A New Parameterizable Power Macro-Model for Datapath Components

Gerd Jochens, Lars Kruse, Eike Schmidt, Wolfgang Nebel

OFFIS Research Institute, Oldenburg
Jochens@OFFIS.Uni-Oldenburg.DE

Abstract

We propose a novel power macro-model which is based on the Hamming-distance of two consecutive input vectors and additional information on the module structure. The model is parameterizable in terms of input bit-widths and can be applied to a wide variety of datapath components. The good trade-off between estimation accuracy, model complexity and flexibility makes the model attractive for power analysis and optimization tasks on a high level of abstraction. Furthermore, a new approach is presented, that allows to calculate the average Hamming-distance distribution of an input data stream. It will be demonstrated, that the application of Hamming-distance distributions, instead of only average values, improves the estimation accuracy for a number of typical DSP-modules and data streams.

1. Introduction

The application of the Hamming-distance of two consecutive input-vectors (*Hd*) as a relative measure for power is common in the area of high-level synthesis for low power [5,6,7,8]. Usually, it is assumed that larger Hamming-distances result in an increased power consumption of the stimulated modules. As a consequence, a number of optimization methodologies focus on minimizing the average switching activity or Hamming-distance of data streams which are provided to design components.

In this paper we present a new parameterizable power macro-model that relates the Hamming-distance to quantitative power figures and therefore allows more detailed power evaluation and optimization tasks[1]. The model is applicable for a wide variety of typical datapath components and fits very well to existing statistical data models and simulation approaches [2,3,9,10]. Estimation results for a number of different modules and input pattern streams are presented; modeling properties and limitations are pointed out.

Furthermore, a new approach is presented that allows to calculate the average Hamming-distance *distribution* of an input data stream. It will be demonstrated that the application of the Hamming-distance distribution, instead of only an average value, will increase estimation accuracy in cases were the distribution is not symmetric and power has a non-linear dependency on the Hamming-distance. These are typical conditions for a number of DSP-data-streams and DSP-modules. Since the combination of the proposed power model and approach for calculating the data dependent model parameters enables fast and flexible power analysis tasks, it is well suited for optimizations of realistic designs on an early stage of the design process.

The remainder of this paper is organized as follows. Section 2 describes previous attempts at power macro-modeling. In section 3 our new power model is presented. Section 4 explains the characterization process and presents evaluation results. In Section 5 we focus on the handling of modules which are parameterizable in terms of the input bit-width. Section 6 describes our approach for determining the model parameters based on statistical data models. The paper concludes with a brief summary.

2. Previous work

Already, a number of macro-models exists which are applicable for power modeling of combinational modules. A good overview of existing approaches for power macro-modeling is given in [1]. In most approaches the assumption is made, that *input patterns* are 'ideal' or may be regarded as 'ideal', which means that:

- possible transitions per bit are 0-0, 0-1, 1-0, 1-1;
- only one transition per cycle and per bit is possible;
- transition characteristics (slopes, starting time, etc.) are equal for transitions of the same type.

Under these assumptions, the number of possible input transitions N is 4^m, for a module with m input-bits. Therefore the power consumption of a module can be exactly modeled by 4^m corresponding power coefficients. Estimation and storing of all necessary coefficients is only possible for small modules. Due to complexity reasons,

[1] This work is partially founded by the BMBF project EURIPIDES under grant number 01M3036G.

the aim is therefore to develop more abstract and simpler models while sacrificing some accuracy.

Unfortunately, most of the existing power models are not parameterizable in terms of the bit width which is a major requirement on a modeling approach for datapath components. Only the DBT-model [2,3] fulfils this requirement, while assuming some word-level statistics that can not always be assured in digital circuits [11].

A number of publications also exists which focus on the problem of estimating and propagating bit-level statistics in terms of word-level statistics [2,3,9,10]. These approaches are based on data-models that (for a class of data streams) can be applied to calculate the average Hamming-distance or Hamming-distance distribution, as will be shown in chapter 6.

3. The Hd power-model

The basic idea of our model is to describe the power consumption of a module as a function of the *Hamming-distance Hd of two consecutive input bit vectors **u** and **v** of length m.* The Hamming-distance *Hd* is defined as:

$$Hd(\mathbf{u},\mathbf{v}) = \left|\{i|(u_i \neq v_i)\}\right| \qquad \text{for } 1 \le i \le m \qquad (1)$$

and delivers the number of different bits of ***u*** and ***v***.

If transitions at the primary inputs are assumed as 'ideal' transitions, as defined in the previous section, *m* different classes of switching events E_i $(1 \le i \le m)$ can be distinguished for a module with *m* input bits, according to the Hamming-distance of the corresponding consecutive input vectors. To every switching event class E_i a power coefficient p_i which is determined from lower level characterization runs and an activator δ_i is assigned. δ_i takes the value '1', if an event of the corresponding class occurred in a cycle, otherwise it takes the value '0'. The *cycle charge consumption*[1] $Q[j]$ is then estimated as follows:

$$Q[j] = \begin{bmatrix} p_1 & \dots & p_m \end{bmatrix} \cdot \begin{bmatrix} \delta_1 \\ \dots \\ \delta_m \end{bmatrix} = P^T \cdot \Delta \qquad (2)$$

with:

P : vector of model coefficients p_i that represent the average charge consumption of type E_i transitions,

Δ : vector of 'activators' δ_i with

$$\delta_i = \begin{cases} 1 \text{ , if } E_i \text{ has occurred in cycle } j \\ 0 \text{ , if } E_i \text{ has not occurred in cycle } j \end{cases}$$

The model can be enhanced by increasing the number of switching event classes, if it is necessary to improve accuracy and/or robustness against data statistic changes. Enhancement is possible by considering word level statistics or additional bit level information. As an example, in this paper we present an enhanced model that considers the number of stable zero bits, as a further criterion to distinguish types of switching events.

[1] Power and charge consumption only differ by a constant factor for a given time period and are applied synonymously in this paper.

As a consequence of applying this additional criterion, the switching event class E_1 that is related to consecutive input vectors with a Hamming-distance of one, is divided into *m* subgroups

$$\{ E_{1,m-1} \quad E_{1,m-2} \quad \dots \quad E_{1,0} \}$$

while the first index gives the Hamming-distance, and the second the number of stable zeros. Event class E_2 is split in *m-1* subgroups

$$\{ E_{2,m-2} \quad E_{2,m-3} \quad \dots \quad E_{2,0} \}$$

and so on. Equation 2 accordingly changes to

$$Q[i] = \begin{bmatrix} p_{1,m-1} & \dots & p_{m,0} \end{bmatrix} \cdot \begin{bmatrix} \delta_{1,m-1} \\ \dots \\ \delta_{m,0} \end{bmatrix} = P^T \cdot \Delta \qquad (3)$$

and the number of coefficients *M* is increased to

$$M = \frac{m^2 + m}{2}$$

Since for modules with a high input bit-width the number of coefficients may be too large, it is also possible to cluster event classes within a certain range of the number of zeros.

4. Model evaluation

We split our model evaluation into two parts. In the first part we describe the characterization process and give some further information concerning the model coefficients. In the second part we focus on accuracy and robustness of the Hd-model. In addition, modeling properties and limitations are pointed out.

4.1. Model characterization and model coefficients

The model coefficients have to be determined once within a characterization step in which module prototypes are stimulated by characterization patterns. Gate- or transistor-level power simulations can be used to ascertain the charge consumption per transition. Based on this simulation results the model coefficients p_i are determined by simply calculating the average charge consumption for transitions that have a Hamming-distance of *i*:

$$p_i = \frac{1}{n} \cdot \sum_{j=1}^{n} Q_i[j] \qquad (4)$$

An average absolute deviation or error ε_i can be calculated by:

$$\varepsilon_i = \frac{1}{n} \cdot \sum_{j=1}^{n} \left| \frac{Q_i[j] - p_i}{p_i} \right| \qquad (5)$$

with:

n: number of transitions in the event class E_i ,

$Q_i[j]$: charge consumption of the j-th transition (element) in event class E_i ,
p_i: coefficient, that represents the average charge consumption for type E_i transitions.
The characterization can be finished after the coefficient values have converged.

Figure 1 presents the model coefficients p_i and the corresponding average deviations ε_i as errorbars for the 16 input-bit prototypes of some analyzed modules from the SYNOPSYS ™ DesignWare library. The p_i's were determined within a characterization task, in which the modules were stimulated by an input sequence of random patterns and the power consumption for each transition was computed with PowerMill.

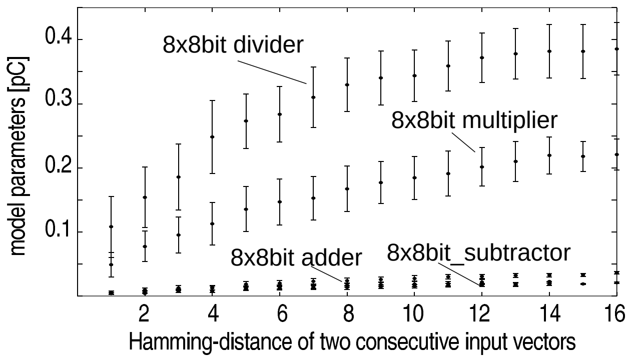


Figure 1: Coefficients p_i for 16 bit input bit variants of some analysed modules.

It is obviously that the Hamming-distance is a good criterion to separate the different types of transitions in terms of power. For most cases we have found total average coefficient deviation $\varepsilon = 1/m \cdot \sum_{i=1}^{m} \varepsilon_i$ of less than 15% which is a good result for the small number of model parameters and acceptable compared to other inaccuracies that have to be considered on higher-levels of abstraction. In addition, it has to be noted that the relative coefficient deviations are decreasing for larger values of the Hamming-distance.

One possibility to enhance the model is to consider the number of stable zero (and/or one) bits as a further separation criterion, as described in section 3. Figure 2 shows the effects of such an enhancement step, for an 8x8 bit csa-multiplier. The dotted lines are the coefficients for the basic Hd-model. The solid lines present coefficients of the enhanced model for the cases where none or all not-switching bits are zero.

The figure illustrates that the resolution of the power model is enhanced, especially for small numbers of i. The average coefficient deviations are decreased and the robustness is increased, due to the enhanced resolution of the model. From the figure it is also obviously that using the basic model parameters for an input stream which has a large number of bits that are constant 1 or 0, would lead to a systematic under- or over-estimation.

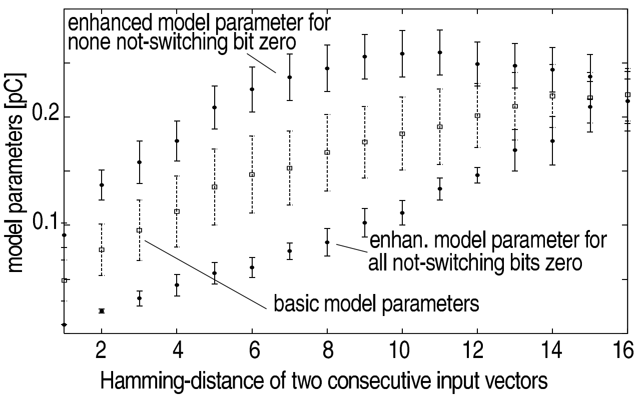


Figure 2: Comparison of basic and enhanced Hd-model parameters for a 8x8 bit csa-multiplier.

4.2. Model accuracy and robustness

The model presented so far can be applied to estimate the power consumption of modules with a fixed input bit-width. To assess the estimation accuracy of our approach comparisons to PowerMill simulations were done for a number of module types and input patterns. Several different sets of data streams, each consisting of 5000 to 10000 input patterns, were generated and applied as input stimuli to analyze the *robustness* of the model against changes of input pattern statistics. The pattern-sets can be classified into:

I) random patterns (statistics as characterization stream),
II) linear quantized music signals (weak correlation),
III) linear quantized speech signals (strong correlation),
IV) video signals (strong correlation),
V) outputs of a binary counter.

Table 1 presents the estimation errors of our basic model in comparison to PowerMill simulations for a number of different module types and input patterns. For the cycle charge consumption the average absolute estimation error ε_a is given, which can be calculated from:

$$\varepsilon_a = \frac{1}{n} \cdot \sum_{j=1}^{n} \left| \frac{Q_{\text{Hd-model}}[j] - Q_{\text{PowerMill}}[j]}{Q_{\text{PowerMill}}[j]} \right| \cdot 100\%$$

For comparisons of the average charge consumption we use the average error, which is defined as:

$$\varepsilon = \frac{\sum_{j=1}^{n} Q_{\text{Hd-model}}[j] - \sum_{j=1}^{n} Q_{\text{PowerMill}}[j]}{\sum_{j=1}^{n} Q_{\text{PowerMill}}[j]} \cdot 100\%$$

with:
n: number of input patterns of the test run,
$Q_{\text{Hd-model}}[j]$: charge consumption in cycle j estimated by Hd power model,
$Q_{\text{PowerMill}}[j]$: charge consumption in cycle j estimated by PowerMill simulation.

From table 1 it becomes clear that significant estimation errors must be expected if the model is used for cycle power estimations. On the other hand, adequate estimation

module type	input bit/ width	cycle charge consumption					avg. charge consumption				
		data types					data type				
		I	II	III	IV	V	I	II	III	IV	V
ripple adder	8	12	33	35	32	44	3	3	7	2	12
	12	7	29	28	36	39	1	3	11	7	19
	16	14	30	46	31	68	2	1	14	5	31
cla-adder	8	9	25	27	22	38	1	6	7	14	13
	12	17	22	35	24	41	1	3	2	10	9
	16	12	19	29	35	58	1	2	12	9	14
absval	8	10	33	21	24	41	2	5	4	6	13
	12	24	27	24	31	40	1	3	9	6	12
	16	23	22	28	33	44	1	7	13	10	15
csa-mul-tiplier	8	28	27	25	29	43	1	3	10	8	23
	12	18	32	23	22	52	1	5	8	8	23
	16	14	30	34	38	62	2	6	14	6	34
booth-cod. wal-lace-tree mult.	8	18	21	45	37	34	4	1	6	12	19
	12	12	25	23	41	37	1	3	11	10	21
	16	34	16	29	44	58	3	7	13	16	24
average	/	17	26	30	32	47	2	4	9	9	18

Table 1: Estimation error of the Hd-model (in %).

results are delivered for average power analyses, considering the relatively small number of model parameters. Trends in the power consumption, e.g. a decrease or increase in power caused by changes of the input statistics, are followed very well by a model, too. This makes the model attractive for power analyses and optimization tasks on a high level of abstraction.

For most cases where the modules have been stimulated with real application data we have found average estimation errors below 15% to 20%. Nevertheless, for input patterns which strongly differ from the characterization patterns errors might be larger (i.e. data type V). In these cases, coefficient adaptation techniques [4] or the application of the enhanced Hd-model is proposed. Table 2 shows the improvement in accuracy by using the enhanced Hd-model for a csa-multiplier (cf. figure 3). Especially for the data type V (binary counter) a large accuracy improvement can be achieved, as in this data stream only positive values (sign-bits are zero) are used for stimulation. This leads to the significant estimation errors for the basic model (c.f. section 4.1).

data types	cycle charge avg. abs. error [%]		average charge error [%]	
	basic Hd-model	enhanced Hd-model	basic Hd-model	enhanced Hd-model
I	28	14	1	0.11
III	25	18	10	7
V	43	42	23	7

Table 2: Comparison of basic and enhanced Hd-model for a csa-multiplier.

5. Parameterizable modules

Our approach of describing the influence of input bit-width changes on the model coefficients p_i is comparable to the approach of describing the module capacitance dependency on the bit-width as suggested in [2,3]. The idea is to consider the dependency of the module complexity on the input bit-width.

We will exemplify the principles and properties of the approach for two module types: (1) ripple adder and (2) carry-save-array multiplier. From the structure of a ripple adder it is clear, that the complexity scales linearly with the input bit-width *m*. Therefore a linear regression function of the following form is used for coefficient calculation:

$$p_i[m] = r_{i,1} \cdot m + r_{i,0} = \begin{bmatrix} r_{i,1} & r_{i,0} \end{bmatrix} \cdot \begin{bmatrix} m \\ 1 \end{bmatrix} \qquad (6)$$

Figure 3 shows the structure of a csa-multiplier. If the input bit-widths are equal ($m_1 = m_2 = m$), the complexity of the multiplication array scales with m^2, and the complexity of the adder part scales with *m*. Therefore we apply a regression function which contains corresponding terms:

$$p_i[m] = r_{i,2} \cdot m^2 + r_{i,1} \cdot m + r_{i,0} \qquad (7)$$

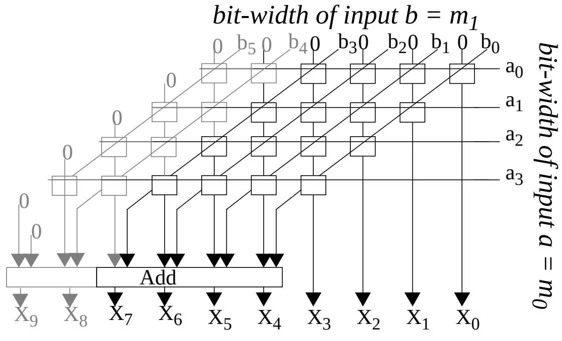
If the input bit-widths m_1 and m_2 differ, the parameters can be calculated from:

$$p_i[m_1, m_0] = r_{i,2} \cdot (m_1 \cdot m_0) + r_{i,1} \cdot (m_1) + r_{i,0} \qquad (8)$$

More generally, these relationships can be written as:

$$p_i = R_i^T \cdot M, \qquad (9)$$

with:
R_i : vector of regression coefficients,
M : vector of complexity parameters.



bit-width of input b = m_1
0 b_5 0 b_4 0 b_3 0 b_2 0 b_1 0 b_0
a_0
a_1
a_2
a_3
bit-width of input a = m_0
Add
X_9 X_8 X_7 X_6 X_5 X_4 X_3 X_2 X_1 X_0

Figure 3: Structural differences between a 4x4bit and a 6x4bit csa-multiplier.

Least mean square regression is applied to determine the vector elements $r_{i,j}$ of R_i, based on the coefficients p_i^{inst} that have been calculated for some module prototypes (instances) with different input bit-widths m^{inst}. The modules which are used for regression are named the *prototype set*. The number of modules in the prototype set is called the *prototype set-size* in the following.

With this, the following matrix equation can be solved for the vector R, that minimizes the modeling error ε:

$$P_i^{inst} = R^T \cdot M^{inst} + \varepsilon \qquad (10)$$

where P_{inst} is the vector of instance coefficients, and M^{inst} the vector of the corresponding complexity parameters.

The influence of varying the prototype set-sizes on the regression accuracy was analyzed by the following procedure: A complete set of prototypes (ALL) with input bit-widths m from 4 to 16 in steps of 2 was generated and characterized. Afterwards, by reducing the number of prototypes two sub-sets of prototypes were built:
SEC: only every second prototype was applied (e.g. coefficients for 4,8,12 and 16 bit module variants),
THI: only every third prototype was applied (e.g. coefficients for 4, 10,16 bit module variants).

For each set a regression task was carried out to determine the corresponding regression vector $R^{\{\mathrm{set}\}}$. Applying these, the coefficients $p_i(R^{\{set\}})$ were calculated from equation 9 and compared to the corresponding coefficients p_i^{inst} which were determined from instance characterization.

The analyses have shown that the differences between the instance coefficients p_i^{inst} and the corresponding coefficients which come out of the regression equation $p_i(R^{\{set\}})$ are small (less than 5% to 10% in most cases), even if the number of prototypes is strongly reduced. This is because the assumed functional dependency of the model coefficients which considers the module complexity fits very well to real dependency. Figure 4 exemplifies the results for some p_i's of the csa-multiplier and ripple-adder.

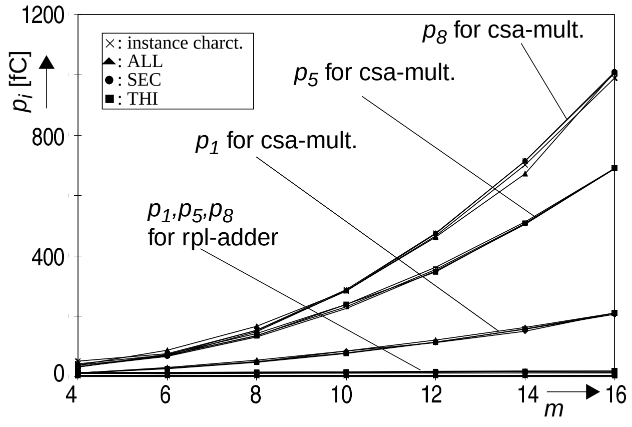


Figure 4: Comparison of coefficients from instance characterizations and from regression equations.

Since the differences between the coefficients from the instance characterization and the regression formula are small, the effects on the model accuracy are also small. Table 3 shows the effects of coefficient errors on the estimation accuracy for a 8x8-bit csa-multiplier and 8-bit ripple-adder, for different input stimuli (see also table 1). Columns 3 to 5 contain the relative difference of the parameters p_1, p_5, p_8 from regression and the ones from the instance characterization. Column 6 contains the relative difference averaged over all coefficients. Columns 7 to 9 contain the according estimation errors for the average power.

Module	Hd-model parameters from		parameter errors for				estim. errors for avg. power for data type		
			p_1	p_5	p_8	avg(p_i)	I	III	V
csa-mult.	inst. charact.		0	0	0	0	1	10	23
	regression	ALL	1	0	2	2	3	10	27
		SEC	1	1	1	4	1	15	29
		THI	5	2	4	4	1	7	24
rpl-adder	inst. charact.		0	0	0	0	1	11	19
	regression	ALL	1	2	5	5	5	9	22
		SEC	5	3	5	3	3	10	24
		THI	0	7	1	5	3	14	24

Table 3: Comparison of coefficient and estimation errors (in %) for different regression tasks.

In conclusion, it can be stated that the complexity functions of modules are well suited for building regression functions for the coefficients p_i. Good regression results and a high confidence in estimated coefficients can be expected, even if the number of prototypes which are applied in the regression process is small.

6. Calculation of model parameters

In this section we focus on the problem of calculating the data dependent model parameters. Until now, we have assumed that the Hamming-distance information is available at all module inputs, e.g. from a bit-accurate, cycle-wise functional simulation of the circuit. While the advantage of such an approach is the high parameter accuracy, one disadvantage is the low performance because of the time-intensive simulation and information capturing. Word-level functional simulation and especially probabilistic simulation may help to overcome this problem [2,3,9,10] for a class of typical DSP-designs, while sacrificing accuracy for performance. To apply these simulation techniques for power estimation and optimization, models and methodologies are required which efficiently allow to calculate the bit-level statistics from word-level statistics.

In the following we present a novel approach which allows to calculate the average Hamming-distance *distribution* of an input data-stream. We will demonstrate that using the Hamming-distance distribution for power estimation instead of a simple average will increase the estimation accuracy for a number of typical DSP-data-streams and components. We will start with a brief description of the data model on which our approach is based. This is followed by the description of techniques to calculate the average Hamming-distance. Finally our approach for calculating the average Hamming-distance distribution is presented and exemplified for some input-signals.

6.1. Data modeling

Landman first noted that for typical DSP data-streams, bit-level statistics can be brought into relation with word-level statistics [2,3]. He showed that a data word can be separated into three regions (cf. figure 5), where data bits:

1) are uncorrelated in space and time,
2) are correlated,
3) can be interpreted as sign-bits.

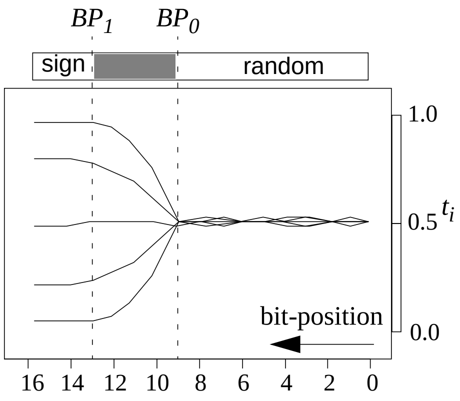


Figure 5: Regions of a data word on bit-level.

The uncorrelated bits are within a region from the LSB to a certain bit-position BP_0 and have a signal probability p_i and transition probability t_i of 1/2, not depending on the word-level data statistics. The sign-bits lie within a region from the MSB down to another break point BP_1 and have signal and transition probabilities which are strongly depending on the word-level data statistics. For the calculation of the transition probabilities of the bits between the sign and the uncorrelated data bits he proposed a linear approximation.

Assuming that input streams are closely approximated by Gaussian processes, he presented some empirical equations for calculating BP_0 and BP_1 in terms of word-level statistics such as mean (μ), variance (σ^2) and autocorrelation (ρ).

Having developed a model relating word-level to bit-level characteristics, in [9] he presented a technique for propagating the word-level parameters μ, σ^2, ρ through a design. He exemplified that for constant-multipliers and adders the statistics of the output can be calculated from the input statistics, efficiently.

In [10] the topic has been taken up again and improved methodologies for break point calculation have been presented which are more accurate and allow the handling of different number representations. Furthermore, the technique for propagation of word-level statistics was improved to also handle multiplexer and delay.

6.2. Calculation of average Hamming-distance

The presented techniques for break point calculation and transition activity estimation of bit groups can be used to calculate the average Hamming-distance for a data-stream:

$$Hd^{avg} = (t_{rand}n_{rand} + t_{sign}n_{sign} + t_{corr}n_{corr}) \quad (11)$$

In this formula the t's give the average switching activity of data bits in different regions, and the n's the number of bits within those regions which can all be calculated based on the formula set presented in [2,3] or [10].

The application of the average Hamming-distance is well suited for cases where power is a linear function of the Hamming-distance and the distribution of the Hamming-distance is symmetric. Since the values of Hd^{avg} are real numbers, for calculating the power consumption of a module based on the Hd-models it is necessary to interpolate between the coefficient values p_i. This can be done using standard interpolation techniques.

Using only the average Hamming-distance Hd^{avg} may result in significant errors if the power is a non-linear function of the Hamming-distance and the distribution is not symmetric which is a typical case for a number DSP-modules and -data streams.

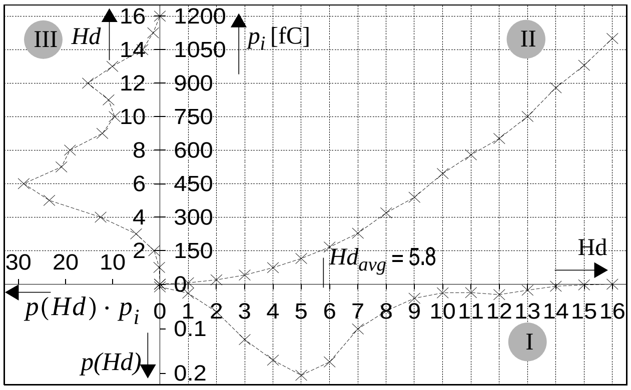


Figure 6: Estimation errors caused by the application of average Hd instead of a distribution.

Figure 6 exemplifies the problem for a field multiplier which is stimulated by an audio signal. Field I contains the probabilities of the possible switching events (Hd-distribution), field II presents the model coefficients versus the possible Hamming-distances and field III the product of the probabilities and corresponding power model coefficients. The average power comes from the summation of the particular power values in field III. Only applying the average Hamming-distance for power calculation without assuming a distribution (e.g. $p(Hd{=}Hd^{avg}) = 1$) would result in an additional error of about 30%, for the given example. As a consequence, efficient approaches for calculating the Hamming-distance distribution are demanded.

In the following we present such an approach which is based on data models introduced in [2,3] or [10] and, in combination with the proposed Hd-power model, can be applied for fast power optimization tasks.

6.3. Calculation of Hd-distribution

First, the number of regions of a data word is reduced (cf. figure 5). Instead of applying a linear approximation for the transition activity of bits in the intermediate region, shifting together the break points by half of the number of intermediate bits will result in the same average transition activity. As a result, only a random part of n_{rand} bits and a sign part of n_{sign} bits have to be distinguished, with:

$$n_{rand} = BP_0 + \frac{BP_1 - BP_0}{2}$$

$$n_{sign} = m + 1 - BP_1 + \frac{BP_1 - BP_0}{2}$$

From the data model it is clear, that the bits in the *uncorrelated region* have a transition activity of 0.5. The Hamming-distance distribution that results from the switching in the uncorrelated region is binomial, so that the probabilities for the occurrence of switching events with a Hamming-distance of *i* can be calculated as:

$$p_i^{rand} = p(Hd^{rand} = i) = \binom{n_{rand}}{i} 0.5^{n_{rand}} \qquad (12)$$

The bits in the *sign region* have a transition activity which strongly depends on the word-level statistics, while the only types of transitions in this region are: 1) all bits switch and 2) all bits are constant. Therefore the probabilities of the different switching events in the sign-region can be calculated from:

$$p_i^{sign} = p(Hd^{sign} = i) = \begin{cases} i = 0 & : 1 - t_{sign} \\ i \neq 0 \wedge i \neq n_{sign} & : 0 \\ i = n_{sign} & : t_{sign} \end{cases}$$

The possible switching events and their probabilities are visualized in figure 7.

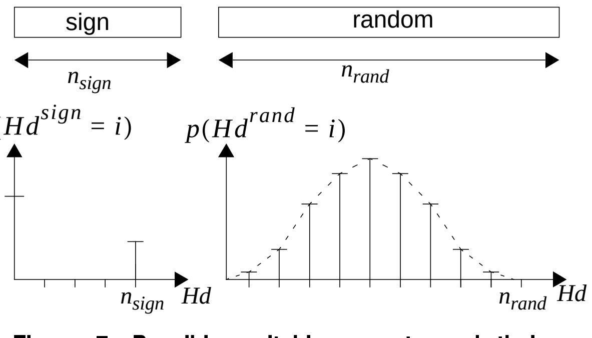


Figure 7: Possible switching events and their probabilities.

Having determined the switching probabilities in the different regions, it is possible to calculate the probabilities for switching events of the complete data word $p(Hd = i)$, which is:

$$p(Hd = i) = p(Hd^{sign} + Hd^{rand} = i) \qquad (13)$$

Therefore, the Hamming-distance distribution can be separated into different regions. In the following we will demonstrate that the probability of switching events within these regions can be calculated by conditional probabilities of corresponding switching events in the sign and random parts of the data word. Figure 8 exemplifies the conditions for a 16-bit data-word, with:

$$(n_{rand} = 10) > (n_{sign} = 6) \qquad (14)$$

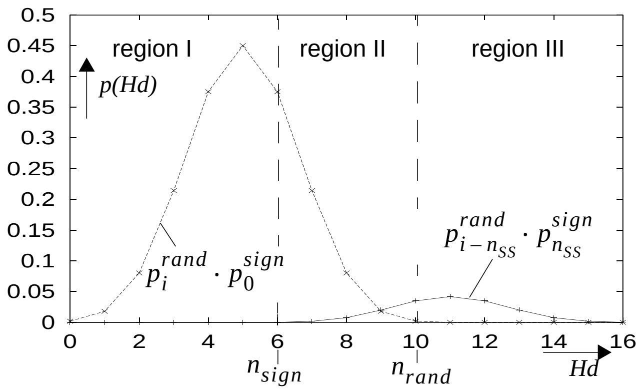


Figure 8: Regions of the Hd-distribution.

A. Region I: $\forall i \quad (0 \le i < n_{sign} \Rightarrow Hd = i)$

The switching events in region I can not stem from a transition in the sign region of the data word. They can only be due to transitions in the random part of the data-word, under the condition that no switching event in the sign region of the data word occurs. Therefore the probabilities in region I of the Hd-distribution can be calculated by:

$$p(Hd = i) = p_i^{rand} \cdot p_0^{sign} \qquad (15)$$

B. Region II: $\forall i \quad (i \ge n_{sign} \wedge i \le n_{rand} \Rightarrow Hd = i)$

In region II a switching event can be forced by:

1) a transition in the random part of the data-word with $Hd^{rand} = i$ under the condition that no switching event occurs in the sign region,
2) a switching event in the sign region ($Hd^{sign} = n^{sign}$) and a corresponding switching event in the random part with $Hd^{rand} = i - n^{sign}$.

This leads to:

$$p(Hd = i) = p_i^{rand} \cdot p_0^{sign} + p_{i-n_{sign}}^{rand} \cdot p_{n_{sign}}^{sign} \qquad (16)$$

C. Region III: $\forall i \quad (m \ge i > n_{rand} \Rightarrow Hd = i)$

The switching events in this region can only appear, if in the random part of the data-word a transition happens with $Hd^{rand} \ge (n_{rand} - n_{sign})$ under the condition that a switching event occurs in the sign part. As a consequence, the probabilities in this region can be calculated from:

$$p(Hd = i) = p^{rand}_{i-n_{sign}} \cdot p^{sign}_{n_{sign}} \qquad (17)$$

Equations (15) to (17) can be combined to an unified formula, which also holds for cases where $n_{sign} \geq n_{rand}$:

$$p(Hd = i) = \delta_{\overline{SS}} \cdot p^{rand}_{i} \cdot p^{sign}_{0} + \delta_{SS} \cdot p^{rand}_{i-n_{sign}} \cdot p^{sign}_{n_{sign}} \qquad (18)$$

with:

$$\delta_{\overline{SS}} = \begin{cases} 0: i > n_{rand} \\ 1: i \leq n_{rand} \end{cases} \quad \text{and} \quad \delta_{SS} = \begin{cases} 0: i < n_{sign} \\ 1: i \geq n_{sign} \end{cases}$$

and $0 \leq i \leq m$, where m is the word-length .

We have evaluated our approach for the audio signals which have been introduced in section 4.2 and found a good matching to the extracted distributions. As an example, figure 9 shows the Hamming-distance distribution for a typical speech signal which has been 1) extracted directly from a data stream and 2) calculated from (18). It can be seen, that the curves fit well.

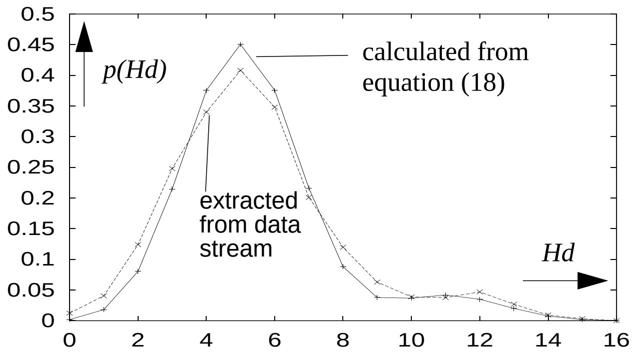


Figure 9: Comparison of extracted and estimated Hamming-distance distribution.

The improvement in estimation accuracy by using the Hamming-distance distribution instead of the average value strongly depends on the distribution form and the non-linearity of the model coefficients. For the example in figure 6, where the model coefficients increase nearly quadratical, an improvement of about 30% is reached for typical audio signals.

Finally it has to be noted, that, even if the approach for calculating the Hamming-distance was described for a one-input data stream, it is easily possible to enhance the approach to handle two- or multiple-input streams. This holds under the assumption that the different input streams can be regarded as uncorrelated.

7. Summary

We have presented a novel power macro-model which is based on the Hamming-distance of two consecutive input vectors. The model is parameterizable in terms of input bit-widths and can be applied to a wide variety of typical data-path components. The good trade-off between estimation accuracy, model complexity and flexibility makes the model attractive for power analysis and optimization tasks on a high level of abstraction. Model characterization is simple and efficient, as a small set of module prototypes is sufficient, because the dependency of the module structure complexity on the input bit-widths is considered.

Furthermore, a new approach to calculate the average Hamming-distance distribution of an input data stream was presented. The application of the Hamming-distance distribution, instead of a simple average, strongly increases the estimation accuracy in cases where the distribution is not symmetric and power has a non-linear dependency on the Hamming-distance. Since the data dependent model parameters can efficiently be calculated applying existing data models and propagation methods, the proposed estimation approach provides a possibility for fast power estimation, which is necessary for optimization of realistic designs at an early stage of the design process.

8. References

[1] Macii, E.; Pedram, M.; Somenzi, F.: High level power modeling, estimation and optimization. To appear in Trans. on Design Automation of Electronic Systems, 1998

[2] Landman, P. E.; Rabaey, J. M.: Architectural power analysis: The dual bit type method. *IEEE Trans. VLSI Syst.*, Vol. 3, pp. 173-187, 1995

[3] Landman, P. E.; Rabaey, J. M.: Activity-Sensitive Architectural Power Analysis, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 15, No. 6, pp. 571 - 587, 1996

[4] Bogliolo, A.; Benini, L.; De Micheli, G.: Adaptive Least Mean Square Behavioral Power Modeling. *Proceedings of European Design & Test Conference*, pp. 404-410, 1997

[5] Musoll, E.; Cortadella, J.: Scheduling and resource binding for low power. *IEEE Int. Symposium on System Synthesis*, pp.104-109, 1995

[6] Musoll, E.; Cortadella, J.: High-level synthesis techniques for reducing the activity of funcitonal units. *IEEE Int. Symposium on Low Power Design*, 1995

[7] Chang, J.-M.; Pedram, M.: Register Allocation and Binding for Low Power. I*EEE Design Automation Conference*, 1995

[8] Chang, J.-M.; Pedram, M.: Module Assignment for Low Power. *IEEE Europ. Design Automation Conference*, 1996

[9] Landman, P. E.; Rabaey, J. M.: Power Estimation for High Level Synthesis. *IEEE European Design Automation Conference*, 1993

[10] Ramprasad, R.; Shanbhag, N. R.; Hajj, N.: Analytical Estimation of Signal Transition Activity from Word-Level Statistics. *IEEE Trans. on Computer-Aided Design of Integr. Circuits and Systems*, Vol. 16, No.7, pp. 718-733, 1997

[11] Tsui,C.-Y.; Chan, K. K.; Wu, Q.; Ding, C.-S.; Pedram, M.: A Power Estimation Framework for Designing Low Power Portable Video Applications. *Proceedings of Design Automation Conference 1997*, pp. 421 - 424, 1997