Introduction to Parallelism and SMP

Andrew Fitz Gibbon

Undergraduate Petascale Education Program

May 24, 2010

Outline

- Parallelism and its Jargon

- Instruction Level Parallelism
  - Looping
  - Pipelining
  - Vectorizing

- SMP and threading

Parallelism!

- Doing multiple things at the same time with multiple execution units in order to solve a problem or parts of a problem.
- “Execution unit” something that does something
- Shared Memory: multi-threading
- Distributed Memory: multi-processing
- Parallelism: Concurrency
- Some of these are used interchangeably
- Instructions, cycles, clocks, pipelines

Instruction Level Parallelism

- The set of techniques for executing multiple things simultaneously inside a CPU core.
  - ILP is lower level than multi-core!

- This helps solve a problem:
  - With tons of circuitry in a core, much of it might be idle. Wasteful!
  - So... use some of it to execute different parts of the program at the same time

DON'T PANIC!

How do we do it?

- Well... we don't really.
  - The compiler and the CPU take care of a lot of it for us.
  - Compilers usually know more than we do.
- But we need to be careful

ILP Flavors

- Superscalar: Doing multiple operations at the same time.
- Pipeline: Doing multiple stages of a complex operation at the same time on different pieces of data. Imagine an assembly line.
- Superpipeline: Multiple pipelines happening simultaneously.
- Vector: Doing the same thing to a bunch of things simultaneously.

– AltiVec: Specific case of Vector processing

Why do we need to be careful?

You won't get much benefit from ILP if:

- Your code is too complicated
- Loops happen in random orders
- Branches, statements depend on too many other parts

Moving up

So if the compiler and CPU do a lot of the heavy work, what can we do?

- Enter SMP, multi-core, and the programming that goes with it

- SMP? Cores? Sockets? Processors? ALUs? TLAs? Dies? Chips? Threads? Processes?

Don't Panic

SMP

Symmetric Multi-Processing

- Multiple cores controlled by one OS, all with access to the same memory

- All memory is not necessarily created equal

  - NUMA: Non-Uniform Memory Access

- “Multi-core” and “SMP” often used interchangeably

Processes and threads

- Process:
  - A construct in the OS/system for a running program
  - Contains everything needed to run a program
  - Code, data, meta-data, control data
- Thread:
  - Similar to Processes; contains code, control-data
  - Single Processes can have multiple threads
  - With multiple threads, all share the process's data