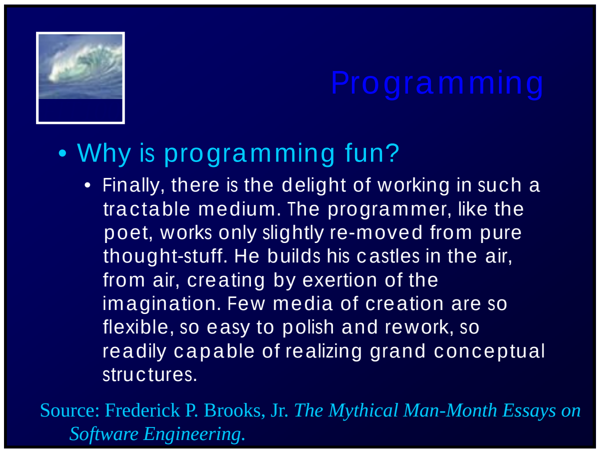
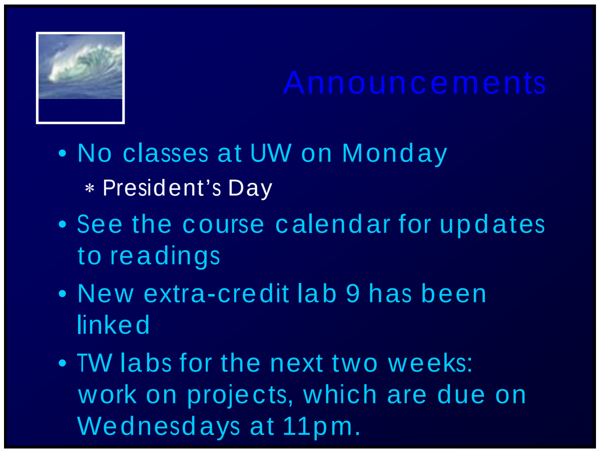


Programming

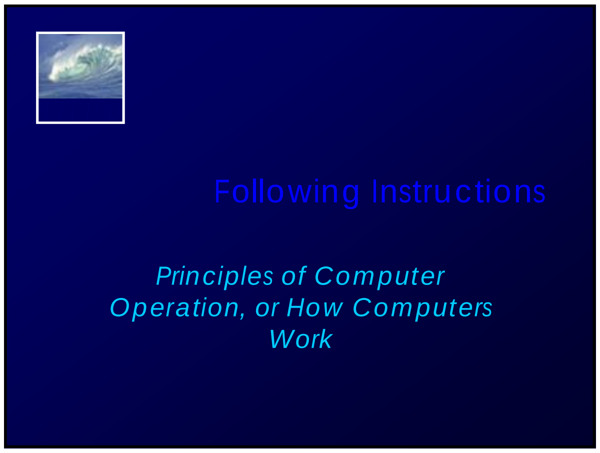
- Why is programming fun?
  - Finally, there is the delight of working in such a tractable medium. The programmer, like the poet, works only slightly re-moved from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of the imagination. Few media of creation are so flexible, so easy to polish and rework, so readily capable of realizing grand conceptual structures.

Source: Frederick P. Brooks, Jr. *The Mythical Man-Month Essays on Software Engineering.*



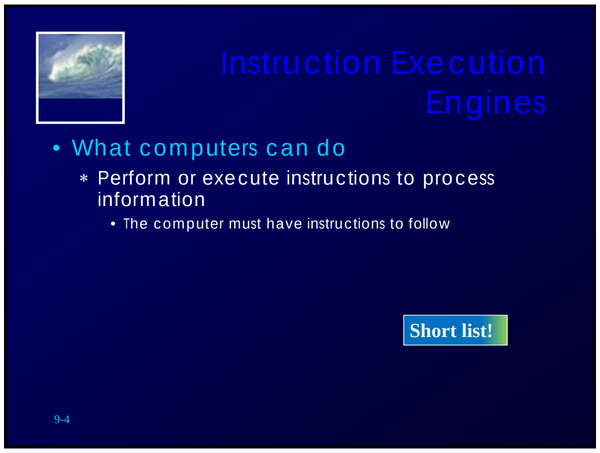
Announcements

- No classes at UW on Monday
  - President's Day
- See the course calendar for updates to readings
- New extra-credit lab 9 has been linked
- TW labs for the next two weeks: work on projects, which are due on Wednesdays at 11pm.



Following Instructions

Principles of Computer Operation, or How Computers Work

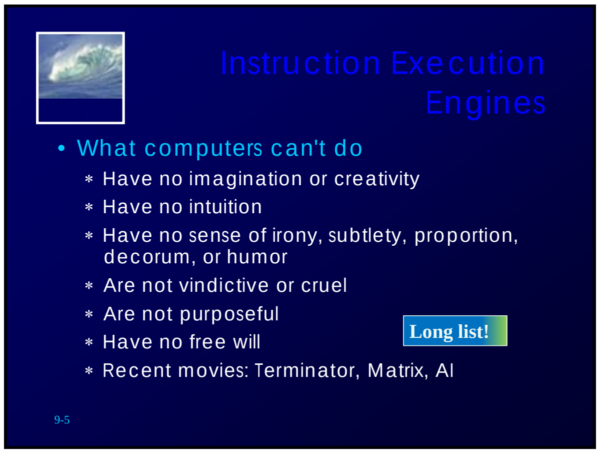


Instruction Execution Engines

- What computers can do
  - Perform or execute instructions to process information
    - The computer must have instructions to follow

Short list!

9-4

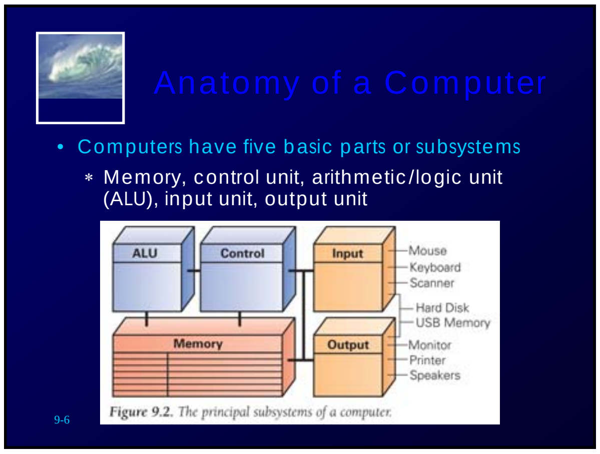


Instruction Execution Engines

- What computers can't do
  - Have no imagination or creativity
  - Have no intuition
  - Have no sense of irony, subtlety, proportion, decorum, or humor
  - Are not vindictive or cruel
  - Are not purposeful
  - Have no free will
  - Recent movies: Terminator, Matrix, AI

Long list!

9-5



Anatomy of a Computer

- Computers have five basic parts or subsystems
  - Memory, control unit, arithmetic/logic unit (ALU), input unit, output unit

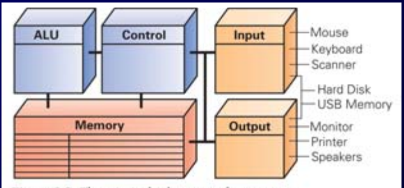


Figure 9.2. The principal subsystems of a computer.

9-6

Memory

- *Memory* stores the program running and the data on which the program operates
- Properties of memory:
  - Discrete locations—1 byte per location!
  - Addresses—For every memory location (byte)
    - whole numbers starting with zero
  - Values—Memory locations store values.
  - Finite capacity—Limited size—data may not "fit" in the memory location.
    - Overflow conditions, buffer overruns

9-7

Byte-Size Memory Location

- A commonly used diagram of computer memory represents the discrete locations as boxes (1 byte each).
- Address of location is displayed above the box.
- Value or contents of location is shown in the box.

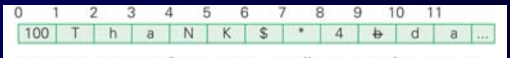


Figure 9.3. *Diagram of computer memory illustrating its key properties.*

9-8

Memory (cont'd)

- 1-byte memory locations can store one ASCII character, or a number less than 256 (0 - 255)
- Programmers use a sequence of memory locations together, ignoring the fact that they all have different addresses
  - Blocks of four bytes are used as a unit so frequently that they are called memory "words"

9-9

Random Access Memory (RAM)

- "Random access" means the computer can refer to (access) the memory locations in any order
- Often measured in
  megabytes (MB) – millions of bytes or
  gigabytes (GB) – billions of bytes
- Large memory is preferable because there is more space for programs and data (which usually equates to less I/O)

9-10

Control Unit

- Its circuitry fetches an instruction from memory, decodes the instruction, and fetches the operands used in it
  - A typical instruction might have the form

    ADD 4000, 2000, 2080 op dest, src1, src2
  - This instruction asks that the numbers stored in locations 2000 and 2080 be added together, and the result stored in location 4000 **[4000] = [2000] + [2080]**
  - Data/Operand Fetch step must get these two values and after they are added, Result Return/Store step will store the answer in location 4000

9-11

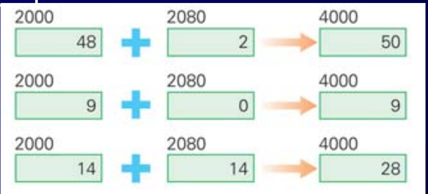
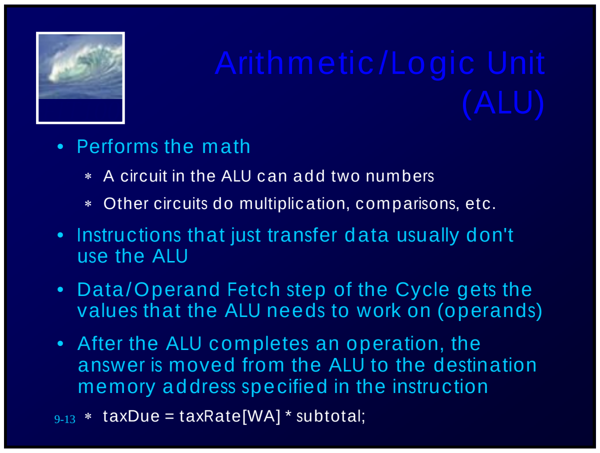


Figure 9.4. *Illustration of a single ADD instruction producing different results depending on the contents of the memory locations referenced in the instruction.*

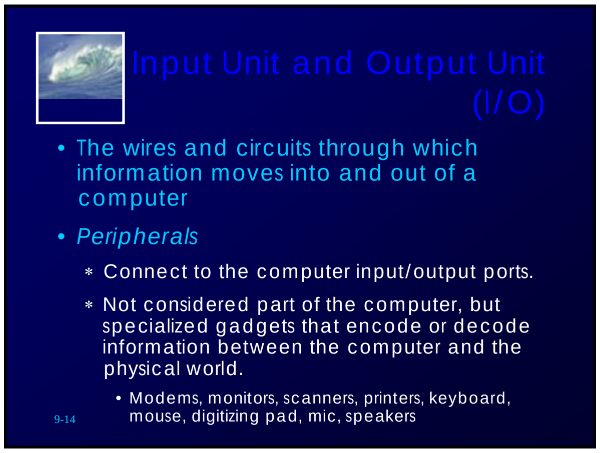
9-12



Arithmetic/Logic Unit (ALU)

- Performs the math
  - A circuit in the ALU can add two numbers
  - Other circuits do multiplication, comparisons, etc.
- Instructions that just transfer data usually don't use the ALU
- Data/Operand Fetch step of the Cycle gets the values that the ALU needs to work on (operands)
- After the ALU completes an operation, the answer is moved from the ALU to the destination memory address specified in the instruction
  - taxDue = taxRate[WA] * subtotal;

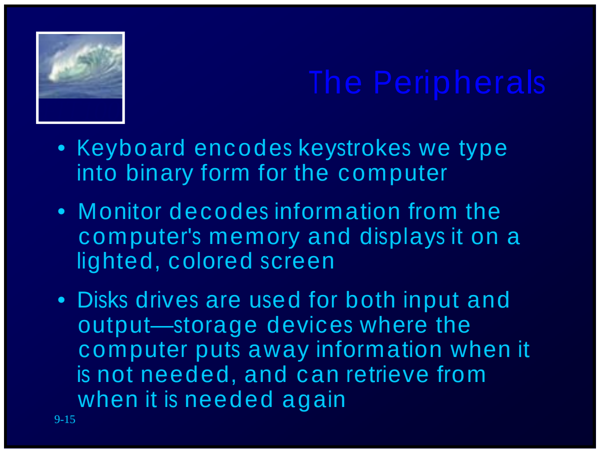
9-13



Input Unit and Output Unit (I/O)

- The wires and circuits through which information moves into and out of a computer
- *Peripherals*
  - Connect to the computer input/output ports.
  - Not considered part of the computer, but specialized gadgets that encode or decode information between the computer and the physical world.
    - Modems, monitors, scanners, printers, keyboard, mouse, digitizing pad, mic, speakers

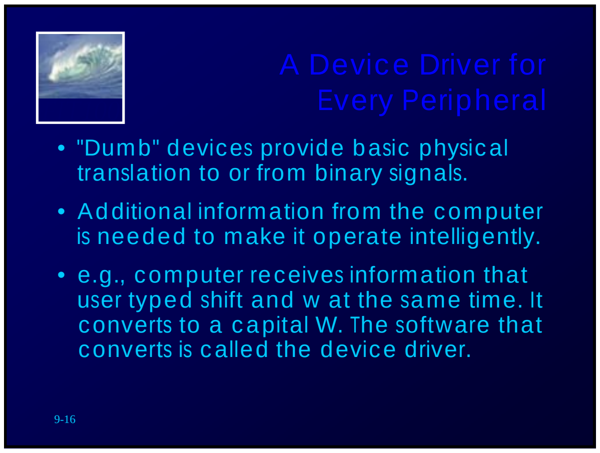
9-14



The Peripherals

- Keyboard encodes keystrokes we type into binary form for the computer
- Monitor decodes information from the computer's memory and displays it on a lighted, colored screen
- Disks drives are used for both input and output—storage devices where the computer puts away information when it is not needed, and can retrieve from when it is needed again

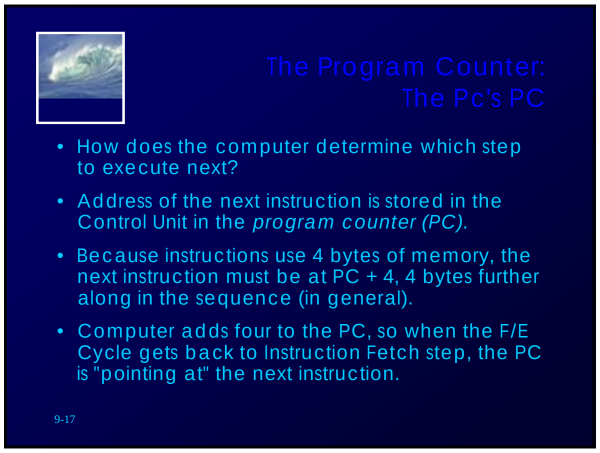
9-15



A Device Driver for Every Peripheral

- "Dumb" devices provide basic physical translation to or from binary signals.
- Additional information from the computer is needed to make it operate intelligently.
- e.g., computer receives information that user typed shift and w at the same time. It converts to a capital W. The software that converts is called the device driver.

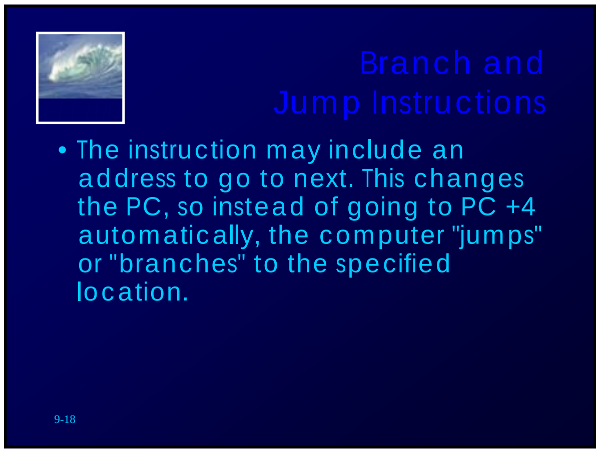
9-16



The Program Counter: The Pc's PC

- How does the computer determine which step to execute next?
- Address of the next instruction is stored in the Control Unit in the *program counter (PC).*
- Because instructions use 4 bytes of memory, the next instruction must be at PC + 4, 4 bytes further along in the sequence (in general).
- Computer adds four to the PC, so when the F/E Cycle gets back to Instruction Fetch step, the PC is "pointing at" the next instruction.

9-17



Branch and Jump Instructions

- The instruction may include an address to go to next. This changes the PC, so instead of going to PC +4 automatically, the computer "jumps" or "branches" to the specified location.

9-18

Instruction Interpretation

- Process of executing a program
  - Computer is interpreting our commands, but in its own language
- Before the F/E Cycle begins, some of the memory locations and the PC are visible in the control unit

9-19

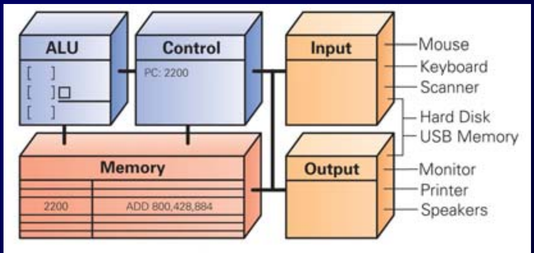


Figure 9.5. Computer before executing an ADD instruction.

The Fetch/Execute Cycle

- A five-step cycle:
  1. Instruction Fetch (IF)
  2. Instruction Decode (ID)
  3. Data Fetch (DF) / Operand Fetch (OF)
  4. Instruction Execution (EX)
  5. Result Return (RR) / Store (ST)

9-21

Animation

- Fetch/Execute Cycle

Cycling the F/E Cycle

- Computers get their impressive capabilities by executing many of these simple instructions per second
- The Computer Clock: Determines rate of F/E Cycle
  - Measured in gigahertz (GHz), or billions of cycles per second

9-23

How Important is Clock Speed?

- Modern computers try to start an instruction on each clock tick
- Pass off finishing instruction to other circuitry (*pipelining*)
  - Five instructions can be in process at the same time
- Does a 1 GHz clock really execute a billion instructions per second?
  - Not a precise measurement. Computer may not be able to start an instruction on each tick, but may sometimes be able to start more than one instruction at a time

9-24

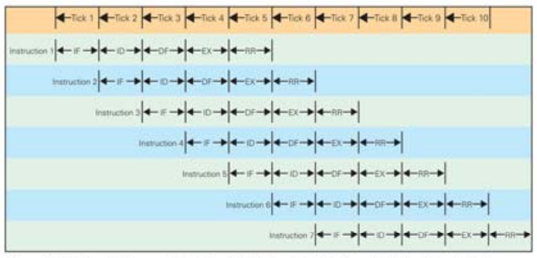
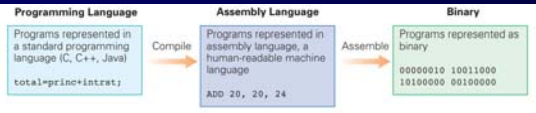


Figure 9.12. *Schematic diagram of a pipelined Fetch/Execute Cycle. On each tick, the IF circuit starts a new instruction, and then passes it along to the ID (Instruction Decode) unit; the ID unit works on the instruction it receives, and when it finishes, it passes it along to the DF (Data Fetch) circuit, and so on. When the pipeline is filled, five instructions are in progress at once, and one instruction is finished on each clock tick, making the computer appear to be running at one instruction per tick.*

Software

- A computer's view of software
  - Sees *binary object file*, a long sequence of 4-byte words (0's and 1's)
- *Assembly language*
  - Alternative form of machine language using letters and normal numbers so people can understand it
  - Computer scans assemble code, as it encounters words it looks them up in a table to convert to binary, converts numbers to binary, then assembles the binary pieces into an instruction


Programming Language
Programs represented in a standard programming language (C, C++, Java)
Compile
Assembly Language
Programs represented in assembly language, a human-readable machine language
Assemble
Binary
Programs represented as binary

```
total=princ+intrst;
```

```
ADD 20, 20, 24
```

```
00000010 10011000
10100000 00100000
```

Figure 9.13. *The three primary forms of encoding software: programming language, assembly language, and binary machine language.*

Operating Systems

- Basic operations that are necessary for the effective use of computer, but are not built into the hardware
- Three most widely used Operating Systems:
  - Microsoft Windows
  - Apple's Mac OS X
  - Unix / Linux
- OS performs booting, memory management, device management, Internet connection, file management

Programming

- Programmers build on previously developed software to make their jobs easier
- Example: GUI Software
  - Frame around window, slider bars, buttons, pointers, etc. are packaged for programmers and given with OS

Integrated Circuits

- Miniaturization:
  - Clock speeds are so high because processor chips are so tiny (electrical signals can travel about 1 foot in a nanosecond)

Integrated Circuits

- Photolithography
  - Printing process. Instead of hand-wiring circuits together, photograph what is wanted and etch away the spaces
  - Regardless of how complicated the wiring, cost and amount of work are the same

9-31

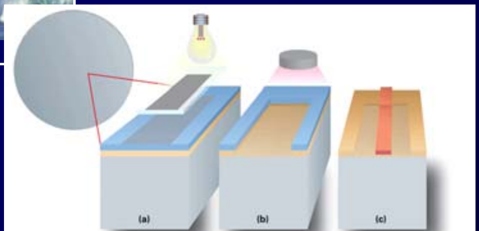


Figure 9.16 Early steps in the fabrication process. (a) A layer of photoresist (blue) is exposed to UV light through a pattern mask (light blue), hardening the exposed areas; (b) after washing away the unexposed photoresist, hot gases etch away (nearly all of) the exposed layer; (c) the remaining resist is washed away and other layers are created by repeating the patterning and etching processes. In later stages of the fabrication process, (d) "impurities" (green) such as

9-32

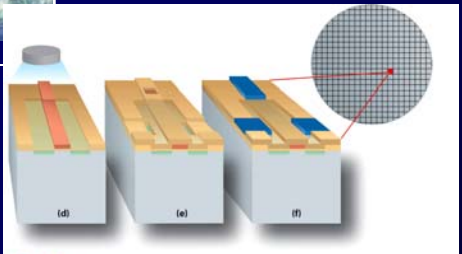


Figure 9.16 *Continued*
boron are diffused into the silicon surface in a process called doping, which improves the availability of electrons in this region of the silicon. (e) After additional layering, etching exposes contact points for metal wires, and (f) a metal (dark blue) such as aluminum is deposited creating "wires" to connect to other transistors. Millions of such transistors form a computer chip occupying a small square on the final fabricated wafer.

9-33

How Semi-Conductor Technology Works

- Integration:
  - Active components and the wires that connect them are all made together of similar materials in a single process
  - Saves space and produces monolithic part for the whole system, which is more reliable
- Silicon is a semi-conductor—sometimes it conducts electricity, sometimes not
  - Ability to control when semi-conductor conducts is the main tool in computer construction

9-34

The On-Again, Off-Again Behavior of Silicon

- A circuit is set to compute x and y for any logical values x and y
- If x is true, the x circuit conducts electricity and a signal passes to the other end of the wire; if x is false, no signal passes
- Same process for y
- If both circuits conduct, x and y are true—logical AND has been computed

9-35

The Field Effect

- Controls the conductivity of the semiconductor
- Objects can become charged positively or negatively
  - Like charges repel each other, but opposites attract. This effect is called the *field effect*.

9-36

The Field Effect (cont'd)

- The gap between two wires is treated to improve its conducting and non-conducting properties
- This is called a *channel* (path for electricity to travel between the two wires)
- An insulator covers the channel
- A wire called the *gate* passes over the insulator
- The gate is separated from the channel by the insulator—does not make contact with the wires or the channel
- Electricity is not conducted between the two wires unless the channel is conducting

9-37

How Does the Channel Conduct?

- The silicon in the channel conducts electricity when it is in a charged field
  - Electrons are attracted or repelled in the silicon material
  - Charging the gate positively creates a field over the channel that conducts electricity between the two wires

9-38

Transistors

- A connector between two wires that can be controlled to allow charge to flow between the two wires, or not
- We have described a MOS transistor (Metal Oxide Semiconductor)

9-39

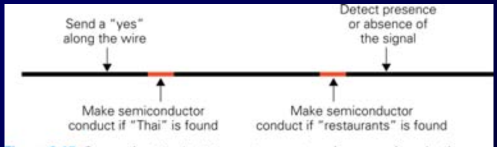

Send a "yes" along the wire
Detect presence or absence of the signal
Make semiconductor conduct if "Thai" is found
Make semiconductor conduct if "restaurants" is found

Figure 9.17 Computing `Thai AND restaurants` using a semiconducting material.

9-40

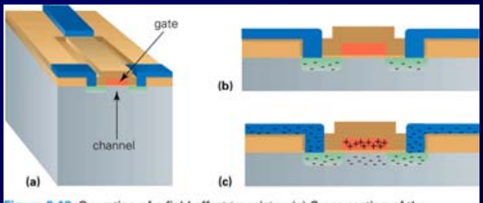


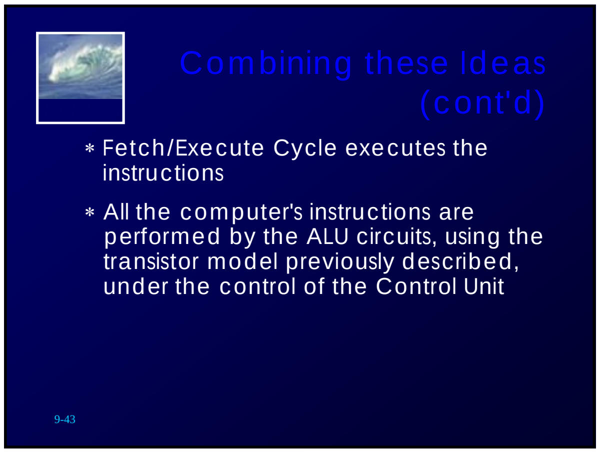
Figure 9.18 Operation of a field effect transistor. (a) Cross-section of the transistor of Figure 9.16(f). (b) The gate (red) is neutral and the channel, the region in the silicon below the gate, does not conduct, isolating the wires (blue); (c) charging the gate causes the channel to conduct, connecting the wires.

9-41

Combining the Ideas

- Put all these ideas together:
  - Start with information processing task
  - Task is performed by application, implemented as part of a large program in a high-level language like C or Java
  - Program performs specific operations; standard operations like print or save are done by OS
  - Program's commands are compiled into assembly language instructions
  - Assembly instructions are translated into binary code
  - Binary instructions are stored on hard disk (secondary memory)
  - Application instructions move into RAM (primary memory)

9-42



Combining these Ideas (cont'd)

- Fetch/Execute Cycle executes the instructions
- All the computer's instructions are performed by the ALU circuits, using the transistor model previously described, under the control of the Control Unit

9-43

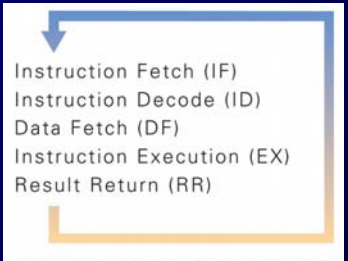


Figure 9.1. *The Fetch/Execute Cycle.*

9-44