Simultaneous optimisation of dynamic power, area and delay in behavioural synthesis

A.C. Williams, A.D. Brown and M. Zwolinski

Department of Electronics and Computer Science
University of Southampton
Hampshire SO17 1BJ
U.K.

Abstract

Concern over power dissipation coupled with the continuing rise in system size and complexity means that there is a growing need for high-level design tools capable of automatically optimising systems to take into account power dissipation, in addition to the more conventional metrics of area, delay and testability. Current methods for reducing power consumption tend to be ad-hoc: for example, slowing down, or turning off idle parts of the system, or a controlled reduction in power supply.

The behavioural synthesis system described in this paper features an integrated incremental power estimation capability, which makes use of activity profiles, generated automatically through simulation of a design on any standard VHDL simulator; accurate circuit-level cell models (generated, again automatically, via Spice simulation); and a comprehensive system power model. This data, along with similar estimators for area and delay, guides the optimisation of a design towards independent, user-specified objectives for final area, delay, clock speed, and energy consumption. In addition, a range of power reducing features are included encompassing: supply voltage scaling, clock gating, input latching, input gating, low-power cells, and pipelined and multicycle units. These are automatically exploited during optimisation as part of the area/delay/power dissipation trade-off process.

The resulting system is capable of reducing the estimated energy consumption of several benchmark designs by factors of between 3.5 and 7.0 times. Furthermore, the design exploration capability enables a range of alternative structural implementations to be generated from a single behavioural description, with differing area/delay/power trade-offs.

1 Introduction

The trend to produce ever more powerful portable electronic equipment has made factors such as battery life and heat dissipation of primary importance in the design of many integrated circuits. As device densities and clock frequencies increase, design with consideration for power consumption is becoming as important as the more traditional area, delay and testability concerns.

Today, digital design, almost without exception, uses synthesis tools at the register-transfer level (RTL), which require a designer to explicitly incorporate power-reducing features within the system design. High-level (behavioural) synthesis tools, however, such as the MOODS (*M*ultiple *O*bjective *O*ptimisation in control and *D*atapath *S*ynthesis) system developed at Southampton [1-11], are designed to *automatically* explore alternative structural implementations from a single design description, and thus is able to include power-reduction.

This paper describes the inclusion of a power-optimisation and reduction capability into the MOODS system, allowing it to simultaneously optimise a design for area, delay and power consumption, automatically making trade-offs between all three criteria. The system uses an integrated incremental power-estimator, capable of estimating the energy consumed by any particular structural implementation of the design. This produces figures which are used, in conjunction with area and delay estimates, to guide optimisation towards the designers objectives, specified in terms of target energy consumption, area, delay, and clock speed.

1.1 Existing System

MOODS provides the ability to automatically translate a design from a behavioural to a structural VHDL description, optimising the structure as it does so. The synthesis process involves the application of small, behaviour-preserving transforms (from a set of eighteen) to an internal representation of the design's behaviour, describing the flow of control and data through the system (ie. control and data path graphs). Transforms are selected, and applied to modify particular parts of these graphs under the control of simulated annealing[1,12] or heuristic[3] optimisation algorithms. These are guided by a global cost function, which evaluates the current configuration with respect to user-specified objectives on area, delay, and now energy consumption. Throughout this process, all elements of the design (functional/arithmetic data path units and states in the controller) are bound to physical units from a technology-

specific cell library. At any point during optimisation, the system holds a complete structural implementation of the design, and is able to use accurate characterisation data from the library to estimate cost.

Transforms concerning area, delay, floating-point accuracy, and testability are described elsewhere [1-11]. The contribution of the work described in this paper is to enable MOODS to optimise with respect to another dimension, that of overall power dissipation. Broadly, the transforms relating to power dissipation encompass:

- Supply voltage scaling - reduces power consumption through a reduction in supply voltage, exploiting (where possible) increased concurrency to offset any associated increase in delay.
- Clock gating - reduces the energy wasted by the clocking of storage units on every cycle, by only enabling the clock when a load is required. This also serves to "turn off" parts of the design by holding the clock steady during inactive periods.
- Input latching - holds inputs steady on high-power functional blocks (such as multipliers) in order to reduce spurious activity. In addition, this eliminates unnecessary activity on unused input bits when units of differing bit-width are shared.
- Input gating - an alternative to input latching which disconnects inputs during inactive periods.
- Low power cells - lower power (and slower) alternative implementations of basic arithmetic/functional units suitable for use in non-performance critical parts of the design, or where an increase in parallelism can accommodate the reduction in performance.
- Pipelined and multi-cycle units - provide increased capacity for reducing power and adding concurrency to accommodate delay degradation introduced by other power-reducing features.
- The exploitation of an *hierarchical module expansion* capability also enables inter-module optimisation, and allows units to be *dynamically* characterised during synthesis. This capability is explained in detail elsewhere[10].

1.2 Related Work

Behavioural synthesis for low-power has become increasingly important as low-power applications have become mainstream [13], concentrating on two key areas: power estimation, and low-power synthesis.

1.2.1 Power Estimation

Calculating a system's power consumption is a much more involved process than determining its area, or delay, as power is almost entirely dependent upon switching activity (for CMOS technologies), and is thus heavily tied to the data flowing through the system [14]. Estimation techniques fall into two broad categories [15]: simulation-based methods [16-17], which estimate the power consumed when a particular data set is applied to the system; and probabilistic methods [18], where an attempt is made to determine "typical" power consumption figures based on probabilistic circuit models and input activity data.

Both techniques have their advantages and disadvantages. Circuit simulation is heavily pattern-dependent (but then so is power-consumption) and can be slow when performed at the transistor level, but it is also the most accurate method. Probabilistic methods tend to be faster, but this generally results in reduced accuracy. In addition, they often require the user to provide input data in a non-standard form (such as activity probability distributions), rather than exploiting existing testbenches. Both techniques can be combined whereby simulation is used to generate signal activity data, which is then used in probabilistic models to estimate system power [19-22].

Power estimation is a key component of many power optimising behavioural synthesis systems as the tools need to be capable of determining whether they have achieved the desired goal. A further complication is the speed of estimation, which must be fast enough to allow the exploration of many alternative design implementations - in other words, an estimator that takes several hours (or minutes) to execute, will not be suitable in a system requiring hundreds (or thousands) of design iterations[20-22]. These systems attempt to model the power consumption of design units (such as adders or multipliers), which are combined in the power estimator to build up a complete system-level power figure. Factors such as data distribution, signal correlation and glitching activity are addressed [21], but most systems are either too slow to be useful, or are specifically targeted at dataflow dominated designs

(such as digital filters), where there is little control logic and few conditional sections to consider.

The system described in [22], *has* focused on more general "control-flow intensive" designs, and has been incorporated into a behavioural synthesis system which iteratively generates alternative optimised implementations, and performs a full power estimation on each to determine whether the designer's power consumption objective has been reached. The system requires around 15 minutes to generate a switching-activity matrix - a one-off operation - along with around 30 seconds estimation time *per iteration* (in contrast to MOODS, here, an iteration represents an *entire* synthesis run, each generating a *complete* implementation).

1.2.2 Low-Power Synthesis

Power-optimisation in behavioural synthesis, like power estimation, has tended to concentrate on dataflow dominated designs. This simplifies the estimation and synthesis processes somewhat, as less attention needs to be paid to arbitrary control flow, and the key system delay metric becomes a single data throughput rate. Most of the systems developed optimise a particular aspect of the power equation - typically signal transition activity and/or switched capacitance [23-29].

[28] generates activity data based on a simulation of a dataflow graph (ie. a behavioural simulation), and performs area and clock-period constrained synthesis while optimising either the switching activity, or the switched capacitance. Synthesis is based on an heuristic algorithm that makes trade-offs targeted at minimising activity or capacitance.

[27] makes use of a combination of heuristic and simulated annealing algorithms to transform a dataflow graph in order to reduce activity and capacitance. This exploits techniques such as re-ordering, minimising data word lengths, and selecting alternative operations. The system targets a restricted architecture and is designed purely for dataflow intensive applications, enabling a simplified power model to be used.

[22] uses a genetic algorithm to perform synthesis, and includes support for multiple supply voltages by enabling the selection of low-voltage units, again in a dataflow dominated design. A number of additional techniques are also described to reduce power consumption, such as supply voltage scaling [30], and clock gating [31] which turns off inactive parts of the system.

The system described here exploits many of the above features, integrated within a multi-dimensional synthesis system that imposes *no* restriction on either the system architecture (ie. it supports the full spectrum of designs from control to dataflow dominated), or the optimisation targets specified (ie. it can minimise and constrain *any* combination of area, delay, clock period and power consumption, and automatically make trade-offs between them). The system includes features such as clock gating, low-power modules and supply voltage scaling at a fundamental level *within* the synthesis loop enabling their automatic inclusion in any synthesised design. Power calculations are dynamically performed by a fast incremental power estimator exploiting low-level power models, and automatically generated activity data, to enable the system to compare alternative design implementations during synthesis.

2 Architecture of the Low-Power Synthesis System

The features integrated into the core synthesis tool to seamlessly include power/energy consumption in the set of objectives specified by the user fall into four groups:

1. Power/energy estimation - MOODS must be able to estimate the energy consumption of any control and data path configuration in order determine the "best".
2. Optimisation - using this information, the system must be able to determine which transforms should be applied to further improve the cost figure.
3. Low-power enhancements - as well as power optimisation, additional power-reducing architectural features can be used. These include clock gating, input gating and latching, low-power library cells, pipelined and multi-cycle (serialised) cells.
4. Library expansion - enhancement of the system’s cell libraries to include low-power variants of core operations provides a further level of trade-offs to be made, typically between delay and power consumption.

Figure 1 shows the block structure and design flow of the power-enabled synthesis system. Shaded blocks indicate items relating to power optimisation and modelling, added or modified for low-power synthesis. The following sections describe each of these components in detail.

2.1 Activity and Energy Estimation

The ability to estimate a system's energy consumption is fundamental to the inclusion of power optimisation in MOODS. Power consumption in CMOS circuits is notoriously difficult to estimate [14-22], as the majority of the power is consumed during signal transitions, with minimal static dissipation[14]. For most systems, then, power consumption is highly data-dependent, and some means of determining typical system transition activity is required. As most of the power goes into driving load capacitance (transistor gate and routing), knowledge of the system's connectivity is also essential to make any meaningful estimate.

The approach taken in MOODS uses two new features to provide activity information, and energy estimation. Instead of attempting to determine a "typical" power consumption figure for the design, MOODS requires the user to provide a testbench which is simulated to generate activity information for all operations in the design. Power optimisation then becomes the process of optimising the energy consumed during execution of the testbench. This approach allows the user to provide the system with activity information in the most practical form, as a comprehensive test suite will almost certainly be created for most designs. MOODS obtains activity information from the testbench via a simulation of the system using any standard VHDL simulator. This is achieved by instrumenting the output of an early optimisation run so that the synthesised block automatically monitors its own activity, and creates the necessary data files for future optimisation. Once a set of activity data has been generated, the operation need only be repeated if the behavioural design changes, and not on each synthesis run.

This approach provides a number of advantages:

- there is no requirement to provide any special testbench, or input activity figures - all data is derived from the VHDL simulation.
- the design may be simulated as a component of a much larger system and the activity data will automatically be created.
- the instrumented output is cycle-accurate (ie. the design has been scheduled). The input to a high-level synthesis system often contains areas with unspecified delays, and can make use of low-level modules to provide timing-critical operations, such as memory interfacing. Interacting with such designs is often

more sensible after synthesis when the complete cycle-by-cycle timing has been defined.

The activity data generated from the above simulation provides MOODS with information on the frequency of operation execution and input signal transitions, with the effects of transforms on this activity data being automatically determined during optimisation.

2.1.1 Energy model

Figure 2 shows a simple fragment of a data path where an add operation is implemented by a single physical adder, with its inputs driven by two registers, **a** and **b**, loaded during operations $\mathbf{op_1}$ and $\mathbf{op_2}$ respectively, and having activation counts ACT_{op1} and ACT_{op2} (assumed to occur independently). The output register **c** has input capacitance C_{load}. The equation used to model the energy consumption of the adder is shown in the figure; the key data required is:

- Operation activity - describes how often a particular behavioural operation is executed. The key factor here is that the activity of any given data path unit is not necessarily a function of how often the implemented operation executes, but how often the inputs change - in fact, the add is only really executed when its output register loads a result. Two further complications: 1) if a unit is multiplexed (unit sharing) between two or more operations, the inputs become partially decoupled, and the multiplexor can be exploited to perform input gating - ie. breaking the path between its input and output; 2) operators can be chained together during optimisation by bypassing any registers between them: the adder may be driven directly from a multiplier, which may also be driven by any number of earlier units. The adder input activity is therefore dependent upon the activity of all units further up the chain.

- Cell energy model - the energy consumed by any unit in the data path is a function of several factors, principally (for CMOS): static power consumption, short circuit power, and switching power [14]. These are characterised for each cell available in the low-level library from a circuit-level simulation using a three-stage model to account for changes to each input, control signal (loads, selects) and clock.

- Load capacitance - of all the contributors to power consumption by far the most significant is capacitive switching power, typically accounting for at least 90% of

the total [14]. This is a function of both the internal capacitance of the cell, and the capacitance of the inputs driven - in the example, a single register. Again, the cell library characterises the input capacitance of each cell input, which may be a function of several parameters, including the cell load capacitance (eg. fixed width shift, or bit reversal functions). Calculation of the load driven by a data path unit must, proceed in the correct order, ie. bottom up.

2.1.2 Incremental Estimation

The key to maintaining a reasonable execution time for MOODS is the incremental estimation capabilities. As each transformation is actually quite simple, (for example share two operations on a single unit by multiplexing the inputs) a full energy estimation of the entire design is only really necessary at the start of optimisation. From then on, the system only calculates the change in the energy consumption due to each transformation. Although this still necessitates a significant amount of calculation, it is only a fraction of that required for a complete system estimation: for example, in the elliptical filter benchmark discussed in section 4.1, estimating the energy change associated with a transformation takes, on average 10ms. With a typical optimisation run performing around 5000 transformations, this amounts to a total time overhead of around 50 seconds and compares favourably with other synthesis-based estimation algorithms, such as [22] which quotes around 30 seconds *per iteration/implementation* for some benchmark designs.

2.1.3 Accuracy

Three issues relate to the accuracy of the estimator figures: cell models, system model and transformations:

- The cell models are developed through circuit simulation of the low-level library cells, and feature a 3-stage model described later. Results obtained are all within 10% of the circuit-level simulation figures, for bit-widths ranging from 1 to 128 bits, depending upon the cell type.
- The system-level model combines the cell data with activity and interconnect information (as discussed above) to estimate the energy consumption when executing a testbench. Because this problem is tightly defined - we do not attempt to determine "typical" data/control flow - a reasonable accuracy of within 20% of the simulated figure can be maintained for small designs.

- From a synthesis standpoint, the *absolute* accuracy of the estimates is largely unimportant when compared to their *relative* accuracy. Because of the nature of the simulated annealing algorithm, the estimator need only be capable of determining whether a transformation increases or decreases the energy consumption. Thus, estimator accuracy is not of importance - speed of estimation takes priority.

2.2 Power-Reducing Enhancements

Energy estimation provides the core power measure for the MOODS cost function, but there are also a number of fundamental limitations in the synthesised architecture that result in energy waste - ie. energy used to perform unnecessary operations. These are addressed through the inclusion of several new transforms, modifications to the system architecture, and enhancements to the cell library mechanism:

1. ***Supply-voltage scaling:*** Power consumption is proportional to the square of the power supply voltage (V_{dd}), thus reducing V_{dd} can have a significant effect on energy consumption [30]. However, as gate delay is proportional to $V_{dd}/(V_{dd} - V_t)^2$ (where V_t is the technology threshold voltage), this also results in a degradation in speed. The inclusion of V_{dd} scaling as a transformation allows MOODS to trade off these factors, accommodating the increased delay through an increase in parallelism and the use or faster units.

2. ***Clock gating:*** In a typical digital circuit, the system clock directly drives all synchronous units (mainly registers or RAM in MOODS), with loading controlled by a separate, synchronous load enable. This results in the consumption of energy on every cycle, even when the register does not load a new value. By including clock gating circuitry to only enable the clock on a register load, this wasted energy can be reduced [31]. Additionally, since CMOS technologies consume minimal static power, holding the clock steady allows parts of the design to be effectively turned off during inactive periods.

3. ***Input latching:*** Another source of wasted power derives from the pre-emptive nature of synthesised architecture. Consider figure 2 again: the adder consumes energy whenever either input changes, even though the result may not be loaded into the output register until several cycles later, when the operation is considered to have been executed. This means that even if the result is not loaded (because

the inputs are used elsewhere, or the add is conditional) energy will be consumed. Latching the inputs only when the operation is executed ensures that the add will only occur when it is required. Furthermore, if a single unit is multiplexed between several operations of different bit-widths, say 16 and 32-bits, the top 16 bits will be ignored during the 16-bit operation, and may result in unnecessary energy consumption. Latching only the required inputs (in this case the lowest 16 bits) ensures the top bits remain stable throughout. Input latches cost area and delay, so synthesis must take this into account when using latched inputs - the result is that usually only high-power units such as multipliers, merit the use of such a technique.

4. ***Input gating:*** An alternative to latching inputs is to gate them so that when the operation is not required the inputs are disconnected (pulled to a constant value). This effectively means the unit performs two executions for each operation (one for the operation, and one on disconnection), but may still be sufficient to save energy, especially as there may be no area overhead if an input multiplexor is already present, as the MUX can simply be disabled to perform the gating.

5. ***Low-power cells:*** MOODS' low-level cell library can include many alternative implementations for each function (eg. ripple-carry, carry-select and carry lookahead adders) with different area, delay and energy characteristics. The inclusion of specific low-power cells, using smaller transistor sizes and alternative topologies allows the system greater flexibility when binding operations to physical units.

6. ***Pipelined and multi-cycle units:*** All of the features above, particularly V_{dd} scaling, can result in a degradation in system performance as a consequence of a reduction in energy consumption. To accommodate this, concurrency can be exploited, either through the use of multiple units, or via pipelining, to increase system throughput and regain performance. Serial cell architectures may also use less energy (eg. combinational vs. Booth multiplier). MOODS possesses an hierarchical module expansion capability [7, 8, 10] that enables pipelined and multi-cycle implementations to be inline expanded into the top-level control and data paths, thus allowing inter-module optimisation to occur, which can include power reduction. Furthermore, expanded modules are described in a technology-neutral format, which is automatically targeted to any low-level library during

synthesis. Thus, their energy characterisation is performed dynamically and does not require any circuit-level simulation or modelling - it simply relies on built-in activity data.

2.3 Multi-Dimensional Optimisation

The underlying concepts and mathematics of simulated annealing can be found in almost any basic text on optimisation[12]. The important issue here is the definition of the overall cost function. In MOODS' case, this is a weighted linear sum of the three design metrics, area, delay, and energy consumption[1,2]. Although absolute accuracy of these figures is desirable, from the perspective of the optimisation routine it is the *relative* accuracies that are important. The weightings of the coefficients are input by the user, and reflect the relative importance of the three dimensions to the human designer.

3 Power Profiler

The power profiler was developed in conjunction with power optimisation to automate the characterisation process. It takes a low-level cell description and automatically generates a circuit level testbench for the characterisation of delay, energy consumption and input capacitance. The tool splits into two parts:

1. Testbench generation automatically creates a circuit level test jig, and a set of test vectors to stimulate the cell. Application of stimuli is separated into three activity phases for each type of input, namely: data inputs, controls (such as register loads, or counter increments), and clocks. Data inputs and controls are random bit patterns, while the clock runs at a constant frequency. In addition, three timebase signals are generated which toggle at the start of each phase for analysis synchronisation during the second part of the process.

2. Profiling takes the results of the simulation and calculates the input capacitance, delay and energy consumed by the cell for various different loads, based on a time history of its execution including supply current and voltage data. Each of the three activity phases is separated to provide independent data for each input, control signal and clock. The resulting figures are then processed to create models suitable for inclusion in the MOODS low-level cell library. These models describe the average energy consumption per input activity, and are separated into three distinct stages: stage 1: data input changes; stage 2: control input changes while

the clock is stable; stage 3: clock active edge occurs while control inputs are stable. Table 1 shows the energy models generated for a ripple-carry adder, and a register when targeting the 0.6μm AMS digital cell library. Some points of note are:

- The adder model only includes a stage 1 activity, as it only has data inputs (ie. it is purely combinational), whereas the register model uses all three stages.
- The adder stage 1 is a function of both the bit-width and the load capacitance since a change on the inputs generates several output changes - ie. several charge/uncharge cycles of C_{load}. The register stage 1, however, is independent of load capacitance as the register output does not change when its input changes. Likewise for stage 2, since the clock is steady. Note that stage 2 "no signal" models the situation where a control signal changes from set to clear (ie. no control signals set).
- Register stage 3 for load enable is dependent on C_{load}, as it is at this point that the register output updates to drive the load. The "no signal" situation models the effect of clocking the register without any controls set - ie. it models idle power (see section 2.2). For a clock gated register, this component is substantially reduced as the clock will only drive a single gate - the clock enable gate.

4 Benchmark Results

To illustrate the effect of the power optimisation and reduction features, two sets of results are presented. The first is a detailed analysis of a benchmark design, optimised with respect to a selection of high-level criteria to demonstrate the trade-offs and improvements attainable. The second is a comparative table showing the achievements of the system on a set of benchmark designs. In all cases, the designs were targeted at the 0.6μm AMS CMOS process (cux) and standard digital cell library.

4.1 Elliptical Filter

Table 2 details the results of a number of different optimisations on a 32-bit elliptical filter high-level benchmark design [32]. For each one the total area, delay and energy consumption is listed, along with the energy consumption due to idle cycles (ie. unnecessary register loads, - section 2.2), the percentage of the total energy this

represents, and the optimised supply voltage. Run times are measured on a PIII-500 machine.

To illustrate the general power-minimisation and optimisation capabilities of the system, figure 3 shows am "unfolded" image of the three dimensional design space; the two axes labelled "energy" identify with each other. The points correspond to those described in table 2, and the optimisation trajectory from 0(unoptimised) to 4(optimised with respect to area, delay and power dissipation) shown. The other trajectories are omitted for the sake of clarity. Close examination of the structure of the final designs is revealing:

The two major contributors to energy reduction are the inclusion of energy estimation in the cost function, thus giving MOODS awareness of the effect of transformations on energy, and supply-voltage scaling. In summary, of the 3.8 times reduction in energy consumption achieved in implementation 4, we can say that around 53% is due to energy estimation, 36% comes from supply voltage scaling, and 11% from specific power-minimising enhancements.
Trajectories 8 and 9 show the effect of adding a delay constraint to energy minimisation. MOODS assigns a higher priority to achieving this figure, than to minimising energy. In both cases, the delay constraint is set to be slightly higher than the minimum delay figure: 1000 ns and 1100 ns. Both the optimised implementations obtained achieved the desired target, but they also achieved an energy consumption reduction of 1.9 and 2.2 times. This is the result of a combination of all the energy reduction mechanisms:

- Supply voltage reduction is used but not as aggressively as when minimising energy consumption alone as the delay effect is also taken into account.
- Gated and latched units have been exploited fully to minimise the idle energy component. These units only have a small effect on delay (due to a few extra gate delays), which can easily be accommodated with some additional concurrency.

4.2 Benchmark Set

Table 3 provides a set of area, delay and energy consumption figures for a selection of benchmark designs illustrating the range of improvements obtainable. The set comprises: elliptical filter (described above), a 16-bit, 15-stage FIR filter, and an implementation of the differential heat release algorithm (DHRC) [32]. For each, the

final area, delay and energy consumption figures are tabulated for three optimisation runs: minimum area, minimum delay and minimum energy consumption. Run times the optimisations range from 15 to 60 seconds.

The success of the system in trading three metrics against each other is clear.

5 Final Remarks

One of the principal design decisions taken during the inception of the MOODS system was to attempt, wherever possible to divorce the optimisation *process* from the nature of the optimisation *problem*. Simulated annealing, by its abstract nature, is ideally suited for this, and has made possible the construction of a system where disparate figures of merit can be sensibly compared to produce an overall *optimum*.

The resulting synthesis system, with the inclusion of the power-optimisation and reduction features described in this paper, has shown itself to be capable of taking into account the complex interactions and trade-offs involved in optimisation, with results showing a reduction in energy consumption of a range of benchmark designs of factors of between 3.5 and 7.0 times. Furthermore, it has demonstrated its capability to perform optimisation with respect to power/energy consumption, *on an equal footing* with area and performance (clock speed and delay).

6 Acknowledgements

This work was supported by EPSRC grant GR/K70748.

7 References

1. Baker K.R., Currie A.J., Nichols K.G., “Multiple Objective Optimisation in a Behavioural Synthesis System”, IEE Proceedings on Circuits Devices & Systems, 140, August 1993, pp 253-260.
2. Baker K. R., Brown A. D., Currie A. J., “Optimisation Efficiency in Behavioural Synthesis”, IEE Proceedings on Circuits Devices & Systems, 141, No. 5, pp 399-406, 1994.
3. Baker, K.R., “Final Report: Application Specific Synthesis Enforcing Testability (ASSET)”, University of Southampton, October 1995.
4. Nijhar, T.P.K, Brown, A.D., “Source Level Optimisation of VHDL for Behavioural Synthesis”, IEE Proceedings on Computers and Digital Techniques, 144, no. 1, January 1997, pp. 1-6.
5. Nijhar, T.P.K, A.D. Brown, A.D., “HDL-Specific Source Level Behavioural Optimisation”, IEE Proceedings on Computers and Digital Techniques, 144, No. 2, March 1997, pp 138-144.

6. Brown, A.D., Baker K. R., Williams, A.C., “Online Testing of Statically and Dynamically Scheduled Synthesized Systems”, IEEE Transactions on CAD Vol. 16, No. 1, pp 47-57, 1997.
7. Williams A. C., “A Behavioural VHDL Synthesis System using Data Path Optimisation”, PhD thesis, University of Southampton, UK, July 1997.
8. Williams, A.C., Brown, A.D., Baidas, Z.A., “Hierarchical Module Expansion in a VHDL Behavioural Synthesis System”, Forum on Design Languages 1998 (FDL’98), 1998.
9. Williams, A.C, Brown, A.D., Zwolinski, M., “Inline Test of Synthesized Systems Exploiting Latency Analysis”, accepted for publication in IEE Proceedings on Computers and Digital Techniques.
10. Williams, A.C., Brown, A.D., Baidas, Z., “Optimisation in Behavioural Synthesis using Hierarchical Expansion: Module Ripping”, submitted to IEE Proceedings on Computers and Digital Techniques, awaiting referees comments.
11. Baidas, Z., Brown, A.D., Williams, A.C., “Floating Point Behavioural Synthesis”, submitted to IEEE Transactions on CAD, awaiting referees comments.
12. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P., “Optimization by Simulated Annealing”, Science, Vol. 220, No. 4598, 13th May 1983, pp. 671-680.
13. Singh, D., Rabaey, J.M., Pedram, M., Catthoor, F., Rajgopal, S., Sehgal, N., Mozdzen, T.J., “Power Conscious CAD Tools and Methodologies: A Perspective”, Proceedings of the IEEE, Vol. 83, No. 4, April 1995, pp. 570-594.
14. Veendrick, H.J.M., “Short-Circuit Dissipation of Static CMOS Circuitry and its Impact on the Design of Buffer Circuits”, IEEE Journal of Solid-State Circuits, pp. 468-473, August 1984.
15. Najim, F.N., “A Survey of Power Estimation Techniques in VLSI Circuits”, IEEE Transactions on Very Large Scale Integration, Vol. 2, No. 4, December 1994, pp. 446-455.
16. Kang, A.M., “Accurate simulation of power dissipation in VLSI circuits,” IEEE Journal of Solid-State Circuits, Vol. SC-21, No. 5, October 1986, pp. 889-891.
17. Vanoostende, P., Six, P., Vendewalle, J., “Estimation of Typical Power of Synchronous CMOS Circuits Using a Hierarchy of Simulators”, IEEE Journal of Solid-State Circuits, Vol. 28, No. 1, January 1993, pp.26-39.
18. Landman, P.E., Rabaey, J.M., “Activity-Sensitive Architectural Power Analysis”, IEEE Transactions on CAD, Vol. 15, No. 6, June 1996, pp. 571-587.
19. Boglioli, A., Benini, L., Ricco, B., “Power Estimation of Cell-Based CMOS Circuits”, Design Automation Conference, 1996.
20. Macii, E., Pedram, M., Somenzi, F., “High-Level Power Modeling, Estimation, and Optimization”, Design Automation Conference, 1997.
21. Raghunathan, A., Dey, S., Jha, N.K., “Register-Transfer Level Estimation Techniques for Switching Activity and Power Consumption”, Proceedings of ICCAD, November 1996, pp. 158-165.
22. Khouri, K.S., Lakshminarayana, G., Jha, N.K., “Fast High-level Power Estimation for Control-flow Intensive Designs”, ISLPED’98.

23. Devadas, S., Malik, S., “A Survey of Optimization Techniques Targeting Low Power VLSI Circuits”, 32nd ACM/IEEE Design Automation Conference, 1995.

24. Musoll, E., Cortadella, J., “High-level synthesis techniques for reducing the activity of functional units”, ISLPD’95.

25. San Martin, R., Knight, J.P., “Optimizing Power in ASIC Behavioural Synthesis”, IEEE Design and Test of Computers, Summer 1998, pp.58-70.

26. Raghunathan, A., Jha, N.K., “Behavioral Synthesis for Low Power”, Proceedings of ICCD, 1994.

27. Chandrakasan, A.P., Potkonjak, M., Mehra, R., Rabaey, J., Brodersen, R., “Optimizing Power Using Transformations”, ”, IEEE Transactions on CAD, Vol. 14, No. 1, January 1995, pp. 12-31.

28. Kumar, N., Katkoori, S., Rader, L., Vemuri, R., “Profile-Driven Behavioral Synthesis for Low-Power VLSI Systems”, IEEE Design and Test of Computers, Fall 1995, pp. 70-84.

29. Lakshminarayana, G., Jha, N.K., “High-Level Synthesis of Power-Optimized and Area-Optimized Circuits from Hierarchical Data-Flow Intensive Behaviors”, IEEE Transactions on CAD, Vol. 18, No. 3, March 1999, pp. 265-281.

30. Liu, D., Svensson, C., “Trading Speed for Low Power by Choice of Supply and Threshold Voltages”, IEEE Journal of Solid-State Circuits, Vol. 28, No. 1, January 1993, pp. 10-17.

31. Benini, L., De Micheli, G., “Automatic Synthesis of Low-Power Gated-Clock Finite-State Machines”, IEEE Transactions on CAD, Vol. 15, No. 6, June 1996, pp. 630-643.

32. Vemuri, R., Roy, J., Mamtora, P., Kumar, N., “Benchmarks for High Level Synthesis”, Laboratory for Digital Design Environments, University of Cincinnati, 1991.

Tables and Figures

Table 1: Adder and register energy models

Table 2: Elliptical filter benchmarks optimisation results

Table 3: Typical benchmark optimisation results

Figure 1: Low power synthesis system architecture

Figure 2: Energy consumption for an add operation

Figure 3: Various optimisation strategies for the elliptical filter

Unit type	Operation stage	Energy per activation model (J) *
Ripple carry adder	Stage 1	-9.1e-12 + 9.8e-12 N + 3.9 C_{load} + 7.7 N C_{load}
Register	Stage 1	1.7e-12 N
	Stage 2, no signals	5.1e-12 N
	Stage 2, load	3.3e-12 N
	Stage 3, no signals	3.4e-12 N
	Stage 3, load	6.6e-12 N + 11.0 N C_{load}

* N is unit bit-width, C_{load} is capacitance driven

Table 1: Adder and register energy models

Optimisation	Area (gates)	Delay (ns)	Energy (µJ)	Idle (µJ)	Idle (%)	V_{dd} (V)	Runtime (s)
0: Design implemented with one clock cycle and data path unit per operation.	29098	3404	101.8	51.9	51.0	5.0	0
1: Minimum area.	5173	3199	78.4	12.1	15.3	5.0	16.0
2. Minimum total delay.	12637	925	75.1	7.7	10.3	5.5	8.2
3. Minimum energy consumption - using all new transforms and power-minimising features.	24555	4257	20.0	0.0	0.0	2.5	6.2
4. Minimum area, delay and energy.	10189	1315	39.6	0.0	0.0	3.6	8.5
5. Minimum energy consumption, all power-minimising features turned off.	23909	1655	46.0	3.7	8.1	5.0	6.0
6. Minimum energy consumption, supply voltage scaling off, all other power-minimising features on.	24502	1995	40.1	0.0	0.0	5.0	6.3
7. Minimum energy consumption, supply voltage scaling on, all other power-minimising features off.	20822	4024	25.1	2.9	11.5	2.5	6.2
8. Delay target 1000 ns, minimum energy - sets an explicit delay target 75 ns above the minimum delay figure.	27110	999	38.7	0.0	0.0	4.4	7.1
9. Delay target 1100 ns, minimum energy - as above but with a slightly higher delay constraint.	27037	1100	34.2	0.0	0.0	4.0	6.8

Table 2: Elliptical filter benchmark optimisation results

Design	Area minimised			Delay minimised			Energy minimised		
	Area (gates)	Delay (ns)	Energy (µJ)	Area (gates)	Delay (ns)	Energy (µJ)	Area (gates)	Delay (ns)	Energy (µJ)
Ellip	5173	3199	78.4	12637	925	75.1	24555	4257	20.0
FIR	3683	6174	90.5	3704	2751	61.8	4394	7949	13.0
DHRC	4323	1607	71.2	4859	900	39.6	6385	4425	11.2

Table 3: Typical benchmark optimisation results

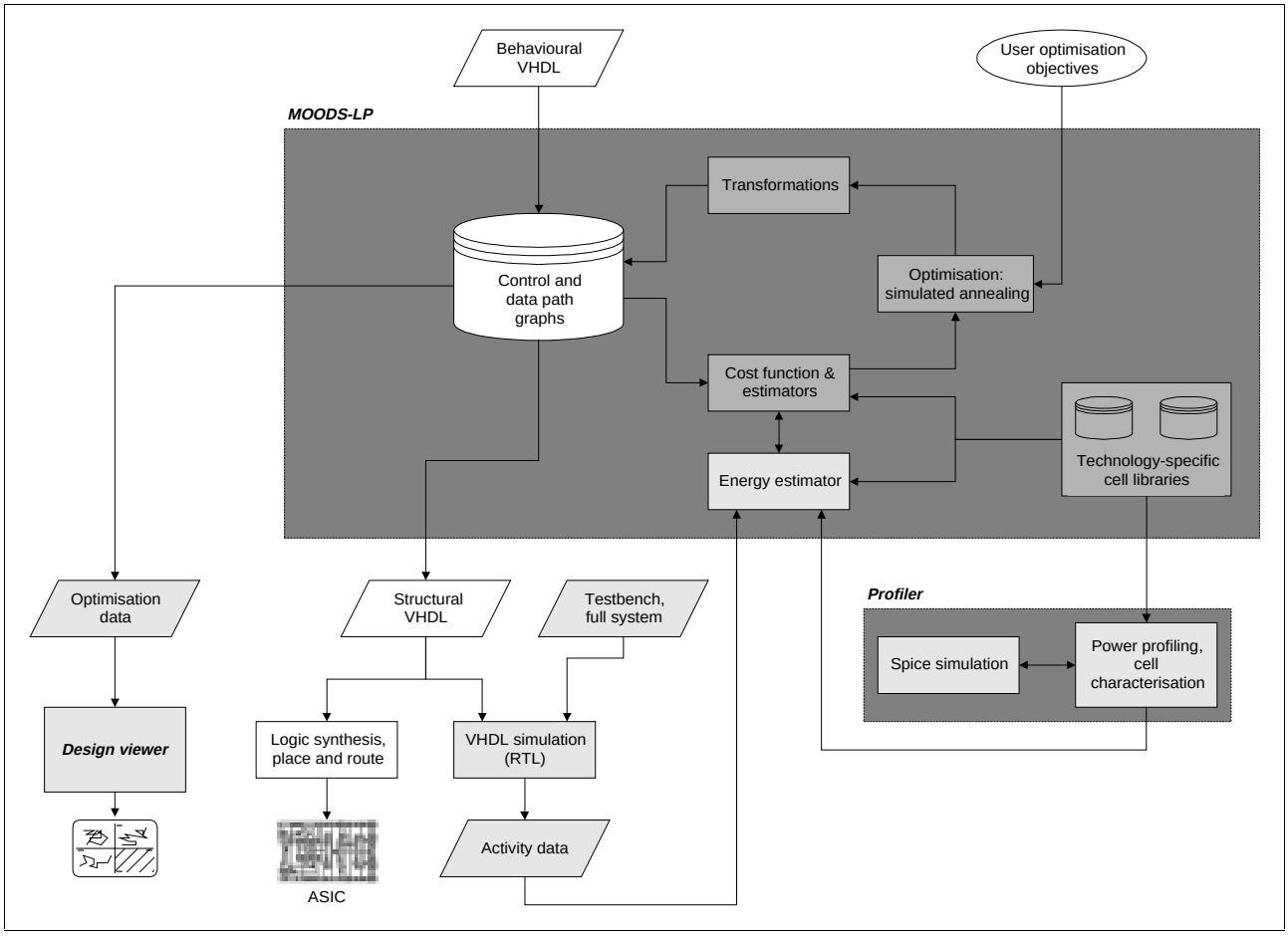

Figure 1: Low power synthesis system architecture

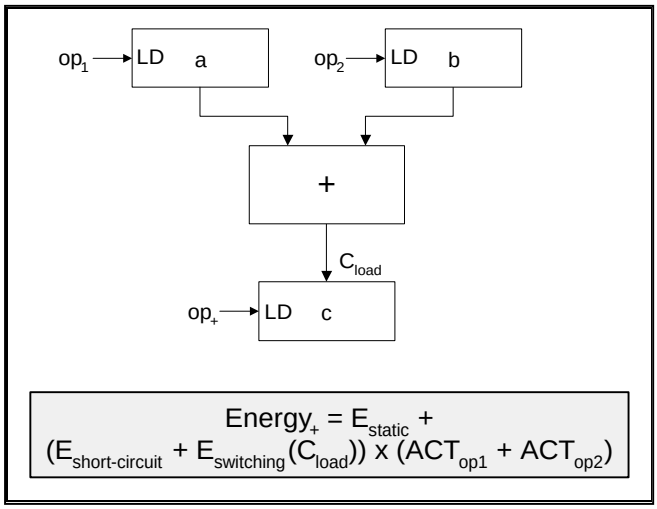

Figure 2: Energy consumption for an add operation

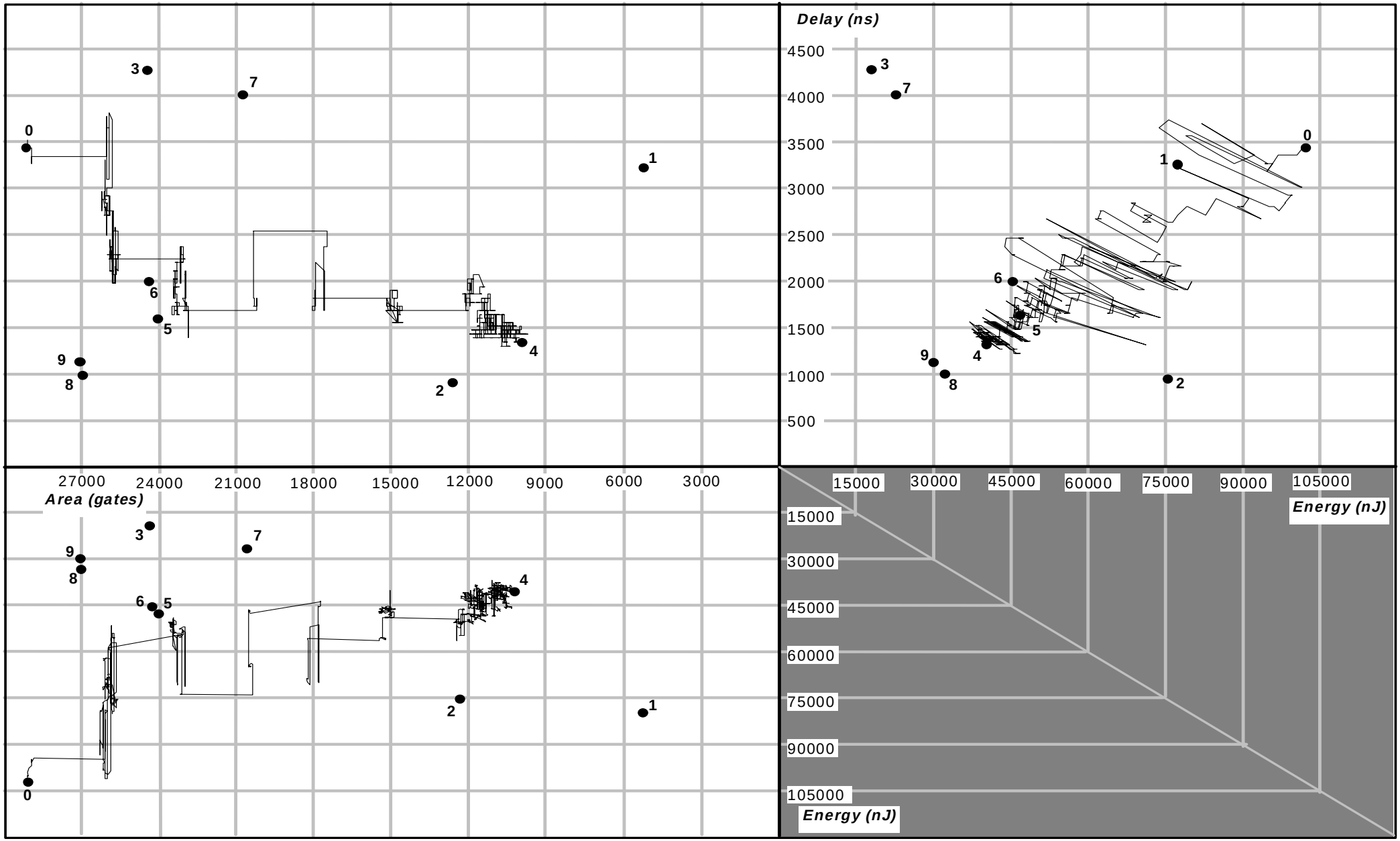

Figure 3: Various optimisation strategies for the elliptical filter