A Low-Power, Area-Efficient Digital Filter for Decimation and Interpolation

Brian P. Brandt, *Member, IEEE,* and Bruce A. Wooley, *Fellow, IEEE*

***Abstract*— The area and power consumption of oversampled data converters are governed largely by the associated digital decimation and interpolation filters. This paper presents a low-power, area-efficient, mask-programmable digital filter for decimation and interpolation in digital-audio applications. Several architectural and implementation features reduce the complexity of the filter and allow its realization in a die area of only 3670 mils^2 (2.37 mm^2) in a 1-μm CMOS technology. The use of simple multiplier-free arithmetic logic and a new memory addressing scheme for multi rate digital filters results in a power consumption of only 18.8 mW from a 5-V supply and 6.5 mW from a 3-V supply. The memory addressing scheme and the programmable functionality of the filter are general enough to implement a wide class of FIR and IIR single-rate and multi-rate digital filters.**

I. Introduction

While the conversion rate and resolution of oversampled data converters are typically determined by their analog components, their power consumption and die area are governed largely by the digital decimation and interpolation filters. Power and die area are increasingly important considerations as oversampled converters continue to find applications in portable battery-operated equipment and penetrate low-cost consumer markets. Furthermore, as the range of applications increases, versatility and programmability become important features of decimation and interpolation filters.

Two principal design goals were adopted for the digital filter described in this paper. First, the filter was designed to make efficient use of both power and silicon die area. The filter leverages several architectural and implementation features to achieve low computational complexity and to minimize the number of data transfers. At the circuit level, power and area were conserved through the use of simple circuits that were designed to operate at low supply voltages. The second design goal was that the filter be versatile. For example, it should accommodate a variety of sigma-delta ($\Sigma\Delta$) modulator architectures, allow the sampling rate change factor to be modified easily, and be able to perform either decimation or interpolation.

Unlike previously reported efficient decimation and interpolation filters, the filter described herein is not hard-wired to perform a specific function. In addition to programmable coefficient values, the entire structure of the filter, such as the number and type of stages (FIR or IIR), is programmable. Moreover, the filter does not require a 1-bit input to perform decimation efficiently. While single-stage decimation filters based on 1-bit inputs [1] can lead to area-efficient implementations, their size and power consumption are proportional to the decimation ratio, M. In contrast, the size and power consumption of the filter reported herein are relatively insensitive to the $\Sigma\Delta$ modulator architecture and the decimation ratio.

Manuscript received August 19, 1993; revised February 1, 1994. This research was partially supported by the Semiconductor Research Corporation under Contract 91-DJ-112.

B. P. Brandt was with the Semiconductor Process and Design Center, Texas Instruments, Inc., Dallas, TX 75243. He is now with the Networking Hardware Division of IBM Inc., Research Triangle Park, NC 27709.

B. A. Wooley is with the Center for Integrated Systems, Stanford University, Stanford, CA 94305-4070.

IEEE Log Number 9402118.

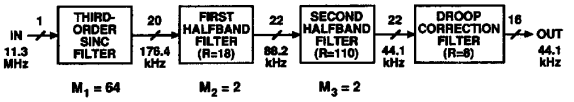


Fig. 1. Four-stage linear-phase decimation filter.

The filter is first described in the context of decimation; in particular, decimation following a second-order $\Sigma\Delta$ modulator. Central to the operation of the filter is a memory addressing scheme for multirate digital filters that is introduced in Section III. Section IV describes the hardware used to implement the filter and the simple modifications required to accommodate alternative $\Sigma\Delta$ modulators. The use of the filter to perform interpolation is discussed in Section V. The design of an experimental prototype of the filter is described in Section VI, and measurement results characterizing its performance are presented in Section VII.

II. Decimation Filter Architecture

While the programmable functionality of the filter allows it to implement a variety of single-rate and multirate filtering operations, it will be introduced by focusing on its application as a linear-phase decimation filter for digital-audio systems. The filter, shown in Fig. 1, has been designed for use in an oversampled A/D converter employing a second-order $\Sigma\Delta$ modulator [2].

The architecture of the filter is designed to minimize arithmetic and hardware complexity. By implementing the filter as a cascade of several linear-phase stages, the ratio of the input rate to the width of the transition band is dramatically reduced for each of the individual stages [3]. This results in a significant reduction in overall hardware complexity as compared to a general single-stage design. The third-order sinc filter [4]

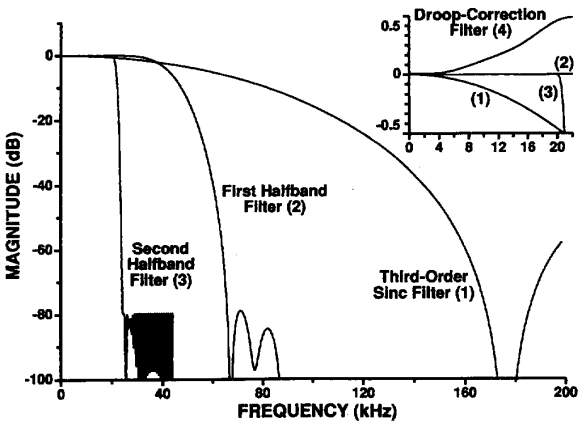


Fig. 2. Magnitude responses of the four stages of the decimation filter.

reduces the sampling rate by a factor of 64. The remaining sampling rate reduction to the Nyquist output rate of 44.1 kHz is accomplished efficiently with two cascaded halfband filters [3]. The nearly flat pass band magnitude response of the halfband filters precludes their use in compensating for the droop in the baseband response of the third-order sinc filter. Thus, a low-order ($R = 8$) droop-correction filter, operating at the 44.1-kHz output rate, is used as a fourth stage.

The magnitude responses of the four filter stages are shown in Fig. 2 . The requirements on the magnitude response of the second halfband filter are the most stringent, as reflected in its considerably increased order ($R = 110$) compared to the other stages. The 4.1-kHz transition band for this filter is less than 5% of the its input rate of 88.1 kHz.

In the two halfband filters all but one of the odd coefficients are zero, thereby reducing their computational complexity by nearly 50% as compared to a general direct-form filter architecture. This reduction, coupled with their symmetric impulse responses, allows the first and second halfband filters to be specified by only 6 and 29 non-zero coefficients, respectively. The requirements of the first halfband filter are satisfied efficiently by one member of a class of halfband filters proposed by Goodman and Carey [5]. The *F9* halfband filter described in [5] was chosen because of the low number of non-zero bits in its six coefficients. The 26 coefficients for the second halfband filter were determined using a combination of the Remez exchange algorithm [6] and a technique described in [7].

The third-order sinc filter depicted in Fig. 3 is implemented as a cascade of three integrators, which operate at 11.3 MHz, followed by a resampling operation and a cascade of three differentiators operating at 176.4 kHz [8], [9]. Such an implementation avoids the need for multipliers or coefficient storage and features both minimal data storage and straightforward decimation ratio programmability. Moreover, the amount of hardware operating at the high sampling rate is kept to a minimum, thereby conserving power.

The first halfband filter is implemented using the *polyphase* structure [3] shown in Fig. 4. The alternating switch, or

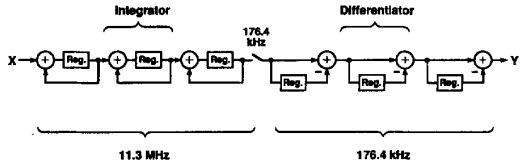


Fig. 3. Third-order sinc filter implementation.

commutator, at the input of the filter directs even inputs to one branch and odd inputs to the other. One output value, y, is formed for each even-odd pair of inputs, x, in order to reduce the sampling rate by a factor of two. Since the resampling operation precedes the arithmetic operations and the delay elements, the components of the filter operate at the output rate rather than the higher input rate.

The tapped-delay lines in the even and odd branches of Fig. 4 are doubled back to exploit the symmetric impulse response of the linear-phase filter and reduce the number of multiplications by a factor of two [10]. Halfband filters have the distinguishing characteristic that every odd coefficient is zero except the center one, C_9, which has a value of 0.5. Thus, the elements shown as broken lines in Fig. 4 may be removed, leaving a very simple odd branch consisting of only four delay terms and a single coefficient multiplication, which may be implemented with a one-bit right shift.

The second halfband filter is also implemented using the polyphase structure of Fig. 4, although more coefficients and delay terms are required. The droop-correction filter does not perform a sampling rate reduction and has non-zero odd coefficients. This stage is implemented using a single tapped-delay line that is doubled back to exploit its symmetric impulse response.

III. A Memory Addressing Scheme for Multirate Digital Filters

In hardware implementations of digital filters, the z^{-1}terms seen in Fig. 4 correspond to data storage elements. Often these storage elements are implemented as registers. In FIR filters, the registers form one or more tapped delay lines in which data is transferred from one register to the next as the filter outputs are computed. To compute each output, the input data samples stored in the registers are circularly shifted to a common point where they are multiplied by the appropriate coefficients. Note that in the formation of each output of the filter, each input sample is shifted N times, where N is the number of taps in the filter. Thus, there are a total of N^2 data transfers for each output of the filter.

The large number of data transfers required in filters based on recirculating shift registers results from the fact that the data is accessed serially. Consequently, all of the data must be shifted for each access. Alternatively, the filter input data may be accessed *randomly* in a memory. In that case, only one input sample is transferred for each data access required in the computation of the filter output. Instead of moving the data from one location to the next in order to implement a tapped delay line, the data can remain in the same physical locations.

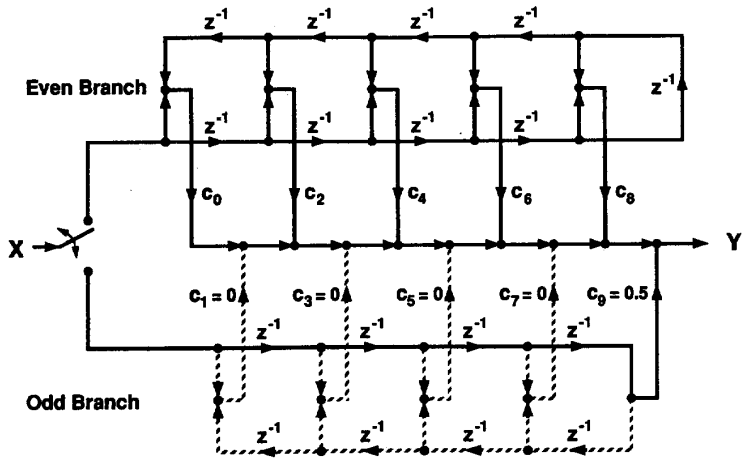


Fig. 4. Polyphase implementation of the first halfband filter.

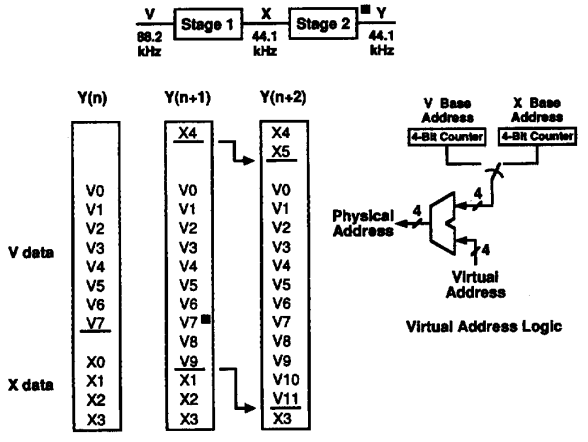


Fig. 5. Illustration of the consequences of unequal data rates in a two-stage filter.

Only the effective, or virtual, address of the data is changed so that the data appears to be in the next location in the delay line. The physical address of the RAM is the sum of the virtual address and a base address, and the base address is incremented once to implement all delays in the filter simultaneously. When the sum exceeds the physical address space of the memory, the physical address will wrap around to the top of the memory if the carry out of the most significant bit of the adder is ignored. Also, if the counter that generates the base address is designed to return to zero when it is incremented past the last physical address, the memory will be accessed in a circular, or modulo, fashion.

A problem arises when the virtual addressing scheme is applied directly to a multistage filter in which stages of differing input data rates share a common RAM. Fig. 5 depicts an example of a two-stage filter in which the first stage has an input rate of 88.2 kHz while the second stage has an input rate of 44.1 kHz. The contents of a 16-word memory are shown for three consecutive output sample periods. The inputs to the first and second stages are labeled Vj and Xj, respectively, where j enumerates consecutive inputs so that smaller j-values denote older inputs. The virtual address logic uses two 4-bit base address counters and a 4-bit adder to address the 16-word memory. The V counter is selected when V data is accessed and the X counter is selected when X data is accessed. The V counter is incremented once for each input to the first stage and counts at twice the rate of the X counter, which is incremented once for each input to the second stage. One new X data sample is added to the RAM for every output sample, Y, of the filter. However, two new V data samples are added in the same time period. Thus, because of the differing input data rates of the two stages, the V data wave front progresses at twice the rate of the X wavefront, as indicated by the arrows in Fig. 5. By the time output sample $Y(n+5)$ is computed, the V data will have overwritten all of the X data.

The memory addressing scheme for multi-rate, multistage filters illustrated in Fig. 6 provides a solution to the problem created by the differing input data rates of the two stages. Instead of storing the V data as a single sequential block as in Fig. 5, it is separated into even and odd blocks, as indicated in Fig. 6. In this simple scheme, each of the two V blocks adds only one new entry per output of the two-stage filter, the same rate at which the X data is added. As the arrows in Fig. 6 indicate, the V and X wavefronts now progress through the memory at identical rates.

An important difference between the addressing schemes illustrated in Figs. 5 and 6 is that in Fig. 6, only one of the two new V data samples replaces an old X sample. The second new V data sample replaces an old V sample, instead. A second important difference is that in the scheme depicted in Fig. 6, all of the data for both stages may be indexed to a single base address counter, which is incremented once for each output of the two-stage filter. That is, every delay in the entire multistage filter is implemented without even a single data transfer simply by incrementing one counter.

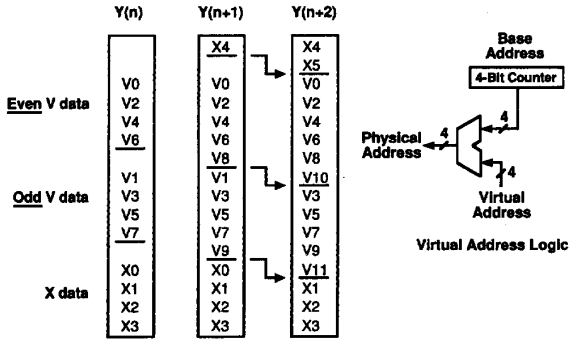


Fig. 6. Memory addressing scheme for multirate, multistage filters.

The memory addressing scheme can be extended easily to accommodate any number of stages of differing input rates and sampling rate change factors. Furthermore, since the scheme requires only one adder and one base address counter, no hardware alterations are required to change the number of stages. The only requirements are that the total data storage required by all the stages not exceed the size of the memory and that the input data for a particular stage be divided into a number of blocks equal to the ratio of that stage's input rate to the output rate of the overall filter. For example, if the input rate for stage 1 in Fig. 5 was 176.4 kHz instead of 88.2 kHz, its input data would be separated into four blocks. Alternatively, if there was an additional stage preceding stage 1 with an input rate of 176.4 kHz and an output rate of 88.2 kHz, its input data would also be divided into four blocks.

The size of the partitioned data blocks varies according to the number of taps in the individual filter stages. Also, the size of the blocks may vary within a given stage. For example, when the first and second halfband filters in Fig. 1 are implemented with the polyphase structure of Fig. 4, the odd branches require only about half the number of storage elements required for the even branches. Once the number and size of the blocks are determined, they may be arranged and abutted so that data that is no longer needed is quickly replaced by new data and the RAM is used at its optimum storage efficiency.

Finally, note that the use of the addressing scheme is not limited to filter stages that reduce the sampling rate of their input. The scheme is equally applicable to interpolation filters such as the one to be described in Section V. For multistage filters that increase the sampling rate, the input data for a particular stage is divided into a number of blocks equal to the ratio of that stage's input rate to the *input* rate of the overall filter. Also, the base address counter is incremented once for each *input* sample rather than once for each output sample of the overall filter.

IV. Implementation

The functional similarities among the last three stages of the filter in Fig. 1 permit their implementation using the processor shown as part of Fig. 7. The processor employs the memory addressing scheme introduced in the previous section

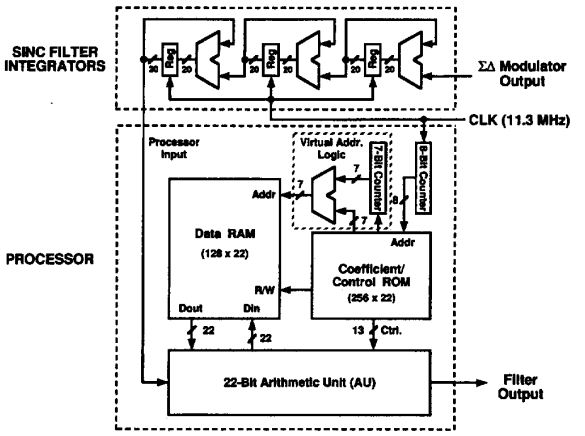


Fig. 7. Block diagram of the decimation/interpolation filter.

to efficiently store the state data for all three stages and to minimize the number of data transfers. The virtual address logic is 7 bits wide in order to address the 128-word data RAM. The coefficient/control ROM stores the 7-bit virtual addresses and also increments the 7-bit base address counter once for each output of the four-stage filter. The 256 locations in the ROM are accessed sequentially via an 8-bit counter driven by the 11.3-MHz filter clock. Each of the 256 ROM locations is accessed once for each output sample of the four-stage filter.

The arithmetic unit (AU) of the processor in Fig. 7 performs the basic arithmetic operations common to the last three stages of the filter in Fig. 1; namely, the addition of pairs of delayed filter inputs stored in the data RAM, the multiplication by coefficients stored in the coefficient/control ROM, and the accumulation of products to form the output of each stage. Moreover, because the three differentiators of the sinc filter shown in Fig. 3 operate at the relatively low data rate of 176.4 kHz, they can also be implemented by this same arithmetic unit. Only the three sinc filter integrators, which operate at 11.3 MHz, require dedicated adders and registers, as indicated in Fig. 7. Register overflows in the sinc integrators are inconsequential if the register word lengths are large enough to accommodate the largest possible output of the sinc filter [8]. In this design a word length of 20 bits in the registers and adders is sufficient.

As a direct result of using a multistage architecture and halfband filters, a total of only 43 multiplications and 84 additions are performed for each output of the four-stage decimation filter. This corresponds to 1.9 million fixed-point multiplications per second or one multiplication every 6 clock cycles. At this relatively low multiplication rate, a significant reduction in hardware complexity is achieved by performing multiplications over several clock cycles as a series of shifts and adds. The average number of shifts and adds per multiplication has been reduced to less than 3.5 by using a computer program that rounds precise coefficients to the nearest canonic signed digit (CSD) coefficients [11]. This reduction allows a single ripple-carry adder within the AU to perform both the

43 multiplications and the 84 additions required for each filter output sample.

The reduction of noise resulting from the rounding of filter *data* has also been considered in the design of the arithmetic unit. For example, coefficient multiplications are performed in the order of increasing coefficient magnitude to compute the output of a stage. This ordering allows smaller products to be accumulated before being added to larger products.

An additional means of reducing data roundoff noise is Horner's multiplication scheme, or nested multiplication. In nested multiplication, partial products, rather than the multiplicand, are shifted and added. For example, a normal multiplication of the multiplicand x by the CSD coefficient 71/256 is performed as:

$$\frac{71}{256}x = \frac{1}{4}x + \frac{1}{32}x - \frac{1}{256}x.$$

The corresponding nested multiplication is performed as:

$$\frac{71}{256}x = \frac{1}{4}\left(x + \frac{1}{8}\left(x - \frac{1}{8}x\right)\right).$$

In this example, the maximum right shift required in the nested multiplication is 3 bits, as compared to 8 bits for the normal multiplication. Also, the average right shift is reduced from 5 to 2.67 bits by nesting.

The use of nested multiplication can have a significant impact on the hardware used to perform the shifts and adds. For example, using normal multiplication, the maximum and average right shifts for the coefficients of the first halfband filter are 14 and 8.75, respectively. The use of nested multiplication reduces the maximum and average shifts to 6 and 2.4, respectively. Thus, the shifter and adder must be approximately 6–8 bits wider for performing normal multiplication than for performing nested multiplication in order to limit data roundoff noise to the same level.

The basic operation in nested multiplication is that of finding the sum or difference of the multiplicand (x) and a shifted partial product. Fig. 8 illustrates the configuration of the small number of simple logic elements in the AU that are used to perform that basic operation, as well as all other arithmetic operations required by the filter. This configuration allows the shifted contents of register B or register C to be added to or subtracted from the contents of register A or zero. The shifter can perform 0 to 7 bit right shifts on its input in order to implement power-of-two multiplications by 1 to 1/128. The rounding logic ensures that the output of the adder is rounded to the nearest least significant bit (lsb). This logic determines the appropriate value of the carry into the adder, Cin, based on the value of the bits that are shifted to the right of the adder's lsb. To avoid biased rounding [12], the output is rounded to the nearest even result in the case where the precise result is exactly halfway between two possible rounded outputs.

While the four-stage filter shown in Fig. 1 was designed to meet the stringent performance requirements of digital-audio signal acquisition, the mask-programmable functionality of the filter allows its frequency response to be easily tailored. Table I illustrates the influence of the frequency response specifications on the number of clock cycles and ROM locations

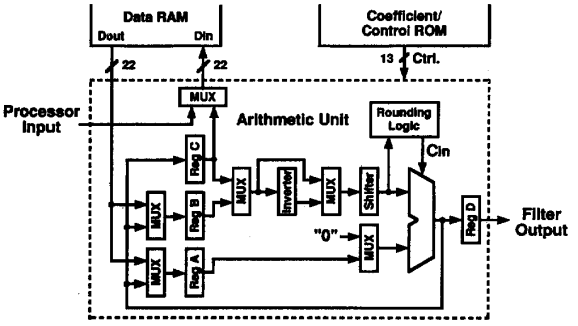


Fig. 8. Arithmetic unit of the processor.

TABLE I

Filter	Transition Band (kHz)	Stopband Attenuation (dB)	Passband Ripple (dB)	Clock Cycles
Four-Stage Decimation Filter	20–24.1	80	0.01	225
Four-Stage Decimation Filter (48 kHz output rate)	20–28	80	0.01	173
Four-Stage Decimation Filter	20–24.1	60	0.01	188
Four-Stage Decimation Filter	20–24.1	40	0.1	136
Five-Stage Interpolation Filter	20–24.1	80	0.01	255

required for the processor shown in Fig. 7. The first table entry indicates that the four-stage decimation filter requires 225 clock cycles to compute each output sample and thus uses 225 of the 256 available ROM locations. If the output rate of the filter is increased from 44.1 kHz to 48 kHz, the lower edge of the stopband may be increased from 24.1 kHz to 28 kHz while preventing a liasing into the 20-kHz passband. The resulting increase in the transition bandwidth reduces the number of required clock cycles and ROM locations to 173, as indicated in the second entry of Table I. Alternatively, if the stopband attenuation is reduced from 80 dB to 60 dB or 40 dB, the number of clock cycles may be reduced from 225 to 188 or 136, respectively.

The hardware implementation of the filter can be extended easily to allow its use at decimation ratios other than 256 and with a variety of oversampling modulator architectures. Changes in the decimation ratio can be implemented by adjusting the amount of decimation performed by the sinc filter while the two halfband stages continue to perform the final factor of four in sampling rate reduction. The sinc filter decimation ratio is adjusted by changing the sampling rate of the switch between the integrators and differentiators in Fig.

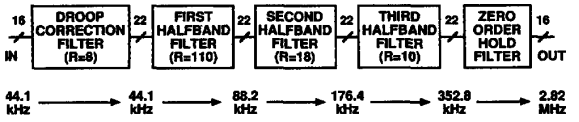


Fig. 9. Five-stage linear-phase interpolation filter for 64-times oversampling.

3. For decimation ratios greater than 256, the word length of the sinc filter integrators must be increased somewhat [8]. An increase in the order of modulator noise shaping can be accommodated through the use of additional sinc filter integrators and differentiators. The differentiators are simply programmed into the processor.

The multibit output produced by cascaded and multibit single-stage $\Sigma\Delta$ modulators can be accommodated by means of a small increase in the word length of the sinc filter integrators. In contrast, decimation filters that depend on a 1 bit input to reduce multiplications to additions cannot be used with cascaded or multibit modulators. Furthermore, the size and power consumption of such filters is approximately proportional to the decimation ratio because they must be implemented as a single stage. The size and power consumption of the filter in Fig. 7 is relatively insensitive to the choice of modulator architecture and decimation ratio because the sinc integrators occupy less than 10% of the total filter area and consume only 16% of the total power.

V. Interpolation

The memory addressing scheme and the programmable functionality of the filter depicted in Fig. 7 are general enough to implement a variety of digital filters. For example, the filter can be mask-programmed to implement the five-stage, 64-times interpolation filter for digital-audio applications shown in Fig. 9. A cascade of three stages increases the sampling rate by 8 times. The use of three halfband filters for this purpose provides increased efficiency as compared to a similar design employing one halfband filter [13]. A zero-order-hold stage provides the remaining factor of 8 increase in sampling rate. The low-order first stage compensates for the baseband droop of the zero-order-hold as well as that of the subsequent D/A conversion. Note that the interpolation filter in Fig. 9 does not require a sinc filter and, as a result, its die area and power consumption are somewhat lower than those of the decimation filter shown in Fig. 1. The memory addressing scheme described in Section III is again used to efficiently store filter data and to reduce the number of data transfers and power consumption.

The magnitude responses of the last four stages of the interpolation filter are shown in Fig. 10. The combination of the last four stages provides more than 80 dB of image rejection except for images centered at multiples of 352.8 kHz. The images at multiples of 352.8 kHz are attenuated by at least 24 dB by the zero-order-hold stage and are further attenuated by the analog reconstruction filter that follows the subsequent D/A conversion.The last entry in Table I indicates that the five-stage interpolation filter uses 255 ROM locations and requires 255 clock cycles for each input sample.

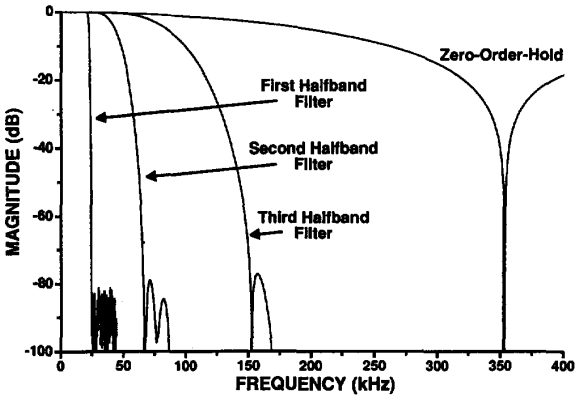


Fig. 10. Magnitude responses of the last four stages of the interpolation filter.

VI. Circuit Design

The circuits used to implement the filter were designed to minimize complexity in order to reduce power consumption and die area. For example, ripple-carry adders [14] were used in the arithmetic unit and the sinc filter integrators. The multiplexers and the shifter in the arithmetic unit were implemented with simple pass transistor logic.

The circuits were also designed to allow operation from a 3-V supply. Static CMOS logic was used throughout the filter except for precharging the bitlines in the RAM and ROM. The bitlines are precharged to V_{DD} and then selectively discharged completely to ground according to the contents of the chosen word in the RAM or ROM. The complements of RAM bitlines that are discharged are actively restored to V_{DD} by cross-coupled PMOS pull-up transistors. The combination of precharging and full bitline voltage swings eliminates static pullup currents and allows single-ended sensing with an inverter.

The registers used in the arithmetic unit and in the sinc filter integrators were implemented using the two-phase register cell depicted in Fig. 11. This cell employs NMOS-only pass gates to avoid the need for complementary clocks. The restoring feedback around the inverters, which employs weak PMOS transistors, facilitates operation of the register cell at low supply voltages. It also allows data to be retained in the register indefinitely in case the register is loaded infrequently.

To allow rapid production of application-specific versions of the filter, the coefficient/control ROM is programmed late in the fabrication process at the "via" mask level. The data RAM utilizes a 6T CMOS static memory cell. The RAM can potentially be implemented using dynamic memory cells to reduce its area. Since every memory location is read, and therefore can be refreshed, at least once for each filter output, a dynamic RAM cell would require a data retention time of only 23 μs. Moreover, if the data can be sensed nondestructively, as in a 3T DRAM cell, refresh may be unnecessary altogether

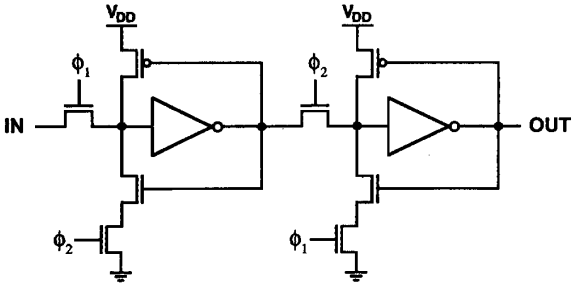


Fig. 11. Two-phase register cell with internal feedback.

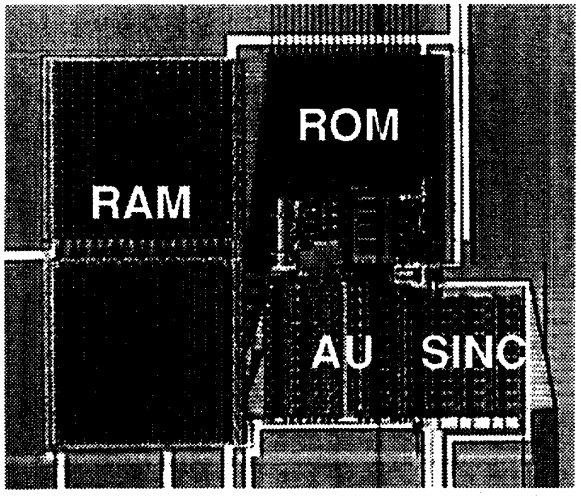


Fig. 12. Die photograph of the experimental digital filter.

since the longest data retention required in the filters depicted in Figs. 1 and 9 is 1.25 ms.

VII. Experimental Results

The digital filter depicted in Fig. 7 has been fabricated in a 1 μm CMOS technology. Fig. 12 is a die photograph of the filter, which has an active area of 3,670 mil^2 (2.37 mm^2). The data RAM occupies 47% of the die area, while the ROM, AU, and sinc filter integrators occupy 17%, 14%, and 8%, respectively.

The filter has been tested and found to be fully functional. Fig. 13 shows the measured baseband spectra before and after rounding the decimation filter output to 16 bits. The data roundoff noise floor maintained internally in the filter is -113.7 dB, a level that is well below the -98 dB noise floor of the 16-bit output. Simulations indicate that a word length of 20 bits in the arithmetic unit and data RAM is sufficient to limit data roundoff noise introduced into the 16-bit filter output to less than 0.2 dB. Two additional bits, for a total of 22, are required to prevent overflow during the nested multiplication operations.

The measured power consumption of the filter as a function of the supply voltage is shown in Fig. 14. When operating from a 5-V supply, the power consumption is 18.8 mW. Note that the filter can operate at a clock rate of 11.3 MHz for

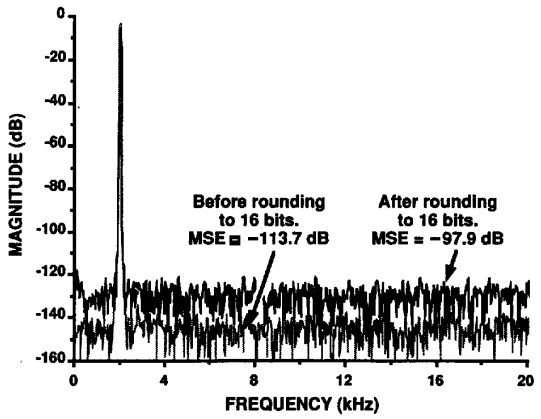


Fig. 13. Measured baseband output spectra (2048-point FFT).

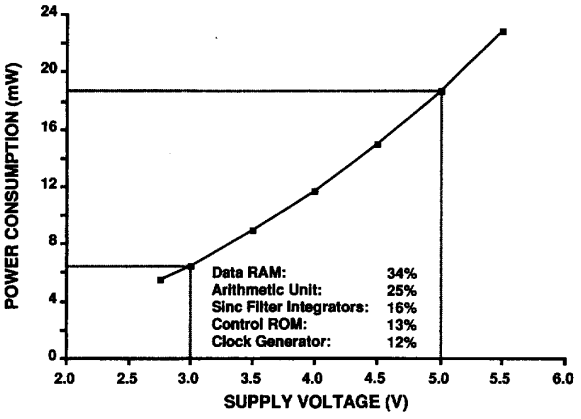


Fig. 14. Measured power consumption for a 11.3-MHz clockrate.

supply voltages as low as 2.7 V, making it suitable for 3 V applications. Furthermore, the power consumption falls to 6.5 mW for a 3-V supply.

The maximum clock rate of the filter is plotted in Fig. 15 as a function of the supply voltage. The second ordinate axis indicates the power consumption at the maximum clock rate. The filter achieves a clock rate of 35.7 MHz while consuming 59.7 mW from a 5-V supply. The critical path is in the arithmetic unit depicted in Fig. 8 and includes three multiplexers, an inverter, a shifter, and a 22 bit ripple-carry addition. Performance improvements could be obtained by emphasizing increased computational power instead of low complexity in the design of the arithmetic unit. For example, a faster adder architecture could be adopted to reduce the delay in the critical path.

For a 3-V power supply, the filter can operate at a clock rate as high as 17.4 MHz, at which point its power consumption is 10.0 mW. Note that the design of the circuits as described in Section VI allows the filter to function at supply voltages as low as 2.0 V, although the 1.6 MHz clock rate would be insufficient for digital-audio applications. However, filtering operations found in speech and telecommunication applica-

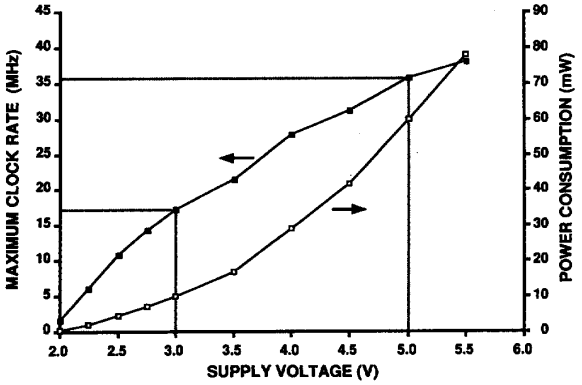


Fig. 15. Measured maximum clock rate and power consumption.

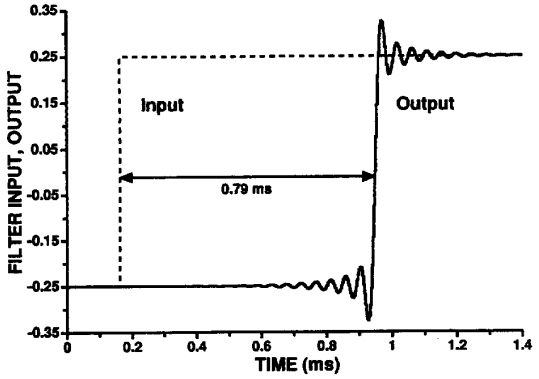


Fig. 16. Measured transient response of the interpolation filter.

tions could be performed by the filter operating on a 2-V supply. At this low supply voltage and clock rate the power consumption falls to 0.44 mW.

Fig. 16 shows the measured transient response of the interpolation filter to a step change in its input. The linear phase characteristic of the filter is evident in the symmetry of the response about the low-to-high transition. A small amount of residual image energy at multiples of 352.8 kHz is evident in the staircase nature of the transient response. The delay introduced by the interpolation filter is 0.79 ms while the decimation filter's delay is 0.68 ms.

For stereo implementations of the decimation and interpolation filters the control/coefficient ROM, the virtual address logic, and the RAM address predecode logic can be shared between the two channels. As a result, the die area and power consumption are less than twice those of single-channel versions of the filters. The estimated die area of a stereo implementation is 6500 mil^2 while the power consumption from 5-V and 3-V supplies is estimated at 33.0 mW and 11.5 mW, respectively.

VIII. Conclusion

Low area and power consumption have been achieved in a digital filter for decimation and interpolation by means of a reduction in computational complexity. The use of multistage architectures comprising sinc and halfband filters reduces the number of arithmetic computations to the extent that they can be performed with simple logic elements instead of a dedicated multiplier. A new memory addressing scheme for multirate filters reduces the number of data transfers and therefore the power consumption. The scheme features flexibility and ease of implementation while allowing the data for several filter stages of differing input rates to be stored efficiently in a single high-density RAM.

The versatility of the filter allows it to accommodate a variety of $\Sigma\Delta$ modulator architectures and decimation ratios. The filter's mask-programmable functionality allows it to perform decimation or interpolation with a variable number of stages. Furthermore, a wide class of FIR and IIR single-rate and multirate filters can be implemented for applications in digital-audio, telecommunications, speech, and ISDN. The filter's combination of small die area, low power consumption, and operation at low supply voltages will allow its use in portable battery-operated equipment. Additional information concerning the filter is available in [15].

Acknowledgment

The authors wish to thank Greg Warwar and S. S. Mahant-Shetti for their assistance in designing the static RAM and Sami Kiriaki for assistance in testing. They also acknowledge Yuichi Miyazawa of the Toshiba Corporation for his contributions to the design of the filter.

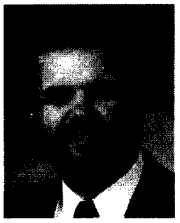
References

[1] J. J. van der Kam, "A digital 'decimating' filter for analog-to-digital conversion of hi-fi audio signals," *Philips Tech. Rev.*, vol. 42, pp. 230–238, April 1986.
[2] J. Candy, "A use of double integration in sigma delta modulation," *IEEE Trans. Commun.*, vol. 33, pp. 249–258, March 1985.
[3] R. Crochiere and L. Rabiner, *Multirate Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1983.
[4] J. Candy, "Decimation for sigma delta modulation," *IEEE Trans. Commun.*, vol. 34, pp. 72–76, January 1986.
[5] D. Goodman and M. Carey, "Nine digital filters for decimation and interpolation," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 25, pp. 121–126, April 1977.
[6] J. McClellan, T. Parks and L. Rabiner, "FIR linear phase filter design program," in *Programs for Digital Signal Processing*, Piscataway, NJ: IEEE Press, 1979, Chap. 5.1.
[7] P. Vaidyanathan and T. Nguyen, "A "TRICK" for the design of FIR half-band filters," *IEEE Trans. Circuits Syst.*, vol. 34, pp. 297–300, March 1987.
[8] E. Hogenauer, "An economical class of digital filters for decimation and interpolation," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 29, pp. 155–162, April 1981.
[9] S. Chu and C. Burrus, "Multi rate filter designs using comb filters," *IEEE Trans. Circuits Syst.*, vol. 31, pp. 913–924, Nov. 1984.
[10] A. Oppenheim and R. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1975.
[11] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 1044–1047, July 1989.
[12] S. Waser and M. Flynn, *Introduction to Arithmetic for Digital Systems Designers*. Holt, Rinehart and Winston, 1982.
[13] N. Sooch, J. Scott, T. Tanaka, T. Sugimoto and C. Kubomura, "18-bit stereo D/A converter with integrated digital and analog filters," in *Preprint 3113, 91st Audio Eng. Soc. Convention, New York*, Oct. 1991.
[14] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, Fig. 8.4, p. 314. Addison-Wesley, 1985.
[15] B. Brandt, "Oversampled analog-to-digital conversion," *Ph.D thesis*, Stanford University, Stanford, CA, Aug. 1991.

Brian P. Brandt (S'84-M'91) received the B.S.E.E. degree from Purdue University in 1985, and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University, Stanford, CA, in 1986 and 1991, respectively. His doctoral research focused on oversampled A/D conversion.

During the summers of 1985 and 1987, he was a Member of the Technical Staff at the Hughes Research Laboratories, Malibu, CA. From 1986 to 1987, he worked at the Gould Research Center, Chicago, IL. From 1991 to 1993, he was a Member of the Technical Staff in the Semiconductor Process and Design Center of Texas Instruments, Dallas, TX. He is currently a Development Staff Member in the Networking Hardware Division of IBM, Research Triangle Park, NC, where he is designing mixed-signal circuits for low-power local area network tranceivers.

Dr. Brandt received the B. Winner Editorial Award at the 1991 International Solid-State Circuits Conference. He was awarded graduate fellowships from the Hughes Aircraft Company and the Semiconductor Research Corporation and was the IEEE Fortescue Fellow for 1985-1986. He is a member of Phi Beta Kappa, Tau Beta Pi, and Eta Kappa Nu.



Bruce A. Wooley (S'64-M'70-SM'76-F'82) was born in Milwaukee, WI on October 14, 1943. He received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from the University of California, Berkeley, CA in 1966, 1968, and 1970, respectively.

From 1970 to 1984, he was a member of the research staff at Bell Laboratories, Holmdel, NJ. In 1980 he was a Visiting Lecturer at the University of California, Berkeley, CA. In 1984 he joined the faculty at Stanford University, Stanford, CA, where he is a Professor of Electrical Engineering and the Director of the Integrated Circuits Laboratory. His research is in the field of integrated circuit design and technology, where his interests include analog-to-digital conversion, digital filtering, broadband amplifier design, circuit architectures for high-speed arithmetic, high-speed embedded memory design, high-performance packaging and test systems, noise in mixed-signal integrated circuits, and circuit design techniques for video data acquisition and high-speed instrumentation.

Prof. Wooley served as the Editor of the IEEE JOURNAL OF SOLID-STATE CIRCUITS from 1986 to 1989. He was the Program Chairman of the 1990 Symposium on VLSI Circuits anhd the Co-Chairman of the 1991 Symposion on VLSI Circuits. He was the Chairman of the 1981 International Solid-State Circuits Conference, and he is also a past Chairman of the IEEE Solid-State Circuits and Technology Committee. He has served on the IEEE Solid-State Circuits Council and the IEEE Circuits and Systems Society Ad Com. In 1986, he was a member of the NSF-sponsored JTECH Panel on Telecommunications Technology in Japan. He is a member of Sigma Xi, Tau Beta Pi, and Eta Kappa Nu. He received an Outstanding Panelist Award for the 1985 ISSCC and the B. Winner Editorial Award at the 1991 ISSCC. He was awarded the University Medal by the University of California, Berkeley, CA and he was an IEEE Fortescue Fellow.