# Gate-level study of the break-even point for fine-grained power gating

Anja Niedermeier, Jos Hulzink, Frank Bouwens, Jos Huisken, and Kjetil Svarstad

***Abstract*—Power consumption in digital systems is a crucial design factor. The dynamic switching power is still dominant at 90nm and above, however, the leakage component of power will become an increasingly important issue in low power design in the future due to the downscaling of technology. Power gating is a seemingly simple method for reducing the leakage current, nonetheless, the constraints and limitations as to what designs best lend themselves to this method are not yet well known and still under research. The work reported herein is a detailed analysis of the impact on fine-grained power gating, and, in addition, a novel methodology for the implementation of power gating in the datapath of a processor is developed. This methodology was tested on a VLIW processor design which was subsequently synthesised and placed and routed. Extracted power consumption values clearly demonstrate that the overhead is mainly determined by additional modules such as the power manager and the isolation cells during active mode and, contrary to the present orthodoxy in the field, not by the energy required to switch on a power domain. Furthermore, it is demonstrated that fine-grained power gating only leads to energy savings when power domains have a significantly low duty cycle and a low number of output signals.**

## I. Introduction

Optimisation of power consumption in digital systems is becoming increasingly important. Especially, in portable embedded devices, for example in medical applications, it is a major constraint in the design process. While power consumption is dominated by dynamic power for 90nm and above, leakage power consumption is expected to gain more and more impact in both absolute numbers [1] as well as in power consumption per area [2] in sub-90nm designs. Therefore, the interest for methods for leakage power minimsation is rising. One promising method is power gating ([3], [4]), where idle blocks are shut off completely, thus saving leakage power.

In this work, a detailed analysis of the benefits and costs of fine grained power gating is performed. Also, a methodology to partition the datapath of a processor into power domains and a workflow to implement power gating is presented. The presented power consumption values were obtained for the typical case ($1.2V, 25°C$) after place and route (P&R) for 100 MHz using a TSMC 90nm library.

The remainder of the paper is organised as follows. The related work is presented in Section II. In Section III, the background about power consumption in CMOS circuits and the principle behind power gating are presented. In Section IV, the analysis of the break-even point for fine-grained power gating is performed. In Section V, the implementation of power gating is shown, followed by a discussion of the results in Section VI. Finally, the conclusions are drawn in Section VII.

## II. Related work

Power gating is a commonly used technique for power management on system level [5] where complete components of a chip, like a processor or memory banks, are switched off. However, it is still an active research topic on a more fine grained level like on the datapath of a processor.

In [6], an exploration of the potential of power gating applied on the level of execution units in the datapath is performed. Also, an analytical equation for the break-even point is derived. For the estimation, the authors use a superscalar processor model. In their analysis for the break-even point the authors assume the power consumed by the power switch to be the only source for the energy overhead. The authors conclude that for an idle period of 10 clock cycles, power gating can bring benefits.

The authors of [1] also perform an analysis of the break-even point for power gating. They include besides the power switch also additional required decap area in their model. They conclude that the overhead which is caused by additional dynamic power consumed by the switch and the additional decap is too high for 130 nm technology, but they assume benefits for future technology.

In [7], an implementation methodology for power gating and an analysis of the overhead are presented. The authors base their methodology on exploiting existing clock-gating control signals. Based on the clock-gating domains, they provide an algorithm to partition the system automatically into power domains. Also the control signals for power gating are derived from the clock-gating control. In their analysis of the overhead, they only consider the power switch. They apply their methodology on a 32-bit RISC embedded CPU and performed synthesis and P&R. Afterwards they performed power analysis by using Toshiba 90nm device models. They conclude that significant amounts of leakage power can be saved at a reasonable area penalty.

In [8], a more detailed analysis of power gating than in the previous papers is presented. The authors include in their analysis of the break-even point in addition to the leakage power savings, power mode transition energy and sleep transistor size also the performance degradation and power mode transition time. They implement a power gated design with 65 nm STMicroelectronics technology and present power numbers

extracted after P&R. They conclude that they can achieve up to 75 % leakage power savings.

To the best of our knowledge, we are the first ones to perform an analytical analysis of the impact of power gating including the energy overhead due to additionally required modules like isolation cells or a power manager. Moreover, we derive an equation for the break-even point including those factors. Our analysis is based on power figures obtained on a post-P&R netlist for 90nm TMSC of a processor with power gating implemented in the datapath.

## III. Background

In this section, the different sources of power dissipation are explained. Then, the principle of power gating will be presented.

### A. Power dissipation in CMOS

The average power dissipation $P_{avg}$ in CMOS circuits can be described by the following equation:

$$P_{avg} = P_{short} + P_{dynamic} + P_{leak} \quad (1)$$

$P_{short}$ is caused by short circuit currents that occurs when both the NMOS and PMOS transistor are in their conductive state for a short time during transitions. $P_{dynamic}$ represents the dynamic power due to switching activity in the circuit. $P_{leak}$ represents the leakage power.

### B. Power Gating

A very good overview about power gating can be found in [3]. In the following, the important design issues are briefly introduced.

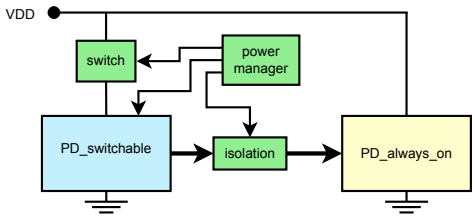


Figure 1. General power gating scheme

A general overview of a power gated system is illustrated in Figure 1. *PD_switchable* represents a switchable power domain. It has outputs to another power domain (*PD_always_on*) which connected to an always-on voltage supply in this example. The outputs of *PD_switchable* are isolated to prevent unknown signals propagating through the design when *PD_switchable* is switched off. A power switch is inserted between the voltage supply *VDD* and *PD_switchable* in order to allow *PD_switchable* to be disconnected from *VDD*. Furthermore, a power manager is integrated into the system to control the power gating procedure.

To cut off the power supply of a power domain, there are two possibilities. Either, a header or a footer switch is used. The header switch is illustrated in Figure 1. It is placed between the supply voltage and the power domain, thus introducing a virtual supply voltage. A footer switch is placed between the module and ground, thus introducing a virtual ground.

When a power domain is switched off, it is unknown which value the output has. Therefore, the outgoing signals have to be forced to either *0* or *1* by using special isolation cells.

One of the challenges with power gating is that registers lose their internal state when they are switched off. When the stored value still has to be present after a shutdown period, the register's state can be retained using special state retention (SR) registers. In addition to their main latch, they have an SR latch connected to a separate supply voltage which is not switched off when the register is switched off. This extra latch is used to store the value when the register is switched off. After switching the power domain back on, the stored value has to be transferred back to the main latch.

The power manager is in charge of providing the control signals to the switches, isolation cells, and, if present, the SR registers. It can be implemented as dedicated hardware module or in software. In Figure 1, the power manager is implemented in hardware. The control sequence for a system with SR can be seen in Figure 2. When no SR registers are present, the save and restore signals are not required. Instead, when registers are included in the power domain, a reset signal has to be provided before de-isolation to set the power domain in a known state.

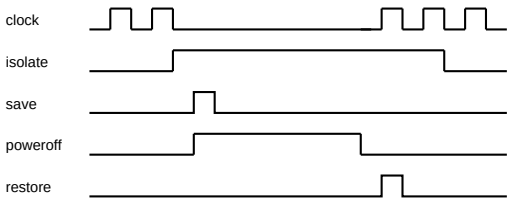


Figure 2. Control sequence for a design with state retention

## IV. Analysis of the break-even point

In order to evaluate if power gating brings energy benefits, the impact of power gating has to be analysed carefully. In this section, the savings and the overhead are analysed. Based on the findings, an analytical equation for the break-even point is derived.

In Figure 3, the power dissipation over time of a power-gated system is depicted. The total energy consumption is composed of the energy which is consumed by the power domain and the energy which is consumed by the additional power gating related modules. The modules which are constantly active, like a power manager, are consuming energy all the time. This energy ($E_{add.modules}$) is determined by the power consumption of the modules ($P_{add.modules}$) multiplied by the total run-time ($t_{total}$). The energy consumption of the remaining components depends on the state of the processors.

During $t_{active}$, the system is in its active state. The energy in this state is consumed by the power domain ($E_{mod,active}$), the isolation cells ($E_{iso,active}$) and the SR registers which consume more energy than normal registers ($\Delta E_{SR,active}$).

At $t_{idle}$, the system has finished the active state and goes immediately to the idle state by switching off the clock. At the same time it is powered off. The powering off process takes place until $t_{off}$. During that time, the power domain still leaks but the leakage energy ($E_{mod,leak,on}$) is converging to the off-level.

Then, the system remains switched off during $t_{down}$. The energy which is consumed depends on the leakage of the power switch(es) ($E_{switch,leak}$), the leakage of the isolation cells ($E_{iso,leak}$) and the leakage of the SR registers ($E_{SR,leak}$).

At $t_{sleep}$, the power domain is switched on again. The switching-on process takes until $t_{on}$. The energy which is consumed during that period is $E_{mod,leak,on}$ and the additional energy required to switch the power domain on ($E_{poweron}$). Afterwards, the system is fully functional.

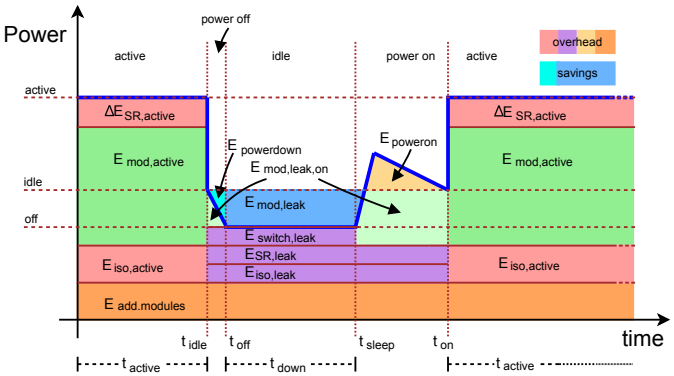


Figure 3. Power over time

The energy savings of a power gated module are determined by the leakage power which the power domain would consume if not switched off ($E_{mod,leak} = P_{mod,leak} \cdot t_{down}$ where $P_{mod,leak}$ is the leakage power of the power domain) and $E_{powerdown}$, which is the difference between the leakage energy the power domain would normally consume during powering off and the energy which is still consumed ($E_{mod,leak,on}$). The total energy savings are defined as follows:

$$E_{savings} = P_{mod,leak} \cdot t_{down} + E_{powerdown} \tag{2}$$

The energy overhead ($E_{overhead}$) is determined by the additional energy consumption during the different states, as explained above. Summarising, the energy overhead can be written as follows:

$$\begin{aligned} E_{overhead} = {} & t_{down} \cdot (P_{switch,leak} + P_{iso,leak} + P_{SR,leak}) \\ & + t_{active} \cdot (P_{iso,active} + \Delta P_{SR,active}) \\ & + t_{total} \cdot P_{add.modules} \\ & + E_{poweron} \end{aligned} \tag{3}$$

where $P_{switch,leak}$, $P_{iso,leak}$ and $P_{SR,leak}$ represent the leakage power consumption of the switches, the isolation and the SR registers. $P_{iso,active}$ is the power consumption of the isolation cells during active mode and $\Delta P_{SR,active}$ represents the additional power consumed by the SR registers compared to what normal registers would consume during active mode

For simplicity's sake, the individual factors are merged, which leads to the following equation:

$$E_{overhead} = t_{down} \cdot \beta + t_{active} \cdot \gamma + t_{total} \cdot \delta + \epsilon \tag{4}$$

with $\beta = P_{switch,leak} + P_{iso,leak} + P_{SR,leak}$, $\gamma = P_{iso,active} + \Delta P_{SR,active}$, $\delta = P_{add.modules}$ and $\epsilon = E_{poweron}$

Building on this analysis, a formula can be derived for the minimum percentage of time that the power domain has to be switched off in order to gain energy savings, i.e. the break-even point. Therefore, the energy savings must be bigger than the energy overhead:

$$E_{savings} > E_{overhead} \tag{5}$$

By using the above definitions for the savings and the overhead, and expressing $t_{active}$ with $t_{total} - t_{down}$, a condition for $t_{down}/t_{total}$ can be found. Also, some of the factors of the above analysis can be omitted because they are neglegible, namley $P_{switch,leak}$, $P_{iso,leak}$, $E_{poweron}$ and $E_{powerdown}$. This will be shown in Section VI where the results are presented. Summarising, the minimum percentage down time is:

$$\frac{t_{down}}{t_{total}} > \frac{\gamma + \delta}{\gamma + \alpha + \beta'} \tag{6}$$

where $\alpha = P_{leak,mod}$, $\beta' = P_{SR,leak}$, $\gamma = P_{iso,active} + \Delta P_{SR,active}$ and $\delta = P_{add.modules}$

## V. Implementation

In this section, the general methodology for implementing fine-grained power gating which has been developed for this work is explained. Also, the specific modifications which were applied to the processor in order to enable power gating are presented.

### A. General implementation flow

The power gating implementation flow is depicted in Figure 4. The first main step is the partitioning of the design into power domains. Therefore, the power consumption of all modules of interest has to be determined. In this work, it was done with PrimeTime from Synopsys [9] on a post-P&R netlist. In principle, also simulated values can be used but that would lead to less accuracy. Also, the utilisation of the modules during the application has to be determined. Using the obtained informations, the system can be partitioned into power domains by grouping modules with high leakage power consumption and similar utilisation profiles into power domains.

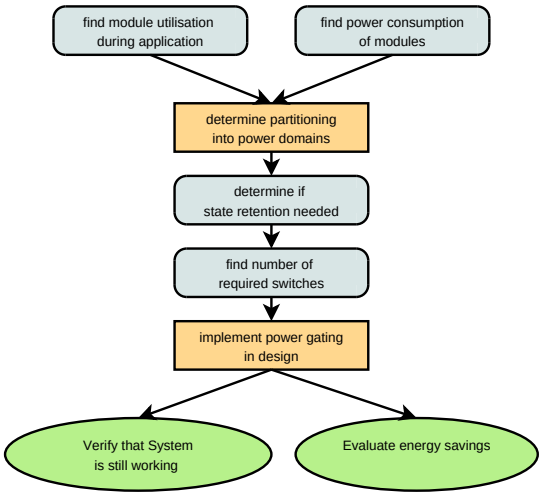


Figure 4. General power gating design flow

Next step is to determine if state retention is required. If a power domain includes registers which have to retain state, SR has to be applied, otherwise not.

Then, the number of required switches has to be identified. This is determined by the voltage drop over the switch and the timing constraints of the design [10].

The following step is the actual implementation of power gating in the design. That means, modifications like presented in Section III-B have to be applied to the design. This will be explained more detailed in the following section.

The final steps are verification that the system is still functionally correct and the analysis of possible energy savings. Verification can be done by gate-level simulations. The analysis of possible energy savings is based on Equation 6.

### *B. Modifications for the design presented in this work*

The design which is used in this work is an improved version of an ultra wide band (UWB) processor as presented in [11]. The processor from [11] consisted of four issue slots, two for scalar operations and two for vector operations. The improved version has merged one of the scalar slots and one of the vector slots, leading to a three issue slot processor with one issue slot for scalar operations only, one combined issue slot for both scalar and vector operations and one for vector operations exclusively. The scalar issue slots contain an address generator for load/ store operations and two ALUs which can work in parallel. The vector issue slots contain a vector adder, an address generator for vector load/store operations, a correlator, and several function units which were implemented to execute specific operation during different stages in the UWB operation. Furthermore, three vector registers are present. For this work, also a multiplier was added to the processor in order to make the processor suitable for a wider range of applications.

For this work, the following applications are used: first, the UWB receiver application as described in [11] is executed,

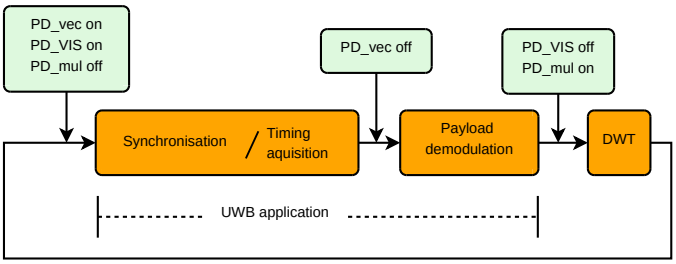


Figure 5. Proposed power-off scheme

consisting of a *Synchronisation / Timing Acquisition Phase* with a *Payload demodulation phase* following. Afterwards, a data decompression algorithm based on the discrete wavelet transform (DWT) algorithm is executed.

The analysis of the power consumption of the processor showed that the vector issue slots and the vector registers are a significant contributor to leakage power consumption. Within the vector issue slots, the main consumer of power are the vector adder and the correlator. In the scalar issue slots, the multiplier is significant.

The analysis of the applications revealed that the correlator has a very high utilisation, but the vector adder is only used during the *synchronisation phase*. The multiplier is not used in the UWB application, only in the DWT application. The vector issue slot and the vector registers are not used in the DWT application. These results lead to the partitioning into power domains as shown in Table I. Based on the previous analysis, a

Table I
POWER DOMAINS IN THE UWB PROCESSOR

| Power Domain | Modules | Gates | outputs |
|---|---|---|---|
| PD_vec | vector adder | 898 | 96 |
| PD_mul | multiplier | 1202 | 32 |
| PD_VIS | vector issue slots, vector registers | 13718 | 144 |

power-off scheme is proposed. *PD_mul* is switched off during the complete UWB application, *PD_VIS* is switched off during the DWT application. *PD_vec* is only switched on during the *synchronisation phase* in the UWB application, otherwise it is switched off. The resulting scheme is depicted in Figure 5. In this design, no state retention is required, thus no SR flip flops are used. For the actual power-shutoff, header switches are used. For isolation, all outputs are forced to zero.

To evaluate the difference between a hardware (HW) based and a software (SW) based power manager, both methods were implemented. The HW based power manager implements a simple state machine following Figure 2. For each power domain, one power manager is instanciated. The power managers are controlled by a control register which was added to the processor. It contains one bit per power domain, where a '1' indicates the power domain is shut off, otherwise it is on. To access single bits of the register, an additional function unit was implemented.

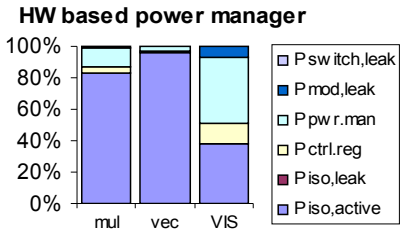


Figure 6. Results for the hardware based power manager

For the SW based power manager, the control signals to the power gating related cells are determined directly from the control register, i.e. the register has dedicated poweroff, isolation, and, when registers are included in the power domain, a reset bit for the power domains. Also, for this approach, the register is not implemented bitwise but can only be accessed completely. This was done so that clock gating can be applied to the register. Writing a value to the register can be performed by using the existing load/ store architecture.

The resulting design was synthesised and placed and routed with 90nm TSMC LP (low power) library for 100 MHz with the Cadence design tools [12]. The power domains were defined with the common power format (CPF) [13] during the design flow.

## VI. Results and discussion

In this section, the results for the impact of power gating are presented. For the analysis, power numbers from the post-P&R netlist were extracted using PrimeTime from Synopsys.

The section is organised as follows. First, the results for the hardware based power manager are presented. Then, the results for the software based approach are shown. In both cases, the power numbers for typical case ($1.2V, 25°C$) are displayed.

### A. The hardware based power manager

In Figure 6, the results for the power consumption distribution of the power gating related components (introduced as $\alpha$, $\beta$ $\gamma$ and $\delta$ in section IV) of the hardware based power manager implementation are depicted. The denotation corresponds to the definitions used in section IV. The terms $P_{pwr.man}$ and $P_{ctrl.reg}$ form $P_{add.modules}$ ($\delta$). They represent the power consumed by the power manager and the control register, respectively. The power numbers which are relevant for the determination of the break-even point, introduced as $\alpha$, $\beta$, $\gamma$ and $\delta$ in Equation 6 (without the factors for SR as in our power domains no SR is required) are presented in Table II. In the last row, the minimum $t_{down}/t_{total}$, calculated using Equation 6, is shown. It can be seen, that for the power domains *PD_mul* and *PD_vec* the overhead is dominated by the power consumed by the isolation cells during active mode ($P_{iso,active}$). The energy overhead for the power domain *PD_VIS* is caused by the isolation cells during active mode ($P_{iso,active}$) and the power manager ($P_{pwr.man}$) to approximately equal parts, a small part is due to the control register ($P_{ctrl.reg}$). Also it is

Table II
Minimum down time for the hardware based power manager

| | PD_mul | PD_vec | PD_VIS |
|---|---|---|---|
| $P_{iso,active}$ | 3.52E-5 | 1.84E-4 | 4.93E-6 |
| $P_{pg\_reg}$ | 1.61E-6 | 1.61E-6 | 1.61E-6 |
| $P_{pg\_ctrl}$ | 5.43E-6 | 5.43E-6 | 5.43E-6 |
| $P_{mod,leak}$ | 2.40E-7 | 7.88E-8 | 9.37E-7 |
| $t_{down}/t_{total}$ | 119 % | 104 % | 204 % |

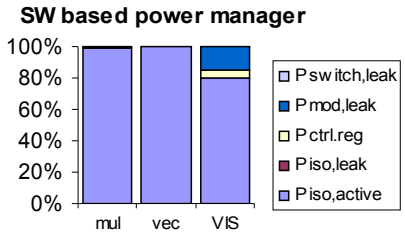


Figure 7. Results for the software based power manager

Table III
Relevant factors for the software based power manager

| | PD_mul | PD_vec | PD_VIS |
|---|---|---|---|
| $P_{iso,active}$ | 3.52E-5 | 1.84E-4 | 4.93E-6 |
| $P_{pg\_reg}$ | 2.72E-7 | 2.72E-7 | 2.72E-7 |
| $P_{mod,leak}$ | 2.4E-7 | 7.88E-8 | 9.37E-7 |
| $t_{down}/t_{total}$ | 100% | 100 % | 89 % |

quite noticable that *PD_VIS* is the only power domain with a leakage power consumption ($P_{mod,leak}$) which is significant enough to be visible in the graph. That also makes sense as it is much larger than the other power domains, see Table I. The required switch-off duty cycle, shown in the last row of Table II, is above 100 % for all power domains, which means that there can never be a benefit reached. The results demonstrate therefore clearly, that power gating would cause extra energy consumption in the system for the proposed power gating architecture.

### B. The software based power manager

As the power manager has shown to be a significant contributor to the overhead, the system was implemented using a software based power manager. The power breakdown of $\alpha$, $\beta$ $\gamma$ and $\delta$ is depicted in Figure 7. The results demonstrate that the overhead is mainly caused by the isolation cells during active mode ($P_{iso,active}$). For *PD_VIS*, additionally the power consumed by the control register ($P_{ctrl.reg}$) has a small influence. The total numbers for the most important factors for the trade-off are presented in Table III. In the last row, the minimum $t_{down}/t_{total}$ calculated with Equation 6 is given. For the power domains *PD_mul* and *PD_vec*, it is exactly 100 %. That means, that for those cases energy savings can never be obtained as the overhead will always be at least as big as the savings. However, for the power domain *PD_VIS*, benefits could be gained when it can be switched off for more than 89 % of the time. To analyse the distribution of

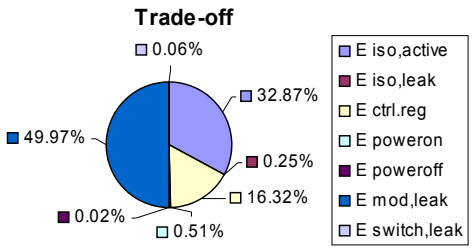


Figure 8. Energy distribution for PD_VIS for the software based power manager

the energy overhead and savings for the case that overhead and savings outweigh each other exactly, that means when *PD_VIS* is switched off for 89 % of the time, the contributors are depicted in Figure 8.

The graph shows that the savings (on the left side of the graph) are dominated by the leakage power consumed by the module when not under power gating control. The energy which is consumed during powering down (introduced as $E_{powerdown}$ in section IV) is negligible. Also, and in remarkable contradiction to prior published conclusions in the field, the energy to power up a module ($E_{poweron}$)) is negligible, also. Instead, the energy overhead is caused mainly by the energy consumption of the isolation cells during active mode ($E_{iso,active}$) and the additional power control modules ($E_{add.modules}$), in this case the power gating control register ($E_{ctrl.reg}$). The leakage of the switch ($E_{switch,leak}$) and the isolation cells ($E_{iso,leak}$) are also marginal. Incidentally, this is also proof that the assumptions that were made during derivation of Equation 6 are valid.

A surprising observation for both implementations is the large difference in power consumption of the isolation cells between the power domains. For *PD_vec* it is a factor of almost 40 compared to *PD_VIS*, which has 1.5 times so many output signals. This is caused the fact that the isolation block of *PD_vec* are in a critical path in the design, therefore extra buffers were required to meet timing constraints. Therefore, it is not only the isolation cells that consume additional power but also buffers.

Summarising, the obtained results show the following: The power domains need a low duty cycle, otherwise the energy overhead will exceed the savings. Also, the size of the power domain is of importance as it has a direct impact of the leakage power consumption which dictates the savings. Furthermore, the number of outputs is relevant as it determines the number of isolation cells.

## VII. Conclusions

In this paper, an implementation of fine-grained power gating on the datapath of a processor was presented. Furthermore, a detailed analysis of the break-even point for power gating was performed.

The results after the first implementation, which relied on a hardware-based power manager, demonstrate that the energy overhead of fine-grained power gating is significant and that it is mainly caused by additional modules, primarily by the isolation cells at the boundaries of a power domain and additional blocks like a dedicated power manager or a control register. Based on that result, power gating was also implemented with software based power management, thus omitting the need of a hardware based power manager. The results showed, that the energy overhead could be reduced significantly.

The results for both implementations demonstrated clearly that the overhead due to the energy required to switch a power domain on, which has been stated as main contributor to the overhead in literature, is neglegible. Instead, the overhead is caused by additional modules, primarly the isolation cells. As the analysed power domains had a typical utilisation profile and the isolation cells are a mandatory part of power gating, they can be considered as new power gating constraint. The obtained results also showed, that fine grained power gating applied on the datapath of a processor hardly can gain benefits, as the leakage energy which could be saved during idle is too low compared to the introduced energy overhead during active mode. It also must be said that leakage power consumed in the datapath is only a small fraction of the total power consumption in a processor, as the major part is consumed in the memories and I/O pads.

## References

[1] H. Jiang, M. Marek-Sadowska, and S. Nassif, “Benefits and costs of power-gating technique,” in *2005 IEEE International Conference on Computer Design: VLSI in Computers and Processors, 2005. ICCD 2005. Proceedings*, 2005, pp. 559–566.

[2] G. Panic, Z. Stamenkovic, and R. Kraemer, “Power gating in wireless sensor networks,” in *Wireless Pervasive Computing, 2008. ISWPC 2008. 3rd International Symposium on*, 2008, pp. 499–503.

[3] M. Keating, *Low Power Methodology Manual for System-on-chip Design*. Springer, 2007.

[4] J. Frenkil and S. Venkatraman, “Power Gating Design Automation,” *chapter in Closing the Power Gap between ASIC and Custom, Springer*, 2007.

[5] N. Pantazis and D. Vergados, “A survey on power control issues in wireless sensor networks,” *IEEE Communications Surveys & Tutorials*, vol. 9, no. 4, pp. 86–107, 2007.

[6] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose, “Microarchitectural techniques for power gating of execution units,” in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2004, pp. 32–37.

[7] K. Usami and N. Ohkubo, “A design approach for fine-grained run-time power gating using locally extracted sleep signals,” in *Computer Design, 2006. ICCD 2006. International Conference on*, 2007, pp. 155–161.

[8] A. Sathanur, A. Calimera, A. Pullini, L. Benini, A. Macii, E. Macii, and M. Poncino, “On quantifying the figures of merit of power-gating for leakage power minimization in nanometer CMOS circuits,” in *IEEE International Symposium on Circuits and Systems, 2008. ISCAS 2008*, 2008, pp. 2761–2764.

[9] http://www.synopsys.com/.

[10] J. Kao, S. Narendra, and A. Chandrakasan, “MTCMOS hierarchical sizing based on mutual exclusive discharge patterns,” in *Proceedings of the 35th annual conference on Design automation*. ACM New York, NY, USA, 1998, pp. 495–500.

[11] C. Bachmann, A. Genser, J. Hulzink, M. Berekovic, and C. Steger, “A Low-Power ASIP for IEEE 802.15.4a Ultra-Wideband,” in *Design, Automation and Test in Europe (DATE) 2009 Proceedings*, 2009.

[12] http://www.cadence.com/.

[13] http://www.si2.org/?page=811.