Many ways to build logic out of MOSFETs

pass transistor logic (most similar to the first switch logic we saw)

dynamic CMOS logic
- Clock=0 *precharges* X through the pMOS
- clock=1 gives the two nMOS time to drain charge on X to ground (if A and B turn them on)
Switches other than transistors

- relays (electro-mechanical)
- diodes
- OR gate

Wikimedia commons
Field-programmable gate array (FPGA)

an alternative to manufacturing your own silicon chip

Look up table (LUT) – a **programmable truth table** in hardware!

“3-LUT” means 3 inputs (so must program 8 rows)

https://en.wikipedia.org/wiki/Field-programmableGateArray
CS 2630
Computer Organization

Combinational logic for arithmetic
Brandon Myers
University of Iowa
Where we are going

Instruction set architecture (e.g., MIPS)

Compiler
- translating source code (C or Java)
- Programs to assembly language
- And linking your code to Library code

Memory system

Processor

I/O system

Datapath & Control

Digital logic

How the software talks To the hardware

How a processor runs MIPS Programs!

How switches (1 or 0) can be used to build Interesting functions: from integer arithmetic to programmable computers
Ingredients for a processor

• Combinational logic
  • arithmetic and bitwise operations
  • multiplexors (pick 1 out of N inputs)
  • control logic
  • branch calculation
  • address calculation

• Synchronous logic (we’ll get to these later)
  • registers
  • program counter, or PC
  • data memory
  • instruction memory

start learning how to build these combinational logic circuits
How do we add two numbers

addu $t0, $t1, $t2

We know: The numbers are physically stored in registers as bits.

The sum of two numbers is a function we can implement using combinational logic.
Start with a 1-bit adder

inputs: carry in, A, B

Even numbered tables; write the Sum truth table and equation

Odd numbered tables; write the Carry truth table and equation

outputs: carry out, sum
Peer instruction: 2-bit adder

Number of rows of truth table for Sum1 output of 2-bit adder?
How many rows in this truth table?

current state (the input)

4GB memory x 8bit/byte = 32 Gbits
32 registers x 4 Byte x 8bit/byte = 1Kbit
PC register x 4 Byte x 8bit/byte = 32 bits

new state (the output)

processor takes one step from current state to new state. How many rows in the truth table of bit 0?

https://commons.wikimedia.org/wiki/File:Performance_PIPER_die.JPG
CS 2630
Computer Organization
Adders, timing, multiplication
Brandon Myers
University of Iowa
How humans add two numbers

• compute $652_{10} + 367_{10}$ on paper

• compute $11010_2 + 01011_2$ on paper
Building larger circuits out of *components*: 2-bit adder built from 1-bit adders

individual components have their own small truth tables; then we just combine the components
32-bit adder

Reflect: We now have a circuit for A+B. What is a small change we can make to get A-B?
Under the rug of combinational logic: computation time
Propagation delay through logic gates

Resistor-capacitor (RC) model of NOT gate when given input 0

input changes

1→0 0→1

1→0 NOT 0→1
Charging a capacitor in an RC circuit

Propagation delay through logic gates

Resistor-capacitor (RC) model of NOT gate when given input 0

When the input changes from 1 to 0, there is a delay before the output of the NOT gate reaches 1

draining the capacitor when input goes 0→1
Back up to the level of logic gates

• In CS2630, when we think about delay, we’ll use the following simplified model
  • propagation delay is higher for more complex gates
  • propagation delay increases with the number of inputs to a gate
    • e.g., 2-input OR gate is faster than a 3-input OR gate
  • propagation delay of a sequence of gates is additive
  • all inputs must be available before logic gate begins computing

Example: assume delay of inverter is 0.5 ns
Practice

• label each wire with the time at which that signal is available
• The two inputs are available at time=0
• delays: AND gate = 2, OR gate = 2, NOT gate = 1
Delay of adder circuit

1-bit adder delay: sum=2
1-bit adder delay: carry=1

Peer instruction: delay to get Sum31?

All inputs (e.g., A0 B0 A1 B1...) are available at time=0
Delay of adder circuit

This adder is called “ripple carry adder”

1-bit adder delay: sum=2
1-bit adder delay: carry=1
Next ...

• We now know how one kind of adder works and how long it takes to compute the result
• Can we build faster adders? Yes! Use hierarchy/trees
Carry-select adder
Peer instruction

What is the delay from the time A and B are available to when Sum is available?

delays:
sum: 1
carry: 1
mux: 1

What is the delay from the time A and B are available to when Sum is available?
Administrivia

• Reminders:
  • DYB tonight 6-7pm dedicated to review
    • bring your prioritized list of specific problems or questions that you’d like to review
    • it will be most useful if you take the practice exam(s) first
  • Midterm on Monday 3/6: open note open book, no devices
Best case delay of adders? \( O(\log n) \)

- **p** means “propagate a carry”
- **g** means “generate a carry”

Important: \( p \) and \( g \) are not functions of \( c \)

8-bit Carry Look-ahead Adder

Best case delay of adders? $O(\log n)$

compute carries

8-bit Carry Look-ahead Adder

use carries to finish the sums

Multiplication

• Evaluate $501_{10} \times 422_{10}$ on paper

• Evaluate $1100_2 \times 1001_2$ on paper
Aside: useful logic circuit notation

### Splitter (2-bit to 2 1-bit)

- Inputs: 00, 01, 10, 11
- Outputs: 00, 01, 10, 11

### Splitter (2 1-bit to 2-bit)

- Inputs: 00, 01, 10, 11
- Outputs: 00, 01, 10, 11

### Same wire used twice

- Inputs: 00, 01, 10, 11
- Outputs: 00, 01, 10, 11

### Cannot attach 2 inputs!
Shifting by a constant

\[ B = A \ll \ll 1 \]
Variable shifter, similar design with 8-bit mux

Can you think of a design that avoids using a huge 8-bit x 8-choice mux?

E.g., using only 8-bit x 2-choice muxes and hardwired shifters?
Peer instruction

• Build a 2-bit multiplier (4-bit output)
• you may use any number of

• example test cases:
  • 01 * 01 = 0001
  • 10 * 01 = 0010
  • 10 * 11 = 0110
  • 11 * 11 = 1001