DESIGN OF HIGH THROUGHPUT, LOW LATENCY, AND LOW COST STRUCTURES FOR LINEAR SYSTEMS

Miodrag Potkonjak
Computer &Communications Research Laboratories, NEC USA, Princeton, NJ

Mani B. Srivastava
Computing Systems Research Lab., AT&T Bell Laboratories, Murray Hill, NJ

ABSTRACT

This paper introduces heuristic transformation techniques to simultaneously optimize throughput and latency of linear time-invariant systems. The technique is based on a properly coordinated manipulation of an arbitrary initial specification by both high level synthesis and symbolic algebraic manipulation tools. The technique produces implementations not only have high throughput and low latency, but also have lower area and power requirements compared to the initial specifications. The effectiveness of the technique is demonstrated on a number of high level synthesis DSP benchmarks.

1. Introduction

Transformations such as lookahead, unfolding, retiming, and algebraic and redundancy manipulations are a key to obtaining high quality implementations from a behavioral description. They are most often used for throughput [Par89, Pot92] and area [Rab91] optimization. However, in the case of embedded and reactive DSP systems both throughput and latency are important and independent metrics of speed [Das85]. Linear systems [Fir86, Del88] are in particular an important class of systems where both throughput and latency are key design parameters.

We will present three different methods in which an arbitrary LTI system with zero initial state can be transformed, so that the resulting structures are simultaneously competitive in three important design parameters: throughput, latency and implementation cost (area). The structures in this section are based on adaptation of well known LTI structures [Fri86]. We will limit our attention to the single-input single-output case. The generalization to the multiple-input multiple-output case is straightforward, but leads to degradation of performance.

It is well known that an arbitrary linear time-invariant system can be represented using a number of different basic structures [Opp75]. We will start with well known structure for implementing LTI system with zero initial state: *Direct Form II*. Although this structure is widely recognized as a low throughput, high latency, high cost alternative, it can be transformed to a structure which has almost optimal latency-throughput trade-off, and incurs an extraordinarily low cost [Sri94]. The initial transformation steps are shown in Figure 1(a-f), and involve the application of distributivity of multiplications over addition, and constant propagation enabled by associativity, so that the number of multiplications is reduced. The next key step is retiming by moving delays forward as much as possible followed by the application of common sub-expression replication, and the repeated application of algebraic laws. The final structure can be transformed in straightforward way using the maximally fast procedure [Pot92] so the latency $T_L = m + a$, and the sample period $T_S = m + 2a$, where m is the delay of the multiplier and a is the delay of the adder (Figure 2).

Note that most multiplications in the resulting structure are of different constants with the same variable. By transforming those multiplications to the bit-level, and assuming that word length is W, all multiplications can be done using only W shifts regardless of values of the coefficients. Therefore this structure provides an answer to the important problem of minimizing the number of shifts in LTI systems - a problem which recently received

0-7803-1775-0/94 $3.00

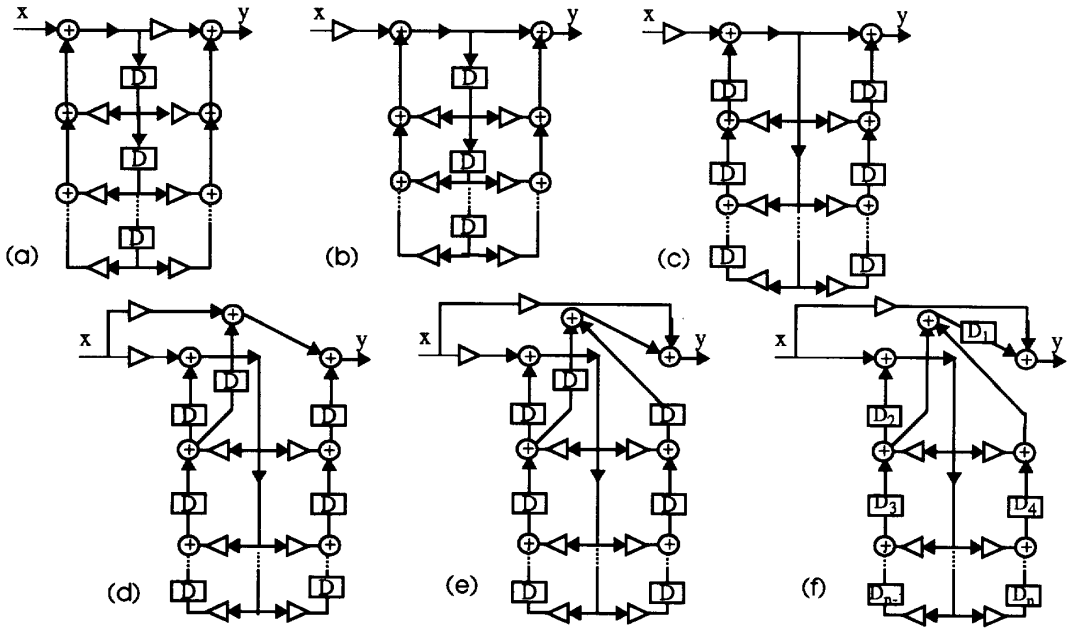

Figure 1: Transforming *Direct Form II* to a Low Latency and High Throughput Structure

	Mul		8 bit			16 bit			32 bit		
	I	N	I	SA	N	I	SA	N	I	SA	N
mat	12	6	39	38	6	83	74	14	152	130	25
ellip	20	8	77	56	8	136	87	16	240	198	32
lin4	30	10	92	57	8	212	128	16	383	279	26
lin5	28	10	81	66	8	191	117	16	348	313	26

Table 1: Comparison of Our Modified Direct Form II Structure with Published Results[Cha93] (I - Initial Design; SA - design optimized using simulated annealing; N - new design; The second and third columns show the number of multiplication; the next 9 columns show the number of shifts.

significant attention [Cha93]. Even when the

$$y = s_1 + cx$$
$$s_1 = s_4 + s_3 + cs_2 + cx$$
$$s_2 = s_3 + cs_2 + cx$$
$$s_k = s_{k+2} + cs_2 + cx$$
$$s_l = cs_2 + cx$$
$$2 \le k \le 2n-2; l = 2n-1, 2n$$

Figure 2: Functional Dependencies for the structure from Figure 1.f. c denotes various constants. Note that $T_L = m + a$ and $T_S = m + 2a$.

number of states is arbitrarily high, the LTI system can be transformed so that the number of shifts is a small constant which depends only on the word length requirements. This provides results that are far superior to all previously published approaches. Note that unlike earlier approaches, this structure does not incur any penalty on either latency or throughput, which on the contrary are actually much improved. The following table shows the improvement over the initial and the best previously published results on 4 examples (*mat1* - 1 input 3 state controller; *ellip* - 4-state 1-input elliptical wave filter; and *lin4* and *lin5* two 5-state 1 input controllers) for 8, 16 and 32 bit designs. The average improvement compared to the initial

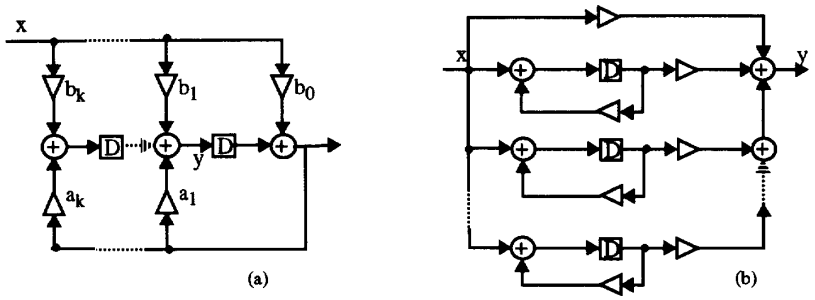

Figure 3: The Second Companion Form (a) and Diagonal form (b)

design is 9.92 times, and compared to designs optimized using simulated annealing is 7.41 times.

There is at least one more well known structure which also provides the same high quality latency-throughput characteristics ($T_L = m+a$ and $T_S = m+2a$). This is the *Second Companion Form* which is shown in graphical format in Figure 3a. Finally, some LTI systems with zero initial states can be transformed into what is known as the *Diagonal Form*, shown in Figure 3b. This form can be transformed to have $T_L = m+\lceil \log_2(R+1) \rceil$ and $T_S = m+a$, where R is the number of delays. More importantly though, the state update matrix in the state space representation is diagonal so that the number of multiplications and additions is substantially reduced in the state update equation. This can be of benefit in implementations where the cost is dominated by multiplication cost, as is often the case in programmable DSPs and microcontrollers. An even larger class of LTI systems with zero initial state can be transformed into the so called *Jordan Form* where the resulting state update matrix as a non-zero diagonal together with some elements that are 1 in the super-diagonal that runs parallel to and just above the main diagonal. This form retains the nice attributes of the diagonal form except that the achievable sample period degrades slightly to $T_S = m+2a$. Unfortunately, the process of finding the diagonal and Jordan forms is often numerically unstable, and caution needs to be exercised in using them.

2. Experimental Results

The approach presented here is implemented using the software platform shown in Figure 4, so that power of high level synthesis tools (HYPER) and algebra manipulation tools (MAPLE) is combined. The user describes the computation in SILAGE, and HYPER translates it into the control data flowgraph (CDFG) format and does the initial simulation. The

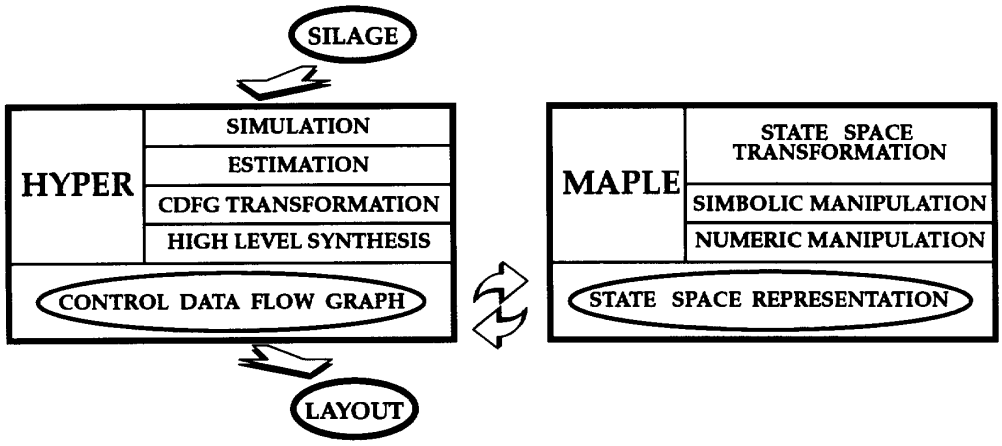

Figure 4: Software Platform Based on Coupling of HYPER 2.1 and Maple V

Design Name	Throughput [cs]		Latency [cs]		Area [mm^2]		Power [nJ/sample]	
	I	N	I	N	I	N	I	N
mat1	4	3	4	2	19.48	3.31	15.9	2.3
ellip	5	3	5	2	18.93	3.75	26.2	3.3
lin4	6	3	6	2	24.33	3.72	42.5	3.8
lin5	6	3	6	2	29.58	4.40	53.6	4.5
5WDF	17	4	16	3	10.02	5.81	21.1	4.1
7FWDF	14	3	12	2	16.45	12.89	49.2	6.2
7IIR	10	3	10	2	20.99	19.16	53.6	12.6
8IIR	18	3	18	2	38.29	7.11	33.4	4.4

Table 2: Improvements in throughput, latency, area and power for the set of benchmark examples. The throughput and Latency are states as the number of control steps, I and N denote the initial implementation and the implementation obtained using the new procedure

CDFG is then translated into Maple format, and a script is used to transform any linear CDFG into the described structures. The result is then fed back to HYPER which estimates the implementation cost and performs word length simulation. HYPER is also used for further optimizations (e.g. replacement of multiplications with constants by shift-and-additions) and final implementation.

We tested the effectiveness of the proposed structures by using, in addition to the mat1, ellip, lin4 and lin5 controllers, 4 linear filters: 5WDF - the popular 5th order wave digital elliptical filter, 7FWDF - 7th order Fetweiss wave digital filter, 7IIR - 7th order IIR cascade filter, and 8IIR - 8th order Avenhaus direct form IIR filter.

Table 2 compares the key design parameters in the initial and the final designs under the assumption of equal throughput specification for both the initial and the final designs.The average improvement of throughput and latency was by factors 3.16 and 4.48 respectively. The area of implementation was reduced by a factor 4.25, while the power was reduced by a factor 7.86.

3. Conclusion

We described how one can simultaneously optimize throughput, latency, area and power of linear systems by a properly coordinated use of high level synthesis and symbolic algebraic manipulation tools. The obtained results in terms of area and throughput are close to the best possible [Sri94], while having significantly lower area and power.

4. References

[[Cha91] B.W. Char, et. al.: "Maple V: The future of Mathematics", Springer-Verlag, New York, NY, 1991.

[Cha93] A. Chatterjee, R.K. Roy: "ARTIST: An Architectural Transformation Program for Optimization", DAC-93, pp. 243-248, 1993.

[Das85] B. Dasarthy: "Timing Constraints of Real-Time Systems: Constructs for Expressing Them, Methods of Validating Them", IEEE Trans. on Software Engineering, Vol. 11, No. 1, pp. 80-86, 1985.

[Del88] D. F. Delchamps: "State-Space and Input-Output Linear Systems", Part II and III, Springer-Verlag, 1988.

[Fri86] B. Friedland: "Control System Design: An Introduction to State-Space Methods", McGraw-Hill, Inc., New York, NY, 1986.

[Kun88] S.Y. Kung: "VLSI Array Processors", Prentice-Hall, Englewood Cliffs, NJ, 1988.

[Lee87] E. A. Lee and D. G. Messerschmitt: "Static Scheduling of Synchronous Dataflow Programs for Digital Signal Processing", IEEE Trans. on Computers 1987.

[Opp75] A.V. Oppenheim, R.W. Schafer: "Digital Signal Processing", Prentice-Hall, Englewood Cliffs, NJ, 1975.

[Par89] K.K. Parhi: "Algorithm transformation technique for concurrent processors", Proc. of the IEEE. Vol. 77, No. 12, pp. 1879-1895, 1989.

[Pot92] M. Potkonjak, J. Rabaey: "Maximally Fast and Arbitrarily Fast Implementation of Linear Computations", IEEE ICCAD, pp. 304-308, 1992.

[Rab91] J. Rabaey, et. al.: "Fast Prototyping of Data Path Intensive Architecture", *IEEE Design and Test*, Vol. 8, No. 2, pp. 40-51, 1991.

[Sri94] M.B. Srivastava, M. Potkonjak: "Transforming Linear Systems for Joint Latency and Throughput Optimization", EDAC-94, 1994.