CSE 378, 06wi – Midterm Topics
Introduction to the MIPS ISA

January 231, 2006

The ISA

- Load-store architecture
- Memory: unit of addressing (byte), unit of transfer (depends on opcode)
- 32, 32-bit GPRs ($0 is special)
- 32-bit instructions
- Instruction set:
    - What kinds of operations are available?
    - Instruction encoding
    - How do you name (encode):
        - Registers
        - Memory
        - Target instructions of branches and jumps

Using the ISA

- What is the role of the compiler?
    - You should be able to "compile" into assembler simple code sequences given in a standard higher-level language (HLL): HLL statements -> instructions; HLL variable declarations -> ?; use of arrays
    - "optimization" – What does it mean? What is an example?
- What is the role of the assembler?
- What is the role of the linker?
- What is the role of the loader?

The Memory Model

- At the hardware level, just a byte addressable store
- Software creates a more useful model:
    - Instruction "segment"
    - Static data area ($gp)
    - Stack ($sp)
    - [Heap]
- How does each piece relate to what you see programming in the higher level language the book uses for examples?

Procedure Call Linkage

- A (software defined) convention
- What does caller do? What does callee do?
- How, and why, does the MIPS linkage convention differ from the one Cebollita uses?
- What features of the ISA are there (just) to support procedure calls?
- (How are arguments passed? Values returned? Local variables created?)