



# **Etnus TotalView on the Cray X1**

**Bob Moench  
May 2, 2003**



# Presentation Overview



- Etnus TotalView Background
- Debugging issues unique to Cray X1
- Current capabilities
- Next release 4Q03
- Still to Come
- Slide show demonstration
- Summary



# Etnus TotalView Background



- Common ancestor with Cray TotalView (CTV)
- Cray purchased software rights to Etnus TotalView (ETV) from Etnus LLC
- Periodic updates included
- Command Line Interface (CLI) version, **totalviewcli**, released December 2002
- Graphical User Interface (GUI) version, **totalview**, in final exposure
  - Based on ETV version 6.1



# Debugging Issues Unique to Cray X1



- Vector registers
- Multi-stream processors (MSPs)
- Distributed Memory (DM) machine
- Remote Translation Table (RTT) shared memory core file sets
- **aprun** launcher



# Current Capabilities



- Debugs command mode executables
- Debugs MSP mode, non-DM executables
- Performs -G0/-g live and core file debugging
- Supports C-Language, C++, Fortran, Assembly Language
- Supports X1 registers, NV1 instruction set, Unicos/mp
- Supports -Ostream0 with streamed libraries



## Next Release 4Q03



- Compensate for relocation of DM processes
- Support RTT shared memory core file debugging
- Support `aprun` launching of DM jobs in concert with the debugger
- Access to information from SSPs 1-3
- Upgrade to latest Etnus TotalView source
- Improve debugger group's regression test suite



# Still to Come



- Direct support for Distributed Memory models
  - MPI, CoArrays, UPC
- Direct support for Shared Memory models
  - PThreads, OpenMP
- Support **-G1/-gp**
- Support for Vector registers
- Support for Watchpoints
- Provide DebugView like capability



# Demo Overview



- A “live” demonstration via slides
  - GUI demo
  - CLI demo



# Process/Thread Window



The screenshot shows the Etnus TotalView Process/Thread window. The menu bar includes File, Edit, View, Group, Process, Thread, Action Point, Tools, Window, and Help. The toolbar below the menu bar includes Group (Control), Go, Halt, Next, Step, Out, Run To, NextI, StepI, P-, P+, T-, and T+. A sub-menu bar for 'controller1' is visible above the toolbar.

**Source code:** The left pane displays the source code for 'controller1.f'. Line 324 highlights a call to 'new(volt\_supply, gpib, 12)'. A red arrow from the 'Current location' box points to this line.

**Arguments, locals, and registers:** The right pane shows the 'Function main in controller1.f' with local variables: MEETER, RESULT, SUPPLY1, SUPPLY2, and VOLT\_SUPPLY. A red arrow from the 'Arguments, locals, and registers' box points to the 'RESULT' variable.

**Current location:** A red box labeled 'Current location' has a red arrow pointing to the line 'call new(volt\_supply, gpib, 12)' in the source code.

**Action points:** The bottom right pane shows 'Action Points' with a list containing '1 controller1.f#337 main'. A red arrow from the 'Action points' box points to this entry.



# Process Status

The screenshot shows the Etnus TotalView interface. At the top, a menu bar includes File, Edit, View, Group, Process, Thread, Action Point, Tools, Window, and Help. Below the menu is a toolbar with buttons for Group (Control), Go, Halt, Next, Step, Out, Run To, Next!, Step!, P-, P+, T-, and Stop. A status bar at the bottom displays "controller1".

The main window title is "controller1". Inside, a large text area displays:

```
Process 1 (315908): controller1 (Stopped)
Thread 1.1 (315908) (Stopped) <Trace Trap>
```

Below this, a code editor window shows C++ source code:

```
322 1----- program main
323
324 ! this module uses Controller related classes, as described in
325 ! Scientific and Engineering C++, by John J. Barton and Lee R. Mackna
326 ! [Addison-Wesley, Reading, MA, 1994], pp. 226-249.
327     use Voltmeter_class
328     use VoltageSupply_class
329     use IVTester_class
330     type (GPIBController_Stub) :: gpib
331     type (Acme130), target :: volt_supply, supply1
332     type (VoltyMetrics), target :: meter
333     type (Volton59), target :: supply2
334     type (IVTester) :: iv
335     real :: result, v_step = 1.0
336 ! first example, p. 228-
337     call new(volt_supply,gpib,12)    ! Supply at GPIB address 12.
338     call set(volt_supply,3.6)
339 ! second example, p. 232
340     call new(meter,gpib,14)
341     call new(supply1,gpib,12)
342     result = orig_checkCalibration(supply1,meter,1.0)
343     Print * 'Acme130 relative error at 1 volt is: ', result
344 ! third example, p. 235
345     call new(meter,gpib,13)
```

A yellow arrow points to line 342. The bottom of the code editor has tabs for "Thread (1)" and "Action Points". The "Thread (1)" tab shows "1.1 T in main". The "Action Points" tab shows "stop 1 controller1.f#337 main".



The 45th CUG Conference

CRAY USER GROUP FLIGHT TO INSIGHT



May 19-22, 2003 • Columbus, Ohio



# Stack Trace Pane

Stack trace

The screenshot shows the Etnus TotalView interface with the "Stack Trace" pane highlighted by a blue border. The pane displays a stack trace for Thread 1.1 (315908) which is stopped at line 340. The stack trace shows the call stack from main() down to \_\_start(). The function main() has two frame pointers: FP=40000200f180 and FP=40000200f200. The code editor window below shows lines 338 through 345, with line 340 highlighted. The "Action Points" window at the bottom shows a stop point at controller1.f#337 main.

```
f90 main,  
      __start,
```

FP=40000200f180  
FP=40000200f200

```
338     call new(volt_supply,qpib,14)    ! supply ac main address 14.  
339     call set(volt_supply,3.6)  
340     ! second example, p. 232  
341     call new(meter,qpib,14)  
342     call new(supply1,qpib,12)  
343     result = orig_checkCalibration(supply1,meter,1.0)  
344     Print *, 'Acme130 relative error at 1 volt is: ', result  
345     ! third example, p. 235  
       call new(meter,qpib,13)
```

Thread (1)	Action Points
1.1 T in main	stop 1 controller1.f#337 main





# Source Pane



Source code

The screenshot shows the Etnus TotalView interface with the Source Pane active. The code is a series of Fortran statements related to instrument calibration and testing. A red rectangular box highlights the right portion of the code area, which includes lines 334 through 344. A yellow arrow points from the word 'result' in line 341 towards the highlighted code. A red button labeled 'STOP' is visible in the margin of the source code window. Below the source code window, there are two smaller windows: 'Thread (1)' showing '1.1 T in main' and 'Action Points' showing 'stop 1 controller1.f#337 main'.

```
332      type (voltmeter), target ::  
333      type (VoltOn59), target ::  
334      type (IVTester) :: iv  
335      real :: result, v_step = 1  
336 ! first example, p. 228  
337      call new(volt_supply, gpib,  
338      call set(volt_supply, 3.6)  
339 ! second example, p. 232  
340      call new(meter, gpib, 14)  
341      call new(supply1, gpib, 12)  
342      result = orig_checkCalibra  
343      Print *, 'Acme130 relative'  
344 ! third example, p. 235  
345      Print *, 'Acme130 relative error at 1 Volt is: ', result  
346      ! third example, p. 235  
347      call new(meter2, gpib, 13)
```



# Action Points Pane

The screenshot shows the Etnus TotalView interface. At the top, there's a menu bar with File, Edit, View, Group, Process, Thread, Action Point, Tools, Window, Help. Below the menu is a toolbar with buttons for Group (Control), Go, Halt, Next, Step, Cut, Run To, Next!, Step!, F-, F+, T-. A status bar at the bottom indicates Process 1 (315900) controller1 (Stopped) and Thread 1.1 (315908) (Stopped) <Trace Trap>. The main window has tabs for Stack Trace and Stack Frame. In the Stack Trace tab, it shows main() and \_\_start(). In the Stack Frame tab, it shows Function "main": No arguments, Local variables: QPIB: (type (QPIBCONTROLLER)) I: 0 (0x00000000). A large blue box highlights the "Action Points" section. Inside this box, a red button labeled "STOP" is shown above the text "1 controller1.f#337 main-". Below this, a code editor window shows lines 339 through 345 of a C program. Line 341 contains a call to "new(meter.qpib,14)". Line 342 contains a call to "new(supply1.qpib,12)". Line 343 contains a call to "orig\_checkCalibration(supply1,meter,1,0)". Line 344 contains a print statement. Line 345 contains a call to "new(meter2.qpib,13)". An orange arrow points from the text "Action points" in a callout box to the "Action Points" tab in the bottom-left corner of the main window.

Action Points

STOP 1 controller1.f#337 main-

339 ! second example, p. 232  
340 call new(meter.qpib,14)  
341 call new(supply1.qpib,12)  
342 result = orig\_checkCalibration(supply1,meter,1,0)  
343 Print \*,"Acme130 relative error at 1 volt is: ", result  
344 ! third example, p. 235  
345 call new(meter2.qpib,13)

Action Points

Thread (1) Action Points

1.1 T in main STOP 1 controller1.f#337 main-



# Stack Frame Pane

Stack Trace      Stack Frame

Process 1 (315900): controller1 (Stopped)  
Thread 1.1 (315908) (Stopped) <Trace Trap>

Function "main":  
No arguments.  
Local variables:  
GPIB: (type (GPIBCONTROLLER))  
I: 0 (0x00000000)  
IV: (type (IVTESTER))  
METER: (type (VOLTYMETRICS))  
RESULT: 0

Arguments,  
locals, and  
registers





# Stepped into New Function

The screenshot shows the Etnus TotalView interface. A callout box labeled "New function" points to the function name "ORIG\_CHECKCALIBRATION" in the stack trace. Another callout box labeled "Out of function button" points to the "Out" button in the toolbar. A third callout box labeled "Stack Trace" points to the stack trace window.

New function

Out of function button

Stack Trace

```
f90 ORIG_CHECKCALIBRATION, FP=40000200  
f90 main,  
__start,
```

Function "ORIG\_CHECKCALIBRATION":  
SUPPLY: (type(BUS130))  
METER: (type(VOLTMETRIC))  
TST\_VOLTAGE: 1  
Local variables:  
ORIG\_CHECKCALIBRATION: 2.60752e-09  
Registers for the frame:  
a0: 0x00000000 (0)  
a1: 0x40000200e98 (7036877779317)  
-> 0x00000000-0 c702d97799999999





# X1 Registers

The screenshot shows the Etnus TotalView interface. A window titled "controller1" displays a stack trace and a stack frame for a function named "ORIO\_CHECKCALIBRATION". The stack trace shows "MAIN" and "\_start". The stack frame details include SUPPLY, METER, TST\_VOLTAGE, and Local variables. The Registers for the frame section lists three registers: a0, a1, and a2. The values for a0, a1, and a2 are shown as 0x00000000 (0), 0x40000200ee98 (7036877779317), and 0x40000000~~~0 /703c077770c10 respectively. A callout box labeled "New function" points to the stack trace area, and another callout box labeled "Out of function button" points to the "Out" button in the menu bar.

New function

Out of function button

Registers for the frame:

a0 : 0x00000000 (0)  
a1 : 0x40000200ee98 (7036877779317)  
a2 : 0x40000000~~~0 /703c077770c10

Function ORIO\_CHECKCALIBRATION in controller1.f

real :: tst\_voltage  
! Relative error at specified test voltage.

Registers for the frame:

a0 : 0x00000000 (0)  
a1 : 0x40000200ee98 (7036877779317)  
a2 : 0x40000000~~~0 /703c077770c10

ORIO\_CHECKCALIBRATION in controller1.f



# Stepping out to Assembly View

The screenshot shows the Etnus TotalView interface. On the left, a stack of memory frames is displayed, with frame 343 highlighted by a yellow arrow. The assembly code in the center pane corresponds to this frame. A blue box highlights the assembly code for frame 343, which includes calls to new() and print statements. Below the assembly code, a memory dump window shows the state of memory starting at address 0x0100a7f4. An orange arrow points from the text "Still in call?" to the memory dump, indicating that the program is still executing within the call instruction at line 343. Another orange arrow points from the text "Just after jsr, but not yet at line 343" to the assembly code, indicating the current execution point.

```
333 ! second example, p. 232
340
341
342 →
343     call new(meter,gpib,14)
344     call new(supply1,gpib,12)
345     result = orig_checkCalib
346     Print *, 'Acme130 relative'
347
348 ! third example, p. 235
349     call new(supply2,gpib,12)
350     result = checkCalibration
351     Print *, 'Acme130 relative'
352     result = checkCalibration
353     Print *, 'Volton59 relative'
354
355 ! fourth example, p. 249
356     call new(iv,VoltageSupply)
357     do i = 0, 9
358       result = current(iv,i)
359     enddo
```

result = orig\_checkCalib  
Print \*, 'Acme130 relative'  
call new(supply2,gpib,12)  
result = checkCalibration  
Print \*, 'Acme130 relative'  
result = checkCalibration  
Print \*, 'Volton59 relative'  
call new(iv,VoltageSupply)  
do i = 0, 9  
result = current(iv,i)

0x0100a7f4: [a62-143] s1.v  
0x0100a7f8: s1.v [a62-143]  
0x0100a7fc: [a62-135] s1.v  
0x0100a800: a1.0.a  
0x0100a804: a2 575.c  
0x0100a808: a1.b 0  
0x0100a80c: a2.d 31680  
0x0100a810: a30.1 a1.a2  
0x0100a814: a1 64384.abcd  
0x0100a818: a2.1 a62+a1  
0x0100a81c: a1.1 |a30+0|  
0x0100a820: [a62+0] s1.l  
0x0100a824: a1.1 |a30+1|

Still in call?

Just after jsr,  
but not yet at  
line 343



# Assembly



```
0x0100a7ec : a59:b 63212
0x0100a7f0 : j, a60 a61, sr
0x0100a7f4 : [a62-143] s1,w
0x0100a7f8 : s1,w [a62-143]
0x0100a7fc : [a62-135] s1,w
0x0100a800 : a1 0:a
0x0100a804 : a2 575:c
```

Still in call?

```
346 result = checkCalibration();
347 Print *, "Home130 relative
348 result = checkCalibration();
349 Print *, "Volton59 relative
350 ! fourth example, p. 249
351 call new(iv_.VoltageSupply);
352 do i = 0..9
353 result = current(iv_.VoltageSupply);
354 enddo
```

Thread (1)  
1.1 T in main

Action Points  
STOP 1 controller1.f#337 main

Just after jsr,  
but not yet at  
line 343



# Halting the Run



Mixed

lang

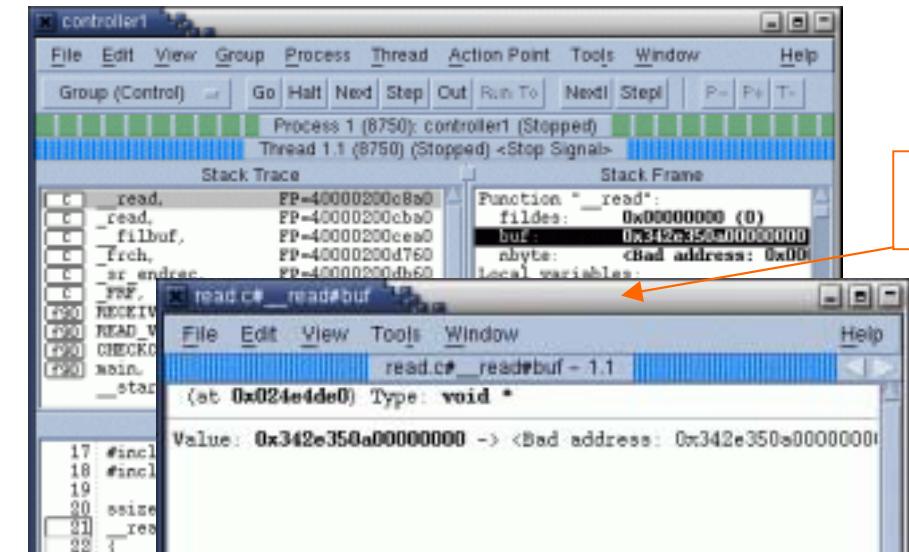
Div

	Stack Trace
C	_read,
C	_read,
C	_filbuf,
C	_frch,
C	_sr_endrec,
C	_FRF,
f90	RECEIVE,
f90	READ_VM,
f90	CHECKCALIBRATION,
f90	main,
	_start,

Diving on  
dataDiving on  
action points



# Diving on Data



Result of diving on *buf*

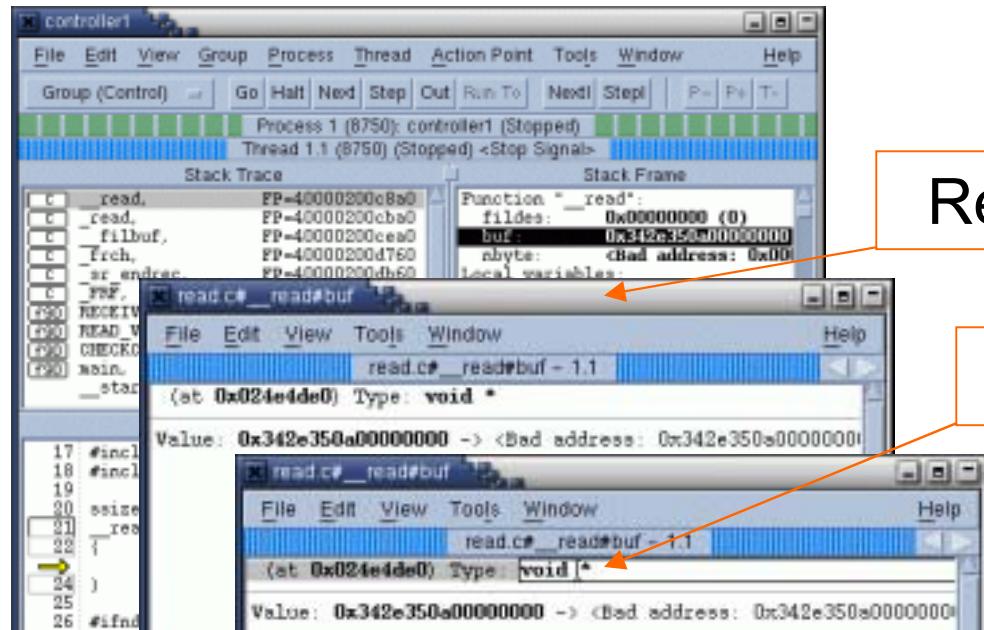
read.c#\_read#buf - 1.1

(at 0x024e4de0) Type: void \*

Value: 0x342e350a00000000 -> <Bad address:



# Diving on Data



Result of diving on *buf*

Editing type of *buf*

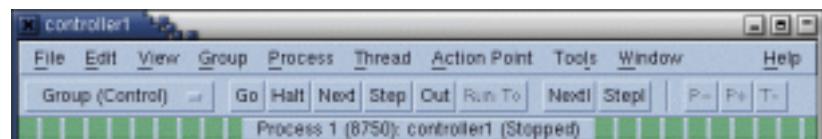
read.c#\_read#buf - 1.1

(at 0x024e4de0) Type : void[\*]

Value : 0x342e350a00000000 -> <Bad address:



# Diving on Data



read.c#\_read#buf - 1.1

(at 0x024e4de0) Type: <string>

Value: "4.5\n"

```
29 #include <sys/types.h>
30 #include <sys/conf.h>
31 #include <sys/conf.h>
32
33 ssize_t
34 read(int filedes,
35       void *buf,
36       size_t nbyte)
37 {
38     LIBLIBC();
39 }
```

read.c#\_read#buf - 1.1

(at 0x024e4de0) Type: <string>

Value: "4.5\n"

Thread (1)  
1.1 T in \_read



# More Data

```
File Edit View Tools Window Help
fch.c#_fch@stat - 1.1
(at 0x40000200cf40) Type: struct fsw

Field          Type            Value
sw_flag        unsigned int:1  0x00 (0)
sw_error       unsigned int:31 0x00000000 (0)
<padding>
sw_count       <char>[4]    (Array)
sw_stat        long           0x0000000000000000 (0)
<padding>
sw_user        <char>[2]    (Array)
sw_iptc        int:32         0x00000001 (1)
sw_sptr        void *         0x02484020 -> 0x0000000000000000 (0)
sw_rptr        int             0x00000000
sw_cswl        int             0xfffffaec (-1300)
<padding>
aiochp        <char>[4]    (Array)
aio_fildes    struct aioch   (Struct)
int            0x00000000 (0)
<padding>
aio_buf        <char>[4]    (Array)
void *          0x013a9668 (& _p_SWPT_cwp0_and_swap
aio_nbytes    unsigned long int 0x000040000200d260 (70368777786952)
aio_offset     long           0xffffffffeba800000000 (-22368189677568)
aio_reqorio   <char>[4]    (Array)
<padding>
aio_sigevent   struct sigevent (Struct)
sigev_notify   int             0x00000000 (0)
cpadding>
sigev_notifyinfo <char>[4]    (Array)
union notifyinfo (Union)
sigev_value    union signal   (Union)
sigev_notify_function void (*) (void) 0x013aee5c : pthread_mutex_unlock+0x
sigev_notify_attributes struct _P_type_20 + 0x013a92a8 (& _p_SWPT_cwp_and_swap)
sigev_reserved  unsigned long int[11]
[0]              0x000040000200d2a0 (70368777786016)
[1]              0x0000000000000000 (0)
[2]              0x000000000000002b (43)
[3]              0x000000000024e6e00 (30694400)
[4]              0x0000000001074324 (17253364)
[5]              0x000000000000002b (43)
[6]              0x0000000000000000 (0)
[7]              0x0000000000000000 (0)
[8]              0x0000000000112a7d0 (17999024)
[9]              0x0fffffa900000000 (1844673809511507)
[10]             0x0000000000000001 (1)
sigev_pad      unsigned long int[6]
[0]              0x00000000010f44d60 (17780064)
[1]              0x0000000001055af4 (17120180)
[2]              0x0000000000000000 (0)
[3]              0x0000000000000000 (0)
```



The 45th CUG Conference

CRAY USER GROUP

FLIGHT TO INSIGHT



May 19-22, 2003 • Columbus, Ohio



# CLI



- Stand-alone or with GUI
- Command line ASCII input
- Tcl based interface
  - ‘d’ prefix on TotalView commands
  - Watch out for special characters!!
    - \$, [ , ], { , }, #, ;, “, space
- Programmable



# CLI Startup

```
sn702> totalviewcli controller1
```



```
d1.<> dbreak 337
```

```
1
```

```
d1.<> dgo
```

Created process 1 (71894), named "controller1"

Thread 1.1 has appeared

Thread 1.1 hit breakpoint 1 at line 337 in  
"main"

```
d1.<> dcont
```



# Application Input



(Acme130 now at address 12 )

(GPIB instrument # 12 sends value 3.5999999 )

(VoltyMetrics now at address 14 )

(Acme130 now at address 12 )

(GPIB instrument # 12 sends value 1. )

(Please enter number for GPIB instrument # 14 )

## 4.5

Acme130 relative error at 1 volt is: 3.5999999

(GPIB instrument # 13 sends value 1. )

(Please enter number for GPIB instrument # 14 )

Thread 1.1 received a signal (Interrupt)



# dwhere



d1.<> dwhere

```
>0 __read      PC=0x0110e810, FP=0x40000200c7e0 [read.c#23]
 1 _read       PC=0x0110ee0c,  FP=0x40000200cae0 [read.c#36]
 2 __filbuf    PC=0x0110c614,  FP=0x40000200cde0 [_filbuf.c#65]
 3 _frch       PC=0x010e51b4,  FP=0x40000200d6a0 [frch.c#224]
 4 _sr_endrec PC=0x0108ece8,  FP=0x40000200daa0 [rf.c#876]
 5 _FRF        PC=0x01041d00,  FP=0x40000200e040 [rf90.c#333]
 6 RECEIVE     PC=0x01003c3c,  FP=0x40000200e460 [cont.f#50]
 7 READ_VM     PC=0x01005af0,  FP=0x40000200e720 [cont.f#142]
 8 CHECKCAL   PC=0x01009928,  FP=0x40000200e9e0 [cont.f#363]
 9 main        PC=0x0100abac,  FP=0x40000200f0c0 [cont.f#348]
10 __start    PC=0x010019d4,  FP=0x40000200f140 [cont]
```



# dlist

d1.<> dlist

```
16  #include <sys.s>
17  #include "sys/sv2/syscall.h"
18  #include <unistd.h>
19
20  ssize_t
21  __read(int fildes, void *buf, size_t nbyte)
22  {
23 >    USUAL_SYSCALL(ssize_t, SYS_read, ARG_LIST3(fildes,
24                                buf, nbyte));
25  }
26
27  #ifndef LIBRESTART_LIBC
28  /* Wrapper for pthread callback */
29  #pragma weak read = _read
```





# dprint



```
d1.<> dprint *buf
```

```
*buf = <Bad address: 0x342e350a00000000>
```

```
d1.<> dwhat buf
```

In thread 1.1:

```
Name: buf; Type: void *; Size: 8 bytes; Addr: 0x024e4de0
```

```
Scope: ##controller1#read.c#__read(Scope class: Any)
```

```
Address class: reference_param(Reference parameter)
```

```
d1.<> dprint *(<string*>*)&buf
```

```
(*(<string*>*)&buf = "4.5\n"
```



# TCL “for”



```
d1.<> for {set i 57} {$i < 64} {incr i} {dprint \$a$i}  
$a57 = 0x00002fffffffffe0 (52776558133216)  
$a58 = 0x0000300000000000 (52776558133248)  
$a59 = 0xfffffb8c00000000 (-4896262717440)  
$a60 = 0x00000000110ee0c (17886732)  
$a61 = 0x00000000110e7c8 (17885128)  
$a62 = 0x000040000200c7e0 (70368777783264)  
$a63 = 0x000040000200c500 (70368777782528)
```



# Summary



- Etnus TotalView is a strong base for our debugging platform.
- There is more to be done.
- Management has increased our resources.