# 試験内容

・MCプロトコル

・RS485通信 19.08.23済

・

デバイス初期値

⇒けっこうめんどくさい記憶

メモ：これなんだった？

IPフィルタ

リアルタイムモニタ p160

位置決め

PWM出力 p291

RAS p162

# FX5U基本

## 基本設定

### 接続

基本的にプログラムの書き込みはEthernetケーブル経由で行う。

（参考）[Ethernet通信](#_Ethernet通信)

（補足）GOTとEthernet接続していれば，usbを用いたGOTトランスペアレント通信も可能。但し，ロギング設定ツールなど，Ethernetケーブル直接続でないと使えない機能やアプリが存在する。

### リモートリセット

プログラムの更新後にはリセットが必要。リモートリセットを有効にしとくと楽。

（但し，Web経由でのリセットを可能にする事になるので，注意が必要）

Works3ナビゲーション：CPUパラメータ → 動作関連設定

時計情報

※先に接続が完了している必要がある。

Works3メニュー：オンライン → 時計設定

## Ethernet通信

### 最初の接続

最初はethernetケーブルで，直接１対１で繋ぐしかない。

・ストレートケーブルでOK。特に，ハブを利用する時はクロスケーブルは使用禁止。

・**ローカルリンクアドレスが設定**（≒IPの設定なし）のパソコンからしかアクセスできないかも？

 左上の → Connection → 直接接続。 （これでプログラムの書き込みなど基本的なことはできる）

### IPアドレス指定

以下の場合はIPアドレス指定が必要：

・GOTトランスペアレントを使う場合。（通信経路が パソコン [usb] → GOT [ether] → FX5U　など）

・パソコンからルーターを介してFX5Uに接続する場合

Works3ナビゲーション：パラメータ → ユニットパラメータ → 

※GOTを使う場合の設定

GOTをダブルクリック → CPUモードで対象のシーケンサシリーズを設定 → GOT（Ethernet）トランスペアレント機能を利用する → 設定 → **シーケンサのIPアドレス**を入力

・MCプロトコル，ソケット通信などether通信を使う場合

※工場出荷状態だとFX5Uは192.168.3.250/24　GOTは192.168.3.18/24が設定されているらしい。

（補足１）/24はサブネットマスクが255.255.255.0 の意。

（補足２）初期IPは機種により異なるらしい。例えばiQRだと192.168.1.38/24 らしい。

#### 相手機器接続構成設定

MELSOFT接続（MX Component），SLMP（MCプロトコル）などの場合，以下の設定が必要

ユニットパラメータ：Ethernetポートの下の方にある「相手機器接続構成設定」の詳細設定

→ Ethernet構成画面 → 構成画面の右側のウィンドウからEthernet機器（汎用）から，それぞれ対象機器を選択してドラッグ＆ドロップする。その後，　を忘れないように！

ソケット通信

「Active接続機器」を追加（状況によりUnpassive/Fullpassive機器）

※Fullpassive の場合は通信相手（とそのポートまで）指定。

MCプロトコル（Ether）

「SLMP接続機器」を追加。シーケンサの再起動が必要。プロトコルは**3Eフレーム**を用いる。

### 接続先指定

パソコン ⇔ シーケンサが直接接続以外の場合，この設定も必要。

（上記のユニットパラメータだけでは通信できない）

操作：画面左の「接続先」→ Connection → その他の接続方法 [radioボタン] → その他の接続方法(o) [ボタン]

パソコン ⇔ シーケンサ間がetherの場合：

パソコン側I/F Ethernetボード

シーケンサ側I/F CPUユニット

**ここが発見しに難い！** ：CPUユニット をダブルクリック → ハブ経由接続

パソコン ⇔ シーケンサ間がUSBの場合（GOTトランスペアレント）

パソコン側I/F シリアルUSB

シーケンサ側I/F GOT

**ここが分かり難い！** ：GOT  をダブルクリック → GOT(Ethernet)トランスペアレント機能を利用する → 設定ボタンを押し，**シーケンサ側のIPアドレス**を入力

※GOT（直結）トランスペアレント機能　とは何？20.08.31

蛇足だが，ユニットパラメータでシーケンサのIPアドレスは入力済なのに，なぜここでもシーケンサ側のIP入力が必要なのかは理解に苦しむ。（接続経路はさておき）

### IPフィルタ

シーケンサにもIPフィルタリング機能があるらしい。ただ本来フィルタリングはルーターなどで行うべきなので使う機会があるか分からないが。　ナビゲーションウィンドウ ： FX5UCPU → ユニットパラメータ→ Ethernetポート → 応用設定 → セキュリティ　※複数の場合は範囲設定のチェックを入れる。

Webページ表示

※urlをhttps にしないように注意！

できる事：デバイス値の読み書き。（書き込みは権限の設定が必要）

CPUエラーチェック

## デバイス

### デバイス範囲表 FX5U

|  |  |  |
| --- | --- | --- |
|  | FX3U | 備考 |
| データ：一般 | D0~D199 | 一般メモリは電源OFF，Run→Stopでクリア |
| データ：キープ（可変） | D200~D7999 |  |
| 拡張レジスタ | R0～R32767 | ？？？ |
| インデックス | V0-V7, Z0-Z7 | ？？？ |
| ロングインデックス | LZ | 2点しかないらしい |
|  |  |  |
| 内部リレー：一般 | M0 ~ M499 |  |
| 内部リレー：キープ | M500 ~ M7679 | [kp] と付ける事にする 19.07.01 |
|  |  |  |
| タイマ：積算 | ST0-T15 | タイマ(T)とは別。タイマは一般メモリ扱い |
| FX5Uではタイマの単位時間ごとにout [100mSec], outh[10mSec], ouths[1mSec] で使い分ける | | |
| タイマ：一般 | T16-T511 |  |
|  |  |  |
| カウンタ：停電保持 | C100～C199 |  |
|  |  |  |
| ステップラダー |  |  |
|  | P0～P4095 | P63はENDになる。使えない。 |

FX5Uはラッチ範囲を自由に設定できる

ナビゲーションウィンドウ → パラメータ → FX5UCPU → CPUパラメータ → ”メモリ/デバイス設定” →”デバイス/ラベルメモリエリア設定” → ”デバイス/ラベルメモリエリア詳細設定” →

但しバッテリ【 FX3U-32BL ；￥3000程度 】が必要

 の設定が必要。

デバイス（高速）設定：詳細設定 X，Y，D，M などは**「高速」の表現**　（少し分かり難い）

デバイス（標準）設定：詳細設定 R，W，SWなど ファイルレジスタ

### デバイス種類

|  |  |  |
| --- | --- | --- |
| 記述法 |  | 補足 |
| D100 | 通常デバイス |  |
| R100 | ファイルレジスタ  （15bitブロック） | 通常デバイスとほぼ同じだが，アクセス速度がやや遅めで，保持にはオプションのバッテリが必要（標準エリア）  RSETで15bit（R0~R32767）のブロックを指定してアクセスする。ブロック番号は0から始まる。  ※**FX5UはR0~R32767**までしかない。 |
| RZ100 | ファイルレジスタ  （連番アクセス） | ZR連番アクセス方式  32768番以降のメモリに直接アクセスする  ⇒ 結局Rと同じところにアクセスする？ |
| D100Z1 | インデックス修飾 | LZロングインデックス 32bit対応しているだけで，使い方は同じ。 |
| L | ラッチリレー | 停電保持できる特殊リレー  Mの停電保持との違いは？⇒どうも無さそう |
| F | アナンシエータ | SM62, SD62~SD79と自動的に関連する内部リレー。ユーザー定義の異常などに用いる |
| V | エッジリレー | iQ-R でEGP/EGFを使うときに |
| B | リンクリレー | ネットワークユニットで用いる？ |
| RD | リフレッシュデータ |  |
| U□￥G123 | ユニット | インテリジェント機能ユニットへのアクセス |
| U□￥D100 | CPUバッファメモリ | RCPU限定らしい？ |
| U□￥HG123 | 定周期通信 | iQRで用いるらしい |
| J□￥D100 | リンクダイレクトデバイス | CC-Link IEの相手先デバイス |

### SM, SDについて

FX3Uにおける8000番台の特殊デバイスの事。デフォルトでコメントが入っている

3Uとの比較

|  |  |  |
| --- | --- | --- |
| デバイス番号 | 概要 | 3U |
| SD210～216 | 時計データ。年，月，日，時，分，秒，曜日 | D8013～8018 |
| SM400, SM8000 | 常時ON | M8000 |
| SM402, SM8002 | イニシャルパルス | M8002 |
| SM409, SM8011 | 0.01秒周期 |  |
| SM410, SM8012 | 0.1秒周期 |  |
| SM412, SM8013 | 1秒周期 |  |

3Uには無かったもの

|  |  |  |
| --- | --- | --- |
| デバイス番号 | 概要 | 備考 |
| SM411 | 0.2秒周期 |  |
| SM413 | 2秒周期 |  |
| SM414 | 2 × n × sec周期 | nはnanoではない。紛らわしい！  nはSD414で指定する。 |
| SM415 | 2 × n × msec | nはSD415で指定する。def.=30 |

よく使うもの

|  |  |  |
| --- | --- | --- |
| デバイス番号 | 概要 | 備考 |
| SD6020 | FX5U付属アナログch1の値 |  |
|  |  |  |
|  |  |  |
| SD8010 | スキャンタイム　但し単位は100μSec | SD8011最小値　　SD8012最大値 |
|  |  |  |

### バッファ

なお，転送方法は以下の２通りがある

TOを用いる

to　ユニット番号␣バッファ番号␣値␣ワード数

（例）to H1 K6016 D100 K1

MOVを用いる（ビットの場合はoutやset）

U + ユニットの先頭アドレス+ ￥ + G + バッファ番号

（例）mov D100 U1￥G6016

補足：このU□￥G△　の表現を「機能デバイス」と呼ぶ。また，Uの後に付ける番号はユニット番号ではない。そのユニットのデバイスの先頭アドレス。￥をつけるものはCPU以外のデバイスを指す。

また，バッファにもインデックスは使用可能（らしい 19.10.28）

|  |  |  |
| --- | --- | --- |
|  | mov | to |
| メリット | 例えばU1￥G6016と入力するだけで，自動的にコメントが入るので，何の設定項目なのかが分かりやすい。 | 配列感覚で，数バイト連続で読み書きできる。 |
| デメリット |  | 例えばK6016がなんの設定項目なのか分かり難い。  ⇒ 対策として値が6016となる定数ラベルを用意して，そのラベルにコメントを書く手がある。 |
| 検索 | U1\G6016 を検索 | K6016 を検索 |

### 自動リフレッシュ

インテリジェンスユニットのバッファ値をシーケンサ内に自動で更新してくれる機能の事。

20.07.09過去に一度試した事があるが，むしろバグを引き起こしそうな気がして、この機能の追及はしていない。

### DX（ダイレクトデバイス）

実はシーケンサのXY入出力はリアルタイム性がない。

Xはプログラムの先頭で一括読み込み。つまり前スキャンのEND辺りの状態。

Yは内部処理に関しては即反映されるが，実際の出力はEND時点？（１スキャン後）となる。

（但し，ユニットの機能デバイス[U□￥G△] は即座に反映される。）

ダイレクトデバイスは即座に反映されるリアルタイム性のある入出力デバイスとなっている。

自動リフレッシュとの違い

ダイレクトデバイスは即時反映。自動リフレッシュはリフレッシュタイミングがある。

GX Work3オペレーティングマニュアル[SH-081214-n]より

### その他

オーバーフロー，アンダーフロー

例えばUnsigned16の場合の例を示す。

|  |  |  |  |
| --- | --- | --- | --- |
| dec | bin | 増減 |  |
| 32767 | 0111 1111 1111 1111 | ↓+1  ↓-1 |  |
| -32768 | 1000 0000 0000 0000 |
| -32767 | 0111 1111 1111 1111 |
| -1 | 1111 1111 1111 1111 | ↓+1  ↓-1 |  |
| 0 | 0000 0000 0000 0000 |
| -1 | 1111 1111 1111 1111 |

自動割り付け

ツール → 自動割付デバイス設定

## ネットワーク

## 3Uと大きく異なる点のまとめ

Works3になって追加された？標準のラダーコマンドがある。

|  |  |  |
| --- | --- | --- |
| アイコン | コマンド |  |
|  | ldpi | 常にONで，立ち上がりの1スキャンだけOFFする。 |
|  | ldfi | 常にOFFで，立ち下がりの1スキャンだけOFFする。 |

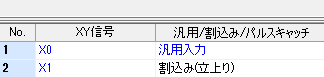
### 割込み

#### 基本

設定の仕様やプログラムの書き方が3Uの時から大分変わっている。

１．EI

EIを使う事自体は変化していないが，**母線と直接結ばない**。ld sm400 EI などとする。

２．高速入力の機能割り当て

ナビゲーションウィンドウ → パラメータ → FX5UCPU → ユニットパラメータ → 高速I/O → 入力機能 → 汎用/割込み/パルスキャッチ → 詳細設定

⇒ 右図のような画面が表示されるので，割込みを選択

元の画面に戻った時に  を忘れないように（三菱のGUIはこういった所が本当に不親切！！

⇒ 続いてユニットパラメータ → 入力応答時間　を設定するのが推奨されているらしい。

機能 割込み番号 内容

|  |  |  |
| --- | --- | --- |
| 種類 | 番号 | 説明 |
| 入力(カウンタ含む)からの割込み | I0～I15 | CPUユニットの入力割込み　内，最大８点まで |
| 高速比較一致割込み | I16～I23 | CPUユニットの高速比較一致割込みで使用 |
| 内部タイマ（定周期） | I28～I31 | 内部タイマによる定周期割込みで使用する割込み |
| ユニットからの割込み | I50～I177 | 割込み機能があるユニットで使用する割込みポインタ |

X0～X17がI0～I15に対応。

定周期

デフォルトでは以下の様になっている

I28 100mSec　　I29 40mSec　　I30 20mSec　　I31 10mSec

Works3のCPUパラメータで設定できる。最低単位は1mSec

補足情報：

DI割込み禁止には引数を１つ取る関数（オーバーロード？）が有り，指定した優先度値以上の割込みを禁止できる（らしい）e.g. DI K3だと優先度1,2 は実行される。（優先度値は大きいほど優先度が低い。）

・高速パルス入出力ユニットを利用する場合はX20以降の入力デバイスで設定。

その場合I50～I177の内１つを任意で選択する。

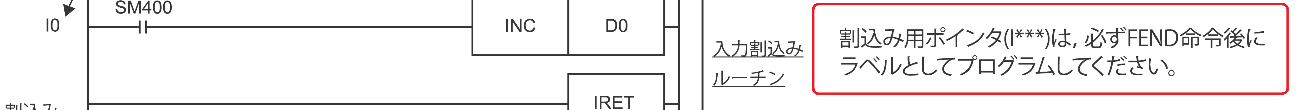
#### 注意点

・割込みルーチン内でOUT したbitデバイスは，SETと同じ効果となる。

⇒ 「SET しか使えない」と覚えた方が良い。

なお，これはタイマやカウンタのbit でも同じ。（この仕様はFX3Uの頃も同じ）

・マニュアル（汎用FUN汎用FB編）[jy997d54701] 8.3. 割込みプログラムからの復帰　より



⇒ FNED命令後であって，**直後**とは書いてないが表現が紛らわしい。加えてマニュアルのサンプルプログラムのほとんどはFENDの直後にIラベルがあるので，ユーザーが誤解しても仕方ない。（Pのルーチンを挟んでも機能した事を確認。20.07.14）

だいたいこんな当たり前の事，ここで改めて言及する必用ある？

#### 多重割込み

先に結論：

安定した処理の為には割込みはむしろ避けるべきで，この機能を使うようではむしろプログラム設計が悪いと言える。

割込み処理中に他の割込みが入る事。この場合，優先順位が高い順に処理が行われる。

設定値は全体禁止か全体許可のみ。

操作：Works3ナビゲーションウィンドウ：FX5CPU→CPUパラメータ→割込み設定

多重割込みを「許可する」

なお，基本的にI番号が若くなるにつれ優先度も高いが，優先度バランス変更可能

操作：ナビゲーションウィンドウ → パラメータ → FX5UCPU → CPUパラメータ → 割込み設定 → ユニットからの割込み優先度設定

設定値：１～３（１が最優先；３は優先度が低い）

#### 関連デバイスや命令

命令：

IMASK 一括してマスクする。KやHなどの固定値が使えず，Dデバイスを使う必要がある。（めんどくさい）

初期値ではすべてのマスクビットがonしている為，通常この命令を使う事は無い。

※IMASK は母線に直結（SIMASKは直結しない：こうゆうのきちんと統一して欲しい）

SIMASK 個別にマスクする。引数に直接割込みデバイス（I1など）を取り，個別に指定する。

HIOEN

HCMOV

DHCMOV

デバイス：

SD757 現在の割込み優先度

SD758 割込み禁止優先度設定値

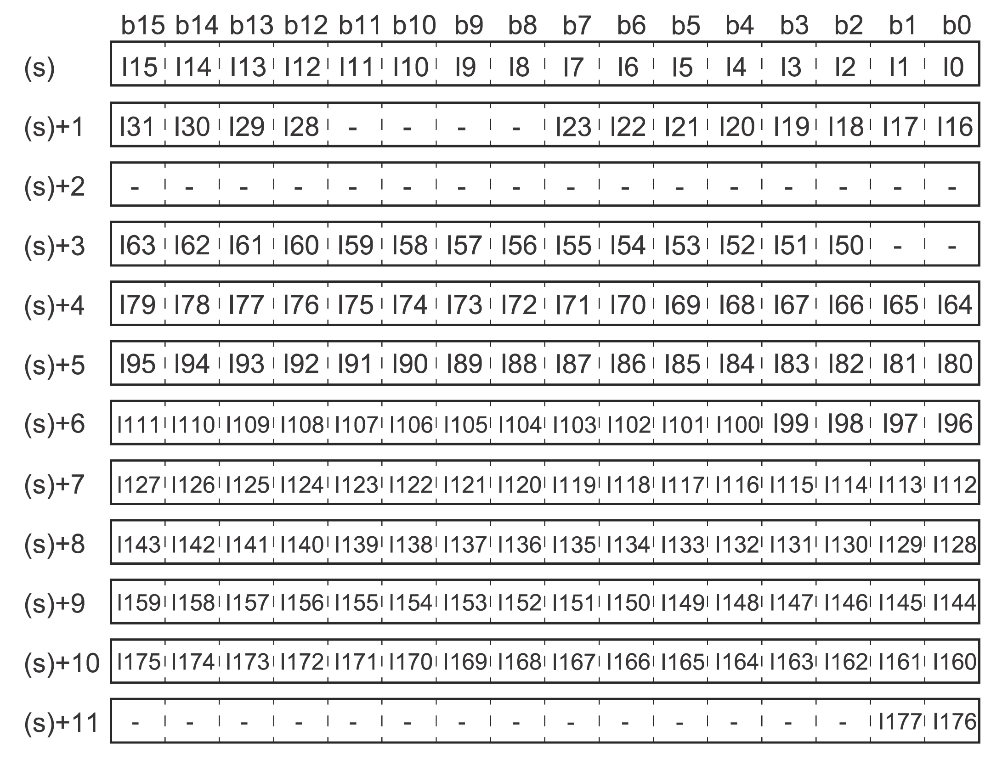
SD1400～1411 IMASKマスクパターン。なお，初期値は0xFF のような，全てのビットがonした値になっている。

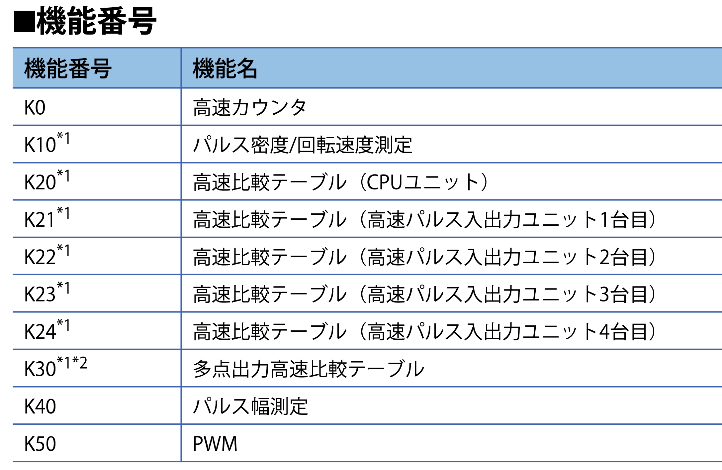
SM8050～8059 個別の割込み禁止(on時)

#### 資料（割込み）

パルスキャッチ

IMASK命令のビットと割込み番号の対応表：



HIOENで指定する機能番号

後学の為に？一応残しておく情報。

高速入出力機能の有効化 20.07.14 不要だった

HIOEN 命令を使用する。引数はS1：機能番号，S2：機能を開始するCH，S3：機能を停止するCH

.b0(CH1) から.b7(CH8) までがCPUユニット用。

## RS232/485通信

チャンネル：

1ch 内臓ポート (485) 2ch 増設ボード（485/232 真ん中）

3,4ch 増設アダプタ（485/232左側）

Works3 設定

ナビゲーションウィンドウ → パラメータ → FX5UCPU →  → 

ボーレート等の設定はここで行う。

|  |  |  |
| --- | --- | --- |
| 形式 | 概要 | 備考等 |
| MELSOFT接続 |  | プログラムなどの読み書き？ |
| 無手順通信 | 通常のデータ通信 |  |
| MCプロトコル | 三菱独自プロトコル | パソコンやマイコンなどと通信。 |
| MODBUS\_RTU |  | 5Uは未対応？ |
| 通信プロトコル支援 | 他機器のプロトコル | 外部機器のプロトコルに合わせた通信 |
| インバーター通信 |  | 最大16台 |
| 簡易PC間リンク | シーケンサのリンク | 内部デバイスを他のシーケンサに転送したりする。 |
| 並列リンク | FX5の２台接続 |  |

### インバーター通信

局番順に接続する必用は無い。局番は以下の様に設定をする。

|  |  |  |
| --- | --- | --- |
| Pr.番号 | 名称 | 詳細 |
| 117 | PU（※１）通信局番 | 0~31 |
| 331 | RS-485通信局番 | 0~31 |
| 435 | CC-Link IE | 0~255 |
| 548 | USB通信局番 | 0~31 |

運転モードの概要

|  |  |  |
| --- | --- | --- |
| モード | LED | 詳細 |
| PU | PU | インバーター本体の操作パネル。またはPU コネクタによる RS-485 通信 |
| 外部運転 | EXT | 端子台経由。外部ボリュームやスイッチなどで指令。 |
| ネットワーク | NET | RS-485端子や通信オプションを使用。 |

## アナログ

### 概要

FX5Uでは特殊な方式でアナログ値を取得できる。

・回数平均，移動平均を指定した場合は１サンプルの時間はシーケンサのサイクルタイムに等しくなる。

⇒　ある程度正確な値が欲しい場合はコンスタントスキャンをする必要があるという事？

（参考）FX5ユーザーマニュアル（アナログ編）[jy997d60401f] 2.3 A/D変換方式

|  |  |  |
| --- | --- | --- |
| 型番 | 取付 | 補足 |
| FX5-4AD-ADP | アダプタ | 入力4ch  ￥58,000 |
| FX5-8AD | ブロック | マルチ入力（電圧，電流，熱電対、測温抵抗体）  ￥98,000 |
|  |  |  |

補足：アダプタは左側，ブロックは右側に付ける

### 関連デバイス

|  |  |  |
| --- | --- | --- |
| 番号 | 名称 | 詳細 |
| SD6020 | **アナログch1出力値** | メインのデバイス。　ch2はD6060 |
| SD6023 | アナログch1平均処理指定 | 0: サンプリング処理　　=END時に普通に取り込む。  1: 時間平均 [1～10000mSec]  2: 回数平均 [ 4～32767回]　 3: 移動平均　[ 2～64回 ]  SD6024とペアで使う。 |
| SD6024 | アナログch1設定値 | 平均時間/平均回数/移動平均設定 |
| SD6026 | 最大値 | 最大（最小）値のホールドをしてくれる  （関連）SM6026 　　最小値 |
| SD6028 | ch1 スケーリング上限値 | 自動的にスケール計算をしてくれる。⇒格納はどこ？  範囲：-32768～+32767　　下限値は SD6029  （関連）SM6028有効／無効 |
| SD6058 | ch1 アラームコード |  |
| SD6059 | ch1 エラーコード |  |
| SD8010 | スキャンタイム現在値　[単位：0.1mSec ] | 通常 0.3μsecほど |
| SD8039 | コンスタントスキャンタイム | 2～20000（0.1ms単位）  WDTの設定時間より短くする必要がある。 |
| SM6020 | A/D変換完了フラグ |  |
| SM6021 | **A/D変換許可，禁止 ch1** | 0:変換許可　1:変換禁止  ch2 SM6061  注意！0，変換許可，共にデフォルト値だと連想するが，まさかの1:変換禁止がデフォルト設定。こんなんばっか。 |
| SM6028 | スケーリング有効無効 | 初期値：無効　※使ってみたが，スケーリングはかえって手間が増える。楽になる場合もないわけでは無いと思うが。 |
| SM6058 | A/D変換アラーム |  |
| SM8039 | コンスタントスキャンモード | 初期値：off |
|  |  |  |

※WDT（ウォッチドッグタイマー）の設定方法

ナビゲーションウィンドウ［パラメータ］［FX5UJCPU］または［FX5UCPU］［CPUパラメータ］”RAS設定

””スキャンタイム監視時間（WDT）設定”

⇒ 初期スキャンと２スキャン目以降で大きく値が異なるらしい。

# FX5U：応用

## 雑多情報

### コンスタントスキャン

スキャンタイムを強制的に一定にする。

ナビゲーションウィンドウ：パラメータ → CPUパラメータ → RAS設定 → コンスタントスキャン設定

条件：「WDTの設定時間」>「コンスタントスキャンの設定時間」>「プログラムの最大スキャンタイム」

※WDT スキャンタイムの監視タイミング時間（ウォッチドッグ）

## ロギング

### 概要

・PLCファームウェアver1.040以降

・設定には「CPUユニットロギング設定ツール」が必要。（ver1.64S以降

（設製造番号P1616Y以降でないとダメらしい）

・記録媒体はSDカード。内部メモリには保存できないらしい

・ログが取られるタイミングは常に毎スキャンのEND処理時。

・4設定 × デバイス点数は128点

・トリガ前のレコードも取得できる。（つまりどこかで常に記録している）

### 初期設定

Windowsスタート→MELSOFT→CPUユニットロギングツール

（又はWorks3メニュー：ツール → ロギング設定ツール　でも良い）

#### 接続設定

CPUユニットロギング設定ツールのメニュー：オンライン→接続先設定

・連続ロギング 指定の時間間隔でデータ収集。但し「条件指定」も選べ，その条件にビット／ワードデバイスを指定できる。

・トリガロギング 指定のタイミングでデータ収集。LOGTRG実行時，もしくは指定ワードデバイスの値が指定の条件を満たした時。

### SDカード

取り出しの際にはSM606（SDメモリカード強制使用停止指示）をONにする。

実行結果はSM607で確認できる。SM607がOnしていれば取り外し可能。

### メモリダンプ

# オプション品

RS-232 アダプタ（左側）FX5-232ADP

RS-485 標準装備。アダプタもある。

アナログ入出力 標準装備。アダプタ有り，但し入出力はそれぞれ別々。

シンプルモーション 80SSC-S (or40SSC-s)

# GX Works3：基本

## 表示

### コメント

コメント：デバイスコメントの事

ステートメント：行中コメントの事

ノート：

### 色及びフォント

フォント設定　自動，ユーザー設定

### コメントの行数を変更する

ツール → オプション： → ラダーエディタ：

表示形式（スクロールバーを真ん中の方へ）→ 行数 行数は３が良いと思う[def. 4]

デバイス／ラベル名表示形式変更

１セル表示，折り返し表示

※違いが良く分からない。19.12.12

### 背景の格子柄

背景の格子柄は非表示にできる。

表示 → グリッド

## Works3特有

### ユニットラベル

表示→ドッキングウィンドウ→部品選択

ユニットラベルのNodeを選んで，任意の機器名のフォルダを右クリック→ユニットラベルの追加。

ユニットのFBにはこの名前の引数が大体あり，FB内の細かい処理を変えている（らしい）。

## その他雑多情報

### 一般情報

・PLCモニタ中に，Shift を押しながらビットデバイスをダブルクリックすると，対象のビットデバイスのOn/Off を変更できる。（これはWorks2でも同じ）

### 注意事項

・DINT2INT などの命令では入力元は常に32bitとみなされ，更に厄介な事にC言語でいう参照型で読み込まれる。つまり誤って16bitのデバイスを指定すると隣のデバイスの**値は0に書き換えられる。**

例）D100, D101などと連番で使っている場合，**D101には常に0の値が格納**されてしまう。

ほんとどうしようもないクソ仕様だと思う。三菱レベル低すぎ。

・16bitの掛け算（MUL）の結果は16bitになる。32bitにしたい場合は事前にINT\_TO\_DINTを行う必要がある。また，32bitのMULの出力結果も32bitになる。

翻って，DMULの出力結果は64bitになる。（非常に分かり難い）

・**CALLでの番号の重複は問題ない**らしい。　例）CALL P1; CALL P1;　は可能（やるべきではないが。）

但し，**CALLとCALLP など，別命令で重複させてはいけない** ⇒ 個人的にはめちゃめちゃ違和感のある仕様

### 比較

= \_U

unsigned比較。対象デバイスを符号無しデータとして処理した後に比較する。

### デバイス関連

#### 機能デバイスのモニタ

機能デバイス（U□￥G△）はデフォルト設定ではモニタできないが，以下の方法でモニタも可能。

Works3のメニュー：ツール → オプション → モニタ → ラダーエディタ → バッファメモリ，リンクメモリをモニタする　を「はい」に。但し，**スキャンタイムがかなり伸びる**らしいので，デバッグの時だけにする。

### 時計

SM213をON するとSD210～SD216に時計情報を呼び出せる。従来通りTRD を用いても良い。

## 命令まとめの概要

パラメータは指令，戻り値関係なしに左から順に1,2,3…と番号をつける

例） s1 s2 d3

s single=16bit

d double=32bit

b bit

any bit/single/double 問わない場合

any16 bit/single 問わない場合

f float=浮動小数点

(out) 戻り値

b[3] 指定のデバイスを先頭として，連続で複数デバイスを使う。この例では3デバイス連続で使用。

n 他の引数と系統がまったく異なる場合など

## 命令まとめ：出力命令

### FF：ビット反転

構文：FF　*b1*

b1 反転するビットデバイス

## 命令まとめ：プログラム制御

抜粋。特に3Uと大きく異なったり，3Uにはなかったものを中心。

### XCALL：条件付きサブルーチンCALL

※マニュアルではCALLとの違いが明確に説明されてない。

以下の様なプログラムを例に挙げる

ld M100 call P1

FEND

P1; ld M0 out M1

RET

END

M1 の結果（見やすくする為1か0でon/offを示す事にする）

|  |  |  |  |
| --- | --- | --- | --- |
| M1の結果  CALL | | M0 | |
| ON | ON→OFF |
| M100 | ON | 1 | 0 |
|  | ON→OFF | **1** | × |

|  |  |  |  |
| --- | --- | --- | --- |
| M1の結果  XCALL | | M0 | |
| ON | ON→OFF |
| M100 | ON | 1 | 0 |
|  | ON→OFF | **0** | × |

つまりXCALLされたかどうかでサブルーチン内のコイル (out側)が強制的にOn/Offされるという事。XCALLの呼び出しの無い時はサブルーチン内の各接点(ld側)は無視され，すべてOFFになる。

結論：一見では理解し難い為，バグの温床になるのでXCALLは使わない方が良い。

但し，どちらかと言うとCALLでの挙動がおかしいようにも思えるのでなんとも言えないところ。

### STL：ステップラダー開始

SET S1

STL S1

/\* 処理 \*/

SET S2

STL S2

/\* 処理 \*/

・ステップラダーS はS0~S4095まであるが，どれか１つOnしたら他のすべてがOffする　と言う仕様らしい（未確認 20.03.25）

・ステップラダーは不揮発メモリで保持している。

疑問

set とout で結果が変わるだろうか？

ZRST

### ZRST：区間内の一括リセット

構文：ZRST　*d1[]　 d2*

d1[]　対象デバイス先頭（ワード/ビット） d2　対象デバイス先頭（ワード/ビット）

e.g. 1011 ⇒ 1101

注意！対象デバイスがそのまま変更されるので，必要であれば先にmovなどでコピーを作成する。

（参考）

## 命令まとめ7：基本命令

### DIS：16ビットデータの4ビット分離

DISP

構文：DIS　*w1　w2　n3*

w1 対象デバイス w2　分離データの格納先

n3 分離数（0は無処理）範囲[1-4]

(opp) UNI

### SMOV：4ビット単位のデータ転送

※SM8168がOFFの時はBCD変換になる。つまり10進で0~9999までの数値しか対応しない。

※対象ビット以外は上書きされない為，気を付けないとバグの温床になる。SMOVは使わない方が良い。

⇒ 普通に0x100などで割ったり論理積の方が楽だし見やすい。「三菱自己満，不要な命令」認定。

（参考）K2M100 などとするとビットデバイスの8bit分扱える。D100.0とするとワードデバイスをビットとして扱える。但し，K2D100.0 という合わせ技は使えない。

## 命令まとめ8.1：ローテーション

### ROR：ビットローテーション右回転

構文：ROR　*w1　 n2*

w1　対象デバイス n2　ローテーション回数

e.g. 1011 ⇒ 1101

注意！対象デバイスがそのまま変更されるので，必要であれば先にmovなどでコピーを作成する。

（参考）

ROL 左回転。e.g. 1011 ⇒ 0111

RCR キャリフラグSM700を利用

## 命令まとめ8.7：実数命令

### DECMP：実数比較

構文：DECMP　*f1　f2　(out)b3[3]*

### DEZCP：実数帯域比較

構文：DEZCP　*f1　 f2　 f3　(out)b4[3]*

f1　下側閾値 f2　上側閾値 f3　比較対象

b4[0]　f3 < f1 b3[1]　f1 ≦ f3 ≦ f2 b3[2]　 f1 < f3

## 命令まとめ8.12：特殊カウンタ命令

### 特殊カウンタの設定

Works3ナビゲーションウィンドウ → パラメータ→ FX5UCPU →ユニットパラメータ → 高速I/O

→ 入力機能 → 高速カウンタ → 詳細設定 → 基本設定

（補足）chは基本的に ≒ X番号と思えばよい。ch1がX0。但し1相2入力などで組み合わせる事もある。

S/W 完全にシングル？

H/W となりのビットをB相で使う？

### SPD：ビットローテーション右回転

※先に高速カウンタの対象チャンネルの設定をする必要がある

構文：SPD　*s1　 n2　 d3*

s1　対象デバイス ビット／ワード

n2　計測時間長さ[mSec] d3

SD4507,SD4506

## 命令まとめ：ステップラダー

※ただし個人的にはステップラダーは多用しない方が良いと考える。

理由：主な理由は後で見た時にどういった処理になるのか分かり難い

他にも，ぱっと見ダブルコイルや同じ接点が無駄に増える，修正が面倒くさい　など。

set S1

stl S1

ld X1 out M1 この間の処理はステートと呼ばれる。

ld X2 SET S2 # ステート内でもステップラダーをOnできる。

retstl

## 命令まとめ8.17：パルス系命令

### SPD：16bitパルス密度測定

### PWM：16bitパルス幅変調

構文：SPD　*s1　 s2　 n3*

s1　オン時間[mSec] s2　パルス周期[mSec]

n3　対象チャンネル番号（Y0～Y7を指定）

関連デバイス：SD5302[32bit],SD5304[32bit]　など

## 命令まとめ：通信

### SP.SOCOPEN：ソケット通信

SP.SOCOPEN U1 16bit1 S2 D1

U1：ダミー文字列 “U5” とする

16bit1：コネクション番号 1～8

16bit2[10]：コントロールデータを格納するデバイスの先頭番号

arg[0] 実行タイプ

0x0000：Works3で設定した「相手機器接続構成設定」の情報を利用

0x8000：arg[1-9] の値を利用

注意！オープン完了状態を示す D10680.0などは，**[b0]～[b7]がコネクションNo.1～8に対応**している。

### TO：他ユニットへのデータ書き込み

(also) DTO, TOP, DTOP (see also) FROM,

TO u1 s1 s2 n3:16

u1：ユニット番号

s1：データ書き込み先のBFM# 先頭アドレス

s2：書き込むデータの先頭アドレス，または値を直接指定

n：書き込むデータのワード数　例）16bitを1つならK1

## 命令まとめ：特殊タイマ命令

抜粋。特に3Uと大きく異なったり，3Uにはなかったものを中心。

### STMR：特殊機能タイマ

構文：STMR　*t1　n2　b3[4]*

t1 使用するタイマデバイス。（100mSecタイマとして扱われる）

n2 タイマ設定値

b3[4] 出力ビット

　b3[0] オフディレイタイマ

　b3[1] オフ後ワンショットタイマ

　b3[2] オン後ワンショットタイマ

　b3[3]　 オンディレイ+オフディレイタイマ

ld 　指令ビット　ani　b3[0] stmr T10 K10 b3[0] とするとフリッカになる。

　b3[1]　オフ／オンを繰り返す 　b3[2]　オン／オフを繰り返す

FX3Sシーケンサには，EEPROMメモリが標準装備

FX3G･FX3GCシーケンサには，EEPROMメモリが標準装備

FX3U･FX3UCシーケンサには，RAMメモリが標準装備

## 命令まとめ17：外部機器通信

※インバーター命令はPLC関連まとめに移動。

## ショートカットキー

F7 接点を作成。テキストボックスを空白にすると空接点が作成できる。

# GX Works3：プログラム設計

## 構造化に関して

Works3になっても構造化機能は全く進化していない。構造化プログラムが作り難い。

中度のステップ数のプログラムでは，無理やり全てをサブルーチン化する事で構造化が多少は達成できると思い実験中。20.04.03

但し，構造化する必要の無いシンプルなプログラムやフルメモリ使う必要のある重度のプログラムはこの限りではない。

ファイル構成：

プロジェクト（例）

├ スキャン

│ └┬ MAIN （プログラムファイル）

│ 　├─ProgPou # ld SM400 call P*n* だらけになる。P0~79まで。

│ 　│ （プログラム # P0はMain （Pは4095まで，P63は使用不可）

│ 　│ 　ブロック） # 最後はFENDで終わる。以降すべてがサブルーチン（以下Subとする）。

│ 　├─Main(P0) # 全体の流れが見通し良くなるように書く。

│ 　│ #　Main内部でSubは禁止。サブプログラムを呼ぶ構造にする。

│ 　│ #　また，複数のサブプログラムに対して同時にCallをトリガ

│ 　│ #　したい事もあるだろうから，その際はP80～99を用いる。

│ 　├─Middle(P1) #　複雑な中核処理を書く。（要らないかも？20.04.03）

│ 　│ #　Mainを読みやすくする為なら，こちらを読みにくくしても良い。

│ 　│ #　ここでの内部Subは控える。但しP1100～1119を利用可能。

│ 　├─ 以下サブプログラム(P2以降)（ここが重要！）

│ 　│ #　**１デバイス／１機能につき１つのプログラムブロックを作成**

│ 　│ #　原則，対象機能に必要な処理をもれなくここに書き込む。

│ 　│ #　内部Subは1000＋10n番台を使う。例）ここでは1020番台。

│ 　│ #　別のサブプログラムのCallは**厳禁**

│ 　└─Config (P2000) # GOTなどからの設定値の処理

│ #　サブプログラム毎の設定があればP2000＋10nを使う

│ #　※但し

├ 定周期 # これ以下のプログラムファイルは原則使わない。

補足）この構想に至った理由：

メンテナンス性を良くする為には，１機能１ファイルにした方が良い。

但し１つのファイルでFENDを使ったら，実行順序がこれ以下になるファイルではサブルーチンしか書けなくなる。つまり２つ以上のファイルでFENDを書けない。

⇒それくらいの事，なんでできないかな。20.04.02

反面，サブルーチン内でCALLを呼ぶのは問題ないので，いっそのこと全てサブルーチン化した方が，１機能１ファイルを達成できる。

分かっている事の箇条書き

・P1 ；SRET；I1 ；IRET；P2 ；SRET； とPとIを順不同に記述するのは問題ない。

## 概要

・プログラム種類は，ラダー，ST，FBD/LDが選択可能

※FBD/LD (Function Block Diagram/ラダー)

特殊処理を行うブロックなどをグラフィック的に結線する

・プロジェクト内に種類の異なるプログラムを混在させる事ができる

## プロジェクトのファイル構成

・１つのPLCに対して，１つの「プロジェクト」の書き込みが可能

・１プロジェクトには，最低１つのプログラムファイルが必要

プロジェクト（例）

├ プログラムファイル１

│ ├─ プログラムブロック１

│ ├─・・・

│ └─ プログラムブロックn

│

├ プログラムファイル２

│ └─ プログラムブロック１

│

└─ プログラム部品

├─ ファンクションブロック

└─ファンクション

### プログラムの順番

プログラムファイル内でプログラムブロックの順番を変更できる。

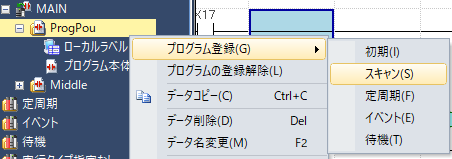
プログラムファイル（例： ）を右クリック → プログラムファイル設定

（注意）ナビゲーションウィンドウ内でも順番を変えられるが，これは実際のプログラム順序とは全くの無関係という，かなり紛らわしいGUI仕様なので注意する事！面倒だが，手動で常に実際とナビゲーションの順序を合わせておくと良い。

### プログラム種類

#### 共通項目

標準で，スキャン，定周期，イベント，待機，実行タイプ指定なし，未定義，のフォルダ分けがされており，プログラムブロックのフォルダ分け，機能分けができる。

すでに作成したプログラムを右クリック

→ プログラム登録　　で後からでも変更可能。

（右図参照）

使ってみての感想(19.12.19)

フォルダ分け自体は良いアイディアだと思うが，実行タイプごとにプログラムファイルが分かれるのはすごく操作しにくい。更に，ぱっと見で何スキャン毎に動作するか，などが分かり難い。

敢えて全部の実行タイプを「スキャン」にするのが，旧式だが，むしろベスト。

##### 設定

新規作成時の内容を変更したい場合は：

“初期”や “スキャン” などのプログラムアイコンを右クリック → プログラム設定　　または，

CPUパラメータ → プログラム設定 → プログラム設定：詳細設定

#### 初期

STOP → RUN時に一回だけ。

・基本的には接点（左）側はld SM400(M8000)。（自動的にイニシャルスキャンのみになる）

・数スキャン必要な処理は避ける。SEND, RECV など

#### 待機

#### 定周期

プログラムファイル作成時に，一定の割込み周期をもった割込みデバイスが選択される。

I28：100mSec周期，I29：40mSec周期，I30：20mSec周期，I31：10mSec周期

⇒ 実行周期を細かく指定できないのは，かなり不便

但し，P（ポインタ）でなくI（り割込み）な点は使い道があるかも。

#### イベント

起動デバイスには，ビットデータ，割込みデバイス，時間経過　の内１つを選択できる。

但しトリガ設定は１プログラムファイルに付き１種類。**プログラム部品ではない**ので，ナビゲーションウィンドウのツリーは煩雑になる事まちがいなし。

ビット

「ビットデータのON」（GUIの表現が分かり難い！）で任意の内外部デバイスが選択できる。

但しデフォルトではSM401が指定され，なぜこれをデフォルトにしたのか全く意味不明。

割込み

時間経過

「時間経過」では分かり難い。時間「周期」とすべきだった。⇒ 指定時間周期でなんどでも発生する。

sec, mSec など細かく設定できる。（これは定周期で実装すべき機能では？）

（どうやって止めるの？）　出力及びタイマの現在値をクリア　の意味がわからない。

#### なし

同系機種でのプログラム共有が可能。オプション機能をの実行タイプを「なし」にしておく。

※以下の方法でプログラムタイプは変更できる。

Works3のナビゲーションウィンドウ→パラメータ→FX5U → CPUパラメータ

→プログラム設定→プログラム設定→プログラム設定：詳細設定

## 具体案

プログラムの扱い

・プログラムの規模にもよるが，３～４つくらいに分ける。

main ：大きな流れや仕様が見渡せるように書く

middle ：main以外のプログラム。特にパッと見では良く分からない複雑なプログラム部。

optional ：オプション品の処理。オプション違いの場合はここだけ修正すれば良い

sub\_prog ：プログラムの先頭がFENDになる。サブルーチンはすべてここに記述。

（三菱の仕様に合わせて，仕方なくこの様な方法をとる）

デバイスの扱い

・ラダー図の左側で使うデバイスの番号の100の位は，極力同じにする。

例）機械制御系100番台，GOT操作200番台， など

・一時変数（デバイス）はD1000台以降を使用。（メインデバイスを際立たせる為）

・一時変数も個々に用意して，可能な限り使い回さない。その方がデバッグが確実になる。

構造化の積極的回避

ポインタ，自作ファンクションブロック，インデックス，FOR~NEXT，インラインSTなどの構造化機能は極力使わないようにする。混乱，バグの温床。

ラベルを使用する際も，可能な限りグローバルラベルだけにする。

以下はそれぞれの理由

FOR ：FORを使った方がむしろ行数が多くなる

インラインST：標準では変数が全てint16とみなされる上，型変換が無い。実数計算では不便。

（ラベルを用意して先に型定義しておく以外に，あまり良い方法がない）

## 構文

### FOR

一般的なプログラムで使えるforループが可能

FOR

（ld= 条件　BREAK　残数　ポインタ）

NEXT

BREAKの詳細：

第一引数の残数は，FORループを抜ける時のループ回数の残数の値が格納される。（なぜ残数？）

第二引数には，ループを抜ける先のポインタを入れる。

ループカウンタとインデックスを併用する時の注意点：

対象がインデックスであってもMULの結果は32bitになるようなので注意。

具体的には，下の例の場合はZ1は常に0 になる

mul D100 K20 Z0

以下の理由から，**FORを使うメリットはあまり無い**と考える。

・自分でわざわざカウンタ変数と，そのステップ数処理を用意しないといけない。なぜfor部分（ループ開始行）を１行で済む仕様にしなかったのか。

・↑の詳細：ループカウンタとなるデバイスの初期化，FOR，カウンタ変数のカウントアップ（INCなど），step処理（MULなど），実際の処理，NEXT　でただのループ処理なのに6行も必要となる。

　⇒　「実際の処理」の部分が長い場合，６ループ以上必要な場合以外はメリットが少ない。

・for とnextは母線と直接繋ぐ必要がある。つまり条件に合わせてスキャン数を節約する事はできない。

・ついでに言うと，FOR は左端に表示されるインターフェイスにすべきだった。非常に見にくい。

・とにかくラダー図が見にくくなる。メンテナンス性向上の為，**むしろFORの使用は積極的に避ける**べき。

## ファンクション

### 特徴

・ファンクション内変数の利用が可能。

・ファンクションブロックと違って戻り値がある

### 注意点

・ファンクション内部の変数は呼び出し毎に初期化されるわけでも無く，値が保持される。

（なんでこんなアホな仕様にしたのか）

### 補足

・EN/ENO　：Function Enable の意らしい。「EN/ENOを使用する」にした場合，ENと言う実行条件ビットを設定し，ENがOnの時のみファンクションを実行する。

・ファンクションブロック風にラダー図の真ん中に配置するのではなく，通常の関数の様にOUT側に記述する方式にしたいが，そういった設計になっていいないらしい。あんまり使いやすさとかは考えていないらしい。

・結局ファンクションブロックとの違いはほとんどない。

・FUNCall [ファンクション名]　で呼び出すらしいが，FUNCall が無い。

## ファンクションブロック(FB)

・

概要

FBを使うのは標準化，「誰が作ってもほぼ同じ」を助ける。

### ファンクションとファンクションブロックの違い

|  |  |  |
| --- | --- | --- |
| 項目 | ファンクション | ファンクションブロック |
| 戻り値 | 有り  呼び出し時，出力側にファンクション名のエリアがあるが，それが戻り値らしい。 | 無し  ユーザー任意でret変数を定義可能なので，戻り値自体は得られる。むしろ，これで良い。 |
| 内部変数 | 破棄？？？　初期化されないが？？？  モニタができない | 保持？？？  内部変数と言うより，戻り値の値は常にmovされるような仕様になっているらしい。 |
|  |  |  |

⇒　結論：違いはほとんどない。

### マクロ型とサブルーチン型

マクロ型

コンパイル時に（？）呼び出し元に展開される。処理速度が速い（？）

サブルーチン型

ポインタと同じく，呼び出し時のみスキャン数が増える事になる。プログラムのサイズが小さくなる。

### ビット引数について

FBの数少ない利点として，入力のビット引数（デバイス）を直列で処理できる。

並列でも処理できるが，無駄に縦幅を増やす事になるので，極力控えた方が良い。

### FBの使いどころ

・同じ計算，又は使うデバイスが異なるだけで同じ処理が数回ある。△

１つのFBを使いまわす場合は，引数を受け持つデバイス（ラベル）を用意する必要がある。

使用回数と同じ数のFBを用意する時は，ラダー図が見にくくなる。

・ポインタへのジャンプの代わりに用いる 〇

通常の関数の概念に反するが，引数を使わずグローバル変数（デバイス）をそのまま利用する。

FBの引数には，メインとなる引数・戻り値のデバイスだけ用意する。

単純にポインタを使用する時より，メインとなるラダー図が見やすくなる（と思う）

### FBラベル

ユニットのデバイスに直結したラベルが標準で用意されている。但し数も多く探しにくいので，テキストボックスを使って検索した方が早い。もしくは類似のラベルを探して要素番号とみられる部分を変更してみる，など。

同名のラベルをコイル側で使うとダブルコイルとなる。

## ラベル

先に結論

最終的に納品する段階では使わない方が無難。

但し，デバイス番号の割り付けなどが決まっていない最初の開発段階では非常に役に立つ。

概要：

注意点：

ラベル一覧の表示中，ソート中（列ヘッダに↓がついている状態）ではラベル追加ができない上，ソート中を記すマークが見にくいので，ラベル編集する時には注意する。（右図参考）

|  |  |
| --- | --- |
| VAR\_GLOBAL | 常用ラベル |
| VAR\_GLOBAL\_CONSTANT | 定数的に使うラベル |
| VAR\_GLOBAL\_RETAIN | 記憶ラベル。※D500以降の記憶デバイスと結びついたラベルは全てこの型である必要がある。  ⇒ ほとんどこの型になるから，型設定も茶番でしかない。 |

記憶（ラッチ領域を標準ラッチ領域に設定した）ラベルは，不揮発性メモリで保持を行う。（バッテリ不要）

## ST言語

概要

・２つ以上の処理を１行に書くことができる。 e.g. w\_val\_1:= 10; w\_val\_2 := 20;

・インデックス修飾も使用可能。

・関数の引数内に計算式を組み込む事も可能。

・バッファ番号も使える。

### インラインST

（操作）編集 → インラインST → インラインSTボックス挿入。又は出力（接点）側でstb と入力。

※インラインSTでは全ての変数（デバイス）がint16とみなされる上，計算式（右辺）による変数（左辺）の自動型変換が無い為，**実数計算などではむしろ不便**。よほど複雑な計算以外では使わない方がむしろ見やすくなる

### 文法概要

// １行コメント e.g. //コメント

(\*\*) 複数行コメント e.g. (\*コメント\*)

+ 加算，- 減算，\*乗算，/ 除算

:= 代入 比較は=

IF

IF タンクリミッタ = TRUE THEN

バルブ := FALSE; (\* リミッタがONしたためバルブを閉じる \*)

ELSE

バルブ := TRUE;

END\_IF;

# GX Works3：シンプルモーション

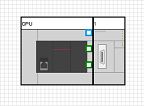
## はじめに

### ユニットの追加（Works3）

Works3のナビゲーションウィンドウ ： ユニット構成図

※先にドッキングウィンドウに「部品選択」が表示されている事を確認（右図参考）

出てない場合は　メニュー：表示(V) → ドッキングウィンドウ → 部品選択(E)

ユニット構成図画面で「部品選択」

FX5シリーズ→シンプルモーション → FX5-40SSC-Sを選択（ドラッグ）

→　 FX5Uの図の右側にドロップ。（右図参考）

今選択したシンプルモーションユニットの図をダブルクリックすると，

ユニットのパラメータ設定画面が表示できるようになる。

### 設定画面の出し方

Works3のナビゲーションウィンドウ → パラメータ → ユニット情報 → FX5-40SSC-S

→ シンプルモーションユニット設定（ダブルクリック）

### 軸番号の設定（物理番号）

MR-J4-10B に内臓のロータリースイッチで物理的に指定する。

なお，dipスイッチはすべてOffにする。（16軸以上の場合はdipスイッチと併用）

**※ロータリースイッチでの軸番号は０から始まる点に注意！**

### 軸番号の設定（ソフトウェア）

シンプルモーションユニット設定

追加したい軸を選択してダブルクリック。

必要情報を入力すると軸の図の色が変わる。

## 基本パラメータ

### 基本設定画面

Works3のナビゲーションウィンドウ → パラメータ → ユニット情報 → FX5-40SSC-S

→ シンプルモーションユニット設定（ダブルクリック）

又は，Works3のナビゲーションウィンドウ → ユニット構成図

→ シンプルモーションユニットの図をダブルクリック

### パラメータ（基本）

以下が一番基本となるパラメータと，それに付随するパラメータ。

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.1 | 単位設定 | 0:mm |
| Pr.2　(32bit)  G2+150n | 1回転あたりのパルス数 | =サーボの分解能　変更する必用はほぼ無い。  但し，Pr.3, Pr.4 との兼ね合いで値を変える事もある |
| Pr.3　(32bit)  G4+150n | 1回転あたりの移動量 | モーター１回転での負荷の移動量　単位はPr.1  （参考） |
| Pr.4  G1+150n | 単位倍率 | １単位辺りのパルス量　（参考）[電子ギア](#_電子ギア)  値：1，10，100，1000 |
| Pr.12 | ソフトウェアストロークリミット上限値 | 単位設定をdegreeに変更すると，この項目の初期値が適応範囲外となってしまうので修正する。  ⇒ -35999999～35999999 となる  （関連）Pr.13　ストロークリミット下限値 |

パラメータ設定の注意点

Pr.1～Pr.90，Pr.95，Pr.116～Pr.119，Pr.127はCd.190シーケンサレディ立ち上がり時に適用される。

また，Cd.190のリセットには数スキャン必要らしい。そうしないと，Cd.190に伴った準備完了やサーボOnがうまくリセットされず，制御ができなくなる。

（書き込み要求のデバイスがあったような？19.12.13　とにかくCd.190のリセット時に更新される）

Works3を使って設定値を書き込むと，シーケンサの再起動毎にその設定値が書き込まれる（初期化されてしまう）　必要に応じてイニシャルパルス時にシーケンサの記憶デバイスでシンプルモーションユニットに書き込む。

### 電子ギア

SSCにおいては，電子ギア＝１単位辺りのパルス量。

電子ギア ＝

例）[Pr.3]1回転あたりの移動量が1000mm；単位倍率1の場合。

電子ギア＝4,194,304(Pr.2) / 1000(Pr.3) = 4194.304

10mm動かしたい場合は 10×電子ギア　なので41943パルス送られる事になる。

補足

・端的に言えば，電子ギアの値が同じになれば，全く同じ結果を得られる。

（例）Pr.2の値を1/10にしても，Pr.4の値を10にしても全く同じ事。その意味では**Pr.4は不要**とも言える。

・電子ギアを変えると，全てのパラメータを見直す必要が出るので注意。

### 電子ギアによる単位系の扱い

電子ギアによってプログラム内部で扱う値は大きく変わってくるが，**マニュアルに（少なくとも分かりやすく）記載がない**のでここにまとめる。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pr.1 | 標準の  単位倍率 | Works3 設定機能  （常に以下の表示） | 速度系  指定方法 | 備考 |
| 0:mm | ×10-4 | 0.1μm単位 | /min |  |
| 1:inch | × 10-5 | 0.00001 inch単位 | /min |  |
| 2:degree | × 10-5 | 0.00001 degree単位 | /min | 標準は360に0が5つと思えばよい。 |
| 3:pulse | そのまま |  | /sec | pulseだけpulse/secで速度指定する。 |

pulse以外の上限値は2,000,000,000；pulseの上限値は1,000,000,000　（参考）signed32の上限値は2,147,483,647

（詳細）

Pr.1でdegree，電子ギアを Pr.2=4,194,304/**36000** に設定すると，プログラム内部で扱う数値は0.01度単位（× 10-2）となる。これは位置決めに**10**の値を使うと，モーターが**0.1度**動くという事。但し，Works3の設定機能では**常にdegree× 105単位で表示**する為，同機能での設定値は**0.360000 degreeと表示される事になり，非常に紛らわしい。**

また，電子ギアを変えると**実現在値（後述）なども変化する。**例えば，この場合では電子ギアの単位倍率と標準の単位倍率は [× 10-2] と[× 10-5] なので**103の差**がある。この時，実際の変化に対し**実現在値は103倍変化**する。（1/1000回転[≒0.36度]の変化で，実現在値では１回転分の値が変化する）

但し，**速度系ではこれと異なる単位系を使っている**のがまた紛らわしい。電子ギアの単位系に**102**を掛けたものが適用され，具体的には上の例では電子ギアでの単位系が0.01度なので，速度系は1度単位となる。

### パラメータ（速度）

注意点

・各指令速度には以下のような条件があり，違反すると逐一エラーを出す。

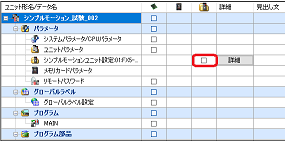
Pr.7 始動時バイアス　　≦　　実際の指令で使われる速度　　≦　　Pr.8速度制限値

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.7 (32bit)  G6+150n | 始動時バイアス速度 | 最低回転速度　ととらえても良い  [上記](#_運転速度に関する注意点)参照 |
| Pr.8 (32bit)  G10+150n | 速度制限値 | 絶対制限値。 |

実際の指令で使われる速度（抜粋）：

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.46 (32bit)  G74+150n | 原点復帰速度 | 条件：Pr.7 ≦ Pr.47 ≦　Pr.46 ≦　Pr.8  シンプルモーションユニット設定機能で設定しても良い  ※Pr.47は近点DOG式（Pr.43=0）の時のみ有効  ※原点復帰方式がデータセット式（Pr.43=6）の時はどちらも必要ない。 |
| Pr.47 (32bit)  G76+150n | クリープ速度　※機械原点復帰時，近点DOGがON後の速度 |
| Pr.31 (32bit)  48+150n | JOG速度制限値  関連（Cd.17 JOG速度） | 条件：Pr.7 ≦ Cd.17 ≦ val ≦ Pr.8 |
| Cd.17  G4317+100n | JOG速度 /min | 条件：Pr.7 ≦ val ≦ Pr.31≦ Pr.8 |
| Da.8 (32bit)  G6004+  1000n+10m | （位置決め）指令速度  **（停止中にしか変更できない）** | Pr.7 ≦ val ≦ Pr.8 速度制限値  （関連）cd.14? Cd.140？  （関連）Md.27，Md.33  （参考）[運転速度に関する注意点](#_運転速度に関する注意点) |

### サーボパラメータの書き込み

Works3→オンライン→書き込み　の時に

シンプルモーションユニット設定にもチェックを入れて，先にパラメータを書き込んでおく必要がある。

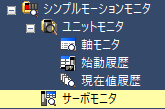
以後，サーボパラメータ設定でFX5U経由での軸書き込みが可能になる。（サーボアンプと直接接続する必要はない）

補足）

インテリジェント機能ユニットのバッファメモリはRAM。シーケンサの電源を落としたら消えてしまう。電源を再投入した場合にはフラッシュROMのデータを読み込む。フラッシュROMに値を書き込むかどうかは「詳細」ボタンで指定できる。

### 診断

軸モニタ

シンプルモーションに関する各パラメータのモニタができる。エラーコードもここで確認する。

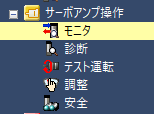
シンプルモーションユニット設定機能のナビゲーション →

シンプルモーションモニタ → サーボモニタ

（又は，シンプルモーションモニタ→ユニットモニタ → 軸モニタ）

「Works3の診断」だと，プロファイルが無いと言われ，診断ができない。（80SSC-Sのプロファイルは存在しないらしい）

サーボモニタ

従来のConfigurator 2 と同じ画面でモニタができる。複数の接続軸を同時にグラフに記録するなどが可能。

グラフ記録したのち，FFT解析で周波数の分布が見られる。

**シンプルモーションユニット設定機能のナビゲーション →**

## その他の基本設定

### ストロークリミット

ハードウェアストロークリミット

各サーボアンプのD型端子 (CN3) の2番(FLS)と12番(RLS)に5VDC信号をB接で入れる。

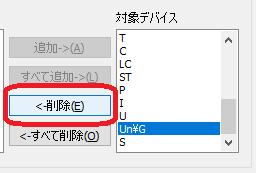
**なお，この信号はPLCでも出せる。**Pr.116，Pr.117 信号選択　をそれぞれ「2:バッファメモリ」 とし，Cd.44で実際の信号を指令する。（１軸4bit分）

ソフトウェアストロークリミット

「送り現在値」か「送り機械値」（※１）が範囲内かどうか（※２）で判断する仮想的なリミットスイッチ。

Pr.15で無効にする事もできる。 ※１ Pr.14で決める。 ※２ Pr.12が上限値，Pr.13が下限値。

### デバイスコメントに関する注意点

ラダー上で機能デバイス（U1\G31500.1　など）を使うと自動的にコメントが入るが，これはユニット構成図などでインテリジェントユニットを追加した時に共通デバイスコメントを書き換えているに過ぎない。その為，他からデバイスコメントの.csvファイルをインポートすると機能デバイス用のコメント部分が空白で上書きされてしまう事がある。

⇒インポートする際に「Un\G」を対象から外せば防ぐ事が可能。（右図参照）

なお，ファイルにエキスポートする際にも同じ様に「Un\G」を対象から外す事が可能な様だが，インポートの際に意図的に対象を外さないと結局空白の文字列が上書きされる為，まったく無意味な機能と言える。

### 強制停止( EM2 )

**サーボアンプのD型端子 (CN3)** の20番に，5VDC信号をB接で入れる必要がある。

又は，各軸のサーボ強制停止を無効にする（PA04）。

サーボパラメータ → 基本設定 → 強制停止(\*AOP1)

（補足）Cd.44　外部入力信号操作というソフト信号パラメータがあるが，EM1, EM2 は無い。

### 緊急停止( EMI )

強制停止とは別に，全軸に適用される「緊急停止」がある。**シンプルモーションユニットの外部入力端子**（D型端子 ） の10番に，24VDC信号をB接で入力する。

High：17.5VDC以上／3.5mA以上　　Low：7VDC以下／1mA以下

もしくはPr.82で緊急停止を無効にする。

（補足）EM2に同じく，ソフト信号であるCd.44に，EMI は無い。

### 停止の種類

設定の必要は無いので蛇足にはなるが，他にも急停止（停止グループ）というものが存在する。

比較表：

|  |  |  |  |
| --- | --- | --- | --- |
| 停止区分 | 内容 | 対象軸 | 結果 |
| 緊急停止 | モーションユニットへの外部からの緊急停止信号 | 全軸 | 即停止  → サーボOFF |
| 強制停止 | サーボアラーム，サーボアンプへの強制停止信号 | 軸ごと | 即停止  → 様々 |
| 停止グループ1 | ハードウェアストロークリミットによる停止 | 軸ごと | 減速or急停止  → エラー中 |
| 停止グループ２ | CPUユニットのエラー発生，"[Cd.190]シーケンサレディ信号"のOFF | 全軸 | 減速or急停止  →エラー中 |
| 停止グループ３ | CPUユニットからの軸停止信号  CPUユニットの軽度エラー発生 | 軸ごと | 減速or急停止  → 停止中 |

### パラメータ（その他の基本設定）

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.15  G23+150n | ソフトストロークリミット有効／無効設定 | **0： 有効** 1：無効 |
| Pr.82  G35 | 緊急停止有効／無効設定  （全軸に送る） | **0：有効**　1：無効  関連：Md.50 緊急停止入力値 |
| Pr.116 | FLS信号選択: 入力種別 | **１：サーボアンプ**　２：バッファメモリ  2 を選択した時は Cd.44(G5928)で値を設定する。  ※Forward/Reverse Limit Signal の略 |
| Pr.117 | RLS信号選択: 入力種別 |
| Pr.118 | DOG信号選択: 入力種別 | 原点復帰の際にクリープ速度に切り替わるポイントを指令する信号の事。 |
| Cd.5  G4302+100n | 軸エラーリセット: 入力種別 | 数スキャン間出し続ける必要があるらしい。  ※最初はエラーを連発するので，実装した方が良い。  ワーニングもリセットする？20.12.03 |
| Cd.44 (32bit)  G5928 | 外部入力信号操作  G5928 1軸分4bit　　×8軸 = 32bit分  G5928 1-4軸 ； G5929 5-8軸 | .b0 軸１上限リミット信号(FLS)  .b1 軸１下限リミット信号(RLS)  関連Pr.116,117 信号選択；Md.30 |
| Md.23  G2406+100n | 軸エラー番号 |  |
| Md.24  G2407+100n | 軸ワーニング番号 |  |
| Md.30  G2416+100n | 外部信号 | .b0下限リミット信号(FLS)  .b1上限リミット信号(RLS) |
| Md.31  G2417+100n | ステータス（抜粋） | b9軸ワーニング検出 b13エラー検出 |
| Md.108  G2477+100n | サーボステータス（抜粋） | b0: レディ ON 　　 b1: サーボON  b7: アラーム中　　b15: 警告中 |
| Md.114  G2488+100n | サーボアラーム | SSCNETの時とSSCNET/Hの時で異なる  サーボドライバの上のLED表示の  SSCNET/H 35.1 ⇒ 0x0351（dec:849）  SSCNET \*00 |

## JOG運転

### 条件

以下の条件が必要（●必須 　・安全の為条件を満たす事を推奨）

●同期用フラグON ●準備完了信号がON ●サーボON

●Pr.31 JOG速度制限値の設定　　●Cd.17 JOG速度の設定

●ハードウェアストロークリミット の範囲内 ・Md.141 Busy信号がOFF

詳細

Md.140.b1同期用フラグ：

PLCに電源を入れて，シンプルモーションユニットへがアクセス可能状態になると自動的にON。

Md.140.b0準備完了信号 ：

シーケンサレディの立ち上がり時に，SSCの**基本**パラメータに問題が無ければ自動的にOnする。

（補足）Cd.190シーケンサレディ：≒ SSCの基本パラメータの検証**指令**デバイス　と捉えた方が良い。

サーボON ：Cd.191全軸サーボONを用いる。

Pr.31 JOG速度制限値：Works3のユニット設定を用いた方が楽。

Cd.17 JOG速度：

１分毎の速度を指定。単位はPr.1 で指定したもの。

補足：rpmで指定したい衝動に駆られる。Pr.3　1回転あたりの移動量 × rpm値　の計算結果を利用すれば良いが，**Pr.3のパラメータの単位と，Cd.17 JOG速度の単位は異なる**点に注意する。

例えばPr.1が2:degreeの場合，Pr.3は0.00001 degree（１×10-5）だが，Cd.17では0.001degree（１×10-3）

ハードウェアストロークリミットの範囲内：（参考）[ストロークリミット](#_ストロークリミット)

### 全軸サーボOn信号の出し方

参考例として，以下の様に全軸サーボOnを出すと良い。

ld Md.140.b1 [同期用フラグ] out Cd.190 [シーケンサレディ]

ld Md.140.b0 [準備完了] out Cd.191 [全軸サーボOn]

※注意！eラーニングのサンプルはかなり酷い事が判明。参考にしない方が良い。20.11.04

#### 機能デバイスのモニタ

初めて触ると，各ビットの動向が気になるはず。機能デバイスも一応モニタする事はできる。

Works3のメニュー：ツール → オプション → モニタ → ラダーエディタ → バッファメモリ，リンクメモリをモニタする　を「はい」に。

※スキャンタイムが大幅に増えるので，実用の前に元に戻す事！

### ファンクションブロック(FB)

FB（ファンクション・ブロック）を使う際は，Works3のドッキングウィンドウの「部品選択」→

ユニット → FX5-40SSCを右クリック → ユニットラベルを追加　をしておく。

JOG運転のFBは，部品選択 → ユニットFB → FX5-40SSC-Sの「JOG/inching operation FB」

なお，引数の２つ目「ユニットラベル」はさきほど追加したユニットラベルの左側にドロップする。

（注意点）FBの対象軸の変更は，**実行指令の立ち上がり時のみ有効**。FBの実行指令を**常にONしていると，軸の変更ができない。**

公開ラベル

SetPositioningData など一部FBでは「公開ラベル」が提供される。

公開ラベルの使い方は 「FB名．公開ラベル名」といった形

例）　mov K0 M\_FX5SSC\_SetPositioningData\_00A\_1.pb\_dPositAdr

ただし，**分かり難いしラベル名が長いのでお勧めできない**。

トラブルシューティング（公開ラベル）

エラー0x110E1957：デバイスまたはラベル種別が不明です

32bitラベルに対して16bit mov などを行うとこのエラーが起こる

FBのラベル

ドラッグ＆ドロップなどでFBを追加するとFBのラベルが自動的に追加されFBを削除しても残る。

これは追加したプログラムのローカルラベルで管理されているので，こちらで削除できる。

（反面，ユニットラベルはグローバルラベルで管理されている）。

（補足２）シンプルモーション用FBについて

以下の理由から，FB を利用するメリットはあまりない。（重要度順の箇条書き）

・FBは縦に長くスペースを取るのでラダーが非常に見にくくなる。

・シンプルモーション用FBのほとんどは，バッファメモリの書き込みによる指令で実装できる。

・FBの各パラメータが期待する振る舞いでない事があり，いちいちマニュアルを見る必要がある。

・FBを利用すると，インテリジェントユニットで扱うパラメータの理解が進まない。

但し，FBを利用した方がメンテナンス性が良くなるケースも存在する。例えば，位置決め指令をインデックス（Z）を利用して実装する場合を考える。

位置決め始動ではG4300+100nで100倍系，軸番号が1000倍系，位置決め番号が10倍系なので，３，４種類のインデックスを用意する必要があり，それに伴う計算処理の事を考えると，ラダーでは非常に多くの行数（ステップ数）を必要とする。このような場合はFBを使った方が良い。

※とはいえ，もともとバッファ番号を1000倍系で統一した設計にすれば良かったのでは…とも思う。

### パラメータ（JOG）

パラメータ設定の注意点

Pr.1～Pr.90，Pr.95，Pr.116～Pr.119，Pr.127はCd.190シーケンサレディ立ち上がり時に適用される。

また，Cd.190のリセットには数スキャン必要な様子。そうしないと，Cd.190に伴った準備完了やサーボOnがうまくリセットされず，制御ができなくなる。

※**太字**はデフォルト値

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.31 (32bit)  G48+150n | JOG速度制限値  （関連） JOG速度 | 条件：Pr.7 ≦ Cd.17 ≦ val ≦ Pr.8  （参考）[運転速度について](#_運転速度について) |
| Cd.16  G4317+100n | インチング移動量 | val = K0 でJOG運転となる。 |
| Cd.17 (32bit)  G4318+100n | JOG速度 /min  ※インチング（Cd.16<> K0）の時，この値は無視される　どの値が適用される？19.10.28 | 条件：Pr.7 ≦ val ≦ Pr.31≦ Pr.8  Pr.1単位設定によってパラメータの桁数が異なる  mm mmの値×102の値を代入（0.01mm単位）  inch インチの値×103の値を代入（0.001inch単位）  degree 角度の値×103の値を代入（0.001度単位）  pulse そのままの値を代入 |
| Cd.181  G30101+10n | 正転JOG始動 |  |
| Cd.182  G30102+10n | 逆転JOG始動 |  |
| Cd.190  G5950 | シーケンサレディ信号 | シーケンサプログラムで能動的に出す。 |
| Cd.191  G5951 | 全軸サーボON | シーケンサプログラムで能動的に出す。  Cd.190 がOnするときに同時もOnさせれば良い。  （関連）Cd.100 |
| Md.140  G31500 | ユニットステータス | b0 準備完了信号  Cd.190シーケンサレディ信号ONで異常が無いとOnする  b1 同期用フラグ  電源を入れて，シンプルモーションユニットへがアクセス可能状態になると自動的にON。これがONしないと何もできない。 |

#### その他注意点

JOG運転ではMd.141 Busy はOnしない。その為，

ld X0　ldi Md.141（軸１Busy）　 out Cd.181正転JOG始動 といった書き方は可能。

## 原点復帰

※Pr.55=1の時は原点復帰の必要がない。（≒絶対位置の装置ではない）

### 条件

以下の条件が必要（●必須　・安全の為条件を満たす事を推奨）

●JOG運転に必要な条件を満たす。

●零点を通過する（違反するとエラーになる）

（補足）原点復帰なしでも，JOG，インチング，パルサ，定寸送り　での運転は可能。

（注意点）原点復帰の移動先は「Pr.45原点アドレス」の設定値で，必ずしもアドレス０地点では無い。

#### 原点復帰が必要な場合

・Pr.55原点復帰未完時動作設定 = 0: 位置決め制御を実行しない の時。

⇒ 逆に言えば，これを**1:位置決め制御を実行する にすると原点復帰の必要が無い**。

・システム立上げ後，一度も機械原点復帰を実施していない（20.11.06 Pr.55関係ある？）

・機械原点復帰始動して機械原点復帰が正常に完了するまで

・シンプルモーションユニット内の絶対位置データの消失。（エラー093CH「原点データ不正」発生)

・サーボパラメータの「回転方向選択／移動方向選択(PA14)」の変更

・サーボアラーム「絶対位置消失」(アラーム番号: 25) 発生時

・サーボ警告「絶対位置カウンタ警告」(警告番号: E3) 発生時

※但し，絶対位置システムでない時は，以下の時に要求してくる。

•システム電源投入時，またはリセット時 • サーボアンプ電源投入時

（関連）PA.03 絶対位置検出システム

### 原点復帰方式

#### 近点DOG式

リトライ：RLS(Reverse Limit Switch) でモーターを反転する事。ドグ点よりRLSに近い位置から復帰時に必要。

カウント式：ドグ点がOnしてから零信号がｎ回数あった地点を原点とする。

#### データセット式

回転や無限長などの相対（インクリメンタル）機械系で用いる。

原点シフト：ソフト的な原点位置の加減調整値。

### その他

#### 電源OFF時の絶対位置保持

絶対位置で制御を行う場合はサーボにバッテリーと，各サーボのPA.03 絶対位置検出システムをOnにしないと絶対位置情報を記憶せず，毎回原点復帰を行う事になる。

⇒ シンプルモーションユニット設定 → サーボパラメータ：構成品

絶対位置検出システムの選択：有効（絶対位置検出システムで使用する）

（MRConfigurator2：サーボパラメータ PA03と同等）

#### 高速原点復帰

機械原点復帰で確立された原点へ位置決めをする。（先に原点復帰済である事が条件）

⇒ 表現が分かりづらい。「定義済原点への再復帰」などとすべきでは。

Cd.3（G4300+100n）にK9002を格納し，Cd.184位置決め始動信号をOnする。

その際，Pr.46 原点復帰速度が用いられる。

19.11.20現在，未だ成功していない　→ 目標位置を0として普通に位置決めした方が早いしメンテナンス性も良い。

### パラメータ（原点）

注意点

Pr.1～Pr.90，Pr.95，Pr.116～Pr.119，Pr.127はCd.190立ち上がり時に適用される。

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.43  G70+150n | 原点復帰方式 | 0: 近点ドグ式  6: データセット式　　など |
| Pr.44  G71+150n | 原点復帰方向 | 0：正転　　1：逆転 |
| Pr.45 (32bit)  G72+150n | 原点アドレス | 機械原点とするABSアドレス値の事。必ずしも=0ではない。 |
| Pr.46 (32bit)  G74+150n | 原点復帰速度 | 条件：Pr.7 ≦ Pr.47 ≦　Pr.46 ≦　Pr.8  （参考）[運転速度について](#_運転速度について)  ※Pr.47はPr.43が0：近点DOG式の時のみ有効  ※Pr.43が6: データセット式の時はどちらも必要ない。 |
| Pr.47 (32bit)  G76+150n | クリープ速度　※機械原点復帰時，近点DOGがON後の速度  通常原点復帰速度の10%位を指定する。 |
| Pr.55  G87+150n | 原点復帰未完時動作設定"  ⇒ 位置決めの禁止　の意；=0 は，運転（位置決め）実行でエラーとなる。少し分かり難い表現。 | 0: 位置決め制御を実行しない（エラーを出して止める）  1: 実行する＝位置決めの許可（運転続行） |
| Cd.3  G4300+100n | 位置決め始動番号 | K9001：原点復帰　K9002：高速原点復帰 |
| Cd.184  G30104+10n | 位置決め始動信号 | 0：off　　　　1：on |
| Cd.44  G30104+10n | 外部入力信号操作  G5928 1軸分4bit　　×8軸 = 32bit分  G5928 1-4軸 ； G5929 5-8軸 | .b0 軸１近点ドグ信号(DOG) |
| Md31  G2417+100n | ステータス | b3 原点復帰要求フラグ |
| Md.119  G2476＋100n | サーボステータス２ | b0零点通過 |
| PC17　サーボパラメータ  28480+100n | 機能選択C-4　(Z相通過設定) | **0: 電源投入後モータZ相通過必要**  1: 電源投入後モータZ相通過不要 |

## 位置決め

以下の条件が必要（●必須 　・安全の為条件を満たす事を推奨）

●JOG運転に必要な条件を満たす 　●機械原点復帰が済んでいる

・Md.141 Busy信号がOFF

### 位置決めデータ

概要：

１軸につき600個（※）のデータが作成でき，「運転パターン」で連続する位置決めデータを作成できる。また，位置決めアドレスはティーチングによる代入も可能。（Cd.38参照）

※Works3のデフォルト設定では100個までしか位置決めデータを表示できない。

Works3：ツール → オプション → インテリジェント機能ユニット → シンプルモーション

位置決めデータ表示指定：表示範囲　を選択範囲　にして[ 開始No.] と[ 終了No.] を指定。

操作：

シンプルモーションユニット設定：位置決めデータ→軸n位置決めデータ

#### 位置決め運転パターン（抜粋）

Da.1

終了 00：単一の位置決めデータのみの使用。これでも単一パターンの連続運転は可能。

連続 01：一度**減速を挟んで**，複数の位置決めデータを次々に実施する。

連続軌跡制御11：**減速を挟むことなし**に，複数の位置決めデータを次々に実施する。

Da.2 制御方式

単位がdegreeで位置決めデータがABSやINCでは0/360を超えるような制御ができない。（エラーになる）

また，**速度制御時はPr.21 速度制御時の送り量現在値　の設定**を変更しないと，現在値情報の変化がないため，その軸を入力軸とする同期制御は動作しなくなる。

#### 制御方式（抜粋）

03h：定寸送り1

インクリメントとほぼ同等だが，位置決め指令の直後に送り現在値をクリア（＝０）してから位置決めを開始する。（送り機械値はクリアされない）

0Ah：２軸の直線補間

X軸方向とY軸方向の２軸を用いて，ワークを斜め（円弧）に移動させる場合に用いる。

１軸のNo20で補間軸に２軸を指定した場合，軸２もNo20の位置決めデータを使う仕様。

83h：Loop

84h LEND(Loop END) と共に使ってループ制御する。

※各制御方式ごとに必要な設定パラメータがマニュアル応用編「10.1 データの種類：位置決め用パラメータの設定項目」（p329）に記載されている

### ブロック始動データ

概要

位置決め番号は7000番台を用いる。

※「ブロック始動データ」という名前に反して，位置決めデータの補助的な役割しか持っていない。

### 位置決め

運転

Da.6移動先アドレスの指定 → Cd.3位置決め始動番号を指定 → Cd.184位置決め始動信号をOn

⇒ 但し位置決めはFBを使った方が良いかも

（補足）Cd.184をオフにする事による停止ではMd.31.b15位置決め完了信号　はオンしない。

停止

Cd.180を利用する。なおCd.180は数スキャン間出し続ける必要があるらしい。

パラメータ

位置決めのバッファ番号は，位置決め番号も関係してくる。例えば，指令速度はG6004+1000n+10m 〔n= 軸番号－1，m= 位置決め番号－1〕 で，軸２の位置決め番号２番の指令速度はG7014となる。

（補足）軸番号が関係してくるのはDa.1~10まで。

#### 複数軸動かしたい場合

Cd.184位置決め始動信号はbit指令なっている為複数軸の起動が可能だが，FBを利用する際はintでの指令なので１軸ずつしか動かすことができない。Md.31.E位置決め完了を待って次の軸へ行くループ的処理を取るか，Cd.184を直接Onするしかない。

#### 位置決め始動信号のリセット

Cd.184位置決め始動信号を直接Onする場合は位置決め完了時（Md.31.Eがon）に能動的にCd.184をリセットする必要がある。

以下の様な形にする。

ld Cd.184 始動信号 and Md.31.b15 始動完了or Md.31.D エラー検出 anb ani Md.141　　rst Cd.184

#### 速度制御

指令停止信号をだすまでDa.8 指令速度に設定した速度のパルスを出力し続ける。

つまり，Cd.184運転指令とCd.180停止指令を出す必要がある。

注意点

また，停止指令による停止ではMd.31.b15 **位置決め完了が出力されない**。

"[Da.1]運転パターン"は，「位置決め終了」を設定する必要がある。

単位が「degree」の場合はソフトウェアストロークリミットチェックが行われない。

速度制御している複数軸の内１つがその速度制限値を超えると、**その運転速度の変化率で他の速度制御の軸の運転制御が抑制される**、という意味不明な仕様になっている。十分注意する。

（参考）FX5U シンプルモーションユニット（応用編）3.2位置決めデータの設定(P93)

（補足）速度・トルク制御の速度制御とは何か違うのか？2021-07-14

#### 単発位置決め

どのような制御方式であれ，指定アドレスきっかりに移動させたい時があるはず。

単発位置決め専用の位置決めデータを作っておくと良い。また，全軸共通の指令番号にしておくと良い。

#### FB利用（位置決め）

Works3のドッキングウィンドウの （縦表示）→  → ユニットFB

→ FX5-40SSC-Sの「Positioning start FB」

なお，引数の２つ目「ユニットラベル」は，非常に分かり難い。

先に部品選択：ユニットラベル → FX5-40SSCを右クリック → ユニットラベルを追加　で追加をしておき，をドラッグ＆ドロップ。

また，注意点としてFBの第３引数である対象軸の変更は，実行指令の立ち上がり時のみ有効。FBの実行指令を**常にONしていると，軸の変更ができない。**

⇒ いっその事，各軸の専用FBを軸の個数ぶん配置した方が良いかもしれない。

なお，不要なFBはナビゲーションウィンドウ：M\_FBLIB　の下階層にあるフォルダを右クリック → データ削除で削除できる。

また，位置決めに関してはバッファ値の直接利用は避けた方が良いかもしれない。

理由：パラメータ番号に軸番号（1000倍系），位置決め番号（10倍系）が必要で，インデックスを使用しても２つ以上のインデックスが必要。

注意点：

出力側の「正常完了」がOn しているのは見たことが無い。**絶対に信用しない事**！

⇒ かわりにMd.31.b15 の位置決め完了　を使う。

但しパラメータ設定のSetPositioningDataは使わない方が無難。

このFBの実行指令をOnすると，公開ラベルの値が全て書き込まれる。その為，ラダーで直接設定したパラメータがあっても，公開ラベルの方に0が入っていたりすると最終的な値は0となってしまい，結果的にエラー1A24が発生する。特に位置決め番号ごとのDa.2制御方式なんかが意図せず変更されてしまうと非常に危険。

*三菱のこういったマヌケ仕様，本当にいいかげんにして欲しい。*

もし使う場合は，FBで現在値が確認できるので，pb\_uCtrlSys(Da.2)，pb\_dPositAdr(Da.6)，pb\_udCmdSpd(Da.8) の値を注視する。

### エラー対応

1A24

以下の場合などに発生する

・誤って未設定の位置決め番号の位置決めを指令した場合。

・指令速度のパラメータ(Da.8) に0が入っている。（Cd.14は基本的には設定の必要は無い）

・指令速度(Da.8)のバッファ番号間違い。Cd.14は\*\*\*6だが，Da.8は\*\*\*4と少し紛らわしい。

・SetPositioningDataでなんかしらの必要パラメータが0で上書きされてしまった。

### パラメータ（位置決め）

※m ＝ 位置決めデータNo -1

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.21  G30+150n | 速度制御時の送り現在値 | **※初期値は送り「現在値の更新を行わない」**になっているので注意！！ |
| Pr.40  G59+150n | 位置決め完了信号出力時間 | 実際の位置決めが終わってからMd.31.b15位置決め完了を出力するまでの時間を設定。  初期値300[mSec] |
| Da.1　G6000 +  1000n +10m .b0~b3 | 運転パターン | 終了，連続，連続軌跡 |
| Da.2　 G 6000 +  1000n +10m .b8~b15 | 制御方式 | ABS，INC，直線，　など  現在値変更 |
| Da.3　 G 6000 +  1000n +10m.b4~b5 | 加速時間 |  |
| Da.4　 G 6000+  1000n +10m .b6~b7 | 減速時間 |  |
| Da.6 (32bit)  G6006  +1000n+10m | 位置決めアドレス／移動量  **（停止中にしか変更できない）** | 位置制御時の目標値　　※制御方式が位置決めでないといけない（ABS,INC又は定寸送り）  （関連）Md.32 |
| Da.8 (32bit)  G6004+  1000n+10m | 指令速度  **（停止中にしか変更できない）** | val < Pr.8 速度制限値  関連cd.14? Cd.140？  Md.27  （参考）[運転速度に関する注意点](#_運転速度に関する注意点) |
| Cd.3  G4300+100n | 位置決め始動番号 | 1-600：位置決めデータNo.  7000-7004：ブロック始動指定 (関連)Cd.4  9001-9004：特殊 |
| Cd.14 (32bit)  G4314+100n | 速度変更値 | （関連）Cd.15 |
| Cd.15  G4316+100n | 速度変更要求  **（位置決め中のみ有効）** | **※減速中に速度変更してはいけない。**  K1 を代入する。処理が終わるとシンプルモーションユニットがK0 を代入する。 |
| Cd.27 (32bit)  G4334+100n | 目標位置（アドレス）変更値 | 位置決め運転中の目標位置変更。 |
| Cd.28 (32bit)  G4336+100n | 目標位置変更値(速度) |  |
| Cd.29  G4338+100n | 目標位置変更要求フラグ  **（位置決め中のみ有効）** |  |
| Cd.180  G30100+10n | 軸停止 | 信号は数スキャン間出し続けないと停止しない。 |
| Cd.184  G30104+10n | 位置決め始動信号  (FBの位置決め開始のトリガ)  Md.141　Busy [ G31501 ] | **自動的にはoffしない**ので，直接ON(=K1)する時は，**offする処理を自分で書く。** ⇒  Md.31.F位置決め完了or Md31.Dエラー: ON ani  Md.141.0 Busy の時にリセットさせる |
| Md.31  G2417+100n | ステータス（抜粋） | .E位置決め完了 |
| Md.32  G2418+100n | 目標値 | 位置決め運転時の目標アドレス |
| Md.141  G31501 | Busy | G31501.b0が軸１用  G31501.b1が軸２用（以下同様） |

## 速度・トルク制御

概要

サーボモーターの出力トルク（＝電流量）を一定にする制御。

トルクは電流に比例する。

・速度制御モード

・トルク制御モード

・押当て制御モード

### パラメータ（速度・トルク制御）

※**太字**はデフォルト値

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Cd.138  G4374+100n | 制御モード切替え要求 |  |
| Cd.139  G2+150n | 制御モード指定 | 0:位置制御モード  10:速度制御モード  20:トルク制御モード  30:押し当て制御モード |

## サーボ調整

### チューニング

まずはワンタッチ調整で大まかに合わせておく

操作：モーションユニット設定機能のナビゲーションウィンドウ：サーボアンプ操作

→　調整 → ワンタッチ調整

※予め，オートチューニングモード２にしておく。

既に位置決め命令が準備できていれば「ユーザー指令方式」を

準備できていなければ「アンプ指令方式」を選択する。

※初期調整のJOGの時はここで止めておく。これ以下の処理は，位置決めデータができてからにする。

オートチューニングモード１

　操作：調整 → 

### マシンアナライザ

→　調整 → マシンアナライザ

### デジタルオシロ

シンプルモーション内の各種パラメータの値を連続的に記録し，グラフ表示する。

⇒ モーションユニット設定機能のナビゲーションウィンドウ：デジタルオシロ

※デジタルオシロには履歴が残る

→ 履歴管理

### ゲイン切替

#### 概要

設定方法：サーボパラメータ → ゲイン切り換え

※サーボドライバに元々ゲイン切り換え機能が付いていて，（シンプル）モーションコントローラではそれをOn/Offするだけ，といった感じ。その為，ほとんどがサーボパラメータでの設定となる。

なお，通常のゲインパラメータはPB06～10で，切り替え後はPB29～32 となる。

（制振制御PB19～22などのパラメータもあることに留意する）

#### 切り換え方法

|  |  |
| --- | --- |
| コントローラの制御指令 | Cd.180 でOn/Offする。Md.108.b4 でモニタする。 |
| 閾値を利用 | 指令周波数，溜まりパルス，モーター速度を閾値として設定できる。  具体的にはPB27に閾値を代入。 |

#### 注意点

・サーボのゲイン調整モードを「マニュアルモード」にする。

・機械運転中に切り替えるのは非常に危険。運転中は動作しないようにする事。

・切り替え後の設定値が有効である事

・ゲイン調整モードの「２ゲイン…」はX-Yテーブルなどでの補間運転で使用。ゲイン切り換えの事ではない。

参考

サーボアンプ技術資料集 7.2 ゲイン切り換え機能

### サーボパラメータ

|  |  |  |
| --- | --- | --- |
| 項目 | バッファアドレス | 備考 |
| サーボシリーズ | 28400+100n | αSTEP/5相(オリエンタルモーター株式会社製) を用いる時は val=97 にする |
| PAグループ PA01～PA18 | 28401+100n～28418+100n |  |
| PA19 | 64464+70n |  |
| PA20～PA32 | 64400+70n～64412+70n |  |
| PBグループ | 28419+100n～28463+100n |  |
| PCグループ | 64413+70n～64431+70n |  |
| PDグループ | 65520+340n～65567+340n |  |
| PEグループ | 65568+340n～65631+340n |  |
| PSグループ | 65712+340n～65743+340n |  |
| PFグループ | 65632+340n～65679+340n |  |
| Poグループ | 65680+340n～65711+340n |  |

このほかに同期制御エリア（サーボ入力軸パラメータなど）にもアクセスできるが，こちらは書き込み付加な項目があるので注意する。

代表的なサーボパラメータ

|  |  |  |  |
| --- | --- | --- | --- |
| No. | 名称 | アドレス | 備考 |
| PA09 | オートチューニング応答性 | 28409+100n |  |
| PA14 | 回転方向選択 | 28414+100n |  |
| PB06 | 負荷慣性モーメント比 | 28424+100n |  |
| PB07 | モデル制御ゲイン | 28425+100n | [rad/s] |
| PB08 | 位置制御ゲイン | 28426+100n | [rad/s] |
| PB09 | 速度制御ゲイン | 28427+100n | [rad/s] |
| PB10 | 速度積分補償 | 28428+100n | [ms] |
| PB11 | 速度微分補償 | 28429+100n |  |

（参考資料）

FX5U ユーザーズマニュアル（応用編）

10.2 バッファメモリアドレス一覧 – サーボパラメータ

10.9 メモリ構成とデータ処理 – バッファメモリのエリア構成

## 同期

### 入力軸パラメーターの設定

シンプルモーションユニット設定機能 → 同期制御パラメータ → 入力軸パラメータ

サーボを使用している場合：サーボ入力軸種別(Pr.300)

指令生成軸を使用する場合：（指令生成軸）有効設定(Pr.340)

### カムデータの作成

モーションユニット設定機能のナビゲーションウィンドウ：

カムデータ → 右クリック → 新規作成

「カム曲線」のComboboxの選択項目で曲線を滑らかにしたりできる。

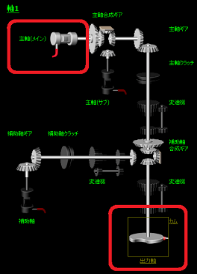
（補足）

カムデータの作成は目標位置の指定であり，同時に速度指令値の指定でもある（時間軸がある為）

同期制御の時は，出力軸（スレーブ）側で速度設定をする必要はない

### パラメーターの設定

**同期する側（スレーブ側）の軸で設定を行う**。

シンプルモーションユニット設定機能 → 同期制御パラメータ

同期パラメータ → 軸n同期パラメータ

基本的には以下の項目を設定する：

入力軸：

Pr.463カム基準位置設定方法

出力軸：

Pr.400種別／軸番号（上で指定した入力軸を選択。）

Pr.439カム軸１サイクル長　　　Pr.441 カムストローク量

Pr.440 カムNo.　　　Pr.460 １サイクル現在値設定方法

注意点

設定画面はテーブル式になっているが，範囲が狭く分かり難い。

軸１の設定をしているつもりが軸２の設定だったりするので注意する。

### 位置決め（運転）

バッファを用いる場合はCd.380 を用いる。.b0が軸１，.b1が軸２用となっている。

FBを 使う場合はStarting/ending sychronousを使う。

いずれも**同期側の軸番号（出力軸）**を指定する。（入力軸ではない。）

また，同期ビットは立ち上がりの信号が要るらしく，主軸（入力軸）の運転が終わった後に同期ビットのリセットが必要。

### 同期エンコーダ軸について

80SSC（８軸）でも同期エンコーダー軸は４軸まで。

### パラメータ（同期：Pr）

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.300  G32800 +10n | サーボ入力軸種別  （関連）  Pr.1 単位設定  Pr.83 桁数 | 0: 送り量現在値 (Pr.1 の単位)  1: 実現在値 (Pr.1 の単位)  2: サーボ指令値　[pulse]  3: フィードバック値　[pulse] |
| Pr.400  G36400 +200n | メイン入力軸番号  種別の入力軸番号と，その扱いを決める。 | 0 無効  1～8サーボ入力軸  　システム設定で指定した入力軸から番号で選択して主軸にする。  2指令生成軸 [201～208]  　システム設定で指定した指令生成軸から番号で選択して主軸にする。  3同期エンコーダー軸 [801～804] |
| Pr.438 | カム軸サイクル単位 |  |
| Pr.439 (32bit)  G36472+200n | カムサイクル長：一回転するのに必要な長さ（≒円周） | 入力軸がどれだけ動いたか，の値を入力する必要がある？19.09.20 |
| Pr.440  G36474+200n | カムNo. | 0:直線カム；1～256:作成したカムを使用  カム軸1サイクル現在値が０の時に取り込まれる |
| Pr.441 (32bit)  G36476+200n | カムストローク量  １ストローク（１モーション）でワークが動く長さ | ストローク比100%の時の具体的長さ  つまり１ストロークでモーターはPr.441/Pr.3回転する事になる。　　関連 Md.411  同期中はカムサイクル位置の零度位置で取り込み |
| Pr.460  G36500+200n | 主軸ギア後1サイクル現在値設定方法 | 0: 前回値  1: 主軸ギア後  2: 入力軸から計算 |
| Pr.463  G36503+200n | カム基準位置設定方法  *Pr.462の値が0か2の場合*  (どこの点で同期を開始するか，だと思われる 19.10.14) | 0: 前回値  1: カム基準位置初期設定値  2: 送り現在値 |
|  |  |  |
| Pr.464  G36504+200n | カム軸1サイクル現在値設定方法 |  |
| Pr.465  G36506+200n | 主軸ギア後1サイクル現在値初期設定値 |  |

### パラメータ（同期：それ以外）

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Cd.380  G36320 | 同期制御始動 | .b0 軸１用  .b1 軸2用　（以下同様） |
| Cd.400  G44080+20n | 主軸クラッチ指令 | 補助軸はCd.403 |
| Md.26  G60909+120n | 軸動作状態  19.11.07　読みだせないが？ | 0: 待機中 　　　　1:停止中  5: 解析中　　　　8:位置制御中  15: 同期制御中 |
| Md.300 (32bit)  G33120+10n | サーボ入力軸現在値 |  |
| Md.301 (32bit)  G33122+10n | サーボ入力軸速度 |  |
| Md.400 G42800+40n | 主軸合成ギア後現在値 |  |
| Md.407 (32bit)  G42812+40n | カム軸サイクル現在値 |  |
| Md.408 (32bit)  G42814+40n | カム基準位置 |  |
| Md.409  G42816+40n | カム軸送り現在値 |  |
| Md.420  G42828+40n | 主軸クラッチON/OFF | ※よほど特別な理由がないとクラッチを切らないと思う。 |

## 同期（仮想軸）

### 指令生成軸

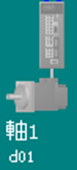
サーボアンプが接続された軸とは独立して制御する？？？

同期の際の入力軸専用の仮想軸。（らしい）

バッファメモリの割り当てが無く，Cd.300（G61970+128n）～Cd.306（G61978+128n）を利用してパラメータを指定していく。Works3を使った設定もできる。

⇒ PLCで動的にパラメータを変えるのは非常にめんどくさく，現実的ではない。

### 仮想サーボアンプ

上の指令生成軸と違って，あくまでサーボとして振る舞う。

シンプルモーションユニット設定機能　のナビゲーションウィンドウ→

システム構成 → 任意の軸のアイコンをダブルクリック。

仮想サーボとして使用する　のチェックを入れる。

あとは通常のサーボと同じ様に設定を行う。

### パラメーター（仮想軸・指令生成軸）

※Pr.460-468 は同期制御開始時に取り込まれる。

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.340  バッファ該当なし | 指令生成軸有効設定 | 電源On時に取り込み |
| Pr.346  バッファ該当なし | 指令生成軸１サイクル長 | 電源On時に取り込み |
|  |  |  |
| Cd.300  G61970+128n | パラメータ番号指定 | 範囲：1~400；（例）=340 とするとPr.340 |
| Cd.301(32bit)  G61972+128n | パラメータ設定値 |  |
| Cd.302  G61971+128n | パラメータ制御要求 | 0000H: 要求なし(制御完了) 　FFFFH: 書込みエラー  0001H: 書込要求　　　0002H: 読出要求 |
| Cd.303  G61974+128n | 位置決めデータ番号指定 | 範囲：1~100 |
| Cd.304  G61975+128n | 位置決めデータ指定 |  |
| Cd.305(32bit)  G61976+128n | 位置決めデータ設定値 |  |
| Cd.306  G61978+128n | 位置決めデータ制御要求 | 0000H: 要求なし(制御完了) 　FFFFH: 書込みエラー  0001H: 書込要求　　　0002H: 読出要求 |
| Md.347(32bit)  61002+120n | 指令生成軸１サイクル現在値 |  |

## 同期（カム位置復元）

### 復元方法

Cd.380同期制御開始は，同時にカム位置復元のトリガでもある。

・カム軸１サイクル現在値復元 (Pr.462=0)

[Md.407]カム軸1サイクル現在値 = [Md.409]カム軸送り現在値 – [Md.408]カム基準位置

カム側の動作角度である，１サイクル現在値を代入し直す。

Md.408カム基準位置　はPr.463で設定方法をを選択

0: 前回値 ：前回基準位置として取り込んだ値を代入する。無い場合は送り現在値を代入。

1: カム基準位置初期設定値 ：Pr.467の値を代入する。（常にこれでいいような気がするが）

2: 送り現在値 ：当然，１サイクルの零点とする位置に居る必要がある。（あまり使わない気が）

Md.409カム軸送り現在値　は現在値取り込み。

解が２つ以上ある場合（往復パターンなど）は Pr.468 カム軸1サイクル現在値初期設定値で検索開始位置を指定（最初に一致した値を取り込む。）

・カム基準位置復元 (Pr.462=1)

[Md.408]カム基準位置 = [Md.409]カム軸送り現在値 – [Md.407]カム軸1サイクル現在値

カム基準位置を代入し直す。Md.409カム軸送り現在値は現在値取り込み。

Md.407カム軸１サイクル現在値　はPr.464での取り込み方法を選択

Pr.464=0 前回値　前回の同期制御**開始**時の現在値？前回の同期制御**終了**時の現在値？

Pr.464=1: カム軸1サイクル現在値初期設定値(Pr.468) の値を利用する。

Pr 462=0 　　：Pr.468の値の位置からサーチを開始する。

Pr 462=1or2：Pr.468にカム軸１サイクル現在値の初期設定値を代入する。

※Pr.468の「初期設定値」という表現が分かり難い。今，現在値まで運転したカムの１サイクル零点はどこ？という話。

Pr.464=2: 主軸ギア後1サイクル現在値　　；Pr.464=3: 補助軸ギア後1サイクル現在値

・カム軸送り現在値復元 (Pr.462=2)

[Md.409]カム軸送り現在値= [Md.407]カム軸1サイクル現在値＋[Md.408]カム基準位置

今の具体的な位置アドレスを示す，カム軸送り現在値を代入し直す。

Md.407１サイクル現在値 は上の例 （Pr.462=1 ）と同じ設定方法。

Md.408基準位置 は 上の例 （Pr.462=0 ） と同じ様に設定する。

### 復元位置計算方法

#### カム位置計算機能（アドバンスド同期制御 5.5カム位置計算機能）

この機能は位置決めパラメータにどのような値が入っていても使用できる。

以下のパラメータを指定：

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Cd.612  G53780 | カム位置計算要求 | 1: カム軸送り現在値計算要求　　　　　⇒ カム軸基準位置復元を行う  2: カム軸1サイクル現在値計算要求　⇒ 送り現在値復元を行う |
| Cd.613  G53781 | カムNo. | 0～256（関連）Md.410実行カムNo. |
| Cd.614 (32bit)  G53782 | カムストローク量 | Pr.441 に相当。（関連）Md.411実行カムストローク量 |
| Cd.615 (32bit)  G53784 | カム軸  1サイクル長 | Pr.439 カムサイクル長 に相当。 |
| Cd.616 (32bit)  G53786 | カム基準位置 | （関連）Pr.467，Md.408 |
| Cd.617 (32bit)  G53788 | カム軸  1サイクル現在値 | （関連）Md.407 |
| Cd.618 (32bit)  G53790 | カム軸送り現在値 | （関連）Md.409 |

結果は「Md.600 カム位置計算結果」に格納される。

Cd.612に代入した値によって復元方法が異なる。（関連）Pr.462

Cd.612=1: カム軸送り現在値計算要求　　　　　⇒ カム軸基準位置復元を行う

Cd.612=2: カム軸1サイクル現在値計算要求　⇒ 送り現在値復元を行う

#### 同期制御解析モードを使った方法（アドバンスド同期制御 5.6.）

概要：

Cd.380同期制御始動をOnする時にCd.381同期制御解析モードがOnになっていると，同期解析モードが始動する。解析が終わると自動的にCd.380がOffし，同時に同期制御モニタデータ群（Md.400～425）が更新される。 特にMd.409が同期復元に必要。

その後Cd.381 をOffすると運転（同期）可能になる。

（補足）

・同期解析モードではMd.141 BUSY信号はONしない。

・Cd.381に対象軸をビットフラグで指定する。Cd.381.b0 が軸１，Cd.381.b1が軸２となる。

（例）２軸と３軸：=0x011　[dec. =3]　⇒　Cd.381にH3 を代入する（K3でも良い）

ここで対象軸となるのは同期軸のみ。；結局，ビット扱いしないと上手くいかないかも

・Cd.380同期制御始動　の対象軸もビット指定。Cd.381のOnと同時かそれ以降にする。

なお，このビットは解析が終わると自動的にOffする

・結果はそれぞれの対象軸のMd.409に格納される。（G42816+40n）

実施結果：(19.11.19)

同期解析モードでの同期復元は以下の様な流れになる：（観察した結果）

・上の方法でMd.409を算出し，スレーブ軸をそのアドレスに移動させる。

・Cd.380同期制御始動をOnして運転を再開する。

・再開後，最初に動くのはマスター軸のみ。スレーブ軸は動かない。

・マスター軸が特定のアドレスに来た時点で，スレーブ軸も動作を始める。

⇒ つまり Md.409で算出されるのは「同期再開位置」のような値らしい。

⇒ 仮に上のような流れだったとすると，「同期再開可能位置の範囲から外れています」みたいなエラーは何だったのか

19.11.19メモ

今まで同期解析モードによる同期は一度も成功しなかったが，今日になって突然成功した。（ver04\_11）

但し，失敗する時は失敗が続く。何か必要なパラメータを見落としているかもしれない。

### パラメーター（復元）

※Pr.460-468 は同期制御開始時に取り込まれる。

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.462  G36502+200n | カム軸位置復元対象  0と2の違いは？回転系の場合は同じ  リニア軸的に使うと選択肢は2番に。 | **0: カム軸 1サイクル現在値復元**（関連） Pr.468  1: カム基準位置復元  2: カム軸 送り現在値復元  （詳細） [復元方法](#_復元方法) |
| Pr.463  G36503+200n | カム基準位置(Md.408) 設定方法  *Pr.462の値が0か2の場合*  (どこの点で同期を開始するか，だと思われる 19.10.14) | 0: 前回値  1: カム基準位置初期設定値→ Pr.467の値  **2: 送り現在値** |
| Pr.464  G36504+200n | カム軸1サイクル現在値(Md.407)  設定方法  *Pr.462の値が1か2の場合* | **0: 前回値**  1: カム軸1サイクル現在値初期設定値→Pr.468の値  2: 主軸ギア後1サイクル現在値  3: 補助軸ギア後1サイクル現在値 |
| Pr.467 (32bit)  G36510+200n | カム基準位置初期設定値  *Pr.463の値が1の場合* | def. =0  関連 Pr.463 |
| Pr.468 (32bit)  G36512+200n | カム軸1サイクル現在値初期設定値 | 関連 Pr.462  Pr.462=0：サーチを開始する位置  Pr.462=1,2かつPr.464=1：初期値  ※どちらの場合も，大体=0 になると思う。 |
| Cd.380  G36320 | 同期制御始動  メモ　何故か16bit値をmovしても動かなかった。完全にbitとして扱う必要がある？ | 16bitで対象軸を指定  .b0 軸１～.b7 軸８  ※これがあるのでFBのSyncは全く必要ないと断言できる。  むしろFBの方は多軸に指令する事ができず不利。 |
| Cd.381  G36322 | 同期制御解析モード | 16bitで対象軸を指定  .b0 軸１～.b7 軸８ |
| Cd.407  G44087+20n | 同期制御変更要求  同期制御中にカム基準位置などを変更できる。  （関連）Cd.406 [G44086+20n ] | 0 カム基準位置移動  1 カム軸1サイクル現在値変更  2 主軸ギア後1サイクル現在値変更  3 補助軸ギア後1サイクル現在値変更  4 カム軸1サイクル現在値移動 |
| Cd.612  G53780 | カム位置計算要求  （Cd.612-618 への値の代入が必要。） | 1. カム軸送り現在値計算要求  2. カム軸1サイクル現在値計算要求  （関連）Md.600 |
| Md.600 (32bit)  G53800 | カム位置計算結果 | Cd.612=0 で計算終了を判断するらしい |

## パラメータ

パラメータとその意味を主要な物だけ抜粋：

背景色が灰色のものは，関連はあるがそれほど重要でない，又は初期値を使うもの。

背景色が橙色のものは，重要パラメータ

注意点

・Gはバッファメモリアドレス。また，n = 軸番号－１；つまり**0から始まる。注意！**

・Pr.1～Pr.90，Pr.95，Pr.116～Pr.119，Pr.127はCd.190シーケンサレディ立ち上がり時に適用される。

また，Cd.190のリセットには数スキャン必要な様子。そうしないと，Cd.190に伴った準備完了やサーボOnがうまくリセットされず，制御ができなくなる。

### パラメータ(Pr.)

|  |  |  |
| --- | --- | --- |
| 番号 | 項目名 | 備考 |
| Pr.9　(32bit)  G12+150n | 加速時間(0)  **※ツールでの設定では 0:1000 などと表示される。これは「０番：値1000の意」　比率ではない** | 指令速度に達するまでの時間  （FX3UのDRVIなどは最高速度に達するまでの時間なので，かなりの違いがある。注意）  Pr.25-27 加速時間1,2,3 (36,38,40+150n)  （関連）Da.3 加速時間番号 |
| Pr.10　(32bit)  G14+150n | 減速時間(0) | Pr.28-30 減速時間1,2,3 (42,44,46+150n)  （関連）Da.4 減速時間番号 |
| Pr.21  G30+150n | 速度制御時の送り現在値 | **※初期値は送り「現在値の更新を行わない」**になっているので注意！！ |
| Pr.22 | 入力信号論理選択 | **1負論理(B接)** 2 正論理(A接) |
| Pr.81  G34+150n | 速度・位置機能選択 | ABSモードはPr.単位設定が2:degree の時のみ有効 |
| Pr.97 | SSCNET 設定 | G106 |
| Pr.116 | FLS信号選択: 入力種別 | **１. サーボアンプ**　２.バッファメモリ  2 を選択した時は Cd.44(G592)で値を設定する。 |
| Pr.117 | RLS信号選択: 入力種別 |
| Pr.118 | DOG信号選択: 入力種別 |
| Pr.119 | STOP信号選択: 入力種別 |
| Pr.405  36408+200n | 主軸クラッチ制御設定  スレーブ側で設定を行う  関連 Md.400 Cd.400 | 160 ON制御モード  4: アドレスモード  指定アドレスでOn/Off を行う  関連 Pr409, 410  161 OFF制御モード  162 高速入力要求 |
| Pr.411 | 主軸クラッチスムージング方式 |  |
| Pr.444 (32bit)  G36482+200n | カム軸位相補正進め時間  （単位 µs　負の値も可能） | 現在のカム位置から敢えて進めたり遅らせたりする場合に用いる |
| Pr.445 | カム軸位相補正時定数 | G36484+200n |
|  |  |  |

### 制御データ(Cd.)

|  |  |  |
| --- | --- | --- |
| Cd.6  G61863+128n | 再始動指令 | 途中で停止された位置決めの再開。 |
| Cd.10 | 加速時間変更値 | G4308+100n |
| Cd.11 | 減速時間変更値 | G4310+100n |
| Cd.12 | 加減速時間変更値許可/不許可 | G4312+100n |
| Cd.14 | 速度変更値 | G4314+100n (32bit) |
| Cd.15 | 速度変更要求 | G4316+100n |
| Cd.38  G4348+100n | ティーチングデータ | 0: [Da.6]位置決めアドレスに書き込む。  1: [Da.7]円弧アドレスに書き込む |
| Cd.39 | ティーチング位置決めNo | 4349+100n |
| Cd.44 | 外部入力信号操作  G5928 1軸分4bit　　×8軸 = 32bit分  G5928 1-4軸 ； G5929 5-8軸 | .b0 軸１上限リミット信号(FLS)  .b1 軸１下限リミット信号(RLS)  .b0 軸１近点ドグ信号(DOG)  .b1 軸１停止信号(STOP)  関連Pr.119 STOP信号選択 |
| Cd.100 | サーボOFF指令 | G4351+100n |
| Cd.108  G4359+100n | ゲイン切換え指令フラグ | 関連 Md.108.b4 |
| CD.130  G4354+100n | サーボパラメータ書き込み要求  （完了すると自動的に0が格納） | 対象となるのは軸制御データ  1. 16bit書き込み　　　2. 32bit書き込み  関連 CD.131, 132 ※但しあまり使わないと思う。 |
| Cd.140(32bit)  G4376+100n | 速度制御モード時指令速度 |  |
| Cd.180  G30100+10n | 軸停止 |  |
| Cd.184  G30104+10n | 位置決め始動 | G30104が軸１用 |
| Cd.190  G5950 | シーケンサレディ | CPUユニットが正常であることをシンプルモーションユニットに知らせる信号。  プログラムで能動的にONさせる。  また，ONすると以下の値が転送される：  オートチューニングモード(PA08)，オートチューニング応答性(PA09)，フィードフォワードゲイン(PB04)，負荷慣性モーメント比(PB06)，モデル制御ゲイン(PB07)，位置制御ゲイン(PB08)，速度制御ゲイン(PB09)，速度積分補償(PB10)，速度微分補償(PB11) |
| Cd.191  G5951 | 全軸サーボON |  |
| Cd.401 | 主軸クラッチ無効指令 | G44081+20n　どう違う？ |
| Cd.402 | 主軸クラッチ強制無効指令 | G44082+20n　即座にOFFらしい |

### モニタデータ(Md.)

|  |  |  |
| --- | --- | --- |
| Md.20 (32bit)  G2400+100n | 送り量現在値 | 移動中は負の値になる？ |
| Md.21 (32bit)  G2402+100n | 送り機械値 | Md.20はサーボ一回転でリセットされる実現在値なのに対し，こちらは累計値 19.09.25 |
| Md.22 (32bit)  G2404+100n | 送り速度 |  |
| Md.23  G2406+100n | 軸エラー番号 |  |
| Md.24  G2407+100n | 軸ワーニング番号 |  |
| Md.26  G60909+120n | 軸動作状態  19.11.07　読みだせないが？ | 0: 待機中 　　　　1:停止中  5: 解析中　　　　8:位置制御中  15: 同期制御中 |
| Md.27 (32bit)  G2410+100n | カレント速度 |  |
| Md.28 (32bit)  G2412+100n | 軸送り速度 | 単軸の場合 =Md.22になる |
| Md.30  G2416+100n | 外部信号 | b0下限リミット信号　b1上限リミット信号  b3停止信号　　　　b4外部指令信号／切換え信号  b6近点ドグ信号 |
| Md.31  G2417+100n | ステータス | b0速度制御中フラグ  b1 速度・位置切換えラッチフラグ  b2 指令インポジションフラグ  b3 原点復帰要求フラグ  b4 原点復帰完了フラグ  b5位置・速度切換えラッチフラグ  b9軸ワーニング検出 b10速度変更0フラグ  b12 MコードON b13エラー検出  b14始動完了 b15位置決め完了 |
| Md.32 | 目標値 | 関連Da.6 |
| Md.33 | 目標速度 | 関連Da.8 |
| Md.44 | 実行中位置決めデータNo. | 位置決めデータNo. |
| Md.47 | 実行中位置決めデータ | Md.44 の詳細 |
| Md.50  G4231 | 緊急停止信号 | シンプルモーションユニットのEMI信号  (ユニット前面D端子の10pin) |
| Md.101 (32bit)  G2450+100n | 実現在値 |  |
| Md.103 (32bit)  G2454+100n | モーター回転数 | これはrpmを10-2単位で表示 |
| Md.108  G2477+100n | サーボステータス | b0: レディ ON 　　 b1: サーボON  b2，b3: 制御モード b4: ゲイン切り替え  b7: アラーム中  b12: インポジション b13: トルク制限中  b14: 絶対位置消失中 b15: 警告中 |
| Md.114  G2488＋100n | サーボアラーム |  |
| Md.119  G2476＋100n | サーボステータス２ | b0零点通過　　b3零速度中  b4速度制限中　b8 PID制御中  なお，零点を示すビットは無いらしい 19.09.25 |

|  |  |  |
| --- | --- | --- |
| Md.107 | パラメータエラー番号 | 常に0 |
| Md.114  G2488+100n |  | サーボアラーム |
| Md.140  G31500 | ユニットステータス | b0 準備完了信号  Cd.190シーケンサレディ信号ONで異常が無いとOnする  b1 同期用フラグ  電源を入れて，シンプルモーションユニットへがアクセス可能状態になると自動的にON。これがONしないと何もできない。 |
| Md.141  G31501 | Busy | G31501.b0が軸１用  G31501.b1が軸２用（以下同様） |
| Md.502 | ドライバ運転アラーム番号 | ？一体何の事？  IAIのアクチュエーターや5相ステッピングのエラーコードが格納される。  *そもそも制御できるの？ 20.07.02* |
|  |  |  |

### (PA.)

|  |  |  |
| --- | --- | --- |
| PA03  G28403+100n | 絶対位置検出システム | val: 0:無効 1:有効 |
|  |  |  |

### (PB.)

|  |  |  |
| --- | --- | --- |
| PB26  G28444+100n | ゲイン切換え機能 |  |
|  |  |  |

## 指定軸のパラメータ

どうやら，共通の値を用いている感じがするのでまとめておく。[19.08.30]

|  |  |  |
| --- | --- | --- |
|  |  | 値の範囲 |
| 一桁 | 単純に軸番号 | [1～8] |
| 100番台 | 高速入力要求（外部指令信号） | [101～108] |
| 200番台 | 指令生成軸 | [201～208] |
| 800番台 | 同期エンコーダー軸 | [801～804] |
| 7000番台 | ブロック始動指定 | [7000～7004] |
| 9000番台 | 特殊  9001: 機械原点復帰　　　9002: 高速原点復帰  9003: 現在値変更　　　　9004: 複数軸同時始動 | |

## 用語

指令生成軸：

指令生成のみを行う軸。サーボアンプが接続された軸とは独立して制御？？

入力軸を駆動する？？

(opp) サーボアンプ軸

### 各現在値の違い

・送り現在値Md.20

機械原点を基準とした指令値。現在値変更でアドレスの変更が可能。

・送り機械現在値Md.21

常に機械原点を基準とし，現在値変更でもアドレスを変更できない。連続回転の場合値は累計されていく。（あまり使わないらしい）

・実現在値 Md.101

指令の結果のフィードバックを受けた現在値。その為，送り現在値（指令値）に対して若干遅れる。

・サーボ入力軸現在値 Md.300 (G33120+10n [32bit])

サーボ入力軸に指定したサーボの現在値。入力軸の送り現在値か実現在値を指定する事が多い。

・主軸合成ギア後1サイクル現在値　Md.400

主軸のメイン入力とサブ入力を合成した後の現在値が格納される。累計値となる。

※１サイクル現在値　＝　カム軸１サイクル長（Pr.439）を上限とし，１サイクル長に対した現在値。

・主軸ギア後1サイクル現在値　Md.401

主軸ギア後の1サイクル現在値が格納される。＝入力軸の送り現在値？

主軸ギア後には減速比の指定が可能であり，その減速比の影響を受けた値。

・カム軸１サイクル現在値　Md.407

カム軸へ入力された移動量から計算された1サイクル現在値が格納される。(位相補正後の値)

カム軸１サイクル長（Pr.439）を上限とし，サイクル長の現在値。

・カム基準位置　Md.408

カム動作の基準（零点）になる位置，角度。

・カム軸送り現在値　Md.409

カム軸制御中の送り現在値が格納される。カムデータとサイクル現在値を基に算出された，実際の指令値。

＝｛（１サイクル現在値／１サイクル長さ）×Pr.441カムストローク量｝＋カム基準位置　と言ったイメージ。

## プログラミング（シンプルモーション）

### ファンクションブロック（詳細）

使いそうなものだけ抽出

|  |  |
| --- | --- |
| Function名 |  |
| StartPositioning | 位置決めの始動／停止を行う |
| JOG | JOG／インチング運転を行う  正逆転，速度の指定が可能。移動量=0 で無限移動 |
| MPG | 手動パルサ運転を行う |
| Sync | 同期制御の開始／終了 |
| EnableSyncEncoder | 同期エンコーダ軸からの入力を有効に |
| DisableSyncEncoder | 同期エンコーダ軸からの入力を無効。 |
| ResetSyncEncoderError | 同期エンコーダ軸のエラー情報の読出しと，エラーリセット |

#### ユニットラベル

一見まったく意味のない引数にも思えるが，この値によってファンクションブロックで設定できる項目が変わるらしい。

また，各パラメータ用のバッファデバイスに直結している「パラメータ」もある。これは最初から設定されているラベル。

#### ユニットラベルの一覧

シンプルモーションユニット設定のヘルプ → バッファメモリアドレス一覧

エクセルのファイルが立ち上がるので，バッファメモリエリア構成一覧　のシートを表示する。

## その他

### フラッシュROM

パラメータエリアa,b,c ，位置決めデータ，ブロック始動データのバックアップを記録する。日常的に書き込む必要は無いが，たまに書き込むようにする。

（参考　マニュアル（応用編）p489）

# 管理

## ファームウェアアップデート

・三菱電機FAサイトからファームウェアアップデート用の.zipファイルをダウンロード。.zipファイルを解凍して作成された$MELPRJ$ フォルダをSDカードに書き込む。

<https://www.mitsubishielectric.co.jp/fa/download/software/search.do?mode=software&kisyu=/plcf&lang=1&select=0&softid=1&viewstatus=&setviewpos=>

注意：シリアルナンバー(S/N)によってファイルを選択する必要がある。

（リンク先サイトでは「製造番号」となっているが，**シリアルナンバーの事**だと思われる。三菱の常として用語の統一がなされておらず，毎度多少の混乱を生んでいる）

・PLCをSTOP状態にして電源をオフ。の電源をONすると，RUN LEDとERR LEDが２秒間隔で点滅を繰り返す。(30秒間)（⇒ ERRとP.RUNが点滅しているが？19.08.26）

・PLCを再起動，又はリセットする。再び２秒間隔で点滅(20秒間)するので，点滅が終わるまで待機。再びPLCを再起動，又はリセット。

・もう一度２秒間隔の点灯（5秒間）。これで完了なので，CPU診断でバージョンを確認。