错误原因

简介

在 TIA Portal 中，可根据不同的错误类型和原因，确定相应的响应机制，而且对错误相应可采用各种不同的编程语言。但具体的响应机制显示则取决于所有的编程语言。

错误的确定需视具体情况而定。例如，加法运算过程中发生溢出可能是因为返回错误值而导致的错误。但在某些情况下，加法溢出是系统预定义的且可以接受，因此不再认为一种错误。

在创建程序代码时，必须充分了解可能会发生的各种状况。例如，在编程通信连接时，就必须意识到所创建的连接随时可能会被中断。为了防止这一状况的发生，必须在程序中设置相应的的故障响应。这是因为发生连接中断时将导致 T_SEND 程序块无法进行消息传输。只有采取了相应措施，在连接中断时系统才能发送信号通知操作员消息无法传输。由于指令 T_SEND 无法防止连接中断，因此可通过 T_SEND 的输出错误信息加以提示。程序员不应该忽略该指令的输出。

在下文中，我们将其统称为错误，即使该错误在系统中为预定义的行为。

导致错误的不同原因，可以将错误原因分为以下几类：

	参数值错误	编程错误	资源错误
说明	指令处理直接中发生的错误。	可导致指令执行中止的编程或访问错误。	由操作系统处理的错误，在程序代码中可编程该错误的响应措施。
错误类型	处理错误	同步错误	异步错误
示例	算术指令中的溢出错误	编程错误 • 查询一个不存在的外设输入 • 通过变量下标访问	发生了程序代码外的特定事件。

		ARRAY 时，下标值超出有效的 ARRAY 限值	
程序或操作系统的响应	不执行由使能输出 ENO 关联的指令。	如果没有编程错误 OB，则操作系统将根据所使用的 CPU 进行响应。	1. 如果未向该事件分配组织块 (OB)，则在事件发生时，操作系统执行默认的系统响应。 2. 如果为该事件分配有组织块 (OB)，则调用该组织块。
程序中的错误处理机制	根据具体的指令，可采取不同的本地错误处理方式2)： • EN/ENO 机制 • 输出参数： ◦ RET_VAL ◦ STATUS ◦ ERROR	全局错误处理1)： • 程序执行错误 OB • 编程错误 OB • I/O 访问错误 OB 使用以下指令，进行本地错误处理2)： • GET_ERROR • GET_ERR_ID	错误组织块 (OB)： • 时间错误 (OB 80) • 诊断中断 (OB 82) • 插入/移除模块中断 (OB 83) • 机架错误 (OB 86) 未分配错误 OB 时可能的系统响应： • 操作系统忽略该事件。 • CPU 切换为 STOP 模式。

- 如有可能，在本地执行错误处理。

如果分配有错误 OB，则在发生相应事件时将调用该 OB。

有关用户程序中诊断功能的应用示例，请访问评估错误 OB 中的错误。

1) 通过组织块，可执行全局错误处理。

2) 在程序代码内，可编程本地错误处理。

说明

异步错误处理

对于 S7-1500 系列的 CPU，错误 OB 采用异步调用方式。这也就意味着，在发生错误时不会立即处理 I/O 访问错误或编程错误 OB，而是根据设定的优先级进行相应的延时处理。如果在完成 I/O 访问错误或编程错误 OB 处理之前又发生了其它错误，则系统也不会调用其它 I/O 访问错误或编程错误 OB。如果要防止系统忽略这些 I/O 访问或编程错误 OB，需设置较高优先级。

错误处理机制概览

可通过以下几种不同的错误处理机制进行参数跟踪或编程或访问错误：

机制	任务	错误处理
使能输入 EN 或 IF 指令	阻止程序代码的执行	本地

使能输出 ENO	指示一个错误	
或二进制结果位		
参数输出		
RET_VAL、STATUS 和 ERROR		
GET_ERROR 和 GET_ERR_ID 指令	响应一个错误	
组织块		全局

参数值错误时的本地错误处理

错误发生后，除了通过本地错误处理机制立即进行响应，也可通过程序代码内进行特定响应。此时，可以在程序块（OB、FB 或 FC）中直接编写本地错误处理方式，但系统仅对发生在该程序块中的错误进行处理。

本地错误处理的优势

- 可根据该错误信息编写程序块中发生相应错误时的响应措施。
- 所编写的错误评估和错误响应不会中断程序的循环运行。
- 本地错误处理不会影响系统性能。如果错误未发生，则不会执行所编写的错误分析和响应措施。

下标列出了各种不同的本地错误处理方式：

错误处理方式	适用范围	说明
EN/ENO 机制 1)	S7-300 / S7-400 / S7-1200 / S7-1500	通过使能输出 ENO 检测特定的运行时错误并进行相应处理。后续指令的执行取决于该使能输出的信号状态。通过 EN/ENO 机制，可有效避免程序崩溃。块状态将以布尔型变量形式进行传递。
输出参数 STATUS 和 ERROR	S7-300 / S7-400 / S7-1200 / S7-1500	STATUS 和 ERROR 参数作为系统函数块 (SFB) 的返回值时，可查询块特定的错误信息，并按照预

		定义的结构进行输出。
输出参数 RET_VAL	S7-300 / S7-400 / S7-1200 / S7-1500	输出参数 RET_VAL 作为顺序功能图 (SFC) 的返回值时，可显示常规的错误代码或特定的错误代码。所谓常规的错误代码对应于所有指令，而特定的错误代码仅适用于特定指令。最多可以输出一个 INT 或 WORD 数据类型的变量。

1) 如果指令的参数未导致任何存储器访问错误，则相关的使能输出 ENO 将返回信号状态“1”，并在输出中返回可查询的有效值。

发生编程错误时的全局和本地错误处理

通过全局和本地错误处理，可立即对发生的错误进行响应而无需将 CPU 切换为“STOP”模式。可通过以下方式处理编程错误和访问错误：

全局错误处理的类型	适用范围	说明
程序执行错误 OB (OB 85)	S7-300 / S7-400	如果未使用 OB 85，则在发生程序执行错误时 CPU 将从 RUN 模式切换为 STOP 模式，并在诊断缓冲区内生成一个条目。
发生编程和访问错误时的 CPU 内部错误处理	S7-1200	发生错误时，CPU 将在诊断缓冲区中生成一个条目并保持为 RUN 模式，无需进行额外编程。
编程错误 OB (OB 121)	S7-300/ S7-400 / S7-1500	如果未使用 OB 121，则在发生编程错误时 CPU 将从 RUN 模式切换为 STOP 模式，并在诊断缓冲区内生成一个条目。
I/O 访问错误 OB (OB 122)	S7-300/ S7-400 / S7-1500	S7-300 / S7-400:

		如果未使用 OB 122，则在发生访问错误时 CPU 将从 RUN 模式切换到 STOP 模式。 S7-1500： 发生 I/O 访问错误时，CPU 将始终保持为 RUN 模式并在诊断缓冲区中生成一个条目。即使未使用 OB 122，也同样如此。

通过 GET_ERROR 和 GET_ERR_ID 指令，可直接在程序代码中集成本地错误处理。也可通过接收有关错误的详细信息并在错误附近的程序中对其进行评估。此时，可以在程序块（OB、FB 或 FC）中直接编写本地错误处理方式，但系统仅对发生在该程序块中的错误进行处理。

本地错误处理的类型	适用范围	说明
GET_ERROR 和 GET_ERR_ID 指令	S7-1200/S7-1500	通过该指令，可获得错误 ID 或详细的错误信息，并在程序代码中编写直接响应。 查询第一个错误信息时，将再次启用系统存储器中该错误所在的存储空间。如果随后发生其它错误，则将输出下一个错误的信息。

采用本地错误处理时，可通过 GET_ERROR 指令进行查询。支持以下几种默认的响应方式：

- 发生写错误时：将忽略该错误并继续程序运行。
- 发生读错误时：程序将继续运行，且算术指令的值将替换为“0”。
- 发生执行错误时：将停止该指令的运行，程序将运行下一条指令。

本地错误处理的优势

- 错误信息存储在系统存储器中，可对其进行查询和评估（如，通过 GET_ERROR 和 GET_ERR_ID 指令）。
- 可根据该错误信息编写程序块中发生相应错误时的响应措施。
- 所编写的错误评估和错误响应不会中断程序的循环运行。
- 本地错误处理对系统性能的影响要低于全局错误处理。如果错误未发生，则不会执行所编写的错误分析和响应措施。
- 如果在程序块中设置有本地错误处理，则在发生错误时将不执行全局错误处理。

说明

要防止在发生错误时 CPU 切换为 STOP
模式，无论是全局错误处理还是本地错误处理必须处理所有的编程错误和 I/O 访问错误。

通过输出参数 RET_VAL 评估错误

有关库块（SFB 和 SFC）错误分析的基础知识

除了输出参数 RET_VAL 外，还可以通过以下两个方式进行错误评估：

- 通过 EN/ENO 机制（LAD、FBD 和 SCL）。
- 通过状态字 (STL) 的 BR 位（二进制结果位）
- 通过输出参数 RET_VAL (return value)

使能输出 ENO
仅能通知发生了错误。如果需要了解所发生的具体错误，则需通过输出参数 RET_VAL 获得更多信息。通过该输出参数可判断 CPU
中该该指令是否成功执行。发生错误时，还可了解执行未成功执行的原因所在。

有关错误分析序列的建议

在评估指令特定的输出参数（如 OUT）之前，通常应执行以下步骤：

1. 首先评估先使能输出 ENO 或 STL 中状态字的 BR 位。
2. 检查输出参数 RET_VAL。

如果使能输出 ENO 或 BR 位指示在指令的执行过程中发生了错误，或者输出参数 RET_VAL 中包含了一个通用错误代码，则指令特定的输出参数将返回一个无效值。

如果使用输出参数 RET_VAL 指示发生了常见错误，则只能通过状态字 BR 位为值“0”来指示。返回值的数据类型为整数 (INT)。返回值通过值“0”指明在指令的执行过程中是否发生了错误。

常规和特定错误代码 (RET_VAL)

输出参数 RET_VAL 中有以下两类错误代码：

- 所有指令都可以输出的常见错误代码，
- 根据指令的特定功能输出的特定特定错误代码。

输出参数 RET_VAL 的数据类型为整型 (INT)。且该指令的错误代码安装十六进值进行分组。如果要检查返回值并与本文档中所列错误代码进行比较，则将以十六进制值形式显示这些错误代码。

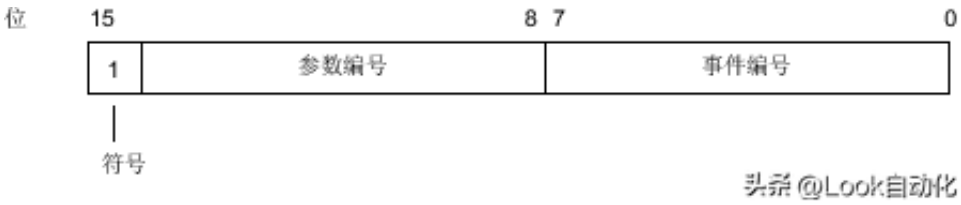
可以编写程序以便对执行指令过程中发生的错误进行响应。从而可以防止由于第一个错误而导致的更多错误。

说明

提供输入参数时出错

在执行包含 RET_VAL 参数的指令且在提供输入参数时出错，则参数 RET_VAL 将输出一个无效的错误代码且不对该指令的输出参数进行评估。

下图以十六进制格式显示了系统函数错误代码的结构。



说明

对常规错误代码的响应

如果在输出参数 RET_VAL
中输入了一个常规错误代码，则可能出现以下几种情况：

- 该指令相关的操作已开始或者已完成。
- 执行该操作时发生特定的指令错误。但在发生一个常规错误后，系统将不再指示该特定错误。

下表列出了一个返回值的常见错误代码。错误代码将显示为十六进制格式。代码编号中的字母 x 仅为一个占位符，表示导致该错误的系统函数参数编号：

错误代码 (W#16#...)	说明
8x01	VARIANT 参数的语法 ID 非法
8x22	读取一个参数时发生超出范围错误。
8x23	写入参数时发生超出范围错误。 此错误代码表示参数 x 完全或部分超出地址范围，或通过 VARIANT 参数指示位的长度不是 8 的倍数。
8x24	读取参数时发生超出范围错误。
8x25	写入参数时发生超出范围错误。 此错误代码表示参数 x 超出系统函数的有效范围。关于无效范围的信息，请参见各函数的描述信息。
8x26	此参数包含的定时器单元编号过高。

	此错误代码表示在参数 x 中指定的定时器单元不存在。
8x27	此参数包含的计数器单元编号过高（计数器编号错误）。
	此错误代码表示在参数 x 中指定的计数器单元不存在。
8x28	读取参数时发生对齐错误。
8x29	写入参数时发生对齐错误。
	此错误代码表示对参数 x 的引用是一个位地址不为 0 的操作数。
8x30	此参数位于只读属性的全局 DB 中。
8x31	此参数位于只读属性的背景 DB 中。
	此错误代码表示参数 x 位于只读属性的数据块中。如果通过系统函数本身来打开该数据块，那么系统函数将始终返回值 W#16#8x30。
8x32	此参数包含的 DB 编号过高（DB 编号错误）。
8x34	此参数包含的 FC 编号过高（FC 编号错误）。
8x35	此参数包含的 FB 编号过高（FB 编号错误）。
	此错误代码表示参数 x 包含的块编号高于所允许的最高编号。
8x3A	此参数包含尚未加载的 DB 编号。
8x3C	此参数包含未加载 FC 的编号。
8x3E	此参数包含未加载 FB 的编号。
8x42	系统尝试从外设输入区读取一个参数时，发生访问错误。
8x43	系统尝试向外设输出区写入一个参数时，发生访问错误。
8x44	出错后，第 n (n > 1) 次发生读访问错误。
8x45	出错后，第 n (n > 1) 次发生写访问错误。

	此错误代码表示对访问所需参数的拒绝被拒绝。
8x7F	内部错误 该错误代码表示参数 x 处发生了内部错误。

特定的错误代码

某些指令在其返回值中将提供一个该指令特定的错误代码，指示这些错误只会在特定的指令中发生。

特定的错误代码包含又以下两个数字：

- 0 至 7 之间的错误类别。
- 0 至 15 之间的错误编号。

指令 GET_ERROR 和 GET_ERR_ID 的用法

简介

通过本地错误处理，可以查询程序块内发生的错误并对相关的错误信息进行评估。可以为组织块 (OB)、功能块 (FB) 和函数 (FC)
设置本地错误处理。如果启用了本地错误处理，则将忽略系统响应。

可在指令 GET_ERR_ID 的错误信息中读取相应的错误编号。例如，通过指令 GET_ERROR 的错误信息，可以确定导致访问错误的参数。要确保指令输出所需的错误信息，必须在用户程序中对待评估错误的各个程序块进行相应编程。使用这些指令时，不会调用任何错误 OB，而且不会在诊断缓冲区写入任何条目。采用这种错误处理方法，可通过编写错误响应方式在发生错误时主动干预程序序列的执行。由于错误可能发生在程序块内的任何位置，因此我们建议在程序块末尾处添加该指令。

指令 GET_ERROR 和 GET_ERR_ID 的主要区别在于输出的错误信息数量不同。

在程序代码中加入其中一条指令后，在巡视窗口的“属性 > 特性”(Properties > Attributes) 下方选择“在块内处理错误”(Handle errors within block) 复选框。该设置不能在巡视窗口内编辑。只需删除所插入的本地错误处理指令，即可取消激活本地错误处理。

说明

块属性“在块内处理错误”(Handle errors within block)

该设置既不会应用于块调用中，也不会传输到所调用的程序块中。如果没有为高级别和低级别的程序块编程专用的本地错误处理方式，则可应用系统设置。

错误输出优先级

在本地错误处理过程中，可通过指令 GET_ERROR 或 GET_ERR_ID 显示发生的第一个错误信息。如果在指令的执行过程中同时发生多个错误，则将根据这些错误的优先级进行显示。下表列出了不同类型错误的优先级：

优先级	错误类型
1	程序代码错误
2	缺少引用
3	范围无效
4	DB 不存在
5	操作数不兼容
6	指定的区域宽度不够
7	定时器或计数器不存在
8	无法写入 DB
9	I/O 错误
10	指令不存在
11	块不存在
12	嵌套深度无效

最高优先级为 1，最低为 12。