Common Mistakes in Adiabatic Logic Design and How to Avoid Them

Michael P. Frank
University of Florida
College of Engineering
Departments of CISE and ECE
`mpf@cise.ufl.edu`

Methodologies in Low Power Design Workshop
Int'l Conf. on Embedded Systems and Applications
Int'l Multiconf. In Computer Sci. & Computer Eng.
Las Vegas, Nevada, June 23-26, 2003

Abstract

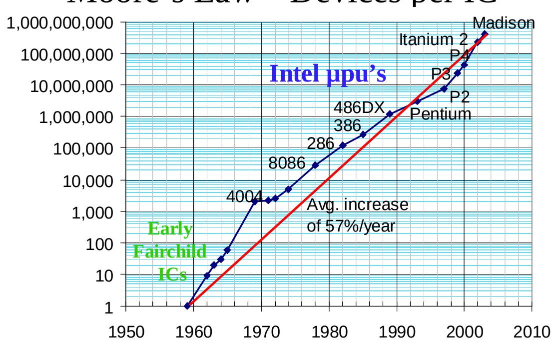
- *Watch out!* Most "adiabatic" logic families are not what I call *truly* adiabatic.
  - Many don't satisfy the general definition of an adiabatic process in physics.
  - Many "adiabatic" logic families aren't even *asymptotically* adiabatic!
  - I give my definition of "true adiabaticity."
- Yet, true adiabatic design will be *required* for most 21st-century computing!
  - At the nanoscale, energy dissipation is by far the dominant limiting factor on computing system performance, esp. for tightly-coupled parallel computations.
  - Truly-adiabatic design is the *only* way to work around the fundamental thermodynamic limits on computing which are rapidly being approached.
- Some of the most common adiabatic design mistakes, and their solutions:
  - Use of fundamentally non-adiabatic components, such as diodes.
  - Turning off transistors while there is nonzero current through them!
  - Overly-constrained design style that imposes a limited degree of logical reversibility and/or asymptotic efficiency.
- Overview of some recent advances in adiabatic circuits at UF:
  - 2LAL (a simple 2-level adiabatic logic)
  - GCAL (General CMOS Adiabatic logic)
  - High-Q MEMS/NEMS based resonant power supplies
  - Analysis of cost-efficiency benefits of adiabatics, & FET energy-dissipation limits

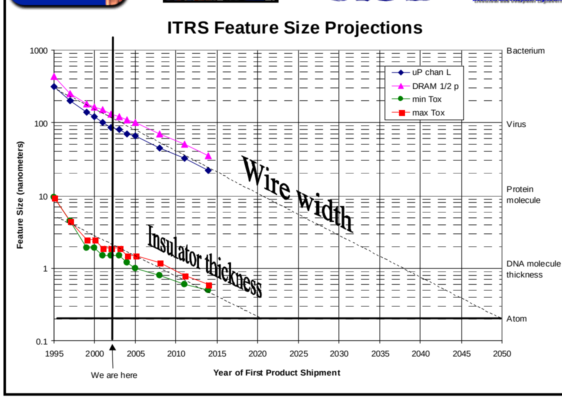
Organization of Talk

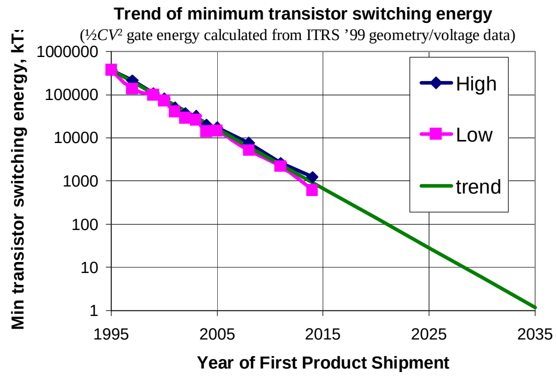
1. Why adiabatic design?
   - Moore's Law vs. Fundamental Limits of Computing
2. What does "adiabatic" mean, anyway?
   - Original, literal meaning vs. modern meaning
3. Adiabatic Circuits & Reversible Computing
   - Dispelling the Misconceptions
4. Common Mistakes to Avoid in Adiabatics
   - Overview of adiabatic design rules
5. Example adiabatic circuit styles:
   - SCRL, 2LAL
6. Other recent advances:
   - NEMS resonators, FET entropy-generation limits
7. Conclusions

Moore's Law vs. the Fundamental Physical Limits of Computing

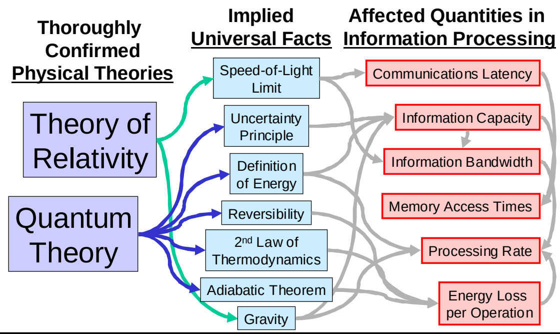
Moore's Law – Devices per IC







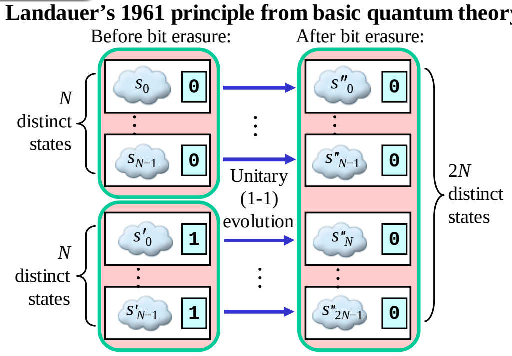
Fundamental Physical Limits of Computing



What is entropy?

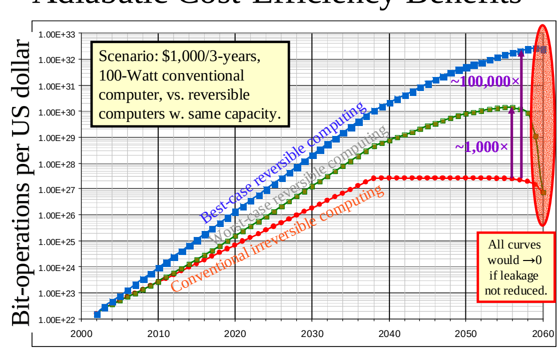
- First was characterized by Rudolph Clausius in 1850.
  - Originally was just defined as *heat ÷ temperature.*
  - Noted to never decrease in thermodynamic processes.
  - Significance and physical meaning were mysterious.
- In ~1880's, Ludwig Boltzmann proposed that entropy is just the logarithm of the number of states, $S = k \ln N$
  - What we would now call the information capacity of a system
  - Holds for systems at equilibrium, in maximum-entropy state
- The modern consensus resulting from 20th-century physics is that entropy is simply the amount of *unknown* or *incompressible* information in a physical system.
  - Contributions by von Neumann, Shannon, Jaynes, Zurek

Landauer's 1961 principle from basic quantum theory



Increase in entropy: $S = \log 2 = k \ln 2$. Energy lost to heat: $ST = kT \ln 2$

Adiabatic Cost-Efficiency Benefits



What is "adiabatic?"

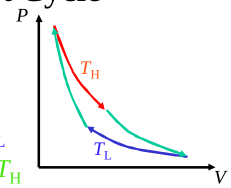
Evolution of the term

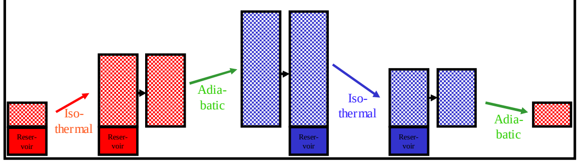
The Carnot Cycle

- In 1822-24, Sadi Carnot analyzed the efficiency of an ideal heat engine all of whose steps were *reversible*, and furthermore proved that:
  - *Any* reversible engine (regardless of details) would have the *same* efficiency $(T_H - T_L)/T_H$.
  - *No* engine could have greater efficiency than a reversible engine w/o producing work from nothing
  - *Temperature* itself could be defined on a thermodynamic scale based on heat recoverable by a reversible engine operating between T_H and T_L

Steps of Carnot Cycle

- *Isothermal* expansion at T_H
- *Adiabatic* (without flow of heat) expansion $T_H \to T_L$
- Isothermal compression at T_L
- Adiabatic compression $T_L \to T_H$



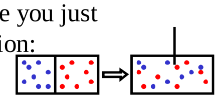


Carnot Cycle Terminology

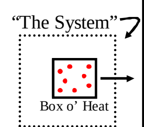
- *Adiabatic* (Latin): literally "Without flow of heat"
  - *I.e.*, no entropy *enters or leaves* the system
- *Isothermal*: "At the same temperature"
  - Temperature of system *remains constant* as entropy enters or leaves.
- Both kinds of steps, in the case of the Carnot cycle, are examples of *isentropic* processes
  - "at the same entropy"
  - *I.e.*, no (known) information is transformed into entropy in either process
- But, the usage of the word "adiabatic" in applied physics has mutated to essentially *mean* isentropic.

Old and New "Adiabatic"

- Consider a closed system where you just lose track of its detailed evolution:
  - It's *adiabatic* (no *net* heat flow),
  - But it's not "adiabatic" (not isentropic)



- Consider a box containing some heat, flying ballistically out of the system:
  - It's not *adiabatic*, (no heat flow)
    - because heat is "flowing" out of the system
  - But it's "adiabatic" (no entropy is generated)



Justifying the Modern Usage

- In an adiabatic process following a desired trajectory through configuration space,
  - No heat flows in or out of *the subsystem consisting of those particular degrees of freedom* whose variation carries out the motion along the desired trajectory.
    - *E.g.*, the computational degrees of freedom in a computational process.
  - No heat flow → no entropy flow
    - Heat is just energy whose configuration info. is entropy
  - No entropy flow → no sustained entropy generation
    - Since bounded systems have a maximum entropy

Quasi-Adiabatic

- *Complete* adiabaticity means absolutely *zero* rate of entropy generation
  - Requires infinite degree of isolation of system from uncontrolled external environment!
  - ∴ Impossible to *completely* achieve in practice.
- Real processes are only adiabatic to the extent that their entropy generation *approaches* zero.
  - Term "quasi-adiabatic" emphasizes imperfection
- *Asymptotically* adiabatic designs conceptually approach 0 in the limit of variation of specified technology design parameter(s)
  - *E.g.*, low device frequency, large device size

Quantifying Adiabaticity

- An appropriate metric for quantifying the *degree of adiabaticity* of any process is just to use the *quality factor* Q of that process.
  - Q isn't just for oscillatory processes any more
- Q is generally the ratio E_{trans} / E_{diss} between the:
  - Energy E_{trans} involved in carrying out a process (transitioning between states along a trajectory)
  - Amount E_{diss} of energy dissipated during the process.
- Normally also matches the following ratios:
  - Physical information content / entropy generated
  - Quantum computation rate / decoherence rate
  - Decoherence time / quantum-transition time

Some Loss-Inducing Interactions

For ordinary voltage-coded electronics:

- Interactions whose dissipation scales with speed:
  - Parasitic EM emission from reactive (C,L) elements
  - Scattering of ballistic electrons from lattice imperfections, causing Ohmic resistance
- Other interactions:
  - Interference from outside EM sources
  - Thermally-actived leakage of electrons over potential energy barriers
  - Quantum tunneling of electrons through narrow barriers (sub-Fermi wavelength)
  - Losses due to intentional commitment of physical information to entropy (bit erasure)

Focus of much work on adiabatics to date

Some Ways to Reduce Losses

- EM interference / emission: Add shielding, use high-Q MEMS/NEMS oscillators
- Scattering: Ballistic FETs, superconductors
- Thermal leakage: high-V_T and/or low temps
- Tunneling: thick barriers, high-κ dielectrics
- Intentional bit erasure: reduce voltages, use mostly-reversible logic designs

Adiabatic Circuits and Reversible Computing

Commonly Encountered Myths, Fallacies, and Pitfalls

(in the Hennessy-Patterson tradition)

Some Claims Against Reversible Computing	Eventual Resolution of Claim
John von Neumann, 1949 – Offhandedly remarks during a lecture that computing requires $kT \ln 2$ dissipation per "elementary act of decision" (bit-operation).	No proof provided. Twelve years later, Rolf Landauer of IBM tries valiantly to prove it, but succeeds only for logically irreversible operations.
Rolf Landauer, 1961 – Proposes that the logically irreversible operations which necessarily cause dissipation are unavoidable.	Landauer's argument for unavoidability of logically irreversible operations was conclusively refuted by Bennett's 1973 paper.
Bennett's 1973 construction is criticized for using too much memory.	Bennett devises a more space-efficient version of the algorithm in 1989.
Bennett's models criticized by various parties for depending on random Brownian motion, and not making steady forward progress.	Fredkin and Toffoli at MIT, 1980, provide ballistic "billiard ball" model of reversible computing that makes steady progress.
Various parties note that Fredkin's original classical-mechanical billiard-ball model is chaotically unstable.	Zurek, 1984, shows that quantum models can avoid the chaotic instabilities. (Though there are workable classical ways to fix the problem also.)
Various parties propose that classical reversible logic principles won't work at the nanoscale, for unspecified or vaguely-stated reasons.	Drexler, 1980's, designs various mechanical nanoscale reversible logics and carefully analyzes their energy dissipation.
Carver Mead, CalTech, 1980 – Attempts to show that the kT bound is unavoidable in electronic devices, via a collection of counter-examples.	No general proof provided. Later he asked Feynman about the issue; in 1985 Feynman provided a quantum-mechanical model of reversible computing.
Various parties point out that Feynman's model only supports serial computation.	Margolus at MIT, 1990, demonstrates a parallel quantum model of reversible computing—but only with 1 dimension of parallelism.
People question whether the various theoretical models can be validated with a working electronic implementation.	Seitz and colleagues at CalTech, 1985, demonstrate working energy recovery circuits using adiabatic switching principles.
Seitz, 1985—Has some working circuits, unsure if arbitrary logic is possible.	Koller & Athas, Hall, and Merkle (1992) separately devise general reversible combinational logics.
Koller & Athas, 1992 – Conjecture reversible *sequential* feedback logic impossible.	Younis & Knight @MIT do reversible sequential, pipelineable circuits in 1993-94.
Some computer architects wonder whether the constraint of reversible logic leads to unreasonable design convolutions.	Vieri, Frank and coworkers at MIT, 1995-99, refute these qualms by demonstrating straightforward designs for fully-reversible, scalable gate arrays, microprocessors, and instruction sets.
Some computer science theorists suggest that the algorithmic overheads of reversible computing might outweigh their practical benefits.	Frank, 1997-2003, publishes a variety of rigorous theoretical analysis refuting these claims for the most general classes of applications.
Various parties point out that high-quality power supplies for adiabatic circuits seem difficult to build electronically.	Frank, 2000, suggests microscale/nanoscale electro-mechanical resonators for high-quality energy recovery with desired waveform shape and frequency.
Frank, 2002—Briefly wonders if synchronization of parallel reversible computation in 3 dimensions (not covered by Margolus) might not be possible.	Later that year, Frank devises a simple mechanical model showing that parallel reversible systems can indeed be synchronized locally in 3 dimensions.

Myths about Adiabatic Circuits & Reversible Computing

- "Someone proved that computing with <<kT free-energy loss per bit-operation is impossible."
- "Physics isn't reversible."
- "An energy-efficient adiabatic clock/power supply is impossible to build."
- "True adiabaticity doesn't require reversible logic."
- "Sequential logic can't be done adiabatically."
- "Adiabatic circuits require many clock/power rails and/or voltage levels."
- "Adiabatic design is necessarily difficult."

Fallacies about Adiabatic Circuits and Reversible Computing

- "Since speed scales as energy dissipation in adiabatic circuits, they aren't good for high-performance computing."
- "If I can't invent an efficient adiabatic logic, it must be impossible."
- "The algorithmic overheads of reversible computing mean it can never be cost-effective."
- "Since leakage gets worse in nanoscale devices, adiabatics is doomed."

Pitfalls in Adiabatic Circuits and Reversible Computing

- Using diodes in the charge-return path
- Forgetting to obey one of the transistor rules
- Using traditional models of computational complexity
- Restricting oneself to an asymptotically inefficient design style
- Assuming that the best reversible and irreversible algorithms are similar
- Failing to optimize the degree of reversibility of a design
- Ignoring charge leakage in low-power/adiabatic design

Reversible vs. Quantum Computing

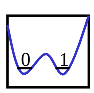
Property of Computing Mechanism	**Approximate Meaning**	**Required for Quantum Computing?**	**Required for Reversible Computing?**
(Treated As) Unitary	System's full invertible quantum evolution, w. all phase information, is modeled & tracked	Yes, device & system evolution must be modeled as ~unitary, within threshold	No, only reversible evolution of classical state variables need be tracked
Coherent	Pure quantum states don't decohere (for us) into statistical mixtures	Yes, must maintain full global coherence, locally within threshold	No, only maintain stability of local pointer states+transitions
Adiabatic	No entropy flow in/out of computational subsystem	Yes, must be above a certain threshold	Yes, as high as possible
Isentropic / Thermodynamically Reversible	No new entropy generated by mechanism	Yes, must be above a certain threshold	Yes, as high as possible
Time-Independent Hamiltonian, Self-Controlled	Closed system, evolves autonomously w/o external control	No, transitions can be externally timed & controlled	Yes, if we care about energy dissipation in the driving system
Ballistic	System evolves w. net forward momentum	No, transitions can be externally driven	Yes, if we care about performance

Adiabatic/Reversible Computing

Basic Models and Concepts

Bistable Potential-Energy Wells

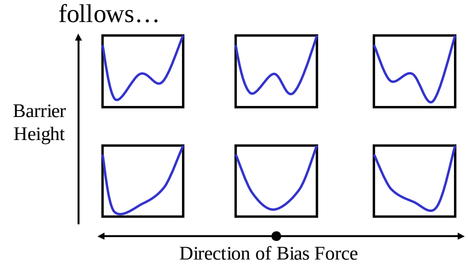
- Consider any system having an adjustable, bistable potential energy surface (PES) in its configuration space.
- The two stable states form a natural *bit*.
  - One state represents 0, the other 1.
- Consider now the P.E. well having two adjustable parameters:
  - (1) Height of the potential energy barrier relative to the well bottom
  - (2) Relative height of the left and right states in the well (bias)



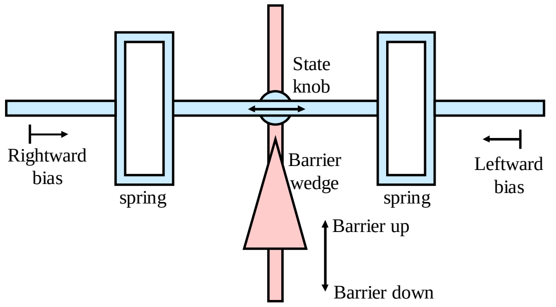
(Landauer '61)

Possible Parameter Settings

- We will distinguish six qualitatively different settings of the well parameters, as follows…

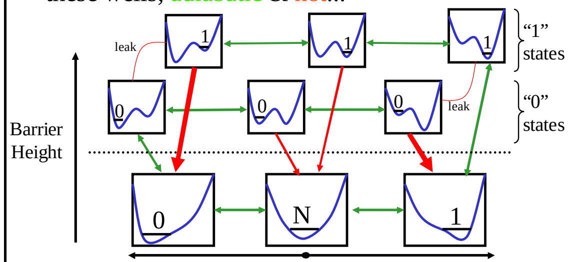


One Mechanical Implementation



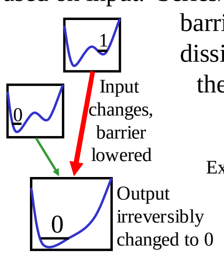
Possible Adiabatic Transitions

- Catalog of all the possible transitions in these wells, adiabatic & not... (Ignoring superposition states.)



Ordinary Irreversible Logics

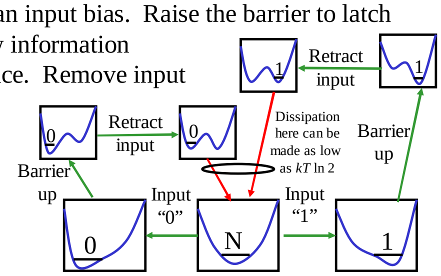
- Principle of operation: Lower a barrier, or not, based on input. Series/parallel combinations of barriers do logic. Major dissipation in at least one of the possible transitions.
  - Amplifies input signals.



Example: Ordinary CMOS logics

Ordinary Irreversible Memory

- Lower a barrier, dissipating stored information. Apply an input bias. Raise the barrier to latch the new information into place. Remove input bias.



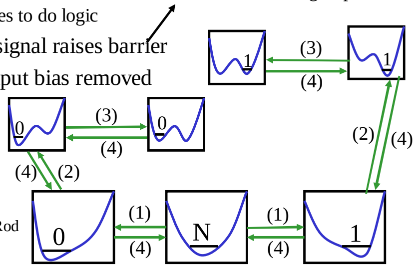
Example: DRAM

Input-Bias Clocked-Barrier Logic

- Cycle of operation:
  - (1) Data input applies bias
    - Add forces to do logic
  - (2) Clock signal raises barrier
  - (3) Data input bias removed

Can amplify/restore input signal in the barrier-raising step.

Can reset latch reversibly (4) given copy of contents.

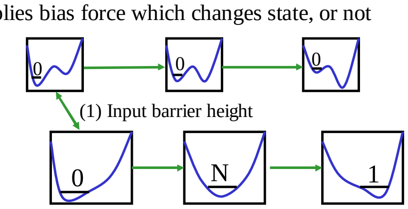


Examples: Adiabatic QDCA, SCRL latch, Rod logic latch, PQ logic, Buckled logic

Input-Barrier, Clocked-Bias Retractile

- Cycle of operation:
  - Inputs raise or lower barriers
    - Do logic w. series/parallel barriers
  - Clock applies bias force which changes state, or not

- Barrier signal amplified.
- Must reset output prior to input.
- Combinational logic only!

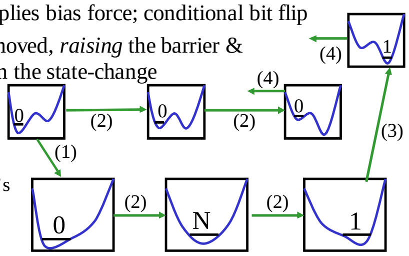


Examples:
Hall's logic,
SCRL gates,
Rod logic interlocks

(2) Clocked force applied →

Input-Barrier, Clocked-Bias Latching

- Cycle of operation:
  1. Input *conditionally lowers* barrier
     - Do logic w. series/parallel barriers
  2. Clock applies bias force; conditional bit flip
  3. Input removed, *raising* the barrier & locking in the state-change
  4. Clock bias can retract



Examples: Mike's 4-cycle adiabatic CMOS logic

Full Classical-Mechanical Model

The following components are sufficient for a complete, scalable, parallel, pipelinable, linear-time, stable, classical reversible computing system:

(a) Ballistically rotating flywheel driving linear motion.

(b) Scalable mesh to synchronize local flywheel phases in 3-D.

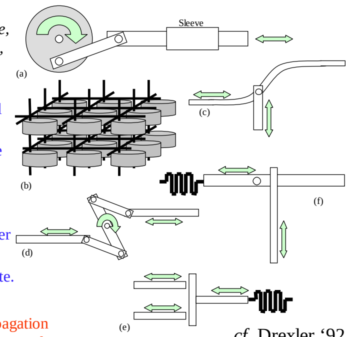
(c) Sinusoidal to flat-topped waveform shape converter.

(d) Non-amplifying signal inverter (NOT gate).

(e) Non-amplifying OR/AND gate.

(f) Signal amplifier/latch.

Primary drawback: Slow propagation speed of mechanical (phonon) signals.



cf. Drexler '92

Common Mistakes to Avoid

In Adiabatic Design

Common Mistakes to Avoid:

- Don't use diodes in charge-return path!
  - Built-in voltage drop kills adiabaticity
- Don't disobey adiabatic transistor rules by:
  - Turning on transistor with voltage across it
  - **Turning off transistor with current thru it!**
    - This one is often neglected
- Use mostly-reversible logic!
  - Optimize degree of reversibility for application
- Don't over-constrain the design family!
  - Asymptotically efficient circuits should be possible

Adiabatic Rules for Transistors

- **Rule 1:** Never turn *on* a transistor if it has a nonzero voltage across it!
  - *I.e.*, between its source & drain terminals.
  - **Why:** This erases info. & causes ½CV^2 disspation.
- **Rule 2:** Never apply a nonzero voltage across a transistor even *during* any on↔off transition!
  - **Why:** When partially turned on, the transistor has relatively low R, gets high $P=V^2/R$ dissipation.
  - **Corollary:** Never turn *off* a transistor if it has a nonzero current going through it!
    - **Why:** As R gradually increases, the $V=IR$ voltage drop will build, and then rule 2 will be violated.

Adiabatic Rules, continued…

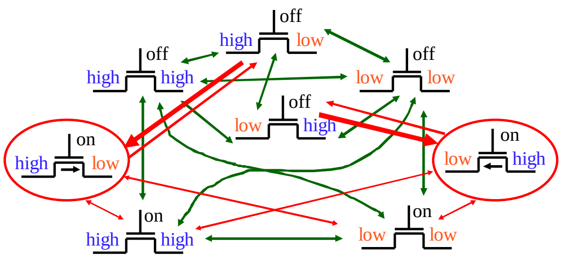
- **Transistor Rule 3:** Never suddenly change the voltage applied across any *on* transistor.
  - **Why:** So transition will be more reversible; dissipation will approach $CV^2(RC/t)$, not ½CV^2.

Adiabatic rules for other components:

- **Diodes**: Don't use them at all!
  - There is always a built-in voltage drop across them!
- **Resistors**: Avoid moderate network resistances, if poss.
  - *e.g.* stay away from range >10 kΩ and <1 MΩ
- **Capacitors**: Minimize, reliability permitting.
  - Note: Dissipation scales with C^2!

Transistor Rules Summarized

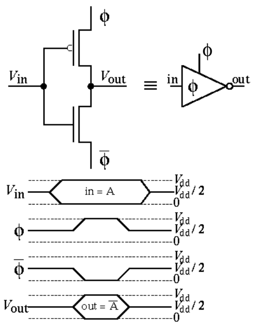
Legal adiabatic transitions in green. (For *n*- or *p*-FETs.)
Dissipative states and transitions in red.



SCRL: Split-level Charge Recovery Logic

The First Pipelined Fully-Adiabatic CMOS Logic
(Younis & Knight, MIT, ’94)

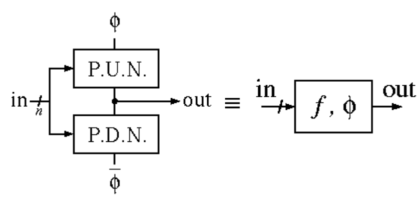
SCRL inverter



Transformation of local state:

Just before transition: in	out		*After transition:* in	out
0	½	⟶	0	1
1	½	⟶	1	0

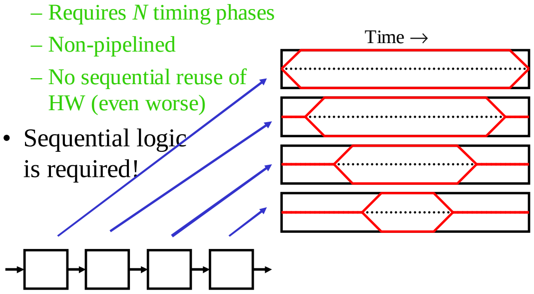
SCRL generalized inverter



- There may be several input signals.
- Uses complementary pull-up and pull-down networks like standard static CMOS.
- Schematic icon labeled with function computed and clock phase.

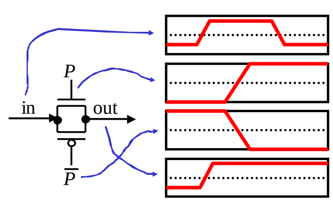
Retractile Logic w. SCRL gates

- Simple combinational logic of any depth *N*:
  - Requires *N* timing phases
  - Non-pipelined
  - No sequential reuse of HW (even worse)
- Sequential logic is required!



Simple Reversible CMOS Latch

- Uses a standard CMOS *transmission gate*
- Sequence of operation:
  - (1) input initially matches latch contents (output)
  - (2) input changes→output changes (3) latch closes
  - (4) input removed

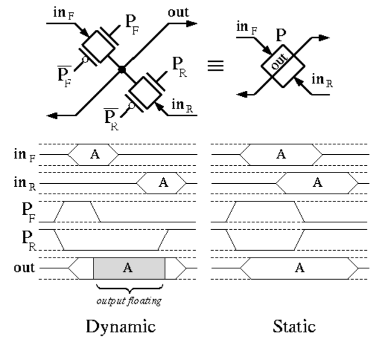


Before input: in	out	*Input arrived:* in	out	*Input removed:* in	out
a	a	a	a	a	a
		b	b	a	b

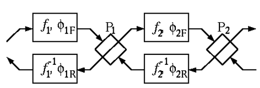
Resetting a Reversible Latch

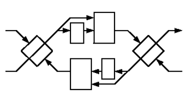
- Can reversibly *unlatch* data as follows: (exactly the reverse of the latching process)
  - (1) Data value *d* stored on memory node M.
  - (2) Present an exact copy of *d* on input.
  - (3) Open the latch (connecting input to M).
    - No dissipation since voltage levels match
  - (4) Retract the copy of *d* from the input.
    - Retracts copy stored in latch also.

SCRL bidirectional latch



SCRL pipeline

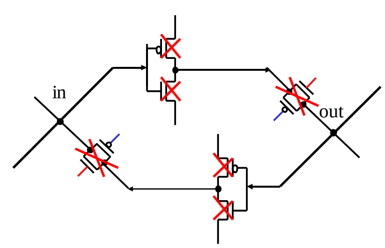




- Each stage may be many gates in parallel, considered as an *n*-input, *m*-output invertible Boolean logic function.
- Reverse stages uncompute the inputs to forward stages.
- Each stage has its own clock phase, until repeat.
- A stage may use 2 nested levels of gates to compute non-inverting functions.
- Sequential circuits with feedback fine, if phases aligned properly.

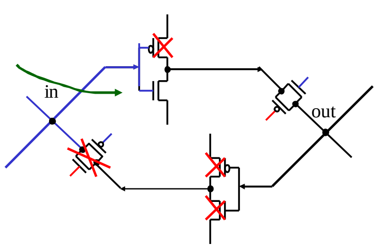
SCRL 6-tick clock cycle

Initial state: All gates off, all nodes neutral.



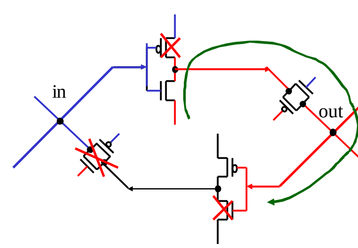
SCRL 6-tick clock cycle

Tick #1: Input goes valid, forward T-gate opens.



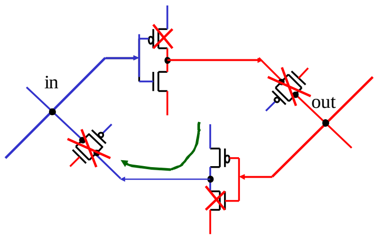
SCRL 6-tick clock cycle

Tick #2: Forward gate charges, output goes valid. (Tick #1 of subsequent gate.)



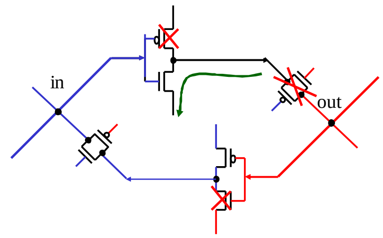
SCRL 6-tick clock cycle

Tick #3: Forward T-gate closes, reverse gate charges.



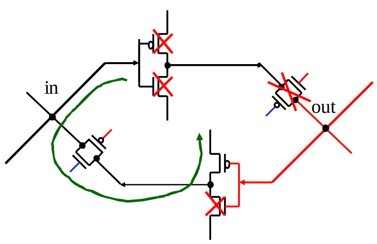
SCRL 6-tick clock cycle

Tick #4: Reverse T-gate opens, forward gate discharges.



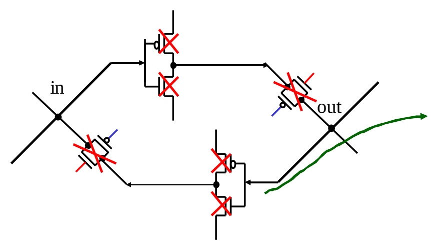
SCRL 6-tick clock cycle

Tick #5: Reverse gate discharges, input goes neutral.

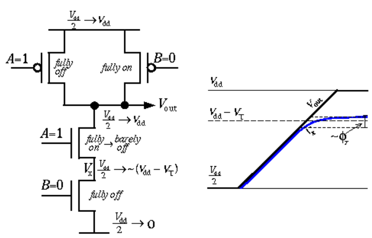


SCRL 6-tick clock cycle

Tick #6: Reverse T-gate closes, output goes neutral.
Ready for next input!

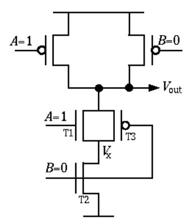


A problematic case for the original SCRL



- Inner n-FET asked to pass a high voltage.
- Barely turns off as inner node voltage approaches threshold.
- V_x crawls up by several ϕ_Ts over a substantial voltage drop.
- In numerical experiments, dissipation was still $\sim 3000\,k_B T$ in this case, when other dissipations were $< k_B T$.
- This dissipation not eliminated unless circuit so slow that leakage dominates.

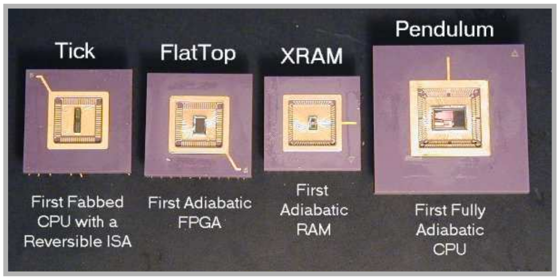
Fixing the problem case



- Add p-FET to pulldown network, as shown.
- Interior node now tracks all the way to V_{dd}.
- General solution: use dual-rail logic and transmission gates everywhere.

Reversible / Adiabatic Chips Designed @ MIT, 1996-1999

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



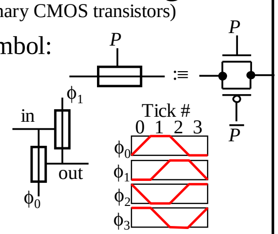
2LAL: 2-Level Adiabatic Logic

A Novel Alternative to SCRL

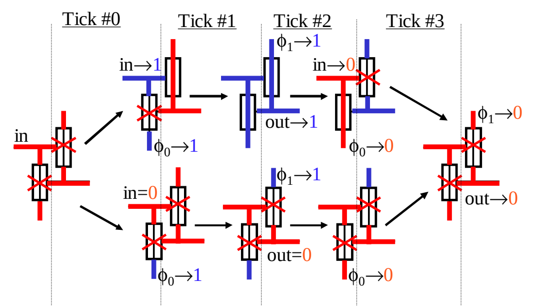
2LAL: 2-level Adiabatic Logic

(Implementable using ordinary CMOS transistors)

- Use simplified T-gate symbol:
- Basic buffer element:
  - cross-coupled T-gates
- Only 4 timing signals, 4 ticks per cycle:
  - ϕ_i rises during tick i
  - ϕ_i falls during tick $(i+2)$ mod 4

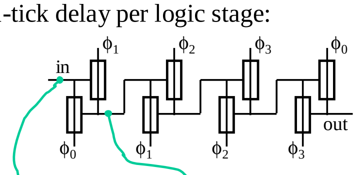


2LAL Cycle of Operation

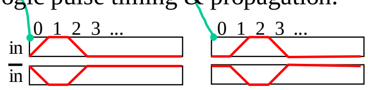


2LAL Shift Register Structure

- 1-tick delay per logic stage:

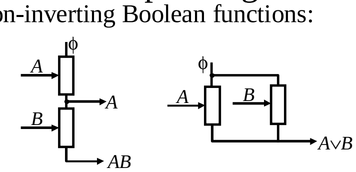


- Logic pulse timing & propagation:

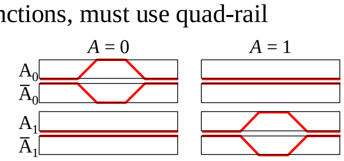


More complex logic functions

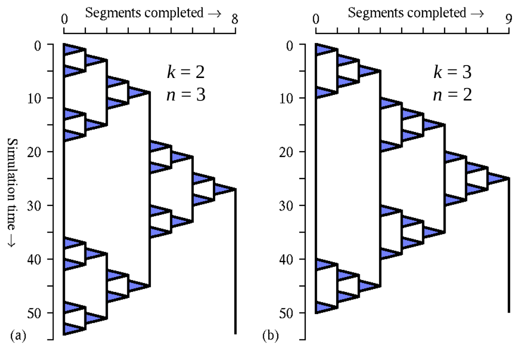
- Non-inverting Boolean functions:



- For inverting functions, must use quad-rail logic encoding:
  - To invert, just swap the rails!
    - Zero-transistor "inverters."



Reversible Emulation - Ben89



GCAL: General CMOS Adiabatic Logic

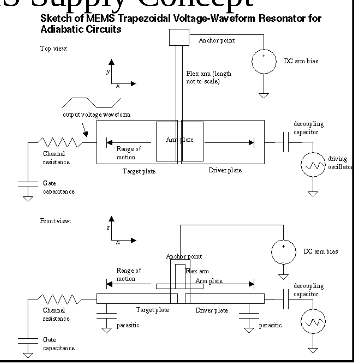
- A general CMOS adiabatic design methodology
- Currently under development at UF
- Notable features:
  - Permits designs attaining asymptotically optimal cost-efficiency
    - For any combination of time, space, spacetime, energy costs
  - Arbitrarily high degree of reversibility
  - Supports minimal 2-level and 3-level adiabatic gates
  - Requires only 4 externally supplied clock/power signals for 2-level logic
    - Or only 12 for 3-level logic
  - Supports mixture of fully-pipelined and retractile logic.
  - Supports quiescent dynamic/static latches & RAM cells
- Tools currently under development:
  - A new HDL specialized for describing adiabatic designs
  - Digital circuit simulator with adiabaticity checker
  - Adiabatic logic synthesis tool, with automatic legacy design converter

MEMS/NEMS Resonators

A Novel Clock/Power Supply Technology for Adiabatic Circuits

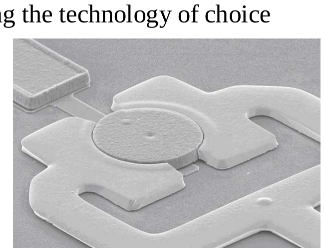
A MEMS Supply Concept

- Energy stored mechanically.
- Variable coupling strength → custom wave shape.
- Can reduce losses through balancing, filtering.



MEMS/NEMS Resonators

- State of the art technologies demonstrated in lab:
  - Frequencies up into the microwave (>1 GHz) regime
  - Q's >10,000 in vacuum, several thousand even in air!
- Are rapidly becoming the technology of choice for commercial RF filters, *etc.*, in embedded communications SoCs (Systems-on-a-Chip), *e.g.* for cellphones.

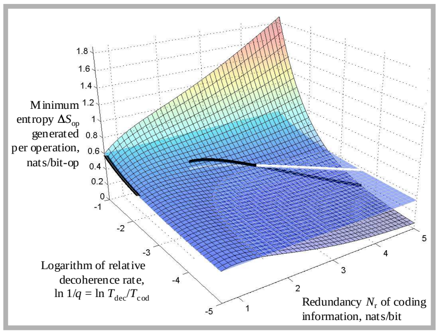


Minimizing Entropy Generation in Adiabatic FET Operations

Taking leakage-voltage tradeoff into account

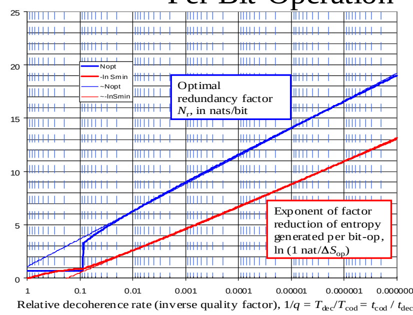
CISE

Minimizing Entropy Generation in Field-Effect Nano-devices



CISE

Lower Limit to Entropy Generation Per Bit-Operation



Scaling with device's quantum "quality" factor q.

- The optimal redundancy factor scales as: $1.1248(\ln q)$
- The minimum entropy generation scales as: $q^{-0.9039}$

CISE

Conclusions

- Logic designs having an ever-increasing degree of adiabaticity will become an *absolute requirement* for most high-performance computing over the course of the next few decades.
- To achieve this, diodes must be avoided, transistor rules must be followed, and an increasing degree of logical reversibility (with asymptotically efficient designs) will be required.
- Some examples of truly-adiabatic design styles were presented, and a general, efficient adiabatic CMOS design methodology is under development.