Self-Managing Power Management Unit

Dominik Macko, Katarína Jelemenská
Faculty of Informatics and Information Technologies
Slovak University of Technology
Bratislava, Slovakia
macko@fiit.stuba.sk, jelemenska@fiit.stuba.sk

***Abstract*—Power consumption is a very important aspect in almost every electronic system design. To minimize the power consumption, many advanced power-reduction techniques have been developed based on a power management. In modern systems the power management unit (PMU) is typically a complex circuit and therefore should also be targeted by power-efficient design techniques. This paper is focused on the design of self-managing PMU that can manage its own power and thus reduce the overall system power consumption. We show that the special power state machine design in the PMU allows to power inactive transition logic elements down during the idle time. We illustrate this design strategy on a simple example where approximately 70% leakage power reduction in transition logic was achieved.**

***Keywords*—low power; power control; power management; power reduction**

I. Introduction

The power consumption is one of the main aspects in modern digital system designs. Whether the reason is to maximize the battery life of the mobile devices or to minimize the system operating cost, the market is pushing the system designers to make power-efficient products.

In general, the power consists of two elements, a static and dynamic power. The first one represents a leakage and it depends on the supply voltage, the switching threshold voltage, and the size of transistor. The dynamic power depends on the switching activity, clock frequency, capacitance, supply voltage, and short-circuit current [1].

Over time, many power-reduction design techniques have been developed, such as clock-gating for a dynamic power reduction, or power-gating for a leakage power reduction (see Table I). Even some design techniques that are not originally created for power reduction might help to reduce it, e.g. advanced design for test techniques [2] or reduction of multiplexer trees [3]. However, many of the advanced techniques are difficult to adopt in the modern complex digital systems designs. To alleviate this difficulty, the low-power design and verification standard was developed (commonly known as UPF – Unified Power Format) [4]. This standard format provides unified power intent through typical design stages (e.g. synthesis, place and route) and ensures that the verification consistently interprets the functional semantics implied by the power intent [5]. The UPF standard simplifies the adoption of power-management strategy in the design process. It offers the constructs to specify power-management elements at the specification stage of the design process. The low-level power-management cells (e.g. power switches, isolation cells, level shifters) can be specified at the RTL (Register Transfer Level) as an addition to the main HDL (Hardware Description Language) functional model. The control signals for these power-management elements have to be generated by the functional model. Usually, a power management unit (PMU) is created that handles such power-related signals. Considering the modern designs, the PMU can contain more than a million of transistors [6]. With such a size, PMU represents the system area with the significant power dissipation.

To address this issue we concentrated our work on the development of a novel PMU design strategy that would enable the PMU to manage its own power consumption.

TABLE I. Power Reduction/Management Techniques

Technique	Description
Clock gating	Disables clock tree part not in use. Synchronous block stops its operation.
Operand isolation	Prevents switching of inactive datapath element.
Logic restructuring	Moves high switching logic to the front and low switching logic to the back.
Transistor resizing	Upsizing reduces dynamic power, downsizing reduces leakage power.
Pin swapping	Swaps the gate-pins in order the switching to occur at pins with lower capacitive loads.
Substrate biasing	Dynamically bias the substrate or the appropriate well in order to raise transistor voltage threshold in inactive mode, thereby reduce leakage.
Multiple supply voltages	Different blocks are operated at different (fixed) supply voltages. Signals that cross voltage domain boundaries have to be level-shifted.
Dynamic voltage scaling	Different blocks are operated at variable supply voltages. Uses look-up tables to adjust voltage on-the-fly to satisfy varying performance requirements.
Adaptive voltage scaling	Different blocks are operated at variable supply voltages. The block voltage is automatically adjusted on-the-fly based on performance requirements.
Frequency scaling	Frequency of the block is dynamically adjusted. Works alongside with voltage scaling.
Power gating	Turns off supply voltage to blocks not in use. Significantly reduces the leakage. Block outputs float and need to be isolated when connected to active block.
State retention power gating	Stores the system state prior to power-down. Avoids complete reset at power-up, which reduces delay and power consumption.
Memory segmentation	The memory is split into several portions. Not-used portions can be powered down [7].

This is an accepted version of the published paper:
D. Macko and K. Jelemenská, "Self-managing power management unit," 17th International Symposium on Design and Diagnostics of Electronic Circuits & Systems, Warsaw, 2014, pp. 159-162.
doi: 10.1109/DDECS.2014.6868781
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6868781&isnumber=6868744

II. Related work

The power management unit (PMU) is a block in the hardware system, which controls the power-management elements of other system blocks. It basically generates the signals controlling the power switches, the isolation and retention cells, and the level shifters in the precise time. Its purpose is to reduce the power dissipation due to an ineffective design, e.g. the unused blocks are turned off. Besides the digital control unit, the PMU usually contains also the integrated monitors and thus automatically determines the system performance requirements. In some cases the user power-saving preference is also monitored. This information is used to determine a next power state of the system (a state related to the power management).

The common design practice is to create a state machine for each system block operating with the power states (see Fig. 1). Based on the current state of the state machine, the specific values of control signals are generated. This approach was used for PMU design in [8]. The more effective solution is to join the system blocks into bigger parts, based on a simple rule – two blocks can be joined together when they are always in the same power state (Fig. 2). These "bigger" parts of the system are called power domains. In this design strategy, the power state machines are generated not for the design blocks, but for the power domains. It eliminates the unnecessary duplicates of the same state machine. Such an approach was used for example in [9, 10]. Another approach reducing the PMU power dissipation is described in [11]. It is focused on increasing the power-efficiency of the active-inactive state transitions.

Although the number of state machines is reduced, the unnecessary power dissipation inside the PMU still occurs. There are applications where the system remains a long period of time in one state before it transits to another (e.g. multimedia applications). In such applications, a leakage power inside the PMU digital control unit is dominant. To reduce power dissipation, one must look inside the FSM (Finite-State Machine) design.

III. Novel PMU design strategy

A state machine usually consists of the transition combinational logic, the state-representing sequential logic, and the output logic (Fig. 3a). Keeping the power management in mind, the states of FSM should directly correspond to the control signals for the specific power domain, i.e. the output logic is not present (Fig. 3b). In such a manner, the flip-flops saving the machine state can be continuously powered and thus be able to retain the controlling signals. The combinational logic controlling the state transition can be turned-off in inactive periods of time – when the power state does not need to be changed. Although the power in the combinational logic stage is turned off, the isolation between combinational and sequential parts is not needed because of integrated controlling mechanism of the flip-flops – the data input has to be active only at the active clock edge and the clock is stopped during the idle time. Even though the number of flip-flops representing the state is not optimal, this FSM design strategy has potential power savings.

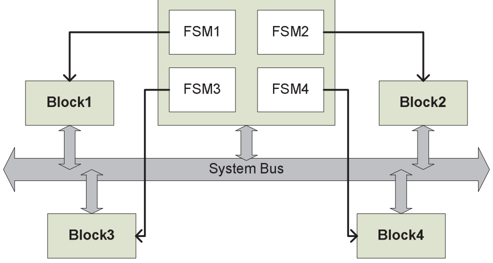

Fig. 1. Power management unit controlling the system blocks.

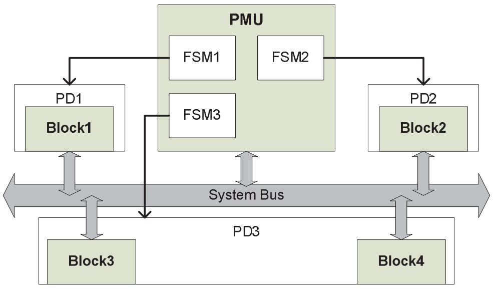

Fig. 2. Power management unit controlling the power domains.

To see the low-power benefits of previously mentioned strategy, we discuss the other FSM architecture (Fig. 3a) for the power-management purpose. This architecture allows us to optimize the number of state elements. Then the output logic represents the function of generating the controlling signals from the states. In such an FSM, in order the controlling signals to be always active we have two options. Firstly, we could let both the state and output logic powered. Basically it consumes more power than previous architecture because of the presence of output logic. It has sense when the output logic is simple, i.e. the power saved from reducing the number of state elements is greater than the power consumed by the output logic. The second option is to use the additional retention cells for the outputs to retain the values of control signals. This should allow us to turn the remaining parts of FSM off. But to preserve the current state, the state logic has to remain powered. The number of retention cells for the outputs is equal to the number of the states in the proposed FSM architecture, thus this option cannot be more power-efficient.

A. Sleep and wake-up handler

There are several possibilities how to put some system part to sleep. When we do not want the combinational part to be controlled by yet another state machine, we can simply generate the sleep signal by comparing the current and the new power state. When the power states are the same, the combinational part is turned off. When the states are different, the sleep signal is inactive, i.e. the system is woken up and the

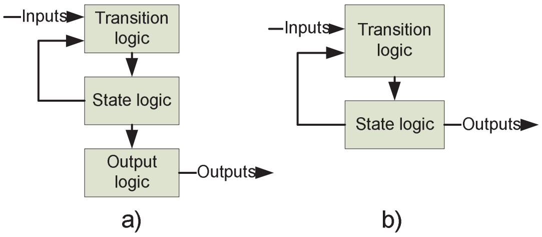

Fig. 3. Final state machine (Moore type) components – a) with the output logic; b) without the output logic.

transition to the next state is handled. Generated sleep signal is used also for gating the clock of the sequential state components – it guarantees that the state is not changing when the transition logic is turned off.

The proposed FSM architecture along with the sleep signal generation is shown in Fig. 4. The power-management elements of transition logic consist of the power switches. They allow the logic gates to be powered down. The power-management elements of sequential state logic contain only the clock-gating logic. It allows stopping the clock signal and thus the active edge of the state flip-flops does not come during the idle time.

The clock signal can be gated in several ways. For example, it can be simply realized by gating the clock signal using standard AND or NOR gate, or by using sophisticated latch-based clock gating cell. These implementations have different area requirements, different timing limitations, and even when the idea is the same, the eventual power consumption varies. Several clock gating techniques, along with their disadvantages, are discussed in [12]. When the designer is concerned about the glitches, the proper solution is to use latch-based clock gating cell. If the flip-flops with integrated clock gating enable signal are available, it is quite easy to connect them to the inverted sleep signal.

Depending on the realization there can be an overhead of a few clock cycles to wake up the system, thus the controlling signals of power management elements in the power domain are latish. When considering the long period of time (billions of cycles) between power state changes, such an overhead is not significant.

IV. Illustrative Example

For a demonstration we show an illustrative example using the proposed FSM design strategy for the creation of power management unit. Let's assume we have a power domain that needs to be managed for changing between the four long-term power states. These states include the *off* state, the *low-voltage* state, the *normal* state, and the *high-voltage* state. All transitions between these states are allowed, i.e. from each power state it is possible to pass to any of the other states. In Fig. 5, the state diagram is shown. These states are not all of the power states needed for power management. Let's assume that blocks in this power domain are communicating with the blocks from other power domains, i.e. the inputs and outputs of this power domain have to be isolated before power shut-down. Therefore a new state managing the isolation is added (Fig. 6). This state is internal and thus cannot come as wanted state at the FSM input. Each state represents the control signal values shown in Table II. Based on this table and allowed transitions in Fig. 6, we can build the transition table (Table III).

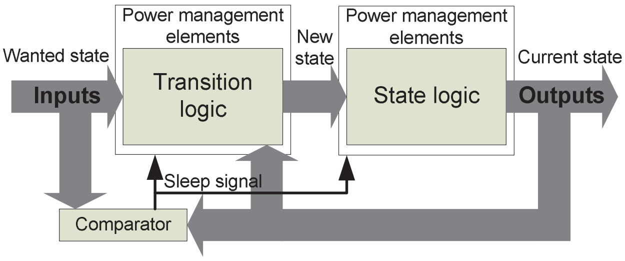

Fig. 4. The proposed FSM components.

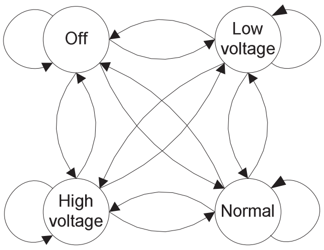

Fig. 5. State diagram of the given power domain.

TABLE II. Power State Control Signal Values

Power state	Control Signals
Off	SW=11, ISO=1
Isolation	SW=10, ISO=1
Low voltage	SW=10, ISO=0
Normal	SW=01, ISO=0
High voltage	SW=00, ISO=0

TABLE III. Transition table

Current state	Input				Next state			
Off (111)	111	100	010	000	111	101	101	101
Isolation (101)	111	100	010	000	111	100	010	000
Low voltage (100)	111	100	010	000	101	100	010	000
Normal (010)	111	100	010	000	101	100	010	000
High voltage (000)	111	100	010	000	101	100	010	000

Based on the transition table, it is possible to design the state machine. Functionality of this simple circuit was verified using the simulator Logisim v2.7.1 (Fig. 7). In this figure, the standard logic gates of the transition logic part were replaced by the custom made gates. These implement the same logic, but contain integrated power switches (the lowest input signal of the logic gates). The clock signal is gated using a low-level latch and the AND logic gate. The clock signal is enabled based on the comparator output. The inverted signal from the latch represents the sleep signal that activates the power switches in the transition logics.

The approximate power effectiveness can be evaluated based on the powered transistor count. For that purpose we need to choose the realization technology. For this simple evaluation we use the design of the logic gates from the NanGate 45nm Open Cell Library [13]. The designed transition logics consisted of 9 logic gates (including not-displayed inverter) – it resulted in 60 powered transistors. With the use of

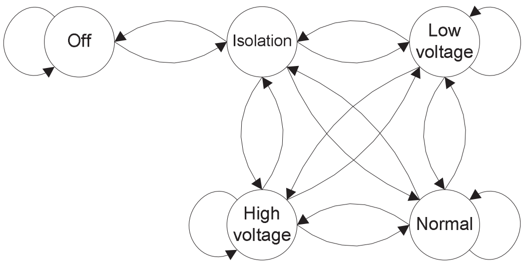

Fig. 6. Extended state diagram of the given power domain.

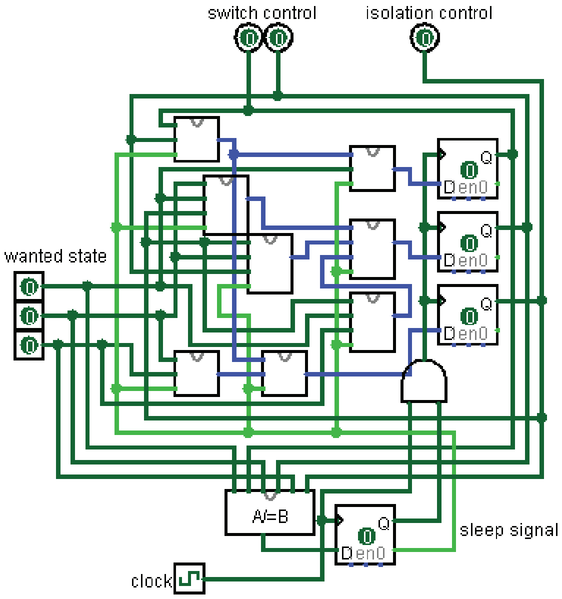

Fig. 7. Logisim design verification.

our method, we had to add the power switches. In the presented example we used a fine-grained power gating, i.e. we integrated the power switch into each logic gate. It resulted in additional 18 transistors. Although it represents a significant area overhead, during the idle period only these 18 transistors stay powered. Since all 78 transistors are powered during an active period, the leakage power consumption is increased by approximately 30%. But, considering billions of cycles between power state changes, the leakage power consumption in active time becomes negligible. Therefore we can assume approximately 70% leakage power reduction in transition logics representing approximately one third of the FSM size.

In some cases the coarse-grained power gating can replace the considered fine-grained power gating using only one power switch for the entire transition logics. In that case the area overhead would be only 3% and the reduction in the number of powered transistors during the idle time would be 97%. The pros and cons of both power gating approaches can be found in [14].

As a result, we designed the final state machine that manages the power states exactly as expected. This state machine manages the powering of its own transition logics. It has a benefit, that during the periods when the power state is not changing, the power to the transition logic is turned off. This means that the leakage power is not dissipating and therefore our design is more power-efficient than the standard FSM design.

V. CONCLUSIONS AND FURTHER WORK

Power management unit is used for generation of the control signals for the power management elements in the advanced power-reduction techniques (e.g. the clock gating, power gating, or the dynamic voltage scaling). In this paper the novel PMU design strategy has been presented. We proposed the self-managing PMU architecture allowing the management of its own power consumption. The effectiveness of this method has been illustrated on the example, where step-by-step design of a simple self-managing FSM has been described. This FSM controls only one power domain with four power states and can save approximately 70% transition logics leakage power in case a change in power state is not required. Considering the complex systems have several power domains with dozens of power states, it is no surprise that the modern PMUs have more than a million of transistors. In such a complex PMUs the static power dissipation is significant and has to be reduced. The proposed PMU design strategy alleviates this problem.

The further work will include the experiments showing the actual impact of the proposed PMU architecture on the leakage power consumption as well as on the dynamic power consumption. The PMU controlling several power domains will be considered.

ACKNOWLEDGMENT

This work was partially supported by the Slovak Science Grant Agency (VEGA 1/1008/12 “Optimization of low-power design of digital and mixed integrated systems“) and COST Action IC 1103 MEDIAN.

REFERENCES

[1] Cadence Design Systems, A practical guide to low power design: User experience with CPF, 2012. http://www.si2.org/?page=1061

[2] M. Siebert and E. Gramatová, “Delay Fault Coverage Increasing in Digital Circuits,” in DSD, IEEE, 2013, pp. 475-478.

[3] M. Maruniak and P. Pištek, “Binary decision diagram optimization method based on multiplexer reduction methods,” in ICSSE, IEEE, 2013, pp. 395-399.

[4] IEEE Standard for Design and Verification of Low Power Integrated Circuits, IEEE, 2013, (IEEE Std 1801-2013).

[5] S. Carver, A. Mathur, L. Sharma, P. Subbarao, S. Urish, and Q. Wang, “Low-power design using the Si2 common power format,” IEEE Design & Test of Computers, vol. 29, no. 2, pp. 62-70, 2012.

[6] M. G. Richard, “Intel's next CPU to include dedicated 'power control unit' to save power,” August 22, 2008, Online, December 2013. http://www.treehugger.com/gadgets/intels-next-cpu-to-include-dedicated-power-control-unit-to-save-power.html

[7] Š.Krištofík and E. Gramatová, “Redundancy Algorithm for Embedded Memories with Block-Based Architecture,” in DDECS, IEEE, 2013, pp. 272-274.

[8] D. Sun, S. Xu, W. Sun, S. Lu, and L. Shi, “Low power design for SoC with power management unit,” in ASICON, IEEE, 2011, pp. 719-722.

[9] T. Coulot, T. Souvignet, S. Trochut et al., “Fully integrated power management unit (PMU) using NMOS Low Dropout regulators,” in EUROCON, IEEE, 2013, pp. 1445-1452.

[10] H. Unterassinger, M. Dielacher, M. Flatscher et al., “A power management unit for ultra-low power wireless sensor networks,” in AFRICON, IEEE, 2011.

[11] M. Alioto, E. Consoli, and J.M. Rabaey, “’EChO’ Reconfigurable Power Management Unit for Energy Reduction in Sleep-Active Transitions,” IEEE Journal of Solid-State Circuits, vol. 48, no. 8, pp. 1921-1932, 2013.

[12] J. Kathuria, A. Khan, and A. Noor, “A Review of Clock Gating Techniques,” MIT International Journal of Electronics & Communication Engineering, vol. 1, no. 2, pp. 106-114, 2011.

[13] Nangate, “Open Cell Library,” Online, January 2014. http://www.nangate.com/?page_id=22

[14] R. Chadha and J. Bhasker, An ASIC Low Power Primer. Springer, 2013.