Chapter 3 Extending OpenDwarfs

What is the purpose of the chapter.

The purpose of this chapter is to outline the development of a testbed that can be used to evaluate approaches to scheduling that are proposed in later chapters (or whatever it is).

Key characteristics of the testbed are that it should

1. Support multiple devices
2. Support multiple problem sizes so it can be applied to embedded systems as well as top end scientific processors
3. Spans as many of the Dwarfs as possible.

(I am assuming that justification for these characteristics are given elsewhere).
Item 1 led to the focus on OCL.

In the related work three potential OCL benchmarks suites were considered. Shock, Rodina and OpenDwarfs. Shock is focused on microbenchmarks rather than complete kernels as required here. Rodina while focused on complete kernels is primarily developed as a benchmark suite to compare languages and does not cover all dwarfs. OpenDwarf has the widest coverage and best candidate for use here.

The problems with OpenDwarf (ie what you are going to report as fixed in this chapter) are that the current release

- Includes bunch of architecture specific optimisations
- Has several parameters fixed (eg?)
- Does not currently support multiple problem sizes (at least in terms of easy scaling)

In this chapter we (titles need fine tuning, but these are your major sections)

- The actual OpenDwarf Benchmark
- Methodology for running Opendwarf
- Hardware Selection
- Actual use of the Benchmark
- Discussions, future work and conclusions

==
Now you would start the first section. Within each section there are subsections that highlight the following contributions.

The actual OpenDwarf Benchmark (section 3.1, but you need to clearly draw out the two issues. Maybe two subsections)

- Review the Opendwarf benchmarks for coverage across the 13 dwarfs. (This includes arguing for new FFT/DCRT benchmarks, and moving k-means to replace map-reduce).
- Enable the automatic generation of benchmarks of different sizes.
    - Issues addressed in this section include, i) parameter hardwiring, ii) need to generate input sets for multiple problem sizes, iii) fix issues with code that has been developed on GPU but show memory violations on the CPU

Experimental Setup (section 3.2.1, 3.2.2, 3.2.3)

- The actual hardware you have used. What drove the decision to use these systems (noting it is in part driven by what is available). Bring out the classification of hardware according to your categories of CPU, gaming GPU, Scientific GPU and MIC.
-

Methodology for running Opendwarf (3.2.4, 3.2.5. Try and keep the sections 3.2.4.1-3.2.4.4 as short as possible. What are the key issues. Details could go to an appendix)

- ~~Development of auto-tuner that enables you to select – for example – the optimal number of threads to us within a block.~~
- Methodology for how you go about defining a problem as “tiny”, “small”, “medium” and “large” (e.g. how you use cache size to influence this)
- Details of what the sizes chosen actually were (table 3.3)

Actual use of the Benchmark (3.3)

- Using the benchmarks to assess and compare performance across the hardware systems that you have chosen.
- Primary analysis is for time. Here there are two aspects. One aspect is purely about presenting result X for specific hardware Y. The second aspect is about the decisions of what you present – the recipe. Meaning that if I had a different bunch of systems I should run experiments to get this data and specifically look at these trends – I don't need to present 20 graphs and plots just these key ones. Remember – not all results need to be in the body of the thesis. Pure repetition of results can be placed in an appendix.
- secondary analysis is for energy (minimal because of need for root access of hardware)

Discussions, future work and conclusions

- Don't start this with what is not addressed. Summarize the actual work of the chapter, then the limitations (old hardware, minimal energy stuff – but you have a recipe that can easily be applied), then stuff about the future.

===
With respect to the result graphs. There are multiple ways to approach this.
IF you present crc first – you need to provide a reason for specifically pulling out this benchmark, i.e. this is only case where MIC performs reasonably. (personally I would not present this first – but it is your thesis).

I would first present all dwarfs for one problem size. I would pick medium. I would note in the caption that MIC is only included for CRC because it is extremely slow in all other cases. You will need two pages to include all the dwarfs. The order should be as per table where they are detailed. Ideally you want to have dwarfs with similar performance characteristics together.

From first plot make an argument for selection only one or two dwarfs for which you present graphs as a function of problem size. CRC could be one of these. Graphs no presented in body of the thesis could be provided in the appendix.

Another possibility is to say that you don't need to include all the CPU, or gaming GPU or scientific GPU data , but just one of each of these. You could have a plot where x axis is dwarf – like what you have for energy!!