Power Modeling of Complex Designs

TFE4580 - Specialisation Project

Embla Trasti Bygland

December 19, 2019

NTNU
Faculty of Information Technology
and Electrical Engineering
Department of Electronic Systems

Preface

The purpose of this project is to investigate several methods for estimating power at the RT-level and lay the foundation for a master thesis on RTL power estimation in cooperation with Nordic Semiconductor ASA.

This report presents several methods for RTL power estimation in its two main chapters. In the following sections these methods are compared to each other and discussed.

During this project several research works have been examined and a few methods have been selected for their diversity to explore a wide range of possibilities for RTL power estimation.

Abstract

In this project a literary review has been made on Register Transfer Level (RTL) power estimation. Several methods have been investigated for accurately estimating the power consumption of a design at the RTL. They can be categorised into two main methods; top-down estimation and bottom-up estimation.

The top-down approaches use physical attributes, estimated or already known, like the number of gates or the area of the design to estimate the load capacitance of the design. Together with the switching activity this load capacitance is used to estimate the power consumption. Contrary, the bottom-up method uses a representation at a lower abstraction level, like the gate level. It runs exhaustive power estimation there and attempts to make a model that can be used on the RT-level. This is typically done by exploiting the relation between the power estimates and the input (and output) switching statistics.

The bottom-up methods are in general more accurate than the top-down approaches, with worst-case average errors as small as 5.56%. This accuracy is due to doing accurate power estimation on a gate level representation of the design instead of the more abstract RT-level. The top-down methods trade of some of the accuracy for improved speed. This is achieved by estimating a parameter closely related to power consumption (like area and gate count) instead of characterising the power consumption directly. Both the top-down method and the bottom-up method have a high relative accuracy.

A fundamental insight gained from this review is that the two main methods make different trade-offs between computation time and accuracy. The bottom-up characterisation stage makes more accurate power estimation, but is also more time consuming. This makes it most suitable for VLSI (Very Large Scale Integration) where there is a lot of IP block reuse. The top-down method offers less accuracy,

but it is faster and does not need a characterisation stage. The relative accuracy is just as good as the bottom-up method, making it suitable if one is making new IP blocks, exploring the design space and are comparing different designs.

For accurate and fast power estimation a top-down estimation method making use of bottom-up models for already existing IP blocks could be used, combining the two methods.

Contents

Abbreviations

ADD	Algebraic Decision Diagram
ASIC	Application Specific Integrated Circuit
BDD	Boolean Decision Diagram
CMOS	Complementary Metal Oxide Semiconductor
Gate level	A gate level representation of a circuit consists of a netlist with gates and storage elements. It can be obtained from the Register Transfer Level representation through logic synthesis.
IC	Integrated Circuit
IP	Intellectual Property
LUT	Lookup Table
Macromodel	Simplified mathematical description of a system
MSE	Mean Square Error
RT-level	Register Transfer Level
RTL	Register Transfer Level
VLSI	Very Large Scale Integration

1 Introduction

1.1 Motivation

With more and more battery driven devices emerging and increasing performance demands on all holds it is more important than ever to design Integrated Circuits (ICs) with low power consumption in mind. To be able to do this it is important to be able to estimate power throughout the design flow.

The closer the design is to a finished product, the easier power estimation is. On a finished IC, one can simply measure it. The earlier in the design process the more abstract the design is. The more abstract the design is the harder power estimation becomes and the more advantages accurately measuring power could possibly have. Several tools do a good job at measuring the power at the gate level, but going back and forth between the gate level and the RT-level to change the design can be costly [1]. Estimating power at the RT-level would also make it easier to find and remove *Power Bugs*. A power bug is when the design violates the power constraints (Appendix A).

A RTL representation of a design is made early in the design flow and is an abstract representation. The RTL representation can be synthesised to a gate level representation where an accurate power estimate can be made and then one can return to the RTL to improve the design or explore the design space further. This jumping back and forth in the design flow can be costly and is not desirable. Optimisations to the RTL have more effect than optimisations done at a later stage in the design process, where placement and routing is done. To optimise a design to consume as little power as possible in an easy way, tools for estimating power at the RT-level are needed.

1.2 Project description

The project assignment (given in Appendix B) is motivated by the desire to estimate power consumption early. Making a system able to estimate power at the RTL was deemed too time consuming and out of scope for this project. Instead the project will be a literature review in preparation for a possible master's thesis. It will investigate the different approaches to RTL power estimation and look for suitable methods for an eventual implementation.

The methods investigated can be divided into two main categories. The **bottom-up** methods which try to estimate power at a less abstract level than the RT-level, making a power model that can be used at the RT-level later and the **top-down** methods which try to estimate physical qualities of the design, like the area or capacitance, and combine this with the switching activity to make a power estimate.

The bottom-up methods are the most accurate, but to make the model they require a long characterisation. The top-down methods are less accurate, but they are mostly characterisation free. The relative accuracy of both methods is high. Which method one choose to use will depend on ones needs and motivation for estimating power and is discussed in Chapter 7.

1.3 Report structure

After this introduction this report consists of the following chapters

2 Theory In this chapter some relevant and useful theory for the project is presented. First some theory regarding CMOS power consumption is given, followed by information on power, voltage and temperature corners in CMOS technology. The ASIC design flow is described to make the motivation behind RTL power estimation clearer. Some general theory on esimta-

tion is presented as this project deals with power estimation and a brief introduction to Binary Decision Diagrams is given as one of the papers presented in Chapter 4 make use of them.

3 Background This chapter presents the general ideas behind the research done on RTL power estimation and divides the approaches into top-down and bottom-up methods. It also presents some EDA tools already providing RTL power estimation.

4 Top-down approaches Here one of the main approaches to RTL power estimation is presented. It is referred to as top-down as it goes from a higher abstraction, trying to estimate not-yet known quantities in the design. Some approaches to this method is examined closer.

5 Bottom-up approaches Here the second of the main approaches to power estimation at the RTL is presented and different approaches to this solution is investigated. It is referred to as bottom-up as it goes from a gate level representation, and tries to make a model that still can be used on a higher abstraction representation being that on the RT-level.

8 Results Presents the achieved accuracies in estimation for the two different approaches. Looks at the differences and similarities between the two methods and compare them.

7 Discussion This chapter looks at different usecases of RTL power estimation and discuss the applicability of the two methods to these usecases. It also discusses the possible combination of the two methods.

8 Conclusion

This chapter concludes the project using the insights gained from the preceeding chapters. Some top-down power estimation methods are suggested for future work.

2 Theory

2.1 CMOS power consumption

The power consumption in digital CMOS-based circuits can be divided into dynamic and static power consumption. The dynamic power consumption is caused by the switching activity in the system and the static power consumption is caused by leakage in the CMOS transistors [2]. The total power consumption of the system will be the sum of these two and is given by Equation (2.1).

$$P_{tot} = P_{dynamic} + P_{static} \tag{2.1}$$

2.1.1 Dynamic power consumption

The dynamic power consumption can be divided into switching power and short-circuit power [2].

The main contributor to the switching power is the power it takes to charge and discharge the output capacitance of a logic gate. It can be calculated as shown in Equation (2.2).

$$P_{SW} = \frac{\alpha}{2} C_L V_{dd}^2 f_{clock} \tag{2.2}$$

α is an activity factor describing how often the output switches (changes value). C_L is the load capacitance on the gate output, V_{dd} is the supply voltage and f_{clock} is the clock frequency.

Another contributor to dynamic power consumption is the short-circuit current. When CMOS logic is in the middle of switching both the NMOS and the PMOS

transistor will be partly on, allowing current to flow from V_{dd} to ground. This is illustrated in Figure 2.1.

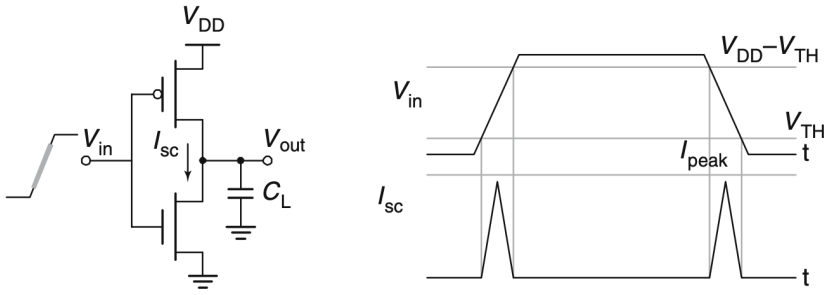

Figure 2.1: Illustration of the short-circuit power in CMOS logic [3]. When V_{IN} rises and falls I_{SC} will flow from V_{DD} to ground for a short period of time.

The short-circuits contribution to power consumption can be calculated using the expression shown in Equation (2.3)

$$P_{SC} = t_{sc} V_{dd} I_{sc} f_{clock} \tag{2.3}$$

where t_{sc} is the duration of the short circuit current, V_{dd} is the supply voltage of the system, I_{sc} is the average short-circuit current and f_{clock} is the clock frequency.

The total dynamic power consumed in the circuit will be the sum of the switching power and the short-circuit power, shown in Equation (2.4).

$$P_{dynamic} = P_{SW} + P_{SC} \tag{2.4}$$

The switching power can in turn be divided into the switching of the nets of the system and the switching of internal signals in a module. If the internals of a system are either unknown, or abstracted away for other reasons one can say that the power consumption is the sum of the switching power consumption of the nets and the "internal power consumption", P_{IN}, consisting of the internal switching power in a module and the short-circuit power contribution (2.5).

$$P_{dynamic} = P_{SW} + P_{IN} \tag{2.5}$$

The total dynamic power can be seen as a sum of the switching power, P_{SW}, of all the nets in the system and the internal power, P_{IN}, of all the cells in the system.

2.1.2 Static power consumption

The static power consumption is caused by the leakage current in the CMOS transistors. It has traditionally been negligible compared to the switching power, but the downscaling of the technologies and the lower supply voltages which in turn leads to a lower V_t for transistors makes the static power consumption just as (or more) important than the switching power. Contributions to the leakage current come from the sub-threshold leakage, the gate leakage and the junction leakage [2].

- Sub-threshold leakage is current leaking from source to drain while the transistor is operating in the weak inversion region ($V_G < V_t$). It increases exponentially when lowering V_t [2] and is the larges contributor to static power.
- Gate leakage is current caused by electrons tunneling through the oxide layer of the gate.
- Junction leakage is caused by potential differences between the drain diffusion region and the substrate. It is often negligible compared to the other two contributors.

Downscaling and small dimensions and higher temperature makes this WORSE.

2.2 Power saving techniques

There are several techniques applied to modern ICs with power reduction in mind. Some of them are implemented in the RTL, others are applied by tools in synthesis

or by other means.

2.2.1 Clock Gating

Clock gating is a technique based on saving switching power by preventing the clock from propagating. This can be integrated on a very low level, preventing the clock from entering registers, or on a larger scale, preventing the clock from propagating into a module or subsystem. When there is no clock all switching activity stops and power is saved. The register clock gating can be done by synthesis tools, while the larger scale has to be applied during the RTL design. In figure 2.2 an example of clock gating is shown. The inputs of a module comes from a register and the clock gating into the register effectively freezes the register values, preventing unwanted activity in the logic module.

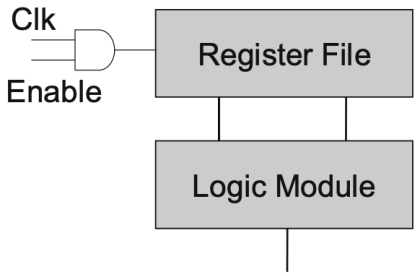

Figure 2.2: A clock gating example [3].

2.2.2 Power Gating

Power gating is a technique where one completely or partly shuts of power in a module or subsystem by shutting of the connection to one or both of the supply rails when will be inactive for a significant amount of time. This lowers both the switching power and the static power, effectively setting it to zero for the wanted amount of time. Turning the power on and off will typically take time and extra logic is needed if one wishes to retain any information when the system is shut down.

2.2.3 Voltage and frequency Scaling

Voltage and frequency scaling is a technique where the supply voltage and frequency of a system will vary, either in different regions of the design or at different points in time. This is more complicated than the power- and clock gating but it allows for more accurately adapting the system to its specification by letting modules run at the lowest voltage and frequency possible to meet the timing demands even if other parts of the system must run on a higher voltage and frequency. It is very applicable to larger designs as the many modules these consist of will have different performance demands and thus could run on different voltages and frequencies.

2.3 Process, Voltage and Temperature corners

An IC will not always perform exactly as expected Variations in the manufacturing and the environment will lead to significant changes in the characteristics of a transistor. To make a circuit operate as expected these variations should be taken into account. The sources of variation are **process variation**, **supply voltage** and **temperature**.

The **process variation** is caused by slight variations in the manufacturing process, such as the concentration of dopants or the oxide thickness.

These variations lead to manufactured transistors having varied characteristics. These are often described as; F (fast) and S (slow) for the corner-cases, where the transistor will operate faster and slower than expected, and T (typical) describing an average transistor. For a CMOS transistor, consisting of one PMOS and one NMOS transistor this yields four operating corner-cases describing a constricted area, in which the pair og transistors will always operate within. FF, SS, SF and FS. The center of this area is (TT), the average transistor. This is illustrated in figure 2.3.

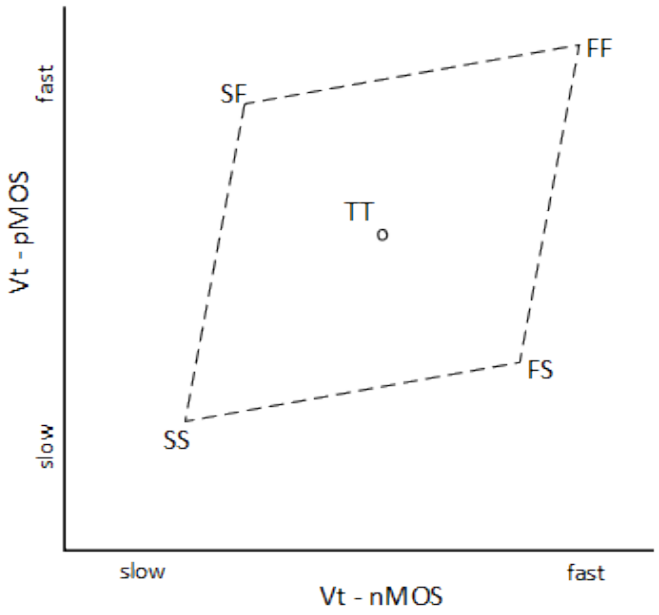

Figure 2.3: CMOS design corners

The variation in **temperature** also affects the operation of the transistor significantly. If the operating temperature is high the transistor will have a higher leakage, which has a negative impact on power consumption.

Lastly, the **supply voltage** can deviate from the intended for many reasons such as the tolerance of the voltage regulators and noise.

Thus, it is not enough to only take an average transistor in the TT corner, operating in room temperature with the intended supply voltage, into account. One also needs to take the transistor in the slow corner operating on a high temperature and a low voltage, and all the other corner-cases into account.

2.4 The ASIC design flow

To understand the motivation behind power estimation at the RT-level it is an advantage to understand the ASIC design flow. The ASIC design flow is a mature process. It is based on proven methodologies and best practices. It is effectively

taking every aspect and stage of the design process into consideration, optimising the time to market [4].

It can be divided into 10 stages as shown in Figure 2.4. The stages are described below.

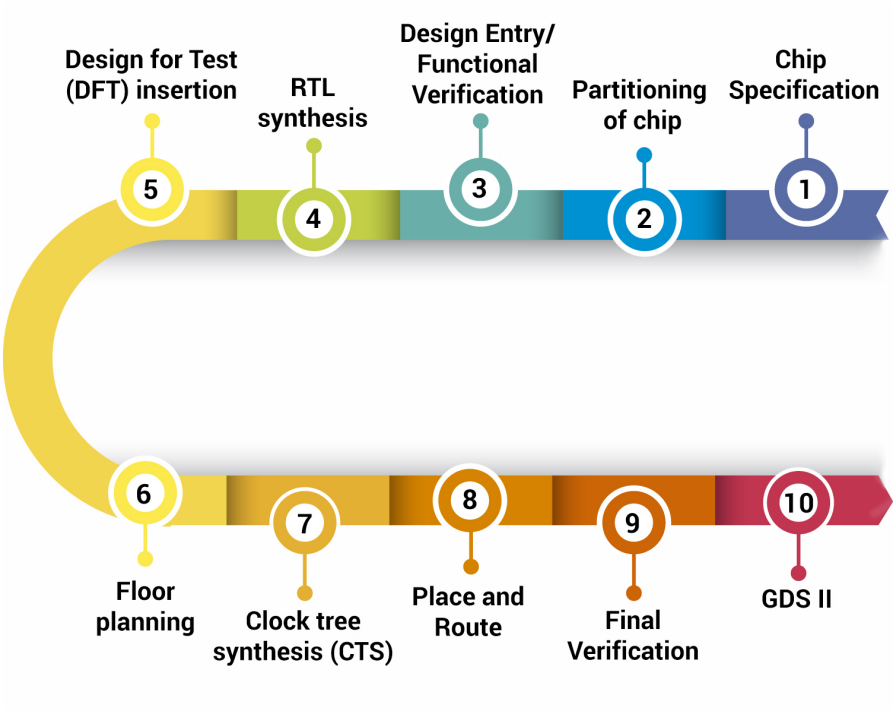

Figure 2.4: Illustration of the ASIC design flow. From [4]

1. **Chip specification**
   Here the features and functionalities of the microarchitecture is decided together with specifications for time area and power.

2. **Partitioning of chip**
   Here design requirements for ASIC design layouts are followed to make a structure for the design. It is partitioned into multiple hierarchical modules while keeping the ASIC's best performance in mind.

3. **Design Entry/Functional Verification**
   RTL and testbenches are made at this stage and the functional verification

is done. It is also confirmed that the design functions as intended by doing behavioural simulation. A lot of code coverage is checked for to verify the correctness of the code.

4. **RTL synthesis**
   Using a logical synthesis tool the RTL code is translated into a gate-level netlist. A database is made for the design and this stage is done when the database is complete and the timing constraints are met.

5. **Design for test insertion**
   Here designs for testing the system are made and introduced to the design. These can for example be scan chains that link all the registers in the design together into one long shift register or a built-in self test for the memory.

6. **Floor planning**
   This is the start of the physical implementation. Here the components are placed on the chip and optimisations are made to minimise the size of the chip and its power consumption and improve signal delays and make the routing of wires between modules as easy as possible.

7. **Clock tree synthesis**
   In this step a clock tree is built, propagating a correct clock through to all the sequential elements in the chip. This should fulfil the timing and area requirements and use as little power as possible. There are many different clock tree structures.

8. **Place and route**
   Here the placement of the circuitry inside the general blocks (determined in the floor planning stage) is done. Signals are then routed inside and between the blocks and their net values are estimated.

9. **Final Verification**
   This stage takes care of verifying the physical design. It has three parts

   - *Layout versus schematic:* Checking that the geometry and layout matches the schematic and the netlist.

- *Design rule checks:* Checking that the geometries in the file that is going to be sent to a foundry matches the rules given by the foundry.
- *Logical equivalence checks:* Checking the logic equivalence between the design pre and post layout.

10. **GDS II**

    The end stage in the design flow is the resulting GDSII file. GDSII is a binary file format used to represent the ASIC in its entirety. This is what is sent to a foundry for production. The product would then have to be packaged, verified and tested before it is delivered to a client.

It is an advantage to see if one has met the power constraints and to be able to compare design solutions and look for power bugs at the Design Entry stage, before the Synthesis stage. Power estimation after synthesis is easier as one gets the gate level representation of the design, which is less abstract than an RTL representation, but if one is to jump back and forth between these stages it can be costly and inefficient.

2.5 Estimation Theory

Estimation is a branch of statistics where one tries to estimate the values of parameters based on measured data. In this project the estimators and estimation methods presented all try to estimate the power consumption of a digital design. In (2.6), the estimator $\hat{\boldsymbol{\Theta}}$ is estimated as a function of the measured parameters $X_1, X_2, ...X_N$.

$$\hat{\boldsymbol{\Theta}} = \boldsymbol{F}(X_1, X_2, ..., X_N) \tag{2.6}$$

The parameters will affect the distribution of the measured data. An estimator will try to approximate the underlying data using the measured data.

The **accuracy** of an estimator is a measurement of how close the estimated value is to the actual value of what we try to estimate. The **relative accuracy** or

fidelity, on the other hand, reflects the estimators ability to be consistent in the ordering of the estimates. If the ordering of different estimates is the same as the real ordering they can deviate from the actual value but still be evaluated correctly compared to each other and we can say we have 100% fidelity.

2.6 Binary Decision Diagrams

In [5] Bogliolo, Benini and Micheli make use of Algebraic Decision Diagrams (ADDs) to estimate the capacitance of a combinatorial circuit. ADDs are a more complex variant of the Binary Decision Diagrams (BDDs). The following section gives a brief introduction to BDDs.

BDDs are widely used to represent digital circuits for different purposes. [6]

By viewing a boolean expression as the most simple atomic if-then-else expressions any boolean expression can be made into a graph describing all its dependencies. Such a graph is called a Binary Decision Diagram (BDD) [7]. A BDD has one or two terminal nodes that are one or zero and a set of variable nodes with two outgoing edges describing their functions for either being low or high. A variable is associated with each node.

[7] uses the expression given in Equation (2.7) as an example.

$$t = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2) \tag{2.7}$$

It can be written as the following list of if-then-else expressions

$$
\begin{aligned}
t &= x_1 \rightarrow t_1, t_0 \\
t_0 &= y_1 \rightarrow 0, t_{00} \\
t_1 &= y_1 \rightarrow t_{11}, 0 \\
t_{00} &= x_2 \rightarrow t_{001}, t_{000} \\
t_{11} &= x_2 \rightarrow t_{111}, t_{110} \\
t_{000} &= y_2 \rightarrow 0, 1 \\
t_{001} &= y_2 \rightarrow 1, 0 \\
t_{110} &= y_2 \rightarrow 0, 1 \\
t_{111} &= y_2 \rightarrow 1, 0
\end{aligned}
$$

It can quite easily be observed that t_{000} is equivalent with t_{110} and t_{001} is equivalent to t_{111}. This also leads to t_{00} being the same as t_{11} leading to a much shorter list of expressions shown in Equation (2.8).

$$
\begin{aligned}
t &= x_1 \rightarrow t_1, t_0 \\
t_0 &= y_1 \rightarrow 0, t_{00} \\
t_1 &= y_1 \rightarrow t_{00}, 0 \\
t_{00} &= x_2 \rightarrow t_{001}, t_{000} \\
t_{000} &= y_2 \rightarrow 0, 1 \\
t_{001} &= y_2 \rightarrow 1, 0
\end{aligned}
\tag{2.8}
$$

This list of unique expressions is what is known as a BDD and can be drawn as a directed acrylic graph, shown in Figure 2.5. Each sub-expression is a node in the graph and the dotted lines out of it corresponds to the else-part of the expression while the hard line corresponds to the then-part.

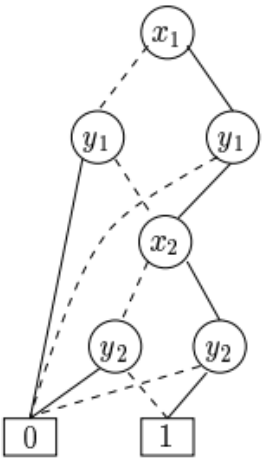

Figure 2.5: A BDD for the expression given in (2.7) from [7]

3 Background

Research on RTL power estimation has been done since the 90s. The main motivation is to optimise the ASIC design flow and minimise the time to market, while still keeping up with state of the art power demands.

Many EDA tool vendors have specific software for this purpose. To mention a few Ansys has PowerArtist [8], Synopsys has Spyglass Power [9], Mentor Graphics has PowerPro [10] and Cadence has Joules RTL Power Solution [11]. Estimating power at an even higher level is also becoming interesting. IEEE recently released a new standard for power modelling at the system level, which is even more abstracted than at the RTL. [12].

There are two main methods for RTL power estimation. They will be referred to as top-down and bottom-up. The top-down method attempts to estimate parameters affecting power, like the total capacitance or area of a design [5], [13], [14], [15]. It then uses the derived estimates together with the switching activity to make an estimate for the power consumption.

The bottom-up method is the most used one [16], [17], [18], [19], [20], [21]. Here one synthesises the RTL to get a gate-level representation. This representation is then used in a characterisation stage, "measuring" the power consumption using a gate-level power estimation tool and relating this to certain parameters that also exist at the RT-level. It uses the information it gets about the power consumption to create a macromodel that can later be used at this higher abstraction to get near gate level accuracy on the estimation.

4 Top-down approaches

The top-down methods for power estimation tries to estimate basic qualities of a design like the number of gates or total capacitance of the design. It then puts this information together with the switching activity on the in- and outputs to make a estimate of the power consumption.

There are many approaches to this. One could try to directly estimate the capacitance or go through the number of gates to do so. It is also possible to use the concept of entropy. One could treat all modules the same or distinguish them based on their type, whether they are a logic block or a memory etc.

Even if one does power estimation at the RT-level it is common to not rely only on the RTL representation, but also get some information from a lower level representation of the design, like the gate level, in a characterisation stage. VLSI makes this less expensive as a module used once probably will be reused at a later point and a more expensive characterisation stage done once is worth more accurate power estimations for all the designs this module is used in.

4.1 Estimating the capacitance

In [5] Bogliolo, Benini and Micheli uses an analytical approach to estimate the power consumption of combinatorial circuits, with a model where the accuracy does not depend on the input statistics of the module, where the upper bounds are pattern dependent some accuracy is traded off for complexity.

They divide the factors contributing to the circuit's power consumption into structural and parasitic phenomena. Structural phenomena are those that can be taken into account at the abstraction level of their model, while the parasitic phenomena are those their model can not take into account.

They make use of a model consisting of a netlist with backannotated capcitances and zero propagation delay. The structural phenomena is the supply current charging capacitances when the circuit switches. Parasitic phenomena are short-circuit currents, internal charge redistribution and glitches.

This abstract golden model leads to a loss of accuracy, but the authors argue that this loss is not critical as relative accuracy might be more important than absolute accuracy at the RT-level. In addition the absolute accuracy could be increased with characterisation if deemed necessary.

The power consumption in the model is described as in (4.1)

$$e(x^i, x^f) = V_{dd}^2 C(x^i, x^f) \tag{4.1}$$

where $C(x^i, x^f)$ is the total switching capacitance (the total capacitance associated with signals that have a rising transition in the time interval $[t^i, t^f]$) and $p(x^i, x^f, T) = e(x^i, x^f)/T$ and T is the time interval over which we are measuring the power. Gates with a rising input transition in the time interval are denoted S_R. It is thus given by (4.2)

$$C(x^i, x^f) = \sum_{g_j \in S_R} C_j \tag{4.2}$$

S_R is given by (4.3)

$$S_R = \{\ g_j, j \leq N \mid g_j(x^i) = 0\ AND\ g_j(x_f) = 1\ \} \tag{4.3}$$

$g_j(x)$ is the output function of gate, N is the total number of gates. (Sum of capacitances of all transitions from 0 to 1).

Due to the definition of S_R the total switching capacitance can also be written as in (4.4)

$$C(x^i, x^f) = \sum_{j=1}^{N} g_j'(x^i) g_j(x^f) C_j \tag{4.4}$$

This equation can be the RTL representation of the capacitance as long as the logic function of each node is known, but the complexity of such a representation is too complex and might not be known if the IP block is third-party.

The model is found by finding the Algebraic Decision Diagram (ADD) from the BDD. An ADD is an extension to the BDD proposed by Bahar et. al in [22]. Here the boolean functionality is allowed to have a real codomain, allowing the output of the boolean functions to have other values than 1 and 0. The ADD directly represents the capacity of the circuit as a discrete function. To get a less complex model the ADD can be simplified.

Node collapsing is the easiest way of simplifying an ADD. Here some pattern dependence is sacrificed for a simpler expression when two nodes are combined into an average node (collapsed). By changing how many nodes we wish to collapse (the degree of compression) we effectively choose how much accuracy we wish to trade off to lower the complexity.

In every case a node is collapsed it replaces a sub-function with a constant value. To chose a node to collapse the variance of the functions associated with all nodes need to be inspected and the nodes with the smallest variance is collapsed. The average value and variance of a node is easily computed using its children.

By changing the number of nodes to keep (not collapse) depending on the complexity of the modules the method was tested on, an average relative error between 4.8% and 18.7% was achieved for different IPs.

4.2 Capacitance models for different entities

In [14] Liu and Svensson use different methods to estimate power for different blocks and different connections. It divides power consumption into the power consumed by:

- Logic
- On-chip memory
- Local and intermediate interconnects
- Clock distribution

- Global bus
- Off-chip driving

It uses a different model to estimate the power consumption of each of these, but all of the models estimate some capacitance and takes activity into account. It achieves higher accuracy for tailoring the capacitance estimates to the type of circuit it is handling.

This model goes pretty far in how much it tries to estimate at such a high level. It uses basic layout strategies to estimate the interconnects and and a general model for the clock distribution to estimate its power consumption.

To test the system the power of two microprocessors were tested and compared to their datasheets. On the Alpha 21064 the estimated total power was $32W$ and the datasheet claimed a total power of $30W$. On the Intel 80386 the estimated total power was $17.2W$ while the datasheet claims $12W$-$16W$. This has an average error of 7.1%.

4.3 Using entropy

In [13] Nemani and Najm uses the concept of entropy in their power estimations. For a given boolean variable, x, with p as the probability of x being 1 i.e. $\mathcal{P}\{x = 1\} = p$ will have an entropy defined by (4.5)

$$H(x) = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{(1-p)} \tag{4.5}$$

This function reaches its maximum for a p of 0.5, which is where x will have the highest switching activity as it is 1 half of the time.

If a discrete variable can have more than two values, the entropy equation can be described as in (4.6)

$$H(x) = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \tag{4.6}$$

Here p_i is the probability of x taking the ith possible value. This means each boolean variable will have a corresponding entropy function with a value corresponding to the variables probability of being 1.

If we have $\boldsymbol{Y} = f(\boldsymbol{X})$ where $\boldsymbol{X}$ and $\boldsymbol{Y}$ are vectors of length n and m. The input entropy of f is the entropy of $\boldsymbol{X}$ and is given by (4.7)

$$H(\boldsymbol{X}) = \sum_{i=1}^{2^n} p_i \log_2 \frac{1}{p_i} \tag{4.7}$$

while the output entropy of f is given by (4.8)

$$H(\boldsymbol{X}) = \sum_{i=1}^{2^m} p_i \log_2 \frac{1}{p_i} \tag{4.8}$$

It can be shown that $H(\boldsymbol{Y}) \leq H(\boldsymbol{Y})$. This means that the entropy of a combinatorial circuit always is less than the entropy of its input.

$H(\boldsymbol{Y})$ can be used to predict the circuit area as shown in equation (4.9)

$$\mathcal{A} \propto \frac{2^n}{n} H(\boldsymbol{Y}) \tag{4.9}$$

The circuit area is proportional to the circuit capacitance and the input switching can be approximated by the transition density given by equation (4.10)

$$\mathcal{D} = \frac{1}{N} \sum_{i=1}^{N} D(x_i) \tag{4.10}$$

where $D(x_i)$ is the transition density of node x_i, the average number of logic transition the node makes per second.

This means that the average power over some time frame can be estimated as shown in Equation (4.11)

$$P_{avg} \propto \mathcal{A} \times \mathcal{D} \tag{4.11}$$

The circuit is treated as a blackbox, and $\mathcal{D}$ and $\mathcal{A}$ are both determined only from the input/output entropies of the block. The entropies are determined through a simulation.

For synchronous circuits one would have to add the power consumption of the latches/registers/flip-flops.

5 Bottom-up approaches

In bottom-up RTL power estimation the power is estimated at gate level where we know more about the implementation. This power information is then related to parameters that also exist at the RT-level; the input and output switching activity.

The method would be most accurate if power is estimated at the gate level for all possible input combinations. For small modules with very few inputs this can be done, but as soon as the modules increase in size this is infeasible. Thus, one is required to be more smart about it and use statistics related to in- and outputs instead.

The power-switching relation is either stored in a lookup table (LUT) or it is fitted to an equation. In a LUT one would need as many dimensions as one has parameters relating to the power. This typically takes up a lot of space as several parameters are necessary to get an accurate relation between the power and the inputs and outputs of the system. The equation-based method is less accurate, but computation is faster and storage space for a LUT is no longer needed.

5.1 Based on in- and output statistics

Gupta and Najm has done a lot of work on top-down power estimation. In [16] they make a three-dimensional LUT. In [17] they extend this model, adding a fourth parameter, making it a four-dimensional LUT with higher accuracy than the previous one. In [18] they make an equation based model based on the same four parameters.

The power is estimated as a function of the parameters P_{in}, D_{in}, SC_{in} and D_{out} (5.1).

$$P_{avg} = f(P_{in}, D_{in}, SC_{in}, D_{out}) \tag{5.1}$$

$\boldsymbol{P_{in}}$ - The average input signal probability. The probability of an input signal being one.

$\boldsymbol{D_{in}}$ - The average input switching activity.

$\boldsymbol{SC_{in}}$ - The average input spacial correlation coefficient. explain more

$\boldsymbol{D_{out}}$ - The average output (zero delay) switching activity.

If we define the signal probability P_i as the average fraction of clock cycles that go by, ending in the signal x_i being one and the transition density D_i as the average fraction of clock cycles in which the signal x_i will make a transition (change its value). We can then define P_{in} and D_{in} as shown in (5.2). D_{out} will be the transition density of the output signals rather than the input signals.

$$P_{in} = \frac{1}{n}\sum_{i=1}^{n} P_i \qquad and \qquad D_{in} = \frac{1}{n}\sum_{i=1}^{n} D_i \tag{5.2}$$

The spatial correlation SC_{in} has a more complicated definition as it describes how the different inputs relate to each other. It is shown in (5.3)

$$SC_{in} = \frac{2}{n(n-1)}\sum_{i=1}^{n}\sum_{j=i+1}^{n} \mathcal{P}\{x_i = 1, x_j = 1\} \tag{5.3}$$

If the temporal correlation is defined as the probability of x_i and x_{i-1} both being one Gupta and Najm proves in [17] that it is represented by P_i and D_i and no extra coefficient is needed to take the temporal correlation into account.

Based on these parameters a 4-dimensional LUT was made and then used at the RT-level in [17]. The entire LUT will not be filled as the parameters are subject to several constraints given in (5.4) (maximum one transition per cycle) and (5.5) (due to SC_{in} being a correlation between the inputs) n is the number of input nodes in the system. These reasoning behind and the derivation of these constraints are given in [17]

$$\frac{D_{in}}{2} \leq P_{in} \leq 1 - \frac{D_{in}}{2} \tag{5.4}$$

$$\frac{nP_{in}^2 - P_{in}}{(n-1)} \leq SC_{in} \leq P_{in} \tag{5.5}$$

To save the time and storage space it would take to build an entire 4-dimensional LUT, an equation-based model is attempted in [1]. It is determined that a linear function is not accurate enough, thus a quadratic equation was explored as shown in (5.6).

$$\begin{aligned}\hat{P}_{avg} = c_0 &+ c_1 P_{in} + c_2 D_{in} + c_3 SC_{in} + c_4 D_{out} \\ &+ c_5 P_{in} D_{in} + c_6 P_{in} SC_{in} + c_7 P_{in} D_{out} \\ &+ c_8 D_{in} SC_{in} + c_9 D_{in} D_{out} + c_{10} SC_{in} D_{out} \\ &+ c_{11} P_{in}^2 + c_{12} D_{in}^2 + c_{13} SC_{in}^2 + c_{14} D_{out}^2\end{aligned} \tag{5.6}$$

In a few cases where the quadratic solution was not performing well it was shown that the 35-coefficient cubic equation would do well.

The method in general performed with higher variance than a LUT based on the same parameters and the average error was worse in most cases. The computation time was, however, much improved.

To determine how many data points were to be used to fit the equation and determine its coefficients a RLS (regressive least square) algorithm was used. The number of points is determined depending on when the error starts to converge towards an unknown mean value or has reached below a defined threshold the number of points is determined.

In [19] Bernacchia and Papaefthymiou makes use of the parameters P_{in} and D_{in} in Gupta and Najms early work. They add parameters for spatial and temporal correlation between the inputs; S_{in} and T_{in}.

Defining an input stream as $x = \langle (x_{11}, x_{22}, ..., x_{1M}), (x_{21}, x_{22}, ..., x_{2M}), ..., (x_{N1}, x_{N2}, ..., x_{NM}) \rangle$ The spatial correlation is defined as the average of the bit-

wise *XNOR* between all possible channel streams $x_i = \langle x_{1i}, x_{2i}, ..., x_{Ni} \rangle$ and $x_j = \langle x_{1j}, x_{2j}, ..., x_{Nj} \rangle$ in a steam, x, S_{in} is given by Equation (5.7).

$$S_{in} = \frac{\sum_{j=1}^{M} \sum_{k=1}^{M} \sum_{i=1}^{N} x_{ij} \oplus x_{ik}}{N \times M \times (M-1)} \tag{5.7}$$

This was decided because the SC_{in} presented by Gupta and Najm in [18] was the average of a bit-wise *AND* between input channel streams which correlates input streams with matching 1's. By using the *XNOR* matching zeros will also increase the correlation.

The temporal correlation T_{in} tries to take what S_{in} misses into account. It convolves a channel stream with a window of itself (5.8). This allows it to see correlation in how the stream evolves in time.

$$T_{in} = \frac{\sum_{j=1}^{M} \sum_{L-1}^{N-L+1} (w_j \otimes x_j)}{N \times M} \tag{5.8}$$

w_j is a window of length L in the channel stream x_j.

The models lack of dependence on output signals gives it an advantage when it comes to computation speed as no outputs need to be calculated before getting a power estimate.

The power is mapped to a nonlinear function given in (5.9).

$$PD_{avg} = f(P_{in}, D_{in}, S_{in}, T_{in}) \tag{5.9}$$

By checking the general power dependency on the four parameters separately they all seem to have a low polynomial dependency. A 3rd degree polynomial is therefore used.

It is thus necessary to calculate 35 coefficients in the characterisation phase. The estimation is effective as no simulation is required to estimate the power, and the characterisation phase is quite effective as well. Measurements, done on ISCAS-85

benchmark circuits the average error was less than 8% in all cases. The $4D$ LUT was more accurate with average errors at most being 5.56%.

In [21] Liu and Papaefthymiou use the parameters P_{in}, S_{in} and D_{in} used by Bernacchia and Papaefthymiou in [19]. To characterise the circuit it is simulated with varying input while a set of power dissipation points $\mathcal{P}$ are made. These points are then curve-fitted to derive an expression g using a minimal mean-square error criterion such that

$$\mathcal{P} = g(P_{in}, D_{in}, S_{in}) \tag{5.10}$$

The power sensitivity to the input metrics is the partial derivative of $\mathcal{P}$ and shows how the power consumption is affected by that input switching.

To obtain data points for the three output metrics P_{out}, D_{out} and S_{out} three functions are derived, again using a minimal square error criterion, f_1, f_2 and f_3. They are given in Equation (5.11), (5.12) and (5.13)

$$P_{out} = f_1(P_{in}, D_{in}, S_{in}) \tag{5.11}$$

$$D_{out} = f_2(P_{in}, D_{in}, S_{in}) \tag{5.12}$$

$$S_{out} = f_3(P_{in}, D_{in}, S_{in}) \tag{5.13}$$

An output metrics sensitivity to an input metric is also found by partially deriving the the function with regards to that input metric.

As only the statistics are computed and the estimation is static the errors propagating through cascaded blocks will get worse for each block they go through.

For a block with n input groups (inputs from 3 different places) $1 + C^1_{3n} + C^2_{3n}$ coefficients need to be calculated. Where C is the binomial coefficient function (find out what this is) A second degree complete polynomial is used to do the curve template for the output metrics and the power estimate.

The method has purely static estimation and in experiments with several IP block systems it has an average estimation error of 7.3%.

5.2 Long input sequences

In [20] Jiang et al. refers to power consumption caused by glitches, short-circuit current and leakage currents as *hazardous power* and the **average** power consumption caused by a node switching state is the *steady-state power*. The *total power* is the sum of the two components. The authors tries to address the problem of RT-level estimation techniques suffering from low accuracy and gate-level estimation techniques suffering from long runtimes and find some trade-off between the two.

Their power estimation method is based on one important observation: For a given module, long input sequences which produce similar *steady-state power* will also produce similar *total power*.

The method has a pre-processing stage and a characterisation stage. In the pre-processing a *cross-coefficient function* is made. It maps the steady-state power of the module to the total power of the module. This function is defined only once. In the characterisation phase the RTL design undergoes zero-delay simulation. Relying on the cross-coefficient function to achieve accurate power estimation.

For long input sequences producing similar steady-state power the hazardous power has the behaviour of a random variable. Among many input sequences producing the same steady-state power the longer input sequences have hazardous power with smaller variance.

A lot of long input sequences are randomly generated and the ones of these closest to a predefined value are picked out. This is the given steady-state power. The variation in the total power in relation to the inputs will then only be the random hazardous power, which is low as the input sequences are long. As the length of each sample input sequence is increased the standard deviation is decreasing. For each range of the designs total power it is necessary to find a long enough

input sequence resulting in it and then use transistor- or circuit-level simulation to estimate power consumption. The expected mean value of the total power for that steady-state power will then be derived. For the other input sequences with similar steady-state power this is the estimate of total power.

For a fixed steady-state power, when the sample input sequences length is increased the total power will converge to a fixed value. There will thus be a one-to-one mapping between a steady-state power and a total power. This is the cross-coefficient function.

For a gradually increasing toggle rate the sample input sequence length is gradually increased for that toggle rate while power is estimated. When power converges the steady-state and total power is reported, corresponds to one point on the cross-coefficient function and the toggle rate is increased and the process is repeated. This makes a graph with $1/step\_size$ points, as the toggle rate goes from 0 to 1.

For parametrised modules only a few selected parameter sizes are chosen and interpolation is used to estimate the power in between. A complexity function is also added to the module, because for some parametrised modules interpolation is not a suitable technique.

The hybrid model described in the article title is not the only estimation technique used in the estimation process. Cycle-based simulation is used to determine the input to each RTL module in the design. And accurate power estimates are made using the hybrid model described. When it comes to controllers and random circuitry in the design; for small modules (input $=< 4$) a LUT is made to estimate power, for bigger modules a Monte-Carlo simulation approach that it has from its source [6] is used (A novel methodology for transistor-level power estimation elns).

Due to being able to do the characterisation at gate level the method yields accurate estimates in short time at the RT-level later. Testing the method thoroughly on simple modules yields errors less than 6%.

6 Results

A number of RTL power estimation techniques have been investigated. They are divided into two main categories; bottom-up and top-down.

The top-down approaches try to estimate physical attributes of the design relating to the capacitance, like the number of gates or the area of the design, and use this together with the switching activity to estimate power. The bottom-up method uses a representation at a lower abstraction level, runs exhaustive power estimation on it on that level in order to make a model that can be used on the RT-level out of these estimates, typically combining it with the input (and output) switching statistics.

The bottom-up methods are the most accurate. The different bottom-up methods had average (worst-case) errors in the range from 5.56% to 8%. The most accurate one being the four dimensional LUT made by Gupta and Najm in [17]. Many of the top-down articles are less focused on results and work more as proof of concept. The top-down method of Bogliolo Benini and Micheli [5] had an average relative error of 18.7% for the IP block fitting their model the worst. The method of Liu and Svensson [14] focuses more on getting an accurate estimate of the total power further into the design process, after placing, clock tree synthesis and routing. Their method was tested on two microprocessors. Their estimated total power has an average error of 7.1% from the total power mentioned in the datasheets of the microprocessors.

When it comes to relative accuracy many of the papers mentions it, but no results proving the relative accuracy are presented. A good assumption could be that the power is closely related to the switching activity and the estimation methods are as well, so a high relative accuracy is achieved in most cases.

A disadvantage of the bottom-up methods is that they require a characterisation

stage. This is more complicated the larger the IP block is and can be very time consuming.

Most methods presented allow for time partitioning, with exeptions being the method of Jiang et al. in [20] and Nemani and Najms entropy-based method [13]. These are developed for estimating the average power consumption within a larger timeframe. Some time accuracy could be achieved in the entropy-based method, but this will probably impact the accuracy of the method.

A comparison of the methods can be seen in Table 6.1.

	Top-down	Bottom-up
Characterisation needed	No	Yes
Computation time	Fast	Fast
Accuracy	Good	Very good
Relative accuracy	Good	Good
Time relativity possible	Yes	Yes
Reliance on VLSI	No	Yes

Table 6.1: A comparison of the top-down and the bottom-up methods

7 Discussion

There are several motivations for estimating power. Determining which motivation is most important in a given usecase is key to determine what method to use in the power estimation. To see whether the design is below a certain power budget, the accuracy is important. To compare different design solutions, relative accuracy is important. If one wants to see when and where a system consume power, the time granularity and the modularity of the estimation is important.

If a new design consists of mostly ready-made IP blocks, these could all already have a macromodel. In this case the power estimation could consist of running a simulation to see how the signals propagate through the macromodels and summing up their power consumption. If macromodels are not yet made for parts of the design, these would have to be made through synthesis and characterisation before power estimation can be done at the RT-level.

If a design is being made from scratch, the design process would consist of finding the optimal solution with the given design requirements. In this design process, getting quick, relatively accurate estimates may be important to effectively explore the design space and narrow down to the best solution(s).

When verifying the power consumption of a design, one has to confirm there are no power bugs present. It is advantageous to see the power consumption of each block in the design and how it develops in smaller timeframes, instead of having an average estimate of the whole design.

The bottom-up estimation methods are the most accurate, but to achieve this accuracy they rely on a long characterisation stage. This stage makes use of a gate level representation of the design. The method thus requires both a synthesis and a characterisation before it is ready to make a power estimate at the RT-level.

The top-down methods, on the other hand, are less accurate and has no characterisation stage. This makes them faster, although some methods still require the design to be synthesised to create a model which can be used in the RT-level power estimation.

A bottom-up method is most suitable in design processes relying on IP reuse, where most IP blocks in a design are pre-existent and used more than once. It is good if one desires an accurate result, for example to check if power constraints are met. Making a macromodel of an IP block could also be a part of characterising it.

The top-down method is more suited if a fast and relatively accurate result is needed from the power estimation. This could for example be the case during exploration of the design space.

To check for power bugs, exploring the power consumption over time is key. This can be achieved by both methods. However, in VLSI many of the IPs already existing have been used in previous designs and they are likely to be error-free for that reason. Thus, a most realistic estimation method for a power bug checking system would be a top-down method.

Some hybrid of the two methods could possibly work well, making it possible to create macromodels for designs if an accurate result is needed and making use of already existing macromodels, but also giving quick estimates using top-down estimation for new designs/new parts of a design.

Nordic Semiconductor's motivation for RTL power estimation can be seen in Appendix A. They wish to monitor the power consumption of their designs at the RT-level during simulation to detect power bugs. They want violations of the power constraints to be reported and want the ability to log power over specified time frames. The top-down method could provide this functionality if it is accurate enough. To improve its accuracy macromodels could be made of existing IP blocks and integrated into the top-down power estimation method. In a possible master's thesis a top-down estimation method, where the accuracy could be increased by

using macromodels for existing IP blocks, is desirable.

8 Conclusion

The bottom-up and top-down are both good methods for RTL power estimation, however they suit different motivations.

When estimating power to see if constraints are met or for some reason wanting a result as accurate as possible the bottom-up method is most suitable. To integrate this method into a design flow it is highly reliant on VLSI as the characterisation stage is time consuming, making reuse a necessity to make it worth the trouble.

When estimating power to explore the design space and eliminating power bugs the top-down method is most suitable as it quickly gives you an estimate with high relative accuracy for the power consumption with some time resolution.

It could also be possible to combine the two methods, using top-down estimation in general and making macromodels of IP blocks after synthesis in the VLSI design process. The use of these macromodels when reusing the IP block could provide a more accurate RTL power estimation, while still keeping the efficiency high by making use of top-down estimation methods.

8.1 Future work on top-down power estimation

Some interesting papers did not make their way into this project due to lack of time to take them all into consideration. One of them is Landman and Rabaeys Dual Bit Type Method [15]. In this method the least significant bits of a signal is considered noise, while the most significant bits are seen as correlating with the power consumption and switching activity of the circuit. Another paper by Landman and Rabaey analyse the activity of the control path and its impact on power [23].

Bibliography

[1] S. Gupta and F. N. Najm, "Analytical model for high level power modeling of combinational and sequential circuits," in *Proceedings IEEE Alessandro Volta Memorial Workshop on Low-Power Design*, pp. 164–172, Mar. 1999.

[2] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*. USA: Addison-Wesley Publishing Company, 4th ed., 2010.

[3] J. Rabaey, *Low Power Design Essentials*. Integrated Circuits and Systems, Springer US, 2009.

[4] K. Chauhan, "ASIC Design Flow in VLSI Engineering Services - A Quick Guide." `https://www.einfochips.com/blog/asic-design-flow-in-vlsi-engineering-services-a-quick-guide/`, 2019. [Online; accessed 04-December-2019].

[5] A. Bogliolo, L. Benini, and G. D. Micheli, "Characterization-free behavioral power modeling," in *Proceedings Design, Automation and Test in Europe*, pp. 767–773, Feb. 1998.

[6] K. Brace, R. Rudell, and R. Bryant, "Efficient implementation of a BDD package," in *27th ACM/IEEE Design Automation Conference*, pp. 40–45, June 1990. ISSN: 0738-100X.

[7] H. R. Andersen, "An Introduction to Binary Decision Diagrams," p. 36.

[8] Ansys. `https://www.ansys.com/products/semiconductors/ansys-powerartist`, 2019. [Online; accessed 09-December-2019].

[9] Synopsys. `https://www.synopsys.com/verification/static-and-formal-verification/spyglass/spyglass-power.html`, 2019. [Online; accessed 05-December-2019].

[10] Mentor. https://www.mentor.com/hls-lp/powerpro-rtl-low-power/power-estimation, 2019. [Online; accessed 05-December-2019].

[11] Cadence. https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/power-analysis/joules-rtl-power-solution.html, 2019. [Online; accessed 05-December-2019].

[12] "IEEE Standard for Power Modeling to Enable System-Level Analysis," *IEEE Std 2416-2019*, pp. 1–63, July 2019.

[13] M. Nemani and F. N. Najm, "Towards a high-level power estimation capability [digital ICs]," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, pp. 588–598, June 1996.

[14] Dake Liu and C. Svensson, "Power consumption estimation in CMOS VLSI chips," *IEEE Journal of Solid-State Circuits*, vol. 29, pp. 663–670, June 1994.

[15] P. E. Landman and J. M. Rabaey, "Architectural power analysis: The dual bit type method," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 3, pp. 173–187, June 1995.

[16] S. Gupta and F. N. Najm, "Power Macromodeling for High Level Power Estimation," in *Proceedings of the 34th Annual Design Automation Conference*, DAC '97, (New York, NY, USA), pp. 365–370, ACM, 1997. event-place: Anaheim, California, USA.

[17] S. Gupta and F. N. Najm, "Power modeling for high-level power estimation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, pp. 18–29, Feb. 2000.

[18] S. Gupta and F. N. Najm, "Analytical models for RTL power estimation of combinational and sequential circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 808–814, July 2000.

[19] G. Bernacchia and M. C. Papaefthymiou, "Analytical Macromodeling for High-level Power Estimation," in *Proceedings of the 1999 IEEE/ACM International Conference on Computer-aided Design*, ICCAD '99, (Piscataway, NJ, USA), pp. 280–283, IEEE Press, 1999. event-place: San Jose, California, USA.

[20] Yi-Min Jiang, Shi-Yu Huang, Kwang-Ting Cheng, D. C. Wang, and Ching Yen Ho, "A hybrid power model for RTL power estimation," in *Proceedings of 1998 Asia and South Pacific Design Automation Conference*, pp. 551–556, Feb. 1998.

[21] X. Liu and M. C. Papaefthymiou, "A static power estimation methodology for IP-based design," in *Proceedings Design, Automation and Test in Europe. Conference and Exhibition 2001*, pp. 280–287, Mar. 2001.

[22] R. Bahar, E. Frohm, C. Gaona, G. Hachtel, E. Macii, A. Pardo, and F. Somenzi, "Algebraic decision diagrams and their applications," in *Proceedings of 1993 International Conference on Computer Aided Design (ICCAD)*, pp. 188–191, Nov. 1993.

[23] P. E. Landman and J. M. Rabaey, "Activity-sensitive Architectural Power Analysis for the Control Path," in *Proceedings of the 1995 International Symposium on Low Power Design*, ISLPED '95, (New York, NY, USA), pp. 93–98, ACM, 1995. event-place: Dana Point, California, USA.

Appendix A: Project motivation by Knut Austbø

We define "power bugs" as issues where the implementation violates power consumption requirements. These requirements are determined by power targets in the device specification (target power consumption for different operation modes/scenarios), and limitations of the power supply infrastructure (regulators, power rails, etc). Power bugs are inherently difficult to detect in simulation, as they typically don't cause functional errors. An example is an IP, that in an idle state keeps requesting a resource, even if it doesn't need the resource. This results in the unused resource being kept active, thus consuming more power than necessary. Such power bugs may not be detected until the device is in production, potentially resulting in expensive respins and delayed time to market.

The idea behind this project task is to make RTL simulation power consumption aware, i.e. modelling the power consumption of the design hierarchy of the device under test. The implementation needs to support the following features:

- Estimating the power consumption in RTL simulation
- Assertions to detect and report violations during simulation
- Logging of power consumption of specified time windows and/or complete duration of the simulation
- Average power consumption calculation of specified time windows and/or complete duration of the simulation
- Debugging/exploration capabilities that shows where in the design hierarchy the power is consumed

Appendix B: Project assignment

Power consumption is a key parameter in modern chip design. Exceeding the power target can result in costly respins and delayed time to market. It is therefore key to estimate and monitor the power consumption during the design process. In this project task, the student will develop a system for modelling power consumption in RTL simulation. The work can be divided in the subtasks:

1. Develop a template for modelling power consumption of IPs and other building blocks
2. Create a system for collecting data from these models and present the data in an informative manner.

SystemVerilog Classes will be used, so previous experience with classes in SystemVerilog, C++ or similar languages is an advantage. The project can be expanded into a Master's Thesis if the student wishes to continue with the project.