Slide 1

Administrivia

- One purpose of the syllabus is to spell out policies (next slides).
- Most other information will be on the Web, either on my home page (here, office hours) or the course Web page (here).

  A request: If you spot something wrong with course material on the Web, please let me know!

Slide 2

Course FAQ

- "What will my grade be based on?" (See syllabus.)
- "When are the exams?" (See syllabus.)
- "What happens if I can't turn in work on time, or I miss a class?" (See syllabus.)
- "What's your policy on collaboration?" (See syllabus.)

Slide 3

Course FAQ, Continued

- "When is the next homework due?" (See "Lecture topics and assignments" page.)
- "When are your office hours?" (See my home page.)

  Note that part of my job is to answer your questions outside class, so if you need help, please ask! in person or by e-mail or phone.

Slide 4

Course FAQ, Continued

- "What computer(s) can I use to do programming homework?"

  Easiest option may be department's Linux classroom/lab machines. You should have physical access (via your TigerCard) to all the classrooms and labs. You should also be able to log in remotely to any that are booted into Linux, or to a cluster of Linux-only machines in ITS's server room (names `diasnn`, where `nn` ranges from 01 to 14).

Slide 5

"Why Do I Have To Take This Course?"

- We could view computer systems (hardware/software) in terms of layers of abstraction:
  - User interface.
  - Operating system / application programs / tools (compilers, e.g.).
  - High-level programming language / ADTs.
  - Machine language / data representations ("it's all 1s and 0s").
  - Hardware (could break this down, maybe, into logical design and EE).
- A goal of a CS degree program is to "demystify" as many of these as we can.

Slide 6

"Why Do I Have To Take This Course?", Continued

- Relating courses to layers of abstraction:
  - Programming courses — bridge gap between user interface and high-level languages.
  - Operating systems course — bridge gap between user interface / applications programs and hardware.
  - Course on compilers, maybe — bridge gap between application programs and machine language.
  - This course — bridge gaps between application programs and machine language (a bit) and between machine language and hardware.

Slide 7

Course Topics

- Defining and measuring performance.
- Assembly language (MIPS because it's simple and representative).
- Machine language (also MIPS).
- Hardware (at level of AND/OR gates).

Slide 8

Why Study Assembly / Machine Language?

- Understand the general principles of how things work at this level helps you:
  - Write more efficient programs.
  - Understand operating systems (which also helps you write more efficient programs).
  - Generally understand better what's really happening in the machine.
- It might be fun.

Slide 9

Minute Essay

- Tell me about your background: What programming classes have you taken (at Trinity or elsewhere) and what language(s) did you use?
- What are your goals for this course? Anything else you want to tell me?