Asymptotically Zero Energy Computing Using Split-Level Charge Recovery Logic

by

Saed G. Younis

S.B. Electrical Engineering, Massachusetts Institute of Technology (1986)
M.S. Electrical Engineering and Computer Science, Massachusetts Institute of Technology (1989)

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1994

Signature of Author ....
Department of Electrical Engineering and Computer Science
June 1, 1994

Certified by ....
Thomas F. Knight, Jr.
Principal Research Scientist of Electrical Engineering
Thesis Supervisor

Accepted by ....
Frederic R. Morgenthaler
Chairman, Departmental Committee on Graduate Students

Asymptotically Zero Energy Computing
Using
Split-Level Charge Recovery Logic

by
Saed G. Younis

Submitted to the Department of Electrical Engineering and Computer Science
on June 1, 1994, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

The power requirement of CMOS circuits, although once thought comparatively low, is rapidly becoming a major concern in the design of personal information systems and large computers.

In CMOS, computation is carried out through periodically shuttling charge among the nodes of the circuit having varying potentials. Thermodynamically, we know that it is possible to shuttle energy packets among reservoirs at differing potentials in a lossless and reversible fashion in so far as that process is carried out as a series of quasistatic changes along a sequence of neighboring equilibrium states.

In this work we present a number of new CMOS logic families within which the transfer of charge between the nodes occurs quasistatically. We present the minimum conditions under which we obtain quasistatic operation and show how they directly lead to the requirement for logical reversibility. We show that operating quasistatically, these logic families have an energy dissipation that drops linearly with operating frequency. This means that the power consumption of these circuits drops quadratically with operating frequency as opposed to the linear drop of conventional CMOS.

The circuit techniques in these new families rely on constructing an explicitly reversible pipelined logic gate, where the information necessary to recover the energy used to compute a value is provided by computing its logical inverse. Information necessary to uncompute the inverse is available from the subsequent inverse logic stage.

We describe our early Charge Recovery Logic (CRL) family as well as the much improved Split-Level Charge Recovery Logic (SCRL) family. We show that for SCRL, such energy saving is achieved with little increases in area and circuit complexity. We demonstrate this by our success in building and testing the first fully quasistatic 8×8 multiplier (SCRL-1) employing SCRL circuit techniques.

KEYWORDS: Low Energy, Reversible logic, Reversible Computing, Adiabatic Switching, Split-Level Charge Recovery Logic, Fredkin Gate. Quasistatic Switching.

Thesis Supervisor: Thomas F. Knight, Jr.
Title: Principal Research Scientist of Electrical Engineering

Acknowledgments

In consulting with Tom Knight throughout this work, many a times I was the recipient of the following advice
"... of course you can do it that way, but if *I* was doing it I would..."
and many a times I went my own way. On reflection, I wish that for the most part, I had not. When it comes to technical intuition of the kind that usually turns out to be true, Tom Knight seems to have an enviable supply. I'm thankful for having worked with him technically and grateful for having known him personally.

I wish to thank Gregory Papodopoulos and Kim Molvig for their commitment and support of this work.

I would also like to thank members of `toms-knights` group including `Sir` Philip Alvelda, `Sir` Mike Bolatsky, `Sir` Andre Dehon and `Sir` Tom Simon for their instrumental help during the design of SCRL-1.

Special thanks to Andy Boughton and Jack Costanza for their generosity, and friendship during the testing of SCRL-1 chip and throughout my hacking days at MIT.

This research is supported in part by the Defense Advanced Research Projects Agency under contract N00014-91-J-1698 and DABT63-92-C-0039

to my family

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 New Ideas, Old Physics

In principle, a computing engine need not dissipate any energy as shown in the work of Bennett, Feynman, and Landauer [3] [11] [27]. Although these authors approach the problem from different disciplines and use different physical as well as theoretical models, they all conclude that the transfer of energy through a dissipative medium dissipates arbitrarily small amounts of energy if that transfer is made sufficiently slowly.

In CMOS based circuits, the node voltages represent one of two logical values. During operation, the logical values of these nodes, and their voltage levels, repeatedly toggle between the two valid levels. Given the capacitance of these nodes, we can view performing computation using CMOS circuits as moving charge from one node to another. In other words, to perform useful work in CMOS, we are continuously forced to place and remove charge from various nodes in the circuit. Charge transfer between nodes of differing potentials is similar to shuttling heat between two heat baths at differing temperatures.

Thermodynamically, we know that all real energy transfer operations are invariably irreversible. Fortunately, however much an idealization reversible processes are, they are still of extreme importance since it is possible in many situations to achieve them to a very high degree of approximation.

As an example, it is possible to shuttle energy between two heat baths at different temperatures while losing arbitrarily small amounts of energy. This is done by inserting an infinite number of heat baths between the two original ones such that the temperature difference between any two adjacent baths becomes infinitesimally small. Shuttling heat packets reversibly between the baths at the two extremes following this gradual temperature staircase results in arbitrarily small energy loss.

In general, if lossless reversibility is to be achieved, a process must be carried out at a slow enough rate so that in effect, the system is always in equilibrium. In this light, the reversible process may be regarded as a series of *quasistatic changes* along a sequence of neighboring equilibrium states.

1.2 Quasistatic Switching

From the above we see that to asymptotically reduce the energy dissipation in CMOS all of the charge movements through the circuits must proceed quasistatically. To achieve this quasistatic operation, one has to guarantee absolute adherence to *two* conditions. The first is to guarantee that charge flow between any two nodes in the circuit occurs in a gradual and externally controlled manner. This means that we forbid any device in our circuit from turning on while there is a potential difference across it. It also means that once the device is turned on, the movement of charge through it must be done in a gradual and controlled manner so as to prevent a potential difference from developing. The second condition is to guarantee that the path followed by the charge does not contain any parts that violate quasistatic behavior. This means that the circuit should not contain any non-linear dissipative elements, *e.g.,* diodes. Once these conditions are guaranteed, the dissipation could be set to a level or asymptotically reduced through external control of the rate of charge movement. This is true since the two conditions assure quasistatic energy transfer and it is only through such techniques that asymptotic energy reduction is possible.

Currently the power consumption of CMOS circuits drops linearly with lower operating frequency. This means that the energy consumed per cycle is constant since the cycle is inversely proportional to frequency. Typically in CMOS, each cycle contains the same amount of computational work and on the average the same amount of charge shuttling. This suggests that in conventional CMOS, the energy consumed per charge movement is always constant. This is analogous to the the worst case scenario in our thermal example in which there were no additional intermediate thermal baths and the transfers between the two baths were done in one step. The reason for the high dynamic dissipation of conventional CMOS is the fact that charge transfer within them happens abruptly, *i.e.,* not quasistatically. The time constant associated with charging a gate through a similar transistor is RC, where R is the ON resistance of the device and C its input capacitance. However, the cycle time can be, and usually is, much larger than RC. An obvious conclusion is that energy consumption can be reduced by spreading the transitions over the whole cycle thus making them closer to quasistatic processes rather than "squeezing" them all inside one RC.

To asymptotically reduce the energy dissipation in CMOS all of the charge movements through the circuits must proceed quasistatically. To achieve this quasistatic operation, one has to guarantee absolute adherence to *two* conditions. The first is to guarantee that charge flow between any two nodes in the circuit occurs in a gradual and externally controlled manner. This means that we forbid any device in our circuit from turning on while there is a potential difference across it. It also means that once the device is turned on, the movement of charge through it must be done in a gradual and controlled manner so as to prevent a potential difference from developing. The second is to guarantee that the path followed by the charge does not contain any parts that violate quasistatic behavior. This means that the circuit should not contain any non-linear dissipative elements, *e.g.,* diodes. Once these conditions are guaranteed, the dissipation could be set to a level or asymptotically reduced through external control of the rate of charge movement. This is true since the two conditions assure quasistatic energy transfer and it is only through that that asymptotic energy reduction is possible. We want to state here that *there is no way to guarantee the two conditions stated above without employing reversible logic.*

In a CMOS circuit, we can always determine and control the potential on one side of a CMOS device since it is usually connected to a power supply rail. The potential on the other side, however, depends solely on the result of the computation. To perform a non-dissipative transition of the output, we must know the state of the output *prior to and during* this output transition. The reason for the need to know the previous state of a node before moving it is quite simple. Suppose that we needed to set the voltage on an internal node to V_{dd}. To do it quasistatically, we connect it to a rail that is currently at the *same* voltage, then we slowly ramp the voltage on the rail to V_{dd}, setting the voltage on the internal node to V_{dd} in the process. If the node was at GND, our rail would swing from GND to V_{dd}. In contrast, if the node was at V_{dd}, we would connect the node to a rail that is always anchored to V_{dd}. To determine which rail to connect to, we have to know the previous value as well as the desired final value of the node voltage. Furthermore, we have to hold on to this information throughout the transition. Stated more clearly, to non-dissipatively set the state of the output we must at all times have a copy of it. The only way out of this circle is to use *reversible logic*.

Recent and independent work by Hall [16] and Merkle [33] showed how to connect *Retractile Cascade* stages to eliminate the power dissipation in the latches. These two proposals are worthy of note since they are the only ones that conform to the two conditions of quasistatic operation outlined above, however both proposals were rather sketchy when it came to the details of the physical implementation of their proposed logic.

1.3 What is Reversible Logic

Reversible logic is a way to perform computation where information entropy is strictly conserved, or in some implementations mostly conserved. In our context we define constant information entropy as always keeping enough information around to be able to accurately retrace all the events, or steps, of the past. The main drive for implementing reversible logic is that of drastically reducing the energy dissipation of computing circuits. As is widely established by now, information entropy and thermodynamic entropy are linked. One cannot increase information entropy without dissipating energy. Fortunately the reverse is just as true. That is, if information entropy is not increased, it is theoretically possible, and as shown here practical, to perform computation while dissipating asymptotically vanishing amounts of energy if the computation is carried out asymptotically slowly. A trivial way of not increasing the information entropy of the system would be to store a copy of every transaction of the system forever. This obviously requires an infinite amount of storage space. A more practical approach to maintaining constant information entropy would be to undo, or reverse, the effect of a computation once the results of that computation had been utilized, *i.e.,* to perform reversible logic operations. To resign oneself to perform computation reversibly, is to empower oneself to perform them while dissipating orders of magnitude less energy than would be dissipated by conventional methods. We define the term asymptotic energy reduction as the ability to perform a computation while consuming asymptotically less and less energy as the computation is performed slower and slower with no theoretical limit on how small the consumption can get. If there was a limit, we do not consider that operation to be asymptotic in energy reduction. In other words, the theoretical line for energy dissipation associated with the process must in the limit reach zero.

It is the fact that dissipation could be reduced by orders of magnitude using reversible logic that convinced us of the need for it. We note here that reversible logic is less restrictive than conservative logic that was proposed by Fredkin and Toffoli in [13]. In addition to preserving the information content, conservative logic also preserves the total number of ones and zeros of the system. As it turns out, conservative logic simplifies some mechanical implementation of reversible logic, due to its dual polarity signaling. However, only reversibility is needed for reducing the energy consumption. In a sense our implementations share their reversible aspect with the reversible and conservative Fredkin gate. Our implementations however differ in their lack of conservation for ones and zeros.

Intuitively, and in contrast to the above, a circuit that has no means of computing the logical inverses of its functions has no means of preserving the information content of its nodes and hence should not be thought of as reversible logic.

1.4 Temporal Reversibility

A number of other proposals sought to lower the energy dissipation by charging and discharging the internal nodes of their circuits in a gradual and control manner. In their work, Koller and Athas [26] were largely concerned with reducing the energy consumption of bus drivers. Since in their circuits, the input to the bus driver was held stable during the SETting and RESETting of the gate, reversibility was in effect performed temporally and hence they were able to achieve asymptotic energy reduction for their bus driver without needing reversible logic. The key here is that the circuit holding the input stable during SETting and RESETting of the rails provided the “infinite storage space” within which the history of the computation is recorded. Koller and Athas however, correctly recognized and reported that when the entire circuit was considered, there was an unavoidable energy dissipation in the pipeline registers. This of course was a consequence of not being able to satisfy the first condition in the pipeline registers due to the absence of reversible logic. We do acknowledge that achieving asymptotic energy reduction in selective parts of the circuit as Koller and Athas have demonstrated could go a long way towards reducing the power consumption of a system. But we note however that the reduced energy consumption would have a lower bound.

1.5 Diode Based Proposals

Previously, we stated the importance of knowing the previous value of a node while we are affecting it in order to achieve quasistatic switching of that node. The previous value was needed so that we can make the correct decision of which rail to connect the node to in order to affect quasistatic charge transfer. A simple diode can however correctly make that decision for us. Recently two separate proposals have been forwarded that achieve energy reduction in CMOS circuits through using diodes. The first is by Denker et al. [9] while the second is by Hinman and Schlecht [17]. Using a diode however violates the second of the two condition we have stated above. The fact is that charge transfer in a forward biased diode is *not* a quasistatic process even if the charge transfer proceeds slowly.

This becomes evident when we trace the path of an electron through the diode. In a forward biased diode, the energy of electrons on the N side of the P-N junction is elevated by the forward voltage to a level that permits them to overcome the built-in potential barrier and diffuse over to the P side of the junction. Once on the P side, they *rapidly fall down* the potential down hill to equalize with the energy of the electrons on the P side. It is this rapid and uncontrollable falling down the potential hill that is not quasistatic in the P-N junction. The fact that the “height” of the fall is always constant irrespective of the rate at which charge is allowed to flow through the diode is the reason for the constant and non-linear value of the diode’s V_{be}. From this we see that charging a node with a capacitance C through a diode dissipates an amount of energy, E_{diode} that is equal to $CV_{dd}V_{be}$, where V_{be} is the forward potential drop of the diode. At best, diode based methods can reduce the energy consumption by a factor equal to V_{dd}/V_{be}. This puts a limit on the energy reduction ratio over conventional CMOS circuit, usually no more than 10. Further, with the current technology push to reduce V_{dd} for CMOS circuits, in some cases below 200mV [4], the energy saving factor of diode based circuits will only get smaller or disappear entirely.

1.6 Charge Sharing verses Quasistatic Operation

Some of the proposed low energy circuits techniques suffer from allowing charge sharing among the internal nodes, which violates the first condition for quasistatic operation.

I stress here that the construction of quasistatic circuits should at all cost avoid charge sharing among the internal nodes since with charge sharing, the energy saving over conventional CMOS could easily be wiped out for all but the simplest of circuits. The point behind quasistatic switching is to recover most of the energy that at one point was deposited on the internal nodes of the circuit. Except for simple circuits, the effective capacitance of the internal nodes of a circuit becomes comparable to the capacitance of the output node. To allow charge sharing to occur on the internal nodes, is to allow a large and undetermined amount of the stored energy to be wasted. This could be as high as 50% of the total charge leading to a poor energy saving factor of less than 2. Furthermore, the indeterminacy of energy loss nullifies the advantage of constant supply capacitance that is provided by dual polarity designs.

1.7 Logic Families and Universality

For a circuit technique to be classified as a computing logic family, it must be universal. This means that it should contain at least one member that is non-monotonic and it must support negation. This is usually overlooked by new proposals for low energy circuits. As we have illustrates in a paper published earlier [41] and will revisit again in in Section 3.2.1, initial attempts at quasistatic circuits usually need augmentation in order to support logical negation and hence be eligible to be considered a logic family. In practice new proposals that do not attempt to construct multiple stage pipelines with stages that are more complex than simple buffers or inverters do not detect the absence of universality in their "logic families". A powerful check to see if a new computing circuit technique is universal is to try and design a circuit that takes in a logical value at its input and that is able at a later stage in the pipeline to produce both the true and the complement copies of that input. Furthermore the circuit must be able to have them arrive simultaneously at a given stage in the pipeline. The power of this test is that in quasistatic circuits, one cannot insert an inverter inline with a signal to get its complement, as is frequently done in CMOS. An inserted gate would also have to be a controlled pipeline stage and that forces the proposed circuit technique to fail the test I have outline above. Experience will show that proposed logic families passing this test, are universal.

The subject of this work is to try and apply the principles of reversible logic to CMOS circuits to achieve full quasistatic operation throughout the system and thus significantly reduce its energy consumption.

1.8 Contributions of this Work

In this document I present a number of new techniques for constructing non-dissipative quasistatic CMOS circuits. We feel that these techniques have a number of distinct advantages which warrant their use in future circuits.

In Chapter 2 I start by examining the ways in which the energy consumed for each charge transfer in CMOS circuits is made arbitrarily small as the process proceeds quasistatically. Initially, I will start by examining the energy dissipation mechanisms in CMOS circuits. I will then attempt to analyze them more closely to identify ways in which charge transfer, and hence computation, can be done quasistatically. The discussion will include the dissipation in the computing circuits as well as the dissipation caused by the action of the semiconductor switches in the power supplies of the system.

In Chapter 3, I describe a number of early implementations of Charge Recovery Logic (CRL) circuits. The common property of all of these circuits is their ability to do computations quasistatically thus consuming arbitrarily small amounts of energy when clocked sufficiently slowly. The discussion will include both Fully-Symmetric CRL as well as N-Channel CRL. It will also show how to string multiple CRL gates in a *non-retracting* pipeline [41]. Both Fully-Symmetric and N-Channel CRL were abandoned however in favor of our more recently discovered and much improved form of CRL which is examined in Chapter 4.

In Chapter 4, I present a much improved family of CRL called Split-Level CRL (SCRL). This form uses 2 times as many devices as conventional CMOS, requires only one wire for every signal, and actively drives all outputs during sampling. Further, I will show how to construct Split-Level CRL circuits using only 2 external inductors for every chip. SCRL sereved as the corner stone of our research since we based our demonstration chip design on its techniques.

Conceptually, Split-Level CRL differs from earlier CRL in two ways. The first is the use of Split-Level voltages. The second is the elimination of the RESET devices and delegating the action of restoring the voltage on SET nodes to gates in the reverse pipeline. As in the previous chapter, the discussion will include how to string multiple SCRL gates in a *non-retracting* pipeline [42].

The circuit techniques in these new CRL and SCRL logic families rely on constructing an explicitly reversible pipelined logic gate, where the information necessary to recover the energy used to compute a value is provided by computing its logical inverse. Information necessary to uncompute the inverse is available from the subsequent inverse logic stage.

To verify the quasistatic operation and behavior of Split-Level CRL, we have fabricated and tested an 8×8 CMOS multiplier chip , labeled SCRL-1, that employed the circuit techniques of Split-Level Charge Recovery Logic. Chapter 5, I describe the design of this demonstration chip. Following that, in Chapter 6, I describe the measurement techniques and their results that verified the lower energy consumption of SCRL-1 as a consequence of quasistatic operation through reversibility.

To my knowledge, SCRL-1 is the first working implementation of a pipelined, reversible logic based, asymptotically zero energy circuit. As such, I am certain that there is a lot more for us to discover and refine than what we have reported so far. In Chapter 7 I try to give some suggestions about directions of future work that could further improve the applicability of SCRL. These include CAD and architecture issues, as well as the design of better quasistatic power supply switches. My opinion is that as far as quasistatic reversible computation is concerned, we've only just begun...

Chapter 2

Quasistatic Switching in CMOS

2.1 Introduction

The subject of this Chapter is to show ways in which the energy consumed per charge transfer is made arbitrarily small as the process proceeds quasistatically.

Initially, we will start by examining the energy dissipation mechanisms in CMOS circuits. We will then attempt to analyze them more closely to identify ways in which charge transfer, and hence computation, can be done quasistatically.

2.2 Energy Dissipation in CMOS

As is widely known, the internal energy dissipation of conventional CMOS circuits is attributable to three major components. The first is due to the static leakage currents between the terminals of MOS devices. The second is due to the brief short between V_{dd} and V_{ss} during switching which is caused by both N-Channel and P-Channel devices being simultaneously ON for a brief time during a swing of $V_{dd} \geq 2V_T$. The third is due to the transient current associated with charging and discharging the gate capacitance C through a device with ON resistance R.

2.2.1 Dissipation Due to Leakage

Dissipative leakage currents occur anytime different potential nodes of the circuit are separated by slightly conductive mediums. Such dissipative "sandwiching" is present in a number of locations in CMOS circuits. These locations can be grouped according to their leakage mechanisms into two groups. The first is the leakage due to the reverse current of PN junctions. The second is the subthreshold conduction current between the source and drain of any MOS device.

The reverse current of a PN junction depends exponentially on temperature. Hence operating at lower temperatures greatly lowers this form of dissipation. In addition, a number of reversed biased junctions that are currently used for device isolation will become unnecessary with the advent of silicon-on-insulator fabrication technology (SOI).

The case for subthreshold conduction is more complicated. Increasing the threshold voltage of the devices in the circuit reduces subthreshold conduction, thus lowering quiescent power consumption. Unfortunately, this increases the ON resistance of the devices as the difference between V_T and the supply voltages decreases. To maintain the same speed

Figure 2-1: Energy analysis model of CMOS circuits.

performance, the devices must be made wider leading to higher dynamic dissipation. We will discuss dynamic dissipation in Section 2.2.3. Here, we stress the fact the fact that in the case that subthreshold conduction becomes appreciable, we can trade some of it for higher dynamic dissipation. Having done that, this thesis will show how to significantly lower dynamic dissipation of CMOS circuits leading to lower overall energy dissipation.

2.2.2 Dissipation due to V_{dd} to V_{ss} Shorting

During switching, a short occurs between V_{dd} and V_{ss} caused by both N-Channel and P-Channel devices being simultaneously ON for a brief time for swings where $V_{dd} \geq 2V_T$ [5]. We can approximate this switching dissipation by

$$V_{dd} \times \frac{1}{2}\frac{V_{dd}}{2R} \times \frac{V_{dd} - 2V_T}{V_{dd}} \times 2RC \tag{2.1}$$

which for a typical $V_{dd} = 4V_T$ simplifies to $\frac{1}{4}CV_{dd}^2$. As V_{dd} drops below $2V_T$ this switching dissipation becomes negligible. Unfortunately, the transfer curve begins to exhibit hysteresis thus limiting the utility of the gate at these power supply voltages. In addition, operating with $V_{dd} \leq 2V_T$ increases subthreshold conduction and increases propagation delays.

In Chapters 3 and 4 we will see how this form of dissipation is completely eliminated in the family of CMOS circuits that we are proposing in this work.

2.2.3 Dynamic Dissipation

It is widely known that the energy dissipation that is associated with charging and discharging the node capacitance, C, of a CMOS gate through a device of ON resistance R is equal to $C\,V_{dd}^2$ per period for a rail-to-rail voltage equal to V_{dd}. We will rederive this result to gain an insight for the real reason of this dissipation. We use as our model the familiar circuit in Figure 2-1. The C in the figure is the capacitance of the driven node while the R is the ON resistance of the driving gate. We start with the capacitor voltage at zero. Using the voltage source we apply a step voltage of V_{dd}. This mimics the action of turning on a MOSFET that drives the gate of another. The current in this circuit follows $i(t) = \frac{V_{dd}}{R}e^{-t/RC}$ and hence the power consumed is $P = \frac{V_{dd}^2}{R}e^{-2t/RC}$. Integrating over the charging time we get

$$E_{dynamic} = -\frac{CV_{dd}^2}{2}\int_{t=0}^{\infty} e^{-2t/RC} = \frac{CV_{dd}^2}{2} \tag{2.2}$$

From the above we see that in order to charge the load to V_{dd} we need CV_{dd}^2 Joules of energy. Half of it dissipated in the resistor during charging, and the other half stored in the charged capacitor. The later part is not lost yet but would be if we discharge the capacitor in a similar fashion, *i.e.,* by a voltage step in the voltage source.

2.3 Quasistatic Charge Transfer

In essence the first two dissipation mechanisms discussed, leakage and V_{dd}-to-V_{ss} short, are related to the particular implementation of CMOS circuits. In general dissipation due to leakage is considerably smaller than dissipation by the other means. In addition, it could be made even smaller if needed, *e.g.,* operating at lower temperature or using SOI. Furthermore, we will illustrate how to eliminate the occurrence of V_{dd}-to-V_{ss} shorting using a number of circuit topologies in In Chapter 3 and Chapter 4. Elimination of the dissipation associated with the repeated charging and discharging of internal nodes, dynamic dissipation, is more complicated. This is because it relates directly to the movement of energy packets between nodes at different potentials, which is governed and limited by the laws of thermodynamics. It is also by far the dominant dissipative mechanism at typical operating frequencies. For this reason, we conclude that to achieve non-dissipative computation we must direct our efforts to reducing dynamic dissipation.

At this point we like to emphasize that the CV_{dd}^2 dissipation is a direct consequence of the way we perform the cycling of the load C and is not an irreducible minimum associated with charging and discharging a capacitive node. Charging a node to V_{dd} from 0 in a period T only requires a current, $i(t)$ such that

$$\int_{t=0}^{T} i(t)dt = Q = CV_{dd} \tag{2.3}$$

However, the energy dissipation is related to the integral of the square of the current

$$E = R\int_{t=0}^{T} i^2(t)dt \tag{2.4}$$

It therefore follows that minimum dissipation results if we charge the load using a current function $i(t)$ that minimized the integral in Equation 2.4 while obeying equation 2.3. Intuitively we can see that this minimum function is none other than

$$i(t) = Q/T = CV_{dd}/T \tag{2.5}$$

To prove that we add the perturbation $v(t)$ to Q/T such that $i(t)$ becomes equal to $(Q/T) + v(t)$. Using Equation 2.4 we calculate the energy dissipation as

$$E = R\left(\int_{t=0}^{T}(\frac{Q}{T})^2dt + \frac{2Q}{T}\int_{t=0}^{T} v(t)dt + \int_{t=0}^{T} v^2(t)dt\right) \tag{2.6}$$

For $i(t)$ to satisfy Equation 2.3, the second integral in Equation 2.6 must equal zero. With $v^2(t)$ always positive, we see that any perturbation of $i(t) = Q/T$ leads to increased dissipation. From the above we see that minimum energy dissipation results when the step function of the voltage source is replaced with a current source with a step of $I = CV_{dd}/T$ and which is turned ON for T seconds. This should not come as a surprise since as T gets larger, the process of charging and discharging the load with constant I becomes more quasistatic.

2.3.1 Energy Dissipation with Current Sources

In the previous section we stated that energy dissipation is minimized by using current steps instead of voltage steps. In this section we will calculate this minimum. The energy dissipated in charging the capacitor using a current step I for T seconds is

$$E = I^2 R \times T = CV_{dd}^2 \times \frac{RC}{T} \tag{2.7}$$

Therefore the energy dissipated in one cycle equals

$$E_{currentsource} = CV_{dd}^2 \times \frac{2RC}{T} \tag{2.8}$$

We see that with a current source, the energy dissipated is less than that with a voltage step by a factor of $2RC/T$ for $T > 2RC$. Since the minimum period for one cycle is $2T$ we see that dissipation with a current source *per cycle* is $CV_{dd}^2 4RCf$. And hence the power dissipation of this circuit becomes

$$P_{currentsource} = CV_{dd}^2 \times 4RC\underline{f^2} \tag{2.9}$$

The quadratic dependence of the power on frequency is in sharp contrast to the familiar linear dependence in conventional CMOS, $P = CV_{dd}^2 f$. Figure 2-2 plots the power dissipation associated with a the voltage step, solid line, and that of the current step, dashed line, as a function of operating frequency. Depending on the operating frequency, constant current cycling of capacitive loads results in orders of magnitude less power consumption when compared to constant voltage cycling method. We note here that quadratic dependence of power on frequency leads to linear dependence on frequency of the *energy consumed per operation*. This gives rise to the possibility of performing computation while consuming asymptotically zero energy.

2.3.2 Multiple Capacitive Loads

The above analysis applied to a single capacitive node. To be useful we must generalize the analysis to a number of simultaneously switching nodes as is the case in actual CMOS circuits. Ideally, each RC circuit that models a CMOS gate driving a capacitive load would have its own separate current source as shown in Figure 2-3. This is not practical with current technology.

Instead, we use the circuit topology in Figure 2-4. Here a single current source provides the current that is needed by all the loads. If the RC time constant is the same for all the branches of the circuit then the voltages on all the capacitors will rise in unison and the above circuit will simplify to one with only one equivalent capacitor equal to the sum of all the capacitors in the circuit and one resistor equal to the parallel combination of the resistors in the circuit. The magnitude of the current in Figure 2-4 being equal to the magnitude of the sum of currents in Figure 2-3.

In practice however, the RC time constants of the separate branches are different. Using a common source, the circuit branches with faster time constants will track the voltage of the current source more closely than branches with slower RC time constants. Under such conditions, the slower branches would lag behind the fast ones and hence the voltages on the loads of different branches would grow apart. With this in mind, the point at which we must turn off the current source is not very well defined. Turning the source off when

Figure 2-2: Normalized power dissipation plot for a voltage step, solid line, and for a current step, dashed line.

Figure 2-3: Model of CMOS circuit with separate driving current sources.

Figure 2-4: Model of CMOS circuit with common driving current source.

Figure 2-5: Voltage waveform at the terminals of a constant current source driving a capacitive load C.

the fast branches reach V_{dd} leaves the slow branches not completely charged and in time, the loads in the slow branches will pull the voltages of all the branches to some voltage below V_{dd}. Waiting for the slow branches to completely charge up results in voltages larger than V_{dd} on the fast branches and thus making the circuit susceptible to latch-up and/or breakdown in most VLSI technologies.

Provided that the capacitive loads are linear, the common current source could be replaced with voltage source that has the waveform shown in Figure 2-5. The initial and final jumps in the voltage of this source are equal to $V_{dd}RC/T$. We observe that for $T \gg RC$ the current source could be replaced by a voltage source that outputs a linear ramp such that

$$V_{source} = \begin{cases} tV_{dd}/T & \text{, for } 0 < t < T \\ V_{dd} & \text{, for } t > T \end{cases} \tag{2.10}$$

With this source, the current at the start of the ramp is not constant but exponentially builds to the desired constant level with RC time constant. The current also exponentially drops off as soon as the voltage of the source levels off at V_{dd}. Given enough time, the voltage on all the nodes would reach V_{dd} regardless of their branch time constant. Because of the above we choose this source as a good compromise for approximating the desired ideal current source driver.

2.3.3 Energy Dissipation with Voltage Ramps

In the previous section we saw how a voltage ramp can approximate a current source. Here we will examine the added energy dissipation of a voltage ramp resulting from its divergence from the ideal current source at the end of the ramp. For a voltage ramp, that follow Equation 2.10, the current $i(t)$, is equal to

$$i(t) = \begin{cases} I(1 - e^{-t/RC}) & \text{, for } 0 < t < T \\ I(e^{-(t-T)/RC} - e^{-t/RC}) & \text{, for } T < t \end{cases} \tag{2.11}$$

where $I = V_{dd}C/T$. To get the dissipation associated with charging a load with a ramp we integrate the power dissipated by the above current into the resistor R over the charging period. We now stress the following two observations. The first is that unlike the case in current sources, load charging in voltage ramps continues after the ramp reaches V_{dd}. For this reason, a voltage ramp must have a higher slope than the one in Equation 2.10 in order to yield the same effective charging time of a current source with $I = CV_{dd}/T$. The second observation is that carrying the time of integration until all transients have settled, *i.e.,* $t = \infty$, for a voltage ramp in Equation 2.10 we find that the dissipated energy is the same as for the current source, *i.e.,* $E = CV_{dd}^2 RC/T$.

From the above observations we see that the energy lost using a ramp equivalent to the current source, E_{ramp} is

$$E_{ramp} = CV_{dd}^2 \left(\frac{2RC}{T - nRC} \right) \tag{2.12}$$

where n is the number of RC time constants needed after the end of the ramp for the voltages on the nodes to reach V_{dd}, within acceptable tolerance.

As expected, the energy dissipation of the voltage ramp rapidly approaches that of an ideal current source, which is the minimum possible, as T becomes $\gg RC$.

2.4 Ramp Generators

We have seen how the energy dissipated in charging and discharging a capacitive load is directly proportional to the slope of the input ramp. In truth however, using a ramp does not reduce the power consumption of the whole system, it merely relocates it. If we are not careful about the design of the ramp generator, it is possible to dissipate in the generator much more energy than that saved in the circuit. Simple ramp generators continuously vary the ratio of the pull-up and pull-down devices in their driving stage to produce the required intermediate voltage values of the ramp. The ramp produced *will* lower the energy dissipated in the circuit it drives, but the constant current path through the pull-up and pull-down devices will waste much more energy.

Note that merely relocating the dissipation from the computing circuit to the supplies still has some advantages. One advantage is to increase packaging density in systems that are otherwise limited by heat removal constraints. In conventional computing circuits, electrical energy is supplied to the circuit by copper wires and waste heat is removed by other mediums, such as forced air or circulating refrigerants. Since copper wires can transfer energy with much higher power densities than other mediums, dense packaging in supercomputers is usually limited by energy removal constraints, not by energy supply constraints. Using quasistatic computing elements, even those with dissipative ramp generators, the same copper wire injecting energy into a dense package is the one used to remove a high percentage of it out of the package thus greatly simplifying cooling and increasing density. Another advantage is that we might be able to use more exotic devices or technologies in the power supplies than in our circuits. This is because power supplies have fewer components and hence their components do not need to be densely packaged or numerously and cheaply produced like circuit components.

If the concern is to lower the overall energy dissipation of the system however, then we have to consider the dissipation occurring in the ramp generators. Currently there are two ways of building ramp generators with little dissipation. The first approximates a ramp by generating a sinusoidal waveform. The second approximates the ramp with a stepping staircase waveform.

In the next sections we will examine the dissipation associated with using either of the two methods. In our examination, we will distinguish between dissipation that happens

Figure 2-6: The two places of system dissipation.

in the computing circuits, from the dissipation that occurs within the ramp generator. In essence, we divide our system according to the energy dissipation mechanisms into the two parts shown in Figure 2-6. We do this to emphasize the fact that the dissipation in the computing circuits is purely a function of the ramp shape, while the dissipation in the ramp generator depends on both the shape of the ramp and the devices used to construct the generator. If in the future we are able to invent a less dissipative generator, we can then calculate the minimum overall dissipation of the system relatively easily. In addition, treating the two separately will more clearly illustrate how one can trade more of one dissipation for less of the other.

2.5 Sinusoidal Ramp Generator

In this section we will examine the dissipation of a ramp generator that approximates the linear ramp with a sinusoid. The reason for this approximation is that it is easy to build energy efficient sinusoidal generators using inductors.

In this section we will show how to build a non-dissipative sinusoidal load driver. We will then calculate the energy dissipated in charging and discharging a capacitive load through conducting but slightly resistive device.

2.5.1 Circuit Dissipation for Sinusoidal Ramps

As we have assumed so far, we model the CMOS circuit performing computation by a resistor R in series with a load capacitor C. The model is based on a lumped element approximation where the R's and C's are the effective R and C as seen from the supply terminals of the chip. For appropriately sized circuits, *i.e.,* circuits where all the branches have the same time constant, we can assume that the gate voltages have roughly the same rise time and therefore we can approximate R by R_{device}/N and C by $C_{device} \times N$ for an N branch chip. Note that R and C depend only on N and the fabrication process technology. We saw that when driven by a linear voltage ramp, as shown in Figure 2-7-a, this equivalent circuit dissipates $CV_{dd}^2(2RC)/T$. A sinusoidal ramp generator replaces the voltage source with the inductor circuit as shown in Figure 2-7-b. To cycle a load capacitor starting at 0 volts through V_{dd} and back to 0, we

1. connect the RC circuit through the inductor to $V_{dd}/2$,
2. we keep the inductor connected until the current reaches zero, signaling a complete polarity inversion in the load capacitor,
3. we disconnect the rail from the inductor and connect it to the V_{dd} to compensate for leakage and noise,

Figure 2-7: Conventional and non-dissipative circuit analysis models.

4. and finally we reverse the above steps to return the load voltage to 0.

The inductor in the above circuit acts as an electrical "flywheel" that forces the shuttling of energy between the capacitor and the $V_{dd}/2$ supply. For now, assume the switch connecting the rail to the inductor is external to the chip. We will revisit this in a later section.

We now examine the dissipation of our proposed circuit. To simplify the algebra, we let $V_{dd} = +V_0$ and $V_{ss} = -V_0$ so that V_0 is equal to half the rail-to-rail voltage V_{dd}. Our R,L,C circuit is described by

$$\frac{d^2V_C}{dt^2} + 2\alpha\frac{dV_C}{dt} + \frac{V_C}{{\omega_0}^2} = 0 \tag{2.13}$$

where V_C is the capacitor voltage, $\alpha = R/2L$, and $\omega_0 = 1/\sqrt{LC}$. For the solution to oscillate, the circuit must be *underdamped*, requiring that

$$2\sqrt{LC} > RC \tag{2.14}$$

and we find that the frequency of oscillation, ω_d, is given by

$$\omega_d = \sqrt{{\omega_0}^2 - \alpha^2} \tag{2.15}$$

where ω_d is the frequency of operation and is equal to π/T. Since R and C are fixed for a given chip, we can only adjust L to affect ω_d. Examining the formula for ω_d we discover that ω_d steadily increases as L decreases up to a maximum, ω_{dmax}, and then sharply decreases as the circuit approaches the *critically damped* point. We find that

$$\omega_{dmax} = \frac{1}{RC} = \frac{1}{R_{device}C_{device}} \tag{2.16}$$

and the smallest inductance we would ever need, L_{min}, is found by

$$L_{min} = R^2C/2 = R^2_{device}C_{device}/2N \tag{2.17}$$

For $V_C(0) = V_0$ and $i_C(0) = 0$ we find

$$V_C(t) = V_0e^{-\alpha t}(cos\omega_d t + \frac{\alpha}{\omega_d}sin\omega_d t) \tag{2.18}$$

and

$$i_C(t) = V_0C\frac{{\omega_0}^2}{\omega_d}e^{-\alpha t}sin\omega_d t \tag{2.19}$$

Figure 2-8 shows a plot of normalized V_c as function of time and normalized ω_d. Initially $V_C(0) = 1$. With ω_d close to ω_{dmax} the voltage drops rapidly, dissipating most of the

Figure 2-8: Plot of $V_c(t, \omega_d)$ showing the effect of fast rise and fall times on energy loss.

capacitor energy on the way down. The reclaimed energy is insufficient to fully charge the capacitor back to the negative rail. As ω_d moves away from $\omega_{d_{max}}$ we in effect spread each oscillation over a longer period of time. We see from the plot that the capacitor retains most of its energy since V_C comes close to the bottom rail. With ω_d only an order of magnitude lower than the maximum, the capacitor recovers most of its energy.

The power dissipated in a single rail-to-rail swing for our gate consists of two components. The first is the power dissipated in R during the swing, E_s, which results in a final voltage that is lower than $|V_0|$. Using Equation 2.18 we find that

$$E_s = \frac{1}{2} C \; V_0^2 \; (1 - e^{\frac{-2\pi\alpha}{\omega_d}}) \tag{2.20}$$

The second is the energy lost in R while charging C to the rail to compensate for the lower peak voltage due to E_s. We will do this by simply connecting the line to the rail and losing some energy in R, E_c, during this process. We find that

$$E_c = \frac{1}{2} C \; V_0^2 \; (1 - e^{\frac{-\pi\alpha}{\omega_d}})^2 \tag{2.21}$$

adding these two terms and multiplying by two to allow for both directions of swing, we find that

$$E_{loss} = 2C \; V_0^2 \; (1 - e^{\frac{-\pi\alpha}{\omega_d}}) \tag{2.22}$$

per period.

Figure 2-9: (a) Linear plot of F_{saving} factor vs. ω_d. (b) Log-Log plot of F_{saving} factor vs. ω_d. In both plots ω_d is normalized to ω_{dmax}.

Using Equation 2.22, we compute the ratio of the power consumption per period of conventional CMOS to that of a sinusoidally driven gate, F_{saving}, as

$$F_{saving} = \frac{2}{(1 - e^{\frac{-\pi\alpha}{\omega_d}})} \tag{2.23}$$

Figure 2-9a shows a linear plot of F_{saving} near ω_{dmax}. In this region, the circuit's performance is close to conventional CMOS but improves rapidly with lower ω_d. As we get away from ω_{dmax}, the graph attains a nearly constant slope as shown in Figure 2-9b and $\omega_0 \simeq \omega_d$. Substituting $R/2L$ for α, π/T for ω_d or ω_0, we approximate $\frac{\pi\alpha}{\omega_d}$ with

$$\frac{\pi\alpha}{\omega_d} = \frac{\pi(R/2L)}{(\pi/T)} \simeq \frac{TRC\omega_0{}^2}{2} \simeq \frac{\pi^2 RC}{2T} \tag{2.24}$$

Expanding the exponential, we get

$$F_{saving} \simeq \frac{4T}{\pi^2 RC} \tag{2.25}$$

for one charging and discharging cycle.

We see that sinusoidal ramps are worse than ideal linear ramps, Equation 2.12, by a factor of $\pi^2/8$ owing to the sinusoidal, instead of the constant, nature of the current in the circuit.

2.5.2 Generator Dissipation for Sinusoidal Ramps

Examining Figure 2-7-b, we see that the only component that could dissipate energy in a sinusoidal generator is the switch. This includes energy dissipated because of the switch's non-zero on resistance as well as energy dissipated in the action of turning it on or off. The case we will analyze is when the switch is made out of MOS devices. The resistance of an N-Channel MOSFET, R_{ON}, is equal to

$$R_{ON} = \frac{V_{DS}}{I_D} = \frac{L}{W\overline{\mu_n}C_o\left[(V_{GS} - V_{TH}) - \frac{V_{DS}}{2}\right]} \tag{2.26}$$

where W and L are the gate width and length of the device, C_o is the gate capacitance per area, and $\overline{\mu_n}$ is the average mobility of the carriers in the channel [35]. In quasistatic switching, we always attempt to minimize V_{DS} to reduce dissipation so that in general $|V_{DS}| \ll |V_{GS} - V_T|$. With this in mind and substituting C_p, the gate capacitance, for WLC_o, we get

$$R_{ON} = \frac{L^2}{\overline{\mu_n}C_p(V_{GS} - V_{TH})} \tag{2.27}$$

The on resistance of a MOSFET depends on the difference between the gate voltage and the channel. In a sinusoidal generator, this voltage varies with time as the load charges. By using a pair of complementary devices for the switch we are able to minimize this dependence since the variation due to the N-Channel device cancels that due to the P-Channel device. This makes R_{ON} almost constant over the period of the ramp. R_{ON} now becomes

$$R_{ON} = \frac{L^2}{\overline{\mu_{ave}}C_s(V_{dd} - V_{THave})} = \frac{\tau_s}{C_s} \tag{2.28}$$

where $\overline{\mu_{ave}}$ is a weighted average of $\overline{\mu_n}$ and $\overline{\mu_n}$ and V_{THave} is the weighted average of the threshold voltages of the N-Channel and P-Channel devices. The energy dissipated in the switch during a single ramp swing has two components. The first is the dissipation due to charging and discharging the gate of the switch and is equal to $2C_sV_{dd}^2$ for a conventionally driven pass gate. The second is the dissipation due to the finite on resistance of the switch. From Equation 2.25 and Equation 2.28 we see that the total dissipation in the switch, E_{switch}, is equal to

$$E_{switch} = 2C_sV_{dd}^2 + \frac{1}{2}C_LV_{dd}^2\frac{\pi^2}{4}\frac{R_{ON}C_L}{T} \tag{2.29}$$

where C_L is the effective load capacitance of the computing circuit be driven.

Finding the minimum with respect to C_s, we find that the optimal gate capacitance of the switch is

$$C_s = C_L\frac{\pi}{4}\sqrt{\frac{\tau_s}{T}} \tag{2.30}$$

and the minimum dissipation in the switch becomes

$$E_{switch} = CV_{dd}^2\pi\sqrt{\frac{\tau_s}{T}} \tag{2.31}$$

The above analysis agrees with that reported by Koller and Athas [1] albeit with some modification because of differing definition of the process time constant of the switch, τ_s.

Adding the dissipation in the computing circuits, the total minimum dissipated energy in the system, E_{total}, becomes

$$E_{total} = C_LV_{dd}^2\left(\pi\sqrt{\frac{\tau_s}{T}} + \frac{\pi^2}{8}\frac{\tau_c}{T}\right) \tag{2.32}$$

where τ_c is the time constant for the circuit branches of the computing part of the system. We see that after accounting for the switch dissipation, the overall energy consumption of the system now drops as $1/\sqrt{T}$ instead of the thermodynamic limit of $1/T$. In general, $\tau_s \simeq \tau c$ when both the switch and the computing circuits use the same process technology. Since T is always greater than τ_s and tau_c for proper operation, we see that the switch dissipation always dominates the dissipation characteristics of the total system.

2.5.3 Nested Sinusoidal Drivers

The above analysis assumed that the switch is driven conventionally and hence dissipated $2C_sV_{dd}^2$ for every switching cycle. The factor of 2 is there because the switch has to turn on and off for both the rising and the falling parts of the driven signal. Koller and Athas [1] have suggested driving the switch with yet another sinusoidal ramp circuit which itself is driven sinusoidally and so on. They report that for N nested drivers, the minimum dissipation would follow $1/T^{(1-2^{-N})}$. They caution that since every driving stage have to be faster than the stage it is driving, we quickly get to the point at which driving the preceding stage sinusoidally, adds overhead but saves no energy. Add to that the fact that each switch must be smaller than the one it drives, which quickly limits the useful nesting when the switch reaches the minimum size device of the technology. They report that with current CMOS technologies, N would not exceed 2 or 3 unless T is very large "on the order of milliseconds or seconds".

Figure 2-10: Typical stepwise ramp generator circuit driving a capacitive load C_L.

2.5.4 Inductor Quality factor

So far we have assumed that the inductor in our RLC circuit enjoyed an infinite quality factor, Q. However, the Q's of commercially available inductors seldom reach higher than 100. The Q identifies the fraction of energy that is dissipated by the inductor in a RLC circuit during one cycle. Attempts to reduce the energy dissipation of the system below Q fold will fail as the inductor irreducible inductor dissipation becomes the dominant factor. This means that the maximum attainable energy saving factor is limited to about Q. We can dramatically improve this limit by using high-temperature superconducting coils. These usually have Q's in excess of several thousands. Unfortunately, such coils require cooling, typically by liquid nitrogen, which increases the cost as well as decreases utility in some application, e.g. portable equipment. The hope however is the discovery of superconducting material that will work at, or slightly below, room temperature. For a more detailed discussion of low temperature operation please see appendix A.

2.6 Stepwise Ramp Generator

Inductors are not the only way to produce a gradual voltage ramp. Another way to produce the gradual charge transfer from one potential to the other is to move the charge one small voltage step at a time. This mimics the situation with the two heat baths at differing temperatures separated by a number of heat baths at intermediate temperatures. We start with a load, C_L, at zero volts that needs to be charged to V_{DD}. We provide N voltage sources each with a voltage that is V_{DD}/N volts greater than the previous one. We also provide a switch connected between the load and each voltage source as shown in Figure 2-10.

To charge the load, we momentarily connect it to each voltage source, using the provided switches, in a sequence from the source with the lowest voltage, V_{DD}/N for N voltage steps, to the one with the highest value, V_{DD}. Reversing the switching sequence brings the load back to zero. Advantages of using a stepwise generator include elimination of inductors with their non-linear effects and more control over the rise and fall time of the ramp since this rise time is now not related to the load capacitance.

Following the same organization used for the sinusoidal case, I will first examine the dissipation of a computing circuit when driven by a stepwise ramp. After that I will examine the dissipation in the stepwise ramp generator itself. For the analysis, I'll use a MOSFET implementation of the stepwise generator.

Figure 2-11: Model of a CMOS computing circuit driven by a stepwise source.

2.6.1 Circuit Dissipation for Stepwise Ramps

To account for only the dissipation in the computing circuit, we use the model shown in Figure 2-11. The model consists of a load capacitor, C_L, representing the effective capacitance of the computing circuit and a resistance, R_{ON}, representing the effective ON resistances of the MOSFET's in the circuit paths to the loads. The driving voltage source in the circuit generates a staircase waveform starting at zero volts and rising towards V_{DD} with V_{DD}/N increments and with n RCR time constants between the steps.

We begin by assuming that the time provided between each step is much longer than the RC time constant of the circuit in Figure 2-11. This will insure enough time between steps for the circuit to settle. From Equation 2.2 we see that each step in the voltage source dissipated $C_L \Delta V^2/2$ joules of energy where $\Delta V = V_{DD}/N$. Since there are N steps during the rise and another N steps during the fall of the ramp voltage, the total dissipation a computing circuit that is driven by a stepwise ramp, $E_{stepwise}$,

$$E_{stepwise} = 2N \times \frac{C_L V_{DD}^2}{2N^2} = \frac{C_L V_{DD}^2}{N} \tag{2.33}$$

The above result should not come as a surprise. Given that the time between steps is always fixed at the value that allows enough time for the most part of the transients to settle, *i.e.,* a few RC's, we see that increasing N is identical to increasing T. Said differently

$$N \simeq \frac{T}{nRC} \tag{2.34}$$

and we see that Equation 2.33 is very analogous to Equation 2.8.

2.6.2 Generator Dissipation for Stepwise Ramps

Svensson and Koller [38] have studied the dissipation in a MOSFET implementation of a stepwise ramp generator. Their generator consisted of a number of large capacitors with each charged to one of the step voltages and each separated from the output by a NMOS switch. Shorting these large capacitors to the output via the MOSFET devices in ascending order produced the stepwise ramp. In their work they show that the optimal number of steps is

$$N_{opt} = \sqrt[3]{\frac{T}{4m\overline{\rho}}} \tag{2.35}$$

where m is the number of process time constants between steps and $\overline{\rho}$ is the weighted average of $(R_{ON} \times C_{gate})$ of the MOSFET devices used. Given this, they calculate the minimum energy dissipated in the control of these switched and due to their non-zero ON resistance,

E_{opt}, to be

$$E_{opt} = \frac{3}{2}\sqrt[3]{\frac{4m\overline{\rho}}{T}}C_L V_{DD}^2 \qquad (2.36)$$

We note that even though the dissipation in the computing circuit driven by a stepwise generator followed $1/T$, the energy dissipated in a MOSFET implementation of the stepwise ramp generator optimally follows $1/\sqrt[3]{T}$. This means that the energy dissipation by the overall system would track $1/\sqrt[3]{T}$. This is worse than the performance of a sinusoidal generator using a MOSFET switch which tracked $1/\sqrt{T}$.

2.7 Alternate Power Switches

The above derivations assumed that the power switch, or switches in the stepwise case, was built out of the same device technology as that of the computing circuit. In both the sinusoidal ramp generator and the stepwise generator cases, the energy dissipation of the computing circuit followed the theoretical line of $1/T$. We saw how this impressive energy saving behavior deteriorated, to $1/\sqrt{T}$ for the sinusoidal case and to $1/\sqrt[3]{T}$ in the stepwise case, due to the non-ideal properties of the MOSFET's used in the generators. However, the economics of the VLSI computing circuit of the system are very different from those of the power supplies, or ramp generators. The technologies for both the computing circuits and the power switches in the ramp generators need not, and as we will show should not, be the same. In the coming analysis we will concentrate on the power switch for the sinusoidal ramp generator. Even though the proposed methods could be equally applied to the stepwise case, we will focus on the sinusoidal generator because of its more attractive energy dissipation curve.

To illustrate the point of using alternate technologies in the ramp generators, let us assume that $\tau_s < \tau_c$. We rearrange Equation 2.32 to find the value of normalized T, T/τ_c, at which the energy dissipation in the switch, E_{switch}, becomes equal to dissipation in the circuit $E_{circuit}$. We find that

$$\frac{T}{\tau_c} = (\frac{\pi}{8})^2\frac{\tau_c}{\tau_s} \qquad (2.37)$$

Note that the energy savings following $1/T$ start with T/τ_c increasing beyond $\pi^2/4$ as demonstrated by Equation 2.25. Before T increases beyond the point indicated by Equation 2.37, $E_{circuit}$ dominates the dissipation and the consumption follows the $1/T$. Increasing T after crossing this point, E_{switch} rapidly dominates the dissipation and the consumption follows $1/\sqrt{T}$. From the above we see that for every order of magnitude that τ_c is larger than τ_s we get an additional order of magnitude through which T can increase while maintaining an energy saving factor that is linear with T. That is an additional decade drop in operating frequency during which E_{switch}, with its inferior $1/\sqrt{T}$ dissipation factor, are still insignificant.

2.7.1 MODFET Switch

To illustrate how useful the above concept could be, we examine the possibility of using a Modulation-Doped FET device, MODFET, for the switch [7, 8]. For a 2μm process through MOSIS, the fabrication house reported a frequency of 35MHz for a 31 stage ring oscillator. Lee, Lee, Miller and Anderson [30] report a frequency of 1.36GHz for a 25 stage oscillator in 1983. Since ring oscillator frequency is linearly dependent on the process time constant, we see that if we use a MODFET having the reported parameters for the switch while using the relatively inexpensive 2μm process in the computing circuits, we can get a ratio of 31 for τ_c/τ_s. This means that for the first 31 fold increase in T, the total energy consumption of the system would follow the thermodynamic limit of $1/T$.

The above illustrates the payoff of mixing advanced technologies for the switch with conventional low cost technologies for the computing part of the system. The suggestion to use MODFET's was to merely serve as an example. Other devices with much lower τ_c exist, specially those with superconducting behavior. Such non-dissipative devices could result in a total system dissipation that follow $1/T$ throughout the entire operating frequency range. Unfortunately, the details of their implementation as well as the operating limits of these superconducting devices are not familiar to this author.

2.7.2 Bipolar-MOSFET Switch

So far we have concentrated on the use of field-effect-transistors for the power switch in the ramp generator. We favored FET's over other devices because of their superior static properties of requiring no additional energy to keep them switched on, and of having a nearly linear ON resistance irrespective of the current through them. We have seen how to reduce the undesirable dissipation of these switches by employing alternate technologies, such as MODFET's, that are highly suitable for the switch. Regardless of the technology used in the fabrication of these devices, the overall dissipation will always track the curve predicted in Equation 2.32. The restriction in Equation 2.32 followed from our assumption that the gate of the switch is controlled conventionally. The attempt in Section 2.5.3 to reduce the dissipation by dropping this assumption yielded little improvement. This comes from the fact that an FET switch has a linear non-regenerative action relating its gate potential to its R_{ON} when V_{DS} is small. Therefore, every nested stage must run more than the minimum 2 times faster than the stage it is driving in order for the circuit to reduce the overall energy dissipation. Our hope is to find a device with *dynamic* amplification properties under low V_{DS} so as to further reduce the dissipation associated with controlling the switch.

A bipolar transistor has that property. The collector current in a bipolar transistor depends exponentially on the base-emitter voltage, V_{BE} and hence the transistor resistance can change by orders of magnitude in response to a small change in input voltage, V_{BE}. Unfortunately, bipolar transistors have two properties that usually exclude them from low power systems. The first is the constant supply of base current, and thus constant dissipation, that is required to keep them turned on. The second is the non-linear and almost constant potential drop, V_{SAT}, across them while they are in saturation. In other words, a bipolar transistor exhibits good dynamic properties when compared to a MOSFET, but has undesirable static power dissipation. We therefore propose a hybrid switching device that aims to make use of the superior dynamic behavior of the bipolar transistor *and* the much desired static properties of a MOSFET. This hybrid switch consists of a bipolar device in parallel with a MOSFET.

To yield an improvement over the single MOSFET scheme, we will control the MOSFET switch by a gradual voltage ramp. The control current for the bipolar transistor, I_B, will be set by a feedback control circuit such that the collector-emitter voltage, V_{CE}, is kept slightly higher than the saturation voltage of the device. This will keep the voltage drop across the transistor to a minimum while maintaining h_{FE} at its nominal level. Low V_{CE} leads to least dissipation in the collector-emitter path, while high h_{FE} leads to minimum control dissipation by minimizing the required I_B for a given I_C.

The switch action proceeds as follows. First the feedback circuit of the transistor is turned on. At the same time we start charging the gate of the MOSFET by the voltage ramp. Initially, the bipolar transistor would carry all the current. As the gate voltage of the MOSFET rises, the voltage across the MOSFET will at some point drop below V_{CE} of the bipolar transistor. This is caused by the falling ON resistance of the MOSFET as it is driven ON stronger and stronger. With V_{DS} dropping below V_{CE}, the MOSFET will carry most of the current and the dissipation through the collector-emitter path of the bipolar transistor is eliminated. In addition, the feedback circuit controlling the bipolar transistor, in an attempt to keep V_{CE} at the programmed value, will continue to reduce the base current of the bipolar transistor, quickly forcing it into cut-off. In our RLC circuit, the current builds up from zero at the beginning of the switching cycle, rises to a maximum level in the middle, and drops down to zero at the end of the cycle. In the hybrid switch above, the bipolar transistor carries most of the current only at the beginning and tail end of the cycle. The hand-off of current from the bipolar transistor to the MOSFET occurs exactly at the time that the potential drop across the device carrying the current exceeds the drop across the other device. The hand-off is made more abrupt by the feedback circuit. What is important to note here is that the constant current, as governed by the inductor, is automatically directed by the circuit to flow through the least dissipative component of the hybrid switch. This, while the gate of the MOSFET is controlled *gradually* to minimize dissipation.

We can calculate the energy dissipated in a bipolar switch that conducts a charge Q as

$$E_{bipolar} = Q(V_{CE} + \frac{V_{BE}}{h_{FE}}) \tag{2.38}$$

For silicon, V_{BE} is is a function of doping and intrinsic carrier concentration and is usually fixed at around 0.6V. V_{CE} is lowest when the device is in saturation. For a transistor with equal doping concentrations in the emitter and collector regions, V_{CE} can theoretically approach zero when the transistor is pushed into deep saturation thus eliminating the dissipation due to collector-emitter voltage drop. Unfortunately, h_{FE} becomes close to 1 when the device is in deep saturation and the dissipation in the base-emitter junction dominates. Since h_{FE} exponentially drops when approaching saturation, it becomes evident that minimum dissipation will occur when the device is operated closer to the edge of the active region rather than in deep saturation. Under such conditions, it is possible to lower V_{CE} below V_{BE} while maintaining negligible base-emitter dissipation due to large h_{FE}.

2.7.3 Micromechanical Switch

For our purposes, the best switch is the one that has the lowest activation energy for a given energy transfer through it. That is the ratio of the energy needed to control the action of the switch to the energy that it is capable of conducting is minimal. In addition, we hope that this switch has a sharp turn-on and turn-off curves as a function of control energy.

Electromechanical relays have what could be the lowest value for this ratio. This results from their very low ON resistance due to their metal contacts. The problem with regular relays is that they are slow. However, by soliciting the help of micromachining, it might be possible to make an electrostatic switch with the properties we are seeking. Those properties include speed, low activation energy dissipation, very sharp turn-on, and low ON resistance. Given this collection of desirable properties, it becomes important to investigate

Figure 2-12: Schematic diagram of a simple Electroquasistatic Micromechanical switch.

the feasibility of using micromechanical structures for our switching purpose. By using metalized contacts, such as aluminum, we are confident that the ON resistance of these switch is orders of magnitude lower than that of a power MOSFET. To yield improvement over MOSFET's, we have to make sure that the activation energy of these micromechanical switches do not exceed that for a MOSFET by the same orders of magnitude. We believe that this is the case.

Driven elecrostatically, a micromechanical relay is quite similar to a MOSFET. In the case of the MOSFET, charge is deposited on the gate of the device to turn it on. Likewise, for a micro switch, charge is delivered to an electrode causing an electrostatic force to move the arm of the switch as shown in Figure 2-12. We know that if the MOSFET is driven conventionally, it will dissipate $C_{gate}V_{DD}^2$ energy for every toggle. Likewise we anticipate that a micromechanical switch that is driven conventionally would dissipate, E_{micro},

$$E_{micro} = C_p(x)V_{on}^2 \tag{2.39}$$

where C_p is the activation plate capacitance of the switch and V_{on} is the electrostatic plate potential required to turn the switch on.

Note that in the case of a micromechanical switch, the capacitance is a function of the lever's position. Naively, one can think that the activation voltage is the voltage required to move the lever from its rest position all the way to its contact position, *i.e.,* V_{on}. More accurately, one would continuously apply a bias voltage, V_{bias}, to the plate of the switch such that the lever is as close to the contact position as possible without actually making a contact. Since a constant bias does not dissipate any energy, we see that the minimum activation dissipation for a micro switch is more accurately described by

$$E_{micro} = C_p(x = contact)(V_{on} - V_{bias})^2 \tag{2.40}$$

The dissipation indicated by Equation 2.40 above is much lower than that predicted by Equation 2.39. It is here that the abrupt turn-on and turn-off properties of a micromechanical switch come into play. This is demonstrated by the fact that we are unable to do the same biasing for a MOSFET since the ON resistance of the device drops linearly with increasing gate voltage for a low V_{DS}.

Ideally, V_{bias} will equal V_{on}. This however provides no contact force that is necessary for reliable current flow. We therefore suspect that under ideal conditions, $(V_{on} - V_{bias})$ will dominantly be a function of contact force and process evenness.

As desirable as abrupt switches are for low dissipation, they could be troublesome in inductive circuits. In these circuits, the gradual increase in the ON resistance of the switching device provides a place into which the inductive currents in the circuit could dissipate. Without it, there is a potential for destructive arcing and current crowding to occur. Fortunately, the switch in a sinusoidal ramp generator is always timed such that it makes or breaks at precisely the moment when the current in the circuit is zero. Furthermore, to make the timing less critical, we propose to parallel this micromechanical switch with a very weak MOSFET. We time the MOSFET so that it turns off only after the micro switch breaks. This, in addition to the nearly zero current during switching off, will greatly reduce the undesirable inductive effects. For the least dissipation to occur, we turn this MOSFET ON anytime *after* the micro switch makes.

2.8 Zero Energy Computing and Reversibility

The discussion throughout this chapter have focused on gradually charging and discharging the nodes of our computing circuit in order to reduce energy dissipation. We have seen that in the ideal case, the dissipation in the computing circuit followed RC/T. At this point we might be led to believe that achieving asymptotically zero energy computing depends only on our ability to build "ideal" ramp generator. As we shall see, this is not entirely correct.

In conventional CMOS, the output node is forced to the logic level representing the result of an operation irrespective of the previous value of the node. In the case that the new output value differs from the old one, the circuit will dissipate $CV_{DD}^2/2$. As long as we are willing to dissipate this amount of energy, the previous value of the node is of little importance. Quasistatic, low energy, switching however depends on the *gradual* charging and discharging of the nodes. At the end of a computing step, the output node must be set to a logic value that reflects the outcome of an operation. Depending on the previous value of the node and the new forced value we get four case. The are a 1 going to a 1, a 0 going to a 0, a 1 going to a 0 and a 0 going to a 1. In a quasistatic computer we connect the nodes representing the first two case to the corresponding V_{dd} and ground. For the third case, we connect the node to a rail that is gradually swinging from 1 to 0 and for the last case we connect the node to a rail that is gradually swinging from 0 to 1. We rely on the old value of the node when determining what to connect it to in order to quasistatically set it to the new value. In otherwords, to quasistatically switch a node, we need to know its previous value *before and throughout* the gradual swing of the altering rail, and herein lies the problem. As the rail starts to swing, it is in effect destroying the piece of information it needs throughout its transition. Naively providing temporary storage for this bit of information relocates the problem to the time or reusing this temporary storage. Without this knowledge it is possible to accidently short a node to a rail at a different potential leading to RC governed discharge time and the familiar $CV_{dd}^2/2$ dissipation.

From the above, we see that achieving quasistatic charge transfer is only part of the solution in reaching asymptotically zero energy computing. The remaining bulk of this thesis will illustrate how to solve the other part of the problem through the use of *Reversible Logic*.

2.9 Summary

In this chapter we have shown that it is possible to shuttle charge through resistive mediums while dissipating vanishingly small amounts of energy by using slow rising voltage ramps. We have examined a number of methods to produce these voltage ramps and analyzed the amount of energy that is dissipated in each method. Finally, we have indicated that quasistatic charge transfer alone is not sufficient to result in asymptotically zero energy computing. In the coming chapters we will see how non-dissipative computing invariably leads to reversible computing and that thermodynamic entropy and information entropy are strongly linked.

Chapter 3

Early Implementations of Charge Recovery Logic

3.1 Introduction

In this chapter we will describe a number of possible implementations of charge recovery logic circuits. The common property of all of these circuits is their ability to do computation quasistatically thus consuming arbitrarily small amounts of energy when clocked sufficiently slowly. That is, their energy consumption drops linearly with frequency. Consequently, their power consumption drops quadratically with frequency.

3.2 Fully-Symmetric CRL Implementation

This section describes our first implementation of charge recovery logic. Even though we think that the advantages of later implementation make this one unfavorable, we nevertheless include it in the hope that some of the ideas in it find a place in future developments.

3.2.1 Fully-Symmetric CRL Gate

In describing our CRL gate we start with a conventional CMOS gate and gradually modify it to produce a gate with all the needed properties. We use the implementation of a NAND gate as an example. Figure 3-1 shows a first attempt at a CRL gate next to a conventional CMOS NAND gate. Here we discard the pull up part of the CMOS gate and replace each N-Channel device with a CMOS pass gate. In addition, we replace the ground rail with a swinging rail. Initially, the rail and all the nodes of the gate are at logic 1, V_{dd}. After the inputs settle, the rail swings from a logic 1 to a logic 0. The output F now represents the value $\overline{A \wedge B}$. The output of this gate can now be used by subsequent stages. Since we use CMOS pass gates for all our switches, we need to have the complement of the output available as well. To generate the complement, we duplicate our circuit but connect it to a rail which swings in the opposite direction, as shown in Figure 3-2. We call the rail that rests at V_{dd} and swings towards ground the *bottom rail* and the one that rests at ground and swings toward V_{dd} the *top rail*.

For reasons that will become evident later, we require that when this circuit is in its reset state, *i.e.,* when the rails are in their rest state, the outputs are at a level that turns off any pass gates they control in a subsequent stage. For this reason we see that the

Figure 3-1: Conventional NAND gate and an early attempt at a CRL NAND gate.

Figure 3-2: Modified gate with dual-rail added.

Figure 3-3: The universal CRL NAND gate.

outputs controlled by the top rail can only drive inputs of N-Channel devices. Similarly, the outputs controlled by the bottom rail can only drive inputs of P-Channel devices. We acknowledge this fact by labeling the outputs with the subscripts N and P. Unfortunately, the gate we have built so far is not universal as we cannot perform negation. This is because a logic 1, TRUE, in this circuit is no longer represented by a voltage level but by the event of the output actually swinging. A quick investigation shows that a swing can only force a subsequent swing. To produce the complement of a swing we borrow an idea from conventional CMOS and augment each half of our gate with a complementary network to produce the universal gate shown in Figure 3-3. Note that so far, we need four times as many devices as a conventional CMOS gate.

To reduce confusion, we will no longer explicitly show the N-Channel and the P-Channel devices of the pass gates. In addition, we will assume that the single line labeled F represents the two-line pair composed of F_P and F_N and that the single line labeled $\overline{F}$ represents both $\overline{F_P}$ and $\overline{F_N}$. Further, we note that a pass gate receiving the output $\overline{F}$, is ON when $\overline{F}$ is at a logic 1. Before we redraw our gate, we will add some functionality that will be needed when we connect copies of this gate in a non-dissipative network. In our gate so far, we identify all the inputs as SET inputs and label them with the subscript S. We then add a second pass gate in parallel with each pass gate already in the circuit. We identify the inputs of these new pass gates as the RESET inputs and label them with the subscript R. We show the completed CRL NAND gate with all modifications in Figure 3-4.

It should be obvious how we can build any logical function based on the above techniques in a way similar to conventional CMOS, except for the additional redundancy.

At this point we note that in this implementation of CRL, there is always a pair of output wires that do not swing during the SETting and RESETting of the gate. Those

Figure 3-4: Completed CRL NAND gate with pipelining support.

wires rely on their node capacitance to maintain their voltage. Unfortunately, it is possible under some inputs for the same output wire pair to continue to be the non-swinging pair for a long time. Since non-swinging wires are always floating, it is possible after some time for the voltage on these nodes to wander due to leakage or capacitive coupling. To prevent this from happening, we add a number of transistors that clamp these floating nodes to the correct supply rail as shown in Figure 3-5. With these additional transistors, we see that all the output wires are actively driven at the end of the SETting swing and therefore provide for periodic refreshing of the voltages on the floating nodes.

So far, our CRL gate asymptotically requires eight times as many devices as conventional CMOS as illustrated by the 2-input AND/NAND gate example. For circuits that require complementary outputs, such as address decoders, the redundancy factor may be somewhat less.

3.2.2 Reversible Pipeline of Fully-Symmetric CRL

In this section we show how to connect CRL gates, or stages, in a non-dissipative pipeline. The main purpose of this method of interconnection is to provide the RESET inputs to each gate at the correct time. We build the pipeline out of copies of an abstraction box shown in Figure 3-6a.

We think of this box as containing a parallel set of CRL gates performing any logical function of an arbitrary number of inputs. Symbolically, the output of the box represents a bundle containing the outputs of the CRL gates internal to the box. The box has two input branches. One is the SET branch and identifies a bundle containing all the SET inputs of the gates internal to the box, the other is the RESET branch and identifies a bundle containing all the RESET inputs. The function computed by the box is identified by the

Figure 3-5: First CRL implementation with output cross-coupled clamps.

Figure 3-6: (a) CRL abstraction box. (b) Timing of the four clock rails.

Figure 3-7: Non-dissipative multi-stage pipeline connection.

letter in the center of the box. Finally, indicated at the lower corner of the box is the clock phase used to control all the CRL gates internal to that box. A clock of ϕ_1 indicates that the top rail is connected to ϕ_1 and the bottom rail to $/\phi_1$, while a clock of $/\phi_1$ indicates that the top rail is connected to $/\phi_1$ and the bottom rail to ϕ_1.

Using this abstraction, Figure 3-7 illustrates how CRL gates are connected to produce a non-dissipative pipeline. The timing of the four clock lines is shown in Figure 3-6b. Note that the box with a function F^{-1} performs the inverse operation of the box with a function F. To SET a box, all the SET inputs must be valid and stable and all the RESET inputs must be idle, *i.e.,* they come from a box that is currently in RESET, so that all the RESET pass gates are OFF. With these inputs, swinging the clock rails of the box away from their rest level will SET the box. To RESET the box, the rails are returned to their rest levels while the SET inputs are idle and the RESET inputs are active and stable.

To follow the operation of the circuit we start with ϕ_1 and ϕ_2 at their rest state and assume that the pipeline has been operating for some time. We follow the propagation of the input a_0 only, even though other parallel activity is going on. From the states of the clocks we see that box F_1 is RESET and its RESET inputs are idle as well. Swinging ϕ_1 SETs F_1 and computes $F_1(a_0)$. Swinging ϕ_2 now SETs F_2 and F_1^{-1} and produces $F_2(F_1(a_0))$ and $F_1^{-1}(F_1(a_0)) = a_0$ respectively. Now swinging ϕ_1 to its rest level RESETs F_1, produces $F_3(F_2(F_1(a_0)))$ and $F_2^{-1}(F_2(F_1(a_0))) = F_1(a_0)$. The circuit is now ready to safely RESET boxes F_2 and F_1^{-1}. One can see that we can continuously drive a new input into the network every ϕ_1 and successfully operate the pipeline in a non-dissipative fashion. In addition, this pipeline can have any arbitrary number of stages and still be driven entirely by ϕ_1, ϕ_2 and their inverses only.

There remains one problem however. At the end of the pipeline the RESET input to F_5^{-1} is not available and hence resetting this box is dissipative. Furthermore, it could not be generated, as this is the place where reversibility is broken. We can however, restore reversibility here through brute force by connecting to the end of this pipeline a mirror, and an inverse, image of itself. The missing input at the end of this extended pipeline that is needed to reverse the last inverse box is now simply a_0. With this topology, we can proceed without any dissipation by continually supplying delayed copies of the input to the pipeline at the inverse input on the far right. The technique of connecting an inverse network to the forward network was previously used in [13] and [11] to eliminate dissipation through recycling the intermediate *garbage* that results in conservative logic.

Admittedly, the above solution is more of theoretical than practical interest. If reversibility needs to be broken, that is, when information loss cannot be avoided, then some dissipation will occur for every lost bit of information. For these situations, we can reduce the dissipation by ending the pipeline with two identity boxes, $I(a) = a$, and use the out-

Figure 3-8: Last pipeline stage connection for dissipation reduction.

put of the lower identity box to reset itself as shown in Figure 3-8. Closer examination shows that the dissipation is $\frac{1}{2}CV_T^2$ per bit per cycle as opposed to $\frac{1}{2}CV_{dd}^2$ for conventional gates. Since the output of an identity box is the same as the input, the resetting swing proceeds normally until the output levels are insufficient to keep the appropriate pass gates on. Because of this, some internal nodes will have a potential that is one V_T away from their reset levels. The next input to the gate will short this potential difference resulting in $\frac{1}{2}CV_T^2$ dissipation per bit. Note that that we only pay this penalty at the last stage of a long pipeline.

3.3 N-Channel CRL

The circuit described in the previous section requires 8 times as many devices as conventional CMOS circuits. The following is a description of alternate implementations that require 2-4 times as many devices only. The tradeoff is that the following circuits require more swinging rails, 6-8 rails, than what the previous implementation needed, 4 rails.

3.3.1 N-Channel CRL Gate

The main idea behind these new circuits is that they have two sections. A front-end section that has only N-Channel devices, and a back-end section that uses pass gates made out of N-Channel and P-Channel devices. The new circuits require two different rails that are not the complement of one another. The first rail, we call the slow rail, controls the front-end section of the circuit. The second, we call the fast rail, controls the back-end of the circuit. Figure 3-9 shows an implementation of a NAND gate using this new techniques.

In the front-end section, all SET N-Channel devices, identified by the letter "S" in the figure are paralleled by RESET N-Channel devices, identified by the letter "R". The pass gates of the back-end section, identified by the letter "P" do not have RESET pass gates. While in the RESET state, all the outputs and internal nodes of the front-end section are at V_{ss}. The slow rail is at V_{ss} as well. In addition, all the outputs and internal nodes of the back-end section are at V_{ss}. The fast rail would be at V_{ss} as well. We assume that all the SET and RESET inputs are at V_{ss}. The circuit is now ready to accept new input on its SET lines. After the SET inputs become valid and stable, we gradually swing the slow rail from V_{ss} to V_{dd}. At the end of the swing, some outputs of the front-end section would remain at V_{ss} while some would swing to $(V_{dd} - V_T)$ depending on the input value and the implemented function. Note that we generate the true and complement of every signal in

Figure 3-9: Schematic diagram of a 2-Input AND/NAND N-Channel CRL gate.

the front-end section so that we could drive both sides of the pass gates in the back-end section. One of the main purposes of the back-end is to regenerate the rail-to-rail logic levels at the output of the gate that could not be generated by the N-Channel only section. After the front-end SETs, we swing the fast rail from V_{ss} to V_{dd}. Depending on the computed result, we could have the pass gates of the back-end set ON or OFF. Those that are set on by the front-end, having their P-Channel side at V_{ss} and their N-Channel side at $(V_{dd} - V_T)$ will swing the outputs they are driving to V_{dd}. The pass gates that are off would have the gates of their P-Channel devices at $(V_{dd} - V_T)$ and the gates of their N-Channel devices at V_{ss}. The N-channel side that is off will remain off during the entire swing of the fast rail, the P-Channel could start conducting just before the end of the swing. However, assuming that the threshold voltages of the P-channel and the N-Channel devices are roughly equal, except for the sign, the channel-to-gate capacitive coupling of the P-channel devices will raise the voltage at their gate during the low to high swing of the fast rail thus insuring that they will remain off. Note that the bootstrapping here is used to shut off devices and not to recover the V_T drop of the front-end. For his reason, the minimal of coupling would still result in keeping the device off and would lead to proper operation. Note that the outputs of the back-end are now rail-to-rail and could drive the front-end of a subsequent stage without V_T degradation.

To RESET the circuit, we wait until the RESET inputs are active and until the SET inputs go idle. First we swing the fast rail back to V_{ss} and then reset the circuit by returning the slow rail to V_{ss}. We need to reset the fast rail before the slow rail because the back-end does not have any RESET devices in parallel with the SET devices. The fact that the fast rail must SET and RESET itself while the slow rail is at the SET level forces it to have a much narrower duty cycle and hence the name *fast* rail. Because of the different duty cycles, we now need a total of eight rail, instead of 4, to run our circuits. The timing of these fast rails relative to the slow rail is shown in Figure 3-10.

In addition to the devices that are used for computing, the new circuit has N-Channel cross coupled pairs that are tied to the outputs of front and back sections of the circuit.

Figure 3-10: Timing of the fast rails in N-Channel CRL.

Figure 3-11: Schematic diagram of an N-Channel CRL gate that rests at V_{ss}.

These devices are used to hold the voltage of the node that does not swing to the rest rail and hence to maintain proper operation in the presence of dark currents.

So far we have assumed that the front-end section RESETs to V_{ss} and SETs to V_{dd}. An example of this is the N-Channel CRL buffer shown in Figure 3-11. This buffers is SET by the slow rail going from V_{ss} to V_{dd}. The slow rail rest level is at V_{ss}. In the diagram, C1 and C2 are clamps that are necessary to prevent the internal nodes of the first stage, nodes (1) and (2), from wandering when not actively driven because of leakage or noise. In addition C3 and C4 are clamps that prevent the output nodes from wandering.

By modifying the cross coupled devices, we can have it so that the front-end SETs to V_{ss} and RESETs to V_{dd}. This modification is shown in Figure 3-12. Here, the gate is SET by the slow rail going from V_{dd} to V_{ss} and is reset by the slow rail returning to V_{dd}. Having done that, we could drive the gates that were supposed to be driven by the complement of a slow clock by the true clock itself. This eliminates the need for the complements of the slow clocks which reduces the needed swinging rails from 8 to 6. We note that unlike the previous buffer, no output clamps are needed. This is true since in the RESET state, the voltage on nodes (1) and (2) is equal to $(V_{dd} - V_{Th})$. This voltage is enough to turn on the N-channel devices of P1 and P2 and hence provide for periodic refreshing of the correct voltage levels at the output.

There remains one problem. N-channel CRL gates rely for their proper operation on the capacitive coupling of the fast rail through the output stage devices to the internal nodes (1) and (2) to boost the voltage on one of these nodes to V_{dd}. Unfortunately, it is possible

Figure 3-12: Schematic diagram of an N-Channel CRL gate that rests at V_{dd}.

Figure 3-13: N-Channel CRL gate with latch-up protection devices.

for the boosted voltage on these internal nodes to be boosted beyond V_{dd}. This could trigger latchup. We prevent this by adding clamping devices C5 and C6 as shown in Figure 3-13. C5 and C6 insure that the voltage on the internal nodes never exceeds V_{dd} by more than a threshold voltage. Through proper device sizing, we can control the bootstrapping action so that the boosted voltage will not appreciably exceed V_{dd} and hence reduce the need for the dissipative action of C5 and C6. We stress here that the addition of C5 and C6 are purely for safety and that they are completely removed form the dynamics of charge movements in the gate.

3.3.2 N-Channel CRL Reversible Pipeline

Multiple stage pipelines of the gate described above is achieved using the same connecting topology that was used for Fully-Symmetric CRL circuits described in the previous section and illustrate in Figure 3-7. The timing of the rails is similar to that of Fully-Symmetric CRL except for the inclusion of the fast rails.

In addition to the reduction of the number of devices needed, the above circuit has the added advantage of simplifying reversibility. We can stack the pass gates of the back-end to do computation just as the front-end. Since it is the function of the whole gate, consisting of the two section, that must be reversible, and not the function of the subsections, we can embed non-reversible functions in the front-end section of our gate and not worry about it so long as the function of the whole gate is reversible. For example, we build an adder gate that is easily reversible from AND and OR gates that are not easily reversible but hidden within the bigger reversible adder block.

3.4 Dynamic Considerations and Nonlinearities

The above analysis assumes that we can, in theory, lump the gate capacitances of MOS devices into one equivalent linear capacitance. In practice however, we need to be more careful. Each rail in CRL feeds a number of branches. Ideally, the effective RC time constant of each branch as seen by the rail is data independent and equal to the RC time constant of the entire rail circuit. A branch with a longer time constant would lag behind during the transition. This would create a potential difference across pass gates that are switched on, leading to dissipation. The effect is minimized by the symmetry of CRL. Because of the existence of the true and complement networks for every output line, a swinging rail is always connected to one and only one output line. Therefore, regardless of the output level, the rail will always drive the same output capacitance. The only difference in RC comes from the fact that the true and complement networks are not identical and as such could contribute different RC's depending on the data. Properly sizing the devices so that the two networks exhibit the same time constant independent of the state of the inputs will eliminate this problem. This is possible for both fully symmetric CRL as well as N-Channel CRL since for both have dual polarity outputs and hence it becomes possible for the rail to see the same effective capacitance irrespective of the computation results.

Another point of consideration is the nonlinearity of the capacitances of MOS devices. An enhancement mode MOS device has a higher gate capacitance while in inversion, *i.e.,* conducting, than when it is off. At the beginning of a SETting swing, all the outputs are at idle and all the devices driven by these outputs are off. At the completion of a SET, devices controlled by a swinging output are on. For this reason, a rail that is SETting a gate sees lower effective capacitance at the start of the swing than at the end. With the inductor anchored to $V_{dd}/2$, the rail will not reach the opposite voltage at the end of a swing. This leads to dissipation when the rail is connected to V_{dd} or V_{ss} after completing the swing. Fortunately, a rail SETting a gate is at the same time RESETting another. We feel that as the number of gates connected to a rail increases in a balanced way, the adverse effects of this nonlinearity is minimized. Intuitively, the effective RC time constant is now equal to $R \times C_{average}$.

In addition to the above, there is also the effect of capacitive coupling. Take an output wire that carries a logical true. The devices that it controls in subsequent gates are on. Because of the gate-to-channel capacitance, when the subsequent gates SET, the varying voltage in the channels of turned on devices capacitively couples to the gate. This dumps, or extracts, charge from the output wire of the previous stage. Again, the effect is minimized due to symmetry. Since each output drives identical devices that are capacitively coupled to the top and bottom rail swinging in opposite directions, this capacitive coupling is almost entirely eliminated by the symmetry. We say almost because due to the capacitance nonlinearity, the symmetry will cancel the coupling when integrated over the entire swing and not instantaneously.

We want to stress here that while minimizing the effects of the above phenomenon improves the power saving factor of CRL circuits, none of the above effects, even when extreme, jeopardizes the logical functionality of CRL circuits. HSpice simulations of the 2-input NAND gate, as well as simulations of other CRL gates and circuits, have demonstrated proper operation of the CRL circuit in the presence of these effects. This is important in simplifying the design of CRL logic. If a designer incorrectly sizes a branch in a CRL circuit, the worst he can expect is higher power dissipation in that part of the circuit and not a disfunctional chip.

3.5 Spice Simulation

Numerous HSpice simulations were conducted to for N-Channel CRL gates and pipelines. Since N-Channel CRL gates rely on bootstrapping action, it was important to examine their robustness. These HSpice simulations have confirmed the theoretical predictions regarding N-Channel CRL operation.

3.6 Circuit Example

Figure 3-14 illustrates the design of a 3-Bit full adder. In this design, pipelining was carried out to its fullest extent. The adder consists of 3 1-bit full adders in the forward direction and 3 1-bit full subtractors in the reverse direction. In this implementation, pipelining was stretched to its limit in the sense that the computation is allowed to retire only one bit of addition every cycle. That is the carry out of an adder will affect the next significant 1-bit adder only during the following cycle. Because, the full 3-bit addition is spread out among the pipeline stages, the needed information to start the reverse pipeline arrives later than when it is needed. Fundamentally, the carry for the addition progresses in-step with the forward pipeline, while the carry for the subtraction must progress in-step with the reverse direction. The only way to satisfy both of these constraints is to wait until all the 1-bit additions have been completed before starting the subtractions. This means that a large number of intermediate results have to wait around until needed by the delayed reverse subtractions. This explains the large number of intermediate reversible registers in the diagram.

In practice, a 3-bit adder in CRL technologies would be built out of a single 3-bit carry-propagate adder that completes the addition in one cycle. This would eliminate the need for the majority of the extra registers in Figure 3-14. We will revisit the subject of CRL adders in more detail when we describe the details of the demonstration chip, SCRL-1, that was built to verify our CRL concepts. The implementation of the 3-bit adder described here is included to show how we could achieve single-cycle heavily pipelined performance if we needed to.

Upon the discovery of Split-Level CRL, the research focus shifted away from the techniques described in this chapter and towards this new and much simpler CRL technique that promised obvious advantages. In the next chapter I shall describe this new and improved technique, which we call Split-Level CRL.

Figure 3-14: Schematic diagram illustrating the connection topology of a 1-cycle throughput 3-bit CRL adder using fast carry-save implementation.

Chapter 4

Split-Level Charge Recovery Logic

4.1 Introduction

In the previous chapter we presented forms of early CMOS charge recovery logic (CRL), with a power dissipation that falls with the square of the operating frequency, as opposed to the linear drop of conventional CMOS circuits. Our original implementation, however, had some drawbacks. It required 16-8 times as many devices as conventional CMOS, used 4-2 wires for every signal, and relied on node capacitances to hold a logic level on half of the wires.

In this chapter we present a much improved form of CRL, Split-Level CRL, that uses twice as many devices as conventional CMOS, requires only one wire for every signal, and actively drives all outputs during sampling. Further, we will show how to construct Split-Level CRL circuits using only 2 external inductors for every chip.

Conceptually, Split-Level CRL differs from earlier CRL in two ways. The first is the use of Split-Level voltages. The second is the elimination of the RESET devices and delegating the action of restoring the voltage on SET nodes to gates in the reverse pipeline.

4.2 Split-Level CRL Gate

We begin by describing the topology and operation of a Split-Level CRL inverter. Like conventional CMOS, SCRL gates can have many inputs and outputs. We select the inverter to simplify the description. A device-level diagram of the SCRL inverter is shown in Figure 4-1. It is identical to a conventional inverter except for the addition of a pass gate at the output and the fact that the top and bottom rails are now driven by clocks rather

Figure 4-1: Split-Level CRL inverter.

Figure 4-2: SCRL abstraction box.

than V_{dd} and GND. We call the clock controlling the top rail ϕ_1 and that controlling the bottom rail $/\phi_1$. We refer to the clocks that control the pass gate as P_1 and $/P_1$.

Initially, the input, ϕ_1, $/\phi_1$, the output, and all internal nodes are at $V_{dd}/2$. In addition, P_1 is at GND and $/P_1$ is at V_{dd}, *i.e.,* the pass gate is turned off. After accepting a valid input, V_{dd} or GND, we turn the pass gate on by gradually swinging P_1 and $/P_1$ to V_{dd} and GND respectively. We now gradually swing ϕ_1 to V_{dd} and $/\phi_1$ to GND. The fact that both ϕ_1 and $/\phi_1$ start at $V_{dd}/2$ and *split* towards V_{dd} and GND respectively is the reason we call this family Split-Level CRL. If the input to the gate was at V_{dd} then the output would follow $/\phi_1$ to GND. If the input was at GND then the output would follow ϕ_1 to V_{dd}. We note that at the end of the ϕ_1, $/\phi_1$ swings, the output is the logical NOT of the input. The output is also actively driven and could now be sampled by another gate later in the pipeline.

After the output is sampled by a later gate, the pass gate of this inverter is turned off thus tri-stating the output. Following that, we return ϕ_1 and $/\phi_1$ to $V_{dd}/2$. This in effect restores all the nodes except the output to $V_{dd}/2$. We are now ready to accept a new input. Please note that allowing the input to change prior to resetting all the nodes to $V_{dd}/2$ could turn some devices on while there is a potential difference across them leading to dissipation.

Remember that the output is still at a valid logic level, not $V_{dd}/2$, and before turning on the pass gate we must restore the level of this output to $V_{dd}/2$ to prevent dissipative charge sharing. The promise is that at the point that the pass gate disconnected the output from the inverter, the output was connected to a different gate that has the job of restoring its level to $V_{dd}/2$. We will show how this is done in the following section.

4.3 Reversible Pipeline Connection and Timing

The reason for not letting a SCRL restore its own output to $V_{dd}/2$ is to allow pipelining. Note that to non-dissipatively restore the output to $V_{dd}/2$, the input to the gate must be held constant during the splitting and restoration of its rails. The same restriction dictates that this gate does not restore itself before the subsequent gate in the pipeline restores itself and so on. This means that a new input to a pipeline must be held constant until the effect of this input propagates all the way to the end of the pipeline and until the restoration of the pipeline starting from the last stage reaches back to the first gate. This form of "pipelining" is obviously not very useful.

In this section we show how to connect SCRL gates, or stages, in a non-dissipative pipeline. The main purpose of this method of interconnection is to provide a way of restoring the level of gate outputs to $V_{dd}/2$ with the right timing. We build the pipeline out of copies of an abstraction box shown in Figure 4-2.

We think of this box as containing a parallel set of SCRL gates performing any logical function of an arbitrary number of inputs. Symbolically, the output of the box represents a bundle containing the outputs of the SCRL gates internal to the box. The input to the box represents a bundle containing all the inputs of the gates internal to the box. The function