Area-Time Tradeoffs in FIR Digital Filters with Broadcast and Pipelined Designs

Ding-Ming Kwai and Behrooz Parhami

Department of Electrical and Computer Engineering
University of California
Santa Barbara, CA 93106-9560, USA

Abstract – Previous designs of programmable finite impulse response (FIR) digital filters have demonstrated that the use of broadcast input data and control can lead to a high performance cost ratio. As we move towards widespread utilization of submicron technologies, such an approach should be reexamined by taking the effect of interconnections into account. In this paper, we show that the contribution of interconnect delay to the cycle time is no longer negligible and will hamper the scalability of the broadcast design. As an alternative, we propose a fully pipelined design in which both data and control signals are restricted to local connections. One important feature of our design is that the data input port can be reused to deliver the coefficients. Hence, the coefficients can be loaded in bit-parallel form with no increase in the number of input pins.

I. Introduction

Previous designs for programmable finite impulse response (FIR) digital filter are essentially based on the transposed direct-form realization. Input data are broadcast to all filter taps and the output responses computed in pipelined fashion [3]–[7], [9]–[11]. Coefficients are loaded by connecting all coefficient registers into a chain which is controlled by a common enable line to activate the shifting. The choice of broadcast control scheme, as a simple extension of the input data bus, makes the control broadcast overhead relatively insignificant, in view of the latter already being in place.

In this paper, we propose a fully pipelined design in which both data and control signals are pipelined through the filter. The pipelined filter's latency of N clock cycles, where N is the number of taps, is the same as that of the broadcast design. However, a comparison of the two designs assuming a constant cycle time is utterly unfair. For the broadcast design, the cycle time is dependent on the filter size, since as N grows, longer interconnects will be involved and larger loads must be driven. While it is conceivable that despite the broadcast overhead increasing the cycle time, the area saving might still lead to a cost-effective design, realistic analyses require an architecture and technology dependent model to justify such a tradeoff.

Using previously established 1.0μm and 0.5μm CMOS technologies, we show that the broadcast design may cease to be cost-effective as the dimensions of devices in integrated circuits are scaled down. The reason is that the increased area to accommodate more pipeline registers can be compensated for by the higher operating frequency, thus maintaining the throughput rate with scaling.

Our presentation is organized as follows. In Section II, we categorize four modes of data and control flows that can be used in designing programmable FIR filters. Section III derives our fully pipelined design. Section IV discusses the cost-performance tradeoffs and the effects of scaling on the above designs. Section V contains our conclusions.

II. Data and Control Flow

For a programmable design, mechanisms must be provided to load and store the coefficients. Given that data and control signals can be either broadcast to or pipelined through the filter, four categories of design can be distinguished.

1. Broadcast control, broadcast data (BCBD): The input data are broadcast to all filter taps and the coefficients are pipelined through the registers via a separate input port. The common "enable" signal is asserted when the coefficient registers are filled up with new values which are inserted in order from $h(0)$ to $h(N-1)$ (see Fig. 1 (a)).
2. Pipelined control, broadcast data (PCBD): The coefficients share the same input port with broadcast data. The control signal is shifted through the filter, sequentially enabling the coefficient registers to store the new values in order from $h(N-1)$ to $h(0)$ (see Fig. 1(b)).
3. Broadcast control, pipelined data (BCPD): The input data and coefficients are pipelined through all filter taps using two separate input ports. The common "enable" signal is asserted until the registers are filled up with the coefficients which are inserted in order from $h(0)$ to $h(N-1)$ (see Fig. 1(c)).
4. Pipelined control, pipelined data (PCPD): The coefficients share the same input port with pipelined data. If all the coefficients are inserted in order from $h(N-1)$ to $h(0)$ before the assertion of control and insertion of data, the pipelined control signal arrives at a filter tap just when its coefficient reaches the register (see Fig. 1 (d)).

Although the PCBD mode has the advantage in that the coefficients can be loaded without increasing the number of pins, the sharing of data input port can seriously aggravate the data broadcast overhead.

0-7803-3694-1/97/$10.00 © 1997 IEEE

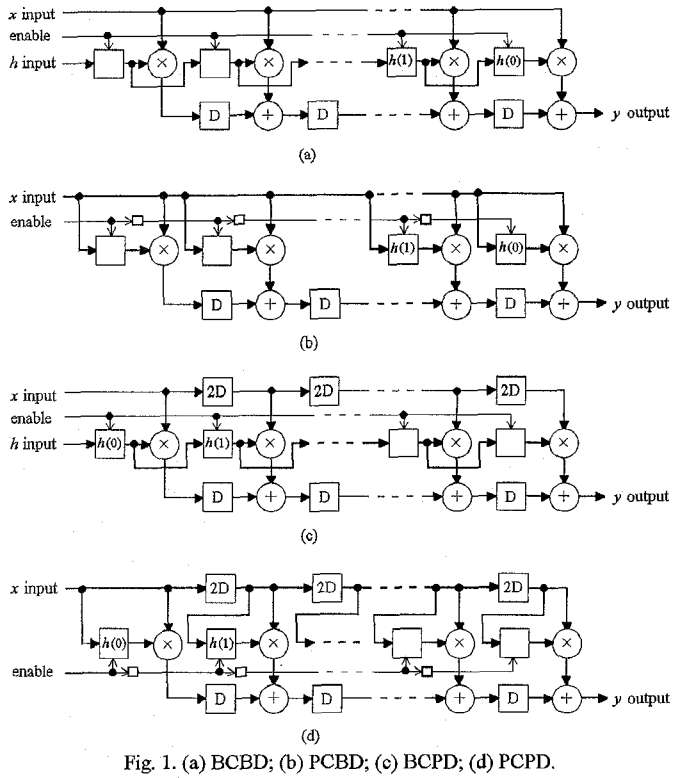

Fig. 1. (a) BCBD; (b) PCBD; (c) BCPD; (d) PCPD.

Most existing designs ignore the control broadcast overhead and adopt the BCBD mode. Their focus, instead, is on optimizing the multiply-and-add operation which leads to the reduction of the circuit depth in each cell. Ironically, the speed gained by such reductions aggravates the effect of propagation delay on long wires. Note that even if the BCPD mode is used, broadcast overhead is not totally alleviated, since we need to broadcast the control signal.

III. Fully Pipelined Design

The fully pipelined design is derived from the direct form by successively applying retiming transformations [8]. Fig. 2 shows the retimed realization of FIR digital filter of order N. The same structure can also be derived by using a dependence method. We refer the reader to [2] for other variants.

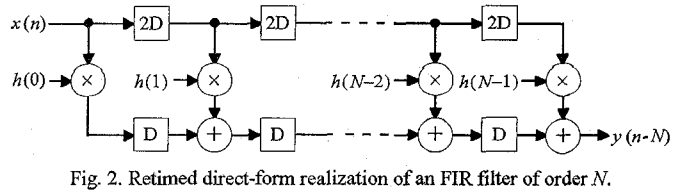

Fig. 2. Retimed direct-form realization of an FIR filter of order N.

A. Derivation of Pipelined Control Signals

We now derive the pipelined control signals in the PCPD mode. The pipelining of control signals with the data signals can be viewed as attaching control tags to data streams $\boldsymbol{X} = \{x(0), x(1), \ldots\}$ and $\boldsymbol{Y} = \{y(0), y(1), \ldots\}$. The coefficients, in the order $h(N-1), h(N-2), \ldots, h(0)$, are loaded via the x input port, prior to the insertion of the first input data item $x(0)$. The loading elongates the data stream $\boldsymbol{X}$ by N and takes N clock cycles for any subsequent change.

To facilitate our derivation, we unroll the signal flow graph of Fig. 2 along time steps into a dependence graph as shown in Fig. 3. The original flow graph can be seen as a projection of the dependence graph along the horizontal direction. The light diagonal lines show the schedule. The dependence graph is augmented with nodes acting to load and store the coefficients; black nodes and shaded nodes to their left. Once the coefficient $h(i)$ turns into the projection (horizontal) direction, it will be stored at the tap. The shaded nodes to the right of the black nodes are needed, since we have to assure $x(i) = 0$ for $i < 0$ in order to apply the recurrence $y(n) = \sum_{i=0}^{n} h(i)x(n-i)$ uniformly. Thus, the output $y(n)$, $0 \le n \le N-2$, passes through taps $h(n+1), \ldots, h(N-1)$ with no change.

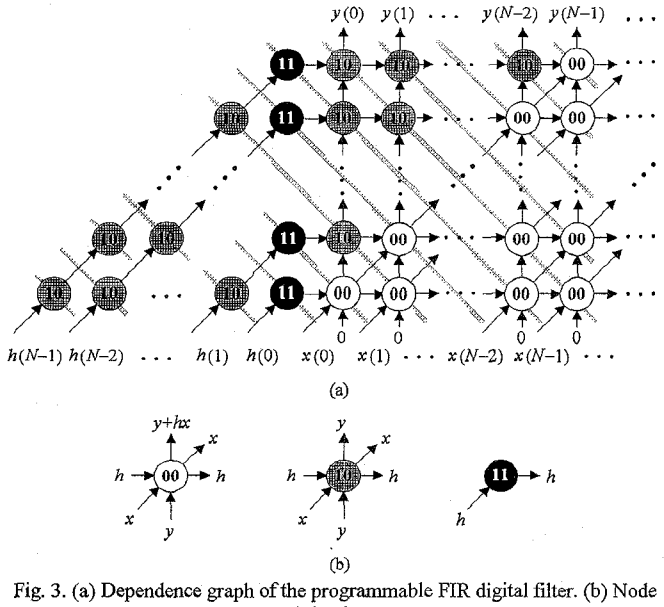

Fig. 3. (a) Dependence graph of the programmable FIR digital filter. (b) Node definitions.

The nodes in Fig. 3 have been labeled with binary x and y tags: 11 for store, 10 for load (or pass), and 00 for multiply-and-add operation. The above tag assignment can be derived as follows. Observe that the nodes for multiply-and-add operations are aligned along the diagonal direction which corresponds to the flow direction of data stream $\boldsymbol{X}$. Hence, we assign an x tag of 0 to distinguish these nodes from the other two node types (i.e., load and store). Similarly, the nodes for store are aligned along the vertical direction which corresponds to the flow direction of data stream $\boldsymbol{Y}$. Hence, we assign a y tag of 1 to distinguish these "store" nodes from the other two node types (i.e., load and multiply-and-add).

The combinations of the x and y tags, when the data streams X and Y meet at the nodes, result in the assignment.

This coding scheme corresponds to simple interpretations for the control signals. An insertion of an x tag of 1 indicates that input data item is a coefficient while a y tag of 1 distinguishes the input data item as the last item in the coefficient sequence.

B. Basic Cell Structure

Based on the above coding scheme, the basic cell for the programmable FIR filter can be obtained, as shown in Fig. 4. The added z data stream allows us to realize a linear-phase FIR filter of order $2N$ or $2N - 1$ by folding it into N taps.

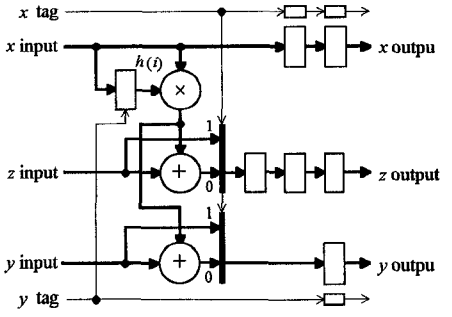

Fig. 4. Basic cell structure with pipelined data and control.

To see this, rewrite the system function in the form of

$$y(n) = \sum_{i=0}^{N-1} h(i)x(n-i) + \sum_{i=0}^{N-1} h(i)x(n+i-2N+1)$$

Because of the symmetry property, we have $h(2N - 1 - i) = h(i)$, where $0 \le i \le N - 1$. Rename the second summation term at the right-hand side as $z(n - 2N + 2i + 1)$, where $z(n) = \sum_{i=0}^{N-1} h(i)x(n - i)$. The computation of $z(n - 2N + 2i + 1)$ is the same as that of $y(n - 2N + 2i + 1)$. The index difference $2N - 2i - 1 = 2(N - i - 1) + 1$ between $z(n - 2N + 2i + 1)$ and $y(n)$ implies a delayed y output with two more pipeline registers inserted between cells and one register added to z output following the last cell.

Fig. 5 shows the first and last cells of the linear-phase FIR filter of order $2N$. In a similar manner, the folded linear-phase FIR filter of order $2N - 1$ can be obtained by removing the register added to z output following the last cell.

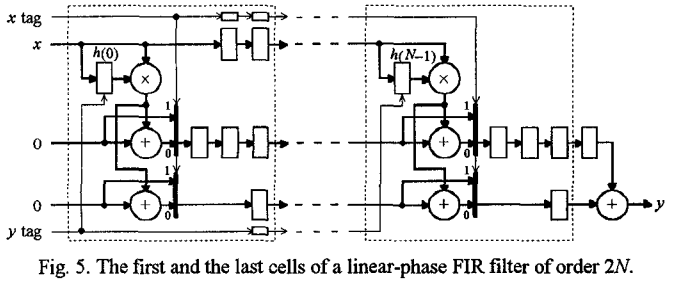

Fig. 5. The first and the last cells of a linear-phase FIR filter of order $2N$.

IV. Cost-Performance Tradeoffs

We compare the various filter designs with respect to cost-performance ratio and also examine their scalability. Only BCBD and PCPD modes are considered, which represent the two extremes with lowest cost and highest performance, respectively. To facilitate our analyses, we assume that data are represented by fixed-point fractional numbers with 16-bit inputs, 16-bit coefficients, and 32-bit outputs. Our concern here is the core array. When implemented on a single chip, the output is often truncated or rounded and extra hardware is required on the periphery.

The coefficients are encoded using a canonic signed-digit (CSD) representation; i.e., in such a way that there are no two consecutive nonzero digits in the representation, so the number of partial products can be reduced. This technique has been used by several FIR filter designs and automatic synthesis tools [4]–[7], [9], [11]. For simplicity, let us select two signed digits to represent each coefficient value

$$h(i) = s_0 2^{-p_0} + s_1 2^{-p_1}$$

The coefficient $h(i)$ has the form (s_0, p_0, s_1, p_1), where $s_0, s_1 \in \{-1, 0, 1\}$, $p_0 \in \{0, 1, \ldots, 13\}$, $p_1 \in \{2, 3, \ldots, 15\}$, requiring $2 \times 2 + 2 \times 4 = 12$ bits for its representation.

Two partial products are generated by shifting the input $x(n - i)$ by p_1 and p_2 bits, respectively, and complementing if required. The partial products are added with the sum and carry from the previous cell by a two-level carry-save adder tree. The true result, however, is not computed until after the last cell to avoid carry propagation delay in each cell.

A. Comparison using 1.0 μm CMOS Technology

We estimate the area and cycle time using a 1.0μm CMOS technology. The layout of the fully pipelined (PCPD) design shows about 20% increase in area for each cell, compared to the broadcast (BCBD) design. Fig. 6 plots the corresponding relative cycle times and area-time products for different values of N. The area-time product, obtained by multiplying the relative area and cycle time, is a commonly used measure of cost-effectiveness.

Fig. 6 shows that for $N \le 128$, the speed improvement resulting form pipelined data and control signals is not enough to offset the area increase. We note that the original cell actually has a shallow circuit depth. By adding control logic to the critical path, our design increases the cycle time when N is small. For existing applications, usually satisfying $N \le 64$, the broadcast design is more cost-effective. In such cases, the propagation delays on long wires do not significantly affect the cycle time. Thus, the broadcast design turns out to be unconditionally better for small N (say, $N \le 16$) and provides a worthwhile speed-for-area tradeoff in the case of larger N (say, $32 \le N \le 64$).

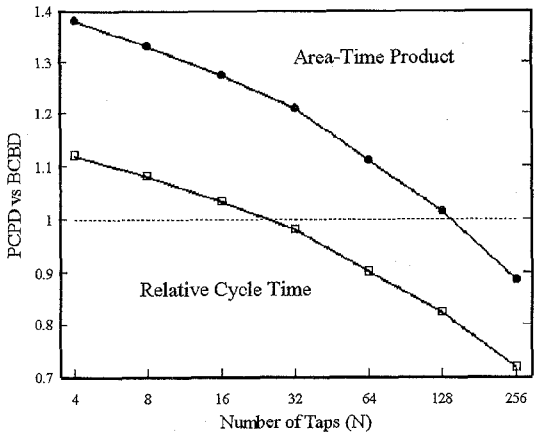

Fig. 6. The relative cycle time and area-time product of the fully pipelined design with respect to the broadcast design, using 1.0μm CMOS technology.

B. Comparison using 0.5 μm CMOS technology

The above results lose their validity with continued downward scaling of feature sizes in integrated circuits to improve performance. As the dimensions are scaled down by a given factor, wire lengths are expected to be reduced by the same factor. However, wire delays are not reduced at the same rate as the device switching times [1].

We thus need to reexamine the relative measures of layout area and cycle time for different values of N, with each change in technology. Fig. 7 shows the results for a 0.5μm CMOS technology. Compared to those of Fig. 6, the relative cycle time now drops more rapidly.

For the broadcast design, the area saved is not enough to offset the speed degradation as N is extended beyond 16. Consequently, its scalability is quite limited. Our fully pipelined design displays its superiority in preserving the speed and density benefits of scaling. We expect that when the technology further advances to the deep submicron regime, the advantages of the fully pipelined design will become apparent for even smaller values of N.

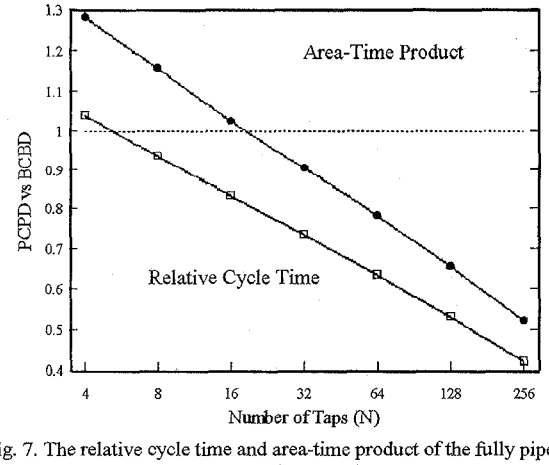

Fig. 7. The relative cycle time and area-time product of the fully pipelined design, with respect to the broadcast design, using 0.5 μm CMOS technology.

V. Conclusion

We have shown that the broadcast design indeed leads to a high cost-performance ratio with a 1.0μm CMOS technology when the filter size is small. This may cease to hold if the fabrication technology is scaled downward or the cascade length is scaled upward. In both cases, propagation delays on long wires become significant.

In view of the significant overhead in broadcasting, we proposed a fully pipelined design. The design thus obtained restricts global connections to clock and power supply. We showed that fully pipelined control is feasible by providing each data item with a one-bit tag. This data-driven scheme needs two control signals, rather than one in the broadcast design, to effect the desired operations. However, in order to deliver the coefficients into the filter, the broadcast design needs a separate input port. The fully pipelined design reuses the data input port for this purpose.

References

[1] S. Bothra, B. Rogers, M. Kellam, and C. M. Osburn, "Analysis of the effects of scaling on interconnect delay in ULSI circuits," *IEEE Trans. Electron Devices*, vol. 40, pp. 591–597, Mar. 1993.

[2] D. J. Evans and M. Gusev, "New linear systolic arrays for digital filters and convolution," *Parallel Computing*, vol. 20, pp. 29–61, Jan. 1994.

[3] M. Hatamian and S. K. Rao, "A 100 MHz 40-tap programmable FIR filter chip," *Proc. Int'l Symp. Circuits and Systems*, May 1990, pp. 3053–3056.

[4] R. A. Hawley *et al.*, "Design techniques for silicon compiler implementations of high-speed FIR digital filters," *IEEE J. Solid-State Circuits*, vol. 31, pp. 656–667, May 1996.

[5] R. Jain, P. T. Yang, and T. Yoshino, "FIRGEN: a computer-aided design system for high performance FIR filter integrated circuits," *IEEE Trans. Signal Processing*, vol. 39, pp. 1655–1668, July 1991.

[6] K.-Y. Khoo, A. Kwentus, and A. N. Willson, "A programmable FIR digital filter using CSD coefficients," *IEEE J. Solid-State Circuits*, vol. 31, pp. 869–864, June 1996.

[7] J. Laskowski and H. Samueli, "A 150-MHz 43-tap half-band FIR digital filter in 1.2-μm CMOS generated by silicon compiler," *Proc. IEEE Custom Integrated Circuits Conf.*, May 1992, pp. 11.4.1–4.

[8] C. E. Leiserson and J. B. Saxe, "Optimizing synchronous systems," *J. VLSI and Computer Systems*, vol. 1, pp. 41–67, 1983.

[9] W. J. Oh and Y. H. Lee, "Implementation of programmable multiplierless FIR filters with powers-of-two coefficients," *IEEE Trans. Circuits and Systems – II: Analog and Digital Signal Processing*, vol. 42, pp. 553–556, Aug. 1995.

[10] L. E. Thon, P. Sutardja, F.-S. Lai, and G. Coleman, "A 240 MHz 8-tap programmable FIR filter for disk-drive read channels," *Dig. IEEE Int'l Solid-State Circuits Conf.*, Feb. 1995. pp. 82–83.

[11] T. Yoshino *et al.*, "A 100-MHz 64-tap FIR digital filter in 0.8-μm BiCMOS gate array," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1494–1501, Dec. 1990.