Yield Prediction for Architecture Exploration in Nanometer Technology Nodes: A Model and Case Study for Memory Organizations

A. Papanikolaou, T. Grabner, M. Miranda, P. Roussel and F. Catthoor
IMEC vzw, Kapeldreef 75, Leuven, Belgium

ABSTRACT

Process variability has a detrimental impact on the performance of memories and other system components, which can lead to parametric yield loss at the system level due to timing violations. Conventional yield models do not allow to accurately analyze this, at least not at the system level. In this paper we propose a technique to estimate this system level yield loss for a number of alternative memory organization implementations. This can aid the designer into making educated trade-offs at the architecture level between energy consumption and parametric timing yield by using memories from different available libraries with different energy/performance characteristics considering the impact of manufacturing variations. The accuracy of this technique is very high, an average error of less than 1% is reported, which enables an early exploration of the available options.

Categories and Subject Descriptors

B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids

General Terms

Design, Performance, Reliability

Keywords

Parametric yield, system exploration, process variability

1. INTRODUCTION

Embedded system that run real-time, power sensitive applications are becoming a very important part of the consumer electronic product market. System level design in the embedded systems domain has primarily been concerned with three main cost metrics: timing, energy consumption and area. Timing is important because embedded system typically run applications with hard real-time deadlines, such as communication services and multimedia applications. Minimizing energy consumption, on the other hand, can not only extend the time between battery recharges, but also enable new handheld applications and high-end mobile computing. A lot of research has been performed on how to minimize energy consumption, by employing run-time techniques such as voltage scaling for instance. These two metrics, together with area which has a direct proportional impact on cost, define the quality of the design. However, the main assumption so far at the system level design abstraction has been that timing and energy consumption of the individual system components and the final system implementation itself are deterministic and predictable. Limited variations have been tackled by embedding worst-case margins in the design of the system components, such as processors and memories, so that the specified performance and energy consumption can be guaranteed for use by the system designers.

Technology scaling past the 90nm technology node, however, introduces a lot more unpredictability in the timing and energy consumption of the designs due to random intra-die process variability. Treating these metrics at the system design level as deterministic values requires the design margins to become so large that they can eat up all the benefits of moving to a more advanced technology node. Therefore some degree of uncertainty will always have to be tolerated in the component. This has to be considered during circuit and even architecture design. It has to lead to new statistical design paradigms [3], such as statistical timing analysis or even more general yield-aware design [16, 12]. Depending on the component being considered (e.g. memory or datapath), energy and/or performance vs. area trade-off decisions have to be made [17]. However, for embedded system design the most critical trade-offs are not made at the component or IP block level, but at the architecture or even at the application level. Therefore solutions for (parametric) yield aware design have started being developed that tackle the problem at the architecture level while allowing some degree of uncertainty in the parametric energy and performance figures of the IP blocks [2, 5]. These solutions aim to tackle the system-level yield loss that is the result of timing violations due to the parametric drift in the performance of the individual system components caused by random process variability. Functional/catastrophic yield loss, due to manufacturing defects needs different solution techniques targeted at lower abstraction levels [6]. They are necessary to tackle the traditional yield issues and the proposed estimation technique for parametric yield is complementary to these. Functional and parametric yield issues are both crucial but anyway require different solution methods so it makes sense to attempt to solve them in a decoupled manner. In a realistic design environment both will be required.

The use of the solutions for system yield results in trade-offs on the main system metrics as noted above. Therefore, techniques and tools allowing to reason in terms of trade-offs between yield energy, performance and area at the architecture level are a must for successful embedded system design at the 65nm technology node and beyond. The ITRS road-map [13] discusses the need of such

CODES+ISSS'06, October 22–25, 2006, Seoul, Korea.

tools in its 2005 Design chapter and predicts that they will become mainstream for design in a few years. This paper is a step toward filling the gap in that direction. It proposes a new procedure of dealing with parametric yield loss at the system level.

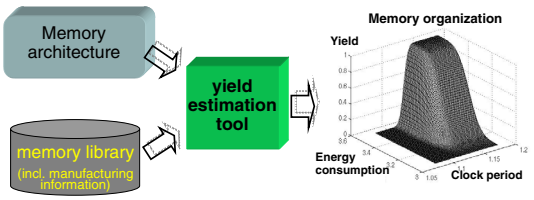

Figure 1: Estimating timing yield and average/worst-case energy for the system's memory organisation

The main requirement of such a tool is that the energy/delay distributions of the individual components must be a priori characterized using models of the manufacturing process of the target foundry. Obviously, the more accurate the characterization is, the better the estimation results will match reality. Indeed, this characterization or calibration task is tedious and not straightforward as it requires also to identify the actual critical path and critical input vector(s) of the IP-block for all relevant technology variation sources [15]. This task is being mostly performed today by IP providers to guarantee the correct functionality of the block irrespective of the manufacturing and operating conditions, but the results are made available only internally or at most to the design teams inside the big Integrated Design Manufactures [17]. Still, at lower level of abstraction the emerging statistical analysis techniques such as SSTA and alike [10] will enable in the meanwhile them to statistically characterize the critical IP blocks of the systems. This situation may be transitional until IP-providers will start distributing it as a natural trend within their current business model pushed by the need to avoid worst case design at nanometer technologies.

In this paper we propose a technique to estimate the system level timing yield and energy consumption under variability. We focus on random variations such as doping fluctuations and line edge roughness that lead to spatially uncorrelated drifts in the electrical characteristics of the devices. Our main architectural focus is the level-1 (L1) data memory organization, because in most embedded processing systems it becomes an energy/delay critical subsystem for any reasonable mapping of the architecture [18]. Memories belonging to the second layer of the memory hierarchy or above typically have more relaxed constraints on their performance requirements, larger cycle times, and they do not contribute that much to the system yield loss, which is dominated by the tight cycle time of the processing elements and the L1 memories. The latter need to respond in a single cycle and the impact of process variability on their performance can severely compromise the yield of the memory organization. Furthermore, typical L1 memory organization are heavily distributed due to the low-power operation requirements of the applications that run on the embedded devices. The increased number of memories increases the probability of timing violations reducing the system level yield. The technique presented in this paper is based on propagating the statistical properties of the energy consumption and the performance of the individual memories to the memory organization. Moreover, it is not limited to Gaussian statistical distributions for the individual memories. It can handle any type of distribution, which is a major assumption in most of the state of the art for individual gates. For complex IP blocks that assumption is no longer valid. Furthermore, one of the major advantages of our technique is that it can also estimate the timing yield of memory organizations that employ configuration options allowing run-time energy/delay trade-offs [5].

2. RELATED WORK

Design for Yield and Design for Manufacturing have become very popular research issues in recent years. The bulk of the work, however, has been concentrated either at the printing level using post processing techniques or at the standard cell level by characterizing the yield of particular layout styles [19, 20, 6]. Techniques of the first type such as Optical Proximity Correction aim to counteract the imperfections in the manufacturing process and to increase the device-level functional yield of the chips by improving the quality of the drawn features.

At higher abstraction levels, research has been focused on the gate-level abstraction level of the circuits. Statistical static timing analysis (SSTA) [4] aims at estimating the parametric performance of a circuit. More recent research has introduced circuit level timing optimizations [11] and combined power and timing optimization [12] on top of statistical timing analysis. Recently a gate level approach has been proposed which can accurately estimate the bivariate timing/leakage power distribution of a combinatorial circuit [10]. The correlations between leakage power and performance are taken into account and the resulting joint distribution at the circuit level is generated. The assumption made in this work is that the underlying gate distributions are Gaussian. Such a technique could be properly adapted to handle the yield estimation of a memory organization, if the memory level distributions are assumed to be Gaussian. It cannot handle the case of configurable memories/components, however, or the case of any kind of component with non-Gaussian energy/performance distribution. But system-level characterization (across architectural components) is not possible in SSTA due to the abstraction level used. In essence, our technique complements SSTA by providing the modeling framework to characterize the interactions between architecture components of the system, once SSTA has characterized the interactions between gates in a component.

The formulation we propose in this paper is a generic technique to estimate the yield at the system architecture level. It takes into account the correlations between dynamic energy and delay that exist in the individual architecture components, i.e. memories, and it can also handle the impact of configurable components on the timing parametric yield and the average and worst-case energy consumption of the system architecture. To the best of our knowledge this is the first attempt at creating such a technique.

3. MOTIVATIONAL EXAMPLE

Let's assume that a memory organization comprises three memories. Memory libraries nowadays allow the choice of high-speed or low-power memories [22, 23]. Since energy consumption is an important cost metric for embedded systems, memories from a low-energy library are chosen for use. Their timing and the energy consumption is described by two-dimensional statistical distributions, shown as clouds in Figure 2. They represent the possible energy/delay characteristics and their probability over a large number of fabricated memories. On the other hand, the designer of the system has a constraint on the target clock period of fabricated system. The left hand side of the figure shows the timing/energy statistical distributions of the memories in the initial composition of the memory organization and the target clock period. Two contours are illustrated per memory, the inner one ("1σ") surrounds about 63% of the population, while the outer one ("3σ") surrounds about 97%. This means that also a non-negligible proportion of the population lies outside the "3 sigma" contour. If the implementation of a given memory has an actual delay larger than the clock period, it

will introduce timing violations in the system and render the given chip non-functional. For instance, the middle memory in the figure clearly creates a lot of yield loss, due to the high probability of timing violations. Substituting it with a similar but faster memory (right top) decreases the probability of timing violations, but incurs significant energy overhead. Substituting it with a configurable memory (bottom right) decreases timing violations, with a much lower penalty in energy.

For the given clock period target, it is evident that after fabrication a significant percentage of the manufactured chips will have some memory violating the clock period. The probability that either the first or the third memory are slower than the constraint exists, even though it is low it must be quantified and considered. The second memory, however, has a large chance of failing to meet the constraint. As a result the final yield of the chip will be low. At this point the system design would need to be assessed in terms of what is the actual energy, performance and yield of the implemented system after manufacturing.

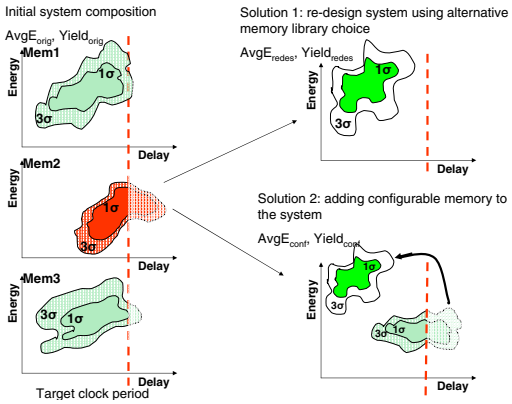

Figure 2: Case study on options to improve yield in the system's memory organisation

Architecture design alternative solutions, such as distributing further the memory organization by memory splitting [21] would lead to using smaller hence faster memories with a positive impact on yield but would require a more complex mapping process. A much less disruptive solution could be to use memories that are significantly faster than the clock constraint, so that the probability that they introduce timing violations is very small at the expense of energy overhead. In Figure 2 this corresponds to substituting the second memory with the one shown in the upper right hand side. The new implementation of the second memory is faster, thus yield is increased, but more energy hungry. Thus the designer can decide which cost metric is more important and make the appropriate memory instance selection on condition that a parametric yield prediction tool is available. This trade-off between energy consumption and yield is difficult to quantify without tool support, especially in the case of distributed heterogeneous architectures like those used for embedded system design. Obviously, substituting more memories than one will increase the number of different possible implementations with different yields and energy consumption.

A second solution scenario already introduced recently [5] to increase the memory organization yield without sacrificing too much energy is to substitute the slow second memory with a run-time configurable memory. They have at least two possible settings, high-speed and low-energy [14]. An embedded delay monitor measures the actual delay of the memory after manufacturing. If the low-energy setting is not fast enough to meet the clock period, then the memory is switched to its high-speed setting. As a result, energy consumption is only sacrificed if the actual memory delay after fabrication is slower than the constraint. This solution will have the same yield as the conventional solution of using only high-speed memories, because the memory has the back-up option of the high-speed setting. But, the average memory organization energy consumption will be somewhere between that of the other two design alternatives, since the configurable memory is not at its high-speed and energy hungry setting for all the chips. Note that this approach is generally applicable to any IP block with a run-time configuration knob, for example V_{dd}/V_{th} [1, 24] tuning in processing elements.

Finally, a third cost metric that is very important, mostly for financial reasons, is the area occupation of the design. Introducing a configurable memory implemented using configurable buffers [14] will introduce a small area penalty, typically about 5-15% over the area of the memory depending on its size. Different library implementations of the same memory can also have very different areas, due to the internal memory design choices [22, 23]. However, estimating area occupation of a design given the area of the components is relatively easy. Thus, we will not consider area further in this paper. Another complication of adding run-time configurable components is the additional design complexity. A feedback loop and monitoring mechanisms need to be added to the design to implement the configuration functionality [5]. In order to find the optimal trade-off for the particular design all these metrics should be considered.

4. ANALYSIS TECHNIQUE

In order to estimate the yield and the average energy consumption of the fabricated memory organization, two inputs are required, see Figure 1. The first is the output of system synthesis or architectural exploration, which defines the composition of the memory organization in terms of which memory types are going to be used and the data to memory assignment decision. The second input is a memory library which includes information about the statistical distributions of the energy and delay of the memories implemented in the target process of the product.

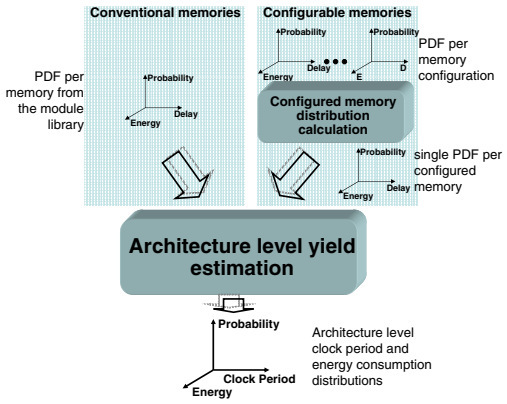

Figure 3: The two steps of the technique, first the post-configuration memory level distributions and then the architecture yield and energy consumption are estimated.

Given these inputs, the estimation of yield and energy comprises two steps, see Figure 3. The first step aims at obtaining the memory level energy and delay distributions. In case of conventional memories this is straightforward, because they are supplied by the memory library. In case of configurable memories (or any configurable IP block in general) though, special treatment is needed. The distribution of a configurable memory after it has been configured

depends on the clock period target. Furthermore, it is not a simple Gaussian distribution and as a result it requires a methodology that can handle any kind of distribution.

Once the memory level distributions are obtained, the memory organization level timing and energy distributions can be obtained. The delay of the memory organization is actually the maximum of the delays of all the memories it comprises and the energy consumption is a weighted sum of the energy consumption of each individual memory. The weights are the number of accesses that are performed to each memory.

4.1 Energy/delay statistical distributions of configurable components

At the level of individual memories, the energy/delay statistical distributions can be obtained via transistor-level simulations or measurements. We performed the characterization using transistor-level simulations [15]. They indicate that in the case of conventional memories energy and delay follow a Gaussian bivariate distribution with non-negligible correlations between them.

Performing similar simulations using configurable memories also indicates that each setting of the configurable memory follows a bivariate Gaussian distribution in energy and delay. The left hand side of Figure 4 shows the energy/delay bivariate probability density functions for a configurable 1KByte memory. The distributions of the two settings are easily identified and do not overlap significantly, barring extreme values.

The step of configuring such a memory to meet a given clock period constraint needs to be handled in a statistical manner as well. The memory is configured to its high-speed setting when the actual delay of its low-energy setting after fabrication fails to meet the clock period. This can be modeled as a sum of two probabilities, the probability that the actual sample of the low-energy distribution meets the clock multiplied by the conditional low-energy distribution and the probability that the actual low-energy sample does not meet the clock times the high-speed distribution. Performing such a mathematical manipulation on the bivariate density functions results in the following formulation for the bivariate distribution of a configured memory (the symbols in the formulae are explained in Table 1):

$$PDF_{conf}(t,e) = \tfrac{1}{2} erfc(\tfrac{T_m-\mu_{t_s}}{\sqrt{2}\sigma_{t_s}}) f_1(e) f_2(t) + f_3(e) f_4(t) UnitStep(T_m - t),$$

where the f_x functions are Gaussian univariate probability density functions with the following moments $f_1(e) = N(\mu_{e_f}, \sigma_{e_f})$, $f_2(t) = N(\mu_{t_f} + \frac{\rho_{te_f}\sigma_{t_f}(e-\mu_{e_f})}{\sigma_{e_f}}, \sigma_{t_f}\sqrt{1-\rho_{te_f}^2})$, $f_3(e) = N(\mu_{e_s}, \sigma_{e_s})$ and $f_4(t) = N(\mu_{t_s} + \frac{\rho_{te_s}\sigma_{t_s}(e-\mu_{e_s})}{\sigma_{e_s}}, \sigma_{t_s}\sqrt{1-\rho_{te_s}^2})$. $erfc$ is the complementary error function. The advantage of staying with a bivariate formulation after the memory configuration is that all the correlations between energy and delay are preserved. This bivariate distribution is illustrated in the right hand side of Figure 4. The effect of the configuration can be clearly seen where the low-energy distribution suddenly drops off when the memory delay becomes equal to the clock period constraint.

In order to evaluate the timing and the energy consumption of the entire memory organization, two different kinds of transformations are needed. Timing is obtained via a maximum operation, while energy consumption involves a weighted sum operation. Performing different transformations on the two axes of a bivariate distribution like the one in Figure 4 does not have a closed form solution unfortunately in the general case. As a result, we have to work with marginal probability density functions, which are the projections of the bivariate on the two axes.

Table 1: Table of symbols used in the statistical calculations

$N(\mu, \sigma)$	PDF of Gaussian distribution with mean μ and std. dev. σ
$N(\mu_1, \sigma_1, \mu_2, \sigma_2)$	PDF of bivariate Gaussian distr.
μ_{t_s}	mean delay of slow memory distr.
μ_{t_f}	mean delay of fast memory distr.
σ_{t_s}	std dev of delay of slow memory distr.
σ_{t_f}	std dev of delay of fast memory distr.
μ_{e_s}	mean energy of slow memory distr.
μ_{e_f}	mean energy of fast memory distr.
σ_{e_s}	std dev of energy of slow memory distr.
σ_{e_f}	std dev of energy of fast memory distr.
ρ_{te_s}	Energy/delay correlation in slow distr.
ρ_{te_f}	Energy/delay correlation in fast distr.
T_m	target clock period

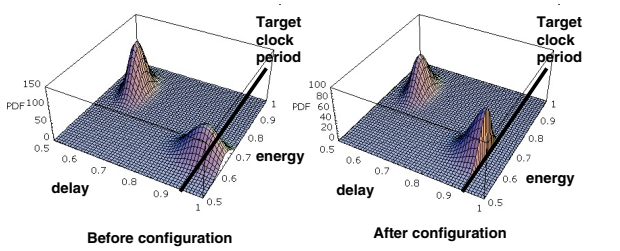

Figure 4: Illustration of the bivariate energy/delay probability density functions of the 1KByte memory before and after the configuration for the target clock period

The correlations that exist between energy and delay at the memory level are fully taken into account at the level of the memory organization. When doing transformations on statistical distributions, the correlations do not change as long as the transformations are monotonically increasing [8]. In our case, both the weighted sum and the maximum operation are increasing, because the energy consumption cannot be negative. So the correlations are not affected by the transformations. Since they are already taken into account in the memory level formulations, they are present in the final memory organization yield and energy consumption formulations.

4.2 Estimation of parametric system level yield

As discussed in previous sections parametric yield loss is incurred when at least one of the memories in the organization fails to meet the clock period. The reason is that if the memory cannot react in time, the wrong data is latched and the I/O behavior of the system is not respected. It is evident that in order to have a functional memory organization, the slowest memory needs to be faster than the clock.

Let's define M as the set of memories in the memory organization. The delay of each of them is random variable D_{m_i}. The delay of the memory organization is also a random variable D_{mo}, which is defined as the max of all the memory delays D_{m_i}.

An analytical formulation of the maximum of a number of probability density function of any type is given by the following formula, for an example of three distributions [25]:

$$f_{max}(t) = \int^t \int^t (f_1(t) f_2(t_2) f_3(t_3) + f_1(t_2) f_2(t) f_3(t_3) + f_1(t_3) f_2(t_2) f_3(t)) dt_2 dt_3$$

For each term in the sum inside the integral, a sample from a different PDF is assumed to be maximum. The first term, for instance implies that the sample from PDF f_1 is the maximum. The mul-

tiplication with the other two PDFs and the cyclic permutation of the PDF including the maximum makes sure all the possibilities are covered. The integration limits make sure that the "secondary" variables t_2 and t_3 never exceed t. Performing the integration we obtain the following compact formulation for discrete distributions: $PDF(D_{mo}) = \sum_{j=1}^{n}(PDF(D_{m_j}) \times \prod_{k \neq j}^{n} CDF(D_{m_k}))$

The reasoning used to obtain this maximum formulation does not depend on the underlying distributions used. It is also generic in the sense that it can handle any number of distributions. As long as the probability density functions of the different random variables are available either in analytical form or as experimental results, this formulation can be used to extract the maximum. In the second case however, the results need to be calculated numerically.

From the probability density function of the memory organization delay we can easily extract the system-level parametric yield, or the percentage of memory organizations that meet the clock period target constraints after fabrication.

4.3 Estimation of energy consumption

The second important cost metric for embedded systems is energy consumption. System designers need to be able to estimate the energy that their chip will consume in order to know if they meet the battery supply or the lifetime between recharging specifications.

The energy consumption of the complete memory organization is a sum of the energy consumption per access of each individual memory multiplied by the number of accesses to each memory. If we assume that E_{mo} is the random variable representing the energy consumption of the memory organization, E_{m_i} is the random variable of the energy consumption per access of memory i and acc_i is the deterministic number of accesses to memory i, then:

$$E_{mo} = \sum_{i=0}^{N} acc_i * E_{m_i},$$

where N is the number of memories in the organization. Such a weighted sum is a straightforward operation on statistical distribution only if they are Gaussian distributions. However, we need a more generic solution that can handle any kind of distribution.

Statistical addition is similar to convolution in the time domain, because each sample of one distribution needs to be added to all the samples of the other distributions. Convolution in the time-domain, however, is equivalent to multiplication in the frequency-domain [7]. The Fourier transform and its inverse can be used to move from the time to the frequency domain and vice versa. This solves the problem of adding random variables that have arbitrary distributions. The only disadvantage of this approach is that analytical closed-form formulae for the distributions we are treating are not available, thus the computations need to be done numerically. The multiplication with the access frequencies is straightforward and will be performed in the time domain before the Fourier transform. Summarizing the above, the probability density function of the energy consumption of the memory organization can be obtained by the following calculations:

$$PDF(E_{mo}) = F^{-1}(\prod F(acc_i * PDF(E_{m_i}))),$$

where the function F represents the Fourier transform and F^{-1} is the inverse Fourier transform.

5. EXPERIMENTAL SETUP

In order to model the impact of process variability on the delay and energy consumption of memories we have performed a number of transistor-level simulations for memories of different sizes using SPICE according to the characterization methodology proposed in [15] at the 65nm technology node. Each memory size has been simulated using a random injection of threshold voltage and beta drift in each of its transistors. A Monte Carlo loop of such simulations was performed to obtain the bivariate energy/delay probability density function for each memory. The 65nm predictive technology model from Berkeley [9] has been used, the interconnect parameters and the information about the transistor level variability were taken from the ITRS roadmap [13]. As illustrated in Figure 4 significant spreads exist in the delay and the dynamic energy consumption of memories under variability.

To evaluate the accuracy of the proposed methodology we have developed a system-level simulator that can evaluate the performance of the memory organization based on the statistical properties of the energy consumption and performance of the individual memories, as described in the previous paragraph. A Monte Carlo loop has been implemented at the level of the memory organization to obtain the bivariate energy consumption and performance statistical distribution. Random points are selected for the energy/delay of each of the memories (based on the characterization results) for each of the simulation runs. 2000 memory organization level simulation runs are performed for the three design alternatives. The resulting distribution is compared to the results of the yield estimation methodology. Comparing to system-level results directly from SPICE is impossible, because SPICE needs about 10 hours to simulate a single complete memory. Going to Monte Carlo simulations of a complete memory organization is infeasible, which highlights the usefulness of the proposed technique.

The three application drivers we have used are multimedia (image processing and audio decoding) and wireless applications, which are representative of the domain we are targeting. The OFDM front-end and error correction application performs the digital baseband signal processing for a digital wireless receiver. The low-power mapping of this application results in a distributed heterogeneous local memory layer comprising 8 memories with sizes ranging from 1 to 8 KByte. The Cavity Detection (CavD) is an image processing application that detects cavities in images, by repeatedly filtering the input image. Its memory organization is distributed but homogeneous, consisting of five nominally identical 8KByte memories. The third application we have used is the MP3 decoder. Mapping it on a low-power memory organization results in a distributed organization, which has only two different memory types. Thus it is a hybrid between the very heterogeneous memory organization of the OFDM and the homogeneous organization of the Cavity Detection. These three application drivers will enable us to evaluate the impact on the system level parametric yield on different kinds of low-power memory organization architectures.

6. RESULTS

The comparison between the results obtained from the simulations and the ones calculated using the proposed methodology are presented in Table 2. The first column reports the estimated timing yield of the memory organization for the clock period that meets the real-time constraint of the respective application. The second and third columns report the estimated average energy of the memory organization that meet the timing constraint and the estimated worst-case energy consumption or the 99.9% energy yield. The next three columns report the same results coming from the simulation of the memory organization. The rows represent the three different design options shown in Figure 2 for each of the three applications.

The accuracy of the proposed yield estimation methodology is very good compared to the simulations, the estimation error has an average value of less than 0.5%. Furthermore, this error seems to be independent of the type of memory organizations. No matter if it is heterogeneous or homogeneous, if it is very distributed or not, the estimation error is very small.

Furthermore, comparing the results for the three different design choices for the OFDM application in Table 2 we can see that the yield is significantly improved by using high-speed (5th row) or

Table 2: Comparison of yield, average and worst-case energy consumption of the different choices illustrated in Fig. 2 for the three applications. The yield results are rounded at the second decimal digit. The energy consumption results are normalized to the simulated average energy using low-power memories.

	estimation			simulation			% error		
	timing yield(%)	average energy	worst-case energy	timing yield(%)	average energy	worst-case energy	timing yield(%)	average energy	worst-case energy
CavD original	51.27	1	1.028	51.1	1.0009	1.027	0.33	9e-4	1e-3
CavD re-design	99.99	1.34	1.396	99.99	1.34	1.391	1e-5	8e-6	0.33
CavD conf	99.99	1.043	1.286	99.99	1.042	1.309	1e-5	0.03	1.78
OFDM original	92.52	1	1.108	90.85	0.9999	1.114	1.8	4e-5	0.5
OFDM re-design	99.99	1.35	1.496	99.99	1.352	1.505	1e-5	2e-5	0.6
OFDM conf	99.99	1.003	1.202	99.99	1.004	1.205	1e-5	7e-4	0.3
MP3 original	95.71	1	1.119	94.8	1.001	1.142	0.95	0.1	2
MP3 re-design	99.21	1.339	1.53	99	1.337	1.54	0.21	0.16	0.64
MP3 conf	99.23	1.006	1.292	99	1.008	1.294	0.2	0.22	0.19

configurable memories (6th row) compared to the initial low power implementation (4th row). The difference between the energy consumption is also very large, the use of configurable memories enables the system to minimize the energy overhead invested for the yield optimization. For the Cavity Detection application things are very different. The memory organization is homogeneous, so any of the 5 memories is a potential source of yield loss. Thus all the memories have to be substituted by high-speed or configurable memories. This incurs a significant penalty in energy consumption in the case of high-speed memories, but the increase in yield is very large. The difference between the energy penalty of the configurable compared to the high-speed memories is evident in this application too. The use of high-speed memories increases the energy consumption by about 34% for processing one frame. The use of configurable memories can significantly reduce the total energy consumption, reducing the penalty to just 4.3% for the same yield. But a slight area overhead will exist. Similar conclusions apply for the MP3 application.

It becomes clear that the yield/energy trade-offs that constitute the optimal memory organization depend not only on the available memory implementations, but also heavily on the set of applications that will be running on that architecture. For instance, memories that are seldom used will benefit from a high-speed implementation due to the lack of area overhead. These kinds of trade-offs justify the usefulness of such a technique/tool for system designers. It is very difficult for a designer to make the proper decision, especially in the case of complex system architectures, without CAD support.

Designers nowadays do not have a way to quantify these yield vs. energy consumption trade-offs and are forced to take worst-case margins to maximize yield. A CAD tool that could quantify these trade-offs would aid the system designer in removing unnecessary margins and in designing a more efficient system.

7. CONCLUSIONS

This paper presents a methodology to accurately estimate and quantify the yield vs. energy consumption trade-offs that can be obtained by using different implementations of memories coming from different libraries during the design of the memory organization. This can aid the system designer into reducing unnecessary design margins that are currently used. The accuracy of the estimation methodology is very large, an average error of less than 1% is reported for each type of memory organization or metric.

8. REFERENCES

[1] A. Srivastava et al, "Concurrent Sizing, Vdd and Vth Assignment for low power design", DATE, 2004.

[2] T. Austin et al., "Making typical silicon matter with Razor", IEEE Computer, March 2004

[3] S. Borkar, "Designing reliable systems from unreliable components: the challenges of transistor variability and degradation", IEEE Micro, Nov 2005.

[4] C. Viswewariah, "Death, taxes and failing chips", DAC, 2003.

[5] A. Papanikolaou et al., "A System-Level Methodology for Fully Compensating Process Variability Impact of Memory Organizations in Periodic Applications", CODES+ISSS, 2005.

[6] P. Gupta, A. Kahng, "Manufacturing-aware physical design", ICCAD, 2003.

[7] A. Papoulis, "The Fourier integral and its applications", McGraw-Hill, 1962.

[8] C. Genest, A.C. Favre, "Everything you always wanted to know about copula modeling but were afraid to ask",Journal of Hydrologic Engineering, 11, 2006

[9] Y. Cao et al,"New paradigm of predictive MOSFET and interconnect modeling for early circuit design", CICC, 2000.

[10] A. Sriristava et al.. "Accurate and efficient gate-level parametric yield estimation considering correlated variations in leakage power and performance", DAC, 2005.

[11] A. Srivastava et al, "Statistical optimization of leakage power considering process variations using dual-V_{th} and sizing", DAC, 2004.

[12] M.Mani et al., "An efficient algorithm for statistical minimization of total power under timing yield constraints", DAC, 2005.

[13] International Technology Roadmap for Semiconductors, 2005 edition, http://public.itrs.net

[14] H. Wang et al., "Variable Tapered Pareto Buffer Design and Implementation Techniques Allowing Run-Time Configuration for Low Power Embedded SRAMs", IEEE Trans. on VLSI, Oct. 2005

[15] H.Wang et al., "Impact of deep submicron (DSM) process variation effects in SRAM design", DATE, 2005.

[16] A. Agarwal et al., "Process variation in embedded memories: failure analysis and variation aware architecture" IEEE Journal of Solid-State Circuits, Sept. 2005.

[17] C. Visweswariah, "Statistical Timing of Digital Integrated Circuits", Microprocessor Circuit Design Forum at ISSCC 2004.

[18] F. Catthoor et al., Custom memory management methodology exploration of memory organization for embedded multimedia system design, Kluwer, June 1998.

[19] J. Yang et al., "Advanced timing analysis based on post-OPC extraction of critical dimensions" DAC, 2005.

[20] PDF Solutions Inc. http://www.pdf.com/

[21] K. Patel et al., "Synthesis of partitioned shared memory architectures for energy-sufficient multi-processor SoC" DATE, 2004.

[22] Artisan Memories http://www.artisan.com/

[23] Virage Logic http://www.viragelogic.com/

[24] L.Benini et al, "System-level power optimization techniques and tools", ACM Trans. on Design Automation for Embedded Systems, April 2000.

[25] Mathematica http://www.wolfram.com/