**ASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY**

*LIGO Laboratory / LIGO Scientific Collaboration*

LIGO- E1200545-v1 Advanced LIGO 5/29/2012

# TwinCAT Library for Error Handling

Daniel Sigg

Distribution of this document:
LIGO Scientific Collaboration

This is an internal working note
of the LIGO Laboratory.

**California Institute of Technology**
**LIGO Project – MS 18-34**
**1200 E. California Blvd.**
**Pasadena, CA 91125**
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

**Massachusetts Institute of Technology**
**LIGO Project – NW22-295**
**185 Albany St**
**Cambridge, MA 02139**
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

**LIGO Hanford Observatory**
**P.O. Box 159**
**Richland WA 99352**
Phone 509-372-8106
Fax 509-372-8137

**LIGO Livingston Observatory**
**P.O. Box 940**
**Livingston, LA 70754**
Phone 225-686-3100
Fax 225-686-7189

http://www.ligo.caltech.edu/

| **Library** | |
|---|---|
| Title | Error |
| Version | 1 |
| TwinCAT version | 2.11 |
| Name space | – |
| Author | Daniel Sigg |
| Description | Handles error conditions for TwinCAT libraries.<br>The library exports an error structure for reporting error conditions. This structure is typically declared as the first element of an interface structure. A handler is provided to report errors. Errors are bit encoded into a DWORD. Error messages for multiple simultaneous conditions are concatenated until the maximum string length is reached. If there are additional error conditions, this is indicated with a '+' character at the end of the string. Multiple error conditions can always be reconstructed from the bit-encoded error code. |
| Error codes | Library dependent |
| Library dependencies: | none |

| **User Interface Type**<br>TYPE ErrorStruct :<br>STRUCT<br>Error: BOOL;<br>ErrorCode: DWORD;<br>ErrorMessage: STRING;<br>END_STRUCT<br>END_TYPE | |
|---|---|
| Type name | ErrorStruct |
| Description | Structure in a user interface to define an error condition |
| Definition | STRUCT |
| Output Tag | Name: Error<br>Type: BOOL<br>Description: Flag indicating any error condition |
| Output Tag | Name: ErrorCode<br>Type: DWORD<br>Description: Bit-encoded value describing the error condition(s) |
| Output Tag | Name: ErrorMessage<br>Type: STRING<br>Description: Human readable text describing the error |

| **Enum Type**<br>TYPE ErrorMethodEnum : (Init, Report, Commit);<br>END_TYPE | |
|---|---|
| Type name | ErrorMethodEnum |
| Description | Describes the handler method |
| Definition | ENUM |
| Enum tag | Name: Init<br>Description: Initializes the error handler |
| Enum Tag | Name: Report<br>Description: Reports an error condition |
| Enum Tag | Name: Commit<br>Description: Commit the error to the interface structure |

| **Function Block**<br>FUNCTION_BLOCK ErrorHandlerFB<br>VAR_INPUT<br>ErrorMethod: ErrorMethodEnum;<br>ErrorCode: DWORD := 0;<br>ErrorMessage: STRING := '';<br>END_VAR<br>VAR_IN_OUT<br>Error: ErrorStruct;<br>END_VAR | |
|---|---|
| Name | ErrorHandlerFB |
| Description | Handles error conditions. |
| Input argument | Name: ErrorMethod<br>Type: ErrorMethodEnum<br>Description: For initialization, report and commit |
| Input argument | Name: ErrorCode<br>Type: DWORD<br>Description: Error code |
| Input argument | Name: ErrorMessage<br>Type: STRING<br>Description: Error message string |
| In/out argument | Name: Error<br>Type: ErrorStruct<br>Description: Error structure |

**Example:**

```
VAR
        ErrorHandler:                   ErrorHandlerFB;
END_VAR

(* Initialize *)
ErrorHandler (ErrorMethod := Init, Error := LowNoiseVco.Error);
(* read and process values... report an error... process further... *)
...
IF  NOT LowNoiseVco.PowerOk THEN
        ErrorHandler (ErrorMethod := Report, Error := LowNoiseVco.Error,
                ErrorCode := 16#01, ErrorMessage := 'Power supply voltages out-of-range');
END_IF;
...
(* Finally commit *)
ErrorHandler (ErrorMethod := Commit, Error := LowNoiseVco.Error);
```