LOW-POWER HIGH-SPEED DSP ARCHITECTURE FOR MAGNETIC DISK PRML READ CHANNEL

by

See-Hoi Caesar Wong

Memorandum No. UCB/ERL M93/72

8 October 1993

LOW-POWER HIGH-SPEED DSP ARCHITECTURE FOR MAGNETIC DISK PRML READ CHANNEL

by

See-Hoi Caesar Wong

Memorandum No. UCB/ERL M93/72

8 October 1993

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Low-Power High-Speed DSP Architecture for Magnetic Disk PRML Read Channel

Abstract

As a result of the rapid development of disk technology, several orders of magnitude improvement in storage density has been made on magnetic disk systems in the last few decades. Currently state-of-the-art disk systems can store up to 350 millions bits per square inch. By the end of this decade, it is predicted that the maximum storage density will be increased to 10 billions bits per square inch. To take advantage of this tremendous increase in disk storage density, there must be a comparable improvement in processing power of the associated electronics. In this report, a parallel DSP architecture, which provides the key functions required in the magnetic disk read channel employing Class IV Partial Response Signaling, will be presented.

To greatly enhance the throughput of the DSP, the proposed architecture creates four time-interleaved channels which allow parallel processing of signal. In certain applications such as portable computers, it is essential to cut down the power dissipation of the disk drive electronics. This report will discuss the advantage of using the parallel architecture from a low-power design point of view, despite that it consumes more silicon area. A prototype with an adaptive equalizer and a Class IV Partial Response Viterbi decoder was fabricated in a standard MOSIS 1.2μm CMOS process. The total silicon area is 57,000 mil^2 (36.6 mm^2). Experimental results show that the chip can achieve a throughput of 50MBits/sec with a power consumption of 70.5mW only.

Acknowledgments

I would like to express my sincere appreciation to my advisor, Professor Paul R. Gray, for his continuous guidance and support. Having the opportunity of working in his research group is certainly an excellent first step for my future career. I would also like to thank Professor Robert W. Brodersen for his help and suggestions on this report.

This research would not have been possible without the dedicated work of my partners, Gregory Uehara and Jacques Rudell. I have to thank Gregory for answering the thousands of questions that I posted to him, especially when I first picked up this project. More importantly, his patience, kindness, and concern for the problems of other people have changed my attitude towards people and life in a positive way. Also, I would like to thank Jacques for being a very nice and responsible partner to me. He generously picked up most of the responsibilities when I was busy with my personal business.

Many other fellow students are very deserving of thanks. My colleagues in Professor Gray's research group, especially Cormac Conroy, Robert Neff, and Thomas Cho, give me useful help and enlightening thoughts throughout my stay here. Special thanks to Edward Liu, Mansun Chan, Kelvin Hui, and Jonathan Hui for bringing fun and relaxing times into my busy life through our regular gathering. They also kindly dedicate time and effort in helping me with my personal problems.

Next I have to mention two very special friends of mine, Grace Tong and Bonnie Han. In addition to being a nice course-work project partner, Grace's cute and interesting personality has made my monotonous life much more colorful. She simply makes me appreciate this world in a way that I never did before. I really feel that I owe a debt of

gratitude to her for her kindness, and I need to apologize for being too demanding on her occasionally. During the times of emotional upset, Bonnie can always listen to me and give me sincere advice. Her caring and sweet presonality makes me feel comfortable all the time. I wish I could have her as a good friend for the rest of my life.

Finally I have to thank my parents, sister, and brother for their endless support and encouragement. I want to share my happiness with all of them in the future years to come.

This research was sponsored by the National Science Foundation and California MICRO program. The prototype was fabricated by MOSIS.

Table of Contents

Chapter 1 Introduction 1

1.1 Background and Motivation 1

1.2 Thesis Organization 2

Chapter 2 Partial Response Maximum Likelihood (PRML) Detection 3

2.1 Channel Model 3

2.2 Sampled Data Detection 5

2.3 Application of PRML to the magnetic disk read channel 7

2.3.1 Partial Response Signaling 7

2.3.2 Maximum Likelihood Detection 9

Chapter 3 Parallel DSP Architecture 10

3.1 Introduction 10

3.2 Pipelining and Parallelism 12

3.2.1 Introduction 12

3.2.2 Advantages of Pipelining Limited by Design Constraints 13

3.2.3 Parallelism over Pipelining 16

3.3 Derivation of the Parallel DSP Architecture 19

Chapter 4 Physical Design of DSP 24

4.1 Layout of Key Functional Blocks and Data Flow 24

4.1.1 Chip Plan 24

4.1.2 Bussing of Coefficients Through the Channels 25

4.1.3 Input Signal Flow 27

4.2 Circuit Implementation of Key Functional Blocks 28

4.2.1 Carry Save Multiplier 28

4.2.2 Carry Select Accumulator 32

Chapter 5 Viterbi Decoder 41

5.1 Introduction 41

5.1 The Different Metric Algorithm 41

5.2 Circuit Implementation 47

Chapter 6 Experimental Results 56

6.1 Introduction 56

6.2 Design of Test Board 56

6.3 Summary of Experimental Results 59

6.3.1 Power Consumption verse Operating Frequency 59

6.3.2 Power Consumption verse Supply Voltage 60

6.3.3 Discrepancies between Simulation and Experimental Results 61

6.3.4 Key Specifications of DSP 63

Chapter 7 Conclusion 64

7.1 Summary of Research Results 64

7.1.1 Parallel Adaptive Equalizer Architecture 64

7.2 Future Directions 65

References 67

Chapter 1

Introduction

1.1 Background and Motivation

In a broad range of communication applications, such as high-speed transmission on twisted pair and wireless communications, sophisticated signal processing techniques are required to enhance detection accuracy. The advancement of VLSI circuits makes possible the implementation of complex signal processing blocks such as digital adaptive equalizer, which are usually composed of thousands of transistors. The tremendous improvement in processing powers of personal computers and workstations in the last ten years brings an enormous demand for the storage and processing extremely large amounts of digital data. Since 1967, the use of magnetic disks has become more and more popular as a result of the rapid development of disk technology. In less than 40 years, several orders of magnitude improvements in cost, capacity, and performance have been made on these magnetic storage systems [1]. As the areal and linear densities increase rapidly, there must be a match in the development of the data detection techniques that serve to retrieve the stored data. For the last 30 years, disk drive engineers have been using the simple Peak Detection method. However, as the storage density continues to increase, its performance will be greatly degraded because of the effects of Inter-Symbol Interference (ISI) and other non-idealities.

To overcome this problem, a new method, the Partial Response Maximum Likelihood (PRML) detection method has been proposed. In the last few years, most research works published in this area emphasized the system-level aspects of a PRML system. However, research in the implementation of such system is still in a relatively "infant" stage. This report describes a new parallel DSP architecture that provides functions

required in the magnetic disk read channel employing Class IV partial response signaling. These functions include adaptive equalization and sequence detection. The key objective of this research project is to achieve a throughput of 100Mbits/sec at a supply voltage of 3.3V with a power consumption of less than 250mW. In addition to this, another important goal is to investigate the impacts of more sophisticated circuit design techniques and fabrication technology on the design of a DSP on an architectural as well as system level.

1.2 Organization of the Report

This report is organized as follows. Chapter 2 gives a brief tutorial on Partial Response Maximum Likelihood (PRML) Detection scheme. It also explains the reason of switching from the traditional Peak Detection method to PRML. Chapter 3 derives the proposed parallel DSP architecture. Comparisons of speed, power, and silicon area are made between the parallel and the pipelined implementation of the proposed DSP. These comparisons lead to the conclusion that the parallel approach is preferable in this case. Chapter 4 describes the physical design of the chip. The organization of the major functional blocks will be described. The circuit implementation of these functional blocks will also be shown. Chapter 5 is a detailed description of the Difference Metric Algorithm used for the Viterbi decoder as well as its actual circuit implementation. Chapter 6 is a brief discussion of the design of the test board as well as the key experimental results. Chapter 7, the last chapter, is a conclusion that summarizes the key points of this research project and the direction for future work in this area.

Chapter 2

Partial Response Maximum Likelihood (PRML) Detection

2.1 Channel Model

The magnetic storage system is in many ways analogous to a data transmission system. A typical magnetic storage channel consists of the write head, the magnetic medium itself, and the read head. These components are similar to the transmit filter, the channel, and the receiver in a digital communication system. Figure 2.1a shows a typical magnetic storage channel, while a typical data transmission channel is shown in Figure 2.1b [2].

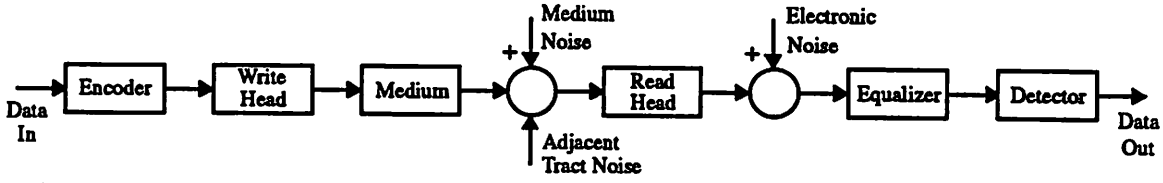


Figure 2.1a Magnetic Storage Channel

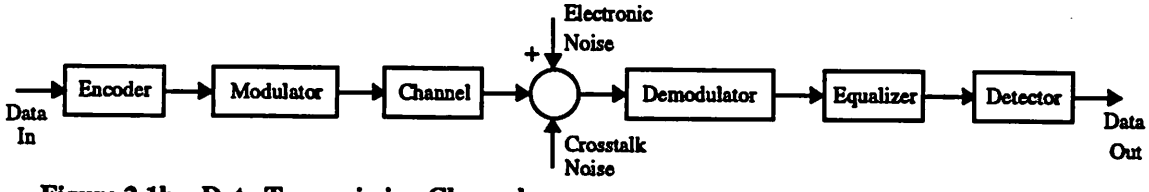


Figure 2.1b Data Transmission Channel

The major difference between these two channels exists between the modulator in the data transmission channel and the write head in the magnetic storage system [2]. In a data transmission system, modulation schemes can be used to increase the effective data rate. For example, we can increase the number of levels in a PAM system if a higher bit rate is desired. The channel bandwidth requirement remains the same because the symbol

rate does not change. The only penalty is that the transmission power has to increase to keep the error probability unchanged. On the other hand, because of hysteresis effect, only two levels (+1 and -1) can be transmitted in a magnetic disk read channel.

At this point, it is helpful to derive a quantitative model for the magnetic disk read channel. Assume that the natural step response of the read channel is Lorentzian [3]:

$$s(t) = \frac{A}{1 + (2\frac{t}{PW_{50}})^2} \tag{2.1}$$

Therefore the pulse (dibit) response is given by

$$p(t,T) = \frac{[s(t) - s(t-T)]}{2} \tag{2.2}$$

Notice that p(t,T) is a function of T. This dependence is important because it shows that the only way to increase linear density is to decrease T [4]. With (2.2), the ideal output of the read channel can be written as

$$x(t) = \sum_k a_k p(t - kT) \tag{2.3}$$

As in a data transmission system, the actual output is corrupted by several noise sources and is given by [5]:

$$y(t) = \sum_k a_k p(t - kT) + n_e(t) + \alpha_m (n_m(t) \bullet h(t)) + \alpha_o n_o(t) \tag{2.4}$$

where $n_e(t)$ = electronic noise, $n_m(t)$ = media noise, α_m = weighting parameter for media noise, $n_o(t)$ = Intertract Interference (ITI), α_o = weighting parameter for $n_o(t)$, and h(t) is the channel impulse response. The electronic noise is additive white-Gaussian and is

caused by the signal amplifier in the read head. Medium noise comes from several sources. The read channel exhibits nonlinear effects that become more pronounced as the transition-spacing decreases. In addition, the channel exhibits random gain fluctuations and spectral variation as the position of the head varies with respect to the medium. Finally, medium thickness variations around the disk also give rise to medium noise. The ITI is the result of recording head positioning error [5]. Unlike electronic and medium noise, ITI is neither white nor Gaussian; as a result, accurate characterization of ITI is very important for designing good data detection systems in a ITI-dominated noise environment.

2.2 Sampled Data Detection

The well known Peak Detection (PD) method has traditionally been used to detect the read back signals received by the read head. In fact, all the commercially available disk drive systems today use peak detectors. Figure 2.2 shows a typical PD system [2]. As a result of the use of NRZI encoding, either a positive or negative step decodes to 1, and no step decodes to 0. The head output, y(t), is rectified and passed through a threshold device, which is used to enable a zero-crossing detector. When the zero-crossing detector is enabled and its input (which is the derivative of the channel output, $\dot{y}(t)$) passes through zero, the zero-crossing detector output will be a "1". This indicates the presence of a peak. Also, the clock rate is derived from the zero-crossings of the time derivative, $\dot{y}(t)$.

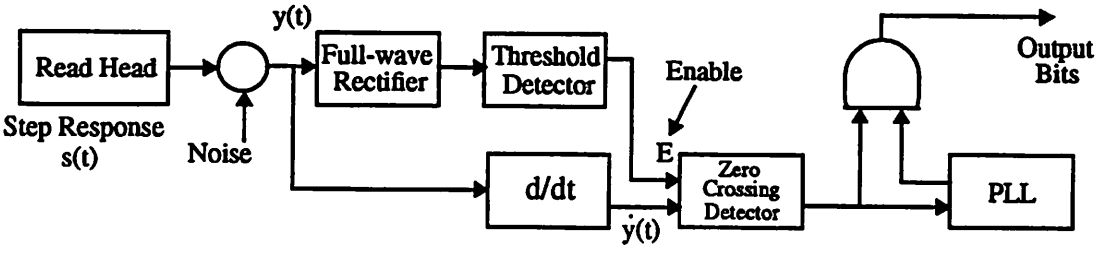


Figure 2.2 Block Diagram for a Typical Peak Detection System

The PD system described above works well if the channel output is free of ISI. However, as shown in (2.2), the only way to increase the linear density is to decrease T (increase clock rate), because the pulse (dibit) response p(t,T) is a function of T. This will, however, lead to severe ISI. Consequently, as the linear density increases, two major types of errors occur in a PD system [2]: (1) "Missing-bit" errors occur when ISI and/or noise cause the output signal peaks to drop below a threshold, and (2) "Peak-Shift" errors occur when the point at which the signal derivative crosses zero is shifted into an adjacent timing window.

One effective solution for these errors is the use of the Run-Length-Limited (RLL) codes. RLL codes are characterized by two parameters (d,k) specifying, respectively, the minimum and maximum numbers of symbols 0 between consecutive symbols 1 in the allowable code sequence. In other words, the "d" constraint increases the minimum transition spacing to improve SNR and reduce missing-bit errors. However, since the code rate is less than unity, the clock rate must be increased to keep the user information rate constant. This will inevitably lead to more peak-shift errors. As a result, the lowest bit error rate is achieved with a trade-off. The most popular (d,k) constrained codes are the 1/2-rate (2,7) and the 2/3-rate codes [6].

The rapid pace of development of computer technology brings a large demand for disk systems with an operating speed on the order of 100 Mbits/sec. As mentioned, the maximum attainable operating speed of a PD system is limited by ISI, and it is believed (by most disk drive engineers) that the best a PD system can do is only about 60 Mbits/sec. Therefore, a new detection scheme must be used to meet the impending need. For this reason, most of the recent research efforts in this area have been directed towards the use of sampled-data detection method. The hope in using sampled data detection is that, with the application of sophisticated digital signal processing techniques, the undesirable effects of ISI can be reduced.

2.3 Application of PRML to the magnetic disk read channel

2.3.1 Partial Response (PR) Signaling

The objective of performing equalizaiton in a magnetic disk read channel is to combat the effect of ISI, thereby increasing storage density. Theoretically, to have no ISI, it is necessary to have a flat spectrum at the output of the equalizer, therefore

$$\sum_{k=-\infty}^{\infty} |P(j(\omega + kT))|^2 = constant \tag{2.5}$$

However, it comes with the undesirable noise enhancement effect. This is especially true for the magnetic disk read channel, because it is inherently PR due to a DC null and high frequency roll off. Therefore, the output spectrum cannot be equalized to a flat spectrum; otherwise there will be infinite noise enhancement. This alone makes the use of the zero-forcing equalizer impractical. Therefore, instead of equalizing to the ideal flat output spectrum, an appropriate coding scheme should be selected such that the output SNR will be maximized at the data rate of interest. This is the motivation for using PR signaling. The writing current i(t) in the NRZ recording system can be written as [7]:

$$i(t) = \sum_{k=0}^{\infty} (2x_k - 1)\, u(t - kT) \tag{2.6}$$

where $\{x_k\}$ ($b_k = 0,1$) is the binary input sequence and $u(t)$ is a rectangular pulse of duration T seconds. From (2.6), it is clear that i(t) is normalized to 2 saturation levels (+1 and -1). Due to the inherent differentiation of the stored magnetization patterns in the read process, the readback voltage at the sampling instants in the absence of noise and ISI is given by:

$$v_{rb}(nT) = 2y_n \tag{2.7}$$

where $y_n = x_n - x_{n-1}$ for $n \geq 1$ and $y_n = 0$ for $n = 0$. It is clear that $\{y_n\}$ is a three-level sequence (-1, 0, and +1). Thus the recording channel can be regarded as a PR channel with a discrete transfer function [7]:

$$G_o(D) = 1 - D \tag{2.8}$$

Minimum bandwidth systems with no ISI are physically unrealizable. However, by introducing a controlled amount of ISI, we can practically implement systems transmitting at the Nyquist rate. Figure 2.3a shows the impulse response h(t) to a step change in the magnetization pattern, and T is the sampling period. In Figure 2.3b, the new time axis t_{new} is defined as [7]:

$$t_{new} = t + \frac{1}{2} T_{new} \tag{2.9}$$

where $T_{new} = \frac{2}{3} T$. Furthermore the function $f(t_{new})$ is given by

$$f(t_{new}) = h(t + \frac{1}{2} T_{new}) \tag{2.10}$$

then, as can be seen from Figure 2.3b, $f(nT_{new}) = 1$ for n = 0, 1 and $f(nT_{new}) = 0$ otherwise. Note that the values at n = 0, 1 are normalized. Therefore the new transfer function is given by

$$F(D) = \sum_{n=-\infty}^{\infty} f(nT_{new}) D^n = 1 + D \tag{2.11}$$

Combining (2.8) and (2.11), the transfer function of the whole system is

$$G_1(D) = G_0(D) F(D) = 1 - D^2 \tag{2.12}$$

which is the well known Class IV Partial Response system [7].

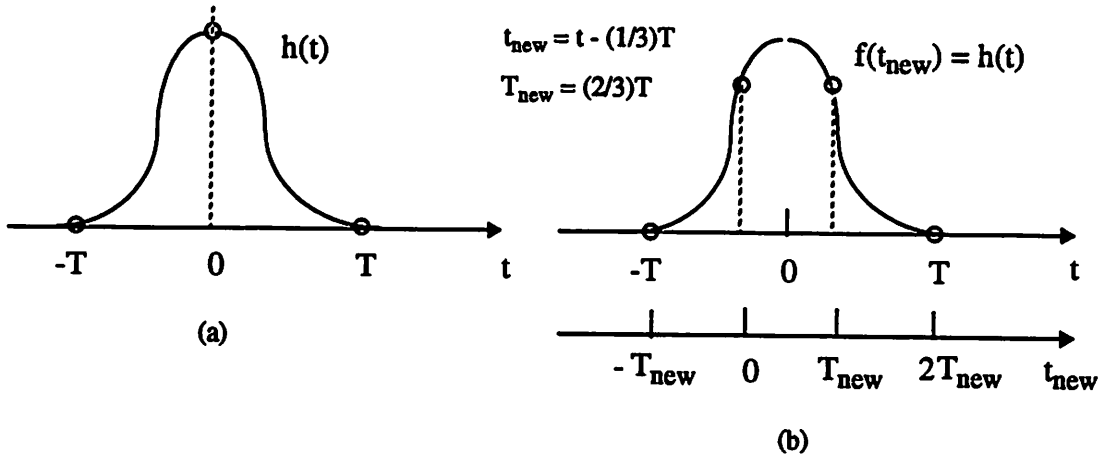


Figure 2.3b Sampling in Class IV Partial Response Channel

2.3.2 Maximum Likelihood (ML) Detection

In a PR coding system, since it is know that a controlled amount of ISI has been introduced, this known correlation between data samples can be utilized by the decoder to achieve a better detection probability by considering a sequence of symbols. Therefore the Maximum Likelihood (ML) detection method is very useful in this case. Since redundancy level(s) are added, there is a loss in SNR when PR coding is used. However, this SNR loss can be recovered with a ML detector. Thus, theoretically, the PRML system can approximate the match filter bound very closely [8]. One more advantage of using the PRML system is that adaptive equalization techniques can be readily applied. One simple approach, which is adopted in the proposed DSP, is to use an adaptive LE followed by the ML detector.

Chapter 3

Parallel DSP Architecture

3.1 Introduction

Figure 3.1 shows the block diagram for a general read channel DSP. The major components include the adaptive equalizer and the Class IV Partial Response (PR) Viterbi decoder. Timing recovery circuitry is also required to acquire timing information for the front end of DSP. In the proposed DSP, since the emphasis is placed on implementing a low-power high-speed signal path which consists of the equalizer and the decoder, this timing recovery block is not included.

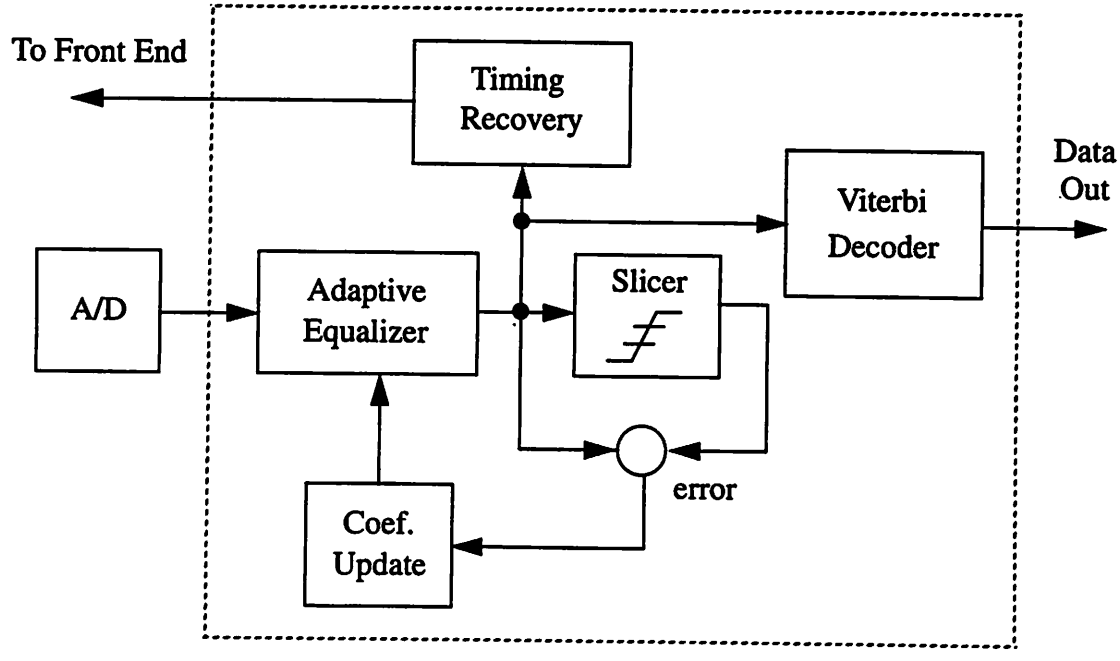


Figure 3.1 Block Diagram of Read Channel DSP

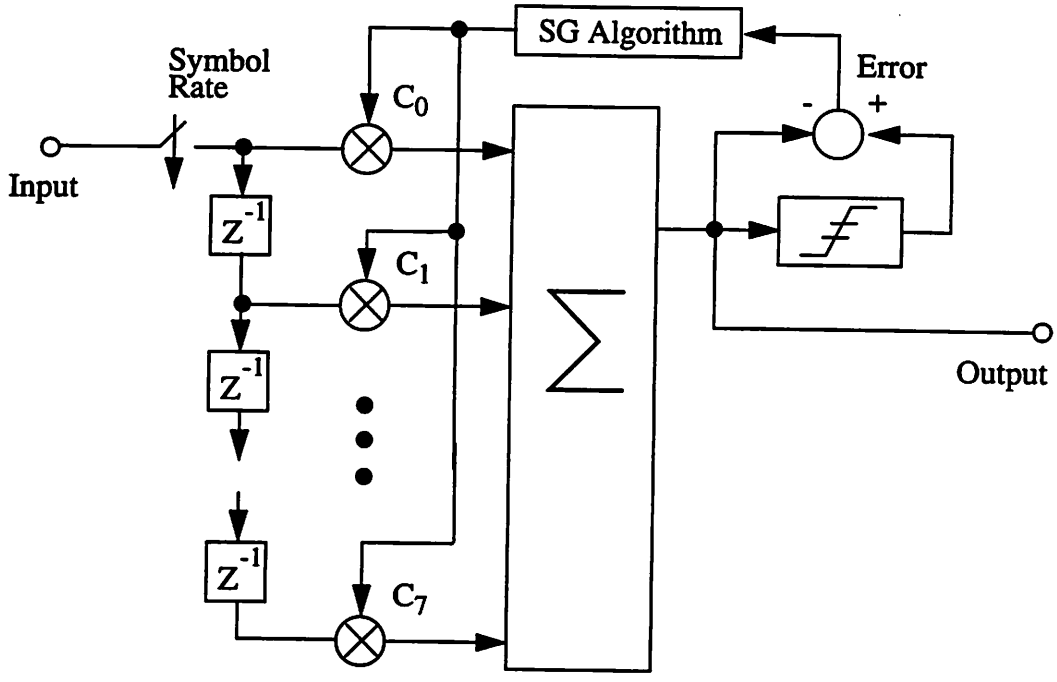


Figure 3.2 Basic Structure of Adaptive Equalizer

As mentioned in Chapter 2, as a result of the 3dB coding gain provided by the Viterbi decoder, a PRML system can approximate the match filter bound closely. In the proposed DSP, a simple approach in which an adaptive equalizer followed by a Viterbi decoder is used. Figure 3.2 shows the basic structure of the adaptive equalizer. It is basically an 8-tap FIR filter with its coefficients updated using the Stochastic Gradient (SG) algorithm. The SG algorithm updates the coefficients in such a way so that the mean square error at the output of the filter is minimized. In a magnetic disk, data bits are arranged in concentric tracks. For the usual case of constant data rate, the effect of ISI will be more limiting at the inner diameter (ID), because transition-spacing at the ID is closer than that at the outer diameter [2]. This is the main motivation for performing adaptive equalization in a discrete-time read channel. Another key component, the Class IV PR Viterbi decoder, serves to decode the channel output symbols back to the original input bits. In other words, it performs the $1/(1-D^2)$ operation. As stated in Chapter1, the target throughput of the proposed DSP is 100 Mbits/sec, while the target power consumption

is 300mW or below. The design of blocks performing complex computations such as multiplication and accumulation with a throughput around 100MHz is difficult. This can be achieved only at the expense of large power consumption. To overcome this problem, two popular approaches used to enhance the throughput of a signal path, pipelining and parallelism, were considered. A detailed analysis and comparison between these two approaches is given in the following section.

3.2 Pipelining and Parallelism

3.2.1 Introduction

As a quick review, a brief explanation of the principles of pipelining and parallelism will be given first. Figure 3.3 shows how to use pipelining and parallelism to build a 50MHz 6 bits by 6 bits multiplier. In other words, all the logic operations required in one multiplication needed to be done within 20ns(T). The key assumption is that, with the best fabrication technology available, the highest achievable operating speed is 25MHz (1/2T). In the parallel implementation, two identical 25MHz (1/2T) multipliers with input latches are used. Multiplexing and Demultiplexing blocks are required at the input and output respectively to distribute the data to the right signal path. Note that the input latches are still being clocked at 25 MHz (1/2T). In other words, each multiplier still has 40ns (2T) to perform one multiplication. In the case of using the pipelined implementation, one 25MHz multiplier is broken into two equal parts, and a set of pipeline latches is inserted in between them to synchronize the data. These two parts are equal in the sense of lengths of the critical paths, rather than physical dimensions. Different from the parallel implementation, both the input and the pipeline latches are being clocked at 50MHz (1/T), the target operating speed. In this case, the first half of the multiplier has 20ns (T) to finish the computations and then transfer the data into the second half of the multiplier through the pipeline latches. As a result, the hardware apparently still has 40ns (2T) to

process all the required logic operations. Judging from this simple example, it seems that the pipelined is preferable, since the parallel approach requires almost twice as much hardware as the pipeline approach. Because of this observation, an early effort was made to look into the possibility of implementing the proposed DSP using the pipeline approach.

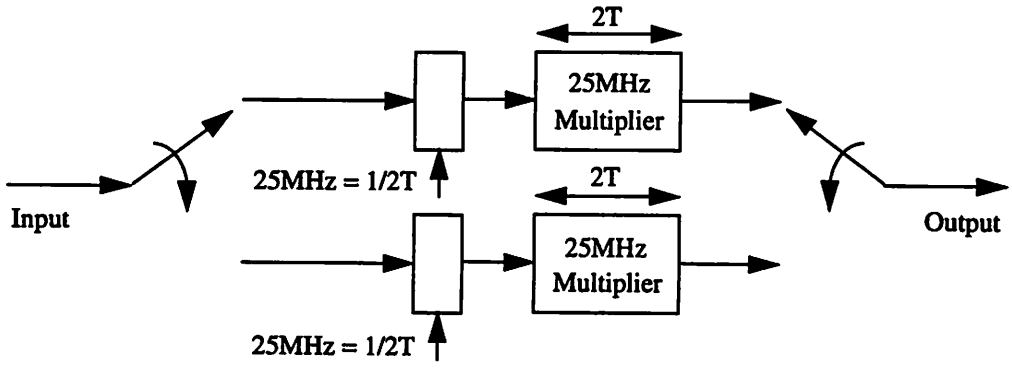


Figure 3.3a Parallel Implementation of a 50MHz Multiplier

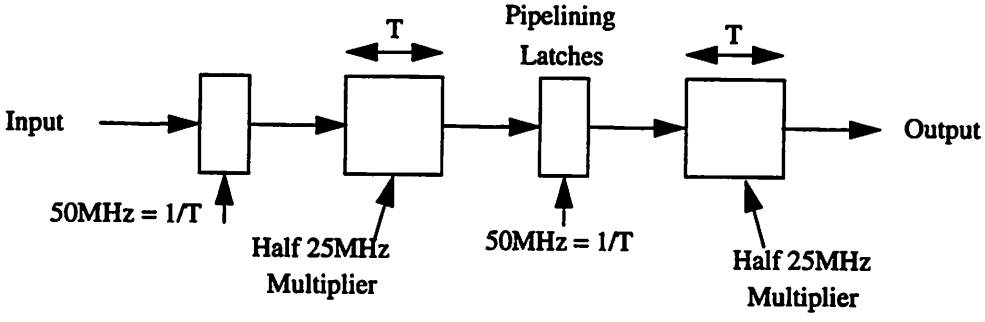


Figure 3.3b Pipelined Implementation of a 50MHz Multiplier

3.2.2 Advantages of Pipelining Limited by Design Constraints

Because of the advancement of VLSI technology, integrated circuits with thousands or even millions of transistors are feasible to manufacture. The sizes of these state-of-the-art integrated circuits often make optimization of performance at the transistor level impossible. For this reason, most DSP and other digital circuit designers have shifted

their attention to the exploration of new design methodology at the architecture level. Among the most popular approaches being utilized today, pipelining is one of the most fundamental and simplest to apply. The basic principle of the application of pipelining for increasing hardware throughput is demonstrated in the example shown in section 3.2.1. From that example, one can observe that the only extra hardware needed are the pipeline latches. These latches are cheap in terms of area and power, and they are very simple to design, especially for high speed applications. The only drawback is that more latency is added to the signal path.

In the Department of Electrical Engineering and Computer Sciences of UC Berkeley, researchers led by Professor Brodersen and Rabaey are looking at the impact of architectural design on the speed and power consumption of digital hardware. In one of their recent publications [9], the following results were demonstrated (see Figure 3.4a). For Figure 3.4a, the key assumptions are that the target throughput is fixed while the supply voltage can be freely adjusted to minimize the power. Data point A represents the case in which no pipelining is being applied. Therefore, the target throughput is achieved by some brute-force approaches such as sizing up the transistors, raising the supply voltage, or using a better fabrication process. The second data point is obtained by lowering the supply voltage so that the target throughput can just be achieved by using one pipeline stage. Note that all other conditions such as the design of the hardware and the quality of the fabrication process remain the same, since the emphasis is placed on investigating the use of pipelining to lower power consumption as the supply voltage is reduced. The third and fourth data points are obtained in the same way. The reduction of power consumption can be explained as follows. For digital CMOS integrated circuits, power will be consumed only when there are logic transitions. In other words, they only consume dynamic power, which is given by

$$Power = C_{total} V_{DD}^2 f \tag{3.1}$$

where C_{total} is the total parasitic capacitance needed to be charged or discharged, and f is the operating frequency. For long channel devices, the maximum attainable speed is proportional to the magnitude of the charging (or discharging) current available, which is, in turn, proportional to $(V_{DD} - V_t)^2$. However, the rapid development of VLSI technology makes the minimum attainable dimensions become smaller and smaller. As a result of velocity saturation, the magnitude of the current available in a short channel device is proportional to $(V_{DD} - V_t)$ instead. Therefore, the maximum attainable speed can be written as

$$Speed = k(V_{DD} - V_t) \tag{3.2}$$

where k is a constant. From (3.1) and (3.2), it is clear that as the supply voltage is reduced, the power consumption decreases in a faster rate than the speed. This explains why the power consumption keeps going down in the first four data points. However, the power consumption goes up again at the fifth and sixth data points. This is due to the fact that the number of pipeline latches increases exponentially with the number of pipeline stages. In other words, the overhead introduced by the pipeline latches limits the usefulness of the pipeline approach.

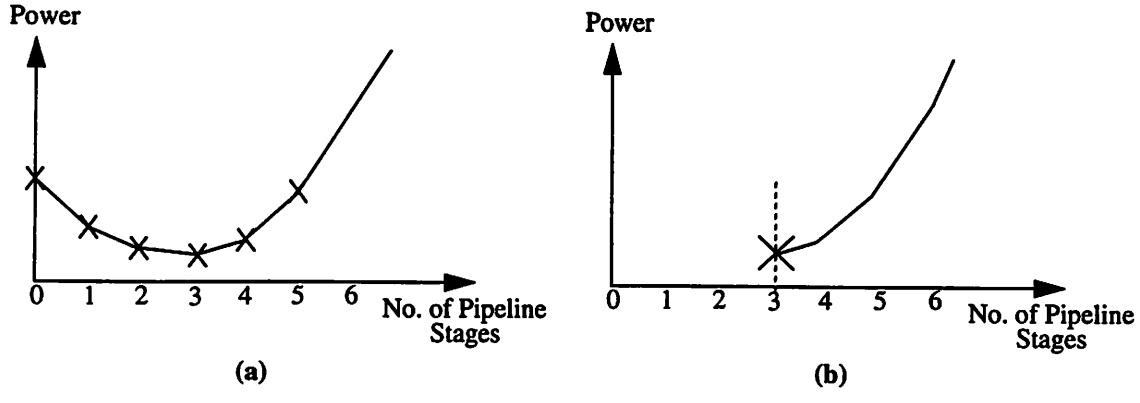


Figure 3.4 Advantages of Pipelining Limited by Design Constraints

For the proposed DSP, however, pipelining is not as useful as described above. This is mainly because it has different design constraints. The key difference is that the magnitude of the supply voltage is fixed at 3.3V in this case. Recall that the target throughput of the proposed DSP is 100Mbits/sec. According to simulation results, this is not achievable when pipelines is not being applied. Assume that with the 1.2um CMOS standard MOSIS process, the fastest multiplier available can be operated at 25MHz only. Therefore, at least three pipeline stages are required to attain the target throughput. Since the supply voltage is fixed at 3.3V, when extra pipeline stages are added to increase the throughput, the power consumption goes up. This argument is recapitulated in Figure 3.4b. From this argument, it is clear that pipelining is not as useful as in other cases. Because of this, the parallel architecture is studied again, and a comparison between the pipeline and the parallel approach is made in terms of power consumption.

3.2.3 Parallelism over Pipelining

To compare the pipeline and the parallel approach in a more general and systematic sense, a more accurate estimation of the power consumption is required. For the sake of simplicity, the readers can still refer to the example of designing a 50MHZ multiplier mentioned before. However, all the equations shown in this section will be written in terms of symbols such as T (period) and E (Energy) to avoid the loss of generality.

One of the advantage of CMOS digital integrated circuits is the absence of static current. Therefore, they consume dynamic power only. As given in (3.1), dynamic power is equal to the product of the total capacitive loading, the square of the supply voltage, and the operating frequency. It can be given more generally as:

$$\textit{Dynamic Power} = \textit{Energy/transition X Operating Frequency} \qquad (3.3)$$

In the example of designing a 50MHz multiplier (see Figure 3.3), the power consumptions for both the parallel and pipelined implementation can be computed by using (3.3). For the parallel implementation, since two 25MHz multipliers are being used in parallel, the power consumption can be written as

$$POWER_{parallel} = 2\ X\ \{\ E_{latch} + E_{multiplier}\ \}\ X\ 1/2T \qquad (3.4)$$

where E_{latch} and $E_{multiplier}$ are the energies consumed by the input latches and the 25MHz multiplier in each transition respectively. Similarly, the power consumption for the pipelined implementation can be written as

$$POWER_{pipeline} = \{\ E_{latch} + E_{multiplier} + E_{p\text{-}latch}\ \}\ X\ 1/T \qquad (3.5)$$

where $E_{p\text{-}latch}$ is the energy consumed by the input latches in one transition. Note that E_{latch} and $E_{multiplier}$ are the same as those in (3.4), since the original 25MHz is not modified at all. By comparing (3.4) and (3.5), it is clear that the pipelined implementation consumes more power than the parallel implementation by $E_{p-latch} \times 1/T$. Therefore, from a pure low-power design point of view, the parallel approach is preferable to the pipelined approach.

Spice simulations were performed to verify the idea put forward in the previous paragraph. The results are shown in Table 3.1. With the 1.2um CMOS double Poly Orbit process, the fastest multiplier available can be operated at 25MHz. This corresponds to the case of zero pipeline stage in which a power of 4.82mW (at 25MHz) is consumed. If the multiplier is pipeline once, then the power consumption is 8.5mW (at 33.3MHz). Different from the example shown in Figure 3.3, the throughput cannot be doubled when a pipeline stage is added. This is due to the finite set-up time in the pipeline registers. A maximum speed of about 45MHz can actually be achieved. However, since the final goal is to achieve a throughput of 100MHz, it makes more sense to clock the pipelined multiplier at the lowest possible rate to get a better power figure. The data for two and three

Number of Pipeline Stages	Speed/ Period	Energy/transition (pJ)					Power (mW)	Components (1)Adders (2)And (3)Registers	Area
		Adders	Registers	And	Clock	Total			
0	25MHz/ 40ns	96.7	50.8	30.7	14.7	193	4.82	(1) 33 (2) 36 (3) 12	1
1	33.3MHz/ 30ns	85.6	98.2	30.8	39.7	254	8.50	(1) 33 (2) 36 (3) 30	1.15
2	50MHz/ 20ns	71.0	156.3	27	75	330	16.5	(1) 33 (2) 36 (3) 53	1.35
3	66.7MHz/ 15ns	68.4	213.8	25.2	116.5	424	28.3	(1) 33 (2) 36 (3) 75	1.51

Table 3.1 Simulation Results for Pipelined Multipliers

Way to Achieve 100MHz	Power	Area
No pipelining (Parallel - 4)	4.81 X 4 = 19.28	4 X 1 = 4
Pipelined Once (Parallel -3)	8.5 X 3 = 25.5	3 X 1.15 = 3.45
Pipelined Twice (Parallel -2)	16.5 X 2 = 32	2 X 1.35 = 2.7

Table 3.2 Comparisons among Implementations with Various Degrees of Pipelining

pipeline stages were also obtained. In Table 3.1, it is interesting to observe that the energies consumed by the adders, the registers, and the AND gates decrease as the number of pipeline stages increases. This is due to the fact that the pipeline latches tend to synchronize the signal flowing through the multiplier, and this lowers the occurrences of glitch or unneccessary transition in the hardware. Another important observation is that the number of pipeline registers increases tremendously as the number of pipeline stages increases.

The power figures shown in Table 3.1 cannot be compared, since they were obtained at different speeds. However, a fair comparison can be made if parallel implementations of each of them are considered to make up to a fixed throughput. In this case, the target throughput of the proposed DSP (100MHz) will be used. The results are shown in Table 3.2. Note that the power consumption goes up with the number of pipeline stages. It can also be seen from Table 3.2 that the major disadvantage of the parallel approach is that it requires significantly larger die sizes. Combining all these results, the conclusion is that, if the extra area overhead caused by the parallel approach can be tolerated, then the parallel approach is preferable to the pipelined approach for the proposed DSP.

3.3 Derivation of the Parallel DSP Architecture

In the last section, the key decision of using four parallel 25MHz multipliers to simulate one 100MHz multiplier is made. At this point, the parallel DSP architecture will be derived step by step. Figure 3.4 is the same as Figure 3.2 with the exception that the 100MHz multiplier in each tap is replaced by four parallel 25MHz multipliers.

The bottleneck caused by the 100MHz multiplier has now been solved. However, accumulating the eight products coming out from the eight taps of the FIR filter at 100MHz is also very difficult to achieve. A simple but effective solution to this problem is to break the 100MHz accumulator up into 4 parallel 25MHz accumulators, as shown in

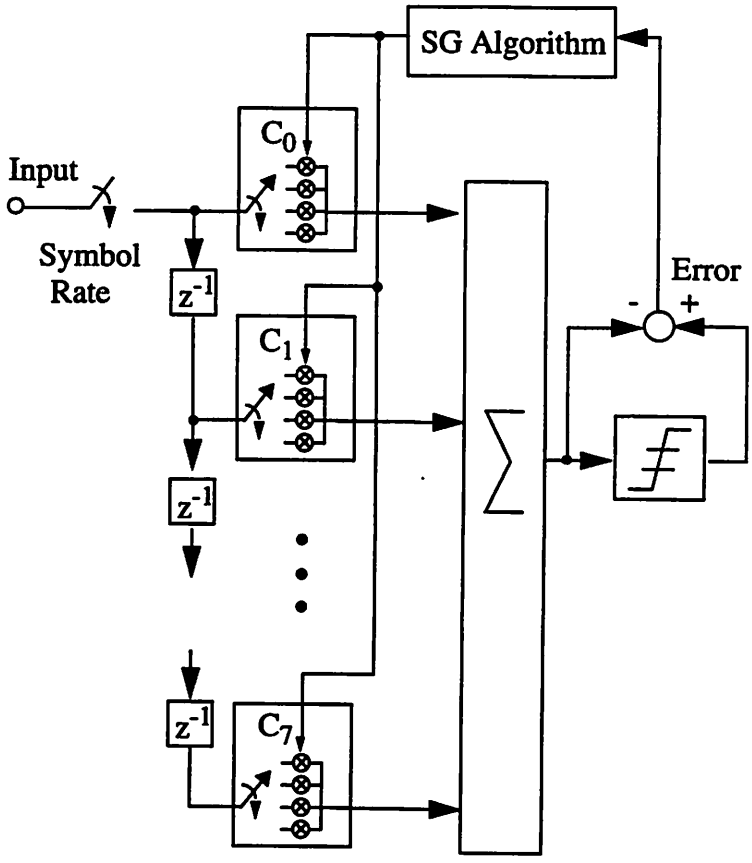


Figure 3.4 Adaptive Equalizer with Parallel Multipliers

Figure 3.5. Since there are four 25MHz multipliers in each tap, it is very straight forward to construct four identical parallel FIR filters by connecting the first 25MHz multiplier in each tap to the first accumulator, the second multiplier to the second accumulator, and so on and so forth. Thus four parallel time-interleaved channels have been created for the signals to flow through. The idea is demonstrated in Figure 3.6. At a certain point in time, say t_0, the eight samples stored in the 100 MHz delay line are transferred to the first filter. After a delay of one T (10ns in this case), the eight samples in the delay line are shifted down by one delay register, and these eight repositioned samples are shifted into the second filter. The same procedure is applied to the third and the forth filter. In this way, from the perspective of the delay line, the four 25MHz FIR filters perform the same function as

a 100MHz filter. Another important feature of this adaptive equalizer is that the coefficients are being updated at 25MHz only. This can be achieved by using the signal coming out of filter 1 only to drive the stochastic gradient algorithm (see Figure 3.5). Unlike some communication channels which have rapid varying characteristics such as a mobile system, the magnetic disk drive read channel is relatively stable. Therefore, the channel characteristics are changing at a much rate than the target throughput of the proposed DSP. By preforming the coefficient update at 25MHz, a significant saving in area and power consumption can be achieved. However, this saving does not come for free. According to system-level simulations, updating the coefficients at one-fourth of the symbol rate causes undesirable effects on the robustness of the algorithm. For example, its convergency

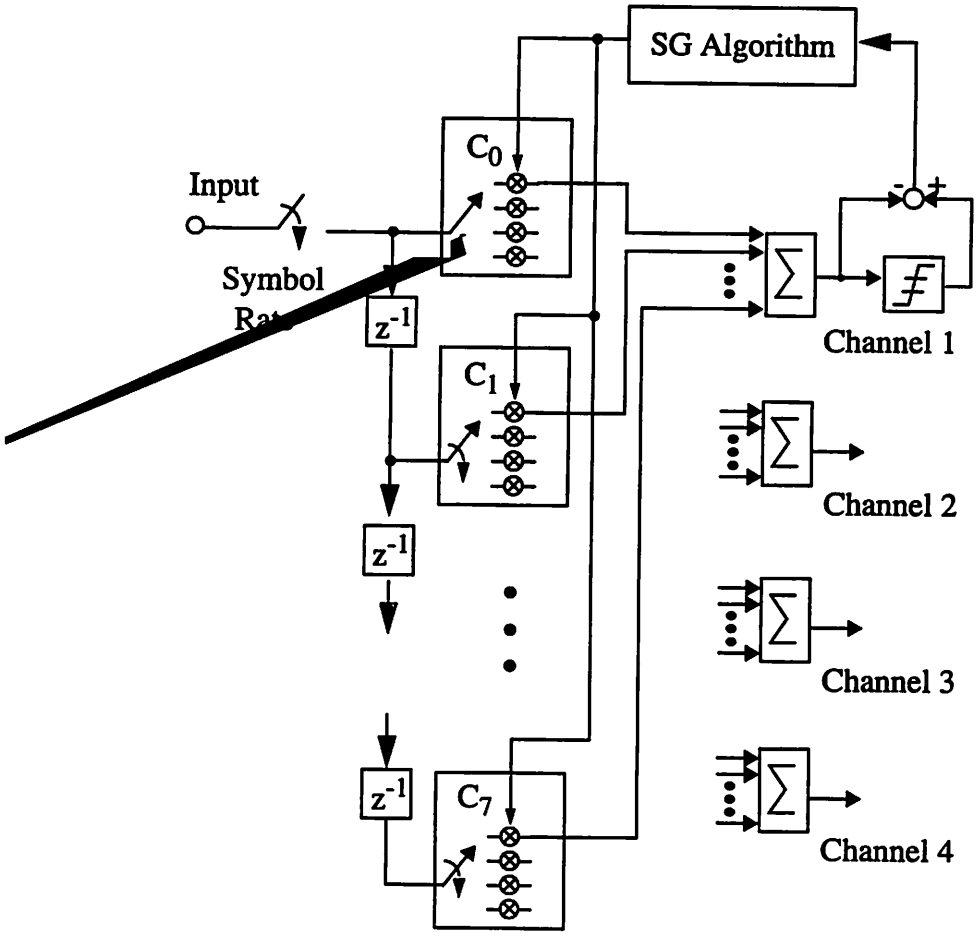


Figure 3.5 Adaptive Equalizer with Four Parallel Channels

becomes slower and more sensitive to the noise level. In addition, the minimum achievable mean square error becomes larger. This subject is described in details in Chris Rudell's master report [18].

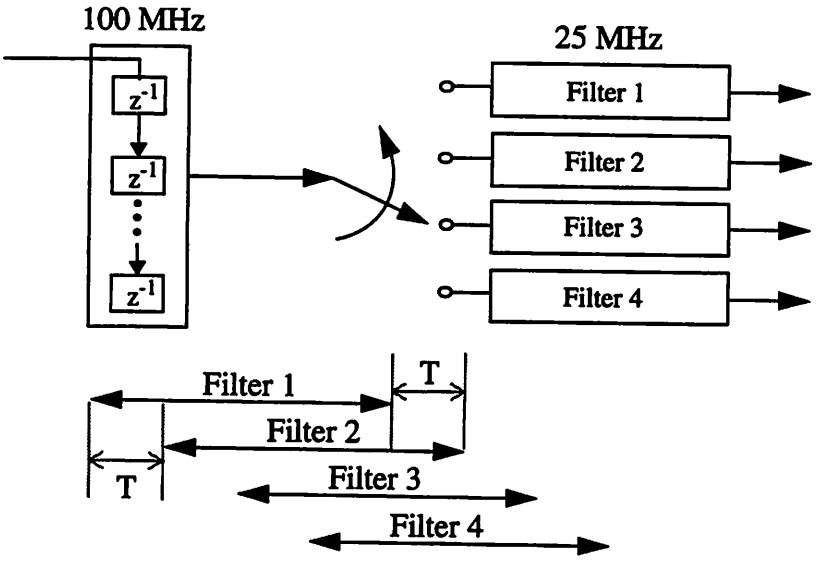


Figure 3.6 Four Time-Interleaved Channels

As described in section 3.1, the output of the adaptive equalizer serves as the input to the Viterbi decoder. As explained in Chapter 2, the $1 - D^2$ operation is achieved by subtracting the current sample by the one two sampling periods ago. Therefore, the odd and even channel output symbols are independent, and can be interpreted as two interleaved $1 - D$ channels at a symbol rate of 50MHz. Thus two parallel 50MHz Viterbi decoders performing the $1/(1 - D)$ decoding operation can be used instead. This is shown in Figure 3.7. The first Viterbi decoder takes its input from the first and the third channel, while the second Viterbi decoder takes its input from the second and the fourth channel. Finally the outputs of these two Viterbi decoders merge, and the chip output is a 100MHz bit stream.

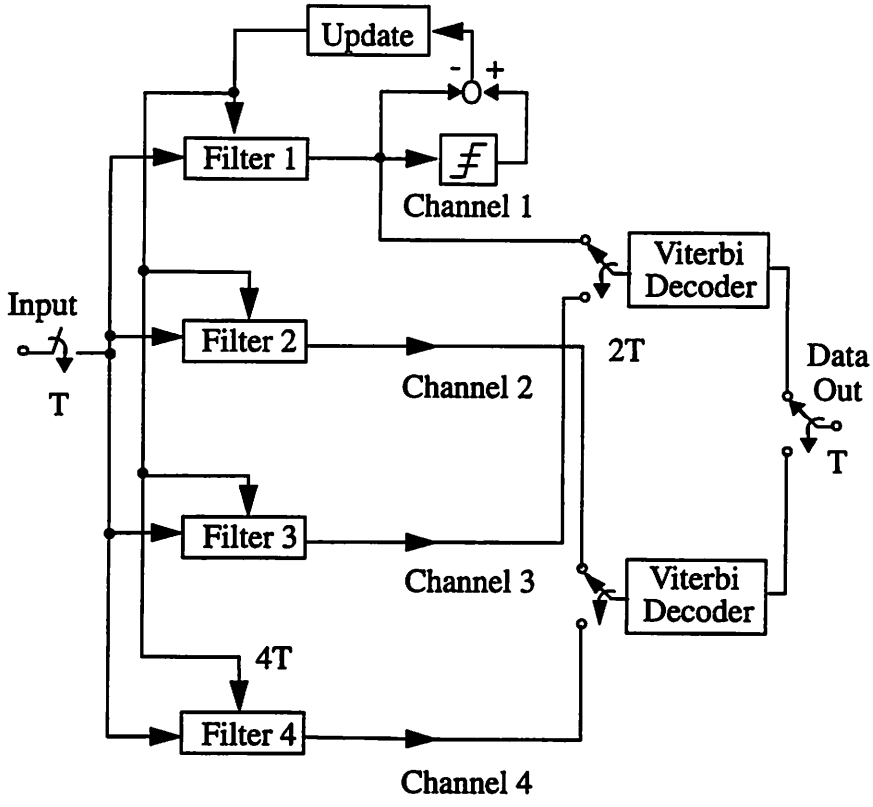


Figure 3.7 DSP Architecture with Four Parallel Channels and Two Parallel Viterbi Decoders

Chapter 4

Physical Design of DSP

4.1 Layout of Key Functional Blocks and Data Flow

4.1.1 Chip Plan

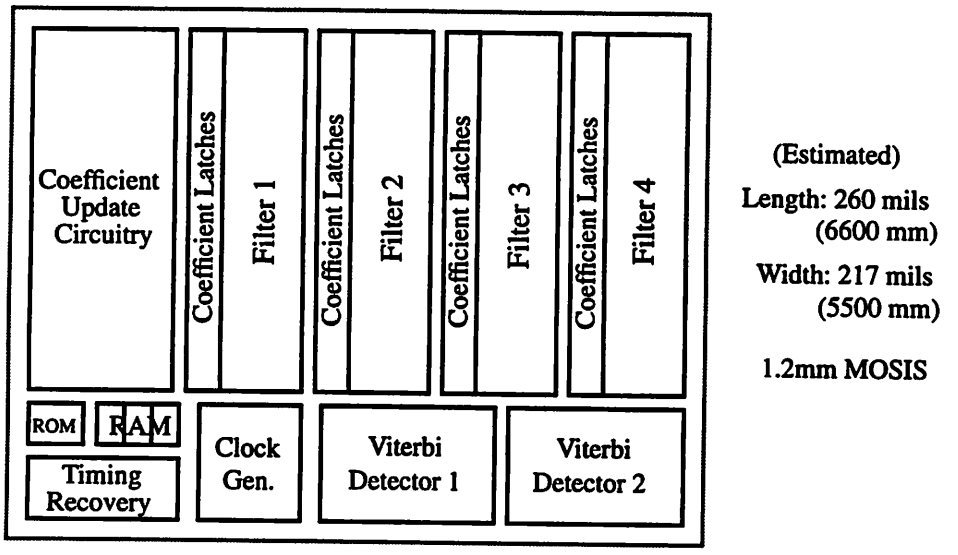


Figure 4.1 Overall Chip Organization

In the last chapter, the design of the proposed DSP on an architectural level is covered. In this chapter, issues more closely related to the physical design of the chip, such layout and transistor level circuit design, will be discussed. First of all, it is a helpful introduction to look at the overall chip plan for the proposed DSP (see Figure 4.1). The top part of the chip accommodates the four parallel FIR filters described before. The coefficient update circuitry is located on the left of these four filters and the two interleaved $1 - D$ Viterbi decoders are found below the filters. In addition, circuitry is required to

generate the four 25MHz clocks which serve to synchronize the data in the four interleaved channels. In a practical commercial chip, ROM and RAM are needed to store initial coefficients for training the adaptive filter. They provide good starting points from which the filter can converge faster and better. Interested readers should refer to Chris Rudell's master report for more details [18]. For the prototype, however, these on-chip memories are not included because it is more convenient to down load the coefficients to the chip with external switches.

4.1.2 Bussing of Coefficients Through the Channels

As mentioned in section 3.3, the coefficients of the filter are being updated at only one-fourth of the symbol rate because of the relatively slow-varying characteristics of the disk drive read channel. Therefore, the length of the updating period is 4T instead, where T is one sampling period. As a result, the same set of coefficients is shared by the four parallel filters within one updating period. As shown in Figure 4.1, the filters with the update circuitry span the whole length of the chip; therefore, it is important to bus the coefficients through the filters in an area and power efficient way.

The most straightforward approach is to drive all four filters directly from the update circuitry. Since there are eight coefficients and each of them is represented as a 6-bit word, this means 48 metal lines need to be routed around these filters until they reach the coefficient latches of the fourth filter. This is obviously a waste of chip area. In the proposed DSP, the outputs of the update circuitry are connected to the inputs of the coefficient latches of the first filter only. The outputs of the coefficient latches of the first filter are, in turn, connected to the inputs of the coefficient latches of the second filter. As described in section 3.3, the four parallel filters can be regarded as four time-interleaved channels. To synchronize the signal flow, four 25MHz (one-fourth of the symbol rate) clocks with a phase offset of 10ns (=1T) from one of them to the next one are required. These clocks

are shown in Figure 4.2. Note that all the latches on chip are positive edge-triggered. Therefore, the coefficients become available at the output of coefficient latches of filter one right after clock one goes high. So as long as the time needed to transfer the data through the interconnects between filter one and two is less then one T (10ns), then filter two will be able to sample the right coefficients. This is certainly not a problem since the interconnects are just metal lines. The coefficients are bussed to the third and fourth filter in the same way.

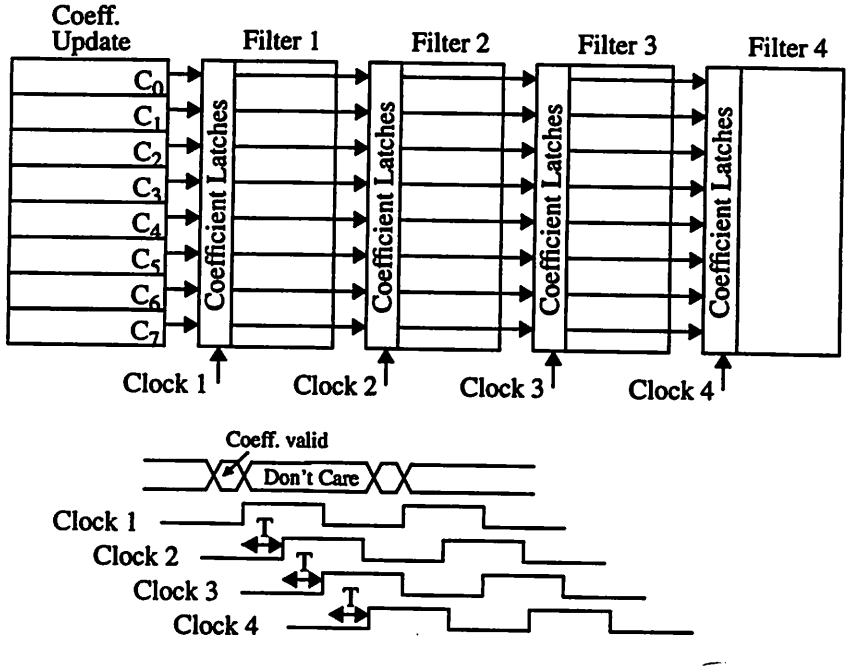


Figure 4.2 Bussing of Coefficients Through the Channels

The advantages of this implementation are as follows. Compared to the straightforward approach, this allows a significant saving in area since it is now not necessary to route those 48 metal lines around the filters. This also represents a corresponding saving in power consumption since it is now not necessary to charge and discharge those 48 metal lines which run across almost the whole length of the chip. For the same reason, powerful drivers at the outputs of the update circuitry are not necessary, since the metal lines con-

necting the first and second filters are too short to cause any significant capacitive loading. Finally, as shown in Figure 4.2, the coefficients at the outputs of the update circuitry only need to be valid until they are sampled by filter one. This makes the design of the update circuitry easier.

4.1.3 Input Signal Flow

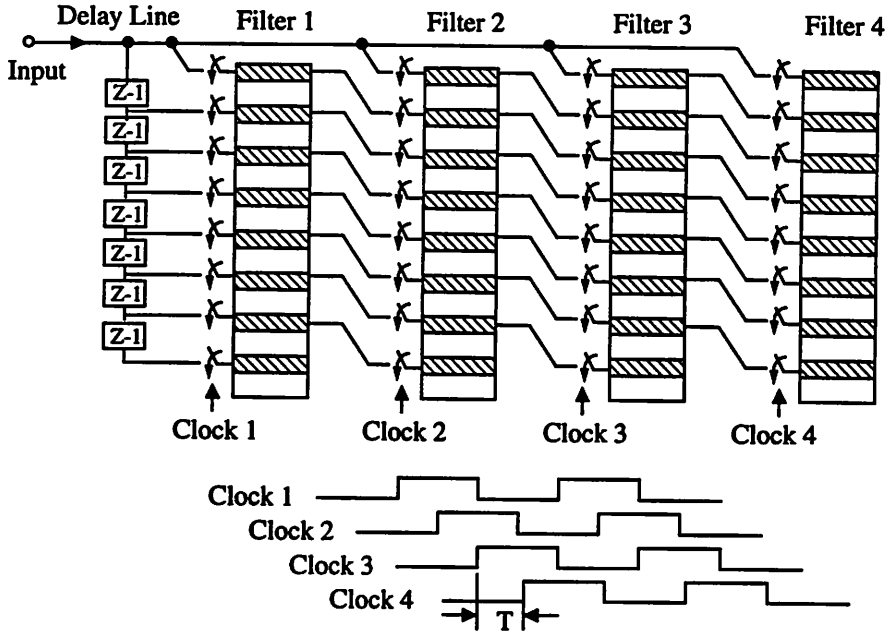


Figure 4.3 Signal Flow Diagram I

As in the case of coefficient bussing, the input samples, including the old ones stored in the delay line and new ones coming in from the front end, need to be distributed to the four parallel filters where they are being equalized. Basically the same approach used in coefficient bussing is adopted here again (see Figure 4.3). In this case, however, the update circuitry is replaced by the delay line. Filter one samples its input from the delay line when clock 1 goes high. Note that the first (or the latest) sample in the delay line is the current input sample. As in the case of coefficient bussing, the input latch in each tap

of filter two samples its input from the output of the corresponding input latch in filter one as clock 2 goes high. However, for correct timing operation, the first tap of filter two is directly connected to the chip input, while the second tap is connected to the output of the first tap of filter one, the third tap to the second tap of filter two, and so on and so forth. This is demonstrated in Figure 4.4. At a certain point in time, input samples x_k through x_{k-7} are sampled by the first filter as clock 1 turns on. After a delay of T, x_k through x_{k-6} are passed to filter 2 and a new input, x_{k+1}, is sampled by the input latch of the first tap of filter two. Thus this is no loss of information even though each filter is running at only one-fourth of the symbol rate.

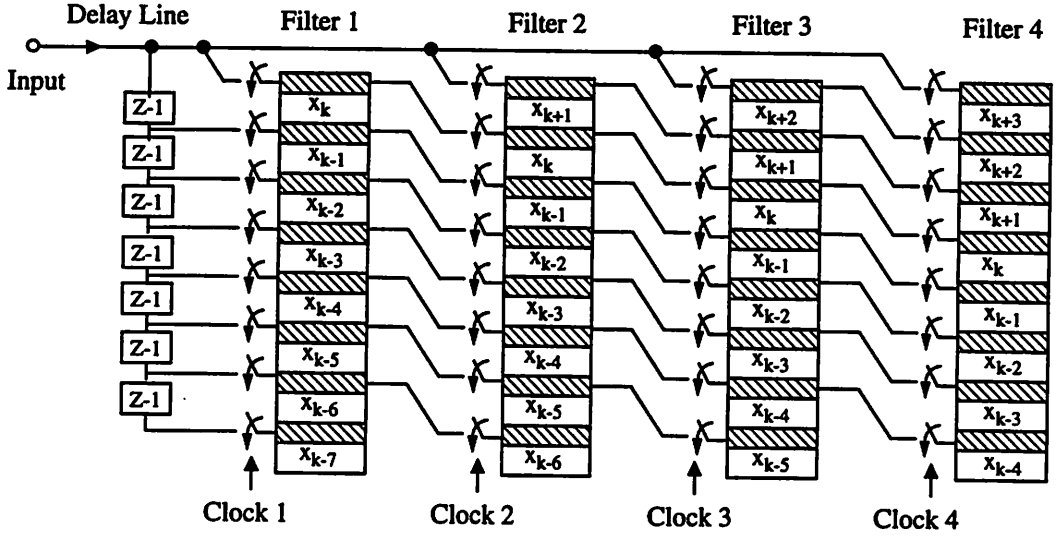


Figure 4.4 Signal Flow Diagram II

4.2 Circuit Implementation of Key Functional Blocks

4.2.1 Carry Save Multiplier

As described in Chapter 3, four parallel filters will be used to achieve the performance of a 100MHz 8-tap FIR filter. Also, the proposed DSP is a 6-bit system. Consequently, $4 \times 8 = 32$ 6-bit by 6-bit multipliers are required. The design of multiplier has a

significant impact on the performance of the entire chip. Recall that the goal is to achieve a throughput of 100Mbits/sec with a power consumption of less than 250mW. These high-speed and low-power specifications make the design of the multiplier especially difficult. At a very early stage of this research project, an effort was made to do a survey of different multiplier architectures.

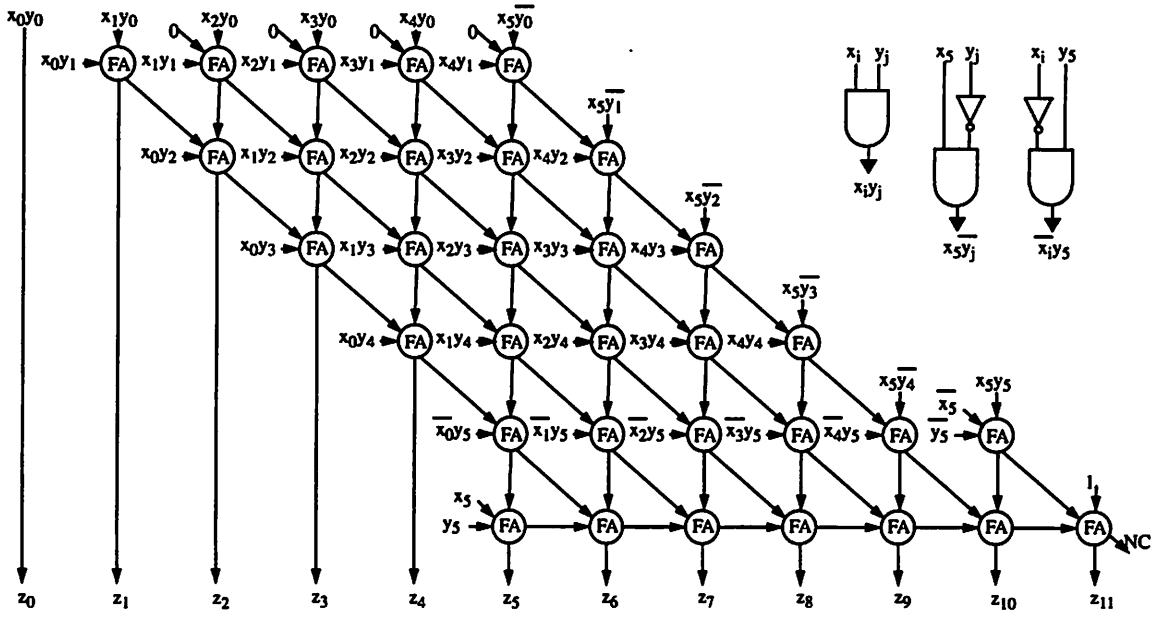
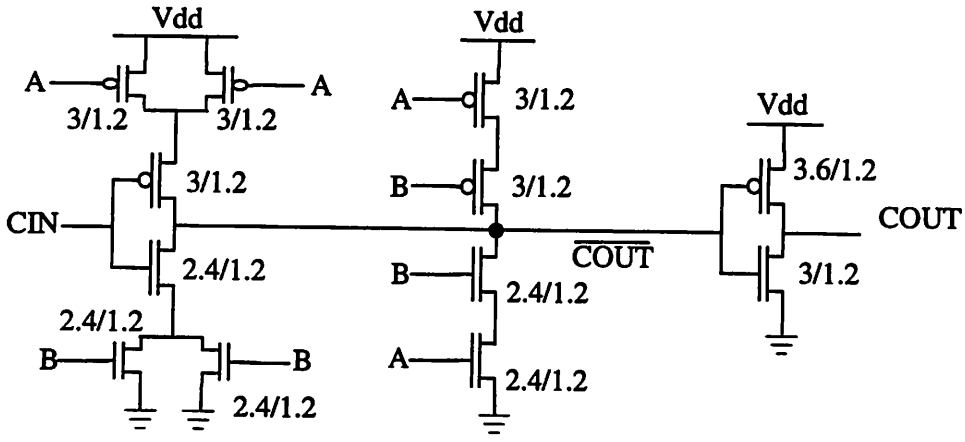


Figure 4.5 Modified Carry Save Multiplier

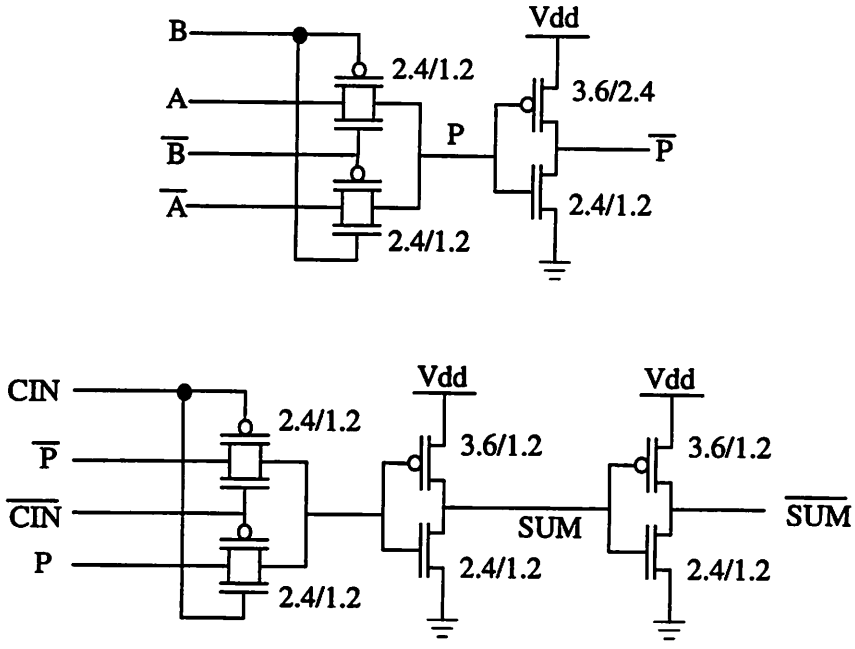
The high-speed requirement of the chip simply rules out the possibility of using serial multiplication methods. In fact, only parallel multiplier structures are studied. It was found that most of the fastest multipliers, which are mainly used the data path of high speed microprocessors, adopt the Wallace tree structure with booth encoding [10], [11]. However, these implementations involve complicated overhead circuitry, which make them less attractive candidates for use in small multipliers. Actually the final choice is the

carry save multiplier, which is shown in Figure 4.5 [12]. In addition, the Baugh-Wooley two's complement parallel array multiplication algorithm is used to lower the number of required full adders by about 20%. Interested readers should refer to [13] for more details. There are several reasons for choosing the carry save structure. First of all, it is a simple linear multiplier structure, and therefore it is easy to implement. Also, it is fast, because it uses a fully parallel multiplication scheme. Since its structure is quite regular, it can be readily re-designed for different precisions such as 8 bits by 8 bits or 10 bits by 10 bits. As shown in Figure 4.5, the critical path mainly consists of 12 adders. So it is important to optimize the delay through each adder. In fact, there are a large number of ways to design the full adder, and the one shown in Figure 4.6 [14] is just one option. Note that the dimensions of the transistors are in micrometer. This structure is chosen by comparing it to other implementations by using SPICE. The entire multiplier was simulated with SPICE, and the results show that it can achieve a maximum speed of 30MHz with a power consumption of 2.5mW.

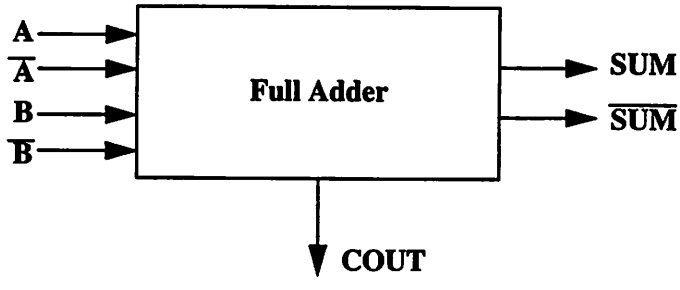


(a) Carry Generation

Figure 4.6 Full Adder Design (to be continued)



(b) Sum Generation



(c) Block Diagram

Figure 4.6 Full Adder Design

4.2.2 Carry Select Accumulator

As described in section 3.3, there are actually four parallel 8-tap FIR filters in the four time-interleaved channels. Therefore, four accumulators are needed. Each of these accumulator should be able to add up the 8 products coming out from the 8 taps of each FIR filter at 25MHz. Figure 4.7 shows the carry select accumulator that adopted in the proposed DSP. The carry select accumulators are actually composed of 7 carry select adders in a binary tree arrangement. Note that each product coming out from each tap is represented as a 10-bit two's complement number. By allowing the width of the datapath to increase from 10 bits to 13 bits after going through three layers of adders, the underflow and overflow problems are effectively avoided.

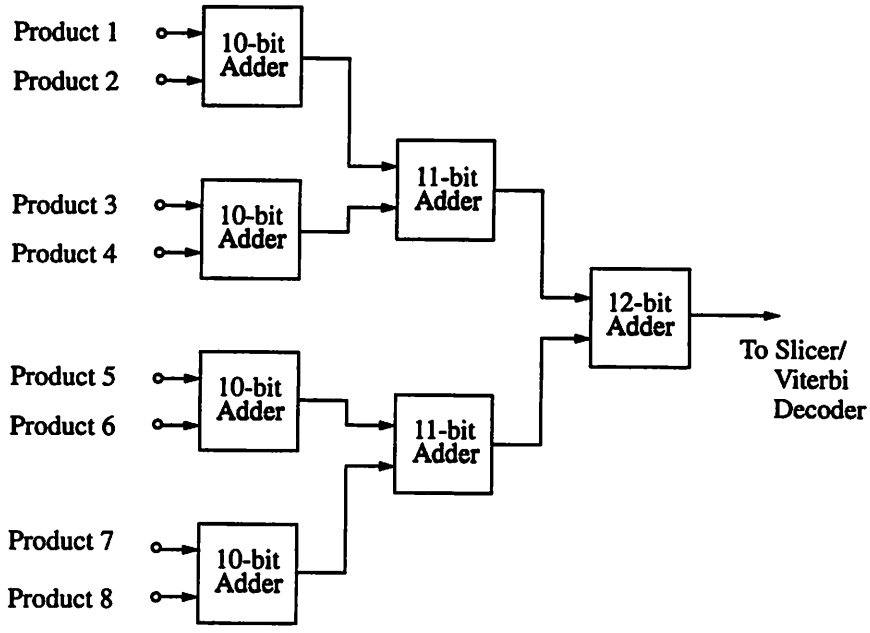
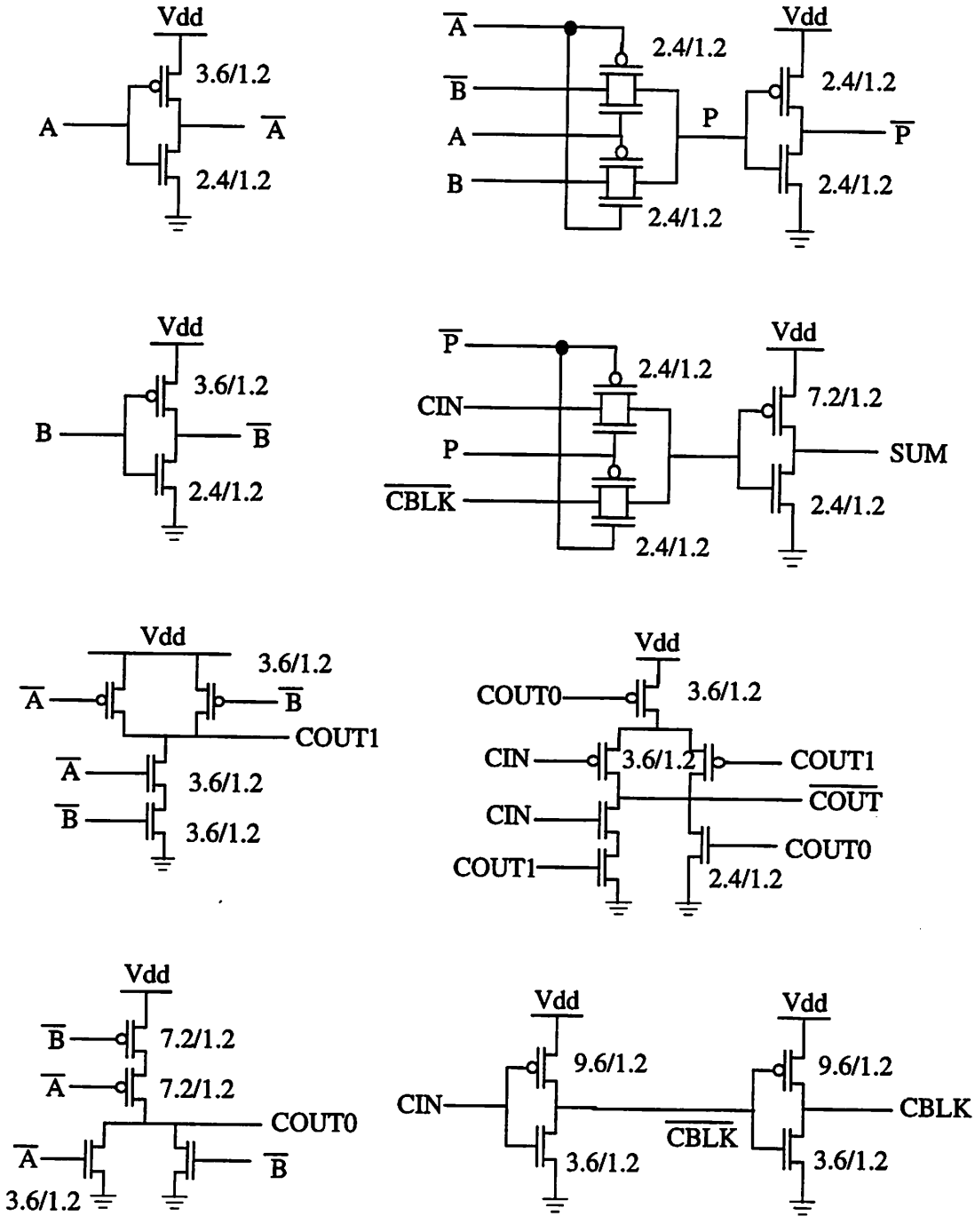


Figure 4.7 Carry Select Accumulator

System-level simulations using Ptolemy show that the critical path delay in the coefficient update loop needs to be no longer than 8T (T = 10ns) for the LMS algorithm to converge properly. Since 4T is needed for the tap multiplication, the accumulator and coefficient circuitry have only 4T to finish all the required computations. In fact, it ends up that the accumulator has only 25ns to add up 8 10-bit numbers. SPICE simulations show that the carry select adder found in the Ultra Low Power Cell Library [14] developed in Professor Brodersen's research group at UC Berkeley can be satisfied the requirement mentioned above. Figure 4.8 shows the designs of all the required leafcells. These designs are taken directly from the library.

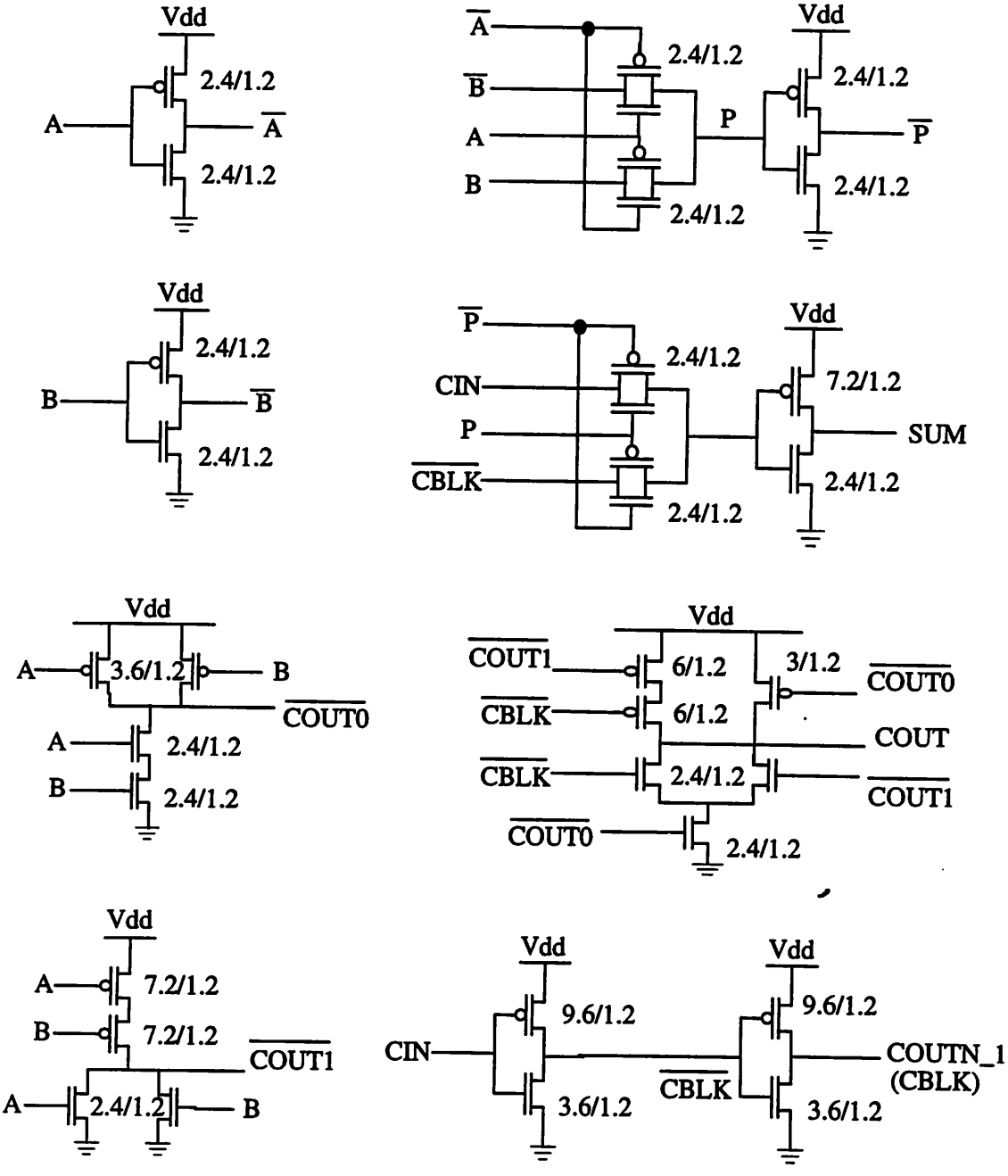
Each carry select adder consists of a chain of leafcells (see Figure 4.8), which are designed in such a way that the critical path delay is optimized by eliminating redundant inverter gate delays. The number of leafcells corresponds to the number of bits in a particular adder. The leafcells are tiled as follows. The leafcells are grouped in several stages. The LSB of each stage utilizes either the Lsb_Even or the Lsb_Odd cell, depending on whether the stage has an even or odd number of bits. The MSB of each stage is always the Msb_Even cell except for the last stage of the adder, which uses the cell CSA_Msb. However, if the last stage has one bit only, then the cell Lsb_Odd will be used instead. Finally, the remaining bits of each stage alternate between the Msb_Even and Msb_Odd cell. To achieve higher operating speed, the number of bits (leafcells) in each stage should be obtained in a more intelligent way. To explain this point, it is helpful to examine the critical path of the adder. In each stage of the adder, two results are produced. The first one is produced assuming a zero carry input, while the other one assumes a one carry input. The goal is to have both of these results ready by the time the previous stage comes up with its final carry output. Then this carry input from the previous stage will be used to select between one of these results. Therefore, the critical path mainly consists of the logic blocks (mainly adders) used to generate the final carry output of the first stage and the multiplexing circuitry in the following stages. In the proposed DSP, as mentioned, the

output of each adder is one bit wider than the input. Also, the two's complement numbering system is adopted. For these reasons, the adder will make mistake at the MSB if the MSB's of the two inputs are different. Because of this, error correction needs to be applied to the adder's MSB. First of all, an exclusive OR gate is used to determine if the MSB's of the two inputs are different. If the XOR gate is set to high, which indicates a mistake has occurred, then the error correction block will tell the adder to copy its second output MSB to the MSB; otherwise no error correction will be performed. This is actually a simple sign extension, and the readers should be able to verify it readily. Although this is rather simple, the delays involved in these operations have to be added to the total critical path delay. In the carry select accumulator, 10-bit, 11-bit, and 12-bit adders are required. Simulations show that the optimal stagings are 2-3-5, 2-3-6, and 2-4-6 respectively. For a more detailed analysis of this carry select adder, interested readers can refer to the section "Carry Select Adder" prepared by Kenway Tam for the Ultra Low Power Cell Library.



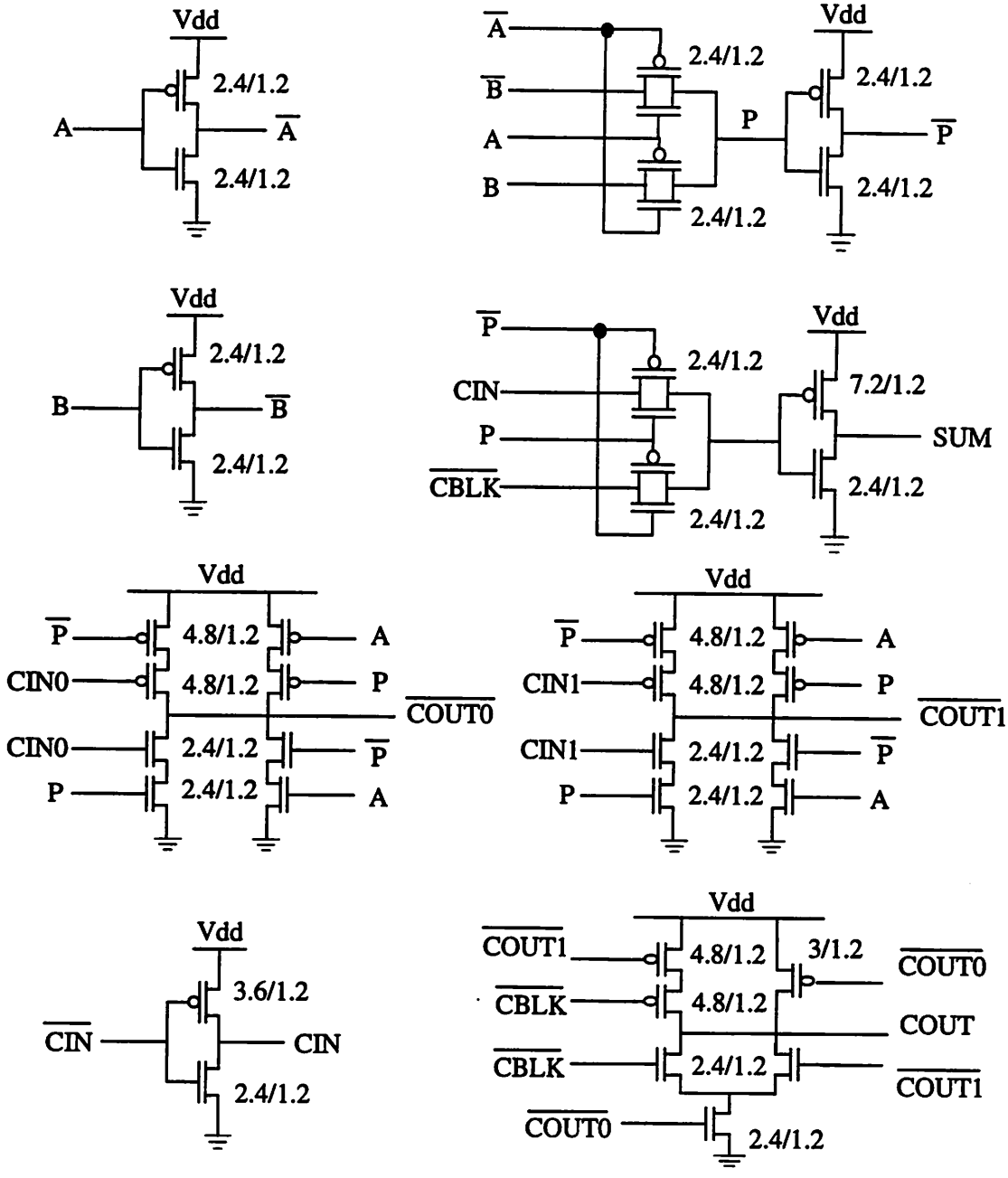
(a) Lsb_Even Subcell

Figure 4.8 Leafcells for Carry Select Adder (to be continued)



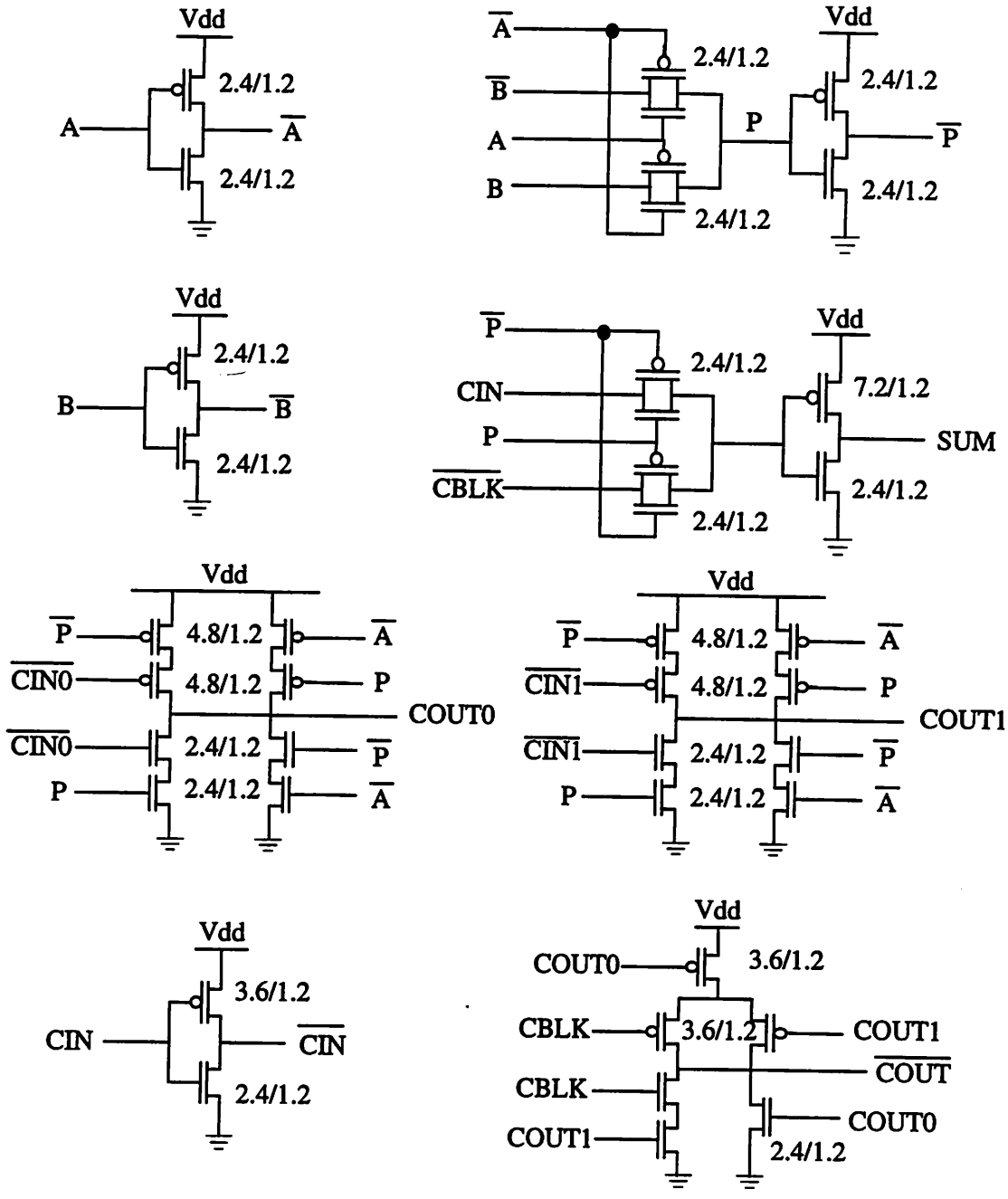
(b) Lsb_Odd Subcell

Figure 4.8 Leafcells for Carry Select Adder (to be continued)



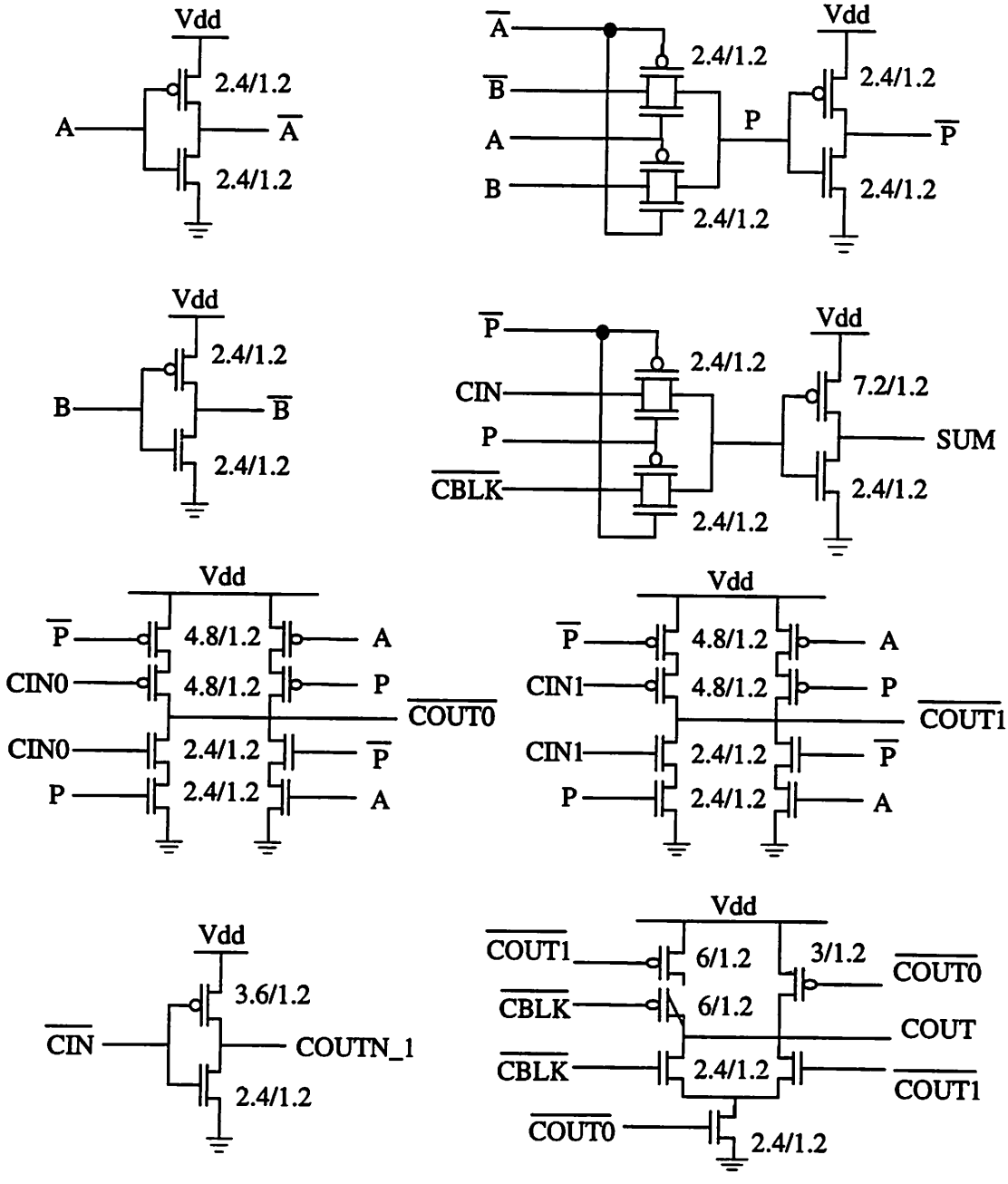
(c) Msb_Even Subcell

Figure 4.8 Leafcells for Carry Select Adder (to be continued)



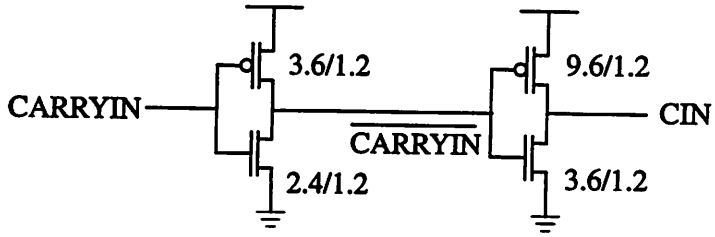
(d) Msb_Odd Subcell

Figure 4.8 Leafcells for Carry Select Adder (to be continued)

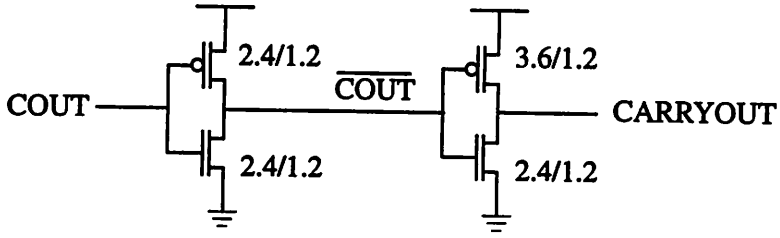


(e) CSA_Msb Subcell

Figure 4.8 Leafcells for Carry Select Adder (to be continued)



(f) Carry Input Buffer



(g) Carry Output Buffer

Figure 4.8 Leafcells for Carry Select Adder

Chapter 5

Viterbi Detection

5.1 Introduction

As explained in Chapter 2, the Partial Response (PR) Signaling scheme encodes the two-level {0, 1} input bits into a set of three-level {-1, 0, 1} channel input symbols. Therefore, redundancy is introduced into the signal. It was proven that the PR signaling scheme, together with Maximum Likelihood (ML) detection, can approximate the match filter bound closely, since the ML detector can take advantage of the correlation in the PR signal to provide a coding gain of 3dB. In addition, it was shown that in section 3.3, the Class IV PR read channel output can be interpreted as two interleaved $1-D$ channels. Consequently, two identical Viterbi decoders performing the $1/(1-D)$ operation at one-half the symbol rate (ie. 50MHz) are required. In the last three decades, a number of variations from the original Viterbi algorithm were published. For the proposed DSP, however, the Different Metric (DM) algorithm proposed by M. J. Ferguson in 1972 was chosen because of its relatively short critical path and simplicity for actual implementation. In section 5.2, the DM algorithm will be derived, while its actual implementation will be described in section 5.3.

5.2 The Different Metric Algorithm

In this section, the DM algorithm will be derived using the approach found in [15]. Note that the equations in this section are derived for the $1-D$ signaling system, which corresponds to the NRZ coding scheme. As a first step, define the input bits, which corre-

spond to the magnetization patterns stored on the disk, as $x_k \in \{0, 1\}$. Therefore the symbols input to the channel can be written as

$$y_k = x_k - x_{k-1} \tag{5.1}$$

Clearly, y_k is a three-level signal where $y_k \in \{-(1, 0, 1)\}$. Since the ML detector is optimal in a channel with Guassian white noise, the channel output is assumed to be

$$z_k = y_k + n_k \tag{5.2}$$

where n_k is additive white Guassian noise. The two-state Trellis diagram shown in Figure 5.1 is an efficient way to represent a sequence of y_k resulting from a corresponding sequence of x_k. Note that the nodes represent the x_k values while the branches represent the y_k values.

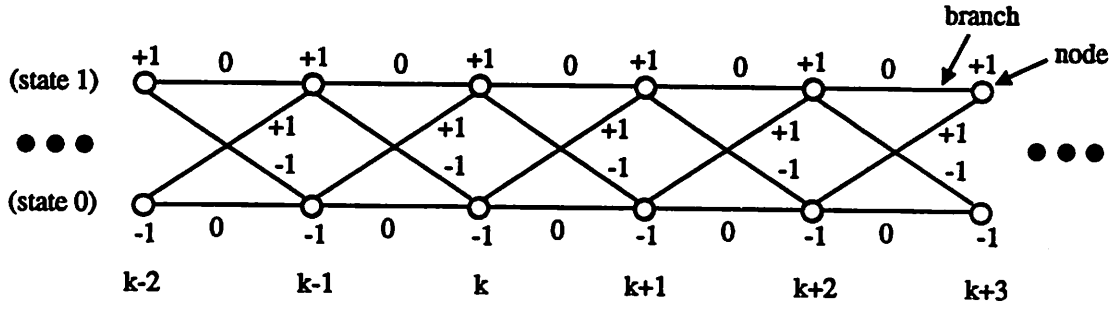


Figure 5.1 Two-State Trellis Diagram

To achieve optimal detection in the presence of Guassian white noise, the Minimum Distance Criterion states that a ML detector needs to minimize

$$\sum_{k=0}^{\infty} (z_k - y_k)^2 \tag{5.3}$$

for an infinite sequence of z_k's. By using (5.1), (5.3) can be written as

$$\sum_{k=0}^{\infty} [z_k^2 + (\hat{x}_k - \hat{x}_{k-1})^2 - 2(z_k)(\hat{x}_k - \hat{x}_{k-1})] \tag{5.4}$$

where $\hat{x}_k$'s are the estimates of the transmitted bits x_k's. Since $\sum_{k=0}^{\infty} z_k^2$ is a constant as long as the transmission power remains the same, minimizing (5.4) is equivalent to maximizing

$$\sum_{k=0}^{\infty} \left[z_k (\hat{x}_k - \hat{x}_{k-1}) - \frac{1}{2} (\hat{x}_k - \hat{x}_{k-1})^2 \right] \tag{5.5}$$

In fact, all the possible sequences of $\hat{x}_k$ can be represented by paths through the trellis. In a practical data transmission system, it is impossible to make detection decisions only after an infinite sequence of symbols has been received. Fortunately, since all the possible paths will pass through either $\hat{x}_k = 0$ or $\hat{x}_k = 1$ at the k^{th} node, the expression (5.5) can be minimized sequentially by keeping track of the accumulated branch metrics up to these two nodes. These two accumulated branch metrics are given by

$$L_k^+ \equiv \underset{\substack{\text{all paths} \\ \text{with } x_k=1}}{maximum} \{ \sum_{m=0}^{k} z_m (\hat{x}_m - \hat{x}_{m-1}) - \frac{1}{2} \sum_{m=0}^{k} (\hat{x}_m - \hat{x}_{m-1})^2 \} \tag{5.6}$$

for all the paths leading to $\hat{x}_k = 1$, and

$$L_k^- \equiv \underset{\substack{\text{all paths} \\ \text{with } x_k=0}}{maximum} \{ \sum_{m=0}^{k} z_m (\hat{x}_m - \hat{x}_m) - \frac{1}{2} \sum_{m=0}^{k} (\hat{x}_m - \hat{x}_{m-1})^2 \} \tag{5.7}$$

for all the paths leading to $\hat{x}_k = 0$.

Note that all possible sums in (5.5) can be represented by paths in a trellis. This can be done as follows. When $\hat{x}_k = \hat{x}_{k+1}$ (ie. $y_k = 0$), the branch running from the k^{th} to the

$(k+1)^{th}$ node contributes nothing to the sum in (5.5). When $\hat{x}_{k+1} = 1, \hat{x}_k = 0$, then the branch contributes $z_{k+1} - 1/2$, while when $\hat{x}_{k+1} = 0, \hat{x}_k = 1$, then the contribution will be $-\hat{z}_{k+1} - 1/2$. Thus a new trellis which contains all the possible sums in (5.5) can be created (see Figure 5.2). In Figure 5.2, note that there are four branches running from the k^{th} to the $(k+1)^{th}$ node (see the boxed section). Two of these branches lead to the node $\hat{x}_k = 1$ and the other two lead to the node $\hat{x}_k = 0$. For the two leading to $\hat{x}_k = 1$, the one with a larger accumulated branch metric will be selected, therefore

$$L_{k+1}^{+} = max\begin{pmatrix} L_k^{+} \\ L_k^{-} + z_{k+1} - 1/2 \end{pmatrix} \begin{matrix} (\text{from } x_k = 1) \\ (\text{form } x_k = 0) \end{matrix} \quad \begin{matrix} (5.8a) \\ (5.8b) \end{matrix}$$

Similarly, the one terminated at $\hat{x}_k = 0$ will be chosen based on

$$L_{k+1}^{-} = max\begin{pmatrix} L_k^{+} - z_{k+1} - 1/2 \\ L_k^{-} \end{pmatrix} \begin{matrix} (\text{from } x_k = 1) \\ (\text{form } x_k = 0) \end{matrix} \quad \begin{matrix} (5.9a) \\ (5.9b) \end{matrix}$$

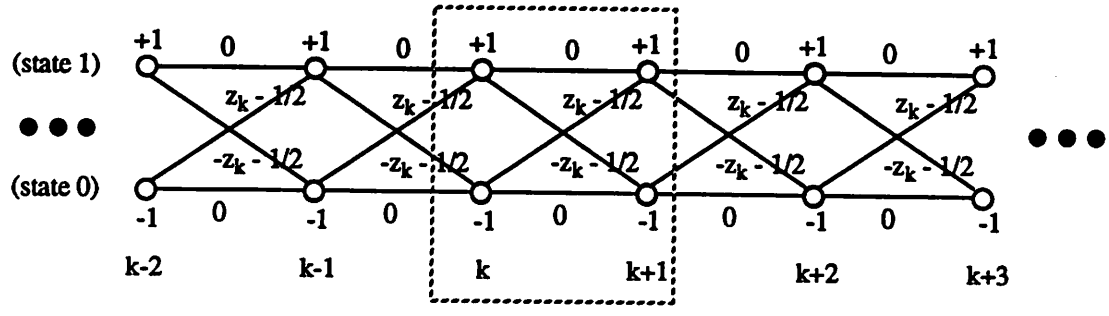


Figure 5.2 Trellis Diagram showing Branch Contributions to Sum in (5.5)

If the paths leading to nodes $\hat{x}_{k+1} = 1$ and $\hat{x}_{k+1} = 0$ end up coming from the same state at the k^{th} node, then there is a merge at that node. On the other hand, if there are two paths left at the k^{th} node, then it is said that there is no merge at that node. The two possi-

ble cases for "no merge" are shown in Figure 5.3. For the case shown in Figure 5.3a to occur, (5.8a) and (5.9b) need to be larger than (5.8b) and (5.9a) respectively. Subtracting (5.8b) from (5.8a) and (5.9a) from (5.9b), these requirements are simplified to

$$-\frac{1}{2} \le L_k^+ - L_k^- - z_{k+1} \le \frac{1}{2} \tag{5.10}$$

On the other hand, for the case shown in Figure 5.3b to occur, the conditions required are

$$and \begin{cases} L_k^+ - L_k^- - z_{k+1} \le -\frac{1}{2} & (5.11a) \\ L_k^+ - L_k^- - z_{k+1} \ge \frac{1}{2} & (5.11b) \end{cases}$$

which is certainly impossible. Now it is clear that the requirement for the case of a merge at state 1 at the k^{th} node can be obtained by subtracting (5.8a) by (5.8b) and (5.9a) by (5.9b), which end up with

$$L_k^+ - L_k^- - z_{k+1} \ge \frac{1}{2} \tag{5.12}$$

Similarly, the condition for a merge at state 0 at the k^{th} node is given by

$$L_k^+ - L_k^- - z_{k+1} \le -\frac{1}{2} \tag{5.13}$$

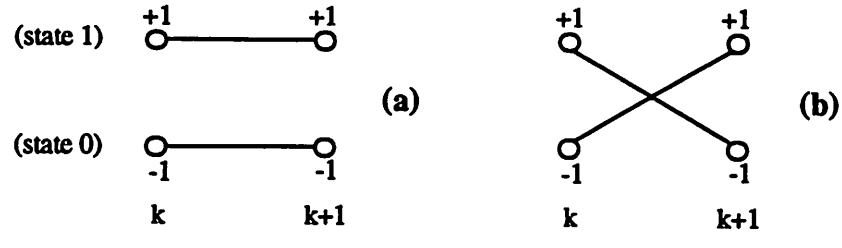


Figure 5.3 Two Possible Unmerged Cases

Define the "Different Metric" (DM) as

$$\zeta_k = L_k^+ - L_k^- \tag{5.14}$$

Note that after a merge at state 1 at the k^{th} node, the DM at the $(k+1)^{th}$ node is simply given by subtracting (5.9a) from (5.8a):

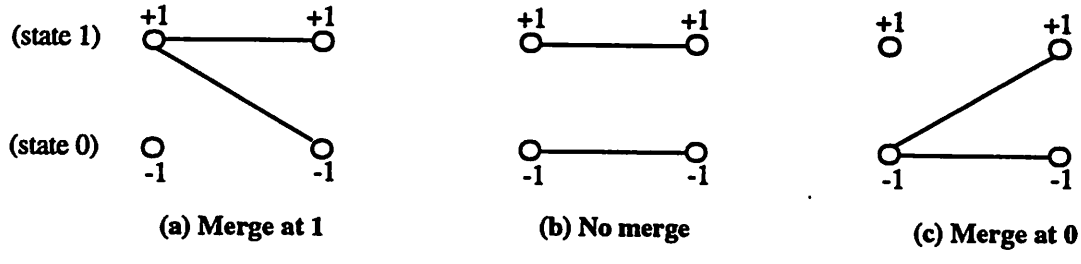
$$\zeta_{k+1} = z_{k+1} + \frac{1}{2} \tag{5.15}$$

Similarly, after a merge at state 0 at the k^{th} node, the DM at the $(k+1)^{th}$ node is obtained by subtracting (5.8b) from (5.9b):

$$\zeta_{k+1} = z_{k+1} - \frac{1}{2} \tag{5.16}$$

Using (5.14), the results derived in (5.10), (5.12), (5.13), (5.15), and (5.16) can be summarized as

$$\zeta_{k+1} = \begin{cases} z_{k+1} + \frac{1}{2}, & \zeta_k - z_{k+1} \geq \frac{1}{2} \quad \text{for a merge at 1} & (5.17a) \\ \zeta_k, & -\frac{1}{2} \leq \zeta_k - z_{k+1} \leq \frac{1}{2} \quad \text{for no merge} & (5.17b) \\ z_{k+1} - \frac{1}{2}, & \zeta_k - z_{k+1} \leq -\frac{1}{2} \quad \text{for a merge at 0} & (5.17c) \end{cases}$$



(a) Merge at 1 (b) No merge (c) Merge at 0

Figure 5.4 Three Possible Branch Progression Modes

If the most recent merge occurs at state 1 at the p^{th} node (that means there is no merge afterwards), then, by (5.15), $\zeta_k = \zeta_p = z_p + 1/2$. In this case, (5.17) can be modified to

$$\zeta_{k+1} = \begin{cases} z_{k+1} + \frac{1}{2}, & z_p - z_{k+1} \geq 0 \quad \text{for a merge at 1} \quad (5.18a) \\ z_p + \frac{1}{2}, & -1 \leq z_p - z_{k+1} \leq 0 \quad \text{for no merge} \quad (5.18b) \\ z_{k+1} - \frac{1}{2}, & z_p - z_{k+1} \leq -1 \quad \text{for a merge at 0} \quad (5.18c) \end{cases}$$

On the other hand, if the most recent merge occurs at state 0 at the p^{th} node, then, by (5.16), $\zeta_k = \zeta_p = z_p - 1/2$. In this case, (5.18) can be modified to

$$\zeta_{k+1} = \begin{cases} z_{k+1} + \frac{1}{2}, & z_p - z_{k+1} \geq 1 \quad \text{for a merge at 1} \quad (5.19a) \\ z_p - \frac{1}{2}, & 0 \leq z_p - z_{k+1} \leq 1 \quad \text{for no merge} \quad (5.19b) \\ z_{k+1} - \frac{1}{2}, & z_p - z_{k+1} \leq 0 \quad \text{for a merge at 0} \quad (5.19c) \end{cases}$$

At this point, the DM algorithm has been derived, and (5.18) and (5.19) are the two key equations that need to be implemented.

5.3 Circuit Implementation

In this section, the circuit implementation of the DM algorithm derived will be discussed. First of all, it is helpful to look the algorithm at a different but more intuitive way. As mentioned, the Viterbi decoder is designed for a $1 - D$ channel. This corresponds to the well-known NRZ code. A convenient way to look at the encoding process is to refer to the state transition shown in Figure 5.5. The first parameter in the parentheses corre-

sponds to the input bit and the second parameter corresponds to the encoded data symbol. Now (5.18) and (5.19) can be interpreted by using Figure 5.5. A "most recent merge at state 1" means that the decoder is currently "waiting" at the branch running from state 1 to state 0 for a new input date symbol for making any further decision. Referring to (5.18), this means that z_p, the "old" input causing the most recent merge, is stored as -1 (plus noise). By mapping (5.18) to the state transition diagram, it should be reasonably easy to justify the following observations. Since the decoder is already sitting at the branch coming out from state 1, it is most unlikely to see a merge at state 1 again. The case "no merge" corresponds to the case in which there is no change of state, which means returning to state 1 without going through state 0. With z_p equals to -1 and the range shown in (5.18c), the state 0 is clearly the most probable next state. The "most recent merge at state 0" can be interpreted in a similar way.

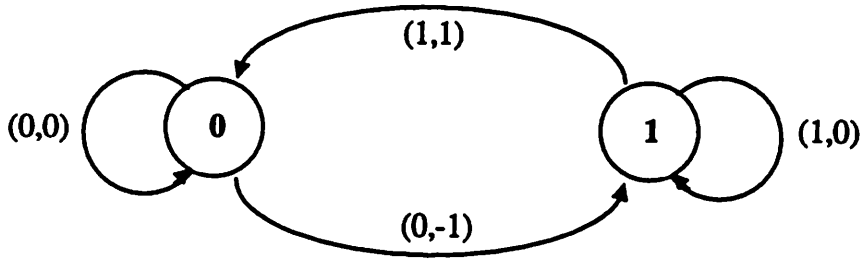


Figure 5.5 State Transition for NRZ Read Channel

From the description above, it is clear that two parameters absolutely need to be stored and being updated regularly [16]. The first one is the current state, or the state at which the most recent merge occurs. From now on, it is labelled as β. β carries only 1 bit of information, with "1" represents state 1 and "0" represents state 0. It decides whether (5.18) or (5.19) should be used. The second parameter is z_p, the "old" input which causes the most recent merge to occur. To make decisions with (5.18) and (5.19), the following operations are required. Obviously, the result of $(z_p - z_{k+1})$ is needed for determining

whether the next step should be a merge at state 1, a merge at state 0, or no merge. Based on the values of $(z_p - z_{k+1})$ and β (the current state), a logic block is required to generate two internal signals. One of them, which is labelled as *change_input*, is used to indicate that whether z_p needs to be updated or not. *Change_input* will be set to high if whenever there is a merge at either state. Therefore, *change_input* will be set to high when either (5.18a) and (5.18c), or (5.19a) and (5.19c), depending on the state at which the most recent merge occurs, are satisfied. The required logic is derived by simplifying and optimizing the corresponding truth table (see Table 5.1) with Expresso. Since $(z_p - z_{k+1})$ is stored as a 6-bit word, the truth table has 2^7 (=128) entries. Since a 6-bit two's complement numbering system is adopted in the proposed DSP, $2^6 = 64$ quantization levels are available to represent numerical values in the system. Among these 64 levels, 2^5=32 of them are allocated to negative values, while the other 2^5-1 = 31 levels are reserved for positive values. To leave some headroom for noise and coefficient variations in an adaptive system, 24 levels are mapped to a magnitude of unity in the PR signaling system. Furthermore, since the output of the $(z_p - z_{k+1})$ subtractor is represented as 6 instead of 7 bits, its value is effectively half the actual value. Consequently, the output of the subtractor are compared to ± 12 levels to determine which range it actually falls into. This truth table is simplified by using Expresso, and the simplified expression is implemented simply with random logic cells. Since the gate level implementation is trivial and not unique, it will be shown here.

In additional to *change_input*, another internally generated signal is required to decide whether β needs to be updated or not. This signal is labelled as *change_beta*. *Change_beta* will be set to high only when there is a transition from state 0 to state 1, or vice versa. Therefore, it will be set to high only when (5.18c) or (5.19a) is satisfied. The gate level logic circuitry is derived by using a truth table similar to the one shown in Table 5.1, and so the design process will not be repeated here.

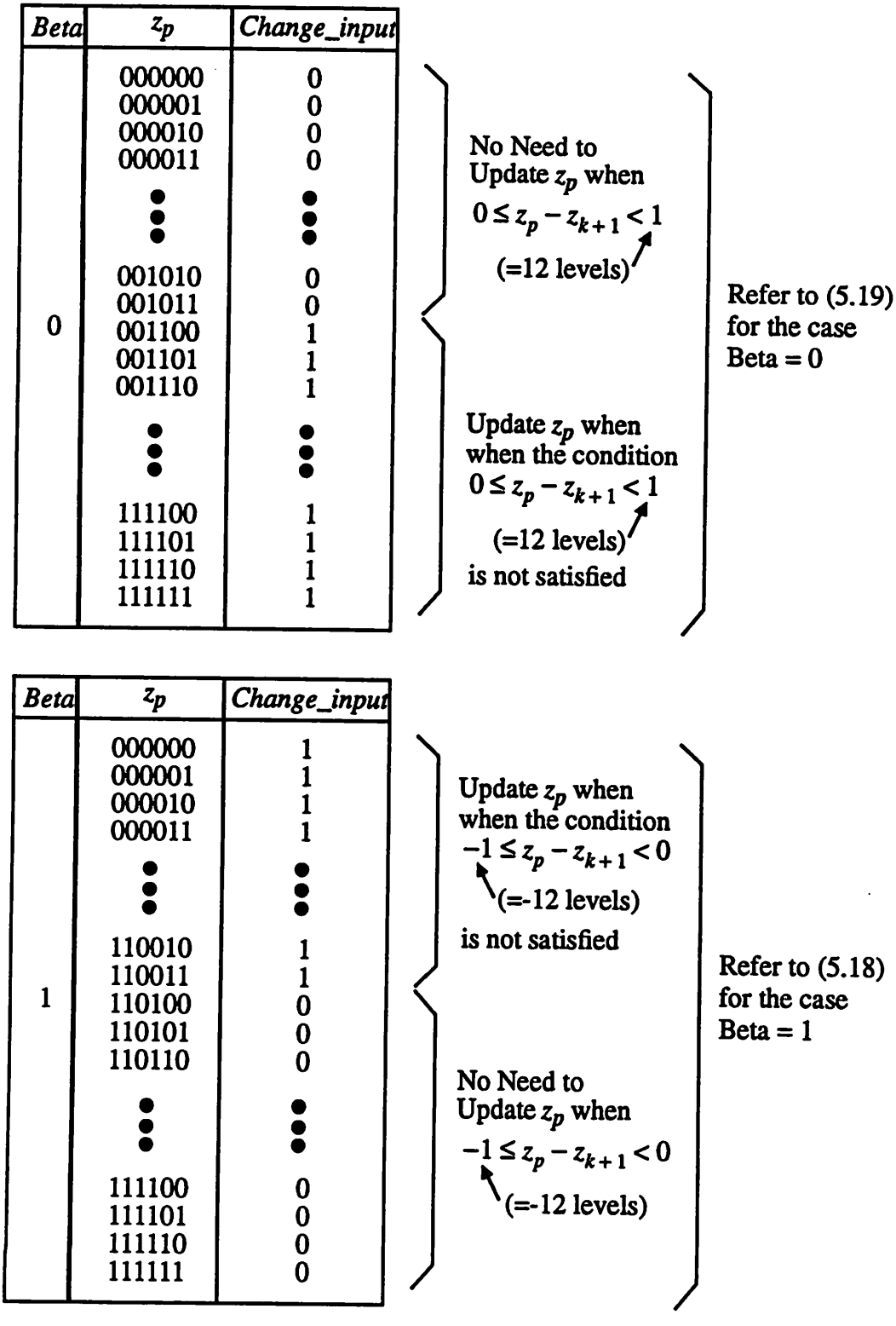


Figure 5.6 Truth Table for the Signal *Change_input*

At this point, since the signals *change_input* and *change_beta* are already available, the circuitry used to update the z_p and β can be described. The one used for updating z_p is shown in Figure 5.7. Note that the data stored at point A is z_{k+1}, the new input coming in from the adaptive filter, and z_p, the old input which causes the most recent merge, is stored at point B. All the data latches are either controlled by a 50MHz Master Clock or its inverted version.

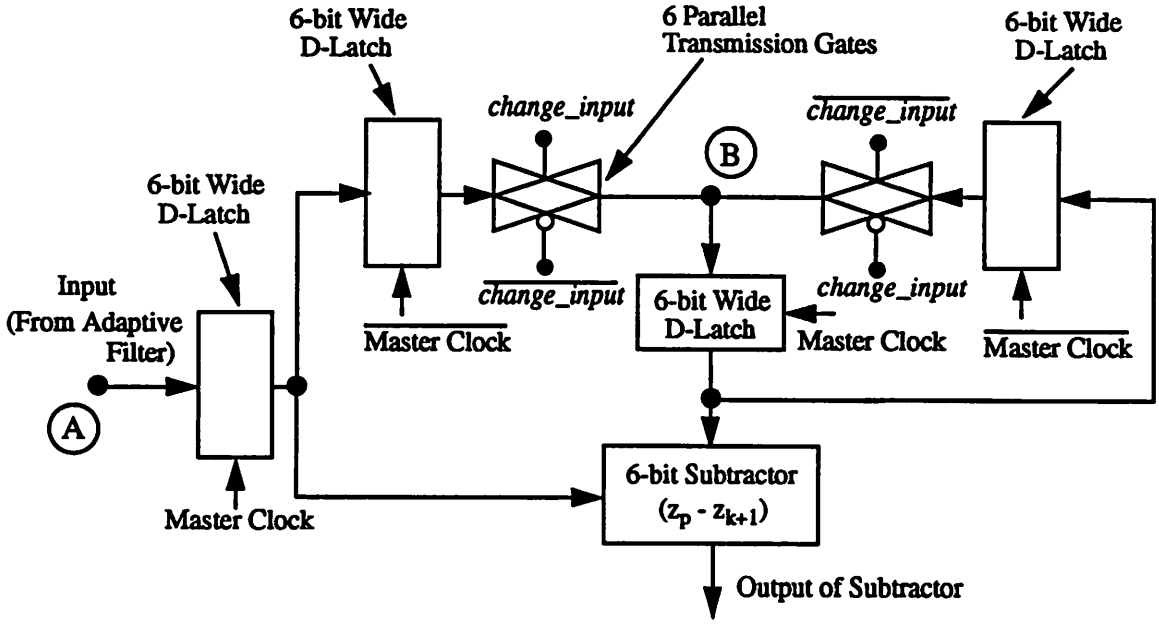


Figure 5.7 Circuitry for z_p (Old Input) Update

When the Master Clock goes high, both z_p and z_{k+1} are latched into the inputs of the subtractor. However, they are hold at the inputs of two sets of 6-bit wide D-latches which are clocked by the inverted version of Master Clock. Thus they are transferred to the inputs of the transmission gates when the Master Clock goes low after half a period (ie. 10ns). As mentioned, the signal *change_input* is a function of the subtractor output and the current state. When it is set to high, the transmission gates on the left are turned on while the

ones on the right are turned off. In this case, the current input, z_{k+1}, will be transferred to point B where it will be stored as the new z_p. On the other hand, if *change_input* is set to low, then the old input, z_p, will be stored at point B again, while the current input will be discarded. This "recursive" approach allows z_p to be re-used as many times as desired. The critical path mainly depends on the delays in the data path of the subtractor and the logic which generates *change_input.*

Figure 5.8 shows the circuitry used for updating β, the current state or the state at which the most recent merge occurs. The idea is very similar to the one used in updating z_p, except that it is simpler in this case since β carries one bit of information only. Since β is either 1 or 0, only an inverter is needed to generate a new state opposite to the current one. When *change_beta* is set to high, this new state will be stored as the new current state; otherwise, it will be discarded and the old state will be re-used. At this point, all the major functional units in the first part of the Viterbi decoder have been described. These functional units are mainly used to generate the signals needed in the second part of the decoder.

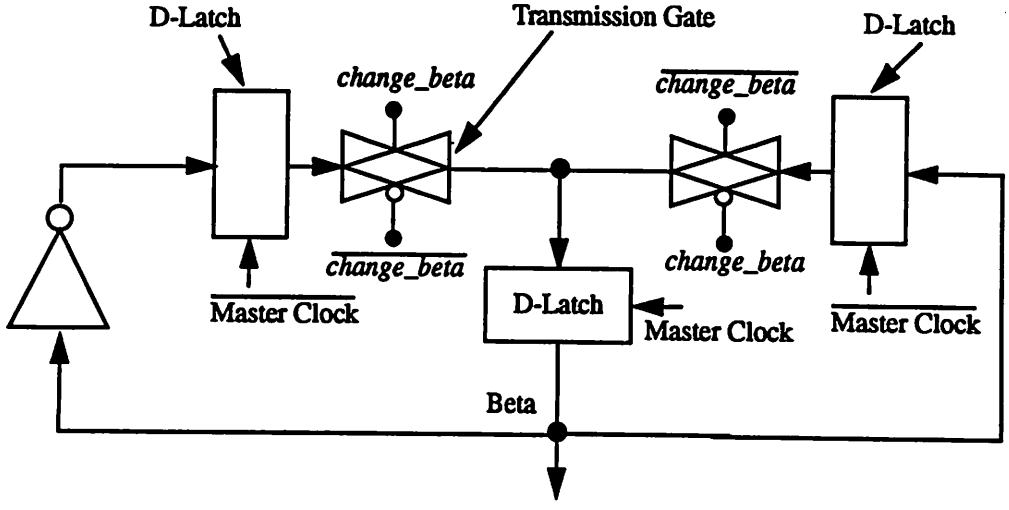


Figure 5.8 Circuitry for Beta Update

The second part of the Viterbi decoder mainly consists of two rows of shift registers [17]. Each of these registers is accompanied with random logic gates performing information exchange between the two register rows. It is shown in Figure 5.9. Recall that two sequences of data bits, one ends at state 1 and the other ends at state 0, need to be kept track of so that the accumulated branch metrics for both cases (L^+, L^-) will be available. The row on the top is used to store the sequence ends at state 0, and the bottom one stores the one ends at state 1.

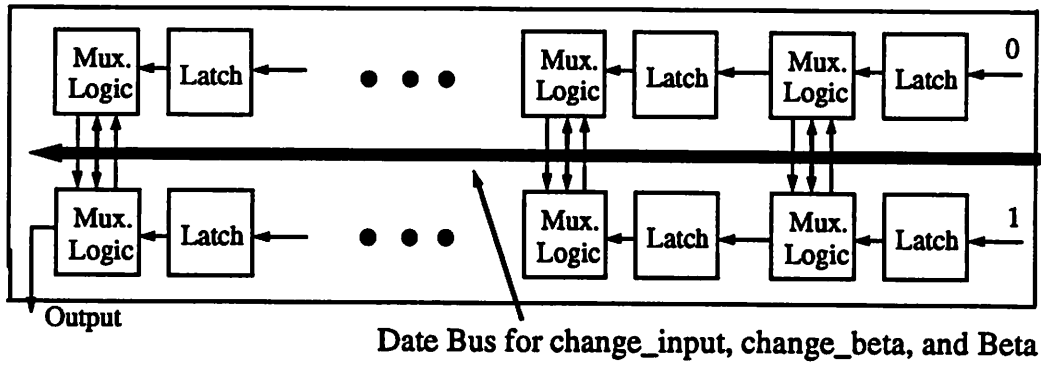


Figure 5.9 Two Rows of Shift Registers

The shift-register part is standard to most of the implementations of the Viterbi algorithm published before, and so only the key features will be described here. Suppose that another 1-bit signal, *copy_0to1*, will be generated. *Copy_0to1* is set to high when the sequence stored in the "zero" shift-register row needs to be copied to the "one" shift-register row, and it is set to low when the information needs to be transferred in the opposite direction. Now, assume that β, the current state, equals to 1. If the signal *change_beta* equals to 1, it indicates a merge at state 0. In this case, *copy_0to1* should be set to low. However, if *change_beta* equals to 0, then *copy_0to1* should be set to high, since a merge at state 1 occurs. Similarly, when β equals to 0, then *copy_0to1* will be set to high when

change_beta equals to 1, and set to low when *change_beta* equals to 0. From the brief description above, it is obvious that *copy_0to1* can be obtained from performing an exclusive OR (XOR) operation on β and *change_beta*. Since the multiplexing logic (see Figure 5.9) is very simple, there is no problem in achieving the required speed at all. Another important design consideration is the depth of the shift register rows. The longer the length of the rows, the larger the probability that their contents converge. In fact, it is a strong function of the kind of codes applied to the PR signal. As a trade-off between power and accuracy, a depth of ten was finally chosen.

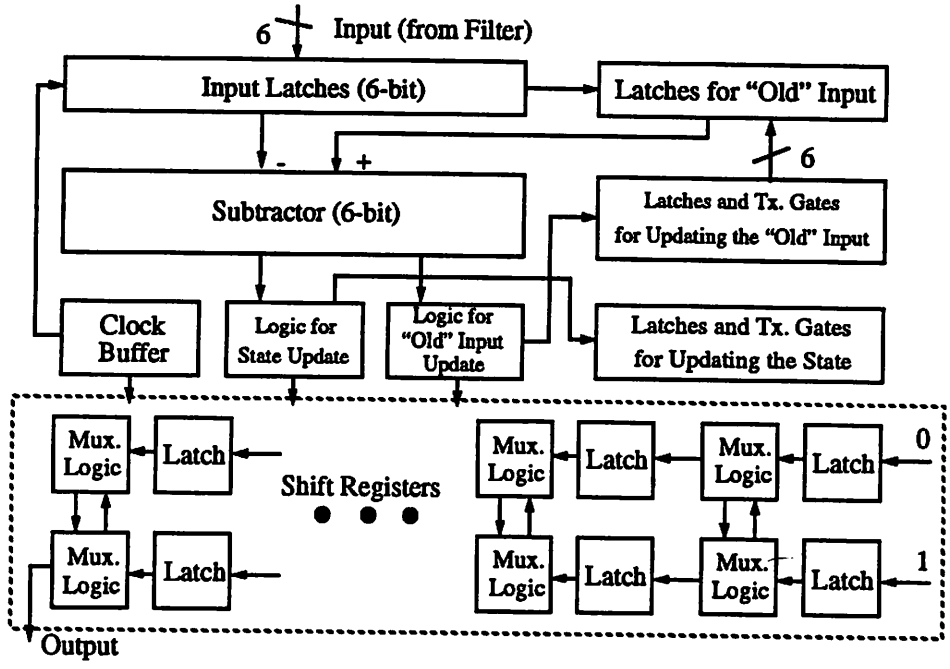


Figure 5.10 Block Diagram of the Viterbi Decoder

The block diagram for the entire Viterbi decoder is shown in Figure 5.10. In fact, the arrangement of the blocks corresponds exactly to that in the actual layout. SPICE simula-

tions show that the designed prototype can achieve a throughput of 100MHz with a power consumption of 20mW only.

Chapter 6

Experimental Results

6.1 Introduction

A chip was fabricated to measure the performance of the proposed DSP. The chip is composed of a total of approximately 70,000 transistors, and the total silicon area (including input and output pads) is 57,000 mil^2 (36.6 mm^2). Figure 7.1 shows the chip photo. The technology adopted is a $1.2\mu m$ single poly, double metal, N-well CMOS process. The prototype was fabricated through MOSIS.

6.2 Design of the Test Board

Figure 7.2 shows a simplified version of the test board. As mentioned in Chapter 4, on-chip RAM and ROM used for storing initial coefficients are omitted, since it is more convenient to down load the coefficients from external switches. These switches are found on the left of the chip. The middle coefficient (CM) is represented as a 5-bit word only, since the sign is always set to 0 (ie. positive) when the DSP is turned on. Similarly, the initial sign bit of the two coefficients adjacent to the main tap is always set to 1 (ie. negative). In addition, the step sizes (BPD and BMD) are also set by external switches. The adaptive equalizer is being trained when the initialization signal (IN) is set to high. The initial coefficients will be used in this training mode. When the IN signal is set to high, the coefficient update circuitry on the chip will be disconnected from the external switches, and the new coefficients will be determined by the update algorithm. The input to the DSP consists of two 50MHz data streams generated by a HP16520A pattern generator. The major outputs of the chip consists of the outputs of the two parallel Viterbi decoders and the outputs of the 4 parallel filters. The outputs of the Viterbi decoders are needed

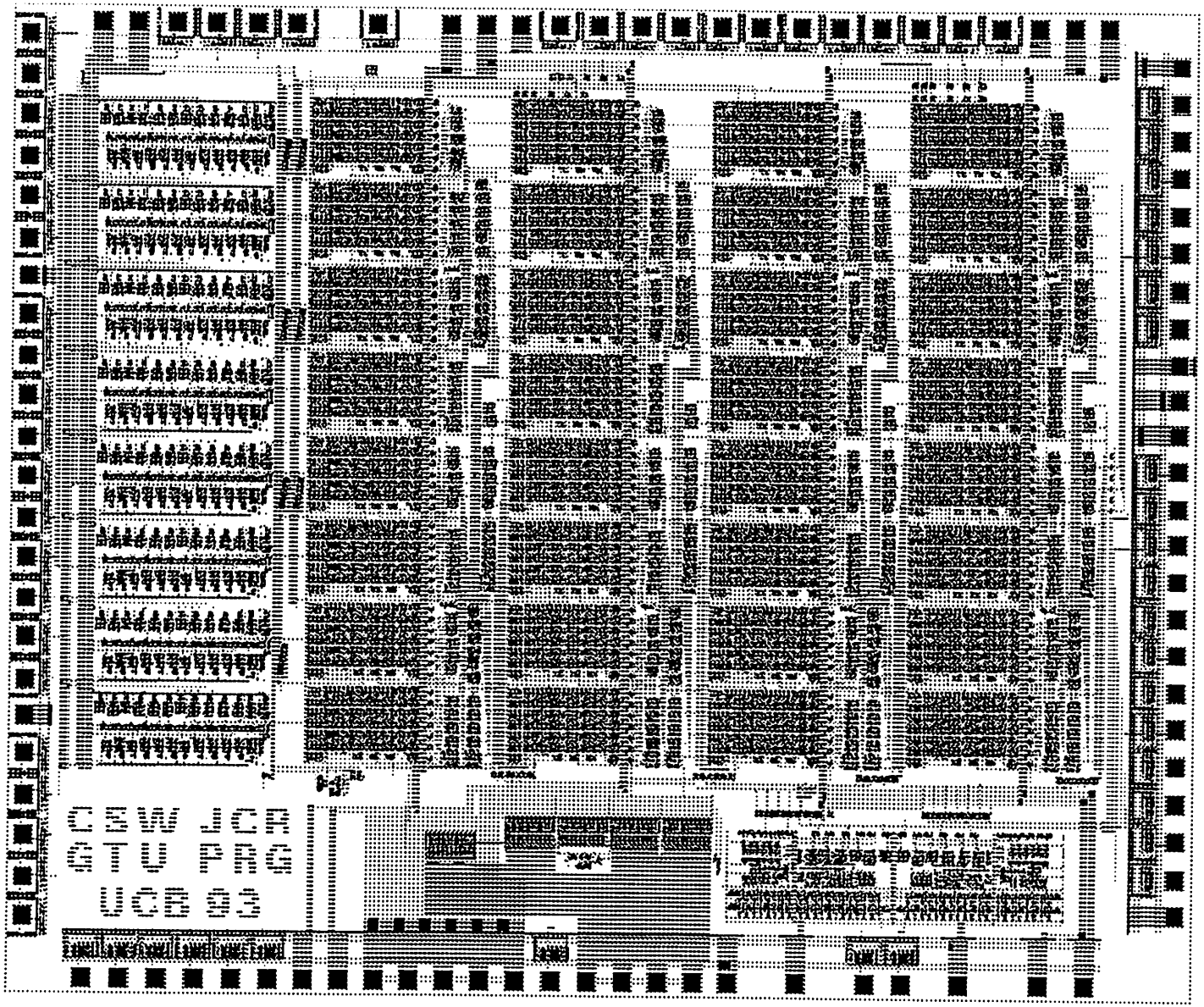


Figure 6.1 Chip Photo for DSP

to estimate the bit error rate (BER) of the DSP, and the outputs of the filters are required to determine the improvement in SNR provided by the adaptive equalizer. The Master Clock is also generated by the HP Logic Analyzer.

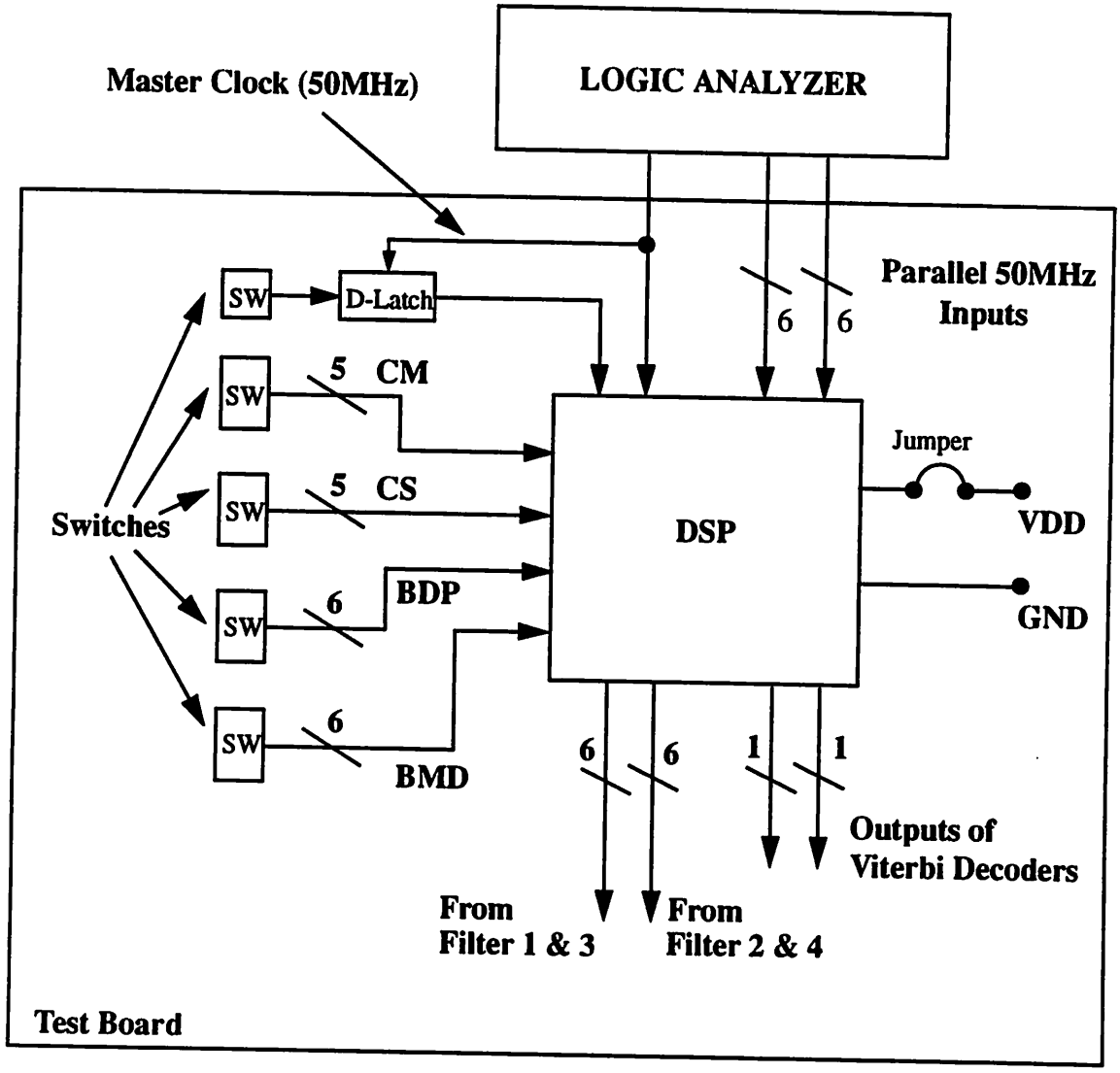


Figure 6.2 A Simplified Version of the Test Set-up

6.3 Summary of Experimental Results

By using the test set-up shown in Figure 6.2, the performance of the prototype was verified and the key results are summarized in this section. Note that testing results show that the prototype cannot achieve the target throughput at a supply voltage of 3.3V. Consequently, measurements are also taken at a supply voltage of 5V in order to demonstrate some of the ideas put forward in the previous chapters.

6.3.1 Power Consumption verse Operating Frequency

To verify that the DSP consumes dynamic power only, the chip power consumption is measured at various operating frequencies. The results are summarized in Table 6.1, and are also plotted in Figure 6.3. The linear dependence of power consumption on operating frequency shows that the DSP dissipates dynamic power only, as asserted in section 3.2.

Operating Frequency (MHz)	Power at Vdd = 3.3V (mW)	Power at Vdd = 5V (mW)
5	2.34	6.9
10	15.27	33.95
20	29	69.25
40	54.25	139.55
50	70.45	178.15
100	DNE	338.75

DNE = Data Not Exist

Table 6.1 Power Consumption verse Operating Frequency

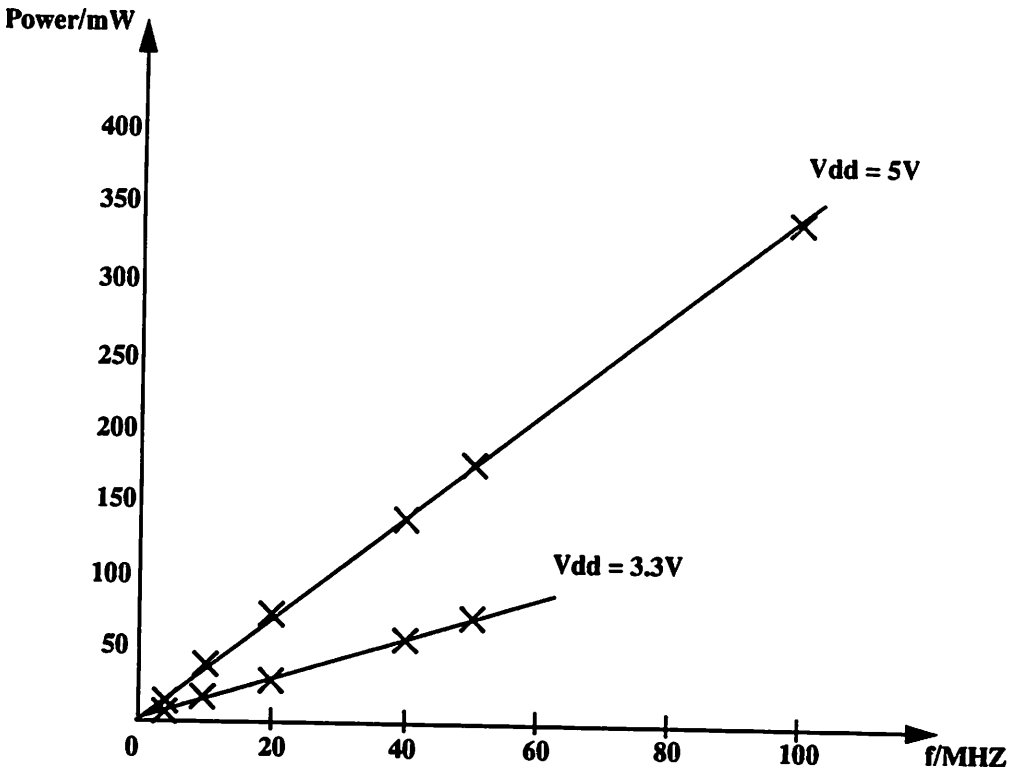


Figure 6.3 Power Consumption verse Operating Frequency

6.3.2 Power Consumption verse Supply Voltage

To investigate the effect of supply voltage on power consumption, the power consumptions of the DSP at various operating frequencies are measured at different values of supply voltage. The results, as shown in Table 6.2, show the quadratic dependence of power consumption on supply voltage, as stated in section 3.2.

Supply Voltage (Volts)	Power @ (mW) 50MHz	Power @ (mW) 40MHz	Power @ (mW) 20MHz	Power @ (mW) 10MHz
3.0	57.63	45.09	22.17	12.63
3.3	70.45	54.25	29.00	15.27
3.5	79.73	62.47	31.32	16.10
4.0	103.2	86.04	47.04	25.80
4.5	138.4	108.3	54.72	27.27
5.0	178.2	139.6	69.25	33.95

Table 6.2 Power Consumption verse Supply Voltage

6.3.3 Discrepancies between Simulation and Experimental Results

The major "upset" in this research project is the fact that the prototype cannot achieve the target throughput, 100 Mbits/sec, at a supply voltage of 3.3V. Because of the limitations imposed by testing set-up and the availability of equipment, the next lower operating frequency at which measurements can be taken is 50MHz. At this frequency, the filter adapts properly at a supply voltage of 3.3V with a power consumption of 70.5mW.

Although measurements cannot be taken at frequencies between 50 and 100MHz, it is estimated that the proposed DSP should be able to reach a thoughtful of 60 - 70 Mbits/

sec. Since it has been proven that the DSP consumes dynamic power only, the power consumptions at 60 and 70Mbits/sec can be extrapolated as 84.5 and 98.6mW respectively. In fact, pre-fabrication simulations show that the DSP can be run at 100MHz. Therefore, the actual experimental result is at least 30% off the expected value. By the time this report was being written, the reasons for this large discrepancy was still under investigation.

It was found that a mistake was made in modeling the fringing capacitance on the metal 1 and metal 2 layers. As a result of this modeling mistake, the capacitive loading caused by most metal lines is about three times as large as the modelled value. The exact amount of extra delay caused by this unmodelled capacitance loading was still under investigation when this report was prepared.

The second reason for the discrepancy is the ideal conditions that being assumed when the simulations were performed. For example, in all the simulations, the master clock was assumed to have a duty cycle of 50% and a rise/fall time of 1ns. However, the master clock that was actually used in the testing process has a rise/fall time of about 5ns and the duty cycle is certainly not 50%. In general, testing IC prototypes at a operating frequency as high as 50 - 100MHz is not trivial at all, and there are many possibilities that can adversely affect the testing results.

Based on preliminary experimental results, it is suspected that the propagation delay through the accumulator in the first filter is the speed limiting factor. As mentioned in chapter 4, the accumulator used is actually a binary tree arrangement of seven carry select adders. Therefore, to increase the throughput of the DSP by another 30%, it is necessary to use a faster adder structure. One possible option is the carry-lookahead adder.

6.3.4 Key Specifications of DSP

Summarizing the most important experimental results obtained, the key specifications of the DSP is shown in Table 6.3. The most important achievement of this research project is the demonstration of a parallel DSP architecture that can reach a throughput as high as 50Mbits/sec with a power consumption of only 70.5mW. In a recent publication, the CMOS logic in a BiCMOS 65Mbits/sec read channel IC dissipates 1W of power [19]. Although experimental data at 65MHz is not available as a result of the limitation imposed by testing equipment, it is predicted that the DSP is able to reach a throughput as high as 60 - 70Mbits/sec with an extrapolated power consumption in the range of 84.5 to 98.6mW only. In fact, this DSP has the lowest power-to-speed ratio reported to date.

Data Rate	**50 Mbits/sec**
Supply Voltage	**3.3V**
Power Dissipation	**70.5 mW**
Active Area	**5.5 mm X 6.6 mm (= 36.3 mm^2)** **217 mil X 260 mil (= 56.4 $kmil^2$)**

Table 6.3 Key Specifications of DSP

Chapter 7

Conclusion

7.1 Summary of Research Results

This research shows that the proposed parallel architecture can greatly enhance the throughput of a DSP, while minimizing the overall power consumption. In additional to magnetic disk drive systems, this architecture is also of potential interest for applications such as high speed data transmission on twisted pair and wireless communications, which requires low power consumption. The most valuable contribution of this research project is found in the design of the adaptive equalizer.

7.1.1 Parallel Adaptive Equalizer Architecture

In general, it is believed that the pipelined approach is a better solution for low-power design. The reason is that, in the pipelined approach, the major hardware overhead involved are merely the pipeline latches, which are cheap in terms of both area and power consumption. Furthermore, they are very easy to design. However, as explained in section 3.2, the parallel approach is actually preferable to the pipelined approach. There are two major reasons for this. The first reason is that the hardware can be clocked at lower speeds by using the parallel approach, while in the case of the pipelined implementation, the hardware is clocked at the target throughput rate. The second reason is that the proposed DSP consumes dynamic power only. These two reasons together explain the extra power overhead, $E_{p-latch} \times 1/T$, introduced by the pipelined implementation, as shown

in section 3.2.3. Also, since the parallel architecture requires more silicon area, it is believed the advantage of the parallel approach will be jeopardized by the excessive power consumption caused by the interconnects. However, as proved in chapter 4, by adopting more intelligent layout strategies, this undesirable extra power consumption can be minimized. For the same reason, the current adaptive filter can be easily extended to one with more parallel stages, if even higher throughput is desired.

7.2 Future Directions

In the future, it is predicted that the issue of IC power consumption will attract more and more attention. In certain applications such as portable computers and wireless communications, it is a "must". In most other applications, the manufacturers would also like to use low power specifications to market their products. Experience gained through designing the proposed DSP shows that the game of low-power design involves two major factors, lower supply voltage and better technology. For the supply voltage, the industrial standard has changed from 5V to 3.3V, and this trend will certainly continue. Lowering the supply voltage is the key to achieving significant power saving, because of the quadratic dependence of dynamic power consumption on VDD, as explained in section 3.2. Another advantage of using lower supply voltage is that, as the minimum channel length becomes shorter and shorter, it is important to keep the electric field across the channel within a certain limit to avoid punch-through and impact ionization. In additional to lower supply voltage, scaled technology can help to minimize power consumption. In the industry, sub-micron technology is now very common. Scaled technology simply means less area and less capacitance, and less capacitance means higher speed and lower power con-

sumption. It also alleviates the area penalty caused by the parallel architecture. Consequently, research in this area will have significant impacts on the future of low-power design.

References

[1] L. D. Stevens, "The evolution of magnetic storage," *IBM Journal of Research and Development*, pp. 663-675, Nov. 1981.

[2] J. M. Ciffio et al, "Adaptive equalization in magnetic-disk storage channels," *IEEE Communications Mag.*, pp. 14-29, Feb. 1990.

[3] J. J. Moon and L. R. Carley, "Partial response signaling in a magnetic recording channel," *IEEE Trans. on Magnetics*, vol. 24, no. 6, pp. 2973-2975, Nov. 1988.

[4] W. L. Abbott, J. M. Ciffio, and H. K. Thapar, "Channel equalization methods for magnetic storage," *ICC '89*, Boston MA., June 1989.

[5] W. L. Abbott, J. M. Ciffio, and H. K. Thapar, "Performance of digital magnetic recording with equalization and offtract interference," *ICC '89*, Boston MA., June 1989.

[6] F. Dolivo, "Signal processing for high-density digital magnetic recording," *Proc. VLSI and Comp. Peripherals*, pp. 1.91-1.96, Hamberg, Germany, May 1989.

[7] H. Kobayashi and D. T. Tang, "Application of partial response channel coding to magnetic recording systems," *IBM Journal of Research and Development*, pp. 368-374, July 1970.

[8] J. W. M. Bergmans, " Density improvements in digital magnetic recording by decision feedback equalization," *IEEE Trans. on Mag.*, vol. 22, no. 3, May 1986.

[9] A. P. Chandrakasan et al, "Low-power CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 27, pp. 473-484, April 1992.

[10] A. D. Booth, "A signed binary multiplication technique," *Q. J. Mech Appl. Math.*, vol. 4, pp. 236-240, 1951.

[11] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron. Comput.*, pp. 14-17, Feb. 1964.

[12] K. Hwang, *Computer Arithmetic, Principles, Architecture, and Design*, New York NY, John Willey, 1979.

[13] C. R. Baugh and B. A. Wooley, "A two's complement parallel array multiplication algorithm, " *IEEE Trans. on Computers*, vol. C-22, no. 12, Dec. 1973.

[14] Ultra-Low Power Library, Internal Documentation, University of California at Berkeley.

[15] M. J. Ferguson, "Optimal reception for binary partial response channels," *Bell Syst. Tech. J.*, vol 51, pp. 493-505, Feb. 1972.

[16] R. W. Wood and D. A. Petersen, "Viterbi detection of Class IV partial response on a magnetic recording channel," *IEEE Trans. Commun.*, vol. COM-34, no. 9, pp. 454-461, May 1986.

[17] H. Kobayashi, "Application of probability decoding to digital magnetic recording," *IBM J. Res. Develop.*, vol. 15, pp. 64-74, Jan. 1971.

[18] J. C. Rudell, Master Report, University of California at Berkeley. to be submitted in Spring 1994.

[19] R. Philpott et al, "A 7MB/sec, mixed signal, magnetic recording channel DSP using partial response signaling with maximum likelihood detection," in *IEEE Proceedings of Custom Integrated Circuits Conference*, May 1993.