Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences

A Split-Rail Bennett-Clocked Implementation of an 8-bit Adiabatic ALU for Use in a MIPS Microprocessor.

A thesis presented by

César Orlando Campos Aguillón

Submitted to the
School of Engineering and Sciences
in partial fulfillment of the requirements for the degree of

Master of Science

In

Electronics Engineering

Monterrey Nuevo León, December 1st, 2014

Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences

The committee members, hereby, certify that have read the thesis presented by César Orlando Campos Aguillón and that it is fully adequate in scope and quality as a partial requirement for the degree of Master of Science in Electronics Engineering,

Dr. Graciano Dieck Assad
Tecnológico de Monterrey
School of Engineering and Sciences
Principal Advisor

Ing. Juan M. Hinojosa Olivares
Tecnológico de Monterrey
Committee Member

Dr. Alfonso Ávila Ortega
Tecnológico de Monterrey
Committee Member

Dr. Jorge Welti Chanes
Associate Dean of Graduate Studies
School of Engineering and Sciences

Monterrey Nuevo León, December 1st, 2014

Declaration of Authorship

I, César Orlando Campos Aguillón, declare that this thesis titled, “A Split-Rail Bennett-Clocked Implementation of an 8-bit Adiabatic ALU for Use in a MIPS Microprocessor” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

César Orlando Campos Aguillón
Monterrey Nuevo León, December 1st, 2014

@2014 by César Orlando Campos Aguillón
All rights reserved

Dedication

To my family for their unconditional love, support and patience during all this time. You are my reason of being and the reason I always push forward.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Dr. Graciano Dieck for the opportunity to do this research, all the advice, the coaching and the many hours spent discussing exciting new prospects together. Your support was invaluable to me from day one and you were always in the disposition to help.

Many thanks as well to Dr. Gregory Snider at University of Notre Dame. You received me with arms open and guided me along the way. For all the advice, insights and priceless experience, thank you.

To Dr. Ismo Hänninen for the joint work done on this project. It is always a pleasure to work with such passionate people like yourself.

To the crew at the University of Notre Dame NURF program for keeping me company during the long hours. And especially to René Celis for all the hard work we did together both abroad in Notre Dame and back at Tec de Monterrey. I can only respect and admire such a hardworking and dedicated peer. Thanks to Jaime Saldaña as well for the support in the final stages of layout design.

I would also like to thank my Committee members for the additional advice and for taking an interest in this project. New technologies sometimes surprise us but you took a leap of faith with me and believed in this project and I appreciate that.

To my school, Tecnológico de Monterrey for the opportunity to study, the tuition support and the many, many years spent in its classrooms and hallways.

Also many thanks to CONACyT for the financial support that sustained me these past two years and allowed me to visit Notre Dame.

A Split-Rail Bennett-Clocked Implementation of an 8-bit Adiabatic ALU for Use in a MIPS Microprocessor.

by

César Orlando Campos Aguillón

Abstract

The Landauer Principle (LP) states that destruction of information in computational systems carries an inherent penalty of $k_B T \ln(2)$ joules of energy dissipation. Conversely, if bit erasure is avoided it is possible to achieve near zero dissipation computation. Recent findings have demonstrated that this assertion is correct and that practical asymptotically zero dissipation systems are possible. Such systems are called adiabatic. Furthermore, several implementations to achieve this operation exist in literature.

This project aims to model, design, simulate and fabricate a near dissipationless adiabatic Large Scale Integration (LSI) system using 0.5 μm CMOS technology. For this purpose, the split-rail Bennett-clocked logic family was chosen. An adiabatic ALU is designed for use in a MIPS processor. Simulation of the device shows that it is able to recover more than 97% of the energy in adiabatic mode at an operating frequency of 10 MHz.

The ALU design and the complete MIPS processor were sent for fabrication using both ON Semiconductor C5 0.5 μm CMOS process through MOSIS and a custom 1 μm process at University of Notre Dame in Indiana.

List of Figures

Figure 1.1 1
Figure 2.1 9
Figure 2.2 10
Figure 2.3 10
Figure 2.4 11
Figure 2.5 12
Figure 2.6 12
Figure 2.7 13
Figure 2.8 14
Figure 2.9 14
Figure 3.1 18
Figure 3.2 19
Figure 3.3 20
Figure 3.4 23
Figure 4.1 26
Figure 4.2 27
Figure 4.3 27
Figure 4.4 28
Figure 4.5 29
Figure 4.6 30
Figure 4.7 30
Figure 4.8 31
Figure 4.9 32
Figure 4.10 32
Figure 4.12 34
Figure 4.13 34
Figure 4.14 35
Figure 4.15 36
Figure 4.16 36
Figure 4.17 37
Figure 4.18 39
Figure 5.1 40
Figure 5.2 41
Figure 5.3 42
Figure 5.4 43
Figure 5.5 44
Figure 5.6 46
Figure 5.7 47
Figure 5.8 47
Figure 5.9 48
Figure 5.10 50
Figure 5.11 51
Figure 5.12 52
Figure 5.13 54
Figure 5.14 55
Figure 5.15 56
Figure 5.16 56

Figure 5.17 57
Figure 6.1 60
Figure 6.2 60
Figure 6.3 61
Figure 6.4 62
Figure 6.5 62
Figure 6.6 63
Figure 6.7 64
Figure 6.8 64
Figure 6.9 65
Figure 6.10 66
Figure 6.11 67
Figure B.1 75
Figure B.2 75
Figure B.3 76
Figure B.4 76
Figure B.5 77
Figure B.6 77
Figure B.7 77
Figure B.8 78
Figure B.9 78
Figure B.10 78
Figure B.11 79
Figure B.12 79
Figure B.13 79
Figure B.14 80
Figure B.15 80
Figure B.16 80

List of Tables

Table 3.1 21

Table 5.1 43

Table 5.2 45

Table 5.3 46

Table 5.4 49

Table 5.5 49

Table 5.6 53

Contents

Abstract v
List of Figures vi
List of Tables........ viii
Chapter 1. Introduction. 1
  1.1 – Power dissipation in CMOS circuits........ 1
  1.2 – Ultimate Shannon Limit and the Landauer Principle. 2
  1.3 – Implementation of Dissipationless Reversible Computing Devices. 3
  1.4 – Problem statement. 4
  1.5 – Outline of the work. 6
Chapter 2. Fundamentals of Adiabatic Switching and Reversible Computing. 7
  2.1 – Rules of Adiabatic Switching Systems. 7
  2.2 – Fully adiabatic logic. 8
  2.3 – Quasi-adiabatic logic. 10
  2.4 – Complex Device Implementations. 14
  2.5 – Why use Bennett-clocked adiabatic logic? 16
  2.6 – Chapter conclusions. 16
Chapter 3 – A Bennett-Clocked Adiabatic Implementation. 18
  3.1 – Proof of concept. 18
  3.2 – System Verilog HDL Model 20
  3.3 – Chapter Conclusion. 22
Chapter 4. Adiabatic Bennett-Clocked Standard Cell Library. 25
  4.1 - General design choices. 25
  4.2 - Standard Cells 26
    4.2.1 - Inverters 26
    4.2.2 – NAND gates 28
    4.2.3 – NOR gate. 29
    4.2.4 – Composite gates. 31
    4.2.5 – XOR gate 32
    4.2.6 – TG-based Multiplexers 33
    4.2.7 – TG-based conditional inverter 35
  4.3 – Power VS Speed Trade-Off for an Adiabatic Inverter Gate. 36
  4.4 – Chapter conclusion 39
Chapter 5. Adiabatic 8-bit ALU with Carry-Lookahead Adder. 40
  5.1 – Design of the ALU. 40
    5.1.1 – Bit slice design. 43
    5.1.2 – Carry computation sub-module design. 45
    5.1.3 – Standalone adiabatic ALU for MOSIS C5 Fabrication. 48
  5.2 – Circuit performance in simulation. 51

5.3 – Fabrication process. 54
5.4. – Chapter conclusion. 58
Chapter 6 – Adiabatic 8-bit MIPS Microprocessor. 59
6.1 – Adiabatic MIPS architecture. 59
6.2 – MIPS-specific layouts. 61
6.2.1 – D-type Flip-Flops. 61
6.2.2 – SRAM cells. 62
6.2.3 – Zero detector. 63
6.2.4 – PC Enable Logic. 63
6.2.5 – Buffering Multiplexers. 63
6.3 – MIPS Processor Layout and Fabrication. 64
6.4 – Chapter conclusions. 67
Chapter 7. Conclusions. 68
7.1 – Particular Objectives. 68
7.1.1 – Objective One. 69
7.1.2 – Objective Two. 69
7.1.3 – Objective Three. 70
7.1.4 – Objective Four. 70
7.1.5 – Objective Five. 70
7.1.6 – Objective Six. 71
7.1.7 – Objective Seven. 71
7.1.8 – Objective Eight. 72
7.2 – Future work. 72
7.3 – Closing Statements. 73
Appendix A – Acronyms 74
Appendix B – Additional layout realizations. 75
Appendix C - SPICE MOSIS Transistor Models and Technology Parameters. 81
Appendix D - SPICE Testbenches for Adiabatic Operation. 84
Appendix E – MOSIS Fabrication Confirmation Document. 91
Bibliography 92

Chapter 1. Introduction.

Ever since Moore's law was first enunciated in 1965 **[1]**, the electronics industry has been striving to keep up with expectations of exponentially rising device complexity. These have been met largely due to the scalability of the Complementary Metal-Oxide Semiconductor (CMOS) process, which has permitted a larger integration as fabrication methods become more refined. Even though Moore's empirical observation has held up until today, it must necessarily come to an end because of physical limits of the devices used for computation.

For the past few decades the electronics industry has identified possible road blocks to Integrated Circuit (IC) development, in an attempt to break through them before they are reached. Several of these have been reported in literature **[2]** **[3]**. They include limits in materials, interconnects, fabrication processes and sheer device complexity, among others.

Out of these, the most critical limitation has arguably been energy dissipation. Several of its consequences are already visible to the end-user in consumer electronics, reflected in high operating temperatures and short battery duration. This accentuates the need for a solution with lower power consumption.

Figure 1.1, from Snider et al. **[4]** shows a plot of the power density (W/cm^2) of several Intel products versus release time. From the middle 1980s to the early 2000s power density grew steadily, mirroring the rise in number of transistors as per Moore's law. In the 2000s, however, this trend was broken to remain below the practical threshold for air-based cooling. From this point onwards, the industry has moved away from increasing operating frequency and has concentrated instead on keeping power dissipation under manageable levels while pursuing Moore's law.

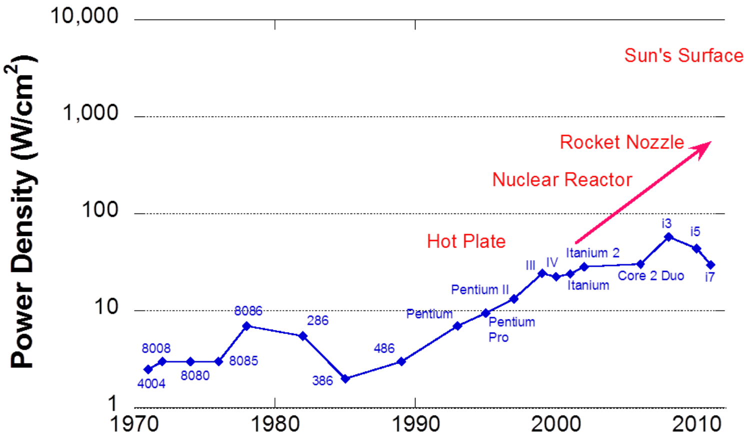


Figure 1.1 – Power density in Intel devices, from **[4]**.

1.1 – Power dissipation in CMOS circuits.

Power dissipation in a CMOS system may be static or dynamic. Dynamic dissipation occurs because of node voltage transitions as the circuit performs useful work. Static dissipation is due to leakage currents inherent to the energized transistor. Equation 1.1, from **[5]** shows total power dissipation for a CMOS circuit. Methods to minimize power consumption exist for both static and dynamic dissipation.

$$P_{total} = P_{dynamic} + P_{static} \tag{1.1}$$

Dynamic dissipation has two components: switching and short circuit power. Equation 1.2 from **[5]** shows this relationship. Switching power is described in more detail in equation 1.3, from **[5]**, where *C* is the load capacitance, V_{DD} is the bias voltage of the circuit, α is the activity factor (how likely the circuit is to prompt a voltage switch in the output node), and f is the operating frequency of the system. Short circuit power is caused by a transient effect during switching, where PMOS and NMOS devices are switched on simultaneously for a brief period. In most cases, switching power is the dominant component of dynamic dissipation.

$$P_{dynamic} = P_{switching} + P_{short-circuit} \tag{1.2}$$

$$P_{switching} = \alpha C V_{DD}^2 f \tag{1.3}$$

Not all factors can be easily optimized to keep switching power at a minimum. The load capacitance depends on the fan out of each gate and is therefore tied to logic design. Activity factor is determined by the task at hand and varies over time. Therefore, we only have control over the power supply voltage and the operating frequency. Both are often selected to be the minimum quantities that can still perform the required tasks.

Unfortunately, minimization of switching power is at odds with that of static power. The main component of static dissipation is subthreshold leakage. It is shown in **[5]** that subthreshold leakage current has a negative exponential dependence on V_{DD}. This sets a limit on the minimum bias voltage before static losses offset gains in dynamic dissipation thus preventing V_{DD} from scaling too low.

The industry has tried a few other approaches to minimize power dissipation, such as multicore computing **[6]** and dark silicon **[7]**. Although they result in lower dissipation, these technologies are bound by a fundamental lower limit which will be explored in the next subsection.

1.2 – Ultimate Shannon Limit and the Landauer Principle.

Widespread agreement exists that a fundamental limit exists on the energy necessary to create a bit of information that is distinguishable from noise **[8] [9] [10]**. This limit, known as the "Ultimate Shannon Limit" (USL), is;

$$USL = k_B T ln(2) \tag{1.4}$$

Where k_B is Boltzmann's constant and *T* is the absolute temperature.

Whenever a bit of information is destroyed, an amount of energy equal to the energy stored in the bit is dissipated into heat. Landauer first stated in 1961 that a system that undergoes bit destruction will necessarily dissipate at least $k_B T ln(2)$ Joules of energy **[11]**. Conversely, a system that does not destroy information has no such fundamental lower limit. This statement is known as the "Landauer Principle" (LP) and forms the theoretical justification for the field of reversible computing. By avoiding bit destruction, it is possible to achieve ultra-low power dissipation in computing.

Bennett extends the usefulness of LP by proving that any irreversible computation can be made reversible by making a copy of the information that would otherwise be destroyed **[12]**. These machines are also proven to be feasible in reality. If both LP and Bennett's statement hold true, then the possibility exists to create a computer with no inferior limit to power dissipation.

However, LP has been heavily criticized by many authors as unrealistic and proofs have been dismissed as flawed either due to arguable key assumptions **[9] [13]** or to poor argumentation **[14] [15]**. This criticism has sparked a trend to move away from charge as a state variable and explore alternative ways to represent information that might not share the same limitations.

Replies to this criticism have also surfaced in literature **[16] [17] [18]** reinforcing some of the points that have fallen under scrutiny. Alternative derivations of LP **[19] [20]** add to its credibility by arriving at the same conclusion from different standpoints. This discussion has mainly taken place in a theoretical front and remains controversial.

Experimental proof of LP has just begun to appear. Berut et al. demonstrate experimentally the first assertion of LP, that is, destruction of information has a lower power dissipation boundary in the USL **[21]**. This is consistent with the theory and even detractors agree that such a boundary exists.

The second statement of LP has been proven experimentally in a number of ways. Boechler et al. show that it is possible to dissipate energy below the USL in an experimental setup by avoiding information destruction **[22]**. This is true even when using charge as a state variable. Snider et al. show that a family of circuits exist that can dissipate energy below the USL and refutes notions that a state variable other than charge is necessary moving forward **[23]**. In another paper, Snider et al. demonstrate that the size of the energy barrier between logic states has no influence over the dissipated energy amount and boldly assert that there's no lower dissipation boundary for systems that preserve information **[4]**.

The groundwork for ultra-low power reversible computing is looking solid, but considerable practical challenges still remain in implementation. Some of these are the construction of power sources that accept energy back and design of reversible computing devices.

1.3 – Implementation of Dissipationless Reversible Computing Devices.

Even though LP states that no inferior energy dissipation limit exists for reversible computing operations, a method to transfer charge to and from a node without dissipation is needed to make (mostly) dissipationless systems a practical reality. Traditional CMOS circuits dissipate half of the bit energy across the active transistors when switching any node state **[5]** and thus are unsuitable for dissipationless computing. It is clear that a new family of electronic logic circuits is needed for the realization of a practical reversible computing system based on LP.

A system dissipates no energy only while it remains in perfect thermodynamic equilibrium **[24]**. However, the core function of an electronic computing system is to switch the potential at an output node based on its inputs. This implies that a transition in the output node breaks the equilibrium while the system updates its value. To truly have this operation be completely dissipationless, it would require an infinite number of infinitesimal voltage transitions, in which the system would remain at equilibrium throughout the whole task.

Practical realizations of such transitions are of course impossible, but can be asymptotically approached by a series of successive changes (i.e. a "voltage stair") at a slow enough rate to achieve effective equilibrium across each step. This process is called adiabatic switching. Younis et al. describe it as a chain of quasistatic changes along neighboring equilibrium states **[25]**. Dissipation throughout the whole charging operation approximates zero asymptotically as the charging rate becomes slower because the intermediate steps are closer together. Since computation is often a time-sensitive process, it follows that a balance must be struck between achieving low power dissipation and computing speed.

Adiabatic switching demands adherence to several restrictions on IC design. Specialized hardware is usually necessary to comply with these requirements. Many realizations of logic gates capable of both adiabatic switching and reversible operation are found in literature **[26] [27] [28]**. For the purposes of this work the Bennett-Clocked adiabatic scheme was selected as described by **[25]** and **[27]**. The reasons for this choice and a comparative analysis of adiabatic techniques will be discussed in further detail in Chapter 2.

Although the main components such as logic gates and interconnection elements are well described, implementations of more complex systems are scarce. Of those implementations found, few reach the Large-Scale Integration (LSI) level and none have been able to produce experimental results **[29] [30] [31] [32] [33] [34] [35]**. Some of these systems will be described in Chapter 2.

Another major obstacle to the implementation of dissipationless computing lies in the power sources used to control adiabatic switching. The waveforms necessary for dissipationless charge transfer can be created with traditional wave generators, but they dissipate power themselves to generate the desired signal. Furthermore, traditional power sources are unable to accept energy coming back into them.

Some of the proposed solutions to the power source issue are described in **[32] [36] [37]** and are based on resonant circuits. A resonant system can circulate energy between its internal inductor and any capacitive load connected to it. A traditional power source can then provide the initial surge and then compensate only for the resistive losses, essentially supplying only the power dissipated inside the circuit. The existence of this problem is acknowledged but is not in the scope of this thesis.

1.4 – Problem statement.

As discussed in the previous subsection, a gap in the knowledge exists regarding complex adiabatic circuit design. The strict requirements of adiabatic switching hardware demand special considerations from the design engineer and present unique challenges not seen in traditional digital IC design.

In particular there is special interest on producing a working adiabatic microprocessor because this would signify an actual useful adiabatic computing device. Even complex components of a microprocessor such as the Arithmetic Logic Unit (ALU) have not been tested experimentally yet. These devices pertain to the LSI category of ICs and pave the road for design of systems with even larger scale integration.

Another gap exists in the power dissipation versus computing speed trade-off. As with traditional CMOS, this depends on technology parameters and circuit design. Since no LSI systems have been produced yet, no benchmark exists to make a quantitative assessment of the advantages of using adiabatic reversible computing.

Based on these problems, the main objective of this work is the design of an adiabatic 8-bit ALU for use in a MIPS microprocessor. The device is modeled first, using a HDL (High Level Description Language) methodology and afterwards, using A CMOS 0.5μm process to prepare the prototype fabrication cycle. The ALU is simulated using a SPICE software and a comparison between dissipations both, in the adiabatic circuit and the traditional CMOS, is performed. The ALU is also sent for fabrication through the MOSIS system using the ON semiconductor C5 technology. Additionally, an adiabatic prototype for a MIPS microprocessor is designed and fabricated using the same methodology.

The particular objectives of this master´s thesis are:

1. To analyze the various schemes that perform dissipationless computing using CMOS technology.
2. To use the split-rail Bennett-clocked adiabatic technique to develop a prototype of an ALU device, considering its advantages over other adiabatic design methods.
3. To model the ALU device using HDL methodology and validate its behavioral performance.
4. To create a standard cell library in layout for the Bennett-clocked method, that contains every device needed for a practical implementation of an ALU.
5. To model the ALU using the ON Semiconductor C5 0.5 μm CMOS process and prepare the fabrication blueprints of the Integrated Circuit (IC).
6. To simulate the ALU using the design couple Electric VLSI-LTspice in order to:
    a. Verify the electrical detailed performance of the circuit.
    b. Validate the performance considering parasitic elements in the IC.
    c. Generate the physical layout and validate its compliance with the DRC (design rule check) and other standard fabrication rules in the C5 process.
    d. Send the IC to MOSIS for fabrication.
7. To develop a trade-off and comparative analysis between the adiabatic ALU and the standard CMOS ALU in terms of frequency and power dissipation.
8. To apply the design methodology for a MIPS processor and to send the adiabatic MIPS for fabrication using the ON Semiconductor C5 process.

They hypothesis for this thesis is that it is possible to create ultra-low power LSI ICs using reversible computing and adiabatic charging techniques. These devices will perform the same useful computing work as their traditional CMOS counterparts but dissipate significantly less energy.

Design of low power digital ICs responds both to a trend in the industry and to a common end-user demand **[4]** as stated on the opening paragraphs of this section. Industry analysts and roadmaps predict that the usefulness of current power-reducing techniques will soon be outlived **[6] [7]**. Because of this, it is of paramount importance that new techniques be developed.

Some of the tasks in this research were performed in a collaboration between Tecnológico de Monterrey and University of Notre Dame, at the Notre Dame facilities in Indiana. Fabrication of the circuits designed for this thesis is being performed at the Notre Dame IC Fabrication Laboratory in 2 µm technology as well as with the MOSIS 0.5 µm process.

1.5 – Outline of the work.

This first chapter describes the theoretical background in which the problem is situated along with key terms concerning reversible computing and adiabatic switching.

Chapter two discussed details of the basic adiabatic circuit realizations and more complex systems found in literature. A description of the technique selected for this work is also found in this chapter.

Chapter three focuses on the Bennett-clocked implementation selected for this work. It begins with a proof of concept developed by Snider et al. **[4]** and a proprietary verification made in simulation. Then briefly describes a set of Hardware Description Language (HDL) tools developed for this project. The chapter finishes with a recount of the challenges and advantages of the proposed technology.

Chapter four is a detailed description of the standard cell library created for the adiabatic ALU design. It contains a description of each cell as well as schematics and layouts. A simulation of the minimum size inverter is used to establish a comparison between adiabatic and standard CMOS operation for a single cell.

Chapter five goes into detail for the adiabatic ALU. It contains a description of the implementation, discussion of many of the challenges encountered in its design and explanation of many of the design choices that were made. A simulation of the entire module is used to make sure that the adiabatic system performs the correct computation and to compare power dissipation of both traditional and adiabatic systems as a function of frequency.

Chapter six focuses on the adiabatic MIPS implementation. It describes the overall architecture of the microprocessor and the additional standard cells that were prepared for this purpose. It also shows the final layout sent for fabrication and identifies every component within the layout.

Finally, Chapter seven consolidates the thesis research with the conclusions derived from the various designs, simulations and discussions. Also, suggestions for future work are made.

Chapter 2. Fundamentals of Adiabatic Switching and Reversible Computing.

Chapter 1 defined the theoretical framework in which adiabatic systems are possible. This chapter explores practical implementations of adiabatic electronic circuits. The first subsection takes a look at power dissipation in an ideal adiabatic system and recounts some theoretical concepts to define the rules that it must follow to dissipate asymptotically zero power.

The second subsection takes a close look at a family of realizations called fully adiabatic. For each of these implementations a description is provided, along with special hardware considerations (such as any physical or logical overhead necessary), advantages and disadvantages of the technology and an example schematic to illustrate the concept. The same is done for quasi-adiabatic implementations in the third subsection of the chapter.

In the fourth subsection complex adiabatic implementations found in literature are described. In this context complex devices are defined as those composed of a couple of logic gates at least. Each example includes information about the device operation, adiabatic technique used, design of the circuit, type of test conducted in the paper and the results obtained. Many of the references report a lower power dissipation for adiabatic circuits than for standard CMOS at relatively low frequencies.

The fifth section of the chapter regards the decision to select Bennett-clocking as the adiabatic technique for the designs of this thesis. Advantages over other technologies are highlighted along with the failings. Finally, the chapter closes with a recap of all the information presented here.

2.1 – Rules of Adiabatic Switching Systems.

As discussed in the previous chapter, adiabatic transfer of charge between nodes is possible only by following a chain of quasistatic transitions. In this context, quasistatic is defined in terms of the RC time constant of the logic gate **[25]**. A given transition is said to be quasistatic if the voltage changes negligibly over the period of a single time constant. It follows that a system must operate at a sufficiently slow frequency for an adiabatic transfer to occur in its nodes. This enables the possibility to have arbitrarily low power dissipation by lowering the operation frequency.

The relationship between dissipated power and the frequency of a device in a fully adiabatic system is enunciated in equation 2.1, from **[4]**:

$$P_{total} = N\left[CV_{DD}^2 f\left(\alpha\frac{f}{f_0} - (1-\alpha)\right) + A\exp\left(\frac{qV_{DD}}{4\eta kT}\right)\right] \tag{2.1}$$

The first term between the square brackets in the expression represents the dynamic power, while the second term is static. N is the number of gates present in the system, α represents the amount of gates that operate in adiabatic mode (α=1 for fully adiabatic systems), f_0 is the characteristic frequency of the system, defined by the reciprocal of RC for the slowest gate present A is a constant and η is the ideality factor for the slope of the subthreshold current (typically 1). Focusing on the first term of the equation, it is clear that dynamic dissipation in an adiabatic system is reduced by a factor of f/f_0 and thus can be made arbitrarily small.

However, adiabatic charging is not the only requirement for dissipationless computing. As Landauer states in **[11]**, the USL can only be overcome by performing a reversible logic operation.

Bennett demonstrated that any computing operation can be made reversible by storing a copy of the intermediate results then "decomputing" the whole operation and erasing all results except for the final output **[12]**. This principle is applicable to any realization, however the task becomes easier if a family of inherently reversible logical operations are employed. Such implementations include Fredkin and Toffoli gates **[38]**.

Practical implementations of adiabatic computing have additional conditions for operation in thermodynamic equilibrium as defined by Starosel'skii **[27]**. These conditions are the existence of three logic states (relaxed, logic 0 and logic 1), that a cell can only receive information while in the relaxed state, inputs must remain fixed for the entirety of the computing cycle and gates must not have backlashes.

These conditions are derived from the principles put forward in the theoretical framework. The first and second conditions ensure that adiabatic circuits are driven externally by sources capable of adiabatic switching. The second, third and fourth ensure that the condition of reversibility is always observed. The fourth condition is generally always met as long as no feedback interconnections are made between gates.

Not every adiabatic system observes strict adherence to these conditions. Systems that comply with all requirements will be called fully adiabatic and can indeed operate with asymptotically zero dissipation. Power dissipation in these circuits follows equation 2.1.

Systems that fail some of the conditions either by merging the relaxed state with logic 0 or 1 or by charging the output node through a diode with a constant forward bias voltage are unable to reach such low dissipations but still offer some advantages over traditional CMOS systems. These will be called quasi-adiabatic systems. Modified versions of equation 2.1 will be examined in more detail when discussing each realization of quasi-adiabatic circuitry.

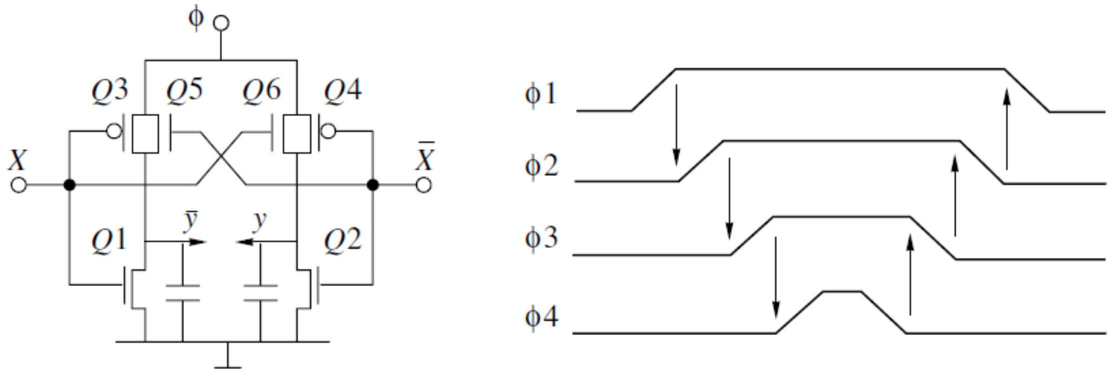
Some designs incorporate sections that operate in traditional CMOS mode along with fully or quasi-adiabatic modules. These can be accounted for by adjusting the activity factor term found in the power dissipation equation.

2.2 – Fully adiabatic logic.

Three families of fully adiabatic logic gates are found in literature. The first of these is known as the 2n2p-2n illustrated in Figure 2.1, adapted from **[27]**. This realization employs a series of nested pulses as shown in the figure when connected in cascade. This configuration can be alternatively described as a single-rail Bennett-clocked retractile cascade circuit. The term Bennett clocking in this context means that the output information is held in place by the clocks and erased in reverse order after full computation.

The relaxed state is represented by having both output nodes be equal to 0 V. This is different from the logic 0 state which has node $y = 0\ V$ and node $\bar{y} = V_{DD}$ thus meeting the first condition for fully adiabatic systems. The second condition is guaranteed by the timing of the nested pulses. The third condition in cascade circuits demands that the output of a gate be always coupled to one of the power rails. This is satisfied by having one of the transfer gates always transparent.

This adiabatic implementation can be modified to execute any of the basic logic operations, has a single power rail and both direct and complementary logic outputs. Unfortunately, it is unable to provide pipelining, it has a complex timing and requires a large overhead: six transistors for an inverter as opposed to two in traditional CMOS.



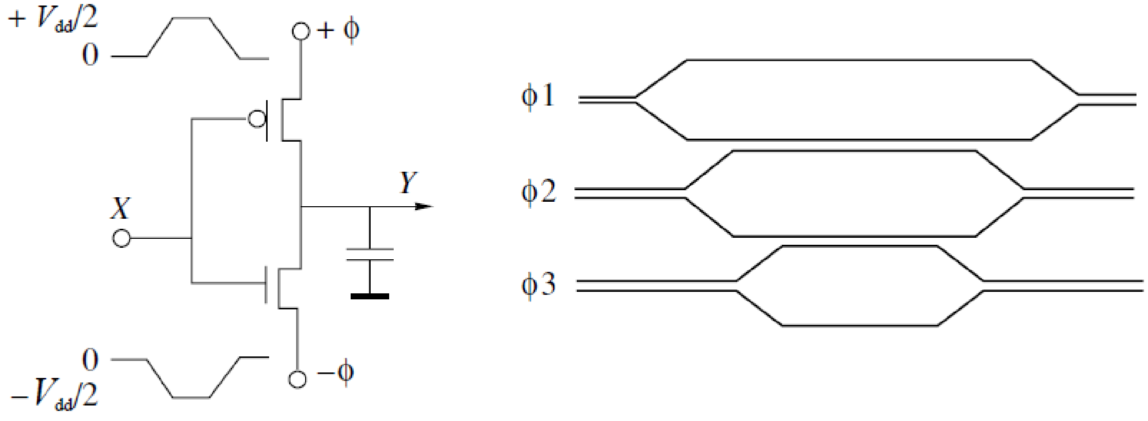
(a) *(b)*

Figure 2.1 – 2n2p-2n adiabatic circuit implementation. (a) shows the schematic for an inverter. (b) shows the nested power clocks used for polarization. Adapted from **[27]**.

The second family of fully adiabatic circuits receives the name of 1n1p logic. Figure 2.2, adapted from **[27]** shows the schematic for a typical inverter and the power clocks that energize the circuit when connected in cascade. Similar to the previous example, such a realization can be described as a split-rail Bennett-clocked retractile cascade circuit. Although both this and the 2n2p-2n circuit employ comparable nested clocks, we will refer to the split-level version when naming the term "Bennett clocking" on future occasions in this work.

Since the power rails are split in this realization, the 3 states for condition 1 are readily visible in the waveforms: the relaxed state corresponds to the midpoint, logic 0 is the lowest voltage and logic 1 is the highest voltage. Condition 2 is guaranteed by the timing of the nested clocks as with the single-rail version. The third condition for cascade systems is satisfied because in this realization either the PMOS or the NMOS will be switched on. As a result, the output is always coupled to one of the power rails.

The main advantage of 1n1p is that the gate design and topology is identical to standard CMOS. Every logic gate is realizable and the lack of overhead means that the same hardware can run in conventional CMOS mode just by connecting the power rails to V_{DD} and V_{SS} instead of the Bennett clocks. Disadvantages of this implementation include the lack of pipelining support and the high complexity of the Bennett clocks. For the split-level version, twice as many sets of clocks are needed than for the single-rail alternative.



(a) *(b)*

Figure 2.2 – 1n1p adiabatic circuit implementation. (a) shows the schematic for an inverter. (b) shows the nested power clocks used for polarization. Adapted from **[27]**.

Younis et al. **[25]** propose a pipelining scheme that makes use of transmission gates to uncouple the output node of a logic gate from the next stage in the cascade circuit in a split-level based system. This technique is known as split-level charge recovery logic (SCRL) and allows a better timing management, enables true pipelining and reduces the number of required clock phases for operation. Figure 2.3 from **[25]** shows a schematic representation of the pipelining system.

In SCRL every logic gate needs its own reverse operation. This causes a very large overhead as every gate has to be duplicated on the layout. This in turn increases the fan out of every gate, the time constant of the circuit, and thus diminishes speed of adiabatic operation. This reduction in speed however, might be offset by the overall speed gains of pipelining. Furthermore SCRL restricts combinatorial logic to the use of inherently reversible logic operations such as the Toffoli family of gates **[38]** which demands a drastically different approach to logic design.

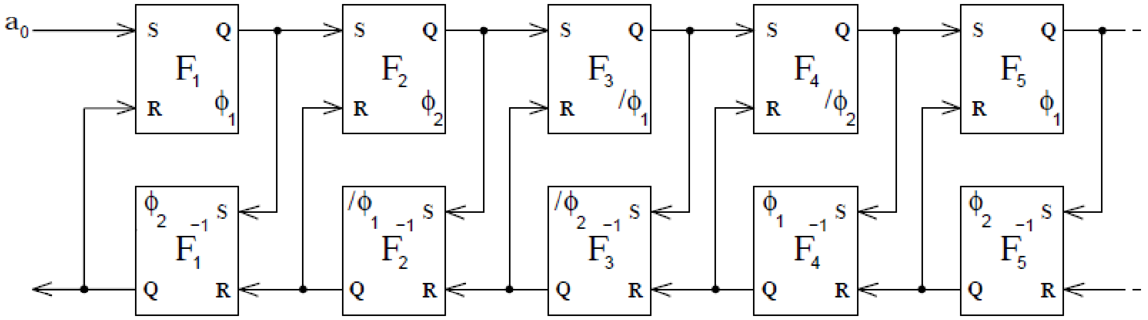


Figure 2.3 – Pipelining approach followed in the SCRL technique. From **[25]**.

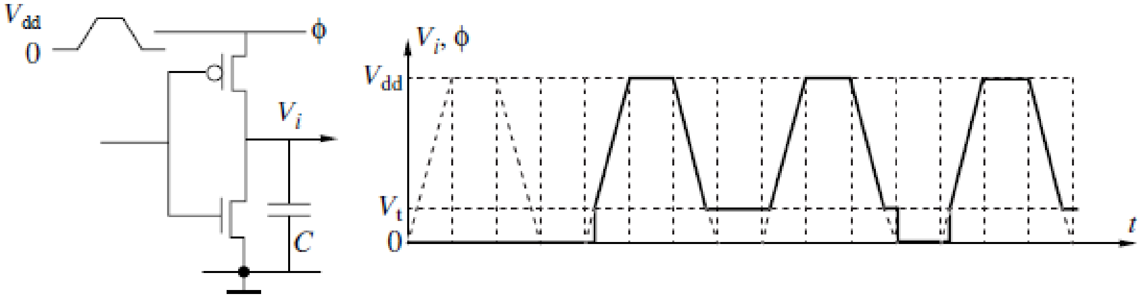
2.3 – Quasi-adiabatic logic.

Quasi-adiabatic systems fail one or more of Starosel'skii's conditions for thermodynamic equilibrium in adiabatic systems. As a result they can't approach zero dissipation but they might still offer some advantages over traditional CMOS operation. This subsection will explore adiabatic implementations that fall under this category.

The 1n1p topology of standard CMOS can also be used to operate in quasi-adiabatic mode by replacing the power clocks with a single-rail version. Figure 2.4, from **[27]** shows this configuration. Notice that the PMOS transistor is forced to pass a voltage lower than its threshold voltage. This causes an abrupt power dissipation while the transistor is in linear mode. As a result, this configuration has a non-zero energy dissipation over a charge and discharge cycle, shown in equation 2.2.

$$E_{dissipated} = CV_t^2 \tag{2.2}$$

As with the 1n1p technology previously described, this adiabatic logic shares the same topology as conventional CMOS and no overhead, but requires complicated nested power clocks and has no pipelining capabilities. The advantage of this configuration over the fully adiabatic mode is that it requires only half the power clocks, however the power dissipation is no longer asymptotically zero.

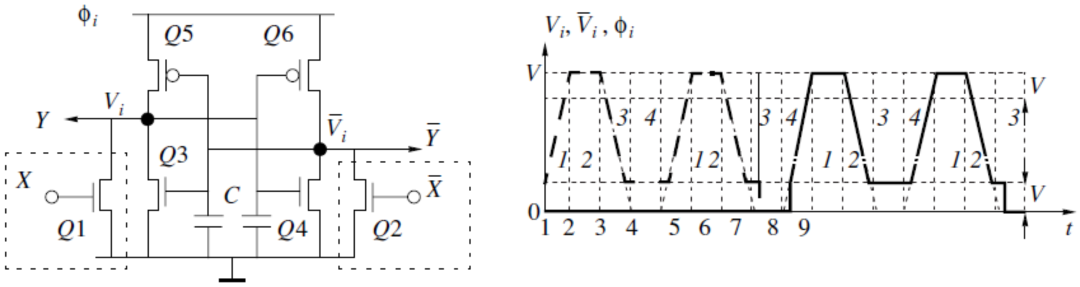


(a) *(b)*

Figure 2.4 – (a) topology of a typical 1n1p quasi-adiabatic logic gate. (b) waveform showing operation of the gate over a single charge-discharge cycle. Solid line is the output, dotted line is the power clock phase. From **[27]**.

A 2n-2n2p configuration is presented in **[26]** and **[27]**. Figure 2.5 shows the topology of this configuration and a plot of its waveform in typical operation. Similar to 1n1p quasi-adiabatic logic, this gate has an abrupt swing in output voltage because it attempts to pass values below the threshold voltage of a PMOS. The dissipated energy for this gate is described by Equation 2.2 too.

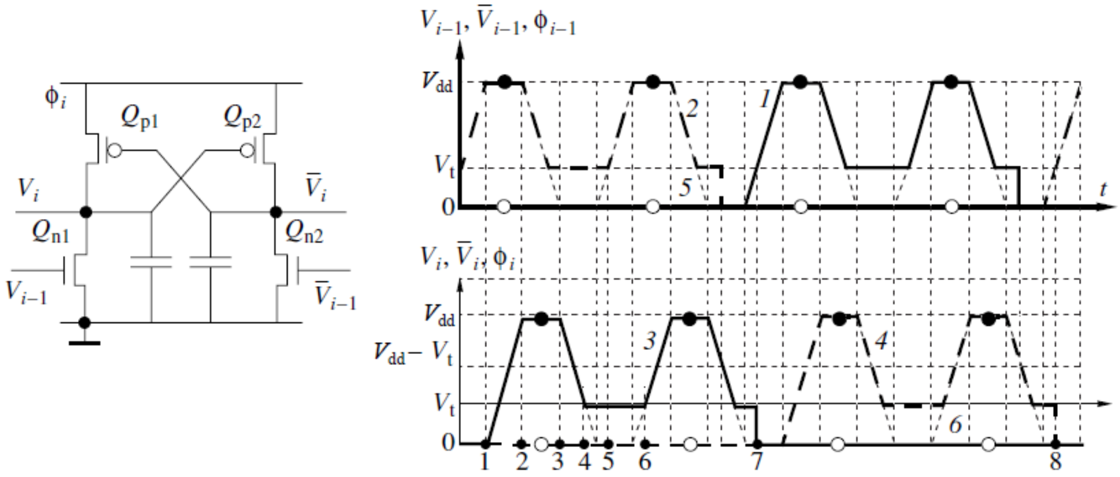
The advantages over 1n1p quasi-adiabatic operation include complementary and direct logic outputs and NMOS tree implementation of logic functions. Disadvantages are a larger overhead for gates with a low number of inputs.



(a) *(b)*

Figure 2.5 – (a) topology of a typical 2n-2n2p quasi-adiabatic inverter. (b) waveform showing operation of the gate over a single charge-discharge cycle. Solid line is the output, dotted line is the power clock phase. Adapted from **[27]**.

Moon and Yeong **[39]** propose a method called Efficient Charge Recovery Logic (ECRL), also known as 2n2p quasi-adiabatic logic. This is a modification of the 2n-2n2p logic shown above. Figure 2.6 shows an inverter gate in ECRL logic along with the waveform of typical operation. Transistors Q3 and Q4 can be removed without affecting overall performance of the logic gate, but they provide grounding for the output nodes during input switching. Therefore ECRL contains some dynamic nodes. Performance of the circuit is similar to 2n-2n2p but the overhead is lower.



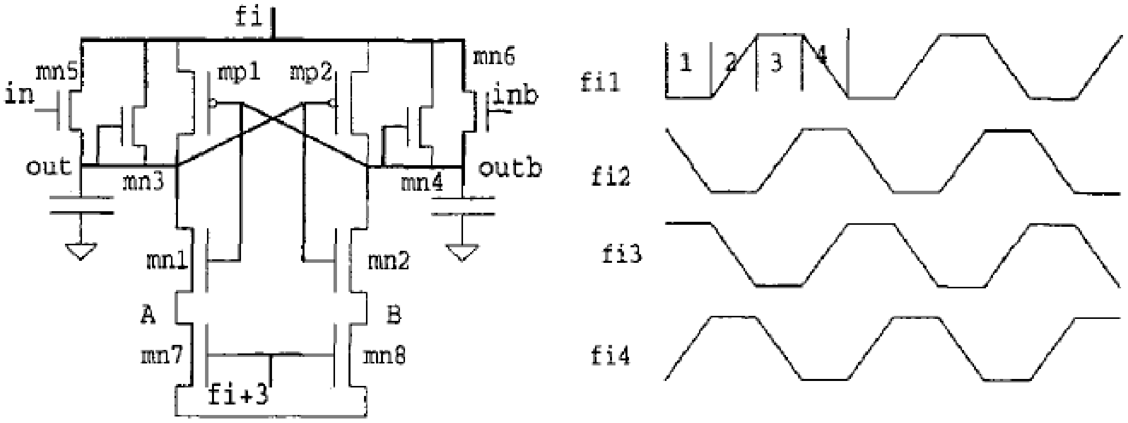
(a) *(b)*

Figure 2.6 – (a) topology of a typical ECRL quasi-adiabatic inverter. (b) waveform showing operation of the gate over a single charge-discharge cycle. The top plot shows waveforms for the direct (solid) and complementary (dotted) input. Bottom plot shows direct (solid) and complementary (dotted) outputs. Notice that the waveforms resemble that of 2n-2n2p logic.

Hongyu et al. **[40]** propose a quasi-adiabatic logic family which they call High Efficient Energy Recovery Logic (HEERL). HEERL builds on the idea of ECRL logic but adds a couple extra transistors to charge the output nodes through transfer gates instead of simple pass transistors. Figure 2.7 shows a schematic of the basic inverter gate and the power clocks needed to energize it.

Assuming input “in” is high and “inb” is low, the power clock will begin charging the “out” node through transistor mn5. When node “out” reaches value V_{tn}, mn3 and mn2 are turned on. This grounds node “outb” through mn2 and mn8, turning mp1 on thus completing a transfer gate with transistors mn3 and mp1 just in time to pass voltages larger than V_{tp}. On discharge the sequence is followed in reverse order but after mn2 is disconnected, the node “outb” is left floating. This is the source of the non-adiabatic dissipation, as the node will drift below 0 V due to parasitic effects. A quantitative analysis shows that the dissipated energy is given by Equation 2.3.

$$E_{dissipation} = \frac{1}{2} C V_{tp}^2 \qquad (2.3)$$



(a) *(b)*

Figure 2.7 – (a) topology of a quasi-adiabatic HEERL inverter. (b) waveforms used to energize the HEERL circuit. Numbers represent the various states: (1) relaxed, (2) charging, (3) energized, (4) recovering. Several phases are needed in a single gate. Adapted from **[40]**.

The main advantage of HEERL circuits over the previously mentioned quasi-adiabatic implementations is the fact that non-adiabatic dissipation is cut by half. It allows NMOS tree logic but has additional overhead even compared to 2n-2n2p configuration. The system is designed with pipelining in mind as it uses power clocks that differ only in their phase instead of nested clocks.

Dickinson and Denker **[41]** have proposed a logic family that uses diodes to direct current flow to and from the power rail. They name this configuration Adiabatic Dyamic Logic (ADL), also described as a 1t-1d (1 transistor 1 diode) system. Each gate uses one diode and a module of tree logic, alternating between NMOS and PMOS in their implementation. The directionality of the diode is also inverted between gates. Figure 2.8 illustrates a chain of inverters using this configuration.

This logic family has an unavoidable abrupt voltage drop equal to the forward bias voltage of the diode used in its implementation and can never approach dissipationless operation. The implementation, however is very simple and requires a single power rail.

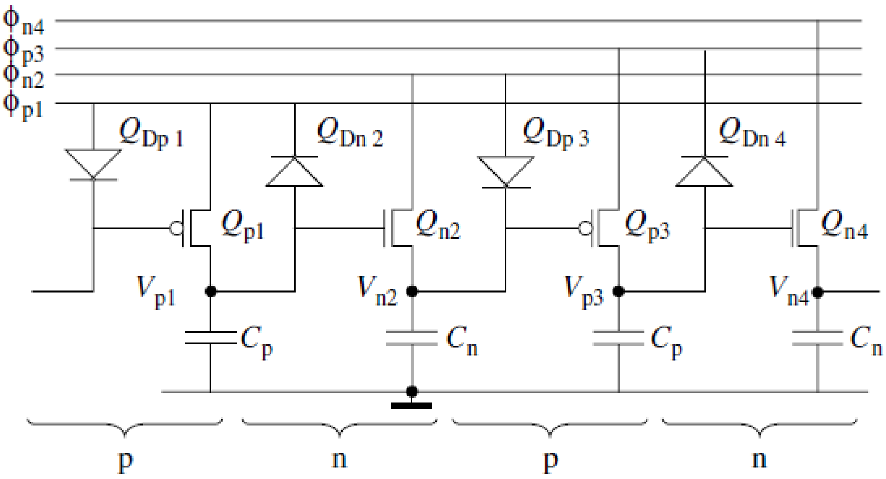


Figure 2.8 – Schematic diagram of an ADL inverter chain. Figure shows 4 inverters in series. From **[27]**

Kramer et al. **[42]** propose a system, which they name 2n-2n2d. Figure 2.9 shows an inverter using this logic. It builds on the previous design by adding a second diode. The goal of this implementation is to present a constant capacitive load to the power supplies, useful for capacitive bank sources that use charge storing to recover energy from the circuit.

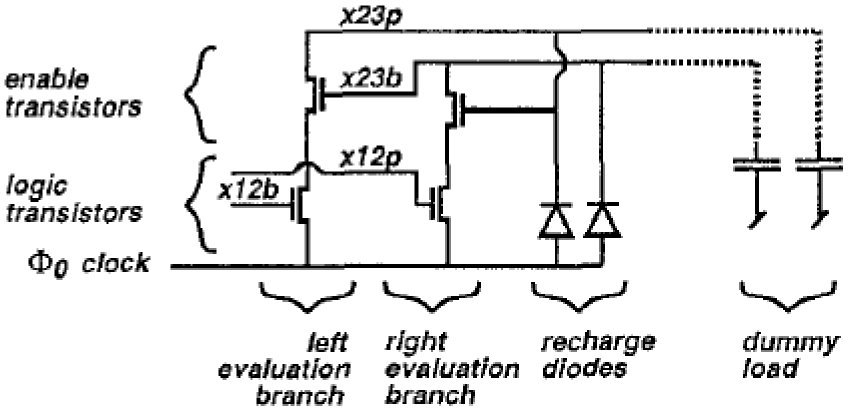


Figure 2.9 – Schematic diagram for a 2n-2n2d inverter. From **[42]**

2.4 – Complex Device Implementations.

Hänninen et al. **[29]** implemented a 4-bit adiabatic multiplier using the split-level Bennett-clocked fully adiabatic technique described in subsection 2.2. The multiplier employs a standard combinatorial structure with no pipelining capabilities. It is able to operate in reversible and irreversible mode by controlling the power-rail inputs. Reversibility is guaranteed by the Bennett clocking power rail scheme. It was laid out manually and fabricated in 2 μm CMOS technology but no experimental test has been performed thus far.

The multiplier was tested in simulation using parameters from the University of Notre Dame 2 μm in-house fabrication process. The simulation was used to verify electrical behavior and compare power dissipation as a function of frequency for both modes of operation. It was found that reversible operation has a dissipation about two orders of magnitude lower than traditional irreversible CMOS for a range of frequencies of up to 30 MHz. By scaling the technology further down it is possible to achieve higher speeds while maintaining this dissipation ratio.

Khazamipour and Radecka **[30]** designed a reversible logic gate capable of performing the AND, OR, NAND and NOR operations. The design is a composite gate that employs three instances of the reversible Toffoli family of logic gates. It employs a SCRL approach to reversibility and allows pipelining.

The composite gates were then used to construct a multi-stage buffer and a three-input AND gate. These devices were tested in simulation and shown to have a dissipation more than 3 orders of magnitude lower than an equivalent traditional CMOS circuit for frequencies below 16 MHz. The circuits were simulated using the Cadence 0.18 μm technology parameters.

Kim et al. **[31]** designed a 4-stage buffer and an 8-bit carry-lookahead adder using multiple adiabatic logic families. The circuits were reproduced using ECRL, 2n-2n2d, and two variations of a custom quasi-adiabatic logic which they name NMOS Energy Recovery Logic (NERL). The designs were laid out by hand using 0.6 μm CMOS technology.

An electrical simulation was performed in order to compare the different quasi-adiabatic methodologies. For frequencies between 1 and 100 MHz the NERL method is shown to dissipate between 2 and 3 times less than ECRL. No comparison to CMOS is made but it is known from the ECRL methodology that it dissipates about 50% of the traditional CMOS power **[31]**. This implies that NERL is able to recover up to 75% of the energy.

Kim et al. **[32]** designed a dynamic multiplier and an Application-Specific Integrated Circuit (ASIC) capable of performing a signal processing algorithm using the ECRL quasi-adiabatic technology. The devices were synthesized and fabricated in a 0.25 μm CMOS technology.

The devices were tested experimentally and the power dissipation was measured. It was found that for an operating frequency of up to 200 MHz, the ECRL dissipate up to 5 times less power as traditional CMOS devices.

Takahashi et al. **[33]** designed an adiabatic 16-bit microprocessor using custom quasi-adiabatic logic. Logic gates employ dual phase sinusoidal power clocks along with a couple of diodes to control current flow direction. An abrupt voltage drop across the diodes limits the system to quasi-adiabatic performance.

The microprocessor was designed using HDL and synthesized using a 0.35 μm library. The extracted netlist was then simulated. Simulation shows a power dissipation of roughly 25% of a traditional CMOS equivalent circuit at an operating frequency of 16 MHz.

Takahashi et al. **[34]** tested a two-stage adiabatic inverter chain using a custom quasi-adiabatic logic that employs diodes to control current flow at the power rails. Simulation showed a power consumption two orders of magnitude lower than that of traditional CMOS operation. The device was also built using discrete components and experimentally tested, but a power dissipation analysis was not performed.

Thomsen **[35]** designed a 4-bit fully adiabatic ALU using SCRL adiabatic logic and Toffoli gates. A custom standard cell library was produced for this purpose. The cells were implemented using 0.35 μm CMOS technology and fabricated. The circuit was tested experimentally for electrical behavior but no analysis of power dissipation was performed.

2.5 – Why use Bennett-clocked adiabatic logic?

As this review has demonstrated, the majority of adiabatic implementations in literature aim to minimize complexity even if that means eschewing some of the power gains only possible in asymptotically zero dissipation logic.

Fully adiabatic systems have higher complexity than quasi-adiabatic implementations. In particular, the single- and split-level Bennett-clocked methods have very strict timing requirements in the power clocks and their generation is a considerable challenge unto itself. SCRL has simpler power clocks but demands use of atypical logic (Toffoli gates) and needs a duplicate of every gate to become reversible. Also, single-rail Bennett-clocked and Toffoli gates have a large transistor count and size. This complexity is demanded by the conditions for adiabatic switching enunciated by **[27]**.

Quasi-adiabatic logic families try to avoid this intricacy by allowing failure of one or more of the conditions. This is usually done by using diodes to direct current flow, letting one or more output nodes float for a brief period of time (thus saving a few transistors) or replacing transfer gate logic with single-transistor pass gates. Any of these modifications comes at the cost of irrecoverable dissipation, seen as abrupt voltage swings in either of the circuit nodes.

For this work, we have selected the split-rail Bennett-clocked family despite the high complexity of the power clocks. Two reasons influenced this decision:

- Fully adiabatic systems are the only ones capable of power dissipation below $k_B T ln(2)$.
- 1n1p topology offers several advantages over other configurations.

Much of the motivation for this work is the experimental confirmation by Snider et al. of LP in practical systems **[4]**, **[23]**. Being able to validate and test dissipationless operation in a larger system would present a strong argument in favor of LP. For this purpose only fully adiabatic implementations will do.

Furthermore, in a practical sense, a fully adiabatic system can operate with arbitrarily low dissipation. Systems that asymptotically approach 100% power recovery are much more attractive than quasi-adiabatic solutions.

Regarding the second motivation, the split-level Bennett-clocked design uses a topology very close to traditional CMOS for every logic gate, in which the only deviation from the norm is the existence of two additional power lines. No additional components such as transistors or diodes are needed. This ensures that the cell complexity and area in layout is minimized.

2.6 – Chapter conclusions.

This chapter contains a detailed review of the available literature regarding reversible computing and adiabatic switching techniques. A practical approach was adopted to answer how to implement a reversible computational system in silicon. The goal of the review was to decide on a particular implementation that best suits the objectives of this work.

The first section of the chapter looks at the practical framework of reversible computing and adiabatic switching. Key concepts are defined and an equation for power dissipation in fully adiabatic systems is discussed. The conditions for practical adiabatic systems are also enunciated and explained in detail.

The second section focuses on the adiabatic techniques that comply with every condition and thus are able to perform asymptotically zero dissipation operation. These systems are named fully adiabatic. They consist of two Bennett-clocked schemes: single- and split-rail plus a SCRL system based on a family of reversible gates. The main advantages and disadvantages for each logic family are discussed and an example schematic for each is included.

The third section of the chapter describes adiabatic implementations that fail one or more of the conditions for dissipationless operation. These are known collectively as quasi-adiabatic systems. Their aim is to reduce complexity at the cost of having irreversible power losses. For each technology a description is provided, along with an example circuit. Since this terrain is mostly unexplored, the list does not aim to be exhaustive, but representative of the main ideas in adiabatic system design. Most papers in the subject use custom logic to best suit their goals.

The fourth section shows a sample of practical adiabatic systems found in literature. The examples in this section are complex devices that contain at least a few logic gates. The majority of systems found in literature employ quasi-adiabatic techniques. Out of the ones that are fully adiabatic, two employ a SCRL Toffoli gate scheme and only one uses Bennett clocking, a 4-bit multiplier.

The fifth section uses the information gathered throughout the chapter to justify the decision to use split-level Bennett clocking. The two cited reasons are its capability to perform dissipationless operation and the simple 1n1p topology common to standard CMOS.

Chapter 3 – A Bennett-Clocked Adiabatic Implementation.

This chapter covers particularities of the split-rail Bennett-clocked implementation selected for this project. The first section covers preliminary work done as “proof of concept” to explore the energy saving capability of adiabatic switching. A simulation was prepared based on the experiment performed by Snider et al. **[4]**. The simulation aims to replicate conditions of the experiment but substitutes some parts that were done by hand for automatic digital circuitry. A discussion of the results is also presented in this section.

The second section of the chapter is a description of the HDL tools employed in design and modeling of the adiabatic ALU and MIPS microprocessor. The model itself is also discussed in detail. A behavioral simulation is done to verify the intended operation of the devices. Time diagrams from this simulation are included for illustrative purposes.

The third section of the chapter discusses some of the challenges revealed during logic design and behavioral modelling. Finally it summarizes the information contained throughout to consolidate the chapter.

3.1 – Proof of concept.

Snider et al. **[4]** propose an experimental test of LP. Figure 3.1 shows the setup of this proof and experimental data gathered from it. The test consists in an RC circuit that can be connected to 4 different waveforms depending on a selector switch. The capacitive load is charged or discharged across a resistor according to the voltage difference between its terminals.

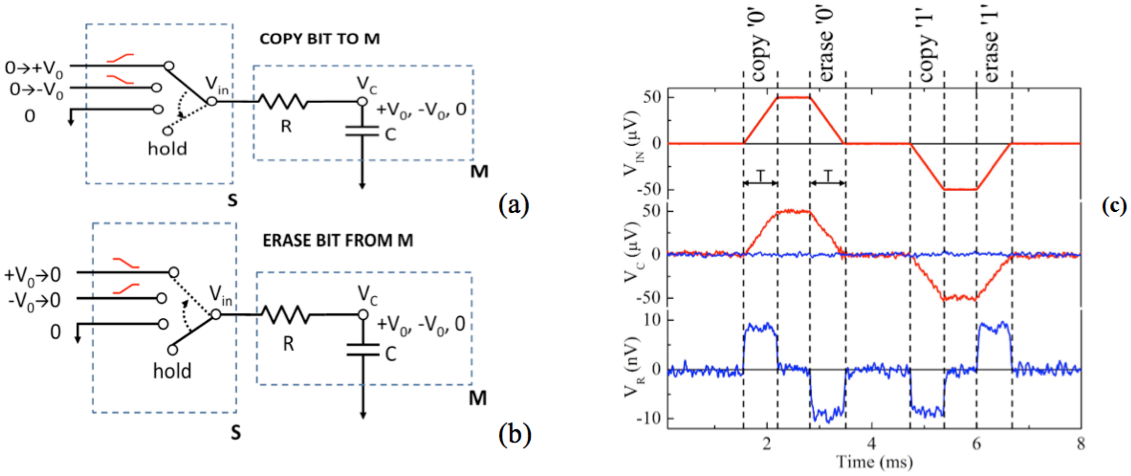


Figure 3.1 – Experimental test of the Landauer Principle. (a) schematic of a bit copy. (b) schematic of a bit erase. (c) waveforms obtained in experiment. Adapted from **[4]**.

At the beginning of a cycle (see Figure 3.1 (a)), both power clocks start from the relaxed state and ramp up to their corresponding logic values, 0 or 1. The ramp has to be much slower than the RC time constant for adiabatic switching to occur. If this condition is true, the capacitor will charge along a series of quasistatic transitions and no energy will be dissipated.

When the capacitor is charged, the RC can be safely switched to a “hold” state. This means that the RC network is disconnected from a driver and thus becomes a dynamic node and will begin discharging along parasitic paths to ground. For this experiment we assume that all devices are ideal and no such discharging takes place.

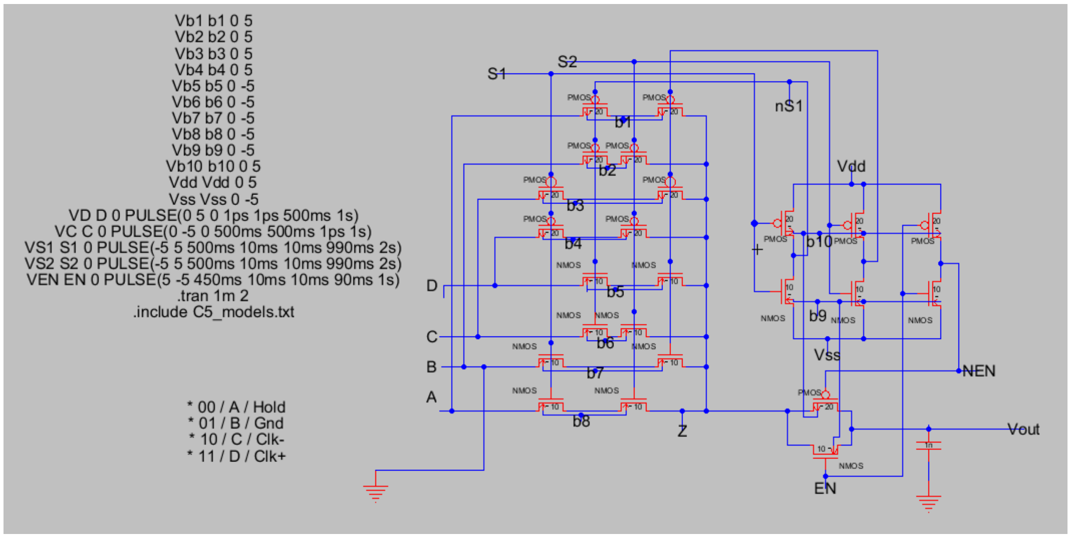
From this hold state three possibilities exist: connecting the RC network to ground, to the positive power clock or to the negative power clock. Connecting the RC circuit to ground dissipates all of the charge stored in the capacitor, which amounts to half of the rail-to-rail voltage multiplied by the capacitive value. By connecting it to the correct power clock, the capacitor will discharge adiabatically (see Figure 3.1 (b)) and no dissipation takes place. By choosing the wrong power clock, an abrupt voltage jump equal to the rail-to-rail potential occurs and all of the bit energy is dissipated.

Choosing the correct connection implies that the bit information is preserved somewhere else, because if the bit was destroyed it would be impossible to choose the correct power clock other than guessing. Half of the time a guess will be wrong on average. Since this carries a penalty of full bit energy dissipation it is clearly undesirable.

The experiment becomes a practical proof of LP if it can be shown that asymptotically dissipationless transfer of charge is possible when choosing the correct connection. Such a result would confirm that preservation of information allows circumvention of USL ($K_B Tln(2)$) dissipation.

The results from the physical realization of this experiment in **[4]** indeed confirm that dissipation lower than USL is possible. In order to better understand the phenomenon, it was decided to perform a simulation of the same experiment as preliminary work for this research.

For the simulation circuit, a 4-to-1 multiplexer based on transfer gates was employed as substitute for the 4-way selector. Transfer gates were chosen because they are able to pass both positive and negative power clocks without dissipation. The input signals were programmed using SPICE code and the simulation was done in LTspice. Figure 3.2 shows the schematic diagram for the experiment setup. A transfer gate isolates the output node of the multiplexer from the RC network. This was necessary because the multiplexer has transient glitches when switching the selector bits. The capacitive load was set at 1 nF.

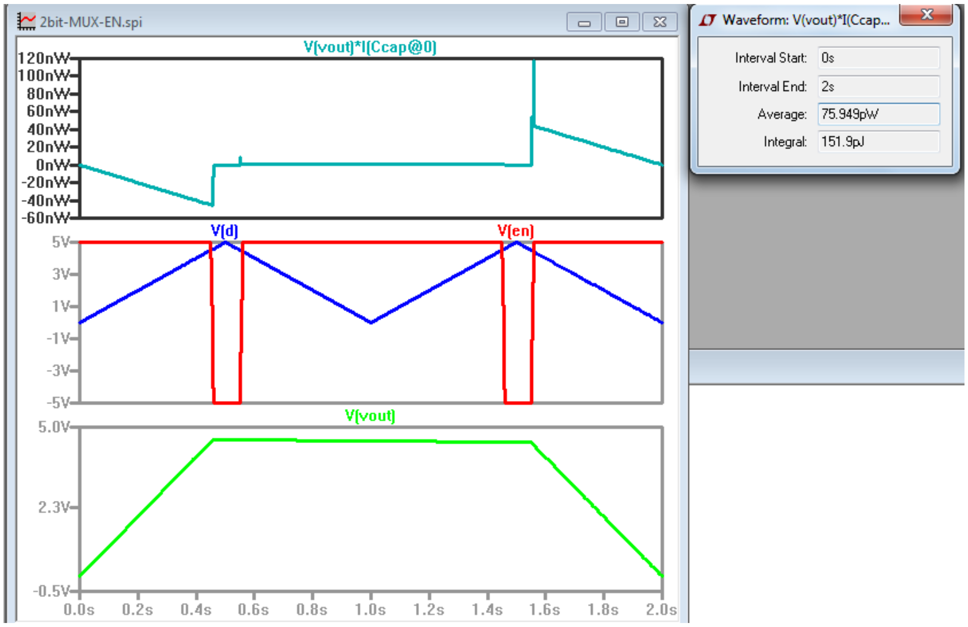


```
Vb1 b1 0 5
Vb2 b2 0 5
Vb3 b3 0 5
Vb4 b4 0 5
Vb5 b5 0 -5
Vb6 b6 0 -5
Vb7 b7 0 -5
Vb8 b8 0 -5
Vb9 b9 0 -5
Vb10 b10 0 5
Vdd Vdd 0 5
Vss Vss 0 -5
VD D 0 PULSE(0 5 0 1ps 1ps 500ms 1s)
VC C 0 PULSE(0 -5 0 500ms 500ms 1ps 1s)
VS1 S1 0 PULSE(-5 5 500ms 10ms 10ms 990ms 2s)
VS2 S2 0 PULSE(-5 5 500ms 10ms 10ms 990ms 2s)
VEN EN 0 PULSE(5 -5 450ms 10ms 10ms 90ms 1s)
.tran 1m 2
.include C5_models.txt

* 00 / A / Hold
* 01 / B / Gnd
* 10 / C / Clk-
* 11 / D / Clk+
```

Figure 3.2 – Schematic diagram of the proof of concept test.

Figure 3.3 shows the simulated output waveform along with a measurement of the dissipated power. Power dissipation was calculated by multiplying the capacitor voltage by its current. This was done to exclude the power consumed by the multiplexer transistors from the measurement. The top waveform is the instantaneous power dissipated at the capacitor node.



Interval Start: 0s
Interval End: 2s
Average: 75.949pW
Integral: 151.9pJ

Figure 3.3 – SPICE simulation of the proof of concept test using a ramping power clock. Energy dissipated over a full charge-discharge cycle is shown to be 151.9 pJ for a load capacitance of 1 nF.

The energy dissipated over a charge and discharge cycle in traditional CMOS is given by equation 3.1:

$$E_{dissipated} = CV_{DD}^2 \tag{3.1}$$

For a bias voltage of 5 V and a capacitive load of 1 nF as used in this experiment, the dissipated energy in abrupt discharging is 12.5 nJ. The adiabatic switching scheme however produces a dissipation of 151.9 pJ, which is 82 times smaller. This is of course influenced by the rising and falling times but it is clear already that adiabatic switching offers considerable advantages over traditional switching methods.

3.2 – System Verilog HDL Model.

A custom behavioral model of Bennett-clocked logic was created using System Verilog HDL. A toolset containing scripts necessary to model adiabatic logic behaviorally was put together. Design of an adiabatic 8-bit ALU and MIPS microprocessor were also created using these tools. Simulation confirms that both the tools and designs work as intended. The implementation of the custom tools and behavioral models was led by Dr. Ismo Hänninen from University of Notre Dame.

Verilog supports logic primitives as well as behavioral descriptions of custom gates and is widely used in the industry to describe, design, simulate and synthesize VLSI designs. Since adiabatic switching circuits are currently an emergent field in electronics, no HDL natively supports any kind of adiabatic logic. However, System Verilog is robust and customizable enough to permit their description by other means.

The custom System Verilog tools contain definitions for logic states not present in traditional CMOS. By default Verilog accepts and interprets the states logic high (1), logic low (0), indefinite (X) and high impedance (Z). Bennett-clocked adiabatic logic has additional states relaxed, charging towards high, discharging from high, charging towards low and discharging from low used to describe the various sections of a power clock waveform and by extension the possible waveforms of the output nodes. Table 3.1 lists all logic states present in adiabatic logic. A script was written to allow Verilog to process and model these states. The System Verilog script also generates the corresponding standard CMOS model for comparison while simulating adiabatic operation.

Table 3.1 – List of valid logic states in Bennett-clocked adiabatic and CMOS logic.

State	***Present in Bennett-clocked adiabatic***	***Present in standard CMOS***
Logic High	***Yes***	***Yes***
Logic Low	***Yes***	***Yes***
Indefinite	***Yes***	***Yes***
High Impedance	***Yes***	***Yes***
Relaxed	***Yes***	***No***
Charging towards High	***Yes***	***No***
Discharging from High	***Yes***	***No***
Charging towards Low	***Yes***	***No***
Discharging from Low	***Yes***	***No***

Since Verilog primitives deal only with the standard CMOS logic states, a new library was written with a behavioral description for each cell needed in the design. These include combinatorial elements such as logic gates and transfer gate-based designs for control of data flow. Although the adiabatic ALU and MIPS microprocessor use sequential elements to store data (for example in registers), it was decided that these be the standard static elements included in the Verilog primitives. In a real implementation they also are standard CMOS elements thus no redefinition is needed for them. This library was later used as basis for a standard cell library, which is described in Chapter 4.

The logic design was done at the gate level using the custom adiabatic library described above. The design of the ALU is customized and uses a carry-lookahead adder to improve latency, a vital parameter when working with Bennett-clocked adiabatic designs. A higher latency demands larger amount of power clock phases. The MIPS processor design is based on the architecture described in **[43]**. The gate-level implementation of each module, however, is an original design because the architecture itself does not demand such specific requirements. Since many choices were made during the layout creation process, a detailed description of the designs will be placed in Chapters 5 for the ALU and 6 for the MIPS microprocessor, during discussion of the layout prepared for fabrication.

For the behavioral simulation of the MIPS microprocessor a testbench was prepared. The testbench has software descriptions of the signals needed for proper operation such as the power clocks. A test program suggested by **[43]** is integrated into the testbench to test all functionalities of the microprocessor.

A simulation of the adiabatic devices was performed using the hardware description in System Verilog. The software used for simulation is Mentor Graphics ModelSim, although any Verilog simulator should able to produce the correct waveforms. Figure 3.4 shows part of a transient test run of the MIPS microprocessor.

In the simulation pictured, the microprocessor is trying to fetch a new instruction from the program memory. As per the MIPS architecture specifications, this operation takes 4 clock cycles. In Bennett-clocked adiabatic systems a cycle is defined as the period between two fully relaxed states. In the first half of the cycle all of the power clocks ramp up to their desired values and the output of the system is only valid when all of them are energized. In the second half of the cycle, all the power clocks ramp down in reverse order to allow reversibility of the computation. The power clocks are depicted in the lower 12 waveforms of the picture, labeled "*powerClksP*" from 11 to 0.

The top waveform, labeled "*instr_std*" shows the instruction register contents. It begins with the 32-bit value 0x00000000 and loads the instruction 0x80020044 at a rate of one byte per cycle starting from the Least Significant Byte (LSB). The row below this one, labeled "*pc_std*" shows the Program Counter (PC) contents. It starts with value 0x00 and counts up with every cycle during the fetch operation. Notice that it remains at 0x00 for two cycles at the very start. This is because the microprocessor has just started operation from a reset or power on and not all of the control signals are ready yet.

The task of calculating the next PC value falls upon the ALU, which operates in adder mode for this purpose. The picture shows the various ALU control signals plus the result in the row named "*aluresult_std*". This result remains valid only for a brief period of time before returning to undetermined status. The signal labeled "*aluresult_reg*" is a static register that samples this value and updates the PC.

3.3 – Chapter Conclusion.

From the HDL design and behavioral simulation many of the characteristics unique to Bennett-clocked adiabatic circuits are already apparent. The principal disadvantage of the Bennett-clocked logic family is that each power clock needs a separate version with inverse polarity, doubling the number of clock phases. Additionally, it demands a very strict timing for the input and output signals of each module. As a result, it can be concluded that Bennett-clocked systems are heavily constrained by their timing. This has an impact on the logic design of the system, because every signal has a hard deadline and it's also desirable to perform tasks with the lowest possible latency.

A useful term when describing combinatorial data paths is that of logic depth. Logical depth is a term defined by Bennett as the time a standard Turing machine takes to complete a given task **[44]**. In this context, logical depth is easily observable because a Bennett-clocked logic gate has a depth of 1 if its operation is performed using a single power clock phase.

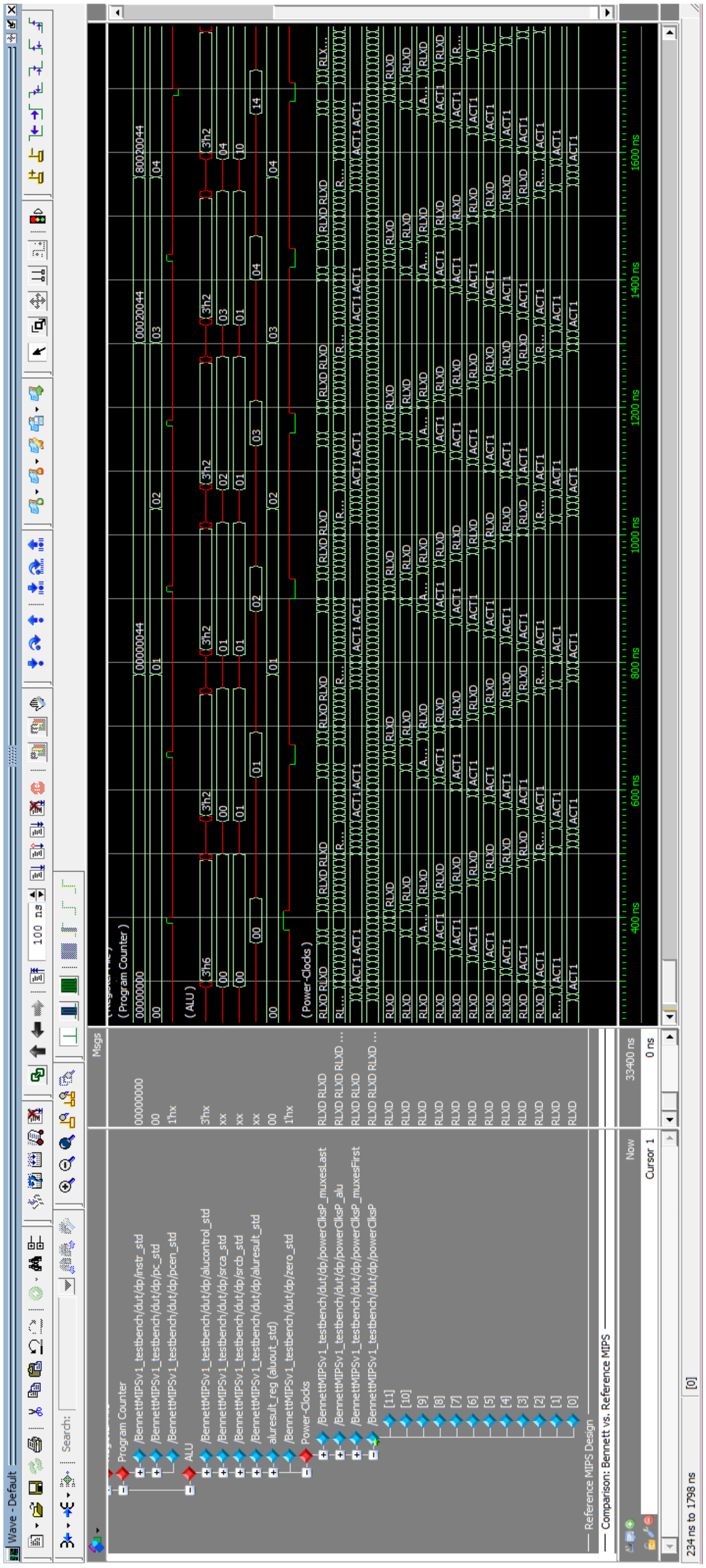

Wave - Default
Search:
Program Counter
/BennettMIPSv1_testbench/dut/dp/instr_std
/BennettMIPSv1_testbench/dut/dp/pc_std
/BennettMIPSv1_testbench/dut/dp/pcen_std
ALU
/BennettMIPSv1_testbench/dut/dp/alucontrol_std
/BennettMIPSv1_testbench/dut/dp/srca_std
/BennettMIPSv1_testbench/dut/dp/srcb_std
/BennettMIPSv1_testbench/dut/dp/aluresult_std
aluresult_reg (aluout_std)
/BennettMIPSv1_testbench/dut/dp/zero_std
Power-Clocks
/BennettMIPSv1_testbench/dut/dp/powerClksP_muxesLast
/BennettMIPSv1_testbench/dut/dp/powerClksP_alu
/BennettMIPSv1_testbench/dut/dp/powerClksP_muxesFirst
/BennettMIPSv1_testbench/dut/dp/powerClksP
Reference MIPS Design
Comparison: Bennett vs. Reference MIPS
Now
33400 ns
Cursor 1
0 ns
Msgs
400 ns
600 ns
800 ns
1000 ns
1200 ns
1400 ns
1600 ns
234 ns to 1798 ns

Figure 3.4 – Part of a transient test run of the MIPS microprocessor.

The overall goal for a Bennett-clocked logic design is to have the smallest possible total logical depth. This minimizes the number of power clock phases necessary and reduces complexity of the system. In the ALU design for instance it was deemed preferable to have a NAND gate with a large number of inputs (high RC time constant) as opposed to a NAND tree with logical depth of at least two. These choices are described in detail in the following chapters of this work.

In closing, this chapter took a look at some of the particularities of Bennett-clocked adiabatic systems. The first section focused on a proof of concept circuit. The experiment itself was described as performed by **[4]** and then a modification for simulation was presented. It was found in this simulation that adiabatic switching of nodes must occur only through direct connections to the power rails, pass transistors of the appropriate type (PMOS for the positive clocks and NMOS for the negative clocks) or transfer gates. It was also confirmed that adiabatic switching causes much lower dissipation than predicted by traditional switching equations.

The second section of the chapter presented HDL design and verification tools along. These tools were developed along with the design of the adiabatic ALU and MIPS microprocessor. A brief description of the custom definitions and the design process is provided. A testbench was prepared for validation of the MIPS design and a simulation run is shown and explained. It is confirmed that the custom logic model is able to produce a working simulation. The design described in this chapter was used as basis for the layout described in more detail in Chapters 4, 5 and 6.

Chapter 4. Adiabatic Bennett-Clocked Standard Cell Library.

This chapter will present the adiabatic standard cell library created for the 8-bit adiabatic Bennett-clocked ALU. The designed cells include logic gates, multiplexers based on transmission gates (TGs) and a conditional inverter based on TGs.

We begin with a discussion of general design choices valid for every cell as well as justifications for each of these choices. In the second section we describe each cell in detail, along with an electric diagram and the layout proposed for fabrication. Some of the cells include a transient simulation to show typical behavior of the circuit.

The third section establishes a comparison between standard CMOS and Bennett-clocked adiabatic operation in terms of power dissipation as a function of frequency for the inverter gate. The comparison employs data from simulation and theoretical calculations. Discussion shows that as frequency increases, the adiabatic circuit is less advantageous than the standard CMOS device.

4.1 - General design choices.

We decided to use the standard cell technique to streamline the design of our ALU. This tool helps lay complex circuits in an orderly fashion and lends itself well to Bennett-clocked design, as each row can contain a single logic level and thus, single power clock phase.

Each cell complies with design rules of ON Semiconductor's C5 process (500 nm), as recommended by MOSIS. In addition to this foundry, the design will be fabricated in Notre Dame's own IC Fabrication Laboratory using a custom process with no additional design rules and a minimum feature size as low as 1 µm. The same design will be used for both processes. The layout will use two layers of metal and a single layer of polysilicon.

Cells have a height of 75λ between the middle of the positive and negative power clock rails. In addition, there are rails for VDD and VSS above and below the cell, respectively. These rails run horizontally through the circuit as cells are designed to be laid out side by side in rows. Each row shares a single power clock phase and needs to have the same logic depth in logic design. Power rails are laid out using the Metal 2 layer.

Data flows from the top to the bottom of the circuit. The only exception to this rule occurs in non-buffered multiplexers, discussed in more detail in section 4.2.6. Because of this directionality, we decided to have inputs on top of each cell and outputs at the bottom. Input and output lines run vertically out of the cell using the Metal 1 layer.

The inner components of each cell are connected using Metal 1 and Polysilicon exclusively. This allows the use of Metal 2 lines as overlapping interconnections between cells. Additionally, we decided to leave space for two lines of dedicated interconnection between standard cell rows. This value was set constant because some cells have a logic depth larger than 1 and need to be several rows tall.

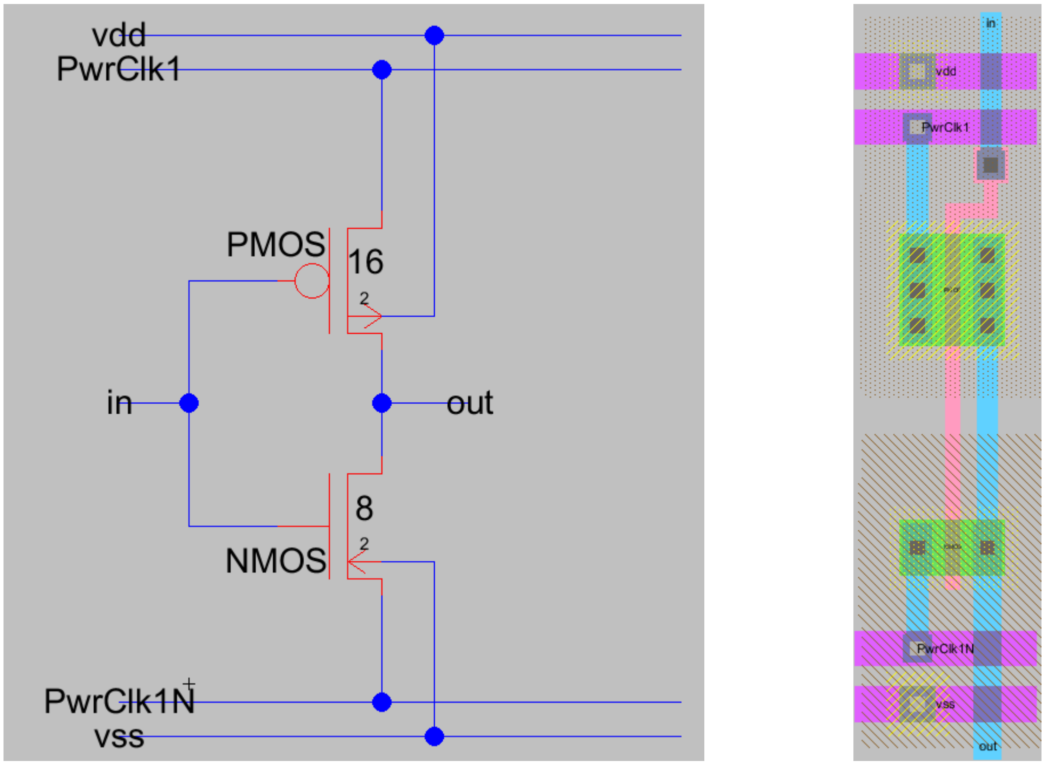
We used Electric VLSI to draw the electric diagrams for each cell and LTspice to simulate their behavior. Layouts were drawn in Tanner Tools L-Edit for fabrication in the Notre Dame process and ported over to Electric VLSI for the MOSIS CMOS 500nm process. The figures illustrated in this chapter are from the Electric VLSI version and are identical to their Tanner Tools counterparts. The layout work was done collaboratively with Tecnológico de Monterrey student René Celis.

4.2 - Standard Cells

The adiabatic standard cell library for the ALU includes two different size inverters, 2-input to 8- input NAND gates, 2-input NOR, 2-input AND, 2-input OR, 2-input XOR, TG-based multiplexers and a TG-based conditional inverter.

4.2.1 - Inverters

The adiabatic inverter gates follow the traditional CMOS topology as shown in the figure 4.1 (a). A PMOS transistor forms a path between the output node and the positive power supply while a NMOS transistor does the same for the negative power supply. Both transistors share the gate input. The only difference for the Bennett-clocked adiabatic version is that the power supplies are Bennett power clocks instead of fixed DC values. VDD and VSS appear as voltage references for the substrate. The Figure 4.1 (b) shows the corresponding layout.



(a) (b)

Figure 4.1 – Adiabatic Inverter gate. (a) Schematic. (b) Layout.

Figure 4.2 shows the behavior of the inverter running in adiabatic mode over two cycles at 100 kHz. The top plot (green) is the input voltage which shows a transition from logic low (-2.5 V) to logic high (+2.5 V) at the 10 µs mark. The second plot (blue) is the output voltage of the circuit. It follows the power clock opposite to the value of the input, charges adiabatically then discharges back to the relaxed state.

In the previous plot, there's a glitch near the switching point; however this glitch disappears when a larger loading capacitance is introduced. Figure 4.2.1.3 shows the same simulation extracted from the layout with parasitic elements. Magnitude of the glitch is reduced because the inverter is loading a larger capacitance (C_{self}) composed by the sum of parasitic capacitances (drain-bulk capacitances, C_{db} , for both the NMOS and PMOS devices).

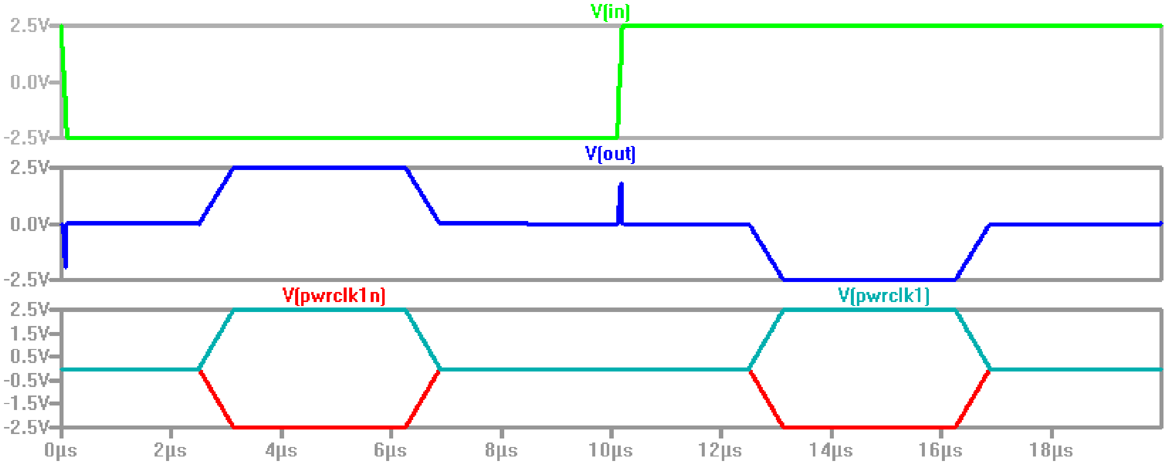


Figure 4.2 – Transient simulation of the inverter in adiabatic mode over two cycles. Operation frequency is 100 kHz. Simulation from schematic.

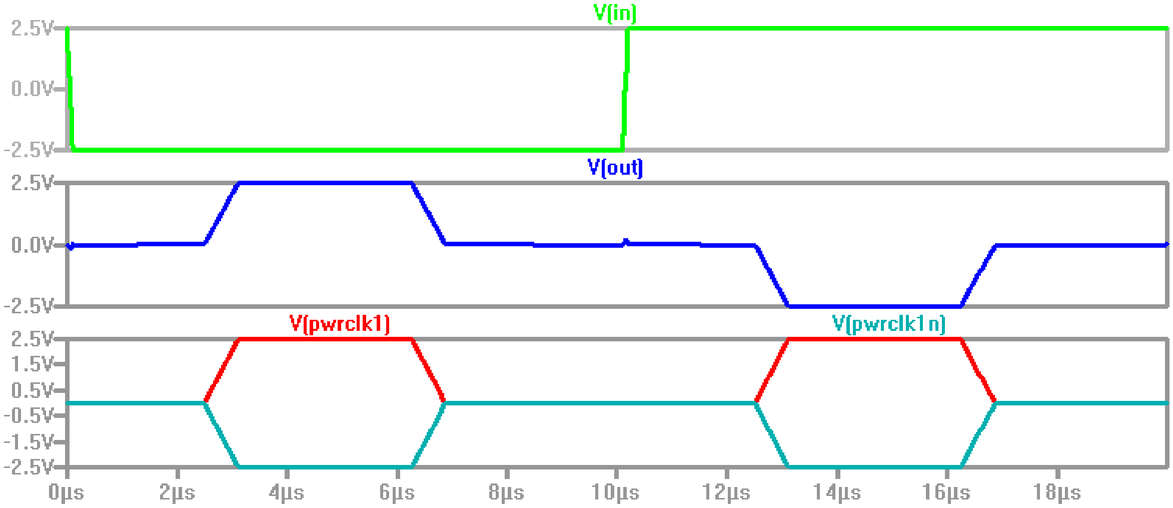
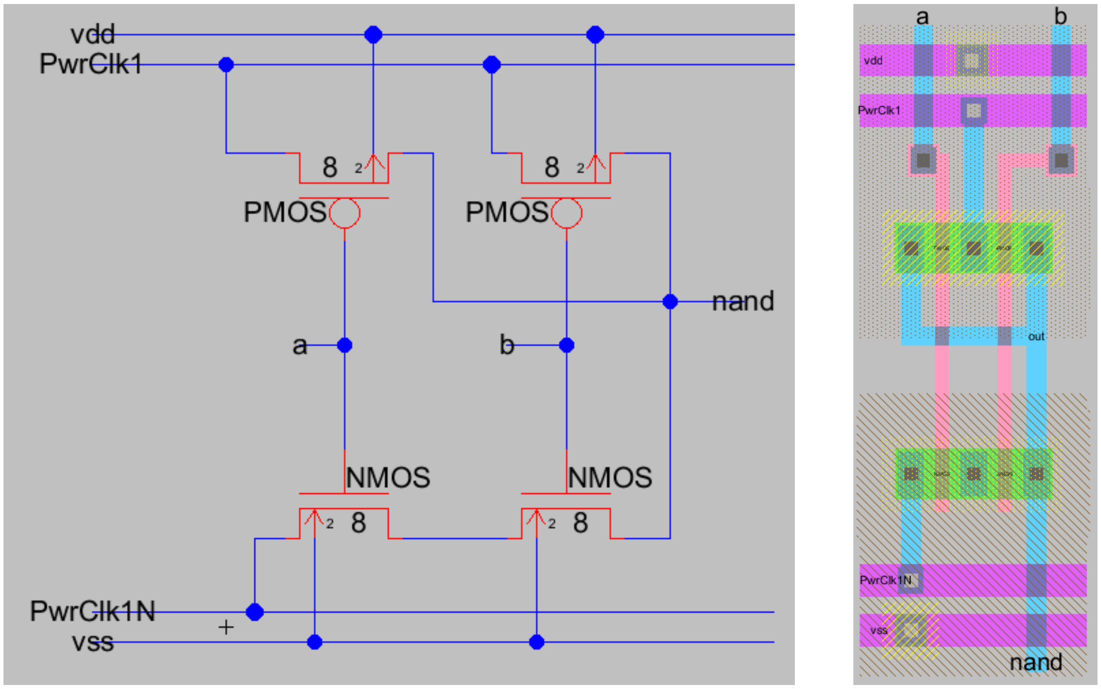


Figure 4.3 – Transient simulation of the adiabatic inverter over two cycles at 100 kHz. Simulation from layout with parasitics. Glitch is reduced due to higher load capacitance C_{self}.

The smallest inverter in the library has a width of 8λ for the NMOS and 16λ for the PMOS. There's also a larger inverter with 1.5 times as much driving capacity, with a NMOS width of 12λ and PMOS width of 24λ. This inverter performs similarly as it follows the exact same topology and proportions. All transistor lengths are 2λ, where λ is one half of the minimum channel length of the device.

4.2.2 – NAND gates

The library includes several NAND gate cells from 2 to 8 inputs. Topology of these cells is similar to the standard CMOS versions, with the same differences described in the Inverter section, that is, power supplies are power clocks instead of static DC values. Transistor sizes were adjusted to match proportions of an inverter with NMOS size 4λ and PMOS size 8λ where possible. However, for the 8-input NAND cell predicted NMOS size was too large to fit the allotted space and a slightly smaller NMOS transistor was used in its place (28λ instead of 32λ). Figure 4.4 shows the schematic and layout for the 2-input NAND gate.



(a) (b)

Figure 4.4 – 2-input NAND adiabatic gate. (a) Schematic. (b) Layout.

Figure 4.5 shows a transient simulation of the 2-input NAND gate running at 100 kHz. Four cycles were simulated to show every possible input state. The first and second plots (green and blue respectively) are the input bits, while the third plot (red) is the output voltage. Output follows the power clock corresponding to the truth table of the NAND gate. In this case, for instance, the output follows the negative power clock (logic low) only when both inputs are high.

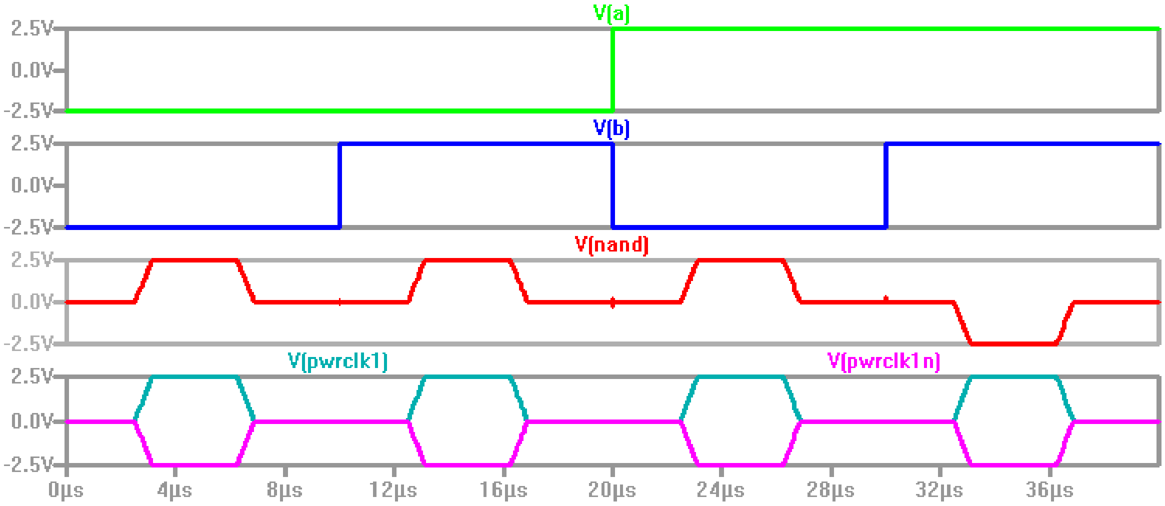
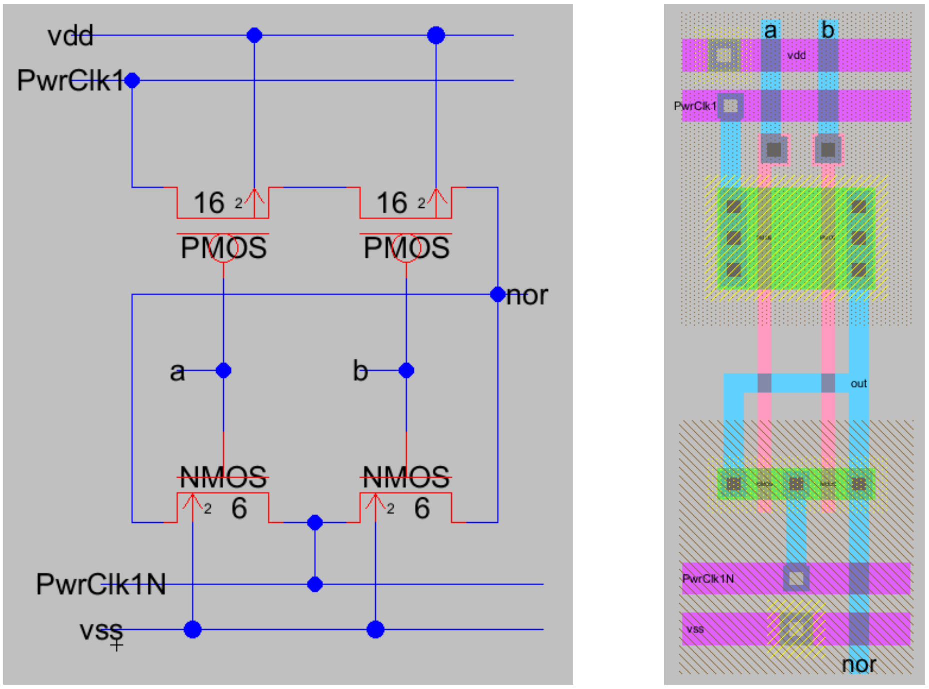


Figure 4.5 – Transient simulation of the adiabatic 2-input NAND gate over four cycles. Operation frequency is 100 kHz. Simulation from layout with extracted parasitics.

The rest of the NAND gates are shown in Appendix B including electric diagram, layout and transient simulation examples. They exhibit a similar behavior to the 2-input NAND gate as they follow the same topology and maintain comparable proportions.

4.2.3 – NOR gate.

The adiabatic ALU uses a 2-input NOR gate in its implementation. The topology of this gate is similar to the standard CMOS version but the power supplies are power clocks instead of DC voltage sources. Transistor sizes were chosen to match the proportions of an inverter with NMOS size 4λ and PMOS size 8λ, however a 4λ NMOS is too small for the SPICE model to work correctly in simulation. Because of this, the size of the NMOS was pushed up to 6λ while the PMOS remained at 16λ. Figure 4.6 shows both schematic and layout for the 2-input NAND gate.



(a) (b)

Figure 4.6 – 2-input NOR adiabatic gate. (a) Schematic. (b) Layout)

A transient simulation of the 2-input NOR gate was created, with an operating frequency of 100 kHz. Four cycles were simulated to see all possible input combinations. Figure 4.7 shows the plotted simulation results. The two topmost plots (green and blue) are the inputs while the third plot (red) is the output voltage. As can be seen on the simulation, the output follows the power clock as dictated by the input values and the NOR truth table. For this case, the output shows a logic high only when both inputs are low, which is consistent with the function of the NOR operation.

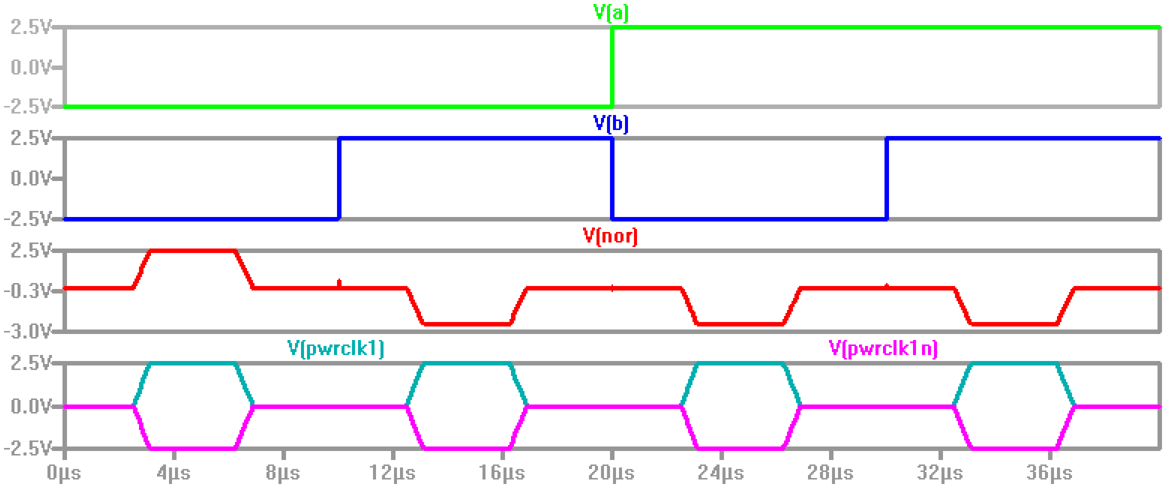
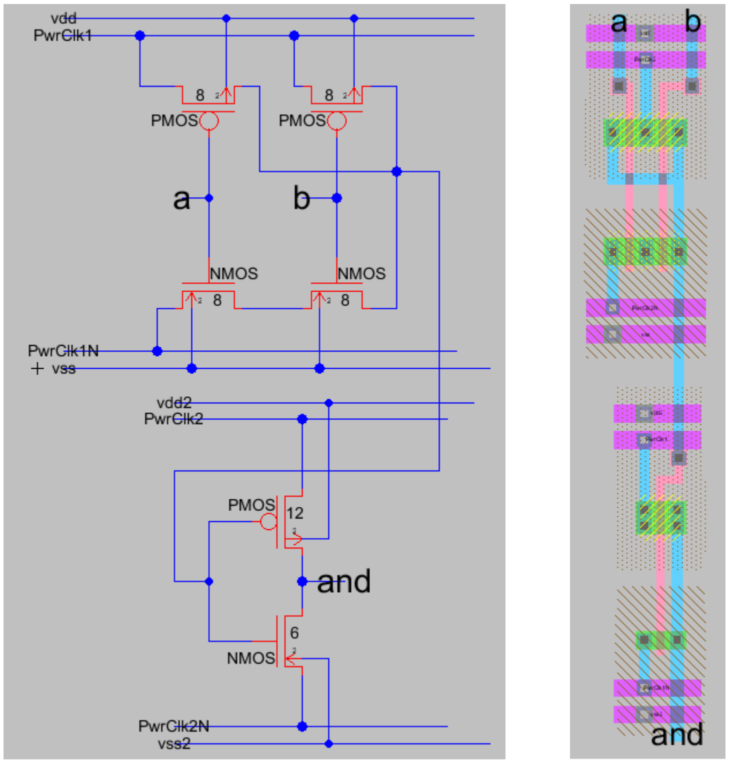


Figure 4.7 – Transient simulation of the adiabatic 2-input NOR gate over four cycles. Operation frequency is 100 kHz. Simulation from layout with extracted parasitics.

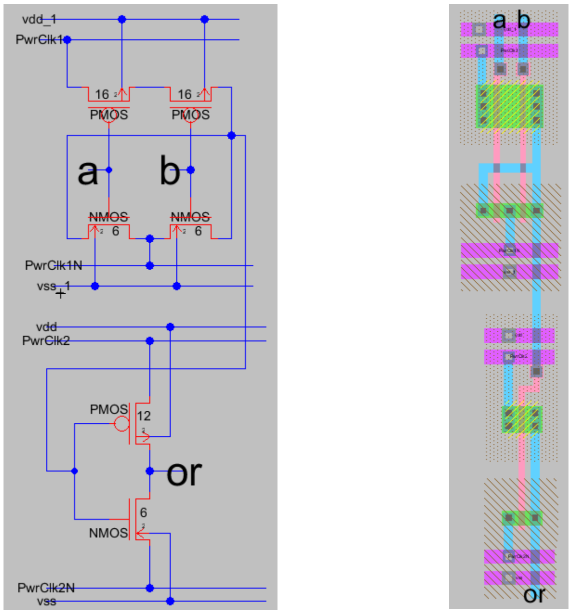
4.2.4 – Composite gates.

Two composite gates are available in the library: AND and OR. In both traditional CMOS and adiabatic Bennett-clocked logic these cells are composed of the base gate in series with an inverter. Because of this, the adiabatic versions are two rows high and have two sets of power rails. Figure 4.8 shows the schematic and layout of the AND gate, and Figure 4.9 shows the OR gate. Transient simulations for these cells can be found in Appendix B.



(a) (b)

Figure 4.8 – Design of the adiabatic composite 2-input AND gate. (a) Schematic (b) Layout.

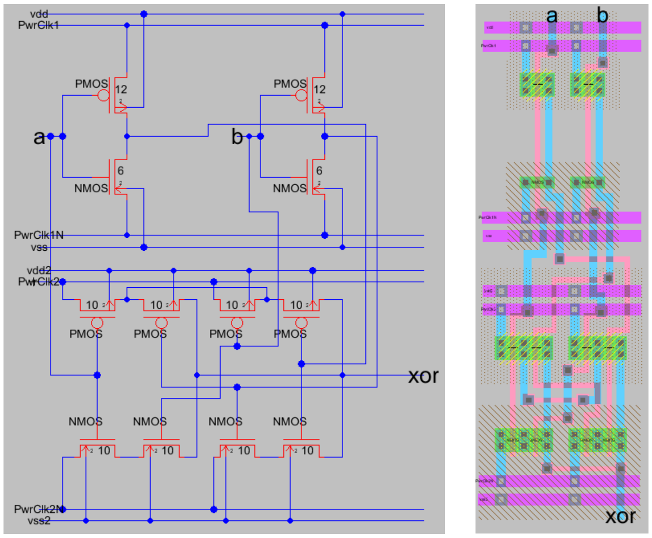


(a) (b)

Figure 4.9 – Design of the adiabatic composite 2-input OR gate. (a) Schematic (b) Layout.

4.2.5 – XOR gate

The XOR gate follows a complex CMOS gate design which requires inputs and their complements. The circuit is also a composite gate, with two inverters for the inputs and the complex CMOS gate in series. The schematic and layout for the XOR gate are shown in Figure 4.10 The first logic level contains only two inverters while the lower level has transistor logic to perform the logic function XOR.



(a) (b)

Figure 4.10 – 2-input XOR adiabatic gate (a) Schematic. (b) Layout.

Figure 4.11 shows a transient simulation of the XOR gate running at 100 kHz. The simulation was created from the layout using parasitic extraction. The top two waveforms are the inputs (green and blue) and the output voltage is plotted third from top to bottom (red). As can be seen, the output follows the truth table for the XOR function. Output is high only when one of the inputs is high and the other is low. Because of clock feedthrough effects between phases, the output waveform has a bit of noise. This effect can be mitigated by connecting the cell to a higher load, for instance when connected to other gates in the ALU device.

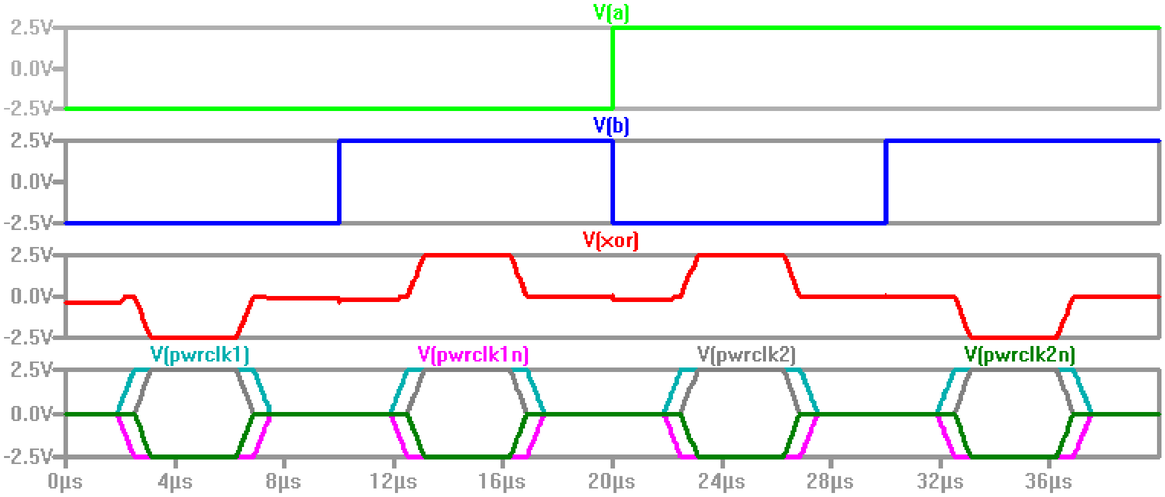


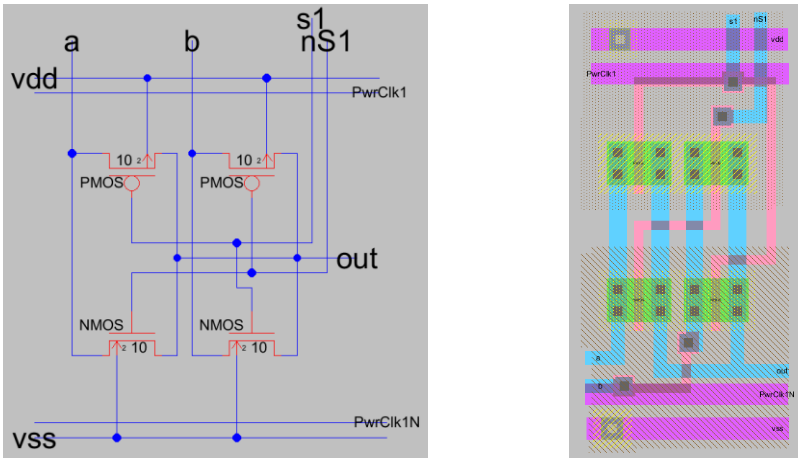
Figure 4.11 – Transient simulation of the adiabatic 2-input XOR gate over four cycles. Operation frequency is 100 kHz. Simulation from layout with extracted parasitics.

4.2.6 – TG-based Multiplexers

The proposed multiplexers are based in transmission gates (TGs). TGs and associated devices do not add logic depth to a Bennett-clocked circuit per se because they do not perform any logic operation on the inputs but simply pass the signal to the output node. However, they need both the selector bits and complements to be ready before the inputs can be energized. Making sure that the selectors are properly computed and energized before attempting data transmission demands special attention in the logic design phase of the work. We called this type of cell a “level 0 gate”.

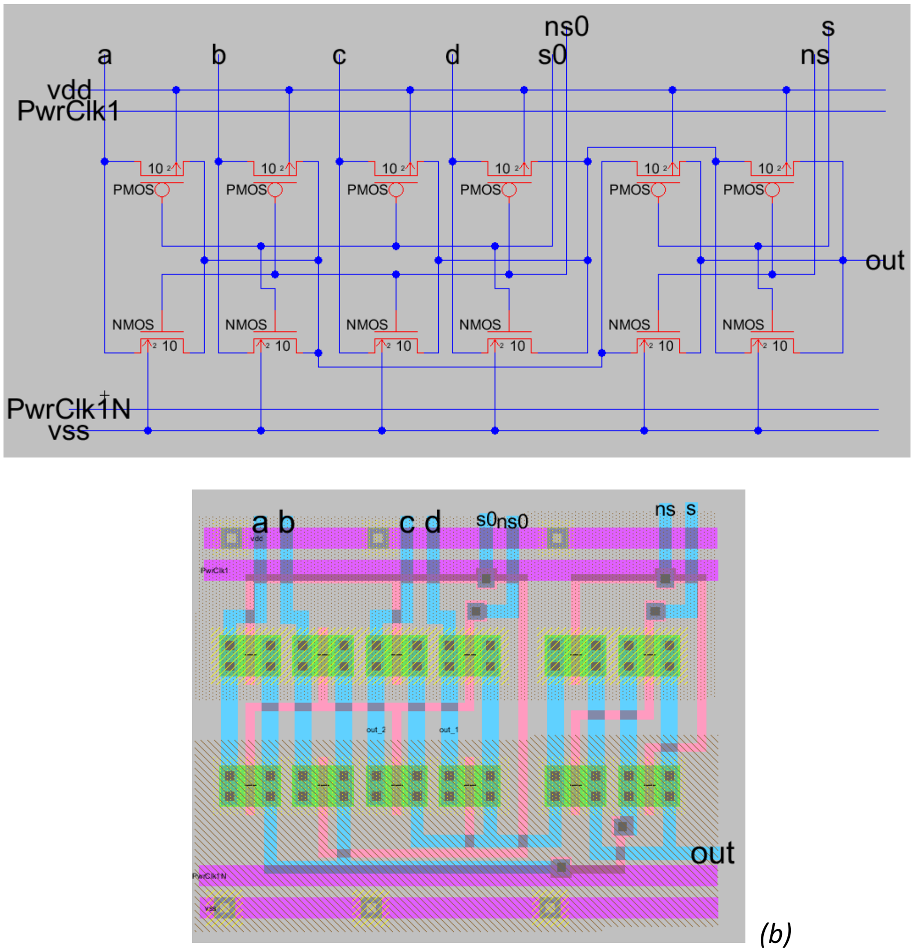
Multiplexers placed at the beginning of a data path have no time available to compute their selector bits beforehand and cannot be level 0 gates. We created an inverting multiplexer (logic depth 1) and non-inverting (logic depth 2) to solve the issue. These cells have internal inverters that compute the selector complements and buffer the inputs. These are discussed in more detail in Chapter 6. The 8-bit adiabatic ALU needs only level 0 multiplexers.

Figure 4.12 shows the schematic diagram and layout for the 2-to-1 level 0 multiplexer. Notice that in the layout signal inputs appear on the left side rather than on top. This is because in most cases this kind of multiplexer has inputs coming from the same logic level. Only the selector bits are above the rails because these have to be energized from a previous level. Figure 4.13 shows the schematic and layout for the 4-to-1 level 0 multiplexer. Even though it works similarly to the 2-to-1 multiplexer, we decided to have the inputs above the cell because it is very wide and an 8-bit bus is likely to need a lot of space, perhaps even demanding an entire row by itself.



(a) (b)

Figure 4.12 – 2-to-1 level 0 adiabatic multiplexer device. (a) Schematic. (b) Layout.



(a)

(b)

Figure 4.13 – 4-to-1 level 0 multiplexer adiabatic multiplexer device. (a) Schematic. (b) Layout.

A transient simulation of the 4-to-1 multiplexer is shown in Figure 4.14. The output node (red) follows the input signal indicated by the selector bits in the top rows (green and blue). For this simulation the selectors are square waves instead of Bennett power clocks to signify that they need to be completely energized before attempting to transmit any data. Transient simulation for the 2-to-1 multiplexer can be found in Appendix B.

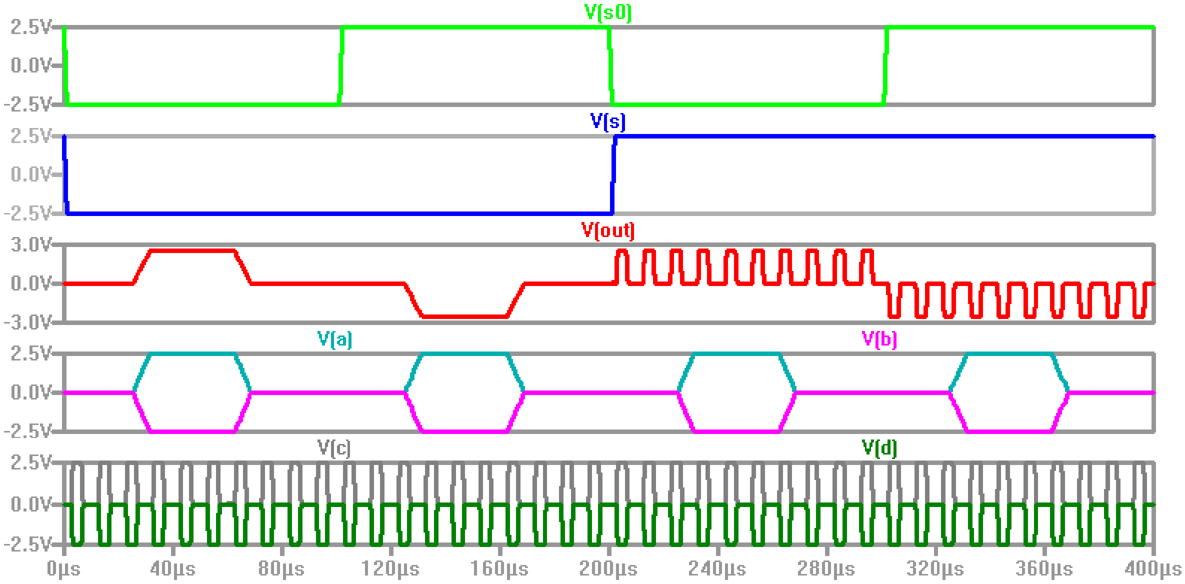
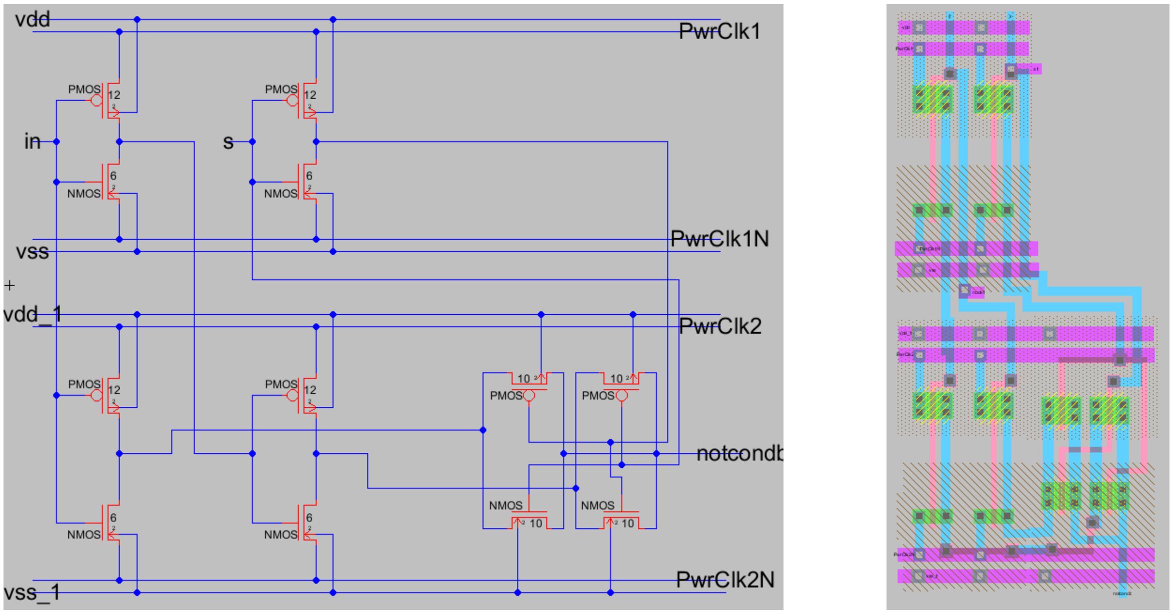


Figure 4.14 – Transient simulation of the 4-to-1 level 0 multiplexer device. Simulation from schematic.

4.2.7 – TG-based conditional inverter.

The conditional inverter performs a logic NOT operation on the input if the control signal is high. Otherwise the output has the same logic value as the input. Our implementation inverts the input and control bits adiabatically, then uses a second inverter to buffer the input and complement and a level 0 multiplexer to select which signal (inverted or direct) to present at the output. The cell has a logic depth of 2. Figure 4.15 shows the schematic and layout of the conditional inverter.

A transient simulation of the cell operation is shown in Figure 4.16. The input (green) is held at a constant logic high. The control bit (blue) determines whether the output (shown in red) is the inverted or the direct input. Even though this cell contains a TG-based multiplexer, the output is buffered and therefore follows the power clock waveform even in non-inverting mode.



(a) (b)

Figure 4.15 – TG-based conditional adiabatic inverter device. (a) Schematic. (b) Layout.

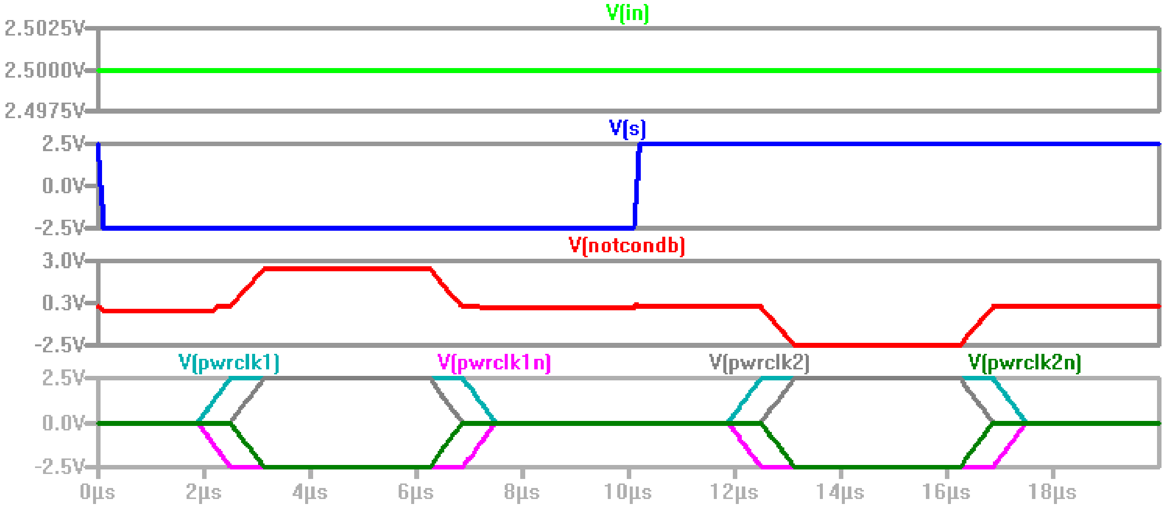


Figure 4.16 – Transient simulation of the conditional inverter device. Simulation from layout with parasitic elements.

4.3 – Power VS Speed Trade-Off for an Adiabatic Inverter Gate.

The main advantage of using adiabatic logic is lower power dissipation. Dissipated power in a digital circuit has two main components: static and dynamic. Using adiabatic techniques lowers total power dissipation because it decouples static and dynamic power. This allows the optimization of both power quantities simultaneously (for example by adjusting operating voltage). Another advantage is that the dynamic power is multiplied by a factor proportional to the operating frequency and the RC time constant of the gate. This relationship implies that power dissipation of adiabatic cells is much lower at relatively low frequencies.

We analyzed the power dissipation for the minimum size inverter shown in subsection 4.2.1 with NMOS size 8λ and PMOS size 16λ. Since the circuit topology is similar to a standard CMOS inverter, the same cell can be used in adiabatic or CMOS mode by connecting it to the power clocks or the DC power supplies respectively. Both operation modes were simulated in LTspice at various frequencies between 100 kHz and 5 GHz.

Figure 4.3.1 shows a comparison between the standard CMOS and adiabatic inverter as a function of frequency. The inverter was loaded with 20 fF for both cases. A square input with half the operating frequency was fed into the inverter. Rail-to-rail voltage is 5V in both cases, well above the threshold value for the 500 nm process.

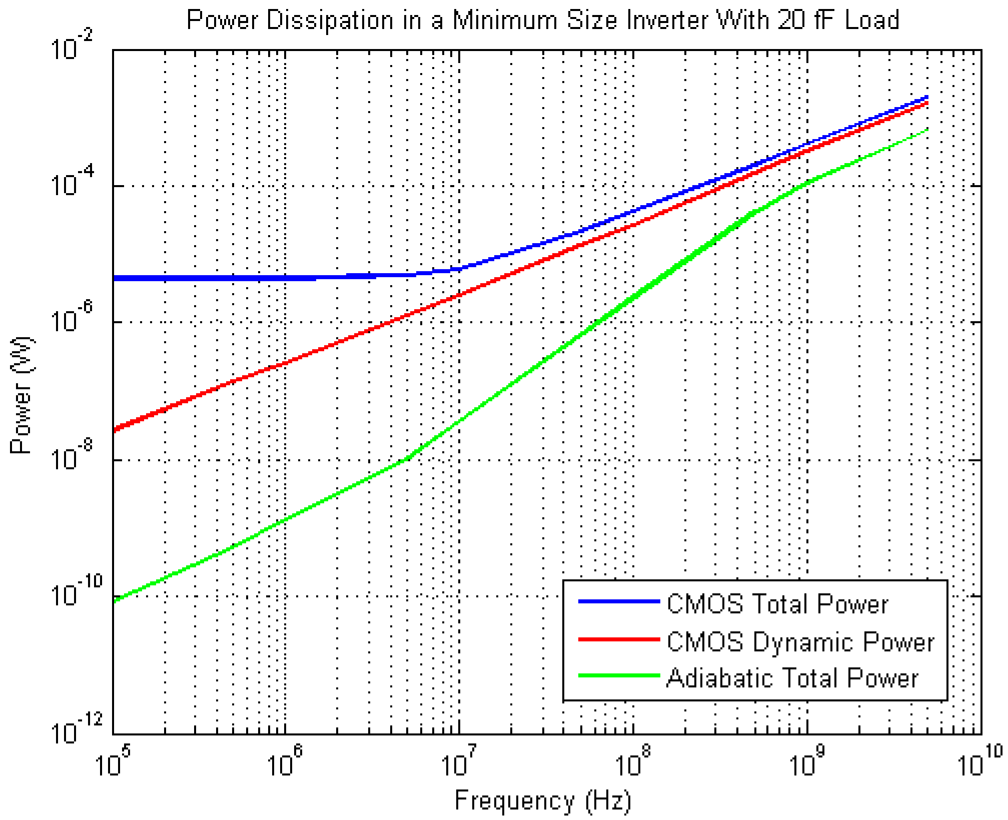


Figure 4.17 – Comparison in simulation of dissipated power for the minimum size inverter. 20 fF capacitive load.

In CMOS mode, total power dissipation was found by multiplying the rail-to-rail voltage and the power supply current. This was done over two cycles to show the average power between a charging cycle to logic high and to logic low. Figure 4.17 shows this plot in blue. At low frequencies, the main contributor to total dissipation is static power. Because of this, total power dissipation of the inverter remains roughly constant until dynamic power takes precedence. This happens around the 10 MHz mark in this example.

At low frequencies it is not possible to compare switching power between modes by looking at the total dissipation of the CMOS mode. To compute dynamic power in CMOS mode we can multiply the output voltage by the current flowing in or out of the capacitive load. Current flowing into the capacitor dissipates across the PMOS transistor while current flowing out dissipates across the NMOS. Since both currents cause power dissipation, we take the absolute value of the product mentioned before. This value is plotted in red in figure 4.17. Unlike static power, dynamic dissipation is proportional to the operating frequency as shown in Equation 4.1:

$$P_{dynamic-CMOS} = CV_{dd}^2 f \qquad (4.1)$$

The total power dissipation in adiabatic mode is plotted in green. This value was found by multiplying the voltage of each power clock by its current and adding them together. As operating frequency increases, adiabatic power dissipation approaches standard CMOS. This is undesirable because adiabatic techniques have large technical overheads. Particularly for Bennett-clocked systems, adiabatic mode operation requires a number of very precise power clock sources.

Dynamic power dissipation in adiabatic circuits is given by Equation 4.2:

$$P_{dynamic} = NCV_{dd}^2 f\left[\alpha \frac{f}{f_0} + (1-\alpha)\right] \qquad (4.2)$$

where *N* is the number of logic gates, *C* is the load capacitance of the gate, V_{dd} is the rail-to-rail voltage, *f* is the operation frequency, α is the fraction of the system that is adiabatic and f_0 is the characteristic frequency of the gate as given by the inverse of the *RC* time constant.

For a single inverter gate as in the simulated example, α and *N* are both equal to 1. We can rewrite the equation as follows:

$$P_{dynamic} = CV_{dd}^2 f \frac{f}{f_0} \qquad (4.3)$$

The first term is identical to the dynamic power in a standard CMOS gate. Substituting that in the equation gives us Equation 4.4:

$$P_{dynamic-adiabatic} = P_{dynamic-CMOS} \frac{f}{f_0} \qquad (4.4)$$

Figure 4.18 plots equation 4.3 in magenta, using the same range and parameters of Figure 4.17. We compare that to the plot of adiabatic total power (green) shown in the previous figure. Both plots show similar values, especially around the middle part of the curve.

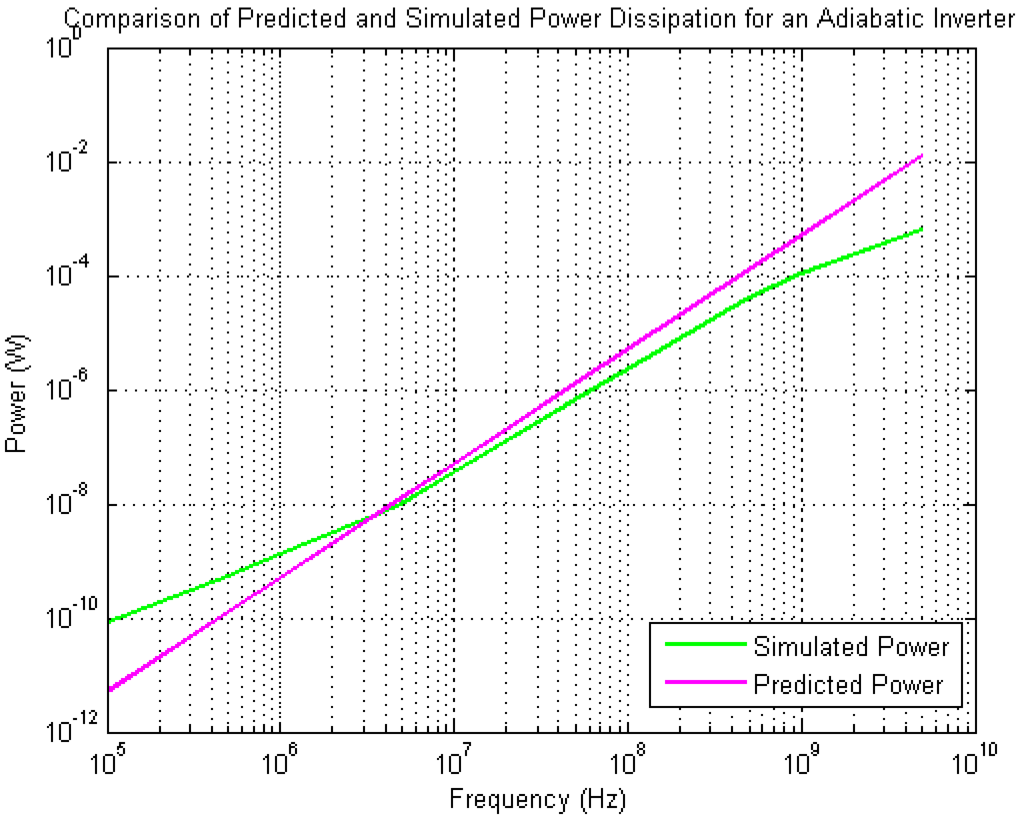


Figure 4.18 – Comparison of predicted and simulated power dissipation for an adiabatic inverter. Range and parameters are the same as in Figure 4.17.

4.4 – Chapter conclusion

This chapter described the standard cell library designed for the 8-bit Bennett-clocked adiabatic ALU. The chapter also provides a discussion of the reasons for using the standard cell technique, as well as considerations and design choices made to conform to technology design rules and requirements.

Each of the cells in the library was described in detail. In most cases, topologies match traditional CMOS design and every deviation from the norm was recorded and shown in this chapter. Cells include a schematic diagram as well as the layout proposed for fabrication in CMOS 500nm process. The Cell´s analysis include a transient simulation to show typical behavior of the cell in adiabatic mode. Some of the cells in the library were not discussed in detail because they are too similar to other cells already shown in this chapter. For those cells, schematics, layouts and simulations can be found in the Appendix B.

Finally, a comparison between Bennett-clocked adiabatic and traditional CMOS operation is provided. The minimum size inverter was chosen to illustrate this comparison. The design allows easy operation of the cell in either mode. Simulations show that adiabatic operation has considerably lower power dissipation at low frequencies. Those results compare favorably with the theory and we conclude that a range of frequencies for operation in adiabatic mode is highly desirable.

Chapter 5. Adiabatic 8-bit ALU with Carry-Lookahead Adder.

This chapter will look at the 8-bit adiabatic ALU in detail. The ALU uses a carry-lookahead adder and is able to perform the addition, subtraction, bitwise AND, bitwise OR and “set if less than” operations required by the MIPS architecture specifications. The ALU is able to operate in Bennett-Clocked adiabatic and standard CMOS modes.

The first section of the chapter will cover the design of the ALU. Block diagrams and the layout are shown for the unit along with a discussion of the design alternatives made. Slight modifications to the design are made and detailed to allow standalone operation and fabrication conforming to the CMOS 0.5 μm ON Semiconductor’s C5 process.

The second part focuses on SPICE simulations based on the layouts shown in the previous section. The unit was simulated using RC conservative parasitic extraction to show the intended behavior of the circuit in adiabatic mode. Additional simulations in both, adiabatic and standard CMOS operation modes, serve to establish a comparison in terms of frequency and power dissipation. Speed and power trade-offs are discussed in this chapter as well. Simulations show that the adiabatic mode has a considerably lower power dissipation for a sensible range of frequencies.

The chapter closes with a series of microscope photographs on the fabrication process of the ALU using CMOS 1μ process at the Notre Dame IC Fabrication Laboratory. This is an on-going project at the University of Notre Dame Nano and Micro-technology group.

5.1 – Design of the ALU.

The adiabatic ALU was designed for use in an 8-bit MIPS microprocessor. To comply with the architecture requirements, the ALU has an adder module, a conditional inverter to perform subtraction, a bitwise AND module, bitwise OR and a multiplexer that allows selection of the output result. Figure 5.1 illustrates a block diagram showing every module included within the adiabatic ALU. Thick lines represent 8-bit buses and the red thin lines are the control signals.

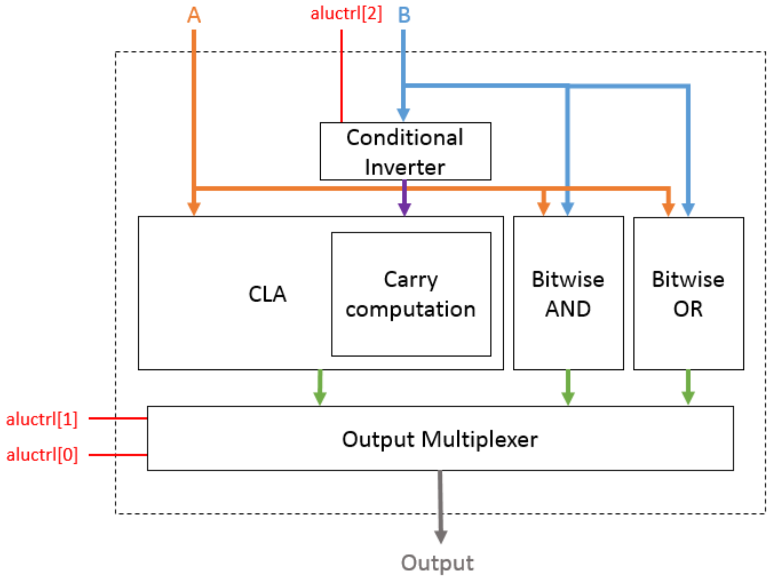


Figure 5.1 – Block diagram of the adiabatic ALU showing internal functional modules.

The MIPS architecture does not specify the need for a flag register, so the carry out bit was ignored. However, a “zero” flag is used as a control signal elsewhere and it was necessary to include a zero detector. This is not considered part of the ALU and is described in detail in Chapter 6 instead, as a stand-alone component of the microprocessor.

The goal of this design was to minimize total logic depth of the system. To this end, a carry-lookahead adder (CLA) was chosen because it computes every bit of the addition operation simultaneously regardless of word length. Other adder types (such as the ripple-carry adder) have a latency that depends on the word size and resulted in higher logic depth values for the 8-bit case. Using a CLA resulted in a total logic depth of 7 levels for the whole ALU.

The ALU’s logic design was led by Prof. Ismo Hänninen at University of Notre Dame, using the custom Verilog HDL tools described in Chapter 3. Tools to perform synthesis of Bennett-clocked logic are not available and therefore, a conversion to a physical layout was done by hand, using the standard cell library described in Chapter 4.

Based on the HDL code, a gate-level electric diagram was prepared in Electric VLSI as a guide to facilitate the layout work. This diagram is far too detailed to be accurately portrayed in printed media, however a downscaled version is shown in Figure 5.2 for illustrative purposes.

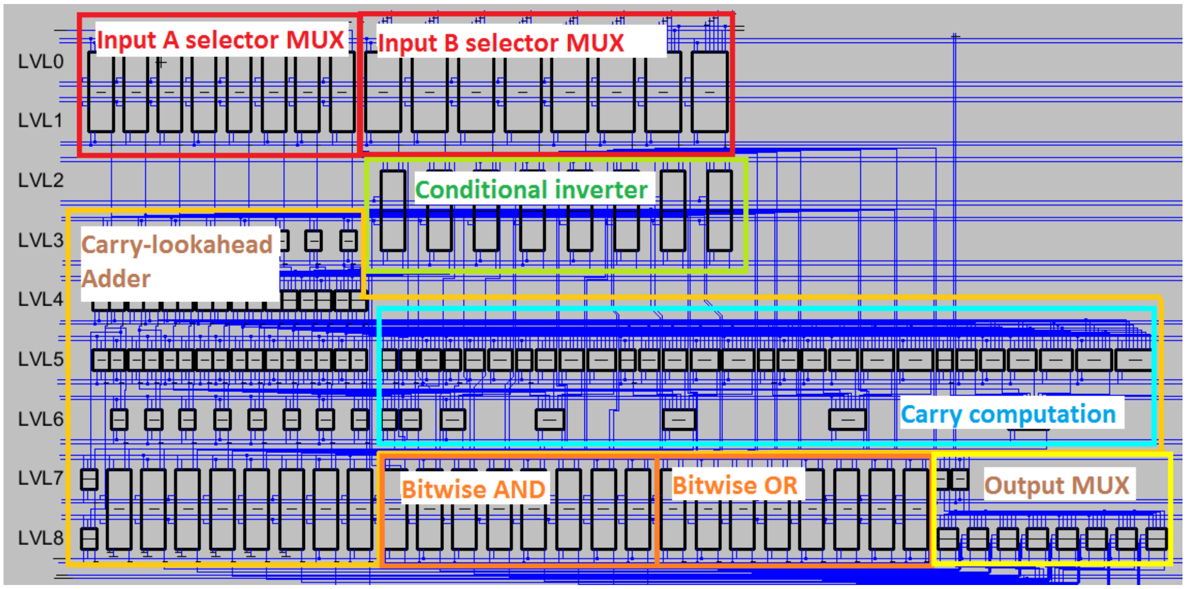


Figure 5.2 – Downscaled gate-level block diagram showing functional modules.

In the block diagram, gates are represented by rectangular blocks rather than the standard electric symbols, to signify that these are not conventional logic gates and to explicitly show their power clock connections. Logic levels are labeled on the left side and each one has 4 power lines running horizontally. These lines are VDD, positive power clock, negative power clock and VSS in order from top to bottom. Functional blocks are marked on the figure and correspond to the modules shown in Figure 5.1, with the exception of the input selector multiplexers. These are not considered part of the ALU but appear in the diagram as part of the same data-path.

Layout design of Bennett-clocked systems demands awareness of the logic level in which each gate is located. Cells that share logic levels in logic design must also share power clock connections physically. Furthermore, data always has to flow down its path and can never feedback into a previous level inside a Bennett-clocked system. These constraints dictate much of the physical arrangement of cells in layout.

To ease placement of cells, comply with the previously mentioned constraints, and diminish complexity and length of power clock routing we decided to have each standard cell row correspond to a single logic level. Almost every cell has inputs on top and outputs at the bottom to allow data flow in the correct direction. The only exceptions are pseudo-cells that are a part of a larger cell and **they are not** to be placed individually.

Unfortunately, this placement creates some issues not present otherwise. Traditional standard cell layouts define a constant row width and arrange cells side by side until the row is full, which results in optimal space utilization provided that signal routing is minimized. This advantage is lost when defining each cell row as a single logic level, because space utilization becomes tied to the logic design. From figure 5.1.2 it can be seen that the bottom levels are much denser than the top ones.

Figure 5.3 shows a downscaled picture of the proposed layout for the ALU. It resembles the block diagram of Figure 5.2, but a few deviations were permitted to improve space utilization.

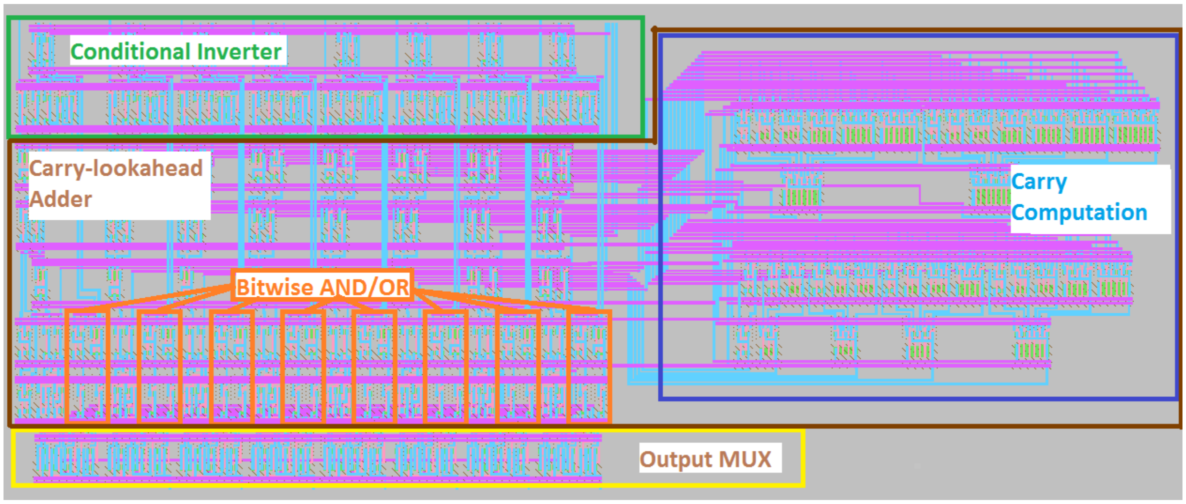


Figure 5.3 – Layout of the adiabatic ALU showing functional modules.

The most populated rows are labeled levels 5 and 8 in figure 5.2. Level 5 has adder logic and the top half of the carry computation sub-module. Level 8 has logic for all 3 possible operations plus the output multiplexer bus. These levels would be more than double the average width of the others if placed in a single row, so the rows were split and the power rails were duplicated as necessary. As a result, the output multiplexer was placed in its own line, tied to the appropriate power clocks; and the carry computation sub-module was designed as a separate subcircuit. By doing this, it was possible to attain a higher transistor density and keep a uniform row width across the ALU.

Another deviation from the block diagram is that the design of the ALU layout follows the data flow direction instead of having individual modules for each function assembled together. Each vertical column or "bit slice" performs all relevant operations for a single bit. Bit slices are then replicated 8 times and placed side by side. As a result, the adder blocks became interweaved with the AND/OR blocks as seen in Figure 5.3. Figure 5.4 shows the layout again with the bit slices and the carry computation sub-module highlighted.

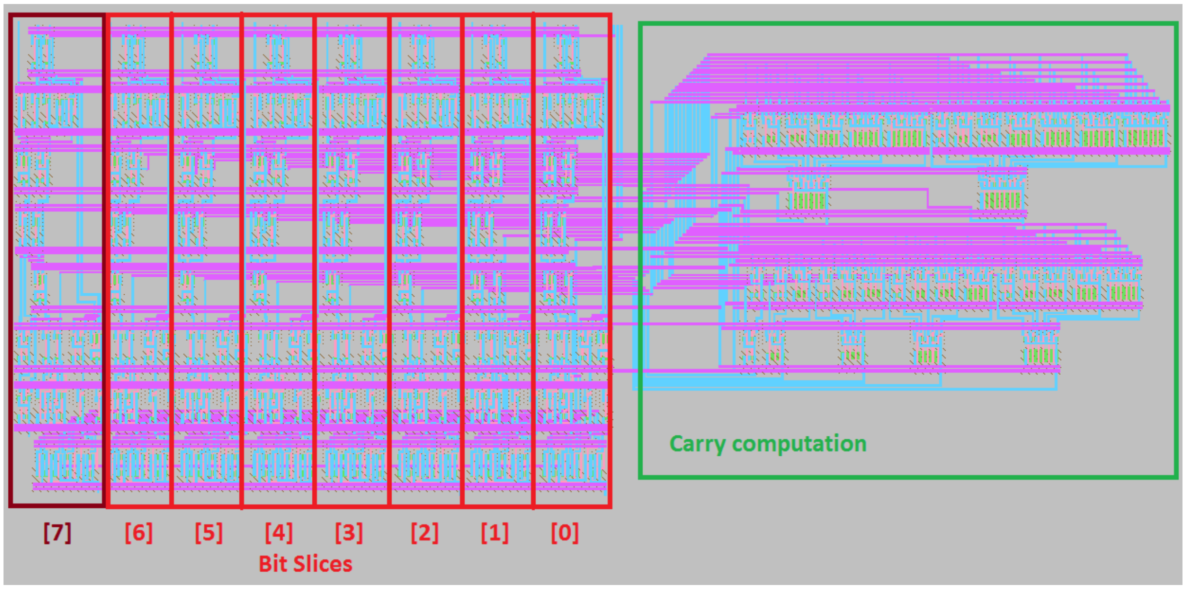


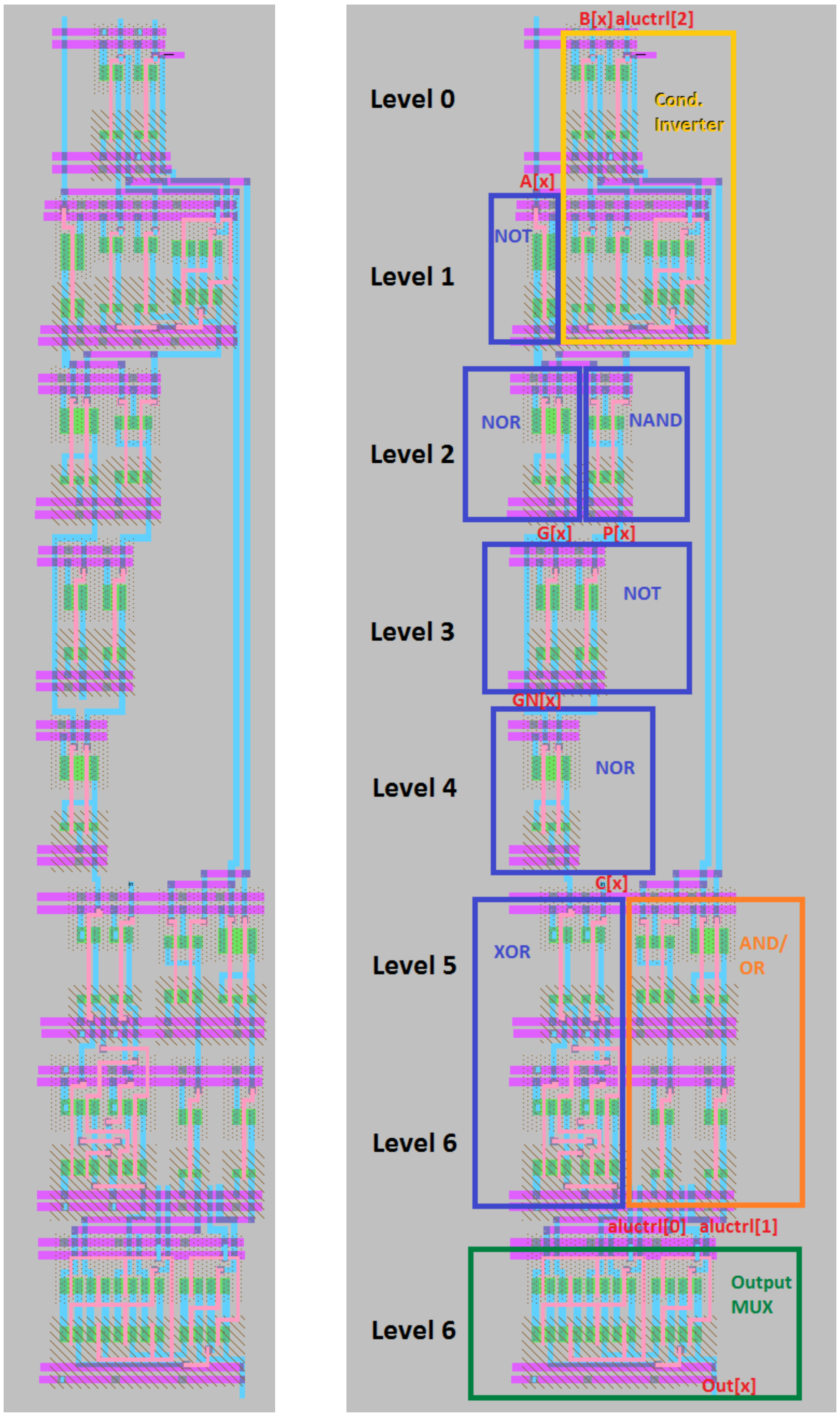
Figure 5.4 – ALU layout showing bit slices and the carry computation sub-module.

5.1.1 – Bit slice design.

Each bit slice contains the necessary components to perform all ALU operations between a single pair of bits A[x] and B[x] as well as a multiplexer that selects the appropriate output. Bit slices interface with the carry computation sub-module in order to perform the carry-lookahead addition. Table 5.1 indicates outputs and inputs for a single bit slice and the logic level at which they are ready or needed respectively. The topmost level of the ALU is called "Level 0" in this table.

*Table 5.1 – Inputs and outputs for a bit slice. The "time slot" indicates earliest time at which signal is ready for an output, and latest time at which a signal should be available for an input. *aluctrl[1-0] require complementary signal. These are computed once and shared among slices.*

Signal	Mode	Time slot
A[x]	Input	Start of Level 1
B[x]	Input	Start of Level 0
aluctrl[2]	Input	Start of Level 0
P[x] (Propagate [x])	Output	End of Level 2
G[x] (Generate [x])	Output	End of Level 2
GN[x] (Inverted Generate [x])	Output	End of level 3
C[x] (Carry [x])	Input	Start of Level 5
aluctrl[1]*	Input	Start of Level 6
aluctrl[0]*	Input	Start of Level 6
Out[x]	Output	End of Level 7



(a) (b)

Figure 5.5 – Bit slice layout (a) and annotated bit slice layout with highlighted components, inputs and outputs (b).

Figure 5.5 (a) shows the layout for a single bit slice. Figure 5.5 (b) has the inner components of the slice highlighted and shows location of inputs and outputs (in red). Highlighted in blue are the components pertaining to the CLA, while the rest are labeled as the modules they represent.

A two levels deep conditional inverter begins the data-path in level 0 and can output either B[x] or its complement depending on the value of aluctrl[2]. Meanwhile, in level 1, an inverter outputs the complement of signal A[x]. In level 2, the NOR of both previous signals outputs the generate G[x] and the NAND outputs the propagate P[x]. Both signals are routed to the carry computation sub-module, which operates in parallel during levels 3 and 4. Back at the bit slice, level 3 inverts both propagate and generate. The inverted generate GN[x] is then routed to the carry computation subcircuit. A NOR operation is computed between the inverted forms of propagate and generate in level 4 at the bit slice.

Just before level 5, the carry signal C[x] arrives from the carry computation sub-module and then undergoes the XOR operation with the output signal of level 4. Bit slice 0 has aluctrl[2] connected instead of C[x]. The XOR takes 2 phases to be completed and outputs the sum or subtraction of A[x] and B[x] depending on the value of the aluctrl[2] signal. Meanwhile, in levels 5 and 6 both AND and OR operations are performed on bits A[x] and B[x].

Finally, a level 0 multiplexer (described in Chapter 4) selects between AND, OR, sum or "set if less than" (SLT) operations using signals aluctrl[1] and aluctrl[0] as selectors. This multiplexer requires both the direct and complement form of the control signals. The complements are computed in level 5 and shared between all slices. The hardware necessary for this inversion is included in the slightly modified bit slice 7. Figure 5.4 shows bit slice 7 in a darker color to highlight this difference.

The SLT operation outputs a constant 0 for bits 7-1 and the value of Sum[7] (sign) for the least significant bit. This ensures that a 0x01 appears at the output when the SLT operation is called and the value of A is lower than B; and a 0x00 appears otherwise. Table 5.2 shows the values of aluctrl required for each operation in the MIPS architecture.

Table 5.2 – ALU control inputs corresponding to each ALU action.

Operation	aluctrl[2-0]
Add	**010**
Subtract	**110**
AND	**000**
OR	**001**
SLT	**111**

5.1.2 – Carry computation sub-module design.

As described earlier in this section and evidenced in Figure 5.2, the carry computation subcircuit of the CLA is large enough and became a separate sub-module. The purpose of this circuit is to receive the "propagate" and "generate" signals from the adder and compute all carries simultaneously.

Table 5.3 shows the input and output signals for the carry computation circuit, as well as the required timing at which they are available. Time slots in this table are counted the same way as in the bit slices, that is, level 0 is the topmost level of a bit slice.

Table 5.3 – Inputs and outputs for the carry computation sub-module. The "time slot" indicates earliest time at which signal is ready for an output, and latest time at which a signal should be available for an input.

Signal	Mode	Time Slot
P[6:0] (Propagate [6:0])	Input	Start of level 3
G[5:0] (Generate [5:0])	Input	Start of level 3
GN[6:0] (Inverted Generate [6:0])	Input	Start of level 4
C[7:0] (Carry [7:0])	Output	End of level 4
aluctrl[2] / Carry In	Input	Start of level 3

Carry computation involves multiple NAND operations. Each carry bit requires N+1 NANDs, where N is the index of the carry bit being computed. Number of inputs of the NAND gates also grows with this index number. Computation of a carry bit requires two instances of a maximum input size gate, with number of inputs equal to N+1; and a single instance of every smaller gate down until the 2-input NAND. Figure 5.6 shows the carry computation circuit for the first, second and third carry bits to illustrate this growth in both amount of NAND gates and number of inputs.

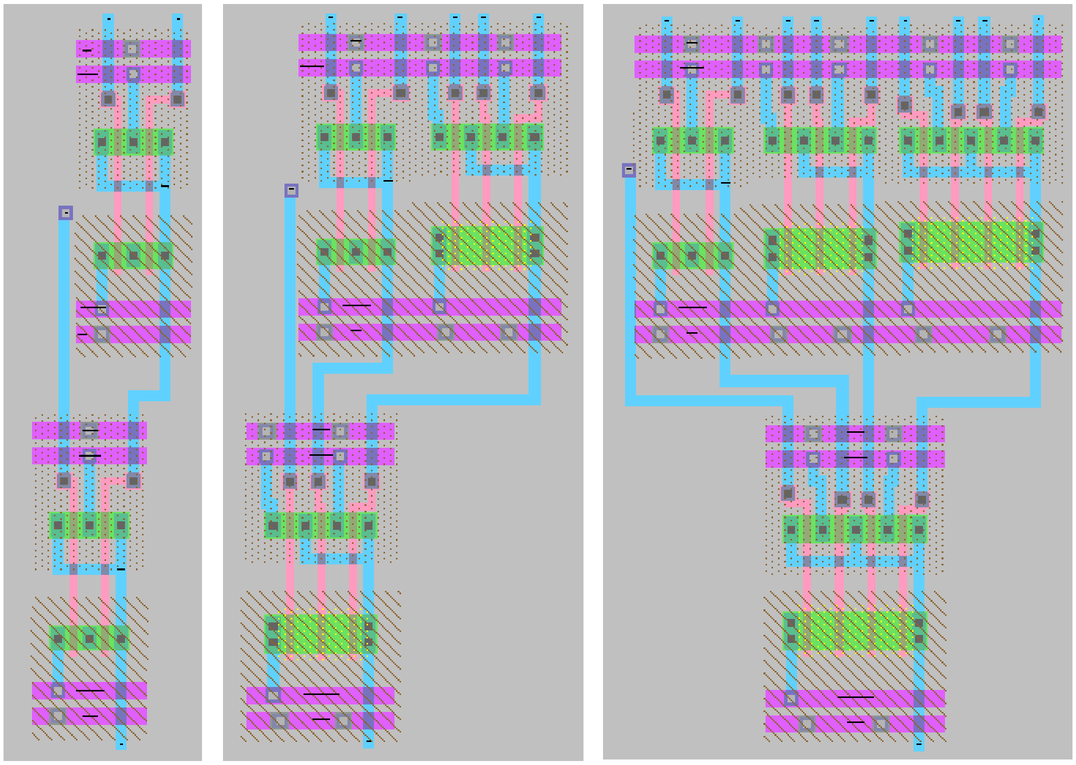

(a) (b) (c)

Figure 5.6 – Carry computation circuits for C[1] (a), C[2] (b) and C[3] (c). Amount of NAND gates and maximum number of inputs in a single NAND gate are both equal to N+1, where N is the index of the carry bit being computed (C[N]).

Figure 5.7 shows the complete carry computation sub-module layout. The carry computation circuit for each bit was designed separately and then assembled together. An annotated version of the layout in Figure 5.8 shows these subcircuits explicitly.

Although the layout is several rows tall, the module has a logic depth of 2 and computes every carry bit simultaneously using levels 3 and 4. Beside the input and output buses, the design includes connections for VDD, VSS and the positive and negative versions of both power clock phases required. These connections are routed internally to their correct power rails.

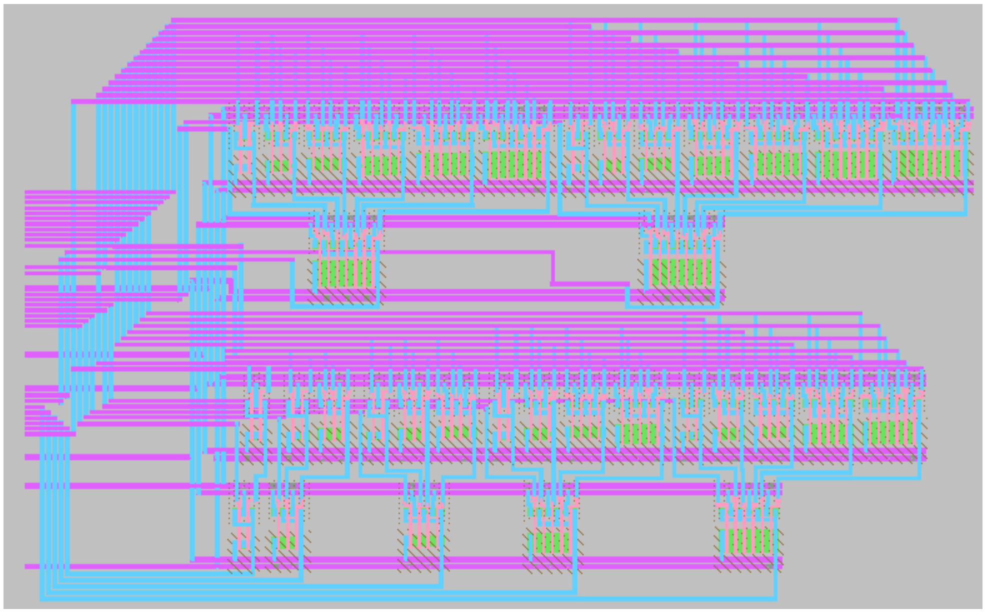


Figure 5.7 – Carry computation sub-module layout.

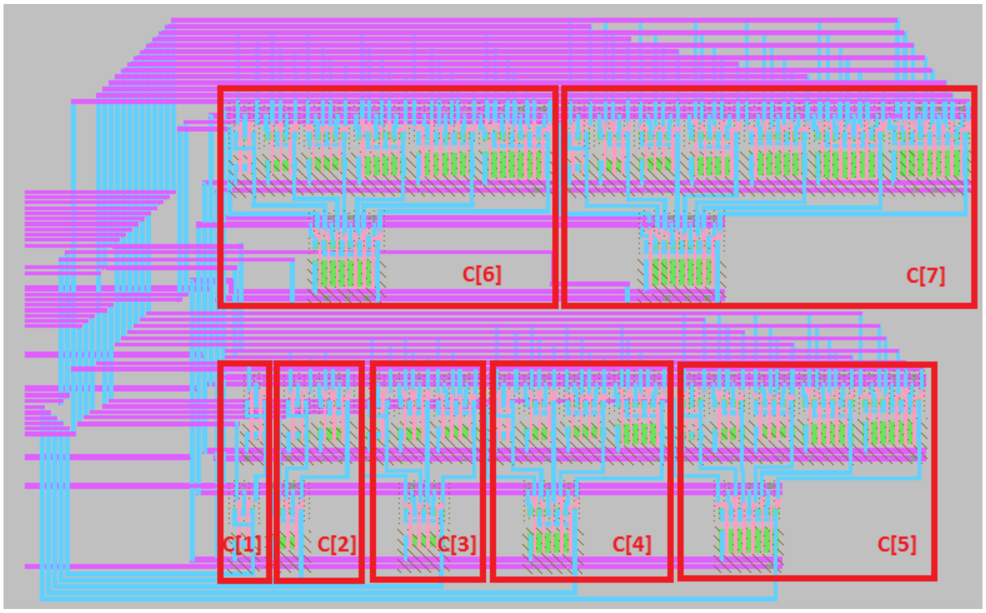


Figure 5.8 – Annotated carry computation sub-module layout explicitly showing individual carry computation sub-circuits.

5.1.3 – Standalone adiabatic ALU for MOSIS C5 Fabrication.

The adiabatic ALU was designed for use in an 8-bit MIPS microprocessor. To fabricate and test the ALU by itself, slight modifications were necessary. Additional constraints are put in place by the die packaging and by fabrication in the C5 process. This subsection details changes made to the original design for standalone operation and MOSIS fabrication using CMOS 0.5 μm process.

Bennett-clocked adiabatic circuits have very strict timing requirements for input signals and impose special waveforms onto the outputs. Inputs need to arrive before the level they are needed in begins charging, and must hold their value until the level discharges. For some cases (for instance the ALU A and B inputs) the acceptable window in which inputs can arrive is only when all power clocks are in the relaxed state (0 V). Outputs are only valid while their level remains energized. This means that for the final output of the system, its value will be valid only while all power clocks are energized.

When preparing a test bench circuit for the standalone adiabatic ALU, the designer should take care of input timing concerns. A programmable system like a microcontroller or FPGA can ensure that the proper signals are fed into the system at the correct times. However, reading the output is complicated because it is only valid for a very short time period. This is an issue that comes up within the MIPS processor too.

The solution to this issue is to place a D-type flip-flop to sample the output and hold it between operation cycles. Figure 5.9 shows a logic gate diagram for a conventional master-slave flip-flop using transmission gates (TG) and adapted from **[45]**.

To get around the restrictive timing, the master latch and slave latches have separate special clocks each, labeled sample and update clock respectively. The sample clock goes high when every Bennett power clock is energized and allows the master latch to acquire the correct output value. The update clock goes high when all Bennett levels are relaxed and updates the output of the flip-flop with the value that was sampled last. The update operation needs to wait until a fully relaxed state, because it might be used as an input to another adiabatic circuit.

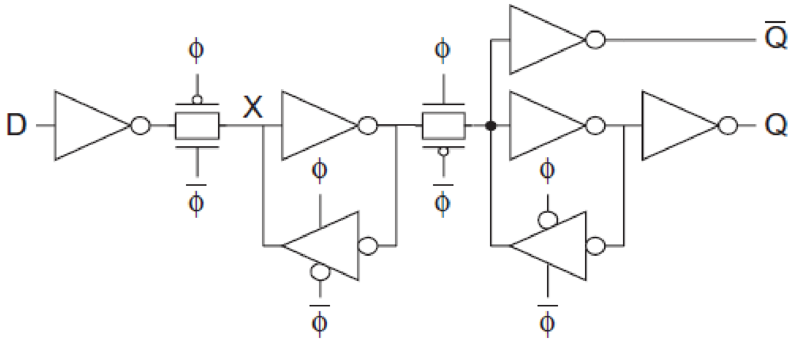


Figure 5.9 – D-type master-slave flip-flop using transmission gates (TG). From **[45]**

Another issue was brought about by the fact that projects fabricated through MOSIS are bonded to a 40-pin package. Table 5.4 shows the original pin count taking in consideration every signal and power rail involved with the adiabatic ALU.

As can be gathered from the table, **it is** not possible to have every signal tied to an external pin due to package limitations. To adjust to the available number of external pins, the carry in bit was omitted and a single data input port was defined. Two 8-bit registers were placed in the circuit using the flip-flop design of figure 5.9. A single enable bit controls which of the two register is written. The optimized pin count is shown in Table 5.5.

Table 5.4 – Maximum pin count for the adiabatic ALU. This considers every possible input, output and power signal involved.

Signal name	Number of pins
Power clocks	14
VDD	1
VSS	1
A input	8
B input	8
ALU control	3
Output	8
Sample clock	1
Update clock	1
Carry in	1
Carry out	1
Total	47

Table 5.5 – Optimized pin count for the standalone adiabatic ALU.

Signal name	**Number of pins**
Power clocks	**14**
VDD	**1**
VSS	**1**
Input port	**8**
A/B selector	**1**
Register clock	**1**
ALU control	**3**
Output	**8**
Sample clock	**1**
Update clock	**1**
Carry out	**1**
Total	**40**

Figure 5.10 shows the final layout of the adiabatic ALU including the two-input and single-output registers. This layout will be placed in a 1.5mm x 1.5mm pad frame and sent to MOSIS for fabrication in a CMOS 0.5µm process.

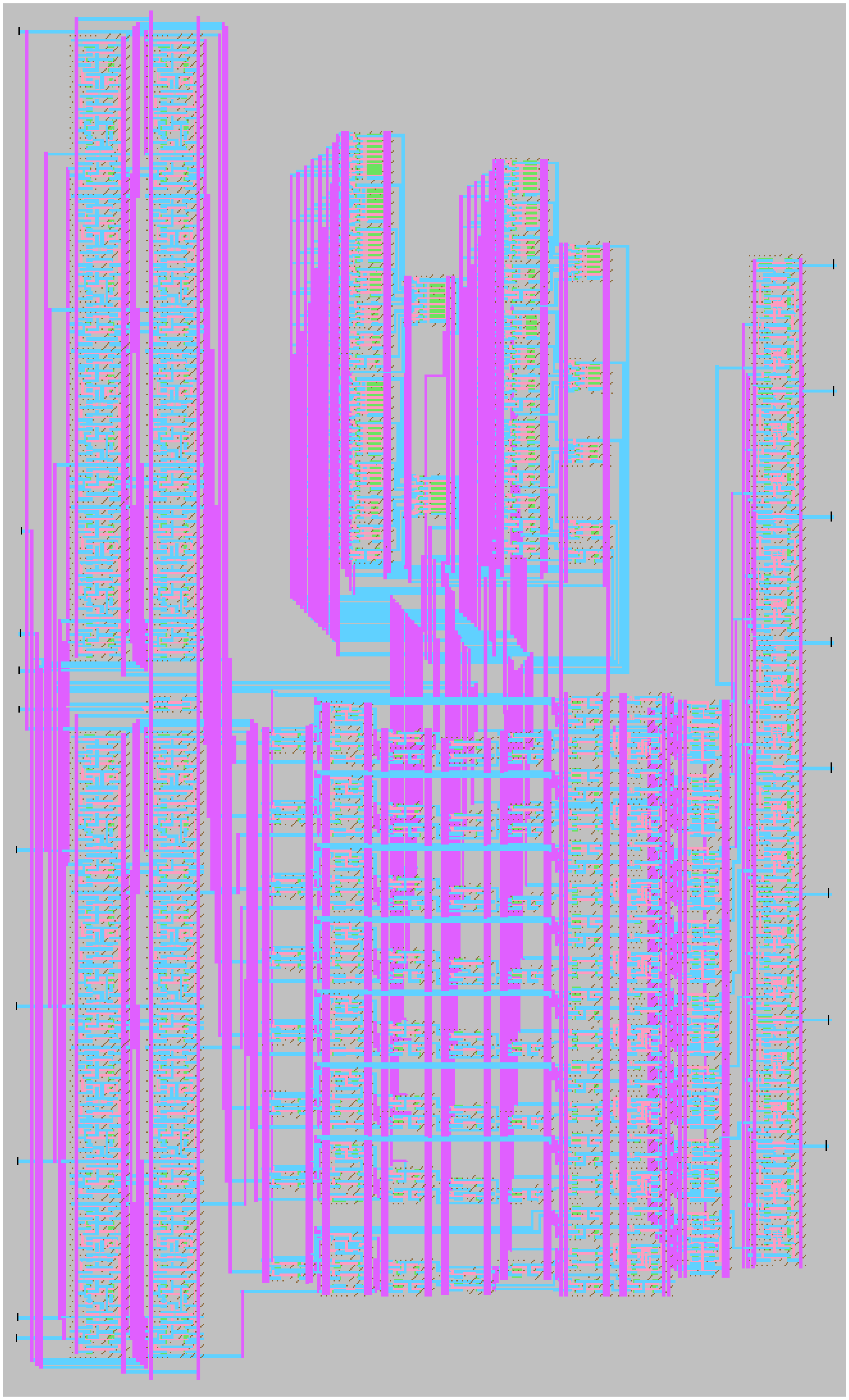


Figure 5.10 – Standalone adiabatic ALU layout including input and output registers.

5.2 – Circuit performance in simulation.

The circuit was tested using SPICE simulations. The first goal in simulation was to observe and validate the intended behavior of the ALU in adiabatic mode. After corroboration of the basic functionality, several more runs were simulated with increasing frequencies, and their total power dissipation was measured. The analysis employs frequencies between 100 kHz and 100 MHz. This study was repeated for the same circuit running in standard CMOS modality. A plot comparing power dissipation of both modes as frequencies increase was created using the data obtained.

The SPICE net lists were created from the layout using Electric VLSI. Conservative RC parasitic extraction was performed using the most recent C5 process parameters **[46]** (see Appendix C for the detailed parameters and the SPICE model of the transistors).

Several test benches were prepared to simulate the circuit at different operating frequencies. SPICE code was used to describe the input signals as well as the 7 sets of power clocks. All non-timed signals were kept the same between test benches and time-dependent signals were scaled to the appropriate values. As a result, all tests are equivalent and only differ in frequency. The SPICE code for all test benches is included in Appendix D.

Figure 5.11 shows the waveforms for the seven power clock phases running at 100 kHz. These clocks were generated using SPICE code and it is assumed that a suitable signal generator will be connected externally to the ALU. No internal circuitry is provided to generate the clocks. At time zero all power clocks are relaxed and the circuit can accept a new set of inputs. Halfway through the period (around 5 µs in this example) all clocks are energized and the circuit output is valid.

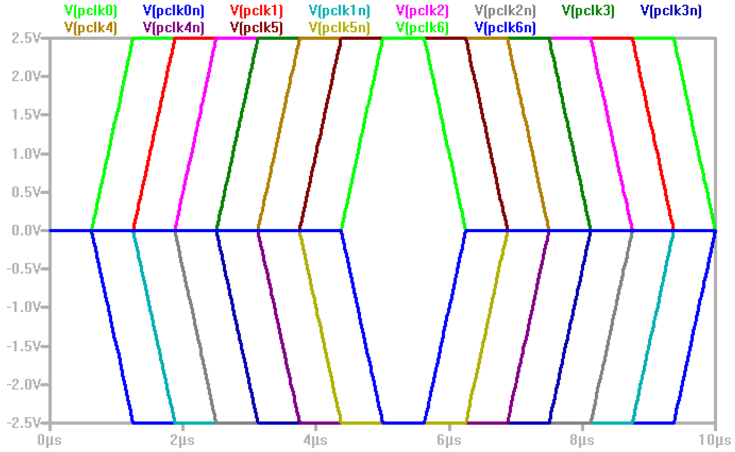


Figure 5.11 – Power clock waveforms from SPICE simulation.

Figure 5.12 was obtained from a transient analysis of the adiabatic ALU intended to show the expected behavior for the system. For this simulation both registers were loaded with the 8-bit number 0xE4 and the ALU control bits were set to perform the addition function. After addition the output bits plus the carry out show 0x1C8 when all power clocks are energized. This is the correct addition operation having both input bytes.

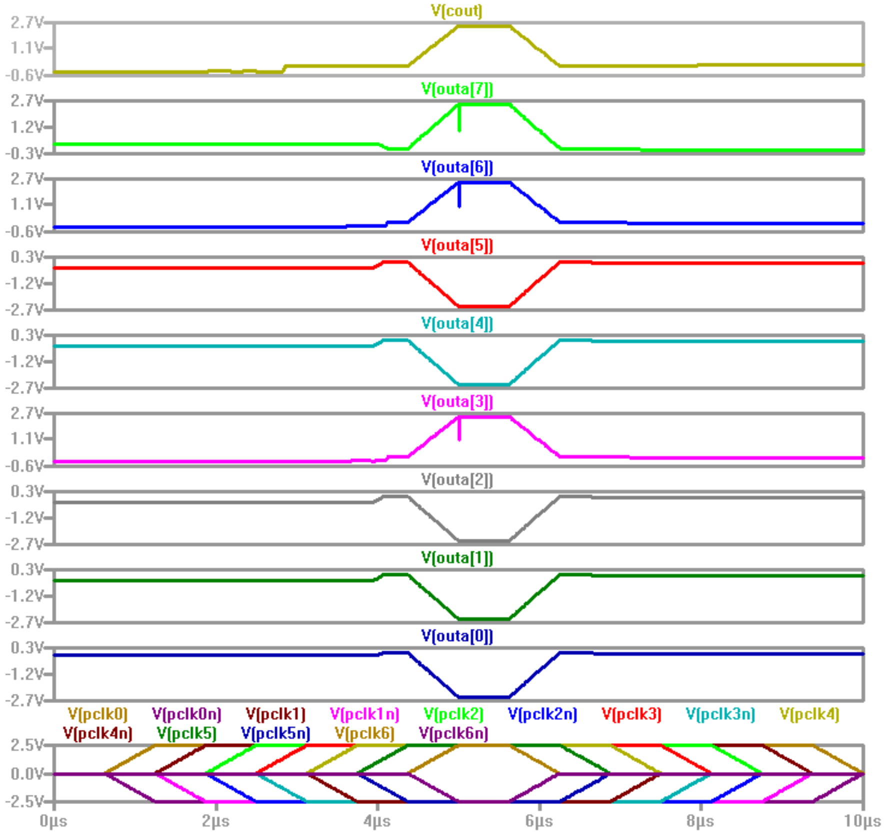


Figure 5.12 – Transient analysis of an addition operation. Inputs are 0xE4 and 0xE4. Output is 0x1C8 when including the carry out bit. This is equivalent to the sum of both inputs.

A different test was prepared for power dissipation and frequency comparisons. The power rail waveforms were kept the same as in the previous example. Transient time was doubled so that the simulation now covers two operation cycles. The data inputs were made to oscillate to have the operations in both cycles be completely different. The first cycle has 0xE4 as the input data for both ports, while the second one has the 1's complement 0x1B. ALU control inputs remain the same in both cycles so the intended operation is always addition.

To run this simulation in adiabatic mode, the power clock rails were connected to the waveforms shown in Figure 5.11. To perform a standard CMOS simulation, the power rails were fed with static DC values 2.5 V for the positive phases and -2.5 V for the negative ones.

Power was measured in adiabatic mode by multiplying the voltage on each power clock rail by the current flowing in or out of the node and then summing them all. For the standard CMOS case, power was calculated by multiplying the current flowing out of the positive power supply by the rail-to-rail voltage. The power supplies VDD and VSS serve as reference for the bulk of each transistor inside the ALU circuitry, but they also energize the input and output flip-flops. It was possible to compare power dissipation just inside the ALU by choosing not to measure the power consumed by the registers.

The simulation was performed over a wide range of frequencies. Table 5.6 shows the data points obtained in simulation. A logarithmic plot prepared with this data is shown in Figure 5.13.

The ratio between dissipated power in CMOS and adiabatic modes was calculated. As frequency increases, this ratio becomes smaller, until the point where adiabatic operation consumes the same power as standard CMOS.

Table 5.6 – Power dissipation data obtained from simulation of Adiabatic and CMOS operation modes. The ratio between both quantities was also calculated.

Frequency	Adiabatic power	CMOS power	Ratio of CMOS/Adiabatic power
100 kHz	**30.109 nW**	**15.399 µW**	**511.4**
500 kHz	**227.36 nW**	**75.72 µW**	**333.04**
1 MHz	**672.16 nW**	**151.47 µW**	**225.3**
5 MHz	**10.35 µW**	**754.86 µW**	**72.93**
10 MHz	**35.77 µW**	**1.3159 mW**	**36.79**
50 MHz	**617.55 µW**	**1.5085 mW**	**2.44**
100 MHz	**2.7798 mW**	**2.8848 mW**	**1.04**

As expected, adiabatic power dissipation is lower than standard CMOS by a wide margin, but this advantage decreases as the operating frequency goes up. At high frequencies, charging ramps are so fast compared to the RC time constant of the adiabatic gates that they appear as abrupt changes in voltage and dissipate completely across the transistors just like in normal CMOS operation. It is also possible that the output is no longer correct at high frequencies, because the duty cycle of the final phases is very short and the load capacitance does not have enough time to charge to its correct value.

The ratio of power dissipation between the ALU operating in adiabatic mode and in CMOS mode, drops to single digit ratios at frequencies between 10 and 50 MHz. A recommendation, therefore, is to run tests slower than 10 MHz.

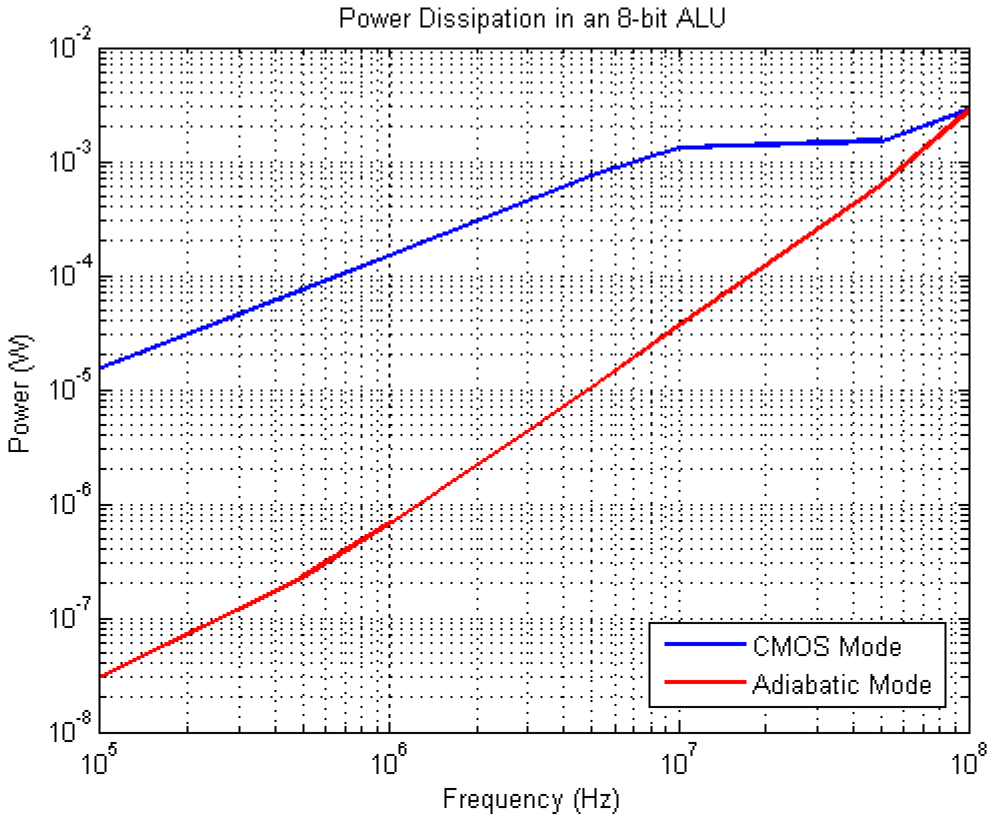


Figure 5.13 – Comparison of power dissipation for the 8-bit ALU in CMOS and Adiabatic operation modes over a wide range of frequencies.

Another frequency limiting factor in this example is found in the registers. The flip-flops used in this design were originally developed for use in the MIPS microprocessor. Because they are intended for low frequency operation inside an adiabatic environment, the internal buffers are minimum size and this results in performance issues at high frequencies.

This limitation is reflected onto the power dissipation plot between the 10 MHz and the 50 MHz mark. In this experiment, CMOS power dissipation stops growing at the same rate beginning with the 50 MHz data point. This is because at this point the flip-flops no longer work and they input the same values (0x00) for both cycles. As a result, the gates inside the ALU charge once at the beginning of the simulation and hold their values for the remainder of the simulation. The result shows lower dissipation as the gates do not dissipate the stored charge or draw current to charge again. In spite of this issue it was decided to keep running the simulations at higher frequencies to find the point at which adiabatic dissipation approaches CMOS power.

5.3 – Fabrication process.

The MIPS microprocessor includes this adiabatic 8-bit ALU and is undergoing fabrication in the IC Fabrication Laboratory at University of Notre Dame. Microscope photographs of the wafer have been taken to document the fabrication process. This section will present scans of the ALU taken from the MIPS photographs.

The first step in the fabrication process is to create the N-well. A P-well is not necessary because that the substrate of the wafer is already P-type. This step does not create a very useful image because no devices are visible yet.

Figure 5.14 shows the ALU after the next fabrication step which is active implantation. Although no transistors are complete yet, their locations are already visible on the die. From the picture, the rough outline of the bit slices (columns) can already be seen along with the output selector multiplexer at the bottom.

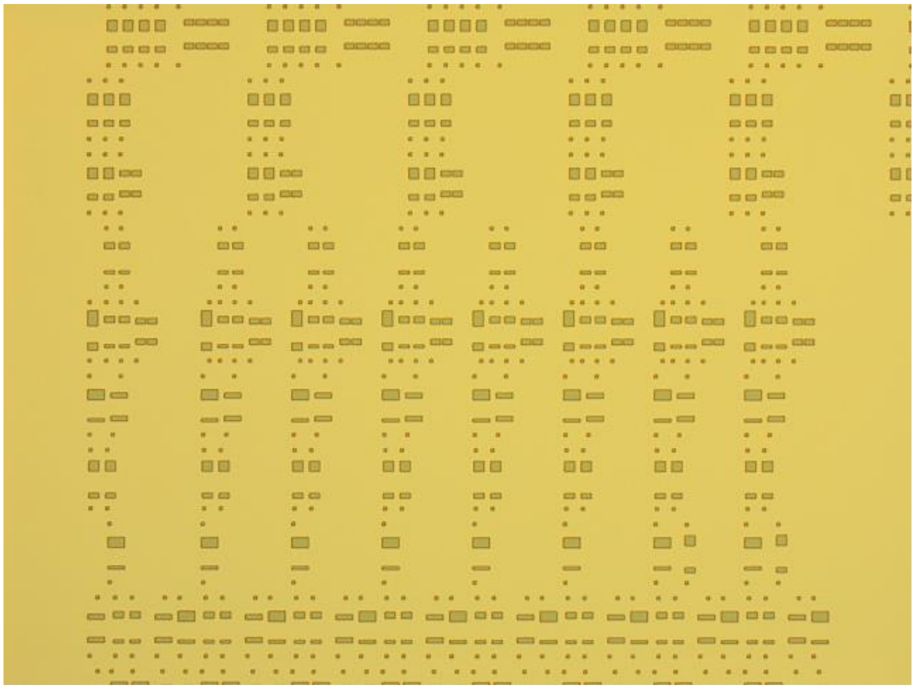


Figure 5.14 – ALU after active implantation

The following step of the process is field oxide (FOX) growth and etching. The result of this step is very similar to the previous image because it uses the same mask to isolate the non-active silicon surface from the outside. Figure 5.15 shows a magnified image of the ALU after this process.

Notice in Figure 5.15 that another dense structure is located on top of the bit slices. This is a level 2 multiplexer bus, called the "input selector multiplexer". This structure will be described in more detail in Chapter 6 and was already shown to be there in Figure 5.2 It's considered part of the data-path but not of the ALU and wasn't included in the standalone version described in subsection 5.1.3.

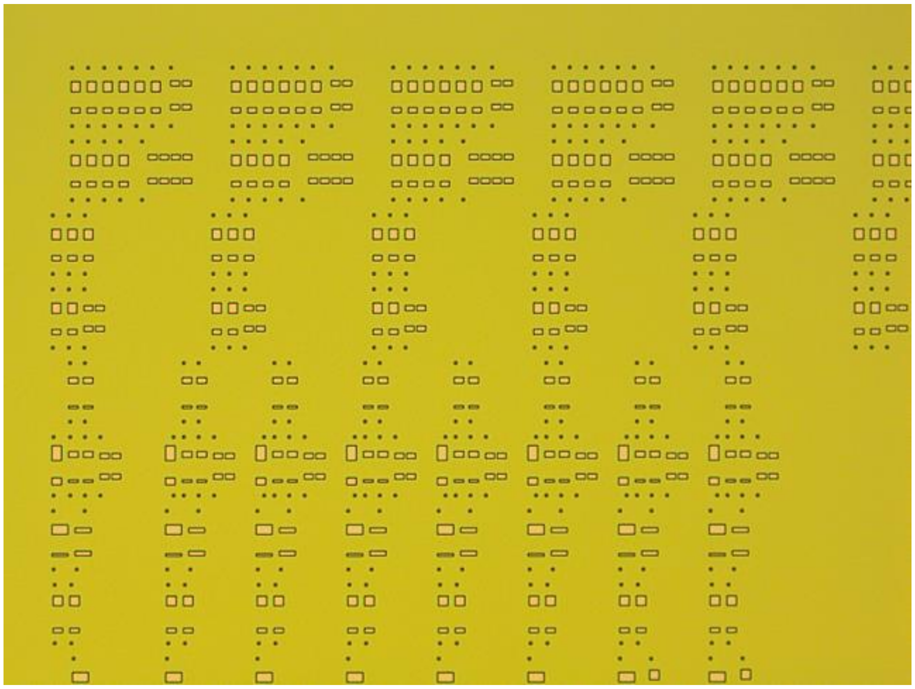


Figure 5.15 – ALU after FOX growth and etching.

After the FOX growth, a polysilicon layer is deposited on top of the wafer and etched. This material creates the gates of each transistor as well as a few connections inside each gate. Figure 5.16 shows the ALU after the step is complete. Some standard cell structures can already be recognized, such as the inverters and multiplexers.

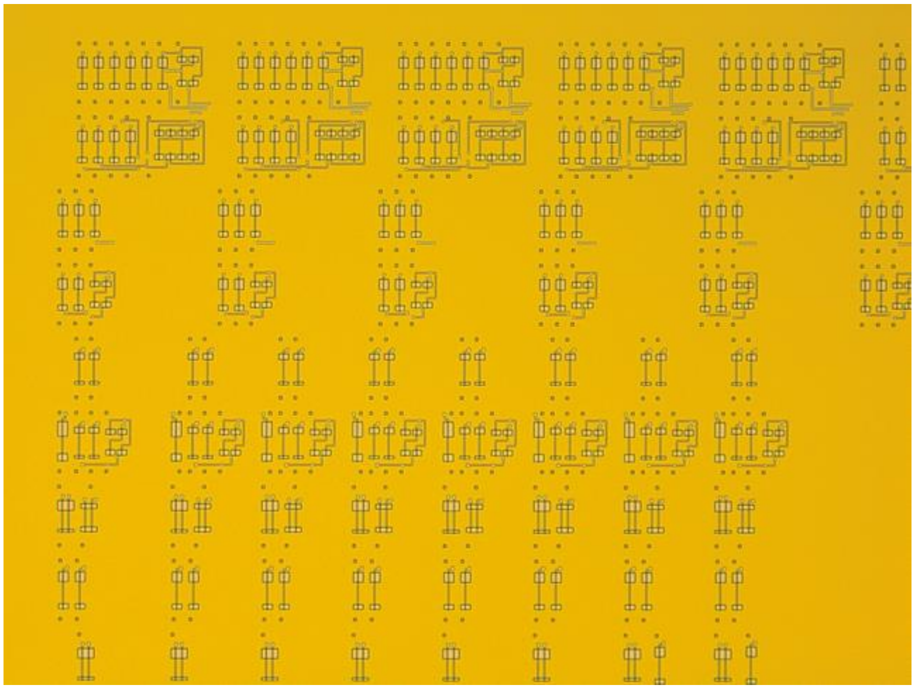
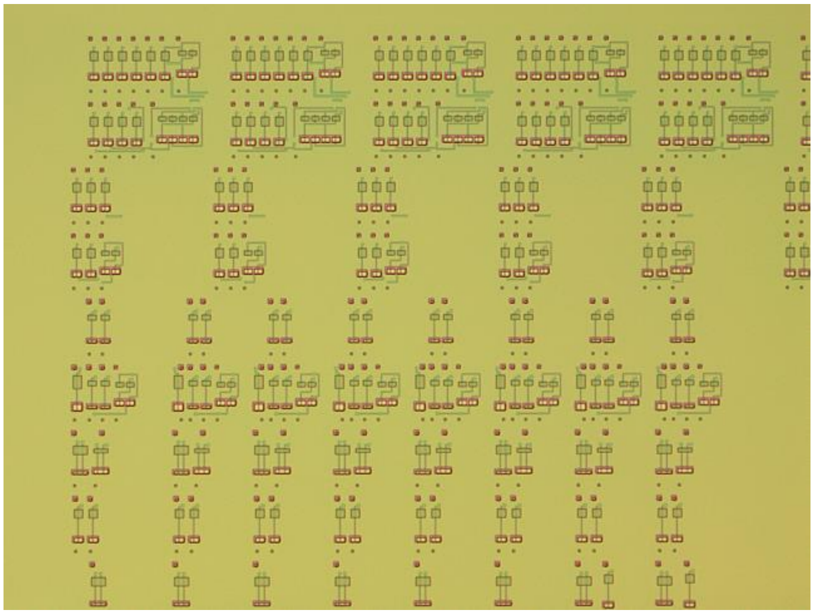
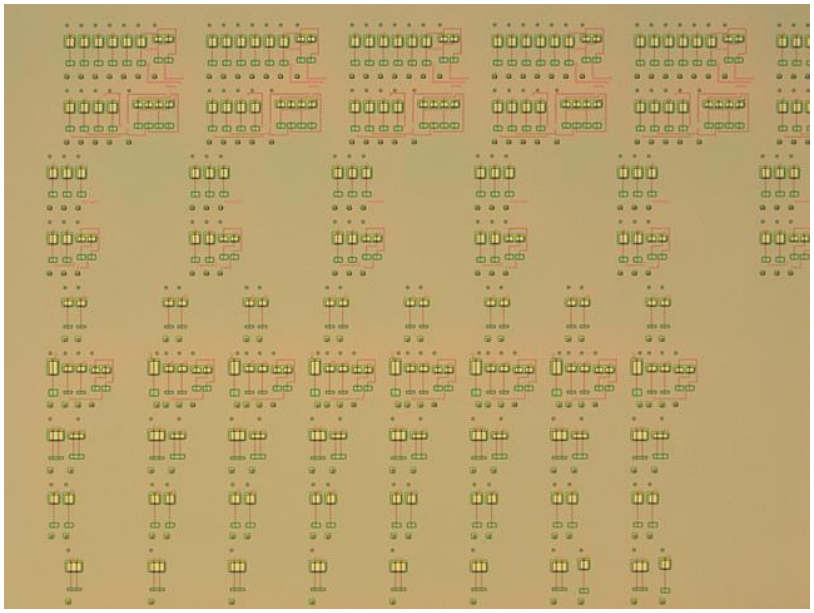


Figure 5.16 – ALU after polysilicon deposition and etching.

The next step in the process is doping of both NMOS and PMOS transistors also known as the N-select and P-select layers respectively. Figure 5.17 shows the aftermath of both processes. At this point transistors are already complete and functional but no metal pathways exist to connect them yet.



(a)



(b)

Figure 5.17 – ALU after semiconductor doping. (a) shows the N-select layer and (b) shows P-select.

5.4. – Chapter conclusion.

In this chapter, the 8-bit adiabatic ALU has been described in detail. The first section deals with the logic and physical design of the ALU. Design choices are described and the rationale behind them is discussed. Layout design was divided in “bit slices” that contain all relevant logic for a single pair of data bits, plus a separate “carry computation” sub-module that contains all logic needed for the simultaneous calculation of carries. These modules are described in subsections 5.1.1 and 5.1.2 respectively. Subsection 5.1.3 describes the modifications done on the ALU to fabricate a standalone version using the **CMOS 0.5μm** process.

The second section of the chapter **focused** on simulation tests of the ALU circuit both in adiabatic and standard CMOS mode. Test methodology is discussed including simulator, net list and test bench parameters. A transient analysis in adiabatic mode shows that the circuit operates as intended and displays the correct output.

Several more analyses were made to measure power dissipation of the circuit at different frequencies. A comparison between adiabatic and standard CMOS modes was made in terms of power consumption. **Simulation tests have shown** that the adiabatic mode consumes less power over a range of operation frequencies. In particular, by running the ALU at 10 MHz, power consumption of the adiabatic mode is 36.79 times lower as that of the standard CMOS mode. For higher frequencies, adiabatic dissipation approaches standard CMOS and **the ALU operating in adiabatic mode does not have any advantage.** Other issues in simulation are discussed as well.

The final section of the chapter shows a series of photographs on the fabrication process of the ALU. Although the process is not yet complete, the most important structures inside the ALU are visible.

Chapter 6 – Adiabatic 8-bit MIPS Microprocessor.

The previous two chapters presented an adiabatic standard cell library for fabrication in a CMOS process and an adiabatic 8-bit ALU design for use in a MIPS microprocessor. This chapter will focus on the overall architecture and design of the processor itself.

The first section is a structural review of the MIPS architecture. It aims to identify the major components of the processor and their relations. A block diagram is shown for the processor and each module is described individually.

The second section contains illustrations for standard cells produced for the MIPS that are not already present in the ALU cells described in Chapter 4. These cells mostly include static CMOS memory components or specialized circuits.

The third section shows the layout for the MIPS microprocessor. This layout was prepared for fabrication in the 1 μm CMOS process at University of Notre Dame and 0.5 μm technology of ON Semiconductor's C5. Each functional module is identified inside the layout. A microscope photograph of the fabrication progress in 1 μm CMOS technology is shown. This process is currently taking place at University of Notre Dame.

6.1 – Adiabatic MIPS architecture.

The microprocessor designed for this work pertains to the MIPS32 architecture. It is a RISC (Reduced Instruction Set Computing) processor with 32-bit instructions. However, the working registers, program counter and ALU are all 8-bit. All instructions take the same number of cycles and no pipelining capabilities exist.

This design follows the standard architecture adapted from the example in **[43]**. Modifications were made to allow Bennett-clocked adiabatic operation. All of the combinatorial logic was replaced by adiabatic logic gates. The memory elements such as the work registers remained traditional static CMOS because they need to be persistent between operation cycles. Additional memory elements were added at the beginning and end of each combinatorial data path to be able to save the outputs, which are valid during brief periods only, and keep the inputs constant during computing cycles.

Figure 6.1 shows a block diagram describing the MIPS processor architecture. The system is divided in 2 main separate blocks: the control unit, shown in blue, and the data path, in black.

The control unit contains a state machine and combinatorial logic to produce every control signal that the processor requires. In the adiabatic implementation this module was designed using Bennett-clocked gates. Special attention was paid to the control signal deadlines because in many cases they are needed at very specific times in the data path. A static register was added to store the 8-bit state and keep it persistent between operation cycles.

The data path contains both combinatorial and sequential elements and is used to perform the actual computing work as dictated by the instructions. Signals from the control unit direct the data flow by manipulating multiplexer selector bits and determine operation mode for the ALU, memory modules and registers.

The datapath was divided in 3 subunits, each separated by sequential elements at the beginning and end. Since sequential elements are persistent between operation cycles, each of the subunits has access to the entirety of the Bennett clock levels to perform whatever combinatorial logic function it requires.

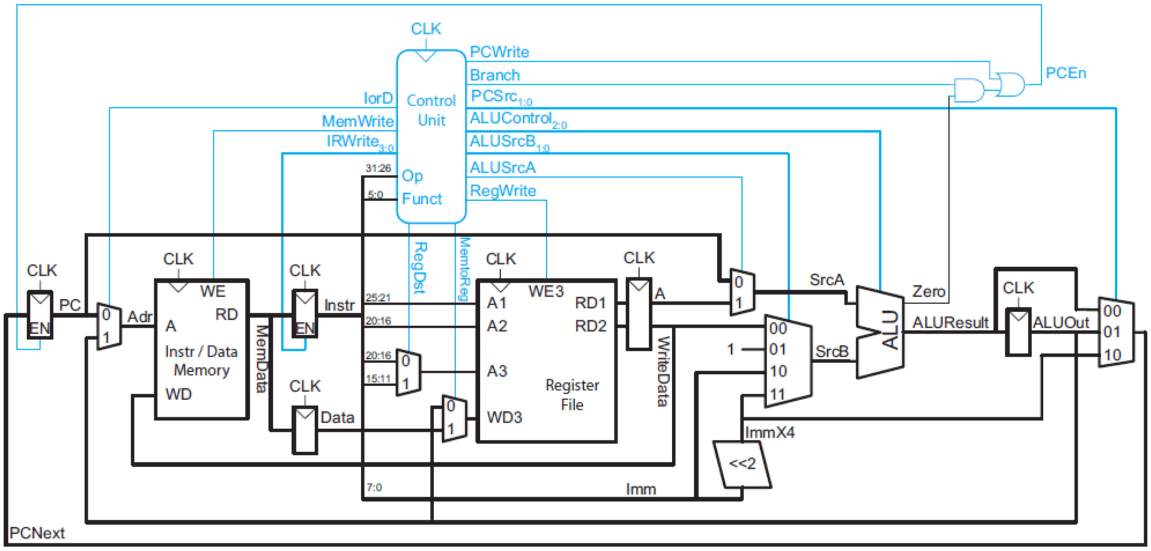


Figure 6.1 – Block diagram schematic of the MIPS architecture.

The first subunit is located in the left part of the block diagram. It is bounded by the PC and the ALU Output registers at the beginning and the Instruction and Data registers at the end. The MIPS architecture has a program data module located in this subunit, however it was decided to have this be an external chip in our implementation. Due to pin count constraints it was necessary to have a bidirectional read/write data bus instead of two separate buses as shown in the diagram. Figure 6.2 shows a wiring schematic that describes the hardware of the bidirectional bus.

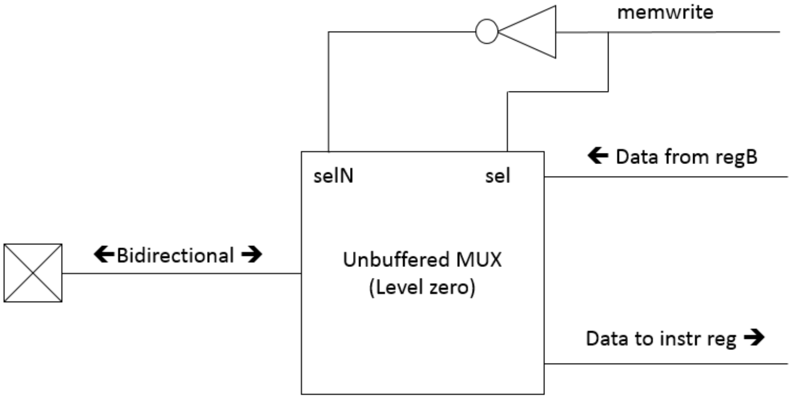


Figure 6.2 – Hardware to implement the bidirectional bus. The “memwrite” signal selects the directionality of the pin.

This subunit contains the adiabatic address select multiplexer, the bidirectional bus hardware for the external memory and the routing connections to the various registers.

The second subunit is confined by the Data and Instruction registers at the beginning and the A and B/WriteData registers at the end. It contains a couple of adiabatic multiplexers to direct data flow plus the register file that houses the work registers. This register file is an 8 by 8 SRAM (Static Random Access Memory) array. Each SRAM cell has a single write port and dual read ports. The decoder for the SRAM lines was implemented using adiabatic logic.

The third subunit shown at the rightmost part of the block diagram is limited by the A and B/WriteData registers at the beginning and the ALU Output and PC registers at the end. This contains the critical datapath, which determines the number of power clocks used in the implementation. It was optimized in logic design to employ the lowest possible number of clock phases, which was 12. The critical datapath is entirely adiabatic as it contains only combinatorial logic elements such as the ALU, zero detector and multiplexers.

6.2 – MIPS-specific layouts.

Some cells are used in the MIPS processor but not contained inside the ALU. These cells will be described in this section. Many of them are static sequential elements that operate in traditional CMOS mode and follow conventional designs.

6.2.1 – D-type Flip-Flops.

The D-type flip-flop was already introduced in Chapter 5 when describing the modifications done to the ALU for standalone fabrication. It follows a master-slave dual latch configuration that allows opportune use of clock phases to sample outputs while they are valid (every power clock energized) and update the output value of the flip-flop when it is valid to do so (every power clock relaxed). Each latch is built using an inverter feedback loop which can be broken using a transfer gate to create an open circuit. Doing so propagates the latch input signal forward.

Figure 6.3 shows the flip-flop layout design used both in the MIPS processor and the standalone ALU of Chapter 5. This particular layout has a reset and enable function as well because a few registers require those capabilities. The layout follows a conventional CMOS design.

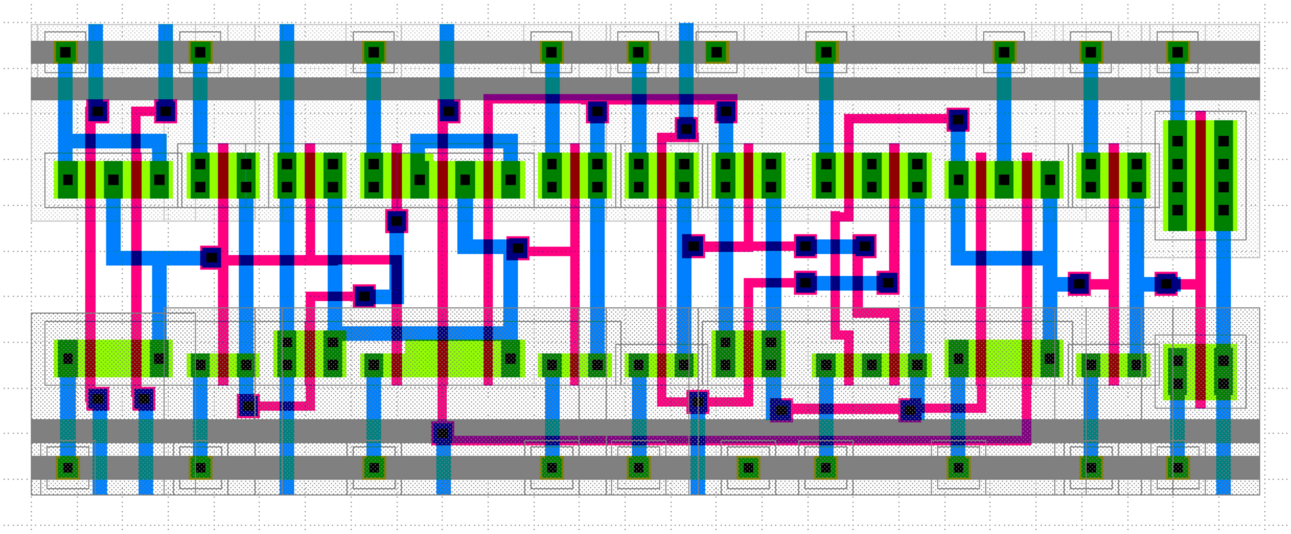


Figure 6.3 – D-type flip-flop layout design with reset and enable.

6.2.2 – SRAM cells.

SRAM cells were constructed following the 6T conventional CMOS design found in [Weste12]. Two independent read ports were built using 2 NMOS pass transistors for each, bringing the total transistor count per SRAM cell to a total of 10. Figure 6.4 shows the layout for a single SRAM cell.

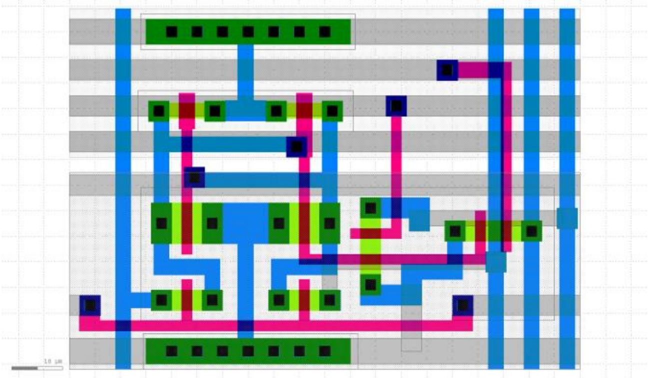


Figure 6.4 – SRAM Unit with single write and dual read ports.

Regarding the SRAM array, precharge was achieved by using a pseudo NMOS configuration and the sense amplifier is simply a push-pull amp (conventional CMOS inverter). The decoder uses adiabatic logic. Since it has the entirety of power clocks available for this and no critical deadline exists, the Bennett logic level in which it is placed is not important. Figure 6.5 shows the finalized register file layout including the SRAM array, precharge system, sense amplifiers and decoding logic.

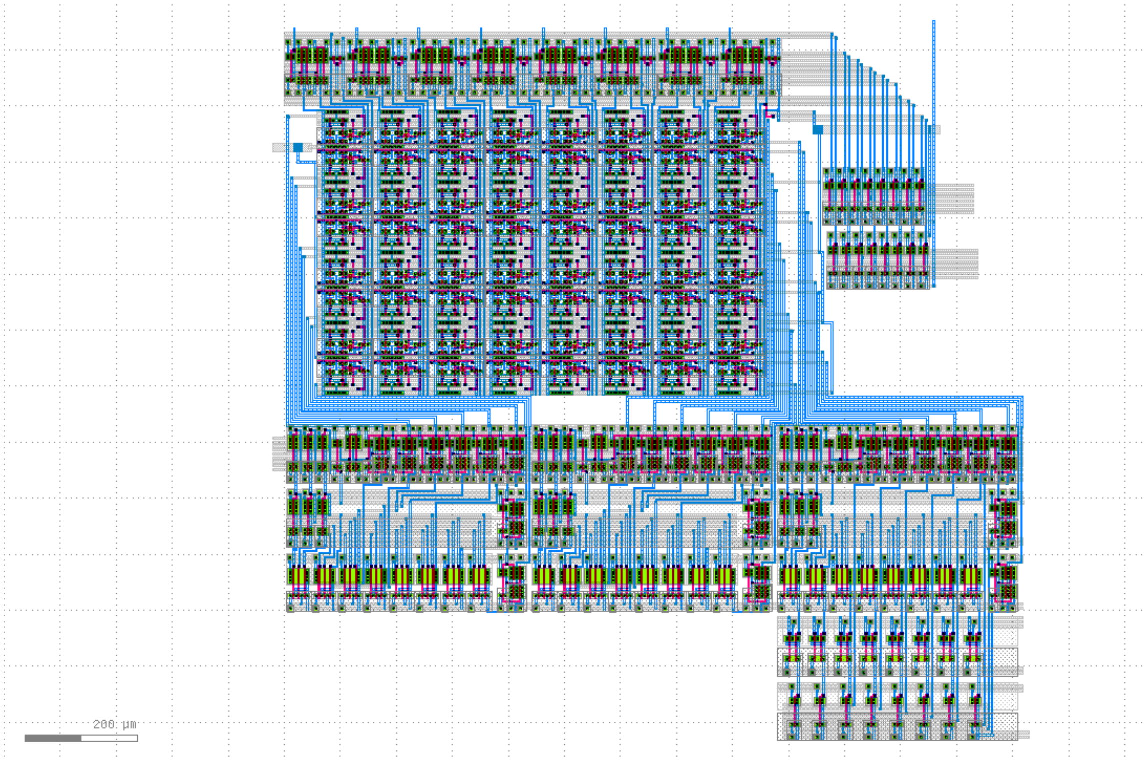


Figure 6.5 – Register file layout.

6.2.3 – Zero detector.

This is a composite logic gate that makes use of two 4-input NOR gates and a single 2-input NAND gate to detect whether the 8-bit ALU output is equal to zero. The output of this cell is low if the input is 0x00 and high otherwise. The cell represents the only use of the 4-input NOR gate in the whole design. Since it has a logical depth of two it uses two sets of power rails. Figure 6.6 shows the layout design for this cell.

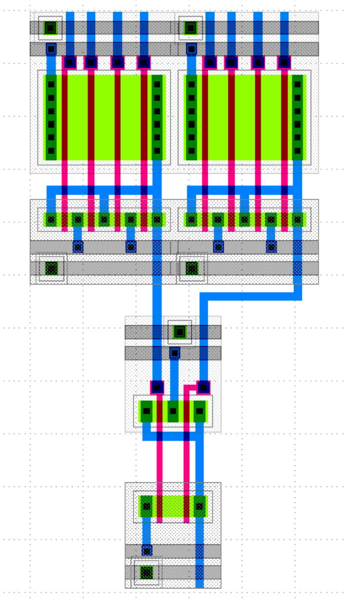


Figure 6.6 – Layout design of the zero detector.

6.2.4 – PC Enable Logic.

A custom complex CMOS gate was designed to compute the PC enable bit for the PC register. Although the MIPS block diagram calls for two combinatorial logic gates to compute this signal, it was found that the equation:

$$PC = Zero * Branch + PCWrite \tag{6.1}$$

can be implemented in a single Bennett level using transistor level complex CMOS logic as long as all the input signals are negated. Recall that the zero detector produces the inverse version of the Zero flag. Since the PC enable computing is the very last step of the critical datapath, the control unit has enough time to invert any signals as needed. Figure 6.7 shows the layout design of this cell.

6.2.5 – Buffering Multiplexers.

Chapter 4 presented a non-buffering family of TG-based multiplexers. For the cases where no time is available to calculate the selector signal complements a 2 Bennett level buffering multiplexer was designed using additional adiabatic inverters to generate all necessary signals. This implementation is needed whenever a multiplexer is located at the beginning of a data path, which happens at least once for every subunit. 2- and 4-input versions were prepared. Figure 6.8 shows the layout of the 4-input buffering multiplexer. The 2-input version is shown in Appendix B.

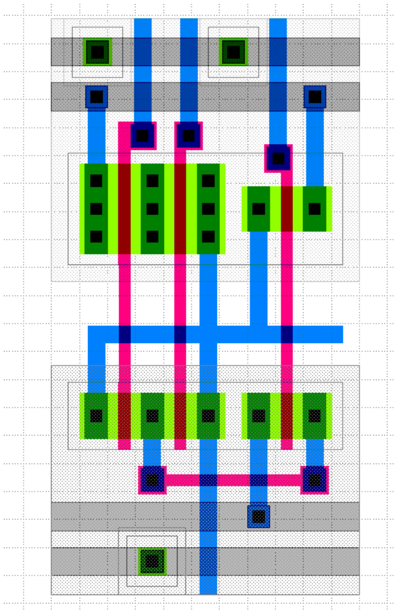


Figure 6.7 – Layout design of the complex CMOS PC enable computing logic cell.

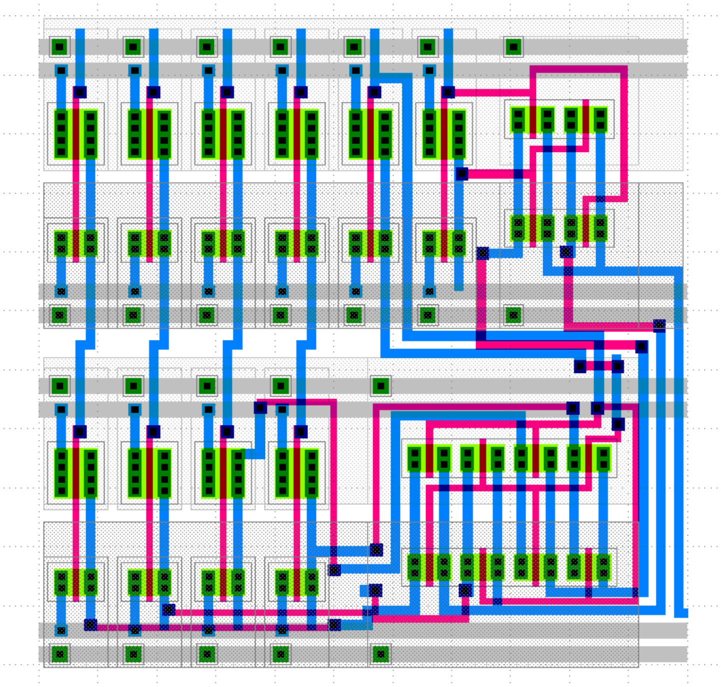


Figure 6.8 – Layout design of the 4-input buffering multiplexer.

6.3 – MIPS Processor Layout and Fabrication.

The MIPS processor was laid out by hand using the Tanner Tools L-Edit software. The layout was checked for DRC compliance with MOSIS rules and sent for fabrication. Figure 6.9 shows the layout for the processor sent for fabrication. Figure 6.10 shows an annotated version of the layout with every module highlighted and labeled.

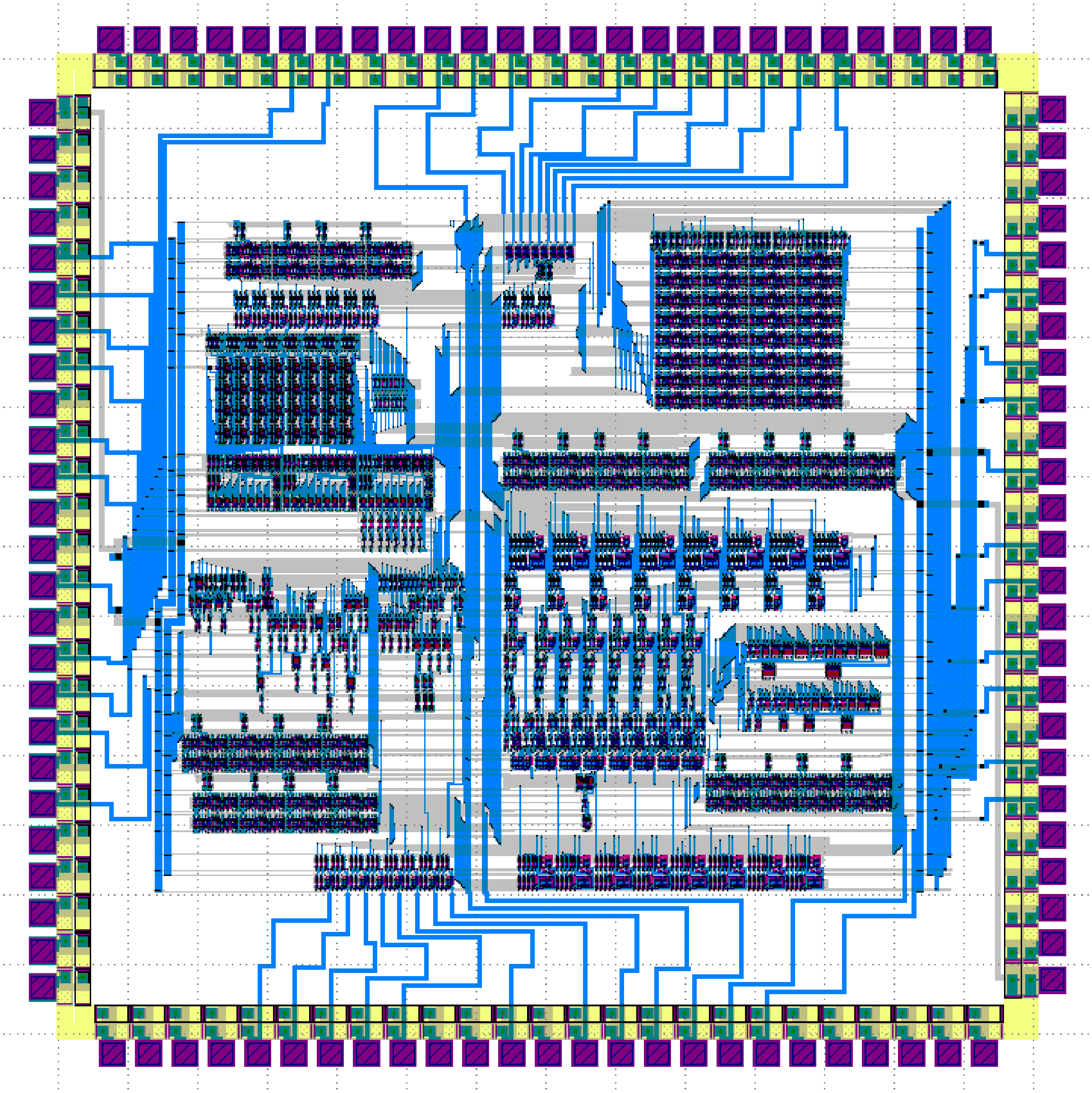


Figure 6.9 – MIPS processor layout sent for fabrication.

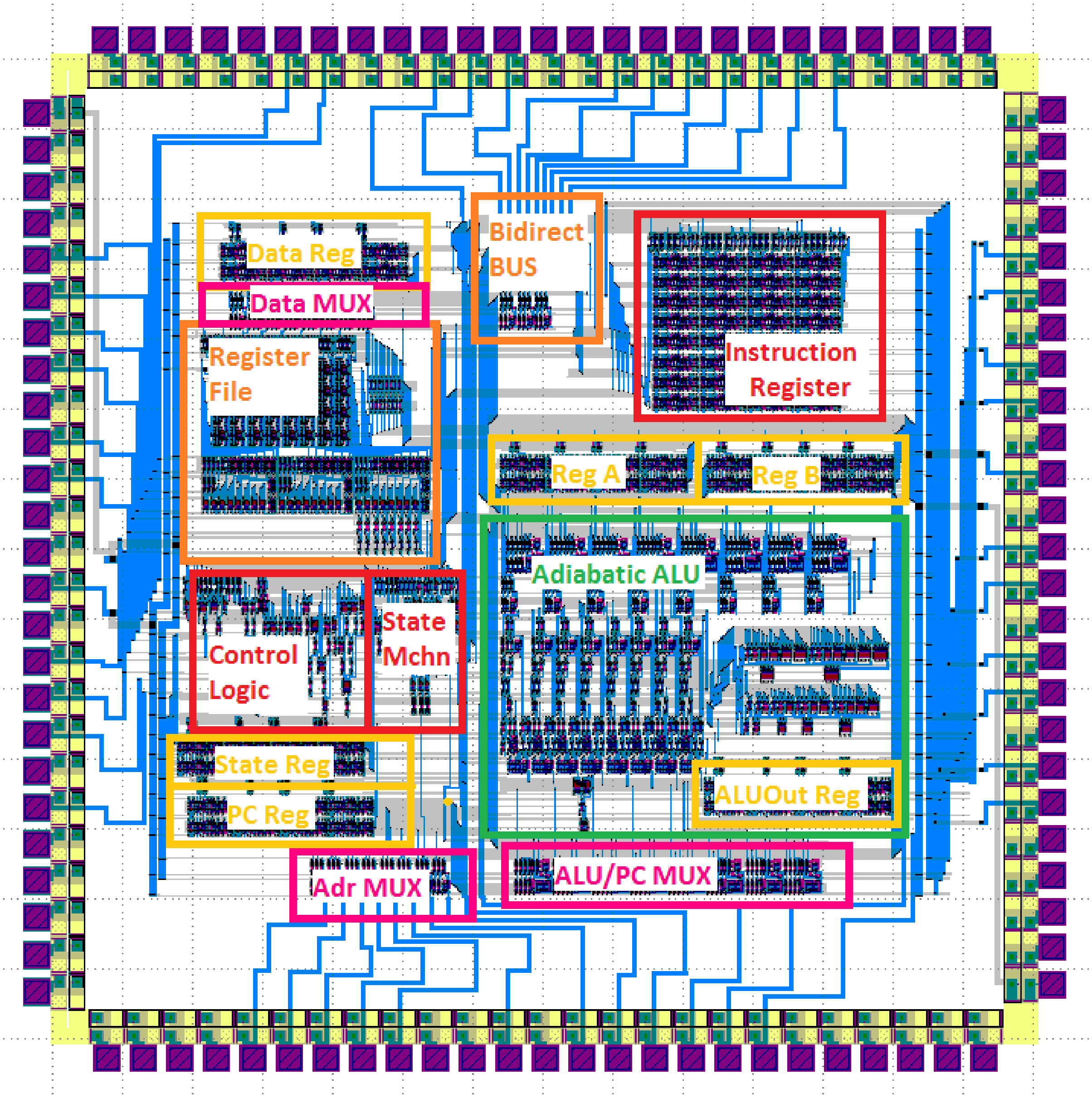


Figure 6.10 – Annotated layout showing internal modules.

The processor is being fabricated both by MOSIS in 0.5 μm technology and by the Notre Dame IC Fabrication Laboratory in 1 μm technology. Microscope photographs of the fabrication process are shown next for the stages that are currently complete.

Currently active implantation, field oxide growth and etching, polysilicon deposition and etching, and p- and n-type doping have been completed. Figure 6.11 shows the current progression on the fabrication process at Notre Dame. At this point the individual transistors are already complete and the disposition of the different functional modules can already be seen on the wafer.

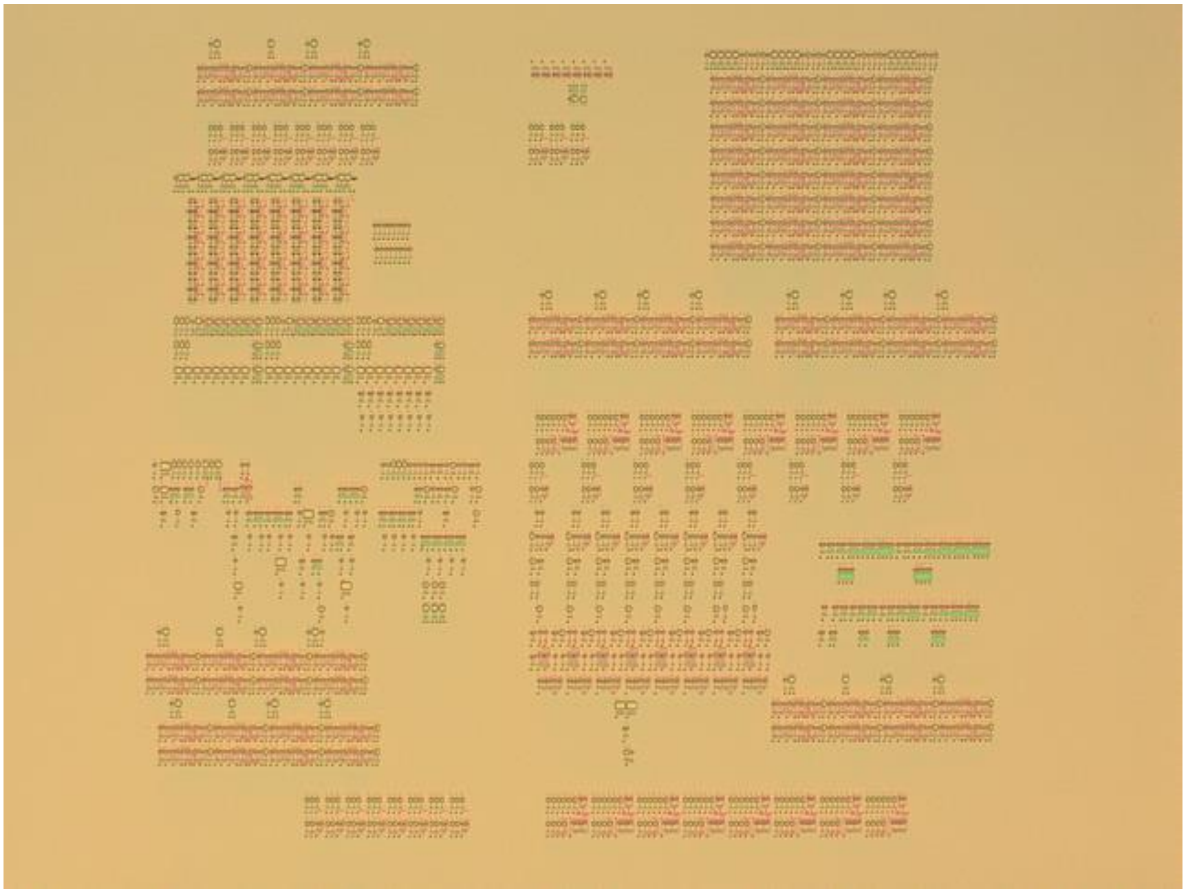


Figure 6.11 – MIPS processor die microscope scan showing current fabrication progress.

6.4 – Chapter conclusions.

This chapter showed the architecture of the designed adiabatic MIPS processor. The first section covered the design of the different modules contained within the architecture. It is clarified that not every section of the processor is fully adiabatic: only the combinatorial logic can perform nearly dissipationless operation. Sequential elements need to be static CMOS because they require data permanence between operation cycles.

The second section of the chapter takes a look at the specialized standard cells used by the MIPS processor but not by the adiabatic ALU. It is interesting to note that a custom complex CMOS logic gate was designed to reduce the logical depth of the system. Evaluating and recognizing opportunities like this is an interesting part of the process, not commonly explored in conventional CMOS and offers great potential for future improvements on the design.

The final section shows the layout for the MIPS microprocessor as well as a microscope photograph of the fabrication progress thus far.

Chapter 7. Conclusions.

After presenting and detailing the work in its entirety, this chapter will consolidate and give closure to the thesis. The objectives defined in the introductory chapter will be revisited and the conclusions gathered throughout the work will be summarized.

The first section of this chapter will take a look at the particular objectives defined in the introduction. For each objective, the relevant work will be summarized and the results will be succinctly presented. The degree in which each objective was completed will be analyzed.

The second section suggests some interesting derivative or subsequent research work that could be performed in the future. Even though the groundwork theory was postulated some 50 years ago, dissipationless computing is still an emergent topic. Many research questions arise from the implementation described in this work.

The final section of the chapter recalls the hypothesis description and creates a discussion on whether it is supported by the results of the research and how. This discussion concludes the thesis.

7.1 – Particular Objectives.

The introductory chapter defined the particular objectives of the thesis as follows:

9. To analyze the various schemes that perform dissipationless computing using CMOS technology.
10. To use the split-rail Bennett-clocked adiabatic technique to develop a prototype of an ALU device, considering its advantages over other adiabatic design methods.
11. To model the ALU device using HDL methodology and validate its behavioral performance.
12. To create a standard cell library in layout for the Bennett-clocked method, that contains every device needed for a practical implementation of an ALU.
13. To model the ALU using the ON Semiconductor C5 0.5 μm CMOS process and prepare the fabrication blueprints of the Integrated Circuit (IC).
14. To simulate the ALU using the design couple Electric VLSI-LTspice in order to:
    a. Verify the electrical detailed performance of the circuit.
    b. Validate the performance considering parasitic elements in the IC.
    c. Generate the physical layout and validate its compliance with the DRC (design rule check) and other standard fabrication rules in the C5 process.
    d. Send the IC to MOSIS for fabrication.
15. To develop a trade-off and comparative analysis between the adiabatic ALU and the standard CMOS ALU in terms of frequency and power dissipation.
16. To apply the design methodology for a MIPS processor and to send the adiabatic MIPS for fabrication using the ON Semiconductor C5 process.

This section will analyze each one and describe how the objective was reached and what conclusions were gathered from it.

7.1.1 – Objective One.

The first objective calls for a qualitative analysis of the various techniques found in literature that enable dissipationless computing through reversibility and adiabatic switching. Chapter 2 presented such an analysis.

The chapter first mentioned four main conditions for dissipationless computing: (1) existence of three logic states; (2) cells can receive information only in the relaxed state; (3) inputs must remain fixed for the entire computing cycle; and (4) gates must not permit backlashes. Two main branches of adiabatic switching circuits were explored: fully and quasi-adiabatic. Fully adiabatic implementations pass all four conditions while quasi-adiabatic may fail one or more while still reducing power dissipation.

Though many adiabatic switching implementations were examined, it was determined that the split-rail Bennett-clocked adiabatic methodology is the most interesting alternative for this work because: (1) it is capable of asymptotically zero power dissipation; and (2) it uses the conventional 1n1p topology of standard CMOS.

As an added objective, some complex implementations found in literature were described. For each of these implementations Chapter 2 covered the intended operation, selected adiabatic technique, type of test performed and results obtained. It was found that few fully adiabatic systems have been designed and out of those, only one uses the split-rail Bennett-clocked method. The value of this work is thus strongly asserted as it is, to the best of our knowledge, the largest design of its category realized thus far.

7.1.2 – Objective Two.

A prototype Bennett-clocked adiabatic ALU was developed through the course of this research work. This effort is presented along Chapters 3, 4 and 5 which show different steps of the design process.

Chapter 3 presented a proof of concept experiment which was necessary to understand and verify the principle of adiabatic switching. A set of HDL tools was described in this chapter. Although the chapter does not go into much detail regarding the HDL code describing the ALU, the tools described therein were invaluable in the logical testing and design of the ALU.

Chapter 4 opens with a discussion of the generalities of the physical design used in the adiabatic ALU. It then details all cells of a standard cell library that was custom built for this project. Each cell shows a transistor level electric design that incorporates the design philosophy of Bennett-clocked adiabatic systems.

Chapter 5 integrates this cell library into a complete functional prototype. Rather than just assembling cells together, many design questions were solved during this process. Synthesis is a crucial part of the development process of a CMOS circuit and it was essentially done by hand since no automatized tools exist yet for adiabatic logic families.

7.1.3 – Objective Three.

In Chapter 3, a System Verilog HDL custom script enabled a behavioral model of the adiabatic ALU and MIPS processor. Although Verilog does not natively support adiabatic logic, an extension was designed that lets the language handle logic states unique to split-rail Bennett-clocked adiabatic systems. Programming state definitions into Verilog involves modeling effort because they are a mathematical abstraction of a situation that occurs physically.

Behavioral models of each logic gate were also prepared. As with the definitions of the logic states exclusive to adiabatic operation, this work entails modeling of a physical realization into an abstraction. The models were then used to construct the ALU and simulate its behavior. A ModelSim plot is used to illustrate this behavior. It is shown that the device works as intended.

7.1.4 – Objective Four.

A complete standard cell library was created for the adiabatic ALU. The library includes every logic gate used in the device design. Both electric schematics and layouts are included for each gate. The electric diagrams are meant for simulation of electric behavior, while the layouts can be used for either simulation or fabrication.

Chapter 4 of this thesis describes the library in detail, including simulations of each component and discussion on the designs. For the most part, Bennett-clocked design resembles conventional CMOS circuits, but some of the cells deviate from this norm. In particular, cells with a logical depth larger than 1 (such as AND/OR cells) or TG-based cells need special considerations. How these considerations affect the cell design varies between implementations and the process is personalized in this sense. Information about the design choices made and specific rules such as distance between rows are detailed to allow further researchers to understand and use this cell library in their designs.

7.1.5 – Objective Five.

The fifth objective entails adapting the previous designs to a specific technology with a definite set of rules and restrictions. Chapter 5 describes the conversion process from a gate level model in HDL to a layout for fabrication. The logic model of Chapter 3 was replicated using the standard cell library of Chapter 4.

As shown in Chapter 5, the conversion process is not straightforward. Even with a description of all the necessary hardware as guideline and a model for each of the components, many choices such as placement and routing have demand attention. In the conversion process it was also necessary to think about the way Bennett-clocked adiabatic gates are connected to their power rails. Much of the placement was decided based on the power rails alone, but it was not the only factor.

As the chapter shows in the various figures it contains, the design began as modular with the power rails as the only guideline. It was decided early on that every cell on a given row had to have the same logic level to ease power clock routing. As the design advanced, however, two modifications were made to the original modular idea: (1) the carry-lookahead computation unit was split because it saturated its logic levels due to sheer number of components; and (2) a "bit slice" vertical directionality was adopted. It is noteworthy that modification (1) came about due to area utilization concerns, while modification (2) was a result of routing optimization. This process is usually performed by a synthesizer but since no automated software capable of Bennett-clocked adiabatic design exists yet, it had to be done by hand.

The final result for this objective is a blueprint layout suitable for fabrication through the MOSIS ON Semiconductor C5 0.5 μm CMOS technology. To prepare the layout for fabrication it was necessary to add sequential elements to the design and to handle a problem regarding number of pins available in the package. These issues were solved successfully and their solutions are also described in the chapter.

7.1.6 – Objective Six.

This objective has a wide scope with many smaller related goals. A simulation of the ALU design was made using the Electric VLSI and LTspice software. This test is shown and discussed in the second section of Chapter 5. The simulation has many purposes, reflected in the secondary goals of this objective: (a) verify electrical behavior; (b) validate performance considering parasitic effects; (c) verify that the layout complies with DRC and other fabrication rules; and (d) guarantee that the circuit can be sent for fabrication.

These goals are accomplished with the transient time simulation of the circuit. It is shown in the text that the circuit performs electrically as intended, given that it computes the correct value according to the inputs in the testbench. Conservative RC parasitic extraction was performed to show that the circuit performs correctly even with these additional elements. In fact it happened that including parasitics in the simulation diminished an unfavorable feedthrough effect caused by smaller parasitics that form part of the SPICE transistor model.

Electric VLSI is able to perform DRC (Design Rule Check), ERC (Electric Rule Check) and well tap analysis. All of these tools were ran to verify that the final layout meets the requirements for fabrication with MOSIS. After verifying all of these rules and validating that the circuit performs the correct operation, it can be guaranteed that the design will work after fabrication.

7.1.7 – Objective Seven.

One of the most important objectives is to verify that adiabatic operation offers considerable power savings because that is the main purpose of all adiabatic techniques. For a fully adiabatic method such as Bennett clocking, it is important to show that dissipation can in fact be arbitrarily low by adjusting the ratio of the operation frequency and the characteristic frequency of the circuit, defined by the reciprocate of the RC time constant.

A comparative analysis is made between the adiabatic ALU and an equivalent conventional CMOS device. The analysis measures total power dissipation of the device at different operating frequency points. The results of this analysis are shown and discussed in Chapter 5. The analysis demonstrates that the adiabatic ALU has much lower dissipation than conventional CMOS and that this value can be made arbitrarily by reducing the operating frequency.

For operation at 10 MHz, the adiabatic ALU dissipates up to 36 times less power than conventional CMOS. This means that power recovery is about 97%. From Chapter 2, it was found in literature that quasi-adiabatic devices can manage up to 75% recovery at comparable speeds. Hänninen et al. **[29]** add to this that device downscaling can support even faster speeds, up to a couple of GHz for 20 nm technology. It is clear that fully adiabatic techniques are highly desirable and that they might become even more useful if they are produced using state-of-the-art fabrication technology.

7.1.8 – Objective Eight.

The final objective defined for the thesis is the extension of the previous methodology to even larger scale integration with an adiabatic MIPS processor device. The device was designed and modeled following the same steps as the adiabatic ALU, usually in parallel. In fact, the ALU is meant for use as a component of the MIPS processor. Chapter 6 summarizes the design process for the adiabatic MIPS.

The processor has additional cells not used inside the ALU. These are described within Chapter 6 and mostly consist of sequential elements. The MIPS architecture has many static registers which cannot by definition operate in adiabatic mode. All of these cause irreversible power dissipation and lower the recovery rate for the system. Nonetheless they are necessary for correct operation of the processor.

The text describes each module of the processor architecture and includes information on the design process for 0.5 μm CMOS technology. The final product of this effort is a layout that is being sent to MOSIS for fabrication. As with the ALU, it was verified that the MIPS layout passes all the required rule checks for fabrication.

7.2 – Future work.

Although this research work aimed to cover a wide variety of objectives, new questions arose during its development that fall outside the scope of this thesis. This section will make some suggestions about possible work for future researchers in the line who may wish to continue with this project.

The direct continuation to this work would be an experimental test of both the adiabatic ALU and MIPS circuits. Both devices are currently undergoing fabrication but a test on silicon needs some non-trivial experiment design. Test cases and power clock generation are some of the matters that need to be solved before an experimental proof can be conducted.

Optimizations all around the designs are always possible. One that comes to mind is to better fit the processor architecture to Bennett-clocked implementation. It is probably possible to find an architecture other than MIPS that does not employ as many sequential elements and is more amicable to adiabatic operation. A scheme for managing and reducing sequential element power consumption can also be adopted to reduce dissipation in these static elements.

Other optimization opportunities within the circuits are in the combinatorial logic chains. By using complex CMOS logic some operations can be reduced in logic depth. This was done for the PC enable logic cell described in Chapter 6.

An interesting lead was found when comparing electric behavior simulations from schematic and from layout with conservative RC parasitics. It was found that for some cases, the parasitic elements would actually improve the output waveforms that were previously deformed by a feedthrough effect. It would be interesting to explore in more detail the role of parasitics in Bennett-clocked adiabatic circuits.

Higher operation speeds can be achieved by downscaling the devices. Another opportunity arises in ensuring this downscaling process is smooth. CMOS in general is a scalable technology, but very small feature lengths require some special considerations regarding design rules. Furthermore, the power clock times need to be adjusted to fit the faster operating frequency.

7.3 – Closing Statements.

The hypothesis enunciated in the introductory chapter says is that it is possible to create ultra-low power LSI ICs using reversible computing and adiabatic charging techniques. It also states that the devices will perform the same useful computing work as their traditional CMOS counterparts but dissipate significantly less energy. It has been sufficiently demonstrated that this is true for both adiabatic ALU and MIPS processor.

The first statement of the hypothesis is proven true by the fact that a design was found to be feasible using the Bennett-clocked adiabatic technique. As fully adiabatic, this technique employs both reversible computing and adiabatic charging. The methodology was robust enough to permit design of LSI systems. This can be extended to even larger scale integration.

The second statement of the hypothesis is shown to be true by the electrical behavior test and comparative analysis performed in simulation for the adiabatic ALU. The results show that: (1) the adiabatic system completes the intended operation without fail or error when an electrical behavior simulation was performed; and (2) it compares very favorably in terms of power dissipation against an equivalent conventional CMOS device for a wide range of frequencies.

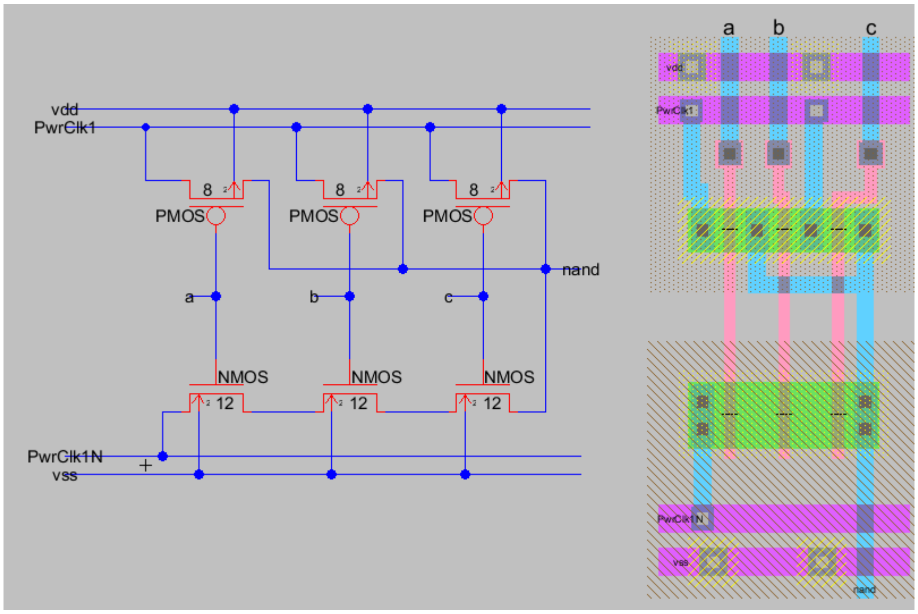
While it might be true that considerable challenges still exist to the implementation of adiabatic circuits, they promise extraordinary benefits. This work shows that the potential exists for creation of very complex devices that dissipate a tiny fraction of the power that conventional systems consume.

Appendix A – Acronyms

CMOS - Complementary Metal-Oxide Semiconductor
IC - Integrated Circuit
LP - Landauer Principle
ALU - Arithmetic Logic Unit
MIPS - Microprocessor without Interlocked Pipeline Stages
SPICE - Simulation Program with Integrated Circuit Emphasis
HDL - Hardware Description Language
ECRL - Efficient Charge Recovery Logic
HEERL - High Efficient Energy Recovery Logic
RISC - Reduced Instruction Set Computing
SRAM - Static Random Access Memory
FOX - Field Oxide

Appendix B – Additional layout realizations.

3-input NAND



(a) (b)

Figure B.1 – 3-input NAND. (a) Schematic. (b) Layout

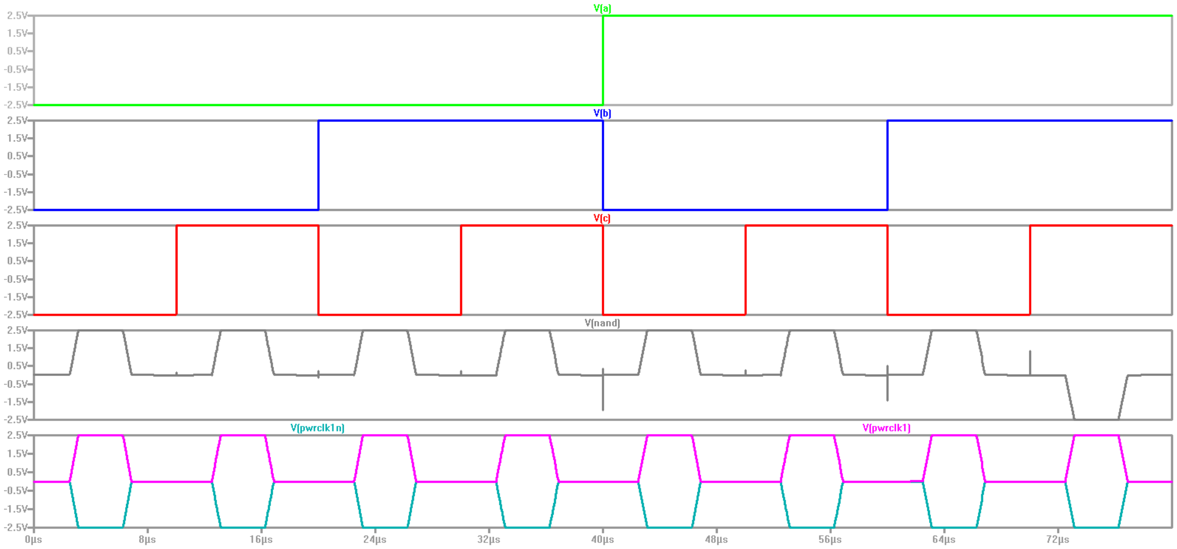
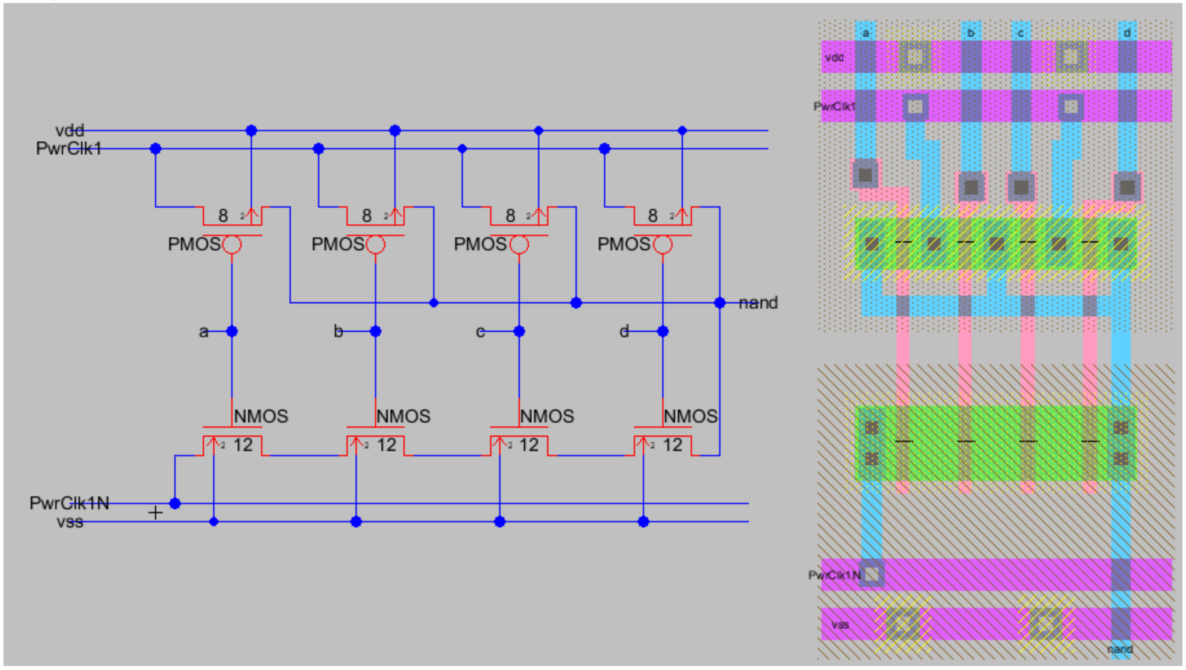


Figure B.2 – Simulation showing behavior of the 3-input NAND gate.

4-input NAND



(a) (b)

Figure B.3 – 4-input NAND. (a) Schematic. (b) Layout

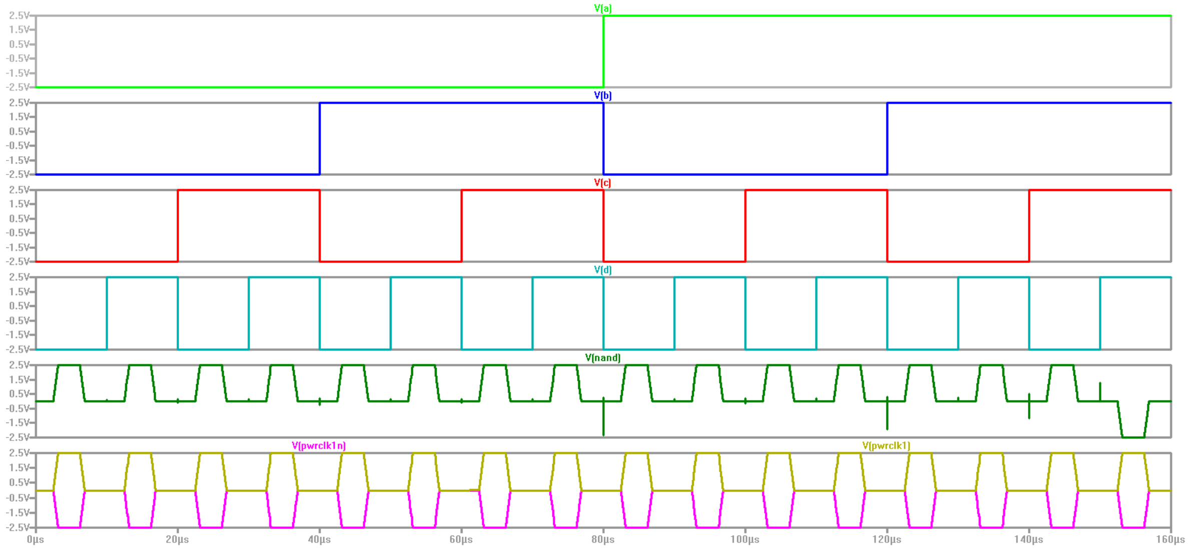
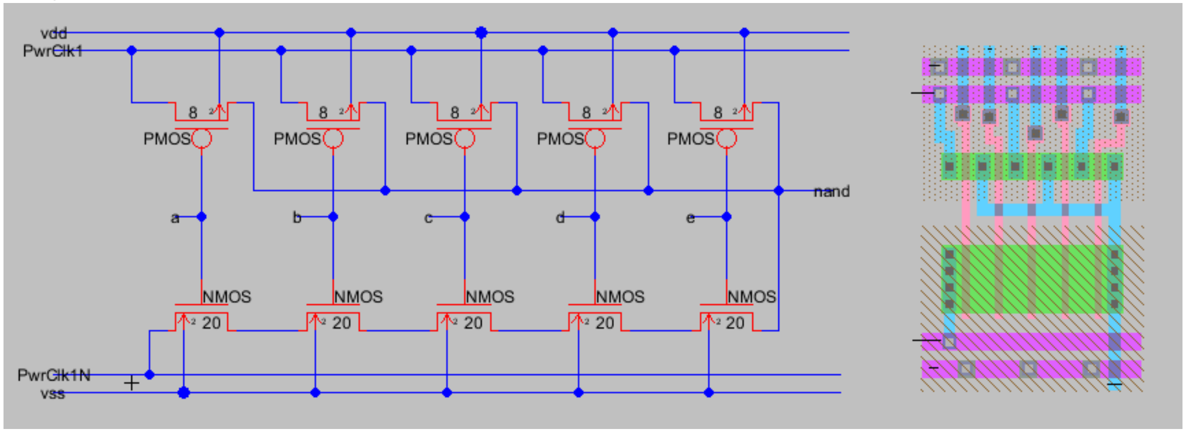


Figure B.4 – Simulation showing behavior of the 4-input NAND gate.

5-input NAND



(a) (b)

Figure B.5 – 5-input NAND. (a) Schematic. (b) Layout

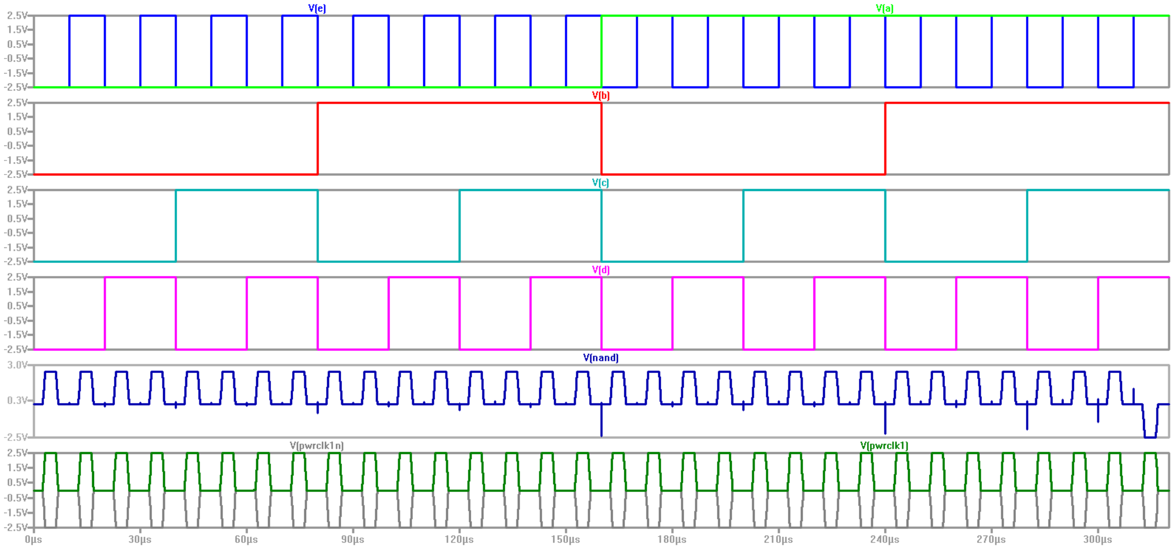
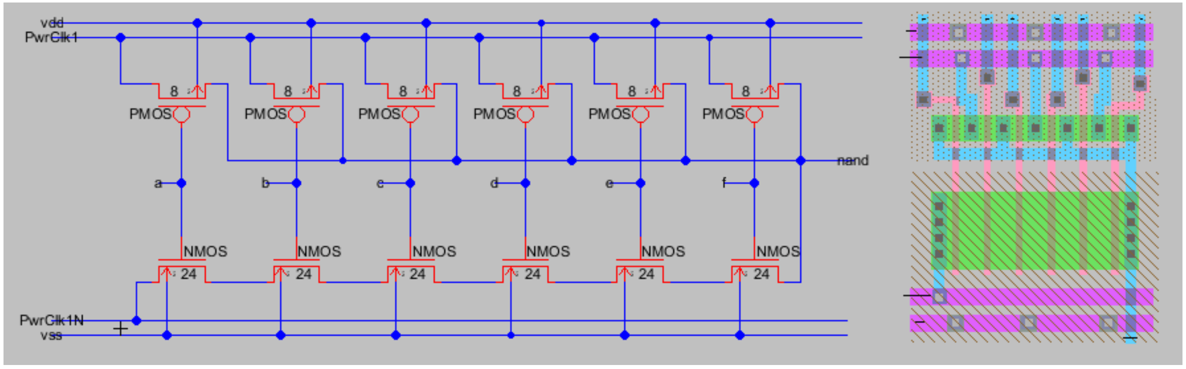


Figure B.6 – Simulation showing behavior of the 5-input NAND gate.

6-input NAND



(a) (b)

Figure B.7 – 6-input NAND. (a) Schematic. (b) Layout

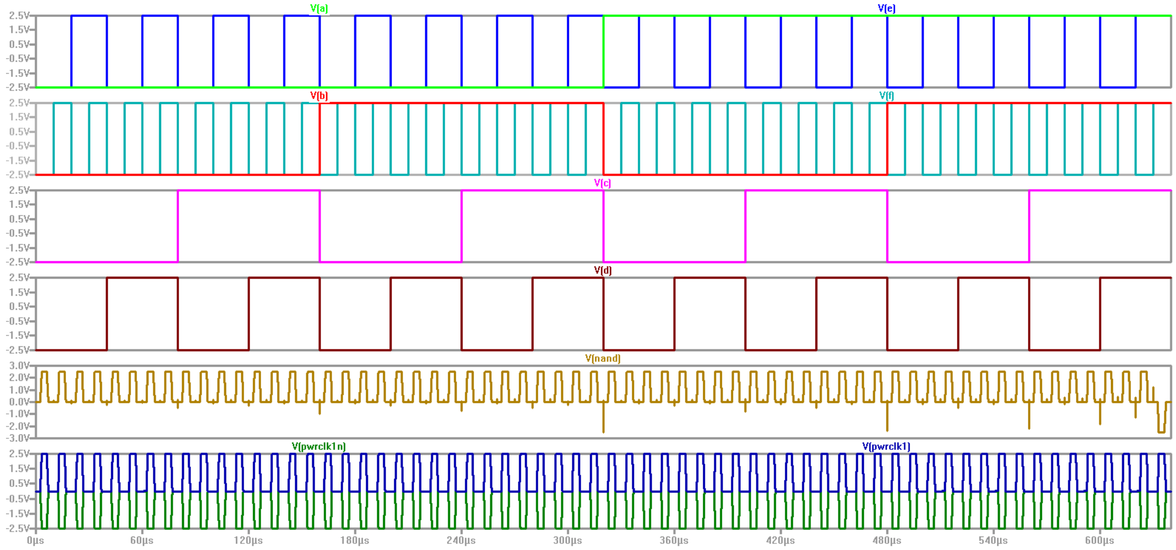
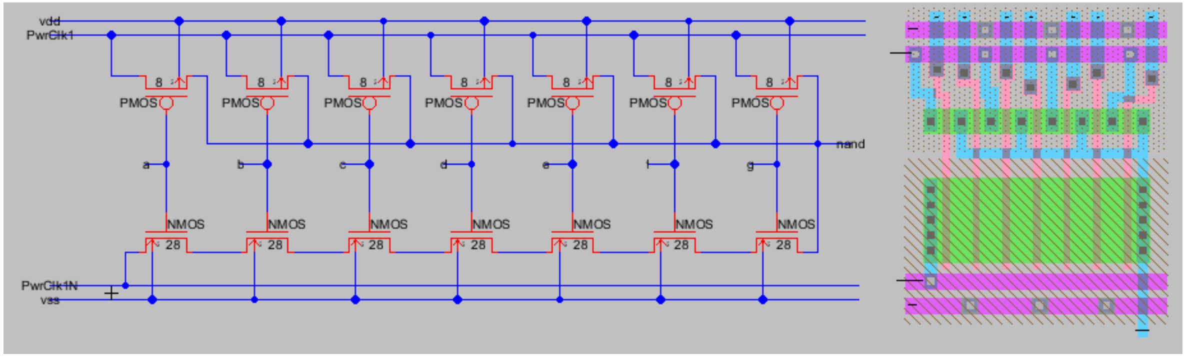


Figure B.8 – Simulation showing behavior of the 6-input NAND gate.

7-input NAND



(a) (b)

Figure B.9 – 7-input NAND. (a) Schematic. (b) Layout

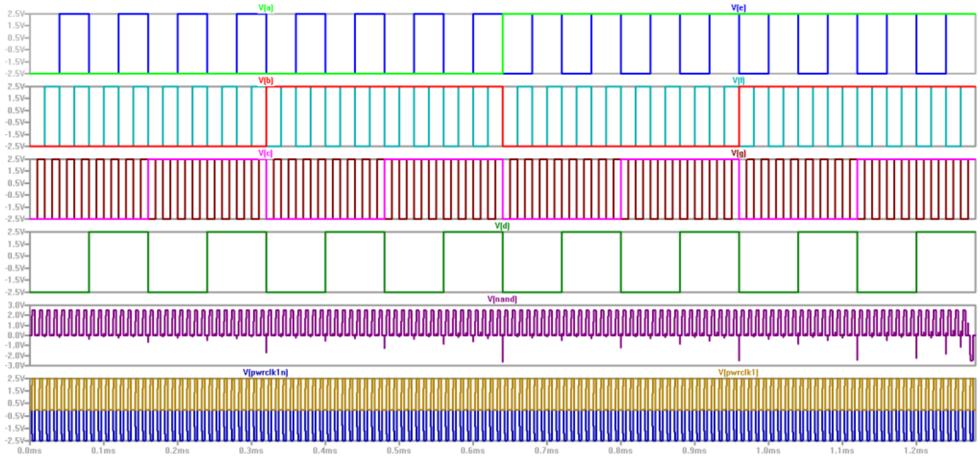
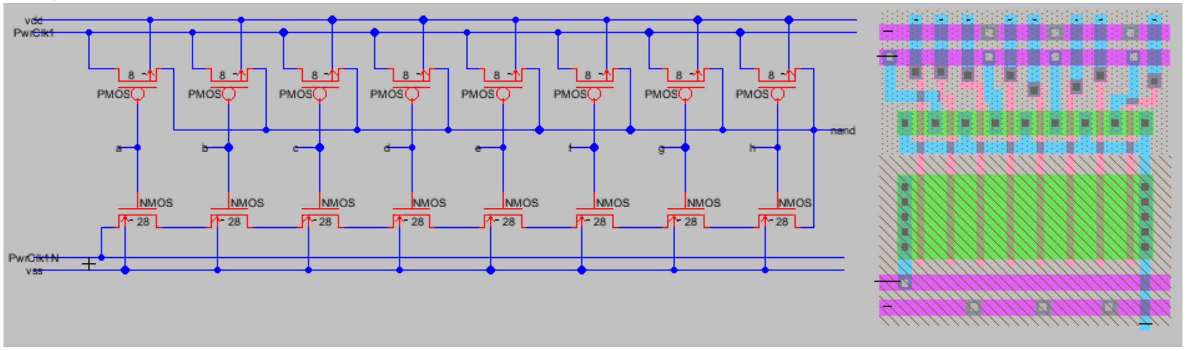


Figure B.10 – Simulation showing behavior of the 7-input NAND gate.

8-input NAND



(a) (b)

Figure B.11 – 8-input NAND. (a) Schematic. (b) Layout

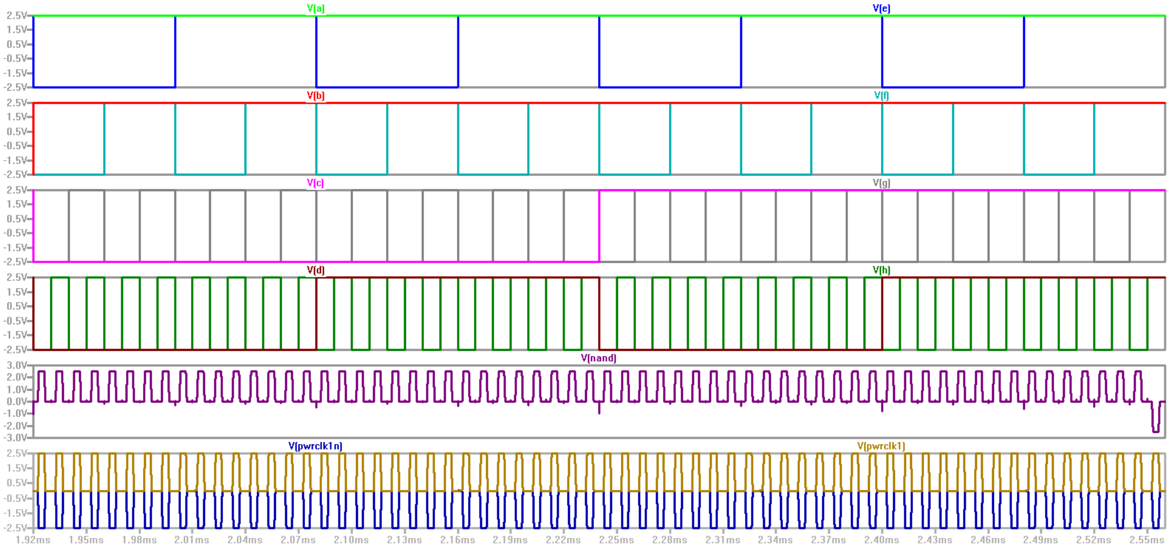


Figure B.12 – Simulation showing behavior of the 7-input NAND gate.

2-input AND

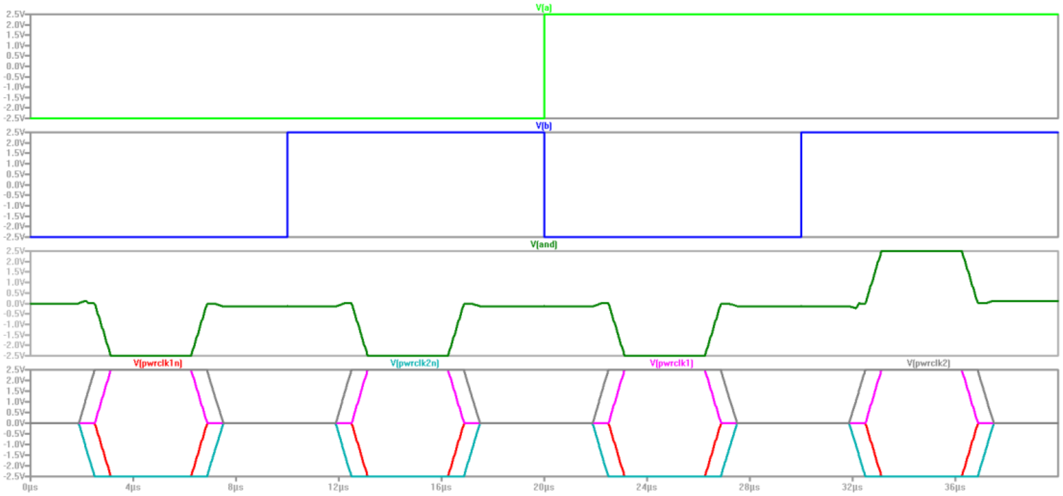


Figure B.13 – Simulation showing behavior of the 2-input AND gate.

2-input OR

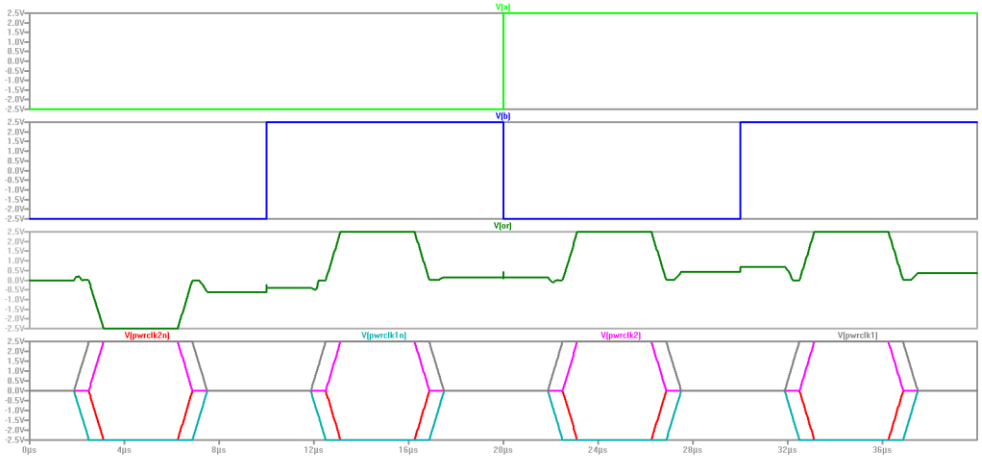


Figure B.14 – Simulation showing behavior of the 2-input OR gate.

2-to-1 level 0 multiplexer.

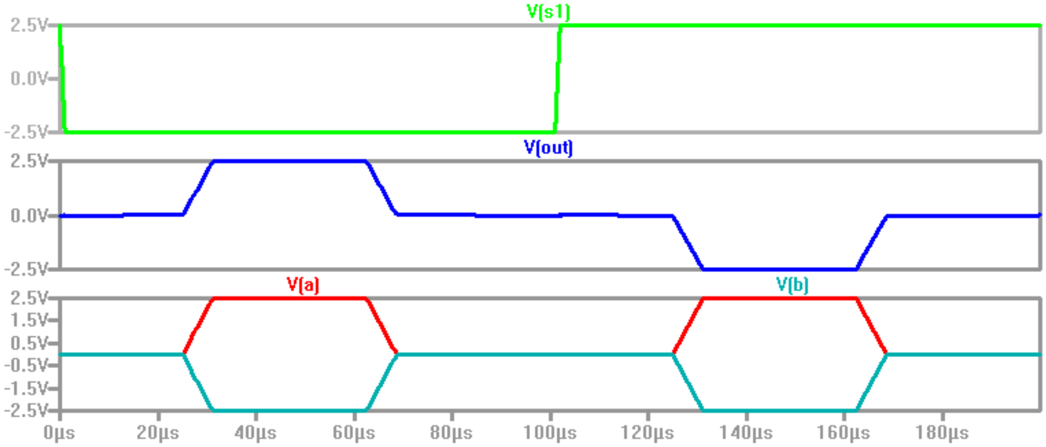


Figure B.15 – Transient simulation of the 2-to-1 level 0 multiplexer device.

2-to-1 level 2 multiplexer.

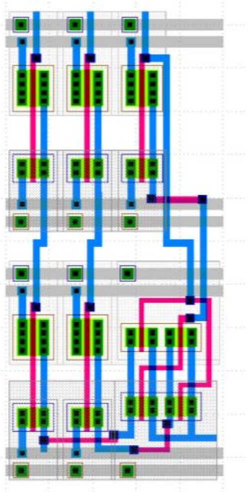


Figure B.16 – Layout of the 2-to-1 level 2 multiplexer.

Appendix C - SPICE MOSIS Transistor Models and Technology Parameters.

```
                           MOSIS WAFER ELECTRICAL TESTS

          RUN: V3BM                                        VENDOR: AMIS (ON-SEMI)
     TECHNOLOGY: SCN05                                FEATURE SIZE: 0.5 microns
                                   Run type: SHR


INTRODUCTION: This report contains the lot average results obtained by MOSIS
              from measurements of MOSIS test structures on each wafer of
              this fabrication lot. SPICE parameters obtained from similar
              measurements on a selected wafer are also attached.

COMMENTS: SMSCN3ME06_ON-SEMI


TRANSISTOR PARAMETERS     W/L      N-CHANNEL P-CHANNEL  UNITS

 MINIMUM                  3.0/0.6
  Vth                                   0.78     -0.88  volts

 SHORT                    20.0/0.6
  Idss                                470      -265     uA/um
  Vth                                   0.67     -0.86  volts
  Vpt                                  12.5     -11.9   volts

 WIDE                     20.0/0.6
  Ids0                                < 2.5     < 2.5   pA/um

 LARGE                    50/50
  Vth                                   0.68     -0.93  volts
  Vjbkd                                10.7     -11.8   volts
  Ijlk                                218.6     <50.0   pA
  Gamma                                 0.49      0.56  V^0.5

 K' (Uo*Cox/2)                         57.3     -18.9   uA/V^2
 Low-field Mobility                   464.63    153.26  cm^2/V*s

COMMENTS: Poly bias varies with design technology. To account for mask
           bias use the appropriate value for the parameter XL in your
           SPICE model card.
                       Design Technology                   XL (um)  XW (um)
                       -----------------                   -------  ------
                       SCMOS_SUBM (lambda=0.30)             0.10     0.00
                       SCMOS (lambda=0.35)                  0.00     0.20


FOX TRANSISTORS           GATE      N+ACTIVE  P+ACTIVE  UNITS
 Vth                      Poly        >15.0    <-15.0   volts


PROCESS PARAMETERS     N+    P+   N_W_U    POLY  PLY2_HR  POLY2   M1   UNITS
 Sheet Resistance      79.4 101.5   801.0  23.0  1025     41.0   0.09  ohms/sq
 Contact Resistance    58.0 150.7          16.6           26.9         ohms
 Gate Oxide Thickness 140                                              angstrom

PROCESS PARAMETERS             M2        M3    N_W      UNITS
 Sheet Resistance             0.10      0.05    795     ohms/sq
 Contact Resistance           0.84      0.85            ohms


CAPACITANCE PARAMETERS   N+   P+   POLY   POLY2   M1   M2   M3   N_W    UNITS
 Area (substrate)       420  719    86            28   12    8    91    aF/um^2
 Area (N+active)                  2472            36   17   12          aF/um^2
 Area (P+active)                  2360                                  aF/um^2
 Area (poly)                              893     63   16    9          aF/um^2
 Area (poly2)                                     56                    aF/um^2
 Area (metal1)                                         33   13          aF/um^2
 Area (metal2)                                              32          aF/um^2
 Fringe (substrate)     367  249                  51   34   26          aF/um
 Fringe (poly)                                    69   39   28          aF/um
 Fringe (metal1)                                       47   33          aF/um
 Fringe (metal2)                                            52          aF/um
 Overlap (N+active)              188                                 aF/um
 Overlap (P+active)              244                                 aF/um


CIRCUIT PARAMETERS                             UNITS
 Inverters                   K
  Vinv                      1.0       2.06  volts
  Vinv                      1.5       2.32  volts
  Vol (100 uA)              2.0       0.46  volts
  Voh (100 uA)              2.0       4.50  volts
  Vinv                      2.0       2.51  volts
  Gain                      2.0     -17.04
 Ring Oscillator Freq.
  DIV256 (31-stg,5.0V)              102.67  MHz
  D256_WIDE (31-stg,5.0V)           159.81  MHz
 Ring Oscillator Power
  DIV256 (31-stg,5.0V)                0.48  uW/MHz/gate
  D256_WIDE (31-stg,5.0V)             1.00  uW/MHz/gate

COMMENTS: SUBMICRON


 V3BM SPICE BSIM3 VERSION 3.1 PARAMETERS

SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8

* DATE: May  1/14
* LOT: v3bm                  WAF: 1003
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                LEVEL   = 49
+VERSION = 3.1            TNOM    = 27             TOX     = 1.4E-8
+XJ      = 1.5E-7         NCH     = 1.7E17         VTH0    = 0.6155718
+K1      = 0.927165       K2      = -0.1083911     K3      = 22.8103193
+K3B     = -9.5490578     W0      = 1.011931E-8    NLX     = 2.526408E-9
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 0.8806411      DVT1    = 0.3803297      DVT2    = -0.2060183
+U0      = 455.1710634    UA      = 1.041298E-13   UB      = 1.579609E-18
+UC      = 1.13123E-11    VSAT    = 1.983149E5     A0      = 0.6022739
+AGS     = 0.1382025      B0      = 1.831247E-6    B1      = 5E-6
+KETA    = -4.902055E-3   A1      = 5.595408E-5    A2      = 0.3
+RDSW    = 1.062E3        PRWG    = 0.0798664      PRWB    = -0.0174716
+WR      = 1              WINT    = 2.216754E-7    LINT    = 8.043861E-8
+XL      = 1E-7           XW      = 0              DWG     = -8.322467E-9
+DWB     = 4.618635E-8    VOFF    = -1.506501E-4   NFACTOR = 1.1555205
+CIT     = 0              CDSC    = 2.4E-4         CDSCD   = 0
+CDSCB   = 0              ETA0    = 2.035354E-3    ETAB    = -4.669429E-4
+DSUB    = 0.0702864      PCLM    = 2.0810693      PDIBLC1 = 8.593454E-6
+PDIBLC2 = 2.091439E-3    PDIBLCB = 0.1197245      DROUT   = 0
+PSCBE1  = 2.246943E8     PSCBE2  = 9.969607E-8    PVAG    = 0
+DELTA   = 0.01           RSH     = 79.4           MOBMOD  = 1
+PRT     = 0              UTE     = -1.5           KT1     = -0.11
+KT1L    = 0              KT2     = 0.022          UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11       AT      = 3.3E4
+WL      = 0              WLN     = 1              WW      = 0
+WWN     = 1              WWL     = 0              LL      = 0
+LLN     = 1              LW      = 0              LWN     = 1
+LWL     = 0              CAPMOD  = 2              XPART   = 0.5
+CGDO    = 1.88E-10       CGSO    = 1.88E-10       CGBO    = 1E-9
+CJ      = 4.176198E-4    PB      = 0.8350587      MJ      = 0.4275656
+CJSW    = 3.668183E-10   PBSW    = 0.8            MJSW    = 0.2085947
+CJSWG   = 1.64E-10       PBSWG   = 0.8            MJSWG   = 0.2019414
+CF      = 0              PVTH0   = -5.305536E-3   PRDSW   = 207.8016369
+PK2     = -0.0810173     WKETA   = -1.413082E-3   LKETA   = -2.178234E-3    )
*
.MODEL CMOSP PMOS (                                LEVEL   = 49
+VERSION = 3.1            TNOM    = 27             TOX     = 1.4E-8
+XJ      = 1.5E-7         NCH     = 1.7E17         VTH0    = -0.9152268
+K1      = 0.553472       K2      = 7.871921E-3    K3      = 1.5
+K3B     = 0.0273558      W0      = 3.3916E-7      NLX     = 9.640668E-9
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 0.818189       DVT1    = 0.3110325      DVT2    = -0.0916704
+U0      = 201.3603195    UA      = 2.48572E-9     UB      = 1.005454E-21
+UC      = -1E-10         VSAT    = 1.047533E5     A0      = 0.7091485
+AGS     = 0.1292184      B0      = 8.124014E-7    B1      = 1.081483E-8
+KETA    = -4.865785E-3   A1      = 3.62251E-4     A2      = 0.6482228
+RDSW    = 2.946814E3     PRWG    = -0.0219441     PRWB    = -0.0584559
+WR      = 1              WINT    = 2.208839E-7    LINT    = 1.111912E-7
+XL      = 1E-7           XW      = 0              DWG     = -2.726688E-9
+DWB     = -2.015846E-8   VOFF    = -0.0692697     NFACTOR = 0.7808767
+CIT     = 0              CDSC    = 2.4E-4         CDSCD   = 0
+CDSCB   = 0              ETA0    = 4.415482E-3    ETAB    = -0.04
+DSUB    = 0.85           PCLM    = 2.5353837      PDIBLC1 = 0.0622214
+PDIBLC2 = 4.508339E-3    PDIBLCB = -0.0220558     DROUT   = 0.2510332
+PSCBE1  = 7.325064E9     PSCBE2  = 7.921606E-9    PVAG    = 4.557354E-4
+DELTA   = 0.01           RSH     = 101.5          MOBMOD  = 1
+PRT     = 0              UTE     = -1.5           KT1     = -0.11
+KT1L    = 0              KT2     = 0.022          UA1     = 4.31E-9
+UB1     = -7.61E-18      UC1     = -5.6E-11       AT      = 3.3E4
+WL      = 0              WLN     = 1              WW      = 0
+WWN     = 1              WWL     = 0              LL      = 0
+LLN     = 1              LW      = 0              LWN     = 1
+LWL     = 0              CAPMOD  = 2              XPART   = 0.5
+CGDO    = 2.44E-10       CGSO    = 2.44E-10       CGBO    = 1E-9
+CJ      = 7.183583E-4    PB      = 0.865527       MJ      = 0.4904275
+CJSW    = 2.447569E-10   PBSW    = 0.8            MJSW    = 0.1910967
+CJSWG   = 6.4E-11        PBSWG   = 0.8            MJSWG   = 0.2261452
+CF      = 0              PVTH0   = 5.98016E-3     PRDSW   = 14.8598424
+PK2     = 3.73981E-3     WKETA   = 0.0123545      LKETA   = -0.0149634      )
*
```

Appendix D - SPICE Testbenches for Adiabatic Operation.

100 kHz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 1ns 1ns 100ns 200ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Ven En 0 PULSE(2.5 -2.5 0 1ns 1ns 400ns 800ns)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 625ns 625ns 625ns 8125ns 10us)
VPClk1 PClk1 0 PULSE(0 2.5 1.25us 625ns 625ns 6875ns 10us)
VPClk2 PClk2 0 PULSE(0 2.5 1.875us 625ns 625ns 5625ns 10us)
VPClk3 PClk3 0 PULSE(0 2.5 2.5us 625ns 625ns 4375ns 10us)
VPClk4 PClk4 0 PULSE(0 2.5 3.125us 625ns 625ns 3125ns 10us)
VPClk5 PClk5 0 PULSE(0 2.5 3.75us 625ns 625ns 1875ns 10us)
VPClk6 PClk6 0 PULSE(0 2.5 4.375us 625ns 625ns 625ns 10us)
VPClk0N PClk0N 0 PULSE(0 -2.5 625ns 625ns 625ns 8125ns 10us)
VPClk1N PClk1N 0 PULSE(0 -2.5 1.25us 625ns 625ns 6875ns 10us)
VPClk2N PClk2N 0 PULSE(0 -2.5 1.875us 625ns 625ns 5625ns 10us)
VPClk3N PClk3N 0 PULSE(0 -2.5 2.5us 625ns 625ns 4375ns 10us)
VPClk4N PClk4N 0 PULSE(0 -2.5 3.125us 625ns 625ns 3125ns 10us)
VPClk5N PClk5N 0 PULSE(0 -2.5 3.75us 625ns 625ns 1875ns 10us)
VPClk6N PClk6N 0 PULSE(0 -2.5 4.375us 625ns 625ns 625ns 10us)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 10ns 10ns 625ns 10us)
VClkSample ClkSample 0 PULSE(-2.5 2.5 5us 10ns 10ns 625ns 10us)
.tran 5ns 20us
.include C5_models.txt

---------------------------CMOS Mode------------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 1ns 1ns 100ns 200ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 20000ps 20000ps 10000ns 20000ns)
Ven En 0 PULSE(2.5 -2.5 0 1ns 1ns 400ns 800ns)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 10ns 10ns 625ns 10us)
VClkSample ClkSample 0 PULSE(-2.5 2.5 5us 10ns 10ns 625ns 10us)
.tran 5ns 20us
.include C5_models.txt
```

500 kHz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 200ps 200ps 20ns 40ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Ven En 0 PULSE(2.5 -2.5 0 200ps 200ps 80000ps 160000ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 125000ps 125000ps 125000ps 1625ns 2000ns)
VPClk1 PClk1 0 PULSE(0 2.5 250ns 125000ps 125000ps 1375ns 2000ns)
VPClk2 PClk2 0 PULSE(0 2.5 375ns 125000ps 125000ps 1125ns 2000ns)
VPClk3 PClk3 0 PULSE(0 2.5 500ns 125000ps 125000ps 875ns 2000ns)
VPClk4 PClk4 0 PULSE(0 2.5 625ns 125000ps 125000ps 625ns 2000ns)
VPClk5 PClk5 0 PULSE(0 2.5 750ns 125000ps 125000ps 375ns 2000ns)
VPClk6 PClk6 0 PULSE(0 2.5 875ns 125000ps 125000ps 125000ps 2000ns)
VPClk0N PClk0N 0 PULSE(0 -2.5 125000ps 125000ps 125000ps 1625ns 2000ns)
VPClk1N PClk1N 0 PULSE(0 -2.5 250ns 125000ps 125000ps 1375ns 2000ns)
VPClk2N PClk2N 0 PULSE(0 -2.5 375ns 125000ps 125000ps 1125ns 2000ns)
VPClk3N PClk3N 0 PULSE(0 -2.5 500ns 125000ps 125000ps 875ns 2000ns)
VPClk4N PClk4N 0 PULSE(0 -2.5 625ns 125000ps 125000ps 625ns 2000ns)
VPClk5N PClk5N 0 PULSE(0 -2.5 750ns 125000ps 125000ps 375ns 2000ns)
VPClk6N PClk6N 0 PULSE(0 -2.5 875ns 125000ps 125000ps 125000ps 2000ns)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 2000ps 2000ps 125000ps 2000ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 1000ns 2000ps 2000ps 125000ps 2000ns)
.tran 1000ps 4000ns
.include C5_models.txt

---------------------------CMOS Mode------------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 200ps 200ps 20ns 40ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 4000ps 4000ps 2000ns 4000ns)
Ven En 0 PULSE(2.5 -2.5 0 200ps 200ps 80000ps 160000ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 2000ps 2000ps 125000ps 2000ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 1000ns 2000ps 2000ps 125000ps 2000ns)
.tran 1000ps 4000ns
.include C5_models.txt
```

1 MHz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 100ps 100ps 10ns 20ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Ven En 0 PULSE(2.5 -2.5 0 100ps 100ps 40ns 80ns)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 62.5ns 62.5ns 62.5ns 812.5ns 1us)
VPClk1 PClk1 0 PULSE(0 2.5 125ns 62.5ns 62.5ns 687.5ns 1us)
VPClk2 PClk2 0 PULSE(0 2.5 187.5ns 62.5ns 62.5ns 562.5ns 1us)
VPClk3 PClk3 0 PULSE(0 2.5 250ns 62.5ns 62.5ns 437.5ns 1us)
VPClk4 PClk4 0 PULSE(0 2.5 312.5ns 62.5ns 62.5ns 312.5ns 1us)
VPClk5 PClk5 0 PULSE(0 2.5 375ns 62.5ns 62.5ns 187.5ns 1us)
VPClk6 PClk6 0 PULSE(0 2.5 437.5ns 62.5ns 62.5ns 62.5ns 1us)
VPClk0N PClk0N 0 PULSE(0 -2.5 62.5ns 62.5ns 62.5ns 812.5ns 1us)
VPClk1N PClk1N 0 PULSE(0 -2.5 125ns 62.5ns 62.5ns 687.5ns 1us)
VPClk2N PClk2N 0 PULSE(0 -2.5 187.5ns 62.5ns 62.5ns 562.5ns 1us)
VPClk3N PClk3N 0 PULSE(0 -2.5 250ns 62.5ns 62.5ns 437.5ns 1us)
VPClk4N PClk4N 0 PULSE(0 -2.5 312.5ns 62.5ns 62.5ns 312.5ns 1us)
VPClk5N PClk5N 0 PULSE(0 -2.5 375ns 62.5ns 62.5ns 187.5ns 1us)
VPClk6N PClk6N 0 PULSE(0 -2.5 437.5ns 62.5ns 62.5ns 62.5ns 1us)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 1ns 1ns 62.5ns 1us)
VClkSample ClkSample 0 PULSE(-2.5 2.5 500ns 1ns 1ns 62.5ns 1us)
.tran 500ps 2us
.include C5_models.txt

---------------------------CMOS Mode-----------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 100ps 100ps 10ns 20ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 2000ps 2000ps 1000ns 2000ns)
Ven En 0 PULSE(2.5 -2.5 0 100ps 100ps 40ns 80ns)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 1ns 1ns 62.5ns 1us)
VClkSample ClkSample 0 PULSE(-2.5 2.5 500ns 1ns 1ns 62.5ns 1us)
.tran 500ps 2us
.include C5_models.txt
```

5 MHz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 20ps 20ps 2ns 4ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Ven En 0 PULSE(2.5 -2.5 0 20ps 20ps 8000ps 16000ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 12500ps 12500ps 12500ps 162.5ns 200ns)
VPClk1 PClk1 0 PULSE(0 2.5 25ns 12500ps 12500ps 137.5ns 200ns)
VPClk2 PClk2 0 PULSE(0 2.5 37.5ns 12500ps 12500ps 112.5ns 200ns)
VPClk3 PClk3 0 PULSE(0 2.5 50ns 12500ps 12500ps 87.5ns 200ns)
VPClk4 PClk4 0 PULSE(0 2.5 62.5ns 12500ps 12500ps 62.5ns 200ns)
VPClk5 PClk5 0 PULSE(0 2.5 75ns 12500ps 12500ps 37.5ns 200ns)
VPClk6 PClk6 0 PULSE(0 2.5 87.5ns 12500ps 12500ps 12500ps 200ns)
VPClk0N PClk0N 0 PULSE(0 -2.5 12500ps 12500ps 12500ps 162.5ns 200ns)
VPClk1N PClk1N 0 PULSE(0 -2.5 25ns 12500ps 12500ps 137.5ns 200ns)
VPClk2N PClk2N 0 PULSE(0 -2.5 37.5ns 12500ps 12500ps 112.5ns 200ns)
VPClk3N PClk3N 0 PULSE(0 -2.5 50ns 12500ps 12500ps 87.5ns 200ns)
VPClk4N PClk4N 0 PULSE(0 -2.5 62.5ns 12500ps 12500ps 62.5ns 200ns)
VPClk5N PClk5N 0 PULSE(0 -2.5 75ns 12500ps 12500ps 37.5ns 200ns)
VPClk6N PClk6N 0 PULSE(0 -2.5 87.5ns 12500ps 12500ps 12500ps 200ns)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 200ps 200ps 12500ps 200ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 100ns 200ps 200ps 12500ps 200ns)
.tran 100ps 400ns
.include C5_models.txt

---------------------------CMOS Mode------------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 20ps 20ps 2ns 4ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 400ps 400ps 200ns 400ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 400ps 400ps 200ns 400ns)
Ven En 0 PULSE(2.5 -2.5 0 20ps 20ps 8000ps 16000ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 200ps 200ps 12500ps 200ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 100ns 200ps 200ps 12500ps 200ns)
.tran 10ps 400ns
.include C5_models.txt
```

10 MHz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 10ps 10ps 1ns 2ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Ven En 0 PULSE(2.5 -2.5 0 10ps 10ps 4ns 8ns)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 6.25ns 6.25ns 6.25ns 81.25ns 100ns)
VPClk1 PClk1 0 PULSE(0 2.5 12.5ns 6.25ns 6.25ns 68.75ns 100ns)
VPClk2 PClk2 0 PULSE(0 2.5 18.75ns 6.25ns 6.25ns 56.25ns 100ns)
VPClk3 PClk3 0 PULSE(0 2.5 25ns 6.25ns 6.25ns 43.75ns 100ns)
VPClk4 PClk4 0 PULSE(0 2.5 31.25ns 6.25ns 6.25ns 31.25ns 100ns)
VPClk5 PClk5 0 PULSE(0 2.5 37.5ns 6.25ns 6.25ns 18.75ns 100ns)
VPClk6 PClk6 0 PULSE(0 2.5 43.75ns 6.25ns 6.25ns 6.25ns 100ns)
VPClk0N PClk0N 0 PULSE(0 -2.5 6.25ns 6.25ns 6.25ns 81.25ns 100ns)
VPClk1N PClk1N 0 PULSE(0 -2.5 12.5ns 6.25ns 6.25ns 68.75ns 100ns)
VPClk2N PClk2N 0 PULSE(0 -2.5 18.75ns 6.25ns 6.25ns 56.25ns 100ns)
VPClk3N PClk3N 0 PULSE(0 -2.5 25ns 6.25ns 6.25ns 43.75ns 100ns)
VPClk4N PClk4N 0 PULSE(0 -2.5 31.25ns 6.25ns 6.25ns 31.25ns 100ns)
VPClk5N PClk5N 0 PULSE(0 -2.5 37.5ns 6.25ns 6.25ns 18.75ns 100ns)
VPClk6N PClk6N 0 PULSE(0 -2.5 43.75ns 6.25ns 6.25ns 6.25ns 100ns)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 100ps 100ps 6.25ns 100ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 50ns 100ps 100ps 6.25ns 100ns)
.tran 50ps 200ns
.include C5_models.txt

---------------------------CMOS Mode------------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 10ps 10ps 0.1ns 0.2ns)
Vin1 in1 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 200ps 200ps 100ns 200ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 200ps 200ps 100ns 200ns)
Ven En 0 PULSE(2.5 -2.5 0 10ps 10ps 0.4ns 0.8ns)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 100ps 100ps 6.25ns 100ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 50ns 100ps 100ps 6.25ns 100ns)
.tran 5ps 200ns
.include C5_models.txt
```

50 MHz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 2ps 2ps 200ps 400ps)
Vin1 in1 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Ven En 0 PULSE(2.5 -2.5 0 2ps 2ps 800ps 1600ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 1250ps 1250ps 1250ps 16.25ns 20ns)
VPClk1 PClk1 0 PULSE(0 2.5 2.5ns 1250ps 1250ps 13.75ns 20ns)
VPClk2 PClk2 0 PULSE(0 2.5 3.75ns 1250ps 1250ps 11.25ns 20ns)
VPClk3 PClk3 0 PULSE(0 2.5 5ns 1250ps 1250ps 8.75ns 20ns)
VPClk4 PClk4 0 PULSE(0 2.5 6.25ns 1250ps 1250ps 6.25ns 20ns)
VPClk5 PClk5 0 PULSE(0 2.5 7.5ns 1250ps 1250ps 3.75ns 20ns)
VPClk6 PClk6 0 PULSE(0 2.5 8.75ns 1250ps 1250ps 1250ps 20ns)
VPClk0N PClk0N 0 PULSE(0 -2.5 1250ps 1250ps 1250ps 16.25ns 20ns)
VPClk1N PClk1N 0 PULSE(0 -2.5 2.5ns 1250ps 1250ps 13.75ns 20ns)
VPClk2N PClk2N 0 PULSE(0 -2.5 3.75ns 1250ps 1250ps 11.25ns 20ns)
VPClk3N PClk3N 0 PULSE(0 -2.5 5ns 1250ps 1250ps 8.75ns 20ns)
VPClk4N PClk4N 0 PULSE(0 -2.5 6.25ns 1250ps 1250ps 6.25ns 20ns)
VPClk5N PClk5N 0 PULSE(0 -2.5 7.5ns 1250ps 1250ps 3.75ns 20ns)
VPClk6N PClk6N 0 PULSE(0 -2.5 8.75ns 1250ps 1250ps 1250ps 20ns)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 20ps 20ps 1250ps 20ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 10ns 20ps 20ps 1250ps 20ns)
.tran 10ps 40ns
.include C5_models.txt

---------------------------CMOS Mode------------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 2ps 2ps 100ps 200ps)
Vin1 in1 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 40ps 40ps 20ns 40ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 40ps 40ps 20ns 40ns)
Ven En 0 PULSE(2.5 -2.5 0 2ps 2ps 400ps 800ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 20ps 20ps 1250ps 20ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 10ns 20ps 20ps 1250ps 20ns)
.tran 2.5ps 40ns
.include C5_models.txt

PULSE(2.5 -2.5 0 100ps 100ps 10ns 20ns)
```

100 Mhz

```
------------------------Adiabatic Mode-----------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 1ps 1ps 100ps 200ps)
Vin1 in1 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Ven En 0 PULSE(2.5 -2.5 0 1ps 1ps 400ps 800ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 PULSE(0 2.5 625ps 625ps 625ps 8.125ns 10ns)
VPClk1 PClk1 0 PULSE(0 2.5 1.25ns 625ps 625ps 6.875ns 10ns)
VPClk2 PClk2 0 PULSE(0 2.5 1.875ns 625ps 625ps 5.625ns 10ns)
VPClk3 PClk3 0 PULSE(0 2.5 2.5ns 625ps 625ps 4.375ns 10ns)
VPClk4 PClk4 0 PULSE(0 2.5 3.125ns 625ps 625ps 3.125ns 10ns)
VPClk5 PClk5 0 PULSE(0 2.5 3.75ns 625ps 625ps 1.875ns 10ns)
VPClk6 PClk6 0 PULSE(0 2.5 4.375ns 625ps 625ps 625ps 10ns)
VPClk0N PClk0N 0 PULSE(0 -2.5 625ps 625ps 625ps 8.125ns 10ns)
VPClk1N PClk1N 0 PULSE(0 -2.5 1.25ns 625ps 625ps 6.875ns 10ns)
VPClk2N PClk2N 0 PULSE(0 -2.5 1.875ns 625ps 625ps 5.625ns 10ns)
VPClk3N PClk3N 0 PULSE(0 -2.5 2.5ns 625ps 625ps 4.375ns 10ns)
VPClk4N PClk4N 0 PULSE(0 -2.5 3.125ns 625ps 625ps 3.125ns 10ns)
VPClk5N PClk5N 0 PULSE(0 -2.5 3.75ns 625ps 625ps 1.875ns 10ns)
VPClk6N PClk6N 0 PULSE(0 -2.5 4.375ns 625ps 625ps 625ps 10ns)
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 10ps 10ps 625ps 10ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 5ns 10ps 10ps 625ps 10ns)
.tran 5ps 20ns
.include C5_models.txt

---------------------------CMOS Mode-----------------------------------------
VClk Clk 0 PULSE(2.5 -2.5 0 1ps 1ps 100ps 200ps)
Vin1 in1 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin2 in2 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin3 in3 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Vin4 in4 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin5 in5 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Vin6 in6 0 PULSE(2.5 -2.5 0 20ps 20ps 10ns 20ns)
Vin7 in7 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Vin8 in8 0 PULSE(-2.5 2.5 0 20ps 20ps 10ns 20ns)
Ven En 0 PULSE(2.5 -2.5 0 1ps 1ps 400ps 800ps)
Valuctrl2 aluctrl2 0 -2.5
Valuctrl1 aluctrl1 0 2.5
Valuctrl0 aluctrl0 0 -2.5
Vdd vdd 0 2.5
Vss vss 0 -2.5
VPClk0 PClk0 0 2.5
VPClk1 PClk1 0 2.5
VPClk2 PClk2 0 2.5
VPClk3 PClk3 0 2.5
VPClk4 PClk4 0 2.5
VPClk5 PClk5 0 2.5
VPClk6 PClk6 0 2.5
VPClk0N PClk0N 0 -2.5
VPClk1N PClk1N 0 -2.5
VPClk2N PClk2N 0 -2.5
VPClk3N PClk3N 0 -2.5
VPClk4N PClk4N 0 -2.5
VPClk5N PClk5N 0 -2.5
VPClk6N PClk6N 0 -2.5
VClkUpdate ClkUpdate 0 PULSE(-2.5 2.5 0 10ps 10ps 625ps 10ns)
VClkSample ClkSample 0 PULSE(-2.5 2.5 5ns 10ps 10ps 625ps 10ns)
.tran 5ps 20ns
.include C5_models.txt
```

Appendix E – MOSIS Fabrication Confirmation Document.

MOSIS Project 91963

Generated on: 27-Nov-14 08:55:08

Project Information

Design Number	**91963**
Date Created	26-NOV-14 03:24:37 pm
Project Status	**Queued for fab**
Target Run	AMI_C5F shared run scheduled on 01-Dec-2014
Run Date Requested	01-DEC-2014
Area	2.218 sq millimeters
Checksum	Binary CRC checksum: 867141269, Binary CRC byte count: 1826816

Administrative Information

Account Name	5507-MEP-INS/ITESM-DMTI
Account Contact Name	Graciano Dieck Assad
Account Contact E-mail	graciano.dieck.assad@itesm.mx
Design Contact E-mail	graciano.dieck.assad@itesm.mx
Design Contact Phone	+52(81)81582011
Cost	1 MEP unit(s) is used.
Quantity Ordered	5
ECCN	EAR99 (Research) Technology ECCN: EAR99
BIS 711 received	No
Approved for export	No

Design Details

Size in X	1489.2
Size in Y	1489.2
Wafer Technology	AMI_C5F
Fabrication Restricted to	AMI
Layout Format	GDS
Top Cell Name	ALUBennettFinal
Fill	MOSIS
Bonding Pad Count (Customer)	40
Bonding Pad Count (MOSIS)	40
Maximum Die Size	7620.0 X 7620.0
Layers (Density)	ACTIVE, CONTACT, GLASS, METAL1(20.0%), METAL2(26.0%), METAL3, N_PLUS_SELECT, N_WELL, POLY(1.1%), P_PLUS_SELECT, P_WELL, VIA, VIA2

Packaging Information

Quantity	Packaged	Package Id	Bonding Diagram From	Die Thickness (mils)
5	Y	DIP40	MOSIS	10

Project Warnings

Warning	Missing layer: POLY2 POLY layer drawn density is 1.1%; estimated additional density added by MOSIS fill operation is 10.5%; minimum required by AMI_C5F is 12.0%.
Warning	You have authorized MOSIS to add fill to your project to meet minimum layer density requirements, but, as noted, there is not enough fillable area available. This project may be excluded from the next run. Also, adding fill could affect the functioning of your design (seehttps://www.mosis.com/pages/Faqs/faq-design#7.0)
Warning	The following export form has not been received and approved: - [U.S. Export Compliance Questionnaire] Designs lacking required export documentation may not be fabricated. For further information see https://www.mosis.com/pages/export/export-paperwork .

Shipping Status

Shipping Status	Production ID	Parts expected at MOSIS	Will be shipped to
Will be shipped	Not Available	Not available	Will be shipped to: Graciano Dieck Assad Itesm Dmti Itesm, Campus Monterrey Investigacion Y Posgrado Eiti,Aulas Iv 64849 Monterrey, Nuevo Leon Mexico Routing Label: Dr. Graciano Dieck Assad Department of Electrical and Computer Engineering E. Garza Sada #2501 Monterrey, M?xico 64849 Aulas IV-212

Bibliography

[1] G. E. Moore, "Cramming More Components onto Integrated Circuits," *Electronics,* pp. 114-117, April 19, 1965.

[2] W. Haensch, E. J. Nowak, R. H. Dennard and P. M. Solomon, "Silicon CMOS devices beyond scaling," *IBM Journal of Research and Development,* vol. 50, no. 4/5, pp. 339-361, Jul-Sep 2006.

[3] J. D. Meindl, Q. Cheng and J. A. Davis, "Limits on Silicon Nanoelectronics for Terascale Integration," *Science,* vol. 293, pp. 2044-2049, 14 September, 2001.

[4] G. L. Snider, E. P. Blair, C. C. Thorpe, B. T. Appleton, G. P. Boechler, A. O. Orlov and C. S. Lent, "There is No Landauer Limit: Experimental Tests of the Landauer Principle," in *2012 12th IEEE International Conference on Nanotechnology (IEEE-NANO)*, Birmingham, United Kingdom, 20-23 August 2012.

[5] N. H. E. Weste and D. M. Harris, "Power," in *CMOS VLSI Design*, 4th ed., Boston, Addison-Wesley, 2011, pp. 181-209.

[6] D. Patterson, "The Trouble with Multi-Core," *IEEE Spectrum,* pp. 28-53, July 2010.

[7] H. Esmaeilzadeh, E. Blem, R. St.Amant, K. Sankaralingam and D. Burger, "Power Challenges May End the Multicore Era," *Communications of the ACM,* vol. 56, no. 2, pp. 93-102, February 2013.

[8] J. D. Meindl and J. A. Davis, "The Fundamental Limit on Binary Switching Energy for Terascale Integration (TSI)," *IEEE Journal of Solid-State Circuits,* vol. 35, no. 10, pp. 1515-1516, October 2000.

[9] R. K. Cavin, V. V. Zhirnov, J. A. Hutchby and G. I. Bourianoff, "Energy Barriers, Demons, and Minimum Energy Operation of Electronic Devices," *Fluctuation and Noise Letters,* vol. 5, no. 4, pp. 29-38, 2005.

[10] J. Von Neumann and A. W. Burks, Theory of Self-Reproducing Automata, University of Illinois Press, 1966.

[11] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM Journal of Research and Development,* vol. 44, no. 1/2, pp. 261-269, Jan/Mar 2000.

[12] C. H. Bennett, "Logical Reversibility of Computation," *IBM Journal of Research and Development,* pp. 525-532, November 1973.

[13] M. Hemmo and O. Shenker, "Entropy and Computation: The Landauer-Bennett Thesis Reexamined," *Entropy,* vol. 15, pp. 3297-3311, 2013.

[14] J. D. Norton, "Eaters of the lotus: Landauer’s principle and the return of Maxwell’s demon," *Studies in History and Philosophy of Modern Physics,* vol. 36, p. 375–411, 2005.

[15] M. Hemmo and O. Shenker, "Von Neumann’s Entropy Does Not Correspond to Thermodynamic Entropy," *Philosophy of Science,* vol. 73, pp. 153-174, April 2006.

[16] C. H. Bennett, "Notes on Landauer’s Principle, Reversible Computation, and Maxwell’s Demon," *Studies in History and Philosophy of Modern Physics,* vol. 34, pp. 501-510, 2003.

[17] D. Reeb and M. M. Wolf, "An improved Landauer Principle with finite-size corrections," *New Journal of Physics,* pp. 1-34, October 2014.

[18] J. Ladyman, S. Presnell, A. J. Short and B. Groisman, "The connection between logical and thermodynamic irreversibility," *Studies in History and Philosophy of Modern Physics,* vol. 38, pp. 58-79, 2007.

[19] D. Chiuchiú, M. C. Diamantini and L. Gammaitoni, "Role of conditional entropy in experimental tests of Landauer Principle," *eprint arXiv:1406.2562,* pp. 1-7, 06/2014.

[20] A. Daffertshofer and A. R. Plastino, "Landauer's principle and the conservation of information," *Physics Letters A,* vol. 342, pp. 213-216, 2005.

[21] A. Bérut, A. Arakelyan, A. Petrosyan, S. Ciliberto, R. Dillenschneider and E. Lutz, "Experimental verification of Landauer's principle linking information and thermodynamics," *Nature,* vol. 483, pp. 187-190, 8 March 2012.

[22] G. P. Boechler, J. M. Whitney, C. S. Lent, A. O. Orlov and G. L. Snider, "Fundamental limits of energy dissipation in charge-based computing," *Applied Physics Letters,* vol. 97, no. 103502, 2010.

[23] G. L. Snider, E. P. Blair, G. P. Boechler, C. C. Thorpe, N. W. Bosler, M. J. Wohlwend, J. M. Whitney, C. S. Lent and A. O. Orlov, "Minimum Energy for Computation, Theory vs. Experiment," in *2011 11th IEEE International Conference on Nanotechnology*, Portland, Oregon, USA, August 15-18, 2011.

[24] V. K. De and J. D. Meindl, "Opportunities for Non-Dissipative Computation," in *Proceedings of the 9th Annual IEEE International ASIC Conference and Exhibit*, Rochester, NY, USA, September 1996.

[25] S. G. Younis, Asymptotically Zero Energy Computing Using Split-Level Charge Recovery Logic, Boston, Massachusetts, USA: Massachusetts Institute of Technology, June 1994.

[26] J. S. Denker, "A Review of Adiabatic Computing," *1994 IEEE Symposium on Low Power Electronics,* pp. 94-97, 1994.

[27] V. I. Starosel'skii, "Adiabatic Logic Circuits: A Review," *Russian Microelectronics,* vol. 31, no. 1, pp. 37-58, 2002.

[28] S. Samik, "Adiabatic Computing: A Contemporary Review," in *2009 International Conference on Computers and Devices for Communication*, 2009.

[29] I. Hänninen, H. Lu, C. S. Lent and S. G. L, "Energy Recovery and Logical Reversibility in Adiabatic CMOS Multiplier," *Reversible Computation,* vol. 7948, pp. 25-35, 2013.

[30] A. Khazamipour and K. Radecka, "Adiabatic Implementation of Reversible Logic," in *48th Midwest Symposium on Circuits and Systems*, 2005.

[31] C. Kim, S. Yoo and S. Kang, "Low-power adiabatic computing with NMOS energy recovery logic," *Electronic Letters,* vol. 36, no. 16, pp. 1349-1350, 3rd August 2000.

[32] S. Kim, C. H. Ziesler and M. C. Papaefthymiou, "Charge-Recovery Computing on Silicon," *IEEE Transactions on Computers,* vol. 54, no. 6, pp. 651-659, June 2005.

[33] Y. Takahashi, D. Tsuzuki, T. Sekine and M. Yokoyamay, "Design of a 16-bit RISC CPU Core in a Two Phase Drive Adiabatic Dynamic CMOS Logic," in *TENCON 2007 - 2007 IEEE Region 10 Conference*, 2007.

[34] K. Takahashi and M. Mizunuma, "Adiabatic Dynamic CMOS Logic Circuit," *Electronics and Communications in Japan,* vol. 83, no. 5, pp. 50-58, 2000.

[35] M. K. Thomsen, "Design of Reversible Logic Circuits using Standard Cells," University of Copenhagen, Copenhagen, 2012.

[36] C. H. Ziesler, K. J, M. C. Papaefthymiou and S. Kim, "Energy Recovery Design for Low-Power ASICs," in *2003 IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2003)*, Tampa, FL, USA, 2003.

[37] R. K. Jana, G. L. Snider and D. Jena, "Energy-Efficient Clocking Based on Resonant Switching for Low-Power Computation," *IEEE Transactions on Circuits and Systems - I: Regular Papers,* vol. 61, no. 5, pp. 1400-1408, 2014.

[38] E. Fredkin and T. Toffoli, "Design Principles for Achieving High-Performance Submicron Digital Technologies," Proposal to DARPA, MIT Lab, 1978.

[39] Y. Moon and D. Jeong, "An Efficient Charge Recovery Logic Circuit," *IEEE Journal of Solid-State Circuits,* vol. 31, no. 4, pp. 514-522, 1996.

[40] D. Hongyu, Z. Runde and G. Yuanqing, "High Efficient Energy Recovery Logic for Adiabatic Computing," in *4th International Conference on ASIC, 2001. Proceedings.* , 2001.

[41] A. G. Dickinson and J. S. Denker, "Adiabatic Dynamic Logic," *IEEE Journal on Solid-State Circuits,* vol. 30, no. 3, pp. 311-315, 1995.

[42] A. Kramer, D. J. S, S. C. Avery, A. G. Dickinson and T. R. Wik, "Adiabatic Computing with the 2N-2N2D Logic Family," in *1994 Symposium on VLSl Circuits Digest of Technical Papers*, 1994.

[43] N. H. E. Weste and D. M. Harris, "Example: A Simple MIPS Microprocessor," in *CMOS VLSI Design*, Boston, Addison-Wesley, 2011, pp. 33-38.

[44] C. H. Bennett, "Logical Depth and Physical Complexity," in *The Universal Turing Machine: A Half-Century Survey*, Oxford University Press, 1988, pp. 227-257.

[45] N. H. E. Weste and D. M. Harris, "Sequential Circuit Design," in *CMOS VLSI Design*, Addison-Wesley, 2011, pp. 375-428.

[46] MOSIS, "MOSIS Wafer Electrical Test," November 2014. [Online]. Available: https://www.mosis.com/cgi-bin/params/ami-c5/v3bm-params.txt. [Accessed 1 November 2014].

Curriculum Vitae

César Orlando Campos Aguillón was born in Saltillo, Coahuila, México on July 20, 1990. He earned the Biomedical Engineering degree from *Instituto Tecnológico y de Estudios Superiores de Monterrey*, Monterrey Campus in December 2012. He was accepted in the master program in Electronics Engineering in January 2013.

This document was typed in using Microsoft Word by César Orlando Campos Aguillón.