Simultaneous Communication and Processor Voltage Scaling for Dynamic and Leakage Energy Reduction in Time-Constrained Systems

Alexandru Andrei[*], Marcus Schmitz[‡], Petru Eles[†], Zebo Peng[†]
Dept. of Computer and Information Science
Linköping University S-58183 Linköping, Sweden
{alean,g-marsc,petel,zebpe}@ida.liu.se

Bashir M. Al Hashimi[‡]
Dept. of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ, United Kingdom
bmah@ecs.soton.ac.uk

Abstract

In this paper, we propose a new technique for the combined voltage scaling of processors and communication links, taking into account dynamic as well as leakage power consumption. The voltage scaling technique achieves energy efficiency by simultaneously scaling the supply and body bias voltages in the case of processors and buses with repeaters, while energy efficiency on fat wires is achieved through dynamic voltage swing scaling. We also introduce a set of accurate communication models for the energy estimation of voltage scalable embedded systems. In particular, we demonstrate that voltage scaling of bus repeaters and dynamic adaption of the voltage swing on fat wires can significantly influence the system's energy consumption. Experimental results, conducted on numerous generated benchmarks and a real-life example, demonstrate that substantial energy savings can be achieved with the proposed techniques.

1 Introduction

To fully exploit the potential performance provided by multiprocessor architectures (e.g. systems-on-a-chip), communication has to take place over high performance buses, which interconnect the individual components, in order to prevent performance degradation through unnecessary contention. Such global buses require a substantial portion of energy, on top of the energy dissipated by the computational components [18, 19]. The minimization of the overall energy consumption requires the combined optimization of both the energy dissipated by the computational processors as well as the energy consumed by the interconnection infrastructure.

Dynamic voltage scaling (DVS) and adaptive body biasing (ABB) are two system-level approaches that can be used to reduce the energy consumed by processors. Both techniques provide an energy/performance trade-off, which can be exploited during run-time. Conceptually, DVS aims to reduce the *dynamic power* consumption by decreasing the operational frequency and supply voltage [8, 23], while ABB is efficient in limiting the *static leakage power* consumption by reducing the operational frequency and increasing the circuit's threshold voltage via body biasing [12]. Until recently, dynamic power has been the main source of power dissipation. However, in deep-submicron CMOS technology (feature size $< 70nm$), leakage power is predicted to become comparable to the dynamic power [4, 12]. Hence, the combination of DVS and ABB will become essential to manage the total energy dissipation of next generation embedded systems [14]. Approaches for combined DVS and ABB in distributed time-constrained systems have been reported [22, 15].

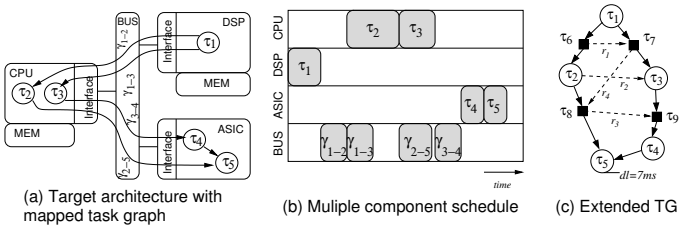
A negative side-effect of the shrinking feature sizes is the increasing *RC* delay of on-chip wiring [10, 19]. The main reason behind this trend is the ever-increasing line resistance. In order to maintain high performance it becomes necessary to "speed-up" the interconnects. Two implementation styles which can be applied to reduce the propagation delay are: (a) The insertion of *repeaters* and (b) the usage of *fat wires*. In principle, repeaters split long wires into shorter (faster) segments [10, 11, 19] and fat wires reduce the wire resistance [18, 19]. Techniques for the determination of the optimal quantity of repeaters are introduced in [10, 11]. An approach to calculate the optimal voltage swing on fat wires has been proposed in [18]. Similar to processors with supply voltage scaling capability, approaches for link voltage scaling were recently presented in [17, 21]. An approach for communication speed selection was outlined in [13]. Another possibility to reduce communication energy is the usage of bus encoding techniques [3]. In [9], it was demonstrated that shared-bus splitting, which dynamically breaks down long, global buses into smaller, local segments, also helps to improve energy savings. An estimation framework for communication switching activity was introduced in [7].

Until now, energy estimation for system-level optimization was treated in a largely simplified manner, with the aim to hide most "irrelevant" information [13, 20]. Communication delay and energy estimations are often based on naive models which ignore essential aspects such as bus implementation technique (repeaters, fat wires), leakage power, and voltage swing adaption. This, however, very often leads to oversimplifications which affect the correctness and relevance of the proposed approaches and, consequently, the accuracy of results. On the other hand, issues like optimal voltage swing and increased leakage power due to repeaters are not considered at all for implementations of voltage-scalable embedded systems.

The aim of this work is to introduce a voltage scaling technique, considering scaling of both communication links and processing elements. The approach is based on suitable delay and energy models, in particular for communication links that are implemented via repeaters or fat wires. As opposed to previous system-level approaches, we take also into account the communication leakage power consumption as well as the dynamic adaption of the voltage swing. The presented work makes the following contributions:

(a) We solve the combined voltage scaling problem for processing elements and communications links. We take into consideration transition overheads in terms of energy as well as time, and also

[*]Supported by CUGS swedish graduate school in computer science
[†]Supported in part by SSF through the STRINGENT excellence center
[‡]Supported in part by EPSRC under grant GR/S95770



(a) Target architecture with mapped task graph (b) Muliple component schedule (c) Extended TG

Figure 1. Architecture and application model

account for supply voltage and body bias scaling, in order to reduce simultaneously dynamic and leakage energy dissipation. The problem is formulated as a nonlinear programming (NLP) problem which solves the voltage scaling optimally in polynomial time for components that can operate over a continuous range of voltages.

(b) Since voltage scaling for components that operate over a set of discrete voltages was proofed to be NP-hard [15], we introduce a simple yet effective heuristic based on the NLP formulation for the continuous voltage scaling problem.

(c) To allow an effective voltage scaling on the communication links, we outline a set of delay and energy models. Further, we take into account the possibility of dynamic voltage swing scaling on fat wires and address the leakage power dissipation in bus repeaters.

The remainder of the paper is organized as follows. Section 2 introduces the used architecture and application models. An illustrative example, outlining the motivation behind the presented work, is given in Section 3. The problem formulation is presented in Section 4. Power and delay models for processors as well as communication links are given in Section 5. Section 6 presents the NLP formulation for continuous voltage scaling and the heuristic used for discrete voltage scaling. Experimental results are given in Section 7. Conclusions are drawn in Section 8.

2 Architecture and Application Model

In this paper, we consider embedded systems which are realized as multiprocessor systems-on-a-chip (SoC). Such architectures consist of several different processing elements (PEs), some of which feature DVS and ABB capability. These computational components communicate via an infrastructure of communication links (CLs), like buses and point-to-point connections. Similar to the PEs, the CLs might be equipped with voltage scaling capability. An example architecture is shown in Fig. 1(a).

The functionality of data-flow intensive applications, such as voice processing and multimedia, can be captured by task graphs $G(\mathcal{T},\mathcal{C})$. Nodes $\tau \in \mathcal{T}$ in these directed acyclic graphs represent computational tasks, while edges $\gamma \in \mathcal{C}$ indicate data dependencies between these tasks (i.e. communications). Tasks require a certain number of clock cycles *NC* to be executed, depending on the PE to which they are mapped. Similarly, if two dependent tasks are assigned to different PEs, π_x and π_y with $x \neq y$, then the communication takes place over a CL, involving a certain number of clock cycles for the data transfer. Further, tasks are annotated with deadlines *dl* that have to be met during run-time.

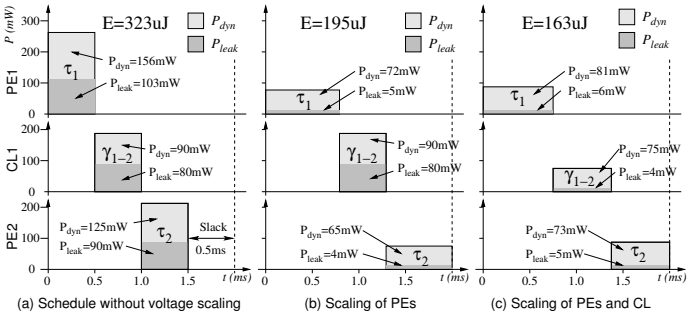
We assume that the task graph is mapped and scheduled onto the target architecture, i.e., it is known where and in which order tasks and communications take place. Fig. 1(a) shows an example task graph that has been mapped onto an architecture, and Fig. 1(b) depicts a possible execution order. To tie the execution order into the specification, we perform the following transformation on the original task graph. Firstly, all communications that take place over communication links are captured by communication tasks, as indicated by squares in Fig. 1(c). For instance, communication γ_{1-2} is replaced by task τ_6 and the edges connecting τ_6 to τ_1 and τ_2 are introduced. $\mathcal{K}$ defines the set of all such communication tasks. Secondly, on top of the precedence relations given by data dependencies between processing or communication tasks, we introduce additional precedence relations $r \in \mathcal{R}$, generated as result of scheduling tasks mapped to the same PE and communications mapped on the same CL. In Fig. 1(c) the dependencies $\mathcal{R}$ are represented as dotted edges. After these transformations we obtain an extended task graph $G_E(\mathcal{V},\mathcal{E})$. We define the sets of all nodes and all edges in the extended graph as $\mathcal{V}$ and $\mathcal{E}$, respectively. Further, we define the set $\mathcal{E}^\bullet \subseteq \mathcal{E}$ of edges, as follows: an edge $(i,j) \in \mathcal{E}^\bullet$ if it connects node τ_i with its immediate successor τ_j (according to the schedule), where τ_i and τ_j are mapped on the same component. For example, in Fig. 1(c), $\mathcal{E}^\bullet = \{(2,3),(4,5),(6,7),(7,8),(8,9)\}$.

3 Motivational Example

In order to motivate the principles behind the proposed techniques, we first illustrate the influence that the combined supply voltage and body bias scaling of bus repeaters has on the system's energy efficiency (Section 3.1). Secondly, we show how dynamic voltage swing scaling can significantly reduce the energy consumption of a fat wire-based bus (Section 3.2).

3.1 Voltage Scaling on Repeater-Based Buses

Consider an architecture consisting of two voltage-scalable processing elements (PE1 and PE2) that communicate via a repeater-based, shared bus (CL1), which also allows voltage scaling. PE1 has to execute task τ_1 and PE2 runs task τ_2. Task τ_2 can only start after receiving data from task τ_1, and it has to finish execution before a deadline of 2*ms*. Fig. 2(a) shows the initial schedule for this system, considering an execution at the nominal voltage settings (highest supply voltage and body bias voltage), i.e., all components run at their highest performance. The diagram



(a) Schedule without voltage scaling (b) Scaling of PEs (c) Scaling of PEs and CL

Figure 2. Voltage scaling on a repeater-based bus

shows the power dissipation (dynamic and leakage) of the individual components over time. For clarity reasons we assume in this example that the processors as well as the repeaters of the bus have the same nominal voltage values ($V_{dd} = 1.8V$ and $V_{bs} = 0V$). Further, we assume that the supply voltages and the body bias voltages of all components can be varied continuously in the ranges $[0.6,1.8]V$ and $[-1,0]V$, respectively. Given the power consumptions at the nominal voltages, we can compute a total energy consumption of the tasks and communication in the initial schedule as $(156+103)mW \cdot 0.5ms + (90+80)mW \cdot 0.5ms + (125+90)mW \cdot 0.5ms = 323\mu J$. As can be observed, at the nominal voltages the system over-performs, leading to a slack of $0.5ms$.

In order to reduce the energy consumption, we can exploit this slack by scaling the voltages of the processing elements. Using the technique described in [15], it is possible to calculate the optimal voltage

settings. The resulting voltages for the execution of tasks τ_1 and τ_2 are $(1.43V, -0.42V)$ and $(1.54V, -0.49V)$, respectively. The corresponding, voltage scaled schedule is shown in Fig. 2(b). The dynamic and leakage power consumptions of the tasks are reduced to $(72mW, 5mW)$ and $(65mW, 4mW)$; however, the execution times have increased to $0.79ms$ and $0.71ms$. Executing the tasks with these settings, the system dissipates an energy of $195\mu J$, a reduction by 39% compared to the energy at nominal voltages.

To demonstrate the importance of combined voltage scaling of the processors and the repeater-based bus, we have produced the voltage scaled schedule shown in Fig. 2(c). Here the supply voltage as well as the body bias voltage of the repeaters have been adjusted with the aim to reduce the dynamic and leakage power dissipation of the bus. The optimal voltage settings for the tasks and the communication can be calculated as $(1.48V, -0.42V)$ for PE1, $(1.77V, -0.61V)$ for PE2, and $(1.59V, -0.50V)$ for the bus repeaters. Correspondingly the power dissipations are given by $(81mW, 5mW)$, $(75mW, 4mW)$, and $(73mW, 5mW)$, thereby, reducing the overall system energy dissipation to $163\mu J$. Compared to the nominal energy consumption a reduction by 49%, which is 10% better than in the case when only the PEs are voltage scaled. □

3.2 Voltage Swing Scaling on Fat Wire-Based Buses

In this example, we demonstrate the influence that a dynamic variation of the voltage swing (the voltage on the wire) has on the energy efficiency of the bus. Fig. 3 shows the total power consumption of a fat wire bus (including drivers and receivers), depending on the voltage swing at which data is sent. These plots have been generated via SPICE simulations using the Berkeley predictive $70nm$ CMOS technology library. The two plots show the total power consumption on the bus for two different voltage settings of the bus drivers and receivers. For example, if the driver connected to PE1 and the receiver at PE2 operate at $1.0V$, the lowest bus power dissipation ($0.55mW$) is achieved by a voltage swing of $0.14V$. Let us assume that the voltages of the driver

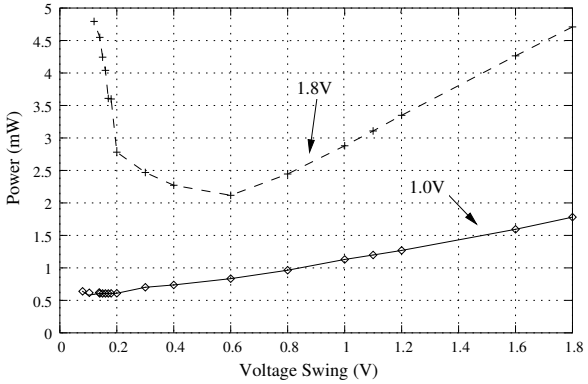


Figure 3. Optimum swing on a fat wire bus

and receiver are changed during run-time to $1.8V$ due to voltage scaling. The bus power/voltage swing relation for this situation is indicated by the dashed line. As we can observe, by keeping the voltage swing at $0.14V$, the power dissipation on the bus will be $4.5mW$. However, inspecting the plot reveals that it is possible to reduce the bus power dissipation by changing the voltage swing from $0.14V$ to $0.6V$. At this voltage swing, the bus dissipates a power of $2.2mW$, i.e., a 51% reduction can be achieved by changing the voltage swing.

Now assume that the driver and receiver voltages are changed back from $1.8V$ to $1.0V$. Keeping the swing at $0.6V$ results in a power of $0.83mW$, which is, compared to the optimal $0.55mW$ at $0.14V$, 33% higher than necessary. □

The examples above have demonstrated that the combined supply voltage and body bias scaling as well as the dynamic voltage swing scaling are effective techniques to reduce power consumption. To exploit their power reduction potential, we have to introduce suitable energy and delay models, which can be used during system-level optimization. These models will be presented in Section 5.

4 Problem Formulation

We assume that all tasks and communications of the extended task graph have been mapped and scheduled onto the target architecture. For each task τ_i its deadline dl_i, its number of clock cycles to be executed NC_i, and the switched capacitance C_{eff_i} are given. Each processor can vary its supply voltage V_{dd} *and* body bias voltage V_{bs} within certain continuous ranges (for the continuous voltage scaling problem), or within a set of discrete voltages pairs $m_z = \{(V_{dd_z}, V_{bs_z})\}$ (for the discrete voltage scaling problem). The power dissipations (leakage, dynamic) and the cycle time (processor speed) depend on the selected voltage pair. Such a voltage pair is also referred to as *performance mode*. We consider that a transition between two different performance modes on a processor requires an overhead in terms of energy and time. Tasks are executed cycle by cycle, and each cycle can potentially execute at a different voltage pair, i.e., at a different performance mode.

For each communication task τ_{ij}, which captures a communication between processing task τ_i and τ_j, the number of bytes b_{ij} is given. Depending on the employed bus implementation style, either using repeaters or fat wires, we have to distinguish between two subproblems:

Repeater Implementation: The communication speed as well as the communication power on bus architectures implemented through repeaters depend on the supply voltage and body bias voltage. Similar to processing elements, these voltages can be varied within a continuous range, or within a set of discrete voltage pairs $m_z = \{(V_{dd_z}, V_{bs_z})\}$, and transitions between different bus performance modes require an energy and time overhead. Furthermore, an energy overhead is required to adapt the bus voltage to the processor voltage. □

Fat Wire Implementation: If communication is performed over fat wires, it is necessary to dynamically adapt the voltage swing at which data is transfered. Furthermore, in order to reduce the power dissipated in the bus drivers and receivers, it is possible to dynamically scale the supply and body bias voltage of these components. While the voltage swing can be scaled without an influence on the bus performance, the operational speed of the bus drivers and receivers is affected through voltage scaling, i.e., the bus performance has to be adjusted in accordance to the driver/receiver speed. In the case of continuous voltage scaling, the value for the voltage swing, the supply voltage, and the body bias voltage can be changed within a continuous range. On the other hand, for the discrete voltage scaling case, the components operate across sets of discrete voltages, referred to as modes. For the voltage swing this set is $n_z = \{V_{sw_z}\}$ and for the bus drivers and receiver the set is $m_z = \{(V_{dd_z}, V_{bs_z})\}$. Of course, changing the voltage swing value as well as the supply and body bias voltages requires an energy and time overhead. □

Our overall goal is to find mode assignments for each processing and communication tasks such that the individual task deadlines are satisfied and the total energy consumption, including overheads, is minimal.

5 Power and Delay Models

We consider an interconnect structure in which PEs as well as CLs can operate at different frequencies and voltages. In Fig. 4(a) we can observe three voltage/frequency islands which can, independently of each other, run at their own voltages and frequencies. The first and second

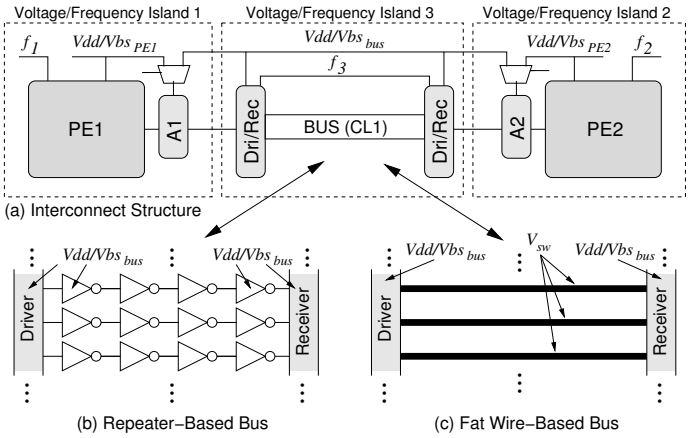


Figure 4. Interconnect structures

island correspond to processing elements, while the third represents the communication link. In addition to the processing cores, the PEs' voltage/frequency islands contain the communication adapters A1 and A2. The main function of these adapters is the translation between the logic state values used by the PEs and the bus. For instance, at a certain moment the logic value 1 might be represented by a voltage of $1.2V$ in the PE, while on the bus the same logic value corresponds to the signal voltage $0.5V$. The adapters are implemented as registers that store one bit for each bus line and they operate in a multiplexed fashion, either at the processor voltage or the bus voltage. As shown in Fig. 4(a), the CLs' voltage/frequency island contains the bus itself, as well as drive and receiver stages, which are responsible for driving the bus load and restoring the originally sent signals, respectively.

In the remainder of this section, we introduce the models that are used to capture the power dissipation and delay of processing elements as well as communication links.

5.1 Processor Models

Digital CMOS circuitry has two major sources of power dissipation: (a) dynamic power P_{dyn} which is dissipated whenever active computations are carried out (switching of logic states), and (b) leakage power P_{leak} which is consumed whenever the circuit is powered, even if no computations are performed. The dynamic power is expressed by [6, 14],

$$P_{dyn} = s_\tau \cdot C_{eff} \cdot f \cdot V_{dd}^2 \tag{1}$$

where s_τ, C_{eff}, f, and V_{dd} denote the switching activity caused by task τ, the charged capacitance, operational frequency, and circuit supply voltage, respectively. The leakage power is given by [14],

$$P_{leak} = L_g \cdot V_{dd} \cdot K_3 \cdot e^{K_4 \cdot V_{dd}} \cdot e^{K_5 \cdot V_{bs}} + |V_{bs}| \cdot I_{Ju} \tag{2}$$

where V_{bs} is the body bias voltage and I_{Ju} represents the body junction leakage current. The fitting parameters K_3, K_4 and K_5 denote circuit technology dependent constants and L_g reflects the number of gates. For clarity reasons we maintain the same indices as used in [14], where also actual values for these constants are given.

Scaling the supply and the body bias voltage, in order to reduce the power consumption, has a side-effect on the circuit delay d and hence the operational frequency [6, 14],

$$f = 1/d = \frac{((1+K_1) \cdot V_{dd} + K_2 \cdot V_{bs} - V_{th1})^\alpha}{K_6 \cdot L_d \cdot V_{dd}} \tag{3}$$

where α reflects the velocity saturation imposed by the used technology (common values $1.4 \leq \alpha \leq 2$), L_d is the logic depth, and K_1, K_2, K_6 and V_{th1} are circuit dependent constants. The execution time of a task $\tau \in \mathcal{T}$ is expressed by $t = NC_\tau / f$, where NC_τ is the number of clock cycles needed to execute the task.

Each time the processor's supply voltage and body bias voltage are altered, the change requires a certain amount of extra energy and time. These energy $\varepsilon_{k,j}$ and delay $\delta_{k,j}$ overheads, when switching from V_{dd_k} to V_{dd_j} and from V_{bs_k} to V_{bs_j}, are given by [14],

$$\varepsilon_{k,j} = C_r \cdot |V_{dd_k} - V_{dd_j}|^2 + C_s \cdot |V_{bs_k} - V_{bs_j}|^2 \tag{4}$$

$$\delta_{k,j} = \max(p_{Vdd} \cdot |V_{dd_k} - V_{dd_j}|, p_{Vbs} \cdot |V_{bs_k} - V_{bs_j}|) \tag{5}$$

where C_r denotes the power rail capacitance, and C_s the total substrate and well capacitance. The time/voltage slopes for adjusting V_{dd} or V_{bs} are captured by the constants p_{Vdd} and p_{Vbs}. Considering that supply and body bias voltage can be scaled in parallel, the transition overhead $\delta_{k,j}$ depends on the maximum time required to reach the new voltages.

5.2 Communication Models

We consider a bus-based communication system as in Fig. 4. Whenever the processing element PE_1 sends data to PE_2 over the bus, V_{dd_1} is converted to the bus voltage V_{dd_3} by the bus adapter of PE_1. At the destination processing element PE_2, V_{dd_3} is converted to V_{dd_2}. Each voltage conversion in the bus adapter requires an energy overhead, which is:

$$E_{adapter} = C_{adapter} \cdot (V_{dd_{PE}} - V_{dd_{bus}})^2 \tag{6}$$

Thus, the total energy consumed when communicating between two processors PE_1 and PE_2 over the bus is given by:

$$E_{comm} = E_{adapter_1} + E_{bus} + E_{adapter_2} \tag{7}$$

Feature size scaling in deep-submicron circuits is responsible for an ever-increasing wire delay of the global interconnects. This is mainly due to higher wire resistances R caused by a shrinking cross-sectional area A of the wire ($R \propto 1/A$). Two approaches to cope with this problem have been proposed: (a) the usage of repeaters [10, 11] and (b) the usage of fat wires [18, 19]. The bus energy E_{bus} of Eq. (7) depends on which of these two approaches is used. The following subsections outline the models to estimate these energy dissipations.

5.2.1 Repeater-based Bus

The wire delay depends quadratically on the wire length, which can be approximated using an *RC* model. In order to reduce the quadratic dependency (i.e., reducing the wire delay), it is possible to break the wire into smaller segments by inserting repeaters. The authors in [19] estimate an increasing number of repeaters with technology scaling down. For instance, up to 138 repeaters are used in $50nm$ technology for a corner-to-corner wire with a die size of $750mm^2$. Technically, repeaters are implemented as simple CMOS inverter circuits, as shown in Fig. 4(b). In accordance, the power dissipated by a bus implemented with repeaters is given by,

$$P_{rep} = N \cdot (\underbrace{s_\tau \cdot C_{rep} \cdot V_{dd}^2 \cdot f}_{P_{dyn}} + \underbrace{V_{dd} \cdot K_3 \cdot e^{K_4 \cdot V_{dd}} \cdot e^{K_5 \cdot V_{bs}} + |V_{bs}| \cdot I_{Ju}}_{P_{leak}}) \tag{8}$$

where N is the number of repeaters, s_τ is the average switching activity caused by communication $\tau \in \mathcal{K}$, C_{rep} is the load capacity of a repeater (the sum of the output capacity of a repeater C_d, the wire capacity C_w, and the input capacity of the next repeater C_g), and V_{dd}, V_{bs}, and f are the supply voltage, body bias voltage, and the frequency at which the repeaters operate. Further, the constants K_3, K_4, K_5, and I_{Ju} depend on repeater circuits.

The bus speed is constrained by the repeater frequency. Since the repeaters are implemented as CMOS inverters, we can use Eq. (3) to approximate the operational frequency f of the bus. Correspondingly, the execution time of a communication $\tau \in \mathcal{K}$ is given by,

$$t = \left\lceil \frac{NB_\tau}{W_{bus}} \right\rceil \cdot \frac{1}{f} \tag{9}$$

where NB_τ denotes the number of bits to be transmitted by communication τ and W_{bus} is the width of the bus (i.e. the number of bits transmitted with each clock cycle). Accordingly to Eq. (8) and (9), the bus energy dissipation is given by $E_{bus} = P_{rep} \cdot t$. Of course, scaling the supply voltage and body bias voltage of the repeaters requires also an overhead in terms of energy and time, which are given in the same way as the overheads required by processor voltage scaling (see Eq. (4) and (5)). Nevertheless, the insertion of repeaters comes at the cost of an increased area and power consumption. For example, the authors' of [19] estimate that in $50nm$ CMOS technology the power dissipated by repeaters will account for 40% of total power consumption.

5.2.2 Fat Wire-Based Bus

Another approach for reducing the wire delay is to increase the physical dimensions of the wire, instead of scaling them down with technology. The usage of "fat" wires, on the top metal layer, has been proposed by [18]. The main advantage of such wires is their low resistance. Provided that $L \cdot R_w / Z_0 < 2ln2$ (L is the wire length, R_w is the wire resistance per unit length and Z_0 its characteristic impedance), they exhibit a transmission line behavior, as opposed to the *RC* behavior in the repeater-based architecture. Using fat wires, the transmission speed approaches the physical limits (the speed of light in the particular dielectric). However, only a limited wire length can be accomplished with the available width of the top metal layer. For example, for a $4mm$ long wire in $180nm$ technology, the authors in [5] obtained a fat wire width of $2\mu m$ on the top metal layer.

The dynamic power consumption of such a fat wire-based bus is mainly due to its large line capacitance. This capacitance is driven by a bus driver, for which the dynamic power consumption is:

$$P_{dri_{dyn}} = s_\tau \cdot f \cdot (C_{dri} + C_w) \cdot V_{dd}^2 \tag{10}$$

where s_τ is the switching activity caused by communication $\tau \in \mathcal{K}$, f is the bus frequency, and C_{dri} and C_w represent the capacitance of the driver and the wire, respectively.

One way to limit the dynamic power is to transmit data at a lower voltage swing, V_{sw}, instead of using the higher bus voltage V_{dd}. Correspondingly, the dynamic power consumed by the driver is given by:

$$P_{dri_{dyn}} = \begin{cases} s_\tau \cdot f \cdot (C_{dri} + C_w) \cdot V_{dd} \cdot V_{sw} & \text{if } V_{sw} \text{ is generated on chip} \\ s_\tau \cdot f \cdot (C_{dri} + C_w) \cdot V_{sw}^2 & \text{otherwise} \end{cases} \tag{11}$$

Furthermore, the driver dissipates a non-negligible leakage power

$$P_{dri_{leak}} = L_g \cdot (V_{dd} \cdot K_3 \cdot e^{K_4 \cdot V_{dd}} \cdot e^{K_5 \cdot V_{bs}} + |V_{bs}| \cdot I_{Ju}) \tag{12}$$

Since the lower swing corresponds to lower signal values, a receiver has to restore the "original" signal. This requires an amplification, for which a dynamic and a leakage power consumption can be calculated as:

$$P_{rec_{dyn}} = s_\tau \cdot f \cdot C_{rec} \cdot V_{dd}^2 \tag{13}$$

$$P_{rec_{leak}} = L_g \cdot (V_{dd} \cdot K_3 \cdot e^{K_4 \cdot V_{dd}} \cdot e^{K_L \cdot (V_{dd}/2 - V_{sw}/2)} \cdot e^{K_5 \cdot V_{bs}} + |V_{bs}| \cdot I_{Ju}) \tag{14}$$

Please note that the leakage power exponentially depends on the difference between the bus voltage $V_{dd_{bus}}$ and the voltage swing V_{sw} (K_L is a technology dependent parameter), i.e., a lower voltage swing results in a higher static energy (while the dynamic power is reduced). In order to find the most efficient solution we need to find an appropriate voltage swing that minimizes the total bus power $P_{bus} = P_{dri_{dyn}} + P_{dri_{leak}} + P_{rec_{dyn}} + P_{rec_{leak}}$. Using the optimal voltage swing can help to significantly reduce the power consumption of the bus [5, 18].

The speed at which the data can be transmitted over the fat wires can be considered to be independent of the used voltage swing V_{sw}. Yet, the bus driver and receiver circuits introduce a delay that is non-negligible and depends on the voltages V_{dd} and V_{bs}. This delay d and the corresponding operational frequency can be calculated according to Eq. (3). In order to lower the power dissipation of the drivers and receivers, it is possible to reduce V_{dd} and/or to increase V_{bs}, which, in turn, necessitates the reduction of the bus speed. However, it is important to note that the optimal voltage swing depends on the V_{dd} and V_{bs} settings of the drivers and receivers. Since these settings are dynamically changed during run-time via voltage scaling, the value of the optimal voltage swing changes as well during run-time. Therefore, in our proposed voltage scaling technique for fat wire-based buses, we consider the dynamic adaption of the voltage swing in order to achieve high energy efficiency.

In addition to the transition overheads in term of energy and time, which are required when scaling the voltages of the drivers and receivers (see Eq. (4) and (5)), the dynamic scaling of the voltage swing necessitates additional overheads. For a transition from V_{sw_j} to V_{sw_k} these overheads in energy and time are given by,

$$\varepsilon_{k,j} = C_{wr} \cdot (V_{sw_k} - V_{sw_j})^2 \quad \text{and} \quad \delta_{k,j} = p_{Vsw} \cdot |V_{sw_k} - V_{sw_j}| \tag{15}$$

where C_{wr} is the wire power rail capacitance and p_{Vsw} is the time/voltage slope.

At this point it is interesting to note that the combination of the fat wire-based and the repeater-based approach would not offer any advantage. Using fat wires eliminates the *RC* behavior of the wire, and thus introducing repeaters would be useless and even increase the energy dissipation and the delay.

6 Combined Voltage Scaling for Processors and Communication Links

In this section, we consider the supply and body bias voltage scaling problem for the processors and the communication links, including the transition overheads in terms of both delay and energy. We first introduce a nonlinear model of the continuous voltage scaling problem, which is optimally solvable in polynomial time, and then outline a heuristic for the discrete voltage scaling case.

6.1 Continuous Voltage Scaling

The nonlinear programming formulation for the continuous voltage scaling problem is given as follows:
Minimize

$$\underbrace{\sum_{k}^{|\mathcal{T}|} E_{dyn_k} + E_{leak_k}}_{\text{computation}} + \underbrace{\sum_{k}^{|\mathcal{K}|} E_{dyn_k} + E_{leak_k}}_{\text{communication}} + \underbrace{\sum_{(k,j) \in \mathcal{E}^\bullet} \varepsilon_{k,j}}_{\text{overhead}} \tag{16}$$

subject to

$$t_k = \begin{cases} NC_k \cdot \frac{(K_6 \cdot L_d \cdot V_{dd_k})}{((1+K_1) \cdot V_{dd_k} + K_2 \cdot V_{bs_k} - V_{th_1})^\alpha} & \text{if } \tau_k \in \mathcal{T} \\ \left\lceil \frac{NB_k}{W_{bus}} \right\rceil \cdot \frac{(K_6 \cdot L_d \cdot V_{dd_k})}{((1+K_1) \cdot V_{dd_k} + K_2 \cdot V_{bs_k} - V_{th_1})^\alpha} & \text{if } \tau_k \in \mathcal{K} \end{cases} \tag{17}$$

$$D_k + t_k \leq D_l \quad \forall (k,l) \in \mathcal{E} \tag{18}$$

$$D_k + t_k + \delta_{k,l} \leq D_l \quad \forall (k,l) \in \mathcal{E}^{\bullet} \tag{19}$$

$$D_k + t_k \leq dl_k \quad \forall \tau_k \in \mathcal{T} \text{ with a deadline} \tag{20}$$

$$D_k \geq 0 \tag{21}$$

$$V_{dd_{min}} \leq V_{dd_k} \leq V_{dd_{max}} \tag{22}$$

$$V_{bs_{min}} \leq V_{bs_k} \leq V_{bs_{max}} \tag{23}$$

$$V_{sw_{min}} \leq V_{sw_k} \leq V_{sw_{max}} \tag{24}$$

The variables that need to be determined in this formulation, in order to minimize the total energy, are the task and communication execution times t_k, the start times D_k, as well as the voltages V_{dd_k}, V_{bs_k}, and V_{sw_k}. The whole formulation can be explained as follows. The total energy consumption (Eq. (16)), with its three contributors (energy consumption of tasks, communication, and voltage transitions) has to be minimized. For all these energies both their dynamic and active leakage components are considered. The dynamic energy of tasks and communications is given by the following equations (derived from the equations discussed in Section 5):

$$E_{dyn_k} = \begin{cases} NC_k \cdot s_k \cdot C_{eff_k} \cdot V_{dd_k}^2 & \text{if } \tau_k \in \mathcal{T} \\ \sum^{N} \lceil \frac{NB_k}{W_{bus}} \rceil \cdot s_k \cdot C_{rep} \cdot V_{dd_k}^2 & \text{if } \tau_k \in \mathcal{K} \text{ on repeaters} \\ \lceil \frac{NB_k}{W_{bus}} \rceil \cdot s_k \cdot C_{fat} \cdot V_{dd_k} \cdot V_{sw_k} & \text{if } \tau_k \in \mathcal{K} \text{ on fat wires (inter)} \\ \lceil \frac{NB_k}{W_{bus}} \rceil \cdot s_k \cdot C_{fat} \cdot V_{sw_k}^2 & \text{if } \tau_k \in \mathcal{K} \text{ on fat wires (extern)} \end{cases} \tag{25}$$

where $C_{rep} = C_d + C_w + C_g$ and $C_{fat} = C_{dri} + C_w + C_{rec}$ are the total capacitances that have to be charged by bus implementation either repeater-based or fat wire-based, respectively. Further, in the case of fat wire implementations we have to distinguish between the chip-intern or chip-extern generation of the voltage swing.

The leakage power dissipation of processors and repeater-based buses are given by,

$$E_{leak_k} = L_g(K_3 \cdot V_{dd_k} \cdot e^{K_4 \cdot V_{dd_k}} \cdot e^{K_5 \cdot V_{bs_k}} + I_{Ju} \cdot |V_{bs_k}|) \cdot t_k \tag{26}$$

and for fat wire-based buses we need to additionally account for the leakage in the receiver (see Eq. (12) and (14)), which is given by,

$$E_{leak_k} = (P_{dri_{leak}} + P_{rec_{leak}}) \cdot t_k \tag{27}$$

The energy overhead due to voltage transitions between two activities running on the same component is given by Eq. (4) and (15).

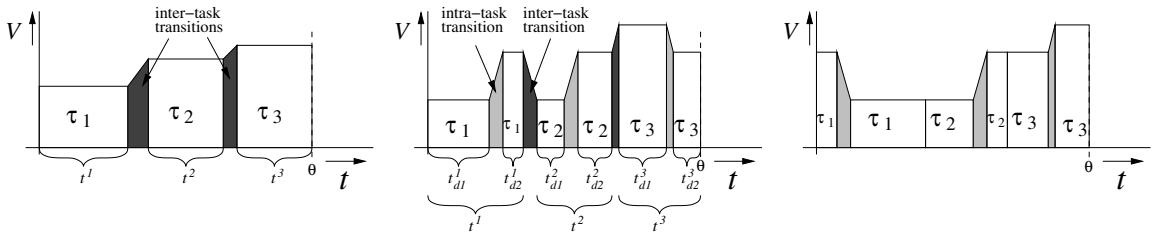
The minimization has to comply to the following relations and constraints. The task execution time has to be equivalent to the number of clock cycles of the task multiplied by the circuit cycle time for a particular V_{dd_k} and V_{bs_k} setting, as expressed by the first line of Eq. (17). The communication time is computed in the second line of Eq. (17) from the number of bits, bus width and the bus cycle time. The time overhead due to voltage transitions on a processor is captured by Eq. (5) and (15). We have to express the following precedence relations: A task τ_l can only start its execution after all its predecessors tasks τ_k have finished their execution (the sum of a tasks start time D_k and its execution time t_k), see Eq. (18). In addition, for two consecutive tasks on the same component, the start time of the later task τ_l can only begin after its predecessors on the same component has finished execution ($D_k + t_k$) and the mode transition has been carried out ($\delta_{k,l}$), see Eq. (19). The sum of a tasks start time D_k and its execution time t_k, which is its finishing time, has to respect the task deadline dl_k, see Eq. (20). Task start times and communication start times must be positive (Eq. (21)). The imposed voltage ranges should be respected (Eq. (22), (23), and (24)). It should be noted that the objective (Eq. (16)) as well as the task execution times (Eq. (17)) are convex functions. Hence, the problem falls into the class of general convex nonlinear optimization problems. Such problems can be solved optimally in polynomial time [15].

6.2 Discrete Voltage Scaling

Since processor designs are most often restricted to a finite set of discrete performance modes, it is not possible to apply the continuously selected voltages directly. In [15], we have demonstrated that discrete voltage scaling is NP-hard. Therefore, we propose the following heuristic to effectively transform the continuously selected voltages, produced by solving the NLP formulation from Section 6.1, into discrete values. According to the operational frequencies that are calculated as result of the continuously selected voltages, the two surrounding discrete performance modes are chosen, $f_{d1} < f_{con} < f_{d2}$. That is, the execution of a task/communication is split into two regions with t_{d1} and t_{d2} being the execution times in mode with f_{d1} and f_{d2}, respectively. Fig. 5(a) and 5(b) indicate this transformation for an application with three tasks. In the continuous scaling case, Fig. 5(a), each of the tasks executes at a single voltage setting, i.e., the voltages are changed only between tasks. In the discrete case, the voltage setting is changed during the task execution.Of course, the required time overhead for the mode change has to be considered as well, i.e., $t = t_{d1} + t_{d2} + \delta$, where t is the task execution time with continuous voltage setting. Please note that the discrete voltage swing in the case of fat wire-based buses is determined by the discrete settings for the drivers and receivers, i.e., for each setting of these components there exists a corresponding voltage swing. In general, executing activities in two performance modes leads to close to optimal discrete voltage scaling [15]. Furthermore, restricting the execution to two settings avoids unnecessary intra-task transitions which cause energy and time overheads. Having determined the discrete performance mode settings, the inter-task transition overheads are reduced by reordering the mode sequence of each task. From a task execution point of view, mode reordering does not have any effect. That is, if a certain task requires 600 cycles to execute, then it is equivalent to first execute 100 cycles at a lower voltage and then 500 at a higher voltage, or to first execute 500 cycles at a higher voltage and then 100 cycles at a lower voltage. We reorder the modes in such a way that a task starts execution with its execution in the lower (higher) performance mode if the preceding tasks on the same component finishes execution in the lower (higher) performance mode. This is outlined in Fig. 5(c). It is worthwhile to mention that this reordering only affects the order in which the voltages are altered, that is, the code execution order inside the task remains unchanged. While this reordering technique is optimal for components that offer two performance modes, this is not true for components with three or more modes. Nevertheless, as demonstrated by our experiments, this heuristic is fast and efficient. Of course, the additional slack produced as a results of the reduced transition times is exploited as well.

7 Experimental Results

To validate the applicability of the presented techniques, we have carried out several experiments using numerous generated benchmarks as well as a realistic GSM voice transcoder example. The automatically generated benchmarks consist of 120 task graphs containing between 50 and 300 tasks, which are mapped and scheduled onto architectures composed of 2 to 5 processors, interconnected via 1 to 4 buses either implemented repeater-based or fat wire-based. In the continuous voltage scaling case the processor voltage pairs (V_{dd},V_{bs}) are varied between $(0.6,-1)V$ and $(1.8,0)V$, while the discrete voltage levels are $m_z = \{(1.8,0),(1.4,-0.2),(0.8,-0.6),(0.6,-1)\}$. The voltage ranges for repeater-based systems are identical to the possible processor volt-

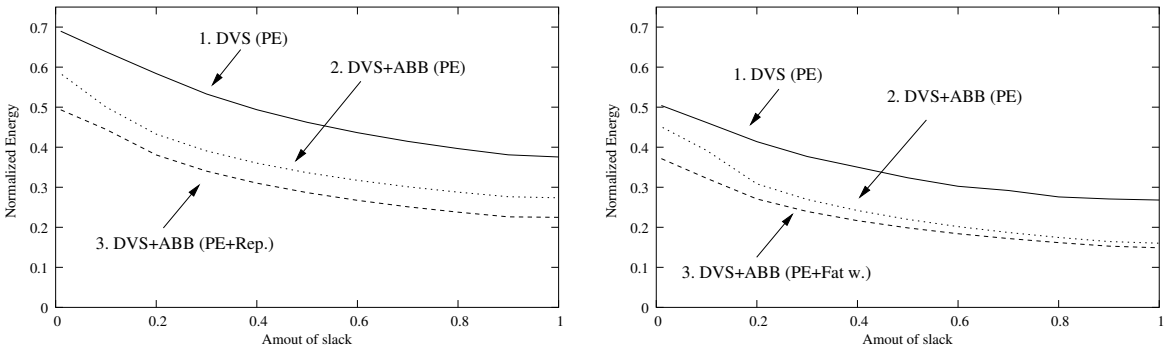


(a) Continuous Voltage Schedule (solely inter−task mode transitions)

(b) Discrete Voltage Schedule (5 performance mode transitions)

(c) Reordered Discrete Voltage Schedule (3 performance mode transitions)

Figure 5. Performance mode reordering



(a) Energy dissipation of a reapter−based system

(b) Energy dissipation of a fat wire−based system

Figure 6. Optimization Results of different Implementations

age settings. For the fat wire-based buses the continuous voltage swing values can be set between 0.2 and $1V$, and for the discrete case it can be adjusted to $m_z = 0.2, 0.3, 0.4, 0.6, 1V$. The technology dependent parameters of these processors and buses were considered to correspond to a CMOS fabrication process in $50nm$, for which the leakage power represents approximately 50% of the total power consumed. For experimental purpose the amount of deadline slack in each benchmark was varied over a range 0 to 100%, using a 10% increment. Furthermore, the amount of communication within the generated benchmarks was varied between 10 to 50% of the total execution time, with an increment of 10%. Overall, these experiments resulted in 2400 performed evaluations, carried out with the aim to achieve representative average values (based on normalized energy value of each evaluated task graph).

The first set of experiments was conducted with the aim to investigate the energy savings that are achievable when dynamically scaling the supply voltage as well as body bias voltage of bus repeaters. The 32bit-wide bus architecture under consideration consisted of 27 repeaters per bit-line of which each has a total length of $27.4mm$. The capacitance of single wire including the repeaters was estimated as $7.2pF$, using the power optimized data from [2]. Fig. 6(a) shows the outcomes of three system configurations for different amounts of system slack. All plots have been normalized against the nominal energy dissipation of unscaled systems with repeater-based buses. The first plot gives the energy consumption for systems in which the repeaters' voltages are kept fixed, while the supply voltage (but not the body-bias voltage) of the processors is dynamically scaled. The second plot represents a system in which the repeater settings are still kept fixed, while combined V_{dd} and V_{bs} scaling is applied to the processors. The third plot indicates the systems in which the repeater-based bus as well as the processors are scaled by changing V_{dd} and V_{bs}. Please note that Fig. 6(a) gives the energy values for systems with a communication amount of 30%, compared to the total execution time. Inspecting the graphs reveals that the highest energy savings are achieved by considering the combined V_{dd} and V_{bs} continuous voltage scaling scheme on the buses as well as on the processors (plot 3). We can also observe that the energy efficiency is increased by approx. 12% if combined voltage scaling is applied on the bus (difference between plot 2 and 3). Generally, the combined V_{dd} and V_{bs} scaling yields higher energy saving (around 30%) than the V_{dd}-only scaling (difference between plot 1 and 2) [1]. Since all plots in Fig. 6(a) represent the results for continuous voltage scaling, it is interesting to note that the proposed heuristic for discrete voltage scaling (Section 6.2) achieves results that are within 4% of the values obtained at continuous voltage levels. This difference is only partially due to the suboptimal nature of the heuristic, but mainly due to the fact that, by definition, discrete voltages cannot achieve the same energy savings then continuous ones. It is important to note that the efficiency difference of about 12% on average, between implementations with and without bus voltage scaling is preserved also when discrete voltage levels are used.

In the second set of experiments, shown in Fig. 6(b), we investigate the achievable energy savings on a fat wire-based bus system, assuming the same bus-width as in the repeater based approach. However, fat wires are considered to be suitable only for short distance connections. In the following we consider a length of $4mm$ with a single line capacitance $609fF$. Similarly to the previous experiments, the plots 1 and 2 represent systems in which only the processing elements are scaled (V_{dd} only for plot 1 and combined V_{dd} and V_{bs} for plot 2), while the third plot indicates systems in which the buses are voltage scaled in terms of V_{dd}, V_{bs}, and V_{sw}. As expected, the fully voltage scalable systems, achieve the best energy savings, with reductions between 4% to 18% compared to systems with fixed bus voltages. Again, applying the heuristic for discrete voltage scaling shows that results comparable to the continuous

[1] Please note that energy savings are achieved even at zero deadline slack. This means that according to the given schedule, the task set finishes on deadline when executed at the highest voltage. However, even in this case, due to the fact that the application executes on a multiprocessor system, initial slack (idle processor time) is available in the system and can be exploited by voltage scaling.

case (within 4%) can be achieved. Please note that we do not try to advocate here neither repeater-based nor fat wire-based approaches and to show that one is better than the other, but rather we use our experiments to validate the applicability of voltage scaling for both approaches. In general it can be said that both approaches have their own niche for which they are most suitable — repeaters for lengthly connections up to the cross-sectional die distance, and fat wires for short distance connections. At this point it is also interesting to note that the optimization times for the individual applications with up to 300 tasks were below 1 minute, using the MOSEK optimization software [1] on a 2GHz AMD Athlon PC.

Experimental experience has shown that with an increasing amount of communication data, the bus voltage scaling approach achieves increasingly higher energy reductions. If, for example, the time required for communications is around 15% of the total execution time, the energy savings due to bus voltage scaling are around 10%. With communication time around 30%, the energy savings become around 16%.

In addition to the above presented results, we have conducted experiments using a real-life GSM voice codec application, in order to validate the real-world applicability of the presented techniques. A detailed description of this application can be found in [16]. The GSM codec consists of 87 tasks and 137 data dependencies, which are considered to run on an architecture composed of 3 processors (with two voltage modes $((1.8V,-0.1V)$ and $(1.0V,-0.6V))$), communicating over a repeater-based shared bus. At the highest voltage mode, the application reveals a deadline slack close to 10%. Switching overheads are characterized by $C_r = 1\mu F, C_s = 4\mu F$, $p_{Vdd} = 10\mu s/V$, and $p_{Vbs} = 10\mu s/V$. Tab. 1 shows the resulting total energy consumptions for six different situations. The

Approach	VS type	E_{tot} (mJ)	Reduc. (%)
Nominal	—	2.273	—
PE (V_{dd})	cont.	2.174	4.4
PE (V_{dd},V_{bs})	cont.	1.931	15.2
Heur. PE (V_{dd},V_{bs})	disc.	2.026	10.9
PE+BUS (V_{dd},V_{bs})	cont.	1.762	22.5
Heur. PE+BUS(V_{dd},V_{bs})	disc.	1.853	18.5

Table 1. Optimization results for voice codec algorithm

first column denotes the used voltage scaling technique and the second indicates if continuous or discrete voltages were considered. The third and fourth column give the energy consumption and achieved reduction in percentage for each scaling approach. For instance, according to the second row, the system dissipates an energy of $2.273mJ$ at nominal voltage settings, i.e., without any voltage scaling. This value serves as a baseline for the reductions indicated in the fourth column. The third and fourths row present the results of systems in which the bus remains unscaled while the processors are either V_{dd} or V_{dd} and V_{bs} scaled over a continuous range. As we can observe, savings of 4.4 and 15.2% are achieved. In order to adapt the continuous selected voltages towards the two discrete voltage settings at which the processor can possibly run, we apply our heuristic outlined in Section 6.2. The achieved reduction in the discrete case is 10.9% (row 5). Nevertheless, as shown by the values given in row 6, it is possible to further reduce the energy by scaling the repeater-based bus. Compared to the baseline, a saving of 22.5% is achieved. Using the discrete voltage heuristic, the final energy dissipation results in $1.853mJ$, which is 18.5% below the unscaled system.

Summarizing the experiments, we have seen that dynamic voltage scaling and adaptive body-biasing on repeater-based communication infrastructures as well as dynamic voltage swing scaling on fat wire-based buses can help to reduce the energy consumption of communication-intensive computing systems.

8 Conclusions

We have proposed a new technique for combined voltage scaling of processors and communication links, which reduces the dynamic and leakage power consumption. This is achieved by scaling the supply and body bias voltage of processors as well as repeater-based buses. Furthermore, in the case of fat wire-based buses, energy efficiency is obtained by additionally scaling the voltage swing. For this purpose, we have used a set of accurate delay and energy models. By applying the introduced technique to numerous examples, including a realistic GSM codec, we have demonstrated that bus voltage scaling can achieve up to 16% higher energy savings when compared to fixed voltage bus systems.

References

[1] MOSEK optimization software. http://www.mosek.com.

[2] K. Banerjee and A. Mehrotra. A Power-Optimal Repeater Insertion Methodology for Global Interconnects in Nanometer Designs. *IEEE Trans. on Electron Devices*, 49(11):2001–2006, Nov 2002.

[3] L. Benini, G. De Micheli, E. Macii, D. Sciuto, and C. Silvano. Address bus encoding techniques for system-level power optimization. In *Proc. DATE'98*, :861–867, 1998.

[4] Shekhar Borkar. Design Challenges of Technology Scaling. *IEEE Mirco*, :23–29, July 1999.

[5] P. Caputa and C. Svensson. Low-Power, Low-Latency Global Interconnects. In *Proc. IEEE ASIC/SOC'02*, :394–398, 2002.

[6] Anantha P. Chandrakasan and Robert W. Brodersen. *Low Power Digital CMOS Design*. Kluwer Academic Publisher, 1995.

[7] W. Fornaciari, D. Sciuto, and C. Silvano. Power Estimation for Architectural Exploration of HW/SW Communication on System-Level Buses. In *CODES'99*, :152–156, May 1999.

[8] Flavius Gruian and Krzysztof Kuchcinski. LEneS: Task Scheduling for Low-Energy Systems Using Variable Supply Voltage Processors. In *Proc. ASP-DAC'01*, :449–455, Jan 2001.

[9] Cheng-Ta Hsieh and Massoud Pedram. Architectural Energy Optimization by Bus Splitting. *IEEE Trans. on CAD*, 21(4):408–414, April 2002.

[10] Y. Ismail and E. Friedman. Repeater Insertion in RLC Lines for Minimum Propagation Delay. In *Proc. ISCAS'99*, :404–407, 1999.

[11] P. Kapur, G. Chandra, and K. Saraswat. Power Estimation in Global Interconnects and its Reduction using a Novel Repeater Optimiazation Methodology. In *Proc. DAC'02*, :461–466,2002.

[12] C. Kim and K. Roy. Dynamic Vth Scaling Scheme for Active Leakage Power Reduction. In *Proc. DATE02*, :163–167, March 2002.

[13] J. Liu, P. Chou, and N. Bagherzdeh. Communication Speed Selection for Embedded Systems with Networked Voltage-Scalable Proceossors. In *Proc. CODES'02*, :169–174, 2002.

[14] S. Martin, K. Flautner, T. Mudge, and D. Blaauw. Combined Dynamic Voltage Scaling and Adaptive Body Biasing for Lower Power Microprocessors under Dynamic Workloads. In *Proc. ICCAD-02*, :721-725, 2002.

[15] A. Andrei, M. Schmitz, P. Eles, Z. Peng, B. Al Hashimi. Overhead-Concious Voltage Selection for Dynamic and Leakage Energy Reduction in Time-Constrained Systems In *Proc. DATE-04*, :518–523, 2004.

[16] M. T. Schmitz. *Energy Minimisation Techniques for Distributed Embedded Systems*. PhD thesis, University of Southampton, February 2003.

[17] L. Shang, L. Peh, and N. Jha. Power-efficient Interconnection Networks: Dynamic Voltage Scaling with Links. *Comp. Arch. Letters*, 1(2):1–4, May 2002.

[18] C. Svensson. Optimum Voltage Swing on On-Chip and Off-Chip Interconnects. *IEEE J. Solid-State Circuits*, 36(7):1108–1112, July 2001.

[19] D. Sylvester and K. Keutzer. Impact of Small Process Geometries on Microarchitectures in Systems on a Chip. *Proceedings of the IEEE*, 89(4):467–489, April 2001.

[20] G. Varatkar and R. Marculescu. Communication-Aware Task Scheduling and Voltage Selection for Total System Energy Minimization. In *Proc. ICCAD'03*, :510–517, 2003.

[21] G. Wei, J. Kim, D. Liu, S. Sidiropoulos, and M. Horowitz. A Variable-Frequency Parallel I/O Interface with Adaptive Power-Supply Regulation. *IEEE J. Solid-State Circuits*, 35(11):1600–1610, Nov 2000.

[22] L. Yan, J. Luo, and N. Jha. Combined Dynamic Voltage Scaling and Adaptive Body Biasing for Heterogeneous Distributed Real-time Embedded Systems. In *Proc. ICCAD'03*, :30–37, 2003.

[23] Frances Yao, Alan Demers, and Scott Shenker. A scheduling model for reduced CPU energy. In *IEEE FOCS*, :374–382, 1995.