

はじめに

このたびは、GLC用ロジックプログラム開発ソフト「Pro-Control Editor Ver.4.3」をご採用いただき、誠にありがとうございます。

この製品を正しくご使用いただくために、マニュアル類をよくお読みください。

また、マニュアル類は必ずご利用になる場所のお手元に保管し、いつでもご覧いただけるようにしておいてください。

おことわり

- (1) 「Pro-Control Editor」(以下本製品といいます)のプログラムおよびマニュアル類は、すべて(株)デジタルの著作物であり、(株)デジタルがユーザーに対し「ソフトウェア使用許諾条件」に記載の使用権を許諾したものです。当該「ソフトウェア使用許諾条件」に反する行為は、日本国内外の法令により禁止されています。
- (2) 本書の内容については万全を期して作成しておりますが、万一お気づきの点がありましたら、(株)デジタル「サポートダイヤル」までご連絡ください。
- (3) 本製品を使用したことによるお客様の損害その他の不利益、または第三者からのいかなる請求につきましても、当社はその責任を負いかねますので、あらかじめご了承ください。
- (4) 製品の改良のため、本書の記述と本製品のソフトウェアとの間に異なった部分が生じることがあります。最新の説明は、別冊ないし電子的な情報として提供していますので、あわせてご参照ください。
- (5) 本書は、(株)デジタルから日本国内仕様として発売された製品専用です。
- (6) 本製品が記録・表示する情報の中に、(株)デジタルまたは第三者が権利を有する無体財産権、知的所有権に関わる内容を含むことがあります。これは(株)デジタルがこれらの権利の利用について、ユーザーまたはその他の第三者に、何らの保証や許諾を与えるものではありません。また本製品に記録・表示された情報を使用したことにより第三者の知的所有権などの権利に関わる問題が生じた場合、(株)デジタルはその責を負いませんのであらかじめご了承ください。

© Copyright 2003 Digital Electronics Corporation All rights reserved.

(株)デジタル 2003 September

商標・商号の権利については「商標権などについて」をご覧ください。

商標権などについて

本書に記載の社名、商品名は、各社の商号、商標(登録商標を含む)またはサービスマークです。本製品の表示・記述の中では、これら権利に関する個別の表示は省略しております。

商標等	権利者
Microsoft, MS, MS-DOS, Windows, Windows 95, Windows 98, Windows Me, Windows NT, Windows 2000, Windowsエクスプローラ, Microsoft Excel, Windows XP	米国Microsoft社
Intel, Pentium	米国Intel社
Pro-face, Flex Network	(株)デジタル
Ethernet	米国Western Digital社
NEC, PC-9800	日本電気(株)
IBM, PC/AT	米国IBM社
Adobe, Acrobat	アドビシステムズ社

なお、上記商号・商標類で、本書での表記と正式な表記が異なるものは以下の通りです。

本書での表記	正式な表記
Windows 95	Microsoft® Windows®95 オペレーティングシステム
Windows 98	Microsoft® Windows®98 オペレーティングシステム
Windows Me	Microsoft® Windows® Me オペレーティングシステム
Windows NT	Microsoft® Windows NT® オペレーティングシステム
Windows 2000	Microsoft® Windows® 2000 オペレーティングシステム
Windows XP	Microsoft® Windows® XP オペレーティングシステム
MS-DOS	Microsoft® MS-DOS® オペレーティングシステム

対応機種一覧

Pro-Control Editor Ver.4.3で対応している機種は以下のとおりです。本書では以下のようなシリーズ名または商品名を用いて説明します。下表の「GPタイプ」とは、GP-PRO/PB for Windows Ver.6.3で選択する時の名称です。

シリーズ名	商品名	型式	GPタイプ	
GLC100シリーズ	GLC100L	GLC100-LG41-24V	GLC100L	
	GLC100S	GLC100-SC41-24V	GLC100S	
GLC300シリーズ	GLC300T	GLC300-TC41-24V	GLC300T	
GLC2000シリーズ	GLC2300シリーズ	GLC2300L	GLC2300-LG41-24V	GLC2300L
		GLC2300T	GLC2300-TC41-24V	GLC2300
	GLC2400シリーズ	GLC2400T	GLC2400-TC41-24V	GLC2400
		GLC2600シリーズ	GLC2600T	GLC2600-TC41-24V

マニュアルの読み方

「GP-PRO/PB C-Package02」のマニュアルは7冊で構成されています。マニュアルの内容は別記の表をご覧ください。マニュアルはPDFファイルとしてCD-ROM(Disc2)に収録されています(導入ガイドのPDFファイルはありません)。また、データファイルとして補足説明や機能の追加・修正情報が添付されている場合があります。[スタート]ボタンをクリックし、[プログラム][Pro-face][ProPB3 C-Package]の順にポイントし、[お読みください]をクリックして表示された内容をご覧ください。

(株)デジタル製ハードウェアに関する詳しい説明は、各機種種の「ユーザーズマニュアル」(別売)をご覧ください。

GP-PRO/PB C-Package02	
導入ガイド	インストール方法と基本的なアプリケーションの開発方法について説明します。
Pro-Control Editor Ver.4.3	
ユーザーズマニュアル(本書)	GLCとの組み合わせに関するソフトウェア的な設定や変数、命令について説明します。
オペレーションマニュアル	準備から運転までの操作を習得するための演習とエラーメッセージの一覧を説明します。Pro-Control Editorで登録した変数をGP-PRO/PB で使用する方法などについても説明します。
GP-PRO/PB for Windows Ver.6.3	
オペレーションマニュアル	GP画面を作成するソフトウェアの操作手順と機能を説明します。
タグリファレンスマニュアル	GPの画面上機能を指定する「タグ」について説明します。
パーツリスト	GP画面を作成するソフトに用意されているパーツと図記号について説明します。
機器接続マニュアル(PLC接続マニュアル)	GPと各社PLC、温度調節器、インバータなどの接続について説明します。

GP-PRO/PB のマニュアルはGP画面の作成について書かれています。GLCの作画を行う場合は、GPをGLCに読み替えて操作してください。

上記のPDFマニュアル以外にもオンラインヘルプで詳しく説明していますのでご覧ください。

タグなどのアドレス設定時は標準インストール時にインストールされるレイアウトシートを利用されると便利です。

レイアウトシートには「デバイス割り付け表」と「タグレイアウトシート」があります。

それぞれMicrosoft Excel データとしてインストールされているのでご利用ください。

各ファイルの場所とファイル名を以下に示します。

なお、Microsoft Excel のご利用方法は該当商品マニュアルを参照ください。

フォルダ名	ファイル名	内容
Pro-face¥ propbwin¥sheet	Device1J.xls	デバイス割り付け表
	TAG1J.xls、 TAG2J.xls、 TAG3J.xls、 TAG4J.xls	タグレイアウトシート

CD-ROM内のPDFマニュアルはAdobe Acrobat Readerで閲覧できます。

お問い合わせ

Pro-Control Editor に関するご質問は、「サポートダイヤル」までお問い合わせください。

お問い合わせ先

月～金 9:00～17:00

大阪 TEL (06)6613-3115

東京 TEL (03)5821-1105

名古屋 TEL (052)932-4093

月～金 17:00～19:00

専用ダイヤル TEL (06)6613-3206

土・日・祝日(12月31日～1月3日を除く) 9:00～17:00

専用ダイヤル TEL (06)6613-3206

ホームページからのお問い合わせには随時承ります。

URL <http://www.proface.co.jp/>

目次

はじめに	1
対応機種一覧	2
商標権などについて	2
マニュアルの読み方	3
お問い合わせ	4
目次	5
表記のルール	9
使用上の注意	10

第1章 コントローラ機能

1.1 動作モード概要	1-1
1.1.1 コントローラ機能概略	1-2
1.1.2 RUNモードの流れ	1-4
1.1.3 GLCのスキャンの概略	1-5

第2章 変数

2.1 変数名	2-1
2.2 変数タイプ	2-3
2.3 配列変数へのアクセス	2-6

第3章 システム変数

3.1 システム変数一覧	3-1
3.1.1 システム変数使用例	3-2
3.2 システム変数詳細	3-3
3.2.1 #AvgLogicTime	3-3
3.2.2 #AvgScanTime	3-3
3.2.3 #Clock100ms	3-4
3.2.4 #ForceCount	3-5
3.2.5 #IOStatus	3-5
3.2.6 #LogicTime	3-6
3.2.7 #Platform	3-6
3.2.8 #ScanCount	3-6
3.2.9 #ScanTime	3-7
3.2.10 #Status	3-7
3.2.11 #Version	3-8
3.2.12 #Year	3-9
3.2.13 #Month	3-9
3.2.14 #Day	3-9

3.2.15 #Time	3-10
3.2.16 #Weekday	3-10
3.2.17 #WatchdogTime	3-10
3.2.18 #FaultCode	3-11
3.2.19 #FaultRung	3-12
3.2.20 #IOFault	3-13
3.2.21 #Overflow	3-13
3.2.22 #Command	3-14
3.2.23 #DisableAutoStart	3-14
3.2.24 #Fault	3-14
3.2.26 #LadderMonitor	3-15
3.2.25 #FaultOnMinor	3-15
3.2.27 #PercentAlloc	3-16
3.2.28 #RungNo	3-16
3.2.29 #Screen	3-17
3.2.30 #TargetScan	3-18

第4章 命令

4.1 命令一覧	4-1
4.2 命令詳細	4-6
4.2.1 NO(a 接点)	4-6
4.2.2 NC(b 接点)	4-7
4.2.3 OUT/M(アウト・コイル)	4-8
4.2.4 NEG(反転コイル)	4-9
4.2.5 SET(セット・コイル)	4-10
4.2.6 RST(リセット・コイル)	4-11
4.2.7 PT(立ち上がり接点)	4-12
4.2.8 NT(立ち下がり接点)	4-13
4.2.9 AND(論理積)	4-14
4.2.10 OR(論理和)	4-15
4.2.11 XOR(排他的論理和)	4-16
4.2.12 NOT(ビット反転)	4-17
4.2.13 MOV(転送)	4-17
4.2.14 BMOV(ブロック転送)	4-19
4.2.15 FMOV(フィル転送)	4-20
4.2.16 ROL(左回転)	4-21
4.2.17 ROR(右回転)	4-22
4.2.18 SHL(左シフト)	4-23
4.2.19 SHR(右シフト)	4-25
4.2.20 ADD(加算)	4-27
4.2.21 SUB(減算)	4-28
4.2.22 MUL(乗算)	4-29
4.2.23 DIV(除算)	4-30
4.2.24 MOD(剰余算)	4-31

4.2.25	INC(インクリメント)	4-32
4.2.26	DEC(デクリメント)	4-33
4.2.27	EQ(比較: =)	4-34
4.2.28	GT(比較: >)	4-35
4.2.29	LT(比較: <)	4-36
4.2.30	GE(比較: >=)	4-37
4.2.31	LE(比較: <=)	4-38
4.2.32	NE(比較: <>)	4-39
4.2.33	TON(オンディレータイマ)	4-40
4.2.34	TOF(オフディレータイマ)	4-42
4.2.35	TP(パルスタイマ)	4-44
4.2.36	CTU(アップカウンタ)	4-46
4.2.37	CTD(ダウンカウンタ)	4-47
4.2.38	CTUD(アップダウンカウンタ)	4-48
4.2.39	BCD(BCD変換)	4-50
4.2.40	BIN(バイナリ変換)	4-51
4.2.41	ENCO(エンコード)	4-52
4.2.42	DECO(デコード)	4-53
4.2.43	JMP(ジャンプ)	4-54
4.2.44	JSR(ジャンプサブルーチン)	4-55
4.2.45	RET(リターンサブルーチン)	4-55
4.2.46	FOR、NEXT(繰り返し)	4-56
4.2.47	PID(PID演算)	4-57
4.2.48	SIN(sin関数)	4-69
4.2.49	COS(cos関数)	4-69
4.2.50	TAN(tan関数)	4-70
4.2.51	RAD(ラジアン変換)	4-70

第5章 LS エリアリフレッシュ

5.1	LSエリアリフレッシュの概要	5-1
5.2	LSエリアリフレッシュの設定	5-2
5.3	GLCとPLCのデータ共有について	5-4
5.3.1	GLCと外部通信機器のデータ共有時の注意点	5-5

第6章 GLC ラダーモニタ機能

6.1	GLCラダーモニタ機能の概要	6-1
6.2	GLCラダーモニタの起動と終了方法	6-2
6.2.1	GLCラダーモニタの運転準備	6-2
6.2.2	GLCラダーモニタの起動方法	6-4
6.2.3	GLCラダーモニタの終了方法	6-4
6.3	GLCラダーモニタの各種機能	6-5
6.3.1	導通モニタ機能(ノーマル表示)	6-5
6.3.2	ラングジャンプ/スクロール機能	6-6

6.3.3	命令拡大機能 (拡大表示)	6-7
6.3.4	GLC 変数モニター機能	6-8
6.3.5	変数 / 命令検索機能	6-9

第7章 I/O ドライバ

7.1	I/O ドライバについて	7-1
7.2	Flex Network ドライバ	7-2
7.2.1	Flex Network ユニットの自己診断	7-2
7.2.2	通信チェック	7-3
7.2.3	エラー S-No.	7-4
7.2.4	I/O モニタ (I/O 工事接続チェック)	7-5
7.2.5	Flex Network 使用時のトラブルシューティング	7-11
7.3	DIO ドライバ	7-13
7.3.1	DIO の自己診断	7-13
7.3.2	I/O モニタ (I/O 工事接続チェック)	7-15
7.3.3	DIO 使用時のトラブルシューティング	7-16
7.4	ユニワイヤ I/F ドライバ	7-20
7.4.1	ユニワイヤ拡張 I/F ユニットの自己診断	7-20
7.4.2	I/O モニタ (I/O 工事接続チェック)	7-21
7.4.3	ユニワイヤ拡張 I/F ユニット使用時のトラブルシューティング	7-23

第8章 エラーと異常処理

8.1	エラーメッセージ	8-1
8.2	エラーコード	8-3
8.3	プログラムの動作異常	8-4

索引

表記のルール



本書は、以下のルールで表記します。

わかりにくいところなどは「サポートダイヤル」までお問い合わせください。「サポートダイヤル」では、(株)デジタル製品についての技術的なご質問・ご相談にお答えします。

なお、パソコンやWindows そのものに関することは、パソコンをお買い上げの販売店、メーカーにお問い合わせください。


安全に関する注意表記

本製品のご使用上、安全に関して重要な説明には、以下の表示を添えています。

表示	意味内容
 警告	この表示を無視して誤った取り扱いをすると、人が死亡または重傷を負う可能性が想定される内容を示します。
 注意	この表示を無視して誤った取り扱いをすると、人が傷害を負ったり、物的損害の発生が想定される内容を示します。
重要	この表示の説明に従わない場合、機器の異常動作やデータの消失などの不都合が起こる可能性があります。
強制	必ず実施していただきたい操作、作業などを表します。
禁止	決して行ってはならない操作、作業などを表します。

説明のための表記

本書では、説明の便宜のため、以下のように表記します。

表記	意味内容
	参考になることから、補足的な説明です。
参照	関連する説明が掲載されている項目(マニュアル名、章・節・項)を示します。
	脚注で説明している語句についています。
Pro-Control Editor	GLC のロジックプログラムを作成/転送/モニタを行う機能をもったソフトウェアです。
コントローラ	GLC に組み込まれている制御機能を指します。
GP-PRO/PB (画面作成ソフト)	GP-PRO/PB for Windows Ver.6.3 を指します。
GLC	(株)デジタル製グラフィック ロジック コントローラの総称です。
外部通信機器	PLC(プログラマブルコントローラ)、温調器、インバータなどの周辺機器を指します。ただし、Flex Network、ユニワイヤ、DIO で接続する機器を除きます。

使用上の注意



- ・ GLCでは人命や重大な物的損傷にかかわる制御は決して行わないでください。

ディスクの取り扱いについて

ディスクの破損・故障を防ぐため、以下の点にご注意ください。

強制 ・ パソコン本体の電源のON/OFFは、ディスクを抜いてから行ってください。

禁止 ・ ディスクドライブのランプが点灯しているときは、CD-ROMを取り出さないでください。

- ・ CD-ROMの記録面の磁性体面(シャッターの中)に手を触れないでください。

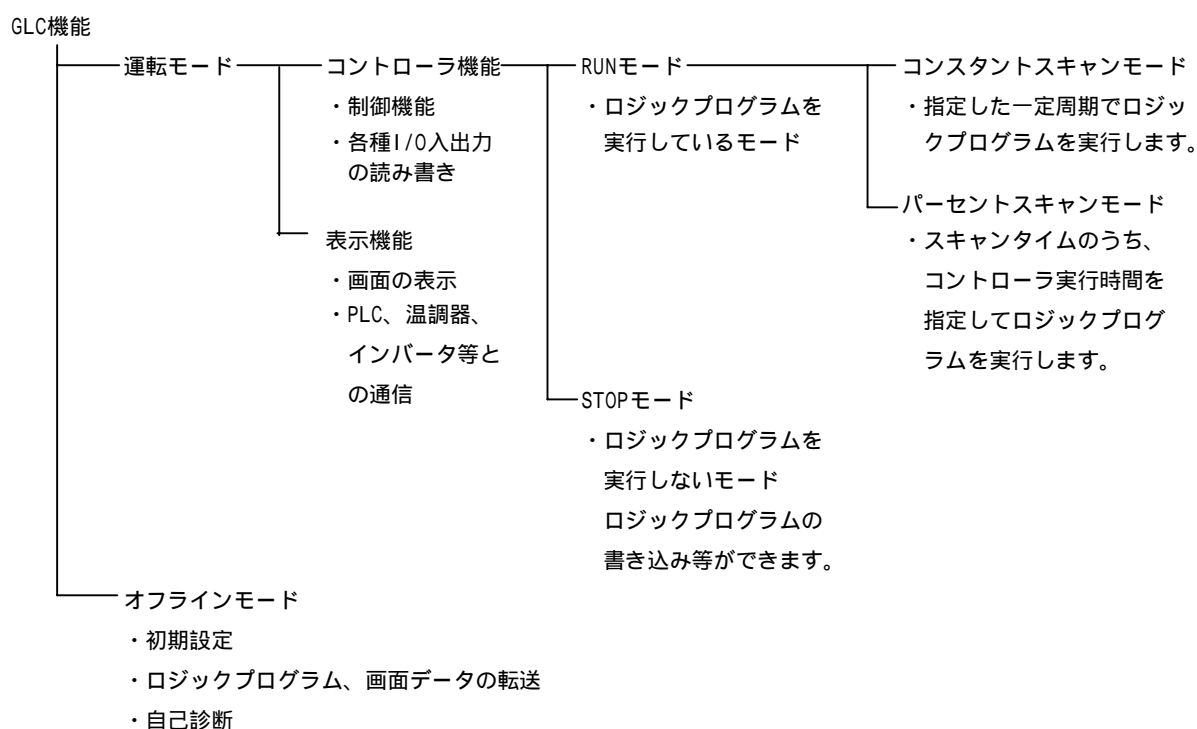
- ・ 極端な高温や低温、湿気やホコリの多い場所にディスクを置かないでください。

第1章

コントローラ機能

1.1 動作モード概要

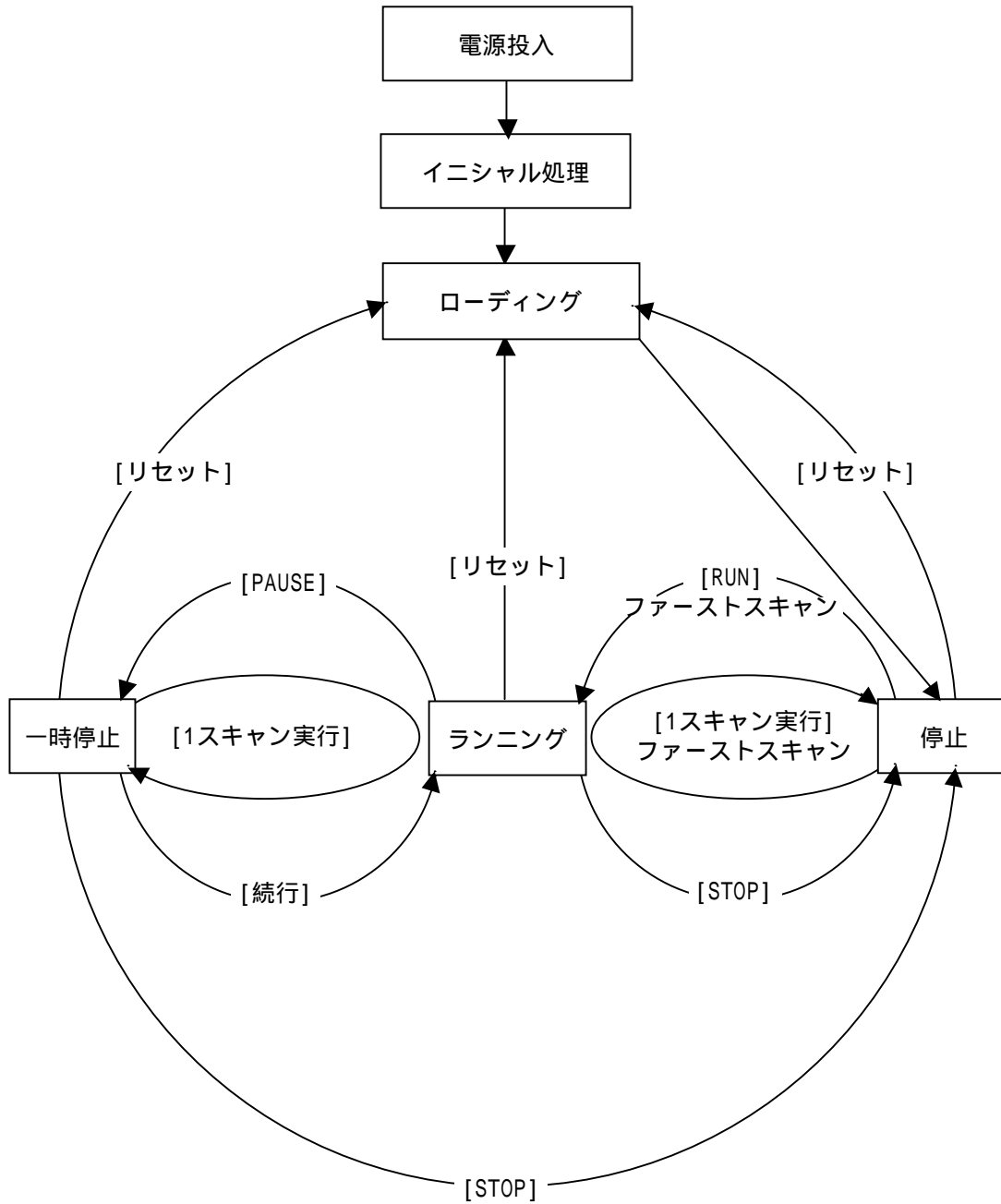
GLCには、画面の表示を実行する機能と制御を実行するコントローラ機能があります。
GLCの動作モードの概要を以下に示します。



- 重要**
- ・動作モードはシステム設計上重要です。この章をよくお読みになり、GLCの動作をご理解の上、安全を考慮したシステム設計を行ってください。
 - ・オフラインモードに入るとコントローラは停止します。運転モードに戻るとGLCはリセットされます。

1.1.1 コントローラ機能概略

コントローラ機能の動きは以下のようになっています。それぞれの状態については次ページ以降で説明しています。



イニシャル処理

ロジックプログラム実行エンジンの初期状態です。ロジックプログラム実行エンジンの初期化の後、コントローラの状態は「ローディング」に移ります。

ローディング

この状態においてメモリからロジックプログラムの読み込みを行います。ロジックプログラムが正しくロードされたかどうかのチェックを行い、正常でなければエラー処理を行います。正しくロードされれば「停止」に移ります。「電源 ON 時の動作モード」が START に設定されている場合は、[RUN] コマンドが自動実行されます。ランニングに移る時に I/O の初期化が行われます。

停止

この状態はコントローラの停止状態です。コマンド（[リセット]、[RUN]、[1 スキャン実行]、[続行]、[PAUSE]）を受けるとそれぞれの状態に移ります。

[リセット] コマンドで「ローディング」に移ります。

このとき変数の初期化が行われます。保持型変数は電断時や GLC のリセット時は直前のデータを保持しますが、モニタリングモード¹や #Command でコントローラのリセットを行ったときは、プログラミングモード²で設定した値を初期値とします。[RUN] コマンドおよび [1 スキャン実行] コマンドで非保持型変数は 0 でクリアされます。

[RUN] コマンドで「ランニング」に移ります。

[1 スキャン実行] コマンドで 1 回だけロジックプログラムを実行します。

ファーストスキャン

I/O 読み込み、START ラベルより上に記述されたロジックプログラムの実行、I/O 書き込みを行います。

ランニング

ロジックプログラム実行エンジンの継続実行状態です。I/O 読み込み、ロジックプログラムの実行、I/O 書き込み、システム変数（#AvgLogicTime、#AvgScanTime など）の更新を行います。

[リセット] コマンドで「ローディング」に移ります。

[STOP] コマンドで「停止」に移ります。

[PAUSE] コマンドで「一時停止」に移ります。

一時停止

この状態はロジックプログラム実行エンジンの一時停止状態です。I/O のウォッチドッグタイムアウトを避けるため、I/O 読み込みと I/O 書き込みを実行します。しかし、ロジックプログラムを実行しないため、出力の状態は変化しません。コマンドを受けるとそれぞれの状態に移ります。

[リセット] コマンドで「ローディング」に移ります。

[1 スキャン実行] コマンドで 1 回だけロジックプログラムを実行します。

[STOP] コマンドで「停止」に移ります。

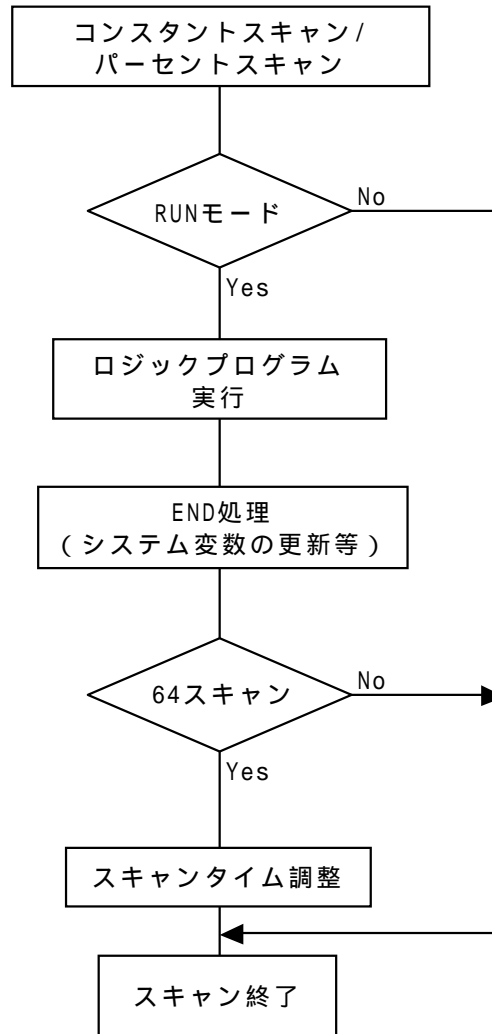
[続行] コマンドで「ランニング」に移ります。

1 コントローラで実行しているプログラムをエディタ上で実施するモード

2 プログラムを作成するモード

1.1.2 RUNモードの流れ

RUNモードの流れは以下のようになっています。



スキャンタイムの調整

スキャンタイムの調整は、64スキャンごとに行われます。コンスタントスキャンモード、パーセントスキャンモード、それぞれのスキャンタイムの調整は、以下のようになります。

コンスタントスキャンタイムモード

$$\text{スキャンタイム} = (\#AvgLogicTime \times 100) \div 50$$

パーセントスキャンモード

$$\text{スキャンタイム} = (\#AvgLogicTime \times 100) \div \#PercentAlloc$$

#AvgLogicTime、#PercentAllocについては、参照 第3章 システム変数

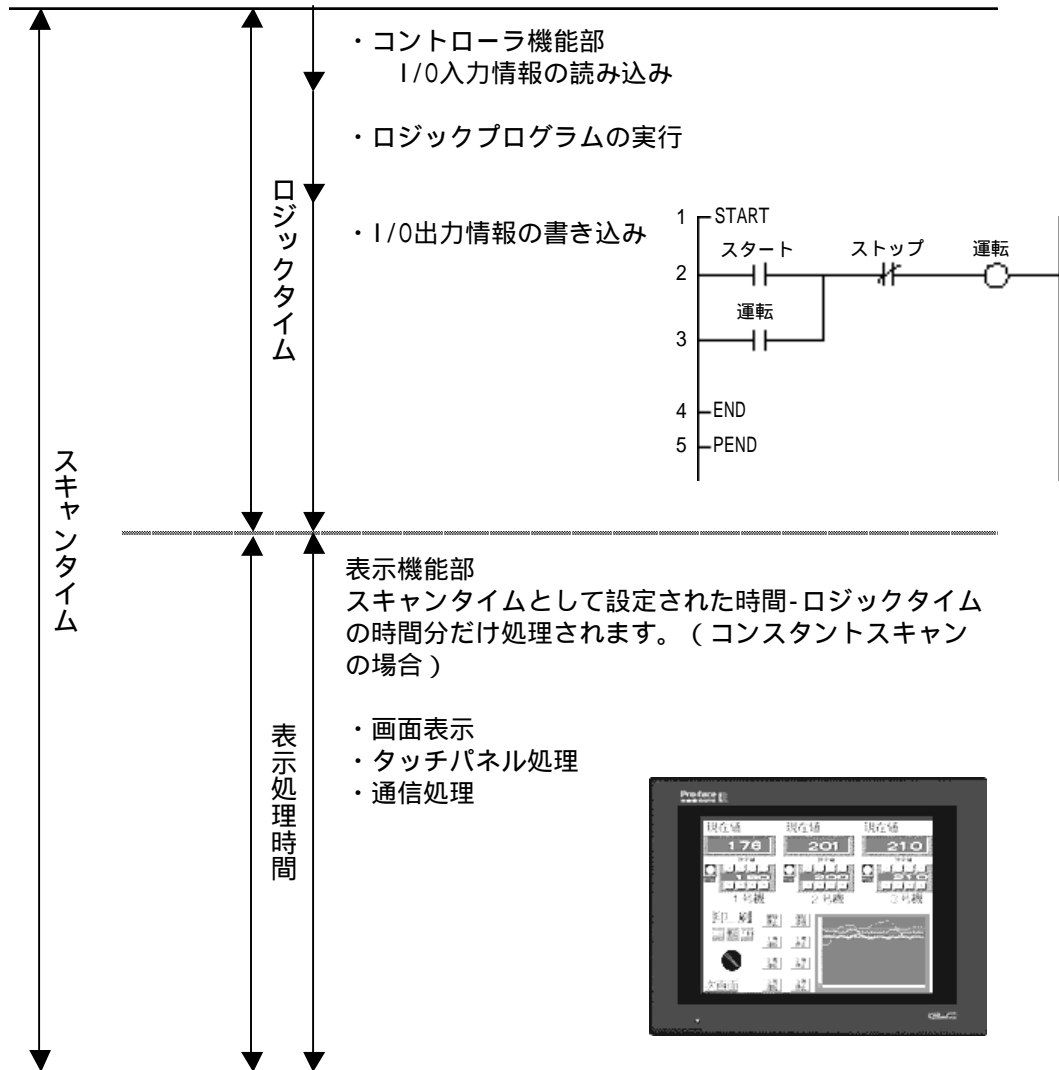
重要 ・ GLCのスキャン時間には以下の誤差が含まれます。

機種	誤差
GLC100シリーズ	約 -0.2%
GLC300シリーズ	約 +0.3%
GLC2000シリーズ	

1.1.3 GLCのスキヤンの概略

GLCのスキヤンタイムモードにはコンスタントスキヤンモードとパーセントスキヤンモードがあります。

概略として、GLCのスキヤンタイムはロジックプログラムを実行するコントローラ部と表示部（画面・タッチパネル処理、外部機器通信処理）が含まれ下図のようになっています。

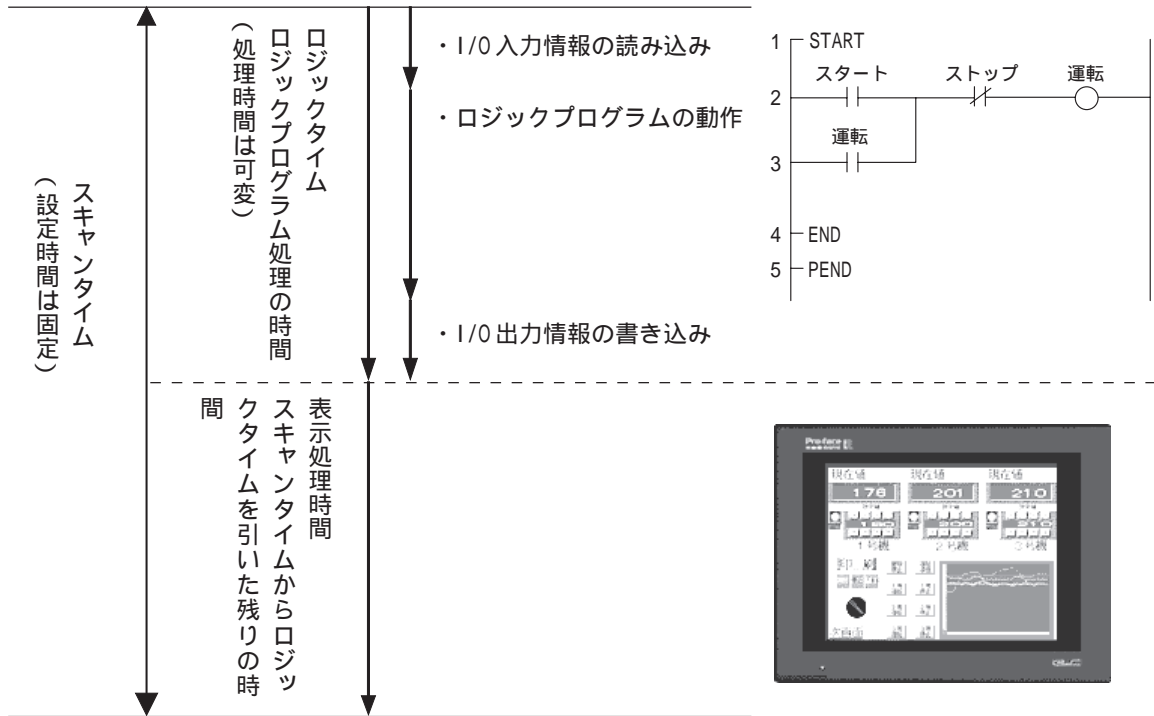


参照 1.1.2 RUN モードの流れ

コンスタントスキャンモード

設定されたスキャンタイムを一定に保ちながら動作するモードです。

画面は監視（データ表示）がメインで操作に関しては少なく、制御（ロジックプログラム）を優先するシステムに適しています。



表示処理時間 = コンスタントスキャン設定値 (ms) - ロジックタイム (可変)

例) コンスタントスキャン 50ms と設定し、ロジックタイムの実行時間が、20ms の場合

表示処理時間 = 50ms - 20ms
= 30ms

ロジックタイムが長くなれば、表示処理を行う時間は短くなります。
したがって GLC 上の表示更新速度が遅くなりますが、ロジックプログラムの処理は、
コンスタントに行われます。

重要

- ロジックタイムがコンスタントスキャン設定値の 50% を超えた場合、スキャンタイムがロジックタイムの 2 倍になるようにスキャンタイムが自動調整されます。

例) コンスタントスキャン設定値が 50ms の場合

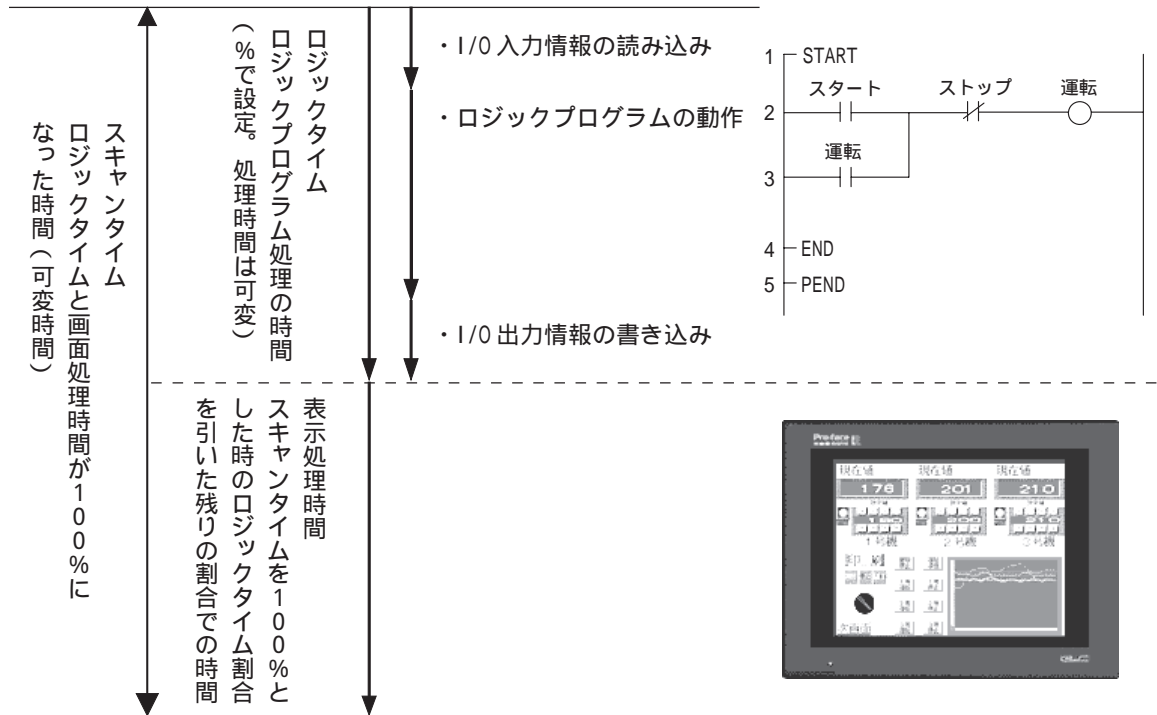
ロジックタイムが 30ms の時はスキャンタイムは 60ms になります。



- スキャンタイムの設定は 10ms 単位で入力してください。
- 設定時間は、GLC を試用運転した後、#AvgScanTime の値を参考に設定してください。参照 3.2.2 #AvgScanTime

パーセントスキャンモード

ロジックタイムを設定された割合としてスキャンタイムを可変させて動作するモードです。ロジックプログラムより画面の操作スピードや切り替えスピードを優先したい場合に用いてください。



スキャンタイム = ロジックタイム ÷ パーセントスキャン設定値(%)

例) パーセントスキャン40%と設定し、ロジックタイムの実行時間が20msの場合

$$\begin{aligned} \text{スキャンタイム} &= (20 \div 40) \times 100 \\ &= 50\text{ms} \end{aligned}$$

よって、

$$\begin{aligned} \text{表示処理時間} &= 50\text{ms} - 20\text{ms} \\ &= 30\text{ms} \end{aligned}$$

ロジックタイムが長くなると、表示処理時間も長くなるので、スキャンタイムは長くなります。

したがって、ロジックタイムが長くなればなるほど、表示処理に割り当てられる時間が長くなるので、GLC上の表示更新速度は速くなりますが、ロジックプログラムの処理周期は遅くなります。

重要

- ・ ロジックプログラムの一命令の処理時間には変化ありません。
- ・ パーセントスキャン設定値は50%を越えて設定することはできません。
- ・ パーセントスキャン設定値を50%にした場合、表示とロジックプログラムの処理が同じ時間になるので表示処理が優先されるわけではありません。



- ・ スキャンタイムの値が10ms単位になるようにパーセントスキャンを設定してください。

MEMO

このページは、空白です。
ご自由にお使いください。

第2章

変数

ここでは、Pro-Control Editor で用いられる変数タイプについて説明します。

ハードウェアに依存しない変数を使用することにより、再利用性の高いプログラムを作成することができます。

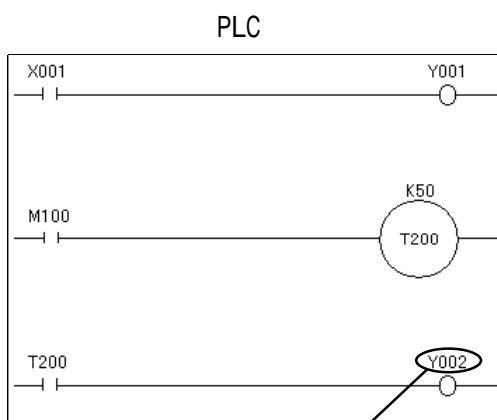
2.1 変数名

Pro-Control Editor では、I/O やカウンタのデータを格納するエリアとして変数を使用します。変数はユーザーで定義し、ロジックプログラムでそのままの名前で使用します。

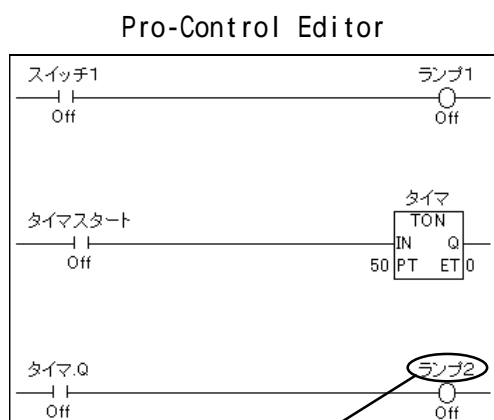
一般の PLC では、データを格納するエリアはデバイスアドレスという形で PLC メーカー特有の名称で扱われます。

	外部入出力	内部リレー	タイマ	データレジスタ
例) M社	X001	M100	T200	D00001
O社	01	1001	TIM000	DM0000
(株)デジタル	スイッチ1	タイマスタート	タイマ	運転時間

Pro-Control Editor ではこのようなデバイスアドレスに任意の名前を付け、**変数**という形でロジックプログラム中で使用します。



各メーカーごとの
デバイスアドレス



変数名
(Pro-Control Editor
でユーザーが設定した
任意の名前)

変数名をつける際の制限は以下の通りです。

- ・ 変数名は最大で半角 20 文字、全角 10 文字 (20 バイト) です。
配列の各要素およびタイマ、カウンタの各専用変数(「.PT」,「.PV」など)も文字数に含まれますので作成時には注意してください。
- ・ 全角文字と半角文字は区別されません。先に登録した変数名が有効となります。
例)「タンク」,「ﾀｸ」の順で登録した場合、「タンク」が有効となります。
- ・ 大文字小文字は区別されません。先に登録した変数名が有効となります。
例)「TANK」,「tank」の順で登録した場合、「TANK」が有効となります。
- ・ 数字で始まる変数名は全角、半角問わず使用できません。
- ・ スペースを含めることはできません。
- ・ 「_」以外の記号は使用できません。ただし、「_」を「__」のように重ねて使用することはできません。(:tank_1、 x :tank__1)
- ・ 「#」で始まる変数名はシステムで予約されているため、使用できません。
- ・ 変数名「LS」、「LSS」は、システムデータエリア、読み込みエリア、特殊リレーとしてシステムで予約されています。ユーザー定義の変数としては使用できません。
(「LS」、「LSS」は半角文字で作成してください。)

参照 第 5 章 LS エリアリフレッシュ



- ・ 変数名を付ける際にシステムによって変数名をブロック分けすると、Pro-Control Editor の変数リストから変数を探す時に便利です。(このとき、ブロック名と変数名の間に「_」を入れると見やすくなります。)

例)システムに複数のコンペア(コンペア A、B、C・・・)がある場合、コンペア A に設置されたモーターやセンサの変数名を

A_ モーター
A_ センサ

とします。

また、ディスクリート(bit)を B、整数(integer)を I、実数(float)を F として

AB_ モーター起動スイッチ
AI_ モーター回転数
AF_ モーター力率

という変数名にすると、接点やコイルに使用する変数と四則演算に使用する変数を区別することができます。

- ・ PLC のデバイス名と同様に変数名を扱いたい場合は、必要量の変数を配列で取ると便利です。

例)	PLC	Pro-Control Editor		PLC	Pro-Control Editor	
	デバイス	配列変数	変数タイプ	デバイス	配列変数	変数タイプ
	外部入力	X[100]	ディスクリート	内部リレー	M[100]	ディスクリート
	外部出力	Y[100]	ディスクリート	データレジスタ	D[100]	整数

変数の設定については参照「Pro-Control Editor オペレーションマニュアル 2.4 変数の作成
あらかじめシステムで予約されているシステム変数については参照 第 3 章 システム変数

2.2 変数タイプ

変数には、大きく分けてディスクリート(ビット)、整数、実数の3つのタイプがあります。また、これらの変数タイプで構成されたタイマとカウンタもあります。

ディスクリート、整数、実数の各変数は、配列指定もできます。配列指定の詳細については、「2.3 配列変数へのアクセス」を参照してください。

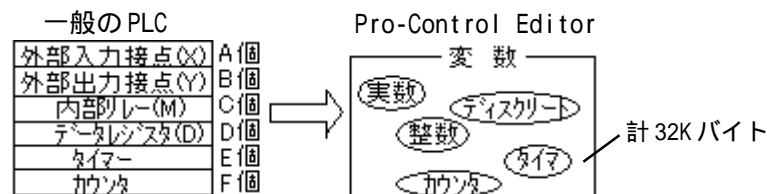
配列変数のサイズ(要素数)は最大で65535まで設定できます。

ただし、GLCの変数格納エリアは約32Kバイトです。全変数のメモリ使用量がこれを越えないように設定してください。

下表にそれぞれの変数が使用するメモリ量を示します。

変数のタイプ	使用するメモリ量(単位:バイト)
ディスクリート	12
ディスクリート配列	20+(要素数×12)
整数	8
整数配列	20+(要素数×8)
実数	16
実数配列	20+(要素数×16)
タイマ	48
カウンタ	80

PLCでは各デバイスごとに使用数に制限がありますが、GLCの変数は変数格納エリアの32Kバイト以内であれば変数タイプに関係なくいくつでも登録できます。



PLCのデバイスとPro-Controlの変数との比較

ディスクリート変数

ディスクリート変数はON/OFFを表す1ビットの長さの変数で、0か1の値を持ちます。

整数変数

整数変数は32ビットの長さの変数で、-2147483648 ~ 2147483647の整数値を持ちます。

実数変数

実数変数は64ビットの長さの変数で、 $\pm 2.225e-308$ ~ $\pm 1.79e+308$ の浮動小数点と0の値を持ちます。小数点以下15桁まで使用できます。

タイマ・カウンタ

タイマとカウンタは複数の専用変数で構成されます。

専用変数の変数タイプはそれぞれ独立しています。

タイマ

タイマ命令に使用する変数には、以下の4つの専用変数があります。

詳しくは、4.2 命令詳細を参照してください。

専用変数	内容	変数タイプ
変数.PT	設定値	整数
変数.ET	現在値	整数
変数.Q	タイマ出力ビット	ディスクリート
変数.TI	タイマ計測ビット	ディスクリート

「変数」には任意の変数名を付けることができます。



- ・変数タイマが非保持の場合でも、専用変数タイマ.PTは保持されたままです。

保持については、変数の属性を参照してください。

カウンタ

カウンタ命令に使用する変数には、以下の7つの専用変数があります。

詳しくは、4.2 命令詳細を参照してください。

専用変数	内容	変数タイプ
変数.PV	設定値	整数
変数.CV	現在値	整数
変数.R	カウンタリセット	ディスクリート
変数.UP	アップカウンタ	ディスクリート
変数.QU	アップカウンタ出力	ディスクリート
変数.QD	ダウンカウンタ出力	ディスクリート
変数.Q	カウンタ出力	ディスクリート

「変数」には任意の変数名を付けることができます。



- ・変数カウンタが非保持に指定されていても、専用変数カウンタ.PVは保持されたままです。
- ・カウンタをリセットしたスキャンでは、カウンタの更新は行いません。カウンタのリセットのために1スキャン必要となります。

保持については、変数の属性を参照してください。

変数の属性

変数には、変数タイプの他に、以下のような属性があります。

ここではそれぞれの属性について説明します。

インターナル

GLC 内部で使用します。外部入出力には使用できません。PLC では、内部リレー (内部レジスタ) に相当します。

入力 / 出力

外部入出力を使用できます。I/O コンフィギュレーションで I/O に割り付けます。PLC では入力 / 出力リレーに相当します。

I/O コンフィギュレーションについては、[参照](#) 「Pro-Control Editor オペレーションマニュアル 2.11 I/O の割り付け」

保持

保持型変数は SRAM で管理されるため、電源断時もデータ値の保持が可能です。また、保持型変数は、プログラミングモードで設定した値を初期値として持っています。電断時や GLC 本体のリセットを行った時は、直前のデータを保持しますが、モニタリングモードや #Command でコントローラのリセットを行った時、またはロジックプログラムをダウンロードした時は、プログラミングモードで設定した値で初期化されます。また、GLC の PRW ファイルを読み込むことで実行結果を Editor で保存することができます。ただし、保持型変数を初期値として使う場合、ロジックの実行中に変化するような設計にすると、Editor で読み込んだときに設定した初期値が失われるので、設計上の注意が必要です。非保持型のデータはロジックプログラム実行時に 0 クリアまたは OFF になります。



- ・ SRAM に保存されたデータは電源 OFF 時のバッテリー切れで失われます。その場合、プログラミングモードでの設定した値で初期化されます。

グローバル

グローバルは、グローバルと非グローバルがあります。画面エディタで部品などの表示機能に使用する変数はグローバルに設定してください。グローバル変数はロジックプログラムを保存することにより、自動的にシンボルエディタに GLC シンボルとして登録され、画面エディタの表示機能でも共有できるようになります。変数リストで複数個選択することでグローバル / 非グローバルの一括変換が可能です。グローバル変数は 2048 個まで設定できます。

[参照](#) Pro-Control Editor オペレーションマニュアル 4.6 変数の属性変更



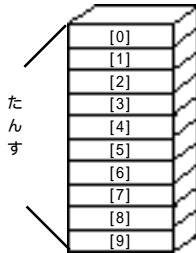
- ・ あらかじめ用意されているシステム変数は初期設定でグローバルに設定されています。

2.3 配列変数へのアクセス

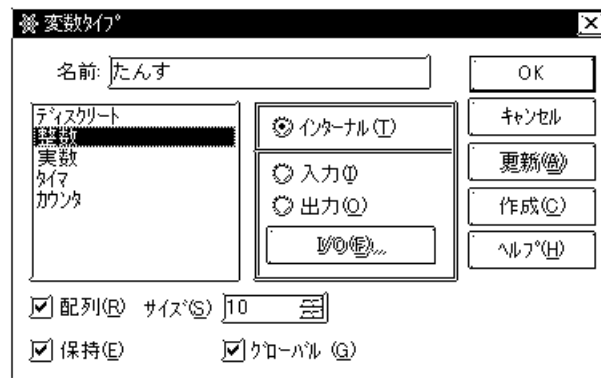
配列変数の要素・ビット・バイト・ワード単位でアクセスする方法を説明します。

配列とは、一つの変数名で複数の要素を宣言して扱う方法です。これにより同じタイプの変数をまとめて登録できます。

例えば、机やたんすの引出しを想像してください。



要素数 10 の配列変数たんすには、たんすという名前の引出しが [0] から [9] まで 10 個用意されているということになります。このたんすの各引き出しをたんす [0]、たんす [1]、...、たんす [9] と呼びます。これら一つ一つの引き出しが PLC でいうところのデータレジスタ一つ一つになります。よってたんすメモリを 10 個使用する場合は、たんすという変数名でサイズ (要素数) 10 の配列と宣言します。変数タイプの設定は以下のようにになります。



ディスクリート配列へのアクセス

ディスクリート配列では、変数名に修飾語 [n] をつけると配列の要素単位でアクセスできます。n にはアクセスする要素番号を指定します。ただし、配列の 1 番目は要素番号 0 になります。

例) ディスクリート配列モーター設定は 10 要素のディスクリート配列です。

7 番目の要素は出力コイル扇風機を制御します。

7 番目の要素を ON にすると、出力コイルが ON になります。

モーター設定の 7 番目の要素にアクセスする場合モーター設定 [6] とします。



整数配列へのアクセス

整数配列へは、配列の要素・ビット・バイト・ワード単位でのアクセスができます。

変数名の直後に[n]をつけることで配列の要素単位でアクセスできます。

ビット・バイト・ワード単位でアクセスするには変数名に下表の修飾語をつけてアクセスします。mにはアクセス単位で何番目にアクセスするかを指定します。

アクセス単位	修飾語
ビット	.X[m]
バイト	.B[m]
ワード	.W[m]

整数配列で要素単位のアクセスをする場合

例えば、整数配列を使用して数値計算、反復情報のトラッキング、データのロギングができます。

例) 整数配列飲料水売り上げが1ヶ月間に売ったソーダの数を記録するときは、以下のようなデータ構造になります。

配列は31の整数型要素で構成され、1ヶ月(31日)の各日に対応しています。

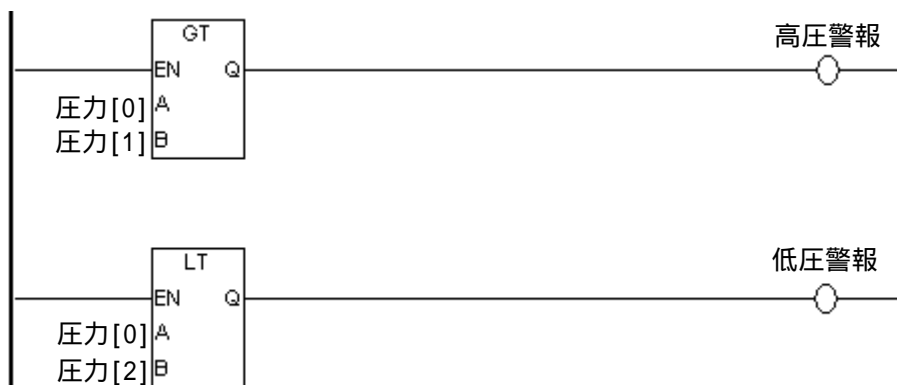
1日目	飲料水売り上げ[0]
2日目	飲料水売り上げ[1]
3日目	飲料水売り上げ[2]
4日目	飲料水売り上げ[3]
	⋮
	⋮
	⋮
28日目	飲料水売り上げ[27]
29日目	飲料水売り上げ[28]
30日目	飲料水売り上げ[29]
31日目	飲料水売り上げ[30]

以下の例では、整数配列圧力には3つの要素があります。

圧力[0]はボイラの現在の圧力、圧力[1]は圧力上限値、圧力[2]は圧力下限値を表します。

圧力が高圧限界より上がったたり、低圧限界より下がったりすると、警報がONになります。

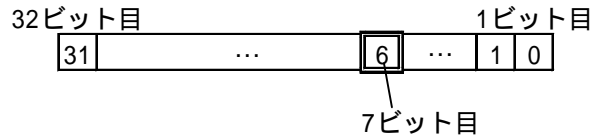
現在の圧力	圧力[0]
圧力上限値	圧力[1]
圧力下限値	圧力[2]



整数配列でビット単位のアクセスをする場合

また、整数配列はディスクリット配列変数と同様に、ビット・バイト・ワード単位でのアクセスと組み合わせてアクセスすることもできます。整数配列変数飲料水売り上げの $n+1$ 番目の要素の $m+1$ ビット目にアクセスするには飲料水売り上げ[n].X[m]のようになります。

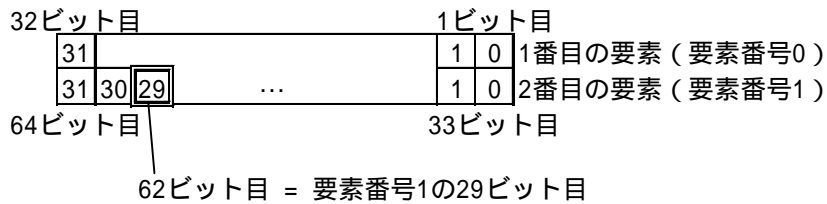
- 例) ・ 整数変数アラームの7ビット目にアクセスする場合
アラーム .X[6]



- ・ 整数配列変数飲料水売り上げの62ビット目にアクセスする場合
飲料水売り上げ .X[61]



または、飲料水売り上げ[1].X[29]



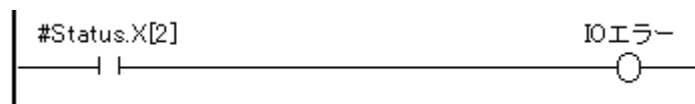
したがって、飲料水売り上げ .X[61] = 飲料水売り上げ[1].X[29]となり、どちらも整数配列飲料水売り上げの62ビット目へのアクセスになります。

- ・ 整数配列変数飲料水売り上げの6バイト目にアクセスする場合
飲料水売り上げ .B[5] または 飲料水売り上げ[1].B[1]
- ・ 整数配列変数飲料水売り上げの5ワード目にアクセスする場合
飲料水売り上げ .W[4] または 飲料水売り上げ[2].W[0]



飲料水売り上げ .X[61]と飲料水売り上げ[0].X[61]は、同じ意味です。

以下の例では、システム変数 #Status の3番目のビットを N0 命令の変数として使用します。#Status の3番目のビットは、GLC に I/O エラーがあるかどうかを通知します。したがって、3番目のビットが ON になると、出力コイル I/O エラーが ON になり、I/O エラーが発生したことを知らせます。



実数配列へのアクセス

実数配列へは、配列の要素単位でアクセスできます。
変数名に修飾語[n]をつけてアクセスします。

nにはアクセスする要素番号を指定します。ただし、配列の1番目は要素番号0になります。

例)・ 実数配列溶液温度の5番目の要素にアクセスする場合
溶液温度[4]

注：画面エディタで扱える変数の数は2048個です。配列の要素も1つの変数になります。例えば、要素数5の配列を変数として扱った数は5になります。

実数配列を使用して数値計算、反復情報のトラッキング、データのロギングができます。

例) 実数配列溶液温度が1時間毎に24時間分の溶液の温度を記録するときは、以下のようなデータ構造になります。

配列は24の実数型要素で構成され、1日のうちの1時間にそれぞれ対応しています。

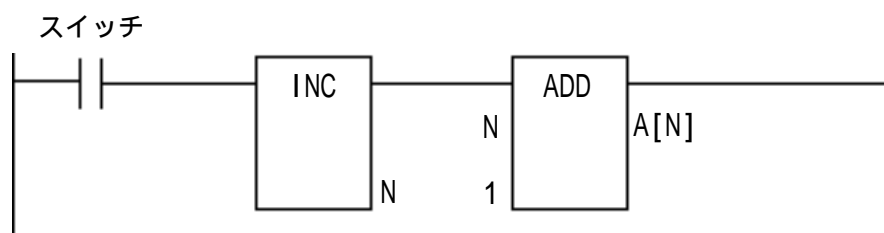
溶液温度[0]は、0:00の温度データに対応します。

溶液温度[0]
溶液温度[1]
溶液温度[2]
溶液温度[3]
⋮
⋮
⋮
溶液温度[20]
溶液温度[21]
溶液温度[22]
溶液温度[23]

配列への間接アクセス

配列の要素[n]を整数変数で間接的にアクセスすることができます。また、.X[m]、B[m]、W[m]といった修飾語の[]かっこ内の番号も間接的にアクセスすることができます。

例えば、下の回路のようにスイッチを押すと、INC命令でNはスキャンする度に1ずつ増加し、ADD命令でNと1を加算した値がA[N]に代入されます。5回スキャンしたとすれば、A[0]に1、A[1]に2、A[2]に3、A[3]に4、A[4]に5が代入されます。ただし、Nの初期値は0とします。



MEMO

このページは、空白です。
ご自由にお使いください。

第3章

システム変数

ここでは、コントローラであらかじめ定義されているシステム変数について説明します。

3.1 システム変数一覧

システム変数はコントローラの状態を表し、動作に影響します。通常の変数と同じように変数タイプを持ち同じ動きをします。システム変数はあらかじめ定義されているため、名称の変更や削除はできません。

区分	システム変数	説明	初期値	変数タイプ	
情報	#AvgLogicTime	64スキャンごとの平均ロジックタイム(読み込み、実行、書き込み)を示します。(単位:ms)	0	整数	読み込み専用
	#AvgScanTime	64スキャンごとの平均スキャンタイム(読み込み、実行、書き込み、表示処理)を示します。(単位:ms)	0	整数	
	#Clock100ms	0.1sのクロックを発生します。	-	ディスクリート	
	#EditCount	GLCでは現在は使用されていません。	-	整数	
	#ForceCount	強制変更された変数の延べ数を示します。	0	整数	
	#IOStatus	I/Oドライバの状態を示します。	-	整数[10]	
	#LogicTime	最新のロジックタイム(読み込み、実行、書き込み)を示します。(単位:ms)	0	整数	
	#Platform	コントローラのプラットフォームを示します。	-	整数	
	#ScanCount	実行されたスキャン数を示します。現在のスキャンは含みません。	0	整数	
	#ScanTime	最新のスキャンタイム(読み込み、実行、書き込み、表示処理)を示します。(単位:ms)	0	整数	
	#Status	コントローラの状態を示します。	-	整数	
	#StopPending	GLCでは現在は使用されていません。	-	ディスクリート	
	#Version	コントローラのバージョンを示します。	-	整数	
	#WCLScan	GLCでは現在は使用されていません。	-	整数	
	#WCLStatus	GLCでは現在は使用されていません。	-	整数	
	#Year	「年」をBCD2桁で格納しています。	-	整数	
	#Month	「月」をBCD2桁で格納しています。	-	整数	
	#Day	「日」をBCD2桁で格納しています。	-	整数	
#Time	「時分」をBCD4桁で格納しています。	-	整数		
#WeekDay	「曜日」を0~6の値で格納しています。	-	整数		
#WatchdogTime	エディタまたはオフラインにて設定された値を示します。(単位:ms)	-	整数		

区分	システム変数	説明	初期値	変数タイプ	
エラー	#FaultCode	最新のエラーコードを示します。	-	整数	読み込み専用
	#FaultRung	エラーが発生したラング番号を示します。	-	整数	
	#IOFault	エラーが発生したときONにします。	-	ディスクリート	
	#Overflow	算術命令または実数から整数への変換でオーバーフローが発生したときONにします。	0	ディスクリート	
設定	#Command	コントローラの動作モードを変更します。	0	整数	書き込み可能
	#DisableAutoStart	電源ON時の動作モードの設定	-	ディスクリート	
	#Fault	ErrorHandlerサブルーチン内で実行を停止するために使用します。	0	ディスクリート	
	#FaultOnMinor	マイナー異常が検出されたときロジックの実行を終了するかどうかを設定します。	0	ディスクリート	
	#LadderMonitor	GLCラダーモニタ機能の起動、操作を行います。	-	整数	
	#PercentAlloc	パーセントスキャンを設定します。 (単位:%)	0	整数	
	#PercentMemCheck	GLCでは現在は使用されていません。	-	整数	
	#RungNo	GLCラダーモニタ機能で表示する先頭ラング番号を設定します。	-	整数	
	#Screen	GLCの「画面切替」画面番号を書き込みます。(BIN/BCD)	0	整数	
	#StopScans	GLCでは現在は使用されていません。	-	整数	
#TargetScan	コンスタントスキャンを設定します。 (単位:ms)	-	整数		



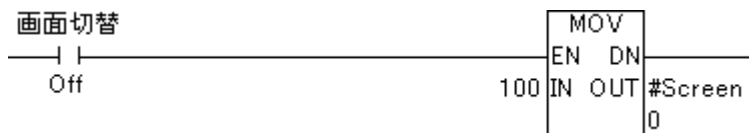
・ #Year、#Month、#Day、#Time はGLCの時計データを格納しています。時計データの変更はGLC本体の初期設定またはシステムデータエリアへの書き込みで行います。

参照 「各GLCシリーズユーザズマニュアル」(別売)「GP-PRO/PB 機器接続マニュアル(PLC 接続マニュアル)」

3.1.1 システム変数使用例

システム変数の使用方法を #Screen を例にあげて説明します。

以下のロジックプログラムは、画面番号 100 のベース画面(B100)に切り替えるためのプログラムです。スイッチを押すと、#Screen に 100 が代入されることにより、画面が切り替わります。



3.2 システム変数詳細

ここでは各システム変数の詳細を説明します。

3.2.1 #AvgLogicTime

#AvgLogicTime は、平均ロジックタイムを ms 単位で格納します。

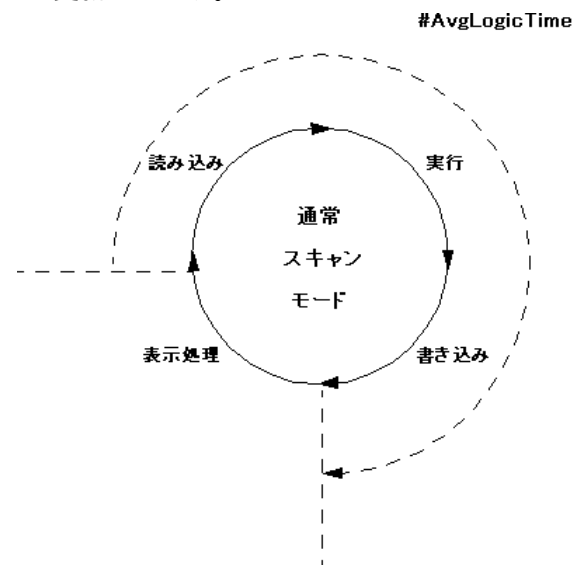
平均ロジックタイムとは、1回のスキャンで I/O の読み込み、ロジックプログラムの実行、I/O の書き込みまでに必要な時間の平均です。

#AvgLogicTime は、64 回スキャンするごとに更新されます。

変数タイプ: 整数

設定: コントローラ

読み込み専用



3.2.2 #AvgScanTime

#AvgScanTime は、平均スキャンタイムを ms 単位で格納します。

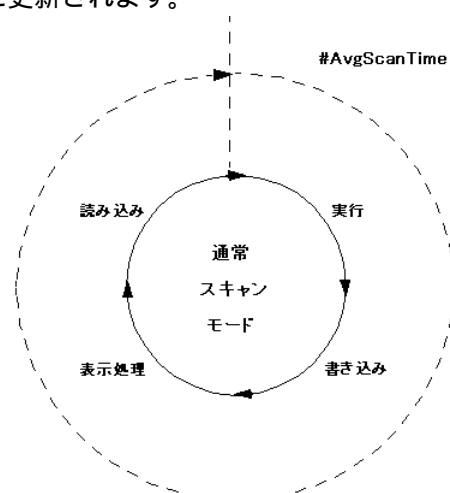
平均スキャンタイムとは、1回のスキャンで I/O の読み込み、ロジックプログラムの実行、I/O の書き込み、表示処理までに必要な時間の平均です。

#AvgScanTime は、64 回スキャンするごとに更新されます。

変数タイプ: 整数

設定: コントローラ

読み込み専用



3.2.3 #Clock100ms

#Clock100ms は 100ms のクロックを発生させます。読み込み専用ですのでクロック値は変更しないでください。

変数タイプ: ディスクリート

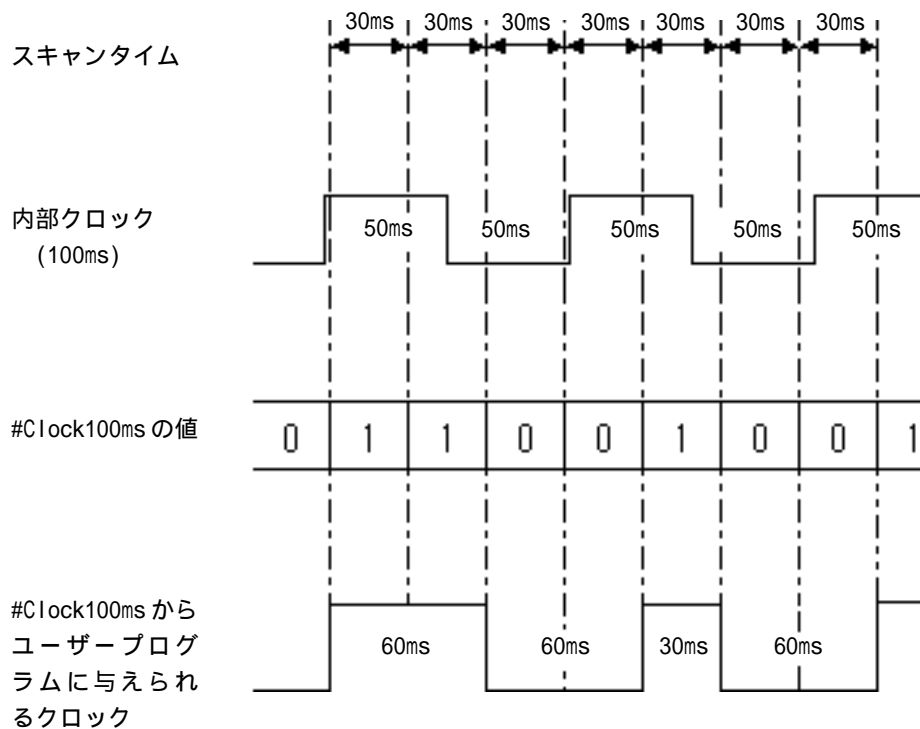
設定: コントローラ

読み込み専用



- GLCのスキャンタイムが50msを超える場合、#Clock100msのクロックは保障されません。
- #Clock100msのクロックは、GLCの各スキャンの初めに内部クロックの100msクロックを読み込んでいるため、誤差が生じます。
- #Clock100ms は、GLC2000シリーズのみ対応です。

スキャンタイムが30msの場合



- #Clock100ms には、スキャンタイムと同等の誤差を含みます。

3.2.4 #ForceCount

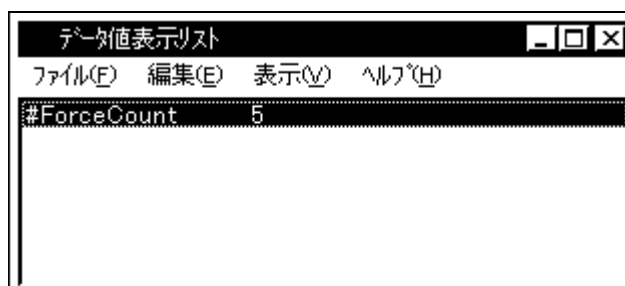
#ForceCount は、現在のロジックプログラムに強制変更された数がいくつあるかを格納します。強制変更については、Pro-Control Editor オペレーションマニュアル 3.2 コントローラの RUN/STOP を参照してください。

変数タイプ: 整数

設定: コントローラ

読み込み専用

[データ値表示リスト]ウィンドウでは、ロジックプログラムに5つの強制変更された変数があることが表示されています。



3.2.5 #IOStatus

#IOStatus は I/O ドライバによりセットされます。I/O ドライバの現在の状態を格納します。I/O ドライバの状態は #IOStatus[1] に格納されます。

#IOStatus の値が 0 のときは、I/O ドライバは正常です。0 以外の値のときは、I/O ドライバによって表示されている内容が異なります。

変数タイプ: 整数[10]

設定: コントローラ

読み込み専用

[データ値表示リスト]ウィンドウでは、I/O ドライバ 1 にエラー 802 が発生したことが表示されています。



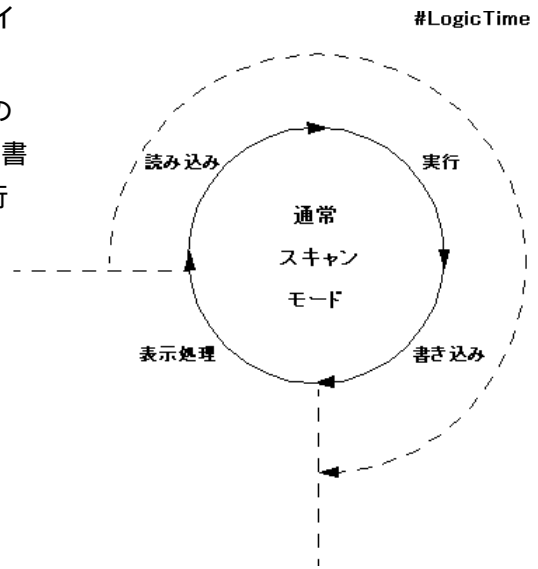
・ I/O ドライバのエラーコードの内容については、「第7章 I/O ドライバ」をご覧ください。

3.2.6 #LogicTime

#LogicTime は、前回のスキャンのロジックタイムを表します。

ロジックタイムとは、1回の スキャンで I/O の読み込み、ロジックプログラムの実行、I/O の書き込みまでに必要な時間です。表示処理を実行している時間は含まれません。

変数タイプ: 整数
 設定: コントローラ
 読み込み専用



3.2.7 #Platform

#Platform は現在コントローラが RUN しているプラットフォームを表示します。

変数タイプ: 整数
 設定: コントローラ
 初期値: 1
 読み込み専用

値	プラットフォーム
16#04	GLC100
16#14	GLC300
16#44	GLC2400
16#84	GLC2300
16#94	GLC2600

3.2.8 #ScanCount

#ScanCount はカウンタで、ロジックプログラムのスキャンが1回終わるごとにインクリメントされます。

#ScanCount の値の範囲は 0 ~ 16#FFFFFFF です。最大値(16#FFFFFFF)を超えた場合、#ScanCount の値は 0 になります。

変数タイプ: 整数
 設定: コントローラ
 読み込み専用



- #ScanCount を確認すると、ロジックプログラムが実行されているかを、容易に知ることができます。

3.2.9 #ScanTime

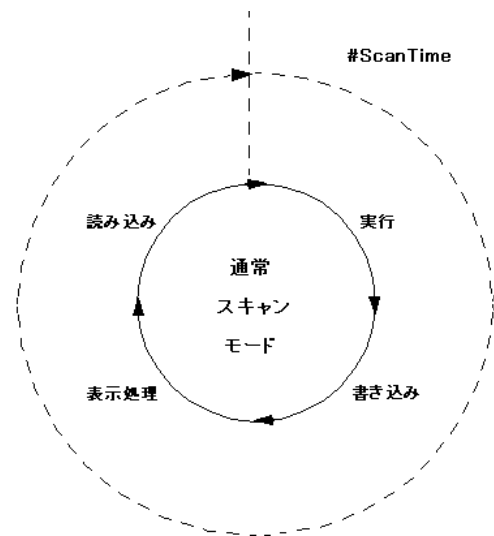
#ScanTime は、最新のスキャンタイムを ms 単位で格納します。

スキャンタイムとは、I/Oの読み込み、ロジックプログラムの実行、I/Oの出力、表示処理までに必要な時間です。

変数タイプ: 整数

設定: コントローラ

読み込み専用



3.2.10 #Status

#Statusは、コントローラの状態を表示します。バイトとビットを、以下のように定義します。

バイト0には、コントローラの現在のエラー状態が表示されます。

バイト1には、エラー状態の履歴が表示されます。コントローラをリセットしたときのみ、0にリセットされます。

バイト2には、現在の動作状態が表示されます。

バイト3は予約エリアです。

変数タイプ: 整数

設定: コントローラ

読み込み専用



- ラッチエラーフラグを使用すると、断続するエラーの検出が容易にできます。#Statusは、16進数で表示してください。

以下のエラーフラグが1ならば、以下の状態であることを示します。

現在のエラー状態

		エラーフラグ
バイト0	ビット0	メジャー異常
	ビット1	マイナー異常
	ビット2	I/Oエラー
	ビット3	予約
	ビット4	読み込みエラー
	ビット5	予約
	ビット6	スキャンエラー
	ビット7	予約

エラー状態の履歴

		ラッチエラーフラグ
バイト1	ビット8	メジャー異常
	ビット9	マイナー異常
	ビット10	I/Oエラー
	ビット11	予約
	ビット12	読み込みエラー
	ビット13	予約
	ビット14	スキャンエラー
	ビット15	予約

現在の動作状態

		コントローラの状態
バイト2	ビット16	RUN中
	ビット17	I/O使用可/使用不可
	ビット18	強制変更有効/無効
	ビット19	PAUSE
	ビット20	予約
	ビット21~23	予約

バイト3	予約
------	----

3.2.11 #Version

#Versionは、コントローラのバージョン番号を表します。#Versionは、16進数で表示されます。

変数タイプ: 整数
 設定: コントローラ
 読み込み専用

バイト番号	内容	V.1.0.0の場合
バイト3	メジャーバージョン	01
バイト2	マイナーバージョン	00
バイト1	予約	——
バイト0	予約	——

3.2.12 #Year

#Year はコントローラに設定されている「年」をBCD2桁で示します。

変数タイプ：整数

設定：コントローラ

読み込み専用

年、月、日、時分は以下のシステム変数で表されます。

例)2001年7月14日6時19分の場合

	年	月	日	時分
システム変数	#Year	#Month	#Day	#Time
値	1	7	14	619



・ #Year は、GLC2000シリーズのみ対応です。

3.2.13 #Month

#Month はコントローラに設定されている「月」をBCD2桁で示します。

変数タイプ：整数

設定：コントローラ

読み込み専用

年、月、日、時分は以下のシステム変数で表されます。

例)2001年7月14日6時19分の場合

	年	月	日	時分
システム変数	#Year	#Month	#Day	#Time
値	1	7	14	619



・ #Month は、GLC2000シリーズのみ対応です。

3.2.14 #Day

#Day はコントローラに設定されている「日」をBCD2桁で示します。

変数タイプ：整数

設定：コントローラ

読み込み専用

年、月、日、時分は以下のシステム変数で表されます。

例)2001年7月14日6時19分の場合

	年	月	日	時分
システム変数	#Year	#Month	#Day	#Time
値	1	7	14	619



・ #Day は、GLC2000シリーズのみ対応です。

3.2.15 #Time

#Timeはコントローラに設定されている「時分」をBCD4桁で示します。

変数タイプ: 整数

設定: コントローラ

読み込み専用

年、月、日、時分は以下のシステム変数で表されます。

例)2001年7月14日6時19分の場合

	年	月	日	時分
システム変数	#Year	#Month	#Day	#Time
値	1	7	14	619



・ #Timeは、GLC2000シリーズのみ対応です。

3.2.16 #Weekday

#Weekdayは、現在の曜日を0～6の値で表示します。

変数タイプ: 整数

設定: コントローラ

読み込み専用

#WeekdayはLS2054の値を反映しています。LS2054は日が変わる(23時59分 00時00分)タイミングで0～6の値を循環インクリメント(・・・5 6 0 1・・・)しています。

#Weekdayの値は0～6の範囲ですが、#Weekdayの値と実際の曜日との対応付けは、ユーザー側で任意に決めてください。対応付けは必ず表示器の画面上からLS2062(書き込み専用)に対し、0～6の値を設定値表示器などで行ってください。



・ #Weekdayは、GLC2000シリーズのみ対応です。

3.2.17 #WatchdogTime

#WatchdogTimeで、ウォッチドッグタイマの値をms単位で設定します。

#ScanTimeがこの値を超えると、メジャー異常が発生します。

参照 8.2 エラーコード

#WatchdogTimeは、初期設定として設定するか、コントローラのRUN中にモニタリングモードで設定できます。通常は、[設定]ダイアログボックスで設定します。

変数タイプ: 整数

設定: ユーザー

初期値: 500ms

書き込み可能

3.2.18 #FaultCode

#FaultCode では、最新のエラー状態が識別されます。

リセットですべて0にクリアされます。

参照 「8.2 エラーコード」

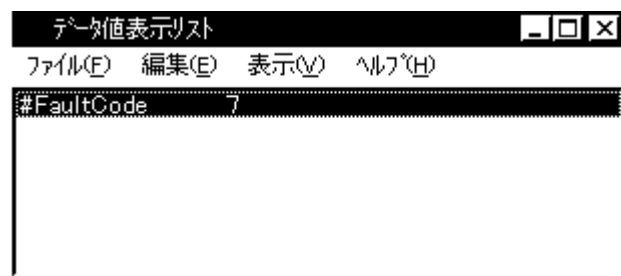
変数タイプ: 整数

設定: コントローラ

読み込み専用

コード	程度	原因
0	正常	エラーはありません。
1	マイナー	算術命令の結果、または実数から整数への変換結果がオーバーフローしました。
2	メジャー	配列の領域を超えて参照されました。
3	メジャー	整数(32ビット)の範囲を超えたビットが参照されました。
4	メジャー	スタックがオーバーフローしました。
5	メジャー	不正な命令コードを使用しています。
6		システムで予約
7	メジャー	スキャンタイムがウォッチドッグタイムを超えました。
8		システムで予約
9	メジャー	ソフトウェアのエラーです。場合によっては、システムをリブートし、回復する必要があります。
10		システムで予約
11		システムで予約
12	マイナー	BCD/BIN変換エラー
13	マイナー	ENCO/DECOエラー ¹
14		システムで予約
15	マイナー	バックアップメモリ(SRAM)のロジックプログラムが壊れています。FEPROMのロジックプログラムが実行されます。 ¹

1 GLC2000シリーズのみで発生するエラーです。



[データ値表示リスト]ウィンドウに、#FaultCode 7が表示されています。スキャンタイムがウォッチドッグタイムを超えたことを表しています。

3.2.19 #FaultRung

#FaultRung は、エラーが発生したラングの番号が格納されます。

エラーがないときは、0が格納されています。

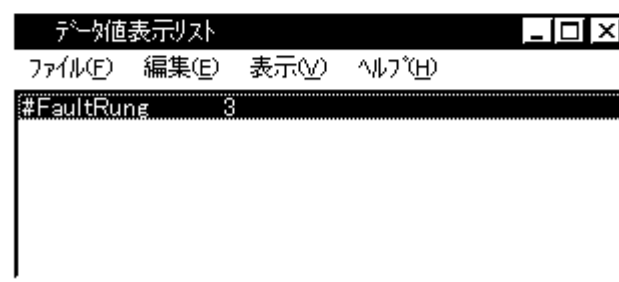
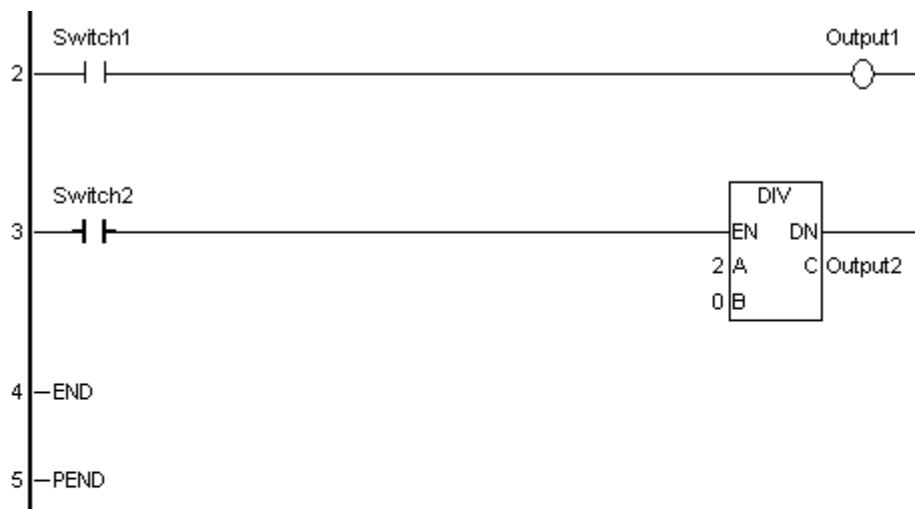
以下の例では、ラング3でエラーが発生したことが示されています。

このエラーは、DIV命令を実行したときに、整数を0で除算したことが原因です。次のエラーが発生するか、またはコントローラをリセットするまで残ります。

変数タイプ: 整数

設定: コントローラ

読み込み専用



3.2.20 #IOFault

I/OドライバでI/Oエラーが発生したときに、#IOFaultがONになります。

このエラーは次のエラーが発生するか、またはコントローラをリセットするまで残ります。

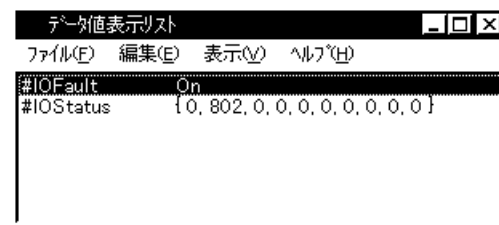
#IOStatusを確認すると、I/Oドライバの詳細な状態を知ることができます。

#IOFaultがONになると、[データ値表示リスト]ウィンドウに#IOFaultが表示されます。

変数タイプ: ディスクリート

設定: コントローラ

読み込み専用



- ・ I/Oドライバのエラーコードの内容については、「第7章 I/Oドライバ」をご覧ください。

3.2.21 #Overflow

#Overflowは、演算エラーが発生するとONになり、次に算術命令または変換をするまでONのままです。

演算エラーには、演算命令時のオーバーフロー、実数から整数への変換時のオーバーフロー、0での除算などがあります。

演算エラーが発生すると、マイナー異常が発生します。参照 8.2 エラーコード

"ErrorHandler" サブルーチンがあれば実行されます。"ErrorHandler" サブルーチンはエラー処理サブルーチンです。"ErrorHandler" という名前であらかじめ作成しておく必要があります。

コントローラは#Faultをもとにして、実行を停止させることができます。

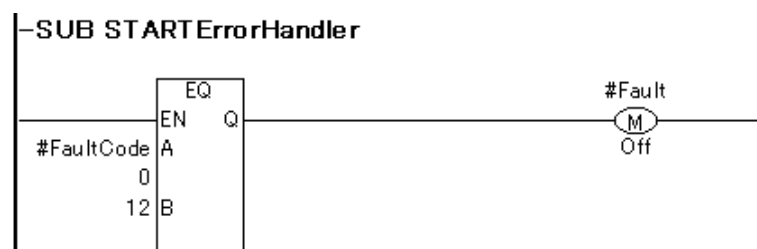
参照 3.2.24 #Fault

変数タイプ: ディスクリート

設定: コントローラ

読み込み専用

例)以下の"ErrorHandler" サブルーチンはBCD/BIN変換エラーを検出して、ロジックプログラムの実行を停止させます。



- ・ 実数から整数への変換でもオーバーフローが発生しない場合は、#OverflowはONになりません。

3.2.22 #Command

#Commandはコントローラに対する制御コマンドとして使用される整数変数です。

コントローラは#Commandを認識した後、ビット7以外を0にリセットします。複数のビットがONになっている場合、最下位ビットが優先されます。

変数タイプ: 整数

設定: ユーザー

初期値: OFF(全ビット)

書き込み可能

ビット0(=1)	コントローラのSTOP
ビット1(=2)	コントローラのRUN
ビット2(=4)	コントローラのリセット
ビット3(=8)	1スキャン実行
ビット4(=16)	続行
ビット5(=32)	PAUSE
ビット7(=128)	I/O使用可

3.2.23 #DisableAutoStart

ONにして電源を入れると、コントローラはSTOPモードで立ち上がります。

OFFにして電源を入れると、コントローラは前回の電断時の動作モード(RUN/STOP)で立ち上がります。

ただし、GLCの初期設定で、コントローラ設定の「電源ON時の動作モード」を「DEFAULT」に設定した場合のみ上記の設定が有効となります。

変数タイプ: ディスクリート

設定: ユーザー

初期値: OFF

書き込み可能

3.2.24 #Fault

#Faultは、"ErrorHandler" サブルーチンの終了時に、コントローラがロジックプログラムの実行を停止するか継続するかを判断する際に参照されます。#FaultをONにすることで、コントローラのロジックプログラムの実行を停止することが可能です。"ErrorHandler" サブルーチンについては、3.2.21 #Overflowを参照してください。

変数タイプ: ディスクリート

設定: ユーザー

初期値: OFF

書き込み可能



・ "ErrorHandler" サブルーチンがないときは、#Fault は意味を持ちません。

3.2.25 #FaultOnMinor

#FaultOnMinor は、"ErrorHandler" サブルーチンが存在しないロジックプログラム実行時にマイナー異常が発生した場合、コントローラがロジックプログラムの実行を停止するか継続するかを判断する際に参照されます。#FaultOnMinor を ON にすることで、コントローラのロジックプログラムの実行を停止することが可能です。参照 8.2 エラーコード "ErrorHandler" サブルーチンについては、3.2.21 #Overflow を参照してください。

変数タイプ: ディスクリート

設定: ユーザー

初期値: OFF

書き込み可能

3.2.26 #LadderMonitor

#LadderMonitor で、GLC ラダーモニタ機能の起動、操作を行います。各操作については、下表の通りです。

変数タイプ: 整数

設定: ユーザー

書き込み可能

ビット位置	項目
0	起動・停止ビット
1	スクロール左
2	スクロール右
3	スクロール上
4	スクロール下
5	画面/ラング切り替え (0:画面、1:ラング)
6	10進/16進切り替え (0:10進、1:16進)
7	予約
8	検索
9	上へ
10	下へ
11	変数検索 (0:しない、1:する)
12	変数選択 (リスト表示)
13	命令検索 (0:しない、1:する)
14	命令選択 (リスト表示)
15	予約
16	コントローラ (0:STOP、2:RUN)
17	検索指定のクリア
18	予約
19	予約
20	画面 (0:その他、1:GLCラダーモニタノーマル表示中)
21~30	予約
31	GLCラダーモニタ起動中



・ #LadderMonitor は、GLC2000 シリーズのみ対応です。

3.2.27 #PercentAlloc

パーセントスキャンモードの場合に使用します。

#PercentAlloc で、GLC の総処理時間に対してコントローラが使用できる割り合いを設定します。スキャンタイムの値が 10ms 単位になるように設定してください。

#PercentAlloc は、初期設定として設定するか、コントローラの RUN 中にモニタリングモードで設定できます。通常は、[設定]ダイアログボックスで設定します。

参照 1.1.2 RUN モードの流れ

変数タイプ: 整数

設定: ユーザー

範囲: 0 ~ 50%

初期値: 50

書き込み可能

3.2.28 #RungNo

#RungNo には、GLC ラダーモニタが起動している場合は表示している先頭ラング番号が格納されています。また、GLC ラダーモニタが起動していない場合は #RungNo にラング番号を書き込むことによって、GLC ラダーモニタ起動ビット(#LadderMonitor のビット 0)が OFF ON 時に指定ラング番号を先頭に GLC ラダーモニタが起動します。

GLC ラダーモニタ機能が有効の場合に使用できます。

変数タイプ: 整数

設定: ユーザー

読み込み専用(GLC ラダーモニタが起動時)

書き込み可能(GLC ラダーモニタが未起動時)



・ #RungNo は、GLC2000 シリーズのみ対応です。

3.2.29 #Screen

#Screenで、GLCの画面を切り替えます。「画面切り替え確認」機能によって#Screenの動作が以下のように異なります。

「画面切り替え確認」が有効の場合は#Screenに切り替え画面番号をセットした後、実際に画面が切り替わると0に初期化されます。[参照](#)「Pro-Control Editor オペレーションマニュアル 第2章 プログラムの作成」

変数タイプ: 整数

設定: ユーザー / コントローラ

初期値: 0

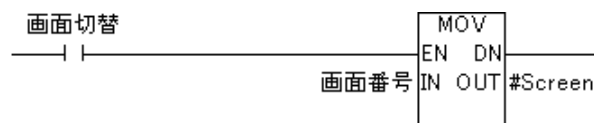
書き込み専用



- #Screenで設定された画面番号は表示させたい画面番号です。現在表示されている画面番号ではありませんのでご注意ください。
- #Screenは、書き込み専用ですので画面が切り替わったことを判別するなどの用途には使用できませんのでご注意ください。現在表示されている画面番号を確認するには、システムデータエリア (LSエリア) を参照してください。[参照](#)「5.2 LS エリアリフレッシュの設定」
- #Screen は、GLC2000 シリーズのみ対応です。



- #Screenで画面切り替えを行う場合、画面切り替えはすべてロジックプログラム上の#Screenで行ってください。タッチ入力による画面切り替えも直接#Screenに書き込まず、下の例のようにロジックプログラムからの起動にて画面切り替えを行ってください。



- 電源投入直後の初期画面を#Screenにて変更する場合には約200ms以上待つか、LSS[0].x[3](LS2032のbit3)の立ち上がり(0 → 1)のタイミングで行ってください。

3.2.30 #TargetScan

コンスタントスキャンモードの場合に使用します。

#TargetScan で、スキャンタイムを 10ms 単位で指定します。

ロジックタイムが一定の場合、#TargetScan の値を大きくすると、表示処理の処理時間を長くすることができます。また、#TargetScan の値を小さくすると、表示処理の時間が短くなります。これは、処理時間の大半をコントローラが使用するためです。

#TargetScan は、初期設定として設定するか、コントローラの RUN 中にモニタリングモードで設定できます。通常は、[設定]ダイアログボックスで設定します。

参照 1.1.2 RUN モードの流れ

変数タイプ: 整数

設定: ユーザー

範囲: 10 ~ 2000ms

初期値: 10ms

書き込み可能

第4章

命令

ここでは、Pro-Control Editor で用いられる命令について説明します。

4.1

命令一覧

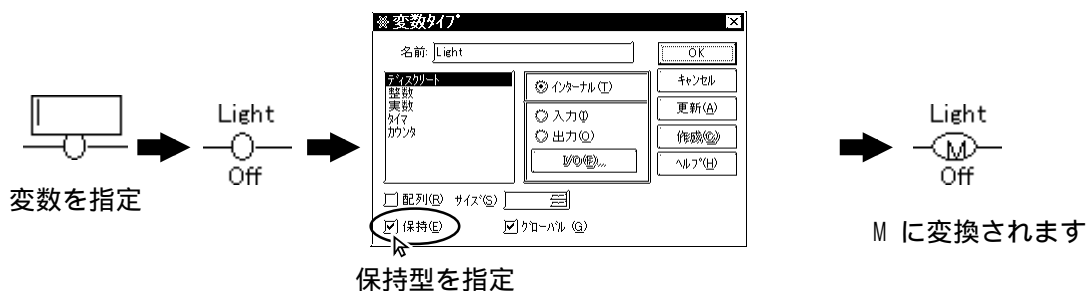
Pro-Control Editor でサポートしている命令を以下に示します。

ディスクリート命令

命令	機能	シンボル	処理内容
NO	a接点	┆┆	論理演算開始 (a接点演算開始)
NC	b接点	┆┆	論理演算開始 (b接点演算開始)
OUT/M ^{*1}	アウト・コイル/ 保持型コイル	○- / -○	出力/保持型変数への出力
NEG/NM ^{*1}	反転コイル/ 保持型反転コイル	○- / -○	反転出力/保持型変数への反転出力
SET/SM ^{*1}	セット・コイル/ 保持型セット・コイル	○S / -SM	セット/保持型変数へのセット
RST/RM ^{*1}	リセット・コイル/ 保持型リセットコイル	○R / -RM	リセット/保持型変数へのリセット
PT ^{*2}	立ち上がり接点	-I┆┆	立ち上がりによる論理演算開始
NT ^{*2}	立ち下がり接点	-N┆┆	立ち下がりによる論理演算開始

*1 これらの命令は、変数が保持型の場合に自動的に右側の命令（保持型命令）に変換されます。入力の際には左側の命令（非保持型命令）で入力してください。保持型変数についての詳細は、参照 2.2 変数タイプ

例) 下図のようにOUT命令の変数を保持型に指定するとM命令に変換されます。



*2 PT/NT 命令の最大数には個数の制限があります。参照 「Pro-Control Editor オペレーションマニュアル 第3章 ロジックプログラムを実行する」

論理演算命令

命令	機能	シンボル	処理内容
AND	論理積		A and B C 常時導通
OR	論理和		A or B C 常時導通
XOR	排他的論理和		A xor B C 常時導通
NOT	ビット反転		\bar{A} C 常時導通

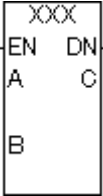
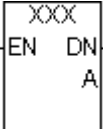
転送命令

命令	機能	シンボル	処理内容
MOV	移動		IN OUT 常時導通
BMOV	ブロック転送		<p>常時導通</p>
FMOV	フィル転送		<p>常時導通</p>


シフト命令

命令	機能	シンボル	処理内容
ROL	左回転		<p>C 常時導通</p>
ROR	右回転		<p>C 常時導通</p>
SHL	左シフト		<p>C 常時導通</p>
SHR	右シフト		<p>C 常時導通</p>

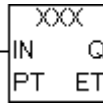
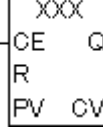
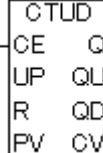
演算命令

命令	機能	シンボル	処理内容
ADD	加算		$A + B$ C 常時導通
SUB	減算		$A - B$ C 常時導通
MUL	乗算		$A \times B$ C 常時導通
DIV	除算		$A \div B$ C 常時導通
MOD	剰余算		$A \% B$ C 常時導通
INC	インクリメント		$A + 1$ A 常時導通
DEC	デクリメント		$A - 1$ A 常時導通

比較命令

命令	機能	シンボル	処理内容
EQ	比較(=)		$A = B$ のとき、導通
GT	比較(>)		$A > B$ のとき、導通
LT	比較(<)		$A < B$ のとき、導通
GE	比較(>=)		$A \geq B$ のとき、導通
LE	比較(<=)		$A \leq B$ のとき、導通
NE	比較(<>)		$A \neq B$ のとき、導通

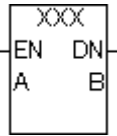
タイマ / カウンタ命令

命令	機能	シンボル	処理内容
TON	オンディレータイマ		参照 4.2.33 TON(オンディレータイマ)
TOF	オフディレータイマ		参照 4.2.34 TOF(オフディレータイマ)
TP	パルスタイマ		参照 4.2.35 TP(パルスタイマ)
CTU	アップカウンタ		参照 4.2.36 CTU(アップカウンタ)
CTD	ダウンカウンタ		参照 4.2.37 CTD(ダウンカウンタ)
CTUD	アップダウンカウンタ		参照 4.2.38 CTUD(アップダウンカウンタ)

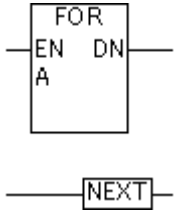



・ タイマ命令には、スキャンタイムと同等の誤差を含みます。

変換命令

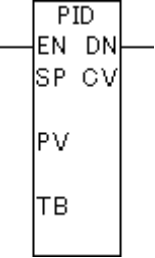
命令	機能	シンボル	処理内容
BCD	BCD変換		A BCD変換 B 常時導通
BIN	バイナリ変換		A バイナリ変換 B 常時導通
ENCO	エンコード		A エンコード変換 B 常時導通
DECO	デコード		A デコード変換 B 常時導通


プログラム制御命令

命令	機能	シンボル	処理内容
JMP	ジャンプ	->>ラベル名	ラベルの位置にジャンプ
JSR	ジャンプサブルーチン	->>サブルーチン名<<-	サブルーチンにジャンプ
RET	リターンサブルーチン	<RETURN>-	呼び出されたJSR命令に戻る
FOR、NEXT	繰り返し		Aで指定された回数、FORとNEXT間のロジックプログラムを繰り返し実行

 ・ ENCO、DECO、FOR、NEXTはGLC2000シリーズのみ対応です。

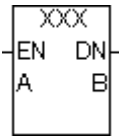
特殊命令^{*1}


命令	機能	シンボル	処理内容
PID	PID演算		EN導通時 SPとPVをPID演算して、CVに出力 EN非導通時 TBとCVにMOV

 ・ PIDはGLC2000シリーズのみ対応です。

*1 特殊命令はプロジェクトに最大数100個まで記述できます。

関数命令

命令	機能	シンボル	処理内容
SIN	sin関数		A[ラジアン] sin(A) B 常時導通
COS	cos関数		A[ラジアン] cos(A) B 常時導通
TAN	tan関数		A[ラジアン] tan(A) B 常時導通
RAD	ラジアン変換 (度 ラジアン)		A ラジアン変換 B 常時導通

 ・ SIN、COS、TAN、RAD は GLC2000 シリーズのみ対応です。

4.2 命令詳細

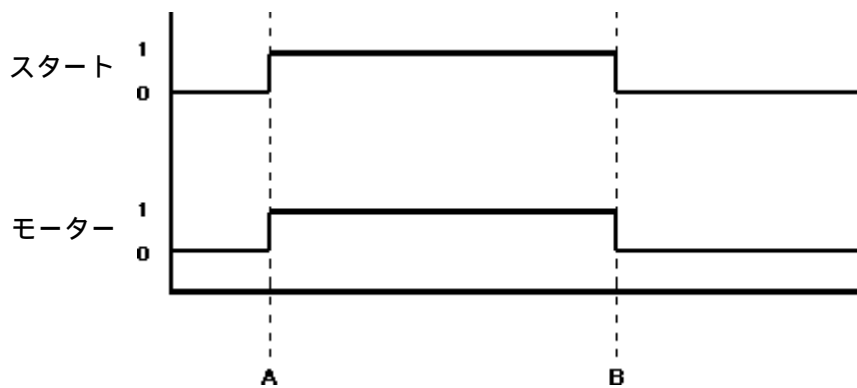
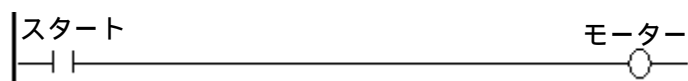
ここでは各命令の詳細を説明します。

4.2.1 NO(a 接点)

変数
—|—

NO 命令を実行すると、変数が ON のときに導通します。

以下の例では、NO 命令の機能について説明しています。



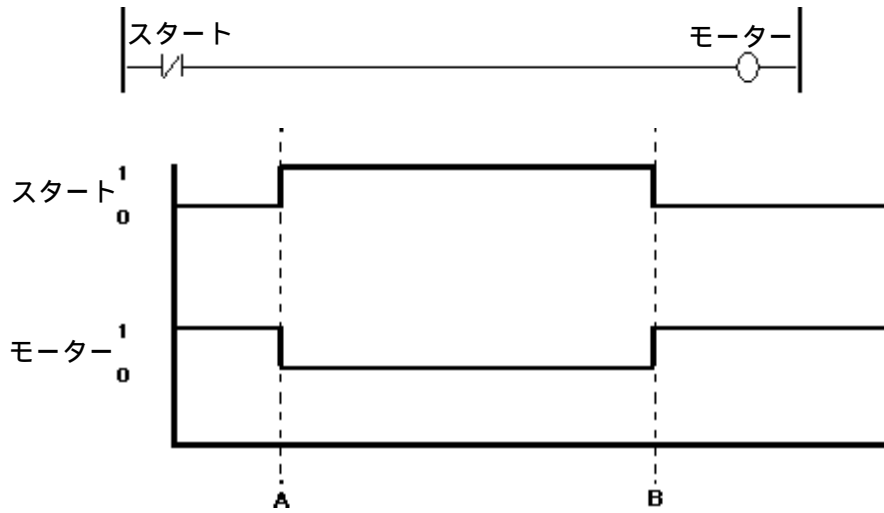
- A 変数スタートがONになると、変数モーターがONになります。
- B 変数スタートがOFFになると、変数モーターがOFFになります。

4.2.2 NC(b 接点)

変数
—|/|—

NC 命令を実行すると、変数が OFF のときに導通します。

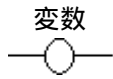
以下の例では、NC 命令の機能について説明しています。



A 変数スタートが ON になると、変数モーターが OFF になります。

B 変数スタートが OFF になると、変数モーターが ON になります。

4.2.3 OUT/M(アウト・コイル)



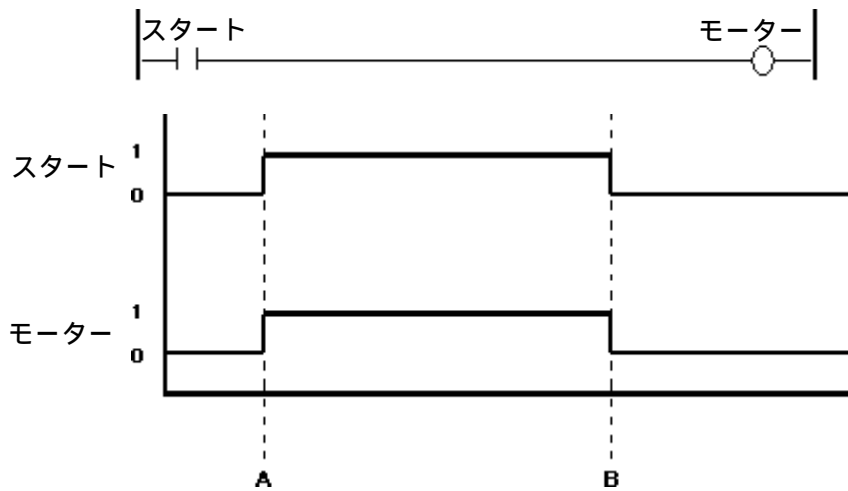
OUT 命令は、I/O に割り付けられた変数の ON/OFF や、内部メモリのディスクリット変数の ON/OFF に使用します。

この命令はコイル型出力命令なので、1 ラングあたり 1 つだけ使用でき、出力命令の右側には他の命令を使用できません。出力命令は右側の母線のすぐ左に置きます。

OUT 命令に割り付けられた変数が、保持型であれば、下記の記号がロジックプログラムに表示されます。



以下の例では、OUT 命令の機能について説明しています。



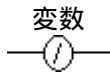
A 変数スタートが ON になり、変数モーターが ON になります。

B 変数スタートが OFF になり、変数モーターが OFF になります。



OUT 命令は非保持型変数のみ使用できます。保持型変数には M (保持型コイル) 命令を使用してください。

4.2.4 NEG(反転コイル)

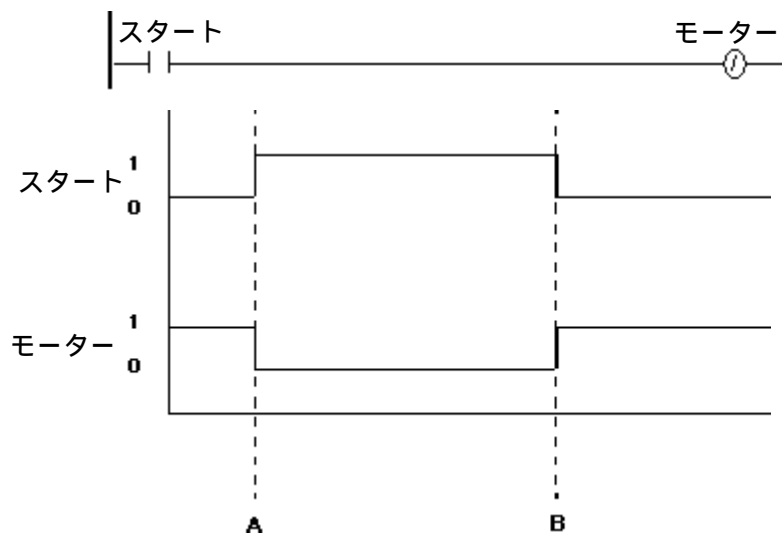


NEG 命令を実行すると、コイルに導通したとき変数が OFF になり、導通しないと ON になります。この命令はコイル型出力命令なので、1 ラングあたり 1 つだけ使用でき、出力命令の右側には他の命令を使用できません。出力命令は右側の母線のすぐ左に置きます。

NEG 命令に割り付けられた変数が、保持型であれば、下記の記号がロジックプログラムに表示されます。



以下の例では、NEG 命令の機能について説明しています。



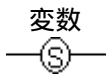
A 変数スタートが ON になり、変数モーターが OFF になります。

B 変数スタートが OFF になり、変数モーターが ON になります。



NEG 命令は非保持型変数のみ使用できます。

4.2.5 SET(セット・コイル)



コイルに導通してから SET 命令を実行すると、変数が ON になります。

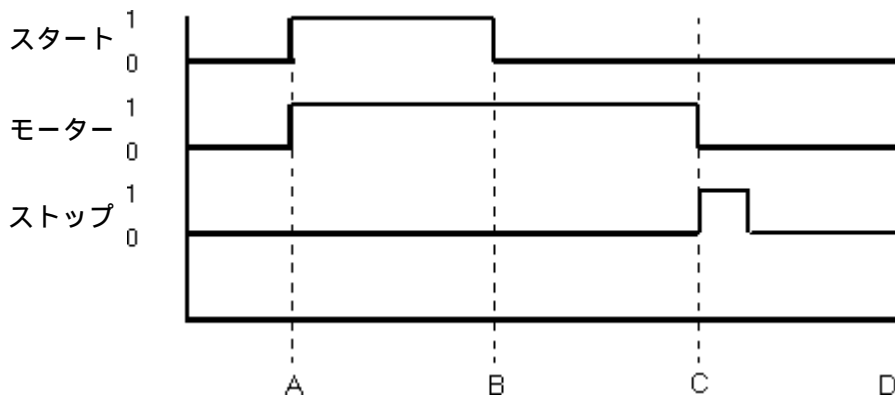
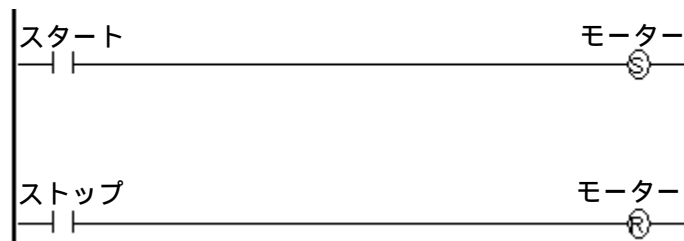
変数は、RST 命令のような他の命令で確実に OFF にされるまで、ON のままです。

この命令はコイル型出力命令なので、1 ラングあたり 1 つだけ使用でき、出力命令の右側には他の命令を使用できません。出力命令は右側の母線のすぐ左に置きます。

SET 命令に割り付けられた変数が、保持型であれば、下記の記号がロジックプログラムに表示されます。



以下の例では、SET 命令の機能について説明しています。



- A 変数スタートが ON になり、変数モーターがセットされます。
- B 変数スタートが OFF になりますが、変数モーターには影響しません。
- C 変数ストップが ON になり、変数モーターがリセットされます。
- D 変数スタートが ON になるまで、変数モーターはリセットのままです。



SET 命令は非保持型変数のみ使用できます。保持型変数には SM (保持型セット・コイル) 命令を使用してください。

4.2.6 RST(リセット・コイル)



RST 命令を実行すると、SET 命令などで ON された変数が OFF になります。

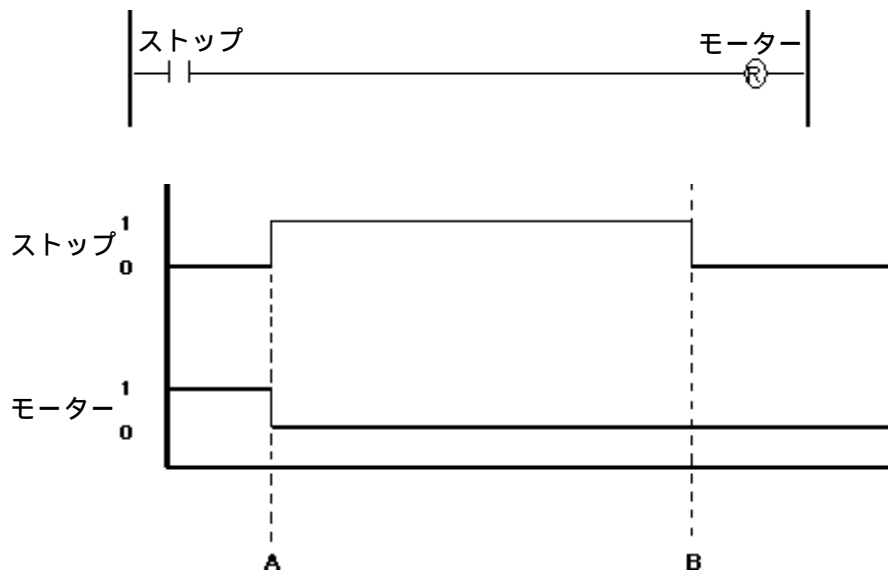
変数は、SET 命令のような他の命令で確実に ON にされるまで OFF のままです。

この命令はコイル型出力命令なので、1 ラングあたり 1 つだけ使用でき、出力命令の右側には他の命令を使用できません。出力命令は右側の母線のすぐ左に置きます。

RST 命令に割り付けられた変数が、保持型であれば、下記の記号がロジックプログラムに表示されます。



以下の例では、RST 命令の機能について説明しています。



- A 変数ストップが ON になると、変数モーターがリセットされます。
- B 変数ストップが OFF になっても、RST 命令でリセットされた変数モーターは、他の命令で ON にされるまで OFF のままです。



- ・ RST 命令は非保持型変数のみ使用できます。保持型変数には RM(保持型リセット・コイル) 命令を使用してください。
- ・ 実数、整数変数を RST 命令でリセットする(0 にする)ことはできません。

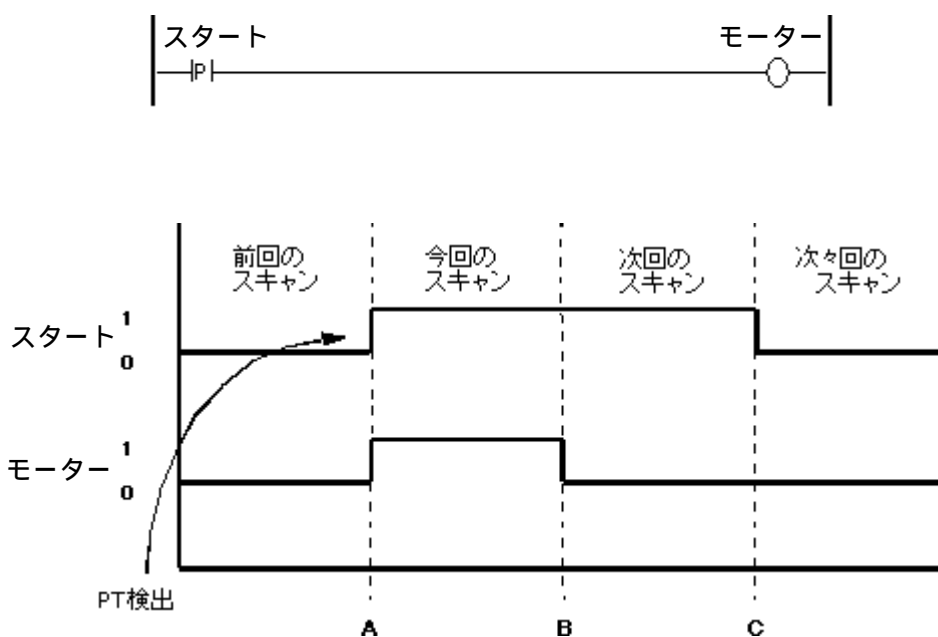
4.2.7 PT(立ち上がり接点)

変数
—PI—

PT命令を実行すると、前回のスキャンでOFFの変数が今回のスキャンでONになったときに、1スキャンの間だけ導通します。

スタートアップ時には、前回のスキャンでの立ち上がり接点の状態はOFFとされます。

以下の例では、PT命令の機能について説明しています。



- A 変数スタートがONになり、変数モーターがONになります。
- B スキャンを1回すると、変数モーターはOFFになります。
- C 変数スタートの立ち上がりを検出しなかったため、変数モーターはOFFのままです。

重要

・ PT(立ち上がり接点)命令およびNT(立ち下がり接点)命令のオペランドに対し、配列やビット指定などの各要素に変数を用いて間接アドレッシングを行う場合には注意が必要です。

前回実行時のオペランドに設定された変数と今回実行時のオペランドに設定された変数の状態を比較して実行しますので、それぞれの指定する変数値が異なる場合は状態比較の対象が異なります。

4.2.8 NT(立ち下がり接点)

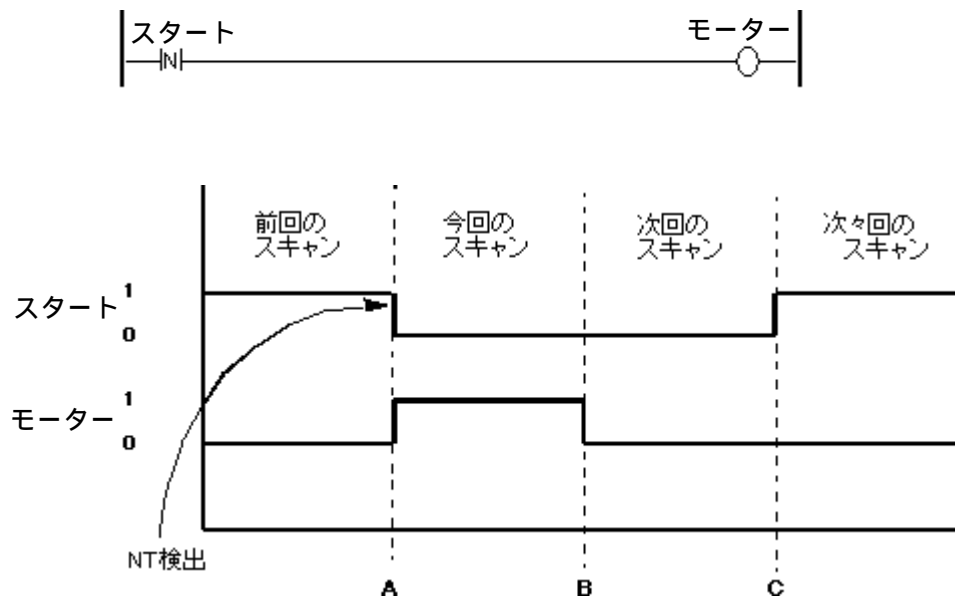
変数

—|N|—

NT 命令を実行すると、前回のスキャンで ON の変数が今回のスキャンで OFF になったときに、1 スキャンの間だけ導通します。

最初にスキャンするときは、前回のスキャンでの状態は OFF とされるので、NT 命令を実行しても導通しません。

以下の例では、NT 命令の機能について説明しています。

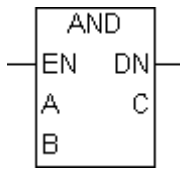


- A 変数スタートが OFF になり、変数モーターが ON になります。
- B スキャンを 1 回すると、変数モーターが OFF になります。
- C 変数スタートの立ち上がりを検出しなかったため、変数モーターは OFF のままです。

重要 ・ NT (立ち下がり接点) 命令および PT (立ち上がり接点) 命令のオペランドに対し、配列やビット指定などの各要素に変数を用いて間接アドレッシングを行う場合には注意が必要です。

前回実行時のオペランドに設定された変数と今回実行時のオペランドに設定された変数の状態を比較して実行しますので、それぞれの指定する変数値が異なる場合は状態比較の対象が異なります。

4.2.9 AND(論理積)



AND 命令を実行すると、AとBのビットがONのときのみ、CのビットがONになります。これ以外ときは、CのビットはOFFになります。

A	演算子	B	C	整数 A	0 1 1 0 ... 1 1 0 0
ON	AND	ON	ON	整数 B	1 1 0 0 ... 0 0 0 1
ON		OFF	OFF	整数 C	0 1 0 0 ... 0 0 0 0
OFF		ON	OFF		
OFF		OFF	OFF		

AND 命令は常に導通します。

AND 命令が実行できる A、B、C の組み合わせは以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数
整数配列	整数配列	整数配列
整数	整数定数	整数
整数配列	整数定数	整数配列

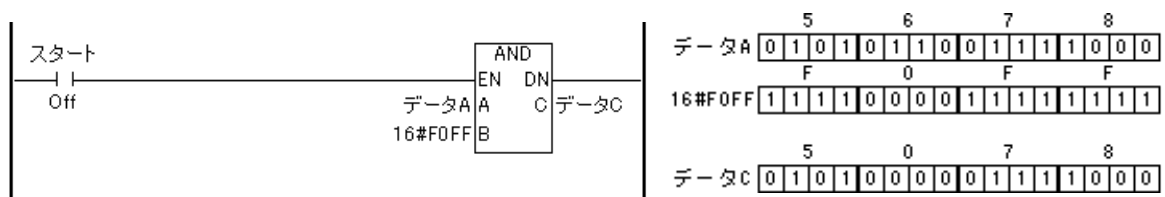
AND 命令には、3種類あります。

1. すべての変数が配列でないときは、32ビットの単純な論理積が演算されます。
2. AとCが配列で、Bが整数配列でないときは、Aのそれぞれの要素とBを論理積演算し、結果は、Cの対応するそれぞれの要素に格納されます。AとCの配列は、同じサイズにしてください。
3. 3つの変数が同じサイズの配列のときは、配列Aと配列Bの論理積が演算されます。結果は配列Cに格納されます。

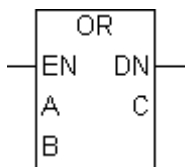
動作例

スタートがONすると、データAのBCD4桁データの3桁目を"0"にマスクし、その結果がデータCに格納されます。

例)データAが"16#5678"(16進数の5678)の場合、データCには"16#5078"が格納されます。



4.2.10 OR(論理和)



OR 命令を実行すると、A または B のビットが ON のときは、C のビットが ON になります。それ以外の場合は、C のビットが OFF になります。

A	演算子	B	C	整数 A	0 1 1 0 ... 1 1 0 0
ON	OR	ON	ON	整数 B	1 1 0 0 ... 0 0 0 1
ON		OFF	ON	整数 C	1 1 1 0 ... 1 1 0 1
OFF		ON	ON		
OFF		OFF	OFF		

OR 命令は、常に導通します。

OR 命令が実行できる A、B、C の組み合わせは以下の通りです。

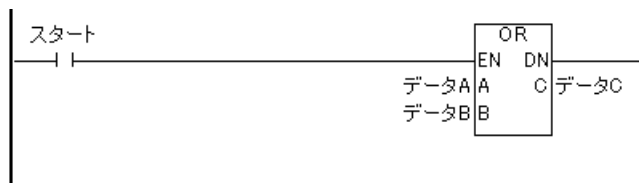
Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数
整数配列	整数配列	整数配列
整数	整数定数	整数
整数配列	整数定数	整数配列

OR 命令には3種類あります。

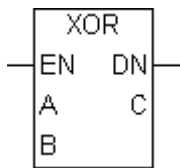
1. A、Bの変数が両方とも整数のときは、32ビットの単純な論理和が演算されます。
2. AとCが配列で、Bが整数配列でないとき、Aのそれぞれの要素とBを論理和演算し、結果は、Cの対応するそれぞれの要素に格納されます。AとCの配列は同じサイズにしてください。
3. 3つの変数が同じサイズの配列のときは、配列Aと配列Bの論理和が演算されます。結果は、配列Cに格納されます。

動作例

スタートがONすると、データAとデータBの論理和がデータCに格納されます。



4.2.11 XOR(排他的論理和)



XOR 命令を実行すると、AかBの一方のビットがONのときは、CのビットがONになります。それ以外の場合は、CのビットがOFFになります。

A	演算子	B	C
ON	XOR	ON	OFF
ON		OFF	ON
OFF		ON	ON
OFF		OFF	OFF

整数 A	0 1 1 0 ... 1 1 0 0
整数 B	1 1 0 0 ... 0 0 0 1
整数 C	1 0 1 0 ... 1 1 0 1

XOR 命令は、常に導通します。

XOR 命令が実行できる A、B、C の組み合わせは以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数
整数配列	整数配列	整数配列
整数	整数定数	整数
整数配列	整数定数	整数配列

XOR 命令には、以下の3種類があります。

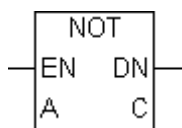
1. A、Bの変数が両方とも整数のときは、32ビットの単純な排他的論理和が演算されます。
2. AとCが配列で、Bが整数配列でないとき、Aのそれぞれの要素とBを排他的論理和演算し、結果は、Cの対応するそれぞれの要素に格納されます。AとCの配列は同じサイズにしてください。
3. 3つの変数が同じサイズの配列のときは、配列Aと配列Bの排他的論理和が演算されます。結果は、配列Cに格納されます。

動作例

スタートがONすると、データAとデータBの排他的論理和がデータCに格納されます。



4.2.12 NOT(ビット反転)



NOT 命令を実行すると、A のビットが OFF のときには C のビットが ON になり、A のビットが ON のときには、C のビットが OFF になります。

A	演算子	C	整数 A	0 1 1 0 ... 1 1 0 0
ON	NOT	OFF	整数 C	1 0 0 1 ... 0 0 1 1
OFF		ON		

NOT 命令は、常に導通します。

NOT 命令が実行できる A、C の組み合わせは以下の通りです。

Aのタイプ	Cのタイプ
整数	整数
整数配列	整数配列

NOT 命令には2種類あります。

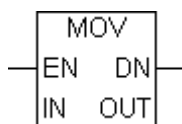
1. A の変数が整数のときは、32 ビットの単純なビット反転が演算されます。
2. A の変数が配列のときは、A の配列全体のビット反転が演算されます。結果は、C に格納されます。このとき、A と C は同じサイズにしてください。

動作例

スタートが ON すると、データ A の値がビット反転されて、データ C に格納されます。



4.2.13 MOV(転送)



MOV 命令を実行すると、IN は OUT にコピーされます。

IN と OUT の変数タイプが違うときは、結果は OUT の変数タイプに変換されます。

配列を転送するときは、IN と OUT を同じ変数タイプ・サイズにしてください。

この命令は常に導通します。MOV 命令が実行できる IN、OUT の組み合わせは次の通りです。

INのタイプ	OUTのタイプ
ディスクリート配列	INと同じサイズのディスクリート配列
整数	整数または実数の変数または配列
整数配列	INと同じサイズの整数配列または変数
整数定数	整数または実数の変数または配列
実数	整数または実数の変数または配列
実数配列	INと同じサイズの実数配列または変数
実数定数	整数または実数の変数または配列

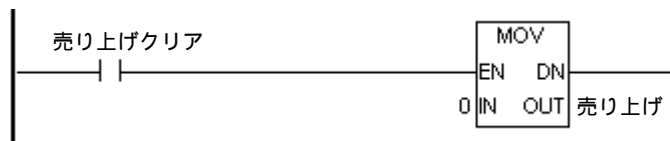


実数から整数に変換したときに、値が大きすぎて転送できないときは、#OverflowがONになります。このとき、結果は不定です。

以下の例では、MOV 命令の使用例について説明しています。

例 1: 変数をクリアする

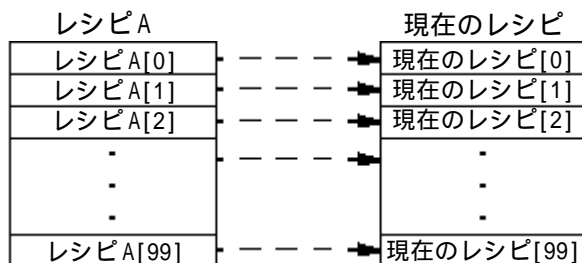
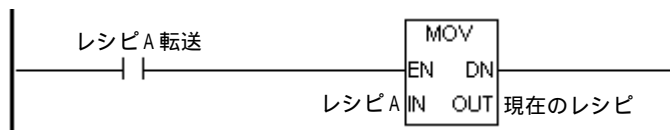
変数をクリアするときは、その変数に0を転送すると簡単です。



例 2: 配列をブロック転送をする

ブロック転送をするときは、同じタイプの2つの配列を指定して転送すると簡単です。

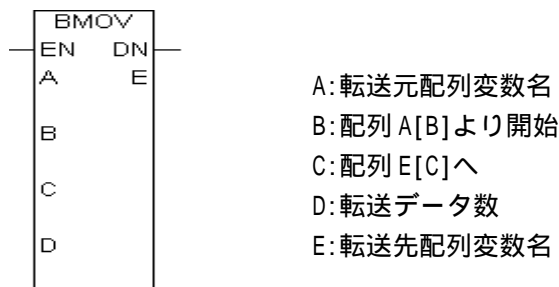
例えば、要素数100の配列レシピAを、同じタイプ、同じサイズの配列現在のレシピに転送するときは、配列レシピAを1回のMOV命令で転送すると簡単です。



ロジックプログラム中で、配列全体を表現する場合は変数名のみ記述してください。

- 例) レシピア
- × レシピア[*]
- × レシピア[100]

4.2.14 BMOV(ブロック転送)



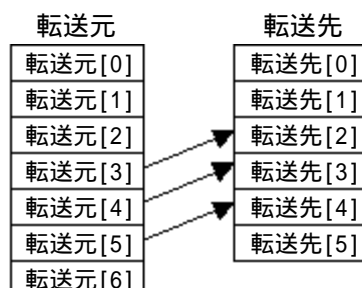
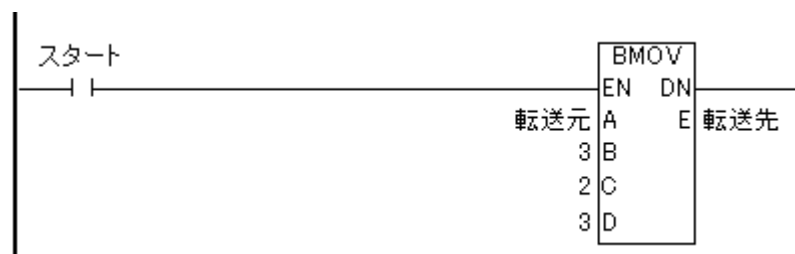
BMOV 命令を実行すると、ある一つの配列の要素のいくつかを、別の配列の要素のいくつかにコピーすることができます。すなわち、配列 A[B]から始まる D個の要素が、配列 E[C]から始まる D個の要素にコピーされます。

整数配列に対してのみ有効です。配列は、違うサイズでも転送できます。この命令は常に導通します。BMOV 命令が実行できる A、B、C、D、E はそれぞれ以下の通りです。

A、Eのタイプ	B、C、Dのタイプ
整数配列	整数
	整数定数

動作例

要素数7の整数配列転送元[3]、転送元[4]、転送元[5]を要素数6の整数配列転送先[2]、転送先[3]、転送先[4]にコピーする場合、以下のようになります。

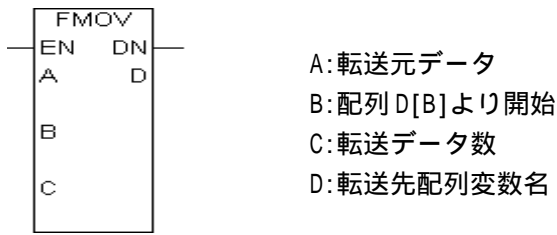


転送元[3]は、転送先[2]にコピーされます。
 転送元[4]は、転送先[3]にコピーされます。
 転送元[5]は、転送先[4]にコピーされます。

コントローラは RUN 中に BMOV 命令で配列 A や E の存在しない要素を参照していないかどうかをチェックします。無効な配列の参照が行われた場合、メジャー異常が発生し、#Fault code に 2 が設定されます。

参照 3.2.18 #FaultCode

4.2.15 FMOV(フィル転送)



FMOV 命令を実行すると、配列 D[B]の要素から C 個分に A を格納します。

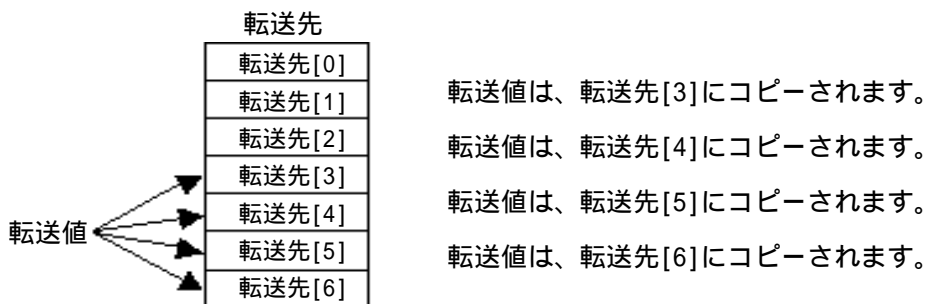
整数配列に対してのみ、有効です。この命令は常に導通します。

FMOV 命令が実行できる A、B、C、D はそれぞれ以下の通りです。

A、B、Cのタイプ	Dのタイプ
整数	整数配列
整数定数	

動作例

要素数 7 の整数配列 転送先[3]、転送先[4]、転送先[5]、転送先[6] に転送値の値をコピーする場合、以下ようになります。

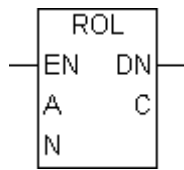


コントローラは RUN 中に FMOV 命令で配列 D の存在しない要素を参照していないかどうかをチェックします。無効な配列の参照が行われた場合、メジャー異常が発生し、

#Faultcode に 2 が設定されます。

参照 3.2.18 #FaultCode

4.2.16 ROL(左回転)



A:回転させる変数名
N:シフトビット数
C:格納先変数名

ROL 命令を実行すると、AのビットがNビット左方向にシフトします。

左端のビット(最上位ビット)は、右端のビット(最下位ビット)にローテーションします。結果はCに格納されます。

この命令は常に導通します。

ROL 命令が実行できるA、N、Cの組み合わせは以下の通りです。

Aのタイプ	Nのタイプ	Cのタイプ
整数	整数または整数定数	整数
整数配列	整数または整数定数	Aと同じサイズの配列
整数定数	整数または整数定数	整数

ROL 命令には2種類あります。

1. AとCが整数のときは、単純に32ビットローテーションされます。Nは、0以上31以下にしてください。
2. AとCが同じサイズの配列のときは、Aは大きなブロックとして扱われます。ビットは要素から次の要素へシフトされます。それぞれの要素内のみでローテーションするわけではありません。Nは0以上(32 × 配列サイズ - 1)以下にしてください。

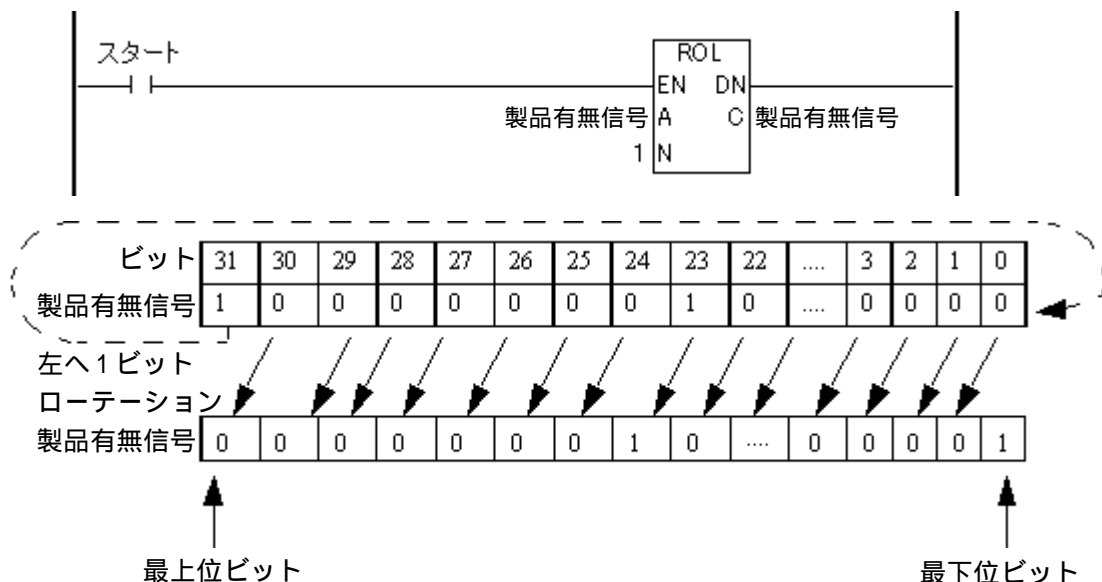


Nが範囲外の場合は、#OverflowがONになります。このとき、結果は不定です。

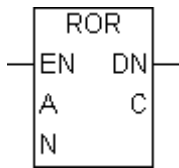
参照 3.2.21 #Overflow

動作例

以下の例では、製品有無信号を使用した1ビットローテーションについて説明しています。



4.2.17 ROR(右回転)



A: 回転させる変数名
 N: シフトビット数
 C: 格納先変数名

ROR 命令を実行すると、AのビットがNビット右方向にシフトされます。

右端のビット(最下位ビット)は、左端のビット(最上位ビット)にローテーションします。

結果はCに格納されます。

この命令は常に導通します。

ROR 命令が実行できるA、N、Cの組み合わせは以下の通りです。

Aのタイプ	Nのタイプ	Cのタイプ
整数	整数または整数定数	整数
整数配列	整数または整数定数	Aと同じサイズの配列
整数定数	整数または整数定数	整数

ROR 命令には2種類あります。

1. AとCが両方とも配列でないときは、単純に32ビットローテーションされます。Nは0以上31以下にしてください。
2. AとCが同じサイズの配列のときは、Aは大きなブロックとして扱われます。ビットは要素から次の要素へシフトされます。それぞれの要素内のみでローテーションするわけではありません。Nは0以上(32×配列サイズ-1)以下にしてください。

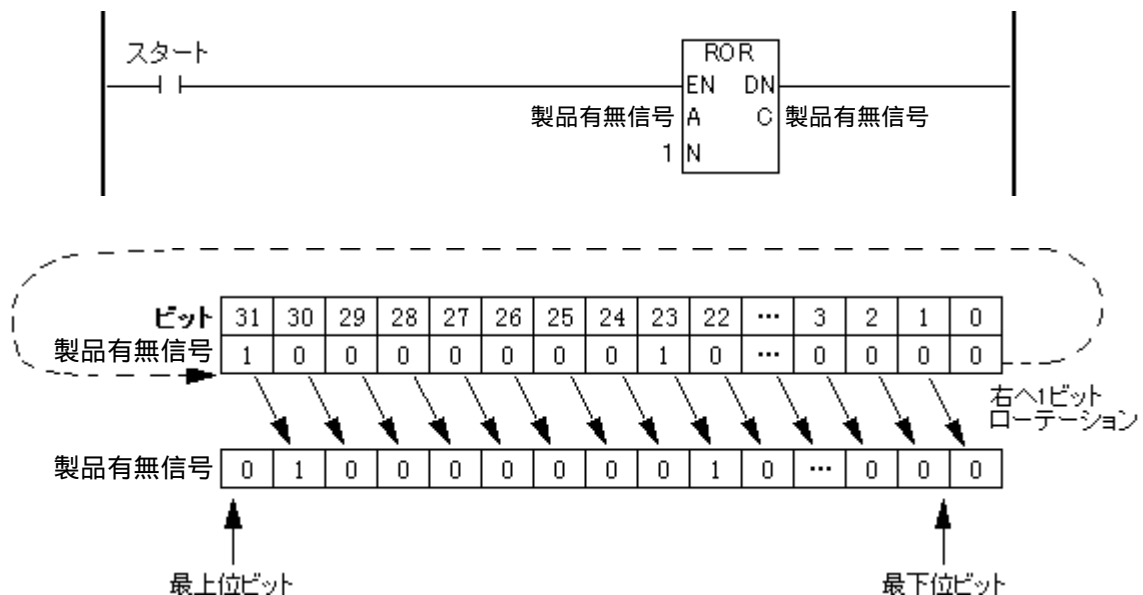


Nが範囲外の場合は、#OverflowがONになります。このとき、結果は不定です。

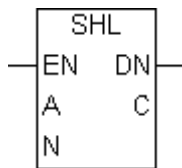
参照 3.2.21 #Overflow

動作例

以下の例では、製品有無信号を使用した1ビットローテーションについて説明しています。



4.2.18 SHL(左シフト)



A: 回転させる変数名
 N: シフトビット数
 C: 格納先変数名

SHL 命令を実行すると、A のビットが N ビット左方向にシフトします。

1 ビットシフトするたびに、左端のビット（最上位ビット）は失われ、右端の空ビットには 0 が格納されます。結果は C に格納されます。

この命令は常に導通します。

SHL 命令が実行できる A、N、C の組み合わせは以下の通りです。

A のタイプ	N のタイプ	C のタイプ
整数	整数または整数定数	整数
整数配列	整数または整数定数	A と同じサイズの配列
整数定数	整数または整数定数	整数

SHL 命令には 2 種類あります。

1. A と C が両方とも配列でないときは、単純に 32 ビットシフトします。N は 0 以上 31 以下にしてください。
2. A と C が同じサイズの配列のときは、A は大きな整数として扱われます。ビットは要素から次の要素へシフトされます。それぞれの要素の左端のビットは失われません。ただし、最後尾の要素の左端ビットは失われます。N は 0 以上 (32 × 配列サイズ - 1) 以下にしてください。



N が範囲外の場合は、#Overflow が ON になります。このとき、結果は不定です。

参照 3.2.21 #Overflow

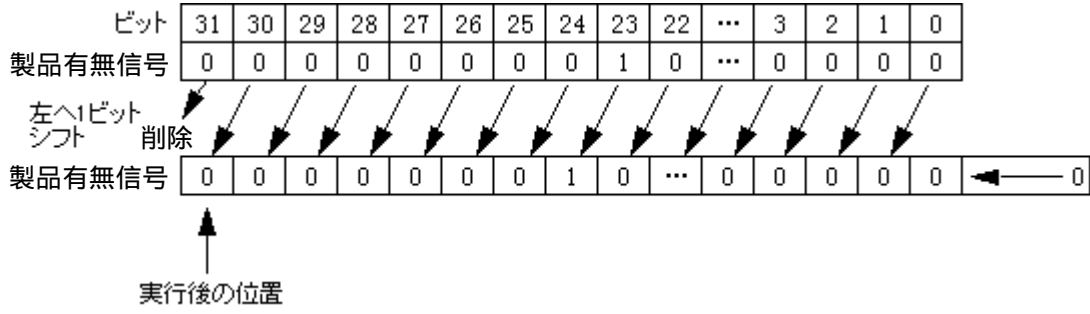
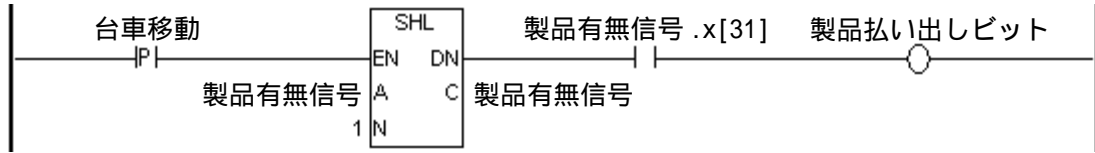
動作例

以下の例では、1 ビット左シフトを実行してビットの位置を把握する方法について説明しています。

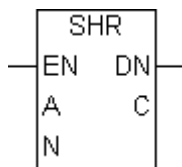
製品有無信号のそれぞれのビットは、実際の製品位置を表します。

台車移動が ON すると、ビットは次の位置へ左シフトします。

ビットが変数の最終ビット位置 (31) にくると、製品払い出しビットが ON になり、動作が終了したことがわかります。



4.2.19 SHR(右シフト)



A: 回転させる変数名
 N: シフトビット数
 C: 格納先変数名

SHR 命令を実行すると、AのビットがNビット右方向にシフトします。

1ビットシフトするたびに右端のビット(最下位ビット)は失われ、左端の空ビットには0が格納されます。結果はCに格納されます。

この命令は常に導通します。

SHR 命令が実行できるA、N、Cの組み合わせは以下の通りです。

Aのタイプ	Nのタイプ	Cのタイプ
整数	整数または整数定数	整数
整数配列	整数または整数定数	Aと同じサイズの配列
整数定数	整数または整数定数	整数

SHR 命令には2種類あります。

1. AとCが配列でないときは、単純に32ビットシフトします。Nは0以上31以下にしてください。
2. AとCが同じサイズの配列のときは、Aは大きな整数として扱われます。ビットは要素から次の要素へシフトされます。それぞれの要素の右端のビットは失われません。ただし、最初の要素の右端ビットは失われます。Nは0以上(32 × 配列サイズ - 1)以下にしてください。



Nが範囲外の場合は、#OverflowがONになります。このとき、結果は不定です。

参照 3.2.21 #Overflow

動作例

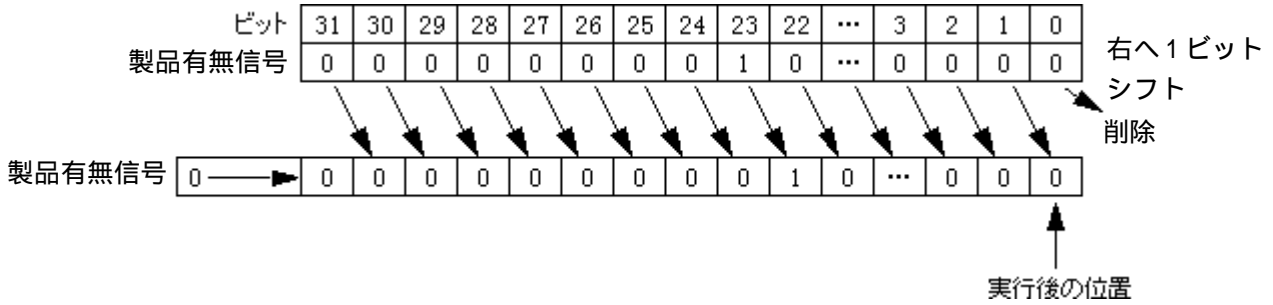
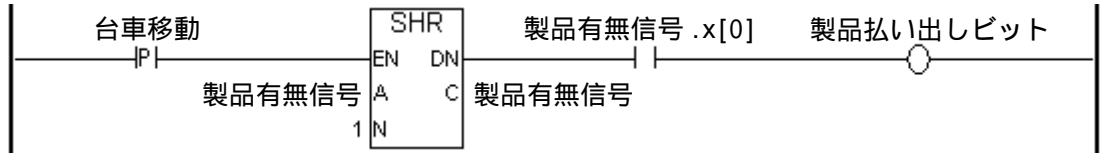
・ビットを扱う場合

1ビット右シフトを実行してビットの位置を把握する方法について説明しています。

製品有無信号のそれぞれのビットは、実際の製品位置を表します。

台車移動がONすると、ビットは次の位置へ右シフトします。

ビットが変数の最終ビット位置(0)にくると、製品払い出しビットがONになり、動作が終了したことがわかります。



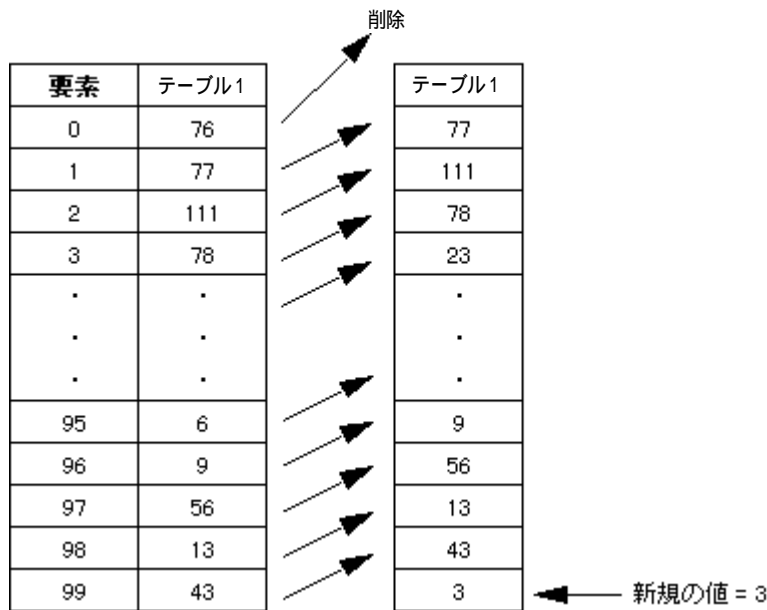
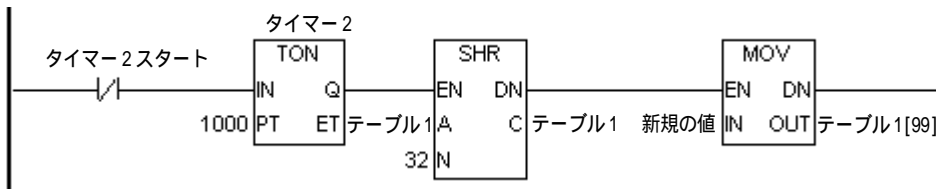
・配列を扱う場合

SHR命令を使用して整数配列の中で各要素の値を移動することについて説明しています。

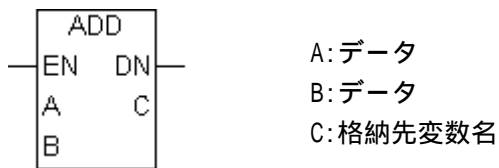
32ビットシフトでは、32ビット整数全体がシフトします。

1秒ごとに、整数配列テーブル1の整数値が要素0の方向に1つつつ移動します。

新規の値は、整数配列テーブル1の最後の要素（テーブル1[99]）に格納されます。



4.2.20 ADD(加算)



ADD 命令を実行すると、A と B が加算され、結果が C に格納されます。

A と B の両方が整数または整数定数のとき、ADD 命令は整数加算をします。

それ以外は、浮動小数点加算されますが、処理速度が遅くなる場合があります。

この命令は常に導通します。ADD 命令が実行できる A、B、C はそれぞれ以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数または実数
整数定数	整数定数	整数または実数
実数	実数	整数または実数
実数定数	実数定数	整数または実数



- ・ 結果CがCの変数タイプで表現できる範囲を超えるときは、#Overflow がONになり、加算の結果は不定です。

参照 3.2.21 #Overflow

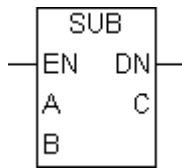
- ・ AまたはBのどちらかが実数のときは、AとBの両方を実数に変換して加算します。ただし、Cが整数のときは、結果はCに格納されるので、小数点以下は切り捨てられます。

動作例

スタートがONすると、データAとデータBが加算され、計算結果がデータCに格納されます。



4.2.21 SUB(減算)



A: データ
B: データ
C: 格納先変数名

SUB 命令を実行すると、A から B が減算され、結果が C に格納されます。

A と B のタイプが整数または整数定数のときは、SUB 命令は整数減算をします。

それ以外は、浮動小数点減算されますが、処理速度が遅くなる場合があります。

この命令は常に導通します。SUB 命令が実行できる A、B、C はそれぞれ以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数または実数
整数定数	整数定数	整数または実数
実数	実数	整数または実数
実数定数	実数定数	整数または実数



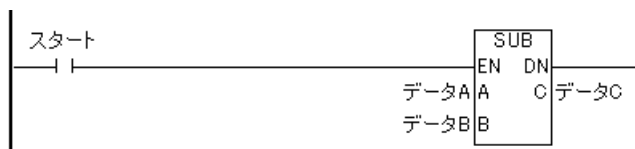
- 結果CがCの変数タイプで表現できる範囲を超えるときは、#Overflow が ON になり、減算の結果は不定です。

参照 3.2.21 #Overflow

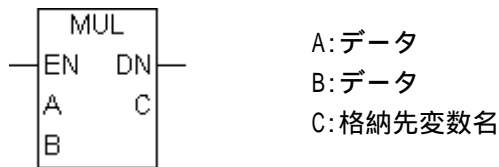
- A または B のどちらかが実数のときは、A と B の両方を実数に変換して減算します。ただし、C が整数のときは、結果は C に格納されるので、小数点以下は切り捨てられます。

動作例

スタートが ON すると、データ A からデータ B が減算され、計算結果がデータ C に格納されます。



4.2.22 MUL (乗算)



MUL 命令を実行すると、A と B が乗算され、結果が C に格納されます。

A と B の両方が整数または整数定数のとき、整数乗算されます。

それ以外は、浮動小数点乗算されますが、処理速度が遅くなる場合があります。

この命令は常に導通します。MUL 命令が実行できる A、B、C はそれぞれ以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数または実数
整数定数	整数定数	整数または実数
実数	実数	整数または実数
実数定数	実数定数	整数または実数



- 結果CがCの変数タイプで表現できる範囲を超えるときは、#Overflow がONになり、乗算の結果は不定です。

参照 3.2.21 #Overflow

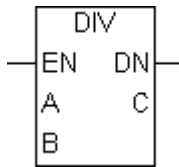
- AまたはBのどちらかが実数のときは、AとBの両方を実数に変換して乗算します。ただし、Cが整数のときは、結果はCに格納されるので、小数点以下は切り捨てられます。

動作例

スタートがONすると、データAとデータBが乗算され、計算結果がデータCに格納されます。



4.2.23 DIV(除算)



A: データ
 B: データ
 C: 格納先変数名

DIV 命令を実行すると、AがBで除算され、結果はCに格納されます。

AとBの両方が整数または整数定数のときは、整数除算されます。

それ以外は、浮動小数点除算されますが、処理速度が遅くなる場合があります。

この命令は常に導通します。DIV 命令が実行できるA、B、Cはそれぞれ以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数	整数	整数または実数
整数定数	整数定数	整数または実数
実数	実数	整数または実数
実数定数	実数定数	整数または実数



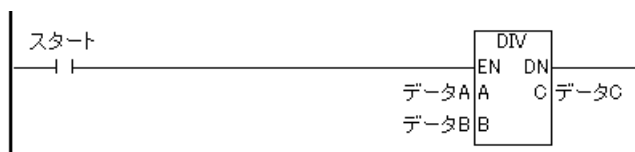
- ・ Bが0のときや、結果CがCの変数タイプで表現できる範囲を超えるときは、#OverflowがONになり、除算の結果は不定です。

参照 3.2.21 #Overflow

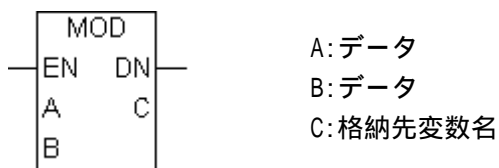
- ・ AまたはBのどちらかが実数のときは、AとBの両方を実数に変換して除算します。ただし、Cが整数のときは、結果はCに格納されるので、小数点以下は切り捨てられます。

動作例

スタートがONすると、データAがデータBで除算され、計算結果がデータCに格納されます。



4.2.24 MOD(剰余算)



MOD 命令を実行すると、A が B で除算され、余りが C に格納されます。A、B が整数または整数定数のときだけ実行されます。

この命令は常に導通します。MOD 命令が実行できる A、B、C は以下の通りです。

Aのタイプ	Bのタイプ	Cのタイプ
整数定数	整数	整数
整数	整数定数	整数



0 で除算したときは、#Overflow が ON になり、剰余算の結果は不定です。

参照 3.2.21 #Overflow

動作例

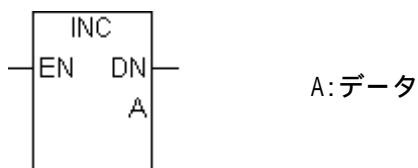
スタートが ON すると、データ A がデータ B で除算され、余りがデータ C に格納されます。



以下の例では、整数 27 が 5 で除算され、結果 2 が C に格納されます。

$$\begin{array}{r}
 A=27 \\
 B=5 \\
 \hline
 5 \overline{) 27} \\
 \underline{25} \\
 2 \leftarrow C=2
 \end{array}$$

4.2.25 INC(インクリメント)



INC 命令を実行すると、Aに1を加えます。この命令は常に導通します。

INC 命令が実行できるAは以下の通りです。

Aのタイプ
整数

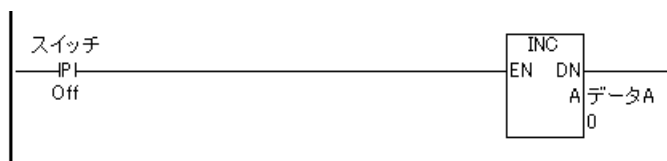


Aが0x7FFFFFFFのとき0x80000000となり、#Overflowがセットされます。

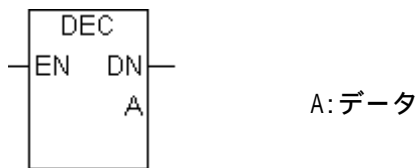
参照 3.2.21 #Overflow

動作例

スイッチがONすると、データAに1加算されます。



4.2.26 DEC(デクリメント)



DEC 命令を実行すると、A から 1 を引きます。この命令は常に導通します。

DEC 命令が実行できる A は以下の通りです。

Aのタイプ
整数



A が 0x80000000 のとき 0x7FFFFFFF となり、#Overflow がセットされます。

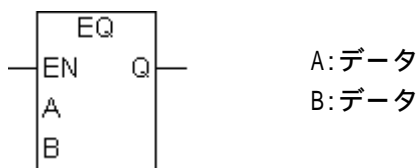
参照 3.2.21 #Overflow

動作例

スイッチが ON すると、データ A から 1 減算されます。



4.2.27 EQ(比較 :=)



この命令は $A = B$ のときに導通します。EQ 命令が実行できる A、B はそれぞれ以下の通りです。

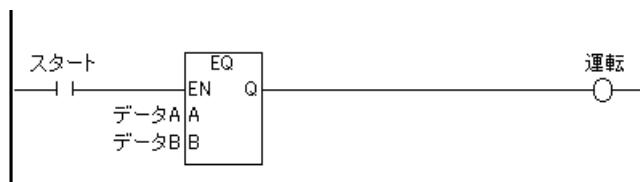
Aのタイプ	Bのタイプ
整数	整数
整数定数	整数定数
実数	実数
実数定数	実数定数



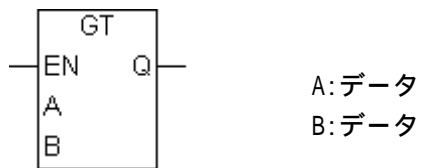
実数値を比較する際、注意が必要です。例えば、演算結果が 1.9999999999999999 になることがあります。これは 2.0000000000000000 と等しくありません。

動作例

スタートが ON で、データ A とデータ B の数値が等しいときに運転します。



4.2.28 GT(比較 : >)



この命令は $A > B$ のときに導通します。GT 命令が実行できる A、B はそれぞれ以下の通りです。

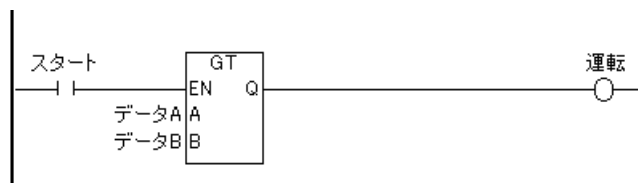
Aのタイプ	Bのタイプ
整数	整数
整数定数	整数定数
実数	実数
実数定数	実数定数



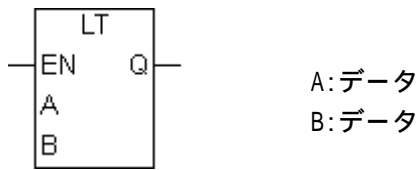
実数値を比較する際、注意が必要です。例えば、計算結果が 2.000000000001 になることがありますが、これは 2 より大きいです。

動作例

スタートが ON で、データ A がデータ B の数値より大きいときに運転します。



4.2.29 LT(比較 : <)



この命令は $A < B$ のときに導通します。LT 命令が実行できる A、B はそれぞれ以下の通りです。

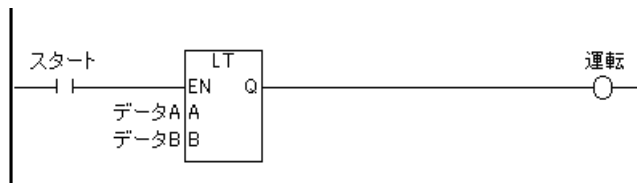
Aのタイプ	Bのタイプ
整数	整数
整数定数	整数定数
実数	実数
実数定数	実数定数



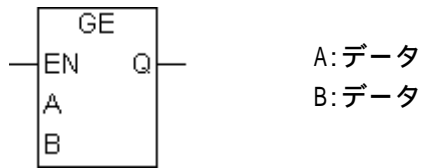
実数値を比較する際、注意が必要です。例えば、計算結果が 1.99999999999 になることがあります。これは 2 より小さいです。

動作例

スタートが ON で、データ A がデータ B の数値がより小さいときに運転します。



4.2.30 GE(比較 : >=)



この命令はA ≥ B のときに導通します。GE 命令が実行できる A、B はそれぞれ以下の通りです。

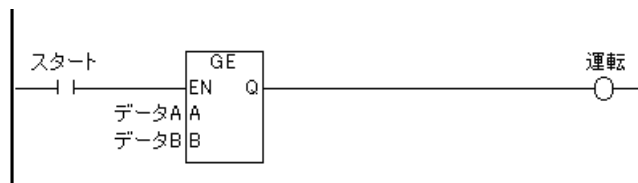
Aのタイプ	Bのタイプ
整数	整数
整数定数	整数定数
実数	実数
実数定数	実数定数



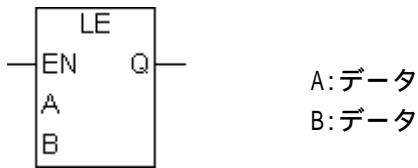
実数値を比較する際、注意が必要です。例えば、計算結果が 1.9999999999 になることがあります。これは 2 以上ではありません。

動作例

スタートが ON で、データ A がデータ B の数値以上のときに運転します。



4.2.31 LE(比較 : <=)



この命令はA ≤ Bのときに導通します。LE命令が実行できるA、Bはそれぞれ以下の通りです。

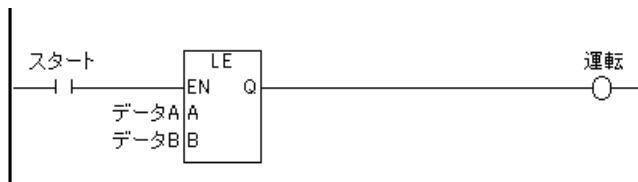
Aのタイプ	Bのタイプ
整数	整数
整数定数	整数定数
実数	実数
実数定数	実数定数



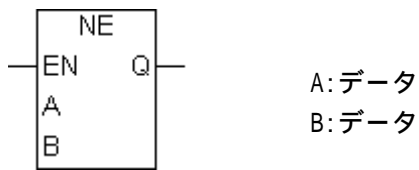
実数値を比較する際、注意が必要です。例えば、計算結果が2.000000000001になることがありますが、これは2以下ではありません。

動作例

スタートがONで、データAがデータBの数値以下のときに運転します。



4.2.32 NE(比較：<>)



この命令はA Bのときに導通します。NE命令が実行できるA、Bはそれぞれ以下の通りです。

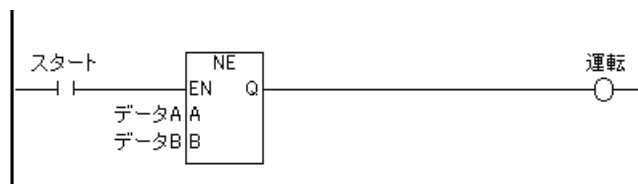
Aのタイプ	Bのタイプ
整数	整数
整数定数	整数定数
実数	実数
実数定数	実数定数



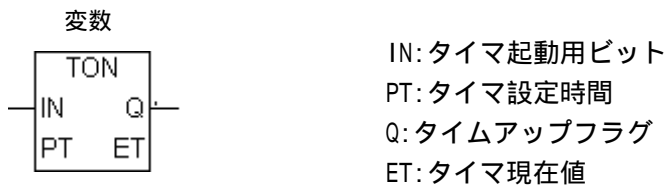
実数値を比較する際、注意が必要です。例えば、計算結果が1.9999999999になることがあります。これは2と等しくありません。

動作例

スタートがONで、データAとデータBの数値が等しくないときに運転します。



4.2.33 TON(オンディレータイマ)



タイマ入力ビット IN が導通してから、設定した時間 PT(ms 単位) 経過後、タイマ出力ビット Q が ON になります。

動作概要

専用変数	内容	変数タイプ
変数.PT	設定値	整数
変数.ET	現在値	整数
変数.Q	タイマ出力ビット	ディスクリート
変数.TI	タイマ計測ビット	ディスクリート

タイマ起動用ビット IN が導通すると、TON 命令に起動がかかるために

- ・経過時間変数 .ET が ms 単位で増加します。
- ・タイマ計測ビット変数 .TI が ON になります。
- ・タイマ出力ビット変数 .Q が OFF になります。

経過時間変数 .ET が増加して、設定時間変数 .PT に等しくなると

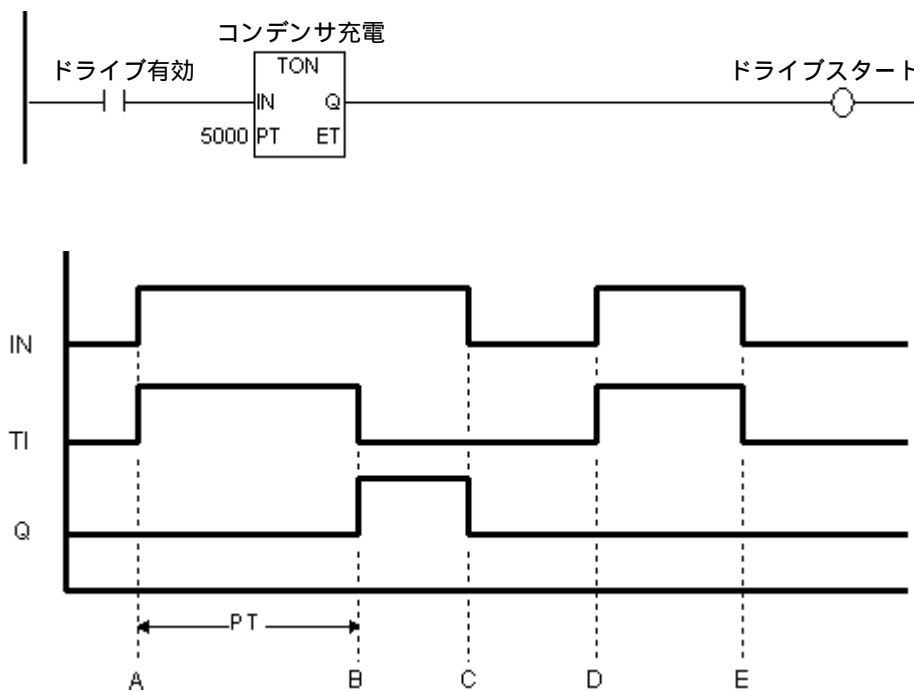
- ・経過時間変数 .ET は現在値を保持します。
- ・タイマ計測ビット変数 .TI が OFF になります。
- ・タイマ出力ビット変数 .Q が ON になり、導通します。

TON 命令に起動をかけるタイマ起動用ビット IN の導通をやめると

- ・経過時間変数 .ET は 0 にリセットされます。
- ・タイマ計測ビット変数 .TI が OFF になります。
- ・タイマ出力ビット変数 .Q が OFF になります。

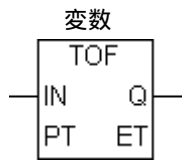
動作例

以下の例では、ドライブ有効をONにしてから5秒後にドライブをスタートさせます。



- A タイマ入力ビット IN がONになり、タイマ計測ビット TI がONになります。タイマの計測が始まり、経過時間 ET が増加します。タイマ出力ビット Q はOFFのままです。
- B 経過時間 ET が設定時間 PT に等しくなると、タイマ出力ビット Q がONになります。経過時間 ET は、設定時間 PT の値のままです。タイマ計測ビット TI はOFFになります。
- C タイマ入力ビット IN がOFFになり、タイマ出力ビット Q はOFFになります。経過時間 ET が0にリセットされます。
- D タイマ入力ビット IN がONになり、タイマ計測ビット TI がONになります。タイマの計測が始まり、経過時間 ET が増加します。
- E 経過時間 ET が設定時間 PT に達する前にタイマ入力ビット IN がOFFになり、タイマ出力ビットはOFFのまま、経過時間 ET が0になります。経過時間 ET は0にリセットされます。

4.2.34 TOF(オフディレータイマ)



IN: タイマ起動用ビット
 PT: タイマ設定時間
 Q: タイムアップフラグ
 ET: タイマ現在値

タイマ入力ビット IN の導通が止まってから、設定した時間 PT (ms 単位) 経過後、タイマ出力ビット Q が OFF になります。

動作概要

専用変数	内容	変数タイプ
変数.PT	設定値	整数
変数.ET	現在値	整数
変数.Q	タイマ出力ビット	ディスクリート
変数.TI	タイマ計測ビット	ディスクリート

タイマ起動用ビット IN が導通すると、TOF 命令に起動がかかるために

- ・経過時間変数 .ET が 0 にリセットされます。
- ・タイマ計測ビット変数 .TI が OFF になります。
- ・タイマ出力ビット変数 .Q が ON になり、導通します。

TOF 命令に起動をかけるタイマ起動用ビット IN の導通をやめると

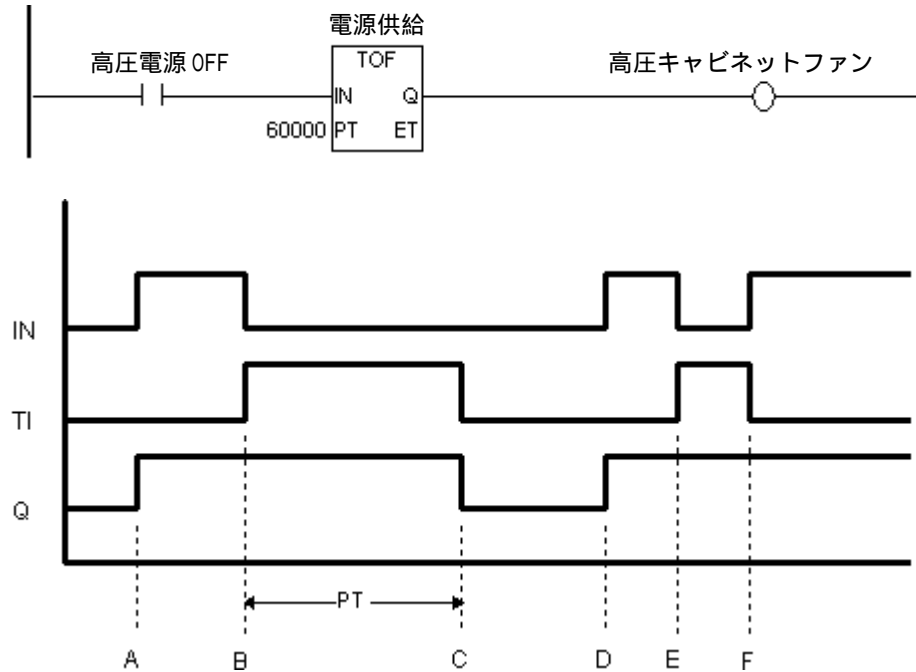
- ・経過時間変数 .ET が ms 単位で増加します。
- ・タイマ計測ビット変数 .TI が ON になります。
- ・タイマ出力ビット変数 .Q は ON のままです。

経過時間変数 .ET が増加して、設定時間変数 .PT に等しくなると

- ・経過時間変数 .ET は、設定値を保持します。
- ・タイマ計測ビット変数 .TI が OFF になります。
- ・タイマ出力ビット変数 .Q が OFF になります。

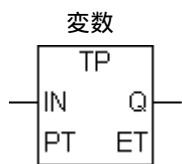
動作例

以下の例では、高圧電源を OFF にしても、ON の間と同じように高圧キャビネットのファンを 1 分間 (60,000ms) 動作させています。



- A タイマ入力ビット IN が ON になります。タイマ計測ビット TI は OFF のままです。タイマ出力ビット Q が ON になります。経過時間 ET が 0 にリセットされます。
- B タイマ入力ビット IN が OFF になります。タイマが計測を開始します (TI は ON になります)。タイマ出力ビットは ON のままです。
- C まず、経過時間 ET が設定時間 PT に等しくなります。タイマ出力ビット Q が OFF になります。タイマは計測を終了します (TI は OFF になります)。経過時間 ET は設定時間と等しいままです (ET=PT)。
- D タイマ入力ビット IN が ON になります。タイマ計測ビット TI は OFF のままです。タイマ出力ビット Q は ON になります。経過時間 ET は 0 にリセットされます。
- E タイマ入力ビット IN が OFF になります。タイマが計測を開始します (TI は ON になります)。タイマ出力ビット Q は ON のままです。
- F 経過時間 ET が設定時間 PT に達する前に、タイマ入力ビット IN が ON になり、タイマは計測を停止します。(TI は OFF になります。)タイマ出力ビット Q は ON のままで経過時間 ET は 0 にリセットされます。

4.2.35 TP(パルスタイマ)



IN: タイマ起動用ビット
 PT: タイマ設定時間
 Q: タイムアップフラグ
 ET: タイマ現在値

1回のタイマ入力ビット IN への導通で設定時間 PT(ms 単位)の間タイマ出力ビット Q が ON になります。

動作概要

専用変数	内容	変数タイプ
変数.PT	設定値	整数
変数.ET	現在値	整数
変数.Q	タイマ出力ビット	ディスクリート
変数.TI	タイマ計測ビット	ディスクリート

タイマ起動用ビット IN が導通すると、TP 命令に起動がかかるために

- ・経過時間変数 .ET が ms 単位で増加します。
- ・タイマ計測ビット変数 .TI が ON になります。
- ・タイマ出力ビット変数 .Q が ON になり、導通します。

経過時間変数 .ET が増加して、設定時間変数 .PT に等しくなると

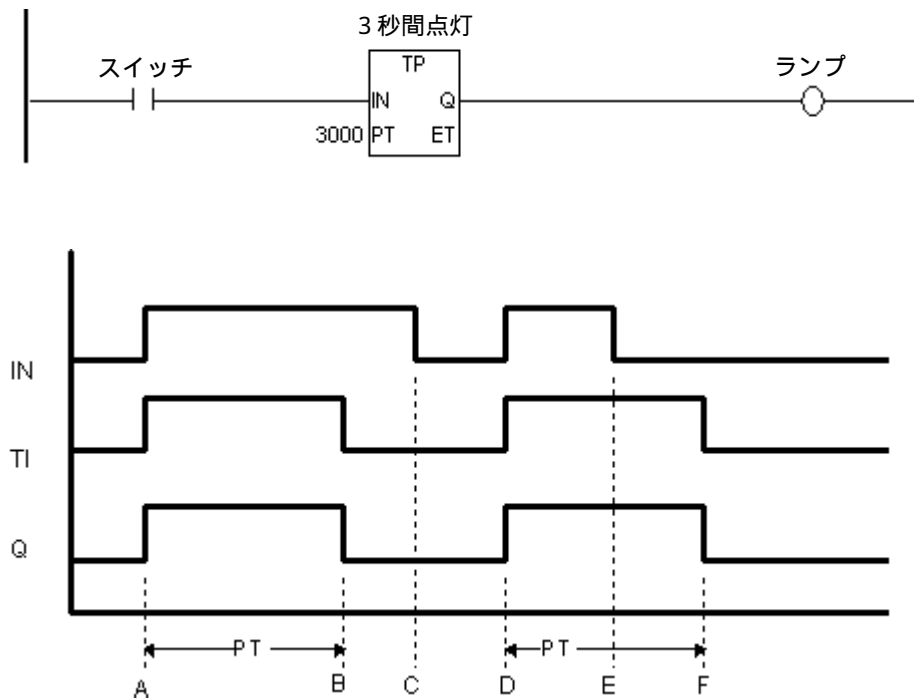
- ・TP 命令に導通しているときは、経過時間変数 .ET は設定値を保持します。
- ・導通していないときは、すぐに 0 にリセットされます。
- ・タイマ計測ビット変数 .TI が OFF になります。
- ・タイマ出力ビット変数 .Q が OFF になり、導通を終了します。

TP 命令に起動をかけるタイマ起動用ビット IN の導通をやめると

経過時間変数 .ET が設定時間変数 .PT に達しているときは、経過時間変数 .ET が 0 にリセットされ、タイマ出力ビット Q は OFF になります。それ以外のときは、計測を続け、タイマ出力ビット変数 .Q は ON のままです。

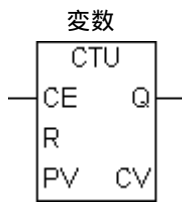
動作例

以下の例では、スイッチを押すと、3秒間だけランプが点灯します。



- A タイマ入力ビット IN が ON になります。タイマが計測を開始します (TI が ON になります)。タイマ出力ビット Q が ON になります。
- B まず、経過時間 ET が設定時間 PT に等しくなります。タイマ出力ビット Q が OFF になります。タイマが計測を終了します (TI が OFF になります)。経過時間 ET は設定時間の値のままです (ET=PT)。
- C タイマ入力ビット IN が OFF になります。経過時間 ET は 0 にリセットされます。
- D タイマ入力ビット IN は ON になります。タイマが計測を開始します (TI が ON になります)。タイマ出力ビット Q が ON になります。
- E タイマ入力ビット IN は OFF になります。タイマは計測を続けます (TI は ON のままです)。タイマ出力ビット Q は ON のままです。
- F まず、経過時間 ET が設定時間 PT に等しくなります。タイマ出力ビット Q が OFF になります。タイマが計測を終了します (TI は OFF になります)。タイマ入力ビット IN が OFF なので経過時間 ET が 0 にリセットされます。

4.2.36 CTU(アップカウンタ)



CE: カウンタ起動用ビット
 R: カウンタリセットビット
 PV: カウンタ設定値
 Q: カウンタ出力
 CV: カウンタ現在値

動作概要

専用変数	内容	変数タイプ
変数.PV	設定値	整数
変数.CV	現在値	整数
変数.R	カウンタリセット	ディスクリート
変数.UP	アップカウンタ	ディスクリート
変数.QU	アップカウンタ出力	ディスクリート
変数.QD	ダウンカウンタ出力	ディスクリート
変数.Q	カウンタ出力	ディスクリート

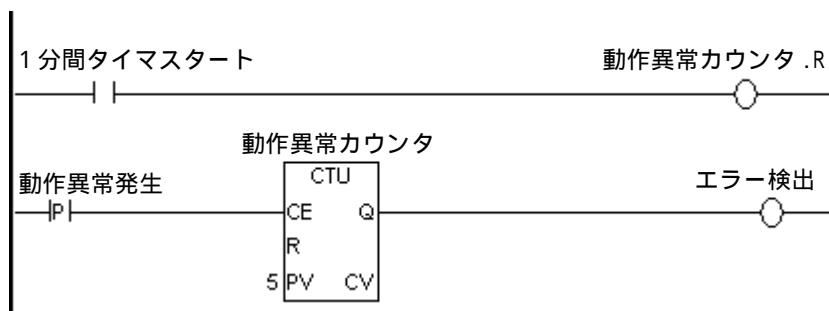
カウンタ起動用ビットCEが導通すると、カウンタリセットビット変数.RがOFFで、現在値変数.CVが設定値変数.PVより小さいときに、現在値変数.CVが1加算されます。

現在値変数.CVが設定値変数.PVと等しくなると、カウンタ出力ビット変数.QがONになり、導通します。

カウンタリセットビット変数.RがONのときは、現在値変数.CVが0にリセットされます。カウンタ出力ビット変数.QもOFFします。

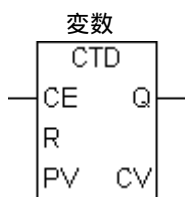
動作例

以下の例では、1分間に動作異常を5つカウントすると、エラーを知らせます。



カウンタはスキャン毎に更新されます。例のようなイベントをカウントするためには、CTU命令の前にPT命令を挿入してください。CTU命令はレベル入力です。

4.2.37 CTD(ダウンカウンタ)



CE:カウンタ起動用ビット
R:カウンタリセットビット
PV:カウンタ設定値
Q:カウンタ出力
CV:カウンタ現在値

動作概要

専用変数	内容	変数タイプ
変数.PV	設定値	整数
変数.CV	現在値	整数
変数.R	カウンタリセット	ディスクリート
変数.UP	アップカウンタ	ディスクリート
変数.QU	アップカウンタ出力	ディスクリート
変数.QD	ダウンカウンタ出力	ディスクリート
変数.Q	カウンタ出力	ディスクリート

カウンタ起動用ビットCEが導通すると、カウンタリセットビット変数.RがOFFのときに、現在値変数.CVが1減算されます。

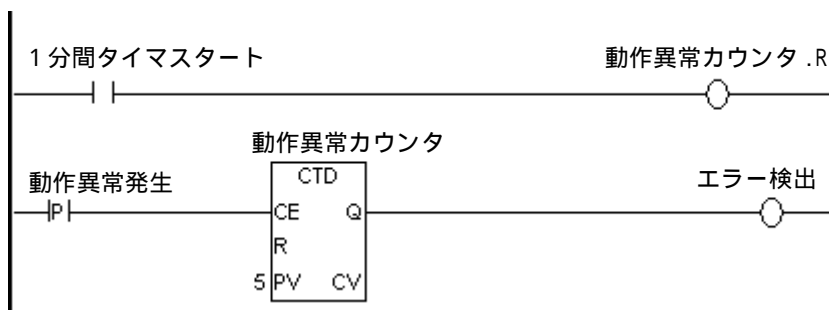
現在値変数.CVが1減算され0以下になると、カウンタ出力ビット変数.QはONになり、導通します。

カウンタリセットビット変数.RがONのときは、設定値変数.PVが現在値変数.CVにセットされます。カウンタ出力ビット変数.QもOFFします。

動作例

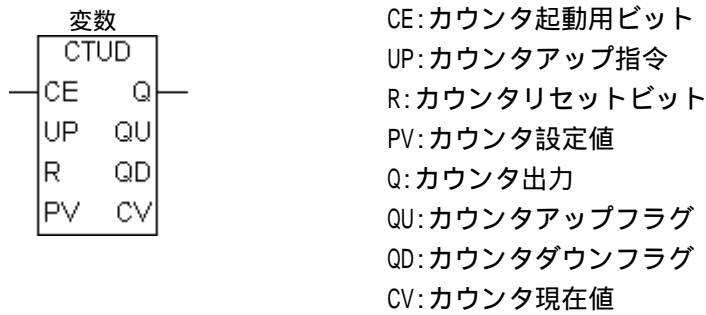
以下の例では、1分間に動作異常を5つカウントすると、エラーを知らせます。

タイマは、カウンタを1分ごとにリセットします。



カウンタはスキャン毎に更新されます。例のようなイベントをカウントするためには、CTD命令の前にPT命令を挿入してください。

4.2.38 CTUD(アップダウンカウンタ)



動作概要

専用変数	内容	変数タイプ
変数.PV	設定値	整数
変数.CV	現在値	整数
変数.R	カウンタリセット	ディスクリート
変数.UP	アップカウンタ	ディスクリート
変数.QU	アップカウンタ出力	ディスクリート
変数.QD	ダウンカウンタ出力	ディスクリート
変数.Q	カウンタ出力	ディスクリート

CTUD 命令は、カウンタアップ指令変数 .UP が ON のときは、CTU 命令(アップカウンタ)を実行するときと同じになります。変数 .UP が OFF のときは、CTD 命令(ダウンカウンタ)を実行するときと同じになります。

実行後に現在値変数 .CV が設定値変数 .PV 以上になったときは、変数 .Q と変数 .QU が ON になります。

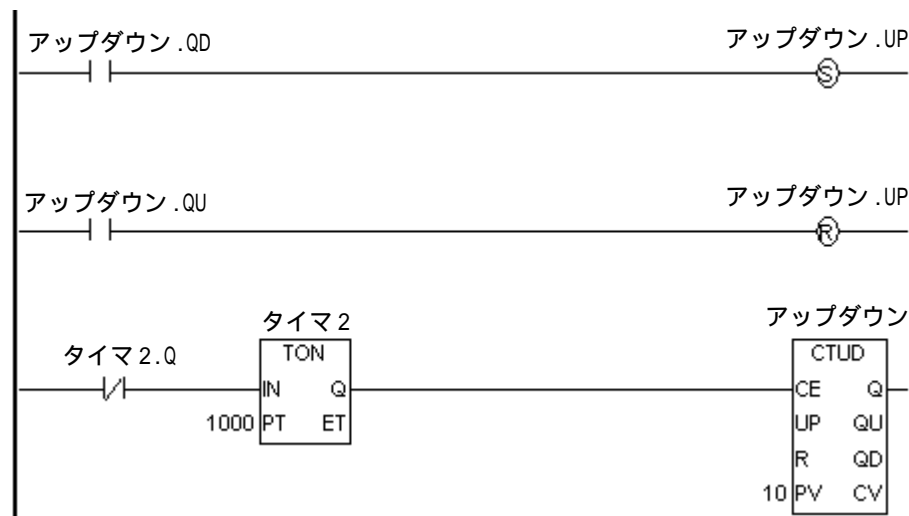
実行後に現在値変数 .CV が 0 以下になったときは、変数 .Q と変数 .QD が ON になります。

動作例

以下の例では、CTUD 命令を実行して 0 から 10 まで繰り返しカウントされています。10 までカウントすると、0 に戻ります。

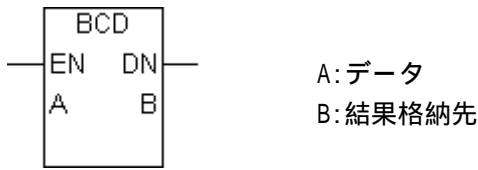
タイマ 2 は、アップダウンカウンタにパルスを毎秒出力します。

カウンタアップダウンが 0 になると UP ビットが ON になり、カウンタアップダウンが 10 (プリセット値) になると UP ビットが OFF になります。



カウンタアップ指令変数 .UP が ON のときにカウンタリセットビット変数 .R が ON になると、現在値変数 .CV が 0 になります。カウンタアップ指令変数 .UP が OFF のときにカウンタリセットビット変数 .R が ON になると、現在値変数 .CV には設定値変数 .PV が入ります。

4.2.39 BCD(BCD 変換)



BCD 命令を実行すると、2進数 のAを2進化10進数 に変換し、結果をBに格納します。
この命令は、エラーが発生すると導通しません。BCD 命令が実行できるA、Bはそれぞれ以下の通りです。

Aのタイプ	Bのタイプ
整数	整数
整数定数	

変換可能なAの最大値は0x5F5E0FFです。Aが大きすぎるとき、#FaultCodeはエラーコードで更新され、#Overflowがセットされます。

参照 3.2.18 #FaultCode、参照 3.2.21 #Overflow



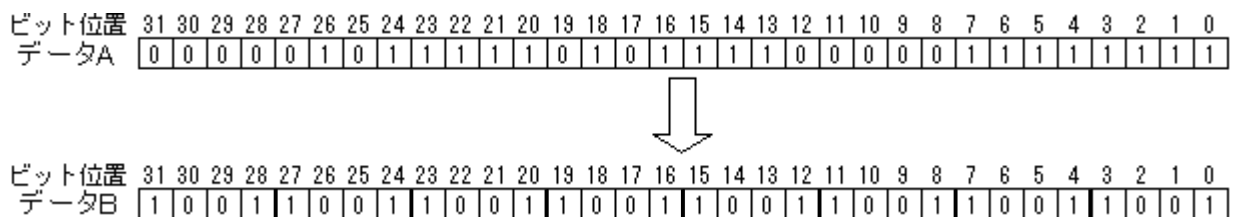
変換できない値を変換しようとした場合、Bの値は不定となります。

動作例

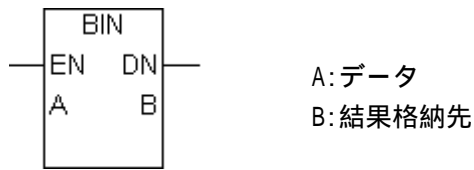
スタートがONすると、データAをBCD変換し、データBに格納します。



例)データAにBINデータ"99999999"を設定し、BCD変換した場合



4.2.40 BIN(バイナリ変換)



BIN 命令を実行すると、2進数 10進数 のAを2進数 に変換し、結果をBに格納します。

この命令は、エラーが発生すると導通しません。BIN 命令が実行できるA、Bはそれぞれ以下の通りです。

Aのタイプ	Bのタイプ
整数	整数
整数定数	

Aが有効な2進数 10進数でないとき、#FaultCode はエラーコードで更新され、#Overflow がセットされます。

参照 3.2.18 #FaultCode、3.2.21 #Overflow



変換できない値を変換しようとした場合、Bの値は不定となります。

動作例

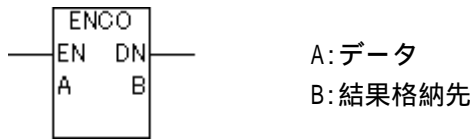
スタートがONすると、データAをBIN変換し、データBに格納します。



例)データAにBCDデータ"99999999"を設定し、BIN変換した場合

ビット位置	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
データA	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
	↓																															
データB	0	0	0	0	0	1	0	1	1	1	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1

4.2.41 ENCO(エンコード)

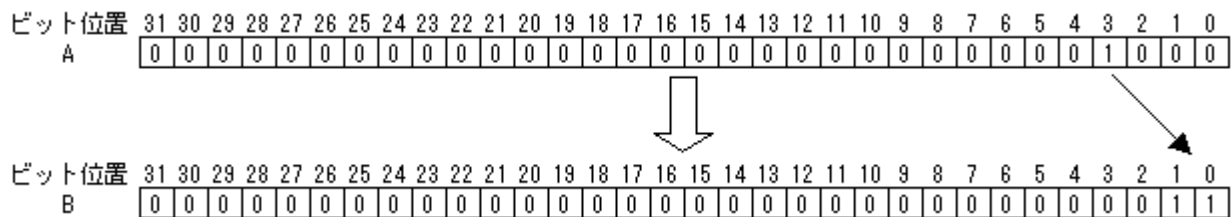


Aに入力された値をエンコードしてBに出力します。Aの32ビットの内、ONしているビット位置を読みとり、2進数の値としてBに出力します。Aに複数ビットがONしている場所は最上位ビット位置を出力します。

この命令は常に導通します。ENCO命令に有効な変数タイプの組み合わせは以下の通りです。

Aのタイプ	Bのタイプ
整数	整数
整数配列	Aと同じサイズの整数配列
整数定数	整数

例：Aに0x00000008を入力した場合、出力Bは0x00000003になります。



- ・入力Aに0が入っているとマイナー異常 #Overflow にし、#FaultCode にエラーコード "13" をセットします。参照 3.2.21 #Overflow
- ・変数の修飾語は対応しません。(ビット指定、ワード指定、バイト指定)
- ・ENCO 命令は GLC2000 シリーズのみ対応です。

4.2.42 DECO(デコード)

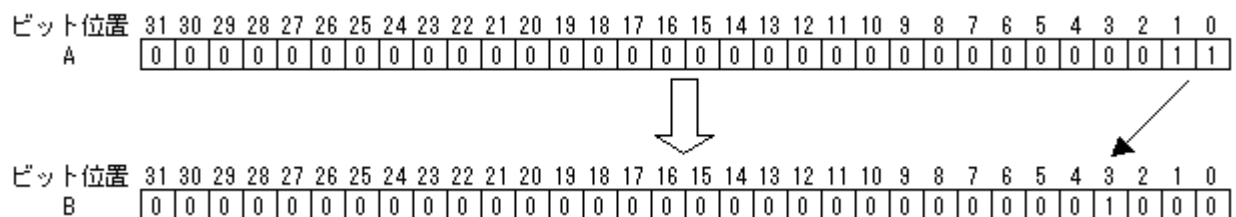


Aに入力された値をデコードしてBに出力します。Aを2進数の値として読みとり、Bの対応するビット位置を立てます。0～31までの入力のみが有効となります。

この命令は常に導通します。DECO命令に有効な変数タイプの組み合わせは以下の通りです。

Aのタイプ	Bのタイプ
整数	整数
整数配列	Aと同じサイズの整数配列
整数定数	整数

例：Aに0x00000003を入力した場合、出力Bは0x00000008になります。



- ・入力Aに0～31以外の値が入力されるとマイナー異常#OverFlowをONにし、#FaultCodeにエラーコード“13”をセットします。参照 3.2.21 #Overflow
- ・変数の修飾語は対応しません。(ビット指定、ワード指定、バイト指定)
- ・DECO命令はGLC2000シリーズのみ対応です。

4.2.43 JMP(ジャンプ)

—>>LabelName

JMP 命令に導通すると、指定したラベルへジャンプします。

JSR 命令と違い、ジャンプ元のラングへは自動的に戻りません。

START、SUB START、SUB END を越えてジャンプすることはできません。

上方向にジャンプすると、無限ループになることがあります。

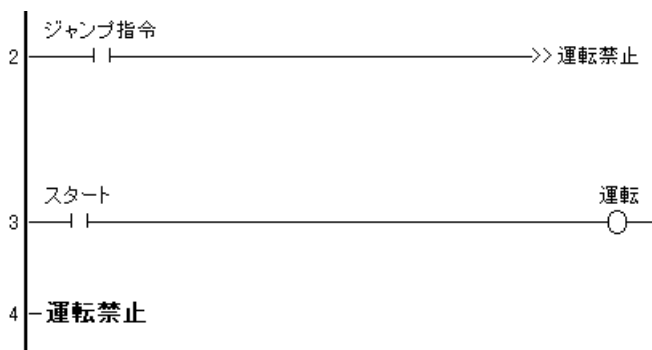


プログラム全体に要する時間がウォッチドッグタイマの値を超えないようにしてください。

参照 3.2.17 #WatchdogTime

動作例

ジャンプ指令が ON していると、ラベル名「運転禁止」のラング 4 へジャンプし、ラング 4 以降を実行します。ラング 3 の命令は実行されません。



4.2.44 JSR(ジャンプサブルーチン)

—>> SubroutineName<<

JSR 命令に導通すると、指定したサブルーチンへジャンプします。

サブルーチンが終了すると、ジャンプ元の JSR 命令へ戻り、次の命令の実行が続けられます。サブルーチン名は重複して設定できません。

JSR 命令は、ラングの最後に置いてください。

制限事項

- ・サブルーチンからのジャンプサブルーチンは、最高 128 回設定可能です。
- ・1回のジャンプでスタックを「1」使用します。ロジックプログラムで使用できるスタック数は合計で 128 です。JSR 命令以外にスタックを使用する命令は FOR ~ NEXT 命令があり、FOR ~ NEXT 命令 1 回で使用するスタックは「2」です。参照 4.2.46 FOR、NEXT(繰り返し)



プログラム全体に要する時間がウォッチドッグタイマの値を超えないようにしてください。

参照 3.2.17 #WatchdogTime

コントローラシステムのエラーを処理するサブルーチン

ErrorHandler というサブルーチンを作成しておくことで、演算エラーやマイナー異常が発生したときに ErrorHandler サブルーチンに自動的にジャンプします。

参照 3.2.21 #Overflow

4.2.45 RET(リターンサブルーチン)

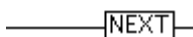
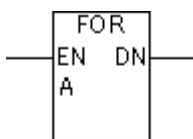
—<RETURN>—

RET 命令に導通すると、サブルーチンから強制的に呼び出し元の JSR 命令に戻り、次のラングから命令の実行が続けられます。

サブルーチンが終了すると、SUB END 命令により自動的に呼び出し元に戻るため、RET 命令は必ずしも使用する必要はありません。

RET 命令は、ラングの最後に置いてください。

4.2.46 FOR、NEXT(繰り返し)



Aで指定した回数、FORとNEXTの間のロジックプログラムを繰り返し実行します。FOR～NEXT命令間の処理を無条件にA回実行すると、NEXT命令の次ステップの処理を行います。

Aが0以下の場合、FORとNEXTの間のロジックプログラムは実行せず、NEXT命令の次ステップの処理にジャンプします。

この命令は、常に導通します。FOR～NEXT命令が有効な変数タイプは以下の通りです。

Aのタイプ
整数
整数配列
整数定数

制限事項

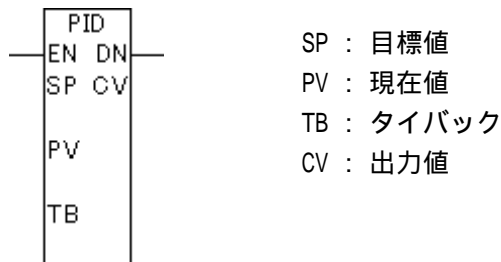
- 必ず、FOR命令に対応したNEXT命令を入れてください。
- 同一ラング上で、FOR～NEXT命令の前後に命令を入れしないでください。
- ネスティングは最高64回入れることができます。64回を超えるとメジャー異常になり # FaultCode にエラーコード4が表示されます。
- 1回のネスティングでスタックを「2」使用します。ロジックプログラムで使用できるスタック数は合計で128です。FOR～NEXT命令以外にスタックを使用する命令はJSR命令があり、使用するスタックは「1」です。参照 4.2.44 JSR命令



- エディタのエラーチェックで表示されるエラー、警告については「Pro-Control Editorオペレーションマニュアル 付録A:エラーと警告」を参照してください。
- # FaultCodeのエラーコードについては 3.2.18 # FaultCode を参照してください。
- ネスティングの回数を指定する時は、プログラム全体に要する時間がウォッチドッグタイマの値を超えないようにしてください。
参照 3.2.17 #WatchdogTime
- FOR～NEXT命令はGLC2000シリーズのみ対応です。

4.2.47 PID(PID演算)

コントロールブロック変数



PID命令は、アナログ入力や温度入力からの測定値（現在値）と、あらかじめ設定された値（目標値）を比較して、現在値と目標値の差をなくすように出力値を調節します。

PID制御を行う場合、P制御、I制御、D制御をそれぞれ自由に組み合わせて制御することができます。後述するそれぞれのパラメータを設定することで、これらの制御を実行します。

PID制御で算出される出力値は、原理的に次式で表されます。

$$CV = KC \left(E + \text{Reset} \int_0^t (E) dt + \text{Rate} \frac{d(E)}{dt} \right)$$

KC : 比例係数
E : 偏差 (SP-PV または PV-SP)
Reset : 積分時間
Rate : 微分時間

後述する[チューニング]タブでサンプリング時間を調節することによって、偏差に乗るノイズの影響を小さくすることができます。偏差のフィルタ結果は次式で表されます。

$$EF_n = EF_{n-1} + \frac{T_{Loop}}{T_{Filter}} (E_n - EF_{n-1})$$

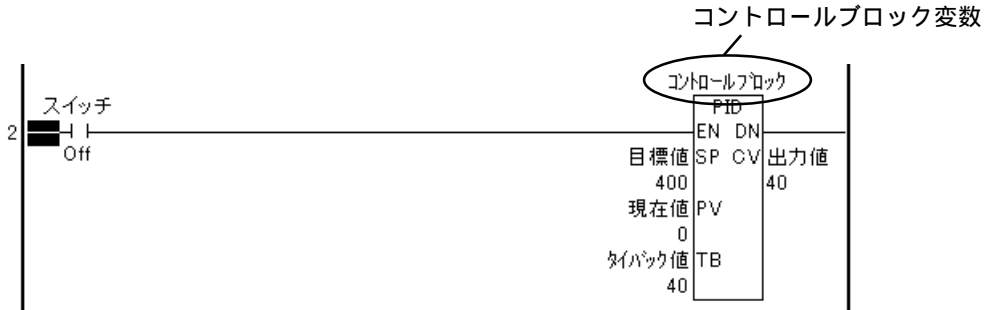
EF : 偏差のフィルタ結果
Tloop : ループ更新時間
TFilter : フィルタ時定数
E : 偏差 (SP-PV または PV-SP)
n : サンプリング回数



・ PID命令はGLC2000シリーズのみ対応です。

動作概要

PID 命令が導通すると、PID 演算を行い操作量を調節して出力します（自動モード）。下図のように非導通の場合は一定の操作量を出力します（手動モード）。手動モードの出力値はタイバックで設定します。



ロジックプログラムでPID命令を使用する際、まずコントロールブロック変数とSP、PV、TB、CVに変数を割り付けてください。

各パラメータの変数

変数	内容	変数タイプ
SP	目標値	整数、整数定数、整数配列
PV	現在値	整数、整数配列
TB	タイバック 命令が導通していない場合、ここで設定された値が出力されます。	整数、整数定数、整数配列
CV	出力値	整数、整数配列

コントロールブロック変数

PID 命令に変数を割り付けると、その変数には下表の要素数 7 の配列が自動的に割り付けられます。要素[0]はステータス、要素[1]～[6]はPID制御の微調整を行うことができます。

要素番号	詳細	
0	ビット0	モード切り替えフラグ
	ビット1	PID命令処理の完了フラグ
	ビット2	PID処理無効範囲フラグ
	ビット3	出力値の上限オーバー
	ビット4	出力値の下限オーバー
	ビット5	積分回数処理オーバー
1	比例係数	
2	毎分あたりの積分回数	
3	1回あたりの微分時間	
4	PID処理無効範囲	
5	オフセット	
6	サンプリング時間	



・ コントロールブロック変数の変数タイプは、保持型変数になります。



・ 比例係数、毎分あたりの積分回数、1回あたりの微分回数に代入される値は、「チューニング」タブの比例係数、積分回数、微分回数の値を 1000 倍した値になります。

コントロールブロック変数の要素[0]のステータス

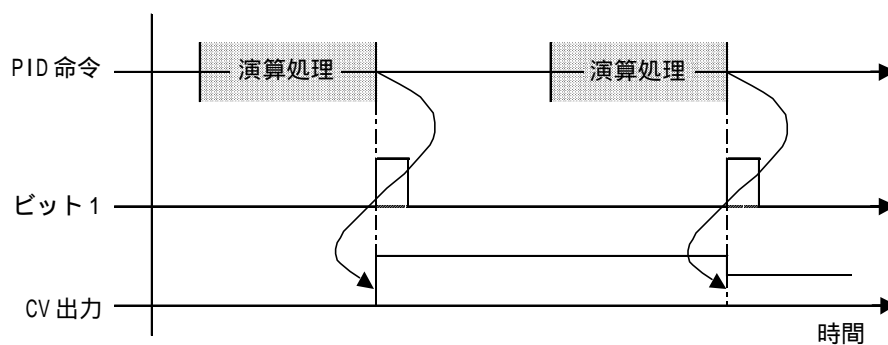
モード切り替えフラグ

ロジックプログラムでPID命令が導通している場合にビット0がONになります。

ビット0	モード
ON	自動モード (PID演算)
OFF	手動モード

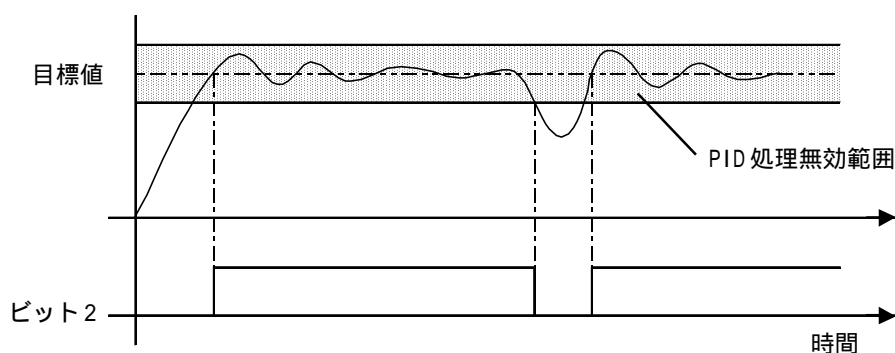
PID 命令処理の完了フラグ

演算処理が終了してCV値を出力するタイミングでビット1がONになります。ビット1は、1スキャンの間ONします。



PID 処理無効範囲フラグ

[PID]ダイアログボックスの[チューニング]タブ、もしくはコントロールブロック変数の要素[4]で指定した範囲内において、現在値が目標値に達したときにONになり、現在値が範囲外となったときOFFになります。

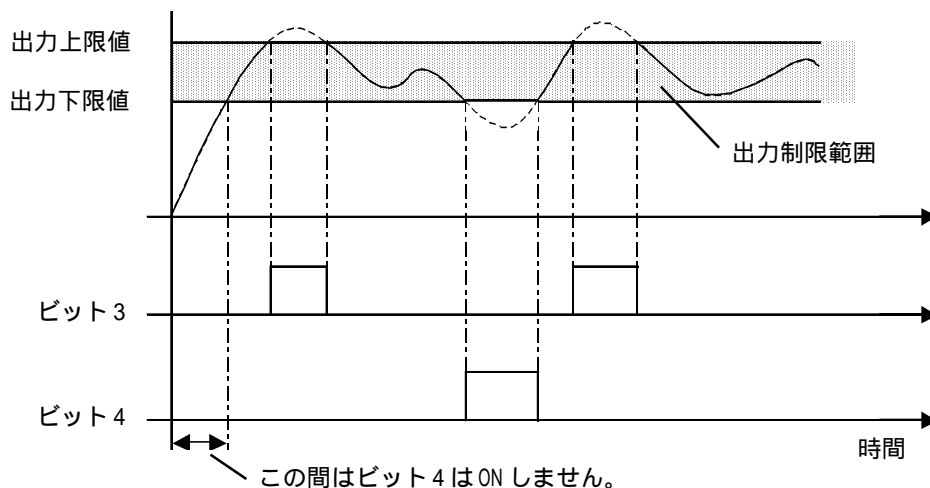


- ・ [PID]ダイアログボックスの[チューニング]タブについては、後述するPID制御の微調整とモニタリングを参照してください。

出力値の上限オーバーと下限オーバー

[PID]ダイアログボックスの[セットアップ]タブで指定した上限に出力値がある場合にビット3がONになり、下限に出力値がある場合にビット4がONになります。

各ステータスのビットがONになってもPID演算は続行され、設定した上限値または下限値で出力されます。



- ・ [PID]ダイアログボックスの[セットアップ]タブについては、後述するPID制御の微調整とモニタリングを参照してください。

積分回数処理オーバー

[PID]ダイアログボックスの[セットアップ]タブで指定した範囲外に積分回数が設定される場合、ビット5がONになります。ステータスのビットがONになってもPID演算は続行され、設定した上限値で出力されます。



- ・ [PID]ダイアログボックスの[セットアップ]タブについては、後述するPID制御の微調整とモニタリングを参照してください。

コントロールブロック変数の要素[1]～[6]

PID制御の微調整を行います。詳しくは、PID制御の微調整とモニタリングを参照してください。

PID制御の微調整とモニタリング

PID命令に、コントロールブロック変数と各パラメータの変数（SP、PVなど）を設定してPID命令をダブルクリックすると、下記の[PID]のダイアログボックスが表示されます。

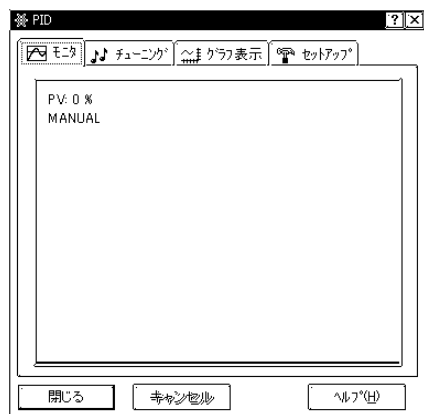
このダイアログボックスでは、PID制御の微調整やモニタリングを行うことができます。



・各パラメータを自動調整するオートチューニング機能はありません。

モニタ

モニタリングモード中にモニタを行うと、そのPID命令の実行結果をモニタすることができます。モニタリングモードの詳細については、[参照](#)「Pro-Control Editor オペレーションマニュアル 第4章 モニタリングモードでの動作確認」



グラフ線種

モニタリングしている各項目の線種および線色は下表のようになります。

項目	線種 / 色	
目標値	黒点線	-----
現在値	黒直線	————
出力値	青直線	————



・グラフの線種 / 色の変更はできません。

チューニング

モニタリングモード中に各値を調節することができます。ここで設定した値は、各パラメータの変数（SP、TB）やコントロールブロック変数の要素[1]～[6]に反映されます。モニタリングモードの詳細については、[参照](#)「Pro-Control Editor オペレーションマニュアル 第4章 モニタリングモードでの動作確認」



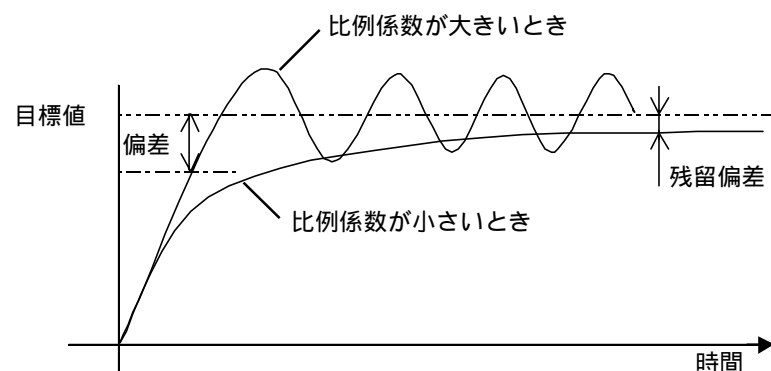
目標値： 目標値を設定します。

タイバック： ロジックプログラムでPID命令が導通していない場合、ここで設定された値が出力されます。

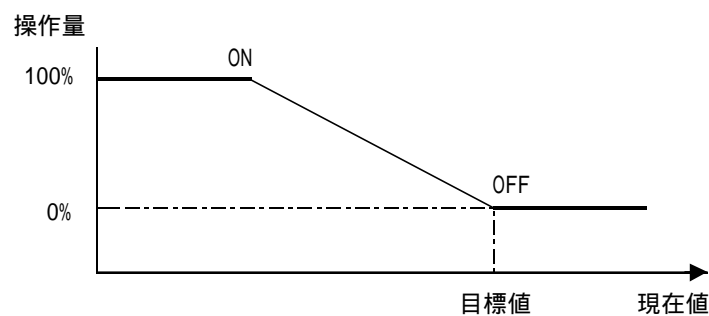
比例係数： 比例制御では、目標値に対して比例帯があり、その範囲内では目標値と現在値との偏差に比例した操作量が出力されます。比例帯は、比例係数より算出することができます。

$$\text{比例帯} = 100 / \text{比例係数} [\%]$$

比例係数を小さくすると、目標値に近づけようとする操作量は小さくなりオーバーシュートをなくしますが、残留偏差が大きくなる原因になります。また、比例係数を大きくすると目標値に近づけようとする操作量は大きくなり目標到達時間は短くなりますが、ハンティングする原因になります。



比例制御では、現在値が目標値より小さければ操作量は100%で最大となり、比例帯の範囲内は目標値と現在値との偏差に比例した操作量が出力され、目標値と現在値が一致（偏差なし）すると操作量が0%になります。



操作量：単位時間あたりの出力量

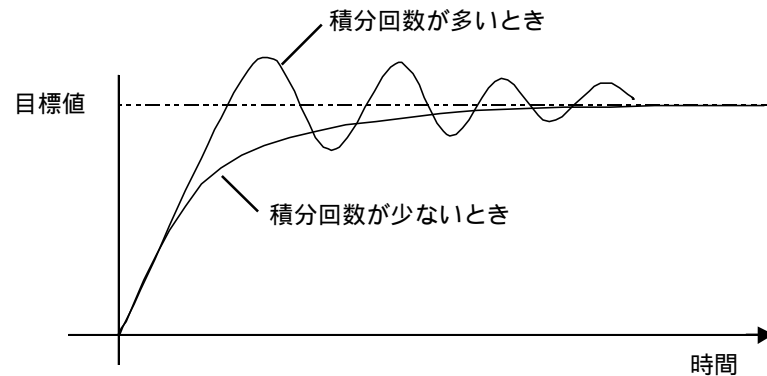
積分回数： 比例制御だけでは、目標値に近づくと操作量が小さくなりすぎ、操作量（制御出力）が偏差を埋めるだけの値が得られなくなります。そのわずかな誤差を残留偏差といい、積分制御をもちいることによって、この残留偏差を無くすることができます。

積分制御では、偏差を時間的に累積して、ある大きさになれば操作量を増して偏差をなくすように調節する制御方式です。

積分回数は、単位時間あたりに積分を行う回数を設定します。

積分回数を多くすると、目標値に近づけようとする操作量は大きくなり目標到達時間は短くなりますが、オーバーシュート、ハンティングの原因となります。

また、積分回数を少なくすると、目標値に近づけようとする操作量は小さくなりオーバーシュート、ハンティングをなくしますが、目標到達時間は長くなります。

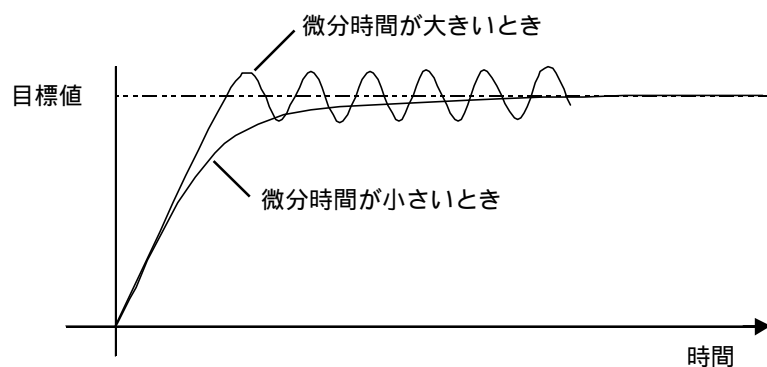


微分時間： 一定の時間（時定数）を必要とする比例制御や積分制御では、外乱に対してすばやく反応できず、すぐには元の目標値には戻せません。

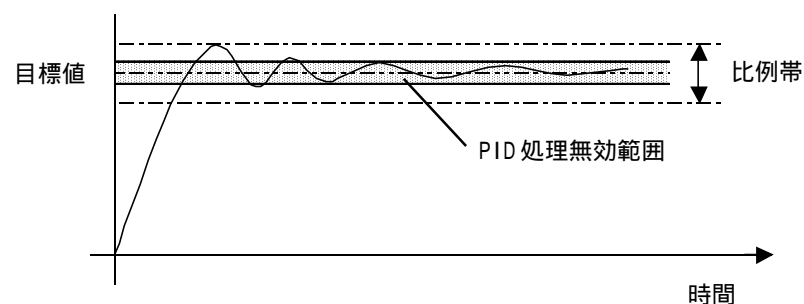
微分制御では、外乱に対して偏差を見て、前回偏差との差が大きいときには大きな操作量を与え、機敏に反応するように働きます。

微分時間を大きくすると、外乱に対しての復旧時間は短くなりますが、オーバーシュート、短い周期のハンティングの原因となります。

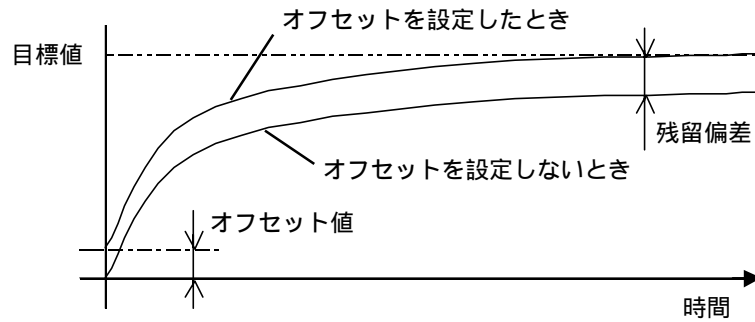
また、微分時間を小さくすると、オーバーシュート、ハンティングをなくしますが、外乱に対しての復旧時間は長くなります。



処理無効範囲： 比例帯の領域内に設定します。「処理無効範囲」では、PID制御は行わず、最小出力の値を出力し、ハンティングのない、滑らかな制御を行います。

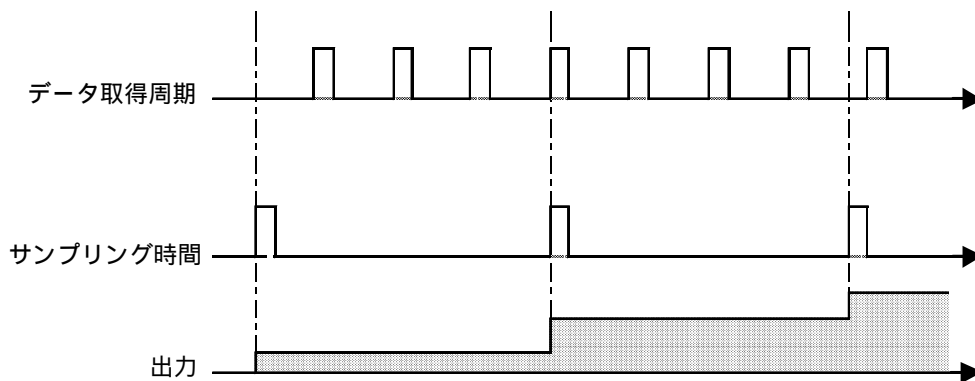


オフセット： オフセット値を設定します。比例制御などで生じた残留偏差を少なくすることができます。



サンプリング時間：「データ取得周期」で取得した接続機器の測定データを平均して、サンプリング時間で設定した時間毎に出力します。よって、サンプリング時間は、「データ取得周期」の値より大きく設定してください。

サンプリング時間を設定することで、接続機器の測定データの1つが思いもよらない値となっても、他の測定データとの平均をとって演算するので出力値への影響は小さくなります。



チューニングタブからコントロールブロック変数への反映

チューニングタブの各設定項目は、パラメータの変数（SP、TB）とコントロールブロック変数の要素[1]～[6]に反映されます。

下表は、チューニングタブとパラメータの変数、チューニングタブとコントロールブロック変数の対応表になります。比例係数、積分回数、微分時間は、チューニングタブで設定した1000倍の値がコントロールブロック変数内に書き込まれます。

チューニングタブ	倍率	→	パラメータの変数	
目標値	×1		SP	変数
タイバック	×1		TB	変数

チューニングタブ	倍率	→	コントロールブロック変数	
比例係数	×1000		比例係数	コントロールブロック変数[1]
積分回数	×1000		毎分あたりの積分回数	コントロールブロック変数[2]
微分時間	×1000		1回あたりの微分回数	コントロールブロック変数[3]
処理無効範囲	×1		PID処理無効範囲	コントロールブロック変数[4]
オフセット	×1		オフセット	コントロールブロック変数[5]
サンプリング時間	×1		サンプリング時間	コントロールブロック変数[6]

グラフ表示

現在値(PV)、目標値(SP)、出力値(CV)、出力無効範囲、出力範囲の値をそれぞれモニタリングすることができます。また、モニタリングの設定もできます。



表示項目

モニタリングしている各項目のグラフ線種および線色は下表のようになります。

項目	線種/色	
目標値	黒点線	-----
現在値	黒直線	—————
出力値	青直線	—————
出力範囲	赤点線	-----
出力無効範囲	グレーゾーン	




・グラフの線種/色の変更はできません。

グラフ表示範囲

上限値：グラフ表示範囲の上限値を設定します。

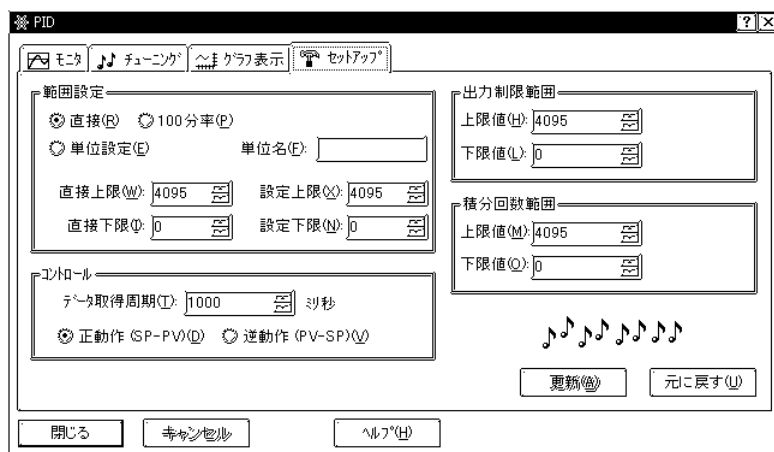
下限値：グラフ表示範囲の下限値を設定します。


表示幅：表示時間の幅を設定します。グラフ表示のサンプリング時間はエディタの[ファイル]メニューから[オプション]を選択し、オプションダイアログボックスの「モニタ」タブの表示周期で変更できます。

 過去のモニタを見ることはできません。

セットアップ

プログラミングモード中に、全パラメータに設定できる範囲（上限値と下限値）をあらかじめ設定することができます。



 モニタリングモード中は設定できません。

範囲設定

「直接」、「100分率」、「単位設定」は、PVのデータ値とモニタリングなどの表示部分での数値との変換率を設定します。直接上限と直接下限の値はPVのデータ値で、設定上限と設定下限の値はモニタリングなどの表示部分での値を指定します。

直接：変換率0で接続機器への入出力の値がそのまま表示されます。

この「直接」を選択した場合は、直接上限と直接下限、設定上限と設定下限の値は下記のような設定になります。

直接上限の値 = 設定上限の値

直接下限の値 = 設定下限の値

100分率：表示部分に100分率された値が表示されます。

この「100分率」を選択した場合は、直接上限と直接下限、設定上限と設定下限の値は下記のような設定になります。

直接上限、直接下限の値 = 接続機器によるユーザー設定

設定上限の値 = 100、設定下限の値 = 0

単位設定 : 表示部分でユーザー設定によるn分率された値が表示されます。

この「単位設定」を選択した場合は、直接上限と直接下限、設定上限と設定下限の値は下記のように設定してください。

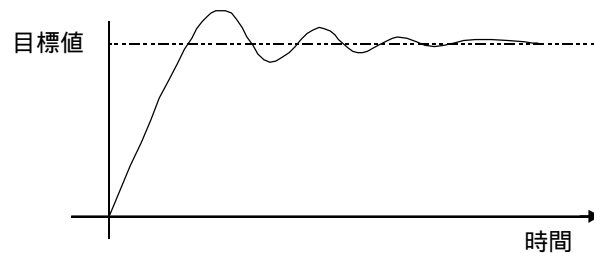
直接上限、直接下限の値 = 接続機器によるユーザー設定

設定上限の値 = n、設定下限の値 = 0

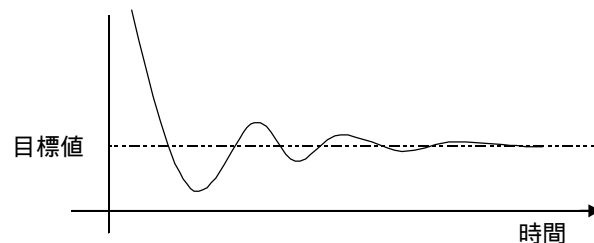
コントロール

データ取得周期 : 接続機器からデータを取得する時間の周期を設定します。

正動作(SP-PV) : 現在値が目標値より小さいときに操作量を増加させる制御を行う場合に指定します。(暖房など)

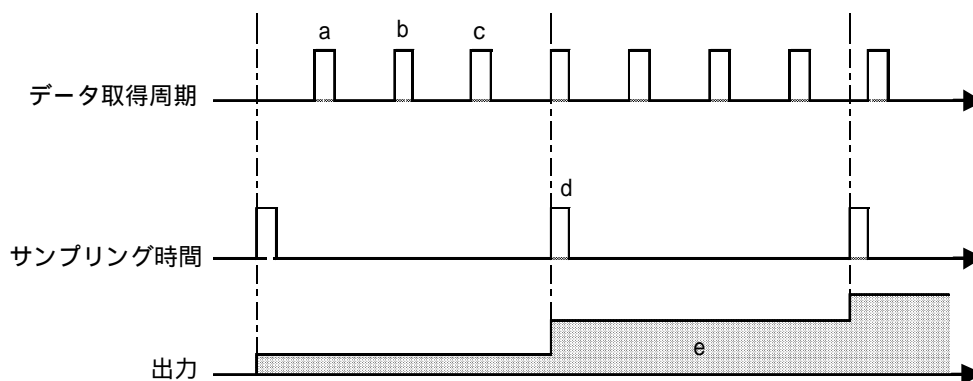


逆動作(PV-SP) : 現在値が目標値より大きいときに操作量を増加させる制御を行う場合に指定します。(冷房など)



データ取得周期とサンプリング時間の出力タイミング例

下記のタイミングチャートでは、データ取得周期の「a」、「b」、「c」で接続機器から取得した測定データを平均し、サンプリング時間「d」のタイミングで「e」が出力されます。



出力制限範囲

出力値の上限値と下限値を設定します。ここで設定された値の範囲外の出力であれば、上限値もしくは下限値の値で出力され、コントロールブロック変数の要素[0]のビット3もしくはビット4のステータスビットがONします。詳しくは、出力値の上限オーバーと下限オーバーを参照してください。

積分回数範囲

コントロールブロック変数の要素[2]の毎分あたりの積分回数の上限値と下限値を設定します。「毎分あたりの積分回数」の値は自然変動しませんが誤入力を防ぎます。

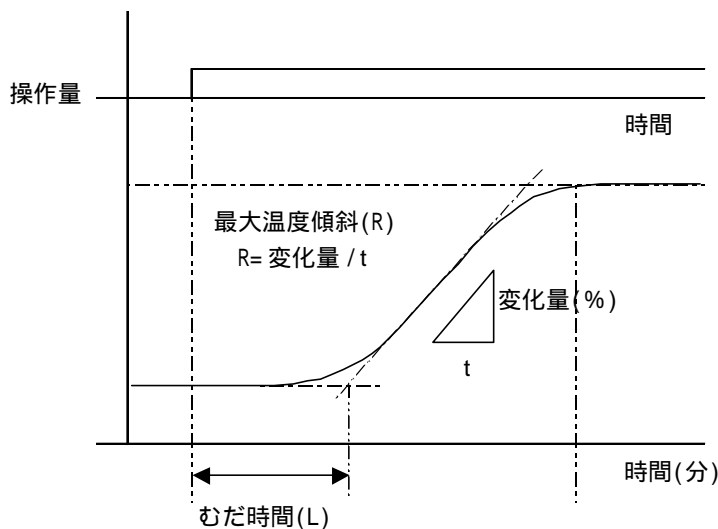
PID 定数の調整法

温度制御の場合を例にして説明します。PIDの制御結果を最適なものにするためには、P (比例要素)・I (積分要素)・D (微分要素)の各定数を最適値にする必要があります。さまざまな制御対象に対して、PID定数を温度特性から導き出す方法としてステップ応答法があります。

ただし、制御対象、用途によっては最適値にならない場合がありますので、そのときは[PID]ダイアログボックスの[チューニング]タブで調整してください。

ステップ応答法

目標値を設定して、制御対象に対して操作量 100% をステップ状に出力します。このときの下表の温度特性グラフより、最大温度傾斜(R)とむだ時間(L)を計測します。



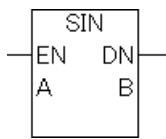
最大温度傾斜(R)とむだ時間(L)を計測した値を下記の方程式に代入して”比例係数”、”積分回数”、”微分時間”の定数を算出します。算出した値を「PID/ チューニング」ダイアログボックスに代入してください。

$$\text{” 比例係数 ”} = 100 / (0.83 \cdot R \cdot L) \quad [\%]$$

$$\text{” 積分回数 ”} = 1 / (2 \cdot L) \quad [\text{回} / \text{min}]$$

$$\text{” 微分時間 ”} = 0.5 \cdot L \quad [\text{min}]$$

4.2.48 SIN(sin 関数)



A: データ (ラジアン)
B: 結果格納先

SIN 命令を実行すると、 $\sin(A)$ を B に格納します。A はラジアンで入力し、結果 B は、 $-1.0 \sim 1.0$ の実数値となります。

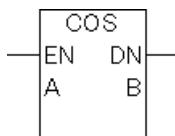
この命令は常に導通します。SIN 命令が実行できる A、B の組み合わせは以下の通りです。

Aのタイプ	Bのタイプ
整数または整数定数	実数
実数または実数定数	実数



・ SIN 命令は GLC2000 シリーズのみ対応です。

4.2.49 COS(cos 関数)



A: データ (ラジアン)
B: 結果格納先

COS 命令を実行すると、 $\cos(A)$ を B に格納します。A はラジアンで入力し、結果 B は、 $-1.0 \sim 1.0$ の実数値となります。

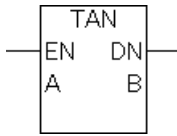
この命令は常に導通します。COS 命令が実行できる A、B の組み合わせは以下の通りです。

Aのタイプ	Bのタイプ
整数または整数定数	実数
実数または実数定数	実数



・ COS 命令は GLC2000 シリーズのみ対応です。

4.2.50 TAN(tan 関数)



A: データ (ラジアン)
B: 結果格納先

tan 命令を実行すると、 $\tan(A)$ を B に格納します。A はラジアンで入力し、結果 B は、 $\pm 2.225e-308 \sim \pm 1.79e+308$ の実数値となります。

この命令は常に導通します。TAN 命令が実行できる A、B の組み合わせは以下の通りです。

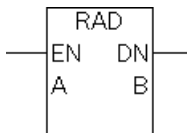
Aのタイプ	Bのタイプ
整数または整数定数	実数
実数または実数定数	実数

A が $(2n-1) \times \frac{\pi}{2}$ (n は整数) 付近では、B は表示可能な値でなくなるため #Overflow が ON となり、解は不定となります。($\pi = 3.1415926535897$) 参照 「3.2.21 #Overflow」



・ TAN 命令は GLC2000 シリーズのみ対応です。

4.2.51 RAD(ラジアン変換)



A: データ (度)
B: 結果格納先 (ラジアン)

RAD 命令を実行すると、角度単位の "度" を "ラジアン" に変換して B に格納します。

この命令は常に導通します。RAD 命令が実行できる A、B の組み合わせは以下の通りです。

Aのタイプ	Bのタイプ
整数または整数定数	実数
実数または実数定数	実数



・ $\pi = 3.1415926535897$ です。
・ RAD 命令は GLC2000 シリーズのみ対応です。

第5章

LS エリアリフレッシュ

5.1 LS エリアリフレッシュの概要

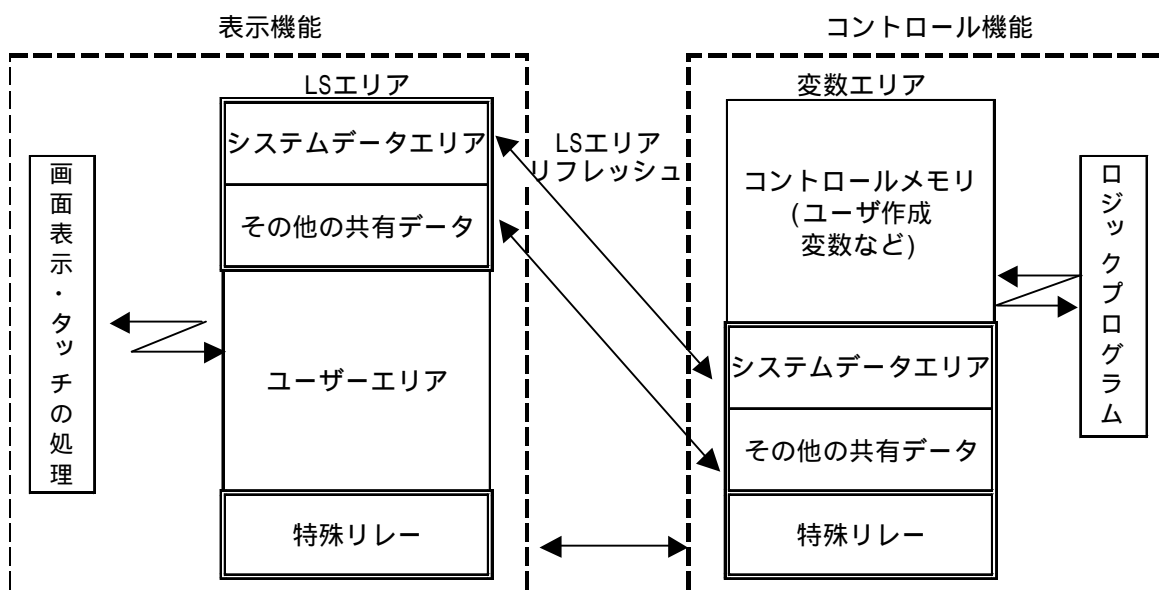
LS エリアリフレッシュ機能

GLCでは、画面切り替えや時計などをLSエリアのシステムデータエリアで管理しています。これらは表示機能として処理されています。

そのためコントロール機能上で画面切り替えや時計などのシステムデータエリア内に割り付けられた機能を使用する場合は、LSエリアを変数として登録し、表示機能とコントロール機能間でLSエリアのデータ共有をする必要があります。

これをLSエリアリフレッシュといいます。

また、システムデータエリア以外に表示機能とコントロール機能で共有させたいデータがある場合にもLSエリアリフレッシュを使用します。



LSエリアリフレッシュのタイミングとロジックシンボルデータ更新タイミングは非同期です。どちらかのトリガでもう一方のデータ更新をロジック上でプログラミングする際はインターロックをかけてください。

5.2 LSエリアリフレッシュの設定

ロジックプログラムにてLSエリアを指定するには、Pro-Control Editorにて変数を登録する必要があります。ここではPro-Control Editorにて登録する方法を説明します。

登録方法

Pro-Control Editorより、[データ]メニュー [変数タイプ]を選択すると、[変数タイプ]ダイアログボックスが表示されます。

「LS」という名前(半角で「LS」と入力)の変数をインターナル整数、配列として登録します。サイズはシステムエリア分で20ワード、その他に共有させたいデータのワード数分を足して算出します。(例: システムデータエリア以外に16ワードのデータを共有したい場合はシステムデータエリア20ワード+16ワード=36ワードを入力)



- ・ 特殊リレーエリアは「LSS」という変数名になります。
- ・ 「LS」のサイズは最大276ワードです。

変数とアドレスとの関係は下表に示した通りです。

変数名 ¹	アドレス	LSアドレス	
LS[0]	0	LS0000	システムデータエリア
LS[1]	1	LS0001	
⋮	⋮	⋮	
LS[19]	19	LS0019	
⋮	⋮	⋮	その他の共有データ
LS[275]	275	LS0275	
LSS[0]	2032	LS2032	特殊リレー
LSS[1]	2033	LS2033	
⋮	⋮	⋮	
LSS[15]	2047	LS2047	

1 変数名: GLCのロジックプログラムで扱うシステム変数

「LSエリア」、「特殊リレー」の詳細については機器接続マニュアル(PLC接続マニュアル)を参照してください。

GLC で表示している画面番号の確認例

GLC の画面表示番号を確認する場合、参照する LS エリアはダイレクトアクセス方式とメモリリンク方式によって異なります。

画面表示番号 : LS[0] ダイレクトアクセス方式
 : LS[15] メモリリンク方式

下図は画面番号が "5" に切り替わったときに "メッセージ" コイルが ON するロジックプログラムです。

ダイレクトアクセス方式 (外部通信機器との接続使用時)

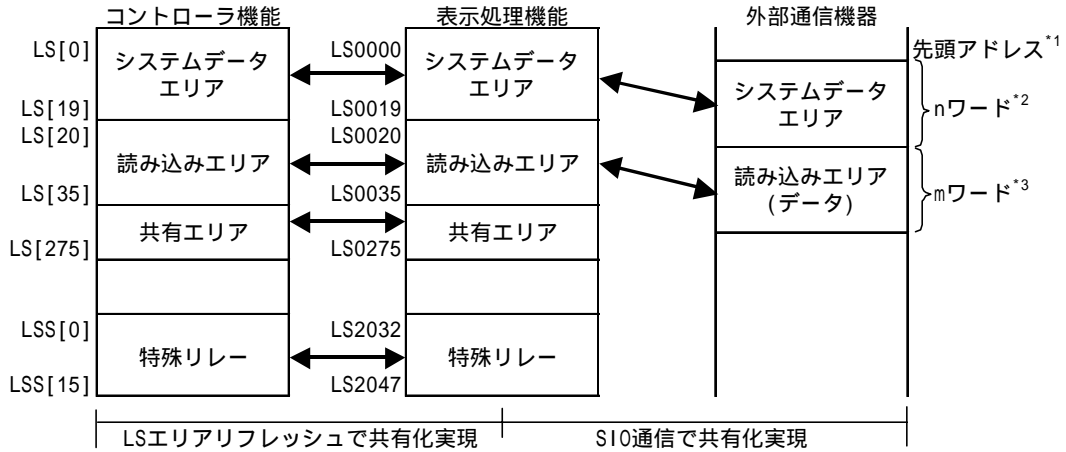


メモリリンク方式 (GLC 単体使用時)



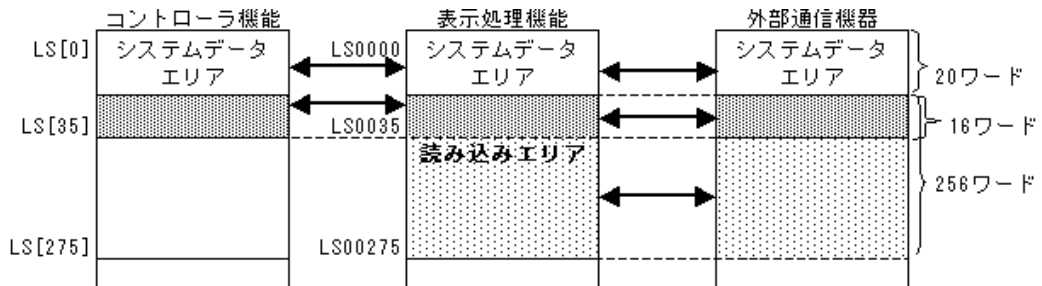
5.3 GLC と PLC のデータ共有について

コントロール機能で外部通信機器のデータを使用する場合は、LS エリアを経由してデータ共有をおこないます。ただし、コントローラ機能と外部通信機器のデータレジスタのデータ共有が16ワードを越えた場合、画面表示機能が著しく低下する場合があります。

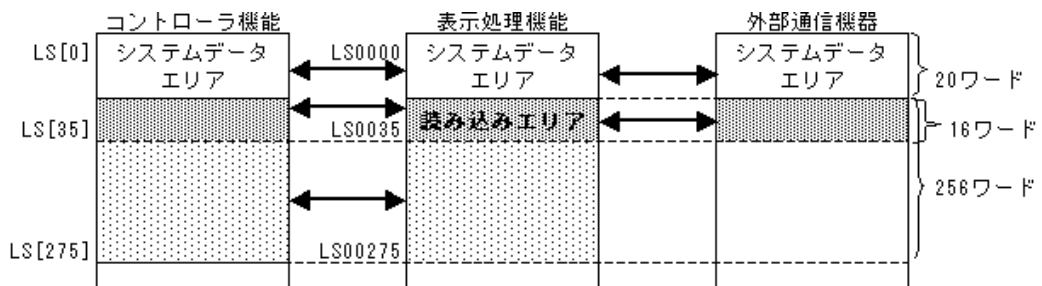


読み込みエリアまたは変数LSを16ワードを超えて設定したい場合、読み込みエリアは256ワード、変数LSのサイズは276ワードまで設定可能です。コントローラ機能、表示処理機能、外部通信機器で共有するデータは16ワードまでに設定することを推奨いたします。

例) 変数LSのサイズを36ワードに、読み込みエリアを256ワードに設定した場合塗り込み部分がデータを共有できるエリアになります。



例) 変数LSのサイズを276ワードに、読み込みエリアを16ワードに設定した場合塗り込み部分がデータを共有できるエリアになります。

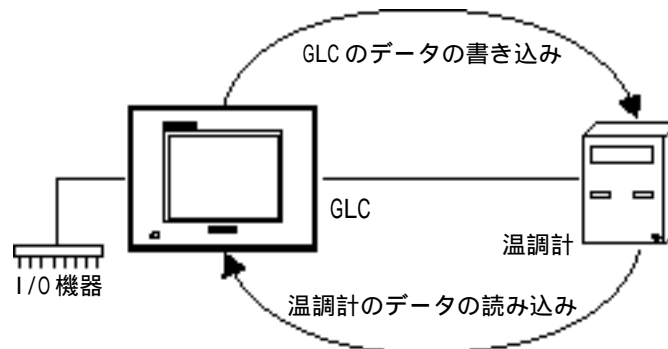


*1 初期設定で指定したシステム先頭アドレスのことです。
 *2 n=0 ~ 20 初期設定で指定したシステムデータエリアの選択項目数によって異なります。
 *3 m=0 ~ 16 初期設定で指定した読み込みエリアの大きさによって異なります。

- 重要**
- ・ コントロールのロジックプログラムと GLC の部品、外部通信機器のロジックプログラムで同一変数にデータ更新した場合、どのデータが優先されるかタイミングにより異なります。
 - ・ GLC において、読み込みエリアにデータ書き込みを行う場合は、部品による書き込みとコントローラ機能のロジックプログラムによる書き込みが競合しないようご注意ください。
 - ・ コントローラ機能の変数 LS および変数 LSS は保持型に設定してください。非保持型に設定しているとロジックプログラム起動時に 0 クリアされます。その後、LS リフレッシュにより表示処理機能の LS エリアが 0 クリアされます。



- ・ 読み込みエリアを上手に使って GLC と外部通信機器のデータ共有を行うと、GLC を外部通信機器の子機として利用したり、FA 向け POP マシンや、生産管理用 I/O 情報収集端末の構築に有効に使用できます。



5.3.1 GLC と外部通信機器のデータ共有時の注意点

GLC と外部通信機器のデータ共有は、コントローラ機能によるシステムエリアの制御と外部通信機器からの読み込みデータをコントローラ機能で参照する場合に活用してください。活用方法としてはコントローラ機能で LS0000 ~ LS0035 と LS2032 ~ LS2047 を頻繁にデータ更新するようなことは避け、初期セットや運転指示変更のパラメータセットなどプリセットに関するデータの授受に限定して使用することをおすすめします。

上記の LS エリアのデータ更新頻度を上げると LS エリアリフレッシュが 1 スキャン内に実行されない恐れがあります。その場合、「外部通信機器との通信異常」などの異常が発生することがありますので注意してください。

変数 LS は整数変数のため、32 ビット長になります。

システムデータエリアが 16 ビット長の場合、下位 16 ビットが有効になります。

MEMO

このページは、空白です。
ご自由にお使いください。

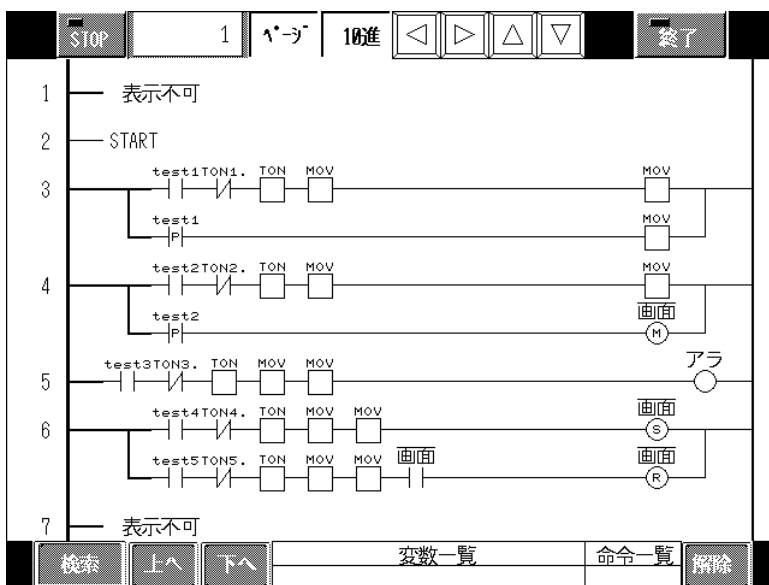
第6章

GLC ラダーモニタ機能

6.1 GLC ラダーモニタ機能の概要

Pro-Control Editor ではロジックプログラムのメンテナンス性向上のため、GLC 本体でのラダーモニタ機能を実現しています。ラダーモニタ中でもロジックプログラムなど全てのプログラムは実行されています。

GLC の画面上にロジックプログラムを表示します。対応機種は、GLC2400 シリーズ、GLC2600 シリーズです。



- ・ GLCラダーモニタ機能では、ロジックプログラムの編集や変数のに格納された値の変更はできません。
- ・ GP-Web、GP-Viewer で、GLCラダーモニタ画面をモニタリングすることはできません。

6.2 GLC ラダーモニタの起動と終了方法

6.2.1 GLC ラダーモニタの運転準備

GLC ラダーモニタ機能を行うためには、作画画面やロジックプログラムと同様に GLC ラダーモニタ用のプロジェクトファイルを GLC へ転送しなければいけません。

ここでは、このプロジェクトファイルの設定方法と注意事項について説明します。

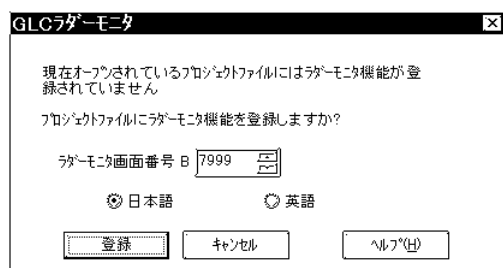
プロジェクトファイルの設定方法

1. プロジェクトマネージャの[画面 / 設定]メニューから[GLC ラダーモニタ]を選択します。

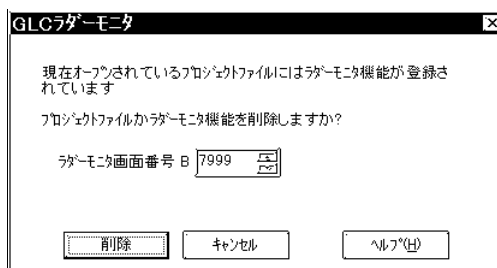


2. 「GLC ラダーモニタ」ダイアログボックスが表示されます。プロジェクト内に GLC ラダーモニタ画面（ラダーモニタで使用するベース画面）が存在するかどうかをチェックし、未登録時には「GLC ラダーモニタ登録」ダイアログボックスが表示され、既に登録されていれば「GLC ラダーモニタ削除」ダイアログボックスが表示されます。

GLC ラダーモニタ登録



GLC ラダーモニタ削除



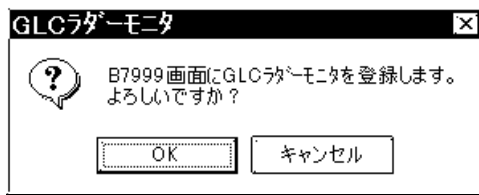
GLC ラダーモニタで使用するベース画面番号を設定し、[登録] ボタンをクリックします。

- 1 ~ 8999 の範囲で設定してください。ただし、「表示画面番号のデータ形式」が BCD の時は、1 ~ 7999 の範囲でのみ動作します。

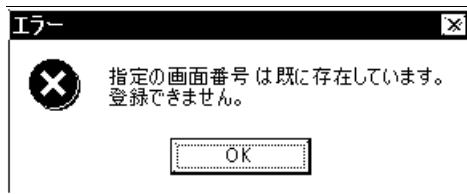


- ・ GLC ラダーモニタで使用するベース画面番号は、既に使用している作画画面を重複しないように設定してください。

3. [登録]ボタンをクリックすると、GLC ラダーモニタ画面の登録が行われます。GLC ラダーモニタ登録時には下記の警告メッセージが表示されます。



既に指定画面が存在する場合は以下のメッセージが表示され、GLC ラダーモニタの画面の登録はおこなわれません。新規に指定画面番号にGLC ラダーモニタ画面を登録する場合は、既存の画面を削除してから再度、登録を行ってください。



- GLC ラダーモニタ画面はベース画面に自動生成されるため、編集可能となっています。しかし、一度でも画面エディタでGLC ラダーモニタ画面を編集すると、以後GLC ラダーモニタ画面と認識しないため、GLC ラダーモニタ登録は自動的に解除されます。

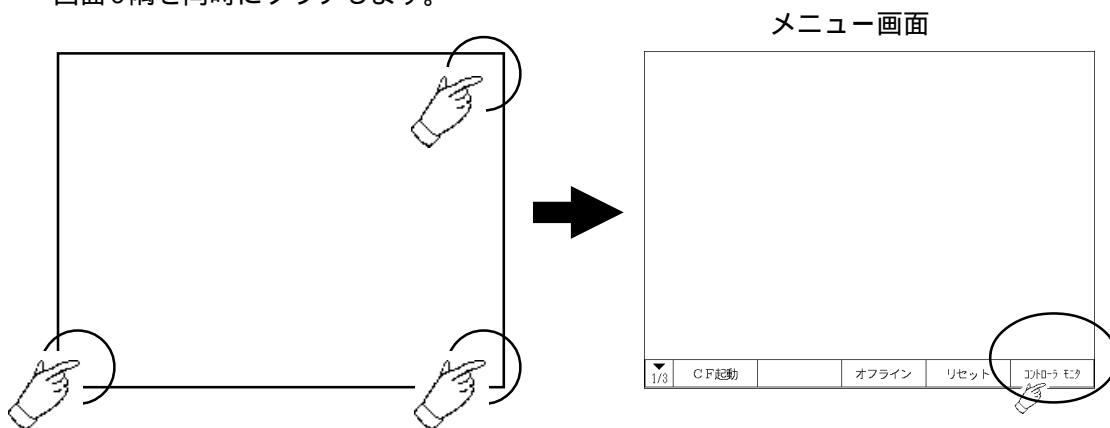
画面エディタの「画面(S)/画面を開く(O)」の「画面を開く」ダイアログボックスで既存のGLC ラダーモニタ画面を削除してから再度、登録を行ってください。
- GLC ラダーモニタ画面を他のプロジェクトへコピーしても認識されません。

画面エディタの「画面(S)/画面を開く(O)」の「画面を開く」ダイアログボックスで既存のGLC ラダーモニタ画面を削除してから再度、登録を行ってください。
- GLC の機種変更をした場合 (GLC2400 GLC2600)、機種変更後の画面はGLC ラダーモニタ画面と認識されません。既存のGLC ラダーモニタ画面を削除してから再度、登録を行ってください。

6.2.2 GLC ラダーモニタの起動方法

GLC ラダーモニタを起動するには、以下の2つの方法があります。

- ・ システム変数 #LadderMonitor の Bit-0 を ON します。参照「3.2.26 #LadderMonitor」(#LadderMonitor のビット操作による GLC ラダーモニタの起動をする場合、階層画面切替モードは使用できません。)
- ・ メニューバーの「コントローラモニタ」をタッチします。メニューバーの表示方法は、画面3隅を同時にタッチします。

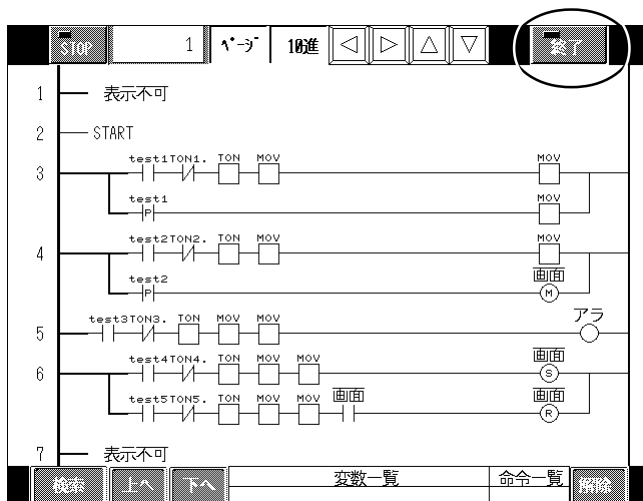


- 重要**
- ・ オンラインエディットにてロジックプログラムを編集後は GLC ラダーモニタを起動することはできません。
 - ・ GLC ラダーモニタ起動中はオンラインエディットを実行できません。
 - ・ GLC に転送されたプロジェクトファイルにパスワードが設定されている場合、GLC ラダーモニタを起動することはできません。

6.2.3 GLC ラダーモニタの終了方法

GLC ラダーモニタを終了するには、以下の2つの方法があります。

- ・ GLC ラダーモニタ画面上の「終了」ボタンを押すことで終了します。
- ・ PLC 等から自動的に画面切り替えを行うことで終了します。



6.3 GLC ラダーモニタの各種機能

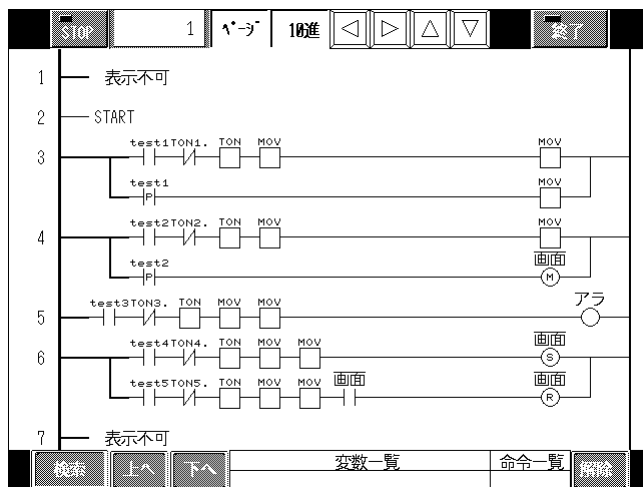
6.3.1 導通モニタ機能（ノーマル表示）

GLCにロジックプログラムが表示され、導通しているロジックプログラムのラングは緑色の太線で表示されます。

1画面に表示可能な命令数は下記の通りです。

GLC2400:横 14 命令、縦 10 ラング（ただし、4 命令分はスクロールで表示可能）

GLC2600:横 18 命令、縦 13 ラング



RUN/STOP 切り替えボタン

GLC コントローラの状態を RUN/STOP に切り替えます。

ラング番号表示ボタン

現在表示している画面の先頭ラング番号を表示します。

スクロール単位切り替えボタン

画面スクロールする単位をラング / ページに選択します。

表示基数切り替えボタン

変数の現在値表示を 10 進数 / 16 進数に切り替えます。

スクロールボタン

表示中の画面を上下左右にスクロールします。

GLC ラダーモニタスイッチ

GLC ラダーモニタの起動 / 終了を切り替えます。

検索ボタン

命令 / 変数指定後に検索の開始をします。

再検索ボタン

検索モードにおいて、上 / 下に再検索します。

変数一覧ボタン

変数モニタリング画面が起動します。

命令一覧ボタン

命令検索のための命令指定画面に移行します。

検索解除ボタン

検索モードを終了します。



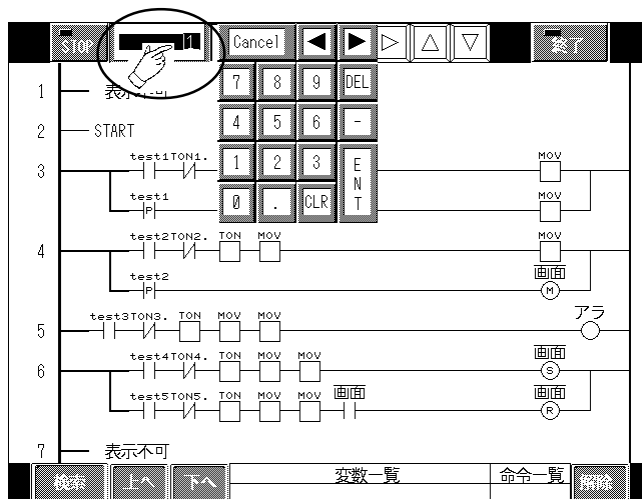
- ・ 横方向に18個を超える命令があるラングの場合、18個目以降はスクロールしても表示されません。
- ・ 縦方向に表示可能数（GLC2400は10、GLC2600は13）を超える命令があるラングは表示されず、「表示不可」と表示されます。
- ・ 縦方向に複数行あるラング（分岐回路）において、スクロールでラング全体が表示されない場合は「表示不可」となります。
- ・ 接点命令とコイル命令の変数名は、先頭から半角5文字（全角2文字）が表示されます。

6.3.2 ラングジャンプ / スクロール機能

1画面に全ロジックプログラムが表示しきれない場合、表示させたいラングへ画面を移動させる機能です。

ラングジャンプ機能

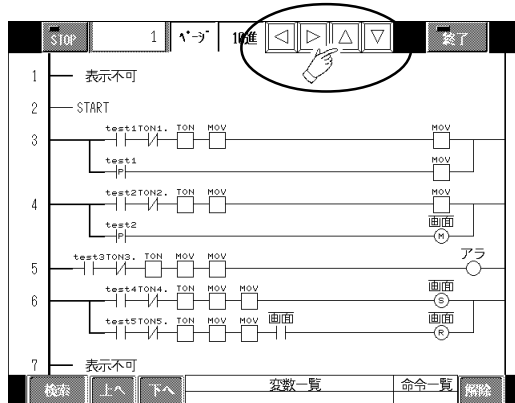
1. ラング番号表示ボタンをタッチすることにより、キーボードが表示されます。



2. キーボードで表示させたいラング番号を入力します。
3. [ENT]ボタンを押すと、入力したラング番号の画面へジャンプします。

スクロール機能

スクロールキーを押すことにより、表示画面を移動させることができます。



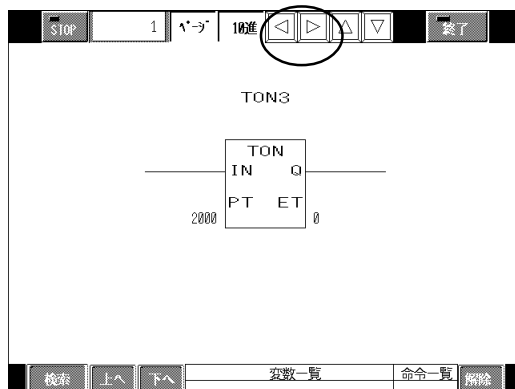
6.3.3 命令拡大機能（拡大表示）

GLCラダーモニタのノーマル表示画面上で命令をタッチすることにより拡大表示されます。

拡大表示することで、変数名はフル表示されます。

またタイマやカウンタ命令やMOV命令などでは、現在値や設定値も表示され、表示基数切り替えボタンにより、現在値の表示を10進/16進で切り替えが可能です。

もう一度、画面をタッチすることによりノーマル表示に戻ります。



- 変数名、変数値の表示は32文字まで可能です。
変数名に関して、32文字目が漢字の1バイト目である場合の表示は、31文字となります。
- BMOVなどで配列変数を指定している場合は、その配列変数の先頭要素の現在値表示となります。

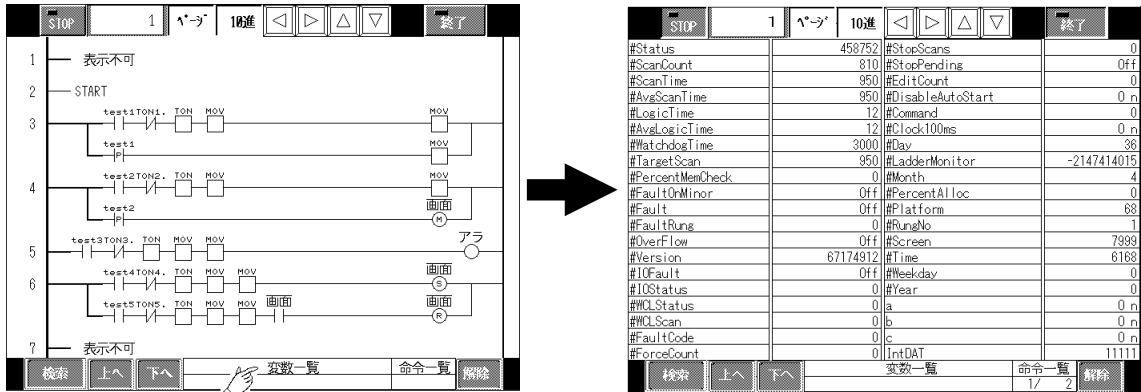
PID命令のパラメータ

PID命令を拡大すると画面右下に7つのパラメータが表示されます。パラメータは以下のようにコントロールブロック変数に割り当てられる要素数7の配列と同様のパラメータを10進数/16進数で表示します。

- Status : ステータスを表示します。
- Kc : 比例係数を表示します。
- Reset : 毎分あたりの積分回数を表示します。
- Rate : 1回あたりの微分時間を表示します。
- Deadband : PID処理無効範囲を表示します。
- Offset : オフセットを表示します。
- DFTC : サンプリング時間を表示します。

6.3.4 GLC 変数モニター機能

変数一覧および各変数の現在値を表示します。



- 変数一覧画面からノーマル画面に戻るには、「解除」ボタンで画面が切り替わります。

変数名の表示色

変数属性	表示色
システム変数	紫
ユーザ変数	白

変数値の表示色

変数タイプ	表示色
ディスクリート	緑
整数	白
実数	水
タイマ	紫
カウンタ	黄



- GLC2600 において、変数名の表示は最大半角 32 文字までとなります。ただし、32 文字目が全角の 1 バイト目であった場合、31 文字までの表示となります。
- GLC2400 では、変数名の表示は最大半角 23 文字までとなります。ただし、24 文字目が全角の 1 バイト目であった場合、23 文字までの表示となります。
- 配列変数は、先頭要素の現在値の表示となります。

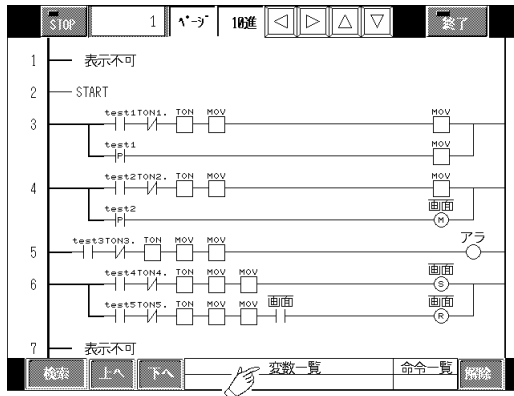
6.3.5 変数 / 命令検索機能

一覧表から指定した変数もしくは命令をロジックプログラムから検索し、該当する変数もしくは命令を水色の枠で囲みます。

変数からの検索

「変数一覧」画面で指定した変数を検索します。
検索方法は下記の通りです。

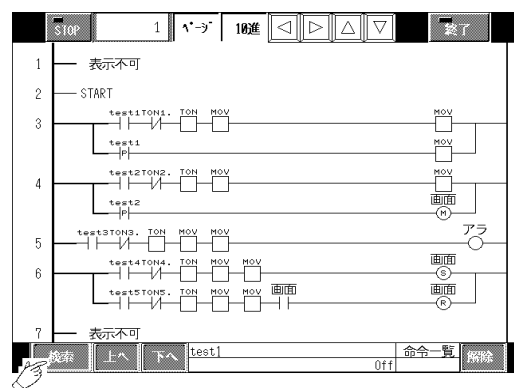
1. ノーマル表示画面の下部にある「変数一覧」ボタンを押し、変数一覧画面を表示させます。



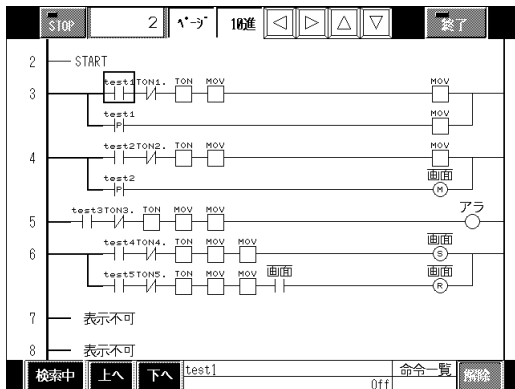
2. 変数一覧画面には、システム変数とロジックプログラムで設定されているすべての変数が表示されています。検索したい変数をタッチして選択します。

変数名	値	変数名	値
#Status	458752	#StopScans	0
#ScanCount	810	#StopPending	Off
#ScanTime	950	#EditCount	0
#AvgScanTime	950	#DisableAutoStart	0 n
#LogicTime	12	#Command	0
#AvgLogicTime	12	#Clock100ms	0 n
#WatchdogTime	3000	#Day	36
#TargetScan	950	#LadderMonitor	-2147414015
#PercentMemCheck	0	#Month	4
#FaultOnMinor	Off	#PercentAlloc	0
#Fault	Off	#Platform	68
#FaultRung	0	#RunNo	1
#Overflow	Off	#Screen	7999
#Version	67174912	#Time	6188
#IOFault	Off	#Weekday	0
#IOStatus	0	#Year	0
#WCLStatus	0	a	0 n
#WCLScan	0	b	0 n
#FaultCode	0	c	0 n
#ForceCount	0	IntDAT	11111

3. 変数一覧画面で変数を選択すると、自動的にノーマル表示画面に戻ります。画面下部に検索する変数名が点滅しているのを確認し、「検索」ボタンを押すと検索を開始します。



4. 検索の結果、該当する変数は、水色の枠で囲まれて表示されます。検索を終了するには、「解除」ボタンを押してください。

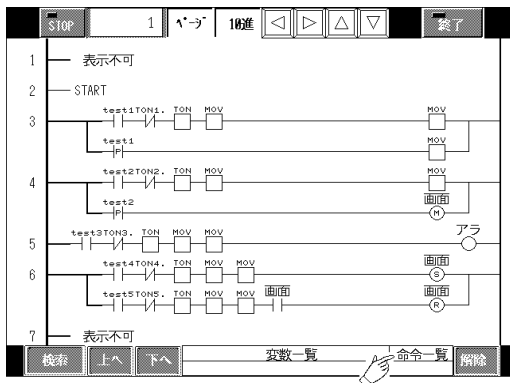


- ・ さらに検索を継続するときには、「上へ」、「下へ」ボタンを押すと次の検索対象変数に移行します。

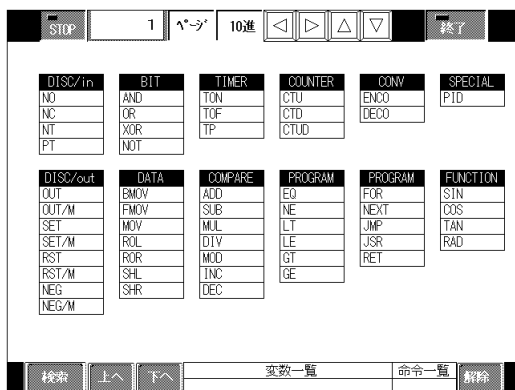
命令からの検索

「命令一覧」画面で指定した命令を検索します。
 検索方法は下記の通りです。

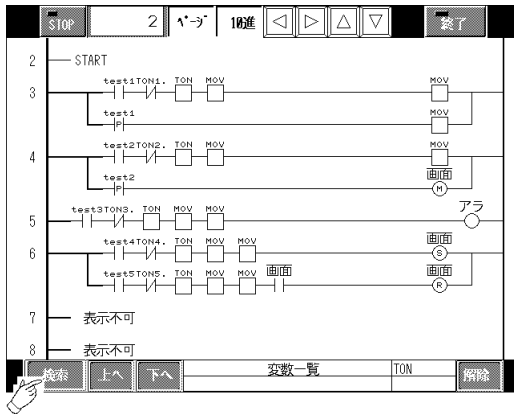
1. ノーマル表示画面の下部にある「命令一覧」ボタンを押し、命令一覧画面を表示させます。



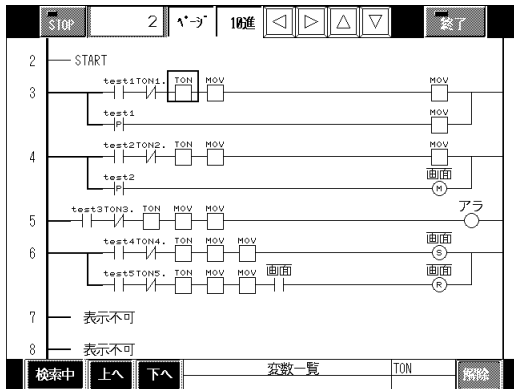
2. 命令一覧画面には、ロジックプログラムで使用可能なすべての命令が表示されています。検索したい変数を選択します。



3. 命令一覧画面で命令を選択すると、自動的にノーマル表示画面に戻ります。画面下部に検索する命令が点滅しているのを確認し、「検索」ボタンを押すと検索を開始します。



4. 検索の結果、該当する命令は、水色の枠で囲まれて表示されます。検索を終了するには、「解除」ボタンを押してください。

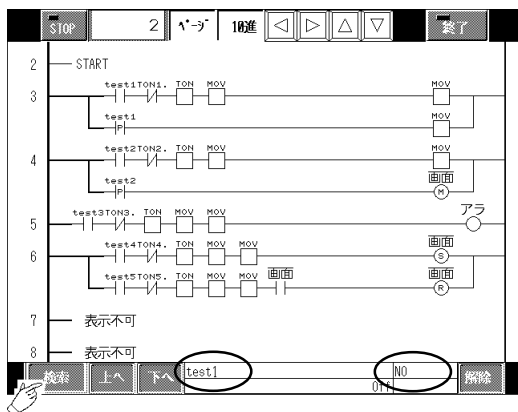


- さらに検索を継続するときは、「上へ」、「下へ」ボタンを押すと次の検索対象命令に移行します。

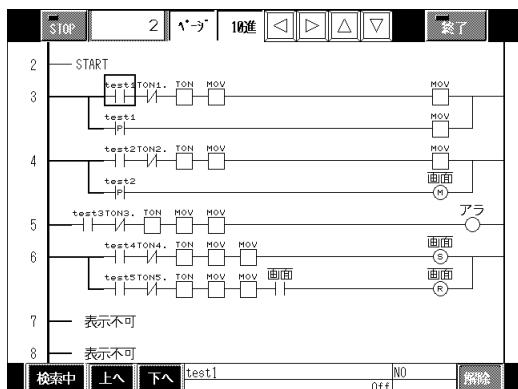
変数と命令からの検索 (AND 検索)

変数検索と命令検索を併せることにより、変数と命令による絞り込んだAND検索が可能となります。検索したい変数と命令を、それぞれの一覧画面より指定して検索します。検索方法は、下記の通りです。

1. どちらか片方 (変数もしくは命令) の一覧画面から検索対象を指定します。
2. ノーマル画面に戻ったとき、もう片方の一覧画面のボタンを押し、一覧画面から検索対象を指定します。
3. 再びノーマル画面に戻ると、画面下部に検索対象である変数と命令が点滅しているのを確認して「検索」ボタンを押すと検索を開始します。下記の画面例では、変数一覧画面から「test1」を選択し、命令一覧画面から「NO」を選択したときのAND検索になります。



4. 検索の結果、該当する命令は、水色の枠で囲まれて表示されます。検索を終了するには、「解除」ボタンを押してください。



- さらに検索を継続するときは、「上へ」、「下へ」ボタンを押すと次の検索対象変数に移行します。

第7章

I/O ドライバ

ここではGLCに内蔵されたI/Oを使用する際に必要なI/Oドライバについて説明します。

7.1 I/O ドライバについて

Pro-Control Editorでは、外部入出力を扱う場合、GLCに装着するI/Oユニットとそれに対応したI/Oドライバが必要になります。I/Oドライバの選択・設定方法は、「Pro-Control Editor オペレーションマニュアル 2.11. I/Oの割り付け」を参照してください。

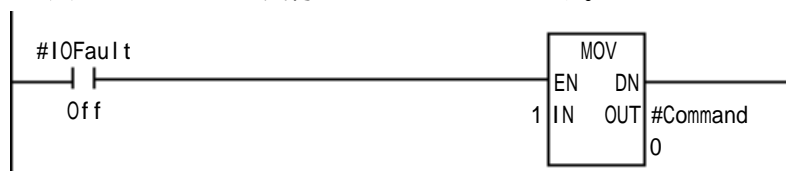
対応ドライバは以下の通りです。

機種	対応ドライバ
GLC100シリーズ	DIOドライバ ユニワイヤI/Fドライバ
GLC300シリーズ	Flex Networkドライバ
GLC2300シリーズ	Flex Networkドライバ
GLC2400シリーズ	
GLC2600シリーズ	



I/Oでエラーが生じた時にコントローラを停止する場合、以下のようなロジックプログラムを作成してください。ただし、異常の検出からロジックプログラム停止まで1スキャンずれることがあります。

下の例では、#IOFaultでI/Oのエラーを検出して#Commandに1を入れてロジックの実行をストップしています。



I/Oにエラーが生じると#IOFaultがONになります。エラーの詳細な情報は#IOStatusで確認することができます。

参照 3.2.20 #IOFault、3.2.22 #Command

7.2 Flex Network ドライバ

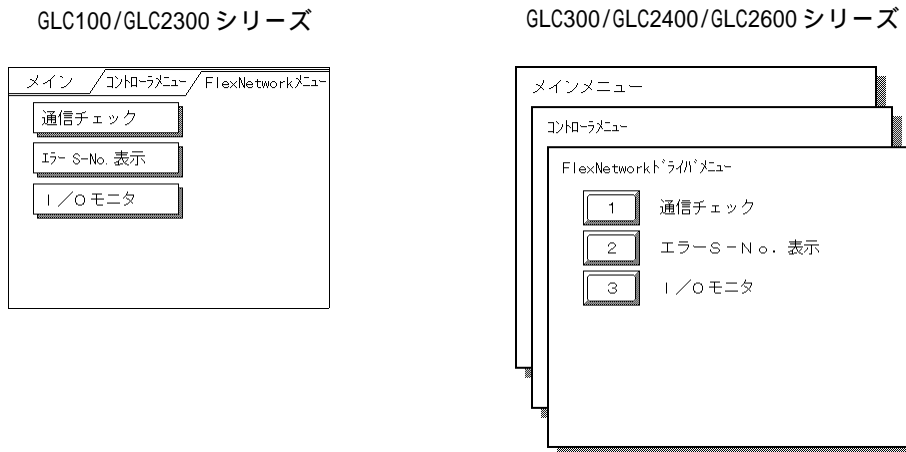
GLCのオフラインモードにあるFlex Network ドライバメニューについて説明します。

Flex Network ドライバメニューを実行するには、あらかじめPro-Control EditorよりFlex Network ドライバをダウンロードしておいてください。また、GLC100、GLC300の場合はFlex Network I/Fユニットが装着されていることも確認してください。GLC2300、GLC2400、GLC2600はFlex Network I/Fが内蔵されています。

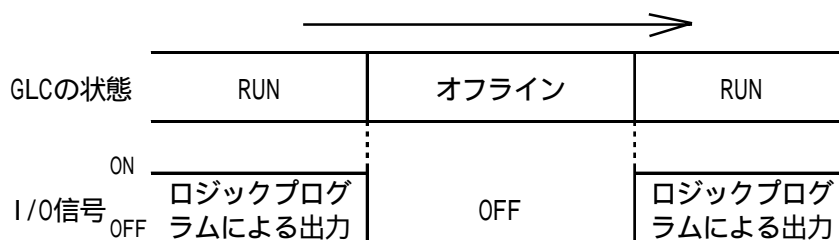
オフラインモードに移る方法は、参照「各GLCシリーズユーザーズマニュアル」(別売)

7.2.1 Flex Network ユニットの自己診断

ここではFlex Network ユニットの自己診断の方法について説明します。GLC本体の自己診断については、参照「GLCシリーズユーザーズマニュアル」(別売)。コントローラメニューで[Flex Network ドライバ]を選択すると、下の画面が表示されます。



- 重要** ・ ロジックプログラムのRUN状態から、オフラインモードへの移行またはリセットした場合のGLCおよびI/O信号の動作は、I/Oユニット側での出力ホールドの設定にかかわらず、以下の通りです。オフラインモードへの移行やリセットは、これらの動作を十分考慮したうえで行ってください。



ただし、リセットの場合は、I/O信号がOFFになるタイミングは不定となります。

7.2.2 通信チェック

接続されている Flex Network I/O ユニットの数と各 I/O ユニットに設定されている S-No. (局番) をチェックします。

通信チェックにより、I/O ユニットについて以下の確認が行えます。

- ・ 接続されている I/O ユニットの確認
- ・ 故障している I/O ユニット (通信部) の確認

通信チェックの手順

[通信チェック] を押すと以下の [通信チェック設定] 画面が表示されます。

[通信速度] は「6Mbps」、「12Mbps」から選択します。通信速度を速くするとノイズの影響を受けやすくなるので、通常は「6Mbps」で使用してください。

GLC100/GLC2300 シリーズ

GLC300/GLC2400/GLC2600 シリーズ

[次頁] ボタンを押すと、以下の [通信チェック] 画面に切り替わります。

[開始] ボタンを押すと、通信チェックが開始されます。

接続されている I/O ユニットの S-No. (局番) が反転表示されます。

GLC100/GLC2300 シリーズ

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	

GLC300/GLC2400/GLC2600 シリーズ

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	

[戻る] ボタンを押すと、[Flex Network メニュー] 画面に戻ります。

7.2.3 エラー S-No.

ロジックプログラムの実行中にエラーコード 841 が発生した場合に、通信回路から外れた I/O ユニットや故障した I/O ユニットの S-No. (局番) をチェックします。

参照 「7.2.5 Flex Network 使用時のトラブルシューティング」

エラー S-No. の手順

[Flex Network ドライバメニュー] 画面で[エラー S-No. 表示]を押すと、以下の [エラー S-No. 表示] 画面が表示され、エラーチェックが開始されます。

接続されている I/O ユニットの S-No. (局番) が表示され、そのうち異常のある I/O ユニットの S-No. が反転表示されます。

GLC100/GLC2300 シリーズ

エラー S-No. 表示						戻る
異常のある S-No. を反転表示。						

GLC300/GLC2400/GLC2600 シリーズ

エラー S-No. 表示							戻る
異常のある S-No. を反転表示。							

7.2.4 I/O モニタ (I/O 工事接続チェック)

GLCとI/Oユニット間の各入出力端子の通信チェックを行います。入力部のチェックは、I/Oユニットの出力信号をGLCでモニタリングし、出力部のチェックは、GLCからI/Oユニットに出力信号を送り、I/Oユニット側でモニタリングしてチェックしてください。

I/O モニタのチェック手順

[コントローラメニュー]画面で[Flex Network ドライバ]を選択し、[Flex Network ドライバメニュー]画面を表示します。

[Flex Network ドライバメニュー]画面で[I/O モニタ]を選択すると、以下の[I/O モニタ設定]画面が表示されます。

GLC100/GLC2300 シリーズ

I/Oモニタ設定		次頁	取消
通信速度 (Mbps)	6		
S-No.	1		
型式	X16TS11		
変数タイプ	ディスクラット		

GLC300/GLC2400/GLC2600 シリーズ

I/Oモニタ設定		次頁	取り消し
通信速度 (Mbps)	6 12		
S-No.	[1]		
型式 (FN→)	X16TS Y08RL Y16SK Y16SC XY08TS AD04AH DA04AH		
変数タイプ	ディスクラット クラット		

・通信速度

「6Mbps」、「12Mbps」から選択します。通信速度を速くするとノイズの影響を受けやすくなるので、通常は「6Mbps」で使用してください。

・S-No. (局番)

「1-63」から選択します。

・型式

「X16TS」、「Y08RL」、「Y16SK」、「Y16SC」、「XY08TS」、「AD04AH」、「DA04AH」の中から選択します。

FN-X32TS、FN-XY16SK、FN-XY16SC、FN-XY32SK、FN-XY32SCについては該当する機種が選択肢にありません。下表の「I/Oモニタで代用する型式」を選択することでチェックすることができます。

I/Oモニタするユニットの型式		FN-X32TS	FN-XY16SK FN-XY16SC	FN-XY32SKS FN-XY32SCS ¹		
I/Oモニタで代用する型式		X16TS	X16TS	Y16SK または Y16SC		
S-No.	+0	入力	0-15	0-15	-	0-7
		出力	-	-	0-15	0-7
	+1	入力	16-31	-	-	8-15
		出力	-	-	-	8-15
	+2	入力	-	-	-	16-23
		出力	-	-	-	16-23
	+3	入力	-	-	-	24-31
		出力	-	-	-	24-31

1 発売に関しましては、弊社までお問い合わせください。

<FN-X32TS の場合>

「X16TS」を代用します。

S-No. に I/O ユニットに設定した局番を指定すると下位 16 ビット(0 ~ 15 ビット目)がモニタできます。

S-No. に I/O ユニットに設定した局番に 1 つ加えた数値を指定すると上位 16 ビット(16 ~ 31 ビット目)がモニタできます。

<FN-XY16SK、FN-XY16SC の場合>

入力は「X16TS」を、出力は「Y16SK」または「Y16SC」を代用します。

入力、出力を一度にモニタすることはできません。

<FN-XY32SK、FN-XY32SC の場合>

「XY08TS」を代用します。

S-No. に I/O ユニットに設定した局番を指定すると 0 ~ 7 ビット目の入出力がモニタできます。

S-No. に I/O ユニットに設定した局番に 1 を加算した数値を指定すると 8 ~ 15 ビット目の入出力がモニタできます。

S-No. に I/O ユニットに設定した局番に 2 を加算した数値を指定すると 16 ~ 23 ビット目の入出力がモニタできます。

S-No. に I/O ユニットに設定した局番に 3 を加算した数値を指定すると 24 ~ 31 ビット目の入出力がモニタできます。

・変数タイプ

「ディスクリート」、「ワード」から選択します。

「FN-AD04AH」、「FN-DA04AH」は「ワード」のみの設定です。

[次頁]ボタンを押すと、次画面が表示されます。

次画面は、各 I/O ユニットの型式によって異なります。ご使用の I/O ユニットの型式をご確認の上、該当する説明をご参照ください。



高速カウンタ、1軸位置決めユニットで、本 I/O モニタを使用することはできません。

< FN-X16TS、FN-XY08TS、FN-Y08RL、FN-Y16SK、FN-Y16SC、FN-XY16SK、FN-XY16SC、FN-X32TS、FN-XY32SK、FN-XY32SC の場合 >

I/O モニタ ([変数タイプ] が「ディスクリット」の場合)

入力部分は入力のあった端子番号が反転表示します。出力部分は端子番号をタッチして反転表示させると出力されます。

[I/O モニタ]画面は選択した[変数タイプ]によって異なります。

GLC100/GLC2300 シリーズ

I/O モニタ		S - N o . 1										戻る			
入力															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
出力															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

GLC300/GLC2400/GLC2600 シリーズ

I/O モニタ		S - N o . 1										戻る			
入力															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
出力															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

1	2	3	4	5	6	7	8	9	0	↑	↓	BS
										←	→	

上記画面は、Flex Network システムの1つの I/O ユニットの最大入出力点数を表示しています。I/O ユニットの機種により、入力点数、出力点数は異なります。0 を先頭に各 I/O ユニットの持つ点数範囲内で使用してください。

入力専用の I/O ユニットの場合は入力部分のみ、出力専用の I/O ユニットの場合は出力部分のみ、入出力混合の I/O ユニットの場合は入力部分、出力部分の両方を使用してください。

I/O モニタ ([変数タイプ] が「整数」の場合)

入力部分は入力のあったデータが表示されます。出力部分はテンキーでデータを入力してください。GLC100、GLC2300 シリーズはデータ表示位置をタッチすると、テンキーパッドが表示されます。データ入力後、[出力] ボタンを押すとデータが出力されます。データ表示は10進数です。

GLC100/GLC2300 シリーズ

I/O モニタ		S - N o . 1										戻る
入力												
0 (0-65535)												
出力												
0 (0-65535) 出力												

GLC300/GLC2400/GLC2600 シリーズ

I/O モニタ		S - N o . 1										戻る
入力												
0 (0-65535)												
出力												
0 (0-65535) 出力												

1	2	3	4	5	6	7	8	9	0	↑	↓	BS
										←	→	

- 重要** ・ 各 I/O ユニットの I/O 点数に応じて、出力できる範囲のデータを入力してください。

I/O点数	入出力範囲
8点	0 ~ 255
16点	0 ~ 65535

[I/O モニタ設定] 画面で選択した「型式」に応じた点数分のデータが I/O ユニットの出力されます。

出力例)

8点出力の I/O ユニットの 8ビットで表現できないデータを設定すると、8ビットを越えるデータは無視されます。

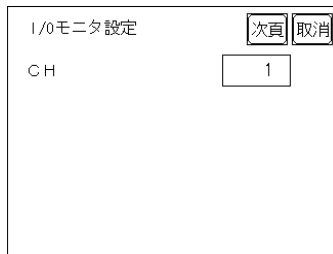


< FN-AD04AH、FN-DA04AH の場合 >

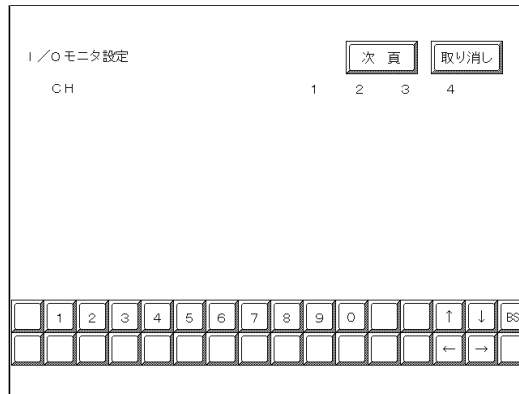
I/O モニタ設定 (チャンネル設定)

チャンネル部分をタッチすると、選択可能な設定内容が順次切り替わります。

GLC100/GLC2300 シリーズ



GLC300/GLC2400/GLC2600 シリーズ



[次頁]ボタンを押すと、次の[I/O モニタ]画面に切り替わります。FN-AD04AH と FN-DA04AH では画面が異なります。

< FN-AD04AH の場合 >

I/O モニタ

入力データを表示します。

GLC100/GLC2300 シリーズ

GLC300/GLC2400/GLC2600 シリーズ

[戻る]ボタンを押すと、[I/O モニタ設定]画面に戻ります。

・ A/D 変換表

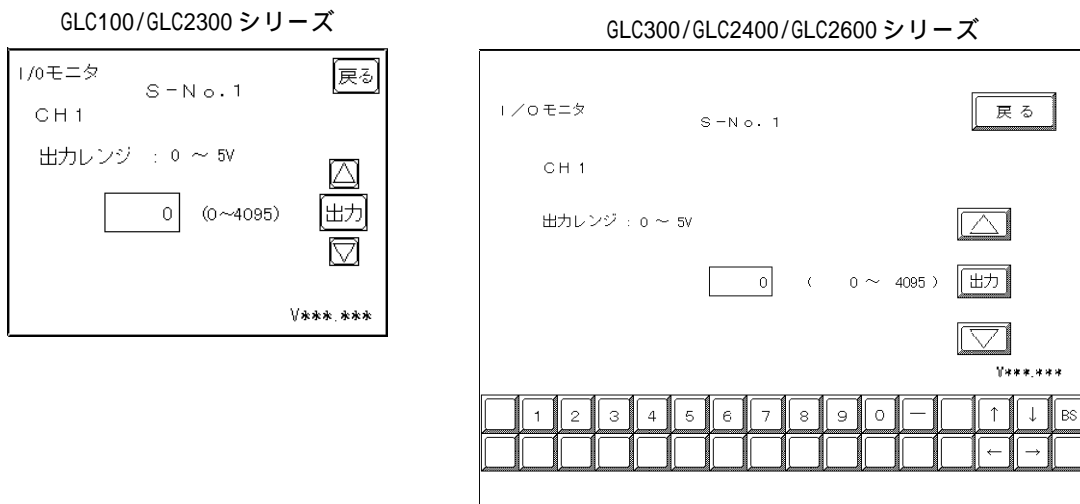
入力レンジ設定	入力範囲
0 ~ 5V	0 ~ 4095
1 ~ 5V	0 ~ 4095
0 ~ 10V	0 ~ 4095
-5 ~ 5V	-2048 ~ 2047
-10 ~ 10V	-2048 ~ 2047
0 ~ 20mA	0 ~ 4095
4 ~ 20mA	0 ~ 4095

- 重要** ・ フィルタタイプ、A/D変換サンプル回数、最大/最小除外設定は、I/Oユニット側に保存されている設定内容で動作します。I/Oユニット側に保存されている設定内容を変更するには、Pro-Control Editor から設定内容を変更し、GLCにロジックプログラムをダウンロードします。その後、ロジックプログラムをRUNモードにして有効になります。
- ・ レンジ切り替えスイッチの設定内容はI/Oユニットの電源投入時のみユニット内部に読み込まれます。レンジ切り替えスイッチの設定を変更する時は、必ずI/Oユニットの電源を一度切ってから再投入してください。
 - ・ I/Oユニット側のレンジ切り替えスイッチの設定内容はロジックプログラムをRUNモードに移行する時に読み込まれます。レンジ切り替えスイッチの設定を変更する時は、ロジックプログラムを一度STOPモードにしてからRUNモードにしてください。レンジが一致していないと、正常にデータが読み込めません。

< FN-DA04AH の場合 >

I/O モニタ

テンキーでデータを入力してください。データ表示位置をタッチすると、テンキーパットが表示されます。データ入力後、[出力]ボタンを押すとデータが出力されます。データ表示は10進数です。



- 重要**
- ・上矢印、下矢印を押すと、加算 / 減算された後に I/O ユニットに出力を行います。
 - ・[戻る]ボタンを押すと、I/O ユニット側で出力ホールド設定にしても、出力がクリアされます。

・ D/A 変換表

出力レンジ設定	出力範囲
0 ~ 5V	0 ~ 4095
1 ~ 5V	0 ~ 4095
0 ~ 10V	0 ~ 4095
-5 ~ 5V	-2048 ~ 2047
-10 ~ 10V	-2048 ~ 2047
0 ~ 20mA	0 ~ 4095
4 ~ 20mA	0 ~ 4095

- 重要**
- ・レンジ切り替えスイッチの設定内容は I/O ユニットの電源投入時のみユニット内部に読み込まれます。レンジ切り替えスイッチの設定を変更する時は、必ず I/O ユニットの電源を一度切ってから再投入してください。
 - ・I/O ユニット側のレンジ切り替えスイッチの設定内容はロジックプログラムを RUN モードに移行する時に読み込まれません。レンジ切り替えスイッチの設定を変更する時は、ロジックプログラムを一度 STOP モードにしてから RUN モードにしてください。レンジが一致していないと、正常にデータが書き込めません。

7.2.5 Flex Network 使用時のトラブルシューティング

ここでは、Flex Network 使用時の異常とその対処方法を示します。参考にしてください。

Flex Network の入力 / 出力異常

Flex Network 使用時の入力 / 出力異常につきましては、「Flex Network ユーザーズマニュアル (別売)」をご覧ください。

エラーコード

I/O エラーは、I/O の読み込み / 書き込みのエラーです。I/O エラーが発生すると、コントローラは #I0Status にエラーコードを書き込みます。発生するエラーの内容と対処方法を説明します。

設定エラー

エラーコード	内容	対処方法
501	I/O ターミナルに割り当てられる内部変数エラー	割り当てられている変数タイプを設定し直してください。
502	出力ターミナルに割り当てられる入力変数エラー	
503	入力ターミナルに割り当てられる出力変数エラー	
504	整数ターミナルに割り当てられるディスクリット変数エラー	
505	ディスクリットターミナルに割り当てられる整数変数エラー	
506	ドライバでサポートされていない変数タイプです	変数タイプを見直してください。
507	ターミナルに変数が割り当てられていません	すべてのターミナルに変数を割り当ててください。
801	ターミナル番号が重複しています	PRW ファイルが破損しているか、PRW ファイルのダウンロード中に障害が発生した可能性があります。
802	S-No. が重複しています	2つ以上の I/O ユニットが同じ S-No. を使用しています。S-No. が重複しないように設定し直してください。
803	S-No. が範囲を超えています	PRW ファイルが破損しているか、PRW ファイルのダウンロード中に障害が発生した可能性があります。
804	アナログユニットで S-No. が範囲を重複しています	2つ以上の I/O ユニットが同じ S-No. を使用しています。アナログユニットは S-No. を 4局占有します。S-No. が重複しないように設定し直してください。
805	高速カウンタユニットで S-No. が範囲を重複しています	2つ以上の I/O ユニットが同じ S-No. を使用しています。高速カウンタユニットは S-No. を 8局占有します。S-No. が重複しないように設定し直してください。
806	1軸位置決めユニットで S-No. が範囲を重複しています	2つ以上の I/O ユニットが同じ S-No. を使用しています。位置決めユニットは S-No. を 4局占有します。S-No. が重複しないように設定し直してください。

初期化エラー

エラーコード	内容	対処方法
821	Flex Network I/Fユニットがありません	内蔵Flex Network I/Fユニットから読み出したID番号が正しくありません。内蔵Flex Network I/Fユニットの故障が考えられます。エラーコードを記録して、(株)デジタル サポートダイヤルまでお問い合わせください。
822	イニシャル異常 イニシャル処理でFlex NetworkドライバとFlex Network I/Fユニットの同期が取れていません	内蔵Flex Network I/Fユニットの故障が考えられます。エラーコードを記録して、(株)デジタル サポートダイヤルまでお問い合わせください。
823	アナログユニット設定異常	通信線に断線がないか、またはI/Oユニットに電源が入っていないか、またはI/Oユニットが故障している恐れがありますので確認してください。

ランタイムエラー

エラーコード	内容	対処方法
841	接続されているI/Oユニットに異常（断線、故障）があります	断線していないか確認してください。参照 Flex Networkユーザーズマニュアル（別売）
842	アナログユニット（A/D変換ユニット）へ入力するセンサの出力信号線の断線	出力信号線に断線が考えられます。センサの出力信号線をチェックしてください。
843	高速カウンタユニットに異常があります	高速カウンタユニットがエラーを検知しました。参照 Flex Network高速カウンタユニットユーザーズマニュアル（別売）
844	高速カウンタユニットのイニシャル異常	通信線に断線がないか、またはI/Oユニットに電源が入っていないか、またはI/Oユニットが故障している恐れがありますので確認してください。
845	高速カウンタユニットとの通信異常	通信線に断線がないか、またはI/Oユニットに電源が入っていないか、またはI/Oユニットが故障している恐れがありますので確認してください。
846	1軸位置決めユニットに異常があります	位置決めユニットがエラーを検知しました。参照 Flex Network1軸位置決めユニットユーザーズマニュアル（別売）
847	1軸位置決めユニットとの通信異常	通信線に断線がないか、またはI/Oユニットに電源が入っていないか、またはI/Oユニットが故障している恐れがありますので確認してください。

内部エラー

エラーコード	内容	対処方法
850 : 859	ドライバエラー システム内に重大なエラーが発生しました	GLCをリセットしてください。その後もエラーコードが表示される場合は、周辺環境によりエラーが誘発されているか、GLC本体の異常が考えられます。エラーコードを記録して、サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。

7.3 DIO ドライバ

GLCのオフラインモードにあるDIOメニューについて説明します。DIOメニューを実行するには、あらかじめPro-Control EditorよりDIOドライバをダウンロードしておく必要があります。また、DIOユニットが装着されていることを確認してください。

オフラインモードに移る方法は、「GLCシリーズユーザーズマニュアル」(別売)を参照してください。

7.3.1 DIOの自己診断

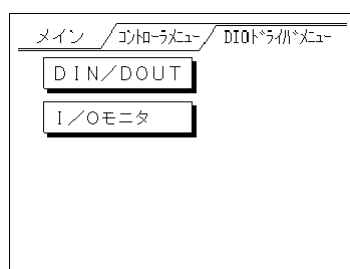
ここではDIOの自己診断の方法について説明します。

GLC本体の自己診断については、
参照 「GLCシリーズユーザーズマニュアル」(別売)

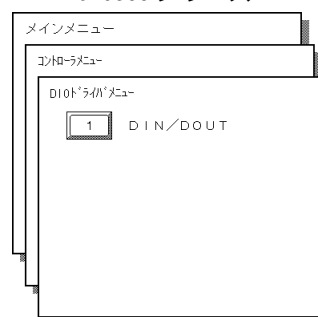
コントローラメニューで[DIOドライバ]を選択すると、下の画面が表示されます。

< DIOドライバを選択する場合 >

GLC100 シリーズ

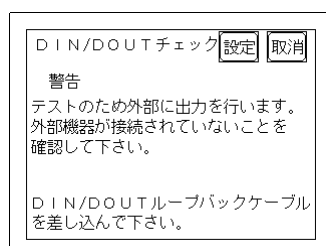


GLC300 シリーズ

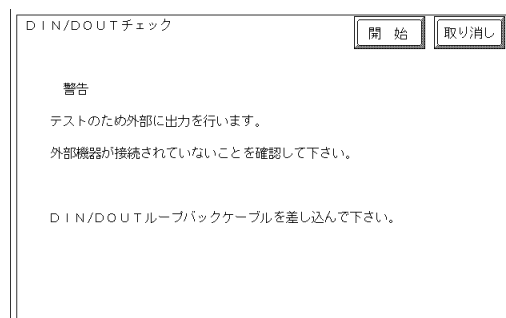


[DIN/DOUT]を押すと下の画面が表示されます。

GLC100 シリーズ



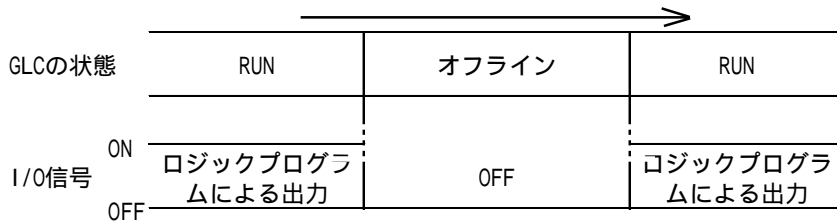
GLC300 シリーズ



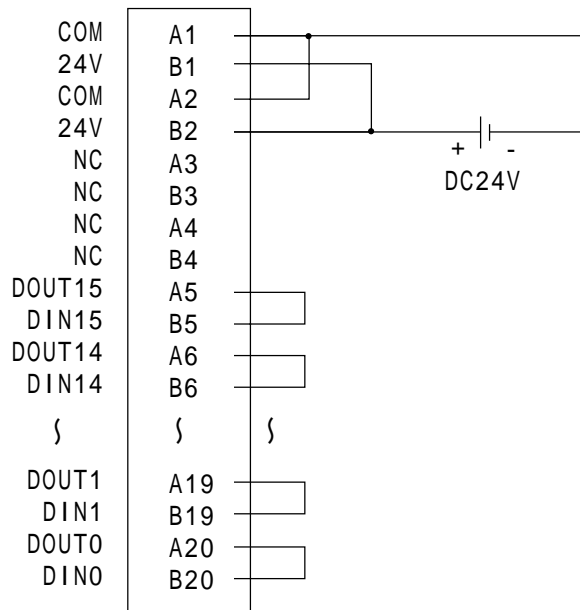
[設定]/[開始]を押すとチェックが開始されます。

このチェックは、出力ユニットから出た信号を入力ユニットで取り込みます。
 チェックを行うにはDIN/DOOUT ループバックケーブルを接続してください。

重要 ロジックプログラムのRUN 状態からオフラインモードへの移行、またはリセットした場合、I/O の信号はOFF されることがあります。I/O の信号がOFF されることを十分考慮して行ってください。
 ただし、リセットの場合は、I/O 信号がOFF になるタイミングは不定となります。



DIN/DOOUT ループバックケーブルの配線は、以下のとおりです。(シンクタイプの場合)



< 推奨品 >

接続方法	メーカー	型番
ハンダ付けタイプ	富士通(株)	FCN-361J040-AU (コネクタ) FCN-360C040-B (カバー)
	(株)デジタル	FN-IFCN01 ^{*1} (コネクタ/カバー)
圧着タイプ	富士通(株)	FCN-363J040 FCN-363J-AU/S FCN-360C0404-B
端子台ユニットタイプ	三菱電機(株)	A6TBX36 (端子台ユニット) AC**TB (ケーブル) (* **は、ケーブル長を表します。)
	横河電気(株)	TA40-ON

*1 製品自体は富士通(株)製と同じです。

7.3.2 I/O モニタ (I/O 工事接続チェック)

DIO ドライバメニューで[I/O モニタ]を選択すると、下の画面が表示されます。

< I/O モニタを選択する場合 >

GLC100 シリーズ

I/Oモニタ設定	<input type="button" value="実行"/>	<input type="button" value="取消"/>
モジュール番号 (No.0-1)	<input type="text" value="0"/>	
入力 変数タイプ	<input type="text" value="ディスクリート"/>	
出力 変数タイプ	<input type="text" value="ワード"/>	

GLC300 シリーズ

I/Oモニタ設定	<input type="button" value="実行"/>	<input type="button" value="取り消し"/>
モジュール番号 (No.0-1)	<input type="text" value="0"/>	<input type="text" value="1"/>
入力 変数タイプ	<input type="text" value="ディスクリート"/>	<input type="text" value="ワード"/>
出力 変数タイプ	<input type="text" value="ディスクリート"/>	<input type="text" value="ワード"/>

<input type="button" value="1"/>	<input type="button" value="2"/>	<input type="button" value="3"/>	<input type="button" value="4"/>	<input type="button" value="5"/>	<input type="button" value="6"/>	<input type="button" value="7"/>	<input type="button" value="8"/>	<input type="button" value="9"/>	<input type="button" value="0"/>	<input type="button" value="↑"/>	<input type="button" value="↓"/>	<input type="button" value="BS"/>
<input type="button" value="←"/>	<input type="button" value="→"/>											

[モジュール番号]は「0」、「1」より選択します。(0、1はそれぞれGLC側のユニット、外側のユニットを表しています。)

[入力変数タイプ]は「ディスクリート」、「ワード」より選択します。

[出力変数タイプ]は「ディスクリート」、「ワード」より選択します。

例えば、[I/Oモニタ設定]画面で「入力変数タイプ:ディスクリート」、「出力変数タイプ:ワード」を選択する場合、画面右上の実行ボタンをタッチすると確定し、[I/Oモニタ]画面が表示されます。

GLC100 シリーズ

I/Oモニタ	モジュール番号	<input type="button" value="戻る"/>						
入力	<input type="text" value="0"/>	<input type="text" value="1"/>	<input type="text" value="2"/>	<input type="text" value="3"/>	<input type="text" value="4"/>	<input type="text" value="5"/>	<input type="text" value="6"/>	<input type="text" value="7"/>
	<input type="text" value="8"/>	<input type="text" value="9"/>	<input type="text" value="10"/>	<input type="text" value="11"/>	<input type="text" value="12"/>	<input type="text" value="13"/>	<input type="text" value="14"/>	<input type="text" value="15"/>
出力	<input type="text" value="1234"/>	(0-65535)	<input type="button" value="出力"/>					

GLC300 シリーズ

I/Oモニタ	モジュール番号	<input type="button" value="戻る"/>						
入力	<input type="text" value="0"/>	<input type="text" value="1"/>	<input type="text" value="2"/>	<input type="text" value="3"/>	<input type="text" value="4"/>	<input type="text" value="5"/>	<input type="text" value="6"/>	<input type="text" value="7"/>
	<input type="text" value="8"/>	<input type="text" value="9"/>	<input type="text" value="10"/>	<input type="text" value="11"/>	<input type="text" value="12"/>	<input type="text" value="13"/>	<input type="text" value="14"/>	<input type="text" value="15"/>
出力	<input type="text" value="1234"/>	(0-65535)	<input type="button" value="出力"/>					

<input type="button" value="1"/>	<input type="button" value="2"/>	<input type="button" value="3"/>	<input type="button" value="4"/>	<input type="button" value="5"/>	<input type="button" value="6"/>	<input type="button" value="7"/>	<input type="button" value="8"/>	<input type="button" value="9"/>	<input type="button" value="0"/>	<input type="button" value="↑"/>	<input type="button" value="↓"/>	<input type="button" value="BS"/>
<input type="button" value="←"/>	<input type="button" value="→"/>											

入力変数タイプ[ディスクリート]の場合、入力のあった端子番号が反転表示します。

出力変数タイプ[ワード]の場合テンキーでデータを入力してください。GLC100シリーズの場合はデータ表示位置をタッチすると、テンキーパットが表示されます。

データを入力後、「出力ボタン」を押すとデータが出力されます。データ表示は10進数です。

7.3.3 DIO 使用時のトラブルシューティング

ここでは、DIO 使用時の異常とその対処方法を示します。参考にしてください。

DIO の入力異常

異常現象	推定原因	対処方法
入力モニタランプは点灯するが、まったく入力できない。	DIOの不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	I/O使用可が未設定	I/O使用可の設定をする
	プログラムの不良	プログラムの修正
入力モニタランプが消灯し、まったく入力できない。	DIOの不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	入力コモン線の配線ミス	コモン線の配線チェック コモン線の断線チェック コモン端子の緩みチェック
	外部入力電圧不良	定格電圧を供給する
	コネクタの接触不良	コネクタを確実にネジで取り付ける
入力すべてがOFFしない。	DIOの不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
特定の入力がONしない。	DIOの不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	プログラムの不良	プログラムの修正
	入力線の配線ミス	入力線の配線チェック 入力線の断線チェック 入力端子の緩みチェック
	外部接続機器の異常	外部接続機器の交換
	入力のON時間が短い	入力のON時間を長くする
特定の入力がOFFしない。	DIOの不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	プログラムの不良	プログラムの修正
入力エリアが不規則にON、OFFする。	外部入力電圧不良	定格電圧を供給する
	端子ネジの緩み	ネジ増し締め
	プログラムの不良	プログラムの修正
	コネクタの接触不良	コネクタを確実にネジで取り付ける
	ノイズによる誤動作	ノイズ対策をする サージキラーの取り付け シールドケーブルの使用

D10 の出力異常

異常現象	推定原因	対処方法
出力モニタランプは点灯するが、まったく出力できない。	D10の不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	出力コモン線の配線ミス	コモン線の配線チェック コモン線の断線チェック コモン端子の緩みチェック
	負荷電源不良	定格電圧を供給する
	コネクタの接触不良	コネクタを確実にネジで取り付ける
出力モニタランプが消灯し、まったく出力できない。	D10の不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	プログラムの不良 出力エリアをすべてOFFにしている	プログラムの修正
	I/O使用可が未設定	I/O使用可の設定をする
出力すべてがOFFしない。	D10の不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
特定の出力がONしない。	D10の不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	出力線の配線ミス	出力線の配線チェック 出力線の断線チェック 出力端子の緩みチェック
	外部接続機器の異常	外部接続機器の交換
特定の出力がOFFしない。	D10の不良	サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
	漏れ電流、残留電流による復帰不良	外部機器の設計変更 ダミー抵抗の追加など
出力エリアが不規則にON、OFFする。	負荷電源不良	定格電圧を供給する
	端子ネジの緩み	ネジ増し締め
	プログラムの不良 出力命令が重複している	プログラムの修正
	コネクタの接触不良	コネクタを確実にネジで取り付ける
	ノイズによる誤動作	ノイズ対策をする サージキラーの取り付け シールドケーブルの使用

エラーコード

I/Oエラーは、I/Oの読み込み / 書き込みのエラーです。I/Oエラーが発生すると、コントローラは#10Statusにエラーコードを書き込みます。ロジックプログラムの実行は続けられます。発生するエラーの内容と対処方法を説明します。

設定エラー

エラーコード	内容	対処方法
501	I/Oターミナルに割り当てられる内部変数エラー	割り当てられている変数タイプを設定し直してください。
502	出力ターミナルに割り当てられる入力変数エラー	
503	入力ターミナルに割り当てられる出力変数エラー	
504	アナログターミナルに割り当てられるディスクリット変数エラー	
505	ディスクリットターミナルに割り当てられる整数変数エラー	
506	ドライバでサポートされていない変数タイプです	変数タイプを見直してください。
801	ターミナル番号が重複しています	2つ以上のターミナルが同じターミナル番号を使用しています。転送に失敗した恐れがあります。プロジェクトファイルを再ダウンロードしてください。
802	重複モジュールがあります	モジュール番号が重複しています。2つ以上のモジュールを設定しないでください。
803	モジュール番号が1を越えています	モジュール番号を0に設定してください。
804	モジュール番号が1から始まっています	モジュール番号を0に設定してください。

ランタイムエラー

エラーコード	内容	対処方法
840	モジュール 0 読み出しデータが不正です。GLCに近い方のD10ユニットから2回連続して読み出した値が異なりました。	入力信号のON時間を長くしてください。
841	モジュール 1 読み出しデータが不正です。GLCに遠い方のD10ユニットから2回連続して読み出した値が異なりました。	入力信号のON時間を長くしてください。
842	モジュール 0 出力データが不正です。内部ループバックチェックにてGLCに近い方のD10ユニットから出力データの不正を検知しました。	ノイズ等の影響がないかご確認ください。
843	モジュール 1 出力データが不正です。内部ループバックチェックにてGLCに近い方のD10ユニットから出力データの不正を検知しました。	ノイズ等の影響がないかご確認ください。

初期化エラー

エラーコード	内容	対処方法
821	WLLファイルに記述されているD10ユニットの数と実際に接続されているD10ユニットの数が一致しません。	接続するD10ユニットの数を設定し直してください。
822	モジュール 0がありません。 GLCに近い方のD10ユニットがありません。	ユニットが確実に装着されていることを確認し、D10ドライバの設定を見直してください。
823	モジュール 1がありません。 GLCに遠い方のD10ユニットがありません。	ユニットが確実に装着されていることを確認し、D10ドライバの設定を見直してください。

内部エラー

エラーコード	内容	対処方法
850 ⋮ 864	ドライバエラー システム内に重大なエラーが発生しました。	エラーコードを記録して、サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。

7.4 ユニワイヤ I/F ドライバ

GLCのオフラインモードにあるユニワイヤドライバメニューについて説明します。ユニワイヤドライバメニューを実行するには、予めPro-Control Editorよりユニワイヤ I/F ドライバをダウンロードしておいてください。また、ユニワイヤ拡張 I/Fユニットが装着されていることを確認してください。

オフラインモードに移る方法は、「GLCシリーズユーザーズマニュアル」(別売)を参照してください。

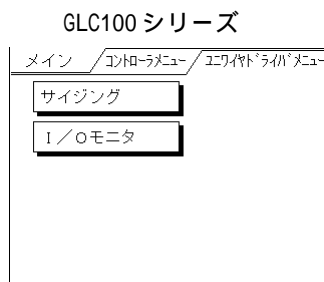
7.4.1 ユニワイヤ拡張 I/Fユニットの自己診断

ここではユニワイヤ拡張 I/Fユニットの自己診断の方法について説明します。

GLC本体の自己診断については、
参照「GLCシリーズユーザーズマニュアル」(別売)

コントローラメニューで[ユニワイヤドライバ]を選択すると、下の画面が表示されます。

<サイジングを選択する場合>



[サイジング]を押すと下の画面が表示されます。

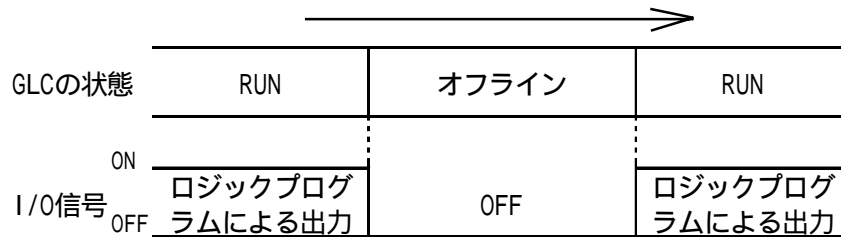


[開始]を押すとサイジングが開始されます。

サイジングとは接続されているターミナルの ID(ターミナル番号)を GLC に記憶させることです。この処理により伝送ラインの断線を検知できるようになります。

- ・ サイジングを行うには、ユニワイヤ拡張 I/F ユニット側面の CERR LED が点滅している必要があります。点滅しない場合は、伝送点数・伝送距離を再設定し再起動してください。
- ・ エラーコードはサイジングを行うと消去されます。
- ・ ID(ターミナル番号)が記憶されていない場合は、断線検知は行われません。

重要 ロジックプログラムのRUN状態からオフラインモードへの移行や、またはリセットした場合、I/Oの信号はOFFされることがあります。I/Oの信号がOFFされることを十分考慮して行ってください。



7.4.2 I/O モニタ (I/O 工事接続チェック)

ユニワイヤドライバメニューで[I/Oモニタ]を選択すると、下の画面が表示されます。

< I/O モニタを選択する場合 >

GLC100 シリーズ

I/Oモニタ設定	<input type="button" value="実行"/>	<input type="button" value="取消"/>
エリア番号 (No.0-15)	<input type="text" value="0"/>	
伝送距離 (m)	<input type="text" value="200"/>	
伝送点数 (点数)	<input type="text" value="128"/>	
入力/出力	<input type="text" value="入力"/>	
変数タイプ	<input type="text" value="ディスクリット"/>	

GLC300 シリーズ

I/Oモニタ設定	<input type="button" value="実行"/>	<input type="button" value="取り消し"/>
エリア番号 (No.0-15)	<input type="text" value="[0]"/>	
伝送距離 (m)	<input type="text" value="200"/>	<input type="text" value="500"/>
伝送点数 (点数)	<input type="text" value="128"/>	<input type="text" value="256"/>
入力/出力	<input type="text" value="入力"/>	<input type="text" value="出力"/>
変数タイプ	<input type="text" value="ディスクリット"/>	<input type="text" value="ワード"/>

1 2 3 4 5 6 7 8 9 0 ↑ ↓ BS ← →

[エリア番号]は「0-15」より選択します。

[伝送距離]は「200m」、「500m」、「1000m」から選択します。

[伝送点数]は「128点」、「256点」より選択します。

[入力/出力]は「入力」、「出力」より選択します。

[変数タイプ]は「ディスクリット」、「ワード」より選択します。

各項目を入力、または設定した後、画面右上の実行ボタンをタッチすると確定し、以下の画面が表示されます。

「I/O モニタ設定」画面で「入力」を選択する場合

GLC100 シリーズ

I/Oモニタ エリア番号 0 終了

入力

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

I/Oモニタ エリア番号 0 戻る

入力

1234 (0-65535)

GLC300 シリーズ

I/Oモニタ エリア番号 0 戻る

入力

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

I/Oモニタ エリア番号 0 戻る

入力

1234 (0-65535)

「I/O モニタ設定」画面で「出力」を選択する場合

GLC100 シリーズ

I/Oモニタ エリア番号 0 終了

出力

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

I/Oモニタ エリア番号 0 戻る

出力

1234 (0-65535) 出力

GLC300 シリーズ

I/Oモニタ エリア番号 0 戻る

出力

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

1	2	3	4	5	6	7	8	9	0			↑	↓	BS
												←	→	

I/Oモニタ エリア番号 0 戻る

出力

1234 (0-65535) 出力

1	2	3	4	5	6	7	8	9	0			↑	↓	BS
												←	→	

入力変数タイプ[ディスクリット]の場合、入力のあった端子番号が反転表示します。
 出力変数タイプ[ワード]の場合、テンキーでデータを入力してください。GLC100シリーズはデータ表示位置をタッチすると、テンキーパットが表示されます。
 データ入力後、「出力」ボタンを押すとデータを出力されます。データ表示は10進数です。

7.4.3 ユニワイヤ拡張 I/F ユニット使用時のトラブルシューティング

ここでは、ユニワイヤ拡張 I/F ユニット使用時の異常とその対処方法を示します。参考にしてください。

ユニワイヤ拡張 I/F ユニットの入力 / 出力異常

ユニワイヤ拡張 I/F ユニット使用時の入力 / 出力異常につきましては、ユニワイヤ拡張 I/F ユニットのユーザーズマニュアルをご覧ください。

エラーコード

I/Oエラーは、I/Oの読み込み / 書き込みのエラーです。I/Oエラーが発生すると、コントローラは#I0Statusにエラーコードを書き込みます。ロジックプログラムの実行は続けられます。ここではユニワイヤ拡張 I/F ユニットを接続したときに発生するエラーの内容と対処方法を説明します。

設定エラー

エラーコード	内容	対処方法
501	I/Oターミナルに割り当てられる内部変数エラー	割り当てられている変数タイプを設定し直してください。
502	出力ターミナルに割り当てられる入力変数エラー	
503	入力ターミナルに割り当てられる出力変数エラー	
504	アナログターミナルに割り当てられるディスクリート変数エラー	
505	ディスクリートターミナルに割り当てられる整数変数エラー	
506	ドライバでサポートされていない変数タイプです	変数タイプを見直してください。
801	ターミナル番号が重複しています	2つ以上のターミナルが同じターミナル番号を使用しています。転送に失敗した恐れがあります。プロジェクトファイルを再ダウンロードしてください。
802	エリア番号が重複しています	2つ以上のエリアが同じエリア番号を使用しています。エリア番号が重複しないように設定し直してください。
803	エリア番号が範囲を超えています	伝送点数128点の場合は0～7Fまで、256点の場合は0～FFまで入出力エリアを設定できます。範囲内になるように設定し直してください。
804	スキャンタイム設定エラー	伝送距離・伝送点数による適切なスキャンタイムに設定し直してください。

初期化エラー

エラーコード	内容	対処方法
821	ユニワイヤ拡張I/Fユニットがありません	ユニワイヤ拡張I/Fユニットから読み出したID番号が不正です。ユニワイヤ拡張I/Fユニットがないときに表示されます。
822	イニシャル異常 イニシャル処理でユニワイヤI/Fドライバとユニワイヤ拡張I/Fユニットの同期が取れていません	ハード異常が考えられます。 <u>参照</u> ユニワイヤ拡張I/Fインターフェイスのユーザーズマニュアル
823	内蔵メモリ異常 ユニワイヤ拡張I/Fユニットのデュアルポートメモリのリード/ライトチェックで異常が発生しました	設定を見直してプロジェクトファイルを再転送してください。問題が解決されない場合は、ハード異常が考えられます。

ランタイムエラー

エラーコード	内容	対処方法
841	ユニワイヤ拡張I/Fユニットがリセットされました	GLCを再起動してください。
842	I/Oリフレッシュ異常	設定を見直してプロジェクトファイルを再転送してください。問題が解決されない場合は、ハード異常が考えられます。
843	D-G間の短絡	D-G間が短絡していないか確認してください。
844	D、Gラインの断線	D、Gラインが断線していないか確認してください。
845	D-24V間に電源が供給されていない	電源が供給されているか確認してください。

内部エラー

エラーコード	内容	対処方法
850 : 864	ドライバエラー システム内に重大なエラーが発生しました。	エラーコードを記録して、(株)デジタル サポートダイヤルまでお問い合わせください。

第8章

エラーと異常処理

8.1 エラーメッセージ

ここでは、GLCの左下に表示されるエラーメッセージについて説明します。ここで説明するエラーメッセージはPro-Control Editorに関するものだけです。

その他のエラーメッセージについては

参照 「GLCシリーズユーザーズマニュアル」(別売)

エラーメッセージ	原因	対処方法
"Invalid ladder file"	GLCにロジックプログラムファイルがダウンロードされていないか、またはGLC上のロジックプログラムファイルが壊れています。	Pro-Control Editorからプロジェクトファイルをダウンロードし直してください。
"Fatal Error: Drive check Failed"	GLC上のI/Oドライバが不正です。	ロジックプログラムファイルに設定されているI/OドライバとGLCにインストールしたI/Oドライバが同じものか確認してください。
"Global Data Area Too Small"	ダウンロード中にファイルが壊れた可能性があります。	プロジェクトファイルを再ダウンロードしてください。解決されない場合は、サポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
"Can't Set Priority"	GLCのシステムファイルに問題があります。ダウンロード中にファイルが壊れた可能性があります。	プロジェクトファイルを再ダウンロードしてください。
"Exception nnn:[mmm:ooo]"	ロジックプログラムに重大なエラーがあります。	エラーの内容をサポートダイヤル(本マニュアルP.4に記載)までお問い合わせください。
"SRAM checksum error"	SRAMに保存されているプロジェクトファイルが壊れています。(GLC2000シリーズのみ)	Pro-Control Editorからプロジェクトファイルをダウンロードし直してください。
"SRAM data broken"	SRAMのバックアップ用バッテリーが切れた可能性があります。このメッセージは警告メッセージです。(GLC2000シリーズのみ)	FEPROMのプロジェクトファイルから実行します。オンラインエディットでロジックプログラムを変更していないか確認してください。
"Watchdog Error"	コンスタントスキャンタイムがウォッチドッグタイムより長くなっています。	ウォッチドッグタイムをコンスタントスキャンタイムより長く設定してください。

エラーメッセージ	原因	対処方法
"Bad Var: xxx"	変数名「xxx」が見つかりません。 次の二つの原因が考えられます。 ・ロジックプログラムファイルがダウンロードされていない。 ・画面上でロジックプログラムファイルに存在しない変数を使用している。	一度ロジックプログラムを保存し、再度画面データをダウンロードしてください。
"Bad Array: xxx"	画面ファイルの要素数とロジックプログラムファイルの配列変数の要素数が違います。	プロジェクトファイルを再ダウンロードしてください。
"Bad Type xxx"	ロジックプログラムの変数「xxx」の変数タイプと画面の変数タイプとが異なります。	プロジェクトファイルを再ダウンロードしてください。
"Unknown register type"	変数タイプが存在しません。	プロジェクトファイルを再ダウンロードしてください。
"Register is missing"	書き込もうとした変数が見つかりません。	
"S100 file index is out of range"	読み出そうとした変数が見つかりません。	
"Too many entries in the S100 file"	変数の数が多すぎます。 使用できる変数は2048までです。	
"S100 file is missing"	S100ファイル(変数格納ファイル)が見つかりません。	
"Over Compile count MAX"	使用している部品が多すぎます。	使用している部品を減らして、プロジェクトファイルを再ダウンロードしてください。
"Exception 65532 [xxxx:xxx]" "Exception 65533 [xxxx:xxx]" "Exception 65534 [xxxx:xxx]" "Exception 65535 [xxxx:xxx]"	GLC上のヒープメモリが不足しています。プログラム、変数を格納するメモリは足りていますが、ロジックプログラムを実行するメモリが不足しています。	ロジックプログラム、変数またはラベルを減らしてEditor上でGLCを再セットアップしてください。配列変数の場合、要素数を減らすことも有効です。また、変数名、ラベル名を短くすることも有効です。
"No Backup logic program in FEPR0M"	モニタリングモード中にロジックプログラムを編集し、ダウンロードした為、本来の保存場所であるFEPR0MでなくSRAMに保存されています。	オフラインから以下の手順でSRAMからFEPR0Mへロジックプログラムをコピーします。 「初期設定/動作環境の設定」 「コントローラの設定/FEPR0Mへのコピー」

8.2 エラーコード

ここではエラーが発生した際に書き込まれる、#FaultCodeのエラーコードについて説明します。

エラーコード	程度	原因
0	正常	エラーはありません。
1	マイナー	算術命令の結果、または実数から整数への変換結果がオーバーフローしました。
2	メジャー	配列の領域を越えて参照されました。
3	メジャー	整数（32ビット）の範囲を超えてビットが参照されました。
4	メジャー	スタックがオーバーフローしました。
5	メジャー	不正な命令コードを使用しています。
6	-	システムで予約
7	メジャー	スキャンタイムがウォッチドッグタイムを越えました。
8	-	システムで予約
9	メジャー	ソフトウェアのエラーです。 場合によっては、システムを再起動する必要があります。
10	-	システムで予約
11	-	システムで予約
12	マイナー	BCD/BIN変換エラー
13	マイナー	ENCO/DECOエラー ¹
14	-	システムで予約
15	マイナー	バックアップメモリ(SRAM)のロジックプログラムが壊れています。FEEPROMのロジックプログラムが実行されます。 ¹



・「メジャー異常」

メジャー異常が発生すると、コントローラはすぐにロジックプログラムを停止します。メジャー異常発生時、本体LEDは赤色に点灯し、断続的にブザーが鳴ります。

・「マイナー異常」

マイナー異常はロジックプログラムの実行が可能なエラーです。

¹ GLC2000シリーズでのみ発生するエラーです。

8.3 プログラムの動作異常

ここではPro-Control Editorのロジックプログラムの動作異常について説明します。

異常現象	推定原因	対処方法
コントロールメモリの電源断保持エリアが保持されない	電池異常	本機交換
	メモリ異常	本機交換
プログラムが正常に動作しない	プログラムの転送ミス	GP-PRO/PB で、プロジェクトファイルを再ダウンロードする。 参照 Pro-Control Editorオペレーションマニュアル 5.2 GLCへのダウンロード
STOPモードになっているのに、I/O出力されている。	出力データRUN/STOP切り換え時、I/O出力の保持が有効になっている。	当機能を無効にする。 参照 Pro-Control Editorオンラインヘルプ
RUNモードになるが、すぐにSTOPモードになってしまう。	命令実行異常などが発生している。または、メジャー異常が発生している。	システム変数 #FaultCodeの内容を確認し、プログラムを修正する。参照 Pro-Control Editorオペレーションマニュアル 3.4 システム変数の表示 プログラムを見直し、システム変数 #Commandに書き込みがないか確認し、プログラムを修正する。 参照 3.2.18 #FaultCode、3.2.22 #Command
Pro-Control Editorでモニタリングモードに入れない。 Pro-Control Editorからロジックプログラムファイルをダウンロードできない。 GP-PRO/PB からプロジェクトファイルがダウンロードできない。	GP-PRO/PB からのプロジェクトファイルのダウンロード中に、転送ケーブルが抜けた、GLCやパソコンの電源が落ちたなどの電氣的ノイズによりGLC内のシステムファイルが壊れた可能性があります。	転送ケーブルが抜けていないか、ノイズなどの影響がないか確認する。 解決しない場合は(株)デジタル・サポートダイヤルまでご連絡ください。
I/O入出力ができない。	I/O使用可 ¹ の設定ができていない。	I/O使用可の設定をしてください。

1 I/O使用可とはGLC本体や、I/Oユニットへの入出力を可能にする動作です。GLCでは、ロジックプログラムのダウンロードを行った後、GLCを運転状態にただけでは外部I/O機器の入出力を行うことはできません(安全のため、デフォルトではI/O使用可は設定されていません。)

I/O入出力を行う場合、あらかじめI/O使用可の設定が必要です。設定方法については「Pro-Control Editor オペレーションマニュアル 3.1. コントローラの設定、3.2. コントローラのRUN/STOP」を参照してください。

索引

D

- DIO ドライバ 7-13
- DIO の自己診断 7-13

F

- Flex Network ドライバ 7-2
- Flex Network の自己診断 7-2, 7-3

G

- GLC 9
- GLC と PLC のデータ共有時の注意点 3-3

I

- I/O ドライバ 7-1
- I/O モニタ 7-5, 7-15

L

- LS 2-2, 5-2
- LSS 2-2, 5-2
- LS エリア 5-1
- LS エリアリフレッシュ 5-1
- LS エリアリフレッシュ実行時の注意点 ... 3-3

P

- PLC 9
- Pro-Control 9

S

- STOP モード 1-1

ア

- アクセス 2-6
- アクセス単位 2-7
- 安全に関する注意表記 9

イ

- 異常処理 8-1
- インターナル 2-5

エ

- エラー 6-1, 7-1, 8-1
- エラーコード 7-11, 7-18, 7-23, 8-3

- エラーと異常処理 6-1
- エラーメッセージ 8-1
- 演算命令 4-3

オ

- オフラインメニュー 7-2, 7-20
- オフラインモード 7-2, 7-20

カ

- カウンタ 2-4
- カウンタ命令 4-3
- 画面作成ソフト 9

ク

- グローバル 2-5

コ

- コンスタント
- スキャンモード 1-1, 1-5, 1-6
- コントローラ 9
- コントローラ機能 1-2

サ

- サイジング 7-20, 7-21

シ

- システム変数 3-1
- システム変数詳細 3-3
- 実数 2-3
- 実数配列へのアクセス 2-9
- 実数配列 2-9
- 実数変数 2-3
- 修飾語 2-6
- 使用上の注意 10
- 商標権 2

ス

- スキャンタイムの調整 1-4

セ

- 整数 2-3
- 整数・整数配列へのアクセス 2-7
- 整数変数 2-3, 2-3

説明のための表記 9

タ

ターミナル 7-20
 タイマ 2-4
 タイマ命令 4-3

テ

データ共有 5-3, 5-4
 ディスクの取り扱いについて 10
 ディスクリット配列へのアクセス 2-6
 ディスクリット配列変数 2-6
 ディスクリット変数 2-3, 2-3
 デバイスアドレス 2-1
 転送命令 4-2

ト

トラブルシューティング 7-11, 7-16

ニ

入力 / 出力 2-5

ハ

パーセントスキャンモード 1-1, 1-5, 1-7
 バイト 2-7
 配列 2-6
 配列への間接アクセス 2-9

ヒ

ビット 2-7
 ビット操作命令 4-1
 表記のルール 9

ヘ

変換命令 4-4, 4-5
 変数 2-1
 変数タイプ 2-1, 2-3
 変数の属性 2-5
 変数へのアクセス 2-6
 変数名 2-1
 変数名の制限 2-2

ホ

保持 2-5

マ

マニュアルの読み方 3

メ

命令 4-1
 命令 4-6
 命令詳細 4-6

ユ

ユニワイヤ拡張 I/F ユニット 7-20
 ユニワイヤ拡張 I/F ユニットの異常 6-6, 6-7, 7-23
 ユニワイヤ拡張 I/F ユニットの自己診断 7-20
 ユニワイヤ拡張 I/F ユニットの入力 / 出力異常 6-6, 6-7, 7-23

ヨ

要素 2-6
 要素番号 2-6

ロ

ロジックプログラム開発ソフト 1
 論理演算命令 4-2

ワ

ワード 2-7