Sense Amplifier-Based Pass Transistor Logic

Louis Poblete Alarcon
Jan M. Rabaey

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2010-173
http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-173.html

December 19, 2010

Sense Amplifier-Based Pass Transistor Logic

by

Louis Poblete Alarcón

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jan M. Rabaey, Chair
Professor Elad Alon
Professor Paul K. Wright

Fall 2010

Abstract

Sense Amplifier-Based Pass Transistor Logic

by

Louis Poblete Alarcón

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Science

University of California, Berkeley

Professor Jan M. Rabaey, Chair

Reducing the energy required per operation is the key to building ultra-low energy systems, and the most effective way of achieving this is to reduce the supply voltage. However, operating CMOS circuits at low supply voltages increases circuit delay, leading to lower circuit performance. In this region, the sub-threshold leakage energy component becomes more pronounced and can even dominate the total circuit energy. Increasing threshold voltages reduces the amount of leakage, but this forces operation in the sub-threshold region where performance and variability become exponentially worse.

The use of the sense amplifier-based pass transistor logic (SAPTL) topology is one approach to reducing the energy per operation. It uses an inverted pass transistor logic (PTL) tree, which inherently has no gain, and hence no power supply connections, eliminating the sources of sub-threshold leakage current. Reducing the threshold voltages of the PTL transistors improves performance, without the leakage current increase associated with conventional static CMOS logic. This reduced threshold voltage also allows the PTL transistors to operate in the super-threshold region, even for very low supply voltages, avoiding the increased delay and variability associated with the sub-threshold operating regime.

Gain is introduced by using drivers and sense amplifiers (SAs) that restore the output voltage swing and provide the appropriate output current to drive the fan-out capacitances. These drivers and SAs are the primary source of sub-threshold leakage, which can be amortized by making the PTL networks complex, and by applying various leakage reduction techniques.

SAPTL-based 90nm test circuits using both synchronous and asynchronous timing schemes have been designed, fabricated and tested. These circuits show leakage and energy characteristics better than the equivalent static CMOS circuits. These test chips also demonstrate rudimentary SAPTL-based design flows using commercially available CAD tools.

Simulation and measurement results of basic synchronous SAPTL building blocks show a

40X-50X reduction in standby current and a 6X reduction in energy when compared to an equivalent CMOS logic block, at the expense of a 10X-30X increase in delay. Operating the SAPTL asynchronously reduces the average delay by 89%. However, adding the necessary handshaking circuitry increases the energy by 31%.

These SAPTL building blocks are used to create a parallel 64-byte asynchronous SAPTL-based CRC generator with a minimum energy point that is 25% lower than that of the static CMOS equivalent, with a 6X delay penalty. Also, due to the nature of the PTL tree, forward-biasing the body of the PTL transistors results in a 10% reduction in delay with no energy penalty.

The advantages of the SAPTL over conventional static CMOS is expected to be more significant as technology continues to scale, where subthreshold leakage continue to prevent supply voltages from being aggressively scaled.

Contents

List of Figures

List of Tables

Acknowledgments

A lot of people have helped me, one way or the other, to finish this thesis, and I would like to take this opportunity to say thank you...

... to my advisor, Jan Rabaey, for his constant support and patience, without which, this thesis would not have been possible.

... to Mircea Stan, who started all this, with talks and ideas about stacks and pass transistors; to Elad Alon, for all the excellent feedback; to John Wawrzynek and Paul Wright, for agreeing to be on my quals and thesis committee and to Marly Roncken and Ivan Sutherland for all the great help and insights.

... to Tsung-Te Liu, for being the best tapeout partner and for taking all the abuse I sent his way with a smile and the ever-present "can do" attitude.

... to the awesome *jansgroup* and the rest of the BWRC crowd, especially to Simone Gambini, Rikky Muller, Michael Mark, Jesse Richmond, David Chen, Matthew Pierson, Ping-Chen Huang, Arash Parsa, Wen Li, Ali Ercan, Jing Yang, Cristian Marcu, Amin Arbabian, and Maryam Tabesh, whose friendship made the whole process of getting this thesis done both fun and stimulating, with coffee and chocolate being the preferred stimulant.

... to Kimiya Hajkazemshirazi, Jayson Hu, Vittoria Lok, Sun Chen and Shuo Liu, for the amazing effort that really helped.

... to the tremendous BWRC faculty and administrative/support staff especially Ali Niknejad, Borivoje Nikolic, Andre Vladimirescu, Gary Kelson, Deirdre McAuliffe-Bauer, Ellen Lenzi, Pierce Chua, Kevin Zimmerman, Brian Richards, Susan Mellers, and Ken Tang for taking care of all of us at BWRC.

... to Tom Boot, for treating all of us like family.

... to Octavian Florescu, for being a great project partner in the early stages of the SAPTL, and to Ehsan Adabi, Bagher Afshar, Subramaniam Venkatraman, Alexandr Simma, Benjamin Rubinstein, Juliet Rubinstein, Leon Barrett, Zachary Anderson, Bonnie Kirkpatrick, Lauren Barth-Cohen, Anupama Bowonder, Pratik Patel, and Eric Chin for the great company and making sure I get out of the lab often enough.

.. to Amy Wu, for the love, friendship and understanding.

... to the University of the Philippines, Diliman and the UP Electrical and Electronics Engineering Department for giving me the time and support I needed everytime I asked for it, and to Richard Hizon, Marc Rosales, Anastacia Ballesil-Alvarez, Maria Teresa Gusad-de Leon and Joy Reyes-Madamba, for keeping my chair warm.

... to my parents, Wilfredo Alarcón and Maria Luisa Alarcón and my sisters Yazmynn and Maria Celine, for the love, encouragement and unwavering support.

... and to God, *Ad maiorem Dei gloriam.*

Chapter 1

Introduction

The ability to design and build logic and computational elements that operate at extremely low energy levels is seen as a very important enabler for systems in various application domains such as mobile devices, wireless sensor networks and bio-medical systems [Sak03]. Devices operating at these low energy levels can also take advantage of alternative energy storage and scavenging methods that can lead to almost indefinite operational lifetimes [CFK$^+$10], as well as new computing and system paradigms.

Technology scaling and supply voltage reduction have been responsible for the continued energy reduction and performance improvement in complementary static CMOS circuits, the most popular logic topology in use today. However, the increased leakage energy brought about by scaling and V_{DD} reduction is starting to limit the minimum energy that static CMOS circuits can achieve. One low-energy alternative to complementary static CMOS circuits is the sense amplifier-based pass transistor logic (SAPTL) topology.

1.1 Overview

This work presents the sense amplifier-based pass transistor logic (SAPTL) as a low energy alternative logic topology to fully complementary static CMOS logic. The SAPTL takes advantage of the inherent decoupling of logic functionality and circuit gain in pass transistor circuits in order to achieve ultra-low energy operation that is lower than static CMOS, especially in cases where leakage energy becomes a significant component of total circuit energy. This decoupling also allows the improvement of performance through threshold voltage reduction without increasing the total SAPTL energy consumption.

1.2 Thesis Flow and Organization

Chapter 2 presents an overview of the low energy design, starting with the basics of how energy is consumed in digital circuits. Techniques on how energy consumption can be reduced, such as sub-threshold leakage current mitigation, as well as the challenges involved, are mentioned. Metrics showing how these reductions affect the performance of these circuits are then introduced.

Pass transistor logic is introduced in Chapter 3 as a low leakage alternative to complementary static CMOS logic, since it operates inherently without built-in gain elements. The penalties of operating without gain, as well as ways to introduce more gain into the circuit are also presented.

Chapter 4 introduces the basic organization of the sense amplifier-based pass transistor logic (SAPTL) circuit, which is built using (1) an inverted pass transistors tree, or stack, and (2) gain elements in the form of drivers and sense amplifiers. This is followed by an analysis of the delay and energy of these SAPTL building blocks and the implications of combining these two blocks in order to implement boolean functions.

The timing behavior of SAPTL is shown in Chapter 5. A two-phase clocking scheme as well as two asynchronous handshaking schemes are presented. The energy and delay characteristics of logic functions using these timing schemes are then compared with their corresponding complementary static CMOS implementations, highlighting design areas where using SAPTL will be advantageous. Chapter 6 shows several examples of the SAPTL circuits, including practical design and implementation issues as well as measurement results of fabricated circuits.

Chapter 7 presents the conclusions drawn from this work, and then recommendations as well as possible future extensions of this thesis are presented in Chapter 8.

Chapter 2

Ultra-Low Energy Design

The continued scaling of CMOS technology provides increased transistor density, as well as increased performance. Ideally, scaling also reduces the the overall circuit power consumption. However, due to nonidealities like leakage, scaling typically results in an increase in overall circuit power consumption and this increase in power that accompanies this scaling trend is preventing us from truly harnessing the benefits of decreasing transistor feature sizes [Sak03]. For applications (1) that are severely energy limited, such as those using implantable electronics [ROSO05] [GN04] or (2) where performance requirements are relaxed and energy is the primary constraint, such as RFID protocol processor systems [RGDMC09] and implantable integrated circuits [KGM09] with clock frequencies in the hundreds of kHz, the energy per operation must continue to decrease, allowing for years of battery life at relatively low operating frequencies and power levels.

The goal, therefore, of ultra-low energy design is to be able to create circuits and systems that consume the least amount of energy possible, while meeting application-specific constraints in the face of device and process nonidealities.

In this chapter, the relationship between energy, supply voltage, performance and MOS subthreshold leakage current of CMOS logic circuits are presented, leading to the main difficulties in reducing their energy per operation (E_{OP}). Various methodologies and techniques currently used to lower E_{OP} are then enumerated. The chapter concludes with a discussion on the possibilities of using pass transistor logic (PTL) as an alternative approach to reducing E_{OP} in certain application scenarios.

2.1 Characteristics of CMOS Logic

A generalized complementary static CMOS logic, shown in Fig. 2.1, is composed of a PMOS pull-up network, connecting the supply rail to the output and an NMOS pull-down network that connects the output to ground. These complementary pull-up and pull-down networks connects the output to either V_{DD} or ground, depending on the logic gate's input.

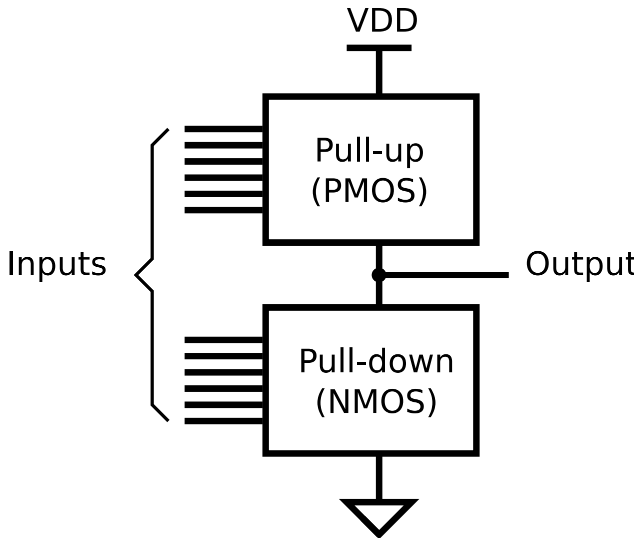


Figure 2.1: A generalized complementary CMOS logic gate.

The behavior of a generic complementary CMOS logic gate can be analyzed in terms of an equivalent CMOS inverter since its pull-up and pull-down networks can be modeled as a single equivalent PMOS or NMOS transistor. Thus, the CMOS inverter and its behavior, as described in this chapter, can be regarded as representative of the entire fully complementary CMOS logic family.

2.1.1 The CMOS Inverter

The current that a CMOS inverter draws from the supply voltage, V_{DD}, can be decomposed into three main components [RCN03], (1) I_{ON}, the switching or dynamic current used to charge up the capacitive load at its output, C_{sw}, (2) I_{leak}, the subthreshold leakage current and (3) I_{sc}, the short-circuit or direct path current, as seen in Fig. 2.2.

For supply voltages significantly above the threshold voltage [SN90][ZBSF05],

$$I_{ON,sat} = \frac{\mu_{eff} C_{ox}}{2} \frac{W}{L_{eff}} \left(V_{DD} - V_{TH}\right)^{\alpha_{mos}} \tag{2.1}$$

where $\alpha_{mos} \approx 1.3$. However, in the subthreshold region it can be expressed as [TN98][BRG05]

$$I_{ON,sub} = \mu_{eff} C_{ox} \frac{W}{L_{eff}} \left(m - 1\right) V_T^2 e^{\frac{V_{DD} - V_{TH}}{m V_T}} \left(1 - e^{-\frac{V_{DD}}{V_T}}\right) \tag{2.2}$$

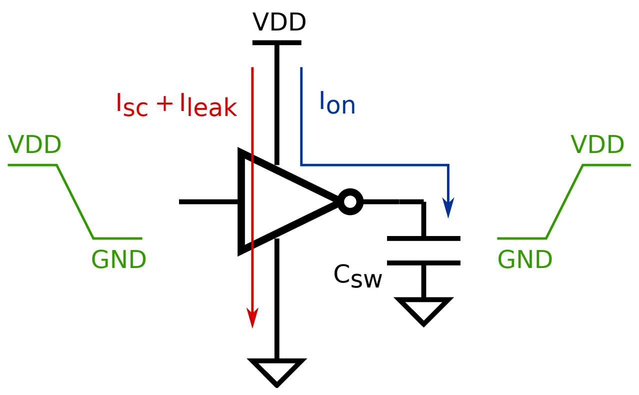


Figure 2.2: A CMOS inverter driving a capacitive load.

where m is defined as

$$m = \left(1 + \frac{C_d}{C_{ox}}\right) = \frac{S_S}{(\ln 10)\, V_T} \tag{2.3}$$

and S_S is the subthreshold slope, given by

$$S_S = (\ln 10)\, V_T \left(1 + \frac{C_d}{C_{ox}}\right) \tag{2.4}$$

where $V_T = \frac{kT}{q}$, C_{ox} is the gate-oxide capacitance per unit area, C_d is the depletion region capacitance per unit area, W is the width and L_{eff} is the effective channel length of the MOS device.

The leakage current is the current that flows when V_{GS} is equal to zero, and is commonly dominated by subthreshold leakage currents [KAB$^+$03], thus

$$I_{leak} = \mu_{eff} C_{ox} \frac{W}{L_{eff}} (m-1)\, V_T^2 e^{-\frac{V_{TH}}{mV_T}} \left(1 - e^{-\frac{V_{DS}}{V_T}}\right) \tag{2.5}$$

Note that at supply voltages less than the sum of the NMOS and PMOS threshold voltages, the short-circuit current, I_{sc}, becomes negligible [CB95]. Though the paths through which I_{leak} and I_{sc} take are the same, I_{sc} flows only when a transient event occurs. The leakage current, on the other hand, is the current that flows when the circuit node voltages settle to their final static values after a transient event.

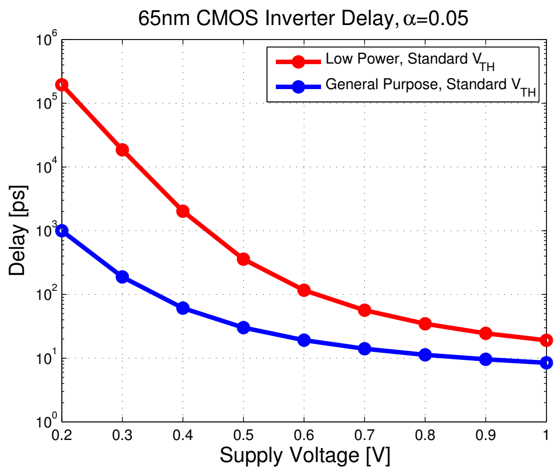


Figure 2.3: CMOS inverter delay as a function of V_{DD}.

2.1.2 Delay

The delay in a CMOS gate can be expressed as

$$D = \frac{C_{sw} V_{DD}}{2 I_{ON}} \tag{2.6}$$

Thus, the delay for above threshold operation can be expressed as a function of the supply voltage, as

$$D_{sat} \propto \frac{V_{DD}}{(V_{DD} - V_{TH})^{\alpha_{mos}}} \tag{2.7}$$

and in the subthreshold region, assuming $V_{DD} > 0.1$ V, such that $1 - e^{-\frac{V_{DD}}{V_T}} \approx 1$, the delay is

$$D_{sub} \propto V_{DD} e^{-\frac{V_{DD} - V_{TH}}{m V_T}} \tag{2.8}$$

As seen in Eqs. 2.7 and 2.8, decreasing the supply voltage increases the delay. Fig. 2.3 shows the delay of two flavors of 65nm CMOS inverters as a function of supply voltage.

In real systems, several CMOS gates are normally grouped into logic or pipeline stages in order to perform a certain function or operation, such as adding two numbers. If the time needed to complete this operation is defined as T_{OP}, then the activity factor of each gate i in the logic or pipeline stage can be defined as

$$\alpha_i = k_i \frac{D_i}{T_{OP}} \tag{2.9}$$

For an inverter, every change at its input causes a change at the outputs, hence, $k = 1$. For multiple-input gates, k can be less than 1 since not all changes at the gate input causes the output to change. For synchronous systems, on the other hand, if f_{sw} is the global clock frequency, where $f_{sw} = \frac{1}{T_{OP}}$, then $\frac{1}{\alpha}$ can represent the average number of clock cycles between each output switching event of a particular gate or subsystem.

2.1.3 Power

The average power consumed by the CMOS inverter for a low-to-high transition at its output can be expressed as the sum of the dynamic and leakage power components

$$P_{ave} = P_{dyn} + P_{leak} = V_{DD}I_{DD} + V_{DD}\left(I_{leak} + I_{sc}\right) = \alpha C_{sw} V_{DD}^2 f_{sw} + V_{DD}\left(I_{leak} + I_{sc}\right) \tag{2.10}$$

where I_{DD} is the total current drawn from the supply. If the voltage swing across the switched capacitance is less than the supply voltage, the dynamic power, in general, is given as

$$P_{dyn} = \alpha C_{sw} V_{DD} V_{swing} f_{sw} \tag{2.11}$$

Combining Eqs. 2.1, 2.5, 2.7 and 2.8, we get the CMOS power expressions in terms of the supply voltage as

$$P_{dyn,sat} \propto \alpha C_{sw} V_{DD}\left(V_{DD} - V_{TH}\right)^{\alpha_{mos}} \tag{2.12}$$

$$P_{dyn,sub} \propto \alpha C_{sw} V_{DD} e^{\frac{V_{DD}-V_{TH}}{mV_T}} \tag{2.13}$$

$$P_{leak} \propto V_{DD} e^{-\frac{V_{TH}}{mV_T}} \tag{2.14}$$

Fig. 2.4 shows the power as a function of supply voltage for two different flavors of inverters in a 65nm CMOS process. As seen in Eq. 2.10, the average power is dependent on the value of the switched capacitance, the switching rate of the output node, the leakage current and the supply voltage.

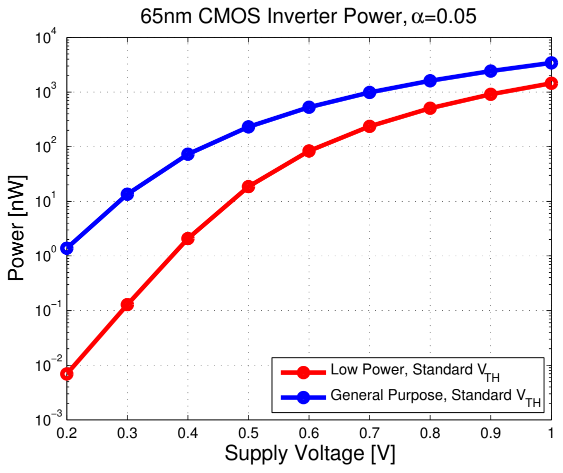


Figure 2.4: CMOS inverter power as a function of V_{DD}.

2.1.4 Energy

The average power is a very important metric, providing insight into how much average current a power source needs to provide a logic gate. Of interest as well is the time integral of the average power over the duration of an operation, or the energy per operation, E_{OP}. The energy required to complete a set of meaningful logic operations typically determines the amount of energy an energy source must be able to provide before it needs to be replenished.

Thus, for a given time needed to complete an operation, T_{OP}, the energy per operation is defined as

$$E_{OP} = P_{ave}T_{OP} = (P_{dyn} + P_{leak})\,T_{OP} = P_{dyn}D + P_{leak}T_{OP} \tag{2.15}$$

The dynamic component of the average power is multiplied only by the delay of the gate since I_{ON} is zero after the output switches. Combining Eqs. 2.10 and 2.15, and ignoring the short-circuit current, we get

$$E_{OP} = C_{sw}V_{DD}^2 f_{sw}D + V_{DD}I_{leak}T_{OP} = \alpha C_{sw}V_{DD}^2 + V_{DD}I_{leak}T_{OP} \tag{2.16}$$

For the simple case when the logic gate's output switches only once per operation, or $D = \alpha T_{OP}$, the energy per operation can be expressed as

$$E_{OP} = E_{dyn} + E_{leak} = \alpha C_{sw}V_{DD}^2 + \frac{V_{DD}I_{leak}D}{\alpha} \tag{2.17}$$

Again, if the voltage swing across the switched capacitance is less than the supply voltage, the dynamic energy, in general, is given as

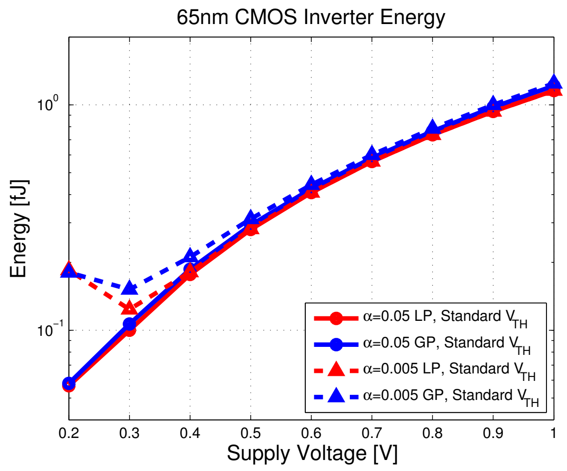


Figure 2.5: CMOS inverter energy per operation as a function of V_{DD}.

$$E_{dyn} = \alpha C_{sw} V_{DD} V_{swing} \tag{2.18}$$

Combining Eqs. 2.5, 2.7 and 2.8, we can express the leakage energy as

$$E_{leak,sat} \propto \frac{V_{DD}^2}{\alpha} \frac{e^{-\frac{V_{TH}}{mV_T}}}{(V_{DD} - V_{TH})^{\alpha_{mos}}} \tag{2.19}$$

$$E_{leak,sub} \propto \frac{V_{DD}^2}{\alpha} e^{-\frac{V_{DD}}{mV_T}} \tag{2.20}$$

Thus, a lower activity factor decreases the dynamic energy component, but at the same time increases the effect of the static or leakage energy component, as shown in Fig. 2.5 for two versions of a 65nm CMOS inverter, and for two different activity factors. For $\alpha = 0.005$, the total energy is dominated by the leakage energy component, which is proportional to the delay as seen in Eq. 2.17.

The tradeoff between energy and delay or performance can be visualized in a more compact manner by combining the Figs. 2.3 and 2.5, with the supply voltage as a swept parameter. Thus, the energy-delay plot of the two 65nm CMOS inverters, in Fig. 2.6, shows the effect of changing the supply voltage, simultaneously on both the inverter energy and delay.

2.2 A Sampling of Low Energy Design Techniques

As seen in Eq. 2.17, the dynamic energy per operation of CMOS gates can be reduced if (1) the effective switched capacitance, αC_{sw}, is lowered, (2) the voltage swing is lowered and/or (3) the supply voltage is lowered.

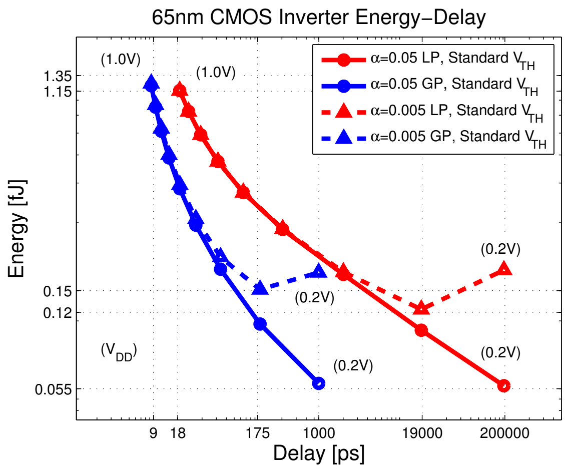


Figure 2.6: The energy-delay plot of the 65nm CMOS inverters.

The effective switched capacitance of a circuit or system can be reduced by lowering the switching activity, α. One way of reducing α is by preventing unnecessary transitions in the clock network by gating the clock [WPW00]. Another way of reducing the switching activity is to create circuit elements such as flip-flops that sense if the incoming data is different from the stored data [KKJ01] [HTH$^+$99]. These conditional flip-flop evaluation methods can prevent unnecessary transitions within the flip-flop itself at the expense of the additional circuitry needed to detect these differences.

Reducing the supply voltage seems to be the best way to reduce power and energy [CB95] [DC97] due to their quadratic dependence on V_{DD}. However, due to the increased gate delays, circuit performance must be traded off for reduced power and/or energy [LS93]. This tradeoff can be made more fine-grained and flexible by using variable and/or multiple (1) supply voltages to reduce energy and (2) threshold voltages to reduce the delay penalty incurred by reducing V_{DD} [KSM$^+$98] [Kur02] [DC97] [CC06]. Reduced output-swing circuits such as pass transistors [SKK97] or bus drivers [NII$^+$93] can also be selectively used to drive nodes with relatively large capacitances, thus allowing more fine-tuned voltage control over individual circuit nodes rather than groups of gates.

This increased flexibility is accompanied by additional overhead costs, such as voltage level converters, increased design complexity and the cost of generating and routing the various supply and threshold voltages that are required. Thus in order to minimize energy, optimal combinations of the supply and threshold voltage have been reported [Sta01], most often leading to subthreshold operation [WC05].

In order to understand how much more energy reductions can potentially be achieved, it is important to determine if theoretical limits on E_{OP} exists, and if these limits exist, can logic gates can operate near these limits or are there barriers preventing current logic circuits from operating close to these limits.

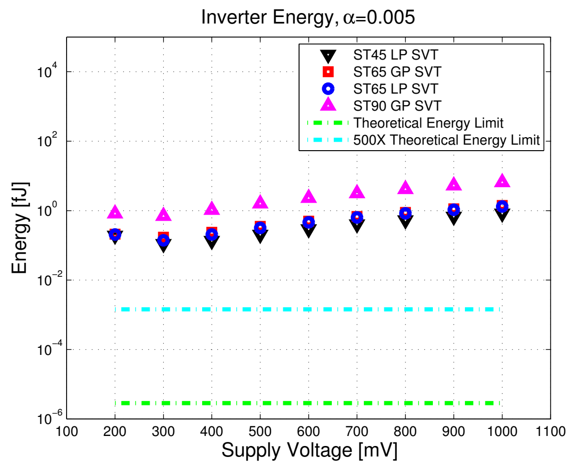


Figure 2.7: Theoretical energy and voltage limits.

2.3 Fundamental Limits

In order to discriminate between two binary states, the transfer characteristic of an inverter must have a slope, m, where $|m| > 1$ [MD00]. This leads to the minimum allowable CMOS supply voltage of

$$V_{DD,\mathrm{lim}} \cong 2V_T \ln\left(2 + \frac{C_d}{C_{ox}}\right) = 2V_T \ln\left(1 + \frac{S_S}{(\ln 10)\,V_T}\right) \cong 2\,(\ln 2)\,V_T \approx 36\,mV \qquad (2.21)$$

This minimum allowable supply voltage, $V_{DD,\mathrm{lim}}$, gives the minimum energy, $E_{OP,\mathrm{lim}}$, needed to move a single electron in order to charge the gate capacitance of a MOSFET, resulting in [MD00] [ZBSF05]

$$E_{OP,\mathrm{lim}} = (\ln 2)\,kT \approx 2.87 \times 10^{-21}\,J \qquad (2.22)$$

Fig. 2.7 shows the relative simulated values of minimum energies and voltages of various CMOS inverter technologies, as compared to $E_{OP,\mathrm{lim}}$. In current technologies, the inverter switching energies are approximately five orders of magnitude larger than the theoretical limit.

Eqs. 2.21 and 2.22 assume ideal and perfectly matched transistors. Other factors can significantly limit the minimum achievable supply voltage and energy. Most notable are (1) the increased device variability and mismatch [TBK$^+$10] that can affect the functionality of the circuit and (2) the increased subthreshold leakage energy components [ITR] that increases the minimum energy that can be achieved.

Although device noise sources such as shot noise, flicker noise and thermal noise can still be considered negligible in digital circuits, external noise sources such as (1) crosstalk, (2) power/ground bounce and (3) substrate noise can significantly degrade the performance and the reliability of digital integrated circuits [NRC08]. These external noise sources can also limit the reduction of V_{DD} [BS99], but is outside the scope of this work.

2.3.1 Variability

Variability arises due to the fact that devices are scaling faster than the manufacturing technology's ability to control the process parameters, resulting in (1) random atomic-level differences between devices and (2) systematic shifts in manufacturing dependent process conditions [BFG$^+$06]. In MOS transistors, variability significantly affects the channel width and length as well as the threshold voltage [UTB$^+$06]. At low voltages, variability can cause (1) logic failure, (2) the inability of an SRAM cell to retain its state, (3) reduced noise margins, (4) increased soft-error failures, (5) exponential degradation of performance and (6) increased subthreshold leakage current [TBK$^+$10] [PdGT04].

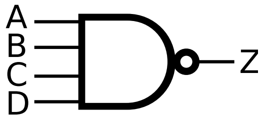
Circuit and architectural approaches to reduce the impact of variability have been reported, such as (1) the use of more stringent design rules and regular circuit layout techniques [KRH$^+$05] [CCL$^+$08] [PN09], (2) variability-aware device sizing [VCT$^+$04], and (3) systems that employ error detection and/or correction blocks that can allow aggressive clocking or voltage scaling [DRL$^+$06] [BTK$^+$09] [Mit10] [TBW$^+$09].

Thus, in order to assure correct functionality in the presence of process variations, additional energy is needed to support these circuit and architectural techniques. This additional energy then raises the minimum energy required to complete an operation.

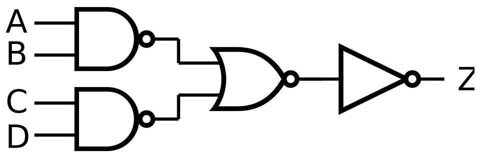
2.3.2 Subthreshold Leakage

The most effective way of reducing the dynamic energy per operation of digital circuits is by reducing the supply voltage, V_{DD}. This however, comes at the expense of increased leakage or standby energy per operation due to the required threshold voltage reduction needed to achieve a given performance goal [Sak03]. This energy wall due to standby energy is now considered a clear long-term threat to the survival of CMOS technology itself, and its management a major challenge [ITR].

Standby energy is commonly dominated by subthreshold leakage currents [KAB$^+$03], which flow in circuits that provide gain, by virtue of their power rail connections. In static complementary CMOS logic, each gate is essentially a gain stage, with a V_{DD} and ground connection, giving each gate (1) signal regeneration, (2) current drive and also (3) a leakage path. Thus, the functionality of the gate is tightly coupled to its gain, and hence to leakage.



(a)



(b)

Figure 2.8: A NAND4 gate implemented using (a) a single gate and (b) multiple gates.

Subthreshold leakage current can thus be reduced (1) by increasing the effective channel length, L_{eff}, (2) by reducing the supply voltage and consequently the drain-to-source voltage, V_{DS} or (3) by increasing the threshold voltage, V_{TH}.

Device-level leakage reduction techniques have been reported, such as multiple threshold voltages (MTCMOS) [MDM+95] [CHC04], as well as the use of non-CMOS devices such as integrated relays [CFK+10]. Circuit-level techniques that exist in literature include the use of boosted-gate MOS (BGMOS) transistors [ITN+00], super cut-off CMOS (SCCMOS) devices [KNS98] and multi-voltage CMOS (MVCMOS) techniques [Sta98].

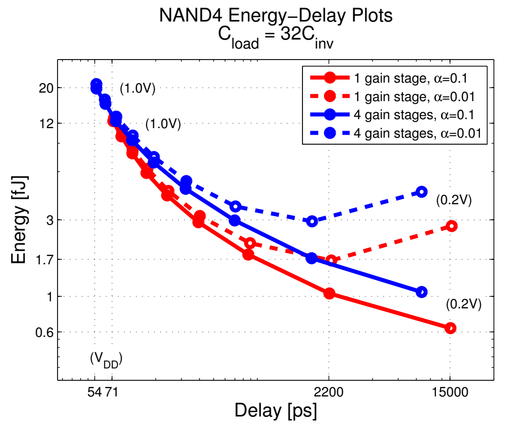
A sampling of other currently used low leakage circuit techniques aside from voltage scaling include the use of (1) non-minimum channel lengths; (2) stacked transistors and (3) various header and footer switch topologies and are summarized in [KNC02], [CSH+03] and [Sak03].

Increasing the transistor channel length reduces leakage by increasing the effective resistance of the leakage path between the supply rails, resulting in delay penalties. Using reverse body bias to increase V_{TH} also leads to the same results, however due to increased channel doping [Yu04], BTBT limits the effectiveness of body bias to reduce leakage, and hence the effective power reduction is reduced by approximately 4X per technology generation [KNB+99].

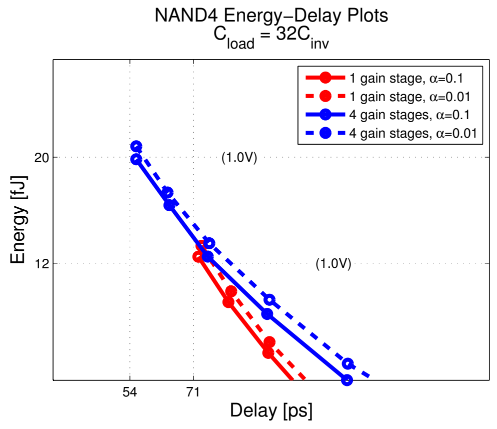
As seen in Fig. 2.6, the leakage energy component prevents further reductions in energy and thus, determines the minimum E_{OP} that can be achieved, which is still several orders of magnitude larger than the theoretical limit in Eq. 2.22. By applying the leakage reduction techniques described above, circuit operation closer to $E_{OP,\lim}$ can be obtained.

2.4 Leakage and Gain

Another way to reduce leakage current is to use more complex gates. By combining more functionality in a single gate, less gates are used resulting in less leakage paths from V_{DD} to ground. Again, this would increase the effective resistance of the leakage path from the supply rails. However, using fewer, more complex gates reduces the amount of gain present in the circuit. Fig. 2.8 shows two different implementations of a NAND4 gate: (1) using a single complex gate, and hence a single gain stage, as seen in Fig. 2.8a, and (2) using multiple gates (gain stages) shown in Fig. 2.8b.



(a)



(b)

Figure 2.9: The CMOS NAND4 gate: (a) energy-delay characteristics and (b) details at high supply voltages.

The energy and delay characteristics of both NAND4 gates as the supply voltage is swept from 0.2 V to 1 V is given in Fig. 2.9. As expected, at high supply voltages near 1 V, the NAND4 implemented using more gain stages can drive the output load faster than the one implemented using a single gate. However, this performance improvement costs more energy. Note that at lower supply voltages, the single, more complex gate consumes less energy due to a smaller effective switched capacitance. At lower activity factors, the leakage energy component is also less due to the fact that it has fewer leakage current paths. Thus, in cases where performance is less critical than energy consumption, using circuits with a reduced number of gain elements may allow lower energy operation than just scaling down the supply voltage of a topology optimized for speed.

2.5 Summary

Lowering the supply voltage has been the most effective method of lowering the energy per operation of CMOS logic gates. However, due to increased variability and standby energy caused by subthreshold leakage currents, the effectiveness of voltage scaling has diminished to the point where further reductions in supply voltage can lead to incorrect operation or an increase the total energy per operation instead of a reduction. Thus, various variability mitigation and leakage current reduction techniques have been reported, in an effort to bring the voltage and energy levels as close as possible to their theoretical minimum values. Though noise and variability are important issues that must be considered carefully, the focus of this work, however will be on the reduction of subthreshold leakage current. Also, since most noise and variability mitigation circuits also introduce additional leakage themselves, the resulting overall leakage current determines the minimum energy per operation that can be achieved.

One key idea presented in this chapter is that due to the inherent gain in static CMOS gates, the subthreshold leakage current and the output drive current share the same path through the logic gate. Thus, most circuit-level leakage reduction techniques must trade-off the amount of leakage reduction to the amount of drive current needed by the system, therefore limiting the leakage current reduction that can be achieved. Chapter 3 shows how pass transistor circuits, by decoupling gain and logic functionality, can change this trade-off.

Chapter 3

Pass Transistor Logic

Pass transistor logic (PTL) has been the focus of many research projects [Tak98] ranging from their use as static CMOS replacement circuits for specialized applications, to computer-aided design (CAD) methodologies that take advantage of various PTL characteristics such as area efficiency and regularity [YYN+90]. In this chapter, a sampling of various pass transistor logic implementations are described, with the intention of highlighting the relationship between logic functionality, gain stages, performance and energy.

3.1 Basic PTL Topologies

Various pass transistor logic implementations have been reported and can be broadly grouped into two classes based on the pass transistor network (PTN) used: (1) NMOS-only PTNs and (2) PTNs that use both NMOS and PMOS transistors [MNO00]. NMOS-only pass transistor networks are less complex, resulting in (1) lower input and signal path capacitances and (2) smaller area. However, due to the threshold voltage drop needed to keep the NMOS network conducting, the high or logic ‘1’ output voltage will be less than the supply voltage, V_{DD}. In order to recover a rail-to-rail output swing, a simple inverter or buffer can be added at the output of the PTN.

One common form of the NMOS-only PTL topology is the complementary pass transistor logic (CPL) [YYN+90]. CPL has complementary inputs and outputs, and usually has CMOS inverters at its outputs to restore the degraded PTL voltage swing due to the NMOS threshold voltage (V_{TH}) drop. Fig. 3.1b shows a CPL implementation of an OR/NOR gate. A number of arithmetic building blocks have been reported using CPL [YYN+90, AKBE96, CSB+97], taking advantage of the low input capacitance and reduced transistor count to increase performance.

A variation of CPL is the double pass transistor logic (DPL) [SOS+93, CSB+97]. It is composed of an NMOS PTN and its PMOS equivalent, as shown in Fig. 3.1c. The additional

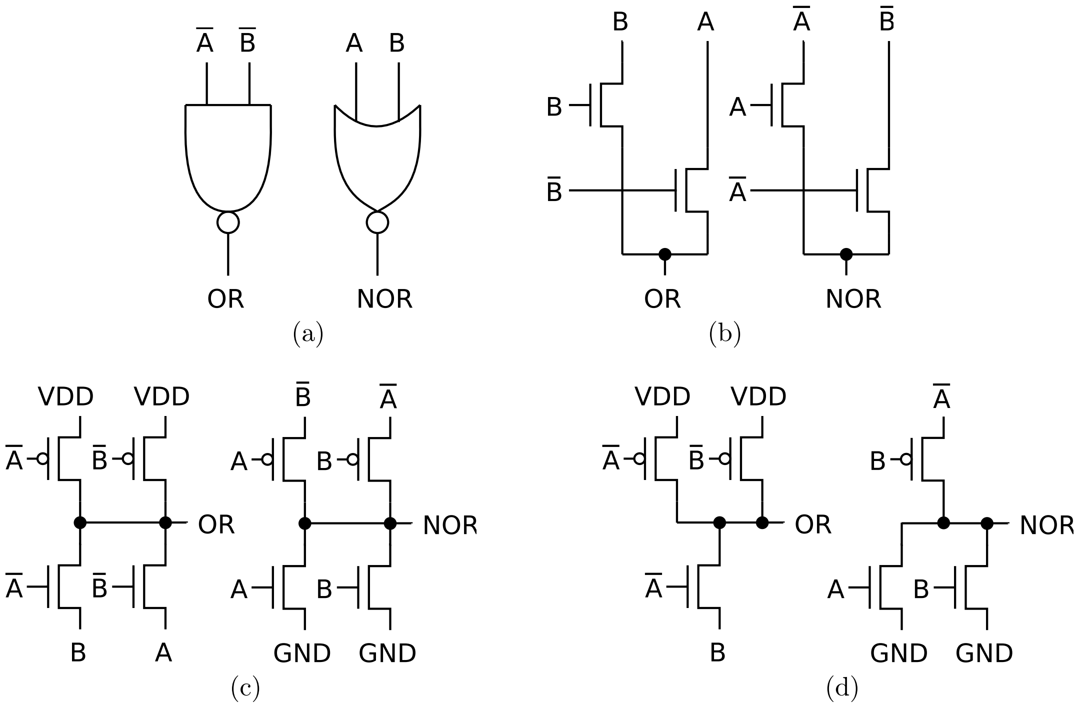
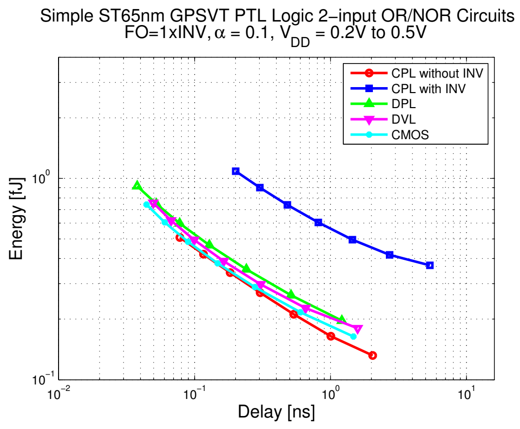


Figure 3.1: OR/NOR implementations in (a) CMOS, (b) CPL, (c) DPL and (d) DVL.

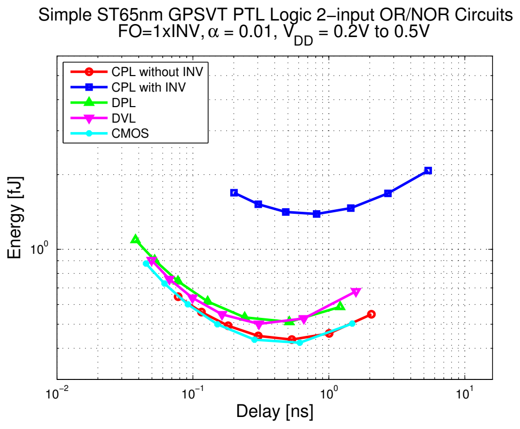
PMOS PTN (1) doubles the signal transmission path, compensating for the performance degradation due to increased path capacitance and (2) allows full swing at the output of the PTN, improving the performance at low supply voltage even with limited threshold voltage scaling. DPL however, has increased input capacitance due to the redundant signal paths present.

Dual value logic (DVL) [OD95, OD97], shown in Fig. 3.1d, is a pass transistor-based logic style derived from DPL by eliminating redundant and slower branches while still maintaining full-rail voltage swing at the output of the PTN. Note that since DPL and DVL circuits use both NMOS and PMOS PTNs, no buffering or voltage swing restoration circuitry is required at their outputs.

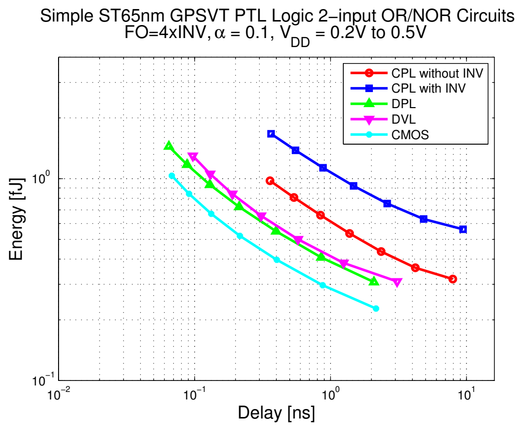
The energy-delay characteristics of 65nm two-input CPL, DPL and DVL OR/NOR PTL gates, as well as the equivalent static CMOS gate (Fig. 3.1a) are shown in Fig 3.1, for two different activity factors, $\alpha = 0.1$ and $\alpha = 0.01$ and for two different capacitive loads. All the inputs to these gates are driven by a minimum-sized inverter. Two versions of the CPL are presented, one with and one without an inverter at the output. The energy numbers include the energy of the driving gates since without these gates, the CPL without inverters at the output, being completely passive, would consume no energy from the supply rails. The supply voltage, V_{DD} is swept from 200 mV to 500 mV and the PTL gate outputs are terminated with equal load capacitances.



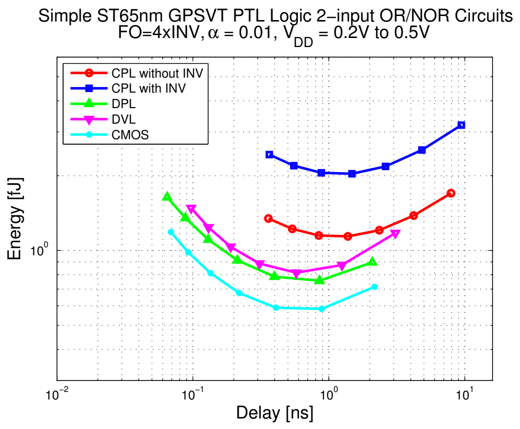
(a)



(b)



(c)



(d)

Figure 3.2: PTL 2-input OR/NOR energy-delay characteristics: (a) $\alpha = 0.1$ and loaded with a 1X INV, (b) $\alpha = 0.01$ and loaded with a 1X INV, (c) $\alpha = 0.1$ and loaded with a 4X INV and (d) $\alpha = 0.01$ and loaded with a 4X INV.

As expected, at very low performance, leakage energy due to the gain stages dominates, and is smallest in the CPL circuit without inverters, i.e., the circuit with the least number of supply rail connections, as shown in Fig. 3.2a. This leakage energy becomes more prominent at lower activity factors (Fig. 3.2b).

Gain, however, plays a very important role in driving larger capacitive loads. Figs. 3.2c and 3.2d shows how an increase in fan-out adversely affects the energy-delay characteristics of circuits without gain. Without gain, the PTL circuits incur significant increases in delay, resulting in a corresponding increase in energy. The inherent gain, and thus drive current of the static CMOS gate, on the other hand, reduces the delay impact of increasing the load capacitance. This smaller delay results in a lower overall energy for high fanout situations.

For low to moderate performance, reducing the amount of gain could lead to a reduction in

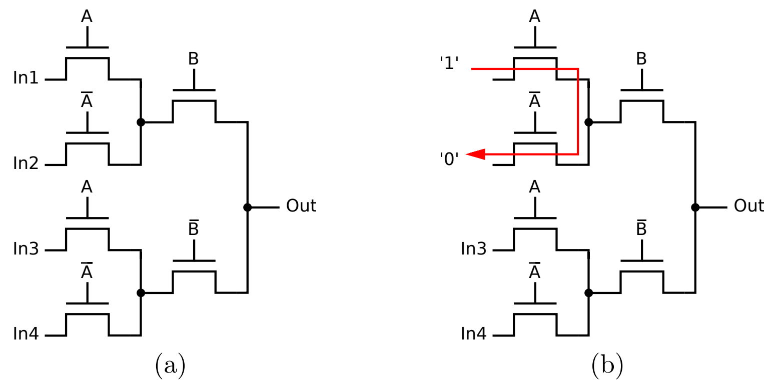


Figure 3.3: The pass transistor multiplexer tree (a) with $N_{depth} = 2$ and (b) a sneak path.

energy. In static CMOS circuits, gain can be lowered by increasing the complexity of a gate, while in PTL circuits, gain elements have to be explicitly added.

3.2 PTL Operation and Circuit Gain

The NMOS-only CPL circuit is purely passive and therefore, does not regenerate the input through gain elements such as inverters or pull-up/pull-down circuitry. This particular topology is of interest since (1) there are no supply rail connections, (2) functionality is achieved without gain and (3) energy is consumed solely through the gain elements and dissipated mostly by the pass transistor network.

3.2.1 Leakage in PTL Circuits

The fact that there are no supply rail connections implies that there are no inherent subthreshold leakage paths from V_{DD} to ground. Consider a simple CPL-based multiplexer PTN shown in Fig. 3.3a, which is a commonly used PTN topology, due to the fact that any logic function can be expanded into multiplexer functions using Shannon's expansion [Sha49].

Leakage currents can, however, still flow via sneak paths, one of which is shown in Fig. 3.3b. These paths form when two mutually exclusive inputs of a pass transistor network have opposite polarities. This behavior is similar to a static CMOS inverter, where leakage current flows through the transistor in the OFF state. The depth of the multiplexer tree, N_{depth}, increases as the number of inputs and leads to an exponential increase in the number of potential leakage paths.

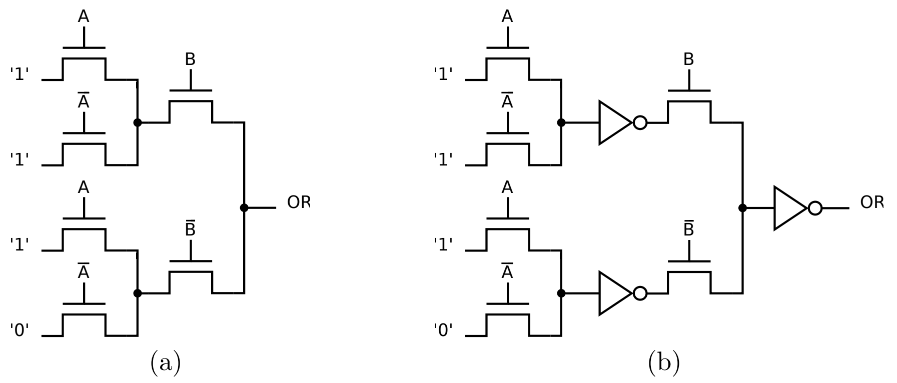


Figure 3.4: A pass transistor network (a) with no gain elements and (b) with gain at the output and internal nodes.

3.2.2 The Effect of Adding Gain

Unlike static complementary CMOS logic, the PTL logic network that determines functionality is decoupled from gain, allowing the selective addition or removal of gain elements as needed. Fig. 3.4a shows a pass transistor multiplexer tree-based OR gate without gain elements added, while Fig. 3.4b shows the same circuit but with additional gain added internally and at the output.

As can be seen from Fig. 3.5, the performance can be improved by increasing the number of gain elements, at the expense of increased energy consumption. Note that in both PTL circuits, all the gate and source inputs are driven by minimum-sized inverters. Similar to Fig. 3.1, the energy of the inverters driving the drain inputs of the pass transistor multiplexer tree is included in the total reported energy.

This additional tradeoff between energy and performance is similar to adding repeaters to long transmission lines [RCN03] and can be used as another dimension in optimizing PTL circuits.

3.2.3 Decoupled Operation

Separate optimization strategies for the pass transistor network and the gain stages can be employed due to the reduced coupling between functionality and gain. For example, consider the case of subthreshold leakage and performance. The energy consumed by the CPL multiplexer in Fig. 3.3 can be expressed as:

$$E = E_{dynamic} + E_{leakage} \tag{3.1}$$

where $E_{dynamic}$ is the energy needed to charge and discharge the capacitances in the circuit, while $E_{leakage}$ is the leakage energy dissipated by the circuit. $E_{dynamic}$ and $E_{leakage}$ for a certain operation period, T_{op}, and output swing, dV, can be further expressed as:

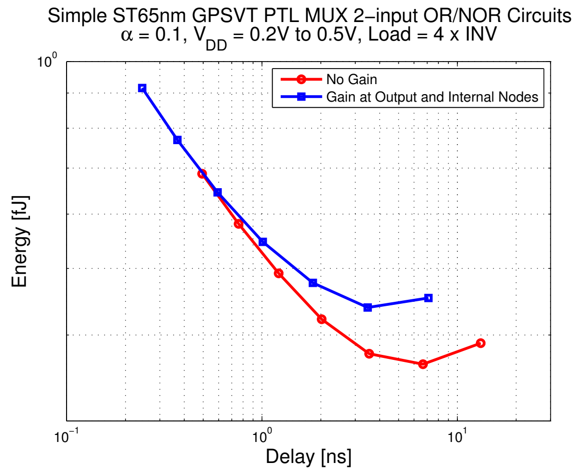


Figure 3.5: The PTL energy and delay as a function of gain.

$$E_{dynamic} = \alpha\, C_{int,driver}\, V_{DD}^2 + \alpha\, C_{PTN}\, V_{DD}\, dV + \alpha\, (C_{int,buffer} + C_{load})\, V_{DD}^2 \tag{3.2}$$

$$E_{leakage} = V_{DD}\, I_{leak,driver}\, T_{op} + V_{DD}\, I_{leak,buffer}\, T_{op} \tag{3.3}$$

Since the subthreshold leakage energy has been confined to the gain elements, and assuming sneak paths can be eliminated as described in Chapter 4, the various low leakage techniques presented in Chapter 2 can be applied selectively to these circuits, such as increasing threshold voltages, forced stacking or adding header and footer switches. Since these techniques are local to the gain elements, their impact on the pass transistor network are reduced.

The delay of the CPL multiplexer on the other hand, can be expressed as:

$$D = D_{driver} + D_{PTN} + D_{buffer} \tag{3.4}$$

A first order model [RCN03] for the delay of the pass transistor network and the CMOS inverters serving as drivers and buffers gives:

$$D \approx \frac{C_{int,driver}\, V_{DD}}{I_{ON,driver}} + \frac{C_{PTN}\, dV}{I_{ON,driver}} + R_{eff,PTN}\, C_{PTN}\, \ln 2 + \frac{(C_{int,buffer} + C_{load})\, V_{DD}}{I_{ON,buffer}} \tag{3.5}$$

The pass transistor network can be optimized for performance by using lower V_{TH} devices, reducing the effective resistance, R_{eff}, of the PTN. This decoupling of subthreshold leakage and performance is normally not possible in static CMOS circuits since the leakage path is the same as the signal path.

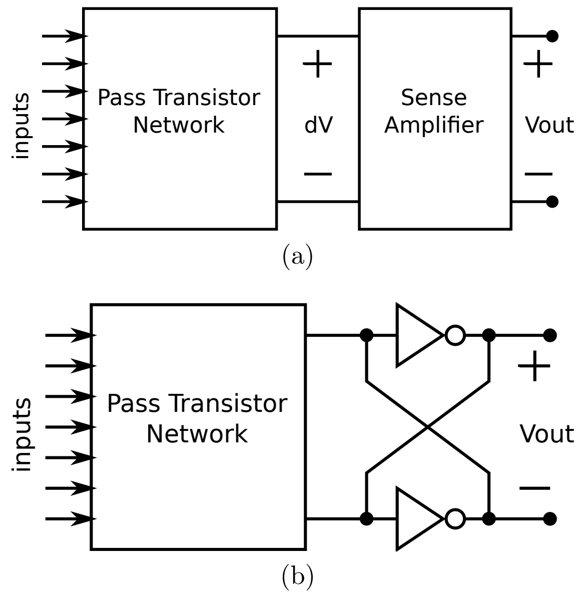


Figure 3.6: Alternative gain elements: (a) a sense amplifier and (b) a regenerative element.

3.3 Alternative Gain Elements

Static CMOS inverters suffer from increased short-circuit currents [RCN03] when its inputs are not swinging from rail to rail. Thus, other popular methods of adding gain to a PTN include using (1) a sense amplifier, or (2) a regenerative element such as a cross-coupled inverter pair.

3.3.1 Sense Amplifiers

An alternative to using CMOS inverters at the output of NMOS-only differential pass transistor networks is to use sense amplifiers as shown in Fig. 3.6a. Sense amplifiers are normally employed in circuits where performance is needed while driving heavily loaded nodes such as those found in memory circuits and long transmission lines [MHU+94].

One advantage of using a sense amplifier (SA) is the isolation of the PTN output node from the driven load. This isolation allows reduced voltage swings at the output of the PTN, thus reducing the energy consumed, while driving the load at full swing at the output of the SA. The sense amplifier can either detect a voltage difference, such as the sense amplifying pipeline flip-flop (SA-F/F) [MHU+94] or a current difference, such as the low power current sensing complementary pass transistor logic (LCSCPTL) [CL96].

In order to work with smaller voltages at the output of the PTN, the effective input offset voltage of the sense amplifier must be made significantly smaller than the SA input voltages. This input offset voltage is due to the mismatch between the SA input devices, and can be reduced by increasing the size of these input devices. Thus, a decrease in the voltage swing at the output of the PTN is almost always accompanied by an increase in the SA input capacitance, which translates into an increase in switching energy. This energy tradeoff prevents the indefinite reduction of the PTN output voltage.

3.3.2 Regenerative Elements

Another way of adding gain is through the use of regenerative elements such as cross-coupled inverters illustrated in Fig. 3.6b. These elements are connected to the output of the pass transistor network in order to boost the voltage swing all the way to V_{DD}. One example of a circuit using regenerative gain techniques is the swing-restored pass transistor logic (SRPL) [PHS96], where a latch-type swing restorer is used at the outputs of a pass transistor network. Another is the capacitor-separated pass transistor logic (CSPL) [YA00], where capacitors are used to (1) isolate the pass transistor network from the cross-coupled inverter stage and (2) remove the input offset of the inverter stage.

Due to the inherent feedback circuits in regenerative elements with two stable states, it is relatively harder to change the state of its outputs. The inputs must be able to overcome this feedback mechanism, and in the process, a significant amount of short-circuit current can flow. This short-circuit current can lead to significant increases in the energy per operation. Since less current is available to charge or discharge the capacitance of the node to be flipped, the total delay can increase as well.

3.4 Summary

A generalized model of pass transistor logic circuits is given in Fig. 3.7, and most PTL families are variations on the central theme of having a passive pass transistor network and a gain element.

Gain and functionality are inherently independent in pass transistor logic circuits due to the passive nature of the PTN. Since the leakage energy is dominated by the gain elements and delay is predominantly due to the PTN, separate optimization and design techniques can be used for these PTL components. This is normally not the case for static CMOS circuits where the signal path is the same as the leakage path, thus, strategies that reduce the signal path resistance to increase performance usually results in reduced resistance during the OFF path as well, resulting in increased leakage.

Chapter 4 introduces the concept of the sense amplifier-based pass transistor logic (SAPTL) as a logic topology that takes advantage of this decoupled nature of PTL circuits in order to reduce the energy per operation of digital logic circuits.

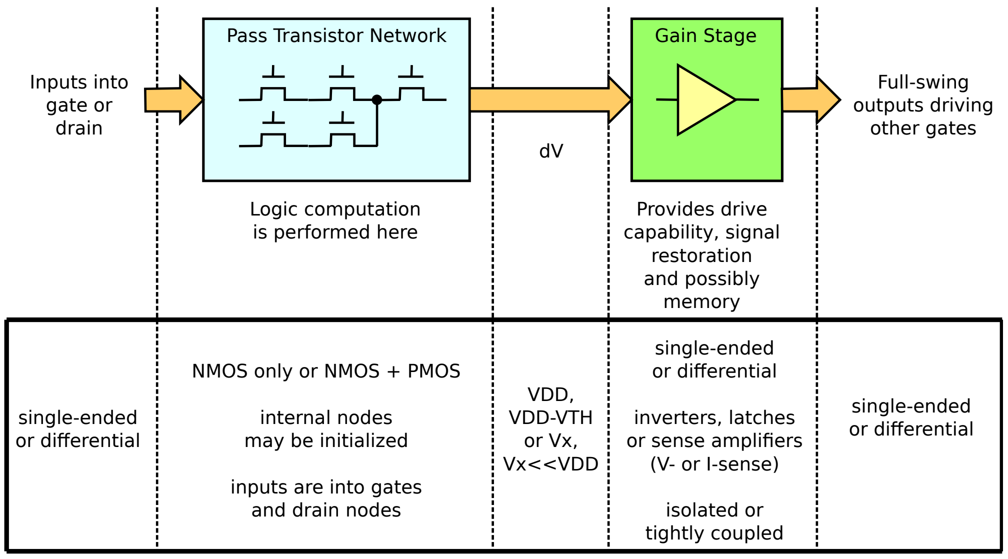

Inputs into gate or drain
Pass Transistor Network
Logic computation is performed here
dV
Gain Stage
Provides drive capability, signal restoration and possibly memory
Full-swing outputs driving other gates
single-ended or differential
NMOS only or NMOS + PMOS
internal nodes may be initialized
inputs are into gates and drain nodes
VDD, VDD-VTH or Vx, Vx<<VDD
single-ended or differential
inverters, latches or sense amplifiers (V- or I-sense)
isolated or tightly coupled
single-ended or differential

Figure 3.7: A generalized PTL model.

Chapter 4

The SAPTL Organization

The passive nature of pure pass transistor networks can provide logic functionality at the expense of reduced performance and noise margins. By adding gain elements to the pass transistor network, performance and noise margins can be improved at the cost of increasing the energy required per operation. However, since PTL functionality and gain are inherently decoupled, the tradeoff between energy and performance can be exploited at a finer granularity, when compared to conventional static CMOS logic gates.

In this chapter the sense amplifier-based pass transistor logic (SAPTL) concept is introduced, followed by a detailed description of its various components. The differences between the SAPTL topology and conventional pass transistor logic (PTL) are discussed, then an analysis of the performance and energy consumption of the inverted PTN or stack, driver, and the sense amplifier is presented.

4.1 The SAPTL logic block

The SAPTL logic block [ALPR07], shown in Fig. 4.1, is composed of three main elements: (1) the inverted pass transistor tree or the stack, (2) the root driver and (3) the sense amplifier. The SAPTL achieves low-energy operation (1) by decoupling the subthreshold leakage current from the stack threshold voltage, $V_{TH,stack}$, allowing for increased performance without the corresponding increase in leakage energy commonly associated with low-V_{TH} operation, and (2) by confining subthreshold leakage currents to well-defined and controllable paths found only in the root driver and sense amplifier.

The total energy consumed by the SAPTL is composed of the following: (1) the energy used by the root driver to energize the stack, (2) the energy used by the sense amplifier to resolve the correct logical levels and drive the inputs of the fan-out stacks and (3) the energy needed to generate the appropriate timing information, either globally, such as in clock distribution networks, or locally, such as in asynchronous handshaking circuits.

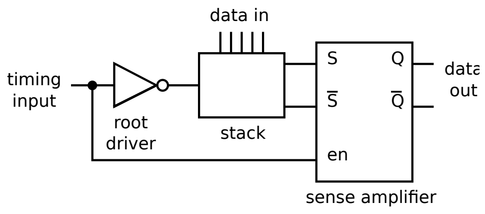


Figure 4.1: The SAPTL logic block.

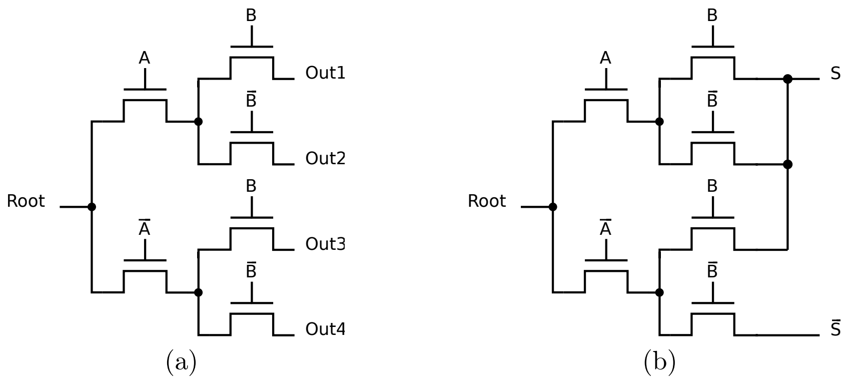


Figure 4.2: The inverted pass transistor tree network (a) with two inputs and (b) configured as a two-input OR/NOR stack.

4.2 The Inverted PTN Multiplexer Tree

One critical disadvantage of using the multiplexer PTN network in Fig. 3.3 is the presence of sneak paths. This disadvantage is due to the fact that multiple drain inputs are present, making it possible to form feedback loops where leakage currents can flow. A major contribution of this thesis is the use of an inverted pass transistor tree network shown in Fig. 4.2a. This inverted PTN network, or stack, only has one drain-connected input, called the root, and outputs that are connected to either of the two pseudo-differential outputs, labeled S and $\overline{S}$. Note that by only having a single root input, all current paths are directed exclusively to one of the two outputs, completely eliminating the possibility of forming sneak paths.

Fig. 4.2b shows how the stack can be configured as a two-input OR/NOR circuit. Each path from the root to either of the outputs represents a minterm or maxterm of the desired logic function, as seen in Fig. 4.3a. Thus, to realize a particular function, all the minterm paths must be connected to the S output, and the maxterm paths connected to the $\overline{S}$ output.

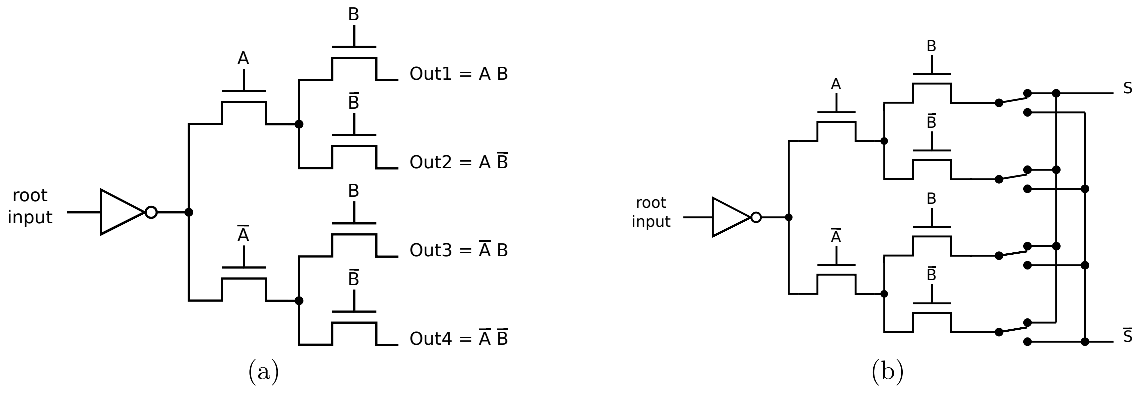


Figure 4.3: The full stack and driver showing (a) the logic paths and (b) switches for reconfigurability.

4.3 The Full Stack

A full stack is a stack that explicitly contains all 2^K possible paths for K complementary inputs, as shown including the root driver, in Fig. 4.3a. This implies that the depth of the stack, N_{depth}, for all paths is:

$$N_{depth} = K \tag{4.1}$$

N_{depth} can also be defined as the number of transistors in series from the root of the stack to the outputs, and due to the nature of the full stack, it is the same for every path. The signal path through the full stack is similar to the signal flow in a binary decision diagram (BDD), where each input forces the signal to choose between one of two possible paths at every node until it reaches one of the stack outputs. Depending on the logic function or application, the full stack can be minimized, as discussed in Sec. 4.6, to reduce area, energy or delay at the expense of flexibility.

Implementing logic functions using the full stack can be done (1) at design time, by connecting the appropriate paths to either S or $\overline{S}$, as seen in Fig. 4.2b, or (2) during run time, as a reconfigurable or programmable logic element by adding programming switches at the stack output as shown in Fig. 4.3b.

The gate or data input capacitances of the full stack can be made equal by making the transistors closer to the root of the stack twice as large as the next one further away from the root. Thus, if the transistors furthest from the root are made minimum-sized with gate capacitance C_G, then the transistors closest to the root will have a gate capacitance of:

$$C_{in} = 2^{N_{depth}-1} C_G \tag{4.2}$$

where each data input to the full stack will then see a input capacitance of C_{in}.

Note that all the paths to the output are equivalent, and only one path is energized under any input. Using a simple first order RC estimate of the stack propagation delay shows that increasing the transistor sizes in this manner has the effect of decreasing the propagation delay of the full stack by reducing the resistance of the signal path near the root of the tree [Elm48].

The root driver, in this case, a simple CMOS inverter, injects an evaluation current into the root of the stack. This causes the voltage of either S or $\overline{S}$ to increase as the selected path through the stack is energized. Alternative root drivers, such as current sources, can also be used. However, due to the simplicity of the CMOS inverter, as well as its ability to provide a good logic '0' or ground at its output, makes it ideal for the SAPTL initialization, as will be seen in Chapter 5.

4.4 The Simple Stack

Since only one path at a time is energized for a full stack, a representative pass transistor structure called the simple stack is presented as a first order approximation of the full stack. The effect of V_{DD}, V_{TH} and N_{depth} on the stack delay and energy are illustrated. The analysis is then extended to the full stack. Note that in order to explore V_{TH} space larger than what body biasing can provide, the simulation technique described in Appendix A is used.

A simple stack is defined as a static CMOS inverter driving N_{depth} pass transistors in series and capacitively loaded at the output as shown in Fig. 4.4a.

4.4.1 A First Order Delay Model

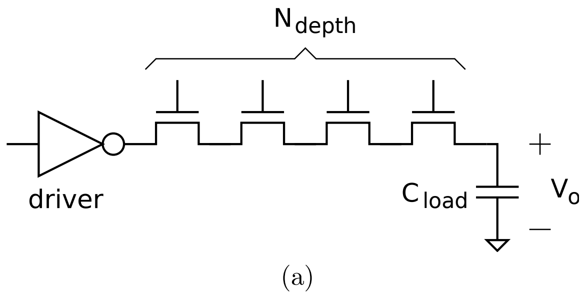
The equivalent rising transient model is shown in Fig. 4.4b. The capacitances associated with each node are:

$$C_{root} = 3C_{int,Driver}S_{Driver} + C_{int,stack} \tag{4.3}$$

$$C = 2C_{int,stack} \tag{4.4}$$

$$C_{load} = C_{int,stack} + C_{fanout} \tag{4.5}$$

Where C_{root} is the stack root capacitance, which includes the output capacitance of the driver, $C_{int,Driver}$ and $C_{int,stack}$ are the intrinsic source/drain junction capacitances of the minimum-sized driver and stack transistors respectively, S_{Driver} is the size of the driver relative to minimum size, and C_{fanout} is capacitive load being driven by the simple stack.



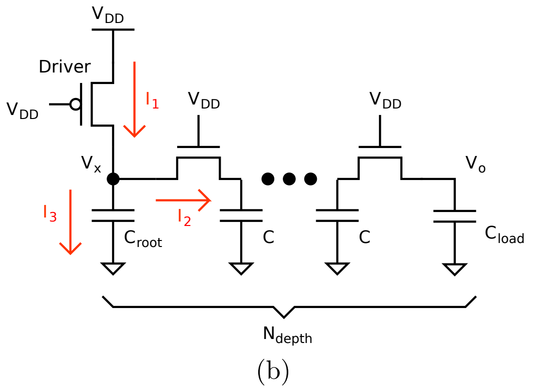


Figure 4.4: The simple stack (a) schematic and (b) rising transient model.

The EKV transistor model [EKV95], due to its accuracy at low supply voltages near the threshold voltage, is used as a basis for deriving the energy and delay models of the simple stack. The inversion coefficient [MWA+10], IC, indicates the degree of transistor inversion. In the subthreshold region, $IC < 1$, while in the super-threshold region, $IC > 1$. Thus, the inversion coefficient can be expressed as

$$IC = IC\left(V_{DD}, V_{TH}\right) = \log\left(e^{\frac{(1+\sigma)V_{DD}-V_{TH}}{2nV_T}} + 1\right)^2 \tag{4.6}$$

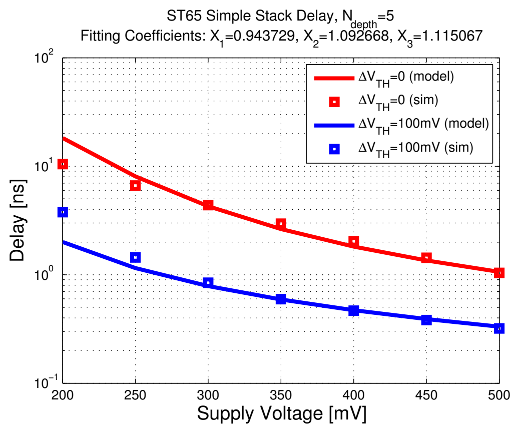
where $V_T = \frac{kT}{q}$, and set n such that $IC = 1$ when $V_{DD} = V_{TH}$, giving

$$n = \frac{\sigma V_{TH}}{2V_T \log\left(e^1 - 1\right)} \tag{4.7}$$

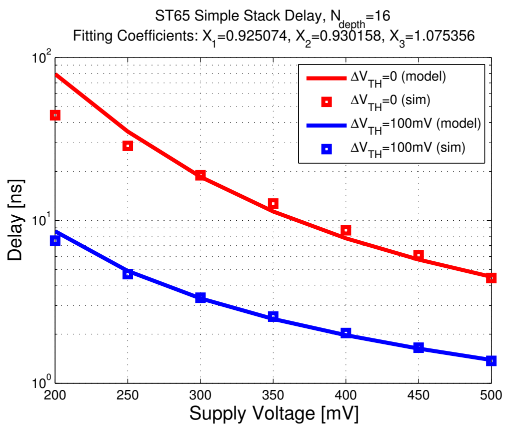
Thus, the MOS transistor drain current can be expressed as

$$I_D = I_S \cdot IC = 2n\beta \frac{W}{L} V_T^2 \cdot IC \tag{4.8}$$

where I_S, also called the specific current, is the drain current when the supply voltage is equal to the threshold voltage and in this case, $W = W_{\min}$ and $L = L_{\min}$.



(a)



(b)

Figure 4.5: The simple stack delay model for (a) $N_{depth} = 5$ and (b) $N_{depth} = 16$.

Using this model, the rising delay at the stack output can be approximated by an effective stack capacitance, C_{stack} being driven by an effective stack current I_{stack} until the voltage reaches an effective output voltage V_o, which is a fraction of the supply voltage. Thus,

$$D_{SS} \approxeq \frac{C_{stack} V_o}{I_{stack}} = X_3 \frac{C_{root} + C\left(N_{depth}^{2X_2} + N_{depth}\right) + C_{load}}{I_{S,stack} \cdot IC\left(X_1 \cdot V_{DD}, V_{TH,stack}\right)} V_{DD} \tag{4.9}$$

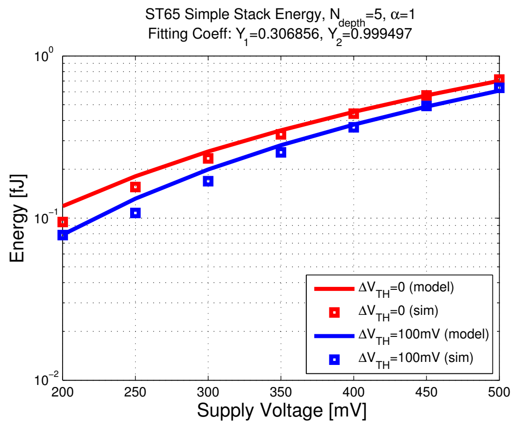
where X_i are fitting coefficients.

Since all transistors are minimum sized, the delay from the root of the stack to its outputs, or *stack delay*, dominates the total delay, since it is much greater than the delay from the input of the driver to the stack root node, or *driver delay*. For larger and more complex stacks though, the size and thus, the output current of the root driver can significantly affect the total delay. Note that the delay model is made to increase quadratically with the stack depth similar to the Elmore delay [Elm48] of an RC network.

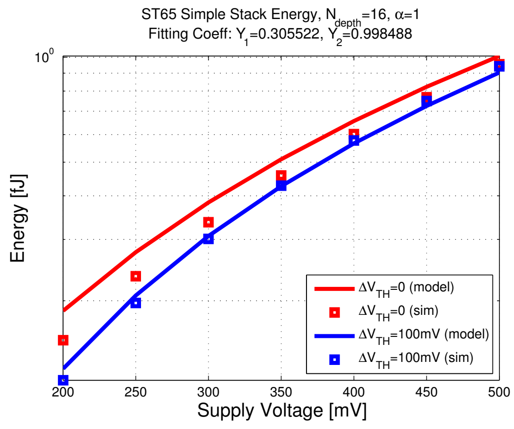
Figs. 4.5a and 4.5b compares the behavior of the simulated simple stack and the corresponding model results for stack depths of 5 and 16 respectively for two different effective threshold voltages.

4.4.2 Simple Stack Energy

The simple stack energy can be estimated by the energy needed by the driver to charge the output node capacitance up to a certain voltage level, and can be expressed as the sum of two components: (1) $E_{SS,stack}$, the energy needed to charge up the root capacitance as well as all the internal node capacitances and (2) $E_{SS,leak}$, the leakage current of the driver. Thus, the total stack energy, E_{SS}, can be expressed as:



(a)



(b)

Figure 4.6: The simple stack energy mode for (a) $N_{depth} = 5$ and (b) $N_{depth} = 16$.

$$E_{SS} = E_{SS,stack} + E_{SS,leak} \tag{4.10}$$

where $E_{SS,stack}$ and $E_{SS,leak}$ are given by

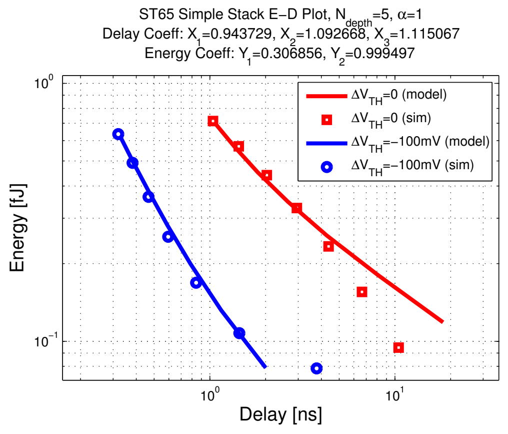
$$E_{SS,stack} = Y_1 \left[2S_{Driver}C_{root}V_{DD}\left(V_{DD} - \Delta V_{TH}\right) + \left(C\left(N_{depth} - 1\right) + C_{load}\right)V_{DD}^2\right] \tag{4.11}$$

$$E_{SS,leak} = Y_2 S_{Driver} V_{DD} I_{leak,Driver} D_{SS} \tag{4.12}$$

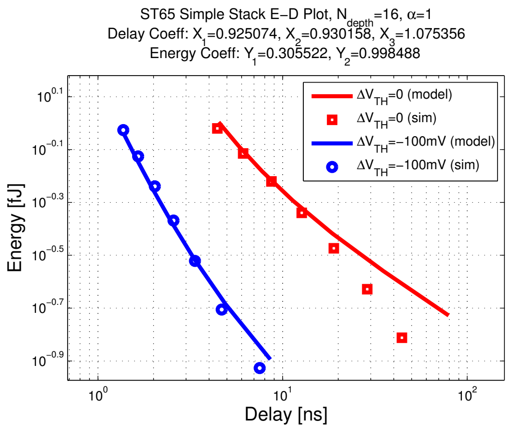
where the constants Y_i are fitting parameters. Note that Y_1 gives an indication of the stack output voltage swing, as a fraction of V_{DD}. Eq. 4.11 assumes that the root drive can be turned off as soon as the stack output reaches this voltage swing, and also takes into account the smaller root voltage needed to produce the same output voltage when the stack threshold voltage is reduced by ΔV_{TH}. A key observation here is that in reducing the stack threshold voltage, both the energy and delay values are reduced.

The simulated and estimated simple stack model energy as a function of supply voltage for stack depths of 5 and 16 are shown in Figs. 4.6a and 4.6b. Figs. 4.7a and 4.7b shows the model energy-delay characteristics, compared to the simulation results, for $N_{depth} = 5$ and $N_{depth} = 16$ respectively.

For supply voltages below 300mV, the stack output voltage can be as low as 100mV or even less. At these output voltages, the current models are less accurate due to the very small MOS drain-to-source currents, and coupled with the distributed nature of the stack capacitances that is not captured by Eqs. 4.9 and 4.11, the discrepancies between the simulated and theoretical energy and delay values are larger. Note that at lower stack threshold voltages,



(a)



(b)

Figure 4.7: The simple stack energy-delay model for (a) $N_{depth} = 5$ and (b) $N_{depth} = 16$.

the stack output voltages are slightly larger, resulting in smaller differences between the simulated and theoretical results.

4.5 Analysis of The Differential Full Stack

The delay of the full stack is similar to the simple stack with the addition of two parasitic effects: (1) the leakage currents through the OFF paths, and (2) the added parasitic capacitances present in the full stack. The stack output rising transient model is shown in Fig. 4.8.

The effective capacitances are now different for each stage of the full stack due to the sizing scheme that balances the input capacitances of the stack, as described in Section 4.3. This sizing scheme, as discussed in Sec. 4.7, allows the implementation of the full stack as an array of simple stacks. The node on-path capacitances are also scaled as

$$C_i = 2^{N_{depth}+1-i} C_{int,stack} \quad i \in 1, 2, \ldots, N_{depth} - 1 \tag{4.13}$$

and the root capacitance as

$$C_{root} = 3C_{int,Driver} S_{Driver} + 2^{N_{depth}} C_{int,stack} \tag{4.14}$$

The effective load capacitance is now dependent on the chosen logic function of the full stack. For K inputs, P_S is defined as the number of paths connected to the S output, where $0 \le P_S \le 2^K$. Similarly, the number of paths connected to the $\overline{S}$ output is $P_{\overline{S}} = 2^K - P_S$. Thus, the ON-path load capacitance can be estimated as:

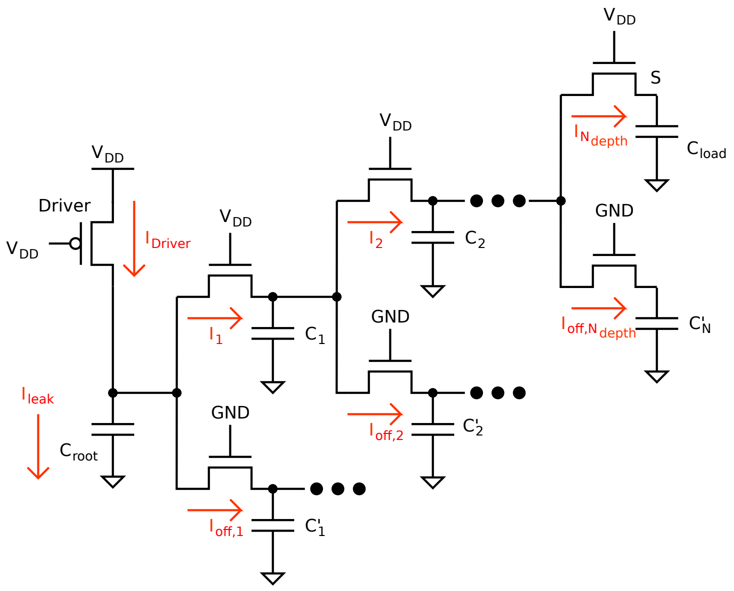


Figure 4.8: The full stack rising transient model.

$$C_{load} = P_S C_{int,stack} + C_{fanout} \tag{4.15}$$

Due to the larger full stack root capacitance, the driver delay contribution becomes significant, and can be expressed as

$$D_{FS,Driver} = X_1 \frac{2S_{Driver}C_{root}}{I_{S,Driver} \cdot IC\left(V_{DD}, V_{TH,Driver}\right)} V_{DD} \tag{4.16}$$

The off-path current, I_{OP}, also becomes non-negligible since reduces the available on-path current used to charge the full stack's internal node capacitances. I_{OP} increases exponentially with the depth of the full stack and can be modeled as

$$I_{OP} = 2^{N_{depth}-1} I_{leak,OP} \tag{4.17}$$

where $I_{leak,OP}$ is the leakage current of one stack transistor. Note that even though this is a transistor leakage current, it can still be considered as contributing to dynamic energy since (1) it is charging a capacitance, in this case the capacitances of the off-path, and (2) it only flows when the stack is energized.

The full stack delay can then be expressed as

$$D_{FS} = D_{FS,Driver} + X_3 \frac{X_2 2^{N_{depth}} C + C_{load}}{I_{S,stack} \cdot IC\left(V_{DD}, V_{TH,stack}\right) - I_{OP}} V_{DD} \tag{4.18}$$

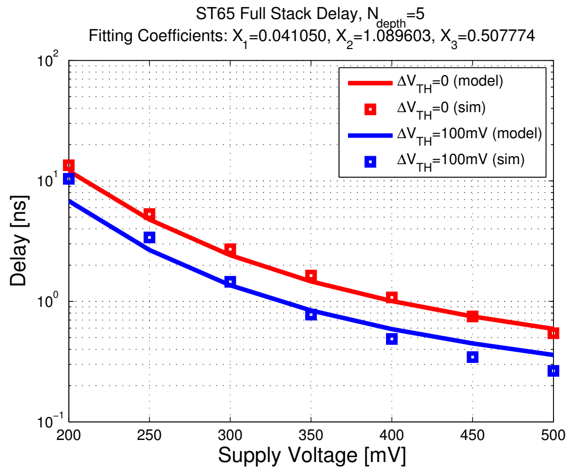


Figure 4.9: The full stack delay model.

where X_i are fitting constants. Note that the total on path capacitance increases exponentially with the depth of the stack, as a consequence of Eq. 4.13. Fig. 4.9 shows the calculated delay of a full stack with $N_{depth} = 5$ as a function of supply voltage for two stack threshold voltages using Eq. 4.18 alongside the simulated delay values.

The energy of the full stack is composed of (1) $E_{FS,stack}$, the on-path energy, (2) $E_{FS,OP}$, the off-path energy and (3) $E_{FS,Driver}$, the driver leakage energy, and each of these energy components are expressed as

$$E_{FS,stack} = 2Y_1 S_{Driver} C_{root} V_{DD} \left(V_{DD} - \Delta V_{TH}\right) + Y_1 \left(C \left(N_{depth} - 1\right) + C_{load}\right) V_{DD}^2 \tag{4.19}$$

$$E_{FS,OP} = Y_3 V_{DD} D_{FS} I_{OP} \tag{4.20}$$

$$E_{FS,Driver} = Y_2 S_{Driver} V_{DD} I_{leak,Driver} D_{FS} \tag{4.21}$$

where again, Y_i are constants used to fit the full stack energy model to simulated data. Thus, the total full stack energy given by

$$E_{FS} = E_{FS,stack} + E_{FS,Driver} + E_{FS,OP} \tag{4.22}$$

Figs. 4.10a and 4.10b show the full stack energy as the supply voltage is swept from 0.3V to 0.5V and the full stack energy-delay plot over the same supply voltage range.

Again, since the distributed nature of the full-stack is not being accurately modeled by Eqs. 4.18 and 4.22, discrepancies are observed between the simulated and theoretical results.

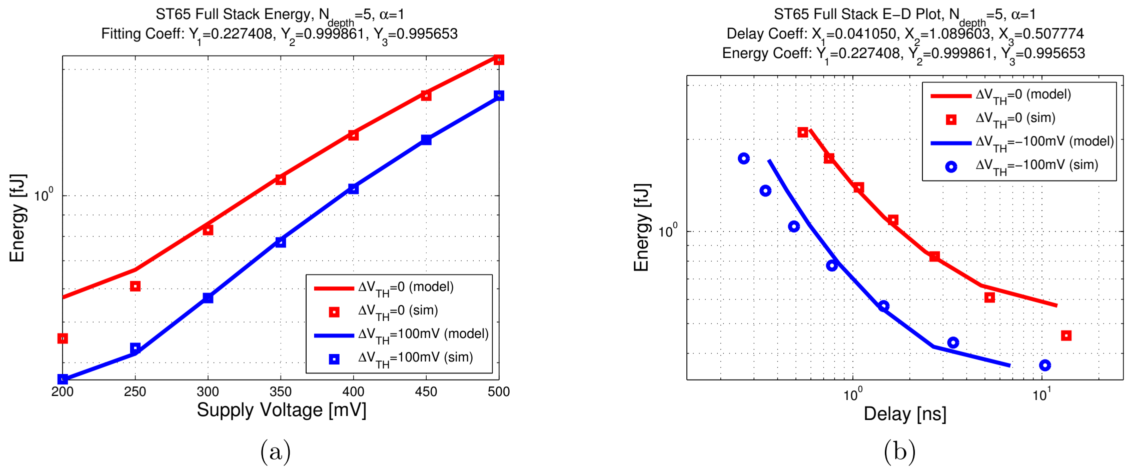


Figure 4.10: The full stack energy model (a) vs. V_{DD} and (b) vs. delay.

These discrepancies are largest at low supply voltages as well as at low stack threshold voltages, where the current flows through the off-paths are estimated and not modeled detail.

Implementing logic functions using a K-input full stack is convenient since it is functionally equivalent to a K-input lookup table (LUT), which is the basic building block of field-programmable gate arrays (FPGAs). Thus, FPGA synthesis and optimization tools can easily be modified to accommodate the SAPTL as the unit logic element [PR07] for both field-programmable and mask-programmable designs. The cost for using full stacks in some cases, could be very prohibitive since the input capacitance, parasitic capacitances and silicon area increase exponentially with the number of inputs. Thus, using a stack tailored to a specific logic function could provide significant energy and performance improvements.

4.6 Stack Minimization

The differential cascode voltage switch logic (DCVSL) [HGDT84], shown in Fig. 4.11 uses a pseudo-differential NMOS pull-down network similar to the SAPTL stack. Algebraic techniques such as the identification of common subexpressions or much simpler and more practical Karnaugh map (K-map) or tabular techniques [CP86] have been used to generate DCVSL trees. These techniques result in reduced transistor counts by sharing as much commonality as possible between the paths that are connected to output, or S for the case of the SAPTL stack, and those that are connected to its complement, or $\overline{S}$.

Minimizing the full stack can lead to a significant reduction in stack energy and delay and this energy and delay reduction is dependent on the logic function that is implemented. As an example, a minimized stack that performs the XOR logic function is shown in Sec. 4.6.1. This minimized XOR stack is compared to its full stack equivalent in terms of both energy and delay.

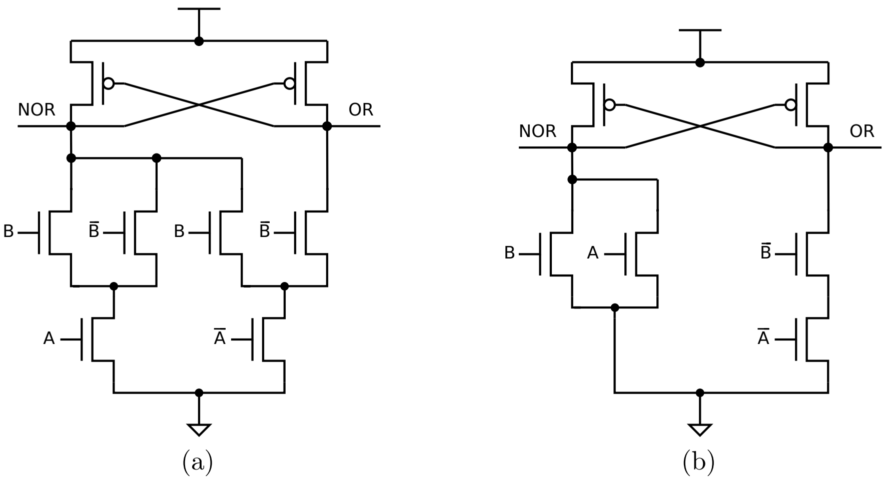


Figure 4.11: The DCVSL OR/NOR gate with (a) a full pull-down tree and (b) a minimized pull-down tree.

4.6.1 The XOR Logic Function

A major component of arithmetic circuits such as adders and multipliers are exclusive-or (XOR) gates. The XOR function of two variables can be expressed as:

$$X_1 \oplus X_2 = X_1\overline{X_2} + \overline{X_1}X_2 \tag{4.23}$$

Extending this definition to three variables gives:

$$X_1 \oplus X_2 \oplus X_3 = \left(X_1\overline{X_2} + \overline{X_1}X_2\right)\overline{X_3} + \overline{\left(X_1\overline{X_2} + \overline{X_1}X_2\right)}X_3 \tag{4.24}$$

Note that an XOR function can be reduced recursively. If the operations are carried out serially, then each variable selects either the complemented or uncomplemented version of the previous XOR operation. Thus, in general,

$$F\left(X_1, \ldots, X_{n-1}\right) \oplus X_n = F\left(X_1, \ldots, X_{n-1}\right)\overline{X_n} + \overline{F\left(X_1, \ldots, X_{n-1}\right)}X_n \tag{4.25}$$

Fig. 4.12a shows an illustration of Eq. 4.25 for one stage of the recursive XOR implementation. A pass transistor implementation of this stage is given in Fig. 4.12b.

A minimized stack realizing a 3-input XOR function, representing the sum function of a full-adder, with a root driver, is shown in Fig. 4.14a. If a full stack is used to realize this function, it would require $3 \cdot 2^3 = 24$ minimum-sized transistors, growing exponentially with the number of inputs, instead of $3 \cdot 4 = 12$, which increases linearly with the number of inputs. This assumes that the two transistors closest to the root in Fig. 4.14a are sized twice as wide compared to the rest of the stack.

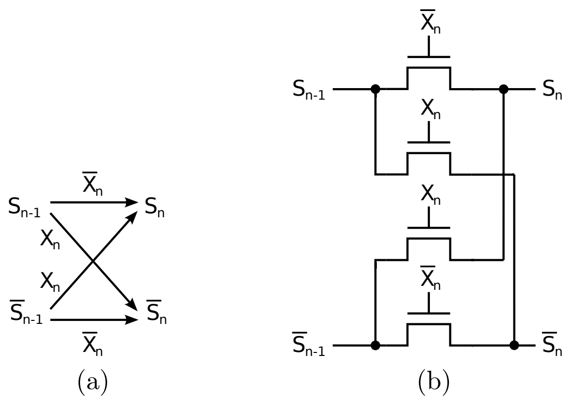


Figure 4.12: The XOR (a) signal flow and (b) the pass transistor circuit segment.

Table 4.1: The XOR stack transistor count.

XOR inputs	Full stack	Full stack (unit size)	Min. stack	Min. stack (unit size)
2	6	8	6	8
4	30	64	14	16
6	126	384	22	24
n	$2 \cdot (2^n - 1)$	$n \cdot 2^n$	$2 + 4 \cdot (n-1)$	$4n$

Table 4.1 shows the transistor counts for varying XOR stack inputs for both full and minimized stacks. Expressing the comparison in terms of the number of unit-sized (minimum sized) transistors highlights the exponential area and capacitance reduction that can be achieved by minimizing the stack.

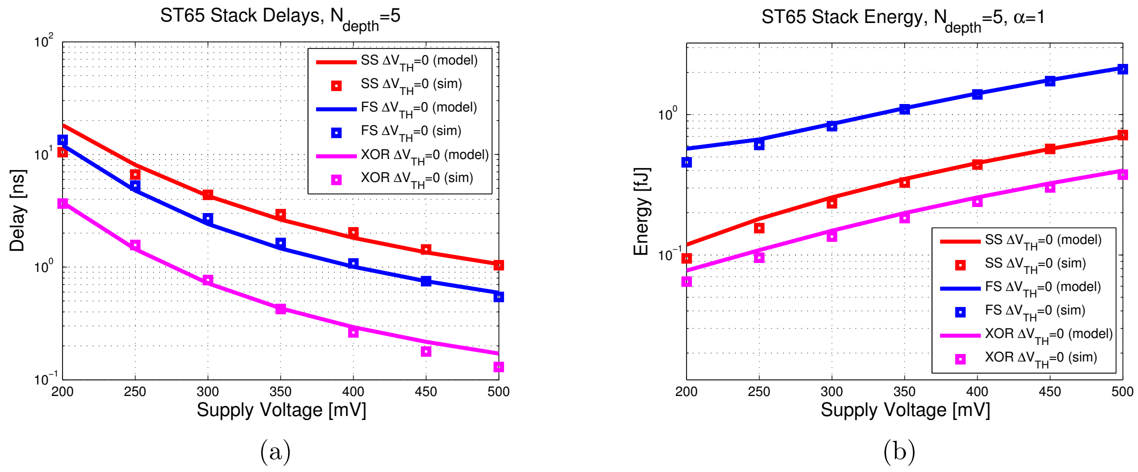
Another advantage of the minimized stack lies in the fact that as the number of inputs is increased, the input capacitance at each data input remains constant at $2C_G$ instead of growing exponentially with the number of inputs as given by Eq. 4.2.

Since the effective transistor sizes along the length of the on-path is constant, the delay of the XOR stack is similar to that of the simple stack, except for the inclusion of the driver delay due to the larger root capacitance. Thus,

$$D_{XOR} = D_{XOR,Driver} + X_3 \frac{C\left(N_{depth}^{2X_2} + N_{depth}\right) + C_{load}}{I_{S,stack} \cdot IC\left(V_{DD}, V_{TH,stack}\right) - I_{OP,XOR}} V_{DD} \tag{4.26}$$

where $D_{XOR,Driver}$ is of the same form as $D_{FS,Driver}$ given in Eq. 4.16. $I_{OP,XOR}$ is the off path current, and instead of growing exponentially with the stack depth, for an XOR stack, it is a constant, and is estimated to be flowing in 3 out of the possible 4 root paths. Thus,

$$I_{OP,XOR} = 3 \cdot I_{leak,OP} \tag{4.27}$$



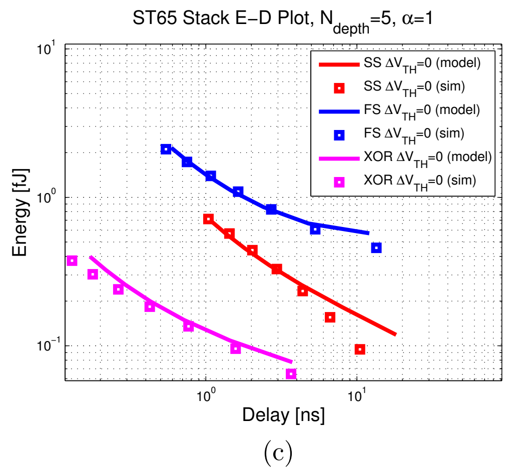
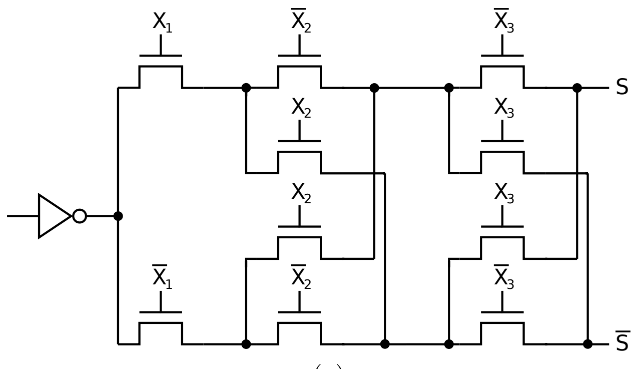


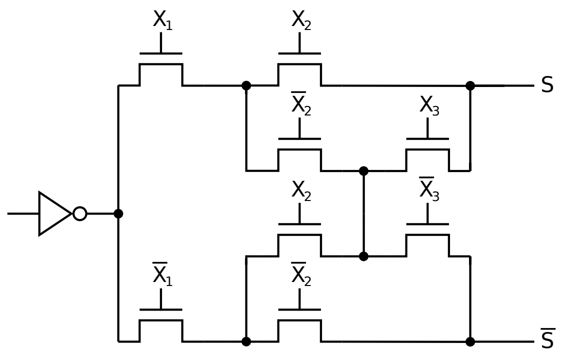
Figure 4.13: The (a) delay, (b) energy and (c) energy-delay plots of the simple stack (SS), full stack (FS) and XOR stack.

The XOR stack energy is of the same form as Eqs. 4.19 to 4.22, except for the off path current, which is given in Eq. 4.27. Comparing the XOR stack with the full and simple stacks in Fig. 4.13, it can be seen that a 3X delay and 5X energy reduction can be achieved through minimization.

Note that the delay of the simple stack is larger than both the full and XOR stacks since due to the minimum-sized transistors used in the simple stack, unlike the scaled transistors of the full stack or the larger root transistors in the XOR stack. Due to its relatively larger delay, the simple stack energy is lower than the full stack energy but higher than the XOR stack.



(a)



(b)

Figure 4.14: The minimized stack-based full-adder (a) sum stage and (b) carry-out stage.

4.6.2 A Full-Adder Implementation

The carry-out function of a full-adder can be expressed as

$$C_{out} = X_1X_2 + X_2X_3 + X_3X_1 \tag{4.28}$$

and using the DCVSL tree generation method found in [CP86], the resulting carry-out stack is shown in Fig. 4.14b. A full-adder using two full stacks requires 48 minimum sized transistors in the stack alone, as compared to 24 transistors in the minimized stack implementation.

Minimizing the stack, however, can result in variable path delays due to unequal path lengths leading to larger data dependent delay variations. This delay spread can be minimized by adding dummy transistors, as seen in Fig. 4.15, where "always-on" transistors are added in series with the shorter paths. Adding extra transistors to match delays can be advantageous in synchronous systems to prevent unnecessarily large voltage swings at the output of the stack, but can be detrimental in asynchronous systems that can take advantage of the average path delay instead of always waiting for the worst-case path.

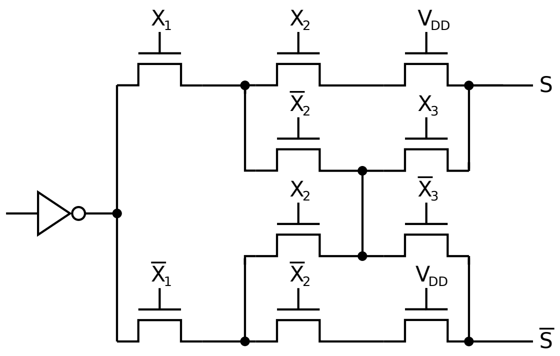


Figure 4.15: The carry-out full-adder function with balanced stack delays.

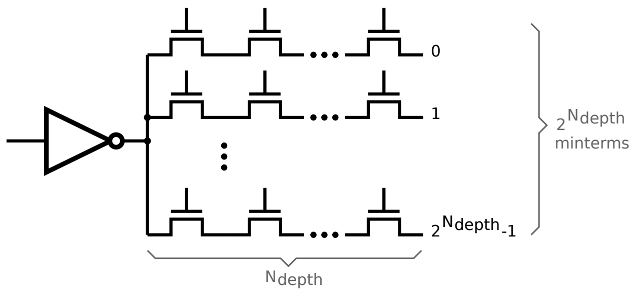


Figure 4.16: Full stack implementation using an array of minimum-sized transistors.

4.7 Stack Implementation and Variability

The full stack can be implemented using $2^{N_{depth}}$ simple stacks, where each simple stack implements a minterm or maxterm, as seen in Fig. 4.16. If all the simple stacks are made up of minimum-sized transistors, then all the paths from the root of the stack to either S or $\overline{S}$ will be the same. This very regular full stack implementation can reduce the effect of random variations especially for deeper stacks [KRH$^+$05].

Note that the capacitance at the stack outputs are dependent on the logic function being implemented. This function-, and thus data-dependency however, can easily be predicted due to the regular nature of the full stack. By adding additional capacitances at the stack outputs can reduce these functionality-dependent delay and energy variations at the expense of increased delay and energy.

In certain cases, however, making the full stack delay and energy independent of functionality or input data may not be needed. For example, in asynchronous environments, lower average energy and delay numbers might be more desirable that circuits with smaller energy and delay variances.

4.8 Differential Stack Operation

Even though the delay is referenced at $V_o = 0.5V_{DD}$, the stack outputs may not always be sampled at this point. Due to the data dependency of the stack delays as well as other non-local or global timing constraints, the sampling by, for example, a voltage-sensing sense amplifier, can occur (1) earlier, which can lead to voltages that are too small to be detected, or (2) later, in which case, the leakage current in the off path can have enough time to charge up the off path output capacitance resulting in a smaller differential output voltage.

Early sampling can be avoided by slowing down the whole system to accommodate the slowest stack. Late sampling however, can pose problems if the system contains stacks with varying depths. If the delay spreads are large enough, the outputs of the stacks with shallow depths are sampled very late, leading to reduced voltage headroom between S and $\overline{S}$.

4.8.1 The Stack Output Voltage

The differential stack output voltage, dV, can be expressed as:

$$dV = \frac{\Delta t}{C}\left(I_{o,S} - I_{o,\overline{S}}\right) \tag{4.29}$$

where $I_{o,S}$ and $I_{o,\overline{S}}$ are the stack output currents at nodes S and $\overline{S}$ respectively. When the stack is energized, and if the on path current flows out of the S output, then the off path leakage current will flow out of the $\overline{S}$ output. At supply voltages higher than 0.5V, $I_{o,S} \gg I_{o,\overline{S}}$, resulting in a dV of around $\frac{V_{DD}}{2}$. However, as V_{DD} is reduced, the on path current becomes smaller, and if V_{TH} is also lowered, the off path current increases. Thus, reducing V_{DD} and/or V_{TH} effectively reduces the $\frac{I_{o,S}}{I_{o,\overline{S}}}$ ratio. This reduction in the on current to off current ratio reduces the differential stack output voltage swing, making it harder for the sense amplifier to resolve the correct stack output state.

4.8.2 The Weak Output Latch

A simple way of increasing the differential voltage at the output of the stack (dV) is to add a weak output latch connected to S and $\overline{S}$ as shown in Fig. 4.17. The cross-coupled long-channel length NMOS pair with $L = 4L_{min}$ provides the stack off-path leakage current a path to ground, thus increasing dV, while ensuring correct functionality.

Fig. 4.8.2 shows the behavior of dV and maximum stack output voltage, $V_{o,\max}$ needed to achieve this dV, as the supply voltage is swept from 0.3V to 0.5V, for two different stack depths. For both stack depths, the weak latch allows (1) a larger dV to develop at the stack outputs, around 50mV more for a stack depth of 5 and about 20mV more for a stack depth

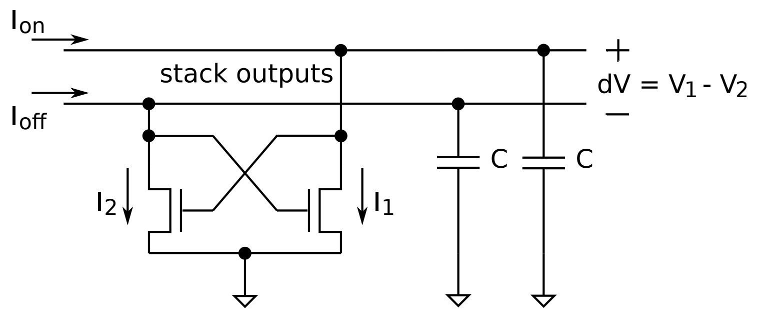


Figure 4.17: The weak stack output latch.

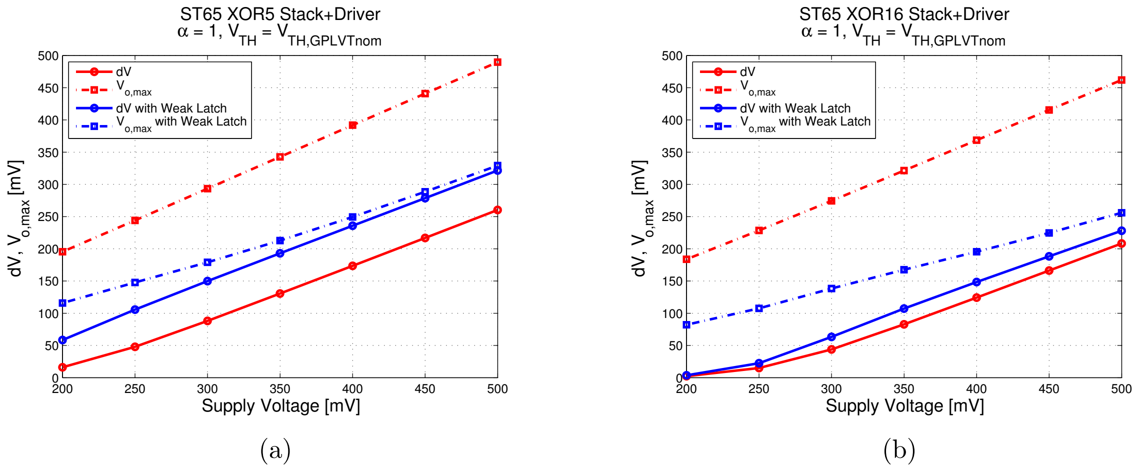


Figure 4.18: The effect of the weak output latch on dV and $V_{o,\max}$ as a function of V_{DD} for (a) XOR5 and (b) XOR16 stacks.

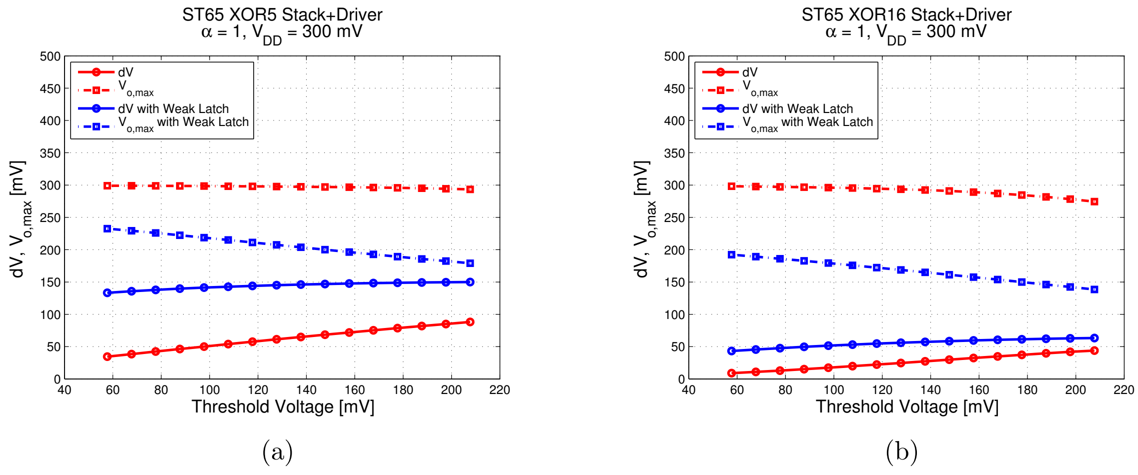


Figure 4.19: The effect of the weak output latch on dV and $V_{o,\max}$ as a function of $V_{TH,stack}$ for (a) XOR5 and (b) XOR16 stacks.

of 16, easing the sense amplifier requirements, and (2) a lower $V_{o,\max}$, approximately 100mV to 200mV less, reducing the energy needed to charge up the stack node capacitances.

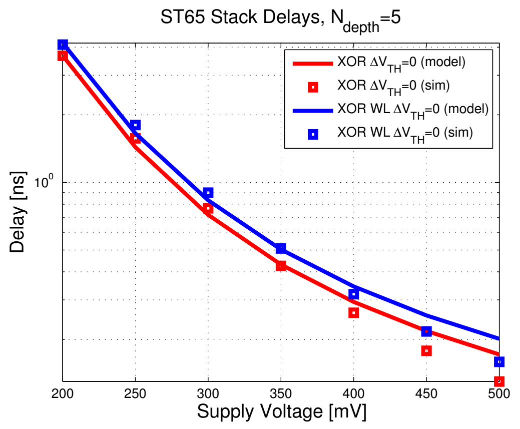
Simulation results showing the effect of the weak latch on the same two XOR stacks, as V_{TH} is reduced, are shown in Fig. 4.19, this time for a fixed supply voltage of 0.3V. Again, due to the weak latch, the dV increases while the $V_{o,\max}$ needed to produce this dV decreases.

Adding the weak latch, however, introduces (1) additional capacitances at the output of the stack, that increases both stack delay and energy, and (2) leakage paths to ground. The leakage path is only conducting current when the stack is energized, thus it is desirable to keep the stack energized only when needed. This can be made possible using an asynchronous timing scheme discussed in Chapter 5.

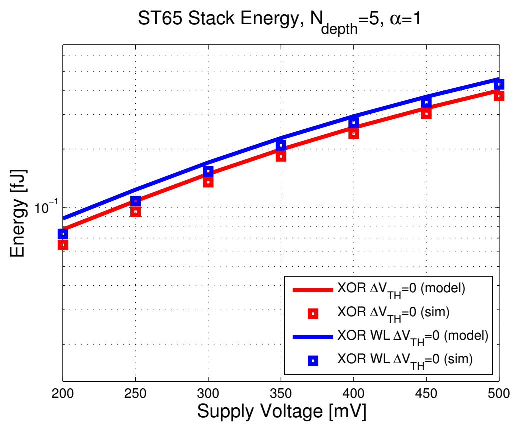
The resulting energy and delay characteristics of the XOR5 stack with and without a weak latch are shown in Fig. 4.20. As expected, both the stack delay and energy increases due to the addition of the weak latch. This increase in energy and delay can be traded off for less stringent sense amplifier design requirements, which can eventually translate in an overall SAPTL performance improvement or energy reduction.

4.9 The Sense Amplifier

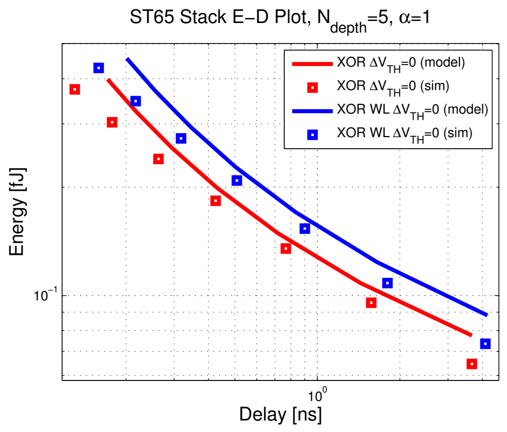
Sense amplifiers can either sense a current difference or a voltage difference. Current sense amplifiers requires a relatively large $\frac{I_{on}}{I_{off}}$ ratio at the output of the stack. However, due to the small gain provided only by the stack driver, and the non-negligible off-path currents, a current sense amplifier would not be a good gain element at the output of the stack. This bad match between the stack and a current sensing scheme is even more pronounced at low



(a)



(b)



(c)

Figure 4.20: The (a) delay, (b) energy and (c) energy-delay characteristics of an XOR5 stack with and without a weak output latch.

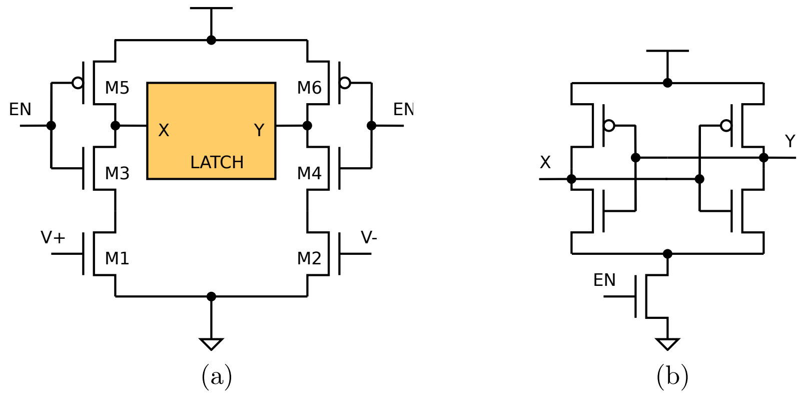


Figure 4.21: The basic SAPTL sense amplifier (a) pre-amplifier and (b) latch.

stack threshold voltages and very deep stacks. Voltage sense amplifiers on the other hand, can make use of the input transistor's transconductance, to amplify the voltage difference at the output of the stack and produce larger current differences. These current differences can then be used to determine the state of a latch. The ability of a voltage sense amplifier to correctly resolve the state of its input is mostly dependent on the mismatch between the input devices. This mismatch can be reduced but at the cost of increased delay and energy. In this work, a voltage sense amplifier is used at the output of the stack to provide gain.

4.9.1 Sense Amplifier Design

The sense amplifier (SA) serves three purposes: (1) it amplifies the low-voltage stack output, dV, ensuring that the output swings from rail-to-rail; (2) it serves as a buffer stage at the output of the stack, so as to improve the overall SAPTL performance; and (3) it precharges the SAPTL outputs to V_{DD}, allowing the reset of the driven fan-out stacks, as discussed in Chapter 5.

The SA shown in Fig. 4.21 consists of two stages. The first stage in Fig. 4.21a acts as a pre-amplifier to reduce the impact of mismatch and to provide amplification to the low-swing stack output. The second stage is a cross-coupled latch, shown in Fig. 4.21b, which retains the result of the SAPTL logic operation even after the SA inputs are pulled back down to ground or logic '0'.

The sense amplifier is constrained by the minimum differential input voltage, dV_{min}, it needs to correctly make a decision in an acceptable amount of time. Another constraint is the output drive strength needed to drive a certain number of fanout gates.

4.9.1.1 Input Detection and Mismatch

The threshold voltage and drain current mismatch is inversely proportional to the square-root of the transistor active area:

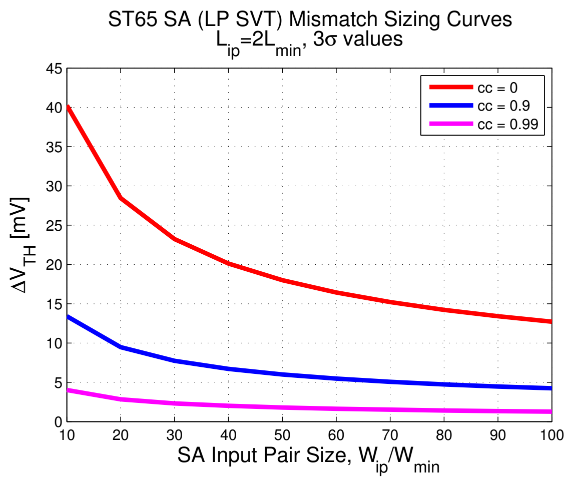


Figure 4.22: Sense amplifier input transistor ΔV_{TH} as a function of input transistor width.

$$\sigma_{\Delta V_{TH}} \approx \frac{K_{V_{TH}}}{\sqrt{WL}} \tag{4.30}$$

$$\sigma_{\frac{\Delta I_D}{I_D}} \approx \sigma_{\frac{\Delta \beta_{eff}}{\beta_{eff}}} \approx \frac{K_{\beta_{eff}}}{\sqrt{WL}} \tag{4.31}$$

The input offset voltage can then be expressed as:

$$V_{os,SA} \approxeq 3\sigma_{\Delta V_{TH}} + \frac{3\sigma_{\frac{\Delta I_D}{I_D}} I_D}{G_m} \approx 3\sigma_{\Delta V_{TH}} \tag{4.32}$$

Thus, the minimum differential input voltage must be:

$$dV = V_S - V_{\overline{S}} > V_{os,SA} = 3\sigma_{\Delta V_{TH}} = 3\frac{K_{V_{TH}}}{\sqrt{WL}} \tag{4.33}$$

Fig. 4.22 shows the relationship between ΔV_{TH} and the SA input pair width for three different correlation coefficient parameters for simulation. These correlation coefficients, or *cc*, increases as better layout matching techniques are used in the physical design of the SA input pair.

Increasing the size of the SA input differential pair reduces the required voltage swings at the output of the stack. However, this voltage swing reduction is accompanied by an increase in SA subthreshold leakage current, and more importantly, an increase in SA input capacitance. This increase in input capacitance increases the stack delay, as well as the switching energy needed to charge this capacitance up to V_S or $V_{\overline{S}}$.

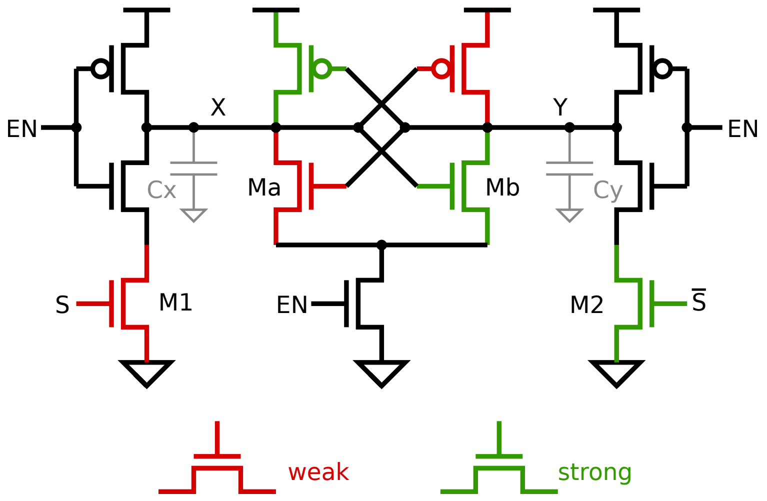


Figure 4.23: The sense amplifier worst-case mismatch scenario.

In order to size the the sense amplifier transistors, Fig. 4.23 shows a worst-case mismatch scenario, where the S input is energized, and a certain amount of voltage rise on the $\overline{S}$ input due to off path stack leakage. In this scenario, transistors M_1 and M_a are assumed to be weaker due to an increase in V_{TH}, while transistors M_2 and M_b are stronger, with smaller threshold voltages.

Mismatch in the latch transistors, in the form of an additional offset at nodes X and Y, must also be considered. If V_S is applied to the S input, and $V_{\overline{S}}$ to the $\overline{S}$ input, then immediately after enabling the sense amplifier, the worst-case mismatch inversion coefficients of the input and latch NMOS transistors, excluding the latch tail transistor, are given by

$$IC_1 = \log\left(e^{\frac{V_S+\sigma V_{DD}-V_{TH}-\frac{\Delta V_{TH}}{2}}{2nV_T}}+1\right)^2 = \log\left(e^{\frac{V_S+\sigma V_{DD}-V_{TH}-\frac{3}{2}\frac{K_{V_{TH}}}{\sqrt{W_{IP}L_{IP}}}}{2nV_T}}+1\right)^2 \tag{4.34}$$

$$IC_2 = \log\left(e^{\frac{V_{\overline{S}}+\sigma V_{DD}-V_{TH}+\frac{3}{2}\frac{K_{V_{TH}}}{\sqrt{W_{IP}L_{IP}}}}{2nV_T}}+1\right)^2 \tag{4.35}$$

$$IC_a = \log\left(e^{\frac{(1+\sigma)V_{DD}-V_{TH}-\frac{3}{2}\frac{K_{V_{TH}}}{\sqrt{W_{latch}L_{latch}}}}{2nV_T}}+1\right)^2 \tag{4.36}$$

$$IC_b = \log\left(e^{\frac{(1+\sigma)V_{DD}-V_{TH}+\frac{3}{2}\frac{K_{V_{TH}}}{\sqrt{W_{latch}L_{latch}}}}{2nV_T}}+1\right)^2 \tag{4.37}$$

Thus, in order to correctly drive the latch, the voltage drop in X after the enable signal is asserted must be greater than the voltage drop in Y, or $\Delta V_X > \Delta V_Y$, and assuming $C_X = C_Y$, the following condition must be satisfied:

$$I_1 + I_a = I_X > I_Y = I_2 + I_b \tag{4.38}$$

and expressing the currents in terms of the inversion coefficients and specific currents,

$$I_{S,1} IC_1 \frac{W_{IP}}{L_{IP}} + I_{S,a} IC_a \frac{W_{latch}}{L_{latch}} > I_{S,2} IC_2 \frac{W_{IP}}{L_{IP}} + I_{S,b} IC_b \frac{W_{latch}}{L_{latch}} \tag{4.39}$$

Eq. 4.39 can then be used to size the SA input pair and the latch transistors. Expressing this inequality in terms of $\frac{I_X}{I_Y}$ yields

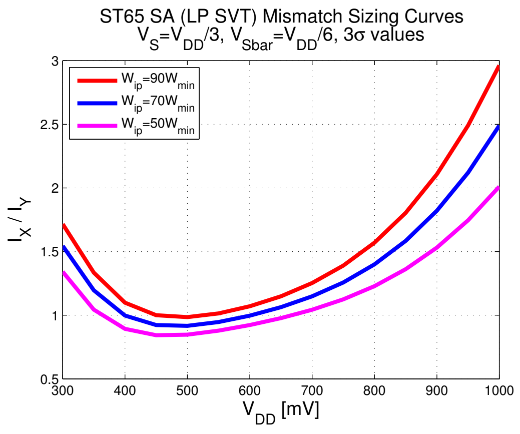
$$\frac{I_X}{I_Y} = \frac{I_{S,1} IC_1 \frac{W_{IP}}{L_{IP}} + I_{S,a} IC_a \frac{W_{latch}}{L_{latch}}}{I_{S,2} IC_2 \frac{W_{IP}}{L_{IP}} + I_{S,b} IC_b \frac{W_{latch}}{L_{latch}}} > 1 \tag{4.40}$$

and is plotted in Fig. 4.24a as a function of supply voltage for three different input transistor width sizes. For supply voltages relatively far from V_{TH}, the transistors can be made large enough to minimize the effect of mismatch and the $\frac{I_X}{I_Y}$ ratio is dominated by the effect of V_S and $V_{\overline{S}}$. However, when V_{DD} is close to the threshold voltage, the mismatch in threshold voltage, ΔV_{TH}, can be large enough to move M_1 into the subthreshold region and M_2 into the super-threshold region, significantly reducing the margins between I_X and I_Y. This effect is seen in the sense amplifier Monte Carlo simulations in Fig. 4.24b, where a slight drop in SA correct decision rates occur near V_{TH}.

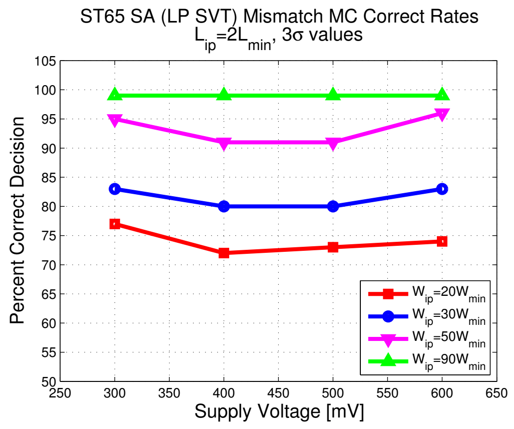
Thus, in the case of the 65nm SA, the setting the width of the input pair to $90W_{\text{min}}$ results in a 3σ-worst-case threshold voltage mismatch of 15mV, thus allowing the detection of signals as small as 50mV at a supply voltage of 0.5V.

4.9.1.2 Output Drive

Additional gain stages or buffers, in this case, CMOS inverters, can be added at the outputs of the sense amplifier to achieve a certain delay constraint in driving the output capacitive load and to isolate the internal SA X and Y nodes from the effect of the fanout topology, at the expense of increased leakage. To maintain the correct precharge state in order to initialize the driven stacks, an even number of inverters must be used, as shown in Fig. 4.25. Sizing an inverter chain for a certain delay and energy requirement has been well studied and can the methods in [RCN03] can be directly applied.



(a)



(b)

Figure 4.24: The effect of V_{DD} on (a) the modeled SA $\frac{I_X}{I_Y}$ ratio and (b) the Monte Carlo simulation of the correct SA decision rates.

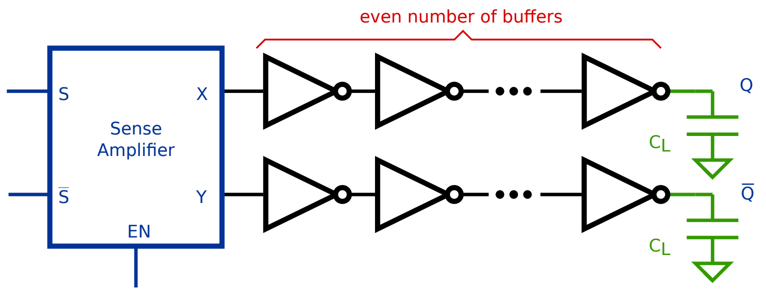
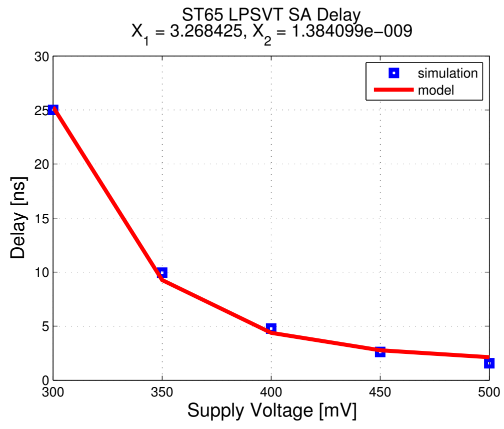
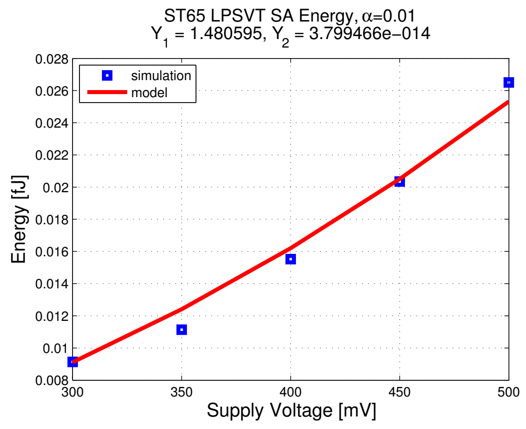


Figure 4.25: Buffering the sense amplifier output.



(a)



(b)

Figure 4.26: The sense amplifier (a) delay and (b) energy models.

4.9.2 Delay and Energy

The delay of the sense amplifier core, excluding the inverter chain, can be estimated as

$$D_{SA} = X_1 \frac{C_X V_{DD}}{I_{SA}} + X_2 \frac{I_Y}{I_X} \tag{4.41}$$

where X_i are fitting parameters. The first term represents the delay needed to completely discharge node X or Y through the pre-amplifier and latch transistors. The second term represents the increase in latch delay due to the smaller initial voltage difference between nodes X and Y when the ratio $\frac{I_X}{I_Y}$ is small. However, as seen in Fig. 4.26a, this factor is negligible.

The sense amplifier energy can be expressed as the sum of the effective switching energy at nodes X and Y and the subthreshold leakage current drawn from the supply rails, thus

$$E_{SA} = Y_1 \alpha C_X V_{DD}^2 + Y_2 V_{DD} I_{SA,leak} D_{SA} \tag{4.42}$$

where Y_i are fitting factors. Fig. 4.26b shows both the estimated and simulated sense amplifier energy as a function of the supply voltage.

The total SAPTL energy, E_{SAPTL}, can then be expressed as the sum of the energy expressions for the full-stack, driver and the sense amplifier, as given by Eqs. 4.22 and 4.42. In order to evaluate the energy needed to implement a function with a certain number of inputs, it is useful to define the normalized SAPTL energy per input, $\frac{E_{SAPTL}}{N_{depth}}$. Fig. 4.27a shows the normalized energy of a full-stack SAPTL gate as a function of stack depth, and in the case of the full-stack, N_{depth} is equal to the fan-in of the SAPTL gate. As can be seen in this figure, at lower stack depths, the energy of the sense amplifier dominates. As the stack depth is

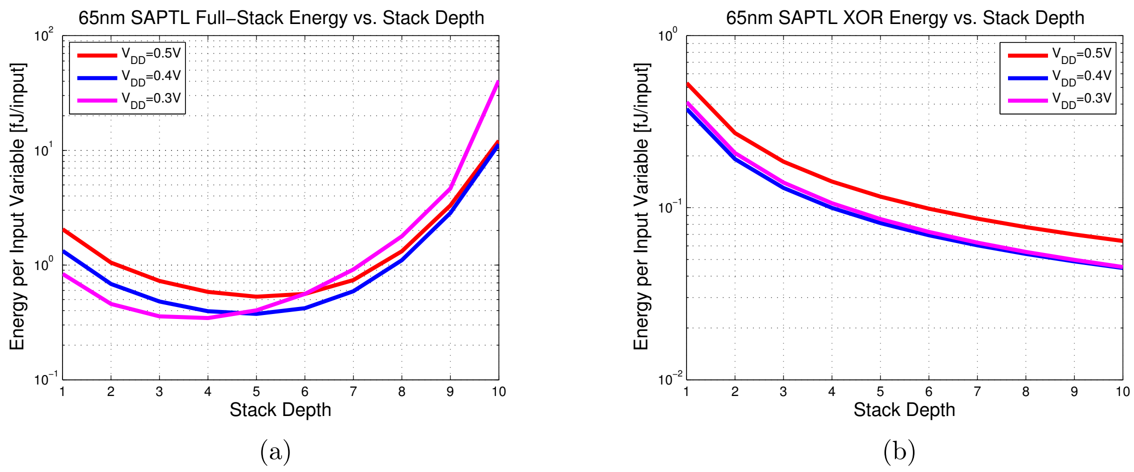


Figure 4.27: The SAPTL energy as a function of stack depth for (a) a full-stack and (b) for an XOR stack.

increased, the energy per input is reduced since the total energy is being shared over several inputs. At stack depths greater than 6, the energy again increases due to the exponential increase in transistor count, and hence switched capacitance, in the full stack. In this case, a stack depth of 5 provides the best energy per SAPTL gate input.

By simplifying the stack, the total energy of the stack does not grow exponentially, allowing for deeper stack depths. This is the case for the XOR stack, shown in Fig. 4.27b. Thus, for very deep XOR stacks, the stack depth is now limited by the sense amplifier's ability to determine the correct stack output state. Note that the graphs in Fig. 4.27 uses the same sense amplifier for all stack depths and the energy needed to generate timing information is not considered.

4.10 Decoupling Functionality and Gain

The inherent separation of logical functionality and circuit gain within the SAPTL block creates three separate circuit domains that can be optimized independently of each other. The burden of implementing boolean functions falls on the stack, while gain is provided by both the driver and sense amplifier.

The delay and energy of the passive pass transistor stack can be reduced by reducing the threshold voltages of the stack transistors, as shown in Secs. 4.4 and 4.5. Since the stack does not have any gain elements, this threshold voltage reduction does not increase the overall standby leakage current. In contrast, the sense amplifier and driver leakage current make up the total SAPTL standby leakage current. Depending on the particular circuit setting the SAPTL is used in, SA and driver drive strength or robustness can be traded off for reduced standby leakage energy.

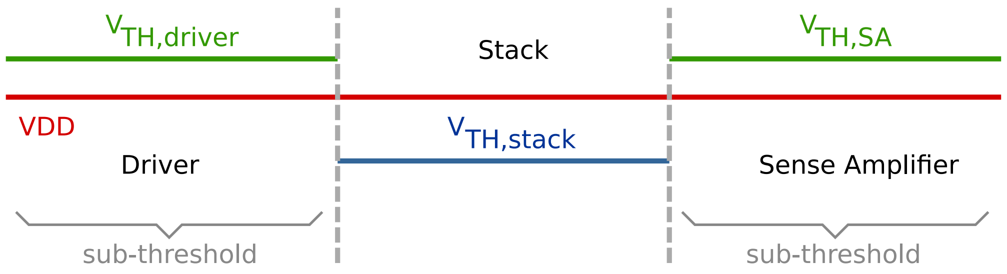


Figure 4.28: A possible SAPTL V_{DD} and V_{TH} scenario where the driver, stack and sense amplifier operate in different regions within one SAPTL block.

One possible voltage scenario is shown in Fig. 4.28, where the stack threshold voltage is set to be lower than the SAPTL supply voltage, optimizing it for performance. On the other hand, the driver and sense amplifier is made to operate in the subthreshold region, thus reducing their respective subthreshold leakage currents. This elevated SA V_{TH} can reduce the robustness of the sense amplifier by increasing the effect of process variations and transistor mismatch, which in turn, increases of the minimum dV the SA can detect.

Note that this decoupled SAPTL operation allows the application of the different leakage mitigation techniques enumerated in Chapter 2 to the SA and driver independent of the stack. Another important characteristic of the SAPTL is the possible temporal separation of the stack and SA functionality. Asynchronous timing schemes that limit the stack operation to certain time periods makes it possible to create high-leakage high-performance SA and driver modes that are only active during these times.

4.11 Summary

In order to carry out logical operations, the SAPTL needs three rudimentary building blocks: the stack, the driver, and the sense amplifier. The purpose of these building blocks are also clearly delineated from each other: the stack provides the logic functionality while the sense amplifier and driver provide gain.

The stack is a passive pass transistor network, whose resulting boolean functionality is encoded in its internal interconnection. This interconnect structure, is either programmable or fixed at design time, and along with the stack data inputs, determines the route the on path current takes through the stack. This on path current is generated by the root driver, and can exit the stack at either one of the two stack outputs, S and $\overline{S}$. The output swing and delay through the stack, as well as the energy needed to energize the on path is heavily dependent on the supply and threshold voltages, root driver current, stack capacitances and stack depth.

Using an array of simple stacks to generate a full stack takes advantage of the inherent robustness of regular structures to the effects of random variations. However, for very deep stacks, the energy and delay penalties due to the exponential increase in transistor count with N_{depth} can be very large. In the XOR stack, for example, minimizing the full stack gives a 5X energy and 3X delay reduction at the expense of reduced flexibility and regularity.

After the root driver injects current into the stack, the SA is then tasked to determine which stack output has been energized. Finally, the SA places the correct full-rail differential function result at its outputs. The built-in latch within the sense amplifier allows the SA to ignore further changes in the stack outputs after making a decision, giving the fanout blocks continued access to the SAPTL's output, as seen in Chapter 5. In order to correctly resolve the stack output voltages, the SA transistor sizes must take into account the mismatch characteristics of the technology node.

The SAPTL achieves reduced energy consumption in performing boolean operations by using a passive PTN and confining the standby leakage energy to the driver and sense amplifier. This decoupling of functionality and gain is a key SAPTL concept which allows each SAPTL component to be optimized independently. By combining the full-stack and sense amplifier models, a stack depth of 4-6 results in the minimum energy per data input. This optimal stack depth can be made larger by minimizing the stack, and thus making the SA energy the dominant energy component.

Using this basic SAPTL organization, larger logic blocks can be built, where system-level tradeoffs such as stack complexity versus sense amplifier leakage can be explored. Chapter 5 then presents timing strategies that can be used to combine and/or cascade SAPTL blocks in order to create larger circuits and systems.

Chapter 5

SAPTL Timing

Implementing logic functions in SAPTL requires the creation of a timing scheme made necessary by the inherent memory element in the sense amplifier. This also allows the decomposition of the logic functionality of a system into smaller cascaded SAPTL blocks, as well as the aggregation of larger logic blocks from smaller ones, but with less gain if the energy-delay tradeoff is acceptable.

In this chapter, the synchronous, two-phase SAPTL timing scheme is presented. This is then followed by a discussion on how asynchronous or self-timed approaches can also be utilized. The performance and energy implications of these timing schemes are then analyzed.

5.1 Synchronous SAPTL Operation

The complete SAPTL block diagram in Fig. 4.1 is repeated here as Fig. 5.1a for convenience. The output of the stack, S and $\overline{S}$, is connected to the inputs of the sense amplifier, shown in Fig. 5.1b, which then generates the logically correct full swing output$Q$ and $\overline{Q}$.

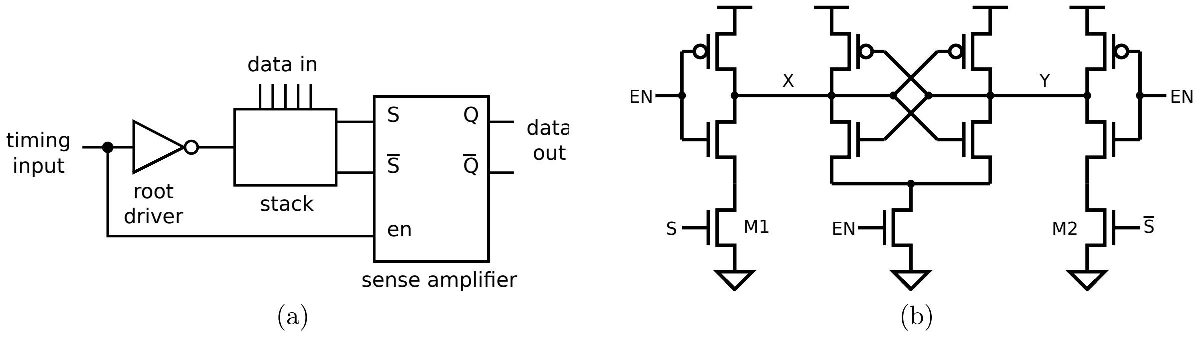


Figure 5.1: The SAPTL (a) logic block organization and (b) sense amplifier.

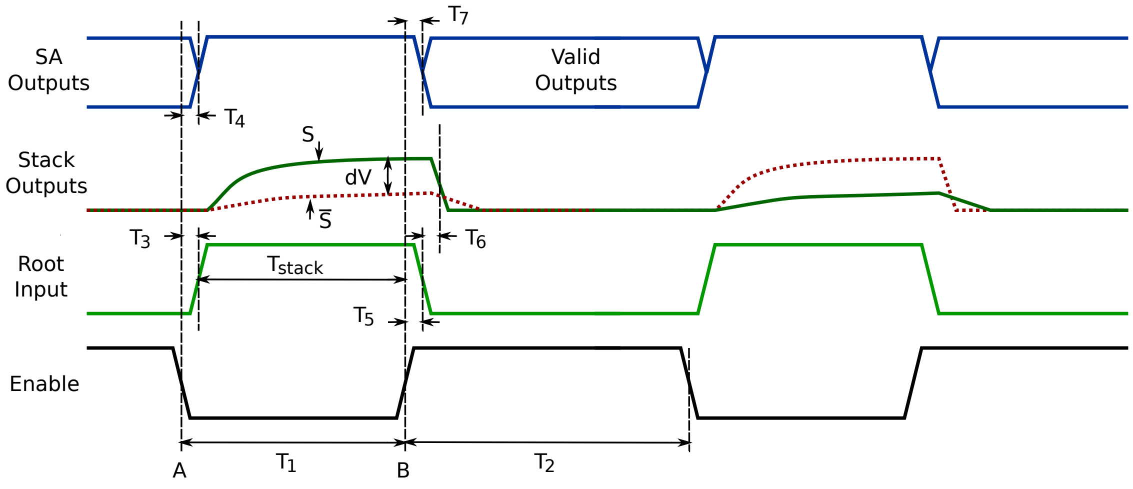


Figure 5.2: The clocked SAPTL timing diagram.

Timing is controlled by the enable signal, EN, which controls two SAPTL events: (1) the injection of a pulse into the root of the stack and (2) the triggering of the sense amplifier, moving it out of precharge mode and into sensing and eventually into storage mode, as shown in Fig. 5.2.

At time A, the root driver is triggered and the sense amplifier is placed into its reset or precharged mode. The previous result stored in the SA is deleted and the outputs are pulled high in preparation for clocking in a new result. At time B, when the stack outputs reach a sufficient dV, the sense amplifier is released from reset, samples S and $\overline{S}$, and then latches the result it deems correct. Also at this time, the root input is brought down to logic '0' and the remaining charge in the stack is drained away.

The stack delay, T_{stack}, is the time it takes for the outputs to rise above the minimum SA input voltage, $S_{\min}$, and to develop a differential voltage greater than $dV_{\min}$. Thus, the minimum time the enable must be held low, T_1, is determined by either (1) the rising delay of the root driver, T_3 and the stack delay or (2) the time needed by the sense amplifier to precharge its outputs, T_4, which can be made small. T_1 can then be expressed as:

$$T_1 > \min\left(T_3 + T_{stack}, T_4\right) \approxeq T_3 + T_{stack} \tag{5.1}$$

The minimum time required before the next SAPTL operation starts, T_2, is determined by either (1) the falling delay of the root driver, T_5 and the discharge or initialization time of the stack, T_6 or (2) the evaluation time of the sense amplifier, T_7. If the sense amplifier is assumed to be much faster than the stack time constants, then

$$T_2 > \min\left(T_5 + T_6, T_7\right) \approxeq T_5 + T_6 \tag{5.2}$$

Thus, the minimum clock period that can be used is

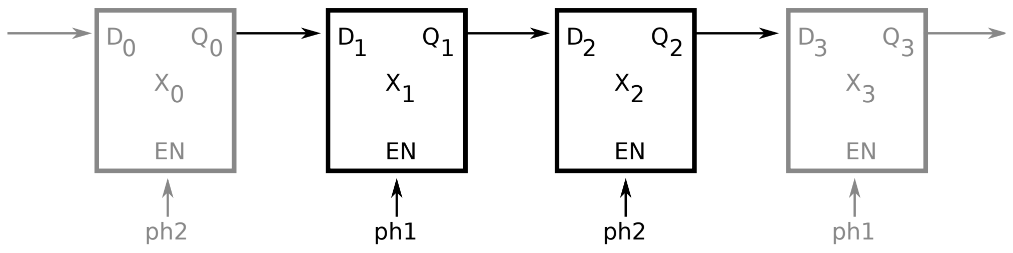


Figure 5.3: Cascaded SAPTL blocks using two-phase clocking.

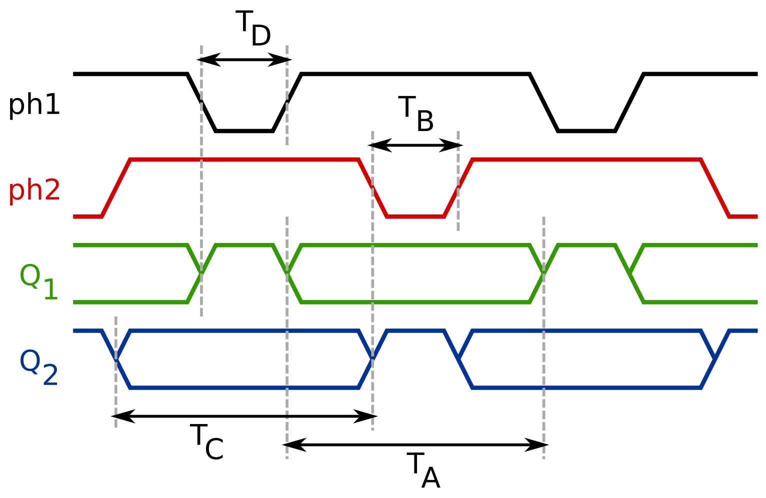


Figure 5.4: The two-phase synchronous SAPTL operation.

$$T_{SAPTL} = T_1 + T_2 \approxeq T_3 + T_{stack} + T_5 + T_6 \tag{5.3}$$

5.1.1 Two-Phase Clocking

Fig. 5.3 shows how two SAPTL blocks can be cascaded a two-phase clocking scheme shown in Fig. 5.4. The clocks $ph1$ and $ph2$ comprise the two components of a two-phase non-overlapping clock scheme that defines the precharge and evaluation phases of cascaded SAPTL stages. The inputs of each SAPTL stage, whose EN input is connected to $ph1$, is driven by a stage whose EN input is connected to $ph2$, and vice-versa.

5.1.1.1 Logic Evaluation

After the rising edge of $ph1$, the sense amplifier of the SAPTL stage X_1 makes a decision and valid data is presented at its outputs (Q_1). This data is valid until the next falling edge of $ph1$. Within this period when Q_1 is valid, $ph2$ makes a transition from high to low, both

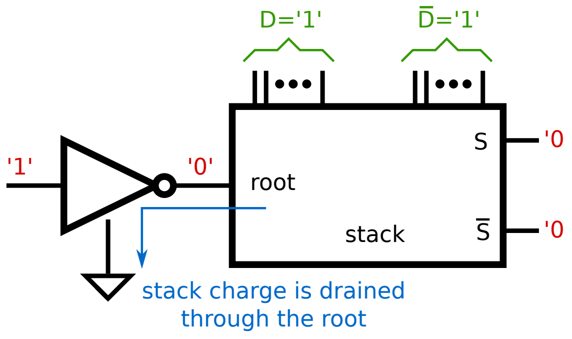


Figure 5.5: Stack initialization.

energizing the stack of the SAPTL stage X_2 and precharging the outputs Q_2. The sense amplifier of stage X_2 then samples its inputs at the rising edge of $ph2$, and after a decision is made, correct data is placed at the outputs Q_2. The data on Q_2 remains valid until the next falling edge of $ph2$. During this period when Q_2 is valid, $ph1$ goes from high to low, where stack evaluation occurs, and low to high, where the sense amplifier evaluation happens.

In this two-phase clocking scheme, the inputs to the stack remain valid during both the stack evaluation phase and sense amplifier evaluation phase.

5.1.1.2 Stack Initialization

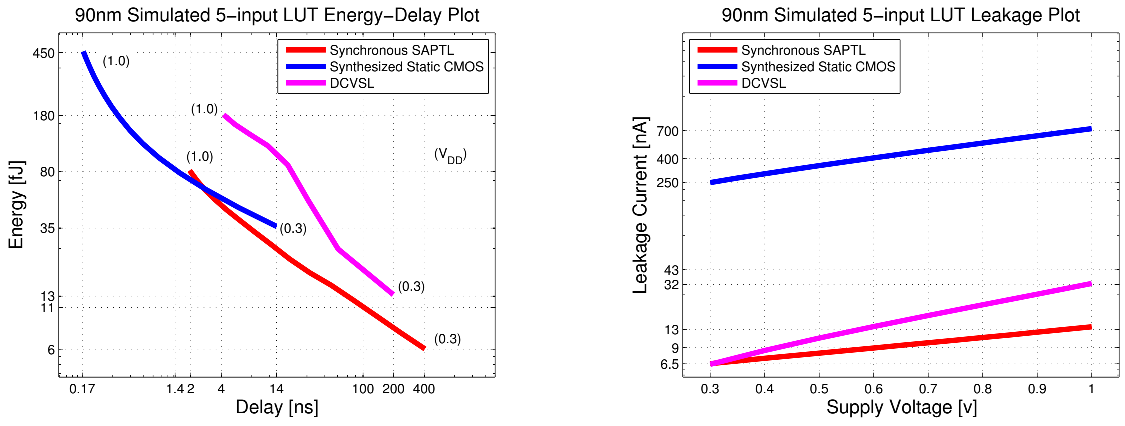
One advantage of precharging the sense amplifier outputs is it allows the initialization of a SAPTL stack by the stage(s) driving it. Since the path energized in the stack is data dependent, the high-impedance nodes present in the unused paths can accumulate charge due to leakage currents flowing to the unused paths from the root node or from the energized path. These leakage currents do not flow to ground, but instead charge up internal stack nodes or the unenergized stack output.

Consider the low phase of clock $ph1$ in Fig. 5.4. During this time, the outputs of SAPTL stage X_1 in Fig. 5.3 are both high, at the same time the root drive of SAPTL stage X_2 is low, as shown in Fig. 5.5. In this configuration, all the stack paths of X_2 are turned on, creating a low impedance path to the root node, where all the excess charges are drained. This initialization forces each stack evaluation to start at a known state, eliminating unwanted interactions between successive SAPTL computations. Similar to the low phase of $ph1$, the low phase of $ph2$ initializes the stack of SAPTL stage X_1.

5.1.2 Synchronous SAPTL Design

The minimum two-phase synchronous SAPTL clock period, $T_{2\phi}$, can be expressed as

$$T_{2\phi} = T_{1,ph1} + T_{7,ph1} + T_{1,ph2} + T_{7,ph2} \tag{5.4}$$



(a) (b)

Figure 5.6: The synchronous SAPTL5 (a) energy-delay characteristics and (b) leakage current.

where $T_{1,ph1}$ ($T_{1,ph2}$) is the time required by the SAPTL stage X_1 (X_2) to develop the appropriate dV at the output of its stack, and $T_{7,ph1}$ ($T_{7,ph2}$) is the time needed by the sense amplifier to sample its inputs, make a decision, and apply the correct values at its outputs. Due to the stack initialization scheme outlined in Sec. 5.1.1.2, $ph1$ stacks are initialized during $ph2$ stack evaluation and vice-versa. The non-overlap time in between the low phases of $ph1$ and $ph2$ should be greater than $T_{7,ph1}$ and $T_{7,ph2}$ respectively.

The energy-delay characteristics of a 5-input SAPTL block (SAPTL5) using a full stack is shown in Fig. 5.6a, as well as the energy-delay characteristics of a 5-input look-up table (LUT) implemented in both synthesized standard-cell CMOS and full-custom DCVSL. The performance of the synchronous SAPTL5 logic block is much less than static CMOS due to the limited number of SAPTL gain stages, thus less current is available to drive both the internal node and load capacitances, which in this case is a fan-out of 5 SAPTL5 blocks. However, this reduction in gain elements leads to a lower energy consumption, as well as lower standby or leakage current, as seen in Fig. 5.6b. Thus, the net effect of the gain reduction in the SAPTL5 circuit is a 12X-30X reduction in performance for a 6X reduction in energy.

5.2 Asynchronous Operation

Using a synchronous two-phase clocking scheme to cascade SAPTL blocks is relatively straightforward, since all the timing information is generated globally. However, there are two major drawbacks in using synchronous design techniques. First, synchronous designs accommodates the worst-case delay scenarios in the choice of clock periods, even if this worst-case scenario occurs very infrequently. Thus, most of the time, the system can potentially operate faster than the chosen clock period.

Secondly, synchronous designs force a computation at every clock cycle even when no new data or computation is required by the system. This forced computation results in dynamic energy usage which is not really needed by the system. This extraneous computation can normally be avoided at the block level using clock-gating techniques [WPW00], but can present difficulties if finer-grained clock control is required.

An alternative to global timing control is to use asynchronous design techniques, where timing information is generated locally. The area, energy and delay costs of generating this timing information locally replaces the cost of global clock generation and distribution.

By generating timing signals locally, each SAPTL block can allocate the appropriate amount of time for a certain computation. These local timing signals are also available to adjacent SAPTL blocks as handshaking signals that indicate when a computation needs to be performed or when a computation has already been completed.

Two asynchronous SAPTL timing techniques are presented here: (1) the delay line-based SAPTL [LAPR09] and (2) the asynchronous-detect sense amplifier-based SAPTL. A comparison of both these asynchronous SAPTL timing schemes with the original synchronous SAPTL implementation then follows. An in-depth analysis of the asynchronous behavior of the SAPTL, including another asynchronous SAPTL scheme called the dual-rail SAPTL, is outside the scope of this work, but can be found in [LAPR09].

5.2.1 The Delay Line-based Asynchronous SAPTL

Fig. 5.7 shows the basic architecture of the delay line-based asynchronous SAPTL (DL-SAPTL). In order to carry out clockless operation, the DL-SAPTL makes use of a timing control path, in addition to the original SAPTL data path. The timing control path in conjunction with a few data path modifications, is responsible for generating local SAPTL timing information.

The timing control path is composed of two major blocks: (1) a delay line and (2) a Muller C-element [Sut89] [SEE96]. The delay line, shown in Fig. 5.8, mimics the worst-case delay of the stack and is energized using the same driver that drives the root of the stack. The $READY$ signal generated by the delay line indicates that the stack outputs have had sufficient time to develop the necessary voltage levels.

The output of the C-element drives the sense amplifier EN input, which acts as a local clock that replaces the global clock signals $ph1$ or $ph2$ in a two-phase synchronous environment. Consider a simple DL-SAPTL FIFO in Fig. 5.9 and its corresponding timing characteristic shown in Fig. 5.10.

When a REQ is received from the driving SAPTL stage, the C-element waits for the delay line's $READY$ signal, then asserts the EN signal. The sense amplifier in turn, samples the outputs of the stack and makes a decision. The C-element then waits for the acknowledge

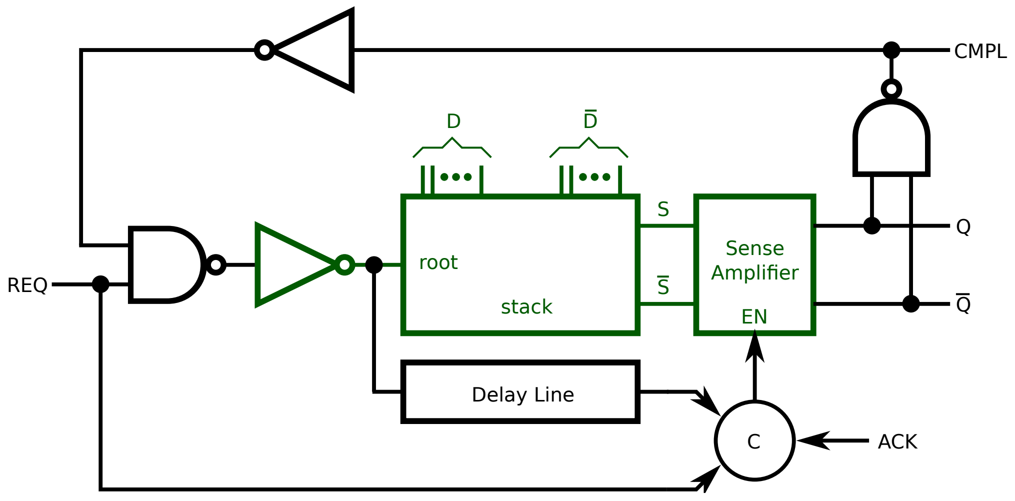


Figure 5.7: The delay line-based asynchronous SAPTL.

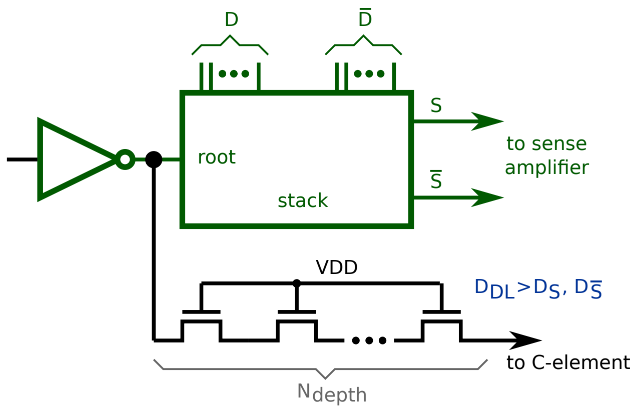


Figure 5.8: The delay line.

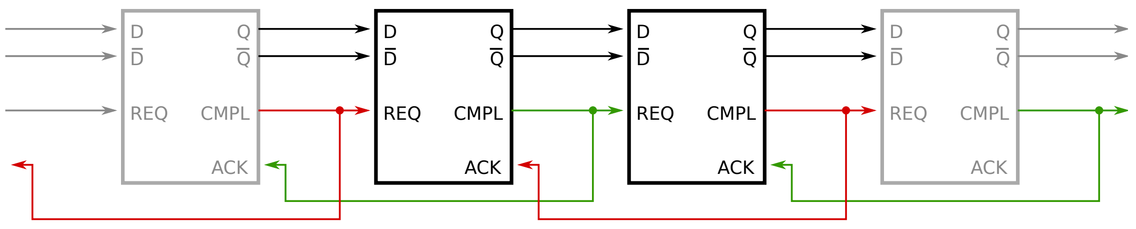


Figure 5.9: Simple DL-SAPTL pipeline.

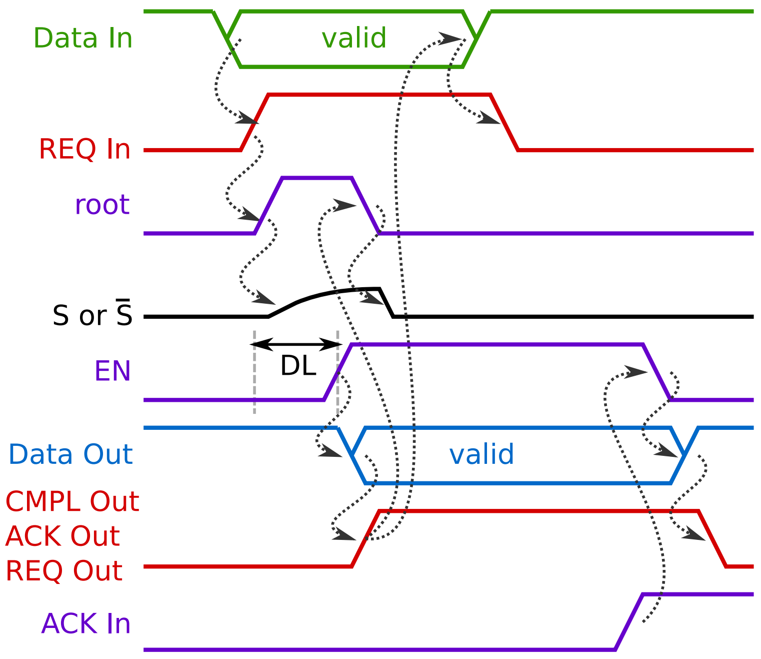


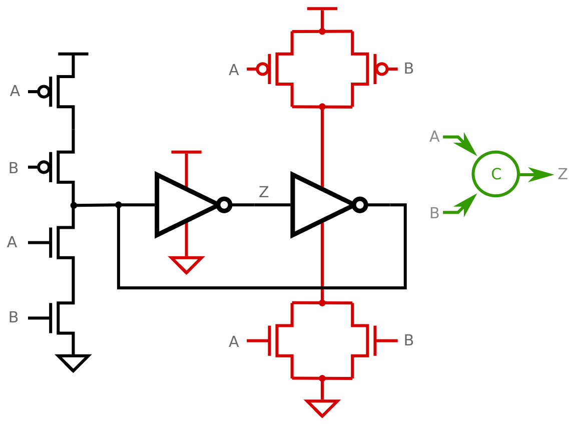
Figure 5.10: The DL-SAPTL timing diagram.

signal, ACK, generated by the next SAPTL stage, indicating that the current output is no longer needed. When the C-element receives the ACK signal, it deasserts the EN signal, thus precharging the sense amplifier outputs, then waits for the next REQ signal.

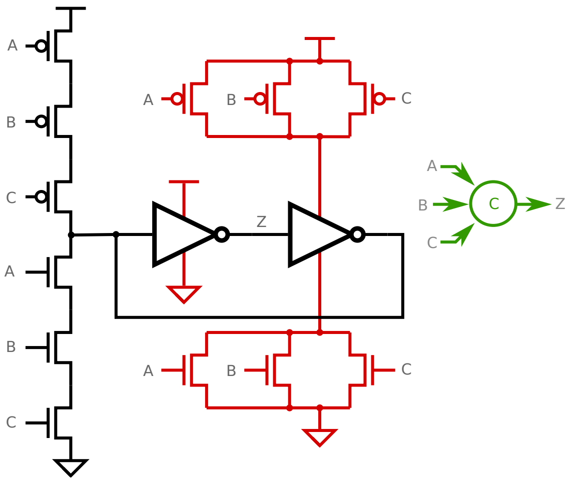
A completion signal, $CMPL$, is generated by the sense amplifier indicates that the operation is complete. Due to the differential signaling used, a SAPTL operation can be considered completed when the sense amplifier outputs are taken out of its precharge state and are given a valid differential boolean value. The $CMPL$ signal can then be generated using a simple NAND gate, and acts as both the REQ signal for the next SAPTL stage, and the ACK signal used by the previous SAPTL stage.

To accomplish the necessary timing generation tasks, the C-element needs a memory element in order to remember its current state. One implementation of a two-input Muller C-element is shown in Fig. 5.11a. The DL-SAPTL, however, requires a three-input C-element, shown in Fig. 5.11b, to accommodate both the REQ signal and the output of the delay line.

The completion signal $CMPL$, is used to turn off the NAND-INV pair at the root of the stack. This NAND gate disables the stack root driver as soon as the sense amplifier makes a decision. This ability to turn off the stack root drive as early as possible prevents the stack outputs from rising above the needed voltage levels, thus reducing the energy used to charge the stack output capacitances. The root NAND gate also prevents a new computation from starting while the results of the current computation is still needed.



(a)



(b)

Figure 5.11: A Muller C-element with (a) two inputs [Sut89] and (b) three inputs.

5.2.2 The Asynchronous SAPTL Operation without a Delay Line

The use of a delay line allows fine-grained timing control up to the individual SAPTL blocks. However, the delay line still has to accommodate the worst-case delay of a single block. To allow even finer-grained timing flexibility, accounting for data-dependent delay variations within each SAPTL block, the sense amplifier can be made to detect if there is enough dV at the output of the stack. This ability to detect signals at the output of the stack asynchronously eliminates the need for a delay line, at the expense of a more complex sense amplifier.

5.2.2.1 The Asynchronous-Detect Sense Amplifier

Fig. 5.12 shows the asynchronous-detect sense amplifier (ADSA), which uses a cross-coupled NOR gate instead of a cross-coupled inverter as the main latch. Using a cross-coupled NOR latch makes the sense amplifier precharge state a stable state, instead of an unstable state when using cross-coupled inverters. The stable ADSA precharge state (1) results in reduced crowbar currents during the transition from the precharge state to a stable data state, and more importantly, (2) allows the early assertion of the EN signal. Note that due to the leakage currents acting on nodes X and Y, this precharge state will eventually degrade and the ADSA will be forced to make a decision. The lifetime of the precharge state, however, is designed to be much longer than the maximum stack delay. Thus, for all practical purposes, this precharge state can be considered stable with respect to the rest of the SAPTL circuit.

Due to the unstable precharge state of the cross-coupled inverter, the original sense amplifier in Fig. 4.21 is forced to make a decision as soon as the EN signal is asserted. Thus, the time when EN is asserted must be precisely controlled.

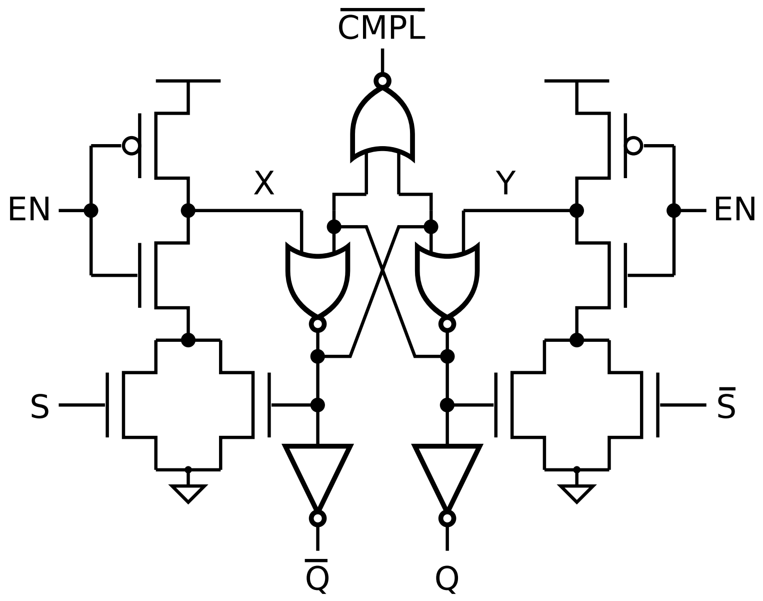


Figure 5.12: The asynchronous-detect sense amplifier.

If EN is raised too early, then the stack outputs S and $\overline{S}$ might not have had enough time to achieve the necessary differential voltage dV. Thus, the sense amplifier's decision will then be determined by the noise and mismatch present at its inputs. If EN is triggered too late, the off-path leakage current may have had enough time to charge up the wrong stack output, reducing dV. This reduced dV can also result in erroneous sense amplifier decisions.

The ADSA, on the other hand, has a stable precharge stage, thus allowing the EN signal to be triggered early. This early triggering on the EN line puts the ADSA into a wait state. In this state, the ADSA waits until a large enough signal is developed in either S or $\overline{S}$. When the voltage on S or $\overline{S}$ becomes large enough, the correct node, either X or Y, is pulled down low enough to change the state of the cross-coupled NOR latch. As soon as the cross-coupled NOR latch makes a decision, it will ignore any other changes in S and $\overline{S}$ unless it is precharged once again. This is made possible by the feedback transistors in parallel with the ADSA input transistors.

Note that precharging the cross-coupled NOR inputs pre-discharge the outputs. Thus, an odd number of inverters must be used to buffer the ADSA outputs as opposed to an even number of inverters required for the cross-coupled inverter-based sense amplifier. This pre-discharging of the cross-coupled NOR outputs also changes the way the completion detection is implemented. In this case, a $\overline{CMPL}$ signal is generated using a simple NOR gate.

The additional complexity added into the ADSA (1) reduces the timing precision needed to trigger the EN signal, and (2) allows the SAPTL stage to dynamically adapt to data-dependent stack delays.

Table 5.1: The SAPTL timing scheme (energy and delay) tradeoffs.

	$V_{DD} = 1$ V			$V_{DD} = 0.3$ V		
	Synch	DL	AD	Synch	DL	AD
Normalized Delay	1.0	0.64	0.49	1.0	0.27	0.11
Normalized Energy	1.0	1.09	1.18	1.0	1.88	1.31

5.2.2.2 The ADSA-based SAPTL Topology

The SAPTL block using the ADSA instead of the original SA and delay line is shown in Fig. 5.13a. The handshake logic of the ADSA-based SAPTL (AD-SAPTL) is also simplified into the cross-coupled NOR circuit, whose timing characteristics are seen in Fig. 5.13b.

This timing scheme is very similar to the DL-SAPTL timing scheme, except that since the ADSA can be triggered early, and the EN signal depends only on the REQ and ACK signals, a relatively simpler handshake circuit is needed. The cross-coupled NOR latch and inverter circuit drives the ADSA's EN input high when the ACK signal is deasserted and a REQ is triggered by the fan-in logic. The EN is then deasserted when an ACK is received from the fan-out blocks.

The root of the stack, again, is brought back to logic '0' as soon as the ADSA makes a decision, minimizing the amount of time the stack is energized, as well as limiting the voltage swings inside the stack. The DL-SAPTL pipeline scheme in Fig. 5.9 is also applicable to the AD-SAPTL since only the internal signals are affected by the ADSA modifications, and externally, the AD-SAPTL is functionally equivalent to the DL-SAPTL.

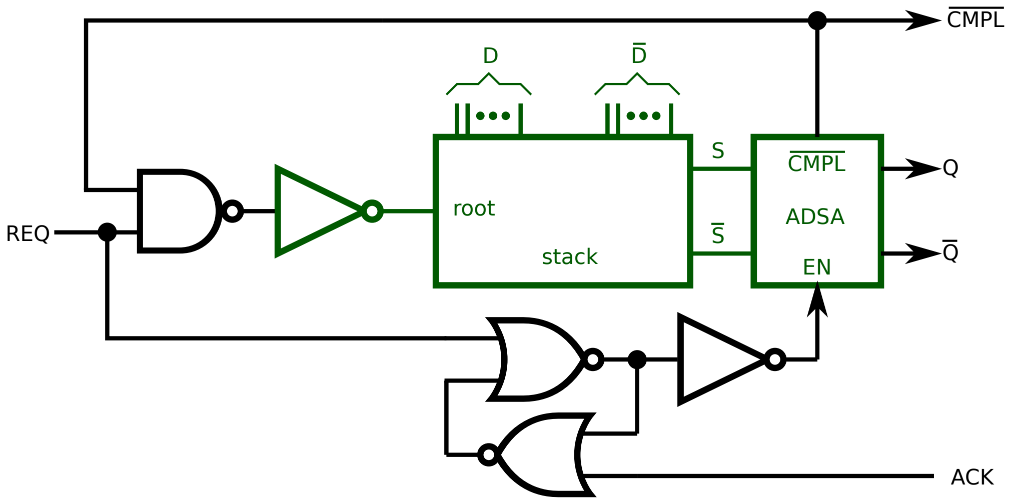
5.2.3 The Cost of Asynchrony

The simulated typical-case 90nm energy-delay characteristics of the three SAPTL timing schemes are shown in Fig. 5.14a. The two asynchronous SAPTL schemes, both with $\alpha = 0.125$, have better performance than the equivalent two-phase synchronous SAPTL, but due to the additional handshaking circuitry, the energy consumption of the DL- and AD-SAPTL are higher for the same supply voltage.

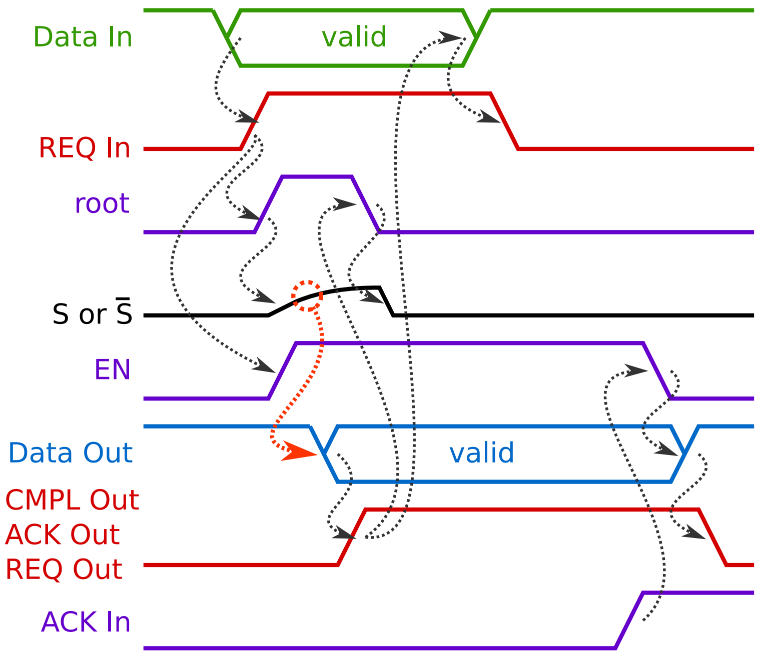
Both asynchronous SAPTL implementations have larger leakage currents due to the added leakage paths present in the handshaking circuitry, as can be seen in Fig. 5.14b. The AD-SAPTL has a slightly lower leakage current than the DL-SAPTL brought about by the removal of the delay line, and hence the reduction in the number of C-element inputs.

The energy and delay numbers of the two asynchronous SAPTL schemes normalized to the synchronous SAPTL scheme are tabulated in Table 5.1 for $V_{DD} = 1$ V and $V_{DD} = 0.3$ V. This table shows the energy cost of the additional handshaking circuitry and the corresponding of delay improvement.

At a supply voltage of 1V, the delay-line and handshake circuits of the DL-SAPTL accounts for the 10% increase in total energy, but allows a 35% reduction in delay. For $V_{DD} = 0.3$ V,

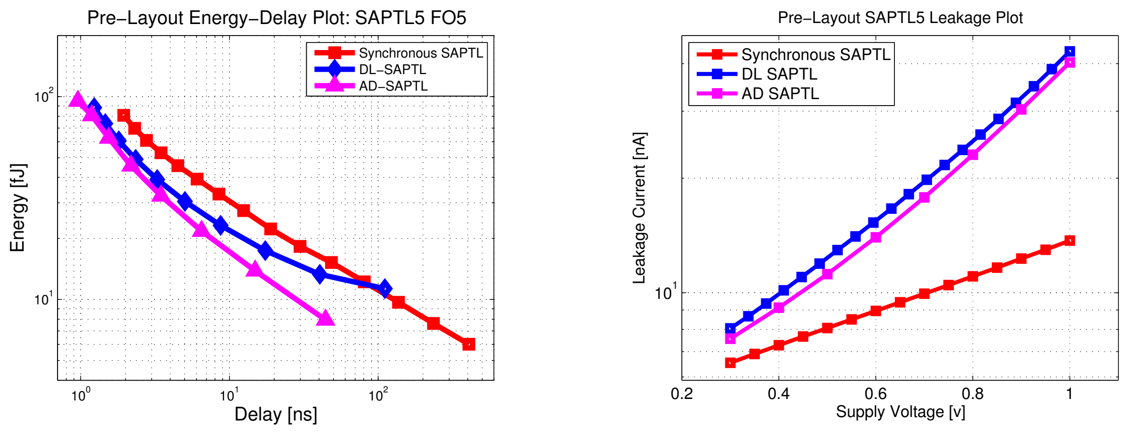


(a)



(b)

Figure 5.13: The ADSA-based SAPTL (a) architecture and (b) timing diagram.



(a) (b)

Figure 5.14: Comparison of the (a) energy-delay characteristics and (b) leakage currents of the AD-SAPTL, DL-SAPTL and the synchronous SAPTL as V_{DD} is increased from 0.3V to 1V.

the energy overhead increases to 88% due to the increased leakage energy component in the handshake circuits, while the delay drops by 73% due to the larger worst-case to average-case difference observed at lower supply voltages.

The AD-SAPTL on the other hand shows an 18% increase in energy at $V_{DD} = 1$ V, which is larger than the DL-SAPTL overhead. This larger energy is due to the more complex ADSA being exposed to the full supply voltage, as opposed to the DL, where most of its transistors are subject to voltages less than V_{DD}. At lower supply voltages where leakage energy dominates, the simpler AD-SAPTL handshake circuits have less leakage than those of the DL-SAPTL and thus only account for a 31% increase in energy relative to the synchronous SAPTL.

In addition, since the delay line needs to accommodate the local worst-case stack delay in the DL-SAPTL, the voltage swings inside the stack tend to be higher than the voltage swings inside the AD-SAPTL stacks. This increased internal voltage swings become more evident at lower supply voltages where the stack delays are larger, allowing the stack on-path and off-path currents more time to charge up the stack capacitances.

The advantage of the AD-SAPTL's finer-grained timing scheme is evident 51%-89% delay reduction when compared to the synchronous SAPTL. This delay improvement is better than the 35%-73% reduction achieved using the DL-SAPTL scheme.

The energy overhead of the handshake circuitry can be reduced by using circuit topologies other than fully complementary static CMOS logic. One possible alternative is to implement the C-elements and other timing circuits using subthreshold current-mode logic [TBLV08], thus reducing the overall energy at low supply voltages.

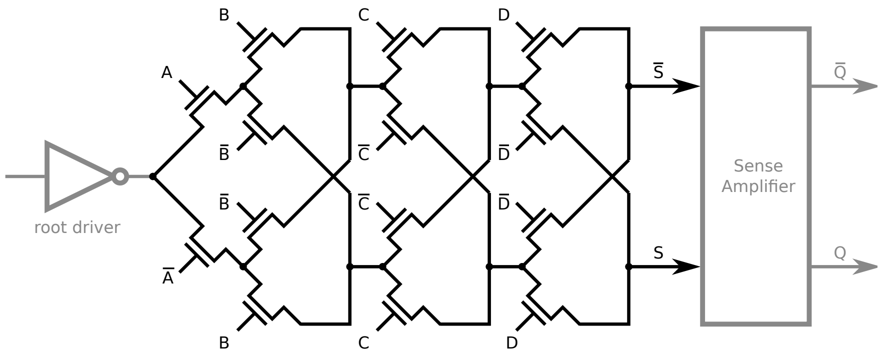


Figure 5.15: 4-input SAPTL XOR stack.

Table 5.2: The normalized energy and delay characteristics of the AD-SAPTL XOR gate.

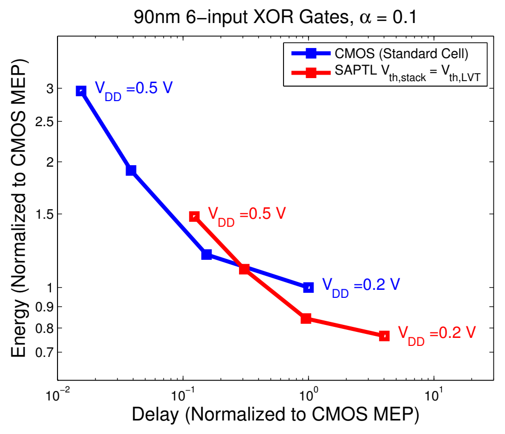
		CMOS		AD-SAPTL			
Inputs	V_{DD} [V]	Delay	Energy	Delay	Energy	$\uparrow D$	$\downarrow E$
6	0.5	0.015	2.950	0.120	1.480	8X	2X
	0.2	1.000	1.000	4.000	0.770	4X	1.3X
16	0.5	0.015	2.390	0.320	1.000	21X	2.4X
	0.2	1.000	1.000	7.100	0.560	7X	1.8X

5.3 An Example: The SAPTL XOR gate

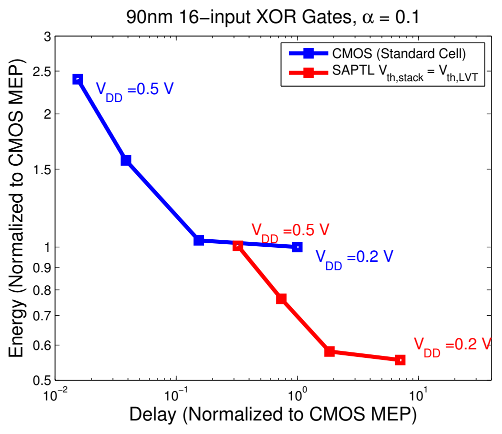
In order to understand how various logic functions are implemented, consider the pass-transistor stack that implements a 4-input XOR function as shown in Fig. 5.15. Each path from the root of the stack to S represents a minterm and each path from the root to $\overline{S}$ represents a maxterm. As can be seen from Sec. 4.6.1, and observed from Fig. 5.15 that the SAPTL implementation of XOR gates is very straightforward.

By increasing the complexity of the stack, in this case, increasing the number of inputs to the XOR gate, the sense amplifier and driver overhead per input can be reduced, at the expense of decreased performance. This can be seen in Fig. 5.16, where the energy and delay of a 6-input and 16-input AD-SAPTL XOR gate are compared to their static synthesized standard-cell CMOS equivalents. With the same V_{TH}, SAPTL reduces energy below the CMOS minimum energy point (MEP) due to the minimal gain elements used, resulting in a lower total leakage current.

For a supply voltage of 0.5 V, the 6-input SAPTL XOR gate consumes half of the energy as the equivalent CMOS gate, with a 10X increase in delay. For a V_{DD} of 0.2 V, the energy of the SAPTL is 60% less than that of the equivalent CMOS gate, at a 4X delay penalty.



(a)



(b)

Figure 5.16: AD-SAPTL XOR gate energy-delay characteristics for (a) 6-inputs and (b) 16-inputs.

5.4 Summary

The SAPTL performs computations in two steps: (1) the initialization phase, where all the internal nodes of the pass transistor stack are reset to ground, and (2) the evaluation phase, beginning when the correct path through the stack is energized, and ends after the sense amplifier makes a decision. Due to this two-phase operation, cascading SAPTL stages requires additional timing signals that provide the needed temporal separation of these two phases.

One way to provide these timing signal is through the generation of two non-overlapping clock signals. These clock signals group the SAPTL stages into two, a $ph1$ stage and a $ph2$ stage. In this synchronous timing scheme, a $ph1$ ($ph2$) stage can only be driven by a $ph2$ ($ph1$) stage, and can only drive a $ph2$ ($ph1$) stage. Though the evaluation times of the two phases do not overlap, the initialization phase of a $ph1$ stage occurs simultaneously with the evaluation of a $ph2$ stage, and vice-versa.

Synchronous SAPTL operation requires that the clock period take into account the worst-case delay present in the system. However, if timing information is generated asynchronously within each SAPTL block, the system performance can operate closer to the average system delay, instead of the worst-case. Two methods are presented: (1) the delay line-based scheme and (2) the ADSA-based topology. Both asynchronous schemes use additional handshaking circuits that are used to separate the initialization and evaluation phases within and among connected SAPTL stages.

By using the AD-SAPTL timing scheme, an 89% reduction in delay can be achieved with an energy increase of 31% at a supply voltage of 300mV, when compared to the synchronous SAPTL for a 5-input LUT. Minimizing the AD-SAPTL stack to implement a 16-input XOR gate results in a 2X reduction in energy with a 10X delay penalty.

Several SAPTL-based circuits using these timing schemes are presented in Chapter 6 The design and implementation details of these circuits are shown, highlighting their energy and delay characteristics in comparison to their respective traditional static CMOS implementations.

Chapter 6

Case Studies

This chapter presents the design and implementation of the three 90nm SAPTL test chips containing building blocks and circuits. These circuits serve as reference points, helping evaluate the feasibility of using the SAPTL topology as a low-energy alternative to static CMOS logic, in different timing environments and computation scenarios. Details of the design, implementation, and test methodologies are described, together with simulation and test results.

6.1 The Synchronous SAPTL Test Chip

In order to verify the functionality and simulation results of the SAPTL topology and its synchronous timing scheme, a 90nm test chip was designed and fabricated. The test chip contained experiments that characterized the behavior of the synchronous SAPTL blocks as the fanout, stack depth, logical functionality, and supply voltages are varied.

In order to characterize the behavior of the SAPTL blocks, two main circuits are used: (a) ring oscillators using a stack and driver as the main gain and delay stage in Fig. 6.1a, used to measure stack performance, and (b) two back-to-back SAPTL blocks in Fig. 6.1b, which is used to measure functionality and energy.

6.1.1 Basic Components

The pass transistor stack is the basic building block for all SAPTL circuits, and in this chip, only full stacks are used. Each full stack is laid out using minimum sized transistors, arranged as an array of simple stacks, where each simple stack represents a minterm, as described in Sec. 4.7 and shown in Fig. 6.2a. Thus, for any stack depth, N_{depth}, $N_{depth} \times 2^{N_{depth}}$ minimum-sized transistors are used. The stack in Fig. 6.2a is still equivalent to the stack in Fig. 4.2, with each non-minimum-sized transistor broken up into its minimum-sized components. Fig.

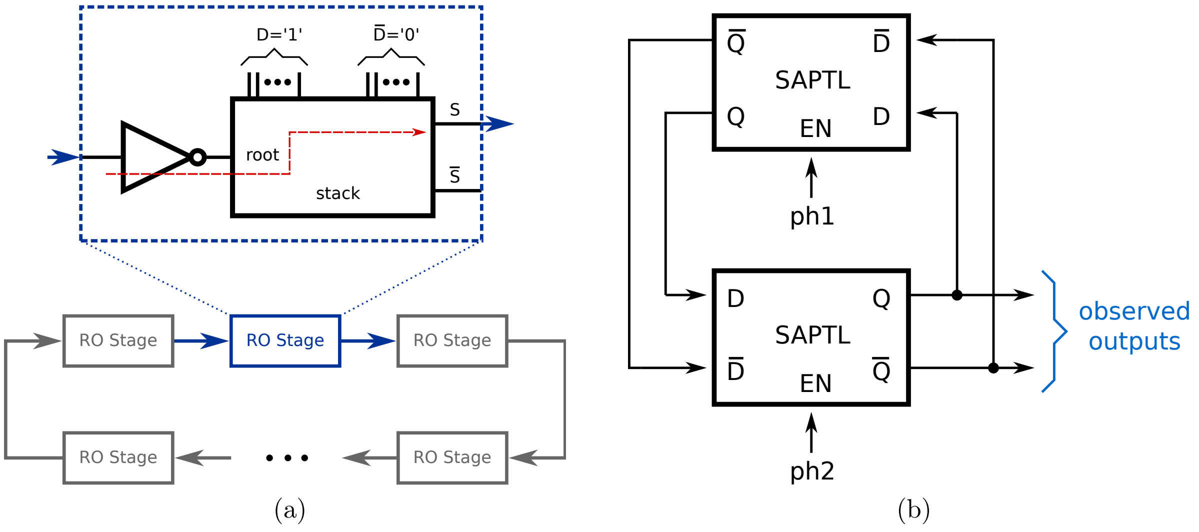


Figure 6.1: The synchronous SAPTL test blocks: (a) the ring oscillator without a sense amplifier and (b) the back-to-back SAPTL blocks.

Table 6.1: 90nm pass transistor stack layout area comparison.

N_{depth}	Width (μm)	Height (μm)	Area (μm^2)
3	8.94	8.02	71.70
5	17.92	10.98	196.76
7	40.36	17.30	698.43

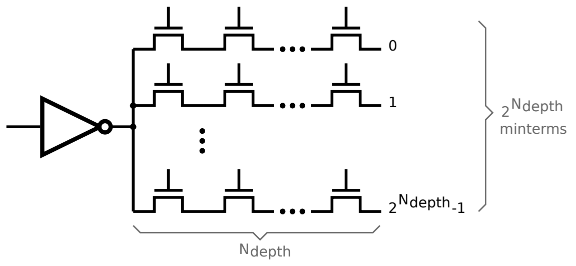
6.2b shows the relative layout sizes of three pass transistor stacks, with stack depths of 3, 5 and 7. These layout sizes are summarized in Table 6.1. These area numbers include the triple-well-surround overhead, and thus scales less than the expected 2X for every increase in stack depth.

The complete full-custom SAPTL logic block layout with $N_{depth} = 5$, is shown in Fig. 6.3. The logic block includes the pass transistor stack and the sense amplifier. The size of the sense amplifier layout is 115 μm^2, which is approximately 37% of the total synchronous SAPTL5 area.

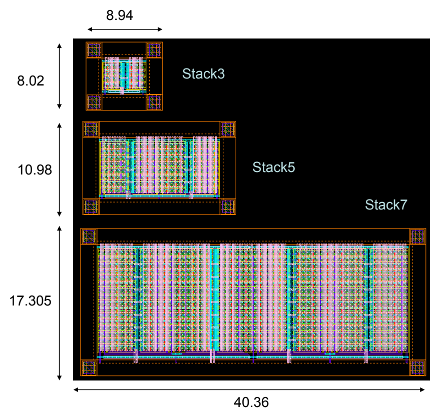
6.1.2 Chip Organization

The overall chip test is composed of (1) a global data and address bus, (2) a power supply distribution network and (3) a global two-phase clock distribution network. Each device-under-test (DUT) tile is connected to this global bus using an independent, separately addressable local test interface, as shown in Fig. 6.4.

The contents of each test tile, shown in Fig. 6.5, consists of (1) a standard-cell CMOS test interface, (2) two power switches, (3) a CMOS-to-SAPTL converter block and (4) the device



(a)



(b)

Figure 6.2: The pass transistor stack implementation showing (a) the stack layout strategy and (b) the relative layout sizes of the 90nm SAPTL stacks.

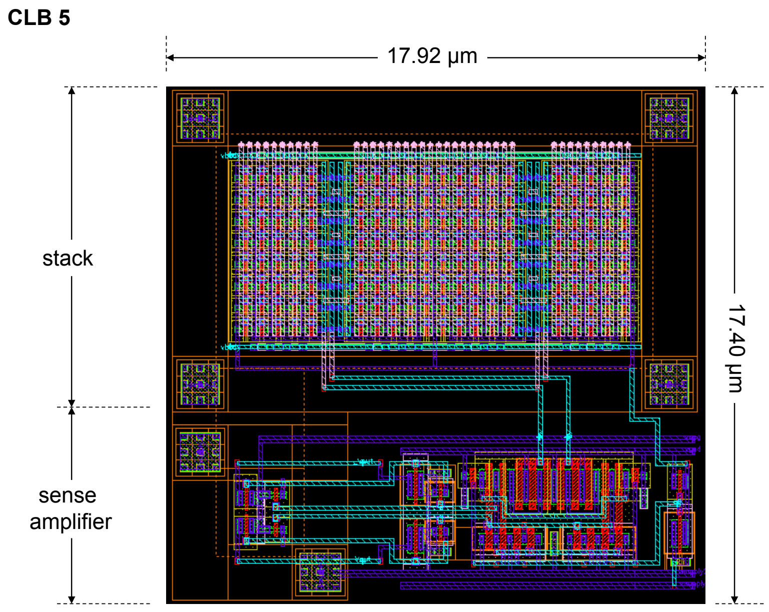


Figure 6.3: The synchronous 90nm SAPTL5 layout.

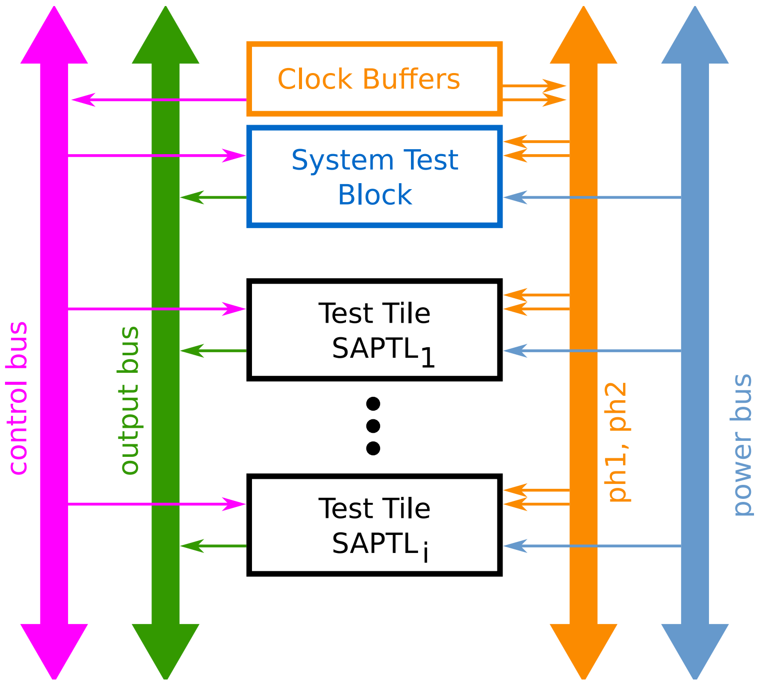


Figure 6.4: Synchronous SAPTL test chip architecture.

under test, which is either a stack and driver ring oscillator or a back-to-back SAPTL block.

The test interface (TI) is connected to the global data bus. This data bus provides a clock and reset signal, as well as address data to all test interfaces. Each TI has a unique hardwired address. Once selected, the TI turns on its the power switch, energizing the DUT connected to it and also initializes the DUT to a known state, as in the case of the back-to-back SAPTL blocks. The data from the DUT, in the form of (1) an oscillator output or (2) the SAPTL Q and $\overline{Q}$ outputs, are then divided down using the TI's fixed divide-by-32 divider, then sent out on the global bus, which are then monitored at the chip's output pins.

A special test interface block, called the system test block, is used to evaluate the global clock and bus lines, making sure that the maximum frequency these clock and bus lines can support are larger than the expected and measured test outputs. The system test block also measures the quiescent leakage current of the power supply lines when no devices are connected. This leakage current data is then subtracted from the measured DUT currents to eliminate the effects of parasitic leakage currents from the disabled power switches of the unused blocks, as well as leakage from the I/O pad ring.

6.1.3 Results

Fig. 6.6a shows the fabricated 90nm test chip measuring 1.2 × 1.2 mm^2 with a 1V I/O pad ring. The test chip was packaged in the Kyocera QC-064347-WZ 64-pin ceramic lead-less

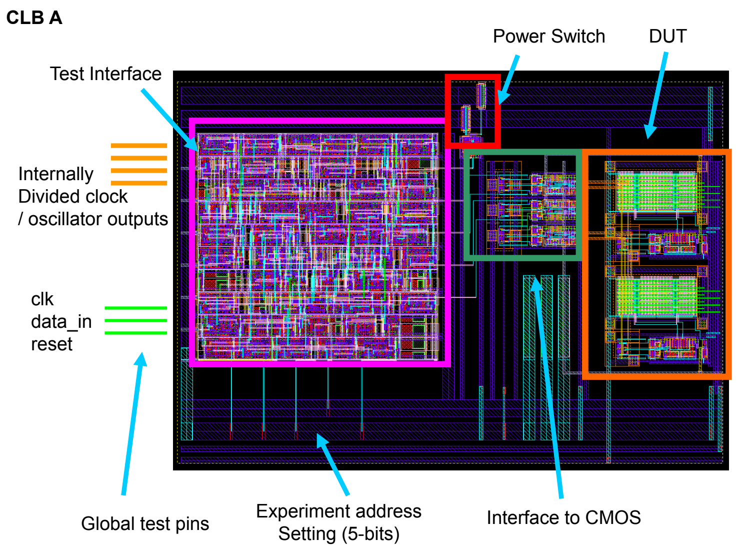


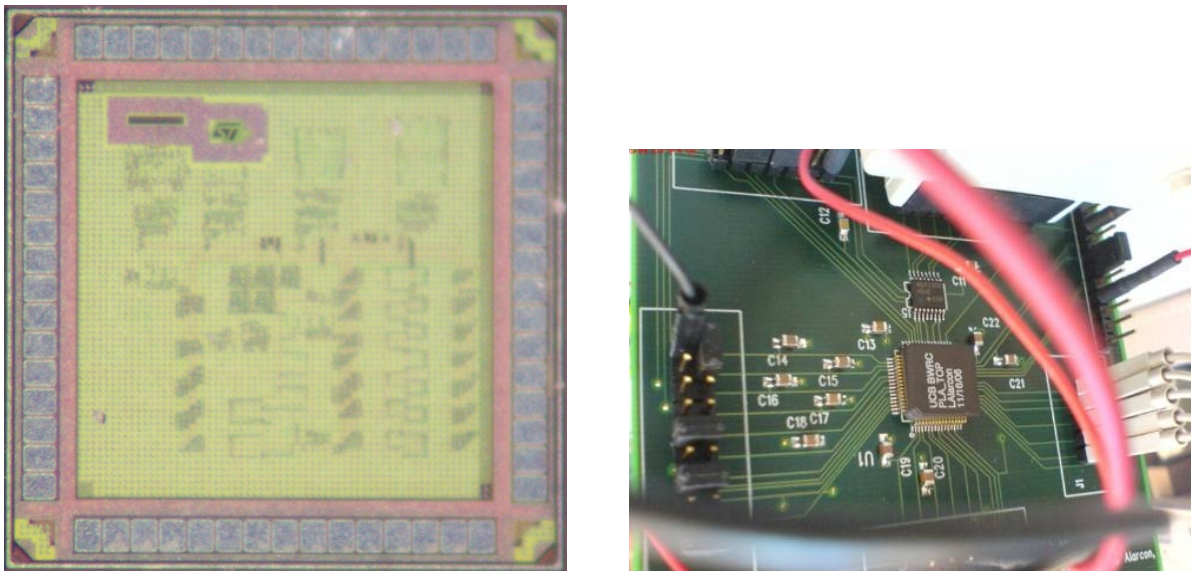
Figure 6.5: Synchronous SAPTL test tile organization.

chip carrier and mounted on a circuit board as shown in Fig. 6.6b. All supply voltages and reset signals are applied externally, as well as the test clock used by the test interface and the two-phase non-overlapping clocks needed by the SAPTL blocks. The addresses used to select the target DUT are applied serially and is processed by each test interface. The test and supply voltages are generated, and their respective currents are measured by Keithley source meters and the HP/Agilent 6626A multiple-output power supply, both devices configured to use four-wire force-sense modes for more accurate current measurements. Data patterns are generated and outputs are observed using the Agilent 16702A logic analysis system.

Determining the delays associated with the stack is one of the primary objectives of this test chip and is obtained using the stack-and-driver ring oscillators. Fig. 6.7 shows the measured stack and driver delays obtained from these ring oscillators, found in Fig. 6.1a, for different stack depths. As can be seen from the measurements, the test chips fall in a slightly faster corner than the typical (TT) case. The 4X average increase in delays from a full stack depth of 3 to 5 and from 5 to 7 reflects the 4X increase in number of transistors, and thus, the effective increase in total switched stack capacitance.

The energy characteristics of the SAPTL are derived from the currents of the back-to-back SAPTL blocks. The energy measurements are summarized in Fig. 6.8. These measurements are averaged over 100 samples for only a single die and taken at different clock frequencies, whose maximum is limited by the on-board clock buffers and level-converters.

Fig. 6.8a shows the good match between the simulated energy and measurement results. Increasing the stack depth, as expected, increases the energy due to the exponential increase in the full-stack SAPTL input capacitance. However, instead of a 4X increase in energy due to the 4X increase in effective switched capacitance, the observed increase in energy is



(a) (b)

Figure 6.6: The synchronous SAPTL (a) test chip and (b) packaged chip and circuit board.

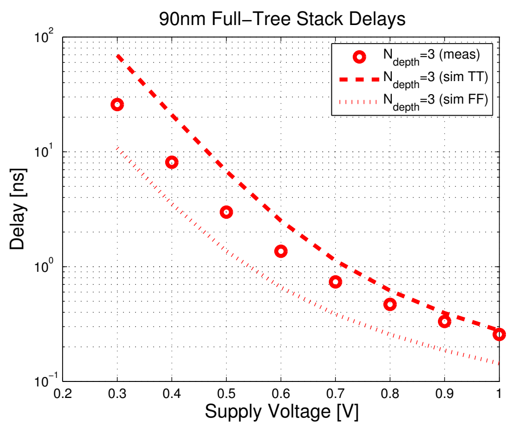
approximately 2.7X from a stack depth of 3 to 5 and from 5 to 7, as seen in Fig. 6.8b. This is a result of the lower voltage swings in the internal nodes of the deeper stacks.

Increasing the SAPTL5 fanout from 1 to 3 and from 3 to 5 increases the energy by approximately 28%, as shown in Fig. 6.8c, from which the SA effective capacitance of 65 fF and the unit stack differential input capacitance of 25 fF can be inferred. The energy of two different stack topologies in Fig. 6.8d shows the AND/NAND and XOR/XNOR stacks exhibiting little difference on the average. This behavior is expected since for a full stack, the AND/NAND configuration exercises both the best case and worst case paths in terms of the on-path to off-path current ratio, while the XOR/XNOR stack exercises the average case path.

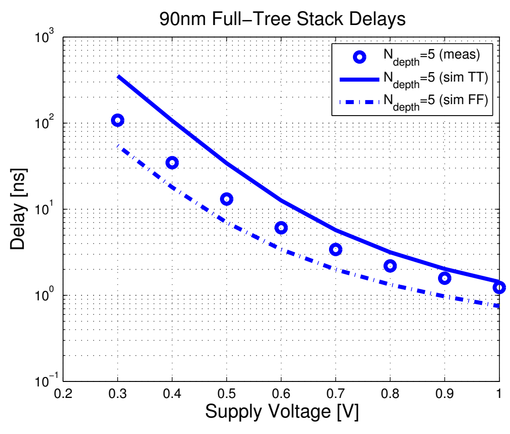
Due to limitations in the design of the test chip and circuit board, two parameters were not directly measured: (1) The maximum operating frequencies of the back-to-back SAPTL blocks, limited by the level converters used to map the 1.8V test signals to the 1V test chip pad frame voltage. However, these frequencies can be inferred using the stack ring oscillator data. (2) The static standby leakage current of the SAPTL blocks. However, these two parameters are successfully measured in the succeeding SAPTL test chips.

6.1.4 Summary

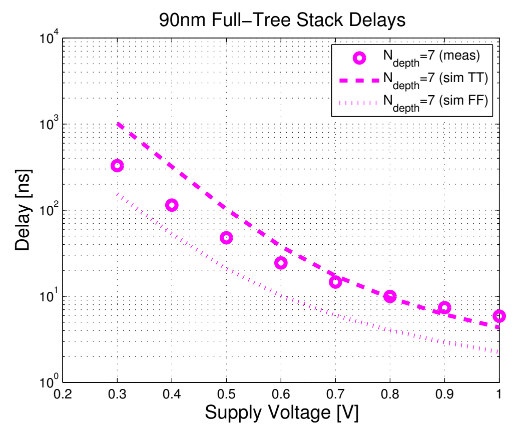
A 90nm test chip was created, with the objective of (1) verifying the functionality and behavior of the basic SAPTL topology and its components and (2) verifying the correctness of the ST90 simulation models used. This test chip also bridges the gap between the basic



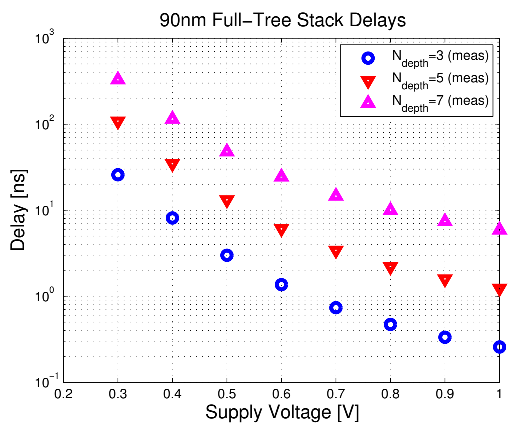
(a)



(b)

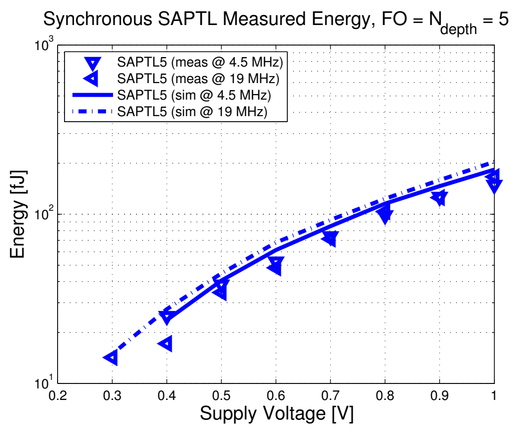


(c)

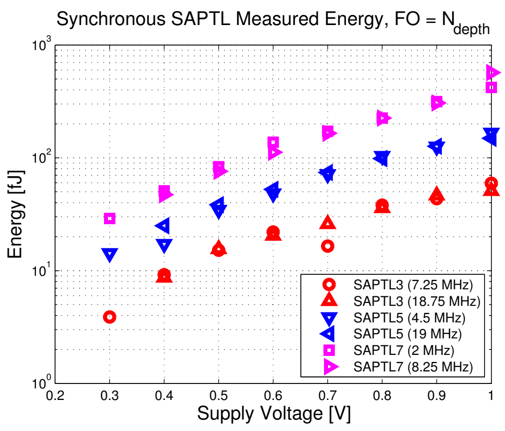


(d)

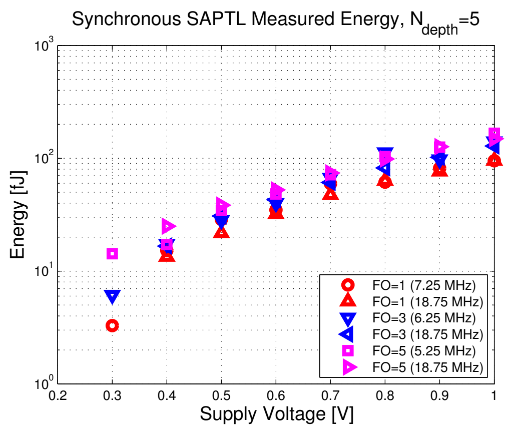
Figure 6.7: Simulated vs. measured stack delays for (a) $N_{depth} = 3$, (b) $N_{depth} = 5$, (c) $N_{depth} = 7$. (d) shows the measured stack delays vs. stack depth.



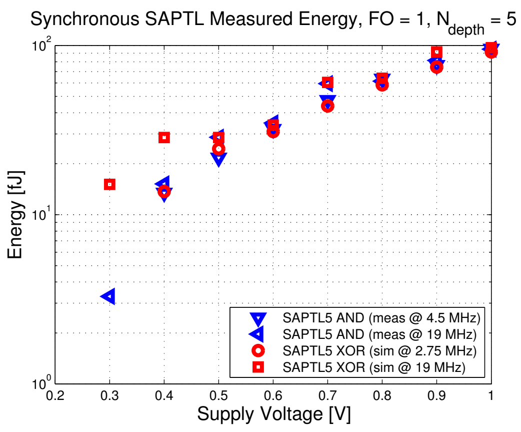
(a)



(b)



(c)



(d)

Figure 6.8: Measured energy vs. V_{DD} of (a) a synchronous SAPTL5 logic block, (b) measured energy vs. $V_{DD}$with varying stack depth, (c) fanout and (d) logic functionality.

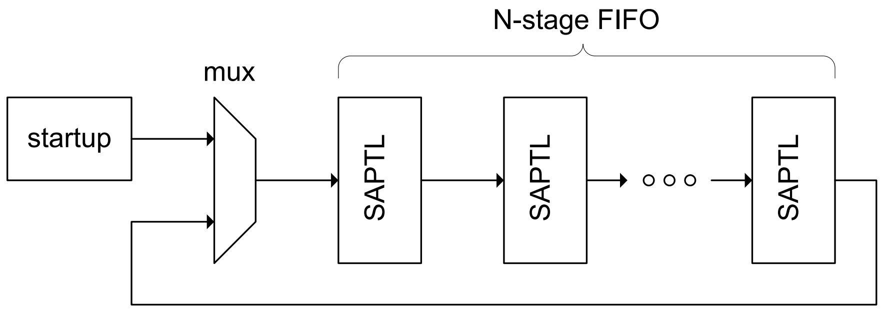


Figure 6.9: Test setup for energy and delay measurements. The energy and delay of the various SAPTL5 implementations were measured using $N = 8$.

SAPTL ideas at the schematic-level and the actual physical circuit. Physical design issues such as (1) full-custom strategies that increase layout regularity, (2) sense amplifier layout and (3) block placement and interconnect routing are explored and implemented.

6.2 The Asynchronous SAPTL Test Chip

The asynchronous SAPTL timing scheme eliminates the need for global clock signals and instead, local handshaking signals are generated in each block. The characteristics of this asynchronous timing scheme is extracted from a 90nm asynchronous SAPTL test chip. Two versions of the asynchronous SAPTL is implemented: (1) the delay-line-based SAPTL, also called the bundled-data SAPTL and (2) hybrid SAPTL or the dual-rail SAPTL [LAPR09]. Only the delay-line-based SAPTL is considered here.

The asynchronous SAPTL test chip contains separate test structures for characterizing (1) active energy and delay, and (2) leakage currents. The energy and delay characterization was done using 8 stages of SAPTL blocks with $N_{stack} = 5$ (SAPTL5) in a ring first-in-first-out (FIFO) structure as shown in Fig. 6.9. Upon reset, a token is inserted into the ring by the startup circuitry and is allowed to circulate within the ring. The FIFO supply current and oscillation frequency were then measured as the supply voltage is varied from $300mV$ to $1V$.

Leakage current measurements were taken from 100 replicas of each SAPTL circuit in its data holding stage, which would result in the worst-case leakage current. These leakage test circuits are connected directly to their own power supply pads and can be tested by measuring the static current at these pads directly and subtracting the pad leakage data.

6.2.1 Design Methodology

In order to reduce the amount of custom layout that is needed, a semi-custom SAPTL design flow was created, as shown in Fig. 6.10. Each SAPTL building block, containing

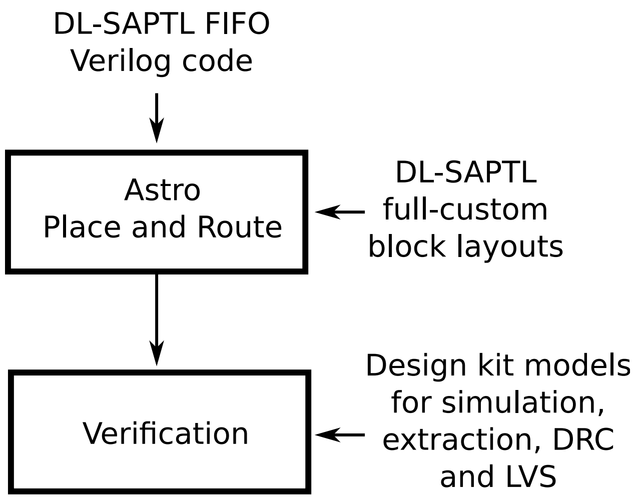


Figure 6.10: The asynchronous SAPTL tool flow.

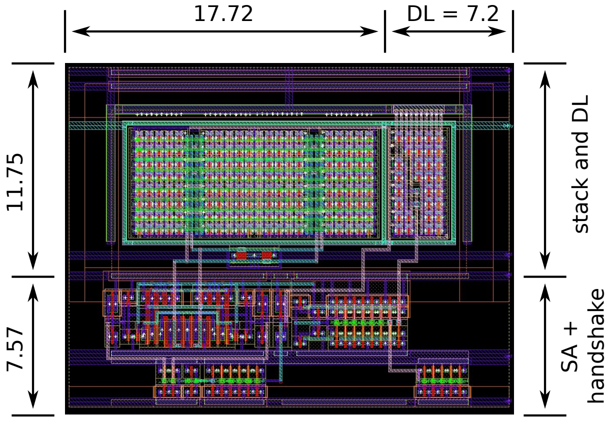
the full stack, sense amplifier and handshaking circuitry for different stack depths, is custom designed. The layout of two asynchronous SAPTL blocks with different stack depths are shown in Fig. 6.11, showing the full stack, the delay line, the sense amplifier and the handshaking circuitry. Note that the delay line increases the effective stack area by 20%-30%. Layout abstracts are then generated for these SAPTL blocks, and placed into libraries that the Astro place-and-route tool can access and utilize.

The FIFO design is then described using Verilog HDL, instantiating the needed SAPTL blocks and specifying the interconnection between these blocks. This Verilog code is then read by the Astro P&R tool, and is then used as the basis for automatic placement of the SAPTL blocks and routing of power, data and handshaking signals. Fig. 6.12 shows four different FIFO circuits that are automatically placed and routed by the Astro P&R tool.

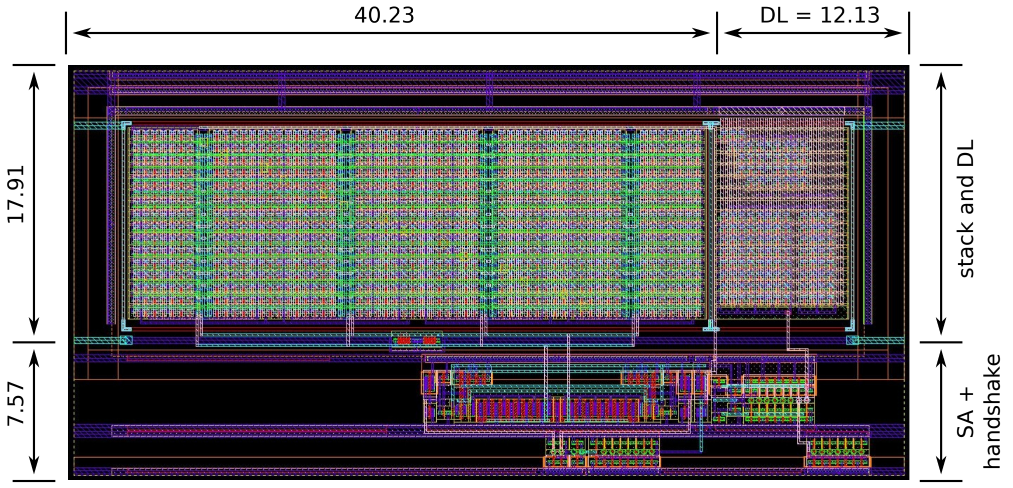
6.2.2 Test Chip Organization

The automatically generated layouts of the FIFO blocks are then connected to its own test interface, shown in Fig. 6.13, which is in turn, connected to the global data and address bus. This modular approach to system testing is similar to the one utilized by the synchronous SAPTL test chip.

The test interface (TI) contains an address decoder that determines if the SAPTL FIFO block connected to it is the one being selected for evaluation. If a block is selected, the TI enables the FIFO's power switch, energizing the circuit. At the same time, the startup circuit injects a token into the FIFO through a FIFO interface that provides the necessary handshaking signals. The startup circuit waits until the token has been consumed by the first stage of the FIFO, then disconnects itself from the ring. Thus, the single token is made to circulate within the FIFO indefinitely.



(a)



(b)

Figure 6.11: The asynchronous (a) SAPTL5 and (b) SAPTL7 layouts.

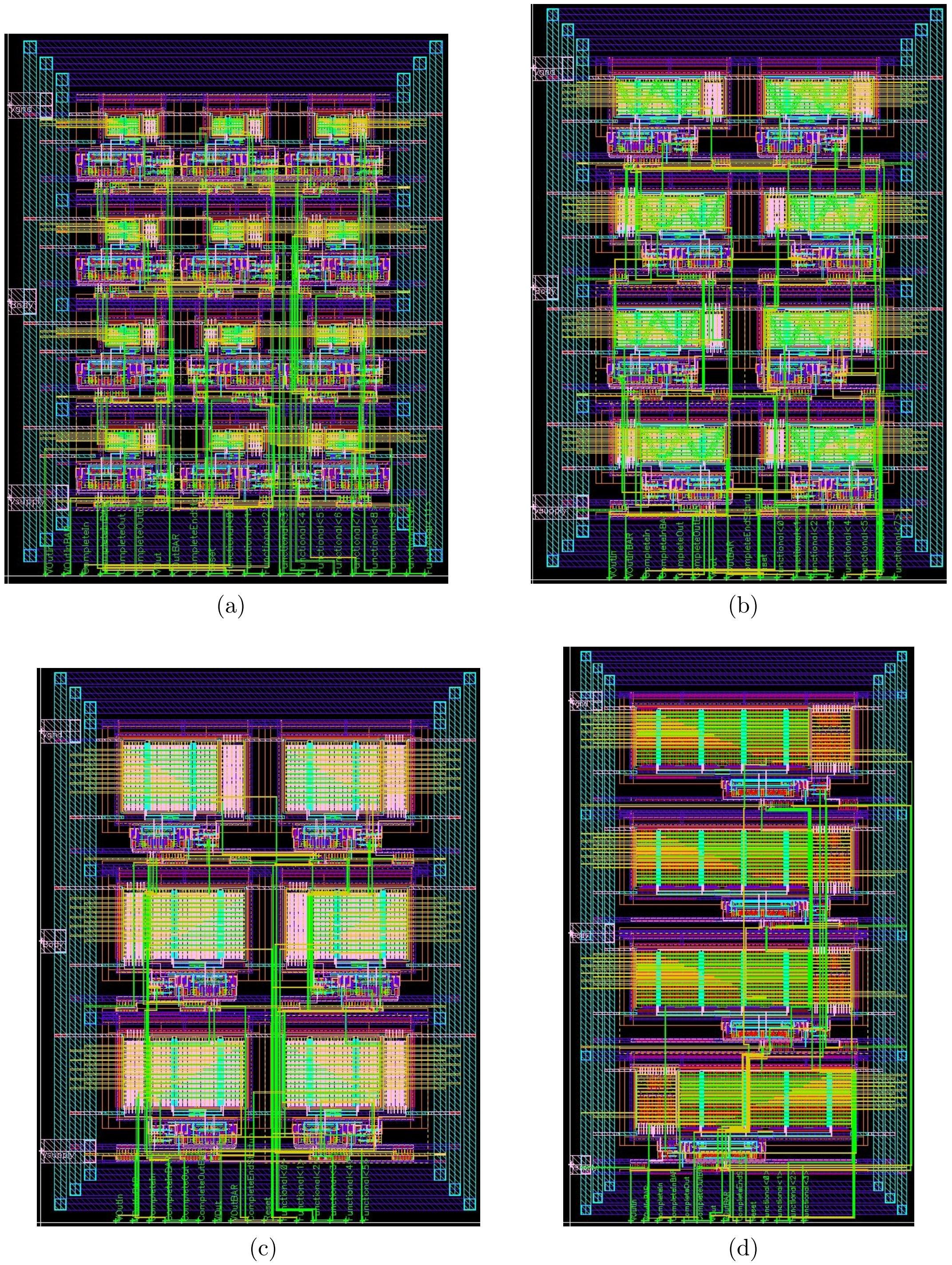


Figure 6.12: The automatically generated asynchronous FIFO circuits using (a) SAPTL3, (b) SAPTL5, (c) SAPTL6 and (d) SAPTL7.

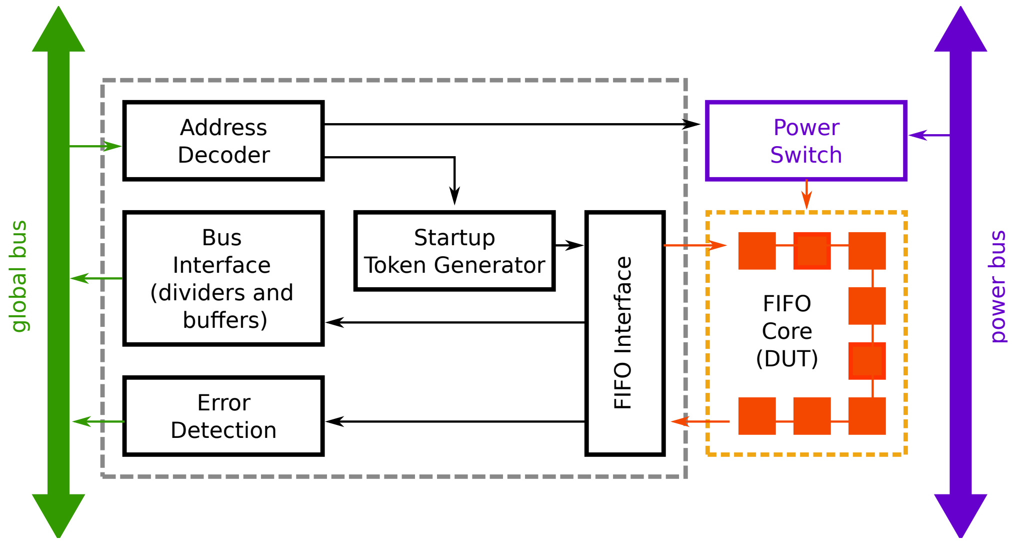


Figure 6.13: The asynchronous SAPTL test tile.

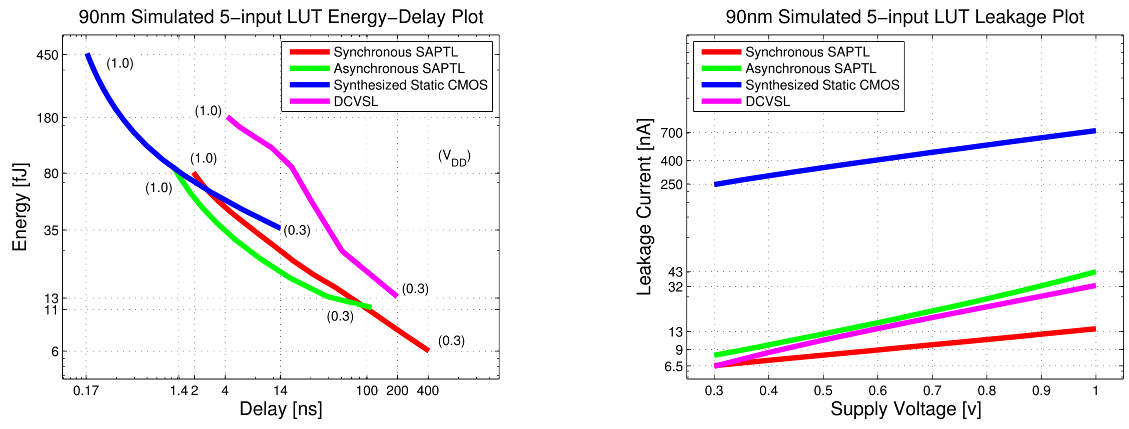
Throughput data is obtained at the chip outputs, through the global data bus, by dividing down by 32 and buffering the completion signal of the first SAPTL block. An error detection block monitors the SAPTL blocks within the FIFO, and raises an error flag if the token disappears of if more than one token in the FIFO is detected.

6.2.3 Results

Fig. 6.14a shows the simulated energy-delay characteristics of the DL-SAPTL5 as compared to the synchronous SAPTL5, as well as the equivalent static CMOS and DCVSL gates. As can be seen from this figure, the DL-SAPTL achieves a 3.8X-5.6X reduction in energy with a corresponding 7.14X-8.2X increase in delay when compared to the equivalent static CMOS implementation. At the low-V_{DD} region, both active and standby energy due to the handshaking circuits start to dominate, limiting the minimum energy that the DL-SAPTL can achieve. The leakage current of the DL-SAPTL5, shown in Fig. 6.14b, is 1.4X to 3.3X higher than the synchronous SAPTL5, again due to the additional handshaking circuitry present. This larger DL-SAPTL leakage results in a 1.8X increase in energy at $V_{DD} = 0.3$ V, but allows a 4X reduction in delay. Note that since the synchronous SAPTL leakage does not include any clock generation or distribution energy, the comparison is somewhat skewed in its favor.

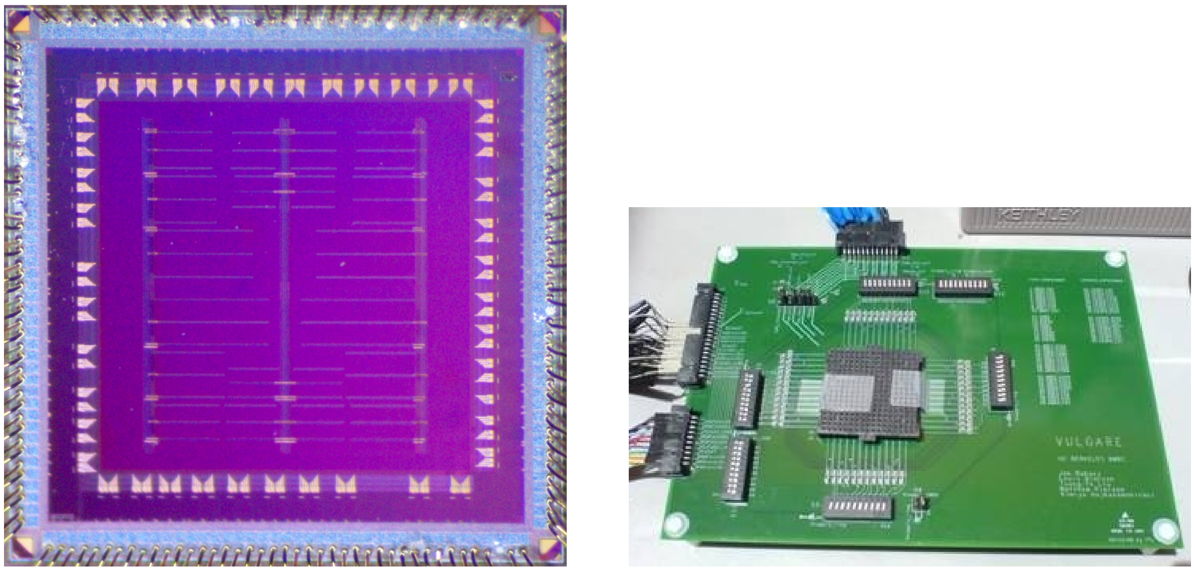
The photograph of the implemented 90nm 3 $\times$ 3 mm^2 asynchronous SAPTL test chip is shown in Fig. 6.15a. Though the chip area is large, only a small portion is dedicated to the DL-SAPTL. The test chip is wire-bonded to a SSM/NTK CPG18027 180-pin ceramic pin-grid array (PGA) package, and tested using the circuit board in Fig. 6.15b.

Energy-delay and leakage current measurements for the DL-SAPTL5, using SVT stack tran-



(a) (b)

Figure 6.14: The asynchronous SAPTL5 (a) energy-delay characteristics and (b) leakage current.



(a) (b)

Figure 6.15: The (a) asynchronous SAPTL test chip, also known as the Vulgare test chip and (b) the Vulgare circuit board.

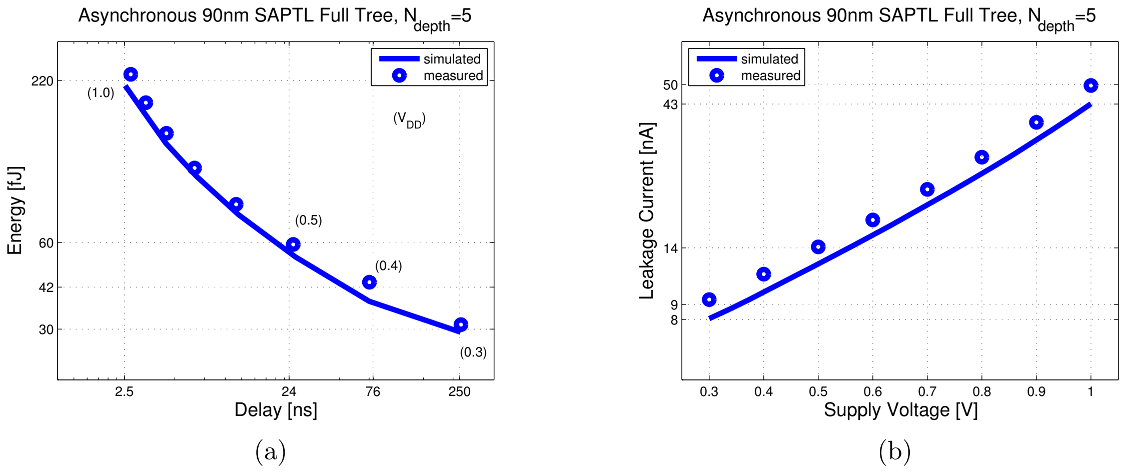


Figure 6.16: The DL-SAPTL5 SVT stack measured (a) energy-delay characteristics and (b) leakage current.

sistors, are shown in Fig. 6.16. These are measurement results from a single die, and averaged over 100 consecutive measurements and match very well with the simulation results after the effects of the pad parasitics have been taken out. Using low-threshold voltage stack transistors improves the delay with almost no change in energy, as seen in Fig. 6.17, with the improvement being as high as 40% at low supply voltages.

6.2.4 Summary

The asynchronous SAPTL test chip contains DL-SAPTL-based FIFO test structures intended to characterize the handshake timing sequence, and the energy costs associated with it. Two versions of each DL-SAPTL FIFOs are implemented, one with standard-V_{TH} stacks and the other with low-V_{TH} stacks. In order to reduce design time, as well as leverage well-established design verification methodologies, a simple semi-custom place-and-route design flow was created. This design flow allowed HDL descriptions of the FIFO blocks to be fed directly into the Astro P&R tool, resulting in reduced wire lengths and final block placements that guarantee this.

Similar to the synchronous SAPTL test chip, a global test strategy is employed, using self-contained and modular test tiles connected to a global bus. This strategy allows test structures to be placed into the chip independently of each other, and then verified to work with the global test system even if other blocks are not yet available. This divide-and-conquer strategy was very useful in distributing work among the different chip designers, as well as making sure that very few catastrophic errors can actually occur and safeguards are put in place to prevent them from occurring.

The ability to build-in some amount of automation in the design of asynchronous SAPTL

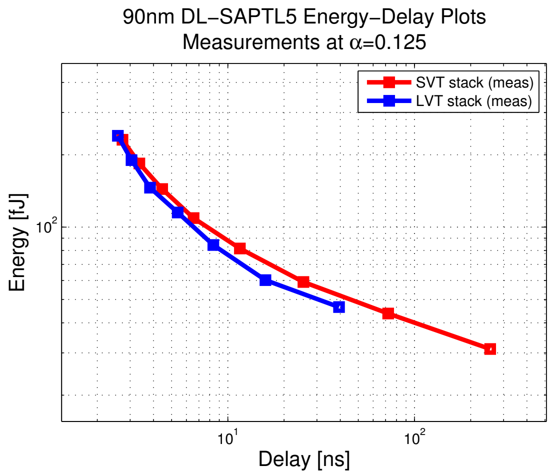


Figure 6.17: The effect of stack threshold on the DL-SAPTL5 energy-delay characteristics.

circuits is a major step forward in creating larger systems in a reasonable amount of time. Most of the design automation concepts, methods and ideas used in the design of the asynchronous SAPTL test chip is again used to design and implement the SAPTL-based CRC generator presented in the next section.

6.3 The SAPTL CRC Circuit

Evaluating the usefulness of the SAPTL based solely on the implemented fundamental building blocks can be misleading. This is due to the fact that the operating environment used to characterize a particular building block is mostly artificial. Also, since most circuits will be made up of various building blocks with different delays and logic depths, it is more meaningful to determine the aggregate behavior of these SAPTL building blocks in an actual computational circuit.

Thus, a 90nm ADSA-based SAPTL CRC circuit was designed and implemented to determine the characteristics of the SAPTL in a larger circuit environment that exercises both (1) the logic functionality of minimized stacks and (2) various handshaking and timing scenarios. By targeting a circuit with a well-defined function, comparisons can also be made with a corresponding static CMOS implementation.

6.3.1 The CRC Algorithm

The cyclic redundancy check (CRC) is a powerful and efficient way to detect errors in communication and storage systems. A serial CRC implementation usually consists of a linear feedback shift register (LFSR). However, in some cases, serial CRC computation speed may be inadequate leading to the use of parallel CRC computation methods [CPR03].

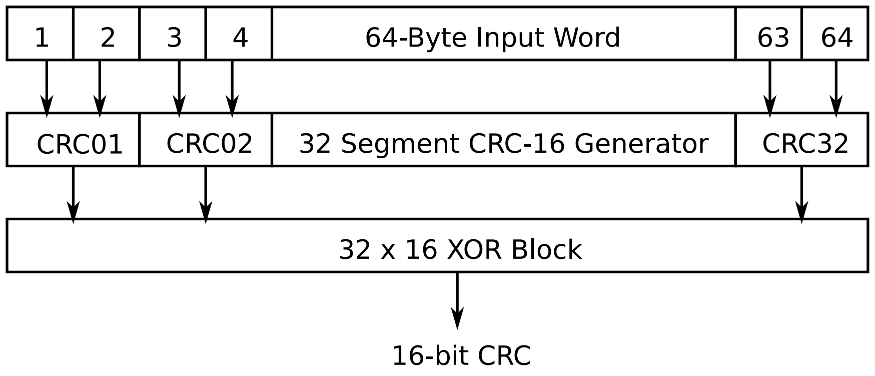


Figure 6.18: The SAPTL CRC-CCITT generator.

The implemented SAPTL CRC generator realizes the CCITT standard [Tel96], and is based on the unrolled out-of-order divide and conquer strategy reported in [Wal07]. Fig. 6.18 shows the architecture of the parallel SAPTL CRC generator. It consists of 32 two-byte 16-bit CRC generators that generate the CRC values associated with each two-byte position. These intermediate CRC values are then combined in the 32×16 XOR block to generate the final 16-bit CCITT CRC value.

6.3.2 Implementation

Each two-byte 16-bit CRC generator segment is composed of 16 logic slices that perform the Galois Field (GF) H-matrix multiplication. Fig. 6.19 shows the organization of these 16-bit CRC generator segments, with each slice representing a GF multiply-accumulate stage. In general, the output of the segment is given by the GF matrix multiplication

$$CRC_i = H^{16i} \times W_{\langle 16(i-1)+15:16(i-1)\rangle} \qquad i \in [1, 16] \tag{6.1}$$

where CRC_i is the i^{th} 16-bit CRC segment output, W is the input word and H is the characteristic 16×16 H-matrix [Wal07]. The values of the particular H-matrix determines which of the inputs need to be XOR-ed together.

A unique H-matrix is associated with each 16-bit CRC segment. Thus, if a particular H-matrix element is '0', the output of the AND gate controlled by that particular H-matrix element can be eliminated, reducing the number of inputs required by the XOR gate. On the other hand, if the H-matrix element is a '1', the input word is simply XOR-ed with other inputs that also correspond to an H-matrix element that is '1'. Each 16-bit segment then can be simplified to an XOR gate whose inputs are determined by the H-matrix.

Pre-calculating the various H-matrices needed by the 64-byte CRC generator results in XOR gates whose inputs range from 7 to 13. Fig. 6.20 shows the SAPTL implementation templates

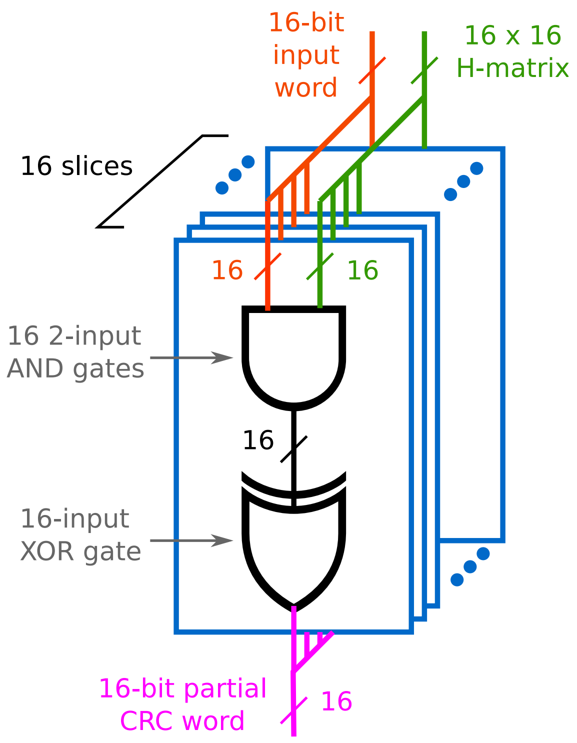


Figure 6.19: CRC16 16-bit segment architecture.

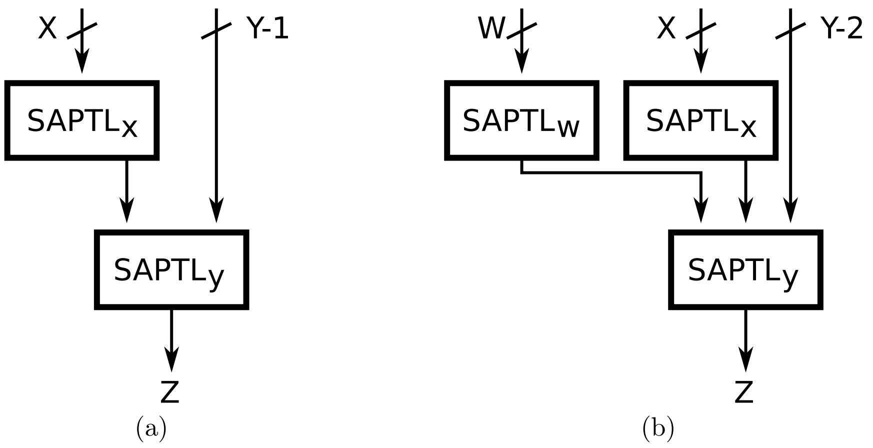


Figure 6.20: SAPTL CRC XOR templates for (a) 7 to 11 inputs and (b) 12 and 13 inputs.

Table 6.2: SAPTL XOR W, X and Y values.

Number of XOR inputs	W	X	Y
7	-	4	4
8	-	4	5
9	-	5	5
10	-	5	6
11	-	6	6
12	5	4	5
13	5	5	5

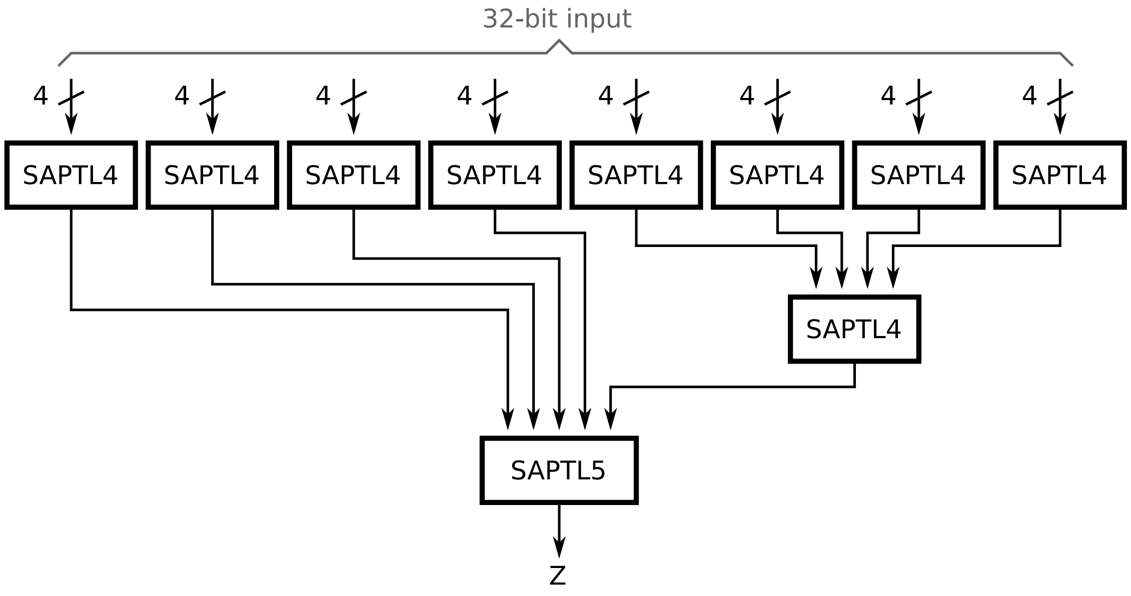


Figure 6.21: SAPTL CRC 32-input XOR.

used to create the different CRC segments, where W, X and Y represent the number of inputs required by the SAPTL XOR gates used as shown in Table 6.2.

The final 16-bit CRC value is computed using 16 32-input XOR gates, and are implemented in a tree-like fashion shown in Fig. 6.21.

The SAPTL CRC segments and final XOR blocks were implemented using a semi-custom layout design flow shown in Fig. 6.22. A Matlab program uses the pre-computed H-matrices to automatically create the the SAPTL XOR segments, based on the templates in Fig. 6.20, then generates a Verilog file containing the appropriate SAPTL blocks and interconnect information.

The entire CRC circuit is built using 4-, 5-, and 6-input SAPTL XOR gates. The custom-made SAPTL gate layouts are then used by the Astro P&R tool together with the generated Verilog file to create the final SAPTL CRC layout.

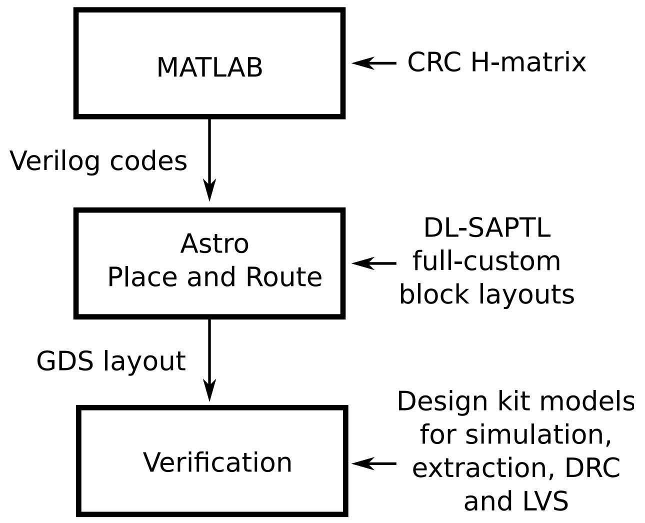

MATLAB
CRC H-matrix
Verilog codes
Astro Place and Route
DL-SAPTL full-custom block layouts
GDS layout
Verification
Design kit models for simulation, extraction, DRC and LVS

Figure 6.22: The SAPTL CRC tool flow.

In order to facilitate the testing and characterization of the SAPTL CRC generator, additional built-in self-test (BIST) circuitry is added on chip. The overall CRC ship organization is shown in Fig. 6.23, and contains the SAPTL CRC circuit, a Test ROM (TROM), a Delay Line (DL), an error detector/comparator and the BIST control unit (BCU).

The TROM contains 18 different 512-bit test vectors that is used to exercise the SAPTL CRC circuit, as well as the corresponding correct CRC values. Upon startup and/or reset, the BCU generates the lowest TROM address and initiates a CRC computation by the appropriate handshaking signal sequence. This handshake sequence is delayed by a fixed time through the DL. As soon as the BCU receives a completion signal from the SAPTL CRC circuit, it increments the TROM address, and initiates another computation.

The DL mimics the longest worst-case critical path of the SAPTL CRC circuit and has selectable delay values of 1, 4, 9 and 99, corresponding to activity factors of 0.5, 0.2, 0.1 and 0.01. An error detection circuit compares the correct CRC value stored in the TROM with the SAPTL CRC output data. An error flag is raised whenever the two values are different, making it easy to determine the lowest operating voltage that can support correct operation.

6.3.3 Simulation Results

The pre-layout simulated energy-delay characteristics of the asynchronous CRC generator, together with its equivalent standard-cell CMOS implementation, are shown in Fig. 6.24, as V_{DD} is swept from 1V to 0.3V with an activity factor of $\alpha = 0.1$. The plot is normalized to 41.2 pJ and 714 ps, which is the simulated energy and delay per operation of the 64-byte parallel CMOS CRC circuit operating at $V_{DD} = 1V$ and $\alpha = 0.1$.

Leakage current is generated in the drivers, SAs and handshaking circuits, which can be

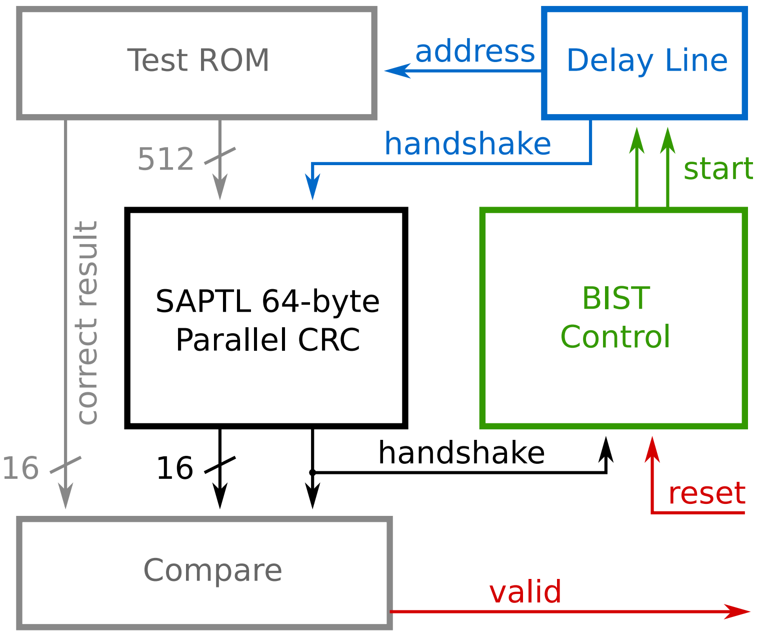


Figure 6.23: SAPTL CRC system block diagram.

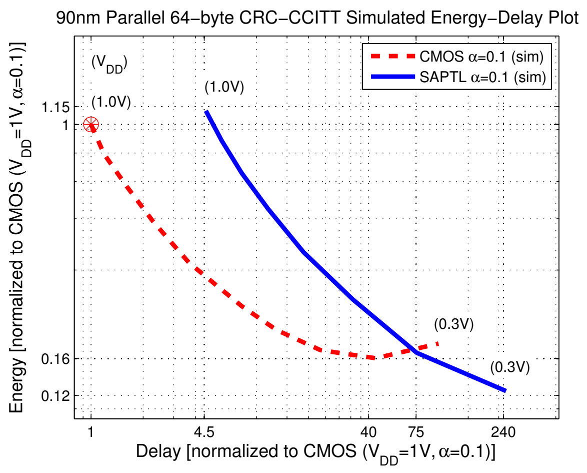
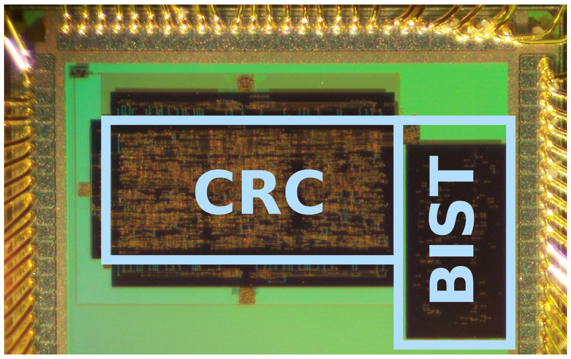
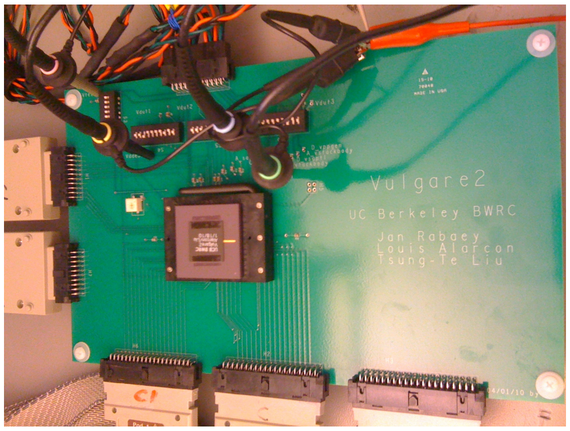


Figure 6.24: Simulated SAPTL CRC-CCITT energy-delay characteristics.



(a)



(b)

Figure 6.25: The AD-SAPTL CRC (a) test chip and (b) circuit board.

minimized by using SAPTL blocks with deeper stacks. In the case of the parallel SAPTL CRC circuit, 1,041 SAPTL blocks are used, with an average stack depth of 5. In an ideal case where leakage is zero, scaling the CMOS supply voltage from 1V to 0.3V results in a 11.11X reduction in energy. The reduced subthreshold leakage energy in the SAPTL CRC generator extends the range of dynamic energy dominated operation, allowing a more effective energy reduction when scaling V_{DD} from 1V to 0.3V, resulting in a 9X reduction in simulated energy, as compared to the 6.2X reduction seen in the static CMOS equivalent over the same voltage range. This reduced leakage allows the SAPTL CRC circuit to achieve a 25% lower minimum energy point occurring at $V_{DD} = 0.3V$, with a 6X delay penalty, which corresponds to approximately 170 ns.

6.3.4 Measurement Results

The SAPTL CRC was implemented in a 90nm triple well CMOS process. A photograph of the 2.1 mm × 2.6 mm die is given in Fig. 6.25a, showing CRC generator, as well as the built-in self-test (BIST) circuitry used to facilitate the measurement and characterization process. The CRC generator itself is 1.1 mm × 0.54 mm in size and the chip was packaged in a SSM/NTK CPG13229 132-pin ceramic PGA. Fig. 6.15b shows the packaged AD-SAPTL CRC test chip mounted on the test board.

The simulation results were verified by measured data and as expected, the measured energy and delay data of the SAPTL CRC generator, shown in Fig. 6.26, is dynamic energy dominated at $\alpha = 0.1$, while leakage starts to dominate at low supply voltages with $\alpha = 0.01$. This leakage energy is mainly due to the handshaking circuits implemented using complementary static CMOS gates. The measured data from the single measured die tracks the pre-layout simulation results, and falls within the expected range, if layout parasitics are considered, as seen in the 90nm asynchronous SAPTL test chip in Sec. 6.2. Post-layout transistor-level

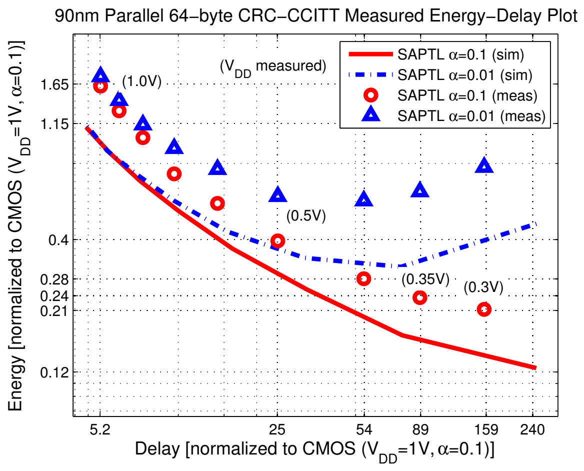


Figure 6.26: Measured SAPTL CRC-CCITT energy-delay characteristics

Table 6.3: Normalized delay (energy) of the SAPTL CRC generator.

	$V_{DD} = 0.5V$ (sim)		$V_{DD} = 0.5V$ (meas)	
$V_{BB,stack}$	$\alpha = 0.1$	$\alpha = 0.01$	$\alpha = 0.1$	$\alpha = 0.01$
0 V	1.00 (1.00)	1.00 (1.00)	1.00 (1.00)	1.00 (1.00)
0.1 V	0.95 (1.00)	0.95 (0.99)	0.98 (0.99)	0.98 (0.99)
0.2 V	0.90 (0.99)	0.91 (0.96)	0.96 (0.99)	0.96 (0.98)
0.3 V	0.86 (0.98)	0.87 (0.96)	0.94 (0.98)	0.93 (0.97)
0.4 V	0.84 (1.00)	0.85 (0.97)	0.92 (0.98)	0.91 (0.97)

simulation was not done due to the extremely long simulation times required. Correcting the measurements results in Fig. 6.26 by including for the expected 30% increase in measured currents due to layout parasitics gives the energy-delay characteristics in Fig. 6.27.

Measurements show a 7.9X decrease in energy by decreasing V_{DD} from 1V to 0.3V, and a 6.9X decrease from 1V to 0.35V. At $V_{DD} = 0.3V$, the ON current that drives the S stack output becomes comparable with the OFF current that drives the $\overline{S}$ stack output, leading to erroneous SA decisions, causing some CRC computational errors to occur. This, however, is not a limitation of the SAPTL topology, but of the sense amplifier architecture used. Since this design uses only one type of SA, one possible solution to address this problem is to tailor the sense amplifier input stage transistor size to the stack depth, pairing SAs with lower input offset voltages with deeper stacks.

Table 6.3 illustrates another major characteristic of the SAPTL: the decoupling of stack threshold voltage from the overall leakage energy. The stack threshold voltage ($V_{TH,stack}$) can be lowered using forward body bias (FBB) resulting in increased performance independent of the sense amplifier energy. As expected, reducing $V_{TH,stack}$ results in an almost 10% decrease in delay, but with almost no impact on energy even at lower activity factors.

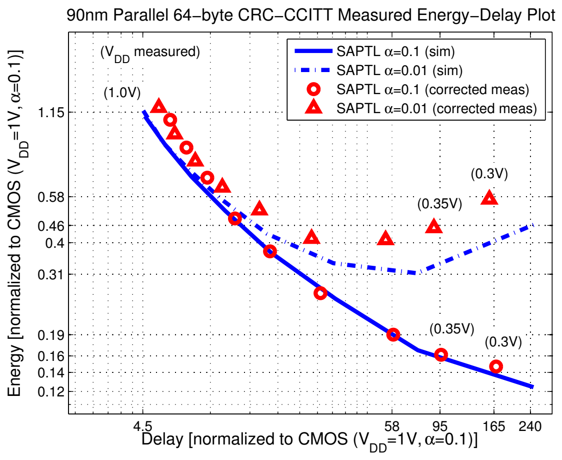


Figure 6.27: The energy-delay characteristics of the measured SAPTL CRC-CCITT circuit with the layout parasitics removed.

6.3.5 Summary

The SAPTL CRC circuit (1) demonstrates the behavior of the SAPTL in an a larger circuit environment compared to previous SAPTL test chips, and demonstrates that the asynchronous handshaking scheme works across varying levels of logic depths and delay paths, and (2) provides a reference point that can be compared to a static CMOS circuit with the same functionality, as opposed to comparing fundamental building blocks carried out in the first two SAPTL chips. Simulation results show that with stack minimization, the SAPTL CRC can achieve a lower minimum energy point when compared to the static CMOS implementation. Furthermore, these simulation results are also consistent with the measured results. The decoupling of stack threshold voltage from energy is evident in both simulation and measurement data, as seen in the increased SAPTL CRC circuit performance with no energy penalty.

6.4 Conclusions

Three 90nm test chips are presented in this chapter: (1) the synchronous SAPTL test chip, (2) the DL-SAPTL test chip and (3) the AD-SAPTL-based CRC test chip. The synchronous SAPTL test chip contains various stack and SAPTL LUT configurations and is used to verify that both stack and sense amplifier behavior can be predicted by simulations. Due to the reduced number of gain elements leading to lower leakage currents, the measured SAPTL energy is dynamic energy dominated at low supply voltages, consuming 10 fJ/operation for a LUT5 at a supply voltage of 300 mV. The DL-SAPTL test chip contains asynchronous delay line-based SAPTL blocks. This asynchronous timing scheme increases the stack area by 20%-30%, and measurement results show an energy increase of up to 1.8X with a 4X improvement in delay.

Both test chips are then used as stepping stones to create a larger, more complex circuit such as a parallel 64-byte CRC circuit. Due to the different stack depths and logic levels present in the CRC circuit, the finer-grained AD-SAPTL asynchronous timing scheme is used in order to reduce the timing overhead. The design and implementation techniques used in the first two SAPTL chips is extended also extended to take into account these different stack depths and logic levels.

The 64-byte parallel AD-SAPTL CRC circuit consumes 5 pJ per operation with a maximum throughput of 5.8×10^6 operations per second at a supply voltage of 300 mV, which is 25% lower in energy and 6X larger in delay when compared to its static CMOS equivalent. However, at this supply voltage some computation errors start to occur. At $V_{DD} = 350$ mV, the CRC circuit exhibits error-free operation with an energy consumption of 9.9 pJ per operation, with a maximum throughput of 1.5×10^7 operations per second.

The decoupling of the stack threshold voltage from the leakage energy is a key SAPTL characteristic. In this case, this characteristic is emphasized when the stack transistor threshold voltages are lowered by forward body biasing. A 10% reduction in delay is observed at a supply voltage of 0.5 V and a forward body bias of 0.4 V without any increase in the total SAPTL energy.

Chapter 7

Summary and Conclusions

Ultra-low energy circuits can enable the emergence of new applications in environments where energy is scarce and/or extremely expensive, such as in the implantable biomedical device space. These new applications are envisioned to driver new design paradigms that include the impact of energy generation and storage on the compute methodology as well as the circuit implementation with the goal of trying to achieve indefinite system lifetimes.

The push to scale devices even further improves circuit performance, but at the expense of increased energy. One major impediment to the reduction of energy per operation in CMOS circuits is the continued increase in subthreshold leakage current as transistor feature sizes decrease. In energy-starved environments where only moderate to low performance is needed, the obvious approach to achieve low energy operation is to reduce the supply voltage. However, at these reduced performance levels where circuit activity is very low, leakage current, taking the form of standby energy, dominates the energy per operation.

The standby current drawn by a logic block flows through paths between the power supply lines and ground, found in circuits that have gain. In complementary static CMOS circuits, every logic gate inherently has gain. In CMOS gates, gain is essential and provides (1) the regeneration of logic levels, improving noise immunity and (2) the drive current needed to propagate the gate output to the inputs of other gates. However, in order to have gain, V_{DD} and ground connections are needed, and CMOS gates will thus exhibit leakage current during standby.

In situations where low activity is the norm, circuits with too much gain will be penalized with increased leakage energy. Reducing the gain through the use of forced stacking, for example, reduces the gain of each gate, as well as the leakage current. However, reducing the gain ironically costs more in terms of additional circuit components needed to reduce this gain, since by its very topology, gain in CMOS gates has already been built-in.

Passive circuits such as pass transistors on the other hand, are circuits without gain, and thus, have no V_{DD} to ground leakage paths. Gain elements such as inverters and other regenerative elements can be added later in the design process, independent of the logic function being implemented.

The sense amplifier-based pass transistor logic (SAPTL) is a logic topology that utilizes an inverted pass transistor tree, or the stack, as a passive logic implementation network. Unlike conventional pass transistor networks, the stack only has feedforward paths from a single root node to two low-swing pseudo-differential outputs. Gain is then added in the form of the root driver and sense amplifier. Note that both the driver and sense amplifier do not contribute anything to the boolean functionality of the logic block. This effective decoupling of functionality and gain is a key SAPTL characteristic.

A system composed of SAPTL blocks can be implemented using more complex or deeper stacks with fewer drivers and sense amplifiers or more drivers and sense amplifiers with shallower, simpler stacks. Thus, the appropriate amount of gain for a given performance and reliability requirement can be controlled at a finer granularity than in static CMOS circuits. The decoupled nature of the stack, driver and sense amplifier also allows separate optimization strategies to be applied to each of these basic SAPTL building blocks.

One such optimization is the lowering of the stack threshold voltage, resulting in improved performance and lower energy consumption. This decoupling of the stack threshold voltage from standby energy is another main key SAPTL property. Thus, the stack can tolerate lower supply voltages while remaining in the super-threshold operating region. Lower stack threshold voltages reduces the delay penalty due to V_{DD} scaling as well as the impact of process variability and mismatch. The creation of reduced and optimized stacks, being similar to the DCVSL pull-down tree, can take advantage of the well-defined DCVSL minimization and optimization techniques. This stack minimization can result in energy reductions of up to 5X, and up to 3X delay reductions for a 5-input XOR stack.

The driver and sense amplifier, on the other hand, being the only source of standby leakage currents, can be optimized using low leakage techniques at the expense of performance. Additionally, the sense amplifier leakage can be traded off for input sensitivity, robustness and/or output drive.

Due to the latch inherent in the SA that needs to be enabled at the end of every computation and due to the stack initialization in order to avoid charge build-up in the passive network, cascading SAPTL blocks requires a pre-determined timing scheme. The most straightforward approach is to use a two-phase non-overlapping scheme that globally separates the stack initialization phase from the evaluation phase, as well as marking the specific point in time when the sense amplifier samples its inputs and latches the correct output value. Thus, cascaded SAPTL blocks are assigned alternating clock phases, with the added benefit of having the driving SAPTL stage initializing the driven stage. The resulting synchronous LUT5 SAPTL achieves a 6X reduction in energy with a 12X-30X increase in delay when compared to its static CMOS equivalent.

Two possible asynchronous timing schemes are presented as an alternative to the synchronous SAPTL timing scheme. Both asynchronous timing schemes incur energy penalties due to the added local handshaking circuitry in exchange for less timing margin. Thus, instead of worst-case design inherent in clocked systems, the average-case performance can be achieved.

The delay line-based asynchronous SAPTL scheme uses a delay line in order to temporally separate the stack evaluation start time from the SA sample time. Since the delay line is matched to its worst-case stack delay, it is possible to use a wide range of stack configurations without having to accommodate the slowest stack delay, resulting in very large margins for the smaller, faster stacks. In addition to the delay line, a Muller C-element is used to perform the basic handshaking operations necessary to cascade the DL-SAPTL blocks. By leveraging this asynchronous behavior, the LUT5 DL-SAPTL can achieve a 73% reduction in delay when compared to the synchronous SAPTL at a supply voltage of 300 mV. However, due to the additional handshake circuitry, this delay reduction results in an 88% increase in energy.

The DL-SAPTL, however, is still limited by the local, data-dependent worst-case delay of its stack. The AD-SAPTL avoids this additional local timing margin by eliminating the delay line and instead, uses a sense amplifier that can be triggered at the same time as the stack, waits until the stack output voltages are large enough, then makes a decision. The additional energy costs due to the more complex sense amplifier is offset by the removal of the delay line, resulting in a simpler handshaking circuitry. Thus for a LUT5 AD-SAPTL block, an 89% reduction in delay and only a 31% increase in energy is obtained when compared to the synchronous SAPTL at $V_{DD} = 0.3$ V.

Three 90nm test chips were implemented showing the behavior of the basic synchronous and asynchronous SAPTL building blocks, as well as a 64-byte parallel CRC generator. In the case of the CRC generator, the AD-SAPTL implementation results in a minimum energy point that is 25% lower than the equivalent static CMOS implementation with a 6X increase in delay. These circuits show the potential of the SAPTL topology to achieve energy points lower than the equivalent complementary static CMOS circuits, in spite of using fully differential signaling and additional memory elements, at the cost of increased delays.

7.1 Contributions

The development of SAPTL as an alternative low energy logic topology has led to the following key contributions, concepts and ideas:

- The development of the inverted pass transistor tree or the stack, that unlike conventional pass transistor tree networks, only allow feed-forward current paths, and thus eliminating sneak leakage paths. In addition, stack energy and delay models are developed, intended to aid future SAPTL development.
- The decoupling of logic functionality from circuit gain in the SAPTL. This decoupling allows independent optimizations to be performed on the stack, sense amplifier and driver. Delay optimizations and topological minimizations are applied to the stack, while leakage, output drive, sensitivity and robustness optimizations are applied to the sense amplifier.

- The development of synchronous and asynchronous SAPTL timing schemes that enable the creation of larger circuits.
- The demonstration of the functionality and behavior of the SAPTL topology through three 90nm test chips. These test circuits also show that both synchronous and asynchronous SAPTL topologies are amenable to design automation, having been built using rudimentary but functional SAPTL-specific automated synthesis, placement and routing methodologies based on commercially available CAD tools.

7.2 Limitations of SAPTL

The separation of logic functionality and gain into two distinct circuit components within SAPTL leads to certain limitations that can render the use of SAPTL-based systems unattractive. In these situations, the costs that are inherent in the SAPTL may outweigh the benefits that can be achieved.

The energy, delay, and area overhead associated with the sense amplifier and the driver makes the SAPTL unsuited for very small logic blocks or systems, where the size of passive logic network is not enough to sufficiently amortize these SA and driver costs. In the extreme case, small SAPTL circuits can contain too much gain, making it a undesirable when compared to the equivalent static CMOS implementation.

In order to cascade SAPTL blocks, a timing scheme is needed even for purely combinational circuits. This timing scheme also adds to the overhead and complexity of the SAPTL circuit which is not present in static CMOS gates. In some applications however, the sense amplifier latch can be used to facilitate pipelining or the implementation of state machines.

Since the SAPTL uses differential signaling, and in the asynchronous timing case, handshaking signals need to be propagated, the amount of interconnect needed to implement SAPTL-based circuits can become a dominant disadvantage. In the case where the technology can support many routing layers, and if the system can be partitioned in a way where most of the logical connections are absorbed in the stack, then this may not be a problem. However, for processes with few routing layers and systems where the resulting logic mapping requires a large number of wiring nets, this may not be the case.

The stack is relatively easy to use when implementing both inverting and non-inverting logic, or both at the same time as in the case of XOR gates. It is desirable to increase the stack fan-in and have very few outputs because it increases the logical complexity of the SAPTL gate and reduces the number of sense amplifiers required. However, in systems where there are more outputs than there are inputs, the number of sense amplifiers driving these outputs can limit the usefulness of SAPTL.

The use of sense amplifiers operating in the subthreshold regime, in order to reduce leakage, can also limit the robustness of SAPTL-based circuits. Without minimization, the use of a full stack can also limit the energy and performance that can be obtained due to the exponential increase in input capacitance with stack depth. While these are very real limitations, these are limitations on the circuit implementation of the SAPTL and not on the basic SAPTL concept of gain and logic decoupling in order to reduce energy consumption.

Taking into account the concepts, ideas and limitations discussed above, several recommendations as well as possible future undertakings are presented in Chapter 8.

Chapter 8

Recommendations and Future Work

The bulk of the this work is concentrated on the development of the basic SAPTL topology, focusing on the behavior of the stack, driver and sense amplifier, as well as the creation of synchronous and asynchronous timing schemes that can serve as the basis for creating larger SAPTL-based circuits. In this context, future SAPTL developments can be made on along several directions.

The design and implementation of more complex sequential circuits and state machines are the most straightforward continuation of this work. Even though timing schemes that allow SAPTL blocks to be cascaded are presented, the logic functions are still purely combinational. Asynchronous FIFO circuits have been constructed for characterization purposes but not as actual sequential units.

At the system-level, exploring SAPTL design issues and opportunities would involve the analysis of timing strategies in conjunction with logic mapping in order to develop methodologies that minimize the energy per operation. Logic mapping tradeoffs such as stack depth vs. sense amplifier and driver cost amortization, at this level, might prove to be complicated enough that algorithms more sophisticated than simple heuristics could be needed. Also of interest in larger designs is the use of mixed timing schemes that involve both global clocking and local handshaking techniques. A system timing scenario similar to the globally-asynchronous-locally-synchronous (GALS) scheme may offer additional degrees of freedom at the system-level. Since the SAPTL can easily accommodated both clocked and self-timed schemes, there could be less overhead and timing issues if both schemes are used in the same system.

For static CMOS gates, the energy due to standby current, as well as the effect of process variability and device mismatch, is expected to increase as technology scales further. Thus, alternative circuit implementations of the various SAPTL components may also improve the viability of SAPTL-based systems. At these extremely scaled processes, the use of current-mode devices, for instance, for root drivers, sense amplifiers and handshaking circuits that work with subthreshold leakage currents rather than suppress it, could produce improved energy and delay characteristics, leveraging the SAPTL's effective decoupling of the stack threshold voltage and leakage current. The inherent regularity of the SAPTL topology can help reduce the impact of variability especially if the stack can be made to operate in the super-threshold region.

Finally, at the device-level, the SAPTL could be another motivating factor in the development for low-threshold devices designed to operate above-V_{TH} even for very low supply voltages. Low loss MEMS-based switches for use as stack transistor replacements can reduce the amount of gain needed by SAPTL circuits even further, again reducing the energy per operation in digital gates.

Bibliography

[AKBE96] I.S. Abu-Khater, A. Bellaouar, and M.I. Elmasry, *Circuit techniques for cmos low-power high-performance multipliers*, Solid-State Circuits, IEEE Journal of **31** (1996), no. 10, 1535–1546.

[ALPR07] L.P. Alarcón, T.-T. Liu, M.D. Pierson, and J.M. Rabaey, *Exploring Very Low-Energy Logic: A Case Study*, Journal of Low Power Electronics **3** (2007), no. 3, 223–233.

[BFG+06] K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer, *High-performance cmos variability in the 65-nm regime and beyond*, IBM Journal of Research and Development **50** (2006), no. 4.5, 433 –449.

[BRG05] UC Berkeley BSIM Research Group, *Bsim3v3.3 release*, `http://www-device.eecs.berkeley.edu/~bsim3`, July 2005.

[BS99] G. Balamurugan and N.R. Shanbhag, *Energy-efficient dynamic circuit design in the presence of crosstalk noise*, Low Power Electronics and Design, 1999. Proceedings. 1999 International Symposium on, 1999, pp. 24 – 29.

[BTK+09] K.A. Bowman, J.W. Tschanz, Nam Sung Kim, J.C. Lee, C.B. Wilkerson, S.-L.L. Lu, T. Karnik, and V.K. De, *Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance*, Solid-State Circuits, IEEE Journal of **44** (2009), no. 1, 49 –63.

[CB95] A.P. Chandrakasan and R.W. Brodersen, *Minimizing power consumption in digital cmos circuits*, Proceedings of the IEEE **83** (1995), no. 4, 498 –523.

[CC06] B.H. Calhoun and A.P. Chandrakasan, *Ultra-dynamic voltage scaling (udvs) using sub-threshold operation and local voltage dithering*, Solid-State Circuits, IEEE Journal of **41** (2006), no. 1, 238 – 245.

[CCL+08] B.H. Calhoun, Yu Cao, Xin Li, Ken Mai, L.T. Pileggi, R.A. Rutenbar, and K.L. Shepard, *Digital circuit design challenges and opportunities in the era of nanoscale cmos*, Proceedings of the IEEE **96** (2008), no. 2, 343 –365.

[CFK$^+$10] G. Chen, M. Fojtik, D. Kim, D. Fick, J. Park, M. Seok, M.-T. Chen, Z. Foo, D. Sylvester, and D. Blaauw, *Millimeter-scale nearly perpetual sensor system with stacked battery and solar cells*, ISSCC Digest of Technical Papers, vol. 53, February 2010, pp. 288–289.

[CHC04] B.H. Calhoun, F.A. Honore, and A.P. Chandrakasan, *A leakage reduction methodology for distributed mtcmos*, Solid-State Circuits, IEEE Journal of **39** (2004), no. 5, 818 – 826.

[CL96] Kuo-Hsing Cheng and Yii-Yih Liaw, *A low-power current-sensing complementary pass-transistor logic (lcscptl) for low-voltage high-speed applications*, VLSI Circuits, 1996. Digest of Technical Papers., 1996 Symposium on, Jun 1996, pp. 16–17.

[CP86] K.M. Chu and D.L. Pulfrey, *Design procedures for differential cascode voltage switch circuits*, Solid-State Circuits, IEEE Journal of **21** (1986), no. 6, 1082 – 1087.

[CPR03] G. Campobello, G. Patane, and M. Russo, *Parallel crc realization*, Computers, IEEE Transactions on **52** (2003), no. 10, 1312 – 1319.

[CSB$^+$97] P.Y.K. Cheung, M.V. Scotti, J. Blake, B. Brewer, R. Grisenthwaite, D. Hitchcox, and P. Shepherd, *High speed arithmetic design using cpl and dpl logic*, Solid-State Circuits Conference, 1997. ESSCIRC '97. Proceedings of the 23rd European, Sept. 1997, pp. 360–363.

[CSH$^+$03] B. Chatterjee, M. Sachdev, S. Hsu, R. Krishnamurthy, and S. Borkar, *Effectiveness and scaling trends of leakage control techniques for sub-130 nm cmos technologies*, Low Power Electronics and Design, 2003. ISLPED '03. Proceedings of the 2003 International Symposium on, Aug. 2003, pp. 122–127.

[DC97] A. Dancy and A. Chandrakasan, *Techniques for aggressive supply voltage scaling and efficient regulation [cmos digital circuits]*, Custom Integrated Circuits Conference, 1997., Proceedings of the IEEE 1997, 5-8 1997, pp. 579 –586.

[DRL$^+$06] S. Das, D. Roberts, Seokwoo Lee, S. Pant, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, *A self-tuning dvs processor using delay-error detection and correction*, Solid-State Circuits, IEEE Journal of **41** (2006), no. 4, 792 – 804.

[EKV95] Christian C. Enz, François Krummenacher, and Eric A. Vittoz, *An analytical mos transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications*, Analog Integr. Circuits Signal Process. **8** (1995), no. 1, 83–114.

[Elm48] W. C. Elmore, *The transient analysis of damped linear networks with particular regard to wideband amplifiers*, J. Appl. Phys. **19** (1948), 55–63.

[GN04] M. Ghovanloo and K. Najafi, *A Modular 32-Site Wireless Neural Stimulation Microsystem*, IEEE Journal of Solid-State Circuits **39** (2004), no. 12.

[HGDT84] L. Heller, W. Griffin, J. Davis, and N. Thoma, *Cascode voltage switch logic: A differential cmos logic family*, Solid-State Circuits Conference. Digest of Technical Papers. 1984 IEEE International, vol. XXVII, feb 1984, pp. 16 – 17.

[HTH$^+$99] M. Hamada, T. Terazawa, T. Higashi, S. Kitabayashi, S. Mita, Y. Watanabe, M. Ashino, H. Hara, and T. Kuroda, *Flip-flop selection technique for power-delay trade-off [video codec]*, Solid-State Circuits Conference, 1999. Digest of Technical Papers. ISSCC. 1999 IEEE International, 1999, pp. 270 –271.

[ITN$^+$00] T. Inukai, M. Takamiya, K. Nose, H. Kawaguchi, T. Hiramoto, and T. Sakurai, *Boosted gate mos (bgmos): device/circuit cooperation scheme to achieve leakage-free giga-scale integration*, Custom Integrated Circuits Conference, 2000. CICC. Proceedings of the IEEE 2000, 2000, pp. 409 –412.

[ITR] *The International Technology Roadmap for Semiconductors (ITRS) 2009 Edition*, Tech. report, http://www.itrs.net/Links/2009ITRS/Home2009.htm.

[KAB$^+$03] N.S. Kim, T. Austin, D. Blaauw, T. Mudge, K. Flautner, J.S. Hu, M.J. Irwin, M. Kandemir, and V. Narayanan, *Leakage Current: Moore's Law Meets Static Power*, IEEE Computer **36** (2003), 68–75.

[KGM09] V. Karkare, S. Gibson, and D. Markovic, *A 130-μw, 64-channel spike-sorting dsp chip*, Solid-State Circuits Conference, 2009. A-SSCC 2009. IEEE Asian, nov. 2009, pp. 289 –292.

[KKJ01] Bai-Sun Kong, Sam-Soo Kim, and Young-Hyun Jun, *Conditional-capture flip-flop for statistical power reduction*, Solid-State Circuits, IEEE Journal of **36** (2001), no. 8, 1263 –1271.

[KNB$^+$99] A. Keshavarzi, S. Narendra, S. Borkar, C. Hawkins, K. Royi, and V. De, *Technology Scaling Behavior of Optimum Reverse Body Bias for Standby Leakage Power Reduction in CMOS IC's*, International Symposium on Low Power Electronics and Design, 1999, pp. 252–254.

[KNC02] J. Kao, S. Narendra, and A. Chandrakasan, *Subthreshold leakage modeling and reduction techniques [ic cad tools]*, Computer Aided Design, 2002. ICCAD 2002. IEEE/ACM International Conference on, 10-14 2002, pp. 141 – 148.

[KNS98] H. Kawaguchi, K.-I. Nose, and T. Sakurai, *A cmos scheme for 0.5 v supply voltage with pico-ampere standby current*, Solid-State Circuits Conference, 1998. Digest of Technical Papers. 1998 IEEE International, 5-7 1998, pp. 192 –193, 436.

[KRH$^+$05] V. Kheterpal, V. Rovner, T.G. Hersan, D. Motiani, Y. Takegawa, A.J. Strojwas, and L. Pileggi, *Design methodology for ic manufacturability based on regular logic-bricks*, Design Automation Conference, 2005. Proceedings. 42nd, 13-17 2005, pp. 353 – 358.

[KSM$^+$98] T. Kuroda, K. Suzuki, S. Mita, T. Fujita, F. Yamane, F. Sano, A. Chiba, Y. Watanabe, K. Matsuda, T. Maeda, T. Sakurai, and T. Furuyama, *Variable supply-voltage scheme for low-power high-speed cmos digital design*, Solid-State Circuits, IEEE Journal of **33** (1998), no. 3, 454 –462.

[Kur02] T. Kuroda, *Low-power, high-speed cmos vlsi design*, Computer Design: VLSI in Computers and Processors, 2002. Proceedings. 2002 IEEE International Conference on, 2002, pp. 310–315.

[LAPR09] Tsung-Te Liu, L.P. Alarcon, M.D. Pierson, and J.M. Rabaey, *Asynchronous computing in sense amplifier-based pass transistor logic*, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on **17** (2009), no. 7, 883 –892.

[LS93] D. Liu and C. Svensson, *Trading speed for low power by choice of supply and threshold voltages*, Solid-State Circuits, IEEE Journal of **28** (1993), no. 1, 10 –17.

[MD00] J.D. Meindl and J.A. Davis, *The fundamental limit on binary switching energy for terascale integration (tsi)*, Solid-State Circuits, IEEE Journal of **35** (2000), no. 10, 1515 –1516.

[MDM$^+$95] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada, *1-v power supply high-speed digital circuit technology with multithreshold-voltage cmos*, Solid-State Circuits, IEEE Journal of **30** (1995), no. 8, 847 –854.

[MHU$^+$94] M. Matsui, H. Hara, Y. Uetani, Lee-Sup Kim, T. Nagamatsu, Y. Watanabe, A. Chiba, K. Matsuda, and T. Sakurai, *A 200 mhz 13 mm2 2-d dct macrocell using sense-amplifying pipeline flip-flop scheme*, Solid-State Circuits, IEEE Journal of **29** (1994), no. 12, 1482–1490.

[Mit10] S. Mitra, *Robust system design*, VLSI Design, 2010. VLSID '10. 23rd International Conference on, 3-7 2010, pp. 434 –439.

[MNO00] D. Markovic, B. Nikolic, and V. G. Oklobdzija, *A general method in synthesis of pass-transistor circuits*, Microelectronics Journal **31** (2000), no. 11-12, 991 – 998.

[MWA$^+$10] D. Markovic, C.C. Wang, L.P. Alarcon, Tsung-Te Liu, and J.M. Rabaey, *Ultralow-power design in near-threshold region*, Proceedings of the IEEE **98** (2010), no. 2, 237 –252.

[NII$^+$93] Y. Nakagome, K. Itoh, M. Isoda, K. Takeuchi, and M. Aoki, *Sub-1-v swing internal bus architecture for future low-power ulsis*, Solid-State Circuits, IEEE Journal of **28** (1993), no. 4, 414 –419.

[NRC08] R. Naik, R. Rao, and Chandrasekhar, *Efficient crosstalk estimation and reduction in high speed designs*, Electronic Design, 2008. ICED 2008. International Conference on, 1-3 2008, pp. 1 –6.

[OD95] V.G. Oklobdzija and B. Duchene, *Pass-transistor dual value logic for low-power cmos*, VLSI Technology, Systems, and Applications, 1995. Proceedings of Technical Papers. 1995 International Symposium on, May-2 Jun 1995, pp. 341–344.

[OD97] ______, *Synthesis of high-speed pass-transistor logic*, Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on **44** (1997), no. 11, 974–976.

[PdGT04] J. Pineda de Gyvez and H.P. Tuinhout, *Threshold voltage mismatch and intra-die leakage current in digital cmos circuits*, Solid-State Circuits, IEEE Journal of **39** (2004), no. 1, 157 – 168.

[PHS96] A. Parameswar, H. Hara, and T. Sakurai, *A swing restored pass-transistor logic-based multiply and accumulate circuit for multimedia applications*, Solid-State Circuits, IEEE Journal of **31** (1996), no. 6, 804–809.

[PN09] Liang-Teck Pang and B. Nikolic, *Measurements and analysis of process variability in 90 nm cmos*, Solid-State Circuits, IEEE Journal of **44** (2009), no. 5, 1655 –1663.

[PR07] M. D. Pierson and J. M. Rabaey, *Automated design for current-mode pass-transistor logic blocks*, Master's thesis, EECS Department, University of California, Berkeley, May 2007.

[RCN03] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2 ed., Prentice Hall, 2003.

[RGDMC09] A. Ricci, M. Grisanti, I. De Munari, and P. Ciampolini, *Improved pervasive sensing with rfid: An ultra-low power baseband processor for uhf tags*, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on **17** (2009), no. 12, 1719 –1729.

[ROSO05] J.N. Rodrigues, T. Olsson, L. Sörnmo, and V. Öwall, *Digital Implementation of a Wavelet-Based Event Detector for Cardiac Pacemakers*, IEEE Trans. on Circuits and Systems I: Regular Papers **52** (2005), no. 12.

[Sak03] T. Sakurai, *Perspectives on power-aware electronics*, Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC. 2003 IEEE International, 2003, pp. 26–29 vol.1.

[SEE96] M. Shams, J.C. Ebergen, and M.I. Elmasry, *A comparison of cmos implementations of an asynchronous circuits primitive: the c-element*, Low Power Electronics and Design, 1996., International Symposium on, 12-14 1996, pp. 93 –96.

[Sha49] C.E. Shannon, *The synthesis of two-terminal switching circuits*, Bell System Technical Journal **28** (1949), no. 1, 59–98.

[SKK97] T. Sakurai, H. Kawaguchi, and T. Kuroda, *Low-power cmos design through vth control and low-swing circuits*, Low Power Electronics and Design, 1997. Proceedings., 1997 International Symposium on, Aug 1997, pp. 1–6.

[SN90] T. Sakurai and A.R. Newton, *Alpha-power law mosfet model and its applications to cmos inverter delay and other formulas*, Solid-State Circuits, IEEE Journal of **25** (1990), no. 2, 584 –594.

[SOS$^+$93] M. Suzuki, N. Ohkubo, T. Shinbo, T. Yamanaka, A. Shimizu, K. Sasaki, and Y. Nakagome, *A 1.5-ns 32-b cmos alu in double pass-transistor logic*, Solid-State Circuits, IEEE Journal of **28** (1993), no. 11, 1145–1151.

[Sta98] M.R. Stan, *Low threshold cmos circuits with low standby current*, Low Power Electronics and Design, 1998. Proceedings. 1998 International Symposium on, 10-12 1998, pp. 97 – 99.

[Sta01] ______, *Low-power cmos with subvolt supply voltages*, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on **9** (2001), no. 2, 394 –400.

[Sut89] I. E. Sutherland, *Micropipelines*, Commun. ACM **32** (1989), no. 6, 720–738.

[Tak98] K. Taki, *A survey for pass-transistor logic technologies-recent researches and developments and future prospects*, Design Automation Conference 1998. Proceedings of the ASP-DAC '98. Asia and South Pacific, Feb 1998, pp. 223–226.

[TBK$^+$10] J. Tschanz, K. Bowman, M. Khellah, C. Wilkerson, B. Geuskens, D. Somasekhar, A. Raychowdhury, J. Kulkarni, C. Tokunaga, Shih-Lien Lu, T. Karnik, and V. De, *Resilient design in scaled cmos for energy efficiency*, Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific, 18-21 2010, pp. 625 –625.

[TBLV08] A. Tajalli, E.J. Brauer, Y. Leblebici, and E. Vittoz, *Subthreshold source-coupled logic circuits for ultra-low-power applications*, Solid-State Circuits, IEEE Journal of **43** (2008), no. 7, 1699 –1710.

[TBW$^+$09] J. Tschanz, K. Bowman, C. Wilkerson, Shih-Lien Lu, and T. Karnik, *Resilient circuits - enabling energy-efficient performance and reliability*, Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on, 2-5 2009, pp. 71 –73.

[Tel96] Telecommunication Standardization Sector of ITU, *Recommendation x.25*, oct 1996.

[TN98] Y. Taur and T. Ning, *Fundamentals of Modern VLSI Devices*, Cambridge University Press, UK, 1998.

[UTB$^+$06] O.S. Unsal, J.W. Tschanz, K. Bowman, V. De, X. Vera, A. Gonzalez, and O. Ergin, *Impact of parameter variations on circuits and microarchitecture*, Micro, IEEE **26** (2006), no. 6, 30 –39.

[VCT$^+$04] A. Vladimirescu, Yu Cao, O. Thomas, Huifang Qin, D. Markovic, A. Valentian, R. Ionita, J. Rabaey, and A. Amara, *Ultra-low-voltage robust design issues in deep-submicron cmos*, Circuits and Systems, 2004. NEWCAS 2004. The 2nd Annual IEEE Northeast Workshop on, 20-23 2004, pp. 49 – 52.

[Wal07] M. Walma, *Pipelined cyclic redundancy check (crc) calculation*, Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on, aug. 2007, pp. 365 –370.

[WC05] A. Wang and A. Chandrakasan, *A 180-mv subthreshold fft processor using a minimum energy design methodology*, Solid-State Circuits, IEEE Journal of **40** (2005), no. 1, 310 – 319.

[WPW00] Qing Wu, M. Pedram, and Xunwei Wu, *Clock-gating and its application to low power design of sequential circuits*, Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on **47** (2000), no. 3, 415 –420.

[YA00] T. Yamashita and K. Asada, *Cspl: a capacitor-separated pass-transistor logic*, ASICs, 2000. AP-ASIC 2000. Proceedings of the Second IEEE Asia Pacific Conference on, 2000, pp. 29–32.

[Yu04] Shaofeng Yu, *Extremely Scaled Planar Bulk CMOS: Challenges and Options*, 7th International Conference on Solid-State and Integrated Circuits Technology, October 2004, pp. 41–46.

[YYN$^+$90] K. Yano, T. Yamanaka, T. Nishida, M. Saito, K. Shimohigashi, and A. Shimizu, *A 3.8-ns cmos 16x16-b multiplier using complementary pass-transistor logic*, Solid-State Circuits, IEEE Journal of **25** (1990), no. 2, 388–395.

[ZBSF05] Bo Zhai, D. Blaauw, D. Sylvester, and K. Flautner, *The limit of dynamic voltage scaling and insomniac dynamic voltage scaling*, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on **13** (2005), no. 11, 1239 – 1252.

Appendix A

Exploring the V_{TH} Space: A Simulation Technique

One way to reduce the effective series resistance of the pass transistors in the stack is to reduce the stack threshold voltage. This in turn decreases the stack delay without a corresponding increase in stack energy.

The stack threshold voltage, $V_{TH,stack}$, can be reduced using forward body bias (FBB). However, the reduction in the stack threshold voltage, $\Delta V_{TH,stack}$, due to FBB is limited to 10% to 30% of $V_{TH,stack}$, as seen in Fig. A.1. In order to explore a larger V_{TH} design space, an artificial method of reducing the threshold voltage is used and is shown in Fig. A.2. In this method, a voltage source is added at the gate of the stack transistors, simulating an effective stack threshold voltage of:

$$V_{TH,stack} = V_{TH,nominal} - \Delta V_{TH} \tag{A.1}$$

By comparing the effects of (1) FBB and (2) the artificial V_{TH} reduction method, as applied to a 21-stage ring oscillator for the various transistor flavors available and for V_{DD} of 0.5 V, an estimate of the resulting effective threshold voltage due to FBB can be obtained. Figs. A.3a and A.3b show the equivalence of the two methods for different transistor flavors, and how the artificial V_{TH} reduction method extends the V_{TH}-space that can be explored beyond the limits of FBB.

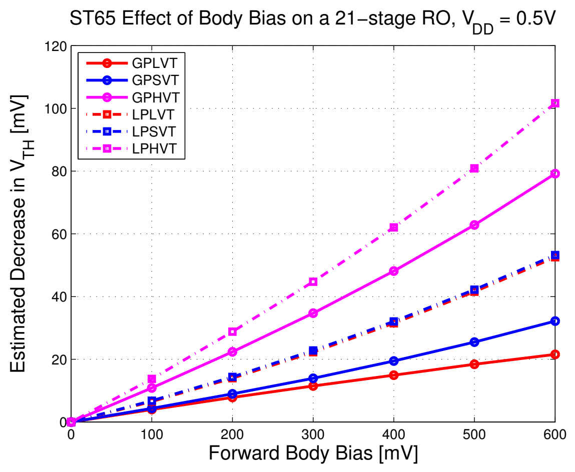


Figure A.1: The effectiveness of forward body bias.

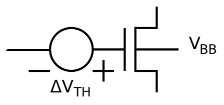
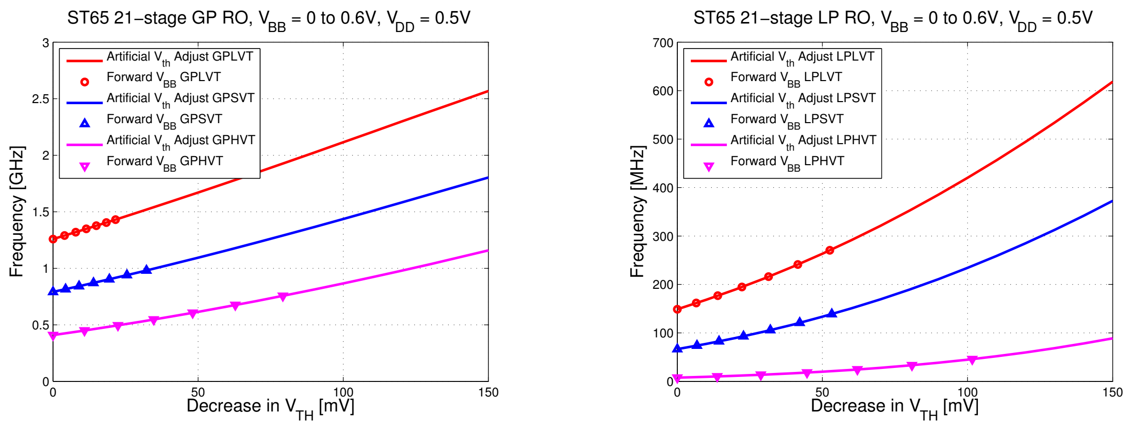


Figure A.2: Artificial threshold voltage reduction.



(a) (b)

Figure A.3: V_{TH} reduction methods for (a) GP and (b) LP devices.