

A SYSTEM AND METHOD FOR OPTIMIZING SIGNAL
PROCESSING VIA A UNIQUE TRUNCATING AND ROUNDING
OPERATION

FIELD OF THE INVENTION

5 The present invention relates to a system and a method for building and operating a system in which a signal is processed piecewise and summed. Further, it relates to a particular combination of truncation and rounding of a signal in digital form.

BACKGROUND

10 A finite impulse response (FIR) filter is a device in which an input sample produces a finite number of output samples. After the finite number of samples expires, the FIR filter output is no longer affected by that particular input sample.

15 A commonly used FIR filter is a digital FIR filter. In select common applications it can be considered as a digital signal processor (DSP). This does not mean that the FIR filter can operate only on digital signals, however.

20 A "digital signal" is a discrete time signal that carries a discrete number of values. Conversely, an "analog signal" is a signal that carries an infinite number of values, whether continuous time or discrete time. A digital signal may be created from an analog signal through sampling and quantizing the analog signal. Sampling an analog signal refers to "chopping" the signal into discrete periods and capturing an amplitude value from the signal in selected ones of those periods. The captured value becomes the

value of the digital signal during that sample period. Such a captured value is typically referred to as a sample.

Quantizing refers to approximating a sample with a value that may be represented on a like digital signal. For example, a sample may lie between
5 two values characterized upon the digital signal. The value nearest (in absolute value) to the sample may be used to represent the sample, or, the sample may be represented by the lower of a two values between which the sample lies. After quantization, a sample from an analog signal may be conveyed as a digital signal. This is the resultant signal upon which the FIR
10 filter operates.

Colloquially, a DSP transforms an input signal (received in digital form) to a digital output signal. For a common use of the digital FIR filter, the transformation involves filtering out undesired spectral portions of the input signal. When filtering an original analog signal, the signal may be
15 represented as a sum of multiple sinusoidal signals. Each sinusoidal signal oscillates at a particular and unique frequency. Filtering is used to remove certain frequencies from a received signal while leaving other frequencies intact.

Transversal filters are filters, of which the FIR filter is a class, in
20 which a certain number of past samples are used along with the current sample to create each output sample.

FIR filters typically employ an instruction set (i.e., actually coefficients) and hardware design for programming of desired signal filtering. A program is a list of instructions which, when executed, performs
25 a particular operation (i. e., a signal transformation). Programs executing on digital FIR filters often do so in "real-time." Real-time programs are programs that must execute within a certain time interval. Regardless of

whether a program executes in a large period of time or a small period of time, the result of executing the program is the same. However, if real-time programs are executed in an amount of time longer than the "allowed" time interval, then they may no longer compute the same result.

5 If the program cannot complete manipulating a sample before the next sample is provided, then the program will eventually begin to "lose" samples. A lost sample does not get processed, and therefore the output signal of the FIR filter no longer contains all of the information provided in the (sampled) input signal to the FIR filter.

10 An FIR filter is normally fixedly programmed to perform specific modifications of a signal in real time. An FIR filter must be capable of executing a fixed number of modifications before the next sample is provided.

 Besides considering the throughput by which an FIR filter is
15 characterized, an additional consideration to their design is cost. One important cost factor is the silicon area occupied by the FIR filter. Those which are manufactured on a relatively small silicon chip are typically less expensive than those requiring a larger silicon chip. Therefore, an easily manufacturable, low cost FIR filter is also desirable.

20 For some slower applications of an FIR filter, such as for use in modems, chip area may be maintained while increasing the effective size of the instruction memory by decreasing the size of the instruction set. One method of decreasing the size of an instruction set is to encode the information in as few bits as possible. Unfortunately, these instructions
25 require complicated decoding circuitry to determine which of the instructions is currently being executed. Such decoding circuitry also may require a large silicon area or a large amount of time, or both. A method for

processing that reduces the number of required operations by truncating or rounding, or both would serve to optimize performance while conserving silicon area.

Other applications that rely on maintaining high throughput, such as applications involving mass data storage systems, e.g., hard disk drives, can benefit from increased speeds available when less bits are required to be processed. In a magnetic disk storage system, for example, information is recorded by inducing a pattern of magnetic variations on the disk, thus encoding the information. The magnetic variations are recorded along concentric circular tracks on the disk. The linear density along any given track is ever increasing together with the number of tracks placed on the disk.

As the recording density is increased, however, the magnetic readback signal from the disk becomes more and more difficult to read and interpret, due in part to inter-symbol interference (ISI). ISI results from process-time overlaps and the reduced spacing between neighboring magnetic flux patterns along an individual track, as well as between those on adjacent tracks. For drives with interchangeable disks, in particular, each disk may introduce its own irregularities into the readback signal due to naturally occurring variations within manufacturing tolerances. Moreover, the irregularities are not uniform even over an individual disk, but depend to some degree on the radial position of the data transducer with respect to the head.

The dynamic power dissipated in conventional filter circuit implementations (assuming the use of CMOS ICs) is given by the relationship:

$$P \propto C \times V^2 \times f \times N_{Gate} \quad (1)$$

where:

- C = the average loading capacitance of a gate in the IC chip, μf
- V = the power supply voltage level, v
- 5 f = the operating frequency, MHz and
- N_{Gate} = the number of gates that are switching at f frequency.

Improved performance is generally realized with a higher operating frequency, f , but comes at the expense of higher power dissipation levels. From Eqn. (1), power consumption also increases in proportion to the
10 number of gates.

A common IC embodiment of FIR filters is a tapped delay line, in which each of the coefficients characterizing the filter corresponds to a separate "tap" along a delay line. The number of gates goes up in proportion to the number of taps. As the number of taps increases, so does latency
15 (clock period delays).

Some features of FIR filters that are important to the design engineer include phase characteristics, stability (although FIR filters are inherently stable), and coefficient quantization effects. To be addressed by the designer are concerns dealing with finite word length and filter performance.

20 When compared with other filter options, such as infinite impulse response (IIR) filters, only FIR filters have the capability of providing a linear phase response, and are inherently stable, i.e., the output of an FIR filter is a weighted finite sum of previous inputs. Additionally, the FIR filter uses a much lower order than the generic Nyquist filter to implement the
25 required shape factor.

Coefficient quantization error occurs as a result of the need to approximate the ideal coefficient for the "finite precision" processors used in real systems. The net result due to approximated coefficients is a deviation from ideal in the frequency response.

5 Quantization error sources due to finite word length include:

- input/output (I/O) quantization,
 - filter coefficient quantization,
 - uncorrelated roundoff (truncation) noise,
 - correlated roundoff (truncation) noise, and
- 10 • dynamic range constraints.

Input noise associated with the analog-to-digital (A/D) conversion of continuous time input signals to discrete digital form and output noise associated with digital-to-analog conversion are inevitable in digital filters. The further propagation of this noise is not inevitable, however.

15 Uncorrelated roundoff errors most often occur as a result of multiplication errors. For example, to retain accuracy to a given place value during multiplication while handling a reasonable number of bits, a decision as to the finite number of bits to be stored must be made. Once a number is specified by the system architect, some of the least significant bits (LSBs) of

20 the remainder of any interim, or partial, product are truncated. This results in "multiplication" noise being propagated through the remaining operations.

Correlated roundoff noise occurs when the products formed within a digital filter are truncated. Correlated roundoff also causes "limit-cycle effect" or small-amplitude oscillations. For systems with adequate

25 coefficient word length and dynamic range, this latter problem is negligible.

However, both overflow and limit-cycle effects force the digital filter into non-linear operation.

Constraints to dynamic range, such as scaling parameters, are used to prevent overflows and underflows of finite word length registers. For an
5 FIR filter, an overflow of the output produces an error. If the input has a maximum amplitude of unity, then worst case output is:

$$y(n) = \sum_{n=0}^{N-1} C_{\max} \times x(n) = s \quad (2)$$

Where:

10 $C_{\max}$ = coefficient at worst case
 s = scaling factor
 $x(n)$ = input
 $y(n)$ = output

Guaranteeing $y(n)$ is a fraction means that either the filter's gain or the input has to be scaled down by "s." Reducing gain implies scaling the filter
15 coefficients to the point where a 16-bit coefficient, for example, would no longer be used efficiently. Another result of this scaling is to degrade frequency response due to high quantization errors. A better alternative is to scale the input signal.

A typical example of a high-speed FIR with five or more coefficients
20 is a Type II FIR. A Type II FIR is based on an array of costly Multiply and Add (MAC) accumulation stages. A conventional system using MAC is constrained to a minimum number of gates to achieve a given partial product accuracy. For example, an 8-tap FIR filter processing a 6-bit input signal requires a minimum of 13 bits after the third tap to adequately accumulate
25 the products. If each of these bits is retained, the number of gates required

impacts the silicon area out of proportion to the increase in accuracy achieved.

Digital implementation of an FIR filter is also limited by the maximum number of logic gates that can be inserted between reclocking stages established by the filter's clock cycle. Thus, for a given digital process, a minimum time to process is established by the propagation time through the critical path. To achieve very high speeds of processing, the critical path is broken into a number of shorter paths that can be addressed at higher clock speeds, i.e., processed within a short clock cycle.

10 The most straightforward digital FIR filters use the "signed 2's complement" numbering system. This numbering system is noted for its simplicity, and is more than adequate for low-speed FIR filters. Other digital FIR filters use a Radix-4 numbering system, which, although offering some improvement, does not fully exploit the 6-bit binary data format
15 universally used in high-speed read channels, for example. Finally, some architectures have used Radix-8 numbering for the sole purpose of encoding coefficients.

Some conventional high-speed systems employing FIR filters use an analog FIR filter placed before an analog-to-digital (A/D) converter. This
20 prevents the FIR filter's latency from accumulating in the sampled timing recovery loop. This method is inherently not well suited to digitally intensive designs.

A "pipelining" method is normally used to achieve better FIR performance at high input data rates. The cost of using this method is
25 increased latency, however. At very high speeds, such as those used in newer systems, conventional pipelining falls subject to the law of diminishing returns. The pipelining "overhead" now consumes a larger

percentage of the benefits gained from higher clock speeds. The overhead consists of a required latching or reclocking stage for every pipelining command. Generally, the performance improvement for one level of pipelining is less than two while the hardware cost increase is greater than
5 two. All the while this is occurring at the very high clock rate of the input data.

Some existing designs always include the FIR filter in the timing recovery loop, increasing latency *ab initio*, and decreasing stability of the embedded loops, both the timing recovery and gain loops, for example.

10 In magneto-resistive (MR) heads using FIR filters, with their inherent response nonlinearities, this constraint is becoming even more unacceptable. There are more modern methods that achieve a fully digital solution, but these are generally extremely complex while covering a disproportionately large area on a silicon chip, for example.

15 There have been several approaches to achieving performance improvement of FIR filters. One involves converting a digital signal to log values, thus avoiding the use of multipliers. A second more traditional technique uses oversampling. Yet another approach uses variations of multiplexing. An example uses a multiplexed data stream, provided as an
20 input to a tapped delay line such that the filter provides a multiplexed output of alternated samples.

For those data streams that have a high dynamic range, a method involving splitting the sampled input signal into two portions and addressing each separately in separate filters has been proposed. Of course, this
25 doubles the number of operations and the hardware required.

To reduce hardware complexity and computational intensity for relatively low-speed applications, such as modems, cascaded arrangements

of data registers receive digitally encoded data and sequentially clock the samples. Each data register has a data capacity greater than twice the code width of a digitized sample, permitting each channel to store both I and Q data. Because the data capacity need be greater than twice the input, the
5 data rate of devices with which this can be used is relatively low.

Some of the above approaches introduce additional complexity not required in the preferred embodiments of the present invention while others may not be suitable for high-speed applications.

SUMMARY

10 A preferred embodiment of the present invention provides a system and method for optimizing the operation of FIR filters and transversal arithmetic circuits having an accumulation function, in general, by using a specific method of truncating and rounding products of binary arithmetic operations. By uniquely truncating and rounding using the method of the
15 present invention, less hardware and, therefore, less silicon area is required. In addition, acceptable accuracy is maintained and power consumption is reduced.

Specifically, for an 8-tap digital FIR filter with 6-bit coefficients of "1/32" resolution, the method involves truncating and rounding of the
20 required 14-bit accumulated signal as it is processed at each of the eight taps. Further rounding can also be applied at the output, if desired. (In addition, by setting an acceptably low probability for overflow and calculating the consequences thereof, the theoretically required 14-bit accumulated signal can be reduced to 13 bits, or fewer, depending on system
25 requirements. When used in conjunction with the combination truncation and rounding of a preferred embodiment, this initial reduction of one or more most significant bits (MSBs) permits an "optimally few" number of

bits to be processed at each stage.) Calculations and simulations demonstrate that either the two or three least significant bits (LSB) can be truncated and the very next LSB value, i.e., from either the third or fourth LSB, rounded to the value closest to 1 or 0. Note if this bit to be rounded is 5 0, it is equivalent to discarding or truncating it.

In a preferred embodiment, this "truncate q-round 1" solution, where q can be a low integer, for example, either 2 or 3, is specifically optimized to an 8-tap FIR requiring 13 bits to describe the resultant accumulated signal (with low probability of overflow). Other FIR filters, or like circuits having 10 differing numbers of stages, would be optimized using that combination of truncating and rounding that yields an acceptable bias (or averaged dc offset, hereafter referred to as bias) on the order of a fraction of the final output LSB's value.

In addition to using a uniform truncation and rounding operation at 15 each stage, the truncation and rounding operation could be customized to achieve different combinations at the different stages of circuit operation.

In some applications it may be beneficial to proceed with a non-uniform truncation and rounding operation, such that fewer bits are truncated in the early stages than in the later stages. The last stage may require only 20 one sub-LSB bit (of weight $1/2$ with respect to the external output), if the final optional output rounding is going to be performed. Each preceding stage requires one more LSB bit for rounding, if high numerical accuracy to the next stage is required. In certain applications, therefore, it might be advantageous for each preceding stage to have one more LSB available for 25 maintaining the numerical accuracy of the output. This embodiment is also covered by the present invention.

In addition, a final rounding operation upon completion of all stages may be employed to enhance system performance at follow-on operations. This embodiment is also intended to be covered by the present invention.

Some of the salient advantages of the present invention are that it:

- 5 • reduces required silicon area of the chip significantly.
- reduces fabrication cost.
- uses less components.
- facilitates increases in operating speed.
- facilitates reductions in latency.
- 10 • reduces overhead.

BRIEF DESCRIPTION OF DRAWINGS

Figure 1 is a sample 8-tap FIR filter showing a multiply-and-add accumulation function (MAC) at each stage for use in conjunction with a preferred embodiment of the present invention.

15 Figure 2 depicts a mass data storage system and a read block diagram showing an environment in which the invention may be practiced.

Figure 3 is a block diagram of the read channel circuit of a preferred embodiment.

Figure 4 depicts a plot of the bias as a result of truncation alone and a
20 plot of the bias as a result of rounding (with implicit truncation).

Figure 5a is a graph of the results of 2-bit rounding (with implicit truncation).

Figure 5b is a graph of the results of 2-bit truncation alone.

Figure 6 is a graph representing the superposition of truncation and
25 rounding effects.

Figure 7a shows operations on the partial products of only a few LSB bits (for illustration purposes), truncation of three LSB bits for a partial product summation and accumulation structure, with and without rounding.

Figure 7b shows operations on the partial products of only a few LSB bits (for illustration purposes), truncation of two LSB bits for a partial product summation and accumulation structure, with and without rounding.

Figure 8a shows operations on the partial products of only a few LSB bits (for illustration purposes), as a result of truncating 3 bits and rounding 1 bit at each stage.

Figure 8b shows operations on the partial products of only a few LSB bits (for illustration purposes), as a result of truncating 2 bits and rounding 1 bit at each stage.

Figure 8c shows operations on the partial products of only a few LSB bits (for illustration purposes), with full resolution, i.e., neither truncation nor rounding.

Figure 9 is a flow diagram of a method of design of a preferred embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

It will be appreciated that any method that minimizes the number of multiplication steps, especially at higher data rates, will also reduce noise and increase inherent accuracy. A further improvement could also be realized if the error introduced by the truncation or rounding process could be minimized in the general case. These include the class of "overflow oscillations." Overflows are caused by additions resulting in large amplitude oscillations. Further, one alternative to overcome noise is to scale the input signal. Scaling results in a reduction in signal-to-noise ratio (SNR),

however, the scaling factor used is normally less than two, not affecting the SNR drastically. As an alternative to this type of scaling, an improvement could be realized if any quantization error introduced by the truncation or rounding process could be minimized in the general case.

5 The class of FIR filters with k coefficients fulfills the relationship:

$$Y(n) = \sum_{j=0}^{k-1} C(j) \times X(n-j) \quad (3)$$

where:

- $C(j)$ = coefficient of the filter with $X(n)$ as an input sample and $Y(n)$ as an output sample
- 10 $Y(n)$ = sum of the products over the interval $j = 0 \dots k-1$
- j = the index
- $X(n)$ = the most recent value of the input sample
- $X(n-j)$ = the sample value associated with $C(j)$

Of course, a serial implementation of the above relation implies that
 15 for every output sample it is necessary to perform, in sequence, " k " products and " $k-1$ " sums. Mathematically, this is the product of a vector, or row, consisting of the k coefficients of the filter and a vector, or column, consisting of the k consecutive input samples. In turn, this requires the provision of a clock signal having a period at least k times as small as that of
 20 the sample size.

Referring to Fig. 1, a preferred embodiment of the FIR filter 100 includes eight nodes 101 with Multiply-and-Accumulate operations carried out at each tap. The filter 100 also has eight cascaded delay elements D_0 through D_7 102. The eight nodes 101 and the delay elements D_0 through D_7
 25 102 are connected to eight multipliers M_0 through M_7 103. The outputs of

the multipliers 103 are connected to summers S_0 through S_6 104 and accumulated through the eight delay elements 102. The coefficients, c_0 through c_7 , 105 for the multipliers M_0 through M_7 103 are provided by a coefficient memory (not shown).

5 Figure 2 shows a sample mass data storage system 1000 from which an analog signal is read, the system implementing, among other elements, a preamp 1002 for amplifying the analog signal 1001 to be input to the read channel circuit 1003 that includes an FIR filter (not shown in Fig.2), processing at later stages, after analog-to-digital conversion, a signal
10 composed of n bits at the input to the FIR filter, and p -bits as processed within the FIR filter, the number p having been pre-calculated, and outputting a signal reduced to s bits 1004, the number s having been pre-calculated, the FIR filter (not shown) employing a structure having r stages (not shown), with sufficient connections to accommodate a processed p -bit
15 signal accumulated in a multiply-and-add accumulation (MAC) sub-system (not shown) of the FIR for output to a digital signal processor 1005.

Figure 3 further depicts the read channel circuit 1003 of Figure 2. Analog circuitry 71 carries the amplified signal to the analog-to-digital converter (ADC) 72 quantized to a 6-bit digital input signal to be processed
20 by FIR filter 75. The number of bits, p , required to be processed internally within the filter 75 is pre-calculated based on a given negligible probability of overflow occurring and an acceptable truncation/rounding error level. The FIR filter uses coefficients, c_i 74, where $i = 0, 1 \dots 7$, that have been pre-determined and stored separately in memory (not shown). The FIR filter 75
25 outputs an 8-bit (i.e., $s=8$) 76 signal that is described using the least number of bits, s , that are suitable to cover the expected range of output values. The output signal is detected in detector 77 from which it is passed to DSP 1005.

A feedback line 43 is also established to insure data synchronization in both phase and time and amplitude. In this example, the output signal 76 is fed into a band/error detection circuit 78 where it is processed as two signals in parallel, and output as the 5 higher-order bits 79 and the 3 lower-order bits 80 of the output signal from the band/error detection circuit 78. These two signals 79 and 80 are then fed to a timing gradient (TG) circuit for processing prior to input as signal 83 to a phase-locked-loop (PLL) 84 for achieving final synchronization at the ADC 72. The signal 83 is also used as input to automatic gain control (AGC) circuitry 82 for feedback and gain adjustment of analog circuitry 71.

In addition to the steps needed to process a signal as diagrammed in Fig. 3, the number of bits needed to describe the signal during processing contribute to the area of silicon "real estate" required. If the number of required bits can be optimized for each design, this ripples down through every required operation and has a significant impact on the required silicon area needed on a chip. Further, fewer lines and gates can lead to more efficient layouts. In order to estimate the total impact of the bit-size of a processed signal on a given filter design, several alternatives may be mathematically evaluated.

Before continuing, it will be important to differentiate the meaning of terms to be used. First is the distinction between the "internal" signal (to the FIR filter) and the "external input" signal. The external signal begins as an amplified analog signal 1001 that is fed to the read channel circuit 1003. There it is converted to a digital signal in an ADC circuit 72 that is internal to the read channel circuit 1003. In an exemplary embodiment, the converted signal comprises a 6-bit digital signal as shown at 73. Once this input signal is provided to the FIR filter 75, wherein it is multiplied and

accumulated within FIR filter 75, it becomes the "internal signal." The value of each bit gets larger as it accumulates through multiple stages of the FIR filter 75. For example, in an 8-tap FIR filter, the theoretical number of bits that will assure zero overflow is 14. Thus this "internal signal" is
 5 processed as a 14-bit signal in commonly used circuits. Of course, it can be shown that if one were to willing accept some reasonable probability of overflow, 13, or even fewer bits, could be used (in this example).

Note the two domains, external and internal, that can be referenced when defining the system's operation. Looking at the system strictly from
 10 either an external view only or an internal view only, inputs and outputs can be viewed as being integer-based with a weight of one assigned to the LSB bit. However, if one approaches the system as internal processing viewed from an external vantage, a proper relation is described as fixed-point based arithmetic with each LSB having a weight of $\frac{1}{2^B}$, where B is defined as the
 15 modulo needed to insure that the FIR filter's coefficients have the required resolution. A common value for B is 5, although some designs call for 6. Thus, in our example, if B were chosen to be 5, each LSB would have its place value scaled by $\frac{1}{2^B} = \frac{1}{2^5} = \frac{1}{32}$ when looked at as an internal signal from an external vantage.

20 A first evaluation compares the "bias" induced for truncation alone and rounding (with implicit truncation) of multi-bit binary numbers. For the general case, the following relationships apply:

$$\Delta_R = \begin{cases} \frac{1}{2}ULP, N > 0 \\ 0, N = 0 \end{cases} \quad \text{(rounding bias)} \quad (4a)$$

$$\Delta_T = -\left(\sum_{i=0}^{N-1} i\right) \frac{1}{M} \text{ULP} = -\frac{M-1}{2} \text{ULP} = -\frac{2^N-1}{2} \text{ULP} \quad 4(b)$$

where:

- M = weight of first non-truncated (or non-rounded) bit
N = number of truncated (or rounded) bits
5 ULP = Unit-in-Last-Position (i.e., existing "internal" LSB)
i = index

(The above relationships show the general case for all processing done internal to a system that embodies an accumulation function, e.g., a FIR filter. In subsequent discussions, specific integer arithmetic examples will
10 be discussed in order to simplify explanations. This means that henceforth "ULP" is assumed to have an integer value of 1.)

A simple comparison of five separate truncations and five separate roundings, all values computed using equations 4(a) and 4(b), is depicted in Table 1.

<u>N</u>	<u>Δ_T</u>	<u>Δ_R</u>
0	0	0
1	$-\frac{1}{2}$	$\frac{1}{2}$
2	$-\frac{3}{2}$	$\frac{1}{2}$
3	$-\frac{7}{2}$	$\frac{1}{2}$
4	$-\frac{15}{2}$	$\frac{1}{2}$
5	$-\frac{31}{2}$	$\frac{1}{2}$

Table 1

A review of the data shows that as N gets large the bias for truncation, Δ_T , tends to a larger negative number while that for rounding, Δ_R , is always $\frac{1}{2}$ for every value of N except zero. Fig. 4 graphically shows Δ_T and Δ_R as N gets large. This observation led to evaluating combinations of bit truncations with bit roundings in order to optimize the number of bits
5 "pragmatically" needing to be processed while minimizing the bias thus incurred.

Fig. 5a shows the effects of rounding (with implicit truncation), given a 2-bit rounding operation, where y , the output, is a function of x , the input,
10 and the instantaneous error, e , is $y-x$. Rounding, in the general sense, can take a number of forms. In the following case, the process of rounding implies observing the LSBs to be rounded, and if the most significant of the LSBs is one, then the number defined by the remaining bits is incremented. All LSBs less significant than the most significant to be rounded are simply
15 discarded. This is the simplest form of the biased method of rounding. For rounding, the "statistical" solution will provide about $\frac{1}{2}$ above and $\frac{1}{2}$ below the ideal decision line 301, with only those values exactly falling at 0.5 of the remaining bit's LSB causing any significant uncompensated error. These "0.5 values" may constitute only a few bits in each sample. For
20 example, if three bits are to be rounded, the two least significant bits are discarded and the third LSB is examined for a possible rounding up operation. For the positive 2-bit range (weight of first non-truncated is that of the third LSB, i.e., 2^2), the optimum solution would lie on the 45° line (i.e., the ideal decision line) 301 in Figs. 5a and 5b (drawn through the example's four repeating cycles), whereas the actual rounding provides a
25 response following the "staircase function" 302 shown in Fig. 5a. (If the error value falls on the ideal decision line, it is at the minimum that would be

statistically expected.) With 2-bit rounding the LSB is discarded and the next LSB would be rounded to the new word (third LSB) if 1, i.e., greater than 0.5; otherwise, it is also discarded.

Fig. 5b shows the effects of truncation alone, given a 2-bit truncating operation. Truncation involves discarding a number of LSBs and taking up from where the last bit was discarded. For the positive sign 2-bit range (weight of first non-truncated is that of the third LSB, i.e., 2^2), the optimum solution would lie on the ideal decision line 301 in Fig. 5b, whereas the actual truncation provides a response following the "staircase function" shown in Fig. 5b. For truncation, the "statistical" solution will always provide a negative bias below the ideal decision line, increasing negatively as the number of bits truncated increases. With 2-bit truncation, both of the two LSBs are discarded, and the third LSB now becomes the new first LSB to be used for processing at the next stage.

Fig. 6 demonstrates the combination of truncation results 601 from Fig. 5b with a single bit rounding of the third LSB 602. From results summarized in Table 1 it can be seen that a judiciously chosen combination of truncation and rounding can provide a response sufficiently close to the ideal decision line 301, since any positive bias (and accumulated error) from rounding can be countered (within acceptable limits) with a negative bias (and negative accumulated error) from truncation. It is important to note that calculations in Table 1 represent bias only and not error. Since these systems multiply and accumulate, the errors accumulate at each tap, so although it appears that rounding is constant at $\frac{1}{2}$, the bias is accumulated at each tap and hence for small numbers of truncated bits, for example, three or less, offsetting of truncation error (as opposed to bias alone) with rounding error is feasible.

Fig. 7a provides an example implementation of the truncation of three LSBs (showing only lower order bits for purposes of illustration) for the Radix-8 encoded example of Figs. 1 and 2, i.e., x_0 , x_1 , and x_2 are omitted, where c_i represents carry out bits and a_i and b_{i-3} represent partial products as presented in U. S. Patent Application Serial No. _____, attorney's docket no. TI-28613, incorporated herein by reference in its entirety. It can be seen that input a_3+b_0 is now the "least" LSB and provides, when added to x_3 , the output, y_3 .

Fig. 7a also provides an example implementation of the rounding of the fourth LSB, i.e., omitting a summing operation for partial product $a_3 + b_0$ and the previous tap x_3 bit. The horizontal lines represent the "carry out" from one summing operation to the next and w_i represents the wire connecting the partial product summations for adding to the previous tap x_i bit for the output y_i .

Fig. 8a shows implementation (showing just a few lower order bits for illustration) of the last stage of truncation of three bits: Fig. 8b provides the implementation of same truncation of Fig. 8a, but with fourth bit rounding. Fig. 8c shows the full resolution, i.e., no truncation/rounding, as starting point for Figs. 8a and 8b.

Note that a fixed point or integer representation can be used.

An example of a simple 5-tap FIR filter using a truncation of 3 bits at each stages' partial product, i.e., $N=3$, the "internal" bias, Δ_T , i.e., that bias encountered at each stage "internal" to the system, is seen to be cumulative, and can be calculated as:

$$bias = Steps \times -\frac{2^3 - 1}{2} = -\frac{35}{2}$$

If the weight of the final output is at 2^5 (for bit 5), then

$$bias = -\frac{35}{2} \times \frac{1}{32} = -0.5469$$

This “external” or “final” bias is referenced to the output port as seen by the weight given it by the last tap (number 5). As can be seen, truncation, used alone, can be improved upon for many purposes.

5 Prior to addressing the reduction of bits during processing within the FIR, it is instructive to determine how many bits are required to properly represent any signal as an input and as an output. Adjusting the number of bits both before and after processing can yield savings in time and silicon “real estate.”

10 An important engineering performance measure is that of bias. If only rounding an LSB, bias for a fixed integer operation is given by Eqns. 4a and 4b.

With respect to “induced” bias, it is evident that some combination of truncation (negative) and rounding (positive) can be arrived at so that the
15 objective of reducing the total number of bits required to be handled can be met.

Running several simulations for a typical 8-tap digital FIR filter using an internal coefficient weight (i.e., resolution), $1/2^B$, where $B = 5$, i.e., $1/32$, on MATLAB[®] with 10,000 samples, yielded the following results for a fixed
20 integer arithmetic example, as measured at the external output:

25

N			External LSB Units	
	<u>internal</u>	<u>round</u>	<u>bias</u>	<u>e_{rms}</u>
	<u>truncate</u>	<u>round</u>		
	0	0	0	
5	0	0	1	0.000 0.000
	0	1	0	-0.357 0.597
	0	1	1	0.142 0.377
	1	0	0	-0.616 0.785
	1	0	1	-0.112 0.334
10	1	1	0	-0.327 0.572
	1	1	1	0.173 0.418
	2	0	0	-0.845 0.920
	2	0	1	-0.342 0.585
	2	1	0	-0.277 0.538
15	2	1	1	0.224 0.493
	3	0	0	-1.285 1.366
	3	0	1	-0.790 0.904
	3	1	0	-0.148 0.572
	3	1	1	0.355 0.665
20	4	0	0	-2.172 2.240
	4	0	1	-1.679 1.767
	4	1	0	0.105 0.879

Table 2

As might be expected, it can be seen that bias can best be minimized by rounding one bit at the output (Line 2 in Table 2). However, a closer look at the table shows that a total of four bits can be eliminated during processing at each stage if using the mode represented as 3 truncations, 1 rounding during processing and no rounding at the output (line 15 of Table 2). This is four times the number of bits that can be dropped during processing over the next best "optimized" solution for minimum root-mean-square (rms) error (line 6 in Table 2). Compare this to the last line (4 truncations and one rounding) of Table 2. Although the bias is less for the last line than that for line 15, the bias is slightly positive for the last line and the rms error is greater than for line 15. Another acceptable alternative is to

truncate 2 and round 1 at each stage (line 11 in Table 2) leading to a slightly better value for e_{rms} while requiring one additional bit at each stage when compared with line 15 of Table 2.

5 Finally, Table 3 shows the effect of truncating two and rounding one at each stage while Tables 4a and 4b provide yet another view of this same operation. From Table 4a it can be seen that there will be no significant negative effect (in this example) from truncating "0's" since the only contribution, including the rounding of the third bit, is from the three LSB bits of a_i , the lower order partial product.

10 In effect, the results can be explained by the following relation:

$$y_i = (a_i + 2^3 b) + x_i \quad (8)$$

where it can be seen that a shift of 3 LSBs to the left yields the required result with a simple addition (within the parenthesis in Eqn. 8) that is then accumulated with x_i . Thus, multiply-and-accumulate in a Radix-8 encoded
15 system can be accomplished as an "add-and-accumulate" operation.

20

25

Start					After T	After R	Error
R		T			R	Result	
0	0	1	1	1	0 0 1	0 1	
0	0	1	1	0	0 0 1	0 1	
0	0	1	0	1	0 0 1	0 1	
0	0	1	0	0	0 0 1	0 1	0
0	0	0	1	1	0 0 1	0 1	+1
0	0	0	1	0	0 0 0	0 0	-2
0	0	0	0	1	0 0 0	0 0	-1
0	0	0	0	0	0 0 0	0 0	0
1	1	1	1	1	1 1 1	0 0	
1	1	1	1	0	1 1 1	0 0	
1	1	1	0	1	1 1 1	0 0	
1	1	1	0	0	1 1 1	0 0	
1	1	0	1	1	1 1 0	1 1	
1	1	0	1	0	1 1 0	1 1	
1	1	0	0	1	1 1 0	1 1	
1	1	0	0	0	1 1 0	1 1	
M=1					M=4	M=8	

Table 3

Table 3 is derived from Eqns. 4(a) and 4(b) with ULPs,

where: "N" LSB bits are "zeroed out"

6-BIT SIGNED COEFFICIENT – RADIX-8 ENCODING

Truncation of 2 and Rounding 1

										Rounded	Truncated	
x_{12}	x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0
a_7	a_7	a_7	a_7	a_7	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0
b_7	b_7	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0			
y_{12}	y_{11}	y_{10}	y_9	y_8	y_7	y_6	y_5	y_4	y_3	y_2	y_1	y_0

Table 4a

5

Rounding Results

										↓	←
x_{12}	x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3		↑
a_7	a_7	a_7	a_7	a_7	a_7	a_6	a_5	a_4	a_3	a_2	
b_7	b_7	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0		
y_{12}	y_{11}	y_{10}	y_9	y_8	y_7	y_6	y_5	y_4	y_3		

Table 4b

where:

x_i = the value from previous stage (tap)

a_i = the lower order partial product for Radix-8 encoding

10 b_i = the higher order partial product for Radix-8 encoding,

and

the coefficients within the boxes in the first column of Tables 3a, 3b, 4a and 4b are sign extensions.

However, if one were to truncate 3 LSBs and round the next LSB (see Tables 5a and 5b), this would introduce yet another source of error in a Radix-8 encoded operation, for example, because now b_0 , as well as a_3 , must be rounded. As shown in Fig. 4b, the rounding of a_2 into the next stage can be done completely within the lower order LSBs. One final item of note is that there is never a contribution from the x_i 's, given that each processing stage implements a uniform rounding operation.

6-BIT SIGNED COEFFICIENT – RADIX-8 ENCODING

Truncation of 3 and Rounding 1

10

									Rounded	Truncated		
x_{12}	x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0
a_7	a_7	a_7	a_7	a_7	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0
b_7	b_7	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0			
y_{12}	y_{11}	y_{10}	y_9	y_8	y_7	y_6	y_5	y_4	y_3	y_2	y_1	y_0

Table 5a

Rounding Results

									↓	←
x_{12}	x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4		↑
a_7	a_7	a_7	a_7	a_7	a_7	a_6	a_5	a_4	a_3	
b_7	b_7	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
y_{12}	y_{11}	y_{10}	y_9	y_8	y_7	y_6	y_5	y_4		

Table 5b

Fig. 9 summarizes the flow diagram of the process of a method for design of a system embodying a preferred embodiment of the present invention. The process starts with pre-calculation of the number of bits needed to adequately process the input signal, pre-calculation of p as done in
5 step 901, and to describe the range of an expected output, pre-calculation of s as done in step 901. It further shows the unique internal truncation 903 and optional internal rounding step 904 and the need to get rid of a number (B) of LSBs, where $B = p - s$ (if no MSB truncation is performed at the output) prior to an optional final rounding as depicted in step 905. The design
10 hinges on initial requirements specified by the system architect 906 and engineering optimization specified by the block designer 907 (traded off against the system architect's initial specification).

The foregoing describes the salient features of the present invention's multistage bit truncation with rounding and pre-calculation of bits required,
15 and should not be interpreted as limiting the application of, method of operation, or uses for the present invention to that specified in the foregoing.

While the invention has been shown with specific equations and methods of implementation, and further described with regard to specific filter types, it will be understood by those skilled in the art that various other
20 changes in the selection of equations and use with different combinations in implementation, or other details, may be changed without departing from the spirit and scope of the invention.

CLAIMS



I claim:

1. A method for optimizing the hardware required for a system that processes an n-bit input digital signal, the system having r stages and employing an accumulation function, in which the number of bits in the internal signal has an optional reduced number of MSBs resulting in p bits, based on the results of a decision made prior to inputting the digital signal at the first stage, comprising:
 - truncating a least significant bit (LSB) at each stage, and
 - implicitly truncating by B bits at the last stage, so as to reduce the number of output bits to s.
- 10 whereby reduction in hardware is enabled and limited error is incurred.
2. The method of claim 1 wherein rounding of a LSB is done at each stage.
3. The method of claim 1 wherein rounding of a LSB is done at the output.
4. The method of claim 2 wherein rounding of a LSB is done at the output.
5. The method of claim 2 wherein at each stage the system truncates three least significant of said LSBs and rounds a fourth LSB of said p-bit digital signal.



6. The method of claim 2 wherein at each stage the system truncates two least significant of said LSBs and rounds a third LSB of said p-bit digital signal.
7. A method for optimizing the number of bits needed to process an n-bit input digital signal processed in a system employing an accumulation function, the system having r stages of internal processing associated with c_i coefficients and for which a pre-calculation is made of optimum values for p, the optimum number of bits needing to be processed, and for s, the number of bits suitable to describe the output signal; where $n < p$ and $s < p$ and $i = r$, and the pre-calculation determines p so that an unacceptable probability of overflow is not incurred, and s, so as to match the range of the expected output signal; comprising:
- expanding the n-bit input signal to only p-bits for processing;
 - truncating a least significant bit (LSB) of the p-bit signal at each stage of internal processing, and reducing the p-bit processed signal to s-bits for describing the output,
- whereby the resultant truncating enables the reduction of the number of bits needed to be processed internally at each stage .
8. The method of claim 7 further comprising rounding an LSB at each stage wherein the LSB to be rounded is more significant than the LSB to be truncated.

9. The method of claim 8 wherein the system truncates a three least significant of said LSBs and rounds a fourth LSB,
wherein the following values are used:

$$n = 6; r > 1, s = 8, \text{ and } p = 13.$$

10. The method of claim 8 wherein the system truncates a two least significant of said LSBs and rounds a third LSB,
wherein the following values are used:

$$n = 6; r > 1, s = 8, \text{ and } p = 13.$$

11. A system into which an n -bit digital signal is input, internally processing a signal defined by p bits, the number p being less than or equal to the theoretical maximum needed for internal processing and having been pre-calculated, and outputting a signal reduced to s bits, the number s having been pre-calculated so as to cover the range of the expected output, the system employing a structure having r stages, associated with c_i coefficients, and sufficient connections to accommodate processed p -bit signals accumulated in an accumulation sub-system, where $n < p$ and $s < p$ and $i = r$, comprising:
- 5 a circuit that enables a reduction in the theoretically required number of gates by ignoring a least significant bit (LSB) of the p -bit signal at each stage during internal processing,
- 10 whereby reduction in hardware is enabled and limited error is incurred.



12. The method of claim 11 further comprising ignoring an additional LSB that is rounded at each stage wherein the rounded LSB is more significant than the truncated LSB.
13. A system into which an n-bit digital signal is input, internally processing a signal defined by p bits, the number p being less than or equal to the theoretical maximum needed for internal processing and having been pre-calculated, and outputting a signal reduced to s bits, the number s having been pre-calculated so as to cover the range of the expected output, the system implementing an FIR filter employing a structure having r stages, associated with c_i coefficients, and sufficient connections to accommodate processed p-bit signals accumulated in a Multiply-and-Add Accumulation (MAC) sub-system of the FIR, where $n < p$ and $s < p$ and $i = r$, comprising:
- a circuit that enables a reduction in the theoretically required number of gates by ignoring a least significant bit (LSB) of the p-bit signal at each stage during internal processing,
- whereby reduction in hardware is enabled and limited error is incurred.
14. The method of claim 13 further comprising ignoring an additional LSB that is rounded at each stage wherein the rounded LSB is more significant than the truncated LSB.
15. The system of claim 14 wherein the system truncates a three least significant of said LSBs and rounds a fourth LSB, wherein the following values are used:



$n = 6; r > 1, s = 8, \text{ and } p = 13.$



16. The system of claim 14 wherein the system truncates a two least significant of said LSBs and rounds a third LSB, wherein the following values are used:

$n = 6; r > 1, s = 8, \text{ and } p = 13.$

17. A mass data storage system into which an n -bit digital signal is input, internally processing a signal defined by p bits, the number p being less than or equal to the theoretical maximum needed for internal processing and having been pre-calculated, and outputting a signal reduced to s bits, the number s having been pre-calculated so as to cover the range of the expected output, the system implementing an FIR filter employing a structure having r stages associated with c_i coefficients and sufficient connections to accommodate processed p -bit signals accumulated in a Multiply-and-Add Accumulation (MAC) sub-system of the FIR, where $n < p$ and $s < p$ and $i = r$, comprising:

a circuit that enables a reduction in the theoretically required number of gates by ignoring a least significant bit (LSB) of the p -bit signal at each stage during internal processing, whereby reduction in hardware is enabled and limited error is incurred.

18. The method of claim 17 further comprising ignoring an additional LSB that is rounded at each stage wherein the rounded LSB is more significant than the LSB to be truncated.

19. The mass data storage system of claim 18 wherein the system truncates a three least significant of said LSBs and rounds a fourth LSB,

wherein the following values are used:

$$n = 6; r > 1, s = 8, \text{ and } p = 13..$$

20. The mass data storage system of claim 18 wherein the system truncates a two least significant of said LSBs and rounds the third LSB,

wherein the following values are used:

$$n = 6; r > 1, s = 8, \text{ and } p = 13..$$

21. The mass data storage system of claim 17 wherein the mass data storage system comprises a disk drive.

22. A method for processing an n-bit digital input signal in a system implementing a sub-system having an accumulation function internally processing a signal reduced to p bits, the number p having been pre-calculated so that p is a sufficient number of bits to process the n-bit input signal through the r stages, associated with c_i coefficients, without incurring an unacceptable probability of overflow, and for outputting a signal having s bits, s being the number needed to describe the output by matching the range of the expected output signal as closely as possible; the sub-system employing a structure having r stages, with sufficient connections to accommodate an internally processed and accumulated p-bit signal, where $n < p$ and $s < p$ and $i = r$, comprising:



15

expanding the n-bit input signal to only p-bits for processing internal to the sub-system;
truncating a least significant bit (LSB) of the p-bit signal; and
reducing the p-bit processed signal to s-bits for describing the output.

23. The method of claim 22 further comprising rounding an LSB at each stage wherein the LSB to be rounded is more significant than the LSB to be truncated.

24. The method of claim 23 wherein the system truncates a three least significant of said LSBs and rounds a fourth LSB,
wherein the following values are used:

$n = 6; r > 1, s = 8, \text{ and } p = 13.$

25. The method of claim 23 wherein the system truncates a two least significant of said LSBs and rounds a third LSB,
wherein the following values are used:

$n = 6; r > 1, s = 8, \text{ and } p = 13.$

5

26. A method for processing an n-bit digital input signal in a mass data storage system implementing an FIR filter, having an accumulation function internally processing a signal reduced to p bits, the number p having been pre-calculated so that p is a sufficient number of bits to process the n-bit digital input signal through the r stages, associated with c_i coefficients, without incurring an unacceptable probability of overflow, and for outputting a signal having s bits, s being the number



10

needed to describe the output by matching the range of the expected output signal as closely as possible; the sub-system employing a structure having r -stages, with sufficient connections to accommodate an internally processed and accumulated p -bit signal, where $n < p$ and $s < p$ and $i = r$, comprising:

15

expanding the n -bit input signal to only p -bits for processing internal to the FIR filter;

truncating a least significant bit (LSB) of the p -bit signal; and

reducing the p -bit processed signal to s -bits for describing the output.

27. The method of claim 26 further comprising rounding an LSB at each stage wherein the LSB to be rounded is more significant than the LSB to be truncated.

28. The method of claim 27 wherein the system truncates a three least significant of said LSBs and rounds a fourth LSB, wherein the following values are used:

$n = 6$; $r > 1$, $s = 8$, and $p = 13$.

29. The method of claim 27 wherein the system truncates a two least significant of said LSBs and rounds a third LSB, wherein the following values are used:

$n = 6$; $r > 1$, $s = 8$, and $p = 13$.

