

# Intel Debugger

## Debugging serial applications

The intel debugger version 13.0 is available, via module intel. The debugger works for applications compiled with C and C++ compiler and the ifort fortran 77/90/95 compiler. The debugger provides java GUI environment. Use X display for running the GUI.

```
$ module load intel
$ idb
```

The debugger may run in text mode. To debug in text mode, use

```
$ idbc
```

To debug on the compute nodes, module intel must be loaded. The GUI on compute nodes may be accessed using the same way as in the GUI section

Example:

```
$ qsub -q qexp -l select=1:ncpus=16 -X -I
qsub: waiting for job 19654.srv11 to start
qsub: job 19654.srv11 ready
```

```
$ module load intel
$ module load java
$ icc -O0 -g myprog.c -o myprog.x
$ idb ./myprog.x
```

In this example, we allocate 1 full compute node, compile program myprog.c with debugging options -O0 -g and run the idb debugger interactively on the myprog.x executable. The GUI access is via X11 port forwarding provided by the PBS workload manager.

## Debugging parallel applications

Intel debugger is capable of debugging multithreaded and MPI parallel programs as well.

### Small number of MPI ranks

For debugging small number of MPI ranks, you may execute and debug each rank in separate xterm terminal (do not forget the X display). Using Intel MPI, this may be done in following way:

```
$ qsub -q qexp -l select=2:ncpus=16 -X -I
qsub: waiting for job 19654.srv11 to start
qsub: job 19655.srv11 ready
```

```
$ module load intel impi
$ mpirun -ppn 1 -hostfile $PBS_NODEFILE --enable-x xterm -e idbc ./mympprog.x
```

In this example, we allocate 2 full compute node, run xterm on each node and start idb debugger in command line mode, debugging two ranks of mympprog.x application. The xterm will pop up for each rank, with idb prompt ready. The example is not limited to use of Intel MPI

### Large number of MPI ranks

Run the idb debugger from within the MPI debug option. This will cause the debugger to bind to all ranks and provide aggregated outputs across the ranks, pausing execution automatically just after startup. You may then set break points and step the execution manually. Using Intel MPI:

```
$ qsub -q qexp -l select=2:ncpus=16 -X -I
qsub: waiting for job 19654.srv11 to start
qsub: job 19655.srv11 ready
```

```
$ module load intel impi
$ mpirun -n 32 -idb ./mympprog.x
```

### Debugging multithreaded application

Run the idb debugger in GUI mode. The menu Parallel contains number of tools for debugging multiple threads. One of the most useful tools is the **Serialize Execution** tool, which serializes execution of concurrent threads for easy orientation and identification of concurrency related bugs.

### Further information

Exhaustive manual on idb features and usage is published at Intel website, [http://software.intel.com/sites/products/documentation/doclib/stdxe/2013/composerxe/debugger/user\\_guide/index.htm](http://software.intel.com/sites/products/documentation/doclib/stdxe/2013/composerxe/debugger/user_guide/index.htm)