Zaradann Manual

Introduction

Zaradann is a flexible low-level tool for testing memory mapped hardware such as VME.

The main document window contains some control fields and a list of read, write and test operations to be executed on specified locations in memory. The list can be edited and saved as a document.

The list of operations can be executed once, several times or continuously. Changing values and/or errors can be highlighted.

There are a number of utility windows: one constructs CAMAC addresses (if CAMAC is being accessed via a MacCC or VME CAMAC branch driver); another searches through memory locations looking for valid addresses; another scans the Nubus slots to locate eg MICRON cards.

The various windows and menus are described in other Help topics.

File Menu

The **File** menu contains the following commands, most of which are the standard ones:

New: create a new list of address operations. The last file saved as prefs will be used as the template for the new file.

Open: open an existing list of address operations. If the *Option/Alt* key is down, the **Loop** command is executed after the file is opened.

Save: save the list to the file that was opened.

Save As: save the list to another file.

Revert to Saved: not yet implemented.

Save As Prefs: save the list to the Preferences file to be used as the default for New files in the future.

Save As C/C++: save the list as C or C++ source code. The flexibility of *Zaradann* gives rise to some overheads in executing the list of operations. For maximum speed, you can save the list as C/C++ and compile it separately. (So far only Read, Write and Test operations have been implemented).

Page Setup: choose page size, orientation, etc.

Print: print the list of address operations.

Quit: exit from the program.

Edit Menu

The **Edit** menu contains the standard editing commands. Note that they apply to whichever item(s) are currently selected, or which contain the insertion point cursor. Ie to edit the list of address operations, you must select one or more address lines. If the selection or insertion point is in one part of an address line, only that field will be edited.

Undo: undo the previous edit command. This is supported for all the standard edit commands and some of the special ones in the **Operations** menu.

Cut: remove the selected address line(s) from the list of address operations. The deleted lines are stored in the clipboard.

Copy: copy the selected address line(s) to the clipboard.

Paste: delete any currently selected address line(s) and replace them with the contents of the clipboard. If there is no selection, the clipboard contents are copied into the list at the current insertion point.

Clear: not implemented.

Select All: select all address line(s) in the list. One of the address lines must already be selected or contain the insertion point.

Show Clipboard: display the current contents of the clipboard.

Operations Menu

The **Operations** menu contains commands to execute the list of address operations as well as some special editing commands.

Loop: starts or stops looping through the list of address operations. This is the same as clicking the **Loop** button in the document window.

Execute: executes the selected address operation(s) once only.

Enable: enables all selected address operations. If the *Option/Alt* key is down, all operations in the document are enabled.

Disable: disables all selected address operations. If the *Option/Alt* key is down, all operations in the document are disabled.

Type: selects the type of operation to be performed. The available choices are Read, Write, Test (ie Write, Read back and check), Bit test, and Soak test (test all possible bit patterns). Further details about these operations are given in the Operations Field topic.

Length: set the word length of the operation. This can be 1, 2 or 4 bytes.

Special Menu

The **Special** menu accesses the utility windows. These are described in more detail in other Help topics.

Memory Scan: allows you to find valid ranges of memory addresses.

Nubus Scan: shows you which Nubus cards are installed in the Macintosh.

Camac Address: constructs VME addresses for Camac operations.

Memory Snoop: allows you to display the contents of contiguous ranges of memory addresses.

Window Introduction

There are a number of features common to many of the windows in the program. They are not exactly Macintosh standard features:

Numeric fields: most of the numeric fields can be displayed as either decimal or hexadecimal. The presence of a dollar sign $ before the field indicates that the field contains a hexadecimal number. You can click on the $ to change to decimal. The $ should disappear. Click on the resulting blank space before the field to switch back to hexadecimal. Most fields are displayed in hexadecimal by default; some in decimal. A few fields are only shown in decimal or hexadecimal and cannot be changed.

String options: a few string fields which can only take a small number of fixed values are implemented as a kind of click through list. Clicking on the field will change the value shown successively through the allowed options. Shift-clicking moves backwards through the list. These fields are usually indicated with a thicker left and right border.

Main Document Window

The Main Document Window shows the list of address operations. This occupies all except the top part of the window. The list is scrollable (up and down, but not left and right) and editable. Lines can be cut and pasted or dragged around as in any spreadsheet type application.

The contents of the address operation lines can also be edited. For a general description of the fields in each line of the address list, see the Address Fields help topic.

The top part of the window contains various control fields. These affect the way the list is executed.

The **Loop** button starts executing the list of operations. The number of loops is set in the field next to the button. If the loop count is more than one, the **Loop** button is changed to **Stop**. Clicking it will stop execution of the list. A loop count of zero is equivalent to an infinite loop. This can only be stopped using the **Loop/Stop** button.

The **Highlight** checkbox determines whether bus errors, Test errors or changes in the value of Read operations are shown on the screen. The loop will execute much faster if highlighting is disabled.

The **Log** checkbox selects whether an error log window is displayed. This is a text form of the Highlight option.

The **Alert** checkbox selects whether an alert window is presented when an error occurs.

The **Pause after loop** field sets the wait time in ticks (1/60 seconds) after each iteration through the list.

The **Pause after change** field sets the wait time in ticks after an error or changed value from a Read operation has occurred. This may be useful if you are waiting for some rare event to occur.

The **Comment** field allows you to describe the whole set of operations in the list. The comment is saved together with the rest of the document.

The **N.main loops** and **N.operations** fields are updated during execution. The N.main loops field shows the number of iterations through the list; the N.operations field shows the total number of VME operations. Since each line in the address list may itself be a loop, this will usually be many times the N.main loops value.

Address List Fields

Each line in the list of address operations consists of a number of fields as follows:

Insertion point: at the very left hand side of each line is an area which can display the insertion point cursor. Clicking in this area will set the insertion point to that line.

Num: this is line number of each line in the list. It is a static decimal field. Clicking on this field will select the line for subsequent cutting, pasting or dragging. Use shift and click to select contiguous groups of lines. Command (or) and click selects discontiguous lines. Option and click pops up the statistics from the previous loop.

On: shows which this line is enabled or not. Only lines which have been enabled will be executed.

Loops: the number of times this line will be executed each time round the main loop. This might be useful in conjunction with the Inc field to write to a large block of memory.

Address: the (starting) memory address at which the operation is to be performed. If the Highlight changes field is enabled and if a bus error occurs during execution the address field will be highlighted with a red border.

Inc: a clickable field with the values "°", "+" or "-". If the Loops field is greater than one, the address and/or data may be incremented or decremented (or neither) on every iteration of the miniloop. However the original Address and Data values are used each time through the main loop.

Op: specifies the operation to be performed. This has two components: the type and length. Full details are given in the Help topic Operations Field.

Data: for the Write and Test operations, this contains the data to be written to the given address. For Bit and Soak operations it specifies the set of bits to be tested. Any bit in the data word set to 1 will be tested, bits set to 0 will not be tested. For Read operations, the datafield shows the value read from the specified address. If the Highlight field is enabled and a Test, Bit or Soak operation reads back a different value from that which was written, the Data field will be highlighted with a red border. If the value read by a Read operation is different from that read on the previous iteration the Data field will be highlighted with a blue border.

Comment: this field allows you to add a short description of the operation performed by this line. Address Operations Field

This is a full description of the Op field in the address list. Further down are the details of the operations.

The operations are shown with their initial letters: R, W, T, B, S for Read, Write, Test, Bit and Soak operations. The lengths are shown as the number of bytes: 1 for Byte, 2 for Word and 4 for Longword.

The **operations** themselves are as follows:

Read: a byte, word or longword is read from the given address. If Highlight changes is enabled, the value read is displayed in the Data field and is highlighted with a blue border if it is different from that read the previous time.

Write: the byte, word or longword in the Data field is written to the given address.

Test: the byte, word or longword in the Data field is written to the given address and then read back. If the value read back is different and if Highlight changes is enabled, the Data field is highlighted with a red border.

Bit: this is a single bit test. Each bit which is set in the Data field is tested in turn by writing a value with that single bit set to the specified address and reading it back. If Highlight changes is enabled, and if the value read back differs from that which was written, the Data field is highlighted with a red border. The program does not indicate which bit(s) failed the test.

Soak: this is similar to the single bit test above, but here every possible pattern of all the bits between the lowest and highest bits in the Data field is written and read back from the specified address. If Highlight changes is enabled, and if the value read back differs from that which was written, the Data field is highlighted with a red border. The program does not indicate which pattern(s) failed the test.

Execution

There are two ways to execute the list of operations:

¥ click the **Loop** button or choose the **Loop** from the **Operations** menu. This executes all the enabled lines in the list as many times as requested in the loop count field.

¥ select one or more address lines and choose **Execute** from the **Operations** menu. This executes the selected lines once (irrespective of whether they are enabled or not).

In order to conform to Macintosh user interface guidelines the address operations are executed a few at a time at each pass through the main event loop.

The reason for this is to allow the Macintosh to respond to menu commands, mouse clicks etc without too much delay even when the number of loops requested is very large.

At present, about 4000 operations or 20 ticks (0.3 secs) whichever takes longer, is carried out before returning control to the operating system and/or the THINK Class Library software empire.

The result is that I/O operations will be carried out in a number of bursts, interspersed with user interface type processing. These bursts of activity may be broken off anywhere in the middle of a loop, but should always be resumed at the same place.

If several documents are open and executing at once, each document will get one burst of IO operations in turn.

Future versions of *Zaradann* may allow user configuration of the execution mode, length of bursts , etc.

Inevitably, the main loop has higher overhead than the loops in each address operation line. So to check a single function at the highest speed it is best to use a high loop count on that line.

Memory Scan Window

The Memory Scan Window allows you to find valid memory locations.

The program will scan through the specified range of addresses, from the from address to the to address. On each iteration it will read a byte, word or longword from the current address and will then increment the current address by the value of the in steps of field.

The program reports the contiguous range of addresses where the read operation produced no bus error. Note that the intermediate values, between successive steps, are not tested. You can use successively finer scans to home in on the exact range of valid addresses.

To use the utility, enter the desired values in the from , to , and step fields. Set the read operation length by clicking on the reading field which changes between byte, word and longword. Then click on the **Scan** button to start the scan.

The values of all the fields are automatically saved in the Preferences file when the window is closed. This means the default values of all fields are those used last time. Nubus Scan Window

The Nubus Scan Window scans all the Nubus slots in the macintosh and displays the cards which are currently installed.

Note: due to a feature of the MICRON card, the slot scan will not show the MICRON card if the last access to the card resulted in a bus error.

Camac Address Window

The Camac Address Window allows you to construct Camac addresses from the combination of branch, crate, station, subaddress and function. This utility can handle both Camac addressed via MacCC cards and also via the CES VME-Camac branch driver. Special Camac functions such as Crate Z, Crate C etc can also be generated.

The fields in the window are:

Nubus slot: the Nubus slot in the Macintosh containing the MICRON card.

Interface: the interface to Camac. Clicking on the field switches between MacCC and CamacBD (for the CES VME Camac branch driver).

VME crate: the VME crate containing the Camac branch driver. The MacCC is always treated as being in crate 8.

Camac branch: the Camac branch number 0—3. The MacCC can only address branch 0.

Camac crate: the Camac crate number 0—7.

Special: some special Camac functions can be selected by clicking on this field. The Camac slot, subaddress and function codes are set appropriately.

Camac slot: the slot number of the Camac module to be addressed. Usually in the range 1—23 except for special functions.

Subaddress: the internal subaddress 0—15.

Function: the Camac function to be executed. Functions 0—7 are Read operations, 16—23 are Write operations, 8—15 and 24—31 are control operations.

After setting appropriate values in the fields, click the **Set Address** button. The VME address of the operation will be written in the text box below the button.

The values of all the fields are automatically saved in the Preferences file when the window is closed. This means the default values of all fields are those used last time.

Memory Snoop Window

The Memory Snoop Window allows you to examine the contents of a contiguous range of memory locations.

When the window is opened it displays a default range of addresses which are one, two or four bytes apart as determined by the units field. The contents of those addresses are initially unknown.

Almost any action in the window will cause the addresses to be read and their contents displayed.

You can force a read by clicking the **Read** button or by changing the units. Also clicking or dragging the scroll bar will change the address range and show the contents of those addresses.

To read a different range of addresses, enter a new base address in the from field and click **Read** .

NB the continuously check box is not yet implemented. The intention is to display the values of the addresses in (more or less) real time.

WARNING: some bits of hardware will change their state even by read accesses only at some addresses. This is the reason the default address range is not read automatically when the window opens.

The values of all the fields are automatically saved in the Preferences file when the window is closed. This means the default values of all fields are those used last time. Preferences

The values of the fields in the Memory Scan, Camac Address and Memory Snoop windows are automatically saved to the *Zaradann* Preferences file.

Additionally, it is possible to save a particular file to be used as the default template for New files. This can be done with the Save As Prefs option in the File menu.

To recover the original defaults, remove the Zaradann Preferences file from the Preferences folder inside the System Folder.

Credits

Zaradann is a development of an original idea by Mikhail Savitski. *Zaradann* adds a few utilities and the possibility to extend and save the address lists as separate documents. Some of the utilities (Nubus and memory scans) were inspired by Bill Haynes VME tools in MPW. The Save as C/C++ feature is an idea from Eric Eisenhandler. Many thanks to various colleagues for testing it, finding bugs and suggesting improvements.

Zaradann is free (yes!), but enthusiastic appreciation, suggestions for improvements and/or reports of bugs may be sent on a (picture) postcard to:

Murrough Landon Physics Department Queen Mary & Westfield College Mile End Road London E1 4NS UK.

or by email to: m.p.j.landon@qmw.ac.uk

Version History

1.0.0 (07-May-1994) Initial version.

1.0.1 (10-May-1994) Some minor bugs fixed. Major bugs (eg Execute command not working) remain until....

1.1.0 (15-Jul-1994) Overhaul:

¥ Execute and Select All command now work as intended.

¥ Enable and Disable commands added.

¥ Iconic style for operations removed (the meaning of the tiny icons wasnt really clear enough).

¥ Added increment/decrement facility for data.

¥ Added facility for error log and alerts.

¥ Added Save As C/C++ command.

¥ Added Memory Snoop utility.

¥ Corrected Camac address of crates >= 4 via MacCC.

¥ Main window colour changed under extreme pressure from colleagues (what was wrong with pink anyway?!).

1.1.1 (27-Jul-1994) A few bug fixes and minor improvements:

¥ MemSnoopWindow: base address is now used and updated correctly.

¥ Shift+click now moves backwards in click through lists.

1.2.0 (7-Sep-1994) Update to use TCL 2.0. Some improvements & bug fixes:

¥ MemScanWindow: scan now works again (the previous version had optimised away the vital read operation!).

¥ MemSnoopWindow: all operations now read all settings from the window.

¥ Loop command: fixed problem with N.Loops=1 and large N.Loops on individual operations.

¥ Execute command: fixed highlight handling and excution of selected but disabled operations.

¥ The Option/Alt key now alters the behaviour of some menu commands (Open, Enable, Disable).

1.2.1 (9-Sep-1994) A few more bug fixes:

¥ Loop command: I think all problems are now really fixed for all operation types for all N.Loop values for both main window and individual operations.

¥ Clicking checkboxes now makes the document dirty .

¥ Fixed backwards wrap around in click through lists.

1.2.2 (6-Oct-1994) Yet more bug fixes:

¥ Loop command: OK so one little bug survived: now the data increment works for large N.Loop values.

¥ Optimisation problem fixed with reading back values in Test, Bit and Soak operations.