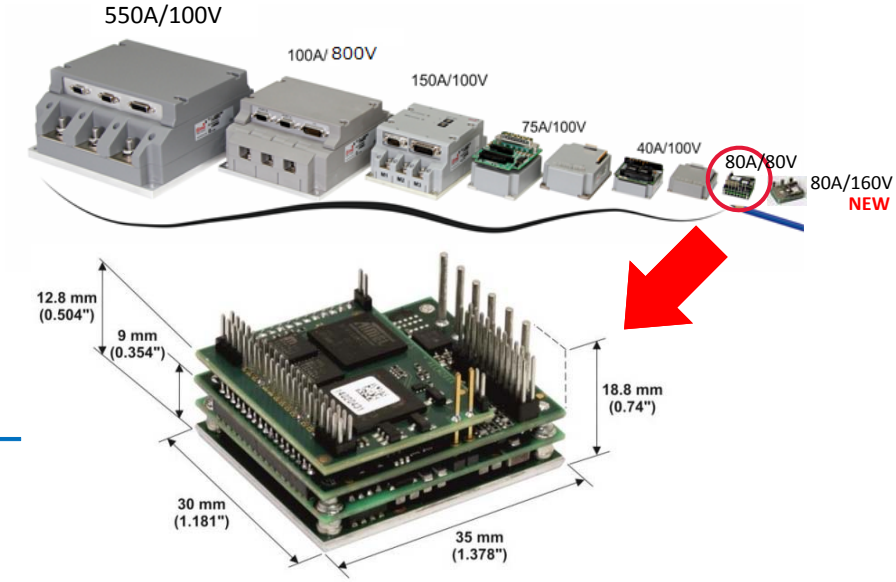
MAKING SMART MACHINES SMARTER

ELMO社ドライバ＆コントローラで
多軸ロボットの運動アルゴリズムを
簡単に設計

エルモ モーション コントロール

- 350万個以上のモータドライバが全世界で稼働中
- 月間8,000個～15,000個のドライバを供給中

EtherCAT



Gold Twitter
ゴールド ツイッター

The Any philosophy

いかなるアプリケーションでもご使用いただけるモータドライバ

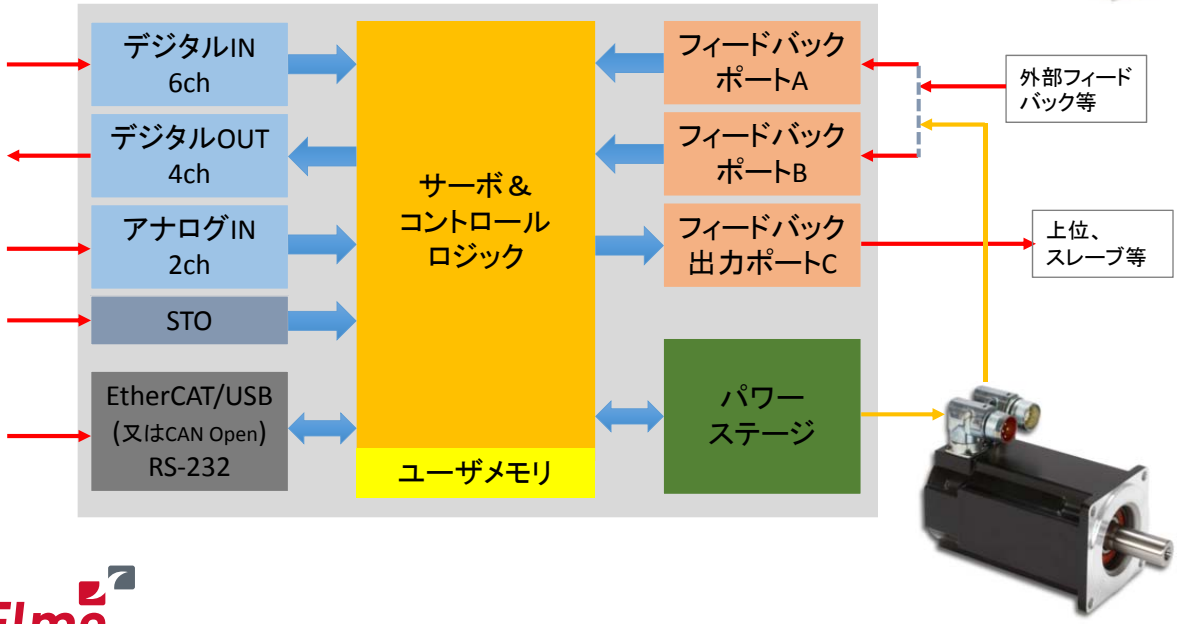
The Any philosophy

Gold Twitterシリーズの場合は

スペース	30mm × 35mm × 14.4mm (22.2g)
モーション	単軸、多軸同期(EtherCAT/CANOpen)
電圧	8-55VDC、10-95VDC、20-194VDC
パワー	80W～5,000W
電流	1A～80A連続 (型番で最適な値を選択)
フィードバック	市販のほとんどのフィードバックに対応
モータ	ACサーボ、DCブラシレス、DCブラシ、DDリニア

Elmo Motion Control

G-Twitterドライバ内部構成



Gold Twitter電圧・電流オプション

電源電圧		連続電流・出力								
60V	8～55VDC	30A	R50A							
		1440W	2200W							
80V	10～75VDC	R80A	R160A							
		5000W	10000W							
100V	10～95VDC	1A	3A	6A	10A	15A	25A	R45A	R50A	R140A
		80W	240W	480W	805W	1210W	2015W	3600W	4000W	11000W
200V	20～194VDC	3A	6A	10A	R15A					
		495W	990W	1650W	2400W					

ピーク電流＝Ic×2　　Rxxタイプは Ic=Ip

Gold Twitter

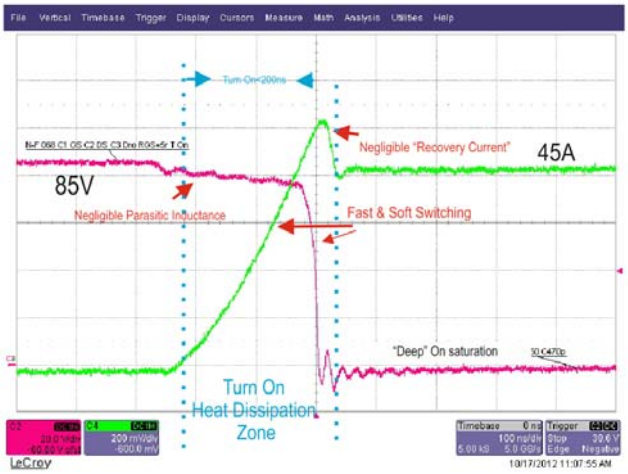
Double Gold Twitter

超 高電流、超小型、超高効率

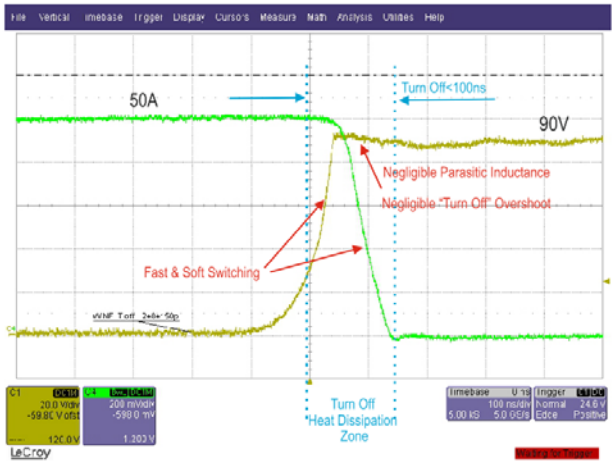
高速＆ソフトスイッチングテクノロジーが理想的PWMを実現

Elmo's Fast And Soft Switching Technology

Gold Twitter オン

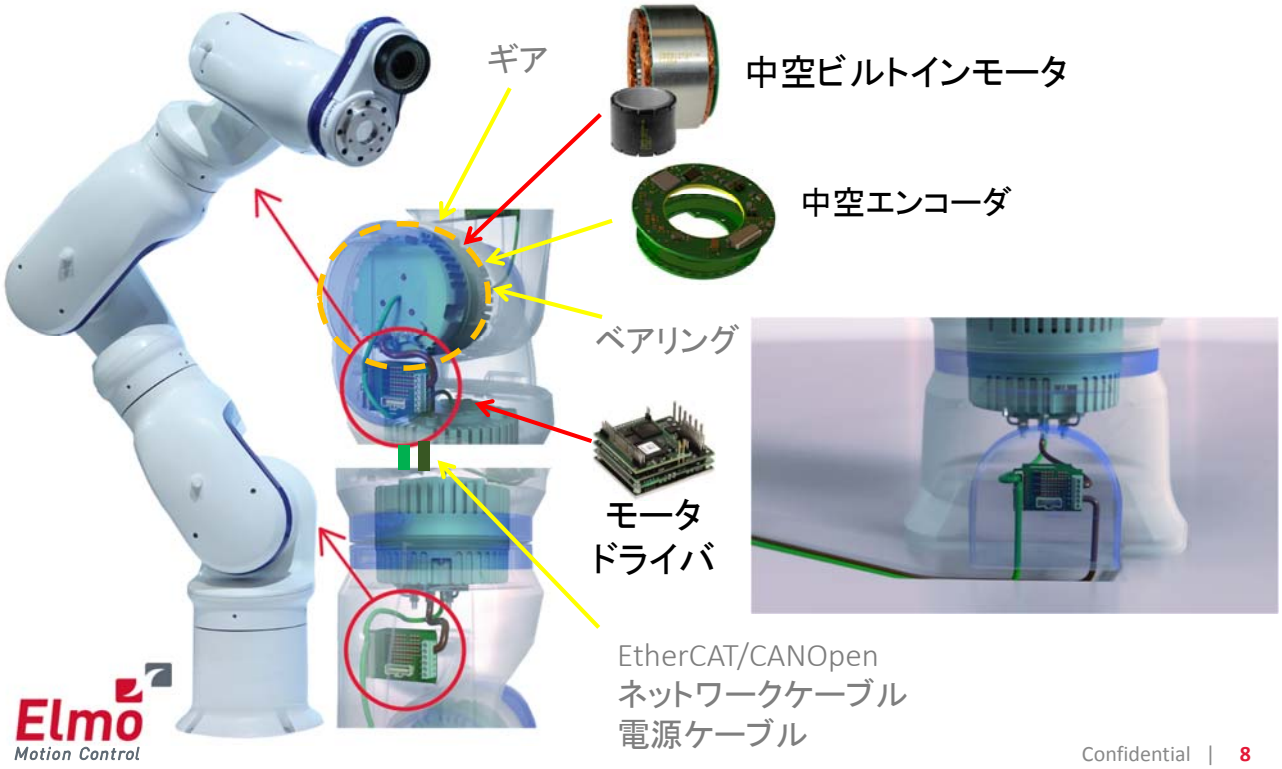


Gold Twitter オフ



Elmo Motion Control

ロボット各関節への組込み



Elmo Motion Control

開発に必要なすべてのソリューションを提供

- サーボドライバ
  - ハイパフォーマンス、インテリジェントドライバ
- EtherCATマスター/コントローラ　Maestro
  - ネットワークベースのモーションコントローラ
  - 最大100軸をコントロール
- 開発環境
  - EASII(Elmo Application Studio)
    マルチファンクション＆ユーザーフレンドリーな開発環境
  - MDS(Maestro Developer Studio)
    モーションコントローラ（Maestro）の開発環境

Elmo Motion Control

マエストロ　モーションコントローラ

- EtherCAT, CANOpenベースのコントローラ
- 最大100軸まで制御
- サイクルタイム 100us (8軸) * 最新世代プラチナシリーズ
- ネットワーク標準
  - DS-301, DS-402, DS-406, その他モーション規格
- プログラミング環境
  - IEC61131-3(PLCopen), .NET and Win32 C++(Host PC), GNU C/C++(Maestro上プログラム)
- 強力な統合開発環境 ”EASII”と“MDS“ が、効率の良いプログラム開発をサポート
- オンボードIO(オプション) : DIO 12in/8out, アナログIO 4in(差動)/4out

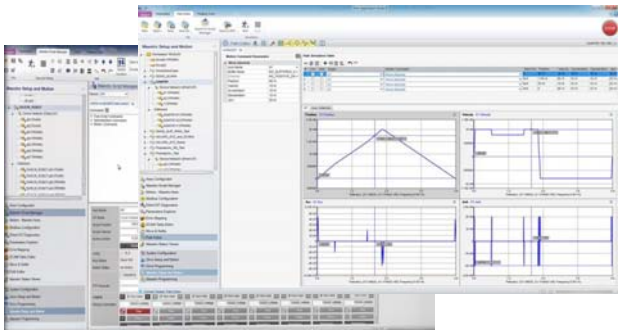
Elmo Motion Control

開発ツール
ELMO Application Studio (EASII)

EASII Elmo Application Studio

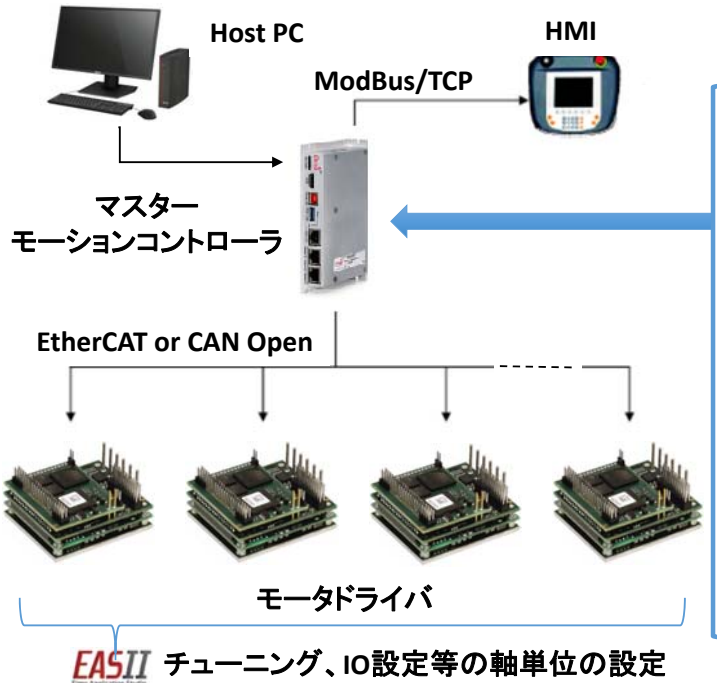
ELMOユーザに無償で提供される開発環境

- ネットワーク設定
- チューニング： オート、エキスパート(ボード線図、ニコラス線図)
- パラメータ設定
- IO設定
- モーションプログラミング：PLCopenに基づくプログラミング
- レコーダ/モニタ
- パスシミュレータ
- SIL(Software In the Loop)



Elmo Motion Control

ELMOコントローラによる制御



- **MDS (Maestro Developer Studio)**
  Maestroモーションコントローラ上のプログラム開発（C++）
  - SILを使用して開発
  - ELMO社APIを使用して開発
- **Visual Studio(VC++)**
  Maestroモーションコントローラを制御するHost PC上のプログラム開発
  - ELMO社APIを使用して開発
- **EASII (Elmo Application Studio)**
  PLCopen（IEC61131-3）を用いてプログラム開発 EASII

Elmo Motion Control

ロボット開発支援機能

- SIL(Software In the Loop)
- ELMO社 API

ロボット制御プログラム開発方法

SIL（Software In the Loop）で開発

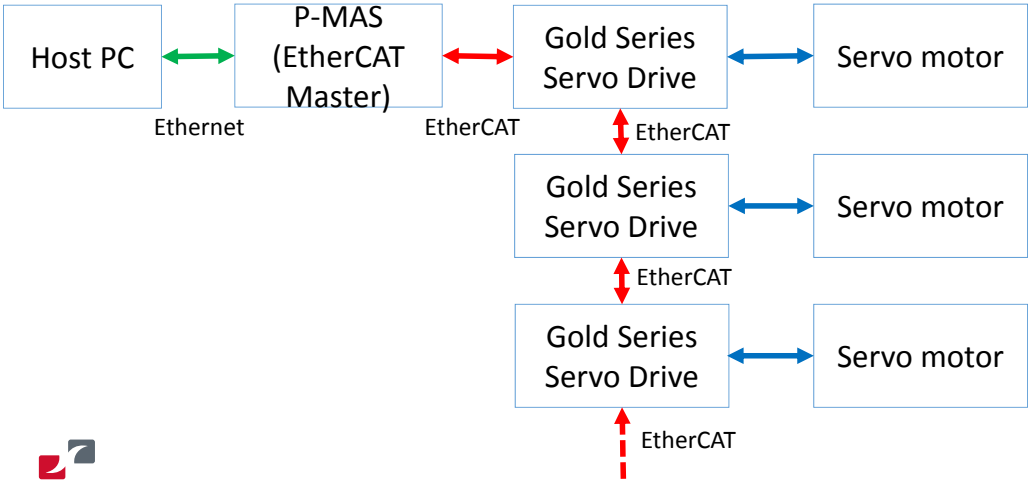
- 自社でアルゴリズムを持っている場合
  - 垂直多関節ロボット等

ELMO社APIで開発

- ELMOで用意されているAPI
  - DELTAロボット
  - SCARAロボット
  - 平面3リンクロボット
  - 直交座標ロボット

SILで開発/制御構成

- Host PC　1台
- モーションコントローラ(P-MAS:Platinum Maestro)　1台
- サーボドライバ(Gold シリーズ)　　軸数分
- サーボモータ　　軸数分



Elmo Motion Control

...(ドライバ接続個数はロボットに依存)

SILの機能

- P-MAS(Platinum Maestro)上で実行可能
- ユーザアルゴリズム(Kinematics)を組み込み可能
- EtherCAT サイクル毎に関数呼び出しする仕組みを用意
- リアルタイム処理が可能

Elmo Motion Control

MDSを用いたSILの開発手順

1. ユーザアルゴリズム(Kinematics)を準備
   - MATLAB/Simulinkを用いてアルゴリズムを設計後、C/C++のコードを生成
   - ユーザアルゴリズムからC/C++のコードを直接作成
2. 新しいCPPのプロジェクトを作成
3. ELMOが用意したテンプレートを選択
4. EtherCATサイクル毎に呼び出されるコールバックを定義し、この中にユーザアルゴリズムを実装
5. 定義したコールバック関数を指定してマルチメディアタイマーを設定

MDS: Maestro Developer Studio

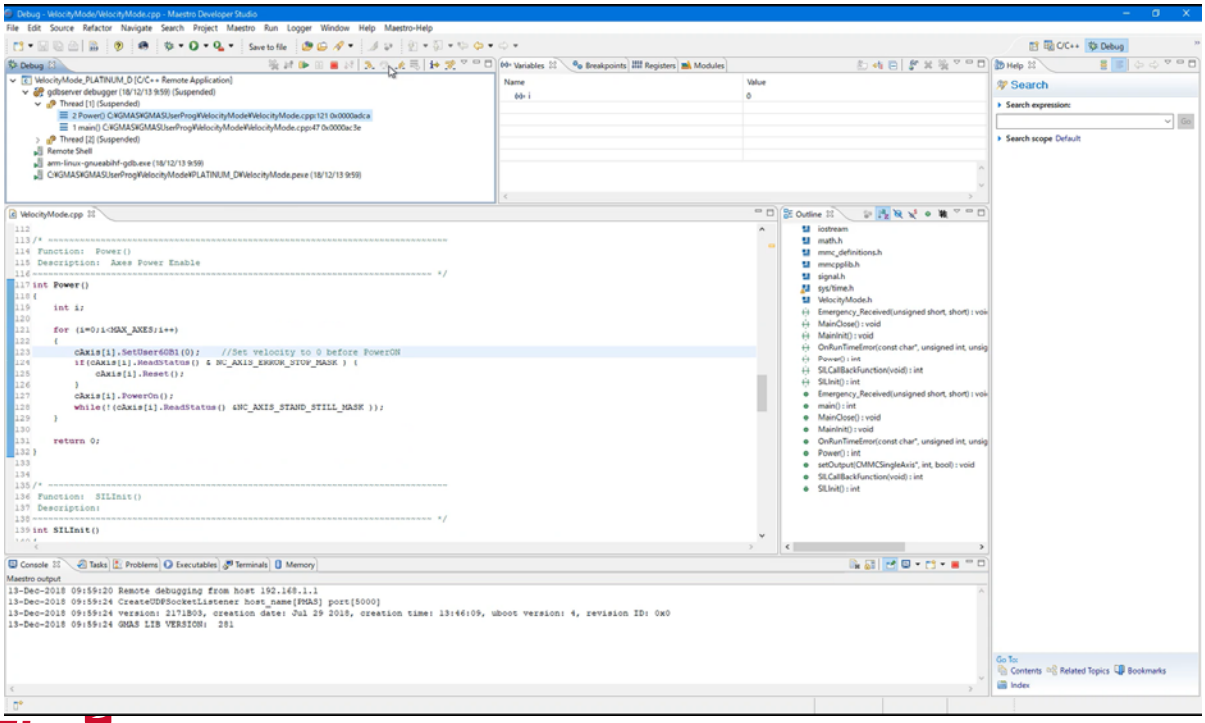
SILサンプルプログラム

```
int SILInit()
{
    int i;
    //タイマーを止める
    MMC_DestroySYNCTimer(gConnHndl);
    //初期位置や速度動作モードの設定などを行う
    cAxis[i].SetUser60B2(0);
    cAxis[i].SetOpMode(OPM402_CYCLIC_SYNC_VELOCITY_MODE);
    // EtherCATサイクルイベント処理ルーチンとEtherCATサイクル何回に一回実行するかを設定（第二引数）
    MMC_CreateSYNCTimer(gConnHndl ,SILCallBackFunction,1);
]
int SILCallBackFunction(void)
{
     //ここにユーザのアルゴリズムを入れる。
     //ユーザアルゴリズムに基づいて
     //位置、速度、トルクをアルゴリズムに従って適宜計算して送出する
     //位置の設定
     cAxis[i].SetUser607A(…);
     //速度の設定
     cAxis[i].SetUser60B1(…);
     //トルクの設定
     cAxis[i].SetUser60B2(…);
    return 0
}
```

SIL初期化処理

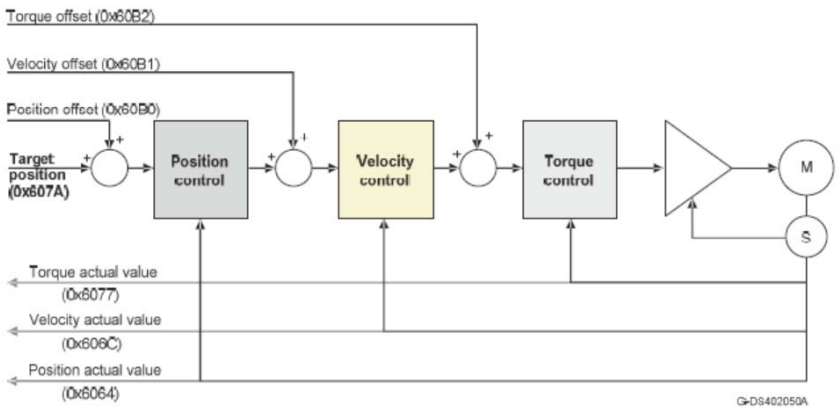
SILのEtherCATサイクルイベント処理ルーチン

MDSの開発画面



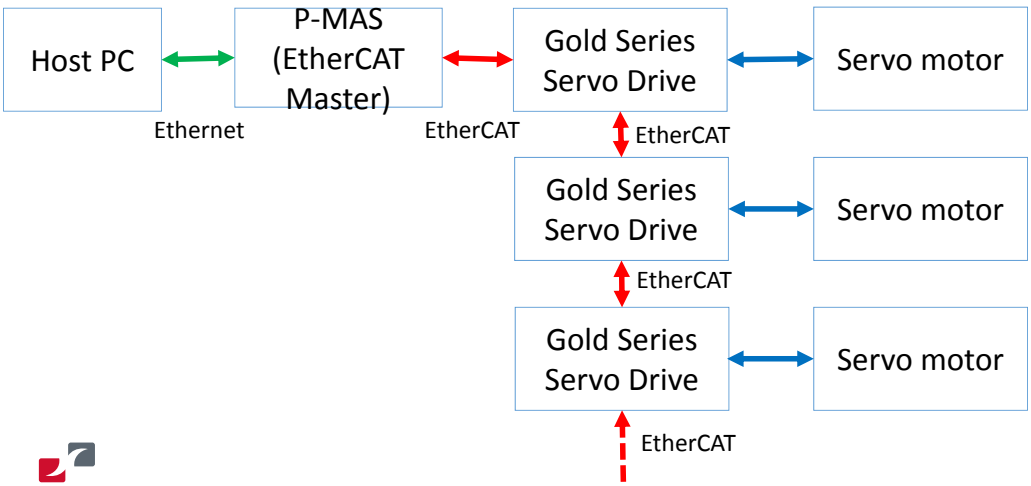
SILで対応している制御モード

- CSP,CSV,CSTを使用する
  - CSP: Cyclic Synchronous Position
  - CSV: Cyclic Synchronous Velocity
  - CST: Cyclic Synchronous Torque



ELMO社APIでの開発/制御構成

- Host PC　1台
- モーションコントローラ（P-MAS:Platinum Maestro）　1台
- サーボドライバ（Gold シリーズ）　　軸数分
- サーボモータ　　軸数分



…（ドライバ接続個数はロボットに依存）

APIを用いたロボットの開発手順

1. 開発するロボットの種類を以下から選択
   - DELTAロボット
   - SCARAロボット
   - 平面3リンクロボット
   - 直交座標ロボット
2. ロボット諸元の入力
3. 初期化ルーチンの出力
4. EASIIを用いて動きを確認
5. APIを使用してプログラムを実装

APIを用いたロボット開発に使うEASII機能

- Kinematic Editor
  - Robotの諸元を入力する
  - 入力した諸元に従って初期化コードを出力する
    - ✓ 機械座標系（MCS）と軸座標系（ACS）の変換テーブルの作成機能
- Group Motion
  - End Effectorを動かす機能
  - End Effectorの動きを記録する機能
  - End Effectorの動きを表示する機能
  - 実機とシミュレータ両方の環境でJogを操作して確認可能
  - 実機とシミュレータ両方の環境で End Effectorの座標の位置をスクリプトとして記録
  - 作成したプログラムをシミュレータ上で動作を確認

APIを用いたロボット開発環境

2種類の環境が用意されています

- Visual Studio(VC++)
  Motionコントローラを制御するHost PC上のプログラム開発
- MDS(Maestro Developer Studio)

1. MDSで新規にC++のプロジェクトを作成する
2. ELMOのプログラムのテンプレートを選択
3. APIを使用してRobotに関連するコードを追加していく
   - ロボットの諸元を初期化するコードを追加
   - ロボットの制御コードを追加

主要なAPI

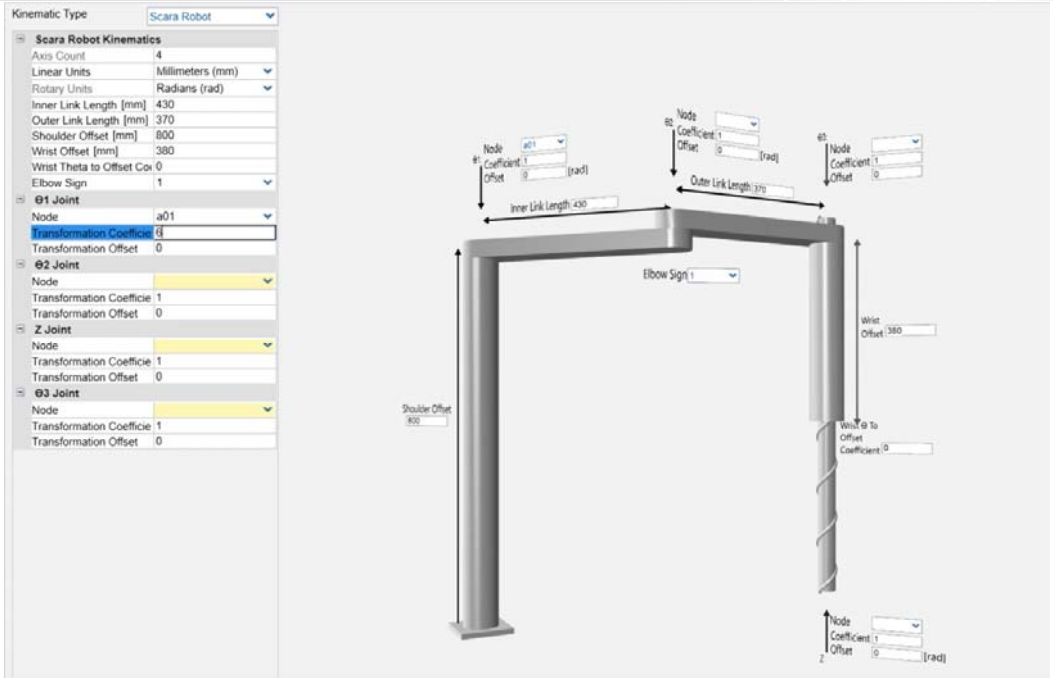
ロボット制御に必要なAPIが用意されています

- モーションコントローラMaestroに用意されているAPI（代表例）
  - 運動学や逆運動学に基づき座標変換を設定
  - 多軸で直線、円弧、スプラインなど軌跡を描く
  - 単軸で位置、速度、トルクを指定
  - Homing処理を行う
  - データレコーディング

Elmo Motion Control

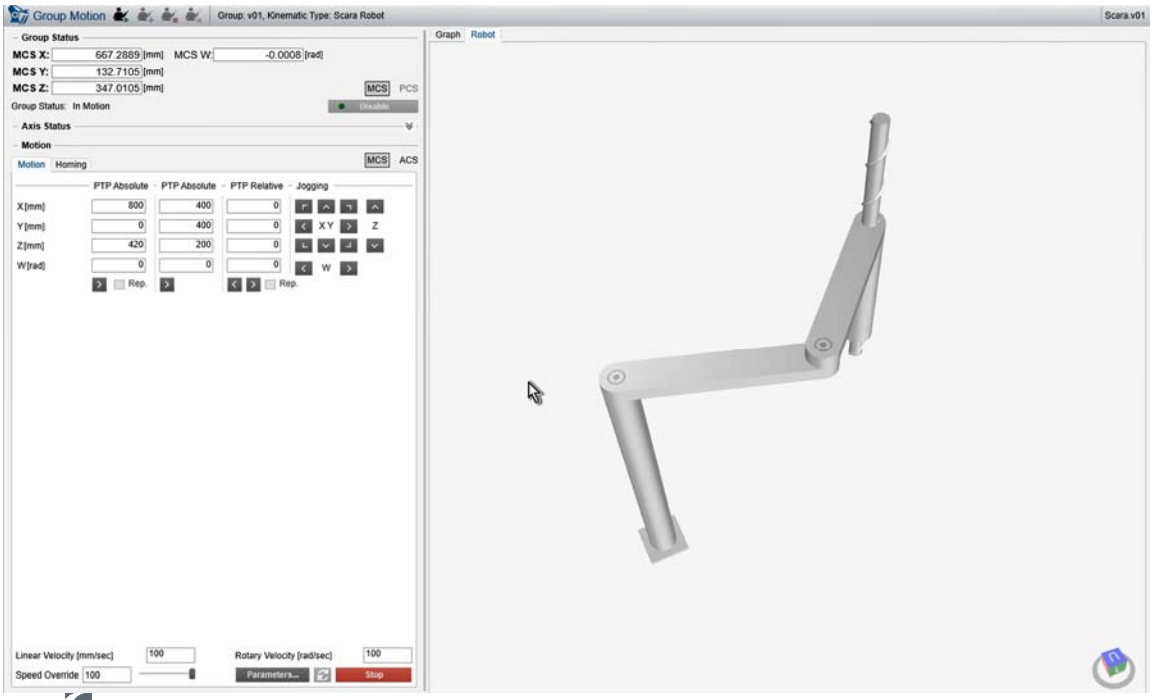
ロボットの諸元入力

EASII Elmo Application Studio



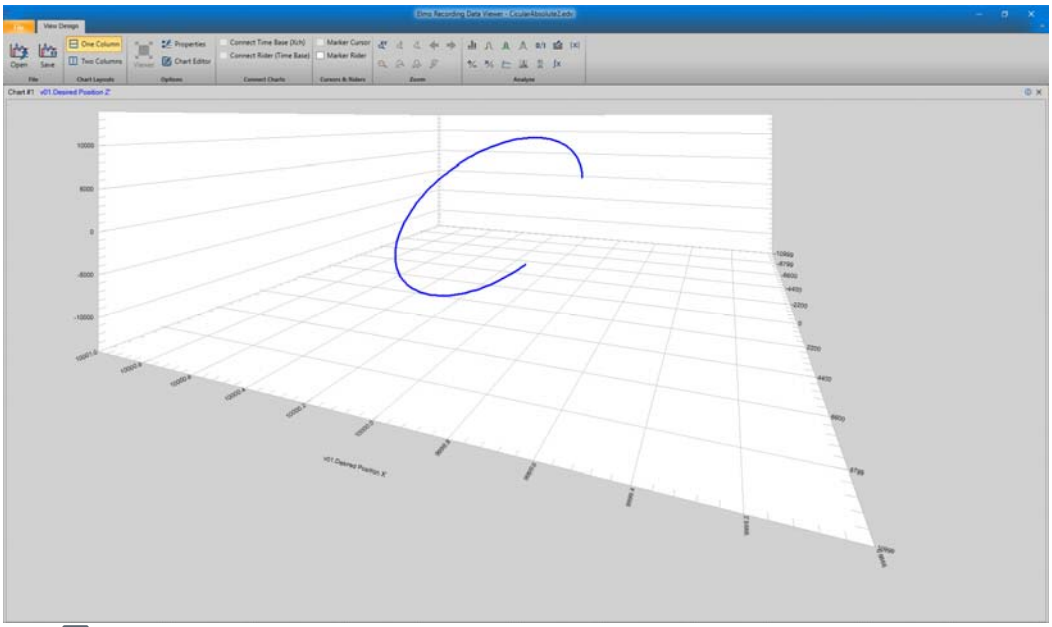
Elmo Motion Control

ロボットの動作確認

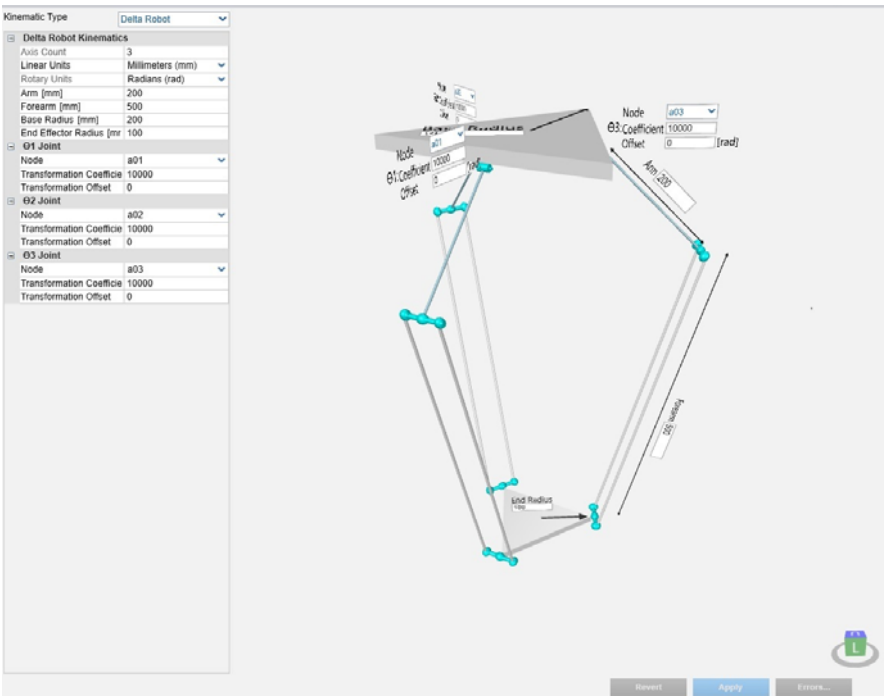


エンドエフェクタの軌跡の記録

エンドエフェクタの軌跡を記録して、要求する運動をしているか確認する



DELTAロボットの開発



お問い合わせ先
日本パルスモーターグループ
NPMハイテクノロジーズ株式会社
〒113-0033　東京都文京区本郷2-16-13　日本パルスモータービル内
TEL：03-3813-8847
Email：sales@npm-ht.co.jp
ホームページ：www.npm-ht.co.jp