VISTA
CONTROL SYSTEMS

Vsystem V4.3 User's Guide Utilities and Scanners

Updated 2026

Table of contents

About this Guide

The information in this document is subject to change with out notice and should not be construed as a commitment by Vista Control Systems, Inc. Vista Control Systems, Inc., assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license. Licensed users may copy this document for internal use.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Vista Control Systems, Inc., or its affiliated companies.

Vsystem , Vdraw , Vaccess, Vgen, Vlogger, Valarm , Vczar, Vscan, Varchive, Vtrend, Vscript, and the Vista Logo are trademarks of Vista Control Systems, Inc.

Windows, Windows XP, Windows 7, 8, 10, 11, and Windows Server are trademarks of Microsoft Corporation.

VMS, AXP, OpenVMS, DECnet, and DECwindows are trademarks of Hewlett Packard Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

XWindows, Motif, and UNIX are trademarks of the Open Group.

GIF and “Graphics Interchange Format” are trademarks of CompuServe, Inc.

Acrobat and Adobe are trademarks of Adobe Systems, Inc.

Other product names mentioned may be trademarks and are mentioned for identification purposes only.

Updated 2025

Conventions Used in This Guide

The following labels guide you to information specific to different operating systems:

WIN Precedes information for the Windows operating systems, which includes Windows XP, Windows 7, 8, 10, 11, and Windows Server.

LINUX Precedes information for the Linux operating system.

VMS Precedes information for the OpenVMS operating system.

Commands are represented by a fixed-width font and special color, for example, `-vlogwin` .

Keywords and qualifiers are represented by italics.

Brackets [ ] in the text (as opposed to those found in commands) enclose optional specifications. For example, the user can replace [channelname] with an actual channel name, but is not required to do so.

Angle brackets <> in the text (as opposed to those found in commands) enclose information that the user must replace with the corresponding specification. For example, the user must replace <channelname> with an actual channel name.

Menu items, buttons, and dialog options are represented by boldface text, for example, Select **Acknowledge** on the **Options** menu in the menu bar.

Unless otherwise specified, directions for using mouse buttons refer to the left mouse button.

Revisions to this document

Date	Vsystem Version	Document Version	Last Patch Updated
Initial Release	4.3	1.0	N/A
January 2014	4.3	2.0	4.3.20
January 2018	4.3	3.0	4.3.26
January 2026	4.3	3.1	4.3.26

Starting Vsystem Applications

Vsystem applications are designed to operate the same across multiple platforms. All Vsystem applications, include the graphical program can be run from the command line with qualifiers and parameters. Windows and OpenVMS both use `/` for qualifiers, whereas `-` is commonly used on Linux.

```
db_show /process db1
db_show -process db1
```

Qualifiers that have values require the use of an equal sign '='. There should not be a space between the `=` and value.

```
vinfo -chan=1 db1
```

Qualifiers can have a list which is enclosed in ()

```
vmodbus -database=(db1,db2)
```

An asterisk is used for wildcard, for example with channel names. On Linux these must be enclosed in quotes.

```
db_access db1::*
db_access "db1::*"
```

Parameters can be a list separated by a comma. Spaces may also be added between the parameters.

```
db_view ds1,ds2
db_view ds1, ds2
```

If an applications takes more than 1 parameter, these are separated by spaces

```
vfind_channel db1 6
```

Most qualifiers can be abbreviated. But must be enough characters to be unique.

```
vscan -fast_rate=5 db1
vscan -fast=5 db1
vscan -f=5 db1
```

For example, vscan includes both noslow and noinit qualifiers.

```
vscan -nos db1
```

String with spaces like specifying a time string need double quotes "" or single quotes ''.

```
vtrend -start="31-may-2025 12:00"
```

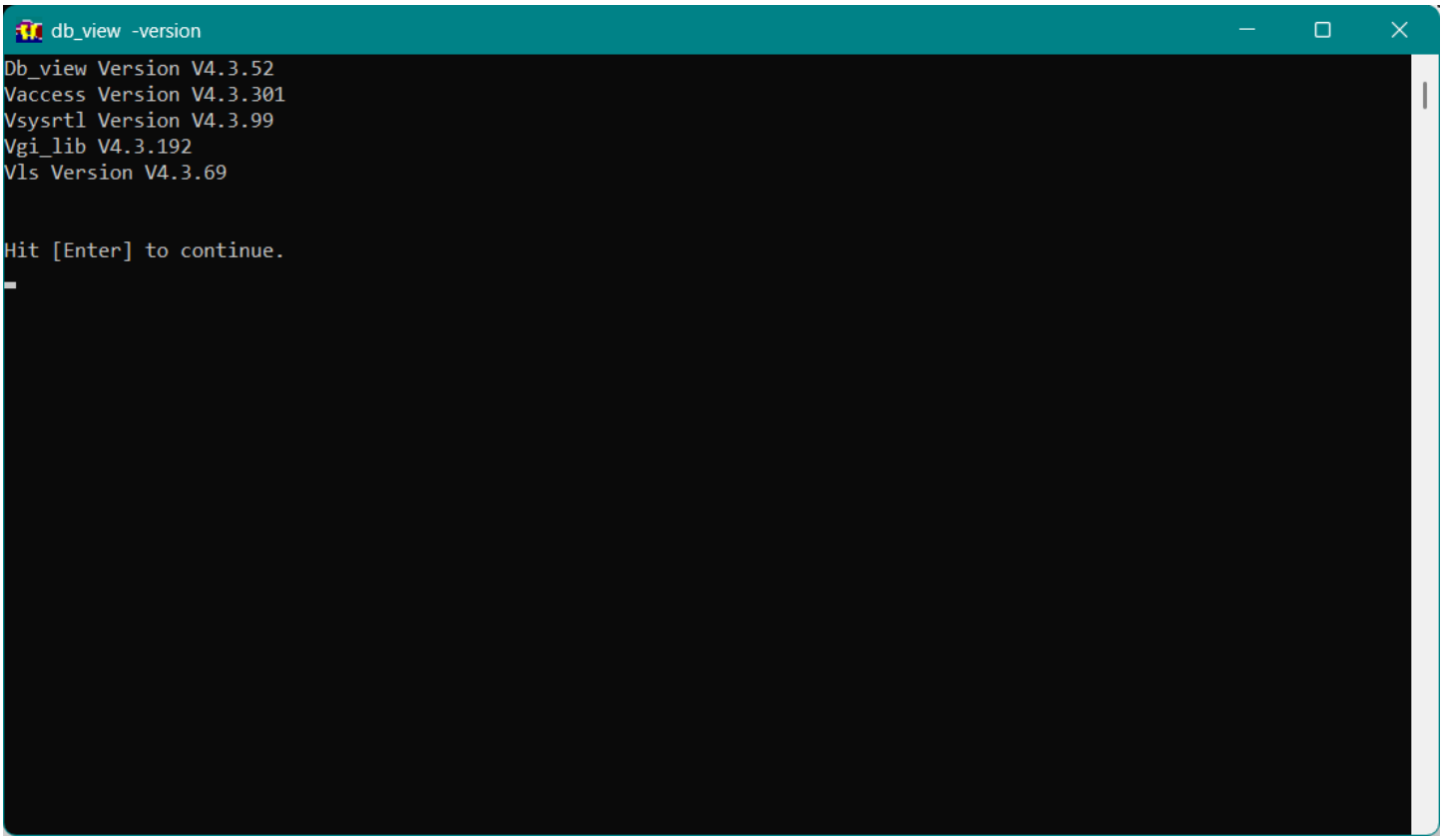
String within a string

```
vlog -sql "select cname from data@tutor.varc where time > '03-May-2001
10:52:00.00'"
```

All Vsystem applications have `help` and `version` qualifiers. The `help` qualifier lists the available qualifier and parameters for the program. The version qualifier lists the versions of the libraries that are used by the program.

```
C:\>db_access -version
Db_access Version V4.3.26
Vaccess Version V4.3.301
Vsysrtl Version V4.3.99


Hit [Enter] to continue.


C:\>db_access -help
  Usage:
      db_access Parameters Qualifiers

  Parameters:
      [ P1 ]
      [ P2 ]

  Qualifiers:
      /NOOUTPUT
      /[No]FORMATFILE [ =filename ]

Hit [Enter] to continue.
```

WIN If you use these qualifiers on a graphical program like db_view or vdraw, you will get an additional window. You will need to hit the **Enter** key to close the window. Even for the command line programs, you will need to hit **Enter**.



You can get a list of versions for all of the Vsystem programs using vinfo with the allversions qualifier

```
vinfo -allversions
```

Using Vsystem Graphical Applications

All Vsystem graphical applications use a library of portable graphics routines, which enables the applications to run on X Windows or Windows. This library has a configuration file, vgi.vdef, that you can use to set up some defaults for the Vsystem Graphics Interface (VGI) library. When you run a Vsystem graphical application, the VGI library searches for a `vgi.vdef` file in the search order listed below. The search will continue through the entire list, reading every located .vdef file. If a duplicated keyword is encountered in any of these files, the last-read value for this keyword will override the previous value(s).

Search order for the `vgi.vdef` file:

The Vsystem root directory.

1. The Vsystem host-specific directory.
2. The user's home directory (translation of HOMEDRIVE and HOMEPATH for Windows, sys$login for OpenVMS, and HOME for Linux).
3. The current working directory.

If you want to specify certain keywords that cannot be overridden by user settings, place these keywords in a `vgi_system.vdef` file; this file is searched for in the following directories:

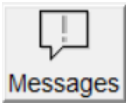
1. The Vsystem host-specific directory.
2. The Vsystem root directory.

The configuration file consists of any of the keywords found in VGI Configuration Keywords, followed by a value. String values that may contain spaces should be enclosed in quotation marks. In addition to the command-line interface, you may also use VGI Configuration Setup to specify the contents of the `vgi.vdef` file.

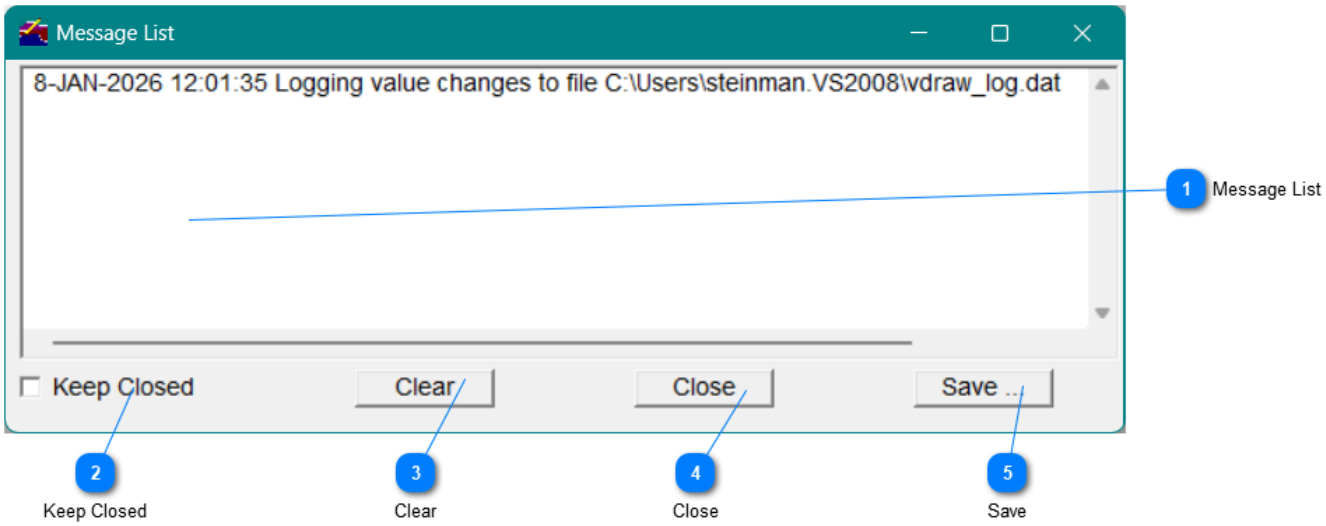
Common Dialog Boxes

Messages

Many Vsystem graphical program will have a messages window that can display and errors or other useful information. You can open this window by selecting the **Messages** on the toolbar. For some porgrams, it may be abbreviated to **Msgs**, or there may not be any text, but the icon will be the same.



You can also select **Messages** from the menu and then **View**.



1 Message List

New message will appear here.

2 Keep Closed

Select this so that the Message List will not popup when a new message arrives.

3 Clear

Select Clear to remove all messages.

4 Close

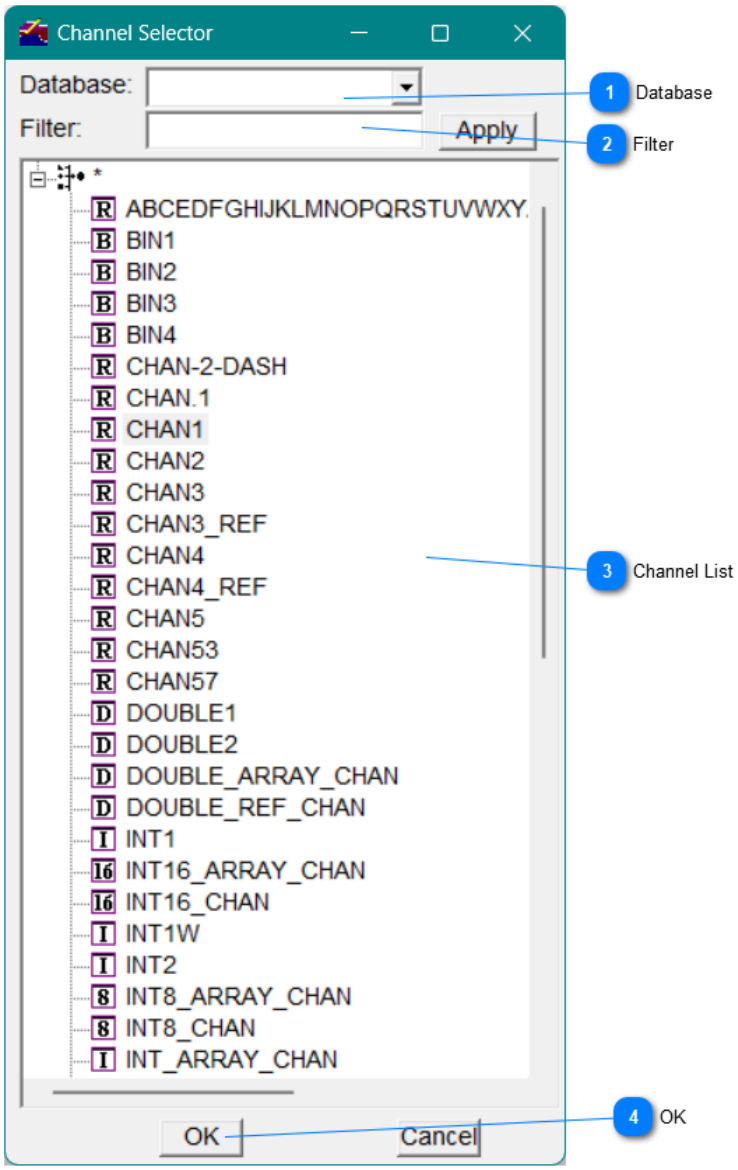
Closes the Message List window.

5 Save

Save the message to a text file.

Channel list and selection

Many programs allow you to select channels from a list. For selecting a single channel, you will see the window below.



1 Database

Select the database to view it's channels.

2 Filter

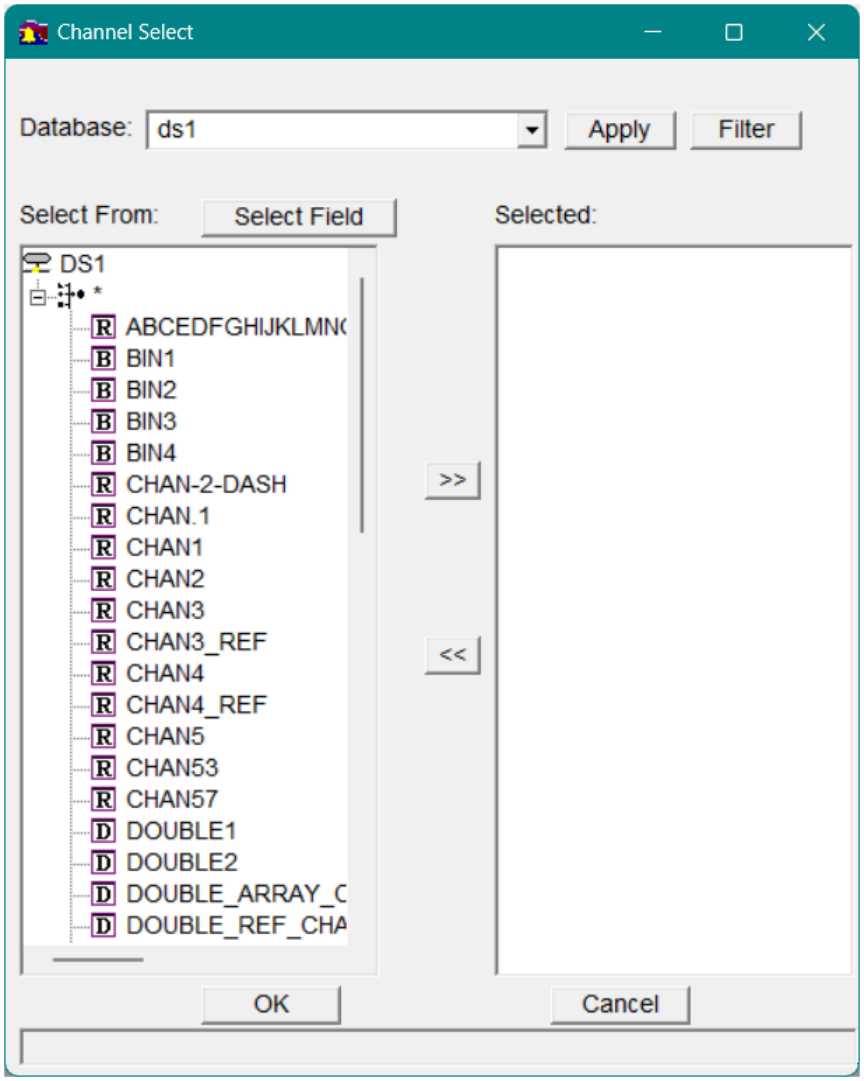
Add a filter. For example, "R*"

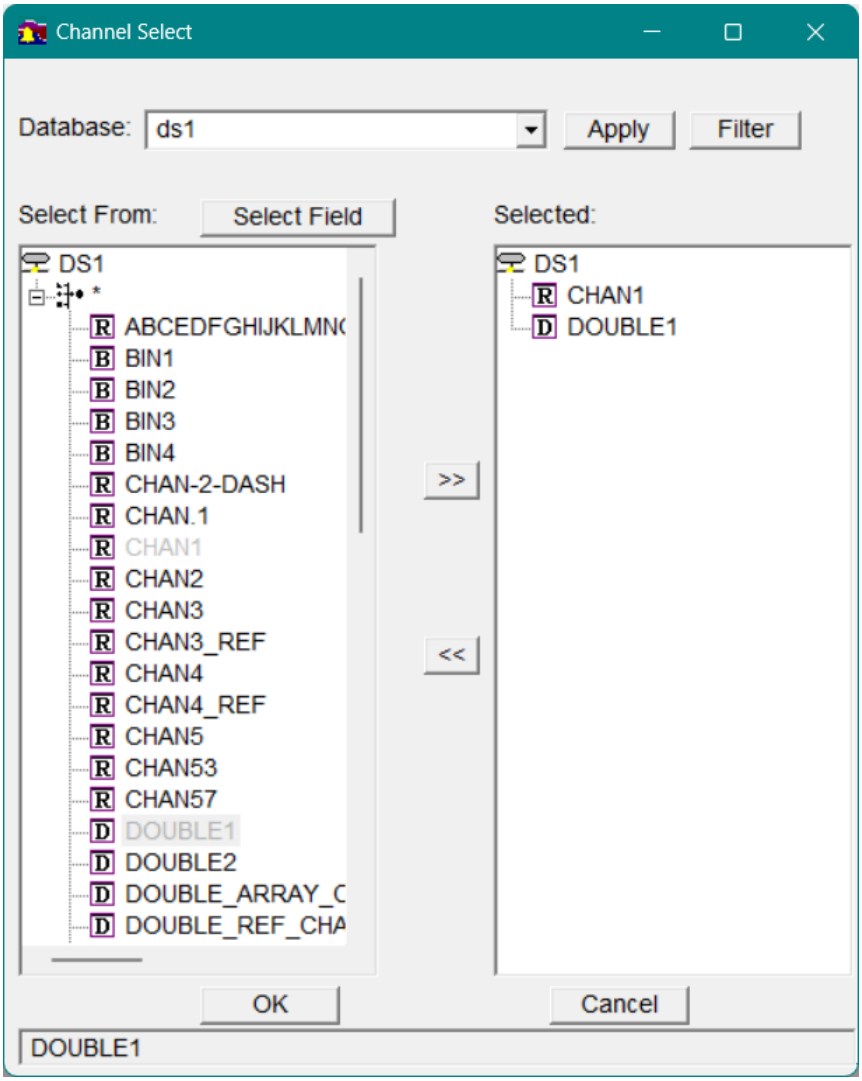
3 Channel List

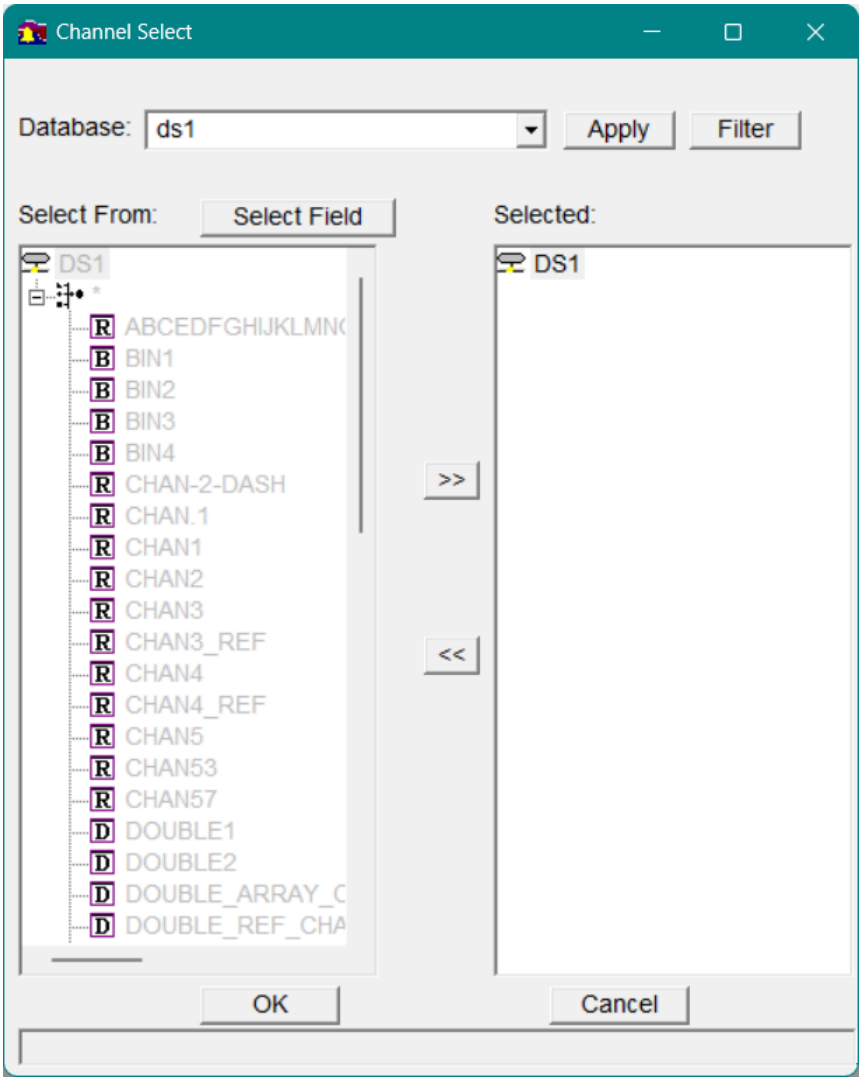
4 OK

Press **OK** when down to save the channel.

For some programs, you can select multiple channels or the whole database.




Channel Select
Database:
ds1
Apply
Filter
Select From:
Select Field
Selected:
DS1
*
ABCEDFGHIJKLMN
BIN1
BIN2
BIN3
BIN4
CHAN-2-DASH
CHAN.1
CHAN1
CHAN2
CHAN3
CHAN3_REF
CHAN4
CHAN4_REF
CHAN5
CHAN53
CHAN57
DOUBLE1
DOUBLE2
DOUBLE_ARRAY_C
DOUBLE_REF_CHA
>>
<<
DS1
CHAN1
DOUBLE1
OK
Cancel
DOUBLE1


Channel Select
Database:
ds1
Apply
Filter
Select From:
Select Field
Selected:
DS1
DS1
ABCEDFGHIJKLMN
BIN1
BIN2
BIN3
BIN4
CHAN-2-DASH
CHAN.1
CHAN1
CHAN2
CHAN3
CHAN3_REF
CHAN4
CHAN4_REF
CHAN5
CHAN53
CHAN57
DOUBLE1
DOUBLE2
DOUBLE_ARRAY_C
DOUBLE_REF_CHA
>>
<<
OK
Cancel

File open and save

VGI Configuration Keywords

Use the configuration keywords below to configure the Vsystem graphical applications. These keywords are case-sensitive; if you add keywords to your vgi.vdef file, make sure to spell the keywords exactly as shown below. You can find an example vgi.vdef file in the examples applications directory.

allowOffScreen

Use the `allowOffScreen` keyword to specify whether or not to allow a window to go onto multiple displays.

channelArchiveSortField

A string that specifies the channel field to use for sorting channels in the channel selector coming from archive files.

channelAreaSort

If set to 1, Vsystem channel selectors will sort channels in another level on the channel list tree, based upon the areas and subareas that have been defined in the database. Once the channels are sorted on area and subarea, they are finally displayed in the lowest level in the tree with either the channel name or the channel label, if the `channelShowLabel` option is on.

channelShow Label

If set to 1, Vsystem channel selectors will display the label for the channel instead of the channel name in the list of channels.

defaultBrowser

`defaultBrowser <browser command>`

Use the `defaultBrowser` keyword to specify a different command to use for the browser. Windows will default to running the HTML file that will bring up whatever the default browser is. Linux and OpenVMS will default to the command netscape .

defaultWinX

Use the `defaultWinX` keyword to specify the opening x position for Vsystem graphical applications.

defaultWinY

Use the `defaultWinY` keyword to specify the opening y position for Vsystem graphical applications.

error

Use the `error` keyword to display X Windows nonfatal errors if the value is 1.

fileDateNumCols

Use the `fileDateNumCols` keyword to specify the number of columns for a file date in a dialog.

fileNameNumCols

Use the `fileNameNumCols` keyword to specify the number of columns for a filename in a dialog.

fileOpenH

Use the `fileOpenH` keyword to specify the height, in pixels, for a file open dialog.

fileOpenW

Use the `fileOpenW` keyword to specify the width, in pixels, for a file open dialog.

fileSizeNumCols

Use the `fileSizeNumCols` keyword to specify the number of columns for a file size in a dialog.

fontName

Use the `fontName` keyword to specify the default font family name used in applications (such as Arial, Times New Roman, and so on).

fontPixelSize

Use the `fontPixelSize` keyword to specify the size of the font, in pixels, used in applications (such as 10, 15, 20, and so on).

fontWeight

Use the `fontWeight` keyword to specify the initial font weight used in applications (the value should be bold or normal).

htmlFileDir

```
htmlFileDir "<filename>"
```

Use the `htmlFileDir` keyword to specify a different location for the HTML files. This would be the top-level directory of the HTML files. The default will be the Vsystem doc html subdirectory.

noBorder

Use the `noBorder` keyword to turn off the window resize borders on all Vsystem graphics applications (maximum, minimum, and resize borders).

noTitleBar

Use the `noTitleBar` keyword to turn off the title bar on the top of all Vsystem graphics applications.

noTitleBarOptions

Use the `noTitleBarOptions` to turn off the title bar options for all windows.

printMargin

For X Windows, use the `printMargin` keyword changes the printer margins to a floating-point value in inches (the default is 0.3 inches).

textPrintFontName

Use the `textPrintFontName` keyword to specify a font family name used when printing text (for example, printing alarms in Valarm). The default font is `"Courier New"`.

textPrintFontSize

Use the `textPrintFontSize` keyword to specify the point size of the font used for printing text. The default size is 10.

toolbarLabels

Use the `toolbarLabels` keyword to display the toolbar button labels. When set to a value of 1, the labels are displayed; when set to 0, the labels are not displayed.

toolbarSize

Use the `toolbarSize` keyword to specify whether large or small icons are displayed in the toolbar (the value should be 32 or 16).

toolbarTips

Set to a value of 1 if you want the `toolbarTips` keyword to display tooltips when the mouse pointer passes over the toolbar; when set to 0, tooltips are not displayed.

turnOffPasteMenu

Use the `turnOffPasteMenu` keyword to turn off the copy/paste menu in text objects for X Windows, to prevent some X Terminals from grabbing the mouse.

useNonIsoFonts

Fonts that aren't ISO have been removed from the applications. If you need to restore them, use the `useNonIsoFonts` keyword to do so.

VGI Configuration Setup

The VGI Configuration Setup utility enables you to edit the contents of a `vgi.vdef` file.

Running VGI Configuration Setup

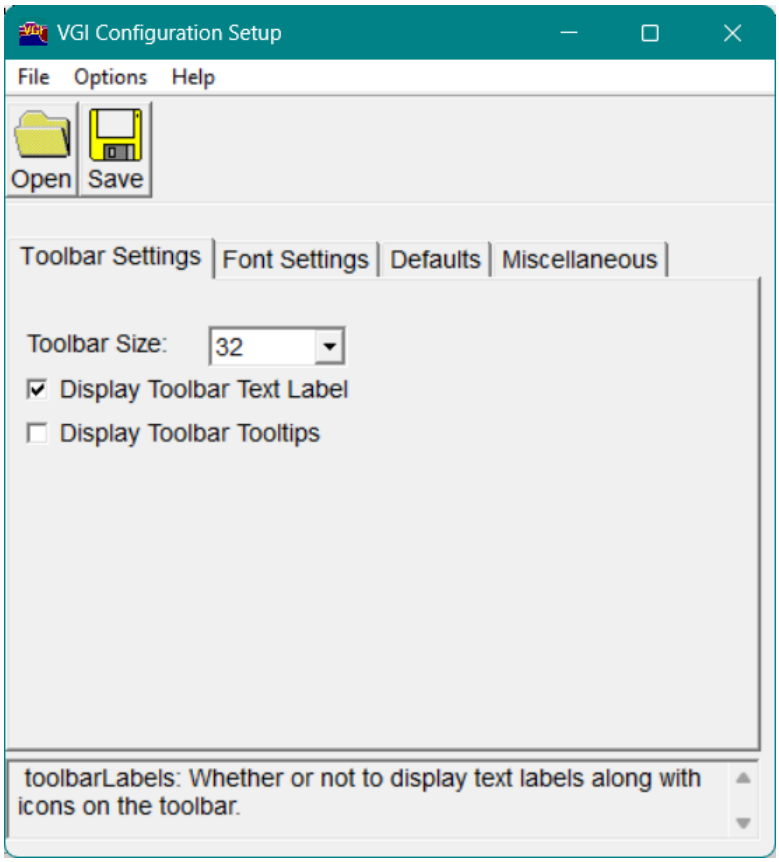
To create a vgi.vdef file

• Enter the following at the system prompt:

```
vgi -setup
```

The VGI Configuration Setup dialog appears, as shown in Figure 1.

Figure1 - VGI Configuration Setup dialog



Opening a File

To open a file for editing

• On the toolbar, click the Open button.

-or-

• On the File menu in the menu bar, select Open .

The values that appear in the fields will correspond to the values of the opened file. If you want to change these values, simply enter new values in the appropriate fields.

Saving a File

To save a file that you have edited

• On the toolbar, click the Save button.

-or-

• On the File menu in the menu bar, select Save .

If this is the first time you have saved this file, the Save As window will appear, in which you can enter a file name. If you have previously saved this file, when you select Save , you will overwrite what you previously saved.

To save a previously saved file under a new name

• On the File menu in the menu bar, select Save As .

The Save As window appears, in which you can enter a new file name.

Restoring Default Values

After making changes in the VGI Configuration Setup dialog, you may decide not to apply them.

To restore the file values to their original state (removing any changes you had made in the VGI Configuration Setup dialog)

• On the Options menu in the menu bar, select Restore Defaults .

> ***Note*** If you have already saved the changes you made in the VGI Configuration Setup dialog, you will need to select Save again after restoring the default values if you want to save the defaults. Otherwise, the values will revert to the last-saved values when you exit.

Getting Keyword Information

Located at the bottom of the VGI Configuration Setup dialog Figure 1 is an area that displays a brief description of the last-selected keyword entry (each field in the dialog has an associated keyword). For a more complete description of a keyword, including defaults where applicable,

Editing Your VGI Configuration Files

In the VGI Configuration Setup dialog, you can edit vgi.vdef files by selecting the following tabs: Toolbar Settings, Font Settings, and Miscellaneous.

Toolbar Settings Tab

Use the options on the Toolbar Settings tab to modify the toolbar settings for your vgi.vdef files.

> ***Note*** If you elect to display tooltips, tooltips pop up when the mouse pointer rests on an area that displays them, providing you with information about that area

Toolbar Size

From the drop-down list, select the size, in pixels, for the toolbar.

Display Toolbar Text Label

Select this option if you want the toolbar to display text labels along with icons.

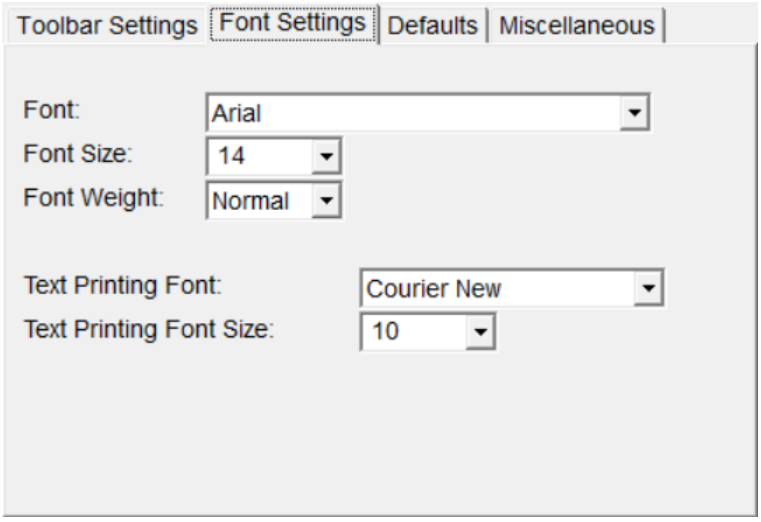
Display Toolbar Tooltips

Select this option if you want the toolbar to display tooltips (tooltips pop up when the mouse pointer rests on an area that displays them, providing you with information about that area).

Font Settings Tab

Use the drop-down lists on the Font Settings tab to modify the fonts for your vgi.vdef files.

Figure2 - Font Settings Tab



Font

From the drop-down list, select the font that you want to use as your default.

Font Size

From the drop-down list, select the default size, in pixels, you want to use for your font.

Font Weight

From the drop-down list, select the initial weight you want to use for your font.

Text Printing Font

From the drop-down list, select the font that you want to use for your default when printing text.

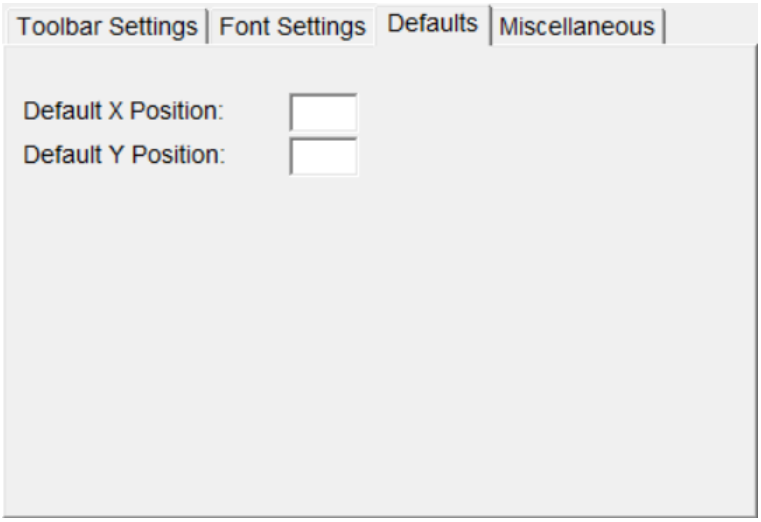
Text Printing Font Size

From the drop-down list, select the default size, in pixels, you want to use when printing text.

Defaults Tab

Use the fields on the Defaults tab to specify the default X and Y coordinates for the opening positions for Vsystem graphical applications.

Figure3 - Defaults Tab



Default X Position

The default X position of the window.

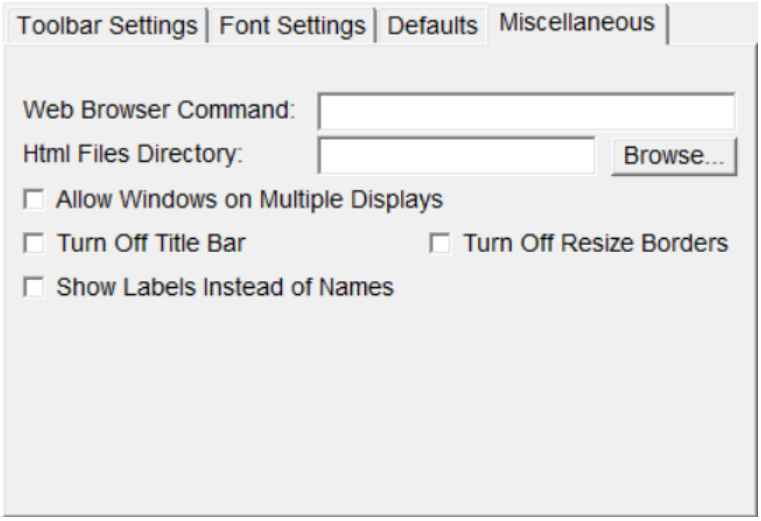
Default Y Position

The default Y position of the window.

Miscellaneous Tab

Use the fields on the Miscellaneous tab to specify how your vgi.vdef files access the Vsystem on-line Help files.

Figure4 - Miscellaneous Tab



Web Browser Command

Enter the command to use the launch the Web browser you want to use to display Help files. For example, "netscape" will cause the help command to open the Web pages with the Netscape Web browser.

Html Files Directory

Enter the location of the Vsystem HTML Help files. Leave this field blank to use the Help files located in the default Vsystem documentation HTML directory, or you can specify the directory location of the HTML files if they were placed in a different directory. Or, use the Browse button to locate the Help files on your machine.

Allow Windows on Multiple Displays

Select this option if you want to allow a window to go into multiple displays.

Turn Off Title Bar

Select this option to turn off the title bar on the top of all Vsystem graphics applications.

Turn Off Resize Borders

Select this option to turn off the window resize borders on all Vsystem graphics applications.

Db_access

The db_access utility enables you to modify the VALUE field for any channel in any database, subject to standard database protection mechanisms.

To run Db_access, enter the following at the system prompt:

```
db_access [qualifiers] [parameters]
```

In addition to channels, Db_access can also specify fields by using

```
@fieldname
```

where fieldname is the name of the field to be modified. For example,

```
db_access db::chan1@label "test"
```

writes "test" to the LABEL field of the db::chan1 channel.

Parameters

The `db_access` command line can include an optional single `db_access` command. Only single-action commands (those not requiring user input) can be included. The inclusion of `db_access` commands on the command line facilitates the use of `db_access` in command procedures. For example,

```
db_access DB::CHAN_1 5.0
```

sets the database channel CHAN_1 to the value 5.0 in the database DB.

```
db_access DB::STRING_CHAN """turn out the lights"""
```

sets the database channel STRING_CHAN to the value "turn out the lights."

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `db_access` command:

formatfile

```
formatfile=<filespec>
```

Use the `formatfile` qualifier to specify a file that contains a list of fields to be printed out for the channel. See the appropriate file for your operating system below for an example file that contains all database fields.

WIN `c:\vsys\bin\all_fields.dat`

LINUX `/vsys/bin/all_fields.dat`

VMS `vsys_root:[exe]all_fields.dat`

The order in which the fields are listed becomes the order in which they are shown in db_access. The format file may include text lines enclosed in quotation marks; db_access prints these text lines at the location between fields, as a method for labeling grouped fields.

nooutput

Use the `nooutput` qualifier to perform a single put to the database without db_access printing anything out. This qualifier is useful in command procedures. For example:

```
db_access -nooutput demo_db::demo:real_out1 1
```

Using Db_access Interactively

If no parameters are specified on the command line, you will be prompted for a channel name and an optional channel value. The channel name should be in the form:

```
<database name>::<channel name>
```

Note Wildcard characters can be used in the channel name, but they are not allowed in the database name.

If no database name is specified, db_access assigns the default database, RT_DATABASE , as the database name. If no channel value is specified after the channel name, Db_access shows the current value of the channel. The display of the value indicates that Db_access was able to find and access the channel successfully. If Db_access is unable to access the channel, it will return to the Db_access prompt to indicate an error in the channel specification; most likely, either the database or the channel was incorrectly specified. Another possible issue is that the database did not contain the specified channel, or that access to the database was denied.

If you provide a value along with the channel name, Db_access puts that value into the channel. The value entered is assumed to be in external units. You can enter integer values in decimal form only.

If the channel carries an array of values, db_access allows you to write values into each place in the array. If you enter an array channel along with a value at the db_access prompt, db_access prompts you with the following message:

```
Batch Mode {yes, no}?
```

If you respond yes , all values in the array are overwritten with the supplied value. If you give any other response, db_access prompts you for each array value, one at a time, with the first value entering the array at the lowest position.

Caution If, in the course of entering values, you press the ENTER key without entering a value, the remainder of the array is overwritten with the original value.

Example

```
CACTUS>db_access
Enter channel name ( . to quit )
<channel name> [optional value to be written]
cactusdb::demo:real_in1
DEMO:REAL_IN1
upper_alarm = DEFAULTED
lower_alarm = DEFAULTED
upper_warning = DEFAULTED
lower_warning = DEFAULTED
hi_equip_lim = 0.000000
```

```
low_equip_lim = 0.000000
hi_disp_lim = 0.000000
low_disp_lim = 0.000000
convert_slope = 0.000000
convert_offset = 0.000000
format_code = (F8.4)
units_code = volts
delta = 0.000000
size = 1
element external value
[0] -152.432434
--------------------------------
Enter channel name ( . to quit )
<channel name> [optional value to be written]
.
exiting...
```

Note that because the alarm and warning limits were not defined for the channel when it was created, Vgen inserted the default maximum and minimum values for the defined channel type, as noted by the text DEFAULTED . (For information about Vgen, see Chapter 2 Vgen in your Vsystem Vaccess Concepts Guide.)

```
CACTUS>db_access "demo:real_in1 100"
DEMO:REAL_IN1
upper_alarm = 200.000000
lower_alarm = -200.000000
upper_warning = 80.000000
lower_warning = -80.000000
hi_equip_lim = 0.000000
low_equip_lim = 0.000000
hi_disp_lim = 0.000000
low_disp_lim = 0.000000
convert_slope = 0.000000
convert_offset = 0.000000
format_code = (F8.4)
units_code = volts
delta = 0.000000
size = 1
element external value
[0] 100.000000
```

Db_capture

The db_capture utility is a graphical interface program that enables you to take a "snapshot" of the current channel values in a database, so that you can restore them later.

To start db_capture, enter the following at the system prompt:

```
db_capture [qualifiers] [parameters]
```

Parameters

The db_capture utility allows up to 128 [channellist] parameters, which can be either channel names or input files that contain lists of channels. These parameters are the channel sources that the utility monitors. If you leave the [channellist] parameter blank, the window starts up with no channels in its channel list. You can specify multiple input files on the command line.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `db_capture` command:

comment

Use the `comment` qualifier to enter text describing the snapshot file. This text appears in the Comment field of the Snapshot Save and Restore Utility dialog and as the first line in the file of a snapshot. For example:

```
db_capture -snap -comment="first phase"
```

confirm

Use the `confirm` qualifier if you want a confirmation message to appear for each restore request.

dir

Use the `dir` qualifier to specify the directory and the extension from which to scan or store the snapshot files. You can specify an extension for the file if you don't want the default extension .snp. The directory and extension you enter with this qualifier appear in the Filename field of the dialog. For example:

```
db_capture -dir=c:\db\message\*.snp -file=snap.snp chan1, file1, ...
```

file

Use the `file` qualifier to specify the filename that appears in the Filename field of the Snapshot Save and Restore Utility dialog. This file will contain the snapshot. You can specify an extension for the file if you don't want the default extension .snp. Note that if you specify a file extension with the dir qualifier, this extension is used for the file.

To use the file qualifier, you also need to specify channel names or input files in the parameter list. For example:

```
db_capture -file=snap.snp channel1, file1, ...
```

You can also use the wildcard asterisk (*) character in this parameter list. For example, if you want to capture all of the channels in the local database FACILITY_DB , enter

```
db_capture -file=snap.snp FACILITY_DB::*
```

If you have a specific list of channels that is rather long, you may want to create a simple ASCII file containing a list of channel names. Make sure that the file contains only one channel name per line, with no extra lines. Suppose, for example, a file called channellist.txt contains the following two lines:

```
FACILITY_DB::BINARY*
CACTUSDB::*
```

To use this file as an input list, enter:

```
db_capture -file=snap.snp channellist.txt
```

This command records all channels in FACILITY_DB that start with the word BINARY and all of the channels in the CACTUSDB database. The values in the channel list are stored in the file snap.dat when you click the Snap button on the Snapshot Save and Restore Utility dialog.

height

Use the `height` qualifier to specify the height, in pixels, of the Snapshot Save and Restore Utility dialog when it appears. The minimum height is 100.

match

Use the `match` qualifier to specify whether the filename and the comment in the Filename and the Comment fields match the selected entry in the list box area.

restore

To restore the contents of the snapshot file to the database (that is, returning the database values to those when the snapshot was taken), enter the restore qualifier and the name of the file you want to restore. For example:

```
db_capture -restore -file=snap.snp -dir=c:\db\message
```

where c:\db\message is the directory and snap.snp is the file containing the snapshot. If you want to change any values in the snap.dat file, edit the file with a standard text editor.

Caution Restoring a snapshot of the database causes instant changes in the channel values of the database. Make sure that any instruments attached to the saved channels can handle these jumps.

You can use the `file` and `dir` qualifiers with the `db_capture -restore` command.

snap

To use the `snap` qualifier, enter the name of the output file and the channels you want to capture. For example:

```
db_capture -snap -dir=c:\db\message\*.snp -file=snap.snp channel1, file1
```

You can use the `comment`, `dir`, and `file` qualifiers with the db_capture -snap command.

title

Use the `title` qualifier to specify a title in the title bar of the Snapshot Save and Restore Utility dialog.

width

Use the `width` qualifier to specify the width, in pixels, of the Snapshot Save and Restore Utility dialog when it appears. The minimum width is 100.

x

Use the `x` qualifier to define where the upper left corner of the Snapshot Save and Restore Utility dialog starts on the x-axis.

y

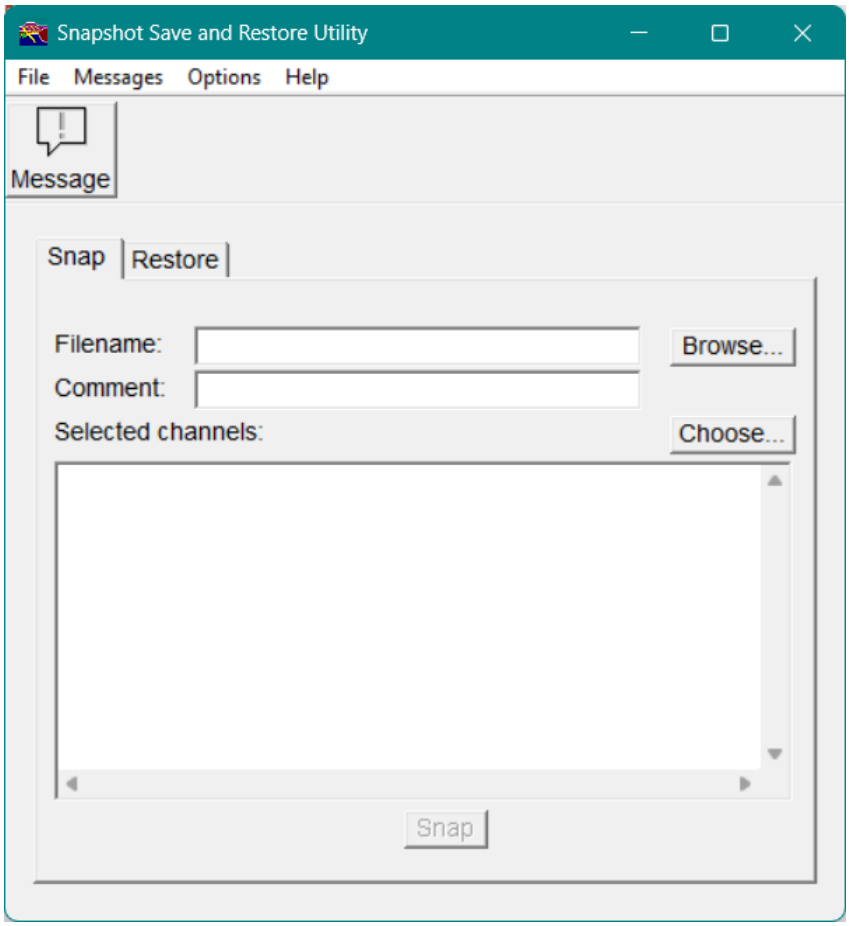
Use the `y` qualifier to define where the upper left corner of the Snapshot Save and Restore Utility dialog starts on the y-axis .

Snapshot Save and Restore Utility

After you start db_capture, the Snapshot Save and Restore Utility dialog appears, as shown in Figure 5.

You can use the Snapshot Save and Restore Utility dialog to record the current channel values of a database and then restore them later. The following section describes how to use the dialog to perform these tasks.

Figure5 - Snapshot Save and Restore Utility dialog



Snapping the Channel Values in the Database

Use the options on the Snap tab to perform a snapshot of the channel values in a database.

Filename

Enter the name of the file containing the snapshot information. You can specify an extension for the file if you don't want to use the default extension .snp. You can also use the Browse button to select a file to record.

Comment

Enter text to describe the snapshot file selected. This text then appears as the first line in the snapshot file.

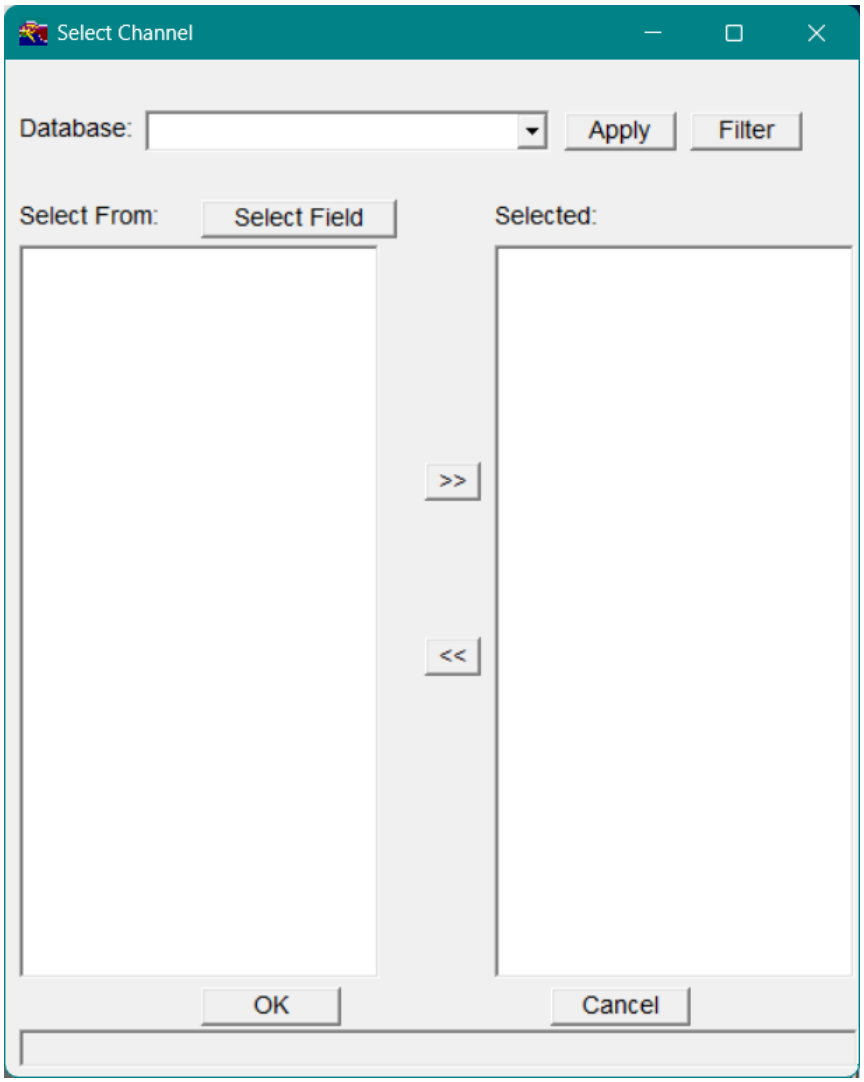
Selected channels

This list box displays the channels selected for snapshot.

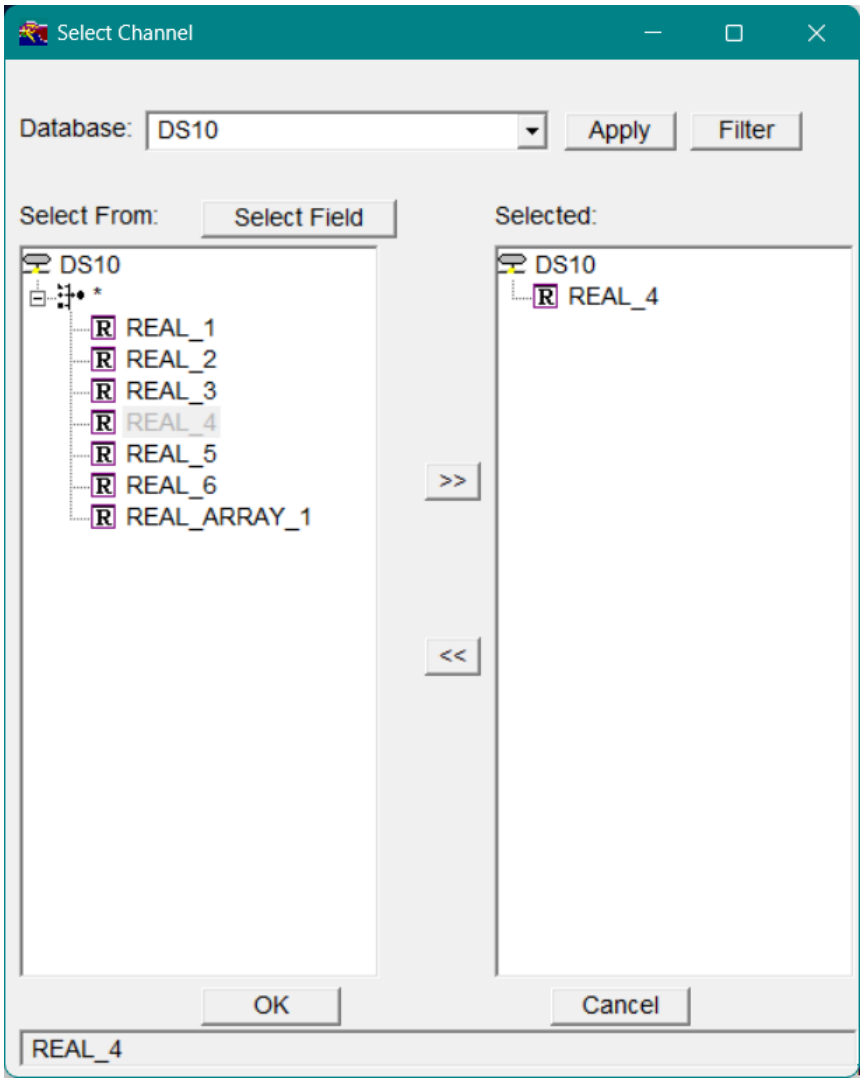
To change the channels listed in the Selected channels list box

Click the Choose button (see See Snapshot Save and Restore Utility dialogSee Snapshot Save and Restore Utility dialog).

Figure6 - Select Channel dialog



The Select Channel dialog appears, as shown in Figure 1-2. In the Select Channel dialog, you can move channels between the Select From list box and the Selected channels list box.



In the Database field, enter the database name.

-or-

Click the down arrow to select a database name.

Select the channels you want to snap and click the >> button.

The channels you select now appear in the Selected list box.

Click OK .

The selected channels now appear in the Selected channels list box of the Snapshot Save and Restore Utility dialog (See Snapshot Save and Restore Utility dialogSee Snapshot Save and Restore Utility dialog).

You can also click the Filter button on the Select Channel dialog to further specify which channels to display.

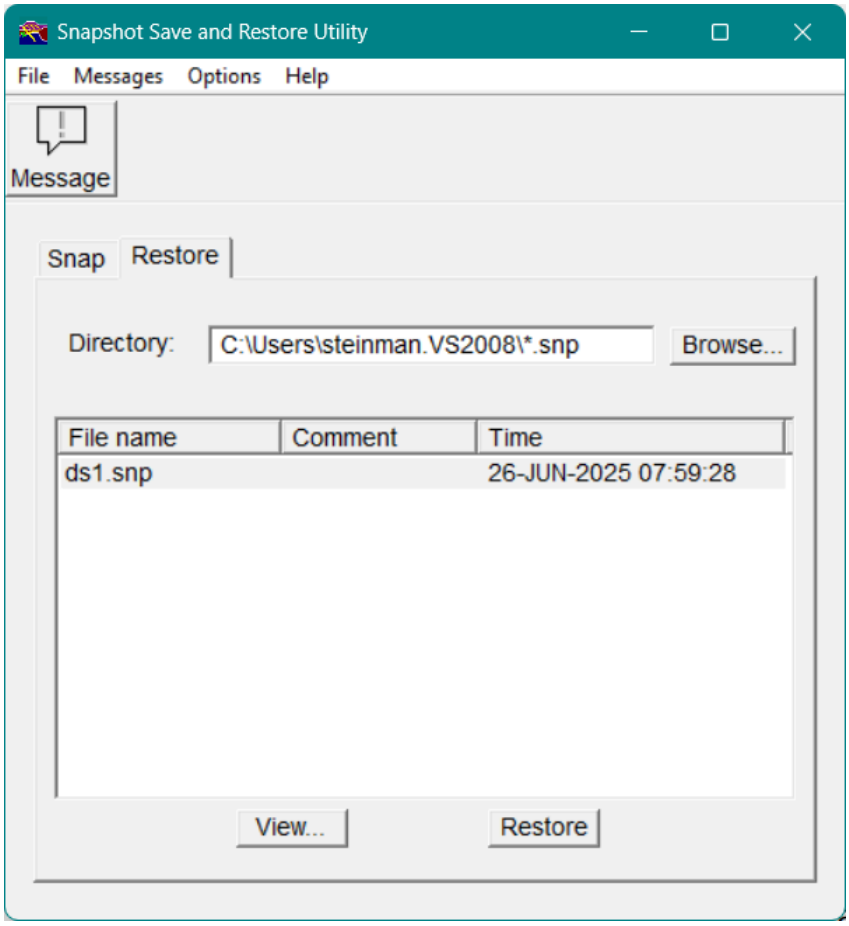
After specifying which channels to snap, click the Snap button below the list box on the Snapshot Save and Restore Utility dialog to start the snapshot.

> ***Note*** Although you can highlight a channel in the Selected channels list box by clicking on it, this has no effect on which channel(s) will appear in the snapshot; all of the channels displayed in the list box are automatically selected for the snapshot.

Restoring the Channel Values in the Database

On the Restore tab of the Snapshot Save and Restore Utility dialog, shown in Figure 7, specify the information to restore the channel values of a database.

Figure7 - Restore tab



Directory

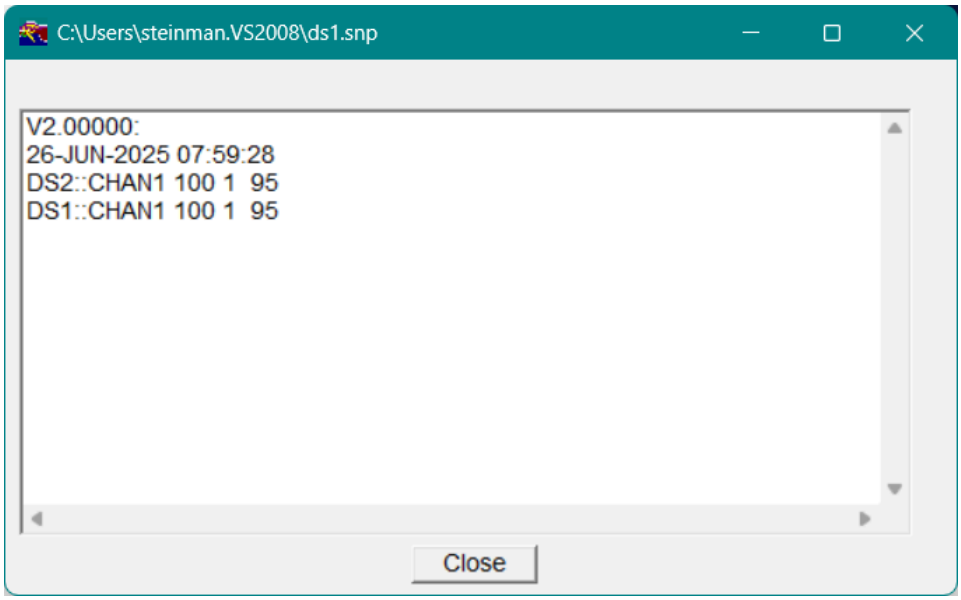
Enter the directory path (with the extension .snp) for the files in which you want to scan and store the snapped channel values. (You can enter a different file extension if you do not want to use the default extension .snp.)

When entering the directory path, you can also use the Browse... button to make your selection.

When the directory path is specified for the snapshot files, the list box displays their filenames, along with their comments and creation times.

View

To review the snapped data in a snapshot file, select the filename in the list box and then click View .



Restore

After reviewing the snapped data, click Restore to restore it.

Confirming the Restore

On the Options menu in the menu bar, select Confirm Restore to enable the confirmation of each intended restore. If you select this option, each time you try to restore the recorded channel values, a Confirmation dialog appears, in which you can confirm the restore.

Viewing the Messages

Use the Message List window to view the error messages reported during your operation of the Snapshot Save and Restore Utility dialog.

To open the Message List window

• On the toolbar, click the Message button.

-or-

• On the Messages menu in the menu bar, select Open Message Window .

The Message List window appears automatically whenever a message is generated. If your process is generating numerous messages and you want to avoid the interruptions, you can elect to keep this window closed; it will continue to log messages as they are generated.

To keep the Message List window closed

• Click the Keep Closed option at the bottom of the Message List window.

-or-

• On the Messages menu in the menu bar, select Keep Closed.

Db_dump

The db_dump utility enables you to get a listing of all of the database fields for a channel. This utility is useful for checking the fields in a database when you are working from a nongraphics terminal (one on which the db_view utility cannot be brought up).

To run db_dump

• Enter the following at the system prompt:

```
db_dump <parameter>
```

Parameter

Db_dump allows a channel to be specified at the command line; if no channel is specified, db_dump prompts you for a channel.

Example

```
C:\Vista\vsys\examples\databases>db_dump
Enter channel name ( . to quit )
->mydb::demo:real_out1
Channel Name: DEMO:REAL_OUT1
Local Channel Index: 35
Label: demo real out channel 1
Type: REAL
In: 0

------- Alarm fields and Limits ---------
Alarms Enabled: 1
Alarm Priority: 0
Alarms Acknowledged: 0
Upper Alarm: 200.0
Lower Alarm: -200.0
Upper Warning: 80.0
Lower Warning: -80.0
Upper Equipment Limit: 1000.0
Lower Equipment Limit: -1000.0
Upper Display Limit: 250.0
Lower Display Limit: -250.0
Alarm Reference Chix: 0
Alarms Delayed: 0

------- Conversion Fields ---------

Convert Enabled: 0
```

```
Slope: 0.0
Offset: 0.0

------- Hardware Fields ---------

------- String Fields ---------

------- Miscellaneous fields ---------
No Survey: 0
I/O Error: 0
Checkpoint Enabled: 0
Soft: 0
Clipping Enabled: 1
Automatic: 0
Constant Enabled: 0
Survey Count: 0
Fast Survey: 0
Slow Survey: 0
Area: 0
Subarea: 0

Delta:: 0.0
Format: (f5.1)
Units: VOLTS

Press return to continue

Timestamp Update Disable: 0
Size: 1
Element External Value Internal Value
[0] 0.0 0.0

---------------------------------
Enter channel name ( . to quit )
->
```

Db_forcex

The db_forcex utility enables an appropriately privileged user to force a process on the computer to run its exit handler and terminate its current image.

To run db_forcex

• Enter the following at the system prompt:

```
db_forcex <pid>
```

Parameter

You must include the process ID (PID) of the process you want to exit.

> ***Note*** VMS On OpenVMS, the PID should be in HEX format; on all other platforms, the PID is in decimal format. For more information on the behavior of this utility on OpenVMS, refer to the OpenVMS system service reference manual for a description of the SYS$FORCEX service. A user with WORLD privilege can force any process to exit; a user with GROUP privilege can force another process in the same group to exit; and nonprivileged users can force only a process of their own to exit.

Example

```
CACTUS>db_forcex 21400064
Forced Exit for process : 21400064
```

Db_map

Once you have created an intermediate database file (.idb) with Vgen, you must first map it into memory with the Db_map utility before you can access it. A shared memory area, containing the event memory (emem) and dynamic memory (dmem), is created for the database during this process. The initial database values are then put into the database. (For information about Vgen, see Chapter 2 Vgen in your Vsystem Vaccess Concepts Guide.)

The Db_map utility also checks whether the database has any running processes connected to it. If there are processes connected, Db_map displays a list of the tasks and exits without mapping the database. You can use the rundown and stop qualifiers to run down and terminate the running processes. You can also use vinfo to check the running status of the processes.

To run Db_map

- Enter the following at the system prompt:

```
db_map [qualifiers] [idb filename] [database name]
```

You may also use the db_map graphical interface, which you can access through one of the following commands:

```
db_map -win
```

-or-

```
db_map
```

The db_map graphical interface is documented in Db_map Graphical Interface.

Parameters

You may use the following parameters with the `db_map` command:

idb filename

The `idb filename` parameter is a required parameter that specifies the intermediate database file, including the directory location if the file is not located in the current directory.

database name

The `database name` parameter is a required parameter that specifies the name of the database; you will use this name when referring to this database from now on in any of the Vsystem tools.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `db_map` command.

diag

Use the `diag` qualifier to print out diagnostic messages during the mapping process. These messages indicate errors that occur while mapping.

dmem

```
dmem=<value>
```

Use the `dmem` qualifier to specify the amount of dynamic pool memory, in kilobytes, to increase or decrease in the database. For example,

```
db_map -dmem=100
```

increases the dynamic pool memory by 100KB.

This qualifier should not be necessary in most cases. This qualifier will increase or decrease any dmem value specified in Vgen.

emem

```
emem=<value>
```

Use the emem qualifier to specify the amount of extra event memory, in kilobytes, to increase or decrease in the database. For example,

```
db_map -emem=-100
```

decreases the event memory by 100KB.

The amount of event memory needed depends on how many processes are connecting to the database at one time and how many events are generated. This qualifier will increase or decrease any emem value specified in Vgen.

help

Use the `help` qualifier to display a summary of usage, parameters, and qualifiers for `db_map`.

initvalue

If an initial value is not defined for a channel in its .adb file, and if the channel is not an array, binary, or string channel, use the `initvalue` qualifier to put a value of 0 to the channel; any alarms, handlers, and conversions will be updated. If this qualifier is not specified, the put to the channel will not occur and any alarms, handlers, and conversions will not be updated. The channel, however, will still be initialized with a 0.

This qualifier has no effect if an initial value is given for a channel in its .adb file.

lock_mem

Use the `lock_mem` qualifier to lock the database shared memory into memory. To use this qualifier, you need to have P_SYSOPS privilege on Linux or similar privilege on other platforms.

By locking the database into memory, you prevent the database from being swapped out to disk. A large, locked database, however, may affect the performance of your machine, because programs may have to be swapped more often.

log

Use the log qualifier to create a log file named `database.log` in the Vsystem hostname log directory. This log file contains diagnostic information, including db_map details, process connections and disconnections, db_rundown details, and so on.

nohandler

Use the `nohandler` qualifier to prevent the handlers from running during the mapping process.

override

Use the `override` qualifier to prevent db_map from checking for processes connected to the database during the mapping process.

restart

Use the `restart` qualifier to use the current database files, if they exist, when restarting processes; otherwise, restart has no effect. This feature is useful if you have to reboot the computer, but want to keep the database image as it was previously mapped.

recover

Use the `recover` qualifier to use a copy of the database backup files as the current database.

rundown

Use the `rundown` qualifier to enable the system to disconnect any processes that are connected to the database before mapping the database. However, this qualifier does not exit the running processes. You need to use the `stop` qualifier to terminate the processes.

stop

Use the `stop` qualifier, together with the `rundown` qualifier, to terminate any processes connected to the database you want to map. If the rundown qualifier is unspecified, the stop qualifier has no effect.

timerd

When you map a database, you can also use the `timerd` qualifier to start the timer daemon. The `timerd` qualifier can take an optional value, in milliseconds, specifying the delay time. (For detailed information about the timer daemon, refer to Starting the Delayed Events Timer Daemon in your Vsystem Vaccess Concepts Guide.) For example:

```
db_map -timerd=500 database.idb my_db
```

user_files

```
user_files=<shared library>
```

Use the `user_files` qualifier to specify shareable libraries that contain user-defined handlers and conversion routines. This qualifier can be used instead of specifying the `VDB_USER_FILES` environment variable.

version

Use the `version` qualifier to report the version number for db_map.

win

Use the `win` qualifier to start the db_map graphical interface (see below).

Db_map graphical interface

To run the Db_map graphical interface

- Enter the following at the system prompt:

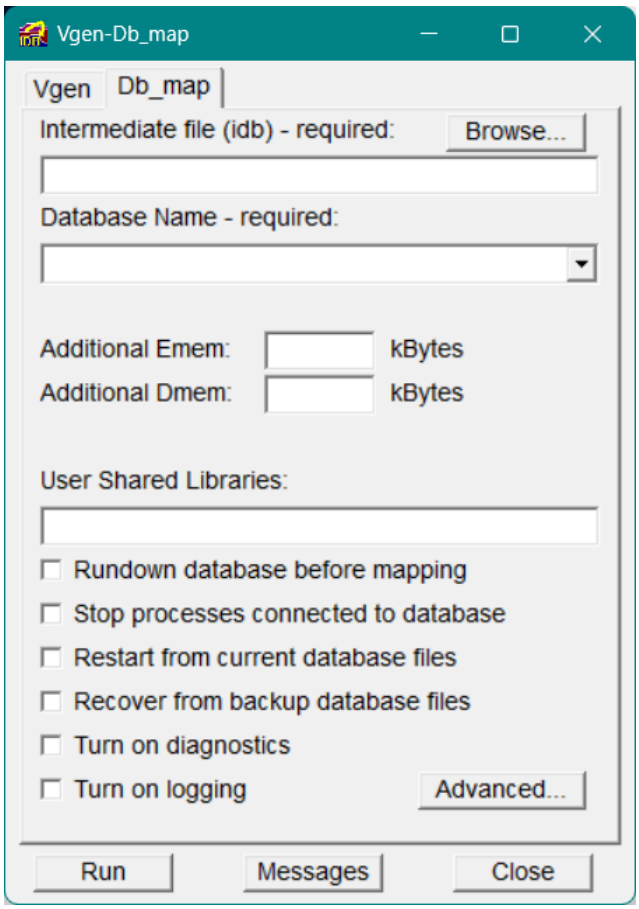
```
db_map -win
```

-or-

```
db_map
```

The Vgen-Db_map dialog appears with the Db_map tab selected, as shown in Figure 8.

Figure8 - Vgen-Db_map dialog



Intermediate file (idb)

Enter the name of the intermediate file, with extension .idb, that you want to map into memory. Or use the Browse button to select one from the list.

Database Name

Enter the name you want to use to identify the database once it has been mapped.

Additional Emem

Enter a value, in kilobytes, for the amount of additional event memory (emem) to reserve in the database. The amount of emem needed depends on how many processes are connecting to the database at one time and how many events are generated. This option will increase or decrease any emem value specified in Vgen. For more information on emem, refer to Event Memory (emem) in your Vsystem Vaccess Concepts Guide.

Additional Dmem

Enter a value, in kilobytes, for the amount of additional dynamic memory (dmem) to reserve in the database. This parameter should not be necessary in most cases, but, if used, will increase or decrease any dmem value specified in Vgen. For more information on dmem, refer to Dynamic Memory (dmem) in your Vsystem Vaccess Concepts Guide.

User Shared Libraries

Enter the name of the shared memory area that will be created by Db_map. This shared memory area, containing the event memory (emem) and dynamic memory (dmem), is created where the database is stored. The initial database values are then put into the database.

Rundown database before mapping

Select this option to run down any processes connected to the database before mapping it. This option does not cause the running processes to exit, however. You need to use the Stop processes connected to database option (below) to terminate the processes. (For more information on running down a database, see See Db_rundown See Db_rundown.)

Stop processes connected to database

Select this option, in conjunction with the Rundown database before mapping option (above), to terminate processes that are connected to the database. If the Rundown database before mapping option is not selected, this option has no effect.

Restart from current database files

If there are current database files, you can use this option to restart processes with these files; otherwise, this option has no effect. This feature is useful if you have to reboot the computer, but want to keep the database image as it was previously mapped.

Recover from backup database files

Select this option to use a copy of the database backup files as the current database.

Turn on diagnostics

Select this option to print out diagnostic messages during the mapping process.

Turn on logging

Select this option to create a log file named database.log in the Vsystem hostname log directory. This log file contains diagnostic information, including Db_map details, engine connections and disconnections, db_rundown details, and so on.

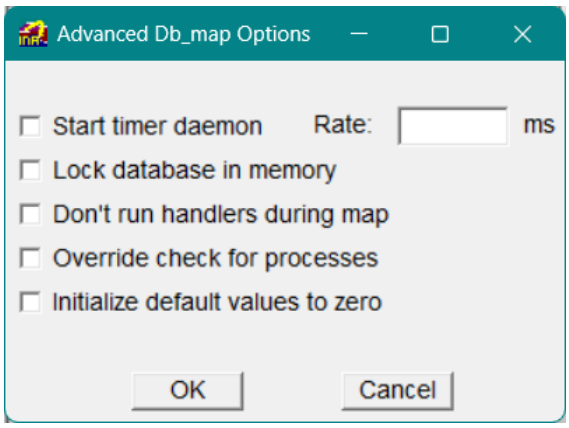
Messages Button

Use the Messages button at the bottom of the Vgen-Db_map dialog to access the Message List window.

Advanced

To display more advanced options, click the Advanced button in the lower right-hand corner of the Db_map tab. The Advanced Db_map Options dialog opens, as shown in Figure 9.

Figure9 - Advanced Db_map Options



Start timer daemon

Select this option to start the timer daemon, which is used for delayed alarms. (For detailed information about the timer daemon, refer to Starting the Delayed Events Timer Daemon in your Vsystem Vaccess Concepts Guide.)

Rate

Enter a value, in milliseconds, for the timer daemon rate. This rate determines how frequently the timer daemon monitors delayed alarms.

Lock database in memory

Select this option to lock the database shared memory into memory. To use this qualifier, you need to have P_SYSOPS privilege on UNIX, or similar privilege on other platforms.

> ***Note*** By locking the database into memory, you prevent the database from being swapped out to disk. A large, locked database, however, may affect the performance of your machine, because programs may have to be swapped more often.

Don't run handlers during map

Select this option to prevent the handlers from running during the mapping process.

Override check for processes

Select this option to prevent Db_map from checking for processes connected to the database during the mapping process.

Initialize default values to zero

If an initial value is not defined for a channel in its .adb file, and if the channel is not an array, binary, or string channel, select this option to put a value of 0 to the channel; any alarms, handlers, and conversions will be updated. If this option is not selected, the put to the channel will not occur, and any alarms, handlers, and conversions will not be updated. The channel, however, will still be initialized with a 0.

This option has no effect if an initial value is given for a channel.

Db_rundown

The db_rundown utility executes the rundown procedure for the specified database. If no database is specified, the default database, RT_DATABASE, is run down.

To start db_rundown

• Enter the following at the system prompt:

```
db_rundown [qualifiers] <database name>
```

You must have DELETE access to a database in order to run down that database. The rundown procedure forces the removal of all images currently connected to the database. Events and wakes scheduled on channels in the database are also removed. You can start a new program to connect to the database, or you can create a new version of the database using the `vgen` and `db_map` commands.

Parameter

You must specify the name of the database to be run down on the command line.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `db_rundown` command:

help

Use the `help` qualifier to display a summary of usage, parameters, and qualifiers for `db_rundown`.

output

`output=<filename>`

Use the `output` qualifier to specify a file in which db_rundown will save its text output.

stop

Use the `stop` qualifier to cause any programs connected to the database to exit. The database is then removed from memory and the database files are deleted.

task

`task=<task number>`

Use the `task` qualifier, in conjunction with the stop qualifier, to accept a task number, as given by `vinfo -task <database>`, and stop the given task.

For example:

```
> vinfo -task alarmdb
Database:ALARMDB
TASK TABLE
       NUM_SLOTS:256
```

```
TASK Process ID Group/Link Status
   1   19721       1    0  Running db_view alarmdb::volume_left
       EVENT_QUE:size= 100, head= 0, tail= 0
   2   19713       2    0  Running valarm_viewer alarmdb::volume_left
       EVENT_QUE:size= 1000, head= 2, tail= 2
```

```
> db_rundown -task=1 -stop alarmdb
```

version

Use the `version` qualifier to report the version number for `db_rundown`.

Example

```
c:\program files\vista\vsys\bin>db_rundown /stop testdb
```

Note You can use `db_rundown` to ensure that no processes are connected to a database before you map a new version of the database. In addition, you can use db_map with the `rundown` and `stop` qualifiers to cause db_map to perform the rundown processing.

Db_show

The db_show utility enables you to look at various properties of a database without actually connecting to the database.

To start db_show

• Enter the following at the system prompt:

```
db_show [qualifiers] <database name>
```

Parameter

You must specify the name of the database whose information you want to view on the command line.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `db_show` command:

process

Use the `process` qualifier to show all of the processes connected to the database.

database

Use the database qualifier to show the characteristics of the database file.

all

Use the all qualifier to show both the processes connected to the database and the characteristics of the database file.

Example

```
W:\vaccess\test> db_show /process ds1
[TSK][Proc ID ][chix] PROCESS USER IMAGE
------------------------------------------------------------
[ 1][000000B1] [ 0] Steinman db_view ds1
[ 2][00000111] [ 0] Steinman db_show /process ds1
Done with exit cleanup.

W:\vaccess\test> db_show /database ds1

Database name: DS1
Number of channels: 35
Created: 4-NOV-1998 12:54:33.87
From the file: test.adb
Disk file: test.idb
Version: 4.0
User Version: 0
VDB memory: 111 512-byte blocks
```

```
AST memory: 595 512-byte blocks
Total number of ASTs: 1903
ASTs available: 1903
```

Db_view

The db_view utility provides you with a graphical interface to view all of the fields within a running database and to change the values in these fields. There are some fields you cannot change with Db_view. These include the channel name, channel type, and the array size.

To start db_view

• Enter the following at the system prompt:

```
db_view [qualifiers] [parameters]
```

Parameter

The parameter may be a comma-separated list of databases. If no parameter is specified and the `VDB_DB_LIST` environment variable is defined, db_view will automatically list the `VDB_DB_LIST` databases.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `db_view` command:

channel

`channel=<channelname>`

If you use the `channel` qualifier, Db_view opens with the specified channel in its window.

mem

Use the `mem` qualifier to display the Db_view Memory window, as shown in Memory Tab. This window enables you to review the dynamic (Dmem) and event (Emem) memory data.

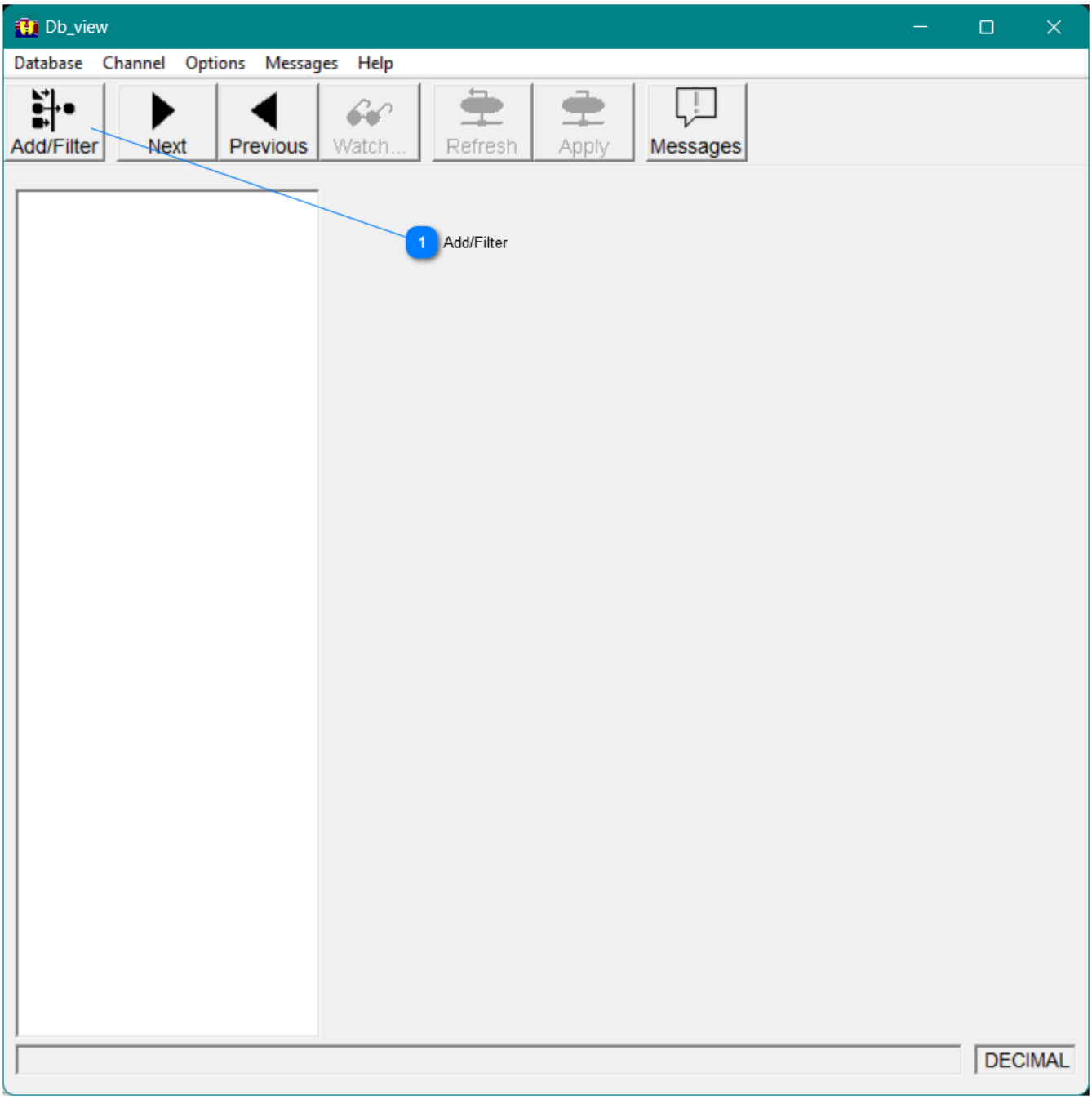
watch=<channelname>

Use the `watch=<chanelname>` qualifier to display a list of all changes made to selected channels. You can have multiple windows open at the same time.

Db_view graphical interface

If you start `db_view` without specifying the parameter, the db_view list box appears empty, as shown in Figure 10.

Figure10 - db_view default window



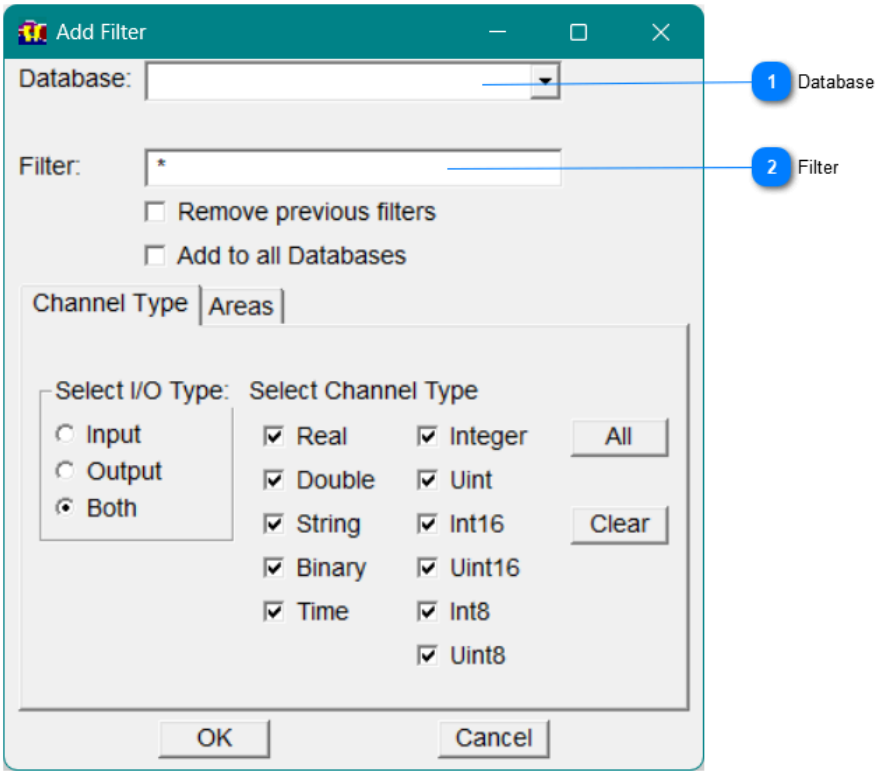
Opening the database

To use db_view, you need to open the database that contains the channels you want to review.

1 Add/Filter

To open a database, click the **Add/filter** button on the toolbar or in the **Database** menu, select **Add Database/Filter**. The Add Filter dialog appears, as shown in Figure 11.

Figure11 - Add Filter dialog



1 Database

In the Database field, enter the database name or select a database from the drop-down list.

2 Filter

Additional Filters can be applied to select only the channels of interest. For example, adding `r*` to the filter and selecting Real for the Channel Type, and the Add to all Databases, will bring up the channel list as shown in Figure 13.

Figure12 - Add/Filter Dialog

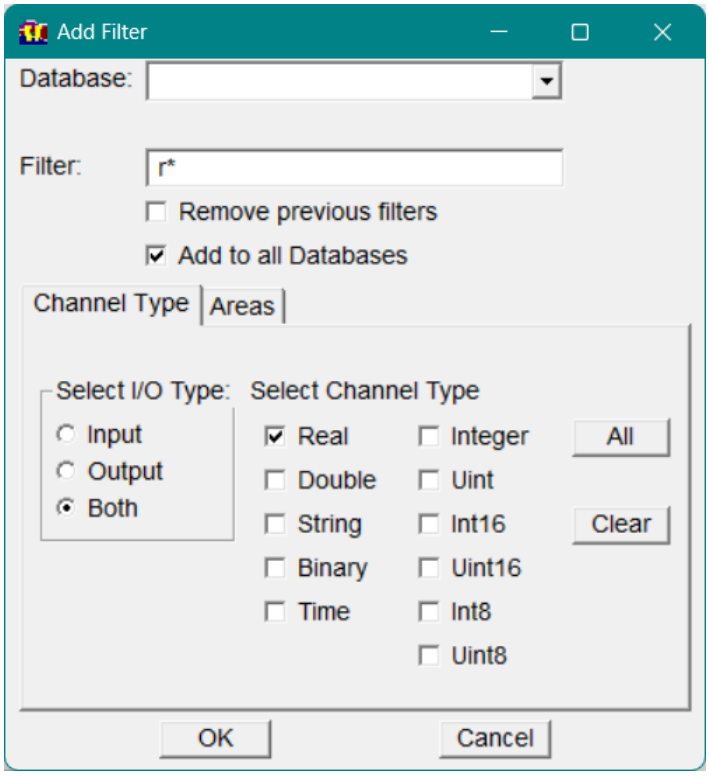
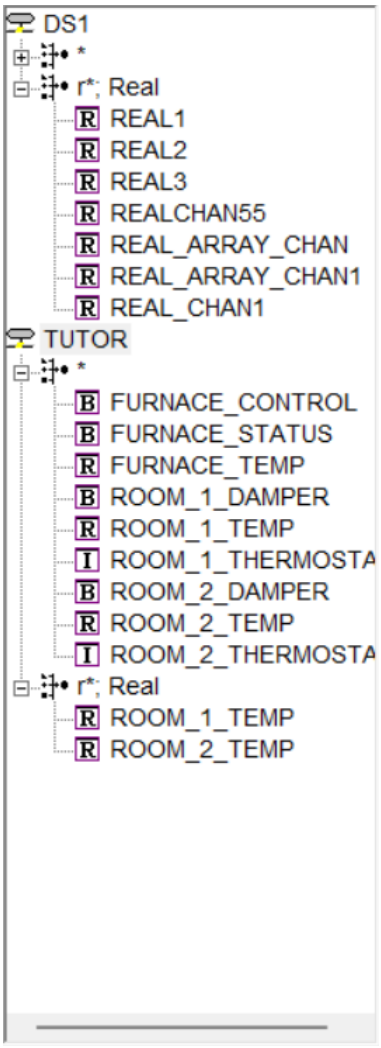


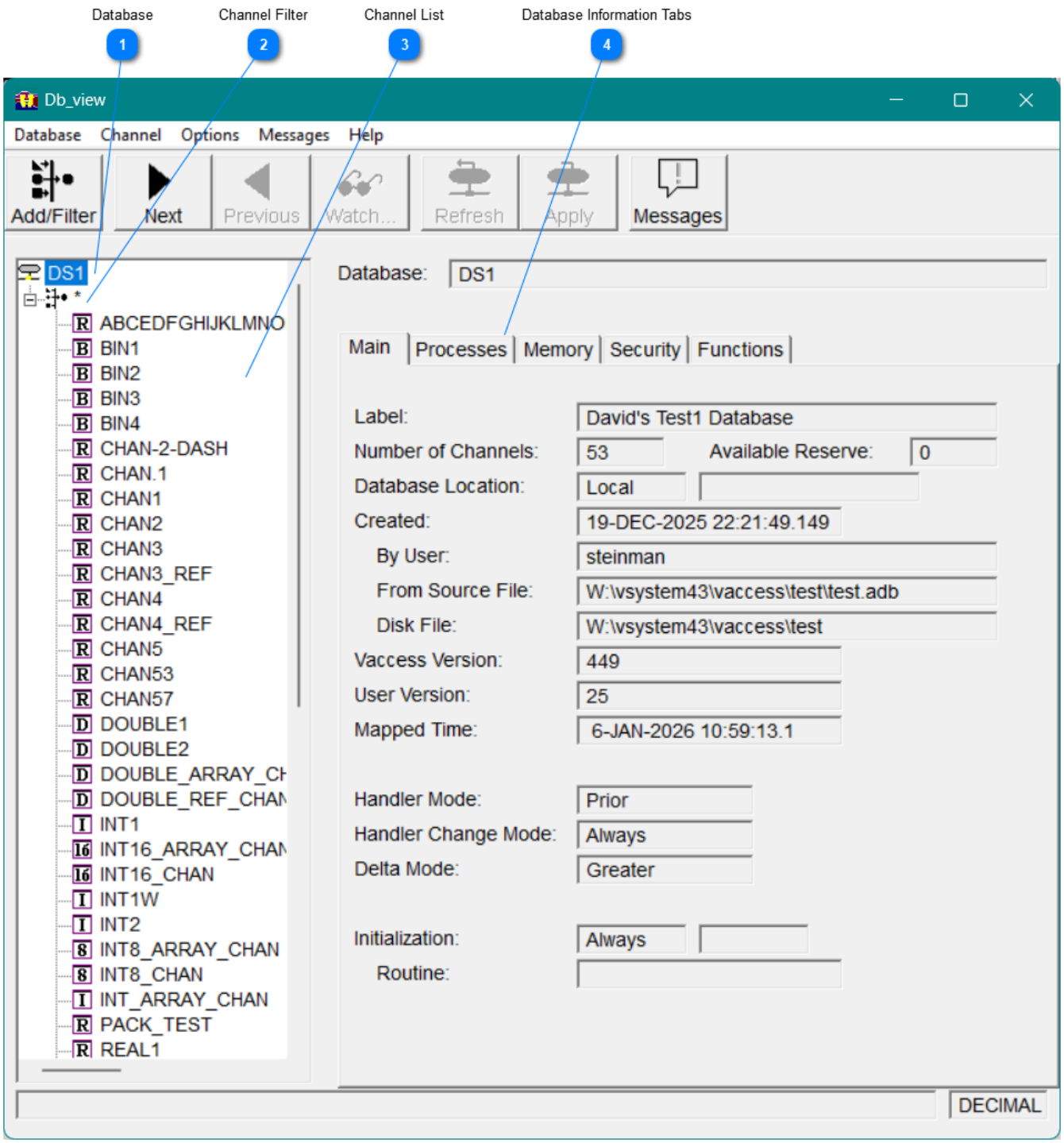
Figure13 - Channel List with filters applied



Viewing database information

When you select a database, the following information will be displayed.

Figure14 - Database Main Tab


Database
Channel Filter
Channel List
Database Information Tabs
1
2
3
4
Db_view
Database Channel Options Messages Help
Add/Filter
Next
Previous
Watch...
Refresh
Apply
Messages
DS1
*
ABCEDFGHIJKLMNO
BIN1
BIN2
BIN3
BIN4
CHAN-2-DASH
CHAN.1
CHAN1
CHAN2
CHAN3
CHAN3_REF
CHAN4
CHAN4_REF
CHAN5
CHAN53
CHAN57
DOUBLE1
DOUBLE2
DOUBLE_ARRAY_CH
DOUBLE_REF_CHAN
INT1
INT16_ARRAY_CHAN
INT16_CHAN
INT1W
INT2
INT8_ARRAY_CHAN
INT8_CHAN
INT_ARRAY_CHAN
PACK_TEST
REAL1
Database: DS1
Main
Processes
Memory
Security
Functions
Label: David's Test1 Database
Number of Channels: 53
Available Reserve: 0
Database Location: Local
Created: 19-DEC-2025 22:21:49.149
By User: steinman
From Source File: W:\vsystem43\vaccess\test\test.adb
Disk File: W:\vsystem43\vaccess\test
Vaccess Version: 449
User Version: 25
Mapped Time: 6-JAN-2026 10:59:13.1
Handler Mode: Prior
Handler Change Mode: Always
Delta Mode: Greater
Initialization: Always
Routine:
DECIMAL

1 Database

2 Channel Filter

3 Channel List

The icon indicates the channel type

4 Database Information Tabs

Select each tab to view more information,

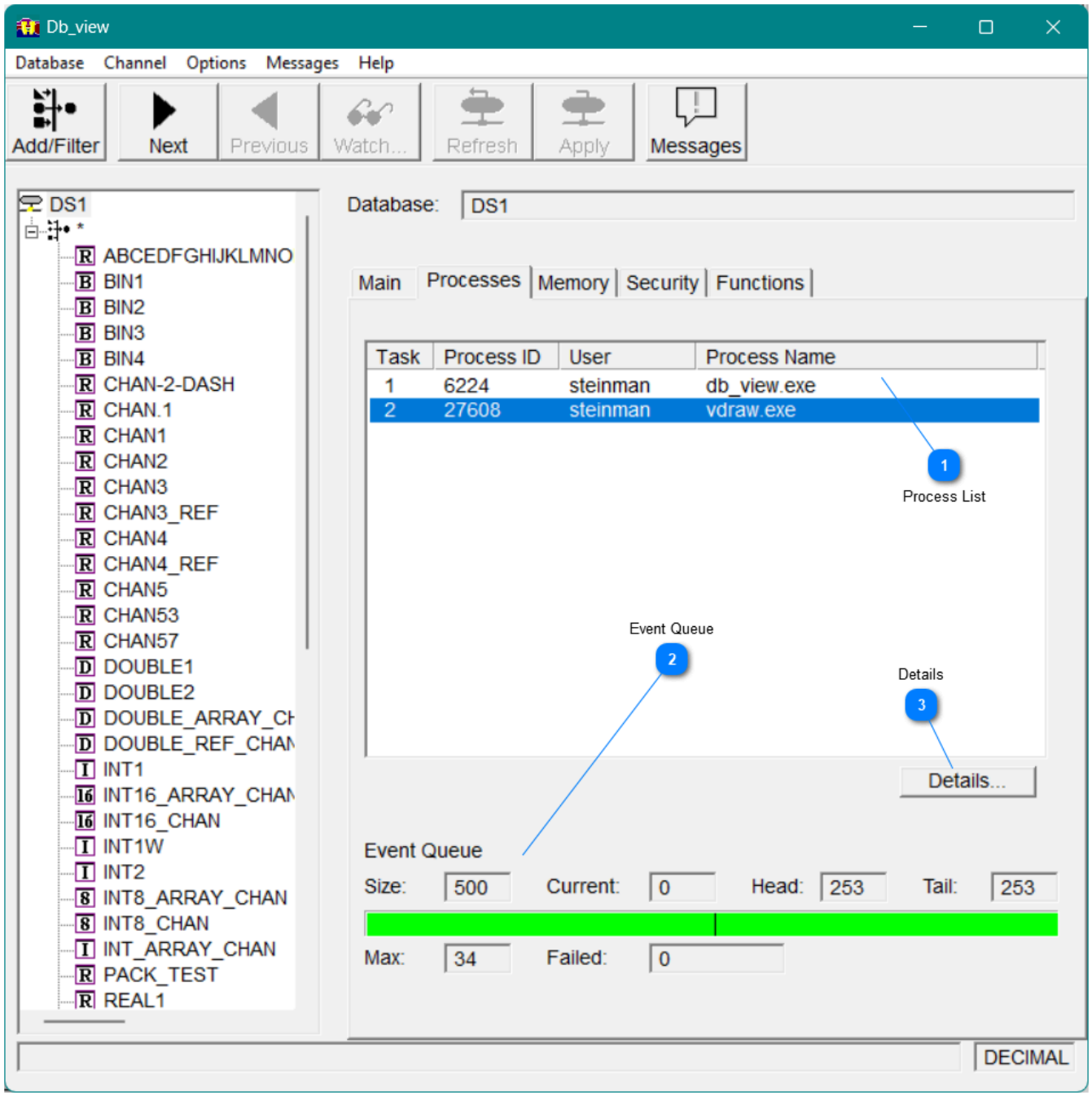
Main Tab

The Main tab displays general information about the database, as show in Figure 14.

Processes Tab

The Processes tab displays information about the processes connected to the database, as shown in Invalid cross-reference. To see more details about a process, select the process and then click the **Details** button at the bottom right-hand corner of the processes list box.

Figure15 - Db_view dialog Processes tab and Task Details window



1 Process List

This is the list of processes connected to the database.

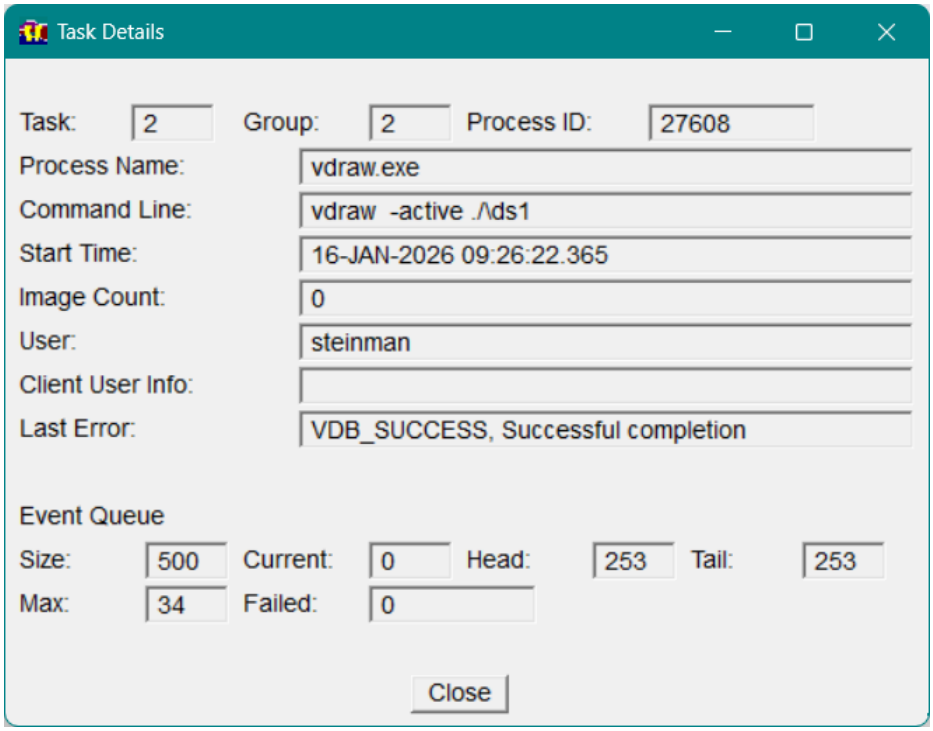
2 Event Queue

These fields display the event queue information of the selected database.

3 Details

Select Details to display more information about the selected process. Figure 16 shows the information.

Figure16 - Task Details

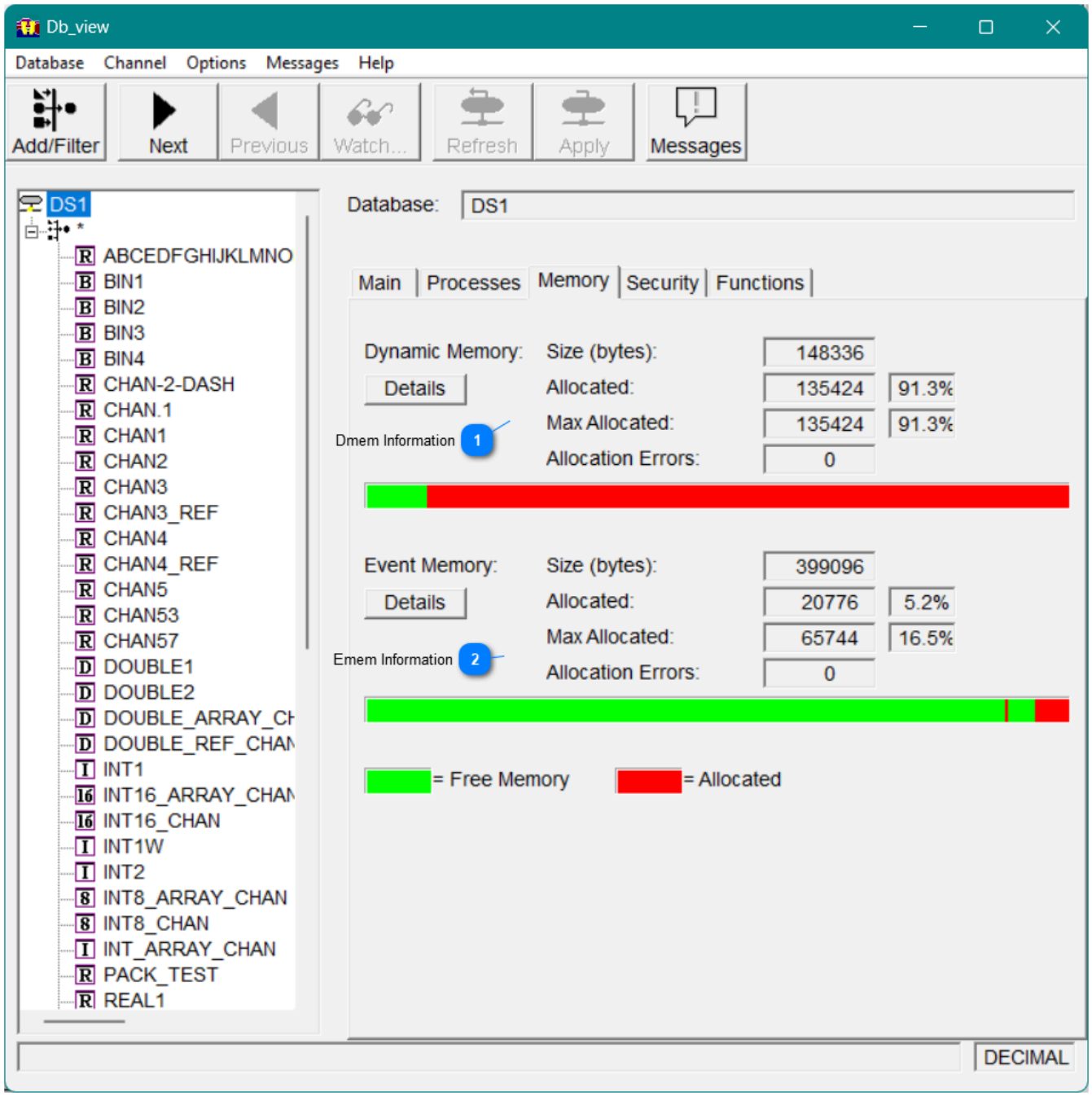


Memory Tab

The Memory tab displays the memory information for the database (including dmem and emem), as shown in Invalid cross-reference. This window also appears if you invoke Db_map with the mem qualifier and a database as the parameter.

The current amount of memory allocated is shown, along with the maximum that has been allocated at any one time. Any memory allocation errors that have occurred are also shown.

Figure17 - Db_view dialog Memory tab



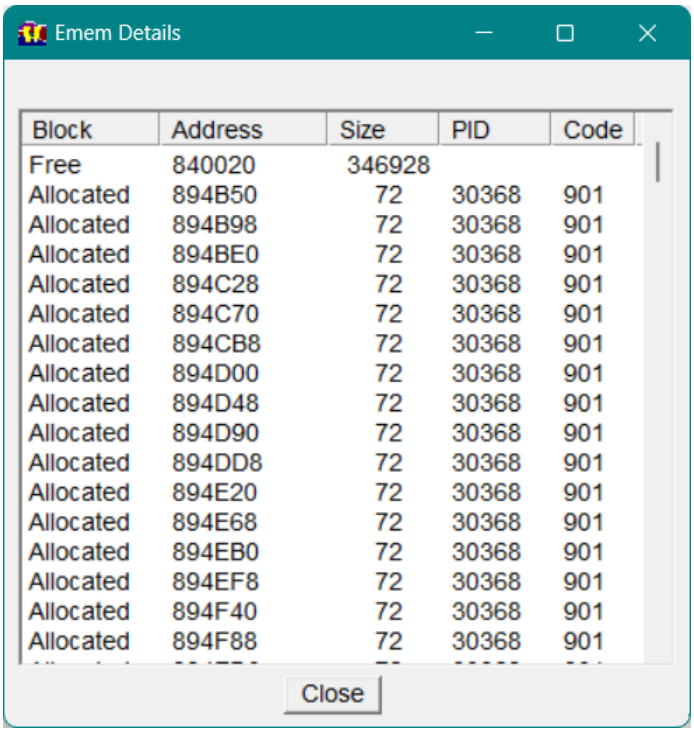
1 Dmem Information

Information for Dmem including errors and maximum allocated. Select the **Details** button for a list of the memory used.

2 Emem Information

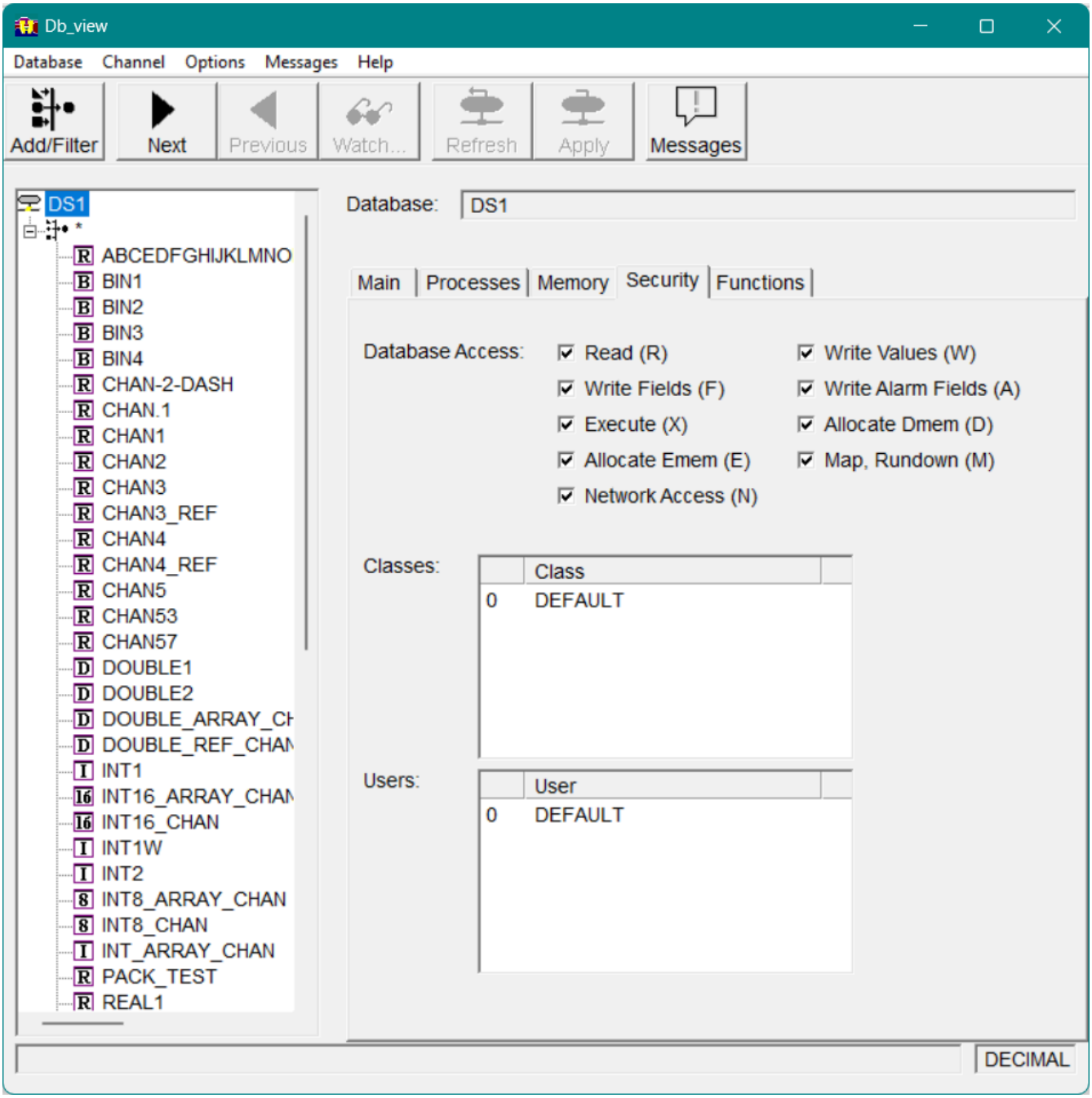
Information for Emem including errors and maximum allocated. Select the **Details** button for a list of the memory used. For example, if you click the **Details** button, the Emem Details window appears.

Figure18 - Emem Details



Security Tab

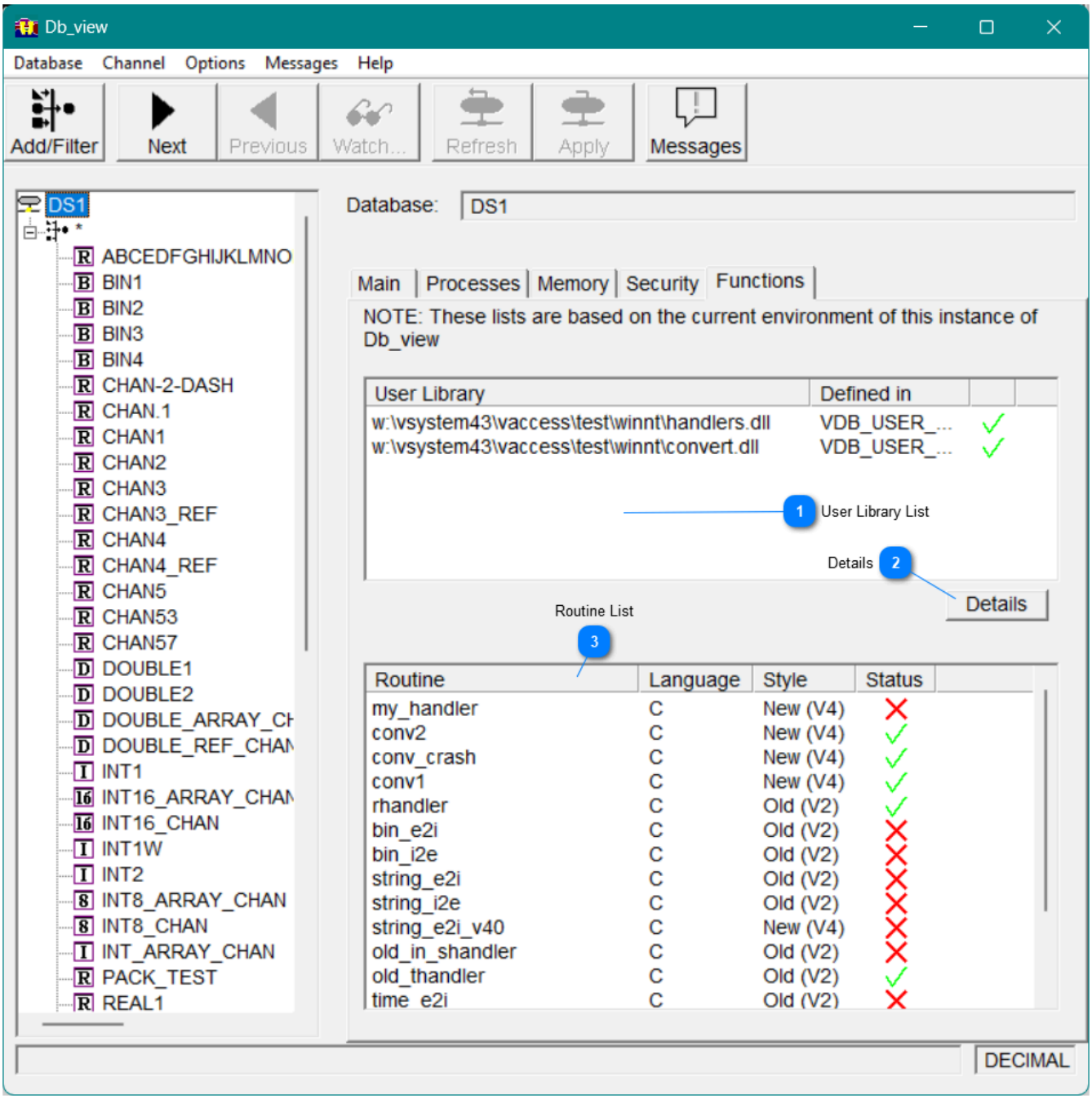
The Security tab displays the access information set for the database, including class and user information, as shown below.



Functions Tab

The Functions tab displays the handlers and conversion routines used in the database, as shown in Figure 19.

Figure19 - Functions Tab



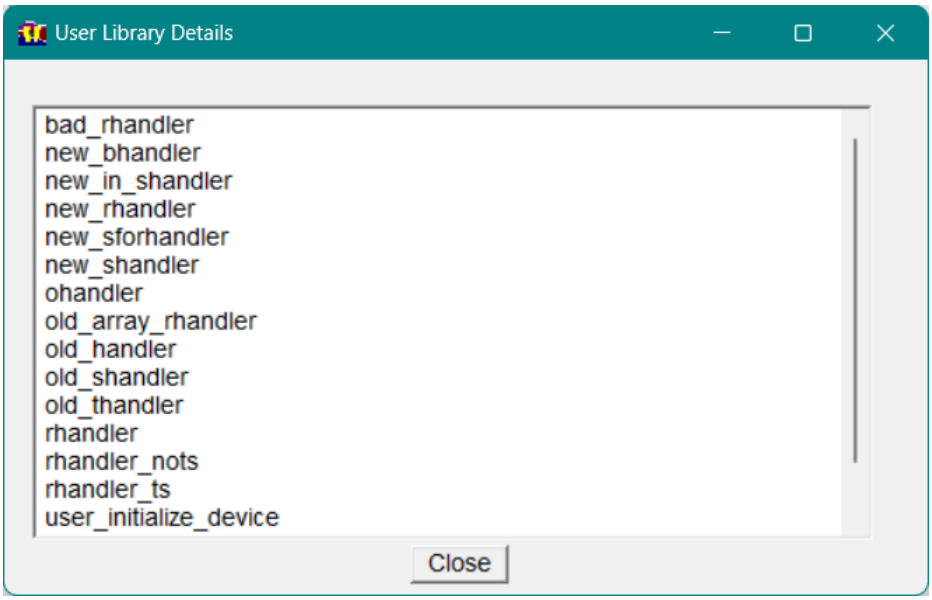
1 User Library List

The User Library list box displays the shared libraries that are being searched for the routines. Defined in display where the User Library was defined which could be the environment variable, VDB_USER_FILES, or in the database. A green checkmark ✓ indicates the library was found, while red x ✗ indicates the library was not found

2 Details

The Details button will display a window that list the routines that are found in the selected library. This can help find routines that where not found.

Figure20 - User Library Details



3 Routine List

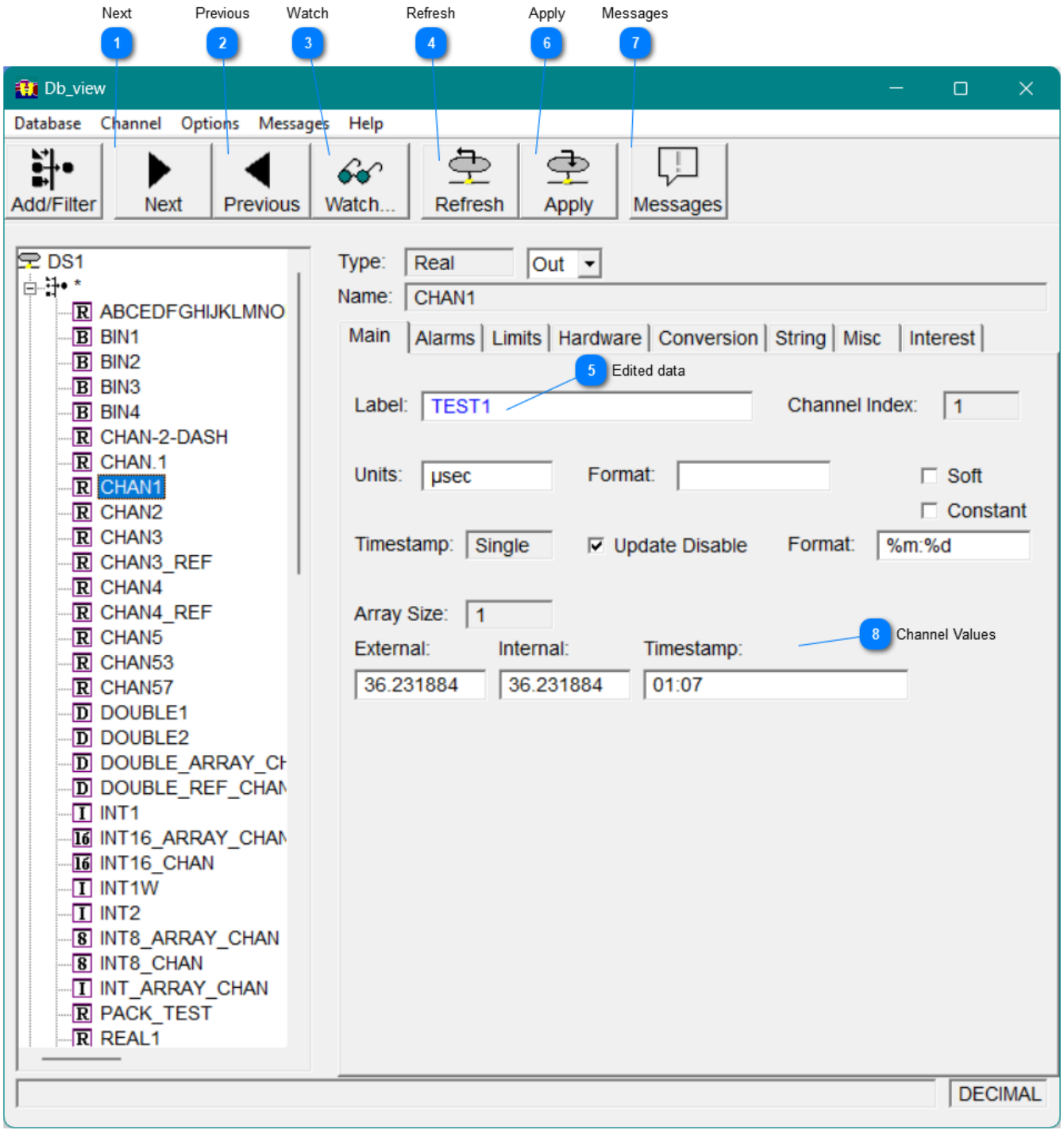
The Routine List displays each routine that has been configured in the database. A green checkmark ✓ indicates the routine was found, while red x ✗ indicates the routine was not found

> ***Note*** This information is based on the environment in which Db_view is running. If another process is run in a different environment, this data may not be valid.

Viewing channel information

Among the field values displayed for a channel are its channel type and name, as well as whether the channel is an input or output channel, as shown in Figure 21.

Figure21 - Channel Information


Next
Previous
Watch
Refresh
Apply
Messages
1
2
3
4
6
7
Db_view
Database
Channel
Options
Messages
Help
Add/Filter
Next
Previous
Watch...
Refresh
Apply
Messages
DS1
*
ABCEDFGHIJKLMNO
BIN1
BIN2
BIN3
BIN4
CHAN-2-DASH
CHAN.1
CHAN1
CHAN2
CHAN3
CHAN3_REF
CHAN4
CHAN4_REF
CHAN5
CHAN53
CHAN57
DOUBLE1
DOUBLE2
DOUBLE_ARRAY_CH
DOUBLE_REF_CHAN
INT1
INT16_ARRAY_CHAN
INT16_CHAN
INT1W
INT2
INT8_ARRAY_CHAN
INT8_CHAN
INT_ARRAY_CHAN
PACK_TEST
REAL1
Type:
Real
Out
Name:
CHAN1
Main
Alarms
Limits
Hardware
Conversion
String
Misc
Interest
5
Edited data
Label:
TEST1
Channel Index:
1
Units:
µsec
Format:
Soft
Constant
Timestamp:
Single
Update Disable
Format:
%m:%d
Array Size:
1
8
Channel Values
External:
Internal:
Timestamp:
36.231884
36.231884
01:07
DECIMAL

1 Next

To view the information of the next channel in the channel list, click the **Next** button or on the **Channel** menu, select **Next**.

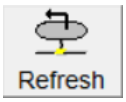
2 Previous

To view the information of the previous channel in the channel list, click the **Previous** button or on the **Channel** menu, select **Previous**.

3 Watch

Select Watch to view changes made to a channel by db_view and other programs. See Displaying changes to a channel using Watch.

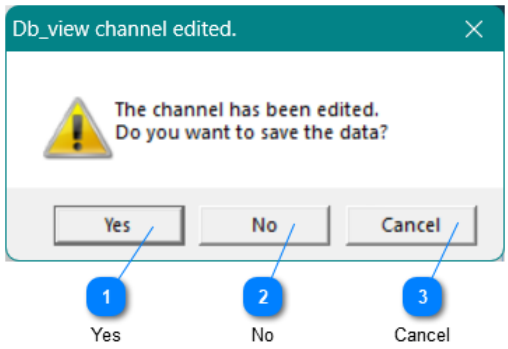
4 Refresh



Select **Refresh** on the toolbar or in the **Channel** menu, to view updated data in the channel or remove any edits that have not been saved.

5 Edited data

A field that has been edited will be shown in blue and the **Apply** button on the toolbar will be enabled. If you try to move to another channel, or select **Refresh**, the following message will be displayed.



1 Yes

Selecting Yes, will save the changes to the channel and then display the new channel.

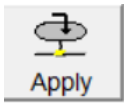
2 No

Selecting No, will not save the changes, but the new channel will be displayed.

3 Cancel

Selecting Cancel, will return you to the same channel and the edited data will still be there.

6 Apply



To apply changes to a channel, on the toolbar, click the **Apply** button or on the **Channel** menu, select **Apply**.

> ***Caution*** Once you have selected the **Apply** option or button, you cannot restore the previous data of the channel.

7 Messages

Displays the Messages window.

8 Channel Values

The information displayed beneath Array Size changes, depending on what type of channel you have selected.

Figure22 - Double Array Channel with Timestamps

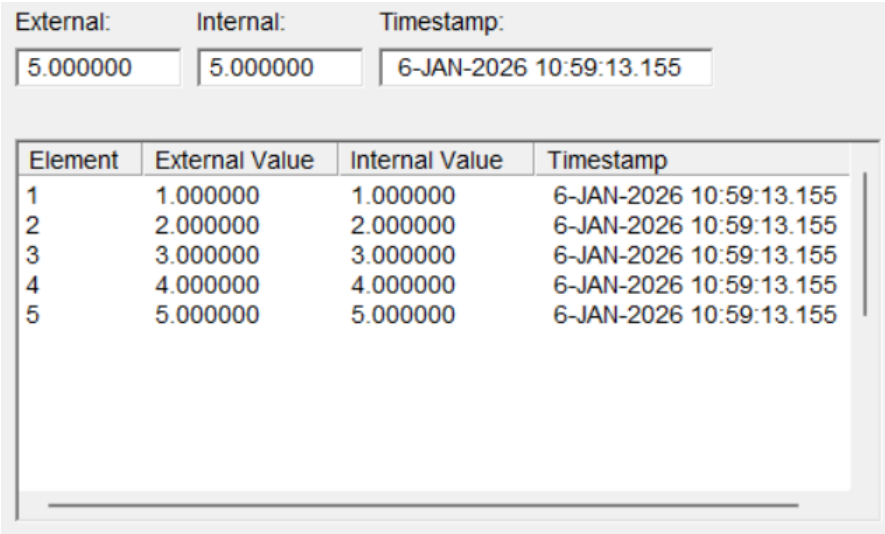


Figure23 - Integer Array Channel without Timestamps

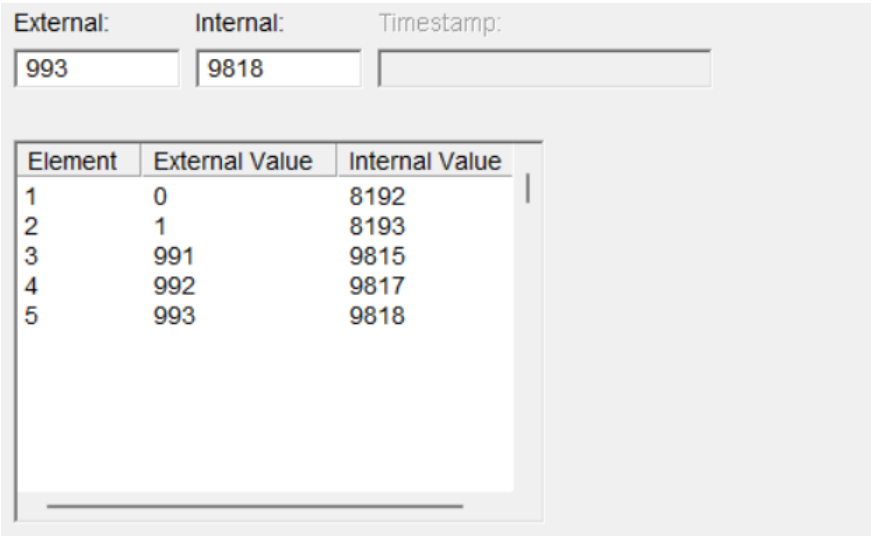
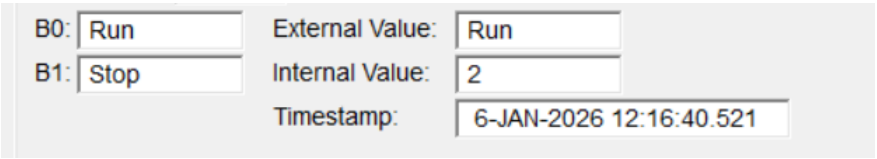


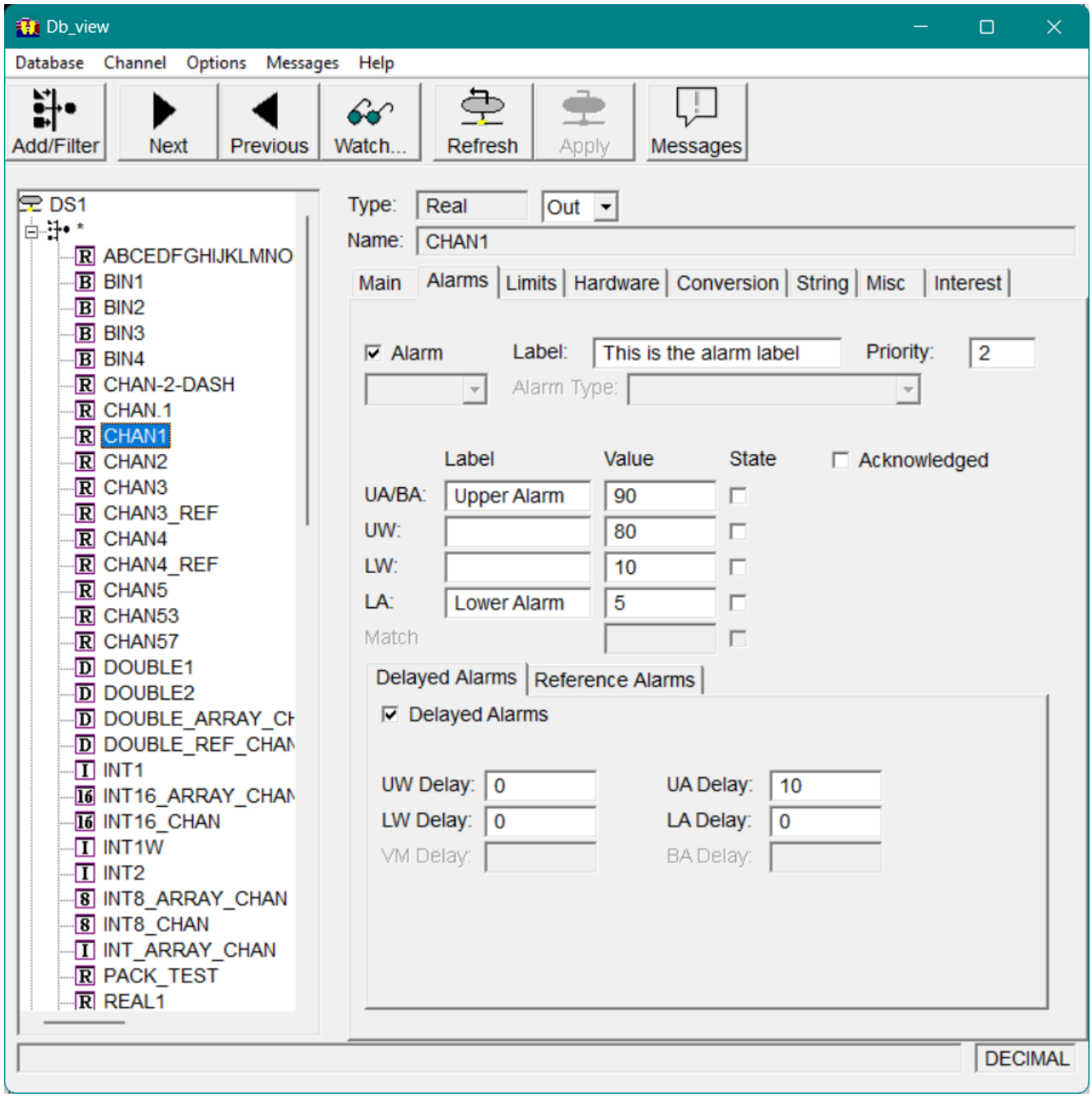
Figure24 - String Channel

Figure25 - Binary Channel



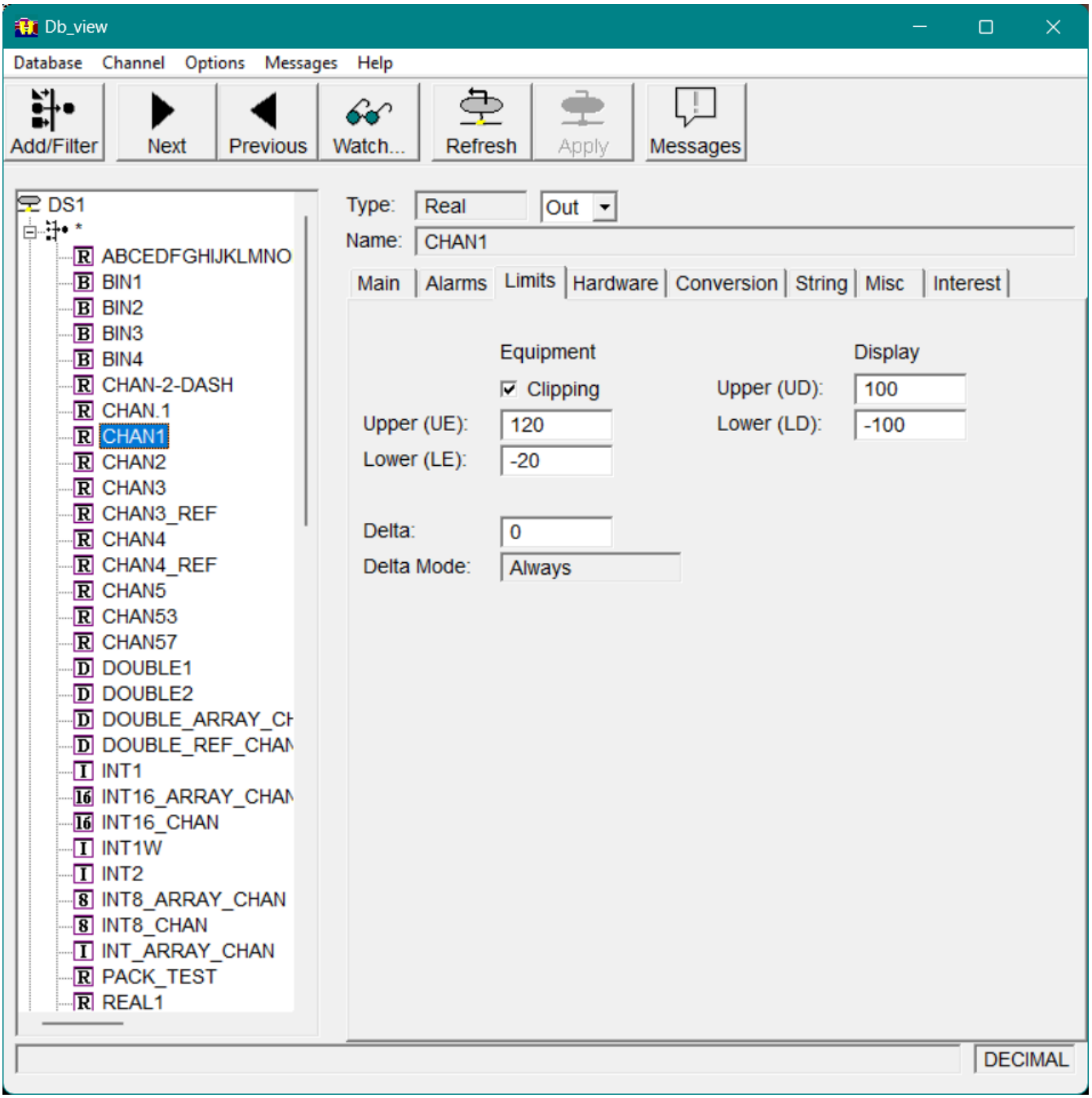
Alarms Tab

The Alarms tab displays the alarm information for the selected channel.



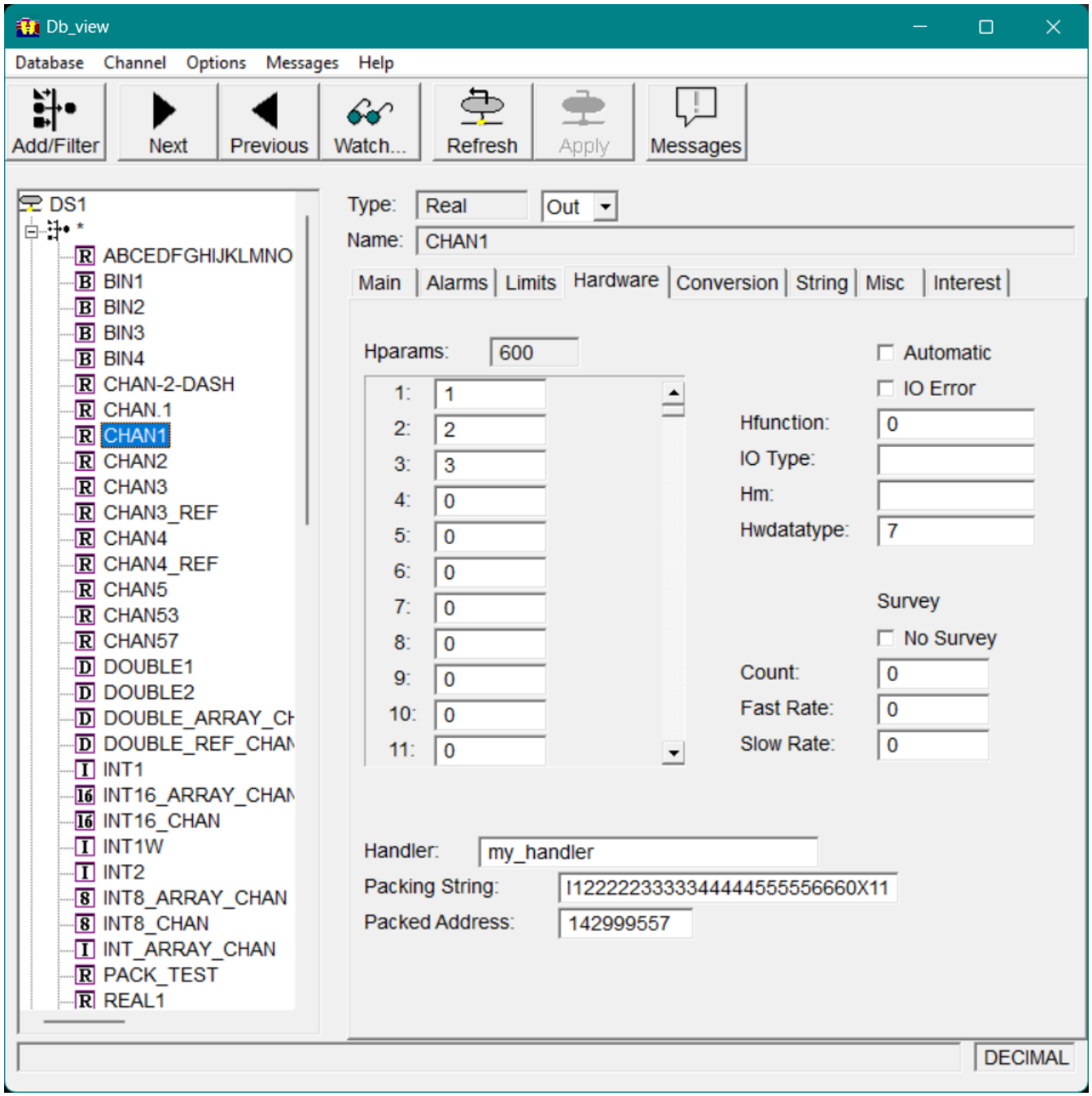
Limits Tab

Use the Limits tab to view the following limits for the selected channel.



Hardware Tab

The Hardware tab displays the following hardware values.



Hparams

The Hparams list box displays the hparams used to store address and configuration information for the hardware.

Automatic

If this option is selected, and if the selected channel is an output channel, whenever a put is performed on the channel, its handler is automatically called.

If the selected channel is an input channel, whenever a get is performed on the channel, its handler is automatically called.

Note that if Automatic is enabled, there should also be a handler specified for the channel (see See HandlerSee Handler).

Db_view dialog Hardware tab

IO Error

This option displays the I/O error status. If an I/O error is indicated, it is usually set by the return of a zero value from the handler of the channel.

Hfunction

This field shows the function to be executed by the handler subroutine of the selected channel. The hfunction is an integer value that can be used to issue different commands to a single handler.

IO Type

This field shows the I/O bus structure used for the selected channel. The only reserved type is CAMAC, which must be used for all CAMAC channels to properly store the CAMAC address. Any other definitions are at the discretion of the staff responsible for implementation.

Hm

This field shows the type of hardware associated with the selected channel, as determined by the hardware module type defined by the system designer. This information is used in hardware communications, as determined by the application.

Hwdatatype

This field shows the data type the hardware is expecting.

Survey

The following field s show the survey fields for the channel.

No Survey

If this option is selected, Vscan will not read the hardware assigned to the selected channel. (For information about Vscan, see Chapter 2 See Vscan.)

Count

The value in this field shows the SURVEY_COUNTER field for a channel. The function of this keyword is assigned by the application designer.

Fast Rate

The value in this field shows the FAST_SURVEY_COUNT field for a channel. The function of this keyword is assigned by the application designer.

Slow Rate

The value in this field shows the SLOW_SURVEY_COUNT field for a channel. The function of this keyword is assigned by the application designer.

Handler

This field shows the name of the user-written subroutine that will be called for the selected channel. This option performs the appropriate hardware I/O for controlling or reading your device.

If See Automatic is enabledSee Automatic, the handler will be automatically called as follows:

• If the selected channel is an output channel, its handler is automatically called whenever a put is performed on the channel.

• If the selected channel is an input channel, its handler is automatically called whenever a get is performed on the channel.

Packing String

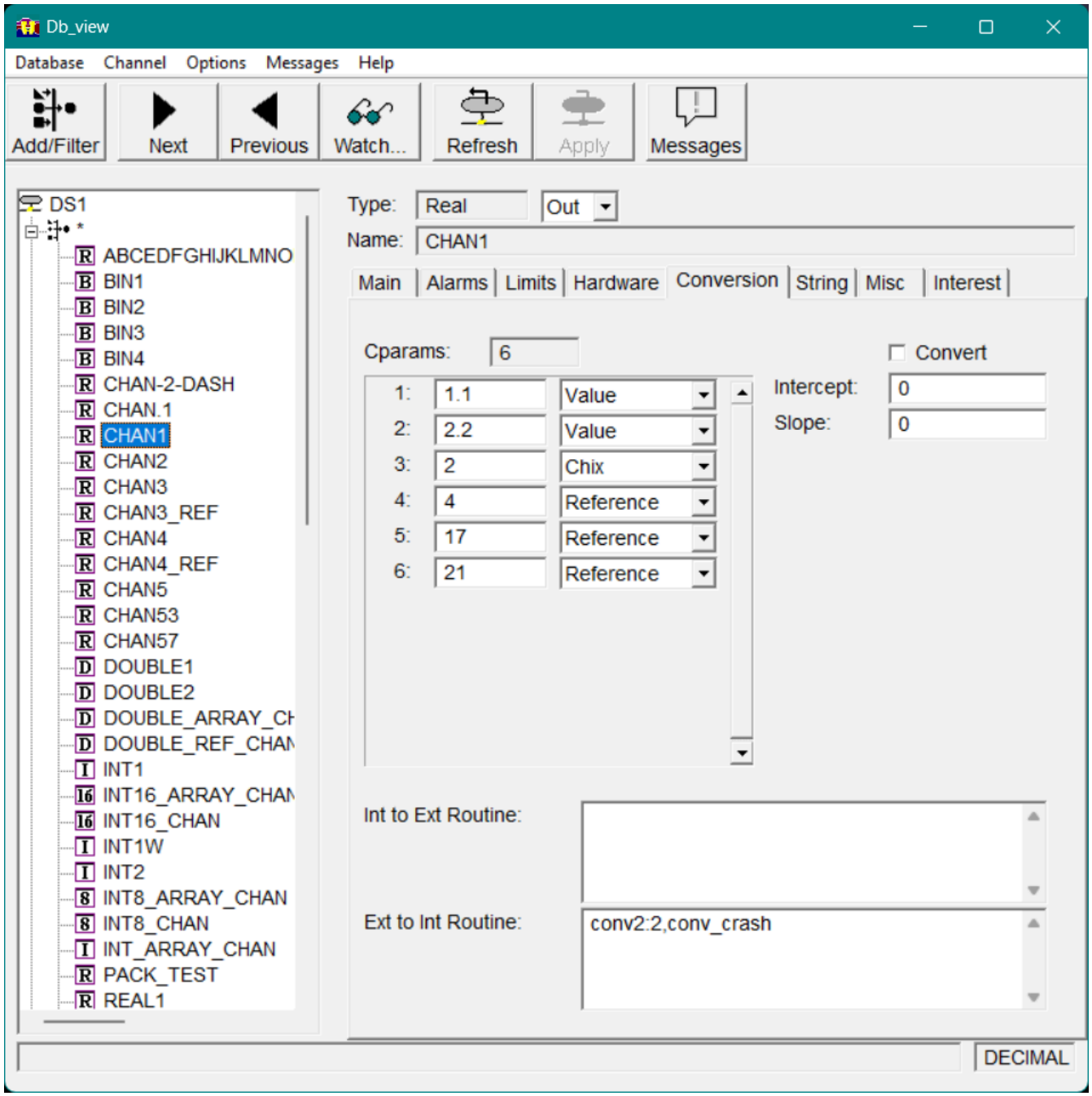
This field shows the name of the packing algorithm used to generate packed hardware addresses.

Packed Address

This field shows the resultant hardware address generated by the packing string for the channel.

Conversion Tab

The Conversion tab displays the following conversion values.



Cparams

If cparams have been defined for the channel, the Cparams list box contains several columns of information. The numbers in the left-hand column show the cparams used to store conversion information for the user-written conversion routines specified by the Vgen keywords inttoext and exttoint . The maximum number of cparams allocated for the channel depends on the largest cparamX keyword specified. (For information about specifying cparams and definitions of the inttoext and exttoint keywords, refer to Chapter 1 Vgen Keywords in your Vsystem Vaccess Reference Guide.)

The value of each cparam is displayed in the center column of the Cparams list box. In the column to the right of this value is a drop-down list specifying the type of cparam, which can be one of the following:

value	A floating-point value.
chix	A channel index.
reference	A channel index that is passed the external value of the specified channel when used for the conversion routine.

Convert

When selected, this option enables the user-specified conversion mechanism on the selected channel. For real and double channels, a linear conversion is used if the slope value is nonzero for the selected channel; otherwise, no conversion is made.

Intercept

The value specified for this option is the value for B in the conversion formula E=MI+B where:

E is the external value (the value seen by the user).

I is the internal value (the value stored in the database).

M is the slope provided with the slope keyword.

Slope

If the value specified for this option is nonzero, linear conversions will be performed. The value specified for this option is the value for M in the conversion formula E=MI+B where:

E is the external value (the value seen by the user).

I is the internal value (the value stored in the database).

B is the intercept provided with the intercept keyword.

Int to Ext Routine

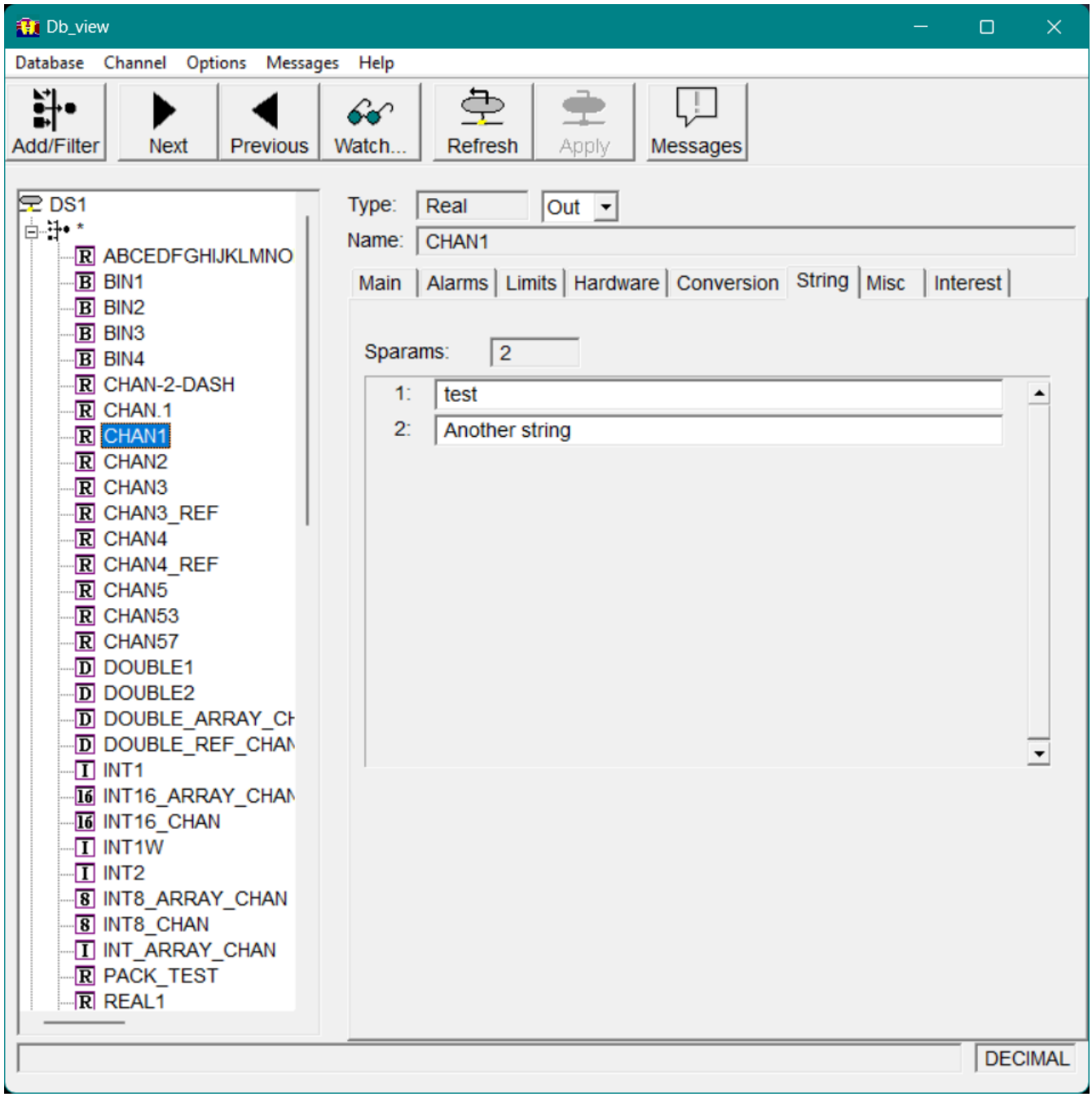
This field shows the name of the routine that performs conversions when performing a hardware put to the database, and converts from internal to external units.

Ext to Int Routine

This field shows the name of the routine that converts the value put into the database from external to internal units.

String Tab

The String tab displays the values for the sparams. This tab lists any string parameters defined for the channel.

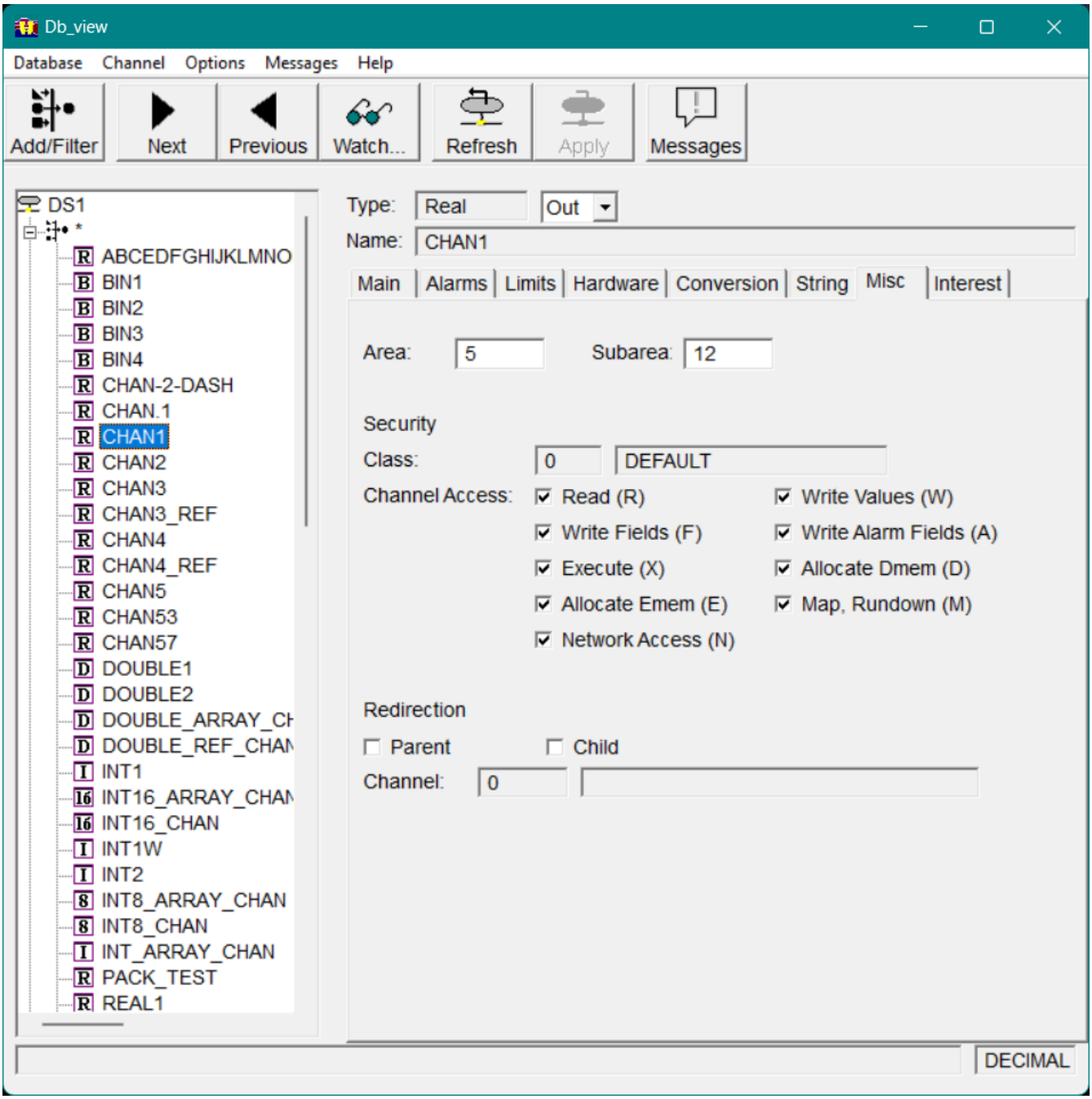


Sparams

The Sparams list box displays the sparams used to store strings for user-defined applications.

Misc Tab

The Misc tab displays the following miscellaneous options.



Db_view dialog Misc tab

Area

This field shows the area specified for the channel.

Subarea

This field shows the subarea specified for the channel.

Security

Class

This field shows the security class specified for the channel for the current user.

Channel Access

These options show the access available to the current user.

Redirection

Parent

If this option is selected, the channel being viewed is the redirection parent.

Child

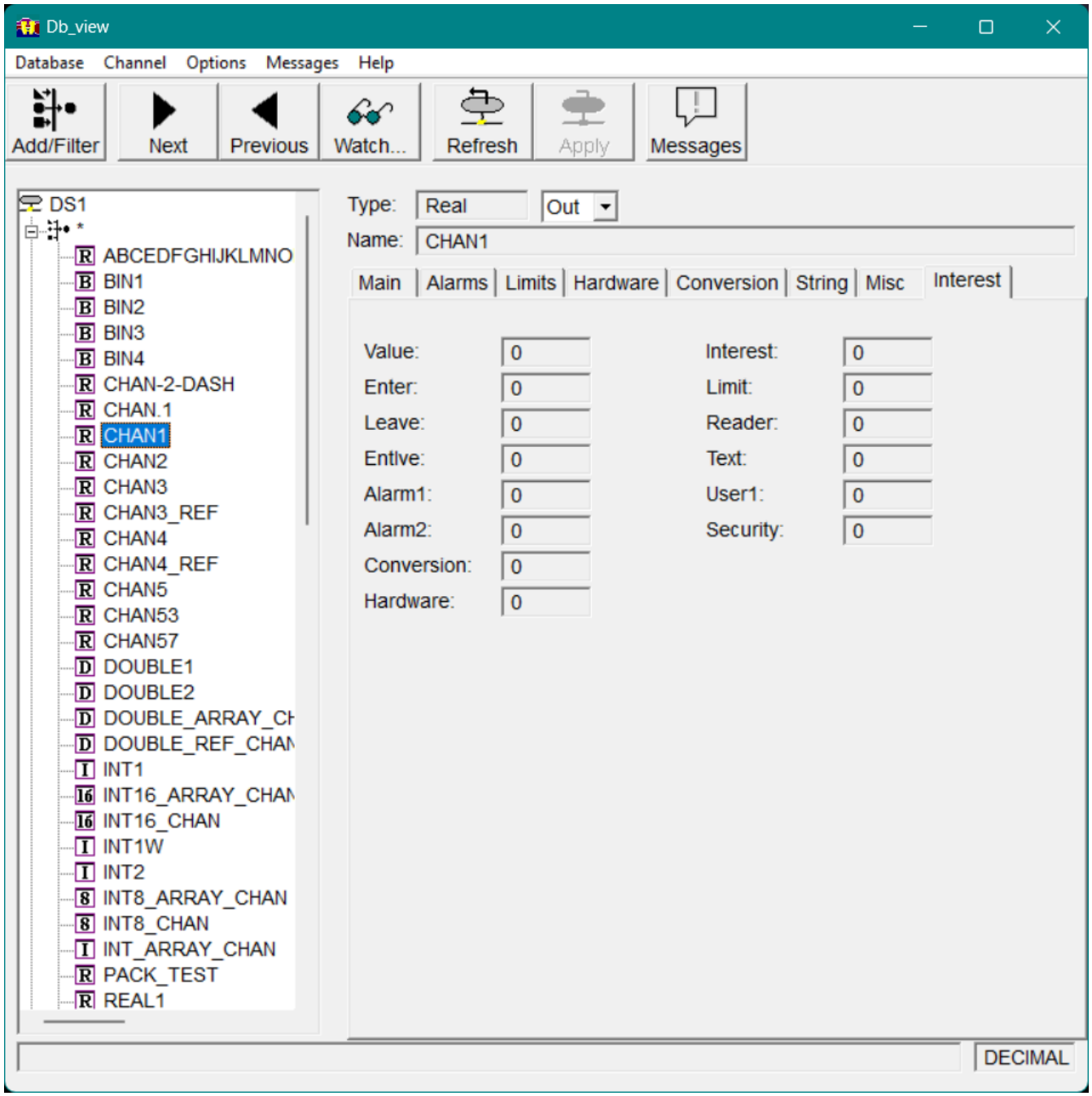
If this option is selected, the channel being viewed is the redirection child.

Channel

These field s show the channel index and name of the associated channel.

Interest Tab

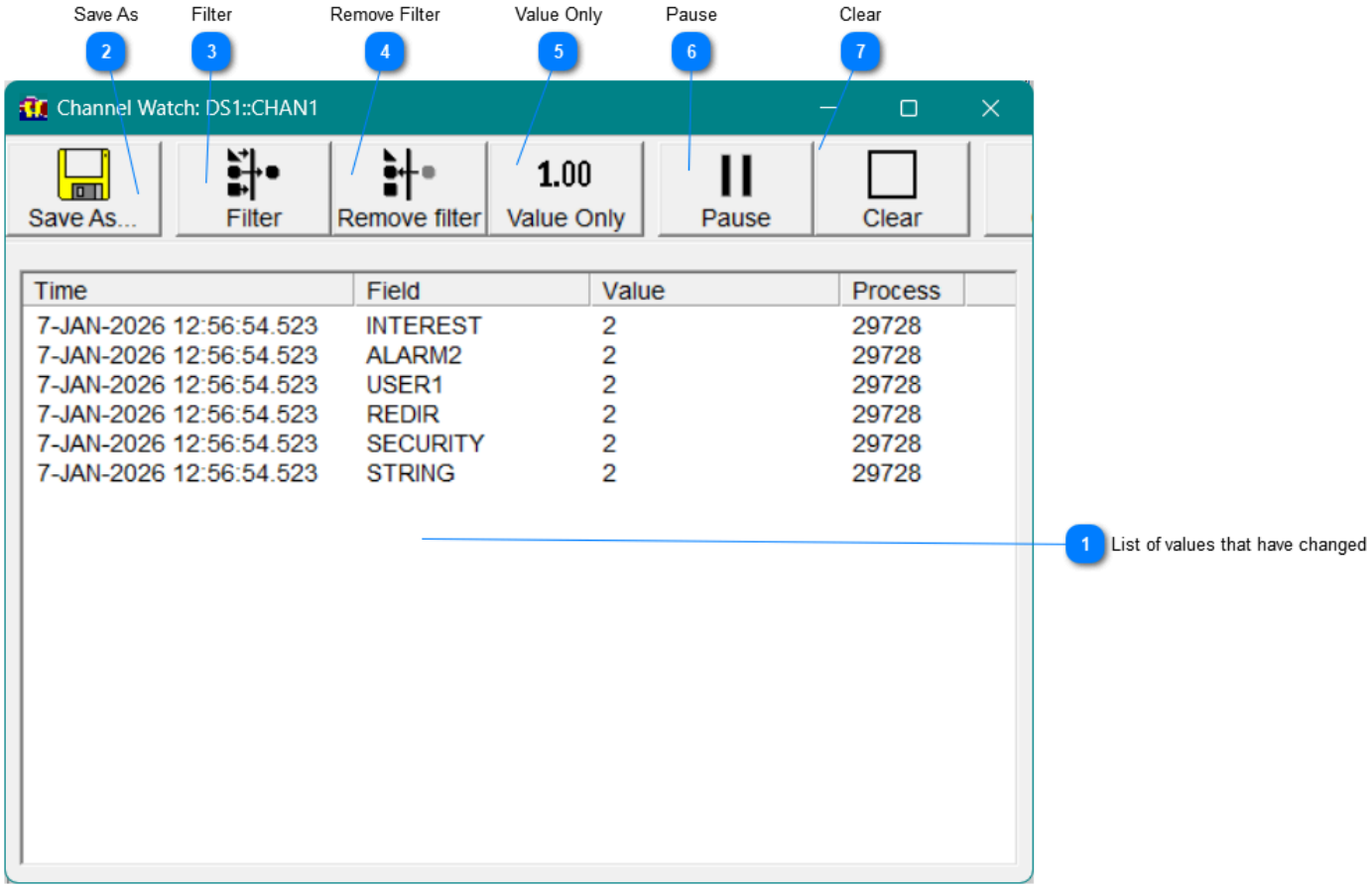
View the various interest values for the selected channel on the Interest tab.



Displaying changes to a channel using Watch

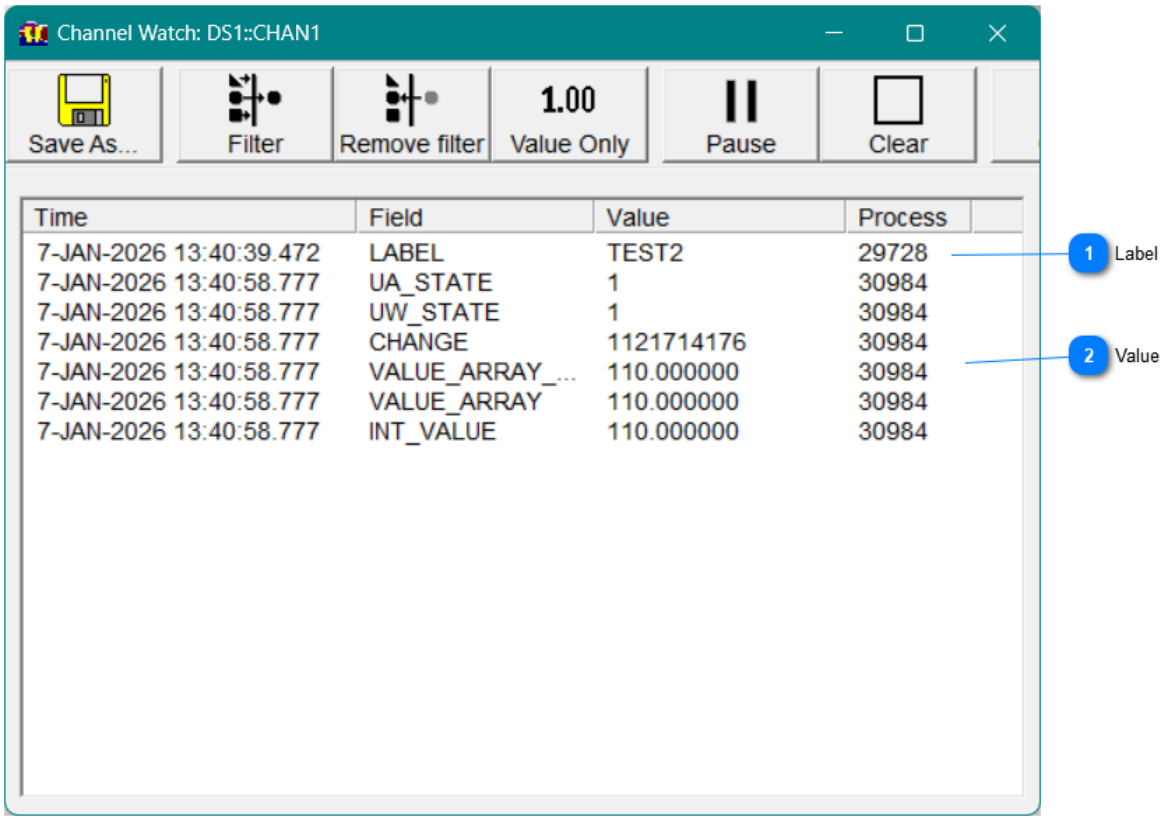
You can view changes that are made to specific channels by selecting them to be "watched". This adds an event to the channel on all of the fields. When a field is changed, the new value wil appear in the list. The Watch window appears, as shown below.

Figure26 - Channel Watch


Save As
Filter
Remove Filter
Value Only
Pause
Clear
2
3
4
5
6
7
Channel Watch: DS1::CHAN1
Save As...
Filter
Remove filter
1.00
Value Only
Pause
Clear
Time
Field
Value
Process
7-JAN-2026 12:56:54.523 INTEREST 2 29728
7-JAN-2026 12:56:54.523 ALARM2 2 29728
7-JAN-2026 12:56:54.523 USER1 2 29728
7-JAN-2026 12:56:54.523 REDIR 2 29728
7-JAN-2026 12:56:54.523 SECURITY 2 29728
7-JAN-2026 12:56:54.523 STRING 2 29728
1
List of values that have changed

1 List of values that have changed

The list includes the time of the change, the changed field, the new value, and the process ID that changed the value. When the windows first appears, there are usually some entries becuase interest has been added to the classes.

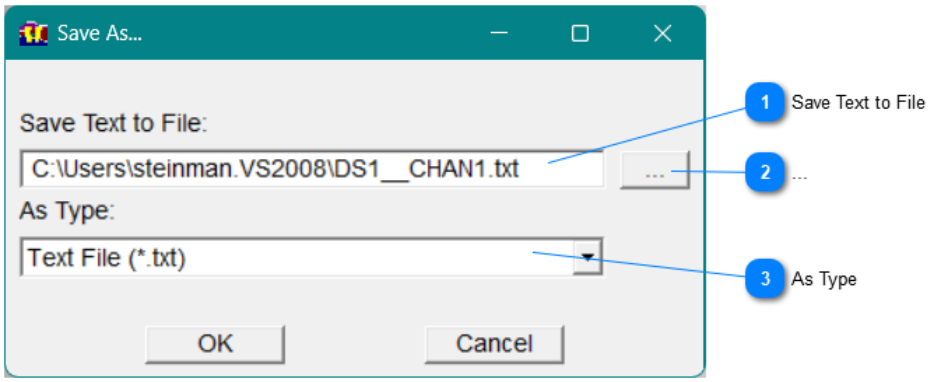


1 The label was changed to TEST2 in db_view

2 The value was changed to 110.0 in vdraw. This caused the upper warning and upper alarm to also be set.

2 Save As

Saves the list to a file.



1 Save Text to File

Enter the filename where the text will be saved.

2 ...

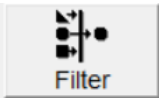
Select to enter the filename using the system dialog.

3 As Type

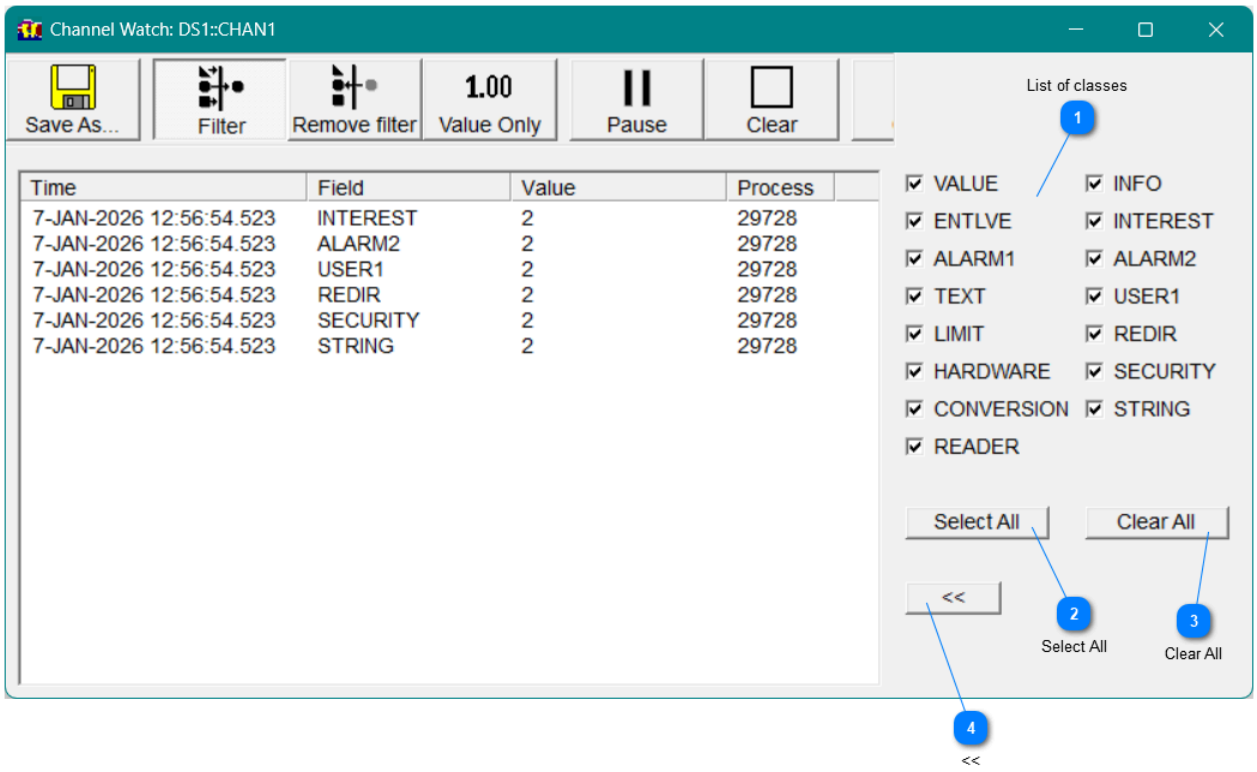
From the drop-down list, select one of the following options:

Option	Description
Text File (*.txt)	Saves the entries as a text file.
Comma Separated Text (*.cvs)	Allows the entries to a text file that uses commas to separate the fields.

3 Filter



When you select the Filter option, the window expands to display a list of classes that you can select or clear to display only the changes you want to see.



1 List of classes

You can select or clear the individual classes.

2 Select All

Select all of the classes.

3 Clear All

Clears all of the classes.

4 **<<**

Closes the Filter selection.

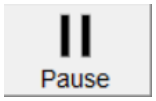
4 Remove Filter

Removes the filter so that events will occur on all classes.

5 Value Only

Only changes to the external value will be shown.

6 Pause



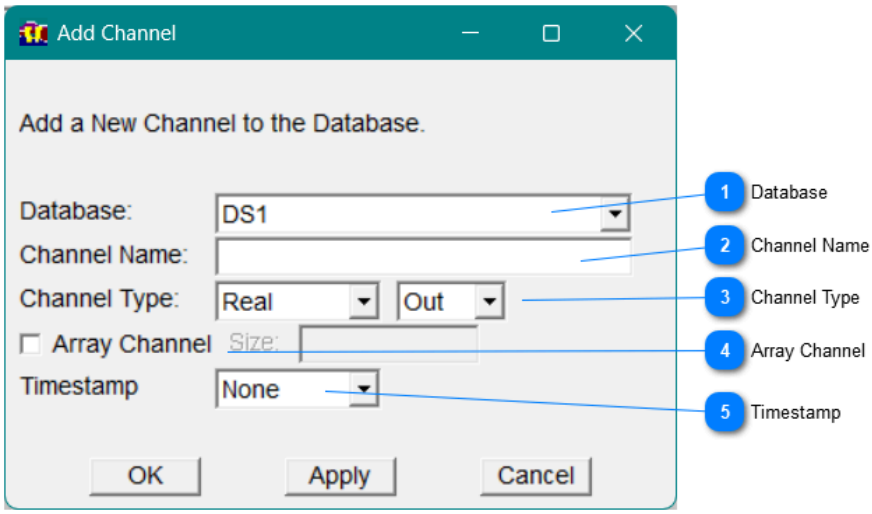
Stops the display from scrolling as new value are added.

7 Clear

Clears the list of all of the values.

Adding Channels

With Db_view, you can add a channel to the database by selecting Add on the Channel menu in the menu bar. The Add Channel dialog appears, as shown.



1 Database

Select a database

2 Channel Name

Enter a unique channel name.

3 Channel Type

Select the channel type and then select input or output.

4 Array Channel

If you need an array channel select this and then enter the size.

5 Timestamp

Option	Description
None	Select this option if the channel has no timestamp.
Single	Select this option if the channel is a single-value channel or an array channel with a single-value timestamp.
Array	Select this option if the channel is an array channel with timestamps.

Vdb2adb

The vdb2adb utility provides a way to convert the .vdb file of a Vsystem database back into an .adb file for an ASCII database, so that you can view and edit an ASCII representation of the Vsystem real-time database channels. You can then generate the output of the modified .adb file back to a .vdb file. You can find the vdb2adb.c source code in the examples applications directory.

When an output file is first opened, the database characteristics that are written to it are taken from the database containing the first channel to be recorded. This means that the initialization information for this file, along with handler and conversion modes, will be the same as those of the first database.

To run Vdb2adb

• Enter the following at the system prompt:

```
vdb2adb [channel_spec] [output_file]
```

Parameters

The first parameter is a <database::channel> specification that can include wildcards.

The second parameter is the file that the generated .adb file is written to. If no file is specified, a file new_database.adb is written in your login directory.

Qualifier

The `-one` qualifier enables you to place one keyword per line in the.adb file. Without this qualifier, Vdb2adb groups keywords and puts several on a line of text.

Examples

The following shows example applications of the Vdb2adb utility:

```
vdb2adb daisydb::demo:real_out*
Writing to file : sys$scratch:new_database.adb
DEMO:REAL_OUT1
DEMO:REAL_OUT10
DEMO:REAL_OUT2
DEMO:REAL_OUT3
DEMO:REAL_OUT4
DEMO:REAL_OUT5
DEMO:REAL_OUT6
DEMO:REAL_OUT7
DEMO:REAL_OUT8
DEMO:REAL_OUT9
-->10 channels processed
vdb2adb vscr_test_db::demo:* demo.adb
Writing to file : demo.adb
DEMO:BINARY_IN1
DEMO:BINARY_IN2
```

```
DEMO:BINARY_OUT1
DEMO:BINARY_OUT2
DEMO:INTEGER10_TEST
DEMO:INTEGER_IN1
DEMO:INTEGER_OUT1
DEMO:REAL_HIST
DEMO:REAL_IN1
DEMO:REAL_IN2
DEMO:REAL_OUT1
DEMO:REAL_OUT2
DEMO:STRING_10
DEMO:STRING_ DEMO:TIME_1
-->14 channels processed
vdb2adb
Enter channel specification ( . to quit )
<channel spec> [optional output file]
->daisydb::*theta* first.adb
Writing to file : first.adb
THETA0
THETA1
THETA_ARRAY1
THETA_ARRAY2
THETA_ARRAY3
-->5 channels processed
Enter channel specification ( . to quit )
<channel spec> [optional output file]
->daisydb::*binary* second.adb
Writing to file : second.adb
DEMO:BINARY_IN1
DEMO:BINARY_IN2
DEMO:BINARY_OUT1
DEMO:BINARY_OUT2
-->4 channels processed
Enter channel specification ( . to quit )
<channel spec> [optional output file]
->daisydb::demo:string_10
DEMO:STRING_10
-->1 channels processed
Enter channel specification ( . to quit )
<channel spec> [optional output file]
->vscr_test_db::*integer*
DEMO:INTEGER10_TEST
```

```
DEMO:INTEGER_IN1
DEMO:INTEGER_OUT1
INTEGER_ARRAY_1
INTEGER_ARRAY_2
INTEGER_ARRAY_3
-->6 channels processed
Enter channel specification ( . to quit )
<channel spec> [optional output file]
->.
exiting...
```

Vfind_channel

You can use this utility to find the name of a channel when you know its channel index. This utility is useful for developers who need information about a channel when they know only its index.

To run vfind_channel

• Enter the following at the system prompt:

```
vfind_channel <database name> <channel index>
```

Parameters

This utility takes two parameters, a `<database name>` and a `<channel index>`, and returns the name of the channel with that channel index.

You can find the source code for this utility in the examples applications directory/find_channel.c.

Vinfo

You can use the vinfo utility to get information about a database.

To run vinfo

• Enter the following at the system prompt:

```
vinfo [qualifiers] <db_name>
```

Parameter

The `<db_name>` parameter is required and specifies the name of the database about which to get information. If no other qualifiers are specified, all information about the database will be displayed.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the `vinfo` command:

attach

Use the `attach` qualifier to display any channels that have been attached by a process.

channel

`channel=<n>`

Use the `channel` qualifier to display information about a specific channel, based on a channel index n .

clear

If the lock, use the clear qualifier to clear any database locks. If the attach qualifier is specified, use the clear qualifier to clear any attaches on channels.

dmem

Use the `dmem` qualifier to display information about the dynamic database memory.

dump

`dump=<filespec>`

Use the `dump` qualifier to write the entire contents of the database to a file.

emem

Use the `emem` qualifier to display information about the database event memory.

hex

`hex=<start address>`

Use the hex qualifier to display memory at the given start address; this may be useful to examine emem or dmem. For example, you can get a start address by using `vinfo -emem` .

locks

Use the `locks` qualifier to display information about database locks.

output

`output=<filespec>`

Use the `output` qualifier to write the output to the specified file.

repair

Use the `repair` qualifier to check for tasks that have exited abnormally. If any such tasks are found, the program cleans up all of the memory used by the tasks.

security

Use the `security` qualifier to display all classes, users, and security masks for the specified database. Use the `security` qualifier with the `update` qualifier to cause the database to update its security.

See also: update

sorted_table

Use the `sorted_table` qualifier to display information about the sorted table. This include the channels and the hardware_sorted table.

tasks

Use the `tasks` qualifier to display information about the tasks connected to the database.

update

Use the `update` qualifier with the `security` qualifier to cause the database to update its security, allowing changes to users and new users to be added to the roles and access files without remapping the database. This qualifier must be run locally on the machine on which the database resides. Currently running programs are not affected; when a new program is executed, it will use the new security.

verbose

Use the `verbose` qualifier to display additional information about the selected items.

Vinfo Output Format

This section describes the different parts of the Vinfo output.

DATABASE HEADER

Displays any locks and the number of channels in the database, as shown.

```
Database: DS1

DATABASE HEADER
        NUMBER_OF_CHANNELS: 53
```

TASK TABLE

Displays a listing of each process connected to the database. In addition, the event queues are displayed, as shown in Figure 1-10See Vinfo display of processes and event queuesSee Vinfo display of processes and event queues.

```
W:\>vinfo -tasks ds1

Database: DS1

TASK TABLE          (PID= Process ID, TID= Thread ID)
    NUM SLOTS: 256
    TASK     PID    TID  Group/Link  Status
       1    29728   7356      1    0  Running  db_view  ds1,tutor
                EVENT QUEUE: size= 100, head= 0, tail= 0
                             max= 0, failed= 0
       2    30984  31068      2    0  Running  vdraw  -active ..\ds1.vcd
                EVENT QUEUE: size= 500, head= 395, tail= 395
                             max= 35, failed= 0
```

CHANNELS

Displays the channel types and names, as shown, and whether the channels are locked. In addition, each event added for that channel is displayed.

```
CHANNELS
CHANNEL 1, type= 100, name= CHAN1
    CHANNEL_QUEUE: 011316F0, class=  0, field_mask=   17, interest= 1
        EVENT_TASK_QUEUE: 01130E00, group= 2, field_mask=   17, add_interest= 1
            ECB: chix=     1, class_mask= 1, field_mask= 4, index= 0
                 task= 2, client_arg= 14D97C80, client_arg_2= 5534900
                 client_event_routine= 1002F8280, client_id= 110
                 ecb_disable= 1
            ECB: chix=     1, class_mask= 1, field_mask= 11, index= 0
                 task= 2, client_arg= 14D96A80, client_arg_2= 5534900
                 client_event_routine= 1002F8280, client_id= 100
                 ecb_disable= 1
```

EMEM

Displays the event memory (emem), as shown. The starting address and size of the block are displayed, along with the process ID of the task that allocated the memory and a code identifying what allocated the memory.

```
EMEM DISPLAY
Start Address= 010D0000
                          ADDRESS      SIZE      PID     DPID  CODE
        Free block         10D0020   379352
        Allocated block    112C9F8       48    30984            302
        Allocated block    112CA28      152    30984            304
        Allocated block    112CAC0       48    30984            302
        Allocated block    112CAF0      152    30984            304
        Allocated block    112CB88       48    30984            302
```

DMEM

Displays the dynamic memory (dmem). It is similar to the Emem display. The starting address and size of the block are displayed, along with the process ID of the task that allocated the memory and a code identifying why the memory was allocated.

SORTED TABLES

Displays the elements in the sorted tables

```
SORTED TABLES
    CHIX_2_SORTED TABLE
                    channel_count= 53
        CHANNEL INDEX:    49  ABCEDFGHIJKLMNOPQRSTUVWXYZABCEDFGHIJKLMNOPQRSTUVWXYZ
        CHANNEL INDEX:    17  BIN1
        CHANNEL INDEX:    18  BIN2
        CHANNEL INDEX:    19  BIN3
        CHANNEL INDEX:    20  BIN4
        CHANNEL INDEX:     3  CHAN-2-DASH
        CHANNEL INDEX:     9  CHAN.1
        CHANNEL INDEX:     1  CHAN1
        CHANNEL INDEX:     2  CHAN2
        CHANNEL INDEX:     4  CHAN3
        CHANNEL INDEX:     5  CHAN3_REF
        CHANNEL INDEX:     6  CHAN4
```

SECURITY

Displays all classes, users, and security masks for the specified database.

```
SECURITY
    NUMBER OF CLASSES: 2
        DEFAULT
        PARAMS
    NUMBER OF USERS: 1
        DEFAULT
            DEFAULT: RWFAXEDMN
            PARAMS: none
```

Vmessage_handler

The vmessage_handler utility monitors a string channel and displays messages whenever new values are put into the channel.

To run vmessage_handler

• Enter the following at the system prompt:

```
vmessage_handler <string channel name>
```

Parameters

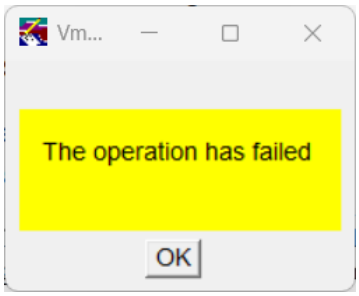
This utility requires one parameter, the `<string channel name>`, and displays a message based on the various commands (below) that can be sent to the string channel along with the message.

Commands

black, red, or yellow

Use these commands (separately) to set the background colors of messages; or, use the color command, below. For example, if the string `"The operation has failed (yellow)"` is put into the string channel, a message with the words "The operation has failed" will display on the workstation, with a yellow background and black text, as shown in Figure 27

Figure27 - Vmessage_handler example of color display



color

```
color=<n>
```

The color command specifies a color number in which to display the message, such as

• 0 for black background, white foreground (text),
• 1 for red background, white foreground (text),
• 2 for yellow background, black foreground (text),

or any other number to use the workstation default colors.

display

```
display=<node:screen>
```

The `display` command specifies the screen on which you want the message to appear by entering the standard X Windows display string, such as `node:0` . The default is the screen on which the message utility is running.

Tip Vdraw puts the X Windows display string of the screen on which it displays into the client user info string (for more information, refer to vdb_init_client_user_info in your Vsystem Vaccess Reference Guide). You can use this method to determine where a put to the database came from for displaying messages (through Vmessage_handler). Do so by calling the vdb_get_client_user_info routine in the handler of the channel and sending the user info string as the value of the display command, as well as the message to a string that Vmessage_handler is watching. For an example of a handler setting up a message string, see the file display_test.c in the examples handlers directory.

VMS When using this method on OpenVMS, be sure to specify the screen Vdraw is to display windows on by either defining the DECW$DISPLAY logical as node:0 or by starting Vdraw with -display node:0 on the command line. Otherwise, you may get an error from the Vmessage_handler about not being able to open display to a device name.

You can find the source code for the Vmessage_handler utility in the file message_handler.c located in the subdirectory x of the examples directory.

Note Once you have sent one of these commands with a message, it becomes the new default for that command. You don't need to send it again unless you want to specify a new value for the command.

exit

Use the `exit` command to exit the handler.

wait

`wait=<n.n>`

Use the `wait` command to specify the length of time the message should appear before it is closed. The value 0.0 makes a message appear until it is acknowledged. The default is to use an acknowledged message.

x

`x=<n>`

Use the `x` command to specify the x-coordinate of the message.

y

`y=<n>`

Use the `y` command to specify the y-coordinate of the message.

Vsetup

This utility builds a vsystem.vdef file for configuring Vaccess and Vczar.

Starting the Vsetup Utility

To create a vsystem.vdef file

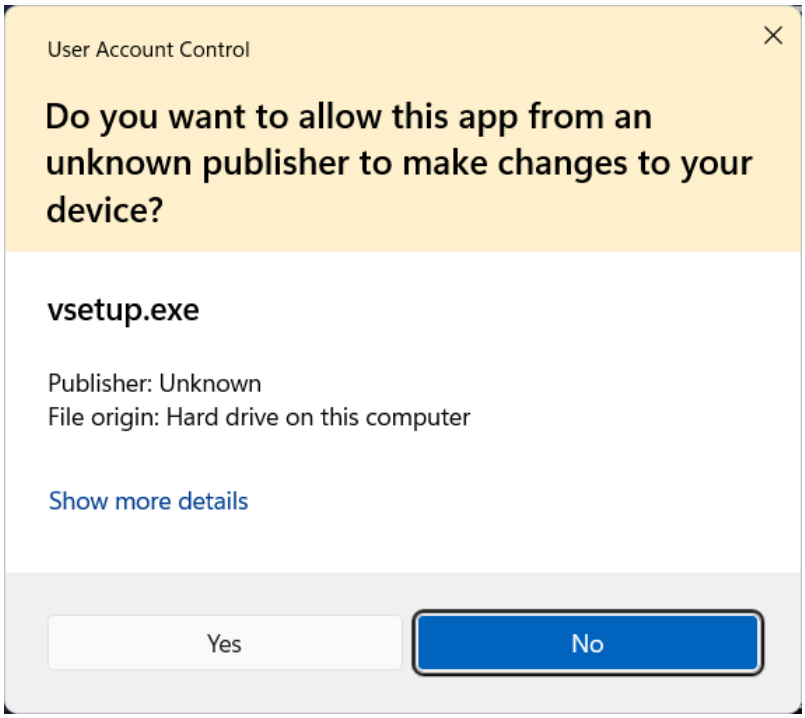
• Enter the following at the system prompt:

```
vsetup
```

When you enter the `vsetup` command, the Vsystem Configuration Setup dialog appears, as shown in Figure 29.

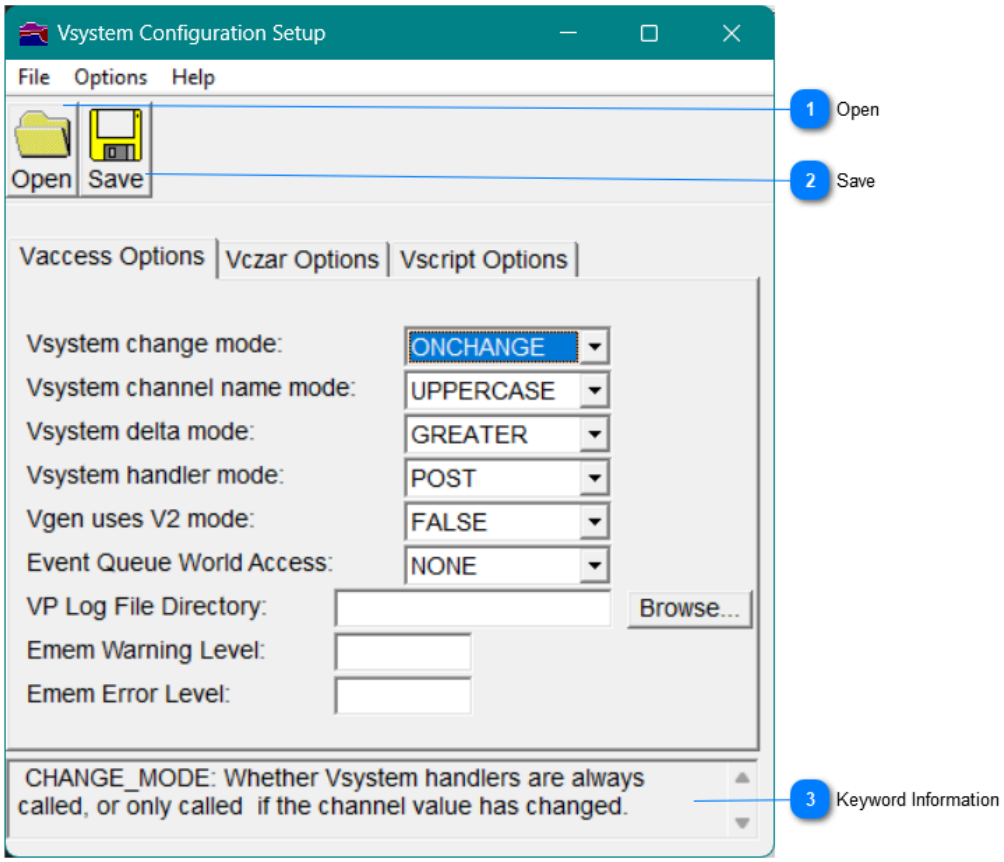
WIN On Windows you may see the User Account Control dialog. Please select **Yes**. This occurs automatically on Windows because setup is in the name.

Figure28 - Windows User Account Control



Using Vsetup

Figure29 - Vsystem Configuration Setup dialog



1 Open

To open a file for editing, on the toolbar, click the **Open** button or on the **File** menu in the menu bar, select **Open**.

2 Save

To save a file that you have edited, on the toolbar, click the Save button or on the File menu in the menu bar, select Save.

If this is the first time you have saved this file, the Save As window will appear, in which you can enter a filename. If you have previously saved this file, when you select Save, you will overwrite what you previously saved.

To save a previously saved file under a new name, in the File menu, select Save As. The Save As window appears, in which you can enter a new filename.

Search order for the vsystem.vdef file:
1. The Vsystem root directory.
2. The Vsystem host-specific directory.
3. The user's home directory (translation of HOMEDRIVE and HOMEPATH for Windows, sys$login for OpenVMS, and HOME for UNIX).
4. The current working directory.

1.

3 Keyword Information

This area displays a brief description of the last-selected keyword entry (each field in the dialog has an associated keyword).

Restoring Default Values

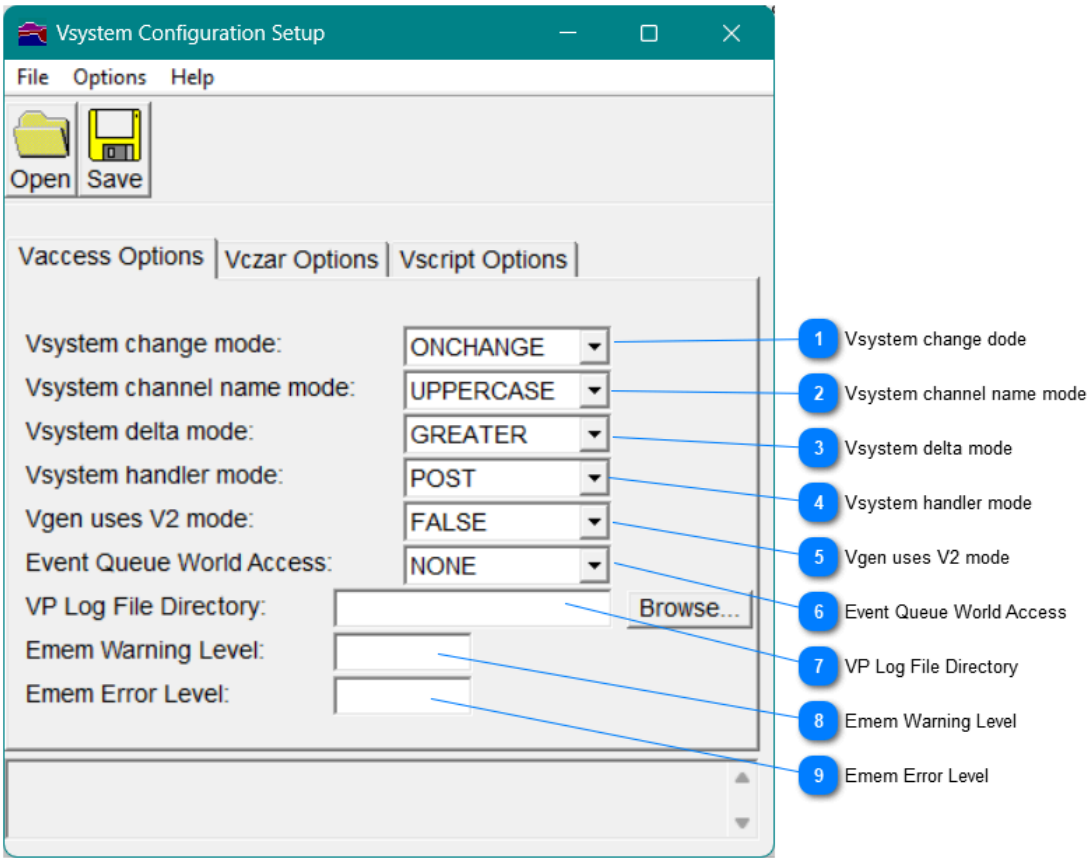
After making changes in the Vsystem Configuration Setup dialog, you may decide not to apply them. To restore the file values to their original state (removing any changes you had made in the Vsystem Configuration Setup dialog), on the **Options** menu, select **Restore Defaults**.

> ***Note*** If you have already saved the changes you made in the Vsystem Configuration Setup dialog, you will need to select Save again after restoring the default values if you want to save the default values. Otherwise, the values will revert to the last-saved values when you exit.

Vaccess Options Tab

The Vaccess Options tab is used to specify the default behaviors for Vaccess. Most of these options can be overridden in the adb file for each database.

Figure30 - Vaccess Options



1 Vsystem change dode

From the drop-down list, select one of the following options to specify the conditions under which handlers will be called:

ONCHANGE	The handlers will be called only when the channel value has changed.
ALWAYS	The handlers will always be called, whether the channel has changed or not.

2 Vsystem channel name mode

From the drop-down list, select one of the following options to specify how channel name case will be handled:

UPPERCASE	Channel names will be set to all uppercase letters; channel name searches will remain case-insensitive.
KEEP_CASE	Channel names will retain their case as entered; channel name searches will remain case-insensitive.
CASE_SENSITIVE	Channel names will retain their case as entered; channel name searches will be case-sensitive.

3 Vsystem delta mode

From the drop-down list, select one of the following options to specify the conditions under which events are generated:

GREATER	Events are generated when the value is changed by greater than the delta of the channel.
EQUAL	Events are generated when the value is changed by greater than or equal to the delta of the channel.

4 Vsystem handler mode

From the drop-down list, select one of the following options to specify the timing for handler executions:

POST	Handlers execute after a channel put.
PRIOR	Handlers execute before a channel put.

5 Vgen uses V2 mode

This option is offered for compatibility only; users should normally not specify this option. From the drop-down list, select TRUE or FALSE.

6 Event Queue World Access

From the drop-down list, select one of the following options to specify the type of access allowed in Vaccess.

NONE	Creates the event queue without world read/write access. This option is the default.
WORLD_RW	Creates the event queue with world read/write access.

7 VP Log File Directory

Use this option to specify the directory into which the log files will be placed by the Vaccess virtual processor. The default is to place the log files in the Vsystem hostname log directory. Use the Browse button to select the directory.

8 Emem Warning Level

Threshold in % when a warning message will be output. By default, this is disabled. Entering a value enables the warning

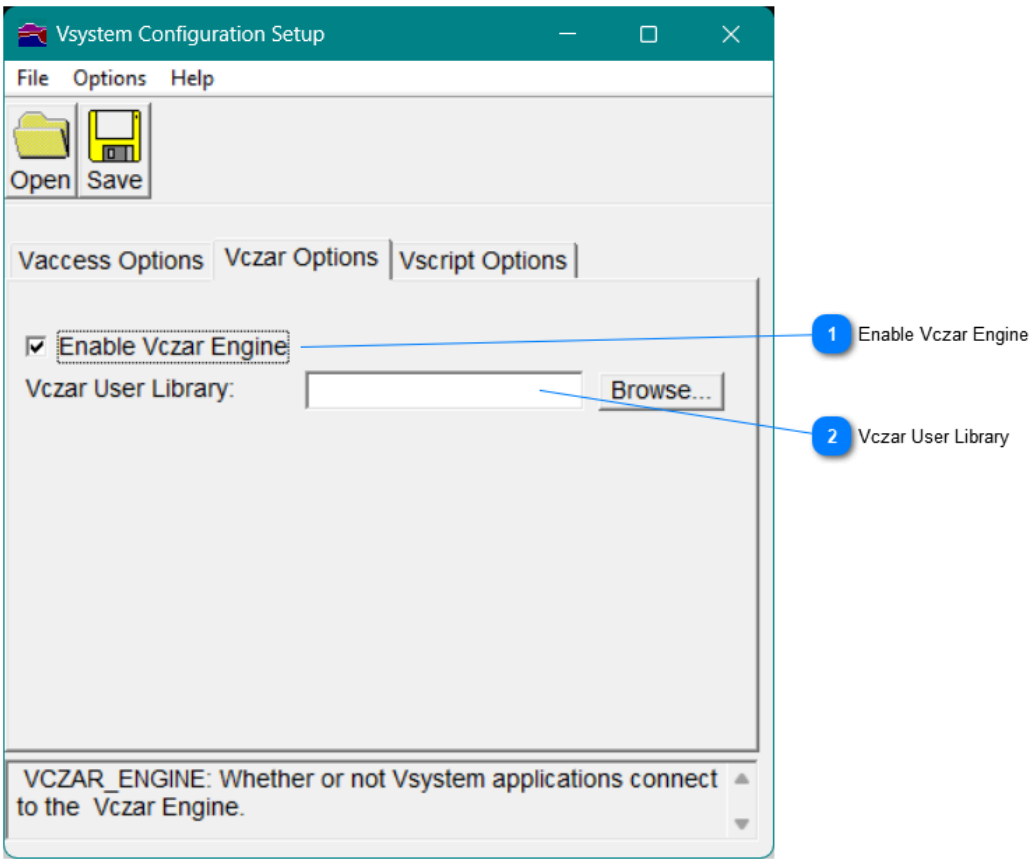
9 Emem Error Level

Threshold in % when a error message will be output and a new process will not be able to connect to the database. By default, this is disabled. Entering a value enables the error.

Vczar Options Tab

Use the options on the Vczar Options tab to specify default behaviors for Vczar.

Figure31 - Vczar Options



1 Enable Vczar Engine

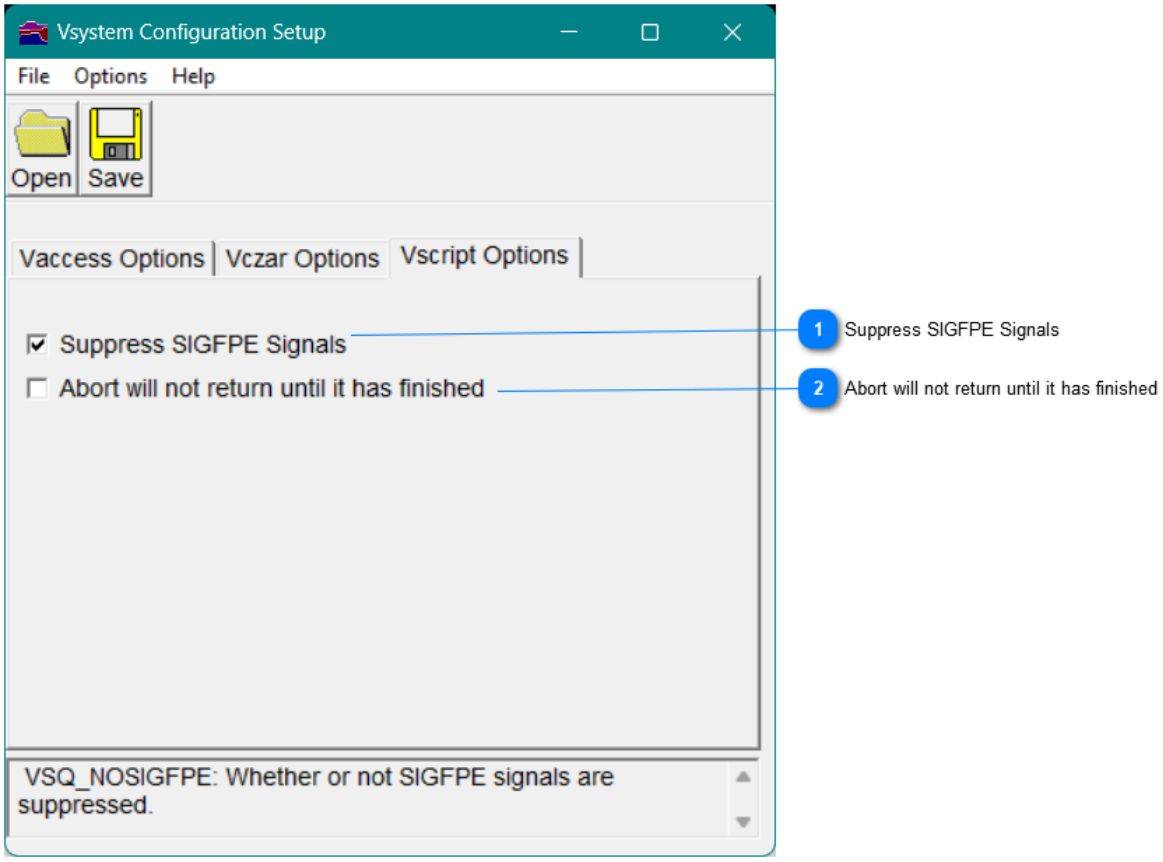
Select this option to connect Vsystem applications to the Vczar engine.

2 Vczar User Library

Enter the name of a shared library file to use as a Vczar user library. Or, use the **Browse** button to select a file from the list.

Vscript Options

Use the options on the Vscript Options tab to specify default behaviors for Vscript.



1 Suppress SIGFPE Signals

Select this option to suppress SIGFPE signal output on Linux.

2 Abort will not return until it has finished

Select this option to ensure that the abort command doesn't return until completion of the abort. This option is useful if you have problems with timing or race when using ABORTs, particularly via the Vscript API.

vsys_build_image.com (for OpenVMS)

VMS This section addresses using and creating shareable images on OpenVMS with the vsys_build_image.com command file.

Shareable Images

Using the command file `vsys_build_image.com` to make a shareable image out of a user-specified object library is a simple procedure compared to performing this task by hand; however, an understanding of shareable images and the OpenVMS linker is an important aid in avoiding problems. Most common problems involve improperly specifying the linking options for the library or generating ident mismatches with the images.

A shareable image is a special type of image that allows multiple processes to share the same piece of code. There are several advantages to this approach. Programs linked against shareable images rather than object libraries tend to be smaller. Also, if the shareable image has been installed, multiple programs linked against the shareable image (for instance `VDB_ACCSHR.EXE`) all access the same copy, thus memory usage for the system is lower. In general, computers running Vsystem have multiple images accessing Vaccess routines, so installing the Vsystem shareable images helps with performance. Also, installed images can be activated into the process context more quickly.

The OpenVMS library routine `LIB$FIND_IMAGE_SYMBOL` examines the symbol table at the top of the shareable image to find the address of the routine that has been requested. The shareable image is then mapped into the address space of the program, and Vsystem can execute the user-written routine. Removing the address resolution step until run time provides a measure of flexibility necessary for a complex system such as Vsystem, especially with the tight integration between the various components of a control system.

Additionally, you can expand the system while it is running without having to relink the images. You can add new hardware support to a system without stopping those parts that don't use the new hardware. Also, the automatic features of Vsystem can be supported because the programs shipped with Vsystem don't have to be relinked to add support for user features.

The Command Procedure

The `vsys_build_image` command procedure automates building a shareable image, but it is not a complete solution. There are complex libraries that cannot be built into shareable images with this procedure. A brief description of `vsys_build_image.com` operations should help explain how to use the command procedure.

As mentioned previously, most of the common problems encountered with vsys_build_image.com involve improper linking. It helps to have some understanding of the linking process. You can find a complete discussion of the linker in the OpenVMS linker manual. One common problem is improperly setting up the user link libraries to link the shareable image.

To create shareable images on OpenVMS

- Enter the following at the system prompt:

```
$@VSYS_COM:VSYS_BUILD_IMAGE.COM lib_file sharable_image -[exclude_file]
[options_file] [link_options]
```

Parameters

This section describes the parameters you can use with vsys_build_image.com.

exclude_file

The `exclude_file` parameter is the file specification for a file containing a list (one per line) of global names to exclude from the list of routines included in the shareable image. The primary use of this feature is to prevent global names that are not routines from being included in the transfer vector. For example, a C variable declared "`globaldef`" shows up as a global name, and this command file tries to include it in the transfer vector for the shareable image, which leads to unresolved references at link time. However, putting the name in the exclusion file prevents this from being a problem and provides a mechanism for hiding some of the routines in a library from the shareable image.

lib_file

The `lib_file` parameter is the file specification for the object library from which to build the shareable image. This is a required parameter.

link_options

The `link_options` parameter enables you to specify linking options for the final link command for the shareable image to be built. Since it is a single parameter, there should be no spaces between the options. In order to link the shareable image for debug and to generate a map, this parameter is set to the string

```
/debug/map/full
```

options_file

The `options_file` parameter is the file specification for additional linker options for linking the shareable image. vsys_build_image.com creates an option file containing the image identification string, the match criteria for the image, the cluster declaration for the transfer vectors, and any necessary `PSECT` attribute commands. If any other linker options are desired, they can be entered into this file; they are appended to the option file created by vsys_build_image.com.

sharable_image

The `sharable_image` parameter is the file specification for the shareable image to be built. This is a required parameter.

Examples

Examples of the full command line are:

```
$@vsys_com:vsys_build_image USERLIB.OLB USERSHR.EXE "" "" /DEBUG
$@vsys_com:vsys_build_image USERLIB.OLB USERSHR.EXE EXCLUDE.DAT OPTIONS.OPT /MAP
$@vsys_com:vsys_build_image USERLIB.OLB USERSHR.EXE "" OPTIONS.OPT /MAP/FULL
```

Vscan

This chapter discusses the features and functions of Vscan. It explains the parameters and qualifiers associated with it, explains how Vscan scans the hardware, and provides suggestions for customizing the utility to fit a particular *system* .

Running Vscan

Vscan provides an active and generic connection between the input hardware and the Vsystem real-time database. By calling specific hardware handlers built into the database, Vscan actively builds the list of input channels to be scanned from the specified Vsystem database(s) and activates the hardware handlers at user-specified rates. The handlers should place new values into the database, thus keeping the real-time database in a state that accurately reflects the state of the facility.

To start Vscan

- Enter the following at the system prompt:

```
vscan <parameter> [qualifier]
```

Parameter

The only required parameter is `database(s)`. This parameter contains a list of database names, separated by commas, that Vscan uses to build scan lists. If no database is specified, Vscan prompts you for a name.

Qualifiers

You can use the following qualifiers, preceded by a "/" or a "-" character, with the `vscan` command:

error

Use the `error` qualifier to specify the number of errors that can accumulate for each channel before the errors are reported to the operator. If this qualifier is absent, no error reports are made. Once the channel has experienced a number of errors equal to the error limit, the internal counter is reset and an error message is sent to the computer operator.

exit

`exit=<channel>`

Use the `exit` qualifier to specify a channel to signal Vscan to exit. Vscan exits if the value of this channel changes, thus assuring that Vscan exits cleanly.

fast_rate

Use the `fast_rate` qualifier to set the rate, in seconds, between scans for the fast scan list. If no fast scan rate is specified, the rate defaults to one second between scans. (Setting the rate to zero turns off this list.) The maximum scan rate is determined by the operating system and load; for OpenVMS, a rate of 0.01 second is achievable.

gate

in

Use the `in` qualifier (default) to tell Vscan to scan all input channels that are not marked "nosurvey" in the database. If only the in qualifier is defined, only the input channels are scanned. Use both the `in` and `out` qualifiers to scan both input and output channels.

noinit

Use the `noinit` qualifier to make vscan, when it first starts, bypass the section where the handlers of the channel are called with the initialization command `HWREADINIT` or `HWWRITEINIT`.

noslow

out

Use the `out` qualifier to tell vscan to scan all output channels that are not marked "nosurvey" in the database. If only the out qualifier is defined, only the output channels are scanned.

slow_rate

Use the `slow_rate` qualifier to set the rate, in seconds, between scans for the slow scan list. If no slow scan rate is specified, the rate defaults to ten seconds between scans. (Setting the rate to 0 turns off this list.)

Vscan Lists of Channels

Vscan maintains the following three lists of channels:

The nosurvey list comprises all input channels marked `nosurvey` in the specified databases. The Vgen keyword `nosurvey` signals Vscan to not read the channel. (For information on defining channels, refer to Chapter 1 Channel Structures in your Vsystem Vaccess Concepts Guide.) Channels may be defined as `nosurvey` for several reasons, which may include the following:

- A channel with hardware problems that needs to be temporarily disabled for repairs.
- An automatic channel (which is read when a get is made for the channel value).
- A channel for which special provisions have been made that Vscan cannot accommodate.
- The slow scan list comprises all input channels that are not marked with the nosurvey keyword and for which there has been no interest expressed by any process.
- The fast scan list comprises all input channels that are not marked with the nosurvey keyword and for which interest has been expressed by at least one other process. Note that the interest must be in the value field of the database, because this is the field Vscan updates.

Interest in a channel is set when an event has been scheduled on a channel. For example, when a Vdraw window is in active mode, any channels connected to control tools have interest set. Once the window is closed or deactivated, interest is no longer set on those channels.

According to its state, a channel can be moved dynamically between the three channel lists. The possible combinations are as follows:

nosurvey -> fast	If the nosurvey attribute is set to false and there is nonzero interest on channel.

nosurvey -> slow	If the nosurvey attribute is set to false and there is no interest on the channel.
slow –> fast	If interest is added to the channel.
fast –> nosurvey	If the nosurvey attribute is set to true.
slow –> nosurvey	If the nosurvey attribute is set to true.

Vscan handler functions

When vscan starts, it invokes all handlers with the HWREADINIT function to allow for any hardware initialization. Initialization also occurs if a channel is moved from the nosurvey list to an active scan list. An initialization flag assures that initialization is performed only once.

The fast and slow scan loop can handle input and output channels. For output channels, the API routine `vdb_setx` is called to invoke the handler for the channel. If an input channel is a numeric channel, the routine `vdb_array_hardware_get` is called.

- For single-value channels, this routine operates like the routine `vdb_hardware_get`.
- For array channels, this routine allows the entire array to be read into the channel from handler-supplied data.

Currently, nonnumeric channels cannot contain arrays; therefore, the routine `vdb_hardware_get` is called. (For descriptions of these routines, refer to Chapter 4 Library of Database Access Routines in your Vsystem Vaccess Reference Guide.)

If you are using array channels as history or ring buffers, you might consider using only the `vdb_hardware_get` routine, because this routine writes values into the channel one value at a time.

Vscan limits itself to the smallest set of functions that should be supported by any input handler. Vista Control Systems reserves function values 0 through 1023. These commands are defined in Vsystem and contained in the command include files located in the Vsystem include directory. You can use function values 1024 through 65535 for your own applications.

Customizing Vscan

You can customize vscan to meet your particular system needs; the vscan source code is provided with Vsystem in the readers subdirectory of the examples directory.

A general solution for customizing the read rate to the physical requirements of each channel is to use as many lists as needed to let each channel use the fast and slow survey rates specified in the database.

Caution VMS For OpenVMS, be aware that unless the process running Vscan is correctly configured for the number of survey rates specified in the database, it is quite likely that Vscan will exhaust its process TQELM quota prior to setting up all of the lists to be read. This situation is a fatal error, because Vscan should not continue to operate if it cannot read all of the specified channels. Prior knowledge of the database can alleviate this problem but is still not a solution.

An even simpler reader than Vscan may use only one scan rate for all channels and may use the automatic input capability of Vsystem. In this case, Vscan does not need to put the data in the database, it can just issue a vdb_xget and discard the resulting data.

An additional approach is to use only one rate per channel. This simplifies some of the other design choices, as well as simplifying the list management, but it doesn't provide any flexibility as the database changes.

For more information on database readers, refer to Chapter 7 Readers and Writers in your Vsystem Vaccess Concepts Guide.

Vsystem Modbus Scanner

The Vmodbus Scanner provides a bidirectional connection between Modbus hardware and Vsystem databases and archives. Input channels are periodically updated with values read from hardware, and values of output channels are written to hardware when they change. A single Vmodbus Scanner process can scan multiple PLCs, possibly connected to different communication ports (serial, TCP, and so on).

Running the Vmodbus Scanner usually involves the following three steps:

See Associating Vsystem Channels with Modbus Data Points See Associating Vsystem Channels with Modbus Data Points

1. *See Defining the Hardware Configuration in a Configuration File* See Defining the Hardware Configuration in a Configuration File
2. *See Starting the Vmodbus Scanner Process* See Starting the Vmodbus Scanner Process

Also included in this chapter are the following sections:

- *See Conversion Routines* See Conversion Routines
- *See Vmodbus Test Utility* See Vmodbus Test Utility
- See Vsystem Modbus Scanner Addressing *See Vsystem Modbus Scanner Addressing*

Note This chapter describes the use of Modbus protocol only in relation to Vista's Modbus Scanner. For information about Modbus protocol and setting up the Modbus hardware, you must contact the manufacturer.

Associating Vsystem Channels with Modbus Data Points

A channel is associated with a hardware data point using the database fields shown in Table 3-1. The Field column shows the default field that serves the given function. You can override the default mapping between functions and fields with the See fields qualifierSee fields.

Table 1 – Functions and associated database fields

Function	Field	Usage
PLCName	MODULE	Defines the name of the PLC.
address*	HPARAM1	Defines the address of the data point in the PLC. See also See offset See offset.
scanRate	HPARAM2	Defines the scan rate or scan schedule of the channel. If the See schedule qualifier See schedule is present on the command line, this field represents the index of the schedule in the list of schedules; otherwise, it is the scan rate. By default, the scan rate is in seconds, but this can be changed with the See rate qualifier See rate . If the value of this field is 0, 1 is assumed.
dataPointType	HWDATATYPE	Defines the type of the data point. Valid dataPointType values are defined in See Vsystem Modbus Scanner Addressing See Vsystem Modbus Scanner Addressing. Note If this field is 0, the TYPE and IN / OUT fields are used to determine the data point type. In the above list, the TYPE and IN / OUT fields are shown in parentheses. Also note that if you use the TYPE and IN / OUT fields, you cannot define an input

		database channel connected to a holding register. If that is what you need, you must use HWDATATYPE.
disposition		Defines the disposition of the input data point. Every time the data point is scanned, the new value is either (1) archived, (2) put into the database, or (3) both. If no disposition field is specified, or if the value of the disposition field is 0, the disposition is determined by the presence of the See varchive qualifier See varchive . If this qualifier is present, the data point is archived, otherwise it is put into the database. (See also See circular, nocircularSee circular, nocircular.)
conversionRoutine		Defines a conversion routine that is called when a value is about to be put into the input channel, or when a value of an output channel is about to be sent to the hardware.
refreshRate		Normally, channel values are archived only when they change. Sometimes, however, it is useful to periodically archive a value even if it doesn't change (to "refresh" the value in the archive). This field defines the refresh rate, in seconds. Note that a value can be refreshed only when the Vmodbus Scanner reads a new value from the hardware. This means that the actual refresh rate can be longer than specified by this field, up to "refreshRate + scanRate + 1" ("+1" is because the resolution of the refresh timer is in seconds). If no refreshRate field is specified, or if the value of the field is 0, the refresh option is disabled for that channel. A -1 value for the field causes the Vmodbus Scanner to always archive the channel.
clockSync		Set this channel field to a time of day, in minutes, to cause the scanner to delay its first scan until this time. Note that all channels in a scan block should have the same clockSync value; only the first channel in each block will be checked, so subsequent clockSync values will be ignored.

* For more information about accessing your data points, see *See Vsystem Modbus Scanner Addressing* See Vsystem Modbus Scanner Addressing.

Defining the Hardware Configuration in a Configuration File

This is an optional step. No configuration file is required when one or more identical PLCs are connected to a single communication port. If multiple communication ports are involved, or if PLCs have different characteristics, the PLCs and ports must be defined in a configuration file, and the configuration file must be specified on the command line with the See configFile qualifierSee configFile.

A configuration file may consist of multiple lines of the following types:

- Comment
- PLC Description
- Port Description

Comment

Comments are lines with the "#" character in the first position.

PLC description

PLC descriptions are lines that start with a PLC keyword. The format of a PLC description is:

```
PLC name [qualifiers]
```

You can use the following qualifiers, preceded by a "/" or "-" character, in a PLC description. Several of these options can also be defined in the See Port description See Port description or on the command line. If such an option is not specified in the PLC description, it is defaulted to the value specified in the port description or on the command line. If such an option is specified in several places, the PLC's specification overrides the others.

See a32 See a32

See class See class

See control_design *See control_design*

See d16 See d16

See endian See endian

See enron See enron

See ic_techSee ic_tech

See map See map

See offset See offset

See port See port

See throughput_channel See throughput_channel

See unit See unit

See word_swapSee word_swap

Port description

Port descriptions are lines that start with a port keyword. The format of a port description is:

```
PORT name [qualifiers]
```

You can use the following qualifiers, preceded by a "/" or "-" character, in a port description. Several of these qualifiers can also be defined in the See PLC description See PLC description or on the command line. If such an option is not specified in the port description, it is defaulted to the value specified on the command line. If such an option is specified in several places, the port's specification overrides the value specified on the command line, but is itself overridden by the value specified in the PLC description.

See a32 See a32

See baudRate See baudRate

See class See class	See control_designSee control_design
See d16 See d16	*See device* See device
See ic_techSee ic_tech	*See offset* See offset
See parity See parity	*See stop* See stop
See timeout See timeout	*See type* See type

Example

```
# This is an example of a configuration file
port p1 -type TCP -device host1
port p2 /type=RTU /device=com2: \
/baudRate=9600 /stop=1 /parity=ODD

plc area1 -port=p1 -unit=2
plc area2 -port=p2 -unit=3
```

Starting the Vmodbus Scanner Process

To start the Vmodbus Scanner from the command line

```
vmodbus [qualifiers]
```

You may need to use a number of qualifiers when you start the Vmodbus Scanner. For example, in the simplest case, you need to specify a Vsystem database, the address and type of modbus hardware, and a list of PLCs. Below is an example showing your database as vmodbus.adb and your modbus hardware as accessible via TCP on the host modbushost. In this case, you would start the scanner using the following command:

```
vmodbus -database vmodbus -type tcp -device modbushost -plc plc1
```

Qualifiers

When starting the Vmodbus Scanner, you can use the following qualifiers, preceded by a "/" or a "-" character.

Note If the "-" form is used, the "=" character is not required. Multiple values for qualifiers specified without the "=" character should be written without parentheses and commas.

See a32See a32

See baudRateSee baudRate

See bufferSizeSee bufferSize

See circular, nocircularSee circular, nocircular

See classSee class

See configFileSee configFile

See controlChannelSee controlChannel

See control_designSee control_design

See d16See d16

See databaseSee database

See deviceSee device

See disconnectSee disconnect

See endianSee endian

See endScanChannelSee endScanChannel

See enronSee enron

See fieldsSee fields

See fsyncSee fsync

See groupTimestampsSee groupTimestamps

See ic_techSee ic_tech

See intervalLengthSee intervalLength

See latencySee latency

See lengthSee length

See mapSee map

See nobannerSee nobanner

See offsetSee offset

See onceSee once

See paritySee parity

See PLCSee PLC

See portSee port

See rateSee rate

See scheduleSee schedule

See sharedSee shared

See showSee show

See staledata_channelSee staledata_channel

See startScanChannelSee startScanChannel

See startupOutputSee startupOutput

See statusChannelSee statusChannel

See stopSee stop

See throughput_channelSee throughput_channel

See timeoutSee timeout

See timeSpanSee timeSpan

See typeSee type

See unitSee unit

See userFilesSee userFiles

See userInitRoutineSee userInitRoutine

See userParameterSee userParameter

See varchiveSee varchive

See wdChannelSee wdChannel

See wdRateSee wdRate

See word_swapSee word_swap

a32

32-bit and 64-bit quantities are normally addressed as 16-bit registers. For example, if you have three 32-bit floating-point values starting at address 1, you need to define them in the .adb file as 1, 3 and 5. Three 64-bit floating values would have to be defined as 1, 5 and 9. Use the a32 qualifier to tell the scanner that 32-bit and

higher quantities are addressed as a whole. With this qualifier specified, the three floating values in the example above can be addressed as 1, 2 and 3 in the .adb file.

Note that the address of the first 32-bit or 64-bit value is always a 16-bit address, whether `a32` is specified or not. Suppose, for example, that you have three 32-bit values starting at address 11. Without an `a32` qualifier, you would address them as 11, 13 and 15. With `a32`, you would address them as 11, 12 and 13.

baudRate

`baudRate=n`

Use this qualifier to define the baud rate for the serial device specified with the See device qualifierSee device. The baudRate qualifier is not implemented on OpenVMS.

bufferSize

`bufferSize=n`

Use this qualifier to specify the buffer size, in bytes, to use when writing archived data to the varchive file. (See also See varchive See varchive.)

circular

Use the `circular` qualifier to archive data in sets of intervals. The oldest intervals are automatically deleted as the archive file exceeds the specified total archive file length.

If you specify the circular qualifier, you must specify the interval length in either kilobytes or in seconds (see the See intervalLength qualifierSee intervalLength).

class

class=n

Use this qualifier to define the conformance class of the PLCs specified on the command line and the default conformance class of the ports and PLCs defined in the configuration file. Valid values are 0 and 1; the default value is 1.

configFile

configFile=filename

Use this qualifier to specify a configuration file for the Vmodbus Scanner.

controlChannel

controlChannel=string-channel-name

Use this qualifier to specify a control channel for the Vmodbus Scanner. If you enter exit into the control channel, the Vmodbus Scanner process exits itself. Putting scan into the control channel forces the scanner to scan all channels without waiting for the next scheduled time.

control_design

Use this qualifier when connecting to a Control Design PLC. This qualifier changes some Modbus commands for compatibility with Control Design PLCs:

• Uses Modbus function 3 in place of 4 and 2.

• Uses Modbus commands 3 and 16 to write digitals, rather than command 5.

• Uses Modbus command 16 instead of 6 to write integers.

• Allows Control Design string and SDI (logbuffer) data types.

Use this qualifier only in a config file; do not use on the command line. (See See Troubleshooting Addressing See Troubleshooting Addressing, for a table showing the command conversions.)

d16

Normally, the Vmodbus Scanner uses "read coils" (FC 1) and "read input discretes" (FC 2) functions to read discretes from PLCs of class 1 (see See class See class). But some PLCs don't appear to support these functions. Use this qualifier to tell the scanner to use FC3 and FC4 functions instead.

database

database=(database,...)

Use this qualifier to specify one or more Vsystem databases that need to be scanned.

device

device=name

Use this qualifier to specify the default communication device. The default device is used to access any PLC for which there is no explicitly specified device in the configuration file, or if no configuration file is used. The device name is operating-system- and communication-type- specific. For a TCP communication type, the device name is the hostname. For example, for a serial communication type, the device name could be "com1:" on Windows or "/dev/ttya0" on a UNIX system.

disconnect

Normally, the communication ports are opened at the scanner startup and remain open until the scanner exits. Use this qualifier to tell the scanner to close the communication port after each scan.

endian

Use this qualifier to determine the "endianness" of the PLC with regard to 32- and 64-bit values (that is, which of the 16-bit registers come first). Available values are "BIG"--MSF order (most significant first)--and "LITTLE"--LSF order (least significant first). The default is "LITTLE".

endScanChannel

endScanChannel=string

Use this qualifier to specify a channel to be archived at the end of each scan.

enron

Use this qualifier to specify options for Enron-style PLC addressing. This flag affects only 32-bit values. When this flag is included on the PLC and/or PORT line, the scanner treats 32-bit values in that PLC and/or port as one register as opposed to two .

The differences between the standard and enron modes are:

• The scanner doesn't subtract 1 from the address, as it normally does.

• The address is treated as the address of the whole value, not the first of the two 16-bit registers.

• The number of points in the request is the number of 32-bit points, not the number of 16-bit registers.

fields

fields=(function:field,...)

This qualifier enables you to redefine the fields used to associate Vsystem channels with Modbus hardware data points.

[UNIX] Examples:

-fields plcName:iotype addr:hparam10 conv:exttoint

/fields=(plcName:iotype,addr:hparam10,conv:exttoint)

fsync

[UNIX/WNT] Use the fsync qualifier (for UNIX and Windows) to copy all new data from the system cache to permanent storage (see the See latency qualifierSee latency). When you use this qualifier, you ensure that data is readable in case of a power outage or system crash.

> Note The fsync qualifier works for shared archive files only.

groupTimestamps

When set to 1, this qualifier causes all channels in a scan to use the initial scan time for their timestamps. This option is not available on OpenVMS.

ic_tech

Use this qualifier when connecting to an IC Tech PLC. This qualifier changes some Modbus commands for compatibility with IC Tech PLCs:

• Uses Modbus function 3 in place of 4 and 2.

• Uses Modbus commands 3 and 16 to write digitals, rather than

command 5.

• Uses Modbus command 16 instead of 6 to write integers.

• Allows IC Tech string and SDI (logbuffer) data types.

Use this qualifier only in a config file; do not use on the command line. (See See Troubleshooting Addressing See Troubleshooting Addressing, for a table showing the command conversions.)

intervalLength

intervalLength=(seconds:n,kbytes:n)

Use this qualifier to specify the interval length, in kilobytes or seconds, for circular mode (specified with the circular qualifierSee circular, nocircular). You must specify at least one of these qualifiers for circular loggers.

A circular archive file consists of intervals. When an interval exceeds the specified interval length, a new interval is created. (See also the See length qualifierSee length.)

latency

latency=<n>

Use the latency qualifier to specify the interval, in seconds, between the time data is archived and the time it becomes readable. This function is available only for archive files created in shared mode (specified with the See shared qualifierSee shared).

length

length=(seconds:n,kbytes:n)

Use this qualifier to specify the total length, in kilobytes or seconds, for a circular archive. If the length of a circular archive exceeds this limit, the oldest interval is deleted.

Examples:

-length sec:3600

-length sec:3600 kb:1000

/length=(sec:3600,kb:1000)

map

Use this qualifier to define the location of data in the PLC. The optional See offset :n qualifierSee offset, defines the initial offset of all data in the PLC. You can also use multiple "type:offset$quantity" options to define register ranges for different data types, where offset is the starting register of the block, quantity is the number of data points, and type is one of the options shown in Table 3-2See Data types for defining register ranges.

Data types for defining register ranges

Type	Definition
binary	input discretes
int16	input 16-bit integer registers

int32	input 32-bit integer values
float	input 32-bit floating-point values
double	input 64-bit double values
outputBinary	output discretes
outputInt16	output 16-bit integer register
outputInt32	output 32-bit integer values
outputFloat	output 32-bit floating-point values
outputDouble	output 64-bit double values

If map is specified, it must include all data points referenced in the database(s). A single map specification may include multiple blocks for the same data type. The blocks should not overlap, and the specifications must be monotonically increasing.

By specifying map , you change the way data points are referred to in a database. Addresses start from 1, with the address "1" corresponding to the first data point in the first "type" specification for the given data type. When you specify map , you automatically enable the a32 mode (see See a32 See a32). The offset : n qualifier affects both digitals and registers. Since digitals and registers are addressed independently in PLCs of class=1, we don't recommend that you use "offset:0" with PLCs of class=1.

Examples:

-map offset:1000 binary:0$100 int16:1000$32 int16:2000$16

/map=(offset:1000,binary:0$100,int16:1000$32,int16:2000$16)

nobanner

Use this qualifier to suppress display of the startup banner.

nocircular

Use the nocircular qualifier to place archived data into an archive file without ever deleting data from this file, causing the archive file to continue to increase in size as the scanner runs. This is the default.

You can specify the total archive file length either in kilobytes or in seconds (see the See length qualifierSee length).

offset

offset=(dataType:value,...)

This qualifier enables you to define offsets of the first data point of certain data types. The default offset is 0.

The actual address of a data point in a PLC is determined by adding offset to the value specified in the address field in the database. Offsets for discretes (binary data points) can be either in bits or bytes, depending on the type of the PLC. For PLCs of class 0 and PLCs of type d16, the offsets are in int16 registers. For other PLCs, discrete offsets are in bits. (For more information on the type of a PLC, see the See class See class, and See d16 qualifiersSee d16). Offsets for all other data types are in int16 registers.

You can define an offset for the data types shown in Table 3-3.

Data types for defining offsets

Type	Definition
binary	input discretes
int16	input 16-bit integer registers
int32	input 32-bit integer values
float	input 32-bit floating-point values
double	input 64-bit double values
outputBinary	output discretes
outputInt16	output 16-bit integer register

outputInt32	output 32-bit integer values
outputFloat	output 32-bit floating-point values
outputDouble	output 64-bit double values

Examples:

-offset binary:1000 double:100 outputBinary:200

/offset=(binary:1000,double:100,outputBinary:200)

once

once=(field,...)

Use this qualifier to tell the scanner to archive the specified string fields in "once" mode.

parity

parity=string

Use this qualifier to define the data parity for the serial device specified with the See device qualifierSee device. Supported values are NONE , EVEN , and ODD . This qualifier is not implemented on OpenVMS.

PLC

PLC=(name[:unit],...)

Use this qualifier to specify one or more PLCs to be scanned. The name of a PLC may be followed by a colon, which in turn is followed by the unit (slave) address. If the unit address is not specified, the unit address from the configuration file is used. If the PLC is not defined in the configuration file, or if no configuration file is used, the unit address is defaulted to 1.

If both the See configFile See configFile and PLC qualifiers are specified on the command line, values defined by the PLC qualifier take precedence.

port

Use this qualifier to define the name of the communication port the given PLC is connected to. This port should be defined in the same configurations file as a port description. If this qualifier is not specified, the necessary communication port characteristics are defaulted to those specified on the command line.

rate

rate=n

Use this qualifier to specify the scan rate unit; the default unit is 1000 milliseconds. The scan rate of a channel, in milliseconds, is determined by multiplying this value by the value of the See scanRate fieldSee scanRate. (See also the schedule qualifier, below.)

schedule

schedule=(schedule1,...)

Use this qualifier to specify when to scan channels. If schedule is not specified, channels are scanned according to the value defined with the rate qualifier. If schedule is specified, any given channel is scanned according to the schedule defined by the See scanRate fieldSee scanRate. Schedules are numbered starting at 1.

shared

Use the shared qualifier to create an archive file in a mode that allows other processes to read it while it is still open (receiving data). Data in a shared archive file becomes readable after an interval specified with the See latency qualifierSee latency.

show

show=(option,...)

Use this qualifier to request that certain information be printed on the standard output. Supported options are shown in Table 3-4.

Standard output printing options

Option	Prints
decimal	Addresses in decimal form. By default addresses are shown in hex.
PLCs	The list of all scanned data points, sorted by data-point type and address.
ports	Information about ports, which includes the number and names of PLCs connected to this port and of data points scanned using this port.
scans	Information about contiguous scan blocks and scan rates.

Examples:

-show plcs ports scans

/show=(plcs,ports,scans)

staledata_channel

When you set this qualifier to an integer channel with an initial value of 0, the scanner writes the number of scans since the last successful scan to the channel. This is useful for monitoring communication between the RTU(s) and the scanner.

startScanChannel

startScanChannel=string

Use this qualifier to specify a channel to be archived at the beginning of each scan.

startupOutput

Set this qualifier to 0 to suppress the initial puts to the PLC upon startup. This option is not available on OpenVMS.

statusChannel

statusChannel=string-channel-name

Use this qualifier to specify a channel that is used to store error messages generated by the Vmodbus Scanner, enabling you to archive messages from more than one scanner process using Vsystem Vlogger.

stop

stop=n

Use this qualifier to define the number of stop bits for the serial device specified with the See device qualifierSee device. This qualifier is not implemented on OpenVMS.

throughput_channel

When this qualifier is set to a floating-point channel, the scanner writes the ratio of success/failture for this PLC to the channel.

timeout

timeout=n

Give this qualifier a time, in seconds, to specify the amount of time the scanner waits for communication from the PLC before returning an IO error.

timeSpan

timeSpan=n

Use this qualifier to instruct the Vmodbus Scanner process to exit automatically after the specified number of seconds.

type

type=communication-type

Use this qualifier to define the communication type of the default device (the device specified with the See device qualifierSee device). Supported values include ASCII , RTU , TCP, TCP/RTU (for RTU-style communication over TCP/IP) and TCP/ASCII (for ASCII-style communication over TCP/IP). The latter two allow for serial-over-TCP functionality.

unit

Use this qualifier to define the unit (slave) address of the PLC. The default value is 1.

userFiles

userFiles=(filename,...)

Use this qualifier to specify one or more shared libraries containing user-supplied conversion routines.

userInitRoutine

userInitRoutine=string

Use this qualifier to specify the user-supplied conversion initialization routine.

userParameter

userParameter=string

Use this qualifier to specify a parameter accessible by any user-supplied conversion routine.

varchive

varchive=filename

Use this qualifier to specify an output archive file.

wdChannel

wdChannel=integer-channel-name

Use this qualifier to specify a watchdog channel that is changed periodically by the scanner to indicate that the scanner is alive. This channel must be an integer channel. The value fluctuates in the range from 1 to 100.

A watchdog channel may be defined as a timestamped channel. This option may be used to store and access the timestamp of the last value change.

wdRate

wdRate=n

Use this qualifier to specify the frequency, in seconds, at which the watchdog channel changes. The default value is 10 seconds.

word_swap

word_swap=1

Use this qualifier to swap the order of the words within 16-bit registers when reading or writing to 32- or 64-bit floats/integers.

Conversion Routines

You can add user-written conversion routines to the Vmodbus Scanner to allow more complex data transformations to be used when converting from internal hardware format to database format, or vice versa. See the conversion_routines.c example for more information.

Vmodbus Test Utility

Vmodbus Test is a test utility that you can use to diagnose problems with the Vmodbus Scanner. With this test program, you can manually execute certain modbus functions. For example, you can read two input registers from an address 100 using the following command:

vmodbus_test exec -type tcp -device modbushost f4 100 2

To display the list of modbus functions supported by the test program

• Enter at the system prompt:

vmodbus_test exec -type tcp -device modbushost -help

vmodbus_test show functions

To display arguments of function #3

• Enter at the system prompt:

vmodbus_test exec -type tcp -device modbushost f3 -help

Vsystem Modbus Scanner Addressing

Modbus addressing can be tricky, because there are several ways that you can refer to addresses in the register space. Generally, the Modbus address tables are referred to in one of the ways shown in Table 3-5.

Modbus address tables

Modern Terminology	Classic Terminology	Modicon Addressing
Discrete Outputs	Coils	0:00000, start reg. 00001
Discrete Inputs	Inputs	1:00000, start reg. 10001
Input Registers	Input Registers	3:00000, start reg. 30001
Output Registers	Holding Registers	4:00000, start reg. 40001

The Vmodbus Scanner starts with register 1 and determines the register table above based on the Vsystem channel type and/or hwdatatype.

Table 3-6 shows the default register tables accessed by in/out and channel types.

Channel types and default register tables

Channel Type	Default Register Table
binary in	1:00000 (Discrete Inputs)
binary out	0:00000 (Discrete Outputs / Coils)
int16	in 3:00000 (Input Registers) IN (as int)
int16 out	4:00000 (Output / Holding Registers) OUT (as int using Modbus command 6)
uint16 in	3:00000 (Input Registers) IN (as unsigned int)
uint16 out	4:00000 (Output / Holding Registers) OUT (as unsigned int)
real in	3:00000 (Input Registers) IN (as 32-bit floating-point)
real out	4:00000 (Output / Holding Registers) OUT (as 32-bit floating-point)
double in	3:00000 (Input Registers) IN (as 64-bit integer)
double out	4:00000 (Output / Holding Registers) OUT (as 64-bit integer)
string in/ out	Special String Register IN/OUT (Control Design PLCs only)

You can also enter the data type manually, using the hwdatatype channel keyword (for a definition of this keyword, see Chapter 1 Vgen Keywords in your Vsystem Vaccess Reference Guide). This might be useful to read a hardware integer into a Vsystem floating-point channel for the purpose of conversion. You might also use this feature if you want to read an input from an Output/Holding Register rather than the Input Registers. Table 3-7 shows the register tables accessed by the various hwdatatypes.

Hwdatatypes and register tables

hwdatatype	Register Table
1	1:00000 (Discrete Inputs) IN (as binary)
2	3:00000 (Input Registers) IN (as int)
3	3:00000 (Input Registers) IN (as unsigned int)
4	3:00000 (Input Registers) IN (as 32-bit floating-point)
5	0:00000 (Discrete Outputs / Coils) OUT (as binary)
6	4:00000 (Output / Holding Registers) OUT (as int using Modbus command 6)
7	4:00000 (Output / Holding Registers) OUT (as unsigned int)
8	4:00000 (Output / Holding Registers) OUT (as 32-bit floating-point)
9	3:00000 (Input Registers) IN (as 64-bit integer)
10	4:00000 (Output / Holding Registers) OUT (as 64-bit integer)
11	Special String Register IN (Control Design PLCs only)
12	Special String Register OUT (Control Design PLCs only)
13	Special SDI Register IN (Control Design PLCs only)

14	Special SDI Register OUT (Control Design PLCs only)
15	4:00000 (Output / Holding Registers) IN (as integer using Modbus command 16)
16	4:00000 (Output / Holding Registers) OUT (as integer) 16
17	0:00000 (Discrete Outputs / Coils) IN (as binary)
18	4:00000 (Output / Holding Registers) IN (as 32-bit floating-point)
19	4:00000 (Output / Holding Registers) OUT (as 32-bit floating-point)
20	4:00000 (Output / Holding Registers) IN (as 64-bit integer)
21	4:00000 (Output / Holding Registers) OUT (as 64-bit integer)

Modicon-Style Addressing

Please note that the Vsystem Modbus Scanner starts with register 1 and does not use the Modicon convention. To convert Modicon register numbers to Vsystem register numbers, subtract

- 1 for 0:00000, Discrete Outputs/Coils
- 10001 for 1:00000, Discrete Inputs
- 30001 for 3:00000, Input Registers
- 40001 for 4:00000, Output/Holding Registers

For example, if your documentation refers to register numbers like 40401, you will enter "400" as the address and then look up your desired data type in the hwdatatype table above. If you wanted to access 40401 as a Float Input, your channel entry in your .adb file might look like this:

$FLOAT_IN1

real

in

[...]

HPARAM1 400

HWDATATYPE 18

The entry for 31234, accessed as an Integer Input, might look like this:

$INT_IN1

int16

in

[...]

HPARAM1 1233

HWDATATYPE 6

Or, because the 30000 register table is the default for an int16 channel, you could leave off the HWDATATYPE in this case.

Binary Addressing

For binary (Discrete) addressing, you must enter the address of the bit you want to read or set. Again, in the following formula, the Vmodbus Scanner starts with register 1. The general formula is:

*(16 * (register number - 1)) + (bit number)*

For example:

Register 3, bit 2 (a.k.a. 00034) is (16 * 2) + 2 = 34.

Register 624, bit 15 (a.k.a. 09983) is (16 * 623) + 15 = 9983.

Troubleshooting Addressing

When setting up a new Modbus system, or converting an existing Modbus system to Vsystem, you may need to experiment with addressing. Because not all vendors use the same Modbus addressing conventions, off-by-one errors and/or Invalid Address replies from the PLC can occur. If you have trouble determining the correct addresses, it may help to determine the values of the first few registers in each table, and then poll them separately using the Vmodbus Scanner. This way, you can work out which addresses correspond to which values. Then you can extend this relation to the entire system.

If you continue to encounter Invalid Address errors, you might double-check the hwdatatype on the channel. The most common cause of this error is attempting to read and/or write an address in the wrong table, as some PLCs do not support all Modbus tables and commands. If you continue to encounter this error, see the table below for a list of Modbus commands sent by each hwdatatype. Check to make sure the hwdatatypes you are using are compatible with your PLCs. You may also need to set the *See class* See class and/or *See enron* See enron qualifiers to match your PLCs' command set.

Please note that the *class, See control_design* See control_design, and See ic_techSee ic_tech qualifiers change the Modbus commands used to read and write registers. See Table 3-8 for details.

Modbus commands used to read/write registers

hwdatatype	Class 1 Command	Class 0 Command	Details
1 (bin in)	*2*	*3*	*/d16 option changes class0 PLCs to command 3, class1 PLCs to 4.* *Control design PLCs use command 3*
2 (int in)	*4*	*3*	*Control design PLCs use command 3*
3 (uint in)	*4*	*3*	*Control design PLCs use command 3*
4 (float in)	*4*	*3*	*Control design PLCs use command 3*
5 (coils out)	*5*	*5*	*Control design PLCs use commands 3 and 16 to read-modify-write*
6 (holding out int, cmd 6)	*6*	*6*	*Control design PLCs use command 16*
7 (holding out uint)	*16*	*16*	-
8 (holding out float)	*16*	*16*	-
9 (64-bit int in)	*4*	*3*	*Control design PLCs use command 3*
10 (holding out 64-bit int)	*16*	*16*	-

11 (string out)	*3*	*3*	*Control design PLCs only*
12 (string in)	*3*	*3*	*Control design PLCs only*
13 (SDI out)	*3*	*3*	*Control design PLCs only*
14 (SDI in)	*3*	*3*	*Control design PLCs only*
15 (holding in int)	*3*	*3*	-
16 (holding out int)	*16*	*16*	-
17 (coils in)	*1*	*1*	-
18 (holding in float)	*3*	*3*	-
19 (holding out float)	*16*	*16*	-
20 (holding in 64-bit int)	*3*	*3*	-

Vsystem OPC Utilities

This chapter discusses the Vsystem OPC utilities. The Vsystem OPC utilities comprise the following three components:

- Vsystem VOPC Scanner, which allows the exchange of data between OPC data sources and Vsystem databases.
- Vsystem OPC Vdb Server, which allows the exchange of data between Vsystem databases and OPC clients.
- Vsystem OPC Test Client a test utility that may be used to diagnose problems with the Vsystem scanner and server, as well as with third-party OPC servers.

The VOPC Scanner provides a bidirectional connection between third-party OPC data sources and Vsystem databases. Input channels are periodically updated with values read from OPC servers. Values of output channels are sent to OPC servers when they change. A single VOPC Scanner process can handle channels from multiple Vsystem databases and one OPC server.

Associating Vsystem Channels With VOPC Items

A channel is associated with a VOPC item using the database fields shown in Table 2. The Field column shows the default field that serves the given function. The default mapping between functions and fields can be overridden by using the See fields qualifierSee fields.

Table 2 – Functions and associated database fields

Function	Field	Usage
ItemName	HM	Defines the VOPC item name.
PLCName	-	Defines the name of a group of channels the channel belongs to. Different groups enable you to use multiple VOPC Scanner processes with the same database. For more information, see the description of the See PLC qulifier See PLC.
ScanRate	HPARAM1	Defines the scan rate of the channel, in milliseconds. If the value of this field is 0, 1000 is assumed.

Vsystem VOPC Scanner

To start the VOPC Scanner from the command line

- Enter:

```
vopc_scanner -database database1 -server OPC-server-name
      [-controlChannel string-channel-name] [-engine Vczar-engine-name]
      [-fields field1 ...] [-node OPC-server-node]
      [-noengine] [-PLC PLC1 ...)
```

Qualifiers

When starting the VOPC Scanner, you can use the qualifiers defined below, preceded by a "/" or a "-" character.

Note If the "-" form is used, the "=" character is not required. Multiple values for qualifiers specified without the "=" character should be written without parentheses and commas.

controlChannel

controlChannel=string-channel-name

Use this qualifier to specify a control channel for the VOPC Scanner. If you enter *exit* into the control channel, the VOPC Scanner process exits itself.

database

database=(name,...)

Use this qualifier to specify one or more Vsystem databases that need to be scanned.

engine

engine=name

Use this qualifier to specify the name under which the scanner registers with the Vczar server. If this name is not specified, "VOPC Scanner" is used.

fields

fields=(function:field,...)

This qualifier enables you to redefine the fields used to associate Vsystem channels with VOPC items.

Examples:

-fields ItemName:sparam1 ScanRate:hparam10

/fields=(ItemName:sparam1,ScanRate:hparam10)

node

node=name

Use this qualifier to specify the node on which the VOPC server is running. The default value is local node.

noengine

Use this qualifier to tell the scanner not to connect to the Vczar server.

PLC

PLC=(name,...)

Use this qualifier to specify one or more groups of channels to be scanned. The groups should be defined using the See PLCName database fieldSee Associating Vsystem Channels With VOPC Items. If this qualifer is specified, an association for the PLCName field must be defined using the See fields qualifierSee fields.

server

server=name

Use this qualifier to specify the VOPC server.

show

show=(option,...)

Use this qualifier to request that certain information be printed on the standard output. The supported option is *Scans* , which provides information about scan groups and rates.

Examples:

-show scans

/show=scans

Vsystem OPC Vdb Server

The VOPC Server allows third-party OPC clients to access data in Vsystem databases.

Default Installation

By default, the server is installed as a Windows service and is configured to serve all Vsystem channels in read-write mode.

Testing the Server

After installing VOPC, use the test client to make sure that everything is okay. For example, try the following command to read the value of a Vsystem channel:

```
vopc_test read Vistanm.OPC.Vdb DbName::ChannelName
```

where `DbName` is an existing Vsystem database with a channel `ChannelName`.

If the test client works, try your client. As you can see from the above command line, the name of the Vsystem OPC server is `Vistanm.OPC.Vdb`.

Configuring the Server

By default, the server is configured to serve all Vsystem channels in read-write mode. This configuration can be changed by using the options described below. First you remove the server from the list of installed Windows services by using the `vopc_server` command with the `remove` qualifier. Then you reinstall the server using the `vopc_server` command with the `install` qualifier and one or more configuration qualifiers.

VOPC Server Qualifiers

You should specify the `vopc_server` command with the `install` qualifier on one line, with a space, followed by "`-`", followed by another space, between `-install` and the rest of the qualifiers.

```
vopc_server -remove
vopc_server -install -
[-database database1 [...]]
[-channel channel1 [...]]
[-readonly]
[-trace n]
```

database

```
database=(database[,...])
```

Use this qualifier to tell the server to serve channels from the specified databases only.

channel

```
channel=(channel[,...])
```

Use this qualifier to tell the server to serve the specified channels only. A channel specification may include an `*` character. The specification `-channel=mydb::*` is equivalent to `-database=mydb`.

controlChannel

```
controlChannel=string-channel-name
```

Use this qualifier to specify a control channel for the VOPC Server. The server accepts the following commands using the control channel:

exit	The server process exits itself upon getting this command.
trace=number	This command dynamically changes the trace values. (For more information, see See traceSee trace.)

readonly

Use this qualifier to tell the server to serve all channels in read-only mode.

trace

```
trace=n
```

This qualifier is used for debugging. It tells the server to log certain debugging information into the log file. When the server is run as a service, the file is located in the standard Vsystem log directory, "%vsys_root %"\your-host-name\log, and is called vopc_server.log. Alternatively, the server can be run from a command line window for debugging, as described in.

The value that can be specified with this qualifier is the sum of one or more of the following values:

1	Trace AddItem calls.
2	Trace almost all method calls.
4	Trace constructor/destructor calls.
8	Trace QueryInterface calls.
32	Direct output to MS-DOS Console window instead of the log file.
64	Trace vdb put calls.

Running the Server From a DOS Window

For debugging, it may be more convenient to run the server in the command line mode from a DOS window. You can do this using the following commands.

```
vopc_server -register -noservice
vopc_server -nodaemon [Qualifiers]
vopc_server -register -service
```

With the `nodaemon` option, you can use the same qualifiers as with the `install` option.

Note After running the server in DOS mode, don't forget to reregister it to run as a service by using the third command in the above list.

Vsystem OPC Test Client

VOPC Test Client is a test utility that can be used to diagnose problems with the Vsystem scanner and server, as well as with third party OPC servers.

Reading Items Using the VOPC Test Client

```
vopc_test read ServerName ItemName1[, ...]
[-async]
[-cache]
[-count=n]
[-itemCount=n]
[-nooutput]
[-statistics]
```

VOPC Test Client Qualifiers

Use the following qualifiers with the VOPC Test Client:

async

Use this qualifier to read using the IOPCAsyncIO2 interface. If `async` is not specified, the IOPCSyncIO interface is used.

cache

Use this qualifier to tell the client to read from the OPC server CACHE . The default is to read from DEVICE .

count

`count=n`

Use this qualifier to tell the client how many times the command should be executed. This qualifier can be useful for performance testing.

itemCount

`itemCount=n`

Use this qualifier to tell the client to read the specified number of items, with names itemName1 ... itemName n .

nooutput

Use this qualifier to tell the client not to display the data that is read.

statistics

Use this qualifier to display performance information.

Writing Items Using the VOPC Test Client

```
vopc_test write ServerName valueType ItemName1 Value1 [...]
```

Valid valueTypes are:

- boolean
- double
- float
- integer
- int16
- int8
- string
- uinteger
- uint16
- uint8

Vsystem Kingfisher Scanner

The Kingfisher Scanner provides a bidirectional connection between Kingfisher hardware and Vsystem databases. Input channels are periodically updated with values read from hardware, and values of output channels are written to hardware when they change. A single Kingfisher Scanner process can scan multiple RTUs, possibly connected to different communication ports (serial and TCP).

At this time, all nonevent log data that is available in the RTU is scannable and (if RTU permissions allow) writeable.

Note This chapter describes the use of Kingfisher protocol only in relation to Vista's Kingfisher Scanner. For information about Kingfisher protocol and setting up Kingfisher RTUs, you must contact the manufacturer, RTUnet (www.rtunet.com).

Associating Vsystem Channels with RTU Data Points

A channel is associated with a hardware data point using the database fields shown in Table 3. The Field column shows the database field that serves the given function. The file Kingfisher.def can be included in your database file to make defining the channels easier.

Table 3 – Functions and associated database fields

Function	Field	Usage
Scanner association	IOTYPE	Specifies the scanner that this channel uses. This field must be "Kingfisher Series II" for this scanner.
RTU name	MODULE	Defines the name of the RTU.
Data type	HWDATATYPE	Defines the type of the data point. Valid data type values can be found in the table below.
Register	HPARAM1	Defines the register of the data point inside the RTU.
Module	HPARAM2	Defines the module for analog and digital I/O.
Channel	HPARAM3	Defines the channel for analog and digital I/O.
Bit	HPARAM4	Defines the bit number used for digital I/O.
RTU	HPARAM5	Defines the RTU when reading network registers from the master RTU.

Table 4 lists the valid datapoint types for a Vsystem Kingfisher Scanner.

Table 4 – Valid datapoint types

Value	Name	Description
1	KF_DTYPE_AM_UINT16	Array module, unsigned int16, uses module and channel. Channel is the AI channel number.
2	KF_DTYPE_AM_INT16	Array module, signed int16
3	KF_DTYPE_AM_BINARY	Array module digital bit (hparam2 is bit number)
4	KF_DTYPE_DIGITAL_MODULE	Digital module
5	KF_DTYPE_LR_BINARY	Local register digital bit (hparam2 is bit number)
6	KF_DTYPE_LR_UINT16	Local unsigned 16-bit register
7	KF_DTYPE_LR_INT16	Local signed 16-bit registers
8	KF_DTYPE_LR_INT32	Local unsigned 32-bit register
9	KF_DTYPE_LR_UINT32	Local signed 32-bit register

10	KF_DTYPE_LR_FLOAT	32-bit local floating-point register
11	KF_DTYPE_TIMER	Timer registers
12	KF_DTYPE_MODULE_STATUS	Module status registers
13	KF_DTYPE_INACTIVITY_MONITOR	Inactivity monitors
14	KF_DTYPE_SYSTEM_REGISTER	System register
15	KF_DTYPE_CLOCK_REGISTER	Clock register
16	KF_DTYPE_PORT_REGISTER	Port registers
17	KF_DTYPE_RTU_NETINFO1	RTU network link info 1
18	KF_DTYPE_RTU_NETINFO2	RTU network link info 2
19	KF_DTYPE_NET_REGISTER	Network registers
20	KF_DTYPE_NET_ANALOG	Network analog registers
21	KF_DTYPE_NET_DIGITAL	Network digital registers
22	KF_DTYPE_NET_FLOAT	Network analog registers read as float
23	KF_DTYPE_PIC_DATA	Video picture data, an unsigned integer (8 bit) array channel (UINT8)
24	KF_DTYPE_PIC_SNAP	Snap picture channel
25	KF_DTYPE_PIC_CONFIG	Configure video
26	KF_DTYPE_NR_BINARY	Read network registers as binary values
27	KF_DTYPE_NET_AM_ANALOG	Read network analogs using channel and module

Note that if the HWDATATYPE field is 0, the TYPE and IN / OUT fields are used to determine the datapoint type.

Sample Database

```
include kingfisher.def

$pic
uint8 30000
in
timestamp_single
KF_SCANNER
KF_DTYPE_PIC_DATA
KF_RTU rtu1

$int_1
int16 in
value 0
KF_SCANNER
KF_REGISTER 1
KF_NETRTU 0
KF_RTU rtu1
```

```
KF_DTYPE_NET_REGISTER

$real_1
real in
slope 0.3
value 0
KF_SCANNER
KF_REGISTER 34
KF_NETRTU 1
KF_RTU rtu1
KF_DTYPE_NET_ANALOG

$real_2
real in
slope 0.3
value 0
KF_SCANNER
KF_REGISTER 35
KF_NETRTU 1
KF_RTU rtu1
KF_DTYPE_NET_ANALOG

$bin_1
binary out
value stop
b0 stop
b1 run
KF_SCANNER
KF_REGISTER 4
KF_BIT 3
KF_NETRTU 1
KF_RTU rtu1
KF_DTYPE_NET_DIGITAL
```

Defining the Hardware Configuration in a File

This is an optional step. No configuration file is required when one or more identical RTUs are connected to a single communication port. If multiple communication ports are involved, or if RTUs have different characteristics, the RTUs and ports must be defined in a configuration file, and the configuration file must be specified on the command line with the See configFile qualifierSee configFile.

Comment

Comments are lines with the "#" character in the first position.

RTU Description

RTU descriptions are lines that start with an RTU keyword. The format of an RTU description is:

```
RTU name [options]
```

Similar to qualifiers, you can precede the RTU keywords with a "/" or a "-" character. Supported keywords are defined below. Several keywords can also be defined in the port description or on the command line. If such a keyword is not specified, it is defaulted to the value specified in the port description or on the command line. If such a keyword is specified in several places, the RTU's specification overrides the others.

configMinute

Use the `configMinute` keyword to specify the minute of the day at which configuration data is obtained for the RTU. The allowable range is 1 to 1440. The default (zero or unspecified) is the RTU address doubled. For example, RTU address 33 would have its configuration scanned at 1:06 AM , local time.

eventInterval

Use the `eventInterval` keyword to define the interval at which event log scans are performed for the RTU. See the See eventInterval qualifierSee eventInterval for allowable values and defaults.

imageInterval

Use the `imageInterval` keyword to define the interval at which full image scans are performed for the RTU. See the See imageInterval qualifierSee imageInterval for allowable values and defaults.

picChannel

```
picChannel=3
```

Use the `picChannel` keyword to specify the channel on the RTU the video is connected to.

picInterval

Use the `picInterval` keyword to specify the time, in milliseconds, at which to take a picture. For example, `picInterval=4000` specifies that a picture will be taken every four seconds.

picQuality

Use the picQuality keyword to specify the picture quality, using a value from 0 to 100. The picture is in JPEG format; hence, a smaller value decreases the picture quality and the size in bytes, and a larger value increases the picture quality and the size in bytes. For example, picQuality=40 will give medium picture quality.

picSize

```
picSize=16
```

Use the `picSize` keyword to specify the size of the picture:

Value	Video	Size of picture
0	PAL	352 W x 288 H

1	PAL	176 W x 144 H
2	PAL	88 W x 72 H
14	NTSC	320 W x 240 H
15	NTSC	160 W x 120 H
16	NTSC	80 W x 60 H

port

Use the port keyword to define the name of the communication port the given RTU is connected to. The port should be defined in the configuration file with a port description. If this keyword is not specified, the necessary communication port characteristics are defaulted to those specified on the command line.

timeout

Use the timeout keyword to define the time out for the RTU. The default is the port default. See the See timeout qualifierSee timeout for allowable values and defaults.

unit

Use the unit keyword to define the unit (slave) address of the RTU. The default value is 1.

updateInterval

Use the updateInterval keyword to define the interval at which the update counters are obtained for the RTU. See the See updateInterval qualifierSee updateInterval for allowable values and defaults.

via

Use the via keyword to define the unit address of an RTU that is this slave RTU's master. The default value is 0, denoting a direct connection. If this is specified, the host parameter is ignored.

Port description

Port descriptions are lines that start with a port keyword. The format of a port description is:

```
PORT name [options]
```

Similar to qualifiers, you can precede the port keywords with a "/" or a "-" character. Supported keywords are listed below. Several keywords can also be defined in the RTU description or on the command line. If such a keyword is not specified, it is defaulted to the value specified on the command line. If such a keyword is specified in several places, the port's value overrides the value specified on the command line, but is itself overridden by the value specified in the RTU description.

Example

```
# This is an example of a configuration file
port p1 -ctype TCP -device host1 -evfile=d:\logs\events.out -evdatabase=evdb
port p2 /ctype=SERIAL /device=com2 /baudRate=9600 /stop=1 /parity=ODD
rtu area1 -port=p1 -unit=2
rtu area1.1 -port=p1 -unit=3 -via=2
```

```
rtu area2 -port=p2 -unit=3
```

Starting the Kingfisher Scanner Process

To start the Kingfisher Scanner from the command line

```
vkf [qualifiers]
```

Qualifiers

When starting the Kingfisher Scanner, you can use the following qualifiers, preceded by a "/" or a "-" character.

Note If the "-" form is used, the "=" character is not required. Multiple values for qualifiers specified without the "=" character should be written without parentheses and commas.

baudRate

`baudRate=n`

Use this qualifier to define the baud rate for the serial device specified with the See device qualifier. The `baudRate` qualifier is not implemented on OpenVMS.

configFile

configFile=filename

Use this qualifier to specify a configuration file for the Kingfisher Scanner.

controlChannel

controlChannel=string-channel-name

Use this qualifier to specify a control channel for the Kingfisher Scanner. If you enter exit into the control channel, the Kingfisher Scanner process exits itself.

ctype

ctype=communication-type

Use this qualifier to define the communication type of the default device (the device specified with the See device qualifier). Supported values include SERIAL and TCP .

database

database=(database,...)

Use this qualifier to specify one or more Vsystem databases that need to be scanned.

device

device=name

Use this qualifier to specify the communications device for TCP or serial communications for a port. For TCP communications, this is the hostname or IP address of an RTU, and is the only IP address associated with a port.

The device specified is used to access any RTU for which there is no explicitly specified port in the configuration file, or if no configuration file is used. The device name is operating-system and communication-type specific. For example, for serial communications, it could be "com1:" on Windows NT or "/dev/ttya0" on a UNIX system.

evDatabase

evDatabase=database

Use this qualifier to define the database for RTU event-log-entry operations. This operation is required for event processing to a file, whether or not Vsystem archiving is used. The database must have the channels, as shown in See RTU Event LoggingSee RTU Event Logging.

eventInterval

eventInterval=n

Use this qualifier to define the interval at which event-log-entry scans are performed for the RTU. The allowable range is 100 milliseconds to 1000 hours. The default is 5 seconds.

evFile

evFile=filename

Use this qualifier to define a file to which RTU event-log entries may be written. The entries are written with a tab between the fields, and the file may be exported to applications such as Microsoft Excel. The event file line format is shown in Table 4-3.

Event File Line Format

Field	Format
1	*Time string, "dd-mmm-yyyy hh:mm:ss"*
2	*Port name, quoted string*
3	*Device, quoted string*
4	*RTU unit number*
5	*RTU name, quoted string*
6	*Type, 0 through 4*

7 *System log reference (type 0) or RTU data ID (types 1 through 4)*

8 *RTU ladder logic ID, quoted string*

9 *Priority, 0 through 7*

10 *User type, 1 through 31*

11 *For types 1 through 4, the value; for type 0, this field is not output*

12 *For type 2, the bit number; for the other types, this field is not output*

evMaxPollEntries

evMaxPollEntries=n

RTU event logs may comprise thousands of entries, which could cause the event processing to spend an inordinate amount of time scanning the event entries for a single RTU. This qualifier is intended to avoid this behavior by specifying the maximum number of event entries to be obtained during a single poll of an RTU. The default is 50.

imageInterval

imageInterval=n

Use this qualifier to define the interval at which full image scans are performed for the RTU. The allowable range is 100 milliseconds to 1000 hours. The default is 30 seconds.

master

master=n

Use this qualifier to define the Kingfisher protocol address of this port. Allowable values are 251 to 255, with the default being 255.

nobanner

Use this qualifier to suppress display of the startup banner.

parity

parity=string

Use this qualifier to define the data parity for the serial device specified with the See device qualifierSee device. Supported values are *NONE* , *EVEN* , and *ODD* . This qualifier is not implemented on OpenVMS.

RTU

RTU=(name[:unit],...)

Use this qualifier to specify one or more RTUs to be scanned. The name of an RTU may be followed by a colon, which in turn is followed by the unit (slave) address. If the unit address is not specified, the unit address from the configuration file is used. If the RTU is not defined in the configuration file, or if no configuration file is used, the unit address is defaulted to 1.

If both the See configFileSee configFile and *RTU* qualifiers are specified on the command line, values defined by the *RTU* qualifier take precedence.

show

show=(option,...)

Use this qualifier to request that certain information be printed on the standard output. Supported options are shown in Table 4-4.

Standard output print options

Option	Prints
RTUs	*Shows a list of all scanned channels for configured RTUs.*
ports	*Information about ports, which includes the number and names of RTUs connected to this port, the port's communications parameters, and the number of data points scanned using this port.*
scans	*Information about scan lists.*
decimal	*Prints addressing data in decimal instead of hexadecimal (the default).*

Examples:

-show rtus ports scans

/show=(rtus,ports,scans)

stop

stop=n

Use this qualifier to define the number of stop bits for the serial device specified with the See device qualifierSee device. The *stop* qualifier is not implemented on OpenVMS.

systemID

This qualifier enables you to change the system ID.

timeout

timeout=n

Use this qualifier to define the time out for the RTU. Allowable values are from 100 milliseconds to 1000 hours. The default is 2 seconds.

updateInterval

updateInterval=n

Use this qualifier to define the interval at which the update counters are obtained for the RTU. The allowable range is 100 milliseconds to 1000 hours. The default is 1 second.

RTU Event Logging

Event logging is supported in Kingfisher Scanner versions 4.3.0.2 and later. The following table is a description of the operation and use of the event scanning.

The channels in the supplied database, vkfevent.adb, have the names and roles shown in Table 4-5.

Database event channel names and functions

Channel	Function
eventCountScanned	*A timestamped array of integers, indexed by RTU unit (0..256). The high-order word is the numbe of entries that have been scanned from the RTU. The low-order word is the index to use for a "get next" scan of the RTU. The timestamp for the RTU is the last time seen from the RTU. This channe is not intended for logging use; it is used by the scanner for housekeeping.*
eventPort	*Port name from which the event originated.*
eventDevice	*Device name from which the event originated.*
eventRTU	*RTU number from which the event originated.*
eventRTUName	*Name of the RTU from which the event originated.*

eventTime	*Date and time at which the event occurred.*
eventType	*RTU event entry type, value 0 to 4:* *0 is a system log* *1 is a 16-bit register* *2 is a register bit* *3 is a 32-bit integer value* *4 is a 32-bit floating-point value*
eventBit	*For event type 2, this is the bit number for the bit register.*
eventPriority	*Priority of event, 0..7.*
eventUsertype	*User type of event, 0..31*
eventID	*System log index or RTU data ID, as documented in RTUnet's document "Implementing a SCADA Interface," page 7. This field does not have the type in the high-order two bits*
eventIDString	*Kingfisher ladder logic reference for data ID. See the Kingfisher toolbox documentation.*
eventAnalogValue	*Value for 16-bit register event.*
eventBinaryValue	*Value (0 or 1) for binary event.*
eventLongIntValue	*Value for 32-bit register event.*
eventFloatValue	*Value for floating-point event.*

*eventLogTrigger** *Channel that gets set to 1 when an event is written to the database.*

* Event logs can be archived by using a triggered archive with eventLogTrigger as the trigger channel, or to a file, with tab-separated fields. Either or both methods may be used.

Vsystem RTP Scanner

The VRTP Scanner provides a bidirectional connection between RTP2000 hardware and Vsystem databases. Input channels are periodically updated with values read from hardware, and values of output channels are written to hardware when they change. A single VRTP Scanner process can scan multiple RTP2000 target nodes, which possibly can be connected to different communication ports.

The VRTP Scanner currently supports the following data types:

• Virtual coils -- input and output 16-bit variables

• Input and output 32-bit float variables -- input and output

All communications with the RTP targets is done via the UDP protocol layered on IP.

Associating Vsystem Channels With RTP Data Points

A channel is associated with a hardware data point by using the database fields shown in Table 5-1. The Field column shows the default field that serves the given function. The default mapping between functions and fields can be overridden by using the See fields qualifierSee fields.

Table 5 – Functions and associated database tields

Function	Field	Usage
PLCName	MODULE	Defines the name of the RTP target node.
objectName	IOTYPE	Defines the name of the RTP object. If this field is not specified, the NAME field is used.
scanRate	HPARAM6	Defines the scan rate of the channel. By default the scan rate is in milliseconds, but you can change this by using the See rate qualifier See rate . If the value of this field is 0, 1000 is assumed.

Defining the Hardware Configuration in a Configuration File

This is an optional step. No configuration file is required when one or more identical RTP target nodes are connected to a single communication port. If multiple communication ports are involved, or if RTP targets have different characteristics, the RTP targets and ports must be defined in a configuration file, and the configuration file must be specified on the command line with the See configFile qualifierSee configFile.

Comment

Comments are lines with the "#" character in the first position.

RTP Target Node Description

RTP target node descriptions are lines that start with a PLC keyword. The format of an RTP target description is:

```
PLC name [keywords]
```

You can use the following keyword, preceded by a "/" or a "-" character, in an RTP target node description. Several of these keywords can be defined in the port description or on the command line. If such a keyword is not specified in the RTP target node description, it is defaulted to the value specified in the port description or on the command line. If such a keyword is specified in several places, the PLC's keyword overrides the others.

port

Use this keyword to define the name of the communication port the given RTP target is connected to. The port should be defined in the same configurations file with a port description. If this option is not specified, the necessary communication port characteristics are defaulted to those specified on the command line.

Port description

Port descriptions are lines that start with a port keyword. The format of a port description is:

PORT name [keywords]

You can use the following keyword, preceded by a "/" or a "-" character, in a port description. Several of these keywords can be defined in the PLC description or on the command line. If such a keyword is not specified, it is defaulted to the value specified on the command line. If such a keyword is specified in several places, the port's keyword overrides the value specified on the command line, but is itself overridden by the value specified in the PLC RTP description.

device

See See deviceSee device.

Starting the VRTP Scanner Process

To start the VRTP Scanner from the command line

- Enter:

vrtp [qualifiers]

When starting the VRTP Scanner, you can use the following qualifiers, preceded by a "/" or a "-" character.

Note If the "-" form is used, the "=" character is not required. Multiple values for qualifiers specified without the "=" character should be written without parentheses and commas.

Qualifiers

You can use the following qualifiers, preceded by a "/" or "-" character, with the *vrtp* command:

See configFileSee configFile

See controlChannelSee controlChannel

See databaseSee database

See deviceSee device

See fieldsSee fields

See nobannerSee nobanner

See PLCSee PLC See rateSee rate

See showSee show

configFile

configFile=filename

Use this qualifier to specify a configuration file for the VRTP Scanner.

controlChannel

controlChannel=string-channel-name

Use this qualifier to specify a control channel for the VRTP Scanner. If you enter *exit* into the control channel, the VRTP Scanner process exits itself.

database

database=(database,...)

Use this qualifier to specify one or more Vsystem databases that need to be scanned.

device

device=name

Use this qualifier to specify the default communication device. The default device is used to access any RTP target for which there is no explicitly specified device in the configuration file, or if no configuration file is used. The device is either the hostname or IP address of the RTP target node.

fields

fields=(function:field,...)

This qualifier enables you to redefine the fields used to associate Vsystem channels with RTP objects.

Examples:

-fields plcName:hm objectName:sparam1 scanRate:hparam1

/fields=(plcName:hm,addr:sparam1,scanRate:hparam1)

nobanner

Use this qualifier to suppress display of the startup banner.

PLC

PLC=(name[:unit],...)

Use this qualifier to specify one or more RTP targets to be scanned. Only channels with the *plcName* field matching the names specified with this qualifier are scanned.

rate

rate=n

Use this qualifier to specify the scan rate unit; the default unit is 1 millisecond. The scan rate of a channel, in milliseconds, is determined by multiplying this value by the value of the *scanRate* field.

show

show=(option,...)

Use this qualifier to request certain information be printed on the standard output. Supported options are shown in Table 5-2See Standard output printing options.

Standard output printing options

Option	Prints
PLCs	*The list of all scanned data points, sorted by data-point type.*
ports	*Information about ports, which includes the number and names of RTP targets connected to this port and the number of data points scanned using this port.*
scans	*Information about contiguous scan blocks and scan rates.*

Examples:

-show plcs ports scans

/show=(plcs,ports,scans)

File-naming Conventions

Vsystem has adopted the naming conventions shown in Table 6 for file extensions:

Table 6 – Vsystem file-naming conventions

`.adb`	ASCII database
`.idb`	Intermediate database
`.seq`	Sequence file (Vscript)
`.varc`	Vista archive file
`.vcd`	Vista control display
`.vchf`	Vista channel files (containing a list of channels)
`.vdef`	Configuration files
`.vexe`	Compiled script
`.vsf`	Vista segment file
`.vsl`	Vista symbol library
`.vwsp`	Vista workspace file

Field Aliases

You may want to give your fields names that are more meaningful to the operators than, say, `cparam1`. You can accomplish this by defining aliases for your fields in a file named `vdb_field_alias.vdef` and saving this file in the Vsystem root directory. In this file, create a list of alias and field pairs, as shown in the following example:

Alias	Field
max	cparam1
min	cparam2
slow	srate
upper_alarm	ua

You can use tabs or spaces to separate the alias from the field; however, you cannot use any spaces in the alias itself. When defining aliases for fields in the file, you can have only one alias and field pair per line.

To use the defined aliases, Vsystem applications first refer to the environment variable `VDB_ALIAS_FILE` for the field alias definition file. If this file is not found, the applications check the Vsystem root directory for the `vdb_field_alias.vdef` file. Any Vsystem application using the `vdb_fields_info` routine automatically refers to the defined aliases. For example, when adding a text control tool in Vdraw, you can specify `db::chan@max`. (For information about the vdb_fields_info routine, see Chapter 4, Library of Database Access Routines , in your Vsystem Vaccess Reference Guide.

Accessible Channel Fields

To access a particular channel field in applications that support field access, place an @ character and the name of the channel field after the channel name. This feature is useful for viewing and changing database information. For example, if you specify the channel name DEMO:INTEGER_INI_@UA, the upper alarm limit is specified (UA), enabling you to get and set that field in a channel.

This appendix provides a table of the field name keywords and shows which fields can be accessed by each of the Vsystem channel types. The channel types are represented by a single letter, as shown in the table below:

B	Binary channels
D	Double-precision floating-point channels
I	Integer type channels, including int16, int8, integer, uint16, uint8, and unsigned integer
R	Real or single-precision floating-point channels
S	String channels
T	Time channels

The Edit column shows whether the field is editable.

Field	B	D	I	R	S	T	Edit	Equivalent Field Mask
alarm	x	x	x	x			yes	VFLD_M_ALARM_ENABLED
alarm_ack	x	x	x	x			yes	VFLD_M_ALARM_ACK
alarmlabel	x	x	x	x	x	x	yes	VFLD_M_ALARMLABEL
alarm_label	x	x	x	x	x	x	yes	VFLD_M_ALARM_LABEL
alarm_priority	x	x	x	x			yes	VFLD_M_ALARM_PRIORITY
alarmtype			x				yes	VFLD_M_AT
alarm_type			x				yes	VFLD_M_AT
alarm1	x	x	x	x	x	x	no	VFLD_M_ALARM1
alarm2	x	x	x	x	x	x	no	VFLD_M_ALARM2
alarmrefchan		x	x	x			no	VFLD_M_ALARM_REFCHAN
alarms_delayed	x	x	x	x			yes	VFLD_M_ALARMS_DELAYED
area	x	x	x	x	x	x	yes	VFLD_M_AREA
automatic	x	x	x	x	x	x	yes	VFLD_M_AUTOMATIC
b0	x						yes	VFLD_M_B0
b1	x						yes	VFLD_M_B1
ba	x						yes	VFLD_M_BA
ba_delay	x						yes	VFLD_M_BA_DELAY
ba_state	x						no	VFLD_M_BA_STATE
balarm	x						yes	VFLD_M_BALARM
chan_access	x	x	x	x	x	x	no	VFLD_M_CHAN_ACCESS

chix	x	x	x	x	x	x	no	VFLD_M_CHIX
chname	x	x	x	x	x	x	no	VFLD_M_CHNAME
chsize	x	x	x	x	x	x	no	VFLD_M_CHSIZE
chtype	x	x	x	x	x	x	no	VFLD_M_CHTYPE
clip		x	x	x			no	VFLD_M_CLIPPING
clip_enabled		x	x	x			yes	VFLD_M_CLIPPING_ENABLED
constant	x	x	x	x	x	x	yes	VFLD_M_CONSTANT
conversion	x	x	x	x	x	x	no	VFLD_M_CONVERSION
convert	x	x	x	x	x	x	yes	VFLD_M_CONVERT_ENABLED
convert_failed	x	x	x	x	x	x	no	VFLD_M_CONVERT_FAILED
count	x	x	x	x	x	x	yes	VFLD_M_SURVEY_COUNT
cparamX	x	x	x	x	x	x	yes	VFLD_M_CPARAMY
(X can be any positive number; Y can be any number from 1 through 10; use VFLD_M_ECPARAM for accessing extended cparams)								
db_name	x	x	x	x	x	x	yes	VFLD_M_DB_NAME
delta		x	x	x			yes	VFLD_M_DELTA
enter	x	x	x	x	x	x	no	VFLD_M_ENTER
entlve	x	x	x	x	x	x	yes	VFLD_M_ENTLVE
exttoint	x	x	x	x	x	x	yes	VFLD_M_PUT_ROUTINE
format		x	x	x			yes	VFLD_M_FORMAT
frate	x	x	x	x	x	x	yes	VFLD_M_FAST_SURVEY
get_routine	x	x	x	x	x	x	yes	VFLD_M_GET_ROUTINE
handler	x	x	x	x	x	x	yes	VFLD_M_HANDLER
hardware	x	x	x	x	x	x	no	VFLD_M_HARDWARE
hfunction	x	x	x	x	x	x	yes	VFLD_M_HFUNCTION
hilim		x	x	x	x	x	yes	VFLD_M_HEQLIM
hm	x	x	x	x	x	x	yes	VFLD_M_HTYPE
hparamX	x	x	x	x	x	x	yes	VFLD_M_HPARAMY
(X can be any positive number; Y can be any number from 1 through 10; use VFLD_M_EHPARAM for accessing extended hparams)								
imatch			x				yes	VFLD_M_MV
in	x	x	x	x	x	x	yes	VFLD_M_IN
int_value	x	x	x	x	x	x	yes	VFLD_M_INT_VALUE
intercept		x	x	x			yes	VFLD_M_OFFSET
interest	x	x	x	x	x	x	no	VFLD_M_INTEREST
inttoext	x	x	x	x	x	x	yes	VFLD_M_GET_ROUTINE
io_error	x	x	x	x	x	x	yes	VFLD_M_IO_ERROR
la		x	x	x			yes	VFLD_M_LA
la_delay		x	x	x			yes	VFLD_M_LA_DELAY

la_delta		x	x	x			yes	VFLD_M_LA_DELTA
la_kintercept		x	x	x			yes	VFLD_M_LA_KINTERCEPT
la_kmax		x	x	x			yes	VFLD_M_LA_KMAX
la_kmin		x	x	x			yes	VFLD_M_LA_KMIN
la_kslope		x	x	x			yes	VFLD_M_LA_KSLOPE
la_state		x	x	x			yes	VFLD_M_LA_STATE
label		x	x	x			yes	VFLD_M_LABEL
ld	x	x	x	x	x	x	no	VFLD_M_LDISPLIM
leave		x	x	x			yes	VFLD_M_LEAVE
limit	x	x	x	x	x	x	no	VFLD_M_LIMIT
lowlim		x	x	x			yes	VFLD_M_LEQLIM
lw		x	x	x			yes	VFLD_M_LW
lw_delay		x	x	x			yes	VFLD_M_LW_DELAY
lw_delta		x	x	x			yes	VFLD_M_LW_DELTA
lw_kintercept		x	x	x			yes	VFLD_M_LW_KINTERCEPT
lw_kmax		x	x	x			yes	VFLD_M_LW_KMAX
lw_kmin		x	x	x			yes	VFLD_M_LW_KMIN
lw_kslope		x	x	x			yes	VFLD_M_LW_KSLOPE
lw_state		x	x	x			yes	VFLD_M_LW_STATE
mv			x				yes	VFLD_M_MV
mv_state			x				no	VFLD_M_MV_STATE
num_channels	x	x	x	x	x	x	no	VFLD_M_NUM_CHANNELS
num_cparam	x	x	x	x	x	x	no	VFLD_M_NUM_CPARAM
num_hparam	x	x	x	x	x	x	no	VFLD_M_NUM_HPARAM
put_routine	x	x	x	x	x	x	yes	VFLD_M_PUT_ROUTINE
reader	x	x	x	x	x	x	no	VFLD_M_READER
refchan		x	x	x			no	VFLD_M_REFCHAN
sec_class	x	x	x	x	x	x	no	VFLD_M_SEC_CLASS
security	x	x	x	x	x	x	no	VFLD_M_SECURITY
size	x	x	x	x	x	x	no	VFLD_M_CHSIZE
slope	x	x		x			yes	VFLD_M_SLOPE
soft	x	x	x	x	x	x	yes	VFLD_M_SOFT
sparamX	x	x	x	x	x	x	yes	VFLD_M_SPARAMY
(X can be any positive number; Y can be any number from 1 through 10; use VFLD_M_ESPARAM for accessing extended sparams)								
srate	x	x	x	x	x	x	yes	VFLD_M_SLOW_SURVEY
subarea	x	x	x	x	x	x	yes	VFLD_M_SUBAREA

survey	x	x	x	x	x	x	yes	VFLD_M_NO_SURVEY
timestamp	x	x	x	x	x	x	yes	VFLD_M_TIMESTAMP_VALUE
timestamp_array	x	x	x	x	x	x	no	VFLD_M_TIMESTAMP_ARRAY
timestamp_disable	x	x	x	x	x	x	yes	VFLD_M_TIMESTAMP_UPDATE_DISABLE
timestamp_single	x	x	x	x	x	x	no	VFLD_M_TIMESTAMP_SINGLE
type	x	x	x	x	x	x	no	VFLD_M_CHTYPE
ua		x	x	x			yes	VFLD_M_UA
ua_delay	x	x	x	x			yes	VFLD_M_UA_DELAY
ua_delta		x	x	x			yes	VFLD_M_UA_DELTA
ua_kintercept		x	x	x			yes	VFLD_M_UA_KINTERCEPT
ua_kmax		x	x	x			yes	VFLD_M_UA_KMAX
ua_kmin		x	x	x			yes	VFLD_M_UA_KMIN
ua_kslope		x	x	x			yes	VFLD_M_UA_KSLOPE
ua_state		x	x	x			no	VFLD_M_UA_STATE
ud		x	x	x			yes	VFLD_M_HDISPLIM
units		x	x	x			yes	VFLD_M_UNITS
uw		x	x	x			yes	VFLD_M_UW
uw_delay		x	x	x			yes	VFLD_M_UW_DELAY
uw_delta		x	x	x			yes	VFLD_M_UW_DELTA
uw_kintercept		x	x	x			yes	VFLD_M_UW_KINTERCEPT
uw_kmax		x	x	x			yes	VFLD_M_UW_KMAX
uw_kmin		x	x	x			yes	VFLD_M_UW_KMIN
uw_kslope		x	x	x			yes	VFLD_M_UW_KSLOPE
uw_state		x	x	x			no	VFLD_M_UW_STATE
vm_delay			x				yes	VFLD_M_VM_DELAY
vm_state			x				yes	VFLD_M_VM_STATE