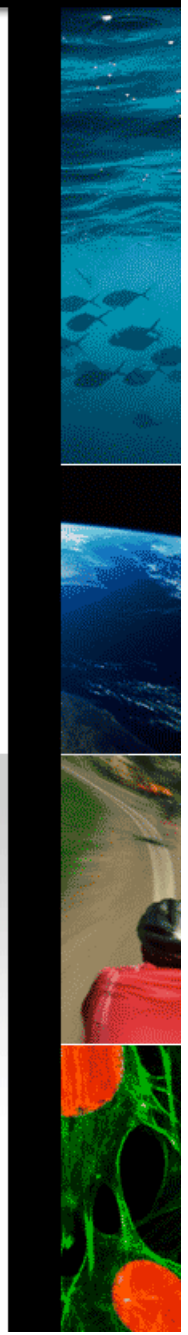
The Supercomputer Company



C/C++ Programming for the Cray XT3 and Cray Red Storm Systems

Geir Johansen

Cray Inc.

Overview

- C/C++ Programming Environment Basics
- Catamount Microkernel Issues
- PGI C/C++ Compiler
- Future Opportunities

Cray XT3 Architecture in 60 Seconds

- Code built on login service nodes running Linux
- Applications run on compute nodes running Catamount microkernel
- X86-64
- Static libraries only
- Each PE has its own memory

Cray XT3 Architecture in 60 Seconds

- Application has dedicated use of processor and memory on compute nodes
- Catamount has a subset of glibc functionality
- I/O performed by service nodes running Linux
- Portland Group compilers only supported compilers for compute nodes

modules

- Modules used to initialize programming environment
- **PrgEnv** modulefile used to load other programming environment modules and system modules
- Current dependencies exist between modules
  - PGI 5.2 C++ and MPT

Programming Environment modules

- pgi - Portland Group compilers
- xt-pe - Compiler drivers
- xt-mpt - Cray MPICH2 MPI-2, Cray SHMEM routines
- acml - AMD Core Math Library
- xt-libsci - Cray XT3 LibSci scientific library
- gcc - Gnu C Library 3.2.3 routines

Compiler Drivers

- cc, CC
- Must be used.
- Sets up compiler flags, header file paths, and library file paths for the Catamount target
- Used for compiling MPI code (not mpicc)
- To compile code for login nodes, use pgcc/pgCC or gcc/g++

Libraries Searched by the C/C++ Compiler Drivers

- libmpich.a - Cray MPICH2
- libacml.a - AMD Core Math library
- liblustre.a - Lustre file system I/O routines
- libpgc.a - PGI C compiler library
- libm.a - Catamount glibc math routines
- libcatamount.a - Catamount system routines
- libsysio.a - File system I/O routines

Libraries Searched by the C/C++ Compiler Drivers (continue)

- libportals.a - Portals routines
- libc.a - Catamount glibc routines
- libC.a - PGI C++ Standard library
- libgcc.a - GNU C library routines
- libsma.a - SHMEM library (**Non-default**)
- libpapi.a - PAPI library (**Non-default**)
- libgmalloc.a -glibc version of malloc (**Non-default**)

Using Linux Libraries

- If a Linux library is being linked in the application, be sure to copy the library from its normal Linux directory (i.e. /usr/lib64)

MPICH2 and C++ Incompatibility

- Name conflict for C++ code including mpi.h and stdio.h (iostream includes stdio.)
- Both header files define SEEK_SET, SEEK_CUR, and SEEK_END
- This is a MPI-2 problem
- Other vendor systems don't implement MPI seek definitions

MPICH2 and C++ Incompatibility

- To ignore MPI seek definitions:
  - Use –DMPICH_IGNORE_CXX_SEEK
  - Or include mpi.h before stdio.h
- To use MPI seek definitions:
  - #undef seek values before including mpi.h
- In future, -DMPICH_ENABLE_CXX_SEEK used to define MPI seek definitions

Target Machine Macros

- Predefined macros to specify code is targeted for the Cray XT3 (i.e. #ifdef statement)
  - __QK_USER__
  - __LIB_CATAMOUNT__

Catamount glibc Support

- Catamount eliminates the overhead of a full OS
- Processor and memory on compute node is dedicated to the application
- Catamount does not support the following glib functions
  - Sockets, pipes, remote procedure calls, or other TCP/IP routines
  - Dynamic process control: fork, exec, system
  - Share memory routines: shm_open

Catamount glibc Support

- Catamount does not support the following glib functions(continue)
  - Dynamic library routines: dlopen
  - Pthreads
  - getcwd call
  - Functions requiring database: getuid
  - Limited support for signals and ioctl
- Work arounds for non-supported glibc functions

malloc, mmap

- Customized version of malloc
  - Tuned for applications with large contiguous data arrays
- heap_info routine gives memory usage info
- glibc version of malloc is available by specifying '-lgmalloc'
- mmap not supported
  - map called with MAP_ANONYMOUS flag can be replaced with a call to malloc

times, clock, and _rtc

- times, clock, and _rtc functions not supported
- Use dclock routine to calculate elapsed time
- gettimeofday, getrusage, MPI_Wtime, and Fortran cpu_time are supported
  - Same clock as dclock
  - dclock has lowest calling overhead
  - User and system time are identical for getrusage

system

- Typically used to call a command
- Usually replaced by library routine
- Example: system(“mkdir /tmp)” becomes mkdir(“/tmp, 0750)

getpid

- Supported, but not usually helpful
- Use nid to get a unique value for each process of a parallel program
- Example:

```
#include <catamount/data.h>
unsigned getnid() { return((unsigned)_my_pnid); }
```

getrlimit, setrlimit

- getrlimit: all but data size, stack size, and number of processes return unlimited
- File I/O limits from service nodes not returned
- setrlimit: value ignored by Catamount

Catamount Standard I/O

- Unbuffered by default -> very slow
- Use setvbuf to buffer stdin, stdout, or stderr
- Example

```
char *buf;
buf = (char *)malloc(1024 char *);
setvbuf(stdout, buf, _IOFBF, 1024);
```

PGI C/C++ Compiler Issues

- PGI 5.2 & and 6.0 object file incompatibilities
- C99 Standard Conformance
  - C++-style comments, use '-B' option
  - Variable length arrays
  - ISO C99 library routines
    - Compiler undefs __USE_ISOC99 macro
    - Catamount glibc and libm has ISO C99 routines
    - Prototypes for routines not included, problem for C++
- PGI profile (-Mprof) does not work on Catamount

C++ Template Instantiation

- PGI 5.2 uses a prelinker to perform template instantiation
- Very problematic in 5.2
  - Need to prelink before archiving object files in a library. Need to alter makefiles, which assume g++ style instantiation
  - The '-g' option was not permitted for building C++ object files for a library. Could not debug code
  - Prelinking requires files to be recompiled, adds to total build time

C++ Template Instantiation

- 5.2 problems (continue)
  - Build process not robust. Required additional supporting files (i.e. *.ti, *.ii) to be maintained. Undefined linking errors were sometime resolved by removing files and performing a clean build
- PGI 6.0 uses gnu-like template instantiation
  - Template instantiated immediately in object file
  - gnu linker discard multiple copies of templates
  - No special options are needed

PGI Compiler Options

- Chapter 2 of PGI User Guide has overview of optimization options
- Options I have seen used for Cray XT3 code:
  - -fastsee
  - -Mnontemporal
  - -Mprefetch=distance:8,nta
  - -Msafeptr
  - -Mipa=fast
  - -Minline=levels:10

PGI Compiler Options

- Options (continued):
  - -Kieee
  - -O3
  - -Minfo
  - -Mneginfo
  - -help
  - -v
  - -tp k8-64, -tp amd64

Future Opportunities

- Large memory support
  - Data >2GB requires –mcmodel=medium option
  - New set of libraries are needed
- Cross Compiler Environment
- Support of additional compilers (Gnu, Pathscale)
- Parallel execution on Linux kernel nodes

Conclusion

- Given current stage of the Cray XT3 life cycle, C/C++ programming environment is very good
- Catamount glibc limitations have not been a major obstacle
- PGI compilers have performed well
  - One exception is PGI 5.2 C++ template instantiation. Fixed in PGI 6.0

Questions? Answers?