XcalableMP
Directive-based language eXtension for Scalable Parallel Programming

Overview

What’s XcalableMP?
XcalableMP(XMP) is a directive-based PGAS language for distributed memory system
Designed by XMP Specification Working Group
Members from academia (U. Tsukuba, U. Tokyo, Kyoto U., and Kyusyu U.), research labs(RIKEN, NIFS, JAXA, and JAMSTEC/ES), and industries(Fujitsu, NEC, Hitachi) in Japan
Omni XMP compiler was developed in "Seamless and Highly-productive Parallel Programming Environment for High performance computing" project funded by MEXT in Japan

Implementation Status
XMP specification ver. 1.1 is available
- Mapping inquiry procedures are expanded
- The specification on coarrays is improved
Omni XMP compiler ver. 0.6 is developed by University of Tsukuba and AICS, japan
Download from http://www.xcalablemp.org
XMP/C and XMP/Fortran Compilers are included
Interface of Scalasca & tlog profiling tools
Supported platforms are Linux cluster, the K computer, Cray XE, XT, and so on

Programming Model

Language Features
- Language extension of C99 and Fortran 95
- SPMD as a basic execution model
- Communication, synchronization, and work-mapping occur when directives are encountered
- All actions are taken by directives for being “easy-to-understand” in performance tuning (different from HPF)

Global-view model
- a[12] is distributed onto 4 nodes
  ```c
  int a[12];
  #pragma xmp nodes p(4)
  #pragma xmp template t(0:11)
  #pragma xmp distribute t(block) onto p
  #pragma xmp align a[i] with t(i)
  #pragma xmp loop on t(i) reduction(+s)
  for(i = 1; i < 10; i++) {
    a[i] = func(i);
    s += a[i];
  }
  ```

Local-view model
- Coarray Fortran like feature
  ```c
  double a[5]:[*]; // Declaration
  b[0:2] = a[3:2]:[2]; // Get Operation
  ```
- Extends C for an array section
  ```c
  array_name[start:length[:step]]:node_number
  ```
  The `array_name[start:length]:node_number` means elements from the array_name[start] to the array_name[start+length-1] located on a node whose number is node_number.

Performance

On the K computer
- SPARC64 VIIIfx 2.0GHz (Single Socket), 8Cores/Socket (128GFlops/Node)
- DDR3 SDRAM 16GB, 64GB/s/Socket
- Torus fusion six-dimensional mesh/torus network, 5GB/s

On HA-PACS
- Xeon E5-2670 2.6GHz (Dual Socket), 8Cores/Socket (332.8GFlops/Node)
- DDR3 SDRAM 128GB, 51.4GB/s/Socket
- Infiniband QDRx2rails
- Fat-tree network, 4GB/s
- Typical linux cluster

For more information, please visit Center for Computational Sciences, University of Tsukuba (#3618)