CSCU9A1: Systems 2

Leslie Smith

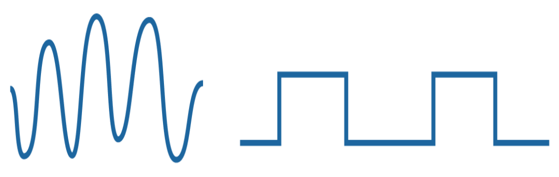
Last updated October 2016

UNIVERSITY OF STIRLING

Lecture 2: Content

- Back to basics: Analogue vs digital, Memory
- Memory elements
- Memory hierarchy
- Permanent memory
- Data representation (elementary)

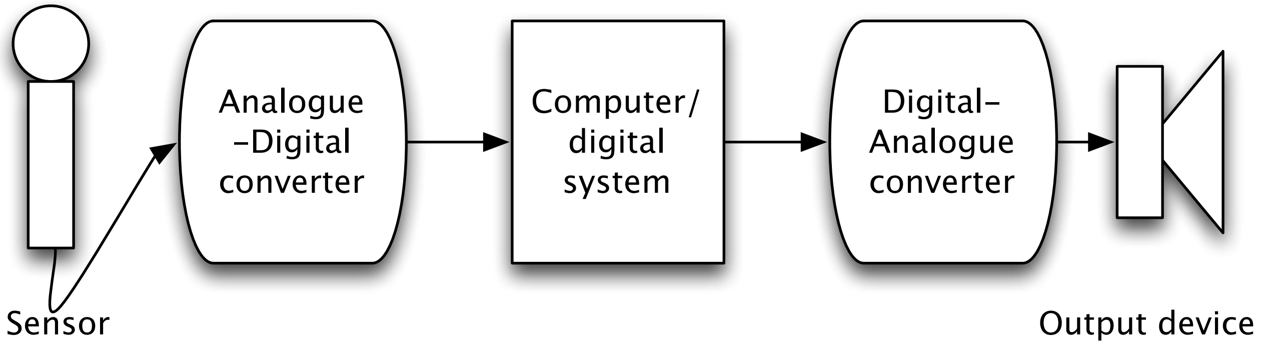
Analog(ue) vs Digital



An analog signal and a digital signal

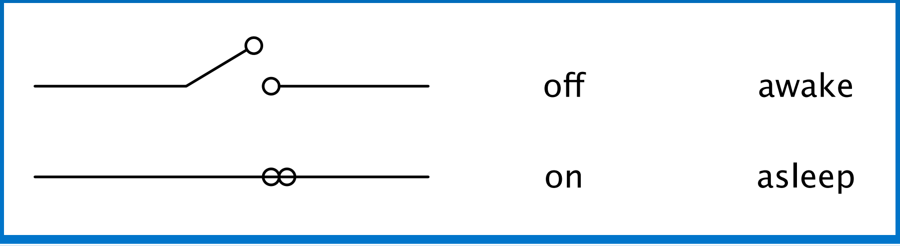
- Analogue (analog in US English) vs Digital
  - Analogue values have a continuous range
    - Like Real numbers: there's another value between any two values
  - Digital values have a finite number of possibilities
    - Like a fixed set of integers, e.g. {0,1,2,3,4,5}. Binary has 2 values only.
- Examples.

  Temperature, speed, acceleration, time vs age next birthday, gender, time in hours:mins, …
- Everything on a digital computer is coded digitally
  - No continuous values
- But the world (and most of our sensors) are analogue
  - Need to be converted prior to input (and often for output too)



Digital memory

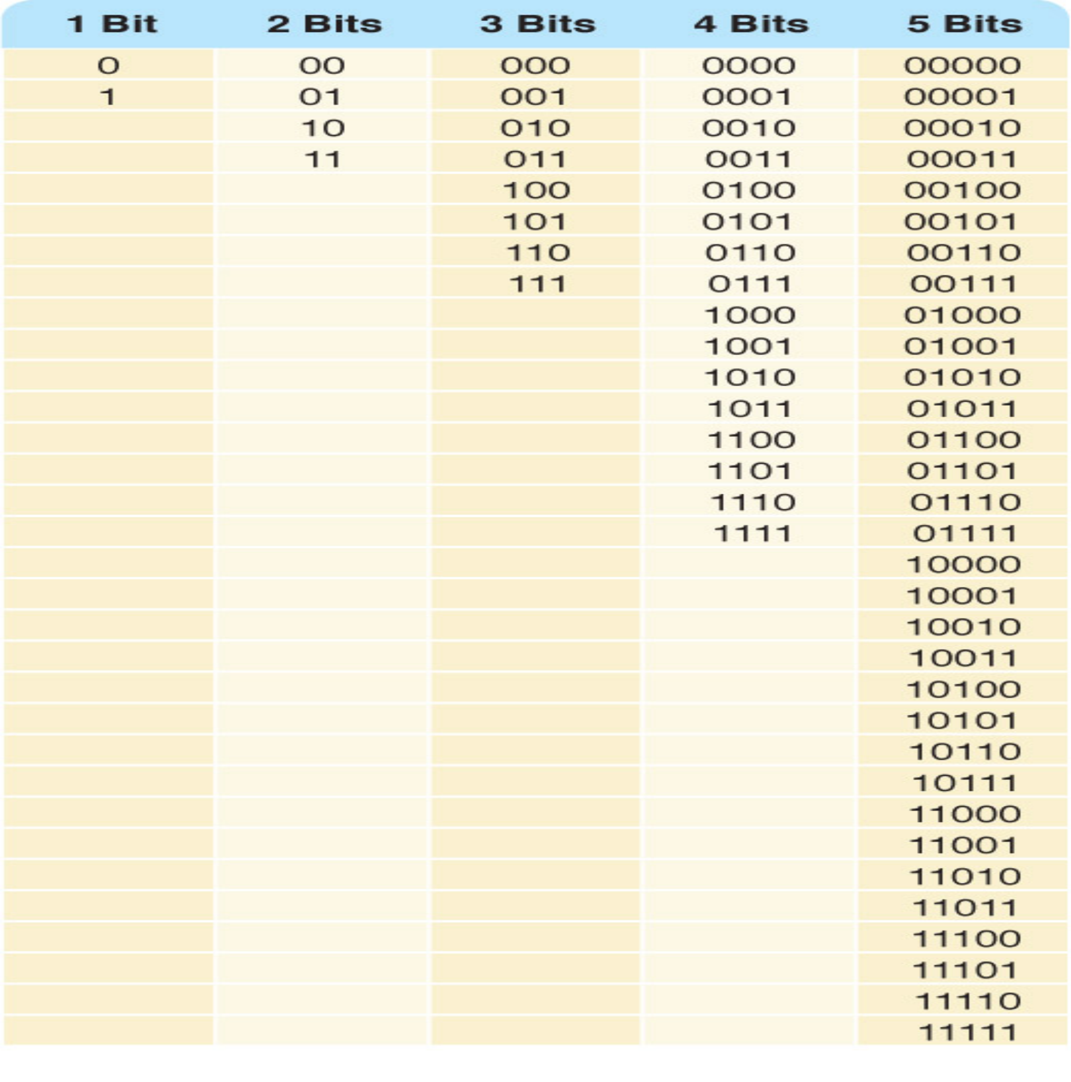
- The simplest digital memory system would have just a single value
  - 1=1, 11=2, 111=3, 1111 =4 …
- But this would require a variable number of storage elements
  - And how would we represent 0?
- The next simplest digital memory system has two values (or two *states*).
  - Call them 0 and 1, True and False, …
- Each memory element is a binary digit, otherwise known as a *bit*.
  - What each bit *means* is contextually defined



Binary representations

- N bits can represent

  2^N different values



1 Bit	2 Bits	3 Bits	4 Bits	5 Bits
0	00	000	0000	00000
1	01	001	0001	00001
	10	010	0010	00010
	11	011	0011	00011
		100	0100	00100
		101	0101	00101
		110	0110	00110
		111	0111	00111
			1000	01000
			1001	01001
			1010	01010
			1011	01011
			1100	01100
			1101	01101
			1110	01110
			1111	01111
				10000
				10001
				10010
				10011
				10100
				10101
				10110
				10111
				11000
				11001
				11010
				11011
				11100
				11101
				11110
				11111

Bits, bytes and words

- A single bit can only store one of two values
- Bits are normally used many at a time.
  - A sequence of 8 bits is often called a *byte*
    - 8 bits implies 2^8 different possible values, which might be interpreted as the integers from 0 to 255 inclusive.
      - Or as the numbers -128 to + 127 inclusive, or …
  - A *word* is a sequence of bytes
    - A word is usually the size of a sequence of bits that the CPU can operate on (virtually always a multiple of 8)
      - How many bytes are in a word depends on the particular CPU
    - Modern laptops and desktops use 64 bit (or 8 byte) words
    - But many machines use 32 but words (4 byte)
    - Some embedded computers use 16 bit words, or even 8 bit words
      - Smaller word length CPUs are simpler, and use less power.

But what might a sequence of bits *mean*?

- Meaning (representation) is ascribed by the program operating on the sequence of bits.
  - Discuss: syntax vs semantics.
- So, a 32 bit word might represent
  - An (unsigned) integer between 0 and 4,294,967,295
  - A (signed) integer between -2,147,483,648 and 2147,483,647
  - 8 (ASCII) characters e.g. ‘aBcDeFgH’
  - 4 (UniCode) characters e.g. ‘ΑΓΩö’
  - An instruction to add the next two words together
  - Plus many other possibilities
    - Limited only by the programmer’s ingenuity

Memory (slide from first lecture)

Memory

A collection of cells, (=words) each with a unique physical address

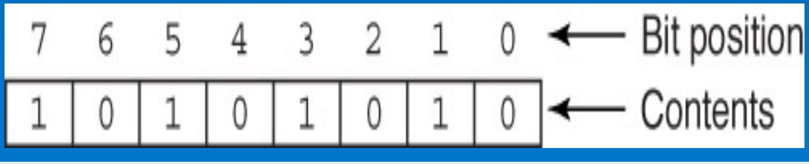
Each cell is made up of a number of bits

8, 16 24, 32, 64

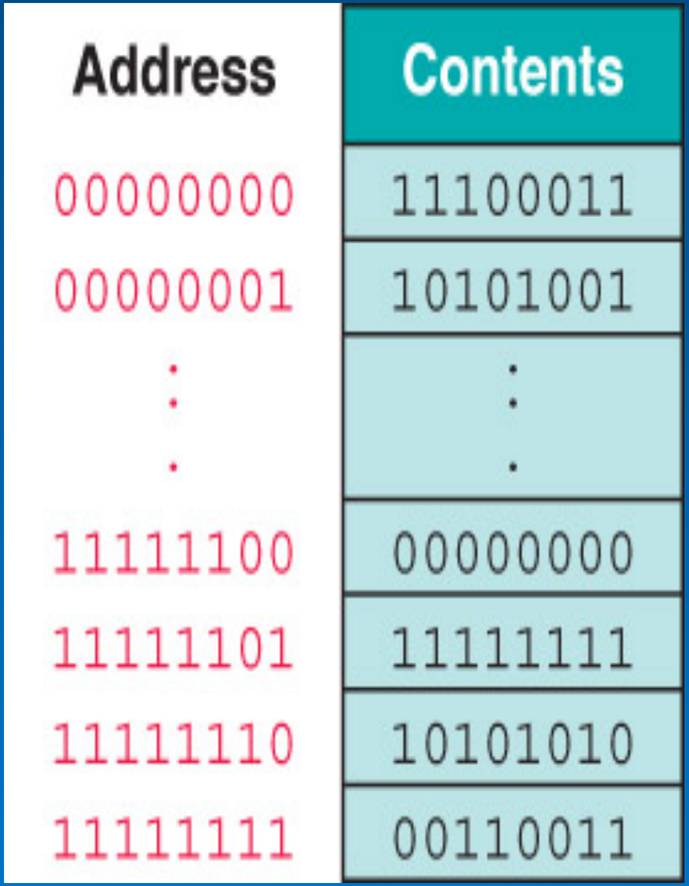
binary digits (bits)

Addresses start at 0, and are usually contiguous

Both addresses and contents are in binary



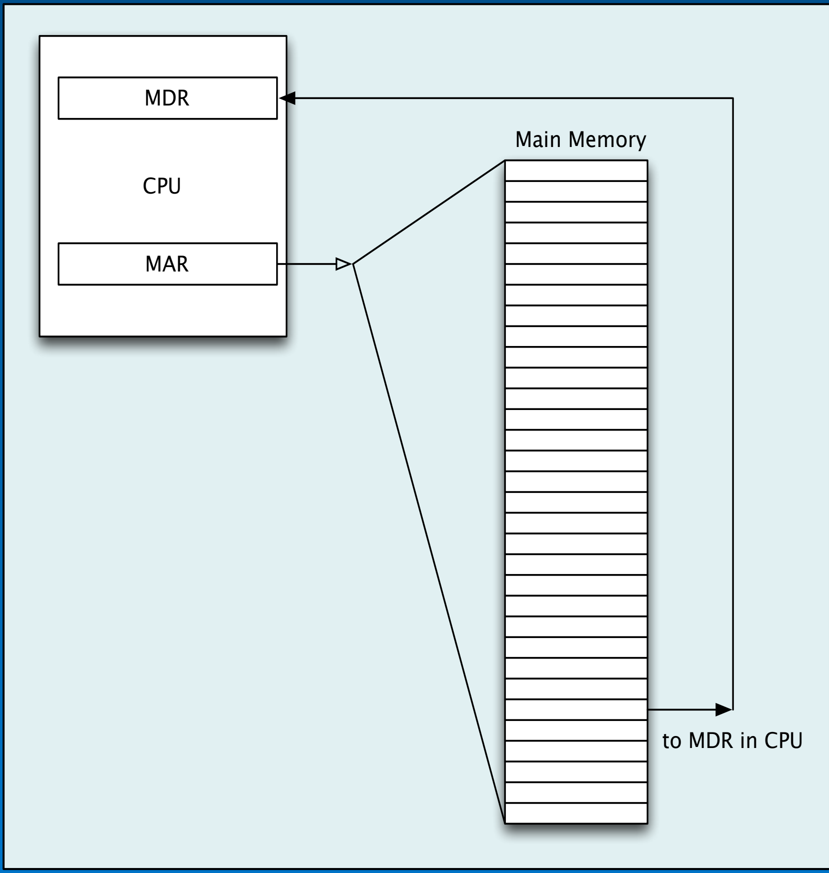
7	6	5	4	3	2	1	0	← Bit position
1	0	1	0	1	0	1	0	← Contents



Address	Contents
00000000	11100011
00000001	10101001
⋮	⋮
11111100	00000000
11111101	11111111
11111110	10101010
11111111	00110011

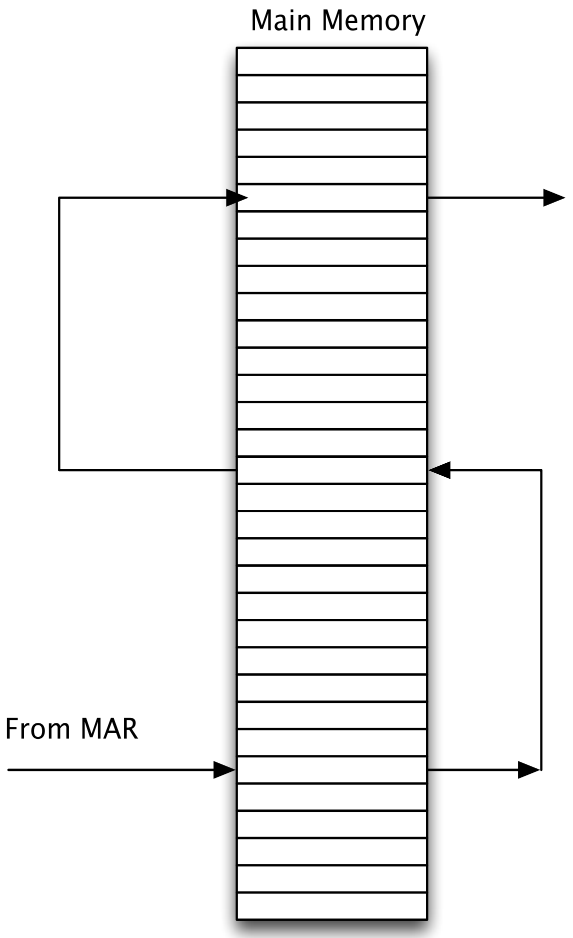
The main memory

- Main memory is accessed using the address of each byte (sometimes of each word).
- Words are <wordLength> long
- The length of addresses depends on the registers in the CPU
  - 32 bit addresses: maximum of 2^{32} words (=4 Gbytes)
  - 40 bit addresses: maximum of 1024 Gbytes.
- The Memory Address Register (MAR) in the CPU is used to select a particular word
- The word is then written to/read from the Memory Data Register (MDR)



Memory and pointers

- Inside the CPU we can easily copy from the MAR to the MDR
  - MDR ← MAR
- So we can use a value in a word as the *address* of another word
  - … and we can continue this:
    - use a word in the memory as the address of the address of a word
    - Or as the address of the address of the address of a word
      - Called *indirection* (or using memory words as *pointers* (or pointers to pointers (or pointers to pointers to pointers, …)))



The memory hierarchy (1)

- There are many different types of memory in a modern desktop computer
  - We have seen:
    - Registers in the CPU
    - Main memory
      - And early computers used only these
  - Modern computers use more levels of memory

Human and Computer memory

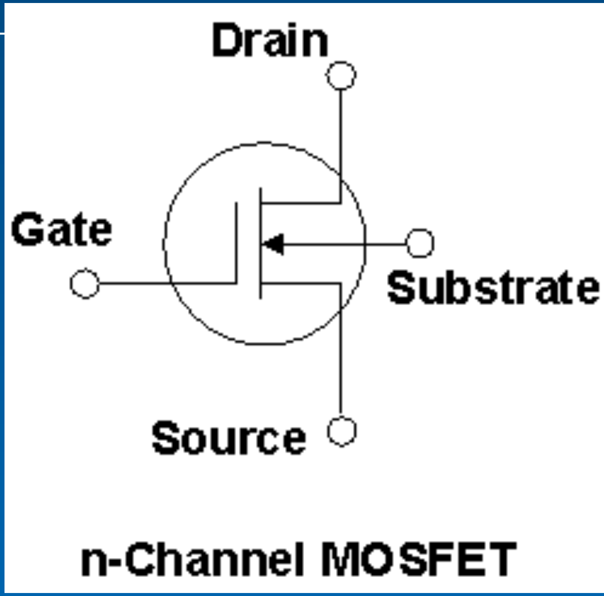
- Think about human memory:
  - we recall some information only for a very short time, and use it when (say), reading;
  - other things are recalled for a bit longer, for example where we put our bicycle or our keys,
    - And sometimes memory is episodic, rather than just words or places
  - and yet other things for longer still, for example the names of our siblings and friends…
- Human memory is not a simple concept!
  - Computer memory is all 1s and 0's
    - But the meaning and use is created by the software

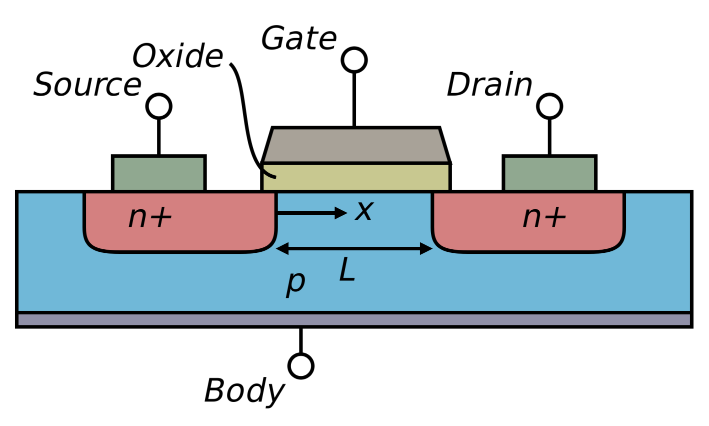
The memory hierarchy (2)

- Computer hardware memory hierarchy
  - CPU registers
  - 1st level cache
  - (2nd level cache)
  - (3rd level cache)
  - Main Memory
  - Hard Disk Memory (May be solid-state on more recent machines)
  - Removable storage (Optical disk, USB sticks, etc.)
  - Cloud storage

Semiconductor memory

- The fast memory inside a computer is electronic
  - That is, made up from transistors
    - Metal Oxide Semiconductor Field Effect Transistors to be precise, usually called MOSFETs
    - And built using a technology called Complementary Metal Oxide Semiconductor (CMOS)
  - Registers, Cache, and main memory require very high speed access (of the order of 10 nanoseconds or less, currently)
    - This type of memory loses its content when the power is removed.
    - That is, it is *volatile*.





The memory hierarchy (2a)

- Computer hardware memory hierarchy
  - CPU registers
  - 1st level cache
  - (2nd level cache)
  - (3rd level cache)
  - Main Memory
  - Hard Disk Memory (May be solid-state on more recent machines)
  - Removable storage (Optical disk, USB sticks, etc.)
  - Cloud storage

Volatile

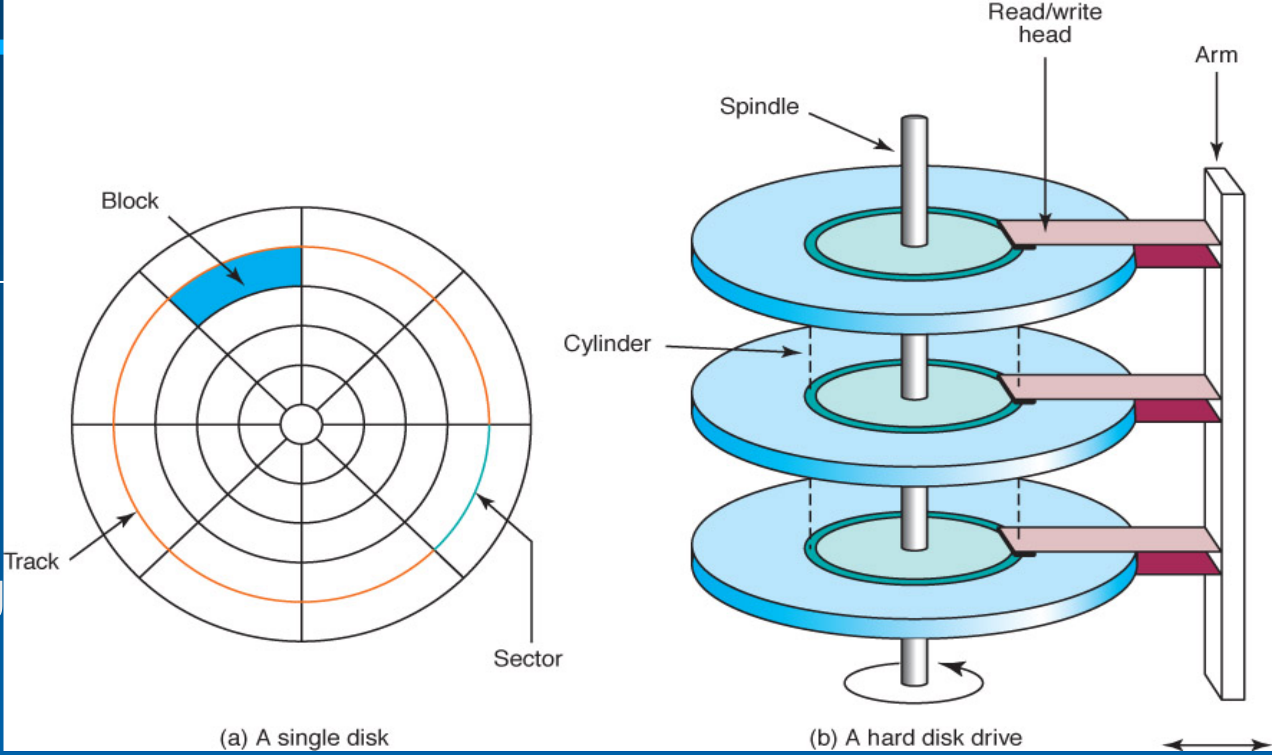
Non- Volatile

Permanent (non-volatile) memory

- Required for storing
  - User information (data)
  - Operating system (Windows 7,8, 10, Linux, Mac OSX)
  - User's applications
- Used to initialise volatile memory when the machine is turned on
  - Or *booted* (sometimes called *bootstrapped*)
- Non-volatile memory takes a number of forms

Disk drives

- One of the oldest forms of non-volatile storage
- Can hold up to 5Tbytes of data
- Access requires placing the read-write head over the data required
- Then waiting for the data to come under the head
- Then reading it
- Takes a while for the data to start to be read
  - Latency
- Then data reading is relatively fast



(a) A single disk

(b) A hard disk drive

–Disk is made up of

–Surfaces, each made up of

–Tracks, each made up of sectors

–Logically, the data is arranged in files

–Files are mapped to physical sectors by the operating system

Optical storage

CD

A compact disk uses a laser to read information stored optically on a plastic disk; data is evenly distributed around track

CD-ROM read-only memory

CD-WORM write once, read many

About 750Mbytes/disk

DVD

Digital Versatile Disk, used for storing audio and video (similar technology: more recent, higher density recording)

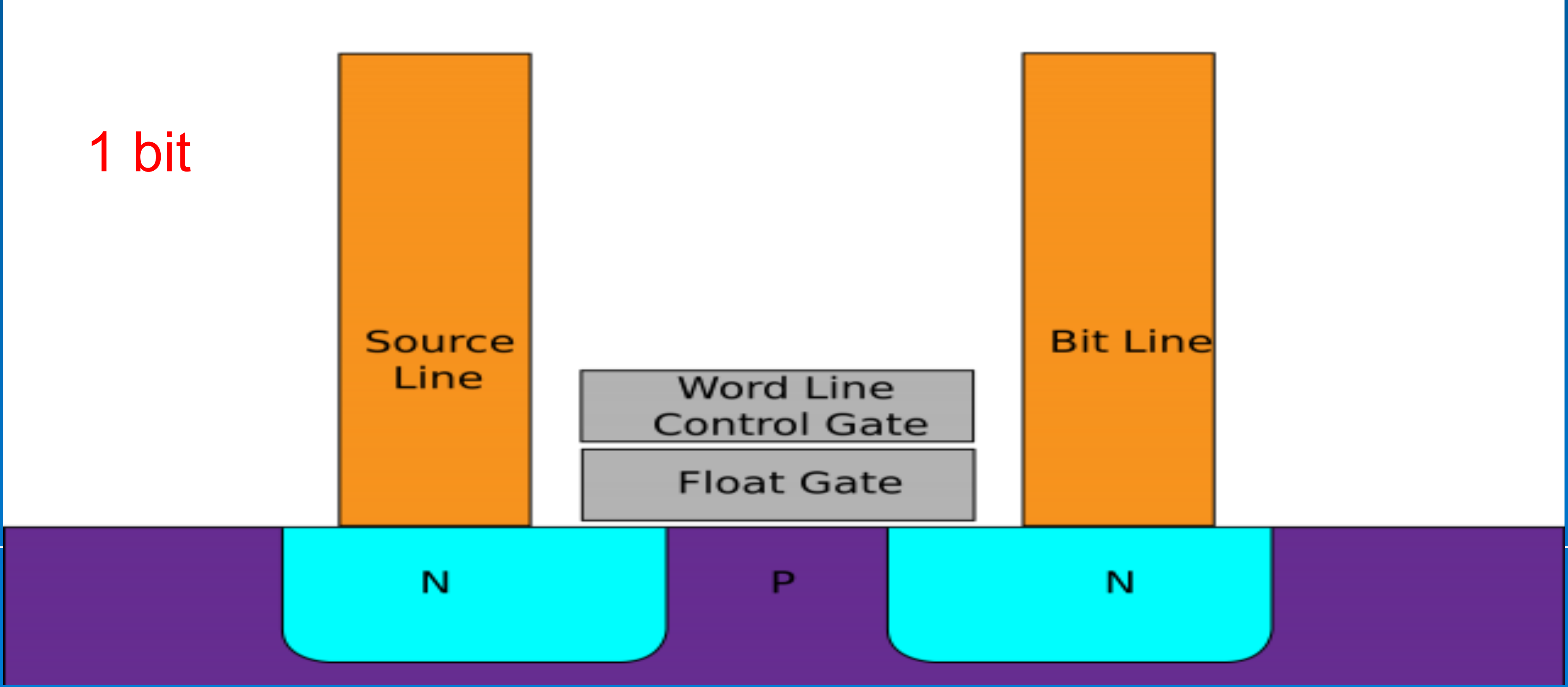
About 4-8Gbytes/disk

Blu-Ray: another more recent form of DVD: uses a shorter-wavelength laser to increase data density.

About 25-50Gbytes/disk

Solid State Memory

- Memory is made up of a large number of non-volatile single bit semiconductor memory elements: sometimes called flash memory



Non-volatile semiconductor storage

- Flash memory technology allows semiconductor memory to retain data even when the power is removed
- Generally not as fast to write as volatile semiconductor memory
  - And generally takes longer to write than to read
- But very robust physically
  - No moving parts
  - Good for portable memory, such as USB sticks
  - Good for Laptops (but not just for laptops)
    - Used in some recent laptop designs (e.g. recent MacBook Pro comes with 256-768Gbytes of solid state drive (flash) memory, and no hard disc).

Solid State Drives

- Not “drives” at all
  - No moving parts
- Large number of flash memory chips
  - Making up 256Gbyt to 1 or even 2 Tbytes of memory
- Read speed is very fast
  - Not quite as fast as RAM, but faster than Disk
- Write speed can be relatively slow
  - But the “drive” technology gets round this using internal RAM buffering and other techniques.

Data representation

- Returning to what might a string of bits *mean?*
- Everything in the computer system
  - At every level of the memory hierarchy
- … is a string of bits
- So a string of bits can represent anything at all that might be represented inside a computer.
  - Which depends entirely on the program using the string of bits
  - And might be:
  - Numbers (integers, floating-point numbers, complex numbers …)
  - Addresses (pointers) of data elements
  - Text (in Latin text, Greek, Cyrillic, Chinese, Korean, …)
  - Images (photographs, diagrams, …)
  - Sounds (speech, music, …)
  - Programs (applications, program sections from the operating system)
  - Pointers to programs,
  - And so on.