Using the GNX Debugger on a Sun® Workstation

National Semiconductor
Application Note 698
Michael Orr
July 1990

1.0 INTRODUCTION

The GNX debugger supports two different kinds of user interfaces, line-oriented and graphics oriented.

The debugger's graphic user interface has two implementations. One is available for graphic terminals in the X-Windows environment, and one is available for use on ASCII terminals.

Since a SunView implementation of the graphic user interface of the debugger is not presently available, users of the debugger on Sun machines were until now limited to using only the ASCII terminal version. This mode does not allow the user to take full advantage of the Sun's mouse and graphics facilities.

This note describes one way in which Sun users can add convenient facilities to the ASCII terminal user interface of the GNX debugger. These facilities include mouse pointing, pull down menus, command buttons, etc.

The note is divided into four sections. Section 1 is an introduction. Section 2 presents a general description of a tool which allows the addition of graphic facilities to nongraphic applications.

Section 4 contains a specification file which when input to the tool described in section 2, provides graphic facilities to GNX/Dbug users on Sun machines. See section 3 for usage and installation notes.

2.0 TOOLTOOL DESCRIPTION

There are two requirements for adding graphic facilities to non-graphic applications on a Sun machine.

1. A tool which opens a simple window which behaves like a terminal and adds graphic facilities (referred to here as *gadgets*) around this window. Any application can run in the window, just like on a terminal.
2. A description file which tells this tool which gadgets to add to which application.

A tool which answers the first requirement, and is already part of the public domain is called Tooltool, and was written by Chuck Musciano from the Advanced Technology Department of Harris Corporation.

Tooltool is a software package that allows the user to take a previously ''unwindowed'' application, and run it in a window to which various ''gadgets'' are attached. These ''gadgets'' can generally be most types of SunView panel items (e.g. buttons, menus, etc.) and other SunView windowing related gadgets (e.g. Sliders). This window also supports mouse interactions such as selection.

A full list of available gadgets can be found in the Tooltool manual.

Tooltool users provide a ''specification'' file detailing the application to be run and the gadgets to be attached to the window in which the application will be run, and the actions associated with each gadget.

Basically each gadget has an action associated with it. This action may be related to the window and gadgets, (e.g. display or hide a button), or sending input to the application. This second type of action sends a sequence of characters to the application. The application understands these characters as if the user has typed them on the keyboard.

2.1 Getting Tooltool

If you currently do not have Tooltool, you can get if from one of the archive sites of Sun sources, or by *Anonymous FTP* from the machine called ''trantor.harris-atd.com'' whose IP address is ''26.13.0.98''. (*Anonymous FTP* is a mechanism by which your computer connects over a telephone line to another computer, where you can login and have files transferred to your machine. Contact your system administrator for details).

Getting the sources of Tooltool from an archive site is an easy process, but since each site has its own procedures, the description of how to get files vary from one archive site to another, and are not detailed here. To find out the archive site closest to you and the way to get files from it to your machine, read the articles in the newsgroup ''comp.archives''.

2.2 Getting More Information

Tooltool's author can be contacted at:

Chuck Musciano
Advanced Technology Department
Harris Corporation
P.O. Box 37, MS 3A/1912
Melbourne, FL 32902
Telephone: (407) 727-6131

Or using electronic mail at

chuck@trantor.harris-atd.com

If you have any questions about this application note you can contact

Michael Orr
National Semiconductor (Israel)
P.O. Box 3007, 46104
Israel
Telephone: +972-52-522255

Or, using electronic mail at

orr@nsc.nsc.com

Sun® is a registered trademark of Sun Microsystems.

 TL/EE10867 RRD-B30M75/Printed in U. S. A.

3.0 HOW TO INSTALL AND USE THE SPECIFICATION FILE

This section explains how to install the specification file and use it.

3.1 Generating and Naming the Specification File

Copy all the lines below the line marked ''CUT HERE'' to a file called **dbug.tt** in the directory where you intend to use the debugger.

3.2 Checking the Availability and Version of ToolTool

Make sure that ''ToolTool'' is installed in a directory which is in your path, and that it is version 2.0 or later. This is easily done as follows: type

tooltool—f dug.tt

If the response is 'tooltool : Command not found' or something similar, it means that tooltool is not installed in any directory mentioned in your $path. (To see the list of directories in your $path type 'echo $path'). To fix this ask your system adminstrator to install tooltool in the right place. (usually /usr/local/bin, or in the same directory with the GNX tools).

If the response is 'dbug.tt : line xxx: syntax error at or near ''dialog'', or something similar, then you have a version of tooltool earlier than 2.0. In that case you should do 2 things: first, ask your system administrator to get and install a more up-to-date version of tooltool; Second, type the following:

/lib/cpp −DOLD_VERSION −C dbug.tt > tmp.tt; mv tmp.tt dbug.tt

This will generate a specification file that your currently available tooltool version can read, at the cost of losing Pop-up windows for confirmation of the ''quit'' command and for debugger alias definition.

3.3 Defining an Alias for Easier Use

Define an alias as follows:

alias DBUG ''tooltool −f dbug.tt''

(You may want to add this alias definition to your .login file.)

Now everything is ready, and all you have to do is invoke the debugger exactly as you usually do, with the only difference being that instead of typing 'dbug' you now type 'DBUG'. (e.g. instead of typing 'dbug -I my__include__dir my__buggy__program core' you now type 'DBUG -I my__include__dir my__buggy__program core')

3.4 Customizing the Specification File to Your Needs

It is easy to customize the specification file for your particular needs, even without familiarity with Tooltool.

The specification file given in section 4 assumes you are running your programs on a board with a NS32CG16 CPU, no MMU, no FPU and using the MON16 monitor. If this is untrue, you simply change the string following the word ''application'' in the specification file. You can use this technique to add buttons and pull-down menus to the line interface mode of the GNX debugger, by simply changing 'dbug' to 'dbg' in the string. Another good use of this feature is to change the string after the 'application' keyword to 'rlogin <remote-machine-name>' where 'remote-machine-name' is the name of another machine such as a sys32/30 or a VAX/VMS. The effect is to get a window in which you are logged in to the remote machine, and in which you can run any commands you want. This window, however, has the associated gadgets for the case where you want to run the GNX debugger on the remote machine.

Other modifications you may want to try are either changing the strings sent to the application by the various gadgets, or adding gadgets using the existing gadgets as templates to add more commands.

4.0 THE TOOLTOOL SPECIFICATION FILE

This section contains a specification file for ''tooltool'' which will run the GNX debugger in a window, and add buttons and pull down menus for the most useful commands.

Almost all gadgets are of the type that send commands to the debugger as if they were typed on the keyboard, and thus serve mainly as convenience shortcuts. I tried to make the specification file highly commented and readable. (Note that familiarity with the GNX debugger commands is assumed.)

```
/* ------ CUT HERE ------ CUT HERE ------ CUT HERE ------ CUT HERE ------CUT HERE */
/*-----------------------------------------------------
 * ToolTool specification file for GNX/dbug
 * Use by 'tooltool -f <this-file-name><anything-to-pass-to-dbug>'
 *   e.g: "tooltool -f dbug.tt buggy core"
 *
 * Note:
 *    This specification file is intended for Tooltool version 2.
 *    if you only have an earlier version, you should do the following:
 *      1. Run this file through the C preprocessor with the
 *         command line-arguments'-DOLD_VERSION -C'
 *      2. Use the resulting file as the Tooltool specification.
 * Author: Michael Orr
 *    orr%taux01@nsc.nsc.com
/*-----------------------------------------------------*/
#define STACK_START_ADDRESS 0x1ffff0  /* modify as needed */



/*
 * The string after 'application' should be the normal command line
 * you usually use to invoke the debugger. This example shows a command
 * line intended for debugging a program on a remote CG16 board
 * with no MMU and no FPU, and using MON16 as the board's monitor.
 */
 application "dbug -mon 16 -cpu CG16 -mmu nommu -fpu nofpu" /* modify as needed */
 size 40 by 80 characters  /*modify to the size you want */
 label "DBUG"              /*Shown in top stripe - change to your favorite */
 gadgets
   top            /* Use 'bottom' to have gadgets under the window */
   proportional
   menu "Start-up"
     "Connect"  menu    /* This menu entry has a pull-right sub-menu */
       /* For remote operations. For native-mode debugging
        * (e.g. on a sys/30) either remove this entry, or simply
        * don't use it in your debugging session.
        */
               "ttya" send "connect link ttya\n";
               "ttyb" send "connect link ttyb\n";
     end_menu
     "Load"   send "load with sp STACK_START_ADDRESS\n";
     "Re-Load" send "load with nocode\n";
     "Run"    send "run\n";
     "Re-Run" send "rerun\n";
     "Arrange"   menu
            "windows" {   /* modify to arrange windows to your liking */
               send "wdelete program\n";
               send "wmove code vr r20\n";
               send "wmove dialog vr r20\n";
               send "wmove code dl9\n";
               send "wmove dialog u20\n";
            }
            "Wreset" send "wreset\n";
      end_menu
   end_menu
   menu "f(Selection)"      /* Use strings selected with the mouse */
     "print"          send format("print %s\n",selection(l));
     "print*"         send format("print* %s\n",selection(l));
     "print&"         send format("print& %s\n",selection(l));
     "Stop in"        send format("stop in %s\n",selection(l));
     "Stop at"        send format("stop at %s\n",selection(l));
     "whatis"         send format("whatis %s\n",selection(l));
     "whereis"        send format("whatis %s\n",selection(l));
     "which"          send format("which %s\n",selection(l));
     "find-forward"   send format("/%s\n",selection(l));
     "find-backward" send format("?%s\n",selection(l));
   end_menu
```

```
menu "run"
  "run"     send "run\n";
  "next"    send "next\n";
  "step"    send "step\n";
  "cont"    send "cont\n";
  "return"  send "return";
  "rerun"   send "rerun\n";
end_menu
menu "CMDi"      /* Assembly level commands */
  "Nexti"   send "nexti\n";
  "Stepi"   send "stepi\n";
end_menu
menu "Env"
  "where"  send "where\n";
  "status" send "status\n";
  "list"   send "list\n";
  "func"   send "func\n";
  "file"   send "file\n";
  "up"     send "up\n";
  "down"   send "down\n";
end_menu
/*
 * Usually you can send input to the debugger by typing into the "dialog"
 * window and to the program by typing in the "program" window, or by
 * typing into the "dialog" window, but prefacing your input with a '@'.
 * I define any text typed into the gadget window as intended for
 * the program, and it is sent to the program (i.e., as if you typed it
 * into the "dialog" window) by prefacing it with a '@'.
 * This lets you remove the annoying narrow vertical program window and
 * make better use of the window area.
 * (note - any output of the program will appear in the dialog box when
 * the "program" window is hidden)
 */
text program
  label "Input to the Program:"
  display 55
  action {
        send format("@%s\n",program);
        program = ""; /* Erase what we just typed */
        }
end_text
/*
 * Definition of buttons
 *
 * Note - some buttons have different actions associated with them
 *      if you click the mouse on them or you click the mouse while
 *      holding down SHIFT or CONTROL. These buttons have an associated
 *      menu which can be seen by pressing and holding down the RIGHT
 *      mouse button on them.
 */

button
  normal "next" send "next\n";  /* just click mouse over the button */
  shift "step" send "step\n";   /* hold down SHIFT and click mouse */
  control "cont" send "cont\n"; /* hold down CTL and click mouse */
end_button
button
  normal "step" send "step\n";
end_button
button
  normal "cont" send "cont\n";
end_button
button
  normal  "print"    send format("print %s\n",selection(l));
  shift   "Print*"    send format("print* %s\n",selection(l));
  control "Print&"    send format("print&%s\n",selection(l));
end_button
```

```
   button
     normal "print*" send format("print* %s\n",selection(l));
   end_button
   button
     normal "list"  send "list\n";
   end_button
#ifdef OLD_VERSION
   button
     normal "Quit" send "quit\n";
   end_button
#else
   button
     normal "quit" popup quit_popup;
   end_button
   button alias_button
     normal "alias" display define_alias;
   end_button
#endif
end_gadgets
/*Dialog boxes -- only supported in Tooltool version 2 and above */

#ifndef OLD_VERSION
dialog define_alias  /* define a new alias or list existing ones */
label "Define aliases"
open  remove alias_button;
close display alias_button;
gadgets
   text alias
     display 40
     label "Alias:"
   end_text
   button
       normal "Ok" {
              send format("alias %s\n",alias);
              alias = "";
              }
   end_button
   button
     normal "List" send "alias\n";
   end_button
   button
     normal "Done" remove define_alias;
   end_button
end_gadgets
end_dialog

dialog quit_popup      /* Exit confirmation Popup */
size 1 by 20 characters
label "Really quit?"
gadgets
   button
     normal "Yes" { remove quit_popup; send "quit\n"; }
   end_button
   button
     normal "Cancel" { remove quit_popup; }
   end_button
end_gadgets
end_dialog
#endif
```

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor Japan Ltd.
Sumitomo Chemical
Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductores Do Brazil Ltda.
Rue Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

National Semiconductor (Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998