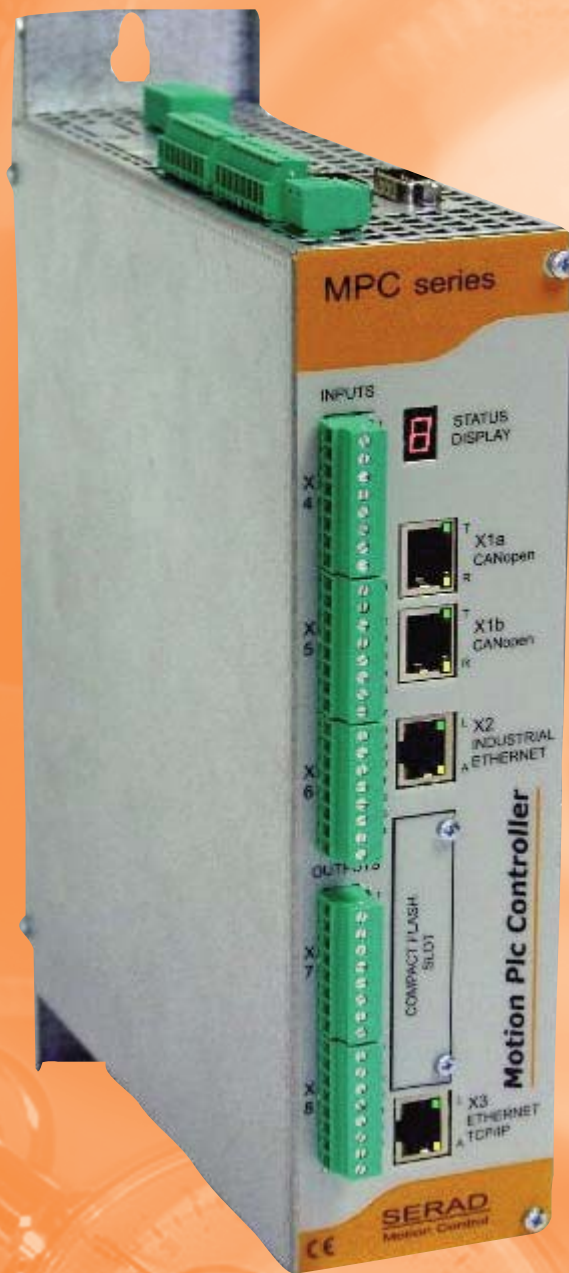


Motion Plc Controller MPC Series

Industrial PC for Automation



Motion

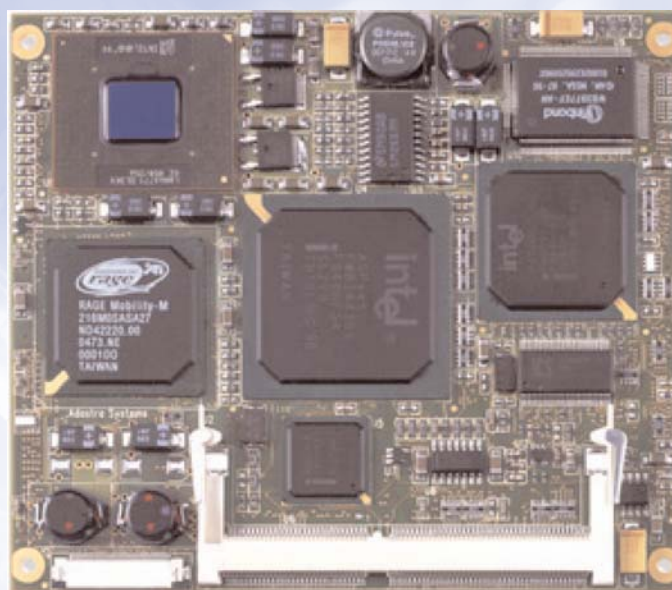
Automation

Industrial
PC

PLC
IEC 61131-3

PC core in an industrial environment

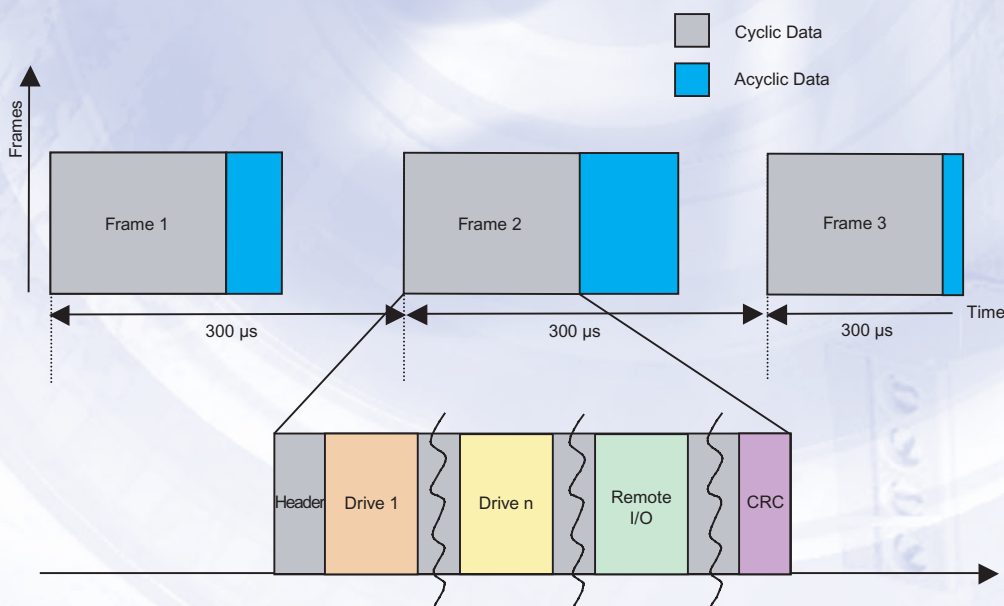
- **Power supply 24 Vdc**
 - **Galvanic isolation**
 - **Tolerates momentary supply loss**
- **Intel® Celeron® processor 400 MHz**
- **Compact Flash memory 64 MB**
- **RAM memory 64 MB**
- **RAM saved data memory 128 KB**
- **Real time clock**
- **Watchdog**
- **Real time multi-tasking core**
- **Operating temperature range from 5 to 45°C**



Motion architecture

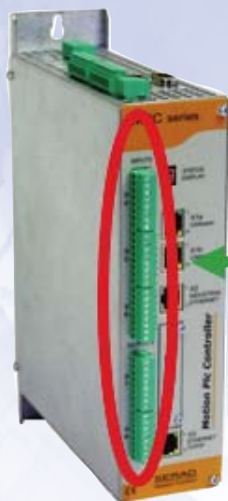
- **Control up to 40 brushless drives via EtherCAT® 100 Mbit/s industrial Ethernet network**
 - **Real time and deterministic Ethernet**
 - **Transfert of cyclic data (commands and position feedback) : 300 µs update cycle time**
 - **Transfert of acyclic data (motor & drive parameters)**
 - **Jitter lower than 1 µs**

Ether**CAT**®



Plc architecture

- **Local Inputs / Outputs : optional I/O module**
 - 32 digital inputs PNP 24 Vdc
 - 24 digital outputs PNP 24 Vdc 0,5A
 - LED state visualization
 - Connection by screw connector blocks
- **Remote Inputs / Outputs**
 - CANopen fieldbus
 - EtherCAT® industrial Ethernet
 - Modbus RTU serial link



Local 56 I/O



Remote I/O

Communication

- **Ethernet TCP / IP 100 Mbit/s**
 - Link with Motion Studio software on PC
 - OPC Server
 - http Server
- **EtherCAT® 100 Mbit/s industrial Ethernet**
- **CANopen 1 Mbit/s**
- **CANbus 1 Mbit/s**
- **Modbus RTU master**
- **Modbus RTU slave**
- **RS 232 / RS 485 ports with free protocols**
- **USB ports**

TCP/IP

Ether**CAT** 

USB TX/RX

CANopen

MPC

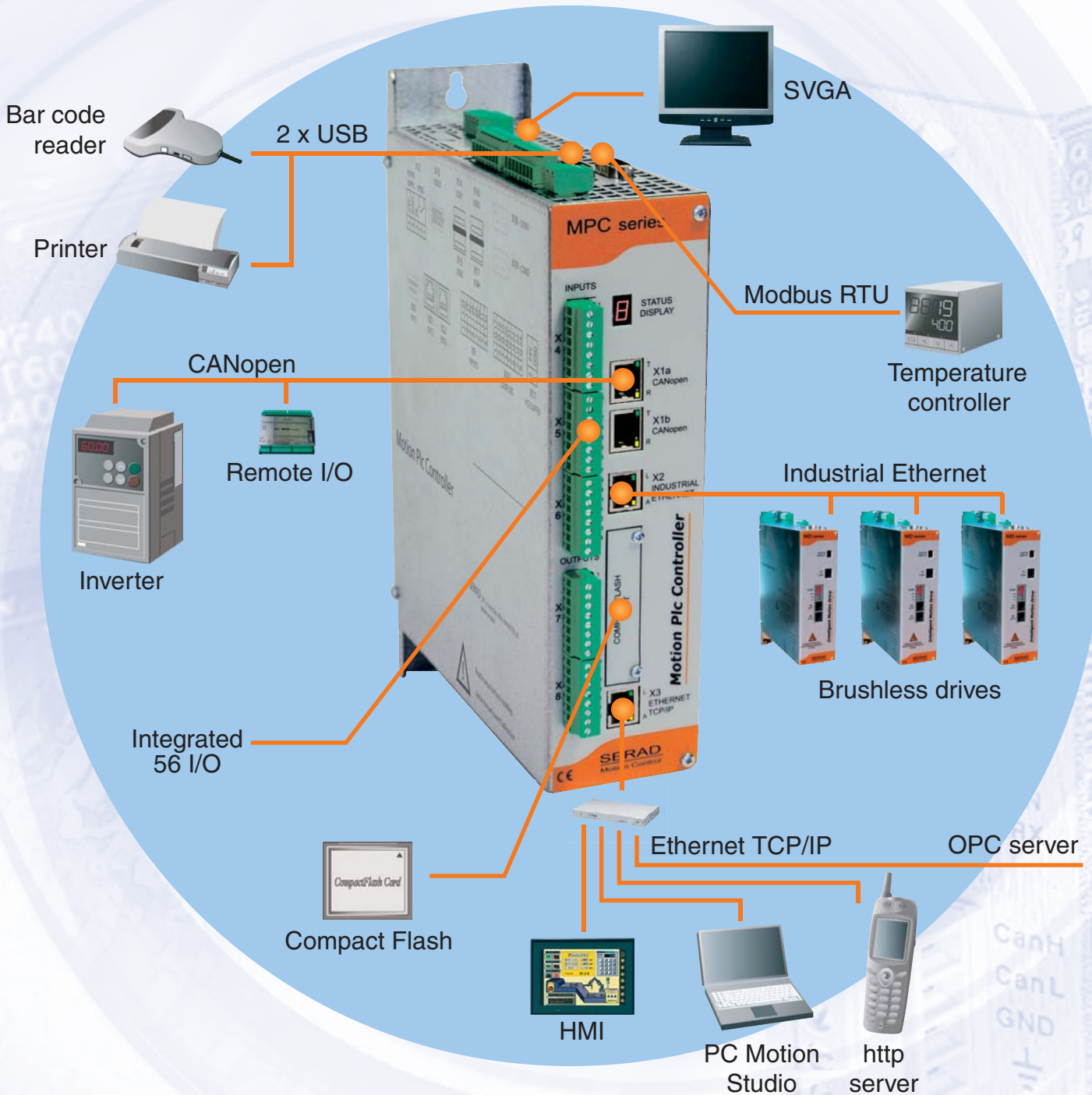
CANbus

Modbus

RS232

RS485

Operation



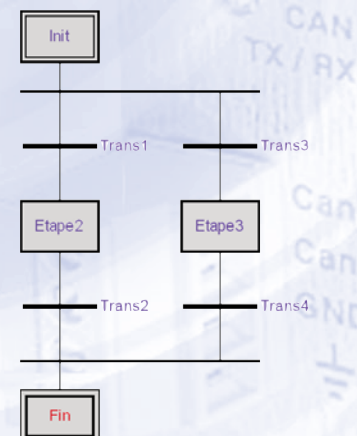
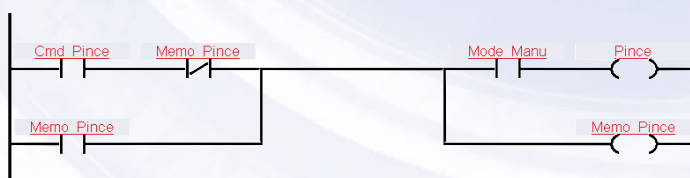
Advanced Motion functions

- **Fast and accurate positioning**
 - Absolute movement
 - Relative movement
 - Infinite movement
 - Speed profile modification on the Fly
- **Electronic gearbox**
 - Adjustable ratio
 - Clutch on the Fly
 - Dynamic phase adjustment
- **CAM profiles**
 - Profile definition with graphic tool
 - Sequence of several profiles
 - Master and slave phase adjustment
- **Registration**
 - Position capture less than 10 μ s
 - Programmable window
- **Synchronized movements**
- **Triggered movements**
 - On fast digital input
 - On position
- **Linear, circular and helical interpolations**
- **Virtual axes**
- **CAMBOX functions**

Plc IEC 61131-3



- **Languages**
 - **LD** ladder
 - **SFC** sequential functions
 - **FBD** function blocks
 - **ST** structured text
 - **IL** instructions list
- **Functions & function blocks**
- **Data types**
 - **BOOL, BYTE, WORD, DWORD**
 - **SINT, INT, DINT, USINT, UINT, UDINT**
 - **REAL, STRING, ...**
- **Variables**
 - **Global, external, local, inputs, outputs**
- **Tasks**
 - **Cyclic**
 - **Acyclic**
 - **Interruptive**



Motion Studio

- Software operating under Windows 2000 / XP
- Easy to use
- Multi-window
- Tool boxes
- Project management
- Simplified setup using tree structure
- Graphical editors
- Setup tools

The screenshot displays the SERAD Motion Studio - StarPrint34 interface. It features a multi-window environment with the following components:

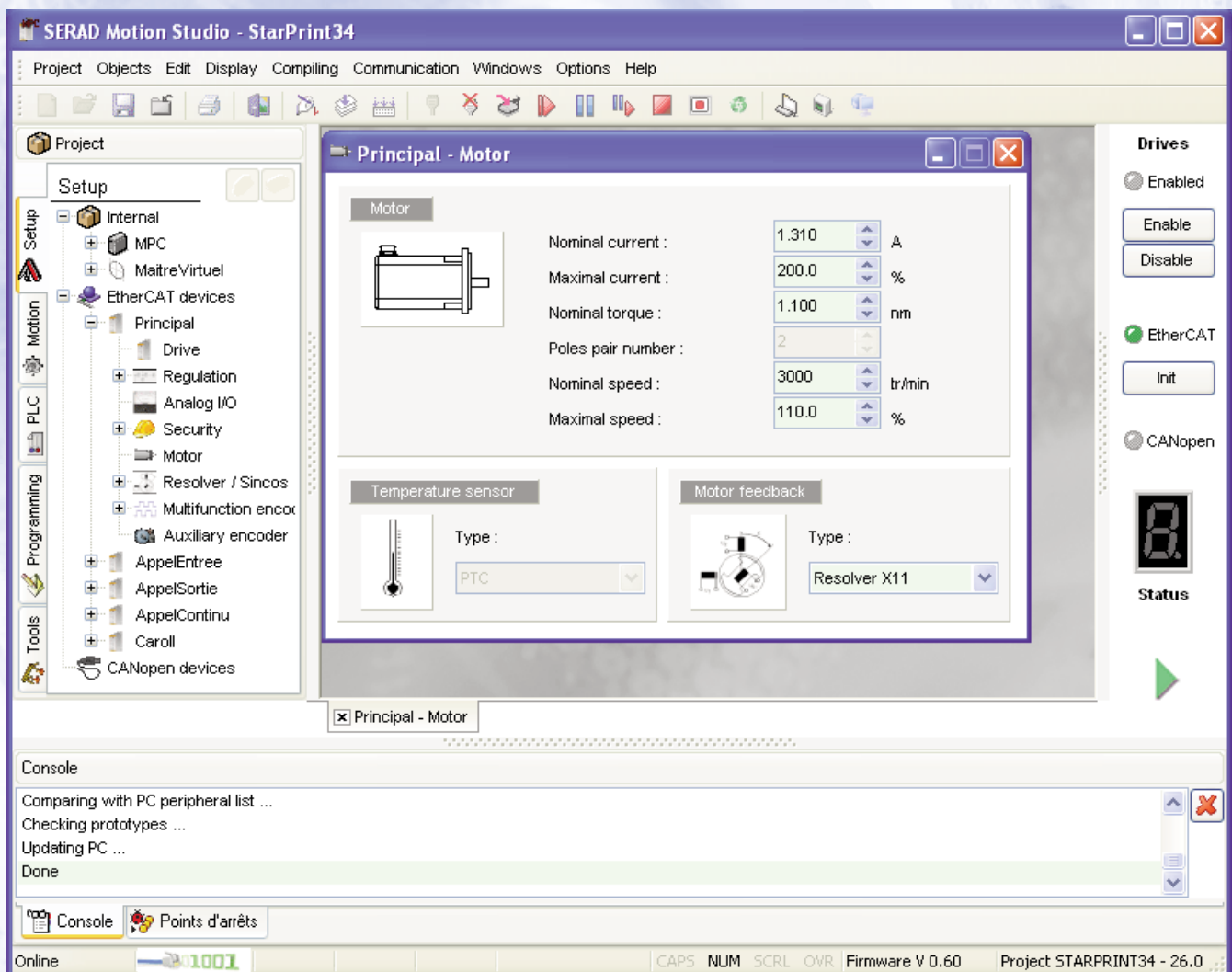
- Project Tree (Left):** A hierarchical view of the project structure, including 'Motion', 'Axes', 'Principal', 'Main', 'Profile', 'Home', 'Control', 'Auxiliary', 'AppelEntree', 'AppelSortie', 'AppelContinu', 'Carroll', 'MaitreVirtuel', and 'Cam profiles'.
- Code Editor (Top Center):** Displays ladder logic code for 'RAM IMPRIM_T'. The code includes conditional statements for speed limits and acceleration/deceleration profiles, such as:


```

      If V1Tempo_Real>=ModuloMaitre Then
      V1Tempo_Real:=V1Tempo_Real-ModuloMaitre;
      End_If;
      DebReculAppel:=V1Tempo_Real;
      FinReculAppel:=Cote;
      (* Formation de la boucle *)
      (* ## 24/02/06 MaitreAcc1 remplacé par MaitreAcc1 ou Acc2! ou Dec1 ou
      V1Tempo_REAL:=DINT_TO_REAL(TRUNC(LgBoucle/Format));
      D_Tmp:=(LgBoucle-Format*V1Tempo_REAL));
      Nbr:=REAL_TO_USINT(V1Tempo_REAL*1.0);
      If (D_Tmp<(MaitreAcc1/2.0)) Then
      Dist:=D_Tmp*2.0;
      Else
      Dist:=MaitreAcc1+(D_Tmp-(MaitreAcc1/2.0));
      End_If;
      For I:=1 To Nbr Do
      If I=1 Then
      G_Tmp:=G1IssementLu;
      S1:=G1IssementLu;
      S1:=SurTensionInit;
      Else
      G_Tmp:=0.0; (*prise en compte gliss. au 1er cycle*)
      S1:=0.0;
      End_If;
      Mvs(Principal,MaitreVirtuel1,SRC_AXE,1000.0,1000.0,0.0,0.0);
      Mvs(AppelSortie,MaitreVirtuel1,SRC_AXE,(MaitreD1+G_Tmp),(AvantD1+S1));
      Mvs(AppelEntree,MaitreVirtuel1,SRC_AXE,(MaitreD1+G_Tmp),ArriereD1,Mat);
      If Format14p=0 Then
      Mvs(Principal,MaitreVirtuel1,SRC_AXE,1000.0,1000.0,0.0,0.0);
      
```
- Speed Profile Graph (Top Right):** A trapezoidal speed-time graph showing a maximum speed of 150.000 mm/s, an acceleration phase of 9.20 s, a deceleration phase of 2.30 s, and an acceleration value of 125.000 mm/s². It also includes settings for 'Urgent deceleration' at 500.000 mm/s².
- Instrument Display (Bottom Center):** A panel showing real-time data for 'Pt', 'Ballast', and 'IGBT' with percentage and temperature scales, and a 'Position' display showing a value of 152.4 degrees.
- Control Panel (Bottom Right):** Includes 'Drives' status (Enabled), 'EtherCAT' and 'CANopen' options, 'States' (Position, HomeAux_s, Aux Pos, HomeAux_s), 'Following err.' (Femax_s, Speed, Move_s), and 'Commands' (P1, P2, 920.00 ms, Acc%, 100.00, Absolute P1).

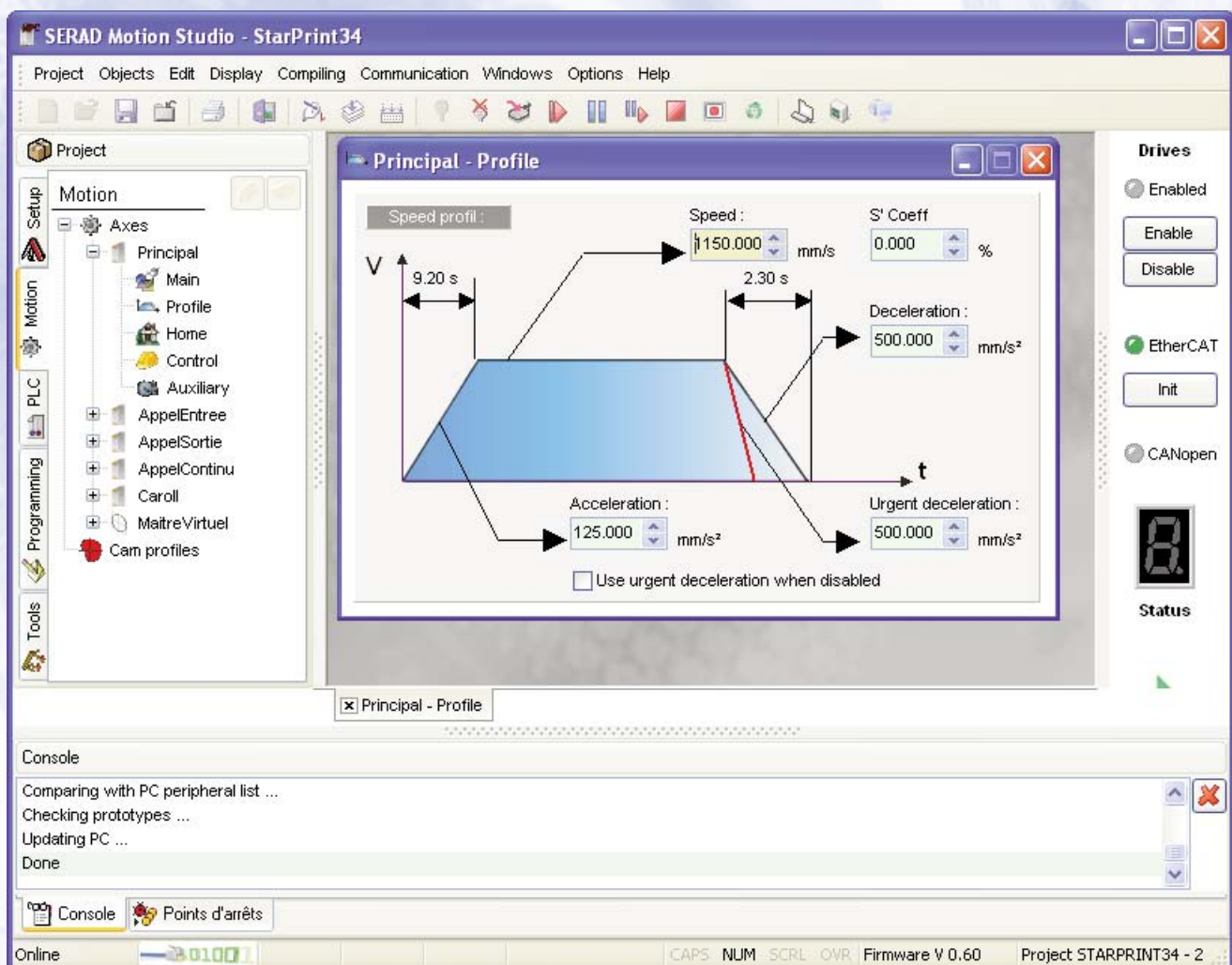
Motion Studio - System setup

- Intuitive navigator
- Tree structured lists
- Simple and fast addition of a peripheral
- Access to the all axis and I/O parameters
- Graphic windows



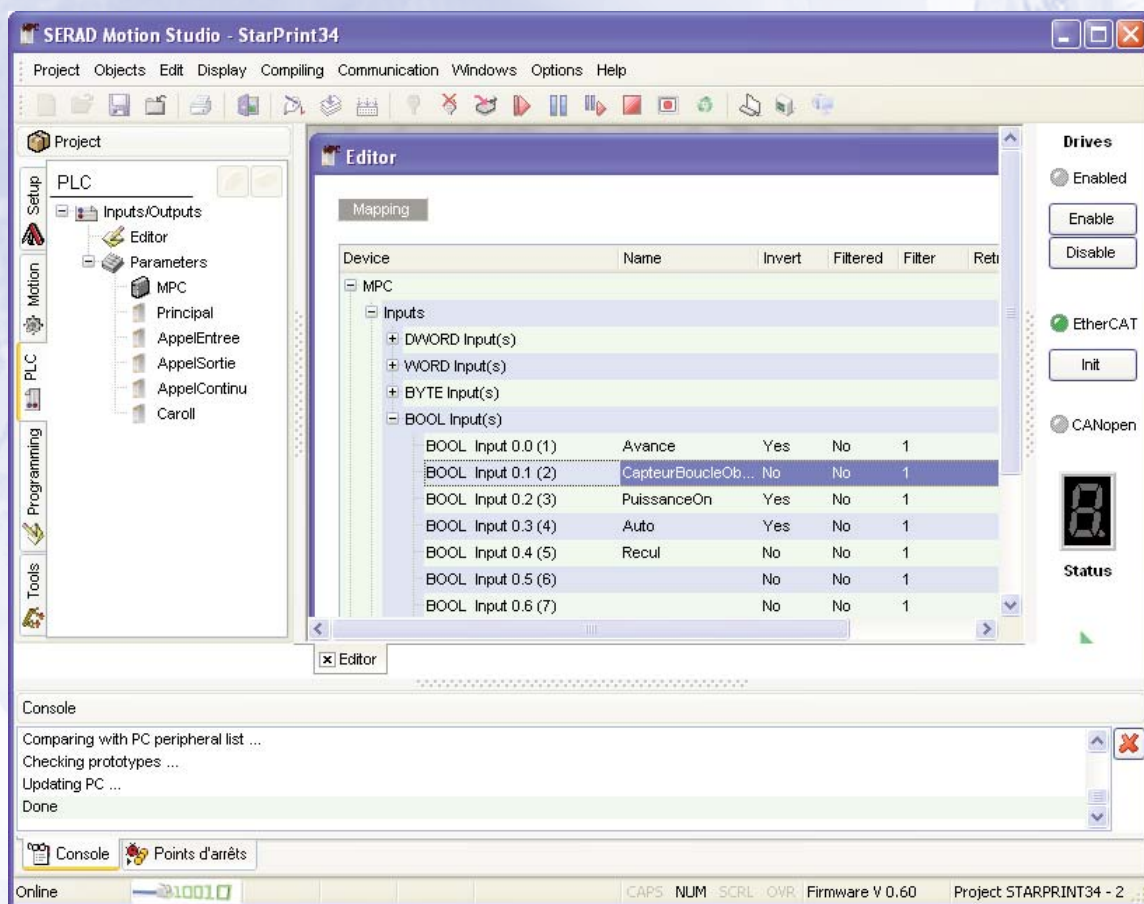
Motion Studio - Motion definition

- Machine mechanics (gearbox ratio ...)
- Axis units (mm, degree, revolution ...)
- Setup of acceleration, deceleration, velocity
- Choice of homing method
- Axis monitoring (hardware limits, following error)



Motion Studio - Plc definition

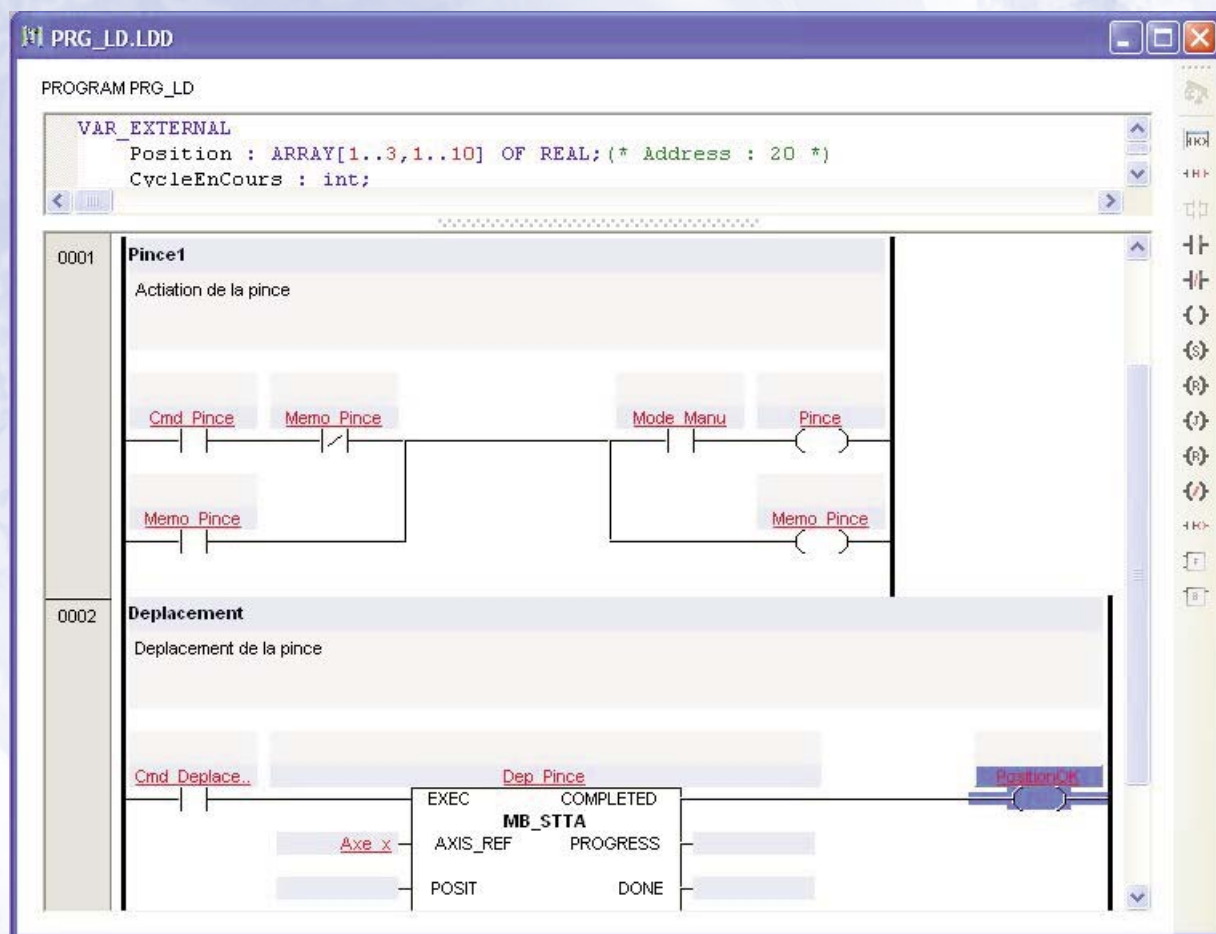
- **Tree structured lists for the I/O modules**
- **Name for each I/O**
(Ex: Start_Button, Clutch_On, Tool_Ok ...)
- **Declaration of each channel in positive or negative logic**
- **Slow or fast filter for each input**
- **Security state for each output**



Motion Studio - Programming

LD Editor - Ladder

- Comment for each network
- NO, NC contacts
- Normal, negated, set, reset coils
- Jumps and labels
- Function or block function calls



Motion Studio - Programming

SFC Editor – Sequential functions

- **Powerful graphic tool**
- **Fast declaration of steps and transition conditions**
- **Jumps and labels**
- **Easy commissioning thanks to the integrated trace mode**

The screenshot displays the SFC Editor window for a program named PRG_SFC.SFC. The interface is divided into two main sections:

- Left Panel (SFC Diagram):** Shows a Sequential Function Chart (SFC) with the following structure:
 - Init** (Start Step) transitions to **Etape2** via **Trans1**.
 - Etape2** transitions to **Fin** via **Trans2**.
 - Init** also transitions to **Etape3** via **Trans3**.
 - Etape3** transitions to **Fin** via **Trans4**.
- Right Panel (Editeur ST):** Shows the corresponding STL (Statement List) code:

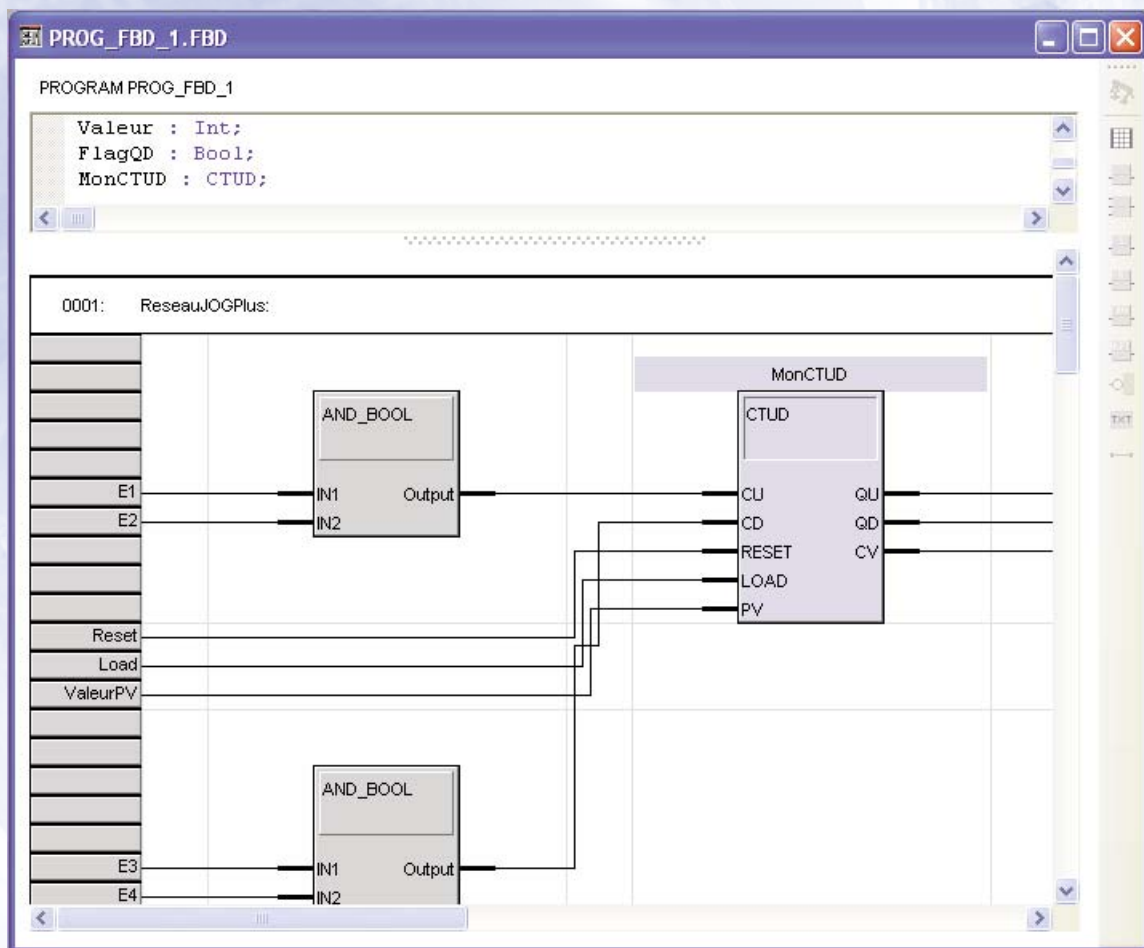

```

      (*programme réalisant les mouveme
      ;
      ;
      (* Prise en compte d'une nouvelle
      Vel(Axe_X,ManualVel);
      Vel(Axe_Y,ManualVel);
      Vel(Axe_Z,ManualVel);
      ;
      (* Mouvement absolu position 0 pa
      CycleEnCours:=1;
      STTA(Axe_X,0.0);
      STTA(Axe_Y,0.0);
      STTA(Axe_Z,0.0);
      WaitBegin(0) OR Wait((MOVE_S(
      ;
      (* Mouvement absolu position 100
      CycleEnCours:=2;
      MovA(Axe_X,position[1,1]);
      CycleEnCours:=4;
      MovA(Axe_Y,position[2,1]);
      
```


Motion Studio - Programming

FDB Editor – Function blocks

- **Powerful graphic tool**
- **Functions Move / Replace**
- **Many standard blocks available**
- **Easy commissioning thanks to the integrated trace mode**



Motion Studio - Programming

ST Editor – Structured text

- **Syntax highlighting**
- **Functions Copy / Paste**
- **Fast text search**
- **List of variables for simple access**
- **Easy commissioning thanks to the integrated trace mode**

The screenshot shows a window titled 'IMPRIM_T.ST' containing the following structured text code:

```

PROGRAM IMPRIM_T

VAR_EXTERNAL
  DefautValidationVar :UINT;
  AccMaitre:Real;
  DecMaitre:Real;
  Format:Real;

FlagSyncro:=0;
WATCHDOGON(0);
ModuloMaitre:=ReadParamR(Principal,_IMD_POSITION_MODULO_V;
Axis(MaitreVirtuel,1);
Axis(Principal,1);
Axis(AppelEntree,1);
Axis(AppelSortie,1);
Axis(AppelContinu,1);
WAIT(&AXISREQUEST_S(MaitreVirtuel)=0); (* Attente fin de de
If (AXIS_S(maitreVirtuel)=0) Then
  DefautValidationVar:=1;
  Wait (DefautValidationVar=0);
End_If;
WAIT(&AXISREQUEST_S(Principal)=0); (* Attente fin de demanc
If (AXIS_S(Principal)=0) Then
  DefautValidationVar:=2;
  Wait (DefautValidationVar=0);
End_If;
  
```


Motion Studio - Programming

IL Editor – Instruction list

- **Syntax highlighting**
- **Functions Copy / Paste**
- **Fast text search**
- **List of variables for simple access**
- **Easy commissioning thanks to the integrated trace mode**

```

PROGRAM PROG_IL

VAR
  IN1_STRING:STRING;
  IN2_STRING:STRING;
  Q_STRING:INT;
  IN_BOOL:

```

```

(*ld'nw'0004*)
Etiquette1:
LD      IN
ST      _ldBOOL_1a          (*ld'and*)
ST      _ldFIND_STRING_1.EN  (*ld'fb'en*)
CAL     _ldFIND_STRING_1(
        IN1      := IN1_STRING, (*ld'fb'in*)
        IN2      := IN2_STRING  (*ld'fb'in*)
        |  Q_STRING := OUT)     (*ld'fb'out*)
LD      _ldFIND_STRING_1.ENO (*ld'fb'eno'absolute*)
ST      _ldBOOL_1a          (*ld'and*)
ST      F_TRIG1.CLK        (*ld'fb'en*)
CAL     F_TRIG1
LD      F_TRIG1.Q          (*ld'fb'eno'replace!*)
RETC

(*ld'nw'0005*)
LD      IN
ST      Q

```

Motion Studio - Programming

CAM profile Editor

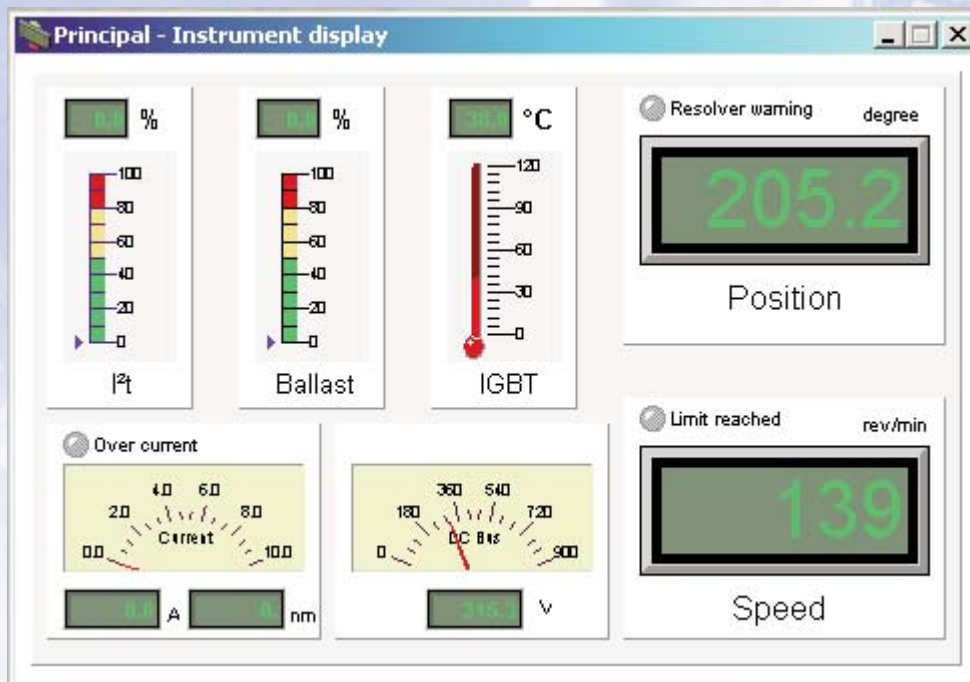
- **Advanced graphic tool**
- **Curves with polynomials in x3**
- **Addition, modification of dots in a simple and fast way**
- **Functions Move / Replace / Zoom**
- **Visualization of position, velocity and acceleration curves**



Motion Studio–Setup tools

Instrument panel

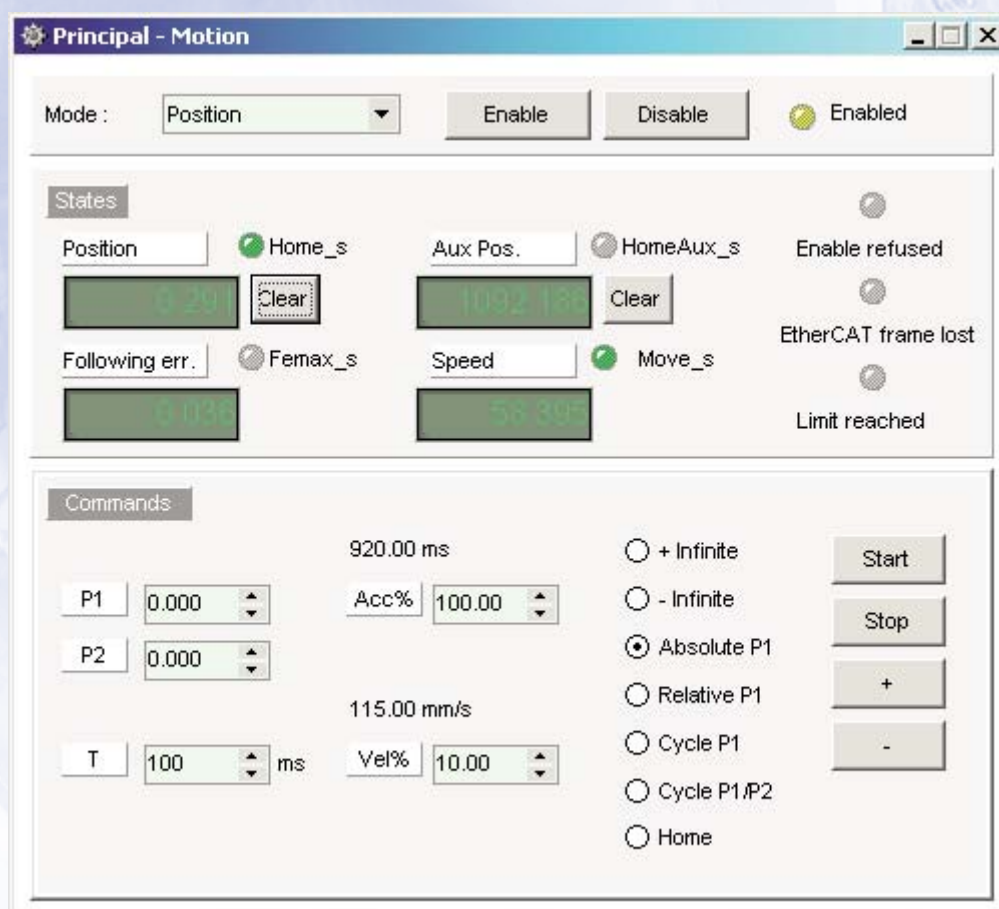
- Visualization of motor position and motor velocity
- Graphs
 - Current and torque
 - I2t control
 - Drive temperature
 - Ballast resistor load
 - Bus voltage



Motion Studio–Setup tools

Motion

- Position, following error, velocity ...
- Visualization of the axis status
- Setup of velocity, acceleration, deceleration
- Jog +/-, go to a position, forward / backward cycle
- Selection of a movement : infinite, absolute or relative
- Start home cycle



Motion Studio–Setup tools

HyperTerminal

- Program state and trace
- Customized visualization of
 - Variables
 - Parameters
 - Axis and I/O states

The screenshot displays the MPC - Hyper-Terminal interface. On the left, the 'Instances' tree shows a hierarchy starting with 'Internal', followed by 'EtherCAT devices' (Principal, AppelEntree, AppelSortie, AppelContinu, Caroll) and 'CANopen devices'. Under 'Programs', the 'BANK1' instance is expanded to show 'ACCMAITRE : REAL' and 'CONFIGPOM : USINT'. Below this, the 'Tasks' section shows a list of tasks with their status and charge percentages: INIT_T (0.00%), BOUCLE_T (42.97%), DIALOG_T (0.00%), CYCLE_T (0.00%), and IMPRIM_T.ST (0.00%).

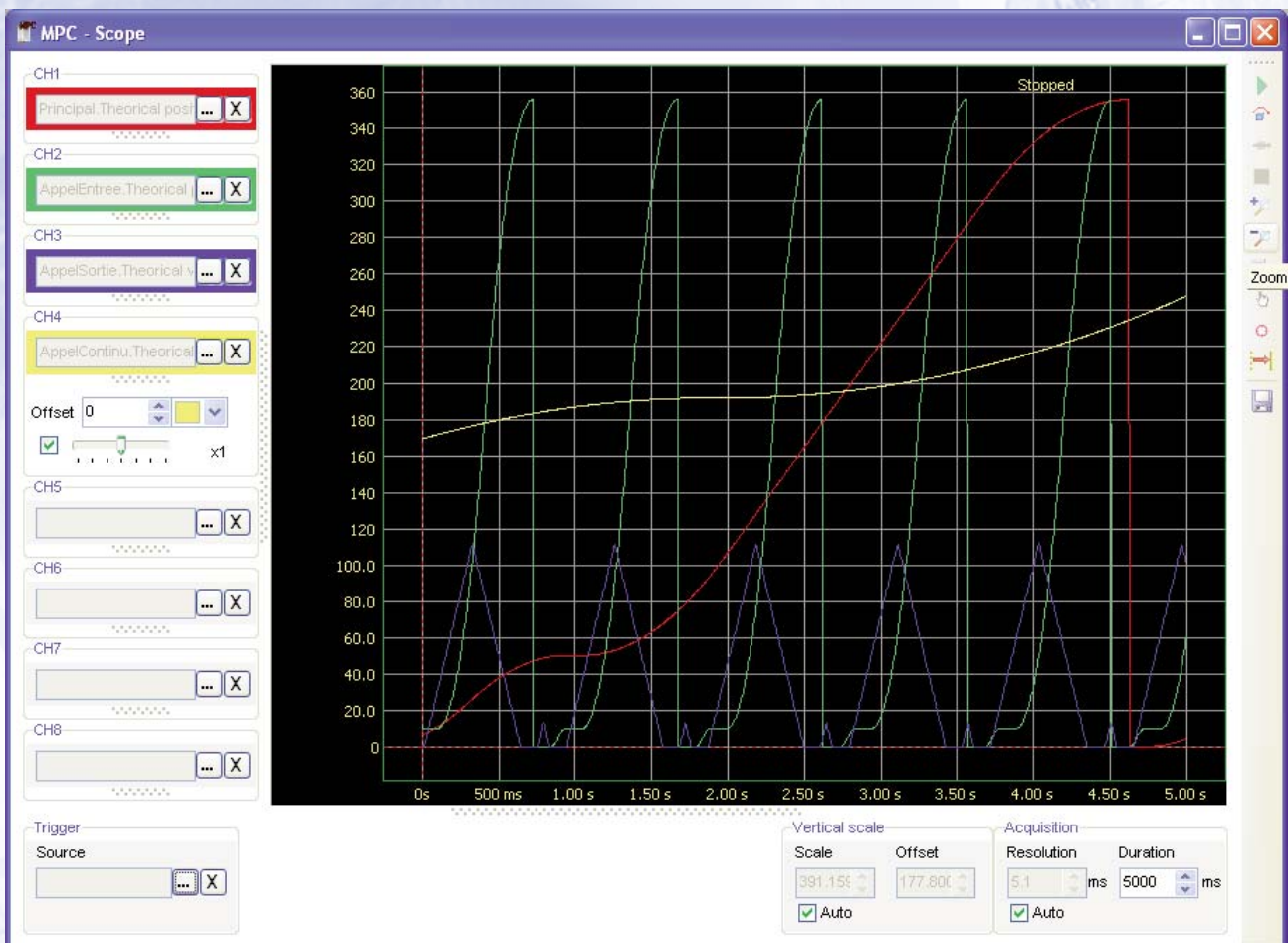
On the right, the 'Views' section shows a list of variables and their current values, with the format set to 'Decimal':

Variable	Value
GLOBAL.CYCLE : USINT	0
GLOBAL.DEMANDEACQUITDEFAULT : 0	0
GLOBAL.DEFAULT : UINT	2
Principal.Position réelle (mm)	353.101
AppelEntree.Position réelle (mm)	319.800
AppelSortie.Position réelle (mm)	13.901
AppelContinu.Position réelle (mm)	334.299
GLOBAL.FLAGSYNCRO : UINT	0
IMPRIM_T.AVANTD1 : REAL	0
IMPRIM_T.AVANTD2 : REAL	0
IMPRIM_T.ARRIERED1 : REAL	0
IMPRIM_T.ARRIERED2 : REAL	0
IMPRIM_T.MAITREACC1 : REAL	0
IMPRIM_T.MAITREACC2 : REAL	0
BANK1.FORMAT : REAL	260
BANK1.ACCMAITRE : REAL	10
BANK1.ZERO : REAL	0
BANK1.DSTABILDEB : REAL	2
BANK1.DSTABILFIN : REAL	0
BANK1.PACC1 : REAL	28
BANK1.PACC2 : REAL	22
BANK1.PDEC1 : REAL	22

Motion Studio–Setup tools

Oscilloscope

- 8 simultaneous channels
- Sampling period of 300 μ s
- Visualization of axis and I/O states
- Continuous or event driven trigger
- Automatic curve scaling
- Zoom function



Motion Studio - Integrated help

- On-line help
- Search of keywords
- Hypertext links
- Parameter and instruction lists
- Detailed syntax for each instruction

English MPC documentation v0.59

Masquer Page précédente Page suivante Imprimer Options

Sommaire Index Rechercher Favoris

- Introduction
- Creating project
- Languages
- Tasks
- Program
- Functions
- Function blocs
- Data
- Motion control programming
 - Introduction
 - Motions buffer
 - Controlled / non-controlled
 - Setting an axis
 - Declaration of an axis in n
 - Declaration of a virtual ax
 - Positioning
 - Absolute movements
 - Relative movements
 - Infinite movements
 - Stopping a movement
 - Triggered movement

Absolute movements

Start a movement: STTA

To launch a movement to an absolute position and not wait till it is over to continue the task execution, we must use STTA.

This instruction is very useful if the velocity or the position to reach changes during the motion.

With this function, absolute error is minimal.

This instruction is not jamming for the task (except if the motion buffer is full).

It uses the current values of acceleration, deceleration and velocity.

The syntax is:

STTA (<Axe>, <Position>)

For example:

```
VELP (X, 100.0); (* normal velocity *)
STTA (X, 2000.0); (*Absolute start to
```