INTELLIGENT MOTION SYSTEMS, INC.

*Excellence in Motion*™

# CANopen

# DSP-402 APPLICATION GUIDE

MDRIVE™ MOTOR+DRIVER *Plus* CANopen

MFORCE™ DRIVE *Plus* CANopen

| MDrivePlus CANopen Change Log | | |
|---|---|---|
| Date | Revision | Changes |
| 10/06/2006 | R100606 | Initial Release |
| 12/20/2006 | R122006 | Added newly supported objects to DSP-402 Implementation |
| 02/05/2007 | R020507 | Added new object support, broke out DSP 402 to separate document. |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |



**CANopen DSP402 Application**

**Revision R020507**

# Contents

## List of Figures



## List of Tables

This Page Intentionally Left Blank

INTELLIGENT MOTION SYSTEMS, INC.
Excellence in Motion™

# CANopen
# DSP-402 APPLICATION GUIDE

*Page Intentionally Left Blank*

# SECTION 1

## Introduction to the MDrivePlus CANopen DSP-402 Implementation

### Introduction

This document describes the Operational Modes and Objects utilized by the MDrivePlus CANopen. The MDrivePlus uses the CiA DSP402 protocol as described the the CiA document *CANopen Device Profile for Drives and Motion Control V2.0B.*

### CAN Message Format

The MDrivePlus is compliant with CAN 2.0B Active Specification. The Data Packets follow the message format shown in Figure 1.1. The Figure is for reference only, please refer to the CAN 2.0B Specification.

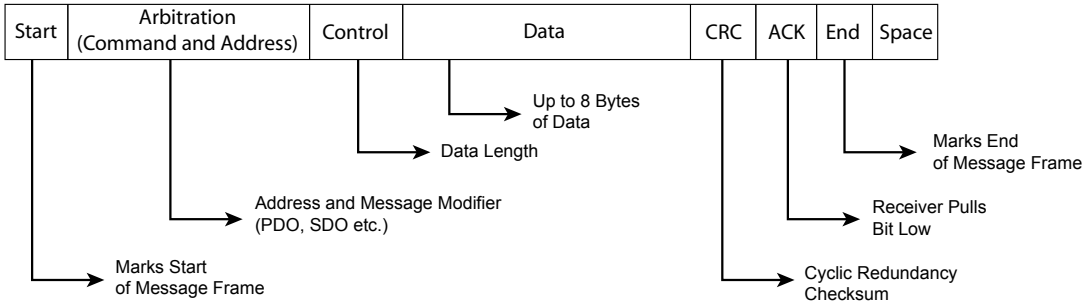



*Figure 1.1: Message Format*

### MDrivePlus Architecture

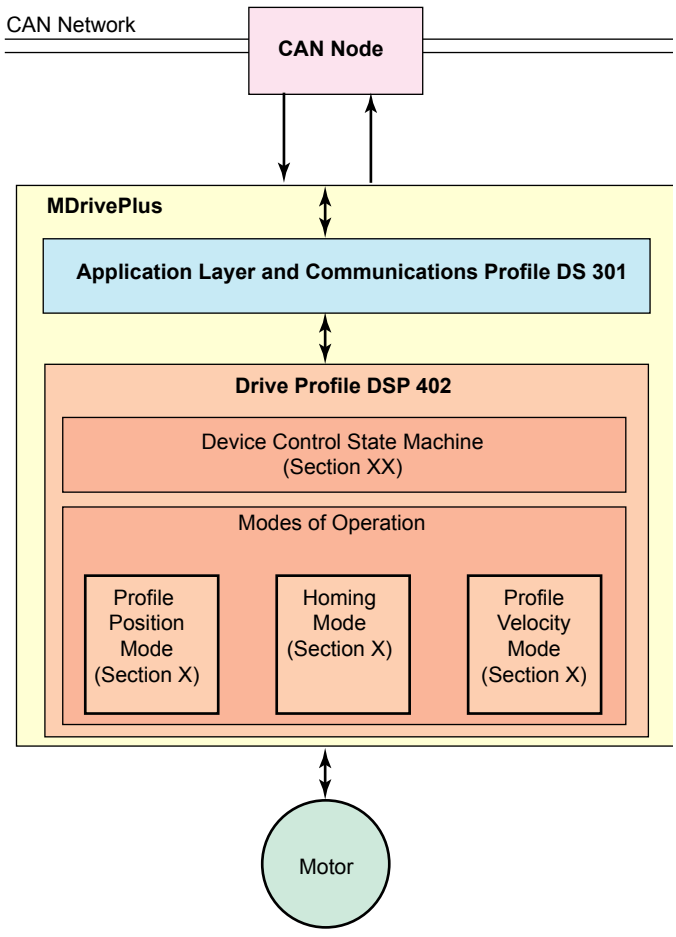



*Figure 1.2: MDrivePlus Architecture*

## Device Control

The starting and stopping of the drive and several mode specific commands are executed by the state machine.

## Modes of Operation

The operation mode defines the behavior of the drive. The following modes are defined in this profile:

### *Homing Mode*

This chapter describes the various methods to find a home position (also: reference point, datum, zero point).

### *Profile Position Mode*

The positioning of the drive is defined in this mode. Speed, position and acceleration can be limited and profiled moves using a Trajectory Generator are possible as well.

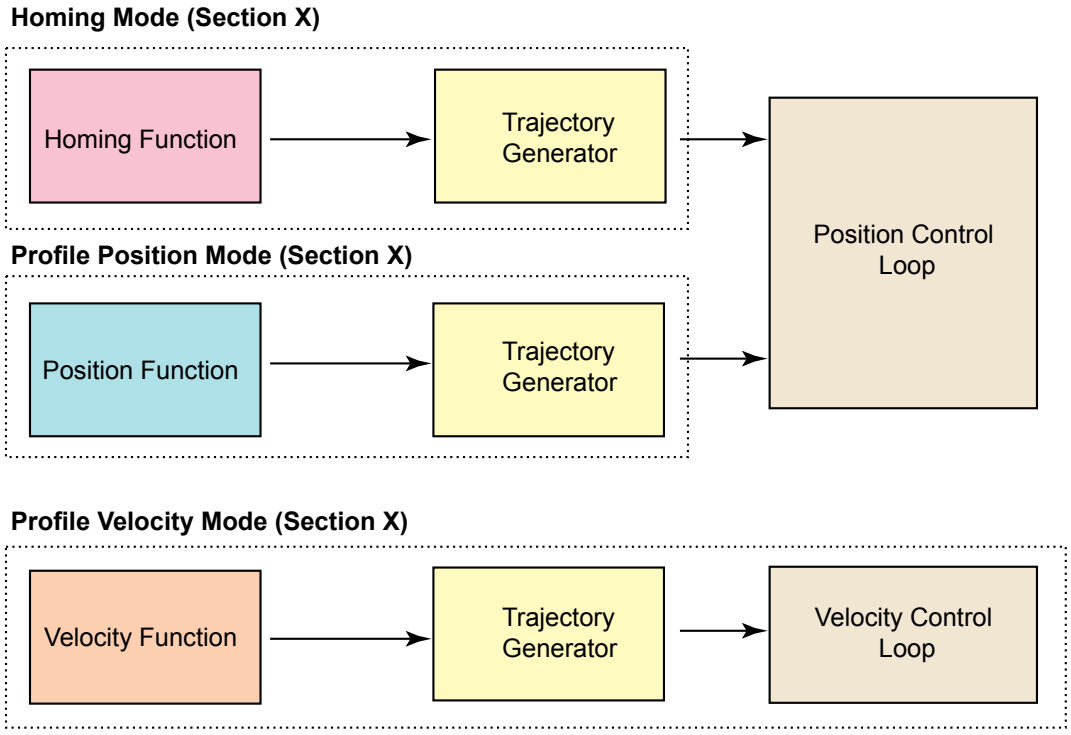



*Figure 1.3: Functional Architecture*

### *Profile Velocity Mode*

The Profile Velocity Mode is used to control the velocity of the drive with no special regard of the position. It supplies limit functions and trajectory generation.

### *Trajectory Generator*

The chosen operation mode and the corresponding parameters (objects) define the input of the Trajectory Generator. The Trajectory Generator supplies the control loop(s) with the demand values. They are generally mode specific.

Each Mode may use its own Trajectory Generator. A general description of its functionality is given in Section X, which is related to the Profile Position Mode.

## Objects and the Object Dictionary

In a CANopen network, a device is controlled by writing to device parameters and reading the status of the device. This is accomplished using a pre-defined dictionary of instructions that can be written and status information that can be read. These pieces of information are called Objects.

The full set of objects are called the Object Dictionary. The Object Dictionary is the interface between the CANopen master, or controller and the MDrivePlus node on a CANopen network.

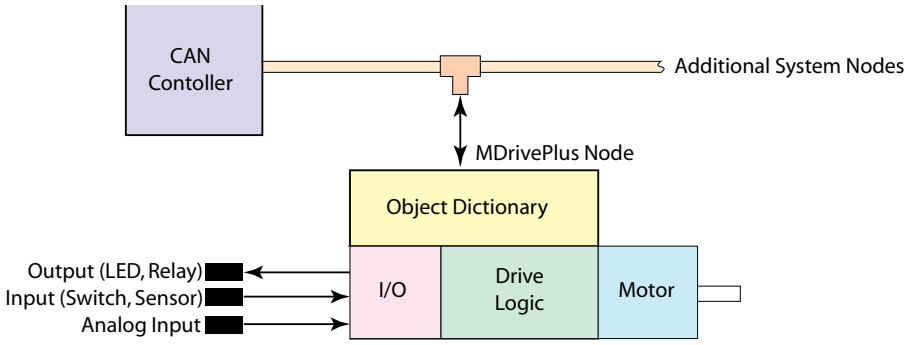



*Figure 1.4: MDrivePlus CANopen Object Dictionary*

Entries within the Object Dictionary are addressed using 16-bit Indexes. In the case of simple variables (VAR) the index references the value of the variable directly. In the case of records and arrays the index addresses the entire data structure.

To allow individual elements of the data structures a sub-index is defined. The fields accessed by the sub-index may be of differing data types.

| Index (hex) | Object |
|---|---|
| 0000 | Not Used |
| 0001 – 004F | Static Data Types |
| 0020 – 003F | Complex Data Types |
| 0040 – 005F | Manufacturer Specific Data |
| 0060 – 007F | Device Profile Specific Static Data Types |
| 0080 – 009F | Device Profile Specific Complex Data Types |
| 00A0 – 0FFF | Reserved for Future Use |
| 1000 – 1FFF | Communications Profile Area |
| 2000 – 5FFF | Manufacturer Specific Profile |
| 6000 – 9FFF | Standardized Device Profile |
| A000 – BFFF | Standardized Interface Profile |
| C000 – FFFF | Reserved for Future Use |

*Table 1.1: Object Dictionary*

## Object Formatting

This manual will display the Object and Entry data using the model detailed below.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$XXXX_h$** | **Index Name** | **VAR** | **I/U** |

#### Index

The Index is the hexadecimal number that represents the index number of the object in the CANopen Object Dictionary. With the exception of IMS Manufacturer specific objects these are defined in CiA Device Profile for Drives and Controls DSP402. The applicable objects are defined in this document as well.

#### Index Name

The Index Name is the general name and description of the object. With the exception of IMS Manufacturer specific objects these are defined in CiA Device Profile for Drives and Controls DSP402.

#### Object Code

VAR - Variable

#### Data Type

Physically, the types consist of one or more bytes. One byte consists of 8 bits (Bit 0 to 7). Bit 0 is the LSB (Least Significant Bit). A byte can also be depicted hexadecimally (0x00 ... 0xff).

If a data type consists of n byte, the following applies:

Data byte 1 (Byte in address x) = highest value byte

Data byte n (Byte in address x+n-1) = lowest value byte

The data coding and the value ranges for the respective data types apply, unless otherwise explicitly stated in the data description of an MDrivePlus communication object.

| Integer (I) | Range | Length |
|---|---|---|
| **± Integer 8** | –128 ... +127 | 1 Byte |
| **± Integer 16** | – 32,768 ... +32,767 | 2 Bytes |
| **± Integer 32** | – 2,147,483,647 ... +2,147,483,647 | 4 Bytes |

| | |
|---|---|
| **Coding** | 2's Complement |

| Unsigned (U) | Range | Length |
|---|---|---|
| **Unsigned 8** | 0 ... 255 | 1 Byte |
| **Unsigned 16** | 0 ... 65,535 | 2 Bytes |
| **Unsigned 32** | 0 ... 4,294,967,295 | 4 Bytes |

| | |
|---|---|
| **Coding** | Binary |

### *Entry Description*

| Access | PDO Mapping | Category | Range | Default |
|---|---|---|---|---|
| **R/W/S/K** | **Yes/No** | **M/O** | **I/U** | **I/U** |

#### Access

R.....Read Access
W.....Write Access
S ..... Storable to Non Volatile Memory (NVM)
K..... Key Required for Write Access

### PDO Mapping

Describes whether (Yes) or not (No) the Index may be mapped to a PDO (Process Data Object). If yes it may be mapped to a PDO, if No the Object must be accessed using SDO (Service Data Objects).

### Category

M.....Mandatory
O .....Optional

### Range

The range of the Index will be expressed as a ± Integer or Unsigned.

### Default

The range of the Index will be expressed as a ± Integer or Unsigned.

## *Sub-Indexes*

An object may have a number of Sub-Indexes which further define the operation of the object, such as I/O Configuration Parameters.

Sub-Indexes are formatted thus:

### Sub-Index X

| | |
|---|---|
| **Description** | Sub-Index Functional Description |
| **Entry Category** | Mandatory/Optional |
| **Access** | R/W/S/K |
| **PDO Mapping** | Yes/No |
| **Value Range** | 1 Byte Hex |
| **Default Value** | 1 Byte Hex |

*Page Intentionally Left Blank*

## Introduction

The access from the CAN network to the drive is done through data objects.

## Process Data Object (PDO)

PDOs are messages in an unconfirmed service. They are used for the transfer of real-time data to and from the drive. The transfer is fast, because it is performed with no protocol overhead what means to transport eight application data bytes in one CAN-frame. The PDOs correspond to entries in the Object dictionary.

### PDO Attributes

1. Two Types: RPDO (Receive) and TPDO (Transmit)
2. Up to 8 Bytes of application data per message frame. No additional protocol overhead is required.
3. Transfer is not confirmed
4. PDOs Require setup, SDOs map each byte of the PDO to one or more Object Entries.
5. PDOs operate using the Producer (TPDO)/Consumer (RPDO) relationship Push-Pull model.
6. Best for transferring data such as Device Status, Set-points etc.

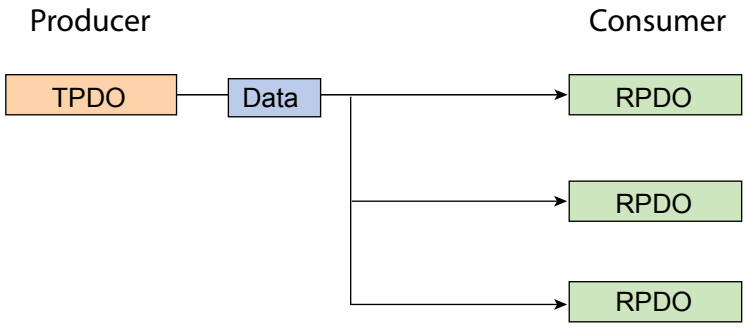



*Figure 2.1: PDO Producer – Consumer Relationship*

## Service Data Object (SDO)

A Service Data Object (SDO) reads from entries or writes to entries of the Object Dictionary. The SDO transport protocol allows transmitting objects of any size. The first byte of the first segment contains the necessary flow control information including a toggle bit to overcome the well-known problem of doubly received CAN frames. The next three byte of the first segment contain index and sub-index of the Object Dictionary entry to be read or written. The last four byte of the first segment are available for user data. The second and the following segments (using the very same CAN identifier) contain the control byte and up to seven byte of user data. The receiver confirms each segment or a block of segments, so that a peer-to-peer communication (client/server) takes place.

### SDO Attributes

1. Can access any Object in the Object Dictionary regardless of size.
2. Transfer is confirmed
3. Direct access to the Object Dictionary
4. Client/Server relationship.
5. Best for setting up configuration parameters.

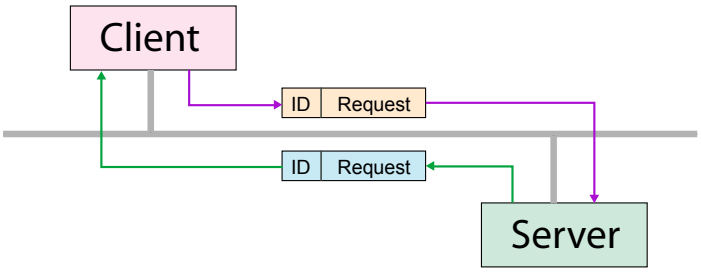



*Figure 2.2: SDO Client – Server Relationship*

## PDO Mapping

The MDrivePlus CANopen allows you to map objects to PDOs to reduce the transfer application data more efficiently. By using the PDO the user can map a PDO to multiple objects (8 Data Bytes max.)

The example will show RPDO $1400_h$ mapped to Control Word ($6040_h$) and Target Position ($607A_h$).

| RPDO Index | Sub-Index | Mapped To Index | Bytes |
|---|---|---|---|
| $1600_h$ | $00_h$ | | |
| $1600_h$ | $01_h$ | $6040_h$ | 2 |
| $1600_h$ | $02_h$ | $607A_h$ | 4 |

## PDO Mapping Procedure (Consumer PDO)

| PDO Mapping Example 1: Profile Position Mode – Mapping ControlWord and Target Position to RPDO1 | | | | | |
|---|---|---|---|---|---|
| **Step** | Action | Index | Sub-Index | Bytes | Value |
| **1** | Place MDrive in PreOperational State | | | — | |
| **2** | Turn Off RPDO1 | $1400_h$ | 01 | — | $8000\ 01C0_h$ |
| **3** | Set $1600_h$ Sub-Index 00 to 0 | $1600_h$ | 00 | — | $0_h$ |
| **4** | Map ControlWord $6040_h$ to $1600.01_h$, Establish New Set Point | $6040_h$ | 00 | 2 | $005F_h$ |
| **5** | Map target_position $607A_h$ to $1600.02_h$ | $607A_h$ | | 4 | Desired Axis Position in Hex |
| **6** | Set $1600_h$.00 to 2 Max Sub-Indexes | $1600_h$ | 00 | — | $2_h$ |
| **7** | Turn On RPDO1 | $1400_h$ | 00 | — | $0000\ 01C0_h$ |
| **8** | Place MDrive in Profile Position Mode | $6060_h$ | 00 | 1 | $1_h$ |
| **9** | Place MDrive in Operational State | | | | |
| **10** | Send PDO to MDrive | | | | |
| **Note: Before re-sending the PDO to the MDrive, the old set-point must be cleared by sending $6040.00_h$ $004F_h$ in a second PDO or in an SDO.** | | | | | |

### Default Mapping Example - Consumer PDO 2

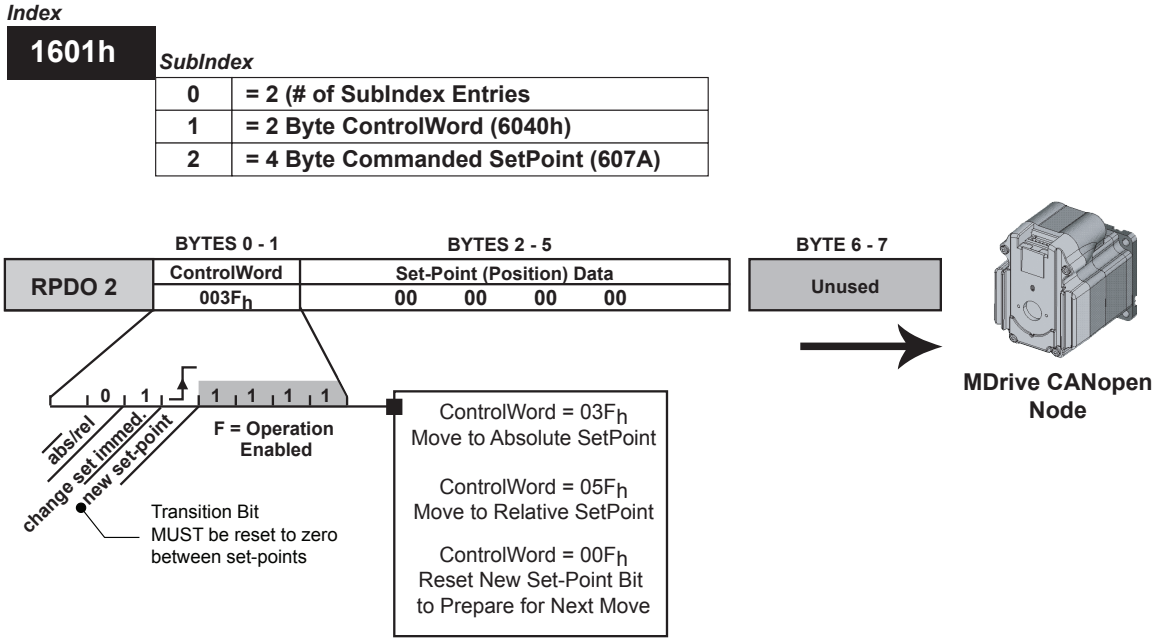



*Figure 2.3: PDO Mapping Showing the Default Mapping for RPDO2*

## PDO Objects

### Consumer PDO1 (RPDO1) 1400h (Object Description)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1400_h$ | Receive PDO1 Parameter | Record | PDO Communications Parameters | Mandatory |

### Consumer PDO1 (RPDO1) 1400h (Entry Description)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | Mfg. Specific |
| **01h** | COB-ID used by PDO | Mandatory | R/W | | $0000\ 0200_h$ + NODE ID |
| **02h** | Transmission Type | Mandatory | R/W | | $255_d$ |
| **03h** | Inhibit Time | Optional | R/W | | |
| **04h** | **Reserved** | | | | |
| **05h** | Event Timer | Optional | R/W | | $0_d$ |

### 1600h (Object Description – Mapping Parameters)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1600_h$ | Receive PDO1 Mapping | Record | PDO Mapping | Mandatory |

### 1600h (Entry Description – Mapping Parameters)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $01_h$ |
| **01h** | 1st Application Object | Mandatory | R/W | | $6040\ 0010_h$ |
| **02h** | 2nd Application Object | Mandatory | R/W | | Mfg. Specific |
| **03h** | **3rd** Application Object | Mandatory | R/W | | Mfg. Specific |
| **04h** | **4th** Application Object | Mandatory | R/W | | Mfg. Specific |
| **05h** | **5th** Application Object | Mandatory | R/W | | Mfg. Specific |
| **06h** | **6th** Application Object | Mandatory | R/W | | Mfg. Specific |
| **07h** | **7th** Application Object | Mandatory | R/W | | Mfg. Specific |
| **08h** | **8th** Application Object | Mandatory | R/W | | Mfg. Specific |

### Consumer PDO2 (RPDO2) 1401h (Object Description)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1401_h$ | Receive PDO2 Parameter | Record | PDO Communications Parameters | Optional |

### Consumer PDO2 (RPDO2) 1401h (Entry Description)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $02_h$ |
| **01h** | COB-ID used by PDO | Mandatory | R/W | | $0000\ 0300_h$ or $8000\ 0000_h$ + NODE ID |
| **02h** | Transmission Type | Mandatory | R/W | | $255_d$ |
| **03h** | Inhibit Time | Optional | R/W | | |
| **04h** | **Reserved** | | | | |
| **05h** | Event Timer | Optional | R/W | | $0_d$ |

### 1601h (Object Description – Mapping Parameters)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1601_h$ | Receive PDO2 Mapping | Record | PDO Mapping | Conditional, if $1401_h$ is implemented |

### 1601h (Entry Description – Mapping Parameters)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $01_h$ |
| **01h** | 1st Application Object | Mandatory | R/W | | $6040\ 0010_h$ |
| **02h** | 2nd Application Object | Optional | R/W | | $6060\ 0008_h$ |
| **03h** | **3rd** Application Object | Optional | R/W | | Mfg. Specific |
| **04h** | **4th** Application Object | Optional | R/W | | Mfg. Specific |
| **05h** | **5th** Application Object | Optional | R/W | | Mfg. Specific |
| **06h** | **6th** Application Object | Optional | R/W | | Mfg. Specific |
| **07h** | **7th** Application Object | Optional | R/W | | Mfg. Specific |
| **08h** | **8th** Application Object | Optional | R/W | | Mfg. Specific |

### Consumer PDO3 (RPDO3) 1402h (Object Description)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1402_h$ | Receive PDO3 Parameter | Record | PDO Communications Parameters | Optional |

### Consumer PDO3 (RPDO3) 1402h (Entry Description)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $02_h$ |
| **01h** | COB-ID used by PDO | Mandatory | R/W | | $0000\ 0400_h$ or $8000\ 0000_h$ + NODE ID |
| **02h** | Transmission Type | Mandatory | R/W | | $255_d$ |
| **03h** | Inhibit Time | Optional | R/W | | |
| **04h** | **Reserved** | | | | |
| **05h** | Event Timer | Optional | R/W | | $0_d$ |

### 1602h (Object Description – Mapping Parameters)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1601_h$ | Receive PDO3 Mapping | Record | PDO Mapping | Conditional, if $1402_h$ is implemented |

### 1602h (Entry Description – Mapping Parameters)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $01_h$ |
| **01h** | 1st Application Object | Mandatory | R/W | | $6040\ 0010_h$ |
| **02h** | 2nd Application Object | Optional | R/W | | $607A\ 0020_h$ |
| **03h** | **3rd** Application Object | Optional | R/W | | Mfg. Specific |
| **04h** | **4th** Application Object | Optional | R/W | | Mfg. Specific |
| **05h** | **5th** Application Object | Optional | R/W | | Mfg. Specific |
| **06h** | **6th** Application Object | Optional | R/W | | Mfg. Specific |
| **07h** | **7th** Application Object | Optional | R/W | | Mfg. Specific |
| **08h** | **8th** Application Object | Optional | R/W | | Mfg. Specific |

### Producer PDO1 (TPDO1) 1800h (Object Description)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1800_h$ | Transmit PDO1 Parameter | Record | PDO Communications Parameters | Optional |

### Producer PDO1 (TPDO1) 1800h (Entry Description)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R | | $02_h$ |
| **01h** | COB-ID used by PDO | Mandatory | R/W | | $4000\ 0180_h$ + NODE ID |
| **02h** | Transmission Type | Mandatory | R/W | | $255_d$ |
| **03h** | Inhibit Time | Optional | R/W | | |
| **04h** | **Reserved** | | | | |
| **05h** | Event Timer | Optional | R/W | | $0_d$ |

### 1A00h (Object Description – Mapping Parameters)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1A00_h$ | Transmit PDO1 Mapping | Record | PDO Mapping | Mandatory |

### 1A00h (Entry Description – Mapping Parameters)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $01_h$ |
| **01h** | 1st Application Object | Mandatory | R/W | | $6041\ 0010_h$ |
| **02h** | 2nd Application Object | Optional | R/W | | Mfg. Specific |
| **03h** | **3rd** Application Object | Optional | R/W | | Mfg. Specific |
| **04h** | **4th** Application Object | Optional | R/W | | Mfg. Specific |
| **05h** | **5th** Application Object | Optional | R/W | | Mfg. Specific |
| **06h** | **6th** Application Object | Optional | R/W | | Mfg. Specific |
| **07h** | **7th** Application Object | Optional | R/W | | Mfg. Specific |
| **08h** | **8th** Application Object | Optional | R/W | | Mfg. Specific |

### Producer PDO2 (TPDO2) 1801h (Object Description)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1801_h$ | Transmit PDO2 Parameter | Record | PDO Communications Parameters | Optional |

### Producer PDO2 (TPDO2) 1801h (Entry Description)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R | | — |
| **01h** | COB-ID used by PDO | Mandatory | R/W | | $4000\ 0280_h$ or $C000\ 0280_h$ + NODE ID |
| **02h** | Transmission Type | Mandatory | R/W | | $255_d$ |
| **03h** | Inhibit Time | Optional | R/W | | $0_d$ |
| **04h** | **Reserved** | | | | |
| **05h** | Event Timer | Optional | R/W | | $0_d$ |

### 1A01h (Object Description – Mapping Parameters)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1A01_h$ | Transmit PDO2 Mapping | Record | PDO Mapping | Conditional if $1801_h$ is implemented |

### 1A01h (Entry Description – Mapping Parameters)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $02_h$ |
| **01h** | 1st Application Object | Mandatory | R/W | | $6041\ 0010_h$ |
| **02h** | 2nd Application Object | Optional | R/W | | $6061\ 0008_h$ |
| **03h** | **3rd** Application Object | Optional | R/W | | Mfg. Specific |
| **04h** | **4th** Application Object | Optional | R/W | | Mfg. Specific |
| **05h** | **5th** Application Object | Optional | R/W | | Mfg. Specific |
| **06h** | **6th** Application Object | Optional | R/W | | Mfg. Specific |
| **07h** | **7th** Application Object | Optional | R/W | | Mfg. Specific |
| **08h** | **8th** Application Object | Optional | R/W | | Mfg. Specific |

### Producer PDO3 (TPDO3) 1802h (Object Description)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1802_h$ | Transmit PDO3 Parameter | Record | PDO Communications Parameters | Optional |

### Producer PDO3 (TPDO3) 1802h (Entry Description)

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R | | — |
| **01h** | COB-ID used by PDO | Mandatory | R/W | | $4000\ 0380_h$ or $C000\ 0380_h$ + NODE ID |
| **02h** | Transmission Type | Mandatory | R/W | | $255_d$ |
| **03h** | Inhibit Time | Optional | R/W | | $0_d$ |
| **04h** | **Reserved** | | | | |
| **05h** | Event Timer | Optional | R/W | | $0_d$ |

### 1A02h (Object Description – Mapping Parameters)

| Index | Name | Object Code | Data Type | Category |
|---|---|---|---|---|
| $1A02_h$ | Transmit PDO3 Mapping | Record | PDO Mapping | Conditional if $1802_h$ is implemented |

### *1A02h (Entry Description – Mapping Parameters)*

| Sub-Index | Description | Category | Access | Value Range | Default |
|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | R/W | | $02_h$ |
| **01h** | 1st Application Object | Mandatory | R/W | | 6041 $0010_h$ |
| **02h** | 2nd Application Object | Optional | R/W | | 6064 $0010_h$ |
| **03h** | **3rd** Application Object | Optional | R/W | | Mfg. Specific |
| **04h** | **4th** Application Object | Optional | R/W | | Mfg. Specific |
| **05h** | **5th** Application Object | Optional | R/W | | Mfg. Specific |
| **06h** | **6th** Application Object | Optional | R/W | | Mfg. Specific |
| **07h** | **7th** Application Object | Optional | R/W | | Mfg. Specific |
| **08h** | **8th** Application Object | Optional | R/W | | Mfg. Specific |

## Introduction

The objects detailed in this section are IMS manufacturer specific configuration objects to conFigure :

- I/O Type
- Run/Hold Current
- Factory Configuration

### Accessibility Codes

R — Read

W — Write

S — Storable to Nonvolatile Memory (NVM)

K — Key Required

## Object 2000h: I/O Discretes (Config)

### Object Description

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2000_h$** | **I/O Discretes** | **VAR** | **Unsigned 8** |

### Entry Description

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | ConFigure I/O as Output | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Output, 0 = Input) |
| **02h** | ConFigure I/O as Sourcing | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Sourcing Only, 0 = Sinking Only) |
| **03h** | ConFigure I/O as Both | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Both Source and Sink, 0 = See Sub-Index 2) |
| **04h** | ConFigure I/O as Polarity In | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Invert Polarity of Digital Inputs, 0 = See Index 60FDh Sub-Index 1 Bits <23...16>) |
| **05h** | ConFigure I/O as Polarity Out | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Invert Polarity of Digital I/O, 0 = See Index 60FEh Sub-Index 1 Bits <23...16>) |

## Object 2002h: I/O Discretes (Config)

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| $2002_h$ | Config Input Switches | VAR | Unsigned 8 |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | ConFigure I/O as Home | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Selects I/O# as the Home Switch) |
| **02h** | ConFigure I/O as Positive Limit | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Selects I/O# as the Positive Limit) |
| **03h** | ConFigure I/O as Negative Limit | Mandatory | R/W | No | 0x00 — 0xFF | 0x00 (1 = Selects I/O# as the Negative Limit) |

## Object 2004h: Input Filter Mask (Config)

The Input Filter Mask Object conFigure s the device to filter the selected inputs. The operation of the Object is shown in Figure 3.1 below.

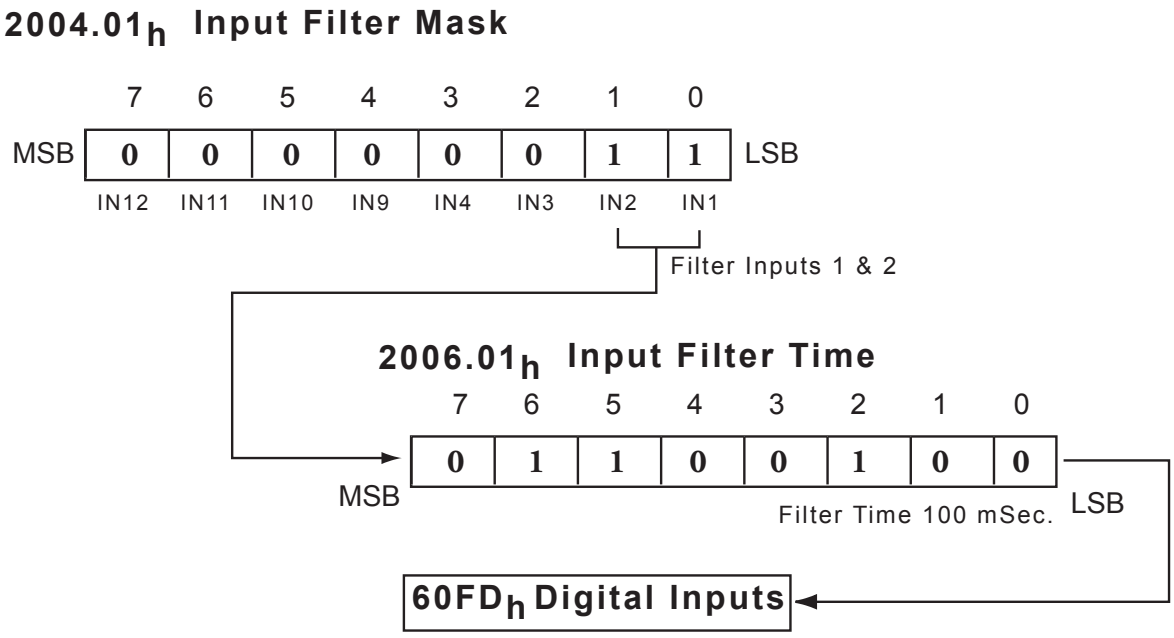



*Figure 3.1: Input Filter Mask*

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2004_h$** | **Input Filter Mask** | **VAR** | **Unsigned 8** |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $01_h$ |
| **02h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $02_h$ |
| **03h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $04_h$ |
| **04h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $08_h$ |
| **05h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $10_h$ |
| **06h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $20_h$ |
| **07h** | Input Filter Mask | Optional | R/W | No | $00_h$ – $FF_h$ | $40_h$ |

## Object 2006h: Input Filter Time (ms)

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2006_h$** | **Input Filter Time** | **VAR** | **Unsigned 8** |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **02h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **03h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **04h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **05h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **06h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **07h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |
| **08h** | Input Filter Time | Optional | R/W | No | 0 – 250 ms | 0 |

## Object 2010h: Analog Input

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| $2010_h$ | Analog Input | VAR | See Entry Desc. |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default | Data Type |
|---|---|---|---|---|---|---|---|
| **01h** | Analog Reading | Mandatory | R/W | Yes | 0 - 1023 | 0 | Unsigned 16 |
| **02h** | **Analog Input Configuration** | Mandatory | R/W | No | 0=5V Scale 8=10V Scale 2=20mA Scale | 0 | Unsigned 8 |
| **03h** | **Input Filtering** | Mandatory | R/W | No | 0 — 31 | 0=No Filtering | Unsigned 8 |

## Object 2020h: Software Limits as Hardware Limits

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| $2020_h$ | Software Limits as Hardware | VAR | Unsigned 8 |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | Limit Reached Flag | Optional | R/W | No | $00_h$ _ $FF_h$ | 0 |
| **02h** | **Limit Reached Mask** | Optional | R/W | No | $00_h$ _ $FF_h$ | 0 |

## Object 2022h: Actual Position Software Limit

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| $2022_h$ | Actual Position Software Limit | VAR | Signed 32 |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | Actual Negative Limit | Optional | R/W | No | Full 32 Bit | $80000000_h$ |
| **02h** | **Actual Positive Limit** | Optional | R/W | No | Full 32 Bit | $7FFFFFFF_h$ |

## Object 2031h: Unit Options (Encoder Enable, Capture In/Trip Out)

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2031_h$** | **Unit Options** | **VAR** | **Unsigned 8** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/W/S** | **No** | **0/1** | **0** |

Note: Encoder functions only apply to the MDrive products. The MForce products do not have closed loop capability.

| Bit Position | 0x80 | 0x40 | 0x20 | 0x10 | 0x08 | 0x04 | 0x02 | 0x01 |
|---|---|---|---|---|---|---|---|---|
| 0 | Reserved | Reserved | Reserved | Reserved | Encoder Disabled | Capture In | Reserved | Reserved |
| 1 | Reserved | Reserved | Reserved | Reserved | Encoder Enabled | Trip Out | Reserved | Reserved |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Example | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

## Object 2032h: Clock Options

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2032_h$** | **Clock Options** | **VAR** | **Unsigned 8** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/W/S** | **No** | **See Table** | **0** |

| Bit Position | 0x80 | 0x40 | 0x20 | 0x10 | 0x08 | 0x04 | 0x02 | 0x01 |
|---|---|---|---|---|---|---|---|---|
| 0 | n/a | n/a | n/a | n/a | n/a | n/a | n/a | n/a |
| 1 | Invert Direction | Invert Step | Reserved | Square Out | Step Up/Dn | Reserved | Quadrature | Step/ Direction |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Example | 0 | 0 | 0 | 0 | 1 | | 0 | 0 |

## Object 2033h: Capture Input

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2033_h$** | **Capture Input** | **VAR** | **See Entry Desc.** |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default | Data Type |
|---|---|---|---|---|---|---|---|
| **01h** | Enable Capture Position | | R/W | Yes | 0/1 | | |
| **02h** | **Enable Capture Input Flag** | | R/W | No | 0/1 | | |
| **03h** | **Capture Input Filter** | | R/W | No | | | |
| **04h** | **Capture In Position** | | R/W | No | | | |

## Object 2204h: Run Current Percent

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2204_h$** | **Run Current %** | **VAR** | **Unsigned 8** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/W/S** | **No** | **1 - 100** | **25** |

| Run Current % By Device | | | |
|---|---|---|---|
| $2204_h$ | MDrivePlus (All) | MForce MicroDrive (Amps RMS) | MForce PowerDrive (Amps RMS) |
| 10 | MDrive Range 0 To 100%<br><br>Actual Current Not required as Motor is appropriately sized to the device. | 0.3 | 0.5 |
| 20 | | 0.6 | 1.0 |
| 30 | | 0.9 | 1.5 |
| 40 | | 1.2 | 2.0 |
| 50 | | 1.5 | 2.5 |
| 60 | | 1.8* | 3.0 |
| 70 | | 2.1 | 3.5 |
| 80 | | 2.4 | 4.0 |
| 90 | | 2.7 | 4.5 |
| 100 | | 3.0 | 5.0 |

Shaded Area Reserved for Future Use

*HC=67 for maximum 2.0 Amp Hold Current

## Object 2205h: Hold Current Percent

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2205_h$** | **Hold Current %** | **VAR** | **Unsigned 8** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/W/S** | **No** | **0 - 100** | **5** |

| Hold Current % By Device | | | |
|---|---|---|---|
| $2205_h$ | MDrivePlus (All) | MForce MicroDrive (Amps RMS) | MForce PowerDrive (Amps RMS) |
| 10 | MDrive Range 0 To 100%<br><br>Actual Current Not required as Motor is appropriately sized to the device. | 0.3 | 0.5 |
| 20 | | 0.6 | 1.0 |
| 30 | | 0.9 | 1.5 |
| 40 | | 1.2 | 2.0 |
| 50 | | 1.5 | 2.5 |
| 60 | | 1.8* | 3.0 |
| 70 | | 2.1 | 3.5 |
| 80 | | 2.4 | 4.0 |
| 90 | | 2.7 | 4.5 |
| 100 | | 3.0 | 5.0 |

Shaded Area Reserved for Future Use

*HC=67 for maximum 2.0 Amp Hold Current

## Object 2211h: Position Present Point Target

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2211_h$** | **Position Present Point Target** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R** | **No** | **$\pm 2^{31}$** | **0** |

## Object 2212h: Position Final Point Target

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$2212_h$** | **Position Final Point Target** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/W/S** | **No** | **$\pm 2^{31}$** | **0** |

## Object 5001h: Configuration

The following object is set at the factory, and is not user configurable.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$5001_h$** | **Options Setting** | | **Unsigned 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/K** | **No** | **N/A** | **Factory** |

## Object 5002h: ASCII Serial Number

The following object is set at the factory, and is not user configurable. It can be read by the user in the event that the contained data is needed for technical or application support.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$5002_h$** | **ASCII Ser. No.** | | **Unsigned 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/K** | **No** | **N/A** | **Factory** |

## Object 5003h: ASCII Part Number

The following object is set at the factory, and is not user configurable. It can be read by the user in the event that the contained data is needed for technical or application support.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$5003_h$** | **ASCII Part No.** | | **Unsigned 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/K** | **No** | **N/A** | **Factory** |

## Object 5004h: Motor Parameters

The following object is set at the factory, and is not user configurable. It can be read by the user in the event that the contained data is needed for technical or application support.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$5004_h$** | **Motor Parameters** | | **Unsigned 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **R/K** | **No** | **N/A** | **Factory** |

## Device Control

The device control function block controls all the functions of the MDrivePlus CANopen and is divided into to sections:

1. Control of the State Machine
2. Operation Mode

### *Control and Status words*

Controlword (Object Index 6040h) controls the state and operation modes of the MDrivePlus CANopen. Statusword (Object Index 6041h) returns the status of the MDrivePlus CANopen.

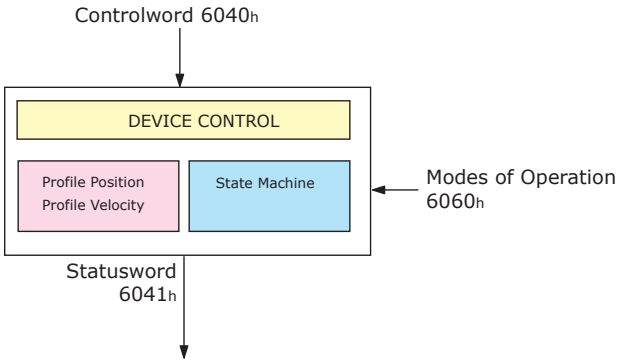



*Figure 4.1: Device Control*

## State Machine

The State Machine describes the status and control sequence of the MDrivePlus CANopen and specifies the Initialization status, the Pre-Operational status, the Operational status, and the Stopped status. See Figure 1.2 for a diagrammatic representation of State machine states and state transitions.

| State Machine States | |
|---|---|
| State | Status Description |
| **Not Ready to Switch On** | Low Level Power Applied.<br>The drive is being initialized or is running a self test.<br>A brake, if present, is applied in this state.<br>The drive function is disabled. |
| **Switch On Disabled** | Drive Initialization is complete.<br>The drive parameters have been set up.<br>Drive parameters may be changed.<br>High Voltage may not be applied to the drive.<br>The drive function is disabled. |
| **Ready To Switch On** | High Voltage may be applied to the drive.<br>The drive parameters may be changed.<br>The drive function is disabled. |
| **Switched On** | High Voltage has been applied to the drive.<br>The Power Amplifier is ready.<br>The drive parameters may be changed.<br>The drive function is disabled. |
| **Operation Enable** | No faults have been detected.<br>The drive function is enabled and power is applied to the motor.<br>The drive parameters may be changed.<br>(This corresponds to normal operation of the drive.) |
| **Quick Stop Active** | The drive parameters may be changed.<br>The Quick Stop function is being executed.<br>The drive function is enabled and power is applied to the motor.<br><br>NOTE: If the 'Quick-Stop-Option-Code' is switched to 5 (Stay in Quick-Stop), the MDrivePlus cannot exit the Quick-Stop-State, but you can transmit to 'Operation Enable' with the command 'Enable Operation'. |
| **Fault Reaction Active**<br>**SUPPORT UNDER DEVELOPMENT** | The drive parameters may be changed.<br>A non-fatal fault has occurred in the drive.<br>The Quick Stop function is being executed.<br>The drive function is enabled and power is applied to the motor. |
| **Fault**<br>**SUPPORT UNDER DEVELOPMENT** | The drive parameters may be changed.<br>A fault has occurred in the drive.<br>The drive function is disabled. |

*Table 4.1: State Machine States*

| State Machine Transitions | | | | |
|---|---|---|---|---|
| Transition Number | From State | To State | Event/Action | |
| **0** | **Start** | **Not Ready To Switch On** | Event: | Reset. |
| | | | Action: | The drive self-tests and/or self-initializes. |
| **1** | **Not Ready To Switch On** | **Switch On Disabled** | Event: | The drive has self-tested and/or initialized successfully. |
| | | | Action: | Activate communication and process data monitoring |
| **2** | **Switch On Disabled** | **Ready to Switch On** | Event: | 'Shutdown' command received from host. |
| | | | Action: | None. |
| **3** | **Ready to Switch On** | **Switched On** | Event: | 'Switch On' command received from host. |
| | | | Action: | The power section is switched on if it is not already switched on. |
| **4** | **Switched On** | **Operation Enable** | Event: | 'Enable Operation' command received from host. |
| | | | Action: | The drive function is enabled. |
| **5** | **Operation Enable** | **Switched On** | Event: | 'Disable Operation' command received from host. |
| | | | Action: | The drive operation will be disabled. |
| **6** | **Switched On** | **Ready to Switch On** | Event: | 'Shutdown' command received from host. |
| | | | Action: | The power section is switched off. |
| **7** | **Ready to Switch On** | **Switch On Disabled** | Event: | 'Quick stop' command received from host. |
| | | | Action: | None |
| **8** | **Operation Enable** | **Ready to Switch On** | Event: | 'Shutdown' command received from host. |
| | | | Action: | The power section is switched off immediately, and the motor is free to rotate if unbraked |
| **9** | **Operation Enable** | **Switch On Disabled** | Event: | 'Disable Voltage' command received from host. |
| | | | Action: | The power section is switched off immediately, and the motor is free to rotate if unbraked |
| **10** | **Switched On** | **Switch On Disabled** | Event: | 'Disable Voltage' or 'Quick Stop' command received from host. |
| | | | Action: | The power section is switched off immediately, and the motor is free to rotate if unbraked |
| **11** | **Operation Enable** | **Quick Stop Active** | Event: | 'Quick Stop' command received from host. |
| | | | Action: | The Quick Stop function is executed. |
| **12** | **Quick Stop Active** | **Switch On Disabled** | Event: | 'Quick Stop' is completed or 'Disable Voltage' command received from host. This transition is possible, if the Quick-Stop-Option-Code is different 5 (Stay in Quick-Stop) |
| | | | Action: | The power section is switched off. |
| **13** | **All States** | **Fault Reaction Active** | Event: | A fault has occured in the drive. |
| | | | Action: | Execute appropriate fault reaction. |
| **14** | **Fault Reaction Active** | **Fault** | Event: | Fault reaction is completed. |
| | | | Action: | The drive function is disabled, the power section may be switched off. |
| **15** | **Fault** | **Switch On Disabled** | Event: | 'Fault Reset' command is recieved from the host. |
| | | | Action: | A reset of the fault condition is carried out if no fault exists currently on the drive. After leaving the 'Fault' state the Bit 'Fault Reset' of the controlword has to be cleared by the host. |
| **16** | **Quick Stop Active** | **Operation Enable** | Event: | 'Enable Operation' command received from host. This transition is possible if the Quick-Stop-Option-Code is 5,6,7 or 8. |
| | | | Action: | The drive function is enabled |

*Table 4.2: State Machine Transitions*

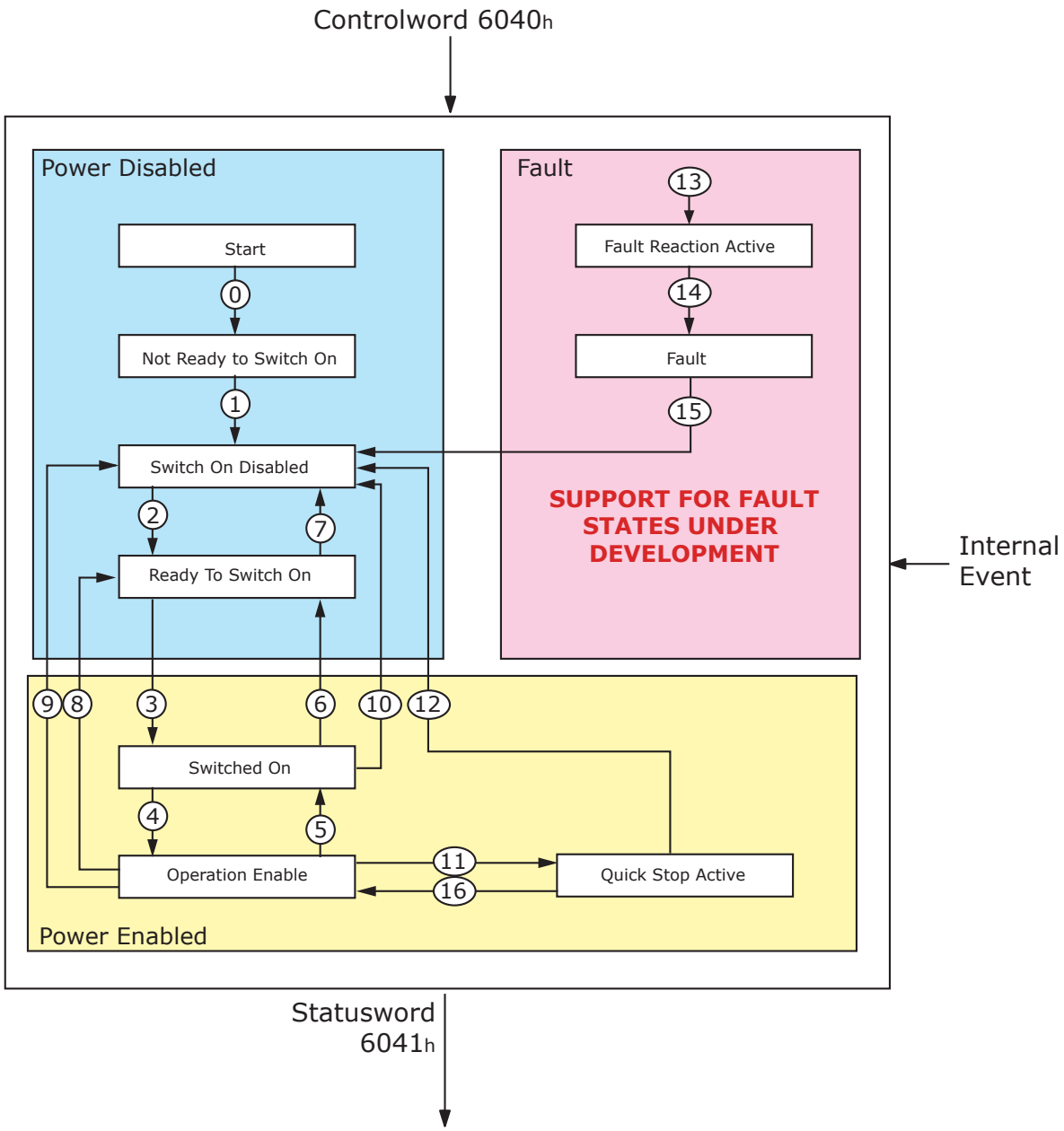



*Figure 4.2: State Machine States/Transitions Block Diagram*

### *Notes On State Transitions*

- Commands directing a change in state are processed completely and the new state achieved before additional state change commands are processed.
- Transitions 0 and 1 occur automatically at drive power-on or reset. All other state changes must be directed by the host.
- Drive function disabled indicates that no current is being supplied to the motor.
- Drive function enabled indicates that current is available for the motor and profile position and profile velocity reference values may be processed.

## Object $6040_h$ — Controlword

Ths controlword is a mandatory, unsigned 16 bit number containing bits for controlling the state and operating modes for the MDrivePlus CANopen.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6040_h$** | **Controlword** | **VAR** | **Unsigned16** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r/w** | **n/a** | **Unsigned16** | **n/a** |

### *Data Description*

MSB LSB

| 15 11 | 10 9 | 8 | 7 | 6 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Manufacturer Specific | Reserved | Halt | Fault Reset | Operation Mode Specific | Enable Operation | Quick Stop | Enable Voltage | Switch On |
| O | O | O | M | O | M | M | M | M |

O=Optional M= Mandatory

### *Device Control Command Bit Patterns (Bits 0-3 and 7)*

| Command | Bit of Controlword (6040h) | | | | | State Transitions |
|---|---|---|---|---|---|---|
| | Fault Reset (Bit 7) | Enable Operation (Bit 3) | Quick Stop (Bit 2) | Enable Voltage (Bit 1) | Switch On (Bit 0) | |
| Shutdown | 0 | X | 1 | 1 | 0 | 2, 6, 8 |
| Switch On | 0 | 0 | 1 | 1 | 1 | 3 |
| Switch On | 0 | 1 | 1 | 1 | 1 | 3 |
| Disable Voltage | 0 | X | X | 0 | X | 7, 9, 10, 12 |
| Quick Stop | 0 | X | 0 | 1 | X | 7,10, 11 |
| Disable Operation | 0 | 0 | 1 | 1 | 1 | 5 |
| Enable Operation | 0 | 1 | 1 | 1 | 1 | 4, 16 |
| Fault Reset | | X | X | X | X | 15 |

*The MDrivePlus CANopen executes the functionality of Switched On

** The MDrivePlus CANopen will do nothing

*Table 4.3: MDrivePlus CANopen Device Control Commands (Bits Marked X are not relevant)*

### *Device Operation Mode Bit Patterns (Bits 4-6 and 8)*

| Operation Mode | Controlword 6040h Bits 4-6, 8 | | | |
|---|---|---|---|---|
| | Bit 8 | Bit 6 | Bit 5 | Bit 4 |
| Profile Position | Halt | Absolute/Relative | Change Set Immediately | New Setpoint |
| Profile Velocity | Halt | Reserved | Reserved | Reserved |
| Homing* | Halt | Homing Operation Start | Reserved | Reserved |

*Homing Mode is currently under development for the MDrivePlus CANopen

*Table 4.4: MDrivePlus CANopen Operation Modes*

## Object $6041_h$ — Statusword

The Statusword is a read-only object that indicates the current state of the drive, no bits are latched. Statusword consists of bits for:

- The current state of the drive.
- The operating state of the mode.
- Manufacturer Specific options.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6041_h$** | **Statusword** | **VAR** | **Unsigned16** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **ro** | **n/a** | **Unsigned16** | **n/a** |

### *Data Description*

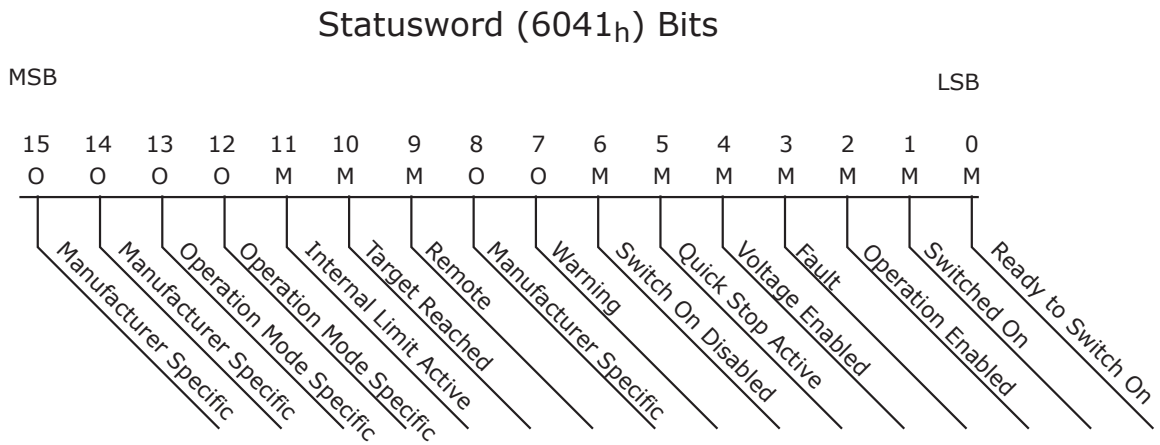



*Figure 4.3: Statusword Bits*

#### *Bits 0-3 and 5-6*

The following bits indicate the status of the MDrivePlus CANopen.

| Status | Bit of Statusword (6041h) | | | | | | |
|---|---|---|---|---|---|---|---|
| | 5 | 6 | 4* | 3 | 2 | 1 | 0 |
| Not Ready to Switch On | 0 | X | X | 0 | 0 | 0 | 0 |
| Switch On Disabled | 1 | X | X | 0 | 0 | 0 | 0 |
| Ready to Switch On | 0 | 1 | X | 0 | 0 | 0 | 1 |
| Switched On | 0 | 1 | X | 0 | 0 | 1 | 1 |
| Operation Enabled | 0 | 1 | X | 0 | 1 | 1 | 1 |
| Quick Stop Active | 0 | 0 | X | 0 | 1 | 1 | 1 |
| Fault Reaction Active | 0 | X | X | 1 | 1 | 1 | 1 |
| Fault | 0 | X | X | 0 | 0 | 0 | 0 |

X=Irrelevant Bit State, *Bit 4 shown for illustration purpose only.

*Table 4.5: Device State Bits for Statusword*

### Bit 4: Voltage Enabled

The Disable Voltage request is active when the voltage_disabled bit is cleared to 0.

### Bit 5: Quick Stop Active

When reset, this bit indicates that the drive is reacting on a quick stop request. Bits 0, 1 and 2 of the statusword must be set to 1 to indicate that the drive is capable to regenerate. The setting of the other bits indicates the status of the drive (e.g. the drive is performing a quick stop as result of a reaction to a non-fatal fault. The fault bit is set as well as bits 0, 1 and 2).

### Bit 7: Warning

A drive warning is present if bit 7 is set. The cause means no error but a state that has to be mentioned, e.g. temperature limit, job refused. The status of the drive does not change. The cause of this warning may be found by reading the fault code parameter. The bit is set and reset by the device.

### Bit 8: Manufacturer Specific

This bit may be used by a drive manufacturer to implement any manufacturer specific functionality.

### Bit 9: Remote

If bit 9 is set, then parameters may be modified via the CAN-network, and the drive executes the content of a command message. If the bit remote is reset, then the drive is in local mode and will not execute the command message. The drive may transmit messages containing valid actual values like a position_actual_value, depending on the actual drive configuration. The drive will accept accesses via service data objects (SDOs) in local mode.

### Bit 10: Target Reached

If bit 10 is set by the drive, then a setpoint has been reached (torque, speed or position depending on the modes_of_operation). The change of a target value by software alters this bit. If quickstop_option_code is 5, 6, 7 or 8, this bit must be set, when the quick stop operation is finished and the drive is halted. If Halt occured and the drive has halted then this bit is set too.

### Bit 11: Internal Limit Active

This bit set by the drive indicates, that an internal limitation is active (e.g. position_range_limit).

### Bits 12-13: Operation Mode Specific

| Operation Mode | Statusword 6041h | |
|---|---|---|
| | Bit 12 | Bit 13 |
| Profile Position | Set Point Acknowledge | Following Error |
| Profile Velocity | Speed | Max Slippage Error |
| Homing* | Homing Attained | Homing Error |

*Homing Mode is currently under development for the MDrivePlus CANopen

*Table 4.6: MDrivePlus CANopen Operation Mode Status*

### Bit 14-15: Manufacturer Specific

These bits may be used by a drive manufacturer to implement any manufacturer specific functionality.

## Object $6060_h$ — Modes of Operation

The performance of the MDrivePlus CANopen depends on the activated Modes of Operation. It is not possible to operates the modes in parallel. The user must select a mode to operate in. An example of exclusive functions are Profile Velocity and Profile Position modes.

The MDrivePlus allows the user to switch dynamically from operation mode to operation mode.

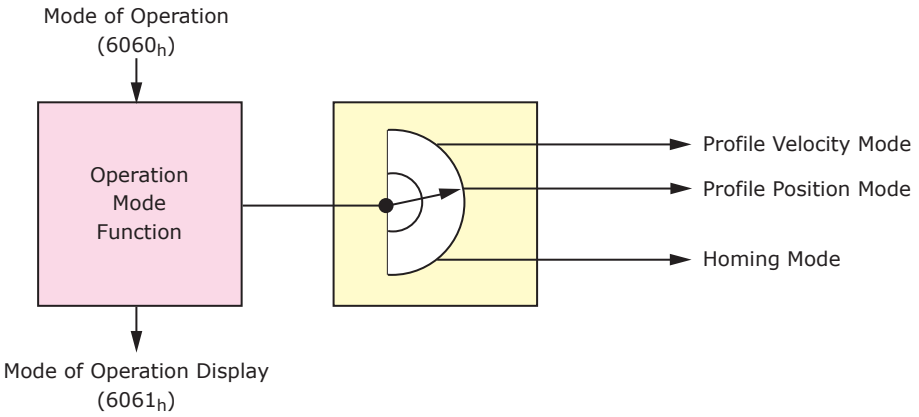



*Figure 5.1: Mode of Operation*

The IMS MDrivePlus CANopen supports the following Modes of Operation:

- Profile Position
- Profile Velocity
- Homing Mode

### Object Description

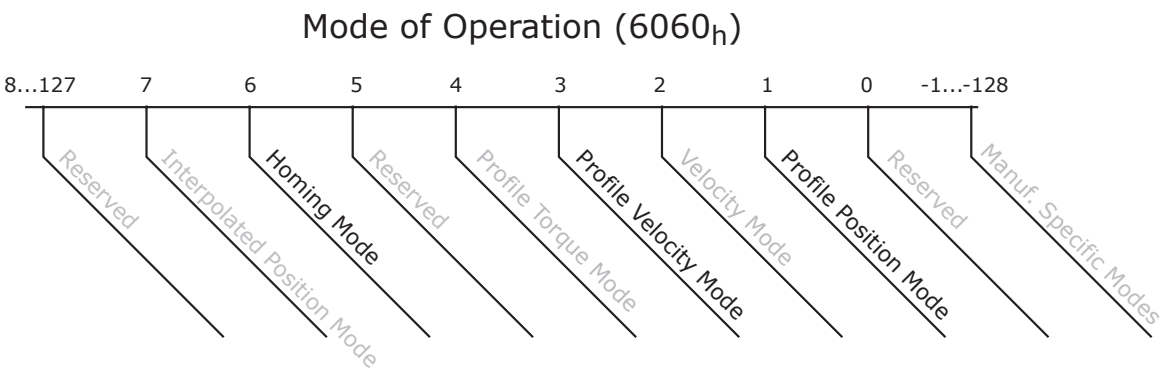
| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6060_h$** | **Mode of Operation** | **VAR** | **±Integer8** |

### Entry Description

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **n/a** | **±Integer8** | **n/a** |

### Data Description

The actual mode is reflected in the modes_of_operation_display (index 6061h), and not in the modes of operation (index 6060h). It may be changed by writing to modes of operation.




Gray Text modes unsupported by MDrivePlus CANopen

*Figure 5.2: Modes of Operation*

## Object $6061_h$ — Modes of Operation Display

The Modes of Operation Display shows the current mode of operation. The meaning of the returned value corresponds to that of the Modes of Operation option code (index 6060h)

### Object Description

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6061_h$** | **Mode of Operation Display** | **VAR** | **±Integer8** |

### Entry Description

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **±Integer8** | **n/a** |

### Data Description

Same as Object 6060h Modes of Operation.

## Object $6502_h$ — Supported Drive Modes

This object shall provide information on the supported drive modes.

### Object Description

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6061_h$** | **Mode of Operation Display** | **VAR** | **±Integer8** |

### Entry Description

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **±Integer8** | **n/a** |

### Data Description

Same as Object 6060h Modes of Operation.

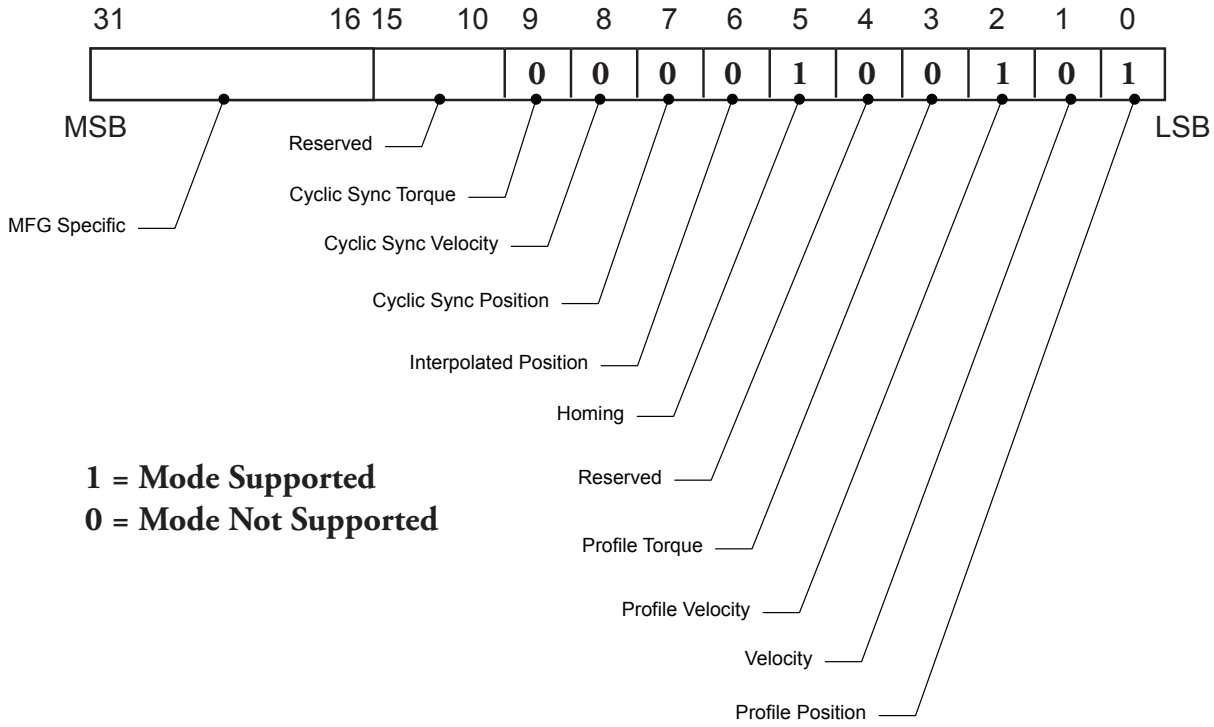



*Figure 5.3: Supported Drive Modes*

*Page Intentionally Left Blank*

## General Information

A ***target_position*** is applied to the Trajectory Generator. It is generating a ***position_demand_value*** for the position control loop described in the Position Control Function Section. These two function blocks are optionally controlled by individual parameter sets.

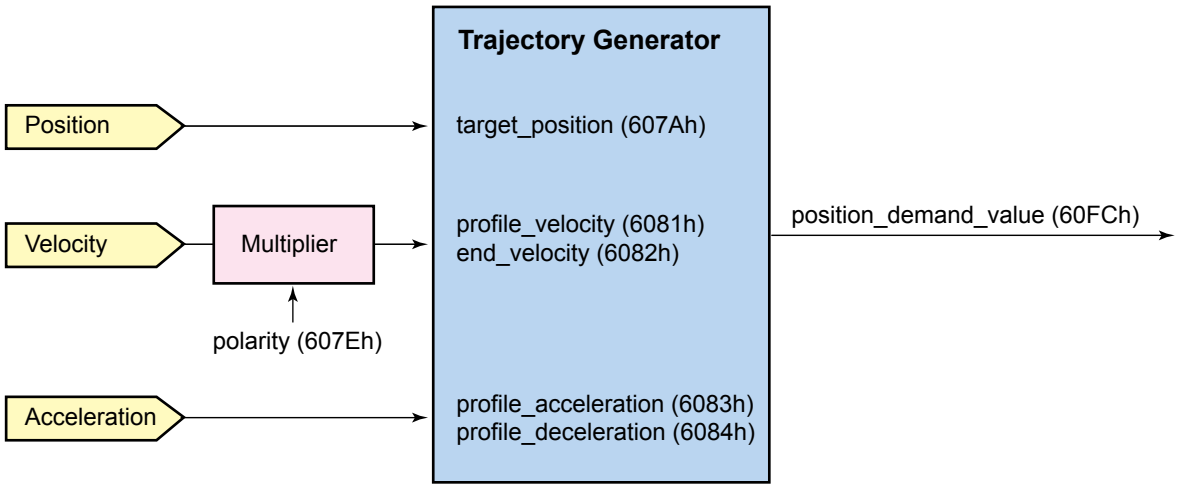



*Figure 6.1: Trajectory Generator Block Diagram*

At the input to the Trajectory Generator, parameters may have optional limits applied before being normalized to internal units. Normalized parameters are denoted with an asterisk. The simplest form of a Trajectory Generator is just to pass through a target_position and to transform it to a position_demand_value* with internal units (increments) only.

For the IMS MDrivePlus CANopen the following values apply:

- target_position — microsteps
- profile_velocity — microsteps/sec
- end_velocity — microsteps/sec
- profile_acceleration — microsteps/sec$^2$
- profile_deceleration — microsteps/sec$^2$
- position_demand_value — microsteps

Note that the MDrivePlus CANopen is fixed at 256 microsteps/full motor step or 51,200 microsteps per motor revolution.

## Input Data Description

| Operating Mode | Description |
|---|---|
| Profile Position | target_position, profile_velocity, end_velocity, profile_ acceleration, profile_deceleration |

## Output Data Description

| Operating Mode | Description |
|---|---|
| Profile Position | position_demand_value |

## Functional Description

There are two different ways to apply ***target_positions*** to a drive, are supported by this device profile.

1. Set of set-points:

   After reaching the ***target_position*** the drive unit immediately processes the next target_position which results in a move where the velocity of the drive normally is not reduced to zero after achieving a set-point.

2. Single set-point:

   After reaching the ***target_position*** the drive unit signals this status to a host computer and then receives a new set-point. After reaching a ***target_position*** the velocity normally is reduced to zero before starting a move to the next set-point.

The two modes are controlled by the timing of the bits ***new_set-point*** and ***change_set_immediately*** in the ***controlword*** and ***set-point_acknowledge*** in the ***statusword***.

These bits allow to set up a request-response mechanism in order to prepare a set of set-points while another set still is processed in the drive unit. This minimizes reaction times within a control program on a host computer.


*Figure 6.2: Set-Point Transmission from Host Computer*

Figure 6.2, Figure 6.3 and Figure 6.4 illustrate the difference between the "set of set-points" mode and the "single set-point" mode. The initial status of the bit ***change_set_immediately*** in the ***controlword*** determines which mode is used. Trapezoidal moves are used as this is the only ***motion_profile_type*** the MDrivePlus CANopen supports.

If the bit ***change_set_immediately*** is "0" (shaded area in Figure 3.2) a single set-point is expected by the drive ❶. After data is applied to the drive, a host signals that the data is valid by changing the bit ***new_setpoint*** to "1" in the ***controlword*** ❷. The drive responds with ***set-point_acknowledge*** set to "1" in the ***statusword*** ❸ after it recognized and buffered the new valid data. Now the host may release ***new_setpoint*** ❹ and afterwards the drive signals with ***set-point_acknowledge*** equal "0" its ability to accept new data again ❺. In Figure 3.3 this mechanism results in a velocity of zero after ramping down in order to reach a ***target_position*** $X_1$ at $T_1$. After signalling to the host, that the set-point is reached like described above, the next ***target_position*** $X_2$ is processed at $T_2$ and reached at $T_3$.

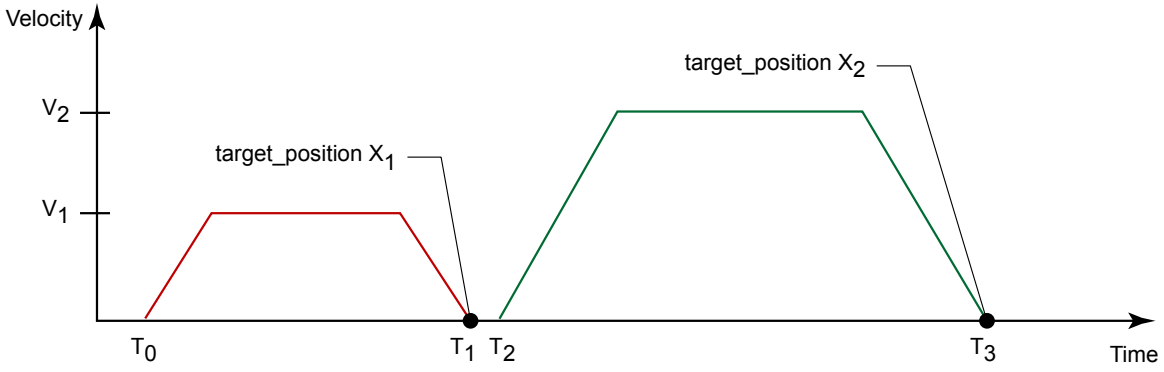



*Figure 6.3: Single Set-Point Mode (Move After a Move) 6040h Bit 5=0*

With change_set_immediately set to "1" ❻, symbolized by the clear area in Figure 6.2, the host advises the drive to apply a new set-point immediately after reaching the last one. The relative timing of the other signals is unchanged. This behavior causes the drive to already process the next set-point $X_2$ and to keep its velocity when it reaches the target_position $X_1$ at $T_1$. Then drive moves immediately to the already calculated next target_position $X_2$.

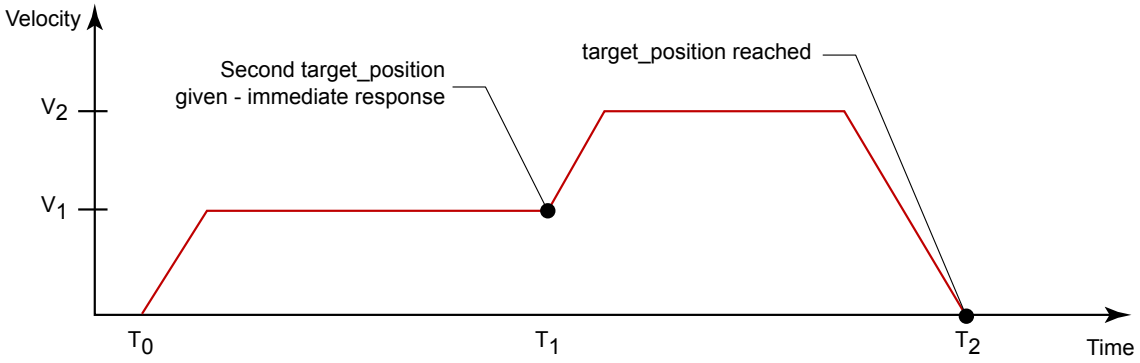



*Figure 6.4: Set of Setpoints (Move on a Move) 6040h Bit 5=1*

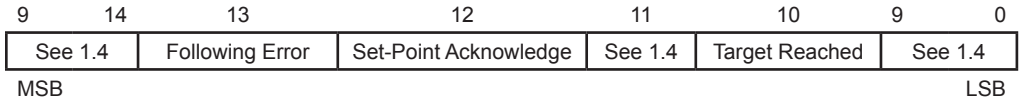
## Controlword ($6040_h$) of Profile Position Mode

| 15 9 | 8 | 7 | 6 | 5 | 4 | 3 0 |
|---|---|---|---|---|---|---|
| See 1.3 | Halt | See 1.3 | abs/rel | Change Set Immediately | New Set-Point | See 1.3 |
| MSB | | | | | | LSB |

| Bit | Name | Value | Description |
|---|---|---|---|
| 4 | New Set Point | 0 | Does not assume target position |
| | | 1 | Assume target position |
| 5 | Change Set Immediately | 0 | Finish the actual positioning and then start the next positioning |
| | | 1 | Interrupt the actual positioning and start the next positioning |
| 6 | abs/rel | 0 | Target position is an absolute value |
| | | 1 | Target position is a relative value |
| 8 | Halt | 0 | Execute positioning |
| | | 1 | Stop motion with profile deceleration |

*Table 6.1: Profile Position Mode Bits of Controlword*

## Statusword ($6041_h$) of Profile Position Mode




| Bit | Name | Value | Description |
|---|---|---|---|
| 10 | Target Reached | 0 | Halt=0: Target position not reached<br>Halt=1: Axis decelerating |
| | | 1 | Halt=0: Target position reached<br>Halt=1: Axis velocity is 0 |
| 12 | Set-Point Acknowledge | 0 | Trajectory generator has not assumed the positioning values yet |
| | | 1 | Trajectory generator has assumed the positioning values |
| 13 | Following Error | 0 | No following error |
| | | 1 | Following error |

*Table 6.2: Profile Position Mode Bits of Statusword*

## Object $607A_h$ — Target Position

The Target Position is the position that the drive should move to in position profile mode using the MDrivePlus CANopen parameters such as velocity, acceleration, deceleration, motion profile type etc. The target position is given in terms of 51,200 units per motor shaft revolution. The target position will be interpreted as absolute or relative depending on the absolute relative flag (bit 6) in the controlword.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$607A_h$** | **Target Position** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 32** | **n/a** |

## Object $6081_h$ — Profile Velocity

The profile velocity is the velocity normally attained at the end of the acceleration ramp during a profiled move and is valid for both directions of motion. The profile velocity is given in steps per second..

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6081_h$** | **Profile Velocity** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 32** | **n/a** |

## Object $6082_h$ — End Velocity

The end velocity defines the velocity which the drive must have on reaching the target position. Normally, the drive stops at the target position, i.e. the end_velocity = 0. The end velocity is given in the same units as profile velocity.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6082_h$** | **End Velocity** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 32** | **n/a** |

## Object $6083_h$ — Profile Acceleration

Profile Acceleration is given in steps/$sec^2$

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6083_h$** | **Profile Acceleration** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 32** | **n/a** |

## Object $6084_h$ — Profile Deceleration

Profile Deceleration is given in steps/$sec^2$

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6084_h$** | **Profile Deceleration** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 32** | **n/a** |

## Object $6086_h$ — Motion Profile Type

The Motion Profile Type is used to select the type of motion profile used to perform a move. The MDrivePlus CANopen is fixed at Value 0: Linear Ramp (Trapezoidal Profile)

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6086_h$** | **Motion Profile Type** | **VAR** | **Integer 16** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 16** | **0** |

## General Information

This chapter describes the method by which a drive seeks the home position (also called, the datum, reference point or zero point). There are various methods of achieving this using limit switches at the ends of travel or a home switch (zero point switch) in mid-travel, most of the methods also use the index (zero) pulse train from an incremental encoder.

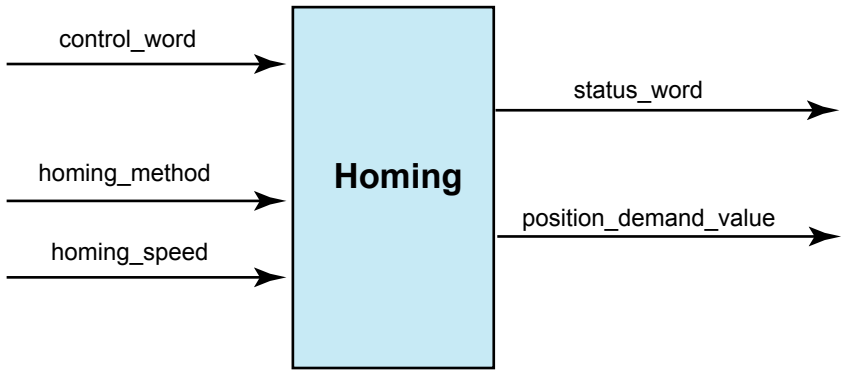



*Figure 7.1: The Homing Function*

### Input Data Description

The user can specify the speeds and the method of homing. There are two homing_speeds; in a typical cycle the faster speed is used to find the home switch and the slower speed is used to find the index pulse. The manufacturer is allowed some discretion in the use of these speeds as the response to the signals may be Dependent upon the hardware used.

### Output Data Description

There is no output data except for those bits in the statusword which return the status or result of the homing process and the demand to the position control loops.

### Internal States

There is only one internal state called homing which is reflected in the bits of the statusword.

## Controlword ($6040_h$) of Profile Position Mode

| 15 9 | 8 | 7 | 6 5 | 4 | 3 0 |
|---|---|---|---|---|---|
| See 1.3 | Halt | See 1.3 | Reserved | Homing Operation Start | See 1.3 |
| MSB | | | | | LSB |

| Bit | Name | Value | Description |
|---|---|---|---|
| 4 | Homing Operation Start | 0 | Homing Mode Inactive |
| | | 0 ➤ 1 | Start Homing Mode |
| | | 1 | Homing Mode Active |
| | | 1 ➤ 0 | Interrupt Homing Mode |
| 8 | Halt | 0 | Execute the instruction of bit 4 |
| | | 1 | Stop motion |

*Table 7.1: Homing Mode Bits of Controlword*

## Statusword ($6041_h$) of Homing Mode

| 9 14 | 13 | 12 | 11 | 10 | 9 0 |
|---|---|---|---|---|---|
| See 1.4 | Homing Error | Homing Attained | See 1.4 | Target Reached | See 1.4 |

MSB LSB

| Bit | Name | Value | Description |
|---|---|---|---|
| 10 | Target Reached | 0 | Halt=0: Home position not reached<br>Halt=1: Axis decelerating |
| | | 1 | Halt=0: Home position reached<br>Halt=1: Axis velocity is 0 |
| 12 | Homing Attained | 0 | Homing Mode not yet complete |
| | | 1 | Homing Mode carried out successfully |
| 13 | Following Error | 0 | No homing error |
| | | 1 | Homing error |

*Table 7.2: Homing Mode Bits of Statusword*

## Homing Offset (607Ch)

This object shall indicate the conFigure d difference between the zero position for the application and the machine home position (found during homing). During homing the machine home position is found and once the homing is completed the zero position is offset from the home position by adding the home offset to the home position. All subsequent absolute moves shall be taken relative to this new zero position. This is illustrated in Figure 7.2. If this object is not implemented then the home offset shall be regarded as zero. The value of this object shall be given in micro steps. Negative values shall indicate the opposite direction.

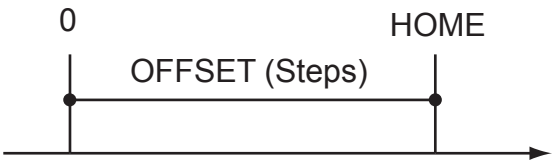



*Figure 7.2: Home Offset*

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$607C_h$** | **Homing Offset** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **Integer 32** | **$0_d$** |

## Homing Method (6098h)

The homing method object determines the method that will be used during homing.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6098_h$** | **Homing Method** | **VAR** | **± Integer 8** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **Possible** | **± Integer 8** | **0** |

### *Data Description*

| Value | Description |
|---|---|
| -128 — -1 | Manufacturer Specific |
| 0 | No Homing Operation Required |
| 1 — 35 | Homing Methods 1 through 35 (See Functional Description) |
| 36 — 128 | Reserved |

### *Functional Description of Homing Methods*

#### Method 1: Homing on the Negative Limit Switch and Index Pulse

Using this method the initial direction of movement is leftward if the negative limit switch is inactive (here shown as low). The home position is at the first index pulse to the right of the position where the negative limit switch becomes inactive.

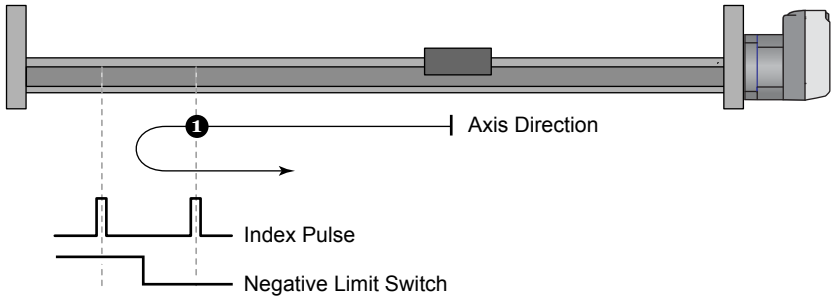



*Figure 7.3: Homing on the Negative Limit and Index Pulse*

### Method 2: Homing on the Positive Limit Switch and Index Pulse

Using this method the initial direction of movement is rightward if the positive limit switch is inactive (here shown as low). The position of home is at the first index pulse to the left of the position where the positive limit switch becomes inactive.

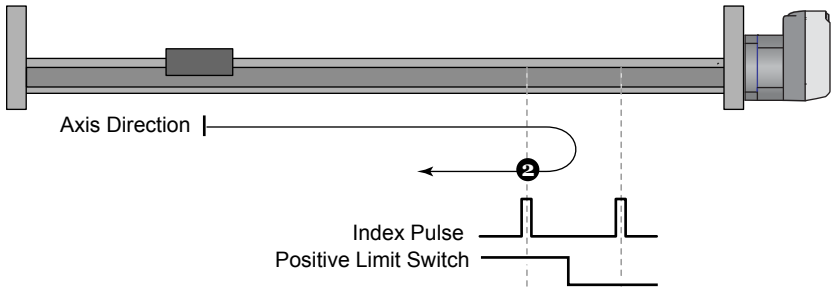



*Figure 7.4: Homing on the Positive Limit and Index Pulse*

### Methods 3 and 4: Homing on the Positive Home Switch and Index Pulse

Using methods 3 or 4 the initial direction of movement is dependent on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is sited so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

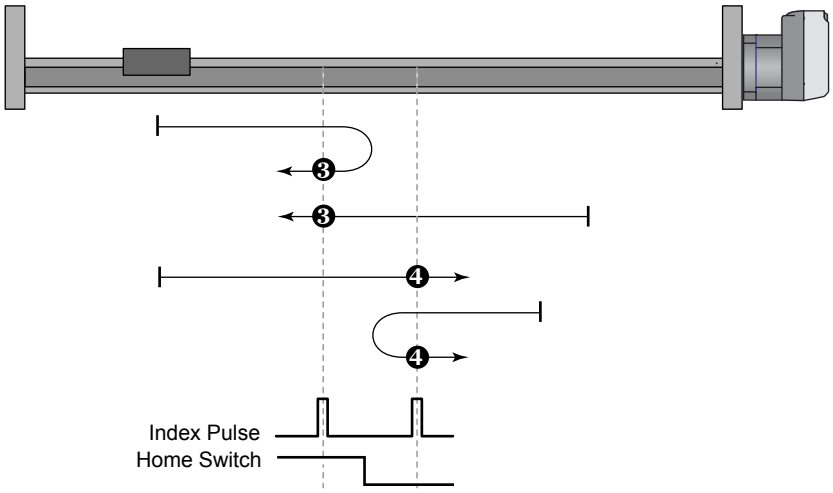



*Figure 7.5: Homing on the Positive Home Switch and Index Pulse*

### Methods 5 and 6: Homing on the Negative Home Switch and Index Pulse

Using methods 5 or 6 the initial direction of movement is dependent on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is sited so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

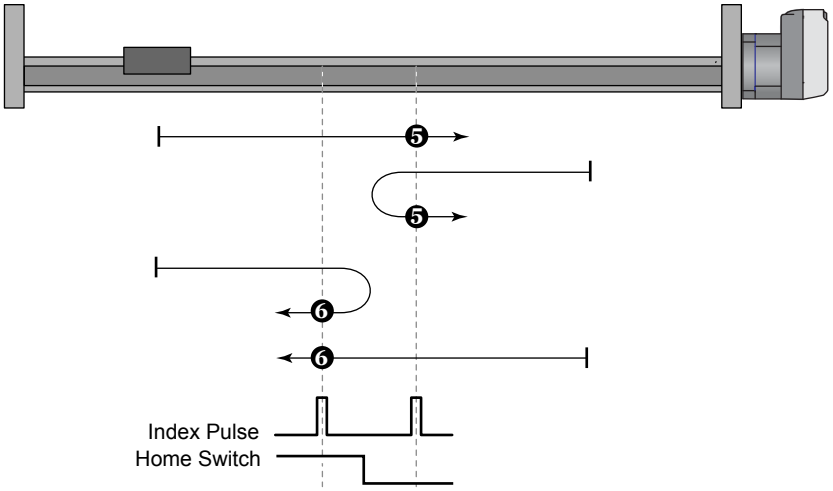



*Figure 7.6: Homing on the Negative Home Switch and Index Pulse*

### Methods 7 to 14: Homing on the Home Switch and Index Pulse

These methods use a home switch which is active over only portion of the travel, in effect the switch has a 'momentary' action as the axle`s position sweeps past the switch.

Using methods 7 to 10 the initial direction of movement is to the right, and using methods 11 to 14 the initial direction of movement is to the left except if the home switch is active at the start of the motion. In this case the initial direction of motion is Dependent on the edge being sought. The home position is at the index pulse on either side of the rising or falling edges of the home switch, as shown in the following two diagrams. If the initial direction of movement leads away from the home switch, the drive must reverse on encountering the relevant limit switch.

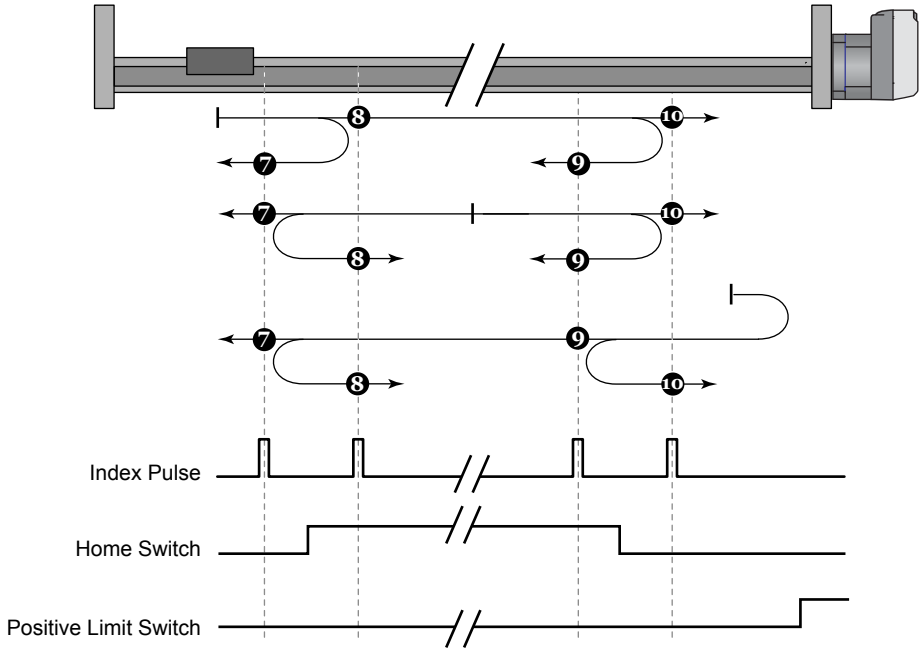



*Figure 7.7: Homing on the Home Switch and Index Pulse - Positive Initial Move*

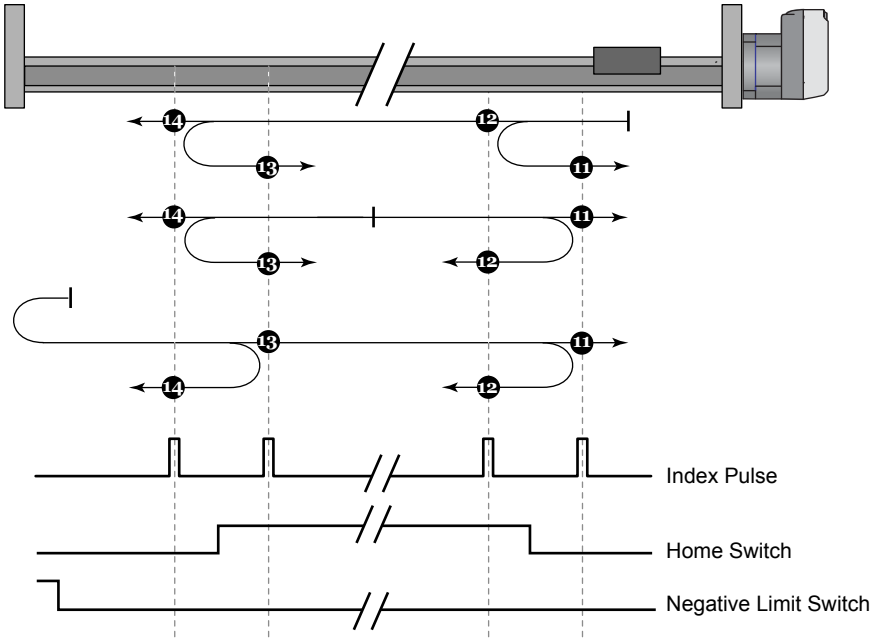



*Figure 7.8: Homing on the Home Switch and Index Pulse - Negative Initial Move*

### Methods 15 and 16: Reserved

These methods are reserved for future expansion of the homing mode.

### Methods 17 to 30: Homing without an Index Pulse

These methods are similar to methods 1 to 14 except that the home position is not dependent on the index pulse but only Dependent on the relevant home or limit switch transitions. For example methods 19 and 20 are similar to methods 3 and 4 as shown in the following diagram.

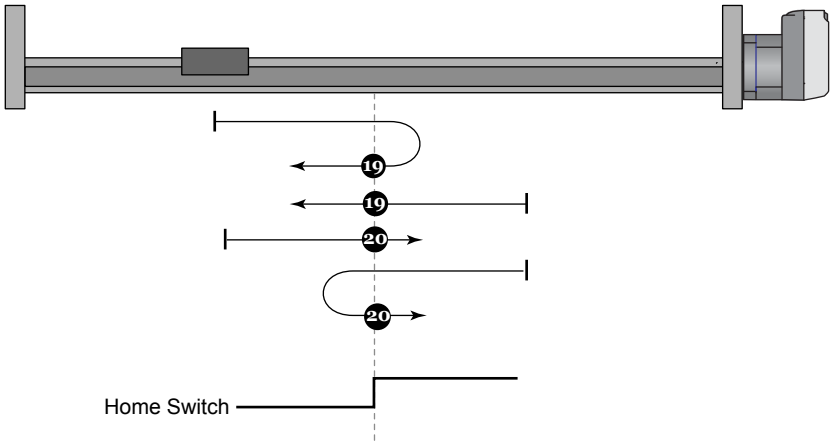



*Figure 7.9: Homing without an Index Pulse*

### Methods 31 and 32: Reserved

These methods are reserved for future expansion of the homing mode.

### Methods 33 and 34: Homing on an Index Pulse

Using methods 33 or 34 the direction of homing is negative or positive respectively. The home position is at the index pulse found in the selected direction.

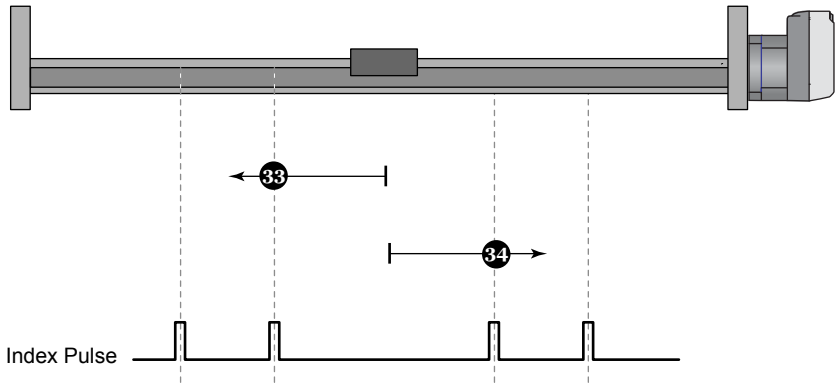



*Figure 7.10: Homing on the Index Pulse*

### Method 35: Homing on the Current Position

In method 35 the current position is taken to be the home position.

## Homing Speeds (6099h)

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6099_h$** | **Homing Speeds** | **VAR** | **Unsigned 32** |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **01h** | Number of Entries | Mandatory | RO | No | 2 | 2 |
| **02h** | Speed during search for switch | Mandatory | R/W | Possible | Unsigned 32 | 0 |
| **03h** | Speed during search for zero | Mandatory | R/W | Possible | Unsigned 32 | 0 |

*Page Intentionally Left Blank*

## General Information

In this chapter, all parameters are described which are necessary for a closed loop position control. The control loop is fed with the ***position_demand_value*** as one of the outputs of the Trajectory Generator and with the output of the position detection unit (***position_actual_value***) like a resolver or encoder as input parameters.

## Object $6062_h$ — Position Demand Value

This object shall provide the demanded position value. The value shall be given in motor steps.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6062_h$** | **Position Demand Value** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **Integer 32** | **n/a** |

## Object $6063_h$ — Position Actual Value Internal

This object shall provide the actual value of the position measurement device, which shall be one of the two input values of the closed-loop position control

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6063_h$** | **Position Actual Value*** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **Integer 32** | **n/a** |

## Object $6064_h$ — Position Actual Value

This object represents the actual value of the position measurement device microsteps.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6064_h$** | **Position Actual Value** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **Integer 32** | **n/a** |

## Object $6065_h$ — Following Error Window

This object shall indicate the conFigure d range of tolerated position values symmetrically to the position demand value. If the position actual value is out of the following error window, a following error occurs. A following error may occur when a drive is blocked, unreachable profile velocity occurs, or at wrong closed-loop coefficients. The value shall be given in user defined position units. If the value of the following error window is FFFF FFFFh, the following control shall be switched off.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6065_h$** | **Following Error Window** | **VAR** | **Unsigned 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **Unsigned 32** | **n/a** |

## Object $6066_h$ — Following Error Timeout

This object shall indicate the conFigure d time for a following error condition, after that the bit 13 of the statusword shall be set to 1. The reaction of the drive when a following error occurs is manufacturer-specific. The value shall be given in milliseconds.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6066_h$** | **Following Error Timeout** | **VAR** | **Unsigned 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **Unsigned 32** | **n/a** |

## Object $6068_h$ — Position Window Time

This object shall indicate the conFigure d time, during which the actual position within the position window is measured. The value shall be given in milliseconds.

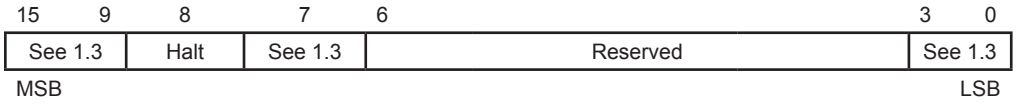
### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$6068_h$** | **Position Window Time** | **VAR** | **Unsigned 16** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **r** | **n/a** | **Unsigned 16** | **Mfg-Specific** |

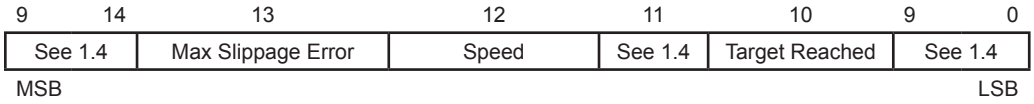
## Controlword ($6040_h$) of Profile Velocity Mode




| Bit | Name | Value | Description |
|---|---|---|---|
| 8 | Halt | 0 | Execute the Motion |
| | | 1 | Stop axis |

*Table 8.1: Profile Velocity Mode Bits of Controlword*

## Statusword ($6041_h$) of Profile Velocity Mode




| Bit | Name | Value | Description |
|---|---|---|---|
| 10 | Target Reached | 0 | Halt=0: Target position not reached<br>Halt=1: Axis decelerating |
| | | 1 | Halt=0: Target position reached<br>Halt=1: Axis velocity is 0 |
| 12 | Speed | 0 | Speed is not equal to 0 |
| | | 1 | Speed is equal 0 |
| 13 | Max Slippage Error | 0 | Maximum slippage not reached |
| | | 1 | Maximum slippage reached |

*Table 8.2: Profile Velocity Mode Bits of Statusword*

## Object $606C_h$ — Velocity Actual Value

This object shall provide the actual velocity value derived either from the velocity sensor or the position sensor. The value shall be given in microsteps per second.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **$606C_h$** | **Velocity Actual Value** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **ro** | **n/a** | **Integer 32** | **n/a** |

## Object 60FF$_h$ — Target Velocity

The Target Velocity is the input to the trajectory generator and the value is given in microsteps/second.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **60FF$_h$** | **Target Velocity** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **n/a** | **Integer 32** | **n/a** |

## Object 60F8$_h$ — Maximum Slippage

This object shall indicate the conFigure d maximal slippage of an asynchronous motor. When the max slippage has been reached, the corresponding bit 13 max slippage error in the statusword shall be set to 1. The reaction of the drive device, when the max slippage error occurs, is manufacturer-specific. This value shall be given in umicrosteps.

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **60FF$_h$** | **Maximum Slippage** | **VAR** | **Integer 32** |

### *Entry Description*

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **rw** | **n/a** | **Integer 32** | **Mfg-Specific** |

## Object 60FD$_h$ — Digital Inputs

This object provides for digital inputs.

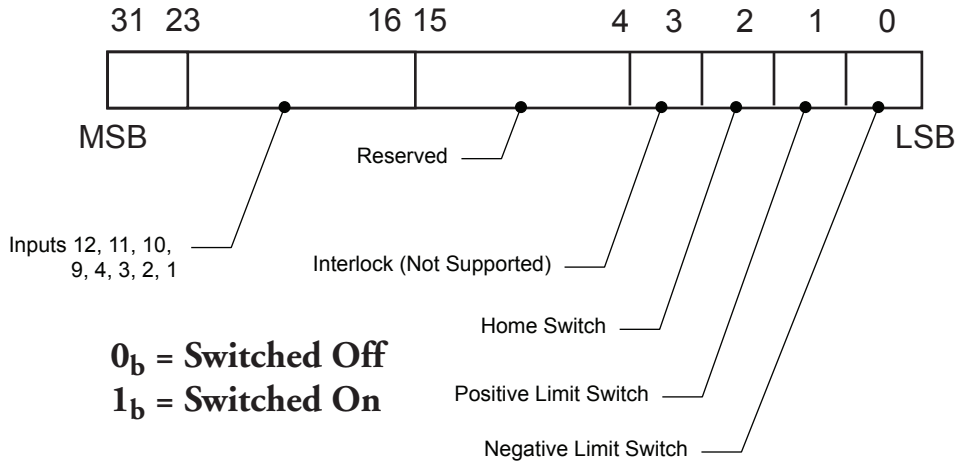



*Figure 10.1: Object 60FD Structure*

### Object Description

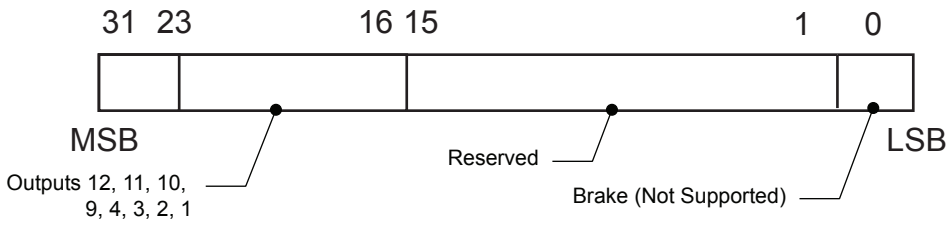
| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **60FD$_h$** | **Digital Inputs** | **VAR** | **Unsigned 32** |

### Entry Description

| Access | PDO Mapping | Range | Default |
|---|---|---|---|
| **ro** | **n/a** | **Unsigned 32** | **n/a** |

## Object 60FE$_h$ — Digital Outputs

This object provides for digital outputs.




**$0_b$ = Switch Off**
**$1_b$ = Switch On**

*Figure 10.2: Object 60FE Structure*

### *Object Description*

| Index | Name | Object Code | Data Type |
|---|---|---|---|
| **60FE$_h$** | **Digital Outputs** | **ARRAY** | **Unsigned 32** |

### *Entry Description*

| Sub-Index | Description | Category | Access | PDO Mapping | Value Range | Default |
|---|---|---|---|---|---|---|
| **00h** | Highest Supported Sub-Index | Mandatory | C | No | $02_h$ | Mfg-Specific |
| **01h** | Physical Outputs | Mandatory | R/W | Possible | Unsigned 32 | 0000 0000$_h$ |
| **02h** | Bit mask | Mandatory | R/W | Possible | Unsigned 32 | 0000 0000$_h$ |

# WARRANTY

### TWENTY-FOUR (24) MONTH LIMITED WARRANTY

Intelligent Motion Systems, Inc. (“IMS”), warrants only to the purchaser of the Product from IMS (the “Customer”) that the product purchased from IMS (the “Product”) will be free from defects in materials and workmanship under the normal use and service for which the Product was designed for a period of 24 months from the date of purchase of the Product by the Customer. Customer’s exclusive remedy under this Limited Warranty shall be the repair or replacement, at Company’s sole option, of the Product, or any part of the Product, determined by IMS to be defective. In order to exercise its warranty rights, Customer must notify Company in accordance with the instructions described under the heading “Obtaining Warranty Service.”

**NOTE:** MDrivePlus Motion Control electronics are not removable from the motor in the field. The entire unit must be returned to the factory for repair.

This Limited Warranty does not extend to any Product damaged by reason of alteration, accident, abuse, neglect or misuse or improper or inadequate handling; improper or inadequate wiring utilized or installed in connection with the Product; installation, operation or use of the Product not made in strict accordance with the specifications and written instructions provided by IMS; use of the Product for any purpose other than those for which it was designed; ordinary wear and tear; disasters or Acts of God; unauthorized attachments, alterations or modifications to the Product; the misuse or failure of any item or equipment connected to the Product not supplied by IMS; improper maintenance or repair of the Product; or any other reason or event not caused by IMS.

IMS HEREBY DISCLAIMS ALL OTHER WARRANTIES, WHETHER WRITTEN OR ORAL, EXPRESS OR IMPLIED BY LAW OR OTHERWISE, INCLUDING WITHOUT LIMITATION, **ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE**. CUSTOMER’S SOLE REMEDY FOR ANY DEFECTIVE PRODUCT WILL BE AS STATED ABOVE, AND IN NO EVENT WILL THE IMS BE LIABLE FOR INCIDENTAL, CONSEQUENTIAL, SPECIAL OR INDIRECT DAMAGES IN CONNECTION WITH THE PRODUCT.

This Limited Warranty shall be void if the Customer fails to comply with all of the terms set forth in this Limited Warranty. This Limited Warranty is the sole warranty offered by IMS with respect to the Product. IMS does not assume any other liability in connection with the sale of the Product. No representative of IMS is authorized to extend this Limited Warranty or to change it in any manner whatsoever. No warranty applies to any party other than the original Customer.

IMS and its directors, officers, employees, subsidiaries and affiliates shall not be liable for any damages arising from any loss of equipment, loss or distortion of data, loss of time, loss or destruction of software or other property, loss of production or profits, overhead costs, claims of third parties, labor or materials, penalties or liquidated damages or punitive damages, whatsoever, whether based upon breach of warranty, breach of contract, negligence, strict liability or any other legal theory, or other losses or expenses incurred by the Customer or any third party.

### OBTAINING WARRANTY SERVICE

Warranty service may obtained by a distributor, if the Product was purchased from IMS by a distributor, or by the Customer directly from IMS, if the Product was purchased directly from IMS. Prior to returning the Product for service, a Returned Material Authorization (RMA) number must be obtained. Complete the form at http://www.imshome.com/rma.html after which an RMA Authorization Form with RMA number will then be faxed to you. Any questions, contact IMS Customer Service (860) 295-6102.

Include a copy of the RMA Authorization Form, contact name and address, and any additional notes regarding the Product failure with shipment. Return Product in its original packaging, or packaged so it is protected against electrostatic discharge or physical damage in transit. The RMA number MUST appear on the box or packing slip. Send Product to: Intelligent Motion Systems, Inc., 370 N. Main Street, Marlborough, CT 06447.

Customer shall prepay shipping changes for Products returned to IMS for warranty service and IMS shall pay for return of Products to Customer by ground transportation. However, Customer shall pay all shipping charges, duties and taxes for Products returned to IMS from outside the United States.

# INTELLIGENT MOTION SYSTEMS, INC.

www.imshome.com

*Excellence in Motion™*


**370 N. Main St., P.O. Box 457**
**Marlborough, CT 06447 U.S.A.**
**Phone: 860/295-6102**
**Fax: 860/295-6107**
**E-mail: info@imshome.com**

**TECHNICAL SUPPORT**
Phone: 860/295-6102 (USA)
Fax: 860/295-6107
E-mail: etech@imshome.com
**Germany/UK**
Phone: +49/7720/94138-0
Fax: +49/7720/94138-2
E-mail: mweber@imshome.com

**U.S.A. SALES OFFICES**
**Eastern Region**
Phone: 862/208-9742
Fax: 973/661-1275
E-mail: jroake@imshome.com
**Central Region**
Phone: 260/402-6016
Fax: 419/858-0375
E-mail: dwaksman@imshome.com
**Western Region**
Phone: 602/578-7201
E-mail: dweisenberger@imshome.com

**IMS MOTORS DIVISION**
105 Copperwood Way, Suite H
Oceanside, CA 92054
Phone: 760/966-3162
Fax: 760/966-3165
E-mail: motors@imshome.com

**IMS EUROPE GmbH**
Hahnstrasse 10, VS-Schwenningen
Germany D-78054
Phone: +49/7720/94138-0
Fax: +49/7720/94138-2
E-mail: info@imseuropehome.com
**European Sales Management**
4 Quai Des Etroits
69005 Lyon, France
Phone: +33/4 7256 5113
Fax: +33/4 7838 1537
E-mail: bmartinez@imshome.com
**Germany Sales**
Phone: +49/35205/4587-8
Fax: +49/35205/4587-9
E-mail: hruhland@imshome.com
**Germany/UK Technical Support**
Phone: +49/7720/94138-0
Fax: +49/7720/94138-2
E-mail: mweber@imshome.com

**IMS UK Ltd.**
25 Barnes Wallis Road
Segensworth East
Fareham, Hampshire PO15 5TT
Phone: +44/0 1489-889825
Fax: +44/0 1489-889857
E-mail: mcheckley@imshome.com

**IMS ASIA PACIFIC OFFICE**
30 Raffles Pl., 23-00 Caltex House
Singapore 048622
Phone: +65/6233/6846
Fax: +65/6233/5044
E-mail: wllee@imshome.com

**DISTRIBUTED BY:**




REV020507

*IMS Product Disclaimer and most recent product information at www.imshome.com.*