ALGORITHMIC LEVEL DESIGN SPACE EXPLORATION TOOL FOR CREATION OF HIGHLY OPTIMIZED SYNTHESIZABLE CIRCUITS

Nazish Aslam[1, 2], Tughrul Arslan[1, 2, 3], Ahmet Erdogan[1, 2, 3]

1: Institute for System Level Integration, Livingston, UK

2: Spiral Gateway Ltd, Edinburgh, UK

3: University of Edinburgh, Edinburgh, UK

ABSTRACT

This paper presents some of the results obtained by using a prototype algorithmic level design space exploration tool currently under development. The tool is based upon multi-objective evolutionary algorithms. The paper highlights the tool's benefits and discusses its current abilities in terms of its experimental applications.

Index Terms— *design automation, high-level synthesis, multi-objective, evolutionary algorithms*

1. INTRODUCTION

In order to cope with the design productivity gap [1], more novel tools are needed. The current options available to a designer are to employ a very reuse-centric methodology, which includes block-based or platform-based design methodologies. These methodologies significantly reduce the design time by promoting extensive reuse of a design and its associated verification IP. However design-for-reuse result in suboptimal designs as they introduce a lot of redundancies in order to accommodate for the increased flexibility of the IP. Another approach to deal with the productivity gap has been to shift to higher levels of abstraction for the design implementation. This is already achievable with today's commercial tools such as Catapult C from Mentor Graphics [2] or System Generator from Xilinx [3]. These tools reduce the number of 'over-the-wall' [4] passes of a design, thus reducing the number of chances for inconsistencies to creep in into the design flow. Furthermore they can also decrease the design time needed as none or very limited RT-level designing is needed by hand. However the limitation is that the final design created is most likely non-optimal. Taking the Catapult C tool as an example, the basic idea is that the designer takes a hardware system specification and creates a detailed design for it in C high-level language. Once ready, the Catapult C tool would directly map the C into synthesizable RTL and perform some limited optimizations thereafter; which can then be taken through the standard synthesis and place-and-route tools. The C behavioral model created by the designer is the design that is almost directly mapped into RTL by the tool. There may still be a need for several 'what-if' [2] design iterations to achieve timing convergence, and also further optimization.

An EDA tool currently under development by our group aims to significantly reduce the design time for digital circuits yet also simultaneously produce highly optimized designs for the situation in hand, by performing a comprehensive design space exploration at the algorithm level. This facility is currently not provided by any of the commercially available EDA tools.

Research into creating similar tools is being undertaken by other groups as well. One example of an automated tool is SPIRAL [12]. SPIRAL automatically generates platform-tuned software implementations of linear DSP transforms, where the aim is to find the fastest implementation of a transform for the targeted processor from a given high-level circuit specification. This tool uses the idea that a single DSP transform may be represented by a large number of different, but mathematically equivalent, formulae. It is observed that when these formulae are implemented in code, their runtime differ significantly. Thus SPIRAL uses a Signal Processing Language (SPL) to input a formula representing the desired transform, which is translated to C code using an SPL compiler. This formula is modified by applying different breakdown and manipulation rules, and the code generated using the SPL compiler. This process can be repeated to create more circuits and evaluated for the fastest.

Many other related works include ongoing research by several groups on Genetic Algorithm (GA)-based exploration tools. These tools differ from each other by having novel algorithm implementations, or unique chromosome encoding methods, etc. One very recent example of a similar tool to ours is of Krishnan and Katkoori [13], who have created a tool for generating several structural RTL netlists for a given linear DSP transform by performing design space exploration. However our works differ in how we structure our GAs; for example our implementation uses Pareto-analysis and a novel niching algorithm to avoid the hill-climbing problem.

This paper gives an introduction to our prototype tool and presents some of its experimental applications and outcomes. The paper aims to highlight the potential of the tool as well as discuss the future direction.

2. PROPOSED EDA TOOL CAPABILITIES

As [5] acknowledge, any modifications made at the behavioral level of a design are much more effective and easier than at RT-level. The goal of developing this tool is to take advantage of this concept and produce high quality digital hardware designs by performing a comprehensive design space exploration at the algorithm level. Furthermore, the tool also aims to drastically reduce the required design time by almost completely eliminating the need to perform any detailed designing of the system at all. The tool aims to produce highly optimized synthesizable circuits from simply reading in a specification for the desired system presented

by some formal specification method. The specification may be specified in a high-level language, such as C, or perhaps by even using some mathematical equations accompanied by some constraints which could fully and accurately specify the required design. At the moment, the high-level behavioral specification is presented in a mathematical form to the tool. A matrix is used to specify the transform to be implemented and some numerical values are also provided to specify, for example, the number of input and output ports of the circuit with their respective bit-widths. Using a mathematical form of specification ensures that it is treated purely as a specification and not as the baseline design for the digital system, which could inherently limit the possible optimizations for the circuit.

Our proposed tool performs circuit creation and circuit optimization simultaneously during the evolution process. Thus any design decisions taken during circuit construction also consider the effects it may have on the circuit's optimization.

It is almost always difficult to select only one implementation of a given circuit and identify it as optimal, since depending upon the requirements of the situation in hand, a different implementation may be considered to be better. For example, a serial, low area FFT may be good for one situation while a high speed parallel FFT better suited for another. Due to the often conflicting design parameters, optimizing for one parameter can make the other parameter(s) worse. Hence, being able to multi-objectively evaluate across several dimensions and preserve all the functionally accurate designs that give varying amounts of optimizations for each dimension can be very useful. Currently the tool aims to optimize across three different dimensions, namely the quantization error, area and longest-path delay. However, additional design dimensions, such as power and testability can also be integrated if suitable characterization metrics and evaluation algorithms are devised. Integrating with other EDA tools for power analysis, and particularly pre-synthesis floor-planning to allow for accurate estimation of delays, is also feasible as the prototype tool can seamlessly integrate with existing commercial tools of any design flow.

3. EDA TOOL IMPLEMENTATION

The prototype tool has been developed by members of the research group and its various algorithms are described in depth in [6], [7] and [8]. A brief overview of the tool's implementation is given here. The tool is based on novel multi-objective evolutionary algorithms along with some complex heuristics. It utilizes a powerful ($\mu + \lambda$) multi-member evolutionary strategy. The tool uses a search strategy which offers a compromise between the search area coverage (hence the time the tool takes to run) and the number of unique solutions found. A high level view of how the algorithm behaves is shown in Figure 1.

The tool starts by creating an initial predetermined size pool of very simple designs where every output of the desired circuit is connected to randomly chosen input. This initial population will now pass through various mutation and evaluation phases to generate circuits satisfying the input circuit specification. Each individual design's information is encoded by a directed graph. The entire population is multi-objectively evaluated by performing a non-dominated Pareto analysis; Pareto analysis allows measure of fitness that is not biased towards any particular optima [11]. The Pareto analysis sorts the individual designs according to their

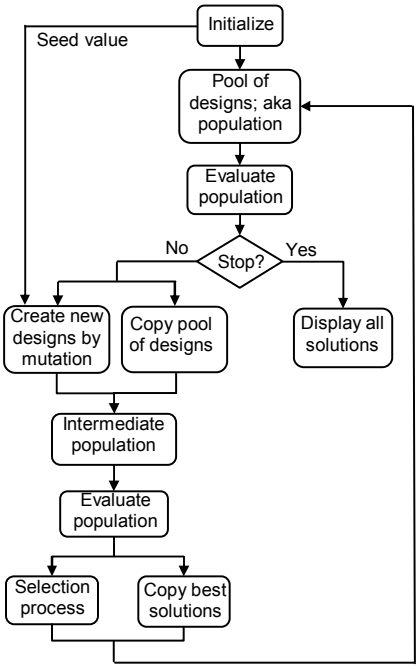

Fig. 1. High level flow chart of the SDA algorithm.

fitness, where fitness is basically measured in terms of the design's performance in all chosen objective functions. The tool currently uses the error in the circuit's response, silicon area and longest-path delay objectives for the fitness calculations. The functional correctness is currently being characterized by means of calculating the n-dimensional Euclidean distance between the evolved circuits behavior against the desired mathematical description of the circuit, where n is the number of inputs of the circuit. The evolved system's response is currently being obtained by calculating its impulse response.

However this verification method is time-consuming to perform with respect to the time used for actual evolution of the designs, thus the next project goal would be to transform the input specification along with the evolved circuits into some canonical representation and perform an equivalence check of the two to ensure that the behaviors of the two are converging.

New designs are formed by performing a mutation on some of the existing designs of the population. When mutating, depending upon the level of functional correctness, the tool invokes the appropriate heuristics algorithm and attempts to make changes which would be more probable to reduce the error significantly. As each design is being represented by a direct graph, there are only a handful of mutations which can take place. These include the addition of a new component, removal of an existing component, modification of a connection, or modification of a shift or negation associated with a wire or component respectively. These mutations are applied according to the embedded heuristics algorithms in the tool. Further mutations may be introduced at a later stage.

Evaluation is again performed on the newly evolved intermediate population. As the original population is also contained in the intermediate population, enough designs have to be discarded to ensure that the population size does not spiral out of control over thousands of generations. This process of discarding designs is critical to ensure that promising designs are not lost during the course of evolution, yet good diversity is

maintained in the population at all times to avoid the hill-climbing problem as much as possible. Thus, elitism has been implemented to ensure that a small number of the fittest individuals are preserved to guarantee that the following generations do not become inferior. The remaining designs of the population which are not elite go through a size-2 tournament selection process. The selection process is aided by an efficient niching algorithm which has been developed in an attempt to avoid premature convergence of the designs by introducing diversity in the population at a greater scale. It attempts to discriminate between the different designs on the Pareto-surface so to avoid convergence towards a single cluster of solutions near one area of the Pareto-surface. This process of evolving designs is repeated until a preset criterion is met for halting the search. The concept of seed value has been implemented so that the paths taken during the evolution process can be regenerated to obtain the same designs again for a given input specification.

Currently the area and longest-path delay values are obtained solely from the component information and the effects of wiring are being ignored which results in inaccurate estimates, as with Deep Submicron technologies, the effect of wiring has become significant [5]. However this aspect of the tool can be improved at a later stage independently from the actual evolving aspect of the tool. The main focus of the existing research has been to create a prototype tool which can comprehensively explore a design space at the algorithmic level for a given design specification.

4. EXAMPLE EXPERIMENTAL DESIGNS

Previous applications of the tool were limited to low complexity circuits, mainly simple FIR filters. However after modifying some heuristics algorithms and introducing new procedures to apply mutations at different probabilities according to the size of the input specification, the tool is tested with more complex circuits. In its current state, the tool is found to be capable of automatically generating dedicated datapath designs of low to medium complexity for linear DSP transforms. The control for these designs can be automatically generated using conventional techniques. A varied version of the tool is also under development [14] to handle non-linear DSP transforms; however the details of it are not covered in this paper.

The designs for 2D Discrete Cosine Transform (DCT) and a parallel 16-point complex Fast Fourier Transform (FFT) have been implemented. Both these transforms are compute-intensive and extensively used in many applications, particularly in the discipline of digital image and signal processing. The RTL netlist for a 2D integer-based DCT/IDCT module was automatically generated which is used as part of a H.264/MPEG4 video coding IP [9]. The tool created this in a matter of a few minutes with a simple switching mechanism implemented between the two. Several versions of circuitry were simultaneously created, each offering different trade-offs between the area and longest-path delay objectives. Given that the DCT/IDCT is integer-based, the specified behavior of the circuit was expected to be fully met with zero tolerance.

Figure 2 illustrates the range of solutions found by the prototype tool for the 4x4 DCT module, and as may be seen, only one of all these designs lay on the Pareto-surface. Similar varieties of designs were generated for the IDCT module as well. This resulted in one 'best' design in terms of both the area and the

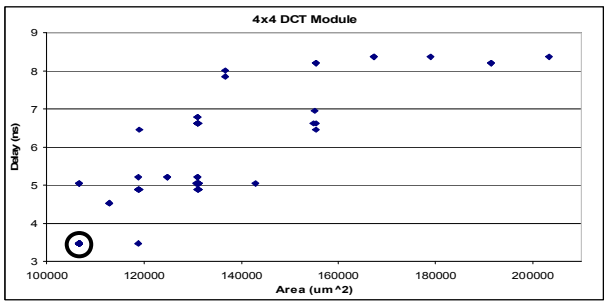

Fig. 2. All functionally accurate designs generated for the DCT module by the prototype tool

longest-path delay for the combined DCT/IDCT module. The design for the entire DCT/IDCT module was synthesized using BuildGates tool from Cadence targeting UMC 0.18µm CMOS technology library. Power results were obtained with Synopsys Design Power by monitoring the switching activities of the design through the gate-level simulation.

The area and delay results for the best DCT/IDCT design found through performing the time-limited design space exploration was next compared against a manually written and optimized DCT/IDCT module as shown in Table 1. The manual creation of the DCT/IDCT module involved deriving a detailed design and implementing it in Verilog HDL. A testbench was also developed to verify the module meets the specification. Upon performing the comparison, it was found that the time required for the creation of such a highly optimized design compared to the manual DCT/IDCT was drastically reduced while no engineering effort was needed to even think about the detailed design at all. Furthermore the creation of a testbench was not compulsory for the automated solution since the solution has already been verified by the tool to exactly meet the specification. The DCT/IDCT coefficients were simply fed into the tool in the form of a matrix, along with the required number of inputs, outputs and their respective widths, and the prototype tool created a range of designs automatically.

Table 1. Comparison of manual and automatically generated DCT/IDCT modules

	Area, μm^2	Delay, ns	Power, mW	Approx. effort
Manual solution	353004.68	3.08	14.976	6 working days
Automated solution	272279.72	3.48	8.741	40 minutes

The longest-path delays are similar, however, a significant improvement was observed in the area and power consumption of the automatically generated design compared to the manual one. The significant savings observed are due to the fact that the tool made selective use of its own custom IP for all the primitive operators and was able to choose and arrange these operators in such a manner that would feel unnatural for a human to derive manually. This example shows the tools ability to create a highly optimized design from its specification in the fraction of the time it took an engineer to create an optimized design for the 2D DCT/IDCT using conventional tools; hence resulting in significantly reduced manpower requirement and design time.

A parallel 16-point complex FFT was also created for a MC-CDMA receiver [10] using the prototype tool. It is a 32-bit FFT implemented in radix-4. Again, the entire design was automatically generated using the prototype tool. The tool was able to generate

several different solutions providing differing trade-offs between area and longest-path delay in a matter of a few hours. This has resulted in the possibility of having several different FFT implementations, all of which give varying amounts of overall area and delay trade-offs. All these different solutions are available in their RTL form to be used, if required. Say the engineer chooses one version of the FFT implementation, however, after going through the latter stages of the design flow, wishes to try out the 'what-if' scenario, it is simply a matter of selecting one of the other already implemented designs and taking them through the latter stages. No further time has to be spent on redesigning and re-verifying new RTL implementations, and all the other designs are guaranteed to give varying performances in terms of area and delay, while remaining functionally accurate.

Once again the designs were synthesized targeting UMC 0.18µm CMOS technology library and the power consumption estimated using the Synopsys Design Power tool. After letting the tool run for a few hours, a range of solutions were generated offering varying trade-offs between quantization error, area and delay. Next, an equivalent FFT was manually created, along with a testbench, to ensure that the initial specification is met. The performance of two automatically generated solutions, where one gave the best area results while the other gave the least delay, were compared against the manual solution; See Table 2. Note that the chosen automated designs give similar quantization errors as the manual solution.

Table 2. Comparison of manual and automatically generated 16-point, 32-bit complex FFT modules

	Area, μm^2	Delay, ns	Power, mW	Approx. effort
Manual solution	937388.86	8.39	191.311	15 working days
Automated solution (1)	377661.62	6.02	42.454	4 hours
Automated solution (2)	449333.96	4.22	48.520	

The prototype tool was able to offer these two extreme solutions, along with several others which lie in between these two extremes in a matter of a few hours. Running the tool for longer allows a larger design space to be explored and potentially generate designs with an even higher performance than those identified already. As can be seen from Table 2, the best automated solutions found far exceeded the performance of the manual solution, yet taking only a fraction of the time to create. The significant area decrease observed between the manual and automated solutions is a result of the prototype tool making use of primitive operators such as addition, subtraction, negation and bit-shifting, to generate the complex multipliers whereas the manual solution was created using a standard booth multiplier module to perform the multiplications. A several fold reduction was also observed in the power consumption of the manual and automated solutions. It is mention worthy again that the prototype tool does not currently use power as one of the multiple objectives which it tries to optimize, however this can be included if appropriate characterization metrics are built into the tool. The current power savings experienced are due to the novel low power primitive operators used by the tool, and also due to the reduction in the design areas which inherently lowers the power consumption of a design. It should also be noted that the same synthesis constraints were specified for the synthesis of the automated and manual solutions.

5. CONCLUSIONS

The paper established the need for new tools to cope with the challenges facing today's engineers. The tool under development which exploits the benefits of performing design space exploration at the algorithm level could have a substantial impact on improving the performance of systems while significantly reducing the NRE costs and design-time simultaneously. Efficient DCT and FFT implementations have been created using the prototype tool with their performance comparable to manually created solutions, showing the steady progress towards achieving the proposed aims.

6. REFERENCES

[1] *International Technology Roadmap for Semiconductors 2001*

[2] McCloud, S. Mentor Graphics, *Catapult C Synthesis Based Design Flow: Speeding Implementation and Increasing Flexibility* October 2003

[3] Ownby, M. and Mahmoud, W. H., *A Design Methodology for Implementing DSP with Xilinx System Generator for Matlab*, Proceedings of the 35th Southeastern Symposium on System Theory 2003, pp. 404-408

[4] Synopsys white paper, *Achieving DFT Closure – The Next Step in Design for Test* October 2001

[5] Wakabayashi, K. and Okamoto, T., *C-Based SoC Design Flow and EDA Tools: An ASIC and System Vendor Perspective*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 19, No. 12, December 2000

[6] Thomson, R. and Arslan, T., *Evolvable Hardware for the Generation of Sequential Filter Circuits,* NASA/DoD Conference on Evolvable Hardware 2002 Proceedings

[7] Thomson, R. and Arslan, T. *An Evolutionary Algorithm for the Multi-objective Optimisation of VLSI Primitive Operator Filters* Congress on Evolutionary Computation 2002

[8] Thomson, R. and Arslan, T. *On the impact of Modelling, Robustness, and Diversity to the performance of a Multi-Objective Evolutionary Algorithm for Digital VLSI System Design* The Congress on Evolutionary Computation, 2003

[9] Kneip, J., Schmale B. and Moller H. *Applying and Implementing the MPEG-4 Multimedia Standard* Micro IEEE, Vol. 19, Issue 6, Nov 1999, pp. 64-74

[10] Hasan, M., Arslan, T. and Thonpson, J. *A Novel Low Power Pipelined Architecture for a MC-CDMA Receiver* Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis ISPA 2003, pp. 1048-1053

[11] Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning* Addison-Wesley 1989

[12] Puschel, M., Moura, J.M.F., Johnson, J.R., Padua, D., et al, *SPIRAL: Code Generation for DSP Transforms* IEEE Proceedings Vol. 93, No.2, pp. 232-275, Feb 2005

[13] Krishnan, V., Katkoori, S., A Genetic Algorithm for the Design Space Exploration of Datapaths During High-Level Synthesis IEEE Transactions of on Evolutionary Computation, Vol. 10, No. 3, pp. 213-229, Jun 2006

[14] Thomson, R., Arslan, T. *The Evolutionary Design and Synthesis of Non-Linear Digital VLSI Systems* NASA/Dod Conference on Evolvable Hardware Proceedings 2003