# Toward the Implementation of Analog LDPC Decoders for Long Codewords

**Shahaboddin Moazzeni**

A Thesis

in The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science (Electrical and Computer Engineering ) at

Concordia University

Montreal, Quebec, Canada

August 2009

*Your file Votre référence*
*ISBN:* 978-0-494-63105-8
*Our file Notre référence*
*ISBN:* 978-0-494-63105-8

# ABSTRACT

**Toward the Implementation of Analog LDPC Decoders for Long Codewords**

Shahaboddin Moazzeni

Error control codes are used in virtually every digital communication system. Traditionally, decoders have been implemented digitally. Analog decoders have been recently shown to have the potential to outperform digital decoders in terms of area and power/speed ratio. Analog designers have attempted to fully understand and exploit this potential for large decoders. However, large codes are generally still implemented with digital circuits. Nevertheless, in this thesis a number of aspects of analog decoder implementation are investigated with the hope of enabling the design of large analog decoders.

In this thesis, we study and modify analog circuits used in a decoding algorithm known as the sum-product algorithm for implementation in a CMOS 90 nm technology. We apply a current-mode approach at the input nodes of these circuits and show through simulations that the power/speed ratio will be improved. Interested in studying the dynamics of decoders, we model an LDPC code in MATLAB's Simulink. We then apply the linearization technique on the modeled LDPC code in order to linearize the decoder about an initial state as its solution point. Challenges associated with decoder linearization are discussed.

We also design and implement a chip comprised of the sum-product circuits with different configurations and sizes in order to study the effect of mismatch on the accuracy of the outputs. Unfortunately, testing of the chip fails as a result of errors in either the packaging process or fabrication.

# Acknowledgements

First of all I have to thank my supervisor Dr. Glenn Cowan who dedicated his precious time in sharing his novel ideas and encouraged me to accomplish this thesis. The idea of this work was initially introduced to me by him. Secondly, I am thankful to my co-supervisor Dr. Warren Gross from McGill University. I also want to make this point that I studied the concepts of analog decoding mostly from the Chris Winstead's PhD. dissertation and hereby I indirectly express my best gratitude to him.

I also appreciate two of my best friends, Mr. Alireza Rabbani and Frank Bernardo for their helpful suggestions and assistances especially in the design and implementation of the chip. I am also grateful to Mr. Tadeusz Obuchowicz in the VLSI lab of Concordia University and Mr. Dong (Hudson) An in the Mixed Signal Lab at McGill University.

This work was supported by the Faculty of Engineering and Computer Science at Concordia University, by the Natural Sciences and Engineering Research Council of Canada, by the Regroupement Stratégique en Microsystème du Québec, and by CMC Microsystems.

Finally, I dedicate this work to my parents and whoever else assisted me either physically or mentally.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Error correcting codes play an important role in modern data transmission systems and digital communication channels including wireless, copper wire and also optical applications. The purpose of coding the information before transmitting them through a noisy channel is that we want to have a reliable set of information at the receiver. The reliability of the received bits or in fact the performance of any error control system depends on the complexity of the coding algorithm which is limited by the well-known statistical limit called the *Shannon Channel Capacity* [1]. Channel capacity is considered as the ultimate rate of communication [2]. There has been a lot of effort among designers to somehow achieve this limit by discovering new coding techniques.

One of the first error-controlling codes which can correct a significant number of errors due to the additive Gaussian noise channel is the *Turbo Code*. Once this code was discovered in 1993 [3], designers tried to produce other types of codes based on *Turbo codes*. They showed that there are other sorts of error controlling codes like Low Density Parity Check codes discovered by Gallager [4] and Block Turbo codes which achieve the performance fairly close to the Shannon Capacity [5, 6, 7, and 8].

In the implementation of decoders, designers have dealt with several issues such as complexity, power consumption, are and speed. A serially implemented decoder must sequentially process each bit of information through the decoding procedure for many times to achieve good performance. In this way, *iterative* decoding would be time consuming and therefore high-speed decoding cannot be achieved. The frequency of clocking the data must be increased to obtain large data rates and this would burn more power. The concept of parallel implementation hence emerged as an alternative approach for performing high-speed iterated estimation of transmitted messages.

Parallel architectures have their own problems. However a very high speed decoding could be achievable as a result of the large parallel implementation. On the other hand it may require a lot of chip area due to the complexity of wiring in digital decoders which results in an increased fabrication cost and again high power consumption.

It was soon discovered that *Analog circuits* can resolve the issue of complexity in digital decoders since very difficult computations can be implemented by very simple analog circuits. However there would still be the same number of blocks in an analog decoder as in a digital decoder. For this reason, many researchers focused their research on analog decoders to eliminate the complexity and high power dissipation of digital circuits and to profit from the energy and area efficient advantages of analog decoders.

Since by its very nature decoding is non-linear [70], it involves non-linear analog circuits to implement decoding operations. Certain analog implementation of weak inversion transistors can be used to implement the required non-linear operation. Analog circuits in this region are in fact turned off and the drawn current from the circuit is very small.

Attracted to the new area of analog decoding, researchers were interested to explore and fully exploit their potentials. Analog Viterbi decoders [17], [18] were the first constructed decoders which outperformed digital implementations by a wide margin [9, 10, 11, 12, and 13]. After Viterbi decoders, iterative codes such as Turbo codes [3], LDPC codes [4], [6] and similar codes [14], like Block Product codes [15] were implemented using analog iterative decoders [32, 37, 40, and 57].

In their investigations, researchers noticed that several important algorithms in the area of error-control coding, signal processing, and computer science can be explained as instances of a general algorithm which operates through message passing on a graph called a factor graph. This algorithm is called *the sum-product algorithm* [4], [21] and several soft-information algorithms such as the Bahl–Cocke–Jelinek–Raviv or (BCJR) algorithm [16] are shown to be instances of this general algorithm. The sum-product algorithm operates on *soft messages* which are described in terms of probability distributions [22, 23, 24, and 25].

Complexity in digital implementations of iterative Turbo codes and low-density parity check codes arises from the complexity of real-number arithmetic of sum-product modules. In contrast, once the sum-product algorithm is implemented with analog circuits, iterations no longer exist and the decoder will be considered as just a continuous-time network which stabilizes when a transmitted codeword has been detected.

Many works have been done in the area of analog decoding. However when compared to digital implementations of similar codes, analog decoders have demonstrated an evident superiority in terms of area and power consumption, yet one cannot find as large codes as have been constructed with digital circuits. The reason can be explained as follows. Since as mentioned before, most analog decoding circuits are composed of transistors that operate in weak-inversion mode (although there are analog decoders that are not in subthreshold region), and by knowing the fact that in this mode, bandwidth is low due to the low drive currents of analog circuits, therefore the throughput (speed) of analog decoders is inherently low. Nevertheless designers have overcome this obstacle through *parallelism.*

A problem with parallel implementation is that the wirings between the modules of sum-product circuit become important. Since the number of sum-product circuit modules increases with the code length, wiring different modules that could be millimeters apart for a large code involves taking into account the unwanted wiring capacitances which will have significant effect on the throughput of decoder. Accordingly, any attempt in order to reduce this effect which corresponds to an increase in the *convergence* speed of the analog decoder would be valuable. One way to reduce the effect of long wires could be the use of buffers to feed the signals.

There are other reasons for not having very large analog codes including the effect of mismatch, lack of automation tools, imperfections of analog circuits and absence of reliable simulation tools for predicting throughput and performance of decoder.

## 1.1 Physical implementations of Analog Decoders

The earliest physical implementations of analog decoders appeared in 1998 when Hagenauer presented the first actual analog decoder [20]. In 1999 Lustenberger, Loeliger et. al published the results for the second fabricated chip [19]. Right after them, in 2000 Moerz, Hagenauer et. al published another real chip results [26]. In all these chips, bipolar junction transistors performed the decoding operation. However, they gave the idea in their papers that subthreshold design also would be possible.

In 2001, Chris Winstead et al. designed a fully-CMOS analog decoder chip [27]. In his thesis, he claimed that his chip has demonstrated micropower analog decoding in a standard CMOS process. In the same year Lustenberger reported a larger BiCMOS analog decoder which employed digital to analog converters at the input in order to simplify the testing however it failed to function correctly [28]. In 2001, again Chris Winstead fabricated a new design of analog decoder circuit including an array of serial to parallel sample and hold capacitors [29, 30].

Gaudet and et. al. were the pioneers in designing the first analog Turbo decoder in 2002 [31] which followed by a complete report for the implemented chip in 2003 [32]. Their Turbo code had a coded length of 48 bits and employed multiple serial input channels which made it possible for the decoder to perform well at higher speeds.

Exploiting the high speed characteristics of SiGe transistors, in 2002, Mores et. al. designed a very high speed analog decoder up to 10 Gbit/sec, however, it implemented a small, weak code [33]. One year later, Huang et. al. from the University of Virginia proposed a high speed SiGe analog Turbo decoder, but due to the failure in their digital-to analog converters at the input, their design was not successful [34,35].

In 2002, the work results of several research groups comprising of researchers at Torino, Padova, and ST Microelectronics led to a design of an analog Turbo decoder used for magnetic recording channels with a codeword length of 500 bits [36]. A revision of the aforementioned decoder was done by Amat et. al. in 2003 by implementing a standard Turbo decoder of 120 length. The fabricated design of this Turbo decoder which was the largest and best performing analog decoder until that date, was published in 2004 [37].

The other significant work on the implementation of analog decoding circuits was the 40 bit length CMOS analog Block Turbo decoder proposed by Perenzoni et. al in 2003 [38]. Their implemented decoder circuit used a serial analog input interface circuit and 16-bit digital output interface. It also included variable gain amplifiers (VGAs) in order to adjust the gain according to the signal to noise ratio of the channel.

Chris Winstead in collaboration with researchers at the University of Utah reported the measurement results for their analog Turbo product decoder in 2004 [30]. This decoder which had been already proposed in 2001 [24] showed a superior performance over known digital designs.

The concept of low-voltage analog decoder circuit took place in 2004 by Nguyen and other researchers at the University of Alberta [22] where for the first time, previously used high power supplies for analog decoders were replaced with the energy efficient power supplies of less than one volt. In the same year, Gioulekas, Birbas and Bilioui designed a low power high speed analog Turbo decoder as one of the first decoders in SiGe technology however to the author of this thesis it is unknown if the implementation results were successful [39].

In the recent years, the area of analog decoding has witnessed numerous valuable works. For instance, in 2005, Hemati and Banihasehmi demonstrated a CMOS analog Min-Sum iterative decoder for an LDPC code [40]. At that time all previously reported analog decoders were based on the exponential characteristic of bipolar or subthreshhold MOS transistors. The proposed circuit was capable of being used for strongly inverted CMOS analog decoders. The implementation results and error correcting performance of the chip in steady state was close to simulation results based on continuous-time iterative decoding and exceeded that of conventional discrete-time decoding.

In 2006, Amat. et. al. and other prominent researchers in this field, proposed a fully analog iterative decoder for a serially concatenated, convolutional code [41]. Their proposed decoder was reconfigurable in both block length and code rate. They also reported the behavioural analysis as well as the impact of precision and mismatch on the performance of their decoder. Amat. et. al., Bendetto and others attempted a CMOS analog decoder for the block length 40 UMTS Turbo code in 2006 [42]. The implementation of the rate-1/3 UMTS turbo code is defined by the 3GPP standard. They also presented a discrete-time model of analog decoding networks which allows very fast simulations as well as predicting complex chip performance in very short time, however the latter has been verified through circuit-level simulations yet this model may give circuit optimization guidelines for complex analog decoder for which circuit-level simulations is impossible. In 2007, Winstead, Gaudet and Schlegel presented a technique for testing analog iterative decoders [43]. They employed digital circuit inside their chip as self-testing equipment which lowered the cost and the complexity compared to alternative mix signal built-in self test techniques. Although this technique was not feasible at the system level, they clearly showed that their decoder core was able to detect catastrophic errors in microseconds.

A novel semi iterative analog Turbo decoding algorithm and its corresponding 40 bits up to 2432 bits decoder architecture configurable were presented in 2007 by Mattieu Arzel and other researchers [44]. The proposed algorithm benefited from a partially continuous exchange of extrinsic information in order to improve decoding speed and correction performance. They also showed that the on-chip area is one tenth of conventionally fully parallelized analog slice turbo decoder.

In 2008, the first integrated realization of a convolutional decoder employing the modified feedback decoding algorithm (MFDA) was presented by Billy Tomatsopoulos and Andreas Demosthenous [45]. The designed decoder uses an analog current-mode computational core and features low-complexity and low-power consumption. Chip measurements were successful and the authors claimed that this approach can be easily extended to the design of an all-analog, soft-decision convolutional decoder.

In order to estimate the performance of decoders including digital and analog, designers compare them in terms of size, speed, power and speed to power ratio which is consumed energy for a decoded bit. Table 1.1.1 summarizes the implemented or synthesized digital and analog iterative decoders from the earliest to very recent.

Table 1.1.1: Summary of published iterative Digital and Analog decoders

| **Reference** | **Year** | **Analog/ Digital** | **Decoding Type** | **Code Length** | **Technology** | **Size (mm²)** | **Speed (bit/sec)** | **Power (W)** | **J/bit** |
|---|---|---|---|---|---|---|---|---|---|
| [47] | 1995 | Digital | PCCC Turbo | 2048 | 0.8μm | 78.32 | 40M | 1.6 | 160n |
| [48] | 2002 | Digital | 3GPP Turbo | 5114 | 0.18μm | 9 | 2.5M | 306m | 123n |
| [49] | 2002 | Digital | LDPC | 2048 | 0.16μm | 52.5 | 1G | 690m | 690p |
| [50] | 2003 | Digital | 3GPP Turbo | 5114 | 0.18μm | 14.5 | 24M | 1.45 | 60n |
| [51] | 2003 | Digital | PCCC Turbo | 432 | 0.18μm | 14.7 | 75.6M | 657m | 8.7n |
| [52] | 2004 | Digital | 3GPP Turbo | 2048 | 0.13μm | 1.07 | 5M | 6.63m | 1.3n |
| [53] | 2004 | Digital | 3GPP Turbo | 5114 | 0.18μm | 0.6 | 5M | 63m | 12.6n |
| [54] | 2006 | Digital | TDMP LDPC | 2048 | 0.18μm | 14.3 | 640M | 787m | 1.23n |
| [55] | 2007 | Digital | LDPC | 1024 | 90μm | 5 | 3.2G | - | - |
| [56] | 2008 | Digital | LDPC | 1944 | 0.1μm | 7.39 | 250M | 76m | 304p |
| [19] | 1999 | Analog | Tailbiting BCJR | 9 | 0.8μm | 1.19 | 100M | 50m | 500p |
| [26] | 2000 | Analog | Tailbiting BCJR | 8 | 0.25μm | 1.68 | 160M | 20m | 125p |
| [27] | 2000 | Analog | Tailbiting BCJR | 4 | 0.5μm | 2.25 | 20M | 3.33m | 165p |
| [32] | 2003 | Analog | PCCC Turbo | 16 | 0.35μm | 1.386 | 13.3M | 185m | 13.9n |
| [37] | 2004 | Analog | 3GPP Turbo | 40 | 0.35μm | 4.07 | 2M | 7.6m | 3.8n |
| [30] | 2004 | Analog | Tailbiting BCJR | 4 | 0.5μm | 0.083 | 2M | 1m | 500p |
| [22] | 2004 | Analog | Hamming Decoder | 4 | 0.1μm | 0.043 | 444K | 283u | 0.64n |
| [40] | 2005 | Analog | LDPC | 8 | 0.18μm | 0.57 | 80M | 5m | 60p |
| [57] | 2005 | Analog | Tailbiting BCJR | 11 | 0.18μm | 0.0266 | 135M | 2.69m | 20p |
| [57] | 2005 | Analog | Turbo Product | 121 | 0.18μm | 2.85 | 1G | 86.1m | 86p |
| [41] | 2006 | Analog | SCCC | Up to 2400 | 0.18μm | 36 | 100M | 40m | 400p |
| [42] | 2006 | Analog | UMTS Turbo | 40 | 0.35μm | 9 | 2M | 10.3m | 11.2n |
| [43] | 2007 | Analog | Hamming | 4 | 0.18μm | 0.138 | 3.7M | 13m | 3.5n |
| [44] | 2007 | Analog | Semi iterative Turbo | Up to 2432 | 0.25μm | 37 | 310K | 12.48m | 40n |
| [45] | 2008 | Analog | Convolutional MFDA | Up to 1024 | 0.6μm | 0.5 | 1M | 2.45m | 2.45n |

From the table there is not a significant superiority in terms of chip area, total power consumption and power to speed ratio of the analog decoders compared to the digital decoders especially for large codes, yet analog implementation have the potential to excel the digital decoders in the mentioned aspects. That is ignoring the block length for a particular code, the analog designs outperform their digital counterparts in terms of size, power and power/speed by several orders of magnitude. However, there is still a limitation in the block length for the analog implementations due to the wiring complexity and the issue of speed. Most of the industrial applications deal with larger bit lengths (i.e. a few thousand). In order to remedy this problem some designers have attempted hardware solutions like reusing analog hardware and performing the interleaving in the digital domain [44], [46] and by doing so they significantly lowered the complexity of the component decoder.

## 1.2 Contributions of the Thesis

The present work can be observed from two aspects. Firstly in this thesis, the basic circuits for an iterative analog de

coder based on the sum-product algorithm, which are equality and check nodes, have been redesigned in 90 nm technology. The original transistor-level design of such circuits is illustrated in [57]. After a complete study and a thorough survey on the designed circuits in 0.18 μm technology, the two circuits were resized for a new technology. In this survey, the effect of mismatch on the outputs of individual block with different sizing was also studied which followed by implementing a chip comprised of these circuits.

Apart from the chip design experience which can be considered as a physical contribution of this thesis, the main contributions of this thesis are the following theoretical contributions:

- Applying a technique used in current mode circuits to improve the speed of analog decoders. This technique can also be applied to other analog circuits for which the input impedance is quite high and therefore the speed is low. The input impedance of the equality and check nodes plays an important role in determining the decoding speed since as a result of wiring these circuits in a real decoder, large wiring capacitance can be added to the input nodes which may have a significant effect on the overall speed. Hence it seems essential to somehow lower this input impedance. It is shown in this thesis that current mode techniques can be applied to solve this issue.
- Having the dimensions for the layout of an individual equality and check node, the worst case wiring capacitance for the longest path in a large code has been estimated.
- Applying the linearization technique to a given LDPC decoder with intent to better study the dynamics of decoders. Thus far there has been no such analytical expression in the literature. It is assumed that by merely having the *H*-matrix for a particular code such as LDPC code and by modelling the interconnecting wiring capacitances as well as the input impedances of every block as RC circuits, one can write the linearized equations which is supposed to provide the required information regarding the dynamics of the system. In fact, the concept of state-space has been employed in the linearization of the non-linear LDPC decoder. It

turned out though that the initial state of the linearized decoder affects the decoding operation that is for the uniform probability mass as the initialization point the decoder fails to function properly. Initial states other than this point (expect other valid codewords which has not been studied in this work) would also mislead the decoding from its correct direction.

## 1.3 Order of the Thesis

Chapter 2 of this thesis gives a brief review on the fundamental of error-control coding and decoding algorithms in a digital communication channels including the principle definitions and basic theorems of coding theory. The tanner graph is also presented as a graphical representation of codes which is used to satisfy the parity check constraints between the bits of a codeword. An introduction to typical decoding algorithms including the sum-product and min-sum algorithms is given. Furthermore, a summary of the known error-controlling codes such as Turbo codes, LDPC codes and Block Turbo codes is presented. The structure of regular and irregular LDPC codes is discussed and a general mathematically-graphically described procedure for decoding LDPC codes based on the sum-product algorithm is presented.

In Chapter 3, translinear circuits and principles of CMOS translinear circuits are introduced. Two fundamental nodes for realizing the sum-product algorithm or the canonical CMOS sum-product circuits (equality and check nodes) are shown. The modified circuits of equality and check nodes for an LDPC decoder based on Winstead's thesis [57] in 90 nm technology are presented. The author has already designed and implemented a chip comprising various configurations for the equality and check nodes with different transistors sizes to observe the effect of mismatch on the performance of these nodes. However, the chip failed to work properly due to possible problems in layout, packaging bounding, etc.

Chapter 4 presents a technique which is normally used in current mode circuits to improve the speed of analog decoders. This approach is in fact a modified current-mirror circuit with an OTA in feedback used to lower the high input impedance of basic current-mirrors in the circuits of analog decoder thereby improving the speed.

In Chapter 5, the author simulates an LDPC decoder in the MATLAB by expressing the parity equations of its *H*-matrix and by taking into account the delay between the nodes of the corresponding factor graph. Moreover by applying the linearization technique for non-linear systems, it is attempted to linearize the LDPC decoder around a linearization point. The idea of linearization finally results in a failure due to a problem with the initialization point which is illustrated thoroughly at the end of this chapter.

Chapter 6 presents the test plan for the implemented chip which is already introduced in Chapter 3 and explains what simple measurements suggested that something is not right with the chip. Finally, Chapter 7 gives conclusions and proposes future work.

# Chapter 2

# Error-Control Codes and Decoding Algorithms

## 2.1 Digital communication channels

Any digital communication channel is prone to be corrupted by noise. Thus the receiver should have the capability to somehow detect and correct the errors. An error-control decoder is one of the important components in a digital receiver which is used to perform this task.

Typically the transmitting device encodes data by adding parity-check information. If we assume the number of data bits as *k*, the encoder would add-up *n-k* parity bits so that the total number of encoded bits of information becomes *n*. This encoded information is then sent through a channel to the receiving device. Due to the noisy structure of the channel, the receiving information will no longer be the same as the original one. For example, in an additive white Gaussian noise (AWGN) channel which is of great interest in communication systems, the received message at the receiver is the original distinct bits of information plus a zero-mean, Gaussian distributed noise, thus a decoder is employed in this system to generate a good estimation of the original encoded information message. Fig 2.1.1 depicts a model for a digital communication system where $\underline{u}$ and $\underline{x}$ represent the information and encoded messages respectively, $\underline{n}$ is the additive noise of the channel and $\underline{r}$ is the channel observation by the receiver. The complete model of a communication system may include some extra components such as a modulator and a demodulator at the transmitter and the receiver sides respectively, however they are not included in our discussion.

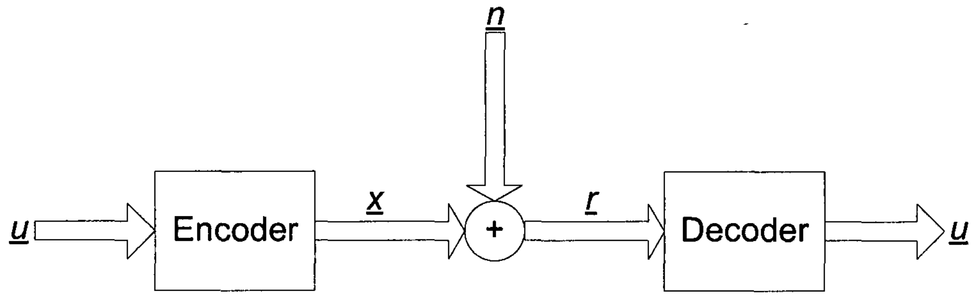

Fig 2.1.1: A model for a digital communication system

In order to have an estimation on the channel observation $\underline{r}$, two types of error-control algorithms can be employed; *soft-decision* and *hard-decision* algorithms which will be studied here.

## 2.1.1 Soft and Hard decision algorithms

In a hard decision algorithm, a decoder must judge on every single bit after digitizing the received data from the channel. In other words, error detection in this kind of receiver will be made after an analog to digital conversion has been done on the received analog information with one bit of resolution.

Unlike hard decision algorithm, the soft decision algorithm either use more than one bit of resolution or keep the analog nature of $\underline{r}$, while translating the analog information into probability format. The decoder could detect the error by performing probability calculations on these soft information bits.

## 2.1.2 Probability, Likelihood ratio and Log-likelihood ratio Domains

In soft decision receivers, there are three main distinct domains in which decoding is defined; *probability*, *likelihood* and *log-likelihood* domains.

In the probability domain, information bits are treated as zero and one probabilities of $p_0$ and $p_1$ respective for performing probability calculation in the soft decision algorithm. The ratio of $\frac{p_0}{p_1}$ for every bit is defined as the likelihood ratio (LR) and this representation domain is called the likelihood ratio domain. Finally, the Napierian logarithm of the likelihood ratio $\ln(\frac{p_0}{p_1})$ is presented in the log-likelihood ratio (LLR) domain. The application of each of these domains will be discussed later in the thesis.

Let's consider antipodal transmissions such as *Binary Phase-Shift Keying* (BPSK) modulation in an *Additive White Gaussian Noise* (AWGN) channel. Each element of $\underline{ri}$ would then have a Gaussian distribution with mean ±1 and variance $N_0/2$, where $N_0$ is the power density of the channel's noise. Then it can be shown that the log likelihood ratio (LLR) for the received sample $\underline{ri}$ has the following equation [57].

$$\ln(\frac{p_{-1}}{p_1}) = \frac{4}{N_0} r_i = X_i \qquad (2.1.1)$$

LLR and probability representation domains are related to each other by the following equations:

$$p_{-1} = \frac{e^{X_i}}{1+e^{X_i}} \tag{2.1.2}$$

$$p_1 = \frac{e^{-X_i}}{1+e^{-X_i}} \tag{2.1.3}$$

### 2.1.3 The Shannon Capacity

In communication theory, the *Shannon capacity* or the *Shannon limit* is presented as a statistical limit for the transmission rate of a specific channel. Claude Shannon showed in 1948 that in order to achieve a reliable transmission over a noisy channel by employing error-control codes, the transmission rate of information bits should not be greater than the channel's capacity which is called the Shannon limit [58]. The performance of error-control codes are preferred to be close to this limit. Turbo codes and LDPC codes are known as Shannon capacity approaching codes.

## 2.2 Linear Block Codes

A *linear block code* refers to a block code which is defined in a linear space. For example if $\underline{x}_1$ and $\underline{x}_2$ are two different codewords in the space of a linear block code, $\underline{x}_3 = \underline{x}_1 + \underline{x}_2$ also belongs to this space. Here, we limit our discussion to binary linear block codes for which the addition is defined in the binary domain. Even number of 1's add up to 0 while odd number of 1's in binary addition gives 1.

### 2.2.1 *G* and *H* matrices

Based on the communication system model presented in Fig 2.1.1, an encoder is located at the transmitter side in order to convert binary vector $\underline{u}$ to another binary vector $\underline{x}$ in the linear code space. For this reason a Generator matrix or *G*-matrix is used to map the uncoded vector $\underline{u}$ to a codeword.

If $\underline{u}$ is a $1 \times k$ binary vector and $G$ is a $k \times n$ matrix where $n$ is the number of bits in a codeword, then the matrix product would form a $1 \times n$ codeword vector $\underline{x}$ in the new space.

$$\underline{x} = \underline{u}.\, G \tag{2.2.1}$$

The coding procedure is done by adding parity bits to the input bits in a logical manner which also defines the *code rate* for specific coding algorithm. The term code rate or information rate is defined by ratio of the non-redundant bits of information to the total bits of information. For example if the code rate is $k/n$, the code generator may generate a total of $n$ bit of data where only $n$-$k$ of them are redundant.

At the receiver, we need to recover the original bits of information $\underline{u}$. Thus, a decoder must detect the errors caused by the channel noise and perform the best estimation on the received bits. *Parity check matrix* or *H-matrix* for the a linear block code is defined based on $G$ matrix and satisfies $G.H^T = 0$. Therefore, for any codeword $\underline{x}$ we must have

$$\underline{x}.H^T = 0 \tag{2.2.2}$$

In order to find the transmitted codeword, several algorithms have been presented which will be discussed in Section 2.3.

## 2.2.2 Factor and Tanner graphs

One common way to graphically represent the linear space of codewords is through *factor graphs* on which the *Boolean constraint functions* are satisfied. Below a simple factor graph for the following constraint is shown. $f(x, y, z) = f_1(x, y, z).f_2(x, y, z)$ has been presented through its factor graph in Fig 2.2.1. It is observed that a factor graph is comprised of two distinct nodes; *variable nodes* and *constraint nodes* which are located at the top and at the bottom respectively. They are also intermediate lines called *edges* which connect different nodes of the graph based on the constraint functions.

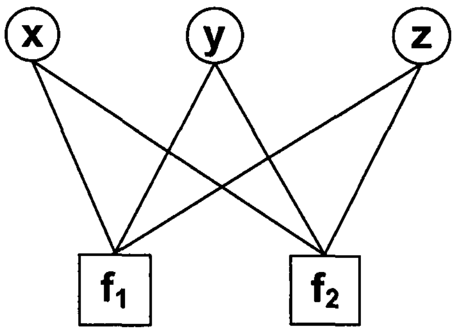

Fig 2.2.1: A simple factor graph corresponding to $f(x, y, z) = f_1(x, y, z).f_2(x, y, z)$

By convention, variable nodes are preferred to be connected to only one constraint node. In order to satisfy this rule, we modify the factor graph by adding new constraint nodes called the equality nodes such that there will be only one edge between each variable node and its corresponding equality node. This new graphical representation is called a *Normal graph* [59] and is depicted in Fig 2.2.2.

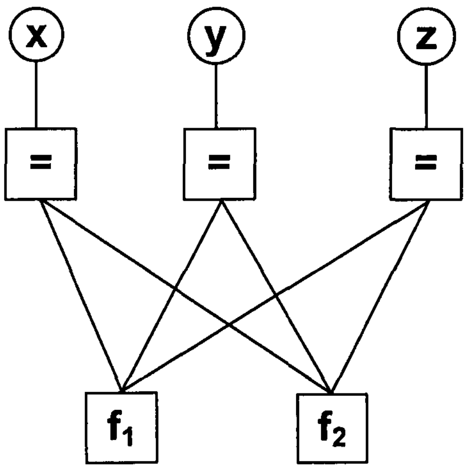

Fig 2.2.2: The normal graph of $f(x, y, z) = f_1(x, y, z).f_2(x, y, z)$

If the parity-check equation for a binary block code is presented through the factor and the normal graph, we will come up with new graphical representations called a *Tanner graph* and a *Normalized Tanner graph* respectively. Since for a codeword $\underline{x}$ to be verified by the decoder, $\underline{x}.H^T = 0$, the Tanner graph will be the constraint graph for $H$. For instance assume that a parity check matrix $H$ is as below:

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

It turns out that there are seven parity check equations for this $H$-matrix which are derived by multiplying every row of $\underline{x}$ by $H^T$. For example the first equation which corresponds to the first row of $H$ or the first column of $H^T$ would be $\underline{x}_1 + \underline{x}_2 + \underline{x}_4 = 0$. This implies that constraint nodes at the bottom of the Tanner graph should be replaced by binary addition or Exclusive-OR nodes since we deal with only 0 and 1 bits in communication systems. These parity nodes whose function is to check the parity equation are known as *check nodes*. The corresponding normalized Tanner graph for the above *H*-matrix has been shown in Fig 2.2.3.

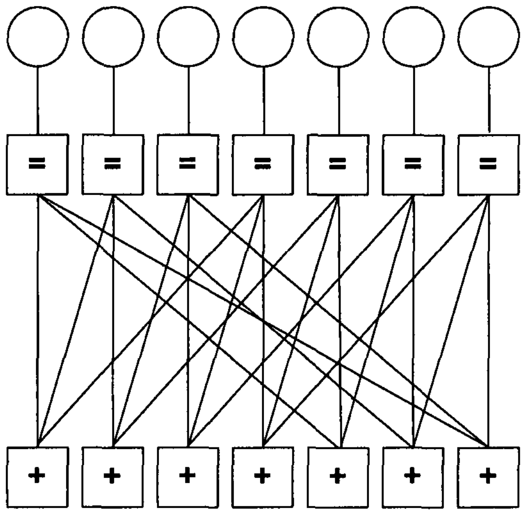

Fig 2.2.3: The normalized Tanner graph for the presented *H*-matrix

## 2.3 Some popular Error-Control Codes

In this section some of the widely used error-control codes will be briefly presented.

### 2.3.1 Turbo Codes

The introduction of the original turbo codes has been as early as 1993 when *Parallel Concatenated Convolutional Codes* (PCCC) was presented [3]. After that many other classes of turbo code were discovered including S*erially Concatenated Convolutional Cod*es and *Repeat-Accumulate Codes* [41].

One of the most significant advantages of turbo codes is that as stated earlier, this code together with LDPC code has the closest performance to the Shannon limit. However, relative high decoding complexity as well as the inevitable latency due to wiring of such codes could make them inappropriate for very fast applications.

### 2.3.2 Low-Density Parity Check Codes

Low density parity check codes or LDPC codes are a class of large linear block codes. The name of low density comes from the fact that the density of 1s is small in comparison with the number of 0s in the *H*-matrix. In other words, they have a sparse parity check matrix.

These codes were first introduced by Gallager in his PhD thesis in early 1960s [4]. But due to the complicated computations required to implement the decoder and encoder for such codes, LDPC codes were forgotten for a few decades. It was not until the work of MacKay [6], that the full potential of these powerful codes became well-known. Basically, there are two different methods to represent LDPC codes; *matrix and graphical representations.*

The very first LDPC codes were introduced through their parity check matrices by Gallager. Based on [62], a Gallager code (LDPC code) is defined by $(dv,dc)$ where $dv$ is the number of 1s in every column and $dc$ is the number of 1s at each row of the parity check matrix. For an $(m \times n)$ parity check or *H*-matrix, if $n$ represents the code length, then the number of rows $m$ can be found from $m = ndv/dc$ if and only if it is a regular LDPC code, otherwise it is an irregular LDPC code. The $(7 \times 7)$ *H*-matrix example in Section 2.2.1 is the matrix representation for a 7-bits length regular LDPC decoder for which $dv$ and $dc$ are both equal to 3. The iterative message passing decoding or the sum-product algorithm which will be presented later can be directly applied to the graphical representation or the Tanner graph of LDPC codes.

### 2.3.3 Block Turbo Codes

There are other types of error-control codes in the literature such as *Block Turbo Codes* (BTC), which are also known as *Block Product Codes*. These sorts of codes have a two-dimensional construction including simple linear block codes. BTC is iterative in nature, which is why the term Turbo has been used for it. Their codeword is comprised of row codes and column codes which finally results in a rectangular codeword structure. Another thing to know about BTCs is that since they are comprised of simple block codes, they are easy to construct. Moreover, some BTCs with very short block lengths have been shown to approach the Shannon limit while other iterative codes such as LDPC codes need to be significantly larger to approach this limit [57].

## 2.4 Introduction to Decoding Algorithms

Among various algorithms, we are interested in two common algorithms which operate on factor or Tanner graphs. They are namely the *sum-product algorithm* and the *min-sum algorithm* which will be studied in this section.

### 2.4.1 The sum-product algorithm

One of the well-known decoding algorithms which has been widely used by designers is the sum-product algorithm [21]. Since the functionality of this algorithm is based on probabilities which are also known as soft messages or beliefs, it is often referred to probability algorithm. Shannon capacity approaching codes such as Turbo and LDPC codes are often decoded based on this algorithm.

Most sum-product factor graphs are comprised of nodes each having three edges. Nodes with more than three edges can be replaced with the cascade of several nodes having only three edges. More than three edges nodes can be modified to the cascade of several three edges nodes. The sum-product algorithm computes the overall conditional probabilities by calculating or processing local constrains at every node iteratively. That means for every single equality node or check node in a sum-product circuit, local processing should be performed in order to satisfy the specific constraint related to that node. If we assume a three-edge node having two inputs $x$, $y$ and one output $z$ as shown in Fig 2.4.1, the local constraint associated to this node would be $f(x, y, z) = 0$.

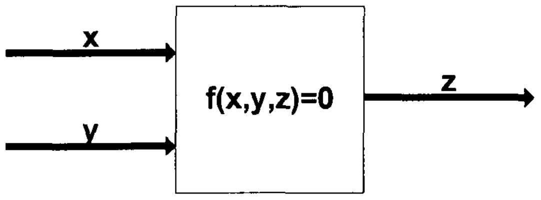

Fig 2.4.1: Function node and its associated constraint

In a factor or Tanner graph of the sum-product algorithm, the function of equality and check nodes must be defined in the three aforementioned domains. If $P_1$ and $P_0$ denote the probabilities of one and zero for every edge of the graph in probability domain, the equality node is defined as

$$P_{0z} = \eta P_{0x} P_{0y} \tag{2.4.1}$$

$$P_{1z} = \eta P_{1x} P_{1y} \tag{2.4.2}$$

where $\eta$ is a positive constant for which $P_{0z} + P_{1z} = 1$ and thereby $\eta = \frac{1}{P_{0x}P_{0y} + P_{1x}P_{1y}}$.

And for a check node we have

$$P_{0z} = P_{0x}P_{0y} + P_{1x}P_{1y} \tag{2.4.3}$$

$$P_{1z} = P_{1x}P_{0y} + P_{0x}P_{1y} \tag{2.4.4}$$

If we substitute $\frac{P_0}{P_1}$ of every edge by $Y$ in the likelihood ratio domain, the above equations can be rewritten as:

$$Y_z = Y_x \times Y_y \tag{2.4.5}$$

for the equality nodes and

$$Y_z = \frac{1 + Y_x Y_y}{Y_x + Y_y} \tag{2.4.6}$$

for the check nodes.

Similarly by replacing $\ln(\frac{P_0}{P_1})$ with $X$ in the log-likelihood ratio domain, one will arrive at the following expressions for equality and check nodes respectively.

$$X_z = X_x + X_y \tag{2.4.7}$$

$$X_z = 2\tanh^{-1}(\tanh(X_x/2) \times \tanh(X_y/2)) \tag{2.4.8}$$

So far we have defined the local constraints for the two function nodes (i.e. equality and check nodes). Before presenting the algorithm lets introduce some notation.

- Variable nodes and received messages are denoted as $v_i$ and $y_i$ respectively.
- The equality nodes are presented as $g_i$ while the check nodes are indicated as $f_j$.
- Probability of the received information bits through variable nodes is shown as *Pi* which is equal to the probability of individual variable node provided that the particular $y_i$ has been received by that node. Thus we can use the following notation for *Pi* as the probability of being one at the input: $Pi = P(v_i = 1 | y_i)$.
- Down going messages from the equality node $g_i$ to the check node $f_j$ is labeled as $q_{ij}$.
- Upcoming message from check node $f_j$ to equality node $g_i$ is denoted as $r_{ji}$.

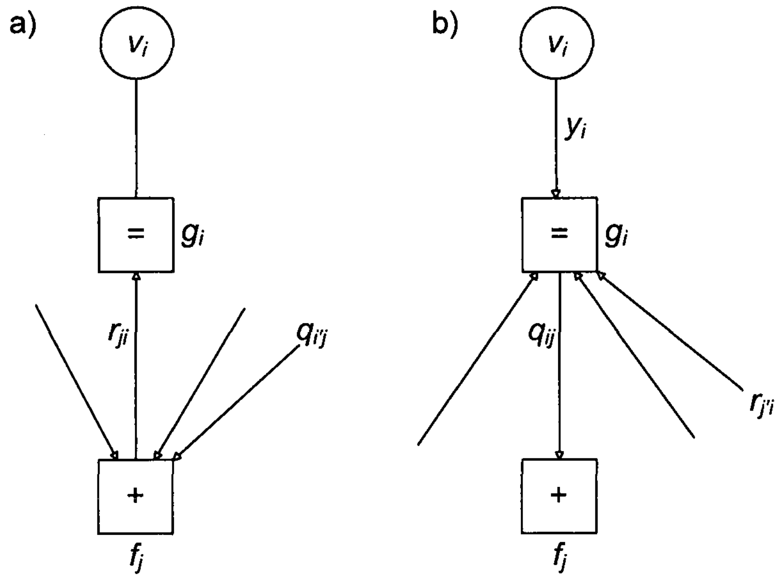

Fig 2.4.2: Illustrating the sum-product algorithm; a) Step 2, b) Step 3

The sum-product algorithm can be described through following steps:

1. All variable nodes and their subsequent equality nodes send their $q_{ij}$ messages to the corresponding check nodes. Since this is the first iteration of the algorithm and no information other than the probabilities of received bits are available, therefore $q_{ij}(1) = P_i$ and $q_{ij}(0) = 1 - P_i$. Note that even if the initially received information bits from the channel form a valid codeword, they should be passed down to the check nodes since no hard decision can be made as this point.

2. Once all the check nodes receive $q_{ij}$ messages, they calculate their response messages $r_{ji}$ back to the equality nodes. For a three-edge check node *($f_j$)*, we assume $q_{1j}$ and $q_{2j}$ each including the probabilities of zero and one as the two inputs coming from different equality nodes. The output probability of zero $r_{ji}(0)$ which goes back to the corresponding equality node *($g_i$)* can be found based on (2.4.3) as:

$$r_{ji}(0) = q_{1j}(1)q_{2j}(0) + q_{1j}(0)q_{2j}(1) \tag{2.4.9}$$

If in the above equation we replace $q_{1j}(0) = 1 - q_{1j}(1)$ and $q_{2j}(0) = 1 - q_{2j}(1)$, one would end up with the following formula:

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2}(1 - 2q_{1j}(1))(1 - 2q_{2j}(1)) \tag{2.4.10}$$

Consequently for check nodes with more than two edges (2.4.10) can be extended as:

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2}\prod_{i' \in f_j \setminus i}(1 - 2q_{i'j}(1)) \tag{2.4.11}$$

and

$$r_{ji}(1) = 1 - r_{ji}(0) \tag{2.4.12}$$

Equations (2.4.11) and (2.4.12) compute the returning messages to the equality node $g_i$ by calculating sum of the products for all $q_{i'j}'s$ except $q_{ij}$ which has also been illustrated in Fig 2.4.2a.

3. At this point, two updates must be done. Firstly, the down going messages from the equality nodes that is shown graphically in Fig 2.4.2b must be computed through following equations:

$$q_{ij}(0) = \eta_{ij}(1-Pi)\prod_{j' \in g_i \setminus j} r_{j'i}(0) \tag{2.4.13}$$

$$q_{ij}(1) = 1 - q_{ij}(0) \tag{2.4.14}$$

where $\eta_{ij}$ is a positive constant to ensure that $q_{ij}(0) + q_{ij}(1) = 1$.

At this step the first iteration is completed. Now the decoder will also update its current estimation of variable $v_i$ based on the following equations.

$$Q_i(0) = \eta_{ij}(1-Pi)\prod_{j \in g_i} r_{ji}(0) \tag{2.4.15}$$

$$Q_i(1) = \eta_{ij}Pi\prod_{j \in g_i} r_{ji}(1) \tag{2.4.16}$$

Hard decision will be made by comparing $Q_i(1)$ and $Q_i(0)$ and by voting for the bigger one. This is usually done within the analog comparators at the output of the decoder.

$$\hat{v}_i = \begin{cases} 1 & if \quad Q_i(1) > Q_i(0) \\ 0 & \quad else \end{cases} \tag{2.4.17}$$

Upon matching of the estimated codeword to the valid codeword and hence fulfilling the parity check equation, decoding algorithm may terminate at this stage, otherwise go to step 2. Here, we say that if one iteration was enough, then the initially received bits of information have been a valid codeword that can be decoded at this step. However, if more than a single iteration was required for this algorithm, it means that the error-correction needs to be done prior to the final hard-decision.

The above explained algorithm is the basis of the sum-product algorithm used in decoders either analog or digital.

## 2.3.2 The min-sum algorithm

Another graphically based decoding algorithm is the min-sum algorithm that can be performed with minor modification to the sum-product algorithm [60]. The min-sum algorithm is also known as a variant of the *maximum-likelihood* (ML) sequence decoding rather than the *a posteriori probability* (APP) decoding which is the most straightforward applications of the sum-product algorithm. The reader could study about ML and APP algorithms in [57]. If we assume that $l$ indicates the number of iteration in the min-sum algorithm which is positive, then the messages passing between equality node ($g_i$) and check node ($f_j$) are given in the log-likelihood ratio domain by [40]:

$$q_{ij}^{(l)} = Pi + \sum_{j' \in g_i \backslash j} r_{j'i}^{(l-1)} \tag{2.4.18}$$

$$r_{ji}^{(l)} = \left( \prod_{i' \in f_j \backslash i} sign(q_{i'j}^{(l)}) \right) \min_{i' \in f_j \backslash i} (|\, q_{i'j}^{(l)} \,|) \tag{2.4.19}$$

where *sign(.)* function is -1 for negative numbers and is +1 for non-negative numbers.

Note that although the same notation as in the sum-product equations has been applied in (2.4.18) and (2.4.19), here the variables should be considered as log-likelihood values.

The output of individual variable node will be updated after each iteration just as in the sum-product algorithm. A hard decision will be made upon satisfying the valid codeword which determines the termination of decoding or up to given maximum iteration number. The variable nodes are updated by the following equation:

$$Q_i^{(l)} = Pi + \sum_{j' \in g_i} r_{j'i}^{(l-1)} \tag{2.4.20}$$

Other types of decoding algorithms exist, most of which are based on the sum-product algorithm. As an example, M*argin propagation* (MP) algorithm that is presented in [61] can be used for approximating the log-sum factors in a conventional sum–product based LDPC decoding algorithm. It was also shown in [61] through simulations that BER performance of margin propagation based LDPC decoders is nearly identical to the sum–product decoders and is superior to the min-sum LDPC decoders.

In the following chapters we will study the analog implementation of the sum-product algorithm which was introduced here. Throughout the rest of the thesis, we will be more concerned about the dynamics of the analog circuits used in the analog decoders.

# Chapter 3

# Analog sum-product circuits

For an analog decoder based on the sum-product algorithm, equality and check nodes with reasonable precision and speed must be designed. In order to have a correct hard decision on the decoded data, the individual block of equality or check node must deliver its output to the following block with a reasonable accuracy. Also, the timing delay associated with each of these nodes can affect the total convergence speed of the decoder. Therefore, designing equality and check nodes are of great importance.

In this chapter, a complete procedure for the design of the canonical sum-product circuits, equality and check nodes is presented. Here, circuits presented in [57] that are in $0.18\,\mu$m technology are modified for 90 nm technology. The effect of mismatch on the outputs of individual blocks with different sizing will be studied. At the end of this chapter, we will present the internal view of an implemented chip comprising equality and check nodes with different topologies and sizes that unfortunately failed to function properly, as we discuss later in Chapter 6.

Before explaining the design procedure, we are going to study the operating regions of a MOS transistor. Later, we will introduce the *Translinear principle* and *Translinear circuits* which are the basis for the analog sum-product circuits.

## 3.1 MOS transistor operation regions

Three regions of operation can be defined for MOS transistors including strong inversion, *weak inversion* also called the *sub-threshold region*, and the *moderate-inversion region.* A three terminal MOS transistor is shown in Fig 3.1.1. Here we are going to have a quick review of each of the mentioned regions.

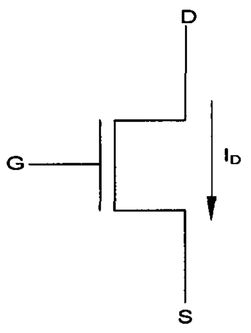

Fig 3.1.1: Basic MOS transistor

### 3.1.1 Strong inversion region

A MOS transistor is in strong inversion region once its gate-source voltage is greater than the threshold voltage (i.e. $V_{gs} >> V_{th}$). Hence the transistor is said to be ON which means a channel has been made between the drain and source terminals.

If in this condition the drain-source voltage becomes high enough so that $V_{ds} > V_{gs} - V_{th}$, then we say the transistor is *saturated* and the current is governed by:

$$I_D = \frac{1}{2}\mu C_{ox}\frac{W}{L}(v_{gs} - V_{th})^2 \qquad (3.1.1)$$

Where $W$ and $L$ represent the width and length of the transistor and $\mu$ and $C_{ox}$ are the mobility of the carriers and the oxide capacitance, respectively.

For $V_{ds} < V_{gs} - V_{th}$, MOS transistor is not saturated anymore and behaves as a voltage dependant resistor which is linear for very small drain-source voltages.

### 3.1.2 Weak inversion region

When $V_{gs}$ of an NMOS transistor is below the threshold voltage it can be either in weak or moderate inversion region. Designers have separated the weak and moderate inversion regions by defining the device's specific current $I_S$ such that when the device current is less than one-tenth of $I_S$, so the transistor is in weak inversion and if it is between one-tenth of $I_S$ and ten times $I_S$ it is in the moderate inversion region. Yet, there is not an exact boundary between these two regions. For very low gate-source voltages ($V_{gs} << V_{th}$) the transistor is said to be in (deep) weak inversion region where most of the current flow is due to diffusion and for gate-source voltages close to the threshold voltage ($V_{gs} \approx V_{th}$) it is in moderate inversion region which is something between the weak and strong inversion. In the weak inversion region, transistor current $I_D$ is governed by [57]:

$$I_D \approx I_0 e^{\frac{v_{gs}}{nU_T}}(1 - e^{-\frac{v_{ds}}{nU_T}}) \qquad (3.1.2)$$

where $I_0$ is a small constant current which is related to $I_S$ , $n = \frac{C_{ox} + C_{dep}}{C_{ox}}$ is often called the subthreshold slope factor and $C_{dep}$ is the depletion capacitance. Finally $U_T = kT/q \approx$ 25mV is the thermal voltage.

It is obvious that for sufficiently large $V_{ds}$, (3.1.2) will be reduced to:

$$I_D \approx I_0 e^{\frac{v_{gs}}{nU_T}} \qquad (3.1.3)$$

This condition is often referred to *saturation region* in weak inversion and is determined when $V_{ds}$ is sufficiently larger than $4U_T \approx 100$ mV(i.e. 200 mV).

For small $V_{ds}$ the transistor is no longer in saturation or in other word is *unsaturated* in the weak inversion. Therefore the second term in (3.1.2) cannot be ignored and consequently $I_D$ is comprised of two currents opposite directions; forward current $I_f$ which is the desired current and reverse current $I_r$.

$$I_D = I_0 (e^{\frac{v_{gs}}{nU_T}} - e^{\frac{v_{gd}}{nU_T}}) = I_f - I_r \tag{3.1.4}$$

## 3.2 The translinear principal

The translinear principle states that in a translinear loop containing translinear devices, the product of clockwise currents is equal to the product of counter-clockwise currents. Translinear devices may include BJT and MOS transistors in weak inversion [63]. In this thesis MOS transistors operating in weak inversion are assumed to be the translinear devices which follow the translinear principle while arranged in a loop of gate source voltage drops. There must be an equal number of $V_{gs}$ rises as $V_{gs}$ drops. This is the conventional translinear principle. However there is also a voltage-translinear principle when all transistors in the loop are biased in strong inversion [63].

Consider the circuit configuration of Fig 3.2.1. Writing the KVL equation in the loop will give:

$$-v_{gs1} + v_{gs2} - v_{gs3} + v_{gs4} = 0 \tag{3.2.1}$$

If all MOS transistors in the loop are biased in weak inversion while saturated, based on (3.1.3) one can write (3.2.1) as:

$$-nU_T \ln(\frac{I_1}{I_0}) + nU_T \ln(\frac{I_2}{I_0}) - nU_T \ln(\frac{I_3}{I_0}) + nU_T \ln(\frac{I_4}{I_0}) = 0 \qquad (3.2.2)$$

$$I_1 I_3 = I_2 I_4 \qquad (3.2.3)$$

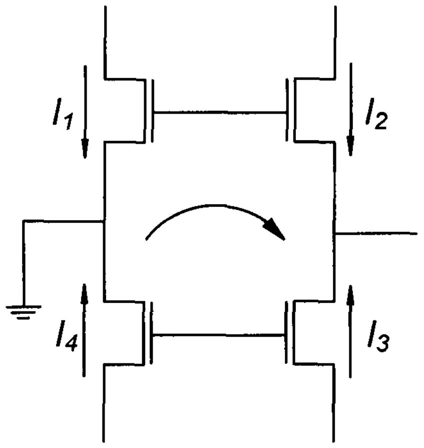

Fig 3.2.1: A basic translinear loop

Equation (3.2.3) reveals the translinear principle in a translinear loop containing translinear MOS transistors.

## 3.3 The canonical sum-product circuits

Analog decoding based on the sum-product algorithm deals with probabilities as interacting signals such that each signal represents probabilities of a variable being 1 and 0. The structures of canonical sum-product circuits are shown in Fig 3.3.1 and 3.3.2. Fig 3.3.1 is showing the standard equality node and Fig 3.3.2 presents a canonical check-node. As it is observed, each circuit is comprised of two inputs and one output which are in current mode. Taking into account the probabilities of 1 and zero for individual input and output we will have a total of 4 inputs and 2 outputs.

The sum-product circuits of Fig 3.3.1 and 3.3.2 can be seen as an application of basic Gilbert cells comprised of several translinear loops [64]. Based on the translinear principal, the intermediate currents for equality nodes are found as below:

$$I_{z0(eq\_node)} = \frac{I_{x0} I_{y0}}{I_{x0} + I_{x1}} \tag{3.3.1}$$

$$I_{z1(eq\_node)} = \frac{I_{x1} I_{y1}}{I_{x0} + I_{x1}} \tag{3.3.2}$$

And for the check nodes are:

$$I_{z0(ch\_node)} = \frac{I_{x0} I_{y0} + I_{x1} I_{y1}}{I_{x0} + I_{x1}} \tag{3.3.3}$$

$$I_{z1(ch\_node)} = \frac{I_{x1} I_{y0} + I_{x0} I_{y1}}{I_{x0} + I_{x1}} \tag{3.3.4}$$

Equations (3.3.1) to (3.3.4) differ slightly from the equations (2.3.1) up to (2.3.4) for equality and check nodes. The use of renormalization circuits at the top of the two nodes makes it possible to derive the complete equations. Since the renormalization circuits are in fact translinear circuits, therefore according to the governing principle we have the following equations for both nodes:

$$I_{out0} = \frac{I_{z0} I_{U}}{I_{z0} + I_{z1}} \tag{3.3.5}$$

$$I_{out1} = \frac{I_{z1} I_{U}}{I_{z0} + I_{z1}} \tag{3.3.6}$$

where $I_U$ is the global unit current used to boost the attenuated currents of $I_{z0}$ and $I_{z1}$ thereby calibrating the output currents to change from 0 to $I_U$ (0 for zero output current and $I_U$ for the maximum output current). This calibration is often known as renormalization.

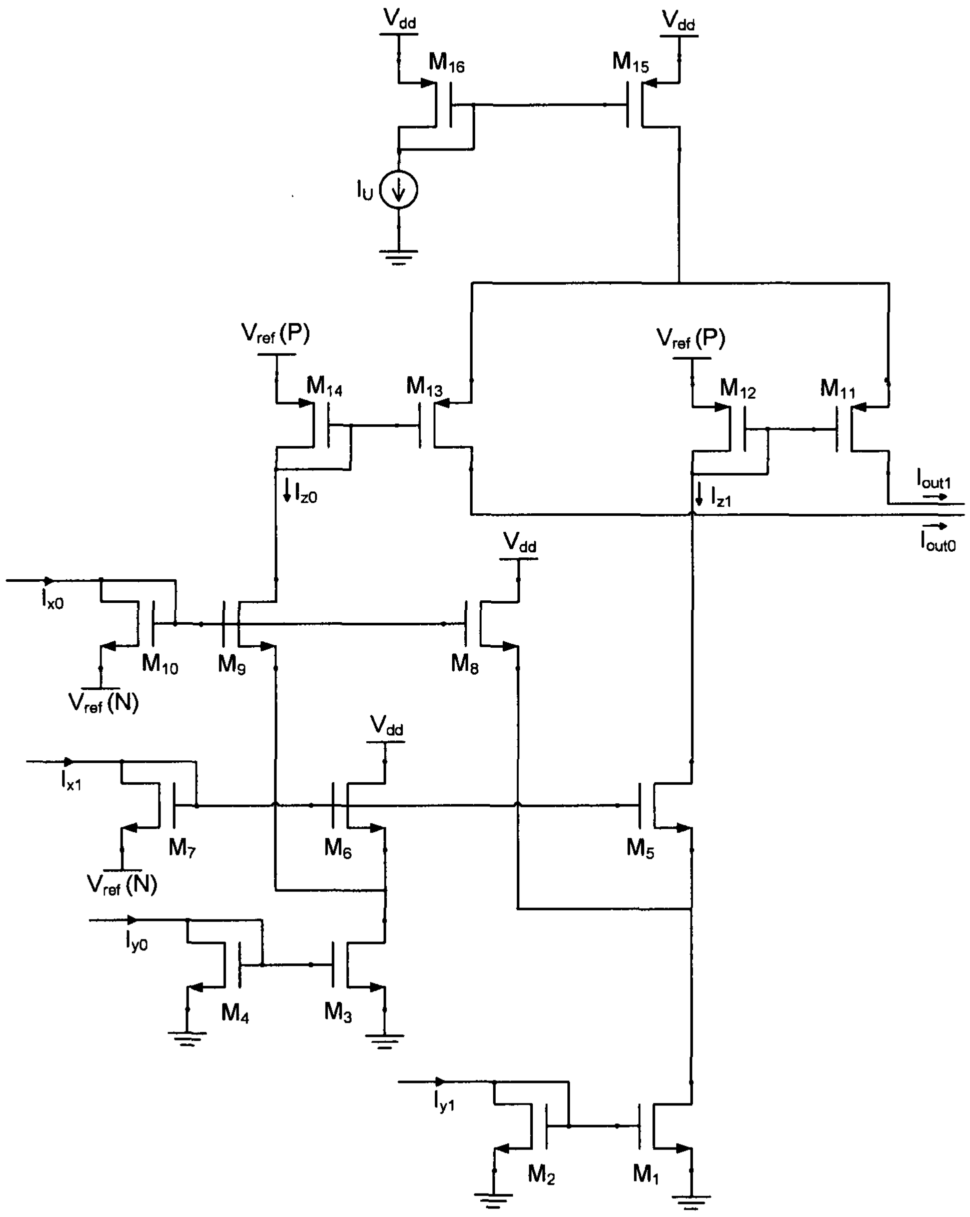

Fig 3.3.1: Canonical sum-product circuit (Equality node)

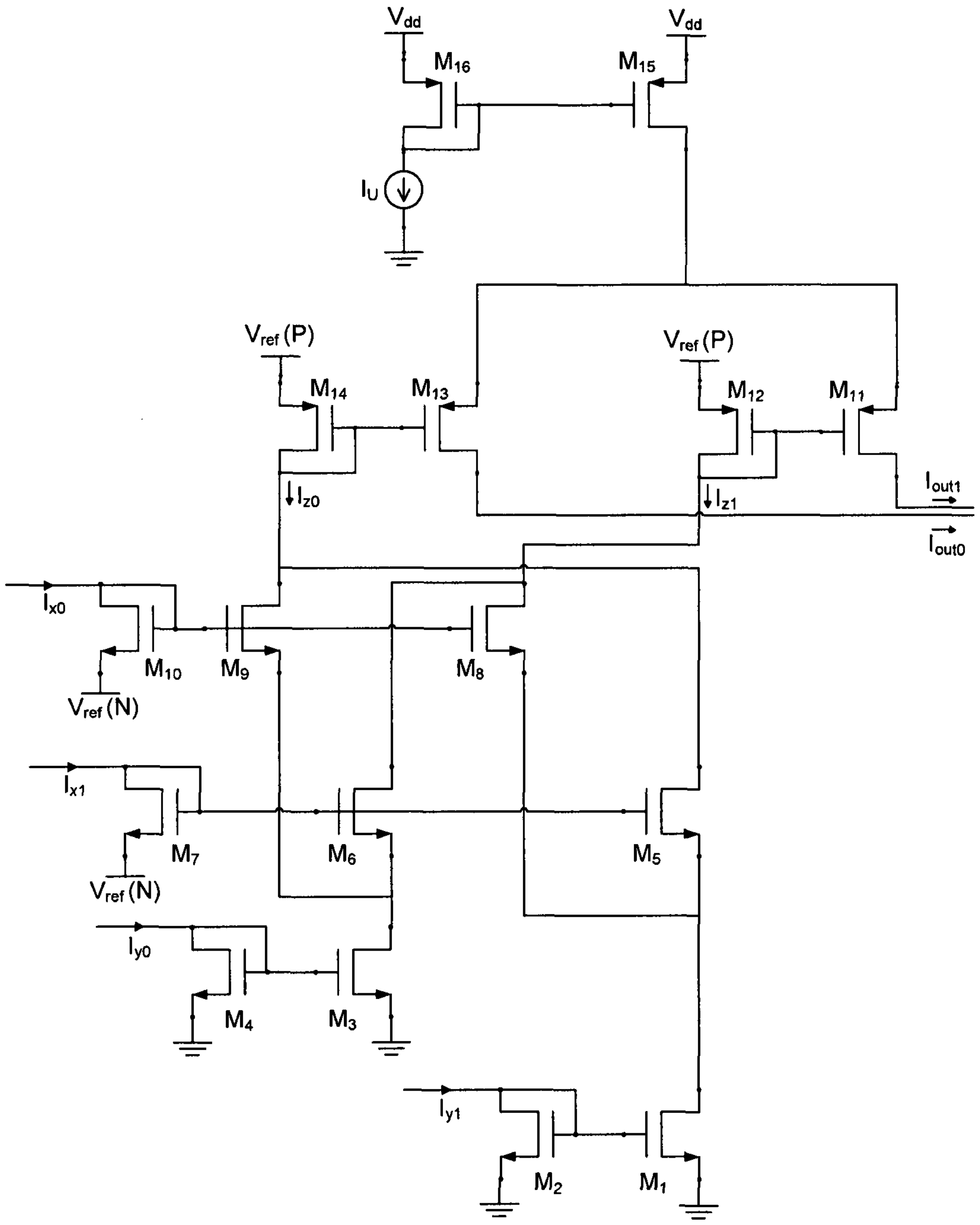

Fig 3.3.2: Canonical sum-product circuit (Check node)

As a result by combining the equations (3.3.1) to (3.3.6) and assuming that $I_{x0}+I_{x1}=I_{y0}+I_{y1}=I_U$, the renormalized equations for equality and check nodes are found as:

$$I_{out0(eq\_node)} = \frac{I_{x0}I_{y0}}{I_{x0}I_{y0} + I_{x1}I_{y1}} I_U \quad (3.3.7)$$

$$I_{out1(eq\_node)} = \frac{I_{x1}I_{y1}}{I_{x0}I_{y0} + I_{x1}I_{y1}} I_U \quad (3.3.8)$$

And,

$$I_{out0(ch\_node)} = \frac{I_{x0}I_{y0} + I_{x1}I_{y1}}{I_U} \quad (3.3.9)$$

$$I_{out1(ch\_node)} = \frac{I_{x1}I_{y0} + I_{x0}I_{y1}}{I_U} \quad (3.3.10)$$

As discussed earlier in order to have the above formulas, we must first make sure that MOS transistors in the translinear circuits are in the weak inversion region. Also for the canonical sum-product circuits all the transistors are assumed to be in saturation [57]. For this reason *V*ref (N) and *V*ref (P) are used to provide a high enough voltage at drain of M1 and a low enough voltage at drain of M15 respectively. By adjusting these two voltage sources, M1 and M15 will be kept in saturation.

Winstead [57] showed in his thesis that the minimum bias voltage required for the equality or check node to function properly can be approximately found through the following formula for the 0.18 μm technology:

$$V_{dd} \geq 0.42V + Vref + V_{TOP} + \frac{U_T}{\kappa} \ln(\frac{I_U}{100nA}) \quad (3.3.11)$$

where *V*ref =*V*dd - *V*ref (P), $U_T$ is the thermal voltage, and κ and $V_{TOP}$ are process dependent parameters. In deriving above the equation it has been assumed that to maintain saturation the $V_{ds}$ of every transistor has to be greater than $4U_T$.

## 3.4 Low-voltage sum-product circuits

Fig. 3.4.1 shows the low-voltage topology of the canonical sum-product circuits which allows *V*ref (N) and *V*ref (P) to be zero and Vdd respectively. The required supply voltage in this topology is lower than that if a canonical circuit as presented by Winstead in his thesis [57]. Eliminating the reference voltages causes M1 and M15 to become unsaturated. Therefore translinear equations for the modified topology would include extra parameters due to the reverse currents of M1 and M15. As an example for the low voltage equality node we will have:

$$I_{z0(eq\_node)} = \frac{I_{x0}I_{y0}}{I_{x0}+I_{x1}+I_{y0}} \tag{3.4.1}$$

$$I_{z1(eq\_node)} = \frac{I_{x1}I_{y1}}{I_{x0}+I_{x1}+I_{y1}} \tag{3.4.2}$$

Since in (3.4.1) and (3.4.2) the denominator is no longer the unit current, dummy transistors, M3 and M4 are added to the circuit so that the normalized equations are maintained.

$$I_{z0(eq\_node)} = \frac{I_{x0}I_{y0}}{I_{x0}+I_{x1}+I_{y0}+I_{y1}} = \frac{I_{x0}I_{y0}}{2I_U} \tag{3.4.1}$$

$$I_{z1(eq\_node)} = \frac{I_{x1}I_{y1}}{I_{x0}+I_{x1}+I_{y1}+I_{y0}} = \frac{I_{x1}I_{y1}}{2I_U} \tag{3.4.2}$$

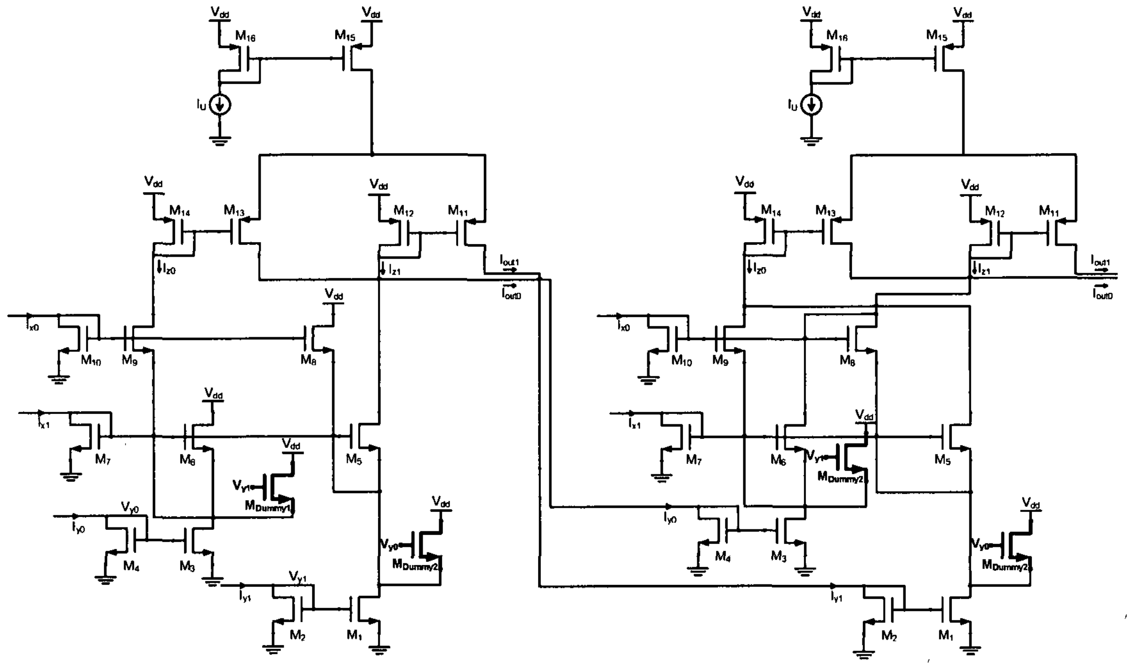
Fig 3.4.1: Low-voltage equality and check nodes including the dummy transistors

Low-voltage sum-product topology will be employed in the circuits of Chapter 4.

## 3.5 Design procedure of the sum-product circuits

Here we are going to explain the design procedure of the canonical sum-product circuits, equality and check nodes in the 90 nm CMOS technology.

For any differential pair to function symmetrically it seems reasonable to have identical loads at both sides. From Fig 3.3.1 it is clear that M6 and M8 are directly connected to Vdd while their corresponding differential pairs M9 and M5 are connected to *V*ref (P) through diode-connected PMOS transistors and thus the equality node is not exactly symmetrical. By adding pull-up diode-connected PMOS loads, $M_{Up1}$ and $M_{Up2}$ as shown in Fig 3.5.1 and by connecting their sources to *V*ref (P) instead of Vdd, it is expected to have symmetrical results for probabilities of one and zero. Four pairs of current-mirror circuits are distinguishable at the inputs of both equality and check nodes. The input diode-connected transistors are used to mirror the input probabilities which are in terms of currents to the circuit.

The first thing which might come to mind in the design of such circuits is that for an accurate transmission of the input data to the decoder core, current-mirror circuits must function satisfactorily. This means that the mirrored current has to be very close to the input current and this involves a sufficient $V_{ds}$ for M1 which is provided by $V_{ref}$ (N) to be around 200 mV as well as quite large $W$'s and $L$'s to avoid mismatch and the effect of $r_o$. However, the former will not be the case for the low-voltage topology as M1 would be unsaturated. On the other hand, since in an analog decoder it is the ratio of the probabilities of zero and one that matters and knowing that the renormalization circuits calibrate the output probabilities with respect to the unit current, exact mirroring at the inputs will not be that important. Nevertheless the effect of mismatch on the output currents should not be underestimated and needs to be studied.

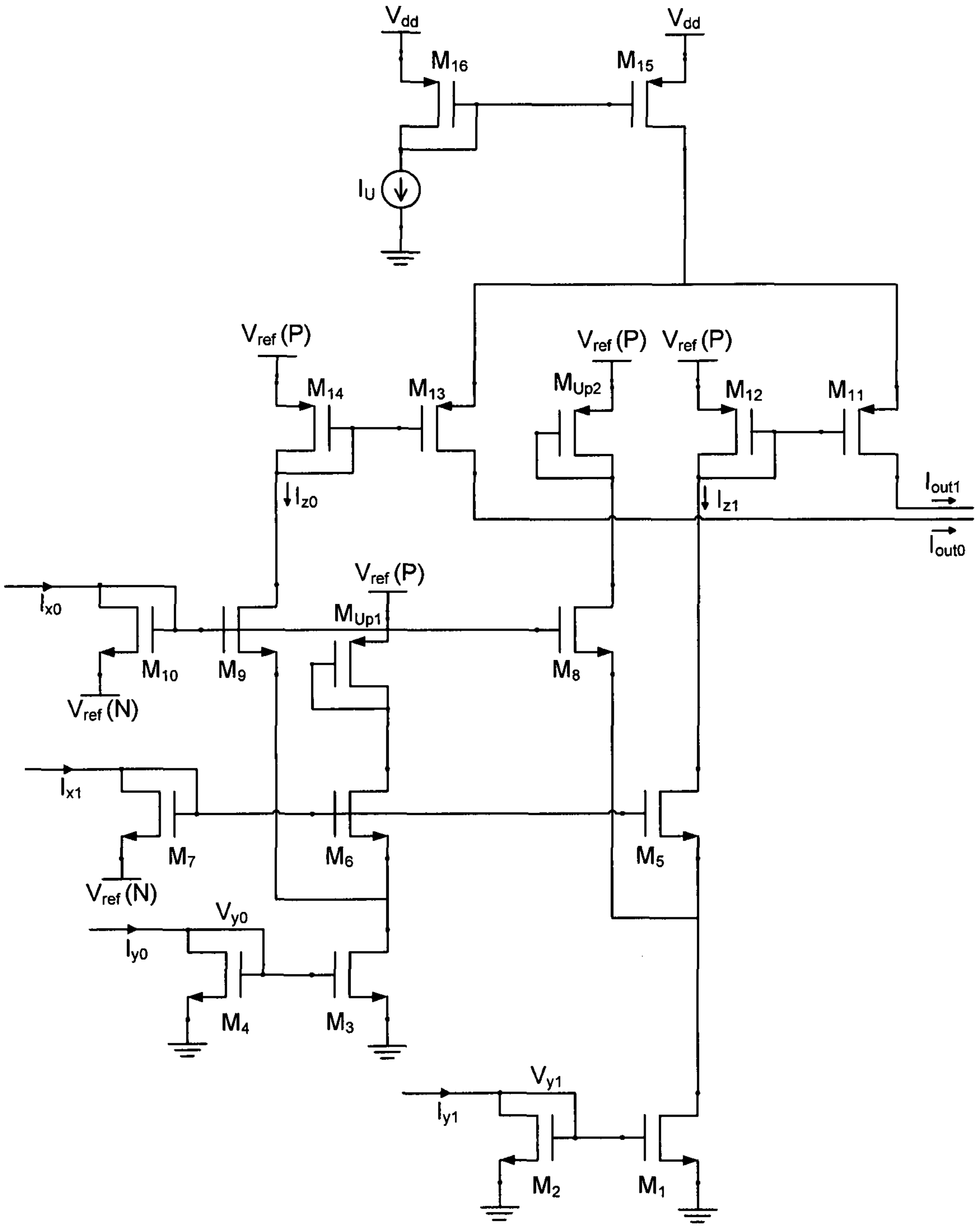

Fig 3.5.1: Symmetrical equality node by adding diode-connected PMOS loads

## 3.5.1 Range of input and output currents

The range of variations for the input currents must be determined such that all transistors remain in weak inversion region for a complete decoding. For this reason a current-mirror circuit has been simulated in 90 nm technology with arbitrary values for $W$ and $L$ ($W$ = 1.2 µm and $L$ = 1 µm) having enough $V_{ds}$ for the second transistor to stay saturated by varying the input current (here $V_{ds}$ = 200 mV). This model has been shown in Fig 3.5.2 and the result of simulation has been presented in Fig 3.5.3.

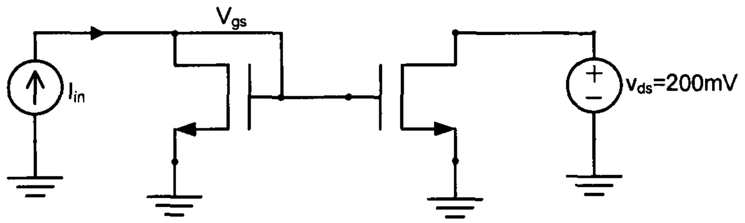

Fig 3.5.2: Current-mirror circuit used to determine the range of input current

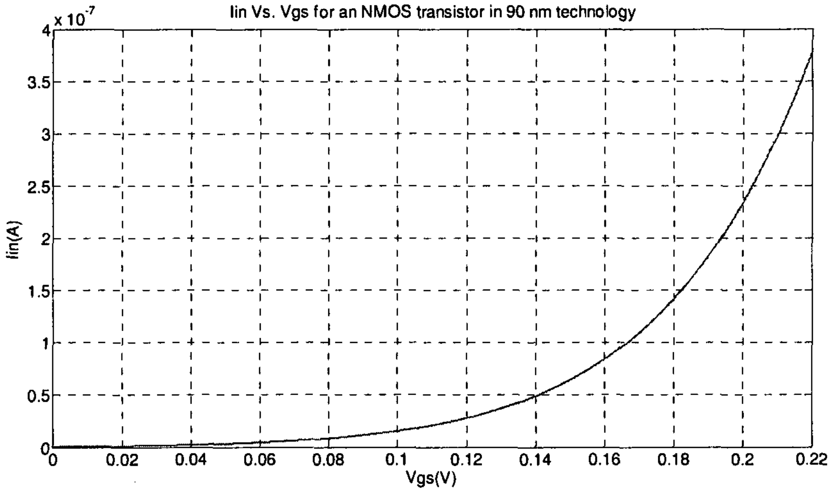

Fig 3.5.3: Plot of $I_{in}$ versus $V_{gs}$ for an NMOS diode-connected transistor in 90 nm technology with $W$ = 1.2 µm and $L$ = 1 µm of a current-mirror circuit where $V_{ds}$ for the second transistor has been set to 200 mV

It was found that the current-mirror circuit will be kept in the sub-threshold region where the gate source voltage is far below the nominal threshold voltage (*$V_{TON}$ = 200 mV*) while the input current changes from 0 to 100 nA. Recall that in the deep weak-inversion region we have $\frac{g_m}{I_D} = \frac{1}{nU_T}$. This can be verified in Fig 3.4.3 that in the deep weak inversion region $\frac{g_m}{I_D} = 32.7 = \frac{1}{nU_T}$, where typical value of $n$ = 1.2 and $U_T$ = 25 mV. However for input current close to 100 nA the transistor may no longer be operating in deep weak inversion region and thus the above equation will be just an approximation in moderate inversion region.

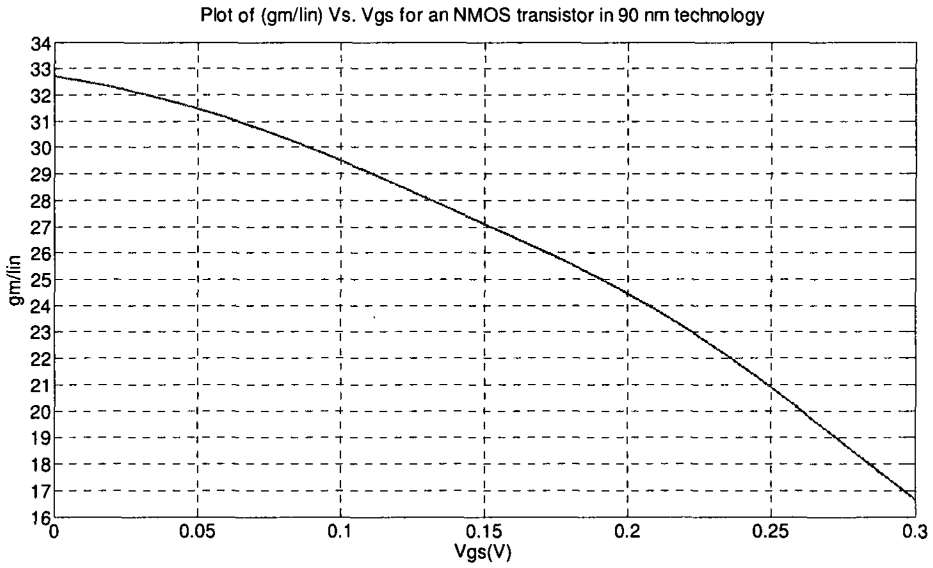

Fig 3.5.4: Plot of $(g_m/I_{in})$ versus $V_{gs}$ for the diode-connected NMOS transistor with $W$ = 1.2 μm and $L$ = 1 μm in 90 nm technology

The conclusion that one can draw from Fig 3.5.3 and Fig 3.5.4 is that 100 nA would be appropriate to keep an NMOS diode-connected transistor with $W$ = 1.2 μm in weak inversion. Since the current is directly proportional to the size of transistor $(\frac{W}{L})$, therefore the transistor will still remain in weak-inversion if its current and $W$ are doubled simultaneously. Assuming that in our discussion $I_U = 100$ nA represents the probability of 1, consequently 0 A would represent the probability of 0.

From the decoding point of view, the output of an equality node would be 1 if and only if both of its inputs are 1 while the output of a check node is 1 if and only if one of its inputs is 1 and the other is 0. Now if both inputs are having the 0.5 probability, so does the output of the equality and check node. If one of the inputs is 0 and the other is 1, we expect the equality node to have a similar behavior as 0.5 probabilities, since in the latter the equality node is unable to determine whether the result would be zero or one. The latter case is known as when the inputs are going to opposite extremes. However, this, as will be seen from simulations later, causes inaccuracies in the result of the equality node so that the output will not be exactly 0.5.

### 3.5.2 Adjusting the reference voltages

As explained earlier, in the canonical circuits of equality and check nodes there are two reference voltages which keep M1 and M15 in saturation. It is reasonable to set $V_{\text{ref}}$ (N) and $V_{\text{ref}}$ (P) for the uniform probability mass (i.e. 0.5 input and output probabilities) since this is the typical initialization point for every decoder before the real data has been received, although other initialization points may be used as will be discussed in Chapter 5.

Having $I_U = 100$ nA, the input current are set to 50 nA which must result in the same 50 nA current at the outputs. From simulations, $V_{ref}$ (N) = 120 mV has been found for a bias voltage of $V_{dd} = 600$ mV which satisfies (3.1.11) and brings M1 in saturation. M15 and M16 on the other hand has to be sized so that its current remains constant and equal to the unit current all the time. Therefore to avoid the undesired impact of variations in the $V_{ds}$ of M15 on the unit current, firstly its length ($L$) has been chosen larger than that of the other transistors and secondly $V_{ref}$ (P) is adjusted so that the output currents are both equal to 50 nA. Simulations showed that this voltage is different for equality and check nodes of the same size.

### 3.5.3 Accuracy of results at the extremes

For an equality node as discussed earlier as the inputs approach the opposite extremes (i.e. one input goes to zero while the other reaches $I_U$), one would find inaccurate results due to imbalance branches in the equality node. Assume that $I_{x0} = 0$ and $I_{y0} = 100$ nA. This leads to $I_{x1} = 100$ nA and $I_{y1} = 0$. It may be deduced that based on the translinear principal $I_{z0} = I_{z1} = 0.5$, however based on the equality node of Fig 3.1.1 or 3.5.1, the paths which lead to generating $I_{z0}$ and $I_{z1}$ are not identical and therefore the output currents $I_{out0}$ and $I_{out1}$ will not be exactly 50 nA as it is expected.

If $I_{x0}$ and $I_{y0}$ are separately varied from 0 to $I_U = 100$ nA, then the output probability of one ($I_{out1}$) would be derived from the simulation as shown in Fig 3.5.5. The equality node has been designed to have an accurate result when $I_{x0} = I_{y0} = 50$ nA which results in $I_{out0} = I_{out1} = 50$ nA as labeled on the figure. The other four values are for cases when the input currents reach the two extremes, 0 and 100 nA. It can be seen that the outputs are very close to 0 and 100 nA as expected while inputs are 0 or 100 nA respectively, however whenever the inputs are approaching the opposite extremes, the expected output of 50 nA cannot be obtained accurately. Although this is not a serious problem in decoders, yet we refer this issue to the inaccuracy at the extremes.

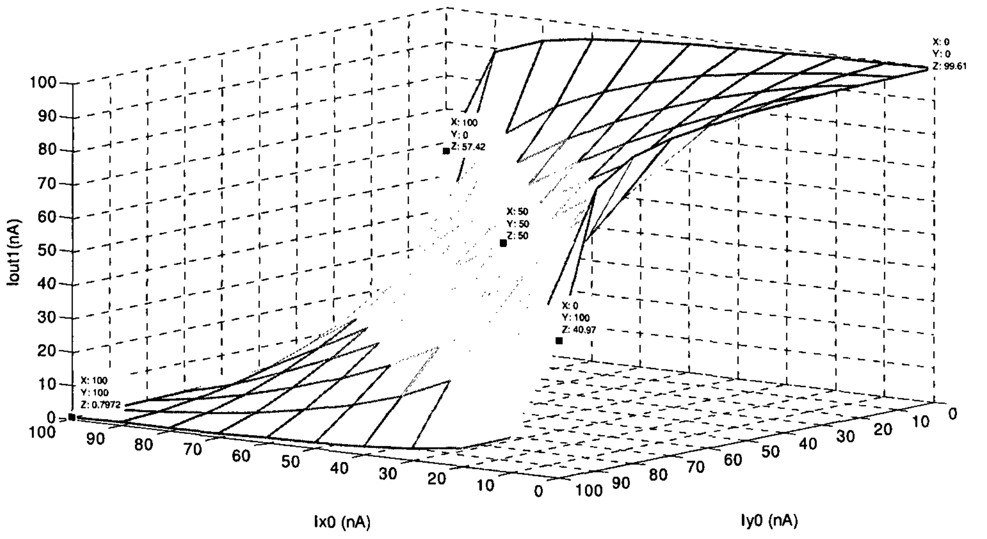

Fig 3.5.5: Results of simulation for the output probability of 1 in the equality node when the input probabilities of 0 are varied from 0 to 100 nA

## 3.5.4 The effect of Mismatch

One of the important factors in analog designs which affects the behavior of the circuit is the device mismatch. Ideally, we assume that transistors of equal length and width exhibit the same properties, while practical situations tell another story. Mismatch is caused by process variation. One kind of mismatch is difference in length and width of the transistor which are produced during the fabrication process.

The other sort of mismatch is the difference in threshold voltage caused by the random variations in the doping level of the channel and gate [69]. It has been shown that the most effect of the device mismatch in weak inversion operation mode region comes from threshold voltage mismatch [65]. This results in the current matching error.

The device mismatches can be considered as a random variable with normal distribution having zero mean and the variance which depends on the size of the transistor.

It has been shown [65] that the variance of threshold voltage mismatch is found by,

$$\sigma^2(\Delta V_T) = \frac{A_{VT}^2}{W \times L} \tag{3.5.1}$$

where $A_{VT}$ is the technology dependant matching parameter and it is about 5 (mV×μm) for 90 nm technology.

Small variations of the threshold voltage would cause the current to change slightly since:

$$\Delta I_D = g_m \Delta V_T \tag{3.5.4}$$

Dividing both sides of (3.5.1) by drain current we have:

$$\frac{\Delta I_D}{I_D} = \frac{g_m}{I_D} \Delta V_T \tag{3.5.3}$$

Equations (3.5.1) and (3.5.3) suggest that by increasing *W* and *L* of the transistors for a specific drain current, one can lower the current variation and thereby reduce the effect of mismatch on the output result.

To see the effect of mismatch on the accuracy of the output currents, Monte Carlo analysis has been done in the equality node for 100 iterations of process variation and mismatch. *W* and *L* for all transistors except M15 and M16 are assumed to be 1.2 μm and

1 μm respectively. M15 and M16 both have $\frac{W}{L}=\frac{2}{2}$. Ideally if we set Iy0=Iy1=50 nA, by increasing $I_{x0}$ from 0 to 100 nA which causes $I_{x1}$ to drop from 100 nA to 0, the output probability of 1 is expected to change as in Fig 3.5.5, however due to the mismatch it is found as shown in Fig 3.5.6.

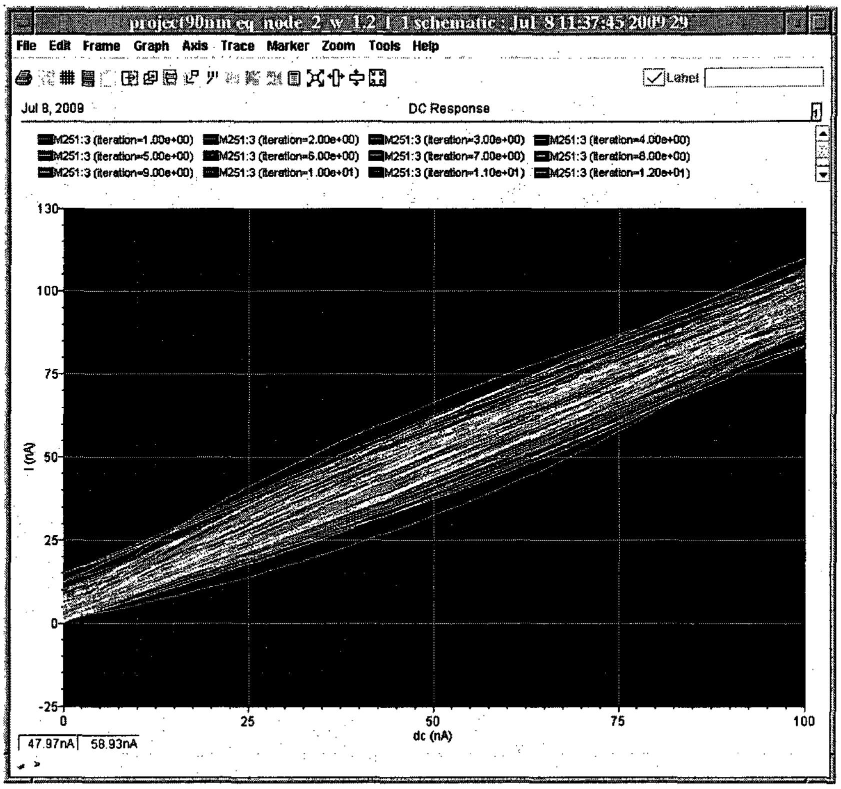

Fig 3.5.6: Simulation results for 100 iterations of process variation and mismatch for the equality node with $\frac{W}{L}=\frac{1.2}{1}$

If the widths and lengths of all transistors (expect M15 and M16) are doubled, a better result would be anticipated. As seen from the new simulation results of Fig 3.5.7, it is obvious that the output currents of the equality node are more condensed and therefore the effect of mismatch has been reduced to some extent.

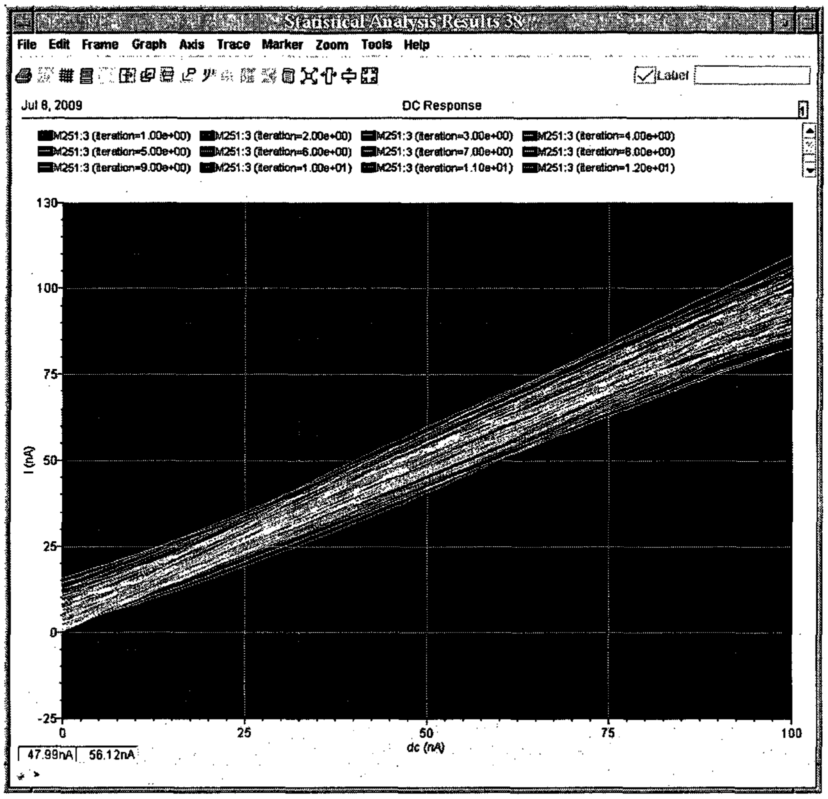

Fig 3.5.7: Simulation results for 100 iterations of process variation and mismatch

for the equality node with $\frac{W}{L}=\frac{2.4}{2}$

## 3.6 Structure of the Chip

In this section we are going to explain the structure of our chip. The main goal to implement the sum-product circuits is to measure the accuracy of results as well as the effect of mismatch and to compare them with the results of simulation. For this reason 64 equality and check nodes with different configurations and sizes have been implemented. The chip also contains voltage-to-current converters, input and output switches and a digital circuit which will be discussed.

### 3.6.1 Blocks

In this chip there are 64 equality and check nodes with different sizing and configuration. These nodes are arranged in groups of 8 similar blocks each containing 8 nodes and is depicted in Fig 3.6.1. The first 4 nodes in each block are the equality nodes and the rest are the check nodes. The arrangement of each block is as follows:

*1st node*: 2-input equality node with small W's and L's (1.2 μm and 1 μm respectively)

*2nd node*: 2-input equality node with doubled W's and L's (2.4 μm and 2 μm respectively)

*3nd node*: 3-input equality node with small W's and L's (1.2 μm and 1 μm respectively)

*4th node*: 3-input equality node with doubled W's and L's (2.4 μm and 2 μm respectively)

*5th node*: 2-input check node with small W's and L's (1.2 μm and 1 μm respectively)

*6th node*: 2-input check node with doubled W's and L's (2.4 μm and 2 μm respectively)

*7nd node*: 3-input check node with small W's and L's (1.2 μm and 1 μm respectively)

*8th node*: 3-input check node with doubled W's and L's (2.4 μm and 2 μm respectively)

Note that 3-input nodes are composed by cascading two stages of 2-input nodes. Input and output switches will select one of these nodes to be measured at a time.

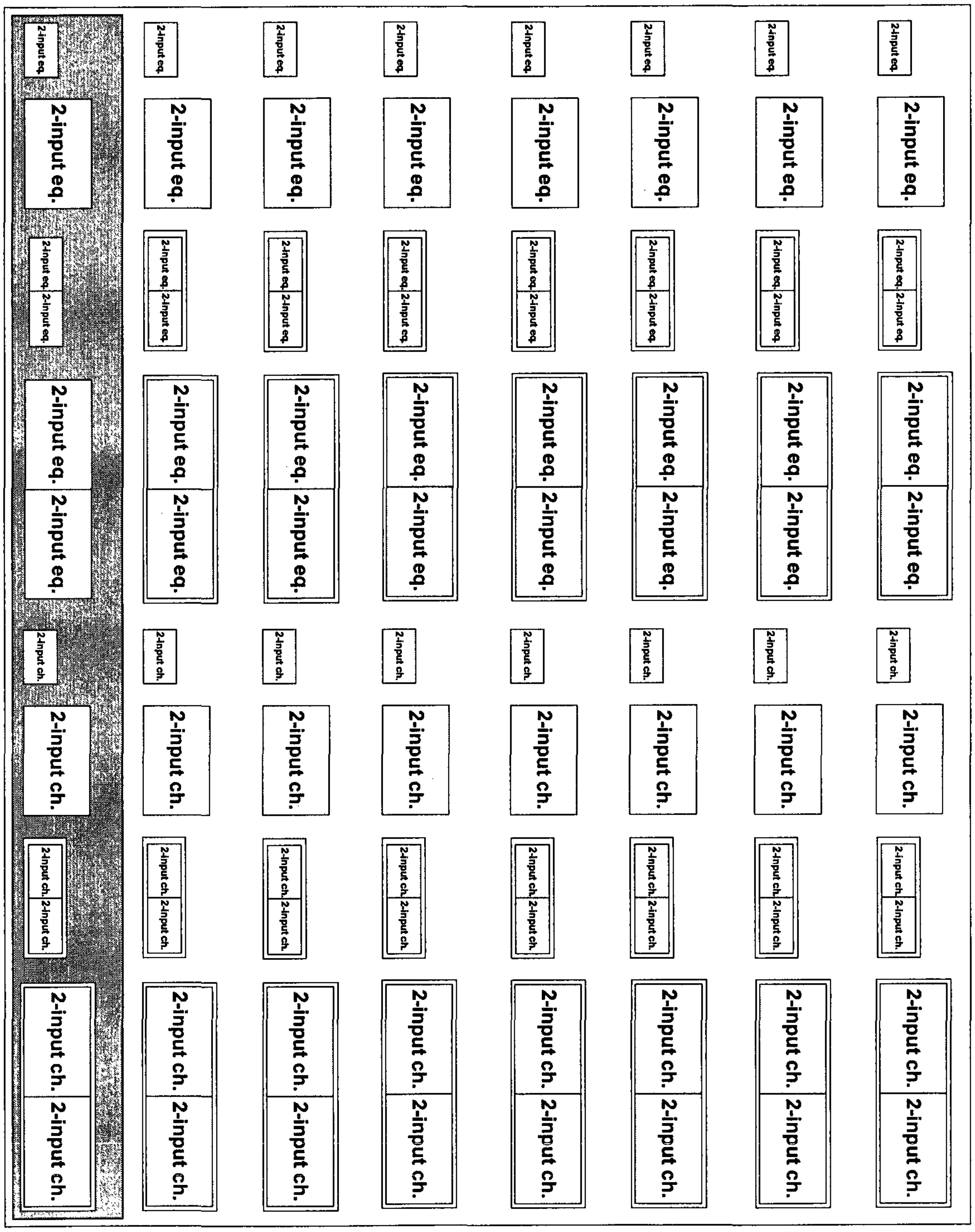

Fig 3.6.1: Diagram of 8 similar blocks each containing 8 equality and check nodes with different topologies and sizes. Shaded area in a column indicates one of the 8 blocks

### 3.6.2 Voltage-to-Current converters

Sum-product circuits are current-mode devices whose inputs and outputs are in current format. However, in a digital receiver the received data which has been corrupted by noise of the channel has analog nature in terms of voltage. Hence there must be a converter prior to the decoder to convert the received voltage mode signals to current mode probabilities for equality and check nodes.

Since the minimum and maximum input current of the sum-product circuit cannot be less than zero and greater than *Iu*, therefore the voltage-to-current (V2I) converters must be designed for which the output is between 0 and *Iu*. The converter circuit is in fact a differential pair as shown in Fig 3.6.1 where the transistors must operate in the sub-threshold region so that the following relationship between the input voltages and output currents is established.

$$I_0 = e^{(Vin/2)} \tag{3.6.1}$$

$$I_1 = e^{-(Vin/2)} \tag{3.6.2}$$

If we take Napierian logarithm from both sides of the above equations we get

$$\ln(\frac{I_1}{I_0}) = \ln(e^{(Vin/2+Vin/2)}) = \ln(e^{Vin}) = Vin \tag{3.6.3}$$

Equation (3.6.3) is referred to the duality between the current-mode probability domain and the voltage-mode log-likelihood ratio [57]. Having $I_1 + I_0 = I_U$, (3.6.3) can be rewritten as:

$$I_0 - I_1 = I_U \tanh(vin/2) \tag{3.6.4}$$

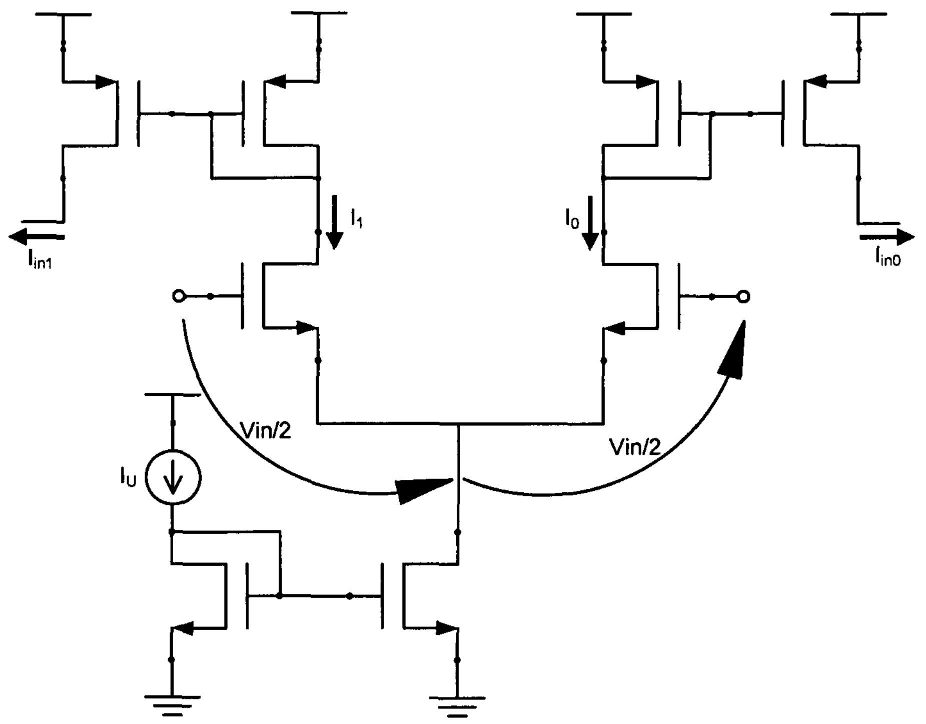

Fig 3.6.2: A differential pair served as the voltage to current converter at the input

Since we have a combination of two-input and three-input equality and check nodes in the chip, so a maximum of three voltage-to-current converters are required to provide all the input currents. Selecting one of the 64 nodes at a time will be done through input and output switches.

### 3.6.3 Input and Output Switches

PMOS Pass-Transistor-Logic (PTL) transistors are employed at the input and output to select between the nodes. Gates of these transistors are fed by the address corresponding to each node. Fig 3.6.3 shows how a single node is going to be selected through input switches. Note that this figure is showing the selecting procedure for one of the inputs which belongs to one of the nodes out of 64 nodes in the chip. Since all of these nodes require at least two input currents, therefore the structure of two of the input switches are as Fig 3.6.3. However in order to provide the third inputs for thirty two 3-input nodes, we need a total of 32 switches which are arranged in two rows of four switches after the third voltage to current converter.

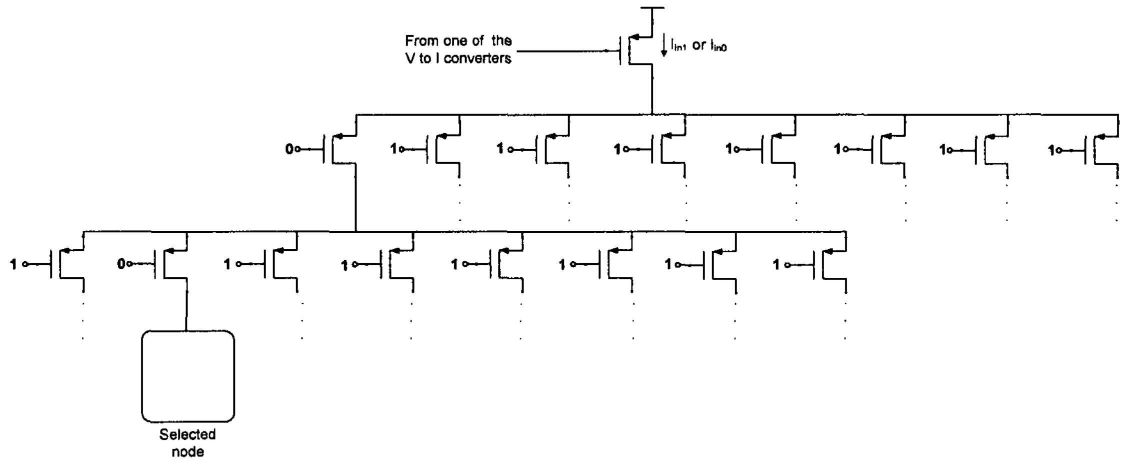

Fig 3.6.3: Configuration of one set of input switches used to feed one of the inputs (probability of one or zero) which belongs to one of the 64 nodes in the chip

Similar switches but with an inverse configuration as shown in Fig 3.6.4 must be used after the nodes to read the output currents coming from individual nodes.

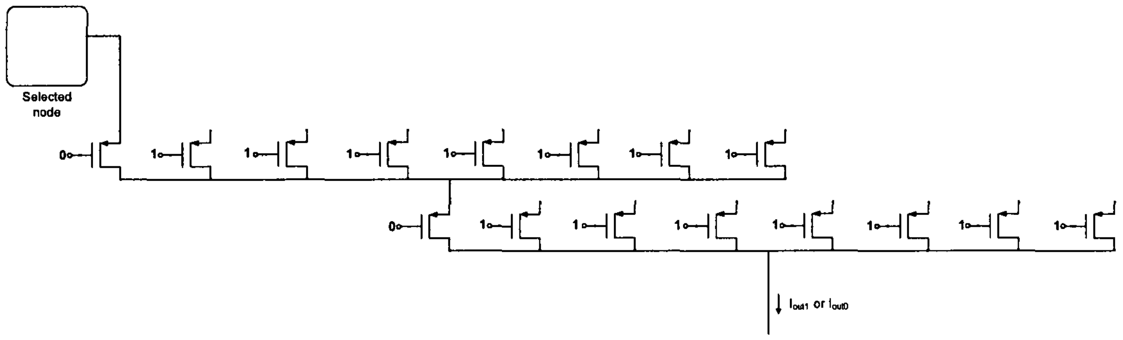

Fig 3.6.4: Configuration of one set of output switches (probability of one or zero) coming from the output of one of the 64 nodes in the chip

### 3.6.4 Digital circuit

Digital input data is delivered to a digital circuit comprised of flip-flops and shift-register circuits on the rising edges of clock. Output of the digital part is the address to the switches which selects the nodes based on Table 3.6.1.

Table 3.6.1: Input data and corresponding selected nodes

| Block | Data | Selected node |
|---|---|---|
| 1 | 000000 | nothing(For Calibration Only) |
| | 000001 | 2-input equality node with doubled W's and L's |
| | 000010 | 3-input equality node with small W's and L's |
| | 000011 | 3-input equality node with doubled W's and L's |
| | 000100 | 2-inputs check node with small W's and L's |
| | 000101 | 2-inputs check node with doubled W's and L's |
| | 000110 | 3-inputs check node with small W's and L's |
| | 000111 | 3-inputs check node with doubled W's and L's |
| 2 | 001000 | 2-input equality node with small W's and L's |
| | 001001 | 2-input equality node with doubled W's and L's |
| | 001010 | 3-input equality node with small W's and L's |
| | 001011 | 3-input equality node with doubled W's and L's |
| | 001100 | 2-inputs check node with small W's and L's |
| | 001101 | 2-inputs check node with doubled W's and L's |
| | 001110 | 3-inputs check node with small W's and L's |
| | 001111 | 3-inputs check node with doubled W's and L's |
| 3 | 010000 | 2-input equality node with small W's and L's |
| | 010001 | 2-input equality node with doubled W's and L's |
| | 010010 | 3-input equality node with small W's and L's |
| | 010011 | 3-input equality node with doubled W's and L's |
| | 010100 | 2-inputs check node with small W's and L's |
| | 010101 | 2-inputs check node with doubled W's and L's |
| | 010110 | 3-inputs check node with small W's and L's |
| | 010111 | 3-inputs check node with doubled W's and L's |
| 4 | 011000 | 2-input equality node with small W's and L's |
| | 011001 | 2-input equality node with doubled W's and L's |
| | 011010 | 3-input equality node with small W's and L's |
| | 011011 | 3-input equality node with doubled W's and L's |
| | 011100 | 2-inputs check node with small W's and L's |
| | 011101 | 2-inputs check node with doubled W's and L's |
| | 011110 | 3-inputs check node with small W's and L's |
| | 011111 | 3-inputs check node with doubled W's and L's |
| | 100000 | 2-input equality node with small W's and L's |

| | | |
|---|---|---|
| 5 | 100001 | 2-input equality node with doubled W's and L's |
| | 100010 | 3-input equality node with small W's and L's |
| | 100011 | 3-input equality node with doubled W's and L's |
| | 100100 | 2-inputs check node with small W's and L's |
| | 100101 | 2-inputs check node with doubled W's and L's |
| | 100110 | 3-inputs check node with small W's and L's |
| | 100111 | 3-inputs check node with doubled W's and L's |
| 6 | 101000 | 2-input equality node with small W's and L's |
| | 101001 | 2-input equality node with doubled W's and L's |
| | 101010 | 3-input equality node with small W's and L's |
| | 101011 | 3-input equality node with doubled W's and L's |
| | 101100 | 2-inputs check node with small W's and L's |
| | 101101 | 2-inputs check node with doubled W's and L's |
| | 101110 | 3-inputs check node with small W's and L's |
| | 101111 | 3-inputs check node with doubled W's and L's |
| 7 | 110000 | 2-input equality node with small W's and L's |
| | 110001 | 2-input equality node with doubled W's and L's |
| | 110010 | 3-input equality node with small W's and L's |
| | 110011 | 3-input equality node with doubled W's and L's |
| | 110100 | 2-inputs check node with small W's and L's |
| | 110101 | 2-inputs check node with doubled W's and L's |
| | 110110 | 3-inputs check node with small W's and L's |
| | 110111 | 3-inputs check node with doubled W's and L's |
| 8 | 111000 | 2-input equality node with small W's and L's |
| | 111001 | 2-input equality node with doubled W's and L's |
| | 111010 | 3-input equality node with small W's and L's |
| | 111011 | 3-input equality node with doubled W's and L's |
| | 111100 | 2-inputs check node with small W's and L's |
| | 111101 | 2-inputs check node with doubled W's and L's |
| | 111110 | 3-inputs check node with small W's and L's |
| | 111111 | 3-inputs check node with doubled W's and L's |

Table 3.6.1 takes into account 63 out of 64 nodes. It had been decided to measure the timing delay in the first node by applying a step input and observing how fast the output would change. However due to the difficulty in applying an accurate step current the speed measurement test was ignored.

Since the inputs to the chip are voltages and the output of every node is in the current-mode, therefore there is a need to measure the input and output currents. This chip is designed so that by selecting the all zero data (as shown in the table) the input current

(Output of the V2I converter) can be plotted with respect to the applied voltage. As a result the input currents are calibrated prior to the actual measurement. To measure the output current on the other hand, a relatively large resistance of about 10 K-ohm is required at the output to convert the current to voltage. Such a large resistance at the output is needed since the output current is expected to be in the range of 0 to 100 nA, therefore in order to be measured by a good voltmeter having sufficient input impedance, the corresponding voltage must be in a reasonable range. Although this may cause speed limiting issues due to a large dominant pole at the output, it is convenient for dc measurements. We will explain in Chapter 6 what simple measurements indicated that something is not right with the chip and so the measurement process fails.

# Chapter 4

# Application of Active Current Mirrors to Improve the Speed of Analog Decoders

A version of this chapter in published in Ref. [71]

This chapter will focus on the issue of speed reduction and delays due to the wirings in large codes based on the sum-product algorithm and particularly large LDPC codes. In previous chapters the structure of equality nodes and check nodes as the modules of the sum-product circuit were presented. It is observed that the current-mirror circuits used at the input of these nodes plays a key role in generating the delay in a circuit of analog decoder. For an analog decoder to function properly, transistors must operate in weak inversion region. Though this region of operation gives rise to low transconductance, and correspondingly high input resistance, the intrinsic speed of individual nodes is adequate, owing to the small capacitance of the input transistors. However, for large codes, as the number of nodes increases, wiring the modules that are millimeters apart generates large wiring capacitance which has significant effect on the throughput of an analog decoder.

Increasing the bias currents of all branches in a large analog decoder will proportionally increase its bandwidth. However, since both speed and power will increase equally, this approach fails to improve the power/speed ratio. Selectively doubling the currents as well as the sizes of only the input diode-connected transistors of a module as a second approach involves doubling the currents and sizes of that of the subsequent module for accurate decoding operation. Thus, increasing the speed of the decoder will be at the expense of large total power dissipation although an improved power/speed ratio would be achieved.

Current-mode design approaches provide a variety of useful features such as improving the bandwidth [66], [67]. An alternative approach to solve the issue of speed in large analog decoders is to replace the basic current-mirrors at the input nodes with active current mirrors and benefit from the current-mode technique to improve the power/speed ratio.

We are going to first present a large LDPC code and estimate the largest possible wiring capacitance for this code which corresponds to the worst case spacing between the modules of the related Tanner graph. The power/speed ratio or the power delay product for a block of equality node in this code will be simulated for three cases; with the basic mirrors, with the boosted (bias currents as well as sizes) input and output mirrors and with the active mirrors at the input. Consequently, it turns out that the final approach is more practical and more beneficial to be employed in analog decoder circuits.

## 4.1 A Large LDPC code

A (3, 6) LDPC code defined by a (512 x 1024) *H*-Matrix is considered for this example. This requires 1024 equality nodes and 512 check nodes having 3 and 6 edges connected to each node, respectively. Taking into account one extra edge for the received information, each equality node would have a total of 4 connected edges. If $d$ indicates the number of edges connected to a bidirectional node, then it can be verified that there must be 3($d$-2) two-input unidirectional nodes within each $d$-edges node [43]. As a result, 6 unidirectional equality nodes and 12 unidirectional check nodes are used in the blocks of equality and check nodes.

Due to some confusion with the correct $H$-Matrix, we initially did all the simulations on the transpose of the aforementioned code. That is, our $H$-Matrix was a (1024 x 512) for which the number of equality and check nodes are 512 and 1024, while the connected edges to every equality and check node are 6 and 3 respectively. By taking into account the one extra edge for the equality nodes, there will be a total of 7 edges connected to each equality node. However, later we came to this conclusion that the extracted LDPC code with the number of its equality nodes less than that of check nodes may not be practical.

It is worth mentioning though that the above confusion will not affect on the conclusion of this work which is the application of active current mirrors in improving the speed of analog decoders. So, for the employed $H$-Matrix there are 15 unidirectional equality nodes and 3 unidirectional check nodes in each block of equality and check nodes. This is shown in Fig 4.1.1. It is observed that in a block of equality nodes, there are 7 separate edges for the inputs as well as for the outputs and each output is generated by the other 6 inputs. In the block of check nodes there will be 3 input edges and 3 output edges and every output is produced by the other 2 inputs.

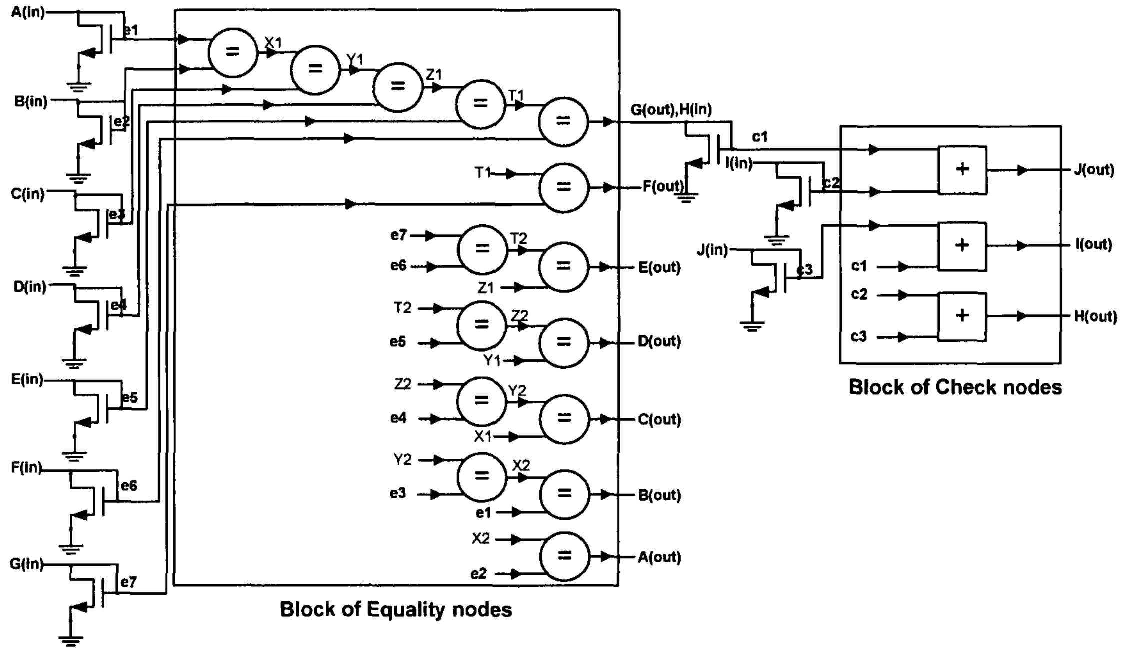

Fig 4.1.1: Blocks of equality and check node for a large LDPC code

and a sample connection between them

In Fig 4.1.1, the equal sign (=) denotes an equality node and addition sign (+) indicates a check node. Although the inputs and outputs of the block of equality and check nodes are currents, the internal signals (e.g., e1, X1, etc) are gate voltages generated by current mirrors. The circuits of unidirectional equality and check nodes are designed based on the low-voltage sum-product topology described in [57]. These nodes have current-mirror circuits at their inputs. A sample connection between two different nodes in equality and check nodes is also depicted in Fig 4.1.1.

The total power dissipation for a 7-input equality node was found to be $P_I$ = 0.9 μW with a supply voltage of 400 mV. Also the timing delay generated by each of the input mirrors was simulated as $D_{I\_noload}$ = 3.0 ns. This is the delay between a step input current applied to *A(in)* and the measured output of the corresponding mirror *e1* which crosses the 63 % of its final value.

For the complete LDPC decoder however, 512 blocks of equality nodes and 1024 blocks of check nodes would be needed. An estimate for the decoder's required area is shown in Fig 4.1.2. Perhaps, we could have used other floor plans which are more optimized in terms of area and spacing, however this block diagram was preferred for the sake of simplicity. In this Figure, each black box, represent a block of equality or check nodes which are repeated in rows and columns. Individual unidirectional equality and check nodes have the same dimension of 26 μm x 13 μm. Assuming 0.2 μm for the width of wires as well as for the spacing between them and noting that each single edge of Fig 4.1.1 is in fact two wires comprised of probabilities of one and zero, the longest wire which is required to connect the farthest equality and check nodes together can be approximated as:

$$L_{\max} = 2457 + 2080 + 1248 + (2496 - 2080)/2 \sim 6400\ \mu m$$

This corresponds to the worst case estimation for a wiring capacitance of about $C_w = 500 fF$ if only Metal 1 is being used. Note that by using higher levels of metal, one can have lower wiring capacitance as a result of having the capacitances in parallel. It was figured out that the added wiring capacitance due to 1 mm extension with Metal1 is 79.5 fF while this would be 62.5 fF and 56.15 fF with Metal 2 and 3 layers respectively.

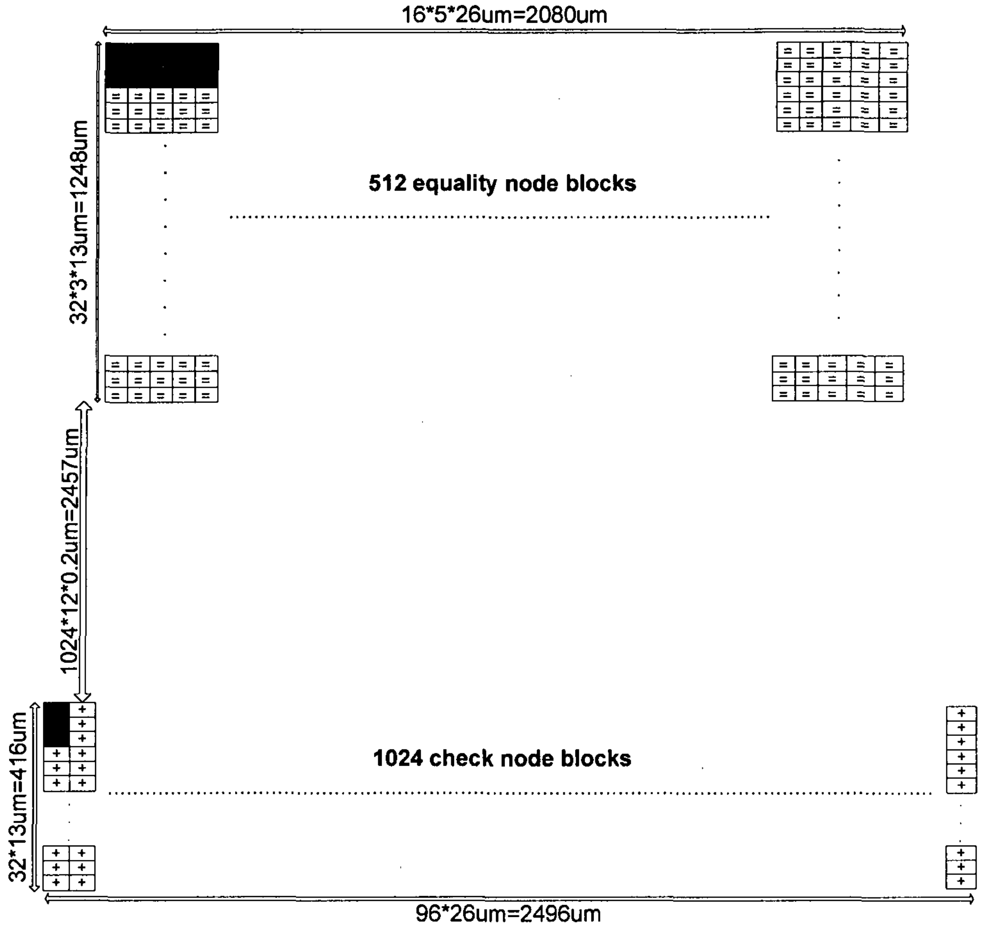

Fig 4.1.2: Approximate block diagram for a large LDPC decoder

### 4.1.1 The input resistance of a current-mirror circuit

A simple current mirror is shown in Fig 4.1.3. The input resistance of such a circuit can be found as:

$$R_{in} = (\frac{1}{g_m} \parallel r_o) \cong \frac{1}{g_m} \qquad (4.1.1)$$

where $g_m$ is the transconductance of M1 and $r_o$ is the drain-source resistance which can be neglected compared to $\frac{1}{g_m}$.

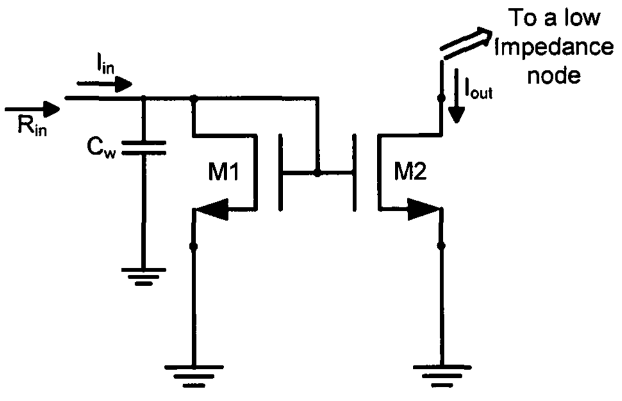

Fig 4.1.3: Basic current-mirror circuit used at the inputs of an equality or check node with the modeled wiring capacitance $C_w$

Ignoring the wiring capacitance, the total capacitance at the input node is:

$$C_T = C_{in} = C_{gs1} + C_{db1} + C_{gb1} + C_{gs2} + C_{gd2} + C_{gb2} \qquad (4.1.2)$$

Recalling that in analog decoder circuits based on the sum-product algorithm, all transistors must be operating in the weak-inversion or sub-threshold region. Therefore the current and consequently the transconductance $g_m$ would be small. As a result the dominant pole or the -3dB point will be determined from the input node:

$$f_{-3dB} = \frac{1}{2\pi R_{in} C_{in}} = \frac{g_m}{2\pi C_{in}} \qquad (4.1.3)$$

Therefore, the circuit will have a moderate intrinsic speed. However, by taking into account the existing wiring capacitance $C_w$, the bandwidth is limited significantly since:

$$f_{-3dB} = \frac{g_m}{2\pi C_T} = \frac{g_m}{2\pi(C_{in} + C_w)} \tag{4.1.4}$$

If the worst case capacitance of 500 fF is to be placed at the input node of the current-mirrors shown in Fig 4.1.1, the timing delay will be found as $D_{I\_maxload}$ = 279 ns for the input mirror which reveals a considerable extra delay due to wiring. Note that in our discussion we are assuming particular dimensions for the transistors (i.e. $W = 1.2$ μm and $L = 0.5$ μm).

Similar results appear from the small signal circuit of the basic mirror shown in Fig 4.1.4.

$$i_{out} = (g_{m1} - sC_{gd2})v_{gs} \tag{4.1.5}$$

$$v_{gs} = \frac{i_{in}}{g_{m1} + 1/r_{o1} + sC_T} \tag{4.1.6}$$

And so the transfer function is:

$$\frac{i_{out}}{i_{in}} = \frac{g_{m1} - sC_{gd2}}{g_{m1} + 1/r_{o1} + sC_T} \tag{4.1.7}$$

Based on (4.1.7), it turns out that the circuit has got one low-frequency pole and one high-frequency, right-half-plane zero. If we ignore the very small $1/r_{o1}$ comparing to the small $g_{m1}$ we get the same result as in (4.1.4) for the -3dB bandwidth of the basic current-mirror circuit.

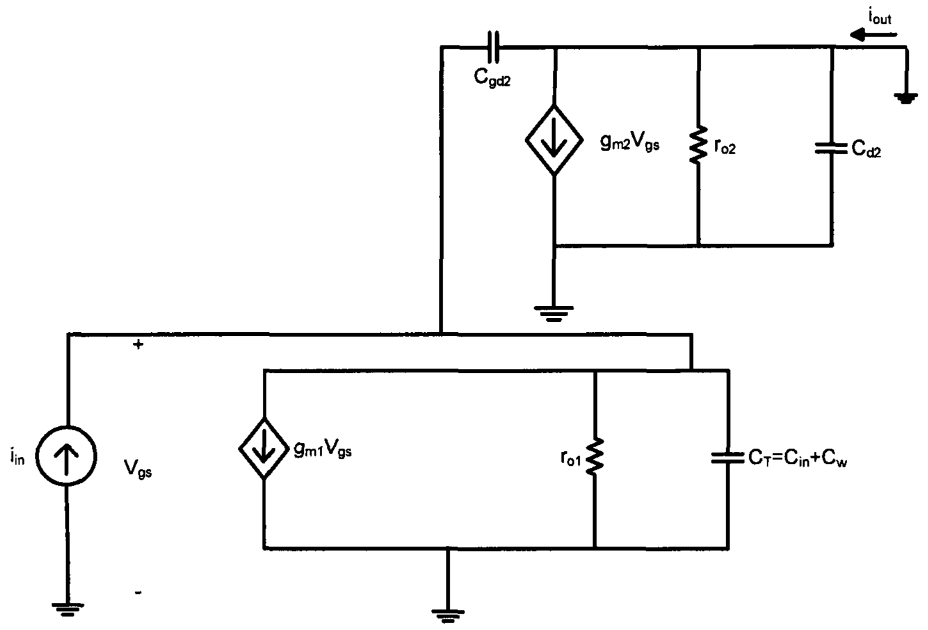

Fig 4.1.4: Small signal circuit of the basic current-mirror

The effect of wiring capacitance on the -3dB bandwidth for a simple current-mirror circuit is shown in Fig 4.1.5. Based on the above definitions, $\frac{C_w}{C_{in}}$ shows the ratio of the wiring capacitance with respect to the total input capacitance without the wiring. In this Figure, the worst case estimated capacitance of $C_w = 500$ fF and the corresponding speed of 483.4 KHz is also indicated. It should be mentioned that the achieved bandwidth without any wiring capacitance has been found to be around 53 MHz!

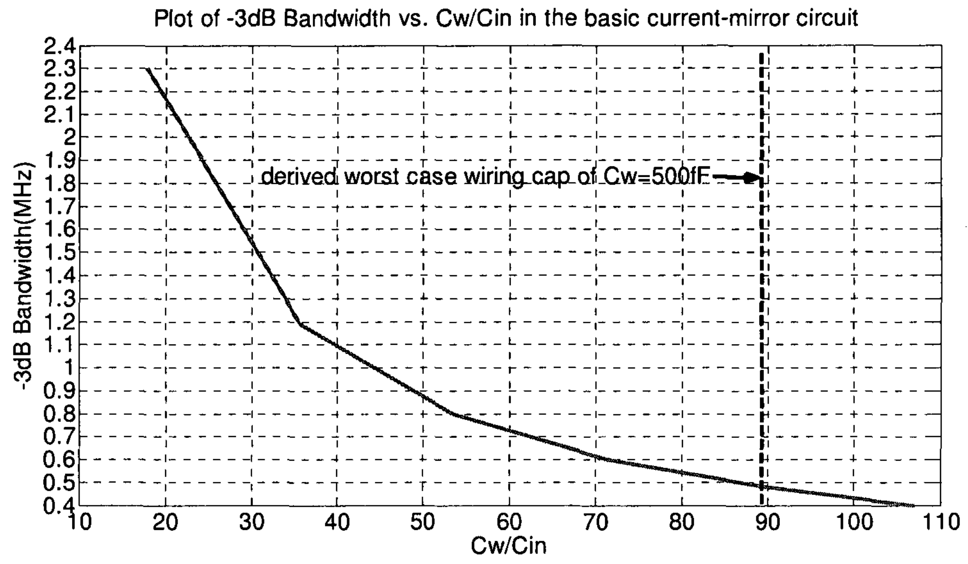

Fig 4.1.5: Plot of -3dB Bandwidth vs. $C_w / C_{in}$ for the basic

current-mirror circuit in the equality node

### 4.1.2 Total delay in a block of equality nodes

For the total delay in a block of equality nodes, Fig 4.1.1, several paths which connect the input and outputs together are considered. The longest path from *A(in)* to *G(out)* involves five equality nodes each generating delay whereas the shortest path from *F(in)* to *G(out)* consists of only one equality node. We further define the delay of the longest path with the basic current-mirrors as $D_{1\_longest}$ and the delay of the shortest path as $D_{1\_shortest.}$

Ignoring the effect of the wiring capacitance, it was found through simulations that $D_{1\_longest\_noload}$ = 79.48 ns and $D_{1\_shortest\_noload}$ = 14.35 ns. However, when the worst case estimated wiring capacitance is to be taken into account, these values will change to $D_{1\_longest\_maxload}$ = 388 ns and $D_{1\_shortest\_maxload}$ = 316 ns which once more indicates a large delay and so a reduction in the speed due to wiring.

### 4.1.3 Block of equality nodes with Boosted circuits

As mentioned before, one way to speed up the analog decoder is by boosting the bias currents and transistor sizes of just the input mirror branches. In order to maintain the right decoding operation we need to do so for the output diode-connected transistors as well since as shown in Fig 4.1.1, the output diode connected transistors are in fact the input transistors for the subsequent blocks. In this case, the total power dissipation of the equality nodes block could be large though it is not doubled. Therefore the power/speed ratio of the block will be slightly improved.

The input bias currents, *A(in)* up to *G(in)* and the corresponding diode connected transistors of the block of equality nodes are increased by a ratio of $k = 6$. Consequently the output bias currents, *A(out)* to *G(out)* and related output transistors (as the inputs of next blocks) should be boosted by the same ratio. The total power dissipation in this case is found to be $P_2 = 3.12\ \mu W$ which is about 3 times the dissipated power for the normal case. However the timing delay of the input current-mirror circuit in the presence of the wiring capacitance is reduced to $D_{2\_maxload} = 46.85$ ns. Also using the boosted circuits, the total delay of the blocks of equality nodes for the longest and the shortest paths are changed to $D_{2\_longest\_maxload} = 132.11$ ns and $D_{2\_shortest\_maxload} = 64.3$ ns.

As a figure of merit, the power/speed ratio or the power delay product for the shortest path is 203.7 fJ. This is somewhat less than that of normal case which is 284.4 fJ. Therefore the power/speed ratio is said to be improved slightly.

## 4.2 Block of equality nodes with modified input current-mirrors

In this Section, a current-mode technique is applied to the basic current-mirrors of a block of equality nodes in order to improve the speed.

### 4.2.1 Modified current-mirror circuit

Fig 4.2.1 shows a modified topology for the traditionally used current-mirror circuit at the input of the equality or check node. It is composed of a negative feedback loop using an operational transconductance amplifier (OTA). In this figure, $V_{ref}$ is the reference voltage of the OTA which is used to keep M1 in the saturation region.

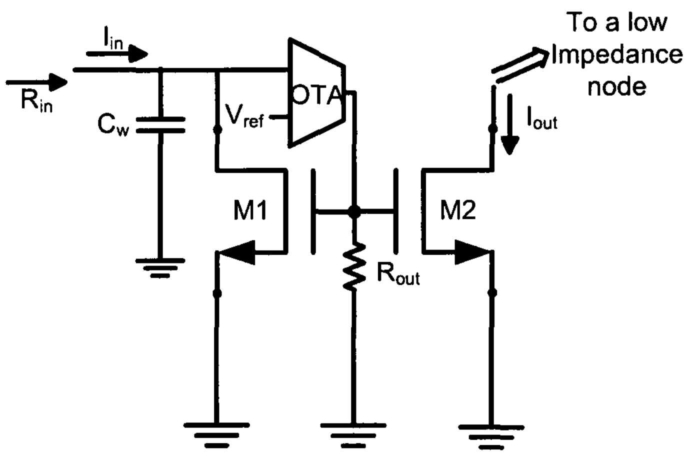

Fig 4.2.1: Modified current-mirror circuit with the OTA feedback

Based on the small signal analysis of this circuit shown in Fig 4.2.2, the input resistance is found as:

$$R_{in} = \frac{1}{1/r_{o1} + G_m g_{m1} R_{out}} \tag{4.2.1}$$

where $G_m$ is the transconductance of the OTA and $R_{out}$ is the output resistance of the amplifier used to convert the output current to voltage.

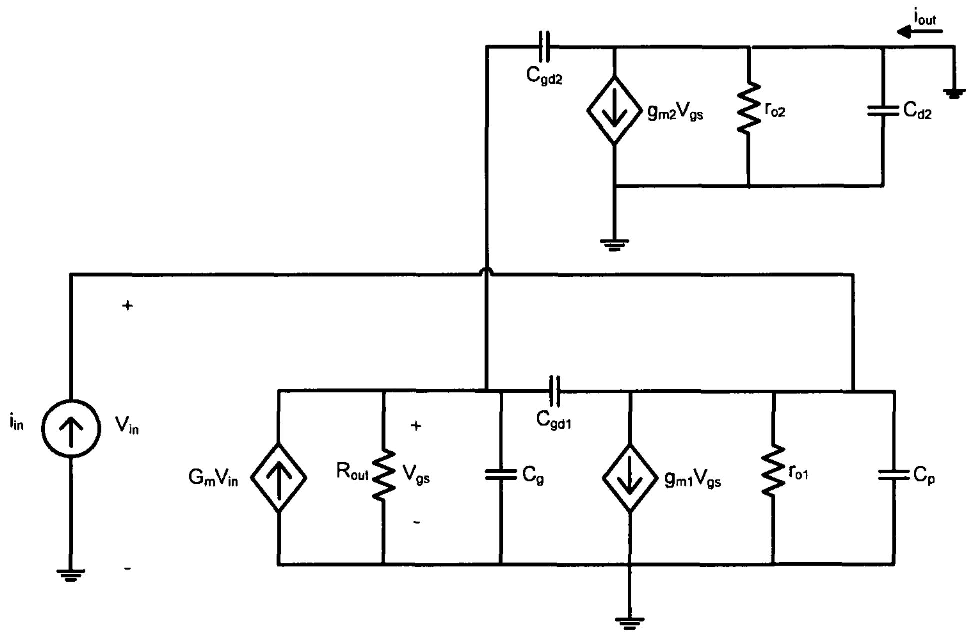

Fig 4.2.2: Small signal circuit of the modified current-mirror

It is clear from (4.2.1), that the input resistance for the modified current-mirror circuit can be reduced by increasing the transconductance $G_m$, hence the -3dB bandwidth will be improved. Fig 4.2.3 shows the plots of -3dB bandwidth for different ratios of $G_m / g_{m1}$ as functions of the input wiring capacitance. Again in this picture, the corresponding bandwidths for the derived worst case capacitance of $C_w = 500\,\text{fF}$ are pointed out. However, this may give rise to low phase margin problems for the modified current-mirror since there will be two high impedance nodes in the loop in this case: one at the drain of M1 and the other at the gates of M1 and M2. As a result, the poles may be close to one another, lowering the phase margin. Nevertheless by choosing the right values for $G_m$, $R_{out}$, and $C_w$, acceptable phase margin could be achievable. In our discussion, the phase margin is always greater than 50°.

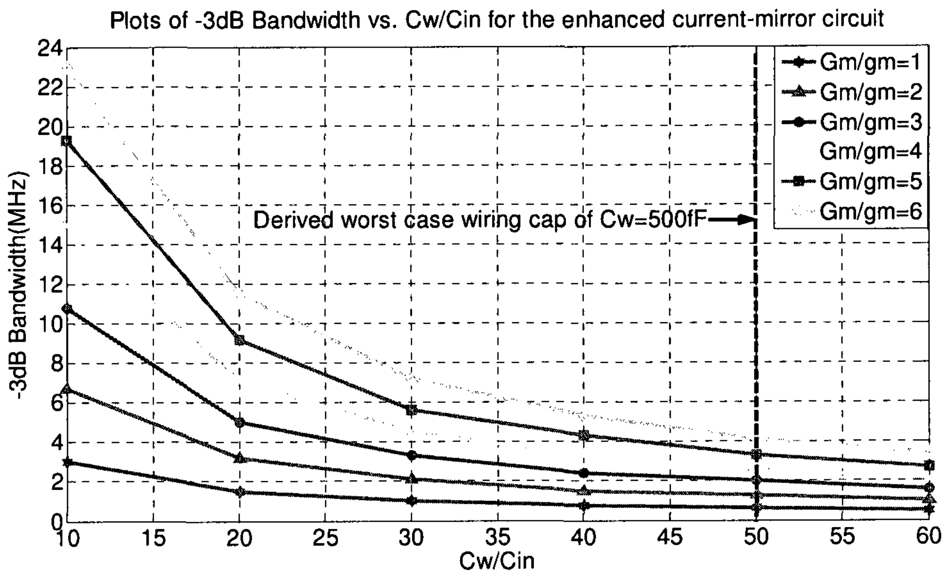

Fig 4.2.3: Plots of -3dB Bandwidth vs. $C_w / C_{in}$ for the modified current-mirror circuit in the equality node

## 4.2.2 Design of the OTA circuit in the modified current-mirror

The schematic of the OTA circuit has been shown in Fig 4.2.4. It is designed such that all the transistors are in weak-inversion. Sizes of transistors, values of $V_{ref}$ and $I_{bias2}$ must be chosen so that M1, M3, and M4 are saturated (i.e. $V_{ds}$ = 200 mV) while operating in the sub-threshold region. Note that saturation of M2 is not required since the low-voltage sum-product topology has been used in the design of the individual equality nodes [57]. To control the transconductance of the OTA, $I_{bias1}$ is increased with respect to the input current ($I_{in}$) of M1 which will be shown to be the same as increasing the ratio of $G_m / g_{m1}$

.

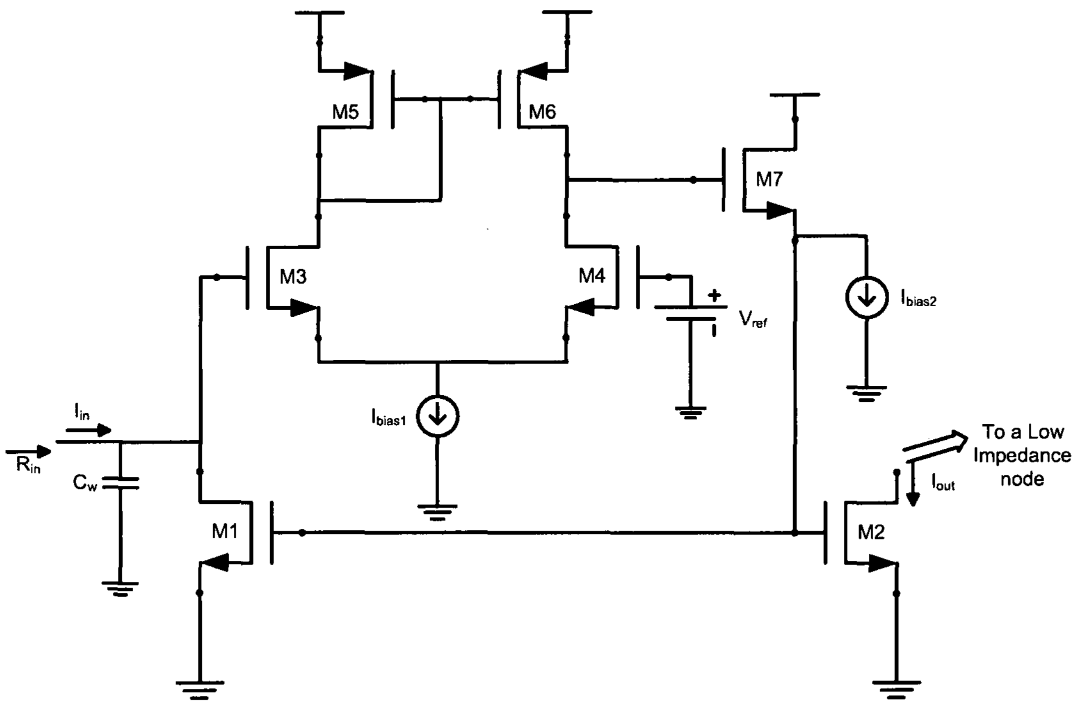

Fig 4.2.4: Modified topology for the current-mirror circuit

showing the schematic of the OTA

It is worth mentioning that without M7, M4 would go into triode. Therefore by adjusting $I_{bias2}$ one can set the gate-source voltage of M7 such that M4 is saturated:

$$V_{d4} = V_{gs7} + V_{gs1} \qquad (4.2.2)$$

Small signal analysis of the above OTA is explained in Fig 4.2.4. Since for transistors in weak-inversion $g_m = \frac{I_D}{nU_T}$ where $n$ is the slope factor and $U_T$ is the thermal voltage, we have:

$$G_m = \left.\frac{I_{out}}{V_{in}}\right|_{V_{out}=0} = \frac{g_{m3,4}V_{in}}{V_{in}} = g_{m3,4} = \frac{I_{D3,4}}{nU_T} = \frac{I_{bias1}}{2nU_T} \tag{4.2.3}$$

Based on (4.2.3), it is observed that by increasing $I_{biasI}$ in Fig 4.2.4, the transconductance of the OTA will be increased and so the parameter $\frac{G_m}{g_{m1}}$ can be interpreted as the ratio of currents of M1 and $I_{biasI}$ in the OTA.

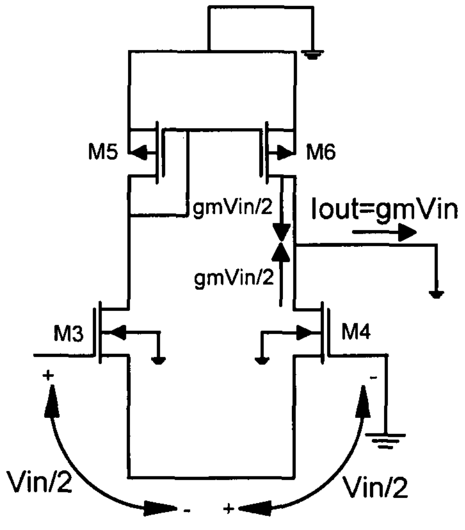

Fig 4.2.5: Small signal analysis of the OTA circuit

## 4.2.3 Calculation of the Power/Speed ratio

The ultimate goal of our study is to find a reasonable solution to the issue of speed in large decoders which yields an enhanced power/speed ratio. Obviously there are several approaches for this reason other than those discussed in this Chapter. We have claimed that our suggested current-mode technique can satisfy the above mentioned goal. In order to prove this, the input basic current-mirror circuits of the block of equality nodes shown in Fig 4.1.1 are replaced with the enhanced mirrors.

In designing the OTA circuits, in addition to the design requirements presented in section 4.2.2, we allowed the total power dissipation of the total block in this case denoted as $P_3$ twice $P_1$ that is $P_3 = 2 \times P_1 = 2 \times 0.9$ µW = 1.8 µW with the same supply voltage of 400 mV providing that most of the power is dissipated by the input enhanced mirrors rather than by the decoding part itself. Note that choosing $P_3$ to be twice $P_1$ is just an arbitrary choice.

It was found that by adjusting $G_m / g_{m1} = 3$, the aforementioned conditions are met. On the plots of Fig 4.2.3 this point corresponds to a -3dB bandwidth of 2 MHz for an input modified current-mirror circuit which is about 4 times the bandwidth of the basic mirror in the presence of the worst wiring capacitance. This was also verified by applying a step current at the input of the modified mirror that gave rise to a timing delay of $D_{3\_maxload} = 64.3$ ns.

Regarding the total delay of the block of equality nodes, it was obtained that $D_{3\_longest\_maxload} = 164$ ns and $D_{3\_shortest\_maxload} = 80$ ns for the longest and the shortest paths respectively.

The results of the equality nodes block with the basic, boosted and modified mirrors have been summarized in Table 4.2.1. From this table, it can be seen that the active current mirrors improve the power delay product for the 7-input equality nodes block for both the shortest path and longest path through the node. The improvement is most significant for the shortest path through the node (~50%). However, with the boosted circuits, there is no considerable improvement.

Table 4.2.1: Summary of results for the equality nodes block with the basic, boosted and modified mirrors in a large (6,3) LDPC analog decoder

| **Quantity** | **Basic Block for $C_w$= 0** | **Basic Block for $C_w$=500 fF** | **Block with Boosted mirrors for $C_w$=500 fF** | **Modified Block for $C_w$=500 fF** |
|---|---|---|---|---|
| Input mirror Delay (ns) | 3.0 | 279 | 46.85 | 64.3 |
| Shortest Path Delay (ns) | 14.4 | 316 | 64.3 | 80.0 |
| Longest Path Delay (ns) | 79.5 | 388 | 132.11 | 164 |
| Total power dissipation | 0.9 | 0.9 | 3.12 | 1.8 |
| P(input mirrors)/P(total) | - | 30% | 53% | 68% |
| Power Delay Product, Shortest Path (fJ) | - | **284.4** | **203.73** | **144** |
| Power Delay Product, Longest Path (fJ) | - | **349.2** | **412.18** | **294.2** |

## 4.3 Conclusion

To reduce the effect of wiring capacitance on the speed of an analog decoder circuit, a current-mode technique was applied. This technique was employed in the current-mirror circuits of a block of equality nodes for a large LDPC code defined by a (512 x 1024) *H*-matrix. Having estimated the worst case wiring capacitance for the code, a significant reduction in the speed of basic current-mirrors was observed. However, replacing the basic mirrors with the enhanced topology showed that the achieved speed in the block of equality nodes is more than twice the speed with the basic current-mirrors, while the total power dissipation is doubled for the modified case. Thus an improvement in the power/speed ratio for the complete decoder is expected if the basic mirrors are replaced with the enhanced mirrors.

# Chapter 5

# Non-Linear vs. Linearized LDPC decoder

A decoder by its very nature is a non-linear circuit whose output is not a linear function of inputs [70]. Soft decision receivers dealing with the information probabilities reveal this nonlinearity through the non-linear blocks of equality and check nodes. It was already observed that these parity-check nodes or constraint nodes have to perform non-linear functions such as multiplication and division in the sum-product algorithm.

Theoretical dynamic analysis which leads to determining the convergence speed of the analog iterative decoders has been provided by some researchers such as Hemati and Banihashemi in [68]. In particular they showed iterative decoding as a fixed point problem and demonstrated a model for continuous-time iterative decoding by including first order RC circuit between the equality and check nodes. Finally they applied a numerical method to convert the obtained differential equations to iterative method for solving the fixed point problem. In their survey, the initial values for the edges in the decoder prior to applying the real received information corresponded to uniform *a priori* probabilities (the 0.5 probability for all edges).

In this Chapter, we are going to define the equality and check nodes by their respective equations. Further we assume a uniform propagation delay (i.e. first order RC circuit) due to the wiring capacitances and the input resistances of these nodes between the equality and check nodes. The corresponding Tanner graph to the $H$-matrix of Chapter 2 which defines a regular LDPC code has been applied for this purpose.

Later in this Chapter, we will consider the nonlinear LDPC decoder as a nonlinear system in the state space. Thus following the linearization technique in the state space, we try to linearize the aforementioned decoder which might help in dynamic analysis of the system. The simulations have been done in MATLAB's Simulink environment.

## 5.1 Non-Linear LDPC decoder

Fig 5.1.1 depicts the Tanner graph for a regular LDPC code with the $H$-matrix presented in Chapter 2. A parallel first order RC circuit has been also taken into account as the uniform distribution delay between the equality and check nodes. Note that in our simulations, the RC circuit has been normalized (i.e. RC = 1sec).

Furthermore, we label the edges entering and leaving a check node by $f_i$ and $y_i$ and those of an equality node by $g_i$ and $x_i$ respectively, where $i$ varies from 1 to the number of edges (in this graph 21). It is also assumed that the processing delays for individual nodes are zero. By establishing the required equations for the nodes, one can model the LDPC decoder in the MATLAB Simulink environment.

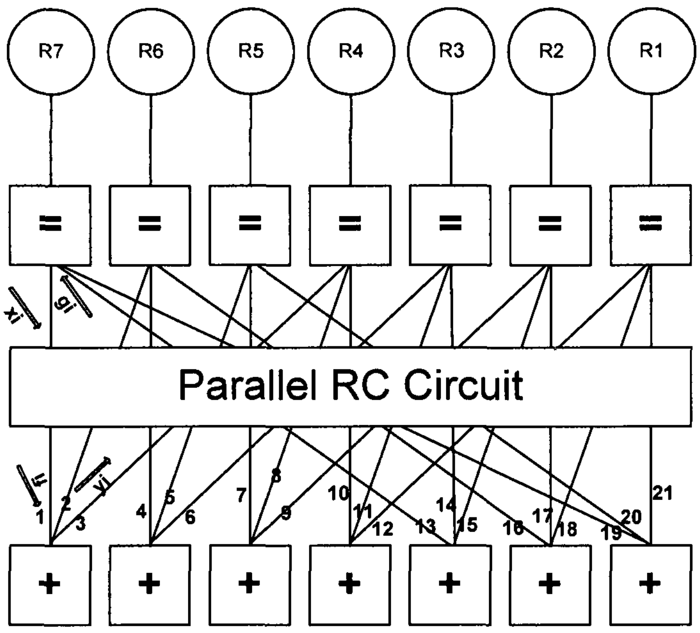

Fig 5.1.1: Tanner graph for a regular LDPC code with the parallel RC delay circuit

In the above graph, variable nodes are fed by the received information bits R7~R1, however the final estimated output coming out of the graph is not shown here. The G-matrix for the corresponding LDPC code is as below:

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

The original information messages to get decoded are:

$\underline{u_1} = [0\ 0\ 0]$, $\underline{u_2} = [0\ 0\ 1]$, $\underline{u_3} = [0\ 1\ 0]$, $\underline{u_4} = [0\ 1\ 1]$,

$\underline{u_5} = [1\ 0\ 0]$, $\underline{u_6} = [1\ 0\ 1]$, $\underline{u_7} = [1\ 1\ 0]$, $\underline{u_8} = [1\ 1\ 1]$.

Thus based on (2.2.1) their respected codewords are as follows:

$\underline{x_1} = [0\ 0\ 0|0\ 0\ 0\ 0]$, $\underline{x_2} = [0\ 0\ 1|0\ 1\ 1\ 1]$, $\underline{x_3} = [0\ 1\ 0|1\ 1\ 1\ 0]$, $\underline{x_4} = [0\ 1\ 1|1\ 0\ 0\ 1]$,

$\underline{x_5} = [1\ 0\ 0|1\ 0\ 1\ 1]$, $\underline{x_6} = [1\ 0\ 1|1\ 1\ 0\ 0]$, $\underline{x_7} = [1\ 1\ 0|0\ 1\ 0\ 1]$, $\underline{x_8} = [1\ 1\ 1|0\ 0\ 1\ 0]$

It can be seen that the code rate in the above codewords is 3/7 meaning that only three of the information bits are useful and the rest four are redundant.

We are going to simulate the modeled decoder to see how the output bits are converged to a true codeword. For this purpose, $\underline{x_1}$ and $\underline{x_6}$ are transmitted through the AWGN channel with BPSK modulation and the signal to noise ratio (SNR) has been set to 1 dB. The decoder has been initialized with the uniform probability mass (all 0.5 probability) prior to decoding. Fig 5.1.1 and Fig 5.1.2 show how the decoder converges to codewords $\underline{x_1}$ and $\underline{x_6}$ respectively.

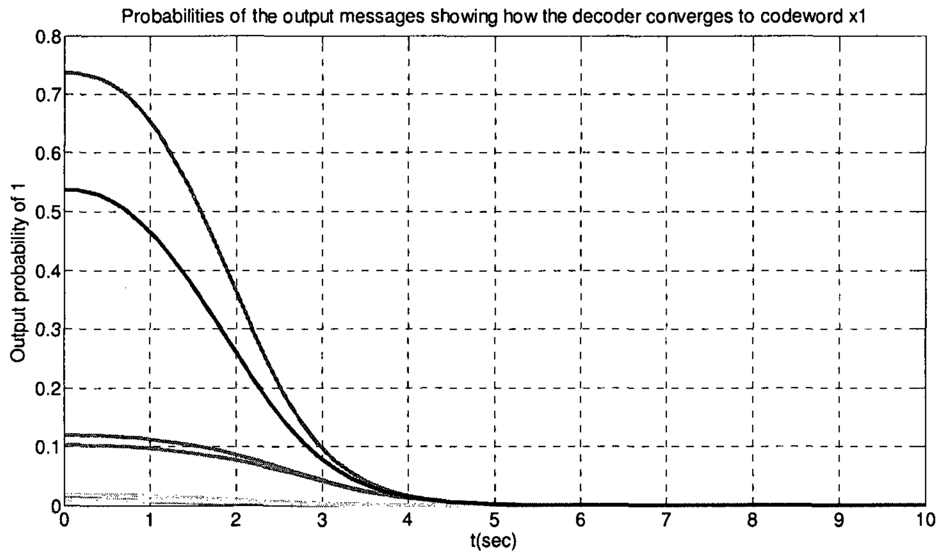

Fig 5.1.1: Output probabilities of 1 for the simulated LDPC decoder showing the convergence to the zero codeword $\underline{x_1}$

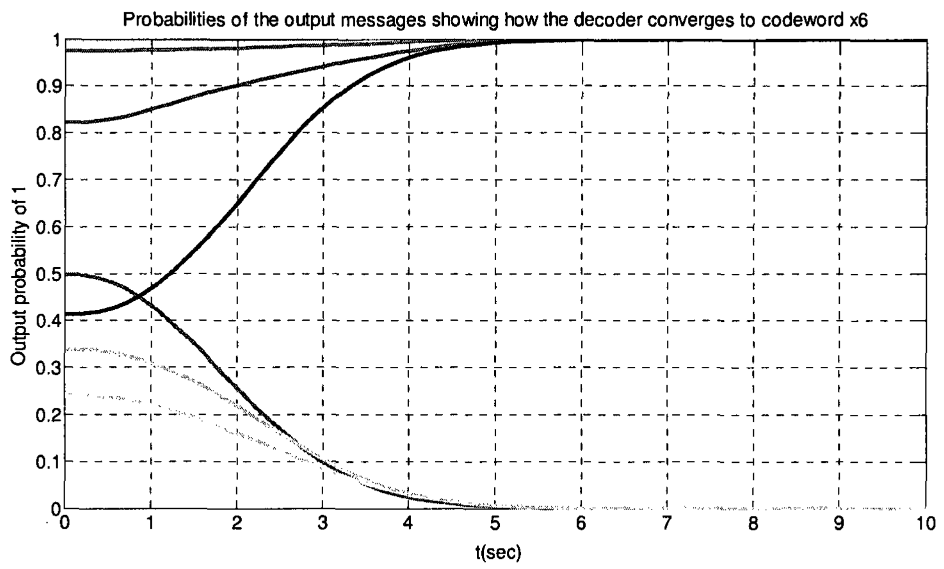

Fig 5.1.2: Output probabilities of 1 for the simulated LDPC decoder showing the convergence to the zero codeword $\underline{x_6}$

## 5.2 Linearized LDPC decoder

Linearization is an effective method for approximating the output of a nonlinear function $y = f(x)$ at any $x = a$ based on the value and slope of the function at $x = b$, given that $f(x)$ is continuous on $[a,b]$ (or $[b,a]$) and that $a$ is close to $b$. Briefly, linearization approximates the output of a function near $x = a$. If $\delta x$ represents small variations of $x$ around $x = a$, then the nonlinear function $y = f(x)$ can be written as below:

$$y = f(x) = f(a) + \left.\frac{\partial f}{\partial x}\right|_{x=a} * \delta x + \left.\frac{\partial^2 f}{\partial x^2}\right|_{x=a} * \delta^2 x + \cdots \qquad (5.2.1)$$

where higher order terms can be ignored as they decay.

In this section the LDPC decoder will be considered as a nonlinear control system in the state space. It is intended to linearize the LDPC decoder by applying linearization technique. The result of this survey could be to predict the behavior of the decoder at small vicinity around the initialization point and to estimate the decoding speed by having the poles and zeroes of the linear system.

### 5.2.1 State-space matrices

In order to linearize the LDPC decoder, first of all we provide the equality and check equations in the likelihood ratio domain where $f(p_0, p_1) = g(p_0, p_1) = R(p_0, p_1) = \frac{p_0}{p_1}$ for every edge of the graph. Since there is a total of 21 parity check equations, we will end up with two large (21 x 1) matrices $[x_i]_{21*1}$and $[y_i]_{21*1}$ for which $x(g_a, g_b, R_c) = g_a g_b R_c$ and $y(f_a, f_b) = \frac{1+f_a f_b}{f_a+f_b}$ respectively. In the recent definitions, *a*, *b*, and *c* are the labels which represent three individual edges either for the equality nodes or the check nodes.

$$[x_i]_{21*1} = \begin{bmatrix} g_{13}g_{19}R_7 \\ g_4g_{16}R_6 \\ g_8g_{10}R_4 \\ g_2g_{16}R_6 \\ g_7g_{20}R_5 \\ g_{11}g_{14}R_3 \\ g_5g_{20}R_5 \\ g_3g_{10}R_4 \\ g_{15}g_{17}R_2 \\ g_3g_8R_4 \\ g_6g_{14}R_3 \\ g_{18}g_{21}R_1 \\ g_1g_{19}R_7 \\ g_6g_{11}R_3 \\ g_9g_{17}R_2 \\ g_2g_4R_6 \\ g_9g_{15}R_2 \\ g_{12}g_{21}R_1 \\ g_1g_{13}R_7 \\ g_5g_7R_5 \\ g_{12}g_{18}R_1 \end{bmatrix} \tag{5.2.2}$$

$$[y_i]_{21*1} = \begin{bmatrix} \frac{1+f_2f_3}{f_2+f_3} \\ \frac{1+f_1f_3}{f_1+f_3}. \\ . \\ . \\ . \\ . \\ \frac{1+f_{19}f_{20}}{f_{19}+f_{20}} \end{bmatrix} \tag{5.2.3}$$

Further,$f_0$, $g_0$ and $R_0$ are assumed to be the initialization points around which one can linearize the above nonlinear functions.

$$[y_i]_{21*1} = [y_i]|_{f_j=f_0} + \left[ \frac{\partial y}{\partial f_j}\bigg|_{f_j=f_0} \right]_{21*21} * \left[\delta f_j\right]_{21*1} \qquad (5.2.4)$$

$$[x_i]_{21*1} = [x_i]|_{g_j=g_0} + \left[ \frac{\partial x}{\partial g_j}\bigg|_{g_j=g_0} \right]_{21*21} * \left[\delta g_j\right]_{21*1} + \left[ \frac{\partial x}{\partial R_j}\bigg|_{R_j=R_0} \right]_{21*7} \left[\delta R_j\right]_{7*1} \qquad (5.2.5)$$

We are interested in deriving a general matrix which describes all the relationships between the edges of the Tanner graph for small deviations from the initialization point. Hence by ignoring the first terms in (5.2.4) and (5.2.5) and expressing $y_i$ and $x_i$ as $\delta y_i$ and $\delta x_i$ respectively, the linearized functions are reduced to:

$$[\delta y_i]_{21*1} = C * \left[\delta f_j\right]_{21*1} \qquad (5.2.6)$$

$$[\delta x_i]_{21*1} = A * \left[\delta g_j\right]_{21*1} + B * \left[\delta R_j\right]_{7*1} \qquad (5.2.7)$$

where $A = \left[ \frac{\partial x}{\partial g_j}\bigg|_{g_j=g_0} \right]_{21*21}$, $B = \left[ \frac{\partial x}{\partial R_j}\bigg|_{R_j=R_0} \right]_{21*7}$ and $C = \left[ \frac{\partial y}{\partial f_j}\bigg|_{f_j=f_0} \right]_{21*21}$.

Taking into account the first order RC delay circuit between the equality and check nodes, the transfer functions from $xi$ to $fi$ and from $yi$ to $gi$ would be:

$$[f_i] = \frac{[x_i]}{1+RCs} \qquad (5.2.8)$$

$$[g_i] = \frac{[y_i]}{1+RCs} \qquad (5.2.9)$$

Consequently by replacing the above transfer functions in (5.2.4) and (5.2.5), we arrive at the following equations:

$$\left[\frac{df_i}{dt}\right] = \frac{1}{RC}([x_i] - [f_i]) \rightarrow \frac{d}{dt}[\delta f_i] = \frac{1}{RC}(A * [\delta g_i] + B * [\delta R_i] - [\delta f_i]) \qquad (5.2.10)$$

And,

$$\left[\frac{dg_i}{dt}\right] = \frac{1}{RC}([y_i] - [g_i]) \rightarrow \frac{d}{dt}[\delta g_i] = \frac{1}{RC}(C * [\delta f_i] - [\delta g_i]) \qquad (5.2.11)$$

If we normalize the above expressions with respect to $\frac{1}{RC}$ , the equations can be rearranged as below:

$$\begin{bmatrix} \dot{\delta f} \\ \dot{\delta g} \end{bmatrix}_{42*1} = \begin{bmatrix} -I & A \\ C & -I \end{bmatrix}_{42*42} * \begin{bmatrix} \delta f \\ \delta g \end{bmatrix}_{42*1} + \begin{bmatrix} B \\ 0 \end{bmatrix}_{42*7} * [\delta R]_{7*1} \tag{5.2.12}$$

where $I$ is a (21 * 21) Identity matrix.

Also for the sake of simplicity we further define $\begin{bmatrix} \delta f \\ \delta g \end{bmatrix} = X, \begin{bmatrix} -I & A \\ C & -I \end{bmatrix} = A_{tot}, \begin{bmatrix} B \\ 0 \end{bmatrix} = B_{tot}, \delta R = \text{U}$.

$$\rightarrow \dot{X} = A_{tot} * X + B_{tot} * U \rightarrow (SI - A_{tot})X = X_0 + B_{tot}U$$

$$\rightarrow X = (SI - A_{tot})^{-1}(X_0 + B_{tot}U) \tag{5.2.13}$$

and $X_0 = \begin{bmatrix} \delta f_0 \\ \delta g_0 \end{bmatrix}$ is the initial condition or the linearization point for the above state space equation.

### 5.2.2 The issue of linearization point

Thus far we have linearized a nonlinear LDPC decoder and derived the state space equation, however the choice of linearization point would be still of a concern. Here we are going to study the issue associated with the linearization point.

Let us assume that the uniform probability mass has been applied to the edges of the graph as the initialization point. This point would be also used to linearize the LDPC decoder, that is why we call it as the linearization point. The uniform probability mass as mentioned earlier requires all edges to have a probability of 0.5. This requirement would be translated to having a value of 1 in the likelihood domain. Therefore based on (5.2.6) and (5.2.7), $A$, $B$, and $C$ can be defined as below:

$$A=\begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0\\
0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0\\
0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0
\end{bmatrix}_{21*21}$$

$$B=\begin{bmatrix}
0&0&0&0&0&0&1\\
0&0&0&0&0&1&0\\
0&0&0&1&0&0&0\\
0&0&0&0&0&1&0\\
0&0&0&0&1&0&0\\
0&0&1&0&0&0&0\\
0&0&0&0&1&0&0\\
0&0&0&1&0&0&0\\
0&1&0&0&0&0&0\\
0&0&0&1&0&0&0\\
0&0&1&0&0&0&0\\
1&0&0&0&0&0&0\\
0&0&0&0&0&0&1\\
0&0&1&0&0&0&0\\
0&1&0&0&0&0&0\\
0&0&0&0&0&1&0\\
0&1&0&0&0&0&0\\
1&0&0&0&0&0&0\\
0&0&0&0&0&0&1\\
0&0&0&0&1&0&0\\
1&0&0&0&0&0&0
\end{bmatrix}_{21*7} \quad \text{and } C=0$$

Note: The reader can easily derive these matrices by taking derivations from $[\delta x_i]_{21*1}$ and $[\delta y_i]_{21*1}$ with respect to $g_j$, $f_j$, and $R_j$ at the linearization point $g_0 = f_0 = R_0 = 1$.

Since $C$ was found to be zero at the uniform probability mass linearization point, then the output messages from the check nodes will not have any sensitivity to the small input variations and therefore $\delta y_i = 0$. Consequently the linearized decoder will not work properly. This issue is raised from the fact that $f_0$ has been set to 1 which causes $C = \left[\frac{\partial y}{\partial f_j}\Big|_{f_j=1}\right]_{21*21}$ to be equal to zero. Therefore in order to avoid this problem we may change the linearization point. This can be done only by setting $f_0$ and $R_0$ to any value except 1 while $g_0$ could be still set to 1. It is worth mentioning that $f_0$ or the initial down going messages from the equality nodes to the check nodes would have the same value as $R_0$ since all the upcoming messages have been already set to 1.

By setting $R_i = f_i = 2$ as a new linearization point the existing problem would be sorted out, however as we will see a new problem in the decoding procedure may occur. For this new setting the probability of being 1 for down going edges of the Tanner graph is $p_1 = 0.3$ while that of upcoming edges is still $p_1 = 0.5$. Now if one the seven input bits in the linearized decoder slightly increases around the linearization point, the affected down going edge is expected to have same slope as in the corresponding non-linear LDPC decoder at the beginning. This fact has been shown in Fig 5.2.1.

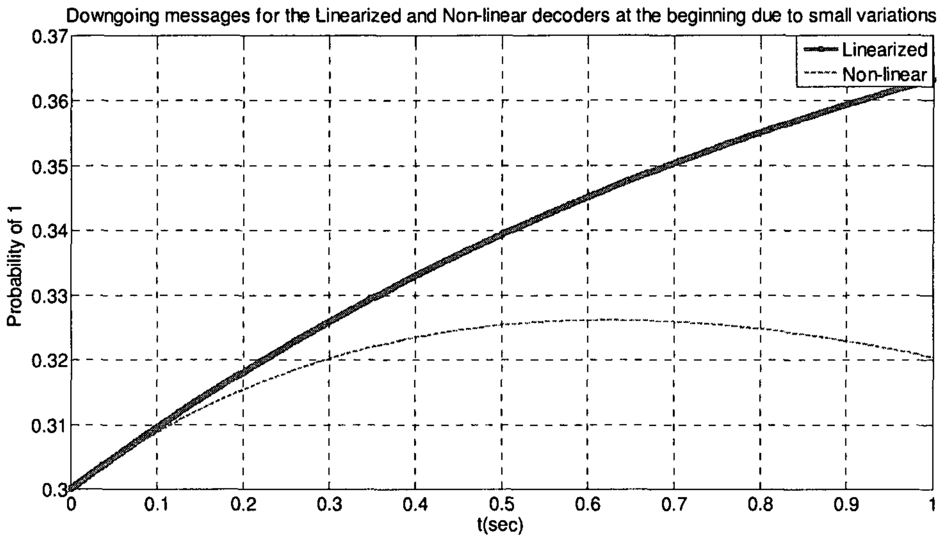

Fig 5.2.1: Same slopes for the down going messages in linearized and non-linear decoders due to small variations of the input bits

However the upcoming messages have different slopes at the beginning based on Fig 5.2.2.

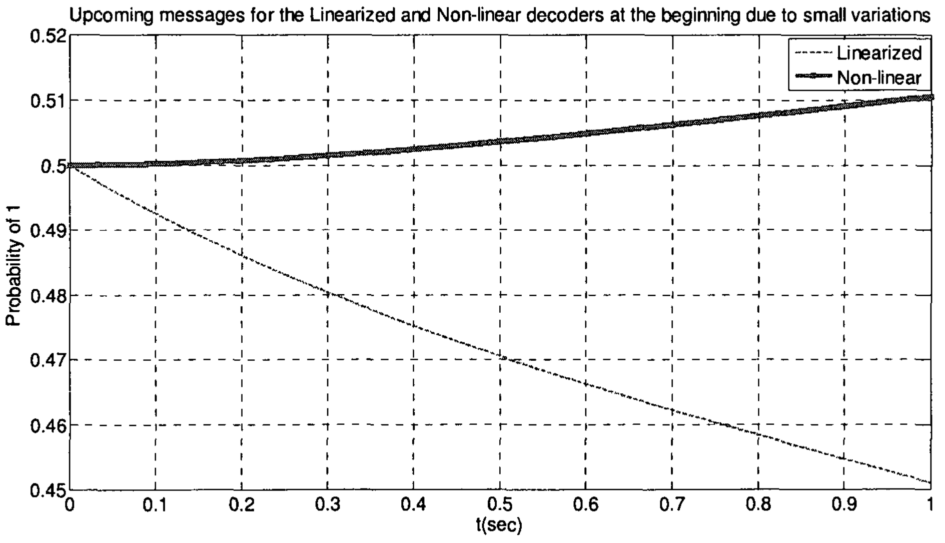

Fig 5.2.2: Different slopes for the upcoming messages in linearized and non-linear decoders due to small variations of the input bits

It should be noted that the functionality of linearized decoder depends on the small variations of input $R_i$ whereas in a real decoder the edges may divert from their initial values even if the inputs are not changed at all. In our example the down going edges are initialized to $f_i = 2$ which means that $p_0 = 2/3$ and $p_1 = 1/3$ in the probability domain. In this condition, if the received samples are equal to the initial values, then the decoder will converge to the zero codeword. This is in contrast with the linearized decoder where a small variation to the inputs must occur in order to start the decoding process.

Finally we conclude the application of linearization technique in decoders. Every nonlinear system can be linearized around a solution provided that the whole system is stable at that solution point, otherwise the output of the system may change even if there is no perturbation at the input. For the decoder case, in order to fulfill the above requirement we need all the edges to have a uniform probability mass (i.e. the 0.5 probability) as the linearization point since at this point the decoder would remain stable unless one of the input bits changes around that point. It was shown that some of the parameters in the linearized decoder around the uniform probability mass become zero which is undesirable and so the uniform probability mass cannot be the right solution point for applying the linearization technique. Note that other valid codewords could also be taken as the stable solutions and therefore the linearization may work around those points, however we will postpone this to our future work.

# Chapter 6

# Testing the chip

Testing a chip could be a time consuming process. It may include thinking of a reasonable test plan, designing a test board, writing the appropriate program for the tester, etc. Meanwhile there is always the inevitable concern of failure in the operation which might be difficult to track down. In Chapter 3, we talked about the details of our chip. Unfortunately the testing of the chip failed!

## 6.1 The internal view of the Chip

As explained earlier, in our chip we have a total of 64 blocks including the equality and check nodes as well as the input and output switches and digital circuits. Moreover, it contains two other designs, unrelated to this work. Fig 5.1.1 shows how the layout of our design which is located in the left half has been separated from other parts of the chip.

A large rectangular shape can be distinguished on the layout view of Fig 6.1.1 which is the body of our design and is composed of our 64 blocks. At the top and the bottom of this rectangle, the input and output switches as well as the voltage to current converters are located. The digital part of the design has been also laid out at the left side of the 64 blocks. A pad ring can be seen all around the left half of the layout which is used for the analog and digital inputs and outputs. Dimensions for every part of the design have been summarized in Table 6.1.1.

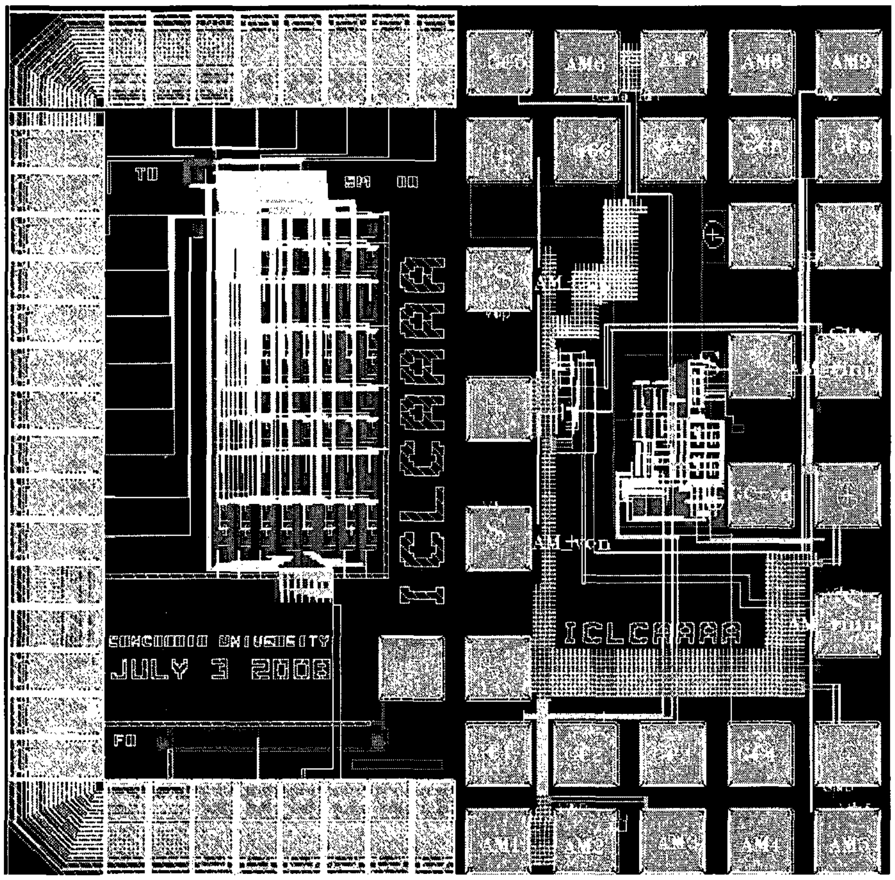

Fig 6.1.1: The layout of the chip with our design at the left

Table 6.1.1: Area distribution of the chip

| | The whole chip | The 64 blocks | The digital circuit | One of the V2I circuits | The input switches | The Output switches |
|---|---|---|---|---|---|---|
| Area (mm$^2$) | 1 | 0.086 | 0.0014 | 0.0002 | 0.0045 | 0.0026 |

The complete layout has been inserted inside a *thin quad flat pack (TQFP)* package with 52 pins. Fig 6.1.2 depicts the bonding diagram between the individual pads of the ring and the package pins. The labeling on the outer edge of the package indicates that only 22 out of 52 pins of the TQFP-52 package are going to be used for our test.

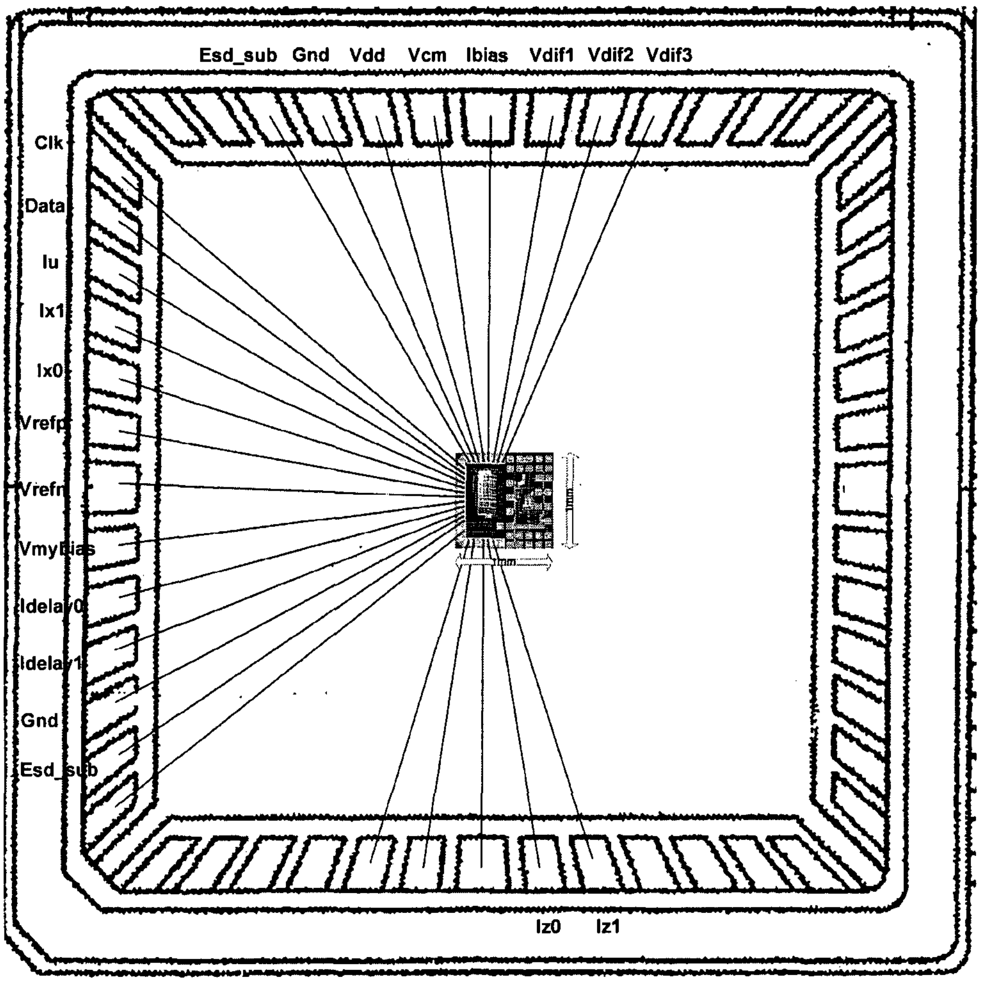

Fig 6.1.2: Bonding diagram of the chip in the TQFP-52 package

### 6.1.1 Arrangement of the pins

The input and output pins in the bonding diagram of Fig 6.1.2 are organized as follows:

1. Input current sources: $\boldsymbol{I_U, I_{bias}, I_{x0}, I_{x1}}$

$I_U$ and $I_{bias}$ are both the global unit currents used in the sum-product circuits. The former is the current source which has been employed in the equality and check nodes (i.e. the 64 blocks) of Chapter 3 while the latter is the unit current applied to the voltage to current converter of Fig 3.6.2.

$I_{x0}$, $I_{x1}$ had been already supposed to provide the input probabilities of 0 and 1 for the first block which is an equality node. However, as mentioned earlier the speed test was ignored.

2. Input Voltage sources: **($V_{dd}$, $V_{mybias}$, $V_{cm}$, $V_{refn}$, $V_{refp}$), ($V_{dif1}$, $V_{dif2}$, $V_{dif3}$), (*clk*, *data*)**

The first group of the voltage sources includes $V_{dd}$ as the largest voltage source in the design which is used for both analog and digital circuits. $V_{mybias}$ on the other hand is the bias voltage for equality and check nodes. $V_{cm}$ is one of the input voltages in the differential pair used in the voltage to current converters of Fig 3.6.2. $V_{refn}$ and $V_{refp}$ are the reference voltages in the canonical sum-product circuits

The second group contains the three differential voltages $V_{dif1}$, $V_{dif2}$, $V_{dif3}$ which are used in the three V2I converters.

There are also two digital inputs for the digital circuit which are *clk* and *data.*

3. Output Currents: $\boldsymbol{I_{delay0}, I_{delay1}, I_{z0}, I_{z1}}$

$I_{delay0}$ and $I_{delay1}$ are ignored as they are the output nodes used in the speed test. $I_{z0}$ and $I_{z1}$ are the output currents indicating the probabilities of 0 and 1 for the selected block.

The *Esd_sub* pins which are used to protect the circuit against the electrostatic discharge are normally grounded.

### 6.1.2 The Test plan

Since we are going to do a DC measurement test, our test plan will be as below. The Teradyne Tester in the Mixed Signal Lab located at the McGill University has been used for this purpose. Based on Table 3.6.1 which matches the input data to the selected node, there will be 64 choices. Initially, the first data would be used in order to calibrate the output current to the applied voltage.

**a)** Before measurement, all *ground* as well as *esd_sub* pins must be connected to the common ground. *Vdd* and *Vmybias* pins are set to 1.2 V and 600 mV respectively which must give the expected currents of -143 μA and -3.1 μA. The common mode voltage, *Vcm* pin will be connected to 395mV voltage source. Initially *Vdif1*, *Vdif2*, and *Vdif3* pins are connected to 395 mV. The voltage at the *Iu* = 100 nA pin is expected, based on the simulation to be near 430 mV. It was turned out from the simulation that setting *Ibias* to 86 nA with the expected voltage of 220 mV will provide a full range of output current changing from 0 to 100 nA. It is also worth mentioning that the direction of current for *Iu* is from inside the chip to the outside world while this is the opposite for *Ibias*.

**b)** The *Clk* pin is fed by a pulse generator as a clock alternating between 0 and 1.2 V. The period of the clock is set to 1 $\mu$sec. Then the input *data* consisting of 6 consecutive bits will be switched in on the rising edges of the clock to select one of the 64 nodes based on the Table 6.1.1.

**c)** Inputting the all zero data within the 6 clock pulses let us calibrating the input voltages *Vdif1, Vdif2* or *Vdif3* with their corresponding currents. The input currents are anticipated to change as shown in Fig 6.1.3 if *Vdif1* varies from 0.2 V to 0.6 V.

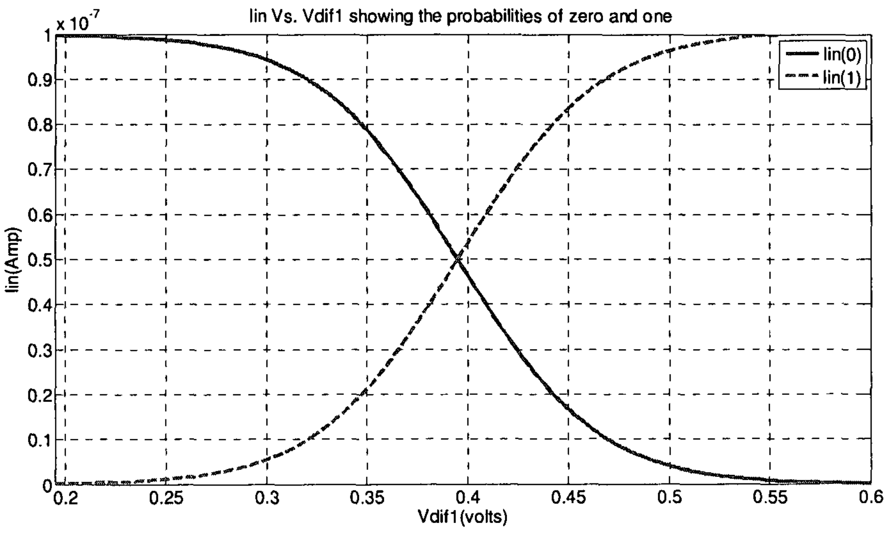

Fig 6.1.3: Variations of the input currents with respect to Vdif1

**d)** After the calibration part, one of the remaining 63 nodes in the block will be selected based on Table 3.6.1. We set *Vrefn* = 120 mV for all nodes. However *Vrefp* must be set according to the selected node that is for the equality (2-inputs or 3-inputs) nodes, *Vrefp* = 406 mV and for the check nodes *Vrefp* = 432 mV. For each of the selected nodes *Vdif1*, is changed according to the corresponding current which was derived from the calibration. We already mentioned that we place two 10 kΩ resistors at the output nodes, *Iz0* and *Iz1*. Using a multimeter with high input impedance, we will measure the voltages across the two output resistors. By running this procedure for all 63 nodes and saving the results, our test will be terminated.

## 6.2 Measurement process

Based on what presented above as a test plan, we provide all the required input voltages and currents for the chip and write several testing programs in Visual Basic. For the four input current sources of $I_U$, $I_{bias}$, $I_{x0}$, and $I_{x1}$ which are in the range of nano-meters, Keithley 2400 Source Meters are employed. Note that however $I_{x0}$ and $I_{x1}$ were supposed to be used in the speed test which was ignored later on, these two pins have been internally connected to two individual diode connected NMOS transistors with $W = 1.2$ μm and $L = 1$ μm and therefore could be ideal for running simple measurements which prove that the chip is alive. All of the input voltages have been connected within two connectors of type SAMTEC-150-01-L-D-VS to the motherboard of the Teradyne tester. The first connector takes care of the DC inputs $V_{dd}$, $V_{mybias}$, $V_{cm}$, $V_{refn}$, $V_{refp}$, $V_{dif1}$, $V_{dif2}$, *and* $V_{dif3}$ while the other one is used to provide digital signals *clk, data*.

Fig 6.2.1 shows the testing environment in the Mixed Signal Lab at McGill University with the test board mounted on the Teradyne tester. The Fluke voltmeter used to measure the voltages across the output 10 kΩ resistors has got an input impedance of 1M-ohm which seems enough for our purpose. The programs are run through the computer shown in the left of the picture. On the test board of Fig 6.2.2, a black clamshell socket for the chip can be seen. The two connectors located beneath the test board are mounted on the tester. Four BNC connectors used for the four input currents and a sample connection within the coaxial cable are also shown in the picture.

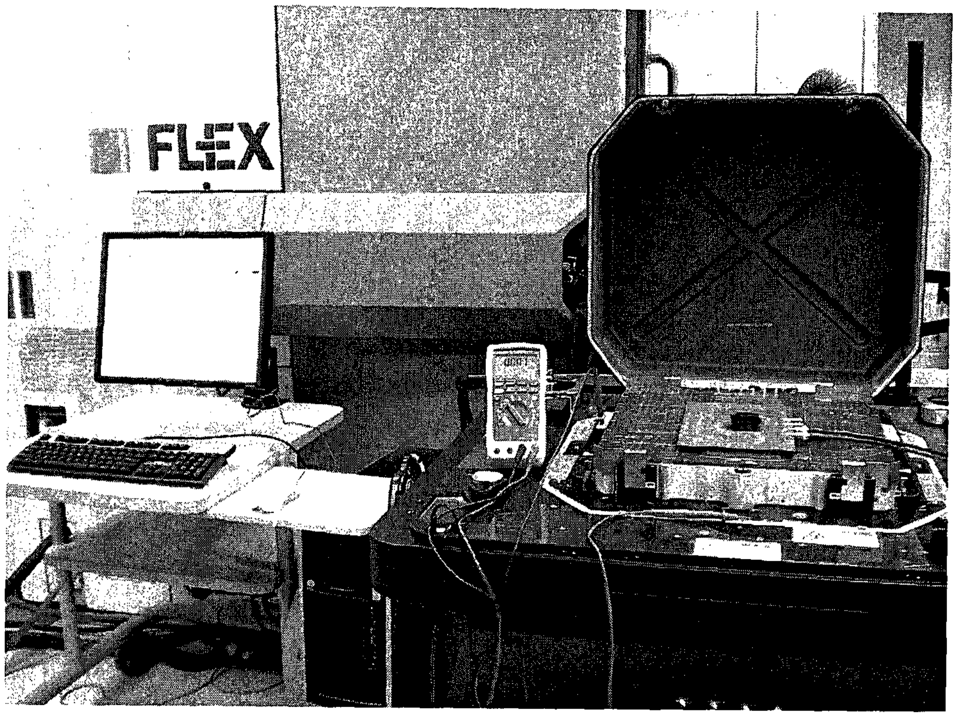

Fig 6.2.1: The Mixed Signal Lab at McGill University

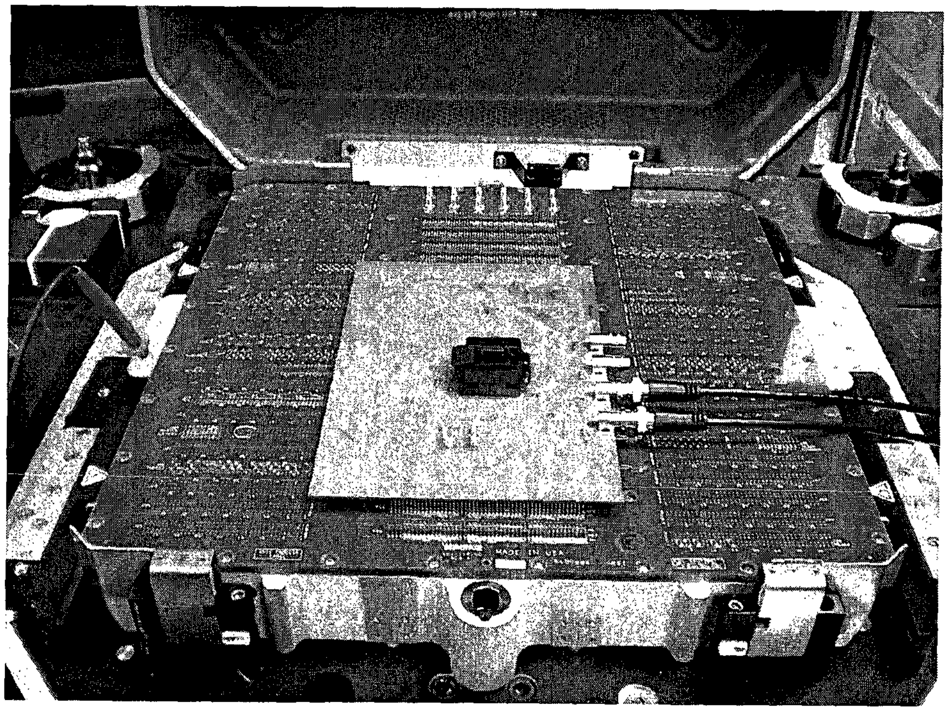

Fig 6.2.2: The test board mounted on the Teradyne tester

### 6.2.1 Simple measurements

In one of the programs, we simply set the required input voltages through the tester and expect to measure them on the test board using the voltmeter. It was figured out that the ground pins on the test board for some of the chips are showing relatively high voltages (i.e. around 8 mV or even more!) whereas for only three of them the ground pin has got 1.2 mV which is tolerable. This phenomenon made us believe that there might to be a problem with the chips.

Further examining the chips, we noticed that *Ix0* and *Vrefp* are internally short-circuited. This unexpected situation made us more curious about an internal failure inside the chip. Yet, we performed another measurement to check the healthiness of the chips. In this measurement, we only applied *Vdd, Vrefn*, *Ix0* and *Ix1* to obtain the V-I characteristics of the two diode-connected transistors for whose source are connected to *Vrefn.* Applying *Vdd* is vital for the pad ring of the chip. By increasing the input currents of *Ix0* and *Ix1*, we expect the gate-source voltage of the two diode-connected transistors to vary close to the simulation results of Fig 3.5.3. It is worth mentioning that since currents are low, hence the input impedance of the diode-connected transistors could be large and so comparable to the input impedance of the fluke voltmeter. Therefore voltage measurement in this case can be done directly through the Keithley devices.

It turned out that since *Ix0* and *Vrefp* are mistakenly short-circuited inside the chips, so the gate-source voltage of the related diode-connected transistor drawing *Ix0* does not change at all. For the other diode-connected transistor drawing *Ix1* from the current-source, we observed an unstable change in the gate-source voltage while increasing *Ix0* from 10 nA to 100 nA which was far from our expectations. Increasing the input current to higher values in the range of μA didn't really help and the problem still existed.

Since the above simple measurements could not fulfill our expectations, therefore we concluded that there might be something wrong with the chip. Later, running the calibration test and measuring constant values (near the zero voltage) across the output resistors showed that the chip is not functioning properly which might be due to a problem in the layout or in the packaging process and hence the test cannot be continued.

# Chapter 7

# Conclusion and Future Works

The goal of this thesis was to study the analog decoding circuits. For this reason we focused on a specific algorithm used in decoding (i.e. the sum-product algorithm) and studied the behavior of the sum-product circuits namely the equality and check nodes. In our survey, we observed the effect of mismatch on the equality node's outputs as well as its behavior in the extremes.

We found out that in the literature large codes are still preferred to be designed by digital circuits rather than with analog circuits. This was partly explained here using the large unwanted capacitance due to the wirings of the nodes in large codes. To solve the problem we applied a current-mode approach to lower the input impedance of the input transistors in each node. For this reason the basic diode-connected transistors at the input of each node which has significant input impedance due to the wiring, was replaced by an active current-mode circuit. By designing the OTA as the active part of the new circuit, we showed through simulation that the achieved speed in this case is more than twice with the basic current-mirrors while the power is only doubled. Thus an improvement in the power/speed ratio of the complete analog decoder is expected.

Later, in order to study the dynamic behavior of decoders we modeled a simple LDPC code in the MATLAB Simulink. We tried to linearize the non-linear LDPC decoder around its initial state as a known solution point. At this point, the effect of initialization point was studied and it was deduced that there is no initialization point at which the linearized decoder can function properly.

The implementation of our designed chip comprising of equality nodes and check nodes with different sizes and configuration was an unfortunate in this work. After a time consuming attempt for testing the chip, finally it was figured out that the chip has not been packaged correctly or there might have been a problem in the layout.

Future works in this area may include both practical and theoretical research. As a practical point of view, firstly one may need to open the chips for debugging purposes. Further small tests which can be performed in order to debug the internal problems of the chip would be to check if the digital part functions properly. This can be done by simply measuring the voltages of the address bits which corresponds to the selected node. Moreover, as a future plan of chip design, one can draw the layout of the blocks of equality and check nodes studied in Chapter 4 but this time for the correct H-Matrix of 1024 bits length LDPC code in order to see the effect of wiring capacitance on the delay between the two connected blocks and observe how the modified blocks can practically solve this issue. For this reason, the new chip could be comprised of two designs. In the first design, a basic block of equality node is wired up to a basic block of check node to get the worst case length. In the second design, the diode-connected transistors at the input of these blocks will be replaced by the modified circuits including our designed OTA.

As a theoretical aspect of the future work, following questions should be answered regarding the implementation of a large analog decoder:

- How large an analog decoder should be to actually outperform the large digitally implemented decoders in terms of area and power/speed ratio as well as the uncoded bit length?
- How does the total speed of a decoder relate to its code length?
- What are the limitations of implementing large analog decoders and how can they be mitigated?
- Can we linearize the LDPC decoder around valid codewords as stable solutions?

To answer the above questions, first we need to compare the digital and analog designs of same code. By comparing the BER for a specific SNR as well as the area and power/speed ratio for the two designs we can predict how these values would change for large codes. We may want to simulate both digital and analog designs of a moderate code and see how the result would change.

Thus far we realized that wiring the modules of a large analog decoder could have a significant impact on the decoder speed. There might be other issues related to the complexity of the wirings in large analog decoders that can be studied in the future. Furthermore, our suggested modified current-mirror circuit that was shown to remedy this issue to some extent, may have practical limitations which needs to be carefully analyzed.

In Chapter 6 of this thesis, we successfully modeled a small LDPC code in the MATLAB Simulink environment. In the future, we plan to model a larger LDPC code for which the performance results is available. We need to find the simplest way to modify the existing model for a large code. Consequently, the comparison results would be more helpful in studying the dynamics of decoders.

Moreover, we are going to linearize the LDPC decoder of Chapter 6 around other valid codewords as stable solution points and simulate the linearized decoder versus the non-linear decoder to see if they have same slopes for both up-coming and down-going edges.

# Bibliography

[1] Claude Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, July 1948.

[2] T. M. Cover and J. Thomas,“Elements of Information Theory,” Wiley, 1991.

[3] Claude Berrou, Alain Glavieux and Punya Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Code,” *IEEE Transactions on Communications,* Oct. 1996.

[4] R. G. Gallager, “Low-Density Parity Check Codes,” MIT Press, Cambridge, MA, 1963.

[5] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of binary block and convolutional codes,” *IEEE Transactions on Information Theory*, 42(2):429–445, March 1996.

[6] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Transactions on Information Theory*, 45:399–431, Mar 1999.

[7] R. Pyndiah, “Near optiumum decoding of product codes: Block Turbo codes,” *IEEE Transactions on Information Theory*, 42(8), August 1998.

[8] Thomas J. Richardson, M. Shokrollahi, and Rudiger Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, pages 619–637, February 2001.

[9] Kai He and Gert Cauwenberghs, "Performance of Analog Viterbi Decoding," *IEEE* 1999.

[10] Hyunjung Kimt, Hongrak Son. Jeonwon Lee, In-cheol Kimt and Iiyongsuk Kimt, "Analog Viterbi Decoder for PRML using Analog Parallel Processing Circuits of the CNN," *2006 10th International Workshop on Cellular Neural Networks and Their Applications, Istanbul, Turkey,* 28-30 August 2006.

[11] Andreas Demosthenous and John Taylor, "A 100Mb/s, 2.8V CMOS Current-Mode Analogue Viterbi Decoder," *IEEE Journal of Solid- State Circuits,* July 2002.

[12] Shakiba, Mohammad Hossein and Johns, David A. and Member, Senior and Martin, Kenneth W, "BiCMOS circuits for analog Viterbi decoders," *IEEE Transactions on circuits and system,* 1998.

[13] Wen-Ta Lee, Ming-Jlun Liu, Yuh-Shyan Hwang and Jiann-Jong Chen, "IC Design of a New Decision Device for Analog Viterbi Decoder," *Solid-state circuits, vol.28, pp. 1294-1302,* Dec.1993.

[14] N. Wiberg, H. A. Loeliger, and R. Kotter, "Codes and iterative decoding on general graphs," *European Transactions on Telecommunications*, pages 513–525, Sept./Oct. 1995.

[15] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Transactions on Information Theory*, 42(2):429–445, March 1996.

[16] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, pages 284–287, March 1974.

[17] A. Acampora and R. Gilmore, "Analog Viterbi decoding for high speed digital satellite channels," *IEEE Transactions on Communications,* 1978.

[18] Hui-Ling Lou. Implementing the Viterbi algorithm, "Fundamentals and real time issues for processor designers," *IEEE signal processing magazine,* 1995.

[19] F. Lustenberger, M. Helfenstein, G. S. Moschytz, H. A. Loeliger, and F. Tarkoy, "All analog decoder for (18,9,5) tail-biting trellis code," In *Proc. European Solid-State Circuits Conference (ESSCIRC)*, pages 362–365, Sept.1999.

[20] J. Hagenauer and M. Winklhofer, "The analog decoder," *Proc. International Symposium on Information Theory*, August 1998.

[21] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[22] V. Gaudet N. Nguyen, C. Winstead and C. Schlegel, "A 0.8V CMOS analog decoder for an (8,4,4) extended Hamming code," In *Proc. International Symposium on Circuits and Systems*, volume 1, pages I – 1116–1119. Vancouver,

Canada, May 2004.

[23] Sae-Young Chung, Thomas J. Richardson, and Rudiger Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Transactions on Information Theory*, pages 657–670, February 2001.

[24] C. Winstead, J. Die, R. Harrison, C. J. Myers, and C. Schlegel, "Analog decoding of product codes," In *Information Theory Workshop*, pages 131–133, August 2001.

[25] Chris Winstead, Nhan Nguyen, Vincent C. Gaudet, and Christian Schlegel, "Low-voltage CMOS circuits for analog decoders," In *International Symposiumon Turbo Codes*, pages 271–274. Brest, France, September 2003.

[26] M. Moerz, T. Gabara, R. Yan, and J. Hagenauer, "An analog 0.25$\mu$m BiCMOS Tailbiting MAP decoder," In *International Solid State Circuits Conference*, pages 356–357, February 2000.

[27] C. Winstead, J. Die, W.J. Kim, S. Little, Y.B. Kim, C. J. Myers, and C. Schlegel, "Analog MAP decoder for (8,4) Hamming code in subthreshold CMOS," In *Advanced Research in VLSI*, pages 132–147, March 2001.

[28] F. Lustenberger, "On the Design of Analog VLSI Iterative Decoders," PhD thesis, Swiss Federal Institute of Technology, 2000.

[29] C. Winstead, J. Die, S. Yu, R. Harrison, C. J. Myers, and C. Schlegel, "Analog MAP decoder for (8,4) Hamming code in subthreshold CMOS," In *Proc. International Symposium on Information Theory*, page 330, June 2001.

[30] Chris Winstead, Jie Dai, Shuhuan Yu, Reid Harrison, Chris J. Myers, and Christian Schlegel, "CMOS analog decoder for (8,4) Hamming code," *IEEE Journal of Solid-State Circuits*, pages 122–131, January 2004.

[31] V. Gaudet, "Toward Gigabit-per-second decoding," In *Proc. Analog Decoding Workshop*. Munich, June 2002.

[32] V. C. Gaudet and P. G. Gulak, "A 13.3-Mb/s 0.35-$\mu$m CMOS analog turbo decoder IC with a configurable interleaver," *IEEE Journal of Solid-State Circuits*, 38(11):2010–2015, November 2003.

[33] J. Hagenauer, M. Moerz, and A. Schaefer, "Analog decoders and receivers for high speed applications," In *Proc. Int. Zurich Seminar on Broadband Comm.*, pages 3–1–3–8, 2002.

[34] W. Huang, V. Igure, G. Rose, Y. Zhang, and M. Stan, "Analog Turbo decoder implemented in SiGe BiCMOS technology," *40th DAC Student design contest*, 2003.

[35] W. Huang, V. Igure, G. Rose, Y. Zhang, and M. Stan, "Analog Turbo decoder implemented in SiGe BiCMOS technology," July 2003.

[36] A. Xotta, D. Vogrig, A. Gerosa, A. Neviani, A. Graell-Amat, G. Montorsi, M. Bruccoleri, and G. Betti, "An all-analog CMOS implementation of a Turbo

decoder for hard-disk drive read channels," *Proc. International Symposium on Circuits and Systems*, pages 69–72, 2002.

[37] Alexandre Graell i Amat, Sergio Benedetto, Guido Montorsi, Daniele Vogrig, Andrea Neviani, and Andrea Gerosa, "An analog Turbo decoder for the UMTS standard," In *Proc. International Symposium on Information Theory*, 2004.

[38] M. Perenzoni, A. Gerosa, and A. Neviani, "Analog CMOS implementation of Gallager's iterative decoding algorithm applied to a block Turbo code," In *Proc. 2003 International Symposium on Circuits and Systems (ISCAS '03)*, volume V, pages V – 813–816, May 2003.

[39] Fotios Gioulekas, Michael Birbas, Alex Birbas, George Biliouis, "A High-speed Analog Turbo Decoder with Low-Energy Consumption," *International Symposium on Communications and Information Technologies (ISCIT),* 2004.

[40] Saied Hemati, Amir H. Banihashemi, and Calvin Plett, "An 80-Mb/s 0.18-μm CMOS Analog Min-Sum Iterative Decoder for a (32,8,10) LDPC Code," *IEEE Custom Integrated Circuits Conference,* 2005.

[41] Alexandre Graell i Amat, Daniele Vogrig, Sergio Benedetto, Guido Montorsi, Andrea Neviani, Andrea Gerosa, "Reconfigurable Analog Decoder for a Serially Concatenated Convolutional Code," *IEEE,* 2006.

[42] Alexandre Graell i Amat, Guido Montorsi, Daniele Vogrig, Andrea Neviani and Andrea Gerosa, "Design, Simulation, and Testing of a CMOS Analog Decoder for

the Block Length-40 UMTS Turbo Code," *IEEE Transactions on Communications, vol. 54, no. 11,* November 2006.

[43] Mimi Yiu, Chris Winstead, Vincent Gaudet, and Christian Schlegel, "Design for Testability of CMOS Analog Sum-Product Error-Control Decoders," *IEEE Transactions on Circuits and Systems-II: Express Briefs, vol. 54, no. 8,* August 2007.

[44] Matthieu Arzel, Cyril Lahuec, Fabrice Seguin, David Gnaedig, and Michel Jézéquel, "Semi-Iterative Analog Turbo Decoding," *IEEE Transactions on Circuits and Systems —I: Regular Papers, vol. 54, no. 6*, June 2007.

[45] Billy Tomatsopoulos and Andreas Demosthenous, "A CMOS Hard-Decision Analog Convolutional Decoder Employing the MFDA for Low-Power Applications," *IEEE Transactions on Circuits and Systems —I: Regular Papers, vol.* 55, no. 9, October 2008.

[46] Matthias Moerz, "Analog Sliding Window Decoder Core for Mixed Signal Turbo Decoder," Institute for Communications Engineering, *Munich University of Technology (TUM), D-80290, Germany.*

[47] C. Berrou, P. Combelles, P. Penard, and B. Talibart, "An IC for Turbo-codes encoding and decoding," *In Proc. 1995 IEEE International Solid-State Circuits Conference (ISSCC'95), pages 90–91,* 1995.

[48] M. Bickerstaff, D. Garrett, T. Prokop, C. Thomas, B. Widdup, G. Zhou, C. Nicol, and R.-H. Yan, "A unified Turbo / Viterbi channel decoder for 3GPP mobile

wireless in 0.18$\mu$m CMOS," *Proc. 2002 IEEE International Solid State Circuits Conference (ISSCC'02)*, pages 90–91, February 2002.

[49] Andrew J. Blanksby and Chris J. Howland, "A 690-mW 1-Gb/s 1024-b, rate- 1/2 low-density parity-check code decoder," *IEEE Journal of Solid-State Circuits*, 37(3):404–412, March 2002.

[50] Mark Bickerstaff, Linda Davis, Charles Thomas, David Garrett, Chris Nicol, "A 24Mb/s Radix-4 LogMAP Turbo Decoder for 3GPP-HSDPA Mobile Wireless." *2003 IEEE International Solid-State Circuits Conference,* 2003.

[51] Bruno Bougard1, Alexandre Giulietti, Veerle Derudder, Jan-Willem Weijers, Steven Dupont, Lieven Hollevoet, Francky Catthoor, Liesbet Van der Perre, Hugo De Man, Rudy Lauwereins, "A Scalable 8.7nJ/bit 75.6Mb/s Parallel Concatenated Convolutional (Turbo-) CODEC," *2003 IEEE International Solid-State Circuits Conference*, 2003.

[52] Jagadeesh Kaza and Chaitali Chakrabarti, "Design and Implementation of Low-Energy Turbo Decoders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 12, no. 9,* September 2004.

[53] IbrahimAl-Mohandes and Mohamed Elmasry, "A low-power 5Mb/s Turbo decoder for third-generation wireless terminals," In *Proc. 2004 Canadian Conference on Electrical and Computer Engineering (CCECE'04)*, volume 4, pages 2387–2390, 2004.

[54] Mohammad M. Mansour and Naresh R. Shanbhag, "A 640-Mb/s 2048-Bit Programmable LDPC Decoder Chip," *IEEE Journal of Solid-State Circuits, vol. 41, no. 3,* March 2006.

[55] Vincent C. Gaudet, Naoya Onizawa, Tomokazu Ikeda and Takahiro Hanyu, "3.2-Gb/s 1024-b Rate-1/2 LDPC Decoder Chip Using a Flooding-Type Update-Schedule Algorithm," *IEEE,* 2007.

[56] Xin-Yu Shih, Cheng-Zhou Zhan, and An-Yeu (Andy) Wu, "A 7.39mm$^2$ 76mW (1944, 972) LDPC Decoder Chip for IEEE 802.11n Application," *IEEE Asian Solid-State Circuits Conference November 3-5, 2008 / Fukuoka, Japan,* 2008.

[57] Chris Winstead, "Analog Iterative Error Control Decoders," *Ph.D. dissertation, University of Alberta, Edmonton, AB, Canada,* 2005.

[58] Claude Shannon, "A mathematical theory of communication," *Bell System Technical Journal,* July 1948.

[59] G.D. Forney, "Codes on graphs: normal realizations," *IEEE Transactions on Information Theory*, pages 520–548, February 2001.

[60] MIT University website, Course page, "Chapter 12: The sum-product algorithm," http://www.ocw.mit.edu.

[61] Chhay Kong and Shantanu Chakrabartty, "Analog Iterative LDPC Decoder Based on Margin Propagation," *IEEE Transactions on Circuits and Systems-II: Express Briefs*, 2007.

[62] Thomas J. Richardson and Rudiger Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, pages 599–618, February 2001.

[63] Bradley A. Minch, "MOS Translinear Principle for All Inversion Levels," Propagation *IEEE Transactions on Circuits and Systems-II: Express Briefs, vol. 55, no. 2,* Feburary 2008.

[64] Barrie Gilbert, "A Precise Four-Quadrant Multiplier with Subnanosecond Response," *IEEE Journal of Solid-State Circuits*, December 1968.

[65] Peter R. Kinget, "Device Mismatch: An Analog Design Perspective," *IEEE International Symposium on Circuits and Systems, ISCAS 2007. Volume , Issue , 27-30 ,Page(s):1245 – 1248,* May 2007.

[66] C. Bartolozzi, S. Mitra, G. Indiveri, "An ultra low power current-mode filter for neuromorphic systems and biomedical signal," *IEEE,* 2006.

[67] L. Barranco, S. Gotarredona, "Current Mode Techniques for Sub-pico-Ampere Circuit Design," *Analog Integrated Circuits and Signal Processing,* 2004.

[68] Saied Hemati and Amir H. Banihashemi, "Dynamics and Performance Analysis of Analog Iterative Decoding for Low-Density Parity-Check (LDPC) Codes," *IEEE Transactions on Communications. vol. 54, no. 1,* January 2006.

[69] Marcelj.M Pelgrom, AAD C.J. Duinmaijer, and Anton P.G. Welbers, "Matching Properties of MOS Transistors," *IEEE Journal of Solid State Circuits, vol. 24, no. 5,* October 1989.

[70] Hans-Andrea Loeliger, Felix Tarköy, Felix Lustenberger and Markus Helfenstein, "Decoding in Analog VLSI," *IEEE* Communications Magazine, April 1999.

[71] Shahaboddin Moazzeni, Glenn E. R. Cowan, "Application of Active Current Mirrors to Improve the Speed of Analog Decoder Circuits," *IEEE* International Midwest Symposium on Circuits and Systems, August 2009.