The Tour

16 July 2018 21:05

- The *hello* program
  - The compilation system

Suffix	Program	Phase	
`hello.c`	Source program (text)	-> preprocessor (cpp)	
`hello.i`	Modified source program (text)	-> compiler (cc1)	
`hello.s`	Assembly program (text)	-> assembler (as)	-S …c
`hello.o` & `*.o` in standard C library	Relocatable object programs (binary)	-> linker (ld)	-c …c
`hello`	Executable object program (binary)		-o …c
		disassembler	-d …o

  - In terminal

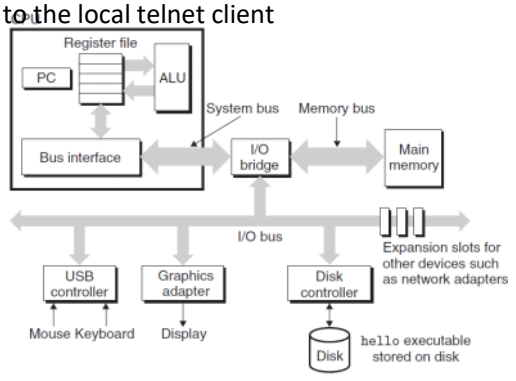
compile `.c` to its executable	`$ make hello.c`
Run	`$ ./hello`

  - Running the `hello` program
    - The shell program reads each characters we typed into a register, then stores it in memory.
    - After hitting `Enter`, the shell then loads the executable `hello` file - copies the code and data in hello object file from disk to main memory using *direct memory access* (DMA).
    - The processor executes the machine-language instructions in main routine - copy string from memory to the register file then to the display devices.

  - Running on a remote machine
    - Use a telnet client running on the local machine to connect to a telnet *server* on a remote machine.
    - Five basic steps:
      1. Type `hello` in the local shell
      2. Local telnet client sends `hello` string to the remote telnet server
      3. Server sends `hello` string to the sell, which runs the `hello` program and passes the output to the telnet server.
      4. The remote telnet server sends "hello, world\n" string to the local telnet client
      5. The client prints the string on display

- Hardware organization of a system
  - Shorts

CPU	Central Processing Unit
ALU	Arthemtic/Logic Unit
PC	Program Counter
USB	Universal Serial Bus



  - Component
    - Buses: a collection of electrical conduits, runing throughtout the system.
      - Designed to transfer fixed-sized chunks of bytes (know as *words*).
      - The number of bytes in a word (the *word size*) is a fundamental system parameter
      - Word sizes of 4 bytes (32 bits) or of 8 bytes (64 bits)

- Processor (CPU): the engine that interprets (or *excutes*) instructions stored in main memory.
  - Each CPU has a specific set of instructions that it can execute.
  - A processor *appears to* operate according to a very simple instruction executing model, defined by its *instruction set architecture*.
  - The speed of a CPU is determined by the *clock cycle*, which is the amount of time between two pulses of an *oscillator*.
  - *Superscalar* processors: the processors that can sustain execution rates faster than one instruction per cycle.
  - CPUs contain some *registers* inside to hold key variables and temporary results.
    - Special registers that visible to the programmer:
      - *Program Counter* (PC): contains the memory address of the next instruction to be fetched, and updated to its successor after the instruction has been fatched.
      - *Stack pointer*: point to the top of the current stack in memory.
      - *Program Status Word* (PSW): contains the condition code bits, which are set by comparison instructions, the CPU priority, the mode (user or kernel), and various other control bits. (it is important for system calls and I/O)
  - a CPU can execute more than one instruction at the same time.
    - *Pipeline* with three stages: separate fetch, decode and execute units. (can be longer)
    - *Superscalar* CPU: there is a holding buffer to store instructions temporarily which are waiting for an available execution unit.
  - The register file: a small storage device that consists of a collection of word-sized registers, each with its own unique name.
  - The ALU computes new data and address values.
- Input/output (I/O) Devices: the system's connection to the external world.
  - Each I/O devices is connected to the I/O bus by either *controller* or an *adapter*.

Controller	Chip sets in the device itselt or on the system's main printed circuit board (*motherboard*)
Adapter	A card that plugs into a slot on the motherboard.

- Main Memory: a temporary storage device that holds both a program and the data it manipulates while the processor is executing the program
  - Physically, it consists of a collection of *dynamic random access memory* (DRAM) chips.
  - Logically, it is organized as a linear array of bytes that own unique address (array index) staring at zero.

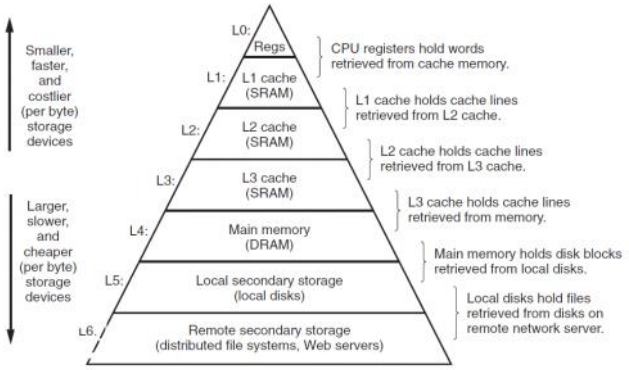
- Caches Matter
  - *Cache memories* (or simply caches): smaller faster storage devices that serve as temporary staging areas for information that the processor is likely to need in the near future, to deal with the *processor-memory gap* (the processor can read data from the register file much faster than from memory).

  - Main memory is divided up into *cache lines*, typically 64 bytes, with addresses 0-63 in cache line 0 and so on.
    - The most heavily used cache lines are kept in a high-speed cache located inside or very close to the CPU.
    - The time the program needs to read a memory word, the cache hardware checks if the line needed is in the cache. If it is, called *a cache hit*, so no memory request is sent over the bus to the main memory.

  - Tow or more cache level:
    - L1 cache:
      - always inside the CPU.
      - Can be accessed nearly as far as the register file.
      - usually feeds decoded instructions in the CPU's execution engine.
      - 16 KB each typically.
    - L2 cache:
      - Holds several megabytes of recently used memory words.
      - On multicore chips, the L2 can be shared by all the cores (Intel multicore chips), or each core has its own L2 cache (AMD).

        - L1 and L2 caches are implemented with a hardware tech known as *static random access memory* (SRAM).

    - *Memory hierarchy*:
        - Main idea: storage at one level serves as a cache for storage at the next level
        -



An example of a memory hierarchy.

- Network
    - A system can be treated as an isolated collection of hardware and software. Modern systems are often linked to other systems by networks. The network can be viewed as just another I/O device.
    - *Internet* is a kind of *global network*, and copying information over a network has become more important.

- Concurrency and Parallelism of Processor
    -

Concurrency	refer to the general concept of a system with multiple, simultaneous activities.
Parallelism	refer to the use of concurrency to make a system run faster.

    - Thread-level concurrency
        - *Multi-core* processors
            - Have several CPUs (or cores) integrated onto a single integrated-circuit chip.
            - Intel Core i7 processor:
                - The microprocessor chip has four CPU cores
                - Each CPU own its L1 and L2 caches
                - sharing the higher levels of cache L3 as well as the interface to main memory.
        - *Hyperthreading* (also called *simultaneous multi-threading*)
            - A tech that allows a single CPU to execute multiple flows of control.
            - It involves having copies of the CPU hardware

multiple copies of some of hardware	such as program conters and register files
only single copies of other parts	such as the units that perform floating-point arithmetic

            - Around 20,000 clock cycles to shift between different threads.
            - Inter Core i7 processor:
                - Each core executing two threads
                - A four-core system
                - Execute eight threads in parallel
    - Instruction-level parallelism
        - *Instruction-level parallelism*: modern processors can execute multiple instructions at one time.
        - *Pipelining*: where the actions required to execute an instruction are partitioned into different steps and the processor hardware is organized as a series of stages, each performing one of these steps.
        - Superscalar operation that provided by superscalar processor.
    - Single-instruction, multiple-data (SIMD) parallelism
        - SIMD parallelism: a mode such that at the lowest level modern processor have special hardware that

allows a single instruction to cause multiple operations to be performed in parallel.