ACS500-MODBUS

Installation & Start-up Manual

ACS500 Modbus™ Protocol For ACS/ACH 500 drives

ACS500-MODBUS-US-04b
EFFECTIVE 3/1/95
SUPERCEDES 10/1/94

ABB Drives

ABB

ACS500-MODBUS
Direct Modbus Interface
For ACS 500, ACH 500, Series B

Installation & Start-up Manual

ACS500-MODBUS-US-04b

EFFECTIVE: 3/1/95
SUPERCEDES 10/1/94

Safety Instructions

General Safety Instructions

Warnings in this manual appear in either of two ways:

- *Dangerous voltage warnings*, preceded by a Dangerous Voltage symbol, indicate the presence of voltages which may cause death or serious injury. These warnings describe procedures to avoid death or serious injury.
- *General warnings*, preceded by a General Warning symbol, indicate situations or conditions which may cause death or serious injury. These warnings describe procedures to avoid death or serious injury.

CAUTIONS inform you of situations or conditions which will damage machinery or cause additional motor-operation down-time if you do not take suggested steps to correct or address such situations or conditions.

Note: *Notes provide you with additional and useful information. Although less urgent than cautions and warnings, notes are important and should not be ignored.*

Warning Symbols

For your own safety please pay special attention to instructions containing these symbols:

This warning symbol indicates the presence of dangerous voltage. This symbol informs you of high voltage conditions, situations, and locations that may cause death or serious injury if you do not follow precautions and proper steps.

This warning symbol indicates a general warning.

This warning symbol indicates an electrostatic discharge hazard.

Warnings, Cautions, and Notes

WARNING! Your drive contains dangerous voltages when connected to the line power. Always check that the ACS 501 is safe, after disconnecting the power, by measuring the DC bus voltage and line input voltage. Failure to check voltages could cause death or serious injury. Only a qualified electrician should carry out the electrical installation.

Note that the Motor Control Card of the ACS 501 is at DC bus voltage potential.

The DC bus capacitors contain dangerous DC voltage levels (1.35 x V_{IN}). After disconnecting the supply, wait at least five minutes after the display readout on the control panel has disappeared before taking any measurements.

Dangerous external control voltages may be present on the relay outputs of the Control Interface Card and Option Cards.

CAUTION: Electrostatic Discharge (ESD) can damage electronic circuits. Do not handle any components without following the proper ESD precautions.

Table of Contents

Appendix B – Modbus Protocol

This page intentionally left blank.

Chapter 1 – Introduction

This chapter describes the purpose and contents of this manual, describes the intended audience, explains conventions used in this manual, and lists related publications.

How To Use This Manual

The purpose of this manual is to provide you with the information necessary to install, start-up, and program an ACS 500 or ACH 500 Adjustable Frequency AC Drive for the direct Modbus RTU mode connection. This manual also describes features and functions which have been added to the drive to support the serial communication, and gives recommendations for external connections, wiring, routing, and cable sizes.

Chapter 1 - Introduction, the chapter you are reading now, introduces you to the *ACS500 Modbus Installation & Start-up manual* and conventions used throughout the manual.

Chapter 2 - Overview of the Modbus connection gives an overview of the direct Modbus serial communication implementation for the ACS 500 and ACH 500 drives. This chapter describes all the different services provided by the network.

Chapter 3 - Installation describes planning for the network installation. This chapter also includes the requirements and connections for the serial interface wiring.

Chapter 4 - Programming describes how to program the ACS 500 and ACH 500 drives for the Modbus. This chapter also lists all the new and modified parameters, which are required for the serial communication network.

Chapter 5 - Start-up Procedure describes safety, installation inspection, how to check and setup the communication parameters.

Chapter 6 - Fault Tracing describes troubleshooting procedures through fault counters, fault queue, and tracing faults to their origins.

Appendix A – Parameter List lists all the parameters from the ACS 500 drive with the units and scaling.

Intended Audience

The audience for this manual has:

- Knowledge of standard electrical wiring practices, electronic components, and electrical schematic symbols.
- Minimal knowledge of ABB product names and terminology.
- Previous experience in installing, operating, and programming the ACS 500 or ACH 500 drives.

The audience for this manual will install, start-up, and diagnose the Modbus network. The audience will also program the ACS 500 drives for the serial communication network.

Conventions Used In This Manual

Listed below are terms and language conventions used in this manual. These terms and conventions are defined here to help you understand their meanings and applications throughout this manual.

Control Panel Display

The Control Panel display is an LCD readout of drive functions, drive parameter selections, and other drive information. Letters or numbers appear in the display according to which Control Panel keys you press.

Control Panel Keys

Control Panel keys are flat, labeled, push-button-type devices that allow you to monitor drive functions, select drive parameters, and change drive macros and settings.

Main

A main is the first level of programming. The Mains organize the Parameters into four main functional groups. A Main in this manual is the number corresponding to Group access. All Groups in the 10s range are accessed on the Control Panel through `CONTROL CONNECTIONS/MAIN 10`. Access Groups in the 20s range through `DRIVE PARAMETERS/MAIN 20`. Access Groups in the 30s range through `PROTECTION PARAMETER/MAIN 30`, and access Groups in the 40's range through `APPLIC PARAMETERS/MAIN 40`.

Group

A Group is a sub-set of a Main. Groups are grouped within Mains according to their 10s, 20s, 30s, or 40s range. For example, Groups numbered 30.1, 30.2, 30.3, and 30.4 are found in `PROTECTION PARAMETER/MAIN 30`. Parameters are accessed through Groups.

Parameter

A parameter is a sub-set of a Group, selected through the Control Panel keys. Parameters in this manual often are expressed as a number, a decimal (.), another number, a decimal, and another number. The first number at the left represents the Main. The number between the decimals represents the Group, for example, 20.2 (Start/Stop). The number at the right represents a Parameter within that group, for example, 4 (Brake Chopper). In this manual, Parameter 4 in Group 20.2 is expressed as Parameter 20.2.4.

Press Press a key on the Control Panel to achieve a desired result. In this manual, individual Control Panel keys are enclosed in square brackets. For example, the Setting mode key is expressed as [*]. Refer to *Chapter 2 – Overview of the ACS 501, Control Panel Operation*, for details.

RTU There are two modes of operation for the Modbus protocol. These are identified as ASCII Mode or RTU Mode. Sometimes the RTU Mode is also called Modbus B. A device talking Modbus in the ASCII Mode can not communicate with a device communicating in the RTU Mode, even though the ASCII Mode and the RTU Mode are similar.

Terminal Block A terminal block is a group of wire connections on a drive. This manual expresses specific terminal blocks and connections as a letter, usually X, a number, a colon (:), and another number. The letter and number to the left of the colon represent the name of the terminal block, for example, X25. The number to the right of the colon represents the terminal connection, for example 16, on the terminal block. In this manual, a terminal connection numbered 16, located on a terminal block named X25, is expressed as X25:16.

Warranty and Liability Information

The warranty for your ABB drive covers manufacturing defects. The manufacturer carries no responsibility for damage due to transport or unpacking.

In no event and under no circumstances shall the manufacturer be liable for damages and failures due to misuse, abuse, improper installation, or abnormal conditions of temperature, dust, or corrosives, or failures due to operation above rated capacities. Nor shall the manufacturer ever be liable for consequential and incidental damages.

The period of manufacturer's warranty is 12 months, and not more than 18 months, from the date of delivery.

Extended warranty may be available with certified start-up. Contact your local distributor for details.

Your local ABB Drives company or distributor may have a different warranty period, which is specified in their sales terms, conditions, and warranty terms.

If you have any questions concerning your ABB drive, contact your local distributor or ABB Drives office.

The technical data and specifications are valid at the time of printing. ABB reserves the right to subsequent alterations.

Related Publications

For related information about the drive, refer to the ABB *ACS 500 Adjustable Frequency AC Drives 2 to 350 HP Programming Manual Including Application Macros (ACS 500-05)* and the *ACS 501 with Option Pack Users Manual (ACS 500-08).*

This page intentionally left blank.

Chapter 2 – Overview of the Modbus connection

This chapter describes the general features of the ACS500 Modbus RTU serial communication connection to the ACS 500 and ACH 500 drives. This overview also gives a short functional description of the Modbus implementation to the drives.

Introduction

The Modbus protocol is a master - slave type serial communication protocol, which has become an industry standard. The Modbus protocol is defined in Modicon publication *Modbus Protocol Reference Guide*. This chapter is not intended to replace the information provided by the book, but to give a rough overview of the Modbus, and also to provide an overview of the Modbus implementation for the drives.

Modbus transactions

Nodes on one serial communication cable communicate using a master-slave technique, in which only one device (the master) can initiate transactions (called 'queries'). The other devices (the slaves) respond by supplying the requested data to the master, or by taking the action requested in the query. Typical master devices include DCS systems, and panels. The ACS 500 and ACH 500 drives are always a slave on the network.

On the Modbus communication network, all the data transfer is done between the master and the slaves. Two slaves cannot directly communicate with each other.

The Modbus allows the master to access one individual drive at a time, or the master can also broadcast a command to every station with one message. There are no replies for broadcasted commands, while for individual slave station access there is always either a slave response message, or due to a communication error, the master station can also time out, and do a retry.

On a RS-485 serial interface, it is possible to connect up to 32 individual nodes on one serial communication cable. This capability with the possibility of having unique slave station numbers on the drives, allows to do multidrop on the Modbus communication link. With this multidrop configuration, there can be one master, but up to 31 drives on one serial communication link.

Transmission modes

There are two standard transmission modes for Modbus network: ASCII and RTU. In addition to the selected transmission mode, the user can typically select the baud rate, parity, etc. during configuration of each controller. The mode and serial parameters must be the same for all devices on the Modbus network.

The Modbus driver on the ACS 500 and ACH 500 drives is programmed to do only the RTU mode of communication.

Message framing

Message framing is described in detail in the *Modbus Protocol Reference Guide* published by Modicon.

Serial configuration

The drive has a selectable baud rate with selections of 1200, 2400, 4800, and 9600 baud. Parity is selectable between None, Even, and Odd.

The drive slave station identification number can be set in a range of 1 to 247.

Supported Function Codes

03 Read Holding Registers

The Read Holding Registers function reads the binary contents of parameter table registers (4XXXX references) from the drive.

Broadcast is not supported.

On the ACS 500 and ACH 500 drives all the parameters are mapped in a fixed way from the main - group - parameter addressing used in the programming panel, into the simulated 4XXXX register area for the Modbus communication.

All the values are represented as 16-bit integers, or packed boolean information.

With one read request from the drive it is possible to read multiple parameters. All the parameters in one read must be within one parameter group. If the read either start, ends, or contains addresses outside parameter groups, the read will return an exception, and no data is returned.

06 Preset Single Register

The Preset Single Register function modifies the binary contents of one parameter table register (4XXXX references) in the drive.

Broadcast is supported.

All the values are represented as 16-bit integers, or packed boolean information.

The service 06 has identical action as the service 16.

16 Preset Multiple Registers

The Preset Multiple Registers function modifies the binary contents of parameter table registers (4XXXX references) in the drive.

Broadcast is supported.

Even if the Preset Multiple Registers can generally be used to modify multiple register values using one transaction, the ACS 500 and ACH 500 drives will only accept writes with one parameter at a time. A multiple register setting will return an exception back from the drive.

Addressing of parameters

All parameters have their individually determined location in ACS 500 parameter structure. Parameters are found by using MAIN, GROUP, and PARAMETER number. To address the parameters from the Modbus side, the parameter table in the drive is mapped in a fixed way to the simulated 4XXXX register area. The numbering is the same as in the local drive panel in the standard version.

The addressing codes for the parameters are listed in the *Appendix - A*. A summary of addressing is in the *Table 2-1, Parameter addressing*.

Table 2-1 Parameter Addressing

Main	Main #	Group #	Param. #	Modbus address
Operating Data	0	0	1 - 23	40001 - 40023
Start-up Data	5	0	1 - 11	45001 - 45011
Control Connections	1	1 - 8	1 - 11	41101 - 41806
Drive Parameters	2	1 - 5	1 - 11	42101 - 42511
Protection / Information	3	1 - 5	1 - 13	43101 - 43503
Application Parameters	4	1	1 - 12	44101 - 44112
Modbus Parameters	5	1	1 - 7	45101 - 45125

The mapping is fixed with the main number as thousands, the group number within the main in hundreds, and the parameter number in tens and in ones.

An example of addressing is the External Reference 1 selection from Main 10, Group 2, parameter 2. This is on the Modbus side accessible at the address of 41202.

Modbus Exception codes

The Modbus network has exception codes that the slave will return to the master when an illegal, or an unknown command has been received. This exception handling is included in the ACS 500 Modbus interface. The following exceptions can be returned by the drive:

Table 2-2 Exception codes

Command	Code	Meaning
	0x01	Unsupported command for ACS 500
	0x03	Illegal data value
Read Holding Regs	0x31	Command length incorrect
Read Holding Regs	0x32	Parameter address incorrect, or read outside of groups
Write Regs	0x61	Command not supported, while control location is KEY-PAD
Write Regs	0x62	Command format incorrect
Write Regs	0x63	Parameter address incorrect
Write Regs	0x64	Backup not allowed while under voltage in ACS 500 DC-bus
Write Regs	0x65	Incorrect code to activate BACKUP to EEPROM
Write Regs	0x66	Set of parameter not allowed while the drive is running
Write Regs	0x67	Parameter is read only

Chapter 3 – Installation

This chapter describes the installation of the Modbus RTU network for ACS 500 Drives using the RS-485 connection. It describes the planning of the network installation, required hardware wirings, and programming considerations for the drive.

Introduction

The Modbus connection to the ACS/ACH 500 series of drives is an industry standard Master - Slave protocol, and is suitable for commercial systems, and for customers who want to do the integration themselves.

The flexibility gives a great degree of freedom for using the network; therefore, it is important to fully plan the functionality of the network before doing the actual implementation on site.

Network planning should include the following topics:

- Define what devices, and in what quantities will be connected to the network.
- Define what control information will be to be sent down to the drives.
- Define what feedback information will to be sent from the drives to the controlling system.

Controlling the drive

The drive can receive control through the Modbus connection by having the master station write into the parameters in the drive. Typical uses is to modify the control word at address 45102 to start, stop, or reset the drive, and external reference 40013 for changing the running speed of the drive.

Control actions which are available are described in detail in *Chapter 4–Programming* of this manual.

The Modbus RTU protocol supports broadcast communication for writing data to the drives. This feature can be used to control simultaneously multiple drives with one command. To use broadcast, write to a slave station address of 0. The broadcasted message is not acknowledged, and no indication of the completion of the write is returned back to the master.

Feedback from the drive

The communication master on the Modbus link can access any parameter in the drive freely for feedback purposes. To get information back from the drive, the master needs to read one, or multiple parameters from the drive.

Hardware installation

The Modbus network is based upon the industrial RS-485 standard. The RS-485 connection is done using a shielded twisted pair cable.

On the Modbus network for the ACS 500 drives the wiring connections described below are recommended. This connection method ensures minimal noise on the network, while keeping the connections simple, and affordable.

The control board, which has both the analog and digital terminals, and the RS-485 port must be grounded by using the on-board resistor/capacitor circuit. Any possible additional grounding wires (like the one connected to X50:8) must be removed before installing the Modbus network.

The digital and analog inputs come directly to this floating control board. Any dangerous voltages connected through these terminals will go directly to the control board, panel, and through the Modbus network to other drives, and to the backside of the panels. Equipment damage or personal injury could be caused by inappropriate connections.

Wiring

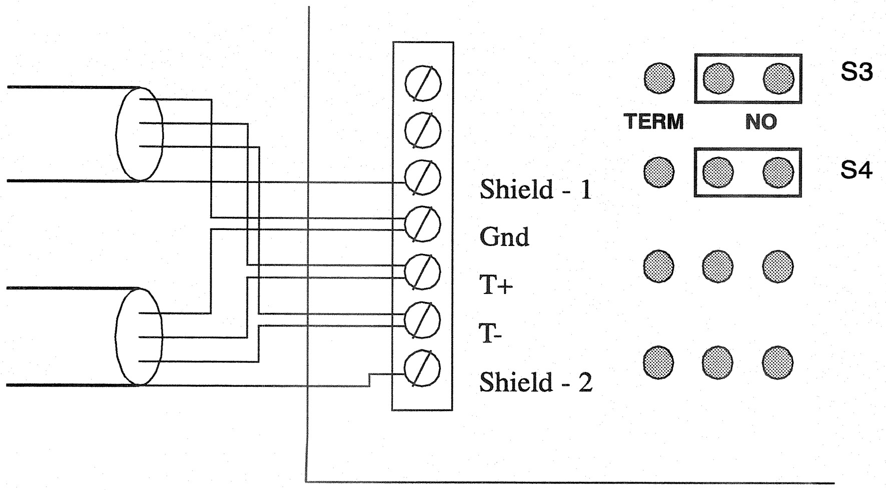
Modbus network should be wired using Belden 9842 or equivalent. The Belden 9842 is dual twisted shielded pair cable with a wave impedance of 120 Ω. One of these twisted shielded pairs is used for the RS-485 link. One of the wires in the other pair is used for the logical ground, leaving one wire unused. For details, see the diagrams below.

The RS-485 link is a daisy-chained bus, without dropout lines. The RS-485 link should also be terminated on both physical ends of the wire to reduce the noise on the network.

Connections

The network should be connected according to the following diagram. In the Figure 3–1 the termination resistors are disconnected.

Figure 3–1 Communication wiring for ACS/ACH 500 Drives



The RS-485 connection is done using one of the twisted pairs in the cable. The T+ terminals are all connected together, and the T- terminals are all connected together. The logical grounds for all of the drives are connected together using terminal four.

The shields at both ends of the cable are connected to the drives. On one end, the shield should be connected to terminal three, and on the other end to terminal seven. The shielding must not be made continuous by connecting the incoming and outgoing cable shields to the same terminals. The proper shield connection is shown in the Figure 3–1.

The connections should be made only while the drive is disconnected from the power source.

The drive should be checked visually to verify that no loose metal pieces from the wire or the shield are left either over the circuit board or over the power components

---

ESD procedures should be followed.

Grounding and termination

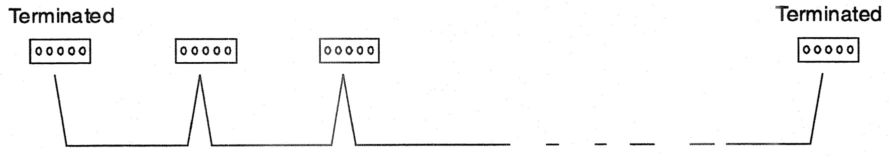
Modbus network should not be directly grounded at any point. All the devices on the network should be well grounded using their corresponding grounding terminals.

As always, the grounding wires should not make any closed loops, and all the devices should be grounded into common ground.

Modbus network must be terminated using 120 Ω resistors at both ends of the network. These resistors are already resident on the ACS/ACH 500 board. To connect the termination resistors, move jumpers S3 and S4 on the control interface board to the terminate position. Figure 3–1 shows the termination resistors disconnected.

The termination should not be done on the intermediate stations on the network.

Figure 3–2 Termination for the link



This page intentionally left blank.

Chapter 4 – Programming

This chapter describes the programming of the ACS 500 and ACH 500 drives for the Modbus network. The reader should be familiar with the ACS 500 drive and the Modbus communication network.

Programming of the Drive

The ACS 500 and ACH 500 drives are programmed through the local programming panel. The operation of the panel is described in detail in the ACS programming manual.

For the direct Modbus connection, there are a group of new parameters. In addition to adding new parameters to the drive, some existing parameters have been modified to support additional functions. All of these parameters are described in detail in this chapter.

Unlike the local panel, all the data being transported through the Modbus network is in binary format. To use the data correctly, the units, and scaling of the parameters and operating data should be verified using the parameter table within this chapter.

Modified parameters

The Modbus interface follows the control location logic, which was introduced with the ACS/ACH 500 drive. With the introduction of the serial communication, the drive now has new control source selection to support control from the serial communication link.

In addition to the control source selection, the operating data `9 CONTROL LOCATION` is used now to also lock out the serial communication link from doing accidental writes to the drive. While the `CONTROL LOCATION` is in `KEYPAD` mode the serial communication can only be used to read parameter values from the drive.

Control locations

Typical control uses for the Modbus write services include:

- Start and stop the drive,
- Change the direction of the drive,
- Receive external reference 1 and/or 2,
- Select the source for both the external references 1 and/or 2,
- Change the reference scaling using offset and gain variables,
- Setup the current limit for the drive,
- Setup the acceleration and deceleration time 1,
- Setup the PI controller gain and integration time.

Main 10 Control Connections

Group 10.1 Start/Stop/Direction

These parameter values can only be altered with the ACS/ACH 500 stopped. The Range/Unit column in Table 4–1 shows parameter values. The text following the table explains parameter values in detail.

Table 4–1 Group 10.1

Parameter	Range/Unit	Description
1 EXT 1 STRT/ STP/DIR	Not Sel / Digital Inputs / Keypad / Standard communication	External control reference R1 start/stop and direction input.
2 EXT 2 STRT/ STP/DIR	Not Sel / Digital Inputs / Keypad / Standard communication	External control reference R2 start/stop and direction input

1 EXT 1 STRT/STP/DIR

This parameter selects the Digital Inputs used for Start/Stop and Reverse commands.

NOT SEL

No Digital Input selected.

DI1

Two-wire start/stop

0 V DC = Stop and 24 V DC = Start. Rotation direction is fixed.

DI1,2

Two-wire start/stop and direction

Start/Stop is connected to DI1 and Reverse to DI2. 0 V DC on DI2 = Forward and 24 V DC = Reverse.

DI1P,2P

Three-wire start/stop

Start/Stop commands are from momentary push-buttons. The stop push-button is normally closed, and connected to DI2. The start push-button is normally open, and connected to DI1. Multiple start push-buttons are connected in parallel, and stop push-buttons in series.

DI1P,2P,3

Three-wire start/stop and direction

Refer to DI1,2. Reverse is connected to DI3. 0 V DC = Forward, 24 V DC = Reverse.

DI1P,2P,3P

Start forward, start reverse, and stop.

Start and direction commands are given simultaneously with two separate momentary push-buttons. The stop push-button is normally closed, and connected to DI3. The start forward push-button is normally open, and connected to DI1. The start reverse push-button is normally open, and connected to DI2. Multiple start push-buttons are connected in parallel, and stop push-buttons in series.

DI6

Two-wire start/stop

0 V DC = Stop and 24 V DC = Start. Rotation direction is fixed.

DI6,5

Two-wire start/stop and direction.

Start/Stop is connected to DI6 and Reverse is connected to DI5. 0 V DC on DI5 = Forward.

KEYPAD

Start/Stop command and Direction command are from the Keypad for Ext 1.

STD. COMM.

Start/Stop and direction command is received from the serial communication. To use this option, a command word must be modified by a write to the address of 45102.

2 EXT 2 STRT/STP/DIR

This parameter selects the Digital Inputs used for Start/Stop and Reverse commands. The choices are the same as Parameter 10.1.1 (Ext 1 Strt/Stp/Dir)

3 LOC/EXT DIRECTION

This parameter allows you to fix rotation direction to FORWARD or REVERSE. If you select REQUEST, the rotation direction is selected by the source defined in parameters 10.1.1 (Ext 1 Strt/Stp/Dir) and 10.1.2 (Ext 2 Strt/Stp/Dir). These selections include digital inputs, keypad, and standard communication. If FAST REV is selected, and Parameter 20.2.3 (Stop function) is set to COAST, the modulator starts to operate in a reverse direction immediately when REVERSE is requested. This procedure results in fast reversing.

Group 10.2 External Reference Select

These parameter values can be altered with the ACS 500 running, except those marked with (O). The Range/Unit column in Table 4–2 shows parameter values. The text following the table explains parameter values in detail.

Table 4–2 Group 10.2

Parameter	Range/Unit	Description
1 EXT 1/EXT 2 SELECT (O)	OP DATA 12/ DI1–DI6/ STD COMM.	External control location selection input.
2 EXTERNAL REF1 SEL (O)	OP DATA 13/Analog and Digital Inputs/ STD COMM.	External reference 1 input.
3 EXT REF1 MINIMUM	0 – 500 Hz (ACS 501) 0 – 120 Hz (ACS 502)	External reference 1 minimum value.
4 EXT REF1 MAXIMUM	0 – 500 Hz (ACS 501) 0 – 120 Hz (ACS 502)	External reference 1 maximum value.
5 EXT REF1 OFFSET	-30–30 Hz	Offset for scaling external reference 1
6 EXT REF1 GAIN	-100–100%	Gain for scaling external reference 1
7 EXTERNAL REF2 SEL (O)	OP DATA 14/Analog and Digital Inputs/ STD COMM	External reference 2 input.
8 EXT REF2 MINIMUM	0 – 500 Hz (ACS 501) 0 – 120 Hz (ACS 502)	External reference 2 minimum value.
9 EXT REF2 MAXIMUM	0 – 500 Hz (ACS 501) 0 – 120 Hz (ACS 502)	External reference 2 maximum value.
10 EXT REF2 OFFSET	-30–30 Hz	Offset for scaling external reference 2
11 EXT REF2 GAIN	-100–100%	Gain for scaling external reference 2

1 EXT 1/EXT 2 SELECT

This parameter defines how to select the external control location (Ext Ref 1/ Ext Ref 2). If you choose `OP DATA 12`, the selection is made with Operating Data Parameter 12 (Ext Ref 1 or 2). If you choose a Digital Input (DI1 – DI6), 0 V DC = Ext Ref 1 and 24 V DC = Ext Ref 2.

If the standard communication is selected, the selection between External reference 1 and 2 is received using serial communication. The command word is telling the drive which one of the references to use. To use this option, the command word needs to be modified by a write.

2 EXTERNAL REF1 SEL

This parameter selects the signal source of External Reference 1.

OP DATA 13

Reference is given from the Keypad, Operating Data Parameter 13.

AI1

Reference from analog input 1.

AI2

Reference from analog input 2.

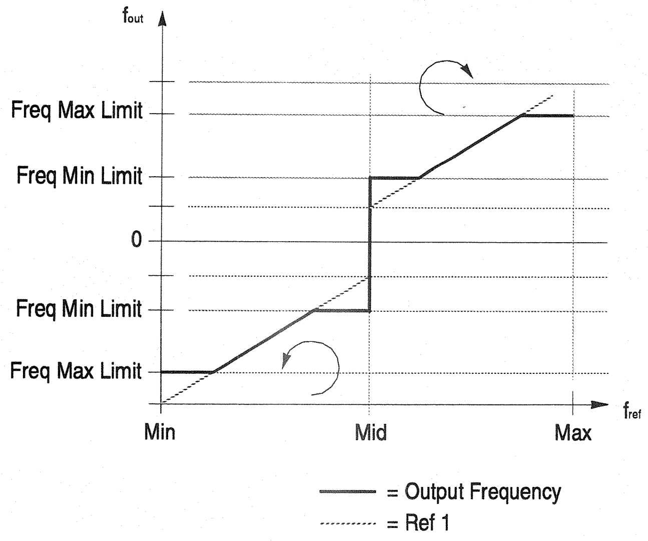
AI1/JOYST

Reference from analog input 1 configured for a joystick. Analog input minimum signal is full speed reverse, and analog input maximum signal is full speed forward. The mid point between minimum and maximum is zero speed.

CAUTION: Minimum reference for joystick must be 0.3 V/0.6 mA or greater. If a 0 – 10 V or 0 – 20 mA signal is used, the drive will run at fmax to Reverse if the control signal is lost. Set Parameter 10.5.1 (Minimum AI1) to 2 V/4 mA or to a value 0.3 V/0.6 mA or greater, and Parameter 30.1.2 (AI < Min Function) to Fault, and the drive will stop in case of lost control signal.

Figure 4-1 shows Joystick control.

Figure 4-1 Joystick Control



DI3U,4D(R)

Speed reference via digital inputs as Floating Point Control, or Motor Operated Potentiometer Control. DI3 increases speed, and DI4 decreases speed. (R) indicates that the reference will reset to minimum frequency when stop command is given. The rate of change of the reference signal is controlled by parameter 20.3.5, ACCEL TIME 2.

DI3U,4D

Same as above except speed reference does not reset to zero on stop command.

DI5U,6D

Same as above.

STD COMM.

This choice selects the standard communication for the source of external reference one. In addition to this selection, a new reference needs to be written to address 40013.

3 EXT REF1 MINIMUM

0 Hz – 500 Hz (ACS 501)

0 Hz – 120 Hz (ACS 502)

This parameter sets the frequency corresponding to the minimum reference. When Parameter 20.1.1 (Minimum Frequency) is changed, this parameter is automatically set to the same value.

4 EXT REF1 MAXIMUM

0 Hz – 500 Hz (ACS 501)

0 Hz – 120 Hz (ACS 502)

This parameter sets the frequency corresponding to the maximum reference. When Parameter 20.1.2 (Maximum Frequency) is changed, this parameter is automatically set to the same value.

5 EXT REF1 OFFSET

This parameter with **EXT REF1 GAIN** is used for scaling the incoming serial reference. **EXT REF1 OFFSET** is added to the received serial reference.

6 EXT REF1 GAIN

This parameter with **EXT REF1 OFFSET** is used for scaling the incoming serial reference. **EXT REF1 GAIN** specifies what percentage of the received serial reference is added to the received serial reference.

External Reference Offset and Gain are also used to scale reference sources `AI1`, `AI2`, and `AI1/JOYST`.

Example: The reference is 20 Hz, EXT REF1 OFFSET is 5Hz, and EXT REF1 GAIN is 10%. The drive reference is now 20Hz + 5Hz + 0.1* 20Hz = 27Hz.

7 EXTERNAL REF2 SEL

OP DATA 14

This parameter selects the signal source for External Reference 2.

Reference is given from the Keypad, Operating Data Parameter 14.

AI1

Reference from analog input 1.

AI2

Reference from analog input 2

DI3U,4D(R)

Speed reference via digital inputs as Floating Point Control, or Motor Operated Potentiometer Control. DI3 increases speed, and DI4 decreases speed. (R) indicates that the reference will reset to zero when stop command is given.

DI3U,4D

Same as above except speed reference does not reset to zero on stop command.

DI5U,6D

Same as above.

STD COMM.

This selection selects the standard communication for the source of external reference two. In addition to this selection, a new reference can be changed by a write to the address of 40014.

8 EXT REF2 MINIMUM

0 Hz – 500 Hz (ACS 501)

0 Hz – 120 Hz (ACS 502)

This parameter sets the frequency corresponding to the minimum reference. When Parameter 20.1.1 (Minimum Frequency) is changed, this parameter is automatically set to the same value.

9 EXT REF2 MAXIMUM

0 Hz – 500 Hz (ACS 501)

0 Hz – 120 Hz (ACS 502)

This parameter sets the frequency corresponding to the maximum reference. When Parameter 20.1.2 (Maximum Frequency) is changed, this parameter is automatically set to the same value.

10 EXT REF2 OFFSET

Same as above.

11 EXT REF2 GAIN

Same as above.

The External Reference Offset and Gain are also used to scale reference sources `AI1`, and `AI2`.

Group 10.4 System Control Inputs

These parameter values can only be altered with the ACS 500 stopped. The Range/Unit column in Table 4–3 shows parameter values. The text following the table explains parameter values in detail.

Table 4–3 Group 10.4

Parameter	Range/Unit	Description
1 RUN ENABLE	Yes/DI1–DI6	Run enable input.
2 FAULT RESET SELECT	Not Sel/DI1–DI6/ On Stop/ Std Comm.	Fault/Warning/Supervision reset input.
3 PARAM LOCK SEL	Op Data 20/ DI1–DI6/ Std. Comm	Parameter lock input.
4 EXTERNAL FAULT	Not Sel/DI1–DI6	External fault input.

1 RUN ENABLE

This parameter selects the source of the Run Enable signal.

YES

Run Enable signal active. Drive is ready to start without an external Run Enable signal.

DI1 – DI6

To activate the Run Enable signal, the selected Digital Input must be connected to +24 V DC. If the Digital Input goes to 0 V DC, the drive will coast to stop.

2 FAULT RESET SELECT

NOT SEL/DI1 – DI6/ON STOP/STD COMM

If you select `NOT SEL`, fault reset is done from the Keypad. If a digital input is selected, fault reset is done from an external switch or from the keypad. Reset is activated by opening a normally closed contact (negative edge on digital input). If `ON STOP` is selected, a fault is reset by giving a stop command from the active STOP signal, or from the Keypad. A stop received using the serial communication will also reset the fault.

If the `STD COMM` is selected, the fault reset can be done using the serial communication. To use this function, a control word must be modified by the communication master. The faults are cleared on the rising edge of the fault reset bit on the control word.

3 PARAM. LOCK SEL	This parameter selects the control location for Parameter Lock. If you select `OP DATA 20`, Parameter Lock is controlled with Operating Data Parameter 20 (Parameter Lock). If you select a Digital Input (1 – 6), 0 V DC equals Open and 24 V DC equals Locked. If the `STD COMM` is selected, the Parameter Lock is controlled with the parameter lock bit from the control word. A control word must be programmed to be modified by the communication master to use this option.
4 EXTERNAL FAULT	**NOT SEL** **DI1 – DI6** Input for external fault device such as a motor overload relay or fire alarm. The device should have a normally closed contact. The digital input must be connected to +24 VDC. If the digital input goes to 0 VDC, the drive will coast to stop and a fault message will display.

Group 10.8 External Communication

The external communication group has all the new parameters to define, setup, and diagnose the serial Modbus network. Any of these new parameters can be modified while the drive is running. The parameters 1 through 3 will only take effect after the next power cycling of the drive.

Table 4–4 Group 10.8

Parameter	Range/Unit	Description
1 DRIVE ID-NUMBER	1 – 32 (GS-Bus) 1–247 (Modbus)	Drive station number
2 PROTOCOL	GS-BUS / MODBUS	Active protocol
3 BIT RATE SELECT	1200 / 2400 / 4800 / 9600	Communication speed for the Modbus network
4 PARITY	None None / Even / Odd	Communication parity bit setup
5 COMMS FAULT FUNCT	None / Fault / Fault + Stop	Communication fault function
6 BAD MESSAGES COUNTER	-32768 – 32767	Free running counter for received bad message packets. Parity, CRC error, or unsupported command.
7 GOOD MESSAGES COUNTER	0–65535	Free running counter for received good message packets

1 DRIVE ID-NUMBER

This parameter identifies the drive slave station number on the Modbus network. Every slave station on the Modbus network has to have a unique station number.

2 PROTOCOL

This parameters selects the active protocol on the drive. You can select between the GS-Bus and Serial Modbus. The GS-Bus has been designed to support the remote panel and the DMT500PC Drive Monitoring Tool software.

Modbus communication supports the DPT500 Drive Programming Tool software.

3 BIT RATE SELECT

This parameter defines the serial communication speed for the drive in baud rate. Each station on the network must use the same communication speed.

4 PARITY

This parameter defines the parity checking used for the drive. Each station on the network must use the same parity setup. Even or Odd parity is more reliable than the parity selection of None.

5 COMMS FAULT FUNCT

This parameter defines what action is taken when the drive detects a communication fault. The actions include the options for ignoring the fault, generating a fault, without stopping the drive, and to generate a fault, and stop the drive.

With the Modbus, the communication loss is detected using bit 12 on the command word. If this bit is changes either from 0 to 1, or from 1 to 0, within a 10 second interval, the drive will not generate a communication loss. If the bit stays unchanged for over 10 seconds, a communication loss is generated.

With the GS-Bus the communication loss is generated if there is no communication from the panel for a specified time.

The action taken in this case is defined by this parameter.

6 BAD MESSAGES CONTER

This is a free running counter displaying the cumulative count of received bad characters on the serial communication.

7 GOOD MESSAGES COUNTER

This is a free running counter displaying the cumulative count of received good characters on the serial communication.

Invisible parameters

The addition of serial communication to the ACS drive has caused a need for adding new parameters for the drive. Some of these parameters are only accessible through the serial interface, not from the local panel. These are used especially for controlling and monitoring the drive through the serial interface.

These parameters are in the main 50, and in group 1. (45101–45107). Only the registers 45101 through 45103 can be modified. Others are read only.

Table 4–5 Invisible parameters

Parameter	Range/Unit	Description
1 STATUS WORD	Bit packed boolean	Drive status word
2 COMMAND WORD	Bit packed boolean	Drive command word
3 BACKUP	-1/358/Other	Backup the values to EEPROM command
4 CUMULATIVE Wh	0 – 999	Cumulative Wh
5 CUMULATIVE kWh	0 – 999	Cumulative kWh
6 CUMULATIVE MWh	0 – 99999	Cumulative MWh
7 AI1 A/D CONVERSION	0–32767	Result of A/D conversion from AI1
8 AI2 A/D CONVERSION	0–32767	Result of A/D conversion from AI2
9 DIGITAL INPUTS	Bit packed boolean	The status of digital inputs
10 RELAY OUTPUTS	Bit packed boolean	The status of relay outputs
11 – 17 CRI VERSION	ASCII characters	CRI Software version
18 – 23 MCR VERSION	ASCII characters	MCR Software version
24 SERIAL NUMBER HI	0 – 9999	Thousands of the drive serial number
25 SERIAL NUMBER LO	0 – 999	Serial number

1 STATUS WORD

The drive status word is a 16-bit packed logical word. Each one of the bits in the word has a separate purpose, or is unused for the ACS 500 drive. The bits have the following meanings:

Table 4–6 Status word bits

Bit number	Description
0	0 = Drive stopped 1 = Drive running
1	0 = Drive running or faulted 1 = Drive stopped and ready to start (no faults)
2	0 = Drive not faulted 1 = Drive faulted
3	0 = Drive in keypad control 1 = Drive in external control
4	0 = DC Voltage normal 1 = DC Undervoltage or DC Overvoltage
5	As bit 4
6	Not in use
7	1 = Flying start
8	1 = Drive running at speed
9	1 = Drive Overtemperature Warning
10	1 = Motor Overtemperature Warning
11	Digital input 1
12	Digital input 2
13	Digital input 3
14	Digital input 4
15	0 = Drive running at forward direction 1 = Drive running at reverse direction

2 CONTROL WORD

The drive control word is a 16-bit packed logical word, which is used to control the drive with a minimum amount of communication network load. Each one of the bits on the control word has a unique meaning and purpose.

Table 4–7 Control word bits

Bit number	Description
0	1 = Stop by ramp
1	1 = Stop by using the existing stop type
2	1 = Stop by coast
3	Not used
4	0 = Stop by using the existing stop type; 1 = Start to speed
5	1 = Select ext ref 2 Takes effect only if activated from 10.02.01
6	1 = Select ext ref 1 Takes effect only if activated from 10.02.01
7	1 = Select ext ref 2 Takes effect only if activated from 10.02.01
8	Not used
9	Not used
10	Not used
11	Not used
12	Communication loss detection
13	1 = Panel lock Takes effect only if activated from 10.04.03
14	1 = Reset fault Takes effect only if activated from 10.04.02
15	0 = Forward 1 = Reverse

Bit 12 has a special purpose. This is used for communication loss detection. If bit 12 does not change its value for 10 seconds, a communication loss fault is generated.

This bit is used typically on PLC controlled applications. The PLC should generate a toggling bit, which is then sent to the control word bit 12. If either there is a communication loss, or the PLC is set to program mode, this bit will stop toggling causing a communication loss fault to the drive.

3 MODBUS BACKUP

This parameter can be used for writing the present values of the parameters into the EEPROM memory in the drive. The serial communication does not automatically save parameter values into the EEPROM, whenever they are modified using the serial communication protocol. To make the changes permanent, they need to be especially written to the EEPROM. This write is initiated by writing a value of 358 to this parameter. Once the write is complete, the parameter value will go back to -1, indicating the completion of the backup. During the write operation, the parameter 3 changes its value constantly.

4 CUMULATIVE Wh

This parameter displays the cumulative energy consumption in watt hours. This parameter rolls over from 999 back to zero. When the watt hours roll over, the kilo watt hours are increased by one.

To obtain the total amount of energy used by the drive the parameters 4, 5, and 6 need to be scaled and added properly together. The total energy is:

Cumulative Energy = $1{,}000{,}000 \times MWh + 1{,}000 \times kWh + Wh$

In this formula, the MWh is the value from parameter `6 CUMULATIVE MWh`, and the kWh is the value from parameter `5 CUMULATIVE kWh`.

5 CUMULATIVE kWh

This parameter displays the cumulative energy consumption in kilo watt hours. This parameter rolls over from 999 back to zero. When the kilo watt hours roll over, the mega watt hours are increased by one.

6 CUMULATIVE MWh

This parameter displays the cumulative energy consumption in mega watt hours.

7 AI1 A/D CONVERSION

This parameter shows the binary value of the analog to digital conversion for analog input number one.

8 AI2 A/D CONVERSION

This parameter shows the binary value of the analog to digital conversion for analog input number two.

9 DIGITAL INPUTS

This parameter contains in its lowest 6 bits, the present status of all the 6 digital inputs. Bit 0 is digital input one, and bit 5 is digital input number 6.

10 RELAY OUTPUTS

This parameter contains in its lowest 3 bits, the present status of all the 3 digital relay outputs. Bit 0 is relay output one, and bit 2 is the relay output number 3.

11 – 17 CRI VERSION

These parameters show the Control Interface board firmware version as ASCII characters.

18 – 23 MCR VERSION

These parameters show the Motor Control board firmware version as ASCII characters.

24 SERIAL NUMBER HI
25 SERIAL NUMBER LO

These two parameters show the complete drive serial number. The ones, tens, and hundreds are in the SERIAL NUMBER LO parameters, while the thousands, tens of thousands, and higher digits are in SERIAL NUMBER HI.

Fault queue

The ACS/ACH 500 has a fault queue which records the latest three faults. The queue can be sent out using the serial communication protocol.

The fault queue values are number coded. The number coding for both warnings and faults is listed below.

Warnings

Warnings are diagnostic messages generated by the drive for the user. These messages do not cause the drive to fault, but they are still recorded into the fault queue.

Table 4–8 lists the warning code numbers which correspond to the warnings in the fault queue parameters.

Table 4–8 Drive warnings

Number	Text in display	Meaning
2	1 SAMI TEMP	Drive overheat
4	2 MOT STALL	Motor stall
6	3 MOT TEMP	Motor overload
12	6 UNDER LD	Under load warning
14	7 AI<2V/4mA	Analog input under 2V/4mA
16	8 EEPROM WR	EEPROM writing warning

Faults

Faults are non-normal situations detected by the drive, which cause the drive to fault. The fault information is both displayed on the drive panel, and is placed into the fault queue.

To get a more detailed description of the corresponding faults see *Installation & Start-up Manual* for ACS drives.

Table 4–9 lists the warning code numbers which correspond to the warnings in the fault queue parameters.

Table 4-9 Drive Faults

Number	Text in display	Meaning
1	1 START/STOP	Start/stop request and status contradiction
3	2 SAMI TEMP	Drive overheat heat sink
5	3 MOT STALL	Motor stall
7	4 MOT TEMP	Motor overload
13	7 UNDER LD	Under load
15	8 OVER CURR1	Over current
17	9 OVER VOLT	Over voltage
19	10 UNDER V 1	Under voltage trip
21	11 AI<2V/4mA	Analog input under 2V/4mA
31	16 POW RANG	Power range programming changed
33	17 RS-485	RS-485 serial communication fault
37	19 IN COMMS	Internal serial communication fault
39	20 CON INT	Control interface (HW) fault
41	21 MOT CONT	Motor controller (HW) fault
43	22 PAR REST	Parameter reading error
45	23 UNDER V 2	Unsuccessful DC-link charging
47	24 GND FAULT	Ground fault
49	25 EXT FLT	External fault

This page intentionally left blank.

Chapter 5 – Start-up procedure

This chapter describes safety considerations and the start-up procedures for the installation of the Modbus network update EPROM for the ACS and ACH 500 drives.

Overview

Modbus wiring

The Modbus network should be wired according to the information given in *Chapter 3 – Installation.*

To avoid any damages caused by loose wires, the following items should be checked before applying power to the drives:

- Check that there are no loose wires or threads on the serial connector.
- Verify that all wires and shields are well connected to the corresponding screw terminals.
- Verify that all wires are connected to the correct screws on the terminal block, and that no wires are swapped.

If there would be a possibility of a mechanical pull on the communication cable, this should be stopped by some method of strain relief. The terminal block is not designed for blocking the mechanical pull.

EPROM update

ACS500 Modbus network update comes as an EPROM update. This EPROM should replace the original EPROM on the drive.

ESD procedures must be followed during the change of the EPROM on the drive. The EPROM must be placed correctly onto the control board. Verify the alignment from the original EPROM residing on the drive.

The EPROM update must be done while the drive is disconnected from the power supply.

After the EPROM change, there will be a fault text PAR REST generated to the drive when the drive is first connected to power. After this fault message, the electrical power must be disconnected, which will clear the fault message. After the second power up the drive will operate normally, and the drive can be programmed normally.

Grounding

Grounding for the serial communication cable is described in *Chapter 3 – Installation.*

The drives must also be grounded from the frame. This grounding should go to the same grounding point for all of the drives connected on the communication network.

⚠ The drive control board is grounded at the factory using a ground wire from terminal block X50, pin 8 to the drive chassis ground. This wire must be removed for Modbus network operation.

Removal of this grounding wire will cause the control boards to float in their potentials. This must be remembered while connecting any possible analog or digital inputs to the drive.

The control boards are also connected together through the RS-485 wiring.

Programming

The drives must be programmed using the drive panel for the installation. The parameters that need to be considered for the serial interface are listed in *Chapter 4 – Programming.*

Chapter 6 – Fault Tracing

This chapter describes trouble shooting procedures for the Modbus network installation using diagnostics counters, fault queues, and drive status displays. Also some possible fault origins are discussed.

Fault diagnostics

This chapter concentrates on the problems and possible remedies for the serial communication connection for the ACS500 Modbus network. For other general fault diagnostics with the ACS 500 or ACH 500 drives consult the appropriate product manual.

The network problems can be caused by multiple sources. Some of these include:

- Loose connections,
- Incorrect wiring, including swapped wires,
- Bad grounding,
- Duplicate station numbers, and
- Incorrect programming and setup for drives or other devices on the network.

The major diagnostic features for fault tracing on the network include Group 10.8 External Communication parameters `5 BAD MESSAGE COUNTER` and `6 GOOD MESS COUNTER`.

This chapter will list some possible communication problems, how to identify them, and will list some possible corrections.

Normal operation

During normal operation of the network, the `GOOD MESS COUNTER` should be constantly advancing on all the stations, and the `BAD MESSAGE COUNTER` should not advance at all.

If problems exist, the `BAD MESSAGE COUNTER` will advance whenever a bad message packet is received, and the `GOOD MESS COUNTER` will advance for each good message packet received.

No Master station on line

How to diagnose: Neither the `GOOD MESS COUNTER` nor the `BAD MESSAGE COUNTER` increases on any of the stations.

How to correct: Check that the master station connected and properly programmed on the network. Verify that the cable is connected, and is not cut or short circuited.

Duplicate station

How to diagnose: Two or more slave stations cannot be addressed. Every time there is a read or write to one given station, the `BAD MESSAGE COUNTER` advances.

How to correct: Verify the station numbers of all stations. Change conflicting station numbers.

Improper grounding

How to diagnose: The `BAD MESSAGE COUNTER` is advancing. If the drives are turned on, the `BAD MESSAGE COUNTER` advances faster.

There might also be `CON INT`, `IN COMMS`, or `MOT CONT` faults coming randomly.

How to fix: Check the wiring, shield connections, and grounding. See that no noise is brought to the drive through analog or digital inputs or outputs. Verify that the grounding wire from screw 8 on terminal block X50 is removed.

Swapped wires

How to diagnose: The `GOOD MESS COUNTER` is not advancing. The `BAD MESSAGE COUNTER` is advancing.

How to fix: Check that the RS-485 line lines are not swapped. Verify that the control boards are connected together using the wire on screw 4 on the serial communication terminal block.

Summary

The problems described here cover the most usual problems on starting up the Modbus network. Intermittent problems might well be caused by marginally loose connections, vibration caused wear on wires, or especially through insufficient grounding and shielding on both the devices and on the communication cables.

If after basic troubleshooting, the problem continues, contact ABB technical support (800) 243 - 4384.

Appendix A – Parameter List

This chapter lists all the parameters used by the ACS 500 drive. This chapter is intended for reference purposes.

All the numbers read and written to the drive are in 16 bit integer format. The scaling and format of the data is described for each one of the parameters in the following table. The units of all the parameters are listed within this chapter. For the message communication it is also necessary to know the parameter address. These addressees are shown in the parameter list below.

For more detailed information, refer to the *ACS 500 Programming Manual*, and to *Chapter 4 – Programming* in this manual.

Parameters

The following table lists all of the parameters. For each parameter, the following information is provided:

- The parameter name, as it appears on the local drive panel.
- The unique register address for the Modbus access of the parameter. This is used for reads and writes as the address.
- The parameter name, as it appears on the local drive panel.
- The scaling for the parameter

Table A-9 Parameter Settings

Parameter	**Address**	**Alternative Settings**	**Units**
START-UP DATA			
A LANGUAGE	45001	0:FINNISH; 1:SWEDISH; 2:ENGLISH; 3:GERMANY; 4:ITALIAN; 5:SPANISH; 6:DUTCH; 7:FRENCH; 8:DANISH	LIST
B APPLICATIONS	45002	1:FACTORY; 2:HAND/AUTO; 3:PI-CONTROL; 4:OPT. PACK; 5:SEQ CTRL	LIST
C APPLIC. RESTORE	45003	0:NO; 1:YES	LIST

D SUPPLY VOLTAGE	45004	0:208; 1:220; 2:230; 3:240;
		0:380; 1:400; 2:415;
		0:440; 1:460; 2:480; 3:500
			LIST

E USER DISPLAY SCALE	45005	0 – 10000	100 = 100
F MOTOR CURRENT -FLA	45006	0 A – 1000 A (printed on the motor nameplate)	[0.1 A] 10 = 1.0A
G MOTOR POWER hp (kW)	45007	0.7 hp – 1340 hp (0.5 kW – 1000 kW) (printed on the motor nameplate)	[0.1 Hp] 10 = 1.0 Hp
H MOTOR POWER FACTOR	45008	0.1 – 1.0 (printed on the motor nameplate)	[0.01] 100 = 1.0
I MOTOR BASE FREQ.	45009	30 Hz – 500 Hz (printed on the motor nameplate)	[10 Hz] 6 = 60 Hz
J MOTOR BASE R.P.M.	45010	200 RPM – SYNC. SPEED (printed on the motor nameplate)	[rpm] 1728 = 1728 rpm
K MOTOR NOM. VOLTAGE	45011	110 V – 575 V (printed on the motor nameplate)	[V] 230 = 230 V

Parameter	Address	Alternative Settings	Units
INVISIBLE PARAMETERS			
1 STATUS WORD	45101	0x0000 - 0xFFFF	See table 4-6
2 COMMAND WORD	45102	0x0000 - 0xFFFF	See table 4-7
3 BACKUP	45103	-1 / 0 / 358	Number
4 CUMULATIVE Wh	45104	Cumulative Wh	[Wh]
5 CUMULATIVE kWh	45105	Cumulative kWh	[kWh]
6 CUMULATIVE MWh	45106	Cumulative MWh	[MWh]
7 AI1 A/D CONVERSION	45107	0 - 32767	Number
8 AI2 A/D CONVERSION	45108	0 - 32767	Number
9 DIGITAL INPUTS	45109	0x0000 - 0x003F	Digital inputs
10 RELAY OUTPUTS	45110	0x0000 - 0x0003	Relay outputs
11 – 17 CRI VERSION	45111 – 45117	'A' – 'Z'	Control Interface board firmware version (30.4.1)
18 – 23 MCR VERSION	45118 – 45123	'A' – 'Z'	Motor Control board firmware version (30.4.2)
24 SERIAL NUMBER HI	45124	0 – 9999	Serial Number, High part
25 SERIAL NUMBER LO	45125	0 – 999	Sereial Number, Low part

Parameter	Address	Alternative Settings	Units
OPERATING DATA			
1 OUTPUT FREQUENCY	40001	Hz	[0.01 Hz] 100 = 1.00 Hz
2 SPEED	40002	RPM; %; USER SCALING	1728 = 1728 rpm
3 MOTOR CURRENT	40003	A	[0.1 A] 10 = 1.0 A
4 % RATED TORQUE	40004	%	[%] 10 = 10 %
5 % RATED POWER	40005	%	[%] 10 = 10 %
6 DC BUS VOLTAGE	40006	% OF RATED NOMINAL	[%] 100 = 100 % of nominal
7 OUTPUT VOLTAGE	40007	V	[V] 1 = 1 V
8 DRIVE TEMPERATURE	40008	degrees C and F	[°C] 25 = 25 °C
9 CONTROL LOCATION	40009	0:KEYPAD R1; 1:KEYPAD PI; 2:EXTERNAL	LIST
10 KEYPAD REF 1	40010	Hz	[0.01 Hz] 100 = 1.00 Hz
11 KEYPAD PI (REF 2)	40011	%	[0.01 %] 100 = 1.00 %
12 EXT REF 1 OR 2	40012	0:REF1; 1:REF2	LIST
13 EXTERNAL REF 1	40013	Hz	[0.01 Hz] 100 = 1.0 Hz
14 EXTERNAL REF 2	40014	%	[0.01 %] 100 = 1.00 %
15 RUN TIME	40015	h/min	[h] 100 = 100 hours
16 KILOWATT HOURS	40016	kWh	[kWh] 1 = 1 kWh
17 LAST-RECD FAULT	40017	FAULT; WARNING	See table s 4-8 and 4-9
18 SECOND-RECD FAULT	40018	FAULT; WARNING	See table s 4-8 and 4-9
19 FIRST-RECD FAULT	40019	FAULT; WARNING	See table s 4-8 and 4-9
20 PARAMETER LOCK	40020	OPEN xxx; LOCKED xxx; OPEN; LOCKED	0: OPEN 1:LOCKED

Parameter	Address	Alternative Settings	Units
10 CONTROL CONNECTIONS			
10.1 START/STOP/DIRECTION			
10.1.1 EXT 1 STRT/STP/DIR	41101	0:NOT SEL; 1:DI1; 2:DI1,2; 3:DI1P,2P; 4:DI1P,2P,3; 5:DI1P,2P,3P; 6:DI6; 7:DI6,5; 8:KEYPAD; 9:STD COMM	LIST
10.1.2 EXT 2 STRT/STP/DIR	41102	0:NOT SEL; 1:DI1; 2:DI1,2; 3:DI1P,2P; 4:DI1P,2P,3; 5:DI6; 6:DI6,5; 7:KEYPAD; 8:STD COMM	LIST
10.1.3 LOC/EXT DIRECTION	41103	0:REVERSE; 1:FORWARD; 2:REQUEST; 3:FAST REV	LIST
10.2 EX REFERENCE SELECT			
10.2.1EXT 1/EXT 2 SELECT	41201	0:OP DATA 12; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6; 7:STD COMM	LIST
10.2.2 EXTERNAL REF1 SEL	41202	0:OP DATA 13; 1:AI1; 2:AI2; 3:AI1/JOYST; 4:DI3U,4D(R); 5:DI3U,4D; 6:DI5U,6D; 7:STD COMM	LIST
10.2.3 EXT REF1 MINIMUM	41203	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.2.4 EXT REF1 MAXIMUM	41204	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.2.5 EXT REF1 OFFSET	41205	-30 Hz – 30 Hz	[0.01 Hz] 100 = 1.00 Hz
10.2.6 EXT REF1 GAIN	41206	-100 % – 100 %	[0.01 %] 100 = 1.00 %
10.2.7 EXTERNAL REF2 SEL	41207	0:OP DATA 14; 1:AI1; 2:AI2; 3:DI3U,4D(R); 4:DI3U,4D; 5:DI5U,6D	LIST
10.2.8 EXT REF2 MINIMUM	41208	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.2.9 EXT REF2 MAXIMUM	41209	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.2.10 EXT REF1 OFFSET	41210	-30 Hz – 30 Hz	[0.01 Hz] 100 = 1.00 Hz
10.2.11 EXT REF1 GAIN	41211	-100 % – 100 %	[0.01 %] 100 = 1.00 %

Parameter	Address	Alternative Settings	Units
10.3 PRESET SPEEDS			
10.3.1 PRESET SPEED SEL	41301	0:NOT SEL; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6; 7:DI1,2; 8:DI3,4; 9:DI5,6; 10:DI1,2,3; 11:DI3,4,5; 12:DI4,5,6	LIST
10.3.2 PRESET SPEED 1	41302	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.3.3 PRESET SPEED 2	41303	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.3.4 PRESET SPEED 3	41304	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.3.5 PRESET SPEED 4	41305	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.3.6 PRESET SPEED 5	41306	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.3.7 PRESET SPEED 6	41307	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.3.8 PRESET SPEED 7	41308	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
10.4 SYSTEM CONTR INPUTS			
10.4.1 RUN ENABLE	41401	0:YES; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6	LIST
10.4.2 FAULT RESET SELECT	41402	0:NOT SEL; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6; 7:ON STOP; 8:STD COMM.	LIST
10.4.3 PARAM. LOCK SEL	41403	0:OP DATA 20; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6; 7:STD COMM.	LIST
10.4.4 EXTERNAL FAULT	41404	0:NOT SEL; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6	LIST

Parameter	Address	Alternative Settings	Units
10.5 ANALOG INPUTS			
10.5.1 MINIMUM AI1	41501	0:0 V/0 mA; 1:2 V/4 mA; 2:READ INPUT	LIST
10.5.2 MAXIMUM AI1	41502	0:10 V/20 mA; 1:READ INPUT	LIST
10.5.3 RC FILTER ON AI1	41503	0.01s – 10s	[0.01 s] 100 = 1.00 s
10.5.4 INVERT AI1	41504	0:NO; 1:YES	LIST
10.5.5 MINIMUM AI2	41505	0:0 V/0 mA; 1:2 V/4 mA; 2:READ INPUT	LIST
10.5.6 MAXIMUM AI2	41506	0:10 V/20 mA; 1:READ INPUT	LIST
10.5.7 RC FILTER ON AI2	41507	0.01s – 10s	[0.01 s] 100 = 1.00 s
10.5.8 INVERT AI2	41508	0:NO; 1:YES	LIST
10.6 RELAY OUTPUTS			
10.6.1 RELAY RO1 OUTPUT	41601	0:NOT USED; 1:READY; 2:RUN; 3:FAULT; 4:FAULT(-1); 5:FAULT(RST); 6:STALL FLT; 7:MOT OT FLT; 8:OT FAULT; 9:FAULT/WARN; 10:WARNING; 11:OT WARNING; 12:REVERSED; 13:EXT. CTRL; 14:REF 2 SEL; 15:PRESET SPD; 16:DC BUS LIM; 17:FREQ 1 LIM; 18:FREQ 2 LIM; 19:CURR LIMIT; 20:REF 1 LIMIT; 21:REF 2 LIMIT; 22:AT SPEED; 23:(P&F AUTOM); 24:AI<MIN	LIST
10.6.2 RELAY RO2 OUTPUT	41602		
10.6.3 RELAY RO3 OUTPUT	41603		

Parameter	Address	Alternative Settings	Units
10.7 ANALOG OUTPUTS			
10.7.1 ANALOG OUTPUT 1	41701	0:NOT USED; 1:OUT FREQ; 2:MOT SPEED; 3:OUT CURR; 4:MOT TORQ; 5:MOT POWER; 6:V/DC BUS; 7:MOTOR VOLT; 8:REFERENCE	LIST
10.7.2 SCALE AO1	41702	10% – 1000%	[%] 100 = 100 %
10.7.3 MINIMUM AO1	41703	0:0 mA; 1:4 mA	LIST
10.7.4 RC FILTER ON AO1	41704	0.01s – 10s	[0.01 s] 100 = 1.00 s
10.7.5 INVERT AO1	41705	0:NOT USED; 1:OUT FREQ; 2:MOT SPEED; 3:OUT CURR; 4:MOT TORQ; 5:MOT POWER; 6:V/DC BUS; 7:MOTOR VOLT; 8:REFERENCE	LIST
10.7.6 ANALOG OUTPUT 2	41706	10% – 1000%	[%] 100 = 100 %
10.7.7 SCALE AO2	41707	0:0 mA; 1:4 mA	LIST
10.7.8 MINIMUM AO2	41708	0.01s – 10s	[0.01 s] 100 = 1.00 s
10.7.9 RC FILTER ON AO2	41709	0:NO; 1:YES	LIST
10.7.10 INVERT AO2	41710	0:NOT USED; 1:OUT FREQ; 2:MOT SPEED; 3:OUT CURR; 4:MOT TORQ; 5:MOT POWER; 6:V/DC BUS; 7:MOTOR VOLT; 8:REFERENCE	LIST

Parameter	Address	Alternative Settings	Units
10.8 EXT COMMUNICATION			
10.8.1 DRIVE ID-NUMBER	41801	1 - 247	Number 1 = 1
10.8.2 BIT RATE SELECT	41802	0:1200; 1:2400; 3:4800; 4:9600	LIST
10.8.3 PARITY	41803	0:NONE; 1:FAULT; 2:FAULT+STOP	LIST
10.8.4 COMMS FAULT FUNCT	41804	0:NONE; 1:FAULT; 2:FAULT+STOP	LIST
10.8.5 BAD MESSAGES COUNTER	41805	0 – 65535	Number 1 = 1
10.8.6 GOOD MESSAGES COUNTER	41806	0 - 65535	Number 1 = 1

Parameter	Address	Alternative Settings	Units
20 DRIVE PARAMETERS			
20.1 FREQ/CURRENT LIMITS			
20.1.1 MINIMUM FREQUENCY	42101	0 Hz – MAX. FREQ.	[0.01 Hz] 100 = 1.00 Hz
20.1.2 MAXIMUM FREQUENCY	42102	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.1.3 FREQUENCY RANGE	42103	0:0 Hz – 120 Hz; 1:0 Hz – 500 Hz (ACS 501 only)	LIST
20.1.4 CURRENT LIMIT	42104	0.5 – 2.0 x I_N (ACS 500)	[% of I_n] 100 = 100% I_n
20.2 START/STOP			
20.2.1 START FUNCTION	42201	0:RAMP; 1:FLYING; 2:TORQ BOOST; 3:FLYING+TQB	LIST
20.2.2 TORQUE BOOST CURR	42202	0.5 – 2.0 x I_N (ACS 500)	[% of I_n] 100 = 100% I_n
20.2.3 STOP FUNCTION	42203	0:COAST; 1:RAMP; 2:DC BRAKE	LIST
20.2.4 BRAKE CHOPPER	42204	0:NO; 1:YES	LIST
20.2.5 DC HOLD	42205	0:OFF; 1:ON	LIST
20.2.6 DC HOLD VOLTAGE	42206	0.01 – 0.1 x V_N	[0.1 % V_{Supply}] 100 = 10% * V_{Supply}
20.2.7 DC BRAKE VOLTAGE	42207	0.01 – 0.1 x V_N	[0.1 % V_{Supply}] 100 = 10% * V_{Supply}
20.2.8 DC BRAKE TIME	42208	0s – 250s	[s] 1 = 1 s

Parameter	Address	Alternative Settings	Units
20.3 ACCEL/DECEL			
20.3.1 ACC/DEC 1 0R 2 SEL	42301	0:NOT SEL; 1:DI1; 2:DI2; 3:DI3; 4:DI4; 5:DI5; 6:DI6	LIST
20.3.2 ACC/DEC RAMP SHAPE	42302	0:LINEAR; 1:S1; 2:S2; 3:S3	LIST
20.3.3 ACCEL TIME 1	42303	0.1s – 1800s	[0.1 s] 10 = 1.0 s
20.3.4 DECEL TIME 1	42304	0.1s – 1800s	[0.1 s] 10 = 1.0 s
20.3.5 ACCEL TIME 2	42305	0.1s – 1800s	[0.1 s] 10 = 1.0 s
20.3.6 DECEL TIME 2	42306	0.1s – 1800s	[0.1 s] 10 = 1.0 s
20.3.7 ACCEL REF2 TIME	42307	0.1s – 1800s	[0.1 s] 10 = 1.0 s
20.3.8 DECEL REF2 TIME	42308	0.1s – 1800s	[0.1 s] 10 = 1.0 s
20.4 MOTOR CONTROL			
20.4.1 SWITCHING FREQ	42401	1.0 kHz – 12.0 kHz (ACS 501); 3.0 kHz (ACS 502)	[100 Hz] 30 = 3.0 kHz
20.4.2 MAX OUTPUT VOLTAGE	42402	0.15 – 1.05 x V_N	[0.1% V_{Supply}] 1000 = 100.0 % V_{Supply}
20.4.3 V/HZ RATIO	42403	0:LINEAR; 1:SQUARED; 2:AUTOMATIC	LIST
20.4.4 FIELD WEAK POINT	42404	30 Hz – 500 Hz	[Hz] 60 = 60 Hz
20.4.5 IR COMPENSATION	42405	0:NO; 1:MANUAL; 2:AUTOMATIC	LIST
20.4.6 IR COMP VOLTAGE	42406	0.01 – 0.15 x V_N	[0.1% V_{Supply}] 100 = 10.0 % V_{Supply}
20.4.7 IR COMP RANGE	42407	0 Hz – FWP	[Hz] 60 = 60 Hz
20.4.8 SLIP COMPENSATION	42408	0:OFF; 1:ON	LIST
20.4.9 NOMINAL SLIP	42409	0.1% – 10%	[0.1 %] 100 = 10.0 %
20.4.10 VOLTAGE LIMIT	42410	0:OFF; 1:ON	LIST

Parameter	Address	Alternative Settings	Units
20.5 CRITICAL FREQUENCIES			
20.5.1 CRIT FREQ SELECT	42501	0:OFF; 1:ON	LIST
20.5.2 CRIT FREQ 1 LOW	42502	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.3 CRIT FREQ 1 HIGH	42503	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.4 CRIT FREQ 2 LOW	42504	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.5 CRIT FREQ 2 HIGH	42505	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.6 CRIT FREQ 3 LOW	42506	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.7 CRIT FREQ 3 HIGH	42507	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.8 CRIT FREQ 4 LOW	42508	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.9 CRIT FREQ 4 HIGH	42509	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.10 CRIT FREQ 5 LOW	42510	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
20.5.11 CRIT FREQ 5 HIGH	4211	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz

Parameter	Address	Alternative Settings	Units
30 PROTECTION/INFORMAT			
30.1 FAULT FUNCTION			
30.1.1 SERIAL FAULT FUNC	43101	0:STOP; 1:PRE SPEED7	LIST
30.1.2 AI <MIN FUNCTION	43102	0:NO; 1:WARNING; 2:FAULT; 3:PRE SPEED7; 4:LAST SPEED	LIST
30.1.3 MOT TEMP FLT FUNC	431013	0:NO; 1:WARNING; 2:FAULT	LIST
30.1.4 MOTOR THERM TIME	43104	300s – 10000s	[s] 120 = 120 s
30.1.5 MOTOR LOAD CURVE	43105	50% – 150%	[%] 50 = 50%
30.1.6 ZERO SPEED LOAD	43106	40% – MOTOR LOAD CURVE	[%] 100 = 100%
30.1.7 BREAK POINT	43107	1 Hz – 500 Hz	[Hz] 60 = 60 Hz
30.1.8 STALL FUNCTION	43108	0:NO; 1:WARNING; 2:FAULT	LIST
30.1.9 STALL CURRENT	43109	0 – 1.5 x I_N	[0.01 I_N] 150 = 1.5 * I_N
30.1.10 STALL TIME/FREQ	43110	0:10s/15 Hz; 2:20s/25 Hz; 3:30s/35 Hz	LIST
30.1.11 UNDERLOAD FUNC	43111	0:NO; 1:WARNING; 2:FAULT	LIST
30.1.12 UNDERLOAD TIME	43112	0 – 600s	[s] 100 = 100 s
30.1.13 UNDERLOAD CURVE	43113	1 – 5	Number
30.2 AUTOMATIC RESET			
30.2.1 NUMBER OF RESETS	43201	0 – 5	Number
30.2.2 TIME WINDOW	43202	1s – 180s	[s] 10 = 10 s
30.2.3 TIME BETW. RESET ATTEMPTS	43203	0s – 120s	[s] 10 = 10 s
30.2.4 OVERVOLTAGE	43204	0:NO; 1:YES	LIST
30.2.5 UNDERVOLTAGE	43205	0:NO; 1:YES	LIST
30.2.6 OVERCURRENT	43206	0:NO; 1:YES	LIST
30.2.7 AI SIGNAL <MIN	43207	0:NO; 1:YES	LIST

Parameter	Address	Alternative Settings	Units
30.3 SUPERVISION			
30.3.1 OUTPUT FREQ 1 FUNC	43301	0:NO; 1:LOW LIMIT; 2:HIGH LIMIT	LIST
30.3.2 OUTPUT FREQ 1 LIM	43302	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.1 Hz] 10 = 1.0 Hz
30.3.3 OUTPUT FREQ 2 FUNC	43303	0:NO; 1:LOW LIMIT; 2:HIGH LIMIT	LIST
30.3.4 OUTPUT FREQ 2 LIM	43304	0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.1 Hz] 10 = 1.0 Hz
30.3.5 CURRENT FUNCTION	43305	0:NO; 1:LOW LIMIT; 2:HIGH LIMIT	LIST
30.3.6 CURRENT LIMIT	43306	0 – 2 x I_N (ACS 500)	[% I_N] 100 = 100% I_N
30.3.7 REF1 FUNCTION	43307	0:NO; 1:LOW LIMIT; 2:HIGH LIMIT	LIST
30.3.8 REF1 LIMIT	43308	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
30.3.9 REF2 FUNCTION	43309	0:NO; 1:LOW LIMIT; 2:HIGH LIMIT	LIST
30.3.10 REF2 LIMIT	43310	0% – 100%	[0.01 %] 100 = 1.00 %
30.3.11 SUPERVIS MESSAGES	43311	0:OFF; 1:ON	LIST
30.4 INFORMATION			
30.4.1 CRI PROG VERSION	43401	(Version in Drive)	Not supported on communication. See also 45117 – 45123
30.4.2 MC PROG VERSION	43402	(Version in Drive)	
30.4.3 TEST DATE	43403	(Date Tested)	

Parameter	Address	Alternative Settings	Units
40 APPLICATION PARAMETERS		(CAN BE SEEN ONLY WITH APPLICATION MACROS)	
40.1 PI-CONTROL		(CAN BE SEEN ONLY WITH PI-CONTROL MACRO)	
40.1.1 PI-CONT GAIN	44101	3% – 800%	[0.1 %] 10 = 1.0 %
40.1.2 PI-CONT I-TIME	44102	0.02s – 320s	[0.01 s] 100 = 1.00 s
40.1.3 PI-CONT MIN LIMIT	44103	0 Hz – PI-CONT MAX LIMIT	[0.01 Hz] 100 = 1.00 Hz
40.1.4 PI-CONT MAX LIMIT	44104	0 Hz – 120 Hz / 0 Hz – 500 Hz (ACS 501); 0 Hz – 120 Hz (ACS 502)	[0.01 Hz] 100 = 1.00 Hz
40.1.5 ERROR VALUE INVERT	44105	0:NO; 1:YES	LIST
40.1.6 ACTUAL VALUE SEL	44106	0:ACT1; 1:ACT1-ACT2; 2:ACT1+ACT2; 3:ACT1*ACT2	LIST
40.1.7 ACTUAL 1 INPUT	44107	0:NO; 1:AI1; 2:AI2	LIST
40.1.8 ACTUAL 2 INPUT	44108	0:NO; 1:AI1; 2:AI2	LIST
40.1.9 ACT 1 MIN SCALE	44109	-1600.0% – 1600.0%	[0.1 %] 10 = 1.0 %
40.1.10 ACT 1 MAX SCALE	44110	-1600.0% – 1600.0%	[0.1 %] 10 = 1.0 %
40.1.11 ACT 2 MIN SCALE	44111	-1600.0% – 1600.0%	[0.1 %] 10 = 1.0 %
40.1.12 ACT 2 MAX SCALE	44112	-1600.0% – 1600.0%	[0.1 %] 10 = 1.0 %

This page intentionally left blank.

Appendix B – Modbus Protocol

This chapter describes the serial Modbus protocol. This chapter is intended for people who need to program a serial Modbus master for their own control system.

The material of this chapter is copyrighted by Modicon, and is used by permission of AEG Schneider Automation (Modicon). This material is included in the Modicon publication *Modicon Modbus Protocol Reference Guide* (PI-MBUS-300 Rev. E).

Introducing Modbus Protocol

The Modbus protocol is a serial master – slave protocol. This appendix describes the Modbus protocol only to the level, which is required to fully access the ACS 500 (and ACH 500) drives. The Modbus protocol defines what is serially transmitted on the communication link. The physical interface for the ACS 500 drives is half-duplex RS-485.

Transactions on Modbus Networks

Standard Modbus ports on Modicon controllers use an RS-232C compatible serial interface that defines connector pinouts, cabling, signal levels, transmission baud rates, and parity checking. Controllers can be networked directly or via modems.

Controllers communicate using a master – slave technique, in which only one device (the master) can initiate transactions (called 'queries'). The other devices (the slaves) respond by supplying the requested data to the master, or by taking the action requested in the query. Typical master devices include host processors and programming panels. Typical slaves include programmable controllers.

The master can address individual slaves, or can initiate a broadcast message to all slaves. Slaves return a message (called a 'response') to queries that are addressed to them individually. Responses are not returned to broadcast queries from the master.

The Modbus protocol establishes the format for the master's query by placing into it the device (or broadcast) address, a function code defining the requested action, any data to be sent, and an error-checking field. The slave's response message is also constructed using Modbus protocol. It contains fields confirming the action taken, any data to be returned, and an error-checking field. If an error occurred in receipt of the message, or if the slave is unable to perform the requested action, the slave will construct an error message and send it as its response.

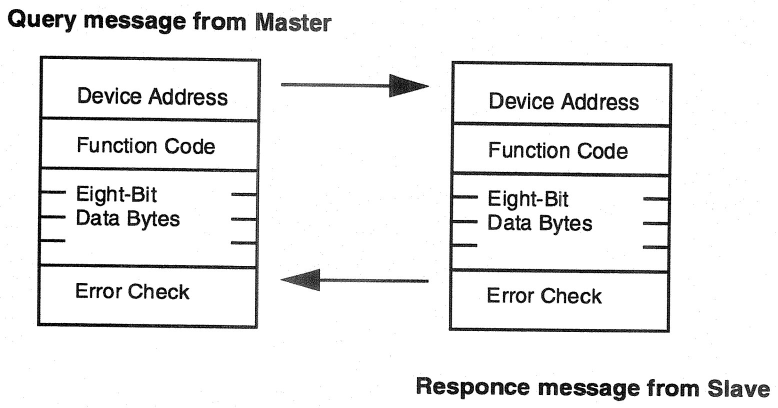


Figure B-1 Master–Slave Query-Response Cycle

The Query: The function code in the query tells the addressed slave device what kind of action to perform. The data bytes contain any additional information that the slave will need to perform the function. For example, function code 03 will query the slave to read holding registers and respond with their contents. The data field must contain the information telling the slave which register to start at and how many registers to read. The error check field provides a method for the slave to validate the integrity of the message contents.

The Response: If the slave makes a normal response, the function code in the response is an echo of the function code in the query. The data bytes contain the data collected by the slave, such as register values or status. If an error occurs, the function code is modified to indicate that the response is an error response, and the data bytes contain a code that describes the error. The error check field allows the master to confirm that the message contents are valid.

The Two Serial Transmission Modes

Controllers can be setup to communicate on standard Modbus networks using either of two transmission modes: ASCII or RTU. Users select the desired mode, along with the serial port communication parameters (baud rate, parity mode, etc.), during configuration of each controller. The mode and serial parameters *must be the same for all devices on a Modbus network.* The selection of ASCII or RTU mode pertains only to standard Modbus networks. It defines the bit contents of message fields transmitted serially on those networks. It determines how information will be packed into the message fields and decoded.

The ACS 500 drive supports only the RTU mode. Only the RTU mode is described in this document.

RTU Mode

When controllers are setup to communicate on a Modbus network using RTU (Remote Terminal Unit) mode, each 8-bit byte in a message contains two 4-bit hexadecimal characters. The main advantage of this mode is that its greater character density allows better data throughput than ASCII for the same baud rate. Each message must be transmitted in a continuous stream.

The format for each byte in RTU mode is:

Coding System:	8-bit binary, hexadecimal 0-9, A-F Two hexadecimal characters contained in each 8-bit field of the message
Bits per Byte:	1 start bit 8 data bits, least significant bit sent first 1 bit for even/odd parity; no bit for no parity 1 stop bit if parity is used; 2 bits if no parity
Error Check Field:	Cyclical Redundancy Check (CRC)

Modbus Message Framing

In either of the two serial transmission modes (ASCII or RTU), a Modbus message is placed by the transmitting device into a frame that has a known beginning and ending point. This allows receiving devices to begin at the start of the message, read the address portion and determine which device is addressed (or all devices, if the message is broadcast), and to know when the message is completed. Partial messages can be detected and errors can be set as a result.

RTU Framing

In RTU mode, messages start with a silent interval of at least 3.5 character times. This is most easily implemented as a multiple of character times at the baud rate that is being used on the network (shown as T1-T2-T3-T4 in the figure below). The first field then transmitted is the device address.

The allowable characters transmitted for all fields are hexadecimal 0-9, A-F. Networked devices monitor the network bus continuously, including during the 'silent' intervals. When the first field (the address field) is received, each device decodes it to find out if it is the addressed device.

Following the last transmitted character, a similar interval of at least 3.5 character times marks the end of the message. A new message can begin after this interval.

The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before completion of the frame, the receiving device flushes the incomplete message and assumes that the next byte will be the address field of a new message.

Similarly, if a new message begins earlier than 3.5 character times following a previous message, the receiving device will consider it a continuation of the previous message. This will set an error, as the value in the final CRC field will not be valid for the combined messages. A typical message frame is shown below.

START	ADDRESS	FUNCTION	DATA	CRC CHECK	END
T1–T2–T3–T4	8 BITS	8 BITS	$n \times 8$ BITS	16 BITS	T1–T2–T3–T4

Figure B-2 Message Frame

How the Address Field is Handled

The address field of a message frame contains eight bits (RTU). Valid slave device addresses are in the range of 0 – 247 decimal. The individual slave devices are assigned addresses in the range of 1 – 247. A master addresses a slave by placing the slave address in the address field of the message. When the slave sends its response, it places its own address in this address field of the response to let the master know which slave is responding.

Address 0 is used for the broadcast address, which all slave devices recognize. When Modbus protocol is used on higher level networks, broadcasts may not be allowed or may be replaced by other methods. For example, Modbus Plus uses a shared global database that can be updated with each token rotation.

How the Function Field is Handled

The function code field of a message frame contains eight bits (RTU). Valid codes are in the range of 1 – 255 decimal. Of these, some codes are applicable to all Modicon controllers, while some codes apply only to certain models, and others are reserved for future use. ACS 500 drive supports codes 3, 6, and 16 (0x03, 0x06, and 0x10 Hex).

When a message is sent from a master to a slave device the function code field tells the slave what kind of action to perform. Example is to read a group of outputs.

When the slave responds to the master, it uses the function code field to indicate either a normal (error-free) response or that some kind of error occurred (called an exception response). For a normal response, the slave simply echoes the original function code. For an exception response, the slave returns a code that is equivalent to the original function code with its most-significant bit set to a logic 1.

For example, a message from master to slave to read a group of holding registers would have the following function code:

0000 0011	(Hexadecimal 03)

If the slave device takes the requested action without error, it returns the same code in its response. If an exception occurs, it returns:

1000 0011	(Hexadecimal 83)

In addition to its modification of the function code for an exception response, the slave places a unique code into the data field of the response message. This tells the master what kind of error occurred, or the reason for the exception.

The master device's application program has the responsibility of handling exception responses. Typical processes are to post subsequent retries of the message, to try diagnostic messages to the slave, and to notify operators.

Contents of the Data Field

The data field is constructed using sets of two hexadecimal digits, in the range of 00 to FF hexadecimal. These are made from one RTU character, according to the network's serial transmission mode.

The data field of messages sent from a master to slave devices contains additional information which the slave must use to take the action defined by the function code. This can include items like discrete and register addresses, the quantity of items to be handled, and the count of actual data bytes in the field.

For example, if the master requests a slave to read a group of holding registers (function code 03), the data field specifies the starting register and how many registers are to be read. If the master writes to a group of registers in the slave (function code 10 hexadecimal), the data field specifies the starting register, how many registers to write, the count of data bytes to follow in the data field, and the data to be written into the registers.

If no error occurs, the data field of a response from a slave to a master contains the data requested. If an error occurs, the field contains an exception code that the master application can use to determine the next action to be taken.

The data field can be nonexistent (of zero length) in certain kinds of messages. For example, in a request from a master device for a slave to respond with its communications event log (function code 0B hexadecimal), the slave does not require any additional information. ACS 500 does not support the function code 0B hexadecimal. The function code alone specifies the action.

Contents of the Error Checking Field

Two kinds of error-checking methods are used for standard Modbus networks. The error checking field contents depend upon the method that is being used.

ASCII

When ASCII mode is used for character framing, the error checking field contains two ASCII characters. The error check characters are the result of a Longitudinal Redundancy Check (LRC) calculation that is performed on the message contents, exclusive of the beginning 'colon' and terminating CRLF characters.

The LRC characters are appended to the message as the last field preceding the CRLF characters.

RTU

When RTU mode is used for character framing, the error checking field contains a 16-bit value implemented as two 8-bit bytes. The error check value is the result of a Cyclical Redundancy Check calculation performed on the message contents.

The CRC field is appended to the message as the last field in the message. When this is done, the low-order byte of the field is appended first, followed by the high-order byte. The CRC high-order byte is the last byte to be sent in the message.

Additional information about error checking is contained later in this appendix.

How Characters are Transmitted Serially

When messages are transmitted on standard Modbus serial networks, each character or byte is sent in this order (left to right):

Least Significant Bit (LSB) ... Most Significant Bit (MSB)

With RTU character framing, the bit sequence is:

With Parity Checking

Start	1	2	3	4	5	6	7	8	Par	Stop

Without Parity Checking

Start	1	2	3	4	5	6	7	8	Stop	Stop

Error Checking Methods

Standard Modbus serial networks use two kinds of error checking. Parity checking (even or odd) can be optionally applied to each character. Frame checking (CRC) is applied to the entire message. Both the character check and message frame check are generated in the master device and applied to the message contents before transmission. The slave device checks each character and the entire message frame during receipt.

The master is configured by the user to wait for a predetermined time-out interval before aborting the transaction. This interval is set to be long enough for any slave to respond normally. If the slave detects a transmission error, the message will not be acted upon. The slave will not construct a response to the master. Thus the time-out will expire and allow the master's program to handle the error. Note that a message addressed to a nonexistent slave device will also cause a time-out. A proper time-out value for ACS 500 drives is 200 ms.

Parity Checking

Users can configure controllers for Even or Odd Parity checking, or for No Parity checking. This will determine how the parity bit will be set in each character.

If either Even or Odd Parity is specified, the quantity of 1 bits will be counted in the data portion of each character (eight for RTU). The parity bit will then be set to a 0 or 1 to result in an Even or Odd total of 1 bits.

For example, these eight data bits are contained in an RTU character frame:

1100 0101

The total quantity of 1 bits in the frame is four. If Even Parity is used, the frame's parity bit will be a 0, making the total quantity of 1 bits still an even number (four). If Odd Parity is used, the parity bit will be a 1, making an odd quantity (five).

When the message is transmitted, the parity bit is calculated and applied to the frame of each character. The receiving device counts the quantity of 1 bits and sets an error if they are not the same as configured for that device. All devices on the Modbus network must be configured to use the same parity check method.

Note that parity checking can only detect an error if an odd number of bits are picked up or dropped in a character frame during transmission. For example, if Odd Parity checking is employed, and two 1 bits are dropped from a character containing three 1 bits, the result is still an odd count of 1 bits.

If No Parity checking is specified, no parity bit is transmitted and no parity check can be made. An additional stop bit is transmitted to fill out the character frame.

CRC Checking

In RTU mode, messages include an error-checking field that is based on a Cyclical Redundancy Check (CRC) method. The CRC field checks the contents of the entire message. It is applied regardless of any parity check method used for the individual characters of the message.

The CRC field is two bytes, containing a 16-bit binary value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The receiving device recalculates a CRC during receipt of the message, and compares the calculated value to the actual value it received in the CRC field. If the two values are not equal, an error results.

The CRC is started by first pre-loading a 16-bit register to all 1's. Then a process begins of applying successive 8-bit bytes of the message to the current contents of the register. Only the eight bits of data in each character are used for generating the CRC. Start and stop bits, and the parity bit if one is used, do not apply to the CRC.

During generation of the CRC, each 8-bit character is exclusive ORed with the register contents. Then the result is shifted in the direction of the least significant bit (LSB), with a zero filled into the most significant bit (MSB) position. The LSB is extracted and examined. If the LSB was a 1, the register is then exclusive ORed with a preset, fixed value. If the LSB was a 0, no exclusive OR takes place.

This process is repeated until eight shifts have been performed. After the last (eighth) shift, the next 8-bit byte is exclusive ORed with the register's current value, and the process repeats for eight more shifts as described above. The final contents of the register, after all the bytes of the message have been applied, is the CRC value.

In ladder logic, the CKSM function calculates a CRC from the message contents. For applications using host computers, a detailed example of CRC generation is contained later in this appendix.

Modbus Function Formats

This chapter describes in detail the data content on every Modbus message supported by ACS 500 drives.

How Numerical Values are Expressed

Unless specified otherwise, numerical values (such as addresses, codes, or data) are expressed as *decimal values in the text of this section*. They are expressed as *hexadecimal values in the message fields of the figures*.

Data Addresses in Modbus Messages

All data addresses in Modbus messages are referenced to zero. The first occurrence of a data item is addressed as item number zero. For example:

- The coil known as 'coil 1' in a programmable controller is addressed as coil 0000 in the data address field of a Modbus message.
- Coil 127 decimal is addressed as coil 007E hex (126 decimal).
- Holding register 40001 is addressed as register 0000 in the data address field of the message. The function code field already specifies a 'holding register' operation. Therefore the '4XXXX' reference is implicit.
- Holding register 40108 is addressed as register 006B hex (107 decimal).

Field Contents in Modbus Messages

Figure B-3 *"Master Query with RTU Framing"* shows an example of a Modbus query message. Figure B-4 *"Slave Response with RTU Framing"* is an example of a normal response. Both examples show the field contents in hexadecimal, and also show how a message is be framed in RTU mode.

The master query is a Read Holding Registers request to slave device address 06. The message requests data from three holding registers, 40108 through 40110. Note that the message specifies the starting register address as 0107 (006B hex).

The slave response echoes the function code, indicating this is a normal response. The 'Byte Count' field specifies how many *8-Bit data items* are being returned. It shows the count of 8-bit bytes to follow in the data for RTU.

For example, the value 63 hex is sent as one 8-bit byte in RTU mode (01100011). The 'Byte Count' field counts this data as one 8-bit item, regardless of the character framing method (ASCII or RTU).

How to Use the Byte Count Field: When you construct responses in buffers, use a Byte Count value that equals the count of 8-bit bytes in your message data. The value is exclusive of all other field contents, including the Byte Count field. Figure B-4 *"Slave Response with RTU Framing"* shows how the byte count field is implemented in a typical response.

QUERY

Field Name	Example (Hex)	RTU 8-Bit Field
Header		None
Slave Address	06	0000 0110
Function	03	0000 0011
Starting Address Hi	00	0000 0000
Starting Address Lo	6B	0110 1011
No. of Registers Hi	00	0000 0000
No. of Registers Lo	03	0000 0011
Error Check		CRC (16 bits)
Trailer		None
	Total Bytes:	8

Figure B-3 Master Query with RTU Framing

RESPONSE

Field Name	Example (Hex)	RTU 8-Bit Field
Header		None
Slave Address	06	0000 0110
Function	03	0000 0011
Byte Count	06	0000 0110
Data Hi	02	0000 0010
Data Lo	2B	0010 1011
Data Hi	00	0000 0000
Data Lo	00	0000 0000
Data Hi	00	0000 0000
Data Lo	00	0000 0000
Error Check		CRC (16 bits)
Trailer		None
	Total Bytes:	11

Figure B-4 Slave Response with RTU Framing

Function Codes

The ACS 500 drive supports three Modbus function codes. These allow the Master to read and write 16-bit integer values to the drive.

03 Read Holding Registers

Reads the binary contents of holding registers (4X references) in the slave. Broadcast is not supported.

Query

The query message specifies the starting register and quantity of registers to be read. Registers are addressed starting at zero: registers 1–16 are addressed as 0–15.

Here is an example of a request to read registers 40108–40110 from slave device 17:

QUERY

Field Name	Example (Hex)
Slave Address	11
Function	03
Starting Address Hi	00
Starting Address Lo	6B
No. of Points Hi	00
No. of Points Lo	03
Error Check CRC	CRC (16-Bits)

Figure B-5 Read Holding Registers – Query

Response

The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

Data is scanned in the slave at the rate of 125 registers per scan for 984-X8X controllers (984–685, etc.), and at the rate of 32 registers per scan for all other controllers. The response is returned when the data is completely assembled.

Here is an example of a response to the previous query:

RESPONSE

Field Name	Example (Hex)
Slave Address	11
Function	03
Byte Count	06
Data Hi (Register 40108)	02
Data Lo (Register 40108)	2B
Data Hi (Register 40109)	00
Data Lo (Register 40109)	00
Data Hi (Register 40110)	00
Data Lo (Register 40110)	64
Error Check CRC	CRC (16-Bits)

Figure B-6 Read Holding Registers- Response

The contents of register 40108 are shown as the two byte values of 02 2B hex, or 555 decimal. The contents of registers 40109–40110 are 00 00 and 00 64 hex, or 0 and 100 decimal.

06 Preset Single Register

Presets a value into a single holding register (4X reference). When broadcast, the function presets the same register reference in all attached slaves.

Query

The query message specifies the register reference to be preset. Registers are addressed starting at zero: register 1 is addressed as 0.

The requested preset value is specified in the query data field. ACS 500 drives use 16-bit values.

Here is an example of a request to preset register 40002 to 00 03 hex in slave device 17:

QUERY

Field Name	Example (Hex)
Slave Address	11
Function	06
Register Address Hi	00
Register Address Lo	01
Preset Data Hi	00
Preset Data Lo	03
Error Check CRC	CRC (16-Bits)

Figure B-7 Preset Single Register – Query

Response

The normal response is an echo of the query, returned after the register contents have been preset.

Here is an example of a response to the query on the opposite page:

RESPONSE

Field Name	Example (Hex)
Slave Address	11
Function	06
Register Address Hi	00
Register Address Lo	01
Preset Data Hi	00
Preset Data Lo	03
Error Check CRC	CRC (16-Bits)

Figure B-8 Preset Single Register – Response

16 (10 Hex) Preset Multiple Regs

Presets values into a sequence of holding registers (4X references). When broadcast, the function presets the same register references in all attached slaves.

The ACS 500 drive allows only one register to be written at one time using one Preset Multiple Regs function.

Query

The query message specifies the register references to be preset. Registers are addressed starting at zero: register 1 is addressed as 0.

The requested preset values are specified in the query data field. ACS 500 use 16-bit values. Data is packed as two bytes per register.

Here is an example of a request to preset one register starting at 40002 to 00 0A in slave device 17:

QUERY

Field Name	Example (Hex)
Slave Address	11
Function	10
Starting Address Hi	00
Starting Address Lo	01
No. of Registers Hi	00
No. of Registers Lo	01
Byte Count	02
Data Hi	00
Data Lo	0A
Error Check CRC	CRC (16-Bits)

Figure B-9 Preset Multiple Registers – Query

Response

The normal response returns the slave address, function code, starting address, and quantity of registers preset.

Here is an example of a response to the query shown above.

RESPONSE

Field Name	Example (Hex)
Slave Address	11
Function	10
Starting Address Hi	00
Starting Address Lo	01
No. of Registers Hi	00
No. of Registers Lo	01
Error Check CRC	CRC (16-Bits)

Figure B-10 Preset Multiple Registers- Response

Exception Responses

Except for broadcast messages, when a master device sends a query to a slave device it expects a normal response. One of four possible events can occur from the master's query:

1. If the slave device receives the query without a communication error, and can handle the query normally, it returns a normal response.
2. If the slave does not receive the query due to a communication error, no response is returned. The master program will eventually process a time-out condition for the query.
3. If the slave receives the query, but detects a communication error (parity, LRC, or CRC), no response is returned. The master program will eventually process a time-out condition for the query.
4. If the slave receives the query without a communication error, but cannot handle it (for example, if the request is to read a non-existent coil or register), the slave will return an exception response informing the master of the nature of the error. The exception response message has two fields that differentiate it from a normal response:

Function Code Field: In a normal response, the slave echoes the function code of the original query in the function code field of the response. All function codes have a most-significant bit (MSB) of 0 (their values are all below 80 hexadecimal). In an exception response, the slave sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 80 hexadecimal higher than the value would be for a normal response. With the function code's MSB set, the master's application program can recognize the exception response and can examine the data field for the exception code.

Data Field: In a normal response, the slave may return data or statistics in the data field (any information that was requested in the query). In an exception response, the slave returns an exception code in the data field. This defines the slave condition that caused the exception. Figure B-11 *"Master Query and Slave Exception Response"* shows an example of a master query and slave exception response.

The field examples are shown in hexadecimal.

QUERY

Byte	Contents	Example
1	Slave Address	0A
2	Function	01
3	Starting Address Hi	04
4	Starting Address Lo	A1
5	No. of Coils Hi	00
6	No. of Coils Lo	01
7	LRC	4F

EXCEPTION RESPONSE

1	Slave Address	0A
2	Function	81
3	Exception Code	02
4	LRC	73

Figure B-11 Master Query and Slave Exception Response

In this example, the master addresses a query to slave device 10 (0A hex). The function code (01) is for a Read Coil Status operation. It requests the status of the coil at address 1245 (04A1 hex). Note that only that one coil is to be read, as specified by the number of coils field (0001).

If the coil address is non-existent in the slave device, the slave will return the exception response with the exception code shown (02). This specifies an illegal data address for the slave. For example, if the slave is a 984-385 with 512 coils, this code would be returned.

A listing of Modicon exception codes is in Table B-1 *"Standard Exception Codes"*.

Code	Name	Meaning
01	ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the slave. If a Poll Program Complete command was issued, this code indicates that no program function preceded it.
02	ILLEGAL DATA ADDRESS	The data address received in the query is not an allowable address for the slave.
03	ILLEGAL DATA VALUE	A value contained in the query data field is not an allowable value for the slave.
04	SLAVE DEVICE FAILURE	An unrecoverable error occurred while the slave was attempting to perform the requested action.
05	ACKNOWLEDGE	The slave has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a time-out error from occurring in the master. The master can next issue a Poll Program Complete message to determine if processing is completed.
06	SLAVE DEVICE BUSY	The slave is engaged in processing a long duration program command. The master should retransmit the message later when the slave is free.
07	NEGATIVE ACKNOWLEDGE	The slave cannot perform the program function received in the query. This code is returned for an unsuccessful programming request using function code 13 or 14 decimal. The master should request diagnostic or error information from the slave.
08	MEMORY PARITY ERROR	The slave attempted to read extended memory, but detected a parity error in the memory. The master can retry the request, but service may be required on the slave device.

Table B-9 Standard Exception Codes

CRC Generation

The Cyclical Redundancy Check (CRC) field is two bytes, containing a 16-bit binary value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The receiving device recalculates a CRC during receipt of the message, and compares the calculated value to the actual value it received in the CRC field. If the two values are not equal, an error results.

The CRC is started by first preloading a 16-bit register to all 1's. Then a process begins of applying successive 8-bit bytes of the message to the current contents of the register. Only the eight bits of data in each character are used for generating the CRC. Start and stop bits, and the parity bit if one is used, do not apply to the CRC.

During generation of the CRC, each 8-bit character is exclusive ORed with the register contents. Then the result is shifted in the direction of the least significant bit (LSB), with a zero filled into the most significant bit (MSB) position. The LSB is extracted and examined. If the LSB was a 1, the register is then exclusive ORed with a preset, fixed value. if the LSB was a 0, no exclusive OR takes place.

This process is repeated until eight shifts have been performed. After the last (eighth) shift, the next 8-bit character is exclusive ORed with the register's current value, and the process repeats for eight more shifts as described above. The final contents of the register, after all the characters of the message have been applied, is the CRC value.

A procedure for generating a CRC is:

1. Load a 1 6-bit register with FFFF hex (all 1's). Call this the CRC register.
2. Exclusive OR the first 8-bit byte of the message with the low-order byte of the 16-bit CRC register, putting the result in the CRC register.
3. Shift the CRC register one bit to the right (toward the LSB), zero-filling the MSB. Extract and examine the LSB.
4. (If the LSB was 0): Repeat Step 3 (another shift).
   (if the LSB was 1): Exclusive OR the CRC register with the polynomial value A001 hex (101 0 0000 0000 0001).
5. Repeat Steps 3 and 4 until 8 shifts have been performed. When this is done, a complete 8-bit byte will have been processed.
6. Repeat Steps 2 through 5 for the next 8-bit byte of the message. Continue doing this until all bytes have been processed.
7. The final contents of the CRC register is the CRC value.

Placing the CRC into the Message

When the 16-bit CRC (2 8-bit bytes) is transmitted in the message, the low-order byte will be transmitted first, followed by the high-order byte. For example, if the CRC value is 1241 hex (0001 0010 0100 0001):

Addr	Func	Data Count	Data	Data	Data	Data	CRC Lo 41	CRC Hi 12

Figure B-12 CRC Byte Sequence

Example

An example of a C language function performing CRC generation is shown on the following pages. All of the possible CRC values are preloaded into two arrays, which are simply indexed as the function increments through the message buffer. One array contains all of the 256 possible CRC values for the high byte of the 16-bit CRC field, and the other contains all of the values for the low byte. Indexing the CRC in this way provides faster execution than would be achieved by calculating a new CRC value with each new character from the message buffer.

The function takes two arguments:

`unsigned char *puchMsg`

A pointer to the message buffer containing binary data to be used for generating the CRC

`unsigned short usDataLen`

The quantity of bytes in the message buffer.

The function returns the CRC as a type unsigned short.

```
/* Table of CRC values for high-order byte */
static unsigned char auchCRCHi [ ] = {
0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,
0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,0xC0,
0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,
0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,
0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,
0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,
0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,
0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,
0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,
0x40,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,
0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,
0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,
0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,
0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,
0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,
0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,
0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,
0x40
};
/* Table of CRC values for low-order byte*/
static char auchCRCLo [ ] = {
0x00,0xC0,0xC1,0x01,0xC3,0x03,0x02,0xC2,0xC6,0x06,0x07,0xC7,0x05,0xC5,0xC4,
0x04,0xCC,0x0C,0x0D,0xCD,0x0F,0xCF,0xCE,0x0E,0x0A,0xCA,0xCB,0x0B,0xC9,0x09,
0x08,0xC8,0xD8,0x18,0x19,0xD9,0x1B,0xDB,0xDA,0x1A,0x1E,0xDE,0xDF,0x1F,0xDD,
0x1D,0x1C,0xDC,0x14,0xD4,0xD5,0x15,0xD7,0x17,0x16,0xD6,0xD2,0x12,0x13,0xD3,
0x11,0xD1,0xD0,0x10,0xF0,0x30,0x31,0xF1,0x33,0xF3,0xF2,0x32,0x36,0xF6,0xF7,
0x37,0xF5,0x35,0x34,0xF4,0x3C,0xFC,0xFD,0x3D,0xFF,0x3F,0x3E,0xFE,0xFA,0x3A,
0x3B,0xFB,0x39,0xF9,0xF8,0x38,0x28,0xE8,0xE9,0x29,0xEB,0x2B,0x2A,0xEA,0xEE,
0x2E,0x2F,0xEF,0x2D,0xED,0xEC,0x2C,0xE4,0x24,0x25,0xE5,0x27,0xE7,0xE6,0x26,
0x22,0xE2,0xE3,0x23,0xE1,0x21,0x20,0xE0,0xA0,0x60,0x61,0xA1,0x63,0xA3,0xA2,
0x62,0x66,0xA6,0xA7,0x67,0xA5,0x65,0x64,0xA4,0x6C,0xAC,0xAD,0x6D,0xAF,0x6F,
0x6E,0xAE,0xAA,0x6A,0x6B,0xAB,0x69,0xA9,0xA8,0x68,0x78,0xB8,0xB9,0x79,0xBB,
0x7B,0x7A,0xBA,0xBE,0x7E,0x7F,0xBF,0x7D,0xBD,0xBC,0x7C,0xB4,0x74,0x75,0xB5,
0x77,0xB7,0xB6,0x76,0x72,0xB2,0xB3,0x73,0xB1,0x71,0x70,0xB0,0x50,0x90,0x91,
0x51,0x93,0x53,0x52,0x92,0x96,0x56,0x57,0x97,0x55,0x95,0x94,0x54,0x9C,0x5C,
0x5D,0x9D,0x5F,0x9F,0x9E,0x5E,0x5A,0x9A,0x9B,0x5B,0x99,0x59,0x58,0x98,0x88,
0x48,0x49,0x89,0x4B,0x8B,0x8A,0x4A,0x4E,0x8E,0x8F,0x4F,0x8D,0x4D,0x4C,0x8C,
0x44,0x84,0x85,0x45,0x87,0x47,0x46,0x86,0x82,0x42,0x43,0x83,0x41,0x81,0x80,
0x40
);
```

```
unsigned short CRC16(puchMsg, usDataLen)
unsigned char *puchMsg ;                     / * message to calculate CRC upon* /
unsigned short usDataLen;                    /* quantity of bytes in message*/
{
    unsigned char uchCRCHi = 0xFF;           /* high byte of CRC initialized*/
    unsigned char uchCRCLo = 0xFF;           /* low byte of CRC initialized*/
    unsigned uIndex;                         /* will index into CRC lookup table*/

    while (usDataLen--)                      /* pass through message buffer*/
    {
        uIndex = uchCRCHi ^ *puchMsgg++;/* calculate the CRC*/
        uchCRCHi = uchCRCLo ^ auchCRCHi [uIndex] ;
        uchCRCLo = auchCRCLo [uIndex] ;
    }

    return (uchCRCHi << 8 | uchCRCLo);
}
```

This page intentionally left blank.

ACS500-MODBUS-US-04b
EFFECTIVE 3/1/95
SUPERCEDES 10/1/94

ABB

ABB Drives Inc.
Standard Drives Division
16250 West Glendale Drive
New Berlin, WI 53151
Telephone: (414) 785-3416
Fax: (414) 785-0397

ABB Drives
P.O. Box 184
00381 Helsinki
FINLAND

Telephone:	+358-0-5641
Telefax:	+358-0-564 2681
Telex:	+57-12440502 str sf

Printed in U.S.A.