Exam 1

Nathan Warner

Computer Science
Northern Illinois University
United States

<div align="center">

Contents

</div>

Conceptual questions on history

- **What is a stored program computer? Who was Atanasoff? Who were Mauchley and Eckert?**: A stored program computer is a computer that stores programs so that you can change them without changing the hardware.

  Atanasoff was the creator of the Antanasoff Berry computer, a machine that solved systems of linear equations. It was a first generation computer that used vacuum tubes.

  Mauchley and Eckert were the creators of a different first generation computer called the ENIAC. This machine also used vacuum tubes.

- **Why were transistors an improvement over vacuum tubes? Why were integrated circuits an improvement over vacuum tubes?**:

  Transistors were an improvement over vacuum tubes because they were solid state, smaller, consumed less power, generated less heat, and were faster.

  Integrated circuits were an improvement over vacuum tubes since they were smaller, more reliable, faster, and consumed less power.

- **What is a VLSI chip? What does VLSI stand for? What significant development in computer design did VLSI make possible?**: Very large scale integrated circuits are large integrated circuits with over 10000 components.

  VLSI made possible the creation of the microprocessor.

- **Newer IBM mainframes used the same architecture as their predecessors (which instructions were available and what they did) but newer hardware designs. How did that decision contribute to the success of IBM in the computer industry?**: By using the same architecture, IBM created an upgrade path, software did not need to be rewritten as they advanced their machines.

This decision led to more customers and customer loyalty, since they didn't need to rewrite software after upgrading their machines.

- **What is backward compatibility? Why is it important? Name one or two hardware families where backward compatibility has significantly influenced design.**: Backwards compatibility is the ability that a new machine can run the software intended for a machine of an older generation.

  This is essential because it allows users to not have to keep rewriting software as the industry advanced.

  Two hardware families where backward compatibility has significantly influenced design are IBM and Microsoft.

- **What is Moore's Law? Why can't it hold true forever?**: Moore's law states that the density of transistors in an integrated circuit double each year.

  This can not hold true forever because eventually you would get a transistor smaller than what is physically possible.

Gates

- **Draw the symbols for the following gates: A AND B, A OR B, NOT A, A XOR B, A NAND B, A NOR B. Give the truth table for each gate.**:

- **Make sure you know the order of operations for Boolean expressions and the various symbols that can be used.**:

    1. ()
    2. NOT
    3. AND
    4. NAND, NOR
    5. XOR
    6. OR

- **How many rows are there in a truth table for an expression with n variables? What does it mean when there is a 1 in the result column?**: 2^n rows, a one in the result column means that the expression is true for that specific input combination.

- **What is sum-of-products form? Why is it useful? If an expression has n variables, how many variables will be in each term of the sum-of-products form? What is the maximum number of terms there could be? How can you derive the sum-of-products form from the truth table?**: Sum-of-products is a standardized way to write boolean functions. It is useful to compare two functions that may not look the same but have the same outputs given the same inputs.

    If a expression has n variables there will be n variables in each term. The maximum number of terms is 2^n, if each row is true.

    To derive the sum-of-products form you look at the truth table. If the result column is one, then you add that columns min-term to the result. The sum-of-products final form is the OR of all the min-terms.

- **Why are Boolean identities useful in hardware design? Make sure you know the names of each of the Boolean identities.**: The identities allow us to simplify complex boolean functions into simpler ones. They allow us to make smaller circuits that are less complicated.

- **What is the dual of an expression? Why is it useful?**: The dual of an expression is when you switch replace AND with OR, or OR with AND, and 1 with 0, or 0 with 1. The duals of expressions are useful because if one expression is true, we know the dual is also true.

- **What is a universal gate? Which gates are universal gates? Why are universal gates useful?**: A universal gate is a gate that can be used to build any other logic function or circuit. The universal gates are NAND and NOR.

    They are useful because they are cheap to make, and you only need one gate.

- **Show that NAND is not associative. Suggestion: draw truth tables for ((A NAND B) NAND C) and (A NAND (B NAND C)).**:

Combinatorial circuits

- **What is a half adder? How many bits can a half adder add? Show a black box representation.**: A half adder is capable of taking two bits and adding them together. The half adder returns the result and a carry-out

- **What is a full adder? How is it different from a half adder? Show a black box representation of a full adder. Show how to implement a full adder with two half adders.**: A full adder can instead add together three bits instead of two. This is useful for adding two bits and the carry from the sum of the previous bits.

- **Show how to implement an adder for 2-bit binary numbers using two full adders.**:

- **How many full adders would it take to add two 16-bit binary numbers? If you used one half adder, how many full adders would you need? Which column would you use the half adder for?**: 16 full adders, 15 full adders, rightmost column (least significant bits)

- **What are the inputs and outputs of a ripple-carry adder? (i.e., draw a black box representation). What does a ripple-carry adder do? What components is it made of?**: The ripple carry adder takes two binary numbers and adds them together, it returns the sum

  It is made by connecting full adders in series, where the carry from the previous addition ripples to the next full adder.

- **What does a decoder do? Draw a black box diagram. What is it useful for? If you have 8-bit addresses, how many input lines does your decoder need? How many output lines? If the input is 0000 0110, which output line(s) will be activated?**: A decoder converts a binary number to a single value (0th, 1st, 2nd, etc)

  It is useful for selecting memory cells given an address. It is also useful for making a single selection based on the input. For example, in a 2-bit ALU the decoder is used to select the operation based on the values of the control lines.

  For an 8-bit address, the decoder needs 8 input lines and $2^8 = 256$ output lines. If the input is 00000110, the 7th output line will be activated (from bottom to top).

- **What does a multiplexer do? Draw a black box diagram.**:

  The multiplexer is the opposite of a decoder, it selects a single input to output based on the value of the control lines.

- **If a multiplexer has 8 selection lines (control lines), how many input lines and how many output lines can it have?**: $2^8 = 256$ input lines and one output line.

- **What does a shifter do? If a shifter has 8 input lines, how many output lines will it have? How many control lines? What do(es) the control line(s) indicate?**: A shifter takes a binary number and shifts each bit either left or right. A shifter has one control line. If the value of the control line is one, right shift, otherwise left shift.

- **What is an ALU? What kinds of operations does it typically handle?**: An ALU is an arithmetic logic unit. It handles both arithmetic and logical operations.

- **If an ALU can handle 16 operations on 32-bit numbers, how many control lines does it need? How many data lines does it need for each input? How many data lines does it need for the output?**: For 16 operations, we need $\log_2(16) = 4$ control lines. For each input, we need 32 data lines. It needs 32 data lines for the output.

Sequential circuits

- **What is the difference between a sequential and a combinatorial circuit? Why does a sequential circuit need a clock line?**: A combinatorial circuit produces output at the instant the input is given (at the speed of light). A sequential circuit is a circuit that allows events to be sequenced. State changes in a sequential circuit are controlled by clock ticks.

  The sequential circuit needs the clock line to make sure that state changes only happen when the clock ticks.

  The clock line synchronizes state changes.

- **What is a clock line? What is the relationship between the clock and state changes in a sequential circuit?**: A clock line is the periodic timing signal (square wave) that tells the sequential circuit to make state changes / update their outputs.

  The relationship between the clock and sate changes in a sequential circuit is that the state changes only when the clock ticks.

- **What is the rising edge? falling edge? edge-triggered circuit? level-triggered circuit? What type of circuits are the ones in this course?**: The rising edge is the moment when the clock signal changes from low to high

  The falling edge is the moment when the clock signal changes from high to low

  Level-triggered circuits are circuits that change state when the clock reaches highest or lowest levels.

  Edge-triggered circuits are circuits that change state when the clock is on the rising or falling edge.

  The circuits in this course are edge-triggered

- **What is feedback? Why is it useful?**: Feedback is when a sequential circuit loops its output back to its input. It allows circuits to retain their state values.

- **What is an SR flipflop? What is a JK flipflop? Why is the JK flipflop an improvement over the SR flipflop?**: An SR flipflop is a feedback circuit that either sets or resets (turns off) state.

  The JK flipflop is the same but fixes the problem in the SR flipflop when $S = R = 1$ (undefined in SR). This improvement allows the circuit to be stable regardless of the input combination.

- **What is a D flipflop? Why is it useful?**: The D flipflop is a feedback circuit that remains the same during clock ticks, the output only changes if the value of D changes.

  Built by sending D and $\bar{D}$ into the inputs S and R of an SR flipflop.

It is the fundamental circuit of computer memory, since its value is constant unless the input changes.

- **What is a characteristic table? How is it different from a truth table?**: A characteristic table is like a truth table but it allows for variables as values in the rows. A truth table does not allow this.

  it expresses state-transition behavior

- **Which type of flipflop is the basic building block of a register? Why? How many are needed to build an n-bit register?**: The D flipflop because its value is constant unless the input changes, which is exactly what we need for computer memory (stable output). To build an n-bit register we need n D flipflops.

- **How many selector lines are needed for a 2^n word memory? What is a write enable bit? When is it on? How many flipflops are needed for a 2^n word memory where every word contains m bits?**: n selector lines. The write enable bit is either on or off, and determines if we change the value stored in a specific word of memory. It is on when we want to change the value of a word in memory. We need m flipflops

- **How many flipflops are needed to build an n-bit counter? What is the smallest number this counter can represent? The largest? What does the count enable line do?**: we need n flipflops. The smallest number is 0, the largest is $2^n - 1$. The count enable line tells us if we want to change the value of the counter on the current clock tick.