# 型番関連

例：FX3U-□□MR/〇S

□□ 接点数。**入力と出力の合計数**なので注意。

MR リレー出力

MT トランジスタ

〇 E：交流電源 D：直流電源

S シンク出力

SS ソース出力

# PLC共通

## リレー出力について

メリット

絶縁耐圧が非常に高い。

接点の電圧・電流も非常に大きい。（大容量の負荷を使用できる）

電圧降下が少ない。

動作可能な温度範囲が広い。

交流のOn/Offができる。

デメリット

接点の摩耗による機械的な寿命が短い。

On/Offの速度が遅い。（mSec単位が限界）

## IEC61131-3

IEC61131-3 を推進するPLCopenと言う団体があるらしい？

omronが運営している PLCopen認定コース　と言うものがあるが、一日で終わるものなので多分大したことない。

PLCの国際標準プログラミングの規格の事。以下の５種類の形式が存在する。

|  |  |  |
| --- | --- | --- |
| 名称 | 概要 | 詳細 |
| IL  Instruction List | アセンブラ形式のリスト |  |
| LD  Ladder Diagram | 電気図面に似た書き方 | 日本では好まれる。 |
| ST  Structured Text | VBのような書き方 |  |
| FBD  Function Block  Diagram | ファンクションブロック | 機能ごとに分けたブロックを線でつなぐ  三菱PLCではST/FBDとなっており併用が可能。 |
| SFC  Sequential Function Chart | フロー図のような書き方 |  |

近年では、関数化は当然として、クラス化、継承、インターフェイスなどに対応したオブジェクト指向ファンクションブロックなど新しい記述方法もでてきているらしい。

# 三菱独自

## 記号など

三菱のマニュアルでは断りもなく，説明も無く，唐突に記号が出てきたりするのでまとめる。

AP

|  |  |  |
| --- | --- | --- |
| 記号 | 名称 | 詳細 |
| AP | 1回転あたりのパルス数 |  |
| AL | 1回転あたりの移動量 |  |
| AM | 単位倍率 |  |
|  |  |  |
| PB |  |  |
|  |  |  |

## ステップラダー

概要

・ステップリレー(S) は内部リレー(M) とほぼ同等。setでオンしてrstでOffする。

・STL Sn ～ STL Sn+1またはSTL Sn ～ RETを１区切りとして「ステート」と呼ぶ。

詳細

・複数のステップリレーが混在する場合，**番号の小さいものから順にオン**する必要がある。

（例）S1,S2 がある場合，S1をオンするまではS2 はオンできない。

⇒ **ステップを逆行する事は想定していない**らしく，どうあがいても実現できなかった。

・STL Snの実行で**Sn-1（直前のステップリレー番号）のみ**RSTする。 分かり難い！！

・STL Snの**直後のみ母線と直接繋ぐ**事が可能。（条件コイルが不要）分かり難い！！

・ステップリレーSnは**out を使ってもset 扱い**になる。 分かる訳ない！！

・ステップリレー(S)は通常の内部リレー(M)同等の使用法も可能。（個人的にはこの使用法を推奨）

・複数のステップリレーを同時にオンする事は可能。

・別々のステート間では**ダブルコイルが認められる。**

（コンパイル時にはエラーだが，シーケンサは実際動く　これは非常に危険な仕様！！）

⇒　ステップリレー(S) を静的に使って同機種オプション違い対応（≒C言語の #if ）は有りかも。

個人的な感想は，ステップリレーに長所が見られず設計理念が見えてこない。何に使うのかさっぱり分からない。マニュアルできちんと「良い例」を見せて欲しい。20.11.16

以下，マニュアルに対する反論：

・シーケンス設計が容易になる

⇒ STL Sn は控えめに言って直感的でなく効果が想像しにくい。⇒ 通常の内部リレー(M)と同様にSET, RSTを用いれば幾分マシ。但し，それなら最初から内部リレー(M)で良い。（「特別な内部リレー」として使うなら可）

事実、〇ヨタ系ではステップラダーを禁止しているらしい。

・ 第三者に対しても厳密な動きが伝えられる為

⇒ 行コメント（ステートメント）を残せばそれでよい。デバイスコメントよりもステートメントの方が使える文字数が多ので誤解が少なくなる。というか，GUI(Works2,3) のstlの視覚効果がsetやoutなどと変わらない為，一目で分かるようにする為には結局コメントを残す事になると思うのだが…。

# シーケンサ（三菱PLC）

## 基本用語

コイル LD（条件）側の事。タイマーではTCとなる。

接点 OUT（結果）側の事。タイマーではTTとなる。

## 機種ごとの違い

### FX3シリーズ

まとめ

|  |  |  |  |
| --- | --- | --- | --- |
|  | 3U | 3G | 3S |
| 高速入力 | 100kHz×6  50kHz×2 | 60kHz×4  10kHz×2 | 60kHz×2  10kHz×4 |
| 高速出力 | 100kHz×3  (op) 200kHz×4 | 100kHz×2or3 | 100kHz×2 |
| 通信ポート | 422 | USB/422 | USB/422 |
| 可変ボリューム | × | ２点 | ２点 |
| アナログ | 増設(ADP，ブロック) | 増設 | 内蔵/増設 |
| 増設 | BD/ADP/ブロック | BD/ADP | BD/ADP（種類少ない） |
| データレジスタ | 8000点 | 8000点 | 3000点 |
| prog容量 | 64kステップ | 32kステップ | 4kステップ |
| 記憶デバイス | RAM | EEPROM | EEPROM |
| 備考 | USB⇔422 の変換が市販されている。  実はこれが一番古いらしい。2005年？ | サービス電源が下(Y) 側にある。  2008年発売？ | 2013年発売？ |

補足

ADPアダプタ；左側 BDボード；正面に付ける ブロック　右側に付ける

### FX3U

一般

M8029 実行完了（様々な命令で共用する）ｍｔｃ

M8329 異常完了（様々な命令で共用する）

割込み

M8460 割込み入力信号

M8347~8377 割り込み信号反転。M8347がY0用で、M8357がY1用。Y0~Y3まで。

位置決め関連

M

### 消費電力

FX3U-16M□/E△ 30W（150mA@200V）　ヒューズ250V 3.15A

FX3U-32M□/E△ 35W（175mA@200V）　ヒューズ250V 3.15A

FX3U-48M□/E△ 40W（200mA@200V）　ヒューズ250V 5A

FX3U-64M□/E△ 45W（225mA@200V）　ヒューズ250V 5A

FX3U-128M□/E△ 65W（325mA@200V）　ヒューズ250V 5A

## GX Works

### ファイル形式

#### Developer ワークスペース形式

.gpj がプロジェクトの本体ファイル。他にResourceディレクトリなどがある。

※基本的にはDeveloperが無いと開けないが，Works2の「他形式プロジェクトを開く」でも開くことができる。

#### 開き方

Works2の場合

プロジェクト(P) → 他形式データを開く(R) → 他形式プロジェクトを開く(R)

#### Developer ワークスペース形式に変換

・シンプルプロジェクトである。FXCPU以外ならラベルを用いたプロジェクトでも可能。

・標準モードである

[ツール] ⇒ [オプション] ⇒ “ デバイスコメントエディタ ” で，“ 標準モード ” を選択

・セキュリティ設定がされている場合はAdministrators 権限に変更

・インラインST未使用

### ユニットデバイスの値をモニタする方法

Works2

オンライン → モニタ → デバイス/バッファメモリ一括モニタ

バッファメモリを選択し、ユニット先頭にユニット番号、アドレスにモニタ先頭アドレスを入力

（例）U0\G230 (230=0xE6) の場合 ユニット先頭：0 アドレス：0xE6



Works3

### CCLINK設定

Works2

ナビゲーション　→　パラメータネットワークパラメータ　→　CC-Link

リモート入力 0x0E0番系のRXを割り当てるデバイス番号を指定する？　2021-05-18

局情報 **接続相手**の情報を入力する

## ビット処理

### K4X0

ビットデバイスの桁指定

・桁数は4ビット単位。例えばK2X004 はX4~X13 となる。

・デバイス指定は４桁指定　□○○○　　例：K4X001

（参考）FX3プログラミングマニュアル[基本・応用命令解説編] 5.4 ビットの桁指定

### D0.0

ワードデバイスのビット指定

# omron

## 概要

### PLCの種類と概要

NX7 モーション，PLC、セーフティ，画像センサ，HMIの統合。

信号のIO（入出力）ユニットは別売り。サーバー的な使い方をする？

NX1 モーション，情報活用・安全対策，Ether。NX7の下位機種？

NX1P 高機能モーション制御，Ether。Sysmacのエントリーモデル

NJ モーション、PLC、セーフティ、画像センサ、HMIの統合

CJシリーズとのハード的互換を持ったSysmacシリーズ　という立ち位置らしい

CP Ether，4軸位置決め

CJ1 小型

CJ2 Ether，USBポート。CJ1の上位互換？

CS1 中型

NJシリーズ PLCの形をした、別物のマシンコントローラらしい

情報元（日付が無いが2015年くらいの記事？）

CJシリーズ 純粋なPLC。言語もラダー。

アナログ4ch　5μSec！と謳っていたが，5μは200kHzなのでそれほど大したこともない。

比較

|  |  |  |  |
| --- | --- | --- | --- |
|  | 値段（ミスミ） | 処理時間(LD) |  |
| omron [NX1P2](https://jp.misumi-ec.com/vona2/detail/222006004613/?KWSearch=NX1P2&searchFlow=results2products&list=PageSearchResult) | 約 \110,000 | 3.3nSec | EtherCAT |
| 三菱 [FX5U](https://jp.misumi-ec.com/vona2/detail/222005980930/?KWSearch=fx5u&searchFlow=results2products&list=PageSearchResult) | 約 \51,000 | 70nSec | CC-Link IE SSL非対応？ |

※NXはどちらかというとQシリーズに近いらしい

## 開発環境

CX-Programmer CS/CJシリーズ向きの開発環境。

2000年頃には既にあり，まだ新バージョンが更新されている

CX-One ラダーだけでなくサーボ設定などもできるらしい

Sysmac Studio NJ/NXシリーズ向きの開発環境

かなり重たいらしい　2020.02.19情報 ⇒　[情報元](https://plc-memo.com/difference/#toc2)

対応機種：<https://faweb.net/product/opc/plc/sysmac/unit>

### その他

ユニットverという特徴的な仕様があり，専用開発ツールのverが一致しないと設定できないらしい。

HMI 三菱で言うGOT（タッチパネル）。プログラマブルターミナルと呼んでいる（らしい）

## 仕様的特徴

### NJシリーズ

・IEC 61131-3に準拠　をうるさいくらい言ってくる

・CPUはIntelのAtomらしい

・IOユニットはCjのものが使用できるが，電源ユニットはNJ専用のものを使用する必要がある。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 型 | 軸数 | 性能 | EtherCAT  スレーブ数 | 入出力点数 | 容量 | 変数容量 |
| NJ101-9000 | 0 | LD3.3ns | 64 | 2560点  (40ユニット) | 3MB | 保持0.5Mb  非保持 2Mb |
| NJ101-1000 | 2 | LD3.3ns | 64 | 3MB |
| NJ101-1100 | 4 | LD2.0ns | 192 | 5MB | 保持0.5Mb  非保持 2Mb |
| NJ101-1200 | 8 | LD2.0ns | 192 | 5MB |
| NJ101-1300 | 16 | LD1.2ns | 192 | 20MB | 保持2Mb  非保持 4Mb |
| NJ101-1400 | 32 | LD1.2ns | 192 | 20MB |
| NJ101-1500 | 64 | LD1.2ns | 192 | 20MB |
| NJ101-1600 | 128 | LD0.37ns | 512 | 不可 | 80MB | 保持4Mb  非保持 256Mb |
| NJ101-1700 | 256 | LD0.37ns | 512 | 不可 | 80MB |

## 変数

Njシリーズは外部との信号，データのやりとりを全て変数を介して行う

I/Oマップの設定テーブルがあり，そこで変数を定義する。

ネットワーク公開の有無，定数化（可否），保持（可否）　などの設定がある。

## sysmac studio

Sysmac製品群（NJシリーズなど）の設定，モニタ，プログラミングを行う。

ロジック、モーション、セーフティのプログラミング

ユニット、軸の設定

EtherCAT 設定

実行状態のモニタ

モーションを含むシミュレーション

カムエディタ

## 用語

POU（Program Organization Unit）

# タッチパネル

## Pro-face

フランス　Schneider製の電機メーカー。

# 外部機器制御

## インバーター

### 通信パラメータ

|  |  |  |  |
| --- | --- | --- | --- |
|  | A820 | E720EX | 備考 |
| RS-485 通信局番 | 331 | 117 | 0-31　def:0 |
| ボーレート | 332 | 118 | 192 (19200) くらいにしておく。 def:192 |
| ストップビット | 333.b0 | 119.b0 | 0：1bit　　1：2bit |
| データ長 | 333.b1 | 119.b1 | 0：8bit　　1：7bit  ⇒ つまりデータ長が7bitでstopbitが1bitの場合 10。 |
| パリティ | 334 | 120 | 0：なし　　1：odd　　**２：even**　一般的なのはoddだが |
| 通信ターミネータ | 341 | 124 | 0：なし　　**1：CR**　　２：CR・LF |
|  |  |  |  |

### 通信制御に関して

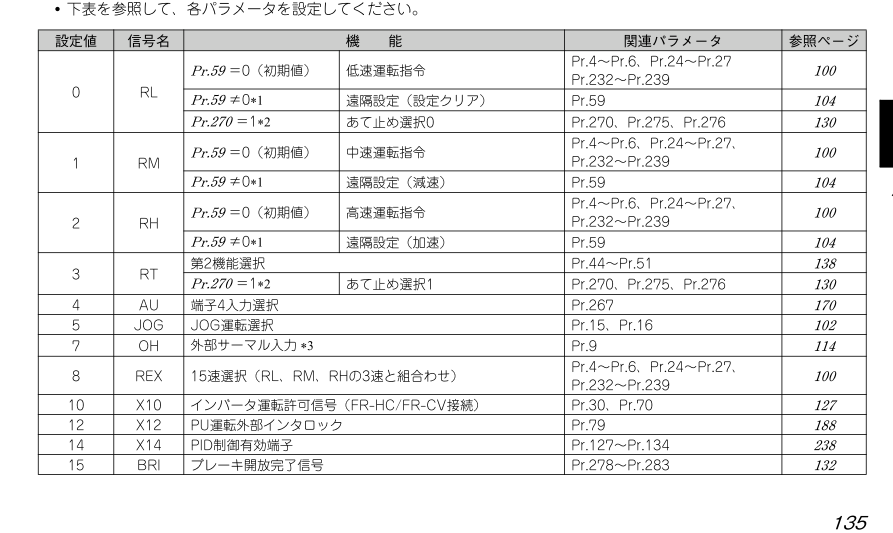
通信（NET）で実際に指令できる命令ビットは制約がある。例えばRES（リセット）は許可されていない。

下表はE700（無印）の場合



また，Pr.178～184で機能を変更できる。

Pr.178～184のいずれかをRT(=3) にする。



※Pr.270はあて止め制御　選択　初期値=0

### 初期設定パラメータ

|  |  |  |  |
| --- | --- | --- | --- |
|  | A820 | E720EX | 備考 |
| **運転モード** | 79 | 79 | 0：ボタン切り替え　1：PUパネル　2：外部／NET運転  3：外部/PU①　4：外部/PU②　などなど  （複雑すぎるので各マニュアル参照。） |
| RS-485 通信リトライ回数 | 335 | 121 | **初期値=1**なので，もう少し増やした方が良い。  調性時：9999[def]　　運転時：1～10が推奨 |
| RS-485 通信チェック時間間隔 | 336 | 122 | 単位0.1sec**初期値=0**なので，設定しないと確実にタイムアウトする。（なぜこんな値を初期値にした？）  調性時：9999[def]　　運転時：1～10が推奨 |
| 通信待ち時間設定 | 337 | 123 | 単位mSec　　9999：通信データ内の値を使用 |
| 通信運転指令権 | 338 | 338 | ０：NET　　１：外部 |
| 通信速度指令権 | 339 | 339 | ０：NET　　１：外部　　２：外部（ない場合指令端子無効） |
| 通信立ち上がりモード | 340 | 340 | ０：Pr.79に準ずる　１：ネットワーク　２：パネルで切り換え可  10：≒1。瞬低前の状態を維持。　　12：10と同様  820　12　(79：)  720 |
| NET モード操作権選択 | 550 | 550 | 9999[def]：自動認識  820　0：通信オプション　　1：RS-485  720　0：通信オプション　　2：PUコネクタ |
| PU モード操作権選択 | 551 | 551 |  |

局番 Pr.117(E720)

Pr.1427　“5000 ～ 5002、5006 ～ 5008” のいずれかに設定（恐らくポート番号）

### 運転パラメータ

動的パラメータとその番号

|  |  |  |  |
| --- | --- | --- | --- |
|  | A820 | E720EX | 備考 |
| 上限周波数 | 1 | 1 | INV [0.01Hz] 0～120　センサレス[rpm] 0～3000 |
| 下限周波数 | 2 | 2 | INV [0.01Hz] 0～120　センサレス[rpm] 0～3000 |
| ３速設定１ | 4~6 | 4~6 | [0.01Hz]  （補足）Pr.4が高速[RH]，Pr.5が中速[RM]，Pr.6が低速[RL] |
| JOG周波数 | 15 | 15 | JOG運転の速度 [0.01Hz] 0～590  ※但しJOGは外部（JOG端子に電気信号を入れる）かPUのみ。  翻って，３速設定は通信によるビット指令が可能。  関連パラメータPr.185 |

静的パラメータとその番号

|  |  |  |  |
| --- | --- | --- | --- |
|  | A820 | E720EX | 備考 |
| 回転速度表示 | 37 | 37 | Pr.505の時の機械速度を設定する。  ※これを設定すると周波数系のパラメータの単位が[rpm]になる。  **[0.01rpm] 単位ではない**ので注意！  ※E700（無印）の場合下記の「注意点」 に留意する。 |
| 速度設定基準 | 505 | 505 | 1～400Hz [0.01Hz] |
| 加速時間 | 7 | 7 | 単位　A820[0.1sec]　A720EX[0.01sec] |
| 減速時間 | 8 | 8 | 単位　A820[0.1sec]　A720EX[0.01sec] |
| 加減速時間単位 | 21 | 無し | val.=0：0.1sec ⇔ val.=1：0.01sec def.=0  ※E720EXではこの項目は無いが、E720では存在する |
| 任意アラーム | 997 | 997 | val.=16~253（def.=9999） |

注意点

E700（無印）シリーズではPr.505が存在しない、**Pr.37では60Hzの機械回転数を設定**。また、E700ではPr.37を設定しても**速度系のパラメータの単位はHzのまま**となる。翻って、IVMCの読書き値はPr.37の影響を受け単位がrpmとなる，といった**初見殺しの仕様**なので、注意！

（関連）Pr.3 基底周波数

おまけ　回転数表示に関するもの

|  |  |  |  |
| --- | --- | --- | --- |
|  | A820 | E720EX | 備考 |
| 回転数速度設定切替 | 144 | ? | ＝モーター極数の事 |
| 設定分解能切換え | 811 | ? |  |

⇒ モーター軸の回転数を表示する事になるらしい。

### その他のパラメータ

|  |  |  |  |
| --- | --- | --- | --- |
|  | A820 | E720EX | 備考 |
| リセット選択/PU抜け検出 |  | 75 | PUケーブルの抜け検出時の動作などを指定する。 |
| 端子機能選択 | 178～189  185JOG |  | STF端子などの入力端子、もしくはインバータ指令のbitに割り当てられた機能を変更する。  E720の初期値=9999（機能無し）  A820の初期値  ⇒ 機能無しにすると通信で制御できるようになる？ |

### 関数（インバーター）

#### IVDR：インバーターの運転制御（書込み）

命令コードを使ったインバーターのデータ書込み。

構文：IVCK　*n1　w2　w3[2]　n4　(out)b5[3]*

n1 インバーターの局番 (K0～K31) w2 命令コード

(out)w3[2]：書込みデータの先頭

n4 使用するch(K1:ch1，K2：ch2) (out)b5[3]：命令実行状態

（補足）IVMCの方ならば運転制御をしつつ運転監視ができるので，わざわざIVDRを使って書込む「命令コード」は限定される。可能性の高いものを下にまとめる。

|  |  |  |
| --- | --- | --- |
| 項目 | 命令コード | 詳細 |
| 特殊モニタ選択NO | HF3 | IVMCとIVCKなどで読み込める特殊モニタを選択する。 |
| インバーターリセット | HFD | H9696リセット  H9966インバーターはACKを返してからリセット |
| 異常内容一括クリア | HF4 | 値でH9696を指定する。異常履歴の一括クリア。 |

#### IVCK：インバーターの運転監視（読出し）

命令コードを使ったインバーターのデータ読出し。

構文：IVCK　*n1　w2　(out)w3[2]　n4　(out)b5[3]*

n1 インバーターの局番 (K0～K31) w2 命令コード

(out)w3[2]：読出しデータの先頭

n4 使用するch(K1:ch1，K2：ch2) (out)b5[3]：命令実行状態

（補足）IVMCの方ならば運転制御をしつつ運転監視ができるので，わざわざIVCKを使って読込む「命令コード」は限定される。可能性の高いものを下にまとめる。

|  |  |  |
| --- | --- | --- |
| 項目 | 命令コード | 詳細 |
| 出力電流 | H70 |  |
| 異常内容 | H74～H77 | H74が1回前（最新）と２回前の異常を8bitずつ，16bitで返す。  H75以降も同様で，H77は7回前と8回前を返す。 |
| 機種名 | H7C | ASCIIコード6桁で返す。空白部分は0x20で返す。  桁＝byte数？（未確認 |

#### IVMC：インバーターの複数コマンド

IVCKとIVDRを同時に実行するコマンド。

構文：IVMC　*n1　w2　w3[2]　(out)w4[2]　n5　(out)b6[3]*

n1 インバーターの局番 (K0～K31) w2 命令コード

w3[2]：書込みパラメータの先頭 (out)w4[2]：読出しパラメータの先頭

n5 使用するch(K1:ch1，K2：ch2) (out)b6[3]：命令実行状態

w2命令コードとパラメータの関係

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| コード | w3[0] | w3[1] | out1[0] | out1[1] |
| 0x0000 | 運転指令(拡張)  （=HF9） | 設定周波数(RAM) | インバーター  ステータスモニタ  (拡張)  (=H79)  ※ビット配列 | 出力周波数  （回転速度） |
| 0x0001 | 特殊モニタ |
| 0x0010 | 設定周波数  (RAM,EEPROM) | 出力周波数  （回転速度） |
| 0x0011 | 特殊モニタ |

※特殊モニタについては，マニュアルに一切の記述がない。使うな，と言う事で良いのかも。

b6[3]

b0：実行中 b1：正常完了 b2：異常完了

※FX3Uには無かった。

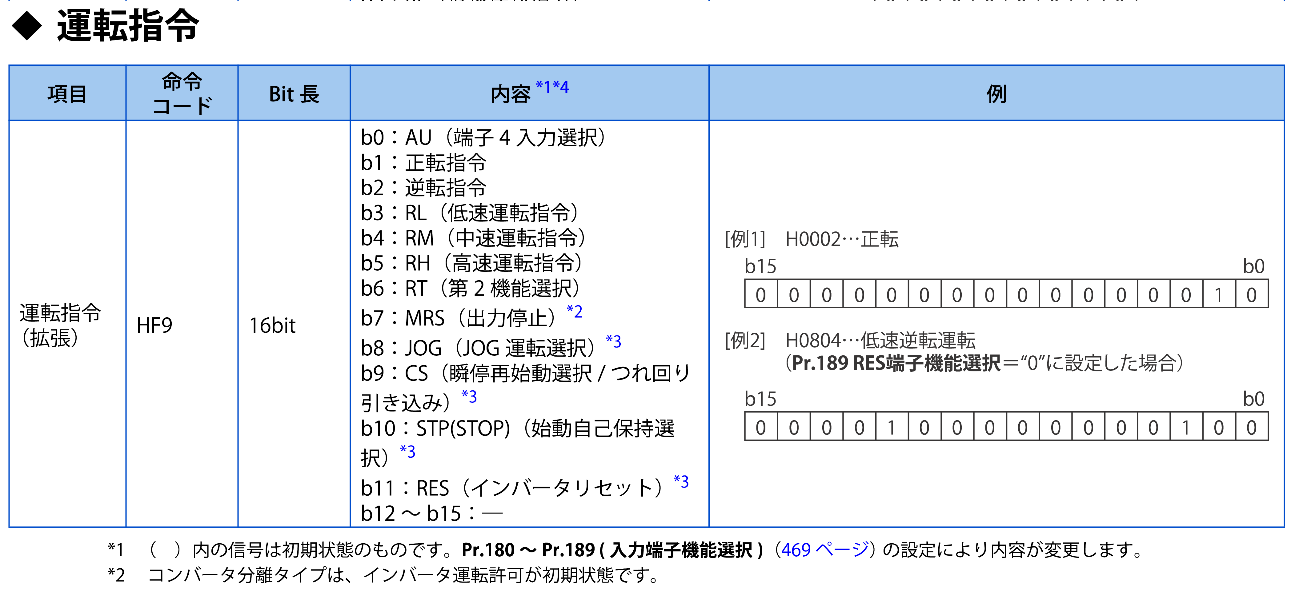
詳細：FX5Uマニュアル（シリアル通信編）5-8(p121)

##### IVMCの補足：

拡張運転指令

（写真はA800の詳細編 5.15(N)通信運転と設定p625，pより）

（E700は4.14.1 入力端子機能選択？



20.12.08メモ：b11インバータリセットは何も起こらなかった。ネットワーク経由では無理らしい。

　→ Pr.184　でリセット以外の機能を割り当てる事は可能

リセットしたい時は IVDRで命令コード 0xFD を選択する。

インバータステータスモニタ（拡張）



#### IVBWR：インバーターのパラメータ一括書込み

インバーターのパラメータの書き込み。（インバーターにより，パラメータ番号は異なる？）

構文：IVMC　*n1　n2　w3[]　n4　(out)b5[3]*

n1 インバーターの局番 (K0～K31) n2 パラメータの配列個数（ワード数）

w3[]：書込みパラメータの先頭　※パラメータ番号，値…　の繰り返し。つまり必ず偶数個になる。

n4：通信ch（K1～K4） （参考）[RS232/485通信](#_RS232/485通信)

(out)b5[3]：命令実行状態

## インバーター命令

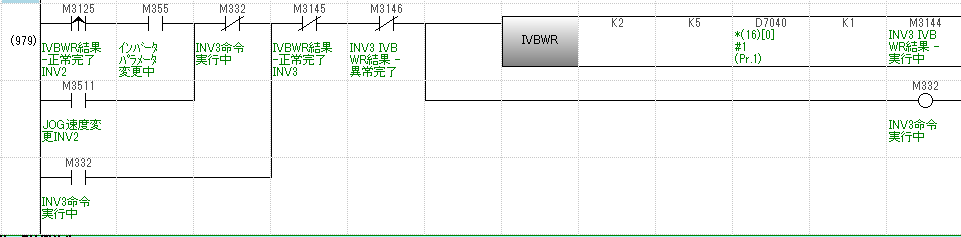
### はじめに

・インバーター系の命令は複数同時に並列実行しても，**１つずつしか実行されない**。

・インバーター系の命令は**基本的に実行終了まで接点側をオンし続ける**必要がある。

⇒ マニュアルには駆動コイルの立ち上がりでOKで，最後まで通信を実行するとあるが，実際には異常終了した。嘘ばっか。（FX5U シリアル通信編 5.8インバーター通信命令の共通事項 p121より）

（例）



補足

・この例のM355は，M3511のような単発（軸３のみ）のパラメータ変更の時に次の軸（軸４）の設定変更へ移行しない為のもの。

・マニュアルではSETを用いた命令になっている。そちらの方がすっきりして良いかも。

# ロギング

## 仕様まとめ

GOT単体でログを取る

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 概要 | 機能 | 備考 |
| ヒストリカルデータ | 画面に表示のみ |  |  |
| SDカード |  |  |  |
| USBホスト |  |  |  |

シーケンサ単体でログを取る

FX3U

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 概要 | 機能 | 備考 |
| コンパクトフラッシュ | FX3U-CF-ADP | 列数（ワード数）254までのcsvファイルを作成 | 専用のCF GT05-MEM-2GCは高額。 |
| フラッシュカード | FX3U-FLROM-1M |  |  |
| Ether使用 | FX3U-ENET-L |  |  |

# memo

## OPC UA

OPC Unified Architecture

工場などで使われる産業用システムをITシステムと連携させるための規格。

マルチベンダー製品間や異なるOSにまたがってデータ交換を可能にする安全で高信頼の産業通信用のデータ交換標準のこと。「Industrial Interoperability（産業用の相互接続性）」を実現することを目標に掲げ、Utilization（活用）、Connection（接続）、Communication（伝達）、Security（安全性）の4つを柱とする。OPC UAはOPC Classicをベースにして2006年に誕生