

# PRIMEHPC FX100

## Tuning Guide

(Former Tuning Tutorial)

FUJITSU LIMITED  
April 2016 Version 2.0

The tutorials consist of the following chapters:

- Chapter 0 Tuning Guide

## Intra-Node

- Chapter 1 Hardware
- Chapter 2 Program Development Environment and Programming Overview
- Chapter 3 Large Page
- Chapter 4 Fortran
- Chapter 5 C/C++
- Chapter 6 PA Event
- Chapter 7 Tuning Tool
- Chapter 8 Intra-Node Tuning

## Inter-Node

- Chapter 9 MPI and Inter-node Tuning

# Purpose and Aim of the Tutorials

- The tutorials explain, in an easy-to-understand manner, the skills required for efficiently exploiting the performance of the FX100.
- The purpose of this document is to provide a useful reference on tuning while exhibiting the real capabilities of user applications.
- The tutorials consist of the following parts:
  - Overview and general description (Chapters 1 to 7 and 9)
    - Tutorials present prerequisite hardware and software knowledge for tuning as well as the prerequisite functions that form a foundation for tuning.
    - Tutorials explain how to compile and execute user applications, and how to use tools.
    - Beginners of performance tuning are recommended to read the Chapter 1 through 7. Also, MPI users are to read outline and explanation part of the Chapter 9.
  - Practices and case examples (Chapter 8)
    - Tutorials explain how to collect and effectively use the information (PA information, sampler information, etc.) required for analysis of problems that have occurred.
    - Tutorials covering response methods explain specific improvement cases so that users can take appropriate measures against problems that have occurred.
    - Middle and high level programmers of performance tuning are recommended to read the Chapter 8 and Example part of the Chapter 9.

## Note:

In this tutorial, a necessary skill is extracted from each manual and explained collectively. There is information that has not been described to the tutorial like the use of functions. Please check the manual about them. This tuning guide was written for Technical Computing Suite V2.0L20.

## ■ Chapter 0 Tuning Guide

- Introduction
- Purpose and Aim of the Tutorials
- Whole Contents

## ■ Chapter 1 Hardware

- Hardware
  - Hardware Configuration
  - SPARC64™ Xifx Chip Overview
  - Tofu Interconnect 2
  - SPARC64™ Xifx Specifications
- Hardware Features
  - Assistant Core
  - HPC-ACE2
  - VISIMPACT
  - Register Extensions
  - Mathematical Function Auxiliary Instructions
  - Sector Cache
- SIMD
  - SIMD Specifications
  - SIMD Technologies

- Chapter 2 Program Development Environment and Programming Overview
  - Software System
  - Parallel Programming Model
  - Program Development Environment
    - Language Specifications
    - Optimization Function
    - MPI
    - Mathematical Libraries
    - Programming Support Function
    - Time measuring function
  - VISIMPACT
    - VISIMPACT Mechanism
    - Automatic Parallelization Function
  - NUMA-aware Execution Method
    - What Is NUMA?
    - First Touch
    - Recommended Execution Method

- Chapter 3 Large Page
  - What Are Large Pages?
    - Functions of Large Pages
    - Purpose of Large Pages
    - Memory Address Conversion and TLB
    - What Is the TLB? (Details)
    - TLB Configuration
    - Default Large Page Size
    - Aspects of Each Page Size
    - Page Size Selection Criteria
  - Functions of Large Pages
    - Ipgparm Command
    - Demand Paging
    - Different Areas Allocated for Different Variable Types
    - Arena Process
    - Thread Heap
    - Environment Variables Used for Tuning
  - Evaluation Functions for Large Pages
    - List of Evaluation Functions for Large Pages
    - Large Page Memory Usage Information
    - Large Page Statistics

## ■ Chapter 4 Fortran

### ■ How to Compile/Execute a Program

- How to Compile a Program
- How to Execute a Program

### ■ Aspects of Compile Information

- Aspects of Compile Information
- Operation Confirmation for Optimization

### ■ Typical Compiler Optimizations

- SIMD
  - Basic Principles of SIMD Optimization
  - SIMD Optimization of Sequential Data
  - Use of Dual Single Precision (8 SIMD)
  - SIMD Optimization of Stride and Indirect Access
  - SIMD Optimization of Complex Types
  - Masked SIMD Optimization
  - Confirmation of SIMD Optimization
- Software Pipelining
  - Basic Principles of Software Pipelining
  - Confirmation of Software Pipelining

### • Loop Optimization

- Loop Optimization
- Loop Interchange
- Loop Fission
- Loop Fusion
- Loop Unrolling
- Loop Collapse

### • Automatic Parallelization

- Simple Loop Slicing
- Loop Slicing through Reduction
- Determination of Whether Parallelization Is Possible
- Confirmation of Automatic Parallelization
- Pipeline Parallel Processing

### ■ Recommended Options (Effects and Impact)

- Recommended Options
- Options with an Impact on Operation Results and Execution
- Other Compile Options
- Optimization Options

### ■ Debug Functions

- Overview of Debug Functions
- Overview of the -Nquickdbg Option
- Overview of the -Haefosux Option
- Overview of the Hook Function

- Chapter 5 C/C++
  - How to Compile/Execute a Program
    - How to Compile a Program
    - How to Execute a Program
  - Supported Language Specifications
  - Aspects of Compilation information
    - Aspects of Compile Information
    - Operation Confirmation for Optimization
  - Typical Compiler Optimizations
    - SIMD
      - Basic Principles of SIMD Optimization
      - SIMD Optimization of Contiguous Data
      - Use of Single Precision and Double Width (8 SIMD)
      - SIMD Optimization of Stride and Indirect Access
      - SIMD Optimization of Complex Types
      - Masked SIMD Optimization
      - Confirmation of SIMD Optimization
    - Software Pipelining
      - Basic Principles of Software Pipelining
      - Confirmation of Software Pipelining
    - Loop Optimization
      - Loop Optimization
      - Loop Interchange
      - Loop Fission
      - Loop Fusion
      - Loop Unrolling
      - Loop Collapse
    - Automatic Parallelization
      - Simple Loop Slicing
      - Loop Slicing through Reduction
      - Determination of Whether Parallelization Is Possible
      - Confirmation of Automatic Parallelization
      - Pipeline Parallel Processing
  - Recommended Options (Effects and Impact)
    - Recommended Options
    - Options with an Impact on Operation Results and Execution
    - Other Compile Options
    - Optimization Options
  - Debug Functions
    - Intrinsic Debug Function
    - Debug Function for Abnormal End Times
    - Hook Function

- Chapter 6 PA Event
  - What Is Cycle Accounting?
  - Within a Node: CPU Tuning Techniques (Summary)
  - Precision PA Visibility Function (Excel Format)
  - PA Graph Components
    - Commit Events
    - Stall Events
  - PA Information
    - PA (Performance Analysis) Information Report
    - Performance Information
    - Memory Throughput Information and Cache Throughput Information
    - SIMD Instruction Information
    - Cache Miss Information
    - Instruction count information
    - Load Balance Information
    - XFILL Flag



- Chapter 7 Tuning Tool
  - Tuning Workflow
  - Profiler Operation Flow
  - Profiler Overview
    - Features of Each Component
    - Information Retrieval and Analysis Procedure (Instant Profiler)
    - Instant Profiler Use Example
    - Retrievable Information (Instant Profiler)
    - Information Retrieval and Analysis Procedure (Advanced Profiler)
    - Advanced Profiler Use Example
    - Retrievable Information (Advanced Profiler)
  - Profiler Visualization
    - Summary View
    - Topology View
      - (1) Profiler Information List
      - (2) Overall Graph
      - (3) Color Histogram
      - (4) Enlarged Information Panel
      - (5) Display Format Switch Tabs
    - Source View
      - Source Information Screen
      - (A) Line Information Display Area
      - (B) Source Code Area
      - (C) Jump Map
    - Call Graph
  - Profiler Use Examples
    - General Use Examples
      - Checking High-cost Parts
      - Checking the Load Balance
      - Checking Cost Distribution
      - Accuracy of sampling (fipp)
    - PA Information Example
    - MPI Information Example
- Tuning Examples
  - Tuning Procedure
  - Sequential Tuning
    - Identifying a High-cost Loop
    - Detailed Analysis of a High-cost Loop
    - Tuning Work
    - Verifying Tuning Results
  - MPI Tuning
    - Cost Information Collection and Simple Analysis
    - Detailed Analysis of MPI Information
    - Tuning Work
    - Verifying Tuning Results
- Precision PA Visibility Function (Excel Format)
  - Data Collection (Execution)
    - Measurement Section Specification
    - Compilation
    - Information Collection by fapp
    - Information Collection Command Example
  - Data Analysis
    - Information Output by fappxx
    - Information Output Command Example
    - Operations in Excel
  - Aspects of Excel Sheets
    - Tabulated Result Example

- Chapter 8 Tuning within a Node
    - CPU Tuning
      - What Is CPU Tuning (Tuning within a Node)?
      - Positioning of CPU Tuning
    - How to Effectively Use PA Information and Tuning Flows
      - How to Effectively Use PA Information
      - Tuning Flow
        - 1. Hot Spot Detection
        - 2. PA Information Collection
        - 3. Breakdown to the Level of Hot Spots
        - 4. Analysis and Diagnosis: Hot Spot (1)
        - 5. Measures and Effects: Hot Spot (2)
    - Analysis and Tuning of Each Hot Spot
      - (Duplicate) Hot Spot (1): IF Construct in the Innermost Loop (Analysis and Diagnosis)
      - (Duplicate) Hot Spot (1): IF Construct in the Innermost Loop (Measures and Effects)
      - Hot Spot (2): Stride Access (Analysis and Diagnosis)
      - Hot Spot (2): Stride Access (Measures and Effects)
      - Hot Spot (3): Ideal Operation (Analysis and Diagnosis)
      - Hot Spot (4): Data Dependency (Analysis and Diagnosis)
      - Entire Evaluation Region (Measures and Effects)
      - Summary
    - Navigation from PA Information to Tuning Techniques
      - Tuning Map
      - Tuning Technique List
    - Scalar Tuning
      - Improvement in Data Access Wait (Improvement in Thrashing)
        - Improvement in Cache Thrashing
          - What Is Cache Thrashing?
          - Tuning Approach to Cache Thrashing (Basics)
            - + Tuning Approach (Basics)
            - + Array Merging
            - + Dimensional Displacement of an Array
            - + Loop Fission
            - + Padding
          - Tuning Approach to Cache Thrashing (Application)
            - + Tuning Approach (Application)
- Improvement in TLB Thrashing
- Improvement in Data Access Wait (Increase in Data Locality)
  - What Is Data Locality?
  - Strip Mining
  - Loop Blocking
  - Sector Cache
  - Loop Interchange
  - Loop Fusion
  - Array Merging (Indirect Access)
- Improvement in Data Access Wait (Latency Concealment)
  - What Is Latency Concealment?
  - Indirect Access Prefetching
  - Prefetching for an Outer Loop
- Improvement in Data Access Wait (Reduced Amount of Access)
  - Memory Throughput and Amount of Memory Access
  - High-speed Store (XFILL)
- Improvement in Operation Wait (Instruction Scheduling Improvement)
  - Factors Hindering Instruction Scheduling
  - Hindering Factor: Improvement of a Loop Containing an IF Statement
    - SIMD Extensions with the Mask (Basics)
    - SIMD Extensions with the Mask (Application)

- Hindering Factor: Improvement in Data Dependency
  - Loop That Has Data Dependency
  - Loop That Has an Unclear Definition Reference Relationship
  - Loop Containing Pointer Variables
- Hindering Factor: Improvement of a Loop with a Few Iterations
  - Loop with a Few Iterations
  - Specification of an Appropriate Number of Unrollings and Suppression of Software Pipelining
  - Cloning
- Various Optimizations
  - Rerolling
  - Facilitation of SIMD Optimization through Changes to Simple Variables
  - Inline expansion: procedure containing use-association of allocatable variable
- Thread Parallelization Processing Tuning
  - Thread Parallelization Ratio Improvement
    - What Is the Thread Parallelization Ratio?
    - Thread Parallelization Ratio Improvement
      - Loop That Has an Unclear Definition Reference Relationship
      - Loop Containing Pointer Variables
      - Loop That Has Data Dependency
  - Execution Efficiency Improvement of Thread Parallelization Processing
    - Improvement in False Sharing
    - Improvement in Load Imbalance
      - Triangular Loop
      - Loops with Irregular Amount of Calculation
        - + Loop Containing an IF Construct
        - + Loop with an Irregular Amount of Calculation
      - Small Loop Iteration Count of a Parallelized Dimension
        - + Parallelization in an Appropriate Parallelized Dimension
    - Usage Taking SSL2 Library Performance into Account (DGEMM)

## ■ Chapter 9 MPI and Inter-node Tuning

- MPI Specifications of the FX100
- Compilation and Execution of an MPI Program
  - Compilation of an MPI program
    - How to compile the program
    - MPI program compile options
  - Execution of an MPI program
    - How to execute the program
    - mpiexec options (global options)
    - mpiexec options (local options)
    - MCA parameters
  - SPMD model and MPMD model
- Performance Improvement
  - Communication contention
  - Eager and Rendezvous
  - Cost of reception wait
  - Acceleration of collective communication
    - Tofu Barrier Communication
    - Tofu-dedicated Algorithms for Collective Communication
  - MPI statistical information

## ■ Tuning Examples

- Problem detection guidelines for communication times
- Overview of tuning examples
  - Effective nonblocking communication using four TNIs
  - Use of trunking
  - Examples of overlapping operations and communications
  - Improvement through the data types used
  - Communication Using the Basic/Derived Datatype
  - Use of assistant cores
  - Example of differences in performance with a specified shape

## ■ Fujitsu Extended Specifications


- Rank Query interface
- Extended RDMA interface

## ■ Troubleshooting

- Debug library
- Memory area-related MPI errors
- Hardware Queue Overflow
- Debug options
  - Deadlock detection function
  - Communication buffer write damage detection function

# Revision History

Version	Date	Revised section	Details
2.0	April 25, 2016	-	- First published



**FUJITSU**

shaping tomorrow with you