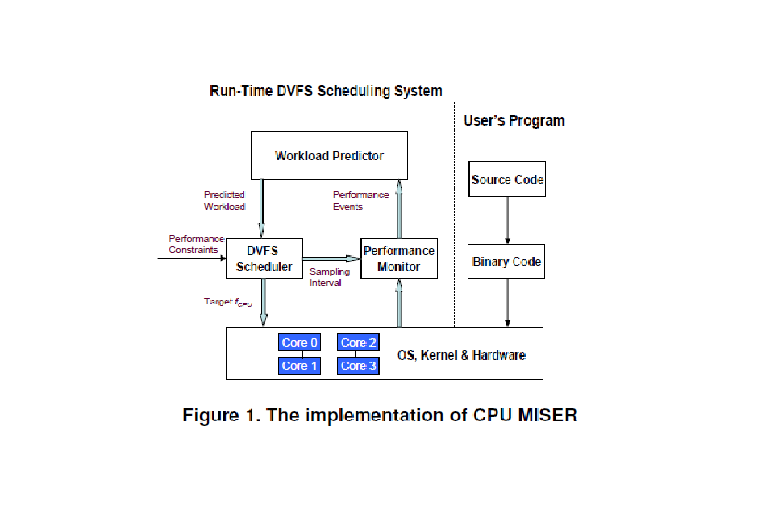
CPU MISER- User's Guide

|  |  |
| --- | --- |
| Contents  Contents  Introduction  About This Manual  Manual Structure  Setup & Operation Guide (Quick Start Guide)  Software User's Guide (this manual) | |
| – INSTALLATION  1.Starting the of Ubuntu:  2.Terminal  3.Login as root if the user isn't  4.Installation of CPU FREQUTILS  5. Checking of the current frequency.  6.Installation of Model Specific Registers  7.Installation of C compilers  8. Installation of Openmpi  8.1 Openmpi is already installed  8.2Checking if Openmpi has been success fully installed  9. Installation of the benchmark  9.1Buliding of benchmarks  9.2 Check if the benchmarks are build.  10.Compilation of the code | ..3 |
| **– EXECUTION AND IMPLEMENTATION OF CPU-MISER** | ..3 |
|  |  |
| 1.Running of the benchmark.  2.Run the benchmark along with the CPU\_MISER C code  2.1 To avoid the FLUCTUATION OF POWER CONSUMPTION  2.2Checking of Running of processes |  |
|  |  |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | |
|  |  |
|  |  |
|  |  |
|  | |
|  | ..10 |
| – SAFETY AND SECURITY | ..10 |
|  | |
|  |  |

1.Introduction

The purpose of this document is to give a detailed description of the implementation of the “CPU-MISER” (**CPU M**anagement **I**nfrastructure for **E**nergy **R**eduction), which is an Operating system Linux Scheduling Software. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications.

This paper presents energy saving characteristics of the CPU MISER when used with application benchmarks. The dynamic frequency scaling optimization is applied to these implementations and serves as verification for the proposed general energy savings model. The developed model provides the minimum of on the compute node energy consumption under a given performance loss tolerance for various processor frequencies.



1.1About This Manual

This manual provides an overview on CPU-MISER and is a step to step guide of how to install the contents of the entire package(along with dependencies).

1.1.1Manual Structure

The following manuals are provided to help you use and install CPU-MISER correctly, and understand all features. Make sure to read them before implementing it.

1.1.2Setup & Operation Guide (Quick Start Guide)

Read the Setup and Operations Guide . The guide includes information you should read before implementing, such as general pre-knowledge for use, installation and setting up of the tools, and how to reduce the frequency with the use of the tools installed.

Also included is information describing how to install the accompanying software to PC, and maintenance/troubleshooting.

1.2.1Software User's Guide (this manual)

This manual describes how to install the tools and packages on UBUNTU a LINUX operating system in-order to implement CPU-MISER.

How to install the tools

This section explains the procedure for installing various kinds of tools and packages on Ubuntu.

1.Starting the of Ubuntu:

press the power button of the machine on which Ubuntu is pre-installed.

login as root in-order to install the tools and packages.

2.Terminal

Select the option of Dash Home from the start menu. click on it n type terminal in the search option

**pic**

3.login as root if the user isnt

type command sudo su press enter

enter root password. press enter. the user is now logged in as root

**pic**

**4.** Installation of CPU FREQUTILS

The following tool changes frequency of the of CPU according to the needs of the user.

In the terminal as a root user type

apt-get cpufrequtils

**pic**

the current results shows that it has been installed successfully.

5. Checking of the current frequency.

Check the number of processors and their frequency of the machine in which Ubuntu is installed

\*\*\*check command – cpufreq-info\*\*\*

6.Installation of Model Specific Registers

MSR(Model Specific Registers) reads the parameters – memory access, instructions retired

apt-get msr-tools

7.Installation of C compilers

Three different C language compilers are being installed inorder to execute the code

apt-get install gcc

**pic**

apt-get install g++

**pic**

apt-get install gfortan

**pic**

**8. Installation of Openmpi**

Openmpi is a high performance and widely portable implementation of the **Message Passing Interface (MPI)** standard. The MSR layer is interacting with the processor registers you are changing frequency and reading the parameters through the MSR(model specific registers).

This code will run only on Linux since it provides the functionality for MSR and the user benchmarks are also operated on the same.

**MPI is the library which will enable the becnchmark**

<http://www.open-mpi.org/software/ompi/v1.6/> :// link to download mpich

Open MPI download //interface that helps to run benchmark( NAS is written in openMPI language/interface)

<http://www.open-mpi.org/software/ompi/v1.6/downloads/openmpi-1.6.5.tar.gz>

Extract the files from the downloads by pasting it to the folder where you wish to save it.

Change directory to the folder where it has been extracted.

run command-> confige file - ./configure (./configure cc=gcc f77=gfortran, if unable to install)

command-> make //binary produce - modules

make install //command used to install/load the modules on the machine.

8.1 If Openmpi is already installed command written below cleans the pre-installed and re-installs the modules

single line command – make clean && make && make install

8.2Checking if Openmpi has been successfully installed and gives the path where it has been installed

check command – which mpirun

9. Installation of the benchmark.

The NAS Parallel Benchmarks (NPB) are a small set of programs designed to help evaluate the performance of parallel supercomputers.

NAS Parallel benchmark via site

<http://www.nas.nasa.gov/publications/sw_instructions.html>

<https://www.nas.nasa.gov/cgi-bin/software/start>

After downloading the benchmark:

Extract the files in a folder

Change directory to NPB3.3/NPB3.3-MPI

run command=>

rename make.def.template to make.def

add the following changes in make.def

\\ask bhaiya the use of these commands and the changes

echo $LD\_LIBRARY\_PATH

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/lib

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/lib

echo $LD\_LIBRARY\_PATH

\*\*:/usr/local/lib:/usr/lib \*\*

\*\*\*\*\*\*modprobe msr // gets a positive value for of the miser code written in C language

9.1 Buliding of benchmarks.

Command=>

make NPROCS=4 (BT) CLASS=C

make NPROCS=2 (CG) CLASS=C

make NPROCS=2 (EP) CLASS=C

make NPROCS=4 (FT) CLASS=C

make NPROCS=2 (IS) CLASS=C

make NPROCS=2 (LU) CLASS=C

make NPROCS=2 (MG) CLASS=C

make NPROCS=2 (SP) CLASS=C

9.2 Check if the benchmarks are build.

command=> cd bin (changes the directory to bin folder)

command=> ls (list the files present in it)

OUTPUT : **pic**

bt.C.4 cg.C.2 ep.C.2 ft.C.4 is.C.2 lu.C.2 mg.C.2 sp.C.4

10. Compilation of the code

C code written for the CPU-MISER is to be compiled

Command=>

gcc miser.**c**

**HOW TO RUN THE CODE AND THE BENCHMARK**

1.Running of the benchmark.

Command=>

./ep.C.2

or mpirun -np 2 ./cg.C.2

the following command initiates the running of the NAS parallel benchmark

**pic**

Record the power on the KILL A-WATT power meter and the time period from the output.

2.Run the benchmark along with the CPU\_MISER C code

Command=>

./ep.C.2 && ./a.out

or mpirun -np 2 ./cg.C.2 && a.out

(NOTE:When using EP benchmark frequency should be less because its CPU based and has high Off –chip memory otherwise it will lead to performance loss)

2.1 To avoid the FLUCTUATION OF POWER CONSUMPTION

Command=>

echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling\_governor

echo userspace > /sys/devices/system/cpu/cpu1/cpufreq/scaling\_governor

2.2Checking of Running of processes

\*in a new terminal\*

command=>

top → f → j (enter)

We'll see the that the process has been allocated resources