# Chapter 7

# Adiabatic circuits

Part I of this thesis explored the general motivation for and properties of reversible machines, without reference to any particular implementation technology. In Part II, beginning with this chapter, we address a variety of engineering and implementation issues in reversible computing, showing ways to actually design, build, and program reversible computers.

Virtually all computers today are built using semiconductor VLSI (very-large scale integration) technology, in which, typically, metal-oxide-semiconductor field-effect transistors (MOSFETs) are wired together to form CMOS (complementary MOS) logic gates.

Unfortunately, the way these CMOS logic gates are currently designed, they are operationally irreversible, and thus have fixed lower bounds on their energy dissipation and entropy generation, which we will analyze in some detail in §7.1. These bounds have consequences for the maximum cost-efficiency of computation using irreversible CMOS (hereafter abbreviated "iCMOS") under various cost measures.

The irreversibility of traditional logic elements has led researchers to ask whether transistor electronics could instead be configured in such a way as to form reversible logic elements. The answer is yes, there is a class of reversible logic circuit styles called "adiabatic" circuits (a somewhat misleading name, as we will explain in §7.3), whose history we will review. We will then describe in detail *SCRL* (split-level charge recovery logic), a particular variant of adiabatic circuit technology that was developed a few years ago by members of our research group [192, 193, 191, 194]. SCRL has several properties that make it particularly suitable for achieving the asymptotic efficiency benefits that we discussed in ch. 6. It is capable of full reversibility. It can be built cheaply and easily using today's commercially available VLSI fabrication processes. And SCRL's energy dissipation roughly matches the TPRA (time-proportionately reversible) model of chapter 6.

Next, we describe our SCRL-based design of FlatTop, a simple adiabatic circuit that we recently fabricated [72]. This circuit can be viewed as the first ever universal reversible processor core, capable, in principle, of fulfilling the scaling laws derived in the previous chapter, and thereby computing asymptotically faster than any irreversible technology in machines at a sufficiently large scale. (However, in practice, the FlatTop chip is really just a proof of concept.)

Unfortunately, with some additional analysis (not contained in this thesis) we have found that the constant factors of present-day VLSI (very large-scale integration) semiconductor technology imply that the cost level at which reversible computing will actually dominate in speed is, in the present technology generation, economically infeasible (roughly speaking, in the range of billions of dollars). In the following chapter, we will review a variety of possible future device technologies for which, in contrast, reversibility wins out for all but the tiniest of machines. Then, chapter 9 will review reversible programming issues.

## 7.1 Maximizing the efficiency of iCMOS

Before we describe SCRL, in this section we will establish a baseline for comparison by roughly estimating the optimum performance of present and future irreversible CMOS, under a variety of cost-efficiency measures.

The central motivation for adiabatic circuits is the avoidance of the $\sim CV^2$ energy dissipation that (as we will see) is necessarily incurred by ordinary irreversible logic circuits whenever they switch a signal from one logic level to another. In this section we briefly review traditional irreversible CMOS logic, and the reasons for its energy dissipation, and analyze in some detail the present and future minimum energy dissipation of such circuits, based on the semiconductor industry's technology projections for the next decade. This analysis can serve as a baseline for comparison when we wish to look at the capabilities of adiabatic circuits.

### 7.1.1 Basic iCMOS review

A detailed description of irreversible CMOS technology can be found in any standard, reasonably recent VLSI textbook, for example [84, 187, 139]. Here, we only briefly review the basics.

CMOS logic gates may appear in the simple *static* form, or in various *dynamic* variations. In this context, "static" and "dynamic" refer to whether the output of the logic gate is always tied to a fixed voltage source, or is sometimes allowed to float freely. The underlying energy issues in the two circuit styles are basically similar, so to ease our analysis, we will focus on the simplest static logic style. The use of alternative styles of irreversible CMOS logic can improve energy efficiency beyond that of ordinary static CMOS by, at most, only a small constant factor.

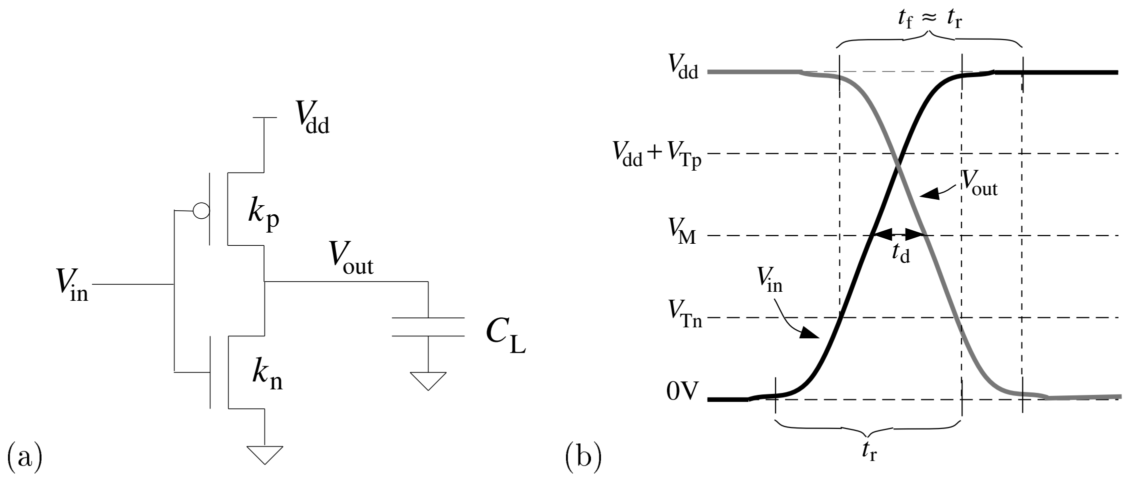



Figure 7.1: (a) An ordinary CMOS inverter, consisting of an n-FET (bottom) and a p-FET (top). Given an input logic value $A$, the inverter computes its inverse $\bar{A}$. With more complex networks of p-FETs and n-FET in the pull-up and pull-down networks, arbitrary inverting logic functions of multiple inputs can be similarly implemented.

(b) Dynamic behavior. The n-FET conducts significantly when $V_{\rm in}$ is above the threshold voltage $V_{\rm Tn}$, and the p-FET conducts when $V_{\rm in}$ is below the level $V_{\rm dd} + V_{\rm Tp}$ (where $V_{\rm Tp}$ is negative). When the input voltage $V_{\rm in}$ goes to the high level $V_{\rm dd}$, representing a logic `1`, the output voltage $V_{\rm out}$ goes to the low level of 0 V, representing a `0`, and vice-versa. The delay $t_{\rm d}$ is affected by the *load capacitance* $C_{\rm L}$, the supply and threshold voltages, and the *gain factors* (also called *device transconductance parameters*) $k_{\rm n}$ and $k_{\rm p}$ of the n-FET and p-FET devices. If the inverter is driven by a similar inverter and if $k_{\rm p} \approx k_{\rm n}$, then the rise and fall times $t_{\rm r}$ and $t_{\rm f}$ of input and output will be about equal.

---

Figure 7.1 shows a static CMOS inverter, consisting of two MOSFETs, one n-type and one p-type. (The n and p refer to whether the primary charge carriers in the device are negatively charged electrons or positively charged "holes.") The p-FET connects the output to a high voltage when the input is low, and the n-FET connects the output to a low voltage when the input is high.

The inverter structure can be generalized to compute any *inverting* boolean function of many inputs (a one-bit function that is monotonically non-increasing in the values of the input bits), by replacing the p-FET and n-FET with appropriate networks of many p-FET and n-FET devices, respectively.

Now, suppose we wish to minimize the total entropy generation of a computation. To see how to do this, let consider how energy dissipation in a CMOS circuit scales with various parameters.

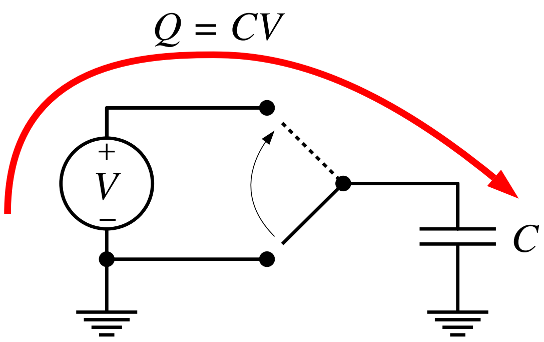



Figure 7.2: Energy dissipation in conventional switching. Whenever a node is switched by connecting it to a constant-voltage $V$ power supply, such as occurs in the ordinary CMOS gate of fig. 7.1, the transfer of the charge $Q = CV$ from the constant-voltage source to the node capacitance $C$ results in an energy dissipation of $\frac{1}{2}CV^2$, where $V$ is the voltage swing and $C$ is the capacitance being switched.

---

#### 7.1.1.1 iCMOS energy dissipation

We have seen that standard CMOS circuits charge and discharge loads by connecting them to constant-voltage power supplies. Because of this, whenever the voltage of a node is switched from one level to another, through a voltage change $V$, the energy dissipated is at least $\frac{1}{2}CV^2$, where $C$ is the total capacitance being switched.

To understand this, refer to fig. 7.2. An amount $Q = CV$ of charge is delivered to the node from voltage $V$, so the energy supplied is at least $E_{\text{supp}} = QV = CV^2$. The energy stored on the node is

$$\begin{aligned} E_{\text{stor}} &= \int dE = \int v(q)\,dq = \int \frac{q}{C}\,dq = \frac{1}{C}\int_0^Q q\,dq \\ &= \frac{1}{C}\;\frac{1}{2}q^2\Big|_{q=0}^{Q} = \frac{1}{2C}Q^2 = \frac{1}{2C}(CV)^2 \\ &= \frac{1}{2}CV^2. \end{aligned}$$

(For additional textual explanation, see [143], §27-4, p. 521.) The remaining energy $E_{\text{supp}} - E_{\text{stor}} = CV^2 - \frac{1}{2}CV^2 = \frac{1}{2}CV^2$ can not be accommodated in the circuit model; no matter the precise mechanism by which it has left the system, this energy is now considered unmodeled, statistical, thermalized; so as a matter of definition we call it "dissipated." Essentially it becomes heat.

This energy dissipation and the accompanying entropy increase occurs under *any* means of setting a voltage by delivering charge directly from a constant-voltage source.

There is no way it can be avoided by, for example, simply adjusting the resistance or inductance of the switch: note that the above analysis depends in no way on such parameters. We call this $\frac{1}{2}CV^2$ dissipation the *switching energy* $E_{\mathrm{sw}}$.

Of course, if the circuit is a small part of a larger system, then the dissipation to heat of a certain amount of energy by the circuit need not imply that this energy will never again be available for work: for example, given a cool thermal reservoir, the computer can, as a side effect, power a heat engine which can recover a portion of the dissipated energy as work. But in the absence of any specific mechanism tailored to capture the supplied energy in a more direct way, the energy $E = \frac{1}{2}CV^2$ must at still at least be *temporarily* dissipated (thermalized), which results in an immediate, irreversible entropy generation of at least $S = CV^2/2T$, where $T$ is the temperature of the circuit.

Moreover, if the switch that we are using is a static CMOS logic gate, there may be additional dissipation in the gate if the n-FET and p-FET are simultaneously conducting significantly during some portion of the input transition, resulting in a current from power to ground through both transistors. Generally this will occur if $V_{\mathrm{dd}} \gtrsim V_{\mathrm{Tn}} + |V_{\mathrm{Tp}}|$.

This *short-circuit dissipation* is somewhat more complex to model than switching energy, because it depends on the dynamic behavior of the CMOS devices and the input signal. But we can make some useful approximations when the input is driven by a similar gate: if $V_{\mathrm{dd}} \gg V_{\mathrm{Tn}} + |V_{\mathrm{Tp}}|$, then a rough dynamic analysis shows that the short-circuit dissipation $E_{\mathrm{ss}}$ will be around $CV_{\mathrm{dd}}^2$, comparable to the switching energy. However, if $V_{\mathrm{dd}} \lesssim V_{\mathrm{Tn}} + |V_{\mathrm{Tp}}|$, then the short-circuit dissipation will be small compared to the switching energy.

Finally, for sufficiently slowly-switching CMOS circuits, another source of dissipation may become important, namely leakage energy due to sub-threshold conduction of MOS devices even when turned off. Due to thermal effects, the transistor off-current cannot be made less than the on-current by more than a factor of roughly $f = e^{V_{\mathrm{dd}}/\phi_T}$, where $\phi_T = k_{\mathrm{B}}T/q_{\mathrm{e}}$ is the thermal voltage of electrons in a device at temperature $T$. Therefore when device idle times are more than a factor of $f$ times larger than switching times, the *leakage energy* $E_{\mathrm{leak}}$ due to the off-current may be the dominant contributor to the total energy dissipation per operation.

### 7.1.2 iCMOS entropy generation.

All of the total energy $E_{\mathrm{tot}} = E_{\mathrm{sw}} + E_{\mathrm{ss}} + E_{\mathrm{leak}}$ dissipated per operation, due to these various causes, becomes heat, generating an amount of new entropy $S = E_{\mathrm{tot}}/T$, where $T$ is the operating temperature of the devices. This entropy generation can be made smaller by raising the operating temperature, but only up to a point, because higher temperatures eventually lead to large leakage currents, and higher operating voltages (because we must obey $V_{\mathrm{dd}} \gtrsim \phi_T$ or else transistors will not be significantly more conductive when on than when off), and will eventually preclude correct functionality altogether, by melting the device, if not by some other effect that cripples the device's functionality far below its melting temperature.

The relationships between all these factors are complex, but if we just assume that there is some maximum temperature $T_{\max}$ for a working CMOS device, and if as an initial rough estimate we place $T_{\max}$ at around 150 degrees Celsius (423 K) (we do not know of any semiconductor chips that operate hotter than this), then this gives us a rough lower bound on entropy generation of $S \geq \frac{1}{2}CV^2/(423K) = 1.18\times10^{-3}CV^2/K = (86\ \mathrm{nat/aJ})CV^2$, which is an unavoidable lower bound as long as $C$ and $V$ are assumed fixed.

We can still lower bound the entropy generation even if $V$ is allowed to vary. Since we know that $V \gtrsim \phi_T = k_{\mathrm{B}}T/q_{\mathrm{e}}$ for correct functionality, we can define a new lower bound in terms of temperature and load capacitance:

$$\begin{aligned} S &\geq \frac{1}{2}CV^2/T \\ &\gtrsim \frac{C(k_{\mathrm{B}}T/q_{\mathrm{e}})^2}{2T} \\ &= \frac{Ck_{\mathrm{B}}^2T}{2q_{\mathrm{e}}^2}. \end{aligned}$$

Or, if we define a quantity $C_T \equiv q_{\mathrm{e}}/\phi_T$, which we will call the *thermal capacitance*, we have

$$S \gtrsim \frac{1}{2} \cdot \frac{C}{C_T}\ \mathrm{nat}. \tag{7.1}$$

For example, at a temperature of 300 K (room temperature), the thermal capacitance is $C_T \approx 6.2\times10^{-3}$ fF, so our entropy bound becomes

$$S \gtrsim (80.7\ \mathrm{nat/fF})\ C.$$

Since $\phi_T \propto T$, $C_T \propto 1/T$ and so the minimum $S$ in (7.1) scales as $T$, showing that actually, when voltage is adjustable, low temperatures are best for minimizing entropy generation, at least until the point where the minimum $V_{\mathrm{dd}}$ is no longer limited only by the thermal voltage.

The form of eq. (7.1) might seem to suggest that ordinary static CMOS circuits could theoretically generate less than a bit of entropy per switching operation, at any given temperature, if the node capacitance is just made sufficiently small compared to the thermal capacitance. However, we will now see that the capacitance and voltage *cannot* together be made low enough to generate less than 1 bit of entropy per switching event, while still permitting reliable operation of these irreversible circuits.

What is the interpretation of the thermal capacitance $C_T$? First, from our above definition, it is the node capacitance for which at least $1/2$ nat of entropy is generated by dissipative switching through a voltage swing of $\phi_T$. But it is also the node capacitance at which the addition of *one electron* raises the node voltage by $\phi_T$. Since individual electrons experience an average excitation equal to the thermal energy $E_T = k_\mathrm{B}T$, the voltage on a node with capacitance only $C_T$ will routinely be different than the expected level by amounts comparable to the thermal voltage. Therefore such a node cannot reliably store a logic value encoded by a voltage difference of only $\phi_T$. For that, a significantly greater capacitance is required.

Additionally, switching a node by tying it to a constant-voltage source at voltage $V$ is a logically irreversible operation, since after the transition is completed, the information about the previous logical state of the node has been lost. (There is simply no mechanism in these simple circuits for retaining this information in a controlled form, just as there is no mechanism, for the $\frac{1}{2}CV^2$ of supplied energy that doesn't end up on the capacitor, of retaining that energy in a controlled form, *i.e.*, not heat.) So, if the node was equally likely to be in either of two states before the operation, one of them at voltage $V$ and the other at voltage 0, and after the operation it is reliably in a single state (corresponding to voltage $V$) then the average entropy generation is at least 1 bit $= \ln 2$ nat. Thus the average energy dissipation must be $k_\mathrm{B}T \ln 2$ to provide for this entropy (see §2.5.3, p. 46).

Since no energy is dissipated in the half of the cases when the node is unchanged, the energy dissipation in the other half of the cases must be twice this, $2k_\mathrm{B}T \ln 2$, in order for the average dissipation to be $k_\mathrm{B}T \ln 2$. In order for the switching energy $\frac{1}{2}CV^2$ to be greater than this, we must have

$$\begin{aligned} \frac{1}{2}CV^2 &\geq 2(\ln 2)k_\mathrm{B}T = 2(\ln 2)E_T \\ C &\geq 4(\ln 2)\frac{E_T}{V^2} \end{aligned}$$

so if $V \approx \phi_T$,

$$\begin{aligned} C &\gtrsim 4(\ln 2)\frac{E_T}{\phi_T^2} \\ &= 4(\ln 2)\, C_T. \end{aligned}$$

For example, at $T = 300$ K, and $V \approx \phi_T$, we have $C \gtrsim 0.017$ fF.

In other words, the minimum average entropy generation per irreversible switching event *must* be

$$S \geq 2 \text{ bits}$$

if reliable erasure of a random bit is to be possible.

More generally, consider nodes in any constant-voltage switching technology in which short-circuit energy and leakage energy are negligible, so that the $\frac{1}{2}CV^2$ switching energy is the only dissipation. Suppose a node is to hold a logic `1` in exactly one randomly-selected case out of $N$ cases (instances distributed in space or in time), and is `0` in the rest of the cases, and we wish that, with high probability, the logic value should become `0` after the switching operation in *every case*, given that the node becomes tied to a constant voltage source whose level represents `0`. The entropy generated is

$$S \geq \ln N \text{ nats,} \tag{7.2}$$

but energy is only dissipated in the single case where the bit is `1`. In order for the environment to hold the increased entropy, the energy dissipated during switching in this case must be $E \geq ST \geq k_{\mathrm{B}}T \ln N$. In other words, node capacitance $C$ and swing voltage $V$ must be such that

$$\frac{1}{2}CV^2 \geq E_T \ln N \tag{7.3}$$

in order for a node to switch correctly in every one of $N$ instances (with high probability). Another way to write this formula is

$$\frac{1}{2}\left(\frac{C}{C_T}\right)\left(\frac{V}{\phi_T}\right)^2 \geq \ln N,$$

which shows explicitly the relationship between node capacitance, thermal capacitance, node voltage, and thermal voltage. If we take $V \approx \phi_T$, we get a lower bound on node capacitance of

$$C \gtrsim 2(\ln N)\, C_T.$$

So, for example, to achieve a nice $N$ value of $10^{27}$, corresponding to a computer with $10^9$ circuit nodes not making a single thermal error in 1 Gs ($\sim$32 years) of operation at a clock speed of 1 GHz, we must have a capacitance *per circuit node* of $C \gtrsim 0.77$ fF, if the operating voltage is almost as low as the room-temperature thermal voltage. Present-day gate capacitances of minimum-sized transistors are already near these levels. If some circuit nodes actually have smaller capacitances than this, supply voltages cannot actually approach the thermal voltage without sacrificing reliability to some extent.

It is interesting that we are able to derive the above relationship between reliability and node capacitance solely on the basis of the entropy generation and the switching energy. This can be compared with the traditional approach of developing a sophisticated thermal noise model, which finds that voltage samples follow a normal distribution, with a mean squared error (on a capacitance-$C$ node) of $\Delta V^2 = k_{\mathrm{B}}T/C$ ([129], §1.12, p. 31), and therefore to sample $N$ instances to an accuracy of $k_{\mathrm{B}}T$ with high reliability, $C$ must scale up with $\ln N$ roughly as we have described, or else there will be a significant chance that a thermal fluctuation will, in one of the $N$ instances, place the voltage sample enough standard deviations out on the tail of the distribution to cause a sampling error, and thus qualitatively incorrect functionality of the logic.

#### 7.1.2.1 Voltage bounds.

To express the above voltage bound quantitatively, we can rewrite eq. (7.3) as a lower bound on the swing voltage $V$ in terms of the node capacitance and thermal constants:

$$V \geq \sqrt{\frac{2E_T \ln N}{C}}$$

or equivalently

$$V \geq \phi_T \sqrt{2 \ln N \frac{C_T}{C}}.$$

Thus, as capacitances decrease towards and below the thermal capacitance $C_T$, minimum node swing voltages must increase to larger and larger multiples of the thermal voltage, proportional to the square root of the capacitance decrease, in order to maintain a given level of reliability. This is an important lower bound on operating voltage which must be taken into account when considering the minimum energy dissipation of irreversible switching circuits.

Of course, continually increasing voltages in order to shrink circuits is unrealistic since high electric fields will cause gate oxide breakdown. So in the long term, as nodes shrink in all dimensions proportionally to some characteristic feature size $\ell$, and node capacitances and voltages decrease proportionally, the above analysis really demands that we must eventually start scaling switching energy down as $CV^2 \sim \ell^3$ (which is intuitive for another reason, namely that otherwise, assuming switching frequency does not decrease, the power dissipation density in the channel would increase indefinitely, which would eventually cause damage such as melting and loss of structure) and the only way to reduce the switching energy in an irreversible circuit while still maintaining thermal reliability is, by the above analysis, to scale the temperature down with $\ell^3$ as well. So as trends in irreversible circuits continue, active cooling to cryogenic temperatures will eventually become a necessity in order to maintain good reliability.

Another lower bound on operating voltage for CMOS circuits comes from the fact that the device thresholds $V_{\mathrm{Tn}}$ and $V_{\mathrm{Tp}}$ cannot be set with complete precision, due to the statistical nature of existing dopant implantation processes. If $V_{\mathrm{dd}}$ is not significantly larger than the variability $\sigma_{\mathrm{VT}}$ of the $V_{\mathrm{T}}$ values, then some devices may fail to switch on and off as desired, and functionality may be compromised.

Our lower bounds on operating voltage may be summed up as follows:

$$
\begin{aligned}
&V \gtrsim \phi_T && \text{(to switch FETs strongly on and off)}\\
&V \geq \phi_T\sqrt{2\ln N\frac{C_T}{C}} && \text{(for reliability despite thermal noise)}\\
&V \gg \sigma_{\mathrm{VT}} && \text{(to avoid defects due to threshold variation).}
\end{aligned}
$$

Actually operating at the minimum voltage that is permitted by the above requirements may or may not maximize cost-efficiency, depending on the particular measure of cost that we are trying to minimize. Let us now see how to choose the operating voltage of irreversible static CMOS circuits so as to maximize efficiency under a variety of cost measures.

To sum up our discussion of entropy generation in irreversible CMOS, we saw that one should arrange that $\frac{1}{2}CV^2$ for each switching event is as small as possible, while remaining above $E_T \ln N$ (see eq. 7.3). Lowering the operating temperature helps decrease entropy production if it allows $V$ to decrease. If $\frac{1}{2}CV^2 = 2E_T \ln N$, then entropy generation must be at least $\ln N$ nats per switching event. This is $\alpha N \ln N$ nats for an error-free computation composed of $N$ primitive operations, if a fraction $\alpha$ of the operations cause switching.

Of course, other technological considerations may prevent $\frac{1}{2}CV^2$ from coming anywhere close to room-temperature $E_T$. The minimum load capacitance is determined by factors such as the minimum transistor gate area, and $V$ is independently bounded below by the variability $\sigma_{\mathrm{VT}}$ of device threshold voltages due to the statistical nature of ion implantation in the channel region. An interesting property of any given CMOS fabrication process is the ration between the minimal $\frac{1}{2}CV^2$ and $E_{T=300\mathrm{K}}$ in properly-functioning logic circuits in that process. In §7.1.3 we estimate this quantity for several present and projected future generations of CMOS technology.

### 7.1.3 The SIA semiconductor roadmap

Table 7.1 shows some parameters for future generations of CMOS VLSI technology as forecasted by the Semiconductor Industry Association, in [145]. We will be referring to these numbers for our calculations throughout the rest of this section.

Given these numbers, we can take a stab at an actual numeric computation of the minimum entropy generation per switching event. See table 7.2. Let us explain these

| | Year of first product shipment | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1997 | 1999 | 2001 | 2003 | 2006 | 2009 | 2012 |
| **Overall characteristics:** | | | | | | | |
| Trans. density,[a] $10^6$ trans/cm$^2$ | 3.7 | 6.2 | 10 | 18 | 39 | 84 | 180 |
| Chip size,[b] mm$^2$ | 300 | 340 | 385 | 430 | 520 | 620 | 750 |
| Clock freq.[c], GHz | 0.75 | 1.25 | 1.5 | 2.1 | 3.5 | 6 | 10 |
| Supply voltage[d], V | 2.5–1.8 | 1.8–1.5 | 1.5–1.2 | 1.5–1.2 | 1.2–.9 | .9–.6 | .6–.5 |
| Max. power[e], W | 70 | 90 | 110 | 130 | 160 | 170 | 175 |
| **Technology requirements:** | | | | | | | |
| $\mu$P drawn channel length[f], nm | 200 | 140 | 120 | 100 | 70 | 50 | 35 |
| DRAM $\frac{1}{2}$-pitch[f], nm | 250 | 180 | 150 | 130 | 100 | 70 | 50 |
| $T_{\mathrm{ox}}$ equivalent[g], nm | 4–5 | 3–4 | 2–3 | 2–3 | 1.5–2 | $< 1.5$ | $< 1$ |
| $CV/I$ delay[g], ps | 16–17 | 12–13 | 10–12 | 9–10 | 7 | 4–5 | 3–4 |
| $V_{\mathrm{T}}$ $3\sigma$ varia.[g], $\pm$mV | 60 | 50 | 45 | 40 | 40 | 40 | 40 |
| Src./drn. junction depth,[g] nm | 50–100 | 36–72 | 30–60 | 26–52 | 20–40 | 15–30 | 10–20 |

Table 7.1: Selected numbers from the 1997 edition of the Semiconductor Industry Association's national semiconductor roadmap [145]. These numbers are used for the calculations in tables 7.2 and 7.4.

[a] Logic transistor density in a packed, high-volume, cost-performance microprocessor, including on-chip SRAM. From [145], p. 14.

[b] Size for a $\mu$processor, year 1, before subsequent shrinks; [145] p. 15.

[c] On-chip local clock frequency for high-performance chips, [145], p. 16.

[d] Minimum logic power supply voltage $V_{\mathrm{dd}}$, [145], p. 17.

[e] Maximum power consumption for a high-performance processor with heat sink, [145] p. 17.

[f] Minimum feature sizes, [145], pp. 14, 85.

[g] The last four lines in the table are all from [145], p. 46.

| | Year of first product shipment | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1997 | 1999 | 2001 | 2003 | 2006 | 2009 | 2012 |
| **Capacitance calculations:** | | | | | | | |
| Gate oxide thickness $\tilde{T}_{\rm ox}$, nm | 4.5 | 3.5 | 2.5 | 2.5 | 1.75 | 1.5 | 1 |
| Gate areal capac. $C_{\rm ox}$, fF/$\mu$m$^2$ | 7.81 | 10.0 | 14.1 | 14.1 | 20.1 | 23.4 | 35.1 |
| Min. gate cap. $C_{\rm gmin}$, aF | 312 | 197 | 202 | 141 | 98.4 | 58.6 | 43.0 |
| Est. min. load cap. $\tilde{C}_{\rm Lmin}$, fF | 5.00 | 3.15 | 3.24 | 2.25 | 1.57 | .937 | .689 |
| **Voltage calculations:** | | | | | | | |
| Transistors/die, $10^6$ | 11.1 | 21.1 | 38.5 | 77.4 | 203 | 521 | 1350 |
| $N$ trans./defect (90% yield), $10^9$ | .111 | .211 | .385 | .774 | 2.03 | 5.21 | 13.5 |
| Defect probability $p$, $10^{-9}$ | 9.01 | 4.74 | 2.60 | 1.29 | .493 | .192 | .0741 |
| Number $n$ of $\sigma_{\rm VT}$'s | 5.75 | 5.86 | 5.96 | 6.07 | 6.23 | 6.37 | 6.52 |
| Est. min. $V_{\rm dd}$: $\tilde{V}_{\rm min}$, mV | 230 | 195 | 179 | 162 | 166 | 170 | 174 |
| **Energy and entropy:** | | | | | | | |
| Switching en. $E_{\rm sw} = \frac{1}{2}CV^2$, aJ | 132 | 59.9 | 51.9 | 29.5 | 21.6 | 13.5 | 10.4 |
| Est. min. ent. $\tilde{S}_{\rm min}$, knat | 23.9 | 10.8 | 9.40 | 5.35 | 3.92 | 2.45 | 1.89 |
| Inefficiency factor | 385 | 174 | 151 | 86.0 | 63.0 | 39.4 | 30.4 |
| Min. perm. energy loss, aJ | .903 | .409 | .354 | .202 | .148 | .092 | .071 |

Table 7.2: Calculations of minimum capacitance, supply voltage, energy dissipation, and entropy generation for irreversible CMOS, based on the SIA roadmap data from table 7.1. The minimum entropy generation per switching operation that is required given a 90% die yield ranges from 24 kilonats to 1.9 knats, which is greater than that required for a thermal reliability of less than 1 error in $10^{27}$ switching operations, by factors ranging from 385 in current technology, to 30 in projected technology for the year 2012.

---

calculations.

First, we perform some calculations to work towards finding out the load capacitance of typical small but non-trivial logic nodes (*e.g.*, NAND gates with output fanning out to 4 similar NAND gates) in each technology generation. This is important because we observed earlier that the lower bound on entropy generation is affected by load capacitance.

**Estimated gate oxide thickness.** We derive an estimate $\tilde{T}_{\rm ox}$ for the gate oxide thickness in each technology generation by taking the average of the high and low values given by SIA, or the high value if no low value is given.

**Gate capacitance per unit area.** The dielectric constant of $SiO_2$ (see table 7.3) is $\approx 351$ fF/cm. If we divide this by $\tilde{T}_{\rm ox}$, we get per-area gate capacitances $C_{\rm ox}$ ranging

| Symbol | Approximate value | Meaning |
|---|---|---|
| $\epsilon_0$ | 8.85 aF/$\mu$m | Dielectric constant of vacuum |
| $\epsilon_{\mathrm{Si}}$ | $11.7\epsilon_0 \approx 105$ aF/$\mu$m | Dielectric constant of silicon |
| $\epsilon_{\mathrm{ox}}$ | $3.97\epsilon_0 \approx 35.1$ aF/$\mu$m | Dielectric constant of $SiO_2$ |

Table 7.3: Some important dielectric constants for semiconductor electronics calculations. Taken from the frontispieces of [84, 139].

from 7.81 fF/$\mu$m$^2$ up to 35.1 fF/$\mu$m$^2$ as the technology improves.

**Minimum gate capacitance.** The gate-to-channel capacitance $C_{\mathrm{gmin}}$ of a minimum-sized transistor can be calculated from $C_{\mathrm{ox}}$, SIA's minimum gate length, and a minimum width which is assumed to be equal to the minimum length; we find values ranging from 312 aF in 1997 to only 43 aF in 2012.

**Estimated load capacitance.** In a minimum-size static CMOS NAND gate adjusted so that the effective gain factor $k$ of pull-up and pull-down networks are equal to the minimal transistor gain factor in the worst case, n-FET and p-FETs will both be sized to about double the minimum width—the n-FETs because two of them are in series, the p-FETs because hole mobility is only about half of electron mobility. An input impinging on the NAND feeds to one transistor of each type, so the load placed on the input by the NAND is about 4 times the minimum gate capacitance, plus fringing capacitances which we will ignore. If the output of each NAND gate fans out to about 4 other NAND gates, then the total load capacitance on the NAND output is only about 16 times the minimum gate capacitance. There is also a contribution from wiring and from the source-drain regions of the gate generating the signal, but if wire lengths are kept short, this can be absorbed into the factor of 4 without decreasing fan-out very much. So based on this, we just multiply $C_{\mathrm{gmin}}$ by 16 to find an estimated load capacitance $\tilde{C}_{\mathrm{L}}$ ranging from about 5 femto-Farads in 1997, to 0.69 fF in 2012.

**Transistors per die.** From SIA's figures for transistor density and chip size, we can calculate the total number of transistors per chip. It ranges from 11 million up to 1.35 billion.

**Transistors per defect.** Suppose we require that the loss in die yield due to random threshold variations should be less than 10%. That is, less than one die in 10 should contain any transistor having a large enough error in its threshold value to cause the transistor not to be in the correct (on vs. off) state when required. Multiplying by the transistors per die gives us the number $N$ of transistors that need be produced on average before one is produced that has such a defect. That is, in a set of $N$ transistors, the expected number of defects should be 1. We might call this $N$ the *process reliability number*. Values range from $\frac{1}{10}$ billion to 13.5 billion, as the number of transistors per chip increases in the SIA projections.

**Defect probability.** The sum of expectation values over a set of independent events is additive. So if the expected number of defects in $N$ transistors is 1, and the statistical results of ion implantation in different transistors is independent (a plausible assumption), the expected number of defects in 1 transistor must be $1/N$. Thus the probability $p$ that a given transistor will have a defect must be $1/N$. The required defect probabilities thus range from $9 \times 10^{-9}$ to $74 \times 10^{-12}$ as the transistor count increases.

**Number of $\boldsymbol{\sigma}_{\mathrm{VT}}$s required.** Roughly speaking, an n-FET in static CMOS will cause incorrect functionality either if it does not turn on when $V_{\mathrm{GS}}$ is $V_{\mathrm{dd}}$, or if it does not turn off when $V_{\mathrm{GS}}$ is zero. Therefore variation in thresholds should leave the threshold within the range 0 V to $V_{\mathrm{dd}}$. To minimize $V_{\mathrm{dd}}$ for a given level of threshold variability while remaining within reliability constraints, the nominal $V_{\mathrm{Tn}}$ should be exactly halfway between 0 V and $V_{\mathrm{dd}}$, so that the transistor only fails if variation places the actual $V_{\mathrm{Tn}}$ far out on one of the tails of the threshold distribution. In this situation, if the total probability of $V_{\mathrm{Tn}}$ being out on the tail in both directions is $p$, then the probability for either side (too high or too low) is $p/2$, since these events are mutually exclusive. (Similarly for p-FETs.)

Given a probability of being out on one tail of at most $p/2$, we can compute a lower bound on how many $\sigma_{\mathrm{VT}}$s are required before we are far enough out from the mean $V_{\mathrm{T}}$ so that the total probability of being that far out is no more than $p/2$. In a normal distribution, the total probability $\mathfrak{R}(n)$ of being at least $n$ $\sigma$s away from the mean in a particular direction is bounded above as

$$\mathfrak{R}(n) \leq \frac{1}{n} \cdot \frac{1}{\sqrt{2\pi}} e^{-n^2/2}. \tag{7.4}$$

And in fact, in the limit of $n \to \infty$, the probability out on the tail approaches this value exactly (*cf.* Feller 1950 [58], ch. VII, p. 175, eq. (1.8).)

Therefore, to have $\mathfrak{R}(n) \leq p/2$, we only require that $n$ be greater than or equal to the value given by solving

$$2/p = \sqrt{2\pi} \cdot n e^{n^2/2}$$

for $n$, which for given $p$ we can easily do numerically by computer using Newton's method. For our $p$'s we find values of $n$ ranging from 5.75 to 6.52 $\sigma_{\mathrm{VT}}$s; this relatively narrow range is afforded by the roughly exponential decay of the tail of the normal distribution.

**Minimum $V_{dd}$.** Now we are finally in a position to actually calculate the minimum value of $V_{dd}$ for each technology generation. With $V_{dd} = 2|V_T|$, we must have $V_{dd} \geq 2n\sigma_{VT}$ in order for the total probability of an error-inducing threshold defect (either too high or too low) to be less than $p$. Given SIA's values for $3\sigma_{VT}$, this yields minimal $V_{dd}$ voltages ranging between 230 mV and 162 mV.

These values are all several times greater than the thermal voltages of 26–34 mV found at normal operating temperature, so transistors that are turned off will conduct several times less current than ones that are turned on (in the worst case, we estimate, by at least a factor of 3), meaning that correct functionality is maintained, and leakage does not contribute very much to energy dissipation in circuits that switch frequently. However, if much lower levels of threshold variability than those given in the SIA roadmap were to be attained, low-temperature operation would be required in order for transistors to be able to turn off sufficiently at the implied lower voltage levels; this would tend to increase entropy generation, and thus reduce the advantages of the lower energy dissipation.

**Minimum energy/switching event.** Given the minimum capacitance of a useful logic node and a minimum swing voltage, we are now in a position to calculate the minimum switching energy $E_{sw} = \frac{1}{2}CV^2$ for such a node.

**Minimum entropy/switching event.** The minimum attainable entropy is lower-bounded by the minimum energy divided by the maximum temperature. Raising the temperature has a variety of complicated effects on CMOS device behavior, so this bound will not be exact. But if we assume that operating temperatures can't be much higher than, say, 127°C (400 K) while preserving correct functionality, we can get a rough lower bound on entropy generation.

**Inefficiency factor.** Suppose we wish the probability of a thermally-induced error to be $10^{-27}$ (this would correspond to, for example, a billion logic nodes switching at 1 gigahertz with only 1 error expected per gigasecond (31 years) of operation). Then the number of nats of entropy generation per switching event only needs to be $\ln 10^{27} \approx 62.2$ in an ideal switching circuit. So our CMOS circuits are generating more entropy than the ideal switching circuits by factors ranging from 385 down to 30. So, thermal reliability does not become the limiting factor on voltages for the foreseeable future of irreversible CMOS, although if the technology could continue to be improved for several more generations beyond the 2012 technology, this might change.

This concludes our discussion of the minimum entropy generation per operation attainable in irreversible CMOS circuits. In summary, entropy generation per switching event is expected to be greater than 1.8 kilonats through at least the year 2012. Insofar as each logic gate operation involves about one switching event, this also corresponds roughly to the entropy generation per primitive operation.

Even if unforeseen technological breakthroughs were to undercut this limit, an entropy generation of at least *tens* of nats per operation is required in order for correct voltage-switching to occur with high reliability in *any* irreversible switching technology, due to the argument that led to eq. (7.2), p. 154.

### 7.1.4 Minimizing permanent energy dissipation in iCMOS

Note that the energy dissipated by a CMOS circuit internally is *not all lost*. If a circuit is maintained at a temperature $T_{\mathrm{H}}$ that is as high as allowed by reliability constraints, in order to minimize entropy production, then most of the energy dissipated internally can be recovered, by using the computer as the heat source for a Carnot-cycle heat engine (*cf.* [143], §19-6, p. 371–376) with a relatively low-temperature reservoir at temperature $T_{\mathrm{L}} \ll T_{\mathrm{H}}$. If the heat is *emitted* from the computer into the heat engine at temperature $T_{\mathrm{H}}$ (*i.e.* no temperature degradation during transport), then a fraction $(T_{\mathrm{H}} - T_{\mathrm{L}})/T_{\mathrm{H}}$ of this heat can be converted to work by the heat engine.

In other words, the total energy that is *really* lost when $S$ entropy is generated is only $ST_{\mathrm{L}}$. If the ∼2.73 K cosmic microwave background can be used as the low-temperature reservoir, then theoretically only $\sim 4 \times 10^{-23}$ J of energy need be permanently lost for each bit of entropy generated in the computer, even though a hundred times more energy than this is temporarily "dissipated" whenever circuit nodes switch. This shows that energy dissipated in a circuit does *not* correspond to a permanent loss of work. It is only the *entropy* generation of the computer that truly determines the ultimate energy cost.

Thus, true energy loss is really minimized by minimizing entropy generation. If a 2.73 K reservoir is available, the minimum energy loss for CMOS ranges from about 0.9 aJ down to .07 aJ, as shown in the last line of table 7.2.

Actually, in practice there will always be some temperature degradation during the transport of heat from the transistors to the heat engine input, due to the non-infinite "heat capacities" of materials. However, if this temperature degradation is small compared to $T_{\mathrm{H}} - T_{\mathrm{L}}$, then the above results will be approximately correct.

Also, in practice, the 3 K microwave background may not actually be readily available as a thermal reservoir. In this case, the ultimate reservoir would be the atmosphere instead, and $T_{\mathrm{L}}$ would be on the order of $300K$, and the minimum energies would be $\sim 100$ times higher.

### 7.1.5 Maximizing per-area processing rate for iCMOS

Consider a cost measure like in §6.2.2, consisting of the rental cost of the land area or floor space required for a computation, or in other words the surface area of the computer times the time taken by the computation. Suppose we wish to minimize that cost measure.

To do this, we would like to know how to maximize the rate of computation that can be achieved per unit of surface area. This rate is limited if there is an upper bound on the rate of entropy removal through the machine, and a lower bound on the entropy generated per operation.

To maximize the computation rate per unit of outer surface area in irreversible CMOS, one should just maximize the ratio between the maximum entropy flux $F_{\mathrm{S}}$ in the cooling system and the entropy generation per operation, then pack in enough layers of circuits below each unit of surface area so that entropy is being generated at a rate per unit area corresponding to $F_{\mathrm{S}}$.

One key parameter to be chosen is the circuit operating temperature $T$. A lower operating temperature means more entropy generation for a given energy dissipation, but also more entropy flux for a given heat flux; these factors cancel out, meaning the relevant quantity is the ratio between the maximum heat flux and the minimum energy dissipation.

However, lower temperature probably means a lower heat flux, since at least some components of heat flow will increase with increased temperature differences between inside and outside. So within the bounds set by the reliability requirements at a given operating voltage, the machine should be operated as hot as possible. This is also consistent with minimizing the total entropy production, given the fixed lower bound on energy dissipation in CMOS, and also with minimizing the total permanent energy loss.

From SIA's figures for chip size and power, we can compute what heat flux they are assuming. If we take this as our maximum heat flux, we can combine this with our calculated figures for the minimum energy dissipation from table 7.2 to find the maximum rate of operation per unit area. See table 7.4.

What is the minimum thickness, in layers and in meters, for an irreversible CMOS machine that achieves this maximum speed per unit area? To determine this, we need to know both the minimum area per logic gate, minimum circuit layer thickness, and the maximum speed of operation of individual logic gates when operating at the minimum voltage.

The minimum area per logic gate is easy to calculate from the SIA figures, and an advanced wafer-thinning or SOI (silicon-on-insulator) process might be expected to achieve a thickness per circuit layer as low as $\sim$10 $\mu$m or less.

However, accurately determining the maximum speed of operation per gate requires a more detailed analysis. We would like to know an effective resistance $R$ for our logic gates, when driven at our minimized power supply voltage, such that in a characteristic time $t_c \approx RC_{\mathrm{L}}$ the output node is charged most of the way to the desired voltage level.

| | Year of first product shipment | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1997 | 1999 | 2001 | 2003 | 2006 | 2009 | 2012 |
| Rate calculations | | | | | | | |
| Heat flux $F_{\mathrm{E}}$, W/cm$^2$ | 23.3 | 26.5 | 28.6 | 30.2 | 30.8 | 27.4 | 23.3 |
| Max rate $\mathcal{R}_{\mathrm{pA}}$, ops/ns-$\mu$m$^2$ | .882 | 2.20 | 2.76 | 5.13 | 7.09 | 10.1 | 11.2 |
| Min $A\mathcal{T}$/op, ns $\mu$m$^2$ | 1.13 | .454 | .362 | .195 | .141 | .0986 | .0891 |

Table 7.4: Calculations of heat flux, rate of operation per outer surface area, and minimum outer-area rental cost per operation for (layered) irreversible CMOS, based on the SIA roadmap data from table 7.1. We find that irreversible CMOS can at best perform only ∼1–10 operations per nanosecond per square micron of outer area, given the heat flux implied by SIA's figures. In other words, the area-time cost per operation in ns $\mu$m$^2$ units ranges from ∼1 to ∼0.1.

---

Unfortunately, determining such an $R$ is rather complex. First, the instantaneous resistance of each MOSFET is not constant during node charging; it varies as the drain voltage changes. In multi-input logic gates, some transistors will in general have varying source voltages as well. Moreover, the effective resistance of the active logic network (pull-up or pull-down) will generally be data-dependent. For example, the pull-up network of a NAND gate, which consists of two transistors in parallel, conducts best if both inputs are low, rather than just one being low. Finally, the instantaneous resistance depends on the supply voltage in a complex way that depends on threshold voltage and source voltage, and that in small devices is affected by a variety of difficult-to-model short-channel effects, such as channel-length modulation, velocity saturation, mobility degradation, and drain-induced barrier lowering (*cf.* [165] and §2.3 of [139]).

We carried through a rough computation by hand (with help from the `Emacs calc` tool) based on a model described in (Rabaey 1996, [139], §2.3, p. 54, eqs. (2.57)–(2.59)), and originally proposed by Toh *et al.* (1988, [165]), which incorporates threshold voltage, mobility degradation, and velocity saturation effects. At this point we are still not completely confident in the accuracy of that calculation, so we will not detail it here. But one tentative result is that when operating at the minimum supply voltages we derived earlier, we end up with a maximum operation frequency that is only slightly higher than those projected by SIA (it is within a factor of 2), although in our analysis, the maximum operation frequency increases less rapidly than in SIA's, as technology improves. The reason for the remaining discrepancy between our calculations and SIA's is unclear, and will probably remain so until we obtain a fuller description of SIA's assumptions and analysis methods.

Another tentative result of this analysis is that for maximizing total computation rate per unit area, our choice of lower supply voltage results in an overall speedup factor (compared to SIA's projections) that ranges from 127 down to 2.5 as technology improves; part of the reason why the improvement decreases seems to be that SIA's choice of supply voltage converges from 10 times ours down to only 3 times ours, as the generations progress.

Unfortunately, the low-voltage approach is not significantly more cost-effective (in terms of rate per unit of silicon surface) than SIA's. This is not surprising, since in these calculation we were not trying to maximize overall cost-effectiveness, but rather only performance per unit of outer area; we were ignoring the material cost of stacking up more layers of surface over that area. The number of circuit layers we need for optimal per-area performance ranges from 80 (in 1997) down to 4 (in 2012), and the cost of these extra layers roughly negates the benefit of the increased performance.

### 7.1.6 Maximizing iCMOS cost-efficiency

When maximizing cost-efficiency in terms of circuit mass or wafer area, rather than outermost-surface area, the analysis becomes more complex. For one thing, it depends on the nature of the computation to be performed.

If the computation consists of many small independent computations, requiring only local communication in 2-D, for example, then the circuitry can be spread out in a single layer, and the task is to maximize the rate per unit of silicon area. In this case, we are not entropy-limited, so we cannot assume, as we did in the previous section, that the optimal operating voltage is just the minimum voltage that is consistent with reliability constraints. Lower signal voltages in general increase the effective resistance of transistors, and lead to longer charging times $t \sim RC$. But the voltage cannot be too large either, or it will cause oxide breakdown and other undesirable effects, and it could possibly cause overheating even if the circuit is just a single layer. An accurate analysis would need to take these concerns into account, as well as all of the complex short-channel effects that arise when scaling to smaller device sizes. Optimizing voltage in the face of all these concerns cannot be done via solving a simple analytical equation; instead one must write a program to search for the optimum point numerically.

Alternatively, if the computation requires frequent communication, such as for example between neighboring cells in a 3-D mesh being simulated, then the analysis is made complex for other reasons, namely because we cannot spread everything out in a single layer without incurring communication delays, as we discussed in ch. 6, §6.2.3.1. If a near-ballistic means for communication is available—such as an optical or transmission-line system of interconnects between processors—then when entropy generation becomes the dominant concern, the optimal structure is the one from that earlier section, in which we lower the clock speed and spread the processing elements out in proportion to the cube root of the logical diameter of the communication network.

But in this case as well, carrying out the relevant calculations for future generations of semiconductor technology would be complex and error-prone, due especially to uncertainties in the technical specifications of the communication network.

In both cases (2-D and 3-D computations), we will deem a detailed numeric calculation of cost-efficiency to be beyond the scope of this thesis, and we will relegate it to the domain of future work. Still, such a calculation will be important eventually, if we wish to be able to compute the exact scale above which reversibility becomes advantageous, in each succeeding technology generation. In the current technology generation, a very rough hand-calculation suggests that even in the most optimistic scenario for reversible computing (namely, ballistic communication between nodes at logical distance $\sqrt{N_{\mathrm{D}}}$, see §6.2.3.2, p. 136), reversibility doesn't improve cost-efficiency until we reach a cost level on the order of \$25 billion. Later we will argue that over time, the case for reversibility ought to improve, for as long as the $RC$ of CMOS technology keeps improving; however, we will also argue that at some point, making further $RC$ improvements will require moving to a radically different technology, such as the superconducting Josephson-junction technology of Likharev [108]. In any case, making more accurate projections of the advantages of reversibility in foreseeable generations of CMOS technology would be very desirable in order to gauge the near-term applicability of this research.

This concludes our analysis of the best possible performance of normal *irreversible* CMOS circuits under various efficiency measures for the foreseeable future. This can serve as a baseline for comparison when looking at reversible circuits.

Now, let us see how reversible circuit techniques came about.

## 7.2 Historical development of adiabatic circuits

We now review the historical development of reversible logic circuit techniques.

**Correcting an attribution.** Toffoli (1980, [161]) suggested that the idea of dissipationless computing using reversible circuits originated with John von Neumann, but I have been unable to confirm this claim. To quote Toffoli's paper:

> The idea that universal computing capabilities could be obtained from reversible, dissipationless (and, of course, nonlinear) physical circuits apparently first occurred to von Neumann, as reported in a posthumous paper (Wigington 1961 [188]).

However, based on a careful reading of Wigington's paper, I believe that Toffoli's characterization of it is incorrect.

Wigington's paper [188] is an explanation of a patent [181] submitted by von Neumann in 1954 and posthumously granted to him in 1957. The paper describes a computing scheme in which logic values are represented by the relative phase of AC signals, rather than by the DC voltage level used in conventional systems. Wigington also cites similar work (on "Parametron" circuits) that apparently occurred around the same time in Japan.

The logic circuits that Wigington discusses do indeed include some elements (namely, nonlinear reactances) that can be assumed to have arbitrarily low dissipation (and whose operation is therefore physically reversible), but the described circuits also include attenuators that are placed in the signal paths *explicitly* in order to dissipate the energy of signals whose phase information is no longer needed, in exact analogy to the practice in traditional DC logic circuits of dumping a node's static energy through a dissipative switch whenever its logic value is no longer needed. Without these attenuators, von Neumann's circuits would not reliably implement the computing scheme described. In fact, the need for dissipation is guaranteed by the logic system used. The fundamental logical operation in von Neumann's AC circuits is to take three binary input signals, and from them generate a boolean output signal whose logic value is the majority value of the inputs. The input signals are consumed in the process, in the attenuators. Since the resulting operation is a many-to-one transformation of the logical state of the circuit, it destroys logical information, and thus *cannot* be implemented in a physically reversible way, by Landauer's argument (§2.5.1, p. 41). No matter how we try to modify von Neumann's circuit, we will fail to achieve dissipationless, reversible operation, so long as the logical state transformation operation that is performed is a many-to-one operation such as Wigington describes.

Therefore, Toffoli's characterization of the von Neumann/Wigington paper seems mistaken. Von Neumann had certainly had a great many original ideas during his lifetime, and it is conceivable that he thought about the idea of reversible, dissipationless computing, but if so, the Wigington paper certainly provides no evidence to support that attribution.

**Earliest adiabatic circuits.** To our knowledge, the first description of a logic circuit technique that seriously attempts to avoid the $\sim CV^2$ dissipation associated with conventional logic is that of Watkins 1967 [186]. Watkins describes a technique whereby capacitive loads in a circuit are charged gradually through the control transistors from a power supply whose voltage fluctuates cyclically according to prescribed waveforms. When the transistors in Watkins' circuit are first turned on, there is no voltage across them and thus no dissipation through them. While the load is being charged up through the transistor, there is a small voltage across it, and thus some dissipation, but this dissipation can be made as low as desired by just lengthening the time taken in the charging cycle. Watkins analyzes the energy dissipation in his circuit, and predicts that the energy per cycle asymptotically approaches zero as the cycle time is increased.

Unfortunately, Watkins' energy analysis appears not to have been quite correct, due to his use of semiconductor diodes to discharge the nodes in his circuit. Such diodes have an intrinsic voltage drop $\phi_T$ across them (*cf.*[94] §2 and [139] §2.2.1, p. 20) which does not decrease as the circuit is run more slowly. Therefore, Watkins' circuits still incur a minimum dissipation of $\sim CV\phi_T$ per operation, no matter how slowly they are run. Therefore, they do not qualify as truly *time-proportionately reversible* (see §6.1.3, p. 123) logic devices that could improve asymptotic cost-efficiency as per the arguments in chapter 6. In any case, for whatever reason, Watkins' proposal appears to have faded into obscurity.[1]

**Inductor-based approaches.** After Watkins, we next see the idea of dissipationless electronic logic crop up independently in a 1978 proposal by Fredkin and Toffoli [73]. In their idea, energy is shuttled around between inductors and capacitors, but is not dissipated substantially, in a circuit that implements a purely reversible primitive operation, namely a 3-input 3-output Fredkin gate. The assertion is that in such a circuit, energy dissipation can be made arbitrarily small if only the quality factor $Q$ of the $LC$ elements can be made arbitrarily large. Unfortunately, the Fredkin-Toffoli approach was not immediately practical, because it appeared to require large numbers of inductors, roughly one for each logic element, whereas VLSI fabrication technology does not well support high-quality integrated inductances.

However, in 1985 the ball was again picked up by Charles Seitz and colleagues [144] at Caltech, who (apparently independently) describe a logic technique similar to Fredkin and Toffoli's, but in which the inductances are instead shared between many logic circuits, and are brought off-chip, and can therefore be implemented using a technology that is more optimized for providing high-$Q$ resonance than is VLSI. The relatively complex switching circuitry that controls the logical operation of the circuit remains integrated on-chip. This was a key step on the road to making resonant circuits practical. However, Seitz *et al.* only worked out the technique in detail for relatively simple circuits; they leave open the question of whether a general logic family could be worked out to implement *any* combinational or sequential logic with arbitrarily little dissipation. Their proposed solution is, in their own words, not "foolproof" and requires careful tuning of the circuit parameters to ensure correct and dissipationless operation.

[1] According to *Science Citation Index*, Watkins' article [186] was only cited a total of four times through 1976, and these citations appear to all be from general review articles, rather than applications of Watkins' research. After 1976, Watkins was not cited at all until Bill Athas and colleagues rediscovered his work in 1997 [4].

**Improvement of adiabatic techniques.** In 1992 and 1993, a sequence of several important developments proceeded to solve the remaining difficulties with adiabatic switching. First, Koller and Athas [92] devised a simple and general adiabatic logic family, but encountered difficulties because their circuits were not fully logically reversible; Koller and Athas noted that whenever their circuits finally needed to forget some information, they were unable to avoid $\sim CV^2$ dissipation, because ultimately, the only way to clear the logical state of a circuit node whose state is unknown is to tie that node to a reference reservoir at a known voltage level, thereby dissipating the energy of the node (if different from the target level). Being unaware of Fredkin and Toffoli's earlier work showing that sequential circuits need never discard information, Koller and Athas did not know any way around this problem, and went so far as to conjecture (incorrectly) that any sequential logic circuits (*i.e.*, circuits containing feedback loops) would require dissipation.

In the meantime, some fully reversible circuit techniques were discovered independently by nanotechnology enthusiasts Hall [79, 80] and Merkle [122, 123, 125]. However, Merkle did not discuss how to implement sequential circuits, and Hall's technique was essentially non-sequential, and thus algorithmically inefficient. It required saving all intermediate results in hardware, using as many clock rails as there were stages in the computation, then reversing the whole process to recover energy before beginning the next sequential stage. In between stages, an irreversible write of the results of the previous stage was required.

**Fully adiabatic CMOS techniques.** The first adiabatic circuit technique to put together all the key elements needed for fully universal adiabatic computing was the CRL ("Charge Recovery Logic") technique of Younis and Knight, developed in our research group in 1993 [192]. Like the original Fredkin-Toffoli technique, CRL could implement arbitrary sequential logic. Like Seitz's technique, CRL took advantage of an off-chip resonant element. Like the Koller-Athas technique, it did not require fine tuning of the circuit elements. Putting together these three elements, the Younis-Knight technique provided the first practically implementable circuit style capable of reliable, asymptotically reversible operation.

The initial version of CRL was somewhat baroque, but it was later refined to another version (called SCRL, for "split-level CRL") that was relatively clean and simple [193, 191, 89]. SCRL was used as the basis for all of the reversible circuit design work that we will describe later in this chapter. In section 7.5, we review SCRL in detail.

**Recent adiabatic circuits research.** Following 1993, there has been a small explosion of literature on and relating to adiabatic circuits of various types. The literature has become too large to review in detail here, but for bibliographical completeness, we include a sampling of some relevant citations: [5, 44, 93, 83, 154, 153, 94, 157, 171, 6, 4].

**Patents on adiabatic circuits** Additionally, a search for recent patents relating to adiabatic and reversible computing turned up the following patents from 1995 through 1997: [30, 48, 61, 62, 63, 89, 125, 131, 142, 190, 159]. Most of these patents were assigned to large organizations such as IBM, MIT, AT&T, Motorola, and Xerox.

## 7.3 A comment on terminology

We'd like to pause briefly here for a comment on the term "adiabatic circuits" itself. Originally in thermodynamics, the word *adiabatic* is an adjective literally meaning "without flow of heat" into or out of the system (*cf.* [143], §18-5, p. 352). So, for example, one way to adiabatically compress a gas would be to compress it inside an well-insulated chamber so that the heat produced cannot escape. Such a compression can be thermodynamically reversible: the gas can be allowed to adiabatically re-expand, pushing back against the piston that compressed it while cooling, and the work that was originally applied to compress the gas can be recovered.

As a result of its frequent usage in such contexts, the term "adiabatic" in applied physics has gradually evolved to the point where it is frequently used to refer not to the lack of heat flow precisely, but rather to the overall *thermodynamic reversibility* (or near-reversibility) of a process. Any process that is thermodynamically reversible (at least in the low-speed limit) has come to acquire the moniker "adiabatic."

Note that this new usage is completely orthogonal to the literal meaning of adiabatic, "no heat flow." A process can involve no heat flow into or out of the system, yet be thermodynamically irreversible: for example, when a partition is removed to allow two different gases originally separated in different chambers to mix together. Conversely, a process can involve heat flow, yet be reversible: for example, if the heat is contained within an insulated box which is physically moved via a reversible mechanism out of the region of space identified as "the system."

A more accurate term for thermodynamically reversible processes might be *isentropic* (literally, "with the same entropy"), since thermodynamically reversible processes are, by definition, those processes that generate no new entropy. However, even this term is not precisely applicable to the circuits referred to as "adiabatic circuits," because the circuits are not *perfectly* isentropic except in the limit of zero clock frequency and low temperature (to stem the flow of leakage currents). The phrase "*asymptotically* isentropic" would therefore be a bit more accurate.

However, some of the circuits that have been referred to as "adiabatic" are not even *asymptotically* isentropic, due to the use of diodes with a built-in voltage drop.

Essentially, the term "adiabatic circuit" is so ill-conceived, and so polluted with inaccurate and inconsistent usage that we wish that it could be dropped altogether. To avoid confusion, we would like to advocate the adoption of the following alternative, more accurate lexicon:

- *energy recovery circuits* (ER circuits) — These are circuits that are designed to recover a substantial portion (but not necessarily all) of the energy invested in logic signals (*e.g.* $CV^2$ static energy). This could include diode approaches.
- *asymptotically isentropic circuits* (AI circuits) — These are ER circuits that, in some appropriate limit (such as low speed and/or low temperature) can generate asymptotically zero entropy per operation. Example: SCRL.
- *time-proportionally reversible circuits* (TPR circuits) — AI circuits in which entropy generation per operation is approximately inversely proportional to the length of time over which operations are performed. Example: SCRL when operated in a regime where leakage currents are small.
- *ballistic circuits* — Hypothetical TPR circuits in which the entropy coefficient is so low that the entropy generation per operation is zero for all practical purposes, even when the circuit is running at its maximum rate. Superconducting technologies would probably be required for this.

However, abolishing an established bit of terminology is, in general, a difficult thing; it confuses people who are accustomed to the old terminology, and complicates keyword searches for material on a given topic. Therefore, despite our academic objections, we bend to popular usage and continue to use the term "adiabatic" when we are referring generally to circuits of any of the above types. When we wish to be more precise than this, we will use one of our more precise terms.

## 7.4 Basic principles of adiabatic circuits

The core insight behind all adiabatic circuits is that the $\sim CV^2$ minimum dissipation in ordinary switching circuits is due primarily to the fact that such circuits charge a node by connecting it to a *constant voltage* power supply (*cf.* the discussion in §7.1.1.1, p. 150).

**Constant current sources.** One alternative means that one might think of for charging up a capacitive load is to instead use a *constant current* power supply, operating at some appropriate current over some desired length of time. See figure 7.3. One may assume that the charging pathway has some effective resistance $R$.

We can analyze the dissipation in this case as follows. Let $C$ be the load capacitance, $V$ the voltage swing, $R$ the resistance in the charging pathway, and $t$ the time

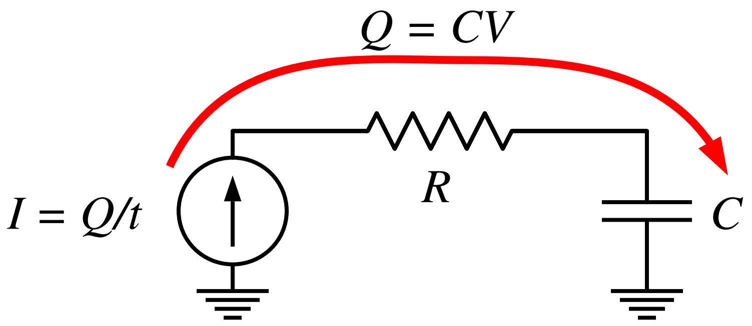



Figure 7.3: Charging a node to voltage $V$ via a constant current over a time $t$. Compare with fig. 7.2. In this case, the dissipation is not $\frac{1}{2}CV^2$, but rather $CV^2\,\frac{RC}{t}$, which becomes arbitrarily small as the charging time $t$ is increased.

---

we wish to take to charge the node. Then the charge delivered is $Q = CV$, the current should be $I = Q/t$, and the energy dissipated in the circuit is

$$\begin{aligned} E_{\text{diss}} &= \int p\,d\tau = \int i^2 R\,d\tau = I^2 Rt = \left(\frac{Q}{t}\right)^2 Rt \\ &= \frac{Q^2}{t}R = \frac{(CV)^2}{t}R = \boxed{CV^2\,\frac{RC}{t}}. \end{aligned}$$

Note that this dissipation scales down proportionally as the charging time increases. Therefore this particular charging process is an example of what we call a *time-proportionately reversible* process.

**Voltage ramps.** Can this constant-current procedure be used when charging nodes in a CMOS logic circuit? Well, it can at least be closely approximated, by using a turned-on transistor in place of the resistor, and using a power supply with a linear voltage ramp in place of the constant current source. (See figure 7.4.) An exact analysis of the energy dissipation in this circuit is more complex, but it can be shown to approach that of the constant-current circuit very closely when $t \gg RC$. At the opposite extreme, when $t \ll RC$, the dissipation approaches that of an ordinary constant-supply-voltage switching circuit as in §7.1.1.1. See fig. 7.5, and see Younis [191] for a more detailed discussion.

The same basic technique can also be used to discharge a logic node, with the supply voltage ramping the other way, from the logic level $V$ back down to 0.

Note however that these low-dissipation characteristics are only maintained as long as we charge and discharge all nodes *only* using this technique. If, on the other hand we ever turn on a transistor when there is a voltage difference across it, there

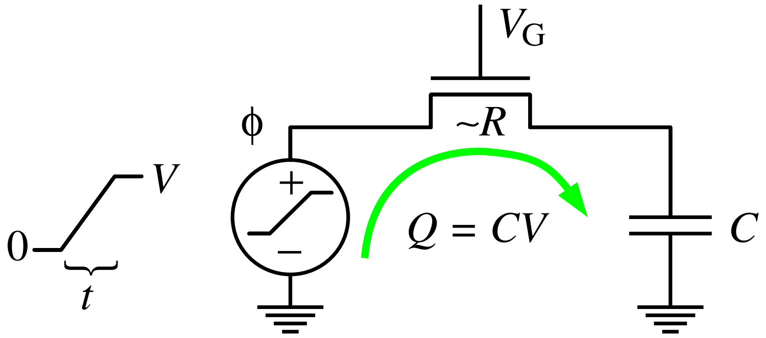



Figure 7.4: Compare with figure 7.3. Instead of an ideal current source, we have power supply that provides a voltage signal $\phi$ that ramps up from 0 to $V$ over a time $t$. Instead of an ideal resistor, we have a turned-on CMOS transistor, with gate voltage biased at some value $V_{\mathrm{G}} > V + V_{\mathrm{T}}$ that allows the transistor to conduct well over the entire voltage range from 0 to $V$, with an approximate effective resistance of $R$. For $t \gg RC$, $E_{\mathrm{diss}} \approx CV^2 \frac{RC}{t}$; for $t \ll RC$, $E_{\mathrm{diss}} \approx \frac{1}{2}CV^2$.

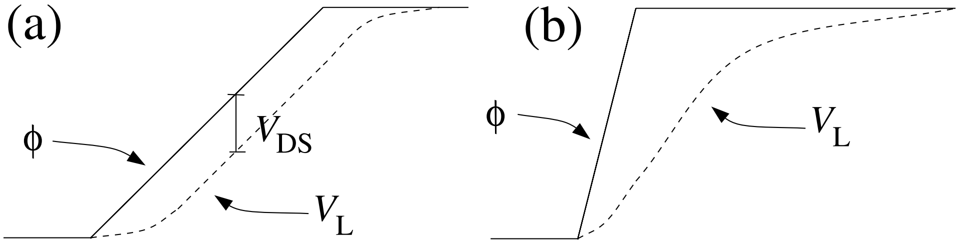



Figure 7.5: Voltage curves for slow and fast adiabatic charging using a voltage ramp. If the supply signal $\phi$ ramps up much faster than the characteristic $RC$ time of the circuit, then the load voltage $V_{\mathrm{L}}$ will lag behind the ramp and approach the supply level in a characteristic exponential-decay curve. When $\phi$ reaches its peak, the voltage difference across the transistor is still almost the full swing $V$, leading to the dissipation being almost $\frac{1}{2}CV^2$.

On the other hand, if $\phi$ ramps up very slowly, $V_{\mathrm{L}}$ will track it, with only a small lag $V_{\mathrm{DS}}$ that is determined by the rise time, the transistor's characteristic transconductance $k$, and the drive voltage $V_{\mathrm{dr}} = V_{\mathrm{G}} - V_{\mathrm{T}}$.

will be a $\frac{1}{2}CV^2$ dissipation, as in any switching circuit. So in a fully adiabatic logic circuit, we need the rule that a transistor can only be turned on if is no voltage difference across it. In truly asymptotically isentropic circuits, this constraint leads to the consequence that logical information cannot be thrown away—since essentially, in these circuits, the only operation that throws away information is to dissipatively connect a node to another one at a different voltage level.

One might object that in this technique we are merely moving the dissipation from inside the circuit to the power supply which must generate this swinging logic signal. But, as we will see later, there are a number of ways to generate the necessary signal using a *resonant* element, in which circuit energy oscillates back and forth between the on-chip capacitance and an off-chip inductance. If the resonant frequency is low and the off-chip elements have a high quality index, the off-chip elements need not dissipate significant energy either.

The above discussion outlines the basic principles of adiabatic circuits, but does not get into the details of how build complex logic circuits using those principles. There are now a number of different adiabatic logic techniques available for doing this. In the next section we review our technique of choice: SCRL.

## 7.5 The SCRL technique

As we said earlier, SCRL was the first adiabatic circuit technique to simultaneously be capable of (1) asymptotically approaching true zero energy per operation, (2) being integrated on a large scale using standard CMOS process technology, and (3) operating in pipelined, sequential fashion. In this section, we review in detail the operation of SCRL, and show some new graphical depictions that we find helpful for understanding its structure and operation.

### 7.5.1 Basic SCRL components

We start by reviewing the basic elements of which SCRL circuits are composed.

Note: The following description breaks down SCRL circuits into functional elements in a slightly different way than in the original work of Younis and Knight [193, 191]. We find our alternative decomposition a bit simpler to explain.

In our version, SCRL circuits are separated into two types of components: (1) clock-driven generalized inverters, and (2) bidirectional latches.

#### 7.5.1.1 SCRL clocked generalized inverters

A clocked SCRL inverter is illustrated in figure 7.6. It is very simple, composed merely of two MOSFET transistors, one n-type and one p-type. In fact, it has exactly the same internal structure as an inverter in ordinary static CMOS (see fig. 7.1 in §7.1.1), but it is wired and used slightly differently. Rather than being connected to constant-voltage supply rails, the FETs are connected to a swinging, clocked power supply. The n-FET is connected to a clock-supply signal $\overline{\phi}$ that swings between 0 and $V_{\rm dd}/2$—that is, half of the full signal voltage—with a particular waveform. The p-FET is connected to a clock $\phi$ that swings between $V_{\rm dd}/2$ and $V_{\rm dd}$. We assume $V_{\rm Tn} \approx V_{\rm Tp}$, and the voltage swing $V_{\rm dd}$ itself is chosen to be more than twice $V_{\rm T}$, so that the transistors will conduct bi-directionally through the entire swing range of the clocks to which they are respectively connected.

The operation of the device is as follows. Initially, all circuit nodes are at the "neutral" level $V_{\rm dd}/2$, representing "no information." Being enhancement-mode devices, both transistors are nonconducting at this time. Then, the input voltage $V_{\rm in}$ swings to a level 0 or $V_{\rm dd}$, representing binary 0 or 1, as in conventional logic. At this point, one of the two transistors becomes conducting (n-FET on input 1, p-FET on input 0), but no current flows because $\phi = V_{\rm out} = \overline{\phi}$.

Next, the rails $\phi$ and $\overline{\phi}$ swing simultaneously, in a roughly linear ramp taking some non-infinitesimal rise time $t_{\rm r}$, to their respective extremes. (They are said to "split,", thus the S in SCRL.) The output level $V_{\rm out}$ will track the rail to which it is connected. At this point the output is considered valid, and its value can, for example, be sampled by a latch (which we will get to in a moment) for later use. In the meantime, $V_{\rm in}$ must remain (roughly) constant at its 0 or 1 logic level—this is crucial for preventing dissipation.

At some point after the subsequent stages have finished using the $V_{\rm out}$ signal, the supply rails are brought back together to $V_{\rm dd}/2$. Again, $V_{\rm out}$ tracks the rail to which it has remained connected this whole time. Then, the input $V_{\rm in}$ is free to return to the neutral level, turning off both transistors again.

Of course, as in ordinary static CMOS, this SCRL inverter structure can be generalized to compute any $n$-input inverting logic function, such as NAND or NOR, by simply replacing the p-FET and n-FET with complementary networks of p-FETs and n-FETs, respectively. (See figure 7.7.) The operation of such gates is essentially the same as that of the simple inverter. All internal nodes are initially at $V_{\rm dd}/2$, and when the rails are split, all nodes that are connected to one or the other rail track it, at least up to a threshold drop away from the extreme point. (No node can be connected to both rails if the networks are properly complementary.) In general, all $n$ inputs must be held constant during the entire time that the rails are non-neutral. Then, when the rails re-merge, all nodes that were pulled away from the neutral level

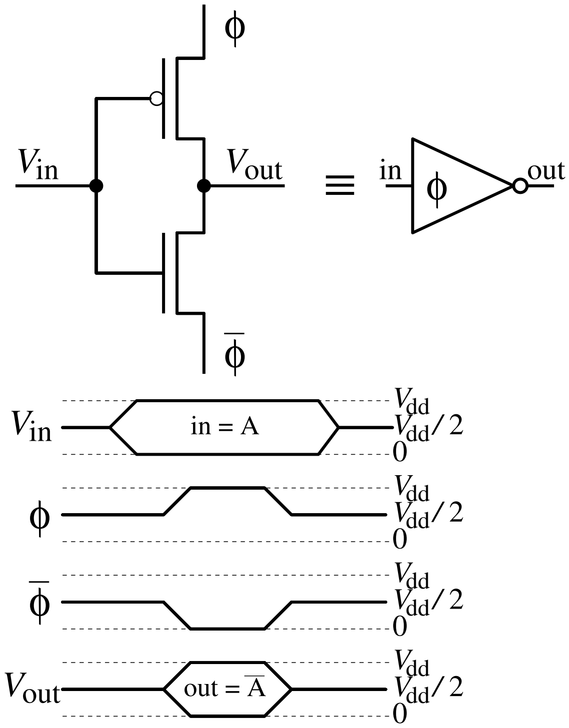



Figure 7.6: An SCRL inverter. It has the same structure as an ordinary static CMOS inverter (fig. 7.1a), except that the supply rails are tied to swinging clock signals rather than to constant-voltage supplies, and there are various additional assumptions and constraints on circuit operation. In between cycles, all nodes are at $V_{\text{dd}}/2$. The input $V_{\text{in}}$ is assumed to swing to either 0 or $V_{\text{dd}}$ (representing a logic 0 or 1 as usual) during some period, and for a shorter period enclosed within this, the clock rails $\phi$ and its logical inverse $\overline{\phi}$ swing at a constant rate from $V_{\text{dd}}/2$ to $V_{\text{dd}}$ and 0, respectively, remain in this state for a time, and then return smoothly to the neutral level $V_{\text{dd}}/2$. (See the timing diagram in the bottom half of the figure.) During this time, the output is valid and can be sampled by a latch (fig. 7.8) to be used in further stages of processing. Note that the input must not change while there is a voltage across either transistor: this is the key property that avoids $CV^2$ dissipation. On the right is an icon convenient for representing this element.

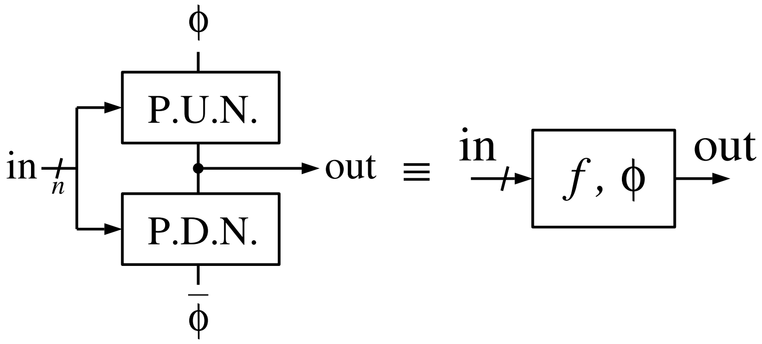



Figure 7.7: Generalized SCRL inverter. As in ordinary static CMOS, the SCRL inverter can be generalized to compute any inverting logic function $f$ (e.g., NAND, NOR) of $n$ inputs by simply replacing the single p-FET with an arbitrary pull-up network of p-FETs, and the n-FET with the complementary network of n-FETs. As in ordinary CMOS, it is best not to make the logic gates have *too* many inputs, due to the impact on conductance if there are many transistors in series in the pullup/pulldown network. Lower conductance decreases speed in static CMOS, and in SCRL, it also increases the entropy coefficient and the minimum energy dissipation. To imitate the conductance of an inverter, the transistors in a multi-input gate can be made wider, but this of course consumes more wafer surface area.

are gradually pulled back.

A warning: Associated with internal nodes in the pullup/pulldown networks of a generalized SCRL inverter, there may be a component of dissipation that does not scale down with frequency. Fortunately, as we will see in §7.6.4, this problem is easy to fix.

#### 7.5.1.2 SCRL bidirectional latches

Given only generalized SCRL inverters, and no other components, one could potentially proceed to create combinational logic of any desired depth, by using a different pair of clock signals for each level of logic, and having the clocks for earlier stages split before the clocks for the later stages do, and re-merge after the clocks for the later stages do. This would be similar to the "retractile cascade" approach of Hall [79, 80]. But the problems with this simple approach are that (1) it would need as many pairs of clock rails as there are stages in the logic, and (2) the earlier stages must remain idle while the later stages are computing, and therefore there can be no pipelining, and no sequential circuits with feedback.

To solve these problems, SCRL introduces an additional component in between logic stages, something we call a "bidirectional latch" (fig. 7.8). Through a "forward"

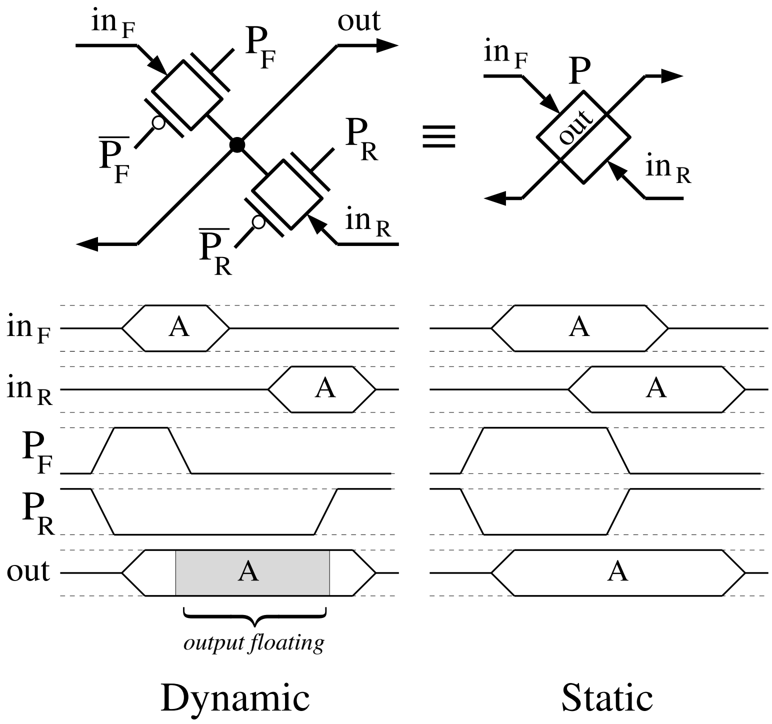



Figure 7.8: SCRL bidirectional latch. This special circuit element, composed of 2 CMOS pass gates with appropriately clocked controls, is essential for being able to pipeline SCRL logic stages and to make arbitrary sequential (as opposed to combinational) circuits of any depth using only a constant number of clock phases. An icon for the element is shown on the upper right.

Initially, nodes $in_{\mathrm{F}}$, $in_{\mathrm{R}}$, and *out* are all neutral. The "forward" F pass gate turns on, and the "reverse" R pass gate turns off. Input signal $in_{\mathrm{F}}$ goes valid with a logic value A, driving the output line through gate F. Before the input signal goes neutral, gate F closes, so that the output signal will continue to remain valid after the input goes neutral. Meanwhile, a later stage of the computation is reconstructing the value A, and presents it again on input $in_{\mathrm{R}}$. After this happens, gate R opens, tying the output node to $in_{\mathrm{R}}$ which is at the same level, so there is no dissipation. Then $in_{\mathrm{R}}$ goes neutral gradually, drawing *out* back to the neutral level. The latch is now ready to process another input on $in_{\mathrm{F}}$.

Depending on the relative timing and the presence/absence of overlap of the $in_{\mathrm{F}}$ and $in_{\mathrm{R}}$ signals, the latch may operate in either a dynamic mode (bottom left), or in a static mode (bottom right).

pass gate "F," the latch is adiabatically drawn from its initial neutral level to the logic level $\mathrm{in}_{\mathrm{F}}$ produced by the preceding logic stage. The pass gate shuts off, and then the latch holds its value, allowing the preceding stage to reset and prepare to accept a new input, while in the meantime the succeeding logic stage uses the value held on the latch as its own input for further computations. But after the succeeding logic stage finishes, there is a small problem: How do we clear the latch to accept a new input from the preceding stage? We cannot just dump the latch to a constant voltage because that would be irreversible and dissipative. Instead, the latch must be discharged adiabatically by a controlling component that knows what level to discharge it from. The *preceding* logic stage no longer knows what value the latch is holding, because it has already gone on to reset itself and process new data (allowing this was the whole point of the latch). However, the key insight is that the *succeeding* logic stage now contains information that depends on the latched value. If that succeeding logic stage has computed some *invertible* function of its inputs, then the value in the latch can be reconstructed based on the information that the succeeding stage has calculated, and using this knowledge, the latch can be reversibly cleared.

To provide this adiabatic "unwriting" functionality, the latch provides a second write port, in the form of a second "reverse" pass gate "R". Logic in the succeeding stage presents a reconstructed copy of the latched value on input $\mathrm{in}_{\mathrm{R}}$, then the reverse pass gate opens, and is drawn back to the neutral level through pass gate R. Also, the latch provides a second "read port" in the form of a wire of the output node leading back to the preceding stage, which uses this input to reconstruct and clear the values stored in the preceding level of latches.

Two different alternative timing disciplines for these latches are shown in the bottom half of fig. 7.8. Note that the pass gates are turned off and on adiabatically by gradually-swinging ramps, and that they are never turned on when there is a voltage across them.

### 7.5.2 SCRL pipelines

Putting it all together, figure 7.9 shows the structure of a complete SCRL pipeline. The arrows in this figure represent many parallel wires, each function block represents a parallel set of logic gates all using the same clock, and each bidirectional-latch icon represents a parallel set of bidirectional latches, all on the same clock. The direction of the arrows shows the direction of information flow. As you can see, each set of "forward" logic gates that computes a logic function is paired with a corresponding set of gates, pointing the other way, which is used to uncompute the latched values from the previous logic stage. Each forward or reverse stage, and each latch, operates on a different set of clock signals. However, after some small number of stages, the earlier clock signals may be used again. This allows arbitrary sequential circuits with feedback loops (such as CPUs) to be constructed.

Normally, each logic stage can only compute an inverting function, and so there is a potential difficulty that if one initially has a value but not its complement, one cannot, in a single later stage, have access both to the value and its complement. This difficulty can be fixed by having a 2-level retractile cascade within each stage of logic, as illustrated in the bottom part of fig. 7.9. An alternative way to fix the problem would be to maintain a dual-rail signaling discipline, with complements of every logic value always available, but this would in general require more area.

### 7.5.3 Timing disciplines

There are a variety of alternative timing disciplines in SCRL. These vary in terms of the number of clock phases, and whether they are dynamic or fully static. The simplest fully-static discipline is 3-phase; the timing diagram for this is illustrated in fig. 7.10. If one wishes to permit dynamic operation (floating nodes), 2 phases will suffice.

We used 3-phase SCRL in our designs, because when running at very slow clock speeds, dynamic circuits would have been vulnerable to incorrect functionality, because (at normal temperatures) the charge stored capacitively on a dynamic node may leak out over long periods. With fully static circuits, we could be more confident that functionality would remain correct even when running at the very slow speeds that minimize energy dissipation, speeds at which the switching currents become nearly as small as the leakage currents.

For more detailed descriptions of the various timing disciplines see Younis 1994 [191].

## 7.6 SCRL circuit analyses

In this section, we carry through a variety of CMOS circuit analyses in order to better understand interesting aspects of SCRL's scaling behavior. First, §7.6.1 presents a simplified model of SCRL that we will use in our analyses. Then in §7.6.2, we derive an expression for the switching energy dissipation in SCRL in any given technology, in terms of raw characteristics of the technology, such as the threshold voltages and transconductance parameters of its transistors. In §7.6.3, we extend this by taking leakage currents into account, and derive analytical expressions showing how to adjust speed and threshold voltage to minimize total dissipation in SCRL at a given temperature. Those analyses are based on a fairly simple model of CMOS transistors, which becomes somewhat inaccurate in very small devices.

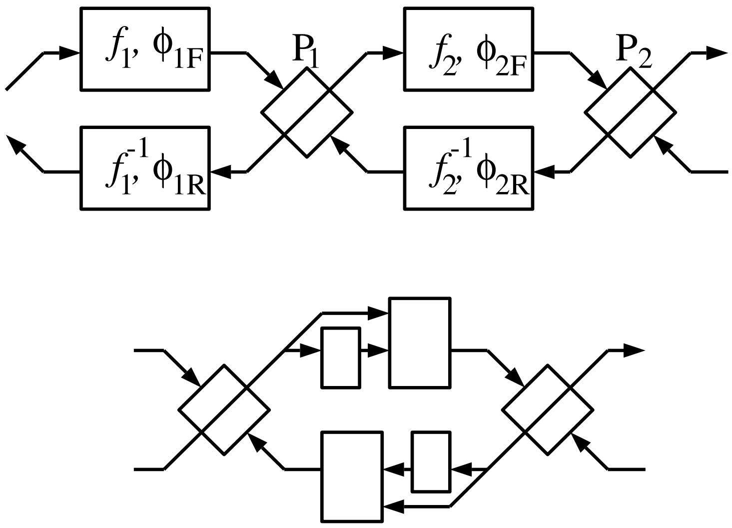



Figure 7.9: SCRL pipeline. A pipeline of arbitrary sequential logic in SCRL can be constructed by chaining together generalized inverters and bidirectional latching in the following way. A parallel set of generalized inverters is grouped together into a multi-input, multi-output function $f_1$ which must be *invertible*, and this block is paired with a corresponding block that computes the inverse function $f^{-1}$. The two blocks are powered by two clock phases $\phi_{1F}$ and $\phi_{1R}$ that are offset relative to each other. Then comes a bidirectional latch $P_1$ and another pair of functional blocks.

The basic operational cycle is that $f_1$ computes and its output $X_1$ is latched onto $P_1$. Then $f_2$ computes and its output $X_2$ is latched onto $P_2$, then $f_2^{-1}$ computes $X_1$ from $X_2$, and "unlatches" $P_1$ back to the neutral level. Now $f_1$ can process a new input and store the result $Y_1$ on $P_1$, at the same time that a further stage $f_3$ is using the value computed from the earlier value $X_1$. In this way, waves of information propagate down the pipeline as they are being processed. The pipeline can even loop back on itself, as long as phases are matched properly. If the clock timing is inverted, information flows in the opposite direction.

The bottom part of the figure illustrates how non-inverting logic functions can be computed in a single SCRL stage by the use of a second intermediate level of logic. The second level uses a clock whose active period is enclosed within that of the first level's clock, like a 2-level version of one of Hall's retractile cascades. With 2 levels per stage in an SCRL pipeline, one can do universal reversible sequential logic.

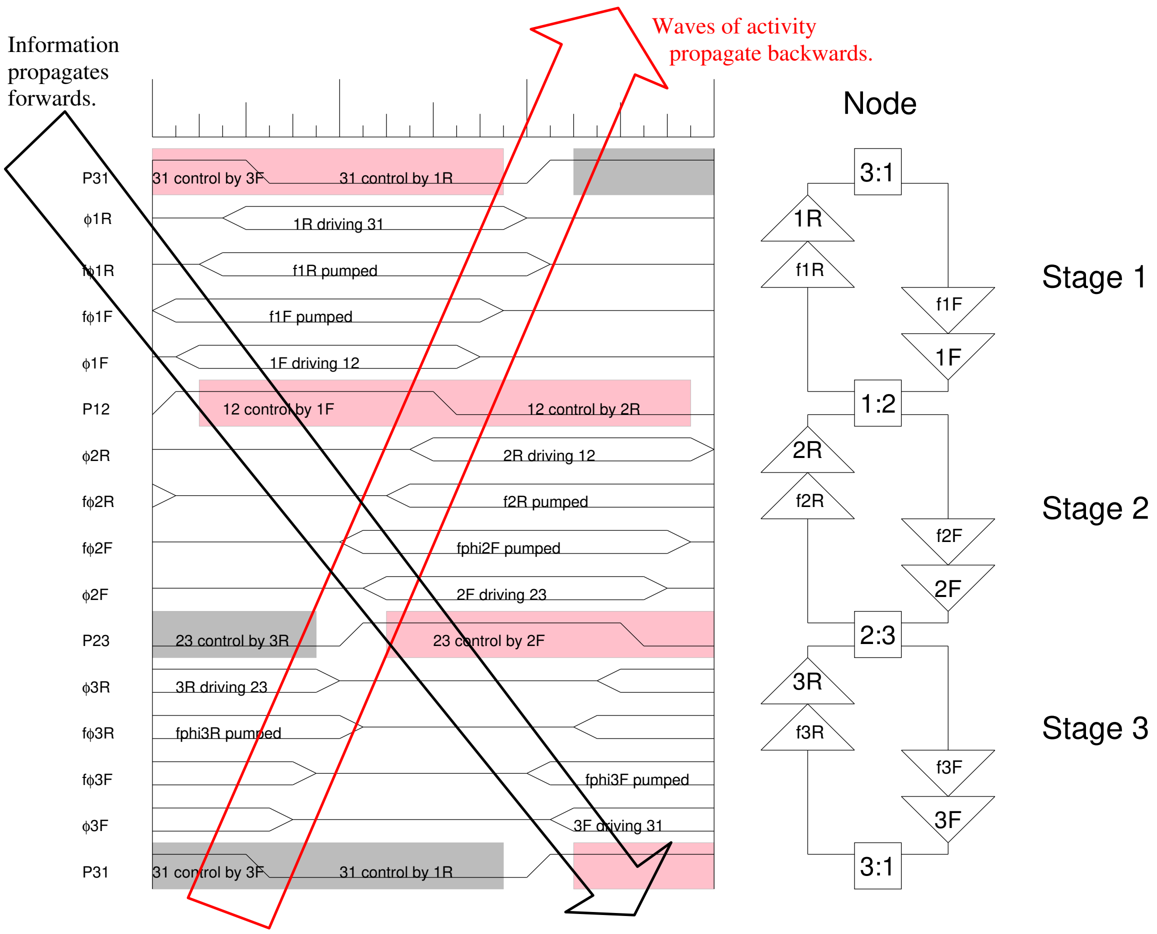



Figure 7.10: Full timing diagram for 3-phase, non-inverting, static SCRL. This was the timing discipline used in our designs. On the right is a vertical representation of a sequence of 3 pipeline stages, using a slightly different notation from that presented in figs. 7.6–7.8: the squares are bidirectional latches, and the triangles are generalized SCRL inverters. Each element's clock is shown exactly to its left on the timing diagram. Time goes from left to right on the timing diagram, and information flows from top to bottom in the pipeline. This scheme requires 16 clock signals and their inverses, and 24 distinct non-overlapping transition steps per complete clock period. The shaded regions indicate times when valid logic values are present on the various latches.

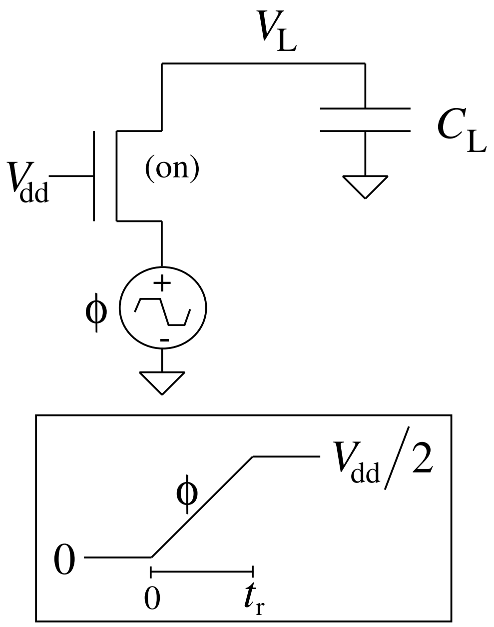



Figure 7.11: Simplified circuit model for SCRL analysis. Compare with figs. 7.6 (p. 176) through 7.8 (p. 178). In the simple model, we only look at a single transition, and we use a single transistor to represent the path through an arbitrary pulldown network and a transmission gate in a latch.

Following this, section 7.6.4 reveals a case in ordinary SCRL where the dissipation seems to be larger than in the ideal model, and shows a way to fix it. Then later, in §7.9, we will talk about some of the long-term limits involved in scaling CMOS and SCRL technology to smaller length scales.

### 7.6.1 A simple SCRL model for analysis

Switching energy is dissipated in an SCRL circuit whenever the voltages on some logic gate's power supply rails $\phi, \overline{\phi}$ change. Energy is dissipated within the transistors of the gate's pullup/pulldown networks, and also in the transistors of the transmission gate in the bidirectional latch attached to the gate's output. However, to simplify the analysis, we will lump together all the turned-on transistors within which dissipation occurs during a transition, and treat them as if they were a single transistor, as in figure 7.11.

We can consider a number of different cases for switching. A gate's output node voltage may be switched either through the gate's pulldown network of n-FETs or through its pullup network of p-FETs. And the switching activity may either be to clear the output or to set the output. When an output node is cleared, its voltage goes from a valid level (0 or $V_{\rm dd}$) to the neutral value $V_{\rm dd}/2$; when it is set, its value goes from $V_{\rm dd}/2$ to 0 or $V_{\rm dd}$.

However, all these cases are symmetrically similar to each other with regards to how their energy dissipation scales with speed, threshold voltage, and temperature. Therefore, rather than analyzing them all separately, we will just consider one case: where the voltage $V_{\rm L}$ on the load capacitance $C_{\rm L}$ on the output node is charged up from 0V to $V_{\rm dd}/2$, through a turned-on n-FET which represents the gate's pull-down network and N pass transistor.

In our analysis, we will ignore any dissipation that occurs during switching in transistors along paths that do not actually connect all the way through to the gate's output. For example, we ignore energy dissipation that occurs when switching with the transmission gate turned off. As another example, referring forward to the NAND gate in figure 7.15 (p. 200), we can see that if input A is high and input B is low, then transistor T3 will be turned on, and so there will be some dissipation through it, even though it does not connect through to the output. Ignoring such dissipations is a simplification that is fairly well justified, because these dissipations involve driving relatively small capacitances compared to the external load. In adiabatic charging, there is a quadratic dependence of dissipation on the capacitance being driven. So the total dissipation we are ignoring should not be large, compared to the dissipation that we are including.

We assume that the p-FETs and n-FETs in the SCRL circuit have been sized so that their gain factors are equal, $k_{\rm n} = k_{\rm p} = k$ (matching the rise/fall delay times), and we assume that the p-FET and n-FET threshold voltages are also equal, $V_{\rm t0n} = V_{\rm t0p} = V_{\rm t0}$, so that the analysis of the dissipation through the pulldown network comes out the same for the pullup network.

### 7.6.2 Switching losses as a function of technology parameters

To determine the energy dissipation of our model circuit (fig. 7.11), we would like to know the voltage on the load at each moment during the transition, $V_{\rm L}(t)$, because this would tell us the instantaneous drain-to-source voltage $V_{\rm DS}(t)$ across the transistor, which we could plug into the device's current-voltage relation to give us the instantaneous current $I(t)$, and thence the instantaneous power, which we could integrate over time to find the total energy dissipation of the transition $E_{\rm tr}$:

$$E_{\rm tr} = \int_{t=0}^{\infty} P(t)dt \tag{7.5}$$

$$= \int_{t=0}^{\infty} I(t)V_{\rm DS}(t)dt \tag{7.6}$$

Unfortunately, $V_{\rm L}(t)$ itself is determined by integrating the current $I(t)$ flowing into the load capacitance $C_{\rm L}$, so that determining closed-form formulas for $I(t)$ and $V_{\rm DS}(t)$ requires solving a tricky differential equation, which we will not attempt here. Instead, we will approximate the energy dissipation by treating the limiting case where the supply rise time $t_{\rm r}$ is very large compared to the characteristic $RC$ time constant of the circuit, where $R$ is the effective resistance of the turned-on transistor. Cases where the rise time is comparable to $RC$ will not be adequately addressed by the below analysis.

To understand this limiting case, refer back to the diagrams in figure 7.5 (p. 173). Diagram (a) shows qualitatively what would happen if the supply rail were to rise very quickly compared to $RC$. Essentially the output voltage would rise at an exponentially-decaying rate and asymptotically approach the supply voltage, just as happens in a regular CMOS inverter whose input switches very quickly. The energy dissipation $E_{\rm fast}$ for this fast-switching case is well known to be, as in §7.1.1.1, p. 150,

$$E_{\rm fast} = \frac{1}{2} C_{\rm L} (\Delta V)^2, \tag{7.7}$$

which in our case is (with $\Delta V = V_{\rm dd}/2$)

$$E_{\rm fast} = \frac{1}{8} C_{\rm L} V_{\rm dd}^2. \tag{7.8}$$

On the other hand, figure 7.5(b) shows what happens in the case which we will now analyze, where the supply rail rises very slowly. The output voltage $V_{\rm L}$ will initially rise slowly, but as the voltage drop $V_{\rm DS}$ across the transistor increases, the current $I(t)$ through the transistor will also rise, until an equilibrium is reached at which point $V_{\rm L}$ is rising at the same rate as the input voltage, but lagging behind it by a small amount $V_{\rm DS} = IR$. Then, when the input voltage stops rising, the output voltage will finish the approach to $V_{\rm dd}/2$ in asymptotic fashion, with an $RC$ time constant.

We note that if the input rises slowly, $V_{\rm DS}$ is always small compared to $V_{\rm dd}/2$, and so $V_{\rm L}(t) \approx \phi(t)$. During the transition, $d\phi/dt$ is constant, and so the current $I = C_{\rm L} \frac{dV_{\rm L}}{dt}$ through the transistor will be approximately constant as well. $I$ will be the quotient of the total charge $Q = C_{\rm L} V_{\rm dd}/2$ that is transfered to the load capacitance, divided by the supply rail rise time $t_{\rm r}$, since that is the time during which almost all of this charge is transfered.

$$I(t) \approx I = Q/t_{\rm r} = \frac{C_{\rm L} V_{\rm dd}/2}{t_{\rm r}} \tag{7.9}$$

Now, armed with this constant current $I$, we can use the standard MOSFET triode-regime current-voltage formula (*cf.* [139], §2.3.2, p. 44, eq. 2.47) to derive a closed form expression for $V_{\rm DS}$. The reason we use the triode-regime rather than the saturation-regime formula is that turned-on transistors in SCRL are never in saturation.[2]

In the following, $V_{\rm GS}$ is the gate-to-source voltage, and $V_{\rm T}$ the threshold voltage. Everything except $k$ (the transistor's gain factor) is here implicitly a function of $t$.

$$I = k\left((V_{\rm GS} - V_{\rm T})V_{\rm DS} - \frac{V_{\rm DS}^2}{2}\right) \tag{7.10}$$

Let's write $V_{\rm GS} - V_{\rm T}$ as $V_{\rm dr}$ (drive voltage) for conciseness.

$$I = k\left(V_{\rm dr}V_{\rm DS} - \frac{V_{\rm DS}^2}{2}\right) \tag{7.11}$$

We can easily solve this equation for $V_{\rm DS}$, using the quadratic formula.

$$\frac{I}{k} = V_{\rm dr}V_{\rm DS} - \frac{V_{\rm DS}^2}{2} \tag{7.12}$$

$$\frac{1}{2}V_{\rm DS}^2 - V_{\rm dr}V_{\rm DS} + \frac{I}{k} = 0 \tag{7.13}$$

$$V_{\rm DS} = \frac{V_{\rm dr} \pm \sqrt{(-V_{\rm dr})^2 - 4\left(\frac{1}{2}\right)\left(\frac{I}{k}\right)}}{2\left(\frac{1}{2}\right)} \tag{7.14}$$

$$= V_{\rm dr} - \sqrt{V_{\rm dr}^2 - 2\frac{I}{k}} \tag{7.15}$$

Now, let us make a further simplification of eq. 7.15. We observe that our earlier approximation, that $I(t)$ was constant, assumed that $t_{\rm r}$ is large, and therefore that $I$ is small (from eq. 7.9). With $I \ll kV_{\rm dr}^2$, this will allow us to approximate eq. 7.15 as follows. We observe that $V_{\rm DS}$ will be approximately linear in $I$ for these small $I$s.

[2]This formula may not be appropriate for turned-on transistors if $V_{\rm dd}$ is about as small as the thermal voltage $\phi_T = k_{\rm B}T/q_{\rm e}$, since then even turned-on transistors may only be in moderate or weak inversion, and the current may scale exponentially with $V_{\rm GS}$ rather than according to the triode formula. This is one area where the present analysis needs refinement.

$V_{\mathrm{DS}}$ will pass through 0 at $I = 0$, and the slope is given by $dV_{\mathrm{DS}}/dI$:

$$\frac{dV_{\mathrm{DS}}}{dI} = \frac{d}{dI}\left(V_{\mathrm{dr}} - \sqrt{V_{\mathrm{dr}}^2 - 2\frac{I}{k}}\right) \tag{7.16}$$

$$= -\frac{1}{2}\left(V_{\mathrm{dr}}^2 - 2\frac{I}{k}\right)^{-\frac{1}{2}}\left(\frac{-2}{k}\right) \tag{7.17}$$

$$= \frac{1}{k\sqrt{V_{\mathrm{dr}}^2 - 2\frac{I}{k}}} \tag{7.18}$$

$$\approx \frac{1}{k\sqrt{V_{\mathrm{dr}}^2}} \quad \text{(for small } I\text{)} \tag{7.19}$$

$$= \frac{1}{kV_{\mathrm{dr}}}. \tag{7.20}$$

Given this slope, and the fact that $V_{\mathrm{DS}} = 0$ when $I = 0$, we can therefore simplify eq. 7.15 to the very concise form

$$V_{\mathrm{DS}} \approx I/kV_{\mathrm{dr}}. \tag{7.21}$$

Now, the drive voltage $V_{\mathrm{dr}}$ is itself actually time-dependent, because it is defined in terms of the gate-to-source voltage $V_{\mathrm{GS}}$, and although the gate voltage is constant, the transistor source voltage changes linearly over time $t_{\mathrm{r}}$, from 0 to $V_{\mathrm{dd}}/2$, following $\phi(t)$.

$$V_{\mathrm{dr}}(t) \equiv V_{\mathrm{GS}}(t) - V_{\mathrm{T}}(t) \tag{7.22}$$

$$= [V_{\mathrm{G}} - V_{\mathrm{S}}(t)] - V_{\mathrm{T}}(t) \tag{7.23}$$

$$= \left(V_{\mathrm{dd}} - \frac{V_{\mathrm{dd}}}{2}\frac{t}{t_{\mathrm{r}}}\right) - V_{\mathrm{T}}(t) \tag{7.24}$$

$$= (V_{\mathrm{dd}} - V_{\mathrm{T}}(t)) - \frac{V_{\mathrm{dd}}}{2}\frac{t}{t_{\mathrm{r}}} \tag{7.25}$$

Moreover, $V_{\mathrm{T}}(t)$ as well will vary along with the supply voltage, due to the changing body effect as the source voltage changes. For example, when the supply voltage is at $V_{\mathrm{dd}}/2$, $V_{\mathrm{T}}$ might be perhaps (as a roughly estimated typical value) 50% above the minimum value $V_{\mathrm{T0}}$ that it has when $\phi = 0V$. Using the correct formulas for $V_{\mathrm{GS}}$ and $V_{\mathrm{T}}$, the energy integral in equation 7.6 would still a bit too complicated to conveniently evaluate, although if we really cared to do it, we could.

But instead, let's just make the rough approximation that $V_{\mathrm{dr}}(t)$ is constant, and is equal to

$$V_{\mathrm{dr}} = \frac{3}{4}V_{\mathrm{dd}} - b_{\mathrm{avg}}V_{\mathrm{T0}}, \tag{7.26}$$

taking the average of the initial ($V_{\mathrm{dd}}$) and final ($V_{\mathrm{dd}}/2$) values of $V_{\mathrm{GS}}(t)$, with an average body-effect factor $b_{\mathrm{avg}} = V_{\mathrm{T}}/V_{\mathrm{T0}}$ for a typical body-effected $V_{\mathrm{T}}$. The reason for expressing the body-effected threshold $V_{\mathrm{T}}$ as a multiple of $V_{\mathrm{T0}}$ is that it will later allow us to derive a very simple expression for the switching energy.

Now, with our approximate constant expressions for $I$ (eq. 7.9) and $V_{\mathrm{dr}}$ (eq. 7.26), we can consider $V_{\mathrm{DS}}$ as given by eq. 7.21 to be roughly constant, which allows us finally to approximate the transition energy integral (eq. 7.6) and derive a fairly simple expression for $E_{\mathrm{tr}}$ in the slow-transition limiting case. We set the upper bound on the integral to be time $t_{\mathrm{r}}$ rather than $\infty$, in observance of the fact that in the slow-transition limit, most of the energy dissipation occurs by time $t_{\mathrm{r}}$.

$$\begin{aligned}
E_{\mathrm{tr}} &\approx \int_{t=0}^{t_{\mathrm{r}}} I(t)V_{\mathrm{DS}}(t)dt && (7.27)\\
&\approx IV_{\mathrm{DS}}t_{\mathrm{r}} && (7.28)\\
&= I\left(\frac{I}{kV_{\mathrm{dr}}}\right)t_{\mathrm{r}} && (7.29)\\
&= \frac{I^2t_{\mathrm{r}}}{kV_{\mathrm{dr}}} && (7.30)\\
&= \frac{\left(\frac{C_{\mathrm{L}}V_{\mathrm{dd}}/2}{t_{\mathrm{r}}}\right)^2 t_{\mathrm{r}}}{kV_{\mathrm{dr}}} && (7.31)\\
&= \frac{C_{\mathrm{L}}^2V_{\mathrm{dd}}^2}{4t_{\mathrm{r}}kV_{\mathrm{dr}}} && (7.32)
\end{aligned}$$

Now, we would like to take another simplifying step, by assuming that our maximum power supply voltage $V_{\mathrm{dd}}$ is being scaled proportionately to $V_{\mathrm{T0}}$, and is equal to

$$V_{\mathrm{dd}} = n_{\mathrm{dd}}V_{\mathrm{T0}} \tag{7.33}$$

where $n_{\mathrm{dd}}$ indicates the scaling factor used for determining $V_{\mathrm{dd}}/V_{\mathrm{T0}}$. SCRL will not work properly if $V_{\mathrm{dd}}$ is too close to the threshold voltage $V_{\mathrm{T0}}$. A reasonable value for $n_{\mathrm{dd}}$ for SCRL might be 4. Anyway, given eqs. 7.33 and 7.26, we can substitute $V_{\mathrm{dd}}$ and $V_{\mathrm{dr}}$ in eq. 7.32 to re-express it in terms of a single voltage parameter $V_{\mathrm{T0}}$, the zero-bias threshold voltage:

$$E_{\mathrm{tr}} = \frac{C_{\mathrm{L}}^2 (n_{\mathrm{dd}} V_{\mathrm{T0}})^2}{4 t_{\mathrm{r}} k \left(\frac{3}{4} n_{\mathrm{dd}} V_{\mathrm{T0}} - b_{\mathrm{avg}} V_{\mathrm{T0}}\right)} \tag{7.34}$$

$$= \frac{C_{\mathrm{L}}^2 n_{\mathrm{dd}}^2 V_{\mathrm{T0}}^2}{4 t_{\mathrm{r}} k \left(\frac{3}{4} n_{\mathrm{dd}} - b_{\mathrm{avg}}\right) V_{\mathrm{T0}}} \tag{7.35}$$

$$= \left(\frac{n_{\mathrm{dd}}^2}{3 n_{\mathrm{dd}} - 4 b_{\mathrm{avg}}}\right) \frac{C_{\mathrm{L}}^2 V_{\mathrm{T0}}}{t_{\mathrm{r}} k}, \tag{7.36}$$

and let us finally just make this a bit more concise by renaming the factor containing $n_{\mathrm{dd}}$ as just

$$c_{\mathrm{dd}} \equiv n_{\mathrm{dd}}^2 / (3 n_{\mathrm{dd}} - 4 b_{\mathrm{avg}}). \tag{7.37}$$

To illustrate what a typical value of $c_{\mathrm{dd}}$ might be, if $n_{\mathrm{dd}} = 4$ and $b_{\mathrm{avg}} = 1.25$ (*i.e.*, average body-effected threshold 25% above $V_{\mathrm{T0}}$), then $c_{\mathrm{dd}} \approx 1.45$.

Anyway, we can now write the transition energy formula (7.36) as just

$$\boxed{E_{\mathrm{tr}} = c_{\mathrm{dd}} \frac{C_{\mathrm{L}}^2 V_{\mathrm{T0}}}{t_{\mathrm{r}} k}.} \tag{7.38}$$

There are a couple of very interesting things to note about equation 7.38, when compared to equations like eq. 7.7 that govern the dissipation in fast SCRL transitions or ordinary CMOS transitions.

The first thing is that the transition energy in eq. 7.38 scales in proportion to the square of the load capacitance, in contrast to traditional CMOS where the $CV^2$ dissipation scales only linearly with capacitance. The reason is that higher capacitance means higher currents through our transistors, and thus a larger voltage drop across them, in addition to greater charge to move across that drop. So in designing SCRL circuits we must be even more careful to get load capacitances small than we are in regular CMOS. Unless most of the capacitance is in the interconnects, minimum-sized transistors are favored. If most capacitance is in transistor gates and PN junctions, then increasing transistor widths increases energy dissipation roughly linearly (not quadratically, because $k$ is scaled too). The flip side of this coin is that SCRL benefits greatly from improved process technologies that allow smaller, less capacitive transistors.

The other very interesting point is that given a constant $n_{\mathrm{dd}}$ ratio between supply and threshold voltages, and everything else but $V_{\mathrm{T0}}$ also constant, the switching energy of SCRL circuits *decreases only linearly with decreasing threshold voltage*, in contrast to the quadratic drop of traditional CMOS due to its $CV^2$ switching energy.

Intuitively, the reason is because as voltages go down in SCRL, the effective on-resistance of our transistors increases, so the voltage drop across the transistors during transitions is increased, causing higher dissipation. In standard CMOS, the voltage drop across the transistors during switching is already as high as possible, and so making them more resistive doesn't affect the dissipation at all.

Equation 7.38 is interesting and useful on its own, because it allows us to predict the switching energy of SCRL circuits constructed in particular process technologies, and helps guide us in designing these circuits. But now, let's go a little further, and use eq. 7.38 as part of a more sophisticated analysis of SCRL energy dissipation that includes the effects of leakage.

## 7.6.3 Minimizing the sum of switching and leakage energy

In this section we explore how to minimize the energy dissipation of SCRL when taking leakage into account. First, in §7.6.3.1 we see how to minimize energy dissipation when the speed of operation is adjustable but all other technology parameters are held fixed. Then, in §7.6.3.2 we will see how to minimize dissipation when the choice of device threshold voltage (and supply voltage) is also adjustable, but other parameters such as device geometry and operating temperature are fixed.

### 7.6.3.1 Adjusting speed to minimize dissipation

One often-cited characteristic of the switching energy of adiabatic circuits, based on equations like eq. 7.38, is that it decreases linearly with increasing transition time $t_{\mathrm{r}}$, leading to the conclusion that the energy per operation of SCRL circuits can be made arbitrarily small by just making the transition time larger. However, given current device technologies, this statement is somewhat misleading, because MOS transistors also have a leakage power dissipation that is always present, and thus contributes a term to total energy per operation that increases linearly with increasing time per operation. This means that there is some speed at which the energy per operation of an SCRL circuit is minimized; at faster speeds, the switching energy dominates, and at lower speeds, the leakage energy dominates. In this section we derive a formula for the optimal rise time for minimizing total energy per operation.

Let us consider what happens to a signal wire in an SCRL circuit during a complete cycle, from the time it first holds one valid value to the time it first holds the next. During this time there will be two complete transitions on the wire: one from the old value to $V_{\mathrm{dd}}/2$, the other from $V_{\mathrm{dd}}/2$ to the new value. The total time for the complete cycle depends on the number of phases in the particular SCRL clocking discipline in question. A complete cycle of the 2-phase SCRL described by Younis [191] is the length of 18 transitions; 3-phase and 4-phase SCRL take 24 transitions (*cf.* fig. 7.10, p. 182), etc. These numbers are probably not minimal. Anyway, let $n_\mathrm{t}$ be the number of transitions per cycle; the total cycle time is then $T = n_\mathrm{t} t_\mathrm{r}$.

Now we can write down an expression for the total energy dissipation associated with this signal wire per complete cycle, including terms for both the transition energy and the leakage energy, where the leakage energy is expressed in terms of $P_\mathrm{leak}$, the average leakage power associated with the signal wire:

$$\begin{aligned} E_\mathrm{tot} &= 2E_\mathrm{tr} + P_\mathrm{leak} T && (7.39)\\ &= 2c_\mathrm{dd}\frac{C_\mathrm{L}^2 V_\mathrm{T0}}{t_\mathrm{r} k} + P_\mathrm{leak} n_\mathrm{t} t_\mathrm{r}. && (7.40)\end{aligned}$$

where the multiplication by 2 comes from the above-mentioned fact that an SCRL wire undergoes two transitions per cycle.

We want to find the $t_\mathrm{r}$ that minimizes $E_\mathrm{tot}$. First, let us collapse everything except $t_\mathrm{r}$ into coefficients $a$ and $b$:

$$\begin{aligned} a &\equiv 2c_\mathrm{dd} C_\mathrm{L}^2 V_\mathrm{T0}/k && (7.41)\\ b &\equiv P_\mathrm{leak} n_\mathrm{t} && (7.42)\\ E_\mathrm{tot} &= \frac{a}{t_\mathrm{r}} + b t_\mathrm{r}. && (7.43)\end{aligned}$$

Figure 7.12 shows how the total energy in eq. 7.43 scales with $t_\mathrm{r}$, regardless of the values of $a$ and $b$. We can see that at very high values of $t_\mathrm{r}$, $E_\mathrm{tot}$ is high because of the high leakage energy, and at very low values of $t_\mathrm{r}$, $E_\mathrm{tot}$ is high because of the high switching energy. In between, there is a point where the total energy is minimized.

We can find a formula for the $t_\mathrm{r}$ at this point; it's just where the derivative of eq. 7.43 equals zero, which turns out to be where the switching energy equals the leakage energy:

$$\frac{d}{dt_\mathrm{r}}\left(\frac{a}{t_\mathrm{r}} + b t_\mathrm{r}\right) = 0 \tag{7.44}$$

$$-\frac{a}{t_\mathrm{r}^2} + b = 0 \tag{7.45}$$

$$t_\mathrm{r} = \sqrt{\frac{a}{b}} = \sqrt{\frac{2c_\mathrm{dd} C_\mathrm{L}^2 V_\mathrm{T0}}{k P_\mathrm{leak} n_\mathrm{t}}} \tag{7.46}$$

$$\boxed{t_\mathrm{r} = \sqrt{\frac{2c_\mathrm{dd}}{n_\mathrm{t}}} \cdot C_\mathrm{L}\sqrt{\frac{V_\mathrm{T0}}{k P_\mathrm{leak}}}} \tag{7.47}$$

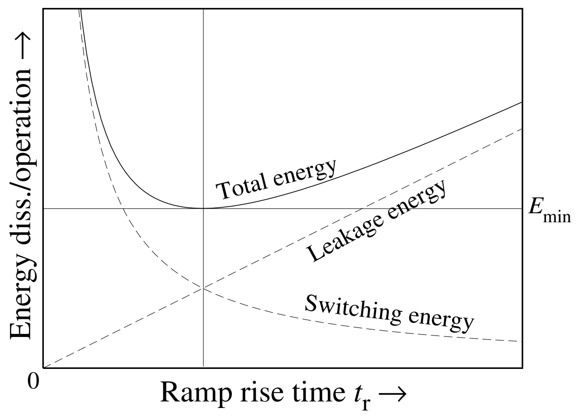



Figure 7.12: How total energy dissipation per operation scales with ramp rise time $t_r$ in SCRL, when leakage is significant. The increasing line is leakage energy, the inversely declining curve is switching energy. Their sum is analytically proven to be minimized when the two components are equal.

---

At this minimum-energy setting for $t_r$, the total energy dissipation is:

$$E_{\mathrm{tot}} = \frac{a}{t_r} + bt_r \tag{7.48}$$

$$E_{\min} = \frac{a}{\sqrt{a/b}} + b\sqrt{a/b} \tag{7.49}$$

$$= \sqrt{\frac{a^2}{a/b}} + \sqrt{b^2 a/b} \tag{7.50}$$

$$= \sqrt{ab} + \sqrt{ab} \quad \text{(note identical terms)} \tag{7.51}$$

$$= 2\sqrt{ab} \tag{7.52}$$

$$= 2\sqrt{\frac{2c_{\mathrm{dd}}C_{\mathrm{L}}^2 V_{\mathrm{T0}}}{k} P_{\mathrm{leak}} n_{\mathrm{t}}} \tag{7.53}$$

$$\boxed{E_{\min} = \left(2\sqrt{2c_{\mathrm{dd}}n_{\mathrm{t}}}\right) C_{\mathrm{L}} \sqrt{\frac{V_{\mathrm{T0}}P_{\mathrm{leak}}}{k}}} \tag{7.54}$$

Looking at eq. 7.54, if we want the energy per operation of an SCRL circuit to be as low as possible, we will want to first minimize the wiring capacitance and other parasitic capacitances we need to drive. Then we'd want to maximize the gain factor $k$ of our transistors. However, if we try to increase $k$ by making the transistors wider, this also increases the capacitance, and the leakage power. So narrower transistors are favored.

Ideally we'd like to get a handle on minimum energy by adjusting the threshold voltage, so as to minimize the quantity $V_{T0}P_{leak}$ in eq. 7.54. But choosing the optimal $V_{T0}$ is actually a bit tricky, since $P_{leak}$ itself depends on $V_{T0}$, in a way which we will now analyze.

#### 7.6.3.2 Adjusting voltages to minimize dissipation

In a single transistor across which there is a voltage drop of $V_{DS} = V_{dd}$, which we will later see suffices to model the leakage through all the transistors attached to a given SCRL signal wire, the leakage power $P_{leak}$ is given by

$$\begin{aligned} P_{leak} &= I_{leak}V_{dd} && (7.55)\\ &= I_{leak}n_{dd}V_{T0} && (7.56)\end{aligned}$$

and $I_{leak}$ for transistors that are supposed to be "off" ($V_{GS} \leq V_T$) is given by a standard formula

$$I_{leak} = I_0 e^{(V_{GS}-V_T)/((1+\alpha)k_B T/q)} \tag{7.57}$$

where $I_0$ denotes the leakage current when the transistor is just barely on the edge of being off (*i.e.*, when $V_{GS} = V_T$). $k_B$ is Boltzmann's constant, $T$ is the absolute temperature, $q$ is the magnitude of the electron charge, and $\alpha$ is a technology-dependent constant fudge factor, which is ideally 0 but in practice is perhaps closer to 1. This factor is needed because real devices are found empirically to have a greater dependence of leakage on temperature than is predicted by the theoretical ideal.

Now, the leakage in SCRL circuits is not really continuous, but fluctuates during the SCRL cycle as different rails split and merge. In static versions of SCRL such as Younis's 3-phase clocking scheme, we can identify two types of leakage: (1) leakage through the middle of a logic gate across a voltage drop of $V_{dd}$ when the gate's supply rails are split, and (2) leakage through a turned-off pass transistor across a voltage drop of $V_{dd}/2$. All these leakages occur through off devices that have a $V_{GS}$ of zero; other off devices with $V_{GS} < 0$ have exponentially less leakage, and so we ignore them. During some transitions, there are also leakages across voltage drops smaller than $V_{dd}/2$. Some of these happen when $V_{GS} < 0$, and the others contribute small amounts to the total leakage power.

One may carry out a careful analysis of leakage based on the timing diagram of Younis's 3-phase clocking cycle. We will not relate the analysis in detail here. However, one finds that for each signal wire, there is leakage inside one of the logic gates that drive that wire during $\frac{22}{24}$ of each cycle, and leakage through a pass transistor for about $\frac{19}{24}$ of the cycle (this latter figure is adjusted to take into account the smaller voltage drops that occur during transitions).

Further, the $I_0$ for the leakage inside logic gates may be different than the $I_0$ for the leakage through the pass transistors, depending on how the devices are sized relative to each other, and also remembering that if a logic gate is not a simple inverter but rather contains several parallel paths, there may be leakage through all of the paths.

However, all of these factors can incorporated into our definition of the effective $I_0$ for the SCRL signal wire, as follows. Let $I_{0\mathrm{G}}$ be the effective $I_0$ in the pullup/pulldown networks of our logic gates (taking into account the widths of devices and number of parallel paths). Let $I_{0\mathrm{P}}$ be the $I_0$ through our pass transistors (taking into account their widths). Then we just define the effective $I_0$ for the single-transistor equivalent model of the SCRL signal wire's average leakage as

$$I_0 = I_{0\mathrm{G}}\frac{22}{24} + I_{0\mathrm{P}}\frac{1}{2}\cdot\frac{19}{24} \tag{7.58}$$

where the $\frac{1}{2}$ compensates for the fact that the leakage through the pass transistors involves a voltage drop of $V_{\mathrm{dd}}/2$ rather than $V_{\mathrm{dd}}$. This substitution is valid because the other factor in eq. 7.57 (the exponential) doesn't depend on the magnitude of the $V_{\mathrm{DS}}$ voltage drop or on which kind of leakage we are looking at, since $V_{\mathrm{GS}} = 0$ for all the significant leakage.

We further note that almost all of the leakage takes place when $V_{\mathrm{GS}} = 0$ and $V_{\mathrm{SB}} = 0$, so that at these times $V_{\mathrm{T}} = V_{\mathrm{T0}}$, and we can substitute $V_{\mathrm{T0}}$ for $V_{\mathrm{T}}$ in eq. 7.57. Further, for conciseness let's define convenient notations for the thermal voltage $k_{\mathrm{B}}T/q$ with and without the $(1+\alpha)$ fudge factor.

$$\phi_T \equiv k_{\mathrm{B}}T/q \tag{7.59}$$
$$\phi'_T \equiv (1+\alpha)\phi_T \tag{7.60}$$

Now we can re-express the leakage current as just

$$I_{\mathrm{leak}} \approx I_0 e^{-V_{\mathrm{T0}}/\phi'_T}. \tag{7.61}$$

Although the above method for estimating $I_{\mathrm{leak}}$ was developed for the particular case of static 3-phase SCRL, it is fairly clear that the same approach could be carried out similarly for other SCRL clocking schemes as well, with appropriate modifications to eq. 7.58. Remember, however, that in 2-phase SCRL, nodes are not always being actively driven, and so high leakages can harm functionality as well as dissipating power; therefore the analysis later in this section will probably not be appropriate for dynamic 2-phase clocking.

Now that we've gotten $I_{\mathrm{leak}}$ expressed in terms of $V_{\mathrm{T0}}$, let's merge eqs. 7.56 & 7.61

back into our expression for $E_{\min}$ (eq. 7.54):

$$E_{\min} = \left(2\sqrt{2c_{\mathrm{dd}}n_{\mathrm{t}}}\right) C_{\mathrm{L}}\sqrt{\frac{V_{\mathrm{T0}}P_{\mathrm{leak}}}{k}} \tag{7.62}$$

$$= \left(2\sqrt{2c_{\mathrm{dd}}n_{\mathrm{t}}}\right) C_{\mathrm{L}} \cdot \sqrt{\frac{V_{\mathrm{T0}}(n_{\mathrm{dd}}V_{\mathrm{T0}})I_0e^{-V_{\mathrm{T0}}/\phi'_T}}{k}} \tag{7.63}$$

$$= \left(2\sqrt{2c_{\mathrm{dd}}n_{\mathrm{t}}n_{\mathrm{dd}}}\right) \cdot C_{\mathrm{L}}V_{\mathrm{T0}}\left(\sqrt{\frac{I_0}{k}}\right) e^{-\frac{1}{2}V_{\mathrm{T0}}/\phi'_T} \tag{7.64}$$

To make this formula easier to work with, we'll express the factor involving the SCRL power and timing parameters $n_{\mathrm{dd}}$ and $n_{\mathrm{t}}$ as just $s$. Also, we note that since $I_0$ and $k$ both scale roughly proportionally to transistor width, the voltage factor $\sqrt{I_0/k}$ is basically independent of transistor width. It scales up with increasing length however (because $k$ scales down proportionally, but $I_0$ does not scale down as much), indicating that SCRL favors designing with minimum-length devices and small gate fan-ins. (Larger fan-ins yield a larger effective length.) In such designs, $\sqrt{I_0/k}$ can be thought of as a width-independent voltage $v_{\mathrm{c}}$ that is characteristic of the particular device technology being used. It can be interpreted as the drive voltage required to turn on a standard-length transistor strongly enough to conduct current at some fixed multiple of the transistor's zero-drive leakage current $I_0$.[3]

Given the above definitions, we can re-express the minimum energy as

$$s \equiv 2\sqrt{2c_{\mathrm{dd}}n_{\mathrm{t}}n_{\mathrm{dd}}} \tag{7.65}$$

$$= 2\sqrt{\frac{2n_{\mathrm{t}}n_{\mathrm{dd}}^3}{3n_{\mathrm{dd}} - 4b_{\mathrm{avg}}}} \tag{7.66}$$

$$v_{\mathrm{c}} \equiv \sqrt{I_0/k} \tag{7.67}$$

$$E_{\min} = sC_{\mathrm{L}}v_{\mathrm{c}}V_{\mathrm{T0}}e^{-\frac{1}{2}V_{\mathrm{T0}}/\phi'_T}. \tag{7.68}$$

Figure 7.13 shows qualitatively how $E_{\min}$ scales as $V_{\mathrm{T0}}$ is changed. Perhaps surprisingly, above a certain point, the minimum energy/op of SCRL actually decreases exponentially as the threshold voltage is increased! This contrasts with the situation in standard CMOS, where higher thresholds mean quadratically larger switching energy, determined by equations like eq. 7.7. The difference in SCRL is that higher

[3]Perhaps $v_{\mathrm{c}}$ is related to the drive voltage needed for strong inversion. This needs further investigation.

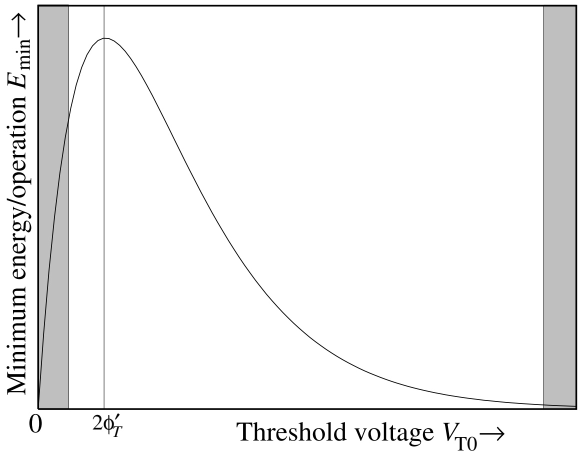



Figure 7.13: How minimum energy/operation scales with $V_{T0}$ in SCRL, as per (7.68). The curve is only meaningful for points to the right of the maximum, but not too far to the right. At the very low voltages at the far left, circuits will not function properly because they will be overwhelmed by leakage currents. At very high voltages far to the right, gate oxides may break down, depending on their thickness. But up to this breakdown point, SCRL minimum energy scales down roughly exponentially, as the ratio $V_{T0}/\phi_T$ is increased.

---

thresholds mean exponentially smaller leakage power, which allows us to run at exponentially slower speeds and still not have leakage dominate the total energy, which thus allows exponentially less energy to be dissipated during our quasistatic charging at high thresholds.

The curve in fig. 7.13 also suggests that at very low thresholds, the energy/op can be made arbitrarily small as well. However, this part of the curve is probably not accurate. Further analysis ([68]) shows that the maximum point on the curve occurs when $V_{T0} = 2\phi'_T$, twice the adjusted thermal voltage. At thresholds near or below the thermal voltage, a $V_{dd}$ that is only a small fixed multiple of the threshold voltage will probably not be high enough to produce strong inversion, and the square-law equation (7.10) will probably not accurately represent the source-drain current of our transistors, upon which the above analysis was based. Moreover, at low thresholds, the high leakage power will call for a very short rise time from eq. 7.47; if the rise time is too short, it will not be large compared to the effective $RC$ of our transistors, which will invalidate the assumptions upon which the analysis of section 7.6.2 was based.

Also, all of the analysis above is only reasonably accurate for relatively large devices. As we mentioned 7.1.5, as transistors shrink below present-day sizes, a variety of short-channel effects become increasingly significant in their influence on the I-V characteristics of the device. As we move deeper into this short-channel regime, the analytical expressions on which the above sections were based will become increasingly inaccurate. It was deemed beyond the scope of the present work to correct all of the above analysis to take short-channel effects into account, although such corrections will be important if adiabatic techniques are to be applied to low-energy computing applications in the near future.

### 7.6.4 Fixing a problematic case for plain SCRL

We believe there may actually be a small problem in ordinary SCRL, as originally described by Younis and Knight, a problem that seems to lead to non-time-proportional dissipation, and perhaps even to a dissipation per operation that is bounded below by $k_{\mathrm{B}}T$ (although this is not yet certain). This particular problem occurs even at low temperatures, at which ordinary leakage currents become exponentially more insignificant.

The problem occurs in multi-input SCRL gates such as NAND and NOR gates, as well as in more complex gates, but not in single inverters. The problem is due to the fact that under some inputs, part of a pull-up or pull-down network may be conducting even if the whole network is not. So for example, in a NAND gate, when the output is high, part of the pull-down network may actually be pulled up as well (see fig. 7.14). So an n-FET, which is designed for passing low voltage, is being asked to pass a high voltage. n-FETs can only conduct well over part of the high range, up to a threshold drop $V_{\mathrm{T}}$ away from the high voltage $V_{\mathrm{dd}}$. So, an internal node in the pulldown network will follow the output ramp closely up to this voltage, but after that its rate of increase will slow down, because the effective resistance of the n-FET increases exponentially as its source voltage rises several thermal voltages $\phi_T$ above the threshold point. Therefore, the voltage drop over this n-FET will become significantly larger than voltage drops in the rest of the circuit, and some non-zero amount of charge will flow over this voltage drop, as the n-FET source voltage gradually edges up to a few thermal voltages above $V_{\mathrm{dd}} - V_{\mathrm{T}}$.

Clearly, this situation will have some impact on the analysis of the energy dissipation of SCRL, but it is not yet clear exactly what the impact will be. It is difficult to derive an analytical expression for the dissipation due to this effect. However, it seems likely that the major result will be that the overall dissipation of SCRL circuits does not decrease anywhere near as quickly as linearly with frequency, once the overall dissipation is low enough that the dissipation due to the above effect becomes significant.

To test this intuition, I performed a simple numerical simulation of this situation for an example circuit, and found that when the ramp time was long enough so that

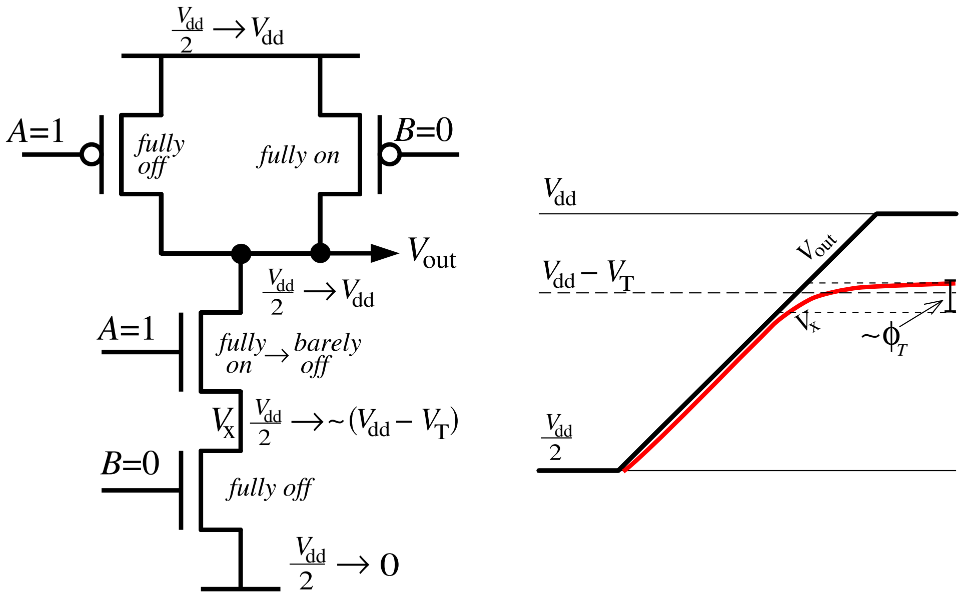



Figure 7.14: A case in the simplest version of SCRL where there may be energy dissipation that does not scale down in proportion to operating frequency. Consider a NAND gate when the inputs are different, and the higher input goes to the innermost n-FET. When the rails split, that n-FET will conduct, and the internal node voltage $V_x$ will track $V_{out}$ arbitrarily closely (depending on the ramp time), until $V_x \approx V_{dd} - V_T$. Then, the FET will begin to cut off, and $V_x$ will lag farther behind $V_x$, while it continues to increase over a range of several additional $\phi_T$'s. In the depletion regime, it takes several $\phi_T$'s worth of $V_{GS}$ voltage change in order for a MOSFET's channel charge (and thus its conductance) to fall off by several factors of $e$. During this time, drain voltage continues to increase at a relatively faster rate and so charge will be falling over a relatively large voltage drop. NOR gates and more complex gates will suffer from this problem as well.

the total dissipation in the earlier part of the ramp was less than $k_{\mathrm{B}}T$, the dissipation in the subthreshold regime was still $\sim 3000\, k_{\mathrm{B}}T$—but it was still decreasing slowly as the ramp time was lengthened. If the ramp time was lengthened far enough, perhaps the source voltage would continue to track the ramp closely all the way up to $V_{\mathrm{dd}}$, and the dissipation in this transistor would be less than $k_{\mathrm{B}}T$—but then we are talking about ramp times so long that energy losses due to leakage would be significant, and greater than $k_{\mathrm{B}}T$, in other parts of the circuit. The logic would no longer function reliably because leakage currents would be comparable to charging currents. Overall, it is not yet clear whether or not this effect leads to a true $\sim k_{\mathrm{B}}T$ lower bound on dissipation in these simple circuits.

Fortunately, regardless of the precise effect, there is a simple way to fix SCRL to prevent this problem from occurring, and restore guaranteed time-proportionate reversibility to SCRL. That fix is to transform the problematic transistors into CMOS pass gates, with appropriate inputs, which conduct well over the entire voltage range that they might encounter. Thus, at low speeds the voltage drop across the transistors will always remain insignificant. The NAND gate problem in figure 7.14 can be repaired via the addition of just a single p-FET, as shown in fig. 7.15.

A more general way to fix all SCRL logic gates would be to just use dual-rail (complementary) logic everywhere, to ensure that the appropriate complementary signal for use in pass gates is available. This would have the advantage that it would also eliminate the need for 2-level retractile cascades in non-inverting logic stages, and would lead to greater data-independence of the overall capacitance of the chip, allowing the resonant power supply signal to be tuned more cleanly. The primary disadvantage would be the need for a roughly factor of 2 increase in the number of gates, wafer surface area, and entropy coefficient. But in the spirit of the asymptotic emphasis of this thesis, we remind ourselves that this is only a small constant factor.

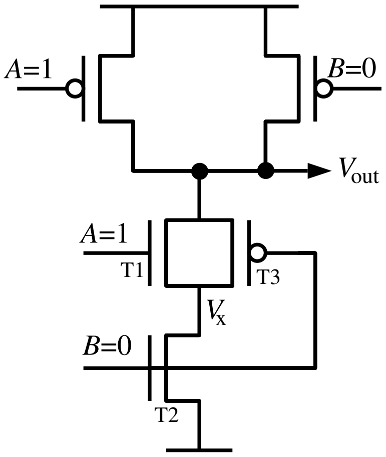
That concludes our discussion of general properties of SCRL. In the next section we discuss the particular SCRL-based circuits that we designed and fabricated.

## 7.7 Experimental SCRL Circuits

The first SCRL circuits to be fabricated were those of Younis [191] in his original experimental tests. Younis fabricated a demonstration chip that included reversible adders and multipliers. Younis tested these chips and measured their power dissipation, and found that it scaled roughly as predicted within the range of sensitivity of the measurements. More accurate measurements of dissipation in other adiabatic circuits have been performed by Solomon and Frank (1995, [153]).

In parallel with my own work, Carlin Vieri has been designing Pendulum, a fully-adiabatic RISC-style processor based on SCRL. I assisted Vieri with various Pendu-

(a)




(b)

| A | B | T1 | T2 | T3 | PDN Conducts? |
|---|---|---|---|---|---|
| 0 | 0 | off | off | on | no |
| 0 | 1 | off | on | off | no |
| 1 | 0 | on → off | off | on | no |
| 1 | 1 | on | on | off | yes |

Figure 7.15: A simple way to eliminate the problem discussed in figure 7.14. A p-FET "T3" tied to input B is placed in parallel with the inner n-FET "T1". T3 will be fully turned on in the problem case where A = 1 and B = 0, and thus node $V_x$ will follow $V_{out}$ all the way up to $V_{dd}$, and so there will be no problematic voltage drop over T1. The truth table on the right verifies that the logic of the pulldown network remains correct in all cases with the addition of T3.

lum instruction set issues, about which I will have more to say in chapter 9. Scott Rixner and I designed and tested Tick, a non-adiabatic 8-bit version of Pendulum that was intended to gauge the complexity of the reversible instruction set design. (Unfortunately, the chip failed to operate fully, due to inaccuracies in the layout design-rule checking software that we used; design modifications would be necessary to get Tick working.) As of this writing, the fabrication of the fully adiabatic Pendulum prototype has recently been completed, and it is now in the testing phase.

Vieri and colleagues have also designed and fabricated XRAM [179], a fully-adiabatic static memory component, whose design I discussed with Vieri, but was not intimately involved with.

The primary SCRL design effort that I have been centrally responsible for (with assistance from other group members) has been the design of FlatTop, a chip comprised of an adiabatic mesh-style array of processing elements that are very simple, yet capable of fully universal reversible computation. The FlatTop chips can be tiled in 2-D or 3-D arrays, and together with the appropriate external resonant rail generators, reversible communications links, and power delivery/entropy removal systems, would constitute a concrete example of a time-proportionately reversible 3-D mesh processor, such as we conjectured was asymptotically optimal in ch. 6. As such, sufficiently large arrays of FlatTop chips would be, in principle, faster for their size than any possible irreversible machine. At least, this would be the case if the FlatTop design was repaired to circumvent the dissipative flaw in SCRL that we discussed in §7.6.4, which had not yet been discovered at the time FlatTop was designed. Also, in reality the FlatTop machine sizes necessary to outperform the fastest irreversible architectures would be astronomically large. But FlatTop is still an important proof-of-concept, demonstrating that it is not only possible but fairly straightforward to design a universal, sequential reversible mesh processor using fully adiabatic circuits.

We now discuss some of the background for the FlatTop design.

### 7.7.1 The Billiard Ball Model

The basic operation of FlatTop is to simulate the "Billiard Ball Model" (BBM) of computation, an idealized physical model of reversible computation that was introduced by Fredkin [74] in the course of some of his early work on reversible logic circuits. The model involves computation using idealized perfect spheres, that move ballistically through 2-D space along precise trajectories, and bounce off fixed walls and each other in perfectly elastic collisions. Fredkin showed that using only this behavior, one can construct elementary reversible boolean logic gates (fig. 7.16) and put them together to compose arbitrary reversible logic circuits.

The BBM is of course an idealization. In reality, to avoid the accumulation of

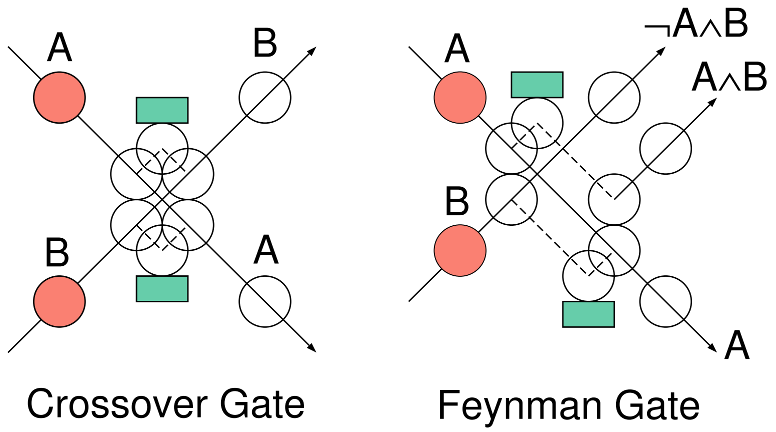



Figure 7.16: Examples of two logic gates in the physical billiard ball model of computation. The "Crossover Gate" on the left permits two ball-signals to effectively pass through each other without delay. The "Feynman Gate" on the right computes a reversible AND/NAND function.

---

errors in ball trajectories, the balls would have to be made to travel along troughs in a potential energy surface, pushed along by waves of potential to keep their global timing consistent, and to make up for frictional losses. In reality, the collisions would not be perfectly elastic—but this is just an example of time-proportionate reversibility, since real collisions between hard objects become more nearly elastic as the speeds involved are decreased. Energy can be injected into the system, at a rate per interaction that declines with the balls' speed, to keep the system progressing forwards at a constant rate. In talks given at MIT, Fredkin has discussed physically plausible mechanisms for performing the above-mentioned types of corrections.

But the main conceptual importance of the model is that it provides a way to see how extremely simple interactions—ball collisions—can be used to build up arbitrarily complex logic circuits. Further, in essence it is a purely digital model. It only cares about discrete positions and times. In contrast, other researchers have investigated analog models of computation in which an unlimited number of decimal places of precision in physical parameters are used to carry information important to computation [177, 151, 148], but such models are not physically realistic, because in quantum mechanics, bounded systems only have a finite number of distinguishable states, as we noted in §2.2; infinite precision is not a physically realistic assumption.

Furthermore, the BBM is ultimately a parallel model of computation—interactions may be occurring in many parts of the space simultaneously—and a 3-D version of it could asymptotically efficiently simulate any (non-quantum) physical algorithm.

Due to its ultimate digital nature, its simplicity, its reversibility, its asymptotic

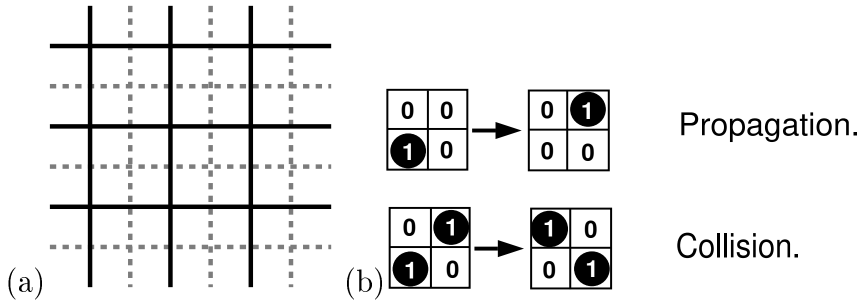



Figure 7.17: The Billiard-Ball Model Cellular Automaton. (a) Updates are performed alternately in two overlapping meshes of 2×2 blocks of cells. This partitioning scheme, also called the "Margolus neighborhood," is an easy way to produce a global reversible dynamics from a local reversible update rule. (b) BBMCA block update rule. In a 2×2 block of cells, a single 1-bit moves to the opposite corner of a block (propagation), whereas 2 bits in opposite corners move to the other 2 corners (collision). All rotations of the illustrated cases also apply. All other configurations remain unchanged. Note that this rule is reversible.

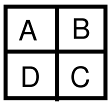
---

efficiency, and its universality, the billiard ball model is a useful starting point for investigations of reversible computation.

### 7.7.2 The Billiard Ball Model Cellular Automaton

Making the ultimately digital nature of the billiard ball model even more apparent, in 1983 Norman Margolus invented a digital cellular automaton, with only 1 bit per cell, that precisely and efficiently simulates the billiard ball model on a 2-D grid of cells [116, 115]. In this "Billiard Ball Model Cellular Automaton" (BBMCA), balls are represented by pairs of 1-bits that move along diagonal paths through the grid of cells. The grid is updated by breaking it into two overlapping meshes of 2×2 blocks of cells (fig. 7.17a), which apply on alternate update steps, and within each block transforming its state according to a simple reversible update rule (fig. 7.17b). The reader should be warned that it is somewhat non-obvious how this update rule leads to behavior that imitates the billiard ball model; see [116] or §2.4 of [115] for a detailed description.

Because of the extreme simplicity of the BBMCA update rule, we chose it as our target for our proof-of-concept mesh architecture, which was therefore named FlatTop, after the local billiards pub, "Flat-Top Johnny's." This architecture is not intended to be particularly efficient in terms of its constant factors, or to be particular easy to program directly. But ultimately, since it can asymptotically efficiently simulate any (non-quantum) parallel architecture, it demonstrates the points




$$S = (A + C)\,(B + D)$$
$$A' = S\,A + \overline{S}\,\overline{A}\,(C + B\,D)$$
... and similarly for B, C, D

Figure 7.18: Boolean logic form of BBMCA update rule. The S bit is true when there are bits in both diagonals, in which case the block should remain *static*, unchanged. If there aren't bits in both diagonals, the block may change, and a given bit (e.g., A) should turn on if it was off and the opposite bit was on (propagation rule) or the two adjoining bits were on (collision rule).

---

we are trying to make about asymptotic scaling. Given a large enough array of FlatTop chips, one could efficiently simulate any alternative architecture or programming model on top of it.

A more practical reversible mesh processor would probably have a design and a programming model closer to that of Vieri's RISC-style Pendulum chip. In fact, given a suitable communication network between neighboring processors, Pendulum itself would probably work fine as a processing element. But FlatTop was relatively easy to design, and it gives us the proof-of-concept we wanted. It also seems entirely possible to design a reversible FPGA element that would be intermediate in complexity and programmability between FlatTop and Pendulum, and might permit greater logical density on a wider variety of problems than either.

### 7.7.3 Logic minimization

After choosing the BBMCA as the target functionality, the next step was to translate the BBMCA update rule into a boolean logic expression that could be easily implemented in a real SCRL logic circuit. After trying several ways of translating the update rule into a boolean formula, we settled on the solution shown in fig. 7.18, which is the simplest such representation of the logic that we have found so far.

The idea behind these formulas is that under the BBMCA update rule, one of two things happen: either the block might change, or it must stay the same. It must stay the same if there are 1 bits in both of the two diagonals across the block. Letting "S" represent this case, we have (with reference to fig. 7.18) that S is (A or C) and (B or D).

Then, the new value of A itself, A′, is fairly easy to compute. If S, then A′ is just A, unchanged. Otherwise, A′ turns on if and only if A was originally off, and either the opposite bit was on (propagation rule), or both of the adjacent bits were on (collision). If all bits are off, they all remain off. One can see by inspection of the possible cases that this rule yields exactly the BBMCA update rules.

To implement this logic in SCRL, we would need at least two logic stages in each processing element: The first stage would take the initial state of the 4 cells in a block as input, and would produce the S signal using a complex gate implementing the formula for S, while passing the input bits through unchanged to the second stage. The second stage would use a complex gate for each cell to compute its new state given S and the original cell states. This was the basic function of each processing element. Since we were using 3-phase SCRL, a third stage was needed to put the data in the correct phase for passing to an adjacent processing element. We also used the third stage to implement a special shift register functionality for initializing the whole array.

### 7.7.4 FlatTop array design

Figure 7.19 outlines how the array of processing elements was connected. The horizontal-vertical grid shows the cell space partitioned according to the Margolus neighborhood into 2 overlapping grids of $2\times2$ blocks, as in fig. 7.17a. The updating of each block is handled by a corresponding processing element, which can be visualized as resting at the center of that block. The state of each cell is passed back and forth along wires between the PE's at the centers of the two diagonally-overlapping blocks that contain the given cell. The array of PE's thus naturally forms a mesh that is oriented at a 45° angle relative to the original CA mesh. We orient the chip edges along this diagonal mesh, so that the wires between processing elements can be parallel to the chip edges, which is a constraint required by some fabrication processes.

One artifact of this design is that, if all the processing elements operate simultaneously, they will actually be simulating two parallel non-interacting BBMCA systems. We can separate the PEs into "dark" and "light" processors in a diagonal checkerboard pattern as pictured. One CA system is the one that is processed by light processors on even-numbered time-steps and by dark processors on odd-numbered steps. The other CA system is the one that is processed by dark processors on even-numbered steps and by light processors on odd-numbered steps. The two systems do not interact at all, and they can be used to represent two completely different configurations of balls and mirrors in the BBM. Using both systems can be seen as a way of making more efficient use of the hardware, which would otherwise be only half utilized.

The two systems can be connected together at one or more chip edges to form one larger system with an alternative topology. In fact, we did this in our prototype FlatTop chip. Also, at various points at the chip edges we passed signals to bidirectional I/O pads to connect to neighboring processors in a larger array. Normally in the BBMCA the grid is 2-D, but there is nothing to prevent a topology that is locally 2-D on-chip, but globally 3-D, with the chips connected in a 3-D mesh. As

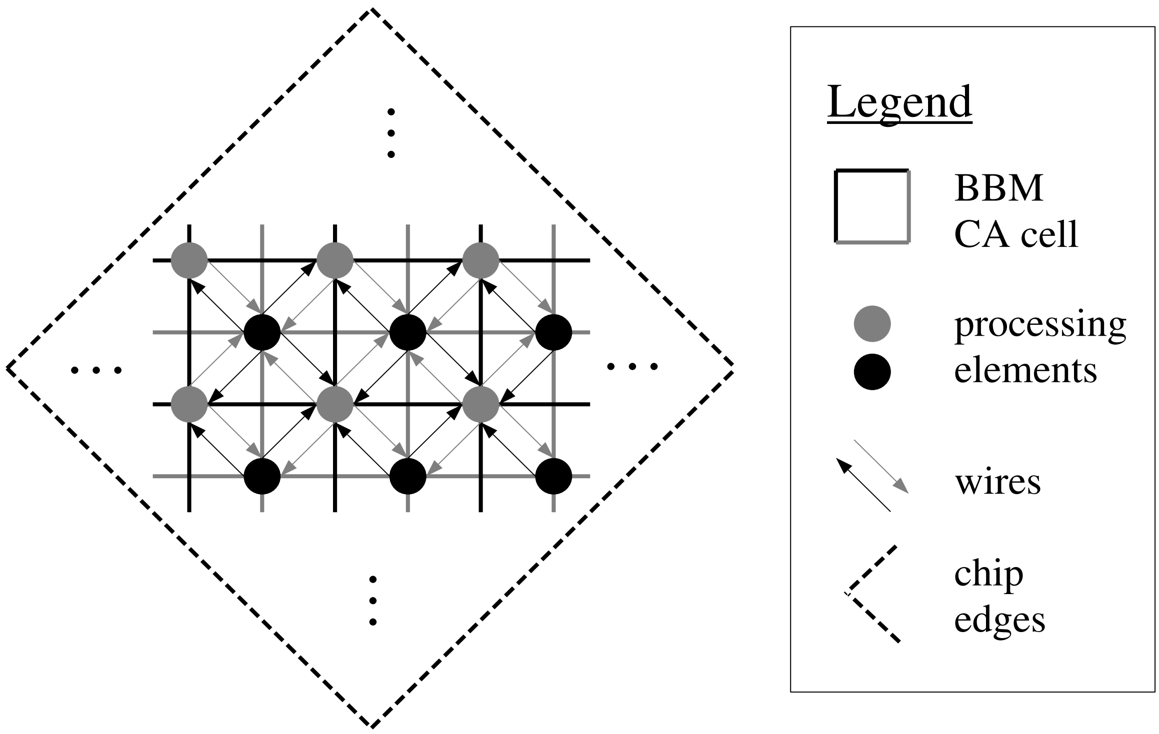



Figure 7.19: Schematic illustration of a grid of FlatTop processing elements. Each 2×2 block of cells in the Margolus partitioning (see fig. 7.17a) is updated by a different PE. The state of a given cell is stored, on alternate time steps, on one or the other of the two wires running between diagonally adjacent PEs. The PEs thus naturally form a square mesh oriented 45° from the BBMCA mesh. To make layout more convenient, the chip edges were oriented parallel to the PE mesh.

| NMOS | | PMOS | |
|---|---|---|---|
| Var | Value | Var | Value |
| $\phi_0$ | 1.1V | $\phi_0$ | 0.8993 |
| $m_{\text{j}}$ | 0.726 | $m_{\text{j}}$ | 0.4905 |
| $m_{\text{jsw}}$ | 0.2451 | $m_{\text{jsw}}$ | 0.2451 |
| $C_{\text{j}}$ | 4.67x10$^{-4}$ F/m$^2$ | $C_{\text{j}}$ | 8.76x10$^{-4}$ F/m$^2$ |
| $C_{\text{jsw}}$ | 3.20x10$^{-10}$ F/m | $C_{\text{jsw}}$ | 2.13x10$^{-10}$ F/m |
| $t_{\text{ox}}$ | 9nm | $t_{\text{ox}}$ | 9nm |
| $\mu_{\text{n}}$ | 978.1 cm$^2$/V$^2$-s | $\mu_{\text{p}}$ | 228.5 cm$^2$/V$^2$-s |
| $C_{\text{ox}}$ | 3.89x10$^{-15}$ F/$\mu$m$^2$ | $C_{\text{ox}}$ | 3.89x10$^{-15}$ F/$\mu$m$^2$ |
| $k'_{\text{n}}$ | 3.80x10$^{-4}$ A/V$^2$ | $k'_{\text{p}}$ | 8.889x10$^{-5}$ A/V$^2$ |

Table 7.5: Device parameters for the HP14 process, from Cadence models. These figures were used in our hand-calculations of the minimum energy dissipation per operation in FlatTop.

long as each chip has at least 3 external connections, a globally 3-D network can be made [185].

Appendix A shows most of the Cadence schematics and layout for the FlatTop unit cell and processor array. The design was simulated using Verilog and functioned as expected in simulation. Individual PEs were simulated in HSPICE to verify that there were no errors causing $CV^2$ dissipation. The chips have been fabricated but have not yet been tested. It is expected that their basic functionality will work, and that fairly low-energy operation is possible, but that the dissipation will not be quite as low as was originally projected due to the SCRL flaw we discussed in §7.6.4.

### 7.7.5 Minimum energy estimation

After designing FlatTop, we carried out an approximate hand-analysis of the circuit, using the device parameters of the fabrication process we used (table 7.5), and the formulas we derived in §7.6.3.1 (p. 190), to estimate the circuit's minimum energy dissipation per operation when operated at 3.3 V, the standard supply voltage for the process (HP14) that was used. As the circuit is clocked more slowly, switching energies decrease proportionally, but leakage energies increase. As we pointed out in §7.6.3.1, the minimum total energy per operation turns out to be achieved at the speed at which switching energy equals the leakage energy (refer back to fig. 7.12, p. 192).

The energy estimation procedures we used are also described in more detail in our conference paper [72] on FlatTop. The upshot was that the optimal cycle time turned out to be around $12\mu s$, at which point FlatTop would dissipate around 10 fJ per cycle per cell, whereas in our estimation the equivalent iCMOS circuit dissipates around 20 pJ, a reduction of energy dissipation by a factor of 2000! When cooled below room temperature, the chip would have less leakage, and even greater energy efficiency could be obtained at lower speeds.

Unfortunately, there has not yet been an opportunity to actually test the energy dissipation in FlatTop. Indeed, such low levels of dissipation would be hard to measure accurately. Also, the actual dissipation is probably higher than we first estimated, since at design time we did not know about the SCRL bug mentioned in §7.6.4. However, if this bug were fixed, we believe that dissipation in the chip would be roughly as predicted.

## 7.8 Resonant power supply techniques

In most of the above, we have glossed over the issue of how to build a resonant power supply that can provide the power/clock waveforms that SCRL requires, in such a way that the energy loss per cycle scales down arbitrarily in proportion to frequency. This particular issue has not been the primary focus of my own research, but it is important for the overall scaling results.

Various techniques for powering adiabatic circuits have been described in the literature: [156, 157, 194, 63, 3]. However, many techniques do not have asymptotically zero dissipation. One interesting recent technique is the one developed in our group by Becker and Knight [12, 13]. This technique uses a transmission line with tuned nonuniformities that allow it to end up resonating with any desired waveform; in particular, it has been used to produce the trapezoidal ramp-shaped waveforms used by SCRL[4]. However, due to nonlinear effects of the signal on resistivity in Becker's transmission lines, the dissipation per cycle does not scale down quite in proportion to the frequency $f$, but apparently rather as $\sqrt{f}$. Thus the scaling results for a reversible system powered by these circuits is not quite as good as the ideal.

An open problem for adiabatic computing is the design of an external resonant element that can provide waveforms usable for adiabatic computing while at the same time retaining the property of having a dissipation per cycle that scales down to arbitrarily small levels, in direct proportion to frequency. It is also important for the cost-efficiency arguments of §6.2 that the cost of the resonant element does not increase substantially as its frequency is decreased. One problem with using Becker's transmission line approach in reversible computing is that the length of the transmission line scales up in proportion to its frequency, therefore increasing the cost

[4] As reported by Becker in personal discussions.

of the system when run at low speeds.

However, we (optimistically) suspect that if enough attention is paid to the issue of designing resonant power supplies, a power supply technique having the desired properties can be found. Certainly it has not been proven, to our knowledge, that no such technique can exist. It is an important area for future work to determine with certainty whether an ideal supply technique can exist, and to design a technique having the most favorable scaling properties that are physically possible.

## 7.9 Scaling SCRL to future technology generations

Although it is difficult to precisely forecast the future performance of CMOS-based technology in the near term (next 10 years), over the long term one can point out some qualitative aspects of how performance scales with technology shrinkage in CMOS-like circuits, based on some fundamental scaling laws, and see how these factors affect SCRL compared with standard irreversible CMOS.

Consider the effect of shrinking all circuit dimensions by a factor of $f_\ell$ (that is, any length $\ell$ is shrunk to $\ell/f_\ell$). Intuitively speaking, one must consider the effect of shrinking all dimensions, rather than just one or two, because any dimension that does not shrink will eventually dominate in terms of its parasitic effects, and one could improve performance by shrinking that dimension as well.

Under $f_\ell$ scaling in all dimensions, the width and length of capacitive elements will decrease, but so will their thickness, so capacitances will decrease by $\sim f_\ell$. Resistive elements will get shorter, but also narrower along the other two dimensions, so their resistance will increase by $\sim f_\ell$. Characteristic $RC$ delays through resistive elements thus do not scale down by much, since these factors cancel out. (In the near term, resistance is dominated by the effective resistance through transistors. These are harder to model accurately. But if their resistance decreases, $RC$'s might decrease for a while, but then eventually the resistance in the wires will come to dominate, so at some point $RC$ cannot decrease further.)

Earlier, we saw that energy dissipation per operation in adiabatic technologies such as SCRL scales as $CV^2\frac{RC}{t}$. If the $RC$ part doesn't improve much beyond a certain point, as we just saw, then what about the $CV^2$ part? This part corresponds to node energy. For a while, $C$ will scale down as $1/f_\ell$, and $V$ at some point will be forced to scale down at least as fast as $1/f_\ell$ because otherwise, as gate oxides get thinner, the electric field through the oxide would increase and at some point would break down. So $CV^2$ eventually must scale down at least as fast as $f_\ell^{-3}$. This makes sense intuitively, because it corresponds to the energy density in the circuit not increasing beyond a fixed maximum level that the circuit's materials can withstand.

But, can $CV^2$ continue to decrease indefinitely? In §7.1.2, p. 151, we already demonstrated that in irreversible CMOS, we can show that $CV^2$ cannot decrease below a reliability-dependent factor times $k_\mathrm{B}T$ from a pure thermodynamics argument. But this reliability-based limit affects SCRL as well. The electrons in a circuit at normal temperatures behave like a thermal gas, and this leads to a well-known noise component called "$kT/C$ noise," because anything that samples a signal will find that the $\sigma^2$ variance in the sampled voltage is $k_\mathrm{B}T/C$, where $C$ is the capacitance of the sampling node (*cf.* p. 155). Such sampling occurs in SCRL all the time; each time a pass gate in an bidirectional latch cuts off, it can can be viewed as sampling the previous logic gate's output voltage on a sampling node whose capacitance is just the load capacitance of the latch output. If thermal noise causes a sampling error comparable to $V_\mathrm{dd}/2$, correct logical functionality can be impaired. Therefore, just like in irreversible CMOS, node energies in SCRL cannot be made smaller than a reliability-dependent factor times $k_\mathrm{B}T$.

Therefore, as typical node energies decrease with $f_\ell^{-3}$ at a given temperature, eventually they will reach this point where further decreases cannot be made without sacrificing reliability, so to continue shrinkage, the temperature will have to start scaling down as $f_\ell^{-3}$ as well. But note that at this point, the entropy generation per operation in SCRL is no longer decreasing along with the shrinkage. We know $S = E/T$, and the dissipation $E$ is decreasing as $f^3$, but now so is $T$. So ultimately, SCRL's entropy generation per operation becomes exactly $RC/t_\mathrm{r}$ times a reliability-dependent constant ($\ln N$), regardless of further circuit shrinks.

Further decreases in entropy coefficients beyond this point would require non-scaling-related decreases in $R$, such as less resistive wiring materials, a switching device with better I-V characteristics than MOSFETs (perhaps micro-electro-mechanical switches, if they could be made small enough), or even superconducting circuit elements. Irreversible CMOS, on the other hand, could not take full advantage of such improvements, because its dissipation per operation does not fundamentally improve with $R$, and its entropy generation per operation is bounded below by $\sim \ln N$ nat.

This leads to a long-term advantage of SCRL as the technology scales. As lengths are shrunk by $f_\ell$, one can create $f_\ell^3$ times as many processing elements out of a given mass of materials. Beyond a point, entropy generation for a fixed mass of reliable irreversible CMOS scales as $\sim$ nat, whereas entropy generation in SCRL is $\sim \frac{RC}{t_\mathrm{r}}$ nat, where we assume we have reached a limit where $RC$ cannot be decreased further. But as the number of processors increases with $f_\ell^3$, the clock frequency of the irreversible machine must be slowed down for some tasks by a factor $f_\ell^{1/3}$, due to the increasing number of processing elements and the heat-removal arguments of §6.2.3.1, whereas the SCRL machine only needs to be slowed by $f_\ell^{1/4}$. Therefore, as $f_\ell$ increases, the overall processing rate $\mathcal{R}_\mathrm{op}$ of the irreversible machine goes as $f_\ell^{2\,2/3}$ whereas the reversible machine goes as $f_\ell^{2\,3/4}$, for a reversible advantage increasing as $f_\ell^{1/12}$ as devices shrink by a factor of $f_\ell$.

The above analysis imagines that it is possible to continue scaling MOSFETs without fundamental differences up to and beyond the point where reliability constraints become dominant. In reality, MOSFETs may hit a wall for other reasons before this (*cf.* [121]), at which point, we might have to switch to a radically different technology for further improvements. Different technologies might have different scaling considerations, but in any irreversible technology, the lower bound of $S = 1$ nat still holds, and in any reversible technology, we expect there will be a characteristic transition time parameter $t_c$, playing a role similar to the $RC$ parameter in CMOS, that approximately gives the entropy coefficient for the technology (the entropy goes below 1 to the extent that $t$ goes above the characteristic transition time). To the extent that this expectation is true, the reversible advantage with shrinking components will be at least the $f_\ell^{1/12}$ factor mentioned above, and if the characteristic time were to scale down with the length scale as well, so much the better for reversible technology. In that case, processing rate would scale as $f_\ell^3$, an advantage of $f_\ell^{1/3}$ over irreversible technology.

## 7.10 Mostly reversible computation

This thesis primarily focuses on the concept of *arbitrarily* reversible computation. But there are of course benefits to be gained from a more limited use of reversibility in digital circuits. For example, one could construct a processor's functional units using fully adiabatic circuits, and use irreversible switching only to update the high-level processor state between instructions. Or, one could identify the highest-power components of a chip (often, the long buses) and apply energy-recovery techniques only to those sections (this is the direction being explored by the ISI ACMOS group [4, 2, 170]).

Such limited uses of reversibility are potentially quite beneficial in highly energy-limited environments such as portable or embedded systems [66, 1]. And since reversibility need not be complete in order to gain substantial energy savings in these applications, the algorithmic overheads for full reversibility that we explored in §3.4 need not apply.

This line of work can lead to immediate practical, commercially viable products, thereby introducing adiabatic circuit concepts to industry. After this introduction, we expect that gradually over time, users will come to demand more and more computational power using less and less energy, and so the degree to which systems will need to rely on reversible techniques will increase. Eventually, we expect designs for energy-limited systems will converge to encompass the arbitrarily-reversible sort of architectures that we discuss in this thesis.

## 7.11 Adiabatic Circuits—Conclusion

In this chapter we reviewed the most immediately feasible technology for reversible computing, namely the technology of adiabatic circuits. We discussed the prototype mesh-style processor that we constructed using this technology, which is a proof of concept that illustrates that adiabatic circuit techniques such as SCRL are powerful enough to implement fully reversible parallel machines such as we proposed in ch. 6.

The most important areas for future work on adiabatic circuits, in the context of exploring the limits of computing, include:

- More thorough analysis of the precise performance characteristics of SCRL (or similar techniques) compared with irreversible CMOS in near-future technology generations, to obtain a more precise estimation of the cost level above which reversibility confers a real advantage.

- Research on directions in VLSI technology (such as MEMS switches or superconducting materials) that might lead to much lower-resistance switches, which would benefit adiabatic techniques much more than irreversible techniques.

- More research on resonant power supplies for adiabatic circuits, in search of a technique whose dissipation scales down in proportion to frequency, asymptotically all the way to zero, while still providing waveforms that are suitable for use in SCRL or comparable adiabatic circuit techniques.

- Similarly, design of reversible or even ballistic interconnection technologies for communication between adiabatic circuit chips.

- Design of good parallel reversible processor architectures, building on FLATTOP and Vieri's Pendulum work.

In summary, the future for reversible computing with adiabatic circuits looks interesting, but several new developments are still needed before adiabatic techniques could become competitive with traditional techniques at affording cost-efficient supercomputing. It may well be that in the short run, more effective cooling systems (which benefit irreversible techniques more than reversible techniques) will be easier to develop. However, in the long run there are limits to how much cooling systems can be improved—and cooling a system does not ultimately reduce total energy. Meanwhile, various factors on the horizon threaten the ability of CMOS circuits to keep shrinking indefinitely.

Eventually, to gain further improvements in machine speed, it seems we may well be forced to jump to an alternative, non-CMOS-like, computing technology. Interestingly, many of the alternative technologies that have been proposed by various researchers are capable of efficient reversible computation. In the next chapter, we briefly survey some of these, and then in chapter 9 we go on to discuss issues in the architecture and programming of reversible machines, regardless of the details of the underlying reversible logic technology.