INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600**

UMI[®]

A Configurable 32nd Order Low Voltage Low Power Digital Filter for Portable Communications Systems

by

DUSAN SUVAKOVIC

**A thesis submitted in conformity with
the requirements for the degree of
Master of Applied Science**

**Department of Electrical and Computer Engineering
University of Toronto**

@ Copyright by Dusan Suvakovic 1998

**National Library
of Canada**

**Acquisitions and
Bibliographic Services**

**395 Wellington Street
Ottawa ON K1A 0N4
Canada**

**Bibliothèque nationale
du Canada**

**Acquisitions et
services bibliographiques**

**395, rue Wellington
Ottawa ON K1A 0N4
Canada**

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-45435-5

Canada

ABSTRACT

In this thesis, a micropower 32nd order digital filter for portable audio applications is designed and implemented in a single threshold 1V CMOS process. The filter features up to 16 biquad sections with programmable coefficients and performs fixed-point computations with 16-bit resolution. The average energy consumption measured on the filter prototype is 330pJ per biquad computation. This result indicates that the implementation of complex DSP functions is possible in a single threshold, low voltage CMOS process while achieving an energy efficiency typical for implementations in multiple threshold processes.

Several new architectural and circuit solutions are introduced as part of the design of this low-power, low voltage filter. The performance of different CMOS gate design styles in a low voltage process is analyzed. Some secondary effects contributing to energy consumption are identified. Emphasis is placed on energy estimation and design optimization. A simple but accurate energy estimation procedure is demonstrated.

ACKNOWLEDGEMENTS

I would like to thank Prof. Salama for his guidance and support throughout this work. I am especially grateful to him for giving me the opportunity to work on this exciting and inspiring project.

My sincere thanks to Jaro Pristupa for superb technical assistance.

Many thanks to my wife Ana for her encouragement, love and understanding.

Financial support provided by Gennum, Nortel, Micronet and ITRC is gratefully acknowledged.

TABLE OF CONTENTS

	Page
CHAPTER 1: INTRODUCTION	1
1.1 Motivation for Designing in Low Power Low Voltage (LPLV) Technologies	1
1.2 Digital Signal Processing in Portable Communication Systems	5
1.2.1 Digital Filters	6
1.3 Power Dissipation in DSP Circuits	9
1.3.1 Dynamic Switching Dissipation	10
1.3.2 Short Circuit Current Dissipation	13
1.3.3 Leakage Current Dissipation	14
1.4 Objective and Thesis Outline	15
CHAPTER 2: DESIGN DESCRIPTION	17
2.1 Design Specification	17
2.1.1 Performance Specification	17
2.1.2 Functional Specification	18
2.2 Design Approach and Constraints	20
2.3 Filtering Algorithm	21
2.4 Filter Architecture	23
2.4.1 Architecture of the Biquad Arithmetic Block	23
2.4.1.1 Selection of MAC Architecture	25
2.4.1.2 Details of MAC Architecture	30
2.4.1.3 Multiplier Architecture	32
2.4.2 Memory Architecture	33
2.4.2.1 Coefficient Memory	33
2.4.2.2 Data Memory	33
2.4.3 Control Logic Operation and Architecture	36
2.5 Circuit Design	39
2.5.1 Complex Cells: Ripple Carry Adder Circuit	40
2.5.2 Circuit Design Styles for Random Logic	44
2.5.3 Latch Circuit Modification	47
2.5.4 Impact of Glitching in Combinational Circuits	50
2.5.5 Memory Circuit Design	54
2.5.5.1 Estimate of Leakage Dissipation in Memory Blocks	55
2.5.5.2 Swing Recovery Buffer at Data Outputs	56
2.6 Layout Design	58
2.6.1 Impact of Layout Design on Energy Consumption	58
2.6.2 Energy Consumption of Global Signals	59
2.6.3 LPLV Filter Layout	62

	Page
2.6.3.1 MAC Layout	62
2.6.3.2 Chip Floorplan	64
2.7 Summary	68
CHAPTER 3: SIMULATION AND EXPERIMENTAL RESULTS	70
3.1 Simulation	70
3.1.1 Energy Estimation at the Circuit Level	70
3.1.2 High-Level Functional Simulation	73
3.1.3 High-Level Estimation of Energy Consumption	73
3.2 Experimental Results	75
3.2.1 Functional Testing of the Prototype	76
3.2.2 Measurement of Energy Consumption	77
3.6 Summary	80
CHAPTER 4: CONCLUSIONS	82
APPENDIX A: ARCHITECTURAL LEVEL ESTIMATE OF MAC ENERGY CONSUMPTION	84
APPENDIX B: A SEQUENTIAL MULTIPLIER WITH IMPROVED ENERGY-DELAY PRODUCT	90
APPENDIX C: PROGRAM "DS3": BEHAVIORAL DESCRIPTION OF THE IMPLEMENTED FILTER	95

LIST OF FIGURES

	Page
CHAPTER 1	
Figure 1.1 I_D vs. V_{GS} in subthreshold region	4
Figure 1.2 Cascade arrangement	8
Figure 1.3 Parallel arrangement	8
Figure 1.4 CMOS gate model	11
CHAPTER 2	
Figure 2.1 Start-stop synchronization	19
Figure 2.2 Block diagram of the implemented filter	22
Figure 2.3 Datapath and organization of the filter arithmetic block	24
Figure 2.4 Direct MAC implementation with add-shift multipliers	27
Figure 2.5 Modified MAC architecture	29
Figure 2.6 Detailed MAC architecture	31
Figure 2.7 Multiplier architecture	32
Figure 2.8 Coefficient memory location, size and datapath	34
Figure 2.9 Block diagram of a coefficient memory cell	34
Figure 2.10 Block diagram of a data memory cell	35
Figure 2.11 Flow chart of the chip control algorithm	37
Figure 2.12 Scheduling and synchronization between arithmetic blocks	38
Figure 2.13 24-transistor optimized CMOS full-adder cell	41
Figure 2.14 18-transistor optimized TG full-adder cell	42
Figure 2.15 Optimized TG FA circuit with inverted carry path	43
Figure 2.16 CPL AND/NAND gate with restore buffers	45
Figure 2.17 NMOS pass transistor implementation of MAC multiplexer	47
Figure 2.18 TG based latch circuit	48
Figure 2.19 Modified latch circuit	49
Figure 2.20 Positive edge triggered D flip flop operation	51
Figure 2.21 Register energy due to input glitching	52
Figure 2.22 Coefficient memory cell	54
Figure 2.23 Worst case leakage paths for coefficient memory cell	55
Figure 2.24 Read buffer with precharge and pull up transistors	57
Figure 2.25 Impact of cell aspect ratio on global signal length	61

		Page
Figure 2.26	MAC floorplan	63
Figure 2.27	Block diagram of the MAC bit slice	64
Figure 2.28	Full adder cell layout	65
Figure 2.29	Filter chip floorplan	67
 CHAPTER 3		
Figure 3.1	Testbench for energy/functional simulation	71
Figure 3.2	Simulated current consumption of the multiplier block	72
Figure 3.3	Photomicrograph of the core area	75
Figure 3.4	Testbench for functional verification	76
 CHAPTER 4		
 APPENDIX A		
 APPENDIX B		
Figure B.1	Architecture of the compressor based sequential multiplier	91
Figure B.2	Compressor architecture	94

LIST OF TABLES

	Page
CHAPTER 1	
Table 1.1 Summary of available Low Voltage Processes 	3
CHAPTER 2	
CHAPTER 3	
Table 3.1 Estimated breakdown of energy consumption 	74
Table 3.2 Typical energy measurements on the filter prototype 	78

LIST OF ABBREVIATIONS

LPLV	Low-Power Low-Voltage
PCS	Personal Communications Systems
IMEC	Interuniversity Microelectronics Center
DSP	Digital Signal Processing
ASP	Analog Signal Processing
SNR	Signal to Noise Ratio
MAC	Multiplier-Accumulator
CPA	Carry-Propagate Adder
RCA	Ripple-Carry Adder
CPL	Complementary Pass-transistor Logic
DPL	Dual Pass-transistor Logic
DPTL	Differential Pass-Transistor Logic
SRPL	Swing Restored Pass-transistor Logic

CHAPTER 1

INTRODUCTION

The rapid development of wireless communications in the past several years has enabled the development of various portable electronic systems that range from pagers and cellular phones to emerging personal communication systems, portable multi-media terminals and hearing aids. Consequently, the demand for the functionality and performance of such systems has been growing steadily. Typical requirements involved in digital portions of portable applications specify the signal processing throughput and/or computational performance. These requirements have to be met while maintaining low operating power to provide long battery life. Moreover, the constraints related to the cost, dimensions and weight of portable products usually dictate the need for the low operating voltage, preferably using a single battery cell. Since the battery technology has been improving rather slowly for the last couple of decades, improvements have been sought through innovations in integrated circuit technology as well as developments of new, more efficient circuit design techniques.

1.1 Motivation for Designing in Low Power Low Voltage (LPLV) Technologies

Various low power digital VLSI design techniques have been developed in the past several years. These range from low power processes and circuit design styles to optimizations at the architectural and algorithmic levels. The common goal in most of these approaches is to accom-

moderate reductions in the supply voltage. This is the most efficient way to reduce power consumption in digital CMOS circuits where total power is roughly proportional to the square of the supply voltage. On the other hand, CMOS circuits are slower when operated at lower supply voltages. In order for the system throughput to be kept constant, the performance degradation must be compensated for by using circuit or algorithmic techniques, most often at the expense of the circuit area.

Transistor channel lengths in digital designs are always chosen to be minimal which, in the case of modern submicron technologies, causes the devices to operate in velocity saturation. Transistor current under velocity saturation conditions is no longer a quadratic function of the voltage [1], but a linear one

$$I = WC_{ox} (V_{dd} - V_t) v_{max} \quad (1.1)$$

where W is the transistor width, C_{ox} is the transistor gate capacitance per unit gate area and v_{max} is maximum velocity. V_{dd} and V_t are the supply voltage and transistor threshold voltage respectively. Propagation delay t_d is then given as

$$t_d \equiv \frac{C_L V_{dd}}{2I_{ave}} = \frac{C_L V_{dd}}{2WC_{ox} (V_{dd} - V_t)} \quad (1.2)$$

where C_L is the load capacitance attached to the gate output node and I_{ave} is the average current flowing into or out of C_L during the observed gate transition.

Based on (1.2), if $V_{dd} \gg V_t$ it follows that circuit propagation delay is virtually independent of the supply voltage. However, as the supply voltage approaches V_t the delay increases rapidly. In conventional CMOS technologies, threshold voltages are in the range of 0.7V to 0.9V. In such technologies, values around 2V are assumed to be the practical lower bound to the supply voltage below which circuits become too slow and digital designs are no longer viable. Therefore, in order to allow further voltage reduction, new CMOS processes with reduced threshold voltages

are required.

Several digital CMOS processes with low transistor threshold voltages have been developed recently for target supply voltage of 1V [2]. Their characteristics are summarized in Table 1.1.

Table 1: Summary of Available Low Voltage CMOS Processes

	supply voltage (V)	threshold voltage V_t (V)	gate length L (μm)	gate oxide thickness t_{ox} ($\text{\AA}$)	subthreshold swing S (mV/dec)
AT&T	1	0.2	0.25	90	80
IMEC	1	0.3	0.5	70	70
STANFORD	1	0.1	0.25	49	80
MIT (SOI process)	1	0.25	0.25	50	70

Voltage scaling is beneficial from a reliability point of view since it reduces the power density and the electric field along the device channel thus allowing for the channel length to be reduced. Consequently, feature sizes of all the new LPLV technologies are $0.5\mu\text{m}$ or smaller. Smaller transistors generally have lower capacitance. Since power consumption of digital circuits is proportional to overall switched capacitance, this results in another positive impact of the low voltage technology on circuit power consumption.

A design implementation in a low voltage technology can reduce the complexity of a portable system. Such systems are operated from batteries whose voltage is sometimes lower than the nominal supply voltage of conventional CMOS circuits. The use of DC/DC converters is necessary in such cases to step-up the battery voltage. However, if the design is implemented in a low voltage technology, this hardware overhead is avoided.

The reduction of transistor threshold voltage increases subthreshold leakage current and

consequently the circuit static power consumption. Transistor current I_D is an exponential function of gate-source voltage V_{GS} in the subthreshold region. As shown in Figure 1.1, the $\log(I_D)$ vs. V_{GS} characteristic is linear in that region. Since a transistor in a digital CMOS circuit is turned off by applying zero voltage between its gate and source terminals, the value of interest for a circuit design point of view is the leakage current I_D for $V_{GS} = 0V$.

Figure 1.1. I_D vs. V_{GS} in subthreshold region

The slope of the characteristic in Figure 1.1 varies for different CMOS processes. It is desirable for a low voltage CMOS process to have the subthreshold slope as steep as possible in order to provide minimal value of I_D for $V_{GS} = 0$. CMOS processes are usually characterized by the reciprocal value of the subthreshold slope known as *subthreshold swing*, denoted as S and expressed in mV/decade.

Low voltage processes from Stanford University and MIT are not conventional CMOS processes because they involve substrate biasing to control threshold voltage in order to reduce leakage current of transistors in the *off* state. Hence these processes are more expensive than the AT&T or IMEC process. The IMEC and AT&T processes both have inherent low threshold voltage. The advantages of the AT&T process are lower threshold voltage (0.2V as compared to 0.3V in IMEC) and smaller feature size that provides higher packaging density. However the IMEC

process has smaller subthreshold swing, which along with a higher V_t gives much smaller leakage current than in AT&T process. For the purpose of this work, the IMEC process, which is commercially available for prototype implementation, was used.

1.2 Digital Signal Processing in Portable Communication Systems

Up until recently, signal processing in portable communication systems has predominantly been performed by analog circuits. The preference of analog signal processing (ASP) over digital signal processing (DSP) stems from its cost effectiveness and lower power consumption, both of these requirements being imperative for the personal communications systems (PCS) market. ASP circuits such as filters, implemented in one of the conventional semiconductor technologies, are far less complex than their digital counterparts involving fewer transistors and occupying a much smaller area. The latter has a direct impact on the yield and total production cost. Another advantage attached to the lower circuit complexity is power consumption. When operated from a 3.3-5V supply voltage, a digital filter consumes more energy than an analog one, implying shorter battery life. Finally, it is convenient to use analog circuits due to their ability to process signals over a wide frequency range as opposed to DSP where the processing is band limited by sampling in the A/D conversion.

Modern digital sub-micron CMOS technologies along with emerging low threshold processes tend to minimize the traditional disadvantages of DSP in portable applications. The high packaging density that can be achieved by implementing a DSP circuit in state-of-the-art processes with device feature size of $0.5\mu\text{m}$ or smaller making their chip area comparable to the area of their analog counterparts. Packaging density of CMOS analog circuits is not affected much by technology scaling. Device sizes are determined by current requirements and avoidance of the short channel effects. In addition capacitor sizes used in analog ICs are practically the same for most processes.

The reduction of power in digital circuits can most effectively be achieved by reducing the supply voltage. The novel low threshold technologies compensate for the loss of throughput associated with lowering of the supply voltage, which makes them attractive for implementation of low power DSP circuits. Again, analog circuits do not take advantage of technology scaling. The power in analog circuits is determined by the signal to noise ratio (SNR) requirement in such a way that the voltage reduction has to be accompanied by an increase in current and capacitor sizes, resulting in unchanged or higher power consumption [3].

With the traditional disadvantages suppressed or eliminated by technological improvements, the use of DSP in portable applications offers processing options that are not available in analog designs. The flexible nature of digital processing allows for easy implementation of complex operations through appropriate programming and re-use of the same hardware blocks, while in the analog implementation all parts of a complex processing function have to be implemented separately. Furthermore, the accuracy of analog circuits depends very much on technology requiring high precision in device matching and low component tolerances. This is particularly true for large designs where an error introduced in one block spreads throughout the circuit. Repeatability of high precision analog circuits is also a serious production problem. Precision in DSP circuits, by contrast, does not depend on technology parameters. Digital circuits are inherently precise.

1.2.1 Digital Filters

Digital filters are typical DSP circuits. They are easily programmable since their function is defined by a set of filter coefficients usually stored in an on-chip memory. This concept provides for easy realization of adaptive filtering by varying of the coefficients in real time. Similar features are virtually impossible to achieve with analog filters.

The transfer function of a higher order digital filter is generally given as a rational function

in z -domain:

$$H(z) = \frac{\sum_{n=0}^N a_n z^{-n}}{1 + \sum_{m=1}^M b_m z^{-m}} \quad (1.3)$$

where $a_0, a_1, \dots, a_N$ and $b_1, b_2, \dots, b_M$ are filter coefficients while z^{-i} denotes delay by i sampling periods.

The transfer function in (1.3) can be decomposed as a product, given by equation (1.4), or alternatively as a sum, given by equation (1.5), of second order functions [4] representing second order filters (biquad filters):

$$H(z) = \prod_{k=1}^K \frac{(1 + c_k z^{-1} + d_k z^{-2})}{(1 + e_k z^{-1} + f_k z^{-2})}, K = \max\left(\frac{N}{2}, \frac{M}{2}\right) \quad (1.4)$$

$$H(z) = \sum_{k=1}^K \frac{A_k (1 + B_k z^{-1})}{(1 + C_k z^{-1} + D_k z^{-2})}, K = \max\left(\frac{N}{2}, \frac{M}{2}\right) \quad (1.5)$$

where c_k, d_k, e_k and f_k are coefficients of the k -th second order filter in the product decomposition and A_k, B_k, C_k and D_k are coefficients of the k -th second order filter in the sum decomposition.

According to equations (1.4) and (1.5), a higher order filter can be implemented as a cascade of second order filters as shown in Figure 1.2a, or as a parallel-summing arrangement of second order filters as shown in Figure 1.3a, respectively. Block diagrams representing algorithms

performed by the second order filters in cascaded and parallel arrangement are shown in Figure 1.2b and 1.3b respectively. The transfer function of a higher order filter $H(z)$ in both cases can be implemented by the repetitive use of a single processing element (PE) able to perform the operation of a second order filter. Different data and coefficient inputs have to be provided to the PE at the beginning of each processing cycle. This approach minimizes the circuit area and it is suitable for DSP applications such as speech signal processing where sampling time is sufficiently long to allow sequential processing. Cascade arrangement is generally preferred to the parallel one since it allows the poles and zeros of the filter to be determined by coefficients of each section [5] and is less sensitive to coefficient-quantization errors [5].

Figure 1.2. Cascade arrangement

Figure 1.3. Parallel arrangement

Speech is the most commonly transferred and processed type of signal in portable communication systems. The upper bandwidth bound of communications systems transferring speech signal is usually limited to 4-6kHz because higher frequencies have little or no influence on human recognition of speech. When digital processing is performed, the sampling frequency of the input signal has to be at least twice the signal bandwidth (Nyquist frequency). In practice, the sampling frequency of the speech signal is unlikely to be higher than 15-20kHz. Therefore a sampling period of at least 50 μ s is available to the DSP circuit to complete the processing based on one sample. Given that typical propagation delays of arithmetic circuits implemented in modern CMOS technologies are in the range of tens of nanoseconds with typical clock frequencies in the 10-100MHz range, this type of digital signal processing is easily achievable without optimization for speed. However, it is power and chip area that need to be optimized in personal communication systems in order to achieve low cost and long battery life.

1.3 Power Dissipation in DSP Circuits

The power dissipation in CMOS DSP systems is attributed to the sources of power consumption in digital CMOS circuits. These are dynamic switching power consumed in charging of circuit capacitances, short circuit current power dissipated during gate switching and static power caused by leakage currents. These sources of power dissipation are discussed in the following sections.

Dissipation in digital circuits can be expressed either in units of power or in units of energy. *Power dissipation* is a critical parameter in very complex integrated circuits operating at high frequencies where it is related to packaging and reliability issues caused by self-heating of such circuits. However, the concept of *energy consumption* is more suitable for portable systems because they are powered from batteries with a limited energy budget. Particularly, in the case of digital signal processing systems where throughput and the amount of computation per sampling

period are known and constant, such systems and their building blocks can be characterized in terms of *energy per computation*. This metric is directly related to the battery life.

Another metric commonly used in portable DSP circuits is *energy · delay* product. It characterizes both energy consumption and performance, the latter being important in meeting DSP throughput requirements. This metric is equally suitable for characterization of large computational blocks (such as multipliers and adders) and basic digital circuits.

The implications of *energy per computation* and *energy · delay* product metrics are illustrated in the following discussion. If two differently designed PE blocks (such as biquad filters) perform the same computational function, then the one characterized by lower energy per computation is more desirable from the standpoint of battery life. Further, if two PEs consume the same *energy per computation*, the one with shorter computational delay (and smaller *energy · delay* product) is preferred. The faster PE can be used more times in a given timeframe thus allowing for a higher throughput. To achieve the same throughput using the slower PE, several instances of the same PE have to be used in parallel, compensating for the insufficient speed and occupying an additional chip area. Finally, if one of the two available PEs has lower *energy per computation* and the other one smaller *energy · delay* product, then the choice between them is made based on the appropriate cost function for the particular design. Generally, a trade-off between circuit area and battery life is involved in this choice.

1.3.1 Dynamic Switching Dissipation

The generalized model of a CMOS gate consisting of NMOS and PMOS transistor networks, driving load capacitance C_L is shown in Figure 1.4. When a low-to-high transition occurs at the gate output, C_L is charged from 0 to V_{dd} . As shown in [1], the energy consumed during the charging of the load capacitance is equal to $C_L \cdot V_{dd}^2$. This energy is taken from the power supply.

The subsequent transition from high to low will discharge C_L , but no energy will be drawn from the power supply. For a gate active during computation γ (where γ can be an addition, multiplication, filtering operation etc.), the average energy dissipated in capacitance switching at the gate's output node during computation γ , can be expressed as

$$\bar{E}_{switching} = \bar{N}_{0 \rightarrow 1} \cdot C_L \cdot V_{dd}^2 \quad (1.6)$$

where $\bar{N}_{0 \rightarrow 1}$ is the average number of transitions from zero to V_{dd} in the observed node during computation γ . Dynamic switching frequency is the dominant source of power consumption in CMOS circuits and most low power techniques applied at different design levels are focused on minimizing either the effective switched capacitance in the design or the supply voltage. The effective switched capacitance is the sum of all node load capacitances throughout the circuit C_{Li} multiplied by their associated average switching activities $(\bar{N}_{0 \rightarrow 1})_i$

$$C_{eff} = \sum_i (\bar{N}_{0 \rightarrow 1})_i \cdot C_{Li} \quad (1.7)$$

Figure 1.4. CMOS gate model

Capacitance C_{Li} associated with the i -th logic gate is an equivalent lumped value that takes into account both the physical capacitance attached to the gate's output node and all internal node capacitances [6]. This simplified model allows easier gate-level energy analysis.

From equation (1.6), the most effective way to reduce the dynamic dissipation is the reduction of the supply voltage due to quadratic dependence of $E_{switching}$ on V_{dd} . If V_{dd} is scaled down, circuit delay will increase introducing an undesirable loss of throughput. Two techniques are available to compensate for the increased delay. The first one is architectural and involves hardware duplication or pipelining to allow for longer critical path [1]. The second technique introduces changes in the CMOS device technology by reducing transistor threshold voltage V_t . As CMOS circuit delay is inversely proportional to $(V_{dd} - V_t)$ the reduction of threshold voltage will result in smaller circuit delay, thus compensating for the smaller V_{dd} .

There are numerous low power techniques aimed at reduction of the effective switched capacitance. The general power dissipation model for DSP VLSI chips presented by Powell and Chau [7] gives a guideline for the high-level power analysis and low power design in the algorithmic and architectural design stage. The total DSP power can be expressed as the sum of three power components:

$$P_{total} = P_{computational} + P_{memory} + P_{I/O} \quad (1.8)$$

It is shown in [7] that the power consumed in memory access is reduced if memory is implemented in a distributed fashion, local to each computational block. This organization reduces memory access rate and the total physical capacitance of the memory data bus in every access. In order to accommodate the suggested memory arrangement, the circuit performing DSP computations should be implemented in several separate processing elements (PE). It is also shown in [7] that such partition of the computational block does not change the total computational power to a first order approximation. Further, power consumed to perform multiplication dominates the power consumption of PEs, so an effort should be made to design power efficient

multipliers. The above conclusions are also valid if DSP dissipation is discussed in terms of energy rather than in terms of power.

Another important component of energy consumption is dissipation in busses and wires connecting major building blocks. These wires are significantly longer than interconnects inside blocks and thus have large parasitic capacitance. It is a function of architectural design to keep these interconnects as short as possible and to make them switch at the slowest possible rate.

Energy efficient design at the gate level involves the analysis and reduction of unnecessary transitions of logic gates caused by different lengths of propagation paths for different input signals. The unnecessary transitions (dynamic hazards) are present in static CMOS circuits and they effectively increase gates' switching activity [1].

Load capacitance C_L in Figure 1.4 consists of three types of capacitances attached to the observed node. These are transistor gate capacitances of the next stage, diffusion capacitances of NMOS and PMOS transistor networks and interconnect capacitances. Minimization of the total load capacitance is a concern of circuit and layout level design. Gate and diffusion capacitance can be reduced by scaling down the device sizes. Interconnect capacitance depends on the wire length and width as well as on the metal layer used. It is generally desirable to make interconnects as short as possible.

1.3.2 Short Circuit Current Dissipation

The current flowing from the power supply to ground during transitions in CMOS gates , when both its NMOS and PMOS blocks are conducting, is known as short circuit current [1]. Short circuit currents are caused by the finite transition time of the input signals applied to the CMOS gate inputs and can be especially large if the output transition time is shorter than the input transition time. Energy consumption associated with short circuit currents is dynamic in nature . It

can be minimized by making input and output rise and fall times of all CMOS gates in a design approximately the same. In most CMOS designs, the short circuit current dissipation does not exceed 10% of the total dynamic dissipation.

1.3.3 Leakage Current Dissipation

Dissipation caused by leakage currents is static and negligible in high threshold CMOS processes. There are two types of leakage current: the leakage current of the reverse biased drain junction and the MOSFET subthreshold leakage. The leakage current for the diode is given by:

$$I_{diode} = I_S \left(e^{\frac{V}{V_T}} - 1 \right) \cong I_S \quad (1.9)$$

where I_S and V_T are diode saturation current and thermal voltage respectively. The subthreshold leakage can be expressed as:

$$I_{ds} = K e^{\frac{(V_{gs} - V_t)}{nV_T}} \left(1 - e^{\frac{-V_{ds}}{V_T}} \right) \quad (1.10)$$

where K and n are constants.

When a MOSFET is turned off, assuming the rail to rail swing in digital CMOS circuits, its gate to source voltage is zero and the term $(V_{gs} - V_t)$ in equation (1.10) becomes $-V_t$. Since circuits implemented a low threshold technology have increased subthreshold leakage current while their overall dissipation tends to be low, the impact of subthreshold leakage power on the total dissipation cannot be neglected.

1.4 Objective and Thesis Outline

The objective of this thesis is to design a digital filter for portable applications, suitable for implementation in a low threshold, 1V CMOS technology. The process selected for the implementation of the design is the IMEC pl5000 process. This process is expected to be interesting for industrial application because it is a low cost 1V process with excellent subthreshold characteristics. Experimental results obtained from measurements performed on the fabricated chip are expected to give an insight into energy and performance characteristics of this type of process when utilized in digital signal processing applications.

A detailed description of the biquad filter design encompassing the filtering algorithm, architecture, circuit level design and floorplan are given in Chapter 2 [8]. Also the design technique applied in order to reduce energy dissipation is discussed in Chapter 2 [9].

The estimates of the total energy per computation, made based on known design parameters and specifications, are presented in Chapter 3. Also, experimental results are described in Chapter 3.

Chapter 4 contains conclusions, design recommendations and suggestions for future work.

REFERENCES:

- [1] A.P. Chandrakasan & R.W. Brodersen, editors, "Low Power Digital CMOS Design", Kluwer Academic Publishers, Boston 1995
- [2] C. A. T. Salama, "LVLP Limitations, Impact on Device and Process Design", Micronet Course on Low Voltage Low Power Integrated Circuit Design, Toronto, June 1996
- [3] M. Snelgrove, "LVLP Analog Design", Micronet Course on Low Voltage Low Power Integrated Circuit Design, Toronto, June 1996
- [4] S. M. Bozic, "Digital and Kalman Filtering", London: Edward Arnold Publishers, 1979
- [5] R. A. Haddad, T. W. Parsons, "Digital Signal Processing Theory, Applications and Hardware", Computer Science Press, New York, 1991
- [6] J.M. Rabaey, M. Pedram, editors, "Low Power Design Methodologies", Boston: Kluwer Academic Publishers, 1996
- [7] S. R. Powel, P. M. Chau, "A Model for Estimating Power Dissipation in a Class of DSP VLSI Chips", IEEE Transactions on Circuits and Systems, vol. 38, pp. 646-650, June 1991
- [8] D. Suvakovic, C. A. T. Salama, "A Configurable 32nd Order Low Voltage Low Power Digital Filter for Portable Communications Systems", accepted for ISCAS'98, Monterey, California, May-June, 1998
- [9] D. Suvakovic, C. A. T. Salama, "Guidelines for Use of Registers and Multiplexers in Low Power Low Voltage DSP Systems", accepted for Great Lakes Symposium on VLSI '98, Lafayette Louisiana, February 1998

CHAPTER 2

DESIGN DESCRIPTION

In this chapter the design of a low power low voltage digital filter for portable communications systems is described. In the first section, the assumed design specification is given. The remaining sections contain the description of the algorithmic, architectural, circuit and layout design stages, with an emphasis on the applied low power technique.

2.1 Design Specification

2.1.1 Performance Specification

The primary purpose of the digital filter designed for implementation in a LPLV process is processing of audio signals in battery operated systems. It was assumed that characteristics of the filter have to meet typical requirements in such systems in terms of throughput and dynamic range.

Personal communications systems can be classified in two groups with respect to the type of the processed audio signal. PCS in the first class are aimed at processing of voice signal. Cellular phones are a typical representative of such systems where bandwidth and dynamic range are constrained to values sufficient for voice recognition. Systems in the other group, such as hearing aids, are meant for processing of general audio signal with a higher bandwidth and dynamic range. Filter specifications used in this design are based on those found in a typical analog hearing aid product [1].

In order to provide the dynamic range of 80dB specified in [1], all arithmetic operations have to be implemented with a minimal resolution of 14 bits. The ratio between the smallest and the largest value that can be expressed with 14 bits is $\frac{2^{14} - 1}{1} \approx 2^{14}$. Expressed in decibels, this dynamic range is equal to $20 \cdot \log(2^{14}) = 84.3dB$. An audio signal bandwidth of 10kHz [1] was assumed, implying the minimum sampling frequency of 20kHz (Nyquist frequency).

The filter order was chosen to be 32. This choice was not based on requirements in any particular portable system. However, this does not diminish the generality of the performance and energy measurement results since the overall performance of the proposed filter architecture is defined by the speed of a single biquad section circuit, while the total energy consumption must be proportional to the number of such sections being active during one sampling period.

The design goal was to achieve as low energy consumption per biquad section as possible, in order to characterize the energy efficiency of a LPLV process. Hence no target value was set for the total energy consumption.

2.1.2 Functional Specification

Programmability of filter coefficients was adopted as a functional requirement in order to accommodate applications requiring programmable coefficients such as hearing aids. The implementation of this feature also provides a testbench for measurements of quiescent leakage current in volatile memory needed for coefficient storage. A serial interface has been provided for external read and write access to the coefficient memory (shown in Figure 2.2b).

The operation of the designed filter is synchronized with the input sampling frequency through the use of start-stop control logic. The amount of computation performed by the filter in each sampling period is fixed and consists of a constant number of clock cycles N_{clk} . In order for the filter to complete its computation within one sampling period, the frequency of its operating

clock f_{clk} has to meet the following condition:

$$f_{clk} \geq N_{clk} \cdot f_{sample} \quad (2.1)$$

where f_{sample} is the input sampling frequency. The filtering operation is initiated in every sampling period by a rising edge of the chip's 'start' input. The 'start' signal tells the filter that a valid data sample is available at its input and it is issued by an external device responsible for delivering data samples to the filter. Upon completion of the filtering operation, the chip activates its 'stop' output and enters the idle mode in which no computation is performed. The 'stop' signal also indicates that a valid result is present at the chip output. The 'stop' signal is deactivated immediately after the subsequent positive transition of the 'start' input is detected. Figure 2.1 shows the described start-stop synchronization.

Figure 2.1. Start-stop synchronization

2.2 Design Approach and Constraints

Based on the given requirements, the design of the specified digital filter was conducted as a design for low power constrained by the speed requirement of performing 16 biquad filter computations in a sampling time interval of 50 μ s.

An assumption was made that silicon cost in a portable audio application should be kept as low as possible. To minimize the area of the arithmetic circuitry, a sequential filtering algorithm was implemented and word-serial multiplication was utilized.

Once the filtering algorithm was defined as a cascade of biquad sections, possibilities of energy savings were considered at the architectural, circuit, and layout design levels.

At the architectural level, an attempt was made to reduce switching activity of datapath blocks involved in biquad computation. Also, consideration was given to architectural suitability of compact and regular layout in order to minimize the overhead in dissipation caused by interconnect capacitance.

At the circuit level, circuits with minimum input and internal capacitance were sought for implementation of datapath blocks with highest switching activity.

At the layout level, an effort was made to minimize interconnect and diffusion capacitance by making the layout as compact as possible. The layout was organized such that nets with higher switching activity had the shortest interconnect, while the longest wires had relatively low switching activity.

2.3 Filtering Algorithm

The filter performs the function of 16 cascaded biquad sections $H_1(z) \dots H_{16}(z)$ shown in Figure 2.2a. The actual structure of the implemented circuit is shown in Figure 2.2b. It includes a single arithmetic block performing the operation of a biquad filter as well as the coefficient and data memory for 16 independent biquad filters. The arithmetic block is repetitively used 16 times with different coefficient/data sets. The multiplexer shown in Figure 2.2b selects between the input signal and the biquad output. The input signal is applied to the biquad filter input in the first biquad operation cycle, whereas the biquad output is fed back to its input in the 15 subsequent cycles. All 16 biquad cycles have to be completed in one sampling period and the final result is available at the biquad output upon completion of the 16th biquad cycle.

The diagram specifying the algorithm performed by the biquad arithmetic block is shown in Figure 2.2c. The algorithm includes four addition and five multiplication operations. Blocks denoted as z^{-1} in Figure 2.2c represent data delayed by one sampling period for the observed biquad section. The delayed data is stored and retrieved from the data memory shown in Figure 2.2b. c_{k1} , c_{k2} , d_{k1} , d_{k2} and g_k are filter coefficients for the k -th biquad section ($k=1,2..16$) and they are stored in the coefficient memory. The control block shown in Fig. 2.2b is responsible for the selection of a proper coefficient and data set for each biquad section.

The multiplier at the biquad output determines the gain of each particular biquad section. The use of this multiplier is not necessary for cascade decomposition since there is no gain coefficient in equation (1.4), but it introduces an additional degree of freedom for selection of numerical values for other filter coefficients as part of an energy management scheme. An actual selection of coefficient values can reduce or increase energy per computation depending on the number of zeros and ones in its binary representation [4]. The proposed filter structure allows that all c_k and d_k coefficients can be scaled by a certain value, selected such that total number of ones in their digital representation is reduced. This scaling would then be compensated for by multiply-

Figure 2.2. Block diagram of the implemented filter:
a) function - 16 biquad sections in cascade
b) implementation - top level block diagram
c) biquad filter algorithm

ing of the biquad output Z_k with the gain coefficient. The structure shown in Fig 2.2c was implemented in order to determine if the introduction of the gain coefficient can reduce the overall switching activity and energy consumption. The implemented filter performs fixed point arithmetics with 16-bit precision of all coefficients. The outputs of all multipliers and adders as well as the filter input and output are also constrained to 16 bits.

2.4 Filter Architecture

As shown in Figure 2.2b, the major building blocks in the filter design are:

- biquad arithmetic block,
- coefficient memory,
- data memory,
- control logic block,
- input data multiplexer and
- circuitry for external access to coefficient memory.

The architecture of each one of these building blocks will be described in one of the following sections.

2.4.1 Architecture of the Biquad Arithmetic Block

The operation performed by the biquad arithmetic block is shown in Figure 2.2c. This block consists of five multipliers and four adders. The two delay blocks shown in the same figure belong to data memory. Excluding the gain multiplier at the biquad output, the remaining part of this diagram is symmetrical in terms of arithmetic operators as well as data flow directions (symmetrical with respect to the vertical axis crossing delay blocks). Using this property, the biquad arithmetic block has been organized as shown in Figure 2.3.

The biquad architecture consists of two identical multiplier-accumulator blocks (MAC) and a multiplier. Each MAC block performs two multiplications and two additions. Signal X_k is calculated by MAC1 in the feedback path as

$$X_k = c_{k1} \cdot X_{k-1} + c_{k2} \cdot X_{k-2} + u_k \quad (2.2)$$

Figure 2.3. Datapath and organization of the filter arithmetic block

The input to MAC2 in the direct path is X_k (calculated by MAC1). MAC2 output is calculated as

$$Z_k = d_{k1} \cdot X_{k-1} + d_{k2} \cdot X_{k-2} + X_k \quad (2.3)$$

The separate multiplier in Figure 2.3 calculates the output Y_k by multiplying Z_k with the gain coefficient g_k .

Most of the computational activity and capacitance switching is performed inside the MAC blocks. Therefore, the MAC design is the most important for the total energy consumption of this digital filter. It is for this reason that in the following section different architectural alternatives are discussed instead of giving a plain description of the implemented architecture.

2.4.1.1 Selection of MAC Architecture

Multiplier is the critical component of the MAC architecture. Compared to adders and other circuits typically found in DSP datapaths (registers, multiplexers etc.) multipliers are by far the most energy, time and area consuming datapath circuits [5, 6].

There are different multiplier architectures, ranging from parallel to bit serial and from combinatorial to sequential and/or pipelined architectures. The fastest are parallel multipliers. They can be designed as array or Wallace tree [5], [7], [9]. Booth's recoding can also be used in combination with one of these two methods to reduce the number of partial products [7], [9].

Although parallel architectures provide the highest speed of multiplication, there are some disadvantages to their implementation in personal communications systems. Firstly, they occupy a large silicon area making chips very expensive. This also means that if a parallel multiplier is implemented on a PCS DSP chip, due to its large size it will most likely be the only multiplier on that chip. The use of a single multiplier affects both the chip performance and its architecture. The effective performance of a single multiplier DSP chip is inversely proportional to the number of multiplications required in the implemented algorithm. Multiplications have to be done one by one, thus reducing the advantage of high speed parallel multiplication. The architecture of a single multiplier chip necessarily involves the use of long, heavily loaded busses in the datapath to and from the multiplier. As stated in [10], this approach significantly increases energy consumption of the chip because it violates the recommended principles of locality and regularity in digital filter implementation.

Secondly, it is not possible to make a general judgement on whether the energy consumption of a parallel multiplier is low or high. It rather depends on design details of a particular implementation. As mentioned in Chapter 1, energy consumption of a digital CMOS circuit is generally proportional to the total circuit capacitance and its average switching activity. The former affects the average energy of Wallace tree multiplier [9] due to its large load capacitances introduced by

irregular layout and long interconnects. The latter affects array multipliers which tend to have larger number of unnecessary transitions per computation than Wallace tree ones [5]. This difference in switching activity increases with the number of multiplier input bits [5].

The disadvantage of a Wallace's tree structure can be contained by reducing layout interconnect lengths, while switching activity of an array multiplier can possibly be reduced by circuit and layout level design efforts to match certain circuit delays and cause some internal transitions to cancel out. However, there is no clear way to determine which one of these architectures consumes less energy in the 16-bit by 16-bit multiplication. It is also impractical to use simulation to determine the energy consumption of a parallel multiplier. Low level simulators such as SPICE are the most accurate although very slow and do not allow for collection of a sufficient data to calculate the average computational energy of a multiplier. SPICE simulation is additionally aggravated by the circuit size of a parallel multiplier. On the other hand, the use of switch level simulators such as IRSIM for energy estimation is imprecise since they do not take into account the phenomena such as short circuit current and parasitic capacitance [8].

Finally, the speed of parallel multipliers cannot be fully taken advantage of in DSP systems processing audio signal due to low sampling rates. Throughput requirements in such systems can be met by slower, sequential multipliers.

For all the reasons described above, a decision was made to design a biquad circuit using a sequential multiplier. The simple add-shift multiplication technique has been applied in the gain multiplier. A modification of this technique has been made in the design of the MAC architecture. The nature of this modification is shown in figures 2.4 and 2.5. Figure 2.4 shows a direct MAC implementation calculating X_k according to equation (2.2), implemented with two add-shift multipliers and two carry-propagate adders (CPA).

Figure 2.4 Direct MAC implementation with add-shift multipliers

Signals $X_{k-1(i)}$ and $X_{k-2(i)}$ represent the i -th bit of the 16-bit long delayed data words X_{k-1} and X_{k-2} respectively. A multiplication is performed in 16 clock cycles. Each bit of the 16-bit filter coefficient c_{k1} is AND-gated by $X_{k-1(i)}$ in the i -th clock cycle of the multiplication and each bit of the 16-bit coefficient c_{k2} is AND-gated by $X_{k-2(i)}$ in the i -th clock cycle. The structure of the computation performed by the architecture shown in Figure 2.4, including all involved terms is given in equation (2.4).

$$X_k = \sum_{i=0}^{15} 2^i \cdot X_{k-1(i)} \cdot c_{k1} + \sum_{i=0}^{15} 2^i \cdot X_{k-2(i)} \cdot c_{k2} + u_k \quad (2.4)$$

Carry-propagate adders in multiplier blocks MUL1 and MUL2 are both 16 bit long. Since data input u_k is represented with 16-bit precision, the multiplication results can be truncated to 16 bits. Therefore CPA1 and CPA2 are also 16 bit adders.

The purpose of the latches connected to multiplier outputs in Figure 2.4 is to prevent unnecessary activity of CPA1 and CPA2 circuits caused by intermediate multiplier results being switched in 16 clock cycles. The latches are updated once the final results are established at multiplier outputs.

Energy in this MAC architecture is mostly consumed by CPAs and multiplier registers. The effect of AND gates on the total energy consumption can be neglected in the first order analysis because a 2-input AND gate is a much smaller circuit than a bit slice of a CPA_{mul}. Therefore its energy is negligible compared to that of a CPA_{mul} for the same input switching rate. The energy consumption of the latches is also negligible because they are activated only once during a biquad computation, while a latch energy per transition is approximately 1/2 of that of a register (since one register consists of two cascaded latches).

In order for X_k to be calculated by the MAC shown in Figure 2.4, each CPA_{mul} has to perform 16 additions and CPA1 and CPA2 have to do one addition each. Hence, the MAC will perform 34 16-bit additions in every biquad operation. Each one of the two 16-bit registers is written in 16 times during the multiplication execution making the total of 32 register updates.

The MAC architecture shown in Figure 2.5 also performs the operation given by equation (2.2) but its total activity is significantly lower than that of the one shown in Figure 2.4.

This architecture is obtained by rearranging of equation (2.4) into equation (2.5).

$$X_k = \left(\sum_{i=0}^{15} 2^i \cdot (X_{k-1(i)} \cdot c_{k1} + X_{k-2(i)} \cdot c_{k2}) \right) + u_k \quad (2.5)$$

Figure 2.5. Modified MAC architecture

The expression $(X_{k-1(i)} \cdot c_{k1} + X_{k-2(i)} \cdot c_{k2})$ in the last equation can assume one of four possible values: 0, c_{k1} , c_{k2} or $c_{k1} + c_{k2}$ corresponding to the combinations (0, 0), (0, 1), (1, 0) and (1, 1) of the two bits $(X_{k-2(i)}, X_{k-1(i)})$ respectively. Therefore $X_{k-2(i)}$ and $X_{k-1(i)}$ can be used as control signals to a 4:1 multiplexer to select one of the four possible adder inputs in each clock cycle.

This way the sum of two products can be calculated by a single add-shift multiplier circuit operated for the same number of clock cycles as if a single multiplication was taking place. However, it is necessary to calculate the sum of coefficients $c_{k1} + c_{k2}$ before the multiplier is started or alternatively, provide additional memory for storage of this sum. The former solution, being more area efficient, was chosen for implementation.

An architectural level analysis of the switching activity and energy consumption was performed for the original and modified MAC architecture. The analysis was supported by SPICE simulation results obtained for particular full-adder, register and multiplexer circuits that were subsequently used in the filter implementation. It showed that the modified MAC architecture consumes approximately 40% less energy than the original one. Consequently, MAC blocks in the filter design were implemented according to Figure 2.5. The energy estimation procedure is explained in detail in Appendix A.

2.4.1.2 Details of MAC Architecture

Figure 2.6 shows a more detailed diagram of the selected MAC architecture. LATCH1 and LATCH2 effectively play the role of the register in Figure 2.5. However, LATCH1 is also driving the output latch LATCH3. LATCH3 is updated only once in a biquad cycle, upon completion of the computation. LATCH3 is necessary to prevent the internal switching activity caused by a sequence of intermediate multiplication results, from spreading outside the MAC block.

The bit length of CPA_{mul} has been extended to 18 bits to provide necessary guard bits for handling of 2's complement computations.

The multiplexer following LATCH1 in Figure 2.6 is supposed to control shifting of intermediate outputs from CPA_{mul} . It has not been implemented in the submitted design. Instead, wired shifting, as shown in Figure 2.5 has been used. Wired shifting has a deficiency of limiting the allowed range of the input signal to a 14-bit value for some typical sets of digital filter coefficients. The transfer function of a typical low-pass or high-pass biquad section has the form

$$H(z) = \frac{(1 + (\pm 2z^{-1}) + z^{-2})}{(1 + e_k z^{-1} + f_k z^{-2})} \quad (2.6)$$

which implies that the term $(\pm 2z^{-1}) + z^{-2}$ in the numerator (corresponding to the MAC opera-

tion in the direct path) has larger weight than the input of the biquad section. When wired shifting is used, the sum of two multiplications is actually divided by two before the input value is added. In order to perform proper computation, the input value has to be scaled down appropriately, thus reducing its resolution to 14 bits. This resolution still meets the specifications given in section 2.1. However, the introduction of the multiplexer shown in Figure 2.6 would provide the full 16-bit resolution with a minor increase of energy and area. The multiplexer would provide right-shift operation for the duration of multiplication computation, whereas left-shift would be performed only once, upon the completion of multiplication, prior to the addition of the external input.

Figure 2.6. Detailed MAC architecture

2.4.1.3 Multiplier Architecture

The gain multiplier of the biquad filter has been implemented as a sequential add-shift multiplier, whose block diagram is shown in Figure 2.7. The same type of CPA is used as in the MAC block.

Figure 2.7 Multiplier architecture

The clock frequency of an add-shift multiplier based on a ripple-carry adder is constrained by the longest propagation delay of the adder. The performance of such sequential multiplier can be improved by using a faster adder or some other multiplication scheme. An alternative sequential multiplier design, which is intrinsically faster while having relatively low energy consumption is proposed in Appendix B. Instead of utilizing the add-shift algorithm, this multiplier is based on adding of partial products through a compression scheme. The proposed architecture is expected to have somewhat higher energy per multiplication, but lower *energy · delay* product than the implemented add-shift multiplier.

2.4.2 Memory Architecture

This digital filter design requires memory for storage of filter coefficients as well as for storage of delayed data. Both coefficient and data memory have been implemented as register files.

2.4.2.1 Coefficient Memory

The implemented coefficient memory consists of three register file blocks as shown in Figure 2.8. Two of them, CM1 and CM2, store coefficients used by MAC blocks and are connected to one MAC each. The third coefficient memory block, CM3 is connected to the gain multiplier and stores biquad gain coefficients. CM1 and CM2 store 2 biquad coefficients for 16 biquad filters each. Therefore the CM1 and CM2 size is 16 cells x 32 bits. CM3 stores only one coefficient per biquad for 16 biquad filters. Hence, its size is 16 cells x 16 bits.

All three coefficient memory blocks have dual-read single-write register file architecture to accommodate external read/write access during filter configuration time, as well as internal read access during filter operation time. A block diagram of a coefficient memory cell with all control signals and data busses is shown in Figure 2.9.

2.4.2.2 Data Memory

Data memory shown in Figure 2.3 consists of two separate blocks for storage of delayed data X_{k-1} and X_{k-2} . X_{k-2} block has one write and one read port, whereas block X_{k-1} has one write and two read ports. The first read port of X_{k-1} is used to provide data to the arithmetic circuitry, while the second read port is connected to the write port of X_{k-2} block. The size of each one of these blocks is 16 cells by 16 bits.

Figure 2.8 Coefficient memory location, size and datapath

Figure 2.9 Block diagram of a coefficient memory cell

In the actual implementation, X_{k-1} and X_{k-2} memories are interleaved by grouping together two memory cells, one belonging to X_{k-1} and the other to X_{k-2} , whose ports are connected to each other. This way, a point to point connection between X_{k-1} and X_{k-2} is provided and the use of a bus is avoided. The block diagram of a dual-memory cell with its control signals and busses is shown in Figure 2.10.

Figure 2.10. Block diagram of a data memory cell

When data is transferred from a X_{k-1} register to a corresponding X_{k-2} register, the switched interconnect capacitance is that of a short wire. Had this transfer been done over a data bus to which 16 X_{k-1} cell outputs and 16 X_{k-2} cell inputs would have been connected (one cell per row), the switched capacitance would have been larger. Consequently, the implemented data memory architecture reduces energy consumption.

2.4.3 Control Logic Operation and Architecture

The control logic block synchronizes the operation of all components of the filter datapath. It operates in a double loop executed in each sampling period. The flow chart of the chip control algorithm is shown in Figure 2.11.

The filter in stand-by mode waits for the detection of START signal. When this signal is detected, the STOP output is cleared and the execution of the outer loop begins. The outer loop counts biquad operations to be executed in a sequence. This number, denoted as n in Figure 2.11, is configurable externally and has to be set once before the filtering operation takes place. n being set, signals selecting the n -th location in the filter's data and coefficient memory are issued. After that, the inner loop execution starts. Biquad operation for the selected n -th section is performed in the inner loop. Once the inner loop execution is finished, n is decremented selecting the coefficients and data for the subsequent biquad section. Upon completion of the last biquad operation, for which $n=0$, the chip issues STOP signal and enters stand-by mode waiting for the detection of the next START signal.

The inner loop is responsible for sequencing of the events inside the arithmetic block consisting of two MACs and one multiplier. The complete biquad arithmetics is performed in 21 clock cycle. The control block issues MAC and multiplier control signals in each clock cycle. Scheduling and synchronization between arithmetic blocks is illustrated in Figure 2.12.

Scheduling of the filter arithmetics is done such that the multiplier operation trails the operation of the two MACs by one biquad processing interval. The multiplication $g_n \cdot Z_n$ takes place in the biquad processing interval, in which MAC1 and MAC2 are processing the $(n-1)$ -th biquad section. The multiplication result is made available at MAC1 external input at the beginning of the 17-th clock cycle in which MAC1 needs to process it. This way, the MAC blocks are not kept idle while the multiplier is processing the MAC2 output Z_n and the number of clock cycles in one biquad processing interval is limited to that required by MACs.

Figure 2.11. Flow chart of the chip control algorithm

The synchronization between MAC1 and MAC2 is accomplished by introducing idle clock cycles in their operation. Data and coefficient multiplication/addition is performed in both MAC1 and MAC2 in 16 clock cycles. Addition of the external signal is done in the 17-th clock cycle in MAC1, while MAC2 is executing no operation. The valid MAC1 output result X_k appears at its double buffered output at the end of the 18-th clock cycle. X_k is used by MAC2 as the external MAC input for addition in the 19-th clock cycle. The valid MAC2 output result Z_k appears at its double buffered output at the end of the 20-th clock cycle.

multiplier	MAC1	MAC2	clock cycle
read g_{k+1} ;	read c_{k1}, c_{k2} ; calculate $ck1 + ck2$;	read d_{k1}, d_{k2} ; calculate $dk1 + dk2$;	0
calculate:	calculate:	calculate:	1
$Y_{k+1} = g_{k+1}Z_{k+1}$	$X_k = c_{k1}X_k^{-1} + c_{k2}X_k^{-2}$	$Z_k = d_{k1}X_k^{-1} + d_{k2}X_k^{-2}$	2
Y_{k+1} available			16
	$X_k = X_k + Y_{k+1}$		17
	X_k available		18
		$Z_k = Z_k + X_k$	19
		Z_k available	20

Figure 2.12. Scheduling and synchronization between arithmetic blocks

One hot encoding is used to implement counters for both the inner and the outer loop. This encoding type was selected because of its low switching activity and because it eliminates the need for decoder circuits.

2.5 Circuit Design

The low power technique applied at the architectural design level consists mostly of the analysis of average switching activity for different architectures performing the same algorithm, in order to select the one with smallest switching activity. At the circuit level however, the objective of low power design is primarily the reduction of physical capacitance in order to reduce the average switching energy given by equation (1.6).

As explained in [11], the total physical capacitance seen by a CMOS gate output is the sum of the load gate capacitance C_g , the interconnect capacitance C_I and the diffusion capacitance C_j . Gate capacitance is directly proportional to the device channel length L and width W . Since the channel length of all MOS transistors in digital circuits is chosen to be minimal for a given technology, C_g can be reduced by reducing the device channel width.

Diffusion capacitance C_g is a sum of two components one of which is proportional to the area and the other to the perimeter of the device diffusion area. Consequently both components are proportional to the device width. Hence, the sum $(C_g + C_j)$ directly depends on transistor sizing.

Interconnect capacitance is a function of the chip architecture and layout, but it is also related to transistor sizing. Circuits with smaller transistors occupy a smaller area providing higher packaging density and shorter distances between circuits and devices, implying shorter interconnects. Shorter interconnects have smaller capacitance.

Therefore, if the device size is minimized throughout a design, the overall physical capacitance will be minimized. Applied on a DSP chip, the minimum device sizing will minimize energy per computation for a given DSP architecture. However, the impact of such device sizing on the circuit speed and its *energy · delay* product is not obvious and depends on many design details. In this thesis, an attempt was made to address this issue through the analysis of behavior

of several known CMOS gate design styles implemented in a LPLV technology, as well as through the design of the specified digital audio filter using smallest possible transistor sizes.

In the filter design, circuits inside major building blocks shown in Figure 2.8 have been designed using minimum sized transistors. Since small transistors have small current driving capability, they must not have large capacitive load, which implies low fanout and short interconnect. Short interconnects were achieved by bit-slicing of the MAC and multiplier block layout. However, high fanout gates and other gates with heavy capacitive loads have been sized larger to have sufficient current driving capability and maintain short transition delays.

Apart from minimum transistor sizing, other circuit level aspects of energy consumption, such as gate design, unnecessary switching activity and leakage current were also considered in the filter design.

2.5.1 Complex Cells: Ripple Carry Adder Circuit

As stated in [9], it has been experimentally proven that if a complex logic function is implemented using basic logic gates such as NAND, NOR, XOR, the total switching dissipation of the circuit will be larger than that of a single complex CMOS gate implementing the same function. Besides, a complex gate is likely to have fewer transistors and shorter propagation delay than a standard cell-like implementation. The full-adder function is a typical complex logic function for which optimized complex gates for VLSI systems are designed. There is at least one such circuit for every CMOS gate design style: standard CMOS, CPL, DPL, DPTL etc.

The circuits considered for the biquad filter RCA implementation are the symmetrical optimized full-adder cell shown in Figure 2.13a and the optimized transmission gate full-adder cell shown in Figure 2.14a.

**Figure 2.13. a) 24-transistor optimized CMOS full-adder cell
b) RCA using the optimized CMOS FA cell**

The full-adder circuit shown in Figure 2.13a has 24 transistors and provides an inverted sum and carry outputs. In order to implement a ripple-carry adder using this circuit it is necessary to invert all the inputs to the full-adder in every other stage as shown in Fig. 2.13b. With inverted

inputs this circuit will still perform the full-adder function, while producing non-inverted outputs. This solution allows for maximum RCA speed as no inverters need to be inserted in the carry propagation path.

Figure 2.14. a) 18-transistor optimized TG full-adder cell
b) RCA using the optimized TG FA cell

A block diagram of an RCA based on the 18-transistor transmission gate circuit in Figure 2.14a is shown in Figure 2.14b. As suggested in [9], an inverter is added every four stages to reduce the degradation of the carry signal due to the distributed RC effect. The four stages following the inverter are based on full adders with inverted carry inputs and outputs. The schematic of

this complementary circuit is shown in Figure 2.15. It also consists of 18 transistors and has almost the same structure as the original one, with the exception of a few internal connections.

Figure 2.15. Optimized TG FA circuit with inverted carry path

The operation of 16-bit adders given in figures 2.13b and 2.14b was simulated in SPICE. In the first simulation setup, widths of all NMOS transistors in both circuits were set to the technology minimum value of $0.8\mu\text{m}$ and widths of all PMOS transistors were set to $1.6\mu\text{m}$. Multiple simulations showed that the transmission gate based adder was significantly faster. In order to match the adders' performance, widths of all transistors surrounded by the dotted rectangle in Figure 2.13a were increased by a factor of two, in all 16 full adders based on this circuit. Repeated simulations with the modified circuit showed almost identical propagation delay for the two types of RCA. In addition to that, the difference in measured energy consumption between the two circuits, for a multiple of input vectors, was virtually indistinguishable.

Since both circuits had the same energy consumption for the same performance, the trans-

mission gate based adder was selected for implementation in the MAC and multiplier circuits due to its smaller size and transistor count. Therefore, the layout of this circuit was expected to be more compact and have lower parasitic capacitances compared to a layout based on the other considered circuit. The adder was implemented with NMOS and PMOS transistor widths of $0.8\mu\text{m}$ and $1.6\mu\text{m}$ respectively.

2.5.2 Circuit Design Styles for Random Logic

Several logic design styles have been proposed that use only NMOS transistors to implement a logic function. Such logic families with differential outputs are Complementary Pass-transistor Logic (CPL) [9], (DPTL) [12], [13] and Swing Restored Pass-transistor Logic (SRPL) [9]. Y-cells proposed in [14] are also based on NMOS transistors but have single ended outputs, while using static CMOS inverters to provide inverted signals when they are required.

Due to avoidance of PMOS transistors that are usually larger than NMOS transistors, these logic families have low input and internal capacitances and high packaging density. Lower capacitances result in increased circuit speed. The major disadvantage of logic families based only on NMOS transistors is their reduced voltage swing due to poor pull-up capability of NMOS transistors. The high logic level driven by a NMOST does not exceed $V_{dd} - V_T$. Therefore, special buffers at the outputs of CPL, DPTL, SRPL or Y-cell logic gates are needed to restore the full logic swing. As an illustration of the use of a restore buffer, a CPL AND/NAND gate and a Y-cell NAND gate are shown in Figure 2.16. In case of the CPL gate, the swing restore buffer consists of two static inverters and one PMOS latch, whereas the Y-cell buffer includes one inverter and one PMOS feedback transistor. In both cases, the role of PMOS transistors in the swing restore buffer is essential for low power design. Although a static inverter would be sufficient to restore the full voltage swing at an NMOS based cell output, the inverter would exhibit static power consumption caused by the voltage of $V_{dd} - V_T$ at its input, which would enable its PMOS transistor to conduct

a non-negligible amount of static current. The purpose of the feedback PMOST is to terminate the inverter's input at the supply voltage providing that its PMOS device is turned-off completely.

In addition to the above mentioned logic families based only on NMOS transistors, Dual Pass-transistor Logic family [9], a modified version of CPL has been considered. DPL gates include both NMOS and PMOS transistors and provide full output swing. However, DPL gates have increased input capacitance compared to CPL.

In order to determine which logic family is likely to give the best results in terms of speed and dissipation when implemented in a LPLV CMOS technology with reduced threshold voltage, several test structures have been designed, simulated and/or implemented in the IMEC LPLV process. These structures include ring oscillators and delay chains based on NAND gates designed in several design styles: conventional CMOS, CPL, DPTL and Dual Pass-transistor Logic (DPL) [9].

**Figure 2.16. a) CPL AND/NAND gate with restore buffers
b) Y-cell NAND gate with restore buffer**

The simulation revealed that for a low voltage, low threshold technology, the speed advantage of pass-transistor gates is totally reversed due to the relatively slow response of swing restore

buffers. This problem is not very pronounced in circuits operating at higher voltages. However in the examined 1 V technology, the difference between the maximum NMOST driven voltage (corresponding to high logic level) at the buffer input, and the V_T of the NMOS transistor in the output buffer is equal to $(V_{dd} - V_T) - V_T = (1 - 0.32) - 0.32 = 0.36V$. In addition to this, the buffer input voltage approaches $V_{dd} - V_T$ exponentially, which implies that the NMOS transistor in the output buffer is weakly turned on and driving very little current until the activation of the feedback PMOST causing positive feedback and completing the transition. This problem is more pronounced in single-ended Y-cells than in differential CPL gates.

To compensate for speed reduction, NMOS transistors in swing recovery buffers can be sized up to increase their initial current driving capability. Regardless of these improvements, based on circuit simulations for 2 input NAND and NOR gates, conventional CMOS gates are faster and consume less energy than any of the NMOS based families with swing restore buffers. Small CMOS gates are more compact than NMOS based ones, whose size is significantly increased by the addition of swing restore buffers. For this reason, all required combinational logic gates in the filter's control logic block were implemented as conventional CMOS gates.

However, some complex gates can be implemented with fewer transistors in one of the NMOS pass transistor families than in the conventional CMOS style. In such cases, the overhead in circuit size and dissipation caused by the introduction of swing restore buffers can become insignificant compared to savings in area and dissipation introduced by the logic part of that gate. These savings are due to smaller transistor count and smaller input capacitance of the NMOS implementation. Such gates can be used to design some parts of a DSP datapath. If they are not located in the critical path, no penalty in terms of performance will be paid.

In this design, a 4 to 1 multiplexer for MAC architecture was implemented using NMOS pass transistors and swing recovery buffer. Its schematic is shown in Figure 2.17.

MUX control inputs are common for all such MUXes in the MAC datapath. These inputs

are decoded outside the MUX cell and connected to all MAC multiplexers as C0 - C3, only one of which can be active at a time.

Figure 2.17 NMOS pass transistor implementation of MAC multiplexer

The obvious advantage of the circuit in Figure 2.17 is its simplicity, with low input and internal capacitance, providing for compactness of the physical implementation.

2.5.3 Latch Circuit Modification

A standard latch circuit with transmission gate input and feedback path is shown in Figure 2.18a. An edge triggered D flip-flop is based on two such latches in cascade as shown in Figure 2.18b. The input capacitance of these circuits, seen by clock or latch control signals and their complements consists of gate capacitances of NMOS and PMOS transistors in data input and data feedback paths. The clock signal is most likely to have the highest switching activity of all signals

in a DSP datapath, which is normally at least two times higher than the switching activity of a register's data input. Hence, the reduction of the clock input capacitance in register circuits can contribute to the reduction of the overall energy consumption.

**Figure 2.18 a) TG based latch circuit
b) D flip-flop with two TG based latches**

The design style used in the modification of the latch circuit shown in Figure 2.19a has been borrowed from the full swing recovery buffer circuit shown in Figure 2.16b, due to some of its properties. First, when the output of this buffer is at the low logic level, the circuit maintains its state due to the PMOS transistor feedback. Second, the input of this buffer can be driven only by a NMOS pass transistor. Finally, in order to turn this buffer into a latch, a positive feedback path for

the case when the output is at the high logic level needs to be provided. This feedback can be implemented using only NMOS transistors.

**Figure 2.19 a) Modified latch circuit
b) D flip-flop with modified latches**

The resulting latch circuit is shown in Figure 2.19 along with the corresponding D flip-flop circuit. The load of its control signals consists only of one NMOS gate capacitance per control input. Assuming that the width of PMOS transistors in transmission gates in Figure 2.18 is double

the size of a NMOS transistor, the input capacitance at control inputs is reduced by a factor of 3.

The circuit shown in Figure 2.19 has been used in the filter design to implement various registers including one-hot encoded counters of the control block, shift registers for external coefficient and data access and some latches in the datapath. Minimum widths of $0.8\mu\text{m}$ have been assigned to input and feedback NMOS transistors as well as to the feedback PMOS transistor. The width of the output PMOS transistor is $1.6\mu\text{m}$, whereas the output NMOS transistor is sized depending on the latch output load and required current drive capability.

Based on SPICE simulations, the modified latch is somewhat slower than the original one, but has the advantage of lower energy consumption in circuits with low data input switching activity. Further, it has fewer transistors and provides for a more compact layout. For this reason, it was used for implementation of register-file cells for coefficient and data storage.

2.5.4 Impact of Glitching in Combinational Circuits

As stated in [5], [8], [9], unnecessary transitions are a major design issue in low power designs with static CMOS gates. These transitions are caused by different delay paths of signals applied to a circuit's inputs. Such glitching behavior can be responsible for a large part of the total energy consumed by a multi-gate circuit. The problem is especially pronounced in combinational circuits with large logic depth, such as compressor schemes for parallel multiplication.

In some circuits, the overhead in energy consumption due to glitching can be reduced by implementing different techniques to match delays of inputs meeting at the same gate. However, delay matching is not possible in carry-propagate adders where carry signal propagates from the lowest to the highest bit position. Therefore, dissipation caused by glitching is an inevitable part of the total energy consumption in all types of CPA based on static CMOS circuits. MAC and multiplier blocks in this design perform multiplication using a carry-propagate adder. The work

reported in [15] claims to reduce CPA glitching energy by employing additional circuitry to achieve self-timed operation. However, no such attempt has been made in this design. Instead, the simplest CPA type has been used, which is known to have the smallest energy consumption per computation [5].

Another source of energy consumption, related to glitching has been noticed in this design. When outputs of a circuit with high glitching activity are driving inputs of a register, unnecessary switching may also occur in the register circuit.

Figure 2.20 Positive edge triggered D flip flop operation

A typical edge triggered D flip flop consists of two identical latches in cascade, one of

them being controlled by the clock signal and the other by the inverted clock. As shown in Figure 2.20, for a positive edge triggered D flip-flop, the second latch is transparent in the first phase of the clock cycle while the first latch is opaque. In the second phase, the first latch becomes transparent and the second one opaque. Hence, all transitions occurring at the flip-flop data input during the second phase will cause internal switching in the first latch, whereas the input transitions occurring in the first phase will not cause any activity in the flip-flop circuit. Consequently, all unnecessary transitions in the combinational circuit driving the register input, that take place during the second phase will cause an overhead in the energy consumption of that register.

In order to assess the importance of this effect, SPICE simulations with energy measurements were performed on the SPICE model of positive edge triggered D-flip flop standard cell from IMEC LPLV standard cell library. Six different stimuli, all of them causing a low-to-high transition at the flip-flop output at the rising edge of the clock, are shown in Figure 2.21 along with the corresponding values of energy consumed by the flip-flop during the clock cycle in which the said transition takes place.

Comparing results for stimuli IN1 and IN2, it appears that there is no significant difference in energy consumption with respect to occurrence of a single transition in the positive or in the negative phase of the clock. Further, if all transitions including the unnecessary ones take place during the positive clock phase (as for stimulus IN3), the consumed energy will still be approximately the same. Also, if the stimulus has the shape like IN5, with an even number of transitions in the positive clock phase, no significant energy will be dissipated on the negative clock edge and energy will be consumed mostly due to the final transition occurring in the second phase.

However, for stimuli like IN4 and IN6, exhibiting an unnecessary transition in the second clock phase, an increase of 60% in energy consumption is observed. This increase can be even higher if more than one glitch occurs during the second clock phase.

Based on the above analysis, from the energy standpoint, it is beneficial to make the clock

Figure 2.21 Register energy due to input glitching

frequency low enough to allow all glitches in combinational logic to settle in the first half of the clock cycle. If this is not possible due to fixed throughput requirements, a small reduction in clock frequency may significantly change the average number of transitions falling into the second phase, thus achieving energy savings. Finally, if the glitches are produced by a CPA circuit, a faster CPA architecture should be considered for implementation. Although a faster adder is likely to consume more energy [5] itself, the total consumption may be reduced when register energy is taken into account.

2.5.5 Memory Circuit Design

Coefficient memory has been implemented using register file cell shown in Figure 2.22a whereas data memory is based on a two-bit cell shown in Figure 2.22b. Based on SPICE simulations, switching energy of memory blocks is negligible compared that of the arithmetic blocks, due to the low memory access rate (one read access per biquad cycle per block for all memory blocks and one write access for the data memory block). However, memory blocks dominate the total chip area and the majority of transistors in the design belong to these blocks. Therefore, it is expected that most leakage dissipation takes place inside memory blocks.

Figure 2.22 a) Coefficient memory cell
b) Data memory cell

2.5.5.1 Estimate of Leakage Dissipation in Memory Blocks

An attempt was made to estimate the total leakage energy of the LPLV filter memory blocks, based on the known cell topology and leakage current specifications for the low-voltage, low threshold process chosen for implementation.

Based on [14], the subthreshold leakage current at room temperature can be estimated as $0.1\text{-}1\text{ nA}$ per micron of transistor width for $V_{gs} = 0$. Worst case leakage paths were identified in Figure 2.23, for the coefficient memory cell for both latch states. A leakage path is a path through a series of transistors in the “off” state, where one end of the path is connected (or driven) to the supply voltage and the other end to the ground. The width of a leakage path is the minimum transistor width along the path. Similar analysis can be performed for the data memory cell.

Figure 2.23 Worst case leakage paths for coefficient memory cell:

a) when logic “0” is stored

b) when logic “1” is stored

The worst case corresponds to the largest possible leakage path width of $4.8\mu\text{m}$, which is created when the stored value is logic zero, while all surrounding busses are set to one. Approximating the data memory cell as two coefficient cells and assuming the subthreshold leakage current of 1nA per micron of transistor width and the leakage path of $4.8\mu\text{m}$, the total maximum leakage current in the coefficient and data memory is $(32 + 32 + 32 + 16) \times 16 \times 4.8 \times 10^{-9} = 8.6\mu\text{A}$. Therefore, the maximum leakage power at 1V supply voltage is $8.6\mu\text{W}$, and the energy per sampling period is $8.6 \times 10^{-6} \times 50 \times 10^{-6} = 430\text{pJ}$. The maximum energy dissipation per biquad section, due to subthreshold leakage is then $(430\text{pJ}) / 16 = 27\text{pJ}$.

RAM blocks tend to be large due to data storage requirements, employing a large number of transistors and having a large number of leakage paths. Since the memory blocks in this design dominate the core area and involve the majority of transistors, their contribution to the total leakage current is much larger than that of the other blocks. Consequently, the leakage dissipation in this design was addressed by analyzing only the leakage energy of memory blocks.

2.5.5.2 Swing Recovery Buffer at Data Outputs

Memory blocks implemented in this design are rather small and do not incur a long delay during read accesses, compared to the required clock cycle. Hence, there was no need for utilization of a sense amplifier at the RAM output data bus. However, a precharging scheme for data output lines was employed since register files have NMOS pass transistors at their outputs, which cannot provide proper high voltage level.

The implemented data read buffer with the precharging and pull-up transistors is shown in Figure 2.24. The precharge transistor M1 is controlled by the memory read signal. Consequently, data output lines are being precharged whenever reading does not take place. In order to prevent discharging of data lines when data is being read from the same location for a rather long time, the

pull up circuit consisting of two PMOS transistors has been provided. According to the algorithm described in section 2.4.3, reading of coefficients and data from their corresponding memory blocks starts in clock cycle 0 of a biquad cycle and lasts until the end of the 16th cycle. The pull up circuit is turned off during clock cycle 0 in each biquad cycle in order to allow the output of the selected memory cell to pull down the precharged line more easily. In the next clock cycle, the pull up circuit is re-activated if the data line is still at the high logic level.

Figure 2.24 Read buffer with precharge and pull up transistors

2.6 Layout Design

2.6.1 Impact of Layout Design on Energy Consumption

Layout design is known to have great impact on the overall energy consumption of a digital chip [8], [9]. Whereas switching activity and gate capacitances are determined in other design stages, values of interconnect capacitances and diffusion capacitances depend very much on the physical design stage.

Diffusion capacitance is partially determined at the circuit design level where transistor sizes are specified. At the layout level, the final values of diffusion capacitances for each transistor are determined. An extensive use of abutment layout style minimizes both the area and the perimeter component of a transistor's diffusion capacitance. As a result, the total switched capacitance is reduced.

As opposed to the straightforward approach towards reducing diffusion capacitance, reduction of the interconnect capacitance and the related energy consumption is a more difficult task. Interconnect lengths are defined between architectural and layout design stages. Statistics provided in [8] show that interconnect capacitance can participate with 40% in the overall switched capacitance of a datapath block. This percentage varies with types of datapath blocks and reaches its maximum in digital filter implementations [8].

Given the significance of interconnect dissipation, an analysis of switching activity at the architectural level may not be a sufficient basis for the selection of the most energy efficient DSP architecture. To provide for correct decision making, feasibility of a compact layout with low interconnect capacitance has to be considered for each examined architecture.

2.6.2 Energy Consumption of Global Signals

SPICE simulations performed in the early stage of the filter design showed that energy, consumed by a datapath block such as a carry-propagate adder or a register, is reduced when all transistors in the simulated circuit are scaled down. Switching of device capacitances is the dominant cause of energy consumption inside datapath blocks. Local interconnects are short and their capacitance is small compared to device capacitance.

However, circuit level analysis of datapath blocks does not take into account the dissipation introduced by switching of global signals used as inputs to these blocks. It is also not obvious whether a global signal capacitance is dominated by input capacitance of datapath circuits or by the parasitic capacitance of the wire distributing that signal to all bit slices in the observed datapath.

In order to optimize the energy consumption of a datapath block it is necessary to make preliminary calculations using wire capacitance specifications for the process selected for design implementation. Other necessary parameters for such calculations are architecture driven. These are: the number of global signals and their average switching frequency, the number of bit slices in the datapath, the input capacitance of a bit slice circuit, and the physical width of a single bit slice.

The analysis of energy consumption of global signals in the MAC block implemented in this design gave the following results:

1. For the minimum width wire in METAL3 layer, used to distribute latch and multiplexer control signals, and for the control signal wire length of 700 μ m, the wire capacitance is approximately equal to 185fF.

2. The total gate capacitance of 18 minimum width transistors driven by the same global signal is 35fF.

3. For $n_{sig}=5$ global MAC input signals, each one having an average number of $N_{avg}=20$ low-to-high transitions during a single biquad computation, the total switching energy is equal to

$$V_{dd}^2 \cdot N_{avg} \cdot n_{sig} \cdot C_{load} = 1V^2 \cdot 20 \cdot 5 \cdot (185fF + 35fF) = 22pJ. \quad (2.7)$$

Several conclusions arise from these results. First, the wire capacitance of a global signal is about 5 times higher than the total circuit input capacitance. Therefore, an effort to reduce the circuit input capacitance in a type of design where minimum size transistors are already used, does not lead to significant energy savings. Instead, an effort to reduce the physical width of the datapath would result in almost proportional reduction of energy consumed by global signals.

Second, the total dissipation of MAC global signals introduces an additional 50% in the datapath energy consumption estimated in section 2.4.1.1. For this reason, the number of global signals should be minimized in the architecture design stage. This happens to be consistent with the choice made in section 2.4.1.1, to implement an architecture that involves fewer registers. Each implemented register introduces another clock distribution wire whose length is equal to the datapath width. This also means that highly pipelined designs might not be suitable for applications that require very low energy consumption due to the length of clock distribution wires.

Third, in order to provide for an efficient estimation of the layout aspect of energy consumption, some up front knowledge of physical properties of building blocks is required. This involves the use of datapath libraries with optimized building blocks and known parasitic capacitances. Also a high level simulation tool, able to estimate energy consumption based on parameters of datapath library components is needed for fast and efficient architectural design for low power.

Finally, a layout design guideline has been observed that helps to reduce the energy consumed by high frequency global signals. It will be explained through the following example.

A DSP datapath block is shown in Figure 2.25. Its has two pipeline stages and a width of 16 bits. Further, it was assumed that a combinational circuit in a single bit slice has the same

dimensions as the one bit register driven by that circuit with the aspect ratio 2:1. Two possible physical arrangements are shown in Figure 2.25 and clock wire length is calculated for both cases. It turns out that the clock wire is significantly shorter for the arrangement in Figure 2.25b and so is the energy dissipated in clock distribution. Generally, in order to reduce the energy of global signals, datapath blocks should be designed to be as narrow as possible at the expense of their length. If standard cells are used, their longer side should be oriented in the datapath direction.

Figure 2.25 Impact of cell aspect ratio on global signal length

Another layout design guideline, following from the process based calculations, can be expressed in the form of a simple rule applicable on this particular process. That is: *The capacitance of 10 microns of wire is approximately equal to the gate capacitance of a minimum size transistor whose width is 0.8 μ m.*

2.6.3 LPLV Filter Layout

The layout of the LPLV filter has been organized hierarchically. It resembles Figure 2.8 and includes the same major building blocks. All computational and memory blocks have been bit-sliced, providing repeatable structures. In addition to that, wherever it was possible, bit slice widths of different building blocks were adjusted to have the same width in order to further alleviate the layout design and make their interconnects short. This way MACs and their corresponding coefficient memory blocks were bit-sliced together. Also, data memory, gain coefficient memory and the inner loop counter of the control logic block were sized to have the same bit slice width.

The organization of the MAC block layout and the top level floorplan are described in the remainder of this section.

2.6.3.1 MAC Layout

MAC is the most critical block in this filter design, both in terms of performance and energy consumption. Therefore, special attention was paid to providing a compact and regular layout for this block.

The floorplan of the MAC block layout including coefficient memory, control signal drivers and external access circuitry is shown in Figure 2.26. Whereas datapath spreads vertically, control signals are laid out horizontally across the coefficient memory and the computational block. The shift register used for external access to coefficient memory is located at the top of the memory block. Buffers driving high fanout control signals are located along the left and right side of the MAC block.

As suggested by Figure 2.26, MAC layout is compact, regular and approximately rectangular. Bit slices of the computational block and the coefficient memory were sized to have the

same width. This was achieved virtually without paying any penalty in area, due to the fact that two coefficients, ck_1 and ck_2 , are required for MAC computation. As shown in Figure 2.27 the width of two memory cells approximately equals the width of the computational bit slice. Connections within a bit slice are kept short and mostly minimum size transistors are used.

Figure 2.26 MAC floorplan

The full-adder circuit is known to be critical for the chip performance and to contribute a great deal to the MAC energy consumption. The layout of the non-inverted and the inverted full adder cell is shown in Figure 2.28. An effort was made to achieve maximum compactness of these cells. Local interconnects have been implemented in two metal layers.

Figure 2.27 Block diagram of the MAC bit slice

2.6.3.2 Chip Floorplan

The floorplan of the filter chip is shown in Figure 2.29. There are three types of long interconnects between major building blocks. These are the datapath bus, the filter address bus and various control signals driven by the control block. Since long wires represent a heavy capacitive load, it is important from the energy consumption standpoint that their switching activity is relatively low compared to that occurring inside the building blocks where interconnects are short.

**Figure 2.28 a) Non-inverted full-adder cell
b) Inverted full-adder cell**

The total length of all datapath busses is approximately equal to $3000\mu\text{m}$. Given the following parameters:

bus activity: $BACT_{avg} = 1$ update per biquad cycle,

assumed probability of a low-to-high transition: $p_{0-1} = 0.25$

bus width: $BW = 16$ bits,

capacitance per micron of wire length (from [14]): $CW = 0.265 \text{ fF}/\mu\text{m}$,

the total energy consumed by all data busses per biquad operation is approximately equal to

$$L \cdot BACT_{avg} \cdot BW \cdot p_{0-1} \cdot CW = 3pJ \quad (2.8)$$

For the filter address bus, which is updated once per biquad cycle, there is exactly one low-to-high transition. For the bus length of 1000 μ m and the same wire capacitance per micron of length, the total energy consumption of the filter address bus is only 0.26pJ per biquad.

Among the remaining signals distributed using long interconnects at the top hierarchical level, the most energy is consumed by the clock signal and it amounts to 8pJ per biquad cycle. Most other signals are switched only once in a biquad cycle. Given their total length and switching activity, the total energy consumption of all top level control signals including the clock is estimated to be less than 20pJ per biquad cycle.

Based on these estimates and estimates of dissipation of major building blocks, the energy consumed by top level interconnects does not exceed 15% of the total chip energy per biquad computation. However, when the energy of long interconnects internal to MAC and multiplier blocks is taken into account, the dissipation attributed to wiring amounts to about 40% of the total dissipation.

The filter layout includes 28000 transistors. Its core area is 1.6mm², whereas the total area including the 40-pin padframe is 5.3mm². The core area is dominated by memory blocks.

Figure 2.29 Filter chip floorplan

2.7 Summary

In this chapter, the design of a digital filter suitable for portable systems has been described with an emphasis on low power design technique. The filter has been implemented in IMEC 1V CMOS process with reduced transistor thresholds.

The filter design served two basic purposes. The first purpose was to provide guidelines for designing of compact, extremely low power DSP circuits required in personal communications systems. Sources of energy consumption were identified at all design levels and their contribution to the total dissipation was estimated. These estimates were used in selecting the architecture, circuit and layout details for implementation of the described filter. Since the estimates were based on process specific parameters, the obtained results are expected to provide a realistic insight in energy costs involved with typical datapath components and computations, when designing with this technology. Further, a couple of specific sources of energy consumption were highlighted.

The second purpose of this design and implementation was to characterize the IMEC LPLV process in terms of speed, energy consumption and leakage current. Also, feasibility of different CMOS design styles was analyzed with respect to reduced transistor thresholds and reduced supply voltage. This analysis was conducted by examining circuits used in the filter design as well as through the implementation of some separate test structures.

REFERENCES

- [1] Gennum Corporation, "Programmable Analog Signal Processor GP520A - Advance Information Note", Burlington, Ontario, 1992
- [2] R. Cavin, W. Liu, "Emerging Technologies: Designing Low Power Digital Systems", IEEE Press, 1996
- [3] J. Monteiro, S. Devadas, "Computer-Aided Design Techniques for Low Power Sequential Logic Circuits", Kluwer Academic Publishers, Boston 1997
- [4] A. P. Chandrakasan, R. Mehra, M. Potkonjak, J. Rabaey and R. W. Brodersen, "Optimizing Power using Transformations", IEEE Transaction on CAD, vol. 14, pp. 12-31, 1995
- [5] J.M. Rabaey, M. Pedram, editors, "Low Power Design Methodologies", Boston: Kluwer Academic Publishers 1996
- [6] S.R. Powel & P.M. Chau, "A Model for Estimating Power Dissipation in a Class of DSP VLSI Chips", IEEE Transactions on Circuits and Systems, vol.38, pp 646-650, 1991
- [7] N. H. E. Weste and K. Eshraghian, Principles of CMOS VLSI Design, Reading, MA, Addison-Wesley, 1993
- [8] A.P. Chandrakasan & R.W. Brodersen, editors, "Low Power Digital CMOS Design", Kluwer Academic Publishers, Boston 1995
- [9] A. Belaouar & M.I. Elmasry, "Low Power Digital VLSI Design", Kluwer Academic Publishers, Boston 1995
- [10] J.Rabaey, L.Guerra, R.Mehra, "Design Guidance in the Power Dimension", International Conference on Acoustics, Speech and Signal Processing, pp. 2837-40, 1995
- [11] C.A.T. Salama - "LVLP Limitations, Impact of Process and Device Design", Micronet Course on LP Design, Toronto 1996
- [12] J. H. Pasternak, "High-Speed Differential Pass-Transistor Logic", PhD Thesis, University of Toronto, 1991
- [13] J. Mokleshi, "Design Automation for Differential Pass-Transistor Logic", MASc Thesis, Toronto 1995
- [14] IMEC, Electrical Parameters CDL50T and CAL50T, Leuven, 1996
- [15] U.Ko, P.T. Balsara, W. Lee "A Self Timed Method to Minimize Spurious Transitions in Low Power CMOS Circuits", IEEE Symposium on Low Power Electronics, pp. 62-67, 1994

CHAPTER 3

SIMULATION AND EXPERIMENTAL RESULTS

In this chapter, the results obtained in simulation and measurement results for the designed digital filter are presented. Special emphasis is put on estimation and measurements of energy consumption of various parts of the design.

3.1 Simulation

3.1.1 Energy Estimation at the Circuit Level

HSPICE was used as a tool for energy estimation and functional verification of all building blocks throughout the design. HSPICE provides means for accurate measurement of average current over a specified time interval. Energy of adders, registers, control and memory blocks was obtained by simulating the transient behavior of these blocks for particular stimuli and having HSPICE measure the average power supply current $\bar{I}_{vdd}$ over an interval T_{settle} that is equal or greater than the maximum delay of the simulated circuit. Assuming a constant supply voltage V_{dd} , the total energy E_{total} consumed by the circuit during T_{settle} can be expressed as

$$E_{total} = V_{dd} \cdot \bar{I}_{vdd} \cdot T_{settle} \quad (3.1)$$

Input capacitance of every gate or functional block presents a load to the circuit that drives that input, since an issue arises whether the actual load capacitance should appear in the testbench for energy estimation of a particular building block i.e. whether the load capacitance of a circuit belongs to that circuit or not. If the load capacitance is included in the testbench the switching behavior of the circuit being simulated will match its behavior in the actual design. However, the

measured energy will include the energy dissipated in charging of both the input and load capacitance of that circuit. If the energy consumption of the whole design is later estimated as the sum of the average energies of its building blocks, it will be overestimated since the input capacitance of each block will be taken into account twice. To correct this imprecision, the energy spent in charging the load capacitor should be subtracted from the value obtained in the simulation for the particular input transition

$$E_{total} = V_{dd} \cdot \bar{I}_{vdd} \cdot T_{settle} - E_{load} \quad (3.2)$$

Further, the simulation testbench has to include a circuit with appropriate drive strength to drive the inputs of the tested circuit. The driving circuit serves two purposes. First, as it is powered by the same V_{dd} supply, the charging energy of the input capacitance will be included in the measurement. Second, the properly selected drive capability of the driving circuit will cause realistic transition times in the internal nodes of the tested circuit thus providing for the accurate timing of short circuit current which affects the total power consumption. The logical choice for the driving circuit is a simple inverter because it is the circuit with the minimum diffusion capacitance and does not have any internal nodes.

The testbench used in simulation to estimate the energy consumption as well as for functional verification of the digital filter building blocks is shown in Figure 3.1

Figure 3.1. Testbench for energy/functional simulation

It is also possible to use a testbench that does not include the load capacitance at all. It is simpler than the previous one and useful when the actual load capacitance is not known. Also there is no need to calculate and subtract the load charging energy from the measured energy. However, this testbench gives less realistic performance characteristic with very fast output transitions which increase the internal short circuit current while giving a shorter propagation delay.

An example of energy consumption calculation is shown in Figure 3.2. It shows a layout based simulation plot of power supply current consumption for the multiplier block. HSPICE measured an average current of $97.6\mu\text{A}$ over an operating interval of 890ns . The total energy dissipated can then be calculated as $1\text{V} \times 97.6\mu\text{A} \times 890\text{ns} = 86.7\text{pJ}$ and is consumed by the 17-bit adder and register as well as several long control lines.

Figure 3.2. Simulated current consumption of the multiplier block

3.1.2 High-Level Functional Simulation

The complexity of the implemented filter circuit, as well as the complexity of its operation over a relatively large number of clock cycles disallow the use of HSPICE for functional verification of the whole design. Functional simulation at the top level was performed using AWSIM - the analog simulator for digital circuits developed at the University of Toronto [1]. The layout based HSPICE netlist of the whole design was used as an input to this simulator thus partially trading simulation accuracy for simulation speed. The simulator was configured with transistor parameter values for the implementation process. The "C" program "ds3" whose listing is given in Appendix C was written based on the desired behavior of the filter to provide reference outputs for the implemented design. The outputs obtained from AWSIM simulations were then compared with "ds3" outputs for the same data input and coefficient set and were found to be identical.

3.1.3 High-Level Estimation of Energy Consumption

Simulation based measurements of the average energy of a complex design typically involve the use of a large number of input vectors. Since no higher level energy estimation tool was available, the average energy of each building block was estimated using the values of energy consumption obtained from HSPICE simulations of building blocks for particular transitions, including parasitic capacitances extracted from layout. It was assumed that all datapath inputs of all building blocks are uncorrelated with equal probabilities of the high and low state. Spatial and temporal correlations [2] of adder and register inputs were consciously neglected in order to simplify the analysis. For all blocks with deterministic switching activity, the energy consumption was calculated as the simulated energy consumption for the particular input switching event factored by the number of such events. Energy of all building blocks was expressed as the total average energy consumed during the execution of one biquad computation. Energy of the clock and control signal traces was calculated based on the trace length in the filter layout, the capacitance

of the used metal layers and the switching activity of each signal. Similarly, the energy of long data buses was estimated by assuming the probability of transition from “0” to “1” for each signal to be equal to 0.25 and using the known length and capacitance of metal wires.

The major components of energy consumption of the designed filter were identified in the previous chapter. Table 3.1 shows the breakdown of the filter energy consumption with respect to different building blocks. The relative contribution of each building block to the overall consumption is also shown. With the exception of leakage energy, the dissipation of all other sources is directly related to their switching activity.

Table 3.1: Estimated breakdown of energy consumption

#	Dissipation source	energy/biquad [pJ]	percentage
1	adders	93	28%
2	registers	102	30%
3	4:1 MUXes	25	7.4%
4	MAC, MUL control lines	60	17.8%
5	memory access	5	1.4%
6	leakage current	27	8%
7	top level wiring	25	7.4%
	TOTAL	337	100%

The energy consumption per biquad computation attributed to leakage current was calculated by estimating the average leakage current for the whole design as described in 2.5.5.1 and assuming that the time slot allocated for a single biquad computation is equal to 1/16 of the sampling period of 50 μ s i.e. 3.125 μ s. This approach is valid if the chip is configured as a 32nd order filter performing 16 biquad computation during the sampling period. However, since the total leakage energy during the sampling period is constant and equal to 16 x 27pJ = 432pJ, the energy per biquad will be approximately 2.5 times higher if the filter is configured to perform only a single biquad computation. In that case the contribution of switching activity will be equal to 337pJ -

$27\text{pJ} = 310\text{pJ}$ and the total energy consumption per biquad will be $432\text{pJ} + 310\text{pJ} = 742\text{pJ}$.

3.2 Experimental Results

The designed digital filter chip was implemented in the IMEC 1V $0.5\mu\text{m}$ CMOS process. The photomicrograph of the chip core area is shown in Figure 3.3.

Figure 3.3. Photomicrograph of the core area

3.2.1 Functional Testing of the Prototype

The functional verification was performed using a HP75000 digital tester. Since the test system provides TTL and 5V CMOS compatible inputs and outputs a testbench shown in Figure 3.4 was built to provide proper voltage level interfacing. The voltage of the tester outputs was simply reduced using pairs of resistors as voltage dividers while the voltage of chip outputs was expanded to the 0-5V range using high-speed analog comparators LM360 (National Semiconductor) with TTL compatible outputs. The negative input of the comparator was set at 0.5V - the 1V CMOS threshold voltage.

Figure 3.4. Testbench for functional verification

Input vectors were created for loading the coefficient memory, configuring the filter for the number of biquads, shifting in of the serial data input and starting the biquad operation. Functional correctness of the biquad operation was verified against the results obtained from program "ds3" that was used as a reference in functional simulations.

The prototype successfully passed the test. It was found to be fully functional at the clock

frequency of 20MHz. The minimum clock frequency required by audio applications assuming the full 32nd order filter configuration is 6.7MHz. Obviously the same architecture could be used to implement even higher order audio filtering by providing more coefficient memory to store coefficients of the additional biquad sections.

3.2.2 Measurement of Energy Consumption

Equation (3.1) can be used to express the energy consumed by the whole filter while performing biquad operation. In order to calculate energy using this formula it was necessary to measure $\bar{I}_{vdd}$ and determine T_{settle} . For this purpose, a 43Ω resistor was inserted in series with the chip power supply providing a voltage drop proportional to I_{vdd} that can be probed by an oscilloscope [3].

The resistor value was chosen such as to keep the voltage drop small enough not to reduce the chip supply voltage significantly. Using HP75000, a set of coefficients was loaded into the memory of the tested chip and the chip was programmed to perform a single biquad operation whenever its "START" input is asserted. Subsequently, repetitive, back-to-back biquad operations were started by the automatic tester, while providing varying data input for the filter for each biquad operation. A Tektronix TDS 380 digital Oscilloscope was used to probe and average the voltage across the inserted resistor. The average current for the period of biquad activity was read from the oscilloscope. This period, determined by the frequency of the filter clock, was used as T_{settle} in equation (3.1).

Energy consumption of a 32-bit shift register used for programming of coefficients was also measured in the described way. In this case the applied input vector pattern was providing the clock and data input only to the shift register while the main filter circuit was kept inactive.

The dynamic energy consumption was measured for several different sets of biquad filter

coefficients.

In order to measure the energy consumed by the core circuit in biquad operation it was necessary to estimate the energy consumed by an input pad when a transition of an input signal occurs. Since the designed prototype circuit requires external clock and the biquad operation is performed in 21 clock cycles, the energy per biquad was obtained by subtracting the energy consumed by an input pad in 21 positive and 21 negative transition. Some typical measurement results are summarized in Table 3.2.

Table 3.2: Typical energy measurements on the filter prototype

setup	structure	energy	measured for
1	input pad	4.8pJ	0-1-0 transition
2	32-bit register	4.9pJ	1 clock cycle, all flip-flops in transition
3	biquad filter	155pJ	all coefficients, data input set to all 0s
4	biquad filter	314pJ	coefficients, data: evenly spread 0s and 1s
5	biquad filter	300pJ	same coefficients, random input
6	biquad filter	446pJ	coefficients all 1, random input

The measurement results for setups 4 and 5 are within 4% agreement with the estimates based on simulation. In both these cases the assumption of even spread of 'zeros' and 'ones' was made. However, for setup 6, where all coefficients were set to all 'ones', an increase of 43% in dynamic dissipation was measured. Further, if all filter coefficients and data input are set to zero (setup 3), no switching activity should occur in arithmetic circuits and related registers since all their input bits will be set to zero. In such case, the total energy consumption per biquad should be determined by the activity of the control state machine and the capacitance of long wires driven by clock signal as well as by the outputs from the control state machine. The measured dissipation is 50pJ higher than the sum of the average energy per biquad of MAC and MUL control lines, memory access and top level wiring. The discrepancy can be attributed to the switching activity of

the control state machine for which, no estimate was made.

Values from Table 3.2 also suggest that the energy consumed in biquad computation strongly depends on coefficient values. Therefore, the final stage of energy optimization would involve the choice of the most energy efficient coefficient set for a particular filter characteristic.

Based on the average measured value of dynamic dissipation and assuming nominal leakage current, the presented design in its full 32-nd order filter configuration could be operated for approximately 11 days from a single 1V battery cell with the energy fund of 30mA hours.

The design presented implements a generalized structure of a second order filter. However, for some types of high pass, low pass and band pass filters (Butterworth, Bessel, Chebyshev I) values of coefficients in the numerator of any second order filter section are equal to either 1 or ± 2 . For some other filter types (Chebyshev II, elliptic), at least one coefficient is equal to one. Hence MAC2 in this design can be somewhat simplified reducing the overall switching activity and switched capacitance.

Further savings can be achieved by taking advantage of more advanced CMOS processes with smaller feature size and by exploring different architectural solutions. A more parallel implementation of MAC units, operating in fewer clock cycles would reduce the total energy consumed by registers although it would necessarily lead to an increase in the design area. Also, the energy consumption could be reduced at the circuit level by optimizing the short circuit current of all building blocks.

Finally, speed improvements could be achieved by replacing the implemented ripple-carry adder by a faster adder type as simulations suggest that the critical path of the whole filter is determined by the propagation delay of this adder.

3.6 Summary

The designed low power digital filter was implemented in a low-voltage CMOS process. Its functionality, performance and energy consumption were verified in two steps.

First verification was performed in simulation. Functional correctness, timing and energy of all building blocks was verified in HSPICE simulations. The overall design functionality was checked using AWSIM - an analog simulator for digital circuits. Since it was not possible to use simulation to obtain the total energy consumption, the average dissipation of the filter was estimated using known energy properties of building blocks and assumed data statistics.

Second verification of the design was performed on the prototype chip. The functionality of the filter was confirmed using automatic test equipment for digital integrated circuits. The prototype operated correctly for clock frequencies up to 20MHz, which is three times higher than the minimum frequency required by the sampling rate for processing of audio signals.

The measured energy dissipation of the prototype, consumed in biquad operation, matches the value of estimated average energy closely, proving the correctness of the applied method of energy estimation.

REFERENCES

- [1] D. M. Lewis, "AWSIM Manual", University of Toronto, 1996**
- [2] A.P. Chandrakasan & R.W. Brodersen, editors, "Low Power Digital CMOS Design", Boston: Kluwer Academic Publishers, 1995**
- [3] J.M. Rabaey, M. Pedram, "Low Power Design Methodologies", Boston: Kluwer Academic Publishers, 1996**

CHAPTER 4

CONCLUSIONS

This thesis has addressed the implementation issues of digital signal processing circuits in low voltage low threshold CMOS technologies. The obtained results show that implementation of a high order filter for micropower audio applications is possible in a 1V CMOS process with reduced, single threshold voltages of $\pm 0.3V$. The functionality of the implemented design was verified on a prototype and its dynamic dissipation was measured to be in the range of 300pJ per biquad computation. Featuring such low energy consumption, the designed filter is directly applicable in battery operated personal communication systems. Further, standard CMOS gates were found to be significantly faster than any of NMOS pass transistor styles. The use of NMOS pass transistor gates is still practical in some cases where such gates can help to achieve more compact layout and lower global wiring dissipation.

Several new solutions are presented in this thesis. First, this complete design presents the first example of implementation of a complex DSP function in a single threshold 1V CMOS process with energy consumption comparable to that achieved in multiple threshold processes for a similar functional complexity. Second, an original multiplier-accumulator architecture was designed that minimizes three components of energy consumption: local switching, global interconnect and static leakage. Third, novel latch, multiplexer and RAM data recovery buffer circuits were designed. Finally, two sources of dissipation were identified that had not been reported previously. These are register glitching and cell orientation dependent dissipation of global signals.

The filter design presented is probably not the most energy efficient one for any particular filter application as it implements a generalized digital filter. However the design features easily predictable energy consumption due to the fact that it consists of relatively small building blocks

whose energy can be determined easily by a circuit simulator. Such approach is useful when designing for known target dissipation.

In future work, optimizations of the designed filter will be performed to achieve more energy savings. One of these changes is replacement of MAC2 in the direct path with two adders. Further, variations of the existing MAC architecture will be made in order to reduce register switching activity and biquad latency of the current design. Also, circuit techniques for reduction of leakage current will be investigated.

Increase in circuit level speed will be sought by replacing the currently used carry propagate adder by a faster adder type.

Analysis of signal processing characteristics of this filter design will be performed in order to define a way to specify coefficients that provide both good filtering properties and low power operation.

APPENDIX A

ARCHITECTURAL LEVEL ESTIMATE OF MAC ENERGY CONSUMPTION

This appendix describes the estimates, made at the architectural level, of energy efficiency of MAC architectures shown in Figure 2.4 and Figure 2.5.

Energy in the MAC architecture in Figure 2.4 is mostly consumed by CPAs and registers included in the multipliers. The effect of AND gates on the total energy consumption can be neglected in the first order analysis because a 2-input AND gate is a much smaller circuit than a bit slice of a CPA_{mul} . Therefore its energy is negligible compared to that of a CPA_{mul} for the same input switching rate. The energy consumption of the latches is also negligible because they are activated only once during a biquad computation, while a latch energy per transition is approximately 1/2 of that of a register (since one register consists of two cascaded latches).

Similarly, the energy consumption of AND gates shown in the modified MAC architecture can be neglected in the first order analysis.

In order for X_k to be calculated by the MAC shown in Figure 2.4, each CPA_{mul} has to perform 16 additions and CPA1 and CPA2 have to do one addition each. Hence, the MAC will perform 34 16-bit additions in every biquad operation. Each one of the two 16-bit registers is written in 16 times during the multiplication execution making the total of 32 register updates.

The modified architecture shown in Figure 2.5 also performs the operation given by equation (2.2). Its total activity in calculating X_k based on the equation (2.2) consists of 18 additions, 17 updates of the multiplexer control inputs and 17 register updates.

This brief activity analysis suggests that the overall switching activity of the modified architecture is significantly lower. However, it is not obvious whether the average energy dissipated in addition and register updates is the same for both architectures since the input signal statistics are different for the two architectures. The relative contribution of the multiplexer circuit to the total energy of the modified architecture is also unknown.

In order to quantify the estimates of energy consumption of the two MAC architectures described above, a number of SPICE simulations was performed for a 16-bit CPA, a 16-bit edge triggered register and a 16-bit 4:1 multiplexer, which are the three key components present in one or both observed architectures. Particularly, the purpose of the simulations was:

1. to determine average values of the energy consumption of the selected register and 4:1 multiplexer circuits, relative to the average energy consumption of the selected CPA circuit, and
2. to examine if there is any difference in the average energy consumption in CPA_{mul} circuits between the two observed architectures.

In this stage of the design, the choice of circuits for implementation had to be made in order to perform SPICE simulations. Since the design goal was to achieve as low overall energy per computation, it was assumed that the implemented datapath circuits should be based on minimum size transistors. Further, using experimental results presented in [1], showing that the average energy consumption of a ripple-carry adder (RCA) is the lowest compared to all other known adder types, all MAC CPAs were implemented as ripple-carry adders. This decision involves a speed-for-low energy trade-off, since RCA is the slowest of all adder types. The RCA was constructed as a chain of 18-transistor pass-gate full adders described in [2]. This full-adder circuit was chosen because it has the smallest number of transistors of all known static CMOS full-adder circuits.

The register circuit used in SPICE simulations consists of two inverting latches with standard CMOS pass-gate inputs, as described in [2]. The circuit consists of 16 transistors.

Multi-bit registers and multiplexers are pure collections of non-correlated gates so their analysis reduces to the analysis of a single gate. For the simulated register circuit, two distinct amounts of consumed energy were measured - E_{clk} for the cycle without register state change and E_{max} for the cycle with a state change. Virtually no difference in energy consumption was observed for output transitions from “high” to “low” and from “low” to “high”.

The energy consumption E_{clk} measured in the cycle without input/output data change is completely attributed to the activity of the clock input. Hence it is possible to calculate the amount of energy caused only by the data activity at the register input as

$$E_{tran} = E_{max} - E_{clk} \quad (A1)$$

where E_{max} is the total energy consumed by a register in a clock cycle in which an output transition occurs. Assuming that probabilities of “low” and “high” states at the 16-bit register inputs are equal and that subsequent input values are not correlated, the probability of the occurrence of a transition p_{tran} is given by equation (A2) where p_0 and p_1 are probabilities of “low” and “high” states at the register output, respectively.

$$p_{tran} = p_0 \cdot p_1 + p_1 \cdot p_0 = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} \quad (A2)$$

Therefore, the average register energy per clock cycle can be expressed as

$$\bar{E}_{reg} = n \cdot (E_{clk} + p_{tran} \cdot E_{tran}) = 1.3pJ \quad (A3)$$

where n is the number of bits in the register.

A similar approach was adopted in evaluation of the average energy of a multiplexer. Assuming the equal probability of “zeros” and “ones” in $ck1$, c_{k2} , $c_{k1} + c_{k2}$, x_{k-1} and x_{k-2} and taking into account that the fourth multiplexer input is set to all 16 “zeros” for most of the time during MAC operation, the probability of a “zero” at the multiplexer output was calculated as

$$p_0 = 0.75 \cdot 0.5 + 0.25 \cdot 1 = 0.625 \quad (A4)$$

Consequently, the probability of a “one” is

$$p_1 = 1 - p_0 = 0.375 \quad (A5)$$

Assuming that MUX control inputs x_{k-1} and x_{k-2} are updated in every clock cycle, the probability of both their values staying the same is $p_{noact} = 0.25$. In case of this event, the energy dissipated by the multiplexer will be $E_{noact} = 0pJ$. The probability of a transition event at a multiplexer output is then calculated as

$$p_{tran} = (1 - p_{noact}) \cdot (p_0 \cdot p_1 + p_1 \cdot p_0) = 0.352 \quad (A6)$$

The probability of the event of control inputs being switched without output transition taking place is

$$p_{notran} = 1 - p_{tran} - p_{noact} = 0.398 \quad (A7)$$

Energy dissipated in the selected 16-bit multiplexer circuit, based on SPICE simulation, for the transition on all multiplexer outputs is $E_{tran} = 0.69pJ$, regardless of the transition direction. Energy dissipated only by switching in the multiplexer control signals, but without any output switching is $E_{notran} = 0.119pJ$.

Finally, the average multiplexer energy can be calculated as

$$\bar{E}_{mux} = 0 \cdot p_{noact} + E_{tran} \cdot p_{tran} + E_{notran} \cdot p_{notran} = 0.29pJ \quad (A8)$$

The energy of a multi-bit ripple-carry adder cannot be obtained by a hand calculation. The total switching activity in RCA depends on values and transitions on all its inputs as well as on the actual circuit delays in its carry chain. Multiple simulations were run on the selected RCA circuit with different input vectors applied. The estimate of the average RCA energy consumption per addition, $\bar{E}_{rca}$ was calculated by averaging the obtained results. Energy consumption of a RCA in

any observed addition strongly depends on both the previously applied and the currently applied input vector. For an extreme case, with a particular combination of inputs, the measured energy per addition was as large as $1.7pJ$. The other extreme is exactly $0pJ$ for the case when no change in adder inputs occurs in a series of consecutive additions. However, for most simulations including those for which the applied sequence of inputs had matched the sequence produced in a real shift-add multiplication, the measured RCA energy was consistently in the range $0.7-1.0pJ$. Hence, the average RCA energy per addition was estimated as

$$\bar{E}_{rca} = 0.85pJ \quad (A9)$$

The above results suggest that the average energy consumed by the 16-bit multiplexer is significantly smaller than that of an adder. From equations (A3) and (A9), the average multiplexer energy relative to the average RCA energy is

$$\bar{E}_{mux} = 0.34 \cdot \bar{E}_{rca} \quad (A10)$$

It is also possible to compare the energy consumption of an adder with that of a register assuming that a MAC RCA performs one addition per clock cycle. It follows from equations (A3) and (A9) that

$$\bar{E}_{reg} = 1.53 \cdot \bar{E}_{rca} \quad (A11)$$

i.e. that, for the selected circuits, the use of a register is more expensive in terms of energy consumption than the use of a RCA.

Regarding the second purpose of these simulations, no significant difference in the average RCA dissipation was noticed between two groups of simulation results, one obtained using sequences of input stimuli matching sequences of RCA inputs generated in a multiple addition operation given by equation (2.5), and the other obtained using sequences of input stimuli matching sequences present in shift-add multiplications.

Based on the simulation results and the above conclusions the average energy consumption of MAC architectures shown in figures 2.4 and 2.5 can be expressed in terms of $\bar{E}_{rca}$. For MAC1 in Figure 2.4

$$\bar{E}_{mac1} = 34 \cdot \bar{E}_{rca} + 32 \cdot \bar{E}_{reg} = \bar{E}_{rca} \cdot (34 + 32 \cdot 1.53) = 84 \cdot \bar{E}_{rca} \quad (A12)$$

while for MAC2 in Figure 2.5, the average total energy is:

$$\bar{E}_{mac2} = 18 \cdot \bar{E}_{rca} + 17 \cdot \bar{E}_{reg} + 17 \cdot \bar{E}_{mux} = 50 \cdot \bar{E}_{rca} \quad (A13)$$

MAC2 architecture has been chosen for implementation because it consumes 40% less energy.

REFERENCES

- [1] J.M. Rabaey, M. Pedram, editors "Low Power Design Methodologies", Boston: Kluwer Academic Publishers 1996
- [2] A. Belaouar & M.I. Elmasry, "Low Power Digital VLSI Design", Kluwer Academic Publishers, Boston 1995

APPENDIX B

A SEQUENTIAL MULTIPLIER WITH IMPROVED ENERGY-DELAY PRODUCT

High speed parallel multipliers often employ a compression scheme such as Wallace's tree, for adding of partial products. Computational delay in such multipliers is a logarithmic function of its complexity [1]. A multiplier design is organized in bit slices. The bit slice n is a compressor whose inputs are partial products corresponding to the n -th bit position, as well as *carry* outputs from the bit slice $n-1$. Its outputs are carry bits feeding into the bit slice $n+1$ and the n -th bit of the final result p_n .

A compressor based 16 by 16 multiplier for implementation in a LPLV DSP system has been designed and simulated with IMEC pl5000 process parameters. Due to the arguments of compactness of such systems stated in Chapter 2 and also in order to make it comparable with the shift-add multiplier implemented in the filter design, the multiplier was designed to operate in 16 clock cycles.

The parallel implementation of a 16 x 16 multiplier involves 32 bit slices of variable sizes. This parallel scheme has been turned into a sequential one by implementing only two largest consecutive bit slices, corresponding to bit positions 15 and 16. Each one of these bit slices has 31 inputs where at maximum, 16 inputs are driven by partial products and 15 by carry signals from the preceding bit slice. These two bit slices are able to perform the operation of any two consecutive bit slices if the inputs not required in the observed computation are driven by logic "zeros".

As shown in Figure A.1, all carry outputs of the first bit slice are connected to the corresponding inputs of the second one. The outputs of the second bit slice are captured in a 15-bit reg-

ister. The outputs of this register feed back into the carry inputs of the first bit slice.

Figure B.1 Architecture of the compressor-based sequential multiplier

The top level architecture of the proposed multiplier shown in Figure A.1 also includes a block responsible for sequential generation of partial products. This block consists of two shift registers and 32 two-input AND gates. The first, 8-bit shift register is used for shifting of even bits $A_0, A_2 \dots A_{14}$ of the 16-bit multiplier A , A_0 being the least significant bit. The second, 9-bit shift

register is used for shifting of odd bits of the multiplier $A_1, A_3 \dots A_{15}$. The 16-bit multiplicand B is applied as a parallel 16-bit input to the partial product generation block. One input of each AND gate is driven by a multiplicand bit B_i , whereas the other is driven by a shift register output.

Two bits, P_{even} and P_{odd} are available in each clock cycle at COMPRESSOR 1 and COMPRESSOR 2 outputs, respectively. In clock cycles $n=1 \dots 16$, these outputs represent bits of the 32-bit multiplication result P as follows: $P_{even} = P_{2(n-1)}$ and $P_{odd} = P_{2n-1}$, where P_i is the i -th bit of P .

The architecture of the two compressors is shown in Figure A.2. Each one of them consists of 15 full-adders. Even without simulation, it is clear that this architecture provides faster multiplication than the add-shift multiplier based on a ripple-carry adder, implemented in the filter design. There are 16 full-adders in the critical path of the latter architecture, as opposed to only 5 in the critical path of the former one.

Schematic based SPICE simulations were performed in order to assess the speed and energy consumption of this multiplier, when it is implemented with the same full-adder and register circuits used in the simulations described in section 2.4.1.1. The average energy per multiplication $\bar{E}_{mul2}$ was found to be 58pJ per multiplication and the measured delay of the critical path is 11ns.

The architecture of a 16-bit add-shift multiplier is shown in Figure 2.7. Most energy consumed during its operation is dissipated in the ripple-carry adder and register blocks. Since it takes 16 clock cycles to complete the multiplication, the average energy consumption of a 16 bit add-shift multiplier can be estimated as

$$\bar{E}_{mul1} \cong 16 \cdot (\bar{E}_{rca} + \bar{E}_{reg}) \quad (A.1)$$

Using the values of average energy consumption for a ripple-carry adder and a 16-bit register given in equations (2.13) and (2.8) respectively, the average computational energy of the

add-shift multiplier amounts to 35pJ. Therefore, the speed improvement achieved by using the compressor based multiplier described in this section is paid for by an increase in dissipation of about 70%.

Based on the first order analysis, the compressor based multiplier has smaller *energy · delay* product. Given $\bar{E}_{mul2} = 1.7 \cdot \bar{E}_{mul1}$ and $(delay)_{mul2} = \frac{5}{16} \cdot (delay)_{mul1}$, it follows that

$$(energy \cdot delay)_{mul2} = 0.53 \cdot (energy \cdot delay)_{mul1}.$$

Also, the energy and latency of the compressor based architecture can be reduced when it is applied in a DSP system that does not require the exact product but instead, allows the accuracy of the multiplication result to be within certain tolerance. For example, it was found that if the acceptable result of multiplication of two 16 bit inputs is a 16 bit number whose accuracy is within ± 1 bit, the compression at the lowest 13 bit positions need not be performed at all. Instead, the multiplication can be started from the 14-th bit position, while setting one of the compressor carry inputs to “one” in the first clock cycle. This way, the multiplication is performed in 10 clock cycles instead of 16, achieving the approximate energy savings of 30% (verified in SPICE simulation). A similar simplification is not feasible with the add-shift algorithm.

Although parasitic capacitances involved in the physical implementation are not taken into account in the analysis at the architectural level, it is certain that the described multiplier provides a speed improvement over the solution implemented in the filter design, at the expense of a moderate increase in dissipation. Therefore, it can present a viable alternative for an add-shift multiplier in systems that require higher throughput.

Regarding physical implementation, it should be noted that the area of this multiplier is about twice that of an add-shift multiplier and that its interconnect structure is rather irregular.

APPENDIX C

PROGRAM “DS3”: BEHAVIORAL DESCRIPTION OF THE IMPLEMENTED FILTER

```
#include <stdio.h>

int mac (unsigned short int, unsigned short int, unsigned short int);
int m3 (unsigned short int);

unsigned short int x1, x2;
unsigned short int g;

void main (void)
{
    short int din, mac1, mac2, out, mac2_dly;
    unsigned short int cycle, c1, c2, d1 , d2;
    char cont;

    printf ("\nc1 = ");
    scanf ("%hx", &c1);

    printf ("\nc2 = ");
    scanf ("%hx", &c2);

    printf ("\nd1 = ");
    scanf ("%hx", &d1);

    printf ("\nd2 = ");
    scanf ("%hx", &d2);

    printf ("\ng = ");
    scanf ("%hx", &g);

    printf ("\nx1 = ");
    scanf ("%hx", &x1);

    printf ("\nx2 = ");
    scanf ("%hx", &x2);

    mac2_dly = 0;
    /* x1 = 0;
```

```

x2 = 0; */

do
{
printf ("\nEnter input value: ");
scanf ("%hx", &din);
printf ("\nInput value is %x in hex or %u in decimal\n", din, din);

mac1 = mac(din, c1, c2);

mac2 = mac(mac1, d1, d2);

out = m3 (mac2_dly);

mac2_dly = mac2;

x2 = x1;
x1 = mac1;
printf ("\nout = %x (inverted: %x)", out, out ^ 0xffff);
printf ("\n x1 = %x", x1);
printf ("\n x2 = %x\n", x2);
printf ("\n mac1 = %x", mac1);
printf ("\n mac2 = %x", mac2);
printf ("\n mac2_dly = %x", mac2_dly);

printf ("\nDo you want to continue ? (y/n) \n");
scanf ("%1s", &cont);

}while (cont == 'y');
}

int mac (min, co1, co2)
unsigned short int min, co1, co2;
{
int i;
int acc;
int coc;

acc = 0;
coc = (short int)co1 + (short int) co2;

for (i = 0; i<=15; i++)
{
switch( (((x1 >> i) & 0x1) << 1) | ((x2 >> i) & 0x1) )
{

```

```

case 0:
acc = acc >> 1;
break;

case 1:
acc = (acc >> 1) + (short int)co1;
break;

case 2:
acc = (acc >> 1) + (short int)co2;
break;

case 3:
acc = (acc >> 1) + coc;
}
}
acc = (acc >> 1) + (short int)min;

return (acc >> 1);

}

int m3(min)
unsigned short int min;
{
int i;
int acc;

acc = 0;

for (i=0; i<=15; i++)
{
if ( (g >> i) & 0x1)
acc = (acc >> 1) + (short int)min;
else
acc = acc >> 1;
}
return (acc >> 1);
}

```