An Architectural Support for Reduction of In-rush Current in Systems with Instruction Controlled Power Gating

Sumanta Pyne
Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Odisha - 769008, INDIA
pynes@nitrkl.ac.in

ABSTRACT

The present work introduces a hardware based technique for reduction of in-rush current in processors with power gating (PG) facility. A PG instruction has been introduced which is responsible in turning on multiple components from sleep to active mode at overlapped time intervals. The supporting hardware for the proposed PG instruction allows overlapped wake-up as long as the resultant in-rush current is tolerable by the system. The efficacy of the proposed method is evaluated on MiBench and MediaBench benchmark programs. The proposed method reduces in-rush current by an average of 35% with average performance loss of 5%.

KEYWORDS

Leakage power; power gating; wakeup; in-rush current; architectural support

ACM Reference Format:

1 INTRODUCTION

The emergence of deep submicron process technology with the decrease in dimensions of the transistor has increased the transistor count and speed of operation at the cost of greater device leakage currents. Power gating (PG) is a technique used to reduce power consumption of VLSI chips, by shutting off the blocks that are not in use, thus reducing standby or leakage power. When a power gated block is switched on from sleep mode to active mode it draws a huge amount of in-rush current due to simultaneous charging of its internal capacitors. In-rush current is several times higher than the actual current required by the block to function in active mode. The flow of in-rush current may cause permanent damage to the circuit and also lead to higher power consumption. It can reduce the battery life for battery-operated systems due to rise in load current. The present work proposes a hardware support for PG instructions that simultaneously activates multiple components. It allows overlapped wake-up with guaranteed tolerable in-rush current. The existing works on management of in-rush current in PG systems are discussed in Sec. 2. The proposed method is explained in Sec. 3. Section 4 covers explanation of the experimental setup with analysis of the results. Finally, Sec. 5 concludes the present work with future directions.

2 RELATED WORK

This section discusses the existing research and development works on reduction of in-rush current in systems with PG. In [1] the authors proposed PG structures in which sleep transistors are turned on in a non-uniform stepwise manner to reduce the magnitude of peak current. Such a long daisy chain can cause long propagation delay and the slowly rising voltage can introduce other problems such as hot electron effects [2].

In [1] at wake-up the weak transistors are turned on first so as to slowly turn on the rush currents. When the design is discharged (charged) to a voltage close to zero ($V_{dd}$), the strong transistor pass is turned on ready for normal operation. A tool named CoolTime [2] guides the designer in setting power switch structure and sequence for controlling in-rush current and wake-up time. An in-rush current limiter circuit in [3] can sense the increased load current and produces sense current having a load current - sense current ratio of 1000:1, hence reducing the in-rush current. Kiong et al. introduced the in-rush current optimization power up flow analysis with PFET removal algorithm [5] to improve the in-rush current. In [6] the wake-up procedure the PG scheme implements a small transistor to control the sleep transistor in two stages to limit in-rush current and reduce wake-up time. In [7] the authors proposed model memory access power gating (MAPG), a low-overhead technique to enable power gating of an active core when it stalls during a long memory access. A novel framework for generating a proper power-up sequence of the switches to control the in-rush current of a power-gated domain has been introduced in [8]. The authors in [9] explain in-rush current reduction techniques like soft-start with the help of voltage regulators to increase rise time. In [10] Kim et al. discussed the reduction of in-rush current by turning on each switch cell at different times. They showed that in-rush current may cause permanent damage to the circuit and also lead to higher power consumption. It can reduce the battery life for battery-operated systems due to rise in load current. The present work proposes a hardware support for PG instructions that simultaneously activates multiple components. It allows overlapped wake-up with guaranteed tolerable in-rush current. The existing works on management of in-rush current in PG systems are discussed in Sec. 2. The proposed method is explained in Sec. 3. Section 4 covers explanation of the experimental setup with analysis of the results. Finally, Sec. 5 concludes the present work with future directions.
current can be reduced even more if signal transition time to switch each cell is adjusted.

At present there is a scope for architecture level in-rush current management in systems with instruction controlled PG.

3 PRESENT WORK

An arrangement for instruction controlled PG is shown in Fig. 1. It has n PG components $C_0, C_1, \cdots, C_{n-1}$. PG is done with the help of the header p-MOS transistors having higher threshold voltage $(V_T)$. The header switches are controlled by an n-bit power gating control register (PGCR) placed in the power gating controller (PGC).

The bits 0, 1, \cdots, n-1 are the PG bits of $C_0, C_1, \cdots, C_{n-1}$, respectively. If any of these bits $\alpha \in \{0, 1, \cdots, n-1\}$ is ‘0’, then the component $C_\alpha$ is in active mode, otherwise $C_\alpha$ is in sleep mode. Let there be two PG instructions switch_off and switch_on each consuming three clock cycles - one cycle in each of instruction fetch (IF), instruction decode (ID) and execution (EX) stages of the instruction pipeline. To put $C_\alpha$ in sleep (or power gated) mode the instruction switch_off($C_\alpha$) is used to set the value of $\alpha^{th}$ bit of PGCR. $C_\alpha$ in sleep mode can be put to active mode with the help of the instruction switch_on($C_\alpha$) which resets the value of $\alpha^{th}$ bit of PGCR. Hence, a program can use this PG facility.

![Figure 1: Instruction controlled PG system](image)

I\_\text{on}(mA) \hspace{1cm} I\_\text{on}(mA)

(a) Overlapped wake-up \hspace{1cm} (b) Non-overlapped wake-up

![Figure 2: In-rush current for different wake-up](image)

When $C_\alpha$ in sleep mode is switched on using switch_on($C_\alpha$) it draws in-rush current $I_\alpha \in \mathbb{Z}_{\geq 0}$ for a period of $w_\alpha$ cycles where $w_\alpha$ is the wake-up time $(T_\alpha)$ of $C_\alpha$. It is considered that $I_\alpha \leq I_{\text{tot}}$ for wake-up of any individual PG component $C_\alpha$, where $I_{\text{tot}}$ is the maximum tolerable in-rush current for a given system. The problem of intolerable in-rush current may arise during wake-up of multiple components during an overlapped time interval. Fig. 2a shows in-rush current ($I_t$) in milliampere (mA) for overlapped wake-up of two components $C_\alpha$ and $C_\beta$ where $\beta \in \{0, 1, \cdots, n-1\}$ and $\beta \neq \alpha$. $w_\beta$ and $I_\beta$ are the wake-up time and in-rush current of $C_\beta$, respectively. The resultant in-rush current is $I_\alpha + I_\beta$.

Simultaneous overlapped wake-up of several components can lead to higher flow of in-rush current resulting higher peak power dissipation and reduction of chip reliability. Hence, it is better to have non-overlapped wake-up as shown in Fig. 2b.

![Figure 3: Overlapped wake-up](image)

A program with switch_on instructions may cause overlapped wake-up of PG components. Considering an assembly language program consuming $N \in \mathbb{N}$ clock cycles. The instructions $I_1$ and $I_2$ are in EX stage at $i^{th}$ and $j^{th}$ cycles, respectively. Where $I_1$ is switch_on($C_\alpha$), $I_2$ is switch_on($C_\beta$) and $1 < i < j < N$. $I_j$ is executed after $\delta_{ij} = j - i$ cycles of $I_1$, $I_2$ and $I_j$ will result non-overlapped wake-up of $C_\alpha$ and $C_\beta$ if $i < j$ and $w_\alpha \leq \delta_{ij}$ as shown in Fig. 3a. Overlapped wake-up occurs if $w_\alpha > \delta_{ij}$ as shown in figures (3b), (3c) and (3d) where $1 < i < j < k < l \leq m < N$. The total in-rush current due to overlapped wake-up may exceed $I_{\text{tot}}$. To prevent this a hardware arrangement for switch_on is proposed that allows overlapped wake-up with guaranteed tolerable in-rush current.

3.1 In-rush aware switch_on instruction

Let switch_on(#components, component_list) be the format of switch_on instruction for the proposed approach, where #components $\in \{1, 2, \cdots, n\}$ is the number of PG components to be activated and component_list is the list of PG components where the $p^{th}$ element component_list[p] $\in \{0, 1, \cdots, n-1\}$ and $p \in \{0, 1, \cdots, \#\text{components} - 1\}$. The hardware support for the proposed switch_on instruction is shown in Fig. 3 where for each PG component $C_\alpha$ an in-rush current table ($I_{\text{tot}}$) is maintained. The tuple $t \in \{1, 2, \cdots, w_\alpha\}$ of $I_{\text{tot}}$ denoted by $I_{\text{tot}}[t]$ stores $I_\alpha$ where $I_\alpha$ is the value of $I_\alpha$ during $t^{th}$ cycle of wake-up of $C_\alpha$. $I_\alpha$ is minimum during cycles 1 and $w_\alpha$. $I_\alpha = I_{\text{pk}}$ during cycle $[\frac{w_\alpha}{2}]$ where $I_{\text{pk}}$ is the maximum or peak value of $I_\alpha$.

Let $b = \lfloor \log_{2} \max\{I_{\text{pk}}, I_{\text{pk}}, \cdots, I_{\text{pk}}\} \rfloor$ be the number of bits required to represent $I_{\text{tot}}[t]$. $I_{\text{tot}}$ is a table having $w_{\text{max}}$ tuples, where $w_{\text{max}} = \max\{w_0, w_1, \cdots, w_{n-1}\}$. $I_{\text{tot}}[t] \in \mathbb{Z}_{\geq 0}$ is the total in-rush current due to overlapped wake-up during cycle $i + t - 1$ where $i \in \{1, 2, \cdots, N\}$ and $t \in \{1, 2, \cdots, w_{\text{max}}\}$, $I_{\text{tot}}[t] \leq I_{\text{tot}}[\leq t]$; $t \in \{1, 2, \cdots, w_{\text{max}}\}$, $\lfloor \log_{2} I_{\text{tot}} \rfloor + 1$ bits are used to represent $I_{\text{tot}}[t]$. There are $w_{\text{max}} \cdot (\lfloor \log_{2} I_{\text{tot}} \rfloor + 1)$-bit adder-subtractors for performing $I_{\text{tot}}[t] \leftarrow I_{\text{tot}}[t] + I_{\text{tot}}[t]$ and $I_{\text{tot}}[t] \leftarrow I_{\text{tot}}[t] - I_{\text{tot}}[t]$ operations $\forall t \in \{1, 2, \cdots, w_{\text{max}}\}$ whenever their corresponding control lines ADD and SUB are high. An $O(w_{\text{max}} \times b)$
Figure 4: Hardware for in-rush aware switch_on

An Architectural Support for Reduction of In-rush Current in Systems with Instruction Controlled Power Gating

GLSVSI ’18, May 23–25, 2018, Chicago, IL, USA

4 EXPERIMENT AND RESULTS

To establish the efficacy of the proposed approach, simulations are carried out on gen5 [13] architecture simulator. McPAT [15] is used for obtaining power values. gen5 is configured with the instruction set and functional units (FUs) of the ARM Cortex-M4F processor [14]. The processor has seven FUs. Integer ALU (ialu) is not power gated because it is used in majority of the instructions. The bits 0, 1, 2, 3, 4 and 5 of PGCR are the PG bits of Floating Point Divider (fpdiv), Floating Point Multiplier (fpmul), Floating Point Adder (fpadd), Integer Divider (idiv), Integer Multiplier (imul), and Barrel Shifter (bshf), respectively as shown in Fig. 6. The size of the instruction cache is considered to be 32 KB. The architectural support for the in-rush current aware switch_on is incorporated within the simulation environment.

McPAT is configured with the power model of ARM Cortex-M4F based on 32nm process technology, where the leakage power dissipation is almost 70% of the total power consumption. Here, the processor clock frequency $f_{clk}$ is 1.0 GHz, and power supply voltage $V_{dd}$ is 0.9 V. $V_{pp}$ of processor’s n-MOS and p-MOS transistors are $V_{pp} = 0.18$ V and $V_{pp} = -0.18$ V, respectively.
switches are ~0.45 V. \( I_{\text{vol}} = 200 \) mA. Table 1 shows the values of load capacitance (\( C_{\text{vol}}^{(a)} \)), maximum operating current (\( I_{\alpha}^{(p)} \)), \( w_\alpha \) and peak \( I_{\alpha}^{(pk)} \) for each \( C_\alpha \) belonging to ARM Cortex-M4F with PG.

| \( C_{\text{vol}}^{(a)} \) (in nF) | \( I_{\alpha}^{(p)} \) (in mA) | \( I_{\alpha}^{(pk)} \) (in mA) | \( w_\alpha \) | \( \text{over} \) | \( \text{avg} \) | \| | | | | | | |
|---|---|---|---|---|---| |
| 6.58 | 5.9 | 3.89 | 2.24 | 1.81 | 0.8 | 

The features leading to generation of naive PG (using \ref{footref} \ref{footref2}) and in-rush current aware PG (IAPG) codes are also added to the GCC compiler for ARM Cortex-M4F.

The proposed technique is tested on MiBench [17] and MediaBench [13] benchmark programs as shown in Table 2 where \#cfow is the number of code fragments with overlapped wake-up. The benchmark programs are compiled using updated GCC compiler. The generated target code is executed on gem5 behaving as ARM Cortex-M4F processor. The performance values are generated by gem5 which is used by McPAT to produce values of peak, average, dynamic and leakage power. The value of in-rush current is obtained from the peak power values. The experimental results are shown in Fig. 7. PG and IAPG are compared with respect to the normalized values of power and delay. Leakage power savings achieved by IAPG is similar to that of PG. Peak power dissipation \( \propto \) in-rush current \( \propto \) number of overlapped wake-up (#overlapped_wake_up). For IAPG #overlapped_wake_up is lesser than that of PG. Hence, peak power and in-rush current for IAPG are lesser than PG. Reduction of in-rush current and peak power dissipation experienced by IAPG lie within 16-47\% and 18-45\%, respectively. This enables IAPG to achieve 4-14\% savings in total average power consumption. Delay \( \propto \) number of switch_on instructions \( \propto \#\text{cfow} \). The proposed switch_on consumes \( \Omega \) (\#components) cycles whereas the naive switch_on takes \( \Omega(1) \) cycles. The loss in performance experienced by IAPG lies within 0.7-10\%.

5 CONCLUSION

The present work introduces a hardware based technique for reduction of in-rush current in PG systems. The proposed method IAPG reduces in-rush current by allowing overlapped wake-up within the limitations of tolerable in-rush current. IAPG is evaluated on standard benchmark programs. IAPG reduces considerable amount of in-rush at the cost of increase in delay, design space and design cost. The future work will investigate to address these issues. Thus making them fit for real-time and safety-critical embedded systems.

REFERENCES