Proceedings Of The IEE Colloquium On Digital Synthesis, London, UK, pp.7/1-7/5, 15th Feb 1996. -

Genetic Synthesis Techniques for Low-Power Digital Signal Processing Circuits

T. Arslan, E Ozdemir, M. S. Bright, and D.H. Horrocks

University of Wales Cardiff
Cardiff CF2 1XH

Introduction

Power dissipation is becoming a limiting factor in the realisation of VLSI systems. The principal reasons for this are maximum operating temperature and battery life for portable applications. Because of the relatively greater complexity, the power dissipation in Digital Signal Processing (DSP) applications is of special significance, and low-power design techniques are now emerging [1]. These vary depending on the level of the design that they target, ranging from the semiconductor technology to the higher algorithmic level.

This paper will describe two different techniques which employ Genetic Algorithms for synthesis of digital circuits. The first employs a multi-objective genetic algorithm for synthesising low power circuits. A gate-level description of the circuit is encoded into a single chromosome and the GA evolves by searching for circuit structures that are optimised for both overall capacitive area and critical path. This in turn will allow operation under reduced power supply voltages [1]. Although, the technique currently operates at gate-level its main advantages become apparent when used within a high-level framework. Our results are illustrated with examples at the gate-level by using the adder and parity checker problems [2]

The second technique operates on a Data Flow Graph (DFG) consisting of high level blocks such as registers, adders, and multipliers. Operations such as retiming and automatic pipelining are used to reduce the critical path of the DFG, hence allowing operation under reduced supply voltage. Our results are illustrated using an 8th order Avenhaus filter with achievement of more than 50% power saving.

Structural Synthesis Technique

The GA is implemented specifically to suit the structural design synthesis problem. For example, the chromosome representation in figure 1 is designed to incorporate individual library cells and their attributes (e.g., inputs and outputs) in a manner which can incorporate circuits of any size.

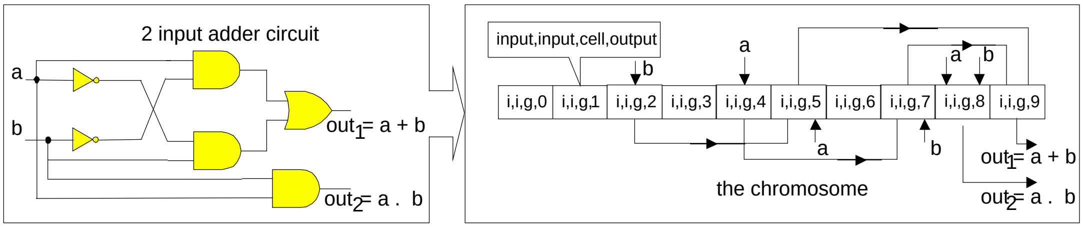

Fig.1 *Structural Design Chromosome*

The representation used is compatible with the *extracted* circuit representation used by the majority of CAD tools (such as those developed by Cadence, Mentor, Plessy, and ES2). This provides the added flexibility of integration with such tools, hence providing a valuable tool to the VLSI circuit designer who is facing increasingly challenging tasks such as designing for low-power, area, speed, etc. In addition, the genetic operators [2] are specifically designed to take the circuit aspects of the structural design into consideration. Our representation and the specifically developed genetic operators are crucial to the flexibility of our genetic algorithm and its superiority over previous implementations.

The GA utilises a given design cell library in the structural design of a multi-input/multi-output logic function. This is achieved through exercising complete freedom with the design cells and using a multi-constrained fitness function that aims to optimise hardware aspects such as speed and area, in addition to that of achieving the correct logic function(s). The GA continuously references one or more design cell libraries throughout its genetic evolution as the chromosomes are processed for fitness calculation. In our case only delay and area parameters are required. The use of the design library within a multi-constrained fitness frame work is illustrated in figure 2.

The *target logic function*, which is to be structurally synthesised, is processed by the fitness evaluation section in order to obtain an optimum set of input/output patterns which can test the functionality of each prospective circuit structure, i.e. a chromosome. A special circuit construction function is used to encode the chromosome into a form that can be analysed for aspects such as connectivity and redundancy. The input/output patterns are then used to evaluate the functionality of the constructed circuit through a sequence of operations which involve propagating logic values through different paths in the circuit and backtracking when necessary. This is especially needed for the cases of multi-output circuits and/or where the circuit includes some unused redundancy. The *functional fitness* section will award a score depending on the degree to which the target logic function is satisfied. A score is also awarded for the delay by tracing the gates in the critical path of the circuit. In addition, a score is awarded for the physical size of the circuit, the *area fitness*, by summing the number of gates and scanning their corresponding cell area from the library. Finally, the functional, delay, and area fitness scores are processed through a weighting function, which considers their relative significance and produces a single global score representing the final fitness. Currently, this is a weighted sum of all the sub-fitness'. The above fitness frame work can accommodate additional sub-fitness functions for considering other aspects in a given circuit.

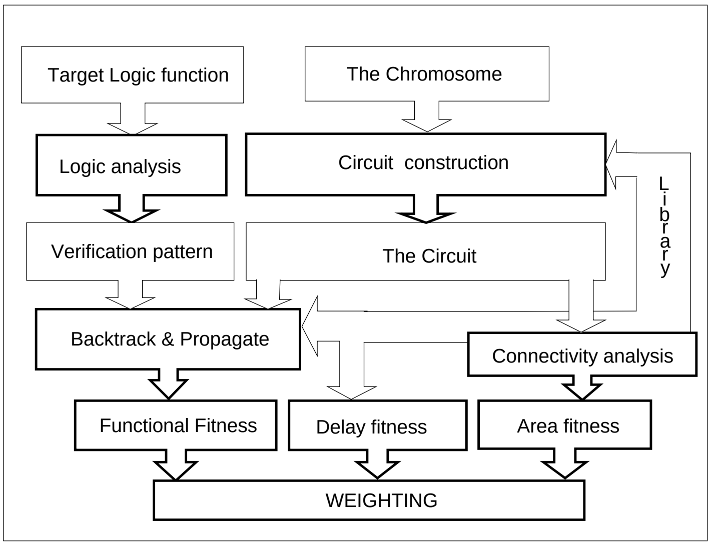

Fig.2 *Multi-objective fitness evaluation*

High-level Technique

This work concentrates on the implementation of techniques to reduce power dissipation at the architectural level. The systems are represented as high level Data Flow Graphs (DFG), they are application specific algorithms implemented in 2μm CMOS technology. A variety of DFG speedup techniques are used to alter the systems' power requirements. A Genetic Algorithm is developed to obtain the optimum power solution from within the large search space.
The main sources of power dissipation are shown in the following equation [1].

$$P_{average} = P_{switching} + P_{short\text{-}circuit} + P_{leakage} \qquad (1)$$

The short-circuit and leakage terms can be reduced to negligible values through the application of appropriate design techniques at other levels of the design [4]. The switching component is the power required to charge/discharge the load capacitor of the CMOS device. The average power dissipation is therefore the average power required to perform all the switching events across the DSP. For a circuit represented as a DFG this can be expressed as follows[5].

$$P_{average} = V_{DD}^{2} * C^{*} * f_{sampling} \qquad (2)$$

C* is the effective capacitance switched within the circuit (switched capacitance). It is a combination of the physical capacitance and switching activity. The switching activity can be difficult to quantify as it depends on many factors such as circuit topology, logic family, etc. [3]. These factors are not finalised at the high level, making high level power estimation a difficult process.

$f_{sampling}$ is the sampling frequency of the DFG. The aim of power minimisation is to reduce power while keeping the systems throughput constant, so $f_{sampling}$ is fixed. Therefore the power dissipation can be decreased by minimising the switched capacitance within the system, reducing the supply voltage or applying a combination of both techniques.

As shown in equation(2) reducing V_{DD} will result in a quadratic decrease in power. Unfortunately this power decrease has a penalty. Previous research has shown that reducing V_{DD} results in an increase in the delay of the system [3]. The relationship between V_{DD} and delay is shown in figure 3.

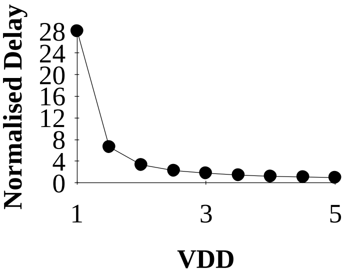

Fig.3 *Relationship between V_{DD} and normalised delay [3]. 2μm CMOS technology.*

If the systems throughput can be increased then V_{DD} can be decreased to a point where the induced delay returns throughput to its original value, thus reducing power while keeping throughput constant. This is known as control step reduction to reduce supply voltage [5].

The speed of the DFG is a changed by applying techniques that alter the longest (critical) path while maintaining functionality. These techniques may also affect the switched capacitance (C*), which will affect the power reduction obtained through V_{DD} reduction. The lowest power solution will be a trade off between speed and capacitance.

Retiming is the process of moving delays around a DFG to minimise the critical path. [6]. Simple retiming is the process of moving delays from all of the inputs of an element and placing them on all of its inputs, or moving them from its outputs to its inputs.

Automatic Pipelining is a specialised from of retiming. Unlimited delay elements are assumed on the DFG inputs, these are then retimed through the DFG to minimise the critical path [6].

Pipelining is performed by inserting delay elements at specific points in the DFG [7]. This can reduce the critical path but increases the latency of the system.

These are applied to the DFG in an order determined by the search algorithm. The order of application is important as it can affect the power reduction achieved. For example, a speed alteration that temporarily increases power may enable the application of another that will achieve a lower power solution.

The techniques discussed above are those that have been implemented in this work to date.

Implementation

A genetic algorithm (GA) [8] is used to determine the optimum power solution for a specific DFG. The power minimisation problem is very complex, even with a small number of speed altering techniques no time efficient algorithm can be developed to obtain the optimum solution [5]. A GA has the ability to escape local maxima, a quality required in this problem as it may be necessary to temporarily increase the power to arrive at an optimum solution.

A chromosome within the population represents an entire DFG. The elements in the DFG are individual genes, the chromosome is shown in figure 4.

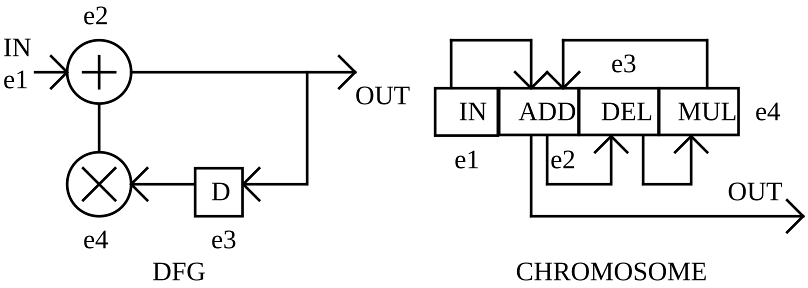

Fig.4 *Chromosome Structure*

The procedure uses a modified standard GA algorithm [8]. Normal bit place mutation and random point crossover cannot be used as they will alter the functionality of the DFG. The genetic mutation operators of a GA are replaced with the speed altering techniques as they operate on the genes of the chromosome, altering the chromosomes' fitness. Crossover is an operator that combines the characteristics of two chromosomes. In this case it can be implemented by identifying how two different DFG's have been mutated to create a new gene with both mutations.
The algorithm flowchart is shown in figure 5. The initial population consists of randomly mutated versions of the original specified DFG. The fitness is calculated by estimating the power of the chromosome, a lower power consumption produces a higher fitness. Chromosomes are selected for mutation using the Roulette Wheel method[8]. This breeds a new population that will consist of new DFG's with new characteristics. This process is repeated to achieve an optimum power solution, that DFG with the greatest fitness over all generations.

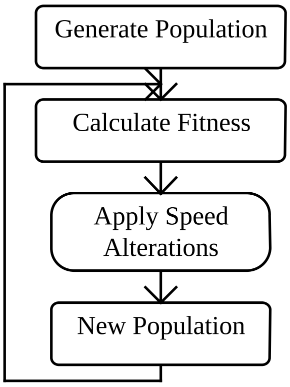

Fig.5 *Flowchart of Genetic Algorithm Optimisation Procedure*

Results

Figure 6 illustrates results obatined with the structural design technique. The graph shows a steady improvement in the circuit solutions evolved by the GA. The GA typically produces a circuit that satisfies all constraints within 60-65 generations, which is a relatively small number of generations considering the large solution space.

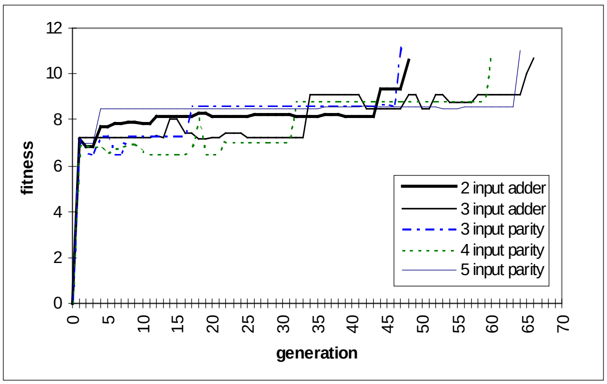

Fig.6 *GA Performance With Example Circuits*

Figure 7. shows the power consumption obtained over time for an 8th order Avenhaus filter (parallel form) [5], expressed as a percentage of its original power consumption. The optimum power solution was typically found within 150 generations. Other circuits have been investigated and have produced comparably favourable results, these include 3rd order FIR filters, Lattice filters and the 8th order Avenhaus filter (direct form).

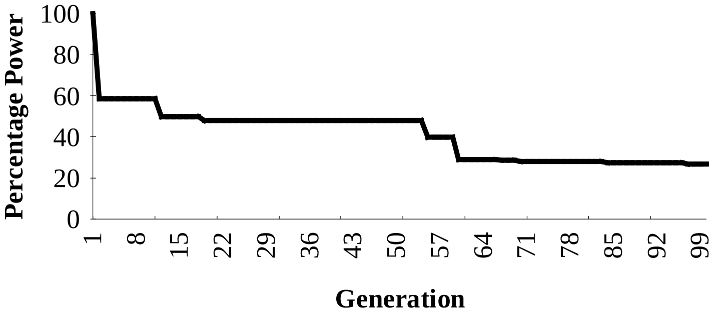

Fig.7 *Normalised Power Consumption Of 8th Order Avenhaus Filter Using High-Level Technique*

Conclusion

The results obtained using the genetic algorithm demonstrate that it is capable of optimising circuits at the high-level and gate-level, under a number of constraints such as speed, area, etc. It provides a framework for the VLSI designer to investigate the effects of different design parameters.

References

[1] Arslan T., Horrocks D.H., Erdogan A.T., "Overview And Design Directions For Low-Power Circuits And Architectures For Digital Signal Processing", *Proceedings Of The IEE Colloquium On Low Power Analogue And Digital VLSI : ASICs, Techniques And Applications*, London UK, pp.6/1-6/5, 2 June 1995.

[2] Louis S.J., Rawlins G.J., "Designer Genetic Algorithms: Genetic Algorithms In Structure Design", Proceedings Of The Fourth International Conference On Genetic Algorithms, San Diego, USA, July 1991, pp. 53-60.

[3] Chandrakasan A.P., Sheng S., Broderson R.W., "Low-Power CMOS Digital Design", *IEEE Journal Of Solid State Circuits*, Vol. 27, No. 4, pp. 473-484, April 1992.

[4] Chandrakasan A.P., Broderson R.W., "Minimising Power Consumption In Digital CMOS Circuits", *Proceedings Of The IEEE*, Vol. 83, No. 4, pp. 498-523, April 1995.

[5] Chandrakasan A.P., Potkonjak M., Mehra R., Rabaey J., Broderson R.W., "Optimising Power Using Transformations", *IEEE Transactions On Computer-Aided Design Of Integrated Circuits And Systems*, Vol. 14, No. 4, pp. 12-31.

[6] Lucke L.E., Parhi K.K., "Data-Flow Transformations For Critical Path Time Reduction In High-Level DSP Synthesis", *IEEE Transactions On Computer-Aided Design*, Vol. 12, No. 7, pp 1063-1068, July 1993.

[7] Parhi K.K., "High-Level Algorithm And Architecture Transformations For DSP Synthesis", *Journal Of VLSI Signal Processing*, 9, pp. 121-143, 1995.

[8] Goldberg D.E., "Genetic Algorithms In Search, Optimisation And Machine Learning", Addison Wesley, Reading, 1989.