Multicore and Multithreaded Processors

- Why multicore?
- Thread-level parallelism
- Multithreaded cores
- Multiprocessors
- Design issues
- Examples

Why Multicore?

- Why is everything now multicore?
  - This is a fairly new trend
- Reason #1: Running out of ILP that we can exploit
  - Can’t get much better performance out of a single core that’s running a single program at a time
- Reason #2: Power/thermal constraints
  - Even if we wanted to just build fancier single cores at higher clock speeds, we’d run into power and thermal obstacles
- Reason #3: Moore’s Law
  - Lots of transistors → what else are we going to do with them?
  - Historically: use transistors to make more complicated cores with bigger and bigger caches
  - But we just saw that this strategy has run into problems

Readings

- Patterson and Hennessy
  - Chapter 7
  - Some recent research papers!
How do we keep multicores busy?

- Single core processors exploit ILP
- Multicore processors exploit TLP: thread-level parallelism
- What’s a thread?
  - A program can have 1 or more threads of control
  - Each thread has its own PC and other arch registers
  - All threads in a given program share resources (e.g., memory)
- OK, so where do we find more than one thread?
  - Option #1: Multiprogrammed workloads
    - Run multiple single-threaded programs at the same time
  - Option #2: Explicitly multithreaded programs
    - Create a single program that has multiple threads that work together to solve a problem

Parallel Programming

- How do we break up a problem into sub-problems that can be worked on by separate threads?
- ICQ: How would you create a multithreaded program that searches for an item in an array?
- ICQ: How would you create a multithreaded program that sorts a heap?
- Fundamental challenges
  - Breaking up the problem into many reasonably sized tasks
    - What if tasks are too small? Too big? Too few?
  - Minimizing the communication between threads
    - Why?

Writing a Parallel Program

- Compiler can turn sequential code into parallel code
  - Just as soon as the Cubs win the World Series
- Can use an explicitly parallel language or extensions to an existing language
  - High performance Fortran (HPF)
  - Pthreads
  - Message passing interface (MPI)
  - CUDA
  - OpenCL
  - Etc.

Parallel Program Challenges

- Parallel programming is HARD!
  - Why?
- Problem: #cores is increasing, but parallel programming isn’t getting easier → how are we going to use all of these cores???
**HPF Example**

```c
forall(i=1:100, j=1:200){
    MyArray[i,j] = (X[i-1,j] + X[i+1,j]);
}

// “forall” means we can do all i,j combinations in parallel
// i.e., no dependences between these operations
```

**Some Problems Are “Easy” to Parallelize**

- Database management system (DBMS)
- Web search (Google)
- Graphics
- Some scientific workloads (why?)
- Others??