Debugging in Serial & Parallel

Basic Debugging

- Instrumentation method: instrument the code with print statement to check values and follow the execution of the program
- Use debugging tools
  - gdb

Introduction to gdb

- GNU debugger “gdb” is a program to help find bugs in the code.
- Compile the code with “-g” option to enable debugging.
  - “gcc -g -Wall -o hello hello.c”
  - “-g” option tells gcc to create a *symbol table* so that gdb can translate machine addresses into information that programmers can read.

Running Within Debugger

Inside a debugger:

- Look at source code listing
- Do a line-by-line execution
- Insert “breakpoints” at certain functional points
- Monitor values of variables
- “Backtrace” when code crashes

Basic gdb Commands

- run: this starts the program.
  - For example, if the program starts with "./prog in_put out_put"
  - In gdb, it starts with "run in_put out_put"
- print: this prints the contents of a variable.
- quit: quit gdb
- continue: continue execution.
- step: execute the next line of code, step into functions.
- next: execute the next line of code, do not step into functions.
- break <line number>: stop execution when the code is in <in number>.
- break <function>: stop execution when it reaches the <function>
- where: print a trace showing the sequence of function calls from main().
- backtrace: gives a stack backtrace showing what the program was doing

http://www.gnu.org/software/gdb/

More Debugging Tools

- idb: part of the Intel compiler suite. It has a special “-gdb” option for using gdb command syntax.
- Idb-gui: GUI for Intel compiler suite debugger
- ddd: a graphic front-end for gdb.
- pgdbg: part of PGI compiler suite.

Memory Allocation Tools

- efence: or Electric Fence, tries to trap any out-of-bounds references when using dynamic memory allocation

Parallel Debugging

TotalView: The “premier” parallel debugger.

- On CRC, use command “module load totaview” to load the debugger
- MPI programs behave as multiple processes within TotalView.
- Compile a “debuggable” executable
- Start the program under Totalview
  - totalview my_program
  - Select ‘Parallel’ tab, and choose “mpich2” from the pull down menu
  - Then select number of tasks
  - Set run arguments
- Now are read to start debugging