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

Thesis Supervisor: Thomas F. Knight, Jr.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Technical Report No. 1500 June, 1994

Asymptotically Zero Energy Computing Using Split-Level Charge Recovery Logic

Saed G. Younis
younis@ai.mit.edu

Abstract: The dynamic power requirement of CMOS circuits is rapidly becoming a major concern in the design of personal information systems and large computers. In this work we present a number of new CMOS logic families, Charge Recovery Logic (CRL) as well as the much improved Split-Level Charge Recovery Logic (SCRL), within which the transfer of charge between the nodes occurs quasistatically. Operating quasistatically, these logic families have an energy dissipation that drops linearly with operating frequency, *i.e.,* their power consumption 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 demonstrate the low energy operation of SCRL by presenting the results from the testing of the first fully quasistatic 8×8 multiplier chip (SCRL-1) employing SCRL circuit techniques.

Acknowledgements: This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the Laboratory's Artificial Intelligence Research is provided in part by the Advanced Research Projects Agency under Office of Naval Research contract N00014-91-J-1698 and DABT63-92-C-0039.

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 Bolotski, `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

Contents

I Introduction and Background 1

1 Introduction 2

1.1 New Ideas, Old Physics . . . 2
1.2 Quasistatic Switching . . . 2
1.3 What is Reversible Logic . . . 4
1.4 Temporal Reversibility . . . 4
1.5 Diode Based Proposals . . . 5
1.6 Charge Sharing versus Quasistatic Operation . . . 6
1.7 Logic Families and Universality . . . 6
1.8 Contributions of this Work . . . 6

2 Quasistatic Switching in CMOS 8

2.1 Introduction . . . 8
2.2 Energy Dissipation in CMOS . . . 8
2.2.1 Dissipation Due to Leakage . . . 8
2.2.2 Dissipation due to V_{dd} to V_{ss} Shorting . . . 9
2.2.3 Dynamic Dissipation . . . 9
2.3 Quasistatic Charge Transfer . . . 10
2.3.1 Energy Dissipation with Current Sources . . . 11
2.3.2 Multiple Capacitive Loads . . . 11
2.3.3 Energy Dissipation with Voltage Ramps . . . 13
2.4 Ramp Generators . . . 14
2.5 Sinusoidal Ramp Generator . . . 15
2.5.1 Circuit Dissipation for Sinusoidal Ramps . . . 15
2.5.2 Generator Dissipation for Sinusoidal Ramps . . . 19
2.5.3 Nested Sinusoidal Drivers . . . 21
2.5.4 Inductor Quality factor . . . 21
2.6 Stepwise Ramp Generator . . . 21
2.6.1 Circuit Dissipation for Stepwise Ramps . . . 22
2.6.2 Generator Dissipation for Stepwise Ramps . . . 23
2.7 Alternate Power Switches . . . 23
2.7.1 MODFET Switch . . . 24
2.7.2 Bipolar-MOSFET Switch . . . 24

2.7.3 Micromechanical Switch 26
2.8 Zero Energy Computing and Reversibility 28
2.9 Summary 28

II Development of Charge Recovery Logic 29

3 Early Implementations of Charge Recovery Logic 30
3.1 Introduction 30
3.2 Fully-Symmetric CRL Implementation 30
3.2.1 Fully-Symmetric CRL Gate 30
3.2.2 Reversible Pipeline of Fully-Symmetric CRL 33
3.3 N-Channel CRL 36
3.3.1 N-Channel CRL Gate 36
3.3.2 N-Channel CRL Reversible Pipeline 40
3.4 Dynamic Considerations and Nonlinearities 40
3.5 Spice Simulation 41
3.6 Circuit Example 41

4 Split-Level Charge Recovery Logic 44
4.1 Introduction 44
4.2 Split-Level CRL Gate 44
4.3 Reversible Pipeline Connection and Timing 45
4.4 SCRL Clocking Variants 47
4.4.1 Two-Phase SCRL 47
4.4.2 Three-Phase SCRL 48
4.5 Non-Inverting Stage 49
4.6 External Inductors 51
4.7 Spice Simulation 51
4.8 Lowering Irreversibility Cost 52
4.8.1 Irreversibility Is Not Free 52
4.8.2 Statistically Controlled Irreversibility 53
4.8.3 Where to Break Reversibility 53

III Implementation and Testing of SCRL-1 Demonstration Chip 54

5 Demonstration Chip Details 55
5.1 Introduction 55
5.2 High Level Multiplier Design 55
5.3 Multiplier Details 57
5.3.1 Reversible Level 1 58
5.3.2 Reversible Level 2 60
5.3.3 Reversible Level 3 63

5.3.4 Reversible Levels 4-8 . . . 64
5.3.5 Reversible Level 9 . . . 64
5.3.6 Reversible Level 10 . . . 64
5.3.7 Reversible Level 11 . . . 66
5.3.8 Reversible Level 12 . . . 67
5.3.9 Rail Connections . . . 67
5.3.10 I/O Connections . . . 68
5.4 Input Pads . . . 69
5.4.1 Operand Input Pads . . . 70
5.4.2 Slow and Fast Rails Input Pads . . . 70
5.4.3 Pass rails Input Pads . . . 70
5.5 Output Pads . . . 70
5.6 Layout of SCRL Circuits . . . 71
5.7 Design Entry and Verification . . . 73

6 Test and Measurement Results **74**
6.1 Introduction . . . 74
6.2 Digital Functional Testing . . . 74
6.2.1 Testing of Forward Pipeline . . . 74
6.3 Testing of Reverse Pipeline . . . 75
6.4 Split-Level Operational Testing . . . 76
6.4.1 Direct Digital Synthesis Testing Approach . . . 76
6.4.2 RC-Controlled Voltage Step Testing . . . 79
6.5 Energy Measurement Procedures . . . 80
6.6 Test Results . . . 81
6.6.1 The Problem of Direct Measurement of SCRL Energy . . . 82
6.6.2 Confirmation of SCRL Energy Flow . . . 83
6.6.3 Confirmation of Node Hand-off in SCRL . . . 84
6.6.4 SCRL verses CMOS Operation . . . 84
6.6.5 SCRL Capacitive Coupling . . . 86
6.7 Summary . . . 88

IV Future Research Areas **89**

7 Future Work **90**
7.1 SCRL-1 Energy Measurement . . . 90
7.2 CAD Work . . . 90
7.3 Architecture Issues . . . 91
7.4 Power Supply Switch . . . 91
7.5 SCRL Circuit Improvements . . . 91

A CRL at Low Temperature **93**
A.1 CRL and Large Systems at LNT . . . 93

B SCRL-1 Pinout **99**

List of Figures

2.1 Energy analysis model of CMOS circuits. 9
2.2 Normalized power dissipation plot for a voltage step, solid line, and for a current step, dashed line. 12
2.3 Model of CMOS circuit with separate driving current sources. 12
2.4 Model of CMOS circuit with common driving current source. 12
2.5 Voltage waveform at the terminals of a constant current source driving a capacitive load C. 13
2.6 The two places of system dissipation. 15
2.7 Conventional and non-dissipative circuit analysis models. 16
2.8 Plot of $V_c(t, \omega_d)$ showing the effect of fast rise and fall times on energy loss. 17
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}. 19
2.10 Typical stepwise ramp generator circuit driving a capacitive load C_L. 22
2.11 Model of a CMOS computing circuit driven by a stepwise source. 22
2.12 Schematic diagram of a simple Electroquasistatic Micromechanical switch. . 27

3.1 Conventional NAND gate and an early attempt at a CRL NAND gate. . . . 30
3.2 Modified gate with dual-rail added. 31
3.3 The universal CRL NAND gate. 32
3.4 Completed CRL NAND gate with pipelining support. 33
3.5 First CRL implementation with output cross-coupled clamps. 34
3.6 (a) CRL abstraction box. (b) Timing of the four clock rails. 34
3.7 Non-dissipative multi-stage pipeline connection. 35
3.8 Last pipeline stage connection for dissipation reduction. 36
3.9 Schematic diagram of a 2-Input AND/NAND N-Channel CRL gate. 37
3.10 Timing of the fast rails in N-Channel CRL. 38
3.11 Schematic diagram of an N-Channel CRL gate that rests at V_{ss}. 38
3.12 Schematic diagram of an N-Channel CRL gate that rests at V_{dd}. 39
3.13 N-Channel CRL gate with latch-up protection devices. 39
3.14 Schematic diagram illustrating the connection topology of a 1-cycle throughput 3-bit CRL adder using fast carry-save implementation. 42

4.1 Split-Level CRL inverter. 44
4.2 SCRL abstraction box. 45

4.3 Non-dissipative multi-stage pipeline connection. 46
4.4 Rail timing for 4 phase SCRL. 47
4.5 Two-Phase SCRL pipeline. 48
4.6 Rail timing for 2 phase SCRL. 48
4.7 Three-Phase SCRL Pipeline. 49
4.8 Rail timing for 3 phase SCRL. 49
4.9 Non-Inverting SCRL Gate. 50
4.10 Timing diagram of Two-phase SCRL with fast rails for non-inverting stages. 51
4.11 Inductive rail driver circuit. 52
4.12 Scaling down stage sizes before breaking reversibility in SCRL. 53

5.1 Block diagram of a 4-Bit SCRL multiplier. 56
5.2 Schematic diagram of reversible level 1. 59
5.3 Schematic diagram of `splitReg1_16` module. 59
5.4 Schematic diagram of reversible level 2. 61
5.5 Schematic diagram of `revInv1_1` module. 61
5.6 (a) Schematic Diagram of a generic 1-bit propagate-generate full adder. (b) Diagram of the 2-level 1-bit adder that are used in SCRL-1 chip. 62
5.7 Schematic diagram of reversible level 3 module. 63
5.8 Schematic diagram of reversible level 9 module. 65
5.9 Schematic diagram of reversible level 10 module. 66
5.10 Schematic diagram of reversible level 11 module. 67
5.11 Schematic diagram illustrating the connection of the rails in SCRL-1. . . . 68
5.12 Schematic diagram illustrating the I/O connections in SCRL-1. 69
5.13 Schematic diagram of SCRL-1 output pad circuit. 71
5.14 Sketch of a typical layout of an SCRL chip. 72

6.1 Block diagram the high speed DAC module. 77
6.2 Block diagram of the components of the RC-controlled voltage step testing approach. 79
6.3 Scope probe locations during energy dissipation measurement. 81
6.4 (a) Scope trace of the rise time of FF1(+) with 2777.5Ω resistor. (b) Scope trace of the rise time of FF1(+) with 2777.5Ω resistor. 85
6.5 (a) FF1(+) waveform under SCRL operation. (b) FF1(+) waveform with the reverse pipeline disabled. 86
6.6 Non-Inverting SCRL Gate. 86
6.7 CMOS capacitive from swinging of FS1(+/-) at varying R_{ext} on FF1(+). . 87

B.1 Pinout map of the PGA84 package of SCRL-1. 100

List of Tables

6.1 Energy measurements for rail FF1(+) in SCRL-1. 83
6.2 Energy measurements for rail FF3(-) and RF4(-) in SCRL-1. 84

Part I

Introduction and Background

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 capacitances 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 as 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

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 that were used to hold on to past values. 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. This is because reversible computation provides us with the needed information to make the correct connection to the restoring swinging rail and thus allows us to restore nodes quasistatically. 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 controlled 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 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 versus 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 reflects itself in a variable effective capacitance of the circuit thus nullifing 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 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 Section 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 Section 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 Section 4.

In Section 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 serves 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 Section, 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. In Section 5, I describe the design of this demonstration chip. Following that, in Section 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 Section 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...

2. Quasistatic Switching in CMOS

2.1 Introduction

The subject of this section 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 circuit nodes at differing potentials 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 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 that in the case that subthreshold conduction becomes appreciable, we can trade some of it for higher

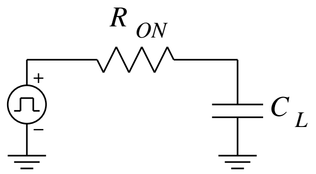


Figure 2.1: Energy analysis model of CMOS circuits.

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 Sections 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

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

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 is dissipated in the resistor during charging, and the other half is 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 Section 3 and Section 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

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

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

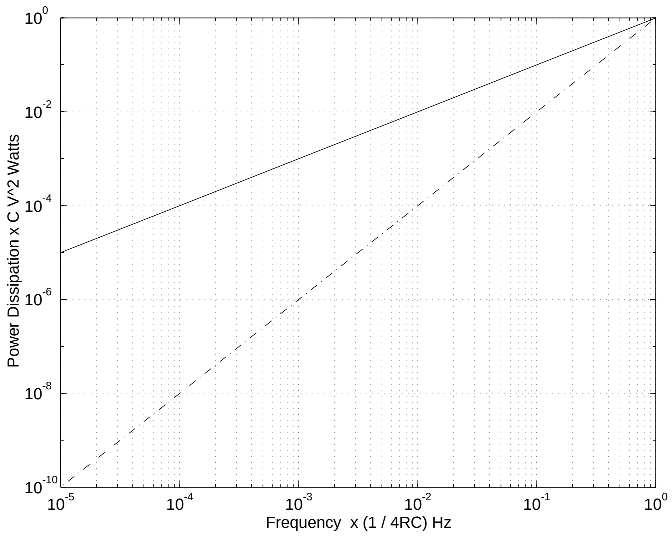


Figure 2.2: Normalized power dissipation plot for a voltage step, solid line, and for a current step, dashed line.

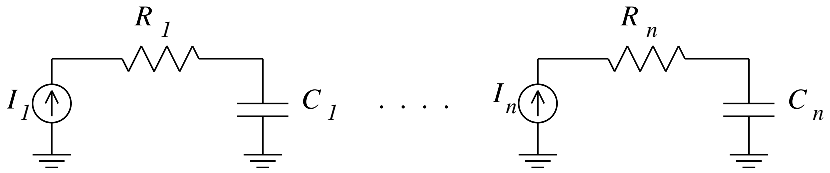


Figure 2.3: Model of CMOS circuit with separate driving current sources.

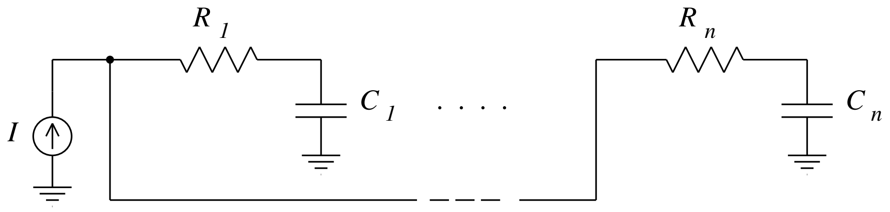


Figure 2.4: Model of CMOS circuit with common driving current source.

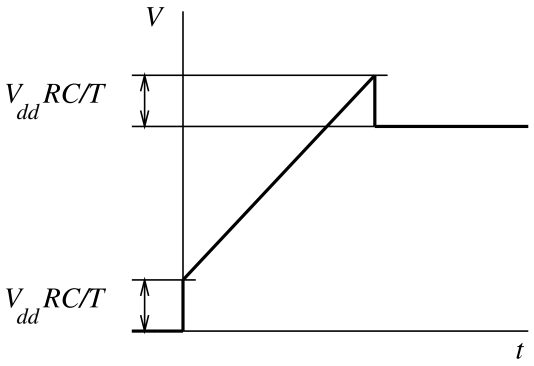


Figure 2.5: Voltage waveform at the terminals of a constant current source driving a capacitive load C.

the 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 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 conduction 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

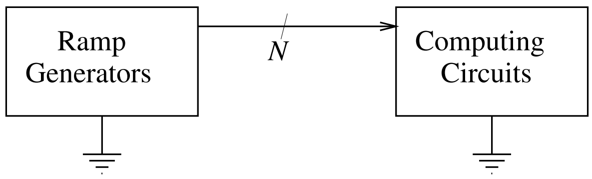


Figure 2.6: The two places of system dissipation.

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 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

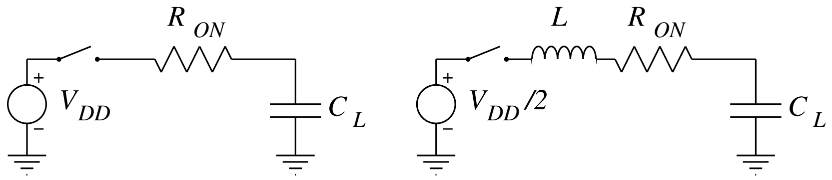


Figure 2.7: Conventional and non-dissipative circuit analysis models.

approximation where the R and C 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,
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

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

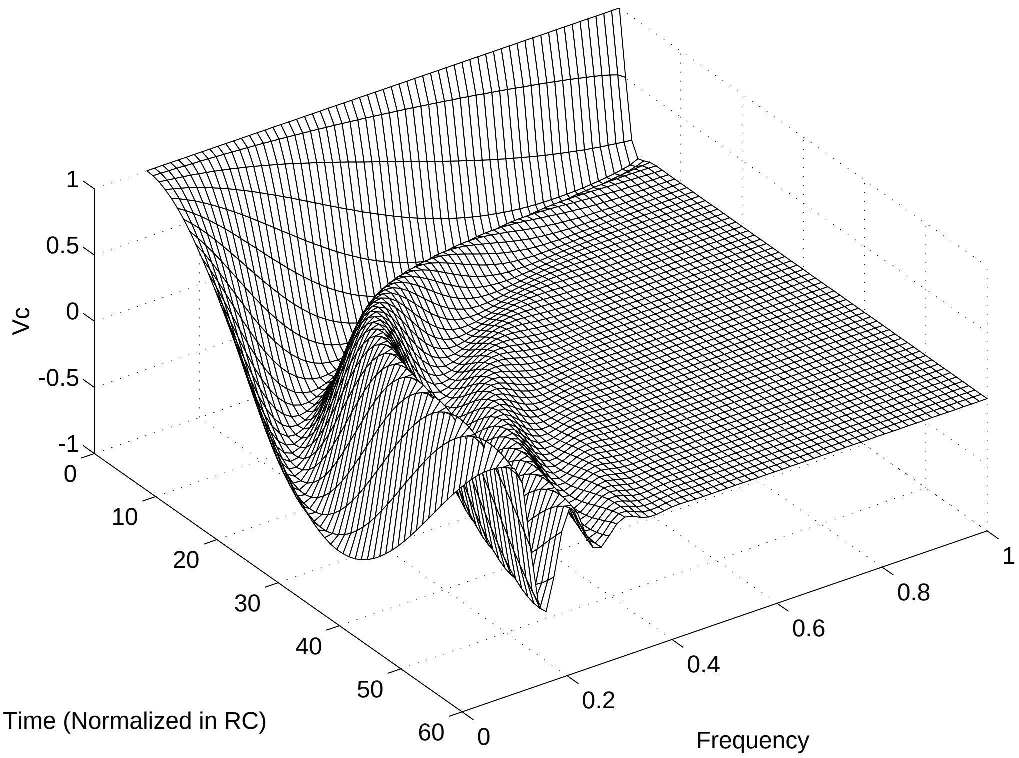


Figure 2.8: Plot of $V_c(t, \omega_d)$ showing the effect of fast rise and fall times on energy loss.

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

$$V_C(t) = V_0 e^{-\alpha t}(cos\omega_d t + \frac{\alpha}{\omega_d} sin\omega_d t) \tag{2.18}$$

and

$$i_C(t) = V_0 C \frac{{\omega_0}^2}{\omega_d} e^{-\alpha t} sin\omega_d t \tag{2.19}$$

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 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 ω_{dmax} 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.

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.

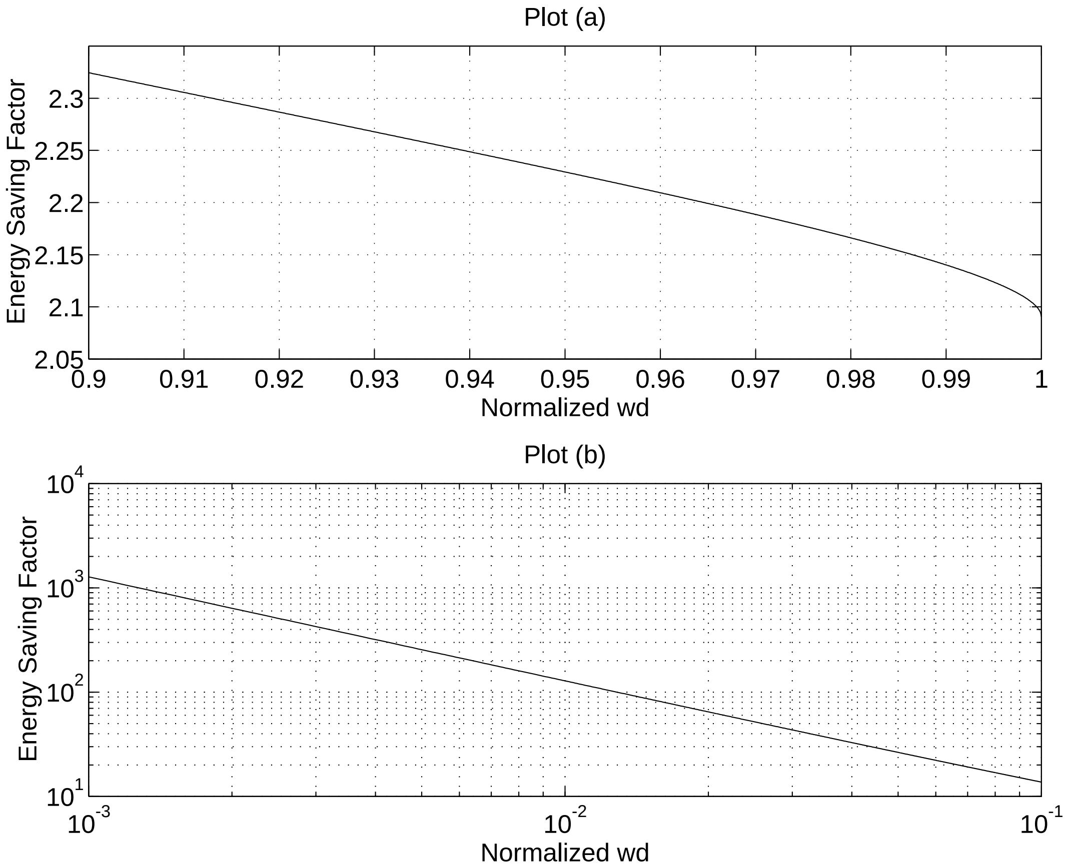


Figure 2.9: (a) Linear plot of F_{saving} factor vs. ω_d. (b) Log-Log plot of F_{saving} factor vs. ω_d. In both plots ω_d is normalized to ω_{dmax}.

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 τ_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".

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

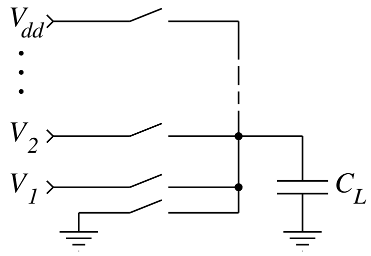


Figure 2.10: Typical stepwise ramp generator circuit driving a capacitive load C_L.

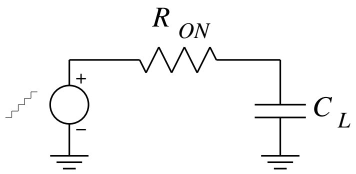


Figure 2.11: Model of a CMOS computing circuit driven by a stepwise source.

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.

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 times the RC 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

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

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

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

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 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 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

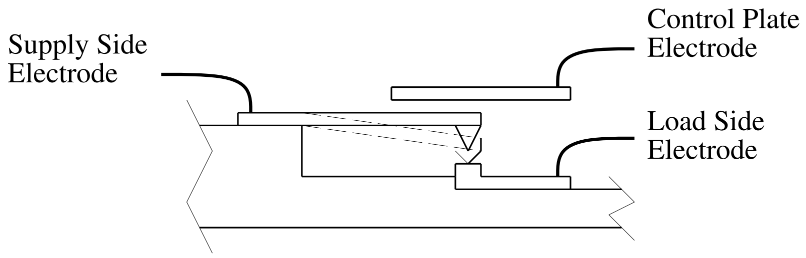


Figure 2.12: Schematic diagram of a simple Electroquasistatic Micromechanical switch.

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 section 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. They 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 building non-dissipative ramp generators 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 section 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 sections we will see how non-dissipative computing invariably leads to reversible computing and that thermodynamic entropy and information entropy are strongly linked.

Part II

Development of Charge Recovery Logic

3.1 Introduction

In this section 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

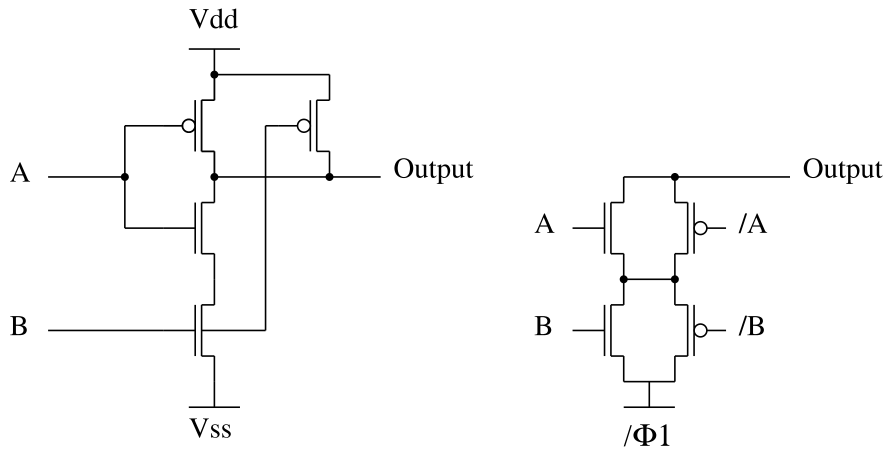


Figure 3.1: Conventional NAND gate and an early attempt at a CRL NAND gate.

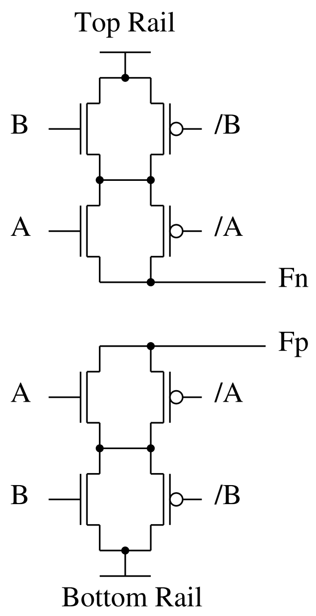


Figure 3.2: Modified gate with dual-rail added.

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 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.

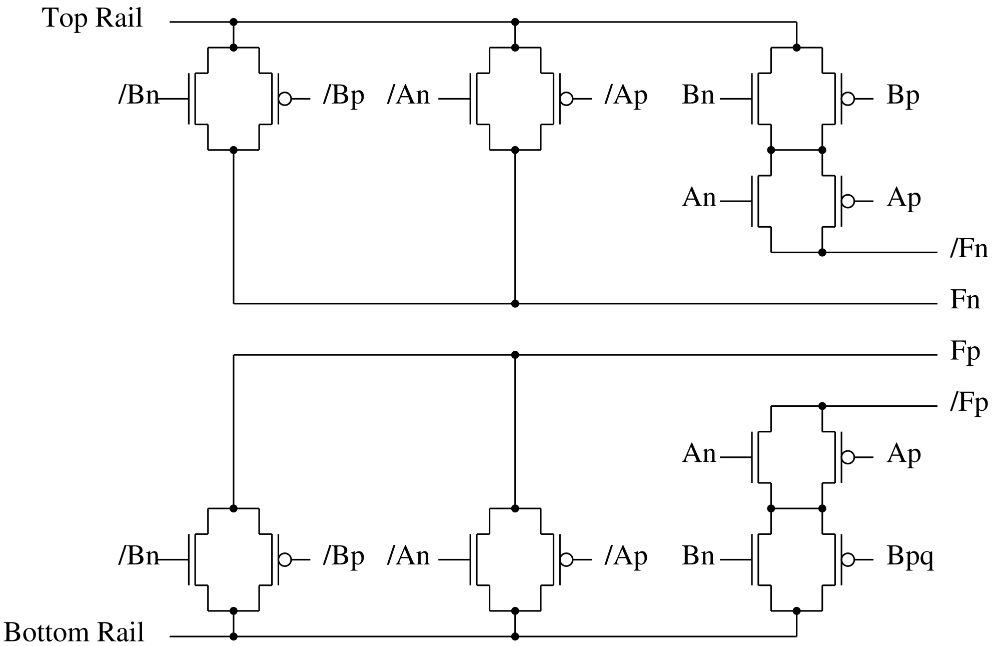


Figure 3.3: The universal CRL NAND gate.

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 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

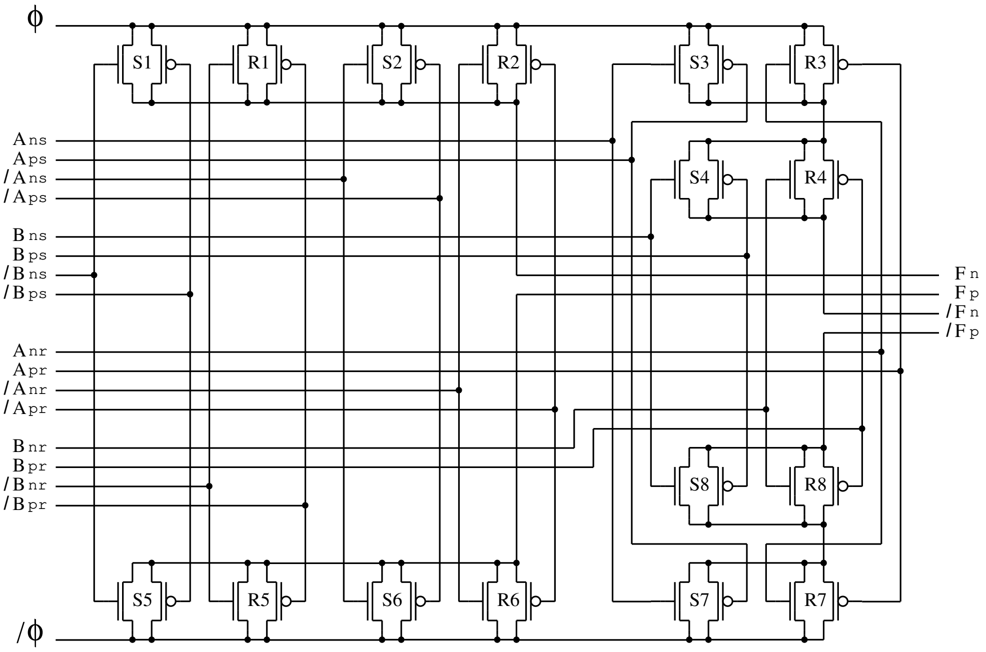


Figure 3.4: Completed CRL NAND gate with pipelining support.

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 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

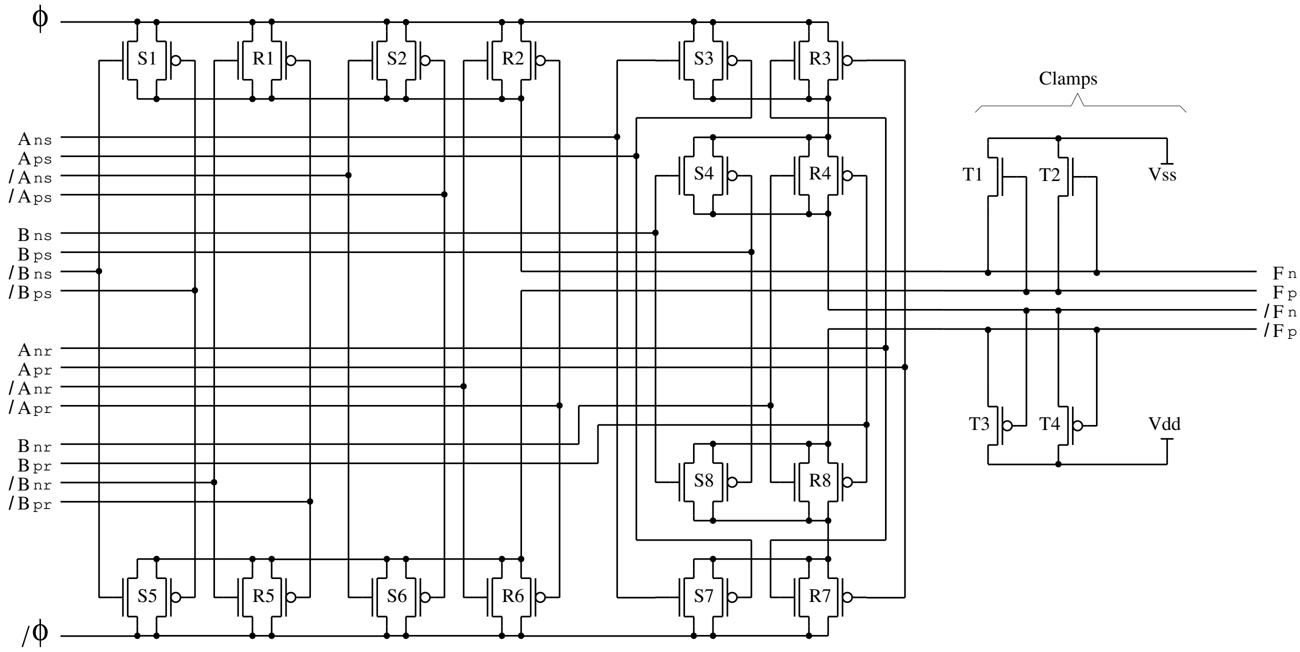
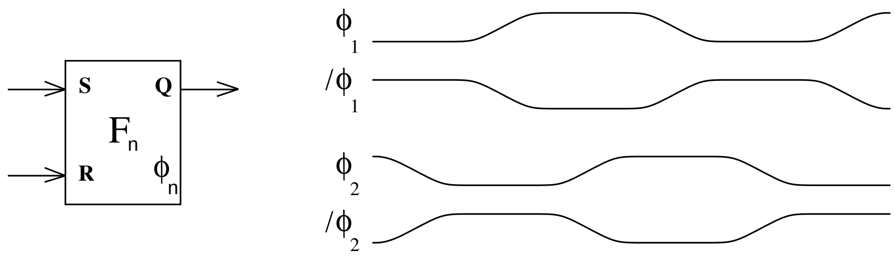


Figure 3.5: First CRL implementation with output cross-coupled clamps.



(a) (b)

Figure 3.6: (a) CRL abstraction box. (b) Timing of the four clock rails.

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.

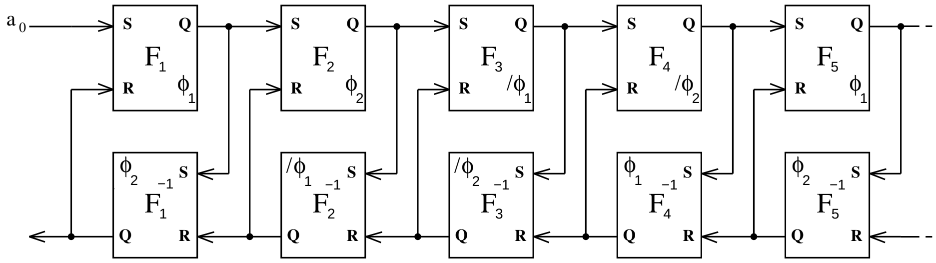


Figure 3.7: Non-dissipative multi-stage pipeline connection.

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 output 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.

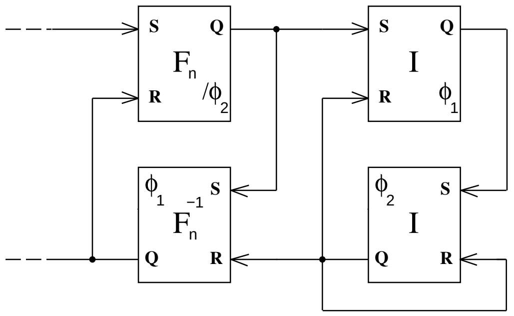


Figure 3.8: Last pipeline stage connection for dissipation reduction.

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 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

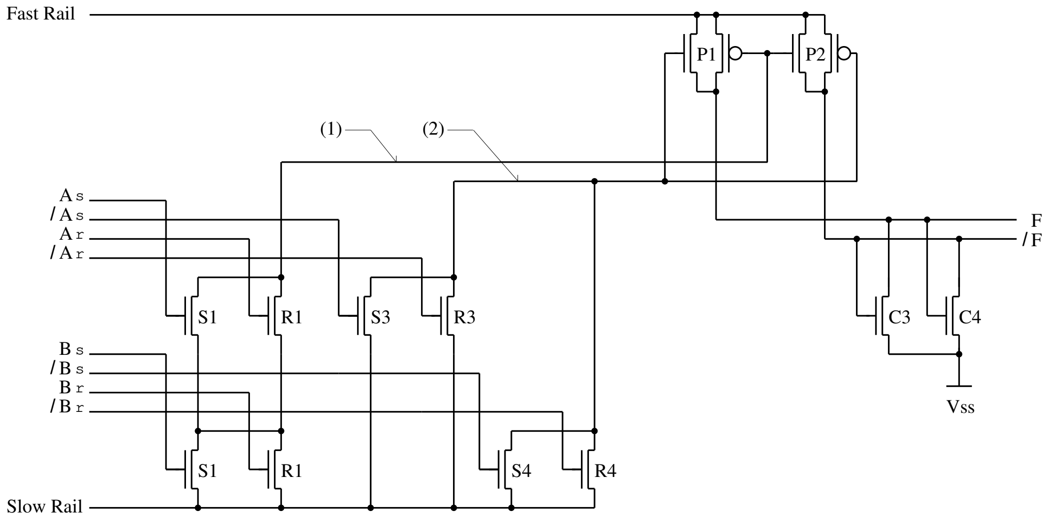


Figure 3.9: Schematic diagram of a 2-Input AND/NAND N-Channel CRL gate.

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. 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.

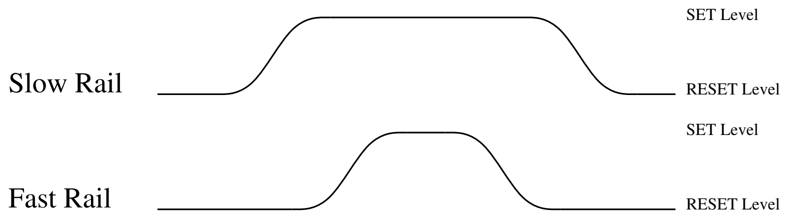


Figure 3.10: Timing of the fast rails in N-Channel CRL.

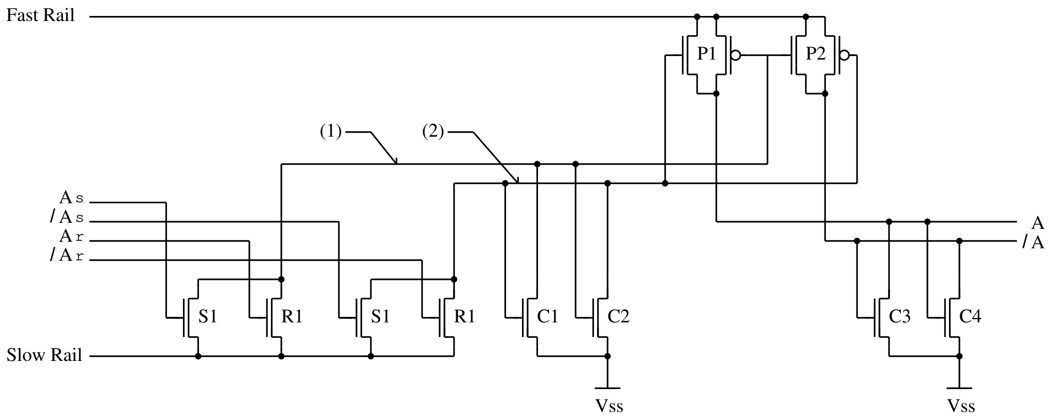


Figure 3.11: Schematic diagram of an N-Channel CRL gate that rests at V_{ss}.

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.

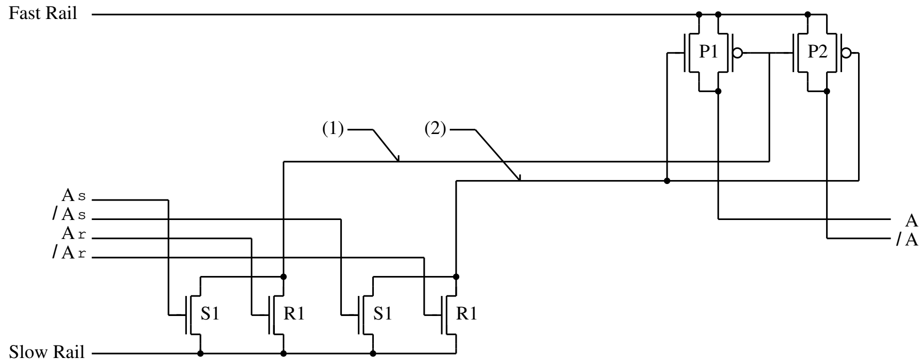


Figure 3.12: Schematic diagram of an N-Channel CRL gate that rests at V_{dd}.

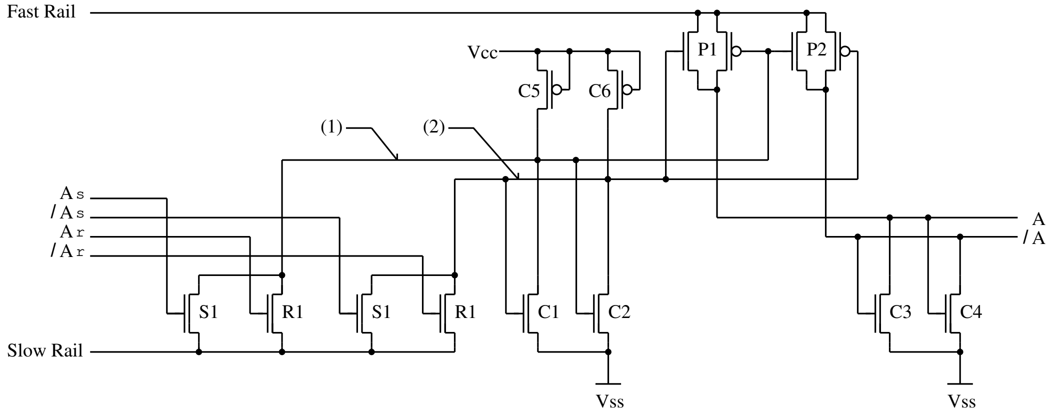


Figure 3.13: N-Channel CRL gate with latch-up protection devices.

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 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

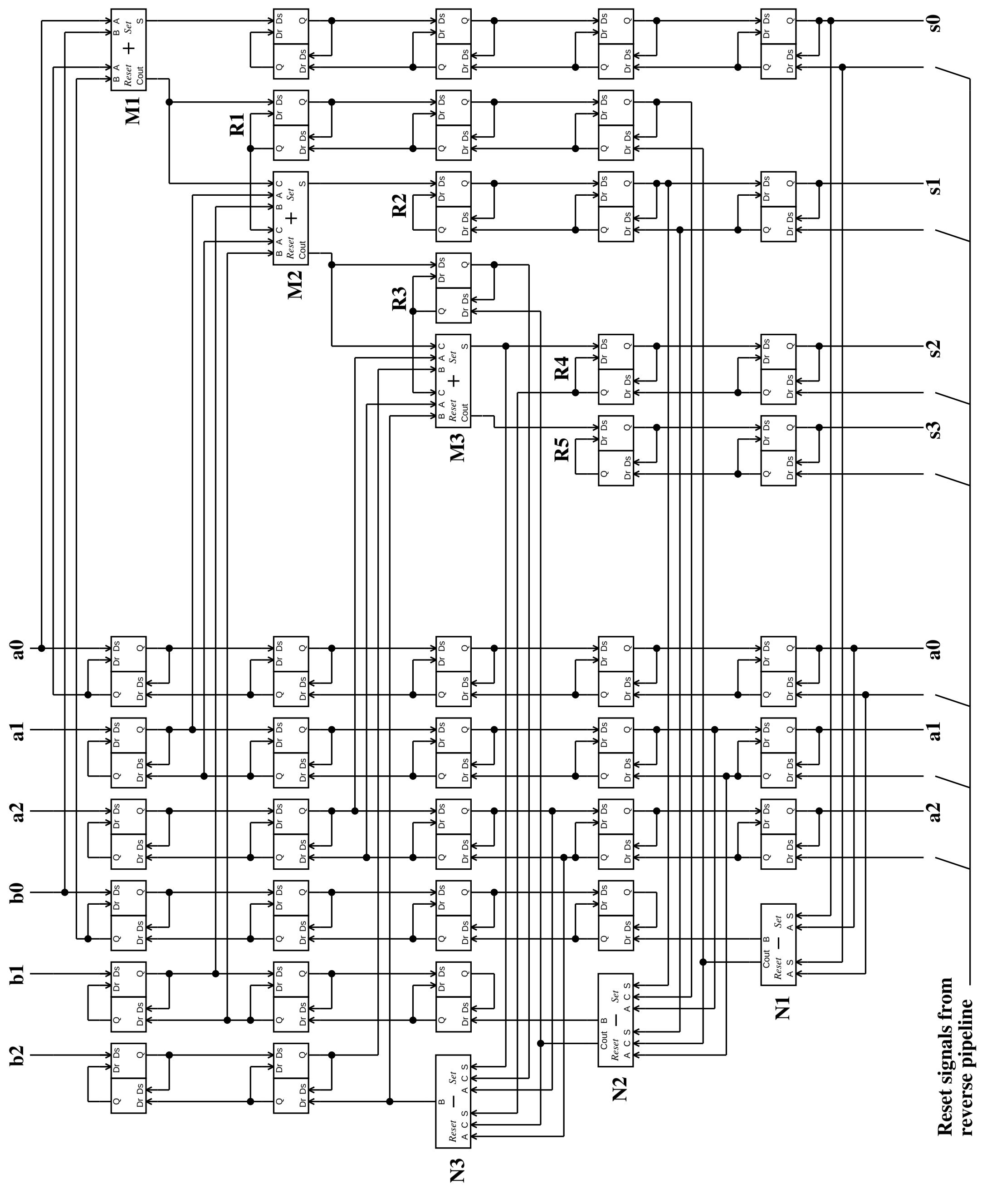


Figure 3.14: Schematic diagram illustrating the connection topology of a 1-cycle throughput 3-bit CRL adder using fast carry-save implementation.

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 section and towards this new and much simpler CRL technique that promised obvious advantages. In the next section I shall describe this new and improved technique, which we call Split-Level CRL.

4.1 Introduction

In the previous section 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 section 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 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

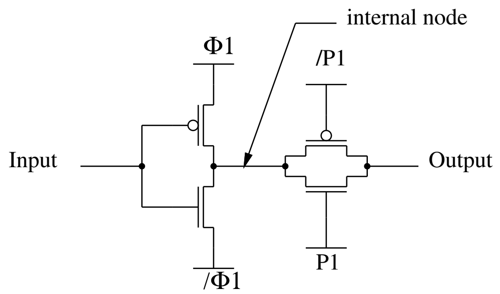


Figure 4.1: Split-Level CRL inverter.

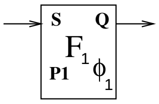


Figure 4.2: SCRL abstraction box.

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

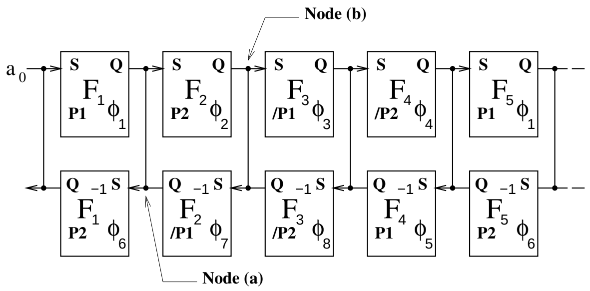


Figure 4.3: Non-dissipative multi-stage pipeline connection.

computed by the box is identified by the letter in the center of the box. Finally, indicated at the bottom of the box are the clocks used to control both the Split-level rails and the pass gate controls of all the SCRL gates internal to that box. A clock of ϕ_1 in the lower right corner indicates that the top rail is connected to ϕ_1 and the bottom rail to $/\phi_1$, while a clock of P_1 in the lower left corner indicates that the pass gate is on when P_1 is high.

Using this abstraction, Figure 4.3 illustrates how SCRL gates are connected to produce a non-dissipative pipeline. Note that the box with a function F^{-1} performs the inverse operation of the box with a function F. The computation proceeds from left to right in the top half of the pipeline and the "uncomputation" proceeds from right to left on the bottom half of the pipeline.

Each line linking SCRL gates is connected to the outputs of two different SCRL gates. For example, node (a) is connected to the output of F_1 and to the output of F_2^{-1}. There are two reasons why no logic fights occur between the gates driving the same line. The first is that when one gate is driving the line the other is tri-stated and visa-versa. The second is that during hand off, the voltages at the output of the gates is guaranteed to be equal. In this pipeline, the forward gates are responsible for gradually swinging an output line from $V_{dd}/2$ to V_{dd} or GND depending on the computation. The reverse pipeline is responsible for restoring the output line from the active levels to $V_{dd}/2$.

To avoid dissipation, the backward gates have to determine the value of the output that they are about to restore to $V_{dd}/2$ and set their output to that level before their pass gate is switched on, *i.e.,* before the line is handed off from the forward gate. To see how this works, we go through the events that occur after a new input, say a_0 is presented to the pipeline. First P_1 turns on the pass gate of F_1 and turns off the pass gate of F_2^{-1}. Next ϕ_1 splits setting node (a) to the valve $F_1(a_0)$. F_2 goes through similar transitions and produces $F_2(F_1(a_0))$ at node (b). Similarly F_2^{-1} produces $F_2^{-1}(F_2(F_1(a_0))) = F_1(a_0)$. Note that at this point the voltage levels at the outputs of F_1 and F_2^{-1} are at the same level which means that it is now safe to hand off node (a) to F_2^{-1} from F_1 by swinging P_1 low. After the hand off, we can restore F_1 by restoring ϕ_1. This could occur even without having to wait until F_2 is restored because F_2^{-1} is still holding node (a) at its valid value. After F_2 is restored F_2^{-1} gradually restores node (a) to $V_{dd}/2$ and hands it over to F_1. The timing diagram for a four phase clocking scheme is shown in Figure 4.4. For $\phi_1 \ldots \phi_8$ in the figure, a high indicated when they are split and a low when they are restored. For $P_1 \ldots P_2$, a high is V_{dd}

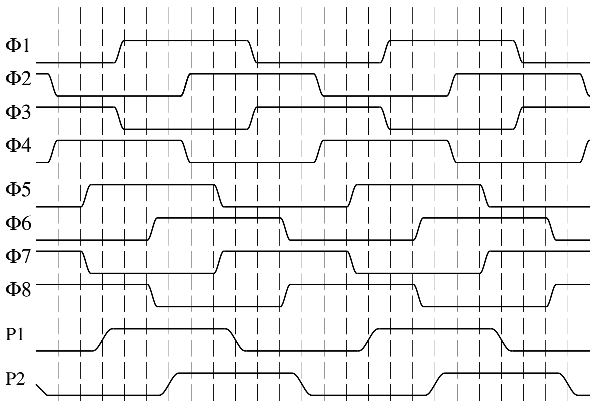


Figure 4.4: Rail timing for 4 phase SCRL.

and a low is GND. With this pipeline, we are able to accept a new input every ϕ_1 without needing to wait for the restoration of later stages.

There remains one problem however. At the end of the pipeline, the input to F_5^{-1} is not restored and hence driving this line is dissipative. Furthermore, it could not be generated, as this is the place where reversibility is broken. This implies the fundamental limit that links information entropy with thermodynamic entropy. If at any moment a piece of information that is vital to reconstruct the past is lost, energy is dissipated. Fortunately, this dissipation occurring only at the end of a long pipeline is negligible.

4.4 SCRL Clocking Variants

In what follows, we will describe a number of alternatives for constructing SCRL circuits. These circuits differ primarily by the number of required phases and/or rails that are needed to control their operation. The pipeline described previously in this attached paper required four-phase clocking. This used four different clock phases in the forward pipeline and four others for the reverse pipeline. By four-phase we mean that the shortest feedback path in the pipeline has to span a minimum of four pipeline stages. In this following sections we will show how to construct SCRL circuits using two-phase, three-phase, five-phase and six-phase pipelines. One might simplistically think that less phases lead to less required rails. This unfortunately is not true since for some implementations the required phases are non-symmetric and therefore the complement of a phase cannot be used for more than one purpose. For this reason, the primary reason for reducing the number of phases is to minimize the number of stages for the shortest feedback path. Additionally, the lower the number of phases that a SCRL circuit uses, the easier it is to understand and apply.

4.4.1 Two-Phase SCRL

For all the implementations that will be described in this section, the basic gate is the same as the one described in the Section 4.2. Figure 4.5 shows a pipeline of a two-phase

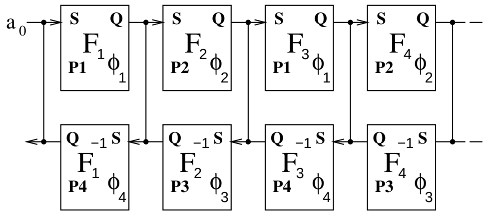


Figure 4.5: Two-Phase SCRL pipeline.

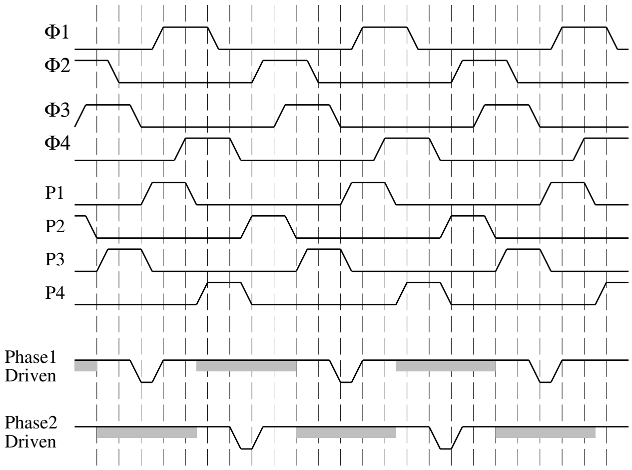


Figure 4.6: Rail timing for 2 phase SCRL.

SCRL implementation. The timing relationships among the rails are shown in Figure 4.6. Two-phase SCRL forfeits the benefit of always actively driving the nodes whenever they are sampled in exchange for achieving two-phase pipelining.

For $\phi_1 \ldots \phi_4$ in the timing diagram, a high indicates the time when ϕ and $/\phi$ are split and a low indicates when they are at $V_{dd}/2$. For $P_1 \ldots P_4$, a high indicates that P is at V_{dd} and $/P$ at GND while a low indicates that P is at GND and $/P$ at V_{dd}. The bottom two timing lines indicate the states of outputs driven by ϕ_1 and ϕ_2 gates. A high there indicates when the output is at an active level of V_{dd} or GND, while a low indicates that the output is at $V_{dd}/2$. The shaded regions in the timing diagrams indicate the times at which the signals are not being actively driven, *i.e.,* floating at an active level.

4.4.2 Three-Phase SCRL

Figure 4.8 shows the timing diagram of a three-phase SCRL implementation. The bottom timing line shows the timing of an output that is driven by a ϕ_1 gate. Note that in three-phase, and higher, implementations the outputs are always actively driven. Figure 4.7 shows a three-phase SCRL pipeline.

It is relatively easy to generalize the above concepts to five-phase, six-phase, etc. Since three-phase systems achieve active driven outputs, the usefulness of higher phase systems

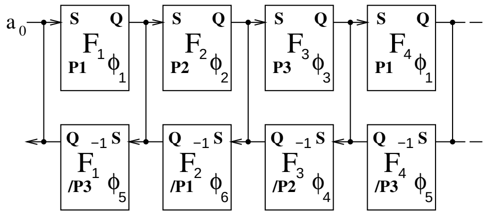


Figure 4.7: Three-Phase SCRL Pipeline.

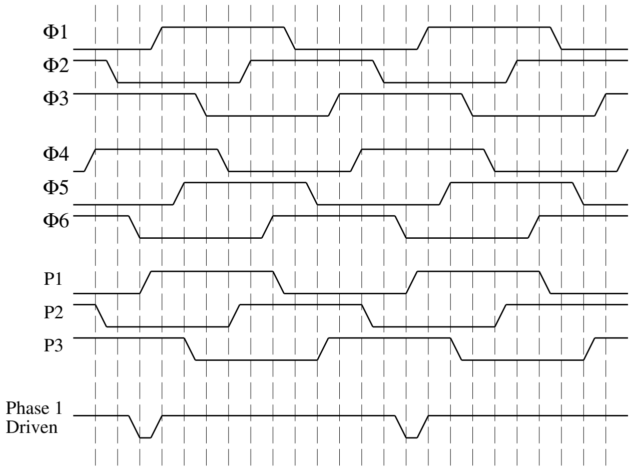


Figure 4.8: Rail timing for 3 phase SCRL.

could be limited.

4.5 Non-Inverting Stage

Since the basic SCRL gate mimics that of conventional CMOS, we find that it is not possible to pass a signal through a SCRL stage without inverting it. For some circuits it is necessary to receive both the true and complement of a logical signal simultaneously at the inputs of a logic gate. Starting with a single signal, it is not possible to have its true and complement arrive at a later stage simultaneously given the circuits we have described so far. In order to pass a signal without inversion we substitute the basic SCRL gate with the one shown in Figure 4.9. Please note that this buffer requires an additional set of controlling clocks we call "fast ϕ_1 and fast $/\phi_1$ for a ϕ_1 gate. The restriction on fast ϕ_1 is that is splits *immediately after* ϕ_1 splits and that is restores *just before* ϕ_1 restores. In other words, the transitions of ϕ_1 contain within them the transitions of fast ϕ_1. For stages where we want to pass a signal without inversion, we use a gate similar to the one in Figure 4.9 and we clock its fast clocks according to the relations described.

In place of the inverters in Figure 4.1 and Figure 4.9 one can put any CMOS gate such as NAND, NOR etc. We can see that an additional benefit of a non-inverting SCRL gate, is that it allows each functional block to have a 2-level logic implementation. This generally

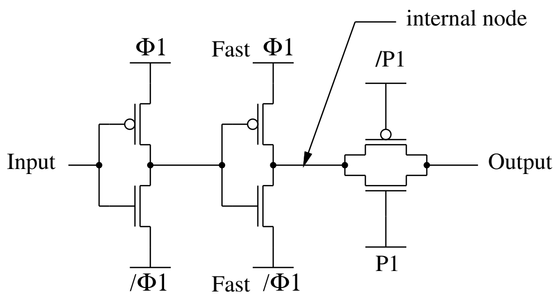


Figure 4.9: Non-Inverting SCRL Gate.

aids in reducing the storage buffers that are sometimes needed for reversibility.

Another benefit of having a 2-level SCRL has to do with optimal step-up ratio of logic gates. It is well known that a CMOS inverter made out of the minimum size devices can optimally drive between 3-5 other inverters of the same size. A minimum sized inverter driving more than this optimal step-up number of loads similar to its size would have a larger delay. Since the power saving of SCRL is referenced to the maximum operating frequency of a similar circuit in conventional CMOS, this longer delay leads to less power savings. For an inverter to drive more than 3-5 loads and maintain the same speed, it must be made out of larger sized devices. Unfortunately, larger devices have larger input gate capacitances and hence present a larger load to the gates that are driving them. To see how this could be a problem, let us consider building a multiplier out of an array of identical 1-Bit SCRL adder gates. The multiplier would consists of an array of gates in which each gate takes its inputs from a previous *identical* gate and provides on its output the data for the inputs of another *identical* gate. Typically in these arrangements, each output would fan-out to drive more than 3-5 loads because each input to a gate feeds a number of devices internal to that gate. For SCRL, just as for CMOS, having an output drive more than 3-5 loads its size is not optimal. As mentioned earlier, increasing the driving capability of a gate so as to be able to drive the loads, *i.e.,* by doubling the width of the devices used in it, also increases the input capacitance, and hence the load, that this gate presents to the *identical gate* driving it. By attempting to increase the driving capability, we also increased the loads, and thus lost the benefit that we where attempting to gain.

Having 2-level SCRL allows for increasing the driving capability of a gate without increasing the load it presents to the other gates. This is done by performing most of the computations in the first level and then using the second level to provide the buffering. For this reason the first stage can consist primarily of minimum sized devices, and thus present the minimum load for the previous gate, while the second stage is made of devices 3-5 times the minimum size to give optimal driving capability.

Finally, the timing diagram of a two-phase SCRL with fast clocks is shown in Figure 4.10. The figure indicates the position of the transitions of the fast rails using the dashed lines.

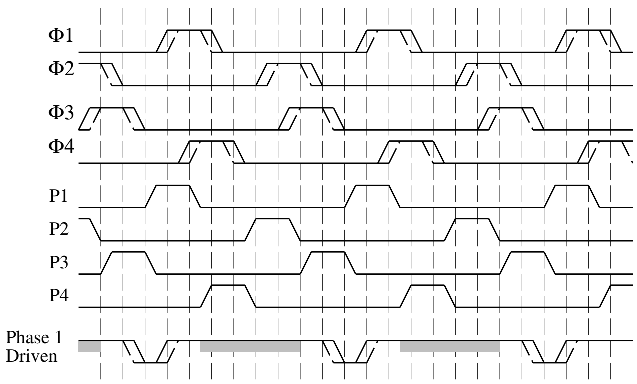


Figure 4.10: Timing diagram of Two-phase SCRL with fast rails for non-inverting stages.

4.6 External Inductors

In Section 2 we examined how quasistaic switching could be achieved with the aid of external inductors. For a 4-Phase SCRL pipeline, we need a minimum of 20 seperate rails to control the operation of the pipeline. Figure 4.11, is a diagram of an inductive rail driver. A rail could be approximated by a capacitor in series with a resistor. The capacitor is the sum of the capacitances that the rail is driving and the resistor is the equivalent resistance of the devices through which the capacitances are driven. Suppose that the initial voltage on the rail was V_{init} and we want to swing the rail to V_{fin}. To start the swing, we connect the rail through an inductor to a DC power supply at $(V_{init} + V_{fin})/2$. Current starts to build up in the inductor and the rail starts the swing towards V_{fin}. At the moment that the current drops back to zero again we disconnect the inductor. The rail should now be at V_{fin}. The action of connecting and disconnecting the rail is performed by the power MOSFET. Please note that the inductor is only necessary during the transition and is otherwise disconnected from the rail. Note further that the current in a disconnected inductor is zero. With this in mind, we should be able to multiplex the inductor among multiple rail circuits so long as these multiplexed rails do not have simultaneous transitions. Examining the timing diagram of Figure 4.4, we see that no more than two transitions occur simultaneously at any moment. By using power MOSFET multiplexors on both sides of the inductor, rather than a MOSFET on one side, we see that the maximum number of required external inductors is 2. Integrating everything but the inductors on a silicon chip means that a Split-Level CRL chip requires 7 additional pins for proper operation. Two of these pins are V_{dd} and GND.

4.7 Spice Simulation

A number of HSpice simulations were carried out on simple SCRL gates. HSpice simulations were also carried out on submodules extracted from the actual layout of our 8×8 demonstration chip, SCRL-1, to check for SCRL circuit operations when parasitics are

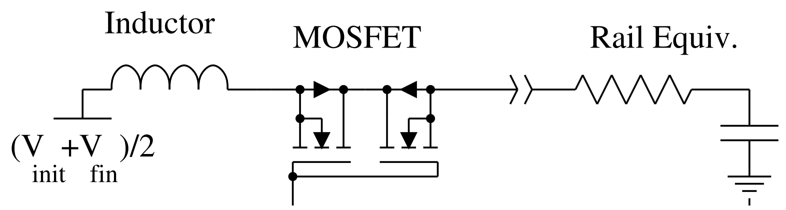


Figure 4.11: Inductive rail driver circuit.

included. All of the simulation results were verified by test measurements on the actual demonstration chip as will be detailed in Section 6.

4.8 Lowering Irreversibility Cost

In CRL and SCRL the reverse pipeline is required to accurately provide a delayed copy of the inputs that were used in the forward pipeline. Without the reverse pipeline we do not have enough information to always correctly compute the delayed copy of the inputs that are required to non-dissipatively reset the stages in the forward pipeline. The penalty of erroneously computing a delayed copy of these inputs is to dissipate energy similar to conventional CMOS for every erroneous bit. Unfortunately, there are situations in which providing the inverse of a function in the forward pipeline is cumbersome. Luckily all is not lost since in most of these cases we could apply a number of techniques that would make the dissipation associated with this irreversibility minimal. We must stress here that even though we might allow the breaking of reversibility at certain selective points in the system with all of the undesirable effects that we have mentioned above, we still insist on employing reversibility throughout the majority of the system. This is in contrast to proposals that do not employ reversibility anywhere in their systems and hence are faced with the undesirable effects at the majority of the nodes in their systems.

4.8.1 Irreversibility Is Not Free

Before we describe how to reduce the dissipative effects of irreversibility we have to warn that irreversibility is not free. The best we can do is reduce the energy penalty associated with irreversibility. We can never eliminate it. A system containing irreversible elements is not a system in which the energy of the system can be asymptotically reduced without a lower bound. The packets of energy that are dissipated at the points where reversibility is broken in the system set a non quasistatic limit on the minimum energy dissipation of the system regardless of operating frequency. However, in environments where we are stuck, the following is included to aid in reducing the cost of increasing the information entropy of the system.

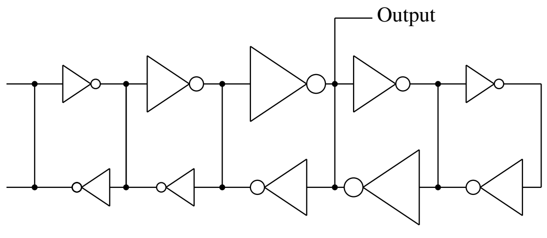


Figure 4.12: Scaling down stage sizes before breaking reversibility in SCRL.

4.8.2 Statistically Controlled Irreversibility

The first technique relies on the observation that we are mainly concerned in reducing the *average*, *not the instantaneous*, power consumption. Gates that computed the inverse function of the gates in the forward direction produce a correct copy of the inputs all the time. Without these inverse gates, we cannot guarantee to be correct all the time. In certain applications however, we can guarantee to be correct *most* of the time. Since dissipative events only occur whenever we guess wrong, being correct most of the time results in substantial energy savings when compared to conventional CMOS without the need for reversibility. To illustrate this we consider an example of an 8-Input NAND gate. This gate outputs a FALSE if and only if all the inputs where TRUE. Otherwise, this gate outputs a TRUE. Assuming that the input bits are random, the probability of the output of this gate being at TRUE is $255/256 = 0.996$. If we always assume the output to be at TRUE, then we will have a dissipative event, caused by a wrong prediction, only 0.3% of the time. In the pipeline in Figure 4.3 let the F_1 be this 8 input NAND gate. Then we can omit F_2^{-1}, assume that this omitted inverse gate output a FALSE all the time, and be right 99.6% of the time. This could be important in situations in which the computation of F_2^{-1} is not feasible or otherwise cumbersome.

4.8.3 Where to Break Reversibility

The second technique concerns the way multi-stage buffering is done in CMOS and in SCRL. To drive a large load in CMOS, one must go through a number of progressively larger devices with each device driving another that is slightly larger than itself until the last one in the chain is large enough to drive the load. In SCRL, each larger stage is paralleled by another stage of comparable size in the reverse direction. If reversibility were broken immediately after the largest stage then dissipation would be large because the fact that the input capacitance of the large reverse gate is significant. To alleviate the problem, we must proceed with the pipeline beyond the last stage with inverters in the forward and reverse direction scaling down the size at each successive stage until we reach the minimum size possible. If reversibility is broken immediately after this minimum size stage, dissipation is minimized due to the much smaller input capacitances of the reverse stage. This is shown in Figure 4.12.

Part III

Implementation and Testing of SCRL-1 Demonstration Chip

5. Demonstration Chip Details

5.1 Introduction

To verify the quasistatic operation and quasistatic behavior of Split-Level CRL, we have fabricated and tested an 8×8 CMOS multiplier chip employing the circuit techniques of Split-Level Charge Recovery Logic. Split-Level CRL was the obvious choice for implementing the demonstration chip because of its simplicity and closeness to conventional CMOS circuits. In this section, I will describe the internals of this demonstration chip. In the following section I will describe the measurement techniques that I used to verify the lower energy consumption of this chip as well as report the results of those measurements.

5.2 High Level Multiplier Design

Before we examine the available alternatives to building a multiplier, I will first review the needed operations to multiply two binary numbers. Let us assume that we want to multiply $A = [a_3, \cdots, a_0]$ by $B = [b_3, \cdots, b_0]$ to produce $C = [c_7, \cdots, c_0]$. The product C would then be

$$C = (b_0 \times A) + (b_1 \times A) + (b_2 \times A) + (b_3 \times A) \tag{5.1}$$

Since in each of the product terms, b_i is a binary bit with a value of either 0 or 1, the multiplication could be carried out by bitwise ANDing b_i with each bit of A. In addition, we note that all the partial products are 4 bits wide and therefore we only need three 4-bit adders to perform the multiplication. We must of course correctly position the consecutive adders so that each addition is left justified to reflect the significance of the b_i bit. So in essence the multiplication of binary numbers involves nothing more than repeated justified additions of the partial products. There are a number of different ways one can build an $N \times N$ multiplier that will reflect the above procedure. The variations come from the way one performs the additions since the calculation of the partial products have already been reduced to trivial bitwise ANDing. As is widely known, the critical path in any adder is the time it takes to propagate the carry all the way across the width of the sum. This is because when adding two numbers, the most significant bit could be affected by the sum in the least significant bit. Simplistically, one would perform each addition allowing enough time for each addition to completely finish, that is allowing enough time for the total carry propagation, before proceeding to add the next partial product. Fortunately, the carry effect is one-directional and bits of higher significance never affect the lower significant part of a sum. With this observation, we see that we need not wait for each addition to fully complete before commencing the next one. More specifically, we can start the following addition as soon as the previous one produces the least significant bits that the next adder needs. As long as the carries in the previous addition are at least one bit ahead of the carries in the current addition, both can proceed concurrently. Indeed this is how most fast

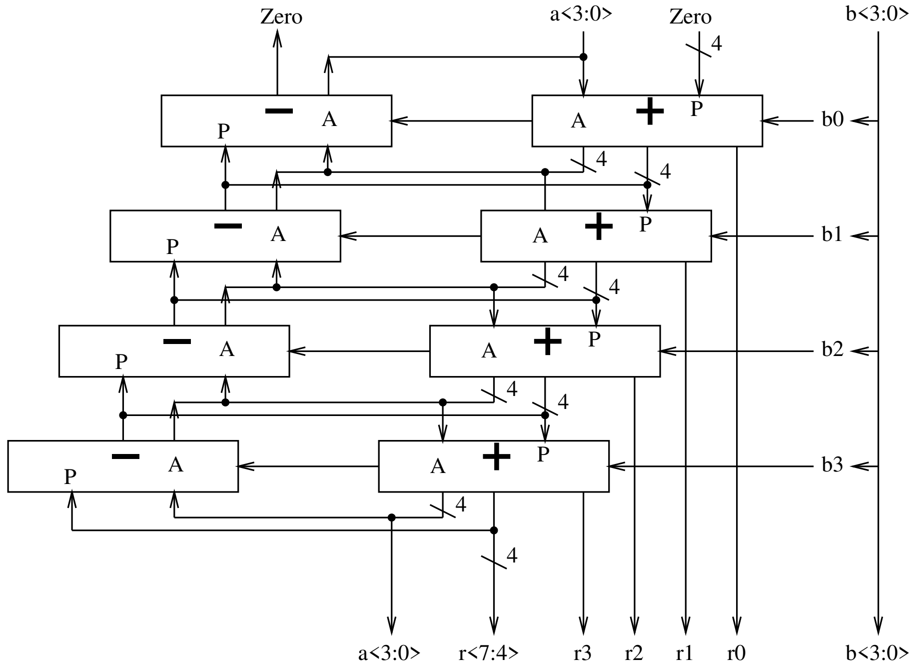


Figure 5.1: Block diagram of a 4-Bit SCRL multiplier.

multiplier are built today. The way in which this overlap is accomplished is the origin for the multitude of options for fast $N \times N$ multipliers.

Unlike conventional CMOS, an SCRL multiplier needs to compute the partial products in both the forward and the reverse directions. For an $N \times N$ multiplier constructed from $N - 1$ adders, the inverse functions needed in the reverse pipeline are nothing but N-bit subtractions. Figure 5.1 illustrates the organization of an SCRL 4-Bit multiplier. Briefly, the "+" blocks take in a partial sum P, the operand A and the corresponding bit from B. The block bitwise AND's each bit of A with the supplied b_i and then sums the result with the partial sum P. In addition to producing the sum, each adder block passes on a copy of A for use by subsequent adders. In practice, the path of operand B contains synchronizing reversible registers. They are omitted from the diagram to reduce clutter. The subtraction blocks are identical to the addition block except that they subtract ($b_i \times A$ from the value presented to their P inputs. Since the P inputs to the first adder are zero, the first adder and the corresponding subtractor could be eliminated.

Because of the need for the inverse computation, our design cannot easily take advantage of the ability to overlap the carry propagation in the consecutive adders. The reason for disallowing the overlap is that it complicates the book keeping necessary to perform the needed subtractions and requires a large number of additional intermediate registers. For this reason, each horizontal adder in our implementation is allowed enough time to completely finish its operation before the following adder can proceed.

In a conventional setting, not being able to overlap the carry propagations leads to slower performance with no additional advantage. Fortunately, this is not the case within the context of SCRL technology. We remind ourselves that the only reason we employ SCRL circuit techniques was to reduce energy dissipation. As we explained in Section 2, energy savings occur only as a result of operating the chip below the maximum possible operating frequency. For this reason, the clock frequency of circuits employing SCRL techniques is always smaller than the maximum possible, more than 10 time slower in most cases.

In conventional CMOS, the designer is mostly concerned with minimizing the *worst case* propagation delay of the critical path of the circuit in order to meet the specified operating frequency target. In SCRL however, the clock period is much larger than the worst case propagation delay of the circuit in order to yield the promised energy savings. For this reason, having to wait for the completion of previous additions before proceeding with subsequent ones will not result in erroneous operation as there is a large margin between operating frequency and delay. Slower circuits however do mean more dissipation since the energy dissipation factor is inversely proportional to the ratio of the operating frequency period to the *average* propagation delay. Note that in SCRL, it is the average and not the worst case delay that is of interest. This is because, unlike conventional CMOS, there a purposely introduced margin between the worst case delay and the operating frequency to save energy. The concern in SCRL towards slow circuits is that they reduce the energy saving advantage of SCRL by eating away at this introduced margin. Since our concern is with the average energy consumption, our interest is therefore in the average circuit delay.

From the above we see that SCRL circuits should be optimized to minimize the average, and not the worst case, propagation delays. Having said that, we note that when the average propagation delay is taken as the measure of comparison, the difference in performance between allowing or disallowing the overlap of the propagation of carries in the adders of our multiplier becomes insignificant. This is because on the average, given random data, a carry propagates only 1.6 bit positions to the left. The pathological case of the carry propagating from the least significant position to the most significant position occurs very infrequently and therefore a non-overlapped implementation would on the average be only 1.6 times slower than the heavily pipelined one.

Because of the above reasons, we choose the non-overlapped implementation since it greatly simplifies the design of the inverse pipeline while not compromising power dissipation.

5.3 Multiplier Details

In this section I will attempt to describe the internal details of the 8×8 multiplier demonstration chip we call SCRL-1. The description will follow the hierarchy of the design from the high level blocks and would proceed on to describe the details of their components and sub components.

At the highest level of the hierarchy, the multiplier consists of 12 reversible pipeline levels. All reversible levels share the property that each contains one and only one reversible pipeline stage. A reversible pipeline stage is defined as a block containing exactly one pipeline stage which computes a logical function and proceeds in the forward direction, accompanied by a matching pipeline stage which computes the inverse of that function and which proceeds in the reverse direction. SCRL-1 was implemented using SCRL circuit techniques that were describes in detail in Section 4. More specifically, each pipeline stage in SCRL-1 chip is a 2-level SCRL implementation that was described in Section 4.5.

5.3.1 Reversible Level 1

This is the first level in the multiplier pipeline. The two operands A and B are fed directly from the chip pins to this level through ESD-protected input pads. The structure of these input pads is slightly different from those used in CMOS and will be described in a later section.

Throughout SCRL-1, operand A is repeatedly presented to the inputs of the 8-bit adders to be conditionally added to the partial sum computed so far depending on the value of the bit b_i from B. In general, adders contain a number of XOR gates operating on every input bit to produce the partial sum or to compute the propagate-generate signals that are needed for processing the internal carries. Noting that an XOR is a gate that computes $(a\overline{b} \vee \overline{a}b)$, we see that providing the inputs to the adders in both their true and complement forms greatly simplifies the design of the adders. In SCRL-1 all the inputs to the adders are provided in their true and their complement form. In addition, the sums produced from each adder are also in dual form since a partial sum from one adder is used as an operand by the next adder. However, SCRL-1 has input pins only for the true copy of the operands. For this reason, the function of reversible level 1 in SCRL-1 is to generate the needed complements of the signals on the input pins and to feed the true and complement copies simultaneously to the adder in reversible level 2 of SCRL-1.

In any multiplier, the first adder retires the two product terms

$$PartialSum = (b_0 \times A) + (b_1 \times A)$$

since at this point the partial sum is zero. We therefore see that the first adder requires bits b_0 and b_1 from operand B to perform its function. For this reason, reversible level 1 provides the dual true and complement copies of only these bits and passes the remainder bits of B in only their complement form. The reason for complementing the unused B bits is that an SCRL inverter needs half as many devices as an SCRL buffer.

Reversible level 1 does not have any logic for the inverse pipeline stage in the reverse direction. This is because the output from a reverse pipeline stage is only needed to eliminate energy dissipation in the previous pipeline stage. Since reversible level 1 is the first level in the SCRL-1 chip, and since we are not concerned with the energy dissipation in the pin drivers to SCRL-1, the reverse component of the pipeline in this level was omitted.

Figure 5.2 shows the block diagram of reversible level 1. It takes in the inputs $bIn\langle 7{:}0\rangle$ and $aIn\langle 7{:}0\rangle$ corresponding to operands A and B, and outputs $bOut\langle 0{:}3\rangle$, $nbOut\langle 7{:}2\rangle$ and $aOut\langle 15{:}0\rangle$. $bOut\langle 0{,}2\rangle$ are the true and complement copies respectively of $bIn\langle 0\rangle$ produced by the module `splitReg1_16 (I10)`. Another copy of `splitReg1_16 (I9)` produces the true and complement of $bIn\langle 1\rangle$. $bIn\langle 7{:}2\rangle$ are inverted by copies of `inv1_1` module to yield $nbOut\langle 7{:}2\rangle$. Finally, the module `splitReg8_2` takes in $aIn\langle 7{:}0\rangle$ and produces $aout\langle 7{:}0\rangle$ which are the true copies of the inputs and $aOut\langle 15{:}8\rangle$ which

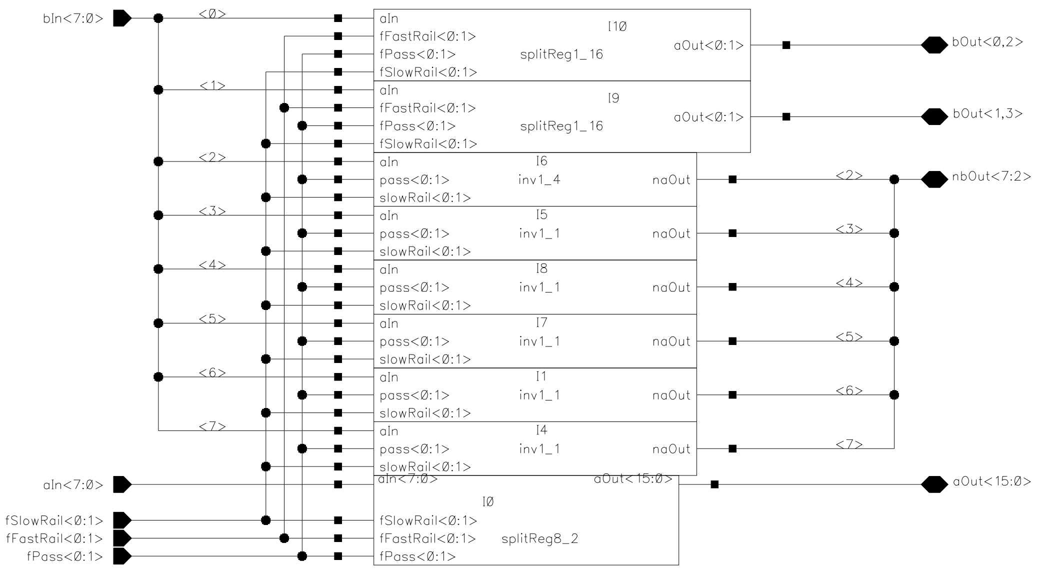


Figure 5.2: Schematic diagram of reversible level 1.

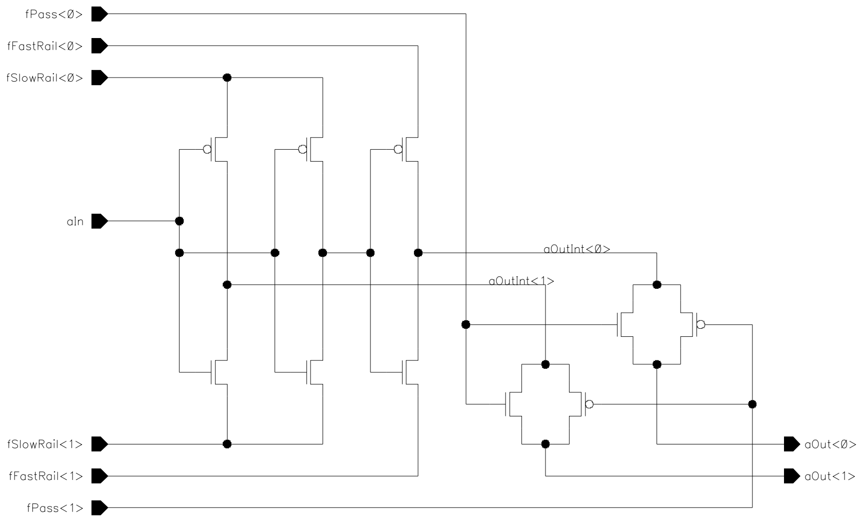


Figure 5.3: Schematic diagram of `splitReg1_16` module.

are the complement. Internally, `splitReg8_2` contains 8 copies of a module similar to `splitReg1_16` except for device sizing.

Figure 5.3 shows the design of the `splitReg_16` module. We examine it in detail to illustrate the structure of a typical SCRL gate. From the left, the first two P-Channel devices and the first two N-Channel devices are part of the first stage of this 2-level SCRL gate and are controlled by the slow rails. They are followed by a CMOS inverter that is part of the second logic stage which is needed to generate the non-inverted signal and are controlled by the fast rails. The remaining devices make up the pass gates that are needed at the output of every SCRL gate and are controlled by the pass rails.

Throughout the diagrams of SCRL-1, `xxxxRail<0>` indicated the positive going rail while `xxxxRail<1>` indicates the negative swinging one. The same is followed for the `pass` rails. The prefix `f` in the rail name indicates that this rail controls a component that is part of the forward pipeline, while the prefix `r` indicates that the rail controls a component that is part of the reverse pipeline. Since reversible level 1 does not have a reverse component, there are no `r` prefixed rails in the diagram. Examining the diagram one could see that the two inverters in the first stage perform the same function and hence might be redundant. This is not the case for the `splitReg1_16` module. The replicated devices are included to maintain the symmetry of the drive capabilities for the two outputs of the module.

5.3.2 Reversible Level 2

Reversible level 2 is the first level that contain inverse blocks for every function in its forward pipeline. Reversible level 2 performs the following functions:

- It produces the true and complement of $nbIn$<2> that will be needed in the next level.
- It passes the other bits of B for later levels.
- It perform the addition of the first two partial products.

Figure 5.4 shows the schematic for reversible level 2. Since this level does contain the inverse pipeline stages, each module in the schematic is a module containing both the forward as well as the reverse components of the pipeline. For example, the module `revInv1_1` not only inverts the unused B bits and passes them on to the next reversible level, it also reads the levels at its output and later computes the "inputs" as sequenced by both the forward and the reverse control rails. As an example, I include Figure 5.5 which contains the schematic for the `revInv1_1` module. Since an inverter is the inverse function of itself, this module contains an inverter module `inv1_1` in the forward direction and another `inv1_1` in the reverse direction. The forward one is controlled by the forward control rails and the reverse one is controlled by the reverse control rails. At this point I wish to emphasize that even though the names aIn and $bOut$ reflect the flow of data in the forward direction, each line is now both an input and an output depending on where we are in the cycle.

Module `revfirstadd8` in the schematic performs the addition of the first two partial products and produces the partial sum in both true and complement form on $rOut$<3:0> and $sOut$<15:0>. $rOut$<3:0> correspond to the true and complement of the two least significant bits of the result and $sOut$<15:0> correspond to the 8 most significant bits of the result. We get 10 bits from the first addition because one of the partial products is left justified by one position to reflect the significance of b_1. The true copies from $rOut$<0:3> correspond to the first two bits of the total product and are not used in the next additions. $sOut$<15:0> are all used in the subsequent addition. Since we are passing both A and B

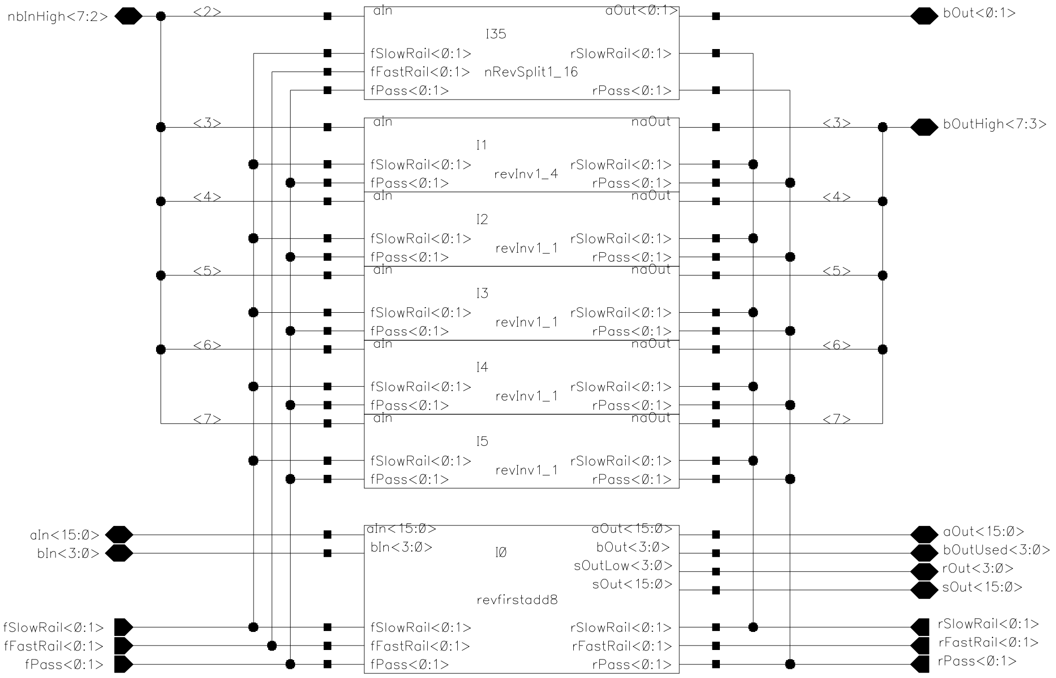


Figure 5.4: Schematic diagram of reversible level 2.

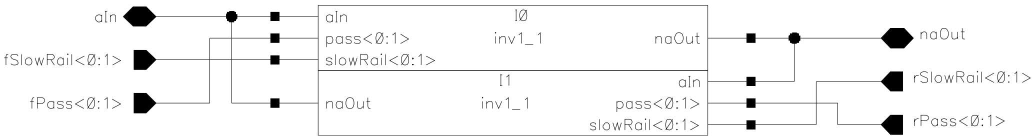


Figure 5.5: Schematic diagram of `revInv1_1` module.

along so far, we do not need to compute the inverse of this addition to produce the delayed copy of the inputs. This is because we already have these inputs in the registers that are passing A and B them.

The 8-bit adders used in SCRL-1 consist of a cascade of 8 1-bit full adders with multiplying AND gated at the inputs to make an 8-bit partial product adder. In general a 1-bit multiplying full adder could look something like Figure 5.6-a. `I0` performs the bit-wise ANDing to produce the partial product. `I1` generates the sum without including the carry-in which is also serves as the carry Propagate signal. `I2` produces the carry Generate signal. The full sum comes out of `I3` while the carry for the next stage is produced by `I5`. The trouble with this circuit is that an 8-bit adder built using it contains a carry path that is roughly 16 logic levels deep. Earlier we have stated our desire that each SCRL-1 adder completely propagate all the carries before the next one is to proceed. The problem is that each stage of SCRL-1 pipeline is at most 2 logic levels deep. We solve this problem

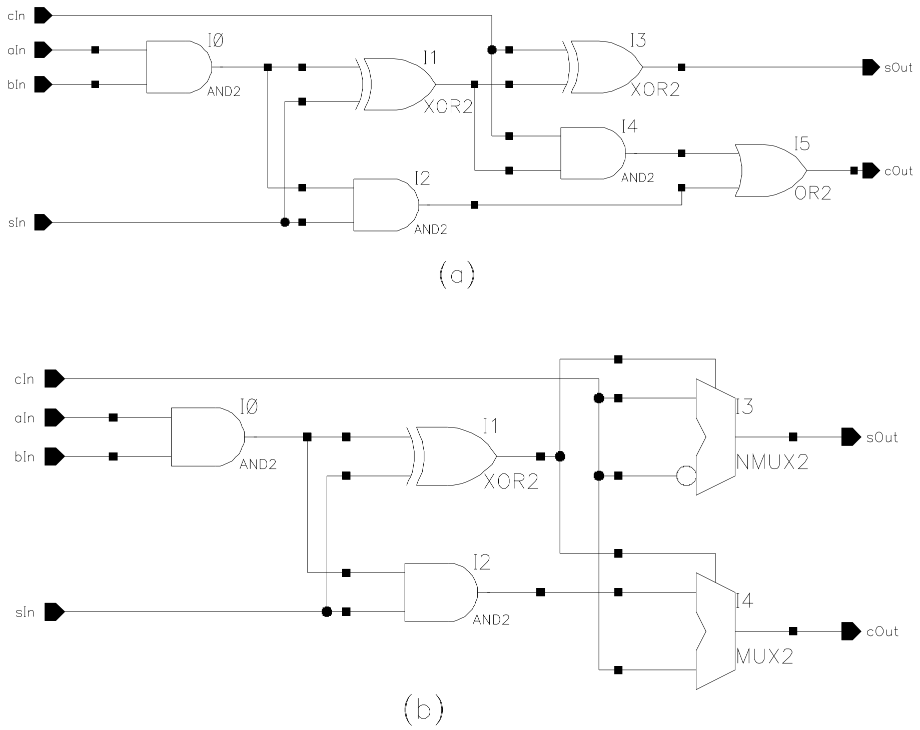


Figure 5.6: (a) Schematic Diagram of a generic 1-bit propagate-generate full adder. (b) Diagram of the 2-level 1-bit adder that are used in SCRL-1 chip.

by redesigning the adder to look like the one in Figure 5.6-b.

In our implementation of SCRL-1, `I0` is lumped with `I2` into one logic level by expanding the terms of the XOR. This produces the carry Propagate signals in one logic level. The action of the logic producing this Propagate signal is of course controlled by the slow rails of the stage. With the Propagate signal generated and stable, the multiplexors `I3` and `I4` are ready to receive signals on their inputs without any dissipative events. At this stage the action of the fast rails activate the logic of `I2`. In addition, the carry-in for the least significant 1-bit adder in the 8-bit chain is also driven active by the action of the fast rails. With this in mind, we see that SETting the fast rails succeeds in producing all the intermediate carries as well as the correct 9-bit result of the addition. The negation at the input of the multiplexor `I3` contains no logic since all the signals are available in their dual form.

With random data, this adder is just as fast as carry-save adders and since the network that the fast rail activates is set up in advance, this adder is not much slower even in the cases where the carry has to propagate for a few places.

In reversible level 2, module **`revfirstadd8`** contains 9 1-bit adders. This is because this adder retires the first two partial products producing a sum that span 10 bits. For this reason also, the adder in this level differs from other adders in SCRL-1 in that it has enough AND gates to produce two partial products instead of one.

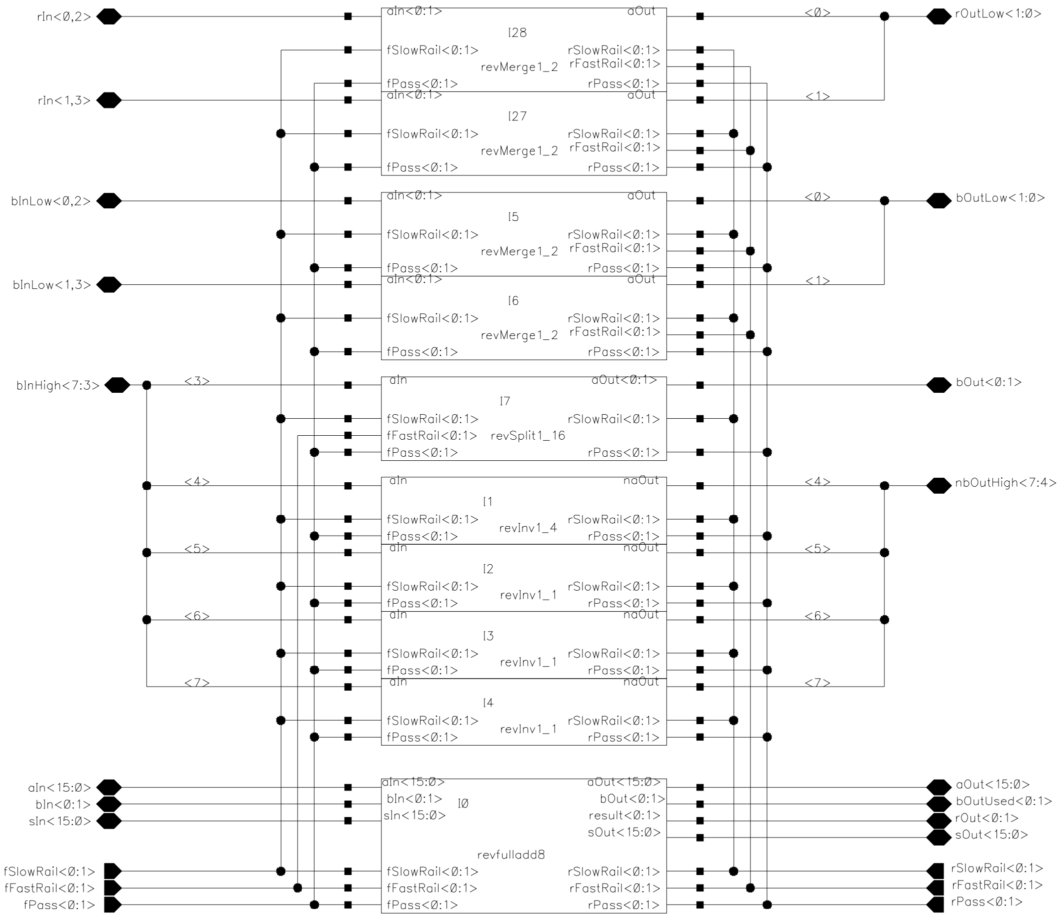


Figure 5.7: Schematic diagram of reversible level 3 module.

5.3.3 Reversible Level 3

Figure 5.7 shows the block diagram of reversible level 3 of SCRL-1 chip. All the sub blocks in the diagram are familiar excepts the instances `I28, I27, I5` and `I6` containing the new module `revMerge1_2`. The function of this new module is quite simple. Presently all the outputs from the previous level are present in their dual forms. However some of these bit are not going to be used anytime later. These include the bit from B that were used in the previous product as well as the least significant bits of the total product. For these signals, it is a waste in circuit area, power consumption as well a wire area to keep them in dual form. For this reason the `revMerge1_2` modules were added in the path of these signals to take in the dual form and output the true copy and to do it in a reversible manner.

As a side note, module names differing only in the numbers appearing at the end of their names are identical in function. For example modules `revInv1_1` and `revInv1_4` are identical in function. The difference in their numbers reflect a difference in their device sizing and in their driving capability.

In this level, module `revfulladd8` is an 8-bit reversible adder. It contains an adder in the forward direction that conditionally sums A to the partial sum S based on the value of b_2. In addition it contains a subtractor that takes the output produced by the adder in the forward direction and conditionally subtracts from it A to produce a delayed copy of the inputs to this reversible level. Please note that `revfulladd8` like `revfirstadd8` contain reversible registers to pass through a copy of A to the next reversible level.

5.3.4 Reversible Levels 4-8

Reversible levels 4 through 8 are all very similar to reversible level 3. Some modules in these reversible levels have names similar to the familiar `revSplitx_x` or `revMergex_x` modules except that they are prefixed with the letter `n`. For split type modules, this indicates that the input data is in complement form. For for the the `merge` type modules, this indicates that the output of the merged true and complement is the complement copy. This is necessary so that the merged output would agree in polarity with the other bits that were merged previously and have since been going through an inversion every pipeline stage.

5.3.5 Reversible Level 9

Figure 5.8 shows the block diagram of reversible level 9. In SCRL-1, the addition of the last partial product is performed in reversible level 8. Since this addition is the last to be performed, the dual form of A is no longer necessary. In addition, since the sum out of this addition is not going into a subsequent adder, its dual form is also not needed. For this reason, reversible level 9 contains two 8-bit merging blocks, `revMerge8_2`. One is used to merge the bits of A while the other is used to merge the sum bits to yield the 8 high bits of the total product. As shown in the diagram, reversible level 9 has some additional `merge` and `inv` blocks to merge and pass both the B bits as well as the least significant bits of the total product.

In the diagram, $rOutLow$<7:0> are the least significant bits of the total product while $rOutHigh$<7:0> are the most significant bits. Also $bOut$<7:0> and $aOut$<7:0> are true copies of the operands A and B respectively.

5.3.6 Reversible Level 10

Figure 5.9 shows the block diagram of reversible level 10. This level performs two functions. The first function is to take the 16 bit product produced by the previous level and to increase its drive 16 times through the `revBuffer1_16` modules in preparation for driving the very large devices at the input of the output pad drivers. Internally, each `revBuffer1_16` module consists of a non-inverting SCRL buffer made of two levels of inverters. The drive is increased by 4 at each of the two inverter stages to achieve a drive of 16. The output bits $rOut$<15:0> are routed directly to the output pin drivers. They are also routed to reversible level 11 for a reason to be discussed later.

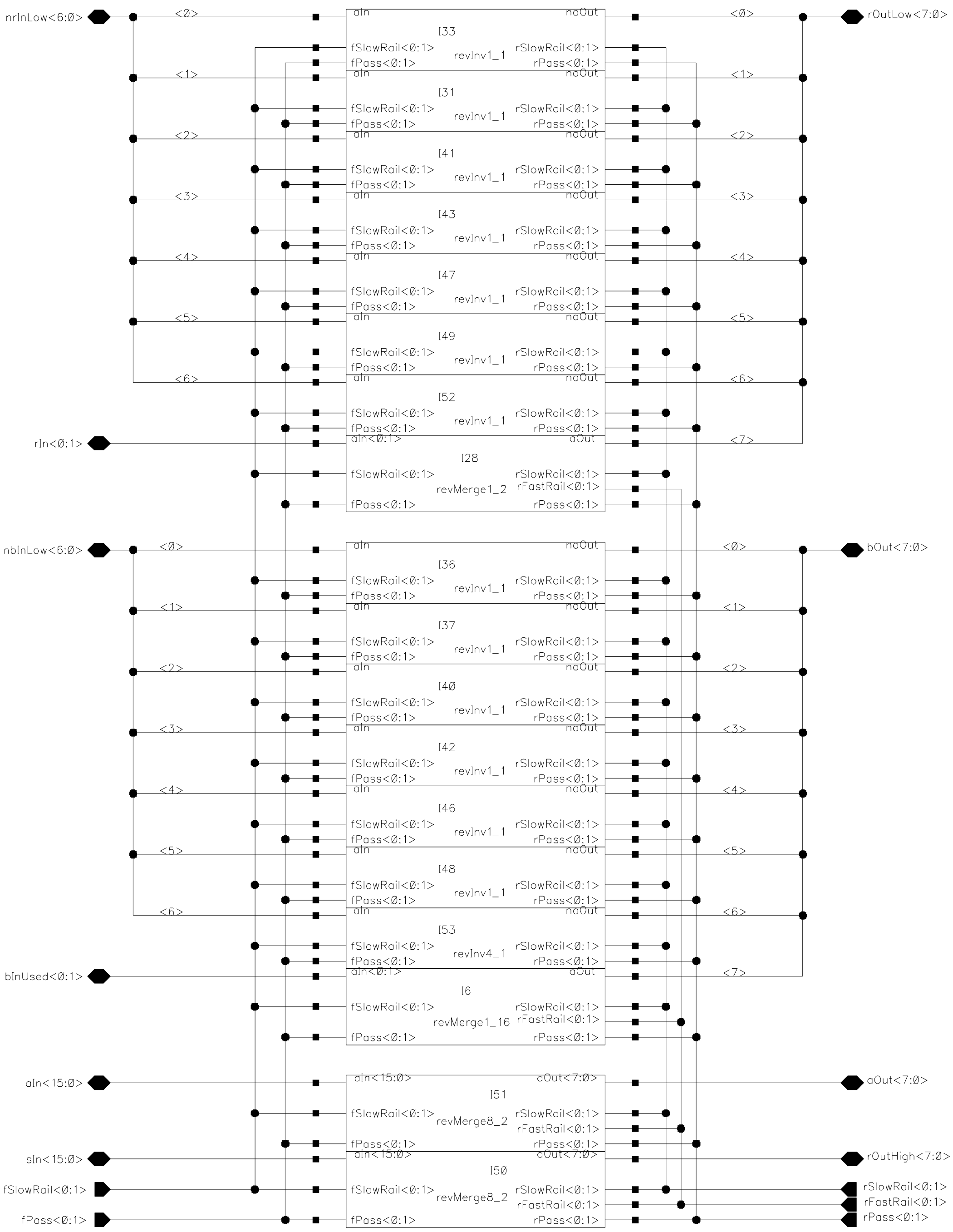


Figure 5.8: Schematic diagram of reversible level 9 module.

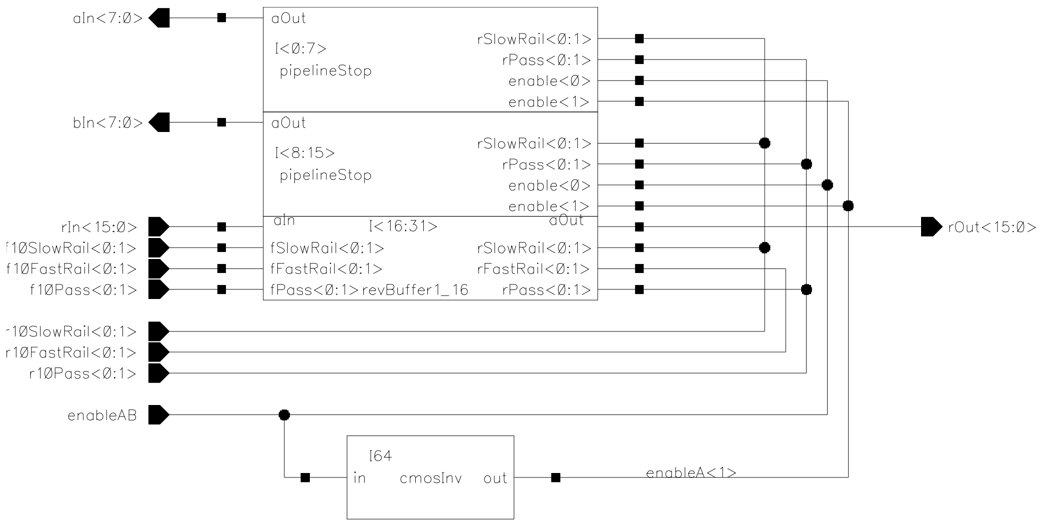


Figure 5.9: Schematic diagram of reversible level 10 module.

The second function of reversible level 10 is to non-dissipatively terminate the pipelines carrying the bits of A and B. Ideally, these bits should be routed to output pins of SCRL-1, and would act as inputs for the reverse pipeline during certain operating phases of the chip. Without that, reversibility would be broken and bits of A and B arriving at this level would "fall on the floor" thus generating information entropy and dissipating CV^2 per bit dropped. Unfortunately, the pin budget of SCRL-1 precluded routing A and B to the chip pins. In practice, one would not worry much about this dissipation as it would be small compared to the total dissipation of the chip. However, since SCRL-1 was designed to measure the energy saving of fully reversible circuits, reversibility had to be "faked". While testing SCRL-1, the input patter would be held constant yielding the same result each cycle. Therefore, if we latch the values of A and B in some static register, we can later use them to drive the reverse pipeline at the place where it was terminated thus simulating reversibility and avoiding dissipation. It is this function that the modules `pipelineStop` perform in reversible level 10. The enable signal in an input pin in SCRL-1 and is controlled to lock-in the value to be used to drive the terminated pipelines when needed. Under normal operation, these latches could be filled with values that are more likely to occur based on the non random nature of the inputs. In the event that a latch value equals the incoming bit, SCRL behavior is preserved and dissipation is eliminated. Therefore in so far as the latched pattern accurately predict the input values, dissipation is reduced.

5.3.7 Reversible Level 11

Figure 5.10 shows the block diagram of reversible level 11. In a true SCRL implementation, the pins carrying the result out of the SCRL-1 chip would also act as inputs to the reverse pipeline at some time during the cycle. This is because each line in SCRL is really both an output for one direction of the pipeline and an input for the other. Adopting this in

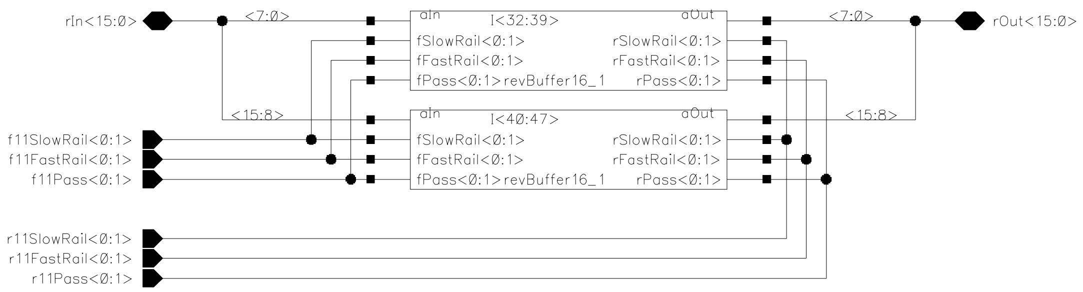


Figure 5.10: Schematic diagram of reversible level 11 module.

SCRL-1 would have meant that the circuit connected to the output pins of SCRL-1 should also be of the SCRL type, so as to drive the reverse pipeline when needed. Since SCRL-1 is the only SCRL chip around, we needed the output pins of SCRL-1 to be true output lines with no need to drive them from the outside world to preserve the functionality of the internal reverse pipeline. With no signal driving the reverse direction, the pipeline carrying $rOut$<15:0> is now terminated with a break in reversibility. This is the same problem that we had when terminating the pipelines of A and B and to that end we use the same `pipelineStop` modules we had used before to reduce the effect of this break in reversibility.

Their is one difference from the previous case here however. The lines carrying $rOut$<15:0> have a drive of 16 at this point. This leads to 16 times more dissipation than the case with A and B whenever the prediction in the latches disagrees with the current value of the data. For this reason, we postpone the use of `pipelineStop` modules for reversible level 12 while using this level to *reversibly* reduce the drive of these signals to minimum according the technique described in Section 4.8.2. This reduction is done in the `revBuffer16_1` modules shown in the diagram.

5.3.8 Reversible Level 12

As mentioned in the previous section, this level contains 16 `pipelineStop` modules to reduce the dissipation of breaking reversibility at the end of the result pipeline.

5.3.9 Rail Connections

Figure 5.11 shows the connections of the control rails in SCRL-1. The module `revMult8` contains within it the 12 reversible levels described above. Up to this point, we have the ability to clock SCRL-1 in accordance with any of the clocking variants that were described in Section 4. However, this would require routing all the control rails of the `revMult8` module in Figure 5.11 to pins on the chip package. Restricting the ability to clock the chip to one clocking variant considerably reduces the needed pins. For this reason, I decided that SCRL-1 would only accept the 4-phase clocking scheme described earlier. In this scheme, the rails controlling level N in the pipeline have the same timing as the rails in level $N + 4$ and could therefore be connected to them. Examining Figure 5.11 we see

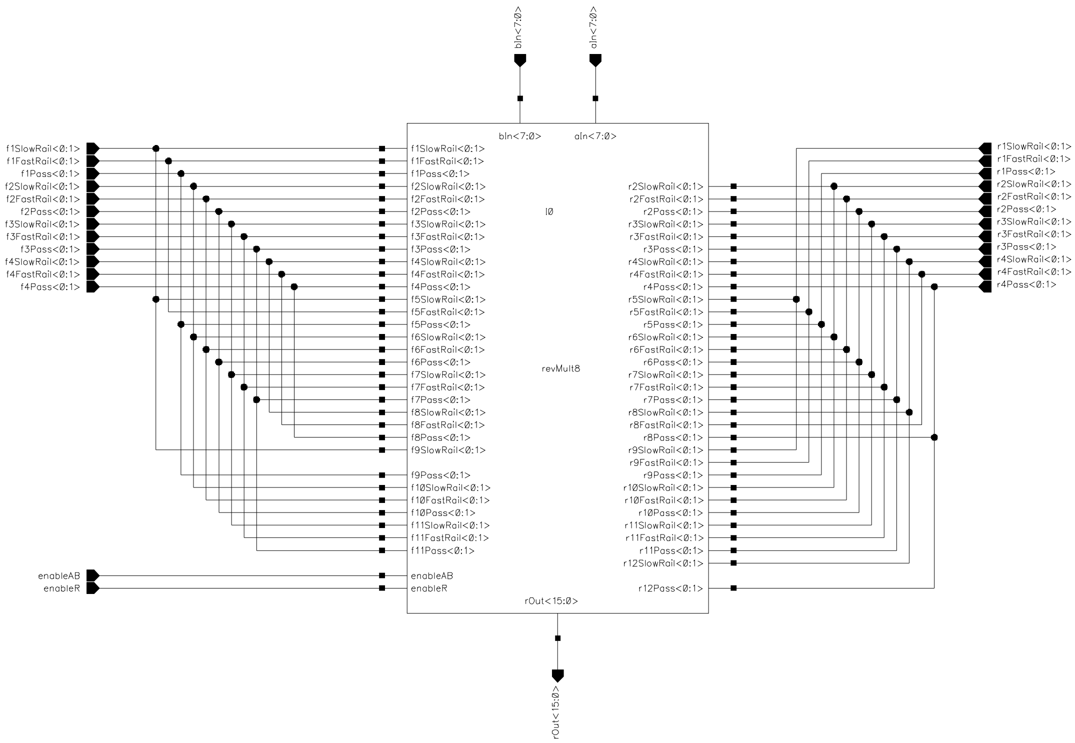


Figure 5.11: Schematic diagram illustrating the connection of the rails in SCRL-1.

how such connections reduce the number of pins that are needed for the control rails. In the diagram, the signal *enableAB* is the enable signal that controls the latches in the `pipelineStop` modules of A and B while *enableR* controls the latches for *rOut* lines.

5.3.10 I/O Connections

Figure 5.12 shows the input/output connections of SCRL-1. This diagram is at the highest level of the schematic hierarchy of SCRL-1. The internal details of module `rails1` in the diagram are the contents of Figure 5.11. In this diagram we see the output pad driver module, `outputPad`. We also see the modules `inputPad` which contain the ESD protection for the inputs to the chip. If we add the I/O lines of the `rails1` module we get a count of 82. Adding to that the separate rails that drive the output pads, *padsSlowRail*<0:1>, and the implicit V_{dd} and GND lines to bias the P-Wells and N-Wells of the chip we get 86. Unfortunately the MOSIS package provided for the size of the die we are using had only 84 pins. To fit within this package, I eliminated the most significant bits from the A and B operand inputs to SCRL-1 and internally shorted aIn<6> to aIn<7>, and bIn<6> to bIn<7>. While SCRL-1 is internally an 8×8 multiplier, externally we can only feed 7 bits of each operand.

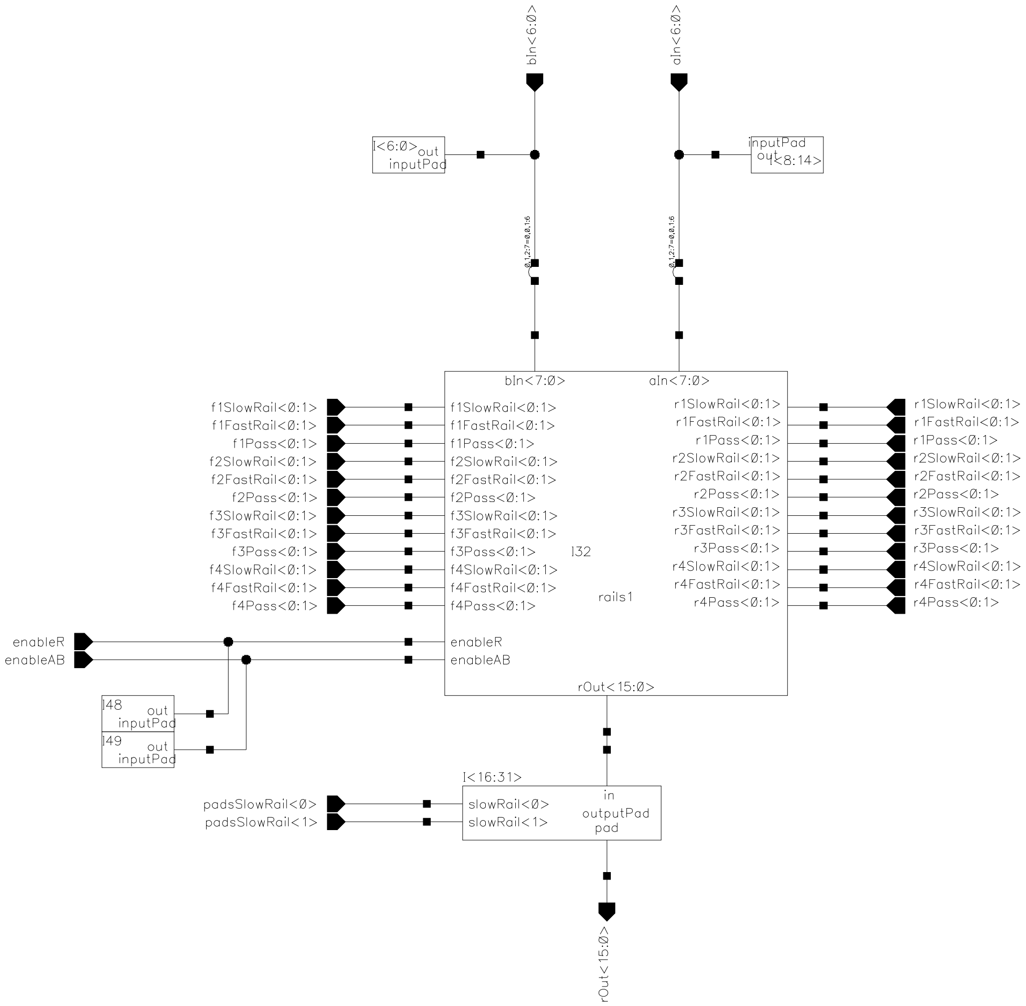


Figure 5.12: Schematic diagram illustrating the I/O connections in SCRL-1.

5.4 Input Pads

SCRL-1 contains a number of input pins with differing requirements. Different input pads are used by fast and slow rails, by pass rails, and by operand inputs. Each with its own unique ESD protection requirements. The competition in SCRL is between maintaining adequate ESD protection while still minimizing energy dissipation because of large diffusion resistors.

5.4.1 Operand Input Pads

These are the pads used to input A and B. They only drive the gates of some devices in reversible level 1. Since very little current is driven through these rails, indeed the current could be zero for static input patterns, a relatively large polysilicon resistor is included in the path. This is followed by 2 reverse-connected ESD protection diodes and finally the signal drives the gates of two large complementary transistor with the source and drain of each shorted together and connected to the respective V_{dd} or GND. The function of these devices is to position a large capacitance at the end of the polysilicon resistor thus increasing the RC time constant of the input path and allowing the diffusion diode time to conduct.

5.4.2 Slow and Fast Rails Input Pads

Both the slow and the fast rails use the same pad design. Note here that the slow and fast rails are the rails through which all the internal node of the computing circuits are charged and discharged. Adding a resistor in their path directly affects the energy consumption of the chip. Fortunately, these rails are connected to the sources and drains of a large number of devices which result in a large RC time constant. Because of the above reasons, the input pads for these rails contain only ESD diffusion diode and direct metal path does exist between the bonding pad and the rails' distribution networks.

5.4.3 Pass rails Input Pads

We remind the reader here that the pass rails are used solely to control the states of the pass gates at the output of every SCRL gate in the circuit. Since this means that a pass rail directly drives the gates of a number of devices, we have to be more careful in our ESD protection to prevent damage. For this reason, the pass rails' input pads include a $\simeq 20\Omega$ polysilicon resistor to slow down transients enough for the reverse biased protection diodes to conduct.

5.5 Output Pads

Figure 5.13 shows the schematic diagram of the output pads used in SCRL-1. It consists of two devices large enough to drive the pin and load capacitance of the output. As for the rest of SCRL-1, these devices are controlled by two swinging control rails and are not connected to V_{dd} or GND. Following those devices are two back biased devices to provide ESD protection. As in conventional CMOS, little added protection is needed in true output pads because the relatively large size of the driver itself provides adequate protection. The back biased devices were added for the output pads in SCRL-1 to provide a conduction path to V_{dd} or GND. This is because the devices in an SCRL pad driver have no connection to either V_{dd} or GND.

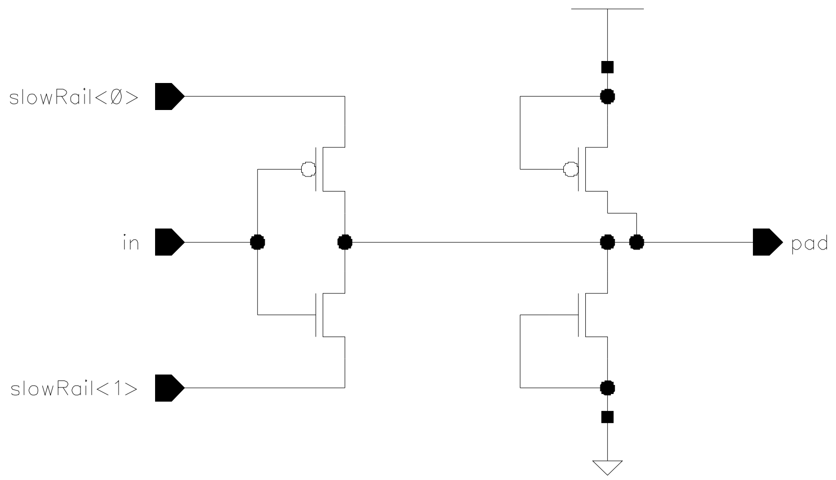


Figure 5.13: Schematic diagram of SCRL-1 output pad circuit.

5.6 Layout of SCRL Circuits

Figure 5.14 shows the typical layout for an SCRL chip. Similar to laying out conventional chips, the layout of SCRL chips starts with the design of the common cells. Unlike conventional CMOS however, the cell library in SCRL must contain additional cells that compute the inverse functions of the modules in the cell library. Figure 5.14 shows how each forward computing cell is paired with its inverse cell which is part of the reverse pipeline. Functionally, each forward cell is always paired up with the same inverse cell. Furthermore, the outputs and inputs of each forward cell always go to the corresponding inputs and outputs of the reverse cell. Because of these two facts, the layout synthesis of the cells in SCRL-1 was performed on design schematics that already contained the circuits for both the forward and the reverse sub cells. This increases the room for layout optimization without affecting the generality of the library since no sub cell is ever used without its inverse. It also cuts down on the wiring because the I/O connections between the forward and the reverse cells are optimized internal to the encompassing cell. We define the library cells that contain both the forward and reverse circuit components as reversible cells.

Reversible cells that run horizontally next to each other and that share the same forward and reverse control rails are called a reversible level, *e.g.,* level 1. Figure 5.14 shows a sketch of the layout for a six level SCRL pipeline. In the figure, each level contains three separate reversible cells. A 2-stage SCRL reversible cell needs 6 forward control rails to control the forward part of the reversible cell and 6 reverse control rails to control the reverse part. The cell also needs V_{dd} and GND to bias the substrate and the wells in its circuits. Since these signals are common to all of the reversible cells on the same level, the distribution of these rails occurs along channels that are parallel with each reversible level. For example, `R_Rail 1`, which refers to the 6 control rails that are needed by the reverse cells of level 1, are channeled horizontally and immediately following the first reversible level of the pipeline, level 1. The signals in `F_Rails 1`, which refers to the 6 control rails that are needed by

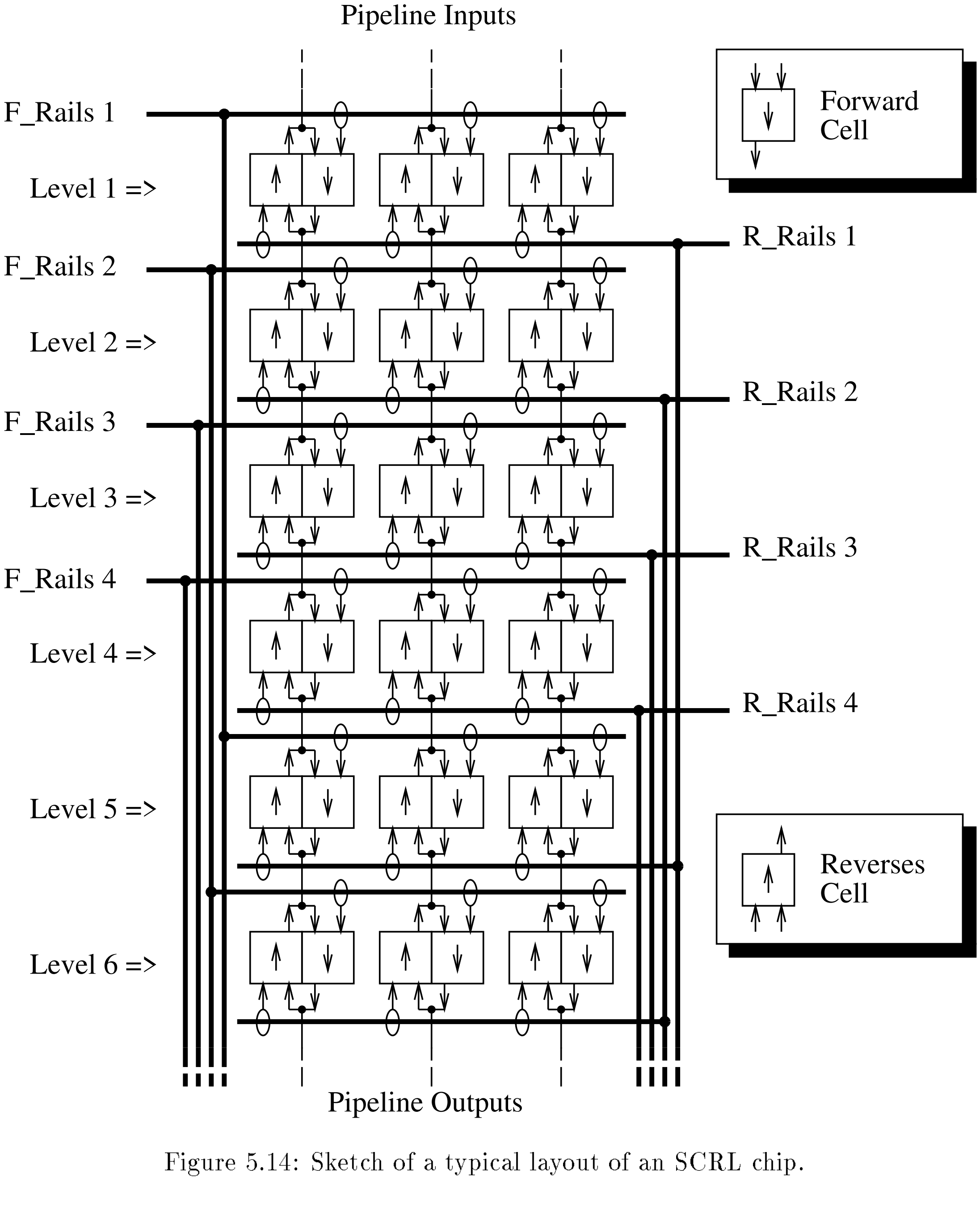


Figure 5.14: Sketch of a typical layout of an SCRL chip.

the forward cells of level 1, are channeled horizontally and immediately preceding, and in parallel with, the first reversible level of the pipeline, level 1. By making sure that all the reversible cells in the library have the same height we can always produce circuits with highly optimized and regular layouts.

The layout in Figure 5.14 is that of a 4-phase SCRL circuit. As such, the rails controlling the cells in level a are the same as the ones controlling the cells in levels $(a \pm 4)$ and hence should be joined. This is accomplished by the rail channels that run vertically in the layout. In SCRL-1, the vertical channels for the forward rails are on the left side while the channels for the reverse rail are on the right. The choice is obviously arbitrary.

Finally, the sketch in Figure 5.14 simplistically shows that the I/O lines from each reversible cell go to the cells that are directly above or directly below it. This is not the case in general. Outputs from a reversible cell could go to any reversible cell in the subsequent level of the pipeline. The cross routing of these I/O signals is generally routed along side the horizontal channels of the control rails.

5.7 Design Entry and Verification

The chip was designed using Cadence custom layout tools. The circuits were entered in schematic form and were verified with verilog simulation files. Each node in the circuit was treated as a `trireg` node in the verilog model in order to detect charge sharing. After verification, the cells' layout were synthesized using Cadence synthesis tools. The resultant layout was then optimized by hand. Upon completion, the layout was compared to the verified schematic including verification of device sizing. The verification included the circuits all the way to the bonding pads of the chip in order to detect all the wiring errors and to guard against incorrectly biased protection diode. An HSpice model was produced for some of the key cells in the design and simulated to verify functionality when parasitics are included.

The entire design resides in the `scrl` cadence library in ~`younis/cadence` directory in at the AI Lab. The chip was fabricated using 2 micron P-Well process by ORBIT through MOSIS.

6. Test and Measurement Results

6.1 Introduction

Testing of SCRL-1 was to accomplish two objectives. The first objective was to verify that SCRL-1 was indeed a working 8 reversible logic multiplier. The second was to show that circuits that are built using SCRL techniques consume considerably less energy than their conventional CMOS counterparts. Specifically, the energy consumption measurements were to show the asymptotically zero energy consumption behavior of SCRL circuits at lower operating frequency.

For the first objective, I was able to verify the complete functionality of SCRL-1 as the design intended, including exhaustive testing of all I/O patterns. The second objective was more difficult to accomplish however. Power measurement of low power circuits is a tricky process. Energy measurement of low power circuits, as it turned out, is even more so. While I was able to show that the energy consumption of SCRL circuits is well below that of conventional CMOS, I was unable to measure exactly how much SCRL-1 was actually consuming. This is because the actual magnitude of this dissipation was buried below the dynamic range of the instrument used, and any attempt to get meaningful data was swamped by roundoff errors. Not knowing the exact values of dissipation at various frequencies made it impossible to verify the predicted asymptotic behavior of SCRL circuits.

In this section I will outline the testing procedures that I used on SCRL-1, and will conclude with suggestions on how further testing could proceed.

6.2 Digital Functional Testing

In this phase of testing no attention was paid to the energy requirements and hence all signals supplied to the chip under test where either 0 or 5 Volts. To achieve this objective, I built a PC interface with a number of output and input lines. Each of the output lines could be independently set under software control. All the lines in the link connecting the PC side to the test fixture side were differential signal lines which therefore allowed the test fixture to float with respect to the power and ground lines of the PC. This was done to eliminate ground loops as well as to isolate the test fixture from the PC noise. C programs running on the PC sequenced the input lines of the SCRL-1 under test and read back the produced outputs for comparison.

6.2.1 Testing of Forward Pipeline

To test the function of the forward pipeline, the test fixture was set up as follows. First, all the slow and all the fast rails were connected to either V_{dd} or GND depending on their polarity. This is effect makes all the SCRL circuits in the chip identical to conventional CMOS. Secondly, each of the pass rails controlling the pass gates in the reverse pipeline were connected to either V_{dd} or GND so as to turn all the reverse pass gates off. Having done this, SCRL-1 is now reduced to a pipelined conventional CMOS multiplier with the forward pass gates serving as pipeline registers.

The first test was to turn all the forward pass gates on so that the whole chip is in effect one combinational level deep and any change in the inputs would show up at the outputs. Then all the possible combinations of operands were introduced and the resulting outputs sampled. The chip produced all the correct results.

The second test was to verify the function of the pass gates in the forward direction. In this tests, the state of the forward pass gates were sequenced in the same order as that under normal SCRL operation. After the predicted number of clock cycles, the correct outputs were observed. This was agian done for an exhaustive set of input combinations.

The third test was to verify the function of the latches that are used in the `pipelineStop` modules of SCRL-1. To test these latched, SCRL-1 was sequenced by controlling the pass gates in the forward direction until a piece of data arrived at the latches inputs. The data was then latched. Then all the previous stages of SCRL-1 were disabled and the output from the latches were enabled by enabling the reverse pass gates at their outputs. At this point, the content of the latches provided the only source to drive the node from which the data had been sampled. Now, the forward pipeline was allowed to proceed from that point. If the latches worked correctly, the values at the outputs should not give any new results. If however, the value from the latches differed from the sampled data, the outputs would be corrupted by the insertion of inconsistent values from the latches in the middle of the forward pipeline. The latches of `pipelineStop` modules locking the values of A, B and R all passed this test under all input combinations.

6.3 Testing of Reverse Pipeline

Even though SCRL-1 is internally a fully reversible implementation, externally both the operand input pins and the result output pins are one-directional. Therefore, testing the reverse pipeline by injecting the result at the outputs and watching for the state of the inputs is not possible. For this reason, the testing of the reverse pipeline had to be done indirectly. In this test, the reverse pipeline was at first fully disabled and the forward pipeline was allowed to proceed for a few cycles to set up some values on the nodes throughout the pipeline. Next, the forward pipeline was disabled and the reverse was allowed to proceed for a few cycles in effect back tracking the computation. Finally the direction was again reversed and the forward pipeline was allowed to proceed all the way to the outputs. In the event that the reverse pipeline was working properly, back tracking the computation for a number of cycles should not affect the final result. SCRL-1 passed this test under all input combinations.

Having passed all of the above tests, it was evident that all the components of this silicon version of SCRL-1 were, from a digital point of view, working correctly. Even though in retrospect one does radiate confidence when asked about what he felt the chances were, I have to admit that seeing reversible logic predict the right value in silicon for the first time was heartening.

6.4 Split-Level Operational Testing

To test the chip for SCRL operation, one must sequence the slow and fast rails between their SET and RESET states in addition to sequencing the Pass rails. However, the slow and fast rails have $V_{DD}/2$ as their RESET level and therefore could not be driven by simple logic devices. Furthermore, to test for the linear relationship between the energy consumption and the clock frequency, the rise and fall times for all the rail had to be controlled. In essence the testing circuit must achieve the following. It must provide the correct voltage level to all the rails that are not moving, and swing those that are moving in a controlled and timely manner. To this end a number of test circuits were designed and tested. The first was based on Direct-Digital-Synthesis approach to generate the swinging signal. It was eventually abandoned in favor of a much simpler test fixture based on RC controlled square waves. What follows is a description of these circuits.

6.4.1 Direct Digital Synthesis Testing Approach

This was the first circuit built to test the SCRL operation of SCRL-1. With Direct Digital Synthesis (DDS), a waveform is directly generated by continuously feeding a Digital-to-Analog Converter (DAC) a stream of numbers representing the voltage values for the desired waveform. The test fixture based on DDS consisted of an array of analog switches in conjunction with two very high speed digital-to-analog converter modules (DAC's) to generate the controlled voltage ramps for every control rail in SCRL-1.

During SCRL operation, each rail goes through 4 different phases of operation. In the first phase, the rail is taken from the RESET voltage level of the rail to the SET voltage level. During the second phase, the rail is held at its SET level while other rails swing. In the third phase, the rail is taken back from its SET level to its RESET level. Finally, during the forth phase, the rail is held at its RESET voltage level. During the phases where the rail is to be held at a certain voltage level, that rail cannot be left floating. This is because capacitive coupling from neighboring rails can shift the voltage on the rail and hence lead to dissipation later on.

Conforming to the above, our first test fixture contained a separate 4 way analog switch for each rail. Two of the switch positions were connected to the RESET and SET voltages of that specific rail. The other two positions of the switches corresponded to the SETting and RESETting phases of operation and were connected to a ramp generator. To SET a rail, the position of its analog switch would be set to connect the rail to the ramp generator. The generator is then triggered to produce a gradual and controlled voltage ramp that is fed to the rail through the switch. Once the ramp reaches it final voltage value, the analog switch of the rail is then thrown to the position corresponding to the SET voltage level of that rail. This helps keep the rail voltage from wandering. A procedure symmetric to the one used for SETting is employed during RESETting.

The analog switch array consisted of 24 74HC4052 analog switches. They were all biased with $V_{EE} = -5$ Volts to give a relatively constant feed through resistance throughout the entire range of the rails swings of 0 to +5 Volts. The sequencing of these switches was done by a finite state machine that can be programmed from a PC.

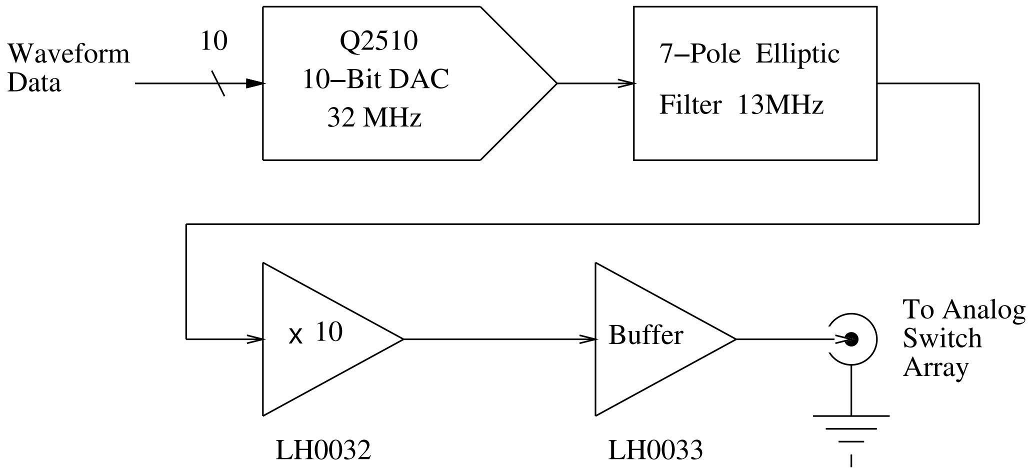


Figure 6.1: Block diagram the high speed DAC module.

SCRL operation depends on the gradual and controlled rise and fall times of the rails. For testing purposes, we must have the ability to vary these rise and fall times. In addition, our ramp generator must be able to operate at frequencies that are high enough so that the dissipation of the circuit does not become limited by leakage currents. To achieve the above a pair of high speed DAC modules were designed and constructed. Figure 6.1 shows the block diagram of one of these modules. The first component of the module is a 50MHz DAC made by Qualcom Inc. (Q2510). It was used because of its impressive speed and low noise coupling from the transitions of the digital inputs. In our design, we clocked this DAC at 32MHz. The Q2510 is a current mode DAC and the output was fed into a 50Ω resistor to convert the current signal to a voltage signal. This 50Ω resistor also became the source impedance of the DAC. In addition to the desired signal, the DAC has a strong frequency component corresponding to its clocking frequency, the fundamental. This was suppressed by including a 7-pole elliptic filter at the output of the DAC. To eliminate reflection and maximize power transfer, the input and output impedances of this filter were designed to be 50Ω. From this we see that the effective impedance as seen by the output pin of the DAC is now 25Ω. Since the Q2510 has a full scale current of 20mA, we see that the full signal swing out of the filter is about 0.5V. For this reason, an operational amplifier was added and configured as a $\times 10$ inverting amplifier. The amplifier selected for the task was a National Semiconductor LH0032 operational amplifier with a slew rate of 500V/μS. The amplifier was followed by a 250mA current buffer so that the source impedance of the whole DAC module is low enough to drive the load of any rail.

The 10-bit data words were read in from a cyclical SRAM data buffer that was previously loaded from the PC. The data buffers had space for four different waveform patters corresponding to the SETting and RESETting of both the slow and fast rails as well as the SETting and RESETting of the Pass rails. At this point, the PC is able to program the sequence of states of the analog switches as well as to download the contents of the cyclical SRAM buffer. This in effect provided us with a very flexible and programmable fixture that is able to control the level and the rate of change of the voltage on each of the control rails of SCRL-1. The finite state machine controlling the states of the switches had extra bits to select which of the four waveform buffers to use. It also had enough bits to configure the data inputs to the chip under test.

After some debugging, I managed to get this test circuit described above to work as designed. There was a number of problems with this approach however. The first and foremost was noise. To generate voltage ramps with the desired rise times, the DAC had to run at ~30MHz. This meant that in addition to the DAC's, all the components of the cyclical SRAM buffers with their counters and registers had to be clocked at the same rate. The toggling of all of these digital nodes at such a frequency injected some noise into the ground plane of the analog part of the circuit. For general use, that noise would not have been a problem. For SCRL however, the noise could lead to false readings. Recall that in SCRL, the energy consumption is linearly related to the slope of the signals at the control rails. Slope here refers to the instantaneous slope and not the average slope. The fact that the average slope is small is not enough to reduce the energy dissipation of the circuit if the signal had a high frequency component riding on the average transition. In this case, the slope affecting the dissipation is really the slope of the high frequency component even though it could be much smaller in magnitude than the slow moving average component. Multiple attempts to suppress this noise did not significantly improve the purity of the slow moving signal.

The second problem involves the harmonics that are generated by the DAC. Even though the elliptic filter correctly removed the fundamental sampling frequency as well as all the harmonics above it, it could not remove the subharmonics that Direct-Digital-Synthesis (DDS) produces. In so far as the generated frequency out of the DAC is relatively close to the cutoff frequency of the DAC, we could expect very nice smooth waveforms from our module. At lower frequencies however, the folding of the spectrum results in a staircase shape of the waveform as the time between samples of different values becomes large. As we have shown in Section 2, the dissipation of an SCRL circuit that is driven by a staircase waveform is inversely related to the number of steps in the waveform irrespective of the average rise time of the full swing. From this we see that once the generated waveform is slow enough for the staircase effect to appear, slowing down the rise time, does not affect dissipation and therefore could yield incorrect test results. One way to fix this is to use a switched capacitor filter with a variable cutoff frequency in place of the fixed frequency elliptic filter. By always changing the cutoff frequency of the filter so that it was always just slightly above the average frequency of the rail, we could greatly reduce the staircase effect in the generated ramp. The problem is that it was difficult to find programmable filters with a pass band at the high frequencies of operation that we desire.

Finally, the above test circuit clocked at 30MHz could not generate signal at frequencies higher than ~1MHz with any accuracy. This is because we need a minimum number of points from a DAC for each swing if we are to produce that swing with sufficient control on the rise and fall times of the generated ramp.

Because of the above reasons, this first test fixture with its approach that relied on DDS were eventually abandoned.

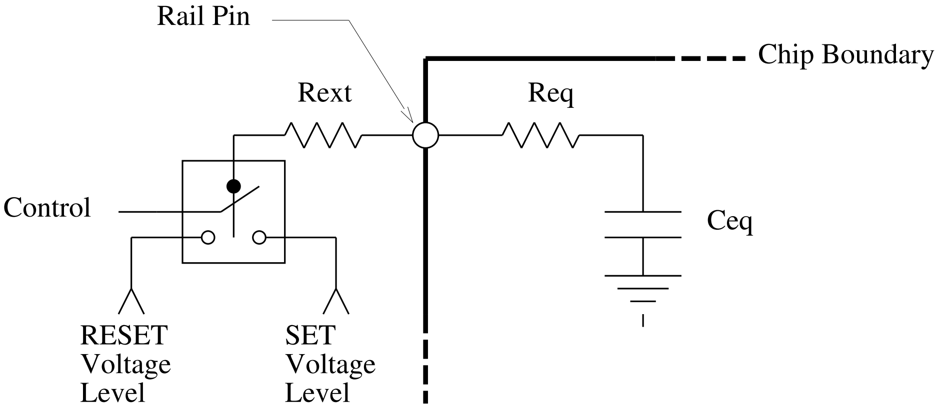


Figure 6.2: Block diagram of the components of the RC-controlled voltage step testing approach.

6.4.2 RC-Controlled Voltage Step Testing

In this method, the waveforms were generated by passing a voltage step through a resistor that would, with the aid of the equivalent capacitance of the rail, slow down the rise/fall time to a predetermined rate. Figure 6.2 illustrates the components of this approach. As seen from the chip boundary, each rail could be modeled by an equivalent capacitance in series with an equivalent resistance. The effective time constant of the rail,τ_{eq}, is then $R_{eq}C_{eq}$. Adding an external resistance in series with the rail having a much larger value than the internal equivalent resistance of the rail, forces the time constant of the new circuit , τ_{ext}, to $\simeq R_{ext}C_{eq}$. Since now $\tau_{ext} \gg \tau_{eq}$, the rise time of the rail becomes governed by R_{ext} and could be adjusted to any value less than τ_{eq}. As illustrated in Figure 6.2, an analog switch is added to toggle the rail voltage between the SET and RESET levels. The analog switch is controlled by digital signals. In our test fixture, the above circuit is duplicated for every independently swinging rail. In total there are 48 separate rails and hence 48 separate switches. For every rail that swings in the positive direction and controls the P-Channel devices of the circuit, there is a rail that simultaneously swings in the negative direction and controls the N-Channel devices. Since both swing at the same time, the analog switches associated with these rails share a control line. This reduces the number of control bits to 24. To produce the correct sequencing patterns on these 24 lines I used a simple microcontroller with 3 external 8-bit decoded registers. The microcontroller was a PIC 16C54 from Microchip Technologies in an 18-pin cerdip package with UV erase window. The registers were CMOS 74HC374's to produce full rail-to-rail output drive.

To guard against ground loops, all the voltage supplies for the test fixture were floating supplies with no connections to earth. In addition, the 5V needed to run the microcontroller and registers were made separate from the 5V supply that was used in SETting some of the rails. The circuit was constructed over a single, copper plate ground. Other needed voltages were +2.5V for the RESET level of the slow and fast Rails and -5.0V to sufficiently bias the analog switches. By reprogramming the microcontroller, alternate rail sequencing could be tested.

The clock signal for the microcontroller came from a programmable waveform generator. To eliminate a potential ground loop, the clock from the generator was coupled to the test fixture trough a differential line receiver. In addition to getting rid of the potential ground loop, the differential line receiver was sensitive enough to detect a 500mV peak-to-peak signal, and provided enough regeneration to accept a sinusoidal signal from the generator. With a low power pure sinusoid fed through differential lines to the test fixture, the noise from this high frequency signal was sharply reduced.

In general, the microcontroller programs started by activating only the forward pipeline for a number of cycle that were enough to consistently setup the pipeline. Following that the reverse pipeline was activated thus starting true SCRL operation. Even though the functionality of the reverse pipeline was already tested logically, this test fixture provided the first opportunity to test the entire chip under true SCRL operating conditions including the splitting and restoration of the slow and fast Rails. The chip again passed this test.

6.5 Energy Measurement Procedures

Figure 6.3 shows the location of the scope probes that were used in measuring the energy consumption. The probes were connected to an HP5411D digitizing oscilloscope. The oscilloscope was connected to a PC using the GPIB interface and software drivers. A program on the PC side calculated the optimal oscilloscope settings, programmed the scope, and retrieved the data samples from the scope for analysis automatically. In our setup, the time during which a rail was moving, was much smaller than the period of operation of that rail. With this in mind, having enough time resolution on the scope screen to accurately observe the rise time of a rail, meant that a single period would span multiple scope windows. For this reason, the program running on the PC also calculated the number of windows that are spanned by a single period and would then control the scope to give multiple delayed snap shots so as to cover the entire period with sufficient time resolution. To reduce sampling noise, the scope was set up so that each reported data point was the result of averaging 10 different samples takes as different times but having the same relationship to the triggering event. In averaging mode, the HP5411D gave 501 points for every window snap shot per channel. Therefore a measurement spanning multiple windows to cover the whole period could yield more than a few thousands data points.

To compensate for probe offset, the PC would take one measurement trace while the rail voltage was held at 2.5V. It would then take the average of all of the 501 points for each channel and subtract 2.5V from it to get the probe offsets of each channel. Later on, all the readings from the scope were adjusted by these offsets before they were used in the calculations.

Knowing the value of R_{ext} for the rail under measurement, we can calculate the power into the rail pin from the following equation.

$$P_{rail} = \frac{(V_2 - V_1) \times V_1}{R_{ext}} \tag{6.1}$$

This yields the instantaneous power consumed at a point in time. Averaging this value

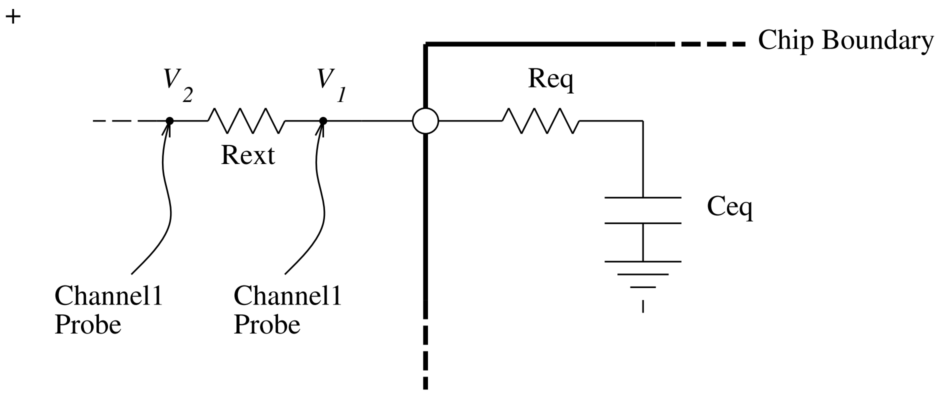


Figure 6.3: Scope probe locations during energy dissipation measurement.

with the value of the previous data point and multiplying the result by the difference in time between the measurements yields the energy packet that was consumed by the rail during the interval between the consecutive data points. The above procedure is nothing more than the time integration of power over the interval using the trapezoidal rule to get the energy. Adding all these packets of energy from each time interval over exactly one period yield the total energy consumed by the rail during a period of operation. Here the RC-controlled voltage step approach provides more accuracy than the previous approach since the potential drop across R_{ext} is always large, almost full V_{dd} at some times, since R_{ext} is by design $\gg R_{eq}$. This is in contrast to the DAC approach where the rise time is controlled by the DAC and hence any resistor that is inserted to measure the current would yield a small voltage difference, probably in the 10mV range.

To measure the energy consumption at a given rise time for the entire chip, the above procedure is repeated for every rail and their energies all added up. The reason for the direct measurement of energy is that we wanted to examine the energy that is dissipated inside the SCRL-1 only. Inserting an external inductor would have made our measurement subject to the Q of the inductor and we then would have had trouble separating the dissipation components.

At one point it was though that an operational amplifier connected so as to directly compute $(V_2 - V_1)$ would greatly increase the dynamic range of the measurement through the elimination of the common mode voltage. Unfortunately, after constructing the circuit, the common mode rejection ratio of this and other high speed amplifiers deteriorated at high frequency to almost 0 dB. At near 0 db rejection ratio, the difference between common mode and differential voltage is non existent making this approach useless. Therefore this approach was abandoned.

6.6 Test Results

As I have mentioned earlier, I was not able to directly verify the linear relationship between rise/fall time and energy dissipation that the theory predicts. Indirectly however, all other measurements seemed to confirm SCRL operation as predicted by both theory and SPICE simulations. This strengthens my conviction that the undetectability of the linear behavior was more due to insufficient accuracy in the measurement techniques rather than to an unforeseen flaw in the theory.

6.6.1 The Problem of Direct Measurement of SCRL Energy

As stated earlier, our test fixture measured the dissipated SCRL energy directly. This meant simultaneously and repeatedly sampling the voltage at the rail and the current into the rail for one full period of operation and then numerically integrating the calculated instantaneous power over the period to get the dissipated energy. It then became obvious that direct energy measurement of SCRL circuits is nearly impossible. To show this, let us track the energy flow for a fast rail that swings between 2.5V and 5V. When the rail is driven from 2.5V to 5V, the driving circuit delivers a certain amount of energy to the rail, we call E_{SET}, where

$$E_{SET} = E_{C_{eq}} + E_{R_{eq}} \tag{6.2}$$

In the above equation, the first energy component is the amount of energy that is needed to charge the equivalent rail capacitance, C_{eq}, from 2.5V to 5V. The second is the energy that is lost in R_{eq} as the result of charging C_{eq} through R_{eq}. When the rail is driven back to 2.5V, the energy delivered to the rail, E_{RESET}, is equal to

$$E_{RESET} = -E_{C_{eq}} + E_{R_{eq}} \tag{6.3}$$

Hence the total energy consumed is

$$E_{SET} + E_{RESET} = 2E_{R_{eq}} \tag{6.4}$$

The problem with measuring this directly is that $E_{C_{eq}}$ is very nearly equal to the energy consumed by conventional CMOS circuit in each cycles. In contrast, $E_{R_{eq}}$ is predictably orders of magnitude smaller than $E_{C_{eq}}$ for all operating frequencies except very close to the maximum operating frequency of the SCRL-1 chip. Since every branch in SCRL-1 has been properly sized to reduce energy consumption, and since each rail drives a circuit that is only one logic level deep, the internal speed of each stage of SCRL-1 is somewhere below a nanosecond. This means that for all operating frequencies below 100MHz, $E_{R_{eq}}$ will literally be orders of magnitude below $E_{C_{eq}}$. The HP5411D only has 6-bits of resolution at our operating range. In addition, each rail voltage was slightly above half of the full scale deflection of the scope and therefore the resolution was effectively only 5-bits. With this resolution, anything below one part in 32 is buried under the quantization noise. Anything that is orders of magnitude below the maximum is obviously, undetectable. For the above reason, we see that the way in which the energy flows in and out of a rail in SCRL, renders direct measurement impossible. Please note that in order to get energy measurement spanning multiple decades of operating frequencies, we would need an instrument with a dynamic range of multiple decades. Therefore, even though a more accurate instrument would get us closer to the desired quantities, no instrument could give us multiple energy measurement points spanning multiple decade of operating frequency is which what is

Rise Time	Trial	$E_{C_{eq}}$	$2E_{R_{eq}}$
500 nS	1	0.13 nJ	-0.06 nJ
500 nS	2	0.13 nJ	-0.09 nJ
500 nS	3	0.13 nJ	-0.05 nJ
500 nS	4	0.13 nJ	-0.07 nJ
750 nS	1	0.13 nJ	-0.06 nJ

Table 6.1: Energy measurements for rail FF1(+) in SCRL-1.

needed to verify the asymptotic behavior of SCRL circuits. For this reason, SCRL energy measurement must be done indirectly.

Table 6.1 lists energy measurements for positive SETting Forward direction fast rail #1, FF1(+). To measure $E_{C_{eq}}$, the power integration was started immediately before the SETting of the rail and then detecting the maximum energy value reached during the integration. This corresponds to the energy sent into the rail before the direction of power flow is reversed. $2E_{R_{eq}}$ was taken to be the value of the integration over the whole period. From the table we see that $E_{C_{eq}}$ is measurable and consistent but $2E_{R_{eq}}$ was not. This reflects the fact that it was buried under the dynamic range of the measurement. The measurements were also mostly negative resulting from a negative bias from the offset voltage of the probes. In addition, the table shows that $E_{C_{eq}}$ is not a function of rise time as we would expect. This is because $E_{C_{eq}}$ is equal to CMOS dissipation which is not affected by rise time.

6.6.2 Confirmation of SCRL Energy Flow

Measurements on rails FF3(-) and RF4(-) are listed in Table 6.2. FF3(-) is the #3 negative SETting fast rail in the Forward direction. RF4(-) is the #4 negative SETting fast rail in the Reverse direction. FF3(-) is the most loaded rail in SCRL-1. The interesting thing in the table is that E_{period} is measurable with some consistency. It is also always negative for FF3(-) indicating that this rail is supplying energy to the outside world. The magnitude of E_{period} is also large enough that these measurement are not due to noise.

To explain these results, we recall the sequence of operations for an SCRL gate. In the forward direction, a gate grabs the output node while it is at 2.5V and sets it to the correct logic level. It then disconnects itself from the node so that it can restore itself without affecting the value on that node. The gate driving that node in the reverse direction is responsible for restoring it to 2.5V. Therefore, fast rails in the forward directions always SET circuit nodes while fast rails in the reverse direction always RESET them. For negative swinging rails, SETting a node means removing energy from it so as to discharge it from 2.5V to 0V. RESETting the node means delivering energy back to it to charge it back up to 2.5V. The opposite is true for positive swinging rails. In SCRL-1, FF3(-) is responsible for SETting the nodes following the third pipeline stages, while RF4(+) is responsible for RESETting them. With this in mind, energy extracted from SCRL-1 through FF3(-) must come from RF4(-), and Table 6.2 shows that it does. From the above, we see that $E_{R_{eq}}$

Rail	Rise Time	Trial	$E_{C_{eq}}$	E_{period}
FF3(-)	500 nS	1	0.31 nJ	-0.64 nJ
FF3(-)	500 nS	2	0.31 nJ	-0.69 nJ
FF3(-)	500 nS	3	0.31 nJ	-0.69 nJ
FF3(-)	750 nS	1	0.30 nJ	-0.79 nJ
FF3(-)	750 nS	2	0.30 nJ	-0.77 nJ
FF3(-)	750 nS	3	0.31 nJ	-0.70 nJ
RF4(-)	750 nS	1	0.30 nJ	+0.72 nJ
RF4(-)	750 nS	2	0.31 nJ	+0.71 nJ
RF4(-)	750 nS	3	0.30 nJ	+0.70 nJ

Table 6.2: Energy measurements for rail FF3(-) and RF4(-) in SCRL-1.

really corresponds to the difference in E_{period} between the pair of rails that affect the same circuit nodes. Subtracting E_{period} of FF3(-) from that of RF4(-) gives us unreliable data since once again the real difference is much less than two quantities. The above energy flow provides confirmation of SCRL operation in the chip.

6.6.3 Confirmation of Node Hand-off in SCRL

Measurements of energy for FF3(-) and RF4(-) confirmed the hand-off that occurs between the forward and reverse parts of the pipeline. However, the effect was not detectable for other rails. This is because the load on other rails is smaller than the load on FF3(-) and RF4(-) in SCRL-1 which resulted in burying E_{period} below the dynamic range of the measurement.

To confirm the effect of the hand-off between the forward and reverse gates, we examine the difference between the rise and fall times of these gates. During the rise time, FF1(+) charges both the internal nodes as well as the output node of all the gates it controls. During the fall time, FF1(+) discharges only the internal nodes of these gates. This means that the effective capacitance of FF1(+) during rise time is larger than its capacitance during fall time. Figure 6.4-a shows the rise time for rail FF1(+) in response to a square wave input with a 2777.5Ω external resistor. The rise time is approximately 500 ns. Figure 6.4-b shows the fall time for FF1(+) in response to a square wave input with the same 2777.5Ω external resistor. As we can see, the fall time is a much faster $\sim$ 300nS confirming the prediction based on SCRL operation.

6.6.4 SCRL verses CMOS Operation

To illustrate the contribution of the reverse pipeline in reducing the energy consumption of CMOS circuits, I ran SCRL-1 with the reverse pipeline disabled. Under such conditions, all the reverse pass gates were disabled and all the reverse slow and fast rails were tied to 2.5V. Figure 6.5 illustrates the waveform trace of FF1(+) rail with correct SCRL conditions as well as with the reverse pipeline disabled, *i.e.,* conventional CMOS conditions. Figure 6.5-a shows FF1(+) under normal SCRL operation. Note that in this mode the

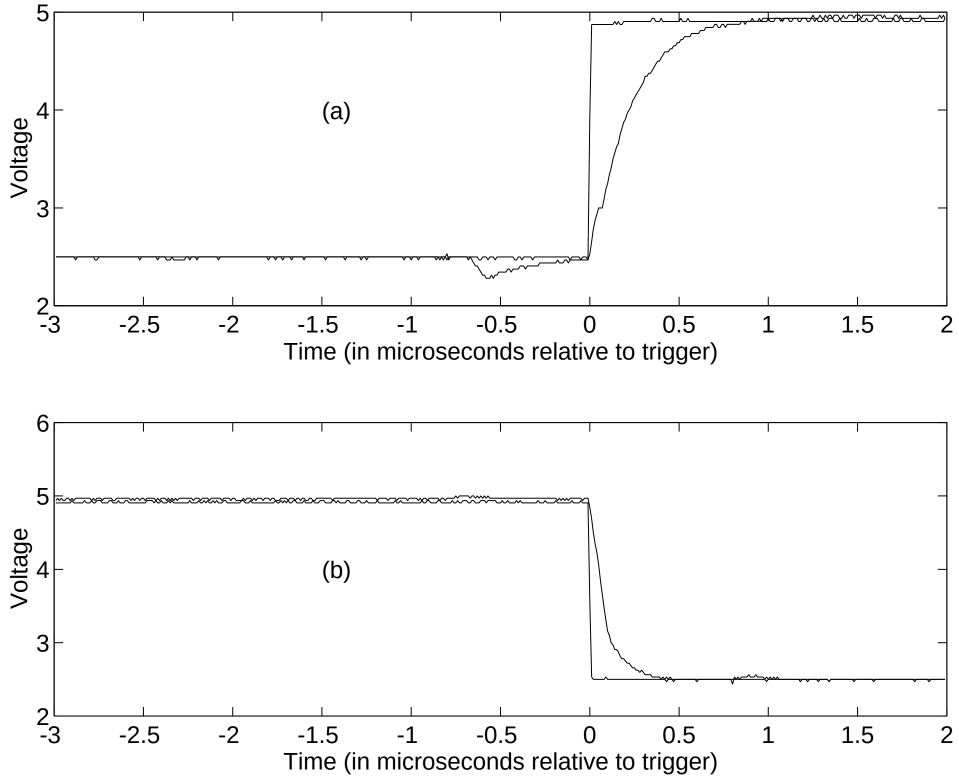


Figure 6.4: (a) Scope trace of the rise time of FF1(+) with 2777.5Ω resistor. (b) Scope trace of the rise time of FF1(+) with 2777.5Ω resistor.

transfer of charge to and from the rail are completely controlled by the rise/fall times of the rail. No uncontrolled charge sharing or voltage resetting occurs and the waveform is fairly smooth.

Figure 6.5-b shows FF1(+) with the reverse pipeline disabled thus mimicking conventional CMOS operation. For both traces, the forward gate controlled by FF1(+) is connected to the output approximately 2.5μS before the rail start rising. Under SCRL conditions, the reverse pipeline would have non-dissipatively RESET the value of that node to 2.5V. Hence when the hand-off occurs under SCRL conditions, no charge moves and no glitch occurs. In the absence of the non-dissipative restoring action of the reverse pipeline, there is no way for the circuit to non-dissipatively restore the voltage on that node to 2.5V. Hence we observe a large spike around the hand off time of $\sim$ 2.5μS before the FF1(+) start rising. There is also no way for FF1(+) to recover the energy in the spike, since it cannot tell the voltage values on all the nodes that are causing it. Integrating the instantaneous power in this spike for the duration of this spike we get 0.2 nJ for FF1(+). This is on the same order as the total energy delivered to C_{eq} in FF1(+). It is also the amount that CMOS dissipated every cycle. From the above we see how the introduction of the reverse pipeline have helped eliminate the spike associated with increased information entropy.

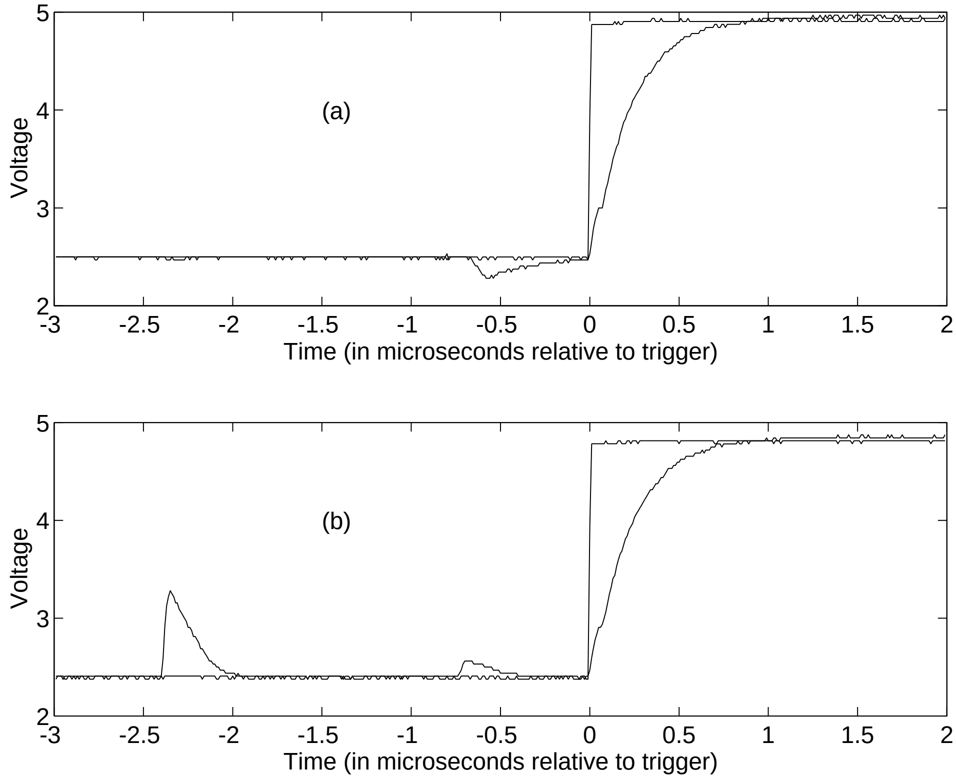


Figure 6.5: (a) FF1(+) waveform under SCRL operation. (b) FF1(+) waveform with the reverse pipeline disabled.

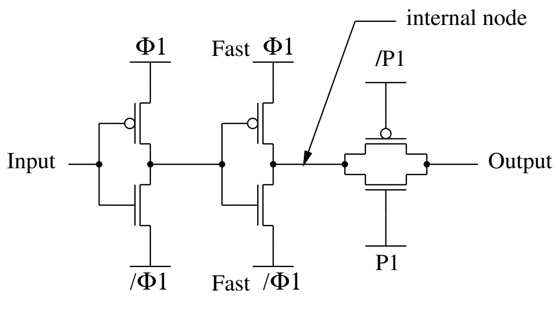


Figure 6.6: Non-Inverting SCRL Gate.

6.6.5 SCRL Capacitive Coupling

Up to this point we have ignored a small voltage "bump" that occurs just before the rail started rising in all of the traces shown so far. The bump was also evident under SCRL conditions. The presence of the bump was not a surprise as I have observed it in HSpice simulations. In addition, its negligible effect on dissipation should become evident as soon as we recognize its cause. Figure 6.6 shows the diagram of a typical 2-level SCRL gate.

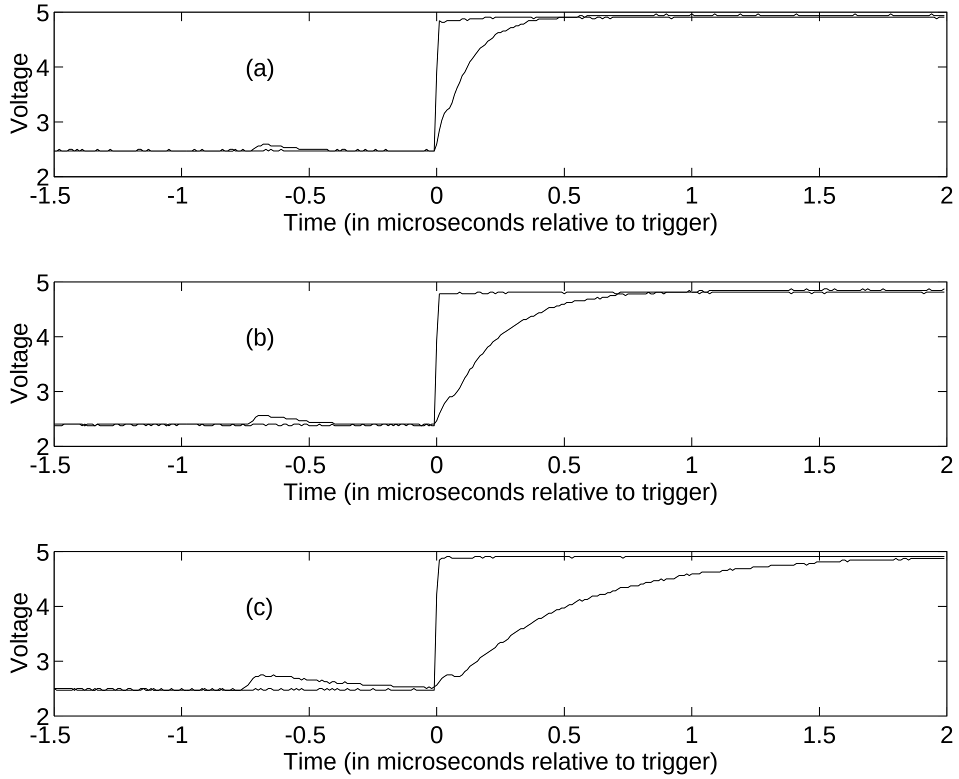


Figure 6.7: CMOS capacitive from swinging of FS1(+/-) at varying R_{ext} on FF1(+).

We will assume that the output pass gate is always on. Starting with valid inputs and all internal node and rails at 2.5V, we proceed to SET the first stage by splitting the fast rails. If the input was at 0V, the output of the first stage would rise to 5V. While rising, this signal would capacitively couple through the gates of the second stage to the output of the gate thus moving the output voltage away from 2.5V. This is possible because at the time that the fast rail starts to SET, the devices in the second stage are all off and hence the output node of the gate is floating. As soon as the output from the first stage climbs a few tenths of a volt above 2.5V, the second stage devices start to conduct and the output voltage is anchored to 2.5V. This obviously leads to dissipation. Fortunately, the problem could be cured in a variety of ways. We can for example introduce a keeper transistor that will connect the output of the gate to 2.5V while the first stage swings.

Another way to reduce its effect is to note that the slower the rise time of the fast rails are, relative to R_{ext} of the slow rails, the smaller the size of the bump becomes. Figure 6.7 shows multiple scope traces of FF1(+) during which the rise time of FS1(+) is kept constant, $\sim$ 500nS, while R_{ext} of FF1(+) is increased. We point the readers attention to how the magnitude of the bump, located around -0.5μS, increases with increased R_{ext}. Also note the kink in the rise of FF1(+). This corresponds to the time that the rise of FF1(+) starts to turn on the devices whose gates FF1(+) is driving. At the point the rail capacitance suddenly increases and the rail rise time momentarily dips due to inductance in the rail circuit. Soon after, the rail resumes its rise. Irrespective of the magnitude of the capacitive coupling, the magnitude of the voltage bump never exceeds the value at which the kink occurs since the sudden added capacitance of the node helps to significantly reduce the capacitive coupling to the fast rail.

6.7 Summary

As stated in this section, we have been able to verify the complete operation of SCRL-1 chip. Furthermore, through a number of energy measurements we have been able to verify the energy movement from the forward to the reverse rails as well as the predicted difference in capacitive load for each rail under SCRL operation. We have also demonstrated the importance of the reverse pipeline by observing the increased dissipation in the form of an RC glitch when the reverse pipeline was disabled. Unfortunately, the experimental verification of the asymptotic behavior of SCRL continued to elude us. However, measurements of other quantities all agreed with that which is predicted under SCRL operation.

One method that might succeed in measuring the energy dissipation of SCRL circuits would be to use an external inductor. By connecting an inductor to the a rail, we effectively have an RLC circuit. Through calibration, we could determine the amount of energy that is lost in the inductor at our operating frequency. Next we start the circuit oscillating and maintain it in that state by constantly replenishing the energy that is dissipated every cycle. By measuring this injected energy, and allowing for inductor dissipation, we should be able to determine the energy dissipation of the rail. We intend to do this in the near future.

Part IV

Future Research Areas

7. Future Work

To my knowledge, SCRL-1, is the first working implementation of a pipelined, reversible computation based, asymptotically zero energy logic. As such, I am certain that there is a lot more for us to discover and refine than what we have reported so far. This section attempts to give some suggestions about where future research in this area could proceed. My opinion is that as far as energy-conscious reversible computation is concerned, we've only just begun...

7.1 SCRL-1 Energy Measurement

Even though we are quite pleased with the results we've gotten so far, one important aspect of SCRL operation has not yet been verified: the linear reduction in energy consumption with the reduction in operating frequency. In the preceding section, I have outlined a number of attempts that were targeted towards verifying this SCRL property. I have also demonstrated that such a measurement could not be performed directly. In the future, we plan to re-attempt this measurement. Two options are now under investigation. The first is to use calibrated LC oscillating tank circuits, while the second is based on calorimetric techniques. The advantages and disadvantages that are offered by both are currently under consideration.

7.2 CAD Work

The main reason behind the likely resistance to the adoption of SCRL, is the requirement of reversible computation. Requiring designers to design the components for both the forward as well as for the reverse pipelines of the circuit is cumbersome and wasteful. This is because in defining the design of the forward modules, one also uniquely defines the function of the reverse and inverse modules. Admittedly, the real problem for the synthesis of the inverse modules is that of existence rather than definition. For an inverse module to exist, the function of the forward module it corresponds to must be bijective. Unfortunately, most simple logic functions are not bijective. However, our experience is that non-bijective functional modules could be embedded in a larger functional blocks that on the whole are bijective. For example, a NAND gate is not bijective, but a crossbar switch made out of a number of NAND gates is. To this end a CAD system could be a big help. An SCRL CAD system would take the specifications to only the forward modules as an input for synthesis. It would then attempt to section the total function of the forward pipeline into a number of modules placing the sectioning lines at points were the resulting modules are bijective, or at least are easily augmented to be so. Once that is accomplished, the rest is relatively straight forward and currently available CAD tools could be called upon to complete the designing process. In summary, the most important contribution of CAD in the area of

SCRL is in automating reversibility or at least advising the engineer upon the optimal place at which reversibility should be broken at times where it had to be. I believe work in this would prove to be both an interesting challenge as well as a rewarding opportunity.

7.3 Architecture Issues

As we are familiar, there is usually more than one way to perform a computing function, each being more suitable than the others under one or more constraints. Until recently, reversibility was not one of these constraints. In SCRL, reversibility is now the most important constraint. Architecture targeted for SCRL circuits should have reversibility in mind at all the levels of the design. This should continue to be true even under the scenario of readily available and efficient SCRL CAD tools. The obvious reason is the fact that implementations that are impossible to make reversible at the circuit level can be easily made so on the architectural level. Well before the existence of any real implementations of reversible logic, in 1981 Ressler [36] addressed some of the architectural as well as some of the CAD issues in building a conservative logic computer.

7.4 Power Supply Switch

In Section 2, we have seen how the dissipative effects of the power MOSFET in the power supply could significantly reduce the energy saving performance of charge recovery circuits. Even though there might be applications where lower dissipation in the computing circuits alone is just as useful, in general, one has to include the dissipation in the power supply when comparing SCRL to conventional CMOS. It should be accepted by now that if the power switch was built out of the same CMOS technology as the computing circuit, the best we can optimize for is an energy consumption that is related to the $\sqrt{T}$. However, this limit does not apply in circuits where the computing part and the power part are built out of different technologies. This presents another research opportunity, where the marriage of SCRL built with CMOS devices and a power switch of alternate technology leads to substantial reduction in the energy consumption of the system.

7.5 SCRL Circuit Improvements

One of the potential problems of multi-level SCRL, is the voltage bump that was described in Section 6.6.5. A number of remedies were also outlined in that Section. We think it is important to pursue the elimination of this effect in SCRL circuits since it could reduce the efficiency of the circuit.

Finally, a major disadvantage of SCRL compared to the earlier implementations of CRL is the larger number of required rails. In going from CRL to SCRL, we managed to reduce the number of devices, cut down the number of wires and simplify the circuits considerably. Unfortunately, the number of required rails rose just as considerably. Our drive to improve CRL came from a feeling that *"simple circuits should not be that complex."* In following this hunch I was fortunate enough to stumble upon SCRL.

Faced now with the large number of required rails in SCRL, I'm thinking of trying this trick again, namely:
"simple circuits should not be this complex."
I have to admit though that I am bracing myself for a possible answer of...
"Well... maybe they just are!"

A. CRL at Low Temperature

It is generally known that operating CMOS systems at low temperatures improves a number of desirable device parameters. This appendix will attempt to show the higher compatibility between CRL and low temperature operation.

A.1 CRL and Large Systems at LNT

Currently, a number of supercomputers operate at temperatures that are lower than room temperature. This trend we believe will continue because of the numerous additional device and system parameter improvements that happen at lower operating temperatures and which can no longer be ignored. In what follows we will enumerate these improvements that we believe will drive some designers to consider circuit operation at temperatures as low as 77K, the boiling point of liquid nitrogen at 1 atmosphere. We will then show how the combination of CRL techniques and low temperature operation produce unparalleled performance by each uniquely offsetting the disadvantages of the other. Our discussion will focus on Si MOSFET's visiting GaAs based devices only briefly. More precisely we will concentrate on enhancement mode MOSFET's which are the basis of CMOS technology.

A large body of work exists that describe the relevant properties of semiconductor materials and devices at low temperature and a good selection of it has been collected in [24]. For silicon, the body of research suggests that most of the parameter improvements are attained at 77K. Operating at temperatures below 77K yields diminishing returns. For this reason, we will consider 77K as our low temperature operating point and compare the performance parameters there to those at room temperature.

Material Parameters

The first improvement in material parameters is the drop in electrical resistance with reduced operating temperature. Kirchman [25] reports the resistance of materials that are used in the interconnection traces of CMOS circuits dorps considerably at 77K. He reports that the conductivity increase is largest for Al interconnecting traces. In VLSI, lower interconnection resistance leads directly to faster operation. This is because in a dense planar circuit, interconnection RC is a major factor in determining operating speeds.

In addition, operating at 77K holds the promise for off chip superconducting interconnections through using high temperature superconducting (HTS) compounds [29] for the traces in a multichip module. For CRL circuits, the energy saving ratio is limited by the Q of the inductor. But at 77K, we can use HTS material to build our inductors with Q's as high as 300,000. Initial investigation have shown that inductors with parameters suitable to CRL circuits, inductance in the microhenries and a critical current of at least 5-10 Amps, are in existence today.

The second advantage is the increase in thermal conductivity of many of the materials that are used in CMOS fabrication. Kirschman [25] reports that for "single crystal semiconductor, Si Ge GaAs; single crystal dielectric trials, sapphire, quarts, diamond; some polycrystaline materials, alumina and beryllia; and relatively pure metals, copper and aluminum," the thermal conductivity increases by as much as an order of magnitude by operating at lower temperatures. Thermal conductivity of p-type silicon is reported to be six times that at room temperature [6]. This is important since the power density of VLSI chips continues to increase as circuit density increases. On the other hand some materials currently employed in VLSI fabrication exhibit monotonically decreasing thermal conductivity with lower temperature such as polymers, glasses and metal alloys [19] [39]. However, the fact that most of these material are used in packaging suggests that careful material selection for packaging that is specific to low temperature operation could reduce the effect of these materials.

The third is increased reliability. It is well known that failure rate is directly related to operating temperature. Little data exists that quantitatively compare the reliability of devices at 77K to those at room temperature. However, we can use Arhenuis relation to predict the factor of reliability improvement due to lower temperature. The Arhenuis relation, formulated to predict the rate of chemical reactions, is widely employed in accelerated tests for device lot characterization [2]. It states that the rate at which a failure occurs, R, is given by

$$R = R_0 e^{-E_a/kT}$$

where R_0 is a constant, E_a the activation energy of the predominant failure mechanism, k the Boltzmann constant, and T is the absolute temperature. Obviously, the activation energy is an empirical parameter that depends highly on the fabrication process as well as the operating conditions and is known with relative certainty only for mature processes and technologies. Comparing the lifetime of devices at 77K to that at room temperature we calculate the improvement factor,

$$F = \frac{R_{298}}{R_{77}} = e^{(-E_a/k)(1/298-1/77)}$$

From MIL-HDBK-217E, $E_a = 0.7$ for MOS devices and we get a 8.86×10^{33} fold increase in the expected device lifetimes which is obviously overly optimistic. The error in our estimates stems from the assumption that the failure mechanism that is dominant at room temperature will continue to be the dominant one at 77K. The fact is that the relatively high rate of the mechanism that is dominant at room temperature completely masks other mechanisms that have lower activation energies and necessarily much lower R_0 that is associated with their occurrence. As the temperature drops, the rate of occurrence of high E_a processes drops off considerably faster, orders of magnitude faster, than the rate of processes with lower activation energy. It is not difficult to see how lower E_a processes will become the dominant failure mechanism at some low temperature.

A more empirical argument is that given in [23]. The argument is based on the lifetimes associated with an array of devices that operate at different temperatures. They used semiconductor devices (operating at 300K), tubes and miniature incandescent lamps (operating at 1000K to 1800K), and incandescent lamps (operating at 2800K t0 3300K) and found that the best fit line had a slope that drops as T^6. Extrapolating to 77K gives a 3360 fold lifetime improvement compared to that at room temperature. Note that this improvement will only be reached after some effort is directed towards identifying and rectifying the failure mechanism that are dominant at 77K and not just by cooling devices optimized for room temperature operation. The two reliability drawbacks of operating at low temperature, or temperature much lower than manufacturing temperature, are the mechanical stresses associated with temperature cycling of the components during power cycling and the difficulty of service. We can reduce temperature cycling by maintaining the machine at low temperature even during power down. In addition, experiments involving thermal stressing of CMOS components shows that components manufactured primarily for room temperature environment continued to operate properly after repeated and accelerated thermal cycling [18] [28]. Hence we feel that components specifically designed for operation at 77K and storage at room temperature would have appreciable thermal cycling tolerance. As for service, fault tolerant machine architectures exhibiting graceful degradation, coupled with the reduced component failure rate at lower temperature should considerably reduce the need for cumbersome service.

In addition to the above, we believe that CRL circuits operating at low temperatures would have higher reliability than conventional CMOS. During conventional CMOS switching, the circuit temperature can rise above the average temperature for a short time. Since the failure rate is related to a high power of temperature, the overall failure rate will be higher than that predicted according to the average operating temperature. As CRL circuits disallow sudden transients, the failure rate of CRL devices will be lower than that of conventional CMOS operating at the same temperature. In addition the peak currents in CRL are kept to minimum thus further reducing other failure effect such as metal migration, cross talk, etc.

The fourth advantage of operating CMOS devices at 77K is the reduced leakage current of PN junctions. The reverse bias leakage current of a PN junction is exponentially related to temperature. Since CMOS devices use PN junctions for isolation, a reduction in leakage currents results in a reduction in power consumption due to leakage which is the major contributor to quiescent power consumption. Another avenue that lower leakage can lead to lower power consumption is in DRAM's. Currently, a DRAM module that is not used must continually be refreshed to retain the data. Typically, a DRAM cell not being refreshed loses its contents after 2ms. Operated at low temperature, DRAM's were shown to maintain data integrity without refreshing for up to 169 hours after which the experiment was stopped [32]. Similar results for other CMOS circuit were observed in [28] where the authors reported a quiescent current of about 10^{-10} Amps which was the limit of their test board isolation. This is even more important for CRL circuits. In CRL it is possible for a node to be charged and left unconnected for a number of clock cycles. Eventually, this node should be "refreshed" if it is to keep the desired voltage level. Like DRAM, the refresh is both dissipative and takes time away from computing. With lower leakage, both CRL and DRAM's achieve lower power consumption and in some cases higher utilization.

The fifth is the advantage of lower operating voltage. To eliminate the probability of falsely triggering a circuit by thermal voltage, the threshold voltage of devices is chosen to be larger than a safe multiple of kT/q. We therefore expect V_T and hence V_{dd} to scale down with temperature as the thermal voltage. Operating at 77K, we can get by with V_T that is $298/77 = 3.87$ times lower than that required at room temperature and consuming 15 times less power due to similarly scaled V_{dd}. Please note that energy saving due to CRL is on top of the savings gained here and provides an avenue for further reducing the energy per operation after the saving from lower V_{dd} has been pushed to the limit.

Enhancement MOSFET parameters

The most important parameter improvement of a MOSFET at 77K is the improved carrier mobility. A number of mechanisms limit the low-field carrier mobility in semiconductors. The first is lattice scattering caused by lattice vibration and is the dominant mechanism at room temperature. Lattice vibration scattering decreases as the temperature drops below room temperature. The second is scattering caused by ionized impurities becoming important at very low temperatures, 4K, due to slower carrier thermal velocity. Since these effects track temperature differently, there is a point at which the effective mobility is maximum. Interestingly enough, this point is very close to 77K [34] [6] [14] [22]. At low-fields, low V_{DS} experimenters found that mobility and hence conductance was 4 times higher at 77K than at room temperature [15]. At high fields, mobility was only 1.7 times that of room temperature. This is because at high electric fields, V_{DS} on the order of few volts, the longer mean free path of the electrons at 77K coupled with high electric fields results in electrons attaining speeds that are higher than their thermal speeds, hot electrons. At this point scattering through optical phonons becomes very efficient at channeling energy from these fast electron to the lattice and hence the electron drift velocity reaches a maximum *saturation velocity* independent of the field applied. In a conventional CMOS circuit both low-field and high field situation occur leading to an increase in mobility at 77K that is between 1.7 and 4 such as 2.4 [28]. Since the core premise of CRL circuits is to avoid switching devices while there is a potential across them, CRL operation is strictly in the low-field region. This means that CRL circuits will always observe of the 4 times increase in mobility at 77K as opposed to the effective 2.4 factor for conventional CMOS at 77K. Mobility is important in that it is directly related to transconductance. A higher transconductance leads to a lower RC time constant for the device and to faster operation. We therefore expect CRL devices to be 4 times faster at 77K. Another speed up mechanism is the reduction of the junction capacitance of the source and the drain with lower temperature. At 77K, the number of ionized impurities decreases due to the onset of "freeze out" and this lower concentration widens the space-charge region thus decreasing the effective capacitance of the reverse biased isolation PN junctions of the source and drain [22]. Note that carrier freeze out does not affect enhancement mode MOSFET's since the carriers in the channel do not come from the thermally ionized impurities of the channel but are due to band bending by the gate voltage.

The second parameter improvement is the steeper sub threshold slope of the I_{DS} versus V_{GS} curve [14]. Previously, we stated that we can get by with a lower V_T at 77K. Our argument was purely based on lower thermal voltage. However, low threshold voltage would result in higher sub threshold conduction resulting in higher power consumption due to sub threshold leakage. Fortunately, the steeper slope, 4 times steeper, of the I_{DS} versus V_{GS} curve that results at 77K reduces sub threshold conduction enough to allow for lower V_T and V_{DD} with the power saved by the lower voltages not lost because of sub threshold leakage. Please remember that energy consumption in CRL circuits due to irreversibility is related to V_T, not V_{DD}, and would drop considerably as V_T becomes small at 77K.

The third improvement is the fact that unlike conventional CMOS, the power delay product of CRL drops with temperature through improved RC products of the devices at 77K.

Enhancement MOSFET are more susceptible to hot electron injection into the gate oxide altering V_T [20]. However, CRL guarantees low-field conditions in the channel because of low V_DS during switching thus drastically reducing the production of hot electrons in the channel.

Another MOSFET parameter improvement is decreased susceptibility to latch-up [10]. This is due to poorer bipolar performance and reduced bulk resistance at 77K. In addition, we feel that latch-up is further reduced by CRL techniques due to the absence of transients and reduced capacitive coupling effects.

For the above we see that the majority of material and device parameter improvements occur at 77K. Furthermore for all of the work referenced above normal CMOS operation of room temperature devices persisted down to 77K. Gaensslenn et al [15] have showed that equations modeling the behavior of enhancement mode MOSFET continued to tracked experimental results down to 77K with minor modifications.

One adverse parameter change is a variation of the threshold voltage with temperature. Experimental data published in [15] [40] shows a 0.25 volts increase in V_T at 77K from that at room temperature. The results are stated to be the same for both long and short channel devices. This is not a serious problem in that this shift could be accurately predicted and compensated for when designing circuits that are intended for 77K operation. However testing of these compensated circuits at room temperature becomes a little tricky as V_T becomes very low resulting in false triggering and high sub threshold conduction.

Finally, operating at temperatures lower than the temperature of the surrounding environment requires additional power for refrigeration. The theoretical coefficient of refrigeration, c, is the ratio of the work expended, W, to the heat pumped, Q, and is equal to

$$c = \frac{Q}{W} = \frac{T_c}{T_r - T_c}$$

where T_c is the temperature of the cooled device and T_r is room, or surrounding environment, temperature. For $T_c = 77$K and $T_r = 300$K, $c = 0.34$. This means that for every Joule our circuit dissipates at 77K, we have to supply a theoretical minimum of $1/c = 2.89$ Joules for refrigeration. The 3.9 times power multiple due to refrigeration is well compensated for by CRL and the improved circuit characteristics at 77K.

The above results are all attributed to experiments on devices that were optimized for room temperature operation. Less data is available for devices fabricated with LNT operation in mind. We believe that such devices should exhibit better parameter improvements due to their optimization. We want to distinguish however between optimization for LNT and operation limited to LNT. We think that devices limited to LNT had better demonstrate a sizable improvement so as to justify forfeiting room temperature testing and characterization. Furthermore, our emphasis above has been on enhancement mode Si MOSFET's. We acknowledge the existence of other devices with far more superior characteristics such as MODFET's and Silicon-on-Insulator (SOI) devices. But since our research is circuits and systems oriented and because the advantages of CRL techniques run orthogonal to improvements due to better devices and operating conditions, we limited our focus to Si MOSFET's.

B. SCRL-1 Pinout

Pin #	Pin Name	Pin #	Pin Name	Pin #	Pin Name
1	r4Pass<1>	29	aIn<3>	57	f2Pass<1>
2	r4Pass<0>	30	aIn<4>	58	f4SlowRail<0>
3	r4FastRail<1>	31	aIn<5>	59	f4FastRail<1>
4	r4FastRail<0>	32	aIn<6>	60	f4FastRail<0>
5	r4SlowRail<1>	33	bIn<0>	61	f4FastRail<1>
6	r4SlowRail<0>	34	bIn<1>	62	f4Pass<0>
7	r3Pass<1>	35	bIn<2>	63	f4Pass<1>
8	r3Pass<0>	36	bIn<3>	64	VCC (Substrate)
9	r3FastRail<1>	37	bIn<4>	65	enableAB
10	r3FastRail<0>	38	bIn<5>	66	enableR
11	r3SlowRail<1>	39	bIn<6>	67	rOut<15>
12	r3SlowRail<0>	40	f1SlowRail<0>	68	rOut<14>
13	r2Pass<1>	41	f1FastRail<1>	69	rOut<13>
14	r2Pass<0>	42	f1FastRail<0>	70	rOut<12>
15	r2FastRail<1>	43	f1FastRail<1>	71	rOut<11>
16	r2FastRail<0>	44	f1Pass<0>	72	rOut<10>
17	r2SlowRail<1>	45	f1Pass<1>	73	rOut<9>
18	r2SlowRail<0>	46	f2SlowRail<0>	74	rOut<8>
19	r1Pass<1>	47	f2SlowRail<1>	75	rOut<7>
20	r1Pass<0>	48	f2FastRail<0>	76	rOut<6>
21	r1FastRail<1>	49	f2FastRail<1>	77	rOut<5>
22	GND (P-Wells)	50	f2Pass<0>	78	rOut<4>
23	r1FastRail<0>	51	f2Pass<1>	79	rOut<3>
24	r1SlowRail<1>	52	f3SlowRail<0>	80	rOut<2>
25	r1SlowRail<0>	53	f3SlowRail<1>	81	rOut<1>
26	aIn<0>	54	f3FastRail<0>	82	rOut<0>
27	aIn<1>	55	f3FastRail<1>	83	padsSlowRail<0>
28	aIn<2>	56	f3Pass<0>	84	padsSlowRail<1>

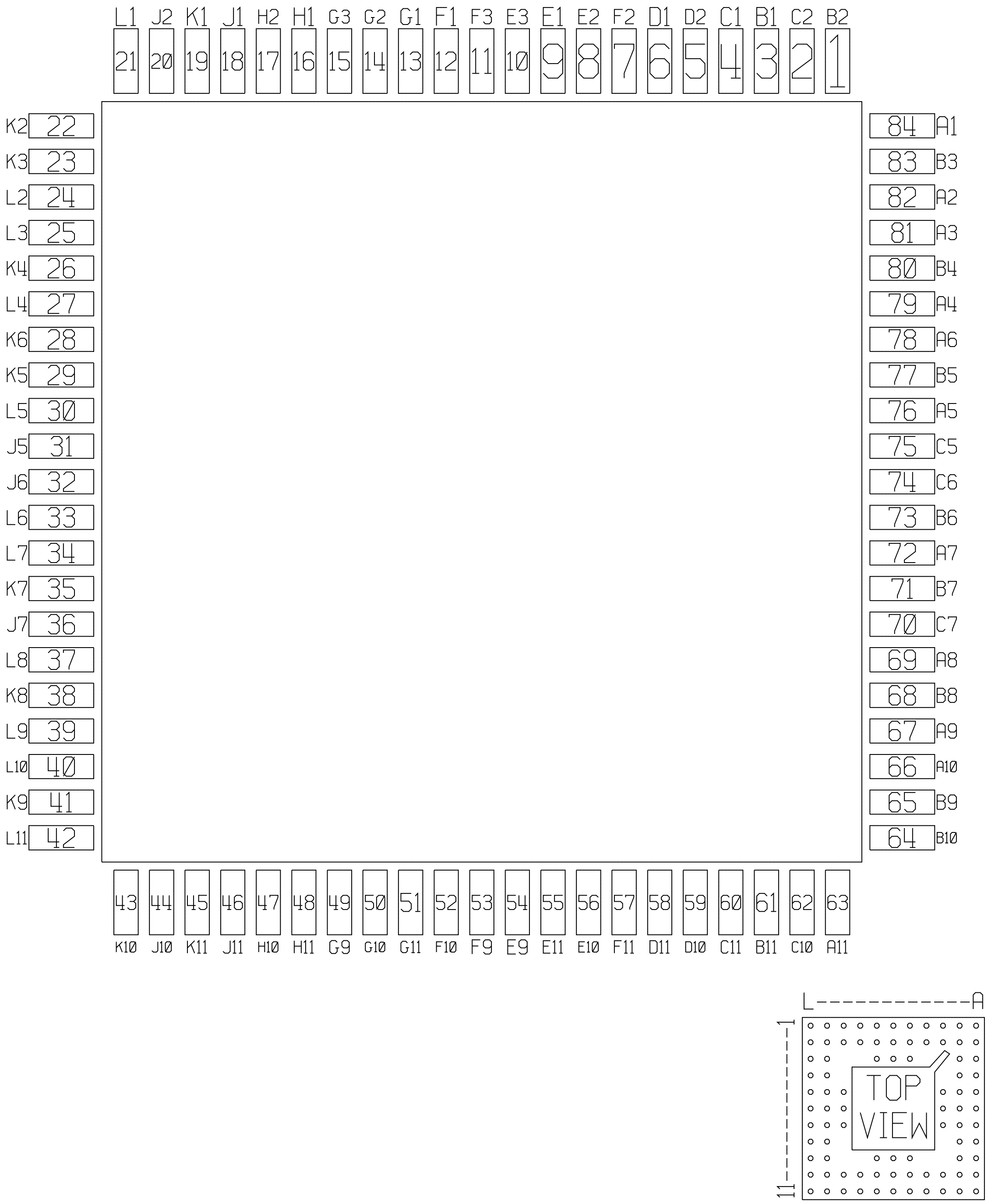


Figure B.1: Pinout map of the PGA84 package of SCRL-1.

Bibliography

[1] Athas, W. C., Koller, J. G., and Svensson, L. , "An Energy-Efficient CMOS Line Driver Using Adiabatic Switching," *Proc. 1994 IEEE Great Lakes Symposium on VLSI,* March, 1994.

[2] Baccaletti, G., Borri, F.R., D'Espinosa, G., Fioravanti, G., and Ghio, E., "Accelerated Tests," *Microelectronic Reliability*, Ed. Emiliano Pollino, pp. 361-374, Artech House, Massachusetts, 1989.

[3] Bennett, C., "The Thermodynamics of Computation – a Review", *International Journal of Theoretical Physics*, Vol. 21, No. 12, 1982, pages 905-940.

[4] Burr, J. B., Shott, "A 200mV Self-Testing Encoder/Decoder using Stanford Ultra-Low-Power CMOS," *IEEE Solid State Circuits Conference*, 1994, pp.

[5] Calebotta, S., "CMOS, the Ideal Logic Family", *National Semiconductor CMOS Databook,* Rev. 1, AN-77, pp. 2-3, 1988.

[6] Coeure, Ph., "Cryogenic Devices," *11th European Solid State Device Research Conf.*, 1981,pp. 153-170.

[7] Daembkes, H. (editor), *Modulation-Doped Field-Effect Transistors: Principles, Design, and Technology,* IEEE Press, 1990. 1990.

[8] Daembkes, H. (editor), *Modulation-Doped Field-Effect Transistors: Applications and Circuits, Transistors,* IEEE Press, 1990.

[9] Denker, J. S., Avery, S. C., Dickinson, A. G., Kramer A., and Wick, T. R., "Adiabatic Computing with the 2N-2N2D Logic Family," *Proceedings of the 1994 International Workshop on Low Power Design,* Napa, 1994, pp. 183-187.

[10] Dooley, J.G., and Jaeger, R.C., "Temperature Dependance of Latchup in CMOS Circuits," *IEEE Electron Devices Lett.*, vol. EDL-5, pp. 41-43, February 1984.

[11] Feynman, R., "Quantum Mechanical Computers", *Foundations of Physics,* Vol. 16, No. 6, 1986.

[12] Fredkin, E., and Toffoli, T. (1978),"Design Principles for Achieving High-performance Submicron Digital Technologies," Proposal to DARPA, MIT Laboratory for Computer Science.

[13] Fredkin, E., and Toffoli, T., "Conservative Logic", *International Journal of Theoretical Physics*, Vol. 21, Nos. 3/4, 1982, pages 219-253.

[14] Gaensslen, F.H., "MOS Devices and Integrated Circuits at Liquid Nitrogen Temperature," *IEEE Int. Conf. Circuits and Computers*, 1980, vol. 1, pp. 450-452.

[15] Gaensslen, F.H., Rideout, V.L., Walker, E.J., and Walker, J.J., "Very Small MOSFET's for Low-Temperature Operation," *IEEE Trans. Electron Devices*, vol. ED-24, pp. 218-229, March 1977.

[16] Hall, J.S., " An Electroid Switching Model for Reversible Computer Architectures," in *Proceedings of Physics of Computation Workshop,* Dallas Texas, October 1992.

[17] Hinman, R. T., Schlecht, M., F., "Recovered Energy Logic: A Single Clock AC Logic," *Proceedings of the 1994 International Workshop on Low Power Design,* Napa, 1994, pp. 153-158.

[18] Howe, D.A., "Integrated Circuits at Cryogenic Temperatures," *Cryog.*, vol. 18, no. 1, pp. 53-54, January 1978.

[19] Hust, J.G., "Thermal Conductivity and Thermal Diffusivity," *Materials at Low Temperatures* Ed. Richard P. Reed and Alan F. Clark, American Society for Metals, Metals Park, OH, 1983.

[20] Itsumi, M., "Electron Trapping in Thin Films of Thermal SiO_2 at Temperatures Between 30 and 300K," *J. Appl. Physics*, vol. 54, no. 4, pp. 1930-1936, April 1983.

[21] Jonscher, A.K., "Semiconductors at Cryogenic Temperatures," *Proc. IEEE*, vol. 52, pp. 1092-1104, October. 1964.

[22] Kamgar, A., Johnston, R.L., "Delay Times in Si MOSFET's in the 4.2-400K Temperature Range," *Solid-State Electronics*, vol. 26, no. 4, pp. 291-294, April 1983.

[23] Keyes, R.W., Harris E.P., and Konnerth, K.L., "The Role of Low Temperature in the Operation of Logic Circuitry," *Proc. IEEE*, vol. 58, pp. 1914-1932, December 1970.

[24] Kirschman, R.K., *Low Temperature Electronics*, IEEE Press, Order Number PC01974, New York, 1986.

[25] Kirschman, R.K., "Cold Electronics: An Overview," *Cryog.*, vol. 25, no. 3, pp. 115-122, March 1985.

[26] Koller, J.G., and Athas, W.C., "Adiabatic Switching, Low Energy Computing, and the Physics of Storing and Erasing Information," in *Proceedings of Physics of Computation Workshop,* Dallas Texas, October 1992.

[27] Landauer, R., "Uncertainty Principle and Minimal Energy Dissipation in a Computer", *International Journal of Theoretical Physics*, Vol. 21, Nos. 3/4, 1982, pages 283-297.

[28] Laramée, J., Auburn, M.J., and Cheeke J.D.N., "Behavior of CMOS Inverters at Cryogenic Temperatures," *Solid-State Electronics*, vol. 28, no. 5, pp. 453-456, May 1985.

[29] Larbalestier, D., "Critical Currents and Magnet Applications of High-T_c Superconductors," *Physics Today*, June 1991.

[30] Lee, C. P., Lee, D. H., Miller D. L., and Anderson, R. J., "Ultra High Speed Digital Integrated Circuits Using GaAs/GaLaAs High Electron Mobility Transistors." *Rec. of the IEEE GaAs Integrated Circuit Symposium,* pp. 162-165, 1983.

[31] Lee, Thomas H., "A fully integrated inductorless FM Receiver," Doctoral thesis, M.I.T. EECS Department, 1990.

[32] Link, W., May, H., "Transistorspeicherzellen bei tiefen temperaturen," *Archiv für Electronik und Übertragungstechnik*, vol. 33, no. 66, pp. 229-235, June 1979 Form Richard Jaeger and Fritz Gaensslen, "MOS Devices and Switching Behavior," *Low-Temperature Electronics*, Ed. Randall K. Kirschman, pp. 90-93, IEEE Press, 1986.

[33] Merkle, R.C., "Reversible Electronic Logic Using Switches", Submitted to *Nanotechnology*, 1992.

[34] Muller, R.S., and Kamins T.I., *Device Electronics for Integrated Circuits*, Second Edition, pp. 28-28, John Wiley & Sons, 1986.

[35] Pierret, R. F., "Field Effect Devices," *Addison-Wesley Modular Series on Solid State Devices*, vol. 4, 1990.

[36] Ressler, A. L., "The Design of a Conservative Logic Computer and a Graphical Editor Simulator," *MIT MS Thesis in EECS,* MIT, January 1981.

[37] Seitz, Charles L. *et al.*, "Hot-Clock n*MOS*," in *Proceedings of the 1985 Chapel Hill Conference on VLSI,* Computer Science Press, 1985.

[38] Svensson, L., and Koller, J., "An Off-chip Driver with Power Dissipation Less than fCV^2," *Submitted to CICC-94*, 1994.

[39] Touloukian, Y.S., Powell, R.W., Ho, C.Y. and Klemens, P.G., "Thermal Conductivity," *Thermophysical Properties of Matter*, Vols. 1 and 2, Plenum Press, New York, 1970.

[40] Vadasz, L., and Grove, A.S., "Temperature Dependence of MOS transistor Characteristics Below Saturation," *IEEE Trans. Electron Devices*, vol. ED-13, pp. 836-866, 1966.

[41] Younis, S., Knight, T., F., "Practical Implementation of Charge Recovering Asymptotically Zero Power CMOS," *Proc. of the 1993 Symposium on Integrated Systems,* MIT Press, 1993, pp. 234-250.

[42] Younis, S., Knight, T., F., “Asymptotically Zero Energy Split-Level Charge Recovery Logic,” *Proceedings of the 1994 International Workshop on Low Power Design,* Napa, 1994, pp. 177-182.