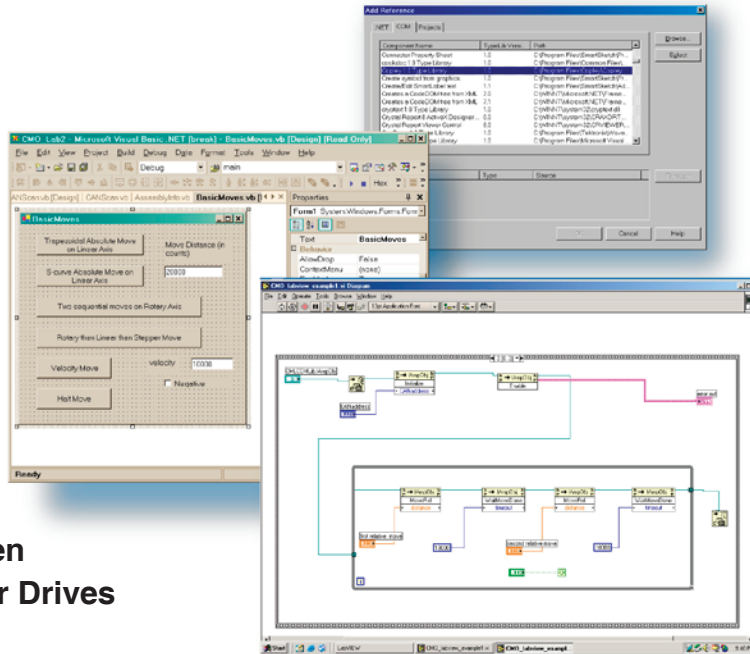


LabVIEW and Visual Basic Software Tools for CANopen Distributed Control



- Supports CANopen Servo & Stepper Drives
- Works with
LabVIEW
Visual Basic 6.0
Visual Basic .NET
Visual C++
Excel

- Automatic Network Management of CAN Bus
- Point-to-Point and Coordinated Motion

DESCRIPTION

CMO greatly simplifies the creation of Windows-based software for controlling Copley Controls digital servo amplifiers and stepper drivers over a CANopen network. CMO gives programmers direct access to an amplifier's CANopen functions from a high-level language without the complexity of low-level CAN bus programming.

CMO can be used with any software that supports the Microsoft COM (Component Object Model, also known as ActiveX), and allows the use of language-independent objects, methods, and parameters. Some typical examples of COM-enabled programs are Microsoft Visual Basic 6.0, Visual Basic .NET, Visual Basic for Applications, Visual C++, Microsoft Office products, and LabView.

CMO provides an interface between high-level languages and applications operating in a Windows environment and low-level communications over a CANopen network. This reduces development time for motion-control applications and frees the programmer from creating, debugging, and managing low-level CANopen communications. Amplifiers can be controlled independently, or linked together into groups that are synchronized over the CAN bus so that complex, multi-axis moves can be executed with coordinated motion. CMO operates under Windows 98 through Windows XP operating in desktop or laptop computers.

Description

WHAT IS CMO?

CMO is a Microsoft COM-compliant software component. Objects are accessible from any COM-enabled program. CMO provides a programming interface to Copley digital servo amplifiers and stepping motor drivers that operate on a CAN bus using the CANopen protocol.

WHY USE CMO?

CMO eliminates low-level coding to support communications over a CAN bus. In addition, CMO eliminates the additional coding needed to support communication with devices operating under the CANopen protocol, the application layer that works over a CAN bus that is designed for motion control and other specialized types. CANopen devices have object-dictionaries that combine dedicated addresses for standard functions and other addresses for device-specific ones.

CMO provides a high-level language interface to low-level functions that is efficient and robust. This greatly reduces development time and time-to-market. At the same time, it enables programmers to focus on their application development and to treat the CANopen interface simply as a library of objects that are ready to use.

HOW DOES IT WORK?

CMO provides an object-oriented interface during program development. CMO communicates with a CAN interface card via the interface driver provided by the manufacturer. At run-time, CMO provides CAN bus control of Copley CANopen products by managing all of the low-level bus communications nec-

GENERAL SPECIFICATIONS

PRODUCT TYPE

Microsoft COM (Component Object Model) compliant software component:
Automation compliant
Single .DLL file

OPERATING SYSTEMS SUPPORTED

Microsoft Windows XP, 2000, NT, ME, 98

CANopen COMPLIANCE

CiA DSP-402, Device Profile for Drives & Motion Control
CiA 301, Application Layer and Communication Profile

HARDWARE REQUIREMENTS (MINIMUM)

400 MHz or greater CPU, with 128 MB RAM
One RS-232 port for amplifier setup with CME 2
One CAN interface device
Copley Controls CANopen servo amplifier or stepping motor driver:
Xenus, Accelnet, Stepnet

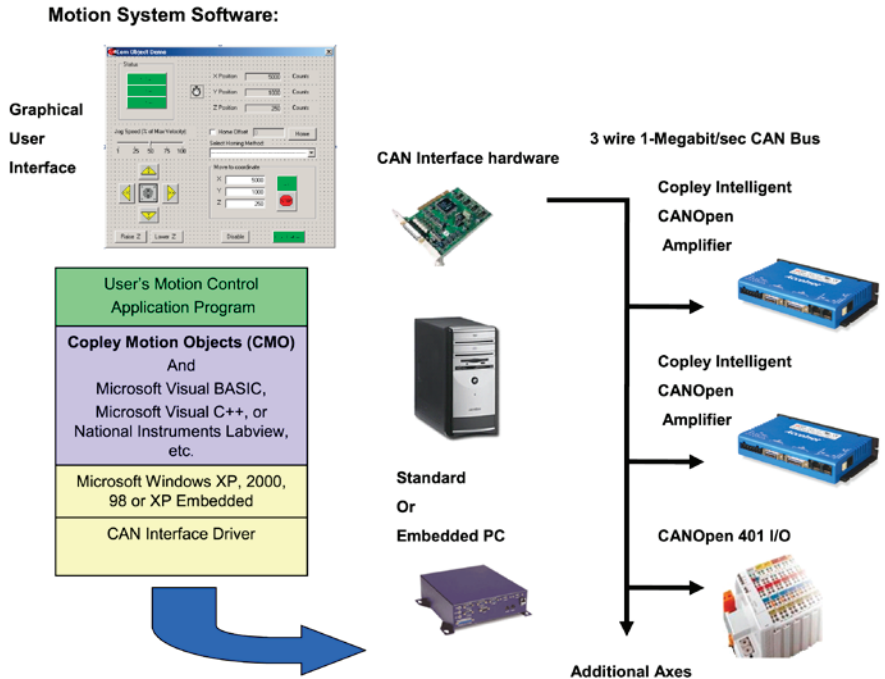
SOFTWARE REQUIREMENTS

CME 2 Version 3.2 or higher ; Copley Controls' application for amplifier setup, tuning, and configuration
Any COM-compliant application for use with CMO. Some examples are:
Microsoft Visual Basic .NET, Visual Basic 6.0, Visual C++, LabView

CANopen HARDWARE SUPPORTED

CAN bus interface products:
Kvaser, lxxat, Vector, National Instruments
I/O products:
Wago

CMO SYSTEM CONFIGURATION

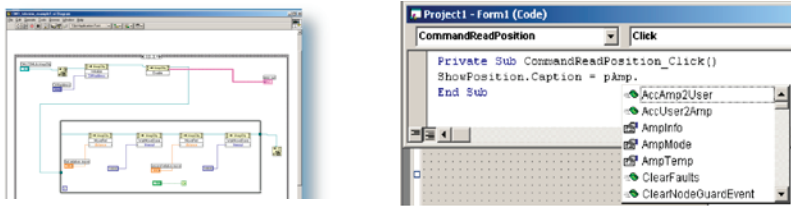


Commonly Used Methods and Parameters

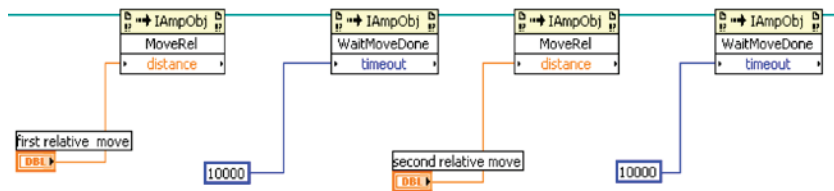
General Function	Method/Property Name	Description
CANopen Object (handles all CANopen Motion DSP-402 standard CAN communications between the PC and the amplifiers)		
CANopen	PortName	The CAN card name and port to be used for CANopen network
	BitRate	The CANopen Bit Rate to be used
	Initialize	Initialize the CANopen network communications
Amplifier Object (handles all control and monitoring communications with an amplifier using CANopen Motion DSP-402 compatible objects)		
Amplifier Initialization	Initialize	Initialize the amplifier on the CANopen network with the appropriate CAN address and start communications
Amplifier Modes and Status Information	Disable	Disable the amplifier
	Enable	Enable the amplifier
	ClearFaults	Clear any latching faults on the amplifier
	ReadEventSticky	Read events with automatic clear
Position and Velocity	PositionActual	Read the actual encoder position
	PositionError	Read the position error (difference between position command and actual position)
Homing	GoHome	Go home executes the homing routine as specified in the home settings object
Quick Stop Support	HaltMove	Halts current move using pre-programmed halt mode
Point-to-Point Move Support	MoveAbs	Perform an absolute point-to-point move using the pre-configured profile settings
	MoveRel	Perform a relative point-to-point move using the pre-configured profile settings
Amplifier Event Processing	WaitMoveDone	Waits for the currently running move to finish, or for an error to occur
	CreateEvent	Create an event that monitors amplifier events for specific conditions
Unit Conversion Functions	CountsPerUnit	Stores a scaling factor for converting between an amplifier's default units (encoder counts) and user-defined units
Profile Settings Object (for setting up motion profile)		
Profile Settings	ProfileType	Gets/sets the profile type (SCurve, Trap, Velocity)
	ProfileVel	Gets/sets profile velocity value (velocity that the motor attempts to reach during the move)
	ProfileAcc	Gets/sets the profile acceleration value (acceleration that the motor uses when starting the move)
Linkage Object (for performing coordinated multi-axis motion)		
Linkage Object Methods	Initialize	Initializes a linkage object by assigning Amplifiers to it
	MoveTo	Multi-axis move. Target positions are passed as an array.
	WaitMoveDone	Wait until the multi axis move is complete. If the move does not complete by the time specified, the function will return
Event Object (for monitoring events from a given amplifier)		
Event Object	Start	Begins monitoring for an event to occur. Generates a single callback to a user subroutine when the chosen event occurs, or timeout has expired
	Stop	Stop monitoring
CopleyMotionLibrary Object (high level object that enables sophisticated debugging)		
Debug logging	DebugLevel	Gets/sets the debug message level. When enabled will record CAN messages from and to all the amplifiers along with actions and faults that are useful in debugging motion programs
IO Object (controls communications with a CANopen DS-401 compatible I/O module)		
Digital I/O	Din8Read, Dout8Write ...	Read a group of 8 digital inputs or write a group of 8 digital outputs
Analog I/O	Ain16Read, Ain16Write ...	Read a 16-bit analog input or write a 16-bit analog output
Event driven I/O	ICreateEvent	Create an event that monitors I/O events for specific conditions

CMO and LabVIEW

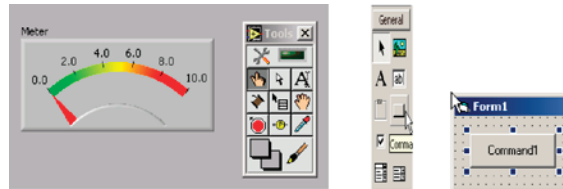
Easy Selection of Methods and Properties



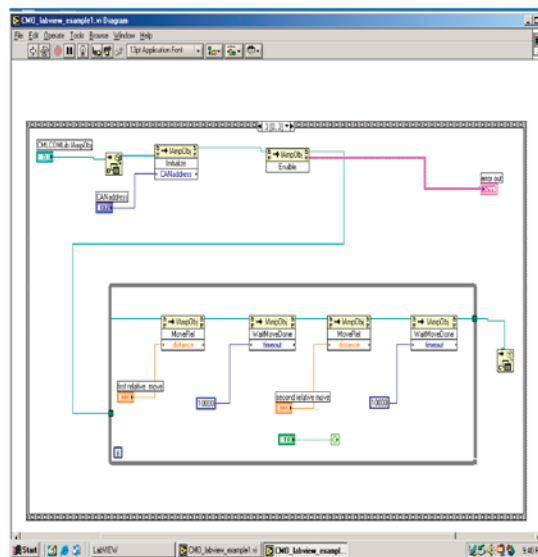
Quickly Create a Powerful and Intuitive Motion Application



Easy Graphical User Interface Design



Create Powerful Motion Control Applications with LabVIEW Building Blocks



LabVIEW is a registered trademark of National Instruments Corporation. ActiveX, Microsoft, Visual Basic, Visual C++, and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other product and company names mentioned herein may be the trademarks of their respective owners.

Rev 1.09_th 03/22/2007