White Paper

Understanding I/O Device CAN Bus Master Capability

Using the SmartMotor™ for Simple, Economical and Effective Total Machine Control

Key Messages

- I/O Device CAN Bus Mastering, its features and how it operates
- Program examples are included to show the simple yet powerful control provided by this technology
- Photos and links to web videos are included, which provide actual working examples of this technology

Summary

In January 2011, Animatics Corporation (now Moog Inc., Animatics) introduced a new servo motor communication technology called “Combitronic™”, which greatly simplifies conventional servo motor communications while also providing a powerful set of features. Note that the ease of use and features provided by Combitronic technology are exclusive to the Moog Animatics SmartMotor™ and are not available from any other motor manufacturer in the industry.[1]

In March 2016, Moog Animatics extended this control by introducing a new capability, I/O Device CAN Bus Master, which allows the SmartMotor to act as a CAN bus master for controlling CANopen I/O devices. When combined with the industry proven Combitronic multi-axis control technology, the SmartMotor can achieve total machine control.

This document describes I/O Device CAN Bus Master capability, its features and how it operates. Program examples are included to show the simple yet powerful control provided by this technology. Additionally, demo descriptions, photos and links to web videos are included, which provide actual working examples of this technology.

What is I/O Device CAN Bus Master?

I/O Device CAN Bus Master capability is a new technology available for the Moog Animatics SmartMotor. It operates in addition to Combitronic technology and allows the SmartMotor to interface with CANopen devices such as:

- Remote digital/analog/mixed signal I/O
- Temperature controllers
- CAN bus absolute or relative encoders
- Pneumatic valve blocks
- Inclinometers
- Load cells

I/O Device CAN Bus Master works with CANopen I/O devices that conform to the CiA 301 specification—it is not limited to just the listed devices.

Like Combitronic technology, I/O Device CAN Bus Master capability operates over a standard “CAN” (Controller Area Network) interface, which is available as an option on the SmartMotor. This is the same basic hardware used in most automobiles as well as in familiar industrial networks such as CANopen® and DeviceNet™.

Important Terminology

Before discussing details of I/O Device CAN Bus Master capability, it is important to understand the following CANopen terms.[2]

- Process Data Object (PDO) communication allows for minimal overhead when transmitting frequently-used data. Typically used for information critical to an ongoing process, which could include the speed, position, control word, etc.

- Service Data Object (SDO) communication is intended for initial setup and occasional access to objects that are not frequently needed. That said, for some applications, it can be used to eliminate PDOs altogether. For an example of this, see Example 1: Timed SDO Poll on page 6.
- Network Management State (NMT) is used to control the general communication functions in the CANopen devices on the network. The primary states are Pre-Operational, Operational, Initialization and Stopped.

What Are the Primary Features?

I/O Device CAN Bus Master basic control allows 8, 16 or 32-bit sized data. The supported protocols are PDO and SDO. The supported profiles include, but are not limited to, I/O profile, Encoder profile and DS4xx profile. In addition, it provides the ability to:

- Dynamically map SmartMotor PDOs, map another device's PDOs and start the NMT state.
- Send/receive, through the SmartMotor, up to five PDOs each for Rx (receive) and Tx (transmit).
- Read/write SDOs in expedited mode only, which works for up to 32-bit data.
- Continuously call SDOs on timer interrupts.
- Achieve sophisticated electronic gearing and camming[3] control over CAN bus.
- Control an entire machine from a program running in just the SmartMotor – no external master (e.g., a PLC) is required.

What Are the Benefits?

The SmartMotor I/O Device CAN Bus Master capability provides the following benefits to the machine builder:

- Simple installation—installation is straightforward because all SmartMotors and CANopen I/O devices install on the same CAN bus.
- Easy implementation—the development process is streamlined by allowing all programming and control from SMI and the SmartMotor.
- An economical solution—machine build and replication costs are reduced by using the SmartMotor as the "master" along with inexpensive CANopen I/O devices.

How Does It Work?

I/O Device CAN Bus Master capability[4] adds functionality that coexists with Combitronic communications. In order to fully understand I/O Device CAN Bus Master capability, it is helpful to first understand how Combitronic technology operates.

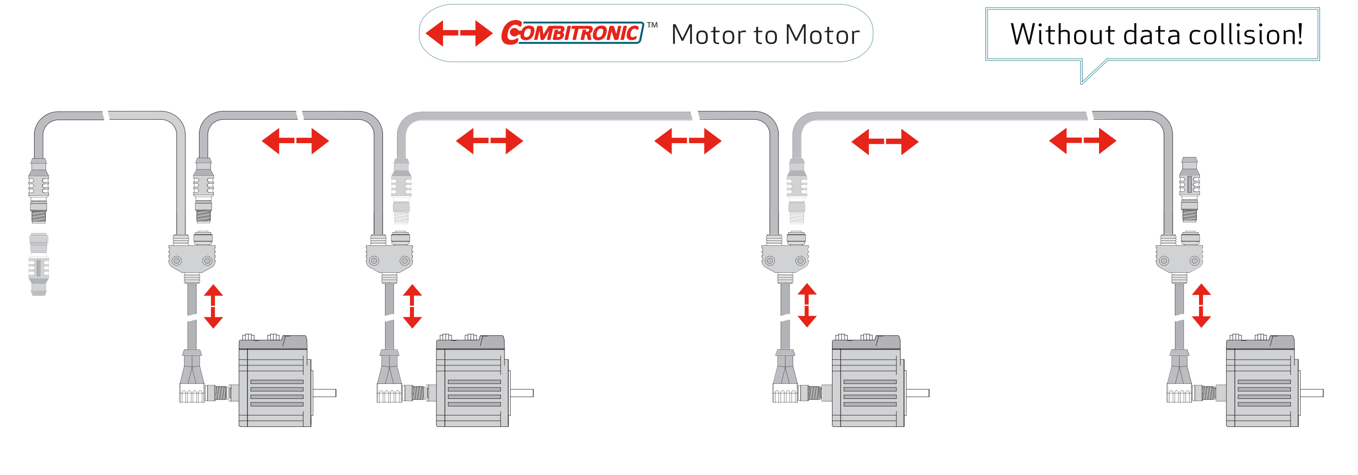
Combitronic technology is a proprietary optional motor-to-motor communication protocol that is available exclusively on the Moog Animatics SmartMotor. It was designed to greatly simplify conventional servo motor communications and provide a powerful set of communication features. It operates on the CAN bus and coexists with either CANopen or DeviceNet protocols. Other than matching baud rates and ensuring unique addresses in each node, there are no other requirements to make it work.

In a traditional network, all commands local to a SmartMotor controller are for that motor only. With the optional Combitronic technology, those same commands can be applied to and reference other motors on the same network as if all the motors were being controlled by a central, multi-axis controller. However, from a user's perspective, the difference in this protocol is that it is not register based or data packet based—it simply uses the typical local commands that are amended with a colon and target node address. For example:

```
G         'Issue Go in local motor
G:2       'Issue Combitronic Go to motor 2
          'on the same network
G:0       'Issue Combitronic global Go to
          'all motors on the same network
```

Refer to the next two figures that show Combitronic communications alone and then combined with I/O Device CAN Bus Master capability. In Figure 1, a Combitronic network of multiple SmartMotors is shown. Each motor can communicate with any other motor on that network, and any motor is capable of being the network master. Up to 120 SmartMotors can exist and communicate on the same network at up to 1 MHz bandwidth[5] with no data collision and no loss of packets.

In Figure 2, I/O Device CAN Bus Master capability is added to the CAN bus network. Now the same SmartMotors that have multi-axis control and communication capability on the Figure 1 network also have control over all of the I/O devices on the Figure 2 network.



Be sure to follow proper guidelines for CAN bus cabling and termination.

Figure 1: Combitronic Motor-to-Motor Communications

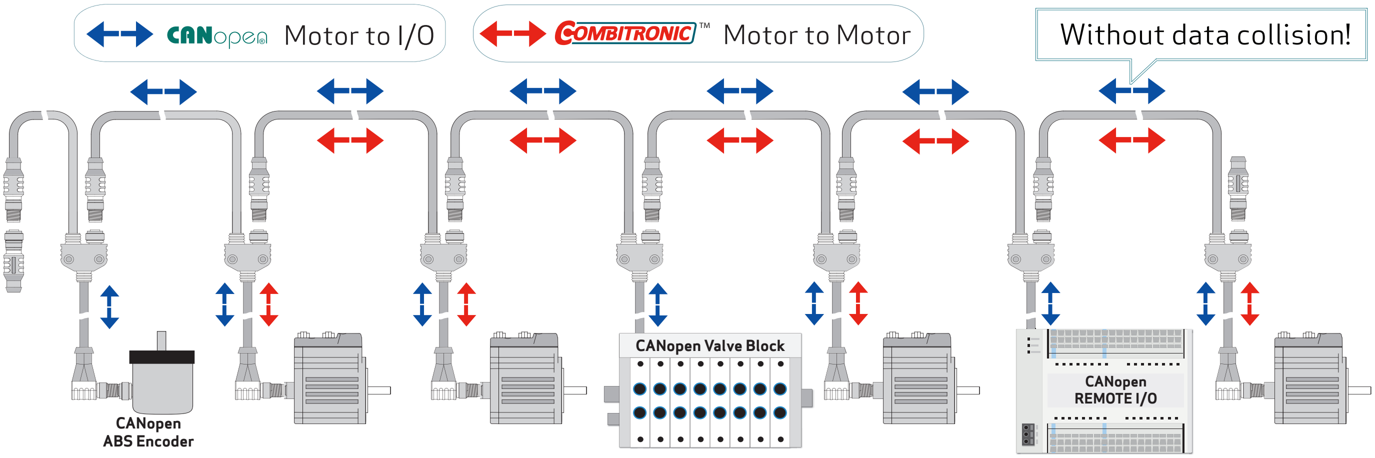
As shown in Figure 2, multiple SmartMotors and multiple CANopen I/O devices reside on the same CAN bus. I/O Device CAN Bus Master capability operates concurrently with Combitronic communications. Combitronic technology is motor-to-motor communications, and I/O Device CAN Bus Master capability is motor-to-I/O communications.

In other words, any motor on the network can communicate with any other motor on that network, and any motor on the network can communicate with any I/O device on that network. This means that the PLC or other CAN bus master that was required before is no longer needed—the SmartMotors alone can handle all multi-axis and I/O control on the CAN bus.

NOTE: The "master" does not need to be the SmartMotor that is receiving all of the data. Therefore, the mastering function can be passed between SmartMotors to optimize the performance and efficiency of the operation.

Commands and Objects

The following sections describe the new commands and supported CANopen objects used to interface with I/O Device CAN Bus Master capability.



Be sure to follow proper guidelines for CAN bus cabling and termination.

Figure 2: Combitronic with I/O Device CAN Bus Master Capability

New Commands

I/O Device CAN Bus Master capability uses the following new AniBasic commands to communicate with the CANopen I/O devices:

- NMT: This command transmits an NMT message to the network; it can command a specific slave or all slaves to enter the commanded state. It uses the form: NMT(target address, desired state)

  ```
  NMT(2,128) 'Tell motor 2 to go pre-operational.
  ```

- SDORD: This command gets (reads) the value from the specified SDO on a specified device. It uses the form: x=SDORD(slave addr, index, sub-index, length)

  ```
  x=SDORD(1,24592,0,2) 'Read 2 bytes from address 1,
                       'object 0x6010, sub-index 0.
  ```

- SDOWR: This command writes a value to the specified SDO on a specified device. It uses the form: SDOWR(slave addr, index, sub-index, length, value)

  ```
  SDOWR(1,9029,0,4,a)  'Write 4 bytes to address 1
  ```

Other related commands provide added functionality:

- B / RB: This command reads (or reports) the status from the specified status word and bit number. It uses the form: =B(word,bit)

  ```
  IF B(4,0)==0         'Check Timer 0 status
  ```

- =CAN / RCAN: This command reads (or reports) an error or other status information about the CAN bus. It uses the form: =CAN(arg)

  ```
  y=CAN(1)             'Get NMT state value and
                       'assign it to variable y
  ```

- CANCTL: This command is used to control CAN network features. Commands execute based on the function argument to control CAN functions. It uses the form: CANCTL(function,value)

  ```
  CANCTL(5,100)        'Set Combitronic timeout
                       'to 100 ms
  ```

For examples of the proper application of these commands, see Programming Examples on page 6.

New Objects

In addition to the new commands, the following new CANopen objects were added:

- 0x2220 8-Bit Mappable Variables: Provides direct read or write access to user variables ab[0]–ab[3].
- 0x2221 16-Bit Mappable Variables: Provides direct read or write access to user variables aw[32]–aw[35].

(The following objects are detailed in the next section.)

- 0x2207: Encoder Modulo Limit
- 0x2208: Encoder Follow Data (8, 16 or 32-bit types)
- 0x2209: Encoder Control
- 0x220a–d: MFMUL, MFDIV, MFA, MFD

Other objects have additional functionality:

- 0x2204: Offers access to 32-bit user variables aaa, bbb, ccc and ddd.

These are available for applications such as general-purpose I/O blocks using PDO communications.

Time Sync Objects

The following objects are used for long-duration velocity, electronic gearing and camming:

- 0x1005 COB-ID SYNC: Specifies the COB-ID used for the Synchronization object (transmit or receive).
- 0x1006 Communication Cycle Period: Defines the communication cycle period in microseconds for transmission of the sync message.
- 0x2207 Encoder Modulo Limit: Defines the encoder modulo limit in units of encoder counts.
- 0x2208 Encoder Follow Data: Accepts data from a network (CANopen) based encoder. Three different data sizes are provided to handle PDO mapping to data sources of 8, 16 and 32 bits.
- 0x2209 Encoder Control: Defines the encoder control for the mode of following a network encoder.
- 0x220a MFMUL (Mode Follow Multiplier): Specifies the multiplier for external encoder mode follow with ratio MFMUL/MFDIV.
- 0x220b MFDIV (Mode Follow Divisor): Specifies the divisor for external encoder mode follow with ratio MFMUL/MFDIV.
- 0x220c MFA (Mode Follow Ascend): Sets the ascend ramp to the specified sync ratio from a ratio of zero.
- 0x220d MFD (Mode Follow Descend): Sets the descend ramp from the specified sync ratio to a ratio of zero.

For more details on these objects, see the *Class 5 SmartMotor CANopen Guide*. For more information on the Mode Follow commands, see the *SmartMotor Developer's Guide*.

Defining the Time Sync Objects

The following code snippets show previously discussed objects 1005h, 1006h, 2207h, 2208h and 220Ah–220Dh being defined for use in later code.

```
'++++ HEX Coded Objects for CAN +++++
. . .
#define x1005  4101   'Object 1005h:   COB-ID Sync
#define x1006  4102   'Object 1006h:   Communication Cycle Period

. . .
#define x2207  8711   'Object 2207h:   External encoder follow max value (where
                          'encoder rolls over) i.e., 10 bit encoder would be 1023
#define x2208  8712   'Object 2208h:   External encoder follow input value
#define x2209  8713   'Object 2209h:   External encoder follow control
#define x220A  8714   'Object 220Ah:   External encoder follow MFMUL
#define x220B  8715   'Object 220Bh:   External encoder follow MFDIV
#define x220C  8716   'Object 220Ch:   External encoder follow MFA
#define x220D  8717   'Object 220Dh:   External encoder follow MFD
. . .

'Set up the sync producer/consumers and set time base.
'This provides time sync so motors' clocks keep in step, and the data
'is transmitted and accepted on sync as well.
SDOWR(mmm,x1006,0,4,10000)        GOSUB10    'define Cycle period  object 0x1006:0
                                             'size 4, 10ms
SDOWR(fff,x1006,0,4,10000)        GOSUB10    'define Cycle period object 0x1006:0,
                                             'size 4, 10ms
SDOWR(mmm,x1005,0,4,128)          GOSUB10    'define Cycle ID x0000 0080 (required
                                             'to avoid error in next line)
SDOWR(mmm,x1005,0,4,1073741952)   GOSUB10    'define Cycle ID, producer x4000 0080
SDOWR(fff,x1005,0,4,128)          GOSUB10    'define Cycle ID, consumer x0000 0080
'Time sync
. . .

'Set up PDO mapping...
. . .

SDOWR(fff,x1600,1,4,570950432)    GOSUB10    'set Receive Map x2208 03 20
SDOWR(fff,x1600,2,4,571015184)    GOSUB10    'set Receive Map x2209 00 10 (follow
                                             'ctrl, 16-bit)
SDOWR(fff,x1600,0,1,2)            GOSUB10    'set Receive Map number of entries x02
SDOWR(fff,x1400,1,4,385)          GOSUB10    'set Receive Com Parameter  COB-ID
                                             'x0000 0181
'Set other objects in follow motor relating to follow mode.
SDOWR(fff,x2207,0,4,xffffffff)    GOSUB10    'set encoder modulo limit
aw[32]=0       'Follow control, map to outgo PDO, follow motor sees as
               '0x2209 control word.)
SDOWR(fff,x220A,0,2,100)          GOSUB10    'set MFMUL
SDOWR(fff,x220B,0,2,100)          GOSUB10    'set MFDIV
SDOWR(fff,x220C,0,4,0)            GOSUB10    'set MFA control word x2209 determines
                                             'if this is master or slave units
SDOWR(fff,x220D,0,4,10000)        GOSUB10    'set MFD  control word x2209 determines
                                             'if this is master or slave units
NMT(0,1)                          GOSUB10    'broadcast to network to go to
                                             'operational state
. . .
```

For the complete program, see the *SmartMotor Developer's Guide*.

Programming Examples

The following programming examples demonstrate the power and simplicity of I/O Device CAN Bus Master capability. Each uses a different method of polling to get data from a CANopen device.

Example 1: Timed SDO Poll

The Timed SDO Poll method can be used when high-speed polling is not required. The following code example uses SmartMotor timer interrupts to poll continuously and one-shot SDO commands to get data.

NOTE: For high-speed polling, a PDO is used to do this automatically, as shown in Example 2: I/O Block with PDO Poll on page 6.

```
'Using SmartMotor interrupts and timers to
'poll via SDO.
'Note PDO mapping is also available; this
'is simplified code.
CANCTL(17,3)             'ENABLE MASTER COMMANDS
ITR (3,4,3,0,300)        'WATCHDOG TMR INTERRUPT
                         '(poll CAN bus device)
EITR(3)                  'ENABLE INTERRUPT 3
ITRE                     'ENABLE ALL INTERRUPTS
TMR(3,30)                'Start timer 3 for 30 msec
PAUSE
END
'====================================================
C300 'CAN bus device has two data packets,
     'loaded into x and y.
  x=SDORD(1,24592,0,2)   'Read 2 bytes from
                         'address 1,object
                         '24592 (0x6179 hex); object
                         '0x6010, sub-index 0.
  e=CAN(4)               'Trap error codes if any.
  y=SDORD(1,24608,0,2)   'Read 2 bytes from
                         'address 1,object 24608
                         '(0x6020 hex); object
                         '0x6020, sub-index 0.
  ee=CAN(4)              'Trap error codes if any.
  IF (e|ee)==0           'Confirm the status of both
                         'SDO operations.
                         'Success if they are zero.
    'Variables "x" and "y" now contain
    'values from CAN device.
  ELSE
    'Place error handling code here.
  ENDIF
  TMR(3,30)              'Run timer 3 for 30 msec
                         'Data polled every 30 msec
RETURNI
```

Example 2: I/O Block with PDO Poll

This program communicates with a Softlink model RT133-3HF00-CAN I/O block. It uses high-speed PDO polling following the default mapping found in the .eds file. Also, the slave detects the baud rate.

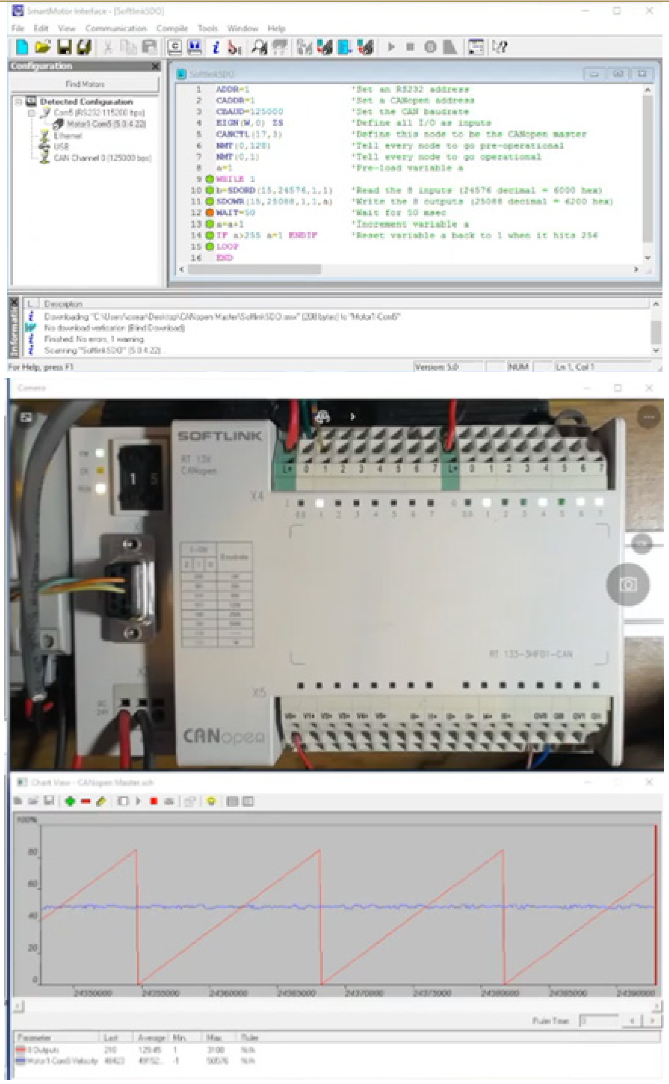
```
'NOTES: Using Softlink RT133-3HF00-CAN; using the
'default PDO mapping found in .eds file; Master
'address is 1, slave address is 15, slave detects
'baud rate.

ADDR=1
CADDR=1
CBAUD=125000
'RUN?
EIGN(W,0) ZS
SILENT

'Enable master
CANCTL(17,3)

'Make sure in pre-op (not operational)
NMT(0,128)

'Change analog output from +/-10V voltage(4) to
'4-20mA(5)
SDOWR(15,8192,13,1,5)

'Disable transmit and receive PDO for master to
'allow changing
'Set bit 31
a=-2147483648           '0x80000000
SDOWR(1,5120,1,4,a)     '0x1400 rx
SDOWR(1,6144,1,4,a)     '0x1800 tx
SDOWR(1,5121,1,4,a)     '0x1401 rx
SDOWR(1,6145,1,4,a)     '0x1801 tx

'Set mapping number of entries to 0
ab[0]=0
SDOWR(1,5632,0,1,ab[0]) '0x1600 rx
SDOWR(1,6656,0,1,ab[0]) '0x1a00 tx
SDOWR(1,5633,0,1,ab[0]) '0x1601 rx
SDOWR(1,6657,0,1,ab[0]) '0x1a01 tx

'Set mapping objects
'ab[2] 0x2220 03 08
a=572523272
SDOWR(1,5632,1,4,a)      '0x1600 rx
'ab[3] 0x2220 04 08
a=572523528
SDOWR(1,6656,1,4,a)      '0x1a00 tx
'aw[32] 0x2221 01 10     'aw[32] 1st analog voltage in
a=572588304
SDOWR(1,5633,1,4,a)      '0x1601 rx
'aw[33] 0x2221 02 10     'aw[33] 1st 4-20mA analog out
a=572588560
SDOWR(1,6657,1,4,a)      '0x1a01 tx
```

```
'Set mapping number of entries to 1
ab[0]=1
SDOWR(1,5632,0,1,ab[0])  '0x1600 rx
SDOWR(1,6656,0,1,ab[0])  '0x1a00 tx
SDOWR(1,5633,0,1,ab[0])  '0x1601 rx
SDOWR(1,6657,0,1,ab[0])  '0x1a01 tx

'Enable transmit and receive PDO for master to
'allow changing
'Clear bit 31
a=399                   '0x0000018f  (f = address 15)
SDOWR(1,5120,1,4,a)     '0x1400 rx mapped to slave tx

a=527                   '0x0000020f  (f = address 15)
SDOWR(1,6144,1,4,a)     '0x1800 tx mapped to slave rx

a=655                   '0x0000028f  (f = address 15)
SDOWR(1,5121,1,4,a)     '0x1401 rx mapped to slave tx

a=783                   '0x0000030f  (f = address 15)
SDOWR(1,6145,1,4,a)     '0x1801 tx mapped to slave rx

'Tell everyone to go operational
NMT(0,1)

b=1

WHILE 1
        IF B(10,1)==1
                Z(10,1)  'Clear event flag
                PRINT("Rx PDO 1",#13)
        ENDIF
        IF B(10,2)==1
                Z(10,2)  'Clear event flag
                PRINT("Rx PDO 2",#13)
        ENDIF
        IF B(10,3)==1
                Z(10,3)  'Clear event flag
                PRINT("Rx PDO 3",#13)
        ENDIF
        IF B(10,4)==1
                Z(10,4)  'Clear event flag
                PRINT("Rx PDO 4",#13)
        ENDIF
        IF B(10,5)==1
                Z(10,5)  'Clear event flag
                PRINT("Rx PDO 5",#13)
        ENDIF

'Set User Bits in Status Word 12 to reflect status of
'8 inputs
        UO(W,0)=ab[2]&255
'Turn on outputs (continuous count up)
        ab[3]=ab[3]+(1*b)
'Set User Bits in Status Word 13 to reflect status of
'8 outputs
        UO(W,1)=ab[3]&254
        WAIT=100
LOOP
END
```

Application Demos

In order to demonstrate the features of I/O Device CAN Bus Master capability, the following application demos were created.

Simple and Inexpensive I/O Device Control



```
ADDR=1                      'Set an RS232 address
CADDR=1                     'Set a CANopen address
CBAUD=125000                'Set the CAN baudrate
EIGN(W,0) IS                'Define all I/O as inputs
CANCTL(17,3)                'Define this node to be the CANopen master
NMT(0,128)                  'Tell every node to go pre-operational
NMT(0,1)                    'Tell every node to go operational
a=1                         'Pre-load variable a
WHILE 1
b=SDORD(15,24576,1,1)       'Read the 8 inputs (24576 decimal = 6000 hex)
SDOWR(15,25088,1,1,a)       'Write the 8 outputs (25088 decimal = 6200 hex)
WAIT=50                     'Wait for 50 msec
a=a+1                       'Increment variable a
IF a>255 a=1 ENDIF          'Reset variable a back to 1 when it hits 254
LOOP
END
```

```
Downloading "C:\Users\cosa\Desktop\CANopen Master\SoftlinkSDO.smx" (208 bytes) to "Motor1-Com5"
No download verification (Blind Download)
Finished. No errors, 1 warning
Scanning "SoftlinkSDO" (5.0.4.22)...
```

Figure 3: Simple I/O Device Control
(Click Image to View Video)

The demo in the previous figure illustrates a simple and inexpensive implementation of I/O Device CAN Bus Master capability. The use of CANopen to master I/O modules allows I/O expansion for the SmartMotor that was previously unavailable. Even machines with large I/O requirements can now be handled by a single SmartMotor.

This demo uses one SDO Read and one SDO Write to read the inputs and set the outputs of a Softlink CANopen I/O module. Two lines of code run in a continuous loop to nearly double the I/O available to the SmartMotor. The I/O can be expanded further using the same commands with different addresses to access multiple I/O modules. In addition to digital I/O, CANopen modules like the Softlink can add analog I/O at a low cost.

Six-Axis Coordinated Motion with I/O Control

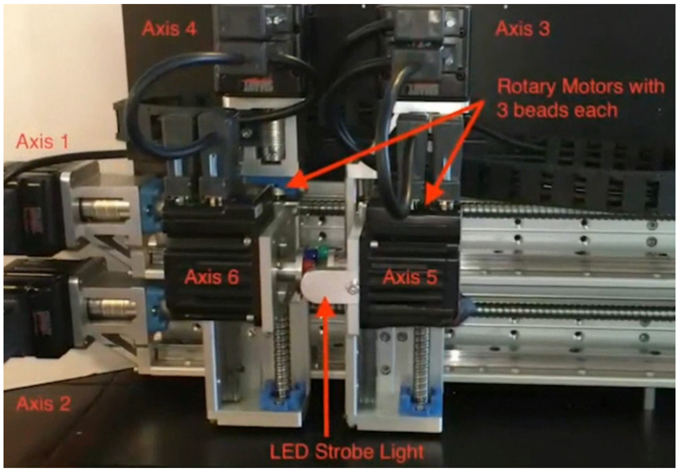


Figure 4: Six-Axis Coordinated Motion with I/O Control (Click Image to View Video)

The demo in the previous figure shows how the I/O Device CAN Bus Master capability and Combitronic technology are used to control six axes of synchronized motion. Additionally, it controls an LED strobe through high-speed programmable interrupts triggered by modulo wrap, positive software limit and cam user status bits.

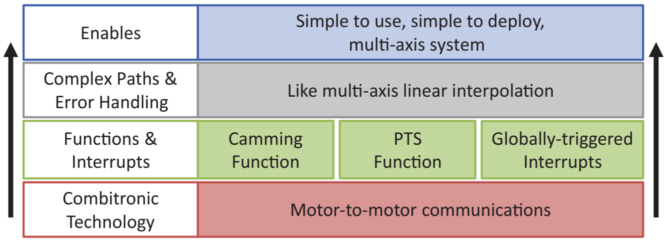
One rotary motor is electronically geared off the main rotary motor while running in dual-trajectory mode. The second rotary motor performs positional moves on top of electronic gearing. Additionally, the system allows for microsecond-level control of the strobe.

This demo highlights the following key benefits:

- All programming and control is accomplished through only the SmartMotors—no external bus master is used or needed.
- All communication is performed over the CAN bus for minimal cabling and ease of integration.
- Time sync of motors, electronic gearing, camming and camming on top of gearing—all done through the CAN bus.
- High-speed interrupt applications, such as high-speed inspection systems, are enabled.

Conclusion

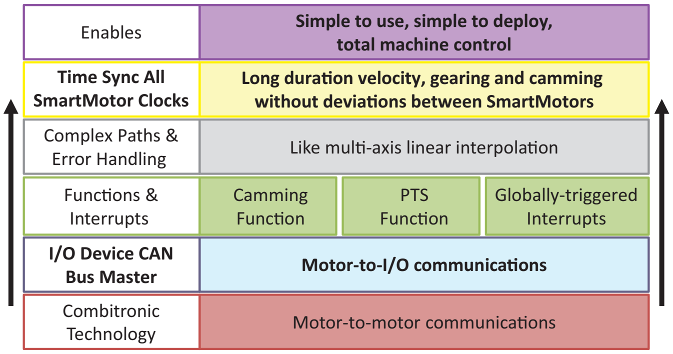
By itself, Combitronic technology provides the foundation for easy and powerful machine design.



Enables	Simple to use, simple to deploy, multi-axis system		
Complex Paths & Error Handling	Like multi-axis linear interpolation		
Functions & Interrupts	Camming Function	PTS Function	Globally-triggered Interrupts
Combitronic Technology	Motor-to-motor communications		

Figure 5: Combitronic Technology, the Foundation to Easy and Powerful Machine Design

By adding I/O Device CAN Bus Master capability on top of Combitronic technology, the machine-control and communications abilities of the SmartMotor are significantly expanded.



Enables	**Simple to use, simple to deploy, total machine control**		
Time Sync All SmartMotor Clocks	**Long duration velocity, gearing and camming without deviations between SmartMotors**		
Complex Paths & Error Handling	Like multi-axis linear interpolation		
Functions & Interrupts	Camming Function	PTS Function	Globally-triggered Interrupts
I/O Device CAN Bus Master	**Motor-to-I/O communications**		
Combitronic Technology	Motor-to-motor communications		

Figure 6: I/O CAN Bus Master Expands the Capabilities of Combitronic Technology

I/O Device CAN Bus Master capability represents an evolutionary step forward in integrated servo motor design. It aids the machine builder by:

- Simplifying motor-to-motor and motor-to-I/O communications,
- Enabling efficient, precise, multi-axis motion control,
- Allowing total machine control with no external master.

These advanced capabilities help machine builders streamline the machine build process and:

- Reduce time to market,
- Minimize the machine footprint,
- Lower machine build and replication cost,
- Improve machine reliability.

I/O Device CAN Bus Master capability combined with Combitronic technology expands the communications, processing and control power of the networked SmartMotors to provide a simple, economical and effective total-machine-control solution.

References

[1] For a brief overview of Combitronic technology, see How Does It Work? on page 2. For a full description of Combitronic technology, see the white paper Understanding Combitronic Technology.

[2] For more CANopen terms and information, see the *Class 5 SmartMotor CANopen Guide*.

[3] Cam mode operation requires user program to configure the cam. For details, see the *SmartMotor Developer's Guide*.

[4] Requires SmartMotor firmware version 5.0.4.31 and later, or 5.98.4.31 and later.

[5] Network bandwidth is limited by the base network protocol selected: CANopen can achieve up to 1 MHz; DeviceNet can achieve up to 0.5 MHz.

Figure 1	Combitronic Motor-to-Motor Communications provided by Moog.
Figure 2	Combitronic with I/O Device CAN Bus Master Capability provided by Moog.
Figure 3	Simple I/O Device Control provided by Moog.
Figure 4	Six-Axis Coordinated Motion with I/O Control provided by Moog.
Figure 5	Combitronic Technology, the Foundation to Easy and Powerful Machine Design provided by Moog.
Figure 6	I/O CAN Bus Master Expands the Capabilities of Combitronic Technology provided by Moog.

About Moog Animatics

Since 1987, Moog Animatics has been designing, manufacturing and marketing motion control products. We bring total automation solutions to numerous industries, including semiconductor, defense, automotive, aerospace, biomedical, textile, security, marine sciences, packaging and many more.

When you need an innovative solution, you need Moog Animatics. We pride ourselves on offering the most creative and complete answers to your motion control questions.

The Moog Animatics headquarters is located in the heart of Silicon Valley, with international offices in Germany and Japan, and a vast network of Moog Animatics-trained Automation Solution Providers around the world.

For more information on how the SmartMotor integrated servo with Combitronic technology and I/O Device CAN Bus Master capability can benefit your application, please call 650.960.4215 or email us at animatics_sales@moog.com.

For product information, visit **www.animatics.com**

For more information or the office nearest you, contact us online, **animatics_sales@moog.com**

MOOG ANIMATICS