Accelerating 3D Elastic Wave Simulation using PGI Accelerator Directives

Proof-of-Concept Guide

NVIDIA Corporation | 2701 San Tomas Expressway | Santa Clara, CA 95050 | **www.nvidia.com**

Table of Contents

Contents

Step-by-Step Instructions

This section describes how to configure the 3D elastic simulation proof-of-concept and demonstrate GPU acceleration using PGI directives. A package has been developed that contains the original FORTRAN source module, the augmented source module with accelerator directives, and a makefile. These procedures were tested on a system running RedHat Enterprise Linux 64-bit operating environment, with a dual socket Westmere x5670 (2.93 GHz, 12-core) CPUs, and dual Fermi M2090 GPUs inserted in PCIe x16-Gen2 slots.

Retrieve Software

A compressed tar file named *3D-Elastic-POC.tar.gz* is available from NVIDIA or PGI that contains the necessary software for the 3D elastic forward wave modeling test. The only additional software required for this proof-of-concept is the Portland Group PGI CDK Accelerator FORTRAN/C/C++ compilers (V11.8 or later). To obtain a free 30-day trial version of the PGI CDK Accelerator kit, please contact:

Tom Toy
Sales Manager,
The Portland Group
thomas.toy@pgroup.com
Direct: +1 (503) 431-7143
Mobile: +1 (503) 919-9853

and provide the name and email address of the individual in your organization who will be investigating the 3D Elastic Proof-of-Concept using PGI Accelerator tools.

Install Software

Follow the instructions to install the PGI software on your system, and since the 3D elastic simulation uses MPI, be sure to install the PGI version of MPI (MPICH2) during this process. You will need cluster root privileges during the installation, and can install on any node, or multiple nodes in the cluster. For support during this process, please reference the PGI installation support FAQ site, or contact Tom Toy.

Setup User Environment

The next step is to adjust the user environment variables to properly reference the PGI tools. An example for Bourne-shell, or c-shell is included, and assumes the PGI tools were installed in the */opt/pgi*.

For bash environment add the following to $(HOME)/.bashrc

```
export PATH=opt/pgi/linux86-64/11.8/bin:$PATH
export MANPATH=/opt/pgi/linux86-64/11.8/man:$MANPATH
export LM_LICENSE_FILE=$LM_LICENSE_FILE:/opt/pgi/license.dat
export PGI=/opt/pgi
export PGRSHA=ssh
```

For csh environment add the following to $(HOME)/.cshrc

```
set path=(/opt/pgi/linux86-64/11.8/bin $path)
setenv MANPATH /opt/pgi/linux86-64/11.8/man:$MANPATH
setenv LM_LICENSE_FILE $LM_LICENSE_FILE:/opt/pgi/license.dat
setenv PGI /opt/pgi
setenv PGRSHA ssh
```

Build Executables

- tar xvzf /tmp/3D-Elastic-POC.tar.gz

  3D-Elastic-POC/
  3D-Elastic-POC/makefile
  3D-Elastic-POC/run_cpu.sh
  3D-Elastic-POC/seismic_CPML_3D_isotropic_MPI_OpenMP.F90
  3D-Elastic-POC/run_gpu.sh
  3D-Elastic-POC/seismic_CPML_3D_isotropic_MPI_ACC.F90
  3D-Elastic-POC/gold/
  3D-Elastic-POC/gold/energy.dat
- cd 3D-Elastic-POC/
- make

  mpif90 -V11.8 -fast -mp -Minfo -mcmodel=medium -DNXSIZE=101 -DNYSIZE=641 -DNZSLICE=64 -DNUM_PROCS=2 seismic_CPML_3D_isotropic_MPI_OpenMP.F90 -o cpu.out.2.64 >& buildcpu.log

  mpif90 -V11.8 -fast -ta=nvidia -Minfo=accel -DNXSIZE=101 -DNYSIZE=641 -DNZSLICE=64 -DNUM_PROCS=2 seismic_CPML_3D_isotropic_MPI_ACC.F90 -o gpu.out.2.64 >& buildgpu.log

Measure Performance

Since this test combines MPI and OpenMP, multiple tests can be performed to measure scalability across CPU cores and GPUs within a single system or across a cluster of nodes. It is possible to adjust the domain decomposition in the z-dimension of the seismic volume to be processed and measure scalability. The makefile is setup to adjust volume sizes when the executable is built, and a corresponding rule to properly invoke "*mpirun*" for the CPU and GPU experiments. The number of OpenMP threads can be adjusted for scalability tests by modifying the *run_cpu.sh* script and changing the *OMP_NUM_THREADS* environment variable.

```
run_cpu:
	/usr/bin/time mpirun -np $(NP) run_cpu.sh cpu.out.$(NP).$(NZSLICE)

run_gpu:
	/usr/bin/time mpirun -np $(NP) gpu.out.$(NP).$(NZSLIXE)
```

The contents of the run_cpu.sh script:

```
#!/bin/bash

export OMP_NUM_THREADS=6
$1
```

The package contains a run script designed to compare 2 MPI rank using 6 OMP threads on the host server, against 2 MPI rank using 2 Tesla GPUs.

```
#!/bin/bash

make -f makefile run_cpu
make -f makefile run_gpu
```

This framework can be extended to compare scalability when the size of the volume is adjusted by modifying the variables NP and NZSLICE (by powers of two) in the makefile.

References

- **“Seismic Convolutional Perfect Matched Layer (CPML)”, Dimitri Komatitsch and Roland Martin**
  - http://web.univ-pau.fr/~dkomati1/README_seismic_cpml.html

Notice

Trademarks

Macrovision Compliance Statement

Copyright