Energy Efficient Filtering Using Adaptive Precision and Variable Voltage

Amit Sinha and Anantha P. Chandrakasan
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology, Cambridge
{ sinha | anantha } @mtl.mit.edu

***Abstract* - A Finite Impulse Response (FIR) filter architecture based on a Distributed Arithmetic (DA) approach with two supply voltages and variable bit precision operation is presented. The filter is able to adapt itself to the minimum bit precision required by the incoming data and also operate at a lower voltage so that it still meets a fixed throughput constraint. As opposed to the worst case fixed precision design, our precision-on-demand implementation has an energy requirement that varies linearly with the average bit precision required by the input signal. We also demonstrate that 50% to 60% energy savings can easily be obtained in the case of speech data.**

I. Introduction

The rapid proliferation of wireless, portable, battery-operated communication devices such as digital cellular phones, has increased the demand for energy efficient implementations of Digital Signal Processing (DSP) systems [1]. Most of these applications have a fixed throughput requirement and processing any faster is wasteful in terms of power. Finite Impulse Response (FIR) filtering is a very frequently used DSP function. A typical FIR filtering operation involves an inner product of a fixed finite length vector (called the impulse response $h[n]$) with shifted samples of the input signal as shown Equation 1, where N is the number of filter taps [2].

$$y[n] = \sum_{k=0}^{N-1} h[k] \cdot x[n-k] \qquad (1)$$

There exists a wealth of techniques proposed in literature for low power implementations of FIR filters. When the filter coefficients are fixed, the flexibility offered by a dedicated multiplier is not required. Distributed Arithmetic (DA) is a bit-serial, multiplier-less technique that exploits the fact that one of the vectors in the inner product is fixed [3]. All possible intermediate computations (for the fixed vector) are stored in a Lookup Table (LUT) and bit slices of the variable vector are use as addresses for the LUT. A four tap DA based FIR filter is shown in Figure 1. In general, an N tap filter requires an LUT of size 2^N. Area-delay tradeoffs have been explored in [4]. For instance, the LUT of size 2^N can be split up into two different LUTs of size $2^{N/2}$ at the cost of an additional adder. The output $y[n]$ is not available every cycle. If the bit precision of the data samples $x[n-k]$ is M, then a valid output $y[n]$ is available every M cycles. In the MSB first implementation of Figure 1, it has been shown in [9] that each successive intermediate value is closer to the final value in a stochastic sense.

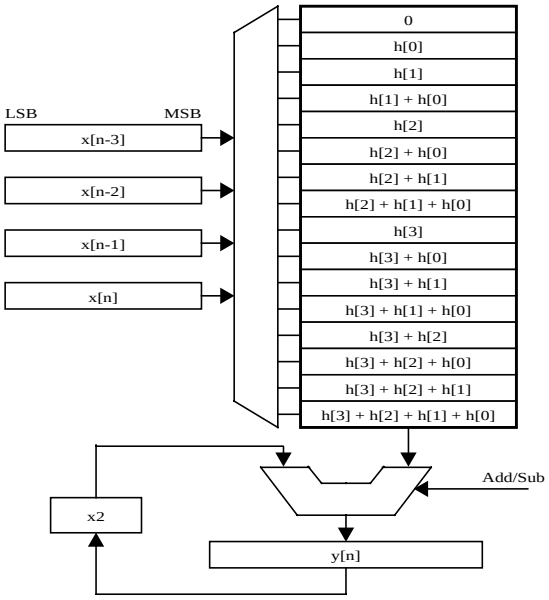

Figure 1. DA implementation of a four tap filter

If we assume that the energy consumption per cycle is E_0, it is straightforward to see that the energy consumption per output sample is ME_0 (i.e. the energy consumption is linearly dependent on the bit precision of data samples). The filter must be designed so that it can accommodate the largest possible input signals (i.e. those requiring the maximum possible bits for representation). In most signal processing applications, the signals we get are correlated and only few samples actually require the maximum bit precision for representation. The average bit precision required by a typical speech file is around 6.9 bits. Most filtering circuits will be designed to accommodate 16 bit data, and in our DA based implementation, with fixed precision, the energy required per output sample would be $16E_0$.

In this paper, we present an approach to save energy, without loss in accuracy, based on a variable bit precision filtering scheme. Approximate filtering strategies have been explored in [5] while some adaptive techniques have been explored in [6][7]. We use only that amount of precision as immediately required by the samples [8]. Let us assume that the maximum precision requirement is M_{max} and the immediate precision requirement is $M \le M_{max}$ .This scales down the energy per output sample by a factor M/M_{max}. In this work, we also exploit the fact that lesser precision implies that the

same computation can be done faster (i.e. in M cycles instead of M_{max}). We therefore switch down the operating voltage such that we still meet the worst case throughput requirement (i.e. corresponding to one output sample every M_{max} cycles when operating at V_{max}) while obtaining quadratic energy savings.

II. Filter Architecture

DA is particularly suited to variable precision filtering. The filter that we have implemented is an 8 tap low pass filter. Both data and coefficients use a 16 bit two's complement representation. The LUT has 2^8 i.e. 256 entries. There are 8 data registers (corresponding to the 8 data samples that are required for every output sample).

A. Determining Precision Requirement

To implement a precision-on-demand scheme, we need to determine the minimum precision that is required to compute the output without any loss in accuracy.

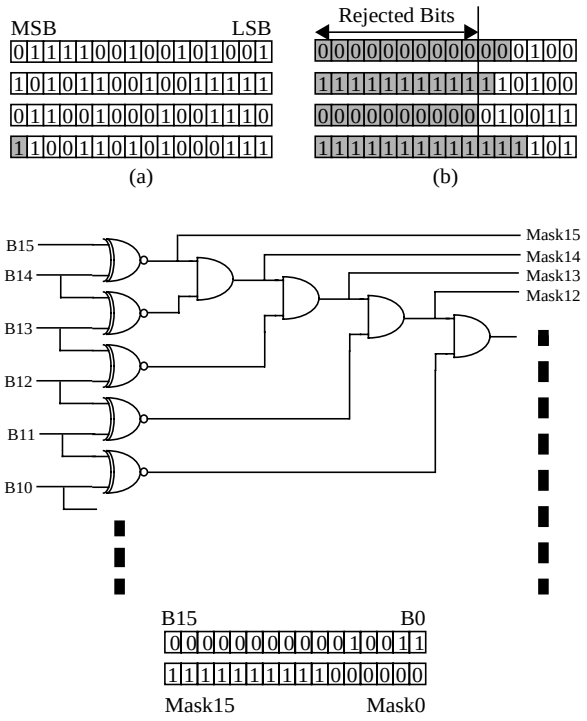

Figure 2. Determining sign extension bits

Consider the two cases shown in Figure 2 (a) and (b). In both cases (a) and (b) we have 4 registers, each with two's complement 16 bit data. The sign extension bits have been shaded in every register. Notice that in case of small numbers the MSBs are '0' for positive quantities and '1' for negative quantities. No accuracy is lost if we reject the sign extension bits. Determining the number of sign extension bits is relatively simple in hardware and can be done by the circuit shown in Figure 2 (c) [9]. The '1' outputs of the 'mask' determine the sign extension bits in each register. The number of sign extension bits that can be rejected is equal to the minimum of the sign extension bits among all the registers. This can be obtained by simply 'anding' all the individual mask outputs from each register. In our DA implementation, the final mask (obtained by 'anding' all the individual masks) determines the number of cycles, M, required for computing the current output. For a 4 tap implementation, with the register contents as shown in Figure 3 (b), the number of cycles that will be required is M = 6, instead of M_{max} = 16. The energy overhead due to the sign extension hardware is small since the duty cycle is low (one per output sample).

B. Just-In-Time Computation

Once the precision requirement for a sample has been determined we know the number of cycles, M, required for obtaining the result $y[n]$. In general, the filter will have to be designed such that it meets a fixed throughput requirement in the worst case operating precision, i.e. we have one output sample $y[n]$ at least every M_{max} cycles. But with a precision requirement $M \le M_{max}$, we would have a valid output earlier. Since, it does not pay to do computations any faster than required, we would have to idle for (M_{max} - M) cycles.

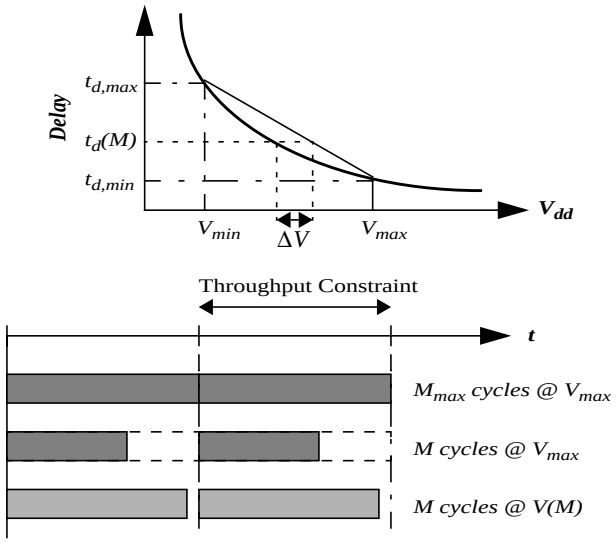

Figure 3. Just-in-time computation

We can exploit this fact, that we have more time for computation, by lowering the supply voltage. This results in quadratic reduction in energy and corresponding increase in computational delay. We lower the supply voltage, V_{dd}, from $V_{dd} = V_{max}$, to $V_{dd} = V(M)$, such that the time required to execute M cycles at $V_{dd} = V(M)$ is equal (but not greater than) the time required to execute M_{max} cycles at $V_{dd} = V_{max}$. This idea is summarized in Figure 3. The concept of voltage dithering has been explored in [11].

Implementing a scheme like this involves a variable supply voltage (a DC/DC converter) with sufficient feedback bandwidth, a variable clock frequency generator along with the precision determining circuit in the original DA implementation. Even the best DC/DC converters do not have suffi-

cient feedback bandwidth to track the fast changes in data precision requirement that filtering involves. Therefore such a scheme is difficult to implement practically.

C. Practical Implementation with Two Supply Voltages

To overcome the limitation posed by the DC/DC converter we implemented and tested a filtering scheme that uses two ROM and Accumulator (RAC) units operating at two different supply voltages, $V_{dd} = V_{max}$ and $V_{dd} = V_{min.}$ The RAC operating at the higher voltage is such that it can produce a valid output every M_{max} cycles and meets the throughput constraint. The RAC operating at the lower voltage is such that it can go through only M_{min} cycles before it exceeds the throughput constraint. If the current precision requirement is M ($M_{min} \le M \le M_{max}$), we execute some fraction, D, of M_{max} cycles, in the higher voltage RAC, and the remaining cycles, $(1-D)$, of M_{min} cycles, at the lower voltage RAC. Of course, the number of cycles we spend at either voltages, must be integral and must add up to M.

$$D \cdot M_{max} + (1-D) \cdot M_{min} = M \tag{2}$$

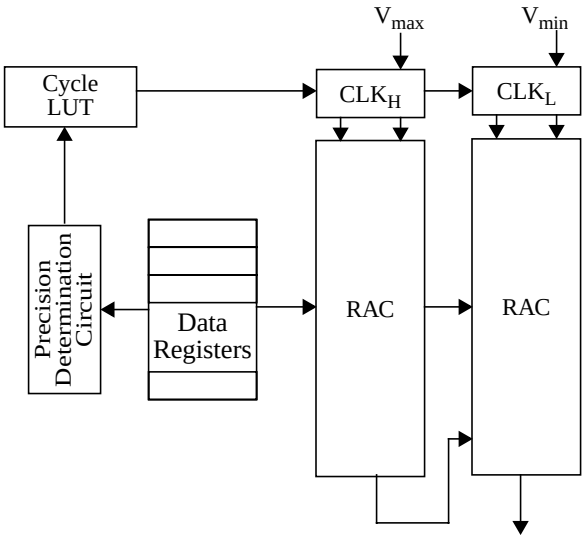

Figure 4. Just-in-time filtering with two supply voltages

Figure 4 illustrates the basic architecture of just-in-time filtering with two RAC units operating at two supply voltages. The details of the control logic and level shifters have been omitted for clarity. The operation is very simple. When a new data sample is loaded into the register, the 'precision determination circuit' computes the minimum precision required for the computation. This determines the total number of cycles, M. The accumulators within each of the RACs are cleared. The 'cycle LUT' then determines the number of cycles to be executed at V_{max} and V_{min} respectively and clocks the corresponding RACs. At the end of DM_{max} cycles at V_{max}, the accumulator contents of the RAC operating at V_{max} are loaded into the RAC operating at V_{min}, and the RAC is clocked for $(1-D)M_{min}$ cycles. The average delay per cycle in this scheme is therefore a simple linear combination of the delays $t_{d,max}$ and $t_{d,min}$. This is suboptimal compared to the arbitrary voltage scheme as shown in Figure 3, in that we are operating at an average voltage which is higher than the optimal voltage by ΔV.

In our implementation, M_{min} was chosen to be 4. If the precision requirement was less than 4, all cycles were executed at V_{min}. Therefore, once again, the two voltage scheme was suboptimal in that we did not exploit all the time we had for computation. The choice of M_{min} is data dependent. We do not want it to very small because most of the cycles would then be spent at V_{max}. At the same time we do not want it to be large because all precision requirements below M_{min} would then be suboptimal. In the next section we demonstrate an algorithm to choose the optimal M_{min} (and therefore V_{min}).

III. Theoretical Analysis

We begin with the assumption that M_{max} is fixed (determined by the maximum bit precision in the data representation which meets a specified accuracy requirement). Let the throughput requirement be one output sample every T_0 time duration. In our DA based implementation, this translates to a RAC that can execute M_{max} clock cycles within T_0 time i.e. the worst case critical path delay should be less than T_0/M_{max}. Given a particular process technology, and RAC design, we have a delay-voltage characteristic similar to Figure 3 which we represent as

$$t_d = g(V_{dd}) \tag{3}$$

Therefore, using $t_{d,min} = T_0/M_{max}$ and the delay-voltage characteristic, we can determine V_{max}.

Suppose, we also choose M_{min} (we shall establish an algorithm to optimally determine M_{min}). Once again, using $t_{d,max} = T_0/M_{min}$ and the delay-voltage characteristic, we can determine V_{min}. Let the instantaneous precision requirement be M. We need to determine the fraction of cycles that we need to execute at each voltage. From Equation 2, we compute

$$D = \frac{M - M_{min}}{M_{max} - M_{min}} \tag{4}$$

Of course, D should be such that DM_{max} and $(1-D)M_{min}$ are integral. It must also satisfy the throughput constraint

$$DM_{max}t_{d,min} + (1-D)M_{min}t_{d,max} \le T_0 \tag{5}$$

Figure 5 illustrates how DM_{max} and $(1-D)M_{min}$ vary for different M. As before, $M_{max} = 16$ and $M_{min} = 4$.

Next we determine the energy consumption per sample as a function of the required precision. Let us assume that the energy dissipated per cycle at $V_{dd} = V_{max}$ is ΔE_{max}. Clearly, in the fixed precision DA, the energy consumption per cycle is $E_{max} = M_{max}\Delta E_{max}$. If we neglect the energy overhead due to the precision determination circuit and extra controls (a valid assumption), the energy as a function of M is

$$E(M) = DM_{max}\Delta E_{max} + (1-D)M_{min}\Delta E_{max}\left(\frac{V_{min}}{V_{max}}\right)^2 \tag{6}$$

and substituting Equation 4, we get

$$E_R(M) = \alpha M - \beta \tag{7}$$

where $E_R(M)$ is the energy normalized with respect to the maximum energy E_{max} and the constants are

$$\alpha = \frac{M_{max} - M_{min}\left(\frac{V_{min}}{V_{max}}\right)^2}{M_{max}(M_{max} - M_{min})} \tag{8}$$

$$\beta = \frac{M_{max}M_{min}\left(1 - \left(\frac{V_{min}}{V_{max}}\right)^2\right)}{M_{max}(M_{max} - M_{min})} \tag{9}$$

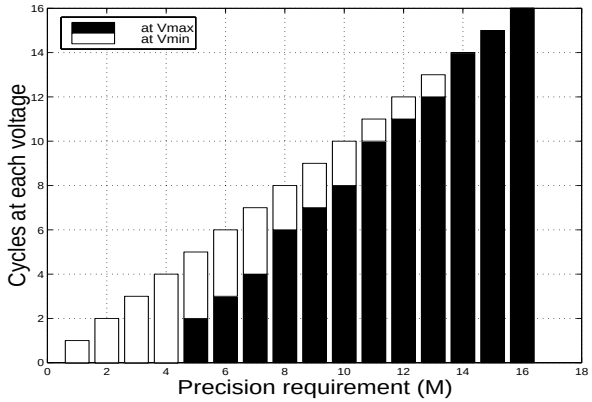

Figure 5. Cycle distribution for different precision requirements

We can clearly see the linear variation of energy with precision requirement. The voltage ratio determines the slope of the variation and also the offset. Equation 7 is valid only for the range $M_{min} \le M \le M_{max}$. As we pointed out before, for $M < M_{min}$, all the cycles are executed at V_{min} (i.e. $D = 0$).

Next, we need to determine the optimum operating voltage, V_{min}, and the corresponding precision. This cannot be done without an *a priori* knowledge of data statistics. We need to know the precision distribution profile for the data i.e. the relative frequency of a particular precision requirement in the filter for the given data. Let us assume that p_M is the probability that the precision requirement in the DA based FIR filter is M. The expected normalized energy per sample required by the filter, for the particular data, would be

$$\bar{E}_R = \sum_M p_M \cdot E_R(M) \tag{10}$$

and substituting Equation 7 we get

$$\bar{E}_R = \alpha \bar{M} - \beta \tag{11}$$

where $\bar{M}$ is the average precision requirement in the filter. To compute the optimum V_{min} we have to set

$$\frac{\partial \bar{E}_R}{\partial V_{min}} = 0 \tag{12}$$

And using $t_{d,max} = g(V_{min}) = T_0/M_{min}$ we can solve Equation 12 and obtain the optimum V_{min} and M_{min}.

IV. Results

In our implementation V_{max} = 2.1 V, V_{min} = 1.3 V, M_{max} = 16, and M_{min} = 4. Figure 6 illustrates the precision distribution for typical speech data. Notice that the distribution peaks around M = 4 which implies that a large fraction of the data would require only the RAC operating at V_{min} to execute all cycles.

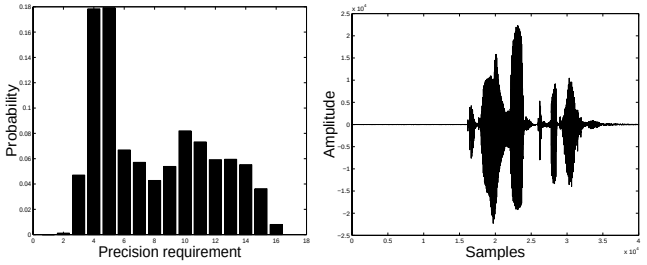

Figure 6. Probability distribution of precision requirement

Figure 7 shows the energy consumption (normalized) sample by sample for the speech data shown in Figure 6. It can be clearly seen that the energy requirement tracks the precision requirement. For example, the first 15,000 samples, where the data values are relatively small, the energy requirement is less than 20% of the maximum requirement. On the other hand, the next 10,000 samples being large (requiring close to the maximum precision) use up a lot more energy. The average energy required in this case is about 40% of the maximum i.e. our scheme saves about 60% energy. This is perfect agreement with our theoretical analysis in Section 3.

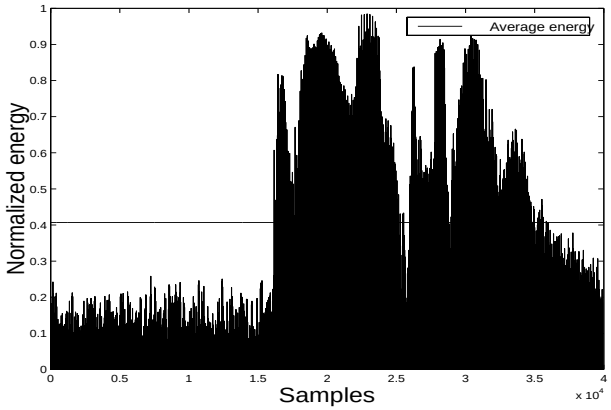

Figure 7. Sample by sample energy requirement

Figure 8 illustrates the time that has to be spent at V_{max} and V_{min} respectively, to meet the throughput requirement at each precision M. Notice that for $M < M_{min}$ (which is 4 in our case), the filter has to idle for some time. Also, all cases $M \ge M_{min}$, are not "just-in-time" because the number of cycles that have to be spent at each voltage is integral.

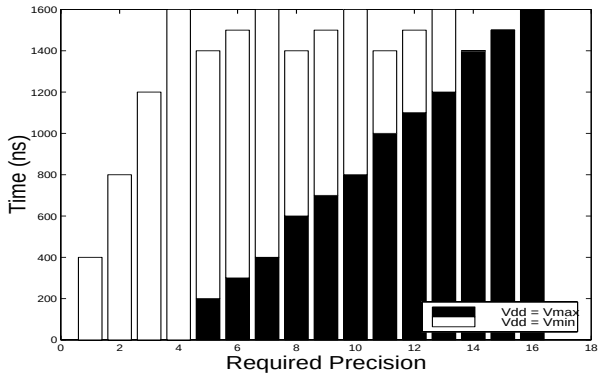

Figure 8. Time/cycles spent at each voltage

Finally, Figure 9 shows the results of our filtering scheme on 37 different speech data blocks. We can clearly see the linear variation of average energy requirement with the average precision requirement as was predicted by Equation 11. Once again the energy is normalized with respect to the maximum energy. For the fixed precision implementation, all the speech blocks would require almost the same energy equal to the maximum energy.

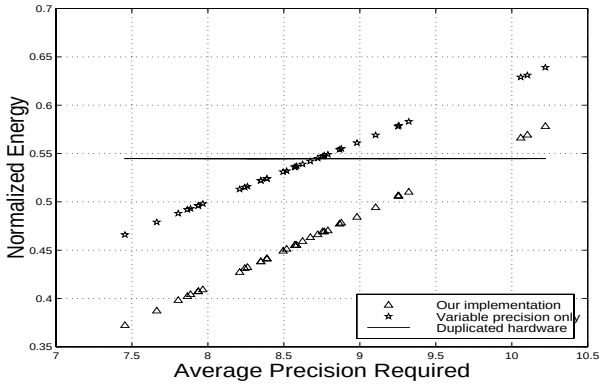

Figure 9. Average energy as a function of average precision

An interesting case to compare would be the classical area power tradeoff that one gets from duplicating hardware [10]. In our case that translates to having two RACs running at some intermediate voltage V_m such that each RAC is able to execute M_{max} cycles in the twice the throughput interval. So, we would have two such units running in parallel in such a fashion that we have one output sample every T_0 time. Clearly, this would result in an energy reduction by a factor $(V_m/V_{max})^2$. The energy reduction is fixed and independent of precision requirement as shown in Figure 9. However, if the data samples are such that the precision requirement is high, then the energy advantage of our scheme would be lost. Also plotted in Figure 9 is the average energy variation with average precision requirement for the just the variable precision case (i.e. variable precision using single RAC, single operating voltage). The difference in slope and offset, as predicted by Equation 8 and Equation 9 respectively, is immediately visible.

V. Conclusion

We have demonstrated an energy efficient DA based FIR filter architecture which uses two supply voltages to adapt itself to the immediate precision requirement of the data and perform just-in-time computation. We have also demonstrated that 50% to 60% energy savings can easily be obtained in the case of speech data with little hardware overhead to the fixed precision circuit. As future work it will be interesting to explore a just-in-time, approximate processing scheme where not only the precision but also the filter coefficients themselves adapt to the data characteristics. This would require an efficient technique to load the LUTs on the fly and also a tap-determination algorithm which decides the number of taps in the filter.

Acknowledgments

This work was supported by an NSF Career Development Award MIP-9501995.

References

[1] A. Chandrakasan and R. Brodersen, *Low Power CMOS Design*, IEEE Press, 1998.

[2] A. V. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*, Prentice Hall, 1997

[3] Stanley A. White, “Applications of Distributed Arithmetic to Digital Signal Processing : A Tutorial Review”, IEEE ASSP Magazine, July 1989, pp. 4-19

[4] M. Mehendale, S. D. Sherlekar and G. Venkatesh, “Area-Delay Tradeoff in Distributed Arithmetic based Implementation of FIR Filters”, X International Conference on VLSI Design, Jan. 1997, pp. 124-129

[5] J. T. Ludwig, S. H. Nawab and A. P. Chandrakasan, “Low-Power Digital Filtering Using Approximate Processing”, IEEE Journal of Solid-State Circuits, Vol. 31, No. 3, March 1996, pp. 395-399

[6] M. Goel and N. R. Shanbhag, “Low-Power Equalizers for 51.84 Mb/s Very High-Speed Digital Subscriber Loop (VDSL) Modems”, IEEE Workshop on Signal Processing Systems, 1998, pp. 317-326

[7] C. J. Nicol, et. al., “A Low-Power 128-Tap Digital Adaptive Equalizer for Broadband Modems”, IEEE Journal of Solid-State Circuits, vol. 32, no. 11, Nov. 1997, pp. 1777-1789

[8] R. Amirtharajah, T. Xanthopoulos and A. Chandrakasan, “Power Scalable Processing Using Distributed Arithmetic”, International Symposium on Low Power Electronics and Design, Aug. 1999 (to be published)

[9] T. Xanthopoulos, “Low Power Data-Dependant Transform Video and Still Image Coding”, Ph.D. Thesis, Massachusetts Institute of Technology, Feb. 1999

[10] A. P. Chandrakasan and R. W. Brodersen, “Minimizing Power Consumption in Digital CMOS Circuits”, Proceedings of the IEEE, Vol. 83, N0. 4, April 1995, pp. 498-523

[11] V. Gutnik and A. Chandrakasan, “An Embedded Power Supply for Low-Power DSP”, IEEE Trans. on VLSI Systems, vol. 5, no. 4, Dec. 1997, pp. 425-435