Exceptional service in the national interest

Adiabatic Circuits: A Tutorial Introduction

Michael Frank
Center for Computing Research
Sandia National Laboratories

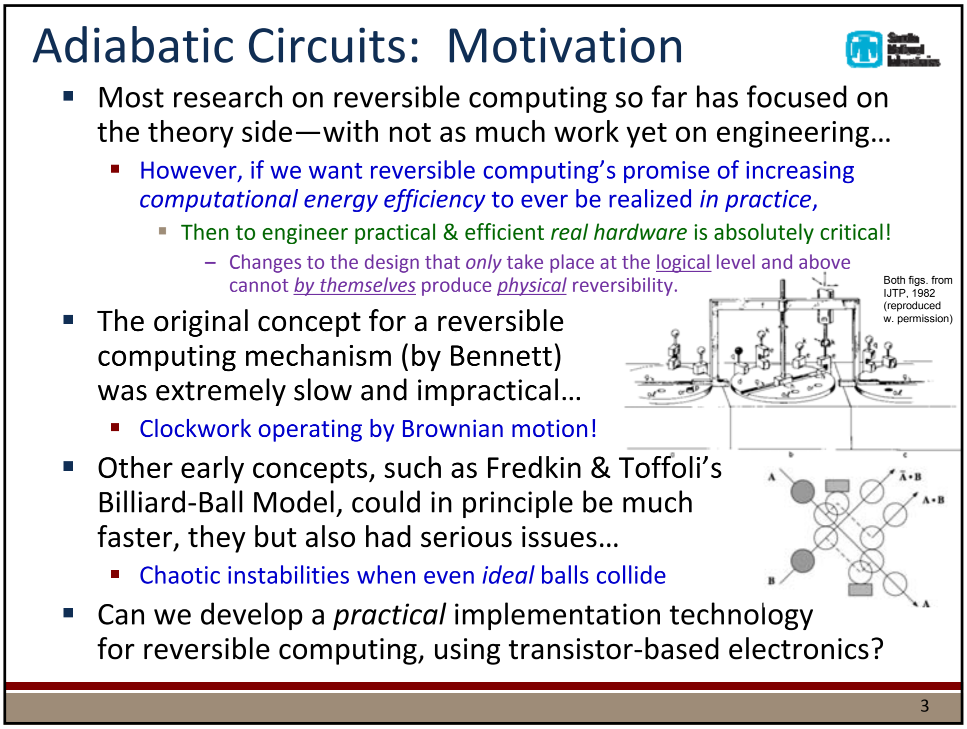
Summer School on Reversible and Quantum Computing
9th Conference on Reversible Computation
Kolkata, India, July 5th, 2017

Approved for Unclassified Unlimited Release
SAND2017-6909 C

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Adiabatic Circuits: Outline of Lecture

- Motivation
  - Adiabatic circuits as an approach to implement reversible computing
- Brief History of Adiabatic Circuits
- Fundamental Principles of Adiabatic Circuits
  - Adiabatic processes – General definitions
  - Adiabatic charge transfer – Energy dissipation analysis
  - Generalized Reversible Computing – Preview of conference talk
    - The correct theoretical foundation for adiabatic design
  - Adiabatic Switching Rules
- Elements of Adiabatic Logic Design
  - Adiabatic two-level and split-level gates
  - Dynamic and static latching
  - Retractile combinational circuits
  - Pipelined sequential circuits
- Power/Clock Generation
  - Strategies for resonator design
- Conclusion

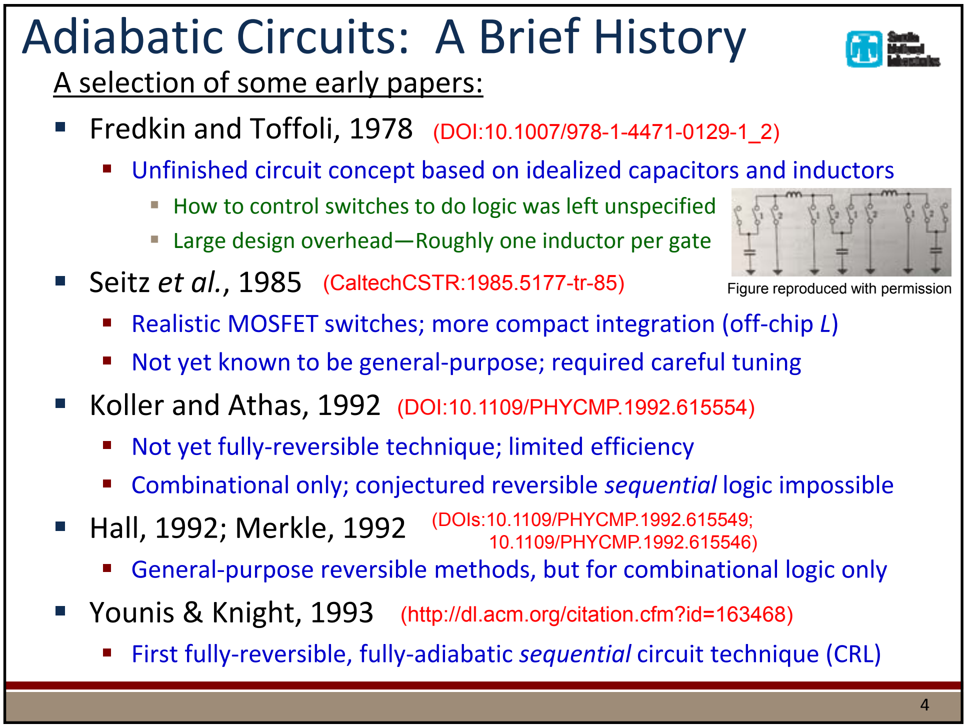


Adiabatic Circuits: Motivation

- Most research on reversible computing so far has focused on the theory side—with not as much work yet on engineering…
  - However, if we want reversible computing's promise of increasing *computational energy efficiency* to ever be realized *in practice*,
    - Then to engineer practical & efficient *real hardware* is absolutely critical!
      - Changes to the design that *only* take place at the logical level and above cannot *by themselves* produce *physical* reversibility.
- The original concept for a reversible computing mechanism (by Bennett) was extremely slow and impractical…
  - Clockwork operating by Brownian motion!
- Other early concepts, such as Fredkin & Toffoli's Billiard-Ball Model, could in principle be much faster, they but also had serious issues…
  - Chaotic instabilities when even *ideal* balls collide
- Can we develop a *practical* implementation technology for reversible computing, using transistor-based electronics?

Both figs. from IJTP, 1982 (reproduced w. permission)

3



Adiabatic Circuits: A Brief History

A selection of some early papers:

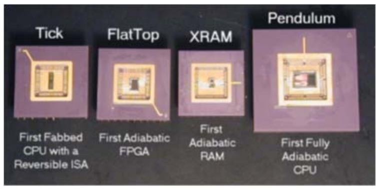
- Fredkin and Toffoli, 1978 (DOI:10.1007/978-1-4471-0129-1_2)
  - Unfinished circuit concept based on idealized capacitors and inductors
    - How to control switches to do logic was left unspecified
    - Large design overhead—Roughly one inductor per gate
- Seitz *et al.*, 1985 (CaltechCSTR:1985.5177-tr-85)
  - Realistic MOSFET switches; more compact integration (off-chip *L*)
  - Not yet known to be general-purpose; required careful tuning
- Koller and Athas, 1992 (DOI:10.1109/PHYCMP.1992.615554)
  - Not yet fully-reversible technique; limited efficiency
  - Combinational only; conjectured reversible *sequential* logic impossible
- Hall, 1992; Merkle, 1992 (DOIs:10.1109/PHYCMP.1992.615549; 10.1109/PHYCMP.1992.615546)
  - General-purpose reversible methods, but for combinational logic only
- Younis & Knight, 1993 (http://dl.acm.org/citation.cfm?id=163468)
  - First fully-reversible, fully-adiabatic *sequential* circuit technique (CRL)

Figure reproduced with permission

4

Adiabatic Circuits: History, cont.

- Younis & Knight, 1994 (Int'l Workshop on Low-Power Design)
  - Simplified 3-level adiabatic CMOS design family (SCRL)
    - However, the original version of SCRL contained a small non-adiabaticity bug which I discovered in 1997
      - The problem is easily fixed, however
- Subsequent work at MIT, 1995-99
  - Myself and fellow students
  - Various chips designed using SCRL →
  - Reversible processor architectures


Tick
FlatTop
XRAM
Pendulum
First Fabbed CPU with a Reversible ISA
First Adiabatic FPGA
First Adiabatic RAM
First Fully Adiabatic CPU

- Substantial literature throughout the late 90s / early 2000s…
  - Too many different papers / groups to list them all here!
- Researchers recently active in adiabatic circuits include:
  - A couple I know in the US:
    - Greg Snider (Notre Dame)
    - Himanshu Thapliyal (U. Kentucky)
  - Various groups in Europe, India, China, Japan (some here today, hello)…

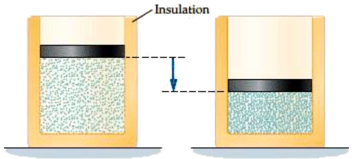
5

Adiabatic Processes: General Definitions

- The word "adiabatic" has a long (>135-year!) history in physics…
  - Derives originally from the Greek *adiabatos* (*άδιάβατος*), "impassable,"
    - In the sense "not to be passed through;" fr. *ά* (not) + *διά* (through) + *βατός* (passable)
  - In practice, in the context of thermodynamics, the word is used to mean:
    - "No [free] energy may pass through the boundary of the system so as to become dissipated out into the system's external environment as heat"
  - For our purposes, we can take it as effectively being a synonym for *isentropic*
    - Meaning, *with the same (unchanging) entropy*
      - Entropy increase = Part of the system's energy is crossing over an abstract boundary from a known/controlled to unknown/uncontrolled state
- For a process to be adiabatic generally requires that the active energy associated with the known/controlled degrees of freedom in the system is *well isolated* from the system's thermal environment, which implies:
  - The process does not happen so *quickly* that uncontrolled modes become excited
    - Rate of the process should be *slow* compared to the system's relaxation timescale
  - But also, it does not happen so *slowly* that the known/controlled energy in the system can leak out from the system to its environment via equilibration processes
    - Time for the process should be *fast* compared for the time for the non-equilibrium aspects of the system to equilibrate with the system's thermal environment
- We can design adiabatic mechanisms that (as they are further refined) approach the ideal of satisfaction of both of these requirements, by
  - Decreasing the relaxation timescale (increase generalized "stiffness" of mechanism)
  - Increasing the equilibration timescale (decrease the rate of energy "leakage")

6

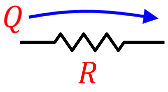
Adiabatic Processes: Example



- Adiabatic compression (or expansion) of an ideal gas under control of a piston in a thermally insulated cylinder…
  - The compression/expansion must be carried out *slowly* enough so as not to excite pressure waves in the gas…
    - Since the energy imparted to such waves would quickly degrade to heat
    - Need: Speed of piston movement << speed of sound in the gas
  - The compression/expansion must be done *quickly* enough so there isn't enough time for heat to be conducted into or out of the cylinder
    - Temperature inside the cylinder would typically be changing as the gas compresses/expands, and is not always the same as environment temp.
      - And, heat conducted from higher→lower temp. yields an entropy increase
    - Time for piston movement << time constant of thermal equilibration

Adiabatic Change Transfer:

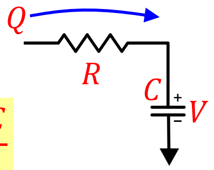
Energy Dissipation Analysis



- Consider passing a total quantity of charge Q through a resistive element of resistance R over a timespan t by means of a constant current, $I = Q/t$.
  - The power dissipation (rate of energy dissipation) in such a current flow is given by $P = IV$, where $V = IR$ (Ohm's Law) is the voltage drop across the resistor.
- The total energy dissipated over time t is therefore:

$$E_{\text{diss}} = P \cdot t = IVt = I^2Rt = (Q/t)^2Rt = Q^2R/t.$$

  - Note the inverse scaling with the time t taken for the charge transfer!
- If the function of the charge transfer is to charge a capacitance C up to the voltage level V, then the quantity of charge transferred is $Q = CV$, and so the energy dissipated can be expressed as:

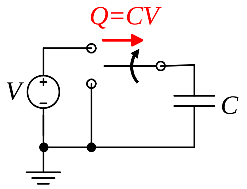


$$E_{\text{diss}} = (CV)^2R/t = C^2V^2R/t = CV^2\frac{RC}{t}$$

Conventional *vs.* Adiabatic Charging

For charging a capacitive load C through a voltage swing V

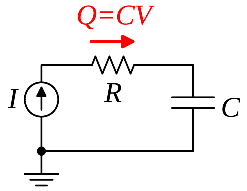
- Conventional charging:
  - Constant *voltage* source



  - Energy dissipated:

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

- Ideal adiabatic charging:
  - Constant *current* source

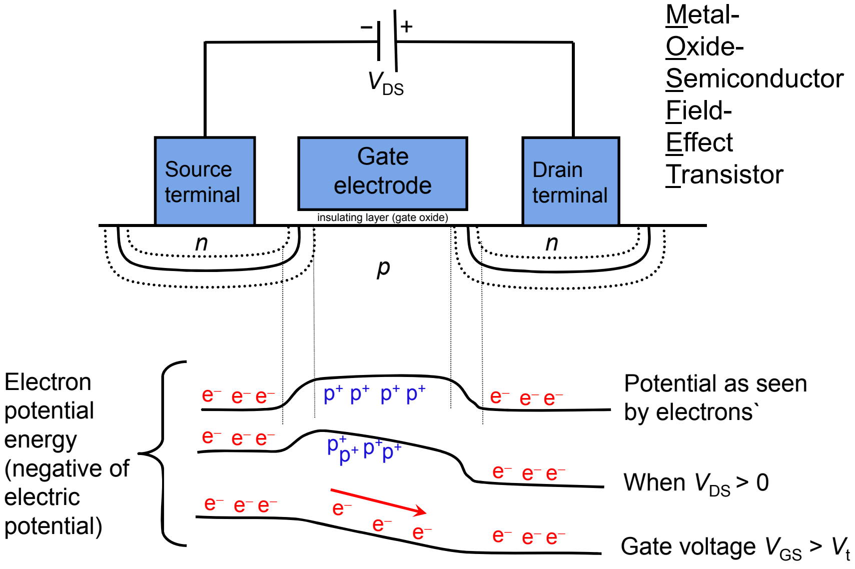


  - Energy dissipated:

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

Note: Adiabatic charging beats the energy efficiency of conventional by advantage factor: $A = \frac{E_{\text{diss}}^{\text{conv}}}{E_{\text{diss}}^{\text{adia}}} = \frac{1}{2}\frac{t}{RC}$

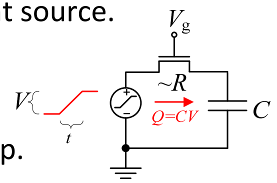
npn MOSFET (*n*-FET)



Metal-
Oxide-
Semiconductor
Field-
Effect
Transistor

Adiabatic Charging via MOSFETs

- Let a voltage ramp *approximate* an ideal current source.
  - The load gets charged *conditionally*, if the MOSFET is turned on (gate voltage $V_g > V + V_t$) during ramp.
    - V_t is the transistor's threshold, typically < ½ volt
- Can discharge the load later using a similar ramp.
  - Either through the same path, or a different path.



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

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

Exact formula:

$$E_{\text{diss}} = s\left[1 + s\left(\mathrm{e}^{-1/s} - 1\right)\right]CV^2$$

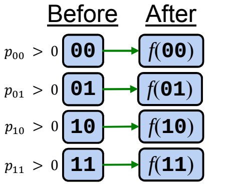
given *speed fraction* $s = RC/t$.

- Note that the (ideal) operation of this circuit approaches *physical reversibility* ($E_{\text{diss}} \to 0$) in the limit $t \to \infty$, but *only* if a certain *precondition* on the initial state is met (namely, $V_g > V_{\max} + V_t$)
  - How does its possible physical reversibility relate to its *computational* function, and to some *appropriate* concept of logical reversibility?
    - Traditional (Landauer/Fredkin/Toffoli) reversible computing theory does **not** adequately address this question, so, we need a more powerful theory!

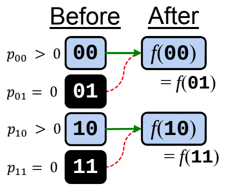
Generalized Reversible Computing

(preview of talk in main conference)

- In contexts where *all* initial computational states can be considered "possible" (*i.e.*, have probability of occurring >0)
  - The traditional definition for logical reversibility of computational operations (*i.e.*, that the transformation of all states is bijective) does correctly identify the logical-level requirements for entropy not to be ejected from the computational state as a result of the operation…
- **BUT**, in operating contexts wherein some of the initial computational states *never arise* (guaranteed by the system design), those states have *probability equal to 0*, and thus *cannot contribute* to the entropy ejected at all – those states *can* merge with others reversibly!
  - Thus, the *correct* statement of the minimum logical-level requirements for reversibility is simply that only *the **subset** of initial states that meet a particular assumed precondition that is obeyed with certainty* must be transformed one-to-one onto final states.
    - In such a case, we say that we have a *logically reversible computation* performing an operation that is itself only *conditionally* (logically) reversible.



Unconditionally logically reversible operation



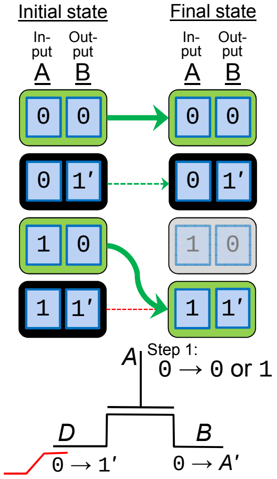
Conditioned logically reversible operation, with the assumed precondition that the 2nd input bit = **0**

Why GRC is the right theoretical foundation for modeling reversible computation in adiabatic circuits!

- E.g.: Even *a single MOSFET* (operated adiabatically) can do a certain (conditioned) reversible COPY operation…
  - Operation sequence is as follows:
    0. Driving node D is initially statically held at 0, input *A* also 0.
    1. Input A is externally supplied (D&B connected iff *A* is high)
    2. Externally transition driver D from 0 to (weak) logic high 1′
    3. Voltage level on node B follows D iff A is strong logic high (**1**)
       - B is then afterwards logically equal to A (with a weak swing)
  - Note: Given a (strong) assumed precondition of $\bar{B}$,
    - *i.e.*, if all initial states with $B = 1$ have prob. 0,
    - this indeed performs a reversible COPY operation, $\mathrm{rCOPY}(A, B \mid \bar{B}\ )$.
  - Note: The output in this case is not full-swing,
    - In this diagram, primes (′) denote reduced-voltage logic high signals
      - If we need full-swing outputs, we can use a transmission gate (parallel nFET/pFET pair) with complementary controls
  - A notation precisely describing this operation's semantics is:
    - $[\overline{AB}]$ if $B = 0$ then $B \coloneqq A$ *(else, leave state unchanged)*
      - The expression $\overline{AB}$ in brackets gives the ***precondition for reversibility*** for the entire operation (the operation is both logically reversible and asymptotically thermodynamically reversible unless $A = B = 1$).
      - The remainder of the statement describes exactly how the state will be transformed in all cases (even if the precondition is not met).
- Note: Traditional reversible computing theory based on unconditionally reversible operations is insufficient to express the logical/physical reversibility of this operation!

Reversible COPY

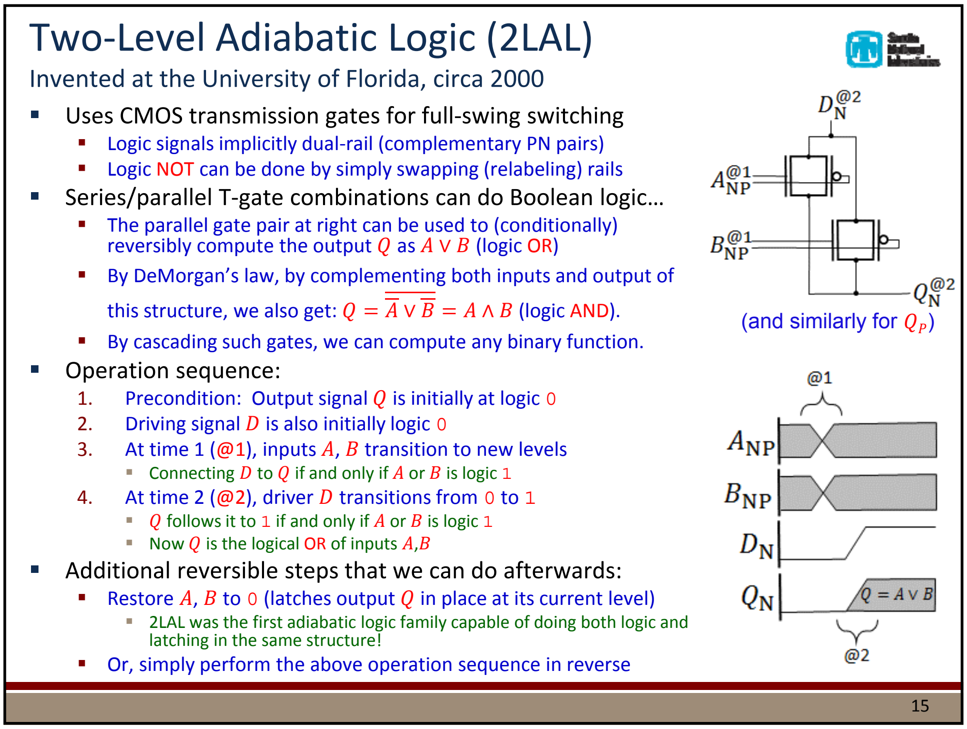
$\mathrm{rCOPY}(A, B \mid \bar{B}\ )$



(Here, D and B have a reduced swing, but a T-gate can easily fix this)

All truly, fully adiabatic circuits are conditionally reversible!

- "Dry switching" rules for designing truly adiabatic circuits:
  - Never close a switch when there's a voltage ≠0 between its terminals
    - E.g., don't turn on a transistor when $V_{\mathrm{DS}} \neq 0$.
  - Never open a switch when there's a current passing through it.
    - E.g., don't turn off a transistor when $I_{\mathrm{DS}} \neq 0$.
    - Only exception to this rule: If there's an alternate path for the current.
  - Never pass current through diodes (which have a voltage drop)
- Violating any of these rules leads to significant dissipation!
- **Theorem:** The operation of a switching circuit carries out a (conditionally) logically reversible computation, in any operation context where the above rules are always satisfied.
  - *It's impossible to erase information in any truly, fully adiabatic logic operation.* → Logically-reversible computing is key to adiabatic design
    - But, the *right* definition of "logically reversible" is our generalized one!

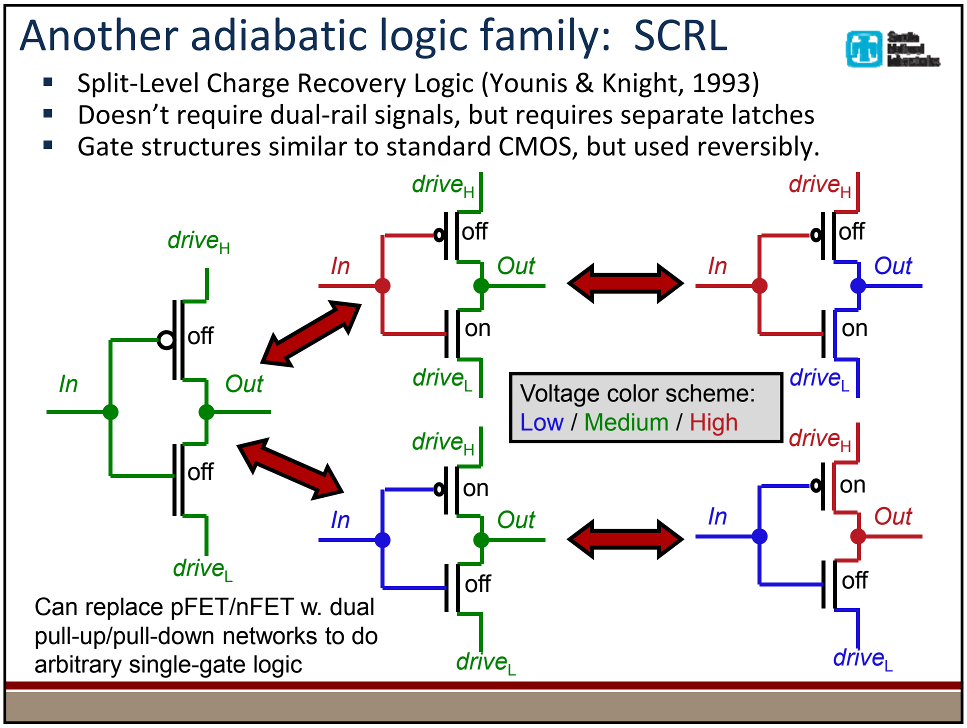


Two-Level Adiabatic Logic (2LAL)

Invented at the University of Florida, circa 2000

- Uses CMOS transmission gates for full-swing switching
  - Logic signals implicitly dual-rail (complementary PN pairs)
  - Logic NOT can be done by simply swapping (relabeling) rails
- Series/parallel T-gate combinations can do Boolean logic…
  - The parallel gate pair at right can be used to (conditionally) reversibly compute the output Q as $A \vee B$ (logic OR)
  - By DeMorgan's law, by complementing both inputs and output of this structure, we also get: $Q = \overline{\overline{A} \vee \overline{B}} = A \wedge B$ (logic AND).
  - By cascading such gates, we can compute any binary function.
- Operation sequence:
  1. Precondition: Output signal Q is initially at logic 0
  2. Driving signal D is also initially logic 0
  3. At time 1 (@1), inputs A, B transition to new levels
     - Connecting D to Q if and only if A or B is logic 1
  4. At time 2 (@2), driver D transitions from 0 to 1
     - Q follows it to 1 if and only if A or B is logic 1
     - Now Q is the logical OR of inputs A,B
- Additional reversible steps that we can do afterwards:
  - Restore A, B to 0 (latches output Q in place at its current level)
    - 2LAL was the first adiabatic logic family capable of doing both logic and latching in the same structure!
  - Or, simply perform the above operation sequence in reverse

(and similarly for Q_P)



Another adiabatic logic family: SCRL

- Split-Level Charge Recovery Logic (Younis & Knight, 1993)
- Doesn't require dual-rail signals, but requires separate latches
- Gate structures similar to standard CMOS, but used reversibly.

Voltage color scheme:
Low / Medium / High

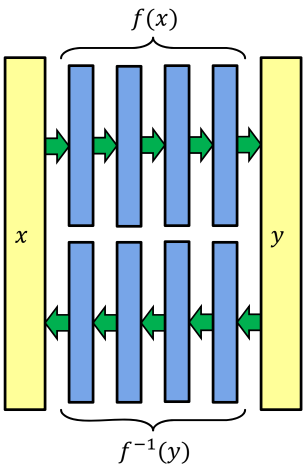
Can replace pFET/nFET w. dual pull-up/pull-down networks to do arbitrary single-gate logic

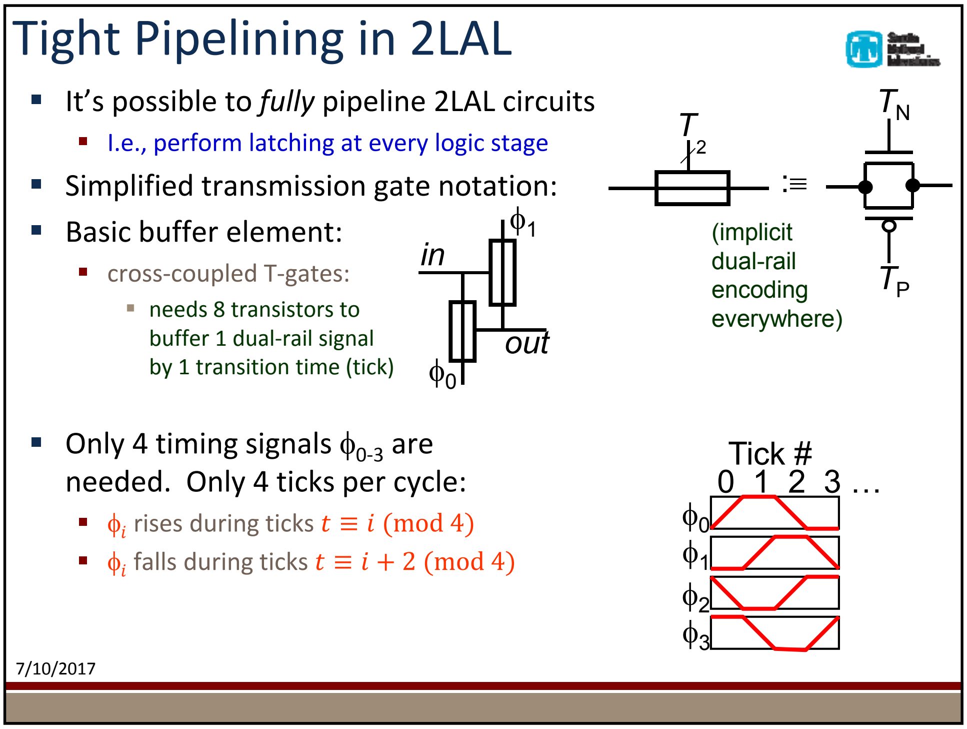
Dynamic vs. Static Latching

- Generally speaking, there are two ways data can be latched (stored) in a CMOS circuit:
  - Dynamic latching: Data is preserved capacitively on a “floating” node
    - Disadvantage: Data can be invalidated by leakage
  - Static latching: Data is preserved by a persistent connection to a voltage reference
    - This tends to be essential to store data long-term without refreshing
      - Particularly useful in adiabatic circuits to prevent $\frac{1}{2}CV_{\mathrm{drift}}^2$ energy loss on refresh
    - Disadvantage: Typically requires extra circuit complexity to implement.
- The same dichotomy applies in adiabatic circuits
  - Note: Leakage still results in extra power dissipation regardless
- To implement static latching in adiabatic circuits:
  - Make sure node is connected to at least one voltage source at all times
    - This may require taking additional steps (we’ll see an example later)

Combinational and Sequential Logic in Reversible Adiabatic Circuits

- A general picture of how to combine combinational and sequential logic in reversible, adiabatic logic designs:
  1. Initially, input x is in register at left.
  2. Evaluate top stages, in sequence, to produce output $y = f(x)$ in register at right.
     - Hall’s “retractile cascade” method
  3. Latch output y in place.
  4. Unroll evaluation of top stages.
- If f is an invertible function, we can then *decompute* the input x as follows:
  5. Evaluate bottom stages, following arrows, to compute a copy of $x = f^{-1}(y)$,
  6. Unlatch (connect) x to the presented copy,
  7. Unroll evaluation of bottom stages.
- At conclusion of this process, information has moved and transformed from x to y.
  - Can then go through further stages, and meanwhile process new inputs (pipelining)…





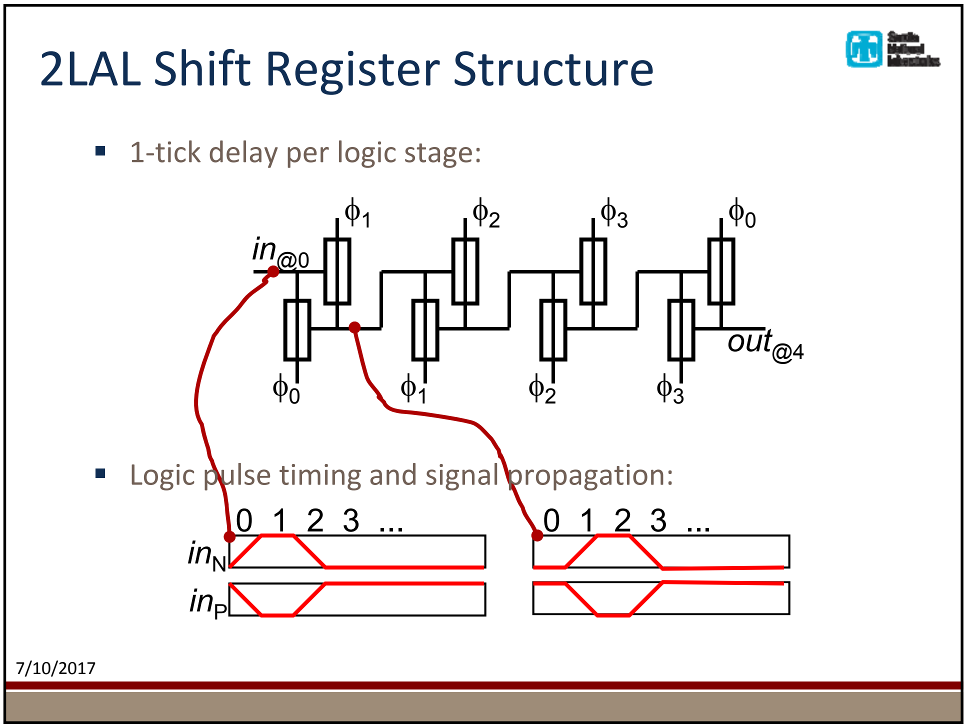
Tight Pipelining in 2LAL

- It's possible to *fully* pipeline 2LAL circuits
  - I.e., perform latching at every logic stage
- Simplified transmission gate notation:
- Basic buffer element:
  - cross-coupled T-gates:
    - needs 8 transistors to buffer 1 dual-rail signal by 1 transition time (tick)

(implicit dual-rail encoding everywhere)

- Only 4 timing signals $\phi_{0\text{-}3}$ are needed. Only 4 ticks per cycle:
  - ϕ_i rises during ticks $t \equiv i \pmod 4$
  - ϕ_i falls during ticks $t \equiv i + 2 \pmod 4$

7/10/2017



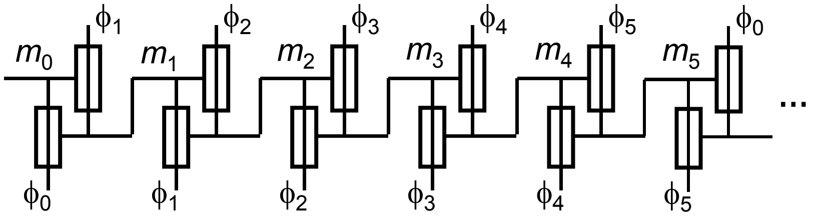
2LAL Shift Register Structure

- 1-tick delay per logic stage:
- Logic pulse timing and signal propagation:

7/10/2017

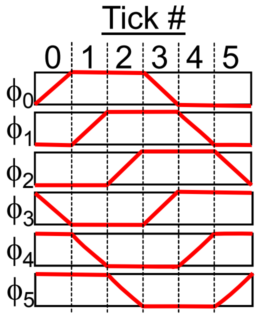
Statically-Latched 2LAL Shift Register

The previous circuit used dynamic latching, but we can also latch statically in 2LAL…



Six timing signals $\phi_{0\text{-}5}$ are needed in this case.
Need six ticks per cycle:

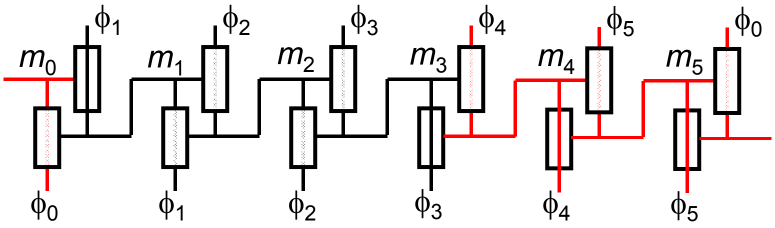
- ϕ_i rises during ticks $t \equiv i \pmod 6$
- ϕ_i falls during ticks $t \equiv i + 3 \pmod 6$

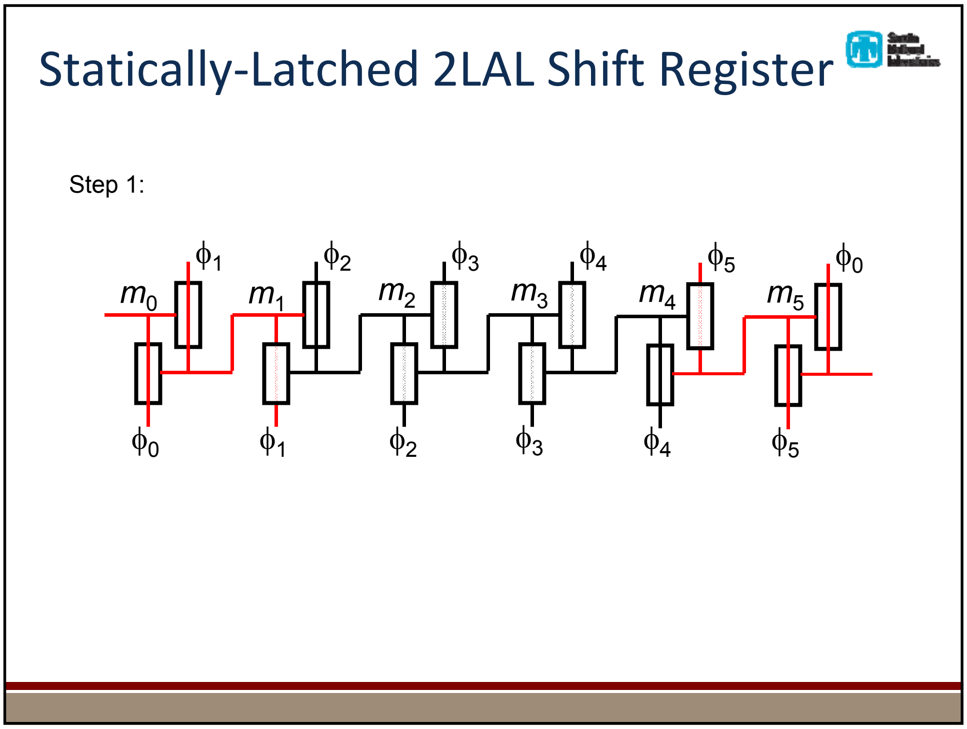


7/10/2017

Statically-Latched 2LAL Shift Register

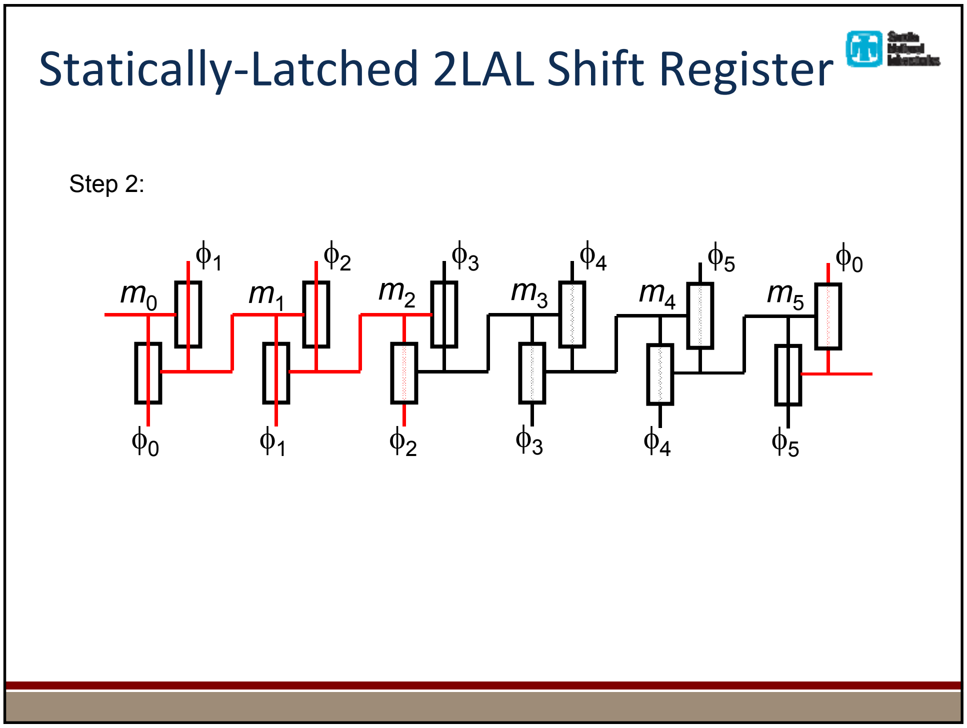
Step 0:





Statically-Latched 2LAL Shift Register

Step 1:

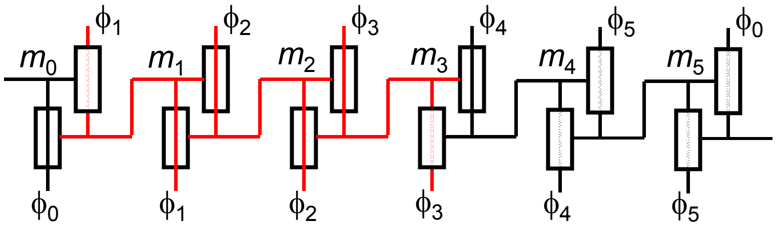


Statically-Latched 2LAL Shift Register

Step 2:

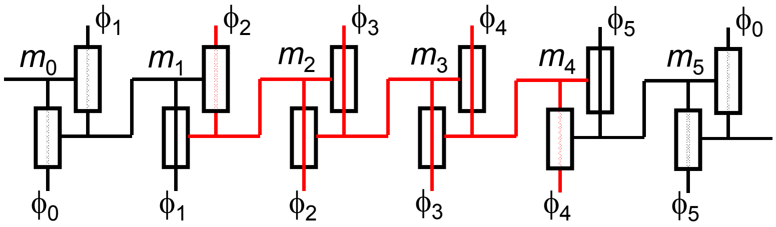
Statically-Latched 2LAL Shift Register

Step 3:



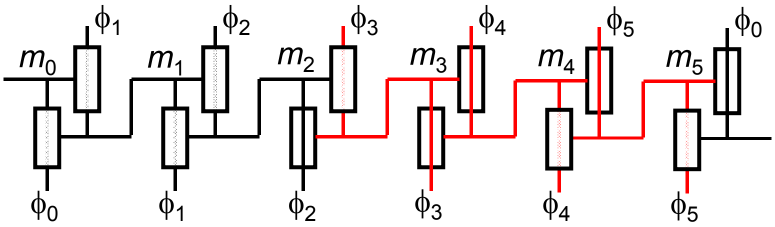
Statically-Latched 2LAL Shift Register

Step 4:



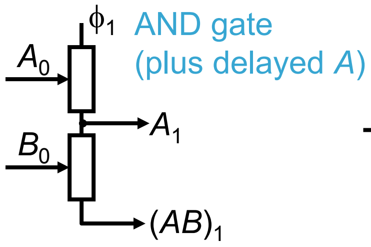
Statically-Latched 2LAL Shift Register

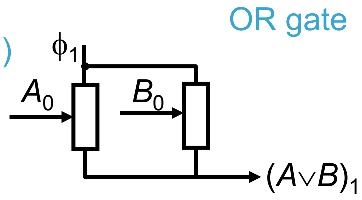
Step 5:



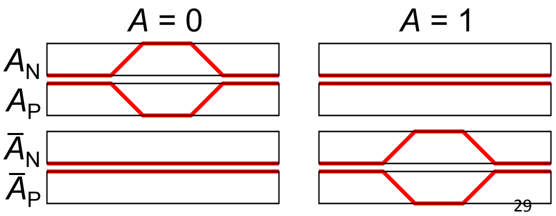
More Complex Logic Functions

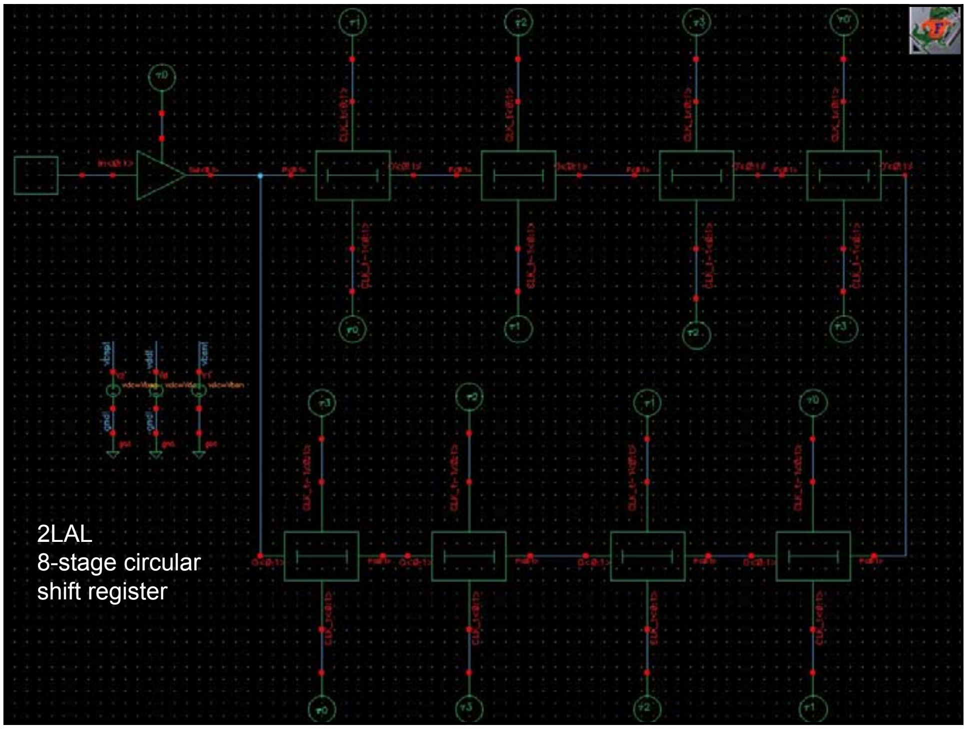
- Non-inverting multi-input Boolean functions:





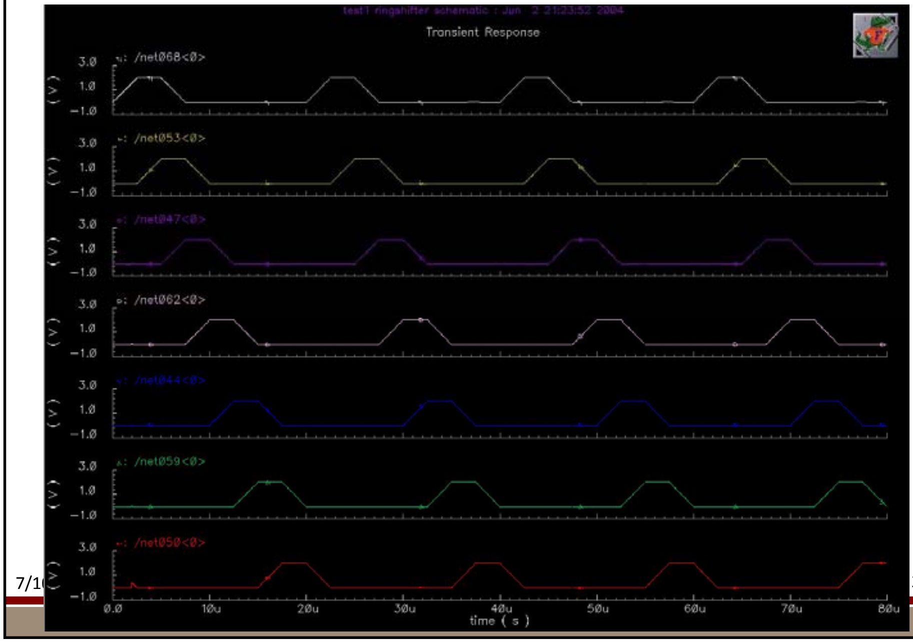
- One way to do inverting functions in fully-pipelined latching logic is to use a quad-rail logic encoding:
  - To invert, swap the complementary pairs
    - Zero-transistor “inverters.”

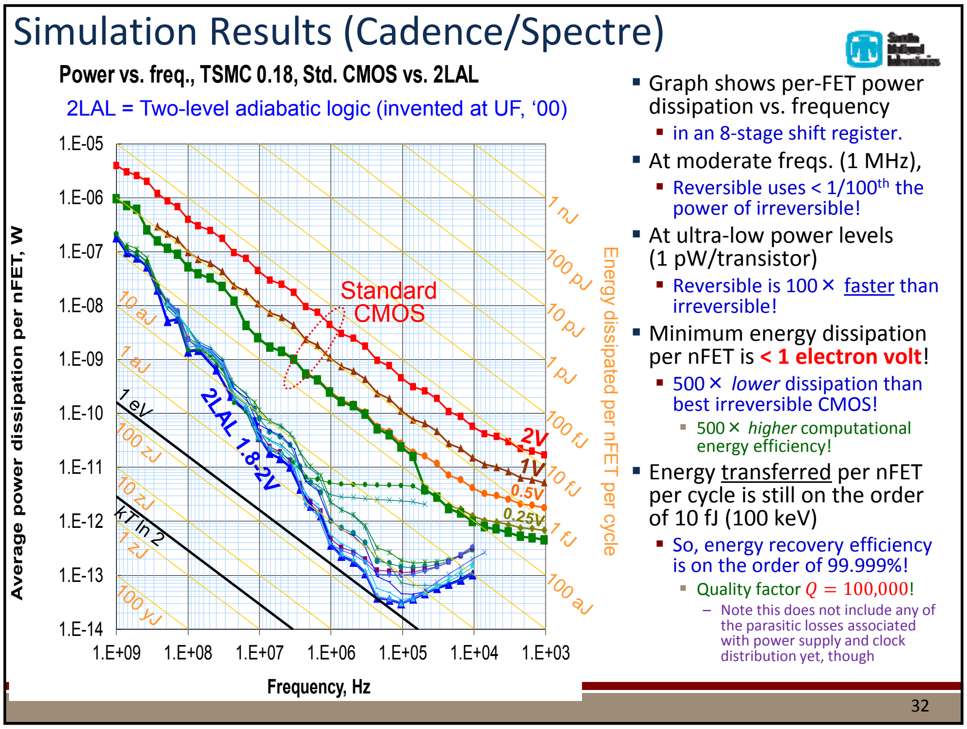




2LAL
8-stage circular
shift register

Pulse propagation in 8-stage circuit



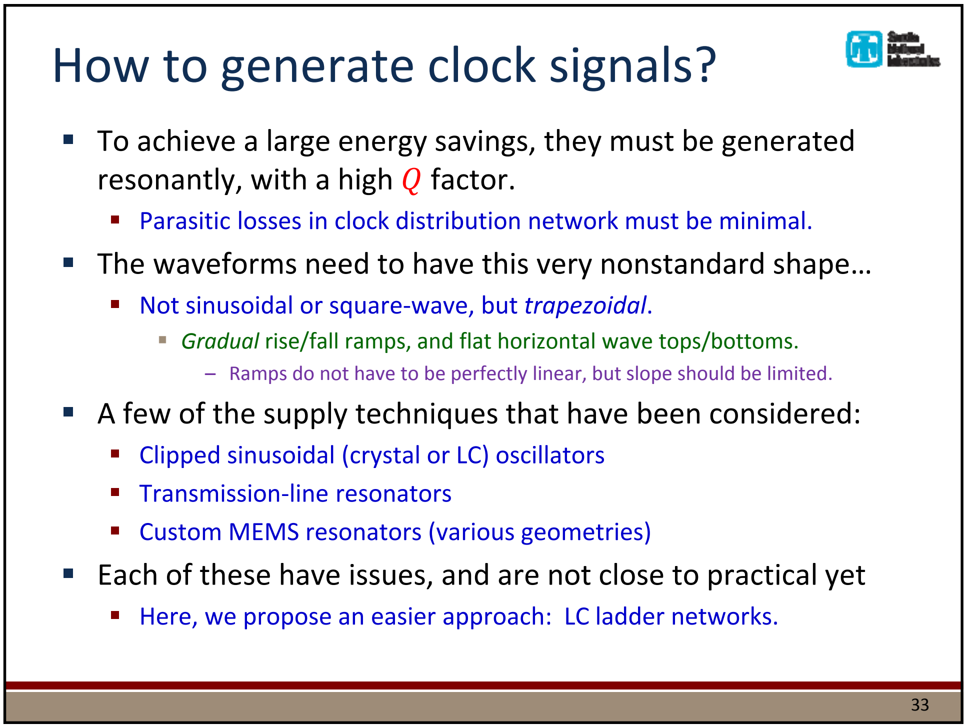


Simulation Results (Cadence/Spectre)

Power vs. freq., TSMC 0.18, Std. CMOS vs. 2LAL

2LAL = Two-level adiabatic logic (invented at UF, '00)

- Graph shows per-FET power dissipation vs. frequency
  - in an 8-stage shift register.
- At moderate freqs. (1 MHz),
  - Reversible uses < 1/100th the power of irreversible!
- At ultra-low power levels (1 pW/transistor)
  - Reversible is 100 × faster than irreversible!
- Minimum energy dissipation per nFET is **< 1 electron volt**!
  - 500 × *lower* dissipation than best irreversible CMOS!
    - 500 × *higher* computational energy efficiency!
- Energy transferred per nFET per cycle is still on the order of 10 fJ (100 keV)
  - So, energy recovery efficiency is on the order of 99.999%!
    - Quality factor $Q = 100{,}000$!
      - Note this does not include any of the parasitic losses associated with power supply and clock distribution yet, though



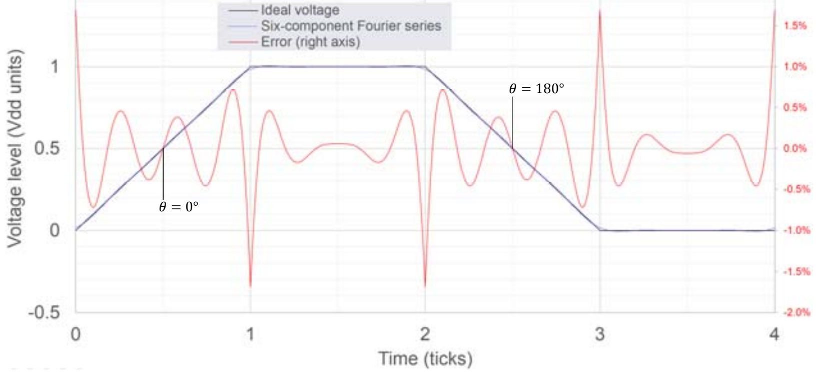
How to generate clock signals?

- To achieve a large energy savings, they must be generated resonantly, with a high Q factor.
  - Parasitic losses in clock distribution network must be minimal.
- The waveforms need to have this very nonstandard shape…
  - Not sinusoidal or square-wave, but *trapezoidal*.
    - *Gradual* rise/fall ramps, and flat horizontal wave tops/bottoms.
      - Ramps do not have to be perfectly linear, but slope should be limited.
- A few of the supply techniques that have been considered:
  - Clipped sinusoidal (crystal or LC) oscillators
  - Transmission-line resonators
  - Custom MEMS resonators (various geometries)
- Each of these have issues, and are not close to practical yet
  - Here, we propose an easier approach: LC ladder networks.

Spectrum of Trapezoidal Wave

- Relative to mid-level crossing, waveform is an odd function
  - Spectrum includes only odd harmonics $f, 3f, 5f, \ldots$
- Six-component Fourier series expansion is shown below
  - Maximum offset with $11f$ frequency cutoff is $< 1.7\%$ of V_{dd}

$$v_{f6}(t) = V_{dd}\left[\frac{1}{2} + \frac{4\sqrt{2}}{\pi^2}\left(\sin\theta + \frac{\sin 3\theta}{3^2} - \frac{\sin 5\theta}{5^2} - \frac{\sin 7\theta}{7^2} + \frac{\sin 9\theta}{9^2} + \frac{\sin 11\theta}{11^2}\right)\right]$$



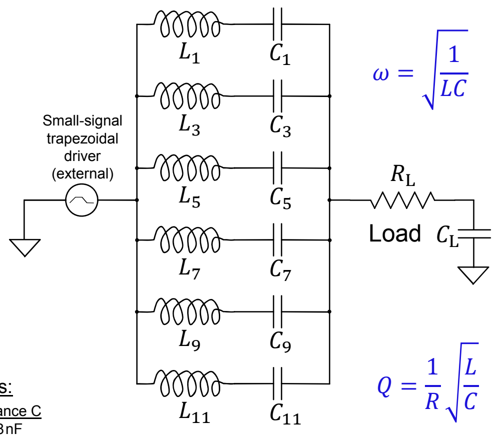
Ladder Resonator Structure

- Can build trapezoidal resonator w. a ladder circuit made of parallel passive bandpass filters, each a sinusoidal LC resonator
  - Each "rung" of ladder passes a different odd multiple of the fundamental clock frequency f
  - Adjust L/C ratio to obtain a target Q value on each path, given parasitic R, C values
- Excite the circuit with a driving signal containing the right distribution of frequency component amplitudes
  - Each frequency component gets amplified by the Q value of its corresponding rung
  - If all rungs are designed to the same target Q, we can just use a trapezoidal driver
- For high Q, clock period must be long compared to the total parasitic RC…
  - Max. possible $Q_n = \frac{1}{2\pi} \cdot \frac{t_{\mathrm{period},n}}{(RC)_{\mathrm{parasitic}}}$

(for $V_{\mathrm{dd}} \approx 1.75\ \mathrm{V}\ \downarrow$)

Example values:

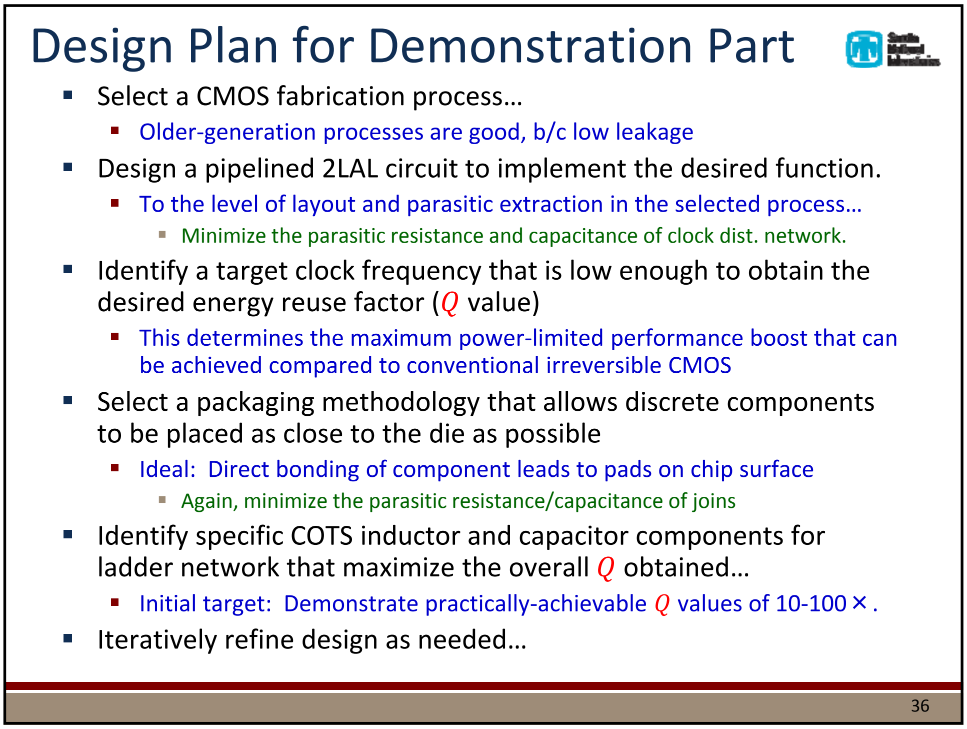
harmonic mode (n)	frequency f	component amplitude Va	inductance L	capacitance C
1	230 kHz	1000.00 mV	691.98 nH	691.98 nF
3	690 kHz	111.11 mV	230.66 nH	230.66 nF
5	1150 kHz	-40.00 mV	138.40 nH	138.40 nF
7	1610 kHz	-20.41 mV	98.85 nH	98.85 nF
9	2070 kHz	12.35 mV	76.89 nH	76.89 nF
11	2530 kHz	8.26 mV	62.91 nH	62.91 nF



$$\omega = \sqrt{\frac{1}{LC}}$$

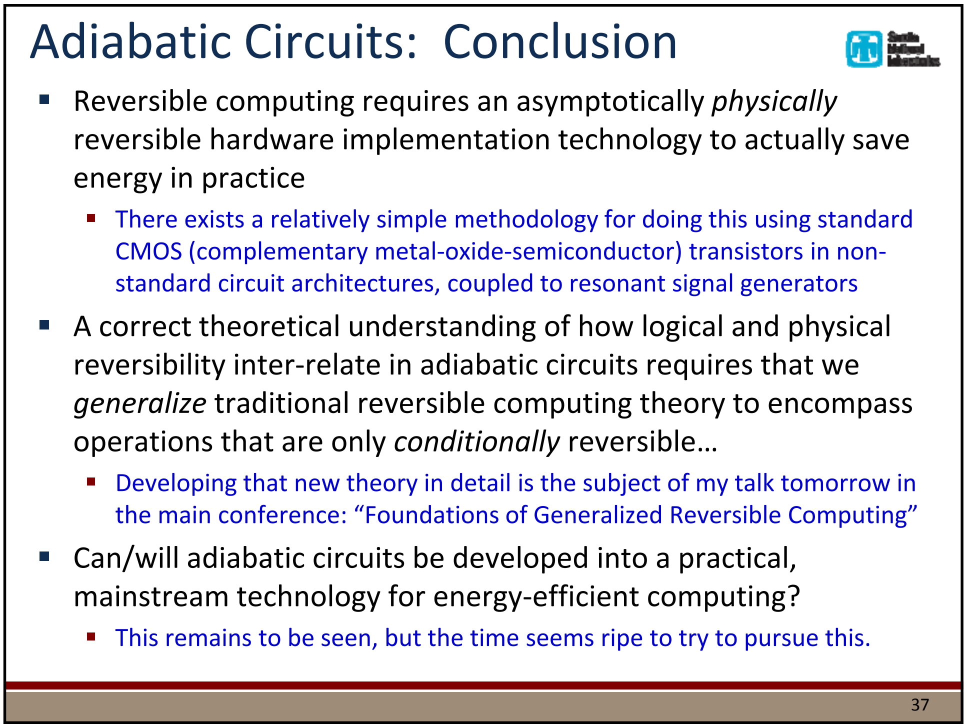
$$Q = \frac{1}{R}\sqrt{\frac{L}{C}}$$

Ladder Resonator for Odd Harmonics



Design Plan for Demonstration Part

- Select a CMOS fabrication process…
  - Older-generation processes are good, b/c low leakage
- Design a pipelined 2LAL circuit to implement the desired function.
  - To the level of layout and parasitic extraction in the selected process…
    - Minimize the parasitic resistance and capacitance of clock dist. network.
- Identify a target clock frequency that is low enough to obtain the desired energy reuse factor (Q value)
  - This determines the maximum power-limited performance boost that can be achieved compared to conventional irreversible CMOS
- Select a packaging methodology that allows discrete components to be placed as close to the die as possible
  - Ideal: Direct bonding of component leads to pads on chip surface
    - Again, minimize the parasitic resistance/capacitance of joins
- Identify specific COTS inductor and capacitor components for ladder network that maximize the overall Q obtained…
  - Initial target: Demonstrate practically-achievable Q values of 10-100×.
- Iteratively refine design as needed…



Adiabatic Circuits: Conclusion

- Reversible computing requires an asymptotically *physically* reversible hardware implementation technology to actually save energy in practice
  - There exists a relatively simple methodology for doing this using standard CMOS (complementary metal-oxide-semiconductor) transistors in non-standard circuit architectures, coupled to resonant signal generators
- A correct theoretical understanding of how logical and physical reversibility inter-relate in adiabatic circuits requires that we *generalize* traditional reversible computing theory to encompass operations that are only *conditionally* reversible…
  - Developing that new theory in detail is the subject of my talk tomorrow in the main conference: "Foundations of Generalized Reversible Computing"
- Can/will adiabatic circuits be developed into a practical, mainstream technology for energy-efficient computing?
  - This remains to be seen, but the time seems ripe to try to pursue this.