Allen-Bradley Micro800 Serial 驱动程序

目录

Allen-Bradley Micro800 Serial 驱动程序

帮助版本 1.036

目录

概述

Allen-Bradley Micro800 Serial 驱动程序 提供将 Allen-Bradley Micro800 串行控制器连接到 OPC 客户端应用程序的可靠方式；其中包括 HMI、SCADA、Historian、MES、ERP 和无数自定义应用程序。

设置

支持的设备

Micro830
Micro850

注意：将通过嵌入式串行端口或插件串行模块建立连接。

通信协议

Rockwell 自动化碎片协议 (基于 DF1 的 CIP)。

DH-485 和 DH+ 支持

要将驱动程序连接到 DH-485 网络需要使用 Allen Bradley KF3 或兼容设备。使用 Allen-Bradley Micro800 Serial 驱动程序 与 DH+ 中的设备进行通信的方式有四种：

- Allen Bradley KF2 或兼容设备。
- 1784-U2DHP USB 转换器。此转换器显示为系统的新串行端口。
- DataLink DL 接口卡 (PCI/ISA/PC104)。这些卡为无缝配置添加虚拟串行端口。
- DataLink DL4500 以太网到 DH+ 转换器。配置用于“以太网封装”的设备。NIC 是必需项。

以太网封装

此驱动程序支持以太网封装，允许驱动程序使用终端服务器与连接到以太网的串行设备进行通信。可以通过导槽属性中的 **Physical Medium** 调用以太网封装模式。

通道和设备限制

此驱动程序支持的最大通道数量为 256。此驱动程序所支持设备的最大数量为每通道 1024 个。

通道属性 - 常规

此服务器支持同时使用多个通信驱动程序。服务器项目中使用的各个协议或驱动程序称为通道。服务器项目可以由具有相同通信驱动程序或具有唯一通信驱动程序的多个通道组成。通道充当 OPC 链路的基础构建块。此组用于指定常规通道属性，如标识属性和操作模式。

标识

"名称"：指定此通道的用户定义标识。在每个服务器项目中，每个通道名称都必须是唯一的。尽管名称最多可包含 256 个字符，但在浏览 OPC 服务器的标记空间时，一些客户端应用程序的显示窗口可能不够大。通道名称是 OPC 浏览器信息的一部分。该属性是创建通道所必需的。
有关保留字符的信息，请参阅服务器帮助中的"如何正确命名通道、设备、标记和标记组"。

"说明"：指定此通道的用户定义信息。
在这些属性中，有很多属性 (包括"说明") 具有关联的系统标记。

"驱动程序"：为该通道指定的协议/驱动程序。该属性指定在通道创建期间选择的设备驱动程序。它在通道属性中为禁用设置。该属性是创建通道所必需的。
注意：服务器全天在线运行时，可以随时更改这些属性。其中包括更改通道名称以防止客户端向服务器注册数据。如果客户端在通道名称更改之前已从服务器中获取了项，那么这些项不会受到任何影响。如果客户端应用程序在通道名称更改之后发布项，并尝试通过原来的通道名称重新获取项，则该项将不被接受。考虑到这一点，一旦开发完成大型客户端应用程序，就不应对属性进行任何更改。采用适当的用户角色和权限管理来防止操作员更改属性或访问服务器功能。

诊断

"诊断数据捕获"：启用此选项后，通道的诊断信息即可提供给 OPC 应用程序，。由于服务器的诊断功能所需的开销处理量最少，因此建议在需要时使用这些功能，而在不需要时禁用这些功能。默认设置为禁用状态。
注意：如果驱动程序不支持诊断，则该属性不可用。
*有关详细信息，请参阅*服务器帮助中的*"通信诊断"和"统计信息标记"。*

通道属性 - 串行通信

串行通信属性可用于串行驱动程序，且随驱动程序、连接类型以及所选选项的不同而变化。以下是可能具有的属性的超集。
单击跳转至下列其中一个部分：**"连接类型"**、**"串行端口设置"**或**"以太网设置"**以及**"操作行为"**。

注意：服务器全天在线运行时，可以随时更改这些属性。采用适当的用户角色和权限管理来防止操作员更改属性或访问服务器功能。

连接类型

"物理媒体"：选择用于数据通信的硬件设备的类型。选项包括"COM 端口"、"无"、"调制解调器"和"以太网封装"。默认选项为 COM 端口。

- "无"：选择"无"表示没有物理连接，此时将显示**"无通信的操作"**部分。
- **"COM 端口"**：选择"COM 端口"可显示和配置**"串行端口设置"**部分。
- **"调制解调器"**：当用电话线进行通信时，选择"调制解调器"，并在**"调制解调器设置"**部分中对该选项进行配置。
- **"以太网封装"**：选择是否将"以太网封装"用于通信，此时将显示**"以太网设置"**部分。
- **"共享"**：验证是否已将连接正确标识为与其他通道共享当前配置。为只读属性。

串行端口设置

"COM ID"：指定在与分配给通道的设备进行通信时要使用的通信 ID。有效范围为 1 至 9991 至 16。默认值为 1。

"波特率"：指定用于配置选定通信端口的波特率。

"数据位"：指定每个数据字的数据位数。选项包括 5、6、7 或 8。

"奇偶性"：指定数据的奇偶类型。选项包括"奇"、"偶"或"无"。

"停止位"：指定每个数据字的停止位数。选项包括 1 或 2。

"流量控制"：选择 RTS 和 DTR 控制线的使用方式。在与一些串行设备进行通信时需要对流量进行控制。选项包括：

- **"无"**：此选项不会切换或添加控制线。
- **"DTR"**：当通信端口打开并保持开启状态时，此选项将添加 DTR 线路。
- **"RTS"**：此选项指定，如果字节适用于传输，则 RTS 线路为高电平。在发送所有缓冲字节后，RTS 线路变为低电平。这通常用于 RS232/RS485 转换器硬件。
- **"RTS, DTR"**：此选项是 DTR 和 RTS 的组合选项。
- **"始终 RTS"**：当通信端口打开并保持开启状态时，此选项将添加 RTS 线路。
- **"RTS 手动"**：此选项将基于为"RTS 线路控制"输入的定时属性添加 RTS 线路。该选项仅在驱动程序支持手动 RTS 线路控制 (或属性共享且至少有一个通道属于提供此类支持的驱动程序) 时可用。"RTS 手动"添加**"RTS 线路控制"**属性时具有如下选项：
  - **"上升"**：该属性用于指定在数据传输前 RTS 线路上升为高电平所需的时间量。有效范围为 0 至 9999 毫秒。默认值为 10 毫秒。
  - **"下降"**：该属性用于指定在数据传输后 RTS 线路保持高电平的时间量。有效范围为 0 至 9999 毫秒。默认值为 10 毫秒。
  - **"轮询延迟"**：该属性用于指定通信轮询的延迟时间量。有效范围为 0 到 9999。默认值为 10 毫秒。

提示：在使用双线 RS-485 时，通信线路上可能会出现“回波”。由于此类通信不支持回波抑制，因此建议禁用回波或使用 RS-485 转换器。

操作行为

- **“报告通信错误”**：启用或禁用报告低级通信错误。启用时，如果出现低级错误，则会将其发布到“事件日志”。禁用时，即使正常请求失败，也不会发布这些相同的错误。默认设置为“启用”。
- **“关闭空闲连接”**：当通道上的客户端不再引用任何标记时，选择关闭通道连接。默认设置为“启用”。
- **“关闭前空闲时间”**：指定在移除所有标记后服务器在关闭 COM 端口前所等待的时间。默认值为 15 秒。

以太网设置

注意：不是所有的串行驱动程序都支持以太网封装。若此组未出现，则无法支持相关功能。

如果要同与以太网终端服务器相连的串行设备进行通信，则可通过“以太网封装”来实现。终端服务器本质上是将以太网上的 TCP/IP 消息转换为串行数据的虚拟串行端口。消息转换完毕后，用户可将支持串行通信的标准设备连接到终端服务器。必须对终端服务器的串行端口进行正确配置，以满足所连串行设备的要求。*有关详细信息，请参阅服务器帮助中的“使用以太网封装”。*

- **“网络适配器”**：用于指示此通道中以太网设备绑定的网络适配器。选择要绑定的网络适配器，或者允许操作系统选择默认项。
  某些特定的驱动程序可能会显示其他“以太网封装”属性。有关详细信息，请参阅“通道属性 - 以太网封装”。

调制解调器设置

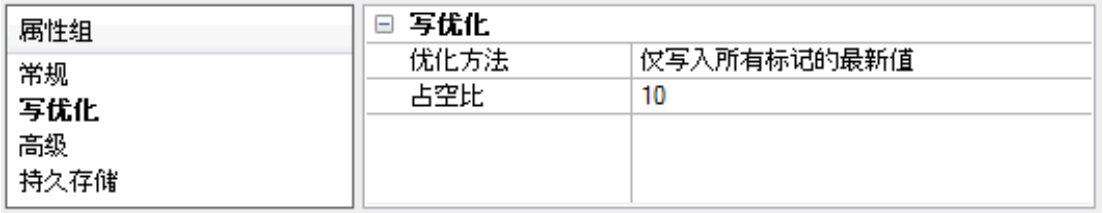
- **“调制解调器”**：指定用于通信的已安装调制解调器。
- **“连接超时”**：指定读取或写入失败前建立连接所等待的时间。默认值为 60 秒。
- **“调制解调器属性”**：配置调制解调器硬件。单击该选项后，将打开供应商特定的调制解调器属性。
- **“自动拨号”**：启用自动拨打电话簿中的条目。默认设置为“禁用”。*有关详细信息，请参阅服务器帮助中的“调制解调器自动拨号”。*
- **“报告通信错误”**：启用或禁用报告低级通信错误。启用时，如果出现低级错误，则会将其发布到“事件日志”。禁用时，即使正常请求失败，也不会发布这些相同的错误。默认设置为“启用”。
- **“关闭空闲连接”**：当通道上的客户端不再引用任何标记时，选择关闭调制解调器连接。默认设置为“启用”。
- **“关闭前空闲时间”**：指定在移除所有标记后服务器在关闭调制解调器连接前所等待的时间。默认值为 15 秒。

无通信的操作

- **“读取处理”**：选择要在请求显式设备读取时执行的操作。选项包括“忽略”和“失败”。“忽略”不执行任何操作；“失败”会为客户端提供一条指示失败的更新信息。默认设置为“忽略”。

通道属性 - 写入优化

服务器必须确保从客户端应用程序写入的数据能够准时发送到设备。为此，服务器提供了优化属性，用以满足特定需求或提高应用程序响应能力。



写入优化

“优化方法”：控制如何将写入数据传递至底层通信驱动程序。选项包括：

- **"写入所有标记的所有值"**：此选项可强制服务器尝试将每个值均写入控制器。在此模式下，服务器将持续收集写入请求并将它们添加到服务器的内部写入队列。服务器将对写入队列进行处理并尝试通过将数据尽快写入设备来将其清空。此模式可确保从客户端应用程序写入的所有数据均可发送至目标设备。如果写入操作顺序或写入项的内容必须且仅能显示于目标设备上，则应选择此模式。
- **"写入非布尔标记的最新值"**：由于将数据实际发送至设备需要一段时间，因此对同一个值的多次连续写入会存留于写入队列中。如果服务器要更新已位于写入队列中的某个写入值，则需要大大减少写入操作才能获得相同的最终输出值。这样一来，便不会再有额外的写入数据存留于服务器队列中。几乎就在用户停止移动滑动开关时，设备中的值达到其正确值。根据此模式的规定，任何非布尔值都会在服务器的内部写入队列中更新，并在下一个可能的时机发送至设备。这可以大大提高应用性能。
  注意：该选项不会尝试优化布尔值的写入。它允许用户在不影响布尔运算的情况下优化 HMI 数据的操作，例如瞬时型按钮等。
- **"写入所有标记的最新值"**：该选项采用的是第二优化模式背后的理论并将其应用至所有标记。如果应用程序只需向设备发送最新值，则该选项尤为适用。此模式会通过在当前写入队列中的标记发送前对其进行更新来优化所有的写入操作。此为默认模式。

"占空比"(Duty Cycle)：用于控制写操作与读操作的比率。该比率始终基于每一到十次写入操作对应一次读取操作。占空比的默认设置为 10，这意味着每次读取操作对应十次写入操作。即使在应用程序执行大量的连续写入操作时，也必须确保足够的读取数据处理时间。如果将占空比设置为 1，则每次读取操作对应一次写入操作。如果未执行任何写入操作，则会连续处理读取操作。相对于更加均衡的读写数据流而言，该特点使得应用程序的优化可通过连续的写入操作来实现。
注意：建议在将应用程序投入生产环境前使其与写入优化增强功能相兼容。

通道属性 - 高级

此组用于指定高级通道属性。并非所有驱动程序都支持所有属性，因此不会针对不支持的设备显示"高级"组。

"非规范浮点数处理"：非规范值定义为无穷大、非数字 (NaN) 或不正规编号。默认值为"替换为零"。具有原生浮点数处理功能的驱动程序可能会默认设置为"未修改"。通过非规范浮点数处理，用户可以指定驱动程序处理非规范 IEEE-754 浮点数据的方式。选项说明如下：

- **"替换为零"**：此选项允许驱动程序在将非规范 IEEE-754 浮点值传输到客户端之前，将其替换为零。
- **"未修改"**：此选项允许驱动程序向客户端传输 IEEE-754 不正规、规范、非数字和无穷大值，而不进行任何转换或更改。

注意：如果驱动程序不支持浮点值或仅支持所显示的选项，则此属性不可用。根据通道的浮点规范化设置，将仅对实时驱动程序标记 (如值和数组) 进行浮点规范化。例如，此设置不会影响 EFM 数据。

有关浮点值的详细信息，请参阅服务器帮助中的"如何使用非规范化浮点值"。

"设备间延迟"：指定在接收到同一通道上的当前设备发出的数据后，通信通道向下一设备发送新请求前等待的时间。设置为零 (0) 将禁用延迟。

注意：此属性并不适用于所有驱动程序、型号和相关设置。

通道属性 - 通信序列化

服务器的多线程架构使通道能够与设备并行通信。尽管这十分高效，但在存在物理网络限制 (如以太网无线电) 的情况下，通信可能会进行序列化。通信序列化将限制在虚拟网络中每次仅使用一个通道进行通信。

术语"虚拟网络"是指使用同一管线进行通信的通道和相关设备的集合。例如，以太网无线电管线是客户端无线电。使用同一客户端无线电的所有通道均与同一虚拟网络相关联。通道能够以"循环"方式轮流进行通信。默认情况下，通道在向另一通道传递通信前，可处理一个事务。一个事务中可包括一个或多个标记。如果控

制通道包含的设备未响应请求，则在事务超时之前，通道无法释放控制权。这会导致虚拟网络中其他通道的数据更新延迟。

通道级别设置

"虚拟网络"：指定通道的通信序列化模式。选项包括"无"和"网络 1 - 网络 500"。默认值为"无"。选项说明如下：

- **"无"**：此选项禁用通道的通信序列化。
- **"网络 1 - 网络 500"**：此选项可指定分配通道的虚拟网络。

"每周期的事务数"：指定通道中可能发生的单一分块/非分块读/写事务的数量。当通道可以进行通信时，将尝试该事务数。有效范围为 1 到 99。默认值为 1。

全局设置

- **"网络模式"**：此属性用于控制委派通道通信的方式。在**"负载平衡"**模式下，每个通道可以逐一轮流进行通信。在**"优先级"**模式下，通道可以根据以下规则 (优先级由高到低) 进行通信：
  - 具有待处理写入操作的通道具有最高优先级。
  - 具有待处理显式读取操作 (通过内部插件或外部客户端接口) 的通道的优先级基于读取的优先级。
  - 扫描读取和其他定期事件 (特定于驱动程序)。

  默认设置为"负载平衡"，这并影响*所有*虚拟网络和通道。

依赖于主动响应的设备不应置于虚拟网络中。在必须进行通信序列化的情况下，建议启用"自动降级"。

由于驱动程序的数据读取和写入方式的差异 (如单一、分块或非分块事务)，可能需要调整应用程序的"每周期的事务数"属性。执行此操作时，请考虑以下因素：

- 必须从每个通道读取多少标记？
- 数据写入各个通道的频率如何？
- 通道使用串行驱动程序还是以太网驱动程序？
- 驱动程序是读取单独请求中的标记还是读取块中的多个标记？
- 设备的定时属性 (如请求超时和 *x* 次连续超时后失败) 是否针对虚拟网络通信媒介进行了优化？

通道属性 - 以太网封装

"以太网封装"可用于无线网络连接 (例如 802.11b 和 CDPD 数据包网络)，并且还经过开发，可以支持多种串行设备。通过终端服务器设备，用户可在工厂中放置 RS-232 和 RS-485 设备，同时仍然允许单个本地化 PC 访问远程挂载设备。"以太网封装"还可以根据需要将各个网络 IP 地址分配到设备。通过使用多个终端服务器，用户可以从单个 PC 访问数百个串行设备。用户可以将一个通道定义为使用本地 PC 串行端口，而将另一个通道定义为使用"以太网封装"。

注意：这些属性仅适用于串行驱动程序。显示的属性取决于所选通信驱动程序和支持的功能。

"网络适配器"：指定网络适配器。

"设备地址"：指定与此设备连接的终端服务器的四字段 IP 地址。IP 指定为 *YYY.YYY.YYY.YYY*。*YYY* 可指定 IP 地址：每个 *YYY* 字节应在 0 至 255 的范围内。每个通道均有其自己的 IP 地址。

"端口"：配置在连接到远程终端服务器时使用的以太网端口。有效范围是 1 至 65535，其中某些数字予以保留。默认值为 2101。

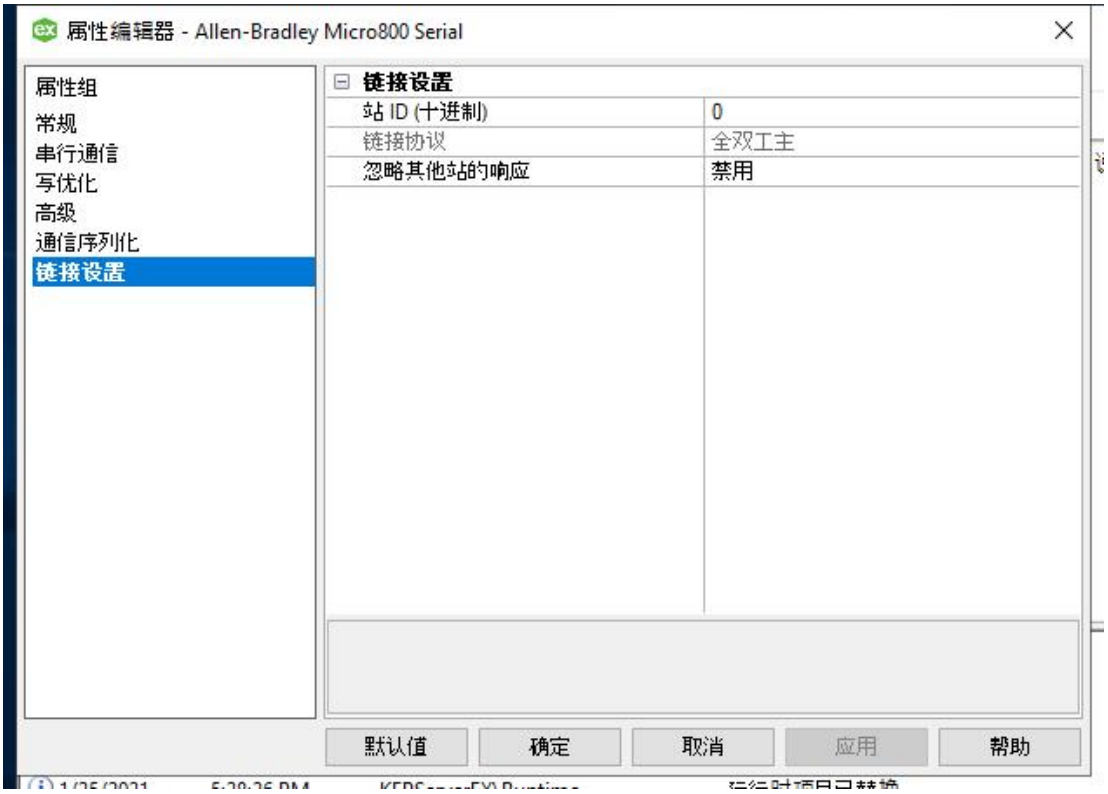
"协议"：指定 TCP/IP 或 UDP 通信，具体取决于正在使用的终端服务器的性质。默认值为 TCP/IP。*有关可用协议的详细信息，请参阅终端服务器的帮助文档。*

重要事项："以太网封装"模式对于实际的串行通信驱动程序是完全透明的。用户必须配置其余的设备属性，如同他们直接通过本地 PC 串行端口连接到设备一样。

"连接超时"：指定为要调整的远程设备建立套接字连接所需的时间。在许多情况下，设备的连接时间比向该同一设备发送正常通信请求所需的时间更长。有效范围为 1 到 999 秒。默认值为 3 秒。

注意：服务器全天在线运行时，可以随时更改这些属性。采用适当的用户角色和权限管理来防止操作员更改属性或访问服务器功能。

通道属性 - 链路设置



链路设置

"工作站 ID"(Station ID)：该属性为本地计算机指定唯一的网络 ID。这应根据其正在通信的设备 (不包括无线调制解调器) 进行设置。格式为十进制。默认值为 0。

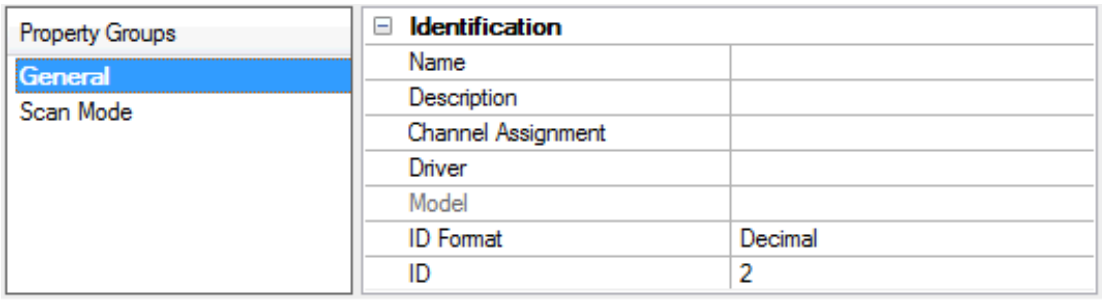
"链接协议"(Link Protocol)：Allen-Bradley Micro800 Serial 驱动程序 支持 "全双工"(Full-Duplex)，用于点对点链接，可实现对等方之间的高性能双向通信。

"忽略其他站的响应"(Ignore Responses for other Stations)：启用后，该属性会限制接收原定发往"工作站 ID" (Station ID) 指定的工作站的响应。此属性仅适用于全双工。默认设置为禁用状态。

注意：如果目标设备在 DH + 或 DH 485 网络上，则必须通过 Serial-to-DH+/DH-485 转换器 (即 KF2/KF3 模块) 进行通信。在这种情况下，进行通信的设备是转换器，而非目标设备本身。此配置的工作站 ID 应设置为转换器的节点地址。DH-4850 的范围是 1 到 63。如果目标设备不在 DH+ 或 DH-485 网络上，则进行通信的设备为 Micro800。此配置的工作站 ID 可以设置为任意的唯一地址。范围是 0 到 255。

设备属性 - 常规

一个设备代表通信通道上的单一目标。如果驱动程序支持多个控制器，则用户必须为每个控制器输入一个设备 ID。



标识

"名称"：指定设备的名称。此为用户定义的逻辑名称，最长可达 256 个字符，并且可以用于多个通道。

注意：尽管描述性名称通常是不错的选择，但浏览 OPC 服务器的标记空间时，一些 OPC 客户端应用程序的显示窗口可能不够大。设备名称和通道名称也成为浏览树信息的一部分。OPC 客户端中，通道名称和设备名称的组合将显示为"通道名称.设备名称"。
有关详细信息，请参阅服务器帮助中的"如何为通道、设备、标记和标记组正确命名"。

"说明"：指定此设备的用户定义信息。
在这些属性中，有很多属性 (包括"说明") 具有关联的系统标记。

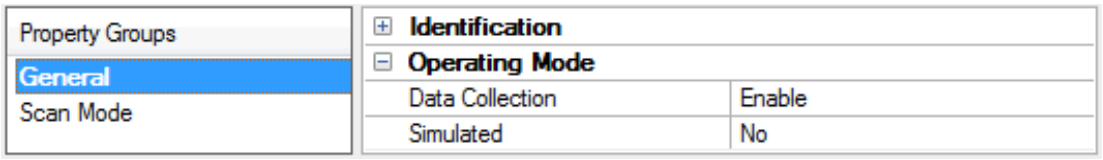
"通道分配"：指定该设备当前所属通道的用户定义名称。

驱动程序：为该设备选择的协议驱动程序。

"型号"：指定与此 ID 关联的设备的类型。下拉菜单中的内容取决于正在使用的通信驱动程序类型。驱动程序不支持的型号将被禁用。如果通信驱动程序支持多个设备型号，则只有当设备未与任何客户端应用程序连接时，才能改变型号的选择。
注意：如果通信驱动程序支持多种模型，则用户应将模型选择与物理设备进行匹配。如果下拉列表菜单中未显示该设备，则选择与目标设备最相近的模型。一些驱动程序支持名为"开放式"的型号选择，该选择使用户无需了解目标设备的具体信息即可进行通信。有关详细信息，请参阅驱动程序帮助文档。

ID：指定设备驱动程序特定的工作站或节点。输入的 ID 类型取决于正在使用的通信驱动程序。对于许多通信驱动程序而言，ID 是一个数值。支持数字 ID 的驱动程序使用户能够输入格式可更改的数值，以适应应用程需要或所选通信驱动程序特点。默认情况下，该格式由驱动程序设置。选项包括十进制、八进制和十六进制。
注意：如果驱动程序基于以太网，或者支持非常规工作站或节点名称，则可使用设备的 TCP/IP 地址作为设备 ID。TCP/IP 地址包含四个由句点分隔的值，每个值的范围在 0 至 255 之间。某些设备 ID 基于字符串。根据不同驱动程序，也可以在 ID 字段中配置其他属性。*有关详细信息，请参阅驱动程序的帮助文档。*

操作模式



数据收集：此属性控制设备的活动状态。尽管默认情况下会启用设备通信，但可使用此属性禁用物理设备。设备处于禁用状态时，不会尝试进行通信。从客户端的角度来看，数据将标记为无效，且不接受写入操作。通过此属性或设备系统标记可随时更改此属性。

"模拟"：使设备进入或退出模拟模式。在此模式下，驱动程序不会尝试与物理设备进行通信，但服务器将继续返回有效的 OPC 数据。模拟停止与设备的物理通信，但允许 OPC 数据作为有效数据返回到 OPC 客户端。在"模拟模式"下，服务器将所有设备数据处理为反射型：无论向模拟设备写入什么内容，都会读取回来，而且会单独处理每个 OPC 项。项的内存映射取决于组更新速率。如果服务器移除了项 (如服务器重新初始化时)，则不保存数据。默认值为"否"。
注意：

1. “系统”标记 (_Simulated) 为只读且无法写入，从而达到运行时保护的目的。“系统”标记允许从客户端监控此属性。
2. 在“模拟”模式下，项的内存映射取决于客户端更新速率 (OPC 客户端的“组更新速率”或本机和 DDE 接口的扫描速率)。这意味着，参考相同项、而采用不同更新速率的两个客户端会返回不同的数据。

“模拟模式”仅用于测试和模拟目的。该模式永远不能用于生产环境。

设备属性 - 以太网封装

“以太网封装”旨在为通过以太网与终端服务器相连的串行设备提供通信。终端服务器实质上是虚拟串行端口。终端服务器会将以太网上的 TCP/IP 消息转换为串行数据。消息转换为串行形式后，用户可将支持串行通信的标准设备连接到终端服务器。

有关详细信息，请参阅服务器帮助中的“如何使用以太网封装”。

“以太网封装”对于驱动程序来说是透明的；配置其余属性时，应与直接连接本地串行端口上的设备一样。

“IP 地址”：输入与设备连接的终端服务器的四字段 IP 地址。IP 指定为 YYY.YYY.YYY.YYY。YYY 指定 IP 地址：每个 YYY 字节应在 0 至 255 的范围内。每个串行设备都可以有其自己的 IP 地址；但是，如果多个设备与单个终端服务器进行多点通信时，则这些设备可能使用相同的 IP 地址。

“端口”：配置在连接到远程终端服务器时使用的以太网端口。

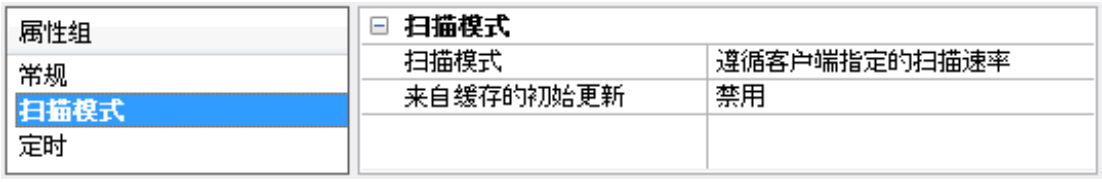
协议：设置 TCP/IP 或 UDP 通信。该选择取决于正在使用的终端服务器的性质。默认协议选项为 TCP/IP。有关可用协议的详细信息，请参阅终端服务器的帮助文档。

注意

1. 服务器全天在线运行时，可以随时更改这些属性。采用适当的用户角色和权限管理来防止操作员更改属性或访问服务器功能。
2. 有效的 IP 地址范围大于 (>) 0.0.0.0 且小于 (<) 255.255.255.255。

设备属性 - 扫描模式

“扫描模式”为需要设备通信的标记指定订阅客户端请求的扫描速率。同步和异步设备的读取和写入会尽快处理；不受“扫描模式”属性的影响。



“扫描模式”：为发送到订阅客户端的更新指定在设备中扫描标记的方式。选项说明如下：

- **“遵循客户端指定的扫描速率”：**此模式可使用客户端请求的扫描速率。
- **“不超过扫描速率请求数据”：**此模式可将该数值集指定为最大扫描速率。有效范围为 10 至 99999990 毫秒。默认值为 1000 毫秒。

  注意：当服务器有活动的客户端和设备项且扫描速率值有所提高时，更改会立即生效。当扫描速率值减小时，只有所有客户端应用程序都断开连接，更改才会生效。
- **“以扫描速率请求所有数据”：**此模式将以订阅客户端的指定速率强制扫描标记。有效范围为 10 至 99999990 毫秒。默认值为 1000 毫秒。

- **“不扫描，仅按需求轮询”**：此模式不会定期轮询属于设备的标签，也不会在一个项变为活动状态后为获得项的初始值而执行读取操作。客户端负责轮询以便更新，方法为写入 _DemandPoll 标记或为各项发出显式设备读取。*有关详细信息，请参阅服务器帮助中的“设备需求轮询”*。
- **“遵循标签指定的扫描速率”**：此模式将以静态配置标记属性中指定的速率强制扫描静态标记。以客户端指定的扫描速率扫描动态标记。

“来自缓存的初始更新”：启用后，此选项允许服务器为存储 (缓存) 数据的新激活标签参考提供第一批更新。只有新项参考共用相同的地址、扫描速率、数据类型、客户端访问和缩放属性时，才能提供缓存更新。设备读取仅用于第一个客户端参考的初始更新。默认设置为禁用；只要客户端激活标记参考，服务器就会尝试从设备读取初始值。

设备属性 - 定时

设备的“定时”属性允许调整驱动程序对错误条件的响应，以满足应用程序的需要。在很多情况下，需要更改环境的此类属性，以便获得最佳性能。由电气原因产生的噪音、调制解调器延迟以及较差的物理连接等因素都会影响通信驱动程序遇到的错误数或超时次数。“定时”属性特定于每个配置的设备。

通信超时

“连接超时”(Connect Timeout)：此属性 (主要由基于驱动程序的以太网使用) 控制建立远程设备套接字连接所需的时间长度。设备的连接时间通常比针对同一设备的正常通信请求所花费时间更长。有效范围为 1 到 30 秒。默认值通常为 3 秒钟，但可能会因驱动程序的具体性质而异。如果驱动程序不支持此设置，则此设置将被禁用。

注意：鉴于 UDP 连接的性质，当通过 UDP 进行通信时，连接超时设置不适用。

“请求超时”：指定一个所有驱动程序使用的间隔来决定驱动程序等待目标设备完成响应的时间。有效范围是 50 至 9,999,999 毫秒 (167.6667 分钟)。默认值通常是 1000 毫秒，但可能会因驱动程序而异。大多数串行驱动程序的默认超时是基于 9600 波特或更高的波特率来确定的。当以较低的波特率使用驱动程序时，请增加超时，以补偿获取数据所需增加的时间。

“超时前的尝试次数”：指定在认定请求失败以及设备出错之前，驱动程序发出通信请求的次数。有效范围为 1 到 10。默认值通常是 3，但可能会因驱动程序的具体性质而异。为应用程序配置的尝试次数很大程度上取决于通信环境。此属性适用于连接尝试和请求尝试。

定时

“请求间延迟”：指定驱动程序在将下一个请求发送到目标设备之前等待的时间。它会覆盖设备关联标记的一般轮询频率，以及一次性读取和写入次数。在处理周转时间慢的设备时，以及担心网络负载问题时，这种延迟很有用。为设备配置延迟会影响与通道上所有其他设备的通信。建议用户尽可能将所有需要请求间延迟的设备隔离至单独的通道。其他通信属性 (例如通信序列化) 可以延长此延迟。有效范围是 0 至 300,000 毫秒；但是，某些驱动程序可能因某项特别设计的功能而限制最大值。默认值为 0，它表示对目标设备的请求之间没有延迟。

注意：不是所有的驱动程序都支持“请求间延迟”。如果不可用，则此设置不会出现。

设备属性 - 自动降级

自动降级属性可以在设备未响应的情况下使设备暂时处于关闭扫描状态。通过将特定时间段内无响应的设备脱机，驱动程序可以继续优化与同一通道上其他设备的通信。该时间段结束后，驱动程序将重新尝试与无响应设备进行通信。如果设备响应，则该设备会进入开启扫描状态；否则，设备将再次开始其关闭扫描时间段。

"故障时降级"：启用后，将自动对设备取消扫描，直到该设备再次响应。

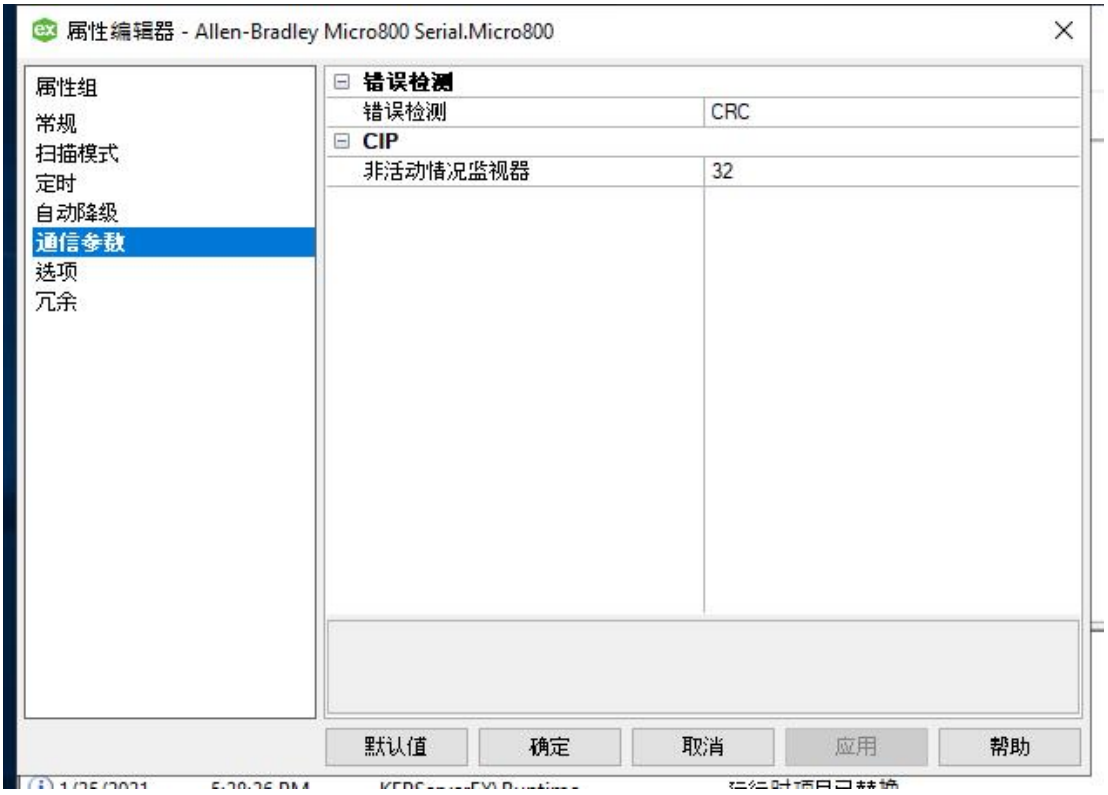
提示：使用 _AutoDemoted 系统标记来监视设备的降级状态，确定何时对设备取消扫描。

"降级超时"：指定在对设备取消扫描之前，请求超时和重试的连续周期数。有效范围是 1 到 30 次连续失败。默认值为 3。

"降级期间"：指示当达到超时值时，对设备取消扫描多长时间。在此期间，读取请求不会被发送到设备，与读取请求关联的所有数据都被设置为不良质量。当此期间到期时，驱动程序将对设备进行扫描，并允许进行通信尝试。有效范围为 100 至 3600000 毫秒。默认值为 10000 毫秒。

"降级时放弃请求"：选择是否在取消扫描期间尝试写入请求。如果禁用，则无论是否处于降级期间都始终发送写入请求。如果启用，则放弃写入；服务器自动将接收自客户端的写入请求视为失败，且不会在事件日志中记录消息。

设备属性 - 通信参数



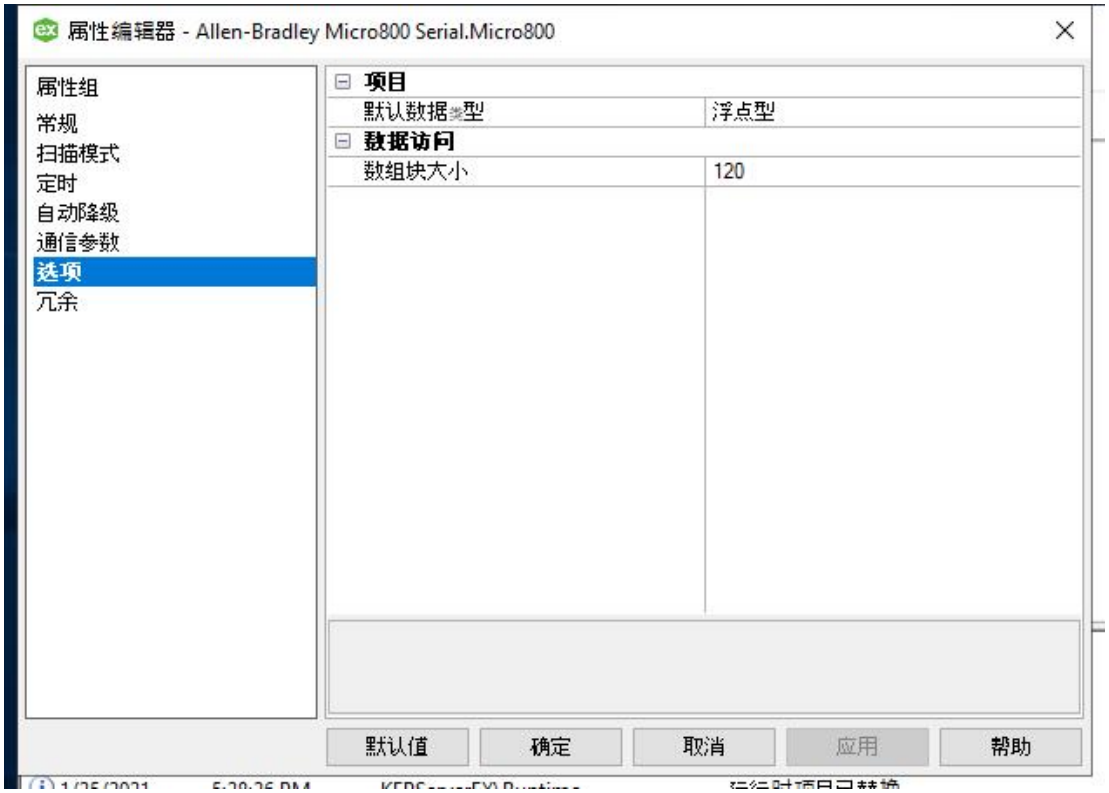
错误检测

"错误检测"：从块校验字符 (BCC) 或循环冗余校验 (CRC) 选择设备所需的错误检测方法/校验方法。默认为 CRC。

CIP

“非活动情况监视器”(Inactivity Watchdog)：指定连接在由控制器关闭之前保持空闲状态 (不进行读/写事务处理) 的时长 (以秒为单位)。通常情况下，此值越大，控制器释放连接资源所需的时间越长 (反之亦然)。默认值为 32 秒。

设备属性 - 选项



项目

默认数据类型：在添加/修改/导入标记时选择默认类型的情况下，选择分配给客户端/服务器标记的数据类型。默认值为浮点型。在客户端中使用“本机”作为其分配的数据类型创建动态标记时，以及在服务器中使用“默认”作为其分配的数据类型创建静态标记时，将为标记分配默认数据类型。

数据访问

“数组块大小”：指定在单个事务处理中要读取的原子型数组元素的最大数量。范围介于 30 到 3840 个元素之间。默认值为 120 个元素。

对于布尔型数组，单个元素将被视为 32 元素位数组。将块大小设置为 30 个元素，则会转换为 960 个位元素，而 3840 个元素则会转换为 122880 个位元素。

设备属性 - 冗余

Media-Level Redundancy 插件提供冗余。

- *有关详细信息，请参阅网站、向销售代表咨询或查阅用户手册。*

性能优化

多个指南可用于 Allen-Bradley Micro800 Serial 驱动程序 以获得最佳性能。有关通信和应用程序级别的优化的详细信息，请从下表中选择一个链接。

优化通信
优化应用程序

优化通信

与任何可编程控制器一样，可使用多种方法来增强整体性能和系统通信。

保持原生标记名称简短

通过指定通信请求中的符号名称从设备中读取以及向设备中写入原生标记。因此，标记名称越长，请求就越大。

已分块的数组元素

要优化原子型数组元素的读取，请在单个请求中以块的形式读取数组，而不是单独读取。每个块中读取的元素越多，性能越好。由于事务处理费用较高且处理会消耗大多数时间，因此，在尽可能多地扫描所需标记的同时要尽可能少地进行事务处理。此为数组元素块的实质。

规定块大小为元素计数。块大小为 120 个元素，意味着在一个请求中最多能够读取 120 个数组元素。块大小最大为 3840 个元素。布尔型数组的处理方式会不同：在协议中，布尔型数组是一个 32 位数组。因此，请求元素 0 将请求位 0 至 31。为了保持讨论的一致性，会将布尔型数组元素视为单个位。可请求的数组元素最大数量 (基于块大小 3840) 汇总如下：122880 BOOL、3840 SINT、3840 INT、3840 DINT、3840 LINT 和 3840 REAL。

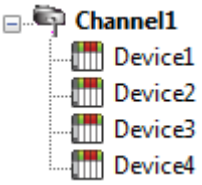
块大小是可调整的，且应该根据当前项目进行选择。例如，如果数组元素 0-26 和元素 3839 是要读取的标记，那么，使用 3840 的块大小不仅过大，而且影响驱动程序的性能。这是因为：尽管只需要 28 个元素，但是会针对每个请求读取 0 和 3839 之间的所有元素。在这种情况下，块大小设为 30 更合适。在一个请求中读取元素 0-26，而在下一个请求中读取元素 3839。

另请参阅：选项

优化应用程序

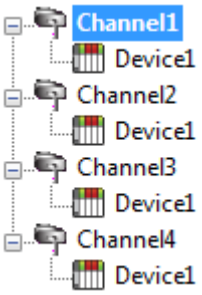
Allen-Bradley Micro800 Serial 驱动程序 旨在提供最佳性能，使得其对系统的整体性能影响最小。即使驱动程序速度很快，也可以利用一系列指南来获得最佳性能。

服务器将诸如 Allen-Bradley Micro800 Serial 等通信协议称为通道。应用程序中定义的每个通道都表示服务器中一个单独的执行路径。一旦定义了通道，则必须在该通道下定义一系列设备。每一个此类设备都代表一个可从中收集数据的 Micro800 CPU。虽然这种定义应用程序的方法提供了高水平的性能，但它不能充分利用 Allen-Bradley Micro800 Serial 驱动程序 或网络。下面显示了使用单个通道配置时应用程序所呈现效果的示例。



每个设备均出现在单个通道下。在此配置中，驱动程序必须尽快从一个设备移动到下一个设备，以有效速率收集信息。随着更多设备的添加或从单个设备请求的信息的增加，整体更新速率会受到不利影响。

如果 Allen-Bradley Micro800 Serial 驱动程序 只能定义一个单通道，则上述示例将是唯一可用的选项；但是，驱动程序最多可以定义 256 个通道。使用多个通道，可通过同时向网络发出多个请求来分发数据集合工作载荷。下面显示了使用多个通道来提高性能时相同应用程序所呈现效果的示例。



每个设备均可在其自身的通道下定义。在此配置中，单个执行路径专用于从每个设备收集数据。如果应用程序的设备数小于等于 256，则可完全按此处所示对其进行优化。

即使应用程序设备数较大，也可改善性能。虽然设备数较小可能是理想情况，但附加通道仍会对应用程序有益。尽管在全部通道上分散设备载荷会使服务器再次从一个设备移动到另一个设备，但是，它可以用极少的设备在单通道上进行处理。

数据类型说明

数据类型	说明
布尔型	单个位
字节	无符号 8 位值
字符	有符号 8 位值
字	无符号 16 位值
短整型	有符号 16 位值
双字型	无符号 32 位值
长整型	有符号 32 位值
BCD	两个字节封装的 BCD，四位十进制数字
LBCD	四个字节封装的 BCD，八位十进制数字
浮点型	32 位 IEEE 浮点
双精度	64 位 IEEE 浮点
日期	64 位日期/时间
字符串	空终止字符数组

地址说明

Micro800 使用一种基于标记或符号的寻址结构，称为“原生标记”。这些标记与常规 PLC 数据项的区别在于，标记名称本身是地址，而不是文件或寄存器编号。

Allen-Bradley Micro800 Serial 驱动程序 允许用户访问控制器的原子型数据类型：BOOL、SINT、USINT、BYTE、INT、UINT、WORD、DINT、UDINT、DWORD、LINT、ULINT、LWORD、REAL、LREAL 和 SHORT_STRING。尽管某些预定义类型为结构，但归根结底，它们基于基于这些原子型数据类型。因此，结构的所有非结构 (原子型) 成员均可供访问。例如，不能将 TIMER 分配给服务器标记，但是，可以将 TIMER 的原子型成员分配给标记 (例如 TIMER.EN, TIMER.ACC 等)。如果结构成员为结构本身，则必须展开这两个结构，以便访问子结构的原子型成员。对于用户定义和模块定义的类型，这种情况更为常见，但这不会出现在预定义类型中。

原子型数据类型	说明	客户端类型	范围
BOOL	单个位值	VT_BOOL	0、1
SINT	有符号 8 位值	VT_U1	-128 到 127
USINT	无符号 8 位值	VT_UI1	0 到 255
BYTE	位字符串 (8 位)	VT_UI1	0 到 255
INT	有符号 16 位值	VT_I2	-32,768 到 32,767
UINT	无符号 16 位值	VT_UI2	0 到 65535
WORD	位字符串 (16 位)	VT_UI2	0 到 65535
DINT	有符号 32 位值	VT_I4	-2,147,483,648 到 2,147,483,647
UDINT	无符号 32 位值	VR_UI4	0 到 4294967296
DWORD	位字符串 (32 位)	VR_UI4	0 到 4294967296
LINT	有符号 64 位值	VT_R8	-1.798E+308 到 -2.225E-308, 0, 2.225E-308 到 1.798E+308
ULINT	无符号 64 位值	VT_R8	-1.798E+308 到 -2.225E-308, 0, 2.225E-308 到 1.798E+308
LWORD	位字符串 (64 位)	VT_R8	-1.798E+308 到 -2.225E-308, 0, 2.225E-308 到 1.798E+308
REAL	32 位 IEEE 浮点点	VT_R4	1.1755 E-38 到 3.403E38、0、-3.403E-38 到 -1.1755
LREAL	64 位 IEEE 浮点	VT_R8	-1.798E+308 到 -2.225E-308, 0, 2.225E-308 到 1.798E+308
SHORT_STRING	字符字符串最大字符数为 80。	VT_BSTR	

另请参阅：高级用例

客户端/服务器标记地址规则
原生标记名称与客户端/服务器标记地址对应。原生标记名称 (通过 Connected Components Workbench 输入) 与客户端/服务器标记地址都遵循 IEC 1131-3 标识符规则。规则说明如下：

- 必须以字母字符或下划线开头
- 只能包含字母数字字符和下划线
- 最多可以包含 40 个字符
- 不能有连续的下划线
- 字符不区分大小写

为了获得最佳性能，请使原生标记名称保持最小。名称越短，单个事务可包含的请求就越多。

客户端/服务器标记名称规则
服务器中的标记名称分配与地址分配不同，因为名称不能以下划线开头。

另请参阅：性能优化

地址格式

进行原生标记寻址有多种方法：可以在服务器中静态寻址，也可以从客户端动态寻址。标记的格式取决于其类型及用途。例如，在“短整型”标记内访问位时，将使用位格式。有关地址格式和语法的信息，请参阅下表。

注意：对于一体化编程组态软件 (CCW) 来说，除数组之外的所有格式均为原生格式。因此，参考原子型数据类型时，可将 CCW 标记名称复制并粘贴到服务器的标记地址字段，并可供使用。

另请参阅：*高级用例*

格式	语法
数组元素	<原生标记名称> [1 维, 2 维, 3 维]
带偏移数组*	<原生标记名称> {列数} <原生标记名称> {行数}{列数}
无偏移数组*	<原生标记名称> {列数} <原生标记名称> {行数}{列数}
位	<原生标记名称>.位 <原生标记名称>.[位]
标准	<原生标记名称>
字符串	<原生标记名称>

*由于这些格式可以请求多个元素，因此数组数据的传递顺序取决于数组标记的维度。例如，如果行数乘以列数 = 4 且原生标记为 3X3 元素数组，则所参考的元素采用 array_tag [0,0]、array_tag [0,1]、array_tag [0,2] 和 array_tag [1,0] 的顺序。如果原生标记为 2X10 元素数组，则结果可能有所不同。有关详细信息，请参阅数组数据排序。

展开的地址格式

数组元素

必须指定至少 1 个维度 (但不能多于 3 个)。

语法	示例	注解
<原生标记名称> [1 维]	tag_1 [5]	不适用
<原生标记名称> [1 维, 2 维]	tag_1 [2, 3]	不适用
<原生标记名称> [1 维, 2 维, 3 维]	tag_1 [2, 58, 547]	不适用

带偏移数组

由于此类数组可以请求多个元素，因此数组数据的传递顺序取决于数组标记的维度。

语法	示例	注解
<原生标记名称> [偏移] {列数}	tag_1 [5] {8}	要读/写的元素数等于行数乘以列数。如果未指定任何行，则行数默认设置为 1。必须为至少一个数组元素寻址。 数组起点处具有零偏移 (所有维度的数组索引均等于 0)。
<原生标记名称> [偏移] {行数}{列数}	tag_1 [5] {2}{4}	

注意：如果行数乘以列数 = 4 且原生标记为 3X3 元素数组，则所参考的元素采用 array_tag [0,0]、array_tag [0,1]、array_tag [0,2] 和 array_tag [1,0] 的顺序。如果原生标记为 2X10 元素数组，则结果可能有所不同。

无偏移数组

由于此类数组可以请求多个元素，因此数组数据的传递顺序取决于数组标记的维度。

语法	示例	注解
<原生标记名称> {列数}	tag_1 {8}	要读/写的元素数等于行数乘以列数。如果未指定任何行，则行数默认设置为 1。必须为至少一个数组元素寻址。

语法	示例	注解
<原生标记名称>{行数}{列数}	tag_1{2}{4}	数组起点处具有零偏移 (所有维度的数组索引均等于 0)。

注意：例如，如果行数乘以列数 = 4 且原生标记为 3X3 元素数组，则所参考的元素采用 array_tag [0,0]、array_tag [0,1]、array_tag [0,2] 和 array_tag [1,0] 的顺序。如果原生标记为 2X10 元素数组，则结果可能有所不同。

位

语法	示例	注解
<原生标记名称> . 位	tag_1 . 0	不适用
<原生标记名称> .[位]	tag_1 . [0]	不适用

标准

语法	示例	注解
<原生标记名称>	tag_1	不适用

字符串

语法	示例	注解
<原生标记名称>	tag_1	要读/写的字符数等于字符串长度，并且必须至少为 1。

*有关 1 维、2 维和 3 维数组如何参考元素的详细信息，请参阅**数组数据排序**。*

标记范围

变量的范围可以是程序的本地变量，或控制器的全局变量。

- 本地变量被分配给项目中的特定程序；它们仅可用于该程序。
- 全局变量属于项目中的控制器；它们可用于项目中的任何程序。

本地变量

本地变量 (程序范围标记) 不能通过控制器的通信端口直接访问，因此在驱动程序内不直接支持本地变量。如果需要访问，请将标记从“本地”变量表剪切并粘贴到“全局”变量表。

全局变量

全局变量 (控制器范围标记) 是在控制器中具有全局范围的“原生标记”。任何程序或任务都可以访问“全局标记”；但是，“全局标记”的引用方式数量取决于其“本机数据类型”和使用的地址格式。

用户定义的数据类型

用户可以创建唯一的数据类型，例如：包含 12 个字符而非 80 个字符的字符串。这些用户定义的数据类型可以用作本地或全局变量。

结构化变量

在 Micro800 控制器中没有结构化变量。用户可以构建唯一的“数据类型”，但每个成员必须具有唯一的名称。

原子型数据类型寻址

下表列出了在给定可用地址格式的情况下，每种“原生数据类型”的建议使用情况和寻址可能性。有关每种数据类型的高级寻址可能性，请单击**“高级”**。

注意：空白单元不一定表示缺少支持。

BOOL

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型	布尔型	布尔型	布尔型数组		

标记	标准	数组元素	带/无偏移数组	位	字符串
高级		(BOOL 1 维数组)	(BOOL 1 维数组)		
示例	BOOLTAG	BOOLARR[0]	BOOLARR[0]{32}		

SINT、USINT 和 BYTE

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型 高级	字节、字符	字节、字符	字节数组、字符数组 (SINT 1/2/3 维数组)	布尔型 (SINT 中的位)	
示例	SINTTAG	SINTARR[0]	SINTARR[0]{4}	SINTTAG.0	j

INT、UINT 和 WORD

标记	标准	数组元素	无偏移数组	位	字符串
数据类型 高级	字、短整型	字、短整型	字数组、短整型 Array (INT 1/2/3 维数组)	布尔型 (INT 中的位)	
示例	INTTAG	INTARR[0]	INTARR[0]{4}	INTTAG.0	

DINT、UDINT 和 DWORD

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型 高级	双字型、长整型	双字型、长整型	双字型数组、长整型数组	布尔型 (DINT 中的位)	高级
示例	DINTTAG	DINTARR[0]	DINTARR[0]{4}	DINTTAG.0	

LINT、ULINT 和 LWORD

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型 高级	双精度、日期	双精度、日期	双精度数组		
示例	LINTTAG	LINTARR[0]	LINTARR[0]{4}		

REAL

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型 高级	浮点型	浮点型	浮点型数组		
示例	REALTAG	REALARR[0]	REALARR[0]{4}		

LREAL

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型 高级	双精度	双精度	双精度数组		
示例	LREALTAG	LREALARR[0]	LREALARR[0]{4}		

SHORT_STRING

标记	标准	数组元素	带/无偏移数组	位	字符串
数据类型 高级	字符串	字符串			
示例	STRINGTAG	STRINGARR[0]			

另请参阅：地址格式

寻址 结构化数据类型

无法在结构级别参考结构：只能寻址原子型结构成员。有关详细信息，请参阅以下示例。

原生标记
MyTimer @ TIMER

有效客户端/服务器标记
地址 = MyTimer.ACC
数据类型 = 双字型

无效客户端/服务器标记
地址 = MyTimer
数据类型 = ??

寻址字符串数据类型

“字符串”是一种预定义的“原生数据类型”，其结构中包含两个成员：DATA 和 LEN。DATA 是“短整型”数组，可存储字符串的字符。LEN 是一种双整数，表示数据中要显示在客户端的字符数。

由于 LEN 和 DATA 是原子型成员，因此必须独立于客户端/服务器对其进行引用。

说明	语法	示例
字符串值	DATA/<最大字符串长度>	MYSTRING.DATA/82
实际字符串长度	LEN	MYSTRING.LEN

读取

读取自 DATA 的字符串将在以下情况下终止：

a. 第一个空终止符出现。

b. LEN 中的值未先出现 (如果 a) 成立)。

c. <最大字符串长度> 未先出现 (如果 a) 或 b) 成立)。

示例
MYSTRING.DATA 在 PLC 中包含 "Hello World"，但 LEN 手动设置为 5。读取 MYSTRING.DATA/82 时将显示 "Hello"。如果将 LEN 设置为 20，则 MYSTRING.DATA/82 将显示 "Hello World"。

写入

将“字符串”值写入数据时，驱动程序也会以所写入数据的长度写入 LEN。如果写入 LEN 的操作由于任何原因而失败，则写入数据的操作也将被视为失败 (尽管数据成功写入控制器)。

注意：此行为专用于字符串类型的原生标记 (或字符串的自定义导数)。以下注意事项适用于要在 UDT 中实施自己的字符串的用户。

- 如果 UDT 存在且其中含有作为字符串引用的 DATA 成员和作为“双整型”引用的 LEN 成员，则无论给定 UDT 的 LEN 意图如何，写入 LEN 的操作均将成功。如果不希望 LEN 与数据的长度相同，则设计 UDT 时必须谨慎，以避免这种可能性。
- 如果 UDT 存在且其中含有作为字符串引用的 DATA 成员，但不含 LEN 成员，则写入 LEN 的操作将静默失败，而不对 DATA 造成任何影响。

示例

MYSTRING.DATA/82 的值为 "Hello World"。MYSTRING.LEN 的值为 11。如果将值 "Alarm Triggered" 写入 MYSTRING.DATA/82，则会将 15 写入 MYSTRING.LEN。如果写入 MYSTRING.LEN 的操作失败，则 MYSTRING.LEN 将保留原有值 11，而 MYSTRING.DATA/82 将显示前 11 个字符 ("Alarm Trigg")。如果写入 MYSTRING.DATA/82 的操作失败，则两个标记均不受影响。

数组数据的排序

一维数组 - 数组 [1 维]

1 维数组数据按升序传递至控制器并从中传出。

适用于 (维度 1 = 0；维度 1 < 维度 1_max；维度 1++)

示例：3 元素数组
数组 [0]
数组 [1]
数组 [2]

二维数组 - 数组 [1 维, 2 维]

2 维数组数据按升序传递至控制器并从中传出。

适用于 (维度 1 = 0；维度 1 < 维度 1_max；维度 1++)
适用于 (维度 2 = 0；维度 2 < 维度 2_max；维度 2++)

示例：3X3 元素数组
数组 [0, 0]
数组 [0, 1]
数组 [0, 2]
数组 [1, 0]
数组 [1, 1]
数组 [1, 2]
数组 [2, 0]
数组 [2, 1]
数组 [2, 2]

三维数组 - 数组 [1 维, 2 维, 3 维]

3 维数组数据按升序传递至控制器并从中传出。

适用于 (维度 1 = 0；维度 1 < 维度 1_max；维度 1++)
适用于 (维度 2 = 0；维度 2 < 维度 2_max；维度 2++)
适用于 (维度 3 = 0；维度 3 < 维度 3_max；维度 3++)

示例：3X3X3 元素数组
数组 [0, 0, 0]
数组 [0, 0, 1]
数组 [0, 0, 2]
数组 [0, 1, 0]
数组 [0, 1, 1]
数组 [0, 1, 2]
数组 [0, 2, 0]
数组 [0, 2, 1]
数组 [0, 2, 2]
数组 [1, 0, 0]
数组 [1, 0, 1]
数组 [1, 0, 2]
数组 [1, 1, 0]
数组 [1, 1, 1]
数组 [1, 1, 2]
数组 [1, 2, 0]
数组 [1, 2, 1]
数组 [1, 2, 2]
数组 [2, 0, 0]
数组 [2, 0, 1]
数组 [2, 0, 2]
数组 [2, 1, 0]

数组 [2, 1, 1]
数组 [2, 1, 2]
数组 [2, 2, 0]
数组 [2, 2, 1]
数组 [2, 2, 2]

高级用例

有关特定原子型数据类型的高级用例的详细信息，请从下表中选择相应链接。

布尔型
SINT、USINT 和 BYTE
整型、无符号整型和字
DINT、UDINT 和 DWORD
长整型、无符号长整型和长字型
实型
长实型
SHORT_STRING

布尔型

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注解
数组元素	布尔型	原生标记必须是 1 维数组。
带偏移数组	布尔型数组	1. 原生标记必须是 1 维数组。 2. 偏移必须位于 32 位边界处。 3. 元素的数目必须是 32 的因子。
无偏移数组	布尔型数组	1. 原生标记必须是 1 维数组。 2. 元素的数目必须是 32 的因子。
位	布尔型	1. 原生标记必须是 1 维数组。 2. 范围限制为 0 至 31。
标准	布尔型、字节、字符、字、短整型、BCD、双字型、长整型、LBCD、浮点型*	无
字符串	不支持。	

*浮点值等于浮点形式 (非 IEEE 浮点数) 的原生标记的面值。

示例

突出显示的示例表示常见用例。

布尔型原子型标记 - booltag = 真

服务器标记地址	格式	数据类型	注解
booltag	标准	布尔型	值 = 真
booltag	标准	字节	值 = 1

服务器标记地址	格式	数据类型	注解
booltag	标准	Word	值 = 1
booltag	标准	双字型	值 = 1
booltag	标准	浮点型	值 = 1.0
booltag [3]	数组元素	布尔型	无效：标记不是数组。
booltag [3]	数组元素	Word	无效：标记不是数组。
booltag {1}	无偏移数组	Word	无效：不受支持。
booltag {1}	无偏移数组	布尔型	无效：不受支持。
booltag [3] {32}	带偏移数组	布尔型	无效：标记不是数组。
booltag .3	位	布尔型	无效：标记不是数组。
booltag / 1	字符串	字符串	无效：不受支持。
booltag / 4	字符串	字符串	无效：不受支持。

布尔型数组标记 - bitarraytag = [0,1,0,1]

服务器标记地址	格式	数据类型	注解
bitarraytag	标准	布尔型	无效：标记不能是数组。
bitarraytag	标准	字节	无效：标记不能是数组。
bitarraytag	标准	Word	无效：标记不能是数组。
bitarraytag	标准	双字型	无效：标记不能是数组。
bitarraytag	标准	浮点型	无效：标记不能是数组。
bitarraytag [3]	数组元素	布尔型	值 = 真
bitarraytag [3]	数组元素	Word	无效：数据类型不正确。
bitarraytag {3}	无偏移数组	Word	无效：标记不能是数组。
bitarraytag {1}	无偏移数组	Word	无效：标记不能是数组。
bitarraytag {1}	无偏移数组	布尔型	无效：数组大小必须为 32 的因子。
bitarraytag {32}	无偏移数组	布尔型	值 = [0,1,0,1,...]
bitarraytag [3] {32}	带偏移数组	布尔型	偏移必须从 32 位边界处开始。
bitarraytag[0]{32}	带偏移数组	布尔型	值 = [0,1,0,1,...]
bitarraytag[32]{64}	带偏移数组	布尔型	语法有效。元素超出范围。
bitarraytag .3	位	布尔型	值 = 真
bitarraytag / 1	字符串	字符串	无效：不受支持。
bitarraytag / 4	字符串	字符串	无效：不受支持。

SINT、USINT 和 BYTE

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注解
数组元素	字节、字符、字、短整型、BCD、双字型、长整型、LBCD、浮点型***	原生标记必须是数组。
带偏移数组	字节数组、字符数组、字数组、短整型数组、BCD 数组**、双字型数组、长整型数组、LBCD 数组**、浮点型数组**	原生标记必须是数组。
无偏	布尔型数组	1. 使用此例，以数组形式获得 SINT 中的位。这不是使用布尔型符号的“短整型”数

格式	支持的数据类型	注解
移数组	字节数组、字符数组、字数组、短整型数组、BCD 数组**、双字型数组、长整型数组、LBCD 数组**、浮点型数组**	组。 2. 仅应用于 SINT 中的位。示例：tag_1.0 {8}。 3. .bit 与数组大小的总和不能超过 8 位。示例：tag_1.1{8} 超出 SINT，tag_1.0{8} 未超出。 如果访问多个元素，则原生标记必须是数组。
位	布尔型	1. 范围限制为 0 至 7。 2. 如果原生标记是数组，则位类引用必须以数组元素类引用为前缀。示例：tag_1 [2,2,3].0。
标准	布尔型*、字节、字符、字、短整型、BCD、双字型、长整型、LBCD、浮点型***	无
字符串	字符串	1. 如果访问单个元素，则原生标记无需是数组。 **注意**：字符串的值是 SINT 值的 ASCII 对等值。示例：SINT = 65 (十进制) = "A"。 2. 如果访问多个元素，则原生标记必须是数组。字符串的值是字符串中所有“短整数”的 ASCII 对等值 (以空值终止)。 字符串中的 1 个字符 = 1 SINT。

*非零值限制为“真”。
**数组的每个元素与 SINT 数组中的一个元素对应。不封装数组。
***浮点值等于浮点形式 (非 IEEE 浮点数) 的原生标记的面值。

示例

突出显示的示例表示“短整型”、“无符号短整型”和“字节”的常见用例。

SINT、USINT 和 BYTE 原子型标记 - sinttag = 122 (十进制)

服务器标记地址	格式	数据类型	注释
sinttag	标准	布尔型	值 = 真
sinttag	标准	字节	值 = 122
sinttag	标准	字	值 = 122
sinttag	标准	双字型	值 = 122
sinttag	标准	浮点型	值 = 122.0
sinttag [3]	数组元素	布尔型	无效：标记不是数组。此外，布尔型无效。
sinttag [3]	数组元素	字节	无效：标记不是数组。
sinttag {3}	无偏移数组	字节	无效：标记不是数组。
sinttag {1}	无偏移数组	字节	值 = [122]
sinttag {1}	无偏移数组	布尔型	无效：数据类型不正确。
sinttag [3] {1}	带偏移数组	字节	无效：标记不是数组。
sinttag . 3	位	布尔型	值 = 真
sinttag . 0 {8}	无偏移数组	布尔型	值 = [0,1,0,1,1,1,1,0]

服务器标记地址	格式	数据类型	注释
			位值为 122
sinttag / 1	字符串	字符串	无效：语法/数据类型不受支持。
sinttag / 4	字符串	字符串	无效：语法/数据类型不受支持。

“短整型”、“无符号短整型”和“字节”数组标记 - sintarraytag [4,4] = [[83,73,78,84],[5,6,7,8],[9,10,11,12],[13,14,15,16]]

服务器标记地址	格式	数据类型	注释
sintarraytag	标准	布尔型	无效：标记不能是数组。
sintarraytag	标准	字节	无效：标记不能是数组。
sintarraytag	标准	Word	无效：标记不能是数组。
sintarraytag	标准	双字型	无效：标记不能是数组。
sintarraytag	标准	浮点型	无效：标记不能是数组。
sintarraytag [3]	数组元素	字节	无效：服务器标记缺少 2 维地址。
sintarraytag [1,3]	数组元素	布尔型	无效：数组元素不允许使用布尔型。
sintarraytag [1,3]	数组元素	字节	值 = 8
sintarraytag {10}	无偏移数组	字节	值 = [83,73,78,84,5,6,7,8,9,10]
sintarraytag {2} {5}	无偏移数组	Word	值 = [83,73,78,84,5] [6,7,8,9,10]
sintarraytag {1}	无偏移数组	字节	值 = 83
sintarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
sintarraytag [1,3] {4}	带偏移数组	字节	值 = [8,9,10,11]
sintarraytag .3	位	布尔型	无效：标记必须引用原子型位置。
sintarraytag [1,3] . 3	位	布尔型	值 = 1
sintarraytag [1,3] . 0 {8}	无偏移数组	布尔型	值 = [0,0,0,1,0,0,0,0]
sintarraytag / 1	字符串	字符串	无效：语法/数据类型不受支持。
sintarraytag / 4	字符串	字符串	无效：语法/数据类型不受支持。

INT、UINT 和 WORD

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注释
数组元素	字节、字符**、字、短整型、BCD、双字型、长整型、LBCD、浮点型****	原生标记必须是数组。
带偏移数组	字节数组、字符数组**、字数组、短整型数组、BCD 数组、双字型数组、长整型数组、LBCD 数组***、浮点型数组***、****	原生标记必须是数组。
无偏移数组	布尔型数组	1. 使用此例，以数组形式获得 INT 中的位。这不是使用布尔型符号的“整型”数组。 2. 仅应用于 INT 中的位。示例：tag_1.0 {16}。 3. .bit 与数组大小的总和不能超过 16 位。示例：tag_1.1{16} 超出 INT，tag_1.0{16} 未超出。 如果访问多个元素，则原生标记必须是

格式	支持的数据类型	注释
	字节数组、字符数组**、字数组、短整型数组、BCD 数组、双字型数组、长整型数组、LBCD 数组***、浮点型数组***、****	数组。
位	布尔型	1. 范围限制为 0 至 15。 2. 如果原生标记是数组，则位类参考必须以数组元素类参考为前缀。示例：tag_1 [2,2,3].0。
标准	布尔型*、字节、字符**、字、短整型、BCD、双字型、长整型、LBCD、浮点型****	无
字符串	字符串	1. 如果访问单个元素，则控制器标记无需是数组。 **注意**：字符串的值是 INT 值的 ASCII 对等值 (限制到 255)。示例：INT = 65 (十进制) = "A"。 2. 如果访问多个元素，则原生标记必须是数组。字符串的值是字符串中所有整数的 ASCII 对等值 (以空值终止，限制到 255)。 字符串中的 1 个字符 = 1 INT (限制到 255)。 不压缩“整型”字符串。为提高效率，请改用 SINT 字符串或 STRING 结构。

*非零值限制为“真”。
**超过 255 的值限制到 255。
***数组的每个元素均与“整型”数组中的一个元素对应。不封装数组。
****浮点值等于浮点形式 (非 IEEE 浮点数) 的原生标记的面值。

示例

突出显示的示例表示 INT、UINT 和 WORD 的常见用例。

INT、UINT 和 WORD 原子型标记 - inttag = 65534 (十进制)

服务器标记地址	类	数据类型	注释
inttag	标准	布尔型	值 = 真
inttag	标准	字节	值 = 255
inttag	标准	字	值 = 65534
inttag	标准	双字型	值 = 65534
inttag	标准	浮点型	值 = 65534.0
inttag [3]	数组元素	布尔型	无效：标记不是数组。此外，布尔型无效。
inttag [3]	数组元素	Word	无效：标记不是数组。
inttag {3}	无偏移数组	Word	无效：标记不是数组。
inttag {1}	无偏移数组	Word	值 = [65534]
inttag {1}	无偏移数组	布尔型	无效：数据类型不正确。
inttag [3] {1}	带偏移数组	Word	无效：标记不是数组。
inttag . 3	位	布尔型	值 = 真
inttag . 0 {16}	无偏移数组	布尔型	值 = [0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1] 位值为 65534

服务器标记地址	类	数据类型	注释
inttag / 1	字符串	字符串	无效：语法/数据类型不受支持。
inttag / 4	字符串	字符串	无效：语法/数据类型不受支持。

INT、UINT 和 WORD 数组标记 - intarraytag [4,4] = [[73,78,84,255],[256,257,258,259],[9,10,11,12],[13,14,15,16]]

服务器标记地址	类	数据类型	注释
intarraytag	标准	布尔型	无效：标记不能是数组。
intarraytag	标准	字节	无效：标记不能是数组。
intarraytag	标准	Word	无效：标记不能是数组。
intarraytag	标准	双字型	无效：标记不能是数组。
intarraytag	标准	浮点型	无效：标记不能是数组。
intarraytag [3]	数组元素	Word	无效：服务器标记缺少 2 维地址。
intarraytag [1,3]	数组元素	布尔型	无效：数组元素不允许使用布尔型。
intarraytag [1,3]	数组元素	Word	值 = 259
intarraytag {10}	无偏移数组	字节	值 = [73,78,84,255,255,255,255,255,9,10]
intarraytag {2} {5}	无偏移数组	Word	值 = [73,78,84,255,256] [257,258,259,9,10]
intarraytag {1}	无偏移数组	Word	值 = 73
intarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
intarraytag [1,3] {4}	带偏移数组	字	值 = [259,9,10,11]
intarraytag .3	位	布尔型	无效：标记必须引用原子型位置。
intarraytag [1,3] . 3	位	布尔型	值 = 0
intarraytag [1,3] . 0 {16}	无偏移数组	布尔型	值 = [1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0] 位值为 259
intarraytag / 1	字符串	字符串	无效：语法/数据类型不受支持。
intarraytag / 3	字符串	字符串	无效：语法/数据类型不受支持。

DINT、UDINT 和 DWORD

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注解
数组元素	字节、字符**、字、短整型、BCD***、双字型、长整型、LBCD、浮点型****	原生标记必须是数组。
带偏移数组	字节数组、字符数组**、字数组、短整型数组、BCD 数组***、双字型数组、长整型数组、LBCD 数组、浮点型数组****	原生标记必须是数组。
无偏移数组	布尔型数组	1. 使用此例，以数组形式获得 DINT 中的位。这不是使用布尔型符号的“双整型”数组。 2. 仅应用于 DINT 中的位。示例：tag_1.0 {32}。 3. .bit 与数组大小的总和不能超过 32 位。示例：tag_1.1{32} 超出 DINT，tag_1.0{32} 未超出。 如果访问多个元素，则原生标记必须是数

格式	支持的数据类型	注解
	字节数组、字符数组**、字数组、短整型数组、BCD 数组***、双字型数组、长整型数组、LBCD 数组、浮点型数组****	组。
位	布尔型	1. 范围限制为 0 至 31。 2. 如果原生标记是数组，则位类参考必须以数组元素类参考为前缀。示例：tag_1 [2,2,3].0。
标准	布尔型* 字节、字符** 字、短整型、BCD*** 双字型、长整型、LBCD 浮点型****	无
字符串	字符串	1. 如果访问单个元素，则控制器标记无需是数组。 **注意**：字符串的值是 DINT 值的 ASCII 对等值 (限制到 255)。示例：SINT = 65 (十进制) = "A"。 2. 如果访问多个元素，则原生标记必须是数组。字符串的值是字符串中所有双整数的 ASCII 对等值 (以空值终止，限制到 255)。 字符串中的 1 个字符 = 1 DINT (限制到 255)。 不封装双整型字符串。为提高效率，请改用 SINT 字符串或 STRING 结构。

*非零值限制为“真”。
**超过 255 的值限制到 255。
***超过 65535 的值限制到 65535。
****浮点值等于浮点形式 (非 IEEE 浮点数) 的原生标记的面值。

示例

突出显示的示例

DINT、UDINT 和 DWORD 原子型标记 - dinttag = 70000 (十进制)

服务器标记地址	格式	数据类型	注解
dinttag	标准	布尔型	值 = 真
dinttag	标准	字节	值 = 255
dinttag	标准	Word	值 = 65535
dinttag	标准	双字型	值 = 70000
dinttag	标准	浮点型	值 = 70000.0
dinttag [3]	数组元素	布尔型	无效：标记不是数组。此外，布尔型无效。
dinttag [3]	数组元素	双字型	无效：标记不是数组。
dinttag {3}	无偏移数组	双字型	无效：标记不是数组。
dinttag {1}	无偏移数组	双字型	值 = [70000]
dinttag {1}	无偏移数组	布尔型	无效：数据类型不正确
dintag [3] {1}	带偏移数组	双字型	无效：标记不是数组。
dinttag . 3	位	布尔型	值 = 假
dinttag . 0 {32}	无偏移数组	布尔型	值 = [0,0,0,0,1,1,1,0,1,0,0,0,1,0,0,0,1,0,...0]

服务器标记地址	格式	数据类型	注解
			位值为 70000
dinttag	字符串	字符串	无效：语法/数据类型不受支持。
dinttag	字符串	字符串	无效：语法/数据类型不受支持。

DINT、UDINT 和 DWORD 数组标记 - dintarraytag [4,4] = [[68,73,78,84],[256,257,258,259],[9,10,11,12],[13,14,15,16]]

服务器标记地址	格式	数据类型	注解
dintarraytag	标准	布尔型	无效：标记不能是数组。
dintarraytag	标准	字节	无效：标记不能是数组。
dintarraytag	标准	Word	无效：标记不能是数组。
dintarraytag	标准	双字型	无效：标记不能是数组。
dintarraytag	标准	浮点型	无效：标记不能是数组。
dintarraytag [3]	数组元素	双字型	无效：服务器标记缺少 2 维地址。
dintarraytag [1,3]	数组元素	布尔型	无效：数组元素不允许使用布尔型。
dintarraytag [1,3]	数组元素	双字型	值 = 259
dintarraytag {10}	无偏移数组	字节	值 = [68,73,78,84,255,255,255,255,9,10]
dintarraytag {2}{5}	无偏移数组	双字型	值 = [68,73,78,84,256] [257,258,259,9,10]
dintarraytag {1}	无偏移数组	双字型	值 = 68
dintarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
dintarraytag [1,3]{4}	带偏移数组	双字型	值 = [259,9,10,11]
dintarraytag .3	位	布尔型	无效：标记必须引用原子型位置。
dintarraytag [1,3] .3	位	布尔型	值 = 0
dintarraytag [1,3] .0 {32}	无偏移数组	布尔型	值 = [1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0] 位值为 259
dintarraytag	字符串	字符串	无效：语法/数据类型不受支持。
dintarraytag	字符串	字符串	无效：语法/数据类型不受支持。

LINT、ULINT 和 LWORD

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注解
数组元素	双精度* 日期**	原生标记必须是数组。
带偏移数组	双精度数组*	原生标记必须是数组。
无偏移数组	双精度数组*	如果访问多个元素，则控制器标记必须是数组。
位	不支持。	不支持。
标准	双精度* 日期**	无
字符串	不支持。	不支持。

*双精度值等于浮点形式 (非 IEEE 浮点数) 的控制器标记的面值。
**日期值采用世界协调时间 (UTC)，而不是本地时间。

示例

突出显示的示例表示“长整型”、“无符号长整型”和“长字型”的常见用例。

“长整型”、“无符号长整型”和“长字型”原子型标记 - linttag = 2007-01-01T16:46:40.000 (日期) == 1.16767E+15 (十进制)

服务器标记地址	格式	数据类型	注解
linttag	标准	布尔型	无效：布尔型不受支持。
linttag	标准	字节	无效：字节不受支持。
linttag	标准	Word	无效：字不受支持。
linttag	标准	双精度	值 = 1.16767E+15
linttag	标准	日期	值 = 2007-01-01T16:46:40.000*
linttag [3]	数组元素	布尔型	无效：标记不是数组。此外，布尔型无效。
linttag [3]	数组元素	双精度	无效：标记不是数组。
linttag {3}	无偏移数组	双精度	无效：标记不是数组。
linttag {1}	无偏移数组	双精度	值 = [1.16767E+15]
linttag {1}	无偏移数组	布尔型	无效：数据类型不正确。
lintag [3] {1}	带偏移数组	双精度	无效：标记不是数组。
linttag .3	位	布尔型	无效：语法/数据类型不受支持。
linttag / 1	字符串	字符串	无效：语法/数据类型不受支持。

*日期值采用世界协调时间 (UTC)，而不是本地时间。

“长整型”、“无符号长整型”和“长字型”数组标记 -
dintarraytag [2,2] = [0, 1.16767E+15],[9.4666E+14, 9.46746E+14] 其中：
1.16767E+15 == 2007-01-01T16:46:40.000 (日期)
9.4666E+14 == 1999-12-31T17:06:40.000
9.46746E+14 == 2000-01-1T17:00:00.000
0 == 1970-01-01T00:00:00.000

服务器标记地址	格式	数据类型	注解
lintarraytag	标准	布尔型	无效：布尔型不受支持。
lintarraytag	标准	字节	无效：字节类型不受支持。
lintarraytag	标准	字	无效：字类型不受支持。
lintarraytag	标准	双精度	无效：标记不能是数组。
lintarraytag	标准	日期	无效：标记不能是数组。
lintarraytag [1]	数组元素	双精度	无效：服务器标记缺少 2 维地址。
lintarraytag [1,1]	数组元素	布尔型	无效：数组元素不允许使用布尔型。
lintarraytag [1,1]	数组元素	双精度	值 = 9.46746E+14
lintarraytag [1,1]	数组元素	日期	值 = 2000-01-01T17:00:00.000*
lintarraytag {4}	无偏移数组	双精度	值 = [0, 1.16767E+15, 9.4666E+14, 9.46746E+14]
lintarraytag {2} {2}	无偏移数组	双精度	值 = [0, 1.16767E+15][9.4666E+14, 9.46746E+14]
lintarraytag {4}	无偏移数组	日期	无效：日期数组不受支持。
lintarraytag {1}	无偏移数组	双精度	值 = 0
lintarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
lintarraytag [0,1] {2}	带偏移数组	双精度	值 = [1.16767E+15, 9.4666E+14]
lintarraytag .3	位	布尔型	无效：语法/数据类型不受支持。
lintarraytag / 1	字符串	字符串	无效：语法/数据类型不受支持。

*日期值采用世界协调时间 (UTC)，而不是本地时间。

REAL

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注解
数组元素	字节、字符**、字、短整型、BCD***、双字型、长整型、LBCD、浮点型****	原生标记必须是数组。
带偏移数组	字节数组、字符数组**、字数组、短整型数组、BCD 数组***、双字型数组、长整型数组、LBCD 数组、浮点型数组****	原生标记必须是数组。
无偏移数组	布尔型数组 字节数组、字符数组**、字数组、短整型数组、BCD 数组***、双字型数组、长整型数组、LBCD 数组、浮点型数组****	1. 使用此例，以数组形式获得 REAL 中的位。这不是使用布尔型符号的 REAL 数组。 2. 仅应用于 REAL 中的位。示例：tag_1.0{32}。 3. .bit 与数组大小的总和不能超过 32 位。示例：tag_1.1{32} 超出 REAL，tag_1.0{32} 未超出。 如果访问多个元素，则原生标记必须是数组。
位	布尔型	1. 范围限制为 0 至 31。 2. 如果原生标记是数组，则位类引用必须以数组元素类引用为前缀。示例：tag_1[2,2,3].0。 **注意：**浮点型将投射到双字型，以允许位参考。
标准	布尔型* 字节、字符** 字、短整型、BCD*** 双字型、长整型、LBCD 浮点型****	无
字符串	字符串	1. 如果访问单个元素，则控制器标记无需是数组。 **注意：**字符串的值是 REAL 值的 ASCII 对等值 (限制到 255)。示例：SINT = 65dec = "A". 2. 如果访问多个元素，则原生标记必须是数组。字符串的值是字符串中所有 REAL 的 ASCII 对等值 (以空值终止，限制到 255)。 字符串中的 1 个字符 = 1 个实数 (限制到 255)。 不压缩 REAL 字符串。为提高效率，请改用 SINT 字符串或 STRING 结构。

*非零值限制为“真”。
**超过 255 的值限制到 255。
***超过 65535 的值限制到 65535。
****浮点值是有效的 IEEE 单精度浮点数。

示例

突出显示的示例表示常见用例。

REAL 原子型标记 - realtag = 512.5 (十进制)

服务器标记地址	格式	数据类型	注解
realtag	标准	布尔型	值 = 真
realtag	标准	字节	值 = 255
realtag	标准	字	值 = 512
realtag	标准	双字型	值 = 512
realtag	标准	浮点型	值 = 512.5
realtag [3]	数组元素	布尔型	无效：标记不是数组。此外，布尔型无效。
realtag [3]	数组元素	双字型	无效：标记不是数组。
realtag {3}	无偏移数组	双字型	无效：标记不是数组。
realtag {1}	无偏移数组	浮点型	值 = [512.5]
realtag {1}	无偏移数组	布尔型	无效：数据类型不正确。
realtag [3] {1}	带偏移数组	浮点数	无效：标记不是数组。
realtag . 3	位	布尔型	值 = 真
realtag . 0 {32}	无偏移数组	布尔型	值 = [0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,...0] 位值为 512
realtag	字符串	字符串	无效：语法/数据类型不受支持。
realtag	字符串	字符串	无效：语法/数据类型不受支持。

REAL 数组标记 - realarraytag [4,4] = [[82.1,69.2,65.3,76.4],[256.5,257.6,258.7,259.8], [9.0,10.0,11.0,12.0],[13.0,14.0,15.0,16.0]]

服务器标记地址	格式	数据类型	注解
realarraytag	标准	布尔型	无效：标记不能是数组。
realarraytag	标准	字节	无效：标记不能是数组。
realarraytag	标准	Word	无效：标记不能是数组。
realarraytag	标准	双字型	无效：标记不能是数组。
realarraytag	标准	浮点型	无效：标记不能是数组。
realarraytag [3]	数组元素	浮点型	无效：服务器标记缺少 2 维地址。
realarraytag [1,3]	数组元素	布尔型	无效：数组元素不允许使用布尔型。
realarraytag [1,3]	数组元素	浮点型	值 = 259.8
realarraytag {10}	无偏移数组	字节	值 = [82,69,65,76,255,255,255,255,9,10]
realarraytag {2} {5}	无偏移数组	浮点型	值 = [82.1,69.2,65.3,76.4,256.5] [257.6,258.7,259.8,9,10]
realarraytag {1}	无偏移数组	浮点型	值 = 82.1
realarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
realarraytag [1,3] {4}	带偏移数组	浮点数	值 = [259,8,9,0,10.0,11,0]
realarraytag .3	位	布尔型	无效：标记必须引用原子型位置。
realarraytag [1,3] . 3	位	布尔型	值 = 0
realarraytag [1,3] . 0 {32}	无偏移数组	布尔型	值 = [1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0] 位值为 259
realarraytag	字符串	字符串	无效：语法/数据类型不受支持。

服务器标记地址	格式	数据类型	注解
realarraytag	字符串	字符串	无效：语法/数据类型不受支持。

LREAL

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注释
数组元素	双精度*	原生标记必须是数组。
带偏移数组	双精度数组	原生标记必须是数组。
无偏移数组	双精度数组	如果访问多个元素，则原生标记必须是数组。
位	布尔型	无效：语法/数据类型不受支持。
标准	双精度*	无
字符串	字符串	无效：语法/数据类型不受支持。

*双精度值是有效的 IEEE 双精度浮点数。

示例

突出显示的示例表示常见用例。

“长实型”原子型标记 - lrealtag = 512.5 (十进制)

服务器标记地址	格式	数据类型	注释
lrealtag	标准	布尔型	无效：数据类型不受支持。
lrealtag	标准	字节	无效：数据类型不受支持。
lrealtag	标准	字	无效：数据类型不受支持。
lrealtag	标准	双字型	无效：数据类型不受支持。
lrealtag	标准	双精度	值 = 512.5
lrealtag [3]	数组元素	布尔型	无效：标记不是数组，且布尔型无效。
lrealtag [3]	数组元素	双字型	无效：标记不是数组。
lrealtag {3}	无偏移数组	双字型	无效：标记不是数组。
lrealtag {1}	无偏移数组	双精度	值 = [512.5]
lrealtag {1}	无偏移数组	布尔型	无效：数据类型不正确。
lrealtag [3] {1}	带偏移数组	浮点数	无效：标记不是数组。
lrealtag . 3	位	布尔型	无效：数据类型不受支持。
lrealtag . 0 {32}	无偏移数组	布尔型	无效：数据类型不受支持。
lrealtag	字符串	字符串	无效：语法/数据类型不受支持。
lrealtag	字符串	字符串	无效：语法/数据类型不受支持。

“长实型”数组标记 - realarraytag [4,4] = [[82.1,69.2,65.3,76.4],[256.5,257.6,258.7,259.8], [9.0,10.0,11.0,12.0],[13.0,14.0,15.0,16.0]]

服务器标记地址	格式	数据类型	注释
lrealarraytag	标准	布尔型	无效：标记不能是数组。
lrealarraytag	标准	字节	无效：标记不能是数组。
lrealarraytag	标准	Word	无效：标记不能是数组。
lrealarraytag	标准	双字型	无效：标记不能是数组。
lrealarraytag	标准	双精度	无效：标记不能是数组。
lrealarraytag [3]	数组元素	双精度	无效：服务器标记缺少 2 维地址。
lrealarraytag [1,3]	数组元素	布尔型	无效：数组元素不允许使用布尔型。

服务器标记地址	格式	数据类型	注释
lrealarraytag [1,3]	数组元素	双精度	值 = 259.8
lrealarraytag {10}	无偏移数组	字节	无效：数据类型不受支持。
lrealarraytag {2} {5}	无偏移数组	双精度	值 = [82.1,69.2,65.3,76.4,256.5] [257.6,258.7,259.8,9,10]
lrealarraytag {1}	无偏移数组	双精度	值 = 82.1
lrealarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
lrealarraytag [1,3] {4}	带偏移数组	双精度	值 = [259,8,9,0,10.0,11,0]
lrealarraytag . 3	位	布尔型	无效：标记必须引用原子型位置。
lrealarraytag [1,3] . 3	位	布尔型	值 = 0
lrealarraytag [1,3] . 0 {32}	无偏移数组	布尔型	无效：语法/数据类型不受支持。
lrealarraytag	字符串	字符串	无效：语法/数据类型不受支持。
lrealarraytag	字符串	字符串	无效：语法/数据类型不受支持。

SHORT_STRING

有关格式的详细信息，请参阅地址格式。

格式	支持的数据类型	注释
数组元素	字符串	原生标记必须是数组。
带偏移数组	不适用	不适用
无偏移数组	不适用	不适用
位	不适用	不适用
标准	字符串	字符串的长度取决于原生标记中包含的长度编码。如果字符串中包含不可打印字符，则这些字符包括在字符串中。
字符串	不适用	需要在标记地址中指定字符串长度。

示例

突出显示的示例表示常见用例。

SHORT_STRING 原子型标记 - stringtag = "mystring"

服务器标记地址	格式	数据类型	注释
stringtag	标准	字符串	值 = mystring。
stringtag	标准	字节	无效：字节不受支持。
stringtag	标准	Word	无效：字不受支持。
stringtag [3]	数组元素	布尔型	无效：标记不是数组，且布尔型无效。
stringtag [3]	数组元素	双精度	无效：标记不是数组。
stringtag {3}	无偏移数组	双精度	无效：标记不是数组。
stringtag {1}	无偏移数组	双精度	值 = [1.16767E+15]。
stringtag {1}	无偏移数组	布尔型	无效：数据类型不正确。
lintag [3] {1}	带偏移数组	双精度	无效：标记不是数组。

服务器标记地址	格式	数据类型	注释
stringtag . 3	位	布尔型	无效：语法/数据类型不受支持。
stringtag / 1	字符串	字符串	无效：语法/数据类型不受支持。

SHORT_STRING 数组标记 - stringarraytag[2,2] = [1,2].[3,4]

服务器标记地址	格式	数据类型	注释
stringarraytag	标准	布尔型	无效：布尔型不受支持。
stringarraytag	标准	字节	无效：字节类型不受支持。
stringarraytag	标准	Word	无效：字类型不受支持。
stringarraytag	标准	双精度	无效：标记不能是数组。
stringarraytag	标准	日期	无效：标记不能是数组。
stringarraytag [1]	数组元素	双精度	无效：服务器标记缺少 2 维地址。
stringarraytag [1,1]	数组元素	布尔型	无效：数组元素不允许使用布尔型。
stringarraytag [1,1]	数组元素	字符串	值："4"
stringarraytag {4}	无偏移数组	字符串	无效：字符串数组不受支持。
stringarraytag {2} {2}	无偏移数组	字符串	无效：字符串数组不受支持。
stringarraytag {1}	无偏移数组	布尔型	无效：数据类型不正确。
stringarraytag [0, 1] {2}	带偏移数组	字符串	值："3"
stringarraytag . 3	位	布尔型	无效：语法/数据类型不受支持。
stringarraytag / 1	字符串	字符串	无效：语法不受支持。

错误代码

以下部分定义可能在服务器的事件日志中遇到的错误代码。有关特定错误代码类型的详细信息，请从下表中选择一个链接。

封装协议错误代码
CIP 错误代码

封装协议错误代码

以下错误代码为十六进制。

错误代码	说明
0001	命令未处理。
0002	命令内存不可用。
0003	数据不正确或不完整。
0064	会话 ID 无效。
0065	标题长度无效。
0069	请求的协议版本不受支持。
0070	目标 ID 无效。

CIP 错误代码

以下错误代码为十六进制。

错误代码	说明
0001	连接失败*
0002	资源不足
0003	值无效
0004	IOI 无法被解密或标记不存在

错误代码	说明
0005	未知目标
0006	请求的数据不适合响应数据包
0007	失去连接
0008	不支持的服务
0009	数据段错误或属性值无效
000A	属性列表错误
000B	状态已经存在
000C	对象型号冲突
000D	对象已经存在
000E	属性不可配置
000F	权限被拒绝
0010	设备状态冲突
0011	回复不适用
0012	片段原型
0013	为执行服务指定的命令数据/参数不足
0014	不支持的属性
0015	指定的数据过多
001A	桥接请求过大
001B	桥接响应过大
001C	属性列表短缺
001D	属性列表无效
001E	嵌入式服务错误
001F	连接时失败**
0022	收到的回复无效
0025	关键段错误
0026	指定的 IOI 字数与 IOI 字数统计不匹配
0027	列表中存在意外的属性

- *另请参阅：0x0001 扩展错误代码
- **另请参阅：0x001F 扩展错误代码

Allen-Bradley 特定错误代码

错误代码 (十六进制)	说明
00FF	一般错误*

- *另请参阅：0x00FF 扩展错误代码
- 对于未列出的错误代码，请参阅 Rockwell Automation 文档。

0x0001 扩展错误代码

以下错误代码为十六进制。

错误代码	说明
0100	正在使用连接。
0103	不支持传输。
0106	所有权冲突。
0107	未找到连接。
0108	连接类型无效。

错误代码	说明
0109	连接大小无效。
0110	未配置模块。
0111	不支持 EPR。
0114	模块错误。
0115	设备类型错误。
0116	修订版本错误。
0118	配置格式无效。
011A	应用程序超出连接数。
0203	连接超时。
0204	未连接消息超时。
0205	未连接发送参数错误。
0206	消息过大。
0301	无缓冲区内存。
0302	带宽不可用。
0303	无可用的筛选器。
0305	签名匹配。
0311	端口不可用。
0312	链路地址不可用。
0315	段类型无效。
0317	未计划连接。
0318	至自身的链路地址无效。

- *对于未列出的错误代码，请参阅 Rockwell Automation 文档。*

0x001F 扩展错误代码

以下错误代码为十六进制。

错误代码	说明
0203	连接超时。

- *对于未列出的错误代码，请参阅 Rockwell Automation 文档。*

0x00FF 扩展错误代码

以下错误代码为十六进制。

错误代码	说明
2104	地址超出范围。
2105	尝试进行超出数据对象末端的访问。
2106	正在使用数据。
2107	数据类型无效或不受支持。

- *对于未列出的错误代码，请参阅 Rockwell Automation 文档。*

事件日志消息

以下信息涉及发布到主要用户界面中“事件日志”窗格的消息。。请参阅有关筛选和排序“事件日志”详细信息视图的服务器帮助。服务器帮助包含许多常见的消息，因此也应对其进行搜索。通常，其中会尽可能提供消息的类型 (信息、警告) 和故障排除信息。

控制器不受支持。| 供应商 ID = <供应商>，产品类型 = <类型>，产品代码 = <代码>，产品名称 = '<产品>'。

错误类型：
警告

从设备接收的帧有误。

错误类型：
警告

可能的原因：

1. 数据包未对齐 (由于 PC 和设备之间的连接/连接断开)。
2. 设备电缆连接不良，导致噪声。
3. 接收到的帧大小错误。
4. TNS 不匹配。
5. 从设备返回了无效的响应命令。

可能的解决方案：
驱动器不经干预即可从此错误中恢复，但电缆或设备本身可能存在需要改正的问题。

对标记的写入请求由于帧错误而失败。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：

1. 由于请求服务代码不正确，导致重试次数过多，对指定标记的写入请求失败。
2. 由于接收的字节数目多于或少于预期，导致重试次数过多，对指定标记的写入请求失败。

可能的解决方案：

1. 电缆或设备本身可能出现了问题。
2. 请增加重试次数以为驱动程序从此错误中恢复提供更多机会。

对标记的读取请求由于帧错误而失败。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：

1. 由于请求服务代码不正确，导致重试次数过多，对指定标记的读取请求失败。
2. 由于接收的字节数目多于或少于预期，导致重试次数过多，对指定标记的读取请求失败。

可能的解决方案：

1. 电缆或设备本身可能出现了问题。
2. 请增加重试次数以为驱动程序从此错误中恢复提供更多机会。

块读取请求由于帧错误而失败。| 块开始 = '<地址>'，块大小 = <数字> (元素)。

错误类型：
警告

可能的原因：

1. 由于请求服务代码不正确，导致重试次数过多，对指定标记的读取请求失败。
2. 由于接收的字节数目多于或少于预期，导致重试次数过多，对指定标记的读取请求失败。

可能的解决方案：

1. 电缆或设备本身可能出现了问题。
2. 请增加重试次数以为驱动程序从此错误中恢复提供更多机会。

无法写入设备上的标记。| 标记地址 = '<地址>'，CIP 错误 = <代码>，扩展错误 = <代码>。

错误类型：
警告

可能的原因：
在对指定标记的写入请求期间，设备在数据包的 CIP 部分返回了错误。

可能的解决方案：
解决方案取决于返回的错误代码。请参阅“CIP 和扩展代码定义”。

也可以看看：
CIP 错误代码

无法从设备读取标记。| 标记地址 = '<地址>'，CIP 错误 = <代码>，扩展错误 = <代码>。

错误类型：
警告

可能的原因：
在对指定标记的读取请求期间，设备在数据包的 CIP 部分返回了错误。

可能的解决方案：
解决方案取决于返回的错误代码。请参阅“CIP 和扩展代码定义”。

也可以看看：
CIP 错误代码

无法从设备读取块。| 块开始 = '<地址>'，块大小 = <数字>，CIP 错误 = <代码>，扩展错误 = <代码>。

错误类型：
警告

可能的原因：
在对指定标记的块读取请求期间，设备在数据包的 CIP 部分返回了错误。

可能的解决方案：
解决方案取决于返回的错误代码。请参阅“CIP 和扩展代码定义”。

也可以看看：
CIP 错误代码

无法写入设备上的标记。控制器标记数据类型未知。| 标记地址 = '<地址>'，未知数据类型 = <类型>。

错误类型：
警告

可能的原因：
指定标记的写入请求失败，因为“原生标记”数据类型当前不受支持。

可能的解决方案：
联系技术支持人员以请求为此类型添加支持。

无法从设备读取标记。控制器标记数据类型未知。标记已取消激活。| 标记地址 = '<地址>'，未知数据类型 = <类型>。

错误类型：
警告

可能的原因：
指定标记的写入请求失败，因为“原生标记”数据类型当前不受支持。

可能的解决方案：
联系技术支持人员以请求为此类型添加支持。

无法从设备读取块。控制器标记数据类型未知。块已取消激活。| 块开始 = '<地址>'，块大小 = <数字>，未知数据类型 = <类型>。

错误类型：
警告

可能的原因：
指定标记的写入请求失败，因为“原生标记”数据类型当前不受支持。

可能的解决方案：
联系技术支持人员以请求为此类型添加支持。

无法写入设备上的标记。不支持数据类型。| 标记地址 = '<地址>'，不支持的数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持客户端标记数据类型，对指定标记的写入请求失败。

可能的解决方案：
将标记数据类型更改为支持的类型。为响应此错误，该标记将被取消激活，并且不会再次对其进行处理。

也可以看看：

寻址原子型数据类型

无法从设备读取标记。不支持数据类型。| 标记地址 = '<地址>', 不支持的数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持客户端标记数据类型，对指定标记的读取请求失败。

可能的解决方案：
将标记数据类型更改为支持的类型。为响应此错误，该标记将被取消激活，并且不会再次对其进行处理。

也可以看看：
寻址原子型数据类型

无法从设备读取块。不支持数据类型。块已取消激活。| 块开始 = '<地址>', 块大小 = <数字> (元素), 不支持的数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持客户端标记数据类型，对指定标记的读取请求失败。

可能的解决方案：
将标记数据类型更改为支持的类型。为响应此错误，该标记将被取消激活，并且不会再次对其进行处理。

也可以看看：
寻址原子型数据类型

无法写入标记。标记数据类型非法。| 标记地址 = '<地址>', 非法数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持标记数据类型，对指定标记的请求失败。

可能的解决方案：
将标记数据类型更改为支持的类型。例如，对于布尔型数组的“原生标记”，数据类型“短整型”是非法的。将数据类型更改为“布尔型”可以纠正该问题。

也可以看看：
寻址原子型数据类型

无法从设备读取标记。此标记数据类型非法。标记已取消激活。| 标记地址 = '<地址>', 非法数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持标记数据类型，对指定标记的请求失败。

可能的解决方案：

将标记数据类型更改为支持的类型。例如，对于布尔型数组的“原生标记”，数据类型“短整型”是非法的。将数据类型更改为“布尔型”可以纠正该问题。

也可以看看：
寻址原子型数据类型

无法从设备读取块。此块数据类型非法。块已取消激活。| 块开始 = '<地址>'，块大小 = <数字> (元素)，非法数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持标记数据类型，对指定标记的请求失败。

可能的解决方案：
将标记数据类型更改为支持的类型。例如，对于布尔型数组的“原生标记”，数据类型“短整型”是非法的。将数据类型更改为“布尔型”可以纠正该问题。

也可以看看：
寻址原子型数据类型

无法写入设备上的标记。标记不支持多元素数组。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：
由于驱动器不支持对指定原生标记的多元素数组访问，对指定标记的读取请求失败。

可能的解决方案：
将标记数据类型或地址更改为支持的类型。

也可以看看：
寻址原子型数据类型

无法从设备读取标记。标记不支持多元素数组。标记已取消激活。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：
由于驱动器不支持对指定原生标记的多元素数组访问，对指定标记的读取请求失败。

可能的解决方案：
将标记数据类型或地址更改为支持的类型。为响应此错误，标记将取消激活，并不会再次处理。

也可以看看：
寻址原子型数据类型

无法从设备读取块。块不支持多元素数组。块已取消激活。| 块开始 = '<地址>'，块大小 = <数字> (元素)。

错误类型：
警告

可能的原因：
由于驱动器不支持对指定原生标记的多元素数组访问，对指定标记的读取请求失败。

可能的解决方案：
将标记数据类型或地址更改为支持的类型或地址。为响应此错误，该块将被取消激活，并且不会再次对其进行处理。

也可以看看：
寻址原子型数据类型

无法写入设备上的标记。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：

1. 设备与主机 PC 之间的连接断开。
2. 连接的通信参数错误。
3. 可能为指定设备分配了不正确的 IP 地址。

可能的解决方案：

1. 验证 PC 和设备之间的电缆连接。
2. 验证是否已为指定设备指定正确端口。
3. 验证分配给指定设备的地址是否与实际设备的地址相符。

注意：
为响应此错误，标记将取消激活，并不会再次处理。

无法从设备读取标记。标记已取消激活。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：

1. 设备与主机 PC 之间的连接断开。
2. 连接的通信参数错误。
3. 可能为指定设备分配了不正确的 IP 地址。

可能的解决方案：

1. 验证 PC 和设备之间的电缆连接。
2. 验证是否已为指定设备指定正确端口。
3. 验证分配给指定设备的地址是否与实际设备的地址相符。

注意：
为响应此错误，标记将取消激活，并不会再次处理。

无法从设备读取块。块已取消激活。| 块开始 = '<地址>'，块大小 = <数字>。

错误类型：
警告

可能的原因：

1. 设备与主机 PC 之间的连接断开。
2. 连接的通信参数错误。
3. 可能为指定设备分配了不正确的 IP 地址。

可能的解决方案：

1. 验证 PC 和设备之间的电缆连接。
2. 验证是否已为指定设备指定正确端口。
3. 验证分配给指定设备的地址是否与实际设备的地址相符。

注意：
为响应此错误，块将取消激活，并不会再次处理。

设备响应 CIP 错误。| 状态代码 = <代码>，扩展状态代码 = <代码>。

错误类型：
警告

可能的原因：
在请求期间，设备在数据包的 CIP 部分返回了错误。请求中的所有读取和写入失败。

可能的解决方案：
解决方案取决于返回的错误代码。请参阅“CIP 代码”。

无法为标记分配内存。| 标记地址 = '<地址>'。

错误类型：
警告

可能的原因：
无法分配构建标记所需的资源。标记未添加到项目中。

可能的解决方案：
关闭任何未使用的应用程序和/或增加虚拟内存量，然后再试一次。

设备响应 DF1 错误。

错误类型：
警告

可能的原因：
服务器发送的响应无效。

可能的解决方案：

1. 驱动程序尝试从此错误中恢复。
2. 解决方案取决于返回的错误代码。

也可以看看：
错误矩阵

无法从设备读取标记。内存无效。| 标记地址 = '<地址>'。

错误类型：

警告

无法从设备读取标记。标记数据类型非法。| 标记地址 = '<地址>'，非法数据类型 = <类型>。

错误类型：
警告

可能的原因：
由于不支持标记数据类型，对指定标记的请求失败。

可能的解决方案：

1. 验证或更正请求的数据类型。
2. 将标记数据类型更改为支持的类型。例如，对于布尔型数组的“原生标记”，数据类型“短整型”是非法的。将数据类型更改为“布尔型”可以纠正该问题。

也可以看看：
寻址原子型数据类型

无法从设备读取标记。内存无效。标记已取消激活。| 标记地址 = '<地址>'。

错误类型：
警告

无法从设备读取块。内存无效。块已取消激活。| 块开始 = '<地址>'，块大小 = <数字> (元素)。

错误类型：
警告

无法在写入设备上的地址。内存无效。| 标记地址 = '<地址>'。

错误类型：
警告

无法从设备读取块。块已取消激活。| 块开始 = '<地址>'，块大小 = <数字>，CIP 错误 = <代码>，扩展错误 = <代码>。

错误类型：
警告

可能的原因：

1. 设备与主机 PC 之间的连接断开。
2. 连接的通信参数错误。

可能的解决方案：

1. 验证 PC 和设备之间的电缆连接。
2. 验证是否已为指定设备指定正确端口。
3. 验证分配给指定设备的地址是否与实际设备的地址相符。

注意：
为响应此错误，块元素将被取消激活，并且不会再次对其进行处理。

也可以看看：
CIP 错误代码

设备标识详细信息。| ID = <ID>，供应商 ID = <供应商>，产品类型 = <类型>，产品代码 = <代码>，修订版本 = '<修订版本>'，产品名称 = '<产品>'，产品 S/N = <编号>。

错误类型：
信息化

设备不支持分段读/写服务。正在自动回退到非分段服务。

错误类型：
信息化

术语表

基于原生标记的寻址

术语	定义
数组元素	原生数组标记中的元素。对于客户端/服务器访问，元素必须是原子。例如，ARRAYTAG [0]。
带偏移数组	地址具有指定原生数组元素的客户端/服务器数组标记。例如，ARRAYTAG [0] {5}。
无偏移数组	地址无指定原生数组元素的客户端/服务器数组标记。例如，ARRAYTAG {5}。
原子型数据类型	预定义的非结构化原生数据类型。例如，SINT、DINT。
原子型标记	用原子型数据类型定义的原生标记。
客户端	使用 OPC、DDE 或专有客户端/服务器协议与服务器连接的 HMI/SCADA 或数据桥接软件包。
客户端/服务器数据类型	在服务器中静态定义或在客户端中动态定义的标记的数据类型。客户端中支持的数据类型取决于正在使用的客户端。*
客户端/服务器标记	在服务器中静态定义或在客户端中动态定义的标记。这些标记与原生标记不同。在引用此类原生标记时，原生标记名称将变为客户端/服务器标记地址。
客户端/服务器数组	行 x 列数据表示格式受服务器和某些客户端支持。并非所有客户端都支持数组。
CCW	Connected Components Workbench。
原生数据类型	在 Micro800 控制器的 CCW 中定义的数据类型。
原生标记	在 Micro800 控制器的 CCW 中定义的标记。
原生数组数据类型	Micro800 控制器的 CCW 中支持的多维数组 (可以为 1、2 或 3 维)。所有原子型数据类型都支持原生数组。并不是所有结构化数据类型都支持原生数组。
数组标记	用原生数组数据类型定义的原生标记。
预定义数据类型	由 Micro800 控制器的 CCW 支持和预定义的原生数据类型。*
用户定义数据类型	受 CCW 支持并由 Micro800 控制器用户定义的原生数据类型。*
服务器	利用此驱动程序的 OPC/DDE/专有服务器。
结构化数据类型	预定义或用户定义的数据类型，由数据类型本质上是原子或结构的数字构成。
结构标记	用结构化数据类型定义的原生标记。

*服务器中支持的数据类型在数据类型说明中列出。

索引

D

I

L

R

S

帮

本

标

不

布

操

常

超

从

错

地

端

短

对

非

封

浮

概

高

故

降

结

仅

控

块

扩

来

连

链

名

模

请

驱

全

日

冗

扫

设

事

术

数

双

通

无

项

协

型

性

选

寻

已

以

用

优

有

原

站

长

支

自

字

遵