



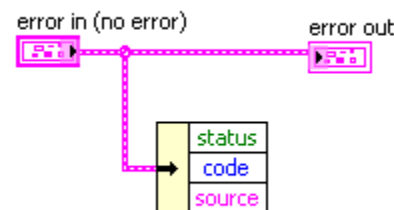
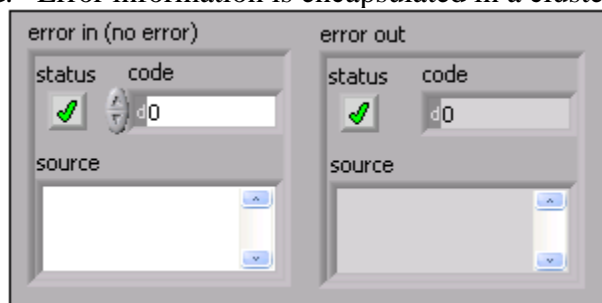
SPEctrometer and Instrument Control Environment

How to use the SPICE Error Handler:

Error trapping is easy. Error propagation and resolution is not easy. If a serious error occurs many levels deep in the program, it is difficult to get the higher level routines to do something sensible. One possibility is to ignore the errors and let the program crash and burn on its on. This approach works surprisingly well, but in the case of SPICE it can definitely lead to inefficient use of an instrument. To combat this problem, it was decided to write a SPICE specific error handler that sits on top of the error trapping and processing mechanism that is built into LabVIEW. This allows us to take advantage of the LabVIEW features for error handling which in turn greatly simplifies the process.

LabVIEW Errors:

Error handling in LabVIEW follows the dataflow model. Just as data flow through a VI, so can error information. As a VI runs, LabVIEW tests for errors at each execution node. If LabVIEW does not find any errors, the node executes normally. If LabVIEW detects an error, the node passes the error on to the next node without executing. The next node does the same thing, and so on until execution is passed all the way back up the calling chain to the end of the execution flow (e.g. drive_motors.vi) where it can be processed. Error information is encapsulated in a cluster:



- **status** is a Boolean value that reports True if an error occurred. Most VIs, functions, and structures that accept Boolean data also recognize this parameter. For example, you can wire an error cluster to the Boolean input of a case structure or a while loop terminator.
- **code** is a 32-bit signed integer that identifies the error numerically.
- **source** is a string that identifies where the error occurred.

SPICE Specific LabVIEW Errors:

SPICE specific errors use the same cluster structure as LabVIEW errors. Error codes 5000-9999 are

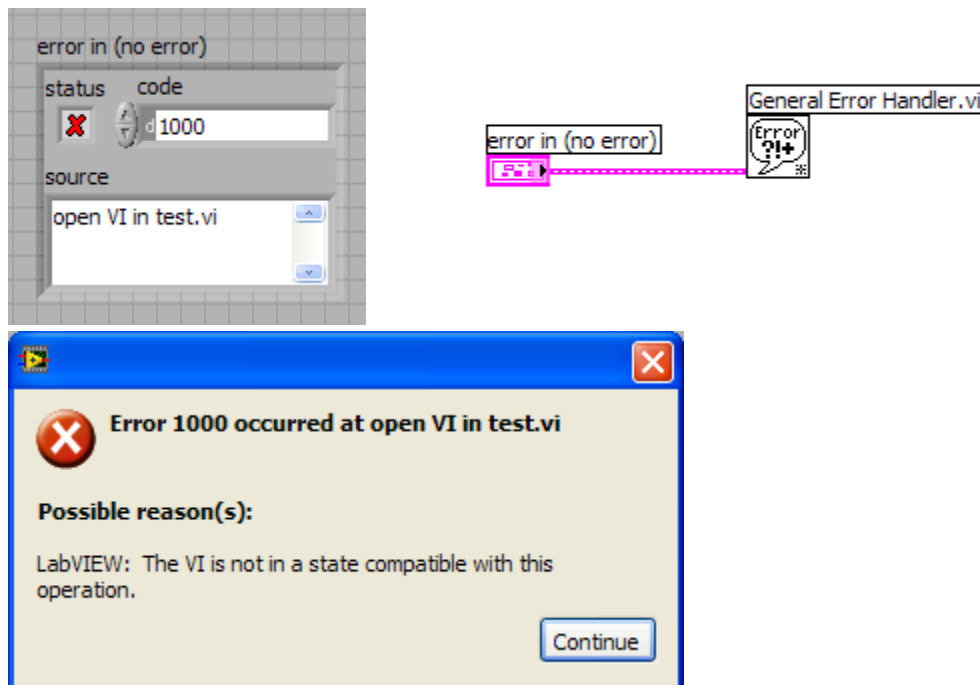
reserved for custom errors and we have designated 6 generic SPICE errors:

- 5000: SPICE Warning
- 5001: SPICE Error
- 5002: SPICE Fatal Error
- 5003: SPICE Device Warning
- 5004: SPICE Device Error
- 5005: SPICE Fatal Device Error

With these errors we have departed a little from the scheme used in LabVIEW itself. Rather than defining many specific error codes with associated detailed error messages, we defined a few generic error codes and pass the detailed error message along with the error cluster as part of the **source** string (which is not used by LabVIEW for anything except constructing the error message). The **status** flag is used in the usual way although the trick of setting the error code but not the status in order to distinguish between an error and a warning is not used, rather the error codes denote the error level. However, the SPICE specific errors will be recognized by the system and treated just the same as LabVIEW generated error clusters.

SPICE Specific Error Messages:

LabVIEW errors consist of two parts, the "source" and the "message". The source of the error is determined from the **source** field of the error cluster and the message (displayed under "Possible reason (s):") is looked up based on the error **code**. For example the following error cluster produces the error dialog shown:



In the case of SPICE specific errors both the message and the source are specified in **source**. The

message is taken from the plain text in source and the source of the error is specified by using "tags". The tag syntax is a keyword and optional parameter delimited by < and > (eg. <keyword parameter>). Keywords are not case sensitive and must not contain white space characters while the optional parameter is case sensitive and may contain any printable character except < or >. There are additional tags that control the behavior of the dialog boxes, writing to log files, program execution, etc.

The available tags are:

<OK xx>	Generate a popup dialog with an "OK" button, the optional parameter "xx" indicates a timeout in seconds, if the timeout is omitted the dialog will not timeout
<continue xx>	Generate a popup dialog with a "Continue" button, the optional parameter "xx" indicates a timeout in seconds, if the timeout is omitted the dialog will not timeout
<continue_exit xx>	Generate a two button popup dialog with "Continue" and "Exit" buttons, the optional parameter "xx" indicates a timeout in seconds, if the timeout is omitted the dialog will not timeout, if the dialog times out the default selection is "Continue", SPICE will exit if the "Exit" button was selected
<exit_only xx>	Generate a popup dialog with an "Exit" button, the optional parameter xx indicates a timeout in seconds, SPICE will exit after the dialog closes
<no_popup>	Inhibit any previously defined or default popup dialogs and timeouts
<log_file>	Write error message to the user log file
<no_log_file>	Suppress writing to the user log file
<device_log>	Write to the device log file
<no_device_log>	Suppress writing to the device log file
<device_index xx>	Used for device errors, xx is the index of the device in device globals, this will cause the name of the device to be included in the error message
<method xx>	Used for device errors, "xx" is the method on the device driver that generated the error
<vi xx>	Used to specify the name of the VI where the error occurred, "xx" is the text VI name
<location xx>	Location where error occurred, used to indicate where inside a VI an error occurred, "xx" is the text that will be included in the source part of the error message
<exit_spice>	Force SPICE to exit after all dialogs/timeouts are completed
<no_shutdown>	Do not attempt to shutdown hardware on exit
<clear_error>	Pass (no error) to error out
<no_labview_error_handler>	Do not pass non SPICE errors through to the built in error handler

In order simplify the process of creating a SPICE specific error cluster, default behavior for each error code has been defined:

5000: SPICE Warning: "OK" popup dialog timed for 60 sec, write error message to the user log file

5001: SPICE Error: "Continue" popup dialog timed for 60 sec, write error message to the user log file

5002: SPICE Fatal Error: Two button "Continue/Exit" popup dialog, no timeout, write error message to the user log file

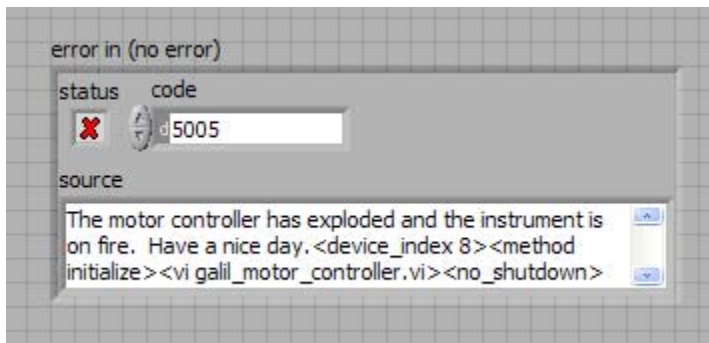
5003: SPICE Device Warning: "OK" popup dialog timed for 60 sec, write error message to the device log

5004: SPICE Device Error: "Continue" popup dialog timed for 60 sec, write message to the user log file and to the device log file

5005: SPICE Fatal Device Error: Two button "Continue/Exit" popup dialog, no timeout, write error message to the user log file and the device log file

These default settings can be modified by including the appropriate tags in the error cluster (eg. inserting `<no_popup>` into **source** will suppress the default popup dialogs). Also the order in which the tags are included is important since tags are processed in the order they appear in **source**. In other words if the following tags are appended to source `<no_popup><OK 45>` a popup dialog with a 45 sec timeout will be generated; whereas appending `<OK 45><no_popup>` will not generate a popup dialog. The default settings are treated "preset" tags that can be superseded by tags appended to **source**.


The following error cluster produces the dialog shown plus an entry in the log file:

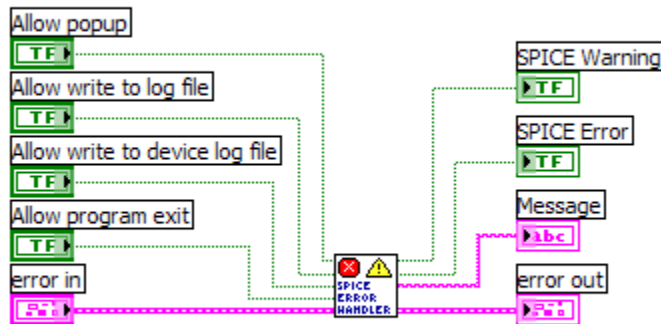




SPICE Specific Error Handler:

When it comes time to process the error cluster, how does LabVIEW know to treat the SPICE specific error clusters differently or what to do with all those "tags" that have been defined? Well, it doesn't. Instead, if you are using SPICE specific errors you need to use pass your error clusters to SPICE

error_handler.vi  rather than the LabVIEW general error handler when you are ready to process the error cluster. SPICE_error_handler.vi can be used as a general replacement for the built-in General_Error_Handler.vi because if SPICE_error_handler.vi gets passed a non SPICE specific error cluster, it just passes it on to General_Error_Handler.vi. The inputs to the SPICE_error_handler.vi are shown below and can be used to override the tags appended to **source**:



Inputs:

Allow popup: [True] Determines if popups are allowed if one is requested.

Allow write to log file: [True] Determines if the VI is allowed to write a message to the log file if needed.

Allow write to device log file: [True] Determines if the VI is allowed to write a message to the device log file if needed.

Allow program exit: [True] Determines if SPICE should exit if directed to by a <tag> or user selection.

error in: [no error] error cluster.

Outputs:

SPICE Warning: True if the input error cluster was a SPICE Warning or SPICE Device Warning; otherwise false.

SPICE Error: True if the input error cluster was a SPICE Error, SPICE Fatal Error, SPICE Device Error, or SPICE Fatal Device Error; otherwise false.

Message: The message text generated from the error cluster.

error out: A copy of (error in) unless the <clear_error> tag was specified.

Create a SPICE Specific Error Cluster:

There is a VI to make creating a SPICE specific error cluster easier:

