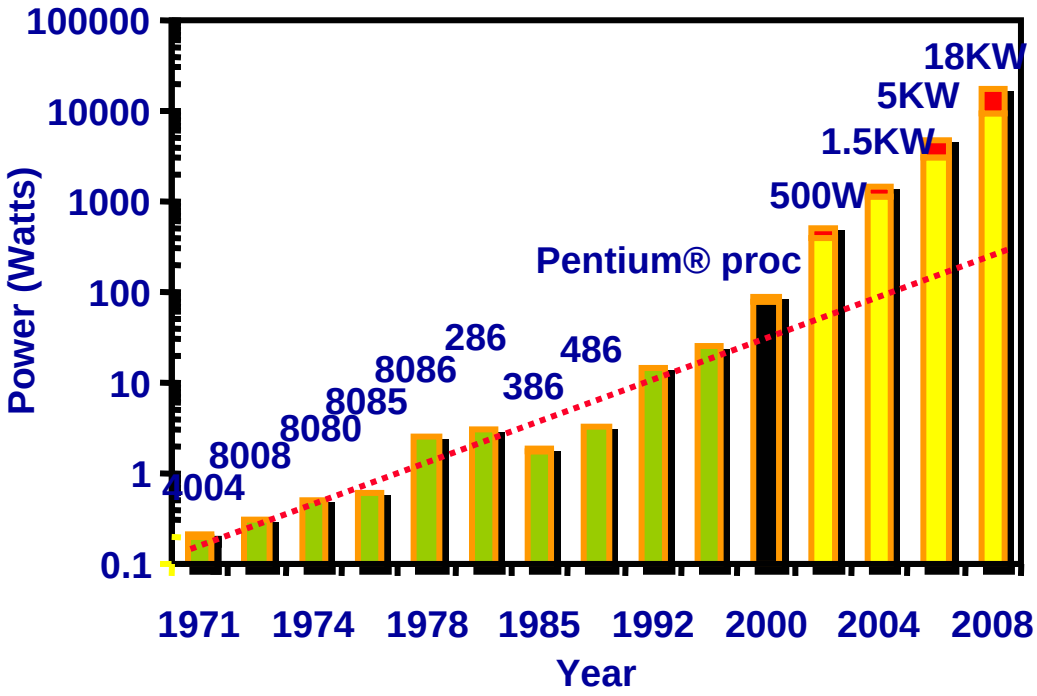
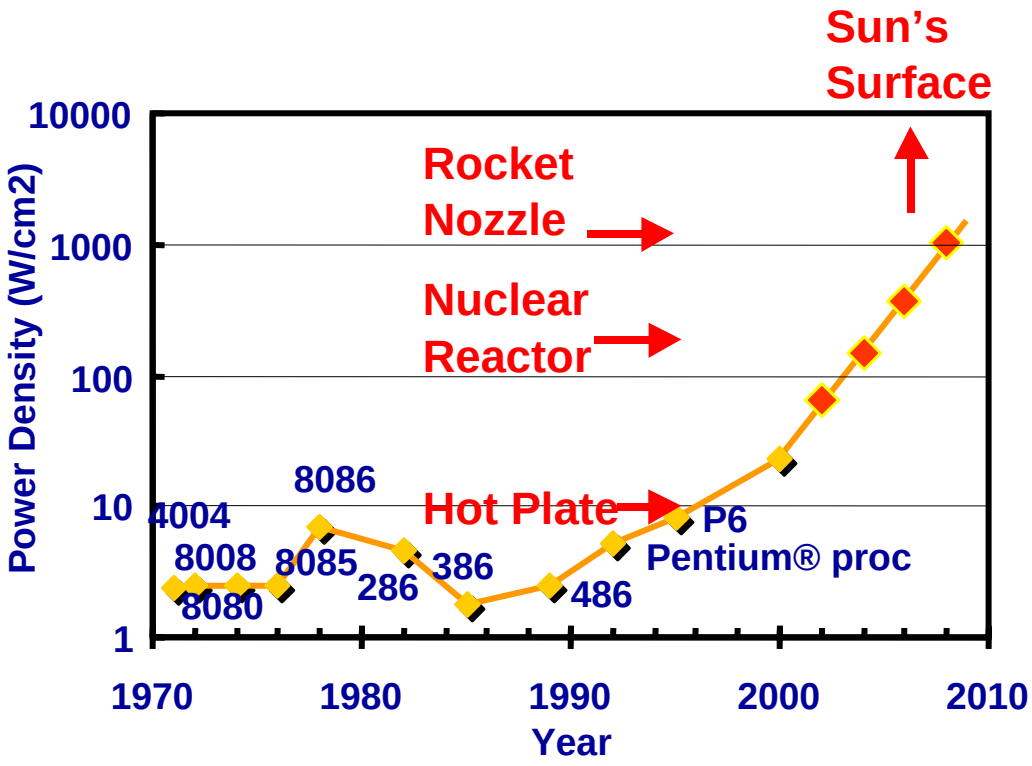



L16: Power Dissipation in Digital Systems

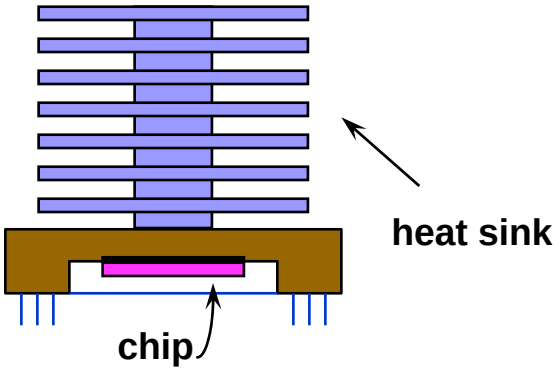
Problem #1: Power Dissipation/Heat



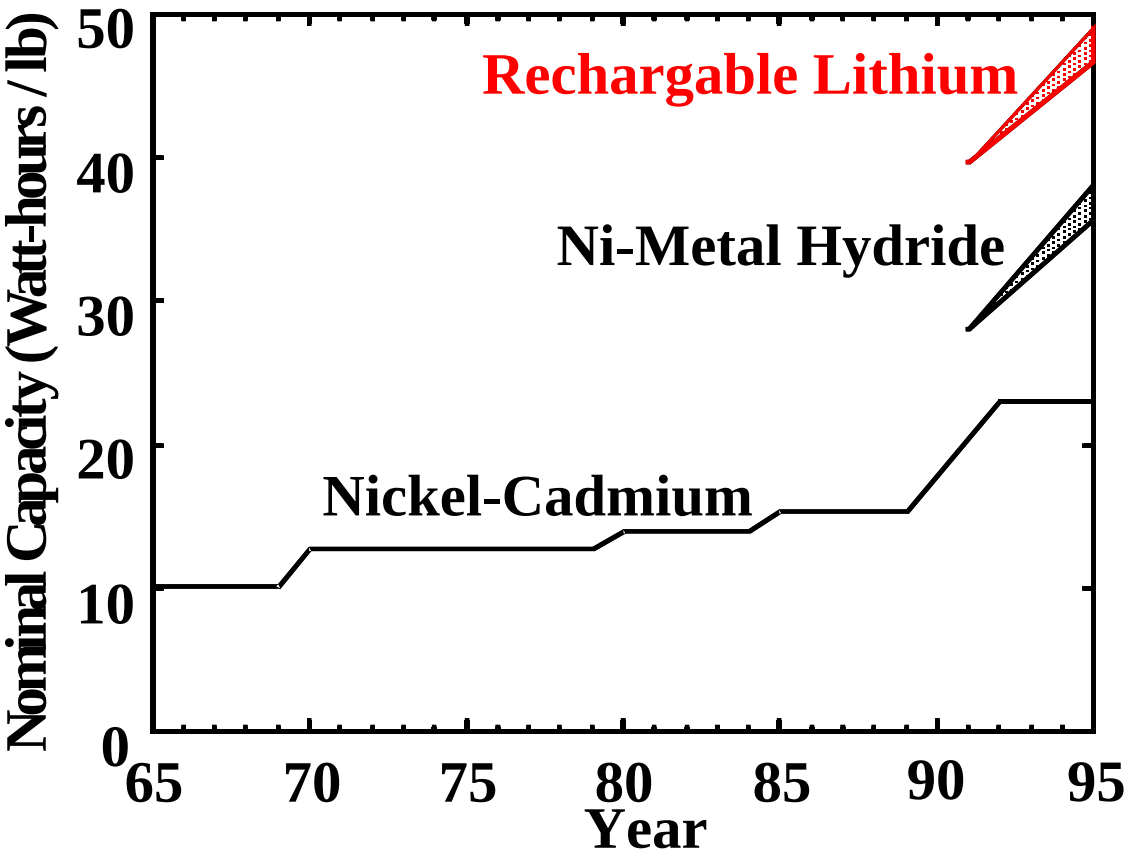


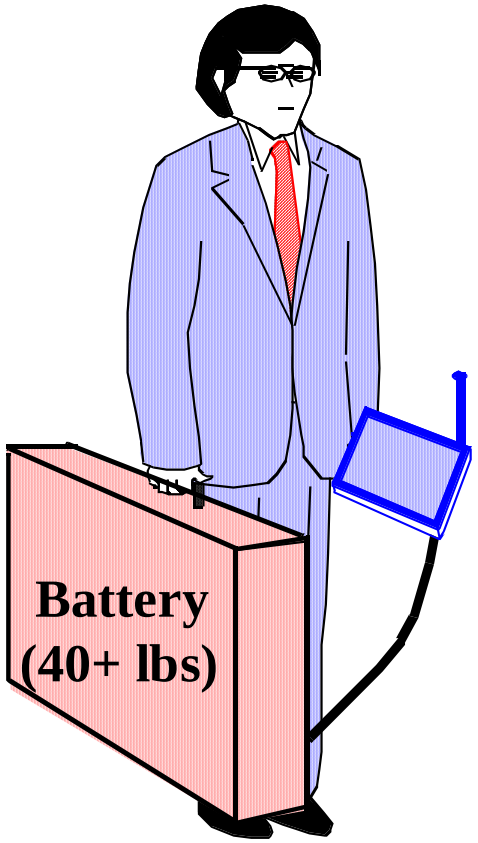
Courtesy Intel (S. Borkar)

How do you cool these chips??



Problem #2: Energy Consumption





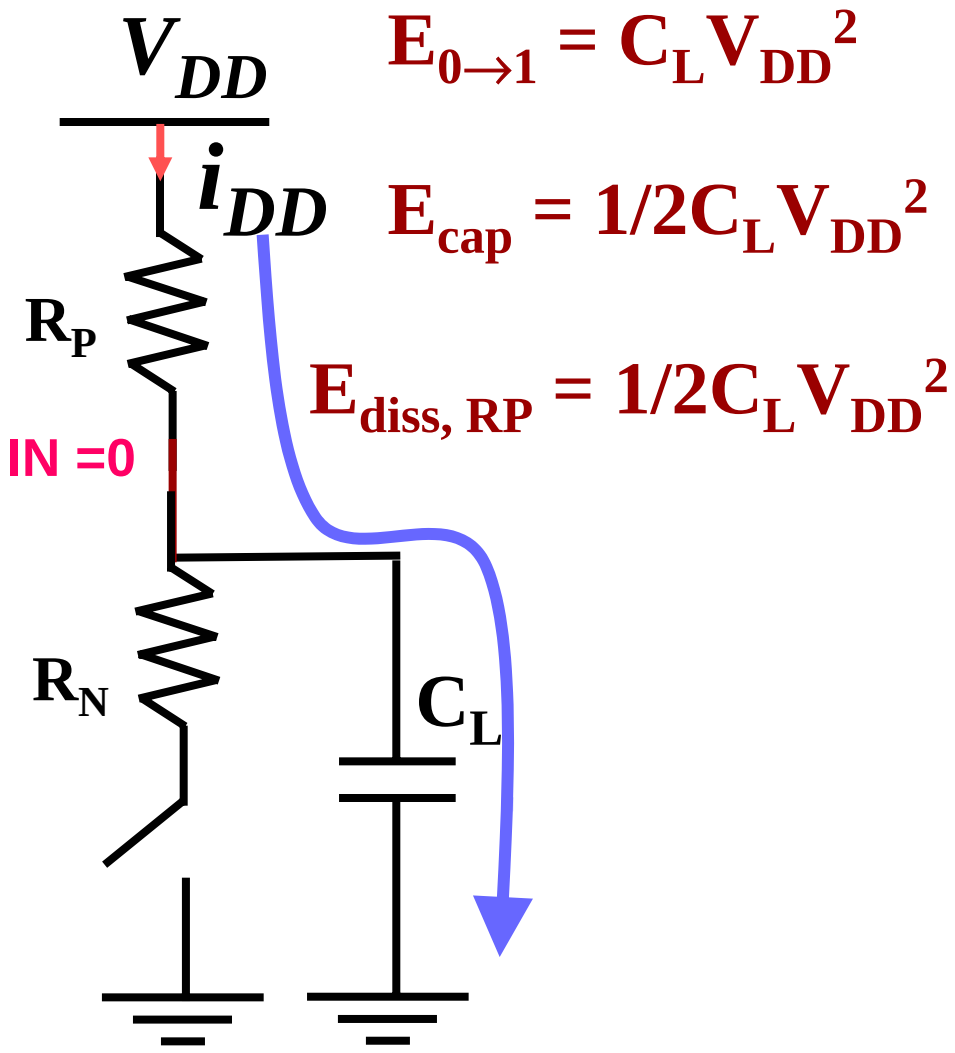
(from Jon Eager, Gates Inc. , S. Watanabe, Sony Inc.)

No Moore's law for batteries…

Today: Understand where power goes and ways to manage it

Dynamic Energy Dissipation

Charging

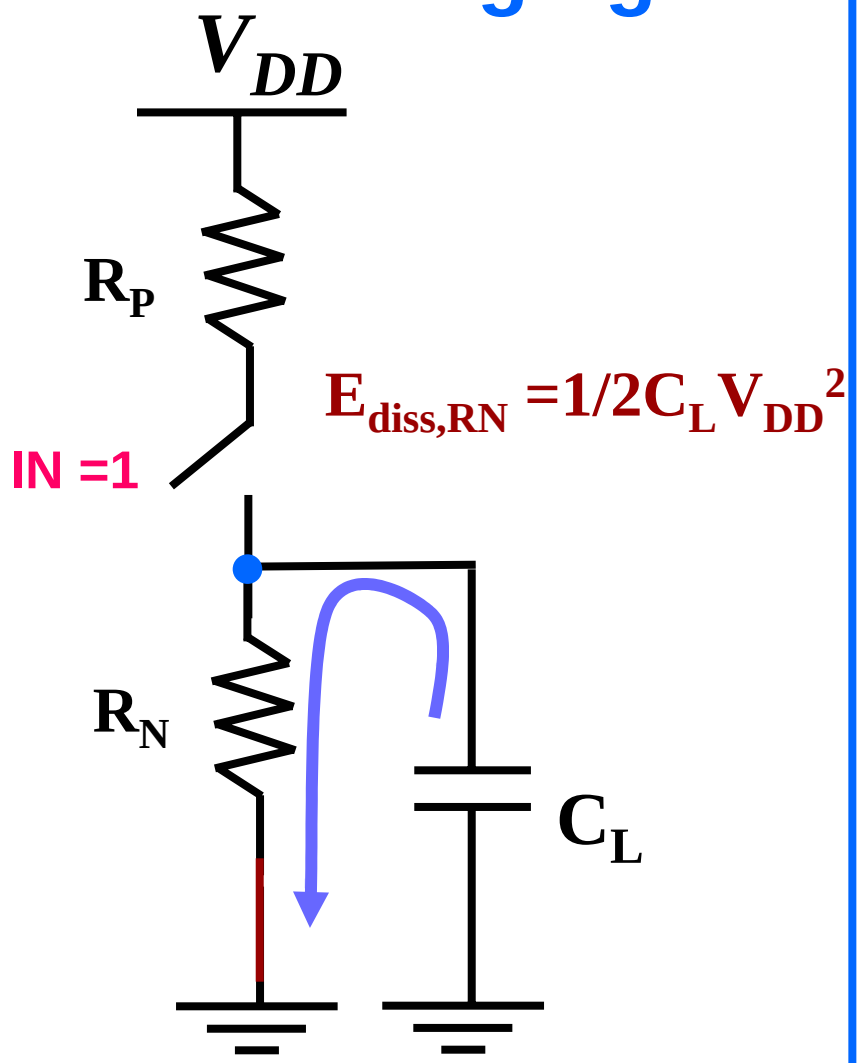


$E_{0\rightarrow1} = C_L V_{DD}^2$

$E_{cap} = 1/2 C_L V_{DD}^2$

$E_{diss,\ RP} = 1/2 C_L V_{DD}^2$

Discharging

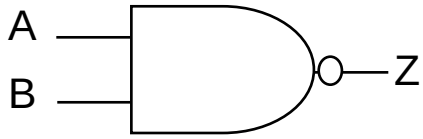


$E_{diss,RN} = 1/2 C_L V_{DD}^2$

$$P = C_L V_{DD}^2 f_{clk}$$

Transition Activity Factor $\alpha_{0->1}$

Current Input	Next Input	Output Transition
00	00	1 –> 1
00	01	1 –> 1
00	10	1 –> 1
00	11	1 –> 0
01	00	1 –> 1
01	01	1 –> 1
01	10	1 –> 1
01	11	1 –> 0
10	00	1 –> 1
10	01	1 –> 1
10	10	1 –> 1
10	11	1 –> 0
11	00	0 –> 1
11	01	0 –> 1
11	10	0 –> 1
11	11	0 –> 0



Assume inputs (A,B) arrive at *f* and are uniformly distributed

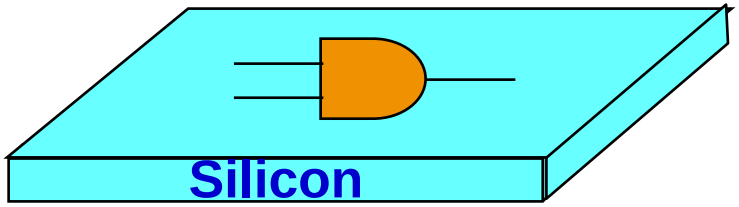
What is the average power dissipation?

$$\alpha_{0->1} = 3/16$$

$$P = \alpha_{0->1} C_L V_{DD}^2 f$$

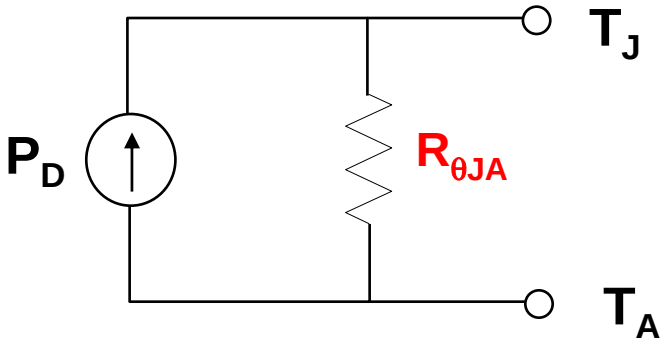
Junction (Silicon) Temperature

Simple Scenario



$$T_j - T_a = R_{\theta JA} P_D$$

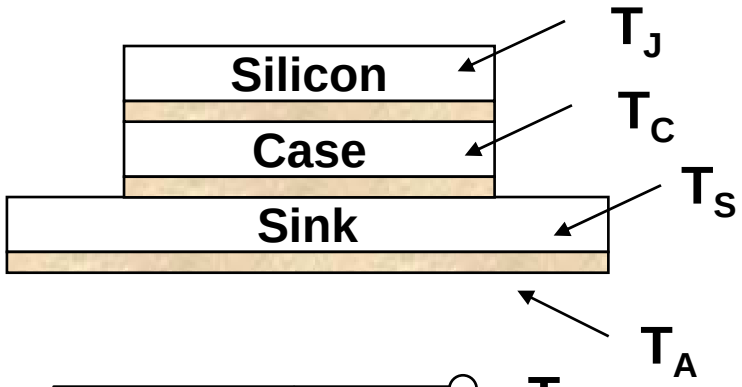
$R_{\theta JA}$ is the thermal resistance between silicon and Ambient

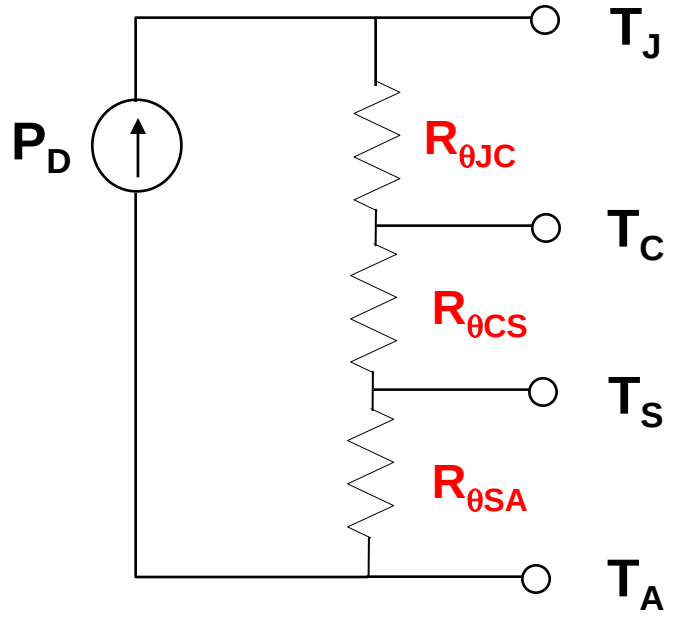


$$T_j = T_a + R_{\theta JA} P_D$$

Make this as low as possible

Realistic Scenario

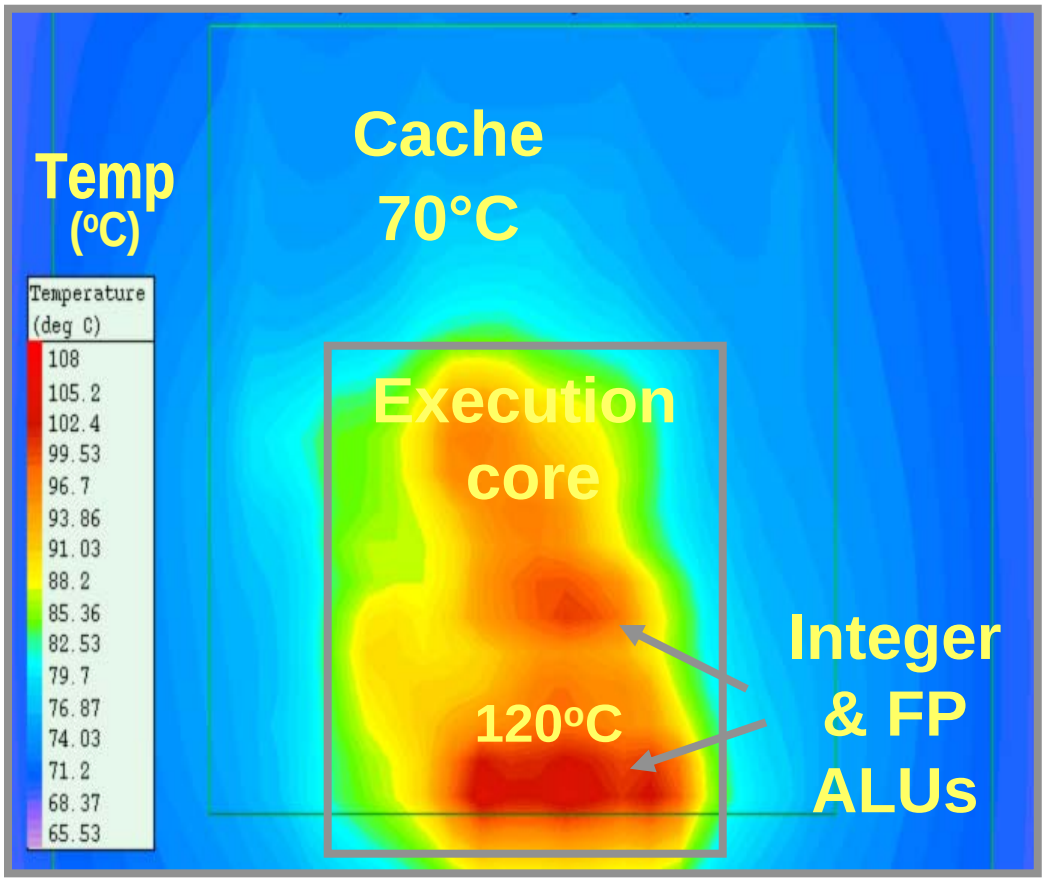




$$R_{\theta CA} = R_{\theta CS} + R_{\theta SA}$$

is minimized by facilitating heat transfer
(bolt case to extended metal surface – heat sink)

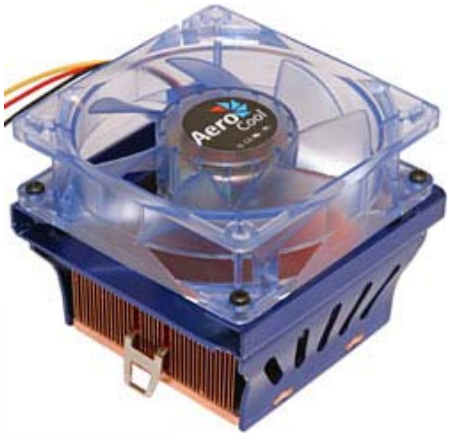
Large Thermal Gradients

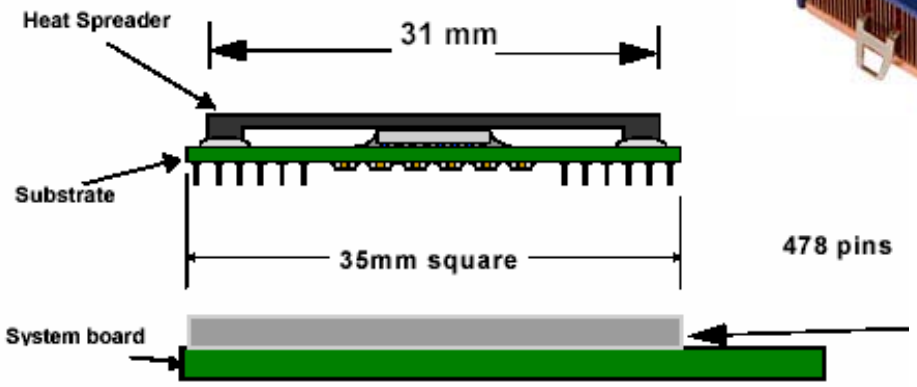


Courtesy of Intel (Ram Krishnamurthy)

Intel Pentium 4 Thermal Guidelines

- **Pentium 4 @ 3.06 GHz dissipates 81.8W!**
- **Maximum T_C = 69 °C**
- **R_{CA} < 0.23 °C/W for 50 C ambient**
- **Typical chips dissipate 0.5-1W (cheap packages without forced air cooling)**





Processor and Core Frequency	Thermal Design Power[1,2] (W)
Processors with VID=1.500V	
2 GHz	52.4
2.20 GHz	55.1
2.26 GHz	56.0
2.40 GHz	57.8
2.50 GHz	59.3
2.53 GHz	59.3
Processors with VID=1.525V	
2 GHz	54.3
2.20 GHz	57.1
2.26 GHz	58.0
2.40 GHz	59.8
2.50 GHz	61.0
2.53 GHz	61.5
2.60 GHz	62.6
2.66 GHz	66.1
2.80 GHz	68.4
Processors with multiple VIDs	
2 GHz	54.3
2.20 GHz	57.1
2.26 GHz	58.0
2.40 GHz	59.8
2.50 GHz	61.0
2.53 GHz	61.5
2.60 GHz	62.6
2.66 GHz	66.1
2.80 GHz	68.4
3.06 GHz	81.8

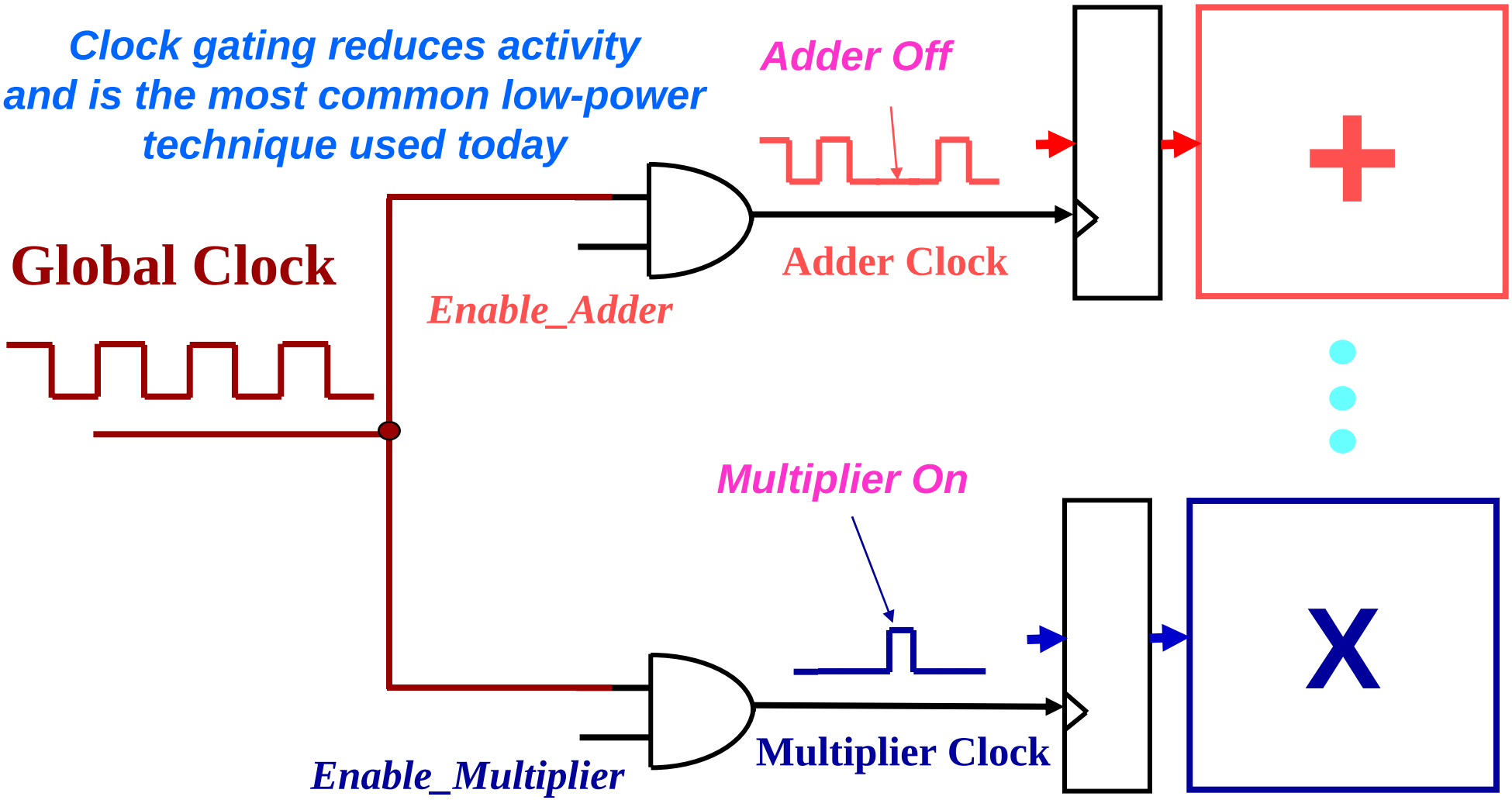
Power Reduction Strategies

$$P = \alpha_{0->1} C_L V_{DD}^2 f$$

- Reduce Transition Activity or Switching Events
- Reduce Capacitance (e.g., keep wires short)
- Reduce Power Supply Voltage
- Frequency is typically fixed by the application, though this can be adjusted to control power

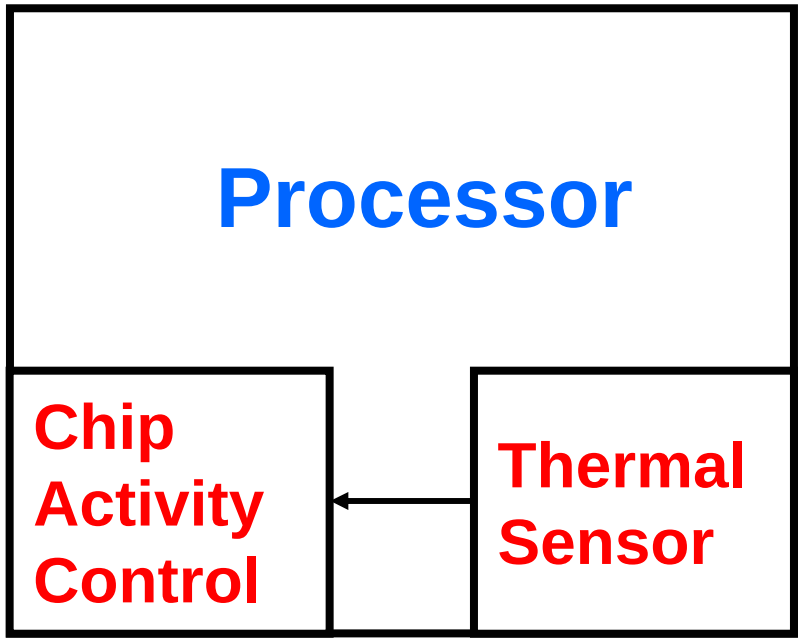
Optimize at all levels of design hierarchy

Clock Gating is a Good Idea!



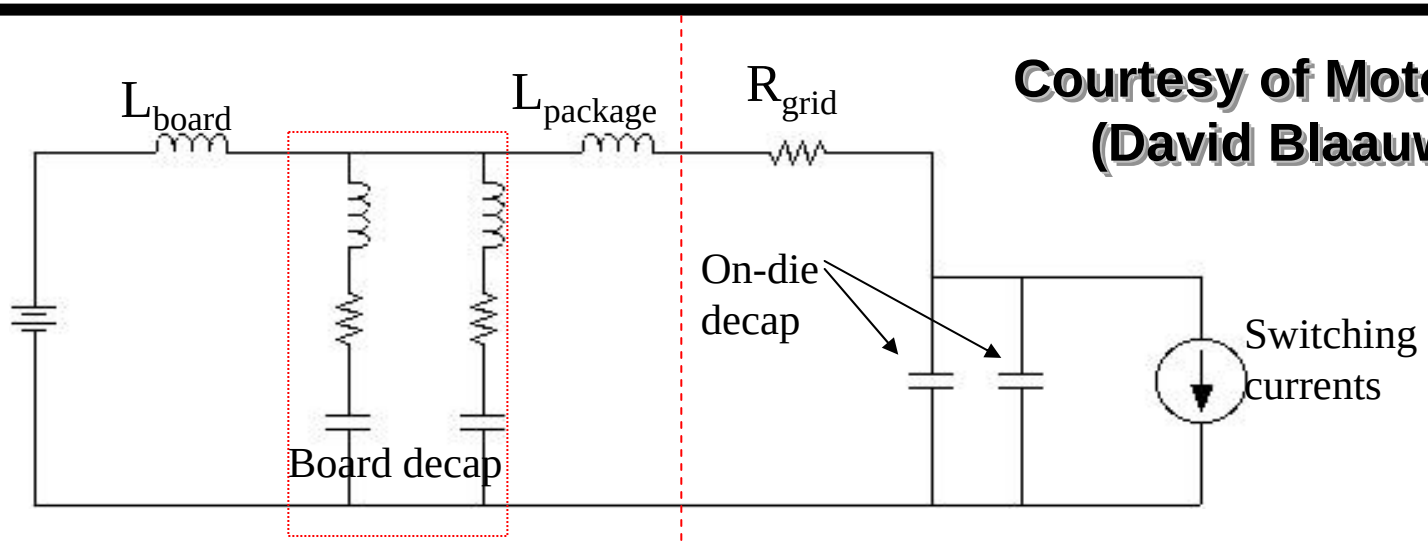
Clock Gating Reduces Energy, does it reduce Power?

Use of Thermal Feedback

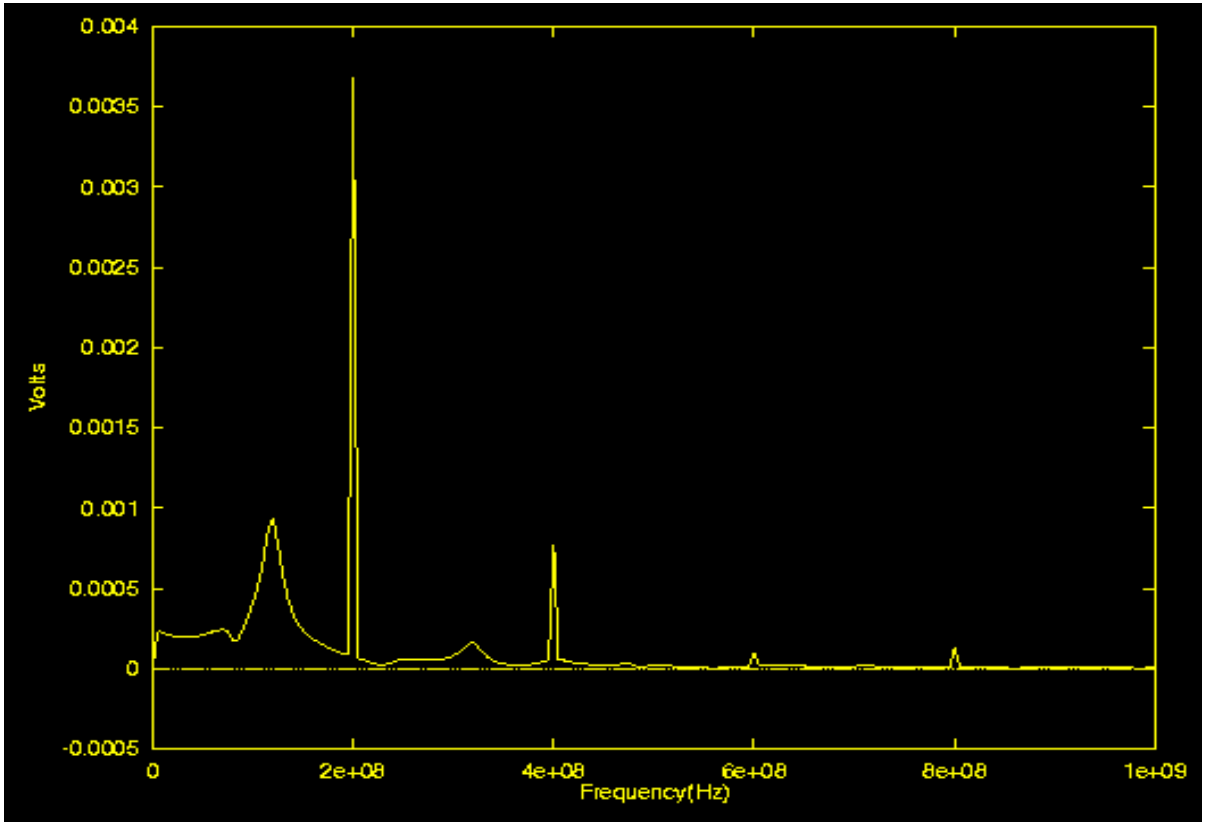


- Note that there is a difference between average and peak power
- On-chip thermal sensor (diode based), measures the silicon temperature
- If the silicon junction gets too hot, then the activity is reduced (e.g., reduce clock rate)

Power Supply Parasitics

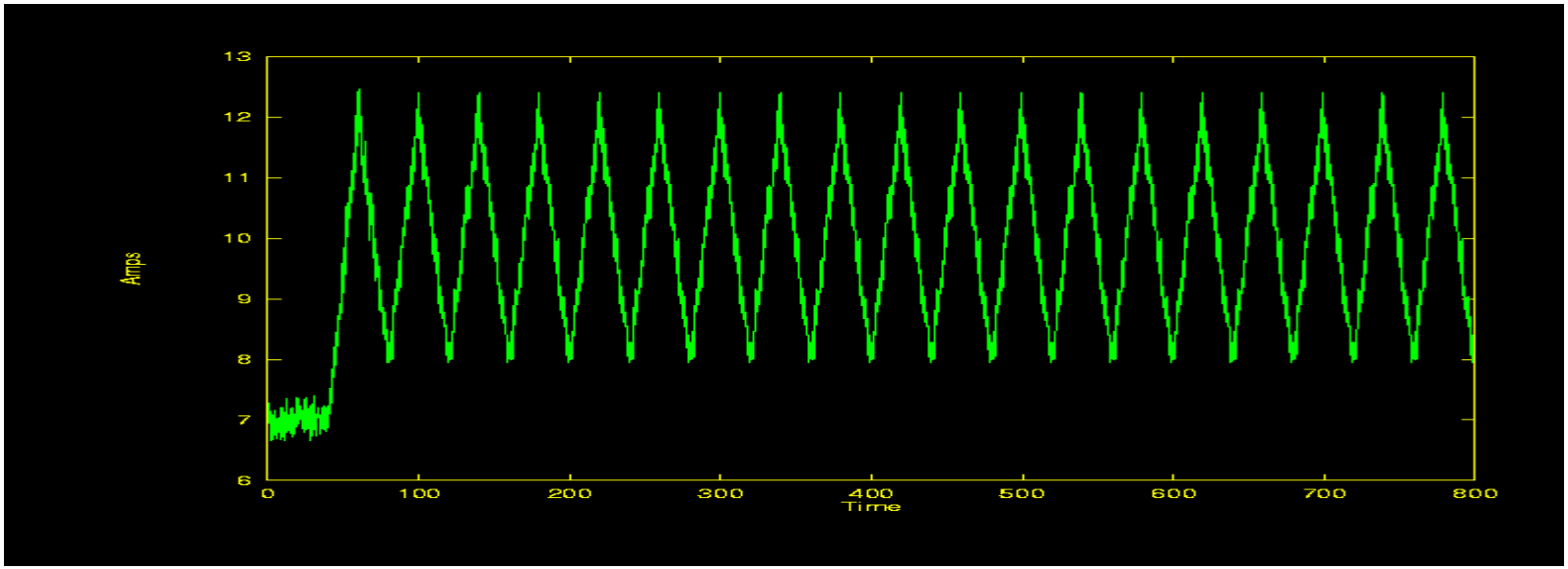


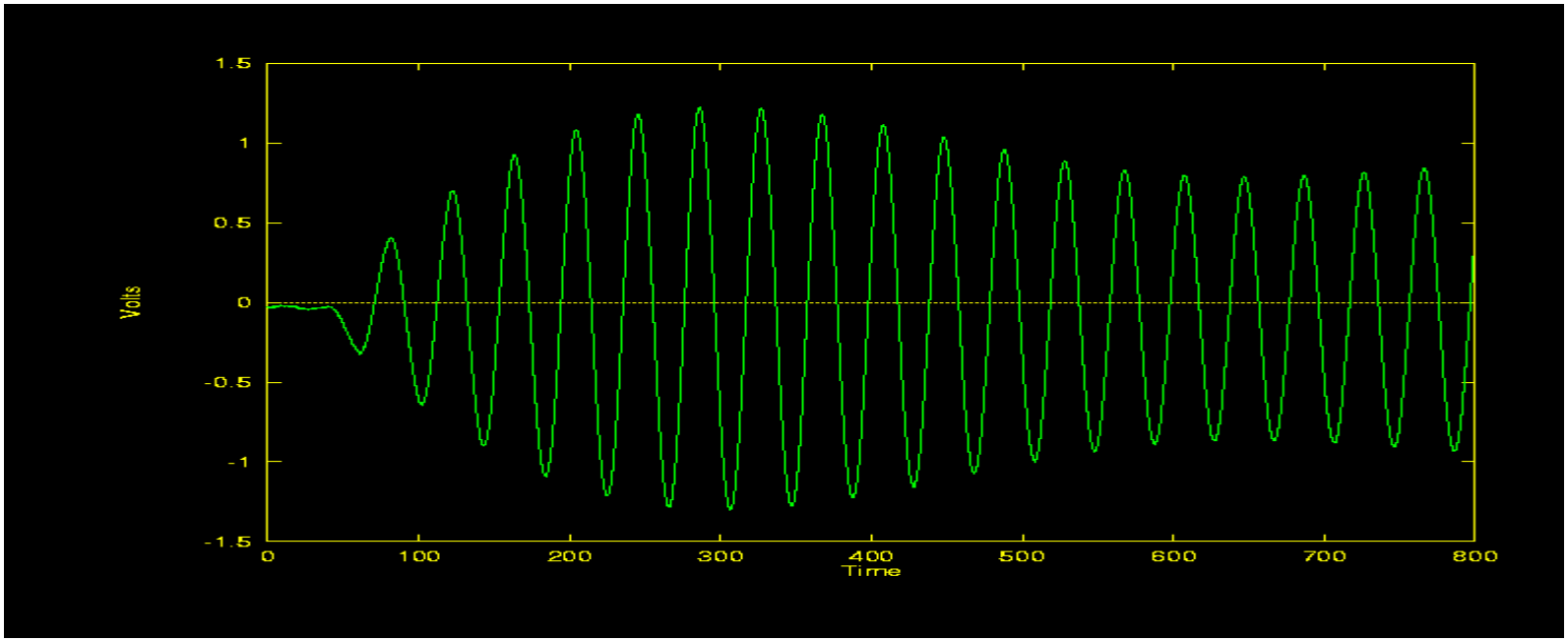
Courtesy of Motorola (David Blaauw)



200Mhz Design

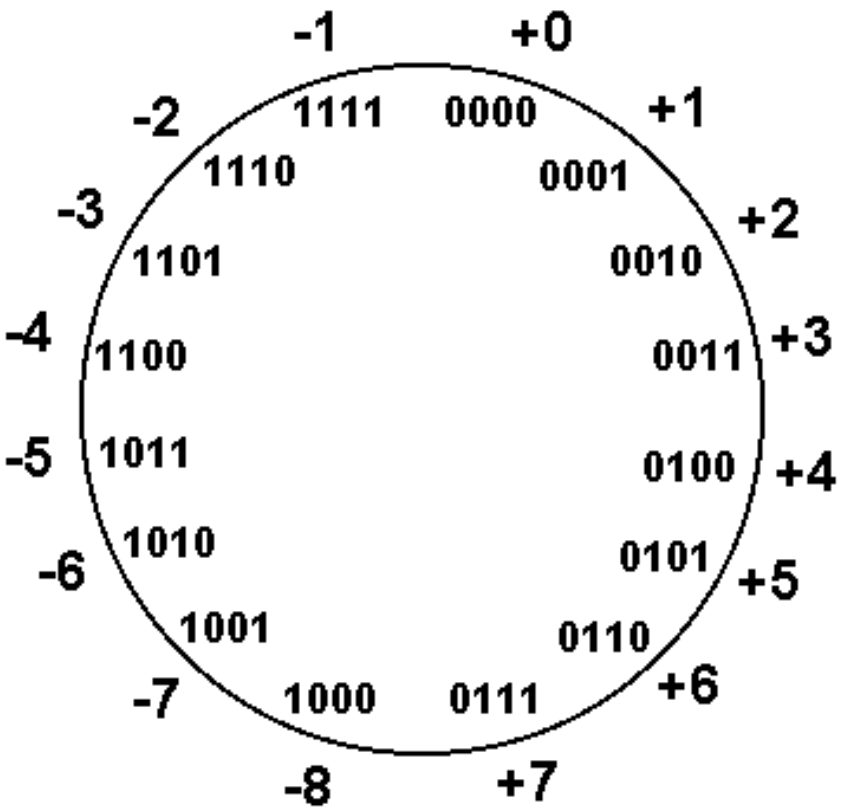
Power Supply Resonance!



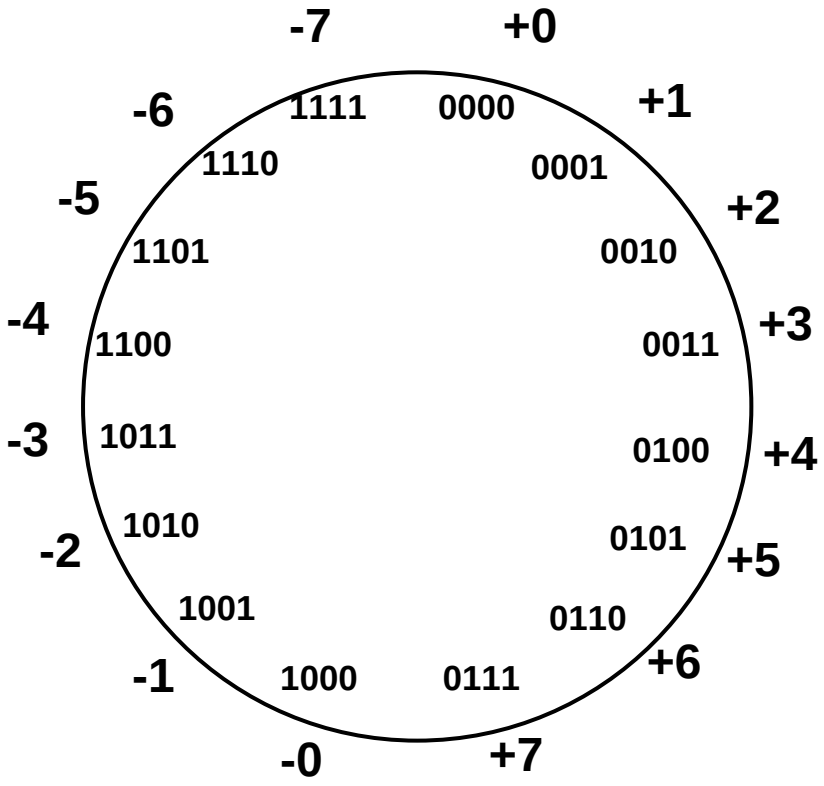


Number Representation: Two's Complement vs. Sign Magnitude

Two's complement



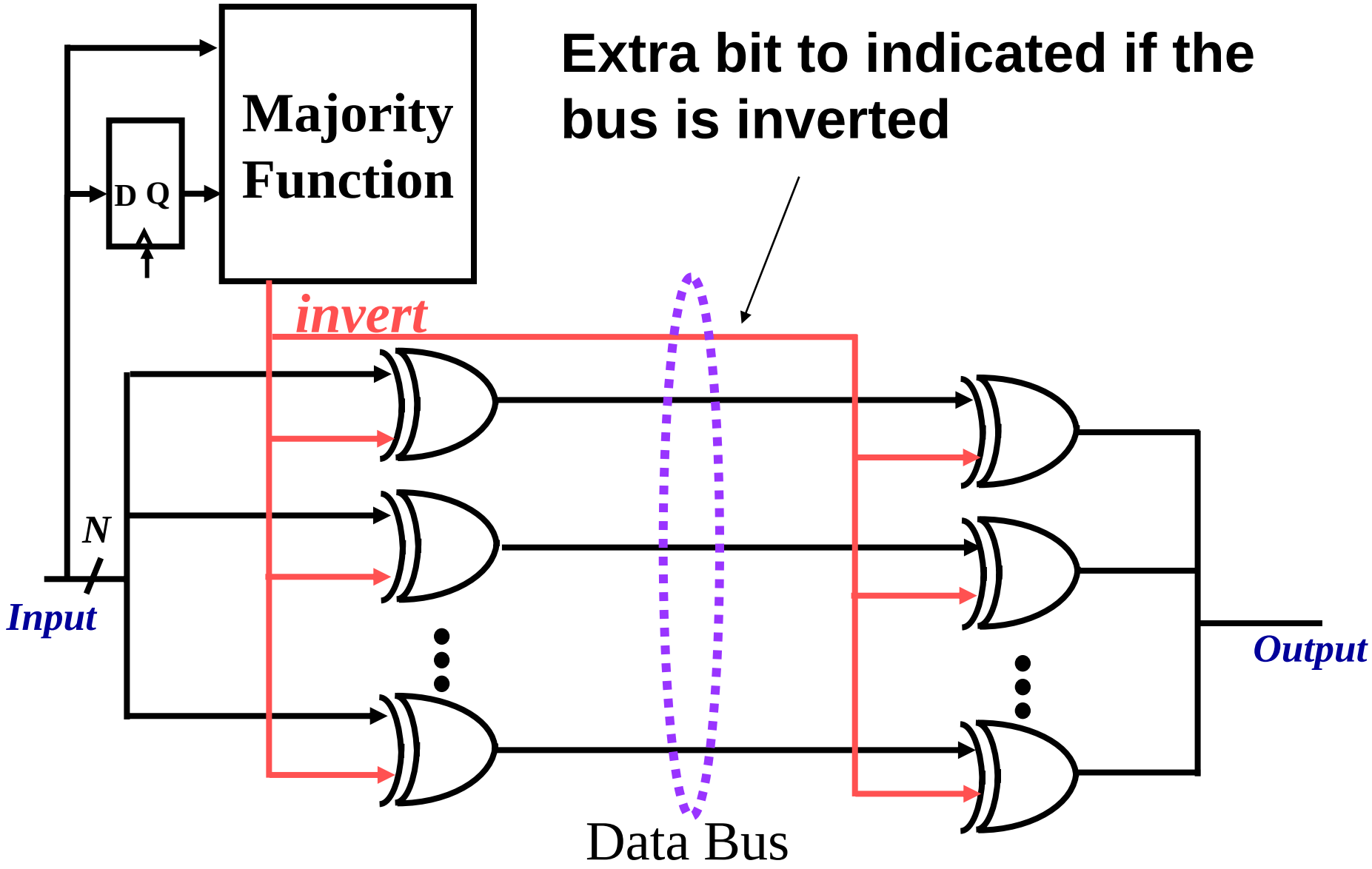
Sign-Magnitude



Consider a 16 bit bus where inputs toggles between +1 and –1 (i.e., a small noise input)

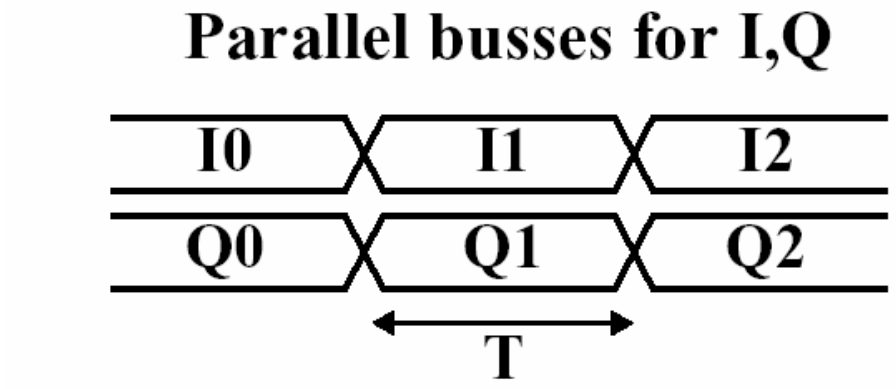
Which representation is more energy efficient?

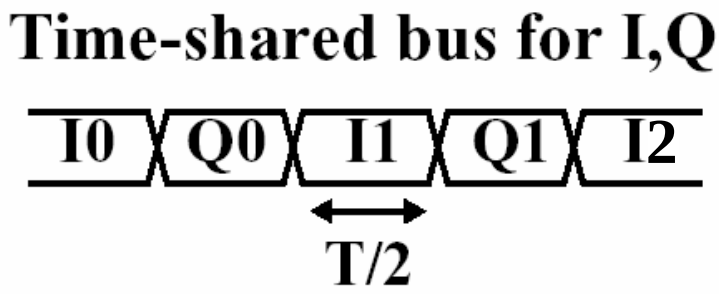
Bus Coding to Reduce Activity

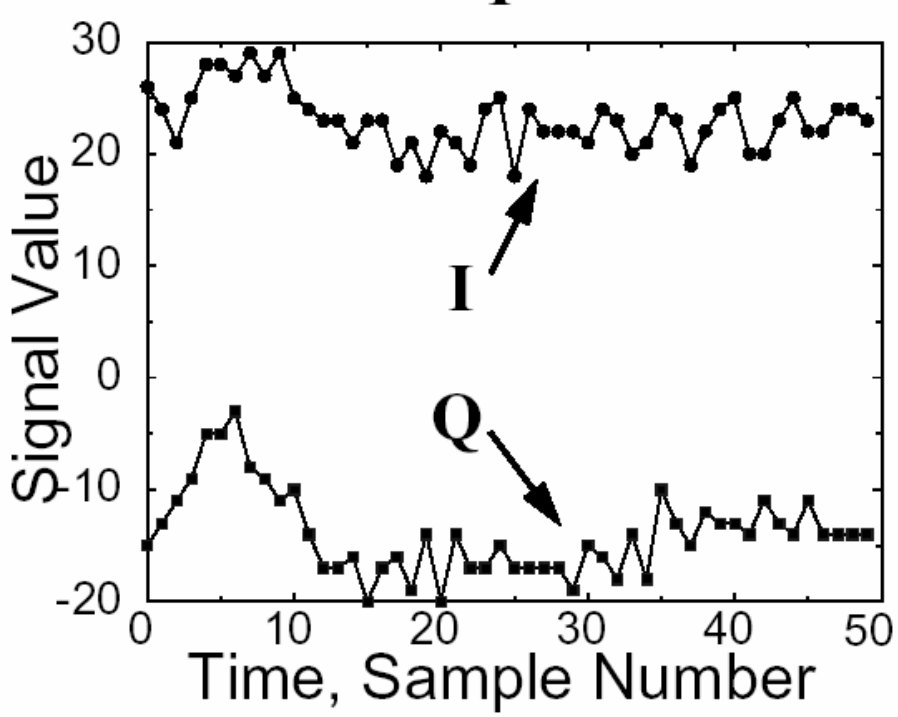


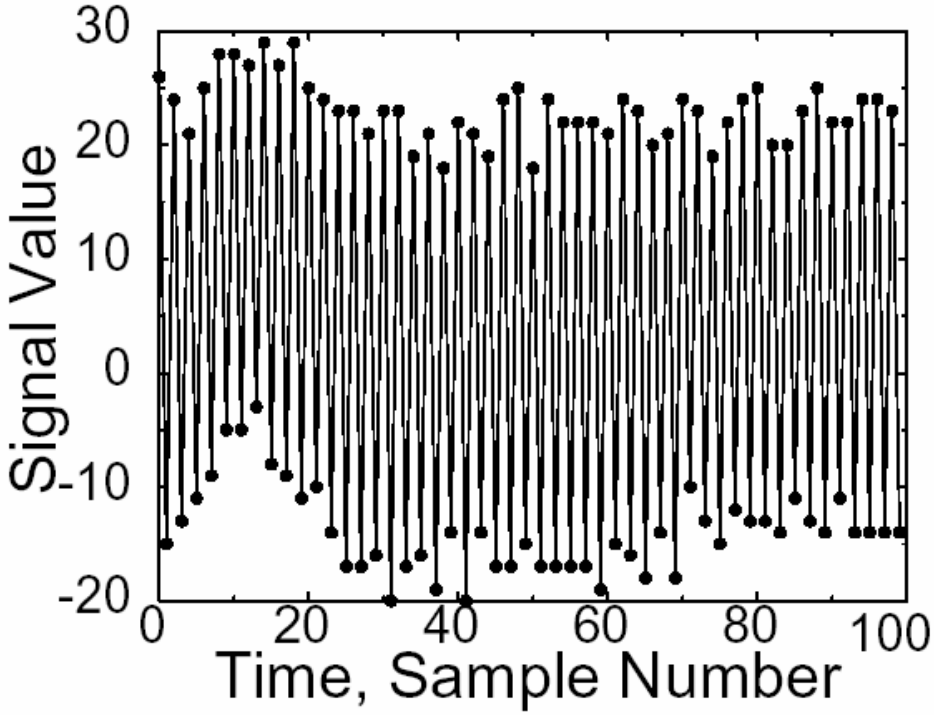
[Stan94]

Time Sharing









Time Sharing Increases Switching Activity

Not just a 6-1 Issue: "Cool" Software ???

CPU

address

16

```
MOV   ip,sp
STMDB sp!,{fp,ip,lr,pc}
SUB   fp,ip,#4
CMP   sp,sl
BLMI  __rt_stkovrf_split_small
ADD   sl,pc,#LDOUBLE24--8
BL    _printf
MOV   sl,#0
LDMDB fp,{fp,sp,pc}
```

MEMORY

	address
a[0]	0111111100000000
a[1]	0111111100000001
a[2]	0111111100000010
a[3]	0111111100000011
⋮	⋮
b[0]	1000000000000000
b[1]	1000000000000001
b[2]	1000000000000010
b[3]	1000000000000011
⋮	⋮

```
float a [256], b[256];
float pi= 3.14;

for (i = 0; i < 255; i++) {
    a[i] = sin(pi * i /256);
    b[i] = cos(pi * i /256);
}
```

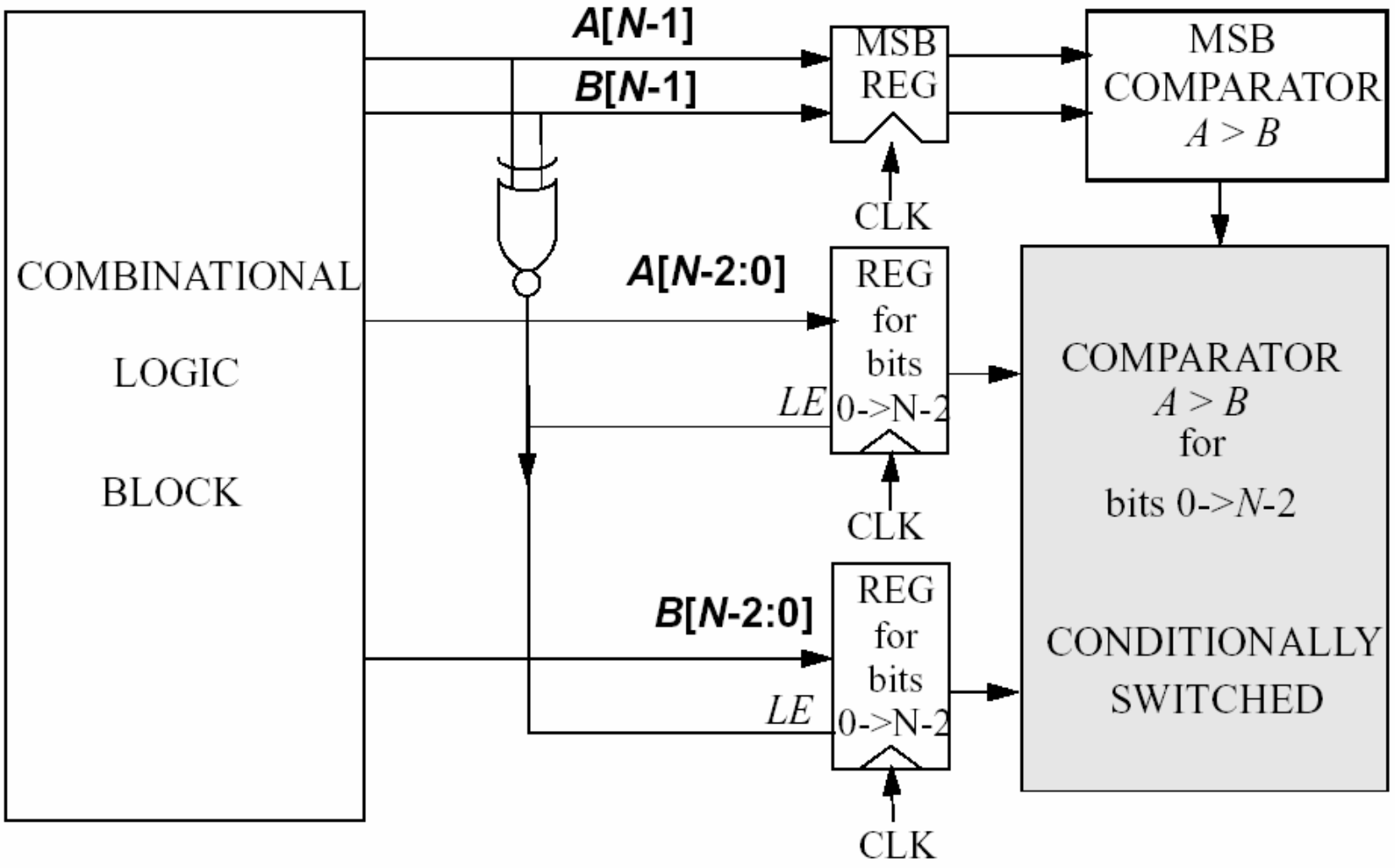
512(8)+2+4+8+16+32+64+128+256
= 4607 bit transitions

```
float a [256], b[256];
float pi= 3.14;

for (i = 0; i < 255; i++) {a[i] = sin(pi * i /256);}
for (i = 0; i < 255; i++) {b[i] = cos(pi * i /256);}
```

2(8)+2(2+4+8+16+32+64+128+256)
= 1030 transitions

Pre-Computation

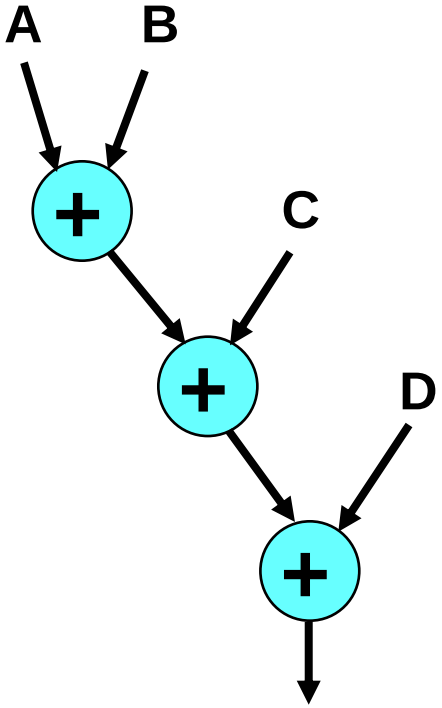


from [Alidina94]

(1994 International Workshop on Low-power Design)

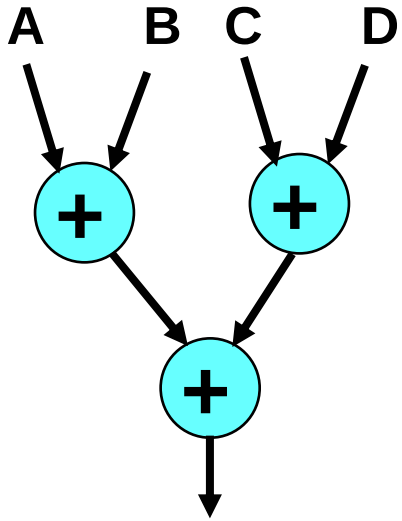
Glitching Transitions

Chain Topology



(((A+B) + C)+D)

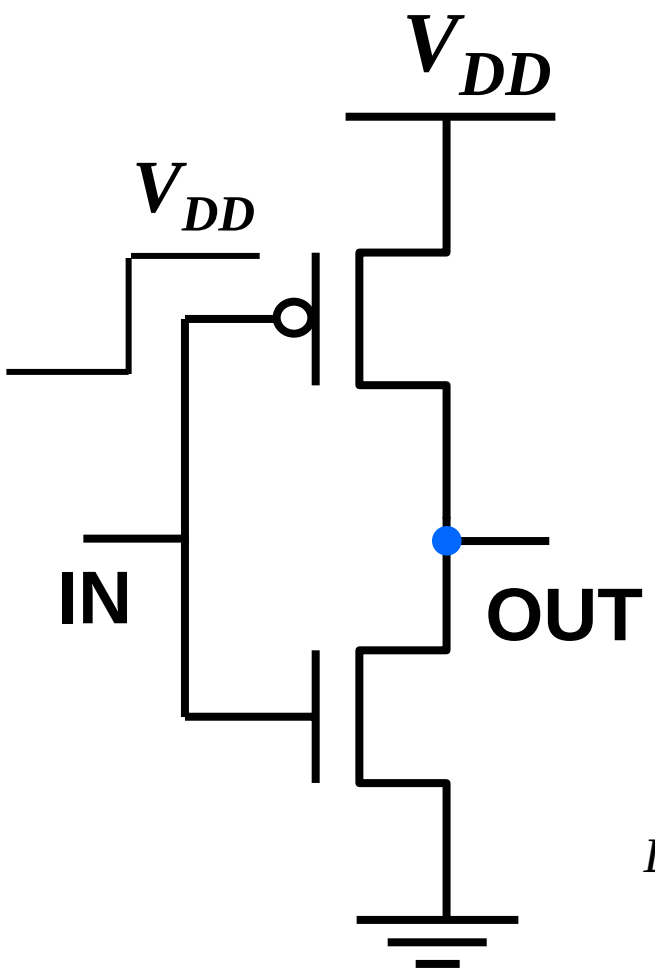
Tree Topology

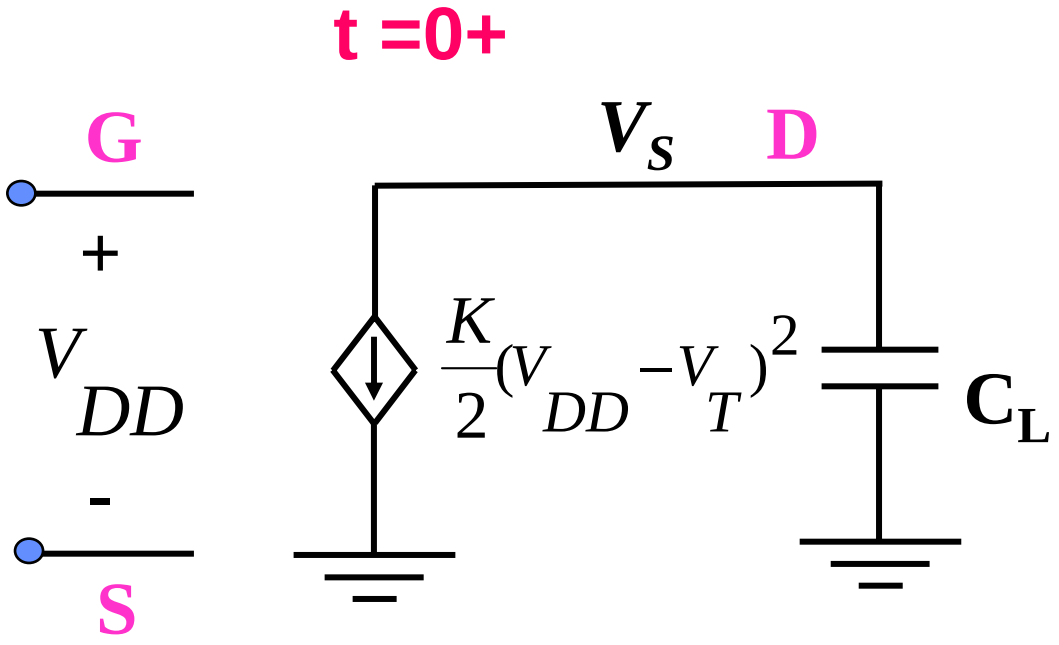


(A+B) + (C+D)

- Balancing paths reduces glitching transitions
- For 4 inputs, 50% less transitions using a tree approach
- Structures such as multipliers have lot of glitching transitions
- Keeping logic depths short (e.g., pipelining) reduces glitching

Reduce Supply Voltage : But is it Free?

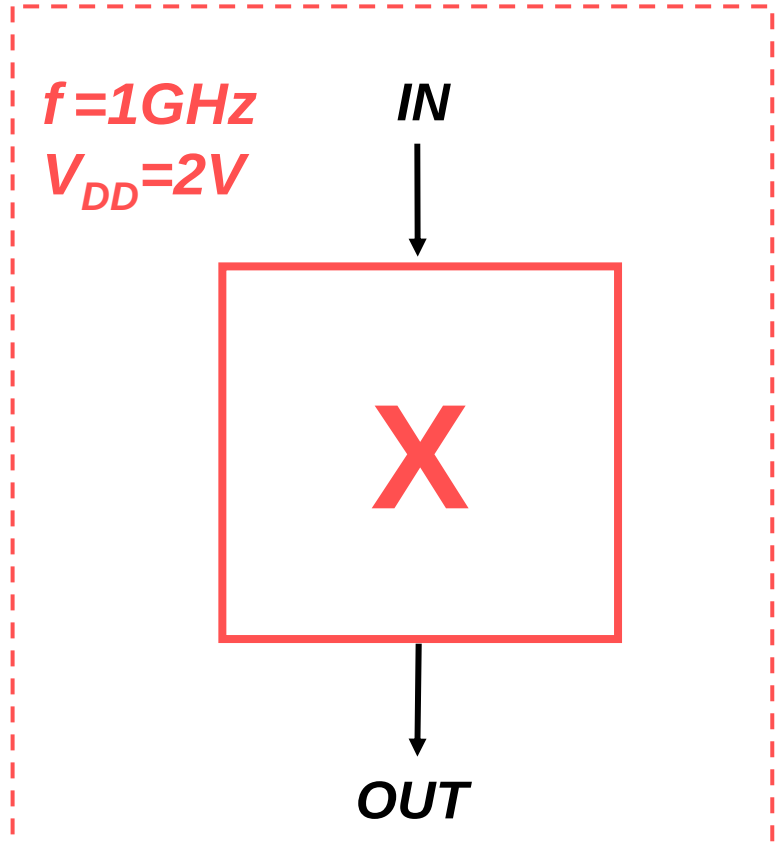


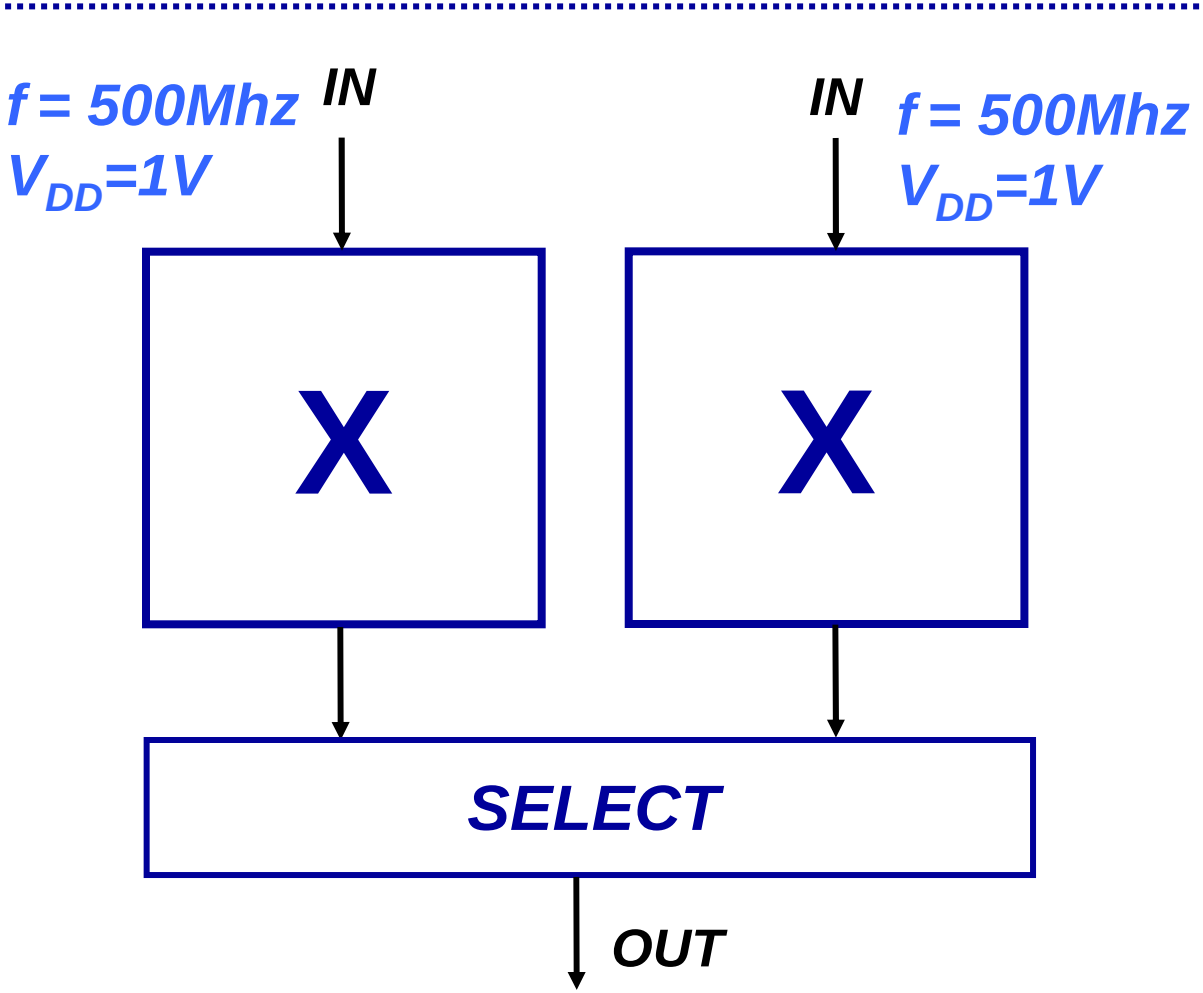


$$Delay = \frac{C_L \cdot \Delta V}{i_D} = \frac{C_L \cdot \frac{V_{DD}}{2}}{\frac{k}{2}(V_{DD}-V_T)^2} \propto \frac{V_{DD}}{(V_{DD}-V_T)^2} \approx \frac{1}{V_{DD}}$$

V_{DD} from 2V to 1V, energy ↓ by x4, delay ↑ x2

Voltage Scaling Using Parallelism



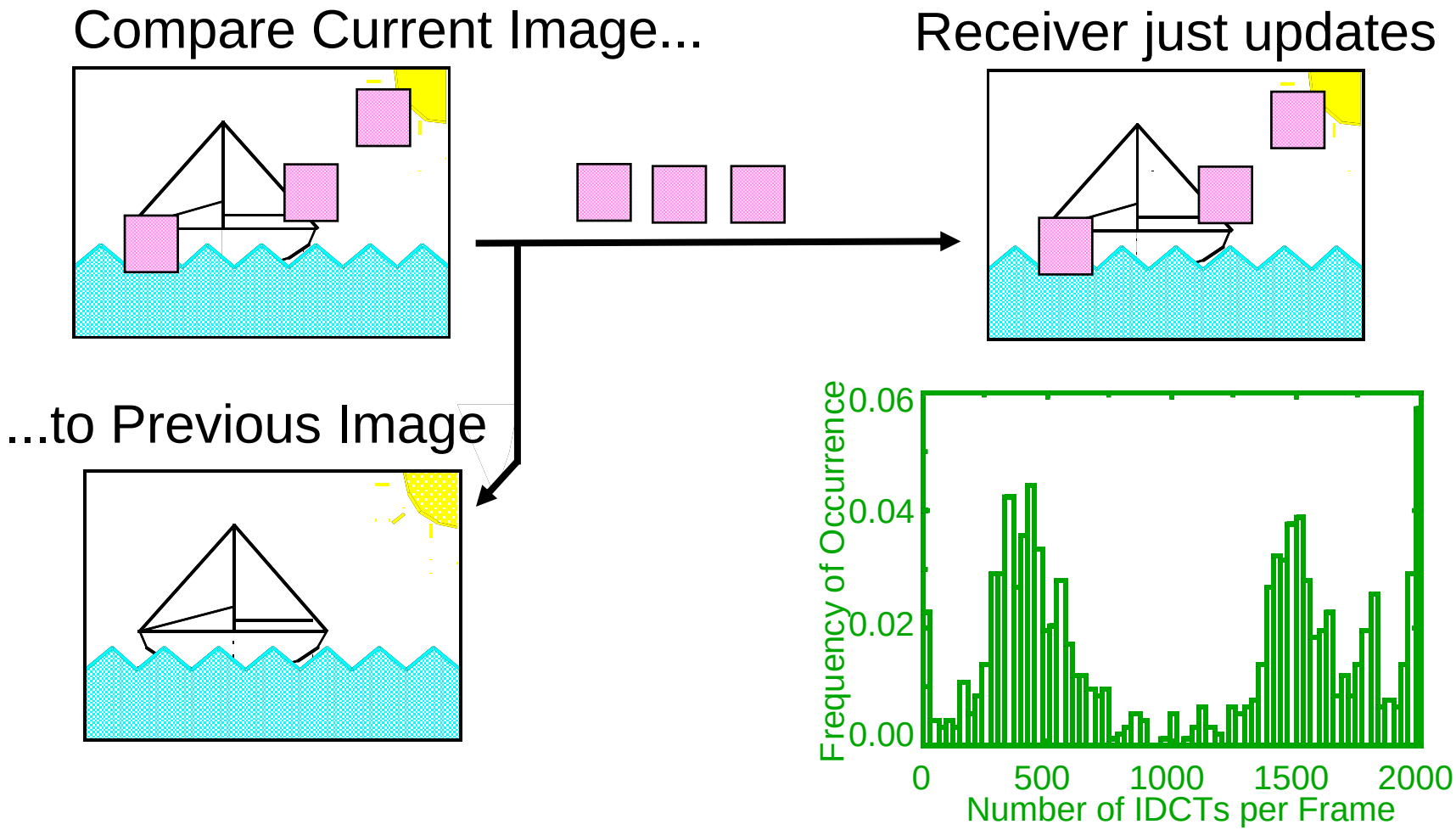


$P_{serial} = C_{mult}\ 2^2\ f$

$P_{parallel} = (2C_{mult}\ 1^2\ f\ /2) = P_{serial}/4$

Trade Area for Low Power

Algorithmic Workload



Exploit Time Varying Algorithmic Workload To Vary the Power Supply Voltage

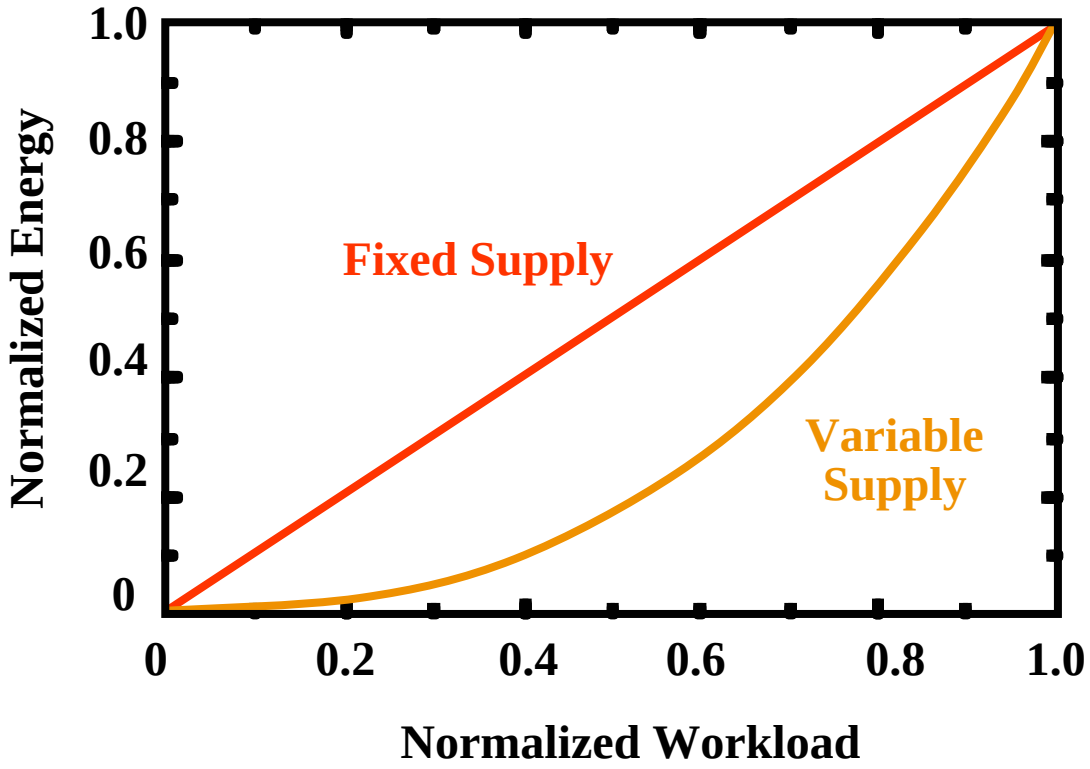
Dynamic Voltage Scaling

Fixed Power Supply

$E_{FIXED} = \frac{1}{2} C V_{DD}^2$

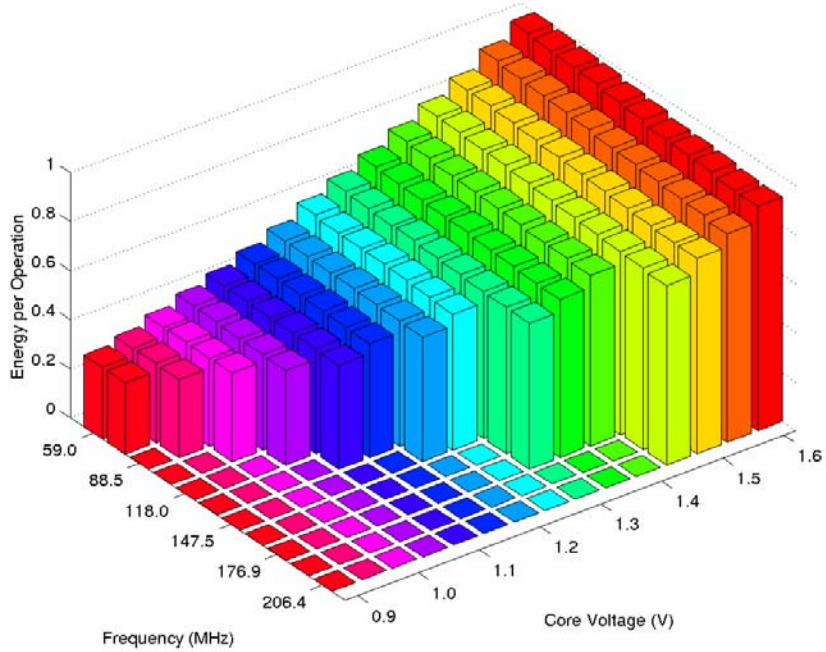
Variable Power Supply

$E_{VARIABLE} = \frac{1}{2} C (V_{DD}/2)^2 = E_{FIXED} / 4$

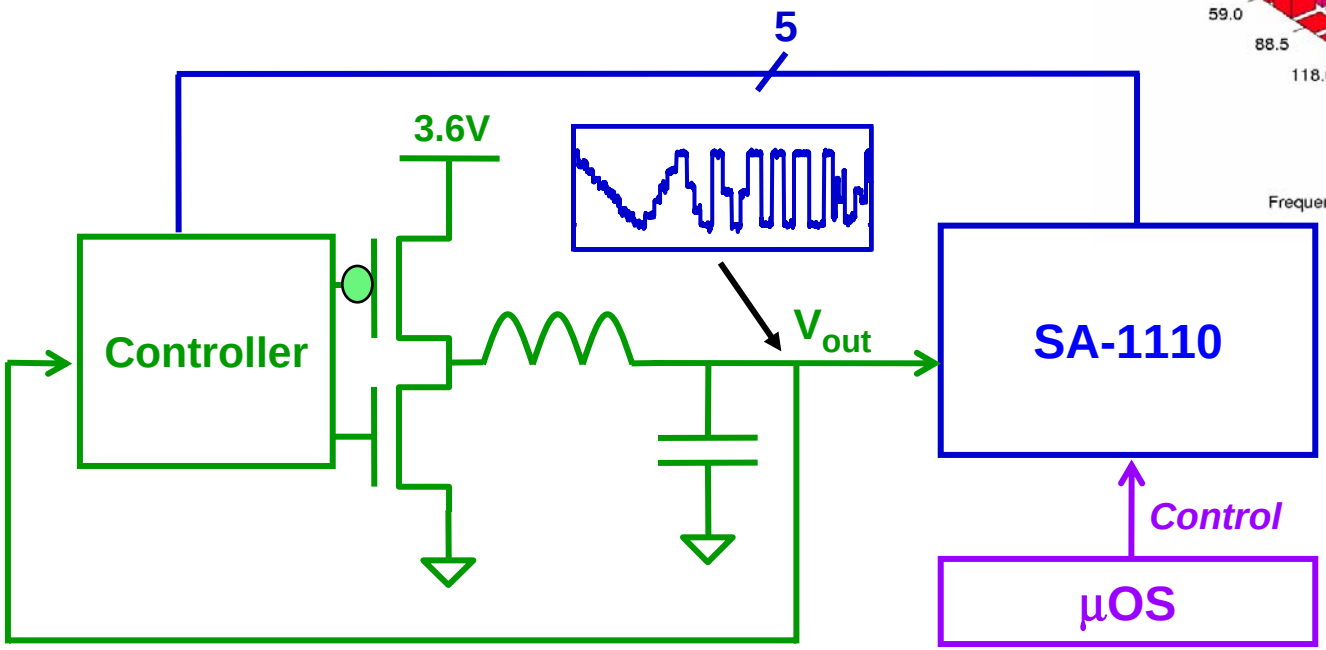


[Gutnik97]

DVS on a Processor



Digitally adjustable DC-DC converter powers SA-1110 core



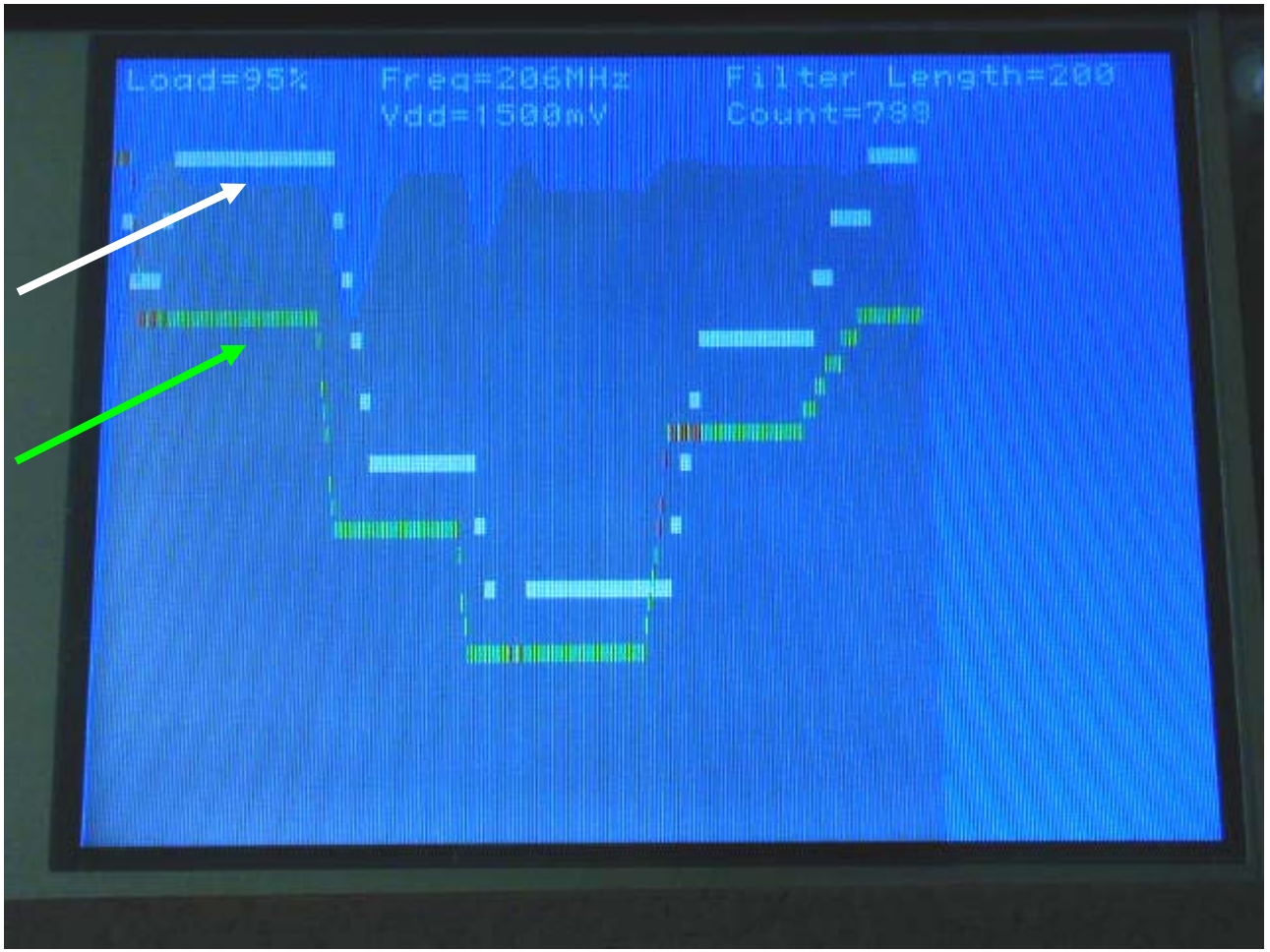
µOS selects appropriate clock frequency based on workload and latency constraints

Software Voltage Demo

- User adjusts number of filter taps
- Frequency/Voltage adjusted appropriately (via eCOS based µOS)

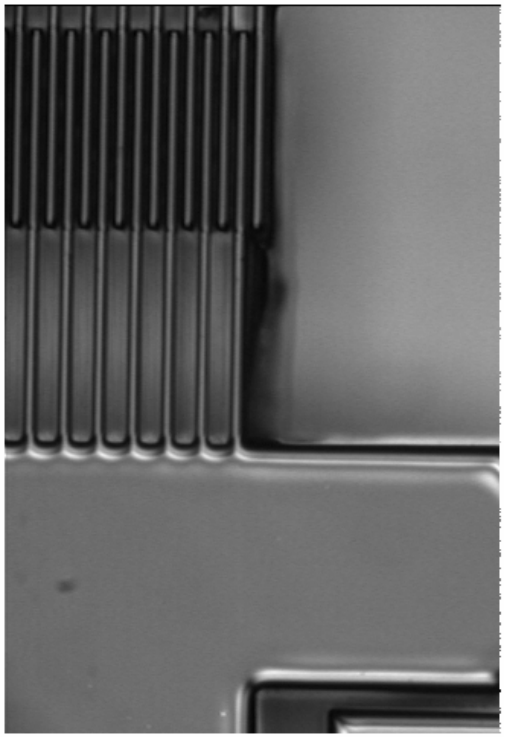
Frequency / Voltage

Workload (filter taps)



Energy Scavenging

MEMS Generator



Jose Mur Miranda/
Jeff Lang

Vibration-to-Electric Conversion

~ 10μW

Power Harvesting Shoes

Joe Paradiso
(Media Lab)

After 3-6 steps, it provides 3 mA for 0.5 sec

~10mW