ABB 传动

安装启动指南

Modbus 适配器模块
NMBA-01

Modbus 适配器模块 NMBA-01

安装启动指南

3ABD 00004103 R0225

生效 : 1999-01-02
替代 : 1997-01-10

概述：

本章介绍在安装及使用 NMBA-01 Modbus 适配器时必须遵守的安全守则，如果忽略，将造成意外的人身伤害和死亡，或者损坏变频器，电机以及被传动设备。在操作和使用适配器时之前，务必阅读本章的内容。

警告和注意事项：

本手册将安全须知分为两部分。一部分是警告，用于说明严重的情况，在进行操作时，若不注意采取措施，将导致严重的故障，人身伤害甚至死亡。一部分是注意事项，出现在需要读者特别注意的地方以及某些问题需要补充说明的地方。注意事项不如警告重要，但是也不应忽视。

警告：

对于那些将造成严重的人身伤害或设备损坏的场合，本手册将用以下标志提醒读者：

危险电压警告：警告有高电压存在，会造成人身伤害或设备损坏，标志旁边的内容讲述了避免危险的方法。

一般警告：对除了电气原因之外，造成人身伤害或设备损坏的情况给予警告，标志旁边的内容讲述了避免危险的方法。

静电释放警告：对可能损坏设备的静电释放给予警告，标志旁边的内容讲述了避免危险的方法。

*注意*当说明书中有需要特别注意的地方，或有需要补充说明的问题时，用以下标志提示读者：

注意！ 意思是应对特殊问题予以特别注意。

补充说明： 给出补充信息，或者指出该主题有可供参考的更多信息。

常规安全须知

警告! 模块的电气安装和维护工作都必须由专业技术人员完成。 工作时请小心，忽略安全守则会导致人体伤害或死亡。

传动装置和相关设备都必须适当接地 。

不要带电操作传动装置，在切断主电源之后，应该至少等待五分钟，待中间电路电容放电完毕再进行操作。最好在进行工作之前检查变频器是否放电完毕(使用电压表)。

在主电路接通电源时，无论电机是否运行，电机电缆端子都处于危险的高电压状态。

当传动装置主电路断电时，在其内部仍存在由外部控制电路引入的危险电压，所以操作时应该注意。否则会造成人身伤害或死亡。

传动装置有一些自动复位功能。如果选用，在故障消失后，传动将自动复位并重新起动运行。当系统中其它设备不能适应此类运行，或者可能因此发生危险时，请不要使用自动复位功能。

目录

第六章 - 通信

第七章 - 故障跟踪

*附录 **A** - 技术数据*

附录 B - Modbus 协议

*附录 **C** -环境条件*

第一章 - 引言

概述

本章对 Modbus 适配器模块 NMBA-01 的安装与启动指南进行了介绍 。

面向的读者

本指南是为在 ABB 公司的传动设备上安装，调试以及使用 Modbus 适配器的人员准备的，读者应具有电工原理，电气接线，电气传动，传动控制盘以及 Modbus 通讯协议等方面的基础知识 。

内容介绍

本指南对 Modbus 适配器模块的安装与启动作了介绍 。

在安装适配器之前，假设传动设备已经安装完毕并能够运行 . 对于传动装置的安装与启动，请参考有关手册 。

安全须知 在本指南的前几页，对指南中所使用的各种警告和注意事项的格式进行了说明。同时也说明了在安装和操作 NMBA-01 模块的过程中应遵守的安全准则。

第一章 – 介绍 包括这本手册的简要介绍。

第二章 – 概述 简要介绍了 Modbus 适配器模块 NMBA-01，发货清单以及制造厂商的责任。

第三章 – 机械安装 包括模块放置和安装的指导。

第四章 – 电气安装 包括模块接线，总线的终端及接地指导。

第五章 – 编程 介绍在通过适配器通讯前，如何为主站和传动设备的通信进行编程。

第六章 – 通信 说明如何通过 NMBA-01 传送数据。

第七章 – 故障跟踪 说明了如何使用状态指示 LED 来跟踪 NMBA-01 的故障。

附录 A – 技术数据 包括关于 Modbus 模块的技术信息。

*附录 B – **Modbus** 协议* 解释了 Modbus 协议。

附录 C 环境条件 介绍 Modbus 模块在运输，存储以及使用期间所允许的环境条件。

本指南中使用的术语

参数 参数是传动设备的一种操作指令，参数可以通过控制盘或 NMBA-01 模块读出和编程。

通信模块 通信模块是一种器件（如现场总线适配器）的参数名或参数选项名，传动设备可以通过它连接到一个外部串行通讯网络（如现场总线），通信模块的功能可由一个传动参数来激活。

NMBA-01 Modbus 适配器模块 NMBA-01 Modbus 适配器模块是 ABB 公司传动产品可选的现场总线适配器之一。NMBA-01 是一个器件，ABB 公司传动产品通过它可以连入 Modbus 串行通讯总线。

数据集与数据字 数据集是 NMBA-01 模块和传动之间通过 DDCS 链路传送的一组数据，每一个数据集由三个 16 位字（亦即数据字）组成。控制字（有时被称为命令字）和状态字，给定值和实际值（见第六章）都是一种数据字；有些数据字的内容是用户可定义的。

广播写 Modbus 网络允许主机同时对所有从机进行写入操作，这一写入操作称为广播写。广播写时，从机不向主机反馈确认信息。

4XXXX 寄存器区 Modicon PLC 有一个带符号的整数数据区，这一数据区用于模拟输出模块或存储临时性数据或者给定值。这些寄存器位于从 40001 开始的地址区域。最后一个寄存器地址取决于可供使用的存贮器容量，但是最大不超过 49999。传动装置通过在这个寄存器地址区读写参数来仿真这个寄存器区。

第二章 - 概述

概述

本章介绍了 Modbus 适配器模块 NMBA-01，并且给出了制造商的质量保证和责任条款（关于 Modbus 的协议，请参阅附录 B)。

Modbus 简介

Modbus 是一种异步串行协议。Modbus 协议不规定物理接口，典型的物理接口是 RS-232 和 RS-485。NMBA-01 使用 RS-485 接口。

NMBA-01 Modbus 适配器模块

NMBA-01 Modbus 适配器模块是 ABB 传动装置的一个可选件，它实现了传动装置和 Modbus 系统之间的连接。在 Modbus 网络中，传动装置被认为是从机。通过 NMBA-01 Modbus 适配器模块，我们可以：

* 向传动装置发送控制命令（起动，停止，允许运行等）。
* 向传动装置发送速度或力矩给定信号。
* 向传动装置中的 PID 调节器发送给定信号和实际值信号。
* 从传动装置中读取状态信息和实际值。
* 改变传动参数。
* 对传动装置进行故障复位。
* 进行多传动控制。

NMBA-01 Modbus 适配器模块所支持的 Modbus 命令和服务将在第六章进行讨论。传动装置所支持的命令，请参阅有关用户手册。

适配器模块安装在标准的导轨上，至于是安装在传动单元的内部还是外部，取决于传动装置的配置。对于模块安装选项，请参考传动装置用户手册。

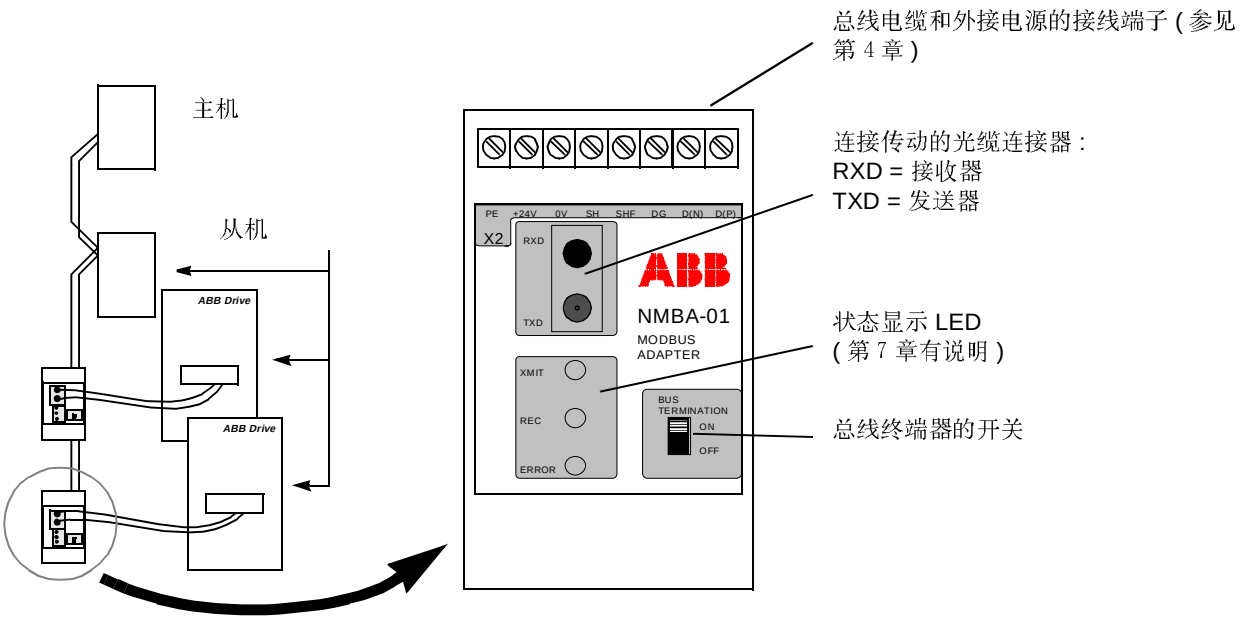

图 2-1 Modbus 连接的结构和 NMBA-01 Modbus 适配器模块。

兼容性 NMBA-01 与下列设备兼容：

* ACS300
* ACS400
* ACS600 单传动
* ACS600 多传动
* ACS600 位置控制 (ACP600)
* ACS600 提升控制 (ACC600)
* ACS600 泵与风机控制 (ACF600)
* DCS500
* 所有支持 Modicon Modbus 串行通讯协议的主站点。

交货检查 NMBA-01 适配器模块的包装箱内包括：

* Modbus 适配器模块 NMBA-01
* 两对 (四件) 光缆
* 安装导轨
* NMBA-01 安装启动指南。

质量保证和责任

ABB 传动装置和可选件的质量保证仅包括制造缺陷。 制造商对由于运输或拆包造成的损坏不负任何责任。

在任何情况下，制造商对由于使用不当，安装不当，异常的温度，粉尘，腐蚀等环境条件，以及由于超过额定容量使用而造成的损坏或故障不负任何责任。制造商也不对上述损坏的附带损失负任何责任。

从发货之日期起，制造商的质保期是 12 个月，最多不超过 18 个月。

用户当地的 ABB 公司或分销商可能有不同的质保期，这已经在他们的销售条款，条件，以及质保条款中说明。

用户有任何关于 ABB 传动的问题，请与当地的分销商或者 ABB 办公室联系。

手册中的技术数据与规定在出版时是有效的。ABB 公司保留今后变更的权力。

第三章 - 机械安装

概述

本章包括模块的安装指导。模块既可以安装在传动单元的内部，也可以安装在传动单元外部，取决于传动装置。对于模块安装选项，请参考传动装置用户手册。

在传动单元外部安装

为模块选择安装位置时，注意下列问题：

* 必须满足光缆的安装要求（参见第四章）。在模块包装中所包括的光缆长度限制了模块和传动单元之间的距离。
* 注意模块和传动单元周围的自由空间（与相邻设备或墙壁的最小间距为 10mm)。
* 应该考虑环境条件（参见附录 C)。模块的防护等级是 IP 20 。
* 模块的地线与安装导轨通过接地夹相连（见*图 3-1*)，安装导轨必须接地。如果导轨的基础接地不良，则应为安装导轨单独接地。地线应尽量短，其截面不应小于 6 平方毫米。*注意：不能使用实芯硬导线，只能使用多芯软导线。*

安装指导：

1. 关断柜体中的所有危险电压。
2. 固定导轨， 并且确保适当的接地(参见上面的指导)。
3. 把模块推在导轨上。 通过用改锥拉开锁住的弹簧， 能够将模块从导轨上卸下(参见*图 3-1*)。

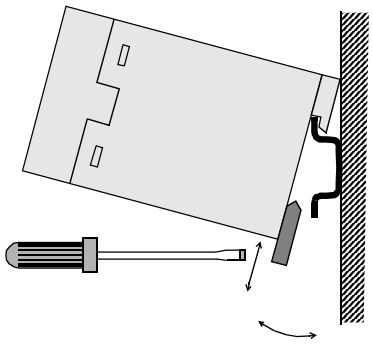

图 3-1　　安装和拆下模块

在传动单元内部安装

传动单元内部的安装工作只能由一个合格的电工完成。

警告！注意电容器组缓慢地释放的电压。注意由外部控制电路通过传动单元的输入输出端子带进的危险电压。

警告！不要触摸印刷电路板。 集成电路对静电放电极端敏感。

安装指导：

1. 停止运行传动。
2. 关断传动的主电源。关断与输入输出相连接的所有危险电压。
3. 等待五分钟以保证在中间电路中的电容器放电完毕。
4. 拆下传动的正面面板。
5. 保证主电电缆， 电机电缆以及电容器组(UDC+ 和UDC-)未接通电源。
6. 为模块定位(参见 传动装置用户手册)。 把安装导轨固定好（注意模块周围的自由空间，与相邻设备或墙壁的最小间距为 10mm)。
7. 把模块推在导轨上。用改锥拉开锁簧， 能够将模块从导轨上卸下(参见*图 3–1*)。

概述

本章包括：

* 电缆安装指导
* 总线终端器安装指导
* NMBA-01 模块接线与接地指导以及总线电缆接地指导

警告！安装模块前，请关闭传动主电源。等待 5 分钟，使传动装置的中间电路放电。关闭与传动单元 I/O 相连的所有外部电源。

电缆布线

使总线电缆尽量远离电机电缆。避免平行布线。在电缆入口处使用套管。小心处理光缆。拔下光缆时，不要抓光缆，应抓住连接器。光缆头对灰尘非常敏感，不要用手触摸光缆头。

光缆的长期拉伸负载为 1N，短期弯曲半径为 25mm。

总线终端器

如果 NMBA-01 模块安装于总线的一端，应将内置的终端电阻接通，否则应将终端电阻断开，终端电阻用于防止信号在电缆中反射。

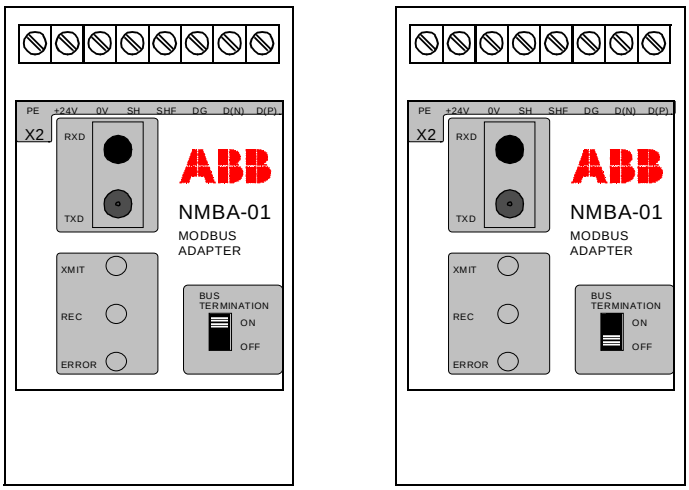

图 4-1　　终端电阻的接通（左）及断开（右）

NMBA-01 通讯连接

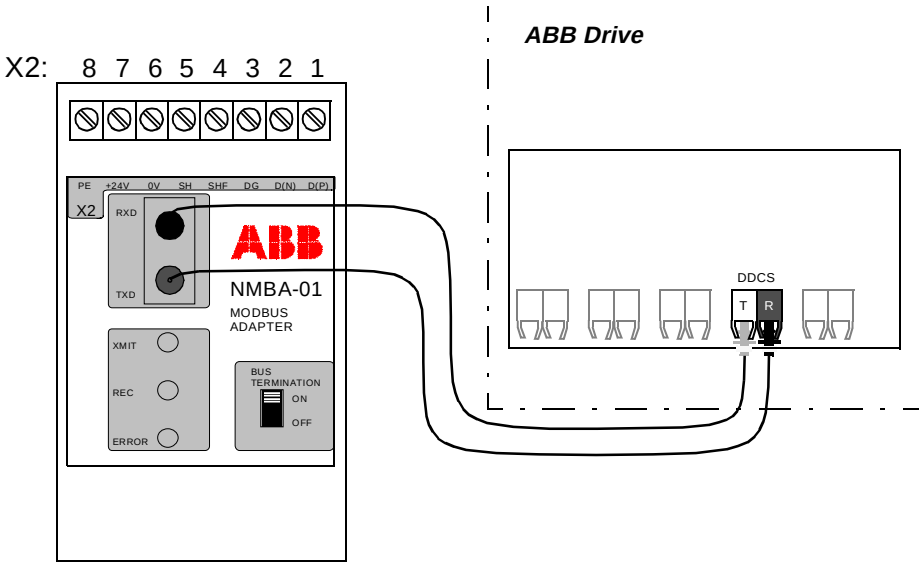

图 4-1 适配器与 ABB 传动装置之间的光缆连接。

NMBA-01 使用光缆与传动装置相连。有关传动装置内部的对应端子，请参阅传动手册。

总线电缆及外部供电电源连接于 NMBA-01 的端子 X2。

表 4-1 端子 X2 的说明

X2		说明
1	D(P)	D(P) = B = 数据线正极（ 双绞线 1 号导体 ） D(N) = A = 数据线负极（ 双绞线 2 号导体 ） DG = 数据地
2	D(N)	
3	DG	
4	SHF	电缆屏蔽交流地线（ 通过 RC 滤波器 ）
5	SH	电缆屏蔽地线（ 直接接地 ）
6	0V	模块的供电电源 (24 V d.c. ± 10 %); 屏蔽电缆 .
7	+24 V	
8	PE	地线

接地

NMBA-01 模块的地线接到了安装导轨上，如果导轨被固定在金属底板上，则模块便自动接地了，这时不需要额外的地线。如果导轨被固定在没有接地的底板上，则导轨必须就近接地。而且，地线必须连接到与供电电缆的屏蔽层相同的接地端子上。（见 3-1 页）

NMBA-01 上有几个内置接地端子（ 见图 4-3 及 4-4 ）

* **PE** 端子 在内部已连接到 NMBA-01 模块的地端，一般情况下这个端子不需要外部接线。
* **SH** 端子 在内部已连接到 NMBA-01 模块的地端，如果 Modbus 总线电缆的屏蔽层没有在其它站点直接接地，则 SH 端子可用于 Modbus 总线电缆屏蔽层接地。
* **SHF** 端子 通过 RC 滤波器在内部接到 NMBA-01 的地端，这一端子一般用于 Modbus 总线电缆屏蔽层接地。
* **DG** 端子 与 NMBA-01 模块地隔离。这一端子用于连接总线电缆的第三根导线。第三根导线—数据地—为所有总线上的模块提供了一个公共参考电位。

---

*注意：*建议使用数据地，它可以提高抗噪声能力，见图 4-3 和图 4-4。

Modbus 电缆屏蔽接地

Modbus 电缆屏蔽层只允许在一个站点直接接地，在其它站点，屏蔽层应通过 RC 滤波器接地。

下列配置图有两种接线方案，最好使用三线连接方案，因为它有较好的抗噪声能力。

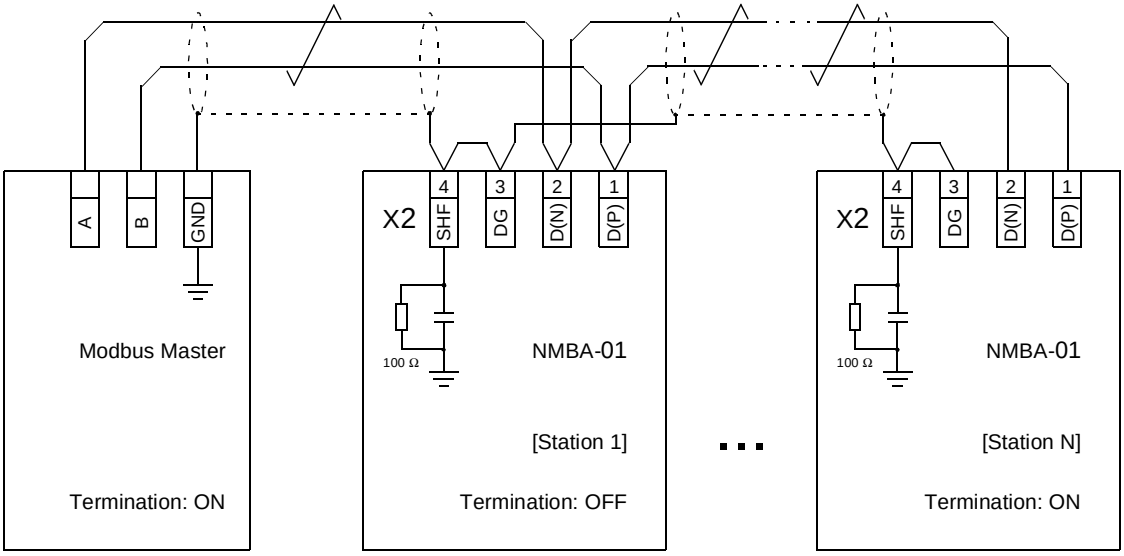

图 4-2　两线制接线方案 (Modbus 实例)

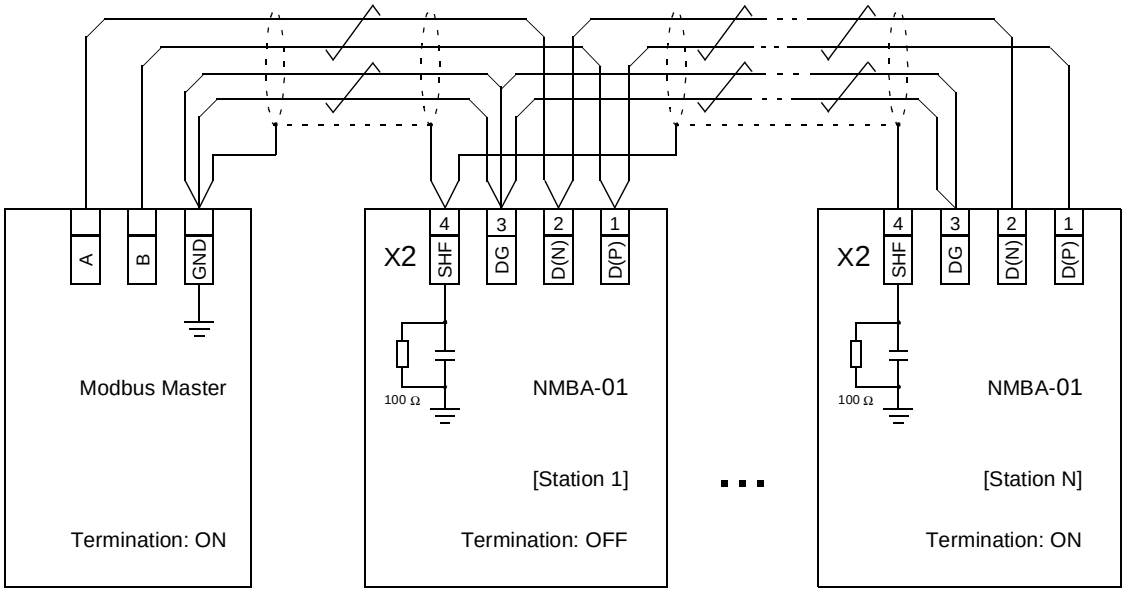

图 4-3　三线制接线方案 (Modbus 实例)

第五章 - 编程

概述

本章介绍 Modbus 主站的配置信息及传动装置通过 NMBA-01 Modbus 适配器模块通讯的配置信息。

系统配置

按第三章及第四章的说明安装好 NMBA-01 Modbus 适配器模块后，还需要为在主站和从机作一些准备工作。

对于主站的通讯系统配置，请参考主站有关手册。

Modbus 通讯配置

激活通讯模块的详细步骤取决于所用的传动装置（一般情况下，需要调整一个参数，以激活通讯功能，参见传动手册）。当传动装置与 NMBA-01 之间建立通讯连络之后，有几个配置被拷贝到传动装置。首先应该检查这些参数（见表 5-1），必要时进行调整。在表格之后，对这些参数可能的选项进行了讨论。（注意，新设的参数在下次上电时才能效）

---

*注意：*参数的组号、序号以及调整过程随着传动装置的不同而不同，请参见具体的传动手册。

表 5-1　　NMBA-01 参数配置

现场总线 参数编号	参数名称	可选设置	缺省设置
1	MODULE TYPE		NMBA-01 Vx.x
2	MODBUS MODE	**(0)** RTU wdg:flt; **(1)** RTU wdg:rst	**(0)** RTU wdg:flt
3	STATION NUMBER	1 – 247	1
4	BAUD RATE	**(0)** 1200; **(1)** 2400; **(2)** 4800; **(3)** 9600; **(4)** 19200	**(3)** 9600
5	PARITY	**(0)** EVEN; **(1)** ODD; **(2)** NONE 2 S.BIT; **(3)** NONE 1 S.BIT	**(2)** NONE 2 S.BIT
6	GOOD MESSAGES	0 – 32767	0
7	BAD MESSAGES	0 – 32767	0
8	DDCS CHANNEL	**(0)** CH0; **(1)** CH3	**(0)** CH0

MODULE TYPE
(模块型号)

本参数是由传动装置识别出的模块型号及软件版本号(如果这个参数无定义，则不能在传动装置与适配器之间建立通讯)。

MODBUS MODE
(Modbus 模式)

硬件通讯的逻辑协议(NMBA-01 仅支持 RTU 模式)以及 Watch-Dog 复位方式，有两个选项：

RTU wdg:flt
远程终端模式 (RTU)，当 Watch-Dog 溢出时，适配器指示错误，需要人工复位。

RTU wdg:rst
远程终端模式 (RTU)，当 Watch-Dog 溢出时，自动复位。

STATION NUMBER
(站号)

Modbus 通讯链路上的每台设备都有一个唯一的站号，这个参数用于给传动装置定义站号。

BAUD RATE
(波特率)

定义通讯速率，有 5 个选项：**1200**，**2400**，**4800**，**9600**，**19200** 波特。

PARITY
(奇偶校验)

定义 Modbus 通讯的奇偶校验方式，同时也定义停止位数。典型的 Modbus 通讯选用 2 个停止位 + 无校验位或 1 个停止位 + 奇或偶校验。

有四种选择：
EVEN, ODD, NONE 1 S.BIT 秸 **NONE 2 S.BIT.**

GOOD MESSAGES
（好消息）

这是一个诊断计数器，NMBA-01 每接到一个有效的消息，则这个计数器加 1，当计满 32767 时，它溢出回零。正常操作时，这个计数器值不断增加。

BAD MESSAGES
（坏消息）

这是一个诊断计数器，NMBA-01 每出现一个通讯错误，则这个计数器加 1，当计满 32767 时，它溢出回零。正常操作时，这个计数器的值很少增加。

DDCS CHANNEL
(DDCS 通道）

选择传动装置所用的 DDCS 通道。这个参数用于 ACS600。

---

*注意：*改变这个参数不会立即升效，若要改变 DDCS 通道参数设置，请断开 NMBA-01 的电源。

控制区

ABB 的传动产品都具有从多通道接收控制信号的功能。包括数字输入、模拟输入、控制盘以及通讯模块。ABB 的传动产品还可以让用户为每一种控制信号(起、停、方向、给定、故障复位等)单独选择信号源。为使传动完全由现场总线来控制，必须把通讯模块作为控制信号源。参见传动手册。

第六章 - 通讯

概述

本章介绍传动装置的 Modbus 通讯

寄存器读／写

传动参数与数据集被定位于 4XXXX 的寄存器区。这一寄存器区可以由外部设备通过 Modbus 来读写。

将数据定位于 4XXXX 寄存器区不需任何参数，它是预先定义的，与传动参数一一对应。

所有参数都可以读写。参数的写操作有数据校验和地址校验，但是有些参数永远不允许写入（实际值），有些参数仅在传动停机时才可写入，有些参数在任何时间都可写入（给定值）。

寄存器地址分配

传动参数位于 4XXXX 区域：

* 40001-40096 用于数据集
* 40101-49999 用于参数

上述地址中，千位和百位对应于参数组号，十位和个位对应于组内参数号。寄存器地址 4GGPP 示于*表 6-1 参数地址*中。表中，GG 是组号。PP 是组内的地址号。

表 6–1　参数地址

	4GGPP	GG	PP
Data sets	4**0001** – 4**0096**	**00** 数据集	**01** 数据字 1.1 **02** 数据字 1.2 **03** 数据字 1.3 **04** 数据字 2.1 **05** 数据字 2.2 **06** 数据字 2.3 **07** 数据字 3.1 ... **94** 数据字 32.1 **95** 数据字 32.2 **96** 数据字 32.3
参数	4**0101** – 4**0199**	**01** 组 01	**01** 参数 01 **02** 参数 02 ... **99** 参数 99
	4**0201** – 4**0299**	**02** 组 02	**01** 参数 01 **02** 参数 02 ... **99** 参数 99
	...	...	...
	4**9901** – 4**9999**	**99** 组 99	**01** 参数 01 ... **99** 参数 99

没有与任何传动参数或数据集对应的寄存器地址为无效地址。如果试图在参数地址之外的区域进行读写，则主控制器将得到一个异常代码。对于传动所支持的数据集，参数组或参数号，请参考有关传动手册。

异常代码

NMBA-01 所支持的 Modbus 异常代码见*表 6-2* 所示。

表 6-2 异常代码。

代码	名称	原因
01	ILLEGAL FUNCTION	不支持的命令。
02	ILLEGAL DATA ADDRESS	地址不存在或读 / 写保护。
03	ILLEGAL DATA VALUE	数据值超限，参数只读。
04	SLAVE DEVICE FAILURE	读多个参数寄存器时，含有一个或多个读保护。
06	SLAVE DEVICE BUSY	DDCS 通道超时。

数据更新

NMBA-01 模块的设计，保证了 Modbus 网络和传动装置之间快速、可靠的通讯。

表 6-3 功能代码

代码	名称	含义
03	读保持寄存器	读保持寄存器 (4X) 中的 2 进制值
06	预置单寄存器	对某一个寄存器 (4X) 预设一个值，广播操作时，这个命令对所有从机的同一寄存器预置相同的值。
16 (10 Hex)	预置多寄存器	对一组寄存器 (4X) 预设一个值，广播操作时，这个命令对所有从机的同一组寄存器预置相同的值。

数据集寄存器的数据更新周期很快，参数寄存器的刷新周期较慢。当进行单寄存器读写时，参数值是从传动装置的存储器中直接刷新的。当进行多寄存器读写时，参数值是在 NMBA-01 的存储器中被刷新的。

多传动控制

一块 NMBA-01 可连接多台传动装置（理论上 247），多传动的控制原理与点对点控制原理相同，NMBA-01 在 Modbus 网络上就象是一个多地址节点。

Modbus 网络可以有多个多传动控制段，但总的传动数量不能超过 247 个。

---

*注意：*一个多传动段上所连接的传动装置只能是一种类型（例如 ACS300 或 ACS600）。

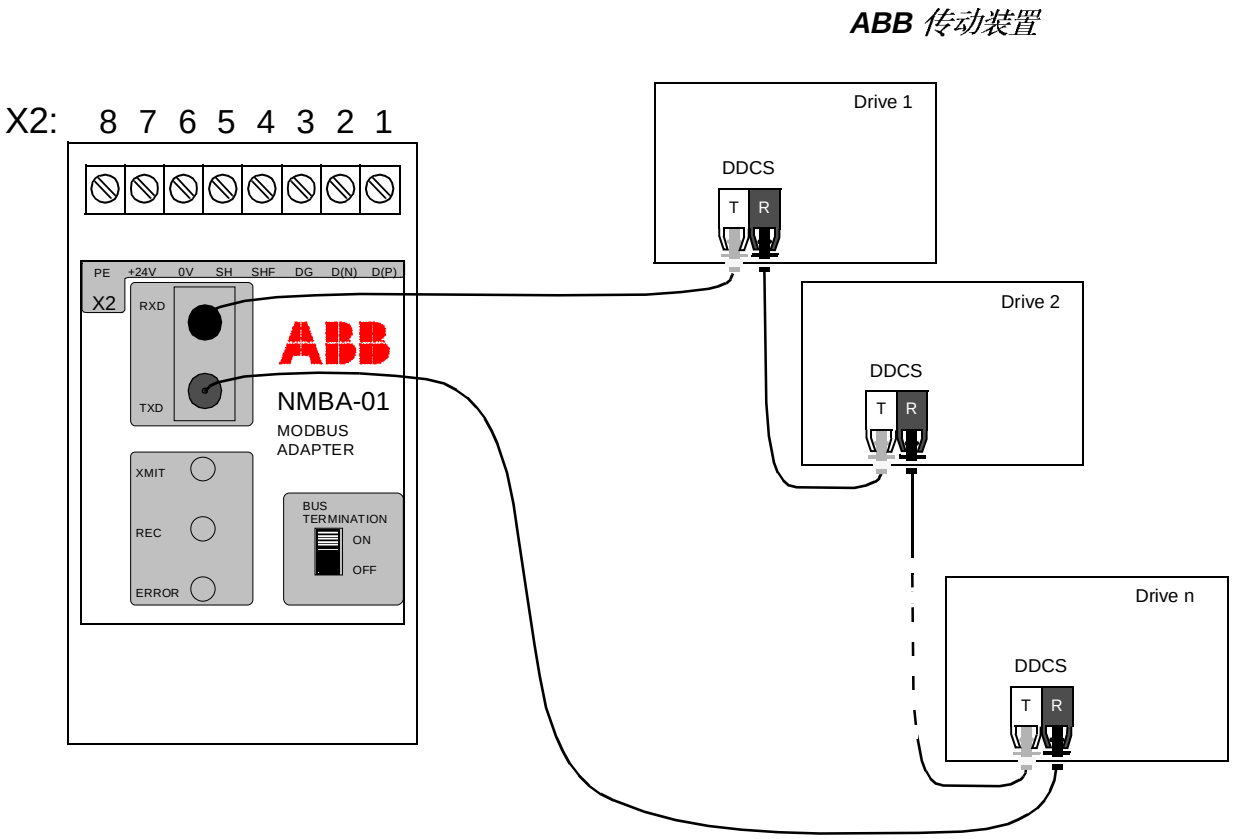

图 6-1　　多传动控制连接方法

多传动控制的通讯连接方式并不显著地增加 DDCS 通讯的延迟，但由于每个参数值都要从传动装置内部存储器来刷新，所以参数寄存器中的数据刷新周期还是比点对点通讯时要长些。当进行多参数寄存器读写时，传输延迟将显著增加。

多传动控制设置

多传动情况下，仅需要设置传动通讯环路中的第一台传动装置的参数。先将 NMBA-01 模块与第一台传动装置进行点对点相连(图 4-2)，用控制盘设置第一台传动装置的现场总线参数 (详见第 5 章)。配置完 Modbus 适配器参数后，再接成多传动控制方式。

NMBA-01 在初始化期间，检查所连接环路上的传动装置台数，环路上所有从机站点被自动编号。例如，如果第一台传动装置的参数 03 STATION NUMBER 是 5 ，则与其相邻的第二台传动的站号就是 6，依此类推。在多传动控制方式中，NMBA-01 仅向其它传动装置发送两个参数，那就是 *01 MODULE TYPE* 和 *03 STATION NUMBER*，它们是只读的。

参数的操作

在一个参数组内，NBRA-01 在一帧内可以读写多达 99 个参数，用户可以读写多传动通讯链中的每一台传动装置的所有参数，就好象每一台传动装置都有自己的 NMBA-01 模块。但是，响应时间可能会长些，取决于所处理的参数数量。

广播命令不能用于修改参数。

数据集通讯

数据集 1、3、5……(从 NMBA-01 到传动)可用于广播消息，但数据集 2、4、6……(从传动到 NMBA-01)不能用于广播消息。

第七章 - 故障跟踪

概述

本章对 NMBA-01 模块最常见问题的发生原因及解决办法，分步列出了诊断信息。

本章分为几节，分别列出了故障现象、可能的原因以及解决办法。

安装问题

检查模块上所有的接线，确保：

* Modbus 电缆正确地接到 X2 端子。
* 24V.D.C 电源接到电源连接器。
* 传动装置与 NMBA-01 之间的光缆已正确连接。
* 光缆连接器的颜色分别与传动装置和 NMBA-01 相吻合。

传动参数设置

从控制盘上不能看到现场总线参数组。

* 根据传动手册，激活现场总线适配器

传动参数可读，但不接受控制命令（起 / 停 / 给定）。

* 确认这些控制命令的信号源是否是现场送线适配器。

PLC 编程

PLC 程序不属于 ABB 的范围，请与 PLC 供应商联系。

状态指示 LED

NMBA-01 模块有 3 个状态指示 LED，从上到下：

* *XMIT* LED, NMBA-01 每向 Modbus 发送一个响应或例外信号，这个 LED 闪一次。
* *REC* LED, NMBA-01 每从 Modbus 接收一个命令，这个 LED 闪一次。
* *ERROR* LED 用于综合故障指示及模块状态指示，下列原因将引起这个 LED 闪亮：
  - 如果接收的命令有奇偶校验错误。
  - 如果接收的命令有 CRC 错误。
  - 如果接收的命令 NMBA-01 不支持，这种情况下 *XMIT* LED 也闪亮。
  - 如果 NMBA-01 模块有错误，错误代码如*表 7-2* 所示。

模块诊断

上电时，NMBA-01 进行上电自检，在自检过程中，面板上的 3 个 LED 指示自检状态，一般情况下上电过程为：

* ROM/RAM 测试时，所有 LED 全亮，测试过后，所有 LED 全灭。
* 在 NMBA-01 与传动装置之间的 DDCS 链路初始化时，*ERROR* LED 将快速闪几秒。
* 根据 NMBA-01 与传动装置之间的数据传输状态，*REC*和 *XMIT* LED 将闪几次。

表 7-1 初始化期间的 Modbus 错误码。

LED 代码	状态	排除方法
XMIT, REC, 及 ERROR LED 全亮	ROM 校验和测试失败	硬件故障，更换 NMBA
REC 及 ERROR LED 亮	RAM 测试失败	硬件故障，更换 NMBA
ERROR LED 亮	DDCS ASIC 寄存器存取测试失败	硬件故障，更换 NMBA

表 7-2 操作期间的错误代码

LED 代码	状态	排除方法
Error LED 偶尔闪一次。	模块正常，奇偶校验或 CRC 校验错，或收到不支持命令。	检查模块接线和接地， 确认仅向模块发送有效的命令
Error LED 连续闪。	NMBA 良好，传动无响应。	接通传动装置电源， 检查光缆连接。
所有 LED 同时连续闪。	Watchdog 溢出。	NMBA 模块的硬件故障，关掉模块电源，等待几秒再打开模块电源，如果问题仍然存在，检查传动的现场总线参数 02 MODBUS MODE 应设为 RTU。
不断重起动。 *ERROR* LED 快速闪烁约两秒，每过几秒重复一次。	传动配置参数写入失败（DDCS 链路通讯故障）。	检查光缆连接，DDCS 通道和传动装置的软件版本。

DDCS 链路

兼容设备： ABB 公司所有型号现场总线适配器模块，ACS 300，ACS400，ACS/ACP/ACF 600，DCS500 传动装置。

最大链接数量： 2 个站点（在多传动控制方式为 3 到 248）

传输介质： 光缆

* 结构：塑料芯，直径 1 mm，塑料外皮
* 衰减：0.31 dB/m
* 最大长度：两个站点之间 10 m
* 光缆技术指标：

参数	最小	最大	单位
存贮温度	-55	+85	°C
安装温度	-20	+70	°C
长期抗拉强度		50	N
短期弯曲半径	25		mm
长期弯曲半径	35		mm
长期拉伸负荷		1	N
弯曲		1000	次

拓扑结构： 点对点（多传动方式为环形）

串行通讯类型： 异步，半双工

传输速率： 4Mbit/s

协议： 分布式传动通讯系统 (DDCS)

连接器： 蓝色 = 接收器；灰色 = 发射器

Fieldbus 通讯链路

兼容设备： 所有可作为 Modbus 主机的 Modbus 设备

连接数量： 247 个站点包括中继器（每段 31 个站点和一个中继器）

介质： 屏蔽双绞线 RS485 电缆

* 终端器： NMBA-01 模块内置
* Modbus 电缆 :Belden 9841(典型)
* 总线最大长度 :1200m

拓扑结构： 多点

串行通讯类型： 异步，半双工

传输率： 1200,2400,4800,9600,19200 bit/s

协议： Modbus

NMBA-01

外壳： 塑料， 外形尺寸 45 x 75 x 105 mm; 防护等级 IP 20

安装： 在标准导轨上

设置： 通过传动接口（控制盘）

电流消耗： 65 mA ，24 V d.c.

连接器：

* 光发射器（灰色）和接收器（蓝色）(Hewlett-Packard 通用连接器）与传动相连。
* 一块 MVSTBW 2,5/8-ST-5,08 (8 极， 最大截面积 2.5 mm^2 ）端子用于现场总线和模块电源的连接：

X2		说明
1	D(P)	D(P) = B = 数据正 (双绞线的 1 号导体) D(N) = A = 数据负 (双绞线的 2 号导体) DG = 数据地
2	D(N)	
3	DG	
4	SHF	电缆屏蔽层 AC 接地 (通过 RC 滤波器)
5	SH	电缆屏蔽层接地 (直接接地)
6	0V	模块电源 (24 V d.c. ± 10 %); 用屏蔽电缆。
7	+24 V	
8	PE	地

一般性能：

* 所有材料均通过 UL/CSA 认可
* 与 EMC 标准 EN 50081-2 和 EN 50082-2 兼容

附录 B–Modbus 协议

本章介绍串行的 Modbus 协议，是为那些需要为用户主机编程的人员准备的。

本章的内容由 Modicon 拥有版权，并得到 AEG Schneider Automation (Modicon) 许可使用，这些内容摘自 Modicon 出版物— Modicon Modbus 协议参考手册 (P1—MBUS—300 版本 E)。其最新版本可以从 Modicon 主页：*http:// www.modicon.com* 得到。

Modbus 协议简介

Modbus 协议是一种串行主 / 从协议。本附录所介绍的 Modbus 协议足够全面地对 ABB 传动装置进行控制。Modbus 协议定义了通讯链路上串行传输的内容。NMBA-01 的物理接口是半双工 RS-485。

Modbus 网络上信息交换

Modicon 控制器的标准 Modbus 接口使用 RS-232C 兼容的串行接口，它定义了连接器的插针、电缆、信号电平、传输速率以及校验方式。控制器可直接入网或通过调制解调器入网。

控制器之间使用主 / 从技术通讯，这里仅有一台设备(主机)可以请求数据交换（叫作“查询”），其它设备(从机)以向主机提供数据或执行主机的查询命令的方式对主机作出响应。典型的主机类型为上位控制器或编程控制盘。典型的从机为可编程控制器。

主机可以与某一个从机通讯，也可以对所有从机发送广播消息，从机单独地向主机回送消息（叫作“响应”），广播消息不需要响应。

Modbus 协议定义了查询消息的格式和从机响应消息的格式。主机查询消息包括地址、功能代码、数据以及错误校验。从机响应消息包括地址、功能代码、数据及错误校验。如果从机在接收消息时出错或是从机不能执行所接收的命令，它将生成一个错误消息并作为响应发送给主机。

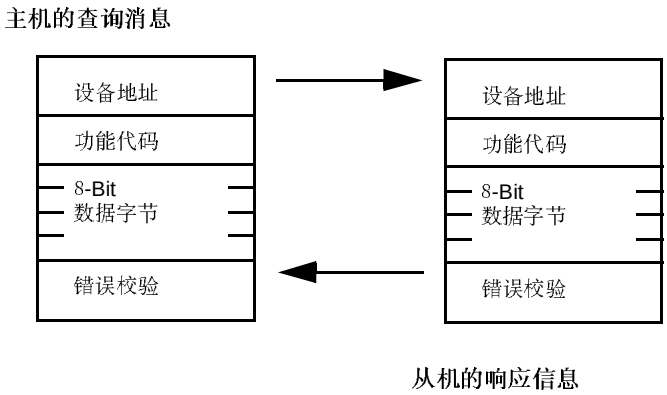

图 B-1 主／从 查询—响应

查询： 查询消息中的功能代码告诉从机做什么。数据是从机执行命令所需要的附加信息。例如，功能代码 03 表示让从机读保持寄存器，并把所读内容返回主机。功能代码之后的数据告诉从机：从哪一个地址开始读，读多少个寄存器。错误校验为从机提供了验证信息正确性的方法。

响应： 如果从机做出常规响应，则响应消息中的功能代码就是查询消息中的功能代码。数据域包含了从机收集的信息，诸如寄存器值、状态等。如果发生错误，则功能代码被修正以指示这是一个错误响应。数据域包含了对错误的描述代码。错误校验使主机可以确认信息的有效性。

两种串行传输模式

Modbus 网络允许为控制器设置两种不同的传输模式，ASCII 模式或 RTU 模式。用户可以选择希望的传输模式以及串口通讯参数（波特率，校验模式等）。**Modbus 网络上的所有设备的传输模式和串口参数必须一致。**选项 ASCII 模式或 RTU 模式只属于标准 Modbus 网络。它定义了网络上传输的信息流的比特内容。它决定了信息如何被变成信息流并如何还原。

NMBA-01 只支持 RTU 模式，所以这里只介绍 RTU 模式。

RTU 模式

当把控制器设为 RTU 模式时，每一个 8 位字节用 2 个 4 位 16 进制字符表示。这种模式的主要优点是，在相同的波特率下，能提供比 ASCII 模式更大的数据吞吐量。每一贞消息必须连续传输。

RTU 模式中每字节的格式为：

码制：	8 位二进制，16 进制 0-9，A-F，每一个 8 位字节用两个 16 进制字符表示。
每字节的位数：	一个起始位
	8 个数据位，低位在前
	1 个校验位（ 奇，偶 ）
	1 个停止位（ 有校验位时 ）或 2 个停止位（ 无校验位时 ）
错误校验：	循环冗余校验（ CRC ）

Modbus 消息页

无论采用何种传输模式 (RTU 或 ASCII)，传输设备都将 Modbus 信息构成消息页，包含头和尾。这就允许接收设备在一开始就读出地址部分，以便确认哪台设备被询址（如果是广播，所有设备被询址），也便于确认消息何时终止。参损的消息可以被探测出并设置错误指示。

RTU 页

在 RTU 模式，消息以至少 3.5 个字符间隔的空闲开始，这是最容易实现的。在网络使用的波特率下，它是一个字符传输时间的整倍数，紧随其后的第一个域是设备地址。

在消息页中，所有数据域只许以 16 进制字符表示。连网设备连续监视网络总线，包括对空闲间隔的监视。当接收到第一个域(地址域)时，所有设备都对地址进行译码，检查是否本设备被询址。整个消息页必须连续传输，如果在一页消息结束前，出现大于 3.5 个字符的时间间隔，则设备将不完整的消息溢出并认为下一个字节是一个新消息页的地址域。

同样地，如果一个新消息的开始紧随前一个消息，其时间空闲小于 3.5 个字符，则设备将把它看作是前一个消息的继续。这将导致错误，因为最后的 CRC 域与所收的内容不一致。典型的消息页表示如下：

开始	地址	功能	数据	CRC 校验	结束
T1-T2-T3-T4	8 BITS	8 BITS	n * 8 BITS	16 BITS	T1-T2-T3-T4

图 B-2 消息页

如何处理地址域

一贞消息的地址是 8 位 (RTU)。从机地址的有效范围是 0-247，每个从机地址被分配在 1-247 的范围，主机依靠将从机地址放在消息贞的地址域的方式对从机询址。从机响应时，它将自己的地址放在响应消息贞的地址域，让主机知道，是哪一台从机在响应。

地址 0 用于广播，所有从机都能识别。当 Modbus 协议用于上位机网络时，可能不能使用广播，或用其它方式代替广播。例如，Modbus Plus 使用了共享全域数据库，以令牌轮回方式，每台设备都能对数据库进行更新。

如何处理功能域

一贞消息的功能码是 8 位 (RTU)，有效代码范围是 1-255。其中，有些代码用于所有 Modicon 控制器，有些代码只适用于某些 Modicon 控制器，还有些保留为将来使用。NMBA-01 仅支持功能代码 3，6 和 16。

当消息贞由主机发送到从机时，功能代码用于告诉从机所要执行的功能。例如读一组输出。当从机响应主机时，功能代码用于指示响应是正常的，还是发生错误了。对于一个正常的响应，从机只是将其所收的功能代码原封不动地发送回去。对于一个出错的响应，从机把其所收的功能代码的最高位置 1，然后发送回去。

例如，主机发给从机的信息是让从机读一组保持寄存器，其功能代码为：

0000 0011 (03H)

如果从机执行无误，则以原功能码作为其响应。如果有错误发生，则从机将回送：

1000 0011 (83H)

除了修改其响应的功能代码，从机还将一个特殊的代码放入响应的数据域，用于告诉主机发生了什么错误，或发生错误的原因。

主机的应用程序应该对错误响应作出处理。一般的处理是重发查询消息或发送诊断消息或通知操作员。

数据域内容 数据域内容由一组 16 进制数组成，16 进制数的范围是 00H-FFH。

从主机到从机的数据域包含从机执行主机命令所需要的额外信息。它可能包括诸如寄存器地址、数量、数据字节计数值等内容。

例如，如果主机要求从机读一组保持寄存器的内容（功能码为 03），则数据域定义了寄存器的起始地址和寄存器数量。如果主机要向从机的寄存器写一组数，则数据域定义了起始地址、寄存器数量、数据域字节计数值以及要写入寄存器的数据。

如果无任何错误，则从机响应的数据域包含了主机需要的数据。如果有错误发生，则数据域包含了一个错误代码。主机的应用程序可根据代码内容决定下一步的措施。

在某些信息中，可以不存在数据域。例如，当主机要求从机回送其通讯记录时（功能码 0BH），从机不需要任何其它信息，功能代码已定义了一切。NMBA-01 不支持功能码 0BH。

错误校验域内容 在标准的 Modbus 网络中，可使用两种校验方法，校验域的内容取决于校验方法。

ASCII 当使用 ASCII 字符贞时，错误校验域包含两个 ASCII 字符。它是对消息内容采取纵向冗余校验 (LRC) 计算的结果，但是不包括起始符‘冒号’和结束符 CRLF（回车换行）。

LRC 字符作为最后一个域被附加于消息中，但是在 CRLF 之前。

RTU

如果消息贞使用 RTU 模式，则错误校验域包含 16 位二进制，由两个 8 位字节组成。校验值是消息贞数据进行 CRC 运算的结果。

CRC 域附加在信息的最后，是信息贞的最后一个域，低位字节在前，高位字节在后。CRC 的高 8 位是消息贞的最后一个字节。

有关错误校验的更多内容在本附录的后几页。

字符如何串行传输

当消息在标准 Modbus 网络串行传输时，每个字节以下述方式传输（从左到右）:

最低有效位 (LSB) ... 最高有效位 (MSB)

RTU 字符贞的位顺序是：

带奇偶校验

起始位	1	2	3	4	5	6	7	8	校验位	停止位

不带奇偶校验

起始位	1	2	3	4	5	6	7	8	停止位	停止位

错误校验方法

标准的 Modbus 串行网络使用两种校验，位校验(奇 / 偶)和贞校验。位校验可用于每个字节的校验，贞校验用于消息贞的校验。在传输前，主机将两种校验生成并附加于消息中。从机在收到消息时对每一个字节和整个消息贞进行检查。

用户可以配置主机，让主机在取消信息交换前等待一个预定的时间。这个时间应足够长，使每一个从机有足够的时间做出正常响应。如果从机发现错误，它就不会执行命令，也不会作出响应，因而出现超时错误，使主机程序处理这一错误。注意，如果对一个不存在的从机地址询址，也会出现超时错误。对 NMBA-01，超时时间可设为 100ms。

位校验 用户可以将控制器配置为偶校验或奇校验或无校验。如果配置了校验，则每一个字节的 1 的个数会被计数，然后将校验位设置为 0 或 1，使传输位的个数为偶数或奇数。

例如，下面的字节：

1100 0101

包含 1 的个数为 4，如果使用偶校验，则校验位应设为零，使 1 的个数保持偶数 (4)。如果使用奇校验，则校验位应设为 1，使 1 的个数为奇数 (5)。

当传送消息时，发送设备计算校验位并把它加在字符贞的最后。接收设备在接收时，对 1 的个数进行计数，当发现与配置的校验方式不符时，认为出错。Modbus 上的所有设备应使用相同的校验方式。

注意，位校验只能检查奇数位错误。例如，如果使用了奇校验，当一个含有 3 个 1 的字节丢失两个 1 时，其结果是，字节中 1 的个数仍然是奇数。

如果不使用位校验，则不传送校验位，也不需进行位校验，用停止位填充传输的字节空位。

CRC 校验 在 RTU 模式，消息贞含有一个校验域，是基于循环冗余校验(CRC)方法生成的。CRC 校验域对整个消息贞的内容进行校验，它的应用与位校验无关。

CRC 域有两个字节，含 16 位二进制数。CRC 值由发送设备计算生成，附加于消息之后。接收设备在接收信息过程中，对消息的 CRC 进行计算，并将计算值与它接收的 CRC 值进行比较，如果两个值不相等，则产生错误。

CRC 的计算从预装 FFH 到一个 16 位寄存器开始，然后将消息的 8 位字节顺序与寄存器中的值运算，仅使用字符贞中的 8 位参与 CRC 运算，起始位、停止位和校验位不参与 CRC 运算。

在 CRC 生成过程中，每一个 8 位字节与 CRC 寄存器的内容进行异或操作，将结果向低位(LSB)移位，高位(MSB)用 0 填充，对移出的 LSB 位进行检查，如果是 1，则寄存器与一个固定的值进行异或操作。如果是 0，则不进行异或。

这一过程重复直到将 8 位全部移出。在最后一次移位之后，下一个字节与 CRC 寄存器进行异或操作，重复上述操作。消息中的所有字节都运算完后，CRC 寄存器中的值就是 CRC 值。在梯形图逻辑中，CKSM 函数就是计算 CRC 值的。对于使用主计算机的场合，生成 CRC 的详细实例示于本附录之后。

Modbus 功能码格式

本节对 NMBA-01 支持的每一个 Modbus 消息贞中的数据进行详解。

数值表达方式

除非另有定义，数值(如地址、代码、数据)*在本节中以十进制方式表示。在消息贞中它们以 16 进制方式表示。*

Modbus 消息中的数据地址

Modbus 消息中的所有地址均以 0 为基准。第一个出现的数据项目，其项目号为 0，例如：

* 在可编程控制器中的“线圈 1”在 Modbus 信息中的数据地址是 0000
* 线圈 127 的地址是 007EH
* 在消息的地址域中，保持寄存器 40001 的数据地址是 0000，功能代码域已定义了对保持寄存器的操作，因此 4XXXX 的基准是隐含的。
* 保持寄存器 40108 的地址是 006BH

Modbus 消息中的域内容

图 B-3 *RTU 贞构成的主机查询*列举了 Modbus 查询消息的例子。图 B-4 *RTU 贞构成的从机响应*是一个常规响应的例子。两个例子说明了以 16 进制数表示的域内容以及如何构成一贞消息。

主机查询要求地址为 06 的从机读三个保持寄存器，寄存器地址为 40108-40110。消息定义了寄存器的起始地址为 0107(006BH)。

从机响应回送了功能代码，表示是一个常规响应。“字节计数”域定义了返回的字节个数。它表示了后续的字节个数。

例如，值 63H 作为 8 位数字节，以 RTU 模式 (01100011) 发送，“字节计数”域记录这个数据，“字节计数”等于 1，与字符贞方式无关 (ASCII/RTU)。

如何使用字节计数：当在缓冲区中组建一个响应时，使用等于消息中字节数的字节计数值，这个值不包含其它域的内容。图 B-4“RTU 贞构成的从机响应”说明了如何构成字节计数域。

查询		
域名	实例 (Hex)	RTU 8-Bit 域
头		无
从机地址	06	0000 0110
功能	03	0000 0011
起始地址高位	00	0000 0000
起始地址低位	6B	0110 1011
寄存器数量高位	00	0000 0000
寄存器数量低位	03	0000 0011
错误校验		CRC (16 bits)
尾		无
	总字节数：	8

图 B–3 RTU 页构成的主机查询

响应		
域名	实例 (Hex)	RTU 8-Bit 域
头		无
从机地址	06	0000 0110
功能	03	0000 0011
字节计数	06	0000 0110
数据高位	02	0000 0010
数据低位	2B	0010 1011
数据高位	00	0000 0000
数据低位	00	0000 0000
数据高位	00	0000 0000
数据低位	00	0000 0000
错误校验		CRC (16 bits)
尾		无
	总字节数 :	11

图 B–4 RTU 页构成的从机响应

功能码

NMBA-01 支持 3 个 Modbus 功能代码，允许主机对传动装置进行 16 位整数读写操作。

03 读保持寄存器

读保持寄存器(4XXXX)中的值，本命令不支持广播。

查询

查询消息定义了要读的保持寄存器的起始地址和寄存器数量，寄存器地址从 0 开始，寄存器 1-16 的地址为 0-15。

下面是请求读寄存器 40108-40110 的例子，从机地址为 17：

QUERY	
域名	实例 (Hex)
从机地址	11
功能	03
起始地址高位	00
起始地址低位	6B
寄存器数量高位	00
寄存器数量低位	03
错误校验 CRC	CRC (16-Bits)

图 B-5 读保持寄存器—查询

响应

在响应消息中，寄存器数据被分解为两字节，每个字节中的二进制数向右对齐。对每个寄存器而言，高位字节在前，低位字节在后。

对于 984—X8X 可编程控制器而言，数据的扫描速率是每秒 125 个寄存器，其它的可编程控制器中，数据扫描速率是每秒 32 个寄存器。

下面是对前一个查询的响应消息：

响应

域名	实例 (Hex)
从机地址	11
功能	03
字节计数	06
数据高位 (寄存器 40108)	02
数据低位 (寄存器 40108)	2B
数据高位 (寄存器 40109)	00
数据低位 (寄存器 40109)	00
数据高位 (寄存器 40110)	00
数据低位 (寄存器 40110)	64
错误校验 CRC	CRC (16-Bits)

图 B-6 读保持寄存器—响应

寄存器 40108 中的内容用 2 个字节表示就是 02 2BH 或十进制的 555，寄存器 40109-40110 中的内容分别是 00 00H 和 0064H，用十进制表示就是 0 和 100。

06 预置单寄存器 给某一个寄存器预置一个值。广播时，这一功能对所有从机的同一寄存器预置相同的值。

查询 查询消息定义了预置寄存器的地址基准，寄存器地址从零开始，寄存器 1 被分配在地址 0。

预置的数值由数据域定义，NMBA-01 使用 16 位值。

下面是将 17 号从机的 40002 寄存器预置为 0003H 的例子：

查询	
域名	实例 (Hex)
从机地址	11
功能	06
寄存器地址高位	00
寄存器地址低位	01
预置的数据高位	00
预置的数据低位	03
错误校验 CRC	CRC (16-Bits)

图 B-7 预置单寄存器—查询

响应 预置完数据后，从机作出正常响应。

下面是一个正常响应的例子：

响应	
Field Name	实例 (Hex)
从机地址	11
功能	06
寄存器地址高位	00
寄存器地址低位	01
预置数据高位	00
预置数据低位	03
错误校验 CRC	CRC (16-Bits)

图 B-8 预置单寄存器—响应

16 (10 Hex) 预置多寄存器 对一组保持寄存器预置数据。广播时，这一功能对所有从机的相同寄存器预置相同的数据。

NMBA-01 允许使用 16 号命令一次对一个或多个寄存器进行预置数。一次写命令中，只能对一个组中的寄存器进行预置。如果对某一个寄存器的写入操作失败，模块(NMBA-01)仍然对其它寄存器写入，但响应消息中将包含有关的错误信息。

查询 查询信息定义了预置寄存器的基准地址。寄存器从地址 0 开始，寄存器 1 被分配在地址 0，预置的数值由数据域定义。NMBA-01 使用 16 位值，每个寄存器被分解为两字节。

下面是将 17 号从机的 40002 寄存器预置为 000AH 的例子：

查询	
域名	实例 (Hex)
从机地址	11
功能	10
起始地址高位	00
起始地址低位	01
寄存器数量高位	00
寄存器数量低位	01
字节计数	02
数据高位	00
数据低位	0A
错误校验 CRC	CRC (16-Bits)

图 B-9 预置多寄存器—查询

响应

正常的响应把从机地址、功能代码、开始地址以及寄存器数量返回主机。

下面是一个响应的例子，与上面的查询对应。

响应	
域名	实例 (Hex)
从机地址	11
功能	10
起始地址高位	00
起始地址低位	01
寄存器数量高位	00
寄存器数量低位	01
错误校验 CRC	CRC (16-Bits)

图 B-10 预置多寄存器—响应

异常响应

除广播消息外，当主机向从机发出查询命令后，主机总是等待从机的响应，从机可能作出下列四种响应之一：

1. 如果从机接收无误，并能处理查询命令，它将作出正常响应。
2. 如果从机由于通讯错误未能收到查询，它不作任何响应，主机的程序最终将作出超时处理。
3. 如果从机收到查询，但有通讯错误，它不作任何响应，主机最终作出超时处理。
4. 如果从机收到查询，但不能执行（例如读一个不存在的寄存器），从机将作出一个异常响应，通知主机错误的类型，异常响应有两个域与正常响应不同。

功能代码域：正常响应时，从机将原来的功能代码送回给主机。所有功能代码的最高位为 0。在异常响应中，从机将原来的代码最高有效位设置为 1，这将使得异常响应的功能代码比正常响应正好大 80H。依靠将功能代码的最高有效位设置为 1，主机可以识别出异常响应，并进一步从数据域查出错误类型。

数据域：正常响应时，从机可能在数据域放置数据（或任何主机需要的信息）。异常响应时，从机在数据域放置一个错误代码，它定义了导致异常的原因。

图 B-11“主机查询与从机的异常响应”示出了一个例子。

所有域的内容以 16 进制数表示。

查询		
字节	内容	例
1	从机地址	0A
2	功能	01
3	起始地址 高位	04
4	起始地址 低位	A1
5	线圈数量 高位	00
6	线圈数量 低位	01
7	LRC	4F
异常响应		
1	从机地址	0A
2	功能	81
3	异常代码	02
4	LRC	73

图 B-11 主机查询与从机的异常响应

本例中，主机向 10 号从机发送查询命令，功能代码(01)是读线圈状态，线圈地址是 1245(04A1H)。注意，只读一个线圈状态，线圈数量由数据域定义。

如果线圈地址不存在，从机作出异常响应，异常代码为 02，它表示这是从机的非法地址。

例如，如果从机是 984-385，带有 512 个线圈，则做出上述异常响应。(1245 地址不存在)

*表 B-1 标准异常代码*列出了 Modicon 的异常代码。

表 B-1 标准异常代码

代码	名称	含义
01	ILLEGAL FUNCTION	从机接收的功能代码不可执行，如果收到“程序执行查询”命令，则表示从机没有可执行程序
02	ILLEGAL DATA ADDRESS	从机不存在这个地址
03	ILLEGAL DATA VALUE	数据域中的数据不可用
04	SLAVE DEVICE FAILURE	从机执行命令时出现不可恢复的故障
05	ACKNOWLEDGE	从机接受命令并执行它，但需要较长的时间来执行，这个响应用于使主机不产生超时错误，主机下一步可以发送“程序执行查询”命令，以便确认从机是否已完成任务。
06	SLAVE DEVICE BUSY	从机正在执行较长的命令，主机应该稍后再发命令。
07	NEGATIVE ACKNOWLEDGE	从机不能执行所收到的命令，是因为没有成功地执行 13 或 14 号命令，主机应向从机发送诊断或错误信息请求。
08	MEMORY PARITY ERROR	从机试图读扩展内存，但出现位校验错误

CRC 生成

CRC 域有两个字节，含 16 位二进制数。CRC 值由发送设备计算生成，附加于消息之后。接收设备在接收消息过程中对信息码的 CRC 进行计算，并将计算值与它接收的 CRC 值进行比较。如果两个值不相等，则产生错误。

CRC 的计算从预装 FFH 到一个 16 位寄存器开始，然后将信息的 8 位字节顺序与寄存器中的值运算。仅使用字符贞中的 8 位参与 CRC 运算，起始位、停止位和校验位不参与 CRC 运算。

在 CRC 生成过程中，每一个 8 位字节与 CRC 寄存器的内容进行异或操作，将结果向低位(LSB)移位，高位(MSB)用 0 填充。对移出的 LSB 位进行检查，如果是 1，则寄存器与一个固定的值进行异或操作。如果是 0，则不进行异或。

这一过程重复，直到将 8 位全部移出。在最后一次移位之后，下一个字节与 CRC 寄存器进行异或操作。重复上述操作，所有消息贞中的字节都运算完之后，CRC 寄存器中的值就是 CRC 值。

生成 CRC 的过程为：

1. 用 FFFFH 装载 16 位的 CRC 寄存器。
2. 将 CRC 寄存器与消息贞中的第一个字节进行异或操作，结果存于 CRC 寄存器中。
3. 将 CRC 寄存器向低位(LSB)方向移位一次，高位(MSB)用 0 填充。检查移出的位。
4. 如果是 0，则重复第 3 步。如果是 1，则将 CRC 寄存器与 A001H 进行异或操作，结果存于 CRC 寄存器。
5. 重复第 3、4 步 8 次，处理完一个字节。
6. 重复第 2-5 步，直到处理完消息贞中的所有字节。
7. CRC 寄存器中的最后值就是 CRC 值。

将 CRC 放置于消息贞中

当传送消息贞中的 CRC 时，低位字节在前，高位字节在后。例如，如果 CRC 值是 1241H(0001 0010 0100 0001)，则传送顺如下：

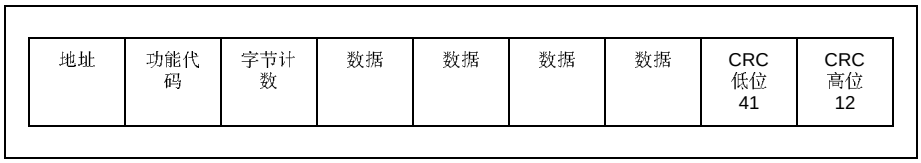

图 B-12 CRC 字节顺序

实例

下页列举了一个执行 CRC 生成运算的 C 语言函数实例。CRC 的全部有效值预装在两个数组内。一个数组内含 CRC 的全部低位有效字节，一个数组包含 CRC 的全部高位有效字节。用索引的方法比用每一个字节计算 CRC 的方法处理速度快得多。

函数使用两个变量：

unsigned char *puchMsg
生成 CRC 的数据缓冲区地址指针。

unsigned short usDataLen
缓冲区中数据的数量 。

CRC 函数以短整数形式返回 CRC 值。

```
/* Table of CRC values for high-order byte */
static unsigned char auchCRCHi [ ] = {
0x00,0xC1,0x81,0x40,0x0l,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,0x00,0xCl,0x81,
0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,0x40,0x0l,0xC0,
0x80,0x41,0x0l,0xC0,0x80,0x41,0x00,0xCl,0x81,0x40,0x00,0xC1,0x81,0x40,0x01,
0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x01,0xC0,0x80,0x41,0x01,0xC0,0x80,0x41,
0x00,0xCl,0x81,0x40,0x01,0xC0,0x80,0x41,0x00,0xC1,0x81,0x40,0x00,0xC1,0x81,
0x40,0x0l,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,
0x00,0xC1,0x81,0x40,0x00,0xCl,0x81,0x40,0x0l,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,
0x08,0xC8,0xD8,0x18,0xl9,0xD9,0xlB,0xDB,0xDA,0xlA,0xlE,0xDE,0xDF,0xlF,0xDD,
0xlD,0xlC,0xDC,0x14,0xD4,0xD5,0x15,0xD7,0x17,0x16,0xD6,0xD2,0x12,0x13,0xD3,
0xll,0xDl,0xD0,0xl0,0xF0,0x30,0x31,0xF1,0x33,0xF3,0xF2,0x32,0x36,0xF6,0xF7,
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 ^ *puchMsg++;/* calculate the CRC*/
        uchCRCHi = uchCRCLo ^ auchCRCHi [uIndex] ;
        uchCRCLo = auchCRCLo [uIndex] ;
    }

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

附录 C – 环境条件

运行环境条件

运行环境条件是指可选模块长期使用时，安装地点的环境条件。

空气温度: 0 至 +50 ℃

相对湿度: 5% 至 95 %，无凝露。如果有腐蚀性气体存在，最大允许相对湿度为 60 % 。

污染等级:
化学气体 : IEC 721-3-3, 3C2 级
固体粉尘 : IEC 721-3-3, 3S2 级

安装现场高度: 海拔 0 至 2000 m。如果安装现场高于海拔 2000 m,请与当地 ABB 代表联系。

振动: 最大 0.3 mm (2 至 9 Hz), 最大 1 m/s^2 (9 至 200 Hz) 正弦波 (IEC 68-2-6)

冲击: 最大 70 m/s^2, 11 ms (IEC 68-2-27)

存储环境条件

存储环境条件是指可选模块在保护性包装内存储时的环境条件。

温度: -40 至 +70 ℃

相对湿度: 小于 95 %, 无凝露

大气压: 70 至 106 kPa

振动: 最大 0.3 mm (2 至 9 Hz), 最大 1 m/s^2 (9 至 200 Hz) 正弦波 (IEC 68-2-6)

冲击: 最大 100 m/s^2, 11 ms (IEC 68-2-27)

运输环境条件

运输环境条件是指可选模块在保护性包装内运输时的环境条件。

温度: -40 至 +70 ℃

相对湿度: 小于 95 %, 无凝露

大气压: 60 至 106 kPa

振动: 最大 3.5 mm (2 至 9 Hz), 最大max 15 m/s^2 (9 至 200 Hz) 正弦波 **(IEC 68-2-6)**

冲击: 最大 100 m/s^2, 11 ms (IEC 68-2-27)

撞击: 最大 300 m/s^2, 6 ms (IEC 68-2-29)

自由跌落 : 250 mm

本页特意留为空白。

NMBA-01/EN
3ABD 00004106 R0225 REV B
EFFECTIVE: 1.2.1999
SUPERSEDES: 1997-01-10

北京 ABB电气传动系统有限公司
中国，北京，100076
北京经济技术开发区宏达北路8号，4号厂房
电话：(8610) 67881248
电传：(8610) 67881260