



SERCOS

--Table of Contents--

| ---Topic--- | pg |
|--------------------------------|-----------|
| Introduction | 2 |
| Overview | 3 |
| Data Types | 7 |
| Communications | 13 |
| Procedures | 17 |
| Telegrams | 19 |
| Topologies | 27 |

Copyright © 2002
Motion Engineering

Sercos- Introduction

[About SERCOS](#) | [Supported Drives](#)

About the MEI SERCOS Controller

The SERCOS/XMP Series controllers from MEI are an extension of the XMP Series motion controllers. XMP Series motion controllers support analog motion control outputs, encoder inputs, and discrete digital I/O. SERCOS/XMP Series motion controllers replace these signals with the SERCOS digital fiber optic network interface. The SERCOS interface only requires two fiber optic connections (one output and one input) to connect to a fiber loop containing up to 8 axes of motion.

Both SERCOS and standard XMP Series controllers share the same basic hardware architecture, onboard firmware, host software and many other features. So, for controller installation procedures, Motion Console application and C programming information, use the XMP's standard documentation.

However, because the SERCOS IDNs are actually implemented in the drive (and not in the controller), there are many SERCOS functions not documented in the XMP documentation, because they are in the drive's documentation (because these functions are associated with the drive and not the controller). Motion Engineering adheres to the specifications set forth by the IEC concerning SERCOS. For more information regarding SERCOS or the SERCOS specification, please contact SERCOS N.A at www.sercos.com.



Supported Drives / Modules

MEI currently supports drive and I/O modules from a variety of manufacturers. If you desire support for a drive or I/O vendor not listed, please contact MEI.

| Manufacturer | Device |
|--------------------|---------------------|
| Indramat | Servo Drives |
| Modicon | Servo Drives |
| Lutze | Digital I/O Modules |
| Pacific Scientific | Servo Drives |
| Kollmorgen | Servo Drives |
| Sanyo Denki | Servo Drives |



[Introduction](#) | [Overview](#) | [Data Types](#) | [Communications](#) | [Procedures](#) | [Telegrams](#) | [Topologies](#)



Copyright © 2002
Motion Engineering

Sercos- Overview

[Summary](#) | [Operation Modes](#) | [Closed-Loop Tuning](#) | [Data Transmission](#)

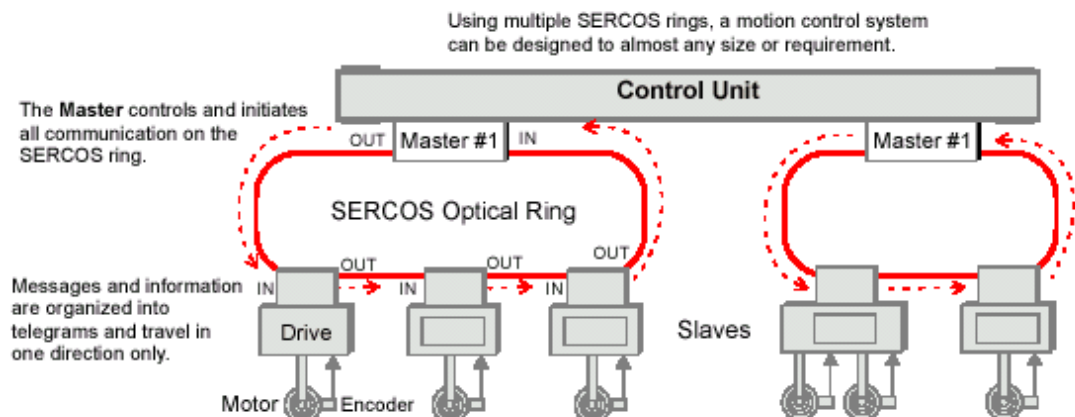
Summary

SERCOS (Serial Real-time COmmunication System) is the international standard for optical communication between motion control units and drive modules. It was developed by the International Electrotechnical Commission (IEC) specifically for motion control and is defined by the IEC 1491 standard. SERCOS supports data rates up to 16 Mbits/sec over a fiber optic ring. Data can be transmitted deterministically in real time based on the loop update rate (cyclic), or at lower bandwidths for less critical operations (non-cyclic). A SERCOS-compatible communication ring must have a single controller (master) and 1 - 254 drive or I/O modules (slaves).

Also, data communication can be performed in either a synchronous or asynchronous manner. The protocol allows the user to configure the communication telegrams to send whatever data is appropriate synchronously. Data that has not been configured to reside in the communication telegrams can be sent or retrieved asynchronously by use of a Service Channel contained within the communication telegrams. Generally, synchronous data is data that is critical to real-time operation (e.g., command and feedback data, status).

SERCOS is a unidirectional serial communications protocol for connecting multiple drives (Slaves) to a motion controller (Master) over a fiber optic ring, in an industrial environment. The control and status information is organized into telegrams, and travels in a serial data stream around the SERCOS ring. All messages are synchronized according to the SERCOS cycle time, the timing of which is configured during initialization by the Master. Each Slave on the ring repeats the telegrams sent to it, sending them to the next device on the ring, and inserting its own telegram into the designated time slot in the serial data stream.

Starting at the output port of the Master, the devices are connected in the ring in a daisy-chain fashion, connecting from the output port of one device to the input port of the next device, and so on, until the ring is closed back at the input port of the Master. Up to 254 drives can be connected on a SERCOS ring, although the systems requirements for update rates and data will usually limit the number of devices to many fewer than that. SERCOS networks can operate at 2, 4, 8 or 16 Mbit/sec. Maximum distances from input port to output port can be 60 meters (plastic fiber) to 250 meters (glass fiber).



Operation Modes

The SERCOS communication interface supports three main operation modes (Torque, Velocity and Position). The operation mode defines the real-time digital messages sent between the controller and the drive(s).

| Mode | To Drive | To Controller |
|-----------------|-------------------------|------------------------|
| <i>Torque</i> | 16 bit Command Torque | 32 bit Actual Position |
| <i>Velocity</i> | 32 bit Command Velocity | 32 bit Actual Position |
| <i>Position</i> | 32 bit Command Position | 32 bit Actual Position |

In addition to the main operation modes SERCOS supports several variations. Since the communication interface is determined by the firmware/software in the controller and the firmware/software in the drive, the real-time data is configurable.

Currently the SERCOS/XMP Series firmware/software supports several operation modes. Some operation modes are drive specific while others are drive independent. Motion Engineering is constantly testing and certifying compatibility between our controller and SERCOS-compatible drives.

Most drives support the three main operation modes (Torque, Velocity and Position). Please consult your drive specific documentation regarding supported operation modes and variations.

In all modes, the controller calculates a 32-bit command position every sample. The command position is based on the current command jerk, command acceleration and command velocity.

In Torque mode, the controller sends a 16-bit command torque to the drive. The drive sends a 32-bit actual position back to the controller. Every sample, the controller calculates a new command torque based on the position error and the control algorithm. The controller closes the position and velocity loop and the drive closes the torque loop.

In Velocity mode, the controller sends a 32-bit command velocity to the drive. The drive sends a 32-bit actual position back to the controller. Every sample, the controller calculates a new command velocity based on the position error and the control algorithm. The controller closes the position loop, and the drive closes the velocity and torque loop.

In Position mode, the controller sends a command position to the drive. The drive sends a 32-bit actual position back to the controller. Every sample, the controller calculates a new command position. The drive closes the position, velocity and torque loop.



Closed-Loop Tuning

A general difference between SERCOS digital drives and conventional analog drives is that digital drives have on-board intelligence and can close position or velocity loops within the drive. In all operation modes, “tuning” requires setting parameters in the drive and the controller. Thus, an understanding of both the controller and drive control algorithm is necessary for successful drive tuning.

The controller tuning parameters can be set using Motion Console (for Windows-based systems). For more information on the controller’s tuning procedures, please consult the Tuning section in the *XMP Motion Controller Hardware Installation* manual.

The drive’s tuning parameters must be set via IDNs, and the values are determined from information supplied by the drive manufacturer. Please consult the drive-specific documentation for more information on the drive’s tuning parameters. While Motion Engineering has considerable experience with the listed drives and can generally offer tuning guidelines for drive parameters, difficult tuning situations may require support from the drive vendor or manufacturer.

In Velocity mode, the controller's CONTROL ALGORITHM output controls the motor's velocity. Therefore, the drive's velocity loop must be tuned and the controller's position loop must be tuned. The tuning procedure is identical to a standard analog output XMP Series controller connected to a velocity-controlled amplifier. Note that the controller's Velocity Feed Forward term is very useful in velocity-controlled systems.

In Position mode, the controller's control algorithm is not used. The drive is responsible for the closed loop control. Therefore, the drive's velocity and position loop must be tuned. The controller's tuning parameters have no effect on the system's response.



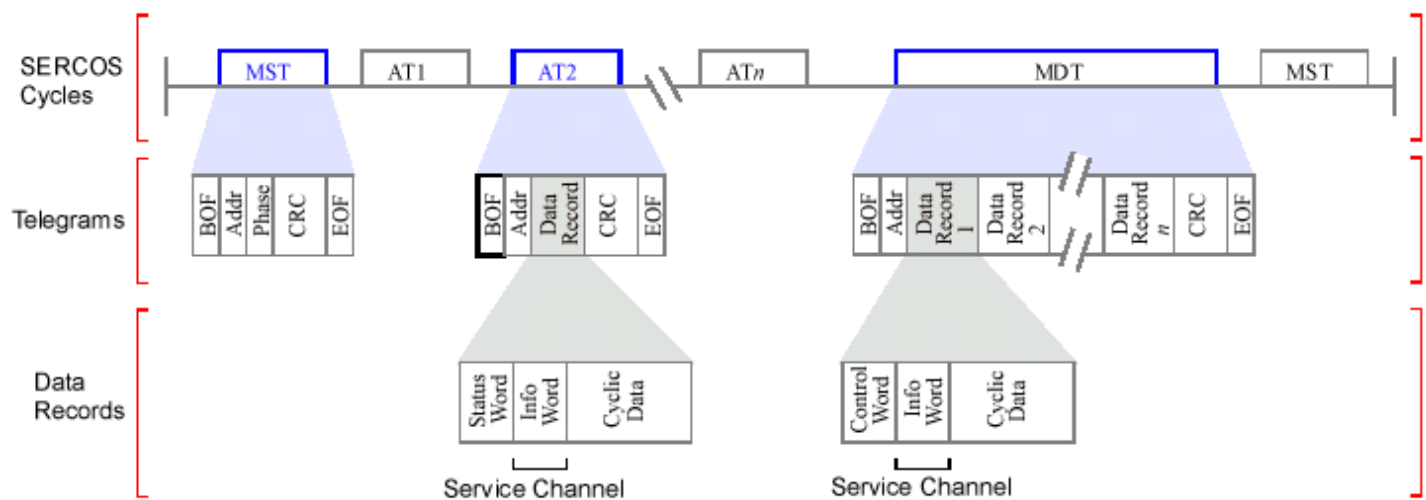
Data Transmission

SERCOS supports two types of data transmission, cyclic and non-cyclic.

Cyclic data is the critical real-time synchronized data sent between the Master and the Slaves (drives, I/O modules). In every SERCOS cycle, the Master sends and receives fixed-length messages to the drives and I/O modules. These messages contain the motion control command signal and the feedback response for each drive or the digital I/O commands and responses for each I/O module. The cyclic data is guaranteed to reach each drive and I/O module and return to the Master at a fixed time interval, the SERCOS cycle.

Non-cyclic data is the noncritical asynchronous data. The cyclic fixed-length messages have space (called the Service Channel) reserved for non-cyclic data. In each SERCOS cycle, the Master may transmit two bytes of non-cyclic data through the Service Channel to each Slave. Note that it may require several SERCOS cycles for the Master to complete the transmission of the non-cyclic data to the Slaves. Typically, transmitting non-cyclic data is much slower than cyclic data.

SERCOS cycles are built using telegrams, which in turn contain data records for all of the Slave drives. Cyclic data is transferred in the cyclic data part of the data records. Non-cyclic data is transferred in the Service Channel of data records. Refer to the next figure.





NEXT

Copyright © 2002
Motion Engineering

Sercos- Data Types

[Summary](#) | [Data Block Structure](#) | [Data Block Structure of IDNs](#)

- [Element 1](#): IDNumbers
- [Element 2](#): Name of Operation Data
- [Element 3](#): Attributes of Operation Data
- [Element 4](#): Operation Data Unit
- [Element 5](#): Minimum Input Value of Operation Data
- [Element 6](#): Maximum Input Value of Operation Data
- [Element 7](#): Operation Data

Summary

And datas in a slave that can be accessed by the host is represented by a data block. Data blocks are assigned identification numbers (IDNs or “ID numbers”). The data block consists of information about the data; for example, IDNumber, name, attributes, units, min and max values, and the operation data itself.



Data Block Structure

The SERCOS protocol is designed to handle many different types of data, characterized by two fundamental types: fixed length data and variable length data.

Fixed length data is either 2 bytes or 4 bytes wide and can be used to represent signed or unsigned integers, hexadecimal values, binary codes, IDNumbers (identification numbers) of other Data Blocks, and procedure commands.

The length of **variable length** data depends on what type of data is present, and is defined by the first two words (32 bits), which specify the actual and maximum length of the data. Variable length data can be used to represent character strings, lists of IDNumbers of Data Blocks, lists of signed or unsigned integers (both 2 or 4 bytes wide), lists of hexadecimal values, etc.

All data (fixed and variable length) can be sent or received via the Service Channel. However, **only** fixed length data is allowed to be configured into a communication telegram (MDT, AT). When a communication telegram is configured to send or receive fixed length data, it is only Element 7 data that will be either sent or received.



Data Block Structure of IDNs

Data is accessed through data blocks referred to as IDNs. An IDN consists of seven elements:

| | |
|------------------|----------------|
| Element 1 | IDNumber |
| Element 2 | Name |
| Element 3 | Attribute |
| Element 4 | Unit |
| Element 5 | Minimum Input |
| Element 6 | Maximum Input |
| Element 7 | Operation Date |

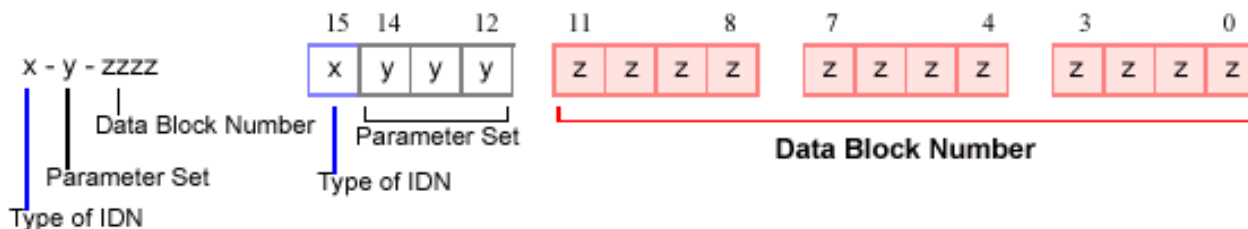
All data exchanged between Master and Slaves has an IDNumber (IDN number) assigned to it. Every IDN has an associated data block which consists of seven elements. The Master can only write Element 7 data; the Master cannot write Elements 1 - 6. Elements 1 - 6 are defined by the drive itself.

| Element | Description | Data Type |
|----------|-------------------------------------|--|
| 1 | IDNumber (Identification number) | binary Expressed as either S-X-XXXX or P-X-XXXX. S denotes IDNs that are defined by the SERCOS Specification. P denotes IDNs that are defined by the manufacturer of the device. |
| 2 | Name | variable length string Contains the name of the IDN. |
| 3 | Attribute | 32 bit binary Contains information about conversion factors and data representation (signed or unsigned integer, fixed or variable length data, etc.) |
| 4 | Unit | variable length string Contains a representation of the units for the data. |
| 5 | Minimum input value | 1 or 2 words |
| 6 | Maximum input value | 1 or 2 words |
| 7 | Operation Data | 1 or 2 words, or string |



Element 1: IDNumbers

Each Data Block has a number assigned to it for identification purposes, called the IDNumber. The IDNumber is represented as either S-X-XXXX or P-X-XXXX. S denotes a Data Block that is defined by the SERCOS Specification. P denotes a Data Block that is defined by the manufacturer. The first 'X' identifies the "data set" that the Data Block belongs to. According to the SERCOS Specification, it is possible to switch between data sets.

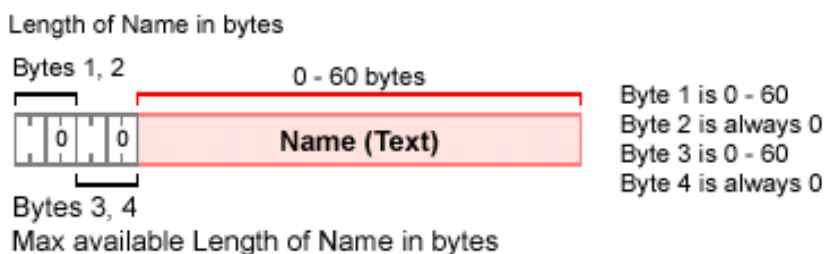


| Bits | Name | Values |
|-------|-------------------|---|
| 15 | Type of IDN | 0 S - Standard data 1 P - Product-specific data as the "S" or "P" part of the IDN notation S-y-zzzz or P-y-zzzz. |
| 14-12 | Parameter Set | 0 - 7, as the "y" part of the IDN notation S-y-zzzz or P-y-zzzz. |
| 11-0 | Data Block Number | 0 - 4095, as the "zzzz" part of the IDN notation S-y-zzzz or P-y-zzzz. |



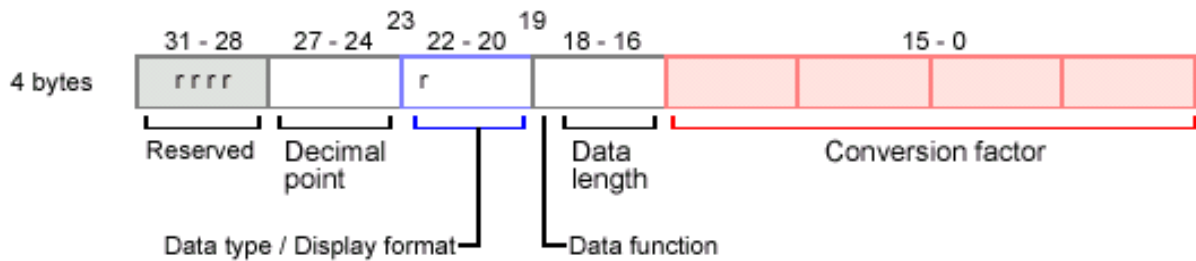
Element 2: Name of Operation Data

The name of operation data is 64 bytes maximum, with 2 length specifications of 2 bytes each, and a character string of 60 characters maximum. Bytes 1 and 2 contain the number of characters in the text. Bytes 3 and 4 contain the maximum number of characters in the text. Since this element is READ-Only, bytes 3 and 4 will contain the same values as bytes 1 and 2.



Element 3: Attributes of Operation Data

Every data block has an attribute (4 bytes) which contains all of the information required to display operation data, using universal routines. If data needs to be scaled (to be displayed), then specific scaling parameters are supplied in the attribute.



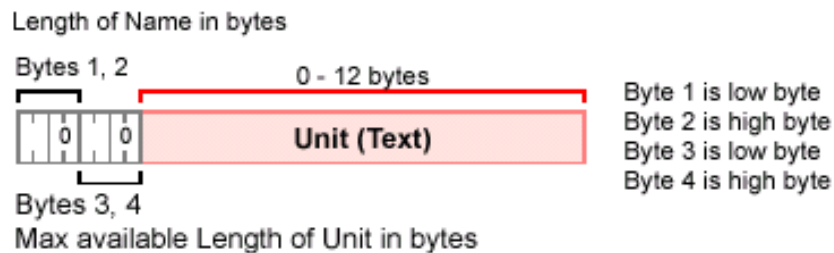
| Bits | Name | Values | | | | | | | | | | | | | | | | | | |
|-----------------------|---|---|-----------|----------------|-------------------|--------|----------------------|------------------|-------------|----------------|----------------------|-------------|-----------------------|------|----------------------|------------|--------------|--|--------------|--|
| 31 | Reserved | | | | | | | | | | | | | | | | | | | |
| 30 | Phase 4 write-protection bit. | 1 = write-protected | | | | | | | | | | | | | | | | | | |
| 29 | Phase 3 write-protection bit. | 1 = write-protected | | | | | | | | | | | | | | | | | | |
| 28 | Phase 2 write-protection bit. | 1 = write-protected | | | | | | | | | | | | | | | | | | |
| 27-24 | Number of places after the decimal point Indicates the position of the decimal point in the data to be displayed. Basically, it's the exponent "x" in 10-x. | 000 No places after decimal point 001 1 place after decimal point 010 2 places after decimal point * 111 15 places after decimal point | | | | | | | | | | | | | | | | | | |
| 23 | Reserved | | | | | | | | | | | | | | | | | | | |
| 22-20 | Data Type & Display Format Used to convert the operation data, and min/max input values to the correct display format. | <table border="1"> <thead> <tr> <th>Data Type</th> <th>Display Format</th> </tr> </thead> <tbody> <tr> <td>000 binary number</td> <td>binary</td> </tr> <tr> <td>001 unsigned integer</td> <td>unsigned decimal</td> </tr> <tr> <td>010 integer</td> <td>signed decimal</td> </tr> <tr> <td>011 unsigned integer</td> <td>hexadecimal</td> </tr> <tr> <td>100 extended char set</td> <td>text</td> </tr> <tr> <td>101 unsigned integer</td> <td>IDN number</td> </tr> <tr> <td>010 Reserved</td> <td></td> </tr> <tr> <td>111 Reserved</td> <td></td> </tr> </tbody> </table> | Data Type | Display Format | 000 binary number | binary | 001 unsigned integer | unsigned decimal | 010 integer | signed decimal | 011 unsigned integer | hexadecimal | 100 extended char set | text | 101 unsigned integer | IDN number | 010 Reserved | | 111 Reserved | |
| Data Type | Display Format | | | | | | | | | | | | | | | | | | | |
| 000 binary number | binary | | | | | | | | | | | | | | | | | | | |
| 001 unsigned integer | unsigned decimal | | | | | | | | | | | | | | | | | | | |
| 010 integer | signed decimal | | | | | | | | | | | | | | | | | | | |
| 011 unsigned integer | hexadecimal | | | | | | | | | | | | | | | | | | | |
| 100 extended char set | text | | | | | | | | | | | | | | | | | | | |
| 101 unsigned integer | IDN number | | | | | | | | | | | | | | | | | | | |
| 010 Reserved | | | | | | | | | | | | | | | | | | | | |
| 111 Reserved | | | | | | | | | | | | | | | | | | | | |
| 19 | Function of Operation Data Indicates whether this data calls a procedure in a drive | 0 Operation data or parameter 1 Procedure command | | | | | | | | | | | | | | | | | | |

| | | |
|--------------|--|---|
| 18-16 | Data Length The data length is required so that the Master is able to complete Service Channel data transfers correctly. | 000 Reserved 001 Operation data is 2 bytes long 010 Operation data is 4 bytes long 011 Reserved 100 Variable length with 1-byte data strings 101 Variable length with 2-byte data strings 110 Variable length with 4-byte data strings 111 Reserved |
| 15-0 | Conversion Factor | An unsigned integer used to convert numeric data to display format. Is set to 1 when it is not needed for data display (e.g., binary display or a character string). |



Element 4: Operation Data Unit

The operation data unit is 16 bytes maximum, with 2 length specifications of 2 bytes each, and a character string of 12 characters maximum. Bytes 1 and 2 contain the number of characters in the text. Bytes 3 and 4 contain the maximum number of characters in the text. Since this element is READ-Only, bytes 3 and 4 will contain the same values as bytes 1 and 2.



Element 5: Minimum Input Value of Operation Data

The minimum input value is the smallest numerical value for operation data that the drive can process. When the Master writes a value to the drive that is less than the minimum value, the drive ignores it and continues to use the previous operation data.

When the operation data is of variable length or a binary number is used, there is no minimum input value of operation data.



Element 6: Maximum Input Value of Operation Data

The maximum input value is the largest numerical value for operation data that the drive can process. When the Master writes a value to the drive that is more than the maximum value, the drive ignores it and continues to use the previous operation data.

When the operation data is of variable length or a binary number is used, there is no maximum input value of operation data.

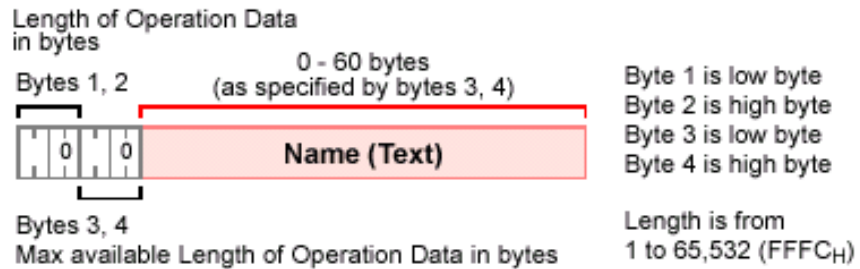


Element 7: Operation Data

In terms of length, there are 3 types of operation data:

- fixed length with 2 bytes
- fixed length with 4 bytes
- variable length up to 65,532 bytes in 1 byte (char), 2 byte, or 4 byte values

Bytes 1 and 2 contain the number of bytes in the text. Bytes 3 and 4 contain the max number of bytes available in the text. Data in the text may be 1 byte, 2 bytes, or 4 bytes wide.



[Introduction](#) | [Overview](#) | [Data Types](#) | [Communications](#) | [Procedures](#) | [Telegrams](#) | [Topologies](#)



Copyright © 2002
Motion Engineering

Sercos- Communications

[Synchronization](#) | [Ring Timing](#) | [Initialization](#)

Synchronization

The Master is responsible for sending a synchronization telegram (MST) at the beginning of each communication cycle. All Slaves (drives) will receive the MST and reset their clocks. In this way, all Slaves will run in phase lock with the Master's clock. Because all Slaves are in phase lock with the Master's clock, commands can be made active in all of the Slaves at the same instant. This means that the Master can coordinate motion between all axes without propagation effects distorting the motion profile. Feedback from the Slaves is handled in a similar manner, and is latched in all Slaves at the same instant.

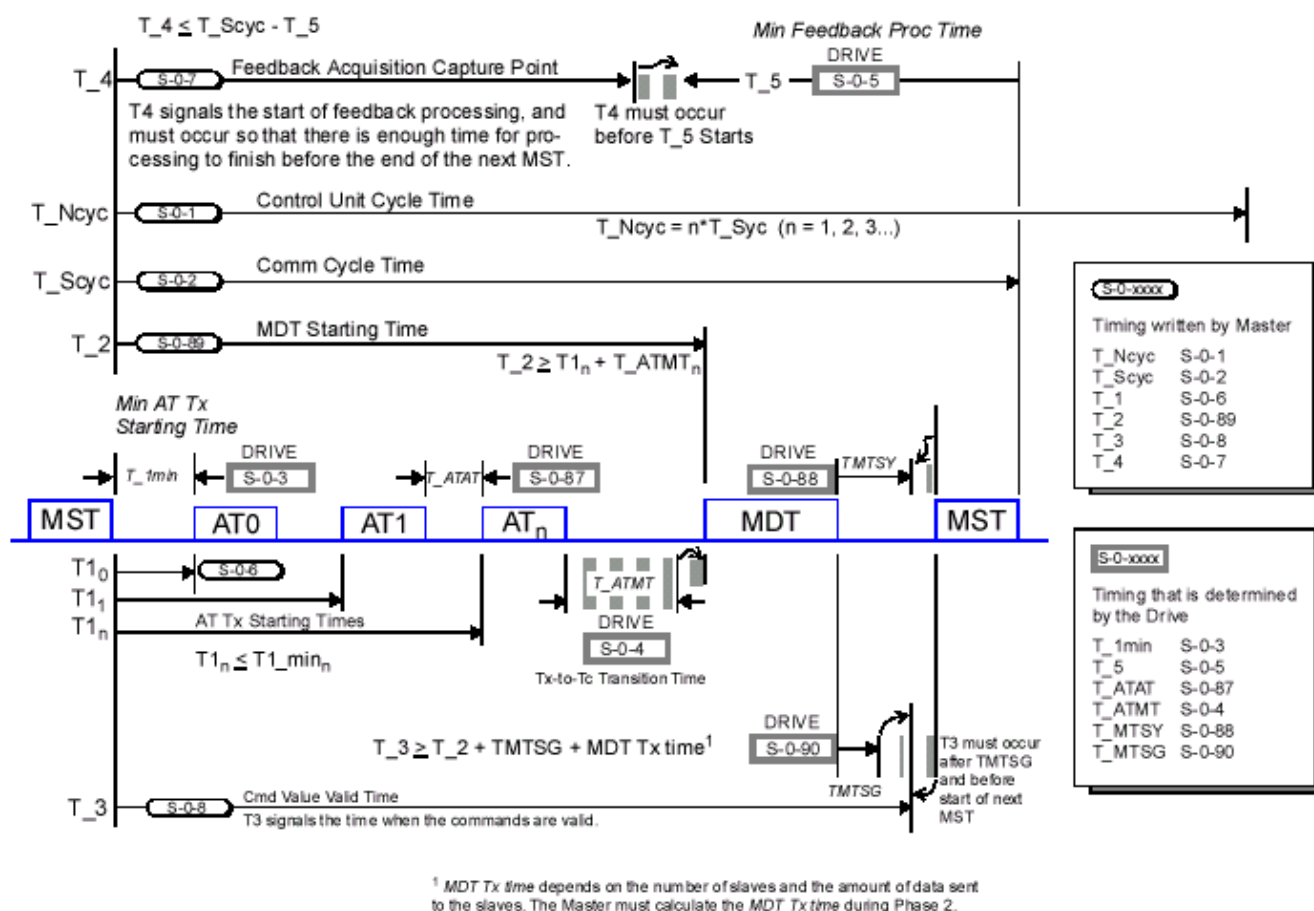


Ring Timing

Data is sent and received by the Master and Slaves through communication telegrams. The communication telegrams are organized over the SERCOS cycle in the manner shown in the next figure.

The SERCOS ring timing is based on the data to be placed in the telegrams (AT, MDT), and on 6 drive parameters that are determined by the type and features of a drive (or drives), and 6 parameters that are written from the Master, with some of these parameters derived or calculated from the drive's timing parameters. The times at which the MDT and AT are sent (relative to the sending of the MST) are determined by the Master and sent to the Slaves during initialization.

During Phase 2, the Master reads parameters from the drives that determine what and when the drives are able to transmit and receive. Using this information and the desired telegram contents, baud rate and cycle time, the Master determines the timing and telegram parameters for each drive. The Master then writes these parameters to the drives.



For a print-friendly version of the above diagram, [click here](#).



Initialization

Before true synchronous data transmission can occur, the system must first be initialized. This is done through a series of **communication phases** (or just phases) in which data is first transmitted asynchronously. The data transmitted during these early phases is used to configure the Master and Slaves for synchronous data transmissions in later phases. The SERCOS protocol defines five phases.

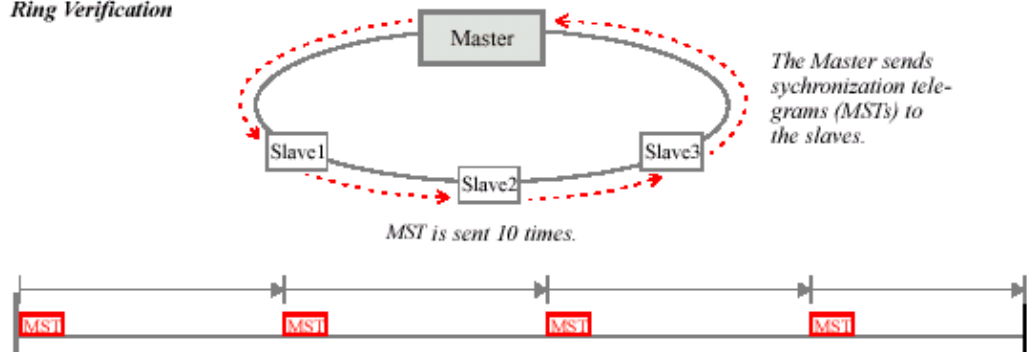
| Phase | Name | Action |
|-------|-------------------------|--|
| 0 | Ring Verification | Master verifies ring closure |
| 1 | Device Verification | Master verifies devices on ring |
| 2 | Telegram Set-Up | Master reads timing data from slaves and sets up telegram timing |
| 3 | Device Parameterization | Master continues to configure devices |
| 4 | Cyclic Operations | Master commands devices cyclically |

On power-up, each drive or I/O module begins an initialization sequence. At this time, each drive and I/O module operates as a repeater, by simply passing received telegrams to the next device on the SERCOS ring.

The Master is only allowed to set the communication phase to the next logical communication Phase (**Phase 0 -> Phase 1 -> Phase 2 -> Phase 3 -> Phase 4**) or directly back to **Phase 0**. If at any time the Master attempts to switch a Slave into a Phase that is not the next logical Phase, then the Slave will immediately return to Phase 0. If at any time the Slave receives two invalid MSTs or MDTs consecutively, the Slave will also switch to Phase 0.

Phase 0:

Ring Verification

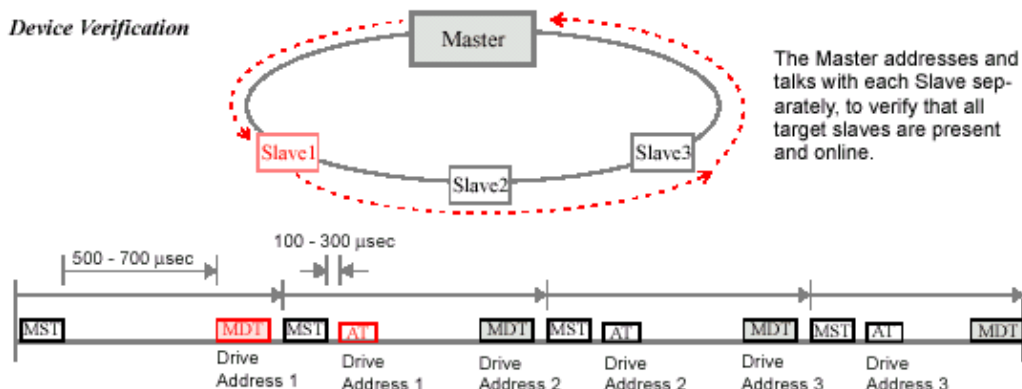


During **Phase 0 no data is exchanged** between the Master and the Slaves. All Slaves must be in “repeater” mode. This means that each Slave will retransmit any signal that it receives. In order to verify that the communications ring is intact and capable of sending telegrams, the Master begins sending Master Synchronization Telegrams (MST) through the SERCOS ring.

The Slaves (drives) simply pass the MST to the next drive in the daisy-chained ring, and eventually because of the ring topology, the MST returns to the Master (i.e., Master will receive its own MST). Phase 0 is completed when the Master receives 10 consecutive MSTs. After the tenth consecutive MST, the Master changes the phase information in the MST to 1, which commands all Slaves to switch into Phase 1 operations.

Phase 1:

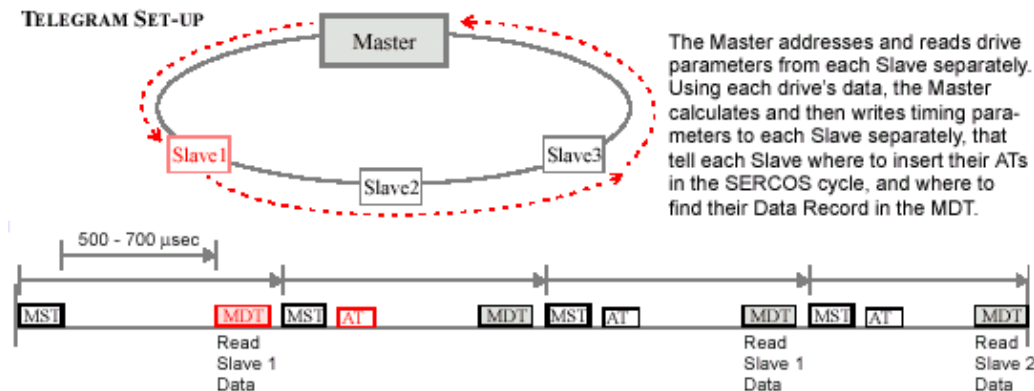
Device Verification



During **Phase 1**, the Master sends out an MDT with the address of a specific Slave in the system. If present within the system, the Slave with the specified address responds by sending an AT back to the Master. The AT that is sent is rudimentary and is intended solely as a confirmation that the addressed Slave is in the system. The Master then repeats this query for all target Slaves. (Note that not all Slaves in the system will be target Slaves. Target Slaves are chosen by the application before initialization begins.)

When the Master receives an AT from each target Slave (in response to a query), the Master changes the phase information in the MST to 2, which commands all Slaves to switch into Phase 2 operations.

Phase 2:



During **Phase 2**, the Master sets up the configurable data portion and calculates the duration and time slots within the SERCOS cycle for all telegrams to be used in Phases 3 and 4. The Master also determines the slave operation mode for all Slaves. In order to do this, the Master requires certain data from the Slaves. The Master obtains this data by sending an MDT addressed to a specific Slave, that uses the MDT's Service Channel to query the Slave for the required data

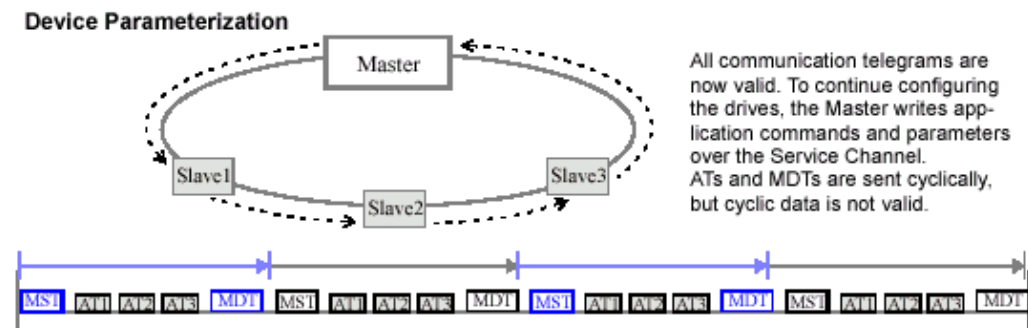
The Slave responds by sending an AT containing the appropriate data, in the AT's Service Channel. Once the Master has determined all parameters, it sends them to each Slave via the MDT's Service Channel. In Phase 2, the Service Channel is active in both MDTs and ATs.

Once all data is transmitted to the Slaves, the Master will initiate the Communication Phase 3 Transition Check S-0-127 procedure for each Slave.

The Communication Phase 3 Transition Check checks the validity of all the data, and if all the data is valid, then the procedure executes successfully. If any data is not valid, the procedure fails and the IDN number of the invalid data is placed in IDN-List of Invalid Operation Data for CP2 S-0-21.

After all Slaves have completed the procedure successfully, the Master changes the phase information in the MST to 3, which commands all Slaves to switch into communication Phase 3.

Phase 3:



In **Phase 3**, the real-time synchronous and asynchronous communication starts, and the Master uses the Service Channel to configure and parameterize (write parameters to) the Slaves. The parameters set in Phase 3 are application-oriented (e.g., conversion factors). During Phase 3, all communication telegram parameters sent in Phase 2 become active. Note that although Configurable Data (in ATs and MDTs) is present in the communication telegrams, some of that Configurable Data may not be valid until Phase 4.

The Master will still send the MST at the beginning of the SERCOS cycle, but will now also send an MDT with a global address at a specific time (all Slaves will receive a global telegram). Each Slave will transmit its AT during its specified time slot. The communication telegrams now contain the Control/Status Word, a Service Channel, and the Configurable Data.

When the Master has finished sending parameters to the Slaves, it will initiate the Communication Phase 4 Transition Check S-0-128 procedure for each Slave.

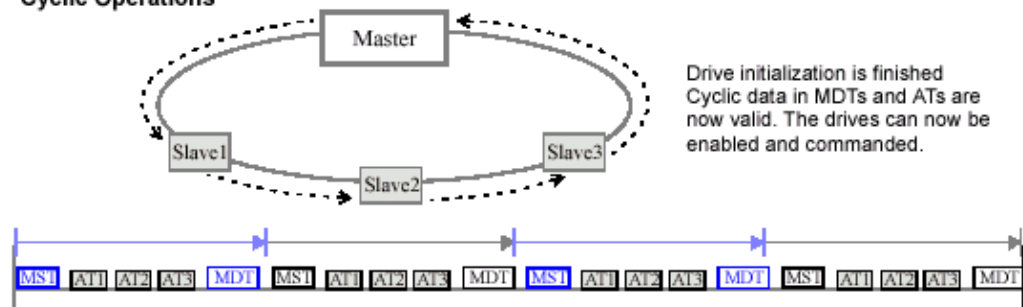
The Communication Phase 4 Transition Check checks the validity of all the data, and if all the data is valid, then the procedure executes successfully. If any data is not valid, the procedure fails and the IDN number of the invalid data is placed in IDN-List of Invalid Operation Data for CP3 S-0-22.



If the Communication Phase 4 Transition Check executes successfully and the Phase information in the MST is equal to 4, the Master switches the drives to Phase 4.

Phase 4:

Cyclic Operations



In **Phase 4**, a final verification of error-free drive operation is completed. This completes the drive initialization. The SERCOS communication loop is now operational. During Phase 4, all Control/Status words, Service Channels, and Configurable Data (in ATs and MDTs) are valid for the target Slaves. The Slaves (drives) are ready to follow commands when enabled. Diagnostics (errors, warnings, status) are enabled.



Sercos- Procedures

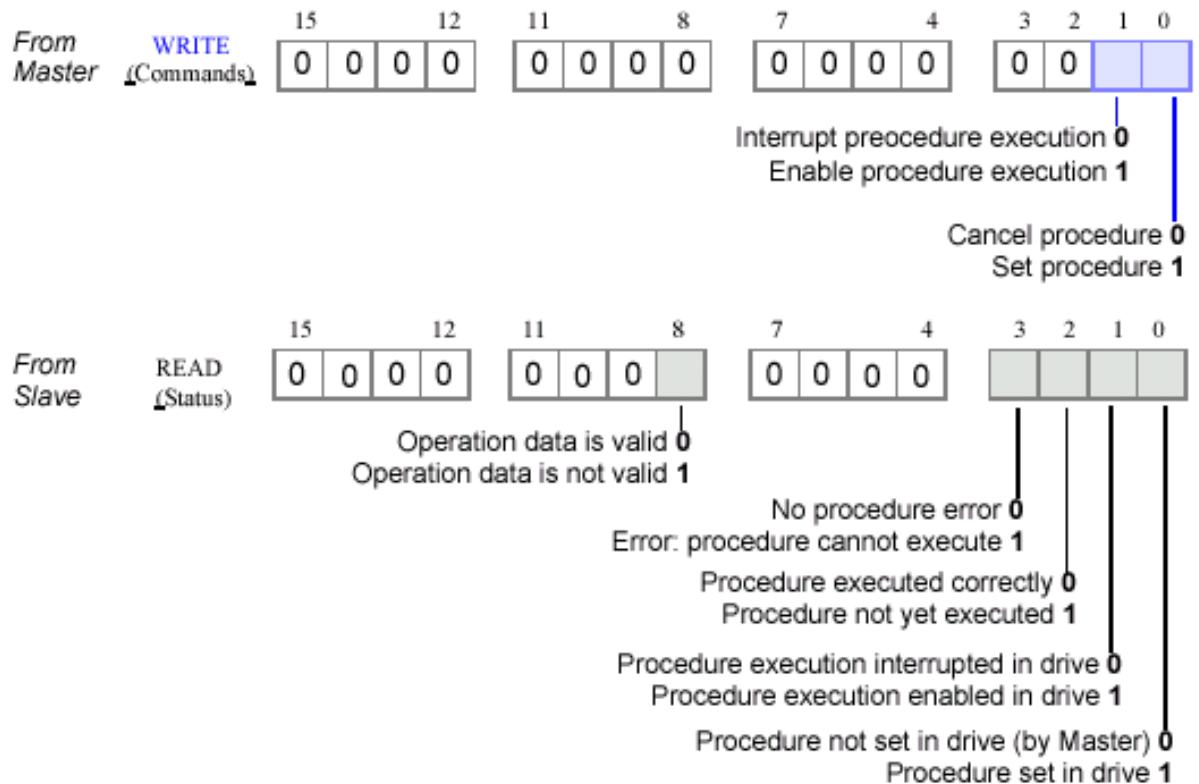
Summary

Many Slaves (drives) come with preprogrammed procedures that the Slave is able to execute without assistance from the Master. Each of these procedures is assigned a Data Block. The Master controls a procedure by reading and writing to Element 7 of the procedure's Data Block. The Master can initiate, interrupt, or cancel a procedure at any time by setting or clearing bits in Element 7 of the procedure's Data Block.

Bits 0 and 1 are responsible for respectively setting and enabling the procedure command. Once set and enabled, the Slave will begin execution of the procedure. To indicate that the procedure is executing, the Slave sets bit 2 of the procedure status word. When the procedure has finished executing, the Slave clears bit 2 of the procedure status word and sets the Procedure Command Change Bit in the Drive Status Word S-0-134. If an error has occurred when executing the procedure, the Slave will set bit 3 of the procedure status word and will also set the Procedure Command Change Bit in the Drive Status Word S-0-135. In either case (successful execution or failure), the Master must cancel the procedure by clearing bits 0 and 1 of the procedure.

In SERCOS, in order for a Master to execute a procedure,

1. The Master writes 0x3 to Element 7 of the desired IDN procedure, which sets and enables the procedure.
2. The Master reads the procedure status word and checks bits 2 & 3 to see if the procedure has finished executing or if there is an error.
3. After reading that the procedure has executed, the Master writes 0x0 to Element 7 of the particular procedure, which cancels the procedure.



[Introduction](#) | [Overview](#) | [Data Types](#) | [Communications](#) | [Procedures](#) | [Telegrams](#) | [Topologies](#)



NEXT

Copyright © 2002
Motion Engineering

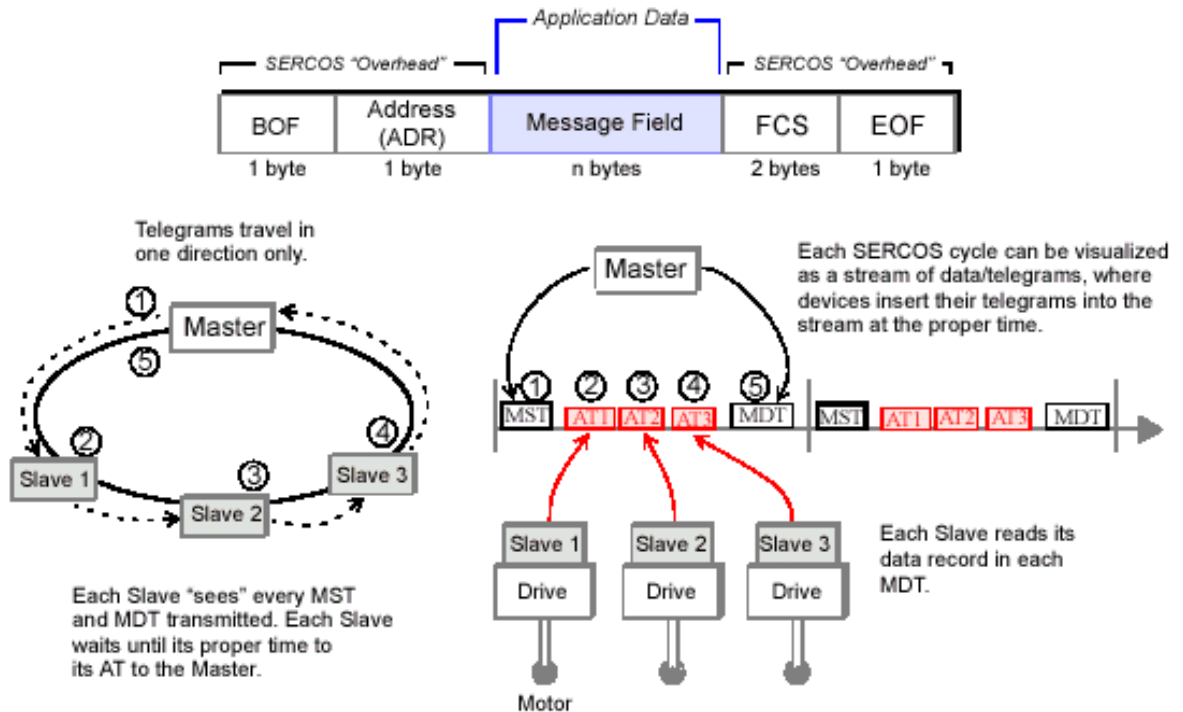
Sercos- Telegrams

[Summary](#) | [BOF Delimiter](#) | [ADR Target Addresses](#) | [Message Field](#)
[FCS \(Frame Check Sequence\)](#) | [EOF \(End of Frame\) Delimiter](#) | [Data Record](#)
[Master Synchronization Telegram](#) | [Master Data Telegram](#) | [Amplifier Telegram](#)

Summary

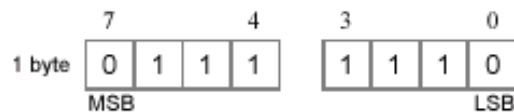
The controller (Master) communicates with the drives and I/O modules (Slaves) using telegrams. A telegram is a structure that contains data, error checking and handshaking information. SERCOS supports three types of telegrams: Master Synchronization Telegram (MST), Master Data Telegram (MDT), and Amplifier Telegram (AT).

Each type of telegram contains 5 types of fields: BOF (beginning of frame), ADR (address), message field, FCS (frame check sequence), EOF (end of frame).



BOF Delimiter

All telegrams have a BOF (beginning of frame) byte denoting the beginning of the telegram. The BOF is always 0111 1110.



ADR Target Addresses

All telegrams have an ADR (address) byte, which denotes the address of a drive (Slave). In a telegram from the Master, the address specifies which drive the information is for. In a telegram from a Slave (drive), the address specifies the sourcing drive. The target addresses for the drives are valid if greater than 0 and less than 255. Typically, a drive's address is set using a selector located physically on the drive.

Address 0 is the "no station" address, and is sometimes used to remove a drive from the ring logically, during troubleshooting. During non-cyclic operations (Phases 0 - 2), the Master can only communicate with one drive per cycle. During cyclic operations (Phases 3, 4), the Master can communicate with all drives.

| From | To | Telegram | Non-Cyclic (Phases 0,1,2) | Cyclic (Phases 3,4) |
|--------|--------|------------|------------------------------|------------------------------|
| Master | Slave | MST | 255 | 255 |
| Master | Slave | MDT | $1 \leq \text{ADR} \leq 255$ | 255 |
| Slave | Master | AT | $1 \leq \text{ADR} \leq 254$ | $1 \leq \text{ADR} \leq 254$ |



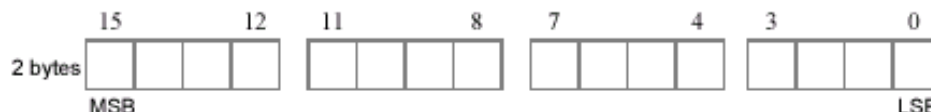
Message Field

Each Telegram (MST, AT, and MDT) contains a message field. The Message Field for the MST consists of one 8-bit word. The lower three bits of the MST contain Phase information. The Message Field for the AT consists of one Data Record. The Message Field for the MDT consists of one Data Record if the MDT's target address is a specific drive. If the MDT's Target address is all drives (255), then the Message Field for the MDT consists of one Data Record per drive in the system.



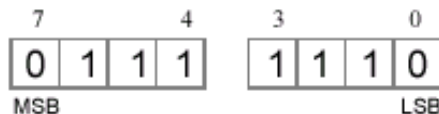
FCS (Frame Check Sequence)

All telegrams have a two-byte FCS number used to check data integrity. The frame check sequence (16 bits) is implemented according to ISO/IEC 3309, 4.5.2.



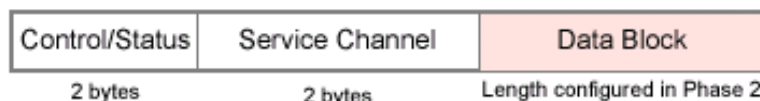
EOF (End of Frame) Delimiter

All telegrams have an EOF byte denoting the end of the telegram. The EOF is always 0111 1110.



Data Record

Data Records are used by both the Amplifier Telegram (AT), and the Master Data Telegram (MDT) to send data. Generally, a Data Record consists of a 16-bit Control/Status word, a 16-bit Service Channel, and a Data Block of 16-bit words.



Amplifier Telegrams contain only one Data Record, because each Slave sends its own Amplifier Telegram.

During Phase 2, the Master sends a Master Data Telegram that is addressed to a specific Slave. Since only one Slave is to receive the MDT, the Phase 2 MDT has only one Data Record. During Phases 3 and 4, the Master sends a Master Data Telegram that is received by all Slaves on the ring. Since all Slaves receive the Master Data Telegram, the Phase 3-4 MDT has one Data Record for each Slave.

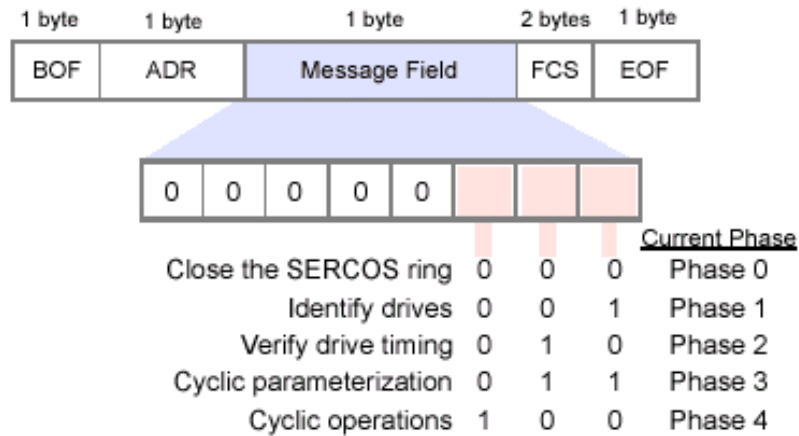
During Phase 2, the Data Block in both the AT and the MDT has a length of 0. This means that all data exchange between Master and Slave must take place using the Service Channel. Because the Service Channel is only 16-bits wide, data exchange can take multiple cycles. Service Channel exchange of data is referred to as **non-cyclic**.

During Phase 2, the Master is responsible for determining the data fields for the Data Blocks in the Phase 3-4 AT and MDT. The size and number of data fields will determine the size of the Data Block. Once defined in Phase 2, the fields and therefore the size of the Data Block is fixed. The data fields in the MDT usually contain command data. The data fields in the AT usually contain feedback data. Because the Data Blocks are configured in Phase 2, there is no overhead using the Data Blocks to send data in Phases 3 and 4. Data exchange using the Data Blocks is referred to as **cyclic**.



Master Synchronization Telegram

The Master Controller uses master synchronization telegrams (MST) to coordinate its transmission cycle timing with the Slaves. The Master initiates a SERCOS cycle by transmitting an MST to all of the drives and I/O modules on the ring. The MST message field contains one 8-bit word, of which the three lowest bits determine the communication phase of the system. The MST is sent at the beginning of the SERCOS cycle in all communication phases.

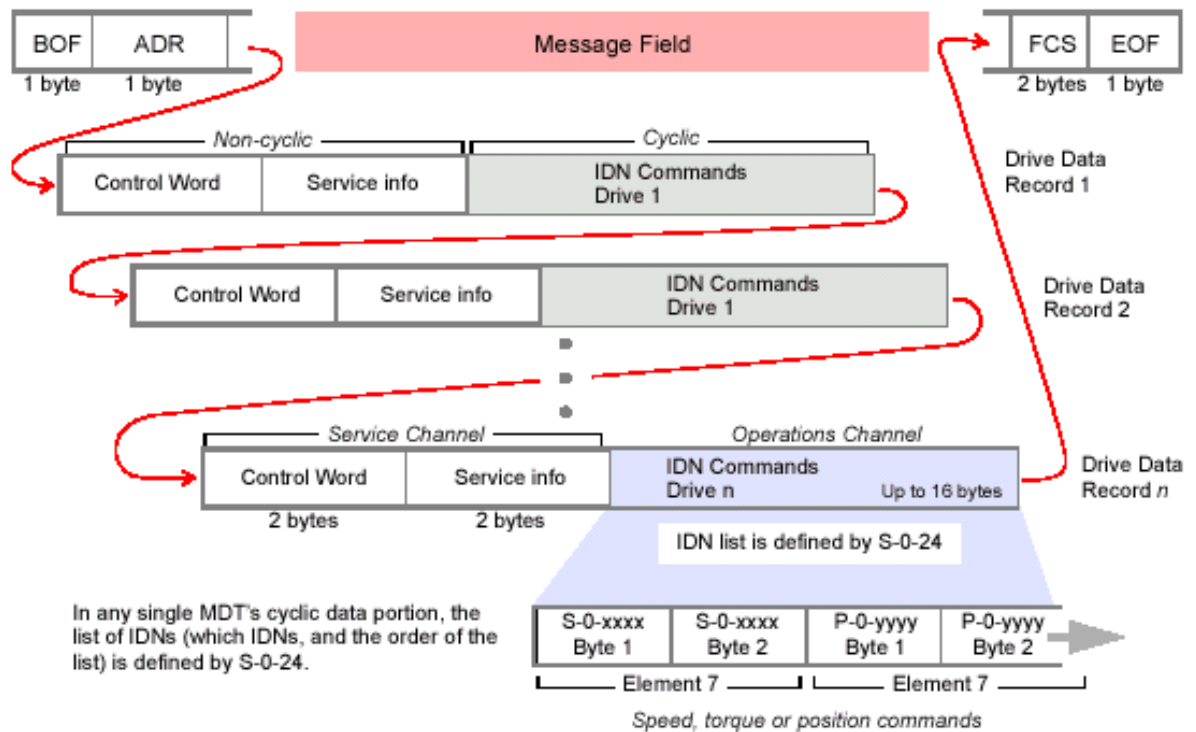


Master Data Telegram

During Phase 2, the Master must communicate with the Slaves in order to configure them for operations in Phases 3 and 4. In order to send or request data, the Master will send a Master Data Telegram (MDT) to a specific Slave (the Slave is addressed explicitly). Since only one Slave is to receive the MDT, the MDT Message Field contains only one Data Record, where the length of the Data Block (inside the Data Record) is 0, i.e., the data block is empty. During Phase 2, the Master sends data to each Slave via the Service Channel.

During Phase 2, the Master informs the Slave of the byte offset into the MDT at which the Data Record resides for that Slave. The length (in bytes) of the Data Block within each Data Record depends on the data fields configured by the Master during Phase 2. The data fields within the Data Block usually contain command information.

During Phases 3 and 4, once per SERCOS cycle the Master sends a Master Data Telegram (MDT) that has a **global** (broadcast) address. Because the global address is used, all Slaves receive the MDT. Since all Slaves will receive the MDT, the MDT Message Field contains one Data Record for each Slave in the system. During Phases 3 and 4, data is sent to a Slave by using the Data Blocks (cyclic) or the Service Channel (non-cyclic).



| Field | Size (bytes) | Description |
|---------------|--------------|---|
| BOF | 1 | Beginning of Frame, which is always 0111 1110. The BOF marks the start of a telegram. |
| ADR | 1 | Target address. In the MDT, ADR = 255 (the broadcast address). |
| *Control Word | 2 | Control word for drive n. Contains operational data. |
| *Service Info | 2 | Contains the non-cyclic data for drive n. |
| *IDN Commands | Variable ** | Contains the cyclic data for drive n. |
| FCS | 2 | Frame check sequence. Contains circular redundancy check (CRC) info. |
| EOF | 1 | End of Frame, which is always 0111 1110. The EOF marks the end of the telegram. |

* These fields comprise the data record. There is one data record per drive in the MDT.

** The length if the IDN Commands field is determined during Phase 2

| Bit | Name & Value | More Detail |
|--------|-----------------------------------|--|
| 15 | 0 Drive OFF 1 Drive ON | Bit 15-13=111, the drive should follow command values. When 1->0, the drive removes torque from the motor, and allows the motor to spin down. |
| 14 | 0 Drive Disable 1 Drive Enable | When 1->0, torque is immediately disabled, independent of bits 15 and 13. |
| 13 | 0 Drive Halt 1 Drive Restart | |
| 12, 11 | Reserved | |
| 10 | Control Unit Synchronization Bit | |

| | | |
|---------|--|--|
| 9, 8 | Operation Mode 00 primary op mode 01 secondary op mode 1 10 secondary op mode 2 11 secondary op mode 3 | Defined by S-0-32 Defined by S-0-33 Defined by S-0-34 Defined by S-0-35 |
| 7 | Real-time Control Bit 2 | S-0-302 |
| 6 | Real-time Control Bit 1 | S-0-300 |
| 5, 4, 3 | Data Block Element 000 Service channel not active, 001 IDN (number) of the op data 010 Name of operation data 011 Attribute of op data 100 Units of op data 101 Min input value 110 Max input value 111 Operation data | -Close service channel or break a transmission in progress -The service channel is closed for the previous IDN and opened for a new IDN. |
| 2 | 0 Transmission in progress 1 Last transmission | |
| 1 | 0 Read service info 1 Write service info | |
| 0 | Master Service Transport Handshake | A toggle bit |

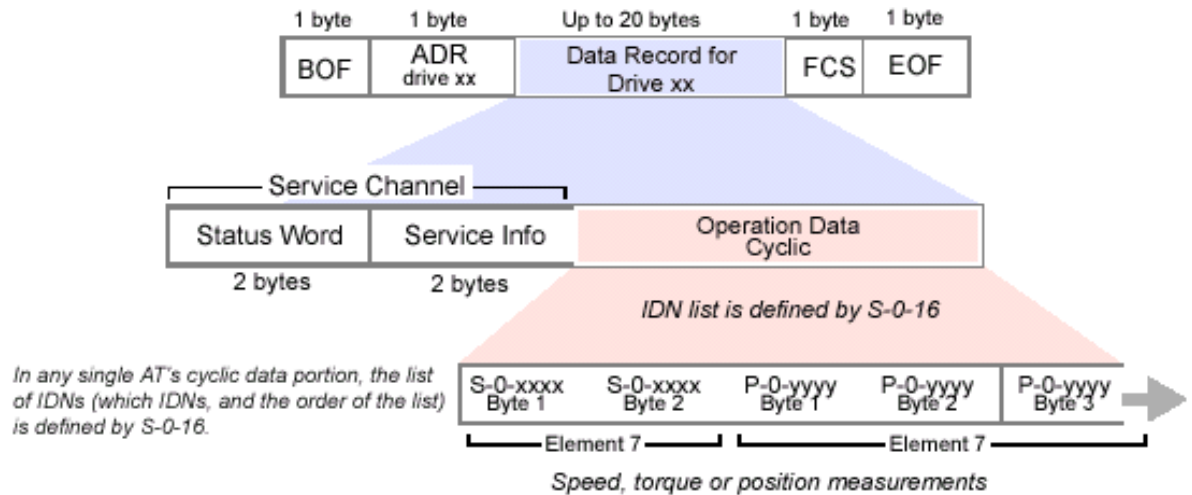


Amplifier Telegram

During Phase 2, the Slave sends an Amplifier Telegram (AT) only when it receives a Master Data Telegram (MDT) that contains that Slave's address. The AT contains one Data Record where the length of the Data Block is 0, i.e., the data block is empty. During Phase 2, the Master sends data to each Slave via the Service Channel.

During Phases 3 and 4, each Slave sends an Amplifier Telegram (AT) every SERCOS cycle, at the time designated by the Master during Phase 2. The AT Message Field contains one Data Record, where the length of the Data Block (inside that Data Record) is determined by the data fields configured by the Master during Phase 2. The data fields within the Data Block usually contain feedback and status information.

During Phases 3 and 4, data is sent to the Master by using the Data Blocks (cyclic) or the Service Channel (non-cyclic).



| Field | Size (bytes) | Description |
|----------------|--------------|---|
| BOF | 1 | Beginning of Frame. The BOF marks the start of a telegram. |
| ADR | 1 | sends address. |
| Status | 2 | Control word for drive n. Contains operational data. |
| Service Info | 2 | Contains the non-cyclic data for drive n. |
| Operation Data | Variable * | Contains the cyclic data for drive n. |
| FCS | 2 | Frame check sequence. Contains circular redundancy check (CRC) info. |
| EOF | 1 | End of Frame, which is always 0111 1110. The EOF marks the end of the telegram. |

* The length of the Operation Data field is determined during Phase 2.

| Bit | Name & Value | More Detail |
|--------|---|--|
| 15, 14 | Ready to operate 00 drive not ready for power-up 01 drive ready for power-up 10 drive power ready 11 drive ready to operate | |
| 13 | Drive Shutdown Error, Class 3 Diags | |
| 12 | Change Bit for Class 2 Diags | |
| 11 | Change Bit for Class 3 Diags | |
| 9,8 | Actual Operation Mode 00 primary op mode 01 secondary op mode 1 10 secondary op mode 2 11 secondary op mode 3 | Defined by S-0-32 Defined by S-0-33 Defined by S-0-34 Defined by S-0-35 |
| 7 | Real-time Status Bit 2 | See S-0-306 |
| 6 | Real-time Status Bit 1 | See S-0-304 |



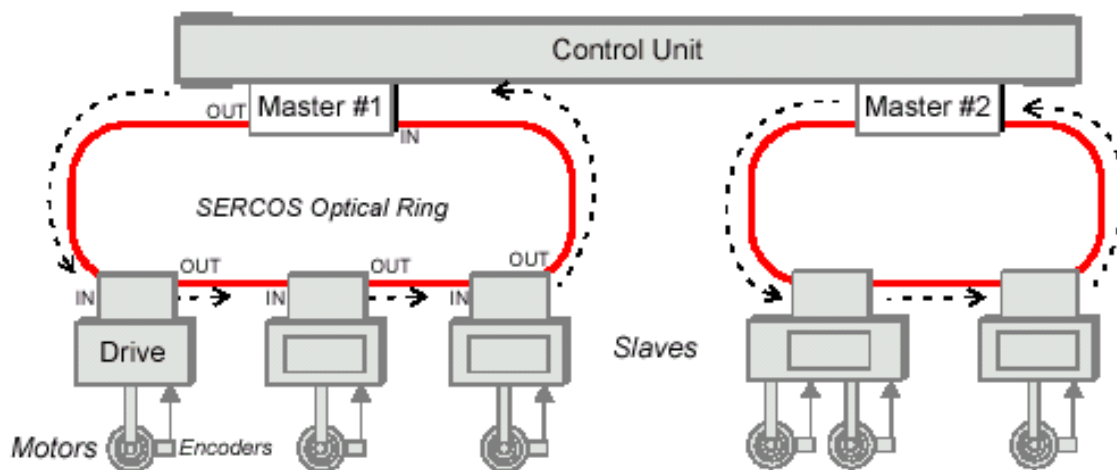
| | | |
|------|---|--|
| 5 | Change Bit Commands 0 No Change in Command Status 1 Changing Command Status | |
| 4, 3 | Reserved | |
| 2 | Error 0 No Error 1 Error in Service Channel | Error message is in drive's Service Channel. |
| 1 | Busy 0 Step Finished 1 Step in Progress | |
| 0 | Master Service Transport Handshake | A Toggle bit |

[Introduction](#) | [Overview](#) | [Data Types](#) | [Communications](#) | [Procedures](#) | [Telegrams](#) | [Topologies](#)



Copyright © 2002
Motion Engineering

Sercos- Application Topologies



SERCOS

A Sercos object represents a Master on the SERCOS ring.

NODE

A Node object represents a Slave Node on the SERCOS ring, which is typically either a drive or an I/O module.

