Low Power Design Guide

Version 30.06.00

Dipl.-Inform. Frank Poppen
eMail: Poppen@OFFIS.De

Low Power Design Guide

1 Introduction

In the early 1970s designing for high speed and minimum area, especially in memories, were the main design constraints. Most of the EDA tools were created to meet these criteria and papers published advertised lower delays and smaller structures. Power consumption was a part of the design process, but was less visible. However, reaching structures below 0.18 µm and having high-performance-chips work in portable devices, power dissipation has become the main design concern in these applications.

This paper addresses designers and will serve as a guideline for system level specification and the remnant of a Low Power Design Flow [9]. One has to accept that it was not possible to summarize the contents of any work done in this working field in this paper and even the carefully selected topics remain superficial. If more information is needed, have a look at the literature listed in appendix D and contact OFFIS. The latter is the intended way, since this paper should be seen as a living document. It will be constantly adapted during the design phase to show more details in the relevant domains. An additional planned chapter is e.g. on how to operate existing design tools, presenting example synthesis scripts etc. Supplementary, OFFIS is going to install a web page at www.lowpower.de, which will contain novelties, tools, congresses, etc. OFFIS is planning to offer a workshop on low power. The web page will show information about this, too. The techniques presented are mainly directed to digital parts of the design. Optimizing software for low power is left out, since this is UPC's domain.

The further document is structured as follows: Chapter 2 gives an overview on the sources of power consumption. This basic knowledge can't be used directly to improve power usage, because the level of abstraction is too low (transistor level), but will help to understand the problem. Chapter 3 shows up several alternatives for power optimization and chapter 4 is an excursion on batteries and how knowledge about their behavior can improve power supply.

2 Sources of Power Consumption

When designers recognized power consumption as a design constraint, simple models were created. Power per MHz is still a commonly used representation of a component. With a closer look at power dissipation, it becomes obvious that the subject is not that simple. Electric current is not constant during operation; peak power is an important concern. The device will malfunction due to electro-migration and voltage drops even if the average power consumption is low.

This chapter gives an overview of the sources of power consumption. A formula for average power is given in equation 1.

equation 1: $$P_{avg} = P_{dynamic} + P_{short} + P_{leakage} + P_{static}$$

The four components are dynamic, short-circuit, leakage and static power consumption. The share of each part depends on the application and technology. In several cases e.g. $P_{leakage}$ might be negligible, in others not. Have a look at 2.1 to 2.4 for a detailed explanation.

2.1. Capacitive Switching Activity

Switching activity is a measure for the number of gates and their outputs that change their bit-value during a clock cycle. To toggle between logic zero and logic one capacities have to be discharged and charged. The electric current i_d that flows during this process causes a power dissipation $P_{dynamic}$. The current is dependent on the capacitive output load C_{out} and the supply voltage V_{dd}. In equation 2 this behavior is reproduced.

equation 2: $$P_{dynamic} = KC_{out}V_{dd}^2 f$$

K is the average number of rising transitions during one clock cycle and f the clock frequency. It is interesting to see, that the supply voltage has a quadratic effect on dynamic power consumption. Reducing power supply will therefore have the greatest effect on saving power, taking into account that typically $P_{dynamic}$ is responsible for 80% of P_{avg}. Unfortunately, designers don't have the freedom to choose the parameters arbitrarily. The chosen technology and the given timing constraints[1] set a minimum and maximum range for the acceptable values.
Until now, we assumed that all charge drawn from the power supply is collected by the output capacitances. Some current flows directly from power to ground and surmises a short-circuit current during bit switching.

2.2. Short-Circuit Currents

CMOS circuits consist of a pull-up and pull-down network (see figure 1), which have a finite input fall/rise time larger then zero. During this short time interval, when the pull-down and pull-up network are conducting both, a current i_{cc} flows from power to ground. This current is called short-circuit current.

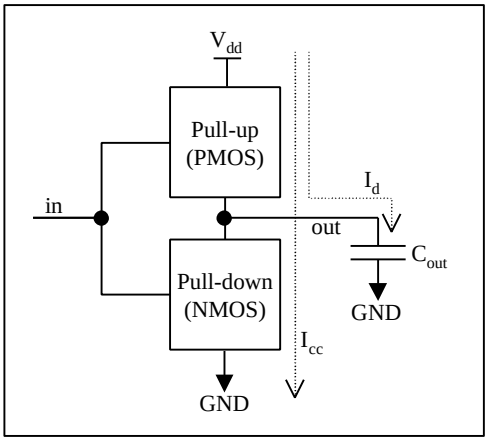

figure 1: CMOS circuit

equation 3: $$P_{short} = K\frac{b}{12}(V_{dd} - 2V_T)^3 ft$$

[1] Reducing V_{dd} slows down charging of capacitances.

The resulting power consumption is calculated by equation 3. The origin of the formula is carried out in [5]. b is the gain factor of a MOS transistor, V_T its threshold voltage and t is the rise/fall time of the gate inputs.
Since the parameters of this equation can not be tuned by designers[2], P_{short} is not going to be taken into consideration during low power design. This is no set-back because several authors observed that short-circuit power dissipation is usually a small fraction of total power usage, around 10%.
Dynamic and short-circuit power depend on switching activity represented in the parameter *K*. As an effect, no power should be lost during idleness of a CMOS circuit, when *K* is zero. The existence of leakage currents shows another piece of reality.

2.3. Leakage Currents

In [8] equation 4 is derived, which shows the structure of leakage power.

equation 4: $$P_{leakage} = \left(I_{diode} + I_{subthreshold}\right) \cdot V_{dd}$$

I_{diode} refers to the currents flowing through the reverse biased diodes that are formed between the diffusion regions and the substrate. $I_{subthreshold}$ refers to currents flowing through transistors that are non-conducting. Again, designers can not modify this parameter.

2.4. Static Power

Static power dissipation depends on a current flow from power to ground during idle time unlike short-circuit power dissipation, which occurs only during switching activity. NMOS circuits show high static power consumptions because power is connected directly to ground when a gate's output denounces logic zero, resolving into a great short circuit current. In well designed and flawless CMOS designs static power should be zero. If not, the reason could be e.g. a bus conflict where multiple drivers attempt to drive a signal to different logic values.

[2] The designers would need to act at transistor level of design space but our design flow is specified from system level to gate level.

3 Low Power Design Methodologies

In this chapter the designer receives practical advise for low power design. This document must not be understood as a complete implementation guide. It is an overview of known techniques gathered from [1] - [8]. This gives an idea of what methodology is applicable. The reader should fall back on the known literature for more details or request more information from OFFIS, if needed.
Chapter 3.1 starts at a low level of the design space and is therefore not of much use for the designer, since the defined designflow ends at the gate level. Techniques that effect lower levels are out of the scope of our design flow. Still, the given information is relevant for a complete understanding of the matter. In 3.2 switching power reduction is addressed. Shown are promising techniques for low power design. Chapter 3.3 depicts proceedings which effect the architectural level and focus on power down modes. Last, 3.4 handles the system level. The advantage of SoC to power dissipation is introduced. Note that the higher the level of abstraction at which a methodology is applied, the more promising are the effects on saving power (compare with figure 2).

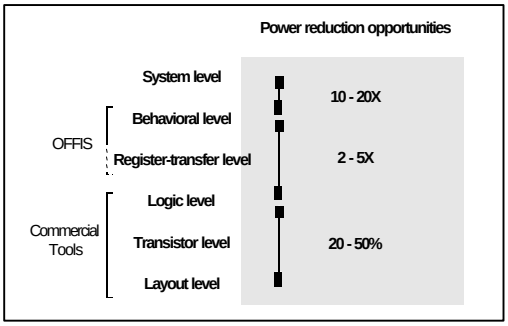

figure 2: Power Reduction Opportunities

3.1. Adapting Process Technology

This chapter was added to give a complete picture on low power techniques. Defacto, it is of no relevance for the designers, as the described effects base on a level of design abstraction which is not in their scope.
In 3.1.1 reducing capacitance is the topic and chapter 3.1.2 illuminates leakage power. 3.1.3 handles the promising methodology of lessening power supply voltage. Power savings through higher density of integration is described in chapter 3.1.4.

3.1.1. Reducing Capacitance

In equation 5 C_{out} is described as the sum of three capacitances:

equation 5: $$C_{out} = C_{fo} + C_w + C_p$$

C_{fo} is the input capacitance of fan-out gates, C_w the wiring and C_p the parasitic capacitance. For deep sub-micron technologies C_w is the most dominant component and unfortunately the hardest one to estimate, too. Complex effects like “cross-talk” have to be considered. Designers are not in charge of placing and routing a design below gate level and have therefore no major influence. Only layouter and technology vendors might be able to have greater effects on this parameter.

3.1.2. Reduce Leakage Power

In the ordinary $P_{dynamic}$ outweighs $P_{leakage}$ but the ratio might tip over, if the design is idle most of the time and switching activity is low. Unluckily these effects are out of our design flow. The technology vendor is in charge of the design flow at this level of abstraction.

3.1.3. Reducing Supply Power

Having a look at equation 2 it is apparent, that reducing supply voltage is the most promising way of saving power since its influence is quadratic. Slowing down switching speed is the penalty for this technique as can be deduced from equation 6 taken from [8].

equation 6: $$T_d = \frac{C_{out} V_{dd}}{I} = \frac{C_{out} V_{dd}}{h\left(W/L\right)(V_{dd} - V_t)^2}$$

T_d is the delay of a CMOS inverter, h is a technology-dependent constant, W and L are respectively the transistor width and length and V_t is its threshold voltage.
Usually a circuit is designed to meet certain timing constraints which will be violated when the supply voltage is reduced. The solution is called “architecture-driven voltage scaling”. The level of concurrency is raised by adding more hardware to the design. Typical methodologies are pipelining and parallelization. This eases the timing restrictions. In spite of having more hardware that is consuming power, the over all power dissipation is reduced because of the quadratic influence of V_{dd}.

3.1.4. Higher Density of Integration

By minimizing the scale of a circuit, its capacitances and therefore its dynamic power dissipation is reduced. The technology is fixed to the structures of the vendors technology in the project anyway, so this paper is not going into details.

3.2. Reducing Switching Activity

In chapter 2.1 the importance of $P_{dynamic}$ for P_{avg} is explained. To reduce power dissipation effectively low power methodologies should affect this source. Since designers have no influence on V_{dd} and only a minor one on C_{out}, switching activity is the remaining rudiment.
Methodologies for reducing the switching activity are given in the following chapters. Glitches are unnecessary transitions. They are covered by chapter 3.2.1. Chapter 3.2.2 handles minimization of operations in algorithms. In most cases, several different calculations compute the same function. Those, which content the least or the cheapest – regarding power consumption – operations, should be chosen for a low power design. Chapter 3.2.3 treats busses and shows several styles for data representation, which result in a lower signal activity. That optimization can be achieved by an intelligent scheduling and binding, which is shown in chapter 3.2.4.

3.2.1. Minimization of Glitches

Gates’ delays are often assumed zero to simplify estimation. In this way an important aspect of reality, glitch power, is left out. In a static logic gate, the output or internal nodes can switch before the correct logical value is being stable. Imagine an AND-gate with two inputs of different delay time and the transition 01 $\rightarrow$ 10. At a zero delay gate the output would be a stable logic zero but in our example the first port has a lower delay, which forces the output to

a temporary logic one, after which it stabilizes on zero again. The power lost during this unnecessary switching activity is called glitching power loss.
The influence of glitch power is illustrated in figure 3. Leaving out glitch power would make the left architecture the less power consuming choice. Amazingly, the alternative on the right side with two adders is the more economical design.

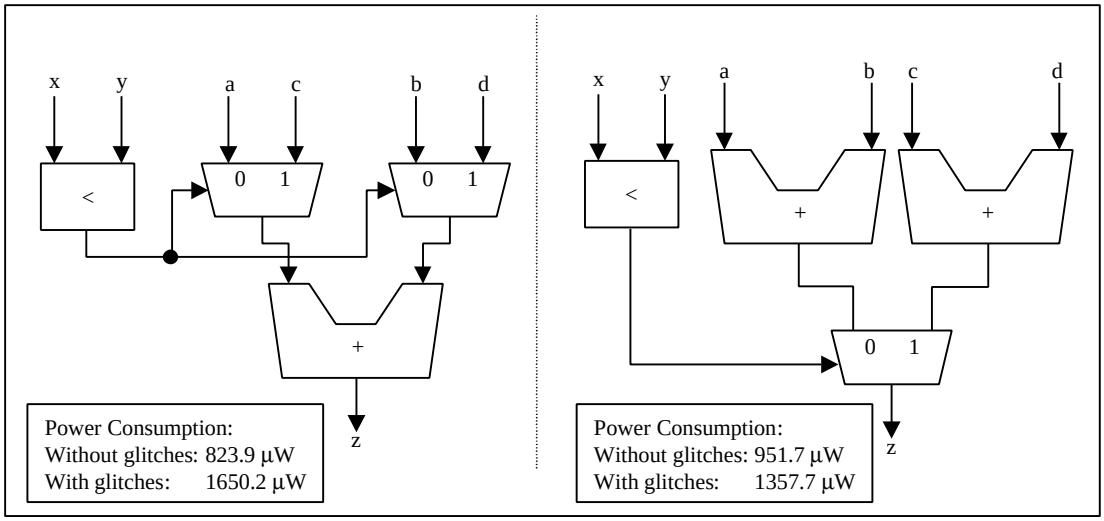

figure 3: Alternative architectures that implement the same function: Effect of glitching

Glitches of a signal node are dependent on the logic depth of it. Generally, nodes that are logically deeper are more prone to signal glitches. One of the reasons is that the transition arrival times are spread longer due to delay imbalances. Also, a logically deep node is typically affected by more input switching and therefore more susceptible to glitches.
One way to reduce glitches is to shorten the depth of combinational logic by adding pipeline registers. This is very effective especially for data-path components such as multipliers and parity trees.

3.2.2. Minimization of the Number of Operations

Minimizing the number of operations to perform a given function is critical to reducing the overall switching activity. To illustrate the power trade-offs that can be made at the algorithmic level, consider the problem of compressing a video data stream using the vector quantization (VQ) algorithm. Detailed information on VQ can be found in [11]. The basic operation is shown in equation 7:

equation 7: $$D_i = \sum_{j=0}^{15} \left(X_j - C_{ij} \right)^2$$

where X_j are the elements of the input vector and C_{ij} are the elements of the codebook vector. Two more algorithms are illuminated in [8]: tree search and differential codebook (equation 7 represents the full-search-methodology). Have a look at table 1 which shows the number of operations needed for each algorithm.

Algorithm	# of Memory Access	# of Multiplications	# of Adds	# of Subs
Full Search	4096	4096	3840	4096
Tree Search	256	256	240	264
Differential Tree Search	136	128	128	0

table 1: Computational complexity of VQ encoding algorithms.

It is obvious, that 4096 subtractions will use a lot more power then zero. The numbers of the other operations change by a factor of 30.
The reason for not going into any details is, that the optimization techniques used in the example are not commonly adaptable to other application. Each algorithm will have to be explored in its own domain. This results in some intellectual handwork. The example shows that this manpower is well spent.

3.2.3. Low Power Bus

Busses are known for their heavy loads, long interconnects and therefore their large capacitances. This is due to their connections to large cores spread across the die of a SoC. Reducing the capacitance is normally not possible and reducing the switching activity is the only chance of reducing power loss. Therefore, coding the transmitted data for minimum switching activity is the methodology of the choice. One-hot coding, gray-coding, bus-inversion-coding and two's complement versus sign magnitude are introduced in the following.
One-hot coding is a simple, redundant coding style. The original bus with a bit width of n is capable of transmitting 2^n different words. Each word is mapped to a single wire in the one-hot coding. A bus using this methodology would need $m = 2^n$ wires. m-n is the degree of redundancy of the encoding. It is ensured, that only two bits will change between the transmission of two data words. Practically, the one-hot is of no relevance because the area (number of wires) is growing exponentially with n.
Another encoding strategy is gray-coding. A gray code sequence is a set of numbers in which adjacent values have one bit difference only. This is of use when the data being transmitted over a bus is sequential and highly correlated. In this case, the number of transitions between two words broadcasted on the bus approaches two. Address bus accesses for instruction fetches are highly correlated and a good example where gray-coding will have good results. In table 2 an example is given. Notice that gray-coding does not have a high redundancy like one-hot.

DECIMAL VALUE	BINARY VALUE	GRAY CODE
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

table 2: Binary and Gray-code representation

A third example is bus-inversion-coding [12]. Before transmitting a data word S_{i+1} the previous word S_i is compared with it. Either $\overline{S}_{i+1}$ or S_{i+1} is transmitted, depending on which representation results in fewer transitions and where $\overline{S}_{i+1}$ is the bit-wise inverse of S_{i+1}. An additional wire indicates how to interpret the data. For bus width smaller then 8 a reduction of activity down to 25% can be achieved. It is possible for wider buses to divide them, e.g. a 32 bit bus into four parts and encode them separately. This makes more then one additional line necessary.
Another choice of data representation is two's complement vs. sign magnitude. In most designs two's complement is chosen because implementation of additions and subtractions are easier to implement. The problems occur, if the dynamic range is much smaller than the maximum possible value and digits around zero are common. In this case the most significant Bits (MSB) will produce transitions very often, when the sign changes. In applications like this sign magnitude is a good alternative.

3.2.4. Scheduling and Binding Optimization

At the behavioral level a design is typically described as a control-data-flow graph (CDFG). A scheduling algorithm assigns each operation of the CDFG to a designated time slot, taking data dependencies into account. A proceeding, which is power-oriented, is presented in [14]. The intuition behind this method is that operations must be scheduled, so that resources which are not performing useful computations in a given control step can be shut down. In this way the methodologies of chapter 3.3 become more applicable.
It has been said, that each operation has to be executed on a resource. The correlation of operations to resources is called binding. Operations that are not scheduled in the same control step can be bound to the same resource and executed sequentially. This is called resource sharing. Due to resource sharing it is possible to implement the same functionality on a reduced set of resources, e.g. one adder instead of two. One must not expect to reduce power by minimizing resources. It is true, that the required chip area will be halved, but since the activity of the one adder is doubled (two additions have to be executed on it instead of only one) its dynamic power dissipation will double as well. How can a binding be optimized for low power then? Scheduling and binding should be chosen to utilize data correlations. Binding operations of a correlated data streams to the same resource will reduce switching activity.
OFFIS developed the high level synthesis tool ORINOCO® (compare with [10]) to automate the determination of a low power scheduling and binding.

3.3. Power Down Modes

Systems have to be designed to meet certain constraints in which they have to operate. Since these limits normally show worst case situations, the system typically is not working at maximum possible performance. Parts of the chip are idle, do not add any functionality to the design at the time, but still consume power. The reasons are unnecessary changes on the inputs of the unused devices and the load they add to the clock-signal, which continuously toggles whether the devices are processing or not. Reasonably these parts should be turned off.
This chapter offers several proceedings for this case. They are sorted by granularity. If great parts of the system are idle for a long time, power supply shutdown is the methodology of the

choice (chapter 3.3.1). For smaller components, or such which must not loose register values, chapter 3.3.2, which handles clock gating, might be more applicable. Chapter 3.3.3 describes the least effective but also least invasive methodology, insertion of enabled flip-flops. Last but not the least shut, down of memory is treated in its own chapter 3.3.4.

3.3.1. Power Supply Shutdown

Shutting down power supply reduces power dissipation to zero. This is the most effective way to save power in idle modules. Several conditions have to be fulfilled to employ this methodology.

1. The power switch will have to be well designed. A resistance and delay value is tied to a real switch. A straight forward implementation would be a transistor with a low ON resistance. Therefore, its width has to be increased, which results in a large capacitance. Buffering circuitry is needed to operate the switch at a satisfactory performance.
2. It takes a delay time of *DT* before supply voltage stabilizes in a switched back on module. This makes the methodology applicable for components with an idle time greater then *DT* only.
3. The design must not contain any storage units like registers or memory because their values would be lost during power down. It is possible to add extra logic to save and later restore the data, but the logic and power overhead for this proceeding has to be well examined.
4. Powering down and up will result in transient noise and voltage drops in a carefully designed power supply grid. These effects must be adequately shielded to avoid functional failures.

The numerated points suggest, that power supply shutdown is applicable for a very coarse level of granularity only and one has to realize that it is very invasive and disturbing to a design.

3.3.2. Clock Gating

Instead of switching off power supply, the clock signal may be halted in idle devices. This reduces switching activity and therefore dynamic power consumption to zero. Inserting clock gates is not as great of an interference to the design as power supply shutdown and can be used on components with lower granularity than mentioned in 3.3.1. This makes clock gating applicable for applications where power shut down is no alternative. Clock gating won't lessen power dissipation to zero since leakage power is unaffected.

The designer has to take into account that the gate increases clock skew and makes testing more complicate. Lastly, glitches on the switch's control signal must be prevented. E.g. a glitch could cause a temporarily false clock turn off/on, which might add an extra rising edge to the clock signal behind the gate. Preservation of the circuits behavior is not guaranteed!

Synopsys advertises their tool power compiler to handle insertion of clock gating automatically (compare with [10]).

3.3.3. Enabled Flip-Flops

As clock gating can be seen as a softer alternative to power supply shut down, enabled flip-flops are the next less aggressive (and less effective) strategy. Registers are replaced by a representative with an enable signal. By enabling these representatives, they behave like general registers. Disabled, the flip-flops' outputs are not changing, which reduces switching activity in the circuit. The most active signal, the clock, is still active though, ensuing a great deal of power dissipation.

Recapitulating it can be said, power management based on enabled flip-flops can be beneficial, but an implementation based on gated-clocks is fundamentally superior.

3.3.4. Memory Partitioning

Farrahi and co-authors [6] propose a memory partitioning (also called segmentation) scheme that reduces power by exposing idleness in memory access. The functionality of memory is to store data when it is written and return it when read. Farrahi suggests to view memory not as a monolithic resource but as a collection of independent memory segments. Each segment has its own clock and refresh signals. Whenever a memory segment is idle, it can be put in a sleep mode where the clock is halted or no refreshes are transmitted. Memory is idle, when no useful information is stored in it. Be aware that memory is not idle, when it is not accessed. It might store vital information which would be lost when the memory is turned off. It might store unimportant information though. A lifetime can be assigned to each variable in a memory element. It defines a time interval which starts when a variable is written, and ends when the variable is last read. A segment is called idle, when it contains no live variables. The partitioning technique attempts to store variables which have overlapping lifetimes in the same segment. Due to this approach, idle time of memory segments is increased and power dissipation is reduced.
Memory segmentation is a binding problem that assumes the knowledge of scheduling information.

3.4. System Design

Regarding to figure 2 on page 7, system level low power design techniques should be most promising for reducing energy consumption. Two methodologies are denoted in this section. Chapter 3.4.1 focuses on low power hardware-software partitioning and shows possible power savings of approximately 77%. Chapter 3.4.2 handles chip's I/O communication, which is responsible for up to 33% of overall system power consumption in typical designs.

3.4.1. HW/SW Partitioning

The MicroPP Plus will be an embedded system with two processor cores, a controller and a DSP. Services can be implemented either in software running on these cores, or in dedicated hardware. In our design flow this will be decided during the step of hardware-software partitioning. This process has great influence on system power. This is illustrated in the following example:
Specific hardware is generally more efficient. 330mW are consumed to perform an addition using a SPARClite processor core in an exemplary technology (0.32µm / 1.8V / 16.8 MHz). A custom adder in the same technology consumes only 2mW plus additional communication overhead. The author of [15] presents the HDTV chromakey algorithm with 22000 lines of code. Only 15 lines, the critical loops, are implemented in hardware, which results in an energy saving of 77%. These are promising results and confirm the predication of figure 2.

3.4.2. Integration of Chip Components

Implementing systems using present day technology results in third or more of total power being consumed at the chip's input/output (I/O) ports. The larger capacitances of chip's boundaries compared to internal gates and higher voltages are the reason for this observation. Typical values for internal capacitances reside around 10's of femtofarads, where I/O pins reach dimensions of 10's of picofarads. Nowadays supply voltages for chip-cores tend to be lower then 2.0V. In industrial systems not all components of a design might be state of the art

and require higher voltages or technical constraints require them. Still, these different components have to communicate over their I/O. This makes dual voltage systems (lower voltage for the cores – higher voltage for I/O) quite common.
From equation 1 the relevance of high capacitances and voltage to dynamic power is known. This makes reduction of switching activity on I/O ports an important task. The methodologies of chapter 3.2 can be applied. Implementing a cache for the special case of external memory helps to reduce I/O traffic and should raise performance as a side effect. But the better way is, decreasing I/O as much as possible by integration of all systems on one chip (SoC). Submicron technology makes this highly integrated circuits possible.

4 Techniques to Maximize Battery Life

A battery is often seen as a storage component that contains a certain amount of power which it is able to release. Reality, once again, is not that simple as is going to be shown in the following.
The power that can be drawn from a battery is not depending on its charging only. To illustrate this, have a look at figure 4 (the data has been taken from Panasonic Nickel-Metal Hydride Batteries Handbook):

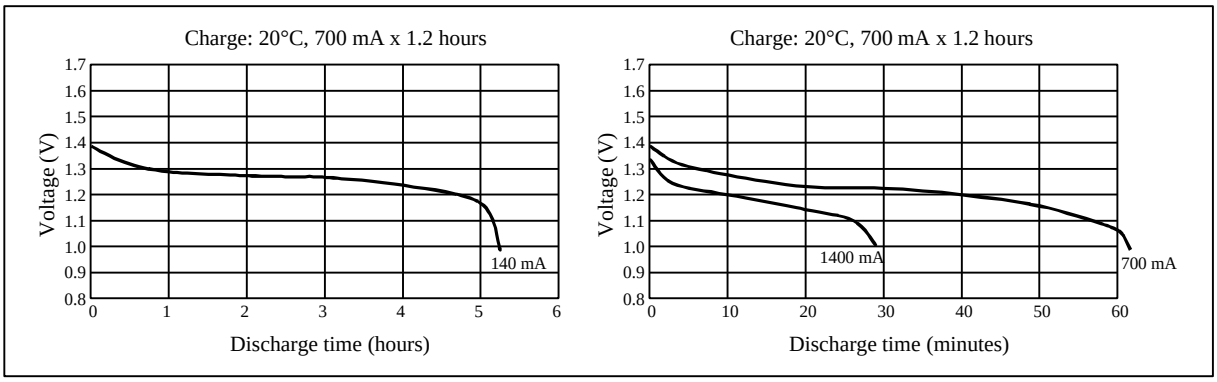

figure 4: Battery discharge lines

The figure shows a battery being charged under the same circumstances but being discharged at three different rates. Have a look at equation 8 to remember the physical magnitude *energy* respectively *work*.

equation 8: $$W = P \cdot t = U \cdot I \cdot t$$

Where W is the drawn energy over the time t, P stands for power and U for the voltage at which the battery is discharged with the current I. Notice that I is constant during discharging. The battery is considered empty, when U drops below 1 V. By applying this knowledge to the diagrams, we get three totally different values for W:

∅U [V]	I [MA]	W [WS]
1.16	1400	2630
1.24	700	3230
1.26	140	3370

table 3:Evaluation of figure 4

What do these numbers tell us? The total power that can be drawn from a battery is highly depending on the average current. The higher *I*, the more power is lost (22% in the given example). Therefore average current should be reduced to a minimum. Further on, cells recover when allowed to relax between discharge. It makes an energy source more efficient, when it is discharged with current bursts instead of a steady flow.
This is as far as this paper is going into the subject. The field of battery design takes complete books to cover. This paper can be enhanced on demand to face more details.

5 Conclusion

It was shown, that the most promising approach to low power design starts at the system level. Over all system power dissipation can roughly be divided into three parts: 33% I/O, 33% cores/memory and 33% control logic. Energy consumption breaks down into dynamic, static, leakage and short circuit power dissipation. Dynamic power, with a share of 80% is the starting-point for most of the introduced methodologies. The other sources are out of the scope of designers, since they would need to act on a rank of abstraction below gate level.
This paper delivered several ideas and methodologies. They have to be evaluated for its relevance. The details of the promising one will then be expanded in an updated version of this document.

Appendix

A Glossary

- Binding [13]:
  A binding is the explicit definition of a mapping between an algorithm's operations and resources e.g. adders, multipliers, memory, etc. A binding may imply that some resources are shared. Operands sharing a resource can not execute concurrently.
- Clock Gating [1], [2]:
  A gated clock is a clock signal that can be stopped by a logic control signal which is computed during normal operation. When the gated clock is stopped, activity and therefore switching power consumption in the connected logic is reduced to zero. Notice that, differently from the supply shutdown case, power dissipation is not nullified. Leakage power is still dissipated.
- Characterization [9]:
  Characterization is the first step in creating a model. E.g.: For a multiplier power consumption model a reasonable number of entities with different parameters will be simulated. Parameters are values like bit width, architecture, basic technology, etc.
  The process of gathering this data is called characterization.
- Instrumentation:
  The process of adding additional functionality to a HDL specification to record activity on certain components, buses and signals. This functionality is needed only during simulation to make a subsequent power estimation possible. Throughout synthesis the original, non-instrumented code is used.
  ORINOCO, a tool developed by OFFIS, automates instrumentation of behavioral VHDL descriptions. Since Verilog is used in our design methodology, instrumentation has to be achieved manually at RT level. ToggleAnalysis, also developed by OFFIS, handles instrumentation of Verilog description at gate level.
- IP:
  IP is the acronym for Intellectual Property. In the frame of this paper, IP refers to the components specified in Verilog during the duration of the project.
- Model:
  A model of a component/IP is created, if it's behavior is too complex or unknown during simulation/estimation. A parameterized functional model of a class of components/IPs will reduce simulation/estimation time by matters of magnitude. One will have to accept an appreciable error depending on the quality of the model.
  The first step in building a model is the characterization. Secondly, model fitting must be executed.
- Model fitting [9]:
  After characterization the model of e.g. a multiplier has to be adapted to the gathered characterization data, which serve as an anchor. The created model will be correct at these points. Values in between are calculated by interpolation.
- Scheduling [13]:
  Whereas a sequencing graph, which can be developed from an algorithm's operands, prescribes only dependencies among the operations, the scheduling of a sequencing graph determines the precise start time of each task. The start time must satisfy the original

dependencies, which limit the amount of parallelism of the operations, because any pair of operations related by a sequence dependency may not execute concurrently.

- SoC:
  System on a Chip. Instead of having a chipset of controllers, memories, bridges, codecs, etc. nowadays submicron technologies allow the cost-efficient integration of an entire system on one die.
- Technology:
  Describes the chosen

B Register of Illustrations

C Register of Equations

D Register of Literature

[1] Gary K. Yeap, “Practical Low Power Digital VLSI Design”, Kluwer Academic Publishers, 1998

[2] Luca Benini & Giovanni De Micheli, “Dynamic Power Management, Design Techniques and CAD Tools”, Kluwer Academic Publishers, 1998

[3] Abdellatif Bellaouar & Mohamed I. Elmasry, “Low-Power Digital VLSI Design, Circuits and Systems”, Kluwer Academic Publishers, 1995

[4] Anand Raghunathan, Nirja K. Jha and Sujt Dey, “High-Level Power Analysis and Optimization”, Kluwer Academic Publishers, 1998

[5] H. J. Veendrick, “Short-Circuit Dissipation of Static CMOS Circuitry and its Impact on the Design of Buffer Circuits”, Journal of Solid-State Circuits, vol. SC-19, no. 4, pp. 468-473, August 1984

[6] M. Farrahi, G. E. Tellez and M. Sarrafzadeh, “Memory segmentation to exploit sleep mode operation”, Proceedings of the Design Automation Conference, pp. 36-41, June 1995

[7] W. Nebel and J. Mermet, “Low Power Design in Deep Submicron Electronics”, Kluwer Academic Publisher, 1997

[8] A. Chandrakasan and R. Brodersen, “Low Power Digital CMOS design”, Kluwer Academic Publisher, 1995

[9] Frank Poppen, “Design Flow MicroPP Plus Documentation”, (unpublished internal paper), 2000

[10] Frank Poppen, “Tools for Power Estimation”, (unpublished internal paper), 2000

[11] A. Gersho and R. Gray, “Vector Quantization and Signal Compression”, Kluwer Academic Publishers, 1992

[12] M. Stan and W. Burleson, “Limited-weight Codes for Low-power I/O”, 1994 International Workshop on Low-power Design, pp. 209-214, April 1994

[13] Giovanni De Micheli, “Synthesis and Optimization of Digital Circuits”, McGraw-Hill Series in Electrical and Computer Engineering, 1994

[14] J. Monteiro, S. Devadas, P. Ashar and A. Mauskar, “Scheduling Techniques to Enable Power Management”, DAC-33:ACM/IEEE Design Automation Conference, pp. 349-352, 1996

[15] Anand Raghunathan, Sujit Dey, Arkady Horak, Trevor Mudge and Kaushik Roy, “Low-Power System Design: Applications, Architectures and Design Methodologies”, 37th Design Atomation Conference, 2000