1. Buses
    a. Key characteristic of bus
        i. Shared transmission medium
        ii. Anything using the bus must obtain the “right” to use the bus first
    b. Functional groups
        i. Data lines
        ii. Address lines
            1. Select where to source from, where to send to
            2. Could pick different parts of the CPU, memory, or I/O
        iii. Control lines
            1. Control use of data and address lines
            2. Variety of signals
                a. Memory write/read
                b. I/O write/read
                c. Transfer ACK (acknowledgment), data has been taken from or placed on bus
                d. Bus request/grant, to give permission to somebody who wants the bus
                e. Interrupt request/ACK, will discuss this later
                f. Clock, same one we’ve been talking about
                g. Reset
    c. Types of buses
        i. Dedicated – permanently assigned to a subset of components, or to one function
            1. Example: data bus versus address bus
            2. Need more buses to send all information
        ii. Time multiplexed – using bus for multiple functions based on clock
            1. Send address for first part of clock, then data for second
            2. Don’t need as many buses, but can’t transmit at same speed a dedicated bus can

2. Bus hierarchies
    a. Remember, shared transmission medium
    b. Multiple bus hierarchies
        i. Same idea as before, create multiple buses
        ii. Reduce length of entire bus, allow parallel data transfer
        iii. Traditional
            1. CPU ↔ local bus ↔ cache ↔ system bus ↔ main memory and expansion bus interface
            2. Expansion bus interface ↔ expansion bus ↔ network and other I/O
        iv. High performance
            1. CPU ↔ local bus ↔ cache/bridge ↔ high-speed bus
            2. High-speed bus ↔ main memory
            3. High-speed bus ↔ video, network, GPUs, expansion bus interface
            4. Expansion bus interface ↔ other I/O
        v. For high performance, place important devices closer to CPU
            1. Skip level of indirection by hooking up directly to high-speed bus

3. Bus specifics
    a. Bus width
        i. How big are our addresses?
        ii. How big is our data?
            1. Larger bus means transferring more data at once
    b. Bus clock time
        i. Bus driven by input clock
        ii. Determines how much data we can transfer at once
        iii. Example below, want to determine throughput (bits / second) of a bus
            1. 32-bit bus, 1 GHz input clock = 1 ns
                a. Frequency (Hz) = 1 / seconds (s)
                b. Giga = 10^9, so 1 GHz = 10^9 Hz
                c. Invert 1 GHz, which is our frequency
                d. Inverting this, we get 1 / 10^9 Hz = 10^{-9} seconds
                e. Nano prefix = 10^{-9}, so 10^{-9} seconds * (10^9 nanoseconds / 1 second) = 1 nanosecond
            2. Bus cycle takes at least 100 input cycles
                a. Will take at least 100 clock cycles to do one bus cycle
                b. So 32 bits / bus cycle * 1 bus cycle / 100 clock cycle = 0.32 bits / clock cycle
                c. Key word is *at least*, so this is a theoretical maximum
            3. Transfer rate = (bits / bus cycle) * (bus cycles / clock cycle) * (clock cycles / time)
                a. 32 bits / bus cycle * 1 bus cycles / 100 clock cycles * 1 clock cycle / 1 ns
                b. 0.32 bits / ns * 10^9 nanoseconds / 1 second = 3.2 * 10^8 bits / s
            4. So 3.2 * 10^8 bps (bits / s) is our theoretical maximum
                a. Can turn this into a more reasonable number by converting to megabits
                b. Mega prefix = 2^{20}
                c. 3.2 * 10^8 bits / second * 1 megabit / 2^{20} megabits = ~ 305 megabits / second
            5. 305 Mbps (megabits per second) is the theoretical maximum throughput for this bus