Argonne NATIONAL LABORATORY

The Small and Large of Debugging on Blue Gene/Q

Scaling Your Science on Mira

May 24, 2016

Applications Performance Engineering

ALCF

U.S. DEPARTMENT OF ENERGY

Outline

- Interactive jobs
- Log files
- Exit semantics
- bgq_stack
- coreprocessor
- Serial debugger (gdb)
- Parallel debugger (DDT)

Interactive runs for tests

- Submit an interactive job to the queue, e.g.
  - qsub –I –t 30 –n 512
- When job "runs", the requested nodes are allocated to you, and you receive a (new) shell prompt.
- This shell behaves like the one in a Cobalt script job
  - Just one difference: do "wait-boot" before proceeding
  - Start your compute node run just like in a Cobalt script job
    - runjob –block $COBALT_PARTNAME --np 512 –p 16 : myprogram.exe
- When you exit the shell, the Cobalt job will end
- Note: When the Cobalt job runs out of time, there is no message. Runjob will fail.
  - Check your job status with "qstat $COBALT_JOBID"

Interpreting your job's log files

.cobaltlog

...
Tue May 19 20:11:47 2015 +0000 (UTC) rloy/264197: Block VST-22260-33371-32 for location VST-22260-33371-32 successfully booted (Initiating).
Tue May 19 20:12:57 2015 +0000 (UTC) Info: task completed normally with an exit code of 10; initiating job cleanup and removal

.error

...
2015-05-19 20:11:42.851 (INFO) [0x40000bfbdd0] 3369:tatu.runjob.client: scheduler job id is 264197
2015-05-19 20:11:42.853 (INFO) [0x400005c34d0] 3369:tatu.runjob.monitor: monitor started
2015-05-19 20:11:42.969 (INFO) [0x400005c34d0] 3369:tatu.runjob.monitor: task record 952637 created
2015-05-19 20:11:42.970 (INFO) [0x40000bfbdd0] VST-22260-33371-32:3369:ibm.runjob.client.options.Parser: set local socket to runjob_mux from properties file
2015-05-19 20:11:45.578 (INFO) [0x40000bfbdd0] VST-22260-33371-32:1142676:ibm.runjob.client.Job: job 1142676 started
2015-05-19 20:11:59.554 (INFO) [0x400005c34d0] 3369:tatu.runjob.monitor: tracklib completed
2015-05-19 20:12:47.393 (INFO) [0x40000bfbdd0] VST-22260-33371-32:1142676:ibm.runjob.client.Job: exited with status 10
2015-05-19 20:12:47.393 (WARN) [0x40000bfbdd0] VST-22260-33371-32:1142676:ibm.runjob.client.Job: normal termination with status 10 from rank 0
2015-05-19 20:12:47.393 (INFO) [0x40000bfbdd0] tatu.runjob.client: task exited with status 10
2015-05-19 20:12:47.394 (INFO) [0x400005c34d0] 3369:tatu.runjob.monitor: monitor terminating
2015-05-19 20:12:47.397 (INFO) [0x40000bfbdd0] tatu.runjob.client: monitor completed

runjob:

"To exit, or not to exit, that is the question…"

- Any rank calls exit(0) → Wait for other ranks to call exit()
- Any rank has uncaught signal → Kill all ranks now*
- Any rank calls exit(1) → Kill all ranks right now*
- Some rank calls exit(n) for n>1
  - Default → wait for other ranks to call exit()
    - ***Probably not what you expect. Program likely to deadlock.***
  - BG_EXITIMMEDIATLYONRC=1 → Kill all ranks right now*

*As soon as some rank(s) cause runjob termination, the other ranks are killed, ***possibly before they have the opportunity to abort***

- You may get fewer core files than you expect.

Lightweight core files

- When run fails, look for core files
  - core.0, core.1, etc.
- Lightweight core files
  - One for each rank that failed *before job teardown*
  - Contain stack backtrace in *address* form
  - Decode to symbolic (useful!) form
- Environment settings to control core files
  - http://www.alcf.anl.gov/user-guides/core-file-settings

Lightweight Core File Example

+++PARALLEL TOOLS CONSORTIUM LIGHTWEIGHT COREFILE FORMAT version 1.0
+++LCB 1.0
Program : /gpfs/vesta-home/rloy/src/test/idie
Job ID : 1142376
Personality:
ABCDET coordinates : 0,0,0,0,0,0
Rank : 0
Ranks per node : 16
[...]
+++ID Rank: 0, TGID: 1, Core: 0, HWTID:0 TID: 1 State: RUN
***FAULT Encountered unhandled signal 0x00000006 (6) (SIGABRT)
General Purpose Registers:
[...]
Special Purpose Registers:
[...]
Floating Point Registers
[...]
Memory:
[...]
+++STACK
Frame Address Saved Link Reg
0000001fbfffb700 0000000001001848
0000001fbfffb8c0 00000000010003e8
0000001fbfffb960 0000000001000438
[...]
---STACK
[...]

Decoding Lightweight Core Files

- bgq_stack [optional_exename] [corefile]

+++ID Rank: 0, TGID: 1, Core: 0, HWTID:0 TID: 1 State: RUN

0000000001001848
abort
/bgsys/drivers/V1R2M2/ppc64/toolchain/gnu/glibc-2.12.2/stdlib/abort.c:77

00000000010003e8
barfunc
/gpfs/vesta-home/rloy/src/test/idie.c:6

0000000001000438
foofunc
/gpfs/vesta-home/rloy/src/test/idie.c:12

0000000001000498
main
/gpfs/vesta-home/rloy/src/test/idie.c:19

[...]

coreprocessor

- Useful when you have a large set of core files
  - Shows symbolic backtrace
  - Groups ranks that aborted in the same location together
  - *Can also attach to a running job to take snapshot*
- Location
  - BG/Q: coreprocessor.pl is in your default PATH
    - Attaching to running job does **not** require administrator
    - coreprocessor -nogui -snapshot=<filename> -j=<jobid>
      - Use the back-end (ibm.runjob) jobid from the .error file, not the Cobalt jobid
- Scalability limit
  - **Absolute maximum** 32K ranks. Practical limit lower.
- Instructions:
  - BG/Q Application Developer Redbook
    - http://www.redbooks.ibm.com/redpieces/abstracts/sg247948.html

coreprocessor window

File Control Analyze Filter Sessions

Group Mode: Stack Traceback (condensed) Session 1 (MMC

```
0 :Compute Node (128)
1 :    0xfffffffc (128)
2 :        __libc_start_main (32)
3 :            generic_start_main (32)
4 :                main (16)
5 :                    Allgather (16)
6 :                        PMPI_Allgather (16)
7 :                            MPIDO_Allgather (8)
8 :                                MPIDO_Allreduce (8)
9 :                                    MPID_Progress_wait (1)
10:                                        DCMF_CriticalSection_cycle (1)
9 :                                    MPID_Progress_wait (7)
10:                                        DCMF_Messager_advance (1)
11:                                            DCMF::Queueing::Lockbox::Device::advance() (1)
10:                                        DCMF_Messager_advance (1)
11:                                            DCMF::Queueing::Tree::Device::advance() (1)
10:                                        DCMF_Messager_advance (5)
11:                                            DCMF::DMA::Device::advance() (2)
12:                                                DCMF::DMA::RecFifoGroup::advance() (2)
13:                                                    DMA_RecFifoSimplePollNormalFifoById (2)
11:                                            DCMF::DMA::Device::advance() (3)
7 :                            MPIDO_Allgather (8)
8 :                                MPIDO_Allreduce (8)
9 :                                    MPIR_Allreduce (8)
10:                                        MPIC_Sendrecv (8)
11:                                            MPID_Progress_wait (8)
12:                                                DCMF_Messager_advance (8)
13:                                                    DCMF::Queueing::GI::Device::advance() (1)
13:                                                    DCMF::DMA::Device::advance() (3)
14:                                                        DCMF::DMA::RecFifoGroup::advance() (3)
15:                                                            DMA_RecFifoSimplePollNormalFifoById (3)
```

gdb

- A single gdb client can connect to single rank of your job
- BG/Q Limitations
  - Each instance of gdb client counts as a "debug tool"
  - Only 4 tools may be connected to a job
    - *At most 4 ranks can be examined*
- Start a debug session using ***qsub –I*** (interactive job)
  - qsub –I –q default –t 30 –n 64
  - See Redbook for more info on starting gdb with runjob
- gdb can also load a compute-node **binary** corefile
  - *Use extreme caution when generating binary corefiles*
- Generally a parallel debugger (e.g. DDT) will be more useful

Allinea DDT

- Licensing
  - 132K-process permanent license for all BG/Q hosts
  - Full machine development license available
    - Also supports Tukey
- Add the softenv key “+ddt”
- Compiling your code
  - Compile –g –O0
  - Note: XL compiler option -qsmp=omp also turns on optimization within OMP constructs. To override, use "noopt", e.g.
    - -qsmp=omp:noauto:noopt
- More details:
  - http://www.alcf.anl.gov/user-guides/allinea-ddt

Allinea DDT startup

- Run using remote client (RECOMMENDED)
  - Download and install Mac or Windows "Remote client" from http://www.allinea.com/products/download-allinea-ddt-and-allinea-map
  - Optional: use ssh master mode so you only need log in once per session
    - Note: supported on Mac OS/X; not supported in Windows <= XP (? for >XP)
    - ~/.ssh/config
      - ControlMaster auto
      - ControlPath ~/.ssh/master-%r@%h:%p
- Run from login node
  - Need X11 server on your laptop and ssh –X forwarding
  - Run ddt and let it submit job through GUI

DDT Remote Client (0)

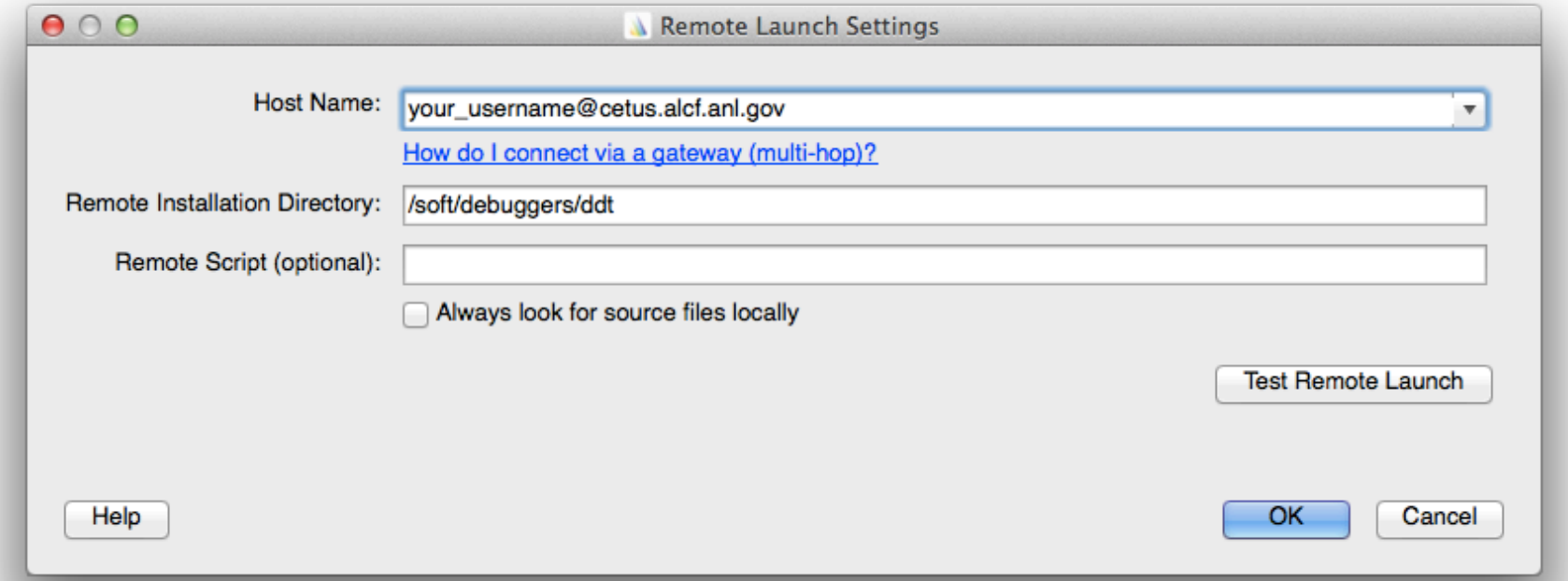
GUI looks just like the regular version

DDT Remote Client (1)

Select "configure" to add a new remote host

DDT Remote Client (2)

Note: this remote installation directory is the default version of DDT, corresponding to +ddt
Click "Test Remote Launch" to verify

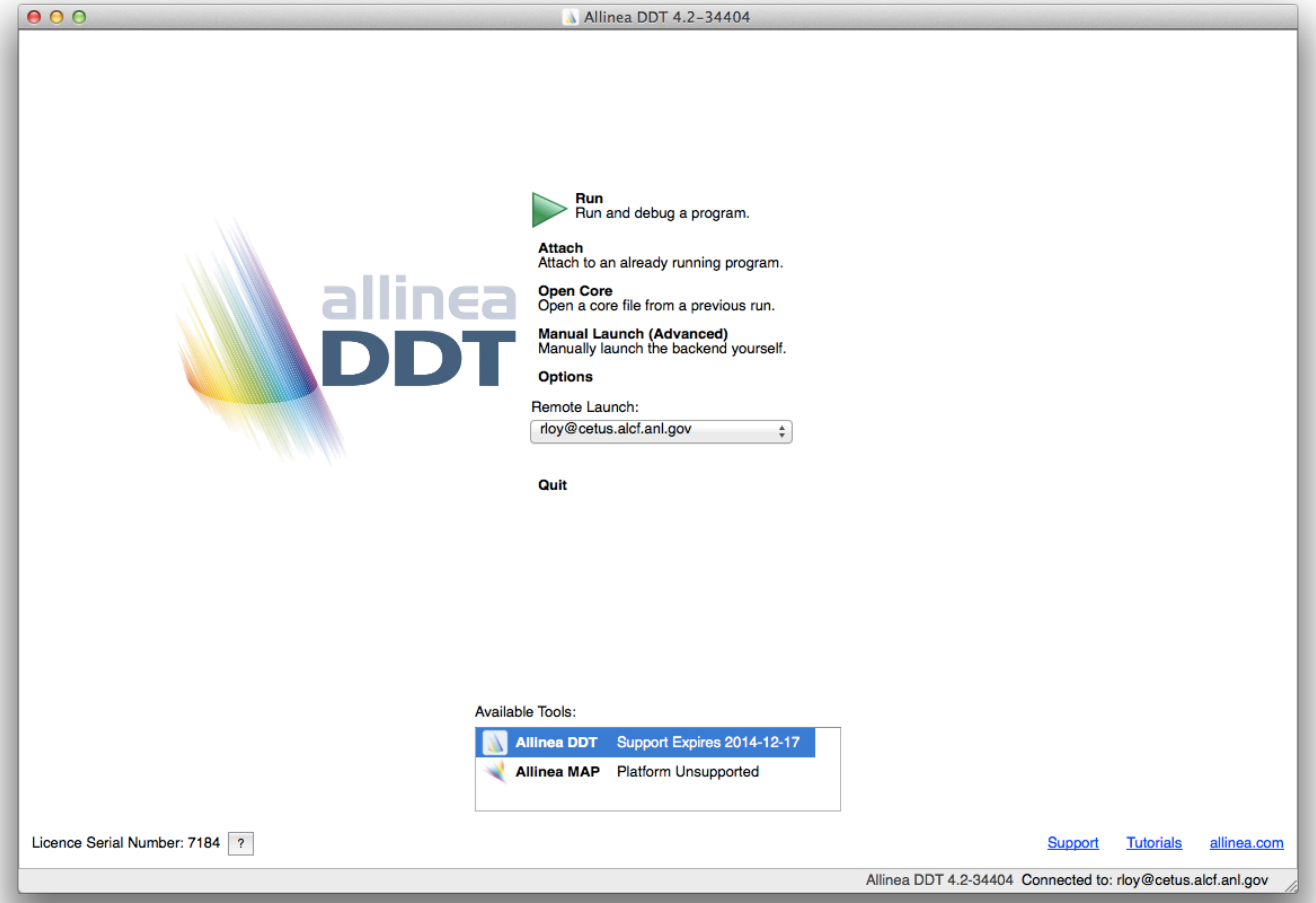


DDT Remote Client (3)

Now that it is defined, select remote machine

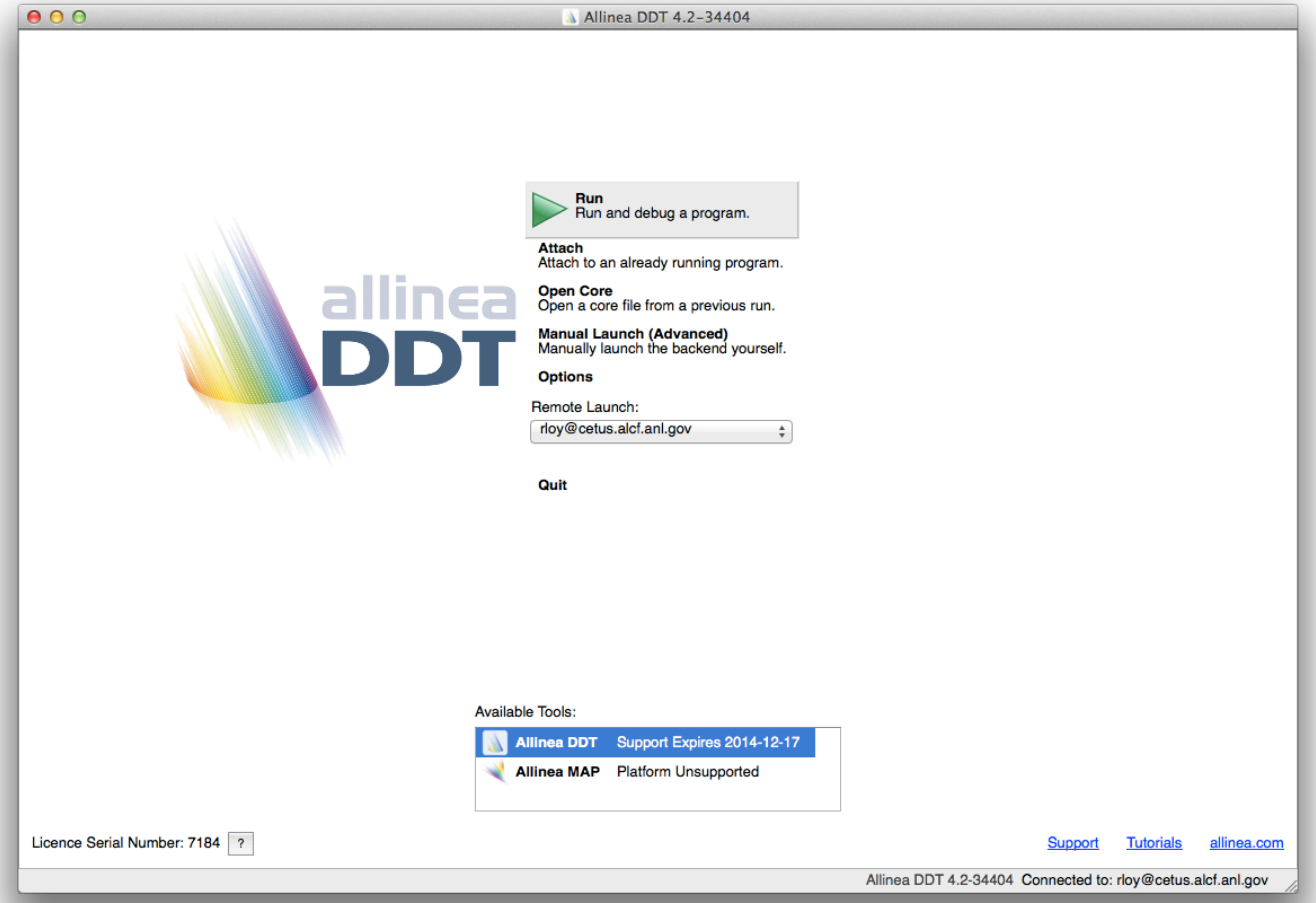
DDT (4)

Connected (note License info in lower left corner)
From this point, remote GUI works same as local



DDT (5)

Click "Run" to start a debugging session

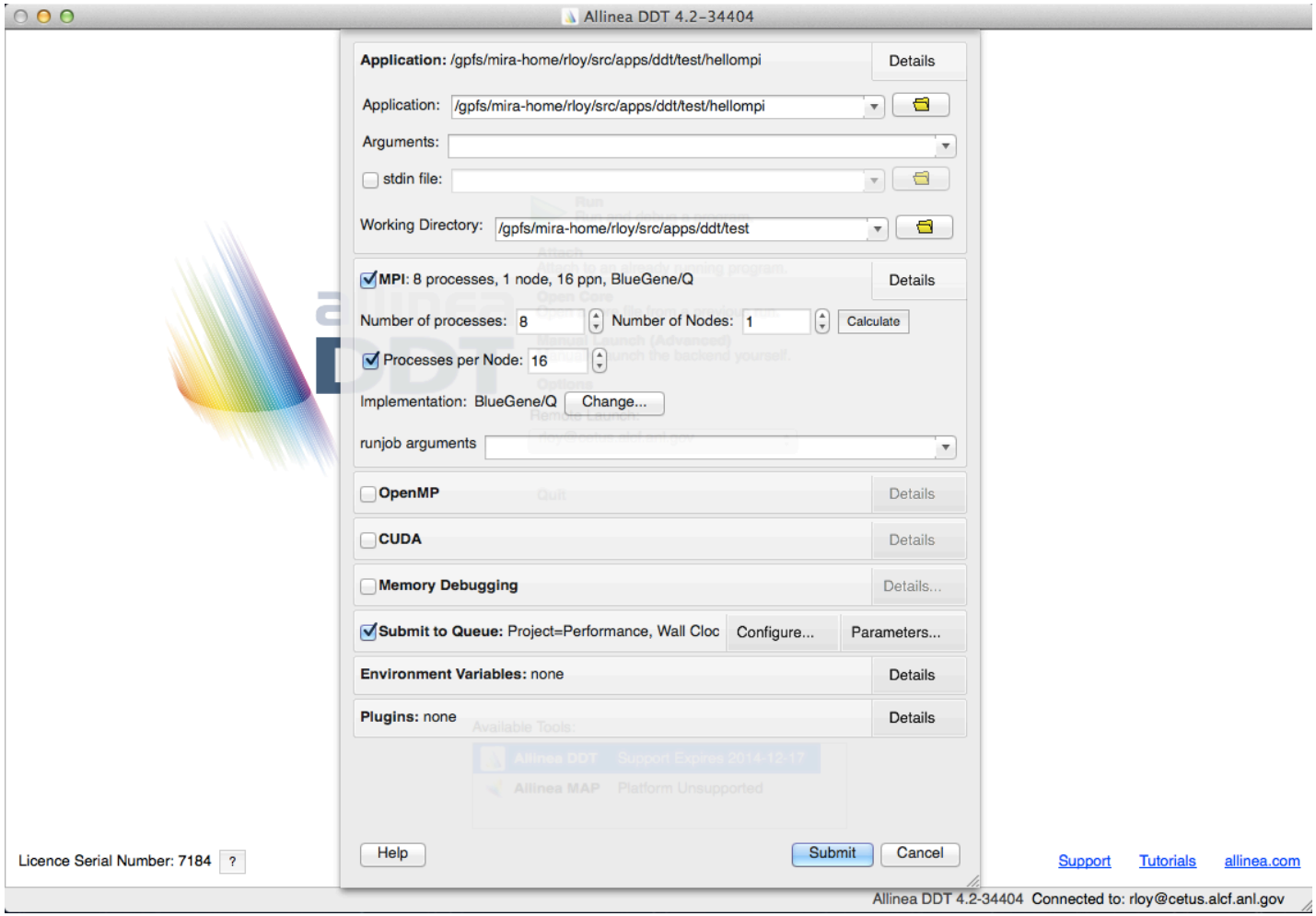


DDT (6)

Remember to set working directory

Important! Enable the checkbox "Submit to Queue"

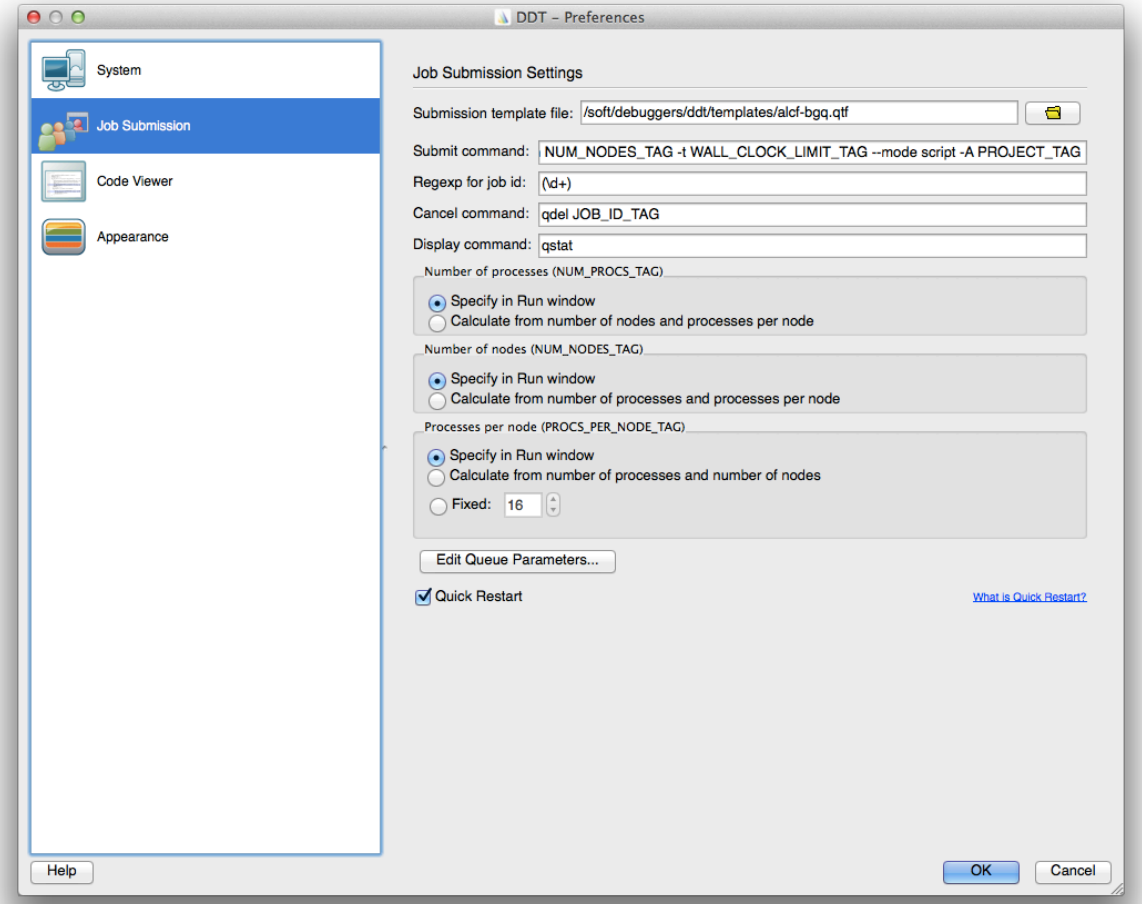
 - click "Configure" and "Parameters" for additional settings



DDT (6.1)

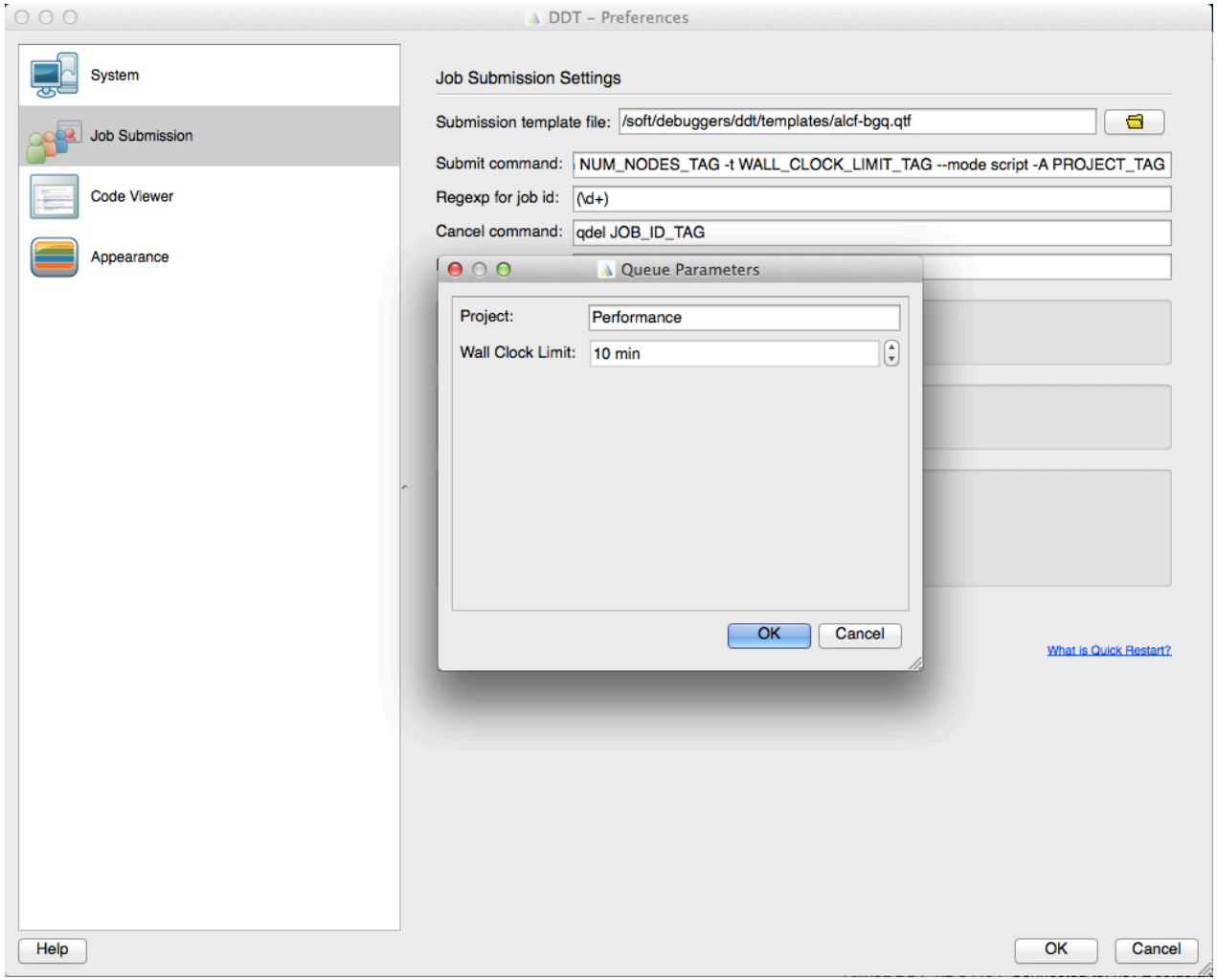
Job submission tab

Use submission template: /soft/debuggers/ddt/templates/alcf-bgq.qtf



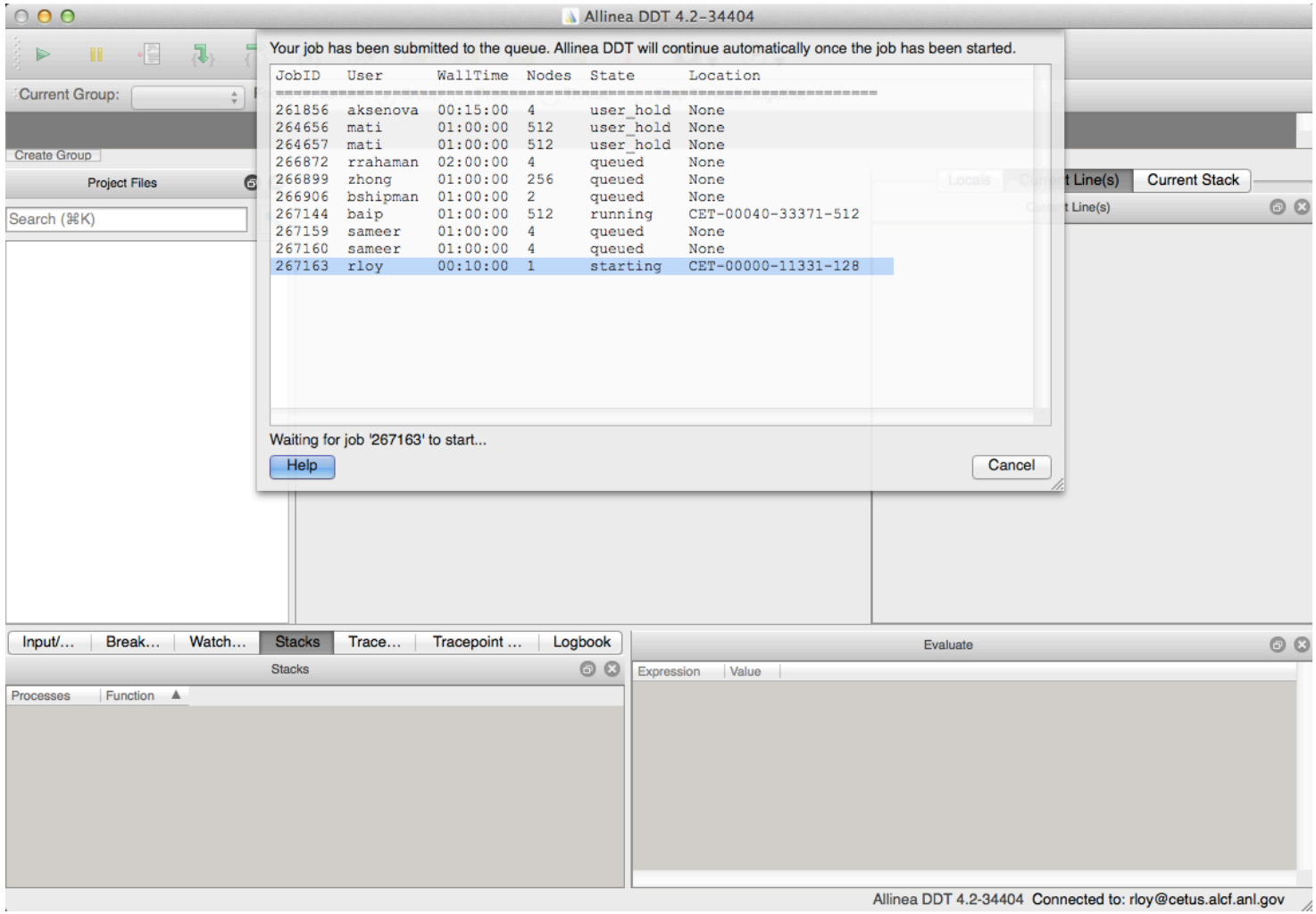
DDT (6.2)

Remember to set your project



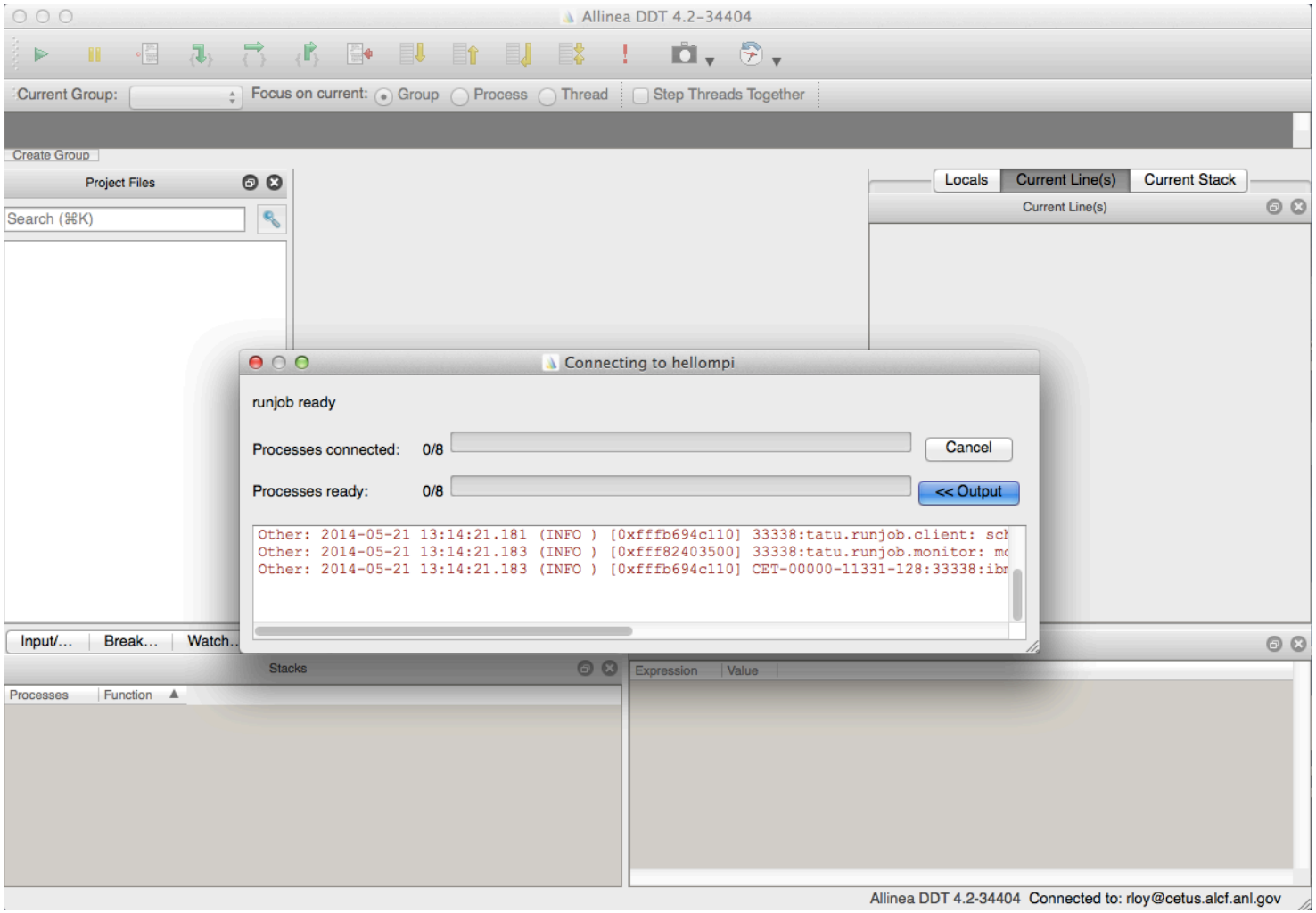
DDT (7)

Job must go through queue



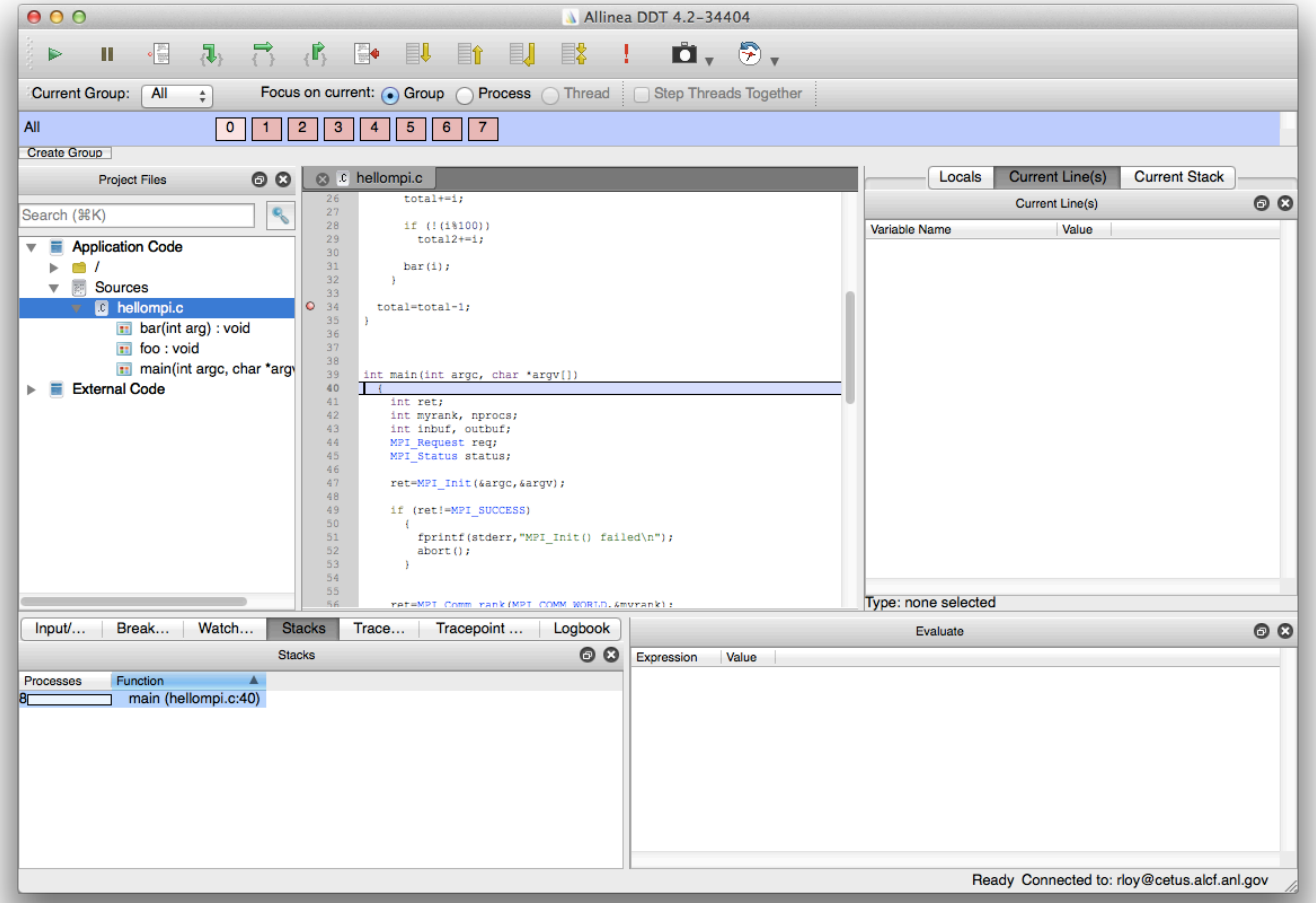
DDT (8)

When job starts running, connection status will show



DDT (9)

Ready to debug!



Questions?

- See also:
  - http://www.alcf.anl.gov/user-guides/mira-cetus-vesta
  - support@alcf.anl.gov