# An Ultra-Low Voltage FFT Processor Using Energy-Aware Techniques

by

Alice Wang

Bachelor of Science in Electrical Science and Engineering
Massachusetts Institute of Technology, 1997

Master of Engineering in Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 1998

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

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
[February 2004]
December 2003



Author .....
Department of Electrical Engineering and Computer Science
December 15, 2003

Certified by .....
Anantha Chandrakasan
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Professor of Electrical Engineering and Computer Science
Chairman, Department Committee on Graduate Students

# An Ultra-Low Voltage FFT Processor Using Energy-Aware Techniques

by

Alice Wang

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

## Abstract

In a number of emerging applications such as wireless sensor networks, system lifetime depends on the energy efficiency of computation and communication. The key metric in such applications is the energy dissipated per function rather than traditional ones such as clock speed or silicon area. Hardware designs are shifting focus toward enabling energy-awareness, allowing the processor to be energy-efficient for a variety of operating scenarios. This is in contrast to conventional low-power design, which optimizes for the worst-case scenario. Here, three energy-quality scalable hooks are designed into a real-valued FFT processor: variable FFT length (N=128 to 1024 points), variable bit precision (8,16 bit), and variable voltage supply with variable clock frequency ($V_{DD}$=180mV to 0.9V, and $f$=164Hz to 6MHz).

A variable-bit-precision and variable-FFT-length scalable FFT ASIC using an off-the-shelf standard-cell logic library and memory only scales down to 1V operation. Further energy savings is achieved through ultra-low voltage-supply operation. As performance requirements are relaxed, the operating voltage supply is scaled down, possibly even below the threshold voltage into the subthreshold region. When lower frequencies cause leakage energy dissipation to exceed the active energy dissipation, there is an optimal operating point for minimizing energy consumption. Logic and memory design techniques allowing ultra-low voltage operation are employed to study the optimal frequency/voltage operating point for the FFT. A full-custom implementation with circuit techniques optimized for deep voltage scaling into the subthreshold regime, is fabricated using a standard CMOS 0.18μm logic process and functions down to 180mV. At the optimal operating point where the voltage supply is 350mV, the FFT processor dissipates 155nJ/FFT. The custom FFT is 8x more energy-efficient than the ASIC implementation and 350x more energy-efficient than a low-power microprocessor implementation.

Thesis Supervisor: Anantha P. Chandrakasan
Title: Professor of Electrical Engineering and Computer Science

# Acknowledgments

First, I am extremely grateful to my advisor, Professor Anantha Chandrakasan. When I first started my PhD, I was new to circuit design and VLSI research, but he took a chance on me and offered me a position in his group. I appreciate his confidence in me, wisdom, honesty, and friendship. He was always available to discuss anything with me from research directions, job decisions, paper submissions, and down-to-the-wire day-of-tape-out circuit advice. I have always looked up to Anantha for his endless pursuit of excellence and high ethical standards. Thanks for keeping my best interests at heart and for the many laughs and good times!

At the heart of MIT are all of the brilliant professors, many whom have been great mentors and granted me much needed advice. I am grateful to Professor Charles Sodini for his support and for allowing me to TA his class, even though I was learning the material at the same time. Thanks to Professor Donald Troxel, my graduate councillor and thesis reader. Professor Troxel has been a part of my graduate career from the very beginning, has given me many insightful comments and advice during our many meetings. Also I want to thank my thesis reader, Professor Michael Perrott. Professor Perrott has always taken an interest in my research and given many helpful suggestions. Thanks for your encouragement and sharing your experience and advice about career decisions.

Spending so many long days and nights at the office would not be a pure joy and pleasure if I wasn't surrounded by such a great group of graduate students. Thanks especially to Ben Calhoun and Fred Lee for sharing your lives with me, for your fellowship, and for your prayers. Also I am grateful to Wendi Heinzelman who greatly helped me in the beginning of my graduate career. She let me tag-along with her on her research until I could get my own footing. Also thanks to Amit Sinha, my cubicle-neighbor of many years. Thanks for the conversations that could swing from low-power research to last night's TV lineup. Also, thanks to Julia Cline who helped me with the multiplier design and for all of the great conversations about subthreshold circuits. I also want to thank Iliana Fujimori Chen for her friendship and much-needed Tosci's breaks from sitting in front of Cadence. I am also thankful for the characters that make Ananthagroup a big party

every day: SeongHwan Cho, Rex Min, David Wentzloff, Jim Goodman, Duke Xanthopoulos, Manish Bhardwaj, Eugene Shih, Paul-Peter Sotiriadis, Travis Simpkins, Raul Blazquez-Fernandez, Puneet Newaskar, Kevin Atkinson, Frank Honore, Nathan Ickes, the Anantha-girls (Nisha Checka, Alexandra Kern, Julia, and Johnna Powell) and the Anantha-Canadians (Daniel Finchelstein, Denis Daly, Naveen Verma, Brian Ginsberg, and Payam Lajevardi). Thank you to Ben, Julia, Daniel and Denis who helped to proofread my work! Also, life in the office would be chaos without our administrative assistant, Margaret Flaherty. I appreciate Margaret for her willingliness to help on so many occasions, for being our co-conspirator, and for her generosity.

One of the greatest things about being at MIT is the tremendous support we receive for the CAD tools both from the support staff and from other graduate students. Thanks to Mike Hobbs and Mike McIllrath for your promptness and willingness to fix any and all computer and software problems. Also I want to thank Scott Meninger, Andrew Chen, and all of the other students on the second floor in Sodinigroup, Hsleegroup and Perrottgroup who have provided tremendous knowledge in the world of circuit design.

It has been a immense privilege to be able to interact with many from industry during my graduate career. Thanks to Wanda Gass for my summer at Texas Instruments. She has been a friendly face at conferences and a great mentor in the field of solid-state circuits. Also I am grateful to Stephen Kosonocky for my summer spent at IBM T.J. Watson. The research from my summer internship at IBM working on the energy-performance contours for subthreshold circuits, became an important turning point in my thesis. My graduate career was supported by Lucent Technologies through the Graduate Research Fellowship Program for Women and by the Intel PhD Fellowship. I would like to thank my mentors, Aon Mujtaba and Ian Young, for their interest in my research. I also had the opportunity to be a part of the DARPA PAC/C project which funded the fabrication of my two chips.

What would my time in Boston be without my friends? My friends are part of my support system especially in the last few years that were filled with sleep-deprivation, stress, and multiple tape-out deadlines. I truly appreciate the close bond I have formed with each of the women who have filtered through 25 Willow Avenue: Maja Razlog, Anna Lysyanskaya, Bonnie Shen, Angela Gong, and Alantha Newman (honorary roommate). You have fed me, comforted me, taught me, and then partied with me. Also a special thanks to Huan

Yao for your constant friendship during the last decade. We have truly seen a lot together! I am also hugely blessed with the girls from Parakaleo. You have truly seen the bad, prayed for me during the worst, and walked with me the entire way. I am grateful to Lucia Wu, Helen Chen, Davina Ling, Hannah Pang, Tinyee Tsai, and Christine Cheuk for your fellowship! There are so many others at Boston Chinese Evangelical Church that I wish to thank for providing a spiritual home and community. Thanks Richard Chung, Hamilton Ho, Candy Liang, Pastor Steve Chin, Pastor Jack Lum, Enoch Liao, and the whole BCEC family.

I am so thankful for my sister and best friend, Elaine. She is the only person who can make me laugh so hard and that's because she knows me the bestest. Elaine, you've been a constant encouragement and I appreciate your honesty and friendship. Thank you for putting up with me and for the knowledge that you will always be there for me. "Lord help the mister who comes between me and my sister."

Also I am grateful to my parents, David and Julie Wang, who have sacrificed a lot to provide for me. Thank you dad for tirelessly teaching me about Fourier integrals way back in high school, even though I did not understand calculus and got easily frustrated. Little did we both know back then, my PhD thesis would be on the very same topic! Also thank you mom for always wanting the best for me, from providing a top-notch education to making sure I ate well and had my flu shots. I am thankful to have you both as my parents and I love you both very much.

All of these great things would not be possible without acknowledging our Creator, our Lord, and our Savior. Thank you God for being the bringer of all good things and for your strength that sustains me. All of this is more than I could ever hope for.

*"Trust in the Lord with all your heart and lean not on your own understanding; in all your ways acknowledge him, and he will make your paths straight." (Proverbs 3:5-6)*

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

In emerging applications such as wireless sensor networks, extending the system lifetime is equivalent to minimizing the energy of communication and computation. A wireless sensor network contains tens to thousands of microsensor nodes that process information from the sensing environment and communicate the information back to the end-user. The microsensor node is required to live up to 5 years while only using the energy from a single "AA" battery. This puts a constraint on the average power dissipated by the microsensor to be less than 10μW. Therefore, the key metric in this application is energy dissipated per function rather than clock speed or silicon area. Another important design consideration is energy-awareness. An energy-aware design has energy-quality scalable "hooks" that allow the processor to be energy-efficient for a variety of operating scenarios. Because the environment around the sensor node will evolve, the microsensor achieves large energy savings by dynamically adjusting energy consumption to changing conditions.

In an application with challenging energy constraints, dedicated hardware is able to achieve high energy-efficiency. Many implementations utilize general purpose hardware for the ease of programmability and interfacing. However, orders of magnitude of energy savings can be achieved through a dedicated processor. By using hardware that is tailored to the application, the overhead of computation is minimized. However, because the application requires adaptability to changing conditions, energy-scalable hooks specific to the application are designed into the architecture of the processor.

Energy savings can also be achieved through supply voltage and clock frequency scaling. For applications without latency constraints, both the supply voltage and clock frequency can be scaled down as low as possible to minimize the switching energy dissipation. However, in some cases, there is a limit to voltage scaling, because the low clock frequencies cause large leakage energy dissipation which increases the total energy dissipated. Exploration into the optimal operating point as a function of parameters of the

applications, such as activity factor, is necessary to minimize energy dissipation. In many cases, the optimal supply voltage for the entire system is below the threshold voltage. In these cases, subthreshold circuit design is needed.

There are very few examples of entire systems designed to operate in the subthreshold regime. It has been shown that combinatorial logic found in arithmetic units or datapath circuits, can scale down to hundreds of millivolt supply voltages [8][30]. However, systems that include large memory structures and complex control logic as well as datapath circuits has yet to be demonstrated. Also, it is typical for low-voltage implementations to rely on special low-voltage process technologies or on body-biasing techniques. This thesis contains the first application specific fast Fourier Transform (FFT) processor that is able to operate at supply voltages as low as 180mV using a standard 0.18μm CMOS logic process. The design of the DSP includes discussion of the implications and issues that arise with scaling to subthreshold levels as low as 100mV. The DSP contains dedicated datapath and memory hardware to perform the FFT and is able to scale between 128 to 1024-point FFT lengths and scale between 8 and 16 bit precision FFTs. The FFT implemented in this thesis is designed to be used in a variety of sensor applications for microsensors.

## 1.1 Microsensor applications

Advances in CMOS technology, high demand for wireless portable devices, and the increasing needs of real-world sensing applications, is fueling much interest in the field of wireless sensor networks. In modern day military and civil applications, large sensor arrays are used to extend the field of view of the end-user. In military applications, radar, sonar, and infrared sensor arrays enable surveillance of battlefields. Commercial arrays are found in traffic and environmental monitoring. Sensing systems usually consist of a few large, expensive, and highly sensitive macrosensors. A large interconnect infrastructure is needed to collect sensor data and power-up the sensors. Macrosensor systems are not robust, as one faulty sensor causes the entire system to fail.

Networks of wireless sensor nodes are replacing macrosensor systems for reasons such as cost efficiency, ease of deployment and fault tolerance. A wireless sensor network is a collection of microsensor nodes which together provide a rich view of the sensing

environment. Microsensor nodes are characterized as being low-cost, compact, and battery-operated. A microsensor node may contain a variety of sensors, including acoustic, seismic, temperature, and biological. Also, each microsensor is equipped with processing power and wireless capabilities. An ad-hoc wireless network relays all useful data about the environment back to the end-user.

In the military sector, there is a directed effort by the Department of Homeland Security to use sensor networks to provide a first-pass alarm to protect against a wide array of chemical, biological and nuclear threats [38]. Furthermore, it is envisioned that wireless sensor networks be used to monitor troop formations and for real-time battlefield surveillance. In commercial applications, networked microsensors enable a variety of applications such as warehouse inventory tracking, location sensing, machine mounted sensing, patient monitoring, and building climate control [3][18][31].

Research into wireless sensor networks yields a wide variety of interesting challenges in the fields of networking protocols, communication and computation hardware, sensor applications, architectures, and the physical layer. One key design challenge is building a multi-domain platform that is programmed for a variety of sensor applications, while still remaining cost-efficient and maintaining energy-efficiency.

The microsensor application example focused on in this thesis is the use of acoustic sensors (microphones) for environment monitoring. Acoustic sensors are highly versatile, passive, and are used in a variety of applications, such as speech recognition, traffic monitoring, and medical diagnosis. This work focuses on the application of source tracking and localization. Since acoustic sensors are passive devices, multiple microphones are needed to provide a rough line of bearing (LOB) to the sound source. When multiple microphones are deployed with high densities and overlapping coverage, then beamforming algorithms are used for detection and tracking of acoustic sources. Multiple beamformers are used to triangulate the source's sound and to track multiple targets which have different sound signatures [70].

## 1.2 Microsensor requirements

In this section, some important aspects of wireless microsensor networks are defined.

- **Low data rate:** A single microsensor node platform contains a multitude of sensors, all having low data-rates between 10 bps up to 100 kbps. For example, the acoustic sensor has a 1kHz sampling rate where each sample has 16 bit precision.
- **High node density:** The field of view of each microsensor is limited, so therefore a single microsensor yields very little information about the environment. Having high node densities (up to 10 nodes per $m^2$) is beneficial in many ways. First, multiple sensor data is aggregated using signal processing to obtain valuable inferences about the environment. Second, high node densities leads to a more robust sensor system by providing redundancy in the sensing environment. If the view of one microsensor node is obstructed or a node fails when the battery is drained, then the network adapts by using other sensors in the area to maintain system integrity. One disadvantage of high node densities is the challenge of managing a large network of sensors and processing the tremendous amount of sensing data in the network.
- **Battery-operated environment:** As the number of sensor increases or as the sensing environment widens, it becomes highly infeasible to depend on a wired interconnect infrastructure. By providing each microsensor with wireless capabilities, each microsensor node is autonomous. The lack of a wired infrastructure also implies that the wireless sensors are battery-operated and energy-constrained. As the wireless sensor size decreases, the battery capacity also decreases, thus placing a large challenge on the microsensor designers. The challenge is to operate using the energy from 1 "AA" battery for up to 5 years. This means that the average power dissipated by the microsensor is constrained be lower than 10μW.

Another possibility for microsensors is the use of ambient energy sources, such as solar, vibration, etc. There also, since the amount of generated power is low, the microsensor node is constrained to only dissipate μW's of energy [37].

- **Diverse operating scenarios:** One of the ways that the microsensor network achieves energy-efficiency is to adapt as the sensing environment or network changes. For example, as the ad-hoc wireless network adapts to changing conditions, the sensor node may play many different roles within the network. Roles that the microsensor may take on are a sensing-node that only gathers raw acoustic data, a relay-node that receives neigh-

boring data and re-transmits it to other sensors across the network, or an aggregation-node that receives multiple data from neighboring sensors, and performs signal processing on the data to make inferences about the environment. For each of these roles, the microsensor must adapt computation and communication energy, in order to optimally manage power dissipation.

Additionally, the system is event-driven. For example, to maximize battery lifetime the node remains in a deep-sleep, low-power state until an acoustic event is detected. After deep-sleep, the processor may perform low-energy/low-quality 128-point, 8-bit FFTs to do simple event detection. Once the node determines that the source is significant, then it performs high-energy/high-quality, 1024-point, 16-bit FFTs for more sophisticated detection, classification or tracking. Therefore, depending on the application, the system is designed to respond to a variety of stimuli and operates over a variety of energy-quality levels.

Since adaptability is key to achieving high energy-efficiency, we propose energy-aware design of the microsensor node.

## 1.3 Energy-aware design

Energy-aware design is the design of systems that are highly energy-efficient over a variety of operating scenarios. This is in contrast to low-power design which has a design goal to minimize energy dissipation for the worst-case scenario. Energy-aware design may not minimize energy dissipation for the worst-case, but in ensuring energy-efficiency for many scenarios energy-aware design is globally more energy-efficient.

Energy-awareness is achieved by designing energy-quality scalable "hooks" into the hardware that the end-user or the operating system uses to scale over a wide range of energy-quality points. For example, in most current microprocessors, the energy hooks available are algorithmic transformation through software manipulation and clock gating. The low-power StrongARM SA-1100 microprocessor improved its energy-awareness by allowing dynamic voltage scaling, where the supply voltage and the clock frequency of the processor is dynamically reduced or increased to adapt energy consumption as the work-

load changes [40]. These hooks are designed into all aspects of the processor beginning from the algorithms all the way to the circuits.

Because the microsensor system is highly energy-constrained, it becomes evident that application specific solutions are needed for the microsensor rather than implementations on general purpose hardware. Application specific solutions dissipate up to 100 times less energy than the equivalent microprocessor or FPGA based solutions. The main reason is that application specific energy-quality hardware are tailored for microsensor functions and application specific hooks are chosen that fit the diverse scenarios that the node encounters. For example, the FFT processor for sensors requires variable-length FFTs and variable bit-precision FFT operation.

In order to achieve minimal energy operation, the supply voltage and clock frequency are scaled down until the optimal operating point is achieved. Subthreshold logic and memory circuit techniques are needed to achieve operations at ultra-low voltage levels.

## 1.4 Contributions of this thesis

In this thesis, the FFT processor is a design driver to showcase energy-awareness in DSPs. Energy-aware techniques for systems, architectures, and circuits, through three separate FFT implementations are discussed and analyzed.

- **Energy-aware Systems:** In a microsensor system, signal processing is performed locally on an array of microsensors. By using distributed network driven voltage and frequency scheduling, large energy savings is achieved. First, the need for local signal processing for global energy-efficiency is shown for a cluster of microsensors and a remote basestation. Then, distribution of computation among neighboring sensors is analyzed. By optimally scheduling voltage supplies and clock frequencies of different sensor nodes achieves global energy-efficiency. These system-level partitioning concepts are demonstrating using an Intel StrongARM sensor node platform developed by the MIT μAMPS project. The minimal energy dissipated by the StrongARM is 48μJ/FFT for a 512-point FFT at 0.85V supply voltage and with clock frequency of 74 MHz.
- **Energy-aware Architectures:** Energy-awareness at the architecture level is needed to enable energy-aware algorithm implementation. Energy-awareness in a FFT

implementation includes hooks for variable-length FFTs and for variable bit-precision. This thesis analyzes and evaluates different energy-aware architectures to enable energy-awareness and verifies the architectures with a scalable FFT implemented in a 0.18μm process. The FFT ASIC is able to scale between 128 to 1024-point FFT lengths and 8 and 16 bit precisions with little energy and area overhead. The chip is built using an ASIC flow which uses a 1.8V standard cell library and memory generators and is able to operate down to 1V.

- **Energy-aware Circuits:** Research into the latency requirements of the FFT for sensors shows that the clock frequencies required for the FFT are much lower than the clock frequencies required by typical wireless applications, such as cellular communication. Focus of the research now shifts to subthreshold circuit design of the FFT where the supply voltage is dropped below the threshold voltage of the circuits. Since at low clock frequencies, leakage energy dissipation exceeds active energy dissipation, there is an optimal operating frequency and supply voltage for minimal energy consumption.

First, simulations of a variable activity factor ring oscillator benchmark circuit are used to evaluate the optimal supply voltage and threshold voltage operating regime in the subthreshold region. Second, the FFT is used as a design platform showcasing low power circuit styles and logic families for memories and combinational logic functional for subthreshold power supplies. The fabricated FFT processor operates down to 180mV using a standard CMOS 0.18μm logic process. At the minimal energy operating point for the 1024-point, 16-bit RVFFT at 350mV supply voltage and 9.62kHz clock frequency, the FFT processor dissipates 155nJ/FFT. This FFT processor is the first system level demonstration of subthreshold circuit operation at supply voltages as low as 180mV.

## 1.5 Overview

This dissertation is an exploration of system-, architectural- and circuit-level considerations of energy-awareness. Many concepts described here can be applied toward general systems, where the key metric is energy-efficient computation and other traditional metrics such as clock frequency/performance, latency and silicon area are be relaxed. All of the concepts are demonstrated for a source-tracking application for acoustic sensors with particular emphasis on the FFT.

# Chapter 2

# Sensor signal processing

Signal processing algorithms and energy-efficient digital signal processors (DSPs) are leveraged to achieve highly energy efficient sensor systems. They allow the sensor network to aggregate data and extend the overall field of view. In a typical sensor system, all data is transmitted to the end-user, where the signal processing is performed. As the total number of microsensors in the network increases, it becomes increasingly difficult to store and transmit all of the data to the end-user. Usually it is not necessary to transmit all of the raw sensor data in a sensor network. Fortunately, it is not the individual nodes' data that is important to the end-user, but the combined knowledge that reliably describes the environment the nodes are sensing. Therefore, by having a DSP on the sensor node, signal processing is performed locally to extract important information from the sensor data and to reduce redundant data in the network. By expending a small amount of energy to do computation locally, a large amount of communication energy is saved [77].

In traditional algorithm design, the design goal is to optimize for the worst case scenario (e.g. minimal power dissipation, minimum latency, maximum accuracy, etc.). In an energy-aware system, where the system operates in a variety of scenarios, design for only the worst case may not be globally optimal. Energy-awareness at the algorithm level is highly desirable because a large range of both energy and quality is achieved by varying algorithm parameters. In sensor networks, high quality refers to metrics such as high speed, low latency, or highly accurate results. Scalable-energy algorithms are highly important for microsensors because the node's resources and environment change dramatically over the system lifetime. For example, when energy is plentiful, the network may be required to produce high quality results. As the energy gets depleted, the network reduces the quality of the results and reduce energy dissipation. Being able to adapt to changing conditions leads to lengthening of the total system lifetime.

Two algorithms that are used for sensor detection and estimation are Line of Bearing (LOB) estimation and Harmonic Line Association (HLA). Line of Bearing estimation triangulates the data from an array of multiple microphones to pinpoint a sound source's location. In HLA, the frequency spectrum of the source is analyzed to determine multiple sources' signatures and to prevent false detections. The two algorithms are used for source tracking of multiple sources.

The functional unit that is used in both the LOB estimation and HLA is the Fast Fourier Transform (FFT). The FFT is a widely used algorithm used to transform signals from the time-domain to the frequency-domain. This chapter includes a brief description of the LOB estimation and HLA algorithms and a detailed explanation of the FFT algorithm.

## 2.1 Previous work

Previous work in energy-scalable algorithms focuses mainly on the idea of incremental refinement [45]. Incremental refinement algorithms allow the user to make trade-offs between computational accuracy and computing resources. The use of incremental refinement is used in the area of approximate signal processing, namely in signal detection using the fast Fourier transform (FFT) and image decoding using the two-dimensional inverse discrete cosine transform (2D IDCT). In these cases, the energy-scalable hook is the processor run-time. Energy is saved by reducing the processor run-time by stopping the program early and outputting a less accurate result. The algorithm implementation provides both high-quality results when running at full utilization, and low-quality results when stopped early.

In literature, there are various implementations of low power and energy-efficient FIR filters [61]. FIR filtering is a commonly used algorithm in sensor signal processing. Approximate processing techniques are proposed for the FIR filter to reduce the total switched capacitance by dynamically varying the filter order based on signal statistics [45]. In wireless sensor networks, algorithmic transformations in software was used to improve the scalability of many algorithms, such as FIR filtering, DCT, and beamforming [62].

## 2.2 Energy-quality scalability

Energy-scalability at the algorithm level is highly desirable because a large range of energy and quality is achieved by varying algorithm parameters and protocol operation. The Energy-Quality (E-Q) graph of an algorithm is the function $Q$(E), representing some quality metric (e.q. mean-square error, peak signal-to-noise ratio etc.) as a function of the computational energy $0 \le E \le E_{\max}$. Using simple modifications, the E-Q behavior of the algorithm is engineered such that it has two desirable traits [62].

Consider two implementations of the same algorithm (I and II). From an energy perspective, II is a more efficient scalable system compared to I if

$$Q_{II}(E) > Q_I(E) \quad \forall E \tag{2.1}$$

In most practical cases, Eq. 2.1 does not hold over all energy values. For example, there might be a preprocessing overhead as a result of which the maximum energy consumptions might be different for the two cases (i.e. $E_{\max,\ II} > E_{\max,\ I}$). Nevertheless, as long as Eq. 2.1 holds over a significant range of computational energies, overall efficiency is assured. Assume that there exists a quality distribution $p_Q$(x). $p_Q$(x) is the probability that the end-user desires quality $x$ and is found from signal statistics. A typical quality distribution is shown in Figure 2.1. The average energy consumption per output sample is expressed as

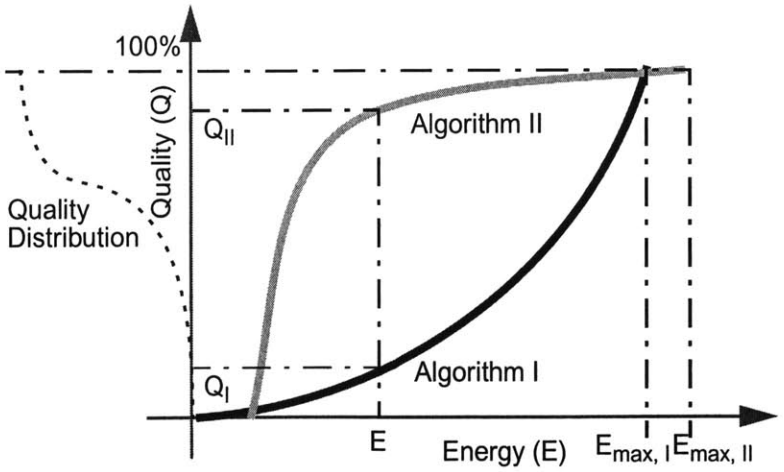


**Figure 2.1: E-Q formal notions. The implementation of Algorithm II is superior to that of Algorithm I because Algorithm II has a more desirable E-Q curve.**

$$\bar{E} = \int p_Q(x)E(x)dx \tag{2.2}$$

where $E$(Q) is the inverse of $Q$(E). When the quality distribution is unknown, it is desired that the E-Q behavior to be maximally concave downwards (with respect to the energy axis), i.e.

$$\frac{\partial^2 Q(E)}{\partial E^2} \leq 0 \tag{2.3}$$

Note that Eq. 2.3 is not always attainable globally across $0 \leq E \leq E_{max}$. However, on an average case, for a given energy availability $E$, we would like the obtainable quality $Q(E)$ to be as high as possible. Therefore, energy-aware design focuses on implementing algorithms that maximize quality for a given energy constraint or minimize energy for a given quality constraint.

## 2.3 Line of Bearing application

Suppose a vehicle is moving over a region where a network of acoustic sensing nodes is deployed (Figure 2.2). The data from a cluster of multiple sensors is used to determine the Line of Bearing (LOB) of the source. The LOB is calculated to be the direction with maximum sound energy detected. In a sensor system with multiple clusters, each cluster calculates a LOB to the source. The LOB estimates are triangulated so that the intersection

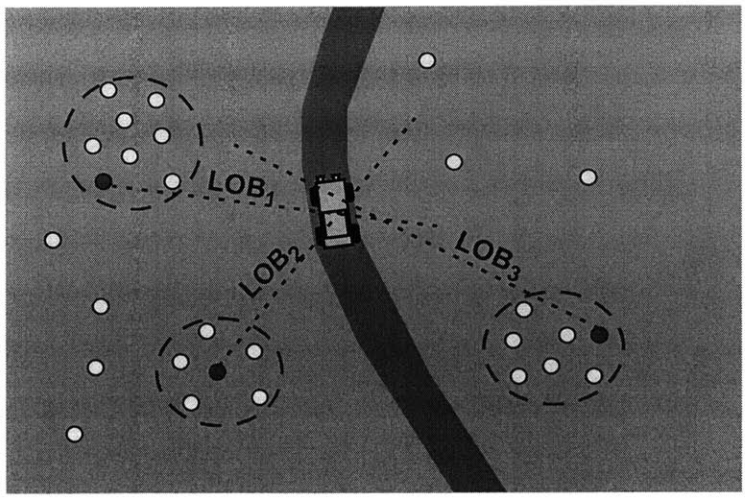


**Figure 2.2: LOB estimates from multiple clusters are triangulated to find the source's location.**

point is the source's location.

There is body of research dedicated to source tracking using multiple sensors. Researchers are studying source tracking for sonar, radar, acoustic, and seismic sensors. The most common algorithms for LOB estimation are beamforming algorithms. A beamformer is a spatial filter which operates on multiple sensor data. By pointing the beam of the filter in the direction of the sound source, the source signal is coherently amplified while ambient noise from all other directions are diminished. The source signal detected has better signal-to-noise ratio (SNR) if the beam is pointed in the direction of the source. Beamforming is also used to determine the direction of arrival (DOA) of the sound source. By pointing the beam in multiple directions, the direction with the most energy relative to the orientation of the sensor array is the DOA of sound. This direction is directly related to the line of bearing (LOB) of the source signal.

Delay-and-sum beamforming is the simplest and most commonly used beamforming algorithm. This algorithm applies delays on the sensor data from multiple sensors before summing over all sensors [42]. Delay-and-sum beamforming is done in the time-domain, frequency-domain or a hybrid of the two. In the time-domain the beam output in a given direction is given by

$$b(t, \theta) = \sum_{m=1}^{M} A s_m(t - \tau_m(\theta)) \tag{2.4}$$

where M denotes the number of sensors, $s_m(t)$ is the data from sensor $m$, $A$ is the constant gain, and $\tau_m$ is the time delays for steering the beam in the desired direction, $\theta$. The correct delays for a given direction of arrival are calculated from the array geometry and by knowing the speed of the wavefront.

One drawback of implementing the time-domain delay-and-sum beamforming algorithm in the digital domain is that it is necessary to oversample the audio signal before beamforming. This is needed in order to get the fine-grained delays necessary for good beamformer performance. An acoustic sensor output which is nominally sampled at a frequency of 1kHz only resolves delays greater than 1 millisecond. By oversampling the signal at 5-10kHz, the beamformer resolves fine-grain delays. This leads to high

computational demands, depending on desired beamformer performance. Another way to achieve high accuracy results is to interpolate the sensor data or to convolve the sampled data with interpolated sinc functions using an FIR filter [42].

Delay-and-sum beamforming is also performed in the frequency-domain. Delays in the time-domain corresponds to phase shifts in the Fourier or frequency-domain. Due to the linear properties of the Fourier transform, the beamformer output in the Fourier domain is given by

$$B(f,\theta) = F\{b(t,\theta)\} = \sum_{m=1}^{M} AS_m(f)e^{-j2\pi f\tau_m(\theta)} \tag{2.5}$$

where $S_m(f)$ is the Fourier transform of the $m$-th sensor output [56]. In the digital domain, the Discrete Fourier Transform or the Fast Fourier Transform (FFT) is used [50]. The main advantage of the frequency-domain is that there is no need for oversampling the sensor output, as compared to the time-domain approach. Thus, the computational complexity of the frequency-domain FFT is related to the size of the FFT.

Another advantage of the frequency-domain approach is that the amount of transmitted data is reduced if the signal of interest has a limited bandwidth. For our acoustic sensor application, the signal of interest has a bandwidth of

$$20\text{Hz} \le f_{\text{source}} \le 256\text{Hz}. \tag{2.6}$$

Only those FFT coefficients in the bandwidth of interest are used in the beamformer computation, which is equivalent to applying a bandpass filter to the data. The beamformer accuracy is improved by reducing the effects of the high and low bandwidth noise.

There are other beamforming algorithms found in literature. The Minimum Variance Distortionless Response (MVDR) beamformer computes the LOB using the maximum likelihood spectrum [24]. The MVDR and the delay-and-sum beamformer both assume that the sensor configuration is known. There are other beamforming algorithms that do not assume a known sensor configuration. Instead, these algorithms use subspace tracking concepts and extract the direction of arrival from the signal subspace common to the multiple sensors. One such algorithm is the Maximum Power Beamforming algorithm [81].

This algorithm is highly complex because it requires matrix multiplication and eigenvector decomposition [43]. Other time-domain algorithms are often used which adaptively update the state of the beamformer, such as the Least Mean Square (LMS) algorithm. The LMS algorithm requires knowledge of the source's time signature [24].

After processing the sensor data using beamforming algorithms, the beamformer outputs are then fed into a LOB estimator. The LOB estimator takes the beamformer output and calculates the signal energy for each direction. The maximum weighted average of signal energy over all directions provides a simple LOB estimate for the signal.

## 2.4 Harmonic Line Association

Harmonic Line Association (HLA) is performed on an acoustic signal to differentiate between different sources and to classify sources. HLA assumes that the desired signal has a strong harmonic signature. Therefore, source's frequency content is characterized by its dominant harmonic and multiples of the harmonic. For example, the harmonics are generated by a vehicular source's running engine or propulsion gear (tire or tracks). The rotation of the engine and propulsion creates stationary and periodic signals [66].

The first step in HLA is to transform the acoustic signal from the time-domain to the frequency-domain through a real-valued Fast Fourier Transform. The algorithm picks out the maximum peak of the frequency spectrum and assumes it to be the $k$th harmonic line of the fundamental frequency. The fundamental frequency is typically bandlimited and in the source-tracking application is assumed to be between 8-20Hz. The magnitudes of the other corresponding harmonics are extracted and are used to calculate the strength of the HLA set [77].

Based on the strengths of multiple HLA sets, the number of sources is estimated. A threshold for HLA strength is chosen based on the user's tolerance for false detections and false alarms. Additionally, the HLA set is as a feature set for automatic target classification or fused with a seismic data for better target classification results [77]. The value and amplitude of the fundamental frequency is highly dependent on many factors such as engine design, vehicle speed and the surrounding environment. Therefore, tracking over several timesteps is needed to differentiate between the invariant characteristics for each

source and the surrounding noise. Tracking is performed by using intelligent pattern recognition or neural networks [66].

The HLA is used jointly with the LOB estimation to perform source-localization and tracking of multiple targets.

## 2.5 Real-valued Fast Fourier Transform

In both LOB estimation and the HLA application the waveform is transformed from the time-domain to the frequency-domain. The transform is performed using the Fast Fourier Transform (FFT).

The FFT is derived from the continuous-time Fourier Transform. The continuous-time Fourier transform is used in signal processing to transform a continuous-time waveform from the time-domain to the frequency-domain. The continuous Fourier Transform is rarely used because it operates on infinitely long waveforms and produces an infinite length frequency waveform.

For digital systems, often an assumption is made that the waveform has a limited frequency band of interest. The input signal is low-pass filtered and then sampled. In our sensor application, the sensor data is sampled by a 12-bit 1kHz A/D. Since the signal of interest is below 250 Hz, the 1kHz sampling rate is adequate for our application.

The Discrete Fourier Transform (DFT) is the Fourier Transform applied for discrete time samples. The DFT's outputs are Fourier coefficients at discrete frequency bands. In 1965, Cooley and Tukey described a fast and efficient way to compute the DFT. By exploiting the periodicity of the DFT, the "fast Fourier transform" was able to compute one DFT in $N/2*\log_2(N)$ time versus the previous methods which computed in $N^2$ time [12].

The complex-valued FFT (CVFFT) of a sequence $s(n)$ is defined to be

$$S(k) = \sum_{n=0}^{N-1} s(n) W_N^{kn} \qquad \text{k=0,1,... N-1} \tag{2.7}$$

where $W_N^{kn} = e^{-j2\pi kn/N}$. A typical assumption is that $s(n)$ and $S(k)$ are both complex sequences. However, in the case of acoustic sensors, the incoming data from the sensors is

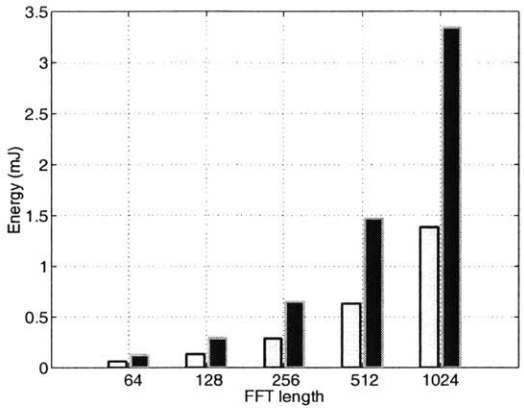


**Figure 2.3: Energy dissipation of the StrongARM as a function of FFT length for the CVFFT vs. RVFFT shows the RVFFT is 2.5x more energy-efficient.**

real. This means that the imaginary half of the inputs to the CVFFT are zero. Similarly half of the outputs of the CVFFT are redundant due to the symmetrical properties of $S$(k). The symmetrical properties of the FFT is exploited to perform an N-point real-valued FFT (RVFFT) by only using a N/2-point CVFFT. Profiling on the StrongARM shows a 2-2.5x improvement in energy-efficiency for the N-point RVFFT over the N-point CVFFT (see Figure 2.3).

The RVFFT algorithm is derived from the CVFFT algorithm. First, the N-point CVFFT of the real-valued sequence s(n) is broken up into two N/2-point FFTs

$$S(k) = \sum_{n=0}^{N/2-1} s(2n) W_{N/2}^{kn} + W_N^{kn} \sum_{n=0}^{N/2-1} s(2n+1) W_{N/2}^{kn} \qquad \text{k=0,1,...,N-1} \qquad (2.8)$$

where now it is represented by the N/2-point FFT of the even indexed inputs and the N/2-point FFT of the odd indexed inputs multiplied by the factor $W_N^{kn}$. Next let $x$(n)=$s$(2n) and $y$(n)=$s$(2n+1) so that S(k) is represented as

$$S(k) = X(k) + W_N^{kn} Y(k) \qquad (2.9)$$

The N/2 CVFFT of the N/2 length signal $z$(n)=$x$(n)+j*$y$(n) for n=0,1,...,N/2-1 is taken. The N/2-point FFT of $z$(n) is defined to be:

$$Z(k) = X(k) + jY(k) = X_r(k) + Y_i(k) + j[X_i(k) + Y_r(k)] \tag{2.10}$$

To recover $X$(k) and $Y$(k) from $Z$(k), the following symmetrical properties for a real-valued signal, $p$(n), is used

$$p(\text{n}) \text{ is real if and only if } P(\text{k}) = P^*(\text{N-k}) \tag{2.11}$$

Using this identity, *X(k)* and *Y(k)*, which are FFTs of real-valued signals are re-written as

$$X(k) = \frac{1}{2}\{Z_r(k) + Z_r(N-k)\} + j \cdot \frac{1}{2}\{Z_i(k) - Z_i(N-k)\} \tag{2.12}$$

$$Y(k) = \frac{1}{2}\{Z_i(k) + Z_i(N-k)\} + j \cdot \frac{1}{2}\{Z_r(k) - Z_r(N-k)\} \tag{2.13}$$

Therefore by taking the N/2 CVFFT of $z$(n), $X$(k) and $Y$(k) are calculated and then substituted into Eq. 2.9 to obtain $S$(k).

From this analysis, the fundamental building block of both the RVFFT and the CVFFT is the radix-2 butterfly. The radix-2 butterfly performs one complex multiplication between the data value and a twiddle factor, and then adds/subtracts the product to another data value. More details about the radix-2 butterfly is found in Chapter 4. All radix-2 butterflies can be displayed in a trellis diagram that describes the flow of computation.

Figure 2.4 shows the butterfly diagram for the 32-point RVFFT based on the equations previously mentioned. The first 4 stages are the traditional CVFFT butterflies and the last stage of butterflies performs the backend processing to calculate the complex FFT coefficients. The number of butterflies performed for an N-point RVFFT is (N/4)*logN. In this thesis, an N-point FFT notation will represent either a N-point RVFFT or a N/2 point CVFFT.

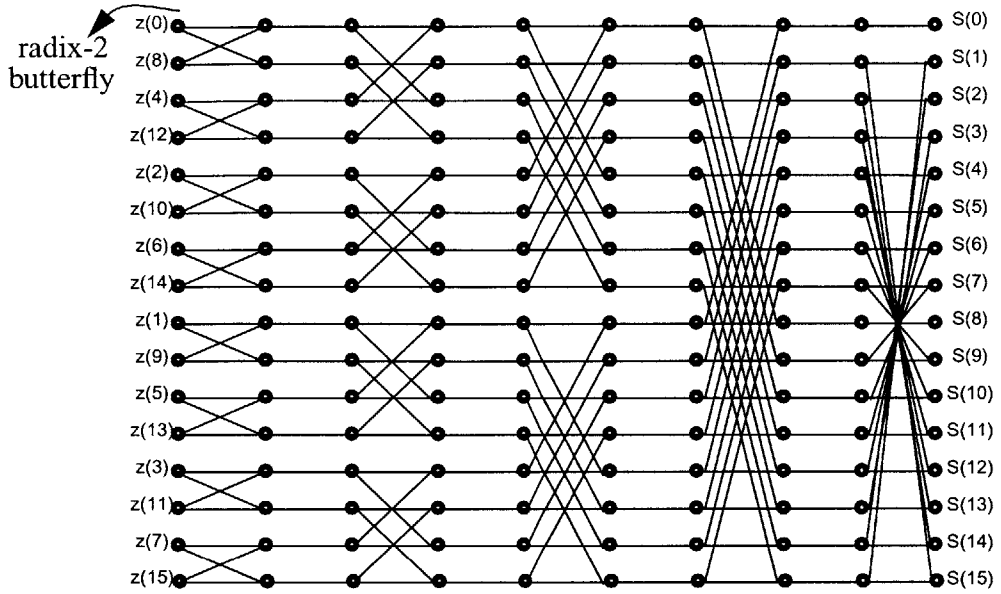


**Figure 2.4: Butterfly diagram of a 32-point RVFFT.**

## 2.6 Summary

As the number of sensors in the network increases, it is more energy-efficient to perform local signal processing. The end-user does not require the raw sensor data, but a function of the data which is less bandwidth. By performing local signal processing, the large costs associated with communication are reduced at the relatively small cost of computational energy expended by each sensor node.

The concept of energy-awareness in algorithms is proposed. A large range of energy-quality scalability is achieved at the algorithm level. Algorithms that are focused on in this thesis are the Line of Bearing estimation algorithm and the Harmonic Line Association algorithm. Both are used in a source-tracking and localization application.

The LOB and HLA algorithms have a common block which is the FFT. For sensor algorithms, the real-valued FFT is used and up to 2.5x energy savings achieved over a complex-valued FFT implementation.

# Chapter 3

# Energy-aware system partitioning

This chapter looks into the role of signal processing in the sensor system, from the entire sensor node architecture all the way to network protocols. A node architecture is being designed as part of the micro-Adaptive Multi-domain Power-aware Sensors project at MIT (μAMPS) [39]. An energy-model is given for the Intel StrongARM low-power microprocessor which is the computational engine of the node. System level research focuses on energy-aware system partitioning of computation within the network to improve energy efficiency.

## 3.1 The MIT μAMPS microsensor node

An energy-efficient node architecture is being developed as part of the low power wireless sensor project at MIT (μAMPS). The first version of the μAMPS node is implemented with various off-the shelf components to verify the microsensor node platform and to demonstrate concepts of energy-awareness. Figure 3.1 shows the architectural overview of the sensor node, which contains four basic modules: sensor, communication, power, and

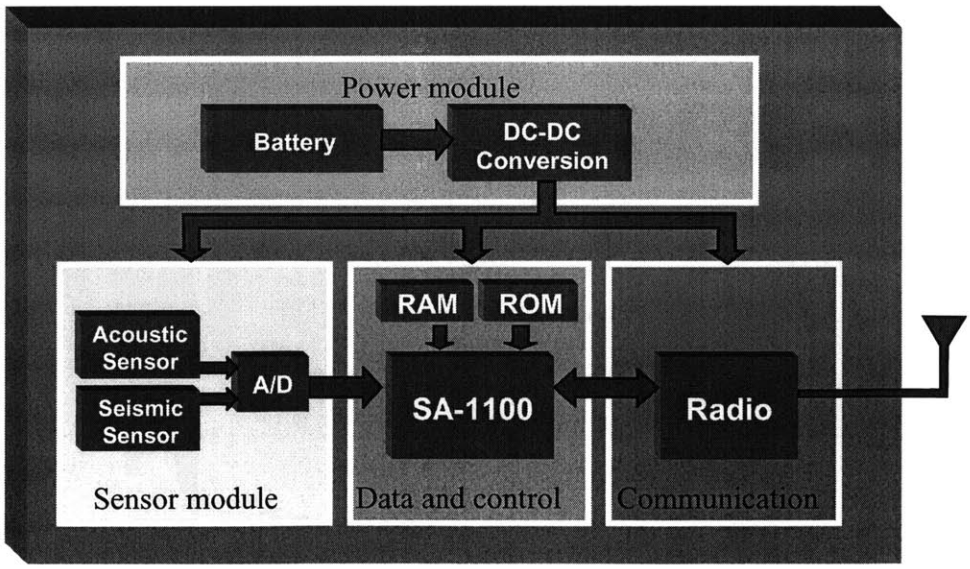


**Figure 3.1: Architectural overview of μAMPS sensor node.**

data and control processing.

### 3.1.1 The sensor module

The sensor module has ports for various sensors including acoustic and seismic sensors and contains an analog-to-digital (A/D) converter. The acoustic sensor is used for the application of source tracking and classification. The data is continuously sampled, and stored in on-board RAM to be processed by the data and control processing module.

### 3.1.2 The communication module

The communication module or radio is used by the sensor node to collaborate with neighboring sensors and with the end-user. The primary component of the radio is a commercial transceiver optimized for ISM 2.45 GHz wireless systems. The communication module is capable of transmitting up to 1 Mbps at a range of up to 10 meters [44].

The radio energy dissipation is dominated by two components: the energy dissipated to run the transmit or receive electronics which is given by $E_{elec}$ (J/bit) and the energy dissipated by the power amplifier in the transmit path which is given by $\varepsilon_{amp}$ (J/bit/m$^2$). We also assume an $d^2$ energy loss for transmission between sensors since the distances between sensors are relatively short [19]. To transmit a $k$-bit packet a distance, $d$, the energy dissipated is

$$E_{tx}(k, d) = E_{elec} \cdot k + \varepsilon_{amp} \cdot k \cdot d^2 \tag{3.1}$$

and to receive the $k$-bit packet, the radio expends

$$E_{rx}(k) = E_{elec} \cdot k \tag{3.2}$$

For our radio, $E_{elec}$ = 50 nJ/bit and $\varepsilon_{amp}$=100 pJ/bit/m$^2$ [26].

### 3.1.3 The power module

The third module is the power module which supplies the variable power for the node. The power for the node is supplied by a single 3.6V DC source, which is provided by a single lithium-ion cell. Regulators are used to generate 5V, 3.3V, and an adjustable 0.9 to

1.6V supply from the battery. The 5V supply powers the analog sensor circuitry and analog-to-digital (A/D) converter. The 3.3V supply powers all digital components on the sensor node with the exception of the processor core. The StrongARM SA-1100 core is powered by a digitally adjustable switching regulator that provides 0.9V to 1.6V in twenty discrete increments. Having a digitally adjustable voltage supply allows the SA-1100 to control its own core voltage and enables dynamic voltage scaling (DVS) techniques [23].

### 3.1.4 The data and control module

The fourth module is the data and control processing module. This module controls the settings to the other modules and performs all of the signal processing for the sensor node. Also, this module is responsible for controlling the energy-awareness of the entire node. For the source tracking application, the sensor node is one of three roles: sensing, aggregation, and relay. If the node is a sensing-node, then the sensor data is passed to from the sensor module directly to the communication module and is transmitted. At a aggregation-node, sensor signal processing is done. The aggregation-node collects sensor data from neighboring sensors and performs the LOB estimation or HLA algorithm. Then the aggregation-node will transmit the signal processing results. A relay-node receives data from neighboring sensors and transmits the raw sensor data to neighboring sensors or the end-user.

The central component of the data and control processing subsystem is the StrongARM SA-1100 microprocessor. The SA-1100 is suitable for use in a sensor node due to its low power consumption, sufficient performance for signal processing algorithms, and static CMOS design. In addition, the SA-1100 can be programmed to run at a range of clock speeds from 79-206 MHz and at voltage supplies from 0.85-1.5V [1]. On-board ROM and RAM are included for storage of sampled and processed data, signal processing tasks, and the operating system.

A simple energy model is needed to model the active energy dissipation of the SA-1100 as a function of supply voltage.

$$E_{\text{comp}} = NCV_{\text{DD}}^2 \tag{3.3}$$

where $N$ is the number of clock cycles per task, $C$ is the average capacitance switched per cycle, and $V_{DD}$ is the supply voltage [10]. For the StrongARM SA-1100, experiments show that $C$ is approximately 0.67 nF [60]. Another component of processor energy dissipated is the leakage energy, which is caused by subthreshold leakage currents between power and ground.

$$I_{\text{leak}} = Ke^{\left(\frac{V_{gs} - V_{th}}{nV_T}\right)}\left[1 - e^{-\frac{V_{ds}}{V_T}}\right] \tag{3.4}$$

where $K$ is a function of technology, $V_T$ is the thermal voltage, $V_{th}$ is the threshold voltage, and $n$ is related to the subthreshold slope. Most processor energy models only consider switching energy, but in low duty cycle systems, leakage energy dissipation is significant [60].

It is also important to model the clock speed of the SA-1100 as a function of $V_{DD}$.

$$f \leq \frac{K(V_{\text{DD}} - V_{th})^{\alpha}}{V_{\text{DD}}} \approx K(V_{\text{DD}} - c) \tag{3.5}$$

where $\alpha$, $K$, $c$, and $V_{th}$, are processor dependent variables. For the SA-1100, the frequency-voltage relation is linearized to simplify the calculations without significant error being introduced into the model. Measurements show that $K = 239.28$ MHz/V and $c$=0.5V.

DVS is one energy-scalable "hook" available on the StrongARM SA-1100. It is used to create an energy-scalable sensor node that is able to adapt energy dissipation in response to changing operating conditions or a variable computational load. For example, assume in a fixed throughput system ($T$), the processor computational energy for task A is given by $E_{\text{fixed}}$, but is completed in half the allotted time, $T/2$. This is non-optimal since after the processor completes the task, it idles for $T/2$ seconds. By reducing the clock frequency by half, the processor is active for the entire $T$ seconds. A lower frequency implies that the voltage supply is reduced approximately by half. According to Eq. 3.3, energy is related to voltage supply squared, so only one-fourth the energy is dissipated.

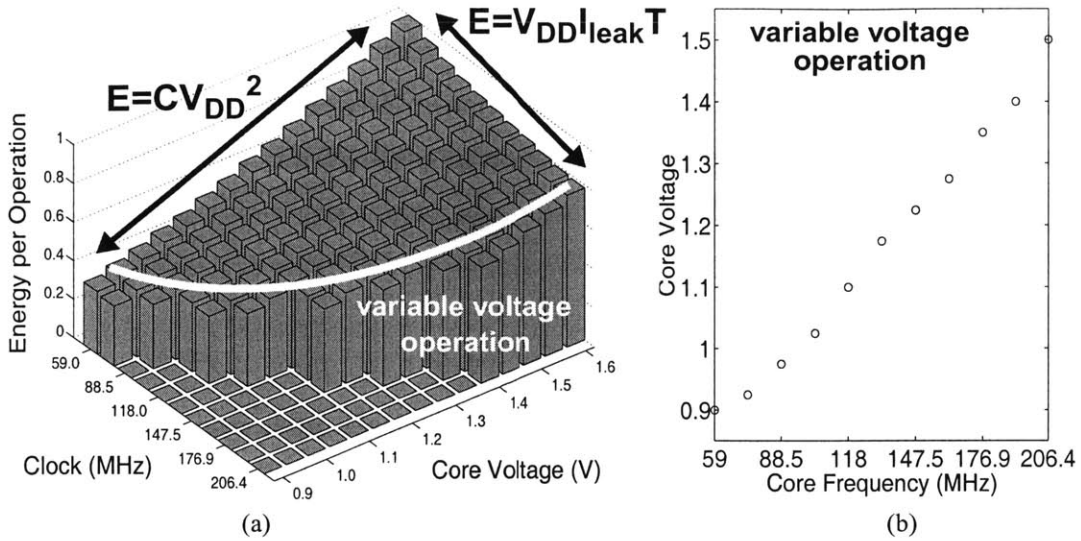


**Figure 3.2: (a) Measured energy consumption characteristics of SA-1100. (b) Operating voltage and frequency pairs of the SA-1100.**

Figure 3.2a depicts the measured energy consumption of a SA-1100 processor running at full utilization [40]. Energy consumed per operation is plotted with respect to the processor frequency and voltage. For a fixed clock frequency, the energy per operation has a quadratic dependence on supply voltage. For a fixed voltage supply, the total energy per operation increases due to the leakage energy. Leakage energy increases with lower clock frequencies because operations occur over a longer clock period. This graph shows that minimum energy dissipation occurs when operating at the lowest possible voltage supply level for a given frequency. Figure 3.2b shows all eleven frequency-voltage pairs for the StrongARM SA-1100 for energy-efficient operation.

Analysis of the power dissipation for the entire StrongARM SA-1110 based sensor node indicates that the processor board takes 48% of the power, the radio takes up 45% of the power and the sensor only 6%. Since all computation is expected to be done on the processor board, it is important that the computation engine be energy-aware.

## 3.2 Energy-aware system partitioning

There are many energy-aware techniques that are used at the system level to improve energy efficiency. In this section, two system partitioning techniques are explored. The first is partitioning computation between the end-user and the sensor cluster. As the dis-

tance between the end-user and sensing environment increases, it is more energy-efficient to do the computation locally within the sensor cluster. This takes advantage of communication vs. computation trade-offs in energy dissipation by reducing the amount of data transmitted. The second system partitioning technique is distributing the computation among the sensors in the cluster. Intelligent system partitioning with DVS yields further energy savings.

### 3.2.1 Previous Work

Past research has shown that system partitioning across a network is used in many applications to reduce energy dissipation. In a wireless system, the basestation has essentially unlimited energy and the mobile is battery-operated. Energy savings is achieved by appropriately partitioning the computation. In the Infopad project [10], system partitioning is considered between portable mobile devices and high powered network servers. The computation (text/graphics functions) is partitioned such that the portable terminal only performs the display updates, and not high-powered (X-server) functions. This did increase the communication energy, but the compuational energy savings for the portable is significantly lowered. Also, partitioning is applied to motion estimation in wireless video terminals. Motion estimation is highly computational and is a large part of the system's computational energy dissipation. However, it is necessary to perform image encoding for compression, because compression reduces the large amount of energy expended for communication. An energy-efficient scheme is to reduce the portable's energy dissipation by performing motion estimation computation on previous frames at the receiving basestation. Then a low-bandwidth reverse wireless link transmits the results back to the portable to do the compression on later frames. This system achieves nearly the same compression rates as the original motion estimation technique, but performs over two orders of magnitude less computation at the portable [54]. For sensors, similar communication vs. computation trade-offs are made between the sensors and the end-user.

System partitioning is also used within the sensor cluster. A sensor cluster is essentially a system of multiple microprocessors linked through a short distance wireless link, and therefore is considered a parallel architecture. Parallel architectures have the same performance as a single architecture by reducing clock rates and voltage supplies. Increas-

ing parallelism is a common technique used in circuit design at the architectural level to reduce energy dissipation when there is a fixed latency. One example is to use two functional units in the datapath, versus one unit. By adding one more functional unit, each unit is allowed to run at half the original rate for the same throughput. Since the clock frequency is decreased, the voltage supply is dropped by half, and the energy is reduced by almost 4 times over the non-parallel case. For a simple adder-comparator 8-bit datapath, it the power is reduced by approximately 2.5 times [10]. However, this does come at an increase of area and overhead control hardware.

System level distributed signal processing has been researched and compared for traditional DSP computation platforms. Distributed computing systems with networks of DSPs are studied extensively due to their potential for handling a wide range of computational demands, and for being cost-effective [65]. "Optimal" partitioning of processing tasks into subtasks and matching of subtasks onto processors has also been done assuming general purpose processors [7]. For example, algorithm adaptation is proposed to react to changes in scheduling of computation around a network. Using this method for a Synthetic Aperture Radar technique, processing delay is reduced while increasing system capacity [34]. However, for multiple microprocessor systems, the metric is performance while for sensor networks, energy is the preferred metric.

### 3.2.2 Energy-efficient system partitioning between end-users and sensors

Examples of networking protocols are direct communication, multi-hop routing, and clustering. In direct communication, as shown in Figure 3.3(a), all microsensors are sensing-nodes and transmit the raw sensor data results directly to the basestation. In multi-hop routing, sensors act as relay-nodes for other sensors' data in addition to sensing the environment, as shown in Figure 3.3(b). Multi-hop routing minimizes the distance an individual sensor must transmit its data, and hence minimizes the dissipated energy for that sensor. However, this approach is not globally energy-efficient. An energy-efficient networking protocol organizes sensors into local clusters, as shown in Figure 3.3(c). Within each cluster is a *clusterhead* or aggregation-node. Because the aggregation-node performs data aggregation and only transmits the results to the end-user, there is a great reduction in

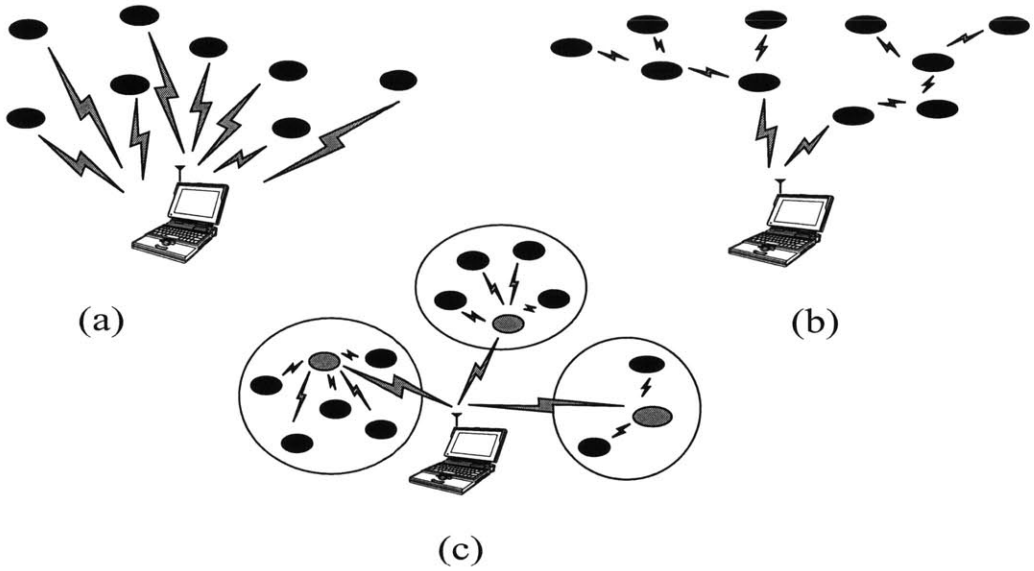


**Figure 3.3: Examples of microsensor networks. (a) Direct communication with end-user. (b) Multi-hop routing communication with end-user (c) Clustering algorithm. The grey nodes represent clusterheads.**

the amount of data that is sent to the end-user and thus achieves energy-efficiency.

In a typical sensor system where there is no energy-constraint, a direct communication protocol is employed. The data is sent via wires from the sensors to the end-user to be processed and the same wires are also used to power the sensors. For a wireless sensor network where energy is highly constrained, direct communication between sensors and the end-user is both cumbersome and energy-intensive. The end-user is usually far away from the sensing area causing communication to the end-user to be very costly. This is true since the energy increases proportionally to the square of distance. Also, as the number of sensors in a network grows larger, it becomes difficult to manage the large amount of data collected from the sensors. And with increased node densities in one location, multiple sensors may view the same event causing a large amount of redundant sensor data. For all of these reasons, direct communication protocols are not used in wireless sensor networks.

An energy-efficient wireless networking protocol for sensor networks is able to reduce energy dissipation through sensor collaboration [27]. Sensor collaboration means that the sensors are able to communicate locally and share information. Since closely located sensors tend to have highly correlated data, sensor collaboration allows for signal processing to reduce redundant information. Also there is a trade-off between communication and

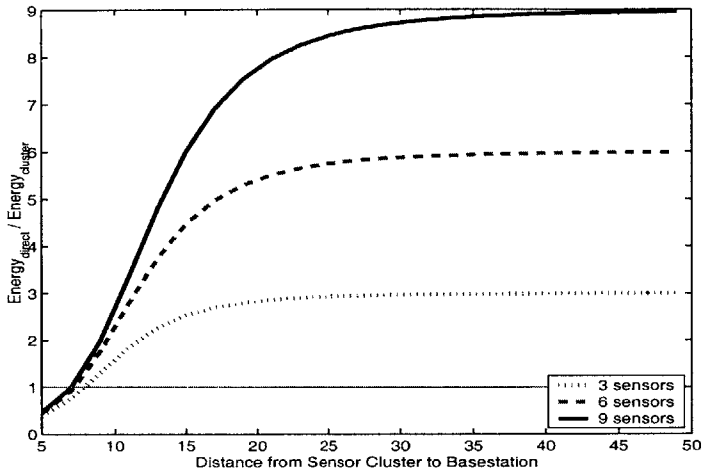


**Figure 3.4: Beamforming done locally at the sensor cluster reduces energy dissipation.**

computation energy. Commercial radios typically dissipate ~150 nJ/bit and the StrongARM dissipates 1 nJ/bit. Therefore, communication is more expensive than computation, allowing one to perform 150 instructions per bit communicated [47]. A custom application specific hardware implementation achieves up to 3 orders of magnitude lower energy than a general purpose processor, so millions of operations are performed per bit communicated.

Clustering protocols that utilize sensor collaboration are able to achieve a great deal of energy savings. Using LEACH (Low Energy Adaptive Clustering Hierarchy), an energy-efficient clustering protocol, the sensors are organized into local clusters, as shown in Figure 3.3c [27]. Each cluster has a clusterhead, which is a sensor that receives data from all other sensors in the cluster. At the clusterhead, data aggregation is performed and the results are transmitted to the end-user. This greatly reduces the total amount of data that is sent to the remote end-user and thus achieves energy-efficiency.

Figure 3.4 compares the amount of energy required to communicate information from an M sensor cluster (M=3,6,9) and to transmit the result to the end-user. The ratio of energy for direct communication vs. energy for a clustering approach ($E_{direct}/E_{cluster}$) is plotted as a function of distance from end-user to sensor cluster. When the distance to the end-user is large (d> 7 m), the communication energy dissipated transmitting data to the end-user dominates, and there is a large advantage to performing local signal processing

(e.g., LOB estimation). There is a tremendous amount of communication energy savings because the only data transmitted to the end-user is the value of the LOB estimate.

### 3.2.3 Energy-efficient system partitioning between sensors and clusterhead

Another way to reduce energy dissipation is to recognize that a multiple sensor cluster is equivalent to a multiple microprocessor system. Because all sensors must be energy-aware and take on multiple roles in the constantly changing network, each microsensor node is equipped with a microprocessor for computation. A multiple microprocessor sensor system allows for distribution of computation among the sensors.

The energy-efficiency of system partitioning is demonstrated by implementing the LOB estimation on multiple sensors. The LOB estimation algorithm is implemented in two different ways. In the *direct* technique, each sensor has a set of acoustic data, $s_i(n)$. This data is transmitted to the clusterhead where the LOB estimation algorithm is performed. This includes seven FFTs, frequency-domain beamforming and LOB estimation from FFT coefficients. The direct technique is demonstrated in Figure 3.5(a). Alternatively, the FFTs are performed at each sensor and then the FFT coefficients are sent to the clusterhead. At the clusterhead only beamforming is performed. This is the *distributed* method and is shown in Figure 3.5(b). If we assume a fixed voltage supply, then performing the FFTs with the distributed technique has no computational energy savings over the direct technique, because the same total amount of computation is being done. However, by having a DVS enabled sensor node, we take advantage of the parallelized computational load by allowing voltage and frequency to be scaled while still meeting latency constraints.

Table 3.1 shows the computation energy for a 7 sensor cluster. In the direct technique, with a computation latency constraint of 20 msec, all of the computation is performed at the clusterhead at the fastest clock speed, $f$=206 MHz at 1.44V. The energy dissipated by the processor is measured to be 6.0 mJ and the latency is 19.2 msec. In the distributed technique, the FFT is parallelized to the sensor nodes. In this scheme, the sensor nodes sense data and perform the 1024-point FFTs on the data before transmitting the FFT data to the clusterhead. At the clusterhead, the beamforming and LOB estimation is done. Since the FFTs are parallelized, the clock speed and voltage of both the FFTs and the

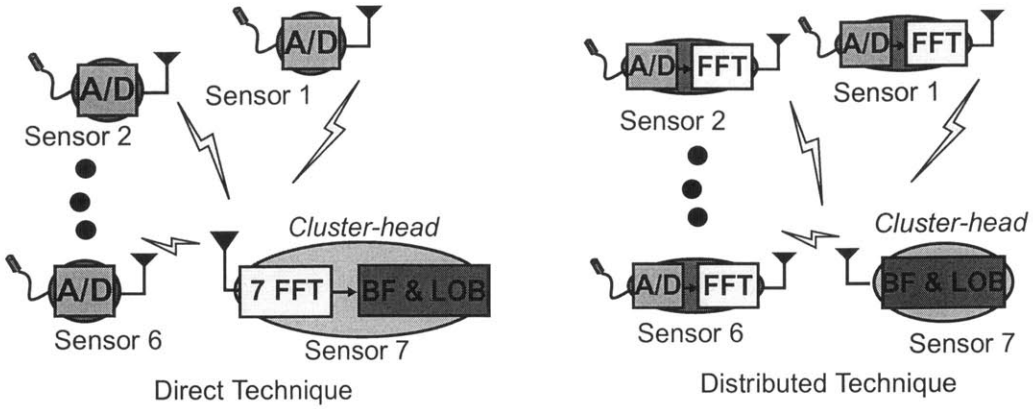


**Figure 3.5: (a) Direct technique: All of the computation is done at the clusterhead. (b) Distributed technique: Distribute the FFT computation among all sensors.**

beamforming is lowered. For example, if the FFTs at the sensor nodes are run at 0.85V voltage supply and 74 MHz clock speed while the beamforming algorithm is run at 1.17V and 162 MHz clock speed with a latency of 18.4 msec, only 3.4 mJ energy is dissipated by the processor. This is a 43.3% improvement in energy dissipation. Thus energy-efficient system partitioning by using parallelism in system design yields large energy savings.

| | | Direct | Distributed |
|---|---|---|---|
| Nodes | $V_{DD}$ | NA | 0.85 V |
| | $f$ | NA | 74 MHz |
| Clusterhead | $V_{DD}$ | 1.44V | 1.17 V |
| | $f$ | 206 MHz | 162 MHz |
| | Latency | 19.2 msec | 18.4 msec |
| | Energy | 6.0 mJ | 3.4 mJ |

**Table 3.1: Energy results for direct and distributed technique for a 7 sensor cluster.**

Figure 3.6 shows the timing diagram for the three cases: Direct technique, Distributed technique w/o DVS, and Distributed technique w/ DVS. For all three scenarios, at time $t$, the sensors begin sensing acoustic data from the microphones. After a block of data is collected, then the nodes begin processing the data. Note, that while the computation is being done, at the same time, new data is being collected from the microphones. In order to minimize buffer size, the LOB estimation computation and inter-sensor communication is completed before the next block of data has been collected.

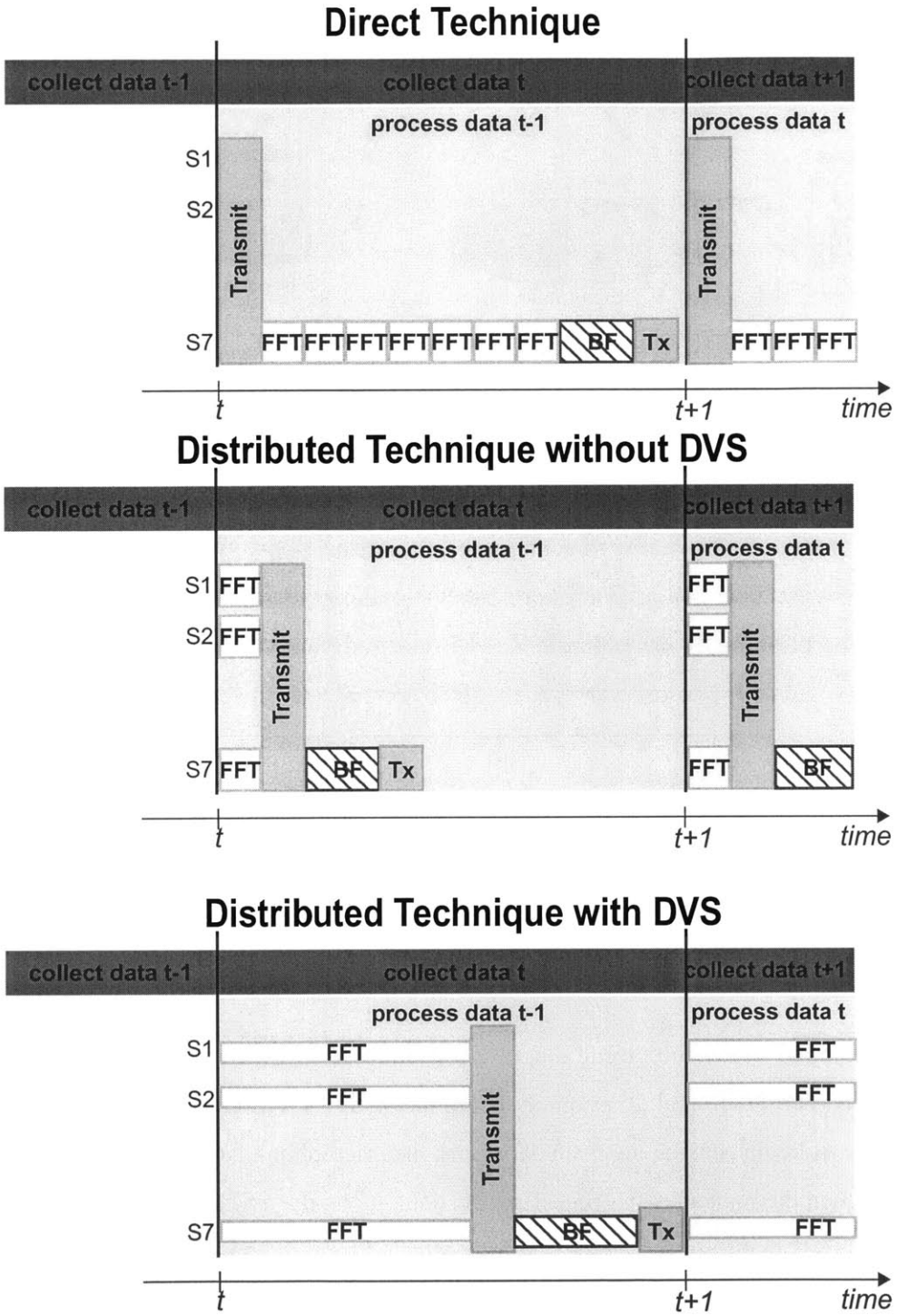


**Figure 3.6: Timing diagrams of three techniques: Direct technique, Distributed technique w/o DVS and Distributed technique w/ DVS.**

In the direct technique timing diagram, the clusterhead sensor (S7) performs all of the signal processing for LOB estimation. In the distributed technique w/o DVS, distributing the FFT to the sensors help improve the throughput of the LOB estimate, but does not decrease the total computation energy dissipated. However, since this is a fixed latency system, the processors are idle for a long period until the next block of data is collected. By using DVS to reduce clock frequency and voltage supply, the processors are active the entire period and energy dissipation is also reduced.

An additional bonus in distributing the FFT is the reduction of communication energy between sensors. Due to the nature of the sound source, the signal of interest has a bandwidth between 20Hz and 256 Hz. This means that after doing the 1024-point FFT, only 236 complex Fourier coefficients need to be transmitted. This means that the communication energy for the distributed technique is reduced by 50% over the direct technique, where all 1024 samples of new sensor data are transmitted.

### 3.2.4 Optimal voltage supply and clock frequency scheduling

By distributing the computation across sensors and using DVS, overall energy dissipation is reduced. The next step is to determine the optimal voltage supply and clock frequency for minimal energy dissipation. Quantifying the optimal operating point is important because the system should adjust operating voltages and frequencies of the sensor nodes to changes in system parameters (e.g. number of sensors, number of samples, etc.). Also, the operating system has the ability to run different tasks (FFT, beamforming, etc.) at different $V_{DD}$ and clock frequencies. A closed form solution of the optimal voltage supply and clock frequency is needed to minimize energy while ensuring the computation is performed within the latency constraint.

Finding the optimal operating points is non-trivial because the energy curve is a non-linear function of frequency and voltage. Figure 3.7 shows the computational energy dissipated for the 7 sensor LOB estimation application for a fixed latency requirement of 20 milliseconds. The curve is plotted as a function of all possible $<f_{fft}, f_{bf}>$ pairs, which satisfy the latency constraint. As a rule of thumb, it is best to run the parallelized task (FFT) at the lowest voltage and frequency level to get large energy savings. However, to satisfy

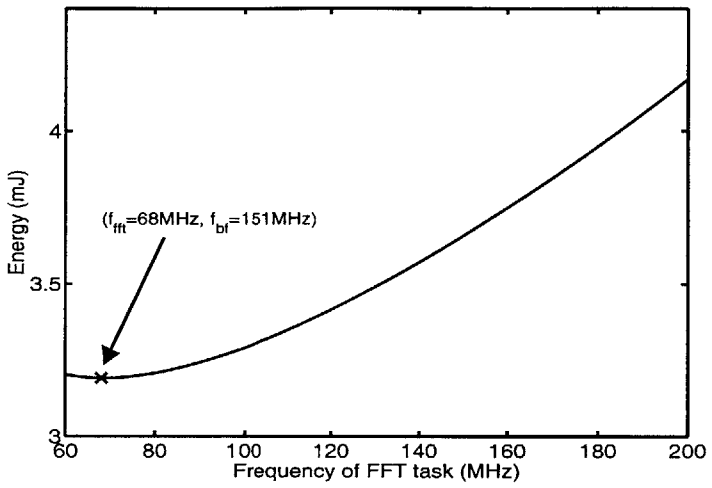


**Figure 3.7: Energy dissipated by 7 sensor cluster for different $\langle f_{fft}, f_{bf} \rangle$ pairs for a latency constraint of 20 msec.**

the timing constraint, beamformer voltage supply and clock frequency is increased. Therefore, there is an optimal operating voltage and frequency for the source tracking algorithm.

In order to find a closed solution for the optimal operating voltage and frequency, we minimize the total computation energy for an $M$ sensor cluster

$$E_{comp} = MN_{fft}CV_{fft}^2 + N_{bf}CV_{bf}^2 \tag{3.6}$$

with the latency constraint that

$$T_{comp} \geq \tau_{fft} + \tau_{bf} = \frac{N_{fft}}{f_{fft}} + \frac{N_{bf}}{f_{bf}}. \tag{3.7}$$

Here, $V_{fft}$ and $V_{bf}$ are the operating voltages of the two tasks, FFT and beamforming, and $N_{fft}$ and $N_{bf}$ are the cycle counts, respectively, and are benchmarked to be

$$N_{fft} = 200.73 \text{ kcycles} \tag{3.8}$$

$$N_{bf}(M) = 319.3M + 341.6 \text{ kcycles} \tag{3.9}$$

The optimal voltage and frequency operating points gives minimum energy dissipation while maintaining the latency constraint. In order to find a closed form solution, first, use the processor frequency equation (Eq. 3.5) to relate the latency constraint, Eq. 3.7, in

terms of voltage supply. A Lagrangian minimization is performed, where the energy is minimized with the latency constraint. We are able to solve for the relation between $V_{bf}$ and $V_{fft}$.

$$(V_{bf} + c) = \sqrt[3]{M}(V_{fft} + c) \tag{3.10}$$

In order to calculate the closed form solution for the optimal operating point, Eq. 3.10 is substituted back into Eq. 3.7 and $V_{fft}$, $V_{bf}$, $f_{fft}$, and $f_{bf}$ is solved for

$$V_{fft} \geq \frac{1}{T_{comp}K}\left[N_{fft} + \frac{N_{bf}}{\sqrt[3]{M}}\right] + c \tag{3.11}$$

$$V_{bf} \geq \frac{\sqrt[3]{M}}{T_{comp}K}\left[N_{fft} + \frac{N_{bf}}{\sqrt[3]{M}}\right] + c \tag{3.12}$$

$$f_{fft} \leq \frac{\left(\sqrt[3]{M}N_{fft} + N_{bf}\right)}{\sqrt[3]{M}T_{comp}} \tag{3.13}$$

$$f_{bf} \leq \frac{\left(\sqrt[3]{M}N_{fft} + N_{bf}\right)}{T_{comp}} \tag{3.14}$$

These equations suggest that it is desirable to run the parallelized task (FFT) at lower voltage and frequencies than that of the non-parallelized task (beamforming) by a factor of $\sqrt[3]{M}$.

Figure 3.8 shows energy dissipated for the direct technique vs. distributed technique with optimal voltage scheduling as $M$ is increased from 3-10. The top line is the energy dissipated for the direct technique and the bottom line is the energy dissipated for the distributed scheme. The discrete measurements (o,*) show the measured energy dissipated by the StrongARM compared to the solid and dashed lines which give the theoretical results given by Equations 3.11-3.14. This graph shows that the processor energy models and theoretical results match closely with the energy dissipated by the StrongARM. The figure shows that between 30-50% energy reduction is achieved with the system partitioning scheme. The amount of energy reduction increases as the number of sensors increases.

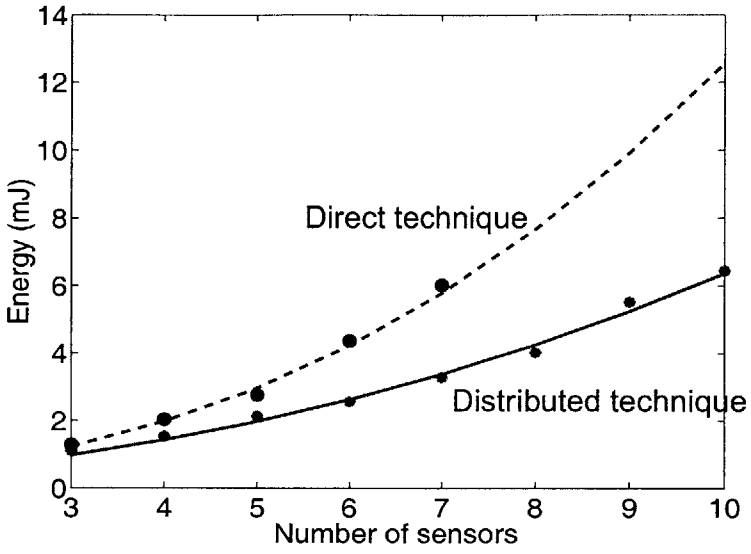


**Figure 3.8: Comparing energy dissipated for the direct technique vs. the distributed technique with DVS.**

In this analysis we assume that the processor is able to operate at continuous voltage and frequency levels. In reality, our implementation only provides a limited voltage supply and frequency range and discrete voltage and frequency levels. In order to use the closed form solutions on our implementation there is a constraint that $T_{\text{comp}}$ fall between the range of

$$\frac{N_{\text{fft}} + N_{\text{bf}}}{f_{\max}} \leq T_{\text{comp}} \leq \frac{N_{\text{fft}} + N_{\text{bf}}}{f_{\min}} \tag{3.15}$$

where $f_{\max}$ and $f_{\min}$ are the maximum and minimum frequencies possible for the processor. If $T_{\text{comp}}$ falls below the lower limit, there is not enough time to complete the computation, and if $T_{\text{comp}}$ is above the upper limit, minimizing energy dissipation means operating at the slowest clock frequency and lowest voltage level. If this constraint is fulfilled, then the optimal operating point is the voltage/frequency given by the theoretical equations.

In wireless sensor networks, the latency requirement is much lower than the 20msec given in the example. The sensor A/D collects 1024 samples of sensor data in one second. Therefore, the latency requirement of the FFT itself is very long (> 500msec). The theoretical equations for the StrongARM processor, predict an optimal voltage supply of 520mV and clock frequency of 3MHz, which is not an operating point available with the current

StrongARM implementation. 48.9μJ/FFT is the minimum energy dissipated by the StrongARM at 0.85V voltage supply and clock frequency of 74MHz, for a 512-point FFT. At the lowest power supply, the processor dissipates 39mW. The StrongARM implementation does not meet the energy constraint, indicating the need for a dedicated FFT processor that operates at lower voltage supplies.

### 3.2.5 Generalized Tasks

The system partitioning scheme given in the above example is generalized to any application where parallelism is exploited to distribute computation among multiple processors. Let us assume there are two tasks, $A$ and $B$, each of which are characterized by their cycle counts, $N_A$ and $N_B$, respectively. $A$ is the task to be parallelized to the $M$ processors and $B$ is the non-parallelized task. Also assume that the bandwidth between the processor does not change for the two schemes. In the direct scheme, the frequency of the processor for each task is set to

$$f_A = f_B = \frac{MN_A + N_B}{T_{\text{comp}}} \tag{3.16}$$

For the distributed scheme, the voltages and frequencies are given in Eq's 3.11-3.14.

The ratio of $E_{\text{direct}}$, the energy of the direct technique to $E_{\text{distributed}}$, the energy of the optimal distributed technique, is calculated to approximately

$$\frac{E_{\text{direct}}}{E_{\text{distributed}}} = \frac{(M \cdot N_A/N_B + 1)^3 + D(M \cdot N_A/N_B + 1)^2}{M\left(N_A/N_B + M^{-1/3}\right)^3 + D \cdot M\left(N_A/N_B + M^{-1/3}\right)\left(N_A/N_B + M^{-2/3}\right)} \tag{3.17}$$

where $D=2cKT/N_B$. This ratio shows that as the number of processors ($M$) increases there is a large opportunity for energy savings because there is more parallelism. Figure 3.9 shows a plot of the energy ratio, $E_{\text{direct}}/E_{\text{distributed}}$, as a function of cycle ratio, $N_A/N_B$, for $M$=5,7,9. This shows that when the computation for task $A$ (parallelized task), is relatively large compared to task $B$, (e.g., $N_A/N_B$=5) then there is a great deal of energy savings. However, even when the computation in task $A$ is small compared to that for task $B$ (e.g., $N_A/N_B$=0.05), there is still a possibility for energy savings of 2x. The upper limit of energy

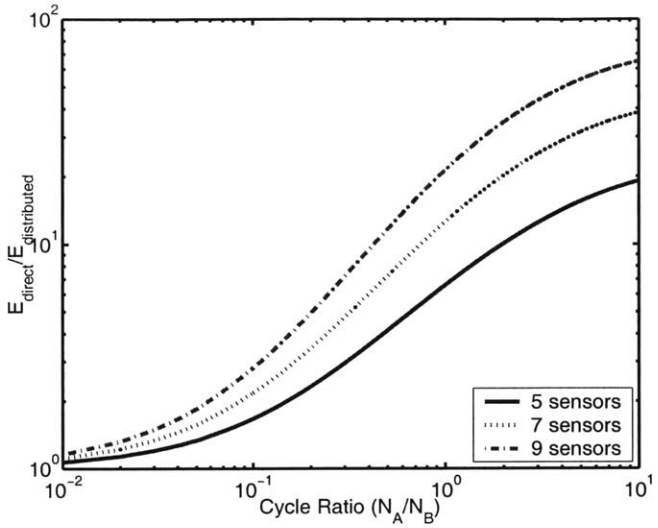


**Figure 3.9: As the cycle ratio increases, energy savings increase for a 7 sensor cluster.**

savings is calculated by taking the limit of the energy ratio as $N_A$ goes to infinity. The limit on energy savings is $M^2$. This result assumes the linearized energy and frequency processor models introduced previously. Therefore, for the general case, there is a potential for a great deal of energy savings.

## 3.3 Summary

In this chapter, I first provided background on sensor networks and the MIT μAMPS sensor node architecture as a system design driver. The frame work of the sensor node is needed to fully understand the impact of energy-aware algorithms on a system-level.

The concept of system partitioning for system-level energy-awareness is introduced. System partitioning across the network takes advantage of communication vs. computation trade-offs to reduce energy dissipation. In general, since computation is less expensive than communication, it is advantageous to do local signal processing at the sensor cluster. This leads to a new way of thinking for the design of networking protocols for these data-rich sensor networks.

System partitioning within the network takes advantage of the multiple microprocessors within the sensor cluster. Parallelizing and distributing computation to all sensors and DVS reduces the energy dissipated. A scheme is introduced to find the optimal voltage

supply and clock frequencies for the FFT and frequency-beamformer tasks. The optimal voltage supply and clock frequency scheme is generalized for an $M$ sensor cluster to show that up to $M^2$ energy savings is achieved.

These examples assume a short latency requirement for the LOB estimation computation in order to see the effect of parallelism on the StrongARM implementation. In reality, the latency requirement is much longer. 48.9μJ/FFT is the minimum energy dissipated by the StrongARM at 0.85V voltage supply and clock frequency of 74MHz, for a 512-point FFT. The StrongARM implementation does not meet the energy constraint, indicating the need for a dedicated FFT processor implementation that is able to scale voltage supply down below 0.85V.

# Chapter 4

# Energy-aware Architectures

The goal of energy-aware architecture design is to allow energy-quality trade-offs in hardware. Hardware knobs in the architecture allow the system to vary the energy consumption as the environment or user requirements change. By tuning the hardware for the desired scenarios of interest, energy-efficiency is achieved.

In this thesis, an FFT processor for microsensor applications is used as a design driver. A dedicated processor allows for FFT specific energy scalability hooks so that both low-energy/low-quality and high-energy/high-quality FFTs are computed efficiently on the same hardware. Our design focuses on a real-valued FFT which scales between 128 to 1024-point FFT lengths and can produce both 8- and 16-bit precision FFTs.

## 4.1 Previous work

Dedicated hardware achieves orders of magnitude energy savings over general purpose programmable hardware by minimizing the overhead of performing computation. Energy-scalable hooks provide enough programmability in the dedicated hardware for efficient scalability. Previous work on dedicated scalable architectures is presented.

Also, previous work on various FFT architectures is discussed. Because the FFT accelerates many signal processing tasks, various FFT architecture implementations are found in literature.

### 4.1.1 Signal processing hardware

General purpose processors, such as the microprocessor or a programmable digital signal processor (DSP) facilitate energy-awareness in algorithms. Energy-scalability on traditional microprocessors is enabled through software transformations of the algorithm. On the low-power StrongARM microprocessor, supply voltage and clock frequency scaling is used to scale energy dissipation for a variable workload [40]. Another programmable

hardware is the Field Programmable Gate Array (FPGA). The FPGA can be configured for a variety of architectures, and exploit parallelism to achieve energy scalability [51].

Dedicated hardware is preferable to the microprocessor and the FPGA in multiple ways. Profiling of the instruction set of various microprocessors and programmable DSPs shows that the average current consumption is fairly uniform across all instructions. The overhead circuitry that is common to all instructions such as cache, global clock, I/O, etc., dominates the power dissipation [60]. Similarly, the FPGA's main downside is the power overhead associated with the interconnect, which dominates the energy dissipated by the chip. By tailoring the hardware to specific functions required by the algorithm, dedicated hardware minimizes switched capacitance and significantly reduces energy dissipation. In a micropower programmable DSP designed for filtering operations of sensor data, the energy dissipation of the application specific DSP is six orders of magnitude lower than the StrongARM [2].

Table 4.1 shows a comparison of the estimated energy dissipation of the FFT implemented on the StrongARM-1100 microprocessor, on a Xilinx FPGA, and on a dedicated FFT ASIC, all operating at 1.5V voltage supplies. The StrongARM-1100 power is estimated using the JouleTrack simulator [59]. JouleTrack compiles the FFT C code to assembly code and estimates the average current dissipated by the StrongARM. The Xilinx FPGA power is estimated using the Virtex Power Estimate Worksheet [72]. The FFT verilog block is synthesized and the breakdown of the FFT into the number of CLB slices and Block RAM are given. These specifications are inputted into the spreadsheet and the power is estimated for a target Xilinx device. Given the power estimates for both the StrongARM and the Xilinx, the energy per FFT is estimated. The energy for the custom FFT are estimated from a *nanosim* simulation a dedicated FFT architecture.

The table shows the energy for the microprocessor is 7x higher than the energy for the FPGA for FFT length=128pt. At 1024-point FFT, the FPGA dissipates higher energy dissipation than the microprocessor. However, the energy dissipation of the dedicated FFT is over two orders of magnitude lower than both the StrongARM and the Xilinx. These results indicate that application specific hardware is critical for applications that require extremely low energy dissipation.

**Table 4.1: Comparing simulated energy dissipation of the three FFT implementations.**

| | 128 pt. | 256 pt. | 512 pt. | 1024 pt. |
|---|---|---|---|---|
| StrongARM | 44 μJ | 46 μJ | 50 μJ | 59 μJ |
| Xilinx | 5.7μJ | 13 μJ | 29 μJ | 65 μJ |
| Custom | **116 nJ** | **269 nJ** | **628 nJ** | **1.49 μJ** |

### 4.1.2 Energy-Scalable Architectures

Traditional low-power design focuses on optimizing for the worst-case operation. In the case where the application is subject to diverse scenarios, new architectures are required to achieve energy-efficiency.

A prime example of an innovative energy-scalable architecture is distributed arithmetic (DA). DA performs a bit-serial multiplication that assumes that one of the vectors in the inner product is fixed. All possible intermediate computations for the fixed vector are stored in a Look-up Table (LUT) and bit slices of the variable vector are used as addresses for the LUT. By accumulating the MSB first, the DA is a variable precision energy-scalable architecture with desirable E-Q characteristics. With a small amount of energy, a less accurate, lower bit-precision result is achieved. Each incremental increase of energy leads to an incremental improvement of quality of the multiply operation [78]. Many applications employ a DA architecture for energy-scalability, such as a low power DCT core to do adaptive bitwidth computation [80].

Another example of a scalable architecture is found in approximate processing techniques used for filtering [35]. The filter uses a feedback loop to dynamically change the filter order based on signal statistics. Powering down parts of the tapped-delay line of an FIR filter reduces the filter order and also reduces the switched capacitance and energy dissipation. This technique is also extendable to infinite impulse response (IIR) filters.

Dynamically reducing switched capacitance for energy-scalability is also explored in the ensemble of point solutions method [6]. This method consists of multiple point solutions, each optimized to a particular scenario. An ensemble of the optimized solutions is chosen to cover the entire scenario space. Low-overhead control logic routes the inputs to

the correct point solution. Clock and data gating of unused point solutions avoids unnecessary switched capacitance.

A multiplier in a speech application employs the ensemble of point solutions method to achieve energy-awareness [6]. First, the probability distribution of the bit-precisions of multiplicands and multipliers in a speech application is analyzed. The probability distribution indicates that with highest probability the input bit-precision is 4 or 5 bits. The least probable bit-precision is the largest possible bit-precision of 16-bits. The ensemble of point solutions method creates an energy-scalable multiplier from multipliers with different input bitwidths: 9-, 11-, 14- and 16-bits. Control logic routes the input data to the correct multiplier based on the detected input precision. Gating logic suppresses the inputs to the unused multipliers and prevents unnecessary switched capacitance. The amount of energy saved by the application is 60% in comparison to a non-scalable solution. The non-scalable multiplier targets the worst case bit-precision (16-bits).

Other applications that explore scalablility in DSPs include encryption processors for Public-Key Cryptography [21], image processing [22], coding [69], automatic target recognition [13], and fuzzy processors [3].

### 4.1.3 FFT architectures

The FFT processor is typically made up of a datapath, data memory and twiddle factor generator. This section gives an overview of previous architectures for the FFT.

Previous datapath architectures in dedicated FFT processors range from a single multiplier ALU to a higher order radix FFT. The most common FFT datapath architecture is a radix-2 butterfly, the basic functional block of the FFT. According to the FFT algorithm, if one butterfly is performed per cycle, then an N-point FFT requires $N/2*\log_2 N$ clock cycles. Going to higher order radix FFTs means shorter FFT latencies by using parallel butterfly modules. A k-radix FFT has $k$ butterfly modules in parallel, and thus only requires $N/k*\log_k N$ clock cycles. Higher radix implementations are often used for high speed applications [16][36][48][57]. Because latency is not a key metric for sensor applications, the FFT processor in this thesis has a radix-2 datapath architecture.

A radix-2 datapath requires multiple port memories because two values are read from memory in one clock cycle. Different techniques are used to avoid hazards when reading from and writing to data memory. Using a dual read/write-port memory is one solution to avoids data hazards when accessing multiple locations in memory. However, the overhead of a dual port memory can be reduced by using specialized FFT specific memory structures [36]. Analysis of the memory accesses required by the FFT is exploited to simplify the FFT memory design. FFT processor memories are usually comprised of static RAM blocks, register files, or shift register banks [57].

Different methods are proposed for twiddle factor generation. One method is to store the twiddle factor data in read-only memories (ROMs) or in programmable register files [48][67]. Twiddle values are also generated by using integer logic or cordic arithmetic [79].

The overall architecture and control logic of the FFT chip depends on the design specifications, such as latency or energy dissipation. Most FFT architecture designs focus on high-speed rather than low-power. Multiple datapath chips, external memory, and external control architectures achieve high throughput FFTs for real-time applications [15][36]. Most implementations have pipelined architectures to improve throughput [5][25][57][79]. A very deeply pipelined architectures that use bit-serial arithmetic and a large mesh network is proposed to exploit the FFT parallelism and achieves high performance [67].

In one implementation that is designed for low power dissipation, a small cache memory is used to store a small subset of the larger FFT memory. The data in the cache is processed quickly, and stored back into the large data memory before another small subset is loaded. The cached-memory structure provides increased speed and increased energy-efficiency by lowering the energy per access and overall switched capacitance [4].

Most FFT processors are designed for applications where the input data is complex-valued. Only in such applications such as video, image, and audio processing where real-valued data is processed, the FFT processor is optimized for real-valued input data by including additional pre- and post-processing hardware [48][79].

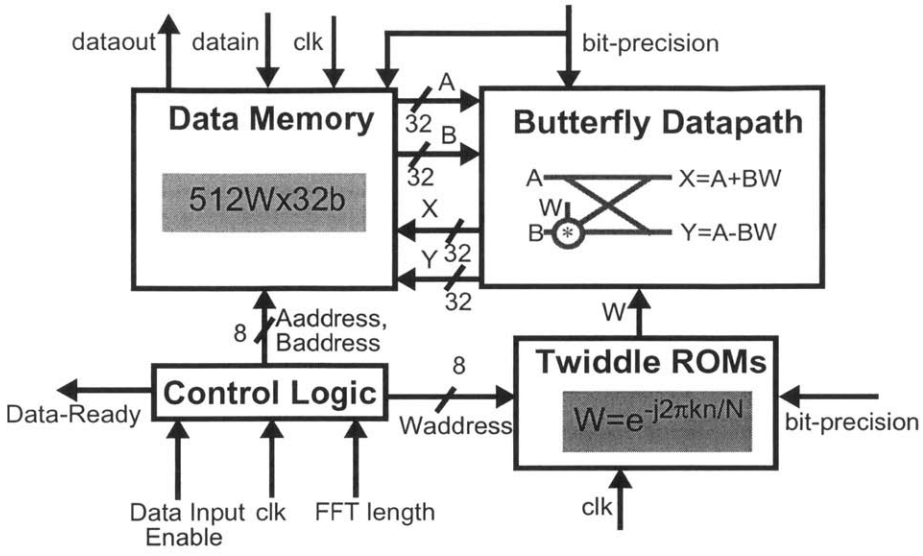


**Figure 4.1: A block diagram of the FFT architecture.**

## 4.2 FFT architecture overview

Figure 4.1 shows the block diagram of the architecture for the FFT core. The FFT core has four functional blocks: Control Logic, Butterfly Datapath, Data Memory, and Twiddle ROMs.

The Control Logic controls the timing of the four functions to perform the N-point real-valued FFT: (1) reordering inputs from the datain bus and storing them into data memory, (2) performing the N/2-point CVFFT, (3) performing RVFFT backend processing and (4) outputting the data onto the dataout bus. The control logic sends control signals necessary for correct functionality, for configuration purposes, and to enable energy-awareness. Examples of important control functions are configuring the datapath, sending correct addresses to the memory blocks, ping-pong control between the two memory banks, interfacing with the input and output bus, and sending energy-scalability information to all blocks.

During the CVFFT and RVFFT processing stages, one butterfly computation is performed per clock cycle using the datapath. The datapath of the FFT consists of the butterfly and the backend processing circuits. The hardware in the datapath includes complex multiplications and complex addition. Additional adders and subtractors perform the

backend RVFFT processing. The E-Q scalable hook enabled in the datapath is bit-precision scaling.

On the clock edge the FFT processor reads two complex values from the memory and one complex twiddle factor from the ROM. Then, one butterfly computation is performed and the two complex outputs are written back into memory on the next clock edge. Energy-scalable design is applied to the memory to enable variable bit-precision (8-bit and 16-bit) and variable memory size (128-1024 Word) through memory partitioning and data/clock gating.

Because the FFT performs in-place computation, the FFT processor only requires a buffer that holds one frame of data (1024-Word x 16bit). In-place computation implies that the outputs from the datapath are written back into the memory read locations. Also in-place computation dictates that reordering the inputs is necessary so that at the output, the FFT coefficients are ordered from the lowest to highest frequency indices in the memory.

There are two possible memory architectures for I/O: a single memory bank structure or a dual memory bank structure. A single memory bank structure has 1024Wx16bit capacity. After one frame is loaded into the memory, processing begins and the memory is inaccessible by the user until the FFT task is performed. A dual memory bank structure is needed if data is being streamed to the FFT at a constant data rate and the FFT is performed concurrently. While one memory bank loads in sensor data, the second memory bank performs the FFT processing. The dual memory structure is also called a ping-pong memory. The ability to ping-pong between memory banks allows a improved throughput at the expense of an extra memory bank. In the FFT ASIC design implemented in this chapter, a ping-pong memory is used and in the subthreshold FFT chip discussed in Chapter 6, the single memory bank is used.

A Read-Only Memory (ROM) provides the Twiddle factors for the CVFFT and the backend processing. The ROM outputs Twiddle factors with 8- or 16-bit precisions.

The energy consumption breakdown of the 1024-point FFT shows that the datapath and memory consumes the majority of the energy. Therefore, the largest energy-scalability range is achieved by focusing on enabling scalability in the datapath and the memory (RAM and ROM).

### 4.2.1 Energy-aware architecture design approach

An area and energy efficient design methodology proposed is the reuse of point solutions method [76]. In most scalable systems, scenarios of interest are similar enough so that common functional blocks are shared and reused. Therefore, the design of all functional blocks is tightly integrated to avoid the additional area penalty. In this section, the datapath and the memory design of the FFT provide a platform for further analysis and comparison between the reuse and the ensemble of point solutions method as well as non-scalable solutions.

## 4.3 Energy-aware datapath

Figure 4.2 shows the butterfly datapath of the FFT. The main engine of the datapath is the CVFFT butterfly, which contains a complex valued multiplication followed by complex addition/subtraction. The function of interest in an FFT butterfly is

$$X = A+B*W \text{ and } Y=A-B*W, \tag{4.1}$$

where A, B, and W are complex inputs, and X and Y are the complex outputs. This is performed for ($\log_2$N-1) stages of the N-point RVFFT. Complex multiplication consists of four Baugh-Wooley (BW) multipliers, an adder and a subtractor. A BW multiplier performs two's complement multiplication.

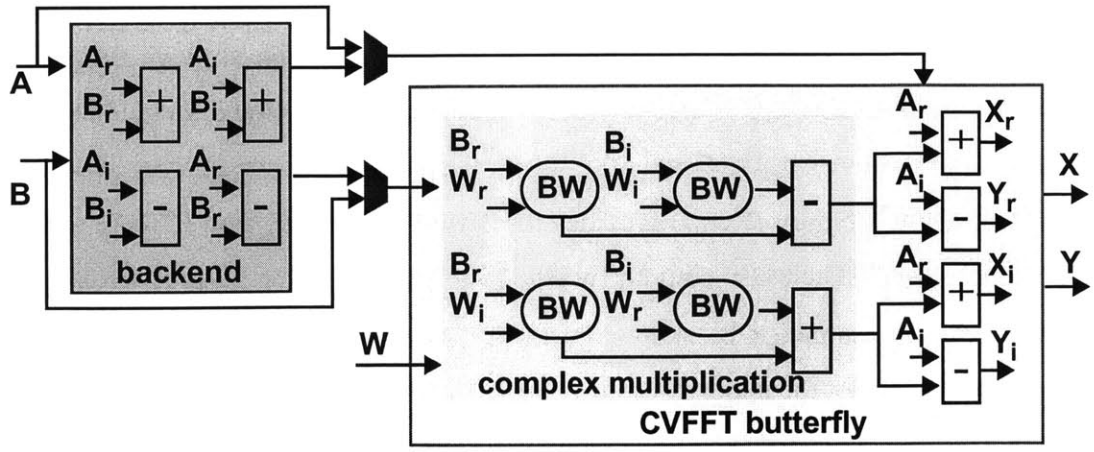


**Figure 4.2: FFT butterfly and backend processing datapath.**

In the last stage of the RVFFT, the inputs are first fed into the backend processing block which performs,

$$A_{backend} = (A_r+B_r) + j\,(A_i-B_i) \tag{4.2}$$

$$B_{backend} =(A_i+B_i) + j\,(A_r-B_r). \tag{4.3}$$

where $A_r$ and $A_i$ represent the real and imaginary parts of complex value A. The backend results are fed into the complex butterfly to compute the N-point RVFFT coefficients from the N/2-point CVFFT coefficients.

By fixing the datapath to be a certain width, the FFT coefficients incurs errors due to rounding off. The maximum bit-precision of the FFT is 16-bit. The memories store 16-bit real-valued data (32-bit complex values). The inputs to the datapath are 16 bits and the outputs from the datapath are rounded off to be 16 bits. Inside of the datapath the bitwidth grows to 34 bits. An example butterfly computation is,

$$X_r= B_r*W_r-B_i*W_i + A_r \tag{4.4}$$

This computation calculates the real part of the X output. The twiddle factors ($W_r$ and $W_i$) are 16 bit fractions that are less than 1. The data ($A_r$, $B_r$, $B_i$) are 16 bit integers. The output to the 16x16 bit multiplier ($B_r*W_r$, $B_i*W_i$) is 32 bits. Then the two 32-bit products are added to $A_r$ which increases the bitwidth to 34 bits. In order to write back a 16-bit integer to memory, the last 15 bits are truncated and the 3 MSB are checked for overflow conditions. Truncation and saturation adds estimation error to the FFT coefficients. This error is the cost for the large energy savings associated with fixed point.

The critical path of the FFT butterfly determines the maximum clock frequency of the FFT. The FFT butterfly critical path includes adders in the 16b Baugh Wooley multiplier, the two's complement subtractors, the backend processing adders, the overhead delay through gating logic and buffering. The critical path also includes additional crossbar logic in the memory module which is discussed in Section 4.4.

The energy-scalable hook designed into the datapath is bit-precision scaling. The datapath performs both 8- and 16-bit precision FFTs. The concept of reuse of point solutions for energy-efficient scalability is introduced using the BW multiplier as an example.

### 4.3.1 Energy-scalable Baugh Wooley (BW) multiplier

The BW multiplier design showcases energy-scalable bit-precision design. Figure 4.3 shows the logic diagram for a 4-bit BW multiplier performing two's complement arithmetic. Figure 4.4 shows the gate level implementation of the logic diagram in terms of adders, NAND, and NOR gates.

A non-scalable design optimizes for the worst case scenario by building a single BW multiplier for the largest bitwidth. In a non-scalable architecture both 8 and 16 bit-precision multiplications are performed on a single 16-bit multiplier. Due to the switching overhead of the two's complement sign extension bits when performing an 8-bit multiplication on a 16-bit multiplier, this solution is non-optimal.

One proposed scalable BW multiplier uses the ensemble of point solutions method. The ensemble implementation contains multipliers of a variety of bitwidths. Control logic routes the data to the appropriate multiplier. Figure 4.5 shows a 8- and 16-bit scalable precision multiplier designed with the ensemble of point solutions method.

A second proposed scalable BW multiplier design recognizes that the blocks for a lower bit-precision multiplier are contained within a higher bit-precision multiplier. For example, in a sign-magnitude array multiplier any quadrant of adders can be used for a lower bit-precision multiplication. For the BW multiplier, the MSB quadrant can be reused for lower bit-precision multiplication without significant modification. The MSB quadrant are those gates associated with the MSB of the inputs. For example, in Figure 4.3, the gates that have lighter shading are reused to create a 2-bit multiplier. These adders are the MSB quadrant because the inputs to those gates are the MSBs ($x_2$, $x_1$, $y_2$, and $y_1$). Any other 2x2 quadrant of the multiplier does not have the correct configuration of NAND and NOR gates and would require additional logic. For a 2-bit multiplier that minimizes switching, the inputs to the LSB adders are gated and the LSB inputs are muxed to the MSB quadrants.

Figure 4.6 demonstrates a BW scalable multiplier for 8- and 16-bits. For 16-bit operation, the entire multiplier is enabled. During 8-bit operation, the AND gates at the LSB inputs of the multiplier prevent switching of all of the darkly shaded adders. The LSB inputs are muxed to the MSB quadrant, and only those adders are used in 8-bit multiplica-

tion. Also gating logic is added to the outputs of the 8-bit multiplier. Without this logic, the outputs can cause switching in some of the 16b adders. This is an example of the reuse of point scenarios solution to implement variable bit-precision.

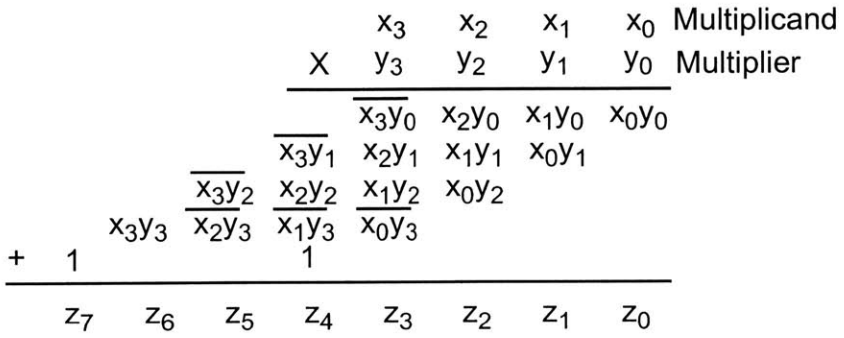


**Figure 4.3: The logic diagram for a 4-bit Baugh-Wooley multiplier.**

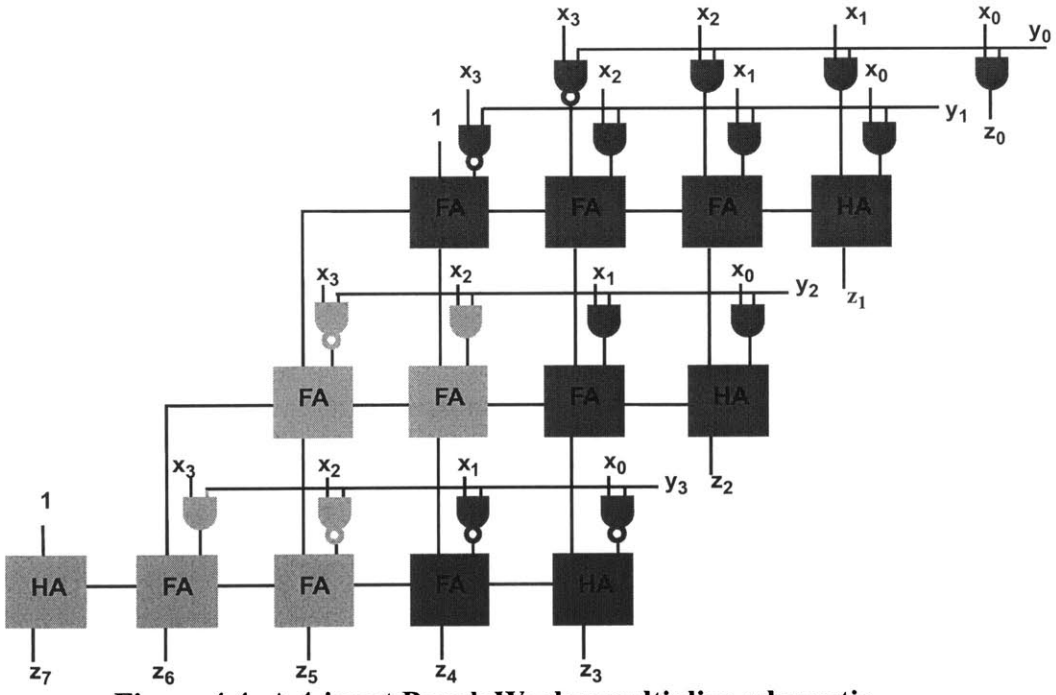


**Figure 4.4: A 4-input Baugh Wooley multiplier schematic.**

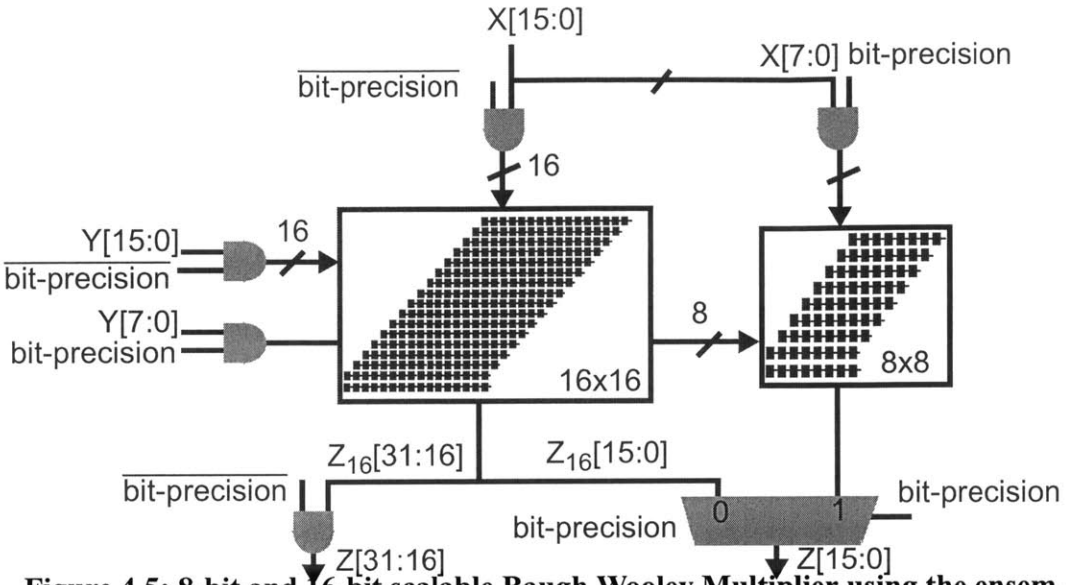


**Figure 4.5: 8-bit and 16-bit scalable Baugh Wooley Multiplier using the ensemble of points method. bit-precision is 1 for 8bit multiplication and is 0 for 16bit multiplication.**

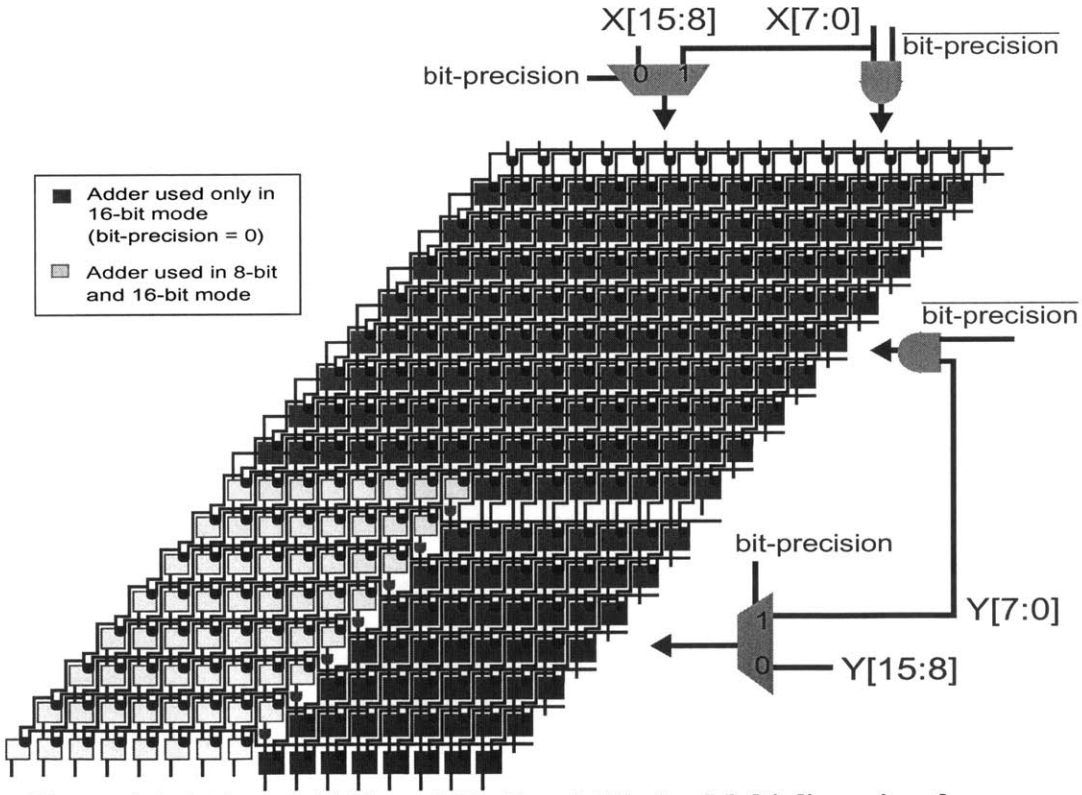


**Figure 4.6: 8-bit and 16-bit scalable Baugh Wooley Multiplier using the reuse method. The 8-bit multiplier is reused for the 16-bit multiplication, thereby adding scalability without a large area penalty.**

### 4.3.2 Results

Three butterfly datapaths are created using a 1.8V standard cell library. Table 4.2 compares layout area, transistor count and energy simulations. The first implementation is the non-scalable datapath that targets the worst-case bit-precision. For the FFT datapath, the non-scalable datapath is the full 16-bit datapath. The second and third implementations are the ensemble of point solutions and the reuse of point solutions datapaths, respectively.

The table shows that the scalable designs incur more area and device overhead than a non-scalable design. The overhead is due to the extra data gating logic needed for scalable designs. The ensemble of points method is 76% larger than the non-scalable design and has 16K more transistors. In comparison with the ensemble method, the reuse method is a more area efficient design. Because the 8-bit datapath reuses hardware in the 16-bit datapath the reuse method is only 7.6% larger than the non-scalable design and has only 3K more transistors.

Both the ensemble and reuse of point solution design methodologies are globally more energy-efficient than a non-scalable solution. For the 8-bit butterfly datapath, the ensemble datapath has 42% less energy dissipation over a non-scalable design, and the reuse datapath has 67% less energy dissipation. However, at the worst case bit-precision of 16-bit, the energy dissipation of the ensemble and reuse designs increases over the non-scalable design, due to the energy overhead from the gating logic. Nevertheless, both the ensemble method and reuse method are more energy-efficient than the non-scalable datapath when the system requires lower bit-precision results.

**Table 4.2: Comparing different FFT datapath implementations**

| | | Non-scalable | Scalable | |
|---|---|---|---|---|
| | | | Ensemble of Points | Reuse of Points |
| Area | | 0.13 $mm^2$ | 0.23 $mm^2$ | 0.14 $mm^2$ |
| Transistor Count | | 47K | 63K | 50K |
| Energy/butterfly | 16-bit | 287pJ | 286pJ | 295pJ |
| | 8-bit | 249pJ | 142pJ | 83pJ |

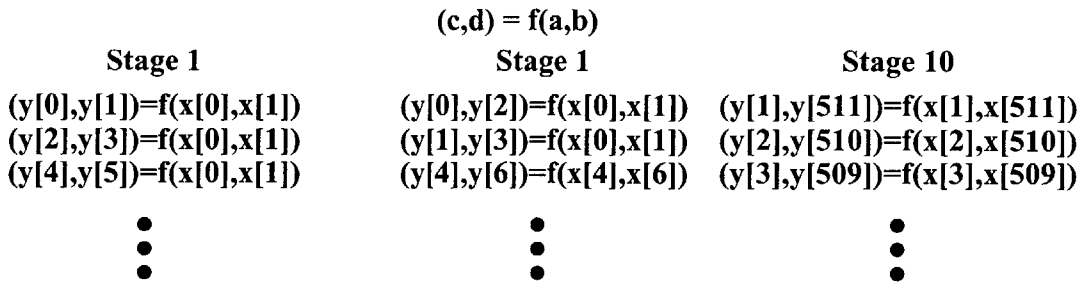


**Figure 4.7: Memory read accesses sequences for each stage of computation for a 1024-point RVFFT. For the first 9 stages, the addresses differ by parity. In the last stage the address differ by MSB.**

These results indicate that the reuse of point solution gives the desired energy-scalability without significant area or device overhead. The ensemble of point solution method also has reduced energy dissipation at the low quality point, but incurs significant area overhead.

## 4.4 Energy-aware memories

For the FFT, the data memory bank is designed to read and write two 32-bit complex values in one clock cycle. 32-bit complex twiddle factors are stored in scalable ROMs.

An important consideration in the design of the FFT memory banks is to avoid memory hazards when accessing two complex values from the memory. One possible design is to use multiple-port (two-read, two-write) data memory that is able to access any two independent memory locations at the same time. Analysis of the memory accesses for the CVFFT show that for every butterfly the memory addresses only differ in their parity [11][52]. The first few memory accesses for each stage of computation of a 1024-point RVFFT are shown in Figure 4.7. For the first 9 stages, the address differ by parity. In the last stage the addresses differ by the MSB, rather than parity.

Therefore, based on these findings, the memory is implemented with four two-port memories instead of one four-port memory. Each four two-port banks is addressable by the addresses parity and by the MSB. A MSB/parity crossbar configuration is used to access the memory while avoiding memory hazards during the CVFFT. This memory architecture with a MSB/parity crossbar is shown in Figure 4.8 for a 1024-point FFT. Each partition is 128 Word addressable and stores a 32-bit complex number.

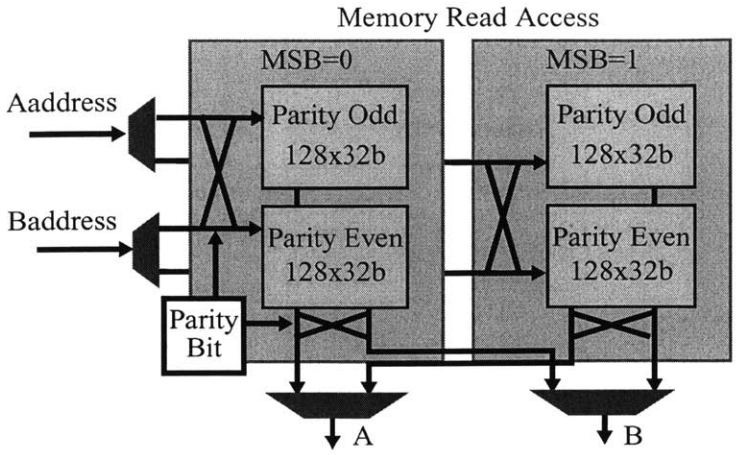


**Figure 4.8: Cross Bar and MSB control to ensure parallel read and write memory access for a 1024-point Real Valued FFT.**

The non-scalable memory designed for the 1024-point FFT is a 512Wx32bit block configured for 128, 256 and 512 length processing. A non-scalable design incurs the most power overhead for the 128-point FFT. To reduce this overhead, a variable sized memory is proposed.

Variable sized memories allow for greater energy-awareness as different FFT lengths are required. An ensemble of point solutions implementation contains the memory banks for four point solutions based on FFT lengths. The memory architecture for the 1024-point FFT is created, then duplicated and scaled for all other scenarios. Muxes are then used to route the inputs and clocks to the appropriate memory block The inputs and clocks to the unused memories are gated. This design is shown in Figure 4.9.

Using the reuse of point solutions method, the memory is shared among all point scenarios. Figure 4.10 shows a memory block designed with the reuse of point solutions method for the 128-1024 point FFT application. When the processor is performing a 128-point FFT, only the four 16Wx32b blocks are used. During a 1024-point FFT all memory blocks are used. The advantage of the reuse of point solutions method is less area required and lower transistor count, at the expense of more complex control logic design.

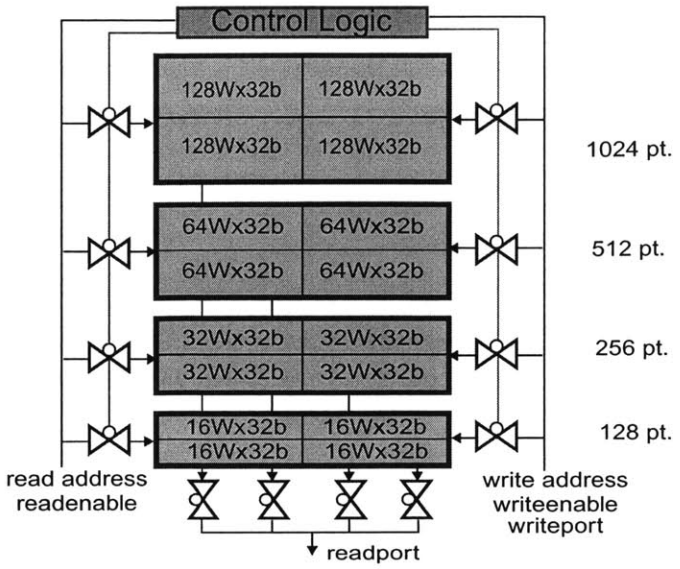


**Figure 4.9: Ensemble of point solutions scalable memory.**

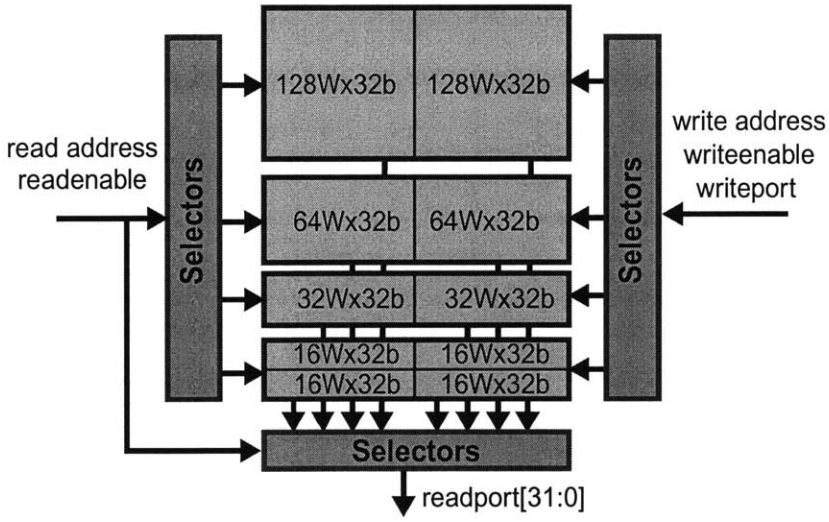


**Figure 4.10: Reuse of point solutions scalable memory.**

Compared to a non-scalable implementation the scalable memories have significant area and device count overhead. The ensemble of point solutions is 4 times larger than the non-scalable memory and has 272K more devices. The reuse of point solutions method has less overhead: only 2.1 times larger and 68K more devices. Both the ensemble and reuse method contain many memory blocks created by the generators. The non-scalable solution has four 128Wx32b memories, and is extremely compact, but is not able to scale energy efficiently.

The energy dissipation per access of the three memory implementations are simulated at 1.5V using *nanosim*. When compared to a non-scalable design, the greatest impact on energy dissipation is seen for smaller FFT lengths. The scalable architecture eliminates the overhead of accessing large memories during 128-point FFT computation. Both the ensemble and reuse of point solutions methods achieve approximately 52% energy savings per memory access over a non-scalable solution.

**Table 4.3: Comparing different memory implementations**

| | | Non-scalable | Ensemble of Points | Reuse of Points |
|---|---|---|---|---|
| Area | | 0.45mm$^2$ | 1.8 mm$^2$ | 0.95 mm$^2$ |
| Transistor Count | | 186K | 458K | 254K |
| Energy/access | 1024pt | 252pJ | 217pJ | 167pJ |
| | 512pt | 252pJ | 159pJ | 142pJ |
| | 256pt | 253pJ | 138pJ | 128pJ |
| | 128pt | 252pJ | 119pJ | 118pJ |

Variable bit-precision of 8- and 16-bit precision is also enabled in the memory structures. Two 16-bit precision memories make up one 32-bit memory block and during 8-bit operation inputs and clocks to the unused memory blocks are gated. The Twiddle ROMs are also designed for variable bit-precision operation in a similar manner. Energy simulations show that variable bit-precision leads to a 57% energy savings at 8-bit precision operation over non-bit-precision scalable designs

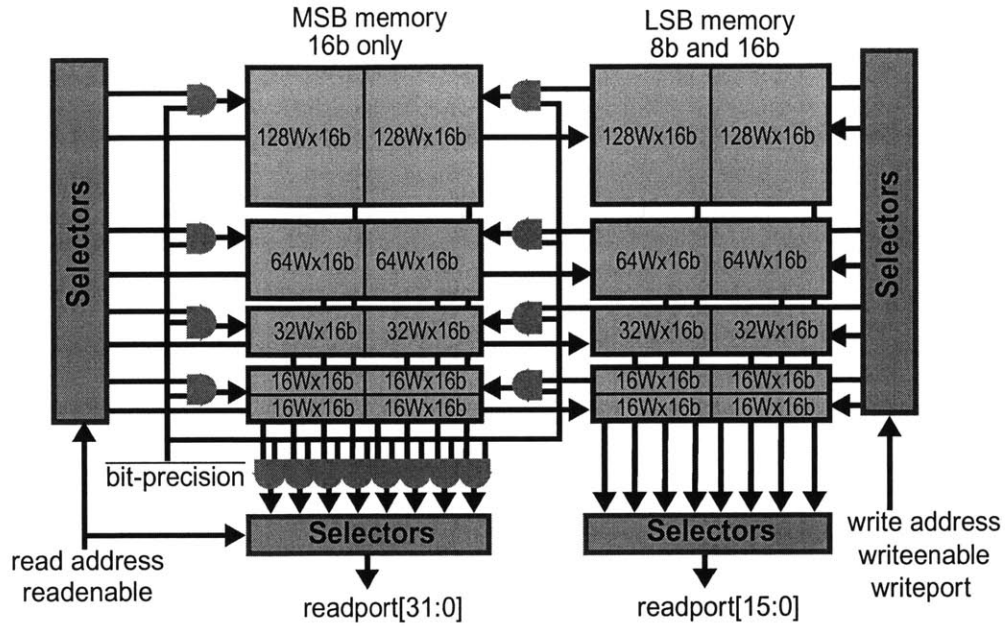


**Figure 4.11: 8- and 16-b scalable data memory. 8-bit precision is enabled with bit-precision = 1.**

.

## 4.5 Energy-scalable systems

System level scalability incorporates the scalability of the datapath and of the memory together with the control logic. The reuse of point solutions method is used in the design of the FFT for reasons of less area overhead, fewer transistors and lower simulated energy dissipation. In the reuse of point solutions FFT, each functional block is designed using the reuse method and brought together with appropriate control logic and an energy-scalable interface.

Two FFT architectures (non-scalable and reuse of points) are created and simulated using *nanosim* at 1.5V supply voltage. The non-scalable FFT length architecture is designed for the worst-case scenario (1024-point, 16-bit). The reuse of point solutions method has a variable bit-precision datapath, variable bit-precision and memory size data memory, and variable bit-precision ROM. The control logic for the both non-scalable and reuse implementations are designed to scale the number of butterflies with FFT length.

The energy-quality scalability characteristics of the non-scalable and scalable FFT are shown in Table 4.4, for 128-1024 point FFT lengths and for 8-bit and 16-bit precision computation. The energy simulations of a non-scalable FFT show that there is very little savings when computing an 8-bit FFT. Because the FFT is performed using two's complement arithmetic, there is switching energy dissipated by the MSB circuitry due to the sign-extension bits of two's complement arithmetic. If the FFT were implemented using sign-magnitude arithmetic, there would be additional energy savings for the 8-bit FFT even on a non-scalable architecture.

Therefore, the simulation results show that there is a definite advantage for an energy-scalable architecture using a reuse of point solution architecture. The scalable architecture is more energy-efficient for all but the high-quality point (1024 point, 16-bit). At the high quality point, there is no energy advantage of the scalable architecture, but rather a disadvantage due to the overhead of the scalability logic. However, the trade-off is at the low quality point (128 point, 8-bit) where the energy of the scalable implementation is 2.7x lower than the non-scalable implementation

**Table 4.4: Comparing a non-scalable RVFFT to the scalable RVFFT using reuse of point solutions method.**

| | | Non-scalable | | Reuse of Points | |
|---|---|---|---|---|---|
| Energy/FFT | FFT Length | 8-bit | 16-bit | 8-bit | 16-bit |
| | 1024-point | 1320nJ | 1448nJ | 575nJ | 1491nJ |
| | 512-point | 607nJ | 750nJ | 240nJ | 629nJ |
| | 256-point | 269nJ | 334nJ | 103nJ | 269nJ |
| | 128-point | 118nJ | 147nJ | 44nJ | 116nJ |

## 4.6 Design flow

Besides demonstrating an energy-scalable design methodology, the FFT processor design also showcases a design flow toolchain incorporating algorithm benchmarking, verilog architectural design and automatic layout generation using a standard cell library. Simulation and verification are done at multiple points and also across layers of implementation.

### 4.6.1 Algorithm benchmarking

Choices made at the algorithm level impacts the energy-efficiency of the entire system. Therefore, it is important to do preliminary algorithm benchmarking to evaluate and quantify energy dissipation for different algorithms in software. Evaluation of the benchmark in *MATLAB* is instrumental in many ways. *MATLAB* benchmarking facilitates an accurate transform of the FFT from floating point to a fixed point processor. First, the FFT is benchmarked using floating point arithmetic for a variety of inputs (sinusoidal wave, DC, vehicle sounds, etc.). Next, a fixed point methodology for *MATLAB* is created to emulate a fixed point processor. The methodology allows for testing different datapath bit-precisions and is used to design high level architectural blocks. A fixed point FFT is implemented using this methodology, and the accuracy of the FFT is verified with the floating point benchmark. Also, the *MATLAB* benchmark is instrumental in creating test vectors from the fixed point benchmark for cross layer verification from *C* code development all the way to hardware testing.

### 4.6.2 Verilog design

Structural verilog implementations of the FFT algorithm are created in order to evaluate different architectural choices for the datapath, memories, and control logic. In the datapath design, bit-precision scalable arithmetic units such as multipliers and adders are created. For the memory, the design uses variable-sized memories created through memory generators. All control logic is designed using structural verilog. A testbench is created which emulates the data and control inputs and outputs to the hardware. Functional verification of the verilog is performed by comparing FFT coefficients to the *MATLAB* fixed point benchmark. The resulting verilog benchmark creates test vectors for layout and hardware verification.

### 4.6.3 Automatic layout generation

A standard ASIC flow creates the layout of the FFT processor. The ASIC flow automatically generates layout using a 1.8V standard cell library and memory generators. The first step is to compile the structural verilog using Synopsys *Design Compiler* into RTL verilog. The RTL verilog describes the FFT using standard cell logic gates. Next, the RTL verilog is read into Cadence *Silicon Ensemble* along with layer specifications of the stan-

dard cells. *Silicon Ensemble* performs place-and-route of the layout with standard cells. The layout is then imported to the Cadence layout viewer for simulation and verification.

A bottom up approach is used in the design of the FFT, where several blocks (scalable datapath, scalable register file memory, control logic, scalable ROMs) are each synthesized independently and verified using test vectors from the verilog benchmark. Top level routing is performed using *Silicon Ensemble*, which includes routing all functional blocks and pad drivers. The top level design includes power and ground routing, clock drivers and floorplanning.

More details about the design flow are found in [46][75].

### 4.6.4 Functional and power verification

In order to verify the functional correctness and to do an accurate power analysis of the FFT ASIC, first, the layout is streamed into the Cadence layout editor. All transistors and their connectivity are extracted into a netlist. Next, Synopsys *nanosim* simulator and power estimator is used. This tool is used to compare the energy-efficiency of different scalable designs. By incorporating test vectors from the verilog testbench, the functionality of the entire FFT algorithm for different types of input signals is checked.

### 4.6.5 Hardware testing

A printed circuit board (PCB) tests the functionality of the fabricated FFT chip. (Figure 4.12). The PCB allows an interface with a pattern generator and a logic analyzer. The pattern generator is programmed to emulate the original verilog testbench, and the logic generator captures the output data. The output data is compared to the *MATLAB* and verilog benchmark for functionality.

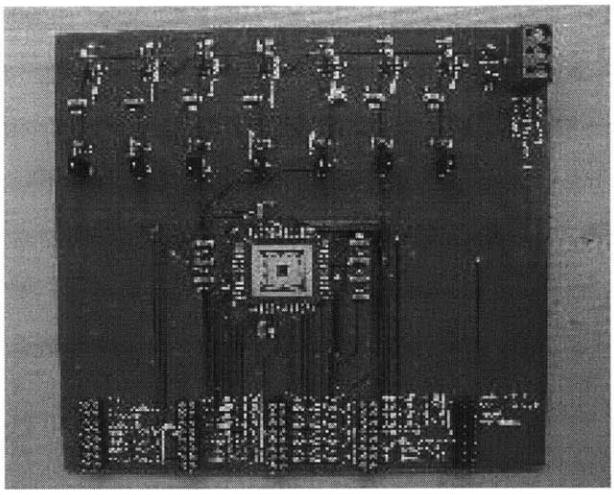

**Figure 4.12: PCB used for system-level performance and functional verification. The PCB interfaces with a pattern generator and a logic analyzer.**

## 4.7 Performance measurements

The energy-aware FFT is implemented in a 0.18 μm process. Figure 4.13 shows a die photograph of the scalable FFT chip. The chip's functional blocks (data memory, butterfly, control, and ROMs) are clearly delineated. The FFT system is fully verified up to 512 point, for 8 and 16 bit operation at 1.5V and for clock speed of 4.8 MHz.

Table 4.5 shows the measured energy dissipation of the FFT. The measured energy dissipation verifies the architecture's ability to scale energy and quality. The energy at the low quality point (128-point, 8b) is 12 times lower than the energy at the high quality point (512-point, 16b).

**Table 4.5: Measured energy dissipation from the FFT ASIC.**

| | 8-bit | 16-bit |
|---|---|---|
| 128 pt. | 46nJ | 81nJ |
| 256 pt. | 121nJ | 216nJ |
| 512 pt. | 304nJ | 564nJ |

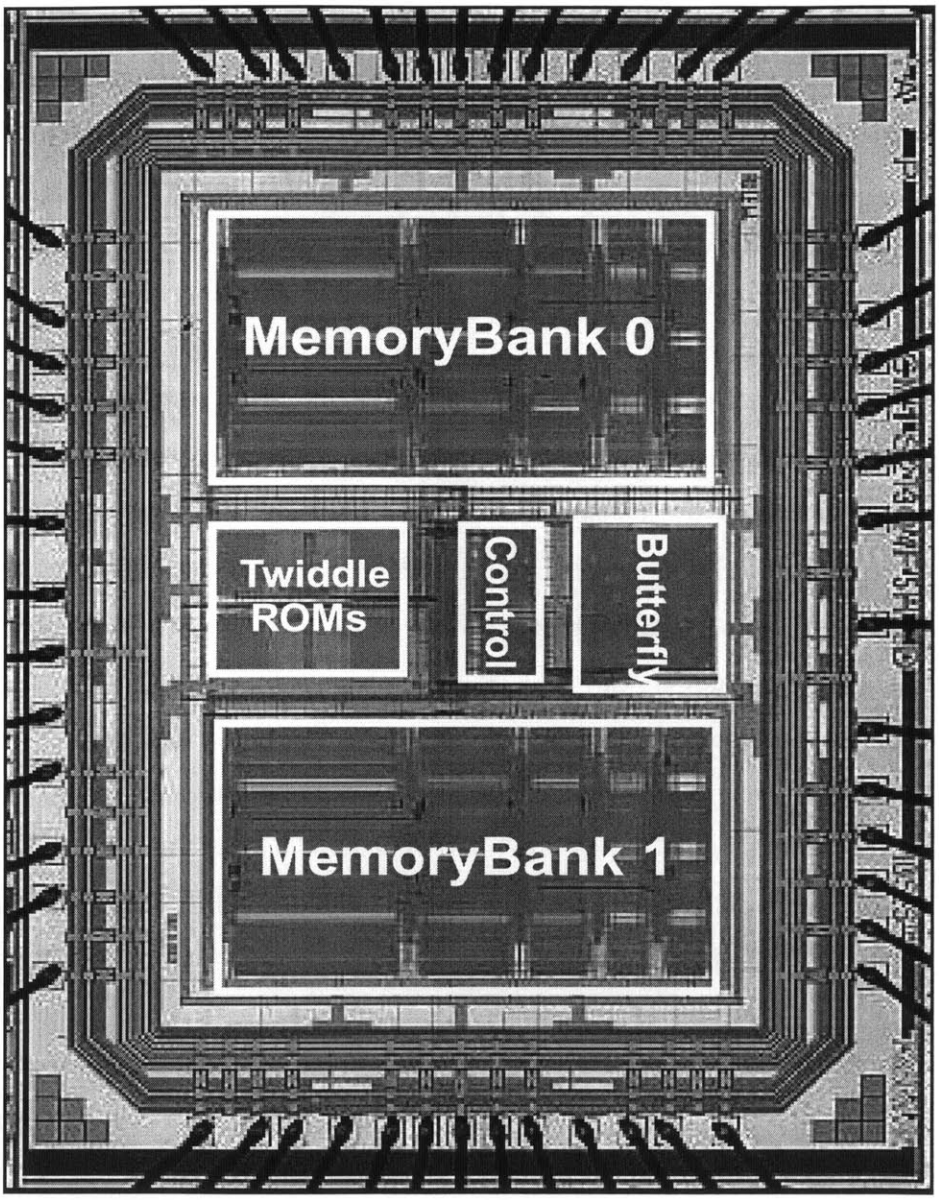


**Figure 4.13: Die photograph of the 128 to 512 point, 8 and 16 bit scalable FFT chip.**

The FFT ASIC energy measurements are compared to the energy simulated in *nanosim* from Table 4.4. The simulated energy dissipation is lower than the measured energy for the 8-bit FFT and higher than the measured energy for the 16-bit FFT at 1.5V. In general there is less than 30% energy difference between measured and simulated energy.

The FFT ASIC operates down to 1V before failing. In order to minimize energy, a new FFT processor design is needed that operates at supply voltages below 1V. Research in the next two chapters focuses on estimating the optimal supply voltage for minimal energy dissipation and circuits that allow supply voltage scaling below the threshold voltage of the devices.

## 4.8 Conclusions

This chapter introduces the concept of energy-aware architectures to enable Energy-Quality scalable algorithms and systems. The energy-awareness of a system is improved by engineering E-Q scalable hooks into the architecture and hardware.

First, the FFT block diagram and high-level architecture are described. System-level details and control signals for the FFT are given.

In this thesis, two energy-aware architectural approaches are evaluated. In the ensemble of point solutions approach, a highly optimized point solution is designed for each scenario. Then high-level routing and control route the inputs to the appropriate point solution, while data and input gating prevent switching in the unused blocks.

The second approach is the reuse of point solutions method. In this approach, more efficient energy-scaling is enabled by recognizing that many of the point solutions share common hardware and functions. For example, there are four scenarios for FFT length: 128, 256, 512, and 1024-point FFTs. Since the underlying hardware is the same as FFT length is varied, the hardware is reused for all scenarios and thus is energy-aware with very little area and transistor count overhead.

The two architectural approaches are evaluated for the FFT datapath and memory. These implementations are compared to a non-scalable approach which only operates for the worst-case. The reuse of point solutions method proves to have the smaller amount of area and transistor count overhead, while still providing energy-scalability.

The FFT is implemented using the reuse of point solutions method and is fabricated in a 0.18μm process. An ASIC design flow is used and the architecture is verified through a fabricated chip. The FFT processor functions for 128, 256, and 512 points and for 8- and 16-bit processing at 1.5V with a clock frequency of 4.8MHz.

# Chapter 5

# Energy-performance contours

Supply and threshold voltage scaling in the strong inversion region is extensively studied, and this work extends the analysis into the subthreshold region. In literature, different metrics are used to study supply and threshold voltage scaling such as energy, delay, power, energy-delay product (EDP), and power-delay product (PDP) [10][20]. In this work, the two metrics of energy and performance are focused on for submicron circuits using a 0.18μm process. A simple characterization circuit is introduced and energy and performance simulations of the circuit operating in the region of $V_{DD}$=0.1-1V and $V_{th}$=0-0.8V provide many insights. This circuit also enables analysis of energy dissipation for variable activity factor. One conclusion from these simulations suggests that that performance in the subthreshold region is adequate for microsensor applications. Also, these results give intuition about trade-offs between energy and performance, and the optimal ($V_{DD}$, $V_{th}$) operating point that minimizes energy dissipation.

## 5.1 Active and leakage energy models

The active energy model is given by

$$E = NC_{avg}V_{DD}^2 \tag{5.1}$$

where N is the number of clock cycles, $C_{avg}$ is the average switched capacitance of the circuit, and $V_{DD}$ is the supply voltage. Reducing the supply voltage gives quadratic energy savings but leads to a reduction in clock frequency unless the threshold voltage is reduced as well. However, lower threshold voltage devices have higher static power dissipation due to large leakage currents.

A simplified subthreshold transistor model for an NMOS transistor is given by

$$I_{ds} = I_o \cdot e^{\frac{V_{gs} - V_{th0} - (\gamma' \cdot V_{sb}) + \eta \cdot V_{ds}}{nV_T}} \cdot \left(1 - e^{\frac{-V_{ds}}{V_T}}\right) \tag{5.2}$$

where $V_{gs}$, $V_{ds}$ and $V_{sb}$ are the gate-to-source, drain-to-source and source-to-bulk voltages applied to the transistor, $V_{th0}$ is the zero-body bias threshold voltage, and $V_T$ is the thermal voltage. The body effect due to source to bulk voltage is modelled by the term $\gamma' V_{sb}$ and DIBL is modelled by the term $\eta V_{ds}$. $I_o$ is a scaling factor which is defined as

$$I_o = \mu_o \cdot C_{ox} \cdot \frac{W}{L} \cdot V_T^2 \cdot e^{1.8} \tag{5.3}$$

where $\mu_o$, and $C_{ox}$ are technology dependent parameters and W and L, represent the width and length of the device respectively [14][58].

In the subthreshold region, process variations play a huge part in affecting the functionality and operating point for a circuit. The process corners are specified by the foundry to give a $3\sigma$ spread of the variations. These include variations due to oxide thickness, threshold voltage variations, and various junction capacitance levels which affect the drain current model, performance, and yield. Particularly in the subthreshold region, these variations cause changes in drain current by orders of magnitude.

Much research is dedicated to alleviate the effects of process variations. One method is to use adaptive body biasing, where the backgates of the PMOS and NMOS are controlled independently to counterbalance any die-to-die process variations. Adaptive body biasing allows the processor to adapt for delay variations [41][68].

## 5.2 Energy-performance contours for optimal $V_{DD}$-$V_{th}$ operation

Exploring the energy and performance for a wide range of voltage supply ($V_{DD}$) and for threshold voltage ($V_{th}$) provides understanding into trade-offs between low power and high performance. In previous research, contours of energy-delay product (EDP) are plotted to give the optimal operating point where EDP is minimized in the strong inversion region [20]. In this work, we explicitly evaluate the two metrics, energy and performance, where the amount of "work" is kept constant. These are important metrics for energy-con-

strained systems such as wireless microsensors, where each task must be run as energy-efficiently as possible for a desired performance. The supply voltage, threshold voltage, and clock frequency are set to minimize energy dissipation.

Energy and performance simulations of a variable activity factor ring oscillator characterization circuit for a 0.18-μm process help to build intuition about optimal $V_{DD}$ and $V_{th}$ scaling. This circuit shows the limits of supply voltage scaling and the benefits of threshold voltage scaling. The simulation results are extrapolated to model the energy-performance of larger systems.

Figure 5.1 shows a schematic of the variable activity factor circuit. The circuit consists of an 11-stage 2-input nand ring oscillator (RO). The ring oscillator's performance is the frequency of the oscillator. The stage depth of the RO is chosen to emulate the delay through a microprocessor pipeline with logic depth of 11. Variable activity factor is enabled by having 9 additional delay chains which are driven by the ring oscillator at the same frequency. The delay chains are enabled/disabled using Selector inputs to vary activity factor. When all of the chains are activated the activity factor is 1, since all nodes in the ring oscillator switch once during the clock period. Activity factor is defined as

$$\alpha_{0 \rightarrow 1} = \frac{n(N)}{N} \tag{5.4}$$

where n(N) is the total number of nodes switching from low to high, and N is the total number of nodes. A variable activity factor circuit is highly convenient since in typical logic gates the activity factor is rarely equal to 1. As activity factor decreases, the switching energy also decreases, and the ratio of leakage to switching energy increases. For example, by activating Select0-Select4 and deselecting Select5-Select9, the activity factor

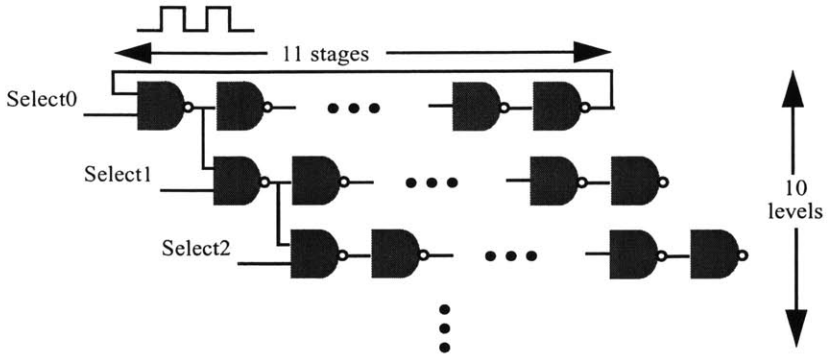


**Figure 5.1: Circuit to characterize energy and performance in subthreshold region for variable activity factor. The results from this circuit are extrapolated for larger circuits.**

is 0.5, because only half of the levels are switching.

Simulations of the energy-performance variable activity factor ring oscillator circuit are done using BSIM3 models of the TSMC 0.18μm process [33]. In these simulations, the voltage supply is varied from 0.1 to 1V and the threshold voltage for both NMOS and PMOS is varied from 0 to 0.8V. Threshold voltage is varied by changing the BSIM parameter VTH0. Minimum widths are used for all devices.

Figure 5.2 shows constant normalized energy curves for $\alpha$=1. The minimum energy point, given by the circle is at ($V_{DD}$, $V_{th}$) = (130,370) mV. At the minimum energy point, the energy per clock cycle dissipated by the circuit is 0.19 fJ/gate, and the ring oscillator clock frequency is 75 kHz. Each energy curve encircling the minimum point is a contour indicating higher energy dissipation. The contour labels show the increase in energy dissipation over the energy at the minimum point.

These energy contours show the relationship between leakage energy and active energy. For a fixed threshold voltage, as the supply voltage drops from 1V to 100mV, both the energy and the clock frequency decrease. The switching energy decreases quadratically until the lowering clock frequencies cause leakage energy to accumulate and increase over the switching energy. The increase in leakage energy causes the total energy to rise in the subthreshold region. For a fixed supply voltage, as the threshold voltage decreases, both energy and clock frequency of the benchmark circuit increase. This increase is due to the exponential dependence of subthreshold leakage current on the threshold voltage. However, for high threshold voltage, the reduced clock frequencies also

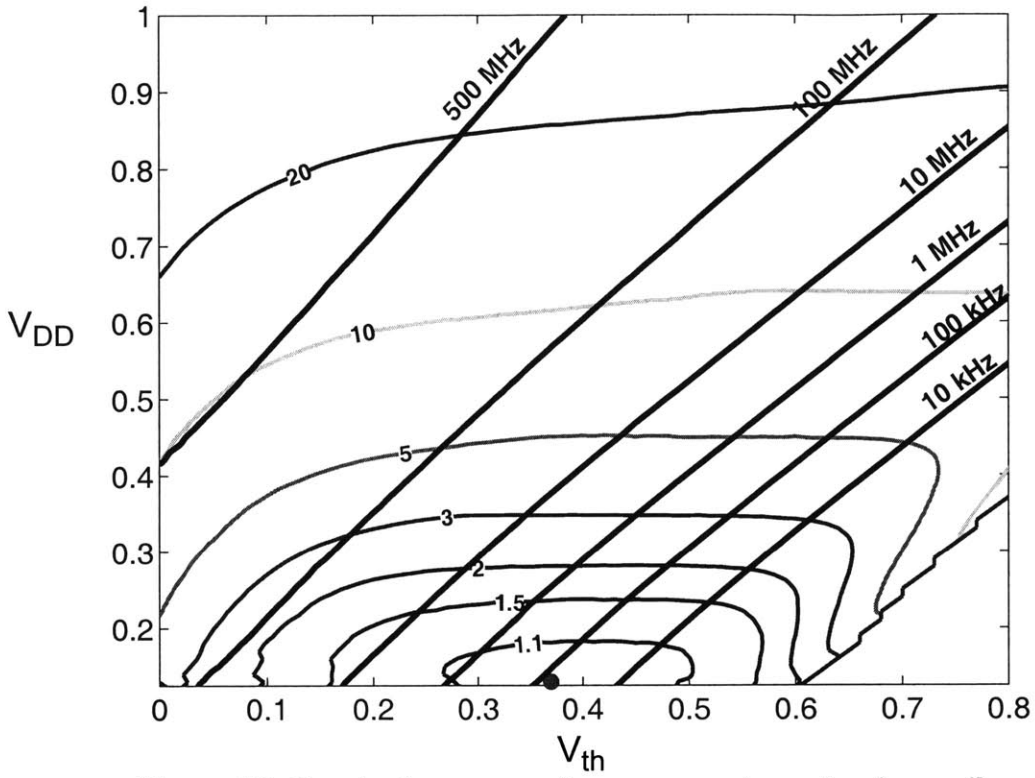


**Figure 5.2: Constant energy-performance contours for ring oscillator circuit with activity factor, $\alpha = 1$.**

cause the leakage energy to increase. At deep subthreshold where $V_{th} >> V_{DD}$, the large amount of leakage energy causes an increase in total energy, thus for a fixed supply voltage there is an optimal threshold voltage that minimizes energy dissipation. The combination of these two effects leads to the energy contour shown in Figure 5.2.

Superimposed on the energy contours are contours of constant performance, given by the diagonal straight lines. The performance of the RO ranges from 10kHz-500MHz. These graphs show insight into performance limits of the devices and their relation to ($V_{DD}$,$V_{th}$). Applications requiring performance less than 10MHz would operate in the subthreshold region for minimal energy dissipation. Those requiring more than 10MHz should operate with $V_{DD} > V_{th}$.

Next, by using the selector inputs, the activity factor is decreased. Figure 5.3 show the effect of the activity factor ($\alpha$=0.5 and $\alpha$=0.1) on the constant energy curves. These curves show that both supply voltage and threshold voltage for the minimum energy point

increase for lower activity factors. Decreased activity factor leads to an increase in the proportion of energy attributed to leakage currents. Therefore, it becomes necessary to operate at a higher threshold voltage to reduce the effect of leakage and at a higher supply voltage to obtain improved performance for higher $V_{th}$.

The optimal ($V_{DD}$,$V_{th}$) of these curves the point where a performance contour is tangent to an energy contour. This is the ($V_{DD}$,$V_{th}$) which gives the minimum energy dissipated for a given clock frequency. The optimal point of operation for this circuit is given by the dotted line in Figure 5.3b.

The optimal operating points ($V_{DD}$, $V_{th}$) for a given performance curve is needed to understand the limits and advantages of scaling supply voltage and threshold voltage scaling. For example, in the case with no latency requirement, there is a limit to the lowest frequency given by the curves. Scaling the clock frequency below 110 kHz is not advisable because the energy dissipation due to leakage increases as clock frequencies decrease. Therefore the best strategy for a long latency requirement is to operate at 110kHz and then gate the power supply of the circuit.

These results are used to extrapolate the results to larger systems, such as the FFT. The activity factor of the system determines the ratio of the active energy to the leakage energy. If the activity factor of a larger system is estimated, then these simple benchmark results are extrapolate to approximate energy-performance contours for that system. To investigate these claims, the energy-performance contours of an adder is examined, and then the energy-performance of the FFT is analyzed to find the optimal ($V_{DD}$, $V_{th}$) that gives minimum energy dissipation.

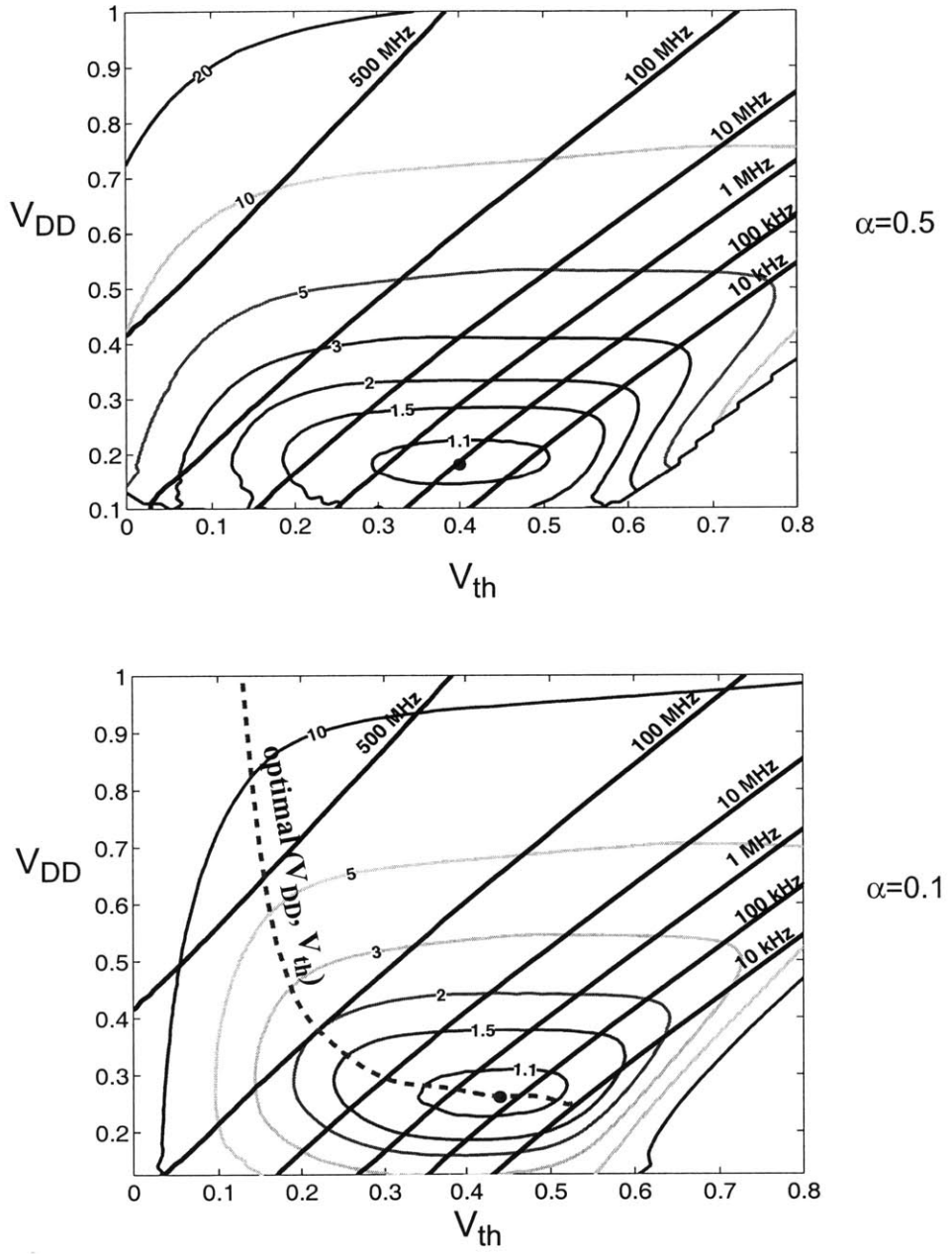


**Figure 5.3: Constant energy-performance contours for decreasing activity factor $\alpha$=0.5 and $\alpha$=0.1.**

## 5.3 System-level energy-performance contours

The results and conclusions from the benchmark circuit leads us to believe that larger systems will have similar energy-performance contours. We will look at the simulated energy-performance contours of larger circuits, such as a 16-bit ripple carry adder.

In a system-level analysis that contains > 100k devices, it is not possible to extract the energy-performance curves from a single *hspice* simulation. However, it is possible to estimate the *hspice* energy-performance curves, if the following parameters about the circuit are known for the nominal voltage supply: average switched capacitance, latency of the critical path, and simulated leakage current. These parameters coupled with the results of the simulations of the benchmark circuits can be used to estimate the energy-performance contours of the adder. This estimate is compared to the *hspice* simulated results. The same estimation technique is then used to estimate the energy-performance contours of the RVFFT.

First, the energy-performance curves for a 16-bit ripple-carry adder are simulated in *hspice* for $V_{DD}$=0-600mV and for $V_{th}$=0-600mV. The performance is the worst case delay through the adder. The adder is simulated for a variety of input vectors to find the average energy dissipation over the entire region. The fully simulated energy-performance curves from the *hspice* simulation are shown in Figure 5.4.

The same energy-performance contours for the adder are estimated by using *hspice* simulations for $V_{DD}$=1V and the *hspice* simulations of the benchmark circuit:

Given:

adder : $E_{avg}$ ($V_{DD}$=1V), $I_{leak}$ ($V_{DD}$=1V), $T_{lat}$ ($V_{DD}$=1V)

benchmark : $I_{leak}(0 < V_{DD} < 1V , 0 < V_{th} < 1V)$

$T_{lat}(0 < V_{DD} < 1V , 0 < V_{th} < 1V)$

Find: adder: $E_{avg} (0 < V_{DD} < 1V , 0 < V_{th} < 1V)$ and

$T_{lat} (0 < V_{DD} < 1V , 0 < V_{th} < 1V)$

Where the $E_{avg}$ is the average energy, $T_{lat}$ is the critical path latency, and $I_{leak}$ is the simulated leakage current. The total energy is calculated to be

$$E_{total} = C_{avg}V_{DD}^2 + V_{DD}*I_{leak}*T_{lat} \quad (5.5)$$

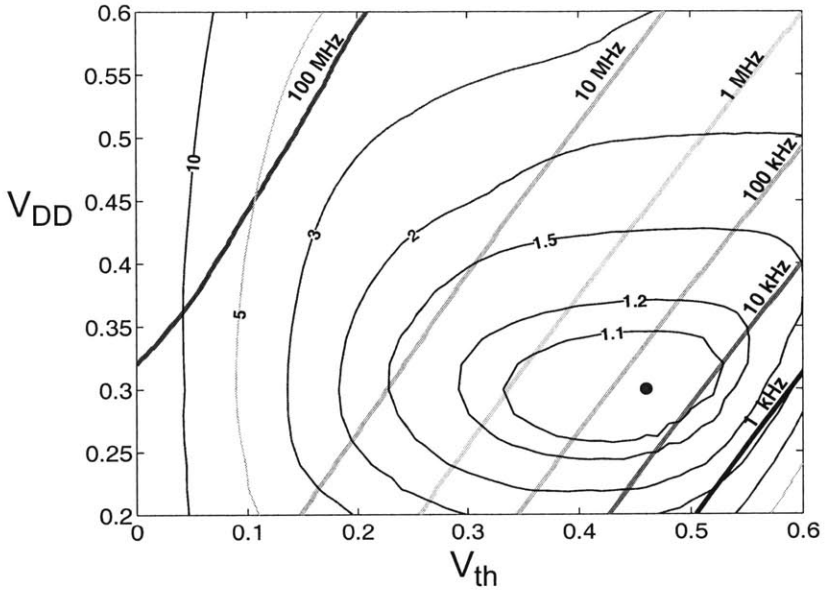


**Figure 5.4: Energy-performance curves for a 16-bit ripple carry adder from an *hspice* simulation.**

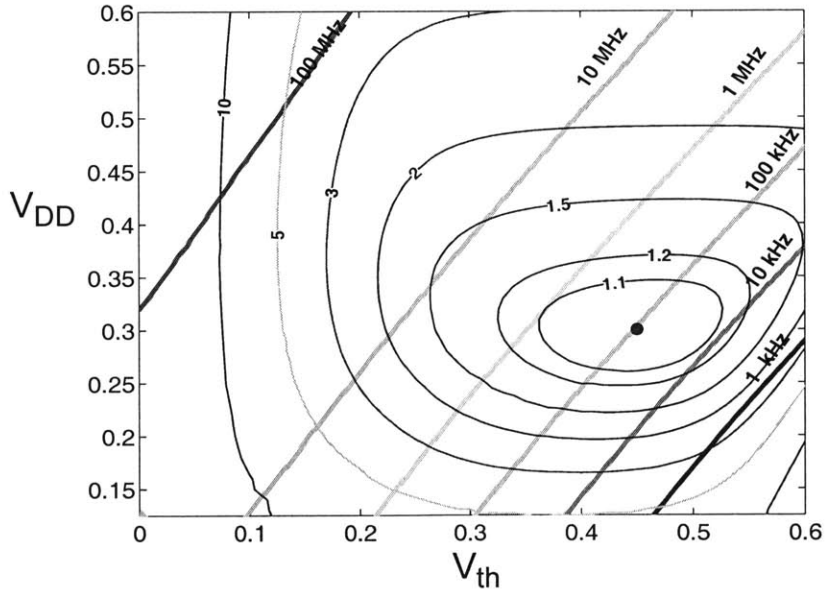


**Figure 5.5: Estimated energy-performance curves for the adder by extrapolating results from the benchmark circuit.**

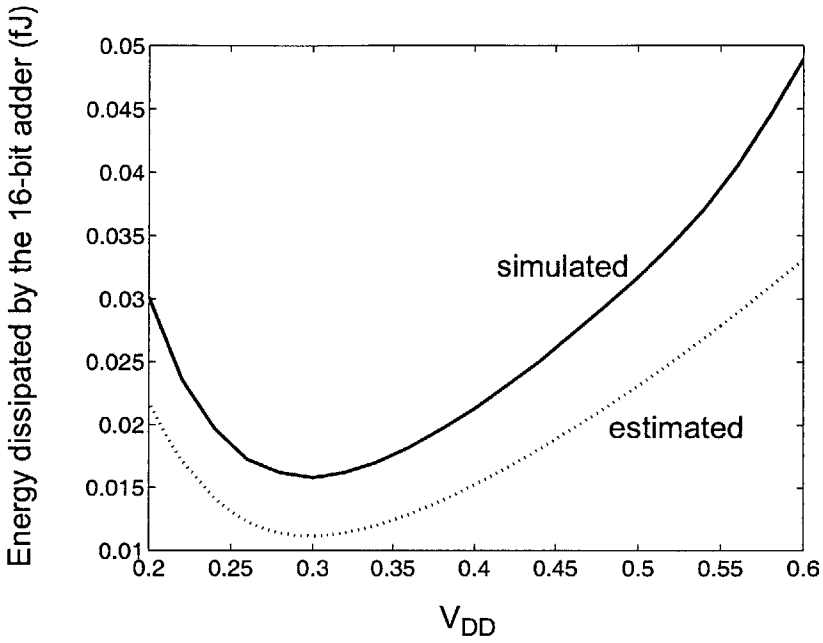


**Figure 5.6: The energy curve for $V_{th}$ fixed at 450mV shows the difference between the simulated energy and the estimated energy to be 7.6 fJ.**

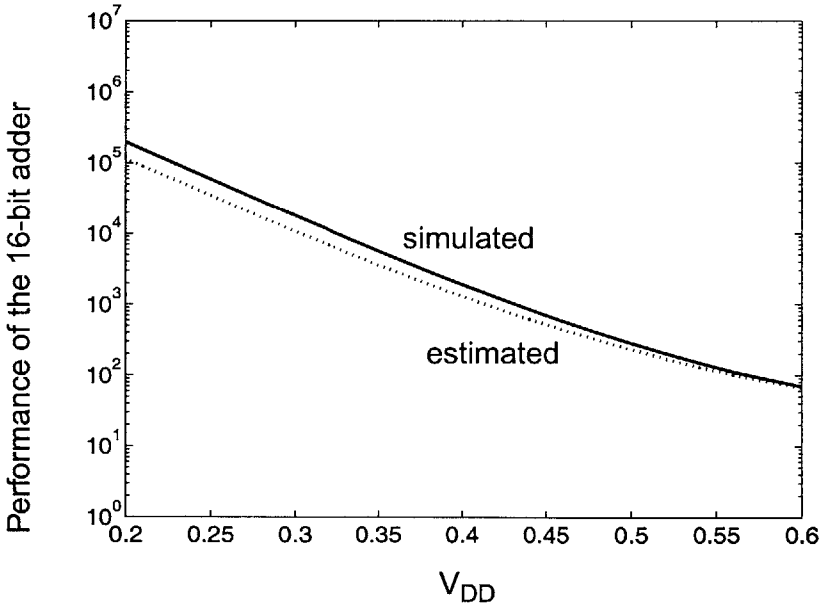


**Figure 5.7: The performance curve of the adder for $V_{th}$ fixed at 450mV shows there is up to 42% difference between the simulated energy and the estimated energy.**

The average switched capacitance ($C_{avg}$) is calculated from the average switching energy, $E_{avg}$. The average switching energy is found by using a *nanosim* simulation of the adder for a long sequence of input vectors. *Nanosim* is a circuit simulator that provides power estimation and timing verification of large scale circuits. At $V_{DD}$=1V, the leakage energy is small compared to the switching energy at 1V, total energy has very little contribution from the leakage current. The *nanosim* simulation gives the average switched capactiance to be 91.5 fF.

An *hspice* simulation of the adder is used to estimate the standby leakage current for $V_{DD}$=1V. This value is compared to the leakage current of the benchmark circuit to give a scaling factor. The leakage energy of the benchmark circuit is extrapolated to estimate the adder's standby leakage for all ($V_{DD}$, $V_{th}$). The performance or the worst case latency through the adder is also simulated in *hspice* at 1V, and compared to the latency of the benchmark circuit at 1V. The latency of the adder for all ($V_{DD}$, $V_{th}$) is estimated by uniformly scaling up the latency of the benchmark circuit. Figure 5.5 shows the energy-performance contours extrapolated from the benchmark circuit results.

Figure 5.6 shows the energy both simulated and estimated when we fix the threshold voltage to be 450mV, the nominal voltage of a 0.18μm process given by the ITRS roadmap [28]. Comparing the estimate curve versus the simulated curve, there is up to 76 fJ or 30% estimation error for large $V_{DD}$. At high voltages, the estimation error is due to incorrectly estimating the switched capacitance of the adder using *nanosim*. At low voltages, the estimation error comes from incorrectly estimating the delay. Figure 5.8 compares the simulated and estimated performance or clock speed of the adder.

Next, these same estimation techniques are used to generate the energy-performance curves for large systems, such as the 1024-point FFT. Figure 5.8 shows the estimated energy-performance curves for the FFT. Also plotted is the curve for optimal ($V_{DD}$,$V_{th}$) for the FFT. In general, these curves indicate that to minimize energy dissipation of the FFT, for a given $V_{DD}$ it is best to operate at voltages lower than $V_{th}$ for minimal energy dissipation. For wireless sensor networks, whose performance requirements are relaxed, this indicates that subthreshold operation of the FFT is the most energy-efficient solution.

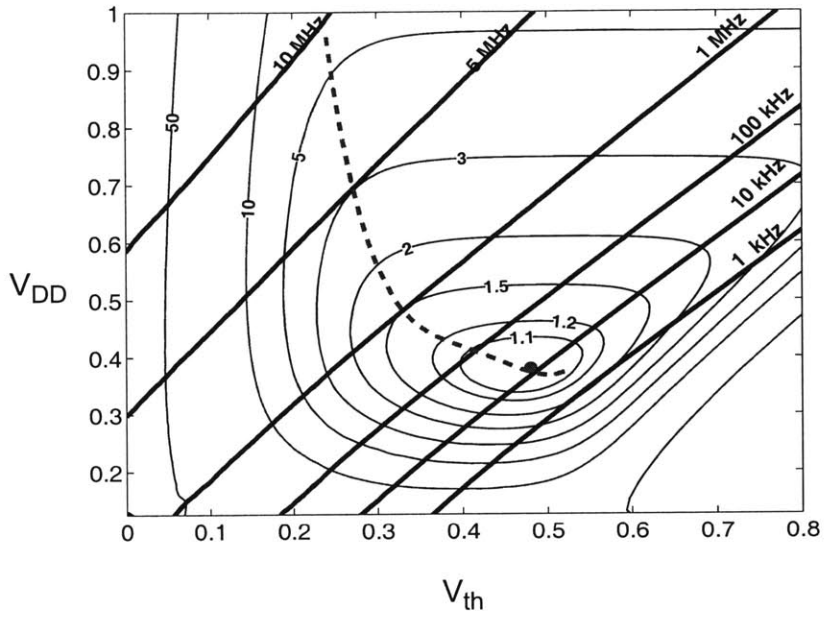


**Figure 5.8: The estimated energy-performance curve for the FFT.**

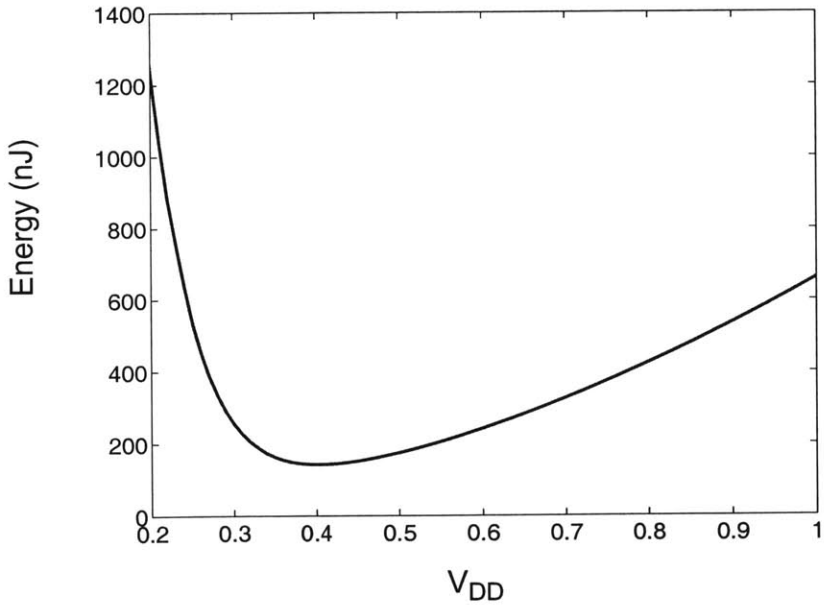


**Figure 5.9: The estimated energy dissipation of the FFT as a function of $V_{DD}$ for a fixed threshold voltage of 450 mV.**

Figure 5.9 shows the energy curve for a fixed threshold voltage of 450mV. This curve indicates that the voltage supply that minimizes energy dissipation is 400 mV, and the energy dissipated by the FFT is approximately 142 nJ. However, since our goal is to study the optimal voltage-frequency operating point, our design will try to achieve minimal voltage supply operation using circuits to function in simulation at 100mV.

These estimates are highly variable depending on the estimated leakage current, average switch capacitance and clock frequencies. However, they do provide motivation towards using subthreshold circuit design to minimize energy dissipation even in large systems.

## 5.4 Summary

Because the performance requirements of wireless sensor networks is around the kHz range, we explore using subthreshold circuits to operate the microsensor node hardware. In this chapter, a simplified model of the subthreshold leakage current is introduced.

The main contribution of this chapter is the study of energy-performance curves for a benchmark circuit operating at $V_{DD}$=0.1-1V and $V_{th}$=0-0.8V. This circuit has variable activity factor which is useful to see the effects of leakage currents on energy and performance in the subthreshold region. This circuit provided valuable insights into subthreshold circuit operation and is used to estimate the optimal $V_{DD}$-$V_{th}$ operating point for minimal energy dissipation at a given clock frequency.

The energy and performance results from the benchmark is extrapolated to estimate the energy-performance curves for much larger circuit blocks. This is demonstrated for a 16-bit adder, and the FFT. The results show that the optimal $V_{DD}$ of the FFT is around 400 mV given a 0.18μm process with nominal threshold voltage around 450mV. This motivates the need for a subthreshold solution for the FFT processor.

# Chapter 6

# Subthreshold circuit design

The energy-scalable FFT ASIC implementation that uses an off-the-shelf standard-cell logic library and memory, only operates down to 1V. The simulated energy-performance contours of the variable activity factor benchmark circuit indicate that operation far below 1V is necessary to minimize energy dissipation. For the FFT, the estimated optimal point is below the threshold voltage, thus motivating the need for subthreshold circuit design methodologies. Although, the optimal voltage supply for a 0.18μm process is estimated to be near 400 mV, the circuits in the custom FFT processor target functionality at a supply voltages as low as 100mV in simulation. The fabricated FFT processor uses a standard CMOS 0.18μm process and operates down to 180mV. The nominal threshold voltage of a 0.18μm process is around 450mV [28].

## 6.1 Previous work

Subthreshold circuit design involves scaling the supply voltage below the threshold voltage where load capacitances are charged/discharged by subthreshold leakage currents. Leakage currents are orders of magnitude lower than drain currents in the strong inversion region, so there is a significant limit on the maximum performance of subthreshold circuits. Traditionally, subthreshold circuits are used for applications which require ultra-low power dissipation and relaxed circuit performance. For example, subthreshold logic circuits have long been used in watch technology since the 1970's, where the watch microprocessor dissipated microwatts of power at 1V operation by relaxing the clock frequency [71][73]. Due to the low performance of subthreshold circuits, their use in applications were limited to timepieces or hearing aid technology [9][32]. Today, the growth in portable device technology and improvements in process technology is spawning new research initiatives into ultra-low voltage operation.

Minimum-voltage supply operation mainly depends on relative strengths of the NMOS and PMOS transistors. For a simple inverter, if the NMOS leakage and process variations cause uncertainty in minimum-voltage operation [49]. Therefore, most research efforts are focusing on low-voltage operation of logic circuits by using either process technologies that are optimized for low-voltage operation or n-well and p-well biasing. An encoder/decoder is designed that is able to operate at 200mV power supplies for a process specially designed with 0V threshold devices [8]. Applying voltages to the n-well or p-well allows for fine-grain threshold voltage control. By using p-well and n-well biasing techniques a multiply-accumulate (MAC) implementation is able to operate as low as 175mV [30]. Also, a low-power FFT processor fabricated with a 0.7μm process, uses 0.5V forward biasing on the n-well to demonstrate 1.1V operation [4]. However, very little previous work addresses minimum-voltage supply operation for a standard-logic process without threshold voltage manipulation. In this work, we recognize that the minimum-voltage supply operation also largely depends on transistor sizing, subthreshold slope, temperature, and process variation effects. This work emphasizes using standard CMOS technology to achieve subthreshold operation.

Some previous work examines different logic families for operation at ultra-low voltages. It has been suggested that psuedo-NMOS (PNMOS) circuits demonstrate better power-delay product in simulation [32]. This leads to overall faster clock speeds for PNMOS circuits. However, because the clock frequencies decrease exponentially with supply voltage, the total accumulated static leakage energy of idle PNMOS circuits is higher than their CMOS counterparts for deep subthreshold operation. Therefore when energy is a metric, not power-delay product, PNMOS circuits are not desirable to minimize energy. Similarly, subthreshold Dynamic logic is suggested for subthreshold operation. For strong inversion circuits, dynamic logic provides high-speed operation [63]. However, in subthreshold, there are many disadvantages of using dynamic logic. At a low supply voltage, there is only a small amount of charge stored on the dynamic node. The dynamic node becomes highly susceptible to noise and idle leakage currents. Other circuit families such as variable-threshold voltage subthreshold CMOS and dynamic threshold voltage CMOS are proposed, where the threshold voltage is changed through p-substrate and n-well control to combat temperature and process variation effects on subthreshold

circuits [64]. This work looks at conventional CMOS circuits and the effects of subthreshold operation on functionality of logic circuits.

Ultra-low voltage memory design is a fairly unexplored topic. As memories begin to dominate the die area of the microprocessor, they contribute to most of the performance limitations and power dissipation of the entire chip. Memory designers are area-constrained to small memory-bit cells with fast read and write accesses. Sub-1V memory designers face problems such as bitline leakage, cell stability and low-voltage sense-amplifier design. As voltages scale down to subthreshold operation, all of these issues are magnified. This thesis looks at these effects on various memory designs and proposes subthreshold memory design techniques for minimum-voltage operation.

## 6.2 Low-voltage metrics

In this section, a new design methodology for subthreshold logic circuits is introduced. New low-voltage metrics are necessary to understand how circuits operate in deep subthreshold. These metrics are used to characterize logic circuits and memories for low supply voltage operation.

In the subthreshold region an important metric of operation is the $I_{on}/I_{off}$ ratio. The $I_{on}/I_{off}$ metric is used to indicate if a logic gate will function properly. $I_{on}$ is defined to be the drive current of the devices and decreases exponentially as supply voltage is lowered. $I_{off}$, which is defined as the idle current of the device. For example, if a node is driven high, then the $I_{on}$ is the current from $V_{DD}$ to the node, and $I_{off}$ is the leakage current from the node to ground. As supply voltage decreases, the $I_{on}$ approaches $I_{off}$.

In strong inversion, for a minimum sized NMOS device, usually the PMOS device is sized 3 times larger for equal rise/fall times and for a switching threshold at $V_{DD}/2$. Figure 6.1a shows the Voltage-Transfer-Curve (VTC) for a 1V inverter sized where $\beta=W_p/W_n=3$. Dropping to 100mV voltage supply, in order that the switching threshold be at 50mV, $\beta$ increases to 10 as seen in Figure 6.1b, because at lower voltages the NMOS has a faster subthreshold slope than the PMOS. At low voltages, the NMOS is more leaky than the PMOS. This curve also shows that going to 100mV supplies leads to lower gain in the

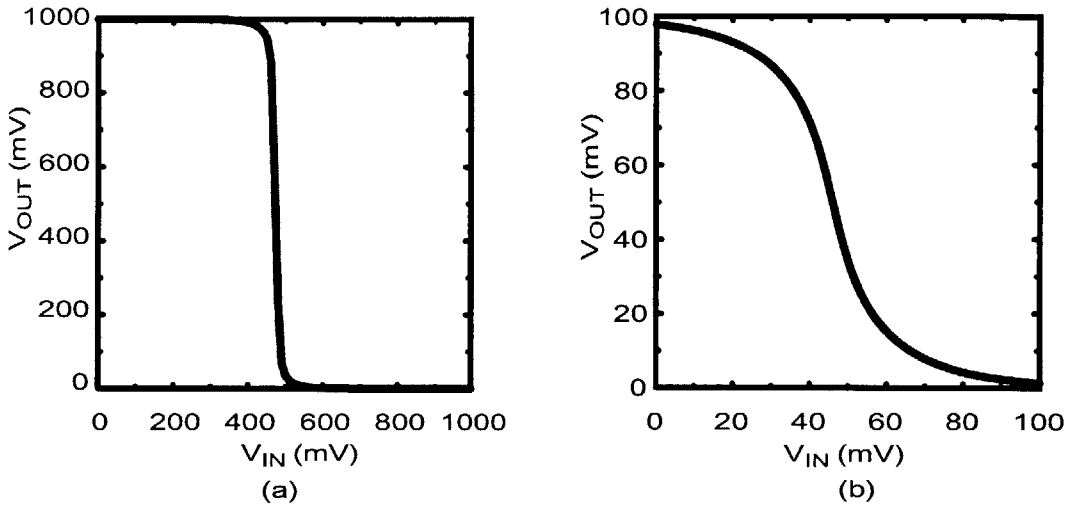


**Figure 6.1: The Voltage Transfer Curves (VTC) for an inverter at (a) 1V and at (b)100mV. The β at 1V operation is 3 and the β at 100mV is 10 for operation at $V_{DD}/2$ switching threshold.**

VTC. Therefore, because the devices act as a resistive divider, the output swing is not fully rail-to-rail.

The new sizing methodology introduced in this work focuses on sizing for minimum-voltage supply while factoring in the effects of process variations. Analysis the PMOS width of a CMOS inverter with a minimum-sized NMOS device is shown. Conventionally, the PMOS width is sized approximately 3x that of the NMOS device, for $V_{DD}/2$ switching threshold and for balanced rise and fall times. In general, the PMOS is sized so that the drive strength is large enough to pull-up the output node to a desired output-high voltage level. As the supply voltage decrease, the PMOS size is analyzed for two cases and simulations assuming a 10-90% inverter output-swing.

Case 1: For the case when '0' is applied to the input, then the PMOS is sized to pull-up the output node to '1'. However, as the voltage supply of the inverter decreases, the leakage current through the inverter begins to approach the drive current. This results in a decreasing $I_{on}/I_{off}$, and a resistive divider that causes the output node to drop below $V_{DD}$. Therefore, in order to drive the output to '1', the PMOS size is increased. Figure 6.2 shows $W_p(min)$ as a function of supply voltage ($V_{DD}$) which is the minimum allowable PMOS width that still drives the output to 90% of $V_{DD}$. The figure shows that for $V_{DD}$<150mV, the PMOS width increases with lowering $V_{DD}$.

Case 2: For the case when '1' is applied to the input, there is a maximum allowable PMOS size. A large $W_p$ cause large PMOS leakage currents which can approach the drive current of the NMOS. Therefore there is a maximum bound on $W_p$, that still allows the output to be driven low. Figure 6.2 also shows $W_p$(max) where the output is 10% of $V_{DD}$.

The intersection of these two curves shows the minimum voltage supply to be 55mV when $W_p$=5.2μm and the shaded region shows the PMOS width that ensure functionality. The simulations in Figure 6.2 assume a typical transistor corner.

The effects of process variations provide uncertaintly about device leakage currents and drive currents. Therefore, process variations can greatly affect the minimum voltage supply operation, and the worst case process corners should be accounted for in minimum voltage analysis. For the inverter, the worst case corners for sizing are for the *fast NMOS/ slow PMOS* (FS) and the *slow NMOS/fast PMOS* (SF), as they affect the appropriate $I_{on}/I_{off}$ ratios. At the FS corner, the fast NMOS is more leaky than the slow PMOS. Therefore, as supply voltages drop, the minimum PMOS width size should be much greater than for the typical transistor case to balance the large NMOS leakage current and the reduced drive strength of the PMOS. The effect is a shift of $W_p$(min) towards higher supply voltages as seen in Figure 6.3. Similarly, the SF corner sets $W_p$(max), which is also shifted towards higher supply voltages. The intersection of the curves at these two process corners in Figure 6.3 show that the inverter is only guaranteed to operate down to 195 mV by sizing the PMOS to be 5.4μm.

The $I_{on}/I_{off}$ metric and the min-max sizing curves are tools that are used to characterize and analyze datapath logic circuits for ultra low voltage operation. In the subthreshold implementation, designs will target 100mV operation for a typical transistor and <250mV operation at worst case process corners. These new tools will be used to analyze the effect of parallel leakage, cell interfacing and stacked devices which do not typically create problems for traditional logic circuits until they are used in subthreshold operation.

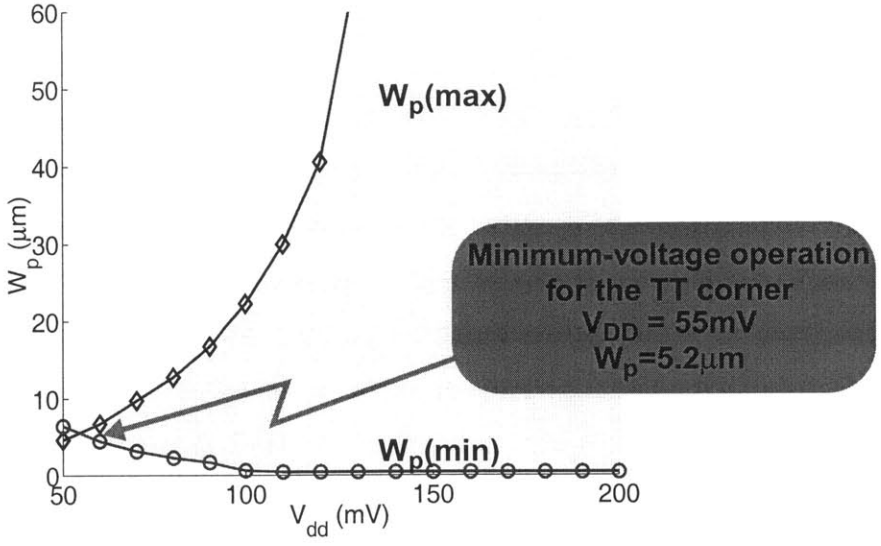


**Figure 6.2: The min-max sizing curves for an inverter at the TT corner. The intersection of $W_p$(max) and $W_p$(min) indicate that the lowest $V_{DD}$ of operation is 55mV. The shaded region gives $W_p$ that ensure functionality.**

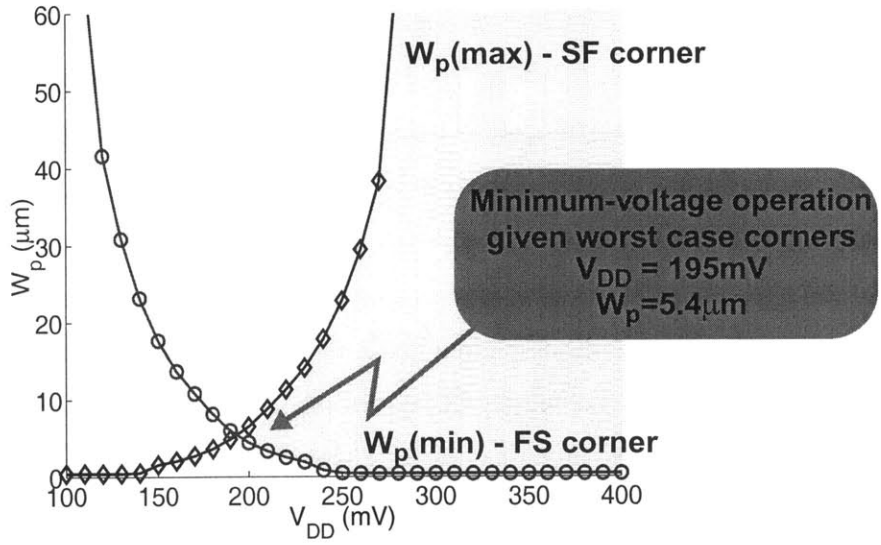


**Figure 6.3: The minimum-voltage operation of the inverter is affected by process variations. Given the worst case corners (FS and SF), we are only able to guarantee operation at $V_{DD}$=195mV.**

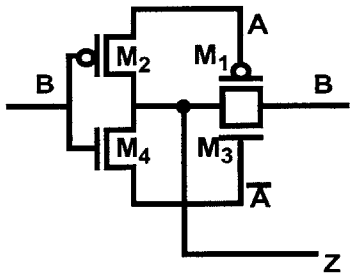


**Figure 6.4: The standard cell tiny XOR gate.**

## 6.3 Subthreshold logic

Logic circuits are known to scale down to the subthreshold region. The previous analysis shows the inverter can function down to 55mV for the typical transistor and down to 195mV for worst case process variations. This chapter includes analysis of other logic circuits commonly found in a standard cell library targetted for 1.8V operation and exposes effects of low-voltage operation. Parallel leakage, stacked transistors, and sneak leakage effects appear at low-voltage operation and require new subthreshold cells for minimum voltage operation.

Parallel leakage is the leakage in multiple devices that are placed in parallel. Too many devices in parallel increases the static leakage current and affects functionality. An example of parallel leakage is illustrated in the XOR gate. Figure 6.4 shows the schematic of the tiny XOR gate commonly used in traditional circuit design. In strong inversion, for the input vectors A=1 and B=0 the output node, the PMOS device $M_2$ pulls up the output node, Z, high. However, once the voltage is scaled down to 100 mV, the leakage of parallel devices $M_1$, $M_3$, and $M_4$ from Z to ground dominates the drive current of $M_2$. The drive current and leakage current flow for this input vector is shown in Figure 6.5. Because there are three "off" devices and on "on" devices, $I_{on}/I_{off}$ is degraded.

The tiny XOR is simulated for all vectors for a supply voltage of 100mV and at the typical transistor corner. All inputs to the XOR are buffered. The simulation shows that for A=1, B=0, the P output is driven to 55mV. This effect is further compounded if process variations are considered in the analysis.

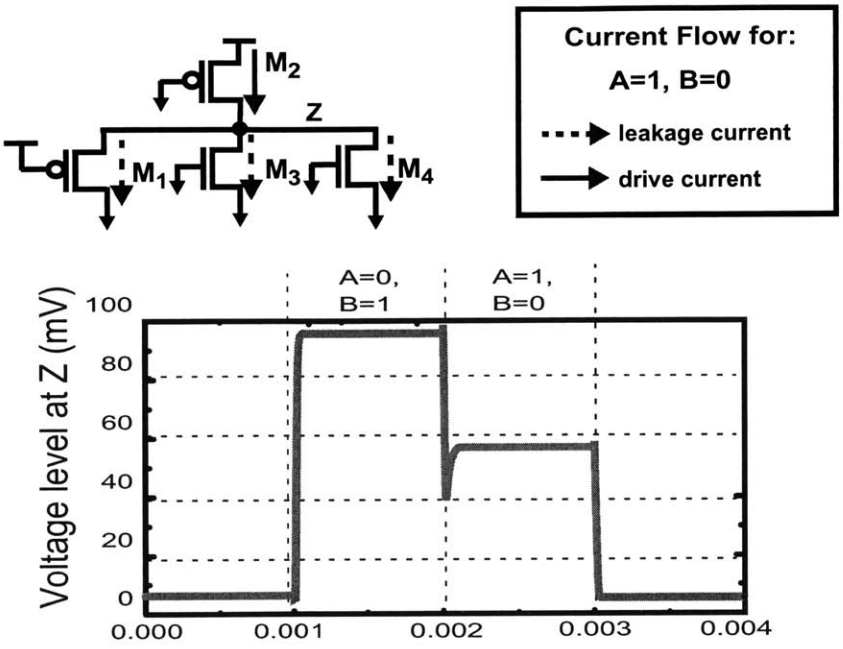


**Figure 6.5: The effects of parallel leakage is compounded at ultra-low voltages. The standard-cell XOR gate illustrates the effect of parallel leakage. For the input vectors A=1 and B=0 at $V_{DD}$=100mV, the output node (Z) is only driven to 55mV.**

The subthreshold XOR using transmission gates (Figure 6.6) is a circuit where parallel devices are minimized (or balanced) for minimum-voltage operation. The transmission gate XOR is simulated for all possible input vectors at 100mV supply voltage and for the typical transistor process corner. All inputs to the XOR are buffered, thereby causing realistic delays at the input transitions. The simulation in Figure 6.7 shows that the transmission gate XOR is able to function at the input vectors (A=1, B=0) that failed at 100mV for the tiny XOR.

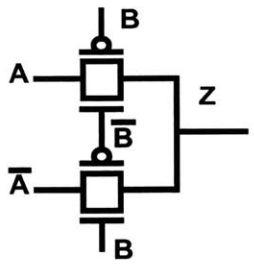


**Figure 6.6: Transmission gate XOR for subthreshold operation.**

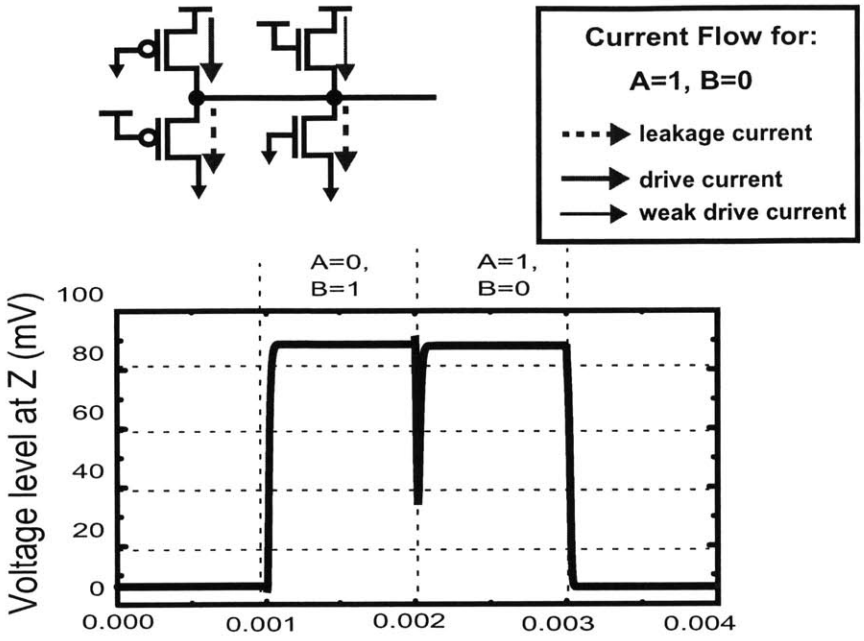


**Figure 6.7: A subthreshold XOR gate has balanced leakage for the same input vectors as shown in the simulated waveform at 100mV.**

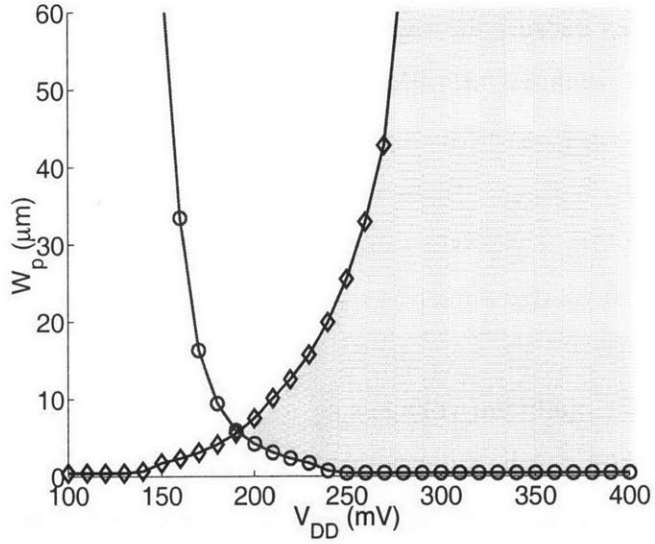


**Figure 6.8: The min-max curves for the subthreshold XOR cell at worst-case corners.**

The drive and leakage current flow schematic shows that there are two devices pulling the output node to high and two devices pulling down. Therefore $I_{on}/I_{off}$ is not degraded, and the XOR is fully functional. Also, because the transmission gate XOR contains both NMOS and PMOS in both pullup and pulldown configuration, it is more immune to process variations. Figure 6.8 shows the min-max curves for the transmission gate XOR at the worst case process corners. The minimum operating voltage is at 190 mV, which is a lower minimum operating voltage than the inverter because the leakage is balanced.

Stacked devices have two effects on the functionality circuits in subthreshold operation. First, when stacked devices are conducting, the effective drive of the two devices is diminished, (e.g. the drive current of two stacked devices is approximately halved). Second, the threshold voltage in a stacked device increases when the source-to-body voltage increases. This leads to a decrease in the leakage current through the stacked device [55].

The effect of both parallel leakage and the stack effect are seen in a 3-input NAND. Figure 6.9a is the min-max sizing curve for a 2-input NAND and Figure 6.9b is the min-max sizing curve for a 3-input NAND. The minimum-voltage operation of the 3-input NAND is 15mV higher than that of the 2-input NAND. For the output-low case all NMOS devices are "on", but because the are stacked NMOS devices, the drive current is reduced. Additionally there are three "off" PMOS devices, there is more leakage from $V_{DD}$ to the output. Therefore, $I_{on}/I_{off}$ decreases at low supply voltages. The worst case output-high that gives the lowest $I_{on}/I_{off}$ is when one PMOS is "on" and one NMOS is "off". Because there are two other "off" PMOS devices, the $I_{on}/I_{off}$ for the output-high is better. The result is that the minimum voltage operation occurs at higher supply voltages, but the $W_p$ sizes decrease.

Table 6.1 has the minimum-voltage supply and the optimal $W_p/W_n$ sizing for the multiple input cells. These results indicate that three and four input NANDs and NORs should be avoided, and for minimum voltage design, stacks should be limited to 2 in minimum-voltage design.

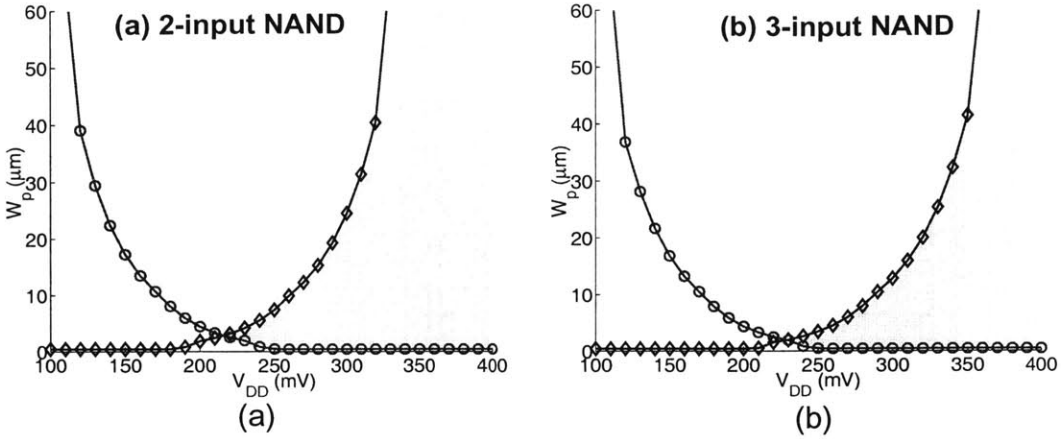


**Figure 6.9: The min-max sizing curves for (a) the 2-input NAND and (b) three-input NAND. The minimum-voltage operation of the 3-input nand is higher than the 2-input NAND due to parallel leakage.**

**Table 6.1: Simulation results from multiple input NANDs and NORs.**

| Standard-Cell name | $W_p/W_n$ | minimum $V_{DD}$ |
|---|---|---|
| 2-input NAND | 6.3 | 215 mV |
| 3-input NAND | 4.5 | 230 mV |
| 4-input NAND | 2.8 | 235 mV |
| 2-input NOR | 23.5 | 215mV |
| 3-input NOR | 32.0 | 230mV |
| 4-input NOR | 40.0 | 240mV |

Since the implementation of the logic blocks of the subthreshold chip is standard-cell based, we need to be careful about the interfaces between different cells. If all of the standard cells are designed in static CMOS, there will not be any sneak leakage paths because the inputs to static CMOS gates are the NMOS or PMOS gates. However, when transmission or pass gates are introduced as in the tiny XOR gate, sneak leakage paths occur and require new buffering considerations.

The sneak leakage effect is seen in the bit adder cell. The tiny XOR cell described previously and its complement (tiny XNOR) are connected using a transmission gate to form an adder cell as seen in Figure 6.10. When the input vectors is $(A,B,C_{in}) = (0,0,0)$ there is a sneak leakage path that affects the Z, $\overline{Z}$, and $\overline{Sum}$ nodes. The cascaded devices between the inputs and the $\overline{Sum}$ signal further degrades the output signal. This indicates that when

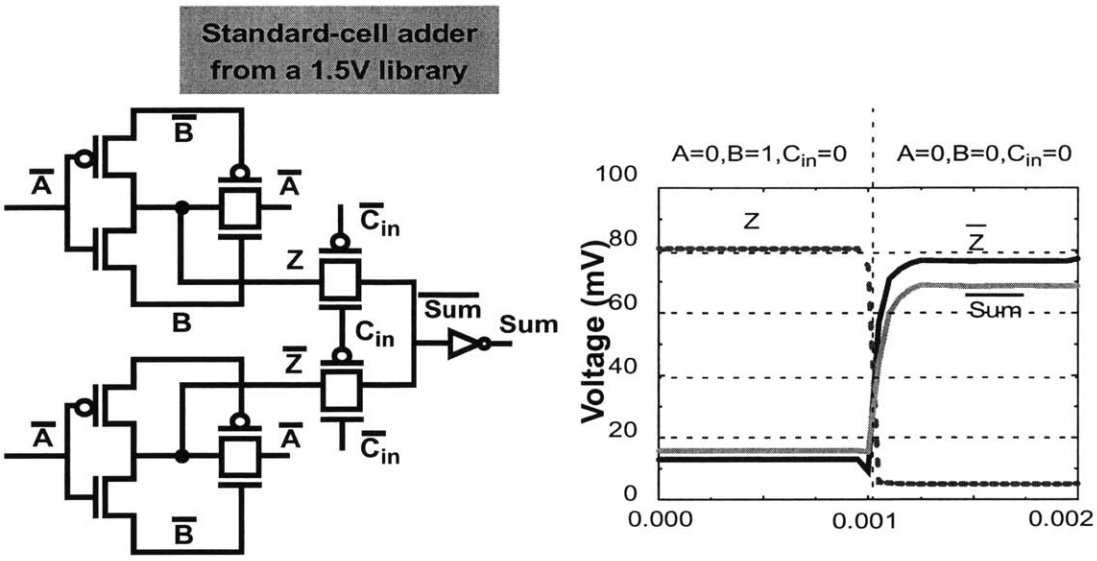


**Figure 6.10: An adder cell from the 1.5V standard-cell library is created by connecting the XOR and XNOR to the transmission gate causes sneak leakage paths to degrade the Z, $\overline{Z}$ and $\overline{Sum}$ nodes.**

designing for subthreshold operation, it is important that the cell interfaces be studied carefully.

Introducing inverters and buffers between cascaded devices is one way to eliminate the sneak leakage path. Inverters at Z and $\overline{Z}$ can ensure clean inputs to the transmission gate. Another way is to reconfigure the adder so that the outputs to the pass gate is the input to the device gates to the transmission gate. This is demonstrated in an adder cell designed for subthreshold. First, the subthreshold adder uses the subthreshold XOR described previously (Figure 6.11). Second, the XOR output, P, drives the selectors of the transmission gates which eliminates cascading devices and the sneak leakage path. The simulated waveform shows the improved $\overline{Sum}$ signal (Figure 6.11). In the subthreshold standard cell library, both the full-adder and half-adder block are designed, and all inputs and outputs are fully buffered. This is to prevent further sneak leakage paths between standard cells.

The full-adder block shown in Figure 6.11 and a similar half-adder block are used to create a custom scalable bit-precision Baugh-Wooley (BW) multiplier. In a full-custom multiplier, the redundant inverters that were included in the standard cell adder are removed. If the outputs of the bit adders are buffered, then the inputs need not be buffered.

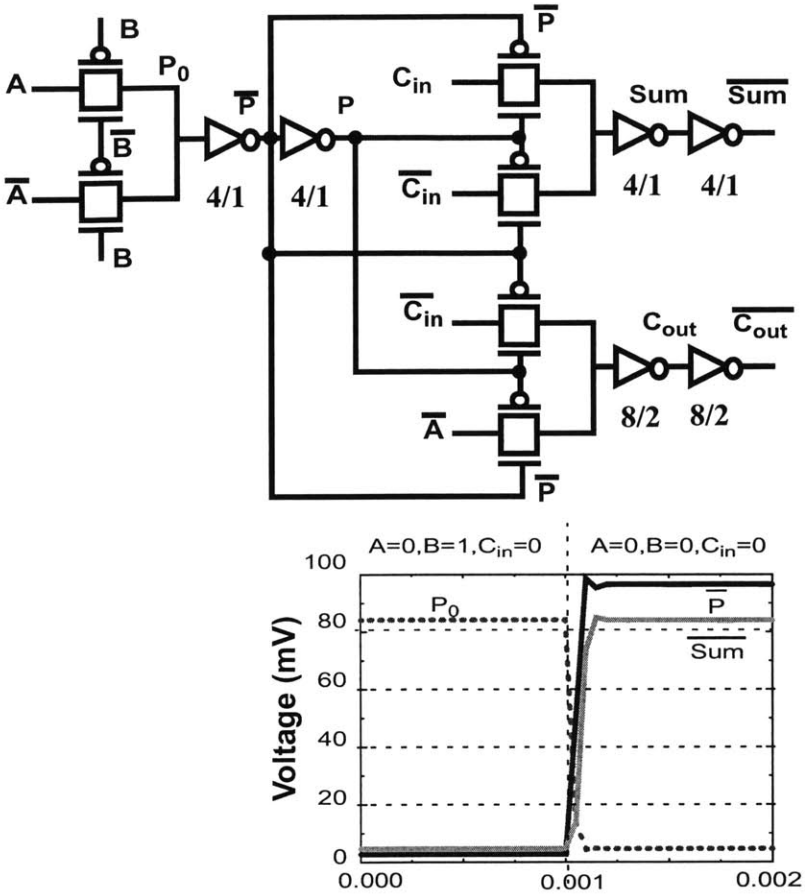


**Figure 6.11: Subthreshold Full-adder Cell design. This cell is used for in the Baugh-Wooley Multiplier.**

The output buffers for Sum and $C_{out}$ and their complements depend on their configuration within the Baugh-Wooley multiplier as shown in a 4-input BW multiplier example (Figure 6.12). The Sum and $\overline{Sum}$ are minimum sized ($W_p/W_n$=4/1) while the $C_{out}$ and $\overline{C}_{out}$ are larger ($W_p/W_n$=8/2). Sum and $\overline{Sum}$ are minimum sized because they drive the B and $\overline{B}$ inputs which are the device gates. However, the $C_{out}$ signal is always connected to the $C_{in}$ inputs to transmission gates. The $I_{on}/I_{off}$ of $C_{in}$ decreases due to leakage through the parallel devices, and larger buffers are needed.

The full-custom BW multiplier is assembled using a Skill script to place and route the adders for a compact layout. The BW multiplier is also designed to operate at 8-bit and 16-bit precision. The BW multiplier layout is shown in Figure 6.13.

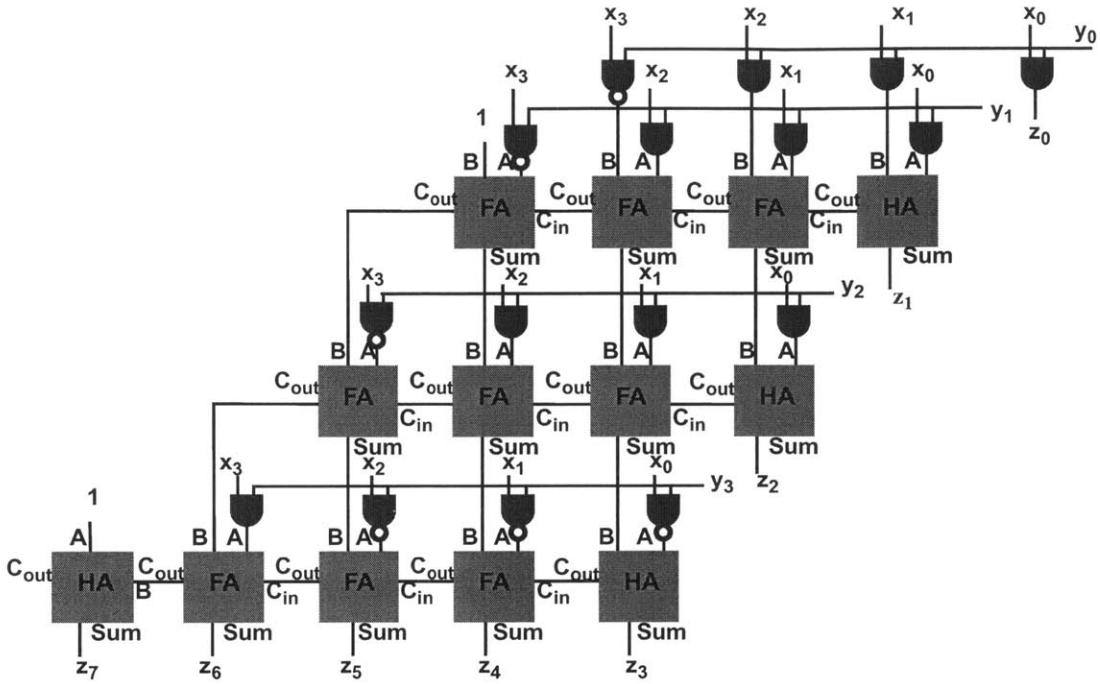


**Figure 6.12: A 4-input Baugh Wooley multiplier schematic used to illustrate input and output connections and buffering.**

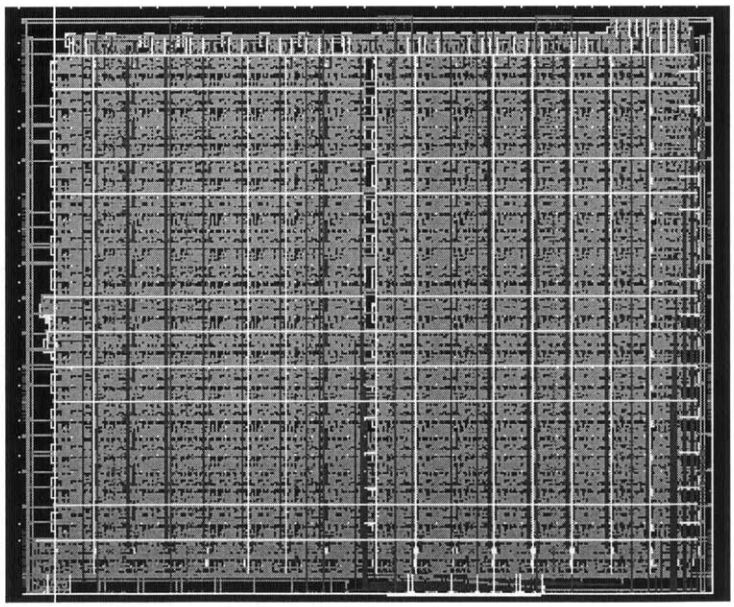

**Figure 6.13: Layout of the Baugh-Wooley multiplier. The multiplier consists of custom subthreshold full adder and half adder cells and are place-and-routed using custom Skill scripts.**

The other datapath logic is designed using the subthreshold standard cell library and a modified synthesis flow. The subthreshold standard cell library is further described in Section 6.6 and the design flow is described in [46][75].

## 6.4 Subthreshold memory

In this section, I discuss subthreshold memory design considerations. I analyze functionality of typical memory designs used in conventional memories such as the 6T SRAM, precharge and psuedo-NMOS bitlines. Next I propose new memory write and read schemes that improve the $I_{on}/I_{off}$ of the memory bitlines, such as negative gate-to-source read devices, tristate-based read and write circuits and a hierarchical read-bitline. The hierarchical read-bitline is used in the design of data memory and ROM's and achieves acceptable $I_{on}/I_{off}$.

In standard memory generators, a 6T SRAM is used (Figure 6.14). The 6T SRAM bit cell achieves small area, and the feedback through the cross-coupled inverters holds the state. NMOS pass transistors are used to write and read from the memory, and sense amplifiers on the bitline are used for fast read accesses. Current memory design chal-

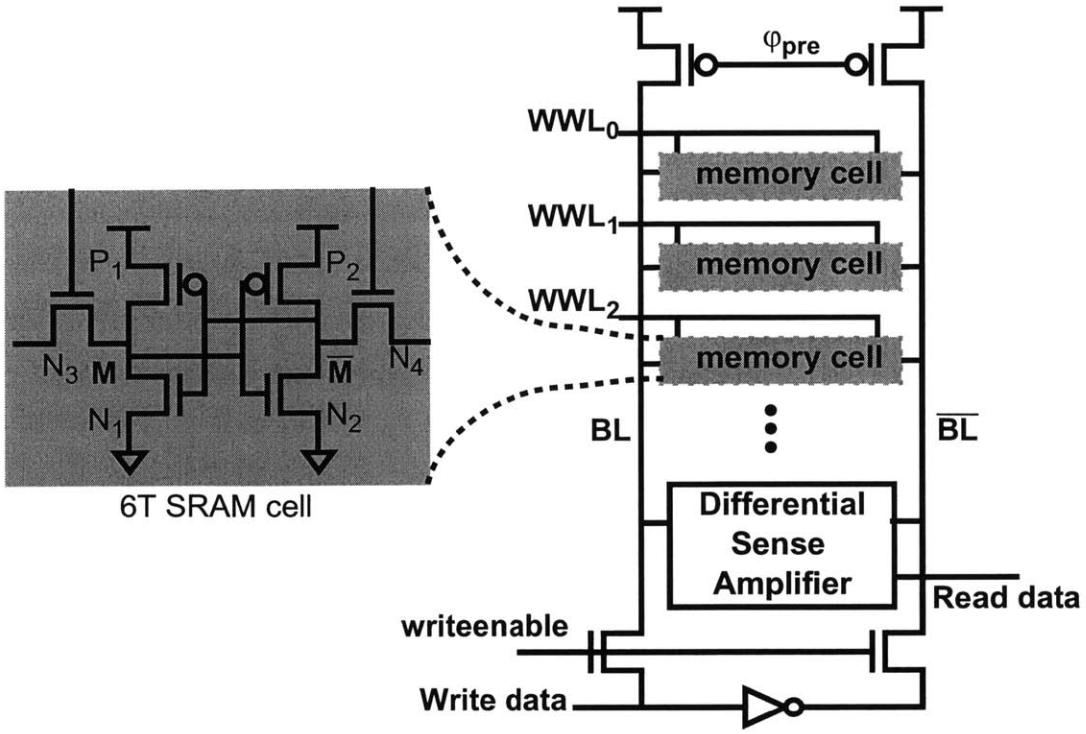


**Figure 6.14: 6T SRAM write and read.**

lenges, such as bitline capacitance, bitline leakage, speed, and sense-amplifiers design, worsen as voltage supplies scale down and when accounting for process variations.

### 6.4.1 Subthreshold write access

For the 6T SRAM bit cell, the NMOS pass transistors are used to transfer the data from the BL and $\overline{BL}$ to the memory cell. As the supply voltage scales down into sub-threshold, it is possible for the 6T SRAM to operate for the typical transistor corner.

At high voltages, the bit cell has minimum size cells to ensure minimum sized caches and memories. Also, the sizing of the devices in the bit cell must be analyzed to ensure that during a read access, the voltage on BL and $\overline{BL}$ does not flip the value stored in M and $\overline{M}$. During a write access, the pass transistors should be large enough to write into the cell. This type of analysis is usually done at high voltages, but is necessary in subthreshold for memories that operate with extremely low supply voltages [53].

For example, in Figure 6.15, initially $V_{high}=V_{DD}$ and $V_{low}=0$. When $V_{BL}= V_{\overline{BL}}=V_{DD}$, it is necessary to size the transistors such that $V_{low}$ is not pulled up through the voltage divider created by $N_2$ and $N_4$. A large $V_{low}$ can cause $N_1$ to conduct and thus changing the state of the memory bit. Figure 6.15 shows the voltage $V_{low}$ as a function of the cell ratio

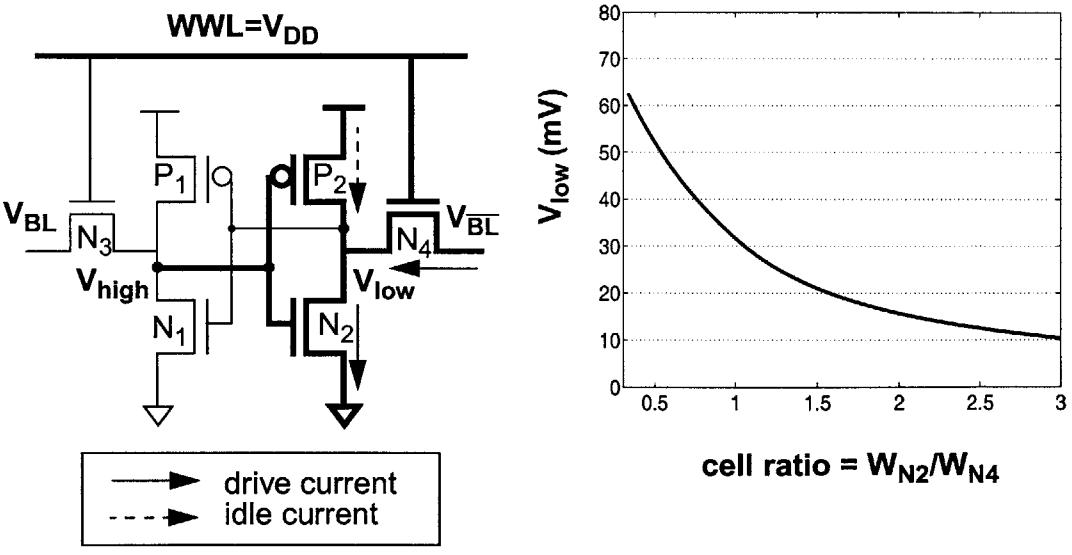


**Figure 6.15: 6T SRAM read access condition.**

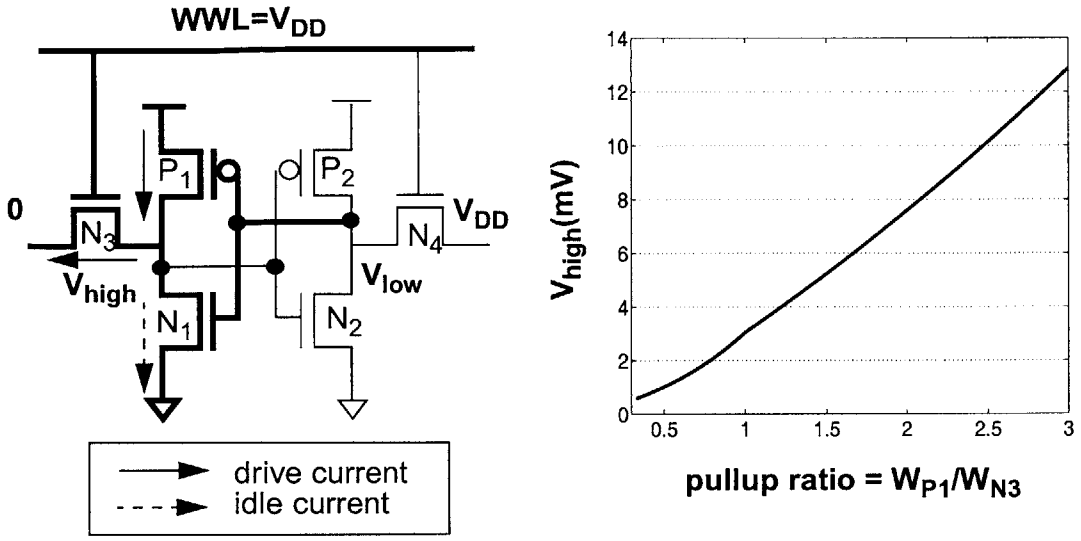


**Figure 6.16: 6T SRAM write access condition.**

($W_{N2}/W_{N4}$). This plot shows that for $V_{DD}$ of 100mV, a cell ratio between 1.5 and 3 is adequate to prevent read hazards. Because for this process, the NMOS is much leakier than the PMOS, the leakage through a minimum sized $P_2$ does not affect the cell ratio. However, if $P_2$ does get appreciably large, then the leakage through $P_2$ will shift the curve towards larger $N_2$.

For the write access as shown in Figure 6.16, initially $V_{high}=V_{DD}$ and $V_{low}=0$. Then when $V_{BL}=V_{DD}$, $V_{BL}=0$, and $V_{row}=V_{DD}$ the pass transistor, N3, is sized to pass the values from BL into the memory. However, because initially $P_1$ is conducting, there is a voltage divider for $V_{high}$. Figure 6.15 shows the voltage $V_{high}$ as a function of the pullup ratio ($W_{P1}/W_{N3}$) for $V_{DD}$=100mV. $V_{high}$ must be a value low enough to switch the inverter and thus write to memory. The figure shows that even for $W_{P1}=3W_{N3}$ there is no problem for the pulldown NMOS to switch the cross coupled inverter. At 100mV, for this technology process, the PMOS has lower leakage than the NMOS at 100mV supply voltage.

By sizing $W_{P1}=W_{N3}$ and $W_{N1}=1.5*W_{N3}$ and all other transistors in the 6T SRAM respectively, the 6T SRAM is able to function at 100mV. This is shown in Figure 6.17. Between 0-2msec, BL is high, indicating that when WWL goes high, a "1" should be written into M. Between 2-4msec, BL goes low and a "0" is written into memory when WWL goes high again.

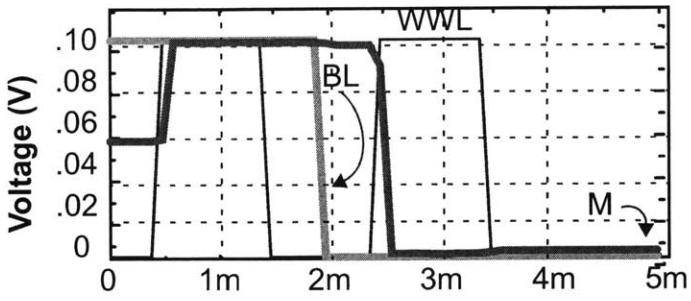


**Figure 6.17: 6T SRAM memory write simulation for $V_{DD}$=100mV for a typical transistor process corner.**

The effect of process variations must be included in the SRAM analysis. At the SF process corner, the PMOS leakage current is larger than for the nominal TT case, and therefore must be considered in the read access analysis. The cell ratio for the SF case at $V_{DD}$=250mV is shown in Figure 6.18(a). The cell ratio for the SF case shows that increasing $W_{N2}/W_{N4}$ to 4 is enough to ensure that $V_{low}$ be 20% of $V_{DD}$.

To satisfy the write access requirement at the SF corner $W_{P1}/W_{N3}$ should be decreased so that N3 strength is enough to pull $V_{high}$ down. Figure 6.18(b) shows the pullup ratio for the SF corner. Assuming that we want to maintain $V_{high}$ at 20% of $V_{DD}$, the pullup ratio is plotted as $V_{DD}$ is increased from 0 to 1V. At low $V_{DD}$, the pullup ratio is extremely small, because P1 is much stronger than N3, and $W_{N3}$ is much greater than $W_{P1}$. Also, the body effect on N3 further reduces the drive strength of the pass transistor. At large $V_{DD}$, when N3 is in strong inversion, its drive current is larger and therefore does not require large widths. P1 can be larger than N3 for $V_{DD}$>700mV. Additionally, the body effect is not as pronounced in strong inversion as in subthreshold. For the SF corner at 250mV, N3 has to be more than 10 times larger than P1.

However, the FS corner must also be considered. At the FS corner, the drive current through P1 must be large enough to switch the bitcell. If $W_{N3}$=10*$W_{P1}$, and from the read access condition, $W_{N1}$=4*$W_{N3}$, then $W_{P1}/W_{N1}$ is set to be 0.025. This condition may be adequate to write for the SF condition, but at the FS corner, the PMOS is much too weak and the inverters do not pass input of zero. Additionally, at the FS corner, for large N3

with respect to N1 and P1, large leakage currents through N3 can also affect the functionality of this circuit.

The switching ratio ($W_{P1}/W_{N1}$) for the FS corner is shown in Figure 6.18(c). This is a minimum bound on $W_{P1}$ needed for $V_{high}$ to be 80% of $V_{DD}$ given that $V_{low}$ is 0. This is similar to the min-max curves of Figure 6.3, except now allowing the PMOS width to be less than the NMOS width. Therefore, there is a trade-off when sizing the NMOS devices. The conditions for read and write access at low voltage and for the SF process corner requires large pass transistors (N3) and pull down transistors (N1) and requires small pullup devices (P1). At the FS process corner, a large pullup device (P1) is needed for the cross inverters to change state. If we fix N1 to be 4 times N3, then the maximum bound

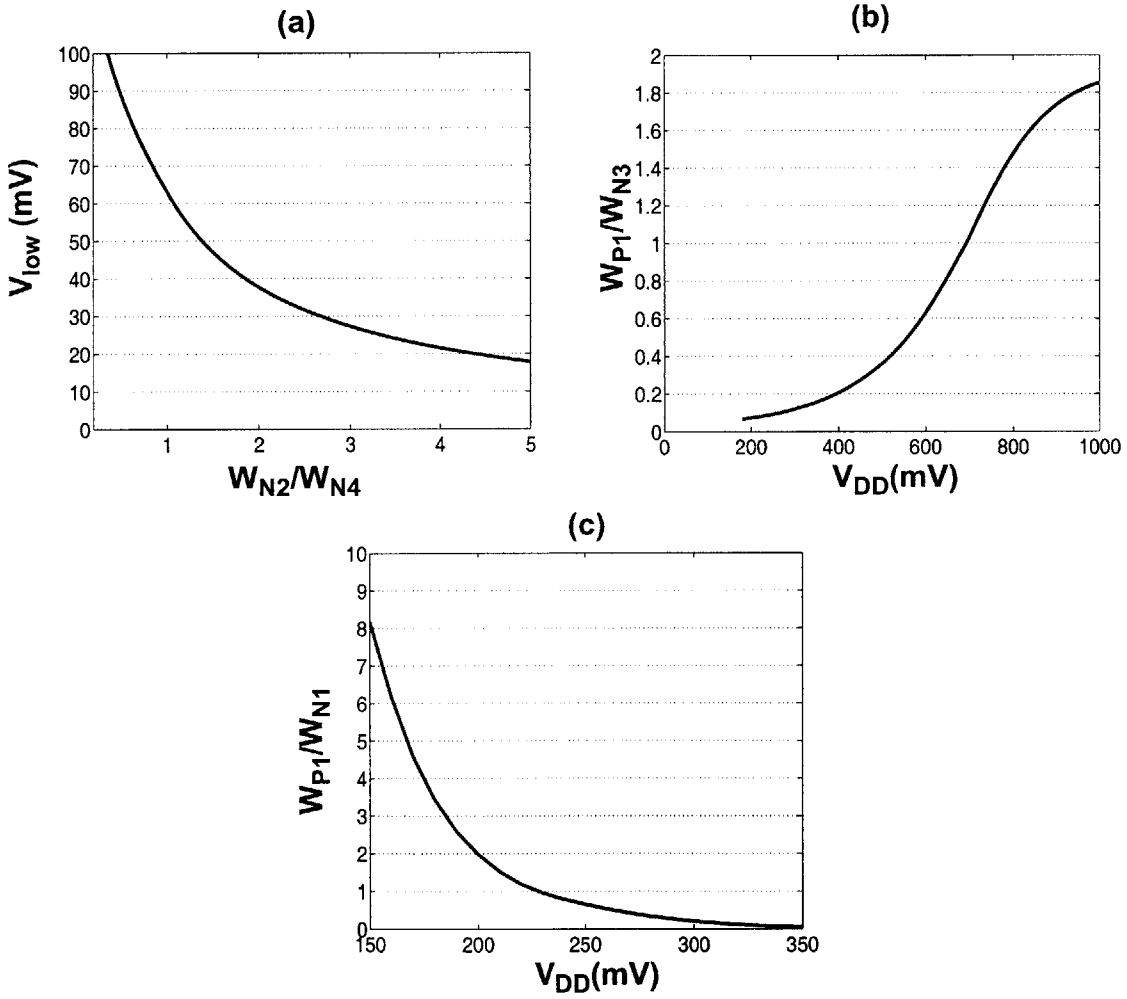


**Figure 6.18: (a) 6T SRAM cell ratio condition for the SF corner. (b) SRAM pullup ratio as a function of $V_{DD}$ in order for $V_{high}$=0.2*$V_{DD}$.(c) Switching ratio for the worst case process corner (FS).**

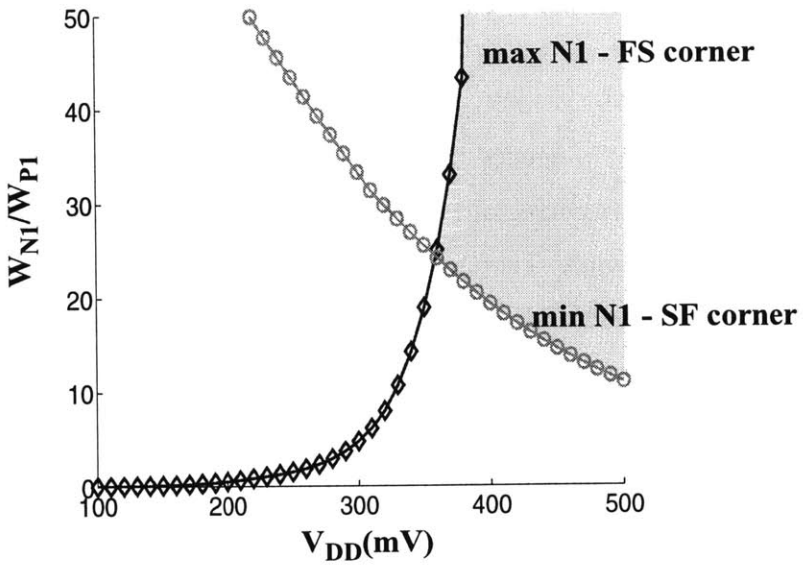


**Figure 6.19: The min-max curves for sizing N1 for the worst case process corners. The minimum supply voltage operation happens at 360mV.**

can be found from the write access condition as seen in Figure 6.19. The minimum supply voltage given worst case process variations is at 360mV when $W_{N1}/W_{P1}$=25 which means that N1 is 25 times the size of P1. Even at 500mV, N1 is 10 times P1 in order to be functional.

A latch based write scheme with $C^2$MOS tristate inverters (Figure 6.20) is more suitable for subthreshold operation. The disadvantage of this approach is more transistors and for the memory bit cell. The main advantage is that static logic is a more reliable solution for the write access at the worst case process corners. Figure 6.20(b) shows the min-max curves for the memory write access and memory cells showing functionality at process corners as low as 215mV.

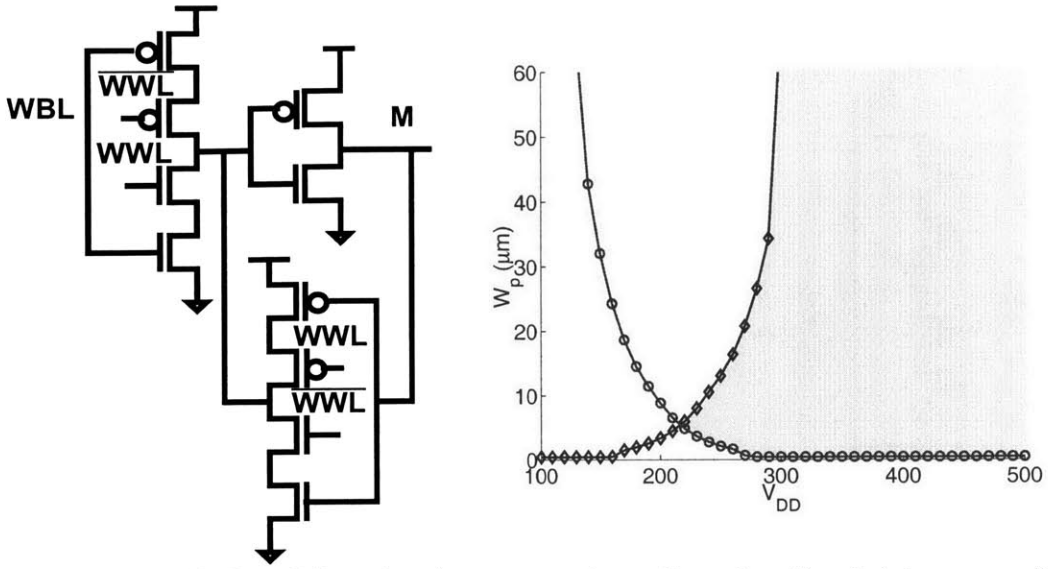


**Figure 6.20: Latch based write access schematic and optimal sizing assuming a minimum $W_n$.**

### 6.4.2 Subthreshold read access

The read operation of the memory in subthreshold is a very challenging design problem. In the 6T SRAM shown in Figure 6.14, after the PMOS precharge phase, the memory value is read out through the pass transistors onto the bitline. A small differential is detected between BL and $\overline{BL}$ by a sense amplifier which restores the output to rail-to-rail. There are many issues that the SRAM designer has to face as voltage supplies scale. One challenge is the sense amplifier design. As the voltage supply drops to 100mV, the voltage differentials are 10's of millivolts. The tiny differential and the effect of noise on dynamic nodes makes the 6T SRAM read access highly unreliable.

Another challenge is bitline leakage, where the leakage through the pass transistors causes the dynamic BL to drop. This effect is especially felt as the clock frequencies are reduced. Bitline leakage also causes a differential between BL and $\overline{BL}$ simply due to process variations. If the differential due to unmatched devices is large enough, it is amplified by the sense amplifiers. Therefore, it is imperative that transistors in memory be well matched when using a differential sense amplifier read bitline scheme.

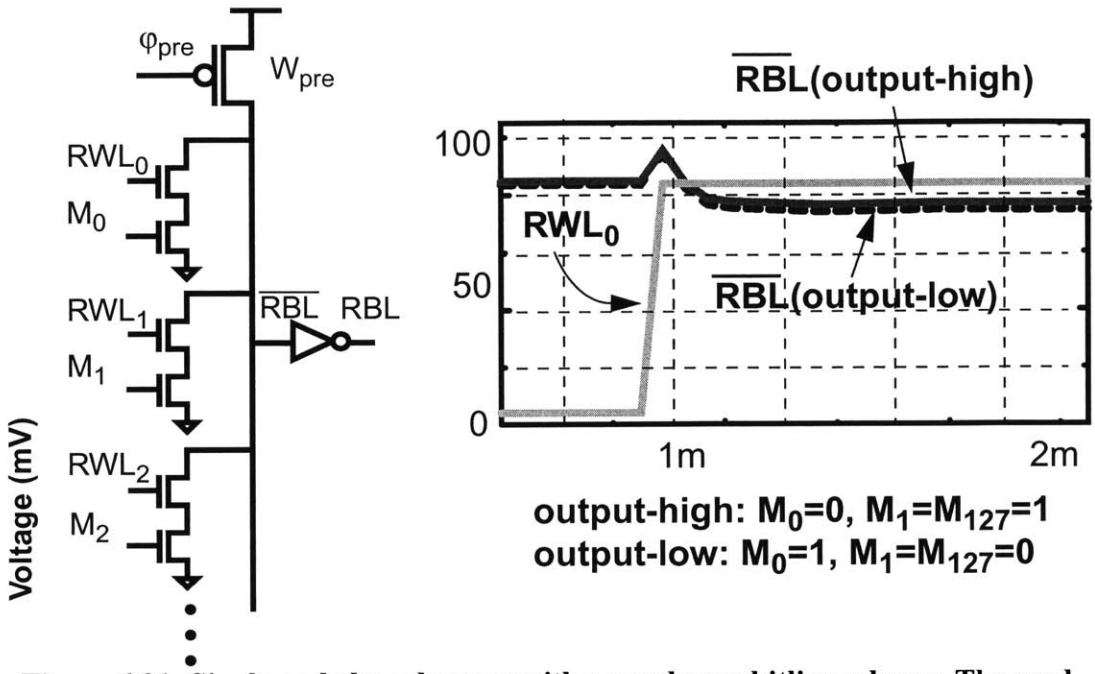


**Figure 6.21: Single-ended read access with a precharge bitline scheme. The read bitline is susceptible to leakage through the idle devices. The simulation shows $\overline{RBL}$ when $M_{1\text{-}127}$=1 and $M_0$=1 and $M_0$=0. Because the drive current is very low compared to the parallel leakage current, the leakage dominates the functionality of the bitline, and $\Delta\overline{RBL}$ is only 2 mV.**

Clock speed is not the key metric in this application, but functionality at the minimum voltage supply. Therefore, the FFT memory does not require sense-amplifier based read-bitlines for fast read accesses. Also, because a latch based scheme is used for the write access, only single-ended read access schemes are considered. Figure 6.21 shows a common single-ended scheme using dynamic logic. The dynamic bitline scheme has minimum width NMOS devices and one PMOS precharge transistor.

In order to size the precharge transistor, the output-high case is analyzed at 100mV voltage supply during the evaluation stage ($\varphi_{pre}$=1). The precharge transistor is sized ($W_{pre}$) for the input vector $M_0$=0, $M_1$-$M_{127}$=1, and $RWL_0$=1. For this input vector, the precharge device's leakage is pulling-up $\overline{RBL}$, but the leakage through the 127 NMOS devices are pulling the output node down. Simulations show that for the output to be held at 90% of $V_{DD}$, $W_{pre}$=1100μm is needed. The simulation of the bitline at 100mV shows that $\overline{RBL}$ is at 90% of $V_{DD}$ (Figure 6.21).

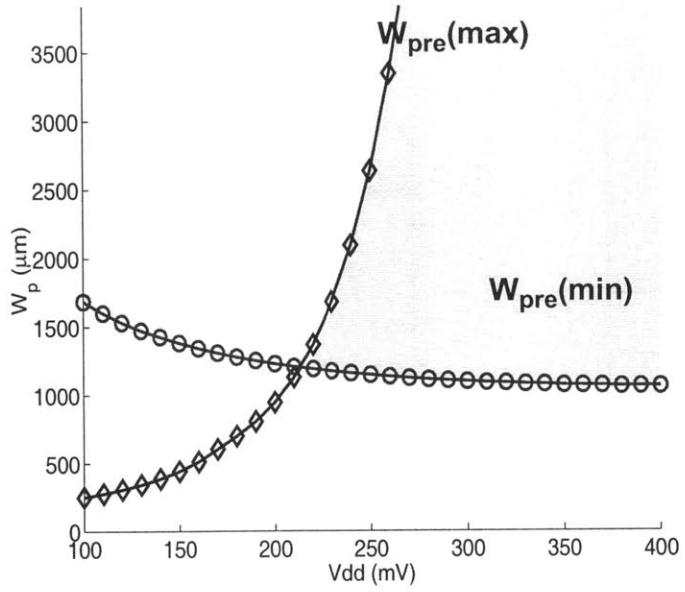


**Figure 6.22: The figure shows the min-max curves for the precharge read bitline for a typical transistor process corner.**

The worst case for output-low is for the input vectors: $M_0$=1, $M_1$=$M_{127}$=0, and $RWL_0$=1. For this input vector, $\overline{RBL}$ is only at 88mV and the bitline is not driven down to 10% of $V_{DD}$. For the output-low case, the $I_{on}/I_{off}$ is very small. The $I_{on}$ of the single "on" NMOS device is much smaller than the high leakage through the precharge transistor. Also the leakage current of the 127 "off" devices is lower since the devices are stacked.

Figure 6.22 shows the min-max curves for the precharge scheme for the typical transistor process corner. $W_{pre}$(min) is the minimum allowable PMOS that is able to pull-up the output to 90% of $V_{DD}$. The PMOS size goes up with decreasing $V_{DD}$, as the 128 leaking NMOS devices have larger subthreshold slope than the PMOS. Therefore $I_{leak}$(NMOS)/$I_{leak}$(PMOS) will increase as the supply voltage decreases, but stays fairly constant for larger $V_{DD}$. However, the figure shows a huge range of $W_{pre}$(max). $W_{pre}$(max) is the maximum allowable PMOS width for the worst-case input vector so that the output is 10% of $V_{DD}$. The reason is that at low supply voltages, the width of the precharge should be small so that the leakage through the PMOS does not dominate the drive current of the NMOS pull-down. However, as the supply voltage increases, the drive current of the NMOS increases exponentially, and quickly dominates over the precharge leakage current. The minimum voltage supply for the precharge scheme is 215mV. At the

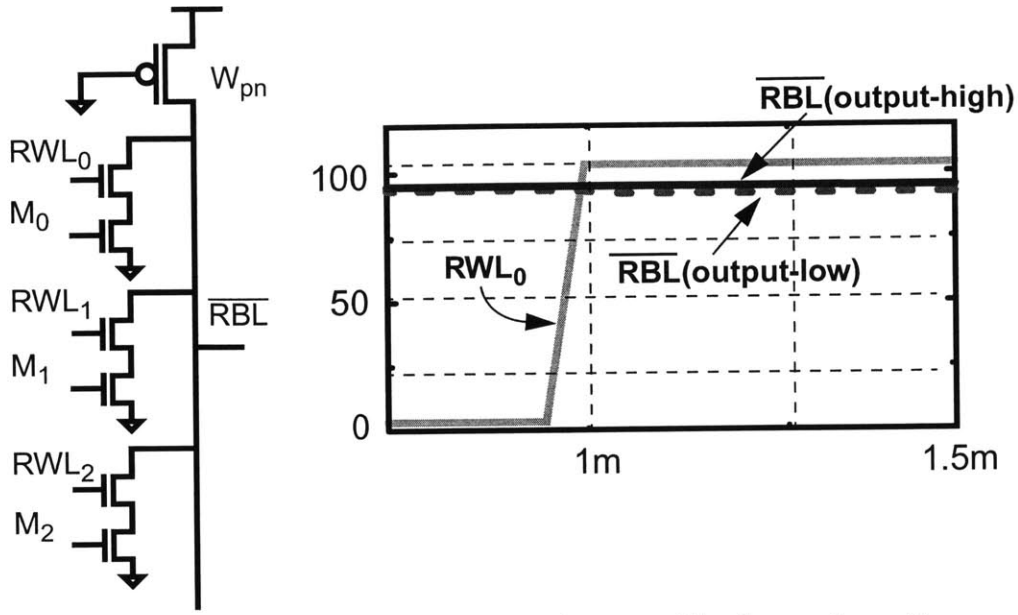


**Figure 6.23: P-NMOS pull-up transistor read-access. The figure shows the min-max curves for a typical transistor.**

worst case corners, the minimum voltage supply is 380mV for a typical transistor processor corner. However, note that the precharge scheme requires extremely large PMOS devices.

A similar design commonly used involves using a psuedo-NMOS (PNMOS) pull-up device instead of the precharge transistor as shown in Figure 6.23. The gate to the PMOS is tied to ground and is "on". The PNMOS scheme is almost identical to the precharge scheme during the evaluate phase. The only difference is substituting a smaller PMOS device that is "on" versus a larger PMOS device that is leaking. The two scheme have the same $I_{on}/I_{off}$. The pull-up current through the PMOS devices is the same if we size for the worst-case pull-up input vector to achieve 90% of $V_{DD}$. The main advantage is that the PNMOS device is much smaller, thus saving silicon area. The worst-case input vector gives $W_{pn}$ of 55μm which is 20 times smaller than for a precharge scheme. However, again for the output-low transition, the bitline is not driven down to zero for the worst-case input vector. The simulation in Figure 6.23, shows that the $\Delta\overline{RBL}$ is only 2mV, the same as the precharge scheme.

Figure 6.24 shows the min-max curve for the PNMOS transistor for a typical transistor process corner. There is a very narrow region of PMOS widths that achieve functionality

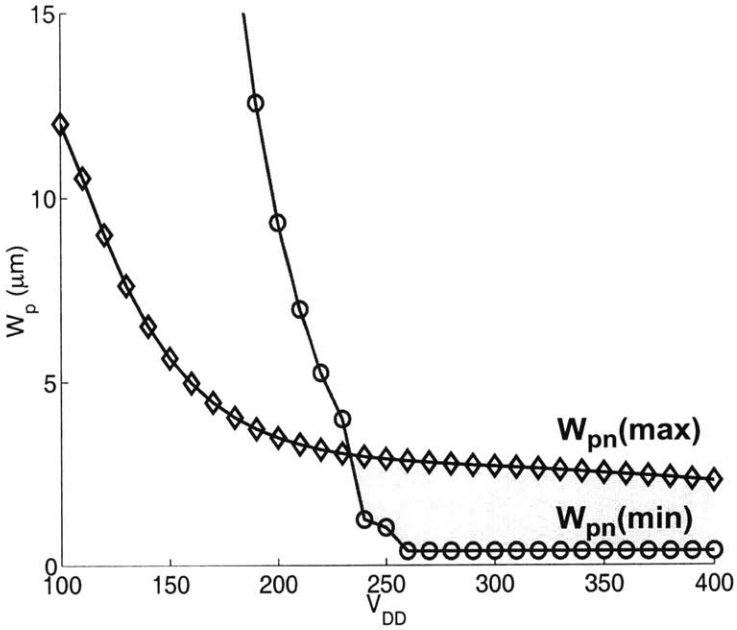


**Figure 6.24: The figure shows the min-max curves for the PNMOS read bit-line for a typical transistor process corner.**

for low-voltages. This is in contrast to the precharge scheme which is shown in Figure 6.22. $W_{pn}(min)$ is the minimum allowable PNMOS width where the output is 90% of $V_{DD}$ for the worst case input vector. For $V_{DD}$>260mV, a minimum PNMOS width provides enough drive current to pull the output high. However, as $V_{DD}$ decreases, the PMOS drive current also decreases exponentially and a larger PNMOS devices is needed. $W_{pn}(max)$ is the maximum allowable PNMOS device where the output is driven to 10% of $V_{DD}$. As the voltage supply increases, the gate-to-source voltage of the PNMOS device also increases, therefore reducing $W_{pn}(max)$. The lowest operating voltage is $V_{DD}$=230mV. However, at the worst case process corners, there are no widths for $W_{pn}$ that satisfy the 10-90% output voltage level constraint. At the SF corner, the minimum width PMOS is too strong for the minimum width NMOS to pull down. At the FS corner, all of the leaking NMOS devices require a larger precharge device.

One way to improve the $I_{on}/I_{off}$ at lower voltage is to use dynamic boosting of the wordlines [29]. Increasing the gate voltage of the wordlines, leads to higher drive currents in the pull-down path. In SRAMs, one way to improve read access speed is to use a charge pump to increase the voltage applied to the wordlines. Figure 6.25 shows how a charge pump can be used to increase the voltage applied to the readwordline from $V_{DD1}$ to $V_{DD2}$.

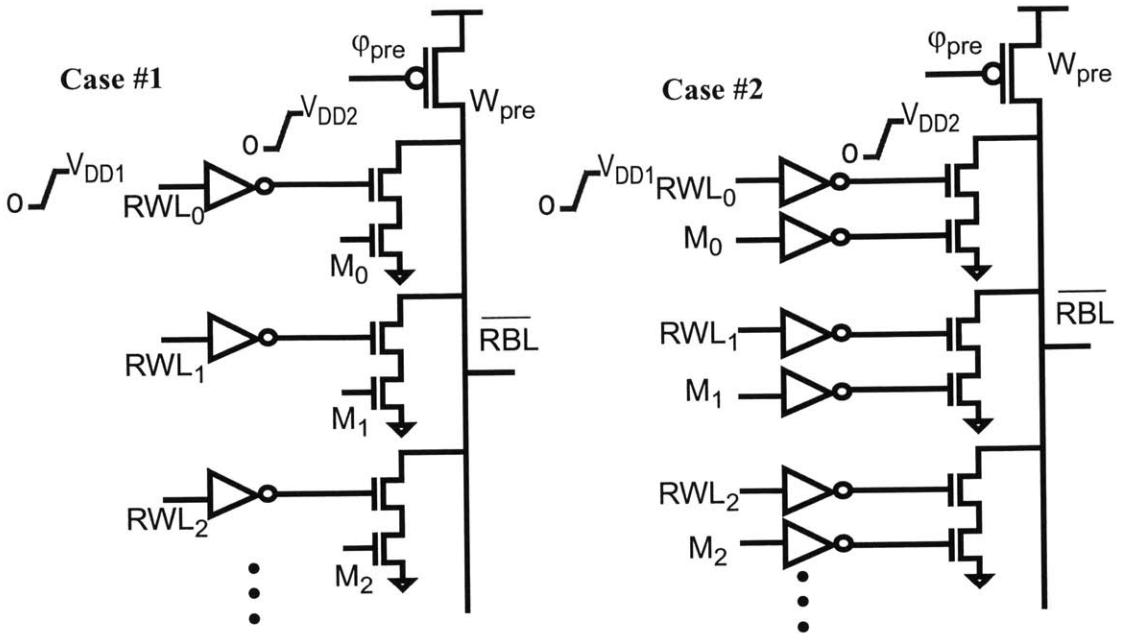


**Figure 6.25: Increasing the voltage on the gate from $V_{DD1}$ to $V_{DD2}$ increases drive currents of the pull-down devices. Case #1 boosts the readwordline, and Case #2 boosts both wordlines and memory inputs.**

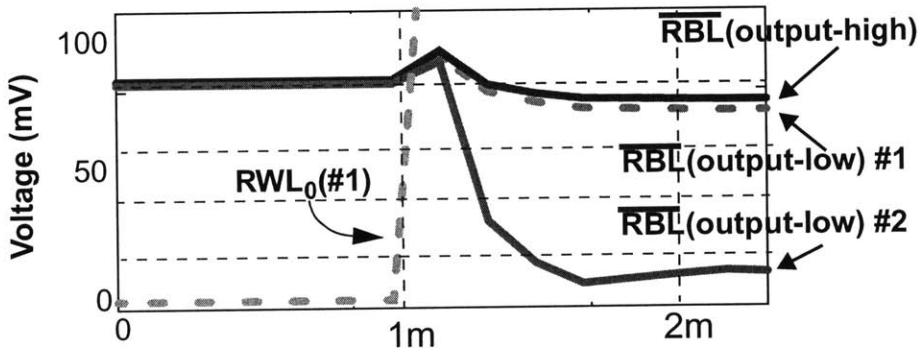


**Figure 6.26: Simulation of the readbitline with higher voltages are applied to the readbitline. Case #1 is when only $RWL_0$ is boosted to 250mV. Case #2 is when both $MB_0$ and $RWL_0$ are boosed to 250mV.**

In case #1, only the read wordlines are boosted and in case #2 both read wordlines and memory bit inputs are boosted.

Figure 6.26 shows a simulation for the two cases when boosting from 100mV to 250mV. As in the nominal precharge design, $W_{pre}$ of 1100μm maintains an output-high of 90mV for the worst case input vector. In case #1, when only the read wordline ($RWL_0$) is boosed to 250mV, then the wordline is pulls down further than in the nominal precharge case and $\Delta\overline{RBL}$=4mV. Because the devices are stacked, the effective pull-down resistance

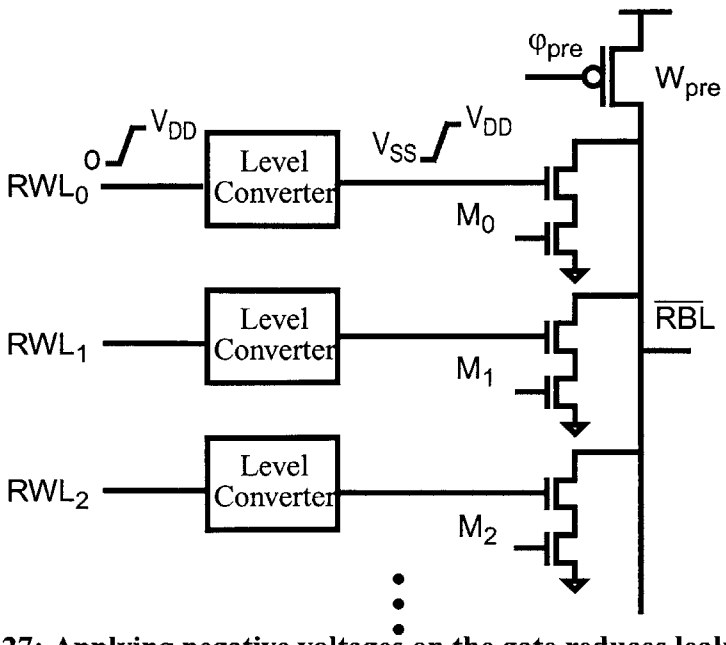


**Figure 6.27: Applying negative voltages on the gate reduces leakage currents through the NMOS pull-down devices.**

is not decreased significantly unless the input voltages to both devices are boosted as in case #2. When applying both wordline and memory input boosting, the wordline is pulled down to 9mV. Wordline and memory input boosting achieves functionality at the expense of an additional supply voltage.

An even better way to improve the $I_{on}/I_{off}$ at lower voltage is to use a concept similar to dynamic boosting of the wordlines.The reverse of the charge pump concept is to apply negative voltages to the wordlines when not being accessed. A negative voltage applied to the gate reduces the leakage current through a NMOS devices. Reduction of leakage current through the idle pull-down devices improves $I_{on}/I_{off}$. Figure 6.27 shows the schematic using level converters and a third power supply ($V_{SS}$) to apply a negative voltage to the read access NMOS transistors. For $V_{gs}$=-250mV, the idle current is 44 times lower, which helps to mitigate bitline leakage effects.

For the worst case input vector, $W_{pre}$ of 50μm maintains an output-high of 90mV (Figure 6.28). For the worst case output-low, $\overline{RBL}$ is driven down to 61mV, a vast improvement over the original precharge scheme without negative wordlines. However, the $I_{on}/I_{off}$ of the bitline is still not large enough for output-low to be 10% of $V_{DD}$. Also, this design requires a triple-well process and additional negative supply voltage.

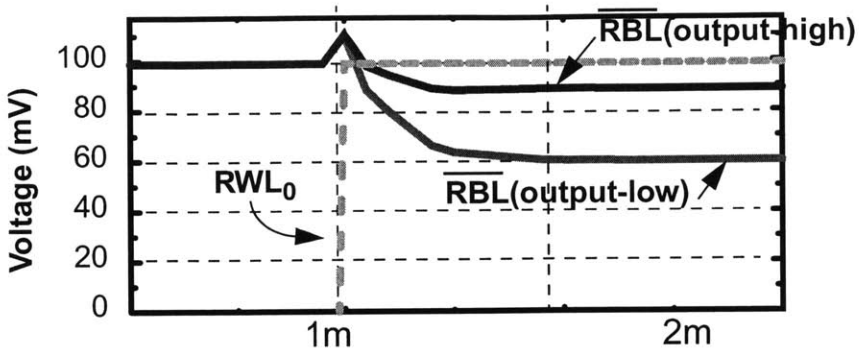


**Figure 6.28: Simulation of the bitline when negative voltages are applied to the read access NMOS transistors when they are not selected.**

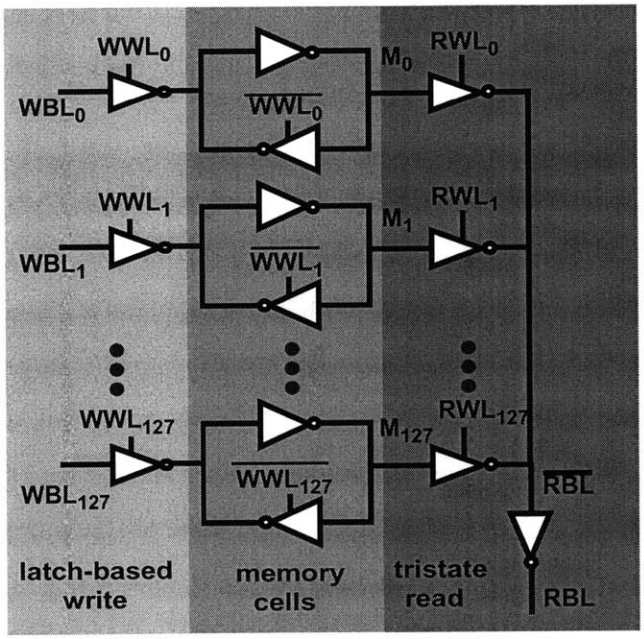


**Figure 6.29: Tristate-read access and Latch-based write access memory.**

In reality, if the gate-to-source voltage decreases to below -200mV, then Gate-Induced Drain Leakage (GIDL) effects appear [14]. GIDL results when a negative voltage on the gate causes high tranverse and lateral electric fields that cause the drain current to increase, thereby cancelling out any benefit of lowering the gate voltage.

A tristate-read scheme using static CMOS tristate cells is proposed (Figure 6.29). When the wordline is high, the tristate passes the data onto the bitline. Otherwise when the wordline is low, the tristate output is high impedance.

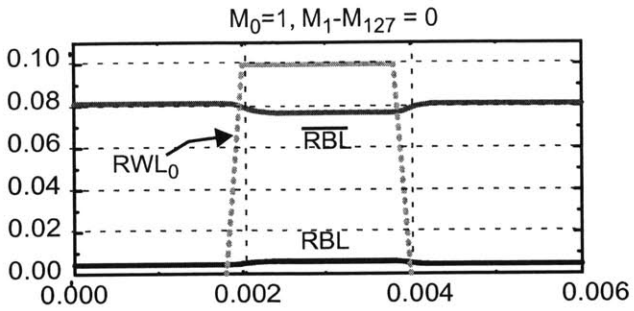


**Figure 6.30: Tristate-based read access also suffers from bitline leakage effects. The worst case input vector for output-high is $\overline{M}_0$=1 and $\overline{M}_1$-$\overline{M}_{127}$=0.**

Even though the tristate gate itself works at 100mV, when connected together as in Figure 6.29, the readbitline suffers from a low $I_{on}/I_{off}$ ratio. The worst bitline leakage for output-high occurs when $M_0$=1 and $M_1$=$M_{127}$=0. The $I_{on}/I_{off}$ ratio at RBL is extremely low due to all of the leakage current through the 127 idle tristate buffers. Figure 6.30 shows a simulation of RBL for the worst-case input vector for a typical transistor process corner.

A read bitline scheme that scales to low voltages and works at process corners is the hierarchical-read bitline (Figure 6.31). A hierarchical bitline is segmented by using a 2-to-

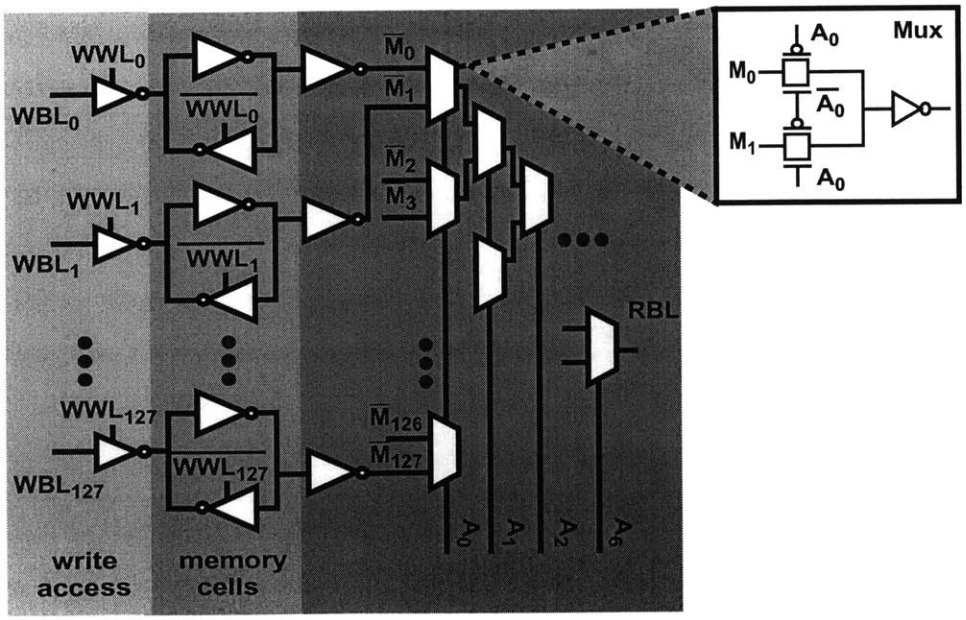


**Figure 6.31: Hierarchical-read access and Latch based write design allows for subthreshold operation.**

1 mux-based approach. The selectors to the muxes are the read-address inputs and the data from the memories is hierarchically passed down through the muxes to the output. In order to avoid stack effect and sneak leakage paths, inverters are inserted between each level of hierarchy. Figure 6.32 shows a waveform of RBL for the worst case leakage. By creating a hierarchical bitline, the effect of parallel leakage for each level of hierarchy is reduced and the design is less affected by process variations.

One 128W memory bitline has 128 memory bit cells and 127 muxes. To save area, the muxes are daisy-chained and arrayed as seen in Figure 6.33. A decoder generates the write wordlines (WWL) which are routed to the bitline horizontally. The Read Addresses ($A_0$-

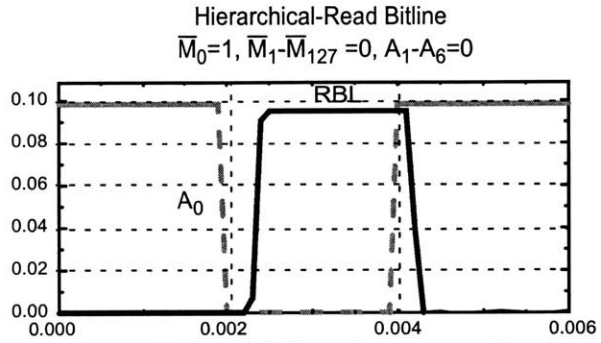


**Figure 6.32: The hierarchical-read bitline does not suffer from parallel leakage effects.**

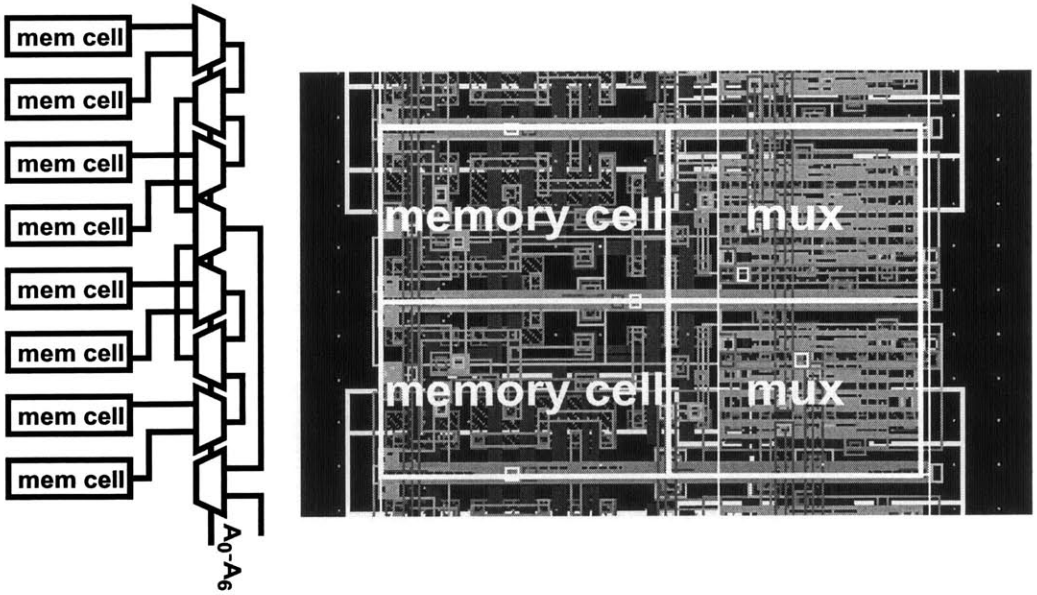


**Figure 6.33: A schematic showing how the muxes are daisy-chained and arrayed to minimize area as seen in the layout.**

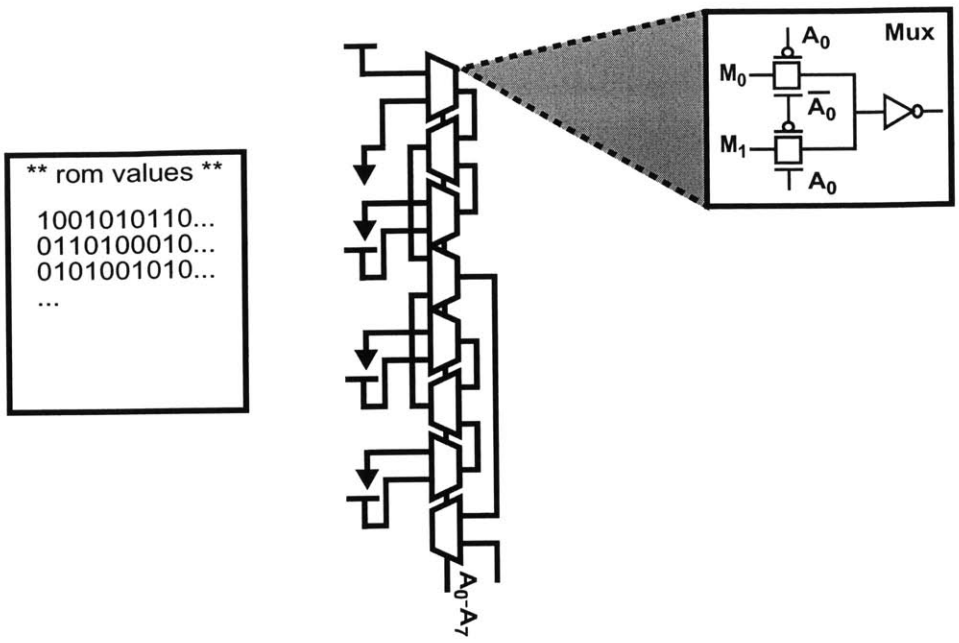


**Figure 6.34: A text file with ROM values is inputted to a Skill code ROM generator to create a hierarchical mux-based ROM.**

$A_6$) are routed vertically up to the selectors of the muxes. Custom Skill scripts array the memory bit cells, the muxes, the decoder logic and buffers for each data memory block. Eight custom 128Wx16b blocks make up the scalable data memory in the FFT processor. More details about Skill coding are found in [74].

## 6.5 Subthreshold read-only memories (ROMs)

Two 256Wx32b ROMs store twiddle factors in the FFT processor. The ROMs utilize the same hierarchical bitline design as the data memory. Instead, the inputs to the muxes are either tied to $V_{DD}$ or ground depending on the value stored in the ROM. The two twiddle ROMs are created using a custom Skill script that reads in a text file for ROM values (Figure 6.34).

## 6.6 Subthreshold control logic

The rest of the circuitry for the FFT processor is created using a subthreshold standard cell library. The subthreshold standard cell library is a combination of static CMOS 1.8V standard cells and subthreshold custom cells that are designed for subthreshold operation. Table 6.2 is a description of modifications made to a1.8V standard cell library.

**Table 6.2: Changes to allow for subthreshold operation**

| Logic-Cell Name | Function |
|---|---|
| Full Adder | This cell takes in three inputs (A,B,CI) and has two outputs (SUM, CO). This cell is re-designed with transmission gates. |
| Half Adder | This cell takes in two inputs (A,B) and outputs the SUM. This cell is re-designed with transmission gates. |
| 2-input NOR, 2-input OR | These cells perform the NOR and OR of two inputs (A,B). The PMOS devices are re-sized for minimum-voltage operation and because they are susceptible to parallel leakage and stack effect. |
| Positive and Negative Edge Registers | The registers take in a clock and D input and output is Q and $\overline{Q}$. They are resized to be functional at ultra-low voltages and at process corners. These proved to be more reliable at low voltages. |
| 3-input and 4-input XOR, XNOR, NAND, NOR, AND, OR | Do not use. |

## 6.7 Design flow

The design flow for the energy-scalable FFT is modified to use the subthreshold standard cell library and to include custom Baugh-Wooley multipliers and custom memory banks (128Wx16b) and ROMs (256Wx16b). The main architectural difference between the subthreshold FFT and the scalable FFT is that the subthreshold FFT has one data memory, while the scalable ASIC has two. When the FFT block is used as an accelerator in a microsensor node subsystem, only one dedicated memory buffer is necessary. Additionally, the subthreshold FFT does not employ the variable memory-size energy-scalable hook.

The *MATLAB* benchmarking done for the scalable FFT is re-used for the subthreshold FFT. The verilog is modified for one data memory and Synopsys *Design Compiler* is used to create the RTL verilog. Cadence *Silicon Ensemble* is used to place and route small blocks, such as the system control logic (counters, FSMs), datapath (two's complement adders and subtractors), and memory control blocks (parity bit address selectors). Custom

skill scripts create the layout for the memory, ROM and Baugh-Wooley multipliers. Each small block as well as the custom memory and BW multipliers is verified using *hspice* at 100mV for the TT process corner and at 250mV for the FS and SF process corners.

### 6.7.1 Skill code place-and-route and functional verification

Skill scripts perform place-and-route of the synthesized and custom blocks to create the layout of the four modules: datapath, data memory, twiddle ROMs and system-level control logic. Skill is a LISP based programming language which Cadence developed to perform any commands through the command interpreter window (CIW). A Skill tutorial is found in [74].

The *hspice* simulator is not used for functional verification of each of the four modules. Because functionality of synthesized and custom blocks are verified at 100mV operation using *hspice*, only verification the connectivity and functionality of the four modules is necessary. Epic's *nanosim* at 1V operation performs high-level verification.

Using skill, the four modules are assembled with pads to create the entire FFT layout. The subthreshold FFT is verified at 1V using Epic's *nanosim* simulator. The pads are custom designed for operation at low-voltages.

### 6.7.2 Hardware Testing

A PCB tests the functionality of the subthreshold FFT chip (Figure 6.35). The Textronix pattern generator has 2-5V output which is level converted down to 500mV to interface with the subthreshold FFT chip. The outputs of the subthreshold FFT chip are level converted up to 5V using LM311 comparators. The Textronix Logic Analyzer is able to detect voltages down to 80mV, but the probes have an input pull-up resistance of 20KΩ which is not overcome by the subthreshold currents. The level converting schematics are seen in Figure 6.36.

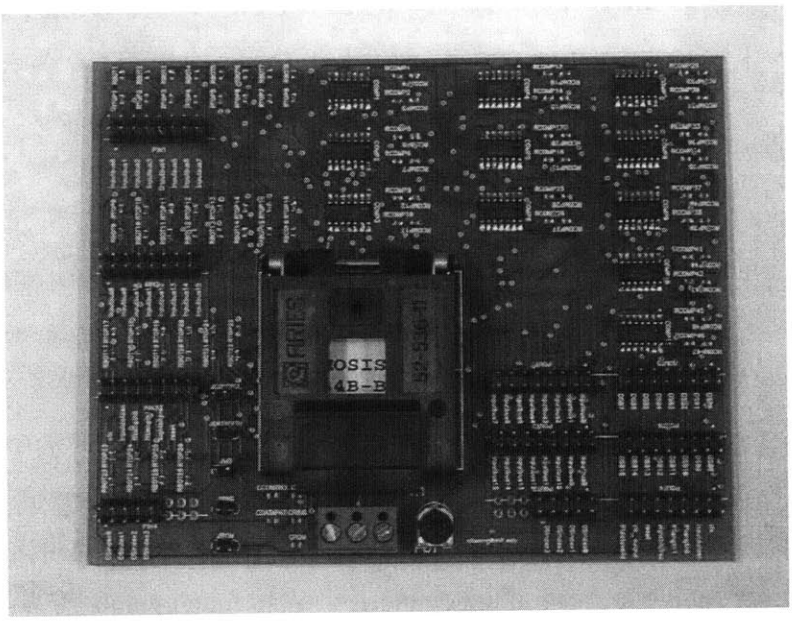

**Figure 6.35: This PCB allows for current measurements and functional verification of the Subthreshold FFT. Additional voltage level converters for the inputs and outputs are shown.**

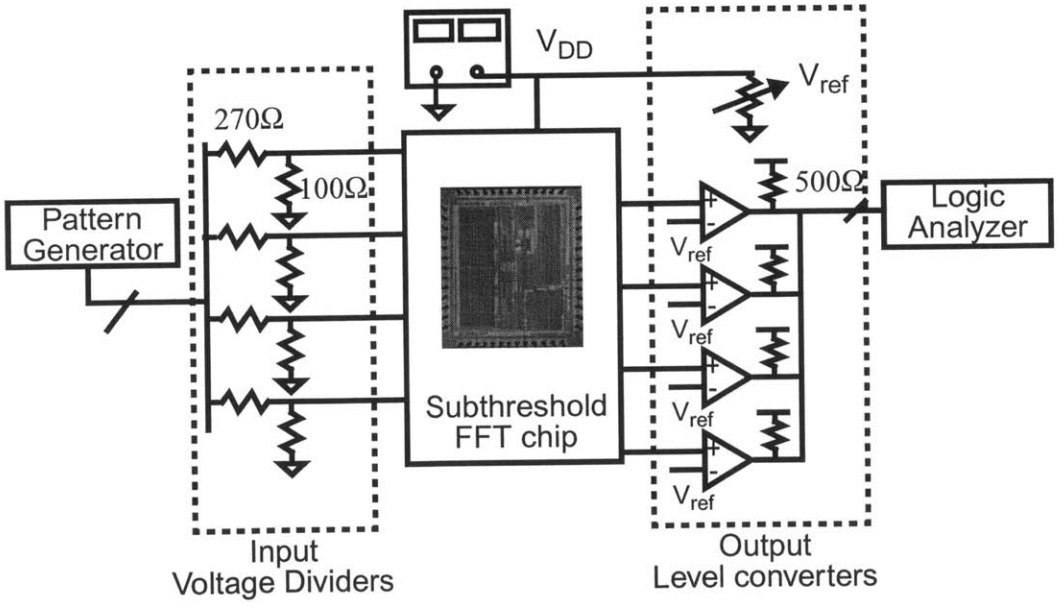


**Figure 6.36: Input and output level-converting circuits for testing the subthreshold FFT.**

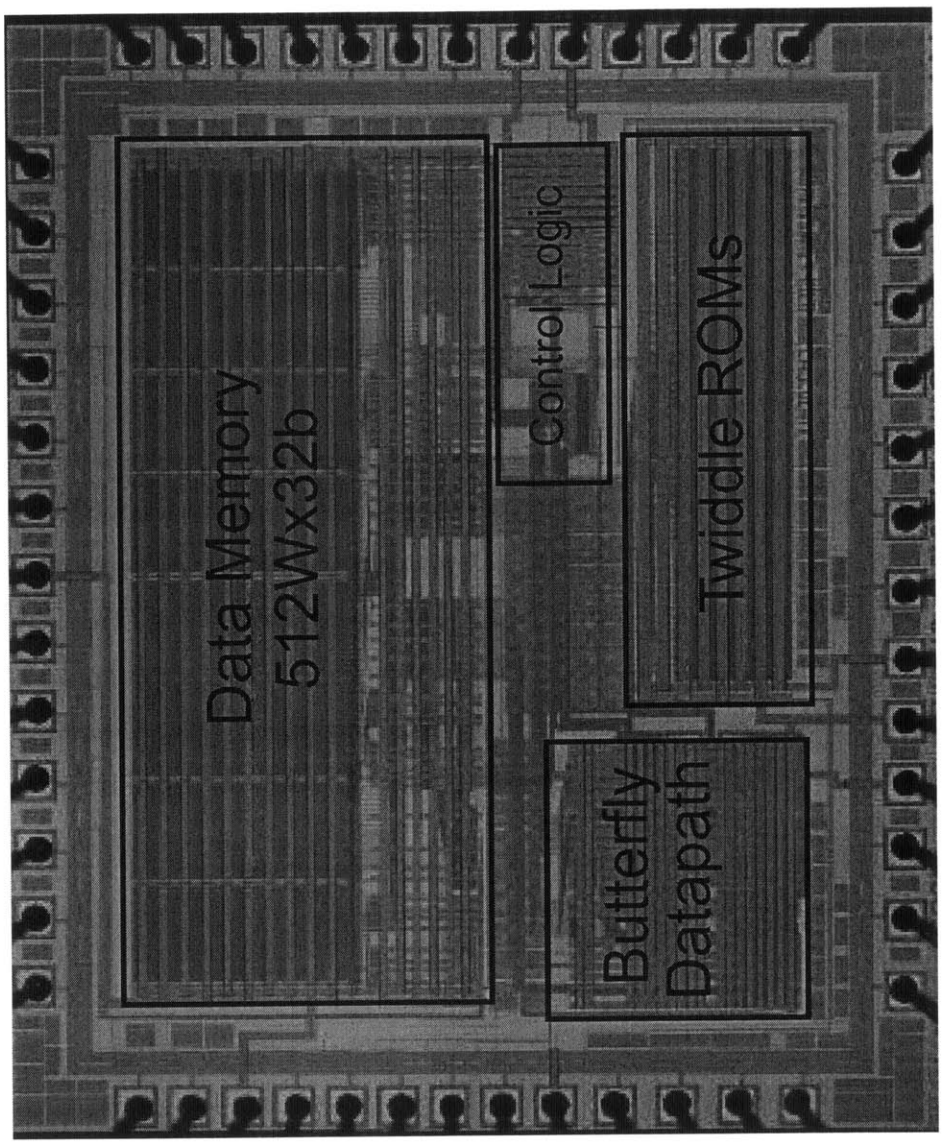


**Figure 6.37: Die Photograph of the subthreshold FFT chip.**

## 6.8 Performance measurements

The energy-aware FFT containing 627,000 transistors is fabricated in a standard 6-metal 0.18μm process. The FFT processor occupies 2.6 x 2.1 $mm^2$. It is fully functional at 128, 256, 512, and 1024 FFT lengths, 8 and 16 bit precision, for voltage supplies 180mV to 0.9V and for clock frequencies of 164Hz to 6MHz.

Figure 6.37 shows the die photograph of the subthreshold FFT chip. The chip's functional blocks (memory, butterfly datapath, twiddle ROMs and control logic) are clearly delineated.

The lowest voltage supply for correct operation is 180mV with a clock speed of 164Hz. The power dissipated at 180mV is 90nW for 1024-point, 16-bit operation. Figure 6.38 shows an oscilloscope plot of various outputs, (the clock, 2 output bits and the data-ready signal) from the FFT chip operating at 180mV. However, the minimum supply voltage does not correspond to the optimal operating point which minimizes energy dissipation.

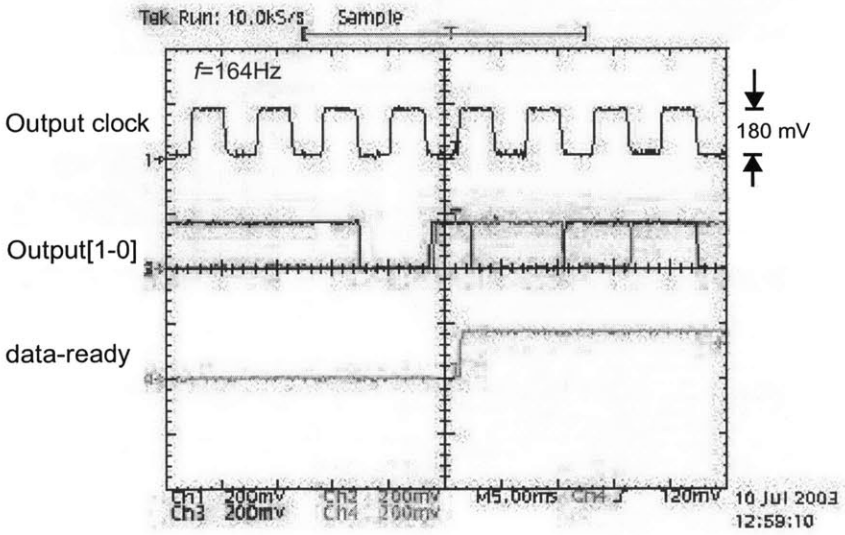


**Figure 6.38: Scope Plot showing outputs from the FFT chip at 180 mV operation.**

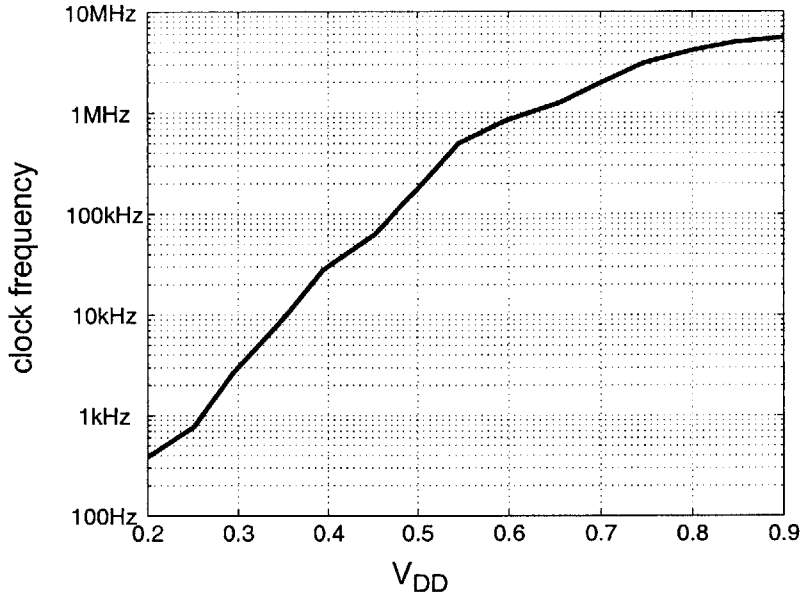


**Figure 6.39: Clock frequency as a function of $V_{DD}$.**

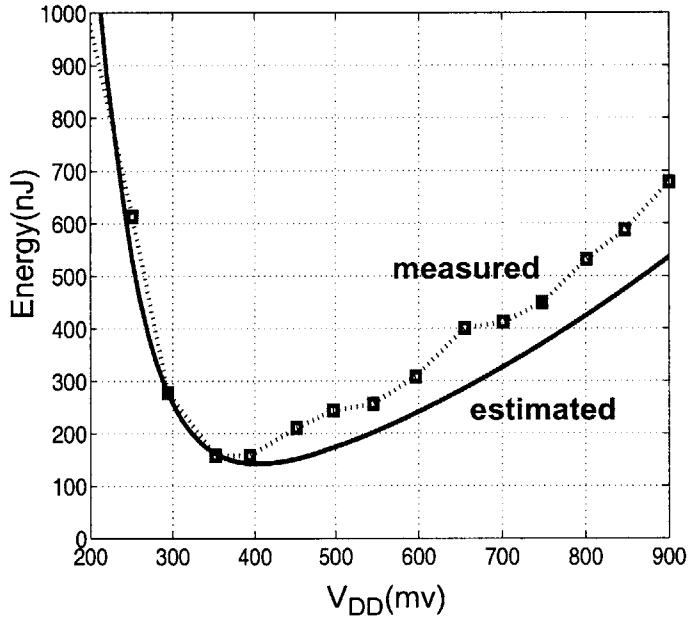


**Figure 6.40: Energy dissipation as a function of $V_{DD}$ for N=1024-point and 16-bit processing. The optimal operating point for minimal energy dissipation is at $V_{DD}$=350 mV.**

Figure 6.39 shows the clock frequency of the FFT as a function of $V_{DD}$. This plot shows the exponential relationship between clock frequency and voltage supply. Figure 6.40 shows the measured and estimated energy dissipated as a function of $V_{DD}$. The energy is measured for the 16-bit, 1024-point FFT over $V_{DD}$=200-900mV for the clock frequencies specified in Figure 6.39.

The optimal operating point for minimal energy dissipation is at $V_{DD}$=0.350V. The power dissipated at 350mV is 590nW at a clock frequency of 9.62kHz. The energy dissipated at the optimal operating point is 155nJ/FFT. The two curves show that there the estimation predicts leakage energy well, but not switching energy. Switching energy was estimated from *nanosim*.

This subthreshold FFT architecture allows for energy-scalable hooks. The chip performs 128 to 1024 point FFTs with 8- and 16-bit precision. Figure 6.41 shows the energy dissipated as supply voltage is scaled for all operating points. The minimum voltage supply for the 16-bit processor is at 350mV which is lower than the minimum voltage of the 8-bit processor (400mV). A decrease in the activity factor of the 8-bit processor causes a higher minimum supply operation. As the activity factor decreases, there is an increase in the effect of leakage energy, causing the optimal operating point to be at a higher $V_{DD}$.

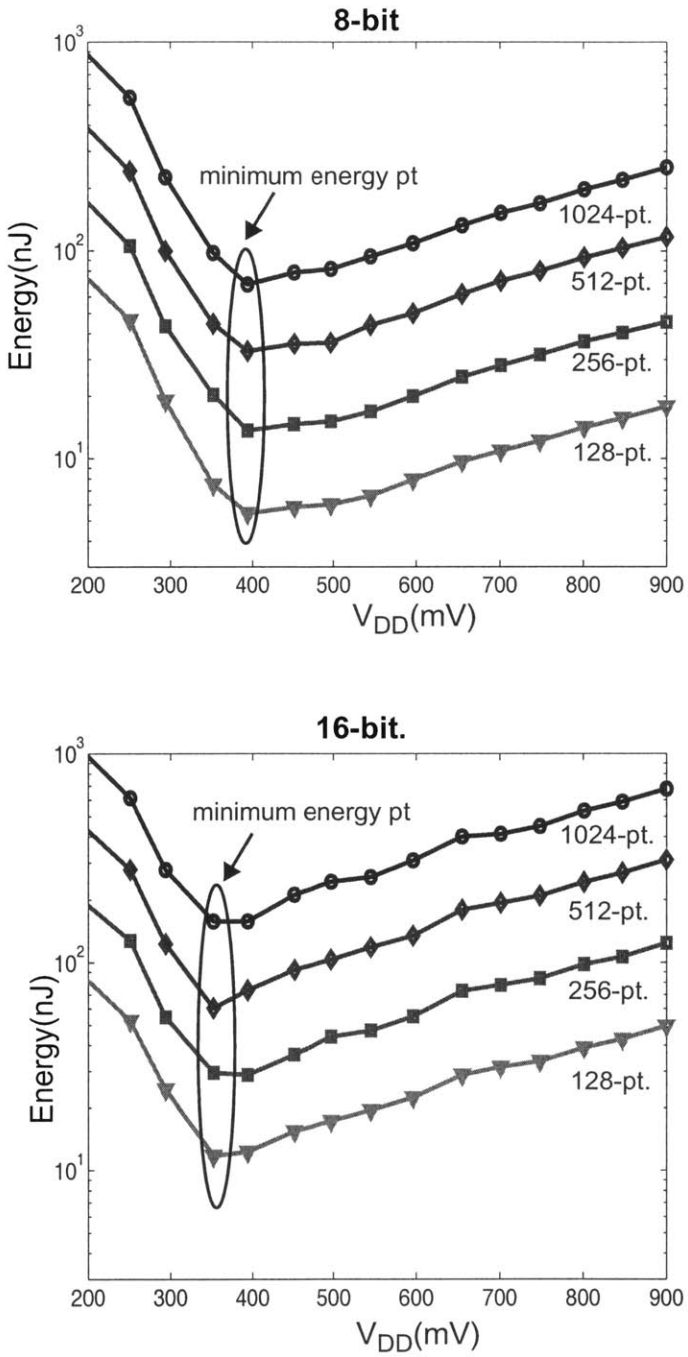


**Figure 6.41: Energy vs. $V_{DD}$ for 8- and 16-bit FFT processing.**

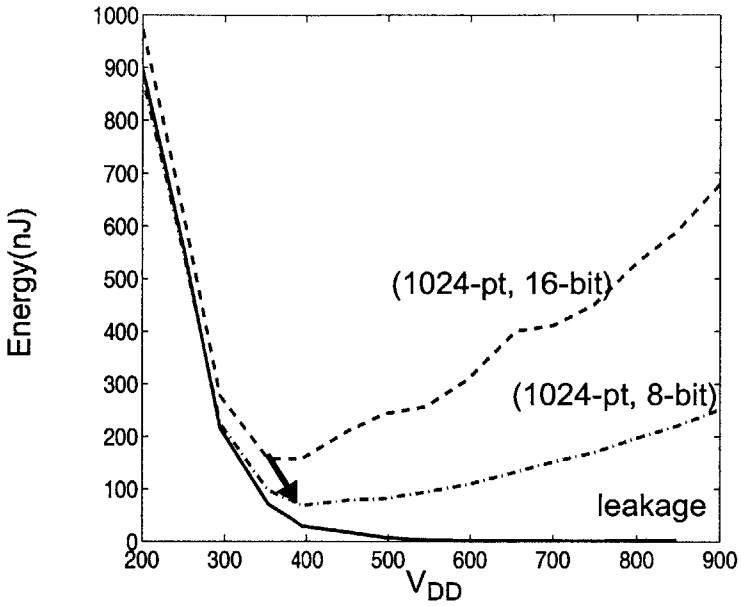


**Figure 6.42: Energy breakdown of total energy dissipation of the FFT between idle and active energy.**

Figure 6.42 shows the total measured energy dissipated for both 8- and 16-bit processing at N=1024-pt and the measured leakage energy. This graph shows that for $V_{DD}$<400mV, the leakage energy begins to dominate the total energy. Because at 8-bit precision the activity factor is lower, the optimal operating point is at a higher $V_{DD}$. At the optimal operating points for both 8-bit and 16-bit processing, the idle leakage energy is approximately 45% of the total energy dissipated.

When comparing this design to other FFT implementations, it is important to normalize over different technologies. Table 6.3 shows a list of parameters from other FFT implementations. The energy is normalized for a common workload which is a 512-point CVFFT. This workload is equivalent to a 1024-point FFT on the subthreshold chip. The energy is normalized for technology and datapath by the following normalization procedure

$$\text{Normalized Energy} = Power \times Time \times \frac{0.18\mu m}{Tech} \tag{6.1}$$

**Table 6.3: Specifications from other FFT processors normalized for a 512-point CVFFT.**

| Processor | CMOS Tech (μm) | Datapath width (bits) | Power | Clock Freq | Execution Time | Energy/ FFT |
|---|---|---|---|---|---|---|
| Wosnitza [79] | 0.5 | 32 | 6W | 66 MHz | 36 μs | 77.5μJ |
| DoubleBW [17] | 0.35 | 24 | 8W | 128 MHz | 4.5 μs | 18.5μJ |
| DSP-24 [15][16] | 0.5 | 24 | 3.5W | 100 MHz | 9.5 μs | 11.9μJ |
| Spiffee [4] | 0.7 | 20 | 9.5mW | 16 MHz | 149 μs | 362nJ |
| Subthreshold custom | 0.18 | 16 | 576nW | 9.62 kHz | 266 ms | 158nJ |
| Subthreshold custom | 0.18 | 8 | 792nW | 27.8 kHz | 92 ms | 68nJ |

The energy scalability of all implementations from this thesis is demonstrated in Figure 6.43. The measured energy of the StrongARM, the Energy-Scalable ASIC and the custom subthreshold chip are shown for FFT length between 128 to 1024-point and for both 8- and 16-bit precision. These results show that our implementation has better energy scalability characteristics, thus making this design globally more energy-efficient than other

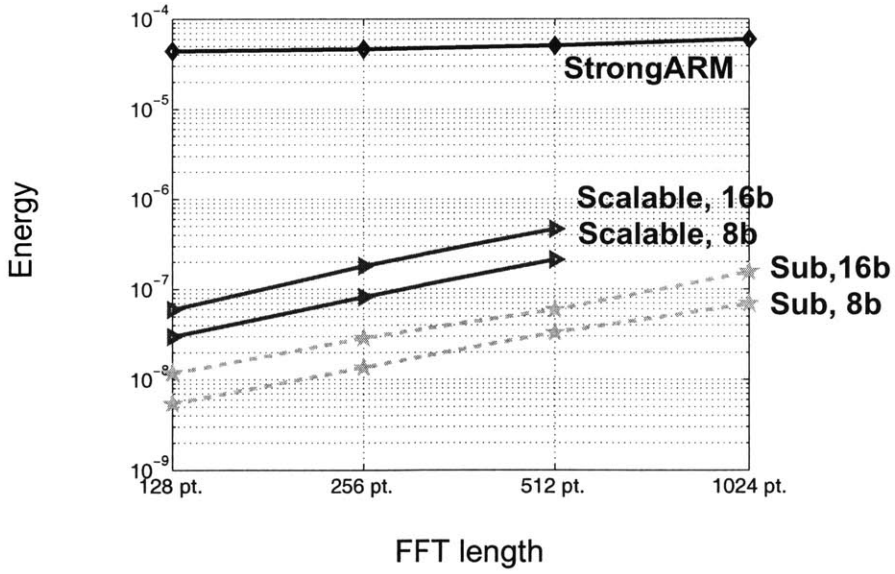


**Figure 6.43: Energy comparison of all FFT implementations.**

designs. However, the voltage scalability of the subthreshold FFT makes it the most energy-efficient FFT processor, due to the ability to operate at subthreshold voltage supplies.

Comparing the three implementations in this thesis (low-power microprocessor, Energy-Scalable ASIC and custom subthreshold IC), the subthreshold IC is 350 times more energy efficient than the low-power microprocessor implementation and 8 times more energy efficient than the ASIC.

## 6.9 Summary

Voltage scaling achieves a large amount of energy savings. Because the microsensor application does not require high speed clocking, by relaxing the clock frequency and scaling down the supply voltage leads to reducing energy dissipation. However, there is a limit to voltage scaling, because lower clock frequencies allow leakage energy to exceed switching energy.

The energy-performance contours show that the estimate of the optimal supply voltage for the FFT is 400mV, which is below the threshold voltage. This motivates the need for a subthreshold implementation of the FFT. We propose logic and memory design techniques that allow circuits to operate as low as 100mV in simulation for a typical transistor. All circuits also function as low as 250mV at the worst case process corners.

For logic design, effects such as parallel leakage, stacking, sneak leakage, and the effects of process variations at cell interfaces appear. New subthreshold cells are designed that are able to function at low voltages, and custom design is done for the Baugh-Wooley multiplier.

Ultra low-voltage memory design is fairly unexplored. Typical memory design uses a 6T SRAM design, with sense amplifier read-access and dynamic precharge lines. For low voltage operation and with process variations, new designs are needed to circumvent bitline leakage and data dependent bitline effects. Various read and write access designs are analyzed, and a hierarchical-read bitline is chosen.

The subthreshold FFT is implemented using subthreshold techniques and is fabricated in a standard 0.18μm process. A modified synthesis toolchain is used to design the FFT

processor. The FFT processor functions for 128, 256, 512 and 1024 points and for 8- and 16-bit processing. The FFT operates at voltage supplies between 180mV and 0.9V and for clock frequencies of 164Hz to 6MHz. The FFT processor functions down at 180 mV with a clock frequency of 164Hz. The optimal operating point for N=1024 point and 16-bit is at 350mV at a frequency of 9.62 kHz and dissipates 590nW power. At the optimal operating point the energy dissipation is 155 nJ/FFT.

When comparing all implementations of the FFT in this thesis, the subthreshold FFT achieves over 350x energy savings over the StrongARM implementation and 8x energy savings over the ASIC implementation.

# Chapter 7

# Conclusions

Energy-aware design is proposed in this thesis to provide energy-scalable hardware hooks that allow the node to be energy-efficient for a variety of operating scenarios. By enabling hooks such as bit-precision scaling and FFT length scaling, energy and quality can be scaled efficiently on the same hardware. Deep voltage scaling is enabled by designing new circuits that operate in deep subthreshold. Our processor is able to scale down to 180mV, using a standard CMOS process technology. Deep voltage scaling allows the processor to operate at the optimal operating point for minimal energy dissipation.

## 7.1 Summary of contributions

In sensor applications, the real-valued Fast Fourier Transform (FFT) is a design driver for energy-aware digital signal processor (DSP) design. Three implementations of the FFT are created and their energy and performance are compared and contrasted: The StrongARM low-power microprocessor, the Energy-Scalable ASIC and the custom subthreshold IC. The main results of these implementations are described and future work and research proposed.

The FFT is benchmarked on the Intel StrongARM processor as part of system-level energy-aware analysis of the Line of Bearing (LOB) estimation application for source-tracking. In general, because communication is more expensive than computation, energy is saved by performing local signal processing at the sensor node. Therefore, by performing the LOB locally at the sensor node, the node needs only send the LOB estimate , rather than transmitting the raw data to the basestation. Also, since all sensors are homogeneous, the sensor cluster operates as a multiple microprocessor system. DVS is used with intelligent system partitioning of the FFT task to all of the StrongARM processors in the sensor cluster. This leads to a 40% energy savings. A tremendous amount of energy savings is achieved by using a dedicated FFT processor to reduce switched capacitance and to operate at lower voltage supplies.

In an FFT processor, FFT specific energy-aware hooks are designed into the architecture to provide energy-efficiency over a variety of scenarios. The hooks designed into the FFT are variable FFT length (128, 256, 512 and 1024 points) and variable bit-precision (8 and 16-bit). Ensemble of point solutions method proposes that multiple optimized point solutions be designed and low-overhead routing be used to route the data to the appropriate point solution. A better approach is the reuse of point solutions method, where common hardware is shared and reused for multiple point solutions. The control logic in the reuse method is more complex, but there is less area and transistor overhead.

The Baugh Wooley multiplier example showed that bit-precision scaling is enabled by recognizing that the MSB quadrant can be reused for a reduced bit-precision multiplication without significant additional circuitry. The LSB inputs of the multiplier and the multiplicand are routed to the MSB quadrant. Additional gating logic such as LSB input gating was needed to prevent any undesired switching in the unused portions of the multiplier. This thesis showed that in combinational logic and in memories it is feasible to introduce bit-precision scaling by using the reuse of point solutions method.

Variable memory-size was another energy-scalable architecture introduced in this thesis. To perform a small FFT length FFT, only a small memory is needed. Simulations show that performing a 128-point FFT on the architecture designed for 1024-point FFTs is not energy-efficient and the overhead of the 1024-point FFT memory leads to higher energy dissipation. By providing different size memories, the 128-point FFT can be performed on the small memories, without significant overhead in area or energy.

Energy simulations of the FFT show that the reuse of point solutions method achieves 2.7 times energy savings through a variable FFT length and variable bit-precision implementation.

The Energy-Scalable FFT is implemented using an ASIC flow that incorporates a 1.8V standard cell library and memory generators. The Energy-Scalable ASIC is fabricated in a 0.18-μm process and operates at 128, 256, and 512 points and 8- and 16-bit precision. At a 1.5V voltage supply the FFT clock frequency is 4.8MHz. The ASIC is able to operate down to 1V, and because the ASIC does not scale down further, a custom implementation is needed to scale voltage supply for further energy-savings.

Other important energy-scalable hooks in this thesis are the supply voltage and clock frequency. Scaling into deep subthreshold enables circuits to operate at the optimal operating point that minimizes energy. A variable activity factor benchmark circuit is used to explore the optimal operating point. The voltage supply and threshold voltage of the circuit is changed for the range of $V_{DD}$=100mV-1V and $V_{th}$=0-800mV, and the energy and performance of the circuit is observed. The results from the variable activity factor benchmark circuit show that operation in the subthreshold regime minimizes energy. Also the trends in the simple benchmark circuit can easily be extended to large systems. Because the activity factor of a circuit is directly related to the ratio of leakage and active energy dissipation, the energy-performance contours of any system is estimated through the activity factor. These results are extrapolated for the FFT system, where the that the optimal voltage supply is estimated to be around 380mV, which is below the threshold voltage of the nominal 0.18μm device.

For a standard logic process, dynamic threshold voltage scaling is not possible. This thesis explores optimum supply voltage for a fixed threshold voltage process. The results from the variable activity factor circuit show that for high activity factors, e.g. $\alpha$>0.1, the optimum point is below the threshold voltage. Implicit in the discussion is minimum supply voltage operation. For certain activity factors (e.g. $\alpha$=1), the optimal supply voltage is the minimum supply voltage. Therefore, it is important to be able to demonstrate circuits that operate as low as 200mV.

In order to operate at such low voltage supplies new circuit design and analysis are needed. In this thesis, traditional circuits that operate at nominal supply voltages (> 1V), do not function well at ultra-low voltage supplies. In this thesis new ways to evaluate circuits operating at low voltage are introduced. The min-max sizing curves show a new way to understand optimum sizing of circuits, and the concept of $I_{on}/I_{off}$ for CMOS circuits give intuition into the functionality of circuits in subthreshold. This thesis also exposed problems such as parallel leakage, sneak leakage paths, and stacked devices that effect functionality at 100mV.

Ultra-low voltage memories is a fairly unexplored field. Many memory configurations, such as the 6T SRAM, that are commonly used for nominal 1V operation were evaluated in subthreshold and shown not to function in subthreshold due to reduced $I_{on}/I_{off}$. The

hierarchical mux-based memory is proposed for subthreshold operation of both RAM's and ROM's and is shown to operate in simulation as low as 100mV.

The custom Subthreshold FFT is designed for minimum voltage operation to study the optimal operating point. The subthreshold FFT is implemented using subthreshold techniques and using a modified synthesis toolchain. The IC is fabricated in a 0.18μm process and functions for 128, 256, 512, and 1024 points and for 8- and 16-bit precision. The Subthreshold FFT processor functions down to 180 mV with a clock frequency of 164Hz. The optimal operating point for N=1024 point and 16-bit is at 350mV at a frequency of 9.62 kHz and dissipated 155 nJ/FFT.

## 7.2 Future work

Because there is little work in energy-awareness this concept can be applied to many existing applications to achieve further energy-efficiency. This is particularily applicable in applications where performance specifications can be traded-off for additional battery lifetime. Only through energy-awareness can global energy-efficiency be achieved.

For the microsensor application, not only the signal processing functions such as beamforming and the FFT should be energy-aware, but the entire system. The system blocks from the A/D, to the radio module, and all DSP functions, require architectures that contain new energy-scalable knobs to provide scalability. For example, a bit-precision scalable A/D that can provide low-precision/low-energy samples all the way to high-precision/high-energy samples is needed.

Energy savings is also achieved by identifying functions that are application specific hardwired for large energy savings. Comparing the performance of the ASIC to the StrongARM microprocessor, the ASIC is able to achieve orders of magnitude energy-efficiency by minimizing the overhead of computation. Algorithmic profiling of sensor applications can be done to identify other functions that are commonly used such as beamformers and filters. In addition, a low-power DSP is needed that is able to perform those signal processing tasks that are not performed by the accelerators. Besides software transformations, the DSP requires a new energy-aware instruction set and scalable hardware that can perform scalable computation such as variable bit-precision scaling.

Additionally, dynamic voltage scaling and clock frequency scaling of the StrongARM microprocessor leads to dramatic energy savings, even though the StrongARM was only able to scale down to 0.85V operation. With the subthreshold FFT results, it is clear that all aspects of the system be designed with deep dynamic voltage scaling considerations down into the subthrehsold region. New designs for subthreshold radio module, DSP, and A/D are needed.

Analysis into the optimal operating point for each module is needed. It is very likely that the different system level blocks will have different optimal supply voltages and clock frequencies, therefore complicating the interfaces between the blocks. New bus interfacing techniques and system-level architectures involving asyncronous concepts are needed for multiple clock frequency domains to avoid timing conflicts between blocks. Level-converting circuits are needed to interface between two modules that operate at different supply voltages. In order to have a completely optimal system for minimal energy dissipation, multiple clock and power supply management is needed. The power management block will need to monitor the environment or user requirements, in order to optimally control the clock frequencies and supply voltages of all blocks.

Further work is needed to verify the energy-performance contours by fabricating the benchmark circuit that allows for variable supply voltage and variable threshold voltage and variable activity factor. Our circuit already allows for variable activity factor through the selector inputs. Variable threshold voltage is achieved by applying forward and reverse bias voltages to the back gates of the devices. Simulations show that back-biasing only provides up to 300mV threshold voltage swing. In order to achieve threshold voltages across a wide range, multiple threshold CMOS is also needed. Multiple instances of the benchmark circuit, which varying threshold voltages with back-biasing achieves a large threshold voltage range. Also, to fabricate this circuit, a triple well process is needed.

Further analysis should also look into the effects of process variations and environmental conditions on the optimal supply and threshold voltage operation. Especially in the subthreshold region, the effects of lithography, device mismatch, temperature, and threshold voltage variation will have a huge impact on the energy dissipated and the performance. The optimal operating point may vary a lot, and therefore in practice feedback control circuitry should be relied on to set the operating point. Therefore, in order to oper-

ate at the optimal supply and threshold voltage levels, new control circuits are needed. The critical path for a circuit can be replicated to monitor the circuit delays as they change for different supply and threshold voltages. However, research is needed to find new methods to monitor the energy dissipated and to dynamically adjust supply voltages and threshold voltages to find the minimum energy dissipation.

There is much future work opportunities in the area of architectures for subthreshold circuits. One interesting research direction is the use of pipelining and massively parallel architectures that increases the activity factor of a circuit, and requires minimum supply voltage operation.

The results from the subthreshold chip open the door to many new research opportunities in minimum supply voltage design and subthreshold circuit considerations. In this thesis, a subthreshold library and methodology is described for minimum supply voltage operation, that used a small subset of circuits. This small subset only includes two-input logic gates, positive and negative registers and adders. Although, this small set of cells was adequate for creating an FFT that functions at ultra-low voltages, a more expansive standard cell library may provide further opportunites to optimize for minimal energy dissipation. Therefore, future work in a complete subthreshold standard cell library is needed.

Also important is the coupling of optimal device sizing to attain minimum energy dissipation. In this thesis, the device sizing is one way to attain minimum voltage operation. However, large sizing may also contribute additional power dissipation. Further energy savings can be achieved by tailoring the device sizing through a parameterized standard cell library.

Because low-voltage memories is fairly unexplored, there is much opportunity for further research in memory designs that operate at low-voltages and across process corners. Also, subthreshold memory layout generators are needed for subthreshold SRAM and ROMs.

Also important is the need for a subthreshold circuit simulator that can verify the functionality of large systems in subthreshold. *hspice* is too slow to run larger circuits. *nanosim* can simulate large netlists in a reasonable time, but will not correctly model the devices for supply voltages below 1V. Either modification of current simulators or a new circuit

simulator is needed to verify large systems running at ultra low-voltages below 1V operation and to estimate the power dissipation of the circuits.

Finally, simulation shows that there is no reason why circuits cannot operate at voltage supplies even lower than 180mV. As technology scales, the devices should be able to operate at lower voltages and have better performance. However, lower threshold voltages also imply higher leakage currents. Looking into future trends and the impact on optimal voltage scaling will expose the trade-offs of energy and performance for future technologies.

Comparing the three implementations in this thesis (low-power microprocessor, Energy-Scalable ASIC and custom subthreshold IC), the Subthreshold FFT is 700 times more energy efficient than the low-power microprocessor implementation and 8 more energy efficient than the ASIC. The Subthreshold FFT is able to achieve minimum energy dissipation operation for a variety of operating points.

The Subthreshold FFT chip's ability to operate as low as 180mV opens up exciting new fields of research. New research initiatives are starting to bring low-voltage operation into existing applications. Besides the many research ideas described here, there are many new opportunities for innovation in subthreshold systems, scalable architectures, and minimal voltage operation circuits.

# Appendix A

# Interfacing to the FFT ASIC

This document contains specifications for using the real-valued FFT ASIC (RVFFT). This ASIC was designed as part of the micro-Adaptive Multi-domain Power-aware Sensors (μAMPS) project at MIT. The RVFFT can perform 128, 256, or 512-point RVFFT's, 64, 128, or 256-point complex-valued FFT's, and 8- or 16-bit precisions.

## A.1 RVFFT overview

The real-valued FFT transforms data from the time-domain to the frequency-domain. The RVFFT is preferred over the complex-valued FFT because the data source is real-valued. The ASIC has special hardware that allows us to perform one N-point RVFFT from one N/2-point CVFFT. For example, a 512-point RVFFT is done with a 256-point CVFFT, then performing backend calculations to calculate the 512-point RVFFT from the CVFFT coefficients.

A block diagram of the interface to the RVFFT ASIC is shown in Figure A.1 In this section, I elaborate on the input/output pins and their functions. Als included are the physical description of the chip, timing diagrams and a short description about using the FFT ASIC in a tripwire application.

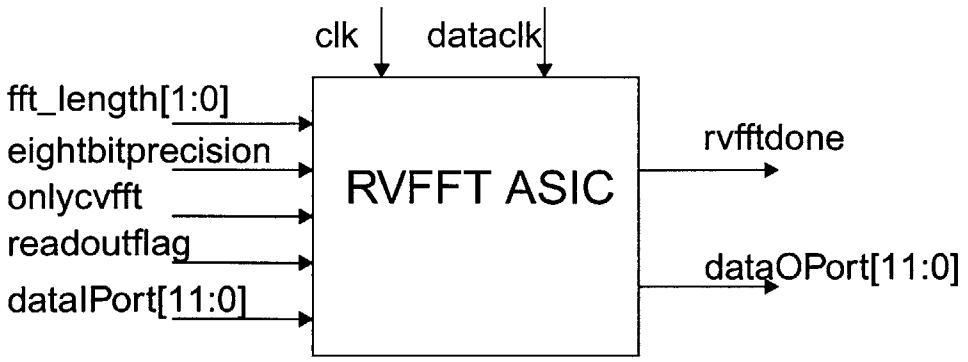


**Figure A.1: Block diagram of the RVFFT ASIC.**

The data is clocked in at the *dataclk* rate, and the FFT is performed at the *clk* rate. There is a ping-pong memory in the FFT ASIC. Data is clocked into one buffer via the *dataIPort* until enough samples are collected to perform the RVFFT. For example, if we want to perform a 128-point RVFFT, then when 128 samples are collected in buffer #1, the FFT starts processing that data and the next sample is stored in buffer #2. This is the function of the ping-pong memory.

The registers *fft_length* and *eightbitprecision* are loaded at the positive edge of the *dataclk*. Therefore these values cannot be changed until after the data is finished being loaded into the data buffer. When the *onlycvfft* input is high the N/2-point CVFFT is performed instead of the N-point RVFFT. This assumes that the input has complex-valued data, rather than real-valued data. For the RVFFT, for each *dataclk* cycle a new real-valued sample is clocked into memory. The values are clocked in on the negative edge of the clock cycle. For the CVFFT, first the real-valued part of the sample is clocked in, and then the imaginary-valued part of the sample.

In one clock cycle, one RVFFT butterfly calculation is performed. The number of cycles to perform an N-point RVFFT is $N/4*\log_2 N$. The number of cycles to perform a N-point CVFFT is $N/2*\log_2 N$. Therefore, after one N-point block is loaded into memory, it takes $N/4*\log_2 N$ cycles before the rvfftdone flag is high and the RVFFT coefficients are pushed onto the *dataOPort*.

**Table A.1: Number of cycles**

| RVFFT Length | Number of Butterflies/Cycles |
|---|---|
| 128-point | 224 |
| 256-point | 512 |
| 512-point | 1152 |

The *dataOPort* clocks out the complex-valued coefficients. The coefficients are clocked out on the positive edge of the *clk* signal. They are outputted according increasing frequency bins. The first value is always the real-value of the complex coefficient and the second value is the imaginary-value.

There is an additional debugging signal *readoutflag*. This helps to debug the memory. If *readoutflag* is high, then the data stored in memory is automatically read-out onto dataOPort.

The following table has the list of inputs and outputs and their descriptions.

**Table A.2: Input/Output descriptions**

| | | |
|---|---|---|
| dataclk | input | The data I/O clock. At the negative clock edge of dataclk, the value of dataIPort is clocked into memory |
| clk | input | The FFT processor clock. One RVFFT butterfly is performed per clock cycles. |
| reset | input | The reset signal resets the control logic when high. |
| fft_length[1:0] | input | Determines the length of the RVFFT to be performed. If only-cvfft is 0 and fft_length is<br>01 - 512-point RVFFT<br>10 - 256-point RVFFT<br>11 - 128-point RVFFT<br>If onlycvfft is 1 and fft_length is<br>01 - 256-point CVFFT<br>10 - 128-point CVFFT<br>11 - 64-point CVFFT |
| eightbitprecision | input | Determines the bit-precision of the RVFFT. If eightbitprecision is<br>0 - 16-bit datapath<br>1 - 8-bit datapath |
| onlycvfft | input | When onlycvfft is high, then the N/2-point CVFFT is per-formed. When onlycvfft is low, then the N-point RVFFT is per-formed. |
| dataIPort[11:0] | input | 12-bit data input port. The data from this port is clocked into memory at the negative edge of the dataclk signal. |
| dataOPort[11:0] | output | 12-bit data output port. Once the FFT is done, the rvfftdone flag goes high, and the FFT coefficients are placed on this port at the positive edge of the clk signal. |
| datastart | input | When the datastart input is high, the FFT ASIC will clock in data on the negative edge of the dataclk. Otherwise the FFT ASIC will idle |
| rvfftdone | output | rvfftdone=1 will indicate that the FFT is done and that the coef-ficients are on the dataOPort bus |
| readoutflag | input | Debugging pin. When it is high, then the contents of memory are automatically read-out onto the dataOPort bus. When it is low, then the CVFFT and RVFFT are performed. |

There are multiple supply voltages to the chip. Each module of the chip has its own power supply to facilitate measuring the current drawn by the module. The entire chip shares one ground (VSS). A list of supply names is given in Table A.3.

**Table A.3: Supply voltage names**

| | |
|---|---|
| VSS | ground signal |
| VDD | pad ring power supply |
| VDD33 | pad ring power supply |
| VDDrfp1 | data memory buffer #1 power supply |
| VDDrfp2 | data memory buffer #2 power supply |
| VDDdatapath | datapath power supply |
| VDDcontrol | control logic power supply |
| VDDrom | rom power supply |

## A.2 Chip Specifications

### 1.2.1 Chip Pin-out

This section will describe the pin-outs. The RVFFT ASIC has a LCC52 package and the pinouts are shown in Table A.4.

**Table A.4: FFT ASIC Chip Pinout**

| Pin | Type | Signal Name | Pin | Type | Signal Name | Pin | Type | Signal Name |
|---|---|---|---|---|---|---|---|---|
| 1 | supply | VSS | 21 | input | dataIPort[2] | 41 | supply | VDD33 |
| 2 | input | fft_length[0] | 22 | input | dataIPort[1] | 42 | supply | VDDrom |
| 3 | input | fft_length[1] | 23 | input | dataIPort[0] | 43 | supply | VDD |
| 4 | supply | VDDrfp2 | 24 | supply | VDDrfp1 | 44 | output | dataOPort[6] |
| 5 | input | dataIPort[11] | 25 | output | rvfftdone | 45 | output | dataOPort[7] |
| 6 | input | dataIPort[10] | 26 | input | datastart | 46 | output | dataOPort[8] |
| 7 | input | dataIPort[9] | 27 | supply | VSS | 47 | output | dataOPort[9] |
| 8 | input | dataIPort[8] | 28 | input | reset | 48 | output | dataOPort[10] |
| 9 | supply | VSS | 29 | input | onlycvfft | 49 | output | dataOPort[11] |
| 10 | input | dataIPort[7] | 30 | supply | VDDrfp1 | 50 | supply | VDDrfp2 |
| 11 | input | dataIPort[6] | 31 | output | dataOPort[0] | 51 | input | readoutflag |
| 12 | supply | VDD | 32 | output | dataOPort[1] | 52 | input | eightbitprecision |
| 13 | supply | VDDcontrol | 33 | output | dataOPort[2] | | | |
| 14 | supply | VDDdatapath | 34 | output | dataOPort[3] | | | |
| 15 | supply | VDDdatapath | 35 | output | dataOPort[4] | | | |
| 16 | input | dataclk | 36 | output | dataOPort[5] | | | |
| 17 | output | dataclk_output | 37 | output | clk_output | | | |
| 18 | input | dataIPort[5] | 38 | input | clk | | | |
| 19 | input | dataIPort[4] | 39 | supply | VDDcontrol | | | |
| 20 | input | dataIPort[3] | 40 | supply | VDDrom | | | |

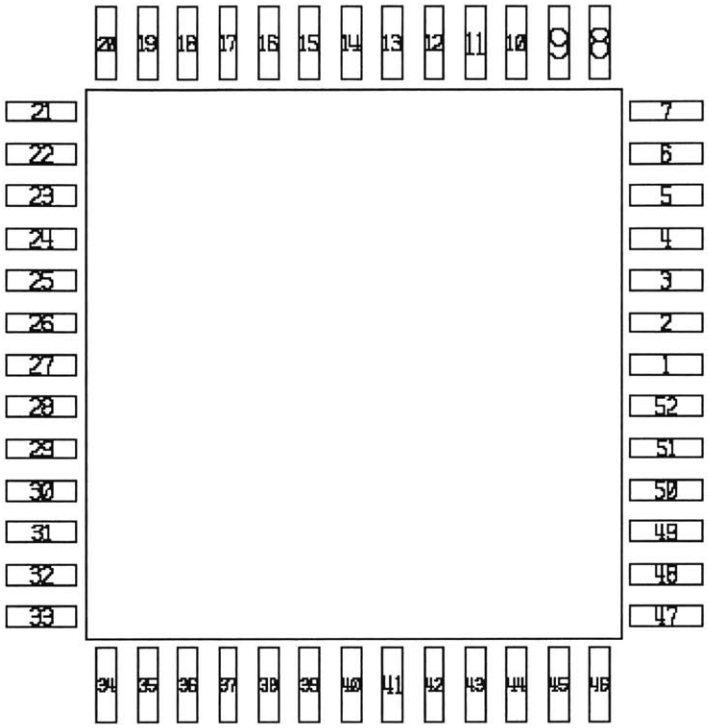


**Figure A.2: FFT ASIC package footprint.**

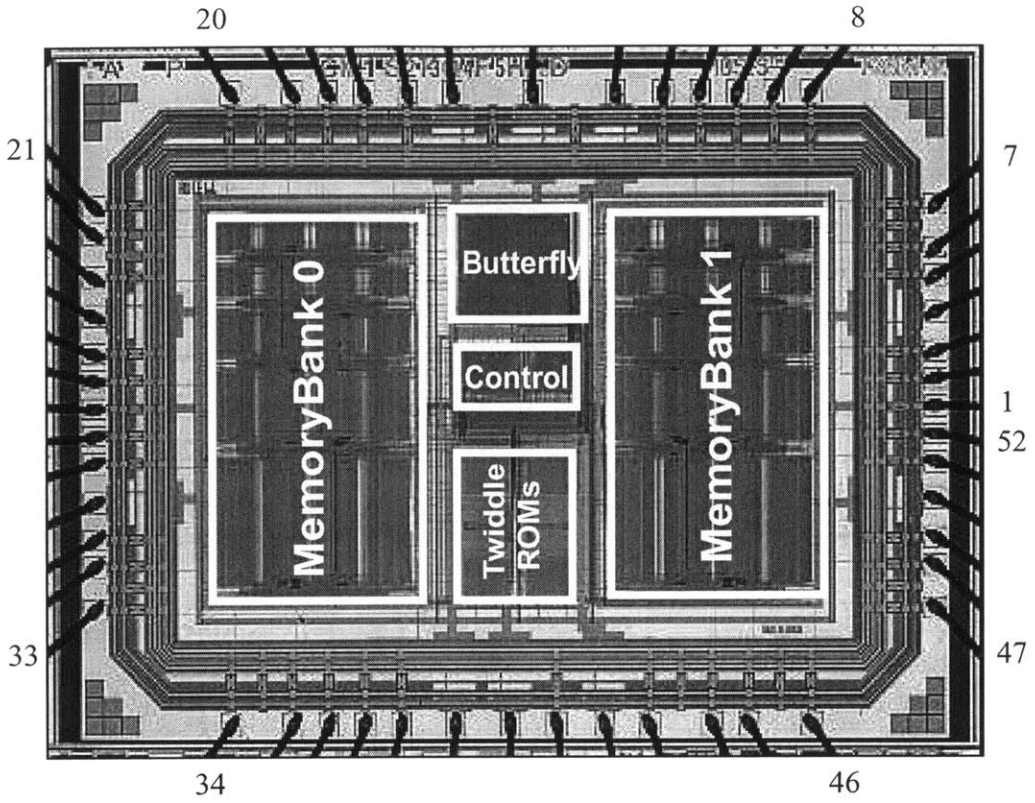


**Figure A.3: Die photograph of the 128 to 512 point, 8 and 16 bit scalable FFT chip.**

The energy-aware FFT is implemented in a 0.18 μm process (MOSIS T28M-AE). Figure A.3 shows a die photograph of the scalable FFT chip. The chip's functional blocks (data memory, butterfly, control, and ROMs) are clearly delineated. The FFT system is fully verified up to 512 point, for 8 and 16 bit operation at 1.5V and for clock speed of 4.8 MHz.

Table A.5 shows the measured energy dissipation of the FFT at 1.5V operation. The measured energy dissipation verifies the architecture's ability to scale energy and quality. The energy at the low quality point (128-point, 8b) is 12 times lower than the energy at the high quality point (512-point, 16b).

**Table A.5: Measured energy dissipation from the FFT ASIC.**

| | 8-bit | 16-bit |
|---|---|---|
| 128 point | 46nJ | 81nJ |
| 256 point | 121nJ | 216nJ |
| 512 point | 304nJ | 564nJ |

The FFT ASIC operates down to 1V before failing. In order to minimize energy, a new FFT processor design is needed that operates at supply voltages below 1V. Research in the next two chapters focuses on estimating the optimal supply voltage for minimal energy dissipation and circuits that allow supply voltage scaling below the threshold voltage of the devices.

## 1. Timing Diagrams

The input is placed on the dataIPort[11:0] bus. When datastart is 1 at the positive clock edge, the value on dataIPort is clocked into memory on the next negative edge of dataclk. If onlycvfft=0, then a RVFFT is performed, meaning that the inputs s[n], are real-valued. This means that for each clock cycle of dataclk, there is a new real-value sample.

RVFFT: onlycvfft=0, readoutflag=0

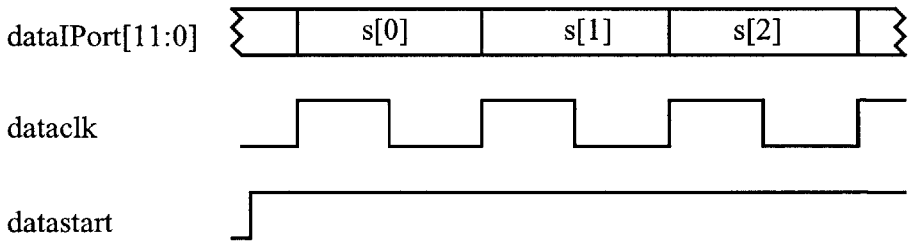


**Figure A.4: Timing diagram of dataIPort for RVFFT.**

If onlycvfft=1, then a CVFFT is performed meaning that the inputs s[n] are complex-valued. This means that two clock cycles of dataclk are needed to store one complex-

value. The real-value of the sample is clocked in the first clock cycle, and the imaginary-value is clocked in the second clock cycle.

RVFFT: onlycvfft=1, readoutflag=0

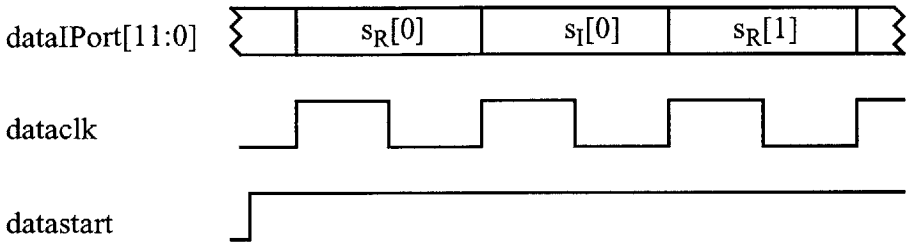


**Figure A.5: Timing diagram of dataIPort for RVFFT.**

The timing of the signals dataclk and clk are important. It is important that clk be constrained so that the FFT is completed before the next block of data is finished. The reason for this constraint is practical. If the time to complete the FFT ($t_{FFT}$) is greater than the time to input one block of data ($t_{inputdata}$) then when the next sample of data is ready to be inputted, then there will not be any data memory available for storage. This means an extra external buffer is needed. This means that a constraint is placed on the dataclk and clk so that:

$$t_{fft} < t_{inputdata}$$

we can define $t_{fft}$ and $t_{inputdata}$ as

$$t_{fft} = (\#\text{ of butterflies}+\#\text{ of samples})/f_{clk} \quad \text{and} \quad t_{inputdata} = \#\text{ of samples}/f_{dataclk}$$

$t_{fft}$ is the number of butterflies and the number of samples, which are the number of cycles to perform one FFT and then the number of cycles to output all of the FFT coefficients onto the bus. $t_{inputdata}$ is just the number of samples. These two relations leads to the constraint that

$$f_{fft} > f_{inputdata} \cdot \frac{(\#\text{ of butterflies}+\#\text{ of samples})}{\#\text{ of samples}}$$

For the RVFFT, the number of samples is N and the number of butterflies is N/ $4*\log_2 N$, so that means the constraint for the RVFFT function is

$$f_{clk} > f_{dataclk} * (\log_2 N/4+1)$$

Since we showed that for a supply voltage of 1.5V, the FFT processor clock frequency $f_{clk}$=4.8MHz, then $f_{dataclk}$<1.37MHz for 1024-point RVFFT.

The reset signal goes high to reset the FFT ASIC. It must be high for at least one clock cycle of the dataclk to ensure that all of the FSMs are reset.

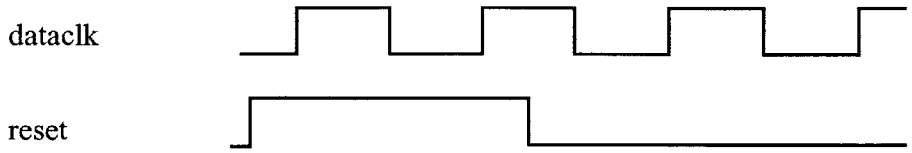


**Figure A.6: Timing diagram of dataIPort for RVFFT.**

After one block of data is loaded into memory, the next sample is stored in the other memory bank. Immediately, the FFT is started and after performing all of the butterflies, the signal rvfftdone goes high, and the data is outputted on the positive edge of the clk signal. Note that there is a one cycle delay before the output is valid. Each complex FFT coefficient is represented by two 12-bit values. The first value is the real-value and the second is the imaginary-value. In Figure A.7 the FFT coefficients are given by $S(f)=S_R(f)+jS_I(f)$.

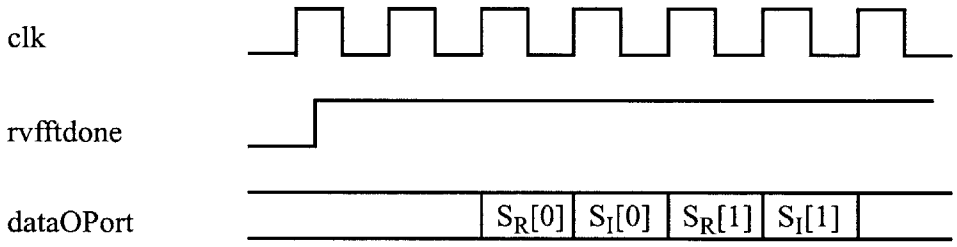


**Figure A.7: Timing diagram of dataOPort and rvfftdone for RVFFT.**

## 2. Testing Schematics

The schematics for the PCB is given by Figure A.8. The inputs to the chip are generated by the Textronix pattern generator (PG). The PG interfaces directly to the chip through the headers. Each signal is paired with the ground references. The output the chip are captured by the Textronix Logic Analyzer (LA). The logic analyzer interfaces directly

to the chip through the headers. Again, each output signal is paired with the ground reference.

The supply voltages are connected directly to a power supply. There are jumpers for each power supply and for current measurements. Also the power supplies have large capacitors to reduce power supply noise.

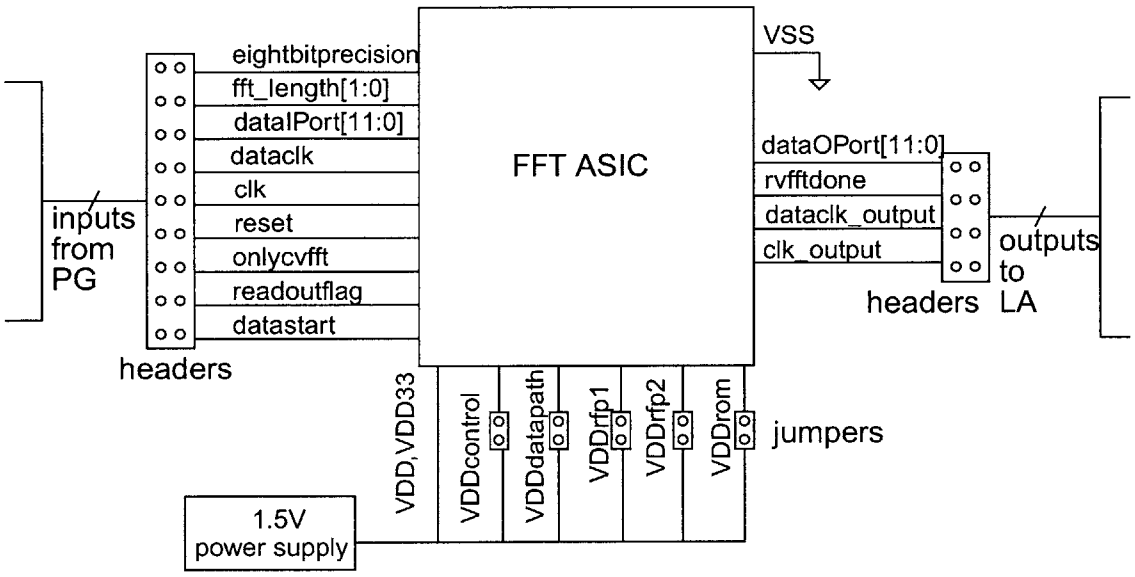


**Figure A.8: PCB schematic to interface to the Textronix Pattern Generator and Logic Analyzer.**

## 3. Tripwire usage

The RVFFT ASIC can be used with a DSP or μProc for a tripwire application. The tripwire application is described by the block diagram in Figure A.9. First a RVFFT is performed on the incoming 512-point block of sensor data using the dedicated ASIC.

Then the data is streamed to a DSP that performs the power spectrum and peak-picking algorithm.

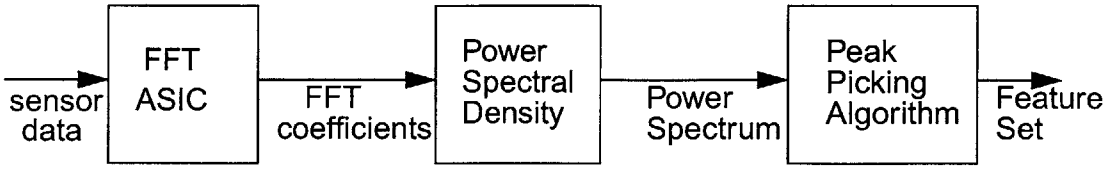


**Figure A.9: Tripwire block diagram**

# Appendix B

# Glossary of Acronyms

This appendix is a list of acronyms found in this thesis.

- A/D: Analog to Digital.
- ASIC: Application Specific Integrated Circuit
- BF: Beamforming
- BW: Baugh Wooley
- CIW: Command Interpreter Window
- CMOS : Complementary MOSfet
- CVFFT: Complex-valued Fast Fourier Transform (see FFT)
- DA: Distributed Arithmetic
- DFT: Discrete Fourier Transform
- DSP: Digital Signal Processor
- DVS: Dynamic Voltage Scaling
- FPGA: Field Programmable Gate Array
- FFT: Fast Fourier Transform. The FFT can be either complex-valued or real-valued. In this thesis, assume that N-point FFT refers to an N-point real-valued FFT
- HLA: Harmonic Line Association
- ISM: Industrial, Scientific and Medical (refers to unlicensed radio bands for Bluetooth or Wireless LAN's. Communicating at the ISM bands does not interfere with industrial, scientific or medical devices)
- LOB: Line of Bearing
- LMS: Least Mean Squares
- LUT: Look-up table
- MSB: Most Significant Bit
- MVDR: Minimum Variance Distortionless Response

- StrongARM or SA-1100: Intel low-power StrongARM microprocessor, the SA-1100 model.
- μAMPS: micro-Adaptive Multi-domain Power-aware Sensors project at MIT.
- $V_{DD}$: voltage supply of the circuit
- $V_{th}$: threshold voltage of the devices
- VTC: Voltage-Transfer Curve

# References


[1] Advanced RISC Machines Ltd., *Advanced RISC Machines Architectural Reference Manual,* Prentice Hall, New York, 1996.

[2] R. Amirtharajah, S. Meninger, J. Mur-Miranda, A. Chandrakasan, and J. Lang, "A micropower programmable DSP powered using a MEMS-based vibration-to-electric energy converter," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers,* February 2000, pp. 362-363.

[3] G. Asada, M. Dong, T. S. Lin, F. Newberg, G. Pottie, and W. J. Kaiser, "Wireless Integrated Network Sensors: Low Power Systems on a Chip," in *Proc. European Solid-State Circuits Conference,* The Hague, Sept. 22-24,1998.

[4] B. Baas, "A Low-Power, High-Performance, 1024-Point FFT Processor," *IEEE Journal of Solid-State Circuits,* vol. 34, no. 11, March 1999, pp. 380 -387.

[5] E. Bidet, D. Castelain, C. Joanblanq, and P. Senn, "A Fast Single-Chip Implementation of 8192 complex point FFT," in *IEEE J. of Solid-State Circuits,* vol. 30, no. 3, March 1995, pp. 300-305.

[6] M. Bhardwaj, R. Min and A. Chandrakasan, "Quantifying and Enhancing Power-Awareness of VLSI Systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 6 , Dec. 2001, pp. 757 -772.

[7] J. R. Budenske, R. S. Ramanuja, and H. J. Siegel, "On-Line Use of Off-Line Derived Mappings for Iterative Automatic Target Recognition Tasks and a Particular Class of Hardware Problems," in *Proc. of the Sixth Heterogeneous Computing Workshop,* April 1997, pp. 96 -110.

[8] J. Burr and J. Shott, "A 200mV Self-Testing Encoder/Decoder using Stanford Ultra-Low-Power CMOS," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 1994, pp. 84-85.

[9] F. Callias, F. Salchli, and D. Girard, "A set of four ICs in CMOS techology for a programmable hearing aid," *IEEE Journal of Solid-State Circuits,* vol. 24, no. 2, April 1989, pp. 301-312.

[10]A. Chandrakasan and R. Brodersen, *Low Power Digital CMOS Design,* Kluwer Academic Publishers, 1995.

[11]D. Cohen, "Simplified Control of FFT Hardware," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-24, December 1976, pp. 577-579.

[12]J. W. Cooley, and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," in *Math. of Comput.,* vol. 19, April 1965, pp. 297-301.

[13]P. David, P. Emmerman, and S. Ho, "A scalable architecture system for automatic target recognition," *Digital Avionics Systems Conference,* October 1994, pp. 414 -420.

[14]V. De, Y. Ye, A. Keshavarzi, S. Narendra, J. Kao, D. Somasekhar, R. Nair, and S. Borkar, "Techniques for Leakage Power Reduction," in *Design of High-Performance Microprocessor Circuits,* ed. A. Chandrakasan, W. Bowhill, and F. Fox, IEEE Press, IEEE, Inc., New York, 2001, pp. 46-61.

[15]Digital Signal Processing Architectures Inc., *DSP-24 Datasheet, Reference DSPA-DSP24DS,* 2002.

[16]Digital Signal Processing Architectures Inc., *DSP-24 User's Guide*, Reference DSP24UG0355, 2002.

[17]DoubleBW Systems BV, *Data Sheet PowerFFT PCI32-Card,* 2002.

[18]D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," *Proc. ACM MobiCom '99*, August 1999, pp. 263 - 270.

[19]M. Ettus, "System capacity, latency and power consumption in multihop-routed SS-CDMA wireless networks," in *Proc. Radio and Wireless Conference (RAWCON 98)*, August 1998, pp. 55-58.

[20]R. Gonzalez, B. Gordon, and M. Horowitz, "Supply and Threshold Voltage Scaling for Low Power CMOS," *IEEE Journal of Solid-State Circuits,* vol. 32, nvolo. 8, August 1997, pp. 1210-1216.

[21]J. Goodman and A. Chandrakasan, "An energy-efficient reconfigurable public-key cryptography processor," *IEEE Journal of Solid-State Circuits,* vol. 36, no. 11, November 2001, pp. 1808-1820.

[22]C. T. Gray, W. Liu, T. Hughes, and R. Cavin, "The design of a high-performance scalable architecture for image processing applications," in *Proc. of the Int. Conf. on Application Specific Array Processors,* September 1990, pp. 722 -733.

[23]V. Gutnik and A. Chandrakasan, "Embedded Power Supply for Low-Power DSP," *IEEE Transactions on VLSI Systems,* vol. 5, no. 4, December 1997, pp. 425 -435.

[24]S. Haykin, J. Litva, and T. J. Shepherd, *Radar Array Processing*, Springer-Verlag, 1993.

[25]S. He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Proc. of IEEE 1998 Custom Integrated Circuits Conference,* 1998, pp. 131-134.

[26]W. R. Heinzelman, "Application-Specific Protocol Architectures for Wireless Networks," Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA June 2000.

[27]W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient Communication Protocol for Wireless Microsensor Networks," in *Proc. of the 33rd Annual Hawaii Int. Conf.,* January 2000, pp. 3005 -3014.

[28]ITRS roadmap, http://public.itrs.net

[29]M. Izumikawa, H. Igura, K. Furuta, H. Ito, H. Wakabayashi, K. Nakajima, T. Mogami, T. Horiuchi, M. Yamashina, "A 0.25-μm CMOS 0.9-V 100-MHz DSP core," *IEEE J. of Solid-State Circuits,* vol. 32, no. 1, January 1997, pp. 52-61.

[30]J. Kao, M. Miyazaki, and A. Chandrakasan, "A 175-mV multiply-accumulate unit using an adaptive supply voltage and body bias architecture," in *IEEE Journal of Solid-State Circuits,* vol. 37, no. 11, November 2002, pp.1545 -1554.

[31]J. Karn, R. Katz, and K. Pister, "Next Century Challenges: Mobile Networking for Smart Dust," *Proc. ACM MobiCom '99*, August 1999, pp. 271-278.

[32]H. Kim and K. Roy, "Ultra-low power DLMS adaptive filter for hearing aid applications," in *Int. Symp. on Low Power Electronics and Design*, August 2001, pp. 352 - 357.

[33]P. Ko, J. Huang, Z. Liu, and C. Hu, "BSIM3 for Analog and Digital Circuit Simulation," *Proc. of IEEE Symp. on VLSI Technology CAD,* January 1993, pp. 400-429.

[34]L. Lee, "Distributed Signal Processing," PhD. Thesis, Massachusetts Institute of Technology, Cambridge, MA, 2000.

[35]J.T. Ludwig, S. H. Nawab, A. P. Chandrakasan, "Low-power digital filtering using approximate processing," *IEEE Journal of Solid-State Circuits,* vol. 13, no. 3, March 1996, pp. 395-400.

[36]S. Magar, S. Shen, G. Luikuo, M. Fleming, and R. Aguilar, "An Application Specific DSP Chip Set for 100 MHz Data Rates," in *Int'l Conf. on Acoustics, Speech and Signal Processing,* vol. 4, April 1988, pp. 1989-1992.

[37]S. Meninger, T. O. Mur-Miranda, R. Amirtharajah, A. Chandrakasan, and J. Lang, "Vibration-to-electric energy conversion," *IEEE Trans. VLSI Systems,* Vol. 9, no.1, February 2001, pp. 64 -76.

[38]R. Merritt, "Planned U.S. sensor network targets terror threats." EETimes, 14 July 2003.

[39]R. Min, M. Bhardwaj, S-H. Cho, N. Ickes, E. Shih, A. Sinha, A. Wang, and A. Chandrakasan, "Energy-Centric Enabling Technologies for Wireless Sensor Networks," in *IEEE Wireless Communications*, vol. 9, no. 4, August 2002, pp. 28-39.

[40]R. Min, T. Furrer, and A. Chandrakasan, "Dynamic voltage scaling techniques for distributed microsensor networks," in *Proc. of the IEEE Comp. Society Workshop on VLSI*, April 2000, pp 43-46.

[41]M. Miyazaki, H. Mizuno, and K. Ishibashi, "A delay distribution squeezing scheme with speed-adaptive threshold-voltage CMOS (SA-Vt CMOS) for low voltage LSIs", *Int. Symp. on Low Power Electronics and Design*, August 1998, pp. 48 -53.

[42]R. A. Mucci, "A Comparison of Efficient Beamforming Algorithms", *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 22, no. 3, June 1984, pp. 548-558.

[43]P. S. Naidu, *Sensor Array Signal Processing,* CRC Press LCC, Boca Raton, 2001.

[44]National Semiconductor Corporation, *LMX 3162 Evaluation Notes and Datasheet,* April 1999.

[45]S. H. Nawab and A. V. Oppenheim, "Approximate Signal Processing," *Journal of VLSI Sig. Proc. Systems,* vol. 15, no.1, January 1997, pp. 177-200.

[46]C. Ng, "Design of a Power-scalable Digital Least-Means Square Adaptive Filter," M. Eng. Thesis, Massachusetts Institute of Technology, Cambridge, MA, June 2002.

[47]L. Nord and J. Haartsen, *The Bluetooth Radio Specification and the Bluetooth Baseband Specification,* Bluetooth, 1999-2000.

[48]J. O'Brian, J. Mather, and B. Holland, "A 200 MIPS Single-Chip 1K FFT Processor," in *IEEE Int'l Solid-State Circuits Conference,* vol. 36, 1989, pp. 166-167, 327.

[49]G. Ono and M. Miyazaki, "Threshold-Voltage Balance for Minimum Supply Operation," *IEEE Journal of Solid-State Circuits,* vol. 38, no. 5, May 2003, pp. 830-833.

[50]A. Oppenheim and R. Schafer, *Discrete-Time Signal Processing*. Prentice Hall, New Jersey, 1989.

[51]T. Pagarani, F. Kocan, D. Saab, and J. Abraham, "Parallel and scalable architecture for solving satisfiability on reconfigurable FPGA," in *Proc. of the IEEE Custom Integrated Circuits Conf.*, May 2000, pp. 147 -150.

[52]M. C. Pease, "Organization of large scale Fourier processors", *JACM*, vol. 16, July 1969, pp. 474-482.

[53]R. Preston, "Register Files and Caches," in *Design of High-Performance Microprocessor Circuits,* ed. A. Chandrakasan, W. Bowhill, and F. Fox, IEEE Press, IEEE, Inc., New York, 2001, pp. 285-308.

[54]W. Rabiner and A. Chandrakasan, "Network Driven Motion Estimation for Wireless Video Terminals," *IEEE Trans. on Circuits and Systems for Video Technology,* vol. 7, no. 4, August 1997, pp. 644-653.

[55]K. Roy, "Leakage power reduction in low-voltage CMOS designs," *1998 IEEE Int'l Conf. on Electronics, Circuits and Systems,* vol. 2, September 1998, pp. 7-10.

[56]P. Rudnick, "Digital Beamforming in the frequency domain," *J. Acoust. Soc. Amer.*, vol. 46, 1969, pp. 1089-1090.

[57]P. Ruetz, M. Cai, "A Real Time FFT Chip Set: Architectural Issues," in *Int'l Conf. on Pattern Recognition,* vol. 2, June 1990, pp. 385-388.

[58]B.J. Sheu, D.L. Scharfetter, P.-K. Ko, M.-C. Jeng, "BSIM: Berkeley short-channel IGFET model for MOS transistors," *IEEE Journal of Solid-State Circuits,* vol. 22, no. 4, August 1987, pp. 558 -566

[59]A. Sinha and A. Chandrakasan, "JouleTrack - A Web Based Tool For Software Energy Profiling," *Proc. of 38th Design Automation Conference,* June 2001.

[60]A. Sinha and A. Chandrakasan, "Energy Aware Software," *VLSI Design 2000*, Calcutta, India, January 2000, pp. 50 -55.

[61]A. Sinha and A. Chandrakasan, "Energy-Efficient Filtering using Adaptive Precision and Variable Voltage," *12th Annual IEEE ASIC Conference,* September 1999, pp. 327 -331.

[62]A. Sinha, A. Wang, and A. Chandrakasan, "Energy Scalable System Design," in *IEEE Trans. on VLSI Systems,* vol. 10, no. 2, April 2002, pp. 135-145.

[63]H. Soeleman, K. Roy, and B. Paul, "Sub-Domino logic: ultra-low power dynamic subthreshold digital logic," *14th Int'l Conf. on VLSI Design,* January 2001, pp. 211-214.

[64]H. Soeleman, K. Roy, and B. Paul, "Robust subthreshold logic for ultra-low power operation," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol 9, no. 1, February 2001, pp. 90 -99.

[65]J. Stankovic, "Distributed Computing" from *Readings in Distributed Computing Systems,* IEEE Computer Society Press, 1994.

[66]G. Succi and T. Pedersen, "Acoustic Target Tracking and Target Identification - recent result," *Unattended Ground Sensor Technologies and Application,* vol. 3081, 1997, pp. 139-145.

[67]G. Sunada, J. Jin, M. Berzins, and A. Wolfe, "COBRA: An 1.2 Million Transistor Expandable Column FFT Chip," in *IEEE Int'l Conf. on Computer Design,* October 1994, pp. 546-550.

[68]J. Tschanz, J. Kao, S. Narendra, R. Nair, D. Antoniadis, A. Chandrakasan, and V. De, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage, " *IEEE J. of Solid-State Circuits,* vol. 37, no. 11 , November 2002, pp. 1396 -1402.

[69]B. Vanhoof, M. Peon, G. Lafruit, J. Bormans, M. Engels, and I. Bolsens, "A scalable architecture for MPEG-4 embedded zero tree coding,"in *Proc. of the IEEE Custom Integrated Circuits,* May 1999, pp. 65-68.

[70]A. Vick, R. Moore, B. Pirnie, and J. Stillion, *Aerospace Operations Against Elusive Ground Targets,* Project Air Force RAND, Santa Monica, 2001.

[71]E. Vittoz, "Micropower Techniques", *Design of VLSI Circuits for Telecommunication and Signal Processing,* ed. J. Franca and Y. Tsividis, Chapter 5, Prentice Hall, 1994.

[72]Virtex Power Estimator Worksheet, http://www.xilinx.com/ise/power_tools/index.htm.

[73]E. Vittoz, "Large-Scale Integration in Watches", in *Solid-State Circuits*, 1976.

[74]A. Wang, "Skill Code for Layout and Place and Route tutorial," Massachusetts Institute of Technology, December 2003.

[75]A. Wang, "Synthesis, Place-and-Route, and Verification tutorial," Massachusetts Institute of Technology, December 2003.

[76]A. Wang and A. Chandrakasan, "Energy-Aware Architectures for a Real-Valued FFT Implementation," in *Proc. of Int'l Symp. on Low Power Electronics and Design,* August 2003, pp. 360-365.

[77]M. Wellman, N. Srour, and D. Hills, "Feature extraction and fusion of acoustic and seismic sensors for target identification," *Peace and Wartime Applications and Technical Issues for Unattended Ground Sensors*, ed. G. Yonas, vol. 3081, April 1997, pp. 139-145.

[78]S. White, "Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review", *IEEE ASSP Magazine*, vol. 6, no. 3, July 1989, pp. 4-19.

[79]M. Wosnitza, M. Cavadini, M. Thaler, and G. Troster., "A High Precision 1024-point FFT Processor for 2D Convolution," *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers,* vol. 41, pp. 118-119, 424.

[80]T. Xanthopoulos, "A Low-power DCT Core using Adaptive Bitwidth and Arithmetic Activity Exploiting Signal Correlations and quantization," *IEEE Journal of Solid-State Circuits,* vol. 35, no. 5, May 2000, pp. 740-750.

[81]K. Yao, R. Hudson, C. Reed, D. Chen, and F. Lorenzelli, "Blind Beamforming on a Randomly Distributed Sensor Array System," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, October 1998, pp. 1555-1567.