

**Reversible Supercomputing  
Beyond the Limits of Moore's Law**

**Dr. Michael P. Frank**  
**CISE/ECE Depts., University of Florida**  
**ECE Dept., FAMU/FSU College of Engineering**

**Invited Talk**  
**Sandia National Labs**  
**May 12, 2004**  
**Host: Erik DeBenedictis**

I am currently at UF but I will be moving to FSU in the Fall.



## Abstract

- The various Moore's law trendlines will soon hit a number of fundamental physical limits.
  - E.g., the fundamental thermodynamic limit ( $k_B \ln 2$ ) on entropy generated per irreversible bit-erasure event.
- Any possible nanocomputing technology based on the usual irreversible logic paradigm must obey this limit!
  - And it directly limits computer performance per unit power.
- To continue power-performance improvements beyond the next 20-30 years will require a near-complete migration to a mostly reversible computing paradigm.
  - This is, by definition, the *only possible alternative!*
- The Reversible Computing Project at UF & FSU is developing integrated CMOS/MEMS chips to demonstrate the practicality of reversible computing in near-term, ultra-low-power applications.

ABSTRACT: Moore's Law will hit a fundamental brick wall in roughly 20 years (if not sooner), as device sizes approach the nanometer scale, and bit energies concurrently approach a lower bound of  $\sim 100$  kT required for reliability. As one consequence, a 100W processing node with  $10^8$  gates (as exist today) could never run faster than  $\sim 2.4$  THz if these bit energies are to be entirely dissipated on each cycle into a room-temperature environment. Note that this represents only about 10 doublings (or 15-20 Moore's Law years) in power-performance beyond present-day processors.

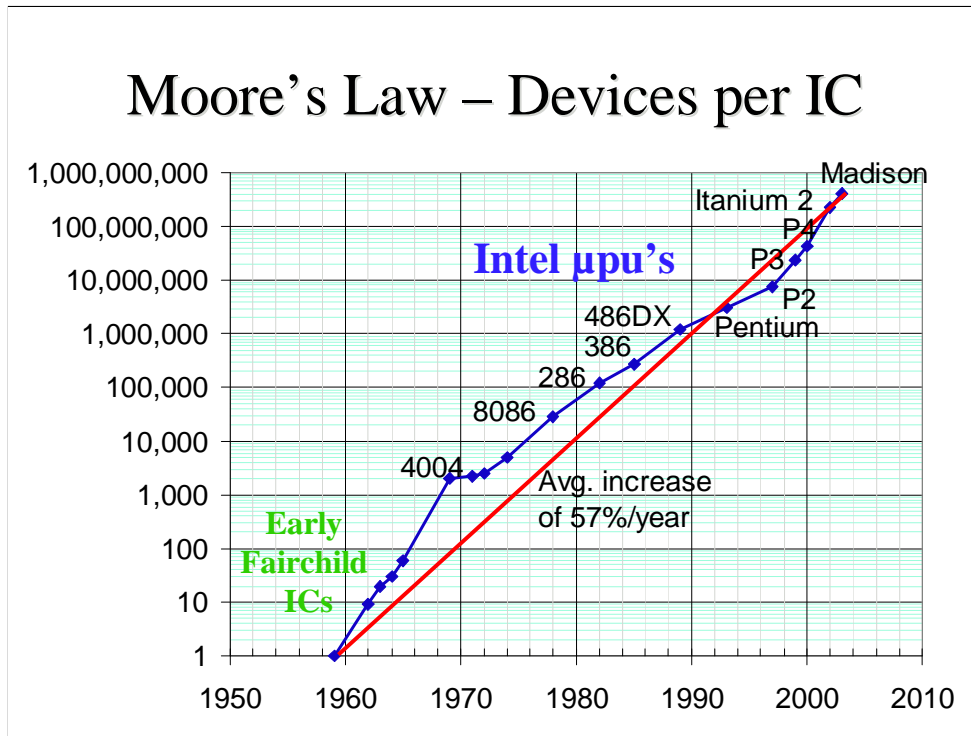
Moreover, not only standard CMOS, but also *\*any possible\** nanocomputing technology that remains based on the traditional "irreversible" computing paradigm suffers from this same sort of constraint on power-performance. Irreversible logic, by definition, relies on continual erasure of old logic outputs, overwriting them with new ones. The energy dissipated per bit-erasure can be reduced to no less than  $\sim 7$  kT, even in the best case of a degenerate nano-device in which the erasure is carried out via an isothermal compression of the phase space.

Clearly, in order for there to be any hope of continuing power-performance improvements beyond the near future, we need to seriously consider what is, by definition, the only possible alternative to irreversible computing: reversible computing. In reversible computing, we *\*decompute\** unwanted bits, rather than erasing them, which allows us to carry out bit transitions adiabatically via ballistic processes, with losses potentially  $\ll$  kT. Reversible computing imposes some overhead in terms of logic hardware complexity, but it *\*improves\** overall system cost-efficiency whenever the cost of energy, and/or cooling constraints, are dominant limiting factors on performance. Moreover, cooling becomes an increasingly stringent limiter of moderately tightly-coupled parallel 3D-mesh computations, as the problem size increases.

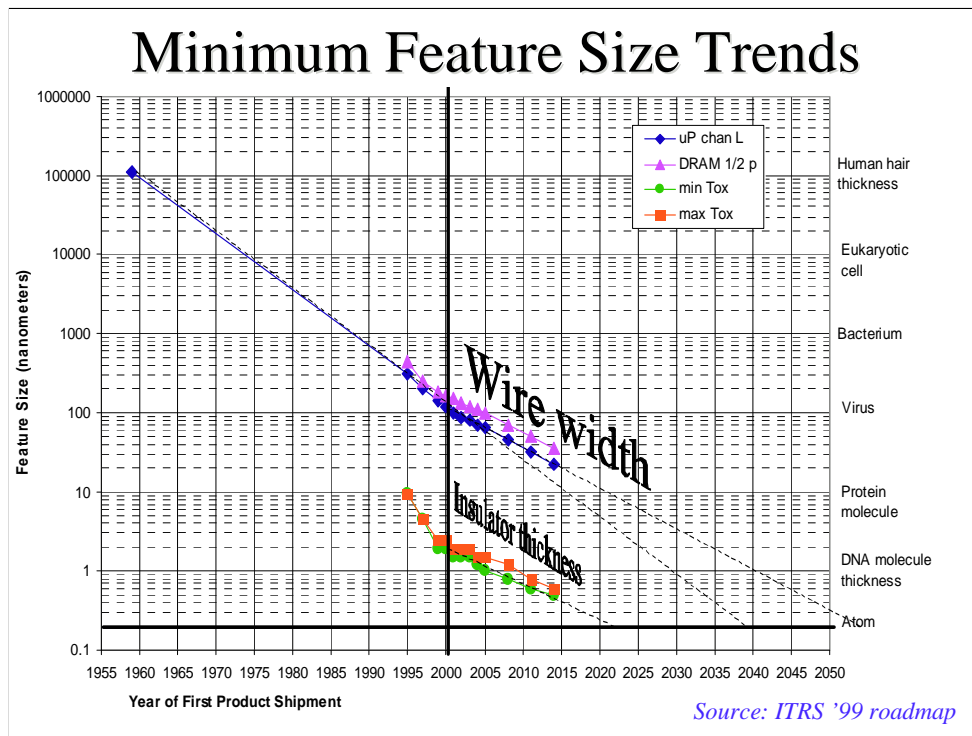
Contrary to some widespread myths, reversible computing violates no laws of physics, nor does it render computer design inordinately more difficult. Conversion of traditional logic designs to mostly-reversible ones can be mostly automated, although hand-optimization can still achieve even higher efficiency in many cases. Reversible computing does require high-Q ballistic devices to carry out most energy transfers, and in the SRC-funded reversible computing project at UF, we are presently designing custom RF MEMS resonators to serve this role in near-term reversible CMOS processors.

BIO: Mike Frank's studies of reversible computing began in the world's first nanotechnology class, taught at Stanford in 1988 by the pioneer K. Eric Drexler. In 1995 at MIT, Mike designed one of the first universal DNA computers, and found it had to be reversible for fundamental thermochemical reasons. Turning to more practical electronic technology, Mike and his fellow students created the world's first fully-reversible processors in CMOS VLSI: Tick, Flattop, and Pendulum. Since graduating from MIT in 1999, Mike has been working at the University of Florida to construct the foundations of a new discipline of Nanocomputer Systems Engineering that takes reversible computing into account. He also teaches courses on computer architecture and the physical limits of computing.

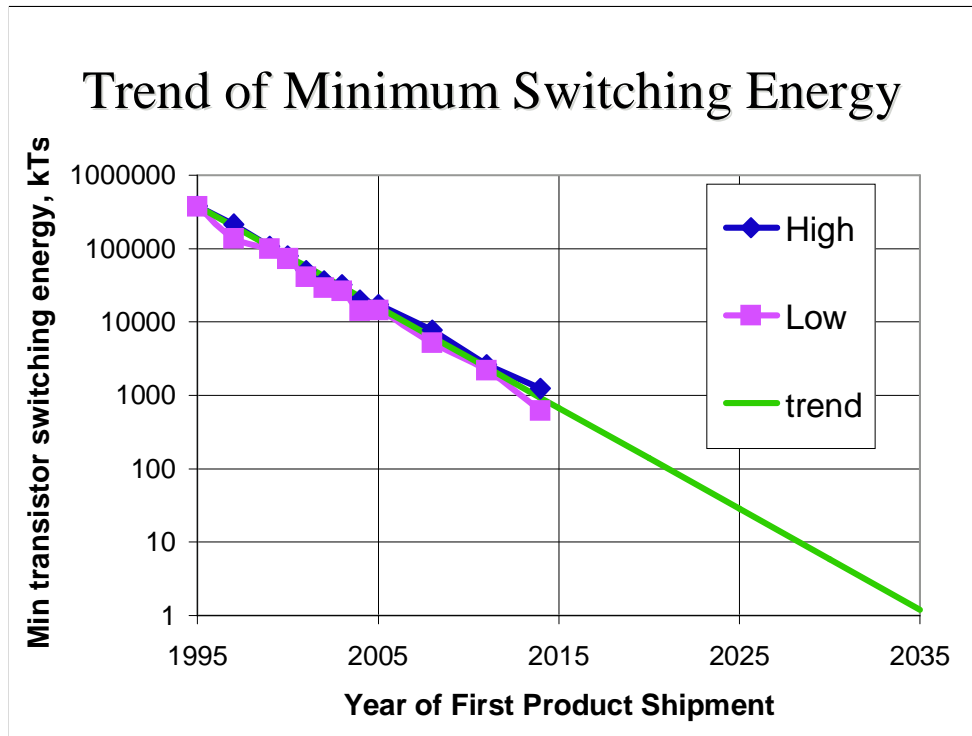
## Moore's Law – Devices per IC



The literal statement of Moore's Law is that the number of transistors per chip doubles every 18-24 months. As we can see from the data in this chart, since the first planar transistors were fabricated in 1959, the number of transistors per chip in processors has increased on average by 57% per year, which corresponds to a doubling every 18 months on average.

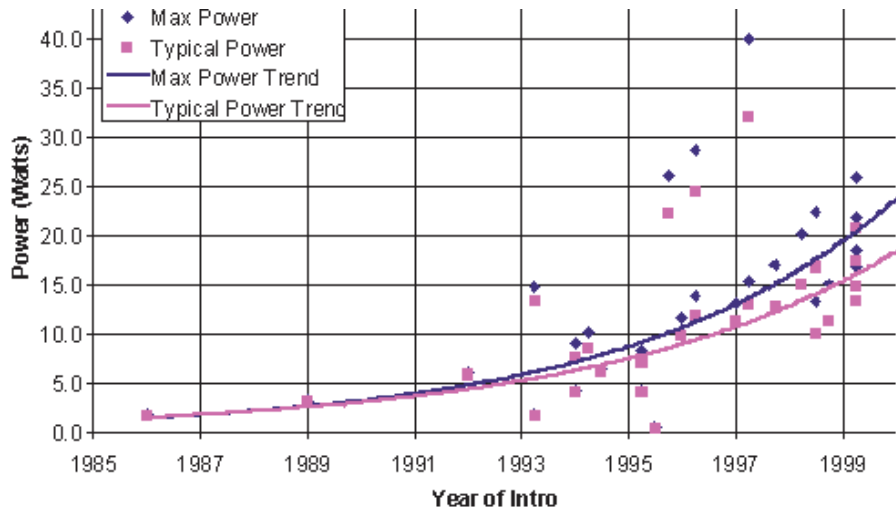


Of course, this trend is enabled by exponentially decreasing transistor sizes. In this chart derived from the 1999 ITRS roadmap, we see that since 1959 we have come halfway (in orders of magnitude) between the thickness of a human hair and the size of an atom. If the trends continue we will hit atom size by about 2040 or so; clearly we cannot scale functional devices below that point. In order for transistors per unit area to continue increasing, we will have to start building up layers of devices in the 3<sup>rd</sup> dimension. But this leads to serious problems in getting rid of waste heat.



In fact, the only reason that the waste heat hasn't been a severe problem already is that transistor switching energies have also been decreasing exponentially as devices get smaller. Here we see minimum energy data derived from the '99 ITRS roadmap. It is expressed in terms of the room-temperature thermal energy, which also corresponds roughly to the number of bits worth of entropy that are generated when that energy is dissipated to heat. Of course, it is an absolute certainty that we will never be able to encode a bit of logical information in less than 1 bit's worth of physical information, so this line absolutely must level off at 1 bit's worth of energy or .7 kT. However, this energy need not be *dissipated* – instead it can be recycled, that is the point of reversible computing. If we don't recycle it, then the curve will level off, and power-performance can't improve any further. In fact, the past trend of decreasing energy hasn't been fast enough to keep power dissipation from becoming an increasing problem.

## Microprocessor Power Trends

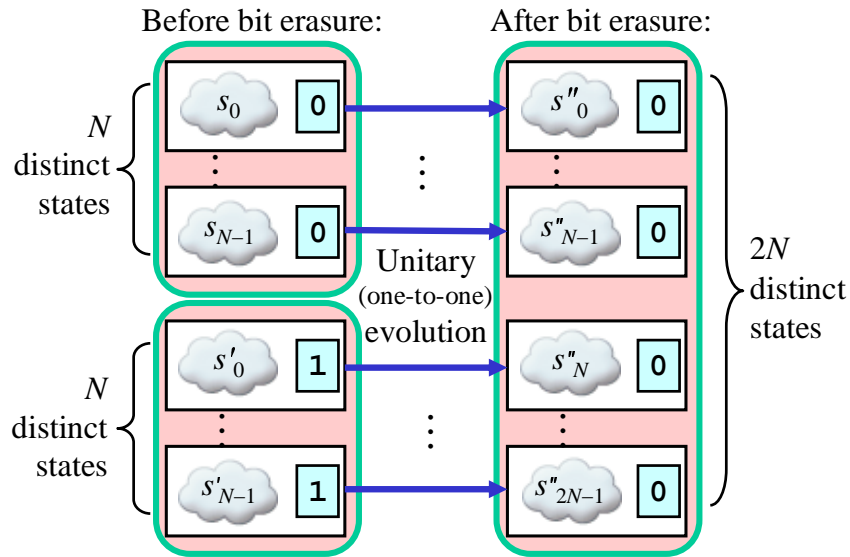


Gunther *et al.*, "Managing the Impact of Increasing Microprocessor Power Consumption," *Intel Technology Journal*, 1<sup>st</sup> quarter 2001.

This chart from Intel illustrates how the actual power dissipation of microprocessors has been trending upwards over the decades. It is not uncommon today to see newly released microprocessors with power dissipation levels well in excess of 100 Watts! If the energy trend levels off, these curves will increase even more steeply if we continue increasing processor performance as quickly as we have been in the past, and we don't decrease the fraction of bit energies that is decreased with each operation, which is what we want to do in reversible computing.

## Landauer's (1961) Principle: The Minimum Energy Cost of Bit Erasure

Anticipated by  
von Neumann '49



Increase in entropy:  $\Delta S = \log 2 = k \ln 2$ . Energy dissipated to heat:  $T\Delta S = kT \ln 2$

Back to the physical limits on computing now.

Let's give an example of the reasoning behind one of these relationships between fundamental physics and information processing. This one is the motivation for reversible computing and was discovered in 1961 by Rolf Landauer of IBM research. Landauer considered the minimum possible physical entropy generation that might result from the erasure of 1 bit of known information. Although Landauer used a more involved argument, the drawing here suffices to prove his point. There are 2 possible logical states of the bit in question, together with some number  $N$  of distinguishable physical states of the rest of the computer, for a total of  $2N$  distinct states of the entire machine. The unitary, one-to-one nature of time evolution in quantum mechanics guarantees that the number of distinct states of a closed system is exactly conserved. Therefore, after the logical bit is erased, there are still  $2N$  states of the machine. There is the same total amount of variability, but it now must all reside in the rest of the machine. If the information is not logically present, it must be in the form of unknown physical information, or entropy. Since the number of states of the rest of the machine was multiplied by 2, the amount of entropy (which is the logarithm of the state count) is increased by an addition of  $\log 2$ . A  $\log 2$  amount of entropy is  $(\ln 2)$  times as large as Boltzmann's constant  $k$ . To release  $k(\ln 2)$  entropy into an environment at temperature  $T$  requires committing  $kT(\ln 2)$  energy to the environment, by the very definition of temperature. Thus, information erasure ultimately implies a minimum energy expenditure proportional to the temperature of the environment. Note that cooling the system to a low temperature  $T$  can not help since the entropy must still eventually get out to the environment, incurring a dissipation of  $k \ln 2$  times the temperature of the environment.

## Reliability Bound on Bit Energy

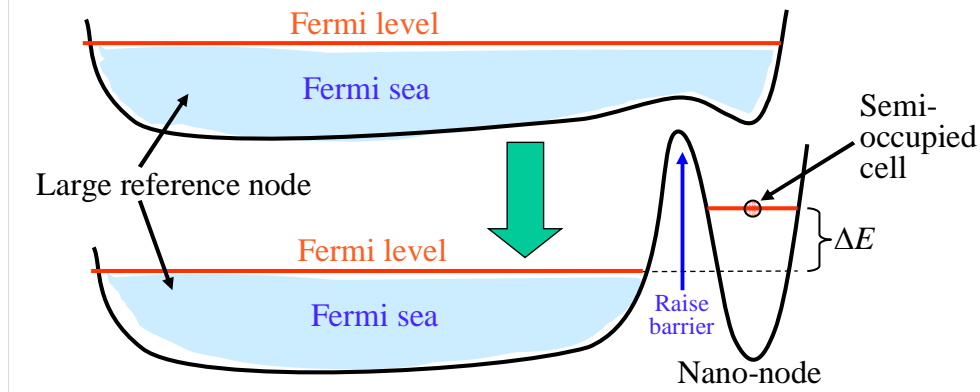
- To reliably store (latch) a bit of data with less than 1 error in  $N$  repetitions requires that:
  - In the equilibrium microstate distribution, when latching, the number of accessible microstates leading to the correct stored bit value should be  $N$  times the number leading to the incorrect bit value.
    - $\therefore$  There should be  $\Delta E \gtrsim k_B T \ln N$  energy difference between storage-cell states having the correct and incorrect bit values, at the time of latching, in a device at temperature  $T$ .
      - This follows directly from the Boltzmann distribution.
    - If and when this energy gets *dissipated* by the device, this would lead to an characteristic entropy increase of  $\Delta S = \log N = k_B \ln N$ .
- **Example:** Reliability factor of  $N=10^{27}$  (1 error in a  $10^9$  device processor running for ~30 years at 1 GHz)
  - Associated entropy:  $\log 10^{27} = k_B \ln 10^{27} \approx 62k_B = 8.6 \times 10^{-22}$  J/K
  - Energy that must be dissipated into a room- $T$  (300 K) environment:  $k_B(300 \text{ K}) \ln 10^{27} = 2.6 \times 10^{-19}$  J (or 260 zJ). Sounds small, but...
  - If this much energy were dumped by each device at a frequency of 1 GHz, the total power dissipated by the entire  $10^9$ -device processor is  $\frac{1}{4}$  W.
  - Can have at most *4 million* such processors within a 1 MW power budget.
  - Maximum speed:  $4 \times 10^{24}$  device-cycles/sec., or 40 EFLOPS
    - Assuming 1 FLOP requires 100,000 device-cycles.

Here is another even more imminent limit on bit energies that will apply if traditional switching techniques are used. This is the bound due to the need to latch bits reliably in the presence of thermal noise. The Boltzmann distribution (or appropriate quantum-statistical distribution) determines the minimum energy difference between two states in order for their probabilities of occupancy to differ by a given factor. If this energy difference is dissipated to heat when erasing a bit, this leads to a corresponding amount of dissipation of  $kT \ln N$ , where  $N$  expresses the number of latching operations that can be performed in between errors. For example, a typical desirable error rate of 1 in  $10^{27}$  leads to an entropy generation of about 60k which corresponds to 260 zeptoJoules or about 1.6 electron volts. This sounds small, but it leads to an absolute minimum of  $\frac{1}{4}$  Watt of power for a billion-transistor processor switching at a Gigahertz. This translates to a maximum power-performance of only about 40 TFLOPS per Watt assuming each FLOP takes 100,000 such latching operations.



## Reliability Bound Example

- Store a bit by raising an energy barrier to isolate electrons on a nano-island (w. discrete spectrum).
  - Probability of trapping an extra electron in a cell at  $\Delta E$  is  $1/(1+e^{\Delta E/kT}) \approx e^{-\Delta E/kT}$ . (Fermi-Dirac distribution.)



Here we imagine raising a potential energy barrier (say via a nearby electrode, or via introducing a spatial gap between conductors) between a nanoscale island and a large reference electrode. If the island is small its energy spectrum will be discretized (it will be a quantum dot) and we can consider storing bits in the occupancy numbers of individual single-electron states or “cells” as I call them. (Actually there would be two degenerate cells at each energy level, one spin-up and one spin-down, although we could do level-splitting using a magnetic field if we liked.) Anyway, if the cell in question ends up with energy  $\Delta E$  above the Fermi level, then the Fermi-Dirac distribution tells us its probability of occupancy. If we wanted it to be unoccupied and it turns out occupied, this is an example of a thermal noise error. Again we see  $\Delta E$  has to be order  $kT \ln N$  if  $1/N$  is the probability of error.

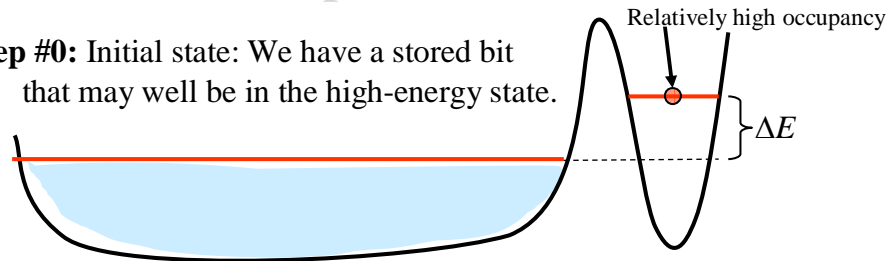
## What Exactly Does this Imply?

- The reliability bound indeed lower-bounds the energy difference between correct and incorrect states at the time that a bit is first stored,
  - The “simple, dumb” way to erase a bit is to remove the barrier, which dumps this energy on the floor...
- But, this is not the only possible way to erase a bit. The  $kT \ln N$  energy need not be dissipated.
  - More cleverly designed erasure mechanisms can reduce the energy dissipation to approach the von Neumann-Landauer bound of  $kT \ln 2$  arbitrarily closely, without sacrificing reliability.

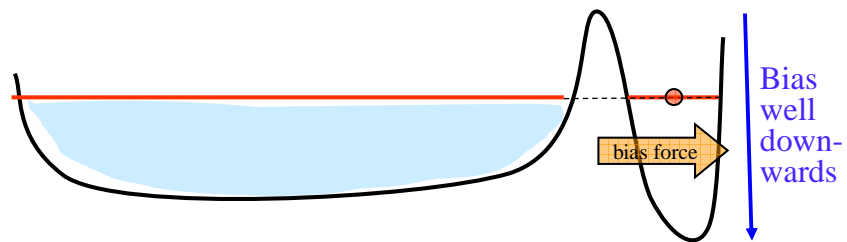
This slide speaks for itself.

## Cheap Bit Erasure

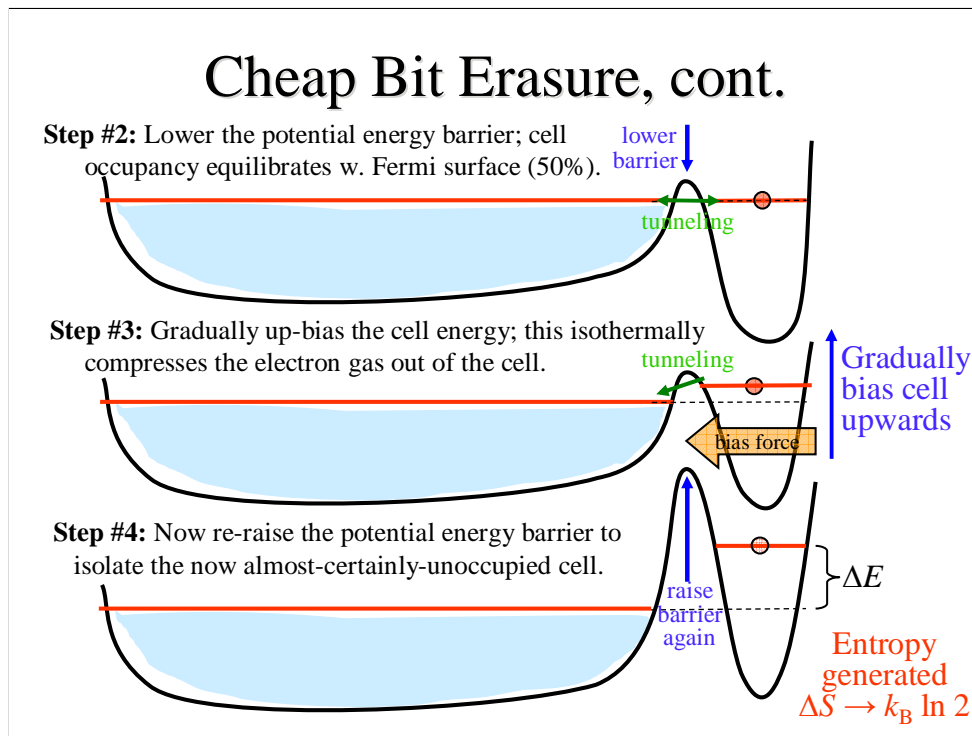
**Step #0:** Initial state: We have a stored bit that may well be in the high-energy state.



**Step #1:** Apply a bias force to the storage cell to lower the cell's energy level to match the reference node's Fermi level.



Here's an example of how we do it. Suppose we have a cell with a relatively high probability of being occupied that has an energy of  $\Delta E$  relative to the Fermi level. If we just lowered the barrier we would dissipate order  $kT \ln N$  energy. But instead, we first apply a bias force to the island (with a nearby electrode) so as to align the energy of the cell in question with the Fermi level of the reference electrode. Then...



...we lower the barrier enough for tunneling to occur through it. This randomizes the cell contents but does not dissipate any energy except that the 1 bit's worth of known information in the bit now becomes entropy, if it was not entropy already. Now, we can un-bias the cell, allowing its energy to gradually return to the previous high level. If we do this slowly enough so that there is non-negligible tunneling through the barrier, the cell occupancy will stay at equilibrium with the states outside. At the end, the cell occupancy will be at a low value due to the Fermi-Dirac distribution, at which point we re-raise the barrier to stop tunneling and lock it in. In this process we have effectively isothermally compressed the electron gas into the smaller phase-space volume available to it when the cell is excluded. This is equivalent to the isothermal compression step in the Carnot cycle. The adiabatic theorem of quantum mechanics guarantees that such a process dissipates asymptotically zero energy in the limit as the process is performed more slowly. So, the only free-energy loss was in the step where our bit was randomized – in this step, 1 bit's worth of entropy was generated. As we compress this entropy out of the system it must go to the external environment and so we must dissipate  $kT \ln 2$  energy to the environment in order to do this.

## Reversible Computing

- A *reversible* digital logic operation is:
  - Any operation that performs an invertible (one-to-one) transformation of the device's local digital state.
- Landauer's principle only limits the energy efficiency of ordinary *irreversible* (many-to-one) logic operations.
  - Reversible logic operations can dissipate much less energy,
    - Since they can be implemented in a thermodynamically reversible way.
- In 1973, Charles Bennett (IBM Research) showed how any desired computation can in fact be performed using *only* reversible operations (with no bit erasure).
  - This opened up the possibility of a vastly more energy-efficient alternative paradigm for digital computation.
- After 30 years of sporadic research, this idea is finally approaching the realm of practical implementability...
  - Making it happen is the goal of the RevComp project at UF.

Now, can even this  $kT \ln 2$  energy dissipation be avoided? It can, but only if we avoid losing track of known bits and thus turning them into entropy. We can avoid this if we stick to reversible operations only, which transform the local digital state in a one-to-one fashion. Bennett of IBM showed in '73 that doing this does not preclude you from still doing any desired digital computation, although it does increase algorithmic complexity (in device-cycles) in some circumstances. Today the idea is finally starting to near the realm of practicality, and that's what we're trying to do in our project – show that it's practical for near-term applications.

## Adiabatic Circuits

- Reversible logic can be implemented using fairly ordinary voltage-coded CMOS VLSI circuits.
  - With changes to the logic-gate/circuit architecture.
- We avoid dissipating circuit node energies when switching, by transferring charges in an *adiabatic* (lit. “without flow of heat”) fashion.
  - *I.e.*, asymptotically thermodynamically reversible.
- There are many designs for purported “adiabatic” circuits in the literature, but most of them contain fatal flaws.
  - Many designers are unaware of (or accidentally fail to meet) all of the requirements for true thermodynamic reversibility.

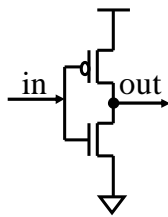
In our project, we implement reversible computing using traditional CMOS devices, in a technique known as Adiabatic Circuits. There has been a lot of literature on this subject since the early 90’s. (It first started being explored in the ’80s.)

## Conventional Logic is Irreversible

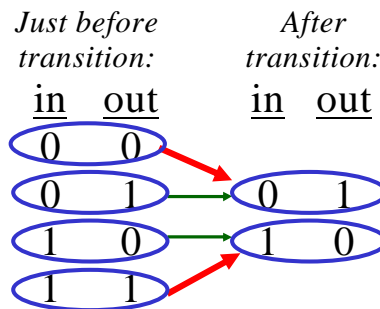
- Logic gate behavior (upon receiving new input):
  - Performs many-to-one transformation of local state!
  - $\therefore$  required to dissipate  $\geq kT$ , by Landauer principle
  - Incurs  $\frac{1}{2}CV^2$  energy dissipation in 2 out of 4 cases.

### Example:

Static CMOS Inverter:



### Transformation of local state:

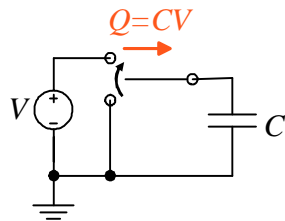


It is important to realize that even traditional one-to-one Boolean operations such as NOT are many-to-one as they are traditionally implemented, because they overwrite their last output on every cycle. So with an inverter, when a new input comes in but the output has not yet had time to change, there are briefly 4 possible states of the device. The operation of this device compresses the logical state space down to only two stable states. So, we know that at least 1 bit's worth of entropy needs to be generated in this process. In fact it is the two transitions that change the output bit that are dissipative;  $\frac{1}{2} CV^2$  energy is dissipated in these cases.

## Conventional vs. Adiabatic Charging

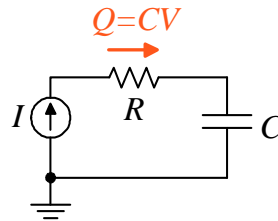
For charging a capacitive load  $C$  through a voltage swing  $V$

- **Conventional charging:**
  - Constant voltage source
- **Ideal adiabatic charging:**
  - Constant current source



- Energy dissipated:

$$E_{\text{diss}} = \frac{1}{2} CV^2$$



- Energy dissipated:

$$E_{\text{diss}} = I^2 R t = \frac{Q^2 R}{t} = CV^2 \frac{RC}{t}$$

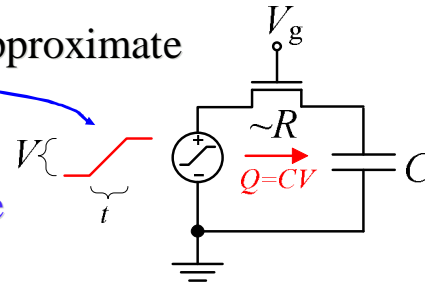
**Note:** Adiabatic beats conventional by advantage factor  $A = t/2RC$ .

So now, what is the difference between conventional and adiabatic charging. Conventionally we switch a bit by connecting it to a constant-voltage reference node at the desired voltage. It is easy to show that  $\frac{1}{2} CV^2$  energy is dissipated in this process, where  $V$  is the voltage difference and  $C$  is the node capacitance. This is independent of the resistance of the switch or the speed of the transition. In ideal adiabatic charging, we use a constant-current source instead, and then the dissipation through a resistive charging path scales down as the charging time is increased (since the current is decreased, and the power goes down as the square of the current). So you can see that the power is less than conventional by the factor  $t/2RC$ .



## Adiabatic Switching with MOSFETs

- Use a voltage ramp to approximate an ideal current source.
- Switch conditionally, if MOSFET gate voltage  $V_g > V + V_T$  during ramp.
- Can discharge the load later using a similar ramp.
  - Through the same, or a different path.



$$t \gg RC \Rightarrow E_{\text{diss}} \rightarrow CV^2 \frac{RC}{t}$$

$$t \ll RC \Rightarrow E_{\text{diss}} \rightarrow \frac{1}{2} CV^2$$

Exact formula:  
 $E_{\text{diss}} = s[1 + s(e^{-1/s} - 1)]CV^2$   
 given speed fraction  
 $s \equiv RC/t$

Athas '96, Tzartanis '98

Now, in practice we approximate the ideal current source by using a variable voltage source that ramps up between two voltage levels over a time  $t$ . This gives approximately the same result. These variable-voltage signals are what we need our external resonator for.

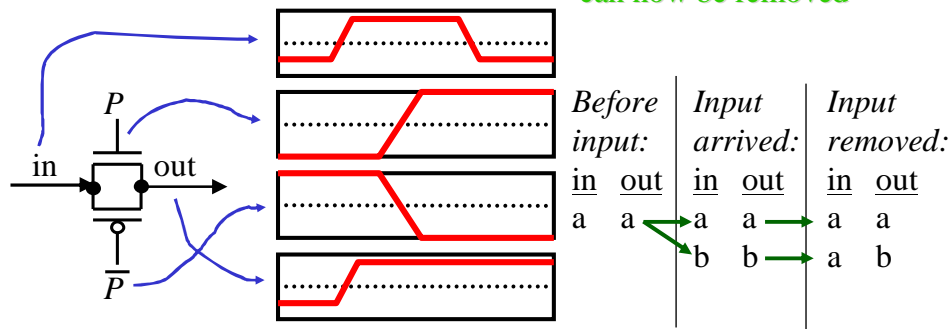
## Requirements for True Adiabatic Logic

- Avoid passing current through diodes.
  - Crossing the “diode drop” leads to irreducible dissipation.
- Follow a “dry switching” discipline (in the relay lingo):
  - Never turn on a transistor when  $V_{DS} \neq 0$ .
  - Never turn off a transistor when  $I_{DS} \neq 0$ . ← Important but often neglected!
- Together these rules imply:
  - The logic design must be logically reversible
  - Transitions must be driven by a quasi-trapezoidal waveform
    - It must be generated resonantly, with high  $Q$
- Leakage power should also be kept manageable.
  - Because of this, the optimal design point will *not* necessarily use the smallest devices that can ever be manufactured!

Now, in order for a logic circuit to be truly adiabatic, you have to follow certain rules. First, never pass current through diodes, so, you have to avoid bipolar transistors, for example – field-effect transistors only! Next, you have to follow a “dry switching” discipline as the old relay electronics people used to call it. To prevent sparking and corrosion on contacts, you never turn on a switch when there is a voltage across it, and never turn it off when there is a current through it. Many of the so-called “adiabatic” circuits you find in the literature actually break the second rule and so are not truly adiabatic. Anyway, together these rules imply that the logic design has to be reversible (since there is no way to erase information under these constraints), and it also turns out you need a voltage waveform that has flat tops and bottoms (is quasi-trapezoidal) because otherwise you’ll never be able to turn off a transistor since there will always be a current through it (except for infinitesimal moments). Finally, the voltage waveform needs to be generated resonantly, with high  $Q$ . Leakage power consumed by nominally turned-off transistors also needs to be kept low, and I believe that because of this the scaling of devices will soon halt. The optimal devices will *not* necessarily be the smallest ones we can manufacture, since these have high leakage rates.

## A Simple Reversible CMOS Latch

- Uses a standard CMOS *transmission gate* (T-gate)
- Sequence of operation:
  - (1) input level initially matches latch 'contents' (output);
  - (2) input changes → output changes; (3) latch closes, charge is stored dynamically (floats); (4) input signal can now be removed

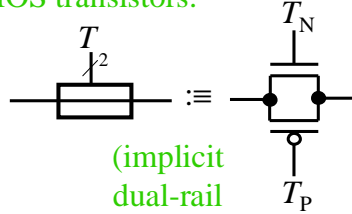


Here's an illustration of how to build an adiabatic latch in CMOS. We can do it with an ordinary transmission gate. Initially the input and output nodes are at the same level and the gate is on. Then the input gradually transitions to the desired level. Then we hold the input steady while we gradually turn the gate off. Then we can retract the input (restore it to some standard state representing "no information"), and the data is latched into place. By reversing this sequence of operations we can adiabatically "unlatch" the information.

## 2LAL: 2-level Adiabatic Logic

A pipelined fully-adiabatic logic invented at UF (Spr. 2000), implementable using ordinary CMOS transistors.

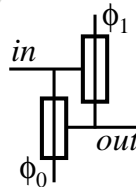
- Use simplified T-gate symbol:



- Basic buffer element:

- cross-coupled T-gates:

- need 8 transistors to buffer 1 dual-rail signal

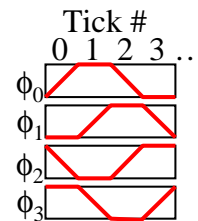


(implicit dual-rail encoding)

- Only 4 timing signals  $\phi_{0-3}$  are needed; only 4 ticks per cycle:

- $\phi_i$  rises during ticks  $t \equiv i \pmod{4}$

- $\phi_i$  falls during ticks  $t \equiv i+2 \pmod{4}$

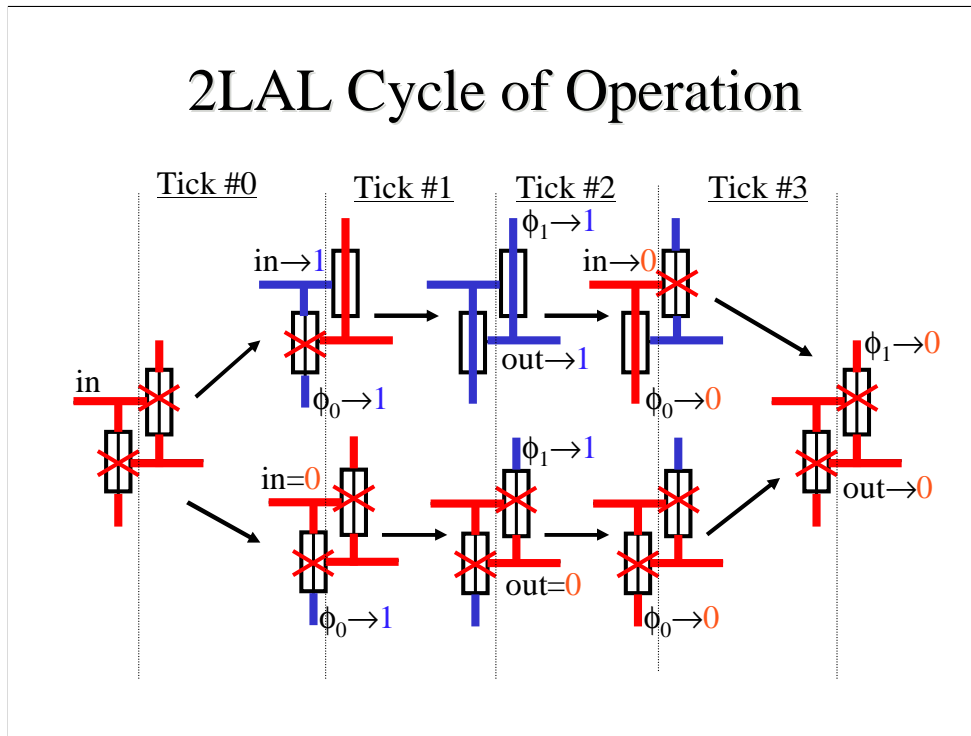


Anyway here is a simple adiabatic logic scheme, here demonstrated using ordinary CMOS transistors, that is based on a new operation paradigm we discovered in Spring of 2000 (called input-barrier, clocked-bias latching). For convenience, we use Hall's electroid (switch) symbol, which can be implemented in CMOS with a parallel nFET/pFET pair (transmission gate).

There is a basic clocked buffer element consisting of a pair of cross-coupled switches.

This logic scheme is more economic than many previous ones because it requires only 4 global timing signals, really just 4 different phases of a single waveform. These are shown in the timing diagram. The top and bottom portions must be flat for at least a full tick. The shape of the transitions is arbitrary (though the slope should be finite everywhere and should scale down with increasing tick length).

## 2LAL Cycle of Operation

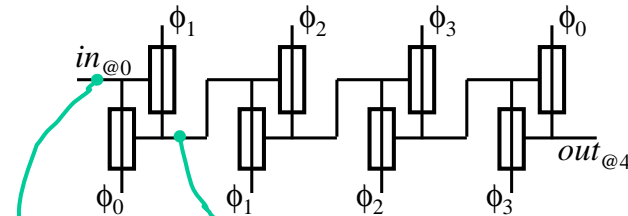


Here is the cycle of operation of the buffer gate in the 2LAL scheme. Initially, all signals are low (red, 0) and the switches are off. Then in tick 0, the input transitions to 1 (at the same time as  $\phi_0$ ), and the output switch turns on, or not (input conditionally lowers barrier). Now in tick 1,  $\phi_1$  goes high (unconditional bias) taking the output with it, or not. This turns on the reverse switch, or not. (If so there is no dissipation since the input is at the same level as  $\phi_0$ .) In tick 2, the input is retracted from its source (and also simultaneously by  $\phi_0$  in the upper case), turning off the output switch (unconditional barrier raising). Now the output information is latched into place. Finally in tick 3  $\phi_0$  reverts to its low state which does not affect anything inside the circuit but prepares us to be able to turn on the forward switch again in the next cycle. Meanwhile, the next gate in the chain restores the output to the zero level. (This particular gate is intended to be used as part of a pipeline of similar gates.)

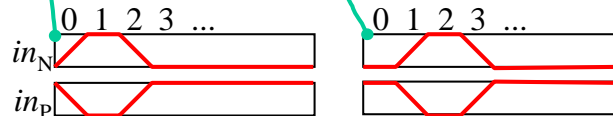
## 2LAL Shift Register Structure

- 1-tick delay per logic stage:

Animation:



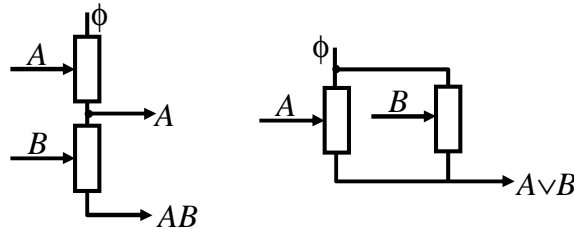
- Logic pulse timing and signal propagation:



Here is how to build a shift register of 2LAL buffers: Just connect them together with incrementing phases on successive clock signals. A pulse introduced at the input will propagate down the chain, 1 stage per tick. If CMOS transmission gates are used for switches, then dual-rail logic must be used.

## More Complex Logic Functions

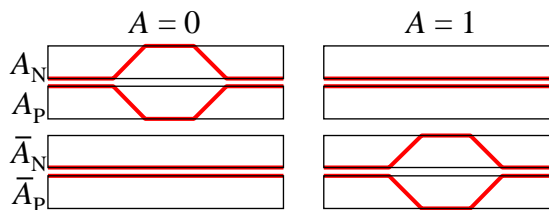
- Non-inverting Boolean functions:



- For inverting functions, we must use a quad-rail logic encoding:

– To invert, just swap the rails!

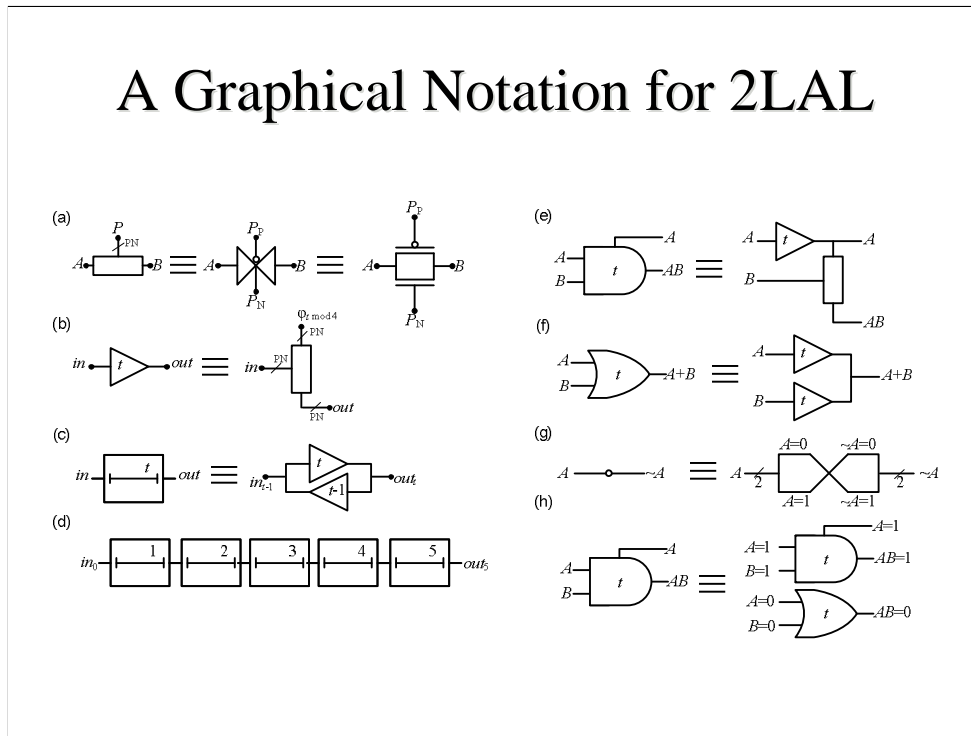
- Zero-transistor “inverters.”



How about more complex functions? Again, series/parallel combinations of input-controlled switches will do the job. (Forward parts shown.) However, one must remember that information on internal nodes (such as the A output of the left circuit) must also be retracted by subsequent gates. Inputs that are not echoed in the output (e.g. B in both these examples) must be retracted separately by some other circuit.

The easiest way to do inverting functions is to use a dual-rail encoding: a pulse on one wire represents 0, while a pulse on another represents 1. (Quad-rail encoding is shown since this is needed if switches are implemented using CMOS transmission gates.) Then a NOT gate is just a renaming of rails. Dual-rail has the further advantage of allowing the total magnitude of back-reactions on the resonant driver to be data-independent.

# A Graphical Notation for 2LAL



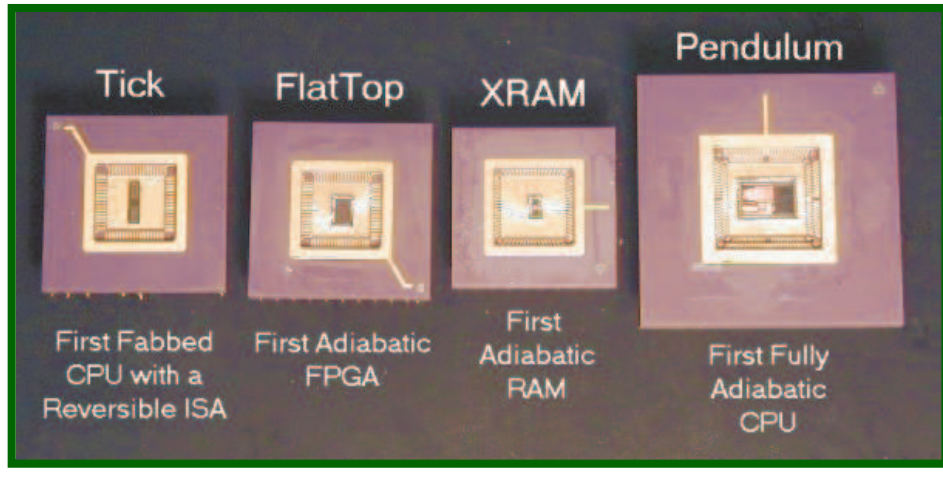
Here is a graphical notation we are using for 2LAL elements as we build our prototype circuits in Cadence.

(a) Fundamental to 2LAL is the CMOS transmission gate, a parallel nFET/pFET pair whose control signal  $P$  is implicitly always a dual-rail pair of active-high (N) and active-low (P) logic signals. (b) A 4-transistor 2LAL buffer for dual-rail pulsed signals consists of two parallel transmission gates controlled by the input, passing a power-clock signal  $\phi_t \bmod 4$  and (implicitly in this drawing) its complementary,  $180^\circ$ -out-of-phase signal  $\phi_{(t+2)} \bmod 4$ . The semantics is that if  $in$  pulses before tick  $\#t$ ,  $out$  will pulse  $@t$  (at tick  $\#t$ ), else it will stay at its initial level (arranged to be F). (c) An 8-transistor adiabatic delay element that moves an input pulse  $@t-1$  to an output pulse  $@t$ . (d) Delay elements with subsequent tick numbers can be chained to make a shift register for input pulses. (e) An AND gate for pulses (8 transistors) consists of two transmission gates in series, and its internal node must be explicitly recognized as an extra output to maintain reversibility. (f) An 8-transistor OR gate for pulses consists of simultaneous transmission gates in parallel. (g) a zero-delay, zero-transistor, non-amplifying NOT bubble is implemented using quad-rail signaling; logic signal  $A$  is implemented as a pair of pulse signals,  $A=0$  and  $A=1$ . A simple renaming of wires suffices to translate  $A=0$  to  $\sim A=1$  and  $A=1$  to  $\sim A=0$ . (h) When fed a quad-rail input signal, an AND gate icon denotes a 16-transistor parallel pair of an AND and an OR (to compute  $AB=0$  pulse). For all the logic gates, inputs may be consumed, if desired, by adding  $@t-1$  reverse buffer elements, like in the delay element c.



## Reversible and/or Adiabatic VLSI Chips Designed @ MIT, 1996-1999

By Frank and other then-students in the MIT Reversible Computing group,  
under CS/AI lab members Tom Knight and Norm Margolus.



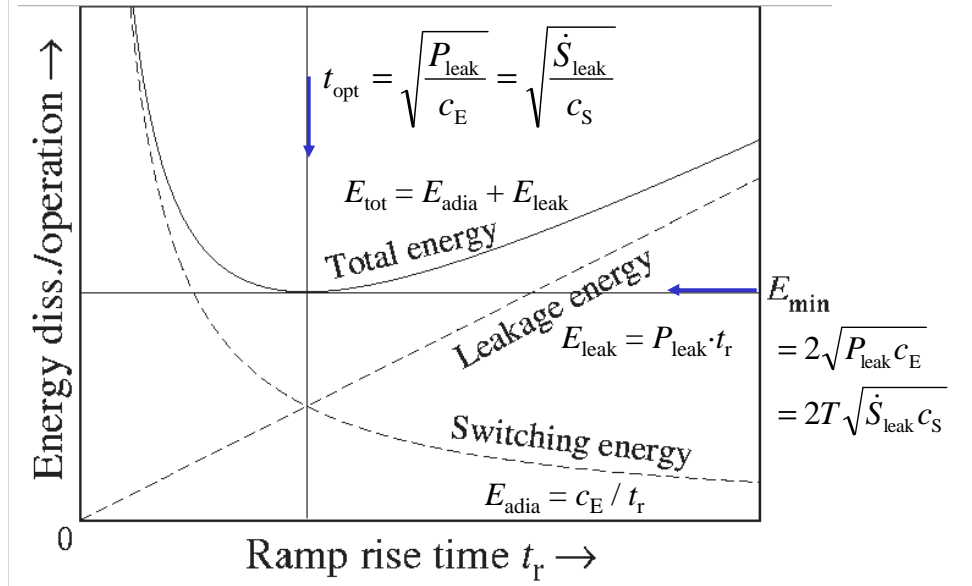
So anyway, using another (more complicated, and buggy) adiabatic logic style, my collaborators and I built these reversible chips at MIT (under Tom Knight and Norm Margolus) to demonstrate that the architectural problems of reversible logic are straightforward to solve. Tick (by myself and Scott Rixner) was a simple non-adiabatic microprocessor implemented in standard CMOS that nevertheless implemented a logically reversible machine-language instruction set, as a proof-of-concept and basis for comparison with fully-adiabatic circuits. FlatTop (me, Nicole Love, Carlin Vieri, Josie Ammer) was the first scalable, universal, fully-adiabatic reconfigurable processor, capable of efficiently simulating 2D and 3D reversible circuits of arbitrary complexity when tiled in large arrays. (FlatTop works by simulating in SCRL the Margolus BBMCA cellular automaton which itself simulates Fredkin's BBM billiard ball model of reversible computing which itself simulates reversible logic networks composed of Fredkin gates, which themselves can simulate arbitrary reversible CAs – the simulation is so indirect that it is not very efficient, but it is universal for reversible CAs with “only” constant-factor overhead. Anyway it is just a proof of concept.) XRAM (Carlin, Josie) was an adiabatic memory with a reversible interface (though I have since invented far more efficient ones), and Pendulum (Vieri, with ISA mods from me) was a complete, fully-adiabatic, MIPS-style microprocessor.

This chip-design work (the Pendulum project, DARPA-funded under the Scalable Computing Systems Program) demonstrated that reversible logic is by no means incompatible with traditional styles of computer organization. It only requires a fairly minor translation of traditional architectural methods.

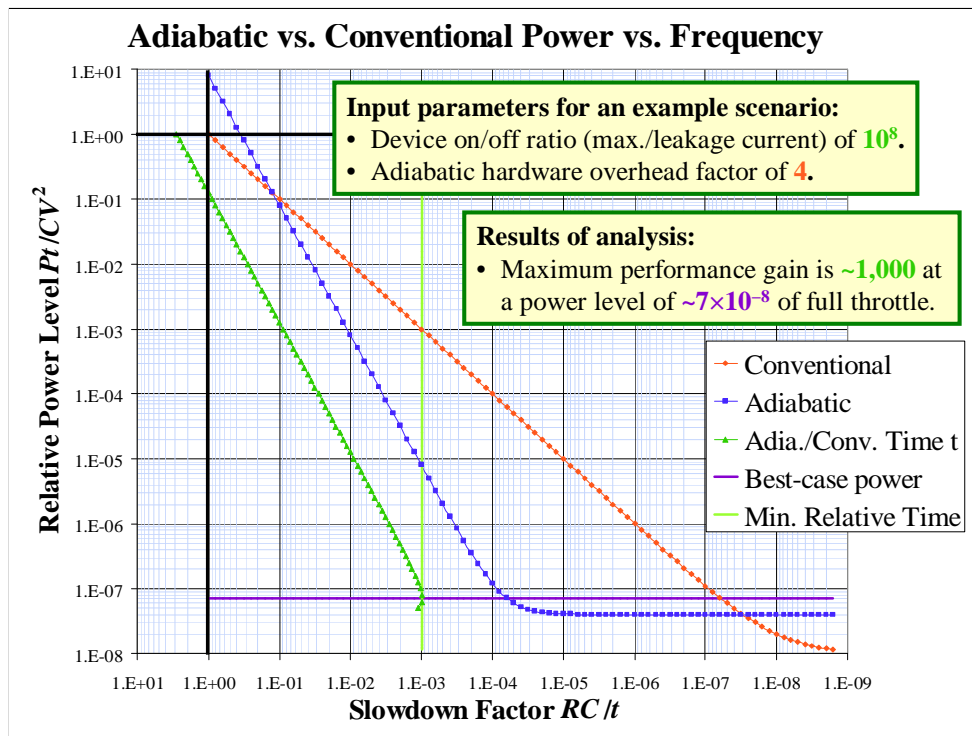
However, this work begged the question: Can reversible computing ever really be cost-effective? Can the overheads of reversible and adiabatic operation ever be outweighed by their energy savings?

It was the goal of my subsequent research to answer this question, using a principled systems-engineering methodology.

## Minimum Losses w. Leakage



One limit on the energy dissipation of adiabatic circuits is due to energy leakage. This happens via several mechanisms in CMOS including thermally-activated subthreshold conduction and tunneling. Any structured system will have some non-zero residual rate of entropy generation to the environment. Anyway, whatever the rate  $P_{leak}$  of energy leakage is, if we also know the energy coefficient (ratio between energy and frequency) in the technology, we can derive an optimal frequency for minimum energy dissipation per operation. If we wish to go beyond this point, we must decrease the leakage rate. Doing this will eventually require making devices larger rather than smaller! (If we stick with standard CMOS devices.)



Now, slowing down devices in order to save energy may sound impractical. But in fact, if power constraints are a limiting factor on performance, it can actually allow us to run faster! Here's why. In conventional technology, power goes down only linearly with frequency. But in adiabatic technology, it goes down quadratically. So if we have a fixed power constraint per-device, corresponding to a horizontal line on the chart, you see that the adiabatic technology does not have to be slowed down as much as the conventional one. The point is that in the long run, power will become more and more of a limiting factor on performance as we can afford to build more and more devices but we cannot power them at their maximum frequency! The power constraint per device will move lower and lower as we assemble greater numbers of devices. The adiabatic approach becomes more and more beneficial, limited only by leakage which therefore must be addressed, for example by keeping devices large enough so that it is negligible.

## Adiabatic Performance Boost

- Approx. performance gain factor of adiabatics, given power level is:

$$g_{\text{adia}} = \frac{\sqrt{P_{\text{g,full}} \left( \frac{P_{\text{g,max}}}{O_{\text{adia}}} - P_{\text{g,lk}} \right)}}{2(P_{\text{g,max}} - P_{\text{g,lk}})}$$

- Where:

- $P_{\text{g,full}} = E_{\text{g,sw}} f_{\text{max}}$  = “Full throttle” switching power per logic gate,
  - $E_{\text{g,sw}} = CV^2$  switching energy per logic gate
  - $f_{\text{max}}$  = “Full throttle” switching frequency  $1/RC$  of gates
- $P_{\text{g,max}}$  = Maximum allowed power dissipation per gate, imposed by constraints on application’s power and/or cooling system
- $O_{\text{adia}}$  = Hardware overhead factor of adiabatic logic design
- $P_{\text{g,lk}}$  = Leakage power dissipation per gate in given technology

- This is maximized when  $P_{\text{g,max}} = P_{\text{g,lk}}(2O_{\text{adia}} - 1)$ , in which case we have:

- where  $R_{\text{on/off}} = I_{\text{on}}/I_{\text{off}} = P_{\text{full}}/P_{\text{lk}}$

$$g_{\text{adia}} = \frac{1}{4} \sqrt{\frac{R_{\text{on/off}}}{O_{\text{adia}} (O_{\text{adia}} - 1)}}$$

- This is  $>1$  when

$16 O_{\text{adia}}(O_{\text{adia}} - 1) < R_{\text{on/off}}$  of transistor technology

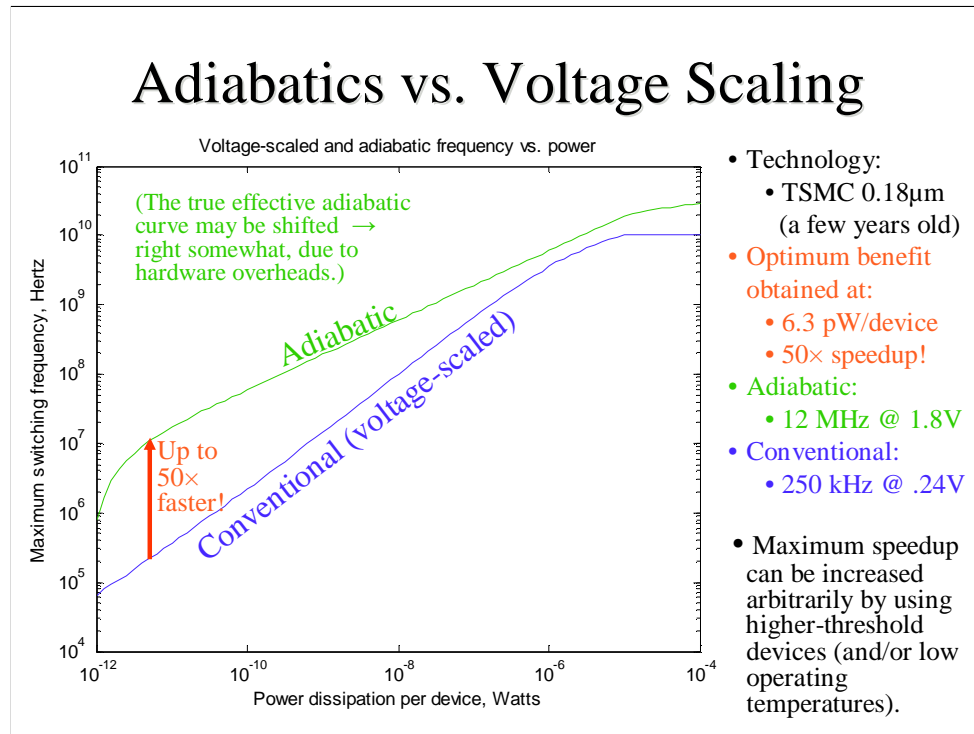
This is the analysis behind the previous slide. I skipped this in my talk.

## Example Ultra-Low-Power Scenario

- Technology scenario:
  - ITRS hp65 (65 nm half-pitch) technology node (expect ~2007).
- Application scenario:
  - 1Mgate processor chip (e.g. TI's C6000 line of GHz DSPs)
  - Requirement for  $\lesssim 100 \mu\text{W}$  processor power dissipation
- Leakage per NAND gate in hp65:  $\sim 13 \text{ pW}$ .
  - $\therefore$  chip would dissipate  $\sim 13 \mu\text{W}$  even at zero frequency!
- Irreversibly switching NAND gate's output takes 250 aJ.
  - 1Mgate chip dissipates 250 pJ per clock cycle
    - $87 \mu\text{W}$  switching power constraint  $\rightarrow$  max. freq.  $\sim 350 \text{ kHz}$ !
    - Max. NAND transition rate = 23 GHz, slowdown  $\sim 66,000 \times$
- **Adiabatic solution:** Using overhead factor  $4\times$ 
  - Run clock at 35 MHz instead of 350 kHz ( **$100\times$  faster!**)
    - But note this is still 660 times slower than max transition frequency.
  - $\therefore$  each switching op dissipates only  $\sim 1/660^{\text{th}}$  the  $CV^2$  energy
    - Or,  $\sim 1/160^{\text{th}}$  even after the  $4\times$  logic overhead is included
  - Leakage power:  $\sim 50 \mu\text{W}$ , switching power:  $\sim 50 \mu\text{W}$ .

A numerical example. Speaks for itself.

# Adiabatics vs. Voltage Scaling



**Maximum frequency vs. power dissipation for adiabatic (upper line) vs. conventional voltage scaling.** Results based on an optimization analysis using a standard device model for the TSMC 0.18 $\mu$ m process technology. At low power levels, the conventional voltage-scaling approach suffers from reduced drive current (increased effective channel resistance) at low supply levels, which limits the maximum operating frequency to a level that is at most roughly proportional to power. In contrast, the adiabatically switched device can continue to be operated at the recommended voltage of the technology (1.8 V), while performance falls off more slowly, roughly with only the square root of the power drop. Near the left of the figure, you can see that by the time we reach an ultra-low-power level of  $6.3 \times 10^{-12}$  W (10 pW) per device, near the lower limit set by leakage power, the adiabatic device is running at  $\sim 50\times$  the conventional one's frequency (12.7 MHz vs. 260 kHz) in this particular analysis.

**UF CONFIDENTIAL – PATENT PENDING**

## **MEMS Resonator Concept**

For discussion, see the MLPD '04 paper.

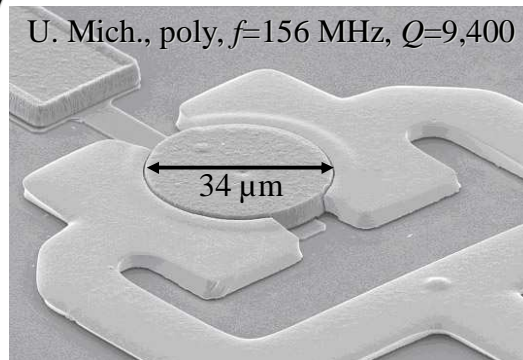
## The Power Supply Problem

- In adiabatics, the factor reduction in energy dissipated per switching event is limited to (at most) the  $Q$  factor of the clock/power supply.
- Electronic resonator designs typically have low  $Q$  factors, due to considerations such as:
  - Energy overhead of switching a clamping power MOSFET to limit voltage swing of an  $LC$  circuit.
  - Low coil count of integrated inductors.
  - Unfavorable scaling of inductor  $Q$  with frequency.
- Our solution:
  - Use electromechanical resonators instead!



## MEMS/NEMS Resonators

- State of the art of technology demonstrated in lab:
  - Frequencies up to the 100s of MHz, even GHz
  - $Q$ 's  $>10,000$  in vacuum, several thousand even in air!
- Rapidly becoming an important technology for commercial RF filters, *etc.*, in communications SoC (Systems-on-a-Chip) *e.g.* for cellphones.

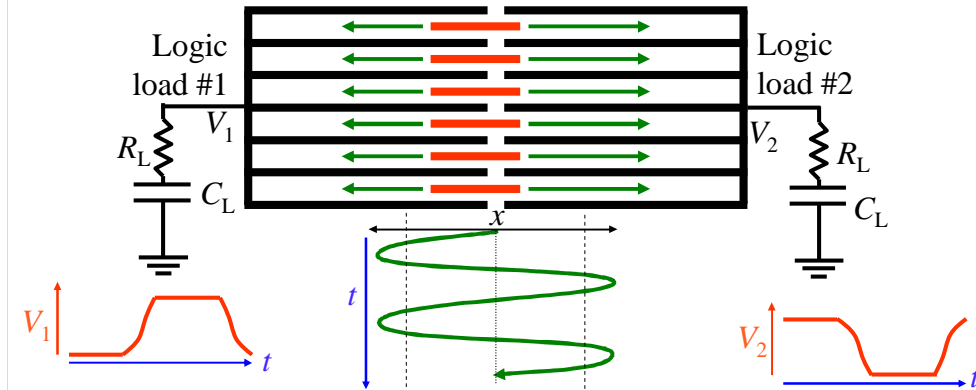


Here is a photo I stole off the web, of a MEMS disc resonator (operates in an expansion/contraction vibrational mode, in which there is a node of motion at the center support point, for low losses). Some resonator structures have been experimentally validated at frequencies up to hundreds of MHz and even GHz, with  $Q$ 's of up to and over 10,000 in vacuum, and several thousand even in air. This is today emerging as a real-world commercial technology for embedded systems-on-a-chip for wireless communications, *e.g.*, chips in cellphones, which need high- $Q$  resonators to build good filters and amplifiers.

Perhaps we could do even better all-electronic resonators using superconducting technology, based on some of the superconducting quantum computing talks on Friday (multi-GHz systems with  $Q$ 's up to 100,000!!)

## The Core Concept

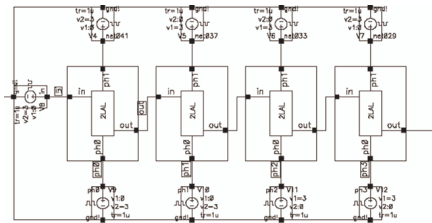
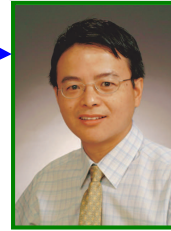
- Imagine a set of charged plates whose horizontal position oscillates between two sets of interdigitated fixed plates.
  - Structure forms a variable capacitor and voltage divider with the load.
- Capacitance changes significantly only when crossing border.
  - Produces nearly flat-topped (quasi-trapezoidal) output waveforms.
  - The two output signals have opposite phases (2 of the 4  $\phi$ 's in 2LAL)



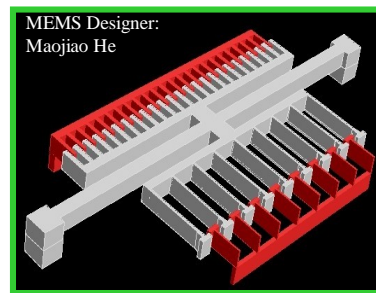
**UF CONFIDENTIAL – PATENT PENDING**  
**MEMS Resonant Power Supply for  
Ultra-Low-Power Adiabatic Circuits**

A.k.a. The “AdiaMEMS” Project

- Part of CISE’s Reversible & Quantum Computing group
  - Collab. with Huikai Xie (MEMS, ECE dept.)
- **Goal:** Demonstrate orders-of-magnitude improvement in power-performance efficiency of digital CMOS circuits.
  - Based on reversible logic in adiabatic circuits powered by high-quality custom microelectromechanical resonators.
- **Funding:** \$40K grant from SRC’s Cross-Disciplinary Semiconductor Research (CSR) Program



VLSI designer: Krishna Natarajan



MEMS Designer:  
Maojiao He

Some notes:

Adiabatic (thermodynamically reversible) circuits are a technique for low power that has been studied within the EE community for some time. But, the major drawback of most adiabatic logic families studied to date is that the inefficiency of conventional power supplies severely limits the energy savings that can be achieved. We are designing custom MEMS (microelectromechanical systems) resonators that are shape-tailored to provide the exact wave shape needed in order to drive adiabatic circuits, with a Q (quality) factor in the thousands. This translates into potential energy savings in the logic of hundreds or thousands of times less dissipation than in conventional CMOS. The project uses rigorous principles of reversible computing theory, which can be considered an offshoot of computational complexity theory, to analyze the design tradeoffs and maximize system-level cost-efficiency in the face of power limits, taking into account the space and time overheads imposed by adiabatic charging and reversible logic. We have done a long-term analysis of technology trends, together with fundamental limits, which indicates that reversible computing techniques will become an absolutely unavoidable physical necessity for improving computer power-performance (and thus practical cost-performance) beyond the next two or three decades. Our goal in this project is to demonstrate the feasibility of these design principles by using today’s manufacturing technology as the basis of a detailed design and feasibility study for an ultra-low-power digital signal processor. We are designing and manufacturing simple prototype parts as a proof-of-concept to concretely demonstrate the energy savings that can be achieved by these methods.

## Key Characteristics of Resonator

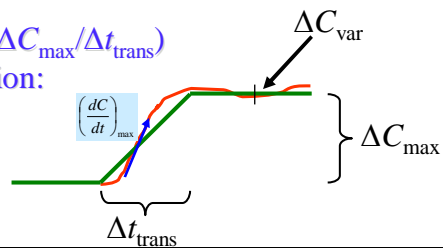
- **Goal:** Produce a near-ideal trapezoidal output voltage waveform resonantly, with high  $Q$ .
- To be optimized w. logic: Resonant frequency  $f$ .
- Key figures of merit:
  - Effective quality factor:  $Q_{\text{eff}} = E_{\text{trans}}/E_{\text{diss}}$
  - Area efficiency:  $E_A = E_{\text{trans}}/A$ .
- Key figures of demerit:

- Maximum transition slope:

$$s_{\text{max}} = (dC/dt)_{\text{max}} / (\Delta C_{\text{max}}/\Delta t_{\text{trans}})$$

- Fractional capacitance variation:

$$v_C = \Delta C_{\text{var}} / \Delta C_{\text{max}}$$



UF CONFIDENTIAL – PATENT PENDING

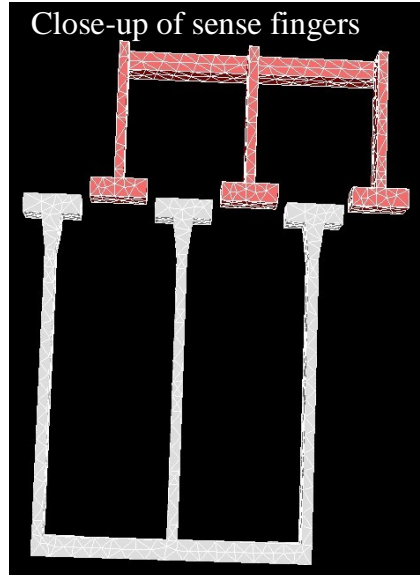
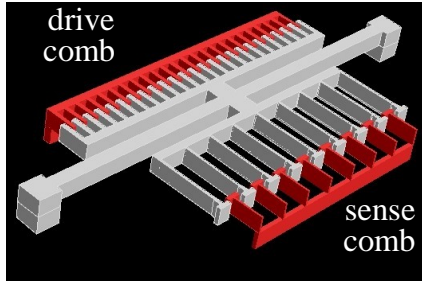
## First MEMS Technology Tried

- MEMS process donated by Robert Bosch corp.
- It is a thin-film technology
  - Though a multi-layer, bulk single-crystal process can be expected to do better.
- Integrated CMOS/MEMS devices will eventually be available in this process.
  - However our initial design was dual-die
    - CMOS side was not mature yet in this process
- Minimum horizontal thickness:  $\lambda = 0.5 \mu\text{m}$
- Minimum horizontal gap size:  $d = 0.1 \mu\text{m}$

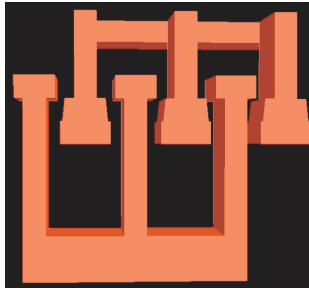
UF CONFIDENTIAL – PATENT PENDING

# Some Early Resonator Designs

By Ph.D. student Maojiao He, under supervision of Huikai Xie



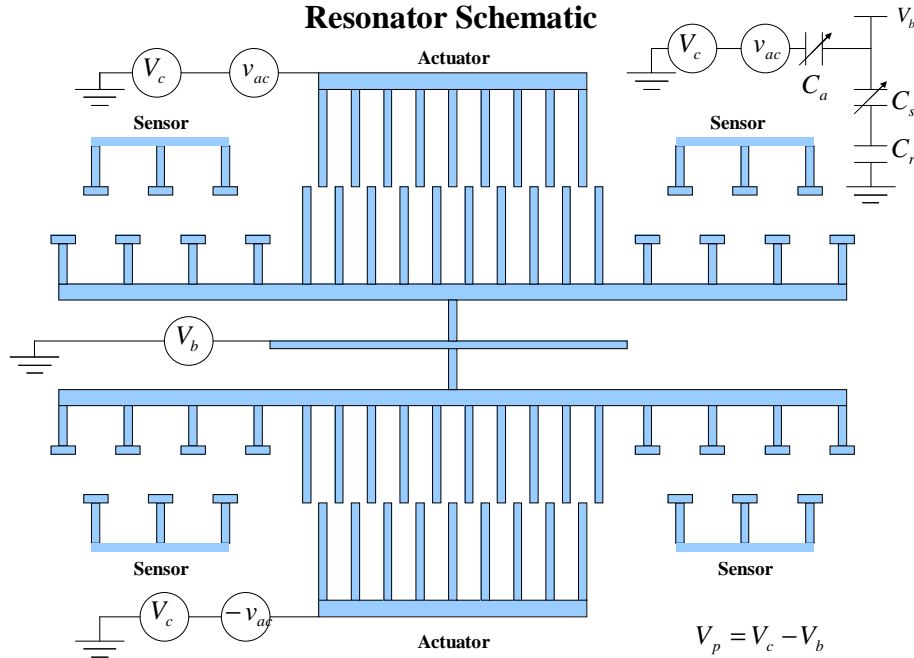
Another  
finger  
design



**UF CONFIDENTIAL – PATENT PENDING**

## Analysis of initial MEMS design

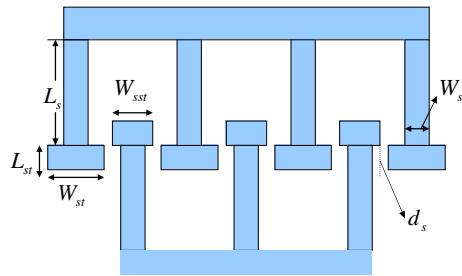
**UF CONFIDENTIAL – PATENT PENDING**  
**Resonator Schematic**





**UF CONFIDENTIAL – PATENT PENDING**

**Sensor Design**



$$d_s = d \quad W_s = \lambda$$

$$L_{st} = \lambda \quad X = 8 L_{st}$$

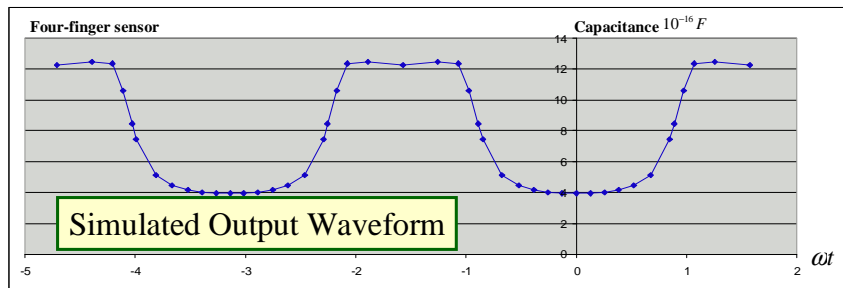
$$W_{st} = W_s + 4 d$$

$$W_{sst} = W_s + 2 d$$

$$L_s \gg d \quad (L_s = 20 d)$$

(Early design w. thin fingers)

$$4 C_{sf} \approx 8 \times 10^{-16} F$$



**UF CONFIDENTIAL – PATENT PENDING**

**Parameter Definitions**

$\lambda$	--Minimum feature size	Defined by technology	$\lambda = 0.5\mu m$
$d$	--Minimum gap size	Defined by technology	$d = 0.1\mu m$
$V_b$	--Bias voltage	Determined by load and limited by breakdown voltage between sense electrodes [1]	$V_b < (110V / \mu m) \times d$
$V_p$	--Drive voltage	Limited by breakdown voltage between sense electrodes [1]	$V_p < (110V / \mu m) \times d$ $V_p = 10V$
$v_{ac}$	--AC component of drive voltage	$v_D(t) = V_p + v_{ac} \sin(\omega t)$	$v_{ac} = 0.2V$
$Q$	--Quality factor	Decided by damping (air, structure) (5,000-10,000)	$Q = 5,000$
$\omega$	--Resonant frequency	Required by system	$\omega = 2\pi \times 500kHz$
$X$	--Vibration amplitude	From CoventorWare Simulation	$X = 8L_{st} = 4\mu m$

**UF CONFIDENTIAL – PATENT PENDING**

**Effective Quality Factor and Optimization**

---

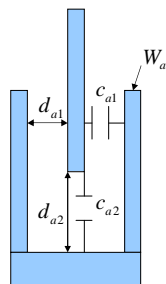
$$\begin{aligned}
 Q_{eff} &= \frac{E_{tfr}}{E_{diss}} = \frac{2QC_s V_b^2}{M_{eq} (\omega X)^2} = \frac{2QN_s C_{sf} V_b^2}{M_{eq} (\omega X)^2} = \frac{2QC_{sf} V_b^2}{M_{eq} (\omega X)^2} \frac{M_{eq} - M_a}{m_s} \\
 &= \frac{2QC_{sf} V_b^2}{(\omega X)^2} \frac{1 - \frac{M_a}{M_{eq}}}{m_s} = \frac{2QC_{sf} V_b^2}{(\omega X)^2} \frac{1 - \frac{N_a A_{af} \rho h \omega^2 X d_{a1}}{2(N_a - 1)Q \epsilon_0 h V_b v_{ac}}}{A_{sf} h \rho} \\
 &= \frac{C_{sf} V_b^2}{A_{sf} h X} \left( \frac{2Q}{\omega^2 X \rho} - \frac{N_a A_{af} d_{a1}}{(N_a - 1) \epsilon_0 V_b v_{ac}} \right) \\
 &\approx \frac{\epsilon_0 L_{st} V_b^2}{A_{sf} X d_s} \left( \frac{2Q}{\omega^2 X \rho} - \frac{A_{af} d_{a1}}{\epsilon_0 V_b v_{ac}} \right) \quad \text{--Large } N_a \quad C_{sf} \approx \frac{\epsilon_0 L_{st} h}{d_s}
 \end{aligned}$$

Optimize $Q_{eff} \Rightarrow$	Maximize $V_b \quad v_{ac} \quad Q$
	Minimize $d_s \quad d_{a1} \quad A_{af} \quad A_{sf} \quad \rho \quad \omega$

I skipped these supporting analysis slides in my talk.

**UF CONFIDENTIAL – PATENT PENDING**

**Actuator Design**



$$W_a = \lambda \quad d_{a1} = d$$

$$L_a = 2X + \lambda$$

$$d_{a2} = X + \lambda$$

$$\frac{\partial C_{a1}}{\partial x} = 2(N_a - 1) \frac{\epsilon_0 h}{d_{a1}}$$

$$\frac{\partial C_{a2}}{\partial x} = (2N_a - 3) \frac{\epsilon_0 h W_a}{d_{a2}^2}$$

$$\frac{\partial C_a}{\partial x} = \frac{\partial C_{a1}}{\partial x} + \frac{\partial C_{a2}}{\partial x} = \frac{\partial C_{a1}}{\partial x}$$

$$\frac{\partial C_{a1}}{\partial x} \gg \frac{\partial C_{a2}}{\partial x}$$

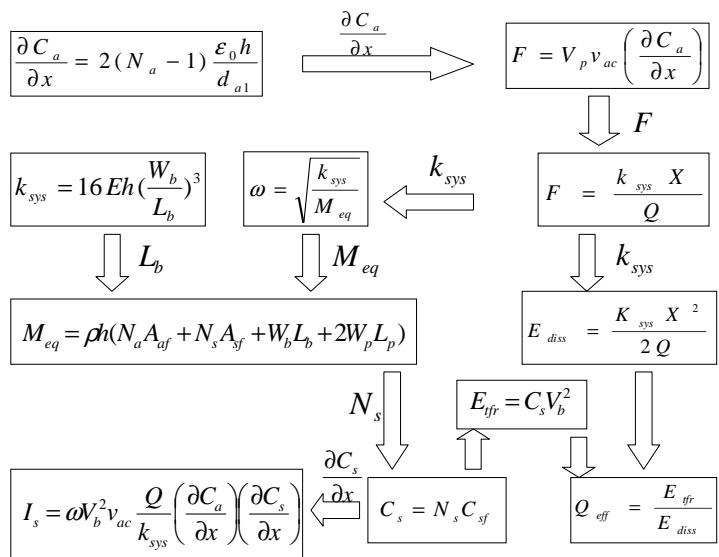
$$\frac{\partial C_{a1}}{\partial x} = 1.346 \times 10^{-9} F / m$$

$$\frac{\partial C_{a2}}{\partial x} = 2 \times 10^{-11} F / m$$

$h$	--Minimum gap size	Defined by technology	$h = 2 \mu m$
$N_a$	--Number of actuation fingers		$N_a = 20$

**UF CONFIDENTIAL – PATENT PENDING**

**Resonator Design Strategy**



**UF CONFIDENTIAL – PATENT PENDING**  
**Energy transferred per area and optimization**

$$\frac{E_{\text{tfr}}}{\text{Area}} = \frac{C_s V_b^2}{L_c (W_b + 2(L_p + 2L_a - X))}$$

$$= \frac{N_s C_{sf} V_b^2}{(N_a (W_a + d_{a1}) + N_s ((W_{st} + W_{sst}) / 2 + d_s)) (W_b + L_a + X + \lambda)}$$

<p>Optimize <math>\frac{E_{\text{tfr}}}{\text{Area}} \Rightarrow</math></p>	<p><b>Maximize</b> <math>C_{sf} V_b</math></p> <p><b>Minimize</b> <math>N_a d_{a1} d_s W_a L_a X</math></p>
---	---

- |  |   |
|--|---|
| <p><math>k_{\text{sys}}</math> --System spring constant</p> <p><math>C_a</math> --Total actuation capacitance</p> <p><math>A_{af}</math> --Area of actuation finger</p> <p><math>A_{sf}</math> --Area of sensing finger</p> <p><math>C_{sf}</math> --Sensing capacitance change per sensing finger</p> | <p><math>W_b</math> --resonator beam width</p> <p><math>L_b</math> --resonator beam length</p> <p><math>W_p</math> --shuttle plate width</p> <p><math>L_p</math> --shuttle plate length</p> <p><math>L_c</math> --comb length</p> |
|--|---|

# Dissipation in Resonator

## Ways to minimize some major sources of dissipation:

- Air damping:
  - Vacuum packaging, small size, or optimize airflow
- Clamping losses to the substrate:
  - Locate support at a nodal point of vibration mode
  - Use impedance-mismatched supports to reflect energy back
- Thermoelastic dissipation:
  - Small size
  - Use high thermal conductivity, stiff materials
  - Utilize modes with uniform compression/expansion
- Surface loss mechanisms:
  - Avoid layered structure at surfaces
- Intrinsic material losses:
  - Prefer single-crystal materials

## Status / Plans for Near Future

- A small prototype resonator design was taped out in a post-CMOS MEMS process (TSMC .35)
  - Will be tested this summer.
- Improved resonator designs afforded by a suitably modified post-CMOS process flow are being developed.
  - I will briefly review some aspects.
- **Next:** Obtain funding (or process donation) for fabricating an integrated CMOS/MEMS test chip (~\$10k).
  - Resonator driving a simple 2LAL shift register or adder pipeline
- Test the various parts separately, & together.
  - Characterize power dissipation using sensitive calorimetry techniques.



# **CMOS-MEMS Process**

**Huikai Xie**

**Department of Electrical and Computer Engineering  
University of Florida  
Gainesville, FL 32611  
Email: [hkxie@ece.ufl.edu](mailto:hkxie@ece.ufl.edu)**

These are my colleague's slides and I went through them fairly quickly.

# Outline

- ❑ Introduction
- ❑ CMOS-MEMS Process
- ❑ 3-D Sensing and Actuation
- ❑ CMOS-MEMS Inertial Sensors
- ❑ Summary

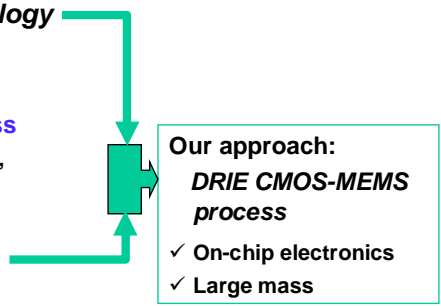
# Why DRIE CMOS-MEMS?

## *Thin-film micromachining technology*

- ✓ On-chip electronics integration
- ✓ Multiple axis integration
- ✓ Size limitation due to residual stress
  - ADI, Bosch, Carnegie Mellon, Samsung, Sandia, UC-Berkeley

## *Bulk micromachining technology*

- ✓ Large mass
- ✓ No integrated interface circuitry
- ✓ Wafer-to-wafer bonding, two-side alignment
  - Bosch, Draper, JPL, Murata, Samsung



Our approach:  
***DRIE CMOS-MEMS process***

- ✓ On-chip electronics
- ✓ Large mass

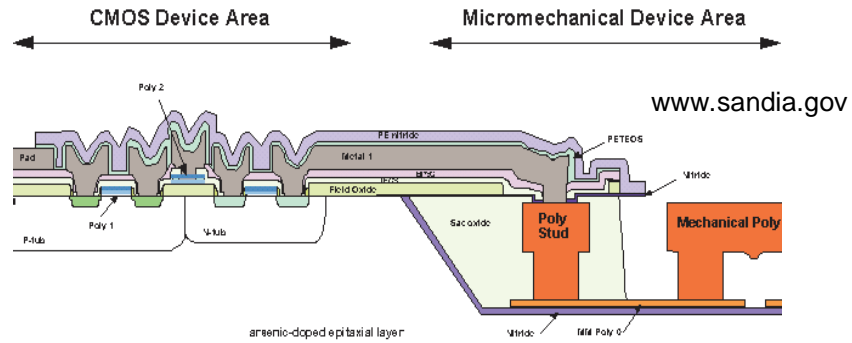
## Why CMOS-MEMS?

- ✓ “Smart” on-chip CMOS circuitry
  - ✓ Multi-vendor accessibility
  - ✓ Scalability
  - ✓ Compact size
  - ✓ More functions
  - ✓ Low cost
- MEMS structures can be made
- Before CMOS processes (“pre-CMOS”)
  - In-between CMOS processes (“intermediate-CMOS”)
  - After CMOS processes (“post-CMOS”)

## CMOS-MEMS Processes

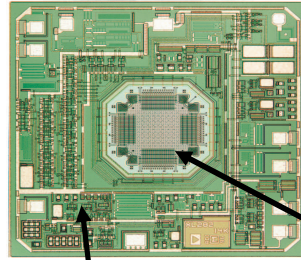
	MEMS planarity	Vendor accessibility	Contamination	Temperature budget	
Pre-CMOS	Best	Limited	Yes	No	Sandia National Lab
Intermediate-CMOS	Good	Very limited	Yes	Yes	Analog Devices, Inc.
Post-CMOS	Varies	Good	No	Yes	Berkeley CMU UF ETH

## Sandia National Laboratories iMEMS



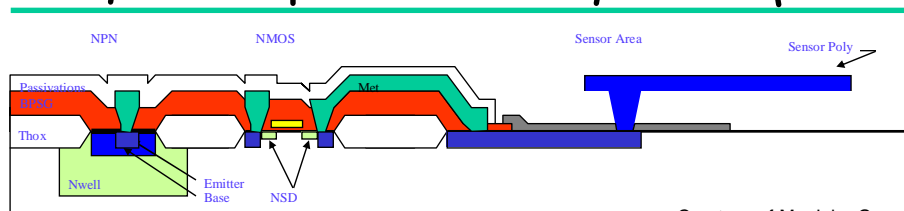
- Pre-etched trench to house MEMS structures
- CMP to planarize the wafer for regular CMOS processing
- Wet etch to release MEMS structures
- Need a dedicated production line

## Analog Devices, Inc. BiMEMS



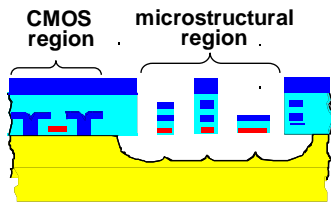
ADXL201 2g, 3V, Dual-Axis Motion Sensor

- Form transistors on bare wafers first
- Then deposit and anneal MEMS structural materials
- No CMP needed
- Only one interconnect metal layer
- Wet etch to release MEMS structures
- **Need a dedicated production line**

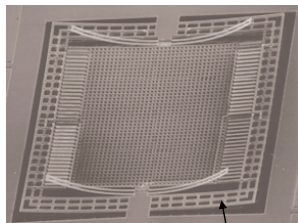


Courtesy of Mr. John Geen  
of Analog Devices, Inc.

## Post CMOS-MEMS Process (thin-film)



G. Fedder *et al.*, Sensors & Actuators  
A, v.57, no.2, 1996



Curl matching frame

- ✓ No lithography needed
  - ✓ Integrated CMOS circuitry
  - ✓ Low parasitic capacitance to substrate
  - ✓ High wiring flexibility
  - ✓ Curling can be matched
- 
- Curling is still an issue
    - Size limitation
    - Temperature performance
  - No bottom electrode for vertical capacitive sensing

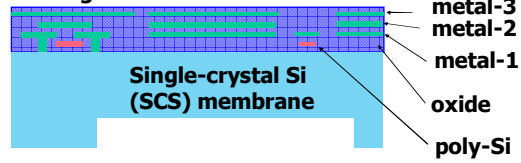


# Post CMOS-MEMS Process (DRIE)

## (a) Backside etch

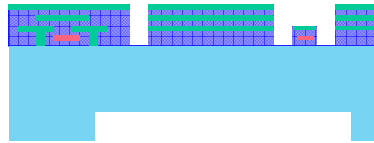
**STS:** 12-sec etching  
 130-sccm SF<sub>6</sub>, 13-sccm O<sub>2</sub>,  
 23 mT, 600 W coil power,  
 12 W platen power;  
 8-sec passivation  
 85-sccm C<sub>4</sub>F<sub>8</sub>, 12 mT, 600 W  
 coil power, 0 platen power.

CMOS-region

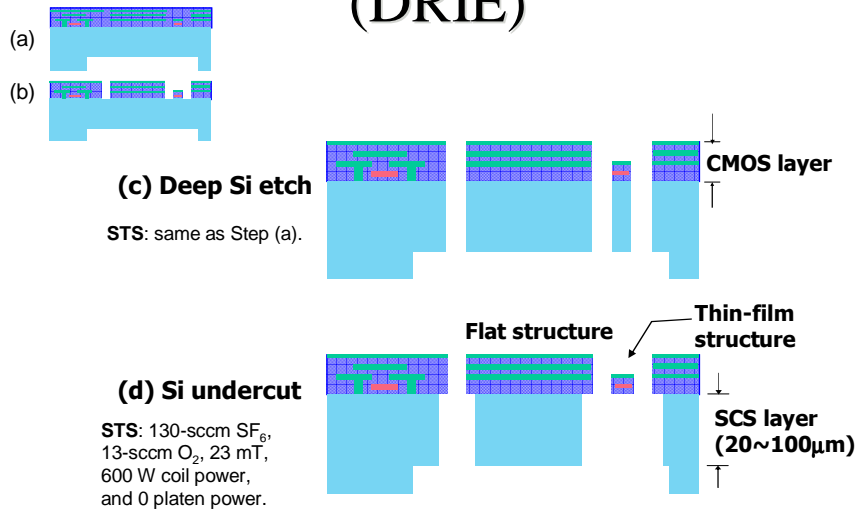


## (b) Oxide etch

**PlasmaTherm-790:**  
 22.5-sccm CHF<sub>3</sub>, 16-sccm  
 O<sub>2</sub>, 100 W, 125 mT for 125  
 minutes and then 100 mT for  
 10 minutes.



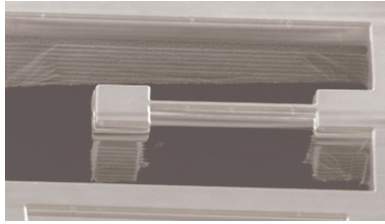
# Post CMOS-MEMS Process (DRIE)



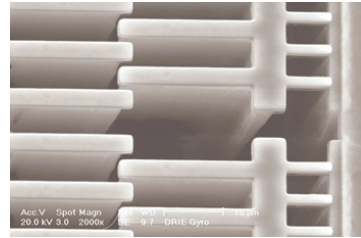
H. Xie et al, J. MEMS, Vol.11, no.2, 2002

## Electrical Isolation of Silicon

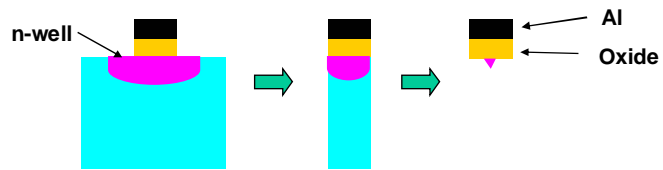
- Electrically isolated silicon island



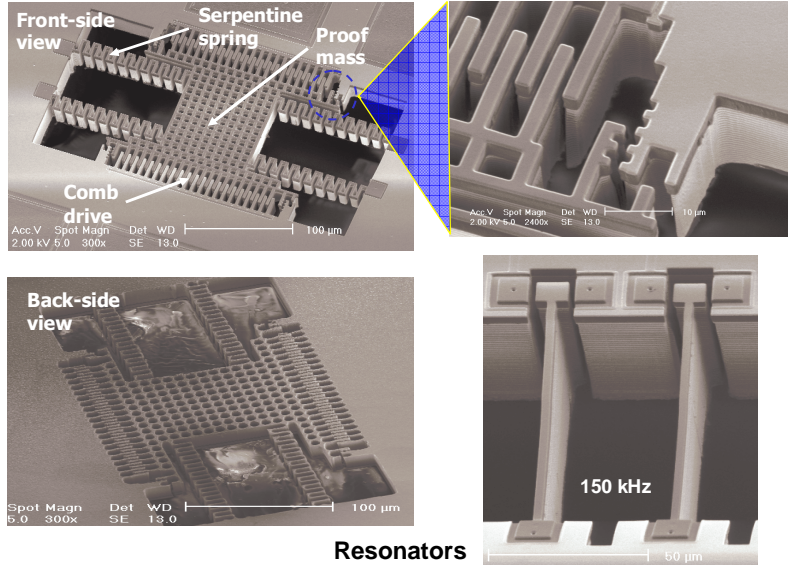
- Electrically isolated comb fingers



- Using n-well to improve undercut yield



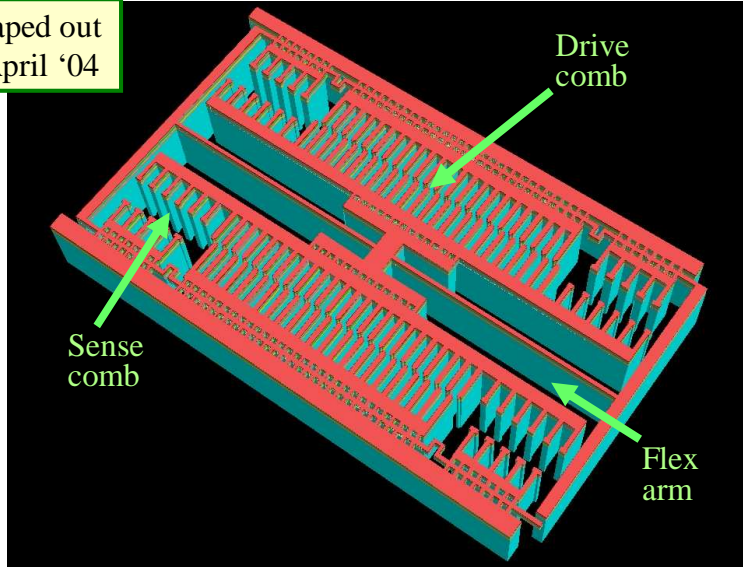
# DRIE CMOS-MEMS Resonators



UF CONFIDENTIAL – PATENT PENDING

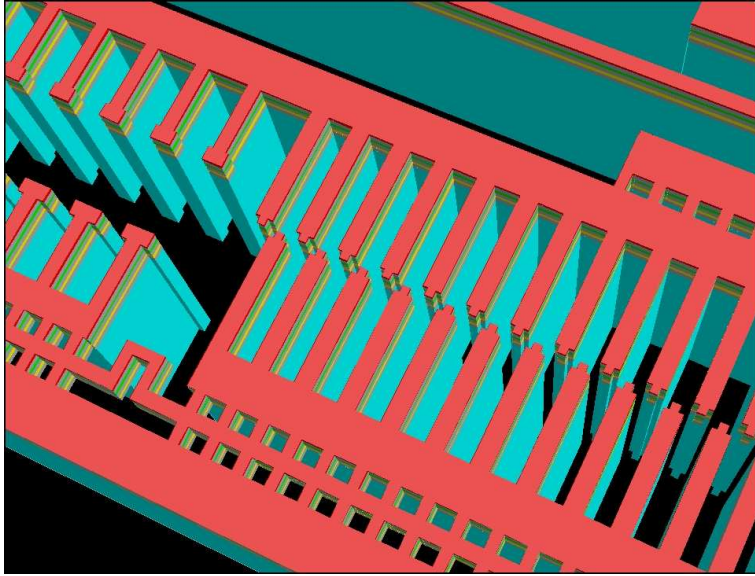
## Post-TSMC35 AdiaMEMS Resonator

Taped out  
April '04



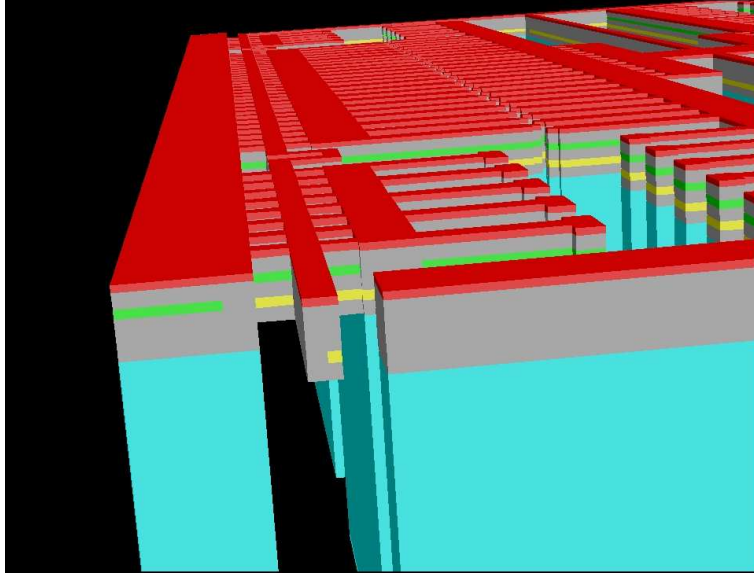
**UF CONFIDENTIAL – PATENT PENDING**

## Close-Up View, Drive/Sense Combs



**UF CONFIDENTIAL – PATENT PENDING**

**Side View, Showing Si Undercut**



## Long-Term Projections

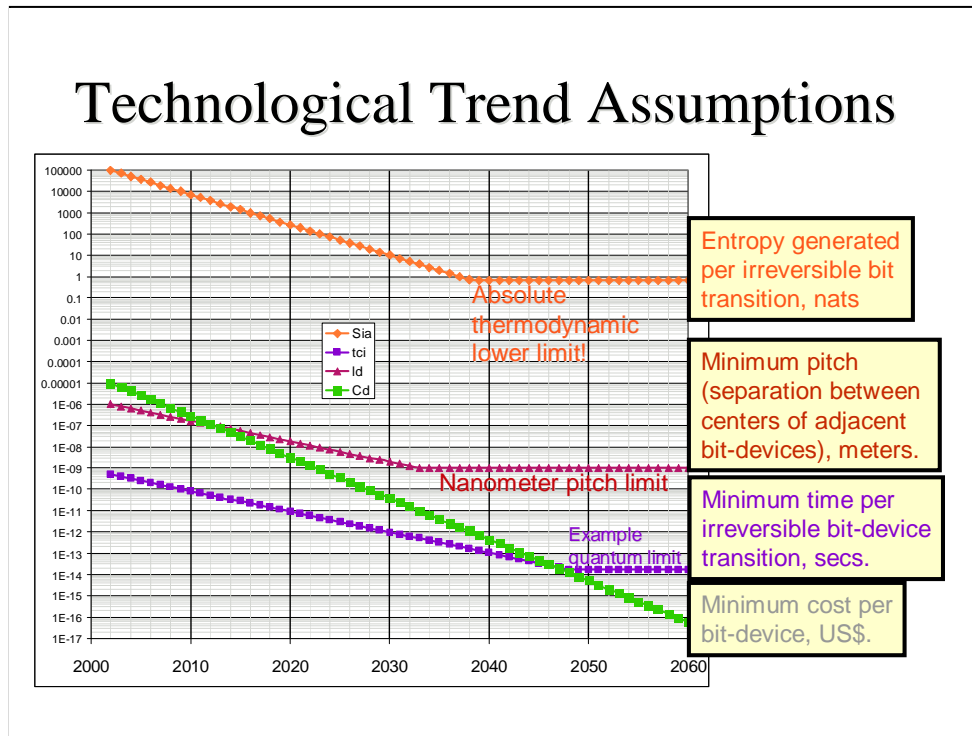
For future computational cost-  
efficiency improvements potentially  
available via reversible computing



## The Future of Reversible Computing

- What if we model how the hardware algorithm overheads for reversible computing scale?
  - Worst case: Increases with roughly  $Q^{1.6}$
- Can reversible computing become practical for general-purpose, high-performance computing?
  - And not just for 'niche' ultra-low-power apps?
- What happens if present technological trends continue until fundamental limits are reached?
  - And, what happens after that?
- We performed an analysis that addresses these questions...

# Technological Trend Assumptions



Now, in order to actually make any kind of statement about the timing of the emergence of the usefulness of reversibility, we had to make some assumptions about how various raw parameters of the device technology would change as a function of time over future decades. Although obviously it is difficult to forecast these developments exactly, there are some strong, steady historical trends, as well as some clear limits to these trends, that together allow us to sketch out a technology scaling model with some confidence in its approximate correctness.

The upper red line shows the entropy generated per irreversible bit erasure, in units of Boltzmann's constant  $k$ . Today in 0.1-micron CMOS technology, this is about 100 thousand. Calculations based on the International Technology Roadmap for Semiconductors show that the industry wants this to decline by 28% per year in the future (historically it has decreased somewhat faster). At this rate, it would reach the absolute thermodynamic minimum of about  $0.7k$  by about the year 2038.

Next, the mahogany line shows average device pitch, or separation between the centers of neighboring devices. This is about 1 micron today if you include space for interconnects. The standard Moore's Law trend is for pitch to decrease by a factor of 2 every 3 years (so that density doubles every 18 months). We assumed that 1 nm (just ~3-4x larger than atomic diameters) is an absolute minimum. This will be reached by about 2033.

The purple line shows clock period, which is about half a nanosecond today and decreases at about the same rate as pitch. The quantum maximum frequency is about half a PetaHertz per electron volt, giving a minimum period of about 2 femtoseconds if we assume no more than 1 eV of energy per bit. The maximum voltages achievable across nanometer-pitch or smaller structures are on the order of a volt, because molecular structure breaks down at much higher voltages than this. (Molecular ionization energies are on the order of a few eV.)

Finally, the green line shows cost per bit-device. The cost per device is on the order of a thousandth of cent today. For example, an Intel Itanium 2 microprocessor with 220 million transistors probably costs on the order of 2200 dollars or less. Moore's Law has cost-per-device decreasing by about half every 18 months. We assume this trend can continue indefinitely, due to improvements in 3D nanomanufacturing technology (self-assembly, nanofabrication, etc.), even after the pitch limit is reached. We should note that even if cost per device does not continue decreasing after devices reach a minimum size, our results will still end up favoring reversible computing.

## Fixed Technology Assumptions

- Total cost of manufacture: US\$1,000.00
  - User will pay this for a high-performance desktop CPU.
- Expected lifetime of hardware: 3 years
  - After which obsolescence sets in due to price drops.
- Total power limit: 100 Watts
  - Practical limit for a laptop much quieter than a hair-dryer!
- Power flux limit: 100 Watts per square centimeter
  - Approximate limit of conduction/air-cooling capabilities
- Standby entropy generation rate: 1,000 nat/s/device
  - Arbitrarily chosen, but achievable *e.g.* by today's DRAMs.

In addition to these assumptions about changing technology parameters, we made the following assumptions about parameters which are held constant for one reason or another.

We held total manufacturing cost constant at \$1,000, on the assumption that individuals will always be willing to pay about this much for a desktop computer or laptop. This figure has not been adjusted for inflation.

We hold the expected lifetime of the hardware to be about 3 years, since in this time the original machine would have lost most of its original value anyway (specifically,  $\frac{3}{4}$  of it), due to the assumed cost trend.

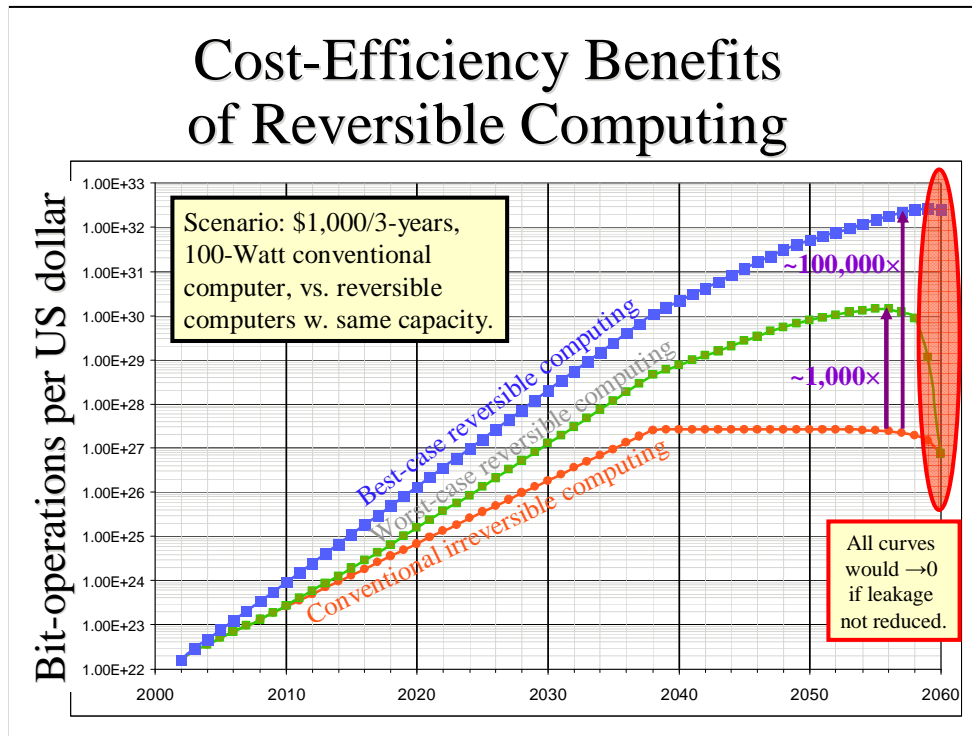
We set a total power limit of 100 Watts, to model the case of a machine that is held in the user's lap and thus cannot get rid of much more waste heat than this without the user experiencing some discomfort (or being annoyed by a noisy fan, think of a 1kW hairdryer).

We set a heat-flux density limit of 100 Watts per cubic centimeter, since this is roughly the most that can be achieved using ordinary air-cooling capabilities. (Actually probably the practical air-cooling limit is even less than this.)

Finally, we model a standby entropy generation rate of 1,000 nats/s/device. This fits the time constant for decay of a DRAM circuit node today which is about 1 millisecond. If a storage node were set at a low voltage level holding just a few nats of physical information, this would then yield the given rate. However, keeping this low of a rate as devices shrink to smaller sizes is a major challenge for nano-device technology. But we know it is possible, since for example Drexler's original mechanical rod-logic interlocks have essentially zero rate of standby entropy generation at room temperature, due to the high energy barriers presented by the steric intermolecular interactions between rigidly-bonded carbon-based structures. However, whether we can truly maintain this low rate in an all-electric or electromechanical technology at nanometer length scales is somewhat of an open research question. This may be the most unrealistic assumption in our current model. I would like to invite other researchers to help me develop a more refined scaling model for this parameter, to see how it would affect the results.

However, I should point out that if the desired low leakage cannot be maintained at say a 1 nm length scale then the answer is obvious: don't go down to this scale. Scaling up the device exponentially reduces tunneling losses but only polynomially increases size, cost, and energy. Therefore there will be an advantage to not going to the scale where leakage is totally dominant.

# Cost-Efficiency Benefits of Reversible Computing



Next I wrote a simple numerical optimization program in C that optimizes this model in each year's technology based on the scaling assumptions. This chart shows number of bit-operations that each technology can perform per US dollar, taking into account both time-amortized manufacturing cost and energy cost at present electric utility rates.

In the long run, energy concerns turn out to dominate the situation, but mostly through their affect on performance due to the cooling constraints, rather than because of the raw cost of energy itself. This reflects the fact that the total cost of the energy used by a 100-Watt computer operating continuously over its 3-year life is currently less than the cost of the computer itself.

The upper, blue line is the cost-efficiency of reversible computers on idealized problems for which the algorithmic overheads of reversibility are nil. The middle, green line is a more conservative model that assumes we find no better reversible algorithms for performing arbitrary computations than one that was discovered in 1989 by Bennett. Finally, the lower, red line shows the best that conventional irreversible computing can offer. Notice that its cost-efficiency hits the thermodynamic brick wall imposed by Landauer's principle by the year 2038, and cannot improve further. In contrast, reversible computing keeps improving. It starts to outperform irreversible computing between now and 2020, and becomes 1,000-100,000 x more cost-efficient by the 2050's.

After 2060, the cost-efficiencies of all technologies drop to 0 in this scenario because devices are so cheap that in order to spend as much as \$1,000 on your computer (as the scenario requires), it has to contain so many devices that it dissipates more than 100 Watts due to leakage when it is powered up, even when it is sitting passively doing nothing at all! Obviously, in practice, the curves would not actually dip – either leakage rates would be further reduced, continuing the upward trend, or the pressure to further reduce device manufacturing cost would halt (due to the dominance of energy cost), and so cost-efficiency would stabilize at the peak level shown.

## Conclusions

- For traditional “irreversible” technology, raw computer performance per unit power dissipated will reach fundamental limits soon.
  - Before many of us will retire.
- These limits are unassailable consequences of the most fundamental facts of known physics.
  - Quantum theory, 2<sup>nd</sup> law of thermodynamics.
- The *only* way to circumvent these limits is with some form of reversible computing technology.
  - At UF we are constructing MEMS-based prototypes that may be practical for ultra-low-power apps within 5 years.
- In the long run, *all* high-performance computing (subject to reasonable power constraints) will require this technology.
  - It is imperative to focus increased effort and attention on the optimization of reversible modes of device operation!