A Pipelined/Interleaved IIR Digital Filter Architecture*

Zhongnong Jiang[†] **and Alan N. Willson, Jr.**

Electrical Engineering Department, UCLA
Los Angeles, California 90095
[†]Mixed-Signal Products, Texas Instruments Inc.
P.O.Box 660199, MS 8729, Dallas, Texas 75266

ABSTRACT

By using a clock rate that is K times the data rate and with interleaved feedback of the output samples, a single expanded digital filter $H(z^K)$ can be made equivalent to a cascade of k identical filters $H^k(z)$ with $1 \le k \le K$. Whereas this novel pipelining/interleaving (PI) technique can equally be employed for implementing high-performance FIR filters, its main benefit lies in that more efficient high-speed IIR filters become achievable, though their highest possible data rates are still limited by the delays of the critical feedback loops. Hardware architectures and design examples with $K = 2$ are presented to show how the PI technique works for implementing high-speed IIR filters made as the sum of two allpass functions.

I. Introduction

Due to their feedback features, high-speed pipelined computations in IIR digital filters are generally unviable. To overcome this shortcoming, several pipelining transformations, which improve an IIR filters' pipelined computation capabilities, have been investigated [1–3]. However, it is also useful to study a new pipelining/interleaving (PI) technique that leads to more hardware-efficient IIR filter designs rather than simply aiming to achieve higher-speed pipelined IIR filters via the techniques described in [1–3]. Though our novel PI technique is also applicable to FIR or multirate filters, this paper will merely focus on its use in pipelined IIR filters made as the sum of two allpass filters [4]. Interested readers are referred to our more detailed Transactions paper [5].

II. Pipelined/Interleaved Digital Filters

If a digital filter having a transfer function $H(z)$ is expanded by a factor of two, i.e., if each delay element is replaced by a cascade of two delays, the resulting filter's transfer function becomes $H(z^2)$. Suppose now we have two independent signal sequences $x_1(n)$ and $x_2(n)$ to be filtered by $H(z)$ creating two corresponding independent output sequences $y_1(n)$ and $y_2(n)$. An alternative to the use of two separate filters for this purpose is the multirate implementation using $H(z^2)$ shown in Fig. 1. (Notice that the transfer functions actually realized, throughout this paper, may have a few additional cascaded sample delays that are usually inconsequential and therefore are not explicitly specified.) This structure uses a single (pipelined/ interleaved) filter to implement two identical filters. In addition to the double registers needed for implementing $H(z^2)$ the input (output) interleaving (de-interleaving) circuitry shown in Fig. 1, and conveniently implemented with commutators [6], is required. Moreover, the clock rate for this implementation must be double the data rate.

If only one input signal sequence is to be filtered, we can feed the first output sequence $y_1(n)$ back as second input $x_2(n)$. In this way $H(z^2)$ is used for implementing $H^2(z)$ as shown in Fig. 2, where a scaling multiplier R is also inserted to ensure an appropriate dynamic-range constraint. This pipelined/interleaved implementation of a digital filter can easily be extended to using $H(z^K)$ with K being an arbitrary positive integer. Fig. 3 gives such an implementation where the clock rate of $H(z^K)$ must be K times the data rate. To prove the equivalence between the multirate system of

*This research was supported by the National Science Foundation under Grant MIP-9632698, by the Office of Naval Research under Grant N00014-95-1-0231, and by the State of California MICRO Grant 95-160.

Fig. 3(a) and the single-rate system of Fig. 3(b) the transmultiplexer analysis illustrated on p. 263 of [6] will suffice. Clearly, for high-data-rate applications K should be chosen relatively small, otherwise a very high clock rate and many more registers would be required.

For $K = 2$ we have investigated how the FIR and IIR filter design procedures are affected by using this PI technique. In general, it is concluded that the equivalent of a nearly ($3N/2$)th-order FIR filter with b-bit coefficient word length can be implemented by an interleaved Nth-order FIR filter with up to 40% shorter coefficient word length by using our PI technique [5].

Unlike their FIR counterparts, IIR filters are not well suited for pipelined computations to boost processing speed, due to their recursion features. However, the above PI technique can easily be used to make IIR digital filtering more efficient for modest processing rates. More specifically, the processing speed of an IIR filter is determined by its critical loop, which usually includes one or more multipliers and adders that are constrained to process data sequentially in each clock period [1–3]. The PI technique introduces pipelining into the critical loop. If the critical loop is then evenly partitioned, and, if the delay associated with the registers is negligible, higher clock rates (consistent with the $z \to z^K$ transformation) become possible and the filter's overall input-to-output data rate can thus remain unchanged. But, by the multiple use of the same IIR filter, as will be shown shortly, we will have achieved more efficient IIR digital filtering than the conventional implementation provides. (This idea was first proposed for IIR digital filtering in a special case [7] where a second-order allpass function is used for implementing two distinct second-order sections with adders being shared, while the number of multipliers remains the same.) Similar to the FIR cases, the analyses in [5] have shown that Nth-order IIR filters using the PI technique can be made equivalent to approximately ($3N/2$)th-order IIR filters. In fact, some efficient IIR filter structures with low passband sensitivity, such as IIR filters made as the sum of two allpass functions [4], are perfect candidates for using the PI technique since these filters usually suffer from high stopband sensitivity. Next, hardware architectures of the IIR filters adopting our PI technique are described in detail.

III. Hardware Architectures

For a lowpass IIR filter, suppose the passband ripple specification is in the range 0.05 to 0.25 dB and a stopband loss greater than 100 dB is desired. Suppose the normalized passband edge frequency is 0.2 and the stopband edge frequency is varied as 0.25, 0.225 and 0.2125. Prototype IIR filters F1, F2 and F3, with 7th-, 9th- and 11th-orders, respectively, exceed the requirements of 0.05 dB passband ripple and 50 dB stopband loss with infinite-precision coefficients [5]. It is known that, under certain conditions, many odd-order IIR filters can be implemented as the sum of two allpass functions [4]. Because the stopband performance of an IIR filter made as the sum of two allpass functions is very susceptible to deterioration due to coefficient quantization, we conduct a simple exhaustive search for better quantized binary coefficients. In our optimization process, no constraints are imposed on the passband's quality as we are aware that this IIR filtering structure always has low coefficient-value sensitivity on passband performance; rather, we search for optimal binary coefficients that yield stopband rejection greater than 50 dB, and require only 7- and 8-bit word length. Passband performance is slightly degraded (typically less than 0.08 dB in comparison to IIR filters that use the normal quantization procedure). The optimization procedure and optimized coefficients are presented in [5].

Now that an Nth-order IIR filter can often be made equivalent to a ($3N/2$)th-order filter when the PI technique is used, an efficient hardware design for a programmable expanded 7th-order (equivalent to 10th-order) IIR digital filter $H(z)$, decomposed as the sum of two allpass subfilters, in which one branch has an expanded first-order allpass plus an expanded second-order allpass function while the other has two expanded second-order allpass sections, is proposed in Fig. 4. When our PI technique (with $K = 2$) is invoked, we actually need to implement $H(z^2)$ rather than $H(z)$, plus the two commutators manipulating the input, the fedback output/input and the actual output sequences, as illustrated by the structure of Fig. 2. As shown in Fig. 5 for an expanded $K = 2$ first-order allpass section $\frac{z^{-2} + \alpha}{1 + \alpha z^{-2}}$ we can place two delay elements (registers) in such locations that the register-to-register circuit delays are balanced as closely as possible. Furthermore, we suggest that carry-save arithmetic be used as much as possible in the hardware implementation in order to reduce the total delay within the critical loop. The delay associated with the carry-save adder array of the $-\alpha$ multiplier plus the subsequent carry-save adder should be nearly the same as the delay relating to a carry-save adder plus the carry-lookahead adder. (Without using the carry-lookahead adder we would have to build two multipliers rather than one, although this might provide a speed advantage.) The actual output of this first-order allpass section can be evaluated by using a pipelined adder that has been intentionally moved out of the feedback loop (see Fig. 6(a)) for further reduction of the total delay within the critical loop. Following a similar strategy, we can readily construct a carry-save style architecture for an expanded second-order allpass section $\frac{\beta + \alpha z^{-2} + z^{-4}}{1 + \alpha z^{-2} + \beta z^{-4}}$ as shown in Fig. 6(b). Since the Fig. 6 structures actually implement $z^{-1}H_1(z^2)$ and $z^{-4}H_2(z^2)$ for the expanded first- and second-order allpass functions, respectively, it becomes necessary to cascade three more z^{-1} delays with the upper path of the Fig. 4 structure. Based on the analysis given in [5], 7- to 10-bit binary coefficients can be considered so that only four to five partial products need be created when the well-known modified Booth multipliers are used.

In summary, our effectively 10th-order IIR filter would now require only seven 10 × 16-bit multipliers (and 32-bit accumulators) for processing 16-bit data. The highest achievable clock rate should be approximately 100 MHz using current CMOS technology. Moreover, our effectively 10th-order IIR filter might well be equivalent to a 100-tap FIR filter in very narrow-band cases. The drawback is that, while the data rate of the FIR filter is the same as the clock rate, the data rate of our pipelined/interleaved IIR filter is half the clock rate.

While nonlinear stability issues have not been discussed in this paper, since IIR filters using our PI technique impose no constraints in adopting any IIR filter structures, existing analysis methods for IIR filters can be used for analyzing noise gains and limit cycles characteristics. Of course, the fedback input sequence, which has been processed in the first IIR filtering procedure, must be properly scaled such that overflow and dynamic range concerns for the second pass through the filter are treated correctly (see Fig. 2).

IV. Conclusion

We have presented a novel pipelined/interleaved IIR digital filter structure that is more efficient than a conventional filter implementation and retains high performance capabilities.

References

[1] H. H. Loomis and B. Sinha, "High speed recursive digital filter realization," *Circuits, Systems, and Signal Processing*, vol. 3, pp. 267-294, 1984.

[2] K. K. Parhi and D. G. Messerschmitt, "Pipeline interleaving and parallelism in recursive digital filters – Parts I and II," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 1099-1135, July 1989.

[3] Zh. Jiang and A. N. Willson, Jr., "Design and implementation of efficient pipelined IIR digital filters," *IEEE Trans. Signal Processing*, vol. 43, pp. 579-590, March 1995.

[4] A. N. Willson, Jr. and H. J. Orchard, "Insights into digital filters made as the sum of two allpass functions," *IEEE Trans. Circuits Syst.*, vol. 42, pp. 129-137, March 1995.

[5] Zh. Jiang and A. N. Willson, Jr., "Efficient digital filtering architectures using pipelining/interleaving," *IEEE Trans. Circuits Syst.*, 1996.

[6] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Englewood Cliffs, NJ: Prentice Hall, 1992.

[7] L. Liu *et al.*, "An interleaved/retimed architecture for the lattice wave digital filter," *IEEE Trans. Circuits Syst.*, vol. CAS-38, pp. 344-347, March 1991.

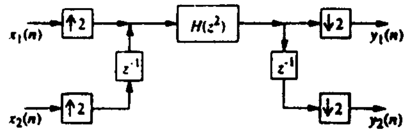

Fig. 1. Digital filtering of two independent signal sequences using a single filter.

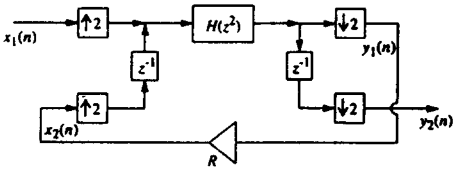

Fig. 2. A cascade of two identical filters realized by using a single filter, and including a scaling multiplier R.

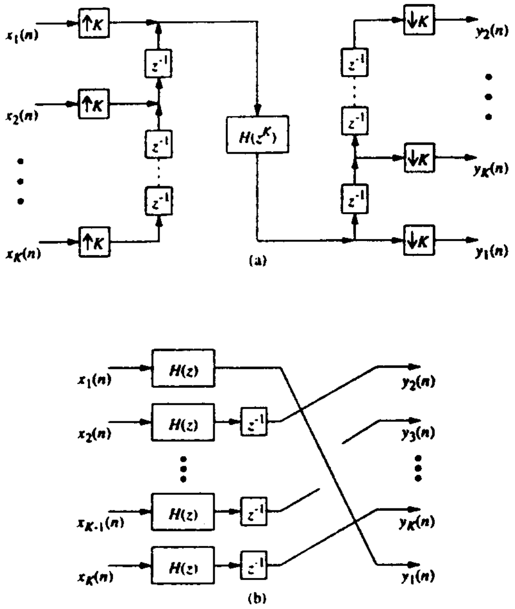

Fig. 3. Digital filtering of K independent signal sequences using a single filter. (a) Transmultiplexer-type model. (b) Equivalent single-rate structure.

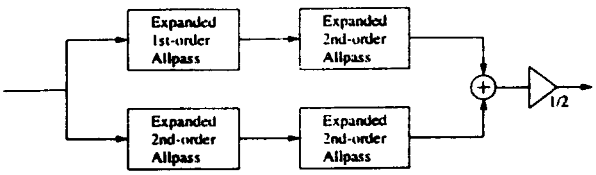
Expanded 1st-order Allpass
Expanded 2nd-order Allpass
Expanded 2nd-order Allpass
Expanded 2nd-order Allpass
1/2

Fig 4. An expanded seventh-order IIR filter.

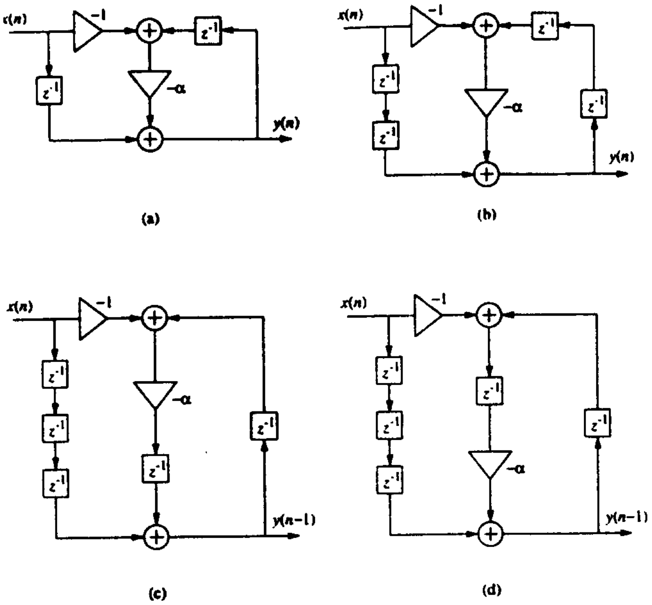

Fig. 5. First-order allpass function: (a) original structure, (b) expanded structure, (c) balancing critical loop computations, (d) final structure.

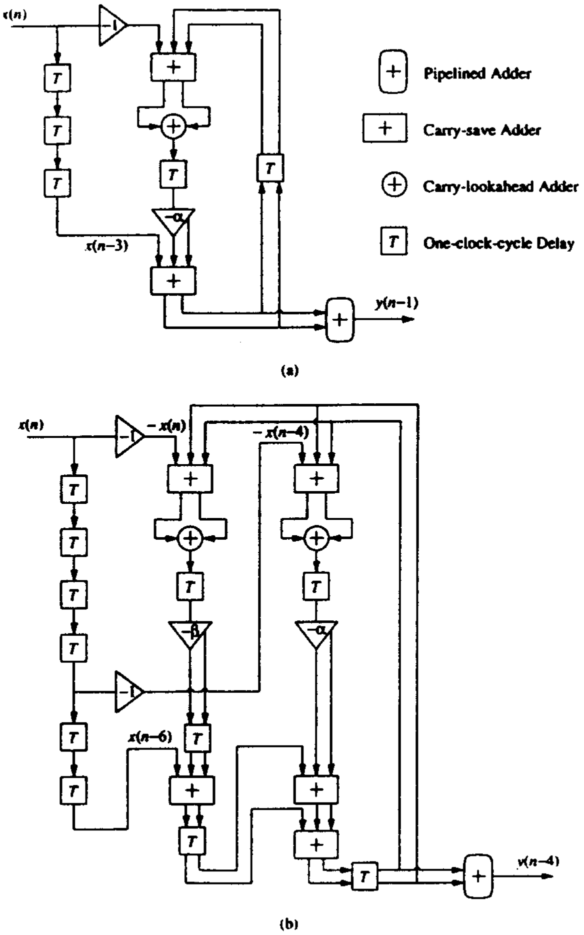

Fig. 6. Hardware structures for expanded allpass functions: (a) 1st-order allpass, (b) 2nd-order allpass.