

# A 550-MSample/s 8-Tap FIR Digital Filter for Magnetic Recording Read Channels

Robert Bogdan Staszewski, Khurram Muhammad, and Poras Balsara

**Abstract**—An area-efficient low-power and low-latency 550-MSample/s FIR filter for magnetic recording read channel applications is presented. A parallel direct type II architecture operates on real-time deinterleaved (even and odd) input data samples and employs a fast low-area multiplier based on selection of radix-8 premultiplied coefficients in conjunction with one-hot encoded bus leading to a very compact layout and reduced power dissipation. The chip has been fabricated using a 0.18- $\mu\text{m}$  L-effective CMOS technology and is currently being used in commercial applications.

**Index Terms**—Digital filter, FIR Filter, high performance, low power, read channel.

## I. INTRODUCTION

STORAGE capacity of media doubles every eighteen months while consistently demanding faster data retrieval rates. The critical electronic part in the high-speed data retrieval process is the *read channel* which culminates the technological advancement in communication theory, as well as high-speed analog and digital VLSI design. In order to meet the challenge of fast data transfer rate, new architectural and circuit design approaches are continually required as the cost and power dissipation of these chips is becoming a major concern.

Present-day read channels employ *partial response maximum likelihood* (PRML) equalization [1] in which spectral shaping of read-back signal is typically performed by a combination of an analog *continuous-time filter* (CTF) followed by a *finite-impulse response* (FIR) filter [2]. The FIR filter can be implemented in one of two ways: either in a sampled analog fashion [4] or in a pure digital fashion [9]–[12]. In the former case, the CTF and FIR are separated by a sample-and-hold circuit, whereas in the latter case, the two blocks are separated by an *analog-to-digital converter* (ADC). The main purpose of the CTF filter is to perform an antialiasing filtering in order to limit the spectral contents of the signal and noise beyond the Nyquist frequency. However, the CTF filters are inherently difficult to tune or adapt to the wide variability of magnetic media, read heads, front-end analog electronics and environmental factors. It is the FIR filter, nowadays, that plays the major role in achieving fine signal equalization to the desired PRML target [8]–[11]. Analog circuits also do not scale down very well, while in contrast, digital FIR filters with feature-rich *least mean square* (LMS) coefficient adaptation algorithms are continually becoming cheaper

with the advances in technology as they are easily scalable. For these reasons, use of CTF or analog FIR filters for modern magnetic recording read channels is gradually diminishing.

An efficient timing recovery in a read channel is critical for fast phase and frequency acquisition as well as acceptable *bit error rate* (BER) performance during subsequent data tracking. The FIR filter is an integral part of the PRML channel timing feedback loops since the fully equalized data samples at its output are used to extract timing information [2]. This is still the case even with systems in which the samples are directly equalized to *extended PR class IV* (EPR4) targets. In that case, the FIR output samples must be further processed in a *maximum likelihood sequence detector* to increase timing robustness. Therefore, it is imperative that these filters are implemented with as little latency as possible in order to ensure sufficient bandwidth and stability margin of the timing loops. In our opinion, the *transpose direct form II* architecture (as in [9]) is best suited to build high-speed and low-latency FIR filters. Other feedback loops, such as gain recovery and LMS adaptation of FIR coefficients, on the other hand, do not usually present such strict latency requirements and shall not be emphasized in this work.

In this paper, we present a 550-MSample/s FIR filter which combines various architectural and circuit design techniques in order to reduce the area, power dissipation, and latency of the filter. The presented architecture uses a novel parallel structure which increases the operational speed by a factor of two while keeping the overall increase in area to less than this factor. Fast radix-8 multiplication is employed using a select and add of pre-multiplied coefficients while using one-hot encoding in buses for reducing the power dissipation. Internal stage results are stored in latches rather than flip-flops for lower time overhead, area, and power dissipation. This also allows taking advantage of irregular critical path delays in different stages by borrowing time from less-critical stages. Together, these techniques result in a filter which requires less area and power dissipation than a conventional multiplier-based implementation.

## II. PARALLEL RADIX-8 FIR ARCHITECTURE

### A. Parallel Operation

Let  $\mathbf{y} \equiv \{y(k): k = 0, 1, \dots\}$  represent the output data stream of an  $N$ -tap FIR filter. Let  $T$  represent the sampled-system unit delay which is also equal to the data-rate clock cycle period. At time instant  $kT$  the output data sample is given by

$$y(k) = \sum_{n=0}^{N-1} c_n u(k-n) \quad (1)$$

Manuscript received December 29, 1999; revised March 29, 2000.

R. B. Staszewski and K. Muhammad are with Texas Instruments Incorporated, Dallas, TX 75243 USA (e-mail: b-staszewski@ti.com; k-muhammad1@ti.com).

P. Balsara is with the Department of Electrical Engineering, University of Texas at Dallas, Richardson, TX 75083 USA (e-mail: poras@utdallas.edu).

Publisher Item Identifier S 0018-9200(00)06438-6.

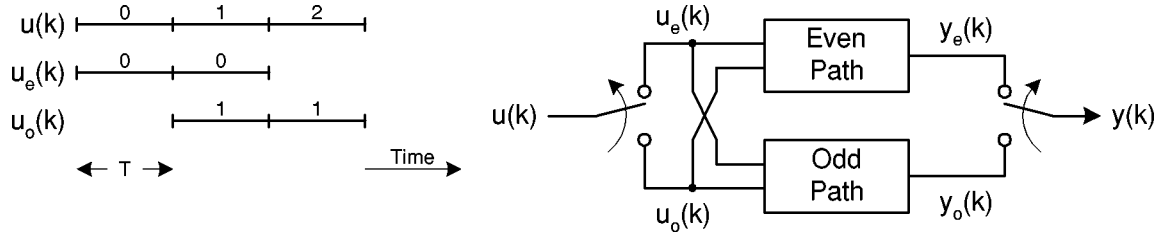


Fig. 1. Parallel FIR filter operation.

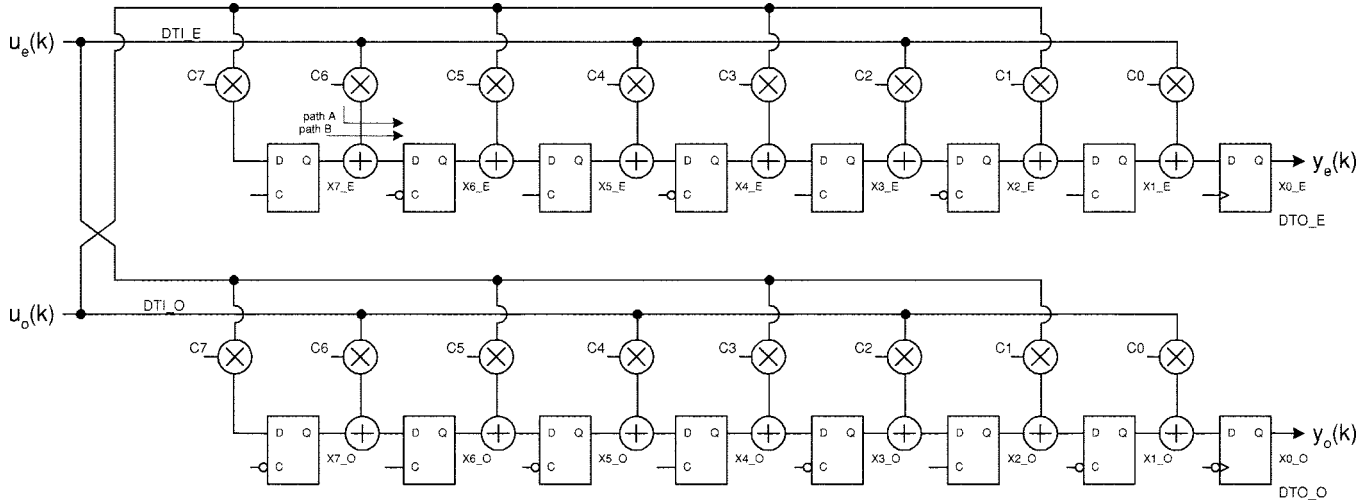


Fig. 2. Parallel transpose-type 8-tap FIR filter structure.

where  $c_n$  and  $u(k)$  represent the  $n$ th tap coefficient and the input data sample at time instant  $kT$ , respectively. A straightforward implementation of (1) performs this filtering operation at full data rate of  $1/T$ . Consequently, the input  $\mathbf{u}$  and the output  $\mathbf{y}$  data streams also run at the rate of  $1/T$ .

The basic idea behind the parallel architecture is to implement the operation of (1) at half the data rate, i.e.,  $1/2T$ . Let us define even and odd data samples derived from  $\mathbf{u}$  at time  $k$  as

$$u_e(k) \equiv \begin{cases} u(k) & \text{for } k \text{ even} \\ u(k-1) & \text{for } k \text{ odd} \end{cases} \quad (2)$$

$$u_o(k) \equiv \begin{cases} u(k) & \text{for } k \text{ odd} \\ u(k-1) & \text{for } k \text{ even} \end{cases} \quad (3)$$

Hence, as soon as the even data sample arrives, it is available to the even path. Similarly the odd data sample is also instantly available to the odd path. This can be accomplished by deinterleaving the  $1/T$  rate data samples to two streams using a  $1:2$  demultiplexer. The two data streams  $u_e(k)$  and  $u_o(k)$  operate at  $1/2T$  rate and are staggered in phase by  $T$ , as shown in Fig. 1. Then we can rewrite (1) as

$$y_e(k) = \sum_{n=0,2,\dots,N} c_n u_e(k-n) + \sum_{n=1,3,\dots,N} c_n u_o(k-n) \quad (4)$$

$$y_o(k) = \sum_{n=0,2,\dots,N} c_n u_o(k-n) + \sum_{n=1,3,\dots,N} c_n u_e(k-n) \quad (5)$$

$$y(k) = \begin{cases} y_e(k) & \text{for } k \text{ even} \\ y_o(k) & \text{for } k \text{ odd.} \end{cases} \quad (6)$$

We observe that the even and odd data streams can be viewed as data streams obtained by  $1:2$  deinterleaving operation while (6) performs the output data stream reinterleaving operation. The reinterleaving can be implemented with a  $2:1$  multiplexer. Equations (4) and (5) represent two separate FIR filters, each operating at  $1/2T$  data rate but with inverted phases, i.e., when one structure finishes its computational cycle, the other is in the middle of the next computational cycle. Each filter structure picks up the alternating time samples of  $1/2T$  rate inputs  $u_e(k)$  and  $u_o(k)$ . The deinterleaved output data streams  $\mathbf{y}_e$  and  $\mathbf{y}_o$  are defined similarly as the deinterleaved input data. The resulting parallel structure is shown in Fig. 1. This structure could also be arrived at by means of  $z^{-1/2}$  retiming using the standard interleaving techniques described in [3].

### B. Parallel Transpose-Type Architecture

Fig. 2 shows the basic idea of parallel transpose-type (*direct type II*) architecture. This structure is derived for real-time data where  $u_e(k)$  is an even deinterleave and appears at the upper input at even time slot  $k$  but is stable during the following odd time slot  $k+1$ . The data  $u_o(k)$  is an odd deinterleave and arrives at the lower input at odd time slot  $k$  but is stable during the following even time slot  $k+1$ . The deinterleaving operation is done without any latency loss and *directly* at the preceding ADC block using two arrays of registers: one clocked by the rising edge of the  $2T$  clock to produce the even ADC data and the other by falling edge of the  $2T$  clock to produce the odd ADC data. (Other deinterleaving method suggested when the ADC data is only available in a single stream format is to use two arrays of level-sensitive latches.) All storage elements

are latched by either positive or negative edge of a single clock, whose period is  $2T$ . Even though both edges are used, this architecture is insensitive to the clock duty cycle variation since all timing paths exist only between the same clock edge type. It should be emphasized that performing the ADC to FIR data handover at  $1/2T$  rate significantly relaxes interface timing requirements at higher frequencies of operation. These two blocks are designed using different circuit techniques, are separated on silicon in order to minimize noise coupling into the analog comparators by the digital gates, and usually employ different clock distribution networks. Consequently, their delay matching is quite poor, thus making it quite a challenging task at the high  $1/T$  rate.

In our application, the FIR filter output is not reinterleaved back into the single stream until much later. It may be advantageous to implement other parts of the read channel in parallel fashion, also operating at the  $1/2T$  rate. This allows high-speed design of peripheral circuitry due to parallel processing. However, the speed advantage may be obtained at the cost of extra area.

The FIR operational speed is doubled since *multiply-and-add* operation is now performed at half the data rate. An important advantage of this structure is that it allows computation sharing amongst the respective even and odd multiply operations in the two parallel paths. By using a numbering system that is higher than radix-2, some operations, such as coefficient premultiplication, can be made common to both parallel paths. Further speed gains are achievable due to amortization of *clock-to-Q* delay of the storage elements over two cycles. This effectively increases the useful time to do computation within a period. Consequently, this time can either be used to reduce area by using smaller and slower cells, or to increase the maximum operating frequency.

The proposed architecture naturally allows the application of latches in the internal clocking stages as a faster, smaller, and lower-power alternative to using flip-flops. Fig. 3 shows a timing relationship within a single stage where the storage elements are latches. There are two important timing paths: path *A* from the input  $u_e$ , and path *B* from the previous-stage output  $x_{n+1}$ . The third timing path from the coefficient input  $c_n$  is not an issue if the coefficients are semistatic, i.e., coefficient changes are needed only when mechanically moving to a different recording zone between read operations. Alternately, a clocking stage could be placed inside the multiplier (it will be shown later that in combination with the Booth encoding of the data this could be easily done at the pass gates that form a multiplexer) such that it falls well below the critical path. All these paths terminate at the data input of the storage element at the current stage  $n$ . In the timing diagram,  $t_{clk-Q}$  is the *clock-to-Q* delay of the previous-stage storage element,  $t_{su}$  is the setup time of the current-stage storage element,  $t_A$  is the delay of the combinatorial path *A* from the data input, and  $t_B$  is the delay of the combinatorial path *B* from the previous stage. During the time when the clock is high, the previous-stage latch is transparent, thus allowing the LSB bits of  $x_{n+1}$ , which get calculated earlier than the MSB bits, to pass to the following stage early in the computational cycle. Consequently, the combinatorial delay  $t_B$  from the previous stage differs for

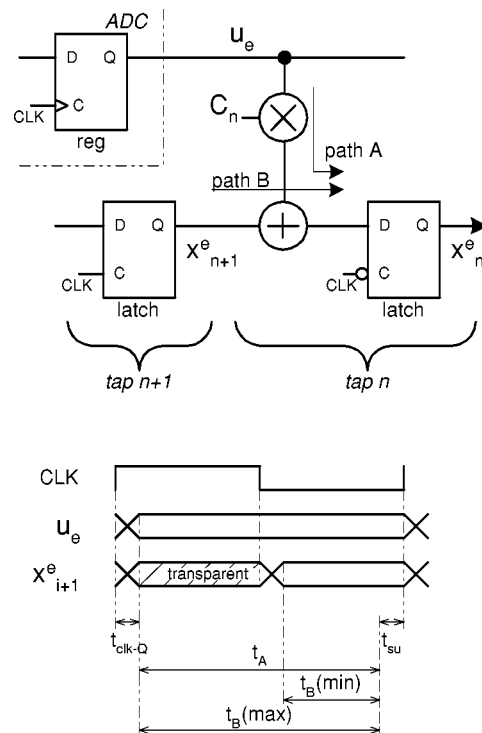


Fig. 3. Timing of a single tap.

various  $x_{n+1}$  bits and is contained within the limits of  $t_B(\min)$  and  $t_B(\max)$ . It should be mentioned that the latches could be replaced with registers. However, this makes a poor choice because one cannot take advantage of the advance propagation of the calculated LSB bits to the next stage and, consequently, this results in lowering of  $t_B(\max)$  to  $t_B(\min)$ . However, it is acceptable for the path *B* to have less computational time than path *A*, because the interstage computation part is obviously faster, so that the above concern might not be an issue. Since tap 0 is the last stage, there is no need for the advance propagation of bits. In such a case one should use registers as the clocking elements.

The proposed architecture takes advantage of the normal irregularity of critical path delays between neighboring stages by borrowing timing slacks from the less time-critical taps, i.e., taps that are earlier in the accumulation cycle, with tap 7 having the least complexity. As a result, the operational throughput could be more than doubled with less than twice hardware cost and area. This is because the register overhead (*clock-to-Q* delay and setup time) for the desired application was found to be 33% of the clock period. However, by using the proposed structure, this overhead reduces to 16.5% of the half-rate clock and allows speed-up by a factor slightly greater than two. This is in sharp contrast to pipelining which is normally used to achieve higher speed. Pipelining also adds latency, area, and power overhead since extra latching or reclocking stages for every pipelining order are required. These stages also add complexity to the clock tree. In addition, the circuit is still clocked at the same high frequency as the data rate which increases the dynamic power dissipation. Our scheme alleviates these problems without the need of input buffering as even and odd data samples are applied at the respective inputs exactly at the time when they are available.

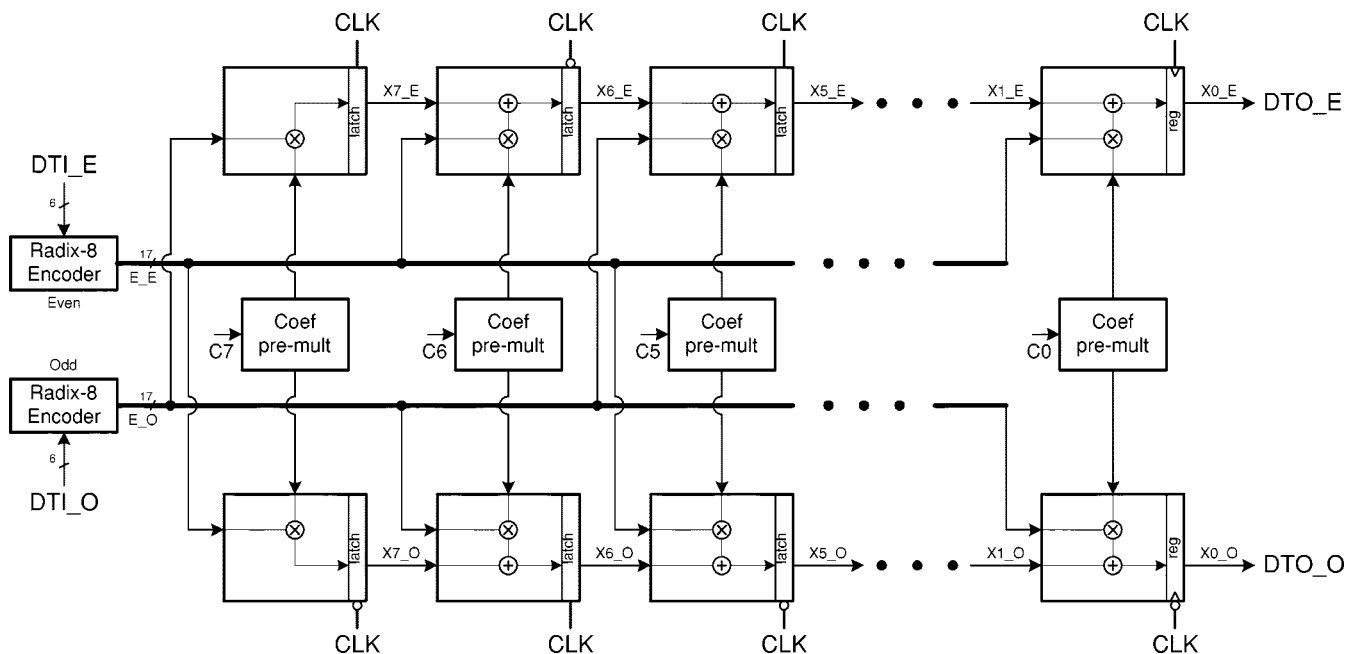


Fig. 4. Parallel 8-tap FIR filter with radix-8 encoding of input data.

It is also reported in [10] that the parallel approach is preferable to pipelining from the power dissipation standpoint.

The proposed architecture features only one extra full-rate ( $1/T$ ) clock cycle latency, due the parallelizing of operations, as compared with an ideal zero-propagational-delay FIR filter.

### C. Radix-8 Booth Architecture

In the proposed architecture, the incoming high-speed 6-bit data is encoded into radix-8 numbering system. Advantage is taken of the “natural” 6-bit resolution of the data samples in read channel applications such that a simple addition operation of the resulting two partial products is thus equivalent to the full multiplication of data and coefficient. In combination with the FIR transpose structure, the basic *multiply-and-accumulate* operation is thus replaced by an *add-and-accumulate* operation, as follows:

$$x_n(k) = c_n u(k) + x_{n+1}(k) \quad (7)$$

$$= \left[ 2^3 c_n \left( \left\{ -4u^{(5)} + 2u^{(4)} + u^{(3)} + u^{(2)} \right\} (k) \right) + c_n \left( \left\{ -4u^{(2)} + 2u^{(1)} + u^{(0)} \right\} (k) \right) \right] + x_{n+1}(k) \quad (8)$$

$$= [2^3 b_n(k) + a_n(k)] + x_{n+1}(k) \quad (9)$$

where

$x_n$	$n$ th tap output
$u^{(i)}$	$i$ th bit of $u$
$a_n$	lower partial product
$b_n$	upper partial product.

The main advantage of radix-8 encoding of data is that the  $3X$  coefficient pre-multiplication is performed off the critical path. The radix-8 *encoding of data* is in contrast with the previously published work in which either radix-4 Booth *encoding of coefficients* is utilized [6]–[9], [11] or no reduction of partial products is performed [10]. Encoding data allows reduction of area

by sharing of resources. Fig. 4 shows the basic concept where both parallel paths operate on alternating input samples and the coefficient pre-multiplication is shared. The physical format of the encoded data for each of the parallel paths consists of two buses. The first bus is a collection of nine wires and is a function of the higher-order 4 bits of the original input data. The second bus is one wire smaller, 8-bit wide, and is a function of the lower-order 3 bits of the original input data. One bit is shared between the two encoded numbers and leads to a redundant arithmetic system. The bits within each bus are encoded in one-hot manner, meaning that at all times an exactly one bit is asserted. This reduces the power dissipation as well as area since both buses run straight to all taps of the FIR filter resulting in a compact layout (see Fig. 7). It further contributes to the performance increase by allowing use of a simple pass-gate multiplexer structure with low propagation delay (Fig. 6).

Each FIR coefficient is pre-multiplied for the following cases:  $-4C$ ,  $-3C$ ,  $-2C$ ,  $-C$ ,  $0C$ ,  $C$ ,  $2C$ ,  $3C$ , and  $4C$ , where  $C$  is the coefficient value. The  $0C$  (zero) and power-of-two pre-multiplications are trivial. Similarly  $-2C$  and  $-4C$  cases are a simple left shift operation of the prenegated  $-C$  coefficient. As a result, only the negation ( $-C$ ) and multiplication-by-three ( $3C$ ) non-trivial operations are required. The multiplier concept diagram is shown in Fig. 5. Since the FIR coefficients in read-channel equalization do not usually change at high rate, pre-computation does not require the high-speed operation of coefficients pre-multiplication. Hence, the critical path of the multiplier does not include the delay of the pre-multiplication. The multiplexer cell in Fig. 6 has a compact layout and has a low average switched capacitance. The high-speed encoded data lines are connected to the gate of nMOS transistors which present a much lower capacitance than the nMOS transistor source inputs and their connected drains. This significantly reduces power while allowing the cell to operate at high speed. The combination of one-hot operation of each bus with the above pass-gate based multiplexer

TABLE I  
 $\mu\text{W}/\text{MSPS}/\text{TAP}/\text{INBITS}/\text{COEFF-BITS}$  FIGURE OF MERIT [9].

Paper	Gate ( $\mu\text{m}$ )	$\mu\text{W}/\text{MSPS}/\text{Tap}/\text{InBits}/\text{Coeff-Bits}$	$P_{DISS}$ (mW)	$f_{sample}$ (MSPS)	Taps, InBits, Coeff-Bits	Area ( $\text{mm}^2$ )	#inter-leaves	direct type
This	0.18	0.23	36	550	8, 6, 6	0.3	2	II
Thon [9]	0.8	6.16	426	240	8, 6, 6	2.9	1	II
Wong [10]	1.2	4.86	50	50	8, 6, 6	36.4	4	I
Ki [11]	0.8	4.86	140	100	8, 6, 6	5.85	1	I
Moloney [12]	0.7	6.25	165	200	5, 6, 4.4	1.1	2	I

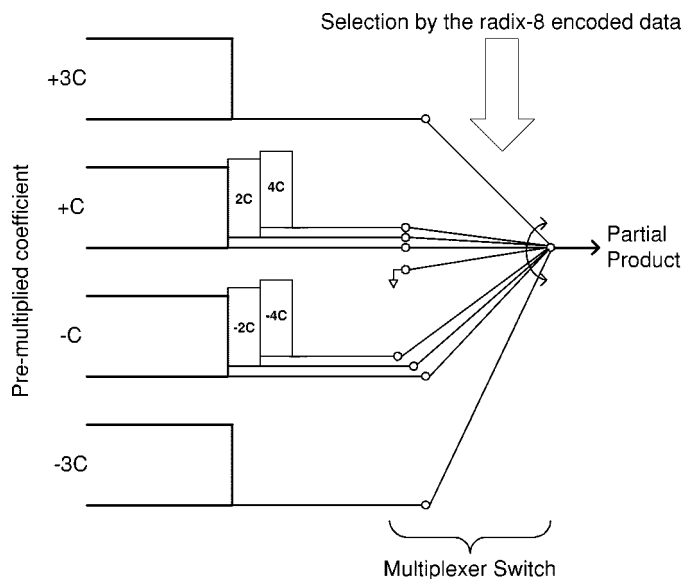


Fig. 5. Radix-8 multiplier based on selection of pre-multiplied coefficients.

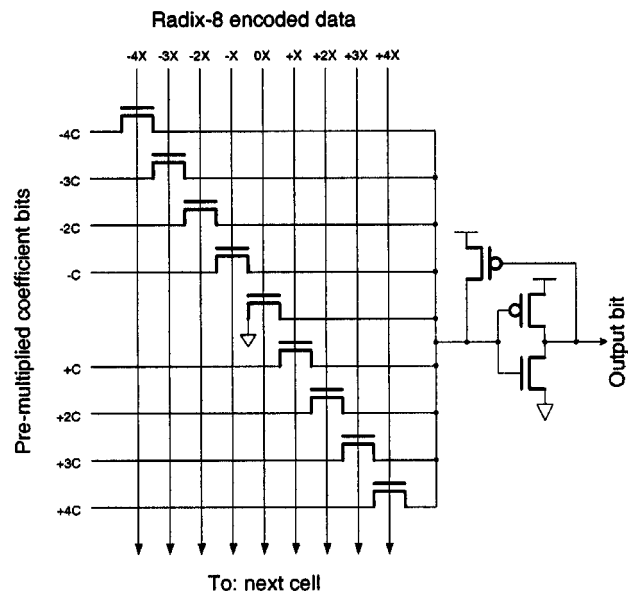


Fig. 6. Multiplexer cell.

significantly reduces the average switched capacitance and results in a fast and low-power multiplier.

In case faster pre-computation of FIR coefficient is desired, the coefficients pre-multiplication operation could be easily re-timed through pipelining. The resulting coefficient update latency of one or two clock cycles is negligible as compared with the slow rate of the LMS adaptation and does not appear in the critical path once the filter has adapted. Since the coefficients are semi-static, during normal operation this extra latency can simply be ignored.

### III. STANDARD CELL DESIGN

This FIR filter is an integral part of a larger read channel device and we decided to follow, as with all other digital blocks, a commercial CMOS standard cell [14] digital flow methodology. The design was modeled using a VHDL description code for system simulations (described in [13]), was synthesized from a *register transfer level* RTL description subset of VHDL and then auto-placed and auto-routed with timing-driven constraints, together with other high-speed digital back-end blocks, such as an LMS coefficients adaptation block, phase and gain detectors, etc. A few standard cells were created specifically for this design and made part of the tactical cell library: the 9-way multiplexer cell of Fig. 6 and its variants, and the low-power latch. The intended floor-plan forced on the auto-place tool, with some help of cell grouping regions, is shown in Fig. 7.

### IV. COMPARISON RESULTS

Table I compares this filter by earlier reported work in a format adapted from [12]. The entries in the third column indicate a metric for efficiency in power reduction, first proposed by [9]. In the table, *Coeff-Bits* represent the bits in the coefficients, *InBits* represent the bits in the input data,  $P_{DISS}$  is the dissipated power and  $f_{sample}$  is the sampling frequency equal to  $1/T$ . There are many factors which contribute to the reduction in the metric shown in the third column. Most of the reduction results from the process advancement. This can be seen by comparing the areas which indirectly reflect the reduction in the average switched capacitances. It is rather a difficult task to accurately compare the area and power improvements of this deep submicron design versus the published implementations based on conventional CMOS process technologies if based only on the feature size metric. Firstly, the interconnect parasitics are now becoming predominant. Secondly, because of its commercial nature, the stated operational speed is guaranteed under the worst-case process/temperature/voltage conditions. Thirdly, the operational speed comes from marketing requirements, and this often leads to disproportional increase in the area and dissipative power just to achieve the sufficient but small improvements in performance.

We also note that the implementation in this paper does not use carry-save arithmetic as in [9]. This reduces the number of registers by a factor of 2 as compared to [9]. In addition, the latency is also reduced by avoiding the merging of carry-save outputs.

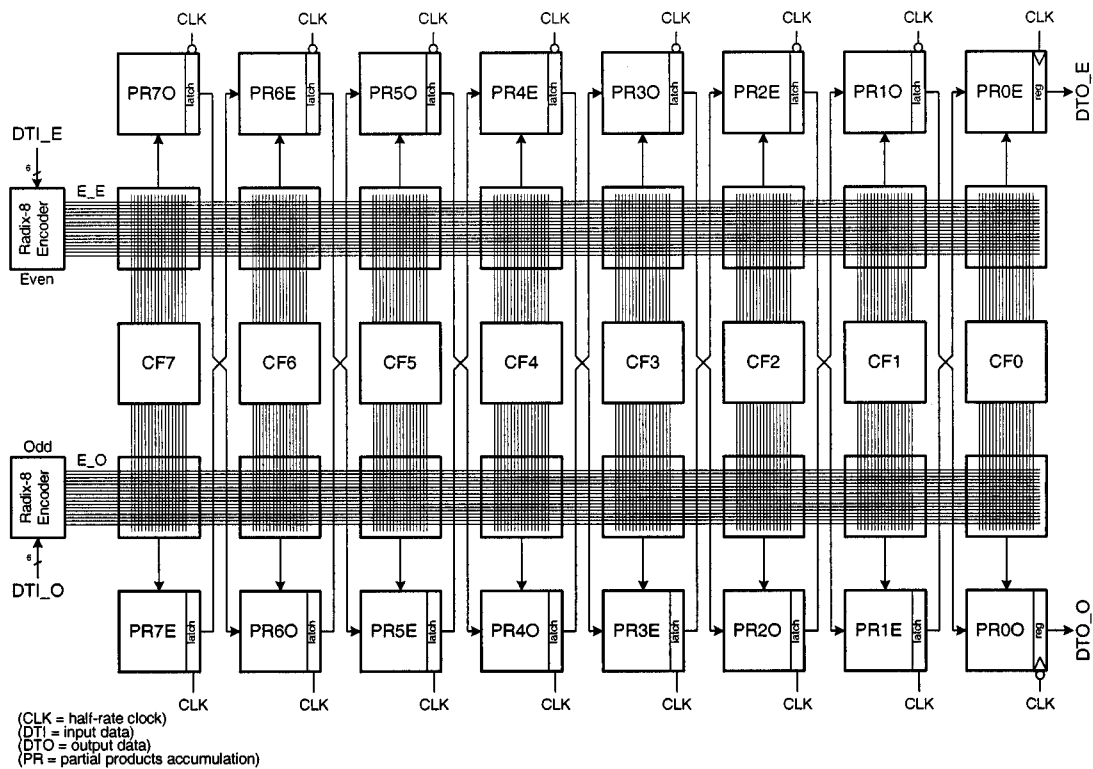


Fig. 7. Floorplan of the 8-tap FIR filter.

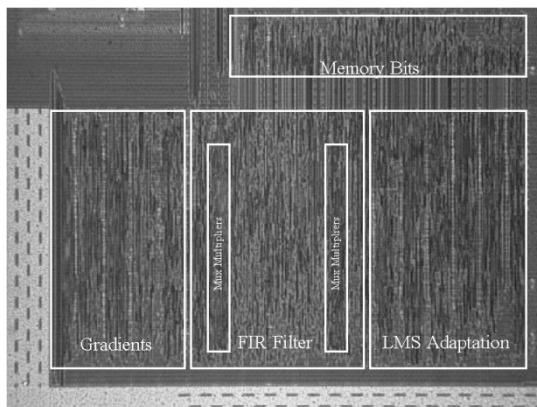


Fig. 8. Chip micrograph of the FIR filter area.

The micrograph of the FIR, LMS and Gradients area of the larger Read Channel IC chip is shown in Fig. 8. The measured results are reported in Table I. The read channel chip is manufactured in a  $0.7 \mu\text{m}$  pitch four-layer metal,  $0.18 \mu\text{m}$  L-effective,  $40 \text{ \AA}$  gate oxide thickness, digital CMOS process utilizing a standard cell library [14]. The nominal voltage supply is 1.8 V. It is being used in commercial hard-disk drives.

## V. CONCLUSIONS

A novel design of a high-speed digital FIR filter for read channel applications that combines a direct type II parallel FIR structure with radix-8 encoding of input data has been presented. This architecture achieves merely one full-rate clock cycle latency over an ideal FIR filter. Advantage is taken of possible circuit sharing between the parallel paths such that the

speedup factor of slightly larger than two is achieved at the cost of less than doubling of area. The architecture naturally lends itself to the use of latches in a manner that does not compromise testability.

## REFERENCES

- [1] H. Kobayashi and D. Tang, "Application of partial-response channel coding to magnetic recording systems," *IBM J. Res. Develop.*, vol. 14, pp. 386–375, July 1970.
- [2] R. Cideciyan *et al.*, "A PRML system for digital magnetic recording," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 38–56, Jan. 1992.
- [3] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*. New York, NY: Wiley-Interscience, 1999.
- [4] S. Kiriaki *et al.*, "A 160-MHz analog equalizer for magnetic disk channels," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1839–1850, Nov. 1997.
- [5] C. Baugh and B. Wooley, "A two's complement parallel array multiplication algorithm," *IEEE Trans. Comput.*, vol. C-22, pp. 1045–1047, Dec. 1973.
- [6] S. Mita *et al.*, "A 150 Mb/s PRML chip for magnetic disk drives," in *ISSCC Dig. Tech. Papers*, Feb. 1996, pp. 62–63.
- [7] W. Abbot *et al.*, "A digital chip with adaptive equalizer for PRML detection in hard-disk drives," in *ISSCC Dig. Tech. Papers*, Feb. 1994, pp. 284–285.
- [8] D. Pearson *et al.*, "250 MHz digital FIR filter for PRML disk read-channels," in *ISSCC Dig. Tech. Papers*, Feb. 1995, pp. 80–81.
- [9] L. Thon *et al.*, "A 240 MHz 8-tap programmable FIR filter for disk-drive read channels," in *ISSCC Dig. Tech. Papers*, Feb. 1995, pp. 82–83.
- [10] C. Wong *et al.*, "A 50 MHz eighth-tap adaptive equalizer for partial-response channels," *IEEE J. Solid-State Circuits*, vol. 30, pp. 228–234, Mar. 1995.
- [11] H. Ki *et al.*, "A high-speed, low power 8-tap digital FIR filter for PRML disk-drive read channels," in *ESSCIRC '97 Conf. Proc.*, Sept. 1997, pp. 312–315.
- [12] D. Moloney *et al.*, "Low-power 200-Msps, area-efficient, five-tap programmable FIR filter," *IEEE J. Solid-State Circuits*, vol. 33, pp. 1134–1138, July 1998.
- [13] R. Staszewski and S. Kiriaki, "Top-down simulation methodology of a 500 MHz mixed-signal magnetic recording read channel using standard VHDL," in *Proc. 1999 Behavioral Modeling and Simulation Conf.*
- [14] Texas Instruments Application Specific Integrated Circuits Macro Library Summary, "TSC6000 0.18- $\mu\text{m}$  CMOS Standard Cells," 1998.