CSC 411

Computer Organization (Spring 2022)
Lecture 11: Compiling/Interpreting

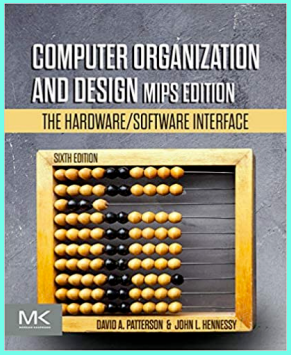
Prof. Marco Alvarez, University of Rhode Island

Disclaimer

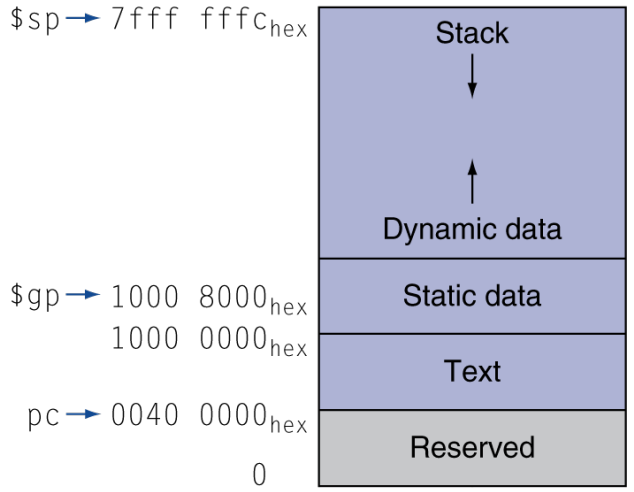
Some of the following slides are adapted from:

Computer Organization and Design (Patterson and Hennessy)

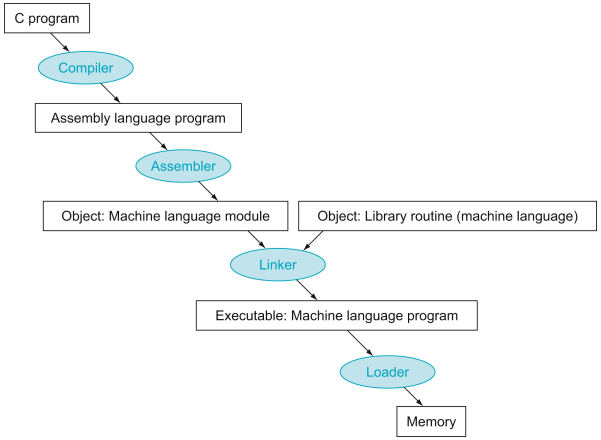
The Hardware/Software Interface



Compiling/interpreting



Translation



To identify the type of file, UNIX follows a suffix convention for files: C source files are named x.c, assembly files are x.s, object files are named x.o, statically linked library routines are x.a, dynamically linked library routes are x.so, and executable files by default are called a.out. MS-DOS uses the suffixes .C, .ASM, .OBJ, .LIB, .DLL, and .EXE to the same effect.

Role of the assembler

- Convert pseudo-instructions into actual hardware instructions
- Convert assembly instructions into machine instructions

Producing an object module

- Assembler (or compiler) translates program into machine instructions
- Provides information for building a complete program from the pieces
  - **Header**: described contents of object module
  - **Text segment**: translated instructions
  - **Static data segment**: data allocated for the life of the program
  - **Relocation info**: for contents that depend on absolute location of loaded program
  - **Symbol table**: global definitions and external refs
  - **Debug info**: for associating with source code

Linking object modules

- Produces an executable image
  - merges segments
  - resolves labels (determine their addresses)
  - patch location-dependent and external refs
- Could leave location dependencies for fixing by a relocating loader
  - with virtual memory, no need to do this
  - program can be loaded into absolute location in virtual memory space

Loading a program

- Load from image file on disk into memory

1. read header to determine segment sizes
2. create virtual address space
3. copy text and initialized data into memory
   - or set page table entries so they can be faulted in
4. set up arguments on stack
5. initialize registers (including sp, fp, gp)
6. jump to startup routine
   - copies arguments to x10, … and calls main
   - when main returns, do exit syscall

Dynamic linking

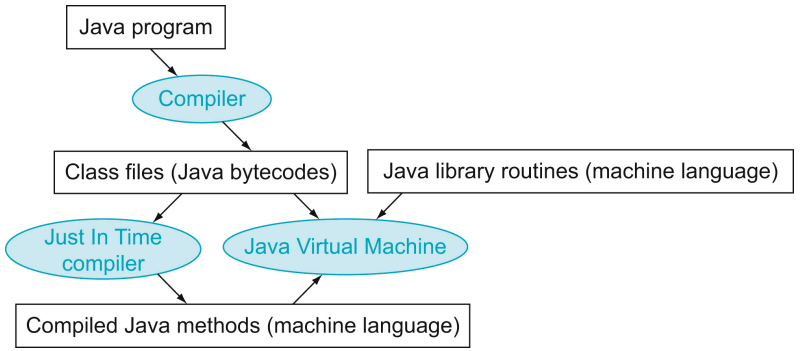
- Only link/load library procedure when it is called
  - requires procedure code to be relocatable
  - avoids image bloat caused by static linking of all (transitively) referenced libraries
  - automatically picks up new library versions

Optimization performance

gcc optimization	Relative performance	Clock cycles (millions)	Instruction count (millions)	CPI
None	1.00	158,615	114,938	1.38
01 (medium)	2.37	66,990	37,470	1.79
02 (full)	2.38	66,521	39,993	1.66
03 (procedure integration)	2.41	65,747	44,993	1.46

Comparing performance, instruction count, and CPI using compiler optimization for Bubble Sort. The programs sorted 100,000 32-bit words with the array initialized to random values. These programs were run on a Pentium 4 with a clock rate of 3.06 GHz and a 533 MHz system bus with 2 GB of PC2100 DDR SDRAM. It used Linux version 2.4.20.

Starting java applications



A Java program is first compiled into a binary version of Java bytecodes, with all addresses defined by the compiler. The Java program is now ready to run on the interpreter, called the Java Virtual Machine (JVM). The JVM links to desired methods in the Java library while the program is running. To achieve greater performance, the JVM can invoke the JIT compiler, which selectively compiles methods into the native machine language of the machine on which it is running.

Interpretation and compilers

Language	Execution method	Optimization	Bubble Sort relative performance	Quicksort relative performance	Speedup Quicksort vs. Bubble Sort
C	Compiler	None	1.00	1.00	2468
	Compiler	01	2.37	1.50	1562
	Compiler	02	2.38	1.50	1555
	Compiler	03	2.41	1.91	1955
Java	Interpreter	–	0.12	0.05	1050
	JIT compiler	–	2.13	0.29	338

Performance of two sort algorithms in C and Java using interpretation and optimizing compilers relative to unoptimized C version. The last column shows the advantage in performance of Quicksort over Bubble Sort for each language and execution option. The JVM is Sun version 1.3.1, and the JIT is Sun Hotspot version 1.3.1.