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.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Iterative Decoding in Analog VLSI

by

Saied Hemati

A thesis submitted to the Faculty of Graduate Studies and
Research in partial fulfillment of the requirements of the
degree of

Doctor of Philosophy

Ottawa-Carleton Institute For Electrical and Computer Engineering
Department of Systems and Computer Engineering
Faculty of Engineering
Carleton University
Ottawa, Ontario
August 2005

CARLETON UNIVERSITY

DEPARTMENT OF SYSTEMS AND COMPUTER ENGINEERING

The undersigned hereby recommend to the Faculty of Graduate Studies and Research the acceptance of the thesis "**Iterative Decoding in Analog VLSI**" submitted by **Saied Hemati** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Dated: August 5th, 2005

Dr. Rafik A. Goubran
Chair, Department of Systems and Computer Engineering

Dr. Christian Schlegel
External Examiner

Dr. Amir H. Banihashemi
Thesis Co-supervisor

Dr. Calvin Plett
Thesis Co-supervisor

CARLETON UNIVERSITY

Date: **August 2005**

Author: **Saied Hemati**

Title: **Iterative Decoding in Analog VLSI**

Department: **Systems and Computer Engineering**

Degree: **Ph.D.** Convocation: **November** Year: **2005**

Permission is herewith granted to Carleton University to circulate and to have copied for non-commercial purposes, at its discretion, the above title upon the request of individuals or institutions.

_____

Signature of Author

Abstract

This thesis explores theoretical and practical aspects of iterative decoding algorithms when they are implemented on analog continuous-time platforms. Analog continuous-time iterative decoding was proposed a few years ago to improve the power/speed ratio of decoder chips that decode capacity achieving codes. It was commonly believed that replacing discrete-time processing modules with analog circuits would not change the dynamics of the iterative decoder. On the contrary, we show that not only does continuous-time iterative decoding have different dynamics, but also its error correcting performance can surpass that of conventional iterative decoders. We also present a simple model for ideal continuous-time iterative decoding.

As a direct consequence of our study on the dynamics of analog decoders, we show that by looking at the decoding as a numerical problem and using advanced numerical techniques, we are able to improve the convergence rate and decoding performance of iterative decoding algorithms.

Furthermore, we devise novel processing modules for implementing affordable high-speed analog min-sum iterative decoders by using strongly inverted CMOS transistors. This is favorable because previously reported analog decoders were either BiCMOS or weakly inverted CMOS designs. The former could be fast but is rather expensive and the latter would be low-power but it is not fast enough for many applications. Our design is modular and the main blocks are constructed based on current mirrors and virtually any accurate current mirror can be used as the main block in our design. As an example, we show how the building modules can be designed in deep submicron CMOS technologies.

We also present an appropriate design methodology for implementing high-degree blocks that can drastically reduce silicon area and power consumption.

To prove the functionality of the proposed circuits, an analog min-sum iterative decoder chip for a (32,8) regular LDPC code was designed and fabricated in 0.18μm CMOS technology. This chip is the first analog min-sum iterative decoder. Also, it is the first functional analog iterative decoder for an LDPC code and is the fastest reported analog CMOS iterative decoder. Measurement results not only show that the proposed circuits are functional but also confirm the validity of our proposed model for ideal analog decoding. In fact, when noise in the channel is dominant compared to the imperfections in the analog iterative decoder, the simulation results based on our proposed model are close to the measurement results.

Acknowledgement

I would like to express my deepest gratitude to my co-advisors, Professor Amir H. Banihashemi and Professor Calvin Plett. Amir let me experience the freedom and the excitement of personal discovery. He always welcomed new ideas and I benefited very much from his knowledge, support, excellent comments, editorial skills, and passion for doing original research. Calvin's encouragements provided me with an endless source of energy and his help, knowledge, and experience were crucial to improve my thesis work.

I am grateful to Professor R. Mason for his helpful suggestions during the circuit design and the measurements. I am also grateful to Professors C. Schlegel, D. Falconer, I. Marsland, E. Gad, and T. Kwasniewski who were members of my thesis committee. It was an honor for me to benefit from their valuable feedback. My thanks also go to Professor M. El-Tanany for his help in providing measurement equipments.

I would also like to thank the staff of Carleton University. My special thanks go to Ms. D. Hebert, Ms. B. Power, Mr. N. Mehta, and Mr. J. Lemieux for their help.

I can not thank enough my friends and officemates for providing me with a warm and pleasant environment during the cold days in Ottawa. In particular, I am very thankful to Reza Yazdani and Pirouz Zarrinkhat with whom I shared the same apartment and office for more than four years. Reza helped me in simulations and theoretical aspects and Pirouz directly contributed in my thesis by providing me with the parity-check and the generator matrices for the (32,8,10) LDPC code used for designing an analog iterative decoder chip. I also appreciate M. Aghtar, R. Kalbasi, F. Zarkeshvari, A. Noah, H. Xiao, E. Yepez, K. Mosharaf, A. Shokrani, M. Sabbaghian, H. Saeedi, G. Yousefi, S. Khabiri, and A. Jafari and many other nice people at Carleton University for their kindness and

friendship. I wish to acknowledge Mr. Nosa Ogbebor for his help in preparation of the layout of the analog (32,8,10) decoder chip and Mr. James Wang for his help in using IBM 0.13μm CMOS technology for designing a 4-input low-voltage Max WTA chip.

I am indebted to my M.Sc. co-advisors the late Professor A. Adibi and the late Professor V. Tahnai who taught me how to think innovatively. I am also indebted to the late Professor S. Feiz who inspired me for doing research. I always bear in my mind the memory of my deceased high school classmates M. Vakili, M. Taebi, and H. Rashidi and my deceased mathematics teachers Mr. Zandi and Mr. Rasti. I wish I could share this happy moment with them.

My research work was generously supported by grants, awards, and scholarships provided by CITO (Communications and Information Technology Ontario), NSERC (National Science and Engineering Research Council Canada), CMC (Canadian Microelectronics Corporation), OGSST (Ontario Graduate Scholarships in Science and Technology) program, OGS (Ontario Graduate Scholarship) program, and Carleton University. I am grateful for their encouragement and support.

I acknowledge, appreciate, and return the love and support of my mother Gohartaj Sharifi. Her great personality, love, and patience have inspired my whole life and have made many seemingly impossible things possible. I dedicate this dissertation to her. I am obliged to my sisters and brothers Nahid, Nasrin, Hamid, and Vahid for their invaluable support and kindness.

Dedication

To My Mother

Gohartaj Sharifi

Table of Contents

List of Figures

List of Abbreviations and Symbols

Abbreviations

ADC	Analog to Digital Converter
ASTR	Absolute Value and Sign to Real Converter
BER	Bit Error Rate
BiCMOS	Bipolar & CMOS Transistors in the Same Silicon Process
BJT	Bipolar Junction Transistor
BP	Belief Propagation
BPSK	Binary Phase Shift Keying
CHK	Check Node
CMOS	Complementary MOS
DAC	Digital to Analog Converter
DCVSL	Differential Cascode Voltage Switch Logic
DNL	Differential Nonlinearity Error
LDPC	Low-Density Parity-Check
LLR	Log-Likelihood Ratio
LR	Likelihood Ratio
MAP	Maximum a Posteriori
Max WTA	Maximum Winner-Take-All
Min WTA	Minimum Winner-Take-All
ML	Maximum-Likelihood
MOS	Metal-Oxide-Semiconductor
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor

WER	Word Error Rate
MS	Min-Sum Decoding Algorithm
RTAS	Real to Absolute Value and Sign Converter
S/H	Sample and Hold
SP	Sum-Product Decoding Algorithm
SS	Successive Substitution
SR	Successive Relaxation
VLSI	Very Large Scale Integration
VAR	Variable Node
3GPP	3rd Generation Partnership Project

Symbols

$\mathbf{0}$	all zero-vector or matrix depending on context
$\mathbf{1}$	all-one vector
$\boldsymbol{a}$	an event
a	a node in a Tanner graph
a	a digital input signal (based on context)
A	an analog input signal in form of a current
b	a node in a Tanner graph
b	a digital input signal (based on context)
$\boldsymbol{b}$	an event
B	an analog input signal in form of a current
$\boldsymbol{c}$	a codeword (a row vector)
c_i	i-th bit in a codeword (coding context)
c_j	a capacitor (electronic context)
$c_{j'}$	a capacitor (electronic context)
$\hat{c}_i$	estimation of c_i
c	a digital input signal (based on context)
C	an analog input signal in form of a current
C_{in}	input capacitance
$\boldsymbol{C}$	a check node
d	a digital input signal
d_i	a binary variable
d_{min}	minimum distance

dx/dy	derivative of x with respect to y
D	an analog input signal in form of a current
e	number of edges in a Tanner graph
E_b	energy of an information bit
f	variable node operation
f	probability density function
g	check node operation
g_m	transistor's transconductance
$\boldsymbol{G}$	generator matrix
h	a function
$\hbar$	a function
$\boldsymbol{H}$	parity-check matrix
i	an index
$\boldsymbol{I}$	identity matrix
I_0	a fixed current
I_{bias}	biasing current
I	current
I_C	collector current
I_D	drain current
I_{leak}	leakage current
I_{ref}	a reference current
I_S	saturation current
j	an index

k	length of message
l	iteration index
L	transistor length
m	number of inputs in a WTA
$\boldsymbol{m}$	a message word (a row vector)
m_i	i-th bit in a message word
$m_{a\to b}^{(l)}$	a message in LLR domain sent from node a to node b in the l-th iteration
m_V	initial weight of variable node V in LLR domain
$M_V^{(l)}$	output of the variable node V in the l-th iteration
n	length of codeword
n	a number
N	maximum number of iterations
N	a number
N_0	one-sided noise power spectral density
N_C	set of variable nodes incident to the check node C
N_V	set of check nodes incident to the variable node V
p_i	probability of receiving a "0" in the i-th bit of the received word
$P(.)$	probability function
P_{leak}	static power consumption because of leakage current
q_i	probability of receiving a "1" in the i-th bit of the received word
Q	a transistor
r	a resistor
r'	a resistor

r_o	output impedance of a transistor
R	received vector from the channel
t	time
t	transpose
T	processing delay
V	a variable node
V	voltage (electronic context)
V_{bias}	a fixed biasing voltage
V_{dd}	supply voltage
V_{eff}	effective voltage
V_{tt}	a fixed biasing voltage
V_{tt2}	a fixed biasing voltage
V_A	Early voltage
V_{BE}	base-emitter voltage
V_{BS}	bulk-source voltage
V_{CE}	collector-emitter voltage
V_{DS}	drain-source voltage
V_{GS}	gate-source voltage
V_T	threshold voltage
V_{Th}	thermal voltage
x	a binary random variable
X	a column vector representing parity-checks' incoming messages
X, Y	incoming messages

y_i	channel output corresponding to c_i
Y	a column vector representing parity-checks' outgoing messages
W	transistor width
Z	an outgoing message
τ	time constant (scalar or a diagonal matrix based on context)
β	relaxation factor
Δt	a small time change
Λ	a message in LLR domain
λ	a message in LR domain
σ	standard deviation
γ	a normalization factor
κ	a technology-dependent positive parameter

Chapter 1

Introduction

In this chapter, main objectives and contributions of this thesis are explained in brief. This chapter is started by looking at the early days of coding theory and microelectronics. This is to show the importance of implementing iterative decoding algorithms in analog VLSI, which is a new and rather non-conventional application for analog circuits and at the same time is an obvious deviation from conventional discrete-time iterative decoding. We notice how implementation complexity changed the fate of coding and delayed its progress. We also explain how complexity issues could further postpone the deployment of the best known codes in many applications and how analog iterative decoding could be helpful in mitigating the implementation complexity.

1.1 Background and Historical Perspective

In July 1948, researchers at Bell Labs unveiled the birth of a twin that soon changed the face of the world. The invention of the first transistor [1] and Shannon's mathematical theory of communication [2] were both made public for the first time in July 1948. During the coming years the importance of these two scientific achievements

have been realized and their impacts have been so great that they considerably changed our life style and we still expect more to come. Among all research work that was carried out in the 20th century, these two bodies of work were so amazing that we can put their names on this century and call it the era of microelectronics or the information technology age.

Modern information theory, the foundation of which was laid by Shannon's epoch making paper [2], revolutionized communication and storage systems. Shannon formalized the concept of information and proved that for an unreliable channel, there exists a number, called the *capacity* of the channel, such that reliable data transmission is only possible for rates not greater than this number. In the same paper, Shannon introduced the concept of codes as ensembles of vectors that are to be transmitted. He also proved the existence of capacity achieving codes. However, he did not show explicitly how these codes can be designed, encoded, and decoded and the ongoing endeavor for designing good codes with efficient encoding and decoding algorithms has kept researchers in the coding community busy from then on. In 1962, Gallager invented low-density parity-check (LDPC) codes in his thesis, along with an elegant iterative decoding scheme [3]. Despite being promising, soon after their invention, LDPC codes were largely forgotten mainly because the proposed decoding algorithm was too complex for the computational resources that were available at that time. Thirty four years later when these codes were reinvented by MacKay and Neal [4] thanks to microelectronics, information theory's twin sister, digital computers were so powerful that it became easy to use simulation to verify the error correcting performance of these codes. In fact, by means of computer simulations it was confirmed that LDPC codes and turbo codes decoded by iterative decoding

show record-breaking error correcting performance and can practically achieve the Shannon limit [5]-[6].

The invention of the transistors revolutionized the computation and signal processing. These small inexpensive low-power active devices soon replaced vacuum tubes in many applications and as Moore predicted in his well-known paper [7] the advantages of integration and miniaturizing the circuits brought about a proliferation of electronics. Interest in digital circuits was dramatically increased by the introduction of very large scale integration (VLSI) systems. In digital circuits a transistor could be simply modeled by a switch that is either open or closed and designers are less concerned about the dynamics of the transistors. This makes the design process considerably simple. Digital circuits could be very robust and their performance would not deteriorate easily by common fabrication-induced problems such as transistor mismatch. However, fabrication imperfections can lead to serious ambiguities about the real geometrical and physical characteristics of the transistors inside a chip.

Thanks to the affordability of digital VLSI chips and the maturity of the digital signal processing algorithms and numerical techniques, the realm of computations has long ago been conquered by digital computers, and analog computers have been abandoned. Analog computers were used to solve computationally demanding problems such as complex nonlinear time-variant partial differential equations by directly mapping the problem onto a circuit and observing its time response [8]. In the 1980s many experts predicted the demise of analog circuits, and later many researchers believed that the analog circuit realm would ultimately become limited to analog-to-digital converters (ADC)

and digital-to-analog converters (DAC) that translate naturally occurring analog signals into bits and vice versa [9].

Currently digital circuit design is quite prosperous in terms of available expertise and tools. Virtually any complex algorithm with arbitrarily high degree of accuracy can be implemented by digital circuits and in the future, digital circuits will become more and more prevalent. However, this statement does not imply that for any signal processing application, a digital approach can come up with an efficient solution. When a massively parallel algorithm, such as an artificial neural network, is to be implemented, the digital approach becomes inefficient. In an artificial neural network numerous processing units with medium accuracy are required and these processing units are highly interconnected. For these types of algorithms digital implementation could be prohibitively costly in terms of power and silicon area consumption. While quite a few power-friendly and area-friendly approaches are known in a digital approach, these improvements are often achieved at the expense of speed and ultimately they fail to improve the speed/power ratio significantly [10].

Research works on biological neural networks revealed that the human brain, which is the most wonderful and powerful known processor, is composed of relatively simple neurons that are not very accurate but are massively interconnected. This observation motivated researchers to devise bio-inspired neural networks and it was soon found that implementation of neural networks provides a unique opportunity for utilizing analog processing modules that, similar to neurons, are not that accurate but could be more efficient in terms of power, speed, and area consumption. Mead's pioneering work on analog retina and cochlea proved the functionality of these bio-inspired analog VLSI processors

[11]. Mead, a leading researcher in semiconductor physics and design of analog and digital circuits, proposed the idea of focusing on strategies which exploit the characteristics of transistor physics directly for computation. His motto was "MOSFET nonlinearities are a feature to be celebrated and utilized, rather than lamented" [11]. While not all of Mead's dreams for analog neural networks have yet come true [12], many amazing analog circuits have been designed based on his idea and analog neural network chips are used in many applications [13]. More recently, it was shown that analog computing supersedes digital computing in computational power [14]. Furthermore, Mead's approach had a great influence on the design of analog iterative decoders [15] that are studied throughout this thesis.

1.2 Thesis Objectives

At the present time channel coding is an indispensable part of any communication system and is used to improve the reliability of data transmission or to provide better power and bandwidth efficiency in a wide range of applications from chip-to-chip communications on a small printed circuit board [16] to deep space missions [17]. It is therefore of great interest to efficiently implement the capacity achieving codes and tailor suitable encoders and decoders for these wide range of applications. Implementing decoders is quite challenging as it is associated with observing the received message at the output of the channel, which is the noise-corrupted version of the transmitted message and then making a decision in favor of the most likely transmitted message among a huge number of possible choices.

Iterative decoding algorithms are used for decoding LDPC codes and turbo codes that are the best known coding schemes. Though these decoding algorithms are subopti-

mal, for long codes their performance is very close to the optimum decoding. In iterative decoding, numerous processing nodes are required and these nodes iteratively exchange information with each other during the decoding process. Conventional digital approach can be used for implementing these decoders, but similar to the artificial neural network case, power and silicon area consumption could be too high for many applications. As an example, in [18] a digital iterative decoder chip is presented that consumes 0.7W and its die area is 52.5 mm^2 and consequently is not suitable for many applications. Furthermore, the digital approach could generate a lot of switching noise that can be harmful for noise sensitive circuits such as low noise amplifiers.

While analog iterative decoders could consume less power and silicon area and generate lower noise and have a better speed/power ratio, no one has provided any theoretical evidence that iterative decoding that is defined and expressed in discrete-time domain remains functional when it is implemented on a continuous-time platform. In this thesis, we attempt to formulate this problem and investigate the dynamics of continuous-time iterative decoding and compare its performance with the conventional discrete-time iterative decoding.

We also noted that none of the previously reported analog iterative decoders use favorable standard CMOS technology in the strong inversion biasing condition. Either they use non-standard technologies such as BiCMOS technology [19]-[24] or they use favorable CMOS technology but not in strong inversion biasing condition [25]-[29]. While the former approach has the advantage of higher speed, it could increase the fabrication cost. The latter approach has the advantage of extremely low power consumption but it is more suitable for low speed applications. In this thesis, we present our proposed

CMOS circuits in strong inversion biasing condition that are suitable for high-speed low-cost applications [30]-[33].

1.3 Thesis Overview

The present thesis is divided into eight chapters. In this chapter we gave some introductory comments on the history and trend of microelectronics and coding. We also explained our motivation for implementing and analyzing the performance of analog iterative decoders. In Chapter 2, we present a short review of coding theory and introduce iterative decoding algorithms and LDPC codes, which are our main focus throughout this thesis. In Chapter 3 the dynamics of continuous-time iterative decoding is investigated and it will be shown that analog decoding can supersede conventional iterative decoding in performance. In Chapter 4, after a short review of the methods that use BiCMOS and weakly inverted CMOS for designing analog iterative decoders, we present a design methodology based on strongly inverted CMOS circuits for implementing min-sum analog iterative decoders. In Chapter 5, we present efficient ways for implementing processing nodes with high degrees. Chapter 6 is devoted to implementation issues of the proposed analog decoder when more advanced fabrication technologies with reduced power supply voltages are used. In Chapter 7, we present the measurement results for a proof-of-concept analog min-sum iterative decoder chip for a (32,8) regular LDPC code. We conclude this thesis by a summary of the achieved goals and possible future works.

1.4 Thesis Contributions

- A novel analysis for the dynamics of analog iterative decoding is presented.

- It is shown that ideal analog continuous-time iterative decoding is superior in performance to conventional discrete-time iterative decoding.
- By proposing simple modifications, the error correcting performance of the best known iterative decoding algorithms are improved.
- It is shown that many proposed variants of iterative decoding algorithms have better decoding performance because they apply more appropriate numerical techniques to the fixed-point problem of iterative decoding.
- A modular design methodology is proposed for designing affordable high-speed CMOS analog min-sum iterative decoders.
- A proof-of-concept analog min-sum iterative decoder chip for a (32,8,10) regular LDPC code is designed, fabricated, and tested. It is the fastest reported CMOS analog iterative decoder and it is the first reported analog min-sum iterative decoder chip. Also, it is the first analog decoder chip for an LDPC code.
- Novel circuits for implementing decoders with high-degree nodes are proposed.
- A general approach for converting current mirrors into current-mode maximum winner-take-all circuits is presented. As an example a novel low-voltage CMOS maximum winner-take-all circuit is designed.

Chapter 2

Iterative Decoding and LDPC Codes

In this chapter the basic concepts of coding theory are introduced. This short introduction provides necessary theoretical background for following the next chapters. Brevity and simplicity of the discussions are our main concern here and therefore, discussions will be more conceptual rather than mathematical. There are quite a few good textbooks on coding theory that can be referred to for further information on the theoretical aspects, see for instance [34].

2.1 A Glimpse at Coding

One of the main goals of communication and storage systems is to duplicate a given message at another venue or time. In a digital system, the message consists of a number of symbols that are selected from an alphabet. For binary systems which are of concern here, "0" and "1" are the only elements of the alphabet. Unfortunately, noise can degrade the performance of communication and storage systems and it is likely that some of the symbols are estimated in error. If the symbols are independent, no error in the retrieved message can be detected.

An efficient source of information tries to remove any existing correlation among the symbols forming the message in order to minimize the number of symbols. This process is called *source coding* or *data compression.* However, in order to detect and correct errors in the retrieved message from a noisy channel, symbols should be related to each other and preferably in a simple way that can be tested easily. This would pave the way for error detection and ultimately would increase the reliability of the system. Therefore, the symbols are made correlated in an intentional and controlled manner by adding redundancy to the original message. This process is called *channel coding.*

Figure 2-1 illustrates how in a simple communication system that utilizes channel coding, an encoder at the transmitter side adds redundancy to the original message and increases the number of bits from 4 to 8. These introduced bits, called *parity bits*, help the decoder to detect and correct an error that has occurred. In this example if we represent the original message by row vector $\boldsymbol{m}$ and represent the output of the encoder by row vector $\boldsymbol{c}$ which is called a *codeword*, then we can write:

$$\boldsymbol{m} = [m_1 \quad m_2 \quad m_3 \quad m_4],$$

$$\boldsymbol{c} = [c_1 \quad c_2 \quad c_3 \quad c_4 \quad c_5 \quad c_6 \quad c_7 \quad c_8].$$

The relationship between $\boldsymbol{m}$ and $\boldsymbol{c}$ can be defined by the following equations:

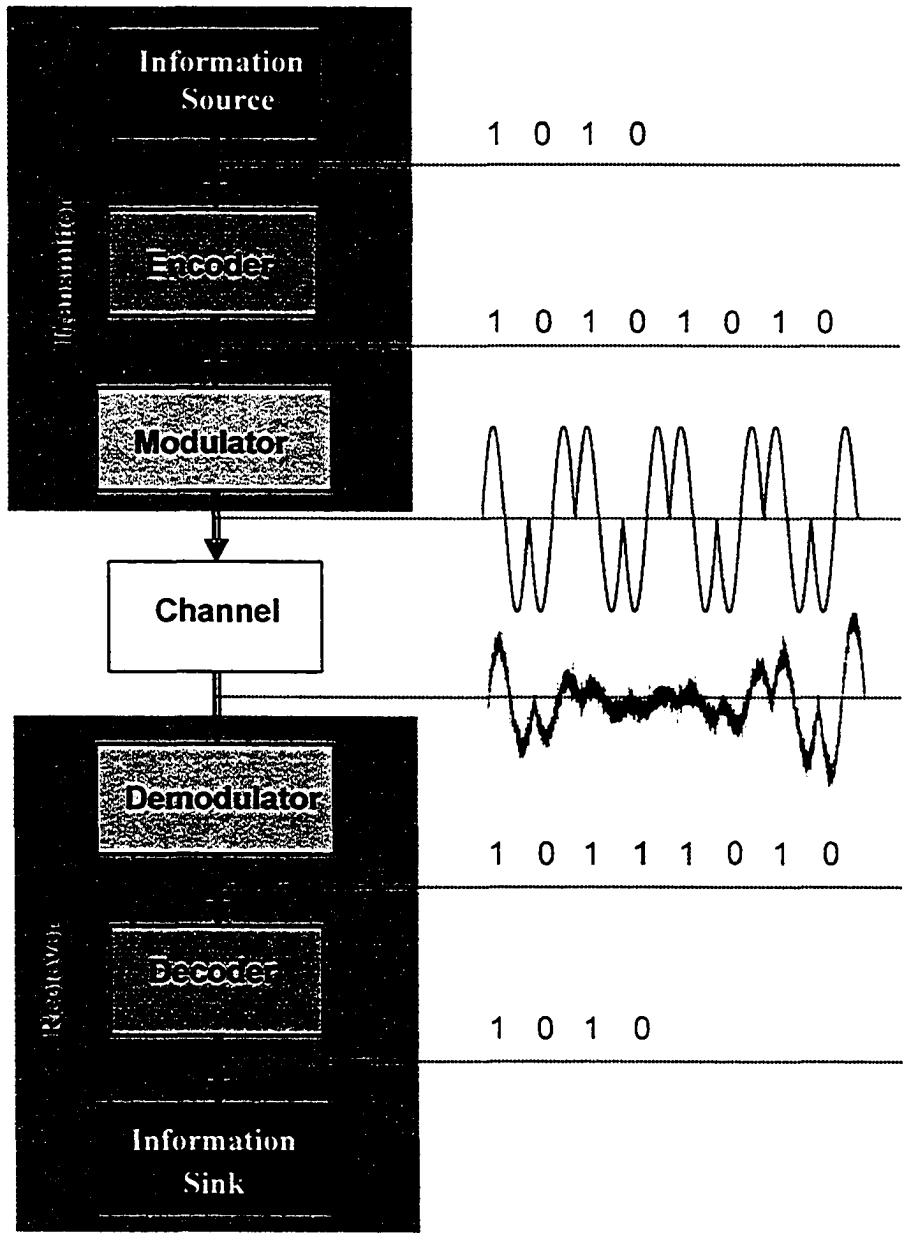


Figure 2-1 A simple communication system that uses channel coding.

$$\begin{cases} c_1 = m_1 \\ c_2 = m_2 \\ c_3 = m_3 \\ c_4 = m_4 \\ c_5 = m_1 + m_2 + m_4 \\ c_6 = m_1 + m_3 + m_4 \\ c_7 = m_2 + m_3 + m_4 \\ c_8 = m_1 + m_2 + m_3 \end{cases} \tag{2-1}$$

where "+" represents the binary addition that is equivalent to an XOR operation. It is more convenient to write (2-1) in matrix form as follows:

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}. \tag{2-2}$$

In (2-2), $\boldsymbol{G}$ is a matrix and is called *generator matrix* and maps any message word onto a codeword. It is simple to verify that the generator matrix in our example is given by:

$$\boldsymbol{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

This example shows a simple linear block code that assigns an 8-bit codeword to each 4-bit message. In coding theory it is common to refer to this code as an (8,4) linear block code. Because the original message directly appears in the codeword, this code is said to be in *systematic* form. It is worth noting that in this example, only 4/8 of bits of each codeword are useful for carrying information and since we are transmitting the parity-bits, the actual (information) transmission rate would be 4/8 of the coded scheme; this ratio is called the *rate* of the code. In fact, we could have 2^8 distinct cases for an 8-bit

word, but we only have 2^4 defined codewords, this is because there is a one-to-one relationship between each message and its corresponding codeword as is clear from (2-2).

It is easy to show that the following equations relate bits in the codewords that are defined based on (2-1).

$$\begin{cases} c_5 = c_1 + c_2 + c_4 \\ c_6 = c_1 + c_3 + c_4 \\ c_7 = c_2 + c_3 + c_4 \\ c_8 = c_1 + c_2 + c_3 \end{cases} \tag{2-3}$$

or equivalently:

$$\begin{cases} c_1 + c_2 + c_4 + c_5 = 0 \\ c_1 + c_3 + c_4 + c_6 = 0 \\ c_2 + c_3 + c_4 + c_7 = 0 \\ c_1 + c_2 + c_3 + c_8 = 0 \end{cases} \tag{2-4}$$

This set of equations is called *parity-check equations* and any codeword satisfies these equations. This is the criterion that is often used at the receiver to determine if the received word is a codeword or not. Equation (2-4) can be expressed in matrix form as follows:

$$\boldsymbol{H}.\boldsymbol{c}^{\mathrm{t}} = \boldsymbol{0},$$

or

$$\boldsymbol{c}.\boldsymbol{H}^{\mathrm{t}} = \boldsymbol{0}. \tag{2-5}$$

Here $\boldsymbol{H}$ is called the *parity-check matrix* and for the example in hand it is equal to:

$$\boldsymbol{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The parity-check matrix completely defines a linear block code. Each row in $\boldsymbol{H}$ shows which bits are checked by a given parity-check equation and each column shows in which parity-check equations, a given bit is involved. Thus, parity-check equations show, on the one hand, how a given bit carries information about other bits and could have an influence on them and on the other hand, which other bits carry information about that specific bit. Iterative decoding that is introduced in the next section is best explained based on this simple observation.

The number of nonzero elements of a codeword is called the *Hamming weight* of that codeword. The *Hamming distance* between two codewords is defined as the number of locations in which the two codewords differ. The *minimum distance* of a code is defined as the smallest Hamming distance between two different codewords in the code. For our simple block code, Hamming distance is equal to 4.

A linear block code can be graphically represented by a *Tanner graph* [35] that is a bipartite graph in which one set of nodes, the *variable nodes* (symbol nodes, bit nodes), corresponds to the code information symbols and the other set of nodes, the *check nodes*, corresponds to the set of parity-check equations. An edge connects the j-th variable node and the i-th check node if a "1" is located at position (i, j) of $\boldsymbol{H}$. Figure 2-2 shows the Tanner graph of our simple block code. The Tanner graph is not the only way that a code can be represented and there are also other ways for graphically representing codes, see for example [36].

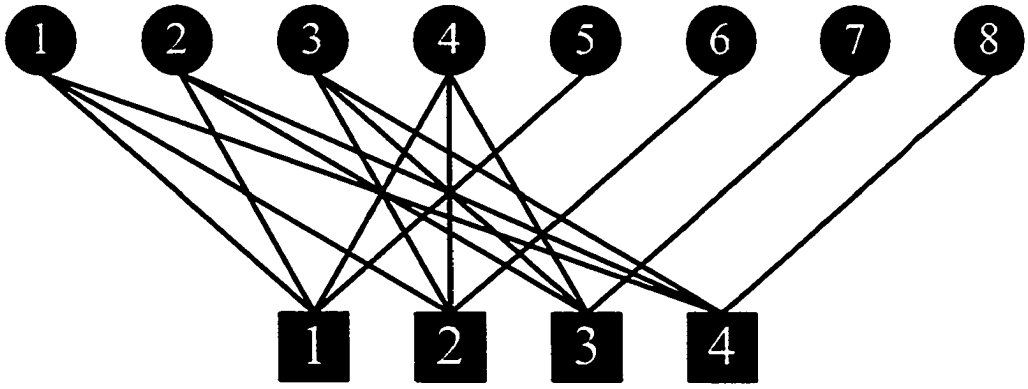


Figure 2-2 Tanner graph of an (8,4) linear block code. Variable nodes and check nodes are shown by circles and squares, respectively.

An LDPC code [3] is a linear block code that has a sparse parity-check matrix. LDPC codes are divided into two basic categories; regular LDPC codes and irregular LDPC codes. In regular LDPC codes, the parity-check matrix contains a small fixed number of ones in each column and another small fixed number of ones in each row. In contrast, irregular LDPC codes have different number of ones in their columns or rows. LDPC codes are among the best known channel codes and as explained earlier in the first chapter, these codes were almost forgotten for more than three decades because their excellent performance was not realized due to limited available computational resources.

2.2 A Glimpse at Decoding

Decoding is a complex decision-making problem. As an example, for a (1008,504) code, the decoder should choose the most likely transmitted codeword from among 2^{504} possible codewords, this shows how complex the decoding process could be. A decoder based on the observed word at the channel's output and by prior knowledge about the coding scheme and probably some information about statistical behavior of the

information source and noise in the channel, tries to estimate the transmitted codeword. The optimum decision-making rule is based on choosing the codeword with the largest a posteriori probability and is called *maximum a posteriori probability (MAP) decision rule*. This means that after the received word has been observed, the most probable codeword is chosen. A decoder that utilizes the MAP decision rule can minimize the probability of error and to do so it needs to know the a priori probability of each codeword. If codewords are equally likely to be transmitted, it is possible to simplify the MAP decision rule and instead of maximizing the a posteriori probability, the decoder can maximize the likelihood function. This simplified decision-making rule is called *maximum likelihood (ML) decision rule*.

In a digital communication system, since noise in the channel deteriorates the transmitted signal, the channel output is no longer in digital form and usually takes real values. A simple approach is to compare the demodulated received signal with a threshold and decide in favor of one or zero. In this method the decoder receives the channel output after the decision is made. This approach does not distinguish between a strong signal and a weak marginal signal as long as they are both on the same side of the threshold. This method is called *hard-decision decoding*. Hard-decision decoding is used in the case for the decoder shown in Figure 2-1. Hard-decision decoding simplifies the decoding by dealing with bits instead of real numbers but it is at the expense of discarding some important information about the received signal and ultimately this reduction in information would increase the error rate. In hard-decision decoding a code with minimum distance d_{min} is $(d_{min}-1)$-error-detecting and $[(d_{min}-1)/2]$-error-correcting. For example, a

code with d_{min} = 10 can detect all error patterns of Hamming weights up to 9, and can correct all error patterns of Hamming weights not exceeding 4.

The case when decision-making is exclusively left to the decoder is called *soft-decision decoding*. This means that the channel output is not compared with a threshold and the decoder instead of a bit, receives a real number that indicates, implicitly or explicitly, the probability of being a zero or a one. In this thesis we only deal with soft-decision decoding.

It is often intractable to apply optimal decoding for practical applications and its complexity is unaffordable [37]. In the next section iterative decoding algorithms, which are affordable but suboptimal are introduced. The complexity of iterative decoders increases almost linearly by the block length. Turbo codes and LDPC codes can approach the channel capacity under these decoding schemes.

2.2.1 Fundamentals of Iterative Decoding

As explained earlier in this chapter, the encoder adds redundancy to the transmitted word and makes the bits in a codeword correlated. Parity-check equations show how bits are related to each other in a codeword. The decoder knows the relationships among the transmitted bits and tries to detect and possibly correct any errors in the received word. In order to see how iterative decoding works, consider the (8,4) block code that was introduced earlier by (2-1) and was graphically represented by the Tanner graph in Figure 2-2. Assume p_i and q_i are defined as:

$$\begin{aligned} P(c_i = 0 \mid y_i) &= p_i, \\ P(c_i = 1 \mid y_i) &= 1 - p_i = q_i. \end{aligned} \tag{2-6}$$

Where c_i is the i-th bit of the codeword, y_i is the channel output corresponding to c_i, and $P(\text{“}\boldsymbol{a}\text{”} \mid \text{“}\boldsymbol{b}\text{”})$ is defined as the conditional probability of event “***a***” given that event “***b***” has been observed.

Then recalling (2-3), c_5 is related to c_1, c_2, and c_4 through the following equation:

$$c_5 = c_1 + c_2 + c_4 . \tag{2-7}$$

Therefore, to make a decision about c_5 only based on prior information about y_1, y_2, and y_4, the following equation can be written:

$$\begin{aligned} P(c_5 = 0 \mid y_1, y_2, y_4) = {} & P(c_1 = 0, c_2 = 0, c_4 = 0 \mid y_1, y_2, y_4) + \\ & P(c_1 = 1, c_2 = 1, c_4 = 0 \mid y_1, y_2, y_4) + \\ & P(c_1 = 1, c_2 = 0, c_4 = 1 \mid y_1, y_2, y_4) + \\ & P(c_1 = 0, c_2 = 1, c_4 = 1 \mid y_1, y_2, y_4). \end{aligned}$$

The process of estimating c_5 based on the information about y_1, y_2, and y_4, is illustrated in Figure 2-3 which is a simplified version of the Tanner graph of Figure 2-2.

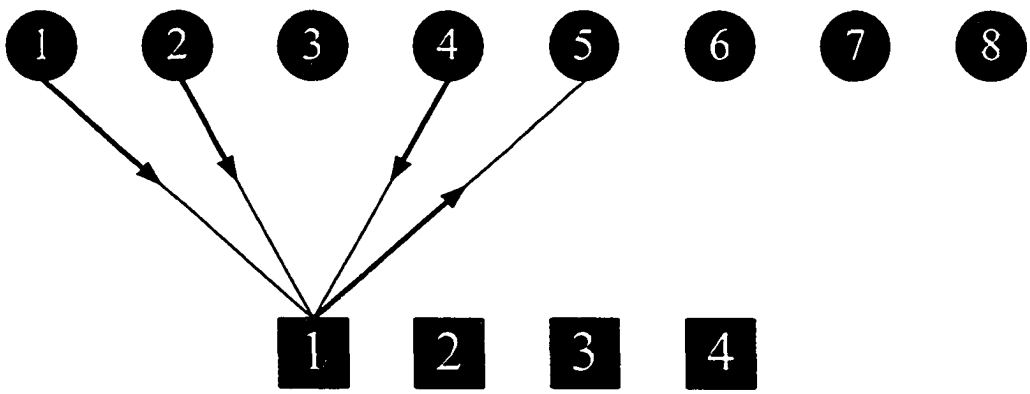


Figure 2-3 A simplified version of the Tanner graph of Figure 2-2, shows how messages are passed in iterative decoding.

$$P(c_5=1\mid y_1,y_2,y_4)=P(c_1=1,c_2=0,c_4=0\mid y_1,y_2,y_4)+$$
$$P(c_1=0,c_2=1,c_4=0\mid y_1,y_2,y_4)+$$
$$P(c_1=0,c_2=0,c_4=1\mid y_1,y_2,y_4)+$$
$$P(c_1=1,c_2=1,c_4=1\mid y_1,y_2,y_4).$$

By assuming independency we get:

$$\begin{cases}P(c_5=0\mid y_1,y_2,y_4)=p_1p_2p_4+q_1q_2p_4+q_1p_2q_4+p_1q_2q_4\\ P(c_5=1\mid y_1,y_2,y_4)=q_1p_2p_4+p_1q_2p_4+p_1p_2q_4+q_1q_2q_4\end{cases}. \tag{2-8}$$

Or equivalently:

$$\left(P(c_5=0\mid y_1,y_2,y_4)-P(c_5=1\mid y_1,y_2,y_4)\right)=(p_1-q_1)(p_2-q_2)(p_4-q_4) \tag{2-9}$$

However, y_5 also has some information about c_5 and to get the best estimation about c_5 it should also be taken into account.

$$\begin{cases}P(c_5=0\mid y_1,y_2,y_4,y_5)=P(c_5=0\mid y_5,\{y_1,y_2,y_4\})\\ P(c_5=1\mid y_1,y_2,y_4,y_5)=P(c_5=1\mid y_5,\{y_1,y_2,y_4\})\end{cases}$$

By applying Bayes rule, and representing the probability density function by f, we have:

$$P(c_5=0\mid y_5,\{y_1,y_2,y_4\})=\frac{f(c_5=0,y_5,\{y_1,y_2,y_4\})}{f(y_5,\{y_1,y_2,y_4\})}$$
$$=\frac{f(y_5\mid c_5=0,\{y_1,y_2,y_4\})f(c_5=0,\{y_1,y_2,y_4\})}{f(y_5\mid\{y_1,y_2,y_4\})f(\{y_1,y_2,y_4\})}$$

However, for a memoryless channel it is clear that if we know the transmitted bit (c_5), the corresponding received signal (y_5) will only depend on channel noise and therefore we have:

$$f(y_5\mid c_5=0,\{y_1,y_2,y_4\})=f(y_5\mid c_5=0)$$

and consequently:

$$P(c_5=0\mid y_1,y_2,y_4,y_5)=\frac{f(y_5\mid c_5=0)P(c_5=0\mid\{y_1,y_2,y_4\})}{f(y_5\mid\{y_1,y_2,y_4\})}$$
$$=\frac{f(y_5,c_5=0)P(c_5=0\mid\{y_1,y_2,y_4\})}{P(c_5=0)f(y_5\mid\{y_1,y_2,y_4\})}$$
$$=\frac{P(c_5=0\mid y_5)P(c_5=0\mid\{y_1,y_2,y_4\})f(y_5)}{P(c_5=0)f(y_5\mid\{y_1,y_2,y_4\})}$$
$$=\gamma P(c_5=0\mid y_5)P(c_5=0\mid y_1,y_2,y_4).$$

Where γ is a constant value. Similarly we can write:

$$P(c_5=1\mid y_1,y_2,y_4,y_5)=\gamma P(c_5=1\mid y_5)P(c_5=1\mid y_1,y_2,y_4),$$

and thus

$$\begin{cases}P(c_5=0\mid y_1,y_2,y_4,y_5)=\gamma\left(p_5.P(c_5=0\mid y_1,y_2,y_4)\right)\\ P(c_5=1\mid y_1,y_2,y_4,y_5)=\gamma\left(q_5.P(c_5=1\mid y_1,y_2,y_4)\right)\end{cases},\qquad(2\text{-}10)$$

or equivalently:

$$\ln\left(\frac{P(c_5=0\mid y_1,y_2,y_4,y_5)}{P(c_5=1\mid y_1,y_2,y_4,y_5)}\right)=\ln\left(\frac{p_5}{q_5}\right)+\ln\left(\frac{P(c_5=0\mid y_1,y_2,y_4)}{P(c_5=1\mid y_1,y_2,y_4)}\right).\qquad(2\text{-}11)$$

Equations (2-9) and (2-11) show how by taking y_1, y_2, y_4, and y_5 into consideration, we can obtain an estimation about c_5 (let us call it $\hat{c}_5$). However, if we use all the channel outputs and do not restrict ourselves to y_1, y_2, y_4, and y_5 then $\hat{c}_5$ can be best estimated. This means that to get a better estimation about c_5, we need better $\hat{c}_1$, $\hat{c}_2$, and $\hat{c}_4$ (estimations of c_1, c_2, and c_4), that comes from the rest of parity-check equations. To keep the independency assumption valid, when we are concerned about $\hat{c}_5$, we do not use (2-7) for updating $\hat{c}_1$, $\hat{c}_2$, and $\hat{c}_4$. Unfortunately, the assumption of independency is violated anyway for this code, because $\hat{c}_1$, $\hat{c}_2$, and $\hat{c}_4$ become dependent on each other through other parity-check equations. This can be simply verified based on the presence of closed

loops of edges (cycles) in the graphical representation of the code in Figure 2-2. There are quite a few cycles in Figure 2-2 and Figure 2-4 shows a cycle of length six.

If the graphical representation of the code was cycle-free, then it would be possible to maximize the information that we can get about each bit of the transmitted codeword (c_1, c_2, ..., c_8), and our approach would be optimal. However, because of the cycles this approach is suboptimum.

This simple example shows the fundamental operations in a powerful decoding method that is based on passing beliefs through the graph and is often called *belief propagation* algorithm. Later in this section we formally introduce this decoding algorithm. Belief propagation algorithm is also known as "sum-product decoding algorithm" because of the multiplication and summation operations used in (2-8) and (2-10). This algorithm was originally proposed by Gallager [3], and was further developed by Tanner [35]. It has also been invented and used independently in the artificial intelligence community [38], and was studied in details in [39] and its optimality when applied to realizations with cycle-free Tanner graphs was shown. It is possible to slightly modify belief propagation to improve its performance on graphs with cycles [40].

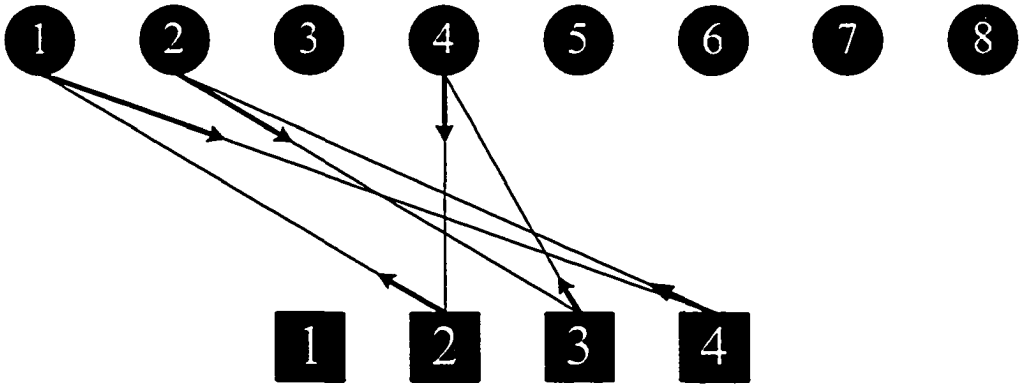


Figure 2-4 A closed loop of six edges, which is called a cycle of length six.

In its conventional expression, belief propagation is performed in the discrete-time domain. As we will see later, one of the contributions of this thesis is in changing this common belief and showing that not only is the discrete-time approach not essential, but it could also degrade the error correcting performance of the decoder for the graphs with cycles.

In its discrete-time expression, in the l-th round of the belief propagation decoding algorithm (l>0), the outgoing messages of the variable node set V ($p_{V\to C}^{(l)}$ and $q_{V\to C}^{(l)}$) and the outgoing messages of the check node set C ($p_{C\to V}^{(l)}$ and $q_{C\to V}^{(l)}$) are computed based on (2-12) and (2-13), respectively. Equations (2-12) and (2-13) can be shown to be generalized form of (2-11) and (2-9). For the variable node set V, p_V and q_V are defined similar to (2-6). $p_{V\to C}^{(l)}$, $q_{V\to C}^{(l)}$, $p_{C\to V}^{(l)}$, and $q_{C\to V}^{(l)}$ similarly represent outgoing messages in the probability domain.

$$\ln\left(\frac{p_{V\to C}^{(l)}}{q_{V\to C}^{(l)}}\right)=\ln\left(\frac{p_V}{q_V}\right)+\sum_{C'\in N_V\setminus\{C\}}\ln\left(\frac{p_{C'\to V}^{(l-1)}}{q_{C'\to V}^{(l-1)}}\right), \tag{2-12}$$

$$\left(p_{C\to V}^{(l)}-q_{C\to V}^{(l)}\right)=\prod_{V'\in N_C\setminus\{V\}}\left(p_{V'\to C}^{(l)}-q_{V'\to C}^{(l)}\right), \tag{2-13}$$

where N_C is the set of variable nodes incident to the check node set C, and N_V is the set of check nodes incident to the variable node set V. It is also assumed:

$$p_{C\to V}^{(0)}=q_{C\to V}^{(0)}=\frac{1}{2} \tag{2-14}$$

It is possible to significantly simplify (2-13) by noting that:

$$\left|p_{V'\to C}^{(l)} - q_{V'\to C}^{(l)}\right| \le 1$$

Therefore:

$$\prod_{V'\in N_C\setminus\{V\}}\left|p_{V'\to C}^{(l)} - q_{V'\to C}^{(l)}\right| \le \min_{V'\in N_C\setminus\{V\}}\left(\left|p_{V'\to C}^{(l)} - q_{V'\to C}^{(l)}\right|\right). \tag{2-15}$$

Then the operation in the check nodes can be approximately expressed by:

$$\left(p_{C\to V}^{(l)} - q_{C\to V}^{(l)}\right) = \left(\prod_{V'\in N_C\setminus\{V\}} sign\left(p_{V'\to C}^{(l)} - q_{V'\to C}^{(l)}\right)\right) \min_{V'\in N_C\setminus\{V\}}\left(\left|p_{V'\to C}^{(l)} - q_{V'\to C}^{(l)}\right|\right), \tag{2-16}$$

where the *sign* function is defined as follows:

$$sign(x) = \begin{cases} 1 & x = 0 \\ \dfrac{x}{|x|} & x \neq 0 \end{cases} \tag{2-17}$$

(2-16) can be further simplified as follows:

$$\ln\left(\frac{p_{C\to V}^{(l)}}{q_{C\to V}^{(l)}}\right) = \left(\prod_{V'\in N_C\setminus\{V\}} sign\left(\ln\left(\frac{p_{V'\to C}^{(l)}}{q_{V'\to C}^{(l)}}\right)\right)\right) \min_{V'\in N_C\setminus\{V\}}\left(\left|\ln\left(\frac{p_{V'\to C}^{(l)}}{q_{V'\to C}^{(l)}}\right)\right|\right), \tag{2-18}$$

This is because when $(p-q)\in(-1,1)$, we have:

$$sign\left(\ln\left(\frac{p}{q}\right)\right) = sign(p-q),$$

and $\ln\left(\frac{p}{q}\right)$ is a monotonically increasing function of $(p-q)$ and despite using different forms for representing the messages, the min function selects the same variable node in

(2-16) and (2-18). This can be simply verified by finding the derivative of $\ln\left(\frac{p}{q}\right)$ with respect to $(p-q)$ as follows:

$$\frac{d\left(\ln\left(\frac{p}{q}\right)\right)}{d(p-q)} = \frac{2}{1-(p-q)^2} > 0.$$

For a binary random variable "x", we define log-likelihood ratio of "x" as:

$$\Lambda(x) = \ln\left(\frac{P(x=0)}{P(x=1)}\right).$$

Then we can rewrite (2-12) and (2-18) as:

$$m_{V\to C}^{(l)} = m_V + \sum_{C'\in N_V\setminus\{C\}} m_{C'\to V}^{(l-1)}, \tag{2-19}$$

$$m_{C\to V}^{(l)} = \left(\prod_{V'\in N_C\setminus\{V\}} sign\left(m_{V'\to C}^{(l)}\right)\right) \min_{V'\in N_C\setminus\{V\}} \left(\left|m_{V'\to C}^{(l)}\right|\right). \tag{2-20}$$

Where $m_{a\to b}^{(l)}$ is the message, in the log-likelihood ratio domain, that node "a" sends to node "b" in the l-th round of the algorithm ($l>0$). Also, m_V is the initial weight of the variable node V in log-likelihood ratio domain that is computed based on p_V and q_V.

An iterative decoding algorithm that implements (2-19) for the variable nodes and implements (2-20) for the check nodes, is called *min-sum* decoding algorithm [39]. It is observed that when the variable nodes and the check nodes communicate with each other in the log-likelihood ratio domain, the only operations are addition and finding the minimum. As we will see later, multiplication of signs in (2-20) can also be regarded as a bi-

nary addition of sign bits and implemented by simple binary adders (XOR gates). Other common names for min-sum are "max-sum", "max-product", and "max-log-map" [41].

In the min-sum decoding algorithm, the check nodes overestimate the messages, as shown in (2-15). This could degrade the performance of min-sum iterative decoding by a few tenths of a dB. However, there are simple modifications that can be applied to this decoding algorithm in order to improve its error correcting performance [41]-[43].

In belief-propagation and min-sum decoding algorithms, the output of each variable node in the l-th iteration is calculated by the following equation:

$$M_V^{(l)} = m_V + \sum_{C' \in N_V} m_{C' \to V}^{(l-1)} . \tag{2-21}$$

$M_V^{(l)}$ represents the soft estimation of the transmitted bit corresponding to the variable node V. The decoding process is stopped, as soon as the decoder converges to a codeword, that is, when hard-decision assignment for the outputs of the variable nodes satisfies all the parity-check equations. If the decoder fails to converge to a codeword, the decoding process would continue up to a given number of iterations.

2.3 Conclusion

In this chapter, very basic concepts of coding and iterative decoding were reviewed. In particular channel and source coding, block codes, parity-check and generator matrices, Hamming distance, Hamming weight, minimum distance, hard and soft decoding, regular and irregular LDPC codes were introduced. The theory behind iterative decoding was conceptually explained and the basic operations in belief propagation and min-sum decoding algorithms were derived through a simple example. This short chapter

provides enough theoretical background on coding for following the next chapters of this thesis.

Chapter 3

Dynamics of Analog Iterative Decoding

In this chapter, we show that conventional iterative decoding can be formulated as a fixed-point problem solved iteratively by the successive substitution (SS) method. Then we investigate the dynamics of a continuous-time (asynchronous) analog implementation of iterative decoding, and show that it can be approximated as the application of the well-known successive relaxation (SR) method for solving the fixed-point problem. We observe that SR with the optimal relaxation factor can considerably improve the error rate performance of iterative decoding for short low-density parity-check (LDPC) codes compared to SS. Our simulation results for the application of SR to belief propagation (sum-product) and min-sum algorithms demonstrate improvements of up to about 0.7 dB over the standard SS for randomly constructed LDPC codes. The improvement in performance increases with the maximum number of iterations and by accordingly reducing the relaxation factor. The asymptotic result, corresponding to infinite maximum number of iterations and infinitesimal relaxation factor represents the steady-state performance of analog iterative decoding. This means that under ideal circumstances continuous-time

(asynchronous) analog decoders can outperform their discrete-time (synchronous) digital counterparts by a large margin. Our results also indicate that with the assumption of a truncated Gaussian distribution for the random delays among computational modules, the error rate performance of analog decoder, particularly in steady-state, is rather independent of the variance of the distribution. The proposed simple model for analog decoding, and the associated performance curves, can be used as an "ideal analog decoder" benchmark for performance evaluation of analog decoding circuits. We also use the proposed model for studying throughput and convergence speed in analog decoders.

3.1 Introduction

Recently, iterative decoding methods have attracted much interest because they can be used for decoding the best known coding schemes such as turbo codes and low-density parity-check (LDPC) codes. The need for floating point computations and the iterative nature of these algorithms have motivated some very recent research on their analog electronic implementation [19]-[33]. The main motivation for performing iterative decoding in the analog domain instead of conventional digital circuits is to improve and to minimize the power/speed ratio and area consumption in VLSI chips. Also, analog circuits generate less noise compared to conventional digital circuits and therefore are attractive when one has to use a decoder in the proximity of certain noise sensitive circuits such as low noise amplifiers. In this chapter, we demonstrate yet another, less expected, advantage for analog decoding, i.e., an intrinsically better performance.

Analog signal processing has been greatly influenced by C. A. Mead's interesting idea of exploiting semiconductor physics and transistor characteristics for performing analog computations in artificial neural networks [11]. While biological neural networks are essentially continuous-time and asynchronous, conventional iterative decoders are not

[39]. In practice, these algorithms have been conventionally implemented by digital synchronous circuits where a clock signal provides global timing among circuit elements and computational modules. In simulations also, the assumption of synchronization among different modules is always made, making the results only applicable to synchronous circuits. So far, to the best of our knowledge, no theoretical analysis for the dynamics of continuous-time asynchronous analog iterative decoding has been provided. For the sake of brevity, in the rest of this thesis, we will use the term "analog decoding" and "digital decoding" to refer to "continuous-time asynchronous analog iterative decoding," and "discrete-time synchronous iterative decoding," respectively.

Despite the missing theory for the dynamics of iterative decoding on asynchronous machines, proof-of-concept analog decoder chips for very short codes have already been designed and fabricated [19]-[33]. In [19], Hagenauer *et al.* designed analog decoders for a tail-biting convolutional code, and a turbo codes with (7,4) Hamming codes as component codes. They also solved the system of nonlinear differential equations corresponding to analog decoding circuits using Runge-Kutta method and reported BER curves that were slightly inferior to the conventional discrete-time simulation results. Solving the differential equations however appeared to be very time consuming and limited the simulations to only simple codes[19]-[20]. In [21]-[23] the fabrication of maximum a posteriori (MAP) decoders for short tail-biting convolutional codes were reported. In [25], the authors have implemented one stage of the alpha metric computation of Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm for an 8-state convolutional code. In [20], [26], and [29] implementation of turbo decoders with interleaver lengths 16, 16, and 40 have been respectively reported. In [27], a MAP decoder for the (8,4) Hamming code has been fabricated. The design of an analog decoder for convolutional duo-binary codes for the DVB-RCS standard was presented in [24]. In [30]-[33], we proposed analog circuits for implementation of min-sum (MS) decoding algorithm.

While in [21] and [22], the authors have reported measured performance for their chips which closely followed the simulated bit error rate (BER), in [26]-[29] a degradation is observed compared to the simulated BER. In all these references, simulated BER curves were obtained based on discrete-time synchronous model. In [19] and [20], in addition to discrete-time simulations, the authors also provided continuous-time simulation results for analog decoding circuits which closely followed the measured and discrete-time simulated BER curves. In [28], the authors proposed a discrete-time model for analog turbo decoding. To perform this, they approximated the operation of each computational module by a linear first-order filter. They then included mismatch in this model and observe 0.3-0.5 dB degradation compared to the conventional synchronous decoding. This brought the mismatched curves in the close vicinity of the measured BER for analog decoding. For turbo codes, [20] and [24] provided examples where analog decoding slightly (by about 0.1dB) outperformed digital decoding with finite number of iterations.

In this chapter, we provide a framework for the analysis of analog iterative decoding of LDPC codes, and show that the dynamics of analog decoding is in fact different than that of digital decoding. For this, the dynamics of analog decoding is modeled by a system of first-order non-linear differential equations. This system can then be solved numerically. We show that, under certain conditions, by applying forward Euler method to the system of differential equations, analog decoding can be approximated as the application of the well-known successive relaxation (SR) method [44]-[46] to the fixed-point problem of iterative decoding. The approximation error tends to zero as the relaxation factor β approaches zero. The value of β plays an important role in the performance and complexity of the SR method. The set of recursive equations resulting from the application of SR to the fixed-point problem of iterative decoding provides a general framework which embodies not only analog decoding ($\beta \rightarrow 0$), but also digital decoding based on flooding (parallel) schedule [39]. The latter corresponds to the application of the

well-known successive substitution (SS) [44] to the fixed-point problem of iterative decoding and is a special case of SR when β=1.

We demonstrate that the error rate performance of SR improves by increasing the maximum number of iterations N and by correspondingly reducing β. The optimal value of β, corresponding to minimum BER, appears to be always less than one. In the limit, as $N \rightarrow \infty$ and $\beta \rightarrow 0$, the performance of SR asymptotically tends to the steady-state performance of analog decoding. Our simulation results indicate that this asymptotic performance is considerably better than the performance at $\beta = 1$, which corresponds to conventional digital decoding. For the tested LDPC codes, up to about 0.7 dB improvement for belief propagation (BP) [39] is observed. This improvement in performance is due to better convergence rate (larger percentage of convergence to correct codewords) for analog decoding. Our results also show that under the assumption of symmetric truncated Gaussian distribution for the delays between the computational modules of an analog decoder, the error rate performance, especially in steady-state, is independent of the variance of the delay distribution.

Adopting alternative numerical methods to improve convergence in iterative decoding of turbo codes was proposed in [47] and was also studied in [48]. In [48], the authors identified turbo decoding as fixed point iteration and examined alternative numerical methods including SR for solving the corresponding fixed-point problem. The final conclusion of [48], however was that for turbo codes the conventional SS is the superior choice in most situations. This is in clear contrast with our results for LDPC codes, where SR in particular provides considerable improvement over SS for both BP and MS algorithms. On the same theme of research, in the past few years and mainly in the context of short LDPC codes, there have been many attempts to improve the performance of iterative decoding algorithms [40]-[43]. We have noted that many of these proposed methods can be interpreted as the application of different iterative numerical methods for solving the fixed-point problem in hand. For instance, in [40]-[43], variants of "damped substitu-

tion method" [48] have been used for improving the performance of MS and BP algorithms.

In addition to showing that analog decoding can outperform digital decoding, one of the main contributions of this work is to provide a new benchmark, as "ideal" analog decoding, for the performance of analog decoding circuits. This would replace the conventional benchmark of discrete-time synchronous decoding.

In this work, we are mainly concerned about the statistical behavior of decoders in general and their error probabilities in particular. As we will see later in this chapter, by using SR method many important aspects of analog decoders can be studied. This includes throughput and statistical convergence speed in analog decoders. Nevertheless, the proposed model can be used to analyze analog decoders for given instances of received values at the output of the channel. Such analysis includes transient and steady-state responses of the decoder, and can be used to efficiently study the behavior of an analog decoder once an estimate of the delay distribution between variable and check modules is available. Our simple model of analog decoding can also be used in designing decoders. As our proposed model is much simpler than the circuit-based model for analog decoding, it will significantly speed up any analysis or design process, and can be used even when the complexity of circuit simulation is prohibitive.

SR method can also be implemented on a synchronous discrete-time platform. For smaller values of β, the resulting improvement over the conventional SS would be at the expense of larger average number of iterations for convergence.

The organization of this chapter is as follows: in Section 3.2, we represent iterative decoding as a fixed-point problem, and explain how conventional iterative decoding is equivalent to the application of the so-called successive substitution method to the fixed-point problem. Section 3.3 is on analog implementation of iterative decoding. We propose our simple model for analog decoding and the corresponding set of first-order differential equations in this section. In Section 3.4, we use the model of Section 3.3 to

analyze the dynamics of analog decoding by numerically solving the differential equations. We demonstrate the accuracy of the model by comparing its results to those obtained by circuit simulations for a small decoder. We also explain how the application of forward Euler to the system of differential equations is equivalent to the application of the so-called successive relaxation method to the fixed-point problem of iterative decoding. In Section 3.5, we present our simulation results and conclude among other things that analog decoding can outperform conventional digital decoding for short LDPC codes. Section 3.6 is devoted to concluding remarks.

3.2 Iterative Decoding

Iterative decoding can be considered as an iterative numerical method for solving the complex problem of decoding by finding the most probable transmitted codeword or bit, based on the observation of channel outputs, noise and channel characteristics, and the structure of the code. For the sake of simplicity, we consider binary (n,k) linear block codes. As we saw in Chapter 2, the codes can be represented by "Tanner graphs" [35], which are constructed based on parity-check equations defining the code. There are two sets of nodes in a Tanner graph: 1) variable nodes (*VAR*) representing code bits, and 2) check nodes (*CHK*) representing parity-check equations. Suppose that the Tanner graph has e edges, and that the iterative algorithm, which for example could be BP or MS, has converged to a solution. Also denote the incoming and outgoing messages to and from *CHK* nodes by column vectors X and Y, respectively. These vectors have length e. We then have:

$$\begin{cases} VAR: & f(Y,R) = X \\ CHK: & g(X) = Y \end{cases}, \tag{3-1}$$

or equivalently,

$$f(g(X),R)=h(X,R)=X, \tag{3-2}$$

where f and g represent variable node and check node operations, respectively, and R is the received vector and has length n. Operations f and g are determined by the decoding algorithm, the type of messages, and channel characteristics [39]. Equation (3-2) is a fixed-point problem [44] as the h function maps X to itself.

Ideally, we would like the iterative algorithm to converge to a fixed-point X, and the hard-decision assignment corresponding to X to be the most likely transmitted codeword or to correspond to the bit-level maximum a posteriori (MAP) decoding. Although this is the case for cycle-free graphs, for graphs with cycles, which inevitably appear in representing good codes, this may not be the case [39]. In practice, in many cases, as soon as the hard-decision assignment at variable nodes is a codeword, the iterative algorithm is stopped. Interestingly, this results in very good performance for practical codes. In fact, one can think of iterative decoding as finding a solution for (3-2) based on iterative numerical methods. In particular, for the most commonly used flooding or parallel schedule [39] that was shown earlier in Chapter 2, iterative decoding can be expressed by the following iterative formula which is based on the application of the well-known *successive substitution* method (Successive substitution is also referred to in the literature as "successive approximation," "nonlinear Richardson iteration," "fixed-point iteration," or "Picard iteration.")[44] to (3-2):

$$X_{l+1} = h(X_l, R) \tag{3-3}$$

where l is the iteration number. Flooding schedule can be easily implemented on a synchronous discrete-time machine where the global timing for message-passing is fully controlled by a clock signal. In this case, messages are passed from *VAR* nodes to *CHK*

nodes, and subsequently from *CHK* nodes to *VAR* nodes in a synchronized iterative fashion. There are however many advantages in implementing iterative algorithms on asynchronous machines using continuous-time analog VLSI [19]-[33] that will be studied in more detail in Chapter 4. In brief, these advantages include higher speed and lower power consumption and lower fabrication area. Also, analog circuits generate less noise compared to conventional digital circuits and therefore are attractive when one has to use a decoder in the proximity of certain noise sensitive circuits such as low noise amplifiers. In the following section, we discuss the dynamics of continuous-time analog iterative decoding.

3.3 Analog Implementation of Iterative Decoding

The dynamics of iterative decoding on continuous-time (asynchronous) machines is different than that of conventional discrete-time (synchronous) ones. In particular, it is fair to say that an analog decoder, even if it is designed based on a flooding schedule framework, may no longer support the (synchronized) flooding schedule as different computational modules may have different processing delays, and as the output of each module (before settling down to its steady-state value) is immediately propagated through the edges of the graph and is applied to the input of the next module in the absence of global synchronization. Asynchronous propagation of the messages is prone to further deviate the decoding process from synchronous decoding due to different propagation delays among different pairs of *VAR* and *CHK* modules. This effect is particularly profound for large circuits with a large range of delays among different modules. Even if we assume that the processing and propagation delays are the same, our analysis indicates that continuous-time analog decoding will still have a different behavior compared to conventional discrete-time decoding.

Assuming that we use analog circuits to implement iterative decoding in a parallel scheduling framework, we can approximate the dynamics of the system by the following differential equations:

$$\begin{cases} \tau_x \dot{X}(t) = f(Y(t-T_x), R) - X(t) \\ \tau_y \dot{Y}(t) = g(X(t-T_y)) - Y(t) \end{cases} \tag{3-4}$$

where τ_x and τ_y are $e \times e$ diagonal matrices which contain different time constants, representing different propagation delays affecting the corresponding X_i and Y_i variables, respectively. This model is derived based on Figure 3-1 where the propagation delay due to interconnections between the nodes, corresponding to the edges of the Tanner graph, as well as the resistive and capacitive loads of the nodes are simplified and modeled by lumped capacitors and resistors based on Elmore's delay approximation [10]. Processing delays or dead-time of the nodes, which indicate the lagging time of the outputs with respect to the inputs, have also been taken into account and represented by column vectors $T_x = (T_{x_1}, ..., T_{x_e})^t$ and $T_y = (T_{y_1}, ..., T_{y_e})^t$. Vectors $X(t-T_y)$ and $Y(t-T_x)$ in (3-4) are defined as $(X_1(t-T_{y_1}), ..., X_e(t-T_{y_e}))^t$ and $(Y_1(t-T_{x_1}), ..., Y_e(t-T_{x_e}))^t$, respectively. In general, processing delay can be different for nodes of different type (*CHK* or *VAR*) or nodes with the same type but different degrees.

Equation (3-4) expresses the dynamics of continuous-time analog iterative decoding by a pair of interrelated first-order nonlinear differential equations. If the analog decoder converges to a fixed-point then the left-hand side of these equations becomes zero and vectors $X(t-T_y)$ and $Y(t-T_x)$ converge to fixed vectors X and Y which are independent of t, T_x and T_y. In this case, equations in (3-4) reduce to (3-1) and thus to (3-2).

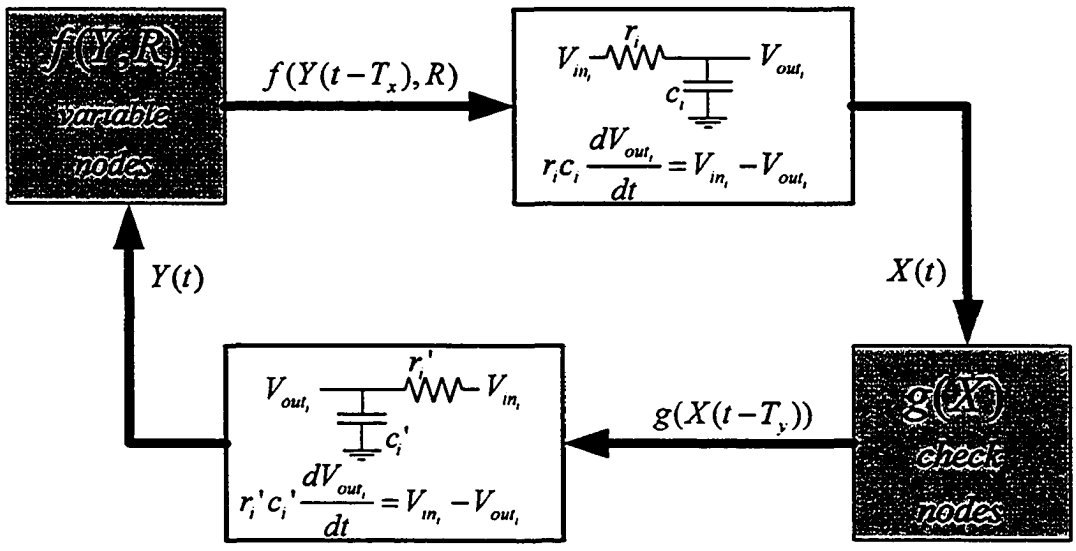


Figure 3-1 A model for continuous-time analog iterative decoding.

Equation (3-4) in general is very difficult to solve. To simplify the analysis, we ignore processing delays compared to propagation delays ($T_x = T_y = 0$). This may be justified for practical analog decoders which are comprised of many low-degree processing modules and long interconnections between these modules. To further simplify the problem, we simplify the model in Figure 3-1 and replace it with the model in Figure 3-2 which is based on the application of Elmore's delay approximation [10] to represent the total round-trip delay in one iteration. The pair of equations in (3-4) will then reduce to the following first-order differential equation.

$$\begin{aligned} \tau\dot{X}(t) &= f\big(g(X(t)),R\big) - X(t) \\ &= h(X(t),R) - X(t) \end{aligned} \tag{3-5}$$

We will demonstrate in the next section through an example that this significantly simplified model still provides a close approximation for the behavior of analog decoding.

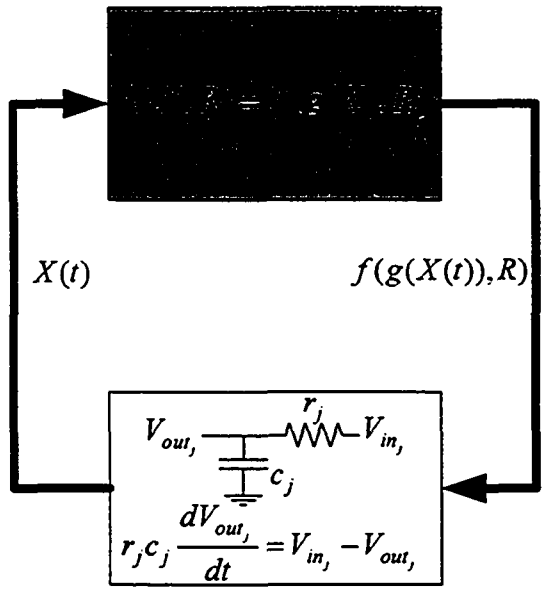


Figure 3-2 A simplified model for continuous-time analog iterative decoding.

In this chapter, we consider belief propagation in the likelihood ratio (LR) domain and min-sum algorithm in the log-likelihood ratio (LLR) domain. For these algorithms, function $h(.)$ in (3-5) corresponds to the following *VAR* and *CHK* operations, respectively (The operations are shown for only two inputs. For larger number of inputs, a cascade of two-input blocks will produce the output):

BP in LR domain:

General form of the messages: $\lambda(p,q) = \frac{p}{q}$

VAR: $f(\lambda_0,\lambda_1,\lambda_2) = \lambda_0\lambda_1\lambda_2$

(λ_0 is the initial message from the channel)

CHK: $g(\lambda_1,\lambda_2) = \frac{1+\lambda_1\lambda_2}{\lambda_1+\lambda_2}$

MS in LLR domain:

General form of the messages: $\Lambda(p,q) = \ln\left(\frac{p}{q}\right)$

VAR: $f(\Lambda_0,\Lambda_1,\Lambda_2) = \Lambda_0 + \Lambda_1 + \Lambda_2$

(Λ_0 is the initial message from the channel)

CHK: $g(\Lambda_1,\Lambda_2) = sign(\Lambda_1\Lambda_2)\min(|\Lambda_1|,|\Lambda_2|)$

In the above equations, p and q are the probabilities of a bit being 0 or 1, respectively, and *sign* function is defined by (2-17). Furthermore, with a slight abuse of notations, we have used f and g to also denote the variable and check node operations for nodes with two inputs in addition to the notation in (3-1).

3.4 Dynamics of Analog Decoding

For a given nonlinear function $h(.)$, it is usually not possible to solve the initial-value problem (3-5) analytically. Numerical methods should then be used instead. In this work, we assume that before applying the initial values corresponding to the received vector R to the decoder, the decoder is in the steady-state corresponding to uniform a priori probabilities, i.e., $X(0^-) = \mathbf{0}$ and $X(0^-) = \mathbf{1}$, in LLR and LR domains, respectively, where **0** and **1** are the all-zero and all-one vectors, respectively. This is the same assumption that was made in (2-14). We also assume that at time $t = 0$, the initial values corresponding to the received vector R are applied to the decoder. These initial values immediately appear on the edges at the output of *VAR* nodes and correspond to $X(0^+)$.

There are several numerical methods that can be used to solve (3-5), each with different stability, accuracy and complexity. In the following, we use *forward* and *backward Euler* methods [45], [49]. These methods are respectively described by the following equations:

$$X(t+\Delta t) = X(t) + \Delta t \tau^{-1}\left(h(X(t), R) - X(t)\right) \tag{3-6}$$

$$X(t+\Delta t) = X(t) + \Delta t \tau^{-1}\left(h(X(t+\Delta t), R) - X(t+\Delta t)\right) \tag{3-7}$$

where $X(t+\Delta t)$ and $X(t)$ are the estimates of $X(.)$ at time instants $t+\Delta t$ and t (for a small value of Δt), respectively.

The method of forward Euler (equation (3-6)) is an explicit integration method that simply estimates $X(t+\Delta t)$ as a function of $X(t)$ and is particularly attractive for its simple implementation. However, forward Euler as well as any other conventional explicit integration method is prone to stability problems and requires a sufficiently small time-step (Δt) in order to remain stable (Dahlquist's first barrier theorem) [50]. Backward Euler, on the other hand is one of the linear implicit integration methods that are absolutely stable (Dahlquist's second barrier theorem) [50]. The downside of backward Euler however is its high complexity. For example, in equation (3-7), for finding an estimate of $X(t+\Delta t)$ a nonlinear equation needs to be solved. This is the price that is paid for using larger time-steps and better robustness and accuracy compared to forward Euler.

To demonstrate that Figure 3-2 provides a good model for analog iterative decoding, we use it to find the time response of (analog) min-sum algorithm applied to the Tanner graph of Figure 3-3. This Tanner graph, which belongs to the (7,4) Hamming code, was also used in [31] to demonstrate the functionality of the designed circuits for *VAR* and *CHK* modules in that paper.

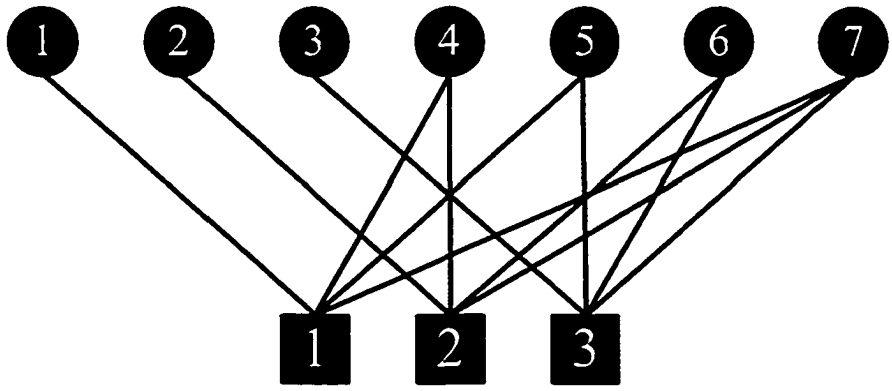


Figure 3-3 A Tanner graph for (7,4) Hamming code.

Using the exact same input vector as the one used in [31] (R = (1, 2, 0.5, 1, 2, 2, -1.7)^t), we apply backward Euler method of (3-7) to solve the differential equation (3-5) while function $h(.)$ is replaced by the combination of *VAR* and *CHK* operations for min-sum algorithm in the LLR domain, given in the previous section. We have also applied higher order linear multi-step numerical integration methods to this problem and the results are close to those obtained by backward Euler method. In our simulations, we have assumed that all the edges have the same time constant, which is estimated to be equal to τ = 20nsec. This estimate is based on the circuits in the next chapters of this thesis and also in [32]-[33], [51], which are much faster than earlier versions reported in [31]. In Figure 3-4 and Figure 3-5, we have shown the LLR values corresponding to the seven *VAR* nodes of the Tanner graph. For each node, this is the sum of all the incoming LLRs and the initial value of the node. These outputs are denoted by "O" and shown by thicker lines. For comparison, in Figure 3-4 and Figure 3-5, we have also given similar curves denoted by "o" obtained by circuit simulation using Cadence Spectre tools and based on the circuits that are presented later for implementing check nodes and variable nodes in Chapter 4 and Chapter 5, respectively. As one can see the two sets of curves are close in general, especially for steady-state responses.

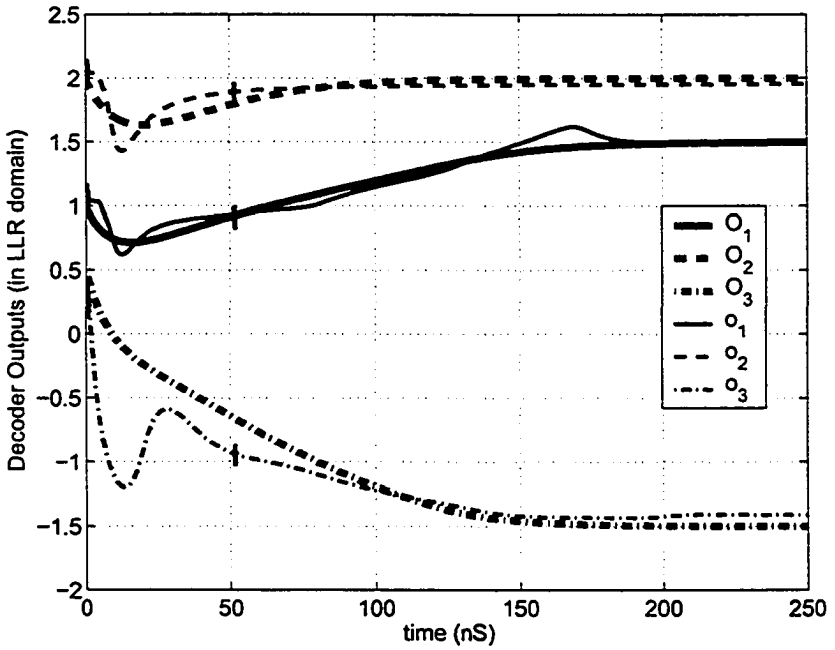


Figure 3-4 Outputs of MS analog decoder based on circuit simulation (o_i) and the simple model of Figure 3-2 (O_i); i: outputs 1-3.

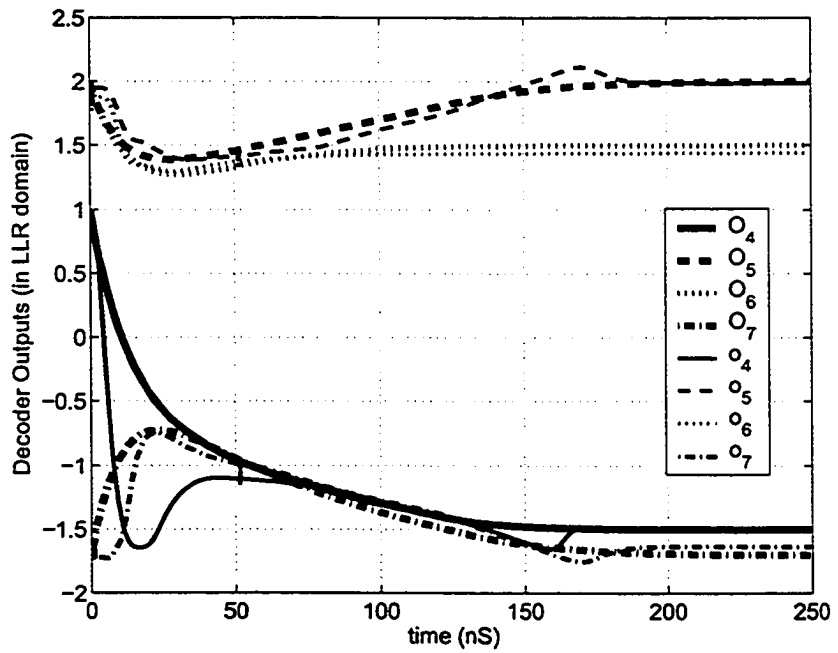


Figure 3-5 Outputs of MS analog decoder based on circuit simulation (o_i) and the simple model of Figure 3-2 (O_i); i: outputs 4-7.

Similarly, we obtained the outputs of the MS decoder for the (7,4) Hamming code and by starting from the same initial conditions as those used in [31] (Note that in [31], $X(0^-) \neq \mathbf{0}$). Again, although the circuits of [31] are very different than those proposed in this thesis, the obtained curves are close to those obtained by circuit simulation and presented in Figures 7, 8 and 9 of [31]. This is an evidence for the universality of the model introduced in Figure 3-2 and equation (3-5), i.e., for any analog decoder, as long as the processing function h, initial values R, and matrix τ of time constants are known, one can use (3-5) to obtain a good approximation for the output signals of the decoder. Our examples show that for the tested MS analog decoders these approximations are close to circuit simulations.

It should be noted that the curves in Figure 3-4 and Figure 3-5 and also the curves in [31] are obtained by performing circuit simulation on a circuit with a few thousand short-channel MOS transistors. This is while, using our model, the simulation is greatly simplified and is reduced to solving only a first-order differential equation with 12 variables (e=12). Clearly, our simple model, which is mostly circuit-independent, is not capable of incorporating the precise characteristics of different circuit elements and is thus less accurate in predicting the actual behavior of an analog decoder integrated circuit. Nonetheless, our model can serve as a simple tool for analyzing "ideal" continuous-time analog decoding and can be used to provide an estimate of the behavior of practical analog decoders including important characteristics such as speed of convergence.

We also applied the forward Euler method of (3-6) to the same example, and observed that as long as we choose Δt small enough, the results are close to those obtained by backward Euler (While there does not seem to exist any general result for the stability of forward Euler when applied to nonlinear differential equations, our experiments with this algorithm for solving (3-5) indicates that choosing $\Delta t/\tau \ll 0.1$ results in time responses close to those obtained by $\Delta t/\tau = 0.1$). The application of forward Euler to (3-5)

is interesting not only because it is arguably the simplest numerical method that can be used to solve (3-5), but also since in the discrete-time domain it reduces to the following recursive equation

$$X_{l+1} = X_l + \Delta t.\tau^{-1}\big(h(X_l, R) - X_l\big), \tag{3-8}$$

which as we will see shortly, will in turn reduce to a well-known iterative method for solving the fixed-point problem of (3-2). In (3-8), $X(t+\Delta t)$ and $X(t)$ of (3-6) are replaced by X_{l+1} and X_l, respectively. As Δt and thus the diagonal elements of $\Delta t.\tau^{-1}$ go to zero, Equation (3-8) provides a better estimate of the solution to (3-5). In particular, as Δt goes to zero and as the number of iterations goes to infinity, we obtain the steady-state response of an analog decoder.

Here, we would like to emphasize that (3-8) can also be used as an iterative algorithm on a synchronous discrete-time platform. For this scenario, we only consider the case where all the diagonal elements of matrix $\Delta t.\tau^{-1}$ are equal (Note that as we will see in the next section, the assumption of equal time constants has insignificant effect on the statistical steady-state response of (3-8)). We use the notation β to denote this common value. Equation (3-8), can then be rewritten as:

$$X_{l+1} = X_l + \beta\big(h(X_l, R) - X_l\big), \tag{3-9}$$

which is equivalent to a well-known iterative method for solving the fixed-point problem of (3-2). This method is referred to as *successive relaxation (SR)* [44]-[46], and is an alternate to the successive substitution (SS) method, described by (3-3). Parameter β is called "*relaxation factor*." For β=1, SR reduces to SS.

In the literature, SR is often used to iteratively solve systems of linear equations [44]-[46]. Parameter β is then optimized to maximize the speed of convergence. The op-

timal value of β, which is often difficult to determine, appears to be between 1 and 2 [45]. In fact, in some literature, the term "overrelaxation" is reserved only for $\beta > 1$, and $0 < \beta < 1$ is referred to as "underrelaxation" [46]. While overrelaxation helps with the speed of convergence for the cases where convergence is guaranteed, underrelaxation is often used for nonlinear equations to help with increasing the chance of convergence [46]. Thus, in (3-9) as β tends to zero, we not only obtain a better estimate of analog decoding, but also expect to improve the chance of convergence for SR.

One should note that, in this work, we are mainly interested in the convergence properties of SR from a *statistical* point of view, i.e., when vector R changes randomly. To be more precise, we are interested in the statistics on how often SR can find the transmitted codeword or the transmitted bit correctly; or equivalently, what the average probability of error for SR is.

Finally, we would like to make the remark that when the propagation delay is the dominant delay factor in an analog iterative decoder, the same approach taken to derive (3-8) and (3-9) from (3-5) can also be used to derive the SR version of (3-4). This can be simply verified by rewriting (3-4) as follows:

$$\tau_z \dot{Z}(t) = \hbar(Z(t), R) - Z(t),$$

where $\tau_z = \begin{bmatrix} \tau_x & 0_{exe} \\ 0_{exe} & \tau_y \end{bmatrix}$, $Z(t) = \begin{bmatrix} X(t) \\ Y(t) \end{bmatrix}$, $\hbar(Z(t), R) = \begin{bmatrix} f([0_{exe} \quad I_{exe}] \times Z(t), R) \\ g([I_{exe} \quad 0_{exe}] \times Z(t)) \end{bmatrix}$, and

0_{exe} and I_{exe} are $e \times e$ zero and identity matrices, respectively. While in this case, the number of differential equations is twice that of (3-5), we have not observed noticeable change compared to the results obtained by the simpler model of (3-5).

3.5 Simulation Results

We perform simulations on an optimized irregular (1268,456) LDPC code [52] and a regular (504,252) code [53] over an additive white Gaussian noise channel with BPSK modulation. In our simulations, we stop iterations as soon as a codeword is detected or when a maximum number of iterations is reached. We have investigated the performance of each code for both belief propagation (BP) and min-sum (MS) algorithms. BP and MS simulations are performed in LR and LLR domains, respectively. For all the reported results, we have either 200 codeword errors per simulation point or 10^8 codewords have been simulated. In the following, we analyze the performance of SR algorithm and analog decoding by first simulating (3-8) for different delay distributions, and then by focusing more on the constant delay scenario to compare the error correcting performance of SR and SS methods. We also use SR method for studying throughput and convergence speed in analog decoders with constant and different delay distributions.

3.5.1 Effect of Delay Distribution

To study the effect of the delay distribution on the error correcting performance of the decoder, we use (3-8) for updating the messages. For delay distributions of the edges we assume symmetric truncated Gaussian distribution in the interval $[\tau_{min}, \tau_{max}]$, where $\tau_{min} = 10\,\Delta t$ and $\tau_{max}=100\,\tau_{min}$. The first constraint is enforced to ensure a good accuracy for the forward Euler algorithm, while the second constraint is derived based on the lengths of wires in a practical layout for an LDPC decoder [18] (note that the delay is proportional to the square of the wire length [10]). We consider Gaussian distributions with 4 different variances. The normalized values of standard deviations are selected as

$\sigma/\Delta t = 0$, 19.8, 198 and 1980. The first value ($\sigma = 0$) corresponds to fixed delay for all the edges, while the last value results in an almost uniform distribution of delays in the interval.

Figure 3-6 shows the bit error rate (BER) and word error rate (WER) curves of BP applied to (1268,456) code for the four delay distributions and three different values of N, the maximum number of iterations, i.e., N=200, 1000, 5000. The curves for N=10,000 has also been obtained but are close to those for N=5000 and thus are not reported.

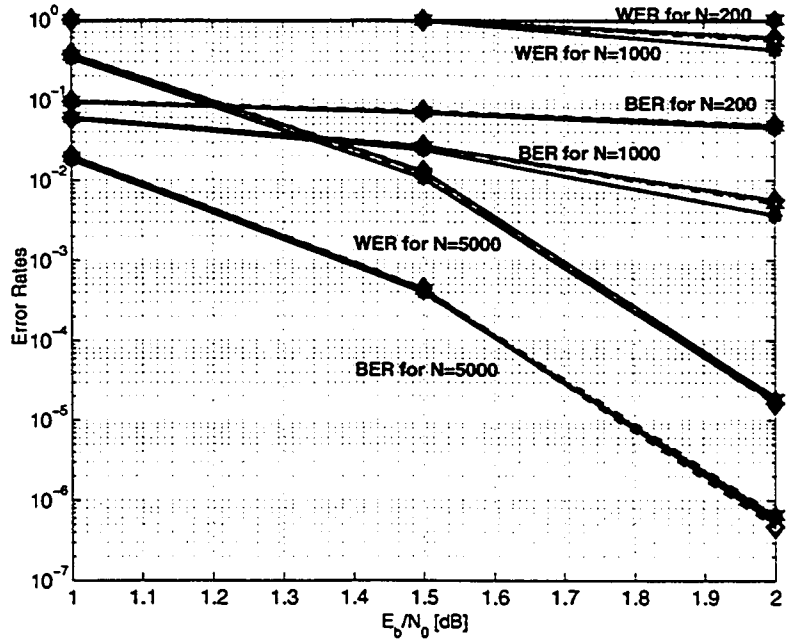


Figure 3-6 Effect of different delay distributions on BP error rate of an irregular (1268,456) LDPC code. N is the maximum number of iterations.

($\sigma/\Delta t$=0: '__', $\sigma/\Delta t$=19.8 : '_ _◇', $\sigma/\Delta t$=198 : '_ .✱' ,and $\sigma/\Delta t$=1980: '__✱')

Figure 3-7 provides similar results for the (504,252) code. These results indicate that in general the performance of analog decoder improves with increasing N which is equivalent to increasing the processing time of the decoder. This performance improve-

ment however saturates and reaches a steady-state when the processing time is large enough. Our results show that when enough time is given to the decoder to settle down in its steady-state performance, this performance is almost independent of the delay distribution.

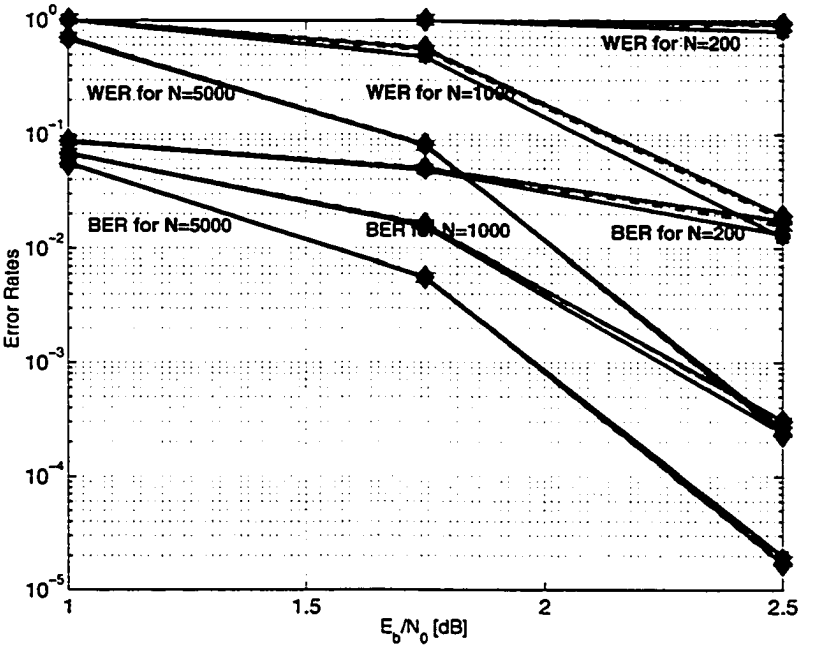


Figure 3-7 Effect of different delay distributions on BP error rate of a regular (504,252) LDPC code. N is the maximum number of iterations.

($\sigma/\Delta t$ =0: '__' , $\sigma/\Delta t$ =19.8 : '_ _◇', $\sigma/\Delta t$ =198 : '_ .✱ ' , and $\sigma/\Delta t$ =1980: '__✱')

It can be seen in both Figure 3-6 and Figure 3-7, for N=5000, the four curves corresponding to different values of σ practically coincide for both BER and WER. Even for smaller values of N, the effect of σ on performance is rather small. Similar observations are made for MS algorithm.

It is worth mentioning that one should not interpret the above results as the independence of the response of analog decoding from delay distribution. Firstly, it is easy to show that the response of the decoder (even the steady-state response) to particular in-

stances of input vector R would change depending on the delay distribution. So the above results are valid in a *statistical* context. Secondly, the results are obtained based on the assumption of truncated Gaussian distribution for delays. One may be able to find other, probably less realistic, delay distributions where the statistical behavior of the decoder (including its steady-state behavior) would be different than the one described above.

3.5.2 Comparison between SR and SS Methods

In this part, we consider SR based on (3-9) for updating the messages. To compare the performance of iterative decoding based on SR (which represents the performance of continuous-time analog iterative decoding when β is small) with that of conventional SS method (which reflects the performance of digital decoding), in Figure 3-8, we show the BER curves of (1268,456) code decoded by relaxed BP (SR applied to BP, SR-BP) versus the relaxation factor β for different E_b/N_0 values.

There are two sets of curves in Figure 3-8, corresponding to maximum number of iterations N equal to 200 and 10,000. It can be seen that for a large range of $\beta < 1$, SR provides improvement over successive substitution (β=1), and that the improvement in general is increased by reducing β and increasing N.

The performance for N=10,000 and β=0.05 can be considered as a good approximation for the steady-state performance of the continuous-time analog implementation of BP decoder. The BER and WER curves for this case are presented in Figure 3-9 along with the curves for SS (discrete-time synchronous BP decoding based on flooding, SS-BP). The maximum number of iterations for SS is also chosen to be N = 10,000. As can be seen, SR outperforms SS by up to 0.5 dB.

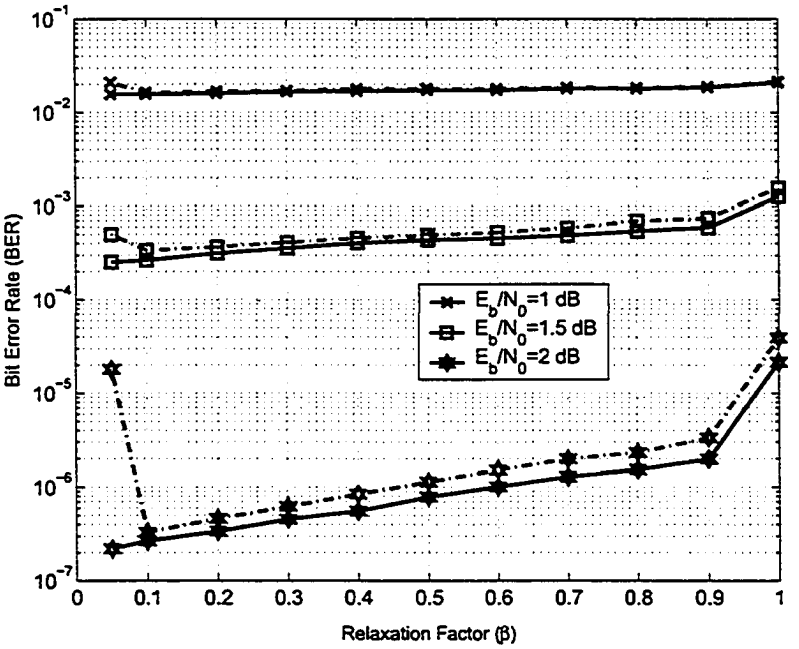


Figure 3-8 BER curves of (1268,456) LDPC code versus the relaxation factor β for N=200 (_._) and 10,000 (___) and different E_b/N_0 values, decoded by SR-BP.

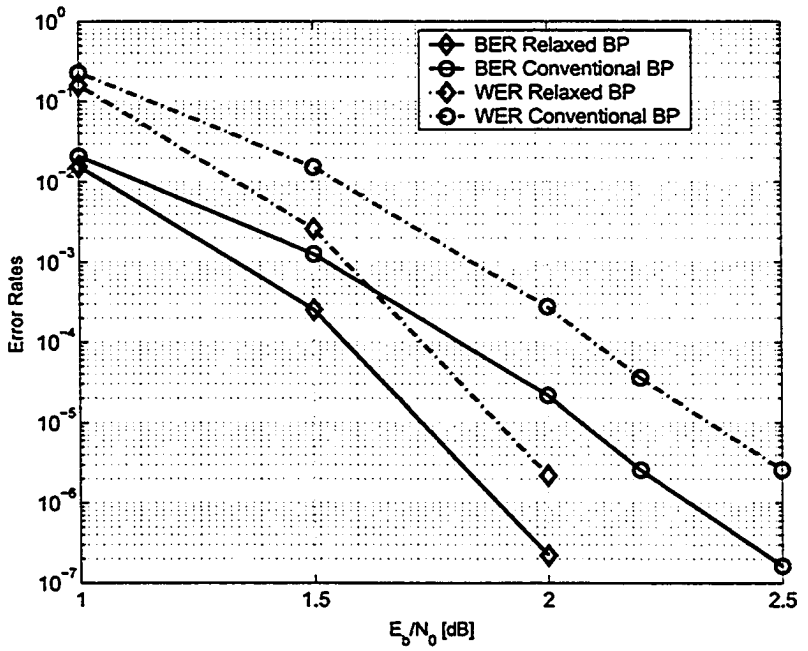


Figure 3-9 BER (___) and WER (_ . _) curves for (1268,456) code, decoded by SR-BP (N=10,000 and β = 0.05) (◊), and SS-BP (N=10,000) (o)

From Figure 3-8, one can see that for SR, the performance gap between N=10,000 and N=200 is very small over the range of $0.1 \leq \beta \leq 1$, which includes β=1 (SS). Over this range, one can then obtain close to the full potential of SR by selecting N = 200. Figure 3-10 shows the average number of iterations vs. β for relaxed BP with N=10,000 at different E_b/N_0 values. As can be seen, the average number of iterations remains almost unchanged over the range of $0.3 \leq \beta \leq 1$ and increases by decreasing β below 0.3. This indicates that there is a complexity cost associated with the better performance of SR on a synchronous discrete-time platform only for smaller values of β. For continuous-time decoding, iterations disappear and are replaced by (or are translated to) the processing time of the decoder.

From Figure 3-8, one can see that for N=200, the performance begins to degrade when β is reduced below 0.1. This can be explained based on the results on the convergence rate of the algorithm. As can be seen in Figure 3-10, by decreasing β below a certain range, the algorithm requires on average more iteration to converge to a solution. Limiting the maximum number of iterations to a small value, such as 200, would then reduce the convergence instances for the algorithm and hence increases the error probability.

At this point, to better position the improvement obtained by SR, we compare it with the result of [54] (BP + order-1 with maximum of 50 iterations), which for this code improves over conventional BP with flooding by about 0.3 dB at BER=10^{-6} [55]. This is about 0.2 dB less than the improvement that SR provides over conventional BP, still with SR having much less complexity compared to the algorithm of [54].

Results of relaxed BP for (504,252) code follow the same trends as those for (1268,456) code. BER and WER curves for relaxed BP with N=10,000 and β = 0.05 are given in Figure 3-11 along with the curves for conventional BP. At low error rates, relaxed BP outperforms conventional BP by about 0.7dB.

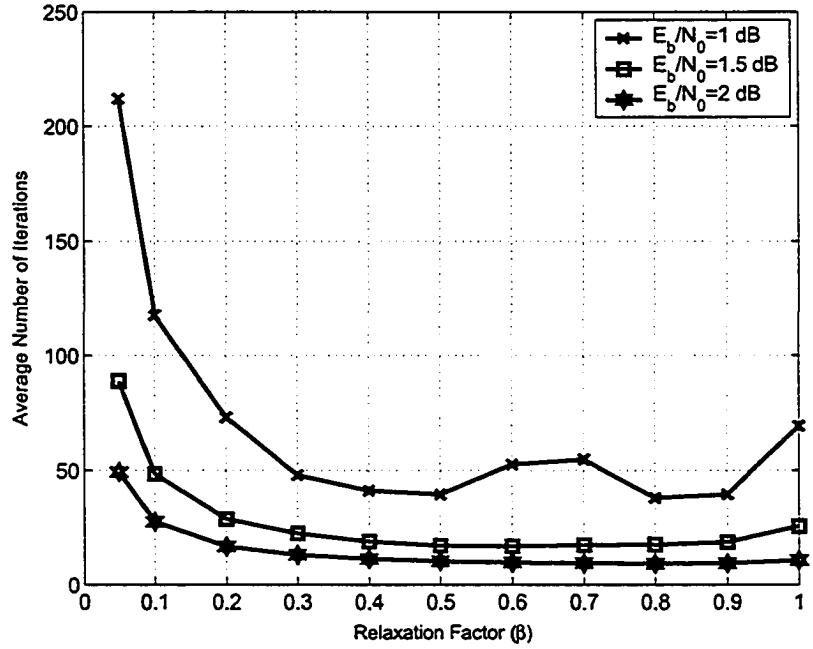


Figure 3-10 Average number of iterations vs. β at different E_b/N_0 values for (1268,456) LDPC code (N=10,000), decoded by SR-BP.

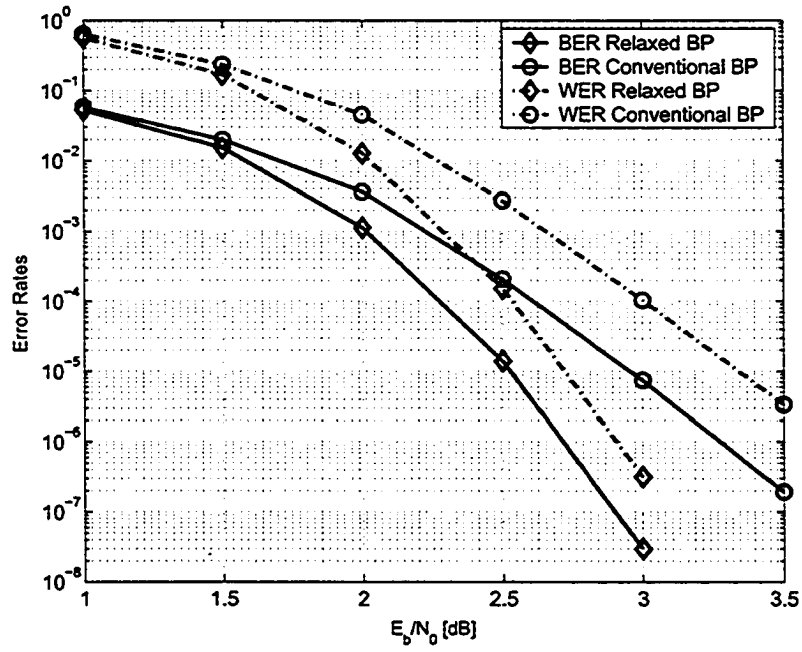


Figure 3-11 BER (___) and WER (_._) curves for (504,252) LDPC code SR-BP (N=10,000 and β = 0.05) (◊), and SS-BP (N=10,000)(o)

Trends similar to those observed for the application of SR to BP are also observed for MS with SR. The BER results of (1268,456) code for relaxed MS with N=200 and 10,000 vs. β are given in Figure 3-12 for different values of E_b/N_0. As can be seen, the performance difference between N=200 and 10,000 is much larger than the corresponding difference for BP in Figure 3-12. This is consistent with the fact that MS usually requires a larger number of iterations compared to BP for convergence (compare Figure 3-13 with Figure 3-10). It is only over the range of $0.9 \leq \beta \leq 1$ that the performance gap between N=200 and N=10,000 is rather small and thus close to the full potential of the algorithm can be achieved by selecting N = 200. As Figure 3-12 shows, over this range the average number of iterations is in fact reduced by decreasing β. Figure 3-12 also shows that the optimal value of β, which minimizes BER, is a decreasing function of N. Similar to BP; the trend seems to indicate that as N tends to infinity, optimal β tends to zero. This optimal asymptotic solution corresponds to the steady-state performance of continuous-time analog decoding. The same trends are observed for the application of relaxed MS to (504,252) code. Figure 3-14 and Figure 3-15 show the BER and WER curves of both codes for relaxed MS with optimal β when N=10,000. For comparison, the curves for conventional MS with flooding are also given. As can be seen, improvements of about 0.5 dB are achievable at large E_b/N_0 values. This improvement in decoding performance occurs by decreasing the average number of iterations for convergence on a synchronous discrete-time platform.

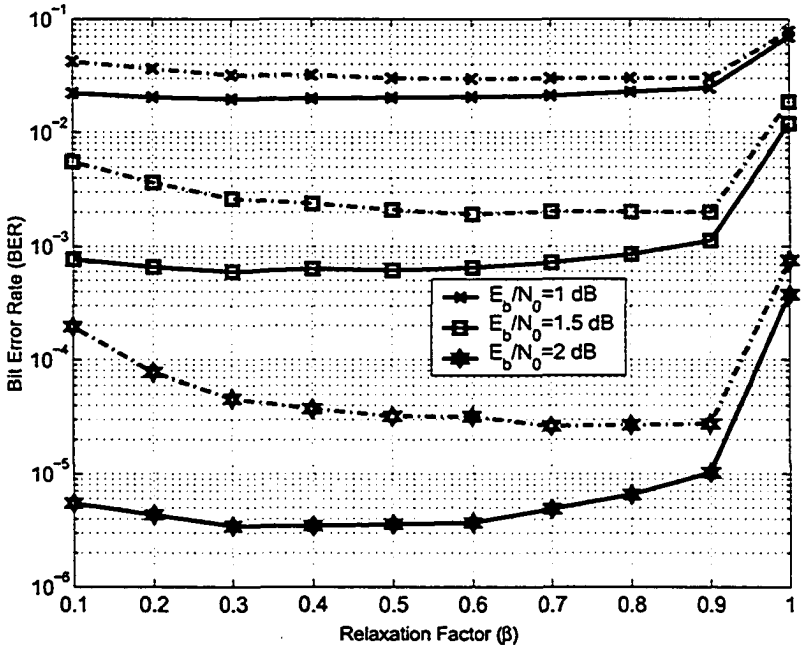


Figure 3-12 BER curves of (1268,456) LDPC code versus the relaxation factor β for N=200 (_._) and 10,000 (___) and different E_b/N_0 values, decoded by SR-MS.

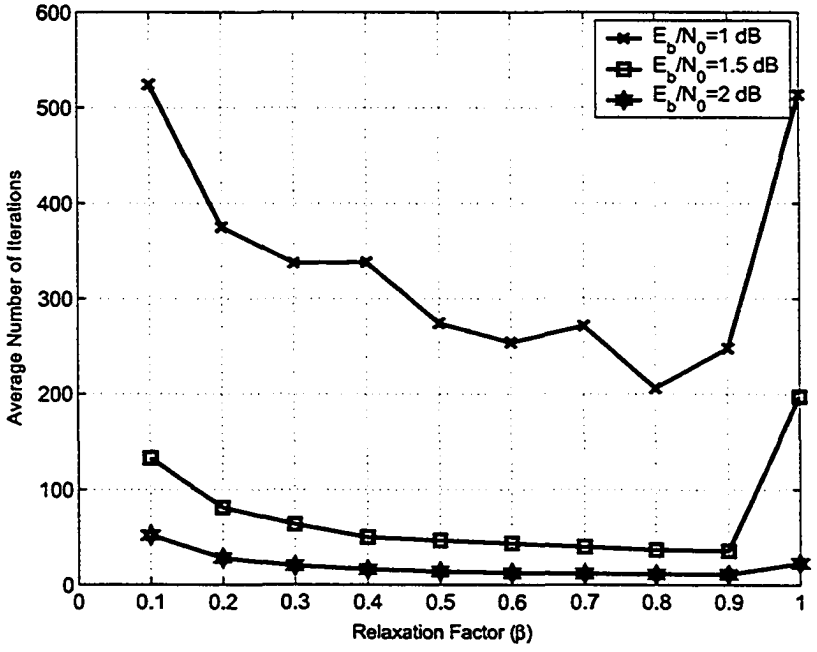


Figure 3-13 Average number of iterations vs. β at different E_b/N_0 values for (1268,456) LDPC code (N=10,000) decoded by SR-MS.

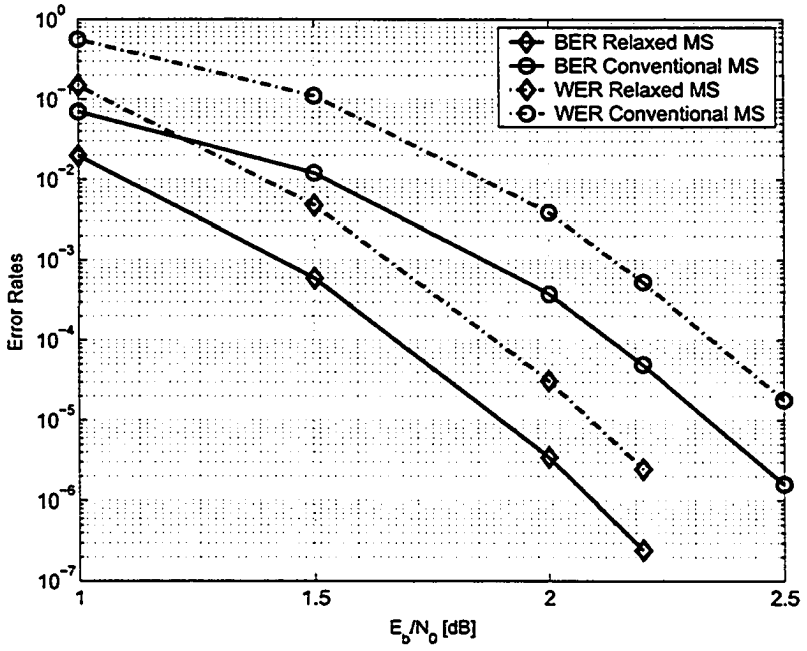


Figure 3-14 BER (___) and WER (_ . _) curves for (1268,456) code, decoded by SR-MS (N=10,000 and β = 0.3) (◊), and SS-MS (N=10,000) (o).

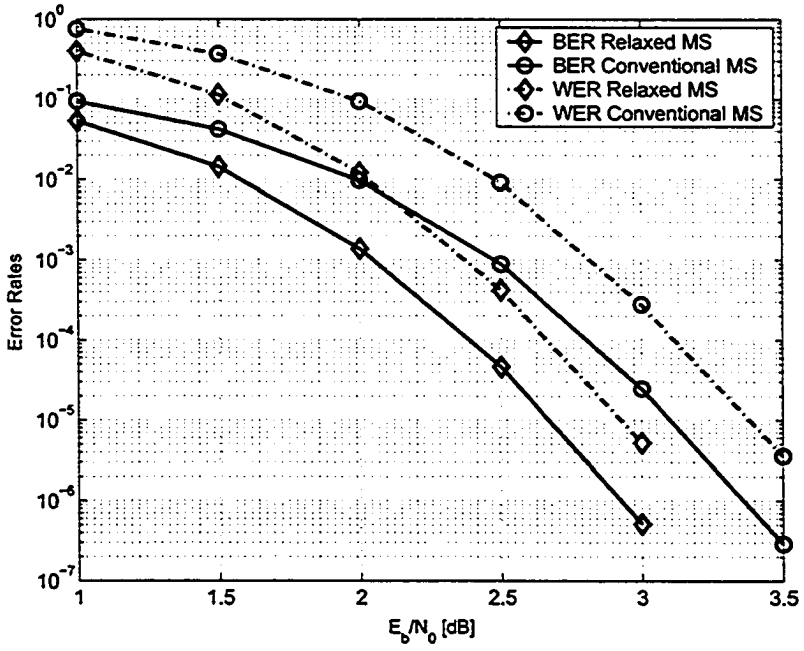


Figure 3-15 BER (___) and WER (_.._) curves for (504,252) LDPC code SR-MS (N=10,000 and β = 0.4) (◊), and SS-MS (N=10,000) (o).

It is important to note that in our simulations all the errors were detectable, and the performance improvement of SR over SS at a given E_b/N_0 is due to the higher rate of convergence to the right codewords.

3.5.3 Convergence Speed and Throughput of Analog Decoders

In this part, the SR method is used for studying the convergence speed of analog decoders and the relationship between the throughput and error correcting performance of analog decoders is investigated. Also, it is shown that average settling time in an analog decoder is a function of code, decoding algorithm, and the average time constant of the decoders' interconnections. This method can be used for selecting suitable codes and decoding algorithms for implementing analog decoders. Furthermore, it can be used for predicting the throughput of an analog decoder during the design process, if the average time constant is known.

To realize the importance of this study, it should be noted that it is impractical to determine error correcting performance of an analog iterative decoder for different throughputs by means of transistor level simulations as such simulations require extraordinary computational power [56]. In [57] the importance sampling method was used for reducing the simulation complexity for estimating the error correcting performance of an analog decoder. Using this method, transistor level simulations are performed for a few hundred cases instead of millions, required normally. But even this small number of cases could be affordable only for small codes. At the same time simulation results based on conventional discrete-time decoding can not be applied, because analog decoding is performed in the continuous-time domain and has different dynamics. In fact, it has been hard or even impossible to design an analog iterative decoder for a specific application,

because there was no way of estimating the ultimate throughput and error correcting performance of an analog decoder chip before fabricating and testing it. Even for a fabricated chip, it was not easy to choose the most appropriate decoding time and making a suitable tradeoff between the throughput and error correcting performance. Furthermore, it was not known which parameters could change throughput of an analog decoder and if there is any way to increase the throughput by proper selection of the code and the decoding algorithm.

In the following, we analyze the average settling time of analog decoders by first simulating (3-6) when all the time constants (delays) are equal, and then we investigate the effect of having different distributions for time constants.

We simulate (3-6) for two different relaxation factors, and choose a small time step to improve the accuracy of the analysis. In Figure 3-16, word error rate curves of an analog BP decoder versus maximum decoding time are presented for (1268,456) LDPC code. Similar curves for (504,252) LDPC code are shown in Figure 3-17. These figures show that as the relaxation factor decreases, error correcting performance improves, which is in agreement with previous observation in this chapter. Also, it is shown that by increasing the maximum decoding time, at first word error rate performance improves very fast, but gradually the rate of improvement decreases, and finally the WER curve saturates and it slowly converges to its asymptotic value. In other words, these figures illustrates how throughput and error correcting performance of the analog BP decoders are related to each other and how by improving one of them, the other one will decline. BER curves have similar trends and thus are not reported.

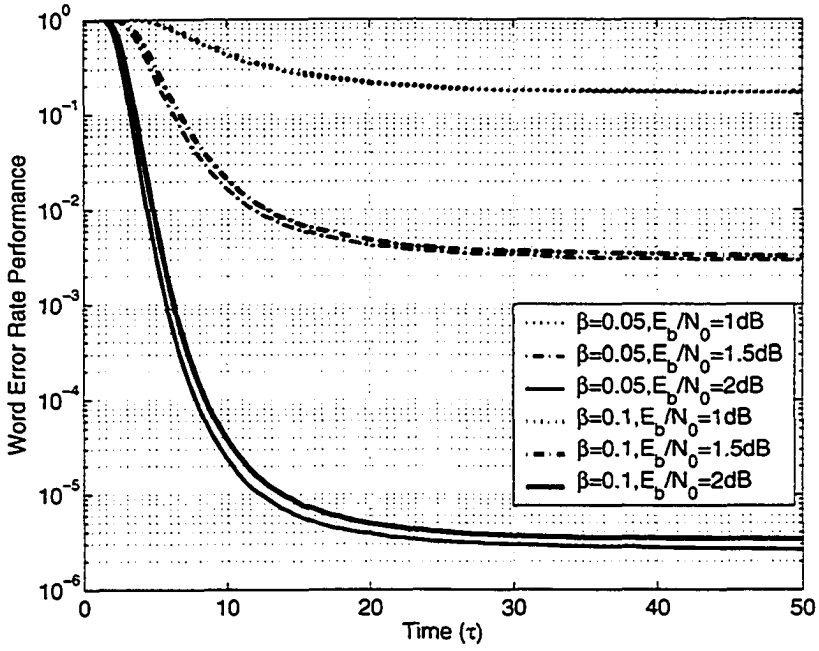


Figure 3-16 WER versus decoding time for (1268,456) LDPC code, decoded by SR-BP decoder with constant delay.

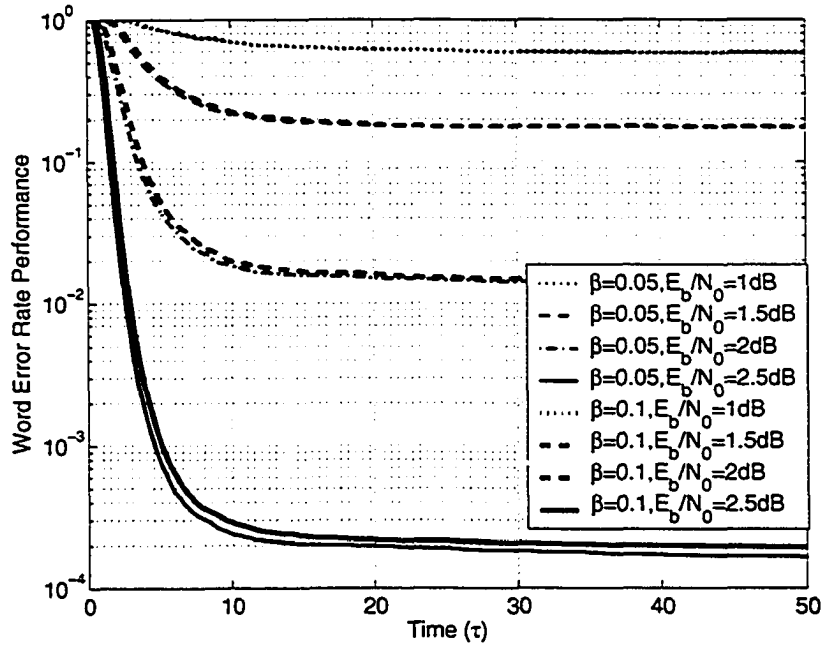


Figure 3-17 WER versus decoding time for (504,252) LDPC code, decoded by SR-BP decoder with constant delay.

A rather fuzzy choice for assigning the maximum decoding time for the analog decoder, can be considered as the time when WER curve is about to converge to its asymptotic value. In Figure 3-16 it is observed that for the (1268,456) LDPC code after about 20τ, the decoding process can be stopped and the loss in the coding gain will be insignificant. Figure 3-17 shows that for the (504,252) LDPC code the decoding process can be terminated earlier and only at about 10τ. Therefore, the throughput of the analog BP decoder for the (1268,456) code and the (504,252) code will be equal to $1268/20\tau=6.34/\tau$ (b/s) and $504/10\tau=5.04/\tau$ (b/s), respectively. This shows that if time constants are the same, the throughput for the (1268,456) code is more than that of the (504,252) code. However, from a practical point of view, a larger decoder consumes more silicon area and the capacitive and the resistive effects of the interconnections will be higher and therefore the time constants are larger. This implies that the ultimate throughput of the decoder for the (1268,456) code can be even less than that of the (504,252) code. Figure 3-18 and Figure 3-19 show similar curves when an analog MS decoder is used. Interestingly, the slope of the WER curves is higher in the saturation region compared to the simulation results for BP analog decoders. Also, the MS decoder settles slower on average compared to what was observed for the analog BP decoder.

To study the throughput of analog decoders with randomly distributed time constants, we use (3-6) for an analog BP decoder and assume the same distribution of time constants as in section 3.5.1. Figure 3-20 and Figure 3-21 show WER curves for this scenario for (1268,456) LDPC code and (504,252) LDPC code and illustrate that for different delay distributions the difference among average settling times is insignificant and by increasing the maximum decoding time, this difference becomes smaller.

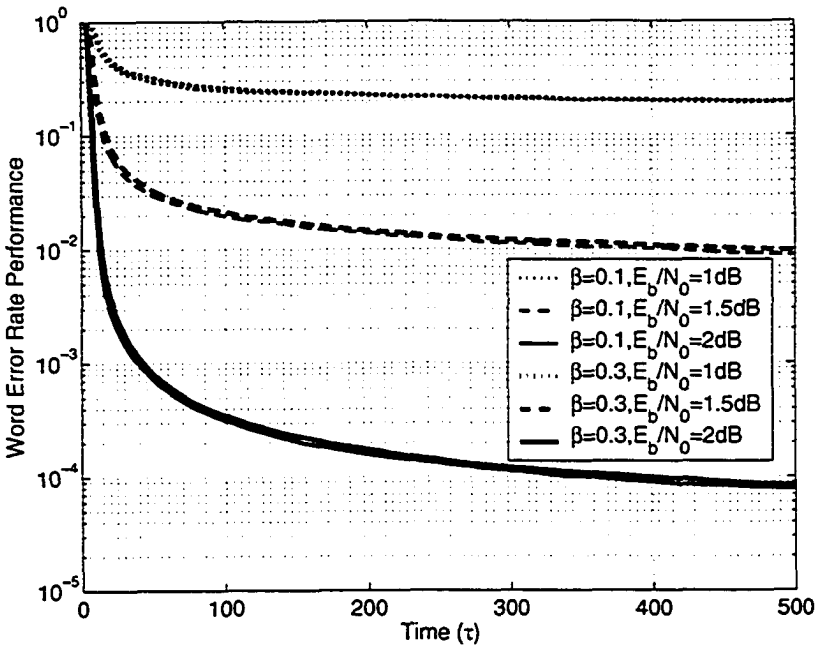


Figure 3-18 WER versus decoding time for (1268,456) LDPC code, decoded by SR-MS decoder with constant delay.

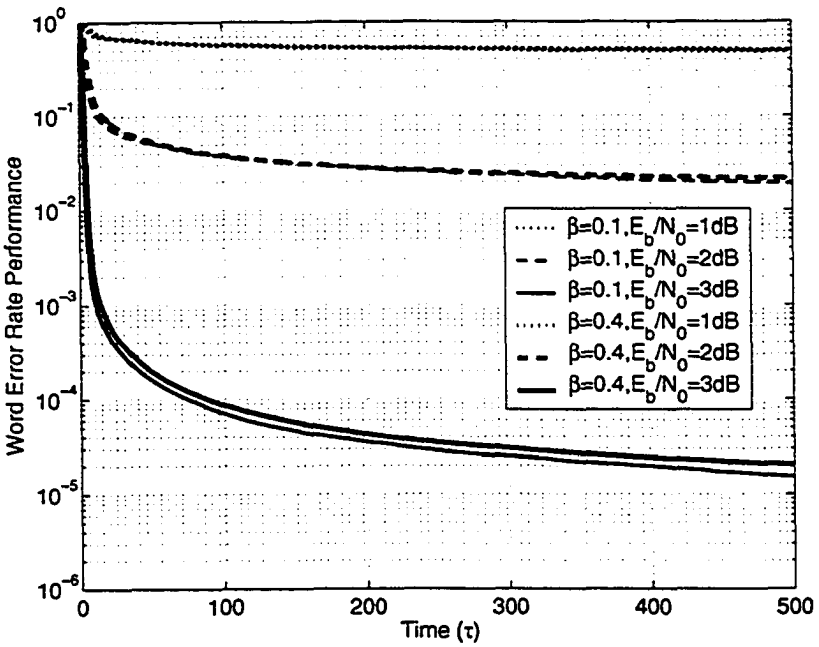


Figure 3-19 WER versus decoding time for (504,252) LDPC code, decoded by SR-MS decoder with constant delay.

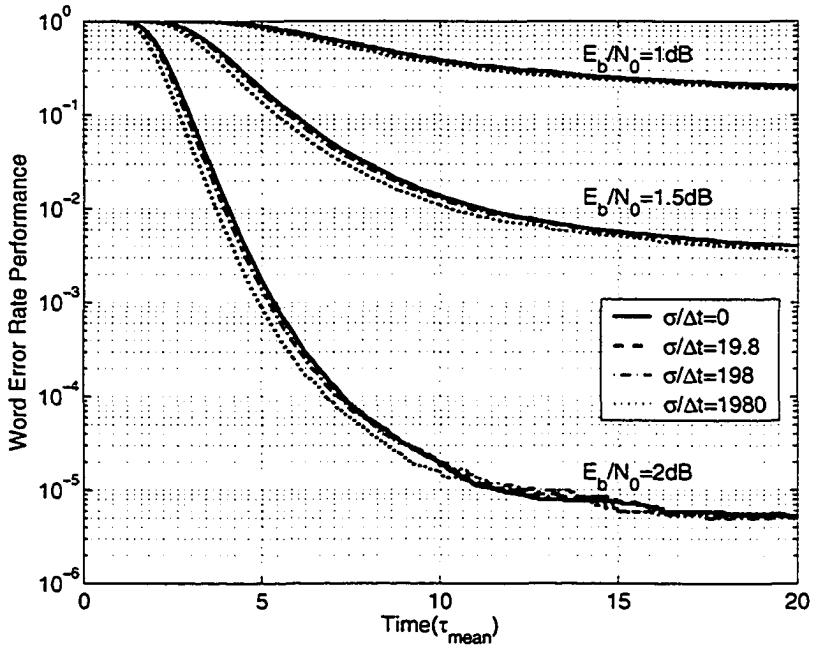


Figure 3-20 WER versus decoding time for (1268,456) LDPC code, decoded by analog BP decoders with random delays.

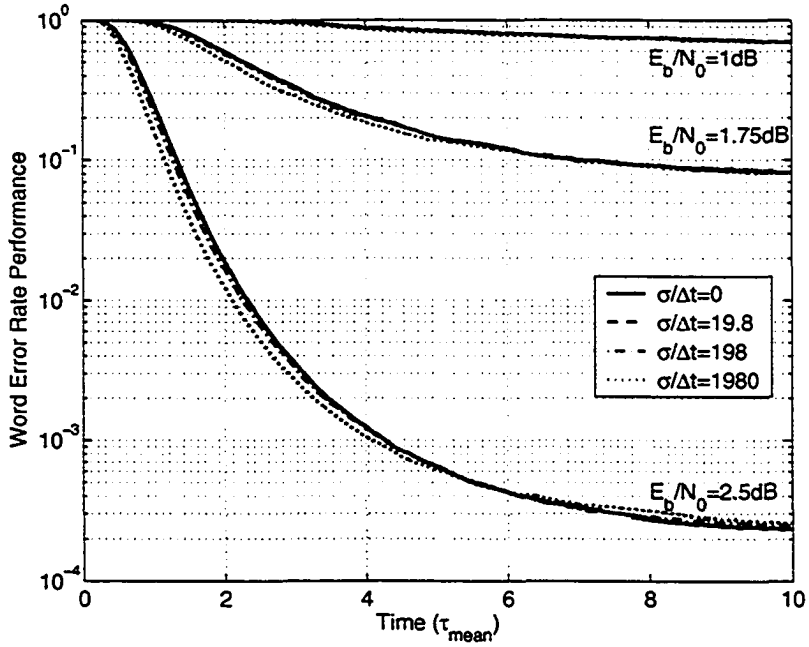


Figure 3-21 WER versus decoding time for (504,252) LDPC code, decoded by analog BP decoders with random delays.

3.6 Conclusion

In this chapter, we modeled continuous-time asynchronous analog iterative decoding by a set of first-order differential equations. Using this model, we approximated analog decoding as the numerical method of "successive relaxation" applied to the fixed-point problem of iterative decoding. We then showed that successive relaxation provides considerable performance improvement over the conventional numerical approach of successive substitution (associated with conventional iterative decoding on a discrete-time synchronous machine). For the tested LDPC codes, improvements up to 0.7dB in E_b/N_0 for a given BER are observed. The improvement increases, and the approximation error for the estimation of the performance of analog decoding decreases, as the maximum number of iterations tends to infinity and as the relaxation factor goes to zero. This means that analog decoding can not only increase the ratio of speed to power consumption but also provide a better performance compared to conventional digital decoding.

Our results show that the performance of analog decoding, particularly in steady-state, is rather independent of the distribution of delays among the computational modules. This is based on the assumption of truncated Gaussian distribution for delays.

In our discussions, we assumed that the analog circuits are ideal in the sense that they have no mismatch and enjoy unlimited accuracy and dynamic range. One important contribution of this work is thus to provide an "ideal" analog decoding benchmark. One can then compare the measured performance of analog decoders against this benchmark, instead of the commonly used synchronous discrete-time benchmark.

In addition to the analysis of analog decoding, our work also suggests a general framework for improving iterative decoding algorithms on graphs with cycles. On such graphs successive substitution does not necessarily converge to the optimal maximum a posteriori or maximum-likelihood solution and the application of successive relaxation

with $\beta < 1$ on a synchronous discrete-time platform can improve the performance. Our results of successive relaxation for the tested codes indicate that for a given maximum number of iterations, choices of $\beta < 1$ exist that improve the performance of conventional iterative decoding ($\beta = 1$) with smaller or almost the same average number of iterations. As an example, for the tested (1268,456) code and belief propagation in LR domain with N=200, by selecting $\beta = 0.6$, about 0.4dB improvement in E_b/N_0 at BER of about 10^{-6} is obtained with about the same average number of iterations. For the same code with min-sum in LLR domain and N = 200, the choice of $\beta = 0.9$ results in about 0.35 dB improvement in E_b/N_0 at BER of about 10^{-5} with the average number of iterations reduced by a factor of about 1.2.

Furthermore, we investigated the relationship between throughput and error correcting performance of an analog decoder. A simple method for making a tradeoff between throughput and error correcting performance was introduced and it was shown that delay distribution has insignificant effect on the average settling time of analog decoders. Also, it was shown that settling behavior is a function of both the code and the decoding algorithm. This approach introduces a new method for predicting throughput of analog decoders during the design process, if the average time constant of the analog decoder is known. Average time constant can be estimated by circuit simulation in a low signal to noise ratio for a relatively small number of cases. Also, this study can be used for selection of suitable codes for analog decoders.

Chapter 4

Analog Min-Sum Iterative Decoder

In this chapter we focus on implementation issues of iterative decoders. We start this chapter by investigating pros and cons of using an analog platform for implementing iterative decoders. Then after a short review of the previously reported analog iterative decoders, a current-mode approach is presented for implementing basic building blocks of an analog iterative decoder. The decoder is based on the so-called min-sum algorithm and can be used to decode powerful coding schemes such as low-density parity-check (LDPC) codes and turbo codes. The proposed circuits can be implemented by standard CMOS technology, which means lower fabrication cost or faster circuits compared to previously reported analog iterative decoders that are based on BiCMOS or subthreshold CMOS technology, respectively.

4.1 Analog or Digital?

As we saw earlier in this thesis, iterative decoding algorithms can be used for decoding the best known channel coding schemes. We also saw how LDPC codes were forgotten for over 30 years because of their decoding complexity. Needless to say, these ca-

pacity achieving codes can not be widely deployed in communication and storage systems unless the existing problems in hardware implementation are addressed properly.

In iterative decoding, numerous processing modules perform floating-point computations simultaneously and communicate with each other based on the graphical representation of the code. Fortunately, the parity-check matrix for LDPC codes is sparse by definition and the number of edges is only a few times larger than the block length. The number of real physical interconnections is however, at least two times higher than the number of edges in the Tanner graph representation of the code. This is because physical interconnections are unilateral but edges in the graph are bilateral and conduct belief messages in two directions. The number of interconnections in a conventional digital approach would be multiplied by the number of bits used to represent each message. As an example, in [18] for a (1024, 512) irregular LDPC code that has 3328 edges in its Tanner graph, each message is represented by four bits and consequently 26624 physical interconnections have been used. This huge number of interconnections generates a huge capacitive load especially because on average, the length of each interconnection in the reported digital decoder chip is 3mm. Interconnections consumed 50% of this 52mm^2 chip, which was fabricated in 0.16μm five-layer metal CMOS technology. In low signal to noise ratios (SNR=0dB) when more errors are likely to happen, the average number of iterations is high. This increases the switching activity [10] of the decoder and power consumption will be about 2W. This is while in higher signal to noise ratios (SNR=3dB) and for the same throughput, the switching activity significantly decreases and the power consumption is reduced to 0.69W. The coded throughput for this chip was reported 1Gbps (data throughput 500Mbps) and the loss of coding gain due to limited accuracy in

the processing modules has been about 0.2dB. It is worth noting that by hardware sharing, it is possible to substantially reduce the area and power consumption in a digital approach, but it would be at the expense of lowering the throughput. In this method, a small number of processing modules are implemented and a control unit utilizes these processing modules sequentially for updating a fraction of stored messages in each clock pulse. Sharing the processing modules reduces that overall throughput but it could be more affordable compared to a fully-parallel decoder chip. For example in [58] a digital reconfigurable decoder chip was fabricated in a 0.18μm six-layer metal CMOS technology that has an active area of $9mm^2$. This decoder chip can be configured for different 3GPP service combinations that include different bit rates for data and voice communication. The power consumption is 0.29W when turbo decoding a 2Mb/s data stream with ten iterations per block. Rate of the code is 1/3 and its constraint length is 4.

The analog approach can be more favorable in terms of the number of the physical interconnections because each massage can be represented by a current or a voltage in a wire in single-ended signaling. For differential signaling (when a message is represented by the difference of two voltages or currents), we need only two wires per message. Since voltage swing in an analog approach is smaller than that of a digital approach, dynamic power consumption per interconnection could be smaller. As the number of interconnections, average length of interconnection, and the power consumption in each interconnection could be smaller in an analog approach, we conclude power consumption and area consumption in the wiring could be considerably lower than a digital approach. Also in the absence of a clock signal which normally charges a huge capacitive load in each clock pulse, more power is saved.

In [18] seven million transistors have been utilized for implementing the check nodes and the variable nodes. These nodes have been significantly simplified by limiting the resolution of the messages and replacing complex computational blocks by simple look-up table operations that were implemented by combinational circuits. If higher resolution is needed, then we would need a few times more transistors. At each clock edge, many transistors switch and subsequently an impulsive current is drawn from the power supply or injected into the ground [10]. In addition to consuming power, this impulsive current could generate lots of noise and disturbance in the chip. As an example, one of the noise generating mechanisms in digital circuits is illustrated in Figure 4-1 for a simple NOT gate where the limited slope of the input signal keeps both n-type and p-type MOS transistors on and causes the power supply and ground to be short circuited for a short period of time [10].

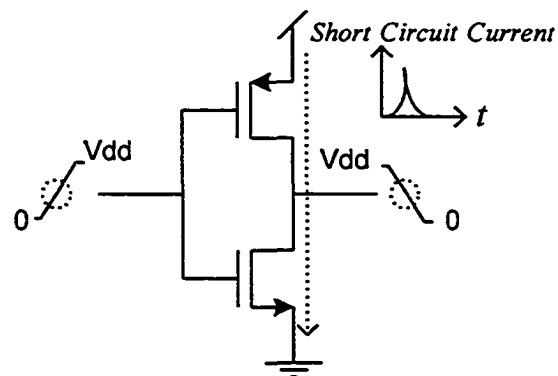


Figure 4-1 Short circuit current in a simple NOT gate.

This could generate lots of noise on the supply rails. These disturbances can deteriorate performance of noise sensitive blocks that are widely used in the receivers [9].

In an analog approach, transistors are not biased as simple switches and by using their characteristics, we may be able to implement a function with substantially fewer

transistors. To achieve this goal, we should exploit analog circuits and efficiently implement the required operations. As we will see later in this chapter, the Gilbert differential multiplier [59] was first found to be very useful for implementing belief propagation algorithm and we will show how minimum winner-take-all circuits and current mirrors would pave the way for efficiently implementing the min-sum decoding algorithm. These modules consume less power and area compared to their digital counterparts. As a simple example, a 4-input adder in an analog circuit can be simply designed by four current sources connected to each other as shown in Figure 4-2. This adder can be implemented with no more than a few tens of transistors. However, we need tens to hundreds of transistors, depending on the number of bits per message, for a 4-input digital adder shown in Figure 4-3. It is interesting to note that while the number of inputs in the analog adder can be increased easily, for the digital approach, in addition to an increase in the number of transistors, it would increase the latency of the adder and makes it slower as we need to use more layers of adders. Nevertheless, one should not overestimate the saving in silicon area because transistors in digital circuits are often smaller.

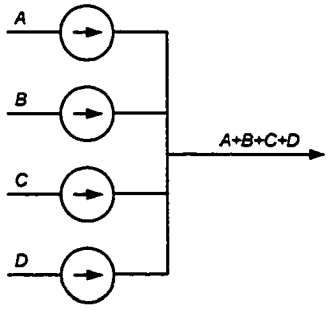


Figure 4-2 A simple 4-input current-mode analog adder.

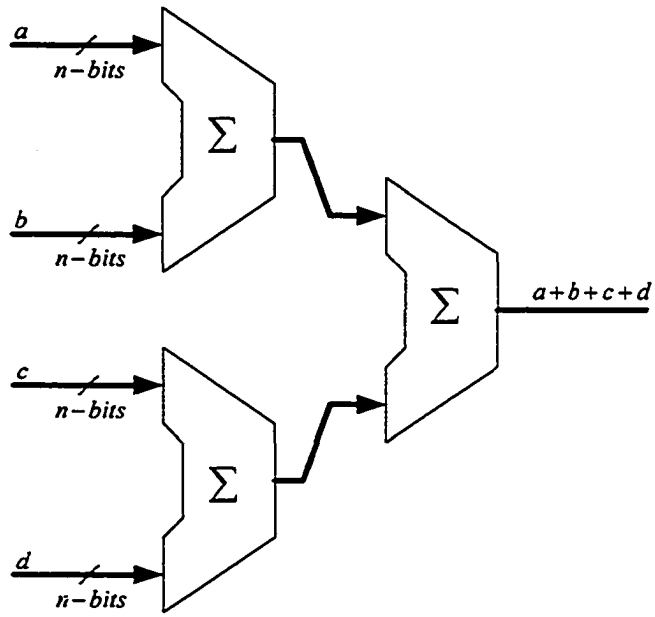


Figure 4-3 A Simple 4-input digital adder can be realized by three two-input full adders.

Analog circuits generate less noise and we do not observe impulsive short circuit currents similar to those that exist in conventional digital circuits. This is one of the advantages of the analog approach that has not been exploited fully. However, in a system-on-chip or highly integrated communication systems, this feature is greatly welcomed.

As shown in the previous chapter, analog decoders enjoy better dynamics and if imperfections and mismatch problems do not degrade the precision of the computational modules, we can observe better error correcting performance. It is however, possible for digital decoders to mimic the dynamics of analog decoders, but as shown earlier, it can increase the average number of iterations required for convergence to the correct codeword and ultimately decreases the throughput.

The fact that in the absence of any controlling signal, the state of an analog decoder naturally moves towards the most likely transmitted codeword in most cases and

ultimately supersedes the error correcting performance of conventional discrete-time decoders, looks very similar to the movement in the steepest descent direction that happens in nature, as shown by a simple example in Figure 4-4. In the discrete-time domain, due to large step-size in a discrete-time trajectory, the right fixed-point could be missed and ultimately the chance of convergence to the correct solution can be smaller. This is a very important observation that could have great impacts beyond coding theory as iterative methods in general and belief propagation in particular, have many applications in different fields [60].

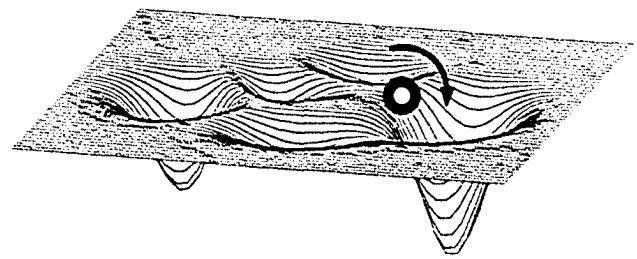


Figure 4-4 A ball naturally rolls in the direction with the greatest downhill gradient.

While the above advantages motivate research work on the theory and implementation of analog iterative decoders, there are many challenges yet to overcome. Design and implementation of digital circuits is rather straightforward and their behavior is more predictable. Digital circuits have proven in practice to be very reliable and robust. For instance, when we compare the area and power consumption or number of the transistors in 4-input adders in Figure 4-2 and Figure 4-3, we should also note that for the analog adder, the accuracy could change easily for different current levels and the performance might significantly change with temperature variations or fabrication imperfections. The

digital adder however is not that sensitive to these parameters. On the other hand, numerous design tools, cell libraries, and testing facilities are available that help the designers to map any complicated algorithm onto digital gates and fabricate integrated circuits with high chance of success.

At the present, however analog design is performed manually and it is very time consuming and error prone. After all, analog circuits have not been traditionally used for highly integrated circuits and even the models are not accurate enough to provide very reliable simulation results. In [15], [56], and [61] the problem of layout design automation for large analog iterative decoders has been addressed. Without design automation, it is not possible to design and implement analog decoder chips for codes that are large enough to be used in real industrial applications.

Also, as the error correcting performance of a decoder should be verified statistically for a large number of inputs, the design process could be very difficult. In fact, it is not practical to use circuit simulators to verify how changing a parameter could change the ultimate error correcting performance of the decoder. In analog circuits, it is possible to improve the accuracy of the processing modules but generally it comes at some cost and it is necessary to make a trade off among the parameters that are involved. However, transistor level simulation even for one test could take several days and it is impossible to run accurate transistor level simulation for millions of cases to be able to obtain the BER curve of the decoder. Without any transistor level simulation for predicting the BER curve, we often have no choice but to do a blind optimization. For instance in the fabricated analog decoder chips, designers have made different assumptions for the transistor sizing, often without solid justification or supporting simulation. There have been some

attempts to estimate the performance of analog iterative decoders by the importance sampling method [57], this method allows a reduction in the number of required simulation points from millions to a few hundred. However, even for a small decoder simulation takes too much time and for example for a (32,8) LDPC code that has been implemented by about 18,800 transistors, simulating one case takes about 30 hours and doing simulation for a few hundred cases is still unaffordable. In contrast, in a digital approach after deciding on the number of bits for representing the messages and the number of bits in computational modules, high-level simulation can predict the performance very fast, even for very large decoders.

Dealing with mismatch problem is known to be problematic for analog chips in general and analog iterative decoders in particular. While we might be able to improve matching locally by proper transistor sizing and by being cautious in layout design, for a large analog chip, the global matching could be poor. The effect of mismatch on the performance of analog iterative decoders was first studied in [62] and later followed by [56], [26], and [28]. More recently in [63] for codes with infinite length the asymptotic effect of transistor mismatch was studied. All these studies confirm that mismatch can degrade the performance of analog iterative decoders. In [26] it was shown that if both global and local mismatch are present, longer codes would be more susceptible to mismatch and a flattening effect in error ceiling is observed. Other studies predict a few tenths of a dB degradation in the error correcting performance of the decoder for typical mismatch values [28], [56], [62], and [63].

Testability is another problem that should be addressed before the mass production of analog decoders can become a reality. This could be a big problem because itera-

tive decoders are relatively fault tolerant and a faulty chip can only be detected by observing degradation in its error rate curves. This is because an iterative decoder can remain functional even if a few interconnections are missing or some transistors are slower than required. Therefore to verify the functionality of a decoder chip and see if it meets the desirable specifications (in terms of error rate curves at a particular throughput), its error correcting performance should be tested statistically. However, performing statistical tests for a large number of decoder chips could be costly. In the digital approach it is possible to generate suitable test patterns to detect any malfunction in the chip. Testing can even be performed by particular on-chip circuits.

It is quite advantageous if the received information from the channel can be stored in analog memories. This will save power and limit further degradation in the received information. However, numerous high speed sample and hold (S/H) blocks would be necessary [64]. Also making non-volatile analog memories could make the design more difficult [65]. In the absence of analog memories, we need to convert input signals into digital form by means of ADC to store the information on digital memories and later use digital-to-analog converters (DAC) to regenerate the received information in analog form. This deteriorates the quality of the received word by introducing more errors.

In brief, there are many advantages in implementing iterative decoding by analog continuous-time circuits, however this is a challenging and ambitious goal and there are many hurdles yet to overcome. In the next section, we look at the previously reported analog decoders. While there has been a significant amount of work on implementing analog channel decoders in general, and specially on analog Viterbi decoder [66]-[69] ,

we limit our discussion to belief propagation and min-sum iterative decoding algorithms that can be used for decoding capacity achieving codes.

4.2 Belief Propagation Analog Iterative Decoders

Recalling (2-8) and (2-10), BP requires basic operations of addition and multiplication of real numbers. If we assume messages are represented in the probability domain then outgoing message (Z) of a check node for only two incoming messages X and Y can be calculated similar to (2-8) as follows [60]:

$$\begin{cases} P(Z=0 \mid X,Y) = p_X p_Y + q_X q_Y \\ P(Z=1 \mid X,Y) = q_X p_Y + p_X q_Y \end{cases} \tag{4-1}$$

Or

$$\left(P(Z=0 \mid X,Y) - P(Z=1 \mid X,Y)\right) = \left(p_X - q_X\right)\left(p_Y - q_Y\right). \tag{4-2}$$

Similarly based on (2-10) the outgoing message (Z) of a variable node for only two incoming messages X and Y can be obtained by:

$$\begin{cases} P(Z=0 \mid X,Y) = \gamma\left(p_X p_Y\right) \\ P(Z=1 \mid X,Y) = \gamma\left(q_X q_Y\right) \end{cases} \tag{4-3}$$

Where γ is the normalization factor and can be defined as follows:

$$\gamma = \frac{1}{p_X p_Y + q_X q_Y}.$$

In [15], (4-1) and (4-3) have been used for introducing soft XOR and soft AND gates, respectively. These two-input soft gates are shown in Figure 4-5. In soft AND gates, normalization factor γ has been ignored, and as we will see later normalization is automatically performed in the proposed circuits and it is possible to ignore this normali-

zation factor as long as outputs of soft AND gates are connected to other soft gates or are interpreted carefully.

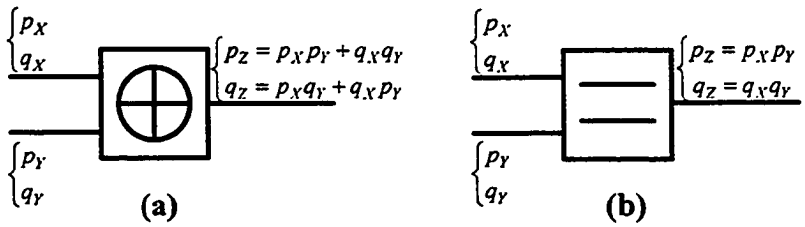


Figure 4-5 Soft gates: (a) Soft XOR gate, (b) Soft AND gate.

Soft gates with more than two inputs can be simply constructed by properly cascading these two-input gates. For instance a 5-input soft XOR is shown in Figure 4-6. In general, for implementing an N-input soft gate, we need N-1 two-input soft gates. These gates can be used for implementing check nodes and variable nodes of any degrees. It is worth noting that a check node of degree six can be implemented by six 5-input soft XOR gates. This is because each of the edges should be updated separately based on the messages that the check node has received from the variable nodes. Equation (4-2) shows that a differential multiplier can be used for implementing a soft XOR. If we discard two terms in (4-1), we can also implement the soft AND gate. As we will see later in this section, a Gilbert differential multiplier [59] can be used for implementing both of these soft gates.

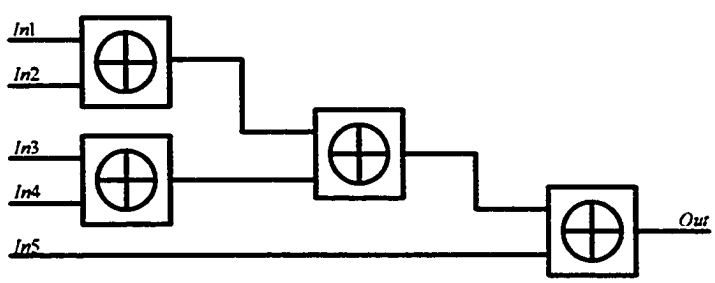


Figure 4-6 A 5-input soft XOR.

A basic building block in the Gilbert differential multiplier is shown in Figure 4-7. As we see in the following, this block splits the fixed current I into two currents that are proportional to p and q. In Figure 4-7, we have:

$$V_{BE_1} + V_{BE_3} = V_{BE_2} + V_{BE_4}, \tag{4-4}$$

where V_{BE} is the base-emitter voltage. But for bipolar transistors, the following equation holds:

$$I_C = I_S\left(1+\frac{V_{CE}}{V_A}\right)e^{\frac{V_{BE}}{V_{Th}}}, \tag{4-5}$$

where I_C is the collector current, I_S is a constant current with typical values between 10^{-14}-10^{-16}A, V_{CE} is the collector-emitter voltage, V_A is the Early voltage with typical values between 15-100V, and V_{Th} is the thermal voltage and is equal to 26mV at room temperature [70]-[71]. We can write:

$$I_C = I_S e^{\frac{V_{BE}}{V_{Th}}}. \tag{4-6}$$

When V_{CE} is much smaller than the Early voltage. By substituting (4-6) in (4-4), we get:

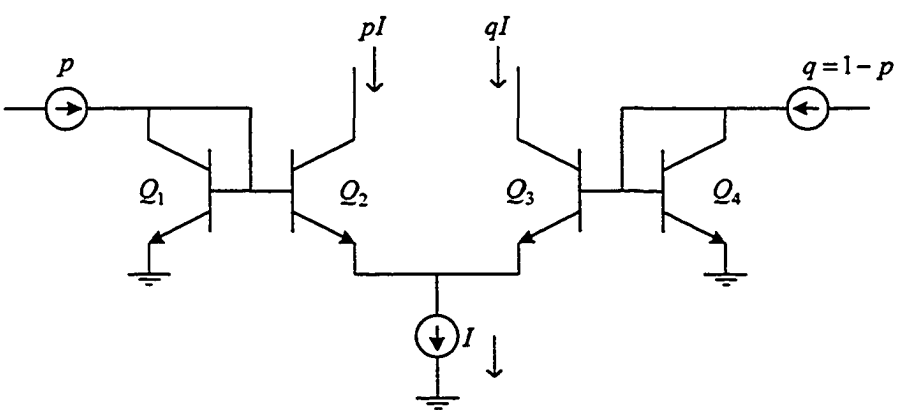


Figure 4-7 A modified emitter coupled circuit acts as a *pq* splitter.

$$V_{Th}\ln\left(\frac{I_{C_1}I_{C_3}}{{I_S}^2}\right)=V_{Th}\ln\left(\frac{I_{C_2}I_{C_4}}{{I_S}^2}\right). \tag{4-7}$$

By ignoring the base currents, we can write:

$$\begin{cases} I_{C_1}=p \\ I_{C_4}=q \\ I_{C_2}+I_{C_3}=I \end{cases} \tag{4-8}$$

Consequently, we have:

$$\begin{cases} pI_{C_3}=qI_{C_2} \\ I_{C_3}+I_{C_2}=I \end{cases} \Rightarrow I_{C_2}=\frac{pI}{p+q}=pI, I_{C_3}=\frac{qI}{p+q}=qI \tag{4-9}$$

This shows that circuit in Figure 4-7 approximately splits I between Q_2 and Q_3 proportional to p and q, respectively. We thus call this circuit a *pq* splitter. It is worth noting that even if the input currents p and q have not already been normalized, the output currents will still be proportionally divided corresponding to their normalized values. This is the reason why the normalization factor in soft AND gates can be ignored and why we do not force current sources to be equal to one. Figure 4-8 shows a symbol for a *pq* splitter.

Figure 4-9 shows how by using *pq* splitters, we can implement soft XOR and soft AND gates defined in Figure 4-5. In a soft XOR gate, currents from different branches are simply added together based on Kirchoff's current law. It is easy to recognize that the soft XOR gate is the well-known Gilbert differential multiplier that has numerous applications in electronic circuits [70]-[71]. Soft AND gate can also be considered as a simplified Gilbert multiplier.

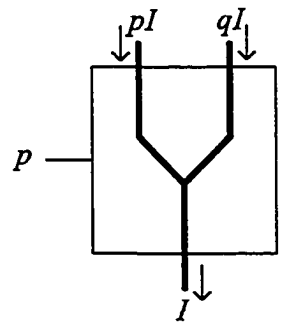


Figure 4-8 Symbol of a *pq* splitter.

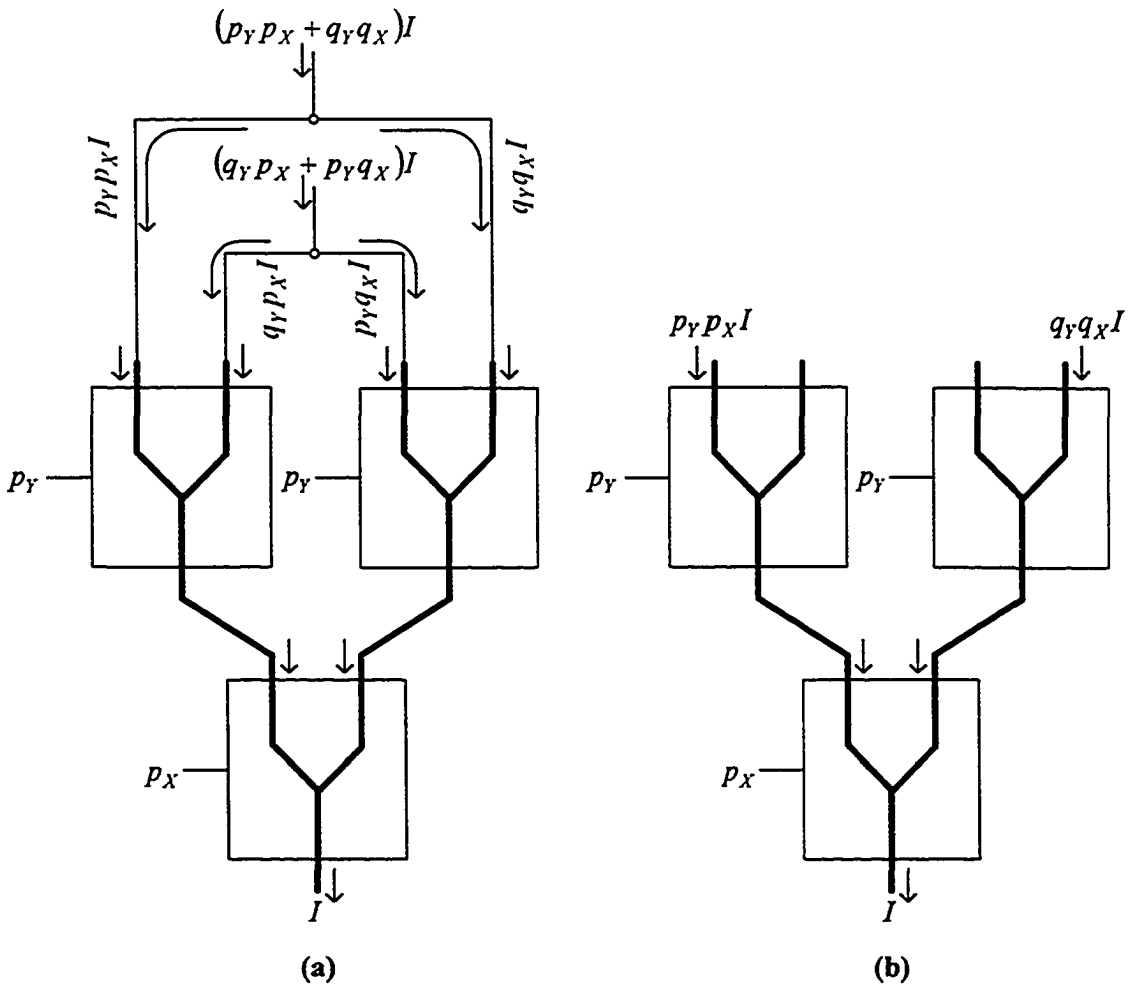


Figure 4-9 Soft gates implementation based on *pq* splitter: (a) Soft XOR gate, (b) Soft AND gate.

This approach for implementing belief propagation was first used by Hagenauer [19]-[21], [72]-[79] and Loeliger [23], [62], [80]-[85]. While there are minor differences in their proposed circuits and the former approach is voltage-mode and the latter is current-mode, both approaches use variants of the Gilbert differential multiplier. In their original forms, these circuits are suitable when bipolar junction transistors (BJT) are used and the transistor characteristic is described by (4-6).

Bipolar technology is suitable for high-speed applications, but it makes the fabrication process expensive and circuits consume more area compared to CMOS circuits. Static power consumption in base-emitter junction in bipolar transistors can also increase the static power consumption. To handle these problems for large analog decoders in [79] hardware sharing was proposed. Soon, it was found that by using subthreshold (weak inversion biasing) CMOS Gilbert multipliers it is possible to implement belief propagation with CMOS circuits[15], [25]-[29], [85]. A CMOS design is more favorable because it is more affordable and consumes less area. Specifically, subthreshold MOS can be used in extremely low power applications at relatively low frequencies and has been used for quite a while for designing analog artificial neural networks (see, e.g., [9], [11], [70], [86]-[88]).

The operation of a subthreshold MOSFET can be described by the following equation:

$$I_D = I_0 \frac{W}{L} e^{\kappa(V_{GS}/V_{Th})} e^{(1-\kappa)(V_{BS}/V_{Th})} \left(1 - e^{-V_{DS}/V_{Th}}\right) \tag{4-10}$$

where I_D is the drain current, V_{GS} is the gate-source voltage, V_{BS} is bulk-source voltage, V_{DS} is drain-source voltage, V_{Th} is the thermal voltage, I_0 is a constant, W is transistor width and L is its length, and κ is a technology-dependent positive parameter [87]. In sub-

threshold MOSFET, V_{GS} would not exceed the threshold voltage and virtually the MOSFET is off and drain current is very small. If bulk and substrate are short circuited and V_{DS} is large enough then we can simplify (4-10) and get the following equation, which is very similar to (4-6).

$$I_D = I_0 \frac{W}{L} e^{\kappa(V_{GS}/V_{Th})} \tag{4-11}$$

This means that soft XOR and soft AND gates can be implemented by using subthreshold MOS transistors very similar to what was proposed for BJTs.

Interestingly, we found that linear Y-fed optical directional couplers can be used for implementing very high speed electro-optical *pq* splitters [89], [90]. This shows the feasibility of implementing very high-speed soft XOR and soft AND gates and ultimately very fast BP decoders. Although based on the current state of integration in electro-optical devices, this approach is very expensive; it could be used in ultra-fast optical links or in the future when these high speed devices become more affordable.

We end this section by mentioning that soft XOR and soft AND gates are the only required processing modules for implementing belief propagation algorithm for an LDPC code. For turbo codes however we need analog interleavers as well [91]. As this thesis has targeted the implementation of LDPC codes, we will not discuss the implementation issues of interleavers.

4.3 Min-Sum Analog Iterative Decoder

In the previous section we saw that belief propagation (BP), which is the best known iterative decoding algorithm, can be implemented by analog BiCMOS or sub-threshold CMOS circuits. BiCMOS circuits are costly and subthreshold circuits are rather

slow. To make high-speed analog iterative decoders more affordable, in this section, we show it is possible to use CMOS circuits for implementing high-speed analog iterative decoders. To do so, we focus on the implementation of the min-sum iterative decoding algorithm (MS) with favorable CMOS circuits in the strong inversion biasing condition. This means MOS transistors are completely turned on and therefore can be quite fast. For many codes, the performance of MS is slightly (a few tenths of a dB) worse than that of BP and as we observed in Chapter 3 it needs more iterations to settle down compared to BP. However, it has lower complexity, and unlike BP, does not require an estimate of noise power in the channel [41]. It has also been shown that MS is more robust against quantization noise than BP [41]. Moreover, there are simple modifications of MS that can perform very close to BP [41]-[43].

In addition, all modules in the previous designs are based on soft XOR and soft AND gates that are modified versions of the Gilbert multiplier, and therefore each block can only have two inputs. Modules with more than two inputs can then be constructed by cascading the two-input soft gates [15]. This increases the latency of the circuit. In the proposed circuits however, modules with larger number of inputs can be fabricated easily and increasing the number of inputs does not increase the delay as much.

In the rest of the section, we first explain the basic operation and structure of an MS decoder. We then propose current-mode CMOS circuits for implementing the computational modules of MS.

4.3.1 Basic Operations & Message Representation

Recalling Tanner graphs and operations in variable nodes and check nodes in the MS decoding algorithm from Chapter 2, we should implement equations (2-19), (2-20),

and (2-21), if messages are represented in the log-likelihood ratio domain. These equations are rewritten to facilitate referring to them, as follows:

$$m_{V\to C}^{(l)} = m_V + \sum_{C'\in N_V\setminus\{C\}} m_{C'\to V}^{(l-1)}. \tag{4-12}$$

$$m_{C\to V}^{(l)} = \left(\prod_{V'\in N_C\setminus\{V\}} sign\left(m_{V'\to C}^{(l)}\right)\right) \min_{V'\in N_C\setminus\{V\}} \left(\left|m_{V'\to C}^{(l)}\right|\right). \tag{4-13}$$

$$M_V^{(l)} = m_V + \sum_{C'\in N_V} m_{C'\to V}^{(l-1)}. \tag{4-14}$$

As can be seen in these equations, in order to implement the required operations in MS, we have to deal with the messages, their magnitudes, and their signs separately. We have plenty of options for representing messages and an analog value in a circuit can be represented by a current or a voltage, either single-ended or differentially or even by means of its absolute value and its sign and each has its own advantages and drawbacks.

The current-mode approach is often more attractive because of its speed and dynamic range. However, some operations fit more easily into voltage-mode circuits, for instance by wiring, we can apply the output of a voltage source into a few different circuits, but for duplicating the output of a current source, we need current mirror. In contrast, in current-mode approach, adding signals is much easier, as shown in Figure 4-2.

Differential signaling is preferable because it is more robust against common-mode noise and global mismatch, but needs twice as many interconnections. In iterative decoding the congestion problems are already severe because of numerous interconnections and so it is preferable to avoid making the routing problem more complicated. The differential approach would also double the capacitive load and increases area and power

consumption. On the other hand, single-ended approaches can be vulnerable against noise and mismatch. Furthermore, representing bipolar signals can be difficult. Figure 4-10 shows that when local mismatch is negligible, it is possible to mitigate global mismatch effects. Suppose we want to regenerate an input current I, in another location in the chip and ideally all our transistors have the same width and length. To do so we consider two scenarios: in Figure 4-10 (a) the input current is duplicated locally and is sent to the desired location in form of a single-ended current, where it is duplicated and used. In this scenario, global mismatch would not deteriorate the accuracy of our chip, though the size of the transistors could be different because of the global mismatch ($\frac{W}{L} \neq \frac{W'}{L'}$). In Figure 4-10 (b), the second scenario is illustrated when instead of a single-ended current; the corresponding single-ended voltage is conducted to the desired location in the chip. Because of the global mismatch transistors would not have the same size and the transferred single-ended voltage would be translated to I' current instead of I.

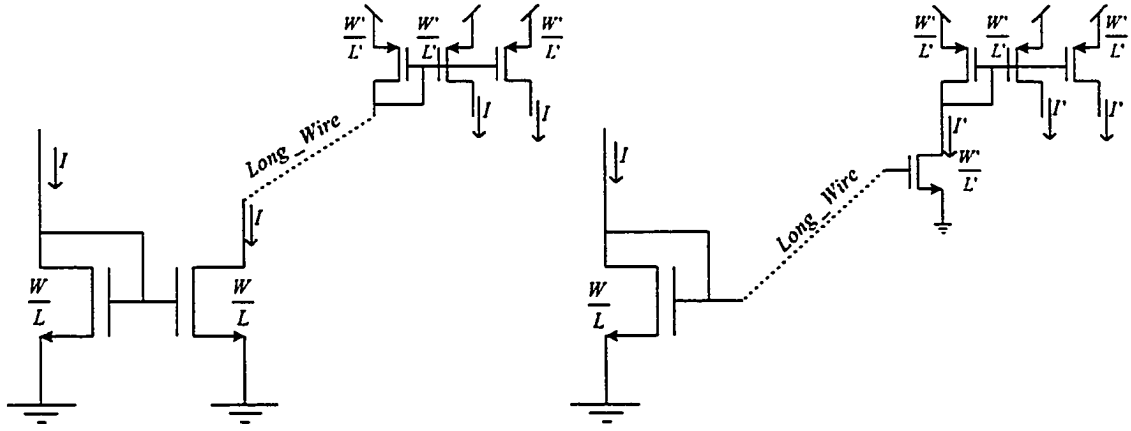


Figure 4-10 If transistors are locally matched, (a) in single-ended current-mode approach global mismatch is tolerated, (b) in single-ended voltage-mode approach global mismatch is not tolerated.

Representing signals by their magnitude and sign brings the digital signal path in close proximity of the analog signal path and ultimately increases the degradation due to noise. Also, if magnitude is represented by a voltage similar to Figure 4-10 (b), it could be sensitive to global mismatch. It also doubles the number of interconnections and can increase area and power consumption similar to the differential method.

In our proposed circuits, inside the variable nodes and for node-to-node interconnections, we use the single-ended current-mode approach, but in the check nodes we locally represent messages by their sign bit and absolute values. In this way, we can considerably simplify the necessary modules.

4.3.2 Implementation of Variable Nodes (Basic Modules)

In a variable node, incoming messages (*extrinsic information*) are added to the received information (*intrinsic information*) from the channel to generate the variable node's outputs. Messages are represented in the form of currents and can be either positive or negative. As mentioned in the previous section, in order to reduce the number of interconnections, we use the single-ended approach.

We define polarity of a current based on its direction. If it is drawn from power supply or p-type MOS transistors, we consider it negative and if it is going into ground or to n-type MOS transistors, we consider it a positive current. This definition is illustrated in Figure 4-11.

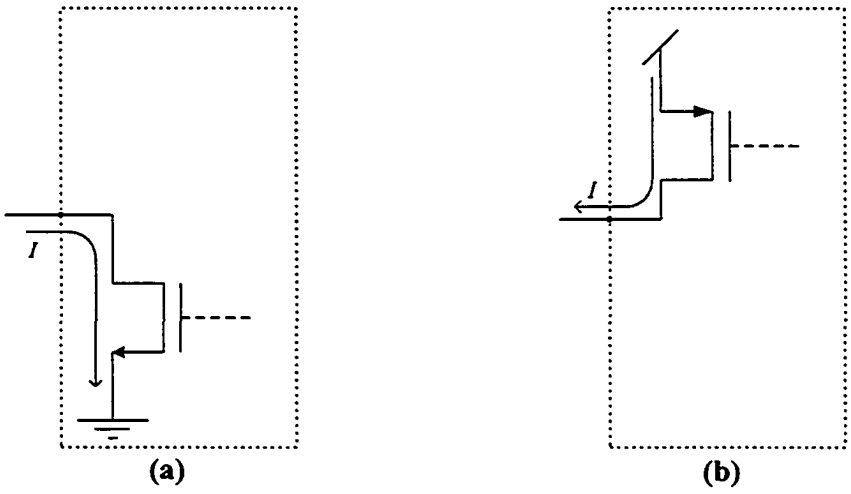


Figure 4-11 Definition of polarity for currents: (a) a positive current, (b) a negative current.

One of the most frequently used blocks in current-mode circuits are current mirrors or current conveyers [70]. Based on the above definition for polarity of currents, a current mirror duplicates the magnitude of the input current but flips its sign. Figure 4-12 shows how we represent current mirrors [70]. It is worth noting that n-type current mirrors can be used for positive currents and p-type current mirrors can be used for negative currents. Since our messages can be positive or negative, we need both types of current mirrors in our circuits. We call this combination a current buffer.

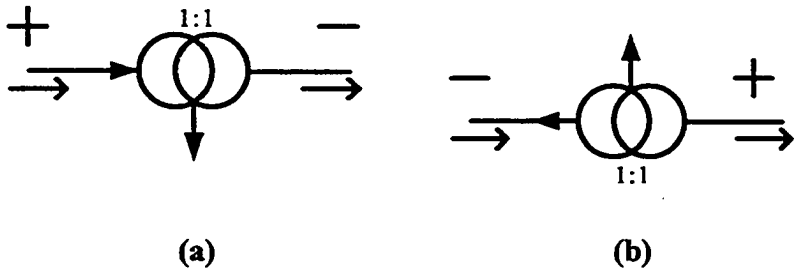


Figure 4-12 Current mirrors' symbol representations (a) n-type, (b) p-type.

Figure 4-13 shows how a current buffer can duplicate positive and negative currents. This figure also shows that a leakage current (I_{leak}) flows when we construct a current buffer in this way. While this leakage current does not degrade the accuracy of our current buffer in ideal case, it increases power consumption. Static power consumption (P_{leak}) because of this leakage current in the current buffer is equal to:

$$P_{leak} = 2 \times I_{leak} \times V_{dd}, \tag{4-15}$$

where V_{dd} is the supply voltage. To minimize this leakage current and reduce power consumption, suitable p-type and n-type current mirrors should be used that can not be on at the same time for the given supply voltage.

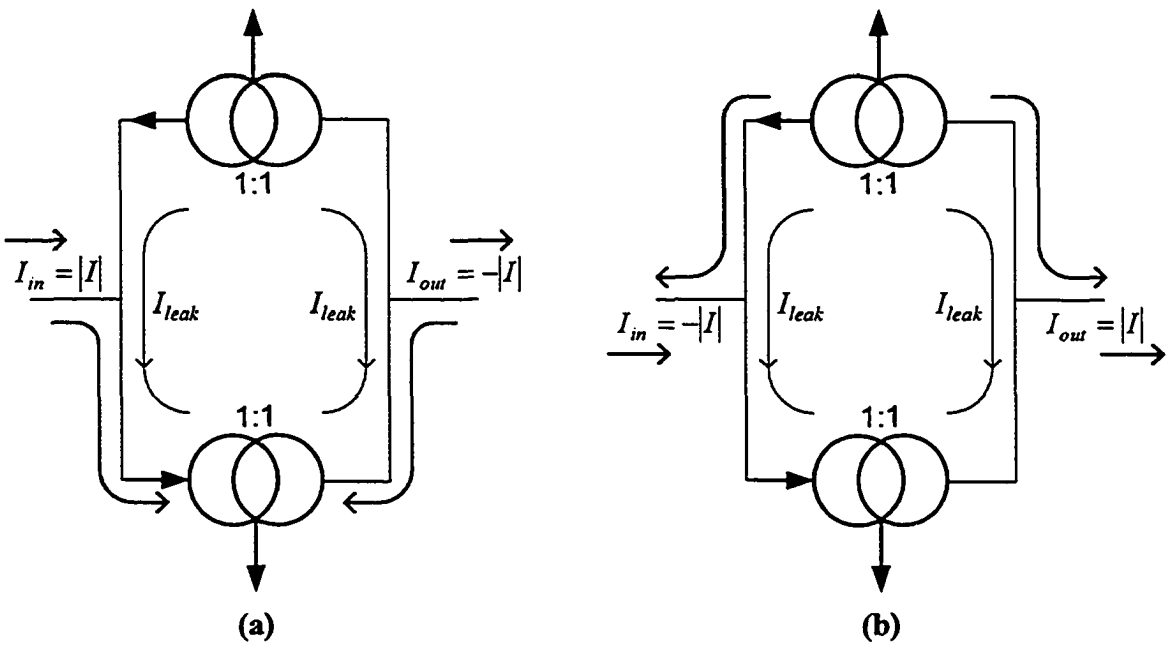


Figure 4-13 Current buffer regardless of the direction of the input current, duplicates it at the output with flipped sign (a) n-type current mirror is active, (b) p-type current mirror is active.

Though leakage current increases static power consumption, it should be noted that leakage keeps transistors in both current mirrors out of the cutoff region when the input current is zero and therefore increases the speed of operation and improves linearity. This is because a completely turned off transistor is too slow and nonlinear. Furthermore, when linearity and speed of the current buffer are important, one way is to always keep transistors in saturation and this can be done if leakage current is set to be larger than the maximum input current.

To see how a variable node is implemented by using current buffers, we consider variable node #1 in Figure 2-2, which is a degree three node and computes its outgoing messages based on the following equations that are derived based on (4-13).

$$\begin{cases} m_{V_1 \to C_1} = m_{V_1} + m_{C_2 \to V_1} + m_{C_4 \to V_1} \\ m_{V_1 \to C_2} = m_{V_1} + m_{C_1 \to V_1} + m_{C_4 \to V_1} \\ m_{V_1 \to C_4} = m_{V_1} + m_{C_1 \to V_1} + m_{C_2 \to V_1} \\ M_{V_1} = m_{V_1} + m_{C_1 \to V_1} + m_{C_2 \to V_1} + m_{C_4 \to V_1} \end{cases} \quad (4\text{-}16)$$

Figure 4-14 shows the required circuits for implementing the above equations. In this figure, current buffers with the same input current share their input-section. A current buffer has at lease two basic sections, an input-section that samples the input current and an output-section that duplicates the input current. By using multiple output-sections, it is possible to reproduce multiple replicas of the input current. This can slightly increase the capacitive load of a current mirror, as the output stages are connected in parallel. However, this would slightly reduce the speed if the number of connected output stages is small.

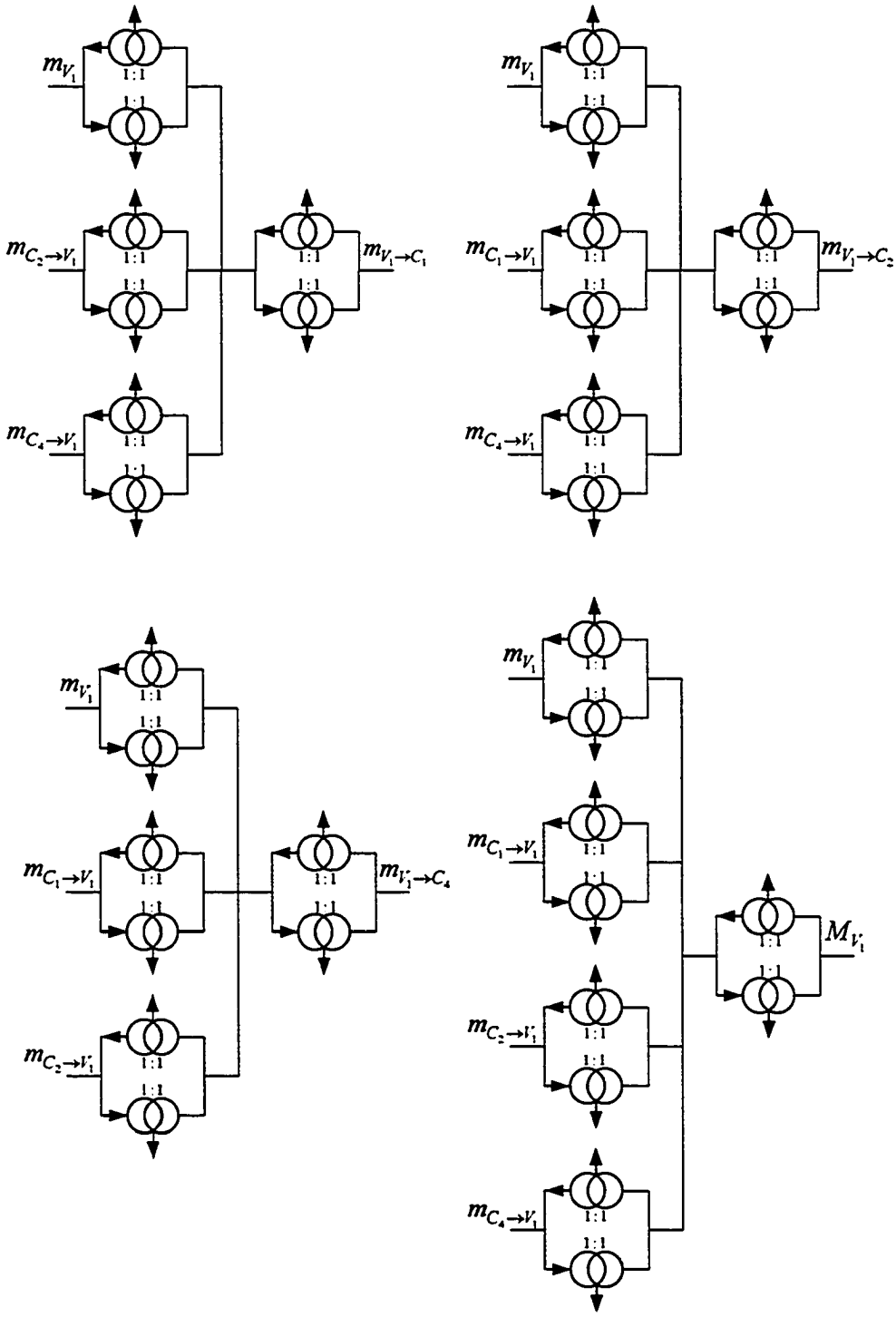


Figure 4-14 Circuits for implementing variable node #1 in Figure 2-2. Buffers with the same input current share their input-section.

In Figure 4-14, the final output is M_{V_1}, which is a current, however, in many applications we prefer having the corresponding sign bit to know the estimated transmitted bit. NOT circuits can translate the sign of M_{V_1} to a bit, as shown in Figure 4-15.

It is important to note that since M_{V_1} is an analog signal, the voltage at the input of the NOT gate might not get close enough to the rail voltages and this can increase leakage current in the NOT gate and subsequently increase the power consumption. This is an important problem because this bit is often one of the chip's outputs and a huge capacitive load should be driven and we need a large NOT gate that could have even higher leakage current. To mitigate this problem, instead of using one large NOT gate, we would use a few cascaded NOT gates. Starting from the smallest NOT gate with the lowest leakage; we gradually increase the gate's size up to a desirable NOT gate with suitable size [10]. Also, it is more desirable if the output voltage of the current buffers could get as close as possible to the rail voltages to limit the leakage current in the NOT gate.

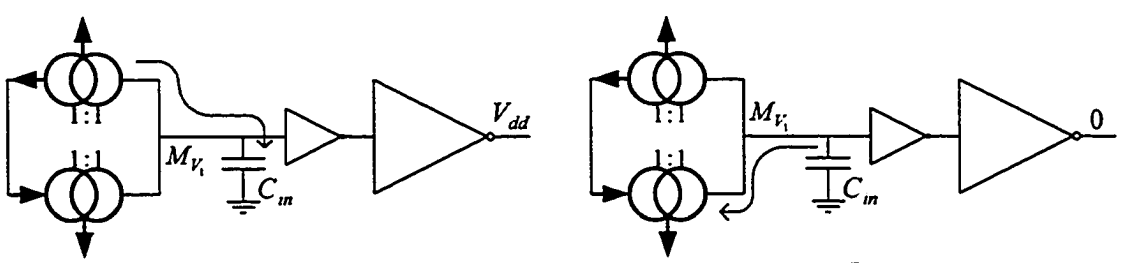


Figure 4-15 NOT gates can be used for detecting the sign of M_{V_1} : (a) M_{V_1} is positive, (b) M_{V_1} is negative.

Finally, we should mention that precision, area, power consumption, and speed of the variable nodes depend mainly on the current buffers and consequently on the current mirrors that have been utilized. A better current mirror makes a better variable node. In the next section, we present how the modules that were introduced in this section can be implemented by MOS transistors.

4.3.3 Implementation of Variable Nodes (Transistor Level)

The problem of designing variable nodes was reduced to design of current mirrors in the previous section. The overall performance of the variable nodes would then depend on the current mirrors that we have used. There are quite a few different current mirror circuits with different performances and we can choose one that best fits our particular application [9], [70]. However, in modern fabrication technologies we have to properly address the problem of transistors' low output impedance and their reduced supply voltages. While the former reduces the accuracy of the current mirrors in general, the latter restrict the swing voltage and ultimately limits the dynamic range. In this chapter, we consider 0.18μm, 1.8V CMOS technology. For this fabrication technology, we use two different cascode current mirrors that are shown in Figure 4-16 and Figure 4-17 .

Figure 4-16 shows a classic cascode current mirror with boosted output impedance. If r_o is the output impedance of one transistor, the output impedance of this current mirror is proportional to $g_m r_o^2$, where g_m is the transistor's transconductance. The minimum voltage at the output of this current mirror is $V_T + 2V_{eff}$ where V_T is the threshold voltage and V_{eff} is the effective voltage and is defined as the difference between gate-source voltage and threshold voltage for a given drain current, that is the minimum drain-source voltage to keep the transistor in the saturation region (constant current region).

The voltage at the input is equal to $2V_T + 2V_{eff}$ [70]. This voltage requirement might not be tolerated in many applications as it could significantly limit the dynamic range. But this current mirror is simple and is self-biased and does not need any external biasing circuitry.

Figure 4-17 shows a high-swing cascode current mirror that not only has high output impedance but also has lower voltage requirements at its input and output terminals. While its output impedance is approximately given by $g_m r_o^2$, the voltage requirement at its input is $V_T + V_{eff}$ and at the output it is only $2V_{eff}$ at the maximum allowed current. The acceptable range for V_{eff} in this current mirror is given by:

$$V_{bias} - 2V_T \le V_{eff} \le \frac{V_{bias} - V_T}{2}, \tag{4-17}$$

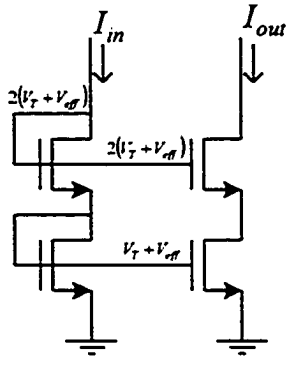


Figure 4-16 An n-type low-swing self-biased cascode current mirror.

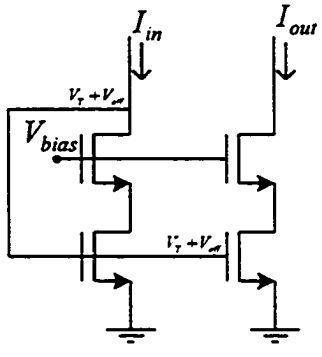
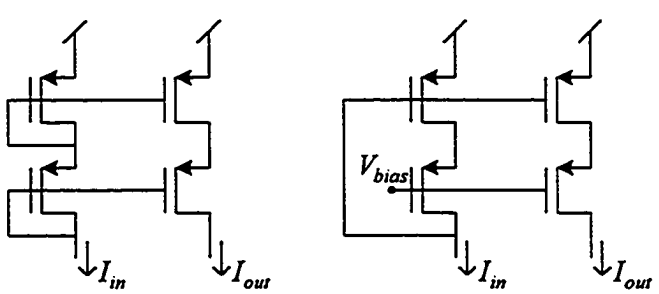


Figure 4-17 An n-type high-swing cascode current mirror.

This inequality is simply derived based on the necessary conditions for biasing the input transistors in saturation. It should be noted that if V_{bias} is greater than $2V_T$, then input current must be adjusted to be greater than the limit as defined by (4-17). Figure 4-18 shows p-type versions of the cascode current mirrors that were introduced in Figure 4-16 and Figure 4-17. Furthermore, Figure 4-19 and Figure 4-20 show simulation results for corresponding current buffers when W_p=2μm, W_n=1μm, L_p=0.4μm, and L_n=1μm. It is important to note that because of the lower mobility of holes than electrons, the output impedance of p-type transistors is higher than n-type transistors and therefore, we use larger W/L for p-type transistors for equal output impedance.

Figure 4-21 shows how by using the self-biased cascode current mirrors shown in Figure 4-16 and Figure 4-18, and the proposed structure for a variable node in Figure 4-14, we can implement variable node #1 in Figure 2-2. The threshold voltage of the NOT gate was set at 0.9 volts. In the output section of the variable node, we use larger transistors to be able to drive the long interconnections that are connected to check nodes. Performance of this node is close to ideal as shown for one output in Figure 4-22. In this figure, it is assumed that all inputs are equal triangular current sources with peak currents of 4μA and -4μA. Figure 4-23 shows the corresponding sign bits.



(a) (b)

Figure 4-18 p-type cascode current mirror (a) self-biased low-swing, (b) high-swing.

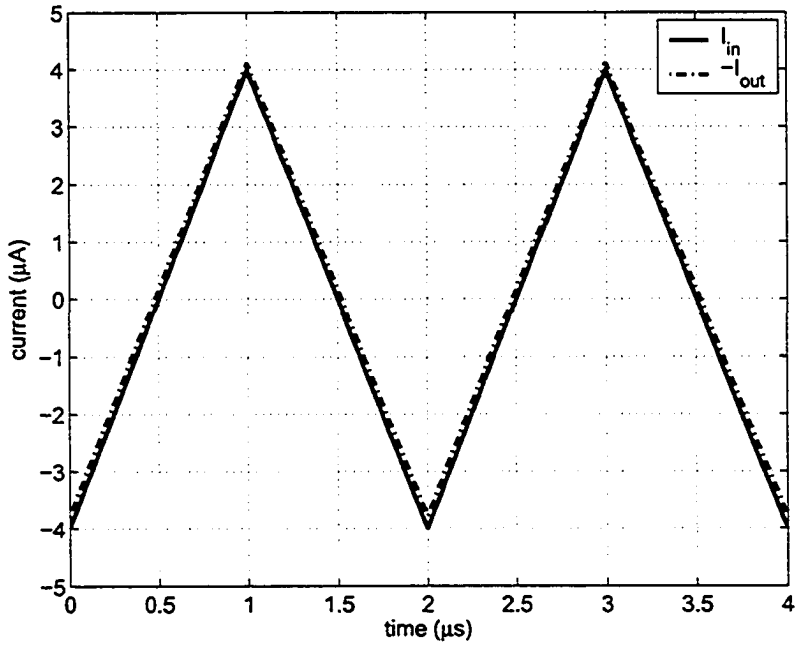


Figure 4-19 Input and output currents in a low-swing current buffer based on Figure 4-16 and Figure 4-18 (a).

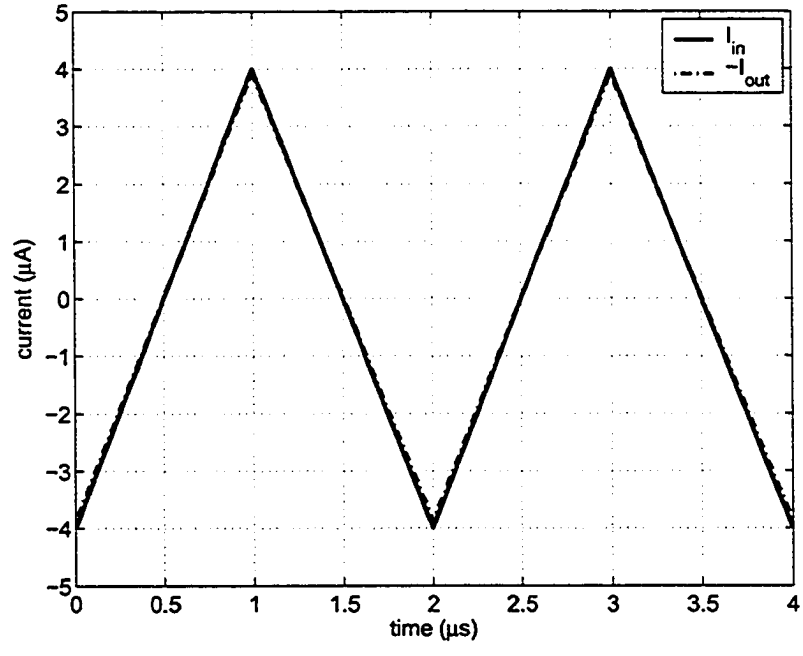


Figure 4-20 Input and output currents in a high-swing current buffer based on Figure 4-17 and Figure 4-18 (b). V_{bias} is equal to 0.9 V.

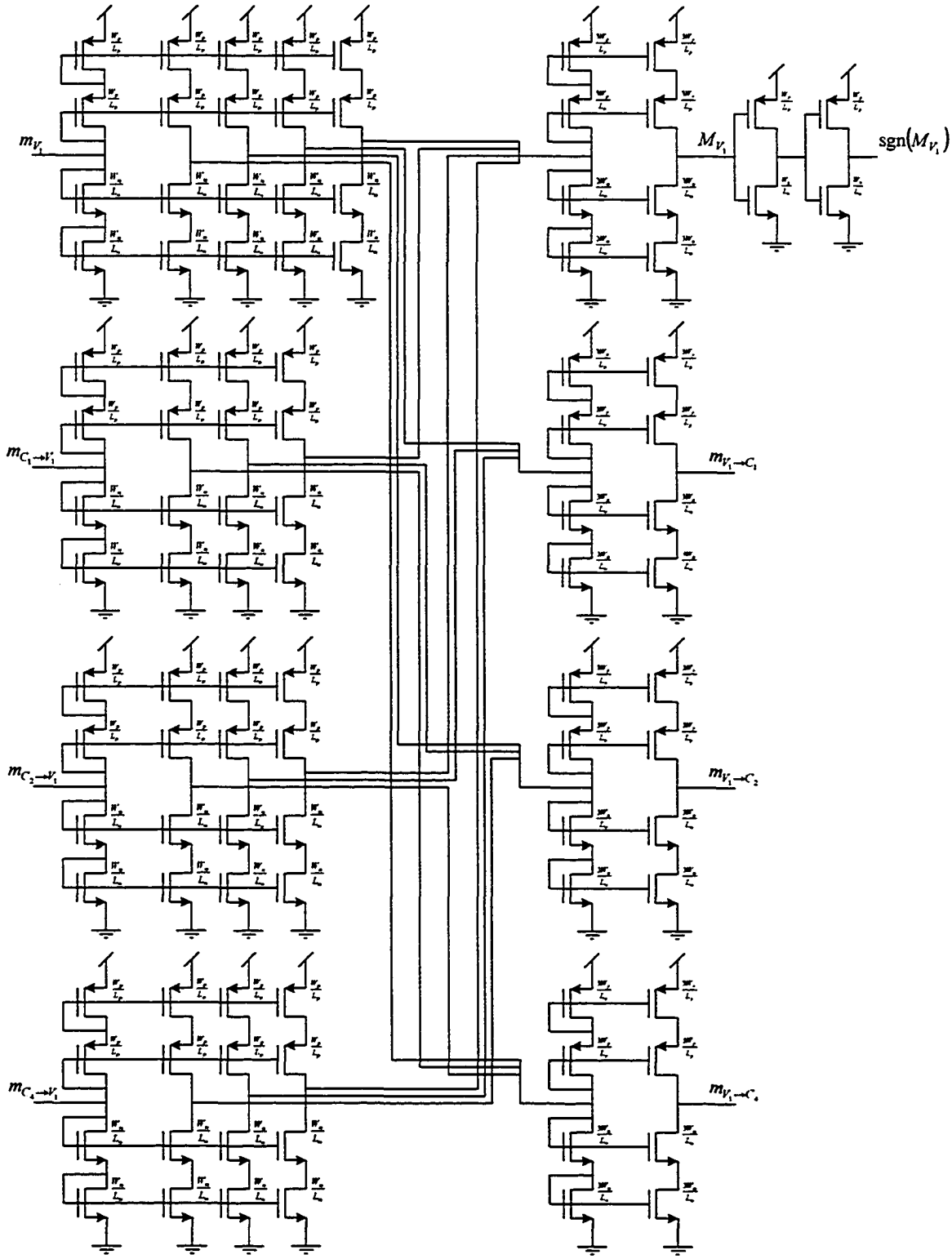


Figure 4-21 Transistor level implementation of variable node #1 in Figure 2-2.

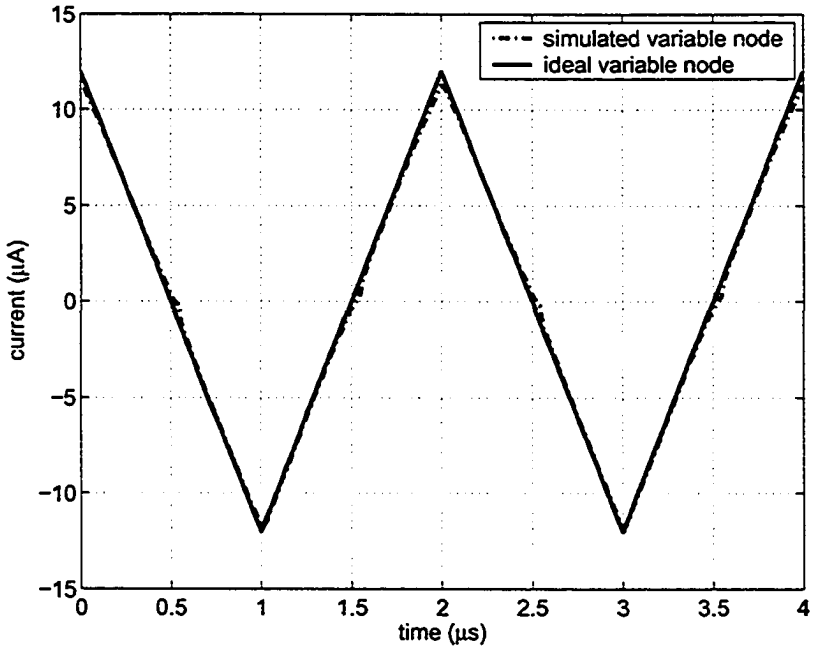


Figure 4-22 Comparison between $m_{V_i \to C_i}$ obtained by transistor level simulation for the variable node in Figure 4-21 and an ideal variable node.

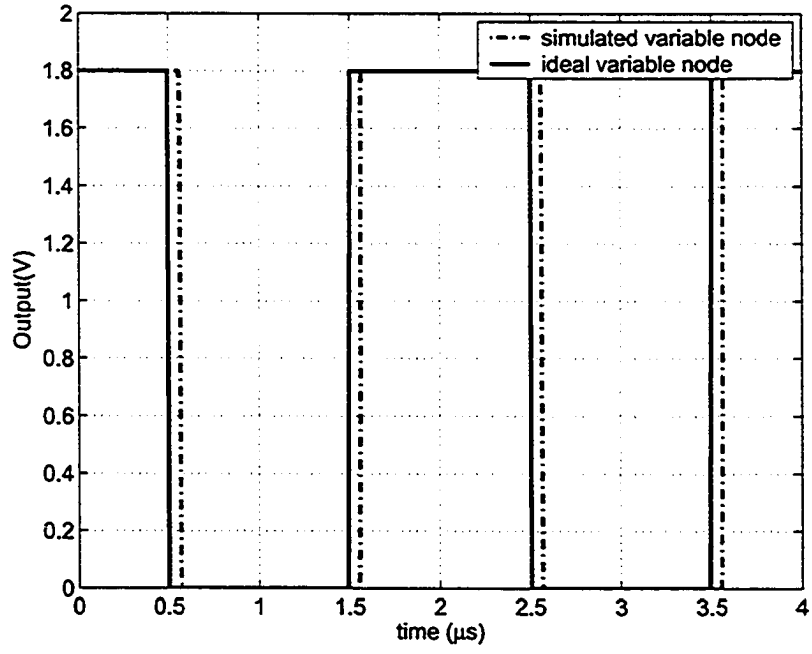


Figure 4-23 Comparison between $\mathrm{sgn}(M_{V_i})$ obtained by transistor level simulation for the variable node in Figure 4-21 and an ideal variable node.

4.3.4 Implementation of Check nodes (Basic Modules)

Operations in check nodes are rather complex compared to what we had for variable nodes. In this section, the basic modules for implementing check nodes are presented. As an example, we focus on implementation of check node #1 in Figure 2-2 which is a node of degree 4. The operation in a check node is given by (4-13) and Figure 4-24 shows the necessary modules for implementing this node. First in the block labeled *sign & magnitude extractor module*, for all incoming messages sign and magnitude are separated. Then the magnitude of each outgoing message to each variable node is computed by finding the minimum value among the magnitudes of all the received messages from other variable nodes that are connected to the check node through edges in the Tanner graph representation of the code. This operation is performed in the *minimum winner-take-all (min WTA) module*. The sign of the outgoing message to each variable node is obtained by multiplying the sign of the messages that have been received from other variable nodes; this operation is performed in the *Sign module*. Then the *sign & magnitude combiner module* combines the computed signs and magnitudes and generates the outgoing messages. Each module in Figure 4-24 consists of a few simpler modules.

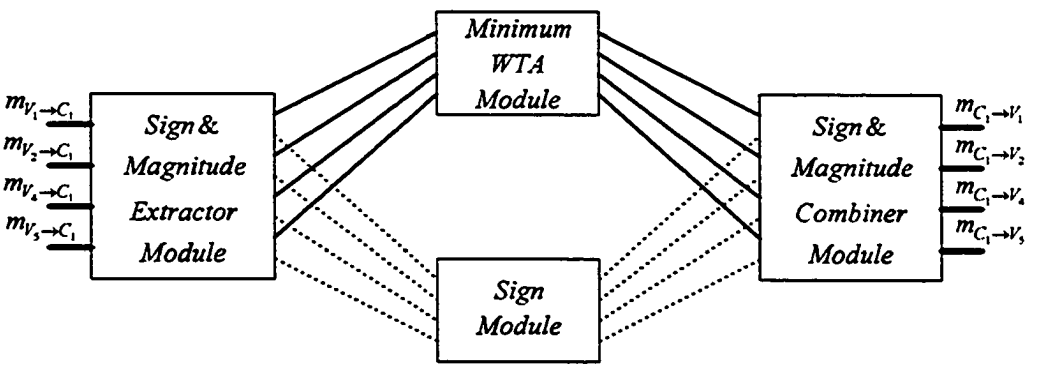


Figure 4-24 Basic modules in the check node #1 in Figure 2-2.

Figure 4-25 shows more details about the structure of the check node. In this figure digital interconnections are shown by dashed lines. Sign & magnitude extractor module consists of four simpler modules, each has one input and two outputs and splits the input signal into its sign and absolute value and we call it *RTAS* (real to absolute value and sign converter) module.

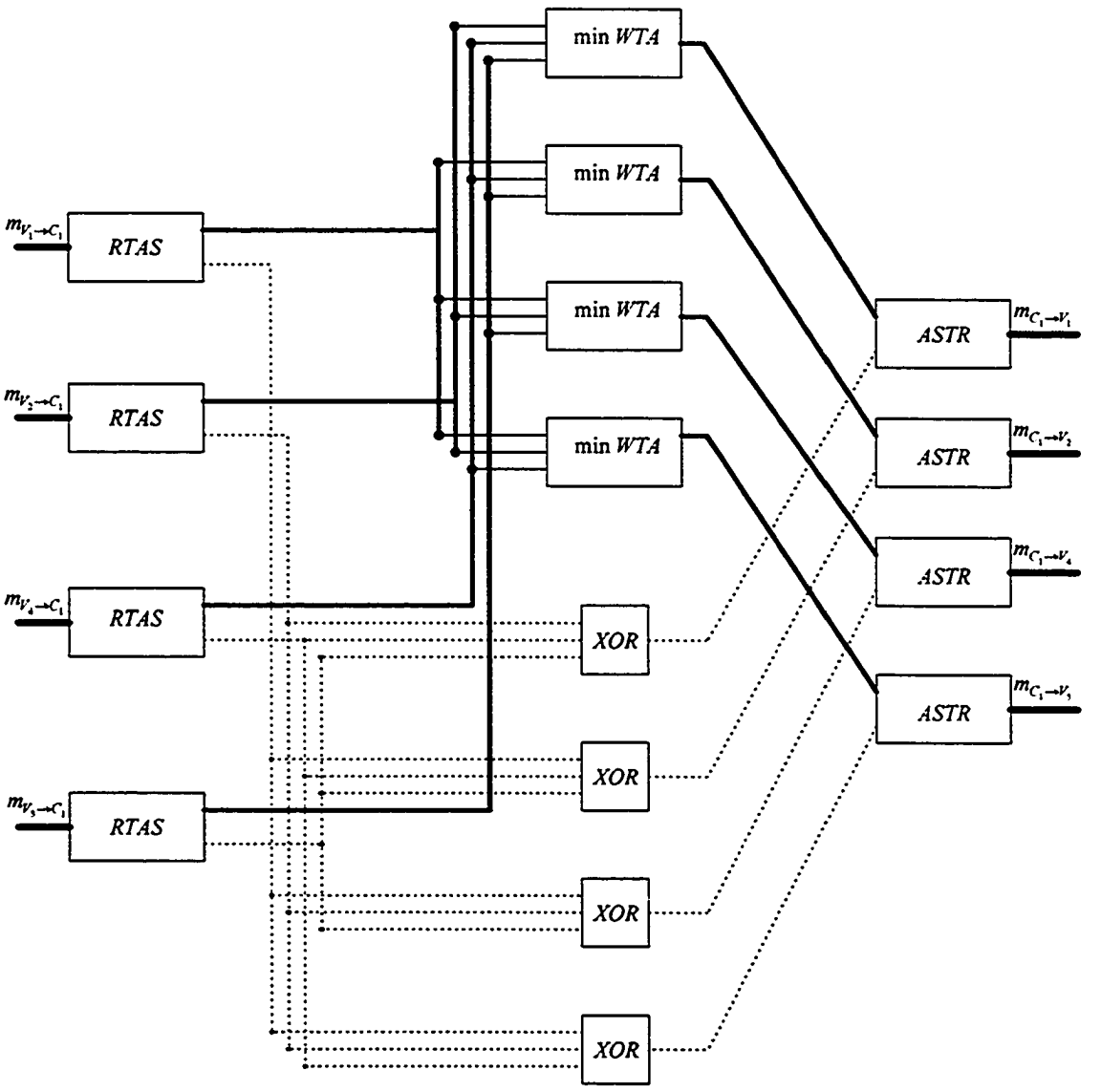


Figure 4-25 A closer look at the modules in the check node #1 in Figure 2-2.

It is worth mentioning that while in equation (2-17), the sign function could return -1 or 1; in hardware implementation it is simpler to work with a sign bit that represents the polarity of the message. For negative number the sign bit is set to be 1 and for non-negative numbers it is set to be 0. In this way, instead of multiplying the signs, we can XOR the sign bits. Consequently, the sign module consists of four 3-input XOR gates. The min WTA module consists of four 3-input minimum winner-take-all circuits. In a min WTA circuit, the minimum input appears at the output. The sign & magnitude combiner module consists of four *ASTR* (absolute value and sign to real converter) modules that combine the sign bit and absolute value and generate real numbers.

4.3.5 Implementation of RTAS Modules

Figure 4-26 shows how an RTAS module works. It is worth noting that leakage current similar to what was explained for variable nodes also exists in the current buffers and wastes power but ideally does not degrade the accuracy of the RTAS module. In order to implement this module we can simply substitute current mirrors in Figure 4-26 with full transistor level circuits. However, it is necessary to make sure that current mirrors are biased properly and there is enough room for voltage swing. In the check node, we typically use the high-swing cascode current mirror that was introduced earlier and shown in Figure 4-17 and Figure 4-18 (b).

A circuit for implementing RTAS module in 0.18μm CMOS technology is given in Figure 4-27. In this figure, the sign bit extractor has been simplified and the output that represents absolute value is a voltage instead of a current. We did this in order to simplify the interconnections among the modules inside the check node. As it was shown in Figure 4-25, the output of the RTAS module should be sent to a few min WTA modules

that are located in close proximity to the RTAS module. Therefore, we conduct the output current to a diode-connected circuit that works as the RTAS output stage and generates a biasing voltage for the high-swing current mirrors inside the min WTA modules. In this figure, V_{tt} is a fixed biasing voltage for high-swing cascode current mirrors.

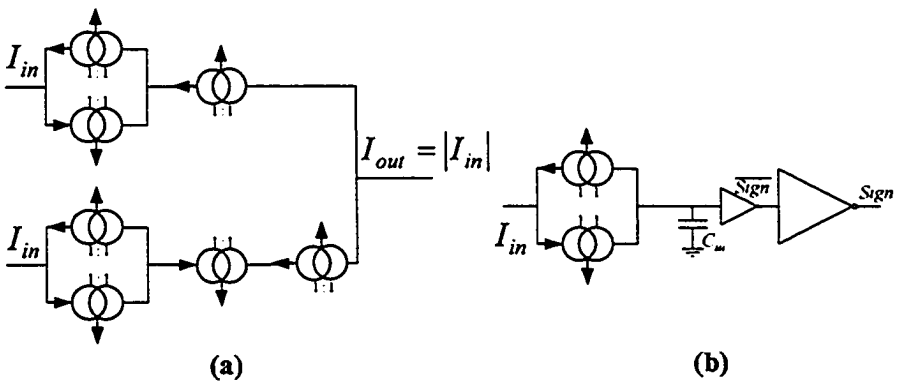


Figure 4-26 An RTAS module, (a) current rectifier, (b) sign extractor.

4.3.6 Implementation of Min WTA Modules

Winner-take-all circuits consist of a few cells that are competing with each other and each cell tries to duplicate its input at the output. The idea of duplicating a signal implies that there should be some kind of mirroring in the circuit. As we will see later in this thesis this idea can be exploited and we can devise maximum or minimum winner-take-all circuits by using current mirrors and suitable feedback circuits. As we will see later, it is easier to implement current-mode max WTA. However, it is not difficult to convert a current-mode max WTA to a min WTA circuit. Figure 4-28 shows how by subtracting input currents from a sufficiently large reference current (I_{ref}) a current-mode max WTA can be converted to a min WTA circuit.

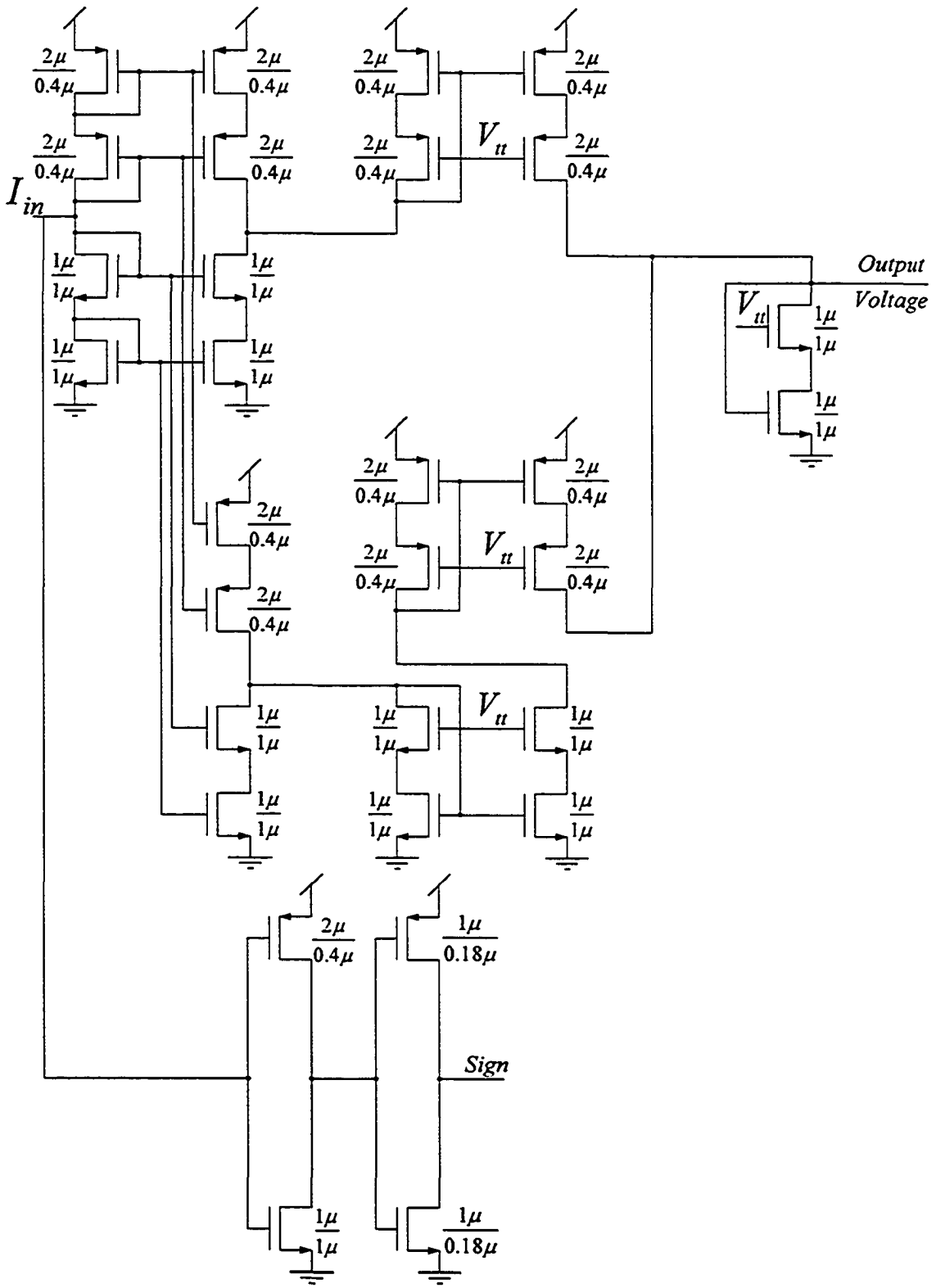


Figure 4-27 A circuit for implementing RTAS module.

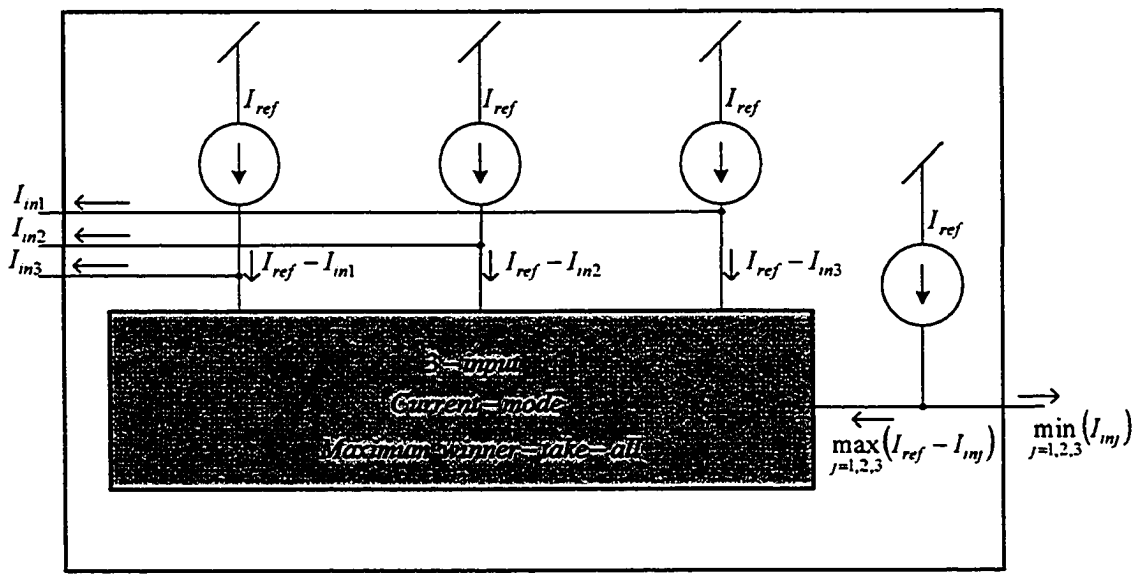


Figure 4-28 A current-mode max WTA can be converted to a min WTA. I_{ref} is a fixed and sufficiently large reference current. I_{in1}, I_{in2}, and I_{in3} are input currents.

Figure 4-29 shows a well-known max WTA circuit [92]. In this figure I_{in1}, I_{in2}, and I_{in3}, which are three input currents are applied to the drain terminals of $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors that are identical and have large output impedances. These three transistors share the same gate-source voltage. This shared voltage is determined by the amount of current that is provided for the diode-connected transistor Q_d through transistors $Q_{f(1)}$, $Q_{f(2)}$, and $Q_{f(3)}$. To explain how this circuit works, first we assume input currents are equal and are greater than zero, we also assume initially that the voltage drop on Q_d is almost zero. Consequently, drain currents for $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors would be smaller than input currents. This would increase drain-source voltages of $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors and therefore gate voltage of $Q_{f(1)}$, $Q_{f(2)}$, and $Q_{f(3)}$ transistors would increase, this would however increases the injected current and voltage drop across Q_d increases and consequently the drain currents in $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors go up. This

feedback increases the shared gate-source voltage up to a balancing point. If for any reason this shared voltage goes higher than what it should be, drain currents in $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors become more than input currents and their drain voltage would drop and reduces the injected current and therefore reduces the voltage drop on the diode-connected Q_d transistor. This negative feedback establishes a suitable gate voltage that can be used for regenerating the input current in the drain of the Q_o transistor which is identical with the $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors. If input currents are not equal, in a similar way the maximum input current that generates the maximum input voltage keeps the feedback path closed and establishes a suitable voltage for the gate of Q_o to regenerate itself at the output. However, the shared gate-source voltage forces $Q_{s(1)}$, $Q_{s(2)}$, and $Q_{s(3)}$ transistors to have the same drain currents, and those transistors, which can not afford that much current would ultimately leave saturation and are biased in triode. This however reduces their drain voltage and consequently their corresponding $Q_{f(.)}$ transistors will be turned off and they can not contribute in keeping the feedback functional. It is worth noting that if the Q_d transistor is identical with the Q_o transistor, the overall currents in the $Q_{f(1)}$, $Q_{f(2)}$, and $Q_{f(3)}$ transistors will also be equal to the winner current. In Figure 4-30 and Figure 4-31, circuit simulations for this WTA are given. For all transistors W/L=1μm/1μm.

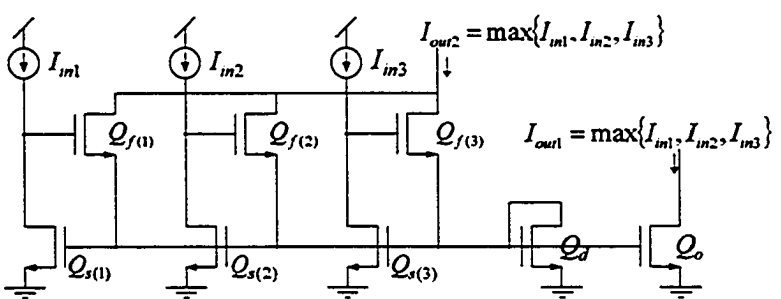


Figure 4-29 A 3-input current-mode maximum WTA circuit.

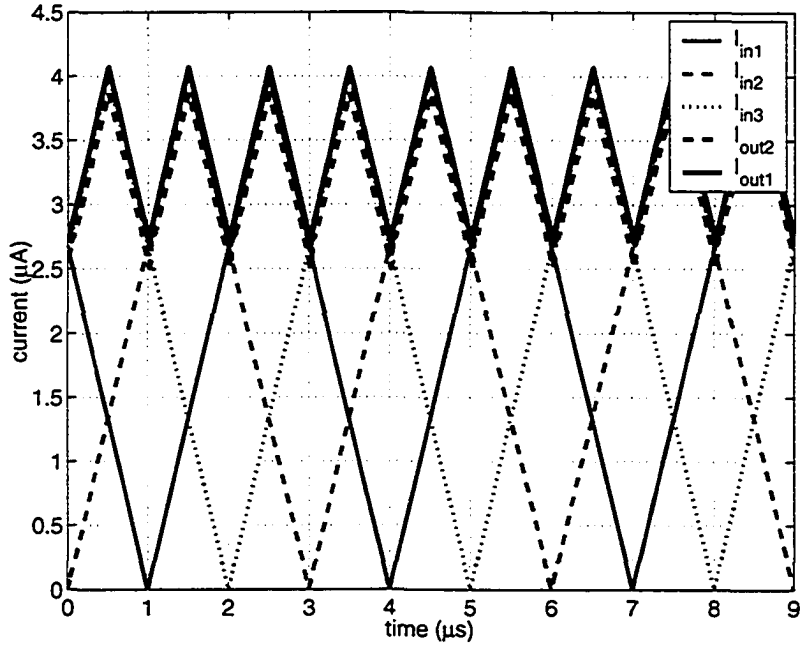


Figure 4-30 Simulated input and output currents of the WTA circuit in Figure 4-29 .

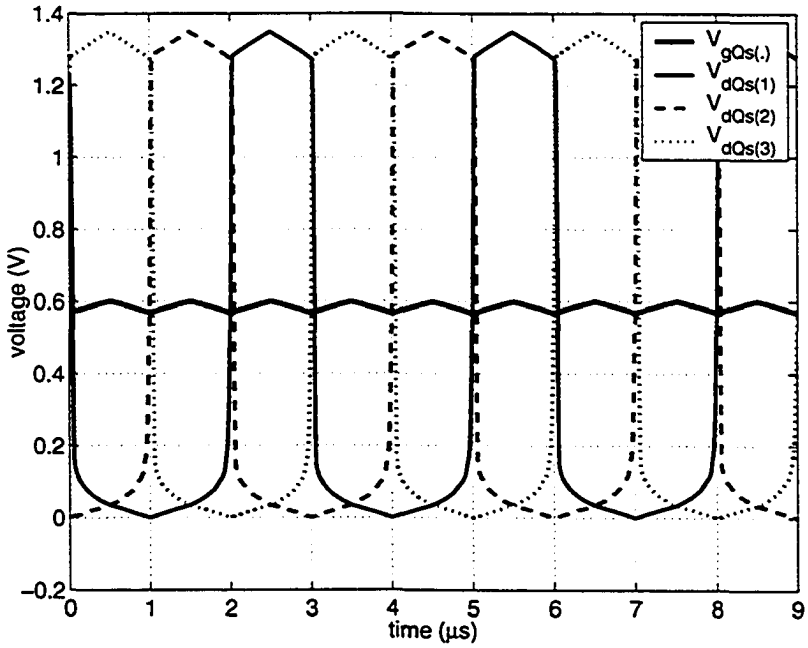


Figure 4-31 Simulated voltages in the WTA circuit in Figure 4-29, corresponding input currents are shown in Figure 4-30.

As explained earlier, current mode max WTA circuits can be easily converted to min WTA circuits. Figure 4-32 shows how a min WTA circuit can be constructed based on max WTA in Figure 4-29. To do this we have first subtracted the input currents from a reference constant current (I_{ref}), which should always be greater than the minimum input current. In fact, it is easy to show that if an input current is greater than I_{ref}, it can not win the competition and regenerate itself at the output cell. The modified input currents are conducted to the max WTA circuit given in Figure 4-29 and the maximum input current is reproduced at the output. By subtracting the output of max WTA circuit from I_{ref} the minimum input current is obtained.

Unfortunately, for deep submicron MOS transistors, short channel effects degrade the accuracy of the max WTA circuit in Figure 4-29 and consequently the accuracy of the min WTA will be degraded too. This is due to the fact that short channel effects degrade the output impedance of MOS transistors [70].

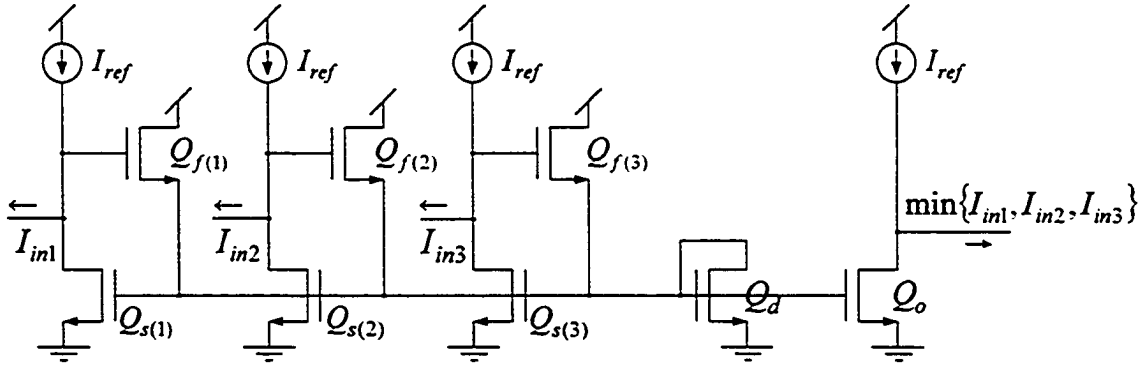


Figure 4-32 A 3-input current-mode min WTA circuit.

In order to implement min WTA in the check nodes, we improve the output impedance of the transistors in Figure 4-32 , by replacing each transistor with a high-swing cascode pair as shown in Figure 4-33. This is similar to what we used for high-swing cur-

rent mirrors and the transistor on the top (Q_{sc}) is biased with a fixed voltage V_{tt} and the overall output impedance would be close to $g_m r_o^2$, where g_m is the transistor's transconductance and r_o is the output impedance of each transistor. In this figure I_{ref} is generated by Q_{ref1} and Q_{ref2} transistors which are biased by V_{bias} and V_{tt2}, respectively. V_{in} is the output voltage of an RTAS module and input currents I_{in1}, I_{in2}, and I_{in3} are mirrored from RTAS modules and this is done by means of $Q_{in(.)}$ and $Q_{inc(.)}$ transistors.

In 0.18μm CMOS technology, we used 2μm/0.4μm p-type MOS transistors and 1μm/1μm n-type MOS transistors. Also V_{tt}, V_{tt2} are set at 0.9 volts and V_{bias} is equal to 1.2 volts. Figure 4-34 shows simulation results for this circuit when three periodic triangular input currents are applied to this min WTA circuit. This figure illustrates that when the output follows one input, the accuracy is much better than when the winner input is changed. This problem is often seen in WTA circuits and increases as frequency increases.

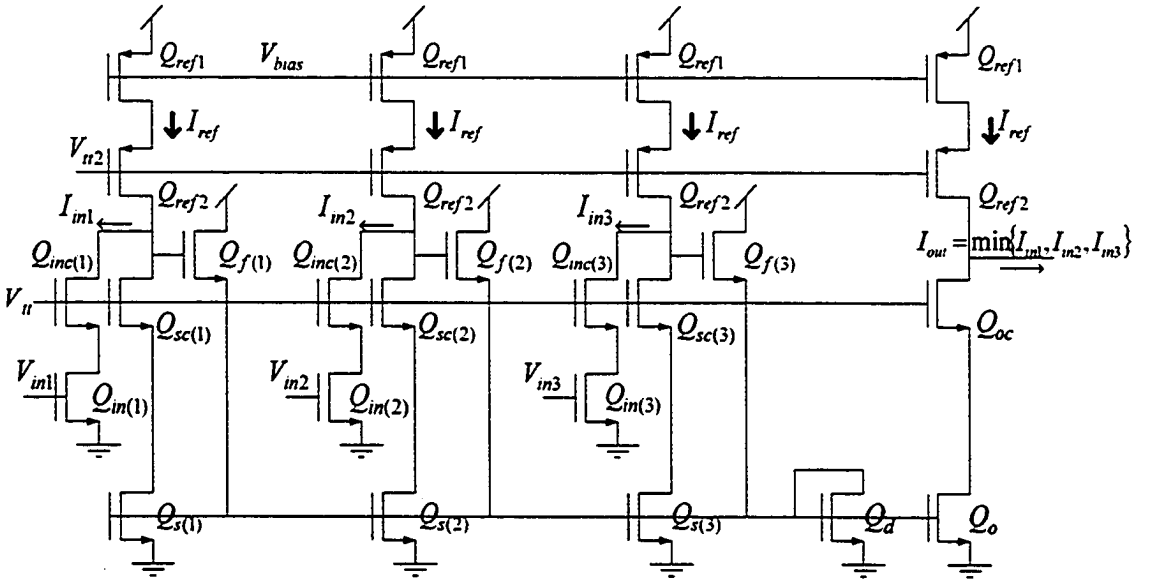


Figure 4-33 A current-mode cascode 3-input min WTA.

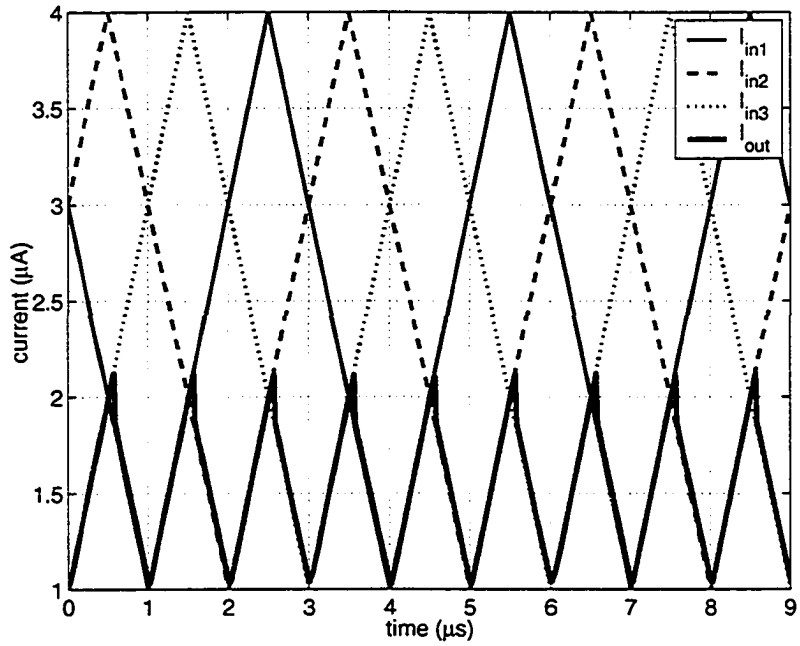


Figure 4-34 Simulation results for the min WTA circuit in Figure 4-33.

4.3.7 Implementation of XOR Gates

It is quite important to limit the switching noise in the check nodes by choosing a suitable XOR gate from a low noise logic family and one of the best choices is DCVSL (Differential Cascode Voltage Switch Logic) XOR gates [10]. A 2-input DCVSL XOR is shown in Figure 4-35. XOR gates with more inputs can be constructed by increasing the cascode layers or by cascading 2-input XOR gates.

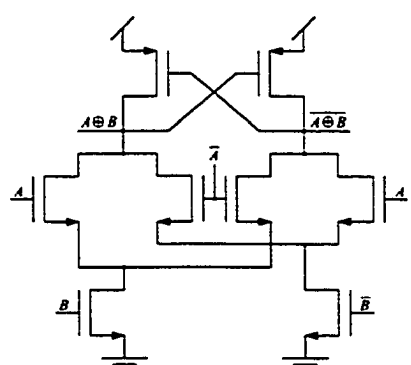


Figure 4-35 A 2-input DCVSL XOR gate.

4.3.8 Implementation of ASTR Modules

The structure of an ASTR module has been illustrated in Figure 4-36. Input current is received from the min WTA module and the sign is generated in XOR gates and these two are combined together to generate the outgoing message of the check node. A circuit for implementing the ASTR module is given in Figure 4-37. This circuit has been obtained by replacing all the current mirrors in Figure 4-36 by high-swing cascode current mirrors. Circuits for implementing analog switches [10] are also given. It is necessary to provide a mechanism to guarantee that when an analog switch is open, there is no leakage current at the output. To do so additional transistors (Q_1 and Q_2) have been added to the analog switches. The Q_1 and Q_2 transistors are turned on when the S_1 and S_2 analog switches are open, respectively. These extra transistors do not permit the opened gate terminal to remain floating but instead connect it to a suitable rail voltage and completely shut down the corresponding transistor.

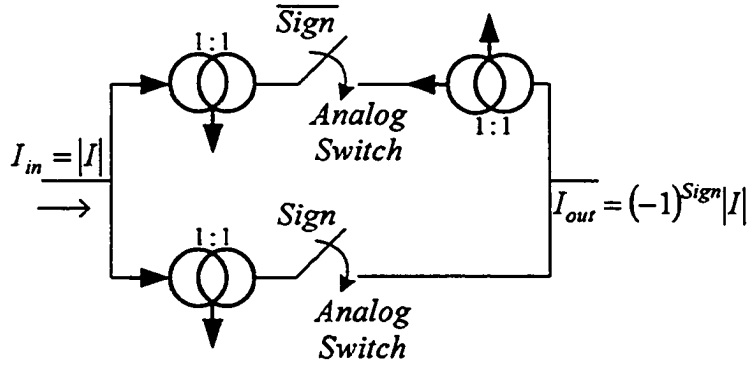


Figure 4-36 ASTR module. An analog switch is closed when the applied signal is 1. Current mirrors with the same input current can share their input sections.

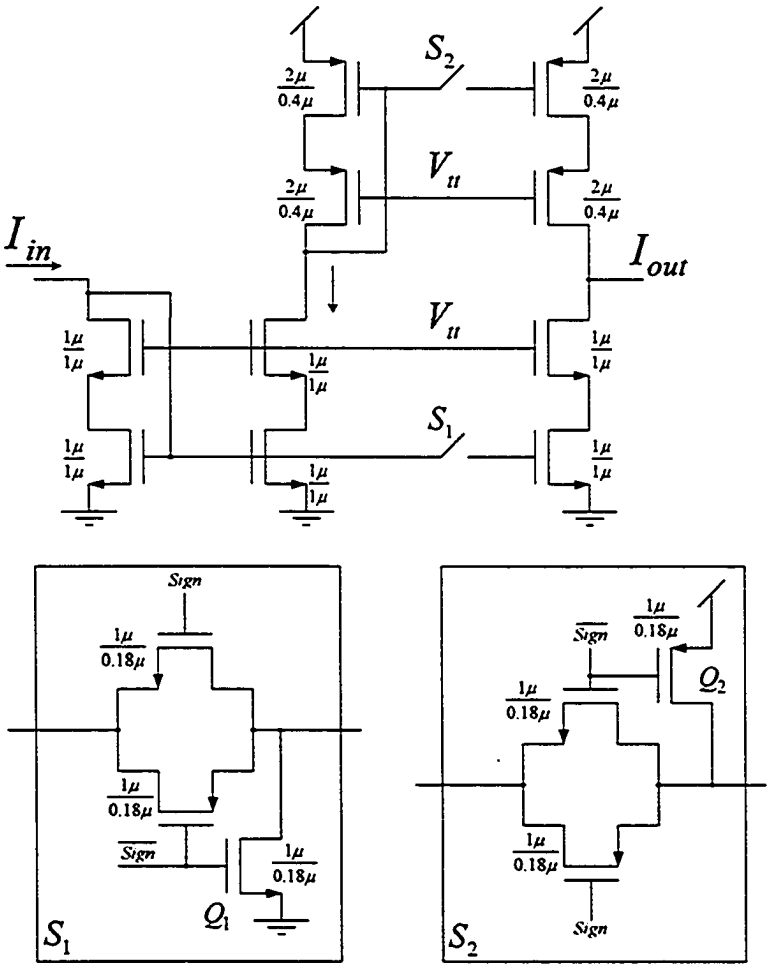


Figure 4-37 A circuit for implementing ASTR module.

To study the accuracy of the proposed modules of a check node, we consider a simple case when one output in the check node should be an exact replica of an input current. This is the case when other input currents are large and binary addition of the related sign bits is zero. We consider two cases of interest, fast and slow variations of the input current and simulate the output current for each case. Corresponding simulation results are shown in Figure 4-38 and Figure 4-39. While Figure 4-38 shows the accuracy is good for low speed operation, Figure 4-39 illustrates some distortion at zero crossing for high speed operation of the check node. This distortion can be explained by noting that when transistors are biased at the cutoff region, they become very slow. It means that

when input current becomes small transistors in the active current buffer can not follow the input variations. Furthermore, when the polarity of the input current changes, active current mirrors change in RTAS and ASTR modules. This means that some current mirrors should be turned off and some should be turned on. However, a completely turned off current mirror is too slow to follow fast variations of the input current. This is the reason why the check node's output can not follow the input precisely.

These figures also show that when magnitude of the input current is greater than a limit, it will be clipped. This clipping effect happens when all input currents in Figure 4-33 are greater that I_{ref} and ultimately I_{ref} appears at the output. To increase this clipping limit, it is possible to increase I_{ref}.

4.4 Conclusion

In this chapter, we considered implementation issues of iterative decoders. First, we analyzed the pros and cons of implementing iterative decoding algorithms using analog VLSI. Then we presented the basic processing modules in the previously reported analog decoders, which are implemented by BiCMOS or weakly inverted CMOS transistors. Finally, we presented current-mode CMOS circuits for implementing min-sum analog iterative decoders. The circuits can be fabricated using standard CMOS technology. This means lower fabrication cost or faster circuits compared to the previously reported analog iterative decoders.

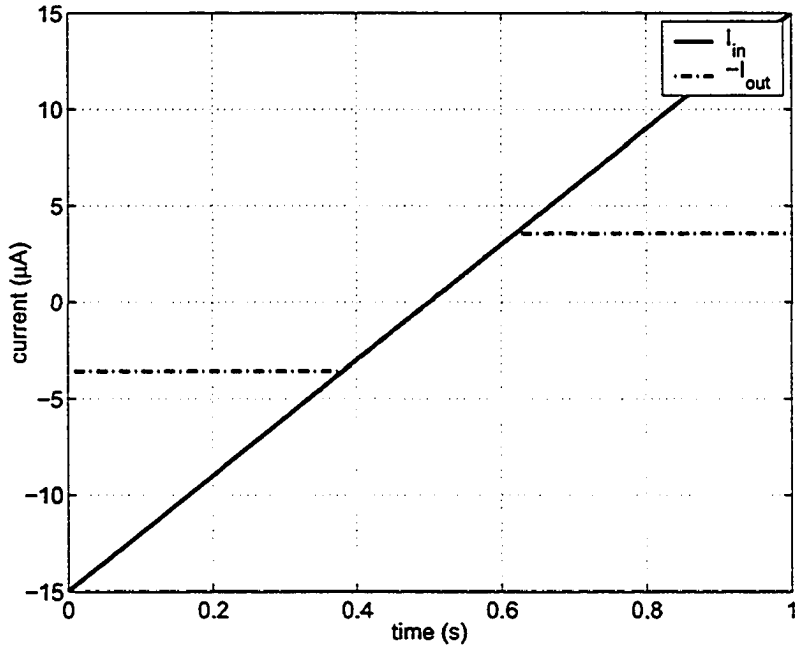


Figure 4-38 Simulated low-frequency transfer response of a check node.

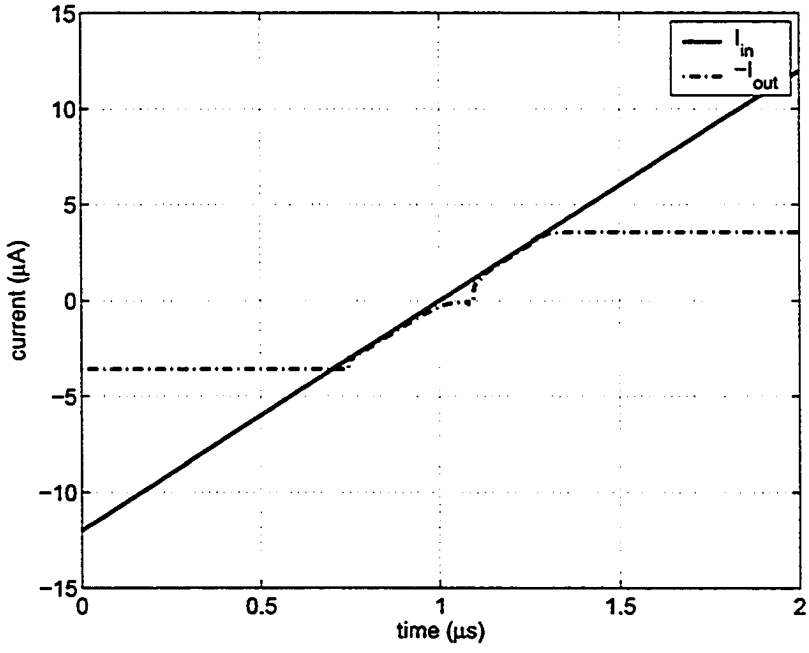


Figure 4-39 Simulated high-frequency transfer response of a check node.

Chapter 5

Min-Sum Analog Decoder with High-Degree Modules

In this chapter, we focus on implementation issues for high-degree variable nodes and check nodes in an analog min-sum iterative decoder and present new CMOS circuits with lower power and area consumption compared to what we presented in the previous chapter. To realize the significance of designing high-degree nodes, one should note that irregular LDPC codes have better performance than regular LDPC codes and the error correcting performance generally improves by increasing the maximum degree of the nodes. In the reported circuits for implementing belief propagation in analog VLSI, as we mentioned earlier, multi-input modules can be implemented by properly cascading two-input soft gates [15]. In the circuits that we proposed in the previous chapter for implementing analog min-sum iterative decoder, modules can simply have as many inputs as we wish and there is no need for cascading simpler modules and consequently increasing the latency of the circuits. However, in this chapter we show that we can further simplify our proposed circuits for high-degree nodes and use silicon area more efficiently.

5.1 High-Degree Variable Nodes

In the previous chapter, we saw how by using current buffers we can implement a variable node. While we mainly focused on a 3-input variable node as an example, it is possible to use the same approach for nodes with higher degrees that often exist in the Tanner graph representation of many good codes. We use the number of current buffers and/or transistors as a measure of complexity for each design. We also distinguish between input and output sections of a current buffer. This is because the input section of a current buffer is shared with other current buffers with the same input current. Also for the sake of simplicity we assume all buffers are identical. We also ignore NOT gates because they would be the same for different methods.

It is simple to verify that for a variable node of degree n (n>1), we need $2(n+1)$ input sections and n^2+2n+2 output sections. The total number of sections is $(n+2)^2$ and if we use cascode current mirrors similar to Figure 4-21, we need $2(n+2)^2$ p-type and $2(n+2)^2$ n-type transistors. As an example for a variable node of degree 18, we need 800 p-type and 800 n-type transistors.

We can significantly simplify the variable nodes by rewriting equation (4-12) as follows:

$$m_{V\to C}^{(l)} = m_V + \sum_{C'\in N_V} m_{C'\to V}^{(l-1)} - m_{C\to V}^{(l-1)}, \tag{5-1}$$

or equivalently:

$$m_{V\to C}^{(l)} = M_V^{(l)} - m_{C\to V}^{(l-1)}. \tag{5-2}$$

This means that we should generate $M_V^{(l)}$ (based on (4-14)) first and then by a simple subtraction we can generate the outgoing messages.

To see how this different interpretation of (4-12) can reduce the complexity of a variable node, we present necessary modules for implementing a 3-input variable node in Figure 5-1. The same node was studied earlier as an example and Figure 4-14 shows the required modules for implementing this node with the previous approach. Transistor level implementation of this node is illustrated in Figure 5-2 and by comparing it with the previous approach that was shown in Figure 4-21, it is observed that the number of transistors has been reduced from 100 to 64. In this figure, we used W_p=2μm, W_n=1μm, L_p=0.4μm, and L_n=1μm in 0.18μm CMOS technology. The threshold voltage of the NOT circuit was set at $V_{dd}/2$.

It should be noted that in the proposed method in the previous chapter, a smaller dynamic range is required compared to the method that is proposed in this chapter. This is because in a variable node the magnitude of M_V could be higher than any other outgoing messages of the variable node simply because it has more inputs. However, in the former method it was not necessary to compute M_V precisely and as long as the sign of M_V is correct, the variable node will be accurate. However, in the latter method it is necessary to compute M_V precisely, otherwise all the messages will be inaccurate. Therefore the latter method requires higher output linearity and hence a wider dynamic range.

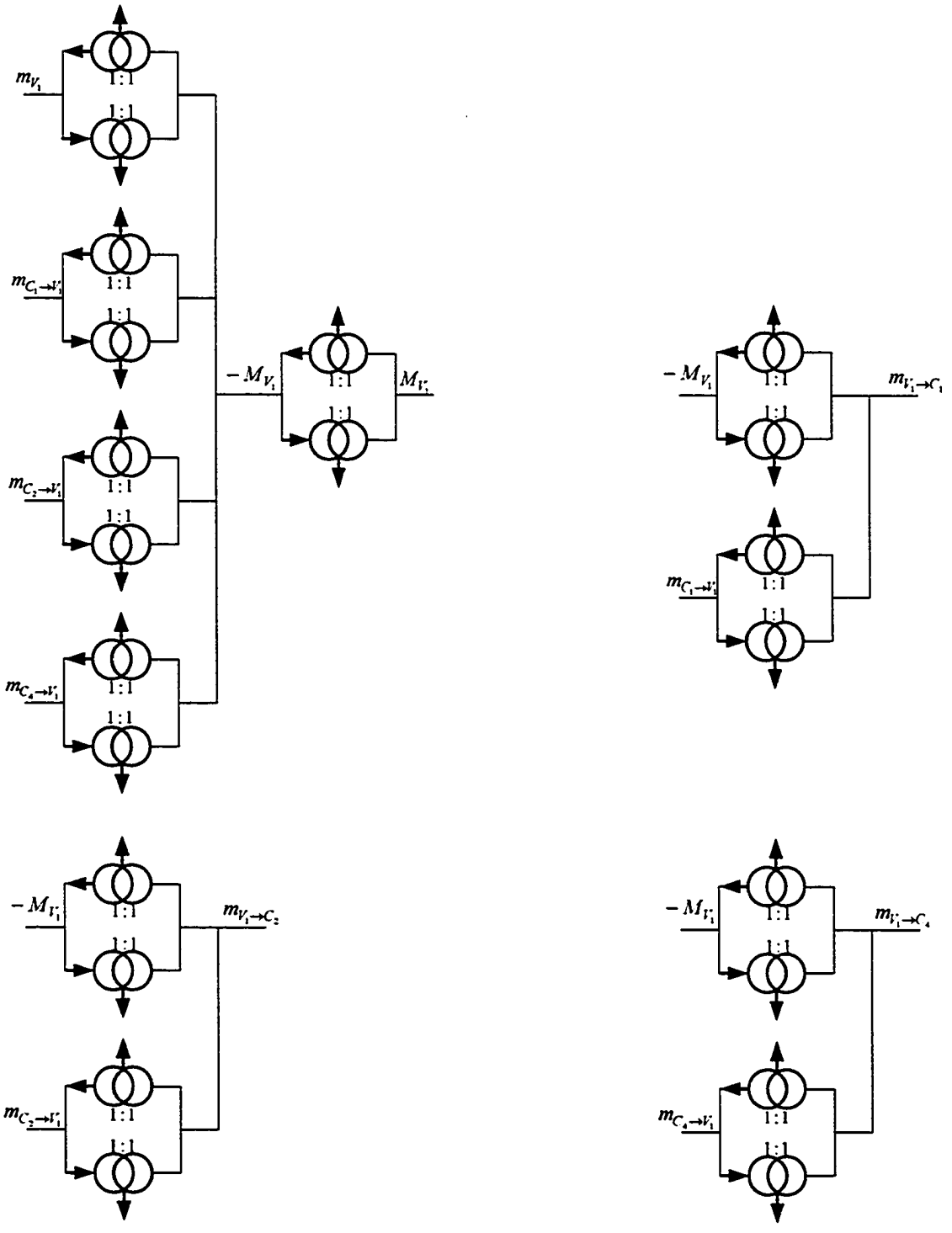


Figure 5-1 Modified circuits for implementing variable node #1 in Figure 2-2. Buffers with the same input current share their input-section.

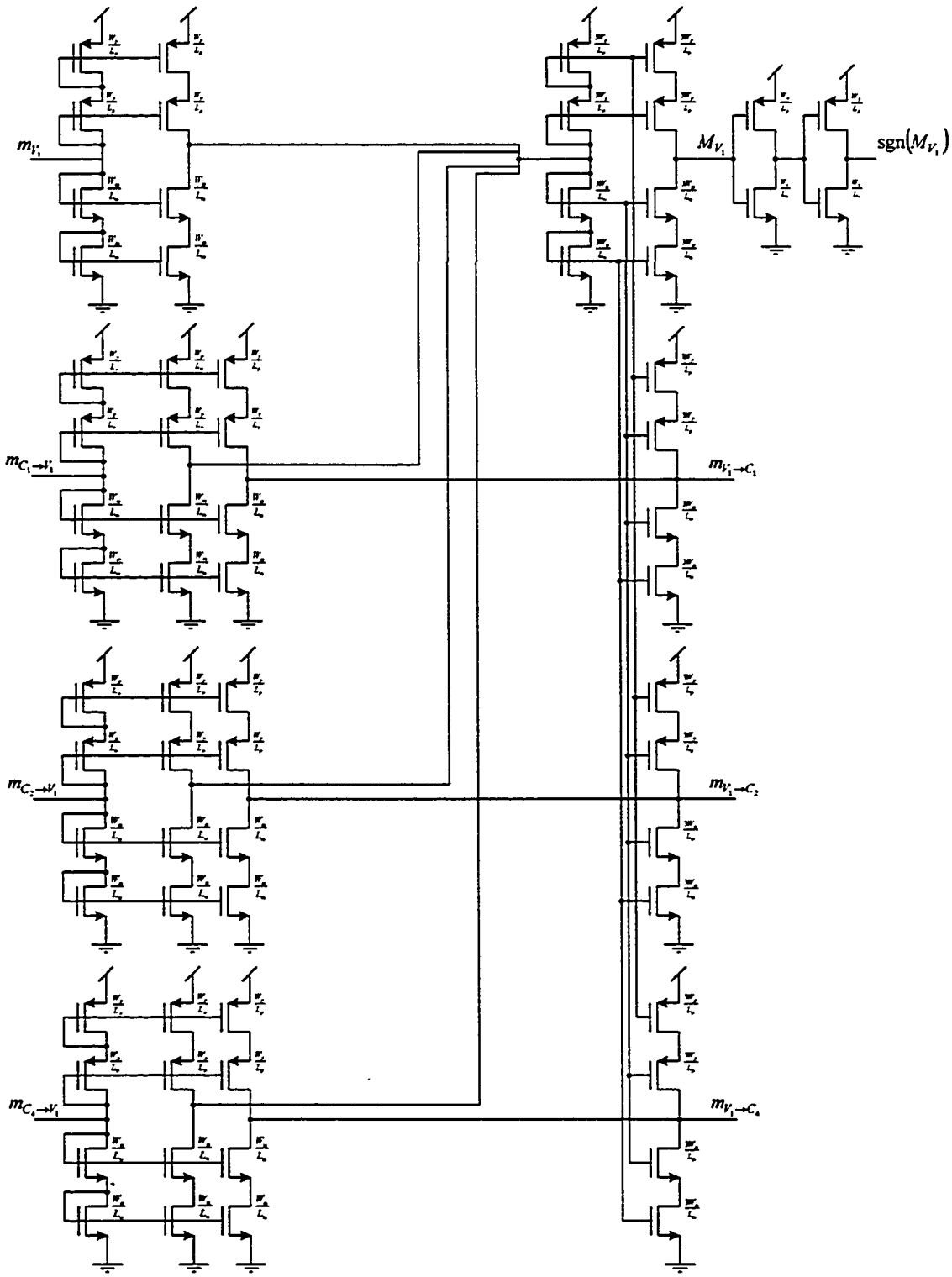


Figure 5-2 Modified transistor level implementation of variable node #1 in Figure 2-2.

In general, for implementing a variable node of degree n (n>1) based on this modified method, we need n+2 input sections and 3n+2 output sections of current buffers and overall, 4n+4 sections are used. If we use cascode current mirrors similar to those presented in the previous chapter, the total number of transistors would be 16(n+1). When n=1, the variable node would be very simple and we only need 20 transistors to implement it. Figure 5-3 shows the difference between these two methods for different variable node degrees, and it is clear that for variable nodes with many interconnections, the modified approach can significantly reduce the number of required transistors. Furthermore, it is worth noting that while power consumption in the input and output parts of the variable nodes in these two methods are almost equal, the difference between the consumed powers in the intermediate sections could be considerably different. In the original method it is simple to show that the maximum current that is drawn from the power supply in the intermediate section is equal to:

$$I_{\max} = (n+1)\left|m_V\right| + n\sum_{C'\in N_V}\left|m_{C'\to V}\right|. \tag{5-3}$$

However, for the modified method this current would be:

$$I_{\max} = \left|m_V\right| + 2\sum_{C'\in N_V}\left|m_{C'\to V}\right|. \tag{5-4}$$

Consequently, in addition to saving in area consumption, we have significantly reduced the power consumption without any penalty in the latency of the variable node.

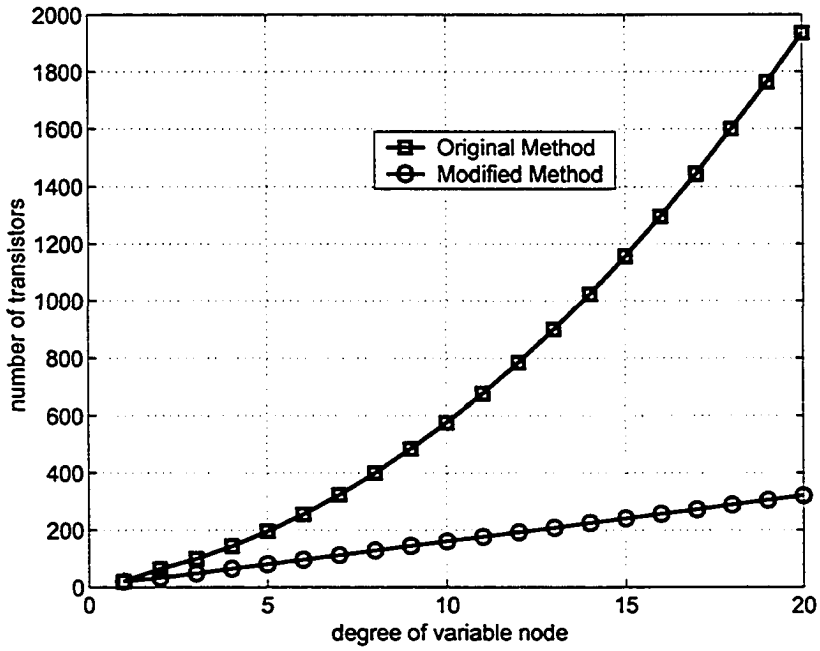


Figure 5-3 Comparison between the number of transistors in the original method and modified method for implementing a variable node of degree *n*.

5.2 High-Degree Check nodes

In our original method, in a check node of degree n, we have n RTAS modules, n ASTR modules, n min WTA modules, and n XOR gates. While RTAS and ASTR modules would not change as we increase n, the complexity of each min WTA module and XOR gate increases at least linearly with respect to n. This however, makes the overall complexity proportional to n^2. If we use the circuits that were proposed in the previous chapter for implementing a check node of degree n, then for each min WTA circuit we should use, $5n$-2 n-type and $2n$ p-type transistors that adds up to a total of $7n$-2 transistors. This means that for a check node of degree n, we would use $7n^2$-$2n$ transistors in WTA circuits.

For implementing XOR gates, we assume that we have cascaded 2-input XOR gates. Needless to say it is possible to implement DCVSL XOR gates with a couple of inputs and we only consider 2-input gates to simplify the discussion. Also, it is worth noting that DCVSL gates are generally much faster than the rest of our circuits and consequently cascading them would not decrease the overall speed. In order to implement an (n-1)-input XOR gate by cascading 2-input XOR gates, we need n-2 gates. As we need n XOR gates with n-1 inputs, the number of gates would be equal to $n(n-2)$ and therefore $8n(n-2)$ transistors are required. As an example if we assume n=20, then we need 2760 transistors for WTA circuits and 2880 transistors for XOR modules.

XOR modules can be greatly simplified by implementing only one n-input XOR gate and then XORing its output with every received sign bit to generate the corresponding sign bit for the outgoing message. This can be done because for any binary variables d_1 and d_2 we have:

$$(d_1 \oplus d_2) \oplus d_1 = d_2. \tag{5-5}$$

Figure 5-4 shows how this simple modification would change the structure of a check node that was presented earlier in Figure 4-25. In this method we need only one n-input XOR gate plus n 2-input XOR gates. The number of required transistors is then given by $8(2n-1)$ which for large n could be much smaller than what we had for the original method. Figure 5-5 shows the difference between these two methods for different n, the difference between the number of transistors in high-degree nodes could be quite significant and amount of saving in silicon area is noticeable. In the modified method as we reduce the number of gates, the overall power consumption decreases as well. This is however at the expense of further delay because of at least one extra XOR operation. How-

ever, if n is large, we have to use many XOR gates and introduced latency due to the additional XOR could be negligible.

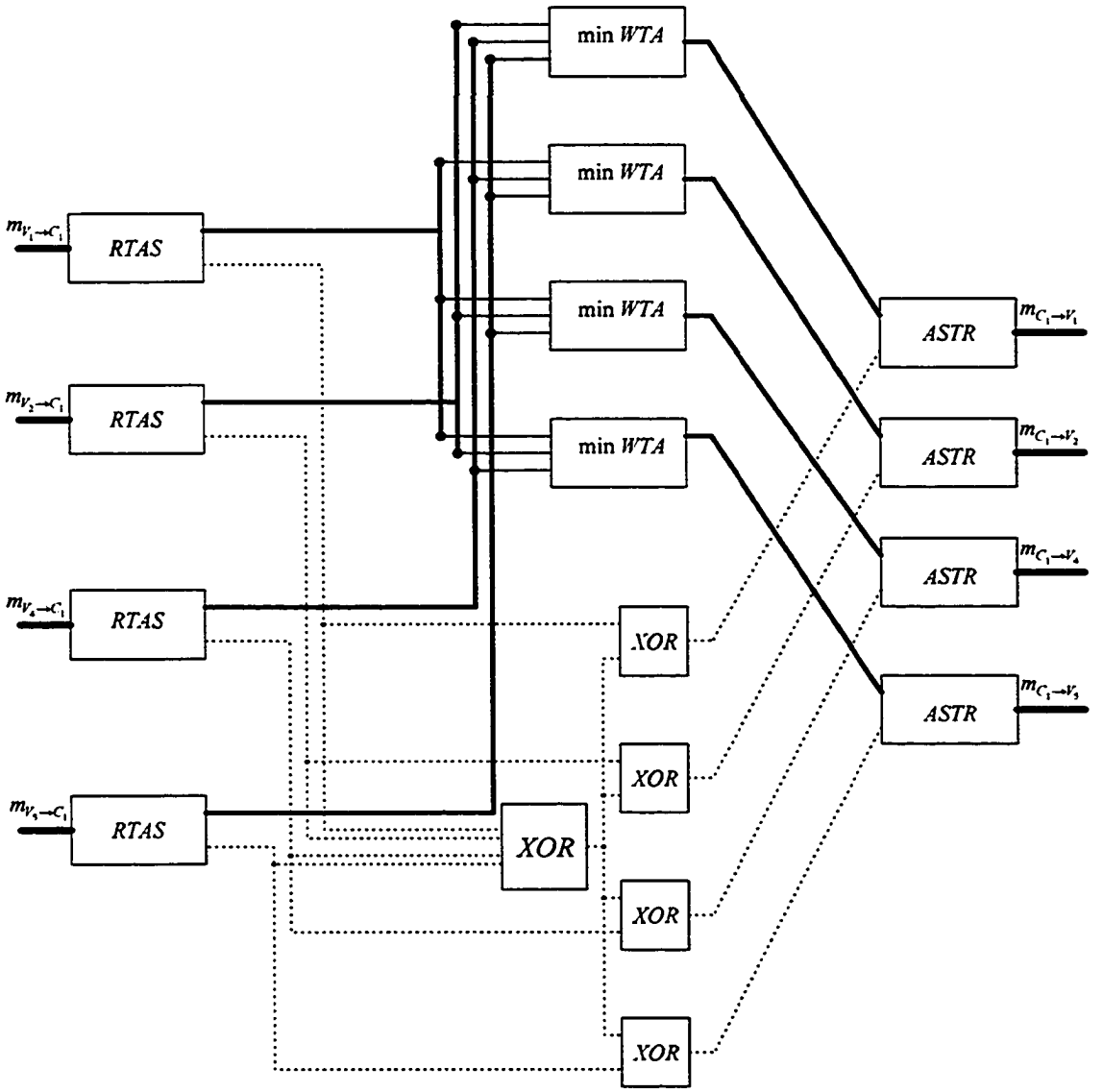


Figure 5-4 Structure of check node #1 in Figure 2-2, with modified XOR module.

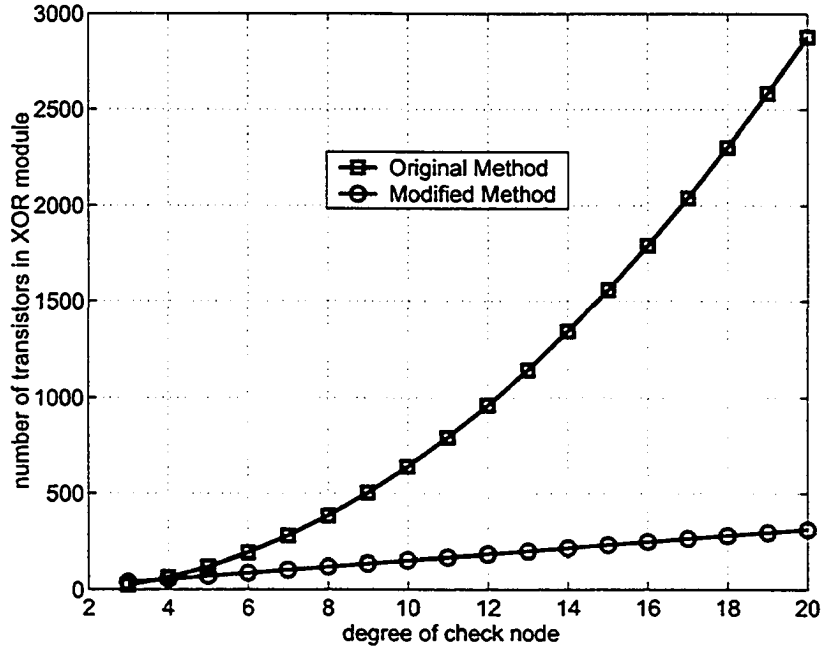


Figure 5-5 Comparison between the number of transistors in the original method and modified method for implementing an XOR module of degree *n*.

We can further simplify check nodes by noting that min WTA modules can only have two possible outputs, which are either equal to the first or the second minimum input. Therefore, it is better to only find the first and the second minimum inputs and then properly connect them to the output terminals. In fact, the operation in the min WTA module can be simply expressed as follows:

$$\left|m_{C\to V}^{(l)}\right| = \begin{cases} 1^{st} \min\limits_{V'\in N_c}\left(\left|m_{V'\to C}^{(l)}\right|\right) & if \left|m_{V'\to C}^{(l)}\right| \succ \min\limits_{V'\in N_c}\left(\left|m_{V'\to C}^{(l)}\right|\right) \\ 2^{nd} \min\limits_{V'\in N_c}\left(\left|m_{V'\to C}^{(l)}\right|\right) & if \left|m_{V'\to C}^{(l)}\right| = \min\limits_{V'\in N_c}\left(\left|m_{V'\to C}^{(l)}\right|\right) \end{cases} \tag{5-6}$$

Interestingly, by minor modifications we can convert a first max/min WTA circuit to a second max/min WTA circuit [93]-[94]. Figure 5-6 shows a simple second max WTA circuit which is the modified version of Figure 4-29.

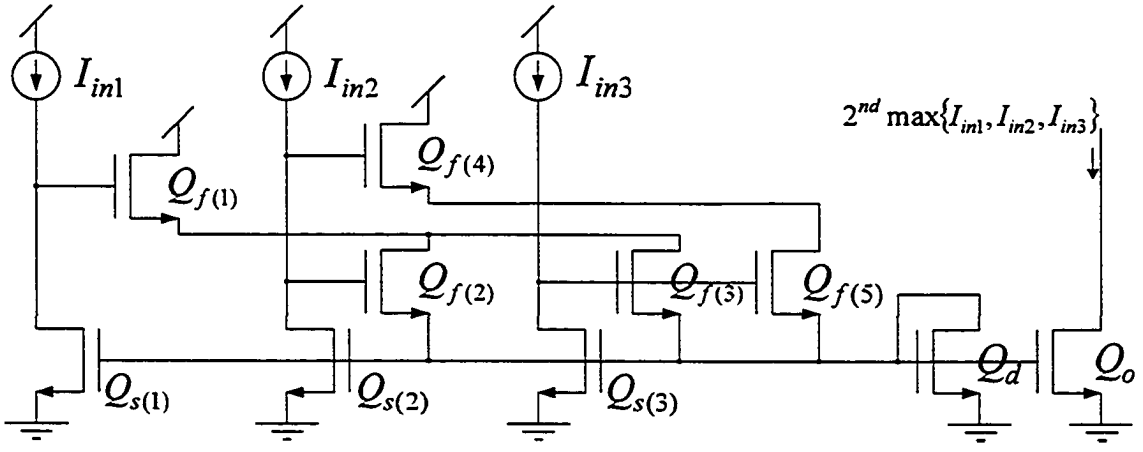


Figure 5-6 A simple 3-input current-mode second max WTA circuit.

The main difference between a second min/max WTA and a first min/max WTA is in its feedback path that has two $Q_{f(.)}$ transistors instead of one. This means that while in a first min/max WTA circuit at least one $Q_{f(.)}$ was on, in a second min/max WTA at least two $Q_{f(.)}$ transistors should always be on to keep the negative feedback functional. In a second min/max WTA circuit the feedback path, in addition to these $Q_{f(.)}$ transistors, include one $Q_{s(.)}$ transistor, which is similar to what we already observed for the first min/max WTA circuits. Also while in a first min/max WTA at least one $Q_{s(.)}$ transistor is always biased in saturation, in a second min/max WTA circuit at least two $Q_{s(.)}$ transistors work in saturation.

To see how the second max WTA circuit in Figure 5-6 works, assume $I_{in2}>I_{in1}>I_{in3}$ and the shared gate-source voltage among $Q_{s(.)}$ transistors is very small and input currents are suddenly applied to the second max WTA circuit. The drain currents of $Q_{s(.)}$ transistors will be very small and the drain voltages of the $Q_{s(.)}$ transistors increase. This increase in the voltages happens because these transistors have not been biased properly to accept the input currents. Since by assumption I_{in2} is the maximum input current, the drain voltage for the $Q_{s(2)}$ transistor would be the maximum drain voltage among

$Q_{s(.)}$ transistors. This means that $Q_{f(2)}$ and $Q_{f(4)}$ will have the maximum gate voltages among $Q_{f(.)}$ transistors. Ultimately, the gate-source voltage for $Q_{f(2)}$ and $Q_{f(4)}$ will be higher than the threshold voltage. However, for these two transistors, the drain current will remain equal to zero, unless $Q_{f(1)}$ or $Q_{f(5)}$ is also turned on. This means that the drain voltages keep increasing until either $Q_{f(1)}$ or $Q_{f(5)}$ turns on. Since I_{in1} is the second largest input current, $Q_{f(1)}$ will be turned on before $Q_{f(5)}$. When $Q_{f(1)}$ is turned on, the feedback path is closed through $Q_{f(1)}$, $Q_{f(2)}$, and $Q_{s(1)}$. If the transistors in the max WTA circuit have high output impedances and the difference between the first and the second maximum input currents is not negligible, then the gate voltage of $Q_{f(2)}$ would be very high when $Q_{f(1)}$ is turned on and $Q_{f(2)}$ acts as a closed switch (it is likely that it is even biased in triode region), so the first largest input current can not control the shared gate voltage among $Q_{s(.)}$ transistors. This means that we can ignore the first maximum input and therefore the circuit reduces to a first max WTA circuit similar to what we observed earlier in Figure 4-29. It is worth noting that when $Q_{f(1)}$ is turned on, the shared gate voltage among $Q_{s(.)}$ transistors will be controlled and set by feedback and regenerates the second winner at the output. All the losers that can not afford that much current go to triode. On the other hand, for the first winner, drain voltage of the $Q_{s(.)}$ transistor goes up to obtain a higher drain current and at the same time limit the incoming current from the current source. It is interesting to note that if I_{in1} surpasses I_{in2}, the feedback path is closed through $Q_{f(1)}$, $Q_{f(2)}$, and $Q_{s(2)}$ and the I_{in2} appears at the output and when $I_{in2} = I_{in1}$, the output current will be equal to I_{in2}.

Figure 5-7 shows circuit simulation for the second max WTA of Figure 5-6. In simulation, we assume all transistors are identical and have the same width and length sizes and W/L=1μm/1μm.

Figure 5-8 shows a 3-input second min WTA which has been designed based on first min WTA circuit in Figure 4-33. For an n-input second min WTA based on the same approach, we need $0.5n^2+6.5n+4$ transistors.

In addition to an n-input first min WTA and an n-input second min WTA, a suitable output stage should be used for WTA module to compare each input with the first min and when input current is equal to the first min, it should send the second min to the output and otherwise the first min is sent to the output. Figure 5-9 shows the structure of such an output stage.

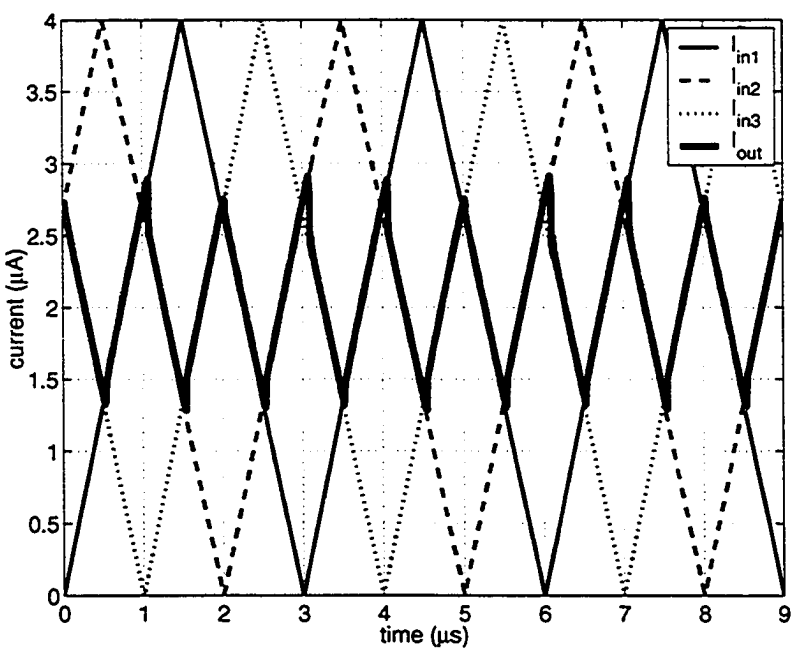


Figure 5-7 Simulation results for the second max WTA in Figure 5-6.

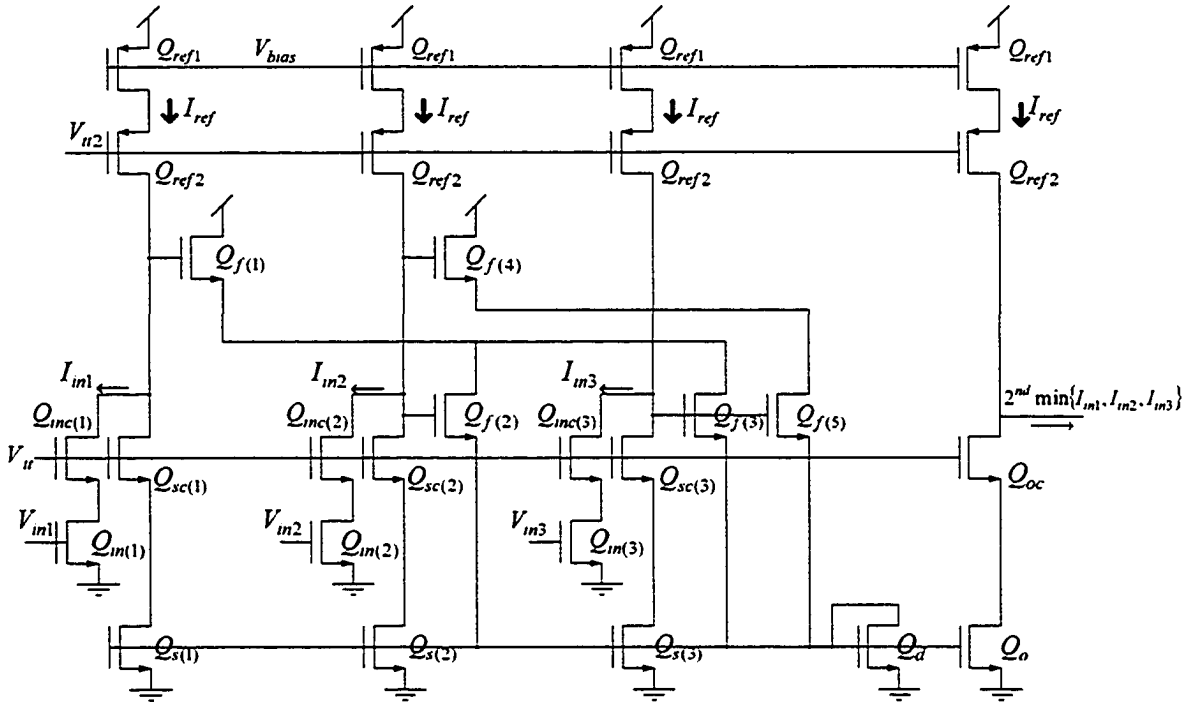


Figure 5-8 A 3-input cascode second min WTA circuit.

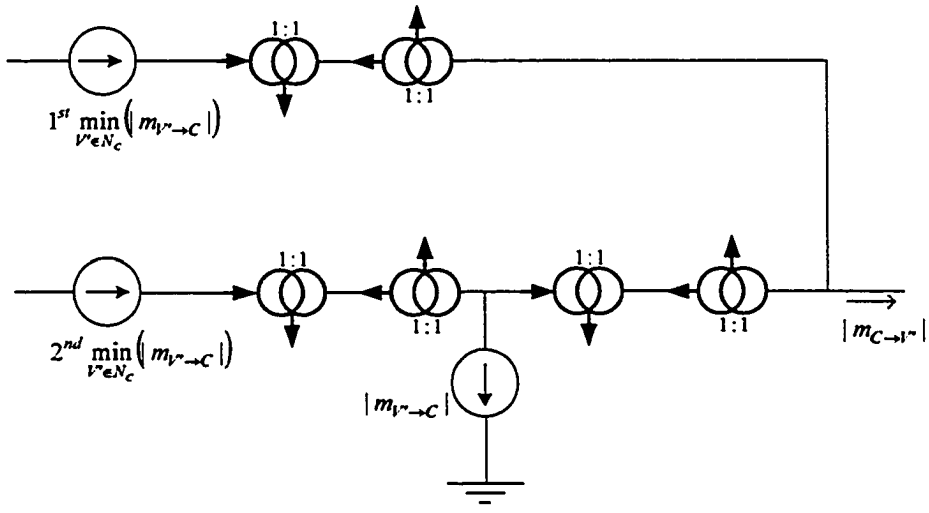


Figure 5-9 Output stage in the modified min WTA module.

In this figure, the output consists of two components; the first min current always appears at the output and is added with another current, which is a function of the difference be-

tween the second min and the corresponding input. If this difference is positive, it indicates that the input has been equal to the first min and the difference between the second min and the first min is conducted to the output and will be added to the first min to generate the second min. If the difference between the second min and the corresponding input is not positive, this difference will not appear at the output. This means that ideally the only current component at the output is the first min input. Therefore, if we substitute ideal current mirrors in Figure 5-9, this module can switch the proper signals to the output. Even in a special case when a few inputs are equal to the minimum value, this circuit remains functional.

A circuit for implementing this block is shown in Figure 5-10. In this figure V_{in} is the output voltage of the RTAS module that corresponds to the V' variable node. The first and the second min inputs are found in the first min WTA and the second min WTA circuits. Figure 5-11 shows how these blocks are placed and utilized in a circuit for implementing the check node #1 in Figure 2-2.

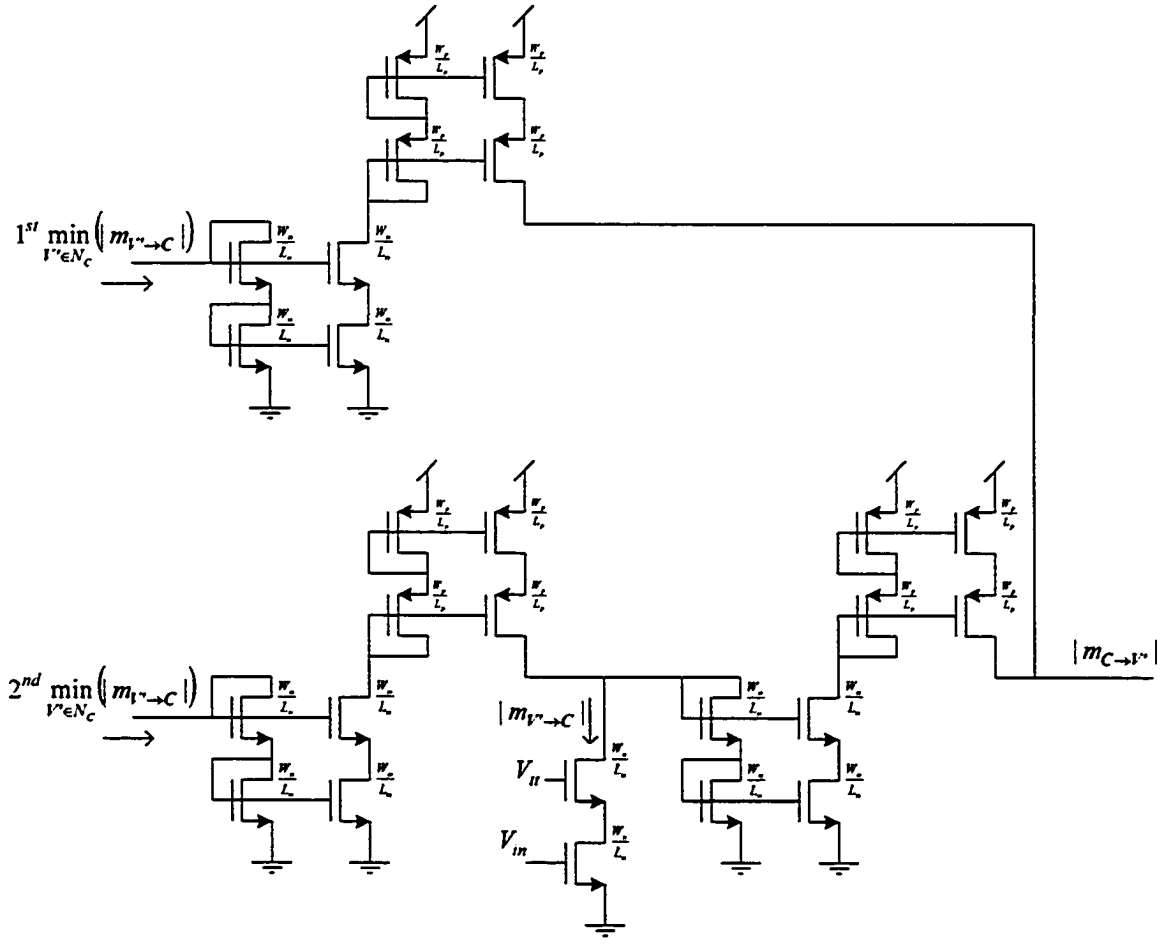


Figure 5-10 Transistor level implementation of Figure 5-9.

If we use the cascode configuration similar to what we used in Figure 5-8 and Figure 5-10 then for implementing a min WTA module with n inputs, we need $7n+5$ transistors for implementing an n-input first min WTA circuit, $0.5n^2+6.5n+4$ transistors for implementing an n-input second min WTA circuit, and $12+14n$ transistors for implementing WTA output stage. Therefore, the total number of transistors would be $0.5n^2+27.5n+21$. This is while in the original method $7n^2-2n$ transistors were required. Figure 5-12 shows that for $n>5$ the modified approach requires fewer transistors and for larger n the difference between the modified and original approaches is quite significant.

In addition to the smaller number of transistors which implies smaller area consumption, power consumption in the modified WTA module is close to $2(n+1)I_{ref}V_{dd}$ watts which can be much smaller than power consumption in the original WTA module, which was close to $n^2 I_{ref} V_{dd}$ watts. However, the modified method has the drawback of higher latency.

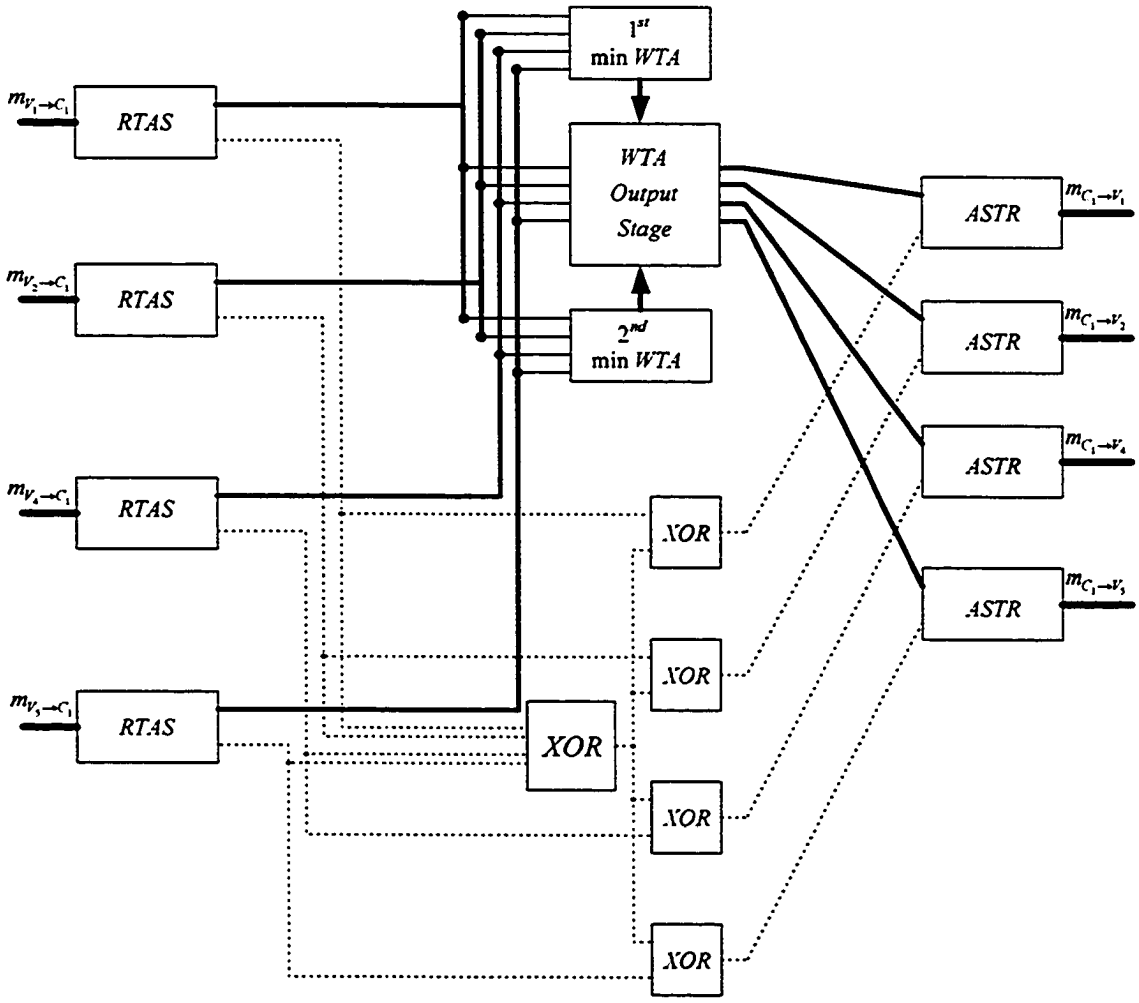


Figure 5-11 Structure of the check node #1 in Figure 2-2, with modified XOR module and modified WTA module.

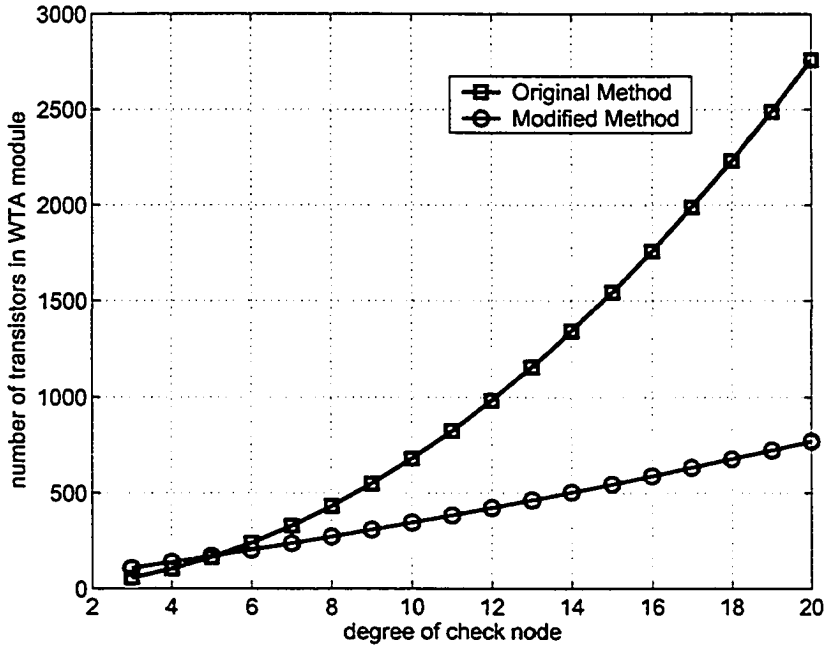


Figure 5-12 Comparison between the number of transistors in the original method and modified method for implementing a WTA module with *n* inputs.

5.3 Conclusion

In this chapter, we focused on implementation issues of analog min-sum iterative decoders, when check nodes and variable nodes with high degrees appear in the Tanner graph representation of the code. While it is possible to use the original approach for this case too, we presented circuits for implementing these nodes with smaller number of transistors and consequently less silicon area consumption. In addition, the proposed approach reduces the power consumption considerably.

Chapter 6

Low-Voltage Min-Sum Analog Decoder

In the previous chapters, we presented a current-mode approach for implementing analog min-sum iterative decoders. We extensively used current mirrors to build the processing modules. The only important block in our design that does not seem to work based on current mirrors is the min WTA module. In this chapter, we show how a current-mode min WTA circuit can be constructed by current mirrors as well.

It is important to be able to represent all the major blocks in our design by a combination of current mirrors because in virtually every technology, accurate current mirrors can be built. As an example, it is well known that design of analog circuits becomes extremely difficult and tricky in modern fabrication technologies, when transistors suffer from short channel effects and supply voltages are small. Current mirrors are commonly used in analog circuits and they have numerous applications. Therefore, design of current mirrors has been the topic of many studies and there are quite a few current mirrors with low-voltage requirements and ongoing research in this area will lead to better current mirrors in the future. Since all the major blocks in our design are based on current mir-

rors, our proposed analog min-sum iterative decoder could then be used wherever accurate current mirrors are available.

To demonstrate how a min WTA module can tolerate the reduced supply voltage, in this chapter we use a previously reported low-voltage current mirror to design a novel high-swing low-voltage current-mode max WTA circuit that can be simply converted to a min WTA circuit and used as a min WTA module in a check node.

6.1 Introduction

There are many designs available for Max WTA circuits (see, e.g., [92]-[99]). However, new CMOS fabrication technologies with shorter channel lengths and increasing output conductance, as well as the persistent trend towards reduced supply voltages have made analog design more challenging. At the same time, many of the conventional circuit design techniques can no longer be applied under such circumstances. It is therefore of great interest to devise circuits with large input/output voltage swing, small voltage requirements and improved accuracy.

For the circuits of [92]-[98], due to the short channel effects, the accuracy degrades drastically when fabricated in more advanced CMOS technologies. Moreover, the proposed circuits are not high swing, and the minimum allowable voltage at the inputs and the output is larger than V_T+V_{eff}, where V_T is the MOSFET threshold voltage and V_{eff} is the effective voltage defined as $V_{gs}-V_T$ [70]. In [99], good accuracy is achieved by using a cascode configuration, but the voltage requirement at the inputs and at the output is still quite high and is equal to $2V_T+2V_{eff}$ and V_T+2V_{eff}, respectively.

In the next section, a method for designing max WTA circuits is introduced and a few examples are provided to show how this method can be utilized.

6.2 Design Methodology

The winner input in a max WTA circuit is duplicated at the output, so there should be some similarities between a current-mode max WTA and a current mirror. Figure 6-1 shows building blocks of a simple current mirror. Input current is sensed in the input leg of the current mirror and then it is translated into suitable biasing conditions. Negative feedback is used to efficiently do the translation task and tune the biasing condition for the input block to accept the incoming current. This biasing condition is then applied to the output leg to regenerate the input current. In this figure, *AMP* block is a high-gain trans-resistance amplifier that converts the difference between the input current and the current of the input stage to V_{out} voltage and could come from the output impedance of the input stage. Since input, amplifier, feedback, and output blocks can be in different forms and shapes, it is sometimes difficult to distinguish these blocks, especially because various blocks may overlap.

It is often possible to convert a current mirror into a current-mode max WTA circuit. Figure 6-2 shows how by modifying the current mirror of Figure 6-1, a 3-input current mode max WTA circuit can be devised.

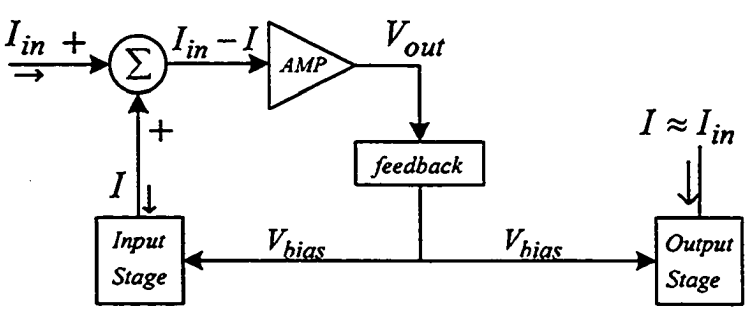


Figure 6-1 Block diagram of a simple current mirror.

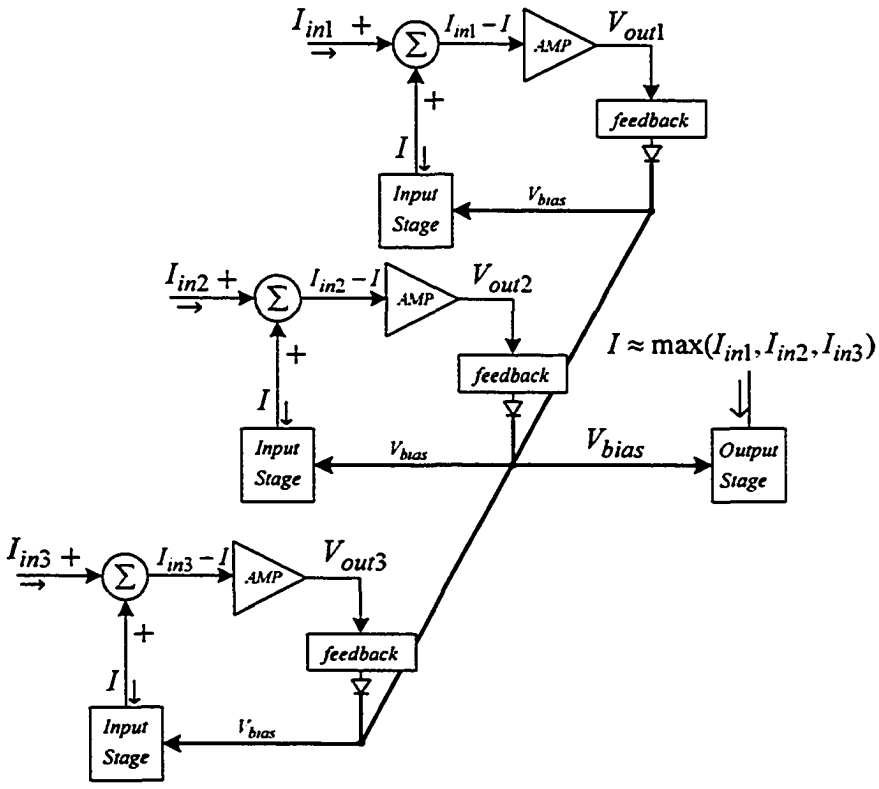


Figure 6-2 A 3-input current-mode max WTA circuit that has been designed based on the current mirror in Figure 6-1. Diodes are assumed ideal.

In Figure 6-2, the highest V_{out} belongs to the maximum input current. This guarantees that the feedback path will be closed only for the maximum input. Thus by adjusting the V_{bias}, the output will be equal to the maximum input current. Diodes in this figure make the feedback conditional. This means that only for the winner input will the feedback path be closed. This approach can be generalized and by modifying the feedback block and making it conditional, a WTA circuit can be designed, even without using any diodes.

For implementing an *n*-input max WTA circuit based on a current mirror, we use *n* input, *n AMP* blocks, and *n* feedback blocks. If the output block is not overlapping with the feedback block, we only use one output block and properly connect it to the rest of

the circuit. This is the case in Figure 6-2. If output and feedback blocks in the original current mirror are overlapping, we need n output blocks that are connected in parallel to each other. However, sometimes it is not necessary to completely repeat the output stages as some parts can be shared among n parallel output blocks. This approach will be illustrated through a few examples and a new low-voltage cascode max WTA is designed based on a current mirror with these specifications.

Figure 6-3(a) shows a well-known Wilson current mirror [71] and its basic blocks. In this current mirror, output and feedback blocks are overlapping. As the *AMP* block comes from the output impedance of the input stage, we no longer show it in the figures. If we want to construct a max WTA circuit with two inputs based on Wilson current mirror, we need two input blocks, two feedback blocks, and two output blocks and we should connect the output blocks in parallel as shown in Figure 6-3 (b). However, output and feedback blocks can share the diode-connected transistor; therefore we do not repeat it. In fact, if we use two diode-connected transistors, it can be easily verified that the output current would be two times larger than required. In this circuit, feedback is not applied unconditionally and the larger input current can bias the circuit properly to stop the feedback from the smaller input. As soon as the feedback in the loser input stops working, the circuit reduces to a Wilson current mirror. The max WTA in Figure 6-3 (b), was first proposed in [92] and in Figure 4-29 we presented a 3-input max WTA of this type. Figure 6-4 shows an improved Wilson current mirror and its corresponding 2-input max WTA. The same argument that was presented for Figure 6-3 can be applied to this circuit as well. The max WTA based on improved Wilson current mirror was proposed as a novel max WTA circuit in [99].

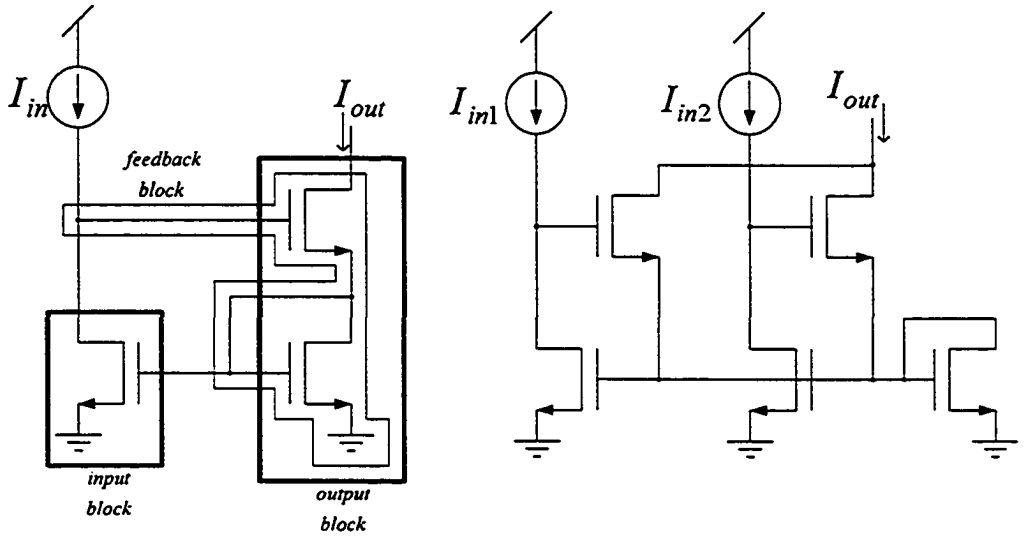


Figure 6-3 (a) Wilson current mirror (b) A 2-input current-mode max WTA based on Wilson current mirror.

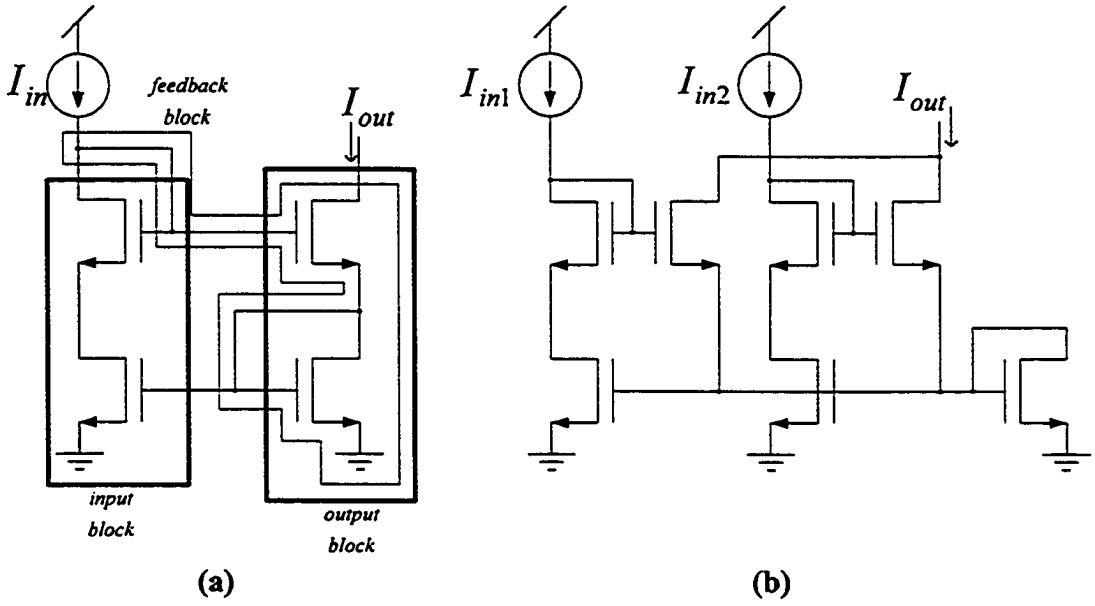


Figure 6-4 (a) Improved Wilson current mirror (b) A 2-input current-mode max WTA based on improved Wilson current mirror.

Figure 6-5 (a) shows a simple current mirror and its building blocks. In this figure, the feedback block is simply a wire connection and works unconditionally. Therefore, we should modify it and make it conditional. Based on the conditional feedbacks that we already saw in Figure 6-3 and Figure 6-4, we can construct the max WTA circuit of Figure 6-5 (b). It is worth noting that as there is no overlap between the feedback and output blocks, we simply use one output block in the constructed max WTA circuit. The constructed max WTA is yet another variant of the max WTA of [92] that was shown in Figure 4-29.

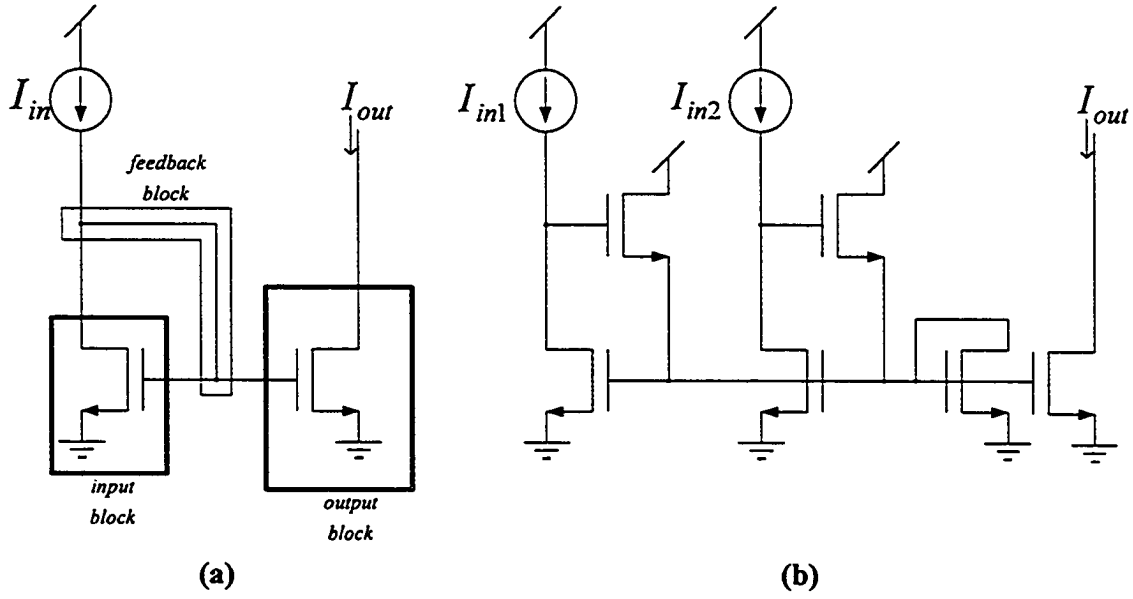


Figure 6-5 (a) a basic current mirror (b) A 2-input current-mode max WTA based on basic current mirror.

Figure 6-6 shows a cascode current mirror and its corresponding max WTA circuit, which has been constructed by replacing the unconditional feedback with a conditional feedback block. In the same way, Figure 6-7 shows how a high-swing cascode current mirror can be used for designing a high-swing cascode max WTA. We extensively used this max WTA circuit in the previous chapters.

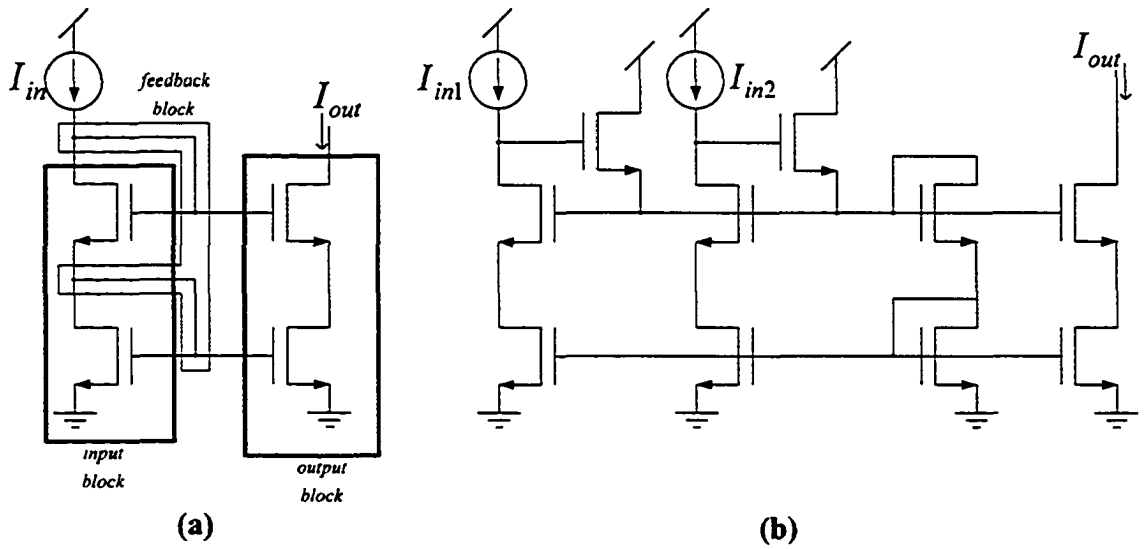


Figure 6-6 (a) Self- biased cascode current mirror (b) A 2-input current-mode max WTA based on self -biased cascode current mirror.

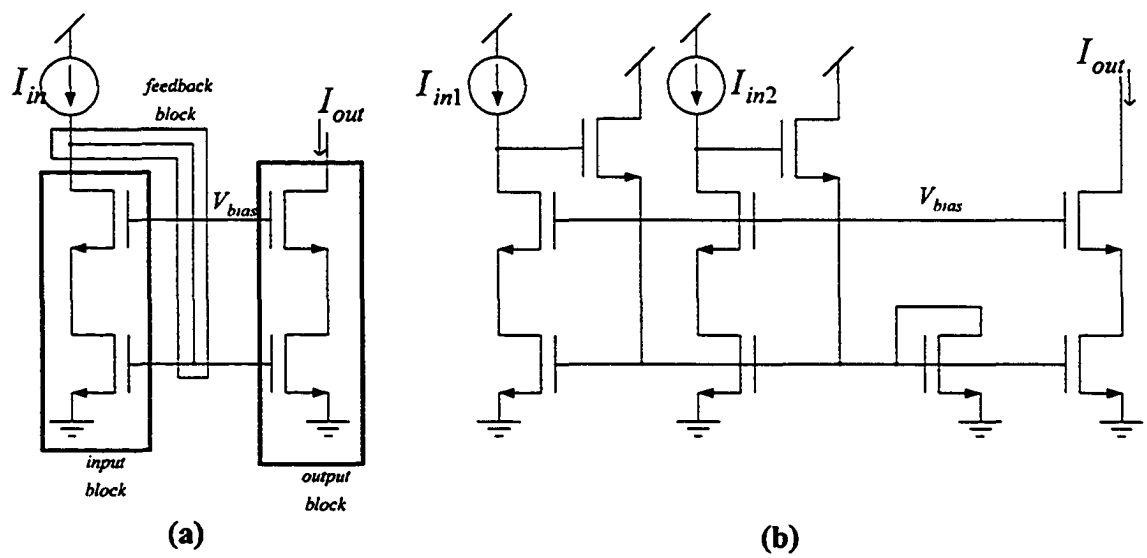


Figure 6-7 (a) A high-swing cascode current mirror (b) A 2-input current-mode max WTA based on high-swing cascode current mirror.

In the next section, we introduce a Max WTA circuit which has high accuracy, high swing and low input and output voltage requirements. To the best of our knowledge, none of the previous work in this area has addressed both the problems due to the short channel effects, and the low-voltage and high-swing requirements, together.

We start the design process by looking for a current mirror which satisfies our desired specifications, and by the approach that was introduced earlier, we convert it into a suitable max WTA circuit. Figure 6-8 shows a current mirror that was proposed in [100] and has both high accuracy and very low-voltage requirement. In this circuit feedback is conditional and therefore we do not modify it. In the next section, we show how a max WTA circuit can be constructed based on this current mirror and investigate its performance and derive some conditions that should be taken into account during the design process.

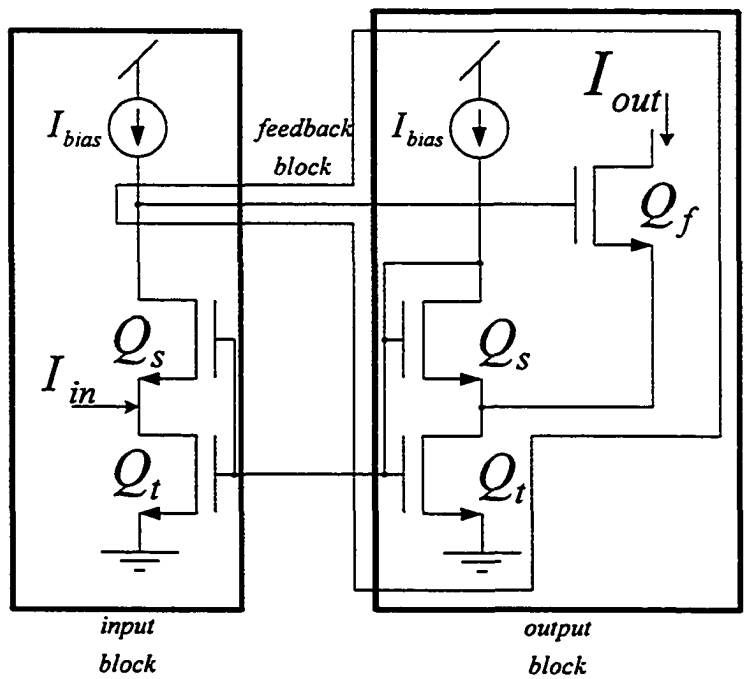


Figure 6-8 A current mirror with low voltage requirements.

6.3 Circuit Description and Operation

Our design for a Max WTA circuit is shown in Figure 6-9. This circuit has m input currents: $I_{in(1)}$, $I_{in(2)}$, ..., and $I_{in(m)}$. The output current I_{out} is expected to be equal to $Max\{I_{in(1)}, I_{in(2)}, ..., I_{in(m)}\}$. Corresponding to each input, there is a cell with 3 transistors. There is also an "output" cell (cell # $m+1$) with 2 transistors. Transistors with the same index (s, t or f) are identical. The biasing current I_{bias}, flowing through each $Q_{s(.)}$ transistor, is fixed. All $Q_{s(.)}$ transistors share the same gate voltage and if they operate in saturation, they also share the same source voltage. It is also obvious that $Q_{t(.)}$ transistors always operate in triode region, since their gate-drain voltages are higher than V_T.

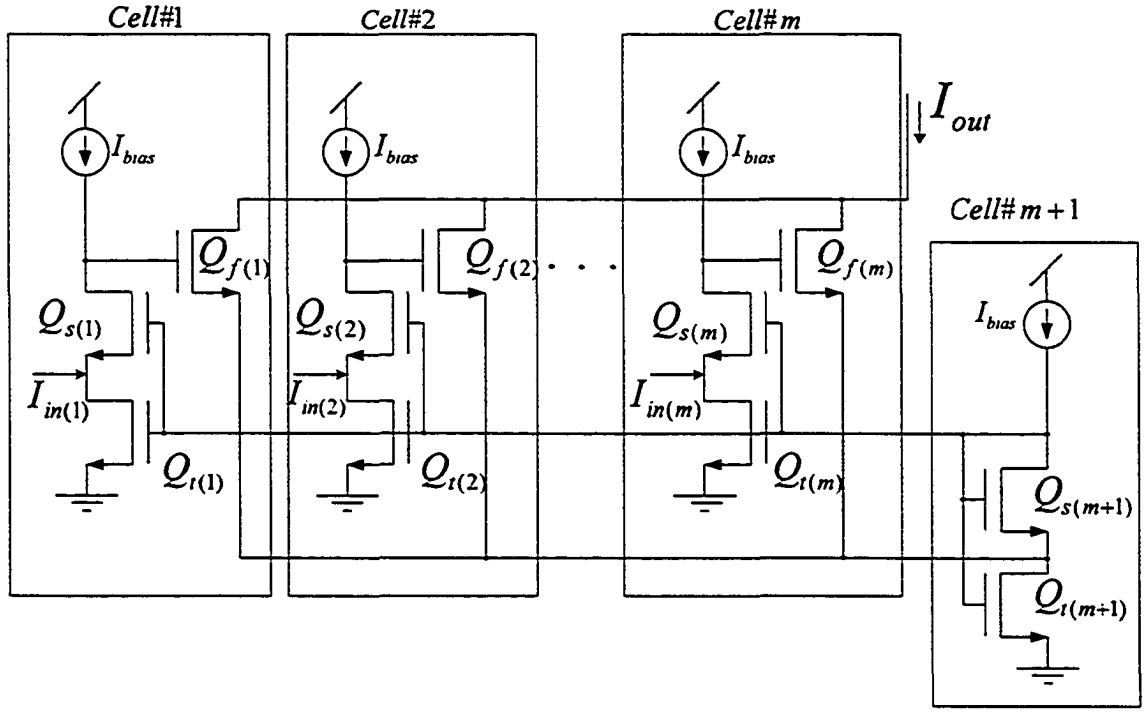


Figure 6-9 Proposed maximum winner-take-all circuit.

As will be shown later in this section, the result of the competition of cells 1,2, ..., m is a shared gate voltage for all $Q_{t(.)}$ and $Q_{s(.)}$ transistors, which is imposed by the maxi-

mum input current. It should be noted that the drain voltage of $Q_{t(m+1)}$ and that of the winner $Q_{t(.)}$ transistor(s) are equal since the corresponding $Q_{s(.)}$ transistors are biased in saturation and have the same drain current and consequently have the same source voltages (For the winner $Q_{s(.)}$ transistor(s) to work in saturation, I_{bias} must be chosen properly as will be discussed later. For $Q_{s(m+1)}$, however, this is guaranteed as it has a diode-connected configuration). Having the same gate, source and drain voltages, $Q_{t(m+1)}$ and the winner $Q_{t(.)}$ transistor(s) will have exactly the same drain current regardless of their region of operation (which is in fact triode) and regardless of how nonlinear and complicated the behavior of the transistors are. As a result, $Q_{t(.)}$ transistors can be built very small or with a high conductance to reduce the voltage requirements at the inputs and the output. However, to limit the mismatch problem it is better to use non-minimum transistors.

The shared gate voltage forces all $Q_{t(.)}$ transistors to have the same drain current and consequently for loser inputs, corresponding $Q_{s(.)}$ will have to leave the saturation region and operate in triode region. This will decrease the drain voltage of loser $Q_{s(.)}$ transistors and will push the corresponding $Q_{f(.)}$ to cutoff. Winner $Q_{f(.)}$ transistor(s) inject a current equal to the maximum input current to the drain of $Q_{t(m+1)}$. In fact, a negative feedback is formed by $Q_{s(m+1)}$, $Q_{t(m+1)}$, and the winner $Q_{f(.)}$, $Q_{t(.)}$ and $Q_{s(.)}$ transistors. This adjusts the shared gate voltage of $Q_{t(m+1)}$ and makes its drain current equal to $I_{bias}+ I_{max}$.

For the proposed circuit to function properly, I_{bias} is chosen to bias the winner $Q_{s(.)}$ in saturation. Assuming that the *k-th* input is the winner, then we must have:

$$V_{gd,Q_{s(k)}} \leq V_T . \tag{6-1}$$

This along with

$$V_{gd,Q_{s(k)}} = V_{gs,Q_{s(m+1)}} - V_{gs,Q_{f(k)}}, \tag{6-2}$$

results in the condition :

$$V_{gs,Q_{s(m+1)}} \leq V_T + V_{gs,Q_{f(k)}} . \tag{6-3}$$

Since $V_{gs,Q_{f(k)}}$ is always greater than V_T, this condition will be satisfied if I_{bias} is chosen such that:

$$V_{gs,Q_{s(m+1)}} \leq 2V_T . \tag{6-4}$$

To improve the accuracy of the circuit and to reduce the power consumption, one would like to decrease I_{bias} as much as possible. This however, decreases the speed of the circuit. A compromise has to be made depending on the application. One can see that the input voltage to the circuit of the proposed max WTA circuit can be as low as V_{eff,Q_i}, which can be very small since $Q_{t(.)}$ transistor can be made small with low output impedance. On the other hand, for the output, the minimum voltage is equal to $V_{eff,Q_f} + V_{eff,Q_i}$. Also, since the circuit has a cascode configuration, the output impedance is much larger than that of a single-stage configuration, and can be approximated by [100]:

$$r_{out} \approx g_{Q_f} r_{o,Q_f} (r_{o,Q_{s(m+1)}} \| r_{o,Q_{t(m+1)}}), \tag{6-5}$$

where r_o and g are the output impedance and the transconductance of the transistors, respectively. Output impedance and therefore accuracy of this WTA circuit is smaller than the high-swing cascode WTA circuit that was shown in Figure 6-7 because in that circuit all the transistors were biased in saturation but here $Q_{t(m+1)}$ is in triode and therefore its output impedance is much smaller than what it could be if this transistor had been biased

in saturation. However, this new WTA has much lower voltage requirement in its inputs and output.

6.4 Simulation Results

The proposed WTA circuit is simulated by Cadence's Spectre simulation tool and 0.13μm UMC (United Microelectronics Corporation) CMOS models. Mismatch as a common problem in all WTA circuits has been ignored. Here, we considered a max WTA with four inputs. The four input signals are chosen to be shifted versions of a periodic triangular current source, which has a period of 100ns, and varies between 0 and 50μA.

Transistor sizes of W/L = 600nm/130nm, 600nm/300nm, and 1μm/150nm, are chosen for $Q_{t(.)}$, $Q_{s(.)}$ and $Q_{f(.)}$ transistors, respectively. We use a supply voltage of 1.2 volts, and a bias current I_{bias} of 10μA. At the output, the voltage is fixed at 0.3 volts to demonstrate the low-voltage requirement of the proposed circuit. The input and output currents are shown in Figure 6-10. As can be seen, the output current follows the maximum of the input currents very closely.

The accuracy when the output current follows one of the inputs is better than 0.1%. This degrades to about 2% when the winner input is changed. This extra degradation, referred to as "corner error" [93], is due to the transient responses of the switched $Q_{f(.)}$ transistors. Voltage variations for the gate of $Q_{t(1)}$ ($V_{gs,Q_{t(1)}}$), the drain of $Q_{t(1)}$ ($V_{ds,Q_{t(1)}}$), and the gate of $Q_{f(1)}$ ($V_{g,Q_{f(1)}}$) are also shown in Figure 6-11. It can be seen that the input voltage requirement for current variations from 0 to 50μA is very low and is less than 0.2 volts.

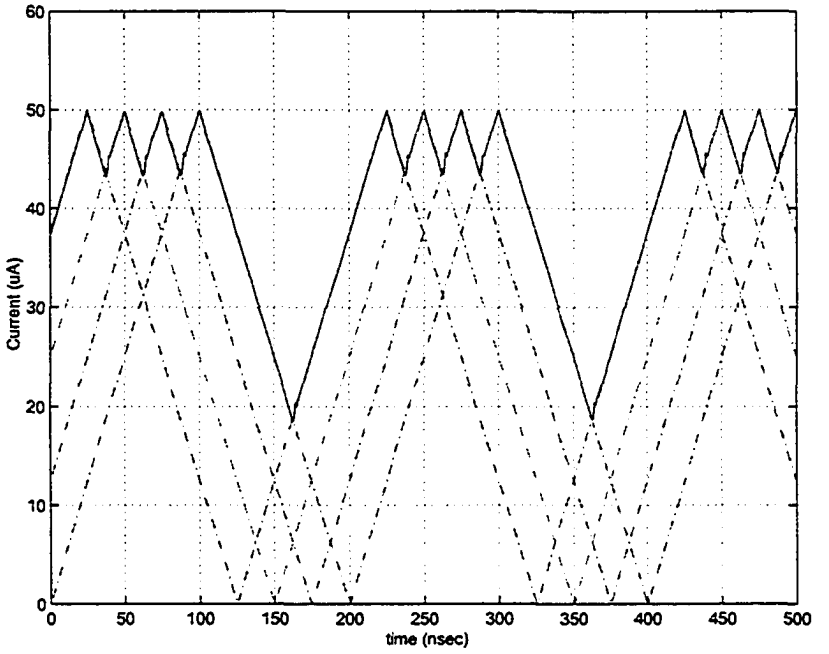


Figure 6-10 Input currents (-.-) and output current (__) in the proposed max WTA.

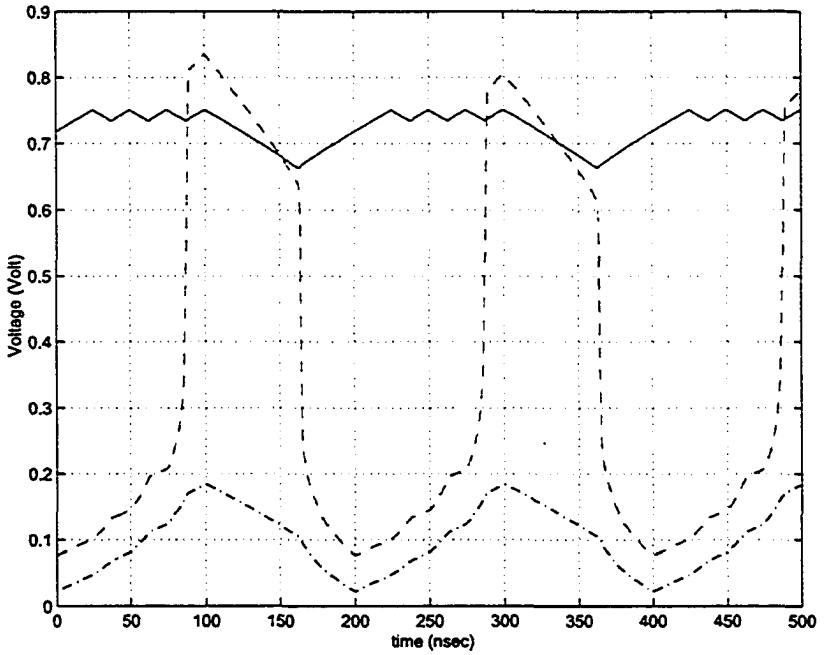


Figure 6-11 Voltage variations at the gate and drain of $Q_{f(1)}$ transistor and at the gate $Q_{f(1)}$ ($V_{gs,Q_{f(1)}}$: __ , $V_{g,Q_{f(1)}}$: –, $V_{ds,Q_{f(1)}}$: -.-).

It is worth emphasizing that all these results are based on the assumption that there is no mismatch in our circuit. Therefore, the results are too optimistic and in practice we expect lower accuracy.

6.5 Conclusion

In this chapter, we presented a method for converting a current mirror into a current-mode max WTA circuit. This is important because it shows that all the modules in the proposed approach for designing analog min-sum iterative decoder are based on current mirrors and consequently virtually in any technology that accurate current mirrors are available, we can implement analog min-sum iterative decoders. As an example of the functionality of the proposed method, we designed a low-voltage high-swing max WTA circuit. The circuit can be implemented by short channel MOS transistors and yet provides a reasonably high degree of accuracy. Simulations show a worst case error of less than 2% when there is no mismatch between transistors and/or biasing currents. Beyond analog decoding, this max WTA could be used in soft computing, and analog signal processing, in general. A Min WTA circuit can also be built based on this circuit by subtracting the input currents from a large reference current.

Chapter 7

An Analog Min-Sum Decoder Chip for a (32,8) Regular LDPC Code

In this chapter, we present the measurement results for an analog min-sum iterative decoder chip that has been designed and fabricated in 0.18μm CMOS technology based on the method proposed in this thesis. This is the first reported analog min-sum decoder and also the first CMOS analog iterative decoder in standard biasing condition. Also, it is the first time that an analog decoder chip has been used for decoding an LDPC code.

In this chapter, first we provide some information about the (32,8) LDPC code that we used for designing the analog min-sum decoder. Then measurement results are compared with the previously reported analog decoder chips and the difference between measurement results and simulation results are justified by considering imperfections in simulations and in the fabricated chip.

7.1 Features of the (32,8) LDPC Code

To prove the functionality of the proposed method for implementing analog min-sum iterative decoder, we designed a (32,8) regular LDPC code, which has a rate ¼ . The

Tanner graph of this code has 32 variable nodes and 24 check nodes each with degrees of three and four, respectively. Generator and parity-check matrices for this code are given in (7-1) and (7-2), respectively.

The minimum distance of this code is 10. This means that if we use hard-decision decoding, by means of this code at least four incorrect bits can be corrected. The Tanner graph representation of this code is shown in Figure 7-1. In the graphical representation of this code there is no cycle of length four. This Tanner graph has 96 edges and consequently 192 physical interconnections are required for connecting check nodes and variable nodes.

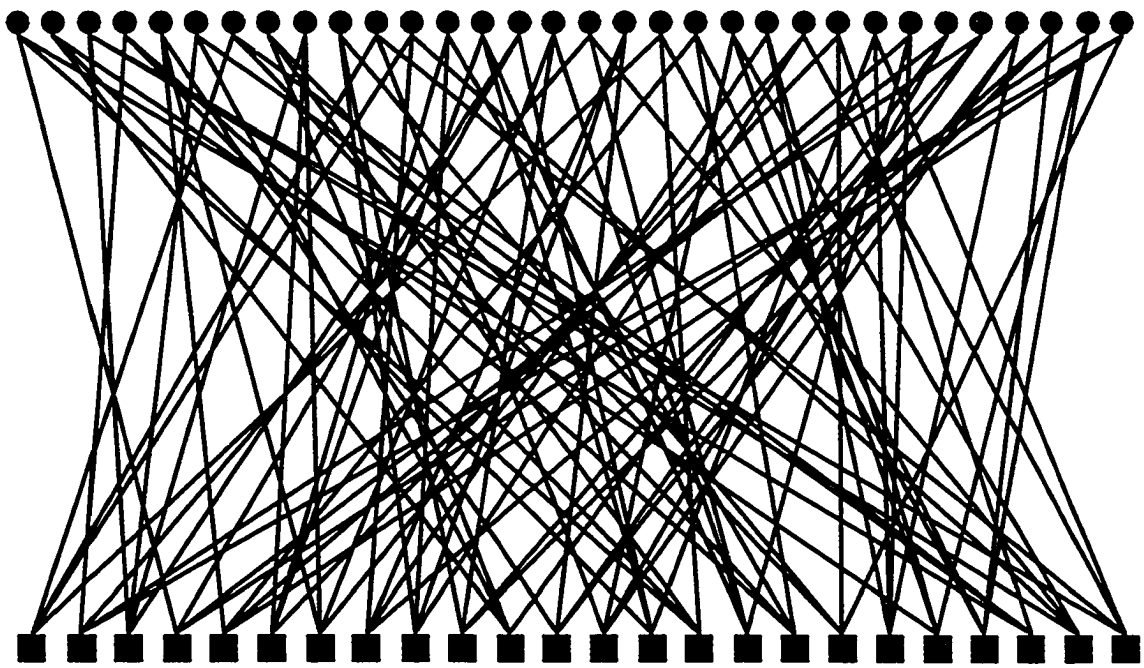


Figure 7-1 Tanner graph of the (32,8) regular LDPC code used for designing an analog min-sum decoder.

$$G^t = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7-1}$$

$$H = \begin{bmatrix}
0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1 \\
0\,0\,1\,0\,0\,1\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0 \\
0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1 \\
0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0 \\
0\,1\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0 \\
0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0 \\
1\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,1\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\
0\,0\,0\,1\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,1\,1\,0\,0\,1\,0\,0\,0 \\
0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0 \\
0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0 \\
1\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0
\end{bmatrix} \tag{7-2}$$

7.2 Implementation Issues and Measurement Results

The design methodology that was introduced earlier in this thesis was utilized for design and implementation of the variable nodes and check nodes. For designing variable nodes, we used the modified approach that was explained in Chapter 5 to reduce the power and area consumption. Since check nodes in this decoder have degree four, the

original approach presented in Chapter 4 was found more advantageous and was used for designing the check nodes. This chip has digital inputs, and for this testing the error correcting performance of the chip was simplified. Each input signal is represented by up to 6 bits, 5 bits for the magnitude and one bit for the sign bit. As we have 32 variable nodes, we used 5 bit address bus and 192 bits on-chip memory to store the received information from the channel. We used 32 5-bit current steering digital-to-analog converters to translate the input digital signals into 32 different current levels. Gain of the DAC modules can be controlled by an off-chip biasing voltage. The DACs' output currents and the stored sign bits are conducted to the ASTR modules to generate bilateral currents for applying to the corresponding variable nodes. The ASTR modules are also used in the check node module as shown in Figure 4-25.

The processing core of this analog decoder chip is 630μm by 910μm and the variable nodes and the check nodes roughly consume 60% of the core area. This relatively low *utilization factor* is due to brute-force manual connection of the processing nodes. This routing approach is far from being optimal and consumes a large area. An automatic router may significantly increase the utilization factor. Figure 7-2 shows the histogram of the interconnection lengths in this chip that looks like the distribution of interconnections in [18]. The average length of the interconnections in this chip is about 690μm.

Before starting the decoding and processing the received word from the channel, it is necessary to reset the decoder and clear up any unwanted state in the analog decoder. Otherwise, the decoder might stick to its initial state regardless of the input information. This could lead to degradation of the overall decoding performance. This phenomenon was observed in both [56] and [31]. To get rid of any unwanted initial state, the check's

outgoing interconnections are connected to the variable nodes through 96 analog switches that are controlled by an off-chip control signal. As there is no feedback when the switches are open, writing the input messages into the chip automatically clears any un-wanted memory. By closing these analog switches decoding starts.

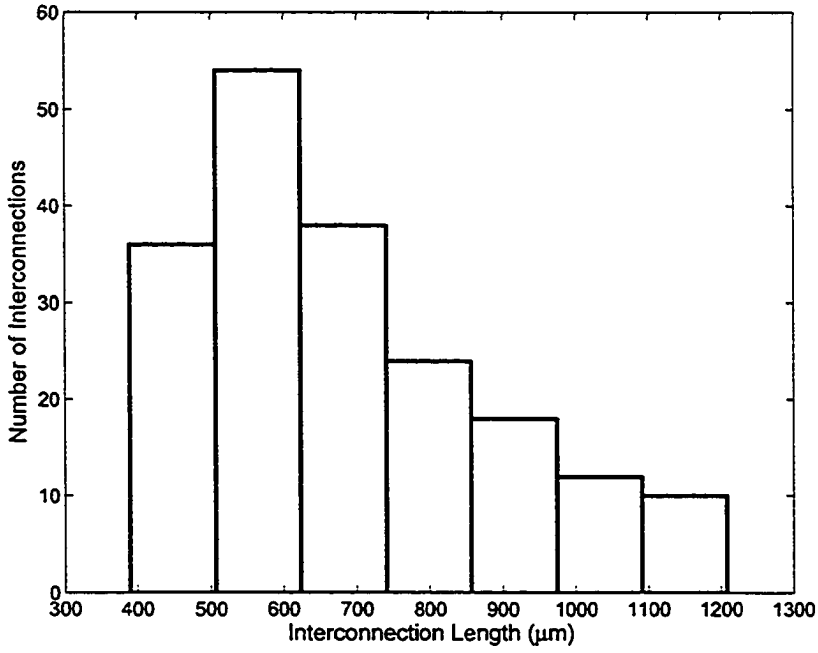


Figure 7-2 Histogram of the interconnection lengths.

To limit the disturbances induced by switching activities in digital gates, digital circuits are surrounded with well-biased guard rings. Also, separate supply and ground connections are used for biasing analog circuits, digital circuits, and substrate. Moreover, as there are not many digital gates in this circuit, and we have used low-noise DCVSL gates in the check nodes, switching noise in this circuit should be rather low. This decoder has 32 digital outputs corresponding to the final decisions that are made on the 32 variable nodes. To minimize the number of output pins, a 32:1 multiplexer is used to conduct the output of the selected variable node to the chip's output. This output signal is read and recorded in an off-chip testing module to analyze error correcting performance

of the chip. Decoding continues until analog switches are opened. To fetch a new block of data, the chip is set to be in the writing mode and the on-chip memory is updated and all the previous steps are repeated. Figure 7-3 shows the structure of this chip.

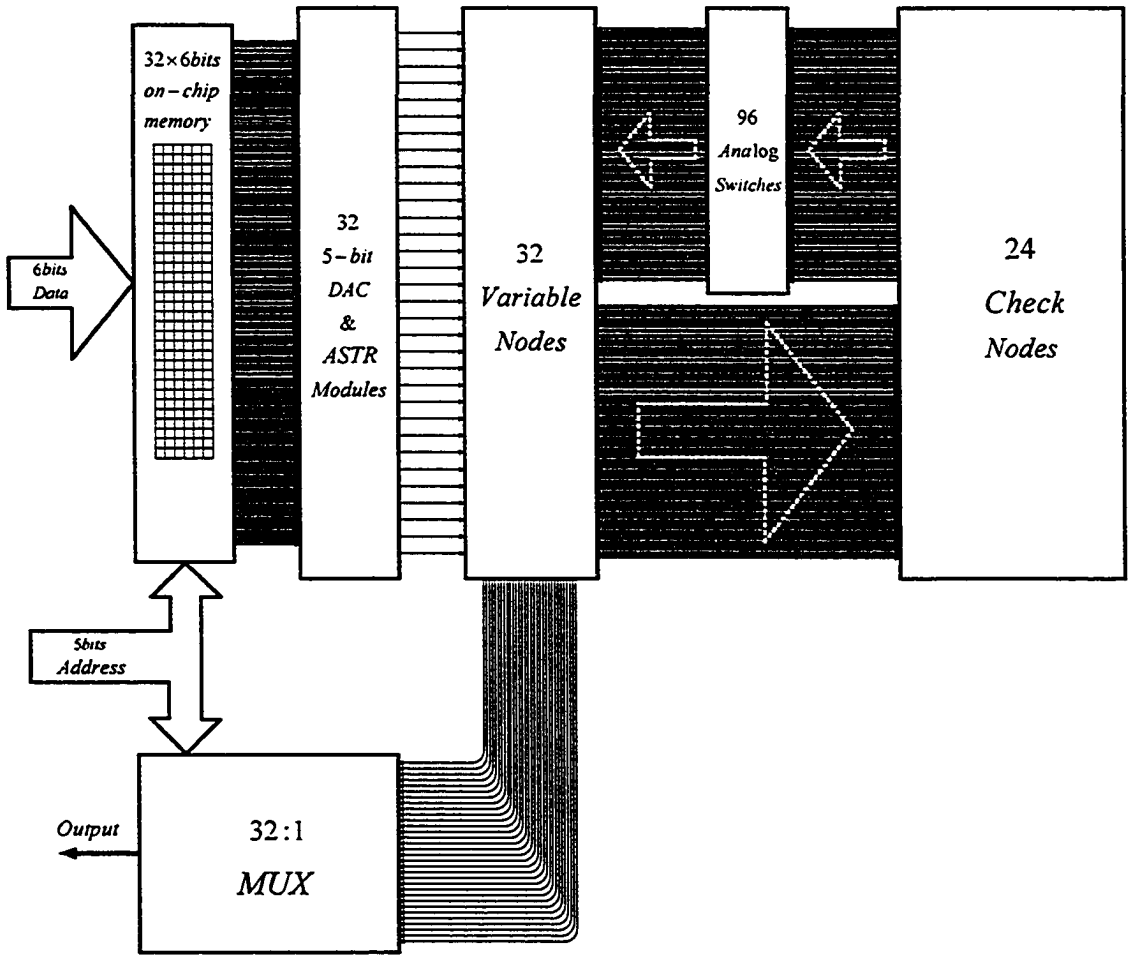


Figure 7-3 Architecture of the implemented chip.

Figure 7-4 shows the microphotograph of the fabricated analog min-sum decoder for the (32,8) regular LDPC code. This chip was fabricated in a 0.18μm, 6-layer metal, 1.8V CMOS technology. 18,800 transistors were used in this chip that includes the decoder and the digital interfaces and DAC modules. Some of the features of this chip are given in Table 7-1 along with the same information for previously reported analog iterative decoder chips. Since different codes and different technologies have been used, it is

difficult to do a fair comparison among these chips. Also, this is important to note that different definitions for throughput have been used by authors and for instance in [26] when throughput is 13.3Mb/s, loss in coding gain at BER of 10^{-3} is about 1.1dB, however in [27] for the reported throughputs this loss is between 0.2dB to 0.8dB and in [28] this loss is about 0.5dB. As it is shown later in this section, we define nominal throughput of the chip based on the settling behavior of the measured WER and BER curves for different decoding time. Based on this definition, loss in coding gain compared to the ideal digital case for our chip is about 0.3dB at a BER of 10^{-3}. Nevertheless, among the reported analog CMOS decoders in this table, our chip obviously is the fastest and has the lowest power/speed ratio. This is mainly because transistors in our design are biased in strong inversion, in contrast other reported CMOS iterative decoders in Table 7-1, are biased in weak or medium inversion.

Error correcting performance of the fabricated decoder chip was tested statistically and its functionality was verified. The best choices for DAC gain and reference currents in the WTA modules in check nodes were found by exhaustive search. To obtain reliable measurement results, in the presence of possible asymmetry in the decoding process, the error correcting performance of the chip was tested by using randomly generated codewords (not always all-zero codeword). Figure 7-5 shows steady-state word error rate curves of the chip for different number of quantization bits for the inputs. This figure shows that the best error correcting performance is achieved for 5-bit quantization. Similar trend is observed for bit error rate curves. This means that by increasing the number of quantization bits from 5 to 6 some circuits start to malfunction.

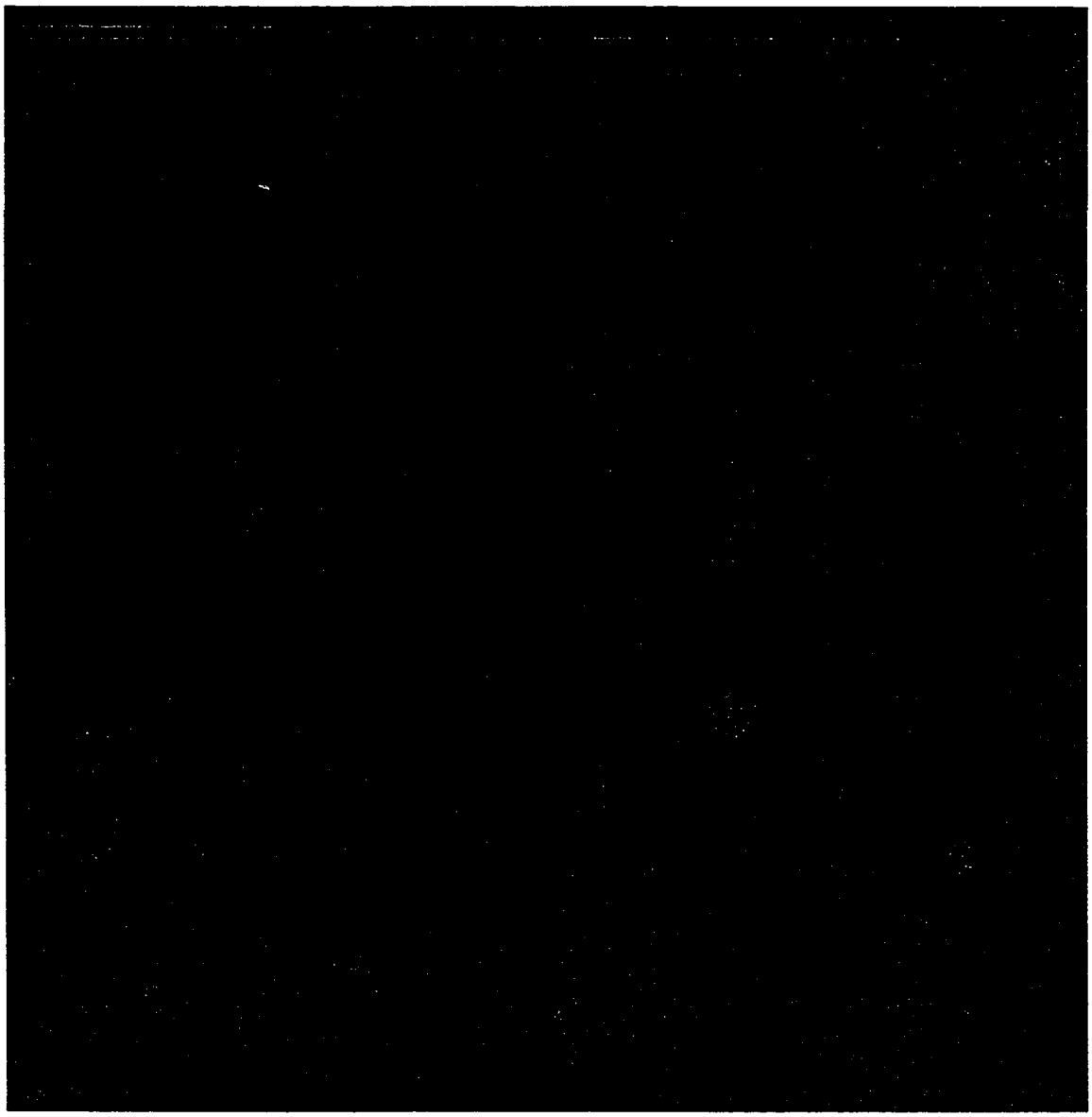


Figure 7-4 Microphotograph of the fabricated analog min-sum decoder chip.

Table 7-1 Analog Iterative Decoder Implementations.

Code	Fabrication Technology	Throughput (Mb/s)	Core (mm^2)	Average Power (mW)	Supply (V)	Number of Transistors	Power/Speed (nJ/b)
(18,9,5) Tail-biting [83]	0.8μm BiCMOS	100	2.89	50	5	BJT (940) p-MOS (650)	0.5
(16,8,3) Tail-biting [21]	0.25μm BiCMOS	320	0.12	20	3.3	BJT (441) n-MOS (356)	0.06
Turbo Code (length 16) [26]**	0.35μm CMOS*	13.3	1.32	185	3.3	CMOS	13.9
(8,4,4) Tail-biting [27]	0.5μm CMOS*	2 (0.02)	0.82	1 (0.016)	3.3	CMOS	0.5 (0.8)
Turbo Code (length 40) [29]**	0.35μm CMOS*	2	4.1	10.3 (7.6)	3.3 (2)	CMOS (26,000)	5 (3.8)
(32,8,10) LDPC Code This work	0.18μm CMOS	24	0.57	5	1.8	CMOS (18,800)	0.2

* Biased in weak or medium inversion (Subthreshold). ** Interleaver length.

On the contrary, ideally we expect to observe some improvement in the performance because quantization error decreases. As we will see later in this chapter, mismatch degrades the accuracy of the proposed modules in the analog decoder and when input currents are small or magnitudes of input currents are very close to each other, mismatch may cause a processing module to fail and ultimately degrades the overall decoding performance. This is the case when the number of quantization bits is increased for a given dynamic range and thus analog current corresponding to the least significant bit (LSB) and analog current steps become smaller. Therefore, mismatch effect becomes more significant, though quantization error decreases.

Figure 7-6 shows variation of the word error rate performance of the analog MS decoder chip versus decoding time for each word and Figure 7-7 shows similar curves for bit error rate of the chip. It is observed that by increasing the decoding time, error correcting performance improves, however, after about 1.3μs (corresponding to a throughput equal to 24Mb/s) rate of improvement becomes small and WER and BER curves settle down. If the decoding process is stopped earlier, loss in the coding gain will be significant and when throughput is about 80Mb/s the performance will be comparable with the uncoded case. Figure 7-8 shows simulated WER curves versus decoding time for the (32,8) LDPC code according to the approach that was introduced earlier in Section 3.5.3. To make a fair comparison, SR-MS with 5-bit quantization at the input and clipping similar to the chip have been used in the simulations.

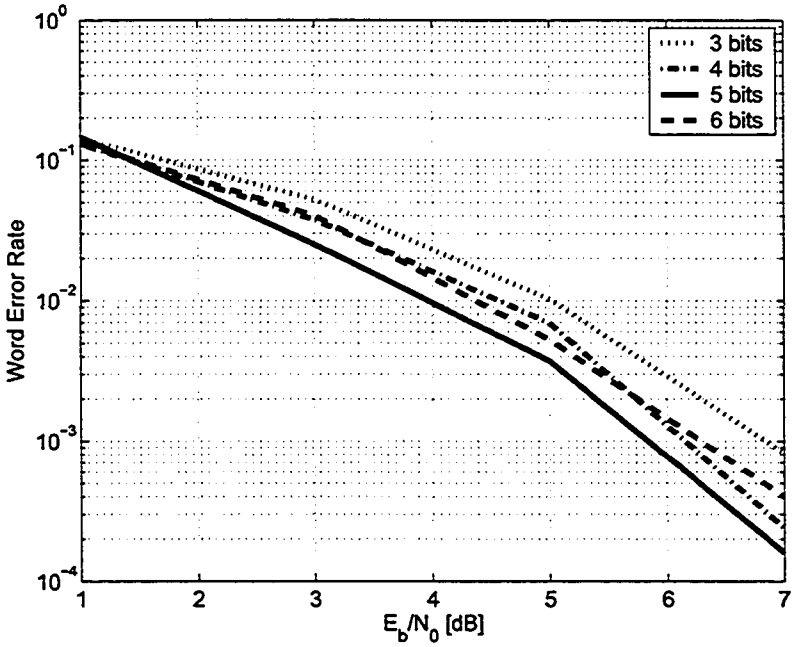


Figure 7-5 Measured steady-state WER performance of the chip for different number of input quantization bits.

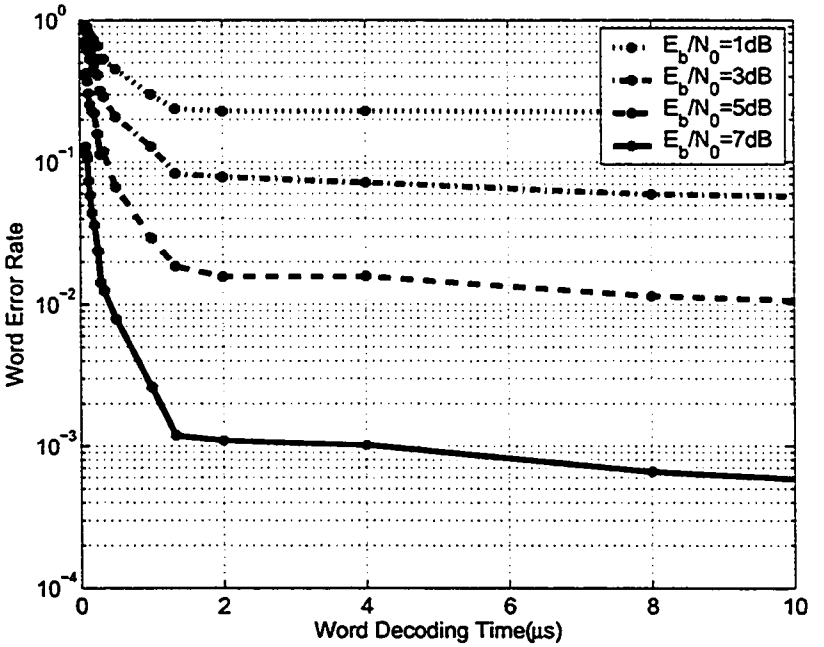


Figure 7-6 Measured WER curves versus word decoding time for the analog decoder chip.

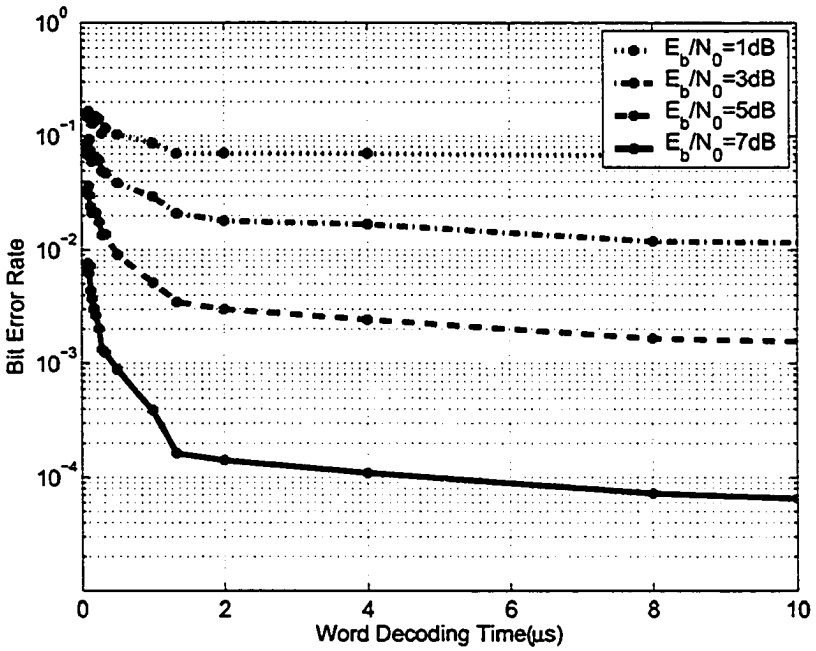


Figure 7-7 Measured BER curves versus word decoding time for the analog decoder chip.

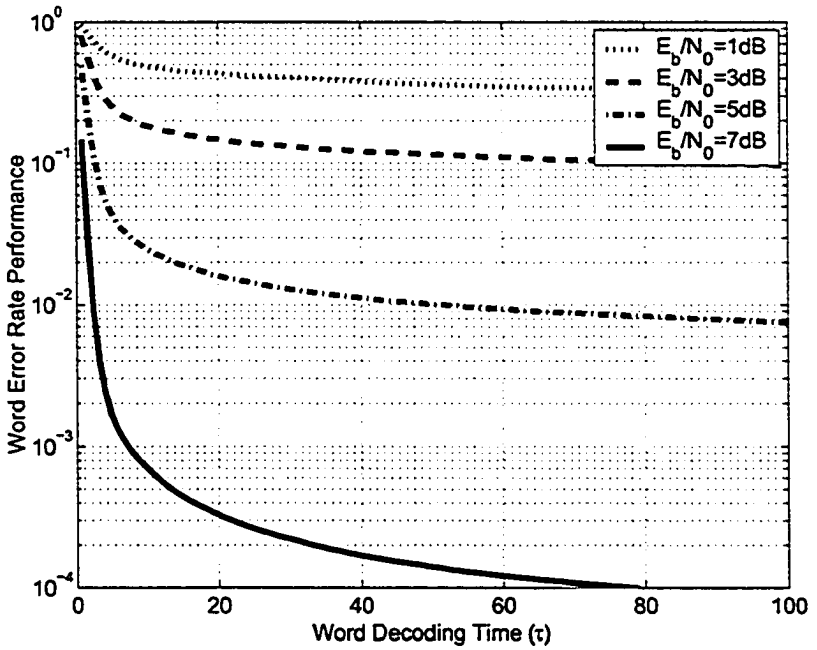


Figure 7-8 Settling behavior based on SR-MS with 5 bits quantization at the input and clipping similar to the chip.

Figure 7-9 shows word error rate curves for the decoder chip corresponding with the 5-bit quantization at the input at 250Kb/s and 24Mb/s together with simulation results for optimum ML sequence decoder and MS decoder based on conventional discrete-time model that uses successive substitution method for updating the messages (SS-MS). Also in the figure, we have the simulation results for relaxed MS (SR-MS) that uses successive relaxation method with quantization and clipping similar to the chip. The maximum number of iterations in the simulations is equal to 50,000 and optimum relaxation factor for SR-MS for this maximum number of iterations, was found to be about 0.8. In Figure 7-10 similar curves for bit error rates are shown.

At low signal to noise ratios, when the noise in the channel is considerably higher than the imperfections in the analog decoder chip, the measured curves surpass the SS-MS curves and are close to SR-MS curves. Measured error rate curves at 250Kb/s are even slightly better than SR-MS curves. However, at high signal to noise ratios the error correcting performance of the chip deteriorates. At 250Kb/s when signal to noise ratio is 7dB error correcting performance of the chip is comparable with SS-MS curves and the performance of the chip is about 1dB better than SS-MS at BER of 10^{-3}. At 24Mb/s the performance degradation is about 0.3dB at BER of 10^{-3} compared to SS-MS curve. In the next section, we will investigate effects of imperfections in implementation and in simulations and try to justify the difference between the measurement results and the simulation results based on the ideal model for the analog MS decoder (SR-MS).

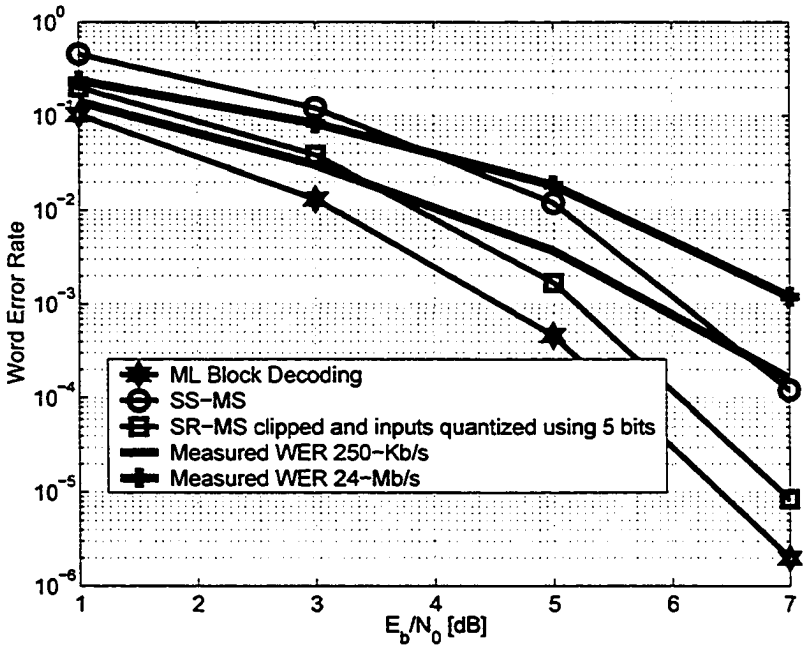


Figure 7-9 WER performance of the chip, ML decoder, ideal MS decoder (SS-MS), and ideal analog MS decoder with clipping and quantization (SR-MS).

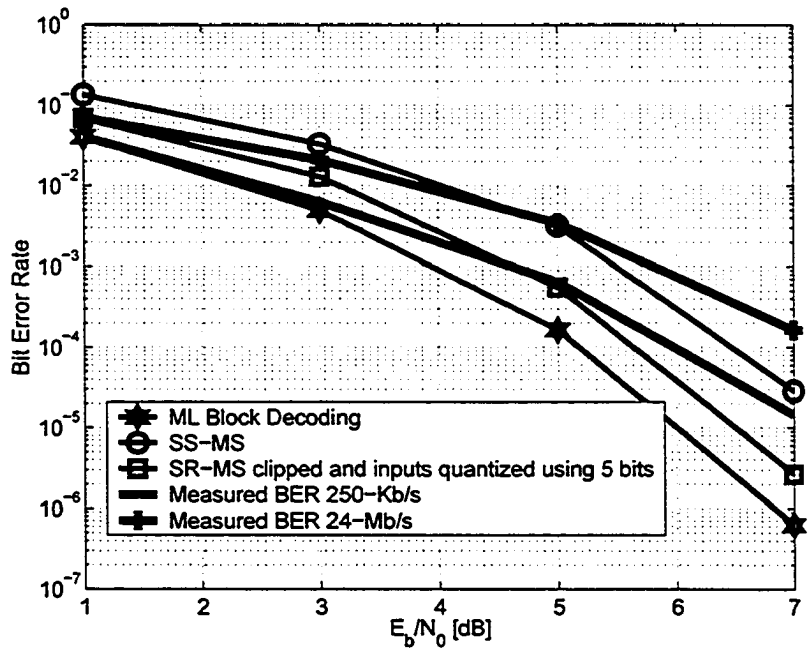


Figure 7-10 BER performance of the chip, ML decoder, ideal MS decoder (SS-MS), and ideal analog MS decoder with clipping and quantization (SR-MS)

7.3 Measurement Results versus Simulation Results

In this section, we focus on imperfections in simulations and in the analog decoder chip and explain the main reasons of deviations of the measurement results from the simulation results in low and high signal to noise ratios. We also explain in more details why increasing the number of quantization bits at the input from 5 bits to 6 bits deteriorates the error correcting performance of the chip.

7.3.1 Imperfection in Simulations

In Chapter 3, we presented a simple model for analog decoding and presented the dynamics of analog decoding by a set of nonlinear differential equations. Then forward Euler integration method was used for solving the dynamics equations and it was emphasized that when time step tends toward zero, approximation error tends to zero and the solution based on this numerical technique tends towards the solution of the dynamics equations. This means that to simulate the asymptotic behavior of an analog decoder ideally the maximum number of iterations should tend towards infinity. However, in simulations this is not practical and there is no choice but to limit the maximum number of iterations and to use larger time steps. On the other hand, in Section 3.5.3 it was shown that when the decoding time is fixed, increasing the time step can degrade the error correcting performance. Therefore, the approximation error in the integration method can be partly responsible for the discrepancy between simulation results and measurement results for low signal to noise ratios. This is where measurement results are slightly better than what is predicated by simulation based on our simple model for analog decoding for a finite maximum number of iteration. In other words, this observation is consistent with the fact

that the convergence chance of the successive relaxation method could degrade when the relaxation factor is increased.

7.3.2 Imperfections in MS Decoder Chip: DAC Modules

DAC modules are used in this chip to facilitate statistical testing for a large number of cases. However, these modules are not perfect and introduce error while translating digital inputs to bilateral analog currents. Figure 7-11 shows a block diagram of a DAC module in the MS decoder chip.

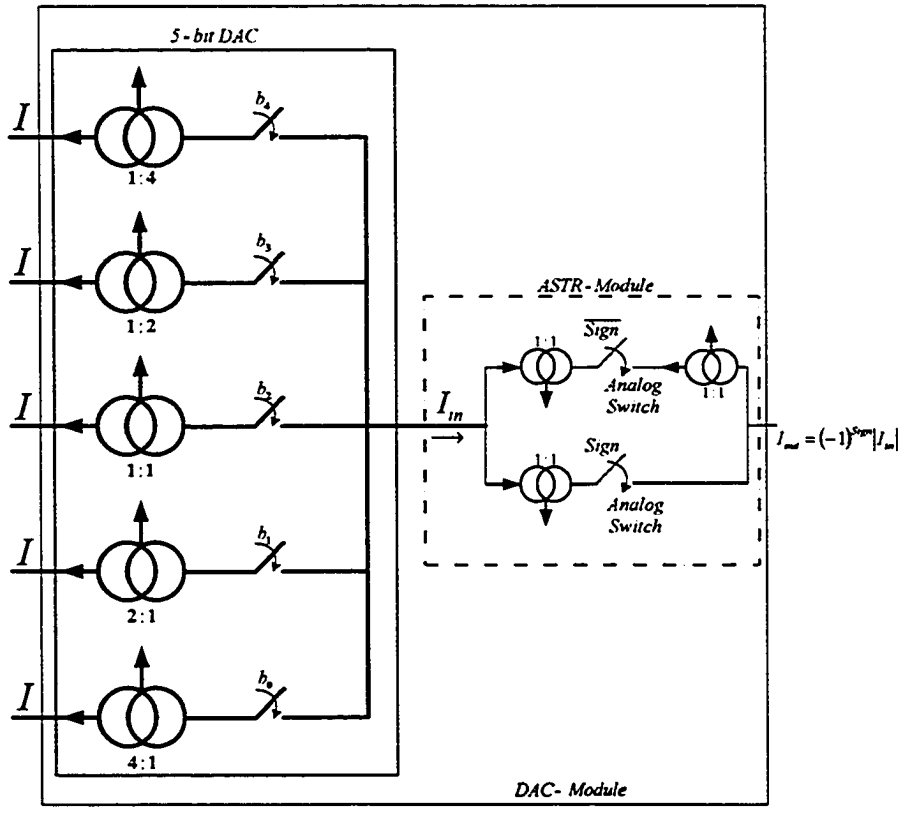


Figure 7-11 Block diagram of a DAC module that consists of a 5-bit current steering DAC and an ASTR module.

In this module, I is a fixed biasing current corresponding to the DAC gain and b_0-b_4 are input bits representing the magnitude. The polarity is represented by the *Sign* bit. A

DAC module consists of a 5-bit current steering DAC and an ASTR module. The output of the 5-bit DAC is always positive and is applied to the *ASTR* module to produce an output current with the same magnitude and with the desired polarity. Figure 7-12 shows the simulated transfer response of the DAC module in the chip. This figure shows output currents corresponding to the input digital values. It can be seen that output currents can change from -2.8μA to 2.8μA and average analog current step size is about 90nA. However, each output current level is not precisely 1 LSB apart from its adjacent levels. This deviation of analog step sizes from 1 LSB is called "*differential nonlinearity (DNL) error*" [70]. Figure 7-13 shows DNL error for this DAC.

Figure 7-13 shows that maximum deviation of analog step sizes from 1 LSB is about 0.4 LSB, which shows that the quality of this DAC circuit is not very good and does not linearly map received information into currents. Also, Figure 7-14 shows the transfer response of the utilized DAC module, when the resolution of which is reduced to 5 bits. Figure 7-15 shows that maximum deviation of analog step sizes in this case is about 0.1 LSB. Figure 7-16 shows simulation results based on SR-MS for an analog decoder with different number of quantization bits at the input. Also, analog levels and clipping effects are chosen similar to the decoder chip shown in Figure 7-12 and Figure 4-38. However, it does not show any significant difference between 5-bit and 6-bit quantization for the input signal. Degradation in the error correcting performance starts when the number of quantization bits is less than 5. Thus, the nonlinearity in DAC module can not directly justify better measured performance for 5-bit quantization.

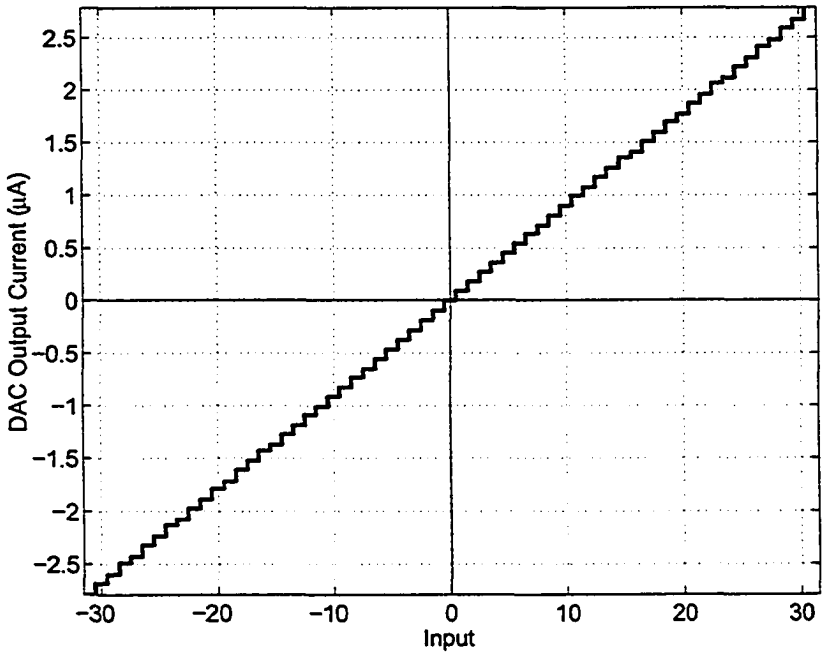


Figure 7-12 Transfer response of the DAC module with 6 bits resolution.

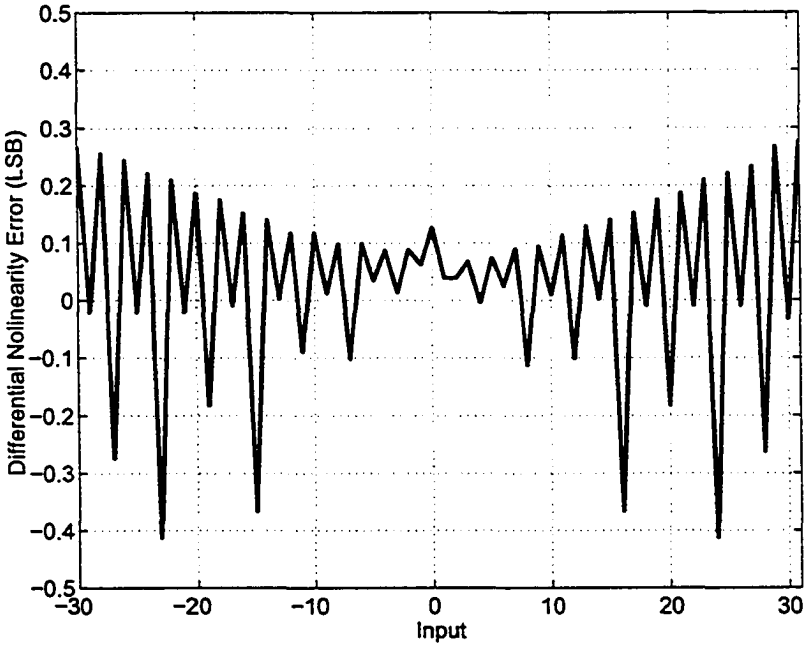


Figure 7-13 DNL error for the DAC module with 6 bits resolution.

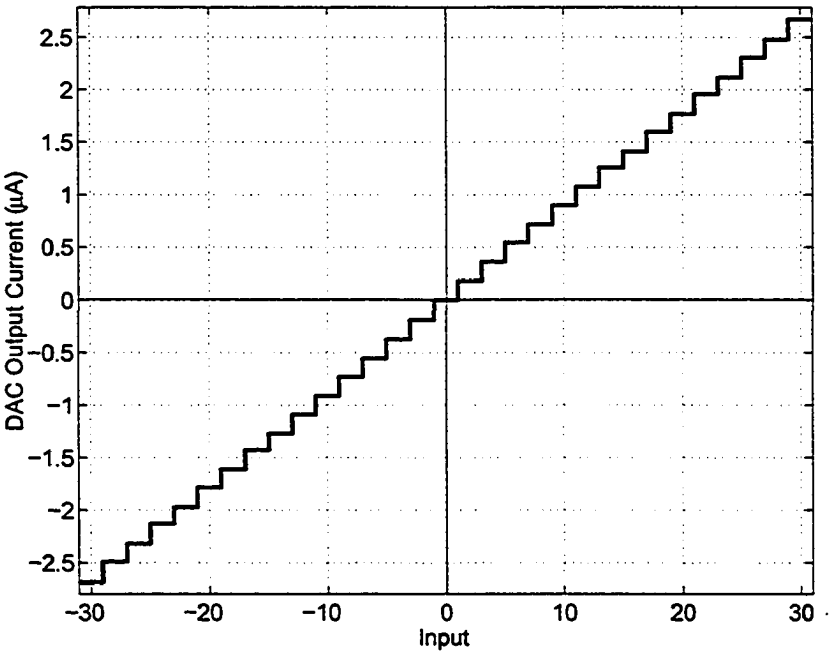


Figure 7-14 Transfer response of the DAC module with 5 bits resolution.

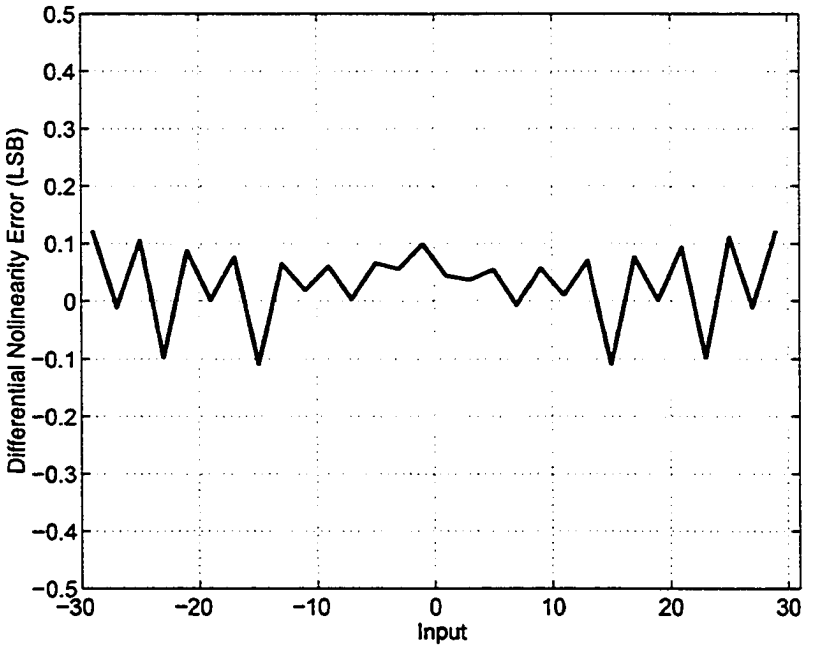


Figure 7-15 DNL error for the DAC module with 5 bits resolution.

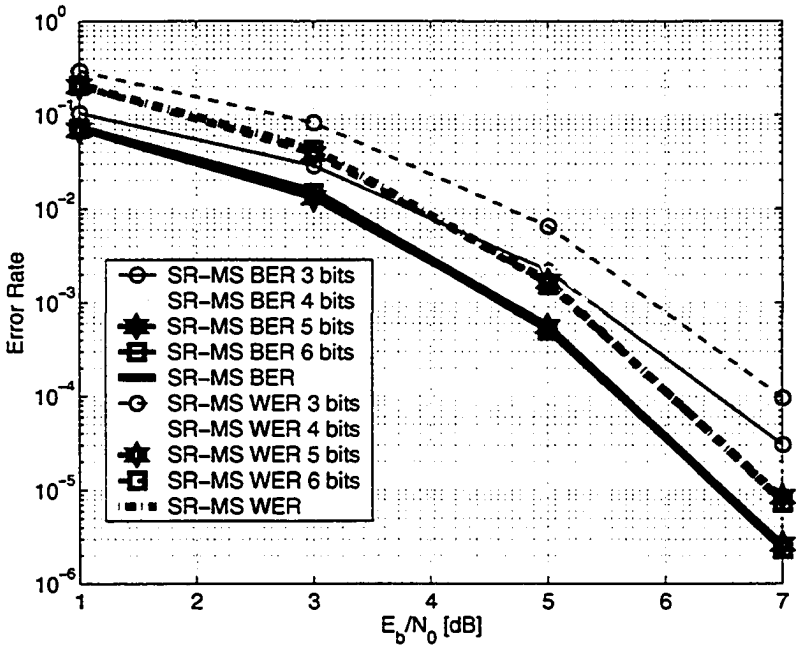


Figure 7-16 Error correcting performance of SR-MS for different number of quantization level for inputs (analog levels and clipping are similar to those of the chip) along with error correcting performance of SR-MS without any clipping and quantization.

However, it is seen that analog step size for 6-bit quantization is very small. This implies that currents can be very small. This makes the circuit susceptible to other imperfections as we will see in the next sections.

It is worth noting that in our simulations for quantized SR-MS and also in the measurements, received information from the channel has been initially clipped and is confined in $[-(1+4\sigma), (1+4\sigma)]$ interval, where σ is the standard deviation of the noise. This interval has been mapped to the available dynamic range of the DAC modules. As Figure 7-16 shows, based on this clipping, the performance of ideal SR-MS, SR-MS with clipping and 6-bit quantization, and SR-MS with clipping and 5-bit quantization are very close and are overlapping. As quantization error in this approach has not degraded the

error correcting performance compared to ideal SR-MS, we have used this clipping without trying to optimize it. However, for digital implementation of SS-MS decoding algorithm it has been shown that if clipping interval and the number of quantization bits are optimized, it is possible to improve the error correcting performance of the quantized and clipped SS-MS and it is likely that the same behavior can be observed in SR-MS as well [41].

7.3.3 Imperfections in MS Decoder Chip: Mismatch

In Chapter 4 we briefly discussed the mismatch problem and mentioned that mismatch is a fabrication imperfection that causes time-independent random variations in physical attributes of identically designed devices. It is widely recognized that mismatch is key to accuracy in analog circuits in general [101], [102] and it appears to be the main factor that limits the precision of the computational modules in analog decoders in particular [15], [62], [63]. Mismatch effects are often divided into local and global variations. Global variation in an integrated circuit is independent of transistor size, however, for local mismatch, the variation of the length (L) depends on the width (W) of the device ($\sigma_L^2 \propto 1/W$) and likewise the variation of W depends on the length of the device ($\sigma_W^2 \propto 1/L$). Parameters such as sheet resistance, doping concentration in channel, carrier mobility, and gate oxide thickness depend on the gate area ($\sigma^2 \propto 1/LW$) [102]. Effect of mismatch on each electrical parameter, such as drain current, is studied by incorporating mismatch in the related physical parameters. In [102] effects of mismatch on the performance of current mirrors, which are the basic building blocks in our MS analog decoder, were studied based on a comprehensive study of different fabrication technolo-

gies. In that work, authors conclude current mirror mismatch depends strongly on L and not on W. It is also shown that for 0.13μm CMOS technology and 0.25μm BiCMOS technology current mismatch between reference current (10μA) and output could be less than 1% only for large transistors. For minimum size transistors mismatch is about 24% in the 0.25μm BiCMOS technology and about 16% percent in the 0.13μm CMOS technology. For these fabrication technologies when transistor sizes are five times greater than the minimum size, current mismatch is about 5%-10%. Since carrier mobility in p-type transistors is less than half of that in n-type transistors, mismatch effect is smaller for p-type transistors. Also, it is shown that mismatch increases when the reference current is reduced. As an example in a current mirror that uses 2μm/2μm n-type MOS transistors in a 0.13μm CMOS technology, when the reference current is reduced from 10μA to 1μA, mismatch doubles and becomes 9%.

In the fabricated MS analog decoder in 0.18μm CMOS technology, we mainly used 2μm/0.4μm p-type and 1μm/1μm n-type MOS transistors. Also, drain currents are always smaller than 3.56μA and currents greater than this value are clipped in WTA modules. On the other hand, currents could be a fraction of 1μA. Therefore, if we consider effect of technology scaling and reduction in the currents and transistor sizes compared to what has been reported in [102] for a 0.13μm CMOS technology, it would be fair to assume mismatch in the processing modules in the fabricated MS decoder chip is between 10%-20% ($0.1<\sigma<0.2$) .

To verify the effect of mismatch on the error correcting performance of the analog decoder chip, we include mismatch in our simple model for analog decoding and multiply each outgoing message by a normal random variable (non-negative) with a mean

equal to 1 and a standard deviation equal to σ, which can be 0.1 or 0.2 that correspond to the estimated mismatch in the analog decoder chip. One hundred different cases have been simulated and corresponding BER and WER curves when σ =0.1 are presented in Figure 7-17 and simulation results for σ=0.2 are presented in Figure 7-18. Along with mismatch curves, we present measured error rate curves for the chip and simulation results when there is no mismatch (SR-MS-5b). These curves show that in high signal to noise ratios, mismatch always degrades the error correcting performance; however, Figure 7-17 shows that there are cases that mismatch can slightly improve error correcting performance in low signal to noise ratios. By increasing the standard deviation, average degradation in the error correcting performance is increased. A comparison between the measured curves with the simulated curves with mismatch implies that predicted degradation due to mismatch is consistent with the measurement results.

Similarly, in [62] a realistic standard deviation for mismatch in short transistors is estimated to be about 10% and simulation results based on successive substitution for a large number of BP decoding networks are presented. It is observed that mismatch can sometimes improve the error correcting performance in low and high signal to noise ratios for discrete-time decoders that use successive substitution method for updating the messages.

No significant change in the error rate curves is observed when number of quantization bits at the input is increased from 5 bits to 6 bits as shown in Figure 7-19 and Figure 7-20. This shows that we can not justify the discrepancy based on our simple model. Nevertheless, it should be noted that the assumption of constant mismatch throughout the simulation, which is not precise, could have brought about this misleading

result and by using a more complex current-dependent model for mismatch we might be able quantitatively justify what happens when LSB becomes small in the analog MS decoder.

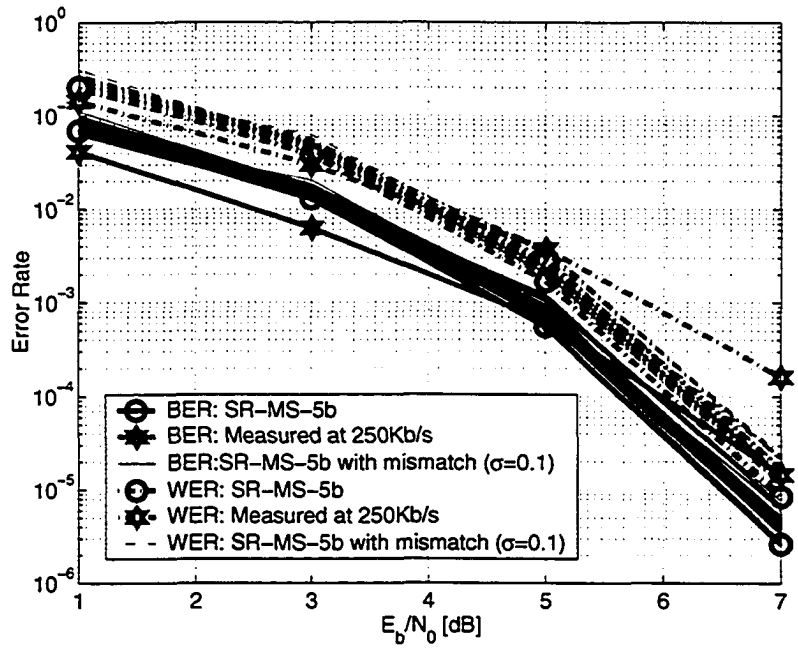


Figure 7-17 Effect of mismatch (σ=0.1) on the analog decoder. SR-MS-5b: Analog MS decoder with clipping and using 5-bit quantization for the inputs.

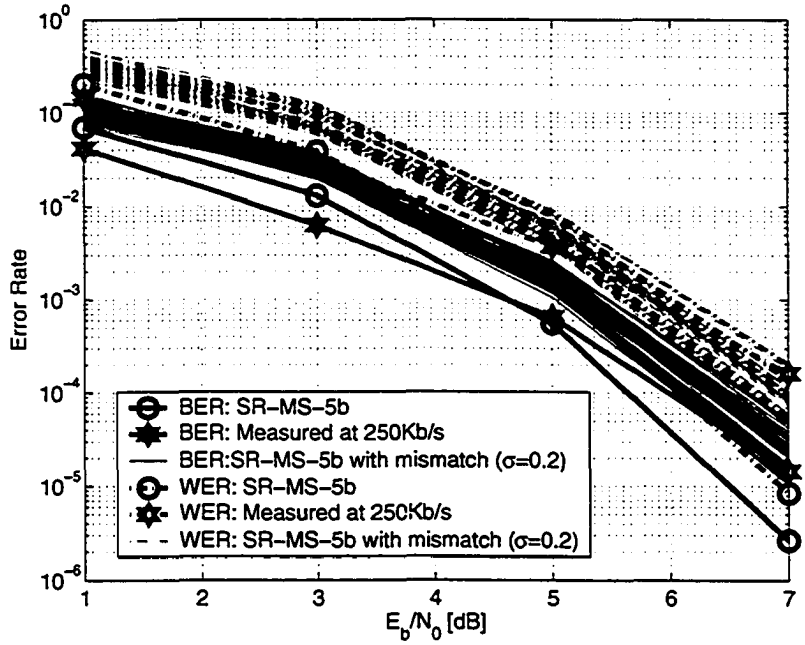


Figure 7-18 Effect of mismatch (σ=0.2) on the analog decoder. SR-MS-5b: Analog MS with clipping and using 5-bit quantization for the inputs.

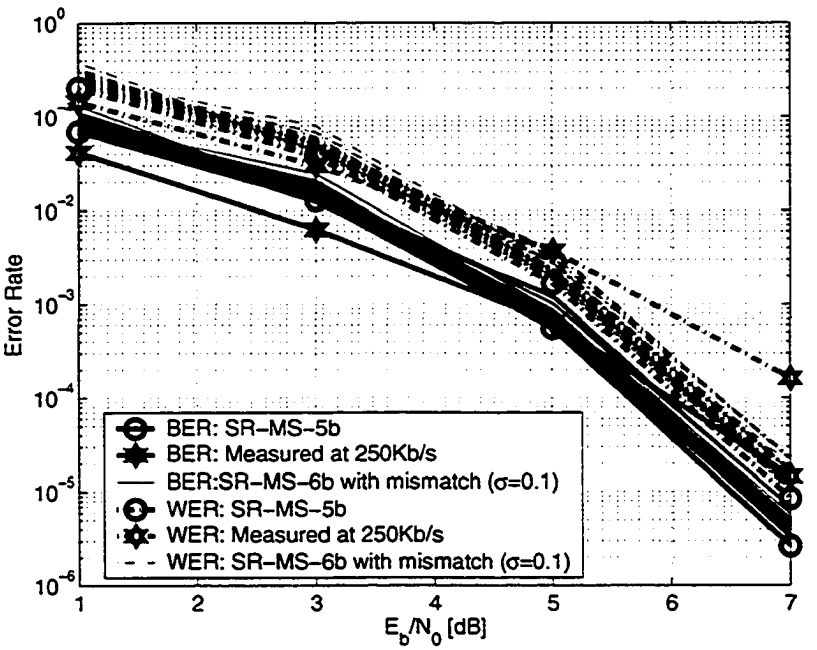


Figure 7-19 Effect of mismatch (σ=0.1) on the analog decoder. SR-MS-6b: Analog MS decoder with clipping and using 6-bit quantization for the inputs.

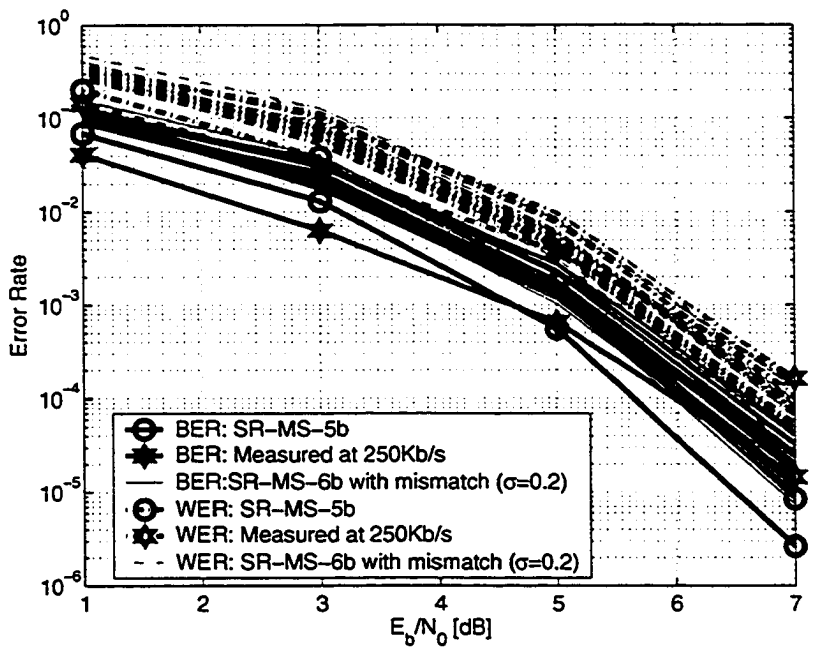
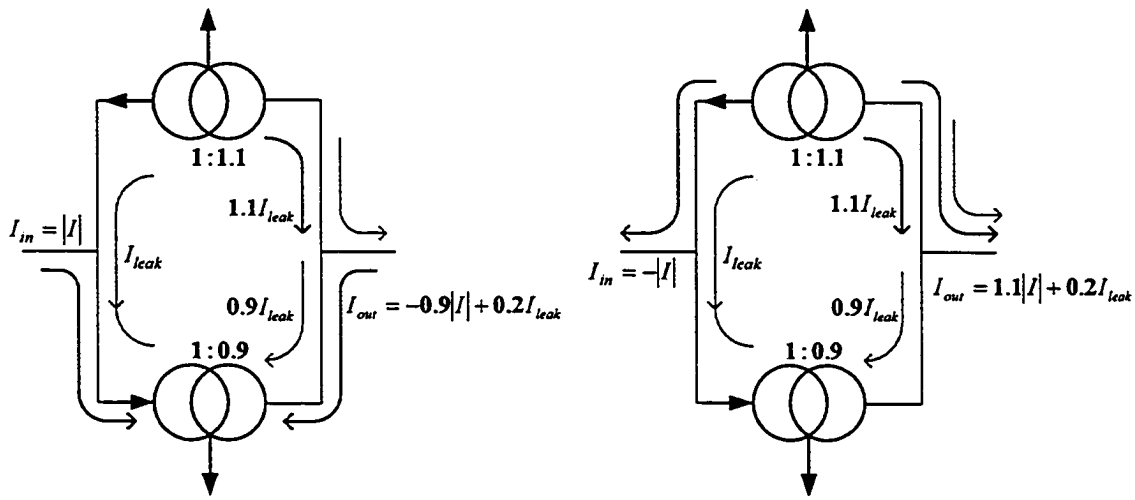


Figure 7-20 Effect of mismatch (σ=0.2) on the analog decoder. SR-MS-6b:Analog MS decoder with clipping and using 6-bit quantization for the inputs.

7.3.4 Imperfections in MS Decoder Chip: Leakage Current

In this section we qualitatively study effect of leakage current on current buffers and sign extractors and show that for small input currents the sign of the message carrying currents can be affected.

In the previous section we assumed that mismatch can only change the magnitude of a message carrying current, however, mismatch effect in a current buffer can be more severe because of leakage current that flows in the loop for small input currents and at the output produces an input dependent offset current. Mismatch can unbalance leakage currents at the input and the output parts of a current buffer and ultimately degrades the accuracy when input current is small as shown in Figure 7-21.



(a) (b)

Figure 7-21 Unbalanced leakage currents caused by mismatch degrades the accuracy of the current buffers: (a) input current is positive; offset can change the sign (b) input current is negative; offset can not change the sign.

When the input current is large, the voltage drop on the active current mirror will be large and can completely shut down the other current mirror and therefore leakage current becomes negligible. Therefore, this phenomenon can be dominant only for small input currents.

Similarly for the sign extractors, if the input current is smaller than the leakage in the input capacitor, as illustrated in Figure 7-22, the sign of the input current can be wrongly detected. Both degrading effects due to leakage current can be mitigated if input currents are large enough. This can also partly justify why increasing the quantization bits from 5 bits to 6 bits at the inputs that associates with a reduction in the size of the LSB degrades the performance.

(a) (b)

Figure 7-22 Degrading effect of leakage current in input capacitor of the sign extractor: (a) input current is positive; leakage current can change the sign (b) input current is negative; leakage current does not change the sign.

7.3.5 Imperfections in MS Decoder Chip: Noise and Nonlinearities

The minimum signal level that an analog circuit can process with acceptable quality is often limited by noise [9]. Inherent noise and interference noise are the main sources of noise in a circuit. Inherent noise refers to random signals that devices generate

naturally and can not be eliminated but can be reduced through proper circuit design. Interference noise refers to unwanted signals induced by outside world or other parts of the circuit and can be significantly reduced by careful circuit wiring and layout [70].

The dominant inherent noise sources in the fabricated analog decoder chip are flicker and thermal noise in MOS transistors and thermal noise in distributed resistors throughout the chip. The power spectral density of flicker noise is inversely proportional to the frequency and the transistor size but is not a function of biasing condition. In CMOS analog decoders flicker noise can be important because transistors are often small and the frequency of operation is not that high. The power spectral density of thermal noise is practically white and it is a function of resistance in the distributed resistors and resistance in the channel in MOS transistors [9], [70].

Switching noise in the logic gates in the fabricated chip and in the test board can be the main sources of interference noise in analog circuits in the fabricated MS decoder. Despite using guard rings around processing modules and utilizing separate supply and ground bond pads for biasing the substrate, logic gates, and analog circuits in the MS decoder, the degrading effects of switching noise will be a factor due to substrate coupling [9], and degradation in the error correcting performance of the decoder chip can be partly because of this noise.

An accurate and comprehensive survey on the effects of noise on the performance of the analog decoder can be very complicated, mainly because of the complexity of the modules, size of this analog circuit, and difficulties in modeling the effect of substrate coupling. To simplify the case, we only study the effect of additive white Gaussian noise (AWGN) on the performance of the analog decoder. We consider an AWGN noise source

for each outgoing edge and add its output to the outgoing message. We assume noise has zero mean and study its effect for different standard deviations (σ). We use our simple model for SR-MS decoding with 5-bit quantization at the input and clipping similar to the chip. Figure 7-23 shows simulated BER curves when independent AWGN noise sources with five different standard deviations are considered. In these simulations, standard deviations are comparable with 1 LSB (0.18μA) and it seems unreasonable to assume higher noise power for each outgoing edge. This figure shows that AWGN noise has insignificant effect on the performance of this analog decoder. The same observation is made for the WER curves.

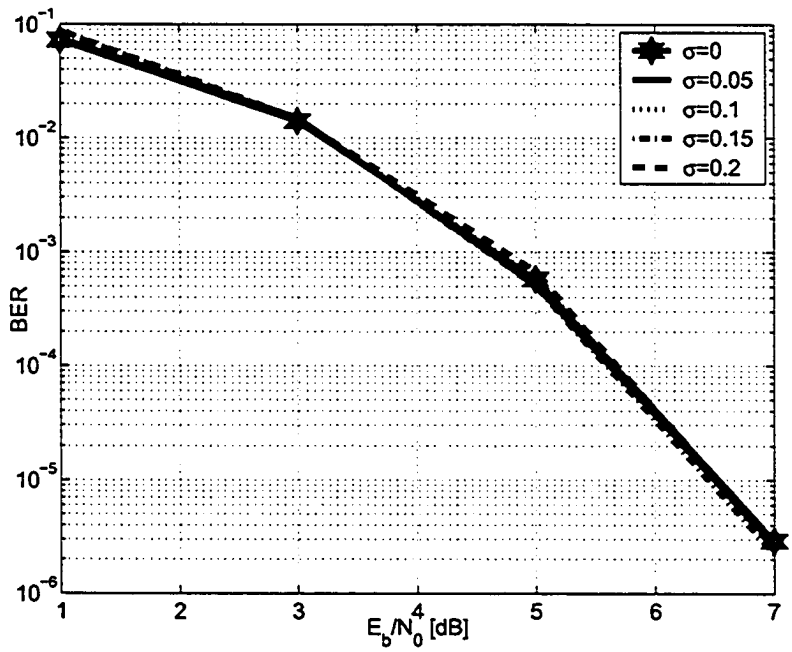


Figure 7-23 Effect of AWGN noise on the BER performance of the SR-MS-5b.

Another factor that seems important in justifying the difference between the error correcting performance of the ideal model and the fabricated chip is nonlinearity in the processing modules. A realistic model for nonlinearity can be very hard to derive and implement as nonlinearities are very complex to formulate and are a function of current and

speed and a detailed study requires accurate statistical transistor level simulations, which is not affordable. To simplify the analysis we model the effect of nonlinearities in each processing module by a random variable that is added to each outgoing message and has a Gaussian distribution. These additive noise sources add some error to each outgoing message to mimic the nonlinearities. As this scenario is similar to what we used for studying the effects of noise, we again refer to Figure 7-23 to show minor nonlinearities have insignificant effect on the overall decoding performance.

7.3.6 Imperfections in MS Decoder Chip: Stopping Criterion

In Chapter 2, it was mentioned that in iterative decoding algorithms, the decoding process is stopped, as soon as the decoder converges to a codeword, that is, when hard-decision assignment for the outputs of the variable nodes satisfies all the parity-check equations. If the decoder fails to converge to a codeword, the decoding process would continue up to a given number of iterations. However, in the fabricated analog decoder chip, decoding time for each block of data is fixed based on the desired throughput and output of each variable node is sampled only at the end of the decoding time interval. In other words, if the decoder converges to a codeword but later abandons that codeword, in our analog decoder chip it is considered as a non-convergence, but in conventional simulations it is reported as a converged case. Requiring stability in the convergence tightens the definition of a successful decoding and can reduce the number of converged cases and ultimately degrades the error correcting performance of our analog decoder chip. Figure 7-24 shows the difference between error correcting performance of SR-MS decoding for the (32,8) LDPC code when iteration number is always 50k (similar to the chip with fixed decoding time) and when maximum number of iteration is 50k (similar to our previous

simulations throughout this thesis). 6-bit quantization at the input and clipping effects similar to the chip have been assumed in the simulations. For each signal to noise ratio at least 200 wrong codewords have been observed at the output of the decoder with the exception of 7dB for the case with fixed number of iterations. For this simulation point only 25 wrong codewords have been observed after running the simulations for a long period of time. This is because the average number of iterations at this signal to noise ratio is about 1.38 and fixing the iteration number at 50,000 significantly increases the simulation time. This figure shows that a part of the degradation in the decoding performance of the MS decoder chip at high signal to noise ratios can be because of the different stopping criterion that has been implemented.

It is possible to include specific continuous-time digital circuits, mainly consisting of asynchronous XOR gates, to verify if parity-check equations are satisfied in the continuous-time domain and hold outputs and stop the decoding as soon as the parity-check equations are satisfied. In this way, the stability requirement for the convergence is relaxed. This will increase switching noise, hence other means must be found to curb this extra switching noise.

7.4 Conclusion

To prove the functionality of the proposed circuits an analog min-sum iterative decoder chip for a (32,8) regular LDPC code was designed and fabricated in 0.18μm CMOS technology. Measurement results not only show that the circuit is functional but also confirm the validity of our proposed model for ideal analog decoding.

Also, we studied the reasons of discrepancy between simulation results and measurement results. Imperfection due to simulation, DAC modules, mismatch, leakage cur-

rents, nonlinearities, noise, and different stopping criterion, were studied. Mismatch was found to be the most important contributing factor in our chip.

By using larger transistors, it is possible to mitigate degrading effect of mismatch. Also by increasing the dynamic range and using larger currents for representing messages, it is possible to boost signal strength and protect it from noise and minimize degrading effect of leakage currents in the chip and improve error decoding performance of analog MS decoder chips.

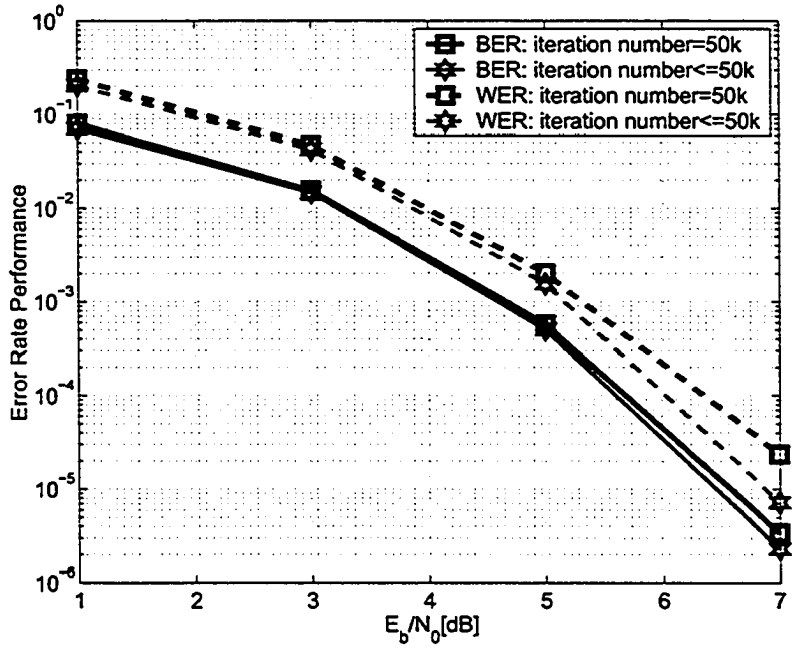


Figure 7-24 Error correcting performance of SR-MS decoding (with 6-bit resolution and clipping similar to the chip) when number of iterations is always 50k and when decoding is stopped as soon as it converges to a codeword (iteration number $\leq$50k).

Chapter 8

Concluding Remarks

In this chapter, we summarize the main contributions and results of this dissertation and present some ideas for further research work on analog iterative decoding.

8.1 Summary of Results

In this dissertation, we studied theoretical and practical aspects of analog iterative decoding. In the theoretical part, we studied the difference between the dynamics of conventional discrete-time and continuous-time iterative decoding. We presented a simple model for continuous-time decoding and showed that it is equivalent to the application of successive-relaxation method for solving the complex problem of decoding. In this regards, not only have we shown that analog decoding enjoys better dynamics, but we have also come up with a new iterative decoding algorithm which surpasses the performance of the best known iterative decoding algorithms. We also recognized quite a few methods that have been proposed recently for improving the performance of iterative decoding as the application of better numerical techniques to the fixed-point problem of iterative de-

coding. In this way, we have gathered seemingly different and diverse methods under the same umbrella.

In practical aspects, we focused on min-sum decoding algorithm and devised the first strongly inverted CMOS analog iterative decoder, which is more affordable, compared to the previously reported BiCMOS analog iterative decoders and are faster than analog decoders that use weakly inverted CMOS transistors. Our very basic building blocks are a p-type and an n-type current mirror and all the different modules in our design can be implemented based on these very simple blocks. This is very important because it makes the design modular and significantly simplifies the design process. This approach can then be used wherever accurate current mirrors can be implemented. This includes advanced CMOS technologies with very small transistors and reduced supply voltages or even BiCMOS technology for very high-speed applications. We also considered the case when nodes in a min-sum decoder have high degrees and designed special processing modules that significantly reduce the power and silicon area consumption. Also, a general approach was presented for converting virtually any current mirror into a current-mode maximum WTA circuit and as an example a novel low-voltage current-mode max WTA was designed. A proof-of-concept min-sum analog decoder chip was designed and fabricated in 0.18μm CMOS technology for a (32,8) LDPC code, which is currently the fastest CMOS analog iterative decoder and it is the first analog min-sum iterative decoder chip and also, it is the first analog decoder for an LDPC code. Measured error rate performance of this chip is close to simulation results based on our proposed model for an ideal analog decoder in low signal to noise ratio, where the channel noise dominates the imperfections in the chip. Degrading effects of imperfections are also stud-

ied for this chip. Mismatch and limited dynamic range were found responsible for degradation in the performance of analog iterative decoder in high signal to noise ratios.

Also, in another direction, we showed that wherever we can implement *pq* splitters (that split incoming signal proportional to $\frac{p}{p+q}$ and $\frac{q}{p+q}$), we can implement an analog iterative decoder based on the belief propagation decoding algorithm and having a Gilbert multiplier is not necessary. This observation can pave the way for implementing very high-speed electro-optical analog iterative decoders.

8.2 Ideas for Further Future Work

We conclude this dissertation by briefly describing a few open questions for future research work on analog decoding.

- In this thesis we expressed the dynamics of a continuous-time iterative decoding as a nonlinear differential equation. It would be very interesting if common theoretical tools in studying nonlinear systems could be applied to this problem for analyzing its characteristics. These could include sufficient conditions for convergence and stability of this problem. Also, as this system of equations would change according to the domain in which we represent the messages (i.e. LLR or LR domain), the overall performance of the decoding algorithms may change as well and it would be of great interest if for each decoding algorithm the best message representation could be found.
- While the functionality of our proposed method for implementing min-sum iterative decoder was verified in practice, the used code is too small to have any practical application. However, even for this small code, the analog chip is too

large for conventional analog layout design. It is therefore, necessary to improve automation tools to facilitate the design process for longer and more practical codes.

- As the fabricated chip was the first analog min-sum decoder, we made many assumptions that can be changed in future work. Assuming single ended message passing and using current-mode max WTA circuits are two examples of these assumptions. It would be interesting if differential message passing could be studied. In fact, it is possible to slightly modify the proposed modules to work with differential messages. Also, it is possible to design voltage mode min WTA circuits and it is interesting to test how this could change the area and power consumption in the check nodes.
- In this thesis, we focused on LDPC codes and Tanner graph representation of the block codes for performing the iterative decoding. Our motivation was to use the parallel structure of the decoder to achieve higher throughputs. However, the proposed modules can be used for decoding turbo codes. Application of the proposed modules to turbo decoders could be studied in another research work.

References

[1] J. Bardeen and W. H. Brattain, "The transistor, a semiconductor triode," *Physical Review*, vol. 74, no. 2, pp. 230–231, July 1948.

[2] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Technical Journal*, vol. 27, pp. 379-423, July 1948.

[3] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21-28, Jan. 1962.

[4] D. J. C. MacKay and R. M. Neal, "Near Shannon-limit performance of low-density parity-check codes," *Electronics Letters*, vol. 32, no.18, pp. 1645-1646, August 1996.

[5] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications Letters*, vol. 5, no.2, pp. 58-60, Feb. 2001.

[6] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding; turbo codes," in *Proc. IEEE Int. Conf. Communications*, pp. 1064-1070, Geneva, May 1993.

[7] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, pp. 114-117, April 1965.

[8] M. G. Rekoff, Jr., *Analog Computer Programming*, C. E. Merrill Books, 1967.

[9] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw-Hill, 2001.

[10] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits, A Design Perspective*. Second Edition, Prentice Hall, 2003.

[11] C. A. Mead, *Analog VLSI and Neural Systems*, Addison Wesley, Reading, MA, 1989.

[12] C. A. Mead, "Silicon models of neural computation," in *Proc. IEEE first Int. Conf. on Neural Networks*, vol. 1, pp. 91-106, June 1987.

[13] E. Fiesler and R. Beale, *Handbook of Neural Computation*, Oxford University Press, New York, 1997.

[14] H. T. Siegelmann, *Neural Networks and Analog Computation: Beyond the Turing Limit*, Birkhäuser , Boston, 1999.

[15] F. Lustenberger, On *the Design of Analog VLSI Iterative Decoders*, Ph.D. dissertation, ETH, Zurich, Switzerland, Nov. 2000.

[16] A. Carusone, K. Farzan, and D. A. Johns, "Differential signaling with a reduced number of signal paths," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, pp. 294-300, March 2001.

[17] J. B. Berner and K. S. Andrews, "Deep space network turbo decoder implementation," in *Proc. IEEE Aerospace Conference*, pp. 1149-1157, 2001.

[18] A. J. Blanksby and C. J. Howland, "A 690-mW 1Gb/s 1024-b rate-1/2 low-density parity-check code decoder," *IEEE J. Solid-States Circuits*, vol. 37, no. 3, pp. 404-412, March 2002.

[19] J. Hagenauer, E. Offer, C. Méasson, and M. Mörz, "Decoding and equalization with analog non-linear networks," *European Trans. Telecommun. (ETT)*, vol. 10, pp. 659-680, Nov.-Dec. 1999.

[20] J. Hagenauer, M. Moerz, and E. Offer, "Analog turbo-networks in VLSI: The next step in turbo decoding and equalization," in *Proc. Int. Symp. Turbo Codes*, ENST, Bretagne, France, pp. 209-218, 2000.

[21] M. Moerz, T. Gabara, R. Yan, and J. Hagenauer, "An analog 0.25μm BiCMOS tailbiting MAP decoder," in *Proc. IEEE Solid-States Circuits Conf. (ISSCC)*, pp. 356-357, San Francisco, USA, Feb. 2000.

[22] F. Lustenberger, M. Helfenstein, H.-A. Loeliger, F. Tarköy, and G. S. Moschytz , "All-analog decoder for a binary (18,9,5) tail-biting trellis code", in *Proc. European Solid-State Circuit Conf. (ESSCIRC)*, pp. 362-365, Duisburg, Germany, Sept. 1999.

[23] H.-A. Loeliger, F. Lustenberger, M. Helfenstein, and F. Tarköy, "Probability propagation and decoding in analog VLSI," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 837-843, Feb. 2001.

[24] M. Arzel, C. Lahuec, M. Jézéquel, and F. Seguin, "Analogue decoding of duo-binary codes," in *Proc. Int. Symp. Inform. Theory and its Applications*, Parma, Italy, pp. 10-13, Oct. 2004.

[25] A. F. Mondragón-Torres, E. Sánchez-Sinencio, and K. R. Narayanan, "Floating-gate analog implementation of the additive soft-input soft-output decoding algorithm," *IEEE Trans. Circuits and Systems I*, vol. 50, no. 10, pp. 1256-1269, Oct. 2003.

[26] V. C. Gaudet and P. G. Gulak, "A 13.3-Mb/s 0.35μm CMOS analog turbo decoder IC with a configurable interleaver," *IEEE J. Solid-States Circuits*, vol. 38, no. 11, pp. 2010-2015, Nov. 2003.

[27] C. Winstead, J. Dai, S. Yu, C. Myers, R. R. Harrison, and C. Schlegel, "CMOS analog MAP decoder for (8,4) Hamming code," *IEEE J. Solid-States Circuits*, vol. 39, no. 1, pp. 122-131, Jan. 2004.

[28] A. G. Amat, S. Benedetto, G. Montorsi, D. Vogrig, A. Neviani, and A. Gerosa, "An analog turbo decoder for the UMTS standard," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 296, 2004.

[29] D. Vogrig, A. Gerosa, A. Neviani, A. G. Amat, G. Montorosi, and S. Benedeto, "A 0.35-μm CMOS analog turbo decoder for the 40-bit Rate 1/3 UMTS channel code," *IEEE J. Solid-States Circuits*, vol. 40, no. 3, pp. 753-762, March 2005.

[30] S. Hemati and A. H. Banihashemi, "Full CMOS min-sum analog iterative decoder," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 347, 2003.

[31] S. Hemati and A. H. Banihashemi, "Iterative decoding in analog CMOS," in *Proc. 13th ACM Great Lakes Symposium on VLSI (ACM GLSVLSI)*, Washington D.C., USA, pp. 15-20, 2003.

[32] S. Hemati, A. H. Banihashemi, and C. Plett, "A high-speed analog min-sum iterative decoder," to appear in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, Sept. 2005.

[33] S. Hemati, A. H. Banihashemi, and C. Plett, "An 80-Mb/s 0.18-μm CMOS analog min-sum iterative decoder for a (32,8,10) LDPC code," to appear in *Proc. IEEE Custom Integrated Circuit Conf. (CICC)*, Sept. 2005.

[34] S. Lin and D. J. Costello, Jr., *Error Control Coding*, Prentice Hall; Second Edition, 2004.

[35] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533-547, Sept. 1981.

[36] A. R. Calderbank, G. D. Forney, and A. Vardy, "Minimal tail-biting trellises: The Golay code and more," *IEEE Trans. Inform. Theory*, vol. 45, no.5, pp. 1435-1455, July 1999.

[37] E. Berlekamp, R. McEliece, and H. van Tilborg, "On the inherent intractability of certain coding problems," *IEEE Trans. Inform. Theory*, vol. 24, no.3, pp. 384-386, May 1978.

[38] J. Pearl, *Probabilistic Reasoning in Intelligence Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., 1988.

[39] N. Wiberg, *Codes and Decoding on General Graphs*. Ph.D. dissertation, Linköping University, Sweden, 1996, available at: http://www.it.isy.liu.se/publikationer/LIU-TEK-THESIS-440.pdf.

[40] M. R. Yazdani, S. Hemati, and A. H. Banihashemi, "Improving belief propagation on graphs with cycles," *IEEE Comm. Lett.*, vol. 8, no.1, pp. 57-59, Jan. 2004.

[41] J. Zhao, F. Zarkeshvari, and A. H. Banihashemi, "On implementation of min-sum algorithm and its modifications for decoding LDPC codes," *IEEE Trans. Comm.*, vol. 53, no. 4, pp. 549-554, April 2005.

[42] J. Chen and M. P. C. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity check codes," *IEEE Trans. Comm.*, vol. 50, no. 3, pp. 406-414, March 2002 .

[43] J. Chen and M. P. C. Fossorier, "Density evolution for two improved BP-based decoding algorithms of LDPC codes," *IEEE Comm. Lett.*, vol. 6, no. 5, pp. 208-210, May 2002.

[44] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, SIAM Press, Philadelphia, USA, 1995.

[45] D. M. Young and R. T. Gregory, *A Survey of Numerical Mathematics*, Addison-Wesley Series in Mathematics, New York, 1973.

[46] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[47] T. Richardson, "The geometry of turbo-decoding dynamics," *IEEE Trans. Inform. Theory*, vol. 46, no. 1, pp. 9-23, January 2000.

[48] P. Moqvist and T. M. Aulin, "Turbo-decoding as a numerical analysis problem," in *Proc. IEEE Int. Symp. Inform. Theory*, p. 485, 2000.

[49] J. D. Lambert, *Numerical Methods for Ordinary Differential Systems: the Initial Value Problem*, John Wiley & Sons, Inc., 1973.

[50] G. Dahlquist, "A special stability problem for linear multistep methods," *J. BIT Numerical Mathematics*, vol. 3, pp. 27- 43, 1963.

[51] S. Hemati and A. H. Banihashemi, "Full CMOS min-sum analog iterative decoder," US Patent (Pending), Application No. 10/832,806, April 27, 2004.

[52] Y. Mao and A. H. Banihashemi, "A heuristic search for good low-density parity-check codes at short block lengths," in *Proc. IEEE Int. Conf. on Comm. (ICC)*, pp. 41-44, 2001.

[53] D. J. MacKay, "Encyclopedia of sparse graph codes," available at *http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.*

[54] M. P. C. Fossorier, "Iterative reliability-based decoding of low-density parity-check code," *IEEE JSAC*, vol. 19, no. 5, pp. 908-917, May 2001.

[55] M. P. C. Fossorier, Department of Electrical Engineering, University of Hawaii at Manoa, Honolulu, HI, private communication, May 2003.

[56] J. Dai, *Design Methodology for Analog VLSI Implementations of Error Control Decoders*, Ph.D. Dissertation, University of Utah, December, 2002.

[57] C. Winstead and C. Schlegel, "Importance sampling for SPICE-level verification of analog decoders," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p.103, 2003.

[58] M. A. Bickerstaff, D. Garrett, T. Prokop, C. Thomas, B. Widdup, G. Zhou, L. M. Davis, G. Woodward, C. Nicol, and R.-H. Yan, "A Unified Turbo/Viterbi channel decoder for 3GPP mobile wireless in 0.18-μm CMOS," *IEEE J. Solid-States Circuits*, vol. 37, pp. 1555-1564, Nov. 2002.

[59] B. Gilbert, "A precise four-quadrant multiplier with subnanosecond response," *IEEE J. Solid-States Circuits*, vol. 3, pp. 365-373, Dec. 1968.

[60] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. Inform. Theory*, vol.47, no. 2, pp. 498-519, Feb. 2001.

[61] J. Dai, C. J. Winstead, C. J. Myers, R. R. Harrison, and C. Schlegel, "Cell library for automatic synthesis of analog error control decoders," in *Proc. IEEE Int. Symp. on Circuits and Systems, ISCAS 2002*, vol. 4, pp. 481-484 , 2002.

[62] F. Lustenberger and H.-A. Loeliger, "On mismatch errors in analog-VLSI error correcting decoders," in *Proc. IEEE Int. Symp. on Circuits and Systems, ISCAS 2001*, vol. 4, pp. 198-201 , 2001.

[63] C. Winstead and C. Schelegel, "Density evolution analysis of device mismatch in analog decoder," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*. p. 293, 2004.

[64] M. Helfenstein, F. Lustenberger, H. -A. Loeliger, F. Tarkoy, and G.S. Moschytz, "High-speed interfaces for analog iterative VLSI decoders," in *Proc. IEEE Int. Symp. on Circuits and Systems, ISCAS 1999*, vol. 2, pp. 428-431 , 1999.

[65] M. O' Halloran and R. Sarpeshkar, "A 10-nW 12-bit accurate analog storage cell with 10-aA leakage," *IEEE J. Solid-States Circuits*, vol. 39, no.11, pp. 1985-1996, Nov. 2004.

[66] M. H. Shakiba, D. A. Johns, and K. W. Martin, "An integrated 200-MHz 3.3-V BiCMOS class-IV partial-response analog Viterbi decoder," *IEEE J. Solid-States Circuits*, vol. 33, no.1, pp. 61-75, Jan. 1998.

[67] M. H. Shakiba, D. A. Johns, and K. W. Martin, "BiCMOS circuits for analog Viterbi decoders," *IEEE Trans. Circuits and Systems-II: Analog and Digital Signal*, vol. 45, no.12, pp. 1527-1537, Dec. 1998.

[68] B. Zand and D. A. Johns, "High-speed CMOS analog Viterbi detector for 4-PAM partial-response signaling," *IEEE J. Solid-States Circuits*, vol. 37, no.7, pp. 895-903, July 2002.

[69] A. Demosthenous and J. Taylor, "A 100-Mb/s 2.8-V CMOS current-mode analog Viterbi decoder," *IEEE J. Solid-States Circuits*, vol. 37, no.7, pp.904-910, July 2002.

[70] D. A. Johns and K. Martin, *Analog Integrated Circuit Design*, first edition, John Wiley & Sons, 1997.

[71] P.R. Gray and R.G. Meyer, *Analysis and Design of Analog Integrated Circuits*, third edition, John Wiley & Sons,1993.

[72] J. Hagenauer, "Dar analog decoder," *German Patent Application*, no. 197 25 275.3, filed 14th June, 1997.

[73] J. Hagenauer and M. Winklhofer, "The analog decoder," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 145, 1998.

[74] J. Hagenauer, "Decoding binary codes with analog networks," in *Proc. Int. Workshop on Inform. Theory*, pp. 13-14, 1998.

[75] J. Hagenauer, A. Schaefer, and C. Weiss, "An all-analog ring network for turbo-detection of convolutionally encoded DPSK signals," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 422, 2000.

[76] M. Moerz, J. Hagenauer, and E. Offer, "On the analog implementation of the APP (BCJR) algorithm," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 425, 2000.

[77] M. Moerz, A. Schaefer, E. Offer, and J. Hagenauer, "Analog decoders for high rate convolutional codes," in *Proc. IEEE Workshop on Inform. Theory*, pp. 128-130, 2001.

[78] J. Hagenauer, M. Moerz, and A. Schaefer, "Analog decoders and receivers for high speed applications," in *Proc. Int. Zurich Seminar on Broadband Communications, Access, Transmission, Networking*, vol.3, pp. 1-8, 2002.

[79] A. Schaefer, M. Moerz, J. Hagenauer, A. Sridharan, and D. J. Costello, Jr., "Analog rotating ring decoder for an LDPC convolutional code," in *Proc. IEEE Workshop on Inform. Theory*, pp. 226-229, 2003.

[80] H.-A. Loeliger, F. Lustenberger, M. Helfenstein, and F. Tarkoy, "Probability propagation and decoding in analog VLSI," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 146, 1998.

[81] H.-A. Loeliger, F. Tarkoy, F. Lustenberger, and M. Helfenstein, "Decoding in analog VLSI,"*IEEE Communications Magazine*, vol. 37, no. 4, pp. 99-101, April 1999.

[82] F. Lustenberger, M. Helfenstein, H. -A. Loeliger, F. Tarkoy, and G. S. Moschytz, "An analog VLSI decoding technique for digital codes," in *Proc. IEEE Int. Symp. on Circuits and Systems, ISCAS 2003*, vol. 2, pp. 424-427 , 1999.

[83] F. Lustenburger, M. Helfenstein, H. -A. Loeliger, F. Tarkoy, and G. S. Moschytz, " All-analog decoder for a binary (18,9,5) tail biting trellis code," in *Proc. European Solid-State Circuits Conf.*, pp. 362-365, 1999.

[84] H.-A. Loeliger, "Analog decoding and beyond," in *Proc. IEEE Workshop on Inform. Theory*, pp. 126-127, 2001.

[85] M. Frey, H.-A. Loeliger, F. Lustenberger, P. Merkli, and P. Strebel, "Analog-decoder experiments with subthreshold CMOS soft-gates," in *Proc. IEEE Int. Symp. on Circuits and Systems, ISCAS 2003*, vol. 1, pp. 25-28 , 2003.

[86] A. G. Andreou, K. A. Boahen, P. O. Pouliquen, A. Pavasovic, R. E. Jenkins, and K. Strohbehn, "Current-mode subthreshold MOS circuits for analog VLSI neural systems," *IEEE Trans. on Neural Networks*, vol. 2 , no. 2, pp. 205-213 , March 1991.

[87] T. Serrano-Gotarredona, B. Linares-Barranco, and A. G. Andreou, "A general translinear principle for subthreshold MOS transistors," *IEEE Trans. on Circuits and Circuits- I: Fundamental Theory and Applications*, vol. 46, no. 5, pp. 607- 616, May 1999.

[88] A. Adibi and S. Hemati, "A novel implementation of the Hebbian neurons by Subthreshold MOSFETs in the presence of mobile ions," in *Proc. 7th Iranian Conference on Electrical Engineering ICEE99*, Tehran, Iran, pp. 41-48, 1999.

[89] S. Hemati and A. H. Banihashemi, "Towards Photonic Decoders," in *Technical digest of the Eighth Micro-optics Conf.*, Osaka, Japan, pp. 218-221, 2001.

[90] A. H. Banihashemi and S. Hemati, "Decoding in Optics," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, p. 231, 2002.

[91] V. C. Gaudet, R. J. Gaudet, and P. G. Gulak, "Programmable interleaver design for analog iterative decoders," *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 49, no. 7, July 2002.

[92] J. Lazzaro, S. Ryckebusch, M. A. Mahowald, and C. A. Mead, " Winner-take-all networks of O(n) complexity," in *Advances in neural information processing systems*, volume 2, pp. 703-711, San Mateo - CA, 1989. Morgan Kaufmann.

[93] I. E. Opris, *Analog rank extractors and sorting networks*, Ph.D. dissertation, Stanford Univ., Stanford, CA, 1996.

[94] I. E. Opris, "Analog rank extractors," *IEEE Trans. Circuits Syst. I*, vol. 44, no. 12, pp. 1114-140, Jan. 1998.

[95] R.G. Carvajal , J. Ramirez-Angulo, and J. Tombs, " High-speed high precision voltage-mode Min/Max circuits in CMOS technology," in *Proc. IEEE Int. Symp. Circuits and Systems*, Geneva, Switzerland, pp. 13-16, May 2000.

[96] Z. S. Gunay and E. Sanchez-Sinenico "CMOS winner-take-all circuits: A detail comparison," in *Proc. IEEE Int. Symp. Circuits and Systems*, pp. 41-44, June 1997.

[97] Y. He and E. Sanchez-Sinencio, "Min-net Winner-take-all CMOS implementation," *Electronics Letters*, vol. 29, no. 14, pp. 1237-1239, July 1993.

[98] B. Maundy, "Min/Max circuit for analog conventional decoders," *IEEE Trans. Circuits Syst. II*, vol. 48, no. 8, pp. 802-806, Aug. 2001.

[99] I. Baturone, J. L. Huertas, A. Barriga, and E. Sanchez-Solano, "Current-mode multiple-input Max circuit," *Electronics Letters*, vol. 30, no. 9, pp. 678-679, April 1994.

[100] V. I. Prodanov and M. M. Green "CMOS current mirrors with reduced input and output voltage requirements," *Electronics Letters*, vol. 32, no. 2, pp. 104-105, Jan. 1996.

[101] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching properties of MOS transistors", *IEEE J. Solid-State Circuits*, vol. 24, no. 5, pp. 1433-1440, Oct. 1989.

[102] P. G. Drennan and C. C. McAndrew, "Understanding MOSFET mismatch for analog design", *IEEE J. Solid-State Circuits*, vol. 38, no. 3, pp. 450-456, March 2003.