# Power-Profiler: Optimizing ASICs Power Consumption at the Behavioral Level

Raul San Martin and John P. Knight
Department of Electronics
Carleton University
Ottawa, Ontario, Canada K1S 5B6

**Abstract - This paper presents a methodology and tool (Power-Profiler) for the optimization of average and peak power consumption in the behavioral synthesis of ASICs. It considers lowering operating voltages, disabling the clock of components not in use, and architectural trade-offs, while also keeping silicon area at reasonable sizes. By attacking the power problem from the behavioral level, it can exploit an application's inherent parallelism to meet the desired performance and compensate for slower and less power-hungry operators.**

## I. INTRODUCTION

Designing low-power ASICs is receiving increasing attention as portable communications and personal assistant devices proliferate. However, the advantages of low-power circuits apply not only to portable devices: smaller and lighter products, higher reliability, lower heat dissipation and cheaper IC packages, longer battery life, etc. Lower power peaks also improve supply currents and voltage levels, and lower electromigration and EMI. The ASIC designer can tackle power consumption at several levels:

The processing/transistor-design/environment level [1,2]:

- Lower the supply voltage; reduce the geometry, the threshold voltage and subthreshold slope; reduce the temperature.

The gate-design level [3]:

- Use static rather than dynamic CMOS; use reversible logic (e.g., adiabatic gates); order the transistor turn-on sequence.

The logic level [4,5]:

- Reduce hazards and other nonproductive transitions; resize the gates; use asynchronous circuits.

The register-transfer design level [6]:

- Choose adjacent states for frequent state transitions.

The behavioral level [7,8,9,10] as applied to ASICs, not processors, is the concern of this paper. Behavioral synthesis attacks the design problem from higher levels of abstraction, allowing exploitation of the parallelism inherent in many ASICs applications (notably, DSP). This allows proper design choices to compensate for slower and less power-hungry operators. This is much harder or impossible to do at lower levels of design.

**32nd ACM/IEEE Design Automation Conference ®**


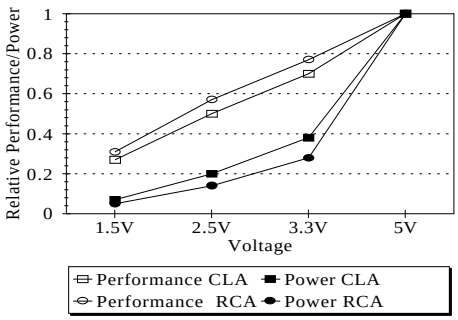


Fig. 1. Speed and power performance of RCA and CLA (relative to 5V).

This paper presents a behavioral synthesis tool **(Power-Profiler**) that can solve both average and peak power problems while also keeping silicon area at reasonable levels. It considers:

**Shut down of operators**. The massive switching activity in large components, such as adders and multipliers, consumes great amounts of energy. By disabling the clock the internal nodes remain at static voltage levels and do not consume power.

**Lower supply voltage**s. In CMOS, power consumption decreases quadratically with voltage while the speed reduction is linear. This is clearly shown in fig. 1, which shows the relative speed and power consumption of 32-bit ripple-carry and carry lookahead adders as the voltage is decreased (reference is 5V).

**Mixed voltage circuits**. Dual voltages on one IC are attractive enough so that they are being considered commercially [11]. Although viable, crosstalk and latchup are among the lower-level problems that must be carefully considered.

**Increased parallelism and use of fast and power-hungry operators only on time-critical data paths**. Slower operators can be used on non-time-critical paths (for example, digit-serial arithmetic can significantly reduce power [7]), while parallelism can be increased to compensate for slower components. Consider the data-flow graph of fig. 2. The multiplications can be implemented serially at 5V, as in schedule A, or in parallel at 3.3V, as in schedule B. The power consumption of the parallel option is lower and its total delay is smaller. However, the extra area used by the second multiplier must also be considered.

A data-flow graph inherently shows what operations may be done in parallel. Since behavioral synthesis works from the data-flow graph, it allows relatively easy exploitation of this parallelism. The data-flow graph also shows what operations are critical to finishing the complete calculation on time, and which can be time-spread without affecting the overall completion time.

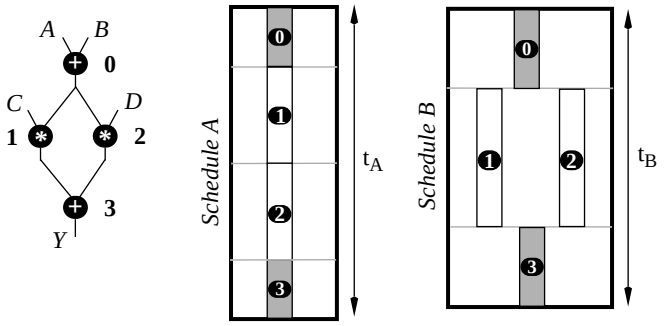


Fig. 2. Behavioral implementation example.

## II. Power profiles

We designed several types of 32-bit operators, including: ripple-carry adder (RCA), carry lookahead adder (CLA), Booth multiplier (BOOTH), Quasi Bit-Serial multiplier (QBS), Array multiplier (ARR32), and a 16-bit digit-serial array multiplier (ARR16). Table I shows their power and delay values when a system clock of 100 MHz is used (i.e., clock steps are 10 ns). These values are obtained from HSpice simulations using 0.8 µm transistor models.

TABLE I DELAY AND AVERAGE POWER OF OPERATORS

| | | 3.3V | 5V |
|---|---|---|---|
| RCA | Total Delay (ns) | 30.0 | 20.0 |
| | Power (mW) | 5.4 | 22.7 |
| CLA | Total Delay (ns) | 20.0 | 10.0 |
| | Power (mW) | 10.5 | 37.3 |
| QBS | Total Delay (ns) | 640.0 | 640.0 |
| | Power (mW) | 11.00 | 28.5 |
| BOOTH | Total Delay (ns) | 320.0 | 160.0 |
| | Power (mW) | 12.7 | 84.0 |
| ARR16 | Total Delay (ns) | 330.0 | 170.0 |
| | Power (mW) | 24.6 | 101.3 |
| ARR32 | Total Delay (ns) | 160.0 | 100.0 |
| | Power (mW) | 143.1 | 295.6 |

The power values are obtained by applying square waves of different frequencies at the inputs in such a manner that a balanced mix of 1s and 0s is applied at any given time. Stochastic and statistical estimates can also be used. More accurate estimates may be obtained if fixed coefficients are used at one of the inputs (as is usually the case with digital filters) or by considering the effects of sign bits and inputs correlation. Fig. 3 shows the energy consumption and delay (at 3.3V and 5V) for each type of adder and multiplier implementing a complete 32-bit operation.

In many operators most of the power is dissipated soon after new inputs are clocked as an initial flurry of activity occurs. Although values may change with the input data, the power distribution is typical. It is possible to obtain a profile of the power dissipation of each operator by simulating it and averaging the power consumed at each specific clock cycle over many samples. Fig. 4 shows the typical power profile, or dynamic power distri-

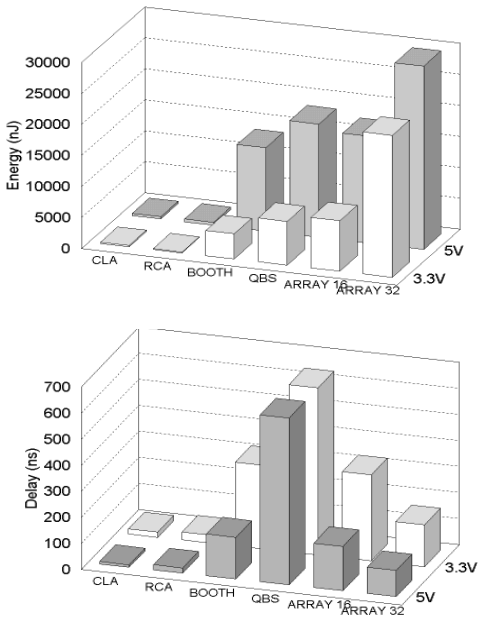


Fig. 3. Energy consumption and delays per 32-bit operation.

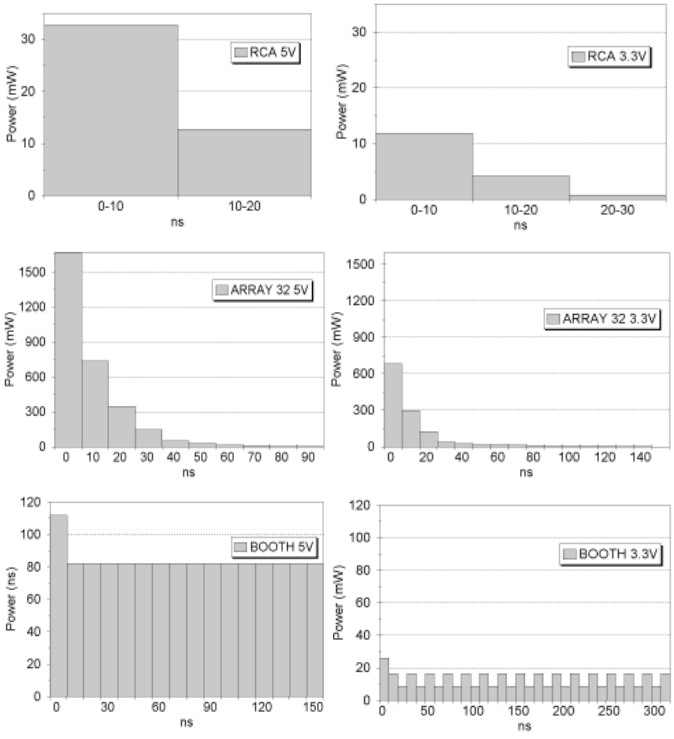


Fig. 4. Power profiles: RCA adder, ARR32, and BOOTH multipliers.

These power profiles reflect the behavior of each architecture. For example, the 5V Booth multiplier uses regular cycles to compute a multiplication. This results in homogenous consumption throughout the execution time. On the other hand, the RCA adder and the ARR32 multiplier do not behave in regular cycles and most switching happens at the start of the computation. This is also reflected on their power profiles. The 3.3V Booth multiplier

needs two clock cycles for each stage since it cannot operate at 100 MHz. The first cycle consumes more power as no switching occurs after the computation is completed. The profiles of the CLA adder, ARR16, and QBS multipliers are obtained similarly.

## III. Effects of Assigment on Average Power

Module selection, i.e. the ***assignment*** of operations to functional units, can have dramatic effects on power consumption. To show this we will consider the operations of the data flow graph of fig. 2 under several different assignments.

Let us assume that the same type of adder implements operations 0 and 3 and that the same type of multiplier implements operations 1 and 2. If $t_{ADD}$ is the delay of an addition and $t_{MUL}$ is the delay of a multiplication, then the execution time under schedule A is:

$$t_A = 2 \times t_{ADD} + 2 \times t_{MUL}$$

while under schedule B the execution time is:

$$t_B = 2 \times t_{ADD} + t_{MUL}$$

$P_{ADD}$ and $P_{MUL}$ are the average power consumptions for the adders and multipliers. Then, average power consumption under schedule A is calculated by:

$$P_A = \frac{2 \times P_{ADD} \times t_{ADD} + 2 \times P_{MUL} \times t_{MUL}}{t_A}$$

Average power consumption under schedule B is given by:

$$P_B = \frac{2 \times P_{ADD} \times t_{ADD} + 2 \times P_{MUL} \times t_{MUL}}{t_B}$$

Consider a supply voltage of 5V with a CLA adder and a QBS multiplier. Using the values of table I, $t_A$ = 1,300 ns and $t_B$ = 660 ns, while $P_A$ = 28.6 mW and $P_B$ = 56.4 mW. If RCA adders or array multipliers or BOOTH multipliers are used instead, the values change completely. Table II shows a few combinations. This shows how different assignments can affect power consumption. If the supply voltage were 3.3V, yet another completely different set of values would be obtained.

TABLE II ASSIGNMENT AND AVERAGE POWER (5V)

| | | RCA | CLA |
|---|---|---|---|
| Multipliers | QBS | $t_A$=1320 ns $P_A$=28.3 mW<br>$t_B$=680 ns $P_B$=55.0 mW | $t_A$=1300 ns $P_A$=28.6 mW<br>$t_B$=660 ns $P_B$=56.4 mW |
| | BOOTH | $t_A$=360 ns $P_A$=77.2 mW<br>$t_B$=200 ns $P_B$=138.9 mW | $t_A$=340 ns $P_A$=81.3 mW<br>$t_B$=180 ns $P_B$=153.5 mW |
| | ARRAY 32 | $t_A$=240 ns $P_A$=250.1 mW<br>$t_B$=140 ns $P_B$=428.8 mW | $t_A$=220 ns $P_A$=272.1 mW<br>$t_B$=120 ns $P_B$=498.9 mW |

Note that the several power values in table II correspond to different delays and therefore should not be compared directly. A normalized delay should be used instead, as shown in section IV.

The solution space of this type of problem grows exponentially with the number of operations of the data flow graph. A typical practical application may contain hundreds of operations and a much larger module library. With 50 additions and 50 multiplications and our library (two types of adders and four types of multipliers) the number of choices is approximately $2^{50}$ x $4^{50}$ = ~$10^{45}$. This is the complexity of the assignment problem alone.

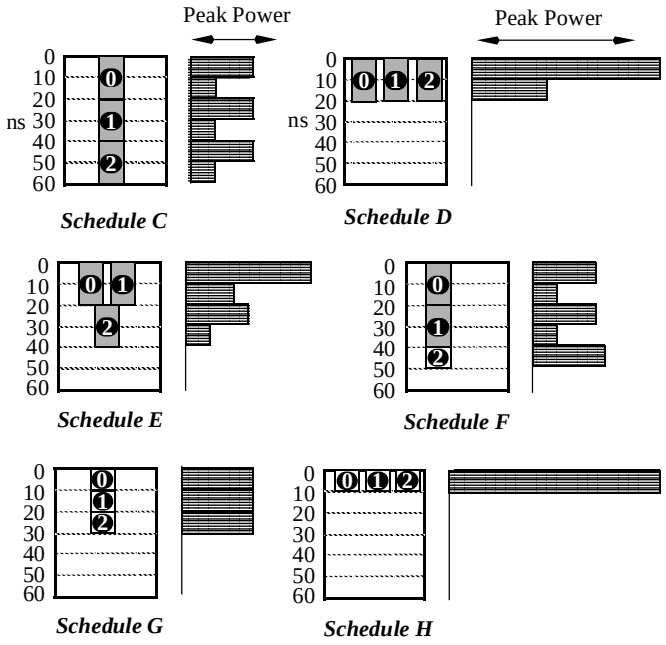


Fig. 5. Effects of different assignments on peak power.

## IV. Peak Power

To estimate peak power, we must consider the power profiles of each operator and the resulting power profile of the entire circuit. Fig. 5 shows the peak power profiles of three additions implemented with ripple-carry and carry lookahead adders.

Peak power and maximum delays are shown in table III. Schedule C has the lowest peak power but it is also the slowest. Schedule H is the fastest of all, has the highest peak power but consumes the lowest average power if the maximum delay allowed is normalized to 60 ns.

This table also shows that peak power and average power optimization are quite different problems. Typically, the total number of combinations, including assignment and scheduling, can reach over $10^{500}$ depending on the size of the data flow graph and on the timing constraint used. In addition to the scheduling of operations to precise time slots, all precedence relations between operations must be obeyed (these can easily reach thousands).

## V. Considering Silicon Area

While increasing system parallelism speeds up a circuit and may compensate for slower operators, it carries an area cost which cannot be overlooked. Table IV shows the number of transistors used by the adders and multipliers used in this paper. Alternatively, the values of this table can be substituted by the actual silicon areas (in $\mu m^2$, for instance) of the library used.

Consider the schedules of fig. 5. While schedule C has the

TABLE III DELAYS AND POWER OF SEVERAL SCHEDULES

| | | Delay (ns) | Peak Power (mW) | Average Power at min. delay (mW) | Average power at 60 ns (mW) |
|---|---|---|---|---|---|
| Schedule | C | 60 | 32.7 | 22.7 | 22.7 |
| | D | 20 | 98.1 | 68.1 | 22.7 |
| | E | 40 | 65.4 | 34.1 | 22.7 |
| | F | 50 | 37.3 | 25.6 | 21.3 |
| | G | 30 | 37.3 | 37.3 | 18.7 |
| | H | 10 | 111.9 | 112.0 | 18.7 |

TABLE IV TRANSISTOR COUNTS OF ADDERS AND MULTIPLIERS

| RCA | CLA | QBS | BOOTH | ARR 16 | ARR 32 |
|---|---|---|---|---|---|
| 1,330 | 2,326 | 1,882 | 3,808 | 11,386 | 32,896 |

lowest transistor count it is also the slowest. Schedule H has the highest transistor count but is the fastest. Schedules D to G are somewhere in between. Many other schedules are possible with intermediate values for delay, transistor counts, and power.

Since operators using a supply voltage of 3.3V are slower, it is possible that for tight timing constraints the number of transistors needed will be much higher. This might be due to added parallelism (thus increasing the number of operators) or to the use of faster architectures (e.g., array instead of Booth multipliers).

Consider schedule A of fig. 6 operating at 5V and using RCA adders and Booth multipliers. The delay is 360 ns and the number of transistors is 5,138. The same configuration with a supply voltage of 3.3V has a delay of 700 ns (schedule A1). If the original delay of 360 ns is desired, the options are to use a schedule such as A2, but using a CLA/ARR32 combination, or schedule similar to B with a CLA/BOOTH combination. Both of these solutions have a delay of 360 ns but their total transistor count is 35,222 and 9,942 respectively. Therefore, in this example a 3.3V implementation needs far more transistors than a 5V implementation. In same cases, the large area may render the 3.3V solution impractical. In other cases, it may not even be possible for a 3.3V solution to meet very fast timing constraints.

## VI. POWER-PROFILER, THE TOOL

Power-Profiler is capable of several types of optimization with many types of constraints:

- Minimize either average or peak power with area and/or delay constraints.
- Minimize delay with area and/or power constraints.
- Minimize area with delay, and/or power constraints.
- Minimize any weighted combination of area, average and peak power with multiple constraints.

These optimizations involve simultaneous scheduling and assignment. The optimizer searches for the best combination of architecture (module selection) and schedule to minimize the desired function while satisfying the given constraints. The search space, the number of precedence relations and the total number of paths from input to output in a data flow graph are typically extremely large (please see table V).

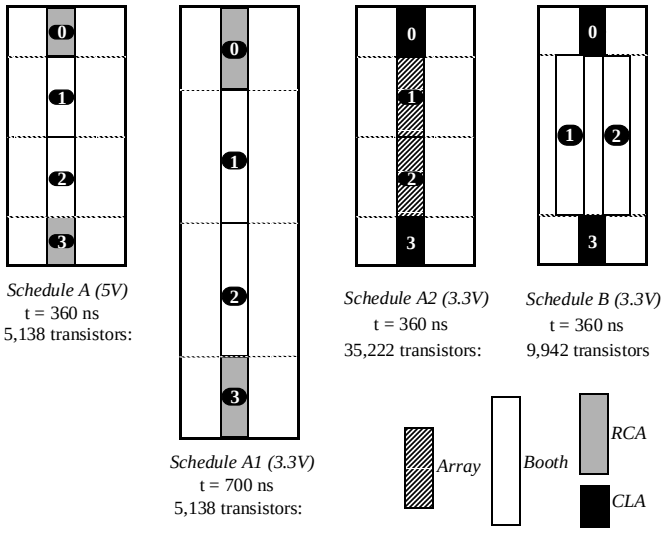


Fig. 6. Schedules, supply voltages and transistor counts.

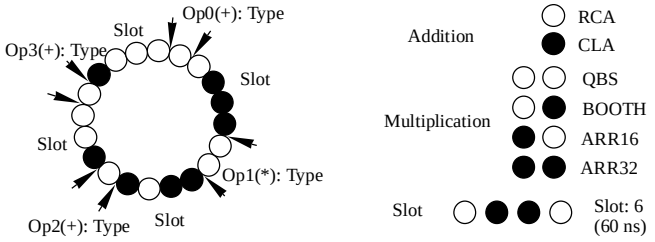


Fig. 7. Typical encoding as a “chromosome”.

Initially we tried heuristic methods to do the optimization. Heuristics tailored to specific problems and optimization objectives may run fairly quickly, but they almost invariably take an excessive time to modify to accommodate almost trivial changes. At this point we tried general optimization methods, with the winner being genetic algorithms [12]. Execution times are still very reasonable (seconds to minutes on a SPARCstation5™). However, reprogramming major changes (e.g., changing the objective functions, constraints, problem type, etc.) could also be done in minutes with only a handful of additional lines of code.

**The problem encoding** is shown in fig. 7. Each operation is represented by two fields. The first one indicates which architecture to use (i.e., CLA, RCA for adders), while the second field indicates the time slot in which the operation is scheduled. The length of the slots is user-defined. With a clock of 100 MHz, each slot represents 10 ns. These slots are relative. For instance, if operation 4 is preceded by operation 3 in the data flow graph, then the slot indicated is relative to the time in which operation 3 is completed. The use of relative time slots guarantees that all precedence relationships are satisfied (precedence feasibility).

**Average power** is calculated as shown on section III. The number of times each type of operator is used is multiplied by its average power and by its delay, giving the total energy used by the operator. All these individual energy values are then added and divided by the overall dataflow graph delay to obtain the

average power consumption:

$$Average\ Power = \frac{\sum_{All\ operators} N_{op} \times t_{op} \times P_{op}}{Total\ Delay}$$

**Peak power** is calculated by checking the total peak power at each time slot. The largest value found is the peak power of the solution being analyzed. **Transistor count** is calculated by checking the maximum number of operators of each type at every time slot. **The objective functions** use cascaded penalties as in [7]. As each objective function is evaluated (e.g. average power), the side constraints (e.g., peak power, transistor count, delay) are also calculated. If any of these exceed their limit by a certain percentage, the same percentage plus a fixed penalty is added to the objective function, thereby reducing its fitness. The advantage of this approach is that close-to-optimal solutions, yet infeasible, are "kept alive" and can be improved in subsequent generations. **Input files** are very simple, consisting of lines describing the operations (addition or multiplication), a list of precedence relations, optimization goals, constraints, clock cycle, etc.

## VII. Applications and Results

Typical applications range in size from 34 to 170 operations (table V). EWF is the fifth-order elliptic wave filter, DCT is the discrete cosine transform and INU5 is the EWF filter unrolled five times. The CPU times required to achieve a very good solution (within 1% of optimal) in this extremely large and complex problem are still about 11 minutes (average of 20 runs).

TABLE V CHARACTERISTICS OF TYPICAL APPLICATIONS

| | EWF | DCT | INU5 |
|---|---|---|---|
| Operations | 34 | 48 | 170 |
| Prec. relations | 47 | 65 | 269 |
| Solution space | $10^{70}$-$10^{90}$ | $10^{100}$-$10^{150}$ | $10^{400}$-$10^{500}$ |
| CPU time (min) | 0.4 | 1.2 | 11.8 |
| Paths | 57 | 52 | 601,692,057 |

**Conventional behavioral synthesis** assigns the operations of a data flow graph to given specific modules. If only Booth multipliers are used in the EWF example, the smallest possible delay at 5V is 590 ns. For faster circuits, array multipliers must be used. Fig. 8(a) shows the average power consumption of the conventional approaches obtained with a limited library (CLA for all additions with ARR32 or BOOTH for all multiplications). The resulting power curves are compared with the solution that selects and mixes components from this set. Fig. 8(b) shows the corresponding transistor counts. Power Profiler is able to bridge the curves of the Booth and array multipliers offering the best combination of power and transistor count. For example, at a delay of 570 ns, the best solution uses one array multiplier and two Booth multipliers, saving 39% in average power and 60% in transistor counts over the array multiplier solution.

**Peak power profiles**, or power distribution graphs, are shown on fig. 9 for both average and peak optimization of the DCT and INU5 examples. In the DCT example, optimization for average power achieves a solution that consumes 227.0 mW while with peak optimization this value rises to 233.1 mW. However, it is clear that with peak optimization the power is much better distributed throughout the entire delay. Similar results are obtained for the INU5 example. In both examples the small penalty in average power is well worth a much superior power distribution. The advantage of minimizing the power peaks is fairly obvious (reduction of 66% for the DCT and 47% for the INU5).

(a)

(b)

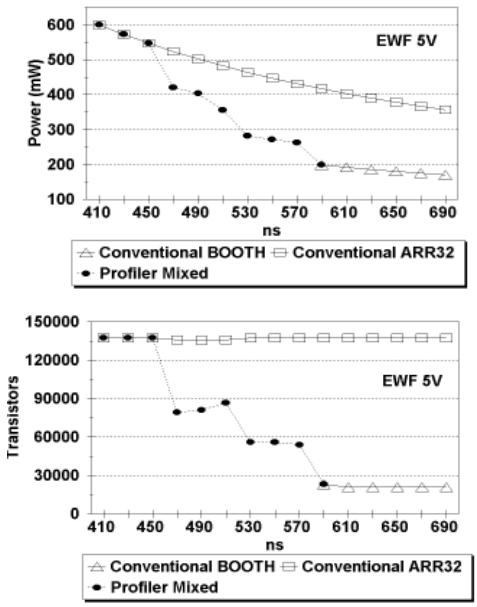


Fig. 8. Comparison with single module approach.

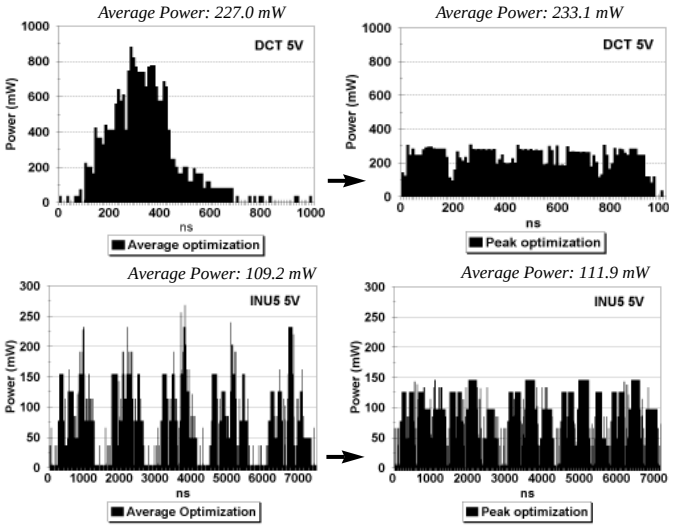


Fig. 9. Peak power profiles for a delay of 1,000 ns.

**Mixed supply voltages** solutions are shown on fig. 10, which shows average power on the INU5. At 3.3V, the minimum feasible delay is 4,000 ns. Faster circuits require the use of 5V supply voltages. Typically, power reductions of up to 70% are achieved by mixing operators of different voltages and combining the best of both worlds (low power with high speed). These good mixed-voltage results are also obtained for the other applications.

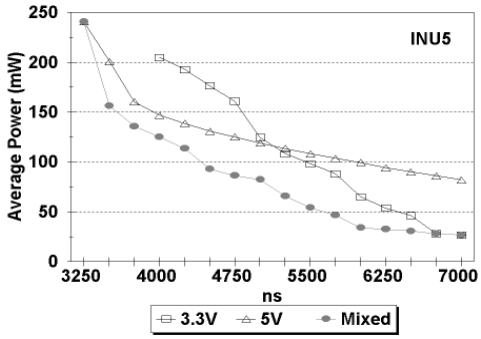


Fig. 10. Considering average power and supply voltages.

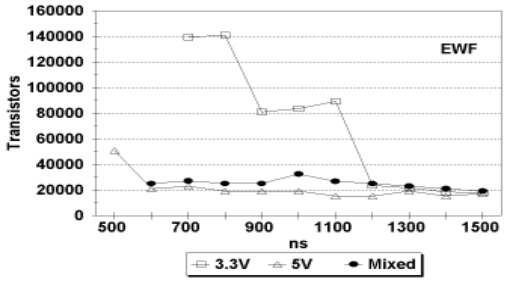


Fig. 11. Considering transistor counts and supply voltages.

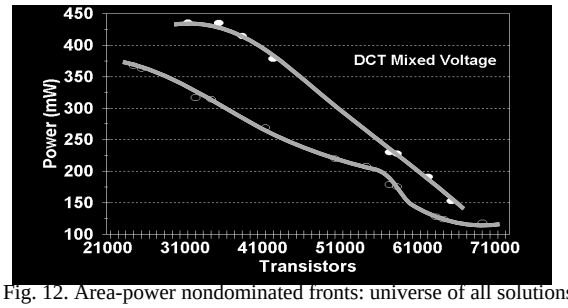


Fig. 12. Area-power nondominated fronts: universe of all solutions.

**Silicon area** is considered for the EWF in fig. 11. While the power consumption at 3.3V may be lower, the transistor count is usually higher than at 5V. It is clear that the increased parallelism necessary with the lower voltage causes an increase in silicon area. This increase may be too high to be accommodated in a particular application.

In many situations the designer needs to consider both area and power. Power-Profiler can be used to automatically generate the area-power nondominated fronts, as shown in fig. 12 for the DCT under two delays. This figure shows the universe of all solutions available to the designer when optimizing **both** area and power.

## VIII. Conclusions

The importance of low-power systems is obvious when we look at today's sales figures of hundreds of millions of dollars for portable devices. However, the benefits of low power apply not only to portable devices but also to ASICs in general.

The current trend towards lower supply voltages is validated by the results presented here. However, lower voltages without optimizing the physical process come at a larger area cost, particularly for high-performance circuits.

There are many advantages in attacking the power problem from the behavioral level. Lower-level tools are time consuming and restrictive. Behavioral tools such as Power-Profiler allow the ASIC designer:

- To find the best combination of operators from a library to meet the desired performance while satisfying constraints.
- Superior exploration of the design space since designs can be evaluated quickly and without restrictive assumptions.
- To avoid costly redesigns by estimating and optimizing power during the early stages of the design.

The tool presented in this paper tackles both energy consumption (through average power) and peak instantaneous power problems. While we used a library consisting of ripple-carry and carry lookahead adders, bit-serial, digit-serial, Booth, and Array multipliers, other operators can be easily incorporated. The tool is very flexible, fast, and very easy to use. Several applications showed that significant reductions in average power can be achieved by correctly selecting the operators and the supply voltages. In addition, by considering the typical power distributions of operators, power peaks can be greatly reduced, obtaining much better power distributions along the time axis.

## References

1 D. P. Foty and E.D. Nowak, "MOSFET technology for low-voltage/low-power applications," *IEEE Micro*, vol. 14, NO 3, pp. 68-77, June 1994.

2 A.P. Chandrakasan, S. Sheng, and R.W. Brodersen, "Low-power CMOS digital design", *IEEE J. Solid-State Circuits*, Vol. 27, No. 4, 1992, pp. 473-483.

3 W. Lee, U. Ko, and P.T. Balsara, "A comparative study of CMOS digital circuit families for low-power applications," *Int. Workshop on Low-Power Devices*, Napa, CA, pp.129-138, April 1994.

4 L. Benini, M. Favalli, and B. Ricco, "Analysis of hazard contribution to power dissipation in CMOS ICs," *Proc. Int. Workshop on Low-Power Devices*, Napa, CA, pp. 27-38, April, 1994.

5 W. Jone and C.Fang, "Timing optimization by gate resizing and critical path identification," *Proc. 30th ACM/IEEE Design Automation Conference.*, pp. 135-139, June 1993.

6 E. Olson and S.M. Kang, "Low-power state assignment for finite state machines.", *Proc. Int. Workshop on Low-Power Devices*., Napa, CA, pp. 63-68, April 24-27, 1994.

7 R. San Martin and J. P. Knight, "PASSOS: a different approach for assignment and scheduling in high-level synthesis", in *Proc. of the 37th Midwest Symposium of Circuits and Systems*, Lafayette, LA, August 1994.

8 R. Mehra and J. Rabaey, "Behavioral level power estimation and exploration," *Proc. Int. Workshop on Low-Power Devices*, Napa, CA., pp. 197-202, April 24-27, 1994.

9 R. San Martin, "Optimizing power consumption, area, and delay in behavioral synthesis", Ph.D. thesis, Carleton University, March 1995.

10 A.P. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, and R.W. Brodersen, "Optimizing power using transformations", *IEEE Tran. on CAD of Integrated Circuits and Systems*, vol. 14, NO. 1, pp. 12-31, January 1995.

11 B. Arnold, "The long road to low-voltage systems," *ASIC and EDA*: *Technol. for System Design*, Verecom, pp.9-18, Oct. 1993.

12 D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Ma, 1989.