Techniques for Low Power Realization of FIR Filters

Mahesh Mehendale

Texas Instruments (India) Ltd.
71, Miller Road
Bangalore 560 052, INDIA
Tel: 080-2259007
Fax: 080-2257849
e-mail: mhm@india.ti.com

S. D. Sherlekar[1], G. Venkatesh[1]

Silicon Automation Systems(India) Ltd.
3008, 12th B Main, 8th Cross, HAL 2nd stage,
Bangalore 560008, INDIA
Tel: 080-5281229
Fax: 080-5284396
e-mail: sds@sasi.ernet.in, gv@sasi.ernet.in

Abstract— In this paper we propose techniques for low power realization of FIR filters on programmable DSPs. We first analyse the FIR implementation to arrive at useful measures to reduce power and present techniques that exploit these measures. We then identify limitations of the existing DSP architectures in implementing these techniques and propose simple architectural extensions to overcome these limitations. Finally we present experimental results on real FIR filter examples that show upto 88% reduction in coefficient memory data bus power, upto 49% reduction in coefficient memory address bus power.

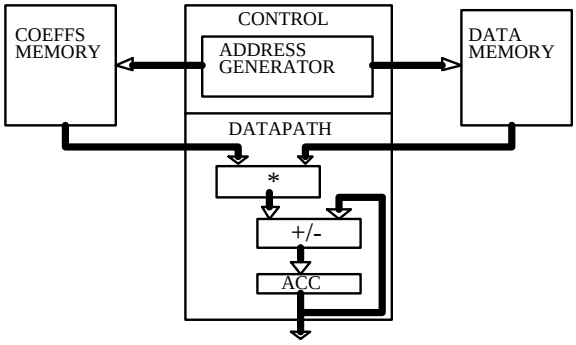

Fig. 1.: Generic DSP Architecture

I. Introduction

Various techniques have been discussed in the literature [1] to minimize power at different levels of design abstraction such as layout, circuit, logic (both combinational and FSMs), architecture and system levels. For applications built around embedded processors, a bigger saving in power is achievable by optimization at system level. Current approaches to system level power optimization [2,3] such as gray coded addressing and instruction rescheduling aim at reducing power in the control path of embedded processors. These techniques do not address potential power reduction in the data path of the embedded systems. In this paper, we present techniques that minimize both the control and data path related power in the realization of Finite Impulse Response (FIR) filters.

FIR filtering is achieved by convolving the input data samples with the desired unit impulse response of the filter. The output Y(n) of an N-tap FIR filter is given by the weighted sum of latest N input data samples.

$$Y(n) = \sum_{i=0}^{N-1} A_i \times X(n-i) \quad \ldots [1]$$

The weights (Ai) are the filter coefficients.

The low power FIR filter synthesis techniques presented in this paper are targeted to a generic architecture (figure 1). The architecture has two separate memory spaces which can be accessed simultaneously. This is similar to the Harvard architecture employed in most of the programmable DSPs[4]. One of the memories can be used to store coefficients and the other to store input data samples. The arithmetic unit performs fixed point computation on numbers represented in 2's complement form. It consists of a dedicated hardware multiplier and an adder/subtracter connected to the accumulator so as to be able to efficiently execute the multiply-accumulate (MAC) operation. An FIR algorithm can be mapped onto this architecture as a sequence of multiply-accumulate operations.

The paper is organized as follows. In section 2, we develop measures that can be used to minimize the power. In section 3 we describe techniques that exploit these measures for realizing low power FIR filters. In section 4, we identify the limitations of the existing DSPs in implementing these techniques and propose simple extensions to address these limitations. In section 5, we present the results for eight FIR filter examples and finally present conclusions in section 6.

II. Power Dissipation - Factors and Measures

A. Components Contributing to Power Dissipation

Each step in the FIR filter algorithm involves getting the appropriate coefficient and data values and performing a multiply-accumulate computation. Thus address

[1] This work was done when the authors were at I.I.T. Bombay

and data busses of both the memories and the multiplier experience the highest signal activity during FIR filtering. These hardware components hence form the main sources of power dissipation.

B. Measures of Power Dissipation in Busses

For a typical embedded processor, address and data busses are networks with a large capacitive loading [3]. Hence signal switching in these networks has a significant impact on power consumption. In addition to the capacitance of each signal, intersignal capacitance also contributes to bus power dissipation. The power dissipation due to intersignal capacitance varies depending on the adjacent signal values. The current required for signals to switch between 5's (0101b) and A's (1010b) is about 25% more than the current required for the signals to switch between 0's (0000b) and F's (1111b) [5].

The Hamming distance between consecutive signal values and the number of adjacent signals toggling in opposite direction thus form measures of bus power dissipation.

C. Measures of Power Dissipation in the Multiplier

Due to high speed requirements, parallel array architectures are used for implementing dedicated multipliers in programmable DSPs. The power dissipation of a multiplier is directly proportional to the number of switchings at all the internal nodes of the multiplier. The number of internal node switchings depend on the multiplier input values. This dependence can be analyzed using the 'Transition Density' [6] measure of circuit activity. It can be shown that the multiplier power can be reduced by reducing the number of 1s in its inputs and also by reducing the Hamming distance between successive inputs.

III. Techniques for power reduction

A. Coefficient Scaling

For a N-tap filter with N coefficients (Ai, i=0, N-1), the output Y(n) is given by equation 1. Scaling the output preserves the filter characteristics in terms of passband ripple and stopband attenuation, but results in an overall magnitude gain equal to the scale factor. For a scale factor K, from equation 1 we get
$K \times Y(n) = K \times \sum_{i=0}^{N-1} A_i \times X_{n-i} = \sum_{i=0}^{N-1} (K \times A_i) \times X_{n-i}$
Thus the coefficients of the scaled filter are given by $(K \times A_i)$. Given the allowable range of scaling (e.g. $\pm 3db$), an optimal scaling factor K can be found such that the total number of ones in the binary representations of $(K \times A_i)$ coefficients is least.

Since scaling changes the bit pattern of the coefficients, it also affects the Hamming distance between the consecutive coefficients. Thus it can also be used as a technique to reduce Hamming distance related multiplier power and also coefficient data bus power.

B. Selective Coefficient Negation

The FIR coefficients are stored in the coefficient memory in 2's complement form. For a given number N and the number of bits B used to represent it, the number of 1s in the 2's complement representation of +N and -N can differ significantly.

For each coefficient Ai, either Ai or -Ai can be stored in the coefficient memory, depending on the value that has lesser number of 1s in its 2's complement binary representation. If -Ai is stored in the memory, the corresponding product $(A_i \times X(n-i))$ needs to be subtracted from the accumulator so as to get the correct Y(n) result. This technique of selective coefficient negation can result in significant reduction in multiplier power.

Since selective coefficient negation alters the bit representation of the coefficients, it also affects the Hamming distance between the consecutive coefficients. Thus it can also be used as a technique to reduce Hamming distance related multiplier power and the coefficient data bus power.

C. Coefficient Ordering

Since the summation operation is both commutative and associative, the filter output is independent of the order of computing the coefficient products. This order however decides the sequence of coefficients appearing on the coefficient memory data bus and hence the power dissipation in the data bus and also the multiplier.

i. Coefficient Ordering Problem Formulation

For a N-tap filter, N! different coefficient orders are possible. We reduce the problem of finding the optimum order to the problem of finding the lowest cost Hamiltonian Circuit in an edge-weighted graph or the traveling salesman problem. Since this problem is NP-complete, heuristics need to be developed to obtain a near-optimal solution in polynomial time.

We formulate the coefficient ordering problem as a traveling salesman problem. The coefficients map onto the cities and the Hamming distances between the coefficients map onto the distance between the cities. The optimal coefficient order thus becomes the optimal tour of the cities, where each city is visited only once and the total distance travelled is minimum. We use the nearest neighbour algorithm to find the optimum coefficient order.

ii. Coefficient Ordering Algorithm

The algorithm for finding the optimum coefficient order is given below :

/* build hamming distance matrix */
FOR each coefficient Ai (i=0,N-1) {
  FOR each coefficient Aj (j=0, N-1) {
    $Hd[i][j] = Count\_no\_of\_Ones(Ai \oplus Aj)$ } }

```
/* initialization */
Coeff. order list = {A0}; Latest coeff. index = 0
/* build the coefficient order */
FOR (i=1, N-1) {
   Find Aj such that ((Aj ∉ Coeff. order list) &
   (Hd [j][Latest coeff. index] is minimum ))
   Coeff. order list += Aj; Latest coeff. index = j }
```

The 'Coeff. order list' gives the desired sequence of coefficient-data product computations.

D. Gray Coded Addressing of Coefficient Memory

As discussed in section 2, the power dissipation in the address bus of the coefficient memory is dependant on the Hamming distance between consecutive memory addresses. Given the order of accessing the filter coefficients, their locations can be decided so as to minimize this Hamming distance. Gray coded addressing results in least address bus power dissipation for sequential access. In this scheme the Hamming distance between any two consecutive addresses is one and consequently the number of adjacent signals toggling in opposite direction is zero.

We now look at application of these techniques for FIR implementation on a commercial DSP, specifically the TMS320C5x family of programmable DSPs.

IV. FIR implementation on TMS320C5x

TMS320C5x is a 16 bit fixed-point DSP from Texas Instruments [7]. It has an advanced Harvard-type architecture with separate program and data memory spaces. The 'C5x performs 2's-complement arithmetic, using a 32-bit ALU and an accumulator. It also provides a hardware multiplier that performs 16x16 bit 2's-complement multiplication with a 32-bit result in a single instruction cycle. With the coefficient values stored in the program memory and the data values stored in the data memory, the FIR filter algorithm can be implemented on TMS320C5x using RPTed MAC (Multiply-Accumulate) instruction.

A. Architectural Support for Coefficient Negation

In case of the coefficients that are negated to reduce the number of 1s, the corresponding coefficient-data product values need to be subtracted from the output during FIR computation. Such subtraction cannot be accomplished in 'C5x in a single instruction.

This limitation can be addressed by adding an MSUB instruction which works the same as MAC, except that it subtracts the product from the accumulator instead of adding it. Since the ALU supports subtraction, no new computation hardware is required to support this new instruction. With the MSUB instruction, the FIR algorithm can be implemented as RPT-MAC for non-negated coefficients, followed by RPT-MSUB for negated coefficients. An even more efficient solution is to combine the MAC and the proposed MSUB instruction into a single instruction with one bit in the opcode indicating addition or subtraction. This bit can be fed to the accumulator for appropriate action.

B. Architectural Support for Gray coded addressing

In 'C5x the MAC instruction, when RPTed, increments the program memory address by 1 after every MAC operation. Gray coded addressing requires the memory addresses to increment in gray fashion. Hence RPTed-MAC cannot be used to implement the desired coefficient addressing.

Gray coded addressing can be achieved by incorporating an additional logic that converts sequential memory addresses to gray coded memory addresses. Such a logic can be implemented using XOR gates. With this logic, RPTed MAC instruction can increment the program memory address in gray sequence after each MAC operation.

C. Configurable DSP implementation

With ever-reducing feature size, it is possible to integrate more and more logic on a single chip. A typical DSP based system includes the programmable DSP and other logic that interfaces with it. Single chip configurable DSPs are now available that provide a gate-array with an embedded DSP core. The desired additional logic can be implemented using the gate array and interfaced with the embedded core. The additional hardware such as the binary to gray code converter required to support the low power realization of FIR filters using the 'C5x can thus be implemented in the gate array and interfaced with the embedded 'C5x core.

V. Results

We present results of various power reduction techniques on 8 low pass FIR filters. All the filters have 16KHz sampling frequency, but vary in terms of cutoff frequencies, passband ripple and stopband attenuation. These filters have been designed using the Parks-McClellan algorithm. The number of coefficients of these filters varies from 16 to 128.

Table I shows the impact of selective negation followed by coefficient scaling on the total number of ones in the 2's complement binary representation of the coefficients. The results show that using these techniques the total number of ones can be reduced by 43% to 63%.

Table II shows the impact of selective negation followed by scaling followed by coefficient ordering on the total Hamming distance and total number of adjacent signal toggles. The results for the 8 FIR filter examples show 66% to 88% reduction in the total Hamming distance and 82% to 94% reduction in the total adjacent signal toggles.

This directly translates into upto 88% reduction in the coefficient memory data bus power.

#taps	#1s	#neg. coeffs	#1s neg + scale	% red red.
16	120	6	68	43.3%
24	212	12	94	55.7%
32	230	12	116	49.6%
48	366	20	178	51.4%
64	498	30	214	57.0%
72	550	34	260	52.7%
96	782	48	314	59.9%
128	984	66	360	63.4%

Table I. Impact of Selective Negation and Scaling

Since the multiplier power depends on the data values as well, it is difficult to evaluate, quantitatively, the power reduction in the multiplier due to the reduction in the total number of ones and the total Hamming distance between successive coefficient values.

#taps	H.D. Init.	H.D. Neg + Scale+ Order	% red.	Togs. Init.	Togs Neg + Scale+ Order	% red.
16	102	34	66.7%	8	1	87.5%
24	158	44	72.2%	20	3	85.0%
32	204	58	71.6%	22	3	86.4%
48	350	76	78.3%	50	4	92.0%
64	452	80	82.3%	54	6	88.9%
72	510	88	82.7%	52	9	82.7%
96	700	106	84.9%	64	6	90.6%
128	952	108	88.7%	84	5	94.0%

Table II. Impact of Selective Negation, Scaling and Ordering

#taps	H.D. init.	H.D. Gray	% red.	Tog. init.	Tog. Gray	% red.
16	30	16	46.7%	7	0	100%
24	46	24	47.8%	11	0	100%
32	62	32	48.4%	15	0	100%
48	94	48	48.9%	23	0	100%
64	126	64	49.2%	31	0	100%
72	142	72	49.3%	35	0	100%
96	190	96	49.5%	47	0	100%
128	254	128	49.6%	63	0	100%

Table III. Impact of Gray Coded Addressing

Table III compares gray coded addressing with the sequential addressing in terms of total Hamming distance and total number of adjacent signal toggles. The results show that with gray coded addressing the total Hamming distance can be reduced by 46% to 49% and the total number of adjacent signal toggles reduced by 100%. This directly translates to 46% to 49% saving in the power dissipation of coefficient memory address bus.

VI. Conclusion

In this paper we have presented techniques for low power realizations of FIR filters on a generic architecture. The generic architecture, which is an abstraction of commercially available programmable DSPs, is a Harvard-type architecture with two separate memory spaces and has a dedicated hardware multiplier. The data and address busses and the multiplier form the main sources of power dissipation for FIR filters mapped onto this architecture. We have presented analysis and experimental results to establish the following measures of power dissipation: (i) Number of ones in the coefficient values; (ii) Hamming distance and the number of adjacent signals toggling in opposite directions between successive coefficient values; and (iii) Hamming distance and the number of adjacent signal toggling in opposite directions between successive coefficient addresses.

We have then presented following techniques to minimize these measures so as to reduce the power: 1. Coefficient scaling which affects measures (i) and (ii); 2. Selective negation of coefficients which affects measures (i) and (ii); 3. Coefficient ordering which affects measure (ii), and 4. Gray coded addressing which affects measure (iii).

We have presented results that show that with these techniques the power dissipation in the coefficient memory address bus can be reduced by upto 49 %, the power dissipation in the coefficient memory data bus can be reduced by upto 88%.

Finally, we have presented implementation of FIR algorithm on TMS320C5x and identified limitations of the architecture in implementing these power saving techniques. We have suggested architectural extensions that address these limitations.

References

[1] Anantha Chandrakasan and Robert Brodersen, "Minimizing Power Consumption in Digital CMOS Circuits", Proceedings of the IEEE, April 1995, pp 498-523

[2] Vivek Tiwari, Sharad Malik and Andrew Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software Power Minimization", IEEE Transactions on VLSI Systems, December 1994, pp 437-445

[3] C-L Su, C-Y Tsui and A.M. Despain, "Saving Power in the Control Path of Embedded Processors", IEEE Design and Test of Computers, Winter 1994, pp 24-30

[4] Edward A. Lee, "Programmable DSP Architectures: Part I", IEEE ASSP Magazine, October 1988, pp 4-19

[5] Jon Bradley, "Calculation of TMS320C5x Power Dissipation Application Report", Texas Instruments, 1993

[6] Farid Najm, "Transition Density: A New Measure of Activity in Digital Circuits", IEEE Transactions on CAD, Feb 1993, pp 310-323

[7] TMS320C5x User's Guide, Texas Instruments, 1993