Introduction to Digital Electronics and Computer Design

JEE 2600

Washington University

Joint Engineering Program

Course Information

- Time: Monday & Wednesday, 4 – 5:30
- Place:
- Professor:
  - Sean Key
  - keys (at) seas (dot) wustl (dot) edu
  - No set office hours – Available by appointment. Also generally available the half hour before or after class.
- TA:
  - TBD
  - Office Hours TBD
- Texts:
  - Digital Design with RTL Design, VHDL and Verilog, 2nd Ed. by Frank Vahid
- Website:
  - classes.engineering.wustl.edu/jee2600
  - Syllabus available on the website.

Expectations

- Adhere to the Academic Integrity Policy
- Show up to every class
- Ask questions!
- Read the book!

Procedures

- Inform the professor before hand if you will miss class.
- If you miss class, you are still responsible for the material covered. Get the notes you missed from another student.
- No test make-ups will be allowed without a doctor's note.
- If you know you will not be able to make a test, make arrangements to take it before hand.
- Since I do not have set office hours, send me email if you have problems with homework or questions about the course.
- We will follow Washington University's snow schedule, not UMSL's. Call 314-550-5877 or 314-453-5555 with ID #1440 if you expect that class will be cancelled due to weather .

Grading

- Homework (10%)
- Quizzes (25%)
- Tests
  - Test 1 (15%)
  - Test 2 (25%)
  - Test 3 (25%)

Homework Policies

- Be sure to do your homework:
  - Worth one letter grade.
    - It might be the difference between an A and a B.
  - A grader will provide advice on missed problems.
  - Homework scores help me know if I need to cover some topics in more depth.
  - If your grade is “borderline”, it may help your grade if all assignments have been turned in.
- If you don’t do your homework, you will have trouble with quizzes and tests.
- Homework is due at the beginning of the lecture on the due date. Answers will be posted that night.
- Feel free to discuss assignments with other students, but turn in your own work.

Homework Format

- Pages must be stapled together.
  - Lost pages are not the grader's responsibility.
- Remove tattered edges.
- Write legibly
  - Illegible problems will not be graded
- When the answer is not a drawing or long text, circle it.
- Heading – top right corner
  - Name
  - JEE 2600
  - Date
- Submit problems in the order assigned
- Label each problem as it appears on the assignment page.

Quiz & Test Policies

- Quizzes
  - Quizzes will be given the last 10 minutes of the class after homework is due.
  - Quizzes will cover the homework, lecture and reading from the previous week.
  - Review homework answers before the quiz!
  - No retakes without a doctor's note. If you will miss a class, inform me before hand.

- Tests
  - 80 minutes long
  - Due exactly at 5:30, when I will leave the room. Any tests that aren't in my hand are a zero.
  - No retakes are allowed without a doctor's note!

- All work submitted in the class **MUST BE YOUR OWN** WORK. However, you may discuss problems with other students.

Topics

- Digital Design Overview
  - Binary & Hexadecimal Numbers, Microprocessors versus Application Specific Circuits
- Combinational Logic Design
  - Boolean Algebra, Logic Gates & Other Components
- Sequential Logic Design
  - Latches, Flip-Flops, Finite State Machines & Controllers
- Complex Combinational Components
- Register Transfer Level Design
  - High Level State Machines
- Programmable Processors
  - Instruction Sets and Assembly Language
- Circuit Optimization
  - Speed versus Size Tradeoffs
- Physical Implementation of Circuits
  - ASICs versus FPGAs

Final Thought Before We Start

- This class does not have to be difficult
  - The topics covered aren't difficult
    - They do require practice!
  - However, we will cover a lot of material.
- Everyone can get an A.
  - Attend all lectures
  - Complete reading assignments
  - Do your homework
  - Study
  - Ask questions
  - Don't cheat!

Digital Design Basics

Number Systems

Number Systems

- In elementary school we learn to count 0 through 9 before adding a second digit to get 10. However, this isn't the only way to count.
- The number of different numbers before adding a second digit is known as the base, or radix.
- Numbers with more than one digit are positional codes
  - The position of the digit determines the weighting
  - i.e. $62.1 = 6 \times 10^{1} + 2 \times 10^{0} + 1 \times 10^{-1}$
- Lets look at some common number systems

Common Number Systems

- Decimal
  - The number system we are most familiar with
  - Base: 10.
  - Digits: 0 – 9
- Binary
  - Base: 2
  - Digits: 0 – 1
- Octal
  - Base: 8
  - Digits: 0 – 7
- Hexadecimal
  - Base: 16
  - Digits: 0 – 9, A – F

Binary

- Because digital circuits operate with values that represent “on” or “off”, binary is a natural fit for digital systems.
- If we need to represent a number larger than 1, we add more bits to the left (binary is a positional code).

Decimal	Binary
0	00
1	01
2	10
3	11

Bad Joke of the Day

- There are 10 kinds of people in the world … those how know binary and those who don’t.

Bits & Bytes

- What is a bit?
  - A single binary digit is known as a bit
  - Represented by a lower case ‘b’
- So, then, what is a byte?
  - A byte is simply a grouping of 8 bits
  - Represented by an upper case B
- Half of a byte is known as a nibble.

- The right most bit is the least significant bit (LSB) because is has the lowest weight
- The left most is the most significant bit (MSB)

Size Prefixes

- There are many prefixes that can be appended to the words ‘bit’ or ‘byte’ that modify the quantity of bytes that are represented
  - Kilo-
    - Generally, this means 1000, but since Bytes are power of two, the closest we can get is 1024 (2^{10})
    - Means 1024 bits or bytes
  - Mega-
    - 1024 times larger than Kilo-
  - Giga-
    - 1024 times larger than Mega-
  - Tera-
    - 1024 times larger than Tera-
  - Example: 2 MB
    - 2 * 1024 * 1024 * 8 = 16777216 bits
  - Note: Hard drives do actually stick to the 1000 multiple!

Conversion: Binary to Decimal

- Multiply each bit by the weight of its position
  - i.e. $10_2 = 1 * 2^1 + 0 * 2^0 = 1 * 2 + 0 * 1 = 2$
- Example 1: convert 1101_2 to decimal
  - Answer: 13_{10}
- Example 2: convert 10001_2 to decimal
  - Answer: 17_{10}

Octal

- Octal is a number system that used to be heavily used, but is not as common today.
- Binary numbers are easily simplified into Octal by grouping three bits together
  - Binary Octal
    - 000 0_8
    - 011 3_8
    - 111 7_8
  - See the handout provided in class for full chart of hex values.

Conversion: octal to binary

- Use direct substitution
  - You will want to memorize the binary values for 0 through 7
- Example 1: 43_8 to binary
  - Answer: 100 011
- Example 2: $110\ 010\ 111_2$
  - Answer: 627_8

Hexadecimal

- Hexadecimal is another number system that easily converts from binary, but with groups of 4 bits.
  - Represents 16 values
  - Uses letters A – F to represent numbers from 10 through 15, respectively.
Decimal	Binary	Hexadecimal
1	0001	0x1 or 1_{16}
9	1001	0x9 or 9_{16}
11	1011	0xB or B_{16}
  - See the handout provided in class for full chart of hex values.
- Hexadecimal is useful for representing nibbles and bytes.

Conversion: Hex to binary

- Use direct substitution
  - You will want to memorize the binary values for 0 through F
- Example 1: $F2_{16}$
  - Answer: 1111 0010

Conversion: Division Method

- A method of converting from one number system to another
  1. Divide by the new radix
  2. Record the remainder
  3. If the quotient is zero, go to step 5,
     otherwise divide the quotient by the radix.
  4. Go back to step 2
  5. Assemble the remainders into a number, last first.

- Example : Convert 18_{10} to hex

18/16 = 1 rem 2
1/16 = 0 rem 1

→ 0x12

Division Method Examples

- Example 1: Convert 10_{10} to binary
  - Answer: 1010_2

- Example 2: Convert 17_{10} to hex
  - Answer: 11_{16}

Common Number Systems Summary

Decimal (radix = 10)	Binary (radix = 2)	Octal (radix = 8)	Hexadecimal (radix = 16)
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Memorize these simple conversions

Number Ranges

- The range of numbers that can be represented based on the number of digits available in the number system.
  - Hardware systems always have a size limit on the number of digits.
- Familiar Example: How many discrete values can be represented by three decimal digits?
  - 000 to 999 → 1000 discrete values.

Number Range Example

- Example: How many discrete values can be represented with 8 bits?
- Lets look at two ways to solve this:
  1. We can figure out the values of the maximum and minimum numbers

     Min → All zeros 0000 0000
     Max → All ones 1111 1111
     $2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 + 2^6 + 2^7 = 255$

     So we have 256 values (from 0 to 255)

  2. We can raise the radix to the power of the number of bits.

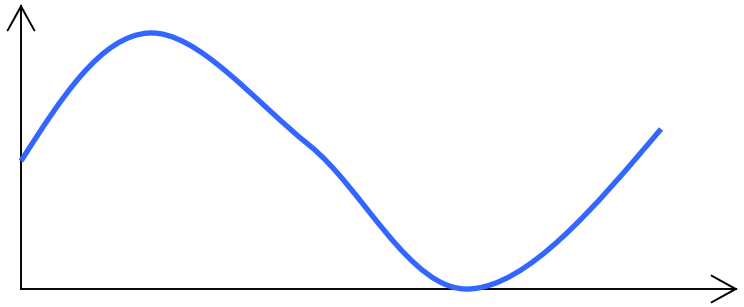
     $2^8 = 256$ (from 0 to 255)

Digital Design Basics

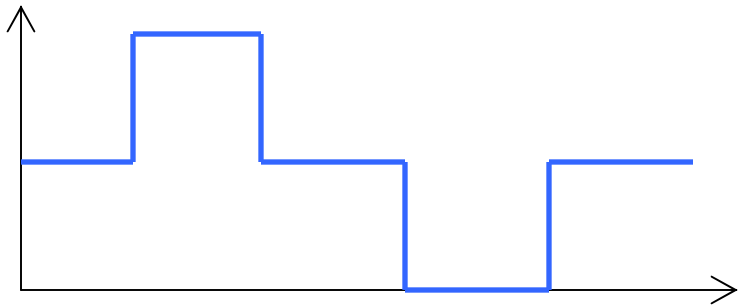
Encoding Signals

What is a Digital System?

- First, what is a signal?
  - A physical phenomenon that has a value at every instant of time.
- Analog Signal
  - Aka. Continuous Signal
  - Can have an infinite number of values
- Digital Signal
  - Aka. Discrete Signal
  - Can have only one of a finite set of discrete values
  - Simplest example → 1 or 0, ON or OFF
    - This is known as binary
- Digital signals are just an abstraction of analog signals
  - Signals themselves are still actually analog, but encoded in a digital fashion.



Analog



Digital

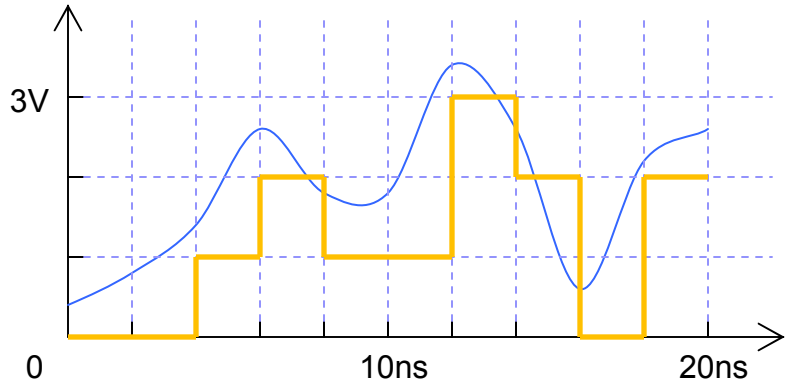
Encoding Analog Signals

- The real world can be sampled and digitized for use in digital systems
  - Sensors are used to measure the physical world
  - Circuits known as analog to digital converters convert the electrical signal generated by the sensor into a digital representation.
    - Some data may be lost due to the digitization.
  - Digital to analog converters can reverse this process
    - They cannot reverse the lost data, though.

- I.E. cell phone
  - Microphone measures your voice and generates an analog signal
  - The analog representation of your voice is digitized and sent to another phone.
  - The digital signal is converter back to analog and placed through a speaker to reconstruct your voice.

Binary Encoding example

- Sample the following 0V to 3V analog signal using 2 bits every 2 ms.
  - 0V → 00
  - 1V → 01
  - 2V → 10
  - 3V → 11



Other types of encodings

- ASCII
  - A binary representation of 128 characters (letters and symbols) using 7 bits
  - See page 10 for a sample of ASCII encodings.
- Gray Code
  - Binary sequence where only one digit changes at a time.
  - Will be covered more in depth in a lab exercise
- Binary Coded Decimal (BCD)
  - 4-bit binary number that stops at 9 to more directly represent binary.

Implementation of Digital Systems

- There are two main methods of implementing a digital design
  - Microprocessors
    - Software driven design
    - Circuitry may be overly complex or high powered for the problem being solved
    - May not be able to make timing requirements
    - Cheap to buy COTS microprocessors
  - Custom digital circuit
    - Circuit designed for specific functionality
    - Operations can be done concurrently to meet timing requirements
    - More expensive to implement.
  - Combination of both methods
- This class focuses on application specific circuits, but we will discuss a simple microprocessor architecture.

Combinational Logic

Switches & Logic Gates

Logic Design

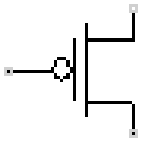
- Logic design is classified into two types:
  - Combinational
    - A circuit whose outputs are a direct function of it's inputs.
  - Sequential
    - A circuit whose output depends not only on the current inputs, but also an internal state.
      - State is effected by past inputs.
- In logic design, a circuit can be through of as a series of switches that implement a desired function based on the value of the inputs
  - i.e. an input indicates ON or OFF (0 or 1).

Switches

- Switches in digital circuits have been implemented with many different technologies.
  - 1930's – Relays
    - An input voltage creates and electromagnet and moves a physical switch
  - 1940's – Vacuum Tubes
    - Somewhat like a light bulb, but only conducted when a third input had the correct voltage applied
  - 1950's – Discrete Transistors
    - Silicon based bi-polar junction transistor in it's own packaging.
  - 1960's through today – Integrated Circuits
    - Many silicon based transistors on a single chip
    - BJTs eventually gave way to MOSFETs

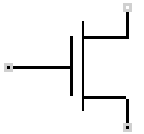
MOSFETs

- Modern digital circuits use combinations of different types of MOSFETs as switches
  - Metal Oxide Semiconductor Field Effect Transistor
  - We won't cover MOSFETs in much detail, just how they are utilized in digital circuits.



pMOS

- Two types of MOSFETs
  - P-type
    - Off with a high voltage (logic 1)
    - On (saturation) with a low voltage (logic 0)
  - S N-type
    - Off with a low voltage (logic 0)
    - On with a high voltage (logic 1)



nMOS

CMOS

- Complimentary MOS
  - A circuit technology that employs both N and P-type MOSFETs
  - Generally made up of small circuit units called ***gates*** that implement a specific function.
  - Only one type of MOSFET (N or P) is on at a time in a gate during static operation.

- Benefits of CMOS
  - Low static power consumption
    - Due to the fact that only one type of MOSFET is on at a time
      - No current path during static operation.
      - Current only flows when switching.
  - High noise immunity
    - Small amounts of noise on the input have little effect on the output.

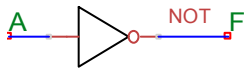
Logic Gates

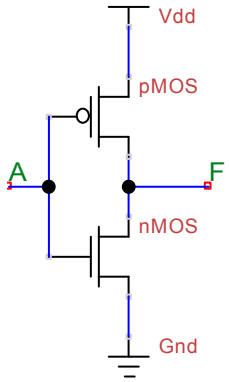
- It is difficult to design a system using a transistor as a switch, so lets use transistors to create building blocks that are easier to work with.
- Logic gates are small circuits of switches that are the basic building blocks used in digital design.
- Standard gate types:
  - NOT (Inverter)
  - Buffer
  - AND
  - NAND
  - OR
  - NOR
  - XOR
  - XNOR

Inverter Gate

$F = \neg A = A' = \overline{A}$

A	F
0	1
1	0

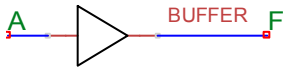


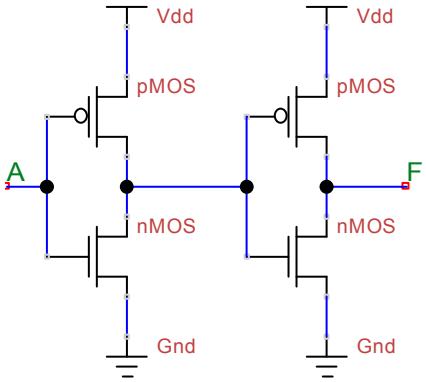


Buffer

F = A

A	F
0	0
1	1





AND Gate

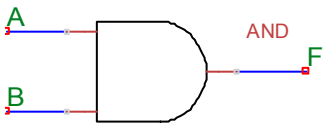
$F = A \cdot B$

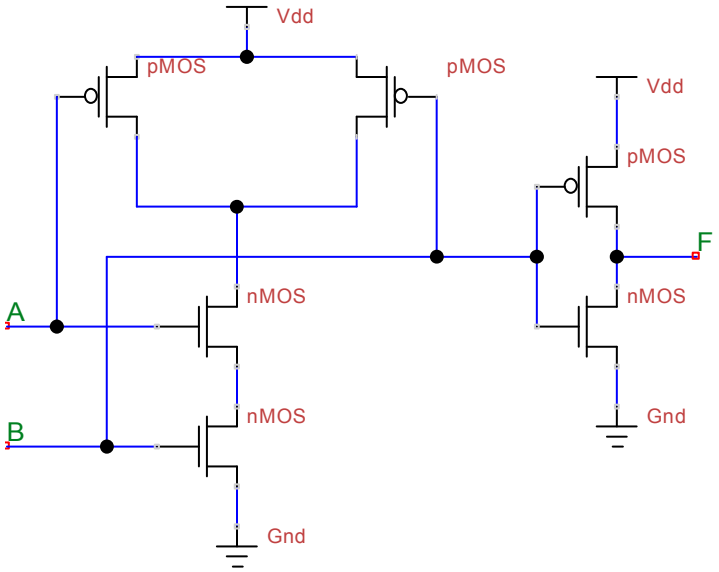
$= A * B$

$= A \wedge B$

$= AB$

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1





Note: AND gates can have more than two inputs.

NAND Gate

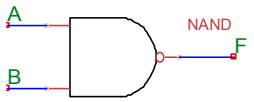
F = (A · B)’

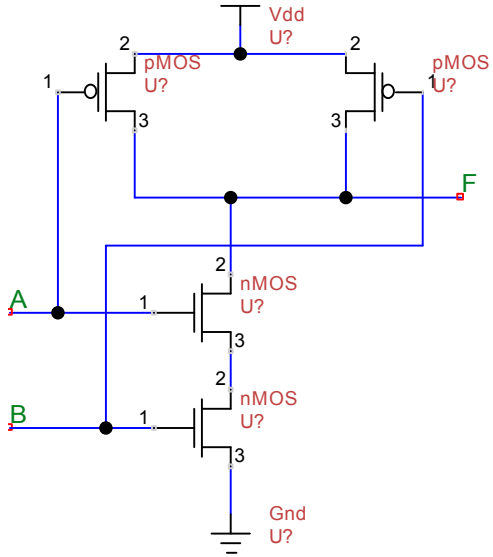
=(A * B)’

= (A ∧ B)’

= (AB)’

A	B	F
0	0	1
0	1	1
1	0	1
1	1	0



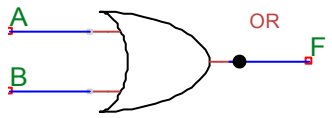


Note: AND gates can have more than two inputs.

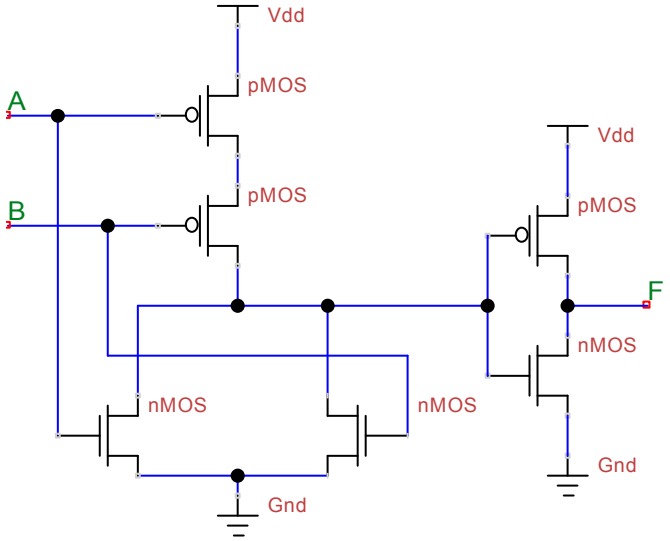
OR Gate

F = A + B

= A v B



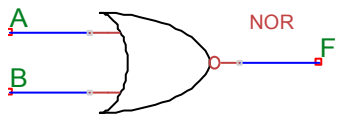
A	B	F
0	0	0
0	1	1
1	0	1
1	1	1



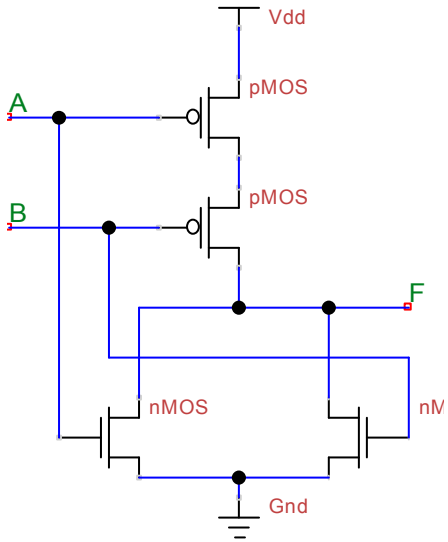
NOR Gate

F = (A + B)’

= (A ∨ B)’

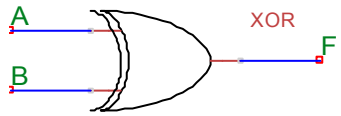


A	B	F
0	0	1
0	1	0
1	0	0
1	1	0



XOR Gate

$F = A \oplus B$

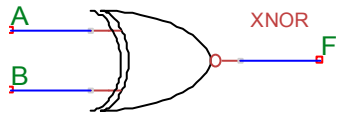


A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

The circuit design is left as an exercise for the student.

XNOR Gate

$F = (A \oplus B)'$



A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

Completeness of Gates

- Using AND, OR and NOT or NAND, NOR and NOT gates, you can create any digital function.
  - The negated versions of functions are generally used in real circuits because they require fewer transistors.
- Any set of gates that can implement any function is complete.
  - We don't actually need all 3 types.
  - A single NAND gate is a complete set because NANDs can be wired to implement NOR and NOT gates.

Your First Circuit

- Design a digital circuit that turns on a light when a user throws a switch or the lights are off and motion is detected.
  - Assume boolean inputs light_switch and motion_detected
  - Light = light_switch
    + (light_switch' · motion_detected)

Note: The diagram we drew is known as a schematic