Multiple Precision for Resource Minimization

George A. Constantinides, Peter Y.K. Cheung
Department of Electrical and Electronic Engineering
Imperial College, London SW7 2BT
{g.constantinides, p.cheung}@ic.ac.uk

Wayne Luk
Department of Computing
Imperial College, London SW7 2BZ
wl@doc.ic.ac.uk

Abstract

This paper presents the Synoptix high level synthesis and precision optimization system for FPGAs. Given abstract specifications in the form of infinite precision signal flow graphs and a set of error constraints, Synoptix will create hardware descriptions of fixed-point arithmetic implementations. The width of each signal is individually optimized in order to achieve the minimal resource utilization while satisfying user-specified constraints such as signal-to-noise ratio. A heuristic for solving the optimization problem is introduced, and the results of implementations on an Altera Flex10k-based reconfigurable computing platform are reported. It is demonstrated that significant area reductions can be obtained by optimizing signal widths individually, compared to the use of a single uniform signal width.

1 Multiple Precision Optimization

Traditionally, flexible DSP algorithms have been implemented using software running on a Digital Signal Processor. This in turn has led to the assumption of fixed-wordlength computations, in line with the width of the DSP's arithmetic unit. The advent of the FPGA and high-level CAD tools have opened up a new spectrum of possibilities for the DSP designer. However DSP design methodology for the FPGA still tends to assume that all computational units have the same fixed wordlength. Within an FPGA implementation, there is no inherent restriction against using different wordlengths at different points within a circuit, and indeed it is shown in this paper that there are valuable gains from doing so.

Synoptix is a high-level synthesis system, taking as input a multiple input, multiple output (MIMO) linear time-invariant (LTI) signal flow graph and a set of error constraints, and producing a hardware description language implementation of an application specific FPGA-based DSP coprocessor. Synoptix frees the designer from working at the bit-level, who can instead concentrate on the DSP algorithm of interest.

The problem of optimizing individual signal-widths within a DSP structure has appeared in several recent publications such as [3, 4]. Such work has tended to view the problem as one of software profiling [3], sometimes combined with format propagation or user input [4]. The approach taken in [4] is to extend the ANSI-C standard, by adding a new data type *fixed*, which monitors fixed-point errors during program execution. Such techniques allow the use of nonlinear and time-varying components. However the quality of the resulting solutions depends heavily on the input data used for simulation, and the time for simulation can be significant [3]. By restricting the problem domain to LTI systems, we are able to avoid these issues through the use of a purely analytical approach.

2 Algorithm and Results

We have developed a library of parameterizable adders, constant coefficient multipliers, multiply-accumulate blocks and registers, whose areas may be easily estimated from their parameters. In the current Synoptix implementation these are built on top of the Altera LPM parameterized macros. The extra functionality includes support for of different signal widths, implicit shifts due to non-matching signal scaling, and MSB don't care conditions arising from a transfer function analysis of the circuit (for example if two inputs to an adder never simultaneously reach their peak).

The optimization procedure used in Synoptix is now described. Initially, a uniform width structure with minimal signal width u is derived through binary search. All signal widths are then scaled-up by a factor $k > 1$. The solution is then iteratively improved. On each iteration, a single signal width is reduced by one bit, until no more reductions are possible. Which signal width to reduce at each iteration is decided by repeatedly reducing each signal width in turn until the implementation violates a noise constraint. At this point the area estimate is noted, and the signal to be reduced is the one producing the largest pay-off in area before

0-7695-0871-5/00 $10.00 © 2000 IEEE

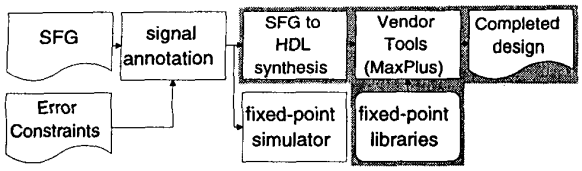

Figure 1. The design flow and tool chain for synthesis of a signal flow graph. The shaded area represents the technology-specific parts of the design flow

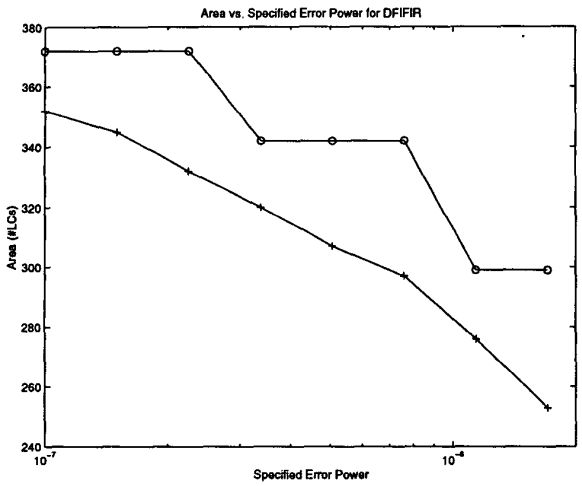

Figure 2. Achieved area against specified error power for a direct-form-I FIR filter ('o' shows uniform-width structures and '+' shows those achieved with Synoptix)

its reduction violates noise constraints. At each stage, the roundoff noise produced by the fixed-point implementation is estimated, using the analytical method described in [1].

Shown in Fig. 1 is the design flow for the Synoptix system. Synoptix has been tested on several benchmark circuits, including an allpass reverberator, a direct-form-I FIR filter, a direct-form-I IIR filter, a direct-form-II^t IIR filter, a complex multiplier, a (2-point) FFT butterfly, and an 8-point complex FFT. HDL code produced by Synoptix has been compiled and implemented within the Sonic reconfigurable computing platform [2]. Shown in Fig. 2 is a graph of actual resource usage (measured in Altera Flex10k logic cells) against specified error-power, for the two's complement direct-form-I FIR filter, which is a representative design in terms of the general shape of the plots achieved.

Two observations can be made from Fig. 2. Firstly, the plot of area for a uniform signal-width goes down in sharp steps. This is because there is a sudden jump when the next-lowest signal width becomes feasible with respect to the constraints on output noise power. The same is not true with the heuristically optimized signal widths, as there are many more variables and therefore although each variable is discrete, the range of achievable noise-powers is much broader. Secondly, the heuristic line lies consistently below the uniform line (by 5 to 15%), indicating that we may consistently save area by employing this synthesis technique. Importantly, if a circuit has more than one outputs, each with different noise constraints, the area gain is also likely to be significantly greater. This is because the uniform signal-width implementation will have to cater for the worst-case constraint, whereas the variable signal-width implementation will allow automatic optimization around the different constraints.

3 Conclusions

We have presented a high-level synthesis method for generating hardware descriptions of fixed-point systems. It has been shown that it is both desirable and possible to optimize signal widths in order to minimize implementation area. We are currently investigating the interaction between dynamic reconfiguration and the traditional high-level synthesis operations of resource binding and scheduling, when the wordlength optimization scheme described in this paper is used.

References

[1] G. A. Constantinides, P. Y. K. Cheung, and W. Luk. Truncation noise in fixed-point SFGs. *IEE Electronics Letters*, 35(23):2012–2014, November 1999.

[2] S. D. Haynes, P. Y. K. Cheung, W. Luk, and J. Stone. Sonic – a plug-in architecture for video processing. In *Proc. FCCM'99*, 1999.

[3] S. Kim, K. Kum, and W. Sung. Fixed-point optimization utility for C and C++ based digital signal processing programs. *IEEE Trans. on Circuits and Systems II*, 45(11):1455–1464, November 1998.

[4] M. Willems, V. Bürsgens, H. Keding, T. Grötker, and M. Meyer. System-level fixed-point design based on an interpolative approach. In *Proc. 34th Design Automation Conference*, pages 293–298, June 1997.