

EECS 151 Disc 1

Rahul Kumar (session 1)
Yukio Miyasaka (session 2)

About Me

Contents

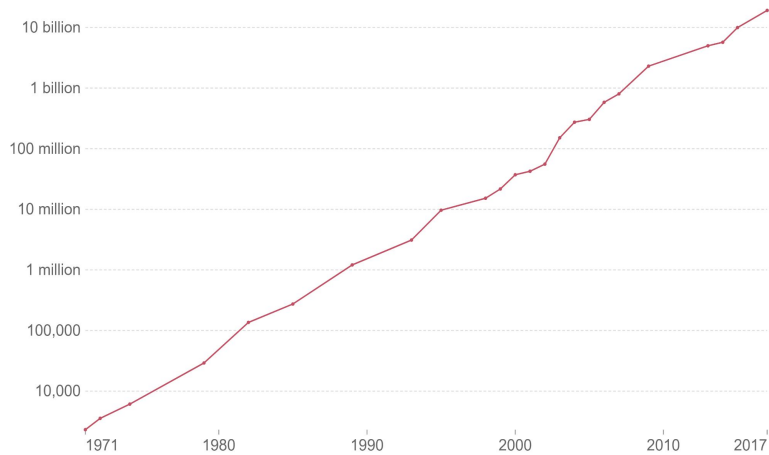
- Moore's law & Dennard scaling
- Pareto optimality
- Die cost
- Design alternatives (ASIC, FPGA, Processor)
- Logic circuits
- LTSpice

Moore's Law

Moore's law: The number of transistors per microprocessor

Number of transistors which fit into a microprocessor. The observation that the number of transistors on an integrated circuit doubles approximately every two years is called 'Moore's Law'.

Our World
in Data



Source: Karl Rupp. 40 Years of Microprocessor Trend Data.

CC BY

- #transistors on microchips doubles every two years (originally 1.5 years)
- Also serving as a milestone for device engineers
- This itself does not mean processor performance improvement ... why?

Dennard Scaling

It is **power** that limits the processor performance: fans can dissipate only 100W

Dennard scaling

- Transistor dimensions: $1/k$
 - Area: $1/k^2$
- **Voltage: $1/k$** (both V_{dd} and V_{th})
 - Delay: $1/k$
 - Power (per transistor): $1/k^2$
 - **Power / Area: 1 (constant)**
- Same circuit \rightarrow Same #transistor
 - Power / Circuit: $1/k^2$

End of Dennard Scaling

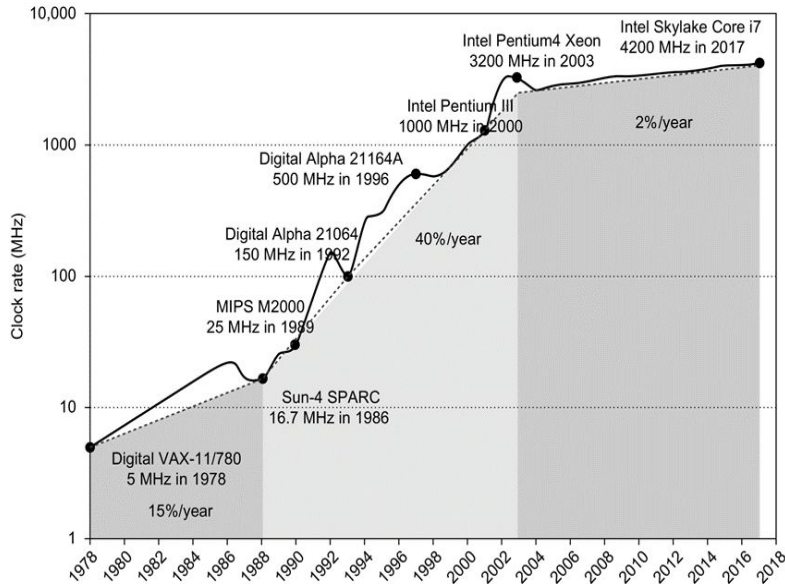
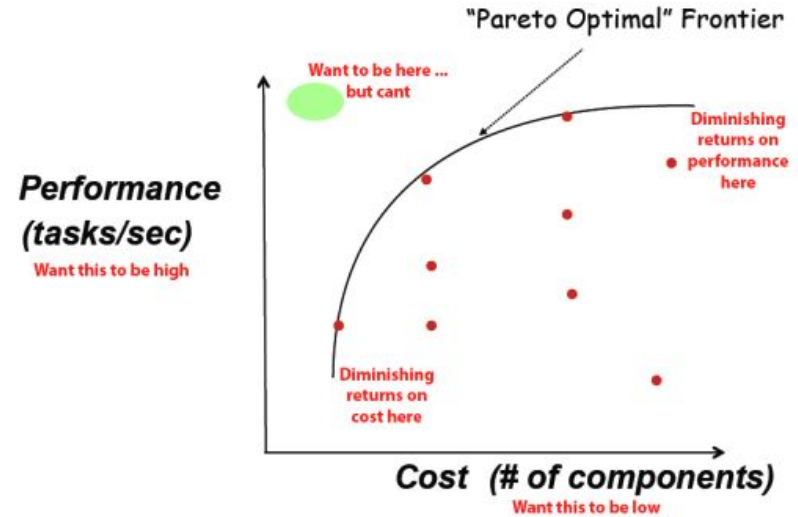


Figure 1.11 from J. Hennessy and D. Patterson, "Computer Architecture: A Quantitative Approach 6th edition"

- Leakage power
 - Power calculation was based only on switching power
 - We'll talk about this after midterm
- Cure: Decrease only V_{dd} (not V_{th})
 - Cancels frequency improvement!
- Countermeasure: Multi-core
 - High V_{dd} and high frequency when using only one core
 - Low V_{dd} and low frequency when using multiple cores
 - Through concurrent execution, multi-core can overcome the frequency decrease

Pareto Optimality

- IC design involves tradeoffs:
 - Performance, Power, Cost
- Performance is multi-dimensional:
 - Frequency
 - CPI (Cycles Per Instruction)
 - Number of cores
 - Functionality (AI accelerator?)

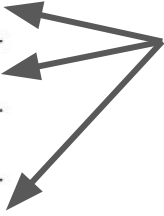


One is not pareto optimal if another is superior (or equivalent) in all criteria

Example: Pareto optimality

f_{max}	Energy	Cost
2.0	20	2.0
1.5	10	1.5
1.5	17	1.5
1.5	20	1.0
1.0	10	1.5
1.0	20	1.0

Compare



Example: Pareto optimality

f_{max}	Energy	Cost
2.0	20	2.0
1.5	10	1.5
1.5	17	1.5
1.5	20	1.0
1.0	10	1.5
1.0	20	1.0

Compare

Wasting energy!

Slower for no reason!

Example: Pareto optimality

f_{max}	Energy	Cost
2.0	20	2.0
1.5	10	1.5
1.5	17	1.5
1.5	20	1.0
1.0	10	1.5
1.0	20	1.0

Die Cost

$$\text{die yield} = \left(1 + \frac{\text{defects per unit area} \times \text{die area}}{\alpha} \right)^{-\alpha}$$

$$\text{Dies per wafer} = \frac{\pi \times (\text{wafer diameter}/2)^2}{\text{die area}} - \frac{\pi \times \text{wafer diameter}}{\sqrt{2} \times \text{die area}}$$

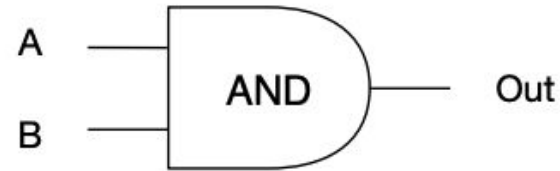
$$\text{Die cost} = \frac{\text{Wafer cost}}{\text{Dies per wafer} \times \text{Die yield}}$$

Logic Circuits

Combinational Logic

- Output is a pure function of inputs
- If an input changes, output changes (almost) immediately
 - We'll cover delay in future

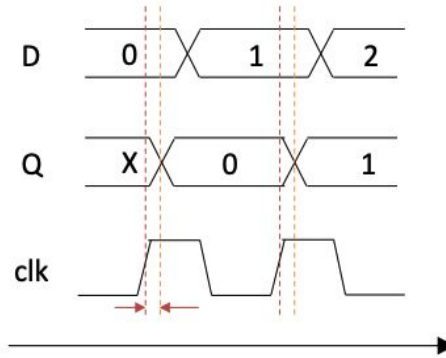
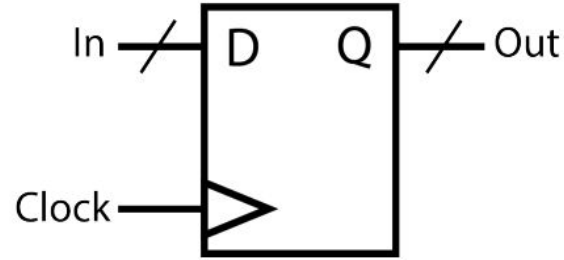
- Represented by:
 - Logic gates
 - Truth table
 - Boolean expressions (later week)



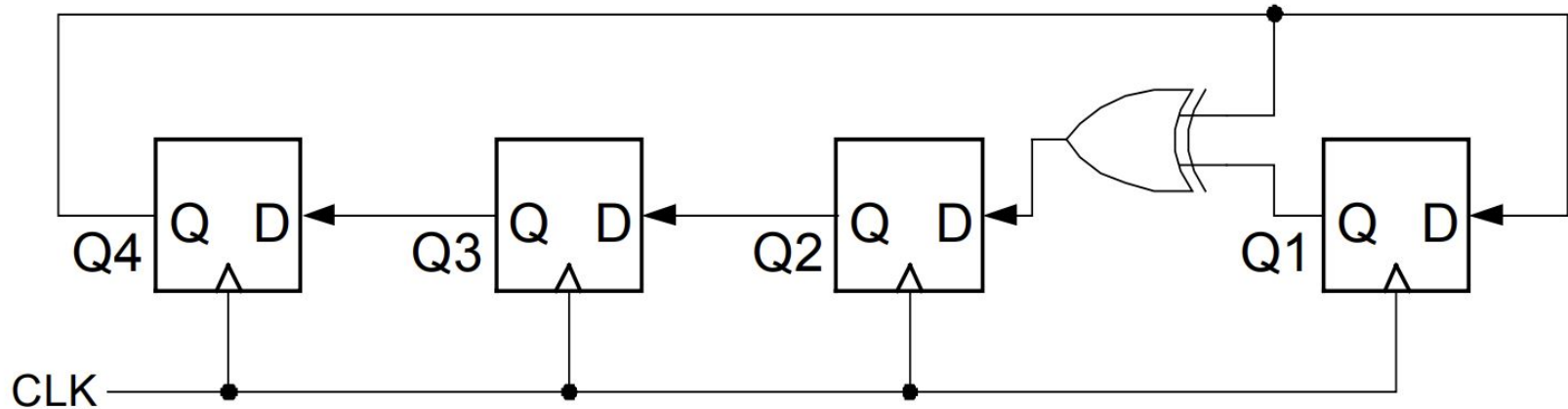
A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

Sequential Elements

- Registers/memories
- Remembers a state
- Clock tells when to read inputs and update outputs

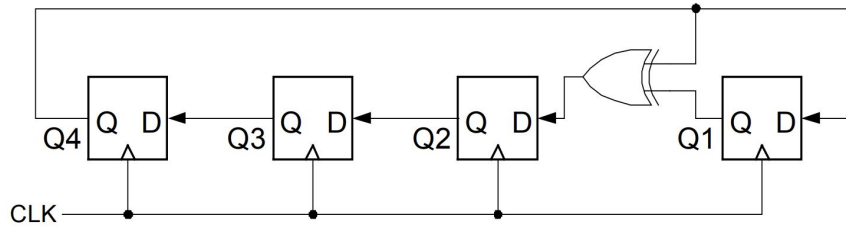


Example: Linear-Feedback Shift Register



- Write down a truth table from reg outputs to reg inputs
 - How many rows do we need?
- Simulate 5 cycles from $(Q1, Q2, Q3, Q4) = (1, 0, 0, 0)$

Truth Table



- 4 inputs $\rightarrow 2^4 = 16$ rows
- Except D2, $D_i = Q_{i-1 \bmod 4}$
- $D2 = \text{XOR}(Q1, Q4)$

Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	0	0	1
1	0	1	0	0	1	1	1
1	0	1	1	0	1	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	0	0	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1

Simulating LFSR

Cycle	Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	1				
1								
2								
3								
4								
5								

Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	0	0	1
1	0	1	0	0	1	1	1
1	0	1	1	0	1	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	0	0	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1

Simulating LFSR

Cycle	Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	1	0	0	1	0
1								
2								
3								
4								
5								

- Copy the corresponding row

Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	0	0	1
1	0	1	0	0	1	1	1
1	0	1	1	0	1	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	0	0	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1

Simulating LFSR

Cycle	Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	1	0	0	1	0
1	0	0	1	0				
2								
3								
4								
5								

- Copy the corresponding row
- Copy D's to the next Q's

Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	0	0	1
1	0	1	0	0	1	1	1
1	0	1	1	0	1	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	0	0	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1

Simulating LFSR

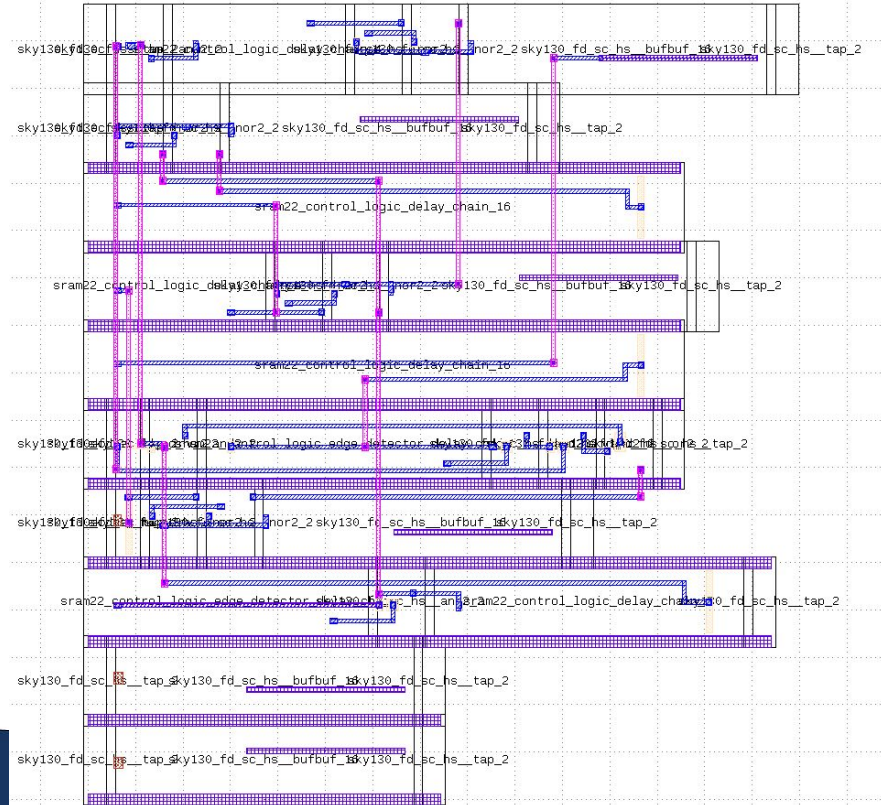
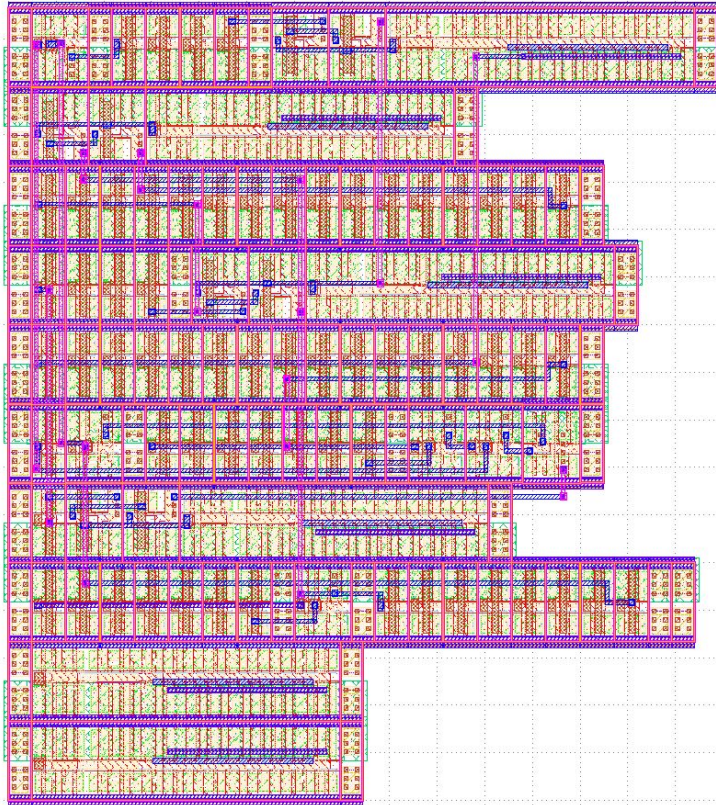
Cycle	Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	1	0	0	1	0
1	0	0	1	0	0	1	0	0
2	0	1	0	0	1	0	0	0
3	1	0	0	0	0	0	1	1
4	0	0	1	1	0	1	1	0
5	0	1	1	0	1	1	0	0

- Copy the corresponding row
- Copy D's to the next Q's

Q4	Q3	Q2	Q1	D4	D3	D2	D1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	0	0	1
1	0	1	0	0	1	1	1
1	0	1	1	0	1	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	0	0	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1

Circuit Implementation

Standard cell ASICs (most popular)



Standard cell ASICs

- EDA tools convert your design (eg. Verilog) to a layout (GDS)
 - Logic synthesis
 - Realize your design using library cells provided by the manufacturer
 - Cells are logic gates, registers, latches, etc.
 - Placement & Routing
 - Place the cells on the regular grid and connect them
 - Also deals with electrical properties such as capacity
- Other things that aren't called standard cells, but you might import as hard macros:
 - Memory blocks (eg. SRAM)
 - ESD protection
 - I/O drivers

Full-custom ASICs

- Actually, you can manually layout every transistor
 - Macro blocks are designed in this way
- But think how many transistors are used in modern processors
 - Can you manually handle one billion transistors?
 - Possible only when it is very simple or highly regular
- Your manual layout might cause electrical issues
 - Transistors and wires interact with each other electrically
 - Standard cells are carefully tested by the manufacturer so that tools can layout them in a safer manner

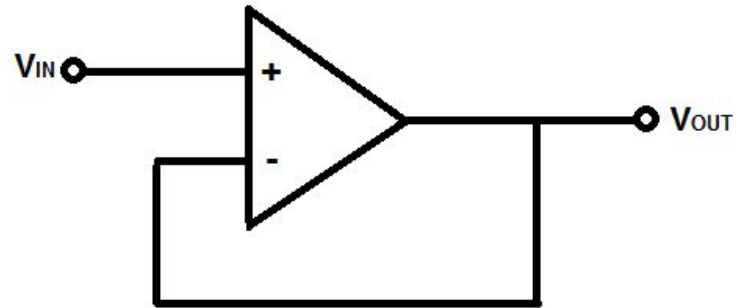
Design alternatives

- ASICs are very expensive to develop:
 - Each die is cheap (\$10) but the mask costs a lot (\$100K~\$1M)
 - Long verification before creating a mask, logically and electrically
- FPGAs
 - Available on the market (\$100~\$1K)
 - Compile, implement, and test in a day (up to a week for large designs)
 - However,
 - Lower performance and energy-efficiency (sacrificed for configurability)
 - Each die costs more than ASIC
- Processors (Software)
 - Sequential execution (easy to design)
 - Communication between functional units, register, and memory
 - Even lower performance and energy-efficiency

Electrical Properties of Transistor

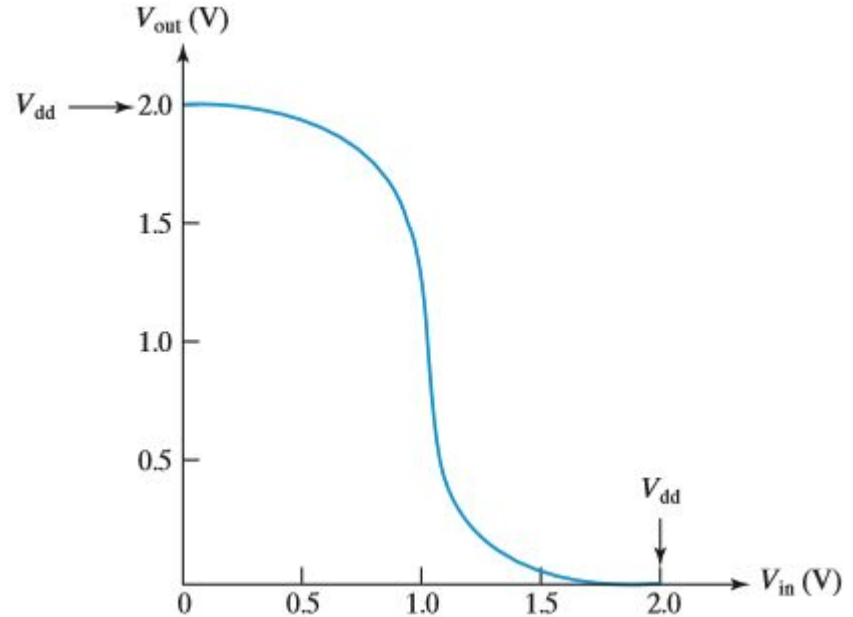
Regenerative Gates

- Lots of noise/interference in large digital system
- Digital logic must be robust to noise/interference
- Why might this be a bad digital buffer?



Regenerative Gates

- Digital gates should be regenerative
- Small variations in inputs should be suppressed
- How can you tell if noise is being suppressed? Look at voltage transfer characteristic (VTC).



SPICE

- Simulation Program with Integrated Circuit Emphasis
- Originally developed at Berkeley
- Many commercial and open source implementations:
 - Hspice, Ngspice, Spectre, LTspice
- Heavily used in analog design
- Used in designing/characterizing standard cells
 - “What is the delay of this inverter?”
- Not used for large digital circuits – too slow!

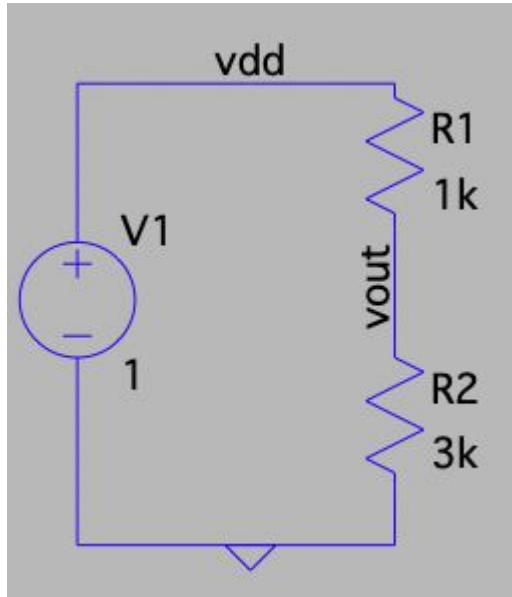
Schematic Entry

Need to tell the simulator about your circuit. Two options:

- Draw it in a GUI
- Write it in text format

Schematic Entry

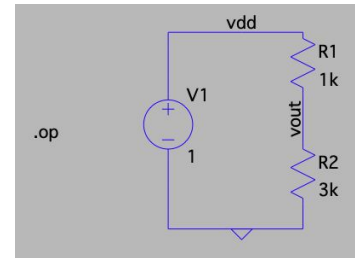
-



```
R1 vdd vout 1k
R2 vout 0 3k
Vdd vdd 0 dc 1
```

Simulation Commands

- Tells the simulator what to simulate
 - op: find the DC operating point
 - dc: sweep a source, find the DC operating point at each value of the source
 - tran: time domain simulation; no linearization
 - Mostly relevant to analog design: ac, noise, pss, qpss, disto, etc.
- Add to your schematic



.op

.op

Viewing Output

- Run the simulation, then plot the output.
- Or, from command line: invoke simulator and print output



```
Title: * voltage divider
Date: Thu Jan 19 14:33:45 2023
Plotname: Operating Point
Flags: real
No. Variables: 3
No. Points: 1
Variables:
    0      v(vdd)  voltage
    1      v(vout) voltage
    2      i(vdd)  current
Values:
0          1.0000000000000000e+00
          7.5000000000000000e-01
          -2.5000000000000000e-04
```