Application Note: AN10125

How to profile an executable on the XMOS simulator

This application note is a short how-to on programming/using the xTIMEcomposer tools. It shows how to profile an executable on the XMOS simulator.

Required tools and libraries

This application note is based on the following components:

- xTIMEcomposer Tools - Version 14.0.0

Required hardware

Programming how-tos are generally not specific to any particular hardware and can usually run on all XMOS devices. See the contents of the note for full details.

1 How to profile an executable on the XMOS simulator

The xTIMEcomposer tools provide support for generating a GNU profiler (gprof) compatible output file. This file contains both the instruction timings (the flat profile) and the call graph information. As an example, compile the following code, ensuring that the generation of debug info (-g) is enabled:

```
int a() {
  int i, j = 0;
  for (i = 0; i < 100; ++i) {
    j += i;
  }
  return j;
}

int b() {
  int i, j = 0;
  for (i = 0; i < 50; ++i) {
    j += i;
  }
  return j;
}

int main() {
  return a() + b();
}
```

2 Profiling from within xTIMEcomposer Studio

Create a new Run configuration using the simulator as the target. To enable profiling output, select the Run configuration and check the *Enable Gprof output* box on the *Simulator* tab. On execution, a number of .gprof files (one per core) will be created at the top level of the project.

On completion, the xTIMEcomposer studio will automatically switch to the profiling perspective, and prompt for both the required *.gprof file and it's corresponding binary. Once selected, the profile information will be displayed, which allows the timing of the program to be analyzed on a source line-by-line basis.

Expand the line in the *gprof* view corresponding to the source file containing the above code. As you can now see, ~50% of the time is spent in function *a*, whereas only ~25% of the time is spent in function *b*. (Note: This is to be expected as *a* contains double the loop iterations.)

3 Profiling from the command line

Run the executable on the simulator using the *-gprof* command line switch:

```
xsim --gprof a.xe
```

This will produce a number of .gprof files, one per core. The *xgprof* tool, supplied in the xTIMEcomposer suite, can be used to analyze the profile from the command line. *xgprof* accepts as input both the generated profile (.gprof) file and the relevant ELF file. The ELF files can be extracted from the XE file as follows:

```
xobjdump --split a.xe
```

xgprof can then be run from the command line in the following way:

```
xgprof image_n0c0.elf tile[0]_core0.gprof
```

The above will generate report for the code running on core 0.