EECS453: Project 1 - Help

Available: Jan 26, 2007

Several people have asked questions about how to download, compile, and install M5. Unfortunately, the web documentation is somewhat out of date, and if you are not a Unix hacker, you might have some trouble figuring out what's gonig on. Hopefully, this document will make the process a little clearer.

For this part of the project, I recommend using the M5 v1.1 web release because the multiproc support is more robust (I think). I have put a link to this on the course webpage.

Note: This should all work on any x86 Linux system. It may or may not work on other systems.

1 Downloading/Unpacking

First, you will need to have fairly recent versions of `gcc/g++`, `python`, and `scons`:

Utility	Version	Download
`gcc/g++`	3.x or greater	`http://www.gnu.org`
`python`	2.3 or greater	`http://www.python.org`
`scons`	0.95 or greater	`http://www.scons.org`

Note: If you have an older version of `python`, you should upgrade `python` before you install `scons`.

Once you have all the necessary utilities, you should download and unpack M5 v1.1 (`http://sourceforge.net/projects/m5sim`). You will need:

- `m5_1.1.tar.gz`

Use the Unix utilities `gunzip` and `tar` to decompress and the main package `m5_1.1` first.:

```
% gzip -d -c m5_1.1.tar.gz | tar -xvf -
```

2 Building

For multiprocessor simulation, you want to use the `ALPHA_SE` (stand alone) mode. This requires building the `ALPHA_SE/m5.opt` binary. This process is relatively straightforward, but takes time (< 30 minutes on a recent Pentium 4). From the point of installation:

```
% cd m5_1.1/m5_1.1/m5/build
% scons ALPHA_SE/m5.opt
```

Note: If you have an SMP machine or an HT enabled Pentium, you can use the `-j N` option in `scons` to perform an *N-way parallel* build.

Hopefully, everything should compile without errors (don't worry about warnings). Also, ignore any complaints about regression tests.

If you see an error involving "MY_SQL", you probably have an old version of `mysql`. You don't need to upgrade – there is a work around: Disable use of `mysql` by inserting the following line (in italics) to the `m5/build/SConstruct` file:

```
# Check for mysql.
mysql_config = WhereIs('mysql_config')
have_mysql = mysql_config != None
have_mysql = False
```

I have put together a basic set of configuration files/benchmarks that you can use for this part of the project. You can download it from the course website (look for `eecs453-m5.tar.gz`). From the point of installation:

```
% cd m5_1.1/m5_1.1/m5/configs
% gzip -d -c <path to eecs453-m5.tar.gz> | tar -xvf -
% cd eecs453
```

This should serve as your working directory for this project. You are now done with the installation and preliminary configuration. Get ready to do the real work. The executable you will use is `m5/build/ALPHA_SE/m5.opt`. You may want to add a symlink. In your work directory:

```
% ln -s ../../build/ALPHA_FS/m5.opt
```

You can now run the simulator (assuming the current the directory is in your path):

```
% m5.opt run.py -ENP=4 -ESYSTEM=Detailed -EL1CACHESIZE=32kB \
                       -EL1BLOCKSIZE=64 - EBENCHMARK=Cholesky
```

The above command line options would have M5 simulate a four cpu system in detailed mode (the mode you should use for this project). In this case, we have specified 32kB L1 instruction and data caches. The blocks will be 64 bytes wide. We run the splash2 benchmark kernel `Cholesky`. By default, the output will be a file called m5stats.txt. You can override this using the `--Statistics.text_file` option.

For example, if you wanted to simulate an 8 node system with 16kB L1 caches, 256 byte blocks, and run the `fft` loop kernel, you could write the output to the file `fft-8-16k-256.txt` with:

```
% m5.opt run.py -ENP=8 -ESYSTEM=Detailed -EL1CACHESIZE=16kB \
                       -EL1BLOCKSIZE=256 -EBENCHMARK=fft \
                       --Statistics.text_file=fft-8-16k-256.txt
```

To make the modifications you need for the project, you should use the command line options. If you are curious about the machine organization, feel free to poke throught he `.py` config files – you might learn something.

You should take a look at the `splash2` directory. It contains the application codes for all of the splash2 benchmarks and loop kernels. Take a look at `splash2/codes/apps` and `splash2/codes/kernels`. You can run any one of them by specifying it with the -EBENCHMARK option as above.

You will probably want to write some scripts to automate data collection – remember you have to get results for many different configurations.