ECE 471 – Embedded Systems
Lecture 7

Vince Weaver

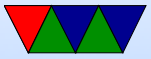
http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

18 September 2019

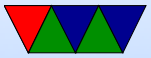
Announcements

- Don't forget HW#2
  - Describe "Something Cool"
  - ANSI escape code
  - `linux_logo`
  - Command line arguments



Loader

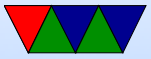
- /lib/ld-linux.so.2
- loads the executable (handles linking in libraries, etc)



Static vs Dynamic Libraries

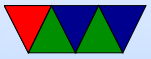
- Static: includes all code in one binary.
  Large binaries, need to recompile to update library code, self-contained, don't have to worry about incompatible updates

- Dynamic: library routines linked at load time.
  Smaller binaries, share code across system, automatically links against newer/bugfixes when system library updated

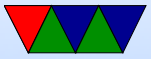


How a Program is Loaded

- Kernel Boots
- `init` started
- `init` calls `fork()` – makes an exact copy of itself
- child calls `exec()` – replaces itself with executable from disk
- Kernel checks if valid ELF. Passes to loader

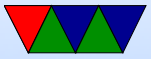


- Loader loads it. Clears out BSS. Sets up stack. Jumps to entry address (specified by executable)
- Program runs until complete.
- Parent process returned to if waiting. Otherwise, init.



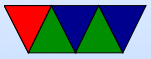
What the OS gives you at start

- Registers
- Instruction pointer at beginning
- Stack
- command line arguments, aux, environment variables
- Large contiguous VM space



Assembly Language: What's it good for?

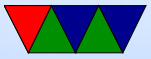
- Understanding your computer at a low-level
- Shown when using a debugger
- It's the eventual target of compilers
- Operating system writers (some things not expressible in C)
- Embedded systems (code density)
- Research. Computer Architecture. Emulators/Simulators.
- Video games (or other perf critical routines, glibc, kernel, etc.)



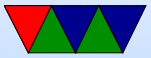
ARM32 Architecture

Note, ARM64 (AARCH64) is very different

- 32-bit
- Load/Store
- Can be Big-Endian or Little-Endian (usually little)
- Fixed instruction width (32-bit, 16-bit THUMB) (Thumb2 is variable)
- arm32 opcodes typically take three arguments (Destination, Source, Source)
- Cannot access unaligned memory (optional newer chips)



- Status flag (many instructions can optionally set)
- Conditional execution
- Complicated addressing modes
- Many features optional (FPU [except in newer], PMU, Vector instructions, Java instructions, divide, etc.)



Registers

- Has 16 GP registers (more available in supervisor mode)
- r0 - r12 are general purpose
- r11 is sometimes the frame pointer (fp) [iOS uses r7]
- r13 is stack pointer (sp)
- r14 is link register (lr)
- r15 is program counter (pc)
  reading r15 usually gives PC+8
- 1 status register (more in system mode).
  NZCVQ (Negative, Zero, Carry, oVerflow, Saturate)