無		
力制御番号	-	-	-	1 ~ 10	<input type="radio"/>	<input type="radio"/>	-	コンプライアンス機能のパラメータテーブル番号
座標系	-	-	-	0. 1. 2	<input type="radio"/>	<input type="radio"/>	-	0 : ベース座標系 1 : ツール座標系 2 : ワーク座標系
目標制御力(力)	P型変数の番号	<input type="radio"/>	N、Nm	0~	<input type="radio"/>	<input type="radio"/>	座標系	<p>ロボットを制御するための力。設定した力以上は出力しません。この値を小さくすると、小さな外力に倣って動作するようになります。</p> <p>(注意) 通常コンプライアンス(センサ無し)の場合、姿勢や動きによって制御力が変化します。実際の作業に合った力に設定してください。</p>
位置偏差許容	P型変数の番号	<input type="radio"/>	mm、deg	0~	<input type="radio"/>	<input type="radio"/>	座標系	手先位置のサーボ偏差許容値を設定します。力センサ有コンプライアンス機能の場合は、力制御開始からの手先位置の移動量を監視しています。
柔らかさ	P型変数の番号	<input type="radio"/>	%	0~100	<input type="radio"/>	<input type="radio"/>	座標系	位置に応じて増加する戻り力の強さの割合を設定します。この値を小さく設定すると、弱い力でも大きく移動させることができます。
粘性	P型変数の番号	<input type="radio"/>	%	0~100	<input type="radio"/>	<input type="radio"/>	座標系	速度に応じて増加する抵抗力の割合を設定します。この値を小さく設定すると、弱い力でも大きく移動させることができます。

RC8Command-Slave 活用ガイド

							く設定すると、速く移動させることができます。	
慣性	P型変数の番号	○	%	0~10 0	○	-	座標系	加速度に応じて増加する抵抗力の割合を設定します。この値を小さく設定すると、速く移動させることができます。
電流制限	J型変数の番号	○	%	0~10 0	-	○	各軸	各軸モータのトルク値(電流値)を設定します。(範囲は0~100[%]) 100[%]が定格で、値を小さくすることでトルク値(電流値)を下げ柔らかさを実現します。
オフセット	P型変数の番号	○	N、Nm	0~	-	○	座標系	あらかじめ倣わせたい方向に誘導するための力を設定します。 注:動作によっては、必ずしも狙った方向に誘導されることは限りません。
各軸偏差許容	J型変数の番号	○	mm、deg	0~	○	○	各軸	サーボ偏差許容値を設定します。力センサ有コンプライアンス機能の場合は、力制御開始からの各軸の移動量を監視しています。
制御割合	P型変数の番号	○	%	0~10 0	○	-	座標系	力センサ有コンプライアンス機能でどれくらいの割合で制御するか決めます。
最大並進速度	-	○	mm/s	-	○	-	-	接触前までの最大速度を決定します。
最大回転速度	-	○	deg/s	-	○	-	-	接触前までの最大回転速度を決定します。

■ ローカル変数

名称	データ型	備考
Tool1_Data	ARRAY[0..9] OF REAL	TOOL1 座標用
Force_Data	ARRAY[0..9] OF REAL	コンプライアンス: 力パラメータ
PosErAlw_Data	ARRAY[0..9] OF REAL	コンプライアンス: 位置偏差許容パラメータ
CurLmt_Data	ARRAY[0..9] OF REAL	コンプライアンス: 各軸電流制限パラメータ
ErAlw_Data	ARRAY[0..9] OF REAL	コンプライアンス: 各軸偏差許容パラメータ
PayLoadCenter_Data	ARRAY[0..2] OF REAL	内部負荷重心

■ ラダー図

// 各種パラメータ データ

初期設定

```

// TOOL1 座標データ
Tool1_Data[0] := REAL#0.0;           //TOOL1 X要素用
Tool1_Data[1] := REAL#0.0;           //TOOL1 Y要素用
Tool1_Data[2] := REAL#90.0;          //TOOL1 Z要素用
Tool1_Data[3] := REAL#0.0;           //TOOL1 RX要素用
Tool1_Data[4] := REAL#90.0;          //TOOL1 RY要素用
Tool1_Data[5] := REAL#0.0;           //TOOL1 RZ要素用
//配列[6]～[9]は使用しません

//コンプライアンス:力パラメータ
Force_Data[0] := REAL#100.0;         //力パラメータ X要素用
Force_Data[1] := REAL#100.0;         //力パラメータ Y要素用
Force_Data[2] := REAL#0.0;           //力パラメータ Z要素用
Force_Data[3] := REAL#10.0;          //力パラメータ RX要素用
Force_Data[4] := REAL#10.0;          //力パラメータ RY要素用
Force_Data[5] := REAL#10.0;          //力パラメータ RZ要素用
//配列[6]～[9]は使用しません

//コンプライアンス:位置偏差許容パラメータ
PosErAlw_Data[0] := REAL#10.0;       //位置偏差許容 X要素用
PosErAlw_Data[1] := REAL#10.0;       //位置偏差許容 Y要素用
PosErAlw_Data[2] := REAL#100.0;      //位置偏差許容 Z要素用
PosErAlw_Data[3] := REAL#10.0;       //位置偏差許容 RX要素用
PosErAlw_Data[4] := REAL#10.0;       //位置偏差許容 RY要素用
PosErAlw_Data[5] := REAL#10.0;       //位置偏差許容 RZ要素用
//配列[6]～[9]は使用しません

//コンプライアンス:各軸電流制限パラメータ
CurLmt_Data[0]:= REAL#50.0;          //各軸電流制限 1軸用
CurLmt_Data[1]:= REAL#30.0;          //各軸電流制限 2軸用
CurLmt_Data[2]:= REAL#30.0;          //各軸電流制限 3軸用
CurLmt_Data[3]:= REAL#50.0;          //各軸電流制限 4軸用
CurLmt_Data[4]:= REAL#30.0;          //各軸電流制限 5軸用
CurLmt_Data[5]:= REAL#50.0;          //各軸電流制限 6軸用
//配列[6]～[9]は使用しません

//コンプライアンス:各軸偏差許容パラメータ
ErAlw_Data[0]:= REAL#50.0;           //各軸偏差許容 1軸用
ErAlw_Data[1]:= REAL#100.0;          //各軸偏差許容 2軸用
ErAlw_Data[2]:= REAL#100.0;          //各軸偏差許容 3軸用
ErAlw_Data[3]:= REAL#50.0;           //各軸偏差許容 4軸用
ErAlw_Data[4]:= REAL#100.0;          //各軸偏差許容 5軸用
ErAlw_Data[5]:= REAL#50.0;           //各軸偏差許容 6軸用
//配列[6]～[9]は使用しません

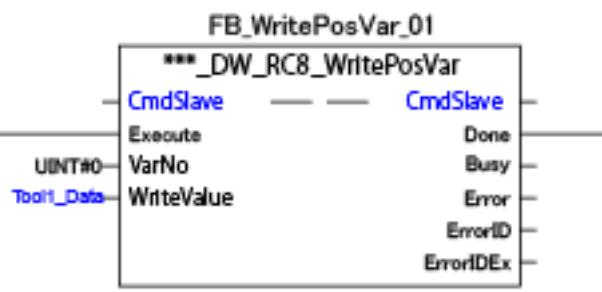
// 内部重心
PayLoadCenter_Data[0] := REAL#0.0;   //内部重心 X要素用
PayLoadCenter_Data[1] := REAL#0.0;   //内部重心 Y要素用
PayLoadCenter_Data[2] := REAL#45.0;  //内部重心 Z要素用

```

RC8Command-Slave 活用ガイド

// パラメータデータの変数書き込み

初期設定



FB「WritePosVar」の引数

- ・P型変数番号 = 0 ToolNo = UINT#0
- ・書込み値 = Tool1_Data WriteValue = Tool1_Data

P0にTool1_Dataを書き込み



FB「WritePosVar」の引数

- ・P型変数番号 = 11 ToolNo = UINT#11
- ・書込み値 = Force_Data WriteValue = Force_Data

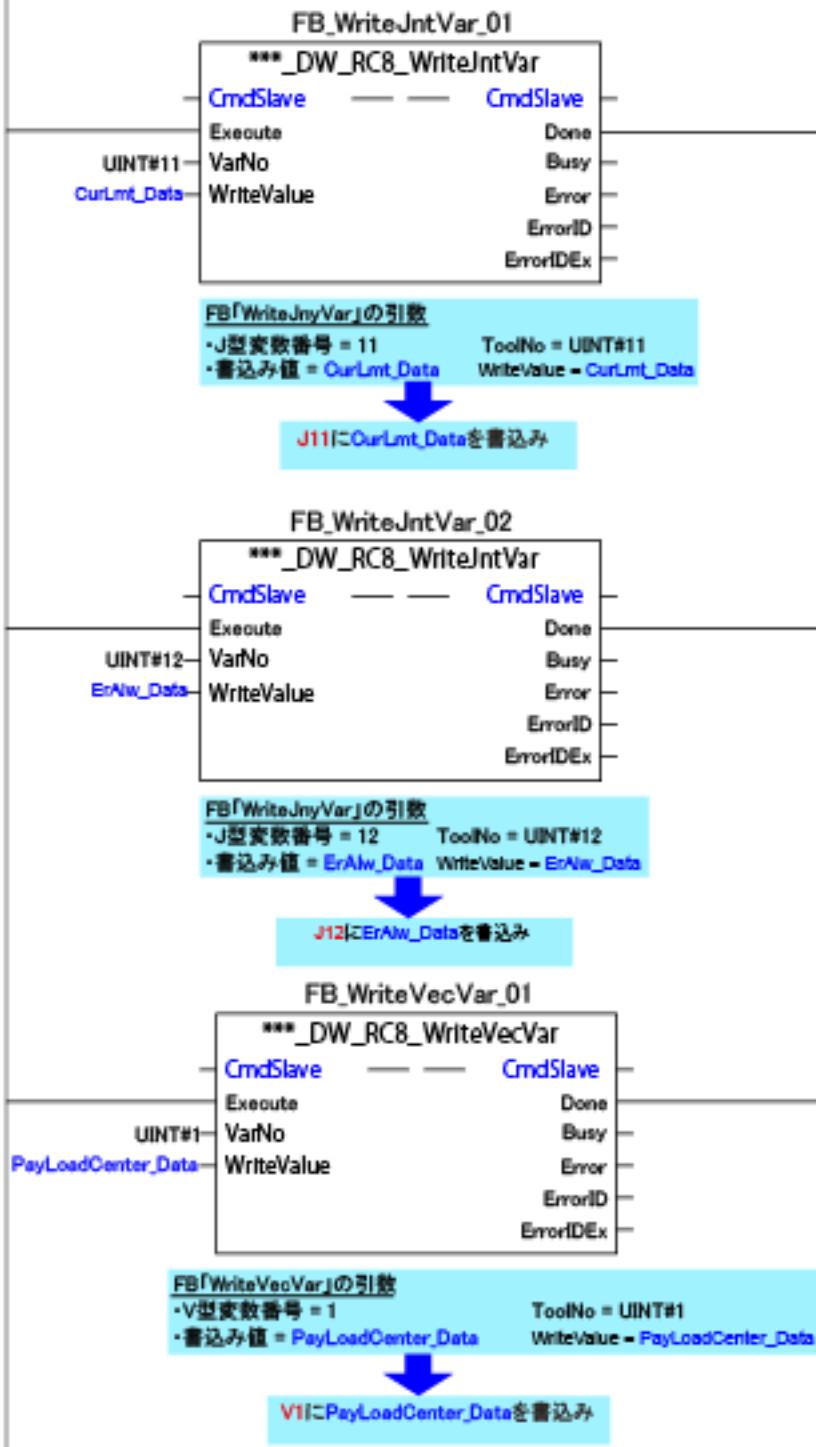
P11にForce_Dataを書き込み



FB「WritePosVar」の引数

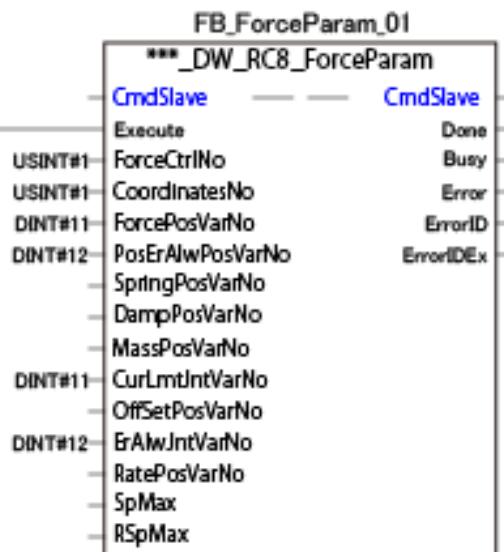
- ・P型変数番号 = 12 ToolNo = UINT#12
- ・書込み値 = PosErAlw_Data WriteValue = PosErAlw_Data

P12にPosErAlw_Dataを書き込み



// コンプライアンスパラメータ設定

初期設定



FB「ForceParam」の引数

- ・力制御番号 = 1 ForceCtrlNo = USINT1#1
- ・座標系 = ツール座標 CoordinatesNo = USINT#1
(0=ベース座標, 1=ツール座標, 2=ワーク座標)
- ・力 = P11 ForcePosVarNo = DINT#11 (P型変数の番号指定)
- ・位置偏差許容 = P12 PosErAlwPosVarNo = DINT#12 (P型変数の番号指定)
- ・各軸電流制限 = J11 CurLmtIntVarNo = DINT#11 (J型変数の番号指定)
- ・各軸偏差許容 = J12 ErAlwIntVarNo = DINT#12 (J型変数の番号指定)

その他の引数は省略



力制御番号(コンプライアンステーブル) 1

【座標系】

- ・力 X=100[N]、Y=100[N]、Z=0[N]、RX=10[Nm]、RY=10[Nm]、RZ=10[Nm]
- ・標準許容 X=10[mm]、Y=10[mm]、Z=100[mm]、RX=10[deg]、RY=10[deg]、RZ=10[deg]

【各軸】

- ・電流制限 J1=50[%]、J2=30[%]、J3=30[%]、J4=50[%]、J5=30[%]、J6=50[%]
- ・偏差許容 J1=50[deg]、J2=100[deg]、J3=100[deg]、J5=100[deg]、J4=50[deg]、J6=50[deg]

// 内部負荷設定

初期設定



FB「PayLoad」の引数

- ・先端負荷質量 = 1000 PayLoad = USINT#1000
- ・負荷重心位置 = V1 PayLoadCenterVecVarNo = UINT#1 (V型変数の番号指定)

その他の引数は省略



内部負荷

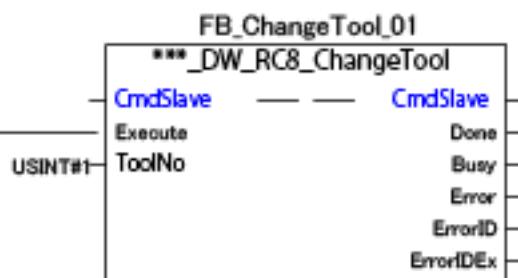
- ・負荷質量 1000[g]
- ・負荷重心 X=0[mm]、Y=0[mm]、Z=45[mm]

// ツール設定

初期設定



FB「Tool」の引数
 ・ツール番号 = 1 ToolNo = USINT#1
 ・設定ツール座標 = P0 PosVarNo = UINT#0 (P型変数の番号指定)

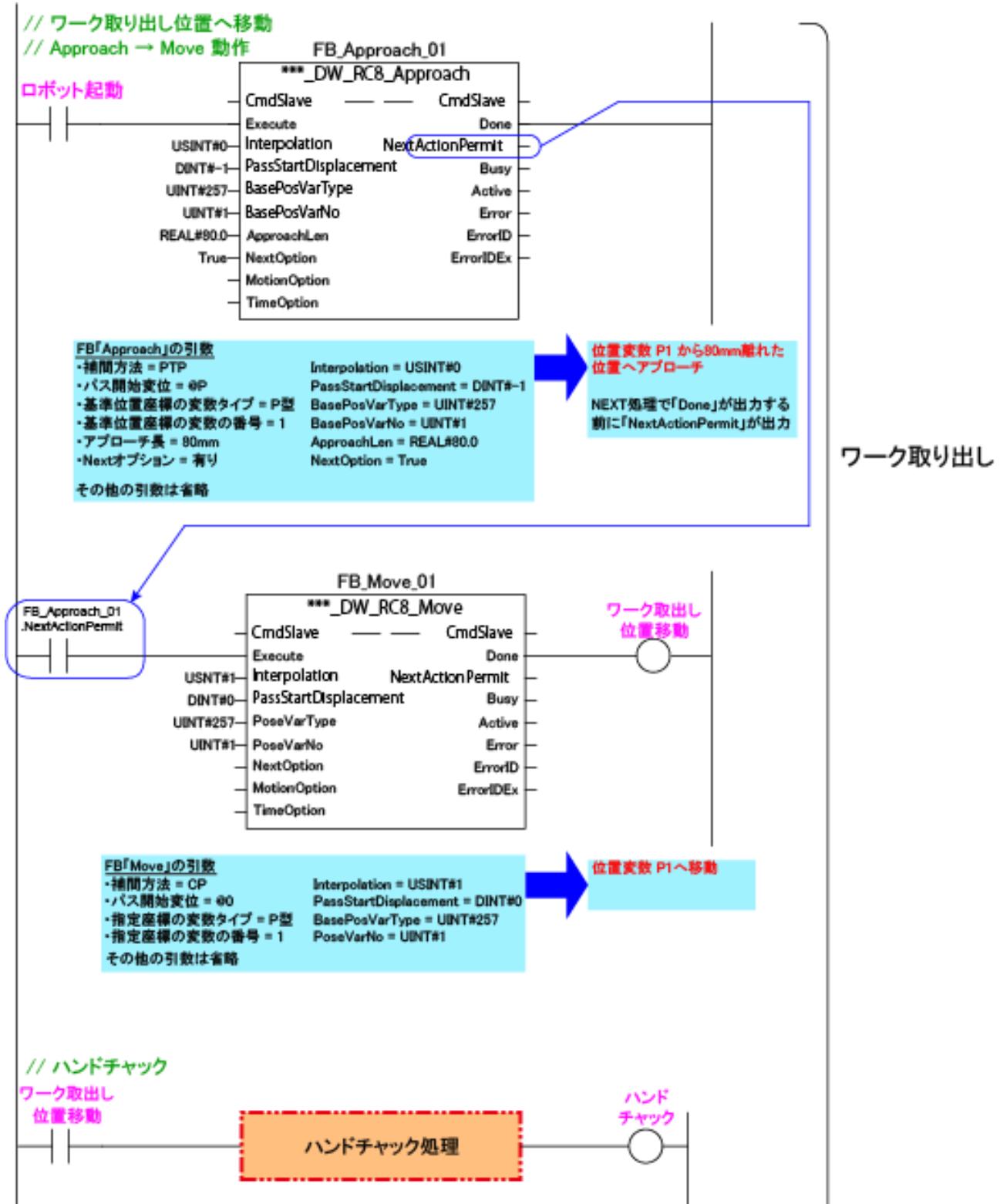


FB「ChangeTool」の引数
 ・ツール番号 = 1 ToolNo = USINT#1

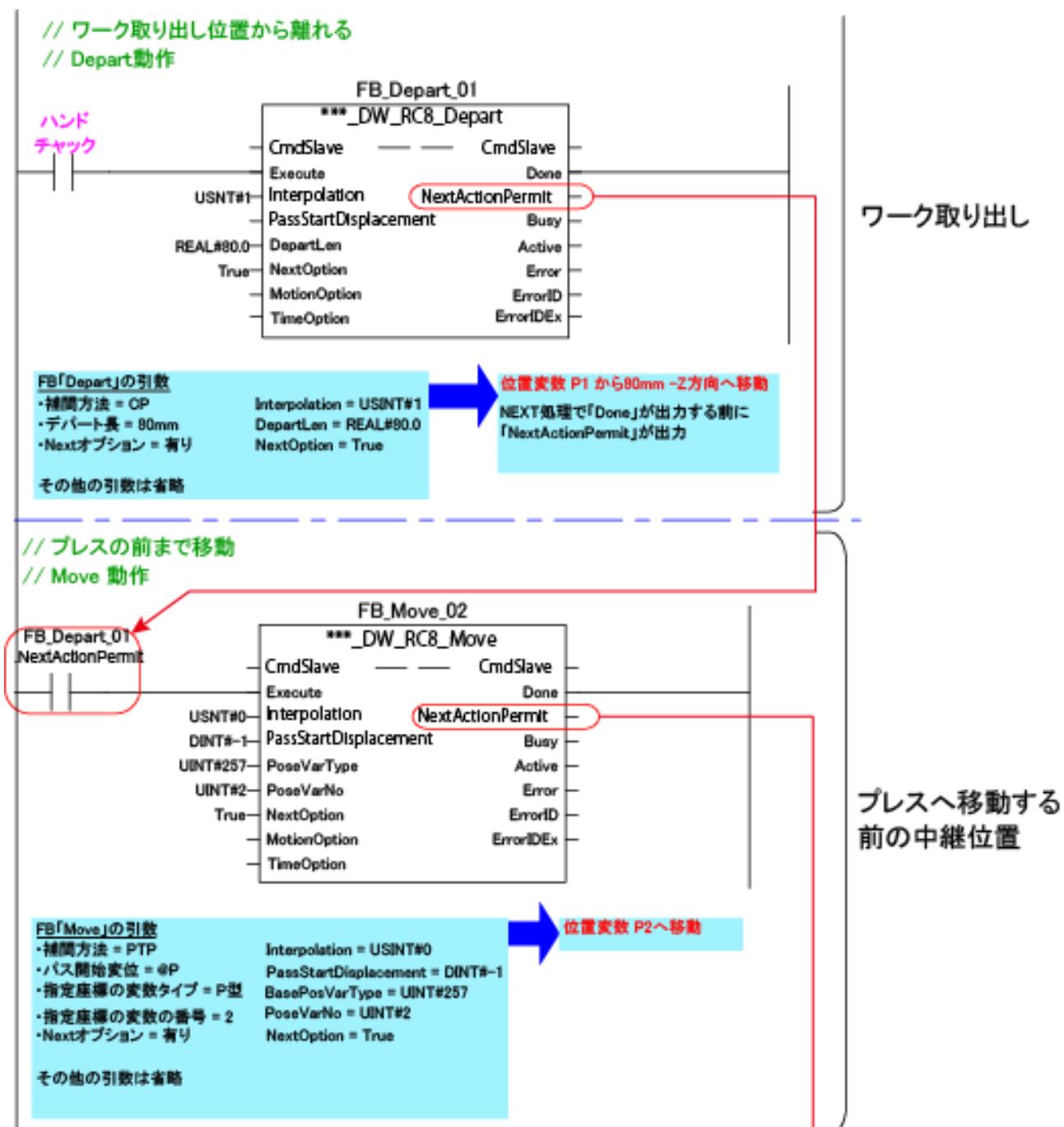
ツール番号「1」に変更

3.3.4. Command-Slave で実施例を実行する場合

■ ラダー図

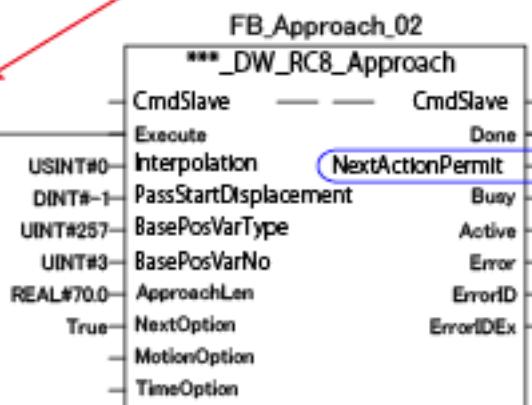


RC8Command-Slave 活用ガイド



// イン位置へ移動

// Approach → Move 動作

**FB「Approach」の引数**

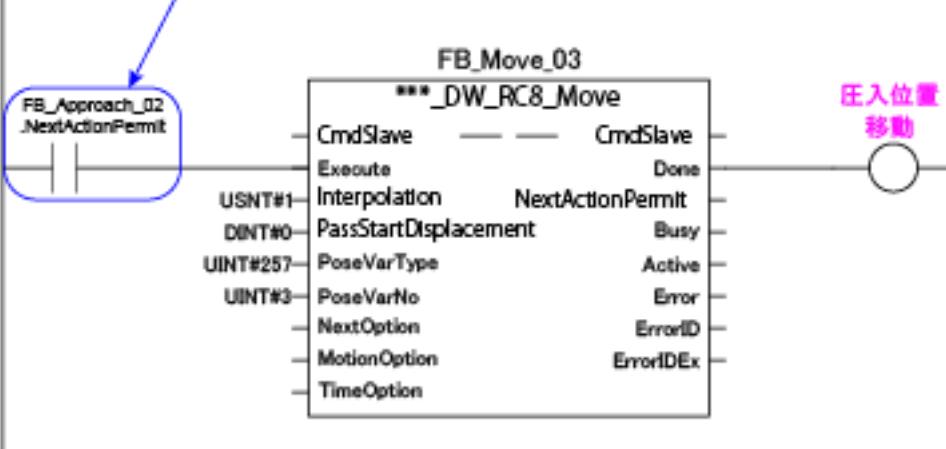
- ・補間方法 = PTP
 - ・パス開始変位 = @P
 - ・基準位置座標の変数タイプ = P型
 - ・基準位置座標の変数の番号 = 3
 - ・アプローチ長 = 70mm
 - ・Nextオプション = 有り
- その他の引数は省略

Interpolation = USINT#0
 PassStartDisplacement = DINT#-1
 BasePosVarType = UINT#257
 BasePosVarNo = UINT#3
 ApproachLen = REAL#70.0
 NextOption = True

位置変数 P3 から 70mm -Z 方向へ
移動

NEXT処理で「Done」が出力する
前に「NextActionPermit」が出力

圧入

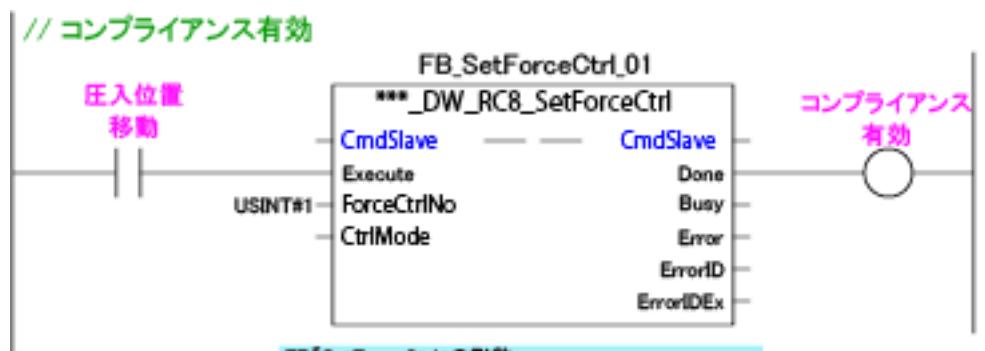
**FB「Move」の引数**

- ・補間方法 = OP
 - ・パス開始変位 = @0
 - ・指定座標の変数タイプ = P型
 - ・指定座標の変数の番号 = 3
- その他の引数は省略

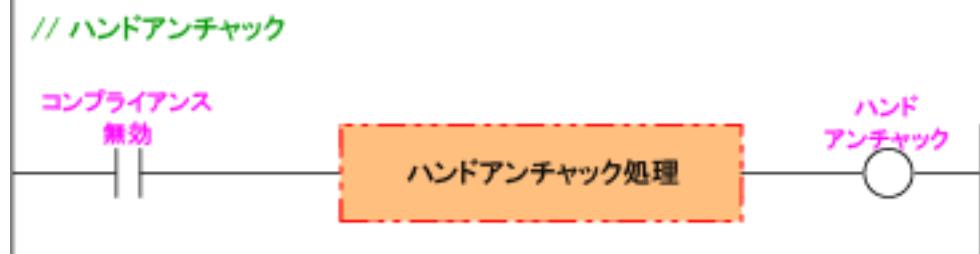
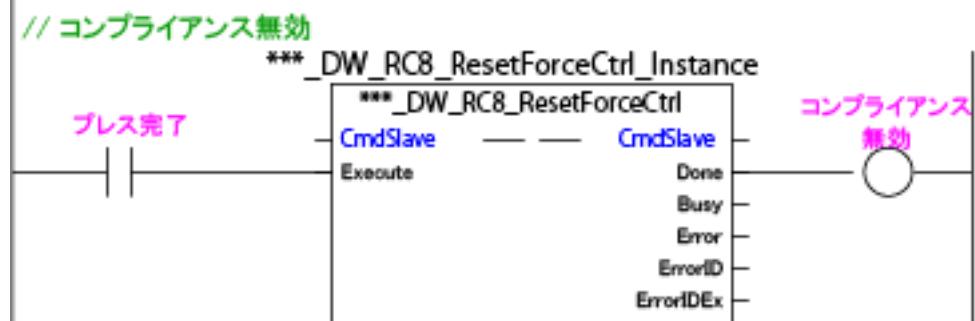
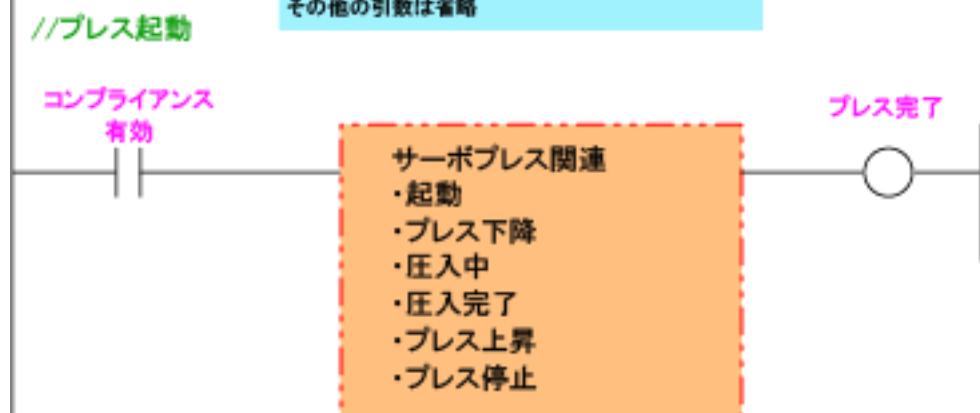
Interpolation = USINT#1
 PassStartDisplacement = DINT#0
 BasePosVarType = UINT#257
 PoseVarNo = UINT#3

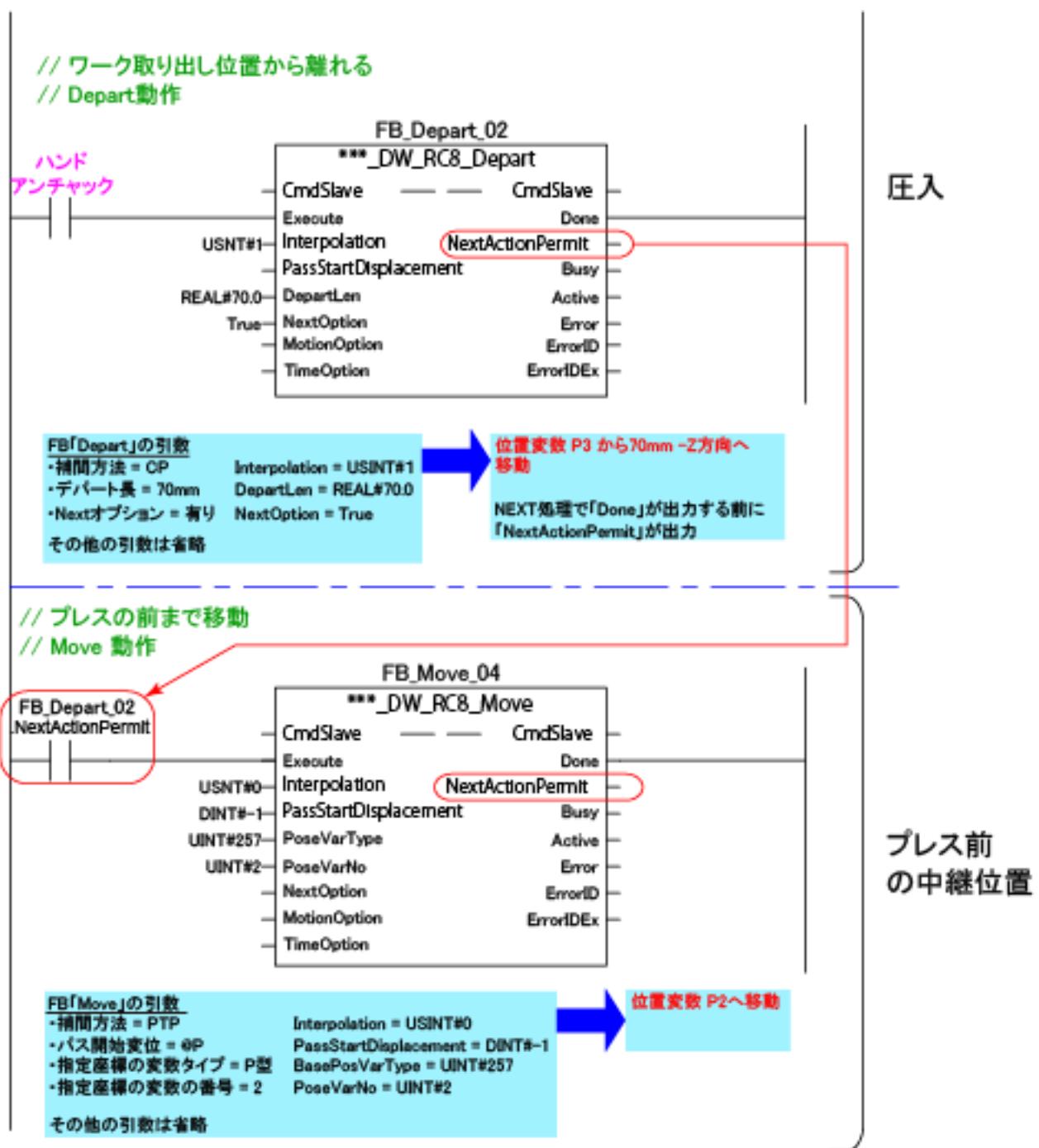
位置変数 P3 へ移動

RC8Command-Slave 活用ガイド



圧入





4. 付録

4.1. 「ロボット Enable 中」=OFF のときに起動できる非動作系の FB

下表で○印の付いたものは、「ロボット Enable 中」=OFF のときでも起動できます。

コマンド	機能	起動可の FB
CIP_DW_RC8_Arrive	次ステップ動作待機	
CIP_DW_RC8_Speed	内部速度指定	
CIP_DW_RC8_Accel	内部加速度指定	
CIP_DW_RC8_Decel	内部減速度指定	
CIP_DW_RC8_CurSpd	内部速度取得 (現在値)	○
CIP_DW_RC8_CurAcc	内部加速度取得 (現在値)	○
CIP_DW_RC8_CurDec	内部減速度取得 (現在値)	○
CIPDW_RC8_ExtSpeed	外部速度指定	○
CIP_DW_RC8_CurExtSpd	外部速度取得 (現在値)	○
CIP_DW_RC8_CurExtAcc	外部加速度取得 (現在値)	○
CIP_DW_RC8_CurExtDec	外部加減速度取得 (現在値)	○
CIP_DW_RC8_Tool	ツール設定	
CIP_DW_RC8_ChangeTool	ツール番号変更	
CIP_DW_RC8_CurTool	ツール番号取得 (現在値)	○
CIP_DW_RC8_Work	ワーク設定	
CIP_DW_RC8_ChangeWork	ワーク番号変更	
CIP_DW_RC8_CurWorl	ワーク番号取得 (現在値)	○
CIP_DW_RC8_Area	エリア設定	○
CIP_DW_RC8_SetArea	エリア有効	○
CIP_DW_RC8_ResetArea	エリア無効	○
CIP_DW_RC8_AreaSize	エリア各辺の長さ取得	○
CIP_DW_RC8_AreaPos	エリア中心位置取得	○
CIP_DW_RC8_TakeArmState	ロボット制御権の取得状態	○
CIP_DW_RC8_CurPos	位置取得 (現在値)	○
CIP_DW_RC8_CurJnt	角度取得 (現在値)	○
CIP_DW_RC8_CurTrn	位置 (同次変換型) 取得 (現在値)	○
CIP_DW_RC8_SysState	RC8 システムステータス取得	○
CIP_DW_RC8_WriteIntVar	I 型変数書き込み	○
CIP_DW_RC8_ReadIntVar	I 型変数読み込み	○
CIP_DW_RC8_WriteFltVar	F 型変数書き込み	○
CIPDW_RC8_ReadFltVar	F 型変数読み込み	○
CIP_DW_RC8_WriteVecVar	V 型変数書き込み	○
CIP_DW_RC8_ReadVecVar	V 型変数読み込み	○
CIP_DW_RC8_WritePosVar	P 型変数書き込み	○
CIP_DW_RC8_ReadPosVar	P 型変数読み込み	○
CIP_DW_RC8_WriteJntVar	J 型変数書き込み	○
CIP_DW_RC8_ReadJntVar	J 型変数読み込み	○
CIP_DW_RC8_WriteTrnVar	T 型変数書き込み	○
CIP_DW_RC8_ReadTrnVar	T 型変数読み込み	○
CIP_DW_RC8_PalletCalsPos	パレット位置取得	○
CIP_DW_RC8_AddPathPoint	スプライン 経路点追加	
CIP_DW_RC8_ClrPathPoint	スプライン 経路点クリア	

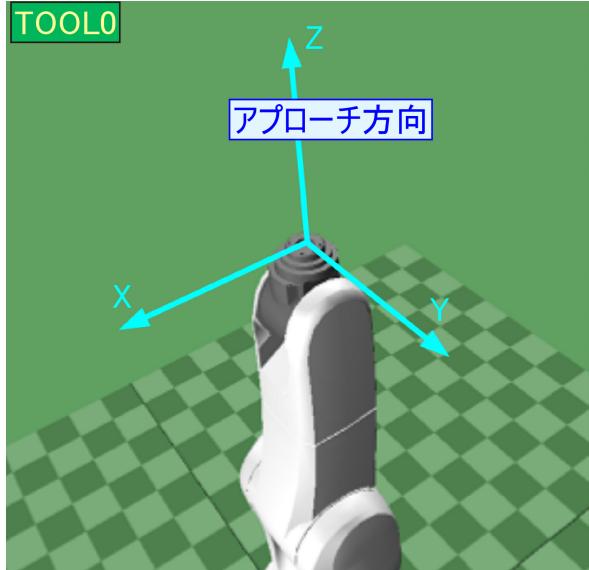
RC8Command-Slave 活用ガイド

CIP_DW_RC8_GetPathPoint	スプライン経路点の位置を取得	
CIP_DW_RC8_GetPathPointCount	スプライン経路点個数の取得	
CIP_DW_RC8_LoadPathPoint	スプライン経路データ読込	
CIP_DW_RC8_PayLoad	内部負荷条件設定	
CIP_DW_RC8_SpeedMode	最適速度機能の設定変更	
CIP_DW_RC8_HighPathAccuracy	高軌跡制御の有効	
CIP_DW_RC8_HighPathAccuracyReset	高軌跡制御の無効	
CIP_DW_RC8_ForceParam	力制御機能のパラメータを設定	
CIP_DW_RC8_SetForceCtrl	力制御機能の有効	
CIP_DW_RC8_RetForceCtrl	力制御機能の無効	
CIP_DW_RC8_SetCurLmt	電流制限機能の設定と有効	
CIP_DW_RC8_ResetCurLmt	電流制限機能の無効	
CIP_DW_RC8_SetZforce	Hシリーズロボットの第3軸 (Z軸)の電流制限機能を推力で指定し有効	
CIP_DW_RC8_ResetZforce	Hシリーズロボットの第3軸 (Z軸)の電流制限機能の推力を無効	
CIP_DW_RC8_SetErAlw	偏差許容値の設定と有効	
CIP_DW_RC8_ResetErAlw	偏差許容値の無効	
CIP_DW_RC8_SetGrvCtrl	重力補償制御機能の有効	
CIP_DW_RC8_ResetGrvCtrl	重力補償制御機能の無効	
CIP_DW_RC8_SetGrvOffset	重力オフセット設定機能の有効	
CIP_DW_RC8_RESETGrvOffset	重力オフセット設定機能の無効	
CIP_DW_RC8_SrvUnLock	指定した軸をサーボロック解除	
CIP_DW_RC8_SrvLock	指定した軸をサーボロック	
CIP_DW_RC8_ClrErr	エラー解除	○
CIP_DW_RC8_RobotEnable	IO スレーブロボット制御権の有効	○
CIP_DW_RC8_Motor	モータ ON/OFF	
CIP_DW_RC8_AreaDetectionOut	エリア検知時出力	○
CIP_DW_RC8_EIPIF	RC8Command-Slave 変数変換用インターフェース (EIP)	○

4.2. ツールの説明

ツールは1~63を設定することができます。ツール0はツール取付面を基準とするメカニカルインターフェース座標を表わし、座標値の編集はできません。ツール座標のZ方向がアプローチ方向になります。

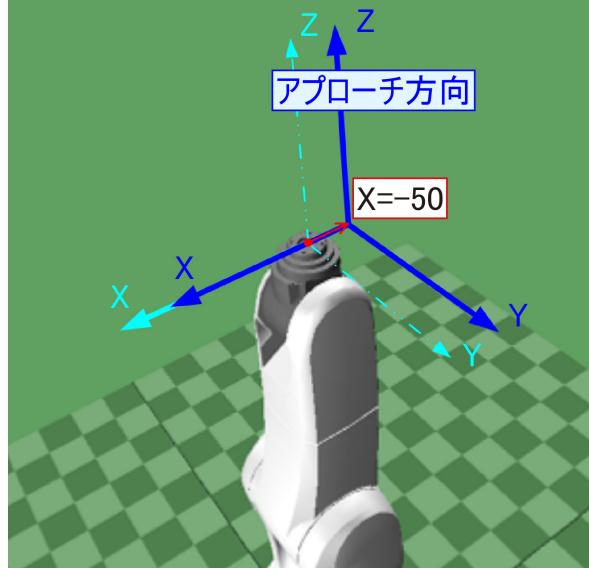
ツール番号	X	Y	Z	RX	RY	RZ
0	0	0	0	0	0	0



「2.8.ツール設定」の例では、ツール1~7の座標を以下の値で設定しています。

(1)ツール1

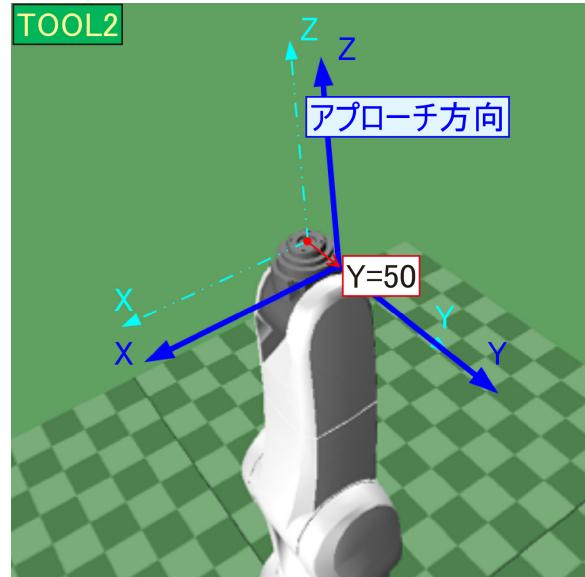
ツール番号	X	Y	Z	RX	RY	RZ
1	-50	0	0	0	0	0



RC8Command-Slave 活用ガイド

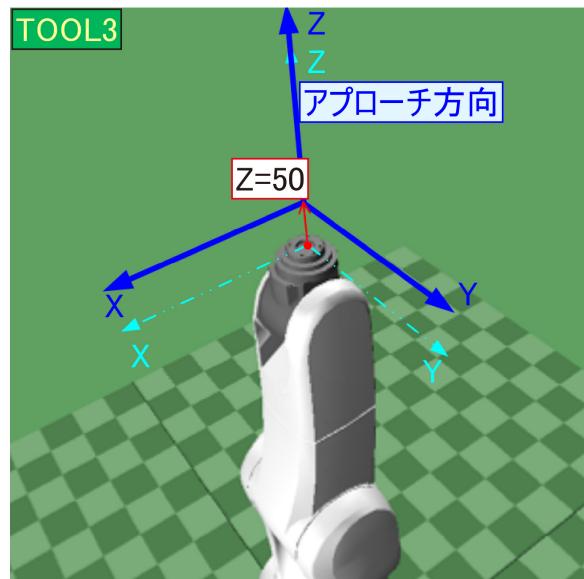
(2)ツール 2

ツール番号	X	Y	Z	RX	RY	RZ
2	0	50	0	0	0	0



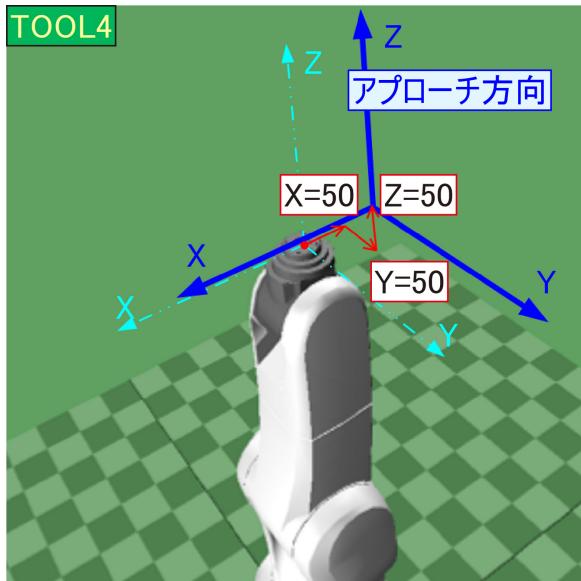
(3)ツール 3

ツール番号	X	Y	Z	RX	RY	RZ
3	0	0	50	0	0	0



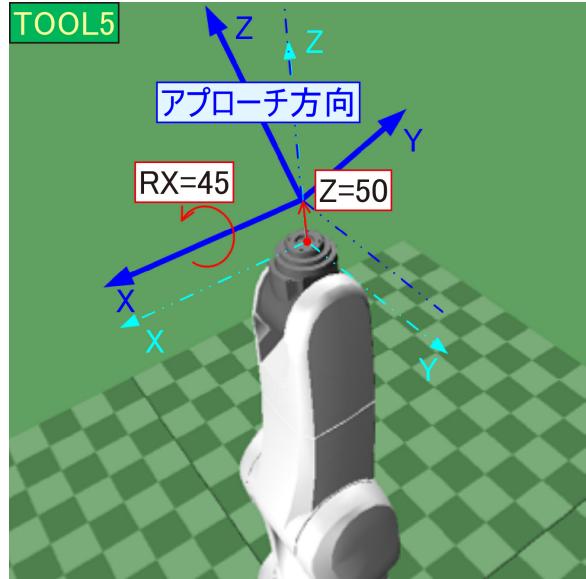
(4)ツール4

ツール番号	X	Y	Z	RX	RY	RZ
4	-50	50	50	0	0	0



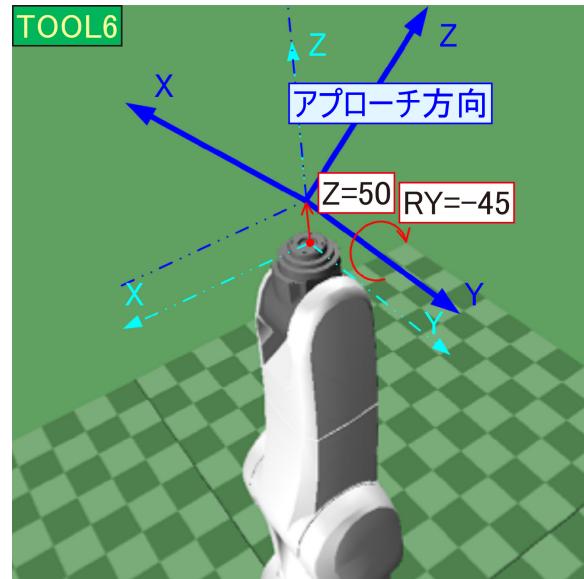
(5)ツール5

ツール番号	X	Y	Z	RX	RY	RZ
5	0	0	50	45	0	0



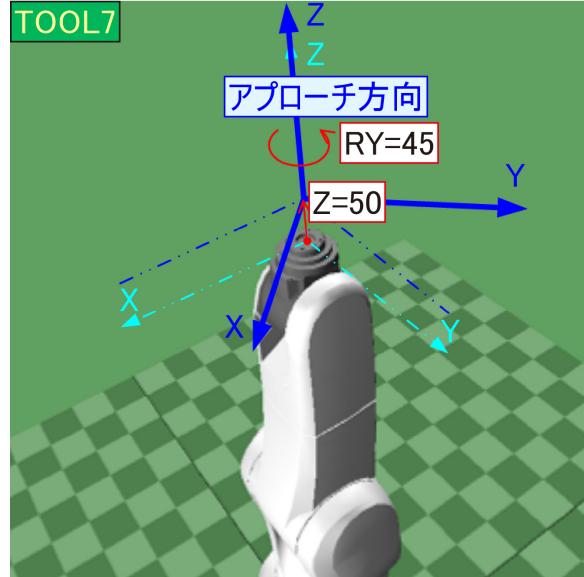
(6)ツール 6

ツール番号	X	Y	Z	RX	RY	RZ
6	0	0	50	0	-45	0



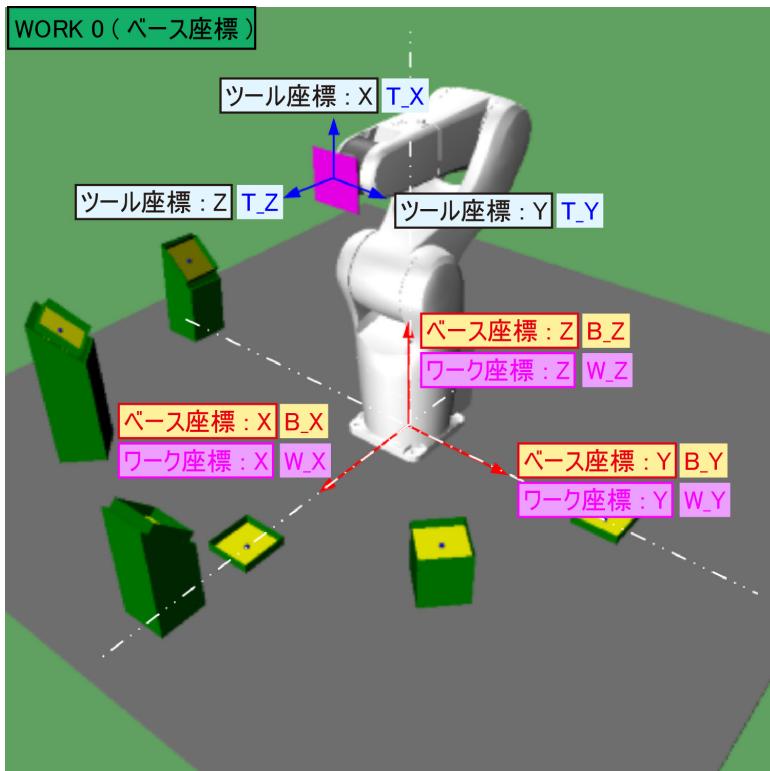
(7)ツール 7

ツール番号	X	Y	Z	RX	RY	RZ
7	0	0	50	0	0	45



4.3. ワークの説明

ワーク座標は1~7の設定ができます。ワーク0はベース座標を表わし、座標値の編集はできません。



ワーク番号	X	Y	Z	RX	RY	RZ
0	0	0	0	0	0	0

上図では、ワーク座標の X、Y、Z の座標軸を W_X、W_Y、W_Z、ベース座標の X、Y、Z の座標軸を B_X、B_Y、B_Z、ツール座標の X、Y、Z の座標軸を T_X、T_Y、T_Z と表しています。

ここではワーク 1~6 の時の位置変数 P10、P11 を以下のように設定した場合を例として説明します。

ワーク 1~6 の設定値

ワーク番号	X	Y	Z	RX	RY	RZ
1	500	0	0	0	0	0
2	350	350	100	0	0	45
3	0	500	0	0	0	90
4	700	0	200	0	-45	0
5	500	-500	300	0	-45	-45
6	0	-700	200	0	-45	-90

位置変数 P10、P11 の値

位置変数	X	Y	Z	RX	RY	RZ	FIG
P10	0	0	300	180	0	180	5
P11	0	0	0	180	0	180	5

RC8Command-Slave 活用ガイド

(1)ワーク 1

- ワーク 1 の設定

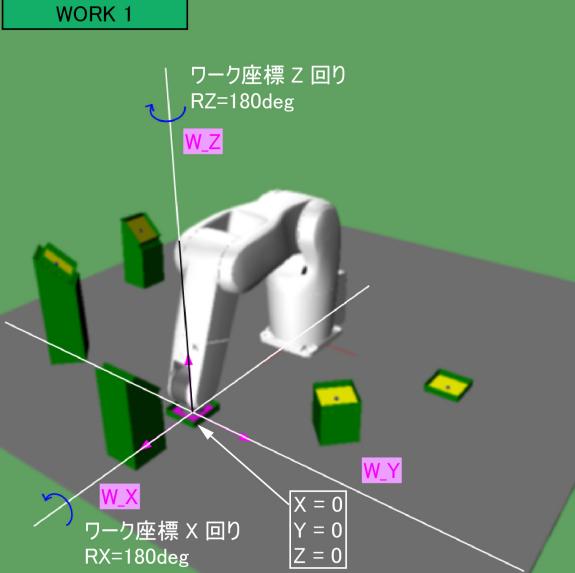
ワーク番号	X	Y	Z	RX	RY	RZ
1	500	0	0	0	0	0

- 位置変数 P10

姿勢	位置																
<p>WORK 1</p> <p>ワーク座標 Z 回り RZ=180deg T_Y ← T_Z ↑ T_X</p> <p>ワーク座標 X 回り RX=180deg T_X ← T_Y ↑ T_Z</p> <p>Z = 300 mm</p>	<table border="1"> <thead> <tr> <th>ワーク 1 での値</th> <th>ベース座標での値</th> </tr> </thead> <tbody> <tr> <td>X=0</td> <td>X=500</td> </tr> <tr> <td>Y=0</td> <td>Y=0</td> </tr> <tr> <td>Z=300</td> <td>Z=300</td> </tr> <tr> <td>RX=180</td> <td>RX=180</td> </tr> <tr> <td>RY=0</td> <td>RY=0</td> </tr> <tr> <td>RZ=180</td> <td>RZ=180</td> </tr> <tr> <td>FIG=5</td> <td>FIG=5</td> </tr> </tbody> </table>	ワーク 1 での値	ベース座標での値	X=0	X=500	Y=0	Y=0	Z=300	Z=300	RX=180	RX=180	RY=0	RY=0	RZ=180	RZ=180	FIG=5	FIG=5
ワーク 1 での値	ベース座標での値																
X=0	X=500																
Y=0	Y=0																
Z=300	Z=300																
RX=180	RX=180																
RY=0	RY=0																
RZ=180	RZ=180																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

● 位置変数 P11

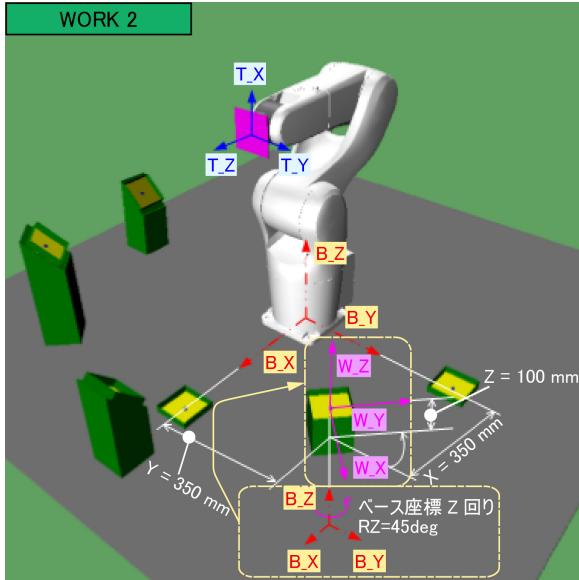
姿勢	位置																
 <p>WORK 1</p> <p>ワーク座標 Z 回り RZ=180deg</p> <p>W_Z</p> <p>ワーク座標 X 回り RX=180deg</p> <p>W_X</p> <p>W_Y</p> <p>X = 0 Y = 0 Z = 0</p>	<table border="1"><thead><tr><th>ワーク 1 での値</th><th>ベース座標での値</th></tr></thead><tbody><tr><td>X=0</td><td>X=500</td></tr><tr><td>Y=0</td><td>Y=0</td></tr><tr><td>Z=0</td><td>Z=0</td></tr><tr><td>RX=180</td><td>RX=180</td></tr><tr><td>RY=0</td><td>RY=0</td></tr><tr><td>RZ=180</td><td>RZ=180</td></tr><tr><td>FIG=5</td><td>FIG=5</td></tr></tbody></table>	ワーク 1 での値	ベース座標での値	X=0	X=500	Y=0	Y=0	Z=0	Z=0	RX=180	RX=180	RY=0	RY=0	RZ=180	RZ=180	FIG=5	FIG=5
ワーク 1 での値	ベース座標での値																
X=0	X=500																
Y=0	Y=0																
Z=0	Z=0																
RX=180	RX=180																
RY=0	RY=0																
RZ=180	RZ=180																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

(2.) ワーク 2

- ワーク 2 の設定

ワーク番号	X	Y	Z	RX	RY	RZ
2	350	350	100	0	0	45

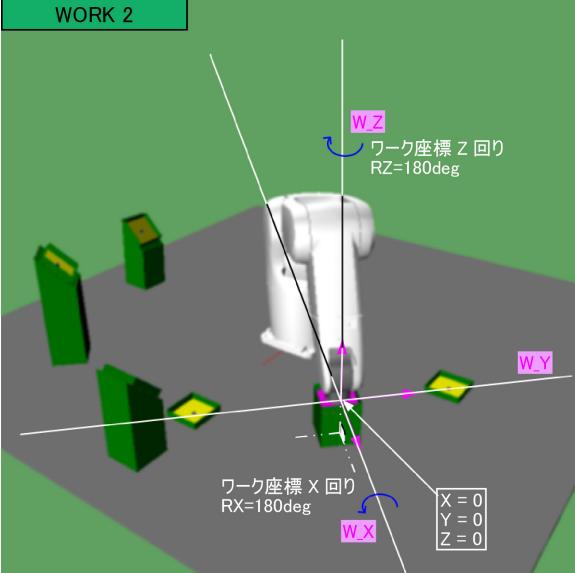


- 位置変数 P10

姿勢	位置																
<p>WORK 2</p>	<table border="1"> <thead> <tr> <th>ワーク 2 での値</th> <th>ベース座標での値</th> </tr> </thead> <tbody> <tr> <td>X=0</td> <td>X=350</td> </tr> <tr> <td>Y=0</td> <td>Y=350</td> </tr> <tr> <td>Z=300</td> <td>Z=400</td> </tr> <tr> <td>RX=180</td> <td>RX=180</td> </tr> <tr> <td>RY=0</td> <td>RY=0</td> </tr> <tr> <td>RZ=180</td> <td>RZ=-135</td> </tr> <tr> <td>FIG=5</td> <td>FIG=5</td> </tr> </tbody> </table>	ワーク 2 での値	ベース座標での値	X=0	X=350	Y=0	Y=350	Z=300	Z=400	RX=180	RX=180	RY=0	RY=0	RZ=180	RZ=-135	FIG=5	FIG=5
ワーク 2 での値	ベース座標での値																
X=0	X=350																
Y=0	Y=350																
Z=300	Z=400																
RX=180	RX=180																
RY=0	RY=0																
RZ=180	RZ=-135																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

● 位置変数 P11

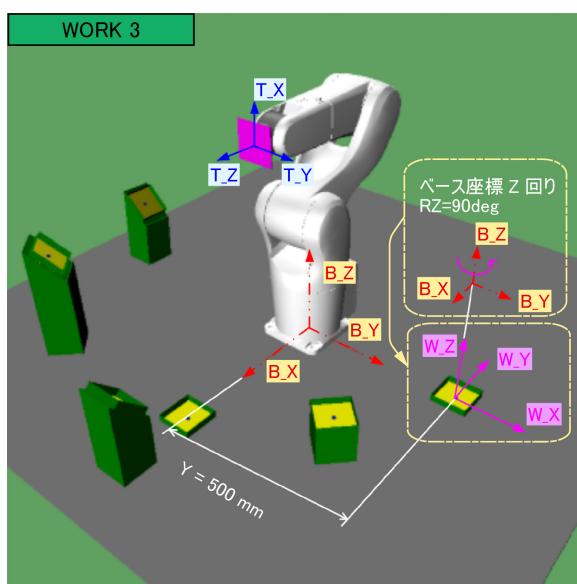
姿勢	位置																
 <p>WORK 2</p> <p>ワーク座標 z 回り RZ=180deg</p> <p>ワーク座標 X 回り RX=180deg</p> <p>ワーク座標 Y 回り RY=0</p> <p>[X = 0] [Y = 0] [Z = 0]</p>	<table border="1"><thead><tr><th>ワーク 2 での値</th><th>ベース座標での値</th></tr></thead><tbody><tr><td>X=0</td><td>X=350</td></tr><tr><td>Y=0</td><td>Y=350</td></tr><tr><td>Z=0</td><td>Z=100</td></tr><tr><td>RX=180</td><td>RX=180</td></tr><tr><td>RY=0</td><td>RY=0</td></tr><tr><td>RZ=180</td><td>RZ=-135</td></tr><tr><td>FIG=5</td><td>FIG=5</td></tr></tbody></table>	ワーク 2 での値	ベース座標での値	X=0	X=350	Y=0	Y=350	Z=0	Z=100	RX=180	RX=180	RY=0	RY=0	RZ=180	RZ=-135	FIG=5	FIG=5
ワーク 2 での値	ベース座標での値																
X=0	X=350																
Y=0	Y=350																
Z=0	Z=100																
RX=180	RX=180																
RY=0	RY=0																
RZ=180	RZ=-135																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

(3)ワーク 3

- ワーク 3 の設定

ワーク番号	X	Y	Z	RX	RY	RZ
3	0	500	0	0	0	90

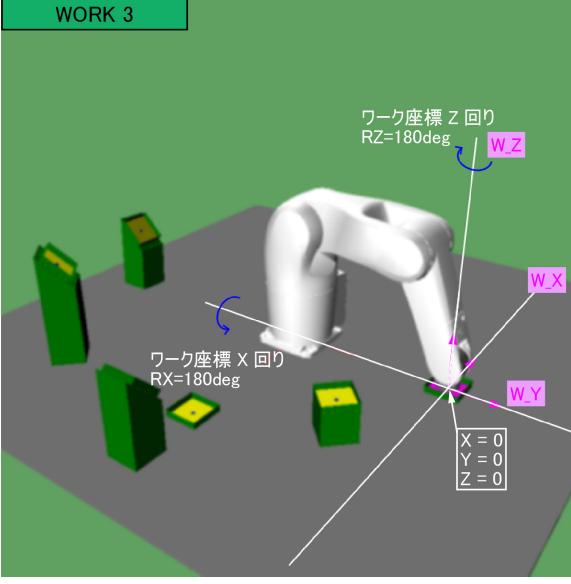


- 位置変数 P10

姿勢	位置																
<p>WORK 3</p>	<table border="1"> <thead> <tr> <th>ワーク 3 での値</th> <th>ベース座標での値</th> </tr> </thead> <tbody> <tr> <td>X=0</td> <td>X=0</td> </tr> <tr> <td>Y=0</td> <td>Y=500</td> </tr> <tr> <td>Z=300</td> <td>Z=300</td> </tr> <tr> <td>RX=180</td> <td>RX=180</td> </tr> <tr> <td>RY=0</td> <td>RY=0</td> </tr> <tr> <td>RZ=180</td> <td>RZ=-90</td> </tr> <tr> <td>FIG=5</td> <td>FIG=5</td> </tr> </tbody> </table>	ワーク 3 での値	ベース座標での値	X=0	X=0	Y=0	Y=500	Z=300	Z=300	RX=180	RX=180	RY=0	RY=0	RZ=180	RZ=-90	FIG=5	FIG=5
ワーク 3 での値	ベース座標での値																
X=0	X=0																
Y=0	Y=500																
Z=300	Z=300																
RX=180	RX=180																
RY=0	RY=0																
RZ=180	RZ=-90																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

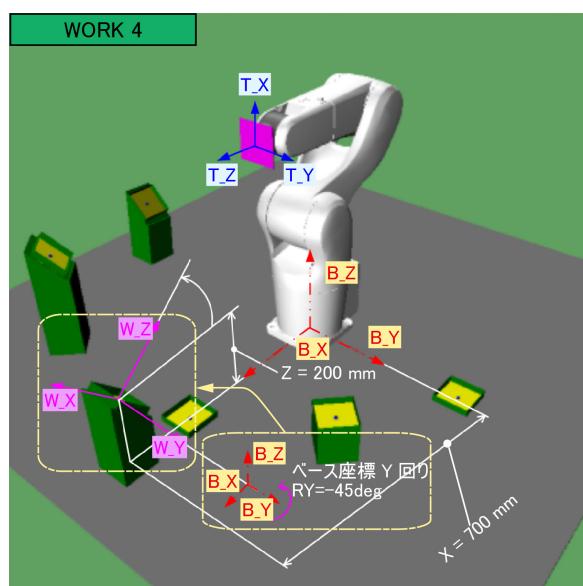
● 位置変数 P11

姿勢	位置																
 <p>WORK 3</p>	<table border="1"><thead><tr><th>ワーク 3 での値</th><th>ベース座標での値</th></tr></thead><tbody><tr><td>X=0</td><td>X=0</td></tr><tr><td>Y=0</td><td>Y=500</td></tr><tr><td>Z=0</td><td>Z=0</td></tr><tr><td>RX=180</td><td>RX=180</td></tr><tr><td>RY=0</td><td>RY=0</td></tr><tr><td>RZ=180</td><td>RZ=-90</td></tr><tr><td>FIG=5</td><td>FIG=5</td></tr></tbody></table>	ワーク 3 での値	ベース座標での値	X=0	X=0	Y=0	Y=500	Z=0	Z=0	RX=180	RX=180	RY=0	RY=0	RZ=180	RZ=-90	FIG=5	FIG=5
ワーク 3 での値	ベース座標での値																
X=0	X=0																
Y=0	Y=500																
Z=0	Z=0																
RX=180	RX=180																
RY=0	RY=0																
RZ=180	RZ=-90																
FIG=5	FIG=5																

(4)ワーク 4

- ワーク 4 の設定

ワーク番号	X	Y	Z	RX	RY	RZ
4	700	0	200	0	-45	0

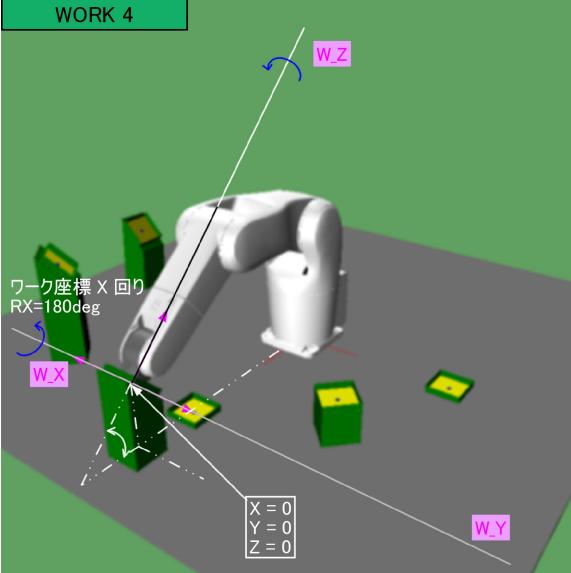


- 位置変数 P10

姿勢	位置																
<p>WORK 4</p> <p>ワーク座標 X 回り RX=180deg</p> <p>ワーク座標 Y 回り RY=0deg</p> <p>ワーク座標 Z 回り RZ=180deg</p>	<table border="1"> <thead> <tr> <th>ワーク 4 の値</th> <th>ベース座標での値</th> </tr> </thead> <tbody> <tr> <td>X=0</td> <td>X=487.87</td> </tr> <tr> <td>Y=0</td> <td>Y=0</td> </tr> <tr> <td>Z=300</td> <td>Z=412.13</td> </tr> <tr> <td>RX=180</td> <td>RX=180</td> </tr> <tr> <td>RY=0</td> <td>RY=45</td> </tr> <tr> <td>RZ=180</td> <td>RZ=180</td> </tr> <tr> <td>FIG=5</td> <td>FIG=5</td> </tr> </tbody> </table>	ワーク 4 の値	ベース座標での値	X=0	X=487.87	Y=0	Y=0	Z=300	Z=412.13	RX=180	RX=180	RY=0	RY=45	RZ=180	RZ=180	FIG=5	FIG=5
ワーク 4 の値	ベース座標での値																
X=0	X=487.87																
Y=0	Y=0																
Z=300	Z=412.13																
RX=180	RX=180																
RY=0	RY=45																
RZ=180	RZ=180																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

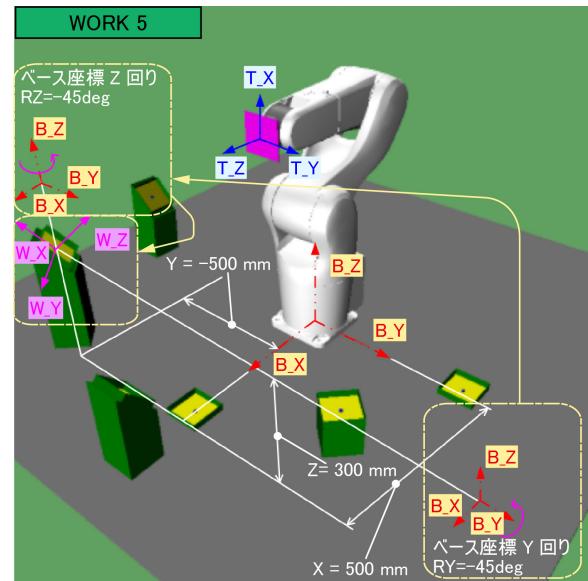
● 位置変数 P11

姿勢	位置																
	<table border="1"><thead><tr><th>ワーク 4 の値</th><th>ベース座標での値</th></tr></thead><tbody><tr><td>X=0</td><td>X=700</td></tr><tr><td>Y=0</td><td>Y=0</td></tr><tr><td>Z=0</td><td>Z=200</td></tr><tr><td>RX=180</td><td>RX=180</td></tr><tr><td>RY=0</td><td>RY=45</td></tr><tr><td>RZ=180</td><td>RZ=180</td></tr><tr><td>FIG=5</td><td>FIG=5</td></tr></tbody></table>	ワーク 4 の値	ベース座標での値	X=0	X=700	Y=0	Y=0	Z=0	Z=200	RX=180	RX=180	RY=0	RY=45	RZ=180	RZ=180	FIG=5	FIG=5
ワーク 4 の値	ベース座標での値																
X=0	X=700																
Y=0	Y=0																
Z=0	Z=200																
RX=180	RX=180																
RY=0	RY=45																
RZ=180	RZ=180																
FIG=5	FIG=5																

(5) ワーク 5

- ワーク 5 の設定

ワーク番号	X	Y	Z	RX	RY	RZ
5	500	-500	300	0	-45	-45

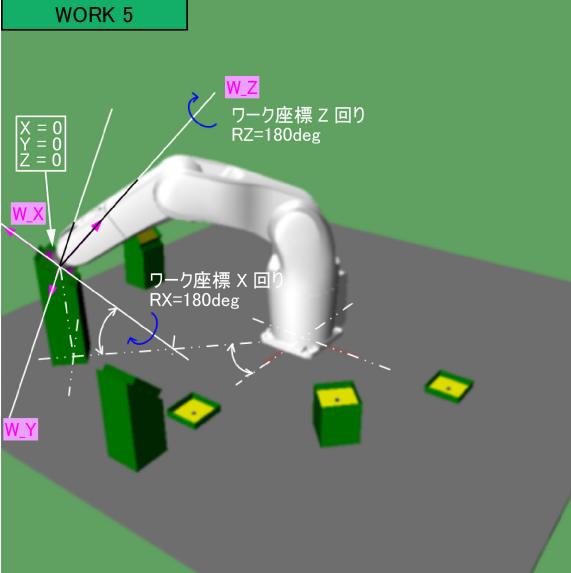


- 位置変数 P10

姿勢	位置																
<p>WORK 5</p> <p>ワーク座標 X 回り RX=180deg</p> <p>ワーク座標 Z 回り RZ=180deg</p>	<table border="1"> <thead> <tr> <th>ワーク 5 の値</th> <th>ベース座標での値</th> </tr> </thead> <tbody> <tr> <td>X=0</td> <td>X=350</td> </tr> <tr> <td>Y=0</td> <td>Y=-350</td> </tr> <tr> <td>Z=300</td> <td>Z=512.13</td> </tr> <tr> <td>RX=180</td> <td>RX=180</td> </tr> <tr> <td>RY=0</td> <td>RY=45</td> </tr> <tr> <td>RZ=180</td> <td>RZ=135</td> </tr> <tr> <td>FIG=5</td> <td>FIG=5</td> </tr> </tbody> </table>	ワーク 5 の値	ベース座標での値	X=0	X=350	Y=0	Y=-350	Z=300	Z=512.13	RX=180	RX=180	RY=0	RY=45	RZ=180	RZ=135	FIG=5	FIG=5
ワーク 5 の値	ベース座標での値																
X=0	X=350																
Y=0	Y=-350																
Z=300	Z=512.13																
RX=180	RX=180																
RY=0	RY=45																
RZ=180	RZ=135																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

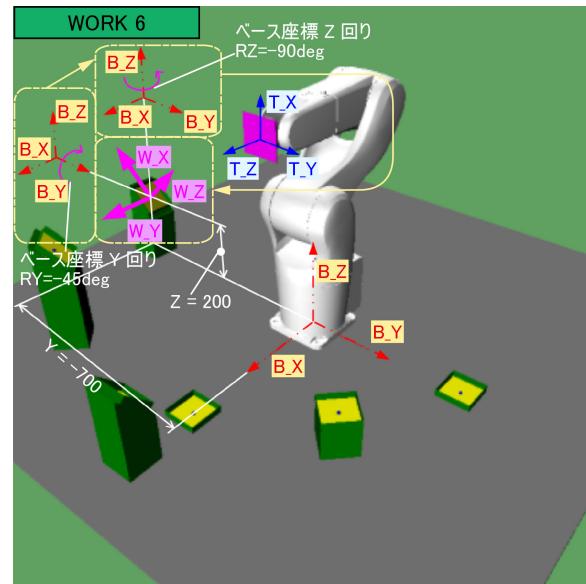
● 位置変数 P11

姿勢	位置																
 <p>The diagram illustrates a robotic arm in a green 'WORK 5' coordinate system. It shows the base coordinate system (W_X, W_Y, W_Z) and the work coordinate system (W_X, W_Y, W_Z). Arrows indicate rotations: 'ワーク座標 Z 回り RZ=180deg' (around Z-axis, RZ=180deg), 'ワーク座標 X 回り RX=180deg' (around X-axis, RX=180deg), and 'W_X' (along X-axis).</p>	<table border="1"><thead><tr><th>ワーク 5 の値</th><th>ベース座標での値</th></tr></thead><tbody><tr><td>X=0</td><td>X=500</td></tr><tr><td>Y=0</td><td>Y=-500</td></tr><tr><td>Z=0</td><td>Z=300</td></tr><tr><td>RX=180</td><td>RX=180</td></tr><tr><td>RY=0</td><td>RY=45</td></tr><tr><td>RZ=180</td><td>RZ=135</td></tr><tr><td>FIG=5</td><td>FIG=5</td></tr></tbody></table>	ワーク 5 の値	ベース座標での値	X=0	X=500	Y=0	Y=-500	Z=0	Z=300	RX=180	RX=180	RY=0	RY=45	RZ=180	RZ=135	FIG=5	FIG=5
ワーク 5 の値	ベース座標での値																
X=0	X=500																
Y=0	Y=-500																
Z=0	Z=300																
RX=180	RX=180																
RY=0	RY=45																
RZ=180	RZ=135																
FIG=5	FIG=5																

(6)ワーク 6

- ワーク 6 の設定

ワーク番号	X	Y	Z	RX	RY	RZ
6	0	-700	200	0	-45	-90

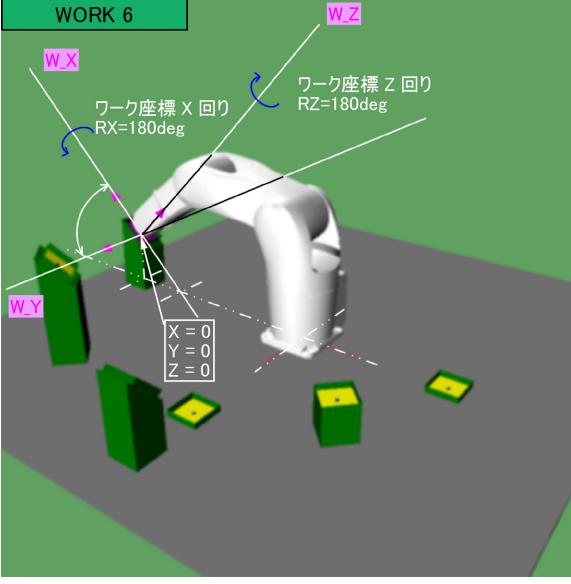


- 位置変数 P10

姿勢	位置																
<p>WORK 6</p> <p>ワーク座標 X 回り RX=180deg</p> <p>ワーク座標 Y 回り RY=45deg</p> <p>ワーク座標 Z 回り RZ=180deg</p>	<table border="1"> <thead> <tr> <th>ワーク 5 の値</th> <th>ベース座標での値</th> </tr> </thead> <tbody> <tr> <td>X=0</td> <td>X=0</td> </tr> <tr> <td>Y=0</td> <td>Y=-487.87</td> </tr> <tr> <td>Z=300</td> <td>Z=412.13</td> </tr> <tr> <td>RX=180</td> <td>RX=180</td> </tr> <tr> <td>RY=0</td> <td>RY=45</td> </tr> <tr> <td>RZ=180</td> <td>RZ=90</td> </tr> <tr> <td>FIG=5</td> <td>FIG=5</td> </tr> </tbody> </table>	ワーク 5 の値	ベース座標での値	X=0	X=0	Y=0	Y=-487.87	Z=300	Z=412.13	RX=180	RX=180	RY=0	RY=45	RZ=180	RZ=90	FIG=5	FIG=5
ワーク 5 の値	ベース座標での値																
X=0	X=0																
Y=0	Y=-487.87																
Z=300	Z=412.13																
RX=180	RX=180																
RY=0	RY=45																
RZ=180	RZ=90																
FIG=5	FIG=5																

RC8Command-Slave 活用ガイド

● 位置変数 P11

姿勢	位置																
	<table border="1"><thead><tr><th>ワーク 5 の値</th><th>ベース座標での値</th></tr></thead><tbody><tr><td>X=0</td><td>X=0</td></tr><tr><td>Y=0</td><td>Y=-700</td></tr><tr><td>Z=0</td><td>Z=200</td></tr><tr><td>RX=180</td><td>RX=180</td></tr><tr><td>RY=0</td><td>RY=45</td></tr><tr><td>RZ=180</td><td>RZ=90</td></tr><tr><td>FIG=5</td><td>FIG=5</td></tr></tbody></table>	ワーク 5 の値	ベース座標での値	X=0	X=0	Y=0	Y=-700	Z=0	Z=200	RX=180	RX=180	RY=0	RY=45	RZ=180	RZ=90	FIG=5	FIG=5
ワーク 5 の値	ベース座標での値																
X=0	X=0																
Y=0	Y=-700																
Z=0	Z=200																
RX=180	RX=180																
RY=0	RY=45																
RZ=180	RZ=90																
FIG=5	FIG=5																

改訂履歴

RC8 Command-Slave 活用ガイド

メーカー：オムロン株式会社

製品：マシンオートメーションコントローラ NJ シリーズ

Version	Date	備考
Ver. 1.0.0	2014/12/08	初版

株式会社デンソーウェーブ

- この取扱説明書の一部、または全部を無断で複製・転載することはお断りします。
- この説明書の内容は、将来予告なしに変更することがあります。
- 本書の内容については、万全を期して作成いたしましたが、万一ご不審の点や誤り、記載もれなど、お気づきの点がありましたら、ご連絡下さい。
- 運用した結果の影響については、上項にかかわらず責任を負いかねますのでご了承下さい。.

株式会社デンソーウェーブ