Motion-Logic Programming Interface

Presentation Structure

- Introduction & Motivation
- Design & Implementation
- Summary & Outlook

Motion-Logic Programming Interface

Application Example (HFFS-Machine)

Cross-Cutter
Shoulder
Conveyor

Unloading Conveyor
Finwheel
Motion Controller

Motion Controller Issues

- Motion Controller (option 1)
  - Legacy hard- and software
  - Proprietary µC hardware
  - Real-time operating system
  - Software coded in C/C++ or similar
  - Proprietary fieldbus interfaces
- Disadvantages
  - Development efforts and cost
  - Adaptation of new technologies
  - Obsolescence of components

- Motion Controller (option 2)
  - Standard hard- and software
  - Industrial PLC and software environ.
  - Software coded IEC 61131 or other PLC programming language
  - Standard fieldbus interfaces
- Disadvantages
  - IEC code is automation technology
  - Technology gap between automation business and computer science

Source: Bosch Rexroth AG
Motion-Logic Programming Interface

Objective

Presentation Structure

- Introduction & Motivation
- Realisation & Implementation
- Summary & Outlook

System Interface Requirements
Motion-Logic Programming Interface
User Application Requirements

- Motion control application requirements
  - High precision **synchronous threads**
    - In particular for equidistant setpoint generation and communication
  - **Asynchronous threads** for non-motion tasks
    - Not necessarily in synch with the motion bus
    - Improves system performance
  - **Free running threads**
    - For low priority tasks

Motion-Logic Programming Interface
Performance Levels Requirements

- **Control**
  - Hardware A
  - Firmware A
  - PLC with Fieldbus Master
  - Standard Motion
  - Basic Technology Func.
- **Robotics & Kinematics**
  - Advanced Technology Func.
  - Standard Motion
  - Standard Motion

Motion-Logic Programming Interface
Derived Design Requirements for the Interface

- Design requirements:
  - Independency from the programming language
  - Well tailored layer architecture
  - Structured according to functionality
  - Object oriented design approach
  - Network support
  - High performance
  - Small footprint and multi-instance capability

Motion-Logic Programming Interface
MLPI Architecture
Motion-Logic Programming Interface
MLPI Structure in Libraries (1)

- Administrative functions
  - API library
    - Connect and disconnect the interface
  - System library
    - System settings and configuration
  - Device library
    - SERCOS devices and configuration

- PLC functions
  - Task library
    - Creating, destroying and configuring tasks
  - InOut library
    - Direct or symbolic access to fieldbus data and data exchange buffers
  - PLC library and Watchdog library
    - Load, start, stop and supervise PLC applications

Motion-Logic Programming Interface
MLPI Structure in Libraries (2)

- Motion functions
  - Parameter library
    - Parameter access to controls and drives
    - Read, write and modify single parameters as well as list parameters and attributes
  - Motion library
    - Create and destroy real axis or virtual axis
    - Parameterization of axis properties
    - Commanding axis with discrete motion, continuous motion and synchronous motion command
  - Robot library
    - Group and ungroup axis to configure kinematics
    - Load, select and execute robot control applications including

Motion-Logic Programming Interface
Deployment Diagram

Motion-Logic Programming Interface
Presentation Structure

- Introduction & Motivation
- Realisation
- Summary & Outlook

Source: Bosch Rexroth AG
Motion-Logic Programming Interface
Summary & Outlook

- Motivation for the coexistence of IEC code and C/C++ code
  - migration purpose of existing legacy applications
  - interfaces for research and scientific applications to standard automation environments
- Broad range of functionality implemented for
  - single axis motion
  - synchronized axis motion
  - coordinated motion for robots
- Implementation of high level interfaces for rapid prototyping and system engineering