Manifold User Guide

June 25, 2012

1 Introduction

Manifold is a parallel discrete event simulation framework for simulation of modern multicore computer architectures. The software mainly consists of two layers: a simulation kernel layer, and a model layer that contains a few computer architecture models. In addition, Manifold also provides a few ready-to-use simulator programs. This user guide describes how to obtain Manifold source code, and how to build and run the simulator programs.

2 Directory Structure

The Manifold source code is organized as follows:

```
ROOT
|... code
     |... doc
     |... kernel
     |... models
     |     |... cache
     |     |     |... mcp-cache
     |     |... memory
     |     |     |... CaffDRAM
     |     |... network
     |     |     |... iris
     |     |... processor
     |           |... zesto
     |... simulator
           |... zesto
                 |... QsimClient
                 |... QsimLib
                 |... TraceProc
```

where *ROOT* represents the root of the source tree.

3 Overview

To build and run the simulator programs that are part of the software package, you will need to perform the following steps:

- [Optional] Install required packages.
- [Optional] Download and build QSim.
- Build Manifold libraries.
- Build the simulator program(s).
- Run the simulators.

The simulators can respectively take instructions from three different sources: trace files, QSim library, and QSim server. Depending on which source you use, some of the steps above may be optional.

The following explains each step in detail.

4 Install Required Packages

Before you proceed, you need to install the following required packages.

- mpi: We have tested with openmpi, so it is recommended.
- libconfig++: The simulators require this package.

5 Download and Build QSim

If you choose to use QSim to get instructions, you need to build and install QSim first.

5.1 Download

QSim is available through SVN check out. The latest version at this writing is 0.1.4, at the following address:

- https://sst-simulator.googlecode.com/svn/qsim/tags/qsim-0.1.4

5.2 Build and Installation

Instructions for building and installing QSim can be found in the INSTALL file in the root directory of QSim source code.

In addition to the QSim libaries, you also need to do the following:
- build the QSim server.
- build and install the QSim client library.

All the instructions are in the INSTALL file.

After you are finished, you installation directory should look like the following, assuming QSIM_INSTALL is the root of the installation directory.

```
$ ls <QSIM_INSTALL>/lib
libqemu-qsim.so  libqsim-client.so  libqsim.so

$ ls <QSIM_INSTALL>/include
mgzd.h  qsim-client.h  qsim.h  qsim-load.h  qsim-net.h  qsim-regs.h  qsim-vm.h
```

6 Download and Build Manifold Libraries

There are two ways to download Manifold source code: from the Manifold website or through SVN checkout. Depending on which way is used to obtain the source code, the build process is slightly different.

6.1 Download Manifold source package

Manifold source package is available at the Manifold website:

- http://manifold.gatech.edu/download

After download, follow the following instructions to build the manifold libraries:

1. Untar the source package.

   ```
   $ tar xvfz manifold-0.8.tar.gz
   ```

2. Go to the code subdirectory.

   ```
   $ cd manifold-0.8/code
   ```

3. Run configure and make.

   ```
   $ ./configure [--prefix=INSTALL_PATH]
   $ make
   ```

   The default installation directory is /usr/local/lib. If you want to install in a different location, the path of that location should be passed to configure. In addition, if QSim is installed in a location other than the default, you need to tell configure that location. Options that you can specify for configure are described below.

4. Optionally, install the libraries.

   ```
   $ make install
   ```

6.2 Download Manifold source code through SVN checkout

Manifold source code is available through SVN checkout at the following address:

- https://svn.ece.gatech.edu/repos/Manifold/tags/0.8

To build the un-packaged source code, you need to have autotools package installed on your machine.

1. From the `code` subdirectory, run `autoreconf`.

   ```
   $ cd code
   $ ./autoreconf -si
   ```

   This would create the `configure` script.

2. Run `configure` and `make`.

   ```
   $ ./configure [--prefix=INSTALL_PATH]
   $ make
   ```

3. Optionally, install the libraries.

   ```
   $ make install
   ```

6.3 Configure options

This section describes all of the options you can use when running the `configure` script.

- `--prefix=PREFIX`
  By default, the header files and libraries will be installed in `/usr/local/include` and `/usr/local/lib`, respectively. If you want to install the files somewhere else, you should use this option, and the files will be installed in `PREFIX/include` and `PREFIX/lib`, respectively.

- `--disable-para-sim`

  By default, the Manifold libraries are built for parallel simulation with MPI. If you do not want to use MPI and therefore only run the simulators in sequential mode, you need to specify this option to disable parallel simulation.

- `--enable-kernel-large-data`

  By default, the maximum size of data that are sent between components is 1024 bytes. If this is not big enough, or the maximum size is not known in advance, then this option should be used.

- `--disable-stats`

  By default, the Manifold kernel and computer architecture models all collect statistics at run time. Use this option to disable run-time collection of statistics.

- `KERINC=KERNEL_LOCATION`

  This option specifies where the kernel header files are installed. This is useful when the kernel and the models are built separately.

- `QSIMINC=QSIM_LOCATION`

  This option specifies the location where QSim is installed. By default, QSim is installed under `/usr/local`. This option is useful when QSim is installed in a different location.

7 Build the Simulator Program

The simulator programs are located in `ROOT/code/simulator/zesto`. There are three subdirectories for three types of simulators, based on how they get instructions:

- Programs under `QsimClient` use QSim server to get instructions. To build these simulators, you must first build and install QSim.
- Programs under `QsimLib` use QSim libraries. To build these simulators, you must first build and install QSim.
- Programs under `TraceProc` use trace files in the format of Intel's Pin.

To build the simulators, follow the following steps. Here we use the simulators under `QsimClient` as an example.

1. Go to the simulator source directory.

   ```
   $ cd ROOT/code/simulator/zesto/QsimClient
   ```

2. Run `make`. It is likely that you need to modify the Makefile so the header files and libraries can be found.

   ```
   $ make
   ```

8 Start the Simulator

In each of the three subdirectories, i.e., `QsimClient`, `QsimLib`, and , there is a program called `zesto_llp`. This program simulates the following system model, where each core node has a processor core, a private L1 cache, and a shared L2 slice.
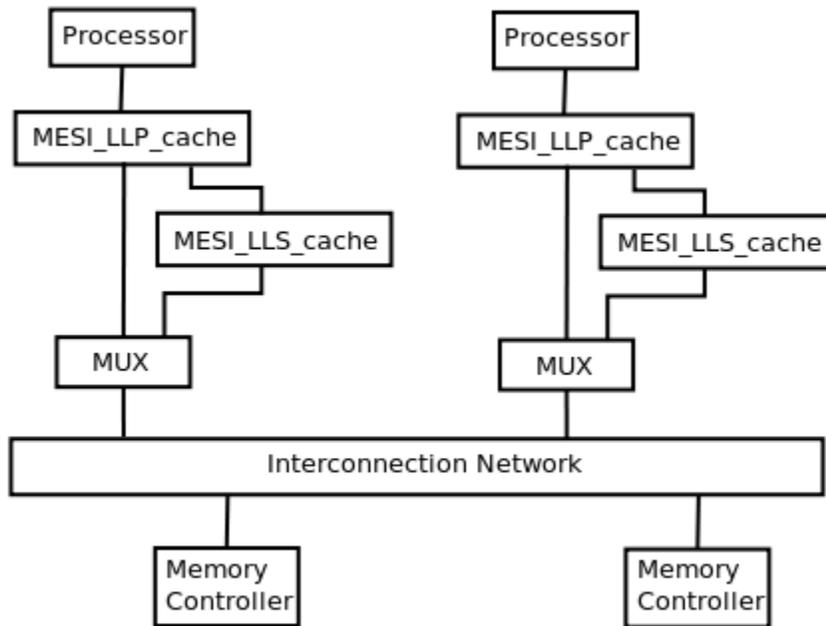
Figure 1 System Model Simulated by zesto_llp.

We describe how to start the simulators in each of the three subdirectories.

8.1 Start the Simulators in QSimClient

These simulators require QSim server be started first.

To start the QSim server, run the following commands:

```
$ cd QSIM_ROOT/remote/server
$ make
$ ./server <port> <state_file> <benchmark>
```

where

- `<port>` is the TCP port number the server uses. Use any number you want.
- `<state_file>` is the state file. QSim is an emulator of a multicore shared-memory machine. The state file is the snapshot of the emulated machine after the OS has booted.
- `<benchmark>` is the tar file containing the application program and its data. See QSim instructions on how to build benchmark tar files.

After the QSim server has started, the simulator can be started.

If QSim is installed in `/usr/local`, do the following,

```
$ cd SIMULATOR_ROOT
$ mpirun -np <NP> <prog> <conf_file> <zesto_conf_file> <server> <port>
```

If Qsim is not installed in `/usr/local`, do the following, assuming QSim installation path is QSIM_INSTALL.

```
$ cd SIMULATOR_ROOT
$ QSIM_PREFIX=<QSIM_INSTALL> LD_LIBRARY_PATH=<QSIM_INSTALL>/lib  mpirun
-np <NP> <prog> <conf_file> <zesto_conf_file> <server> <port>
```

where

- `<NP>` is the number of logical processes (LPs), or MPI ranks. For parallel simulation, currently the simulators support 1, 2, or N+1 LPs, where N is the number of simulated cores.
- `<prog>` is the simulator, including `zesto_llp`, and `zesto_l1l2`.
- `<conf_file>` is the configuration file for the system being simulated. The system configuration is defined in libconfig format.
- `<zesto_conf_file>` is the configuration file for Zesto, our processor model.
- `<server>` is the name or IP address of the QSim server.
- `<port>` is the TCP port number used by the QSim server.

For example:

```
$ mpirun -np 2 zesto_llp conf2x2_iris_torus_llp.cfg 6.cfg localhost 12345
```

8.2 Start the Simulators in QSimLib

Simulators in this subdirectory can only be run with 1 LP, or in sequential mode.

If QSim is installed in `/usr/local`, do the following.

```
$ mpirun -np 1 <prog> <conf_file> <zesto_conf_file> <state_file> <benchmark>
```

If Qsim is not installed in `/usr/local,` do the following, assuming QSim installation path is QSIM_INSTALL.

```
$ QSIM_PREFIX=<QSIM_INSTALL> LD_LIBRARY_PATH=<QSIM_INSTALL>/lib  mpirun
-np 1 <prog> <conf_file> <zesto_conf_file> <state_file> <benchmark>
```

where

- `<prog>` is the simulator, including `zesto_llp`, and `zesto_l1l2`.
- `<conf_file>` is the configuration file for the system being simulated. The system configuration is defined in libconfig format.
- `<zesto_conf_file>` is the configuration file for Zesto, our processor model.
- `<state_file>` is QSim's state file.
- `<benchmark>` is the application tar file.

For example:

```
$ mpirun -np 1 zesto_llp conf4x1_ring_llp.cfg 6.cfg myState_16 myBench.tar
```

8.3 Start the Simulators in TraceProc

These simulators use Pin traces.

```
$ mpirun -np <NP> <prog> <conf_file> <zesto_conf_file> <trace_file_basename>
```

where

- `<NP>` is the number of logical processes (LPs), or MPI ranks. For parallel simulation, currently the simulators support 1, 2, or N+1 LPs, where N is the number of simulated cores.
- `<prog>` is the simulator, including `zesto_llp`, and `zesto_l1l2`.
- `<conf_file>` is the configuration file for the system being simulated. The system configuration is defined in libconfig format.
- `<zesto_conf_file>` is the configuration file for Zesto, our processor model.
- `<trace_file_basename>` is the base name of the trace files. All trace files must have the same base name and be named <base_name>0, <base_name>1, <base_name>2, etc. For example, if the trace files are `myFile0, myFile1`, then the base name is `myFile`.

For example:

```
$ mpirun -np 2 zesto_llp conf2x2_iris_torus_llp.cfg 6.cfg myTrace
```

9 Common Problems

The following is a list of commonly encountered problems, and how to solve them.

- mpirun: command not found
  Solution: If you are using openmpi, install the openmpi-bin package.

- simulation_stop has incorrect type.
  Solution: Open the configuration file, append an 'L' to the number you specify for simulation_stop. For example, if it was "simulation_stop = 1000", change it to "simulation_stop = 1000L".

- cp: cannot stat `./libqemu.so': No such file or directory
  system("cp ./libqemu.so /tmp/qsim_WKLK7m") returned 256.
  Solution: Specify LD_LIBRARY_PATH as shown above.

- cp: cannot stat `/usr/local/lib/libqemu-qsim.so': No such file or directory
  system("cp /usr/local/lib/libqemu-qsim.so /tmp/qsim_eIwV0x") returned 256.
  Solution: Specify QSIM_PREFIX as shown above.