

# Using DDT

## Debugging programs with DDT

**Peter Towers**

**HPC Systems Section**

**Peter.Towers@ecmwf.int**



© ECMWF January 28, 2016

# Allinea DDT

- **DDT is a very popular interactive debugger**
  - Developed by Allinea Software in the UK
  - Modern graphical user interface
  - Highly scalable to large numbers of tasks/threads
- **Cray provide a 2048 task licence**
  - No limit to threads per task
  - Support from Cray backed up by Allinea
- **Version 4.2.1 (default) and 5.0.1 installed**
  - Version 5 also called Allinea Forge
  - Version 6 will be available from Cray later this year
- **Version 4.2.2 also available on the latest desktops**
  - Uses the same licence

# DDT Features

- **Examine sources**
- **Set breakpoints**
  - **Pause at a line of source**
  - **Conditional breakpoints supported**
- **Examine variables, Fortran modules, stack**
- **Set tracepoints**
  - **Output values of variables at a line without pausing**
- **Set watchpoints**
  - **Pause when a variable changes value**

# DDT Features

- **Step through execution line by line**
  - Step to the next line or into a function
  - Step over a line
  - Step out of a function
- **Catch signals**
  - Segmentation violations
  - Floating point exceptions
- **Track and debug memory usage**
  - Record all memory allocations
  - Find memory overwrites
- **Examine message queues**
- .....

# Compilation

- **DDT uses the GNU Debugger (GDB) under the covers**
- **Compile with symbolic debug information turned on**
- **With CCE use**
  - g for debug info and no optimisation
  - G1 or -G2 for debug info with optimisation
- **With Intel use**
  - g for debug info

## Launching DDT in batch

```
export DISPLAY=<your workstation>:0.0
```

```
module load ddt/5.0.1.3_42607
```

```
ddt -n ? -mpiargs 'more aprun args' a.out
```

**For example**

```
ddt -n 4 -mpiargs '-N 4 -ss -cc cpu -d 6' a.out
```

# Simple Job Script

```
#!/bin/ksh
#PBS -q np
#PBS -N ddtdemo
#PBS -j oe
#PBS -o job.out
#PBS -l EC_total_tasks=4
#PBS -l EC_threads_per_task=6
#PBS -l EC_hyperthreads=1
#PBS -l walltime=00:60:00

cd $HOME/Debug

module load ddt

export DISPLAY=juliet:0.0

export OMP_NUM_THREADS=6

ddt -n 4 -mpiargs '-ss -cc cpu -N 4 -d 6' ./hello_mpi
```

# Live Demo Using IFS

- **TL159 running on 1 node**
  - Initially with 4 tasks x 6 threads
  - Then with 24 tasks x 1 thread
- **Latest RAPS14 benchmark code (CY41R2)**
- **Run in batch**
- **Selected screen shots from DDT follow in the presentation material**
- **Now for the live demo.....**



# Before launching executable

The screenshot shows the Alinea Forge 5.0.1-42607 interface. The main window has a menu bar (File, Edit, View, Control, Tools, Window, Help) and a sidebar with the Alinea Forge logo and navigation options: **RUN** (Run and debug a program), **ATTACH** (Attach to an already running program), **OPEN CORE** (Open a core file from a previous run), **MANUAL LAUNCH (ADVANCED)** (Manually launch the backend yourself), and **OPTIONS** (Remote Launch: Off). The 'Run' dialog box is open, showing configuration for the application `/perm/systems/sy6/RAPS14.dbg/bin/ifsMASTER`. It includes settings for MPI (4 processes, Cray X-Series), OpenMP (6 threads), and Environment Variables (none). The 'Run' button is highlighted.

# Paused after MPI\_INIT

The screenshot displays the DDT debugger interface with the following components:

- Menu Bar:** File, Edit, View, Control, Tools, Window, Help.
- Toolbar:** Includes icons for running, stepping, and other debugging actions.
- Current Group:** All. Focus on current: Group, Process, Thread, Step Threads Together.
- Project Files:** A tree view on the left showing various Fortran modules like mpi\_barrier\_mod.F90, mpi\_broadcast\_mod.F90, etc., with mpi\_init\_mod.F90 selected.
- Code Editor:** Shows the source code for mpi\_init\_mod.F90. Line 161 is highlighted: `LTHSAFEMPI = .FALSE.`. The code includes MPI initialization logic with conditional compilation for different architectures.
- Locals/Current Line(s) Panel:** Shows the variable `LTHSAFEMPI` with a value of `.TRUE.`.
- Stacks Panel:** Shows the call stack with the current process `master (master.F90:71)` and the current function `mpi_init (mpi_init_mod.F90:161)`.
- Evaluate Panel:** An empty panel for evaluating expressions.
- Status Bar:** Shows "Ready".

# After selecting cnt4.F90 and setting a breakpoint at line 505

The screenshot displays the DDT (Data Display Tool) debugger interface. The main window shows the source code of the file `cnt4.F90`. A red circle indicates a breakpoint is set at line 505. The code at this line is `DO JSTEP=NSTAR2,ISTOP`. The interface includes a menu bar (File, Edit, View, Control, Tools, Window, Help), a toolbar with various execution and debugging icons, and several panels: 'Project Files' on the left showing a tree view of files; 'Current Line(s)' on the right showing the current line of code; 'Stacks' at the bottom left showing the current call stack with `mpl_init (mpl_init_mod.F90:161)` at the top; and 'Evaluate' at the bottom right for evaluating expressions. The status bar at the bottom right shows 'Ready'.

# Hitting the breakpoint

The screenshot shows the Alinea DDT debugger interface. The main window displays the source code of `cnt4.F90` with a breakpoint set at line 506. A dialog box titled "Program Stopped ->@cctmom2" is overlaid on the code, indicating that the program has stopped at the breakpoint. The dialog box contains the following text:

```
Processes 0-3:  
Process stopped at breakpoint in cnt4 (cnt4.F90:506).  
 Always show this window for user-defined breakpoints
```

Below the dialog box are two buttons: "Continue" and "Pause".

The interface also shows a "Locals" window on the right, which is currently empty. The "Input/Output" window at the bottom left shows the following text:

```
signal_drhook(SIGQUIT=3): New handler installed at 0x5534ee0; old  
signal_drhook(SIGTERM=15): New handler installed at 0x5534ee0; old  
signal_drhook(SIGXCPU=24): New handler installed at 0x5534ee0; old  
signal_drhook(SIGSYS=31): New handler installed at 0x5534ee0; old  
MPI_BUFFER_METHOD: 2 128000000  
Reading /perm/systems/sy6/RAPS14.dbg/data/ifsdata/MCICA  
Reading /perm/systems/sy6/RAPS14.dbg/data/ifsdata/RADRRTM  
Reading /perm/systems/sy6/RAPS14.dbg/data/ifsdata/RADSRM
```

The "Input/Output" window also has a text input field and a "More" button.



# At the breakpoint

The screenshot displays the Allinea DDT debugger interface. The main window shows the source code of a Fortran program with a breakpoint set at line 506. The code includes conditional logic for setting `JSTEP` and `ISTOP` based on `NCONF` values, and a call to `CALL USER_CLOCK`. The `Locals` panel on the right shows the current values of `ISTOP` (12), `JSTEP` (0), and `NSTAR2` (0). The `Input/Output` panel at the bottom left shows system messages related to signal handlers and MPI buffer methods.

Current Group: All | Focus on current: Group | Process | Thread | Step Threads Together

Current Line(s)

Variable Name	Value
ISTOP	12
JSTEP	0
NSTAR2	0

Input/Output

```
signal_drhook(SIGQUIT=3): New handler installed at 0x5534ee0; old
signal_drhook(SIGTERM=15): New handler installed at 0x5534ee0; old
signal_drhook(SIGXCPU=24): New handler installed at 0x5534ee0; old
signal_drhook(SIGSYS=31): New handler installed at 0x5534ee0; old
MPI_BUFFER_METHOD: 2 128000000
Reading /perm/systems/sy6/RAPS14.dbg/data/ifsdata/MCICA
Reading /perm/systems/sy6/RAPS14.dbg/data/ifsdata/RADRRTH
Reading /perm/systems/sy6/RAPS14.dbg/data/ifsdata/RADSRTH
```

Note: Allinea DDT can only send input to the aprun process with this MPI implementation

Type here ('Enter' to send):  More

Ready

# Setting a trace point at line 511 for JSTEP and ZT1 every 3<sup>rd</sup> step

The screenshot displays the Allinea DDT (Data Display Tool) interface. The main window shows a Fortran source file with the following code snippet:

```
498 ENDF
499
500 IF (LSLAG.AND.(NCONF == 131.OR.NCONF == 40)
501   LFINDVSEP=.TRUE.
502 ELSE
503   LFINDVSEP=.FALSE.
504 ENDF
505
506 DO JSTEP=NSTAR2,ISTOP
507
508   CALL USER_CLOCK(PTOTAL_CP=ZT1)
509
510   !* 3.1 Time filtering constraint
511
512 IF (LJCDFI.AND.NCONF==1) THEN
513   isub=1
514   ! Step should be local to subwindow
515   IF (NSUBDFI>1) THEN
```

The 'Edit Tracepoint' dialog box is open, showing the following configuration:

- Location:**
  - Line: File: `x11057/perm/systems/sy6/RAPS14.dbg/src/ifs/control/cnt4.F90`
  - Line Number: `511`
  - Function
- Applies To:**
  - Process Group: `All`
  - Process: `All`
  - Thread: `All`
- Tracepoint Output:**
  - Tracepoint Name:
  - Variables Logged: `JSTEP`, `ZT1`
- Hit Limits:**
  - Start on the n-th pass: `1`
  - Trigger every n-th pass: `3`
  - Log at most n times: `Forever`
- Condition:
- Language: `Auto`

Buttons: `Help`, `OK`, `Cancel`

# Setting a conditional breakpoint at line 507 for step 8

The screenshot displays the Alinea DDT interface with the following components:

- Top Panel:** Menu bar (File, Edit, View, Control, Tools, Window, Help) and toolbar with execution and debugging icons.
- Project Files:** A tree view on the left showing the project structure, with 'cnt4.F90' selected.
- Source Editor:** The main window showing Fortran code for 'cnt4.F90'. The code includes conditional logic and a call to 'CALL USER\_CLOCK'. Line 507 is highlighted.
- Breakpoint Dialog:** A modal window titled 'Edit Breakpoint' is open, showing the configuration for a breakpoint at line 507. The 'Condition' field is set to 'JSTEP.eq.8'.
- Input/Output:** A window at the bottom left showing system messages and MPI-related output.

```
498      ENDIF
499
500      IF (LSLAG.AND.(NCONF == 131.OR.NCONF == 401
501         LFINDVSEP=.TRUE.
502      ELSE
503         LFINDVSEP=.FALSE.
504      ENDIF
505
506      DO JSTEP=NSTAR2,ISTOP
507
508         CALL USER_CLOCK (PTOTAL_CP=ZT1)
509
510         !*   3.1   Time filtering constraint b
511
512         IF (LJCDFI.AND.NCONF==1) THEN
513             isub=1
514             ! Step should be local to subwindow
515             IF (NSUBDFI>1) THEN
```

**Edit Breakpoint ->@cctmom2<**

Location:

- Line: File: x11057/perm/systems/sy6/RAPS14.dbg/src/ifs/control/cnt4.F90
- Line Number: 507
- Function

Applies To:

- Process Group: All
- Process: All
- Thread: All

Hit Limits:

- Start on the n-th pass: 0
- Trigger every n-th pass: 1
- Stop after n hits: Forever

Condition: JSTEP.eq.8

Language: Auto

Buttons: Help, OK, Cancel

# Trace point output after 8 steps

The screenshot displays the Allinea DDT - Alinea Forge 5.0.1-42607 interface. The main window shows the Fortran source code for 'cnt4.F90'. A tracepoint is set at line 508, which is `CALL_USER_CLOCK(PTOTAL_CP=ZT1)`. The 'Tracepoint Output' window at the bottom left shows the following data:

Tracepoint	Processes	Values logged
cnt4 (cnt4.F90:511)	4, ranks 0-3	JSTEP: <b>1</b> ZT1: <b>7.2e+01 from 7.2e+01 to 7.2e+01</b>
cnt4 (cnt4.F90:511)	4, ranks 0-3	JSTEP: <b>4</b> ZT1: <b>2.11e+02 to 2.11e+02</b>
cnt4 (cnt4.F90:511)	4, ranks 0-3	JSTEP: <b>7</b> ZT1: <b>3.41e+02 to 3.41e+02</b>

The 'Locals' window on the right shows the current value of ZT1 as 340.757296. The code editor shows the following snippet:

```

500 IF (LSLAG.AND.(NCONF == 131.OR.NCONF == 401.OR.NCONF == 601.OR.NCONF == 1301))
501   LFINDVSEP=.TRUE.
502 ELSE
503   LFINDVSEP=.FALSE.
504 ENDIF
505
506 DO JSTEP=NSTAR2,ISTOP
507
508   CALL_USER_CLOCK(PTOTAL_CP=ZT1)
509
510   !* 3.1 Time filtering constraint based on digital filter
511
512 IF (LJCDFI.AND.NCONF==1) THEN
513   isub=1
514   ! Step should be local to subwindow
515 IF (NSUBDFI>1) THEN
516   ipersub=NSTOP/NSUBDFI
517   istep=MOD(jstep,ipersub)

```



## Running a 24 x 1 configuration

- **IFS runs until it gets a Floating Point Exception**
- **Caused by a divide by zero**
- **The physics time slice has gone to zero**
- **Will use a watch point to find out where**

# Hitting the divide by zero

The screenshot shows the Alinea DDT debugger interface. The main window displays the source code of a Fortran program. The error message indicates that the program stopped in the file `qnegat.F90` at line 89, where a floating point divide by zero occurred. The error message also provides the reason/origin and offers options to continue or pause the program.

Project Files:

- qanada.F90
- qapabl.F90
- qapabo.F90
- qapass.F90
- qapavu.F90
- qapcad.F90
- qapdgu.F90
- qaprex.F90
- qaref.F90
- qasset.F90
- qavara.F90
- qmfixer.F90
- qmfixer2.F90
- qneg.F90
- qnegad.F90
- qnegat.F90

Source Code (qnegat.F90):

```
81 !
82 !
83 ! ZTMST=TWODT
84 ! IF (NSTEP.EQ.NSTART) ZTMST=0.5*TWODT
85 IF (LHOOK) CALL DR_HOOK('QNEGAT',0,ZHOOK_HANDLE)
86 ASSOCIATE(REPQMI=>YRECND%REPQMI)
87 ZTMST=PTSE
88
89 ZCONS1=1.0/ZCONS1
90
91 ! INITIALI
92 IF (.NOT.
93
94 ! ***
95
96 !
97
98 ! *
```

Program Stopped <@cctmom2>

Process 4:

Process stopped in qnegat (qnegat.F90:89) with signal SIGFPE (Arithmetic exception).

Reason/Origin: floating point divide by zero  
Your program will probably be terminated if you continue.  
You can use the stack controls to see what the process was doing at the time.

Always show this window for signals

Continue Pause

Locals:

Variable Name	Value
ZCONS1	-9.1702764791777932e-05
ZTMST	0

Input/Output:

```
*** LVECTOR= F
*** IVECTOR= 1
*** LPREPROC= F
*** LWCONORMS= F
*** LMSSTG_WAM= F
=====
13:26:29 STEP 1 H= 1:00 +CPU= 6.304
13:26:34 STEP 2 H= 2:00 +CPU= 5.764
```

Note: Alinea DDT can only send input to the aprun process with this MPI implementation

Type here ('Enter' to send):  More

23 processes playing

# After reaching a breakpoint in callpar.F90 and setting a watch point on tsphy

The screenshot shows the Alinea DDT debugger interface. The main window displays the source code of `callpar.F90` with a breakpoint set at line 422. The code snippet is as follows:

```
414  
415 ! -----  
416  
417 !*      0.      INITIALIZATION  
418 !  
419 !      0.1    Initialization of constants  
420  
421 ! this should be moved to setup with some LCLOUDITER key in physi  
422 LL_CLOUDITER=.TRUE. .AND. (.NOT. LPHYLIN).AND. (.NOT. YREPHY%LSPCR  
423  
424 LLSLPHY = LSLPHY .AND. LSLAG  
425 LLRAIN1D = .FALSE. ! key reserved for 1D var rain  
426  
427 ! constants  
428 ZRG=1.0_JPRB/RG  
429 ZRCPD=1.0_JPRB/RCPD  
430  
431 ! Some sort of security (I wonder why this is not in the setup)
```

The Watchpoints window shows a watchpoint configured for the variable `tsphy` in the scope `#0 callpar`. The watchpoint is set to trigger on a "write only" event and is implemented in hardware.

Processes	Scope	Expression	Trigger On	Implemented in	
<input checked="" type="checkbox"/>	All	#0 callpar	tsphy	write only	hardware

The Locals window shows the current state of variables, with `tsphy` highlighted and its value set to 3600.

Variable Name	Value
ts8	0x8bb980 <yoeaeratm_>
ts89	65928
ts9	0x8bct20 <yoeaad_>
ts96	65748
ts99	65929
tendency_cml	
tendency_dyn	
tendency_loc	
tendency_tmp	
tendency_vdf	
tsphy	3600
ygfl	( numflds = 17, nders = 0, numspflds = 0, nur
yrchem	
yrchemfs	( lecumfs = .FALSE., iregcv = .FALSE. )
yrdphy	( ncss = 4, nrxp = 0, nrxp2 = 0, nrxp = 0, ncsi
yreaeratm	( naerconf = -99, niniday = 19000101, nrx3dae

# After the watch point triggers in sltend.F90

The screenshot shows the AAllinea DDT 4.2.1-36484 interface. The main window displays the source code for `sltend.F90`. A watchpoint has been triggered at line 415, `IF (LHOOK) CALL DR`. A dialog box titled "Program Stopped <@cctmom2>" is overlaid on the code, providing details about the stop:

- Process 4:
- Process stopped at watchpoint "tsphy" in sltend (sltend.F90:415).
- Old value: 3600
- New value: 0
- Always show this window for watchpoints
- Buttons: Continue, Pause

The "Locals" window on the right shows the current state of variables:

Variable Name	Value
zepsilon	2.2204460492503131e-14
zfac	1.0454158174220698
zfacc	0.28346868942621073
zhook_handle	3.2430270181003537e-315
zpp	([1] = 94379.106594719313, ...)
zpt	([1] = 264.15581262010329, ...)
	([1] = 0, ...)
	([1] = 0.0016206026841087756, ...)
	0.0028522486915119531
	0.10197162129779283
	([1] = 0.017217980924555294, ...)
	([1] = 0.0017161058851406594, ...)
	([1] = 262.82154429644334, ...)
	([1] = 1.0909490209890669, ...)
	([1] = 0.017217980924555294, ...)
	([1] = 1.0909490209890669, ...)

The "Input/Output" window at the bottom left shows the output for process 4:

```
Other: *** IVECTOR= 1
Other: *** LPRKPROC= F
Other: *** LWOUNORMS= F
Other: *** LSMSSIG_WAM= F
Other: =====
aprun: 12:41:03 STEP 1 H= 1:00 +CPU= 6.364
aprun: 12:41:08 STEP 2 H= 2:00 +CPU= 5.848
```

Note: AAllinea DDT can only send input to the aprun process with this MPI implementation.

Type here (Enter to send):  More

23 processes playing



# Line 412 did it!

The screenshot shows the Alinea DDT - Alinea Forge 5.0.1-42607 interface. The main window displays Fortran code for `mpi_init_mod.F90`. Line 412 is highlighted, containing the statement `if(myproc.eq.5.and.nstep.eq.3) tsphy=0.0`. The `Locals` window on the right shows the following variables and values:

Variable Name	Value
zepsilon	2.2204460492503131e-14
zfac	1.0454158174220698
zfaccc	0.28346868942621073
zhook_handle	3.2439983906853292e-315
zpp	([1] = 94379.106594719313, ...)
zpt	([1] = 264.15581262010329, ...)
zqad	([1] = 0, ...)
zqold	([1] = 0.0016206026841087756, ...)
zqp1env	0.0028522486915119531
zrg_r	0.10197162129779283
zsuba	([1] = 0.017217980924555294, ...)
zsubq	([1] = 0.0017161058851406594, ...)
zsubt	([1] = 262.82154429644334, ...)
zzfac	([1] = 1.0909490209890669, ...)
zzsuba	([1] = 0.017217980924555294, ...)
zzzfac	([1] = 1.0909490209890669, ...)

The `Input/Output` window at the bottom left shows the following output:

```
*** LVECTOR= F
*** IVECTOR= 1
*** LPREPROC= F
*** LWCOUNORMS= F
*** LSHSIG_WAH= F
-----
13:43:45 STEP 1 H= 1:00 +CPU= 6.304
13:43:51 STEP 2 H= 2:00 +CPU= 5.772
```

Note: Alinea DDT can only send input to the aprun process with this MPI implementation  
Type here ('Enter' to send):  More

23 processes playing

# Allinea Map

- **A product that complements DDT**
- **A simple code profiler for MPI application**
- **Version 5 supports OpenMP as well**
- **Designed to be lightweight at scale**
  - 1000 samples per thread
  - Sampling rate adjust automatically
- **Same source code browser as DDT**
- **Find hot spots in the application**

# Using Alinea Map

- **Compile with debug info**
  - **-G2 for Cray compiler, -g for Intel/Gnu**
- **Load the DDT module**
  - **module load ddt/5.0.1.3\_42607**
- **Execute make\_profiler\_libraries script**
  - **Creates libmap-sampler.so libmap-sampler-pmpi.so**
- **Link libraries into application**
  - **-lmap-sampler -lmap-sampler-pmpi**
- **Run in batch to collect data via map --profile**
  - **Creates a “profiling file”**
- **Analyse interactively via map “profiling file”**

## Using Alinea Map in batch

```
module load ddt/5.0.1.3_42607
```

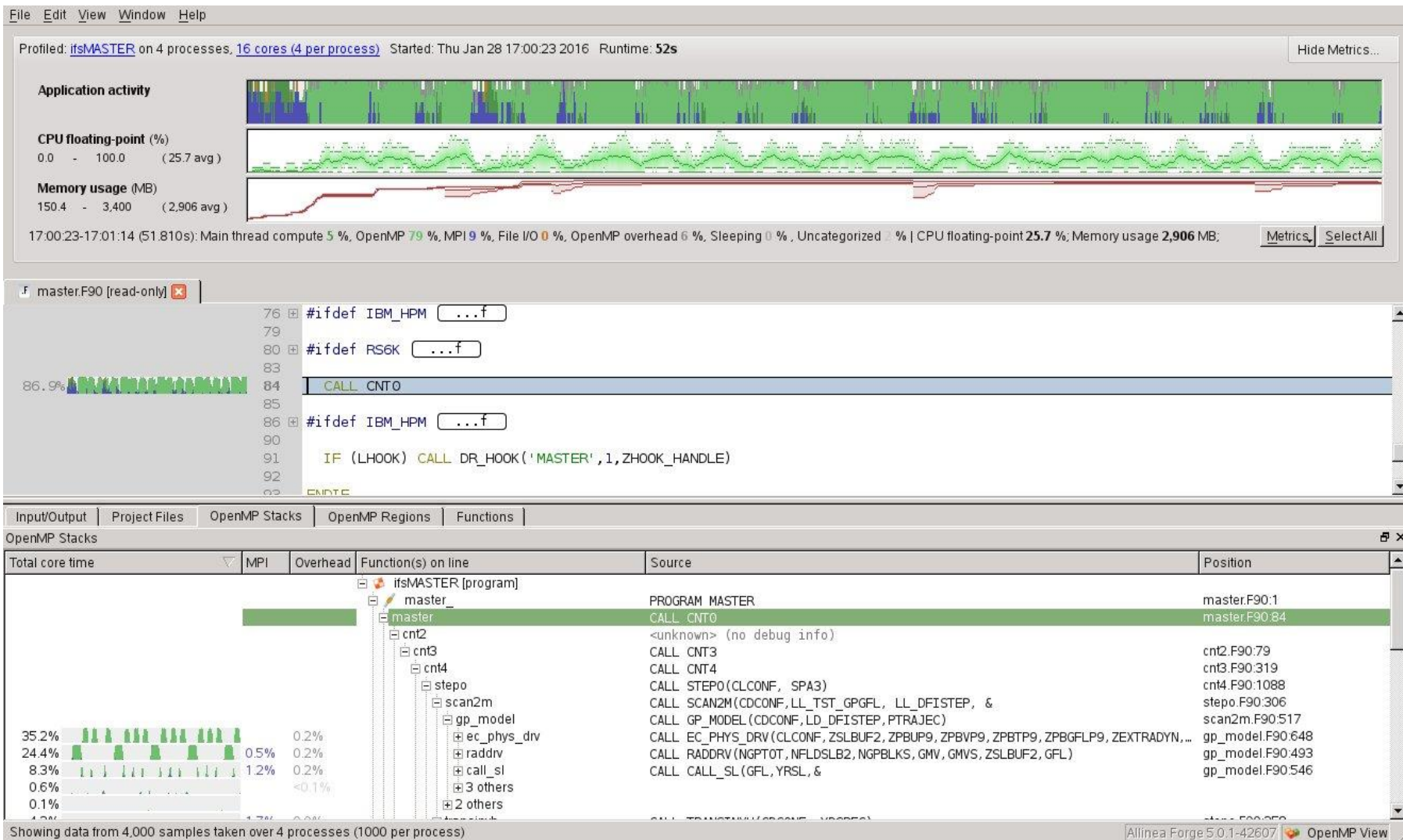
```
map --profile -n ? --mpiargs 'more aprun args' a.out
```

**For example**

```
map --profile -n 4 --mpiargs '-N 4 -ss -cc cpu -d 6' a.out
```



# Allinea Map - Screenshot



# Using Alinea Map

- **See the DDT/MAP userguide for more information**
  - `/opt/cray/ddt/4.2.1.2_36484/doc/userguide.pdf`
  - `/opt/cray/ddt/5.0.1.3_42607/doc/userguide-forge.pdf`

# Very Simple DDT Example

```
tar -xzf ~trx/exercises/debug.tgz
```

```
cd Debug
```

```
compit
```

```
edit job.ddt and set DISPLAY
```

```
qsub job.ddt
```

Try setting

- a breakpoint at line 26
- a watch point on nthreads
- a conditional trace point on line 50 for nstep 50

Find the error output and trace output when it fails