Slide 1

Administrivia

- Reminder: Java program for Homework 1 due today. Plan is to talk next time about results.

Slide 2

Controlling Threads in Java

- Preferred method of controlling one thread from another uses "interrupted" status. (Early version of Java provided other methods, e.g., `stop` — now deprecated.)
- Set status with `interrupt` (instance method).
- Check status with `isInterrupted` (instance method) or `interrupted` (static method), or by catching `InterruptedException` thrown by `wait`, `sleep`, `join`, etc.
- Example — bounded buffer test program (`TestBoundedBuffer.java` on sample programs page).

Slide 3

New Features in Java 5.0 for Multithreading

- Lots of new stuff for concurrent programming Java 5.0 (a.k.a. 1.5). Short examples in later versions of "hello world" program (`Hello3.java`, `Hello4.java`, `Hello5.java` on sample programs page.
- Look at API for `java.util.concurrent` for more ...

Slide 4

Not-So-Simple Point-to-Point Communication in MPI, Again

- For not-too-long messages and when readability is more important than performance, `MPI_Send` and `MPI_Recv` are probably fine.
- If messages are long, however, buffering can be a problem, and can even lead to deadlock. Also, sometimes it's nice to be able to overlap computation and communication.
- Therefore, MPI offers several other kinds of send/receive functions, including:
  - Synchronous (`MPI_Ssend`, `MPI_Recv`) — blocks both sender and receiver until communication can occur.
  - Non-blocking send/receive (`MPI_Isend`, `MPI_Irecv`, `MPI_Wait`) — doesn't block, program must explicitly test/wait.
  - Which is faster/better? probably best to try them and find out. (Sample programs `exchange*`.)

Slide 5

Minute Essay

- This wraps up the quick PAD I-level tour of our three environments. Any questions at this point?