CSE 378, 06wi – Lecture 4 Main Points
Introduction to the MIPS ISA

January 11, 2006

Instruction Encoding (cont.)

- Branches are encoded in I-format: the "immediate" value is a signed offset, in units of instructions (4 bytes) from the PC of the instruction following the branch

- j is encoded as a 6-bit opcode and a 26-bit immediate giving the memory address to set the PC to (the destination instruction of the jump). (How can 26-bits name a 32-bit address?)

Build (Compile/Assemble/Link) and Run Time Concepts

- The executable file must be *relocatable*, because we can't know at build-time what memory addresses it will be allocated when it runs.
- The executable contains: the program instructions; size and initial values of the statically allocated variables; a relocation table
- The loader allocates memory and sets register $gp (the *global pointer*) to point to the beginning of the static data region
- The compiler/assembler have generated instructions to access (static) variables stored in main memory using statically known offsets from $gp
- Because branch addressing is PC-relative, and the offsets from the branch to the target instruction are known at compile/assemble time, they aren't a problem.
- Because the jump instruction encodes an absolute address, it must be *patched* by the loader (once the absolute address of the jump destination is known)

- The value of the label given to a data item in assembly code is an offset (past $gp). The assembler (linker) decides these offsets at build-time.
- The value of a label on an instruction is its offset (past the first line of code).
- These symbols are just shorthand for their values – a more convenient way for humans to understand the code. When the program executes, it's executing machine instructions, which contain only machine addresses.
- Therefore, ideas like "scope" in programming languages are implemented by the language/compiler – they don't mean anything once the program is actually executing. (The compiler ensures that the code it produces obeys the scope rules of the language, but the CPU itself doesn't know about, or care about, those rules.)