ACSIJ Advances in Computer Science: an International Journal, Vol. 3, Issue 1, No.7 , January 2014
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

# A Robust and Energy-Efficient DVFS Control Algorithm for GALS-ANoC MPSoC in Advanced Technology under Process Variability Constraints

**Sylvain Durand*, Hatem Zakaria†, Laurent Fesquet‡, Nicolas Marchand***

* *Control Department of GIPSA-lab, CNRS, University of Grenoble*
*Grenoble, France*

† *Electrical Engineering Department, Benha Faculty of Engineering, Benha University*
*Benha, Egypt*

‡ *TIMA Laboratory, Grenoble University*
*Grenoble, France*

*E-mail:* `sylvain@durandchamontin.fr`

**Abstract**

*Many Processor Systems-on-Chip* (MPSoC) have become tremendously complex systems. They are more sensitive to variability with technology scaling, which complicates the system design and impact the overall performance. Energy consumption is also of great interest for mobile platforms powered by battery and power management techniques, mainly based on *Dynamic Voltage and Frequency Scaling* (DVFS) algorithms, become mandatory. A *Globally Asynchronous Locally Synchronous* (GALS) design alleviate such problems by having multiple clocks, each one being distributed on a small area of the chip (called island), whereas an *Asynchronous Network-on-Chip* (ANoC) allow to communicate between the different islands. A robust technique is proposed to deal with a GALS-ANoC architecture under process variability constraints using advanced automatic control methods. The approach relaxes the fabrication constraints and help to the yield enhancement. Moreover, energy savings are even better for the same perceived performance with the obtained variability robustness. The case study is an island based on a MIPS R2000 processor implemented in STMicroelectronics 45nm technology and validated with fine-grained simulations.

***Keywords:*** *Predictive control, low power MPSoC, process variability robustness, DVFS, GALS, ANoC.*

## 1. Introduction

The upcoming generations of integrated systems, as Many Processor Systems-on-Chip (MPSoC), require drastic technological evolution since they have reached limits in terms of power consumption, computational efficiency and fabrication yield. Moreover, *process variability* is one of the main problems in current nanometric technologies. This phenomenon refers to unpredictability, inconsistency, unevenness, and changeability associated with a given feature or specification. It has became one of the leading causes for chip failures and delayed schedules at a sub-micrometric scale and complicates system design by introducing uncertainty about how a fabricated system will perform [1]. Although a circuit or chip is designed to run at a nominal clock frequency, the fabricated implementation may vary far from this expected performance. Moreover, some cores may behave differently inside the same chip [2]. To solve these problems, MPSoCs in advanced technology require a dynamic power management in order to highly reduce the energy consumption. Architectural issues are also needed for helping the yield enhancement of such circuits with strong technological uncertainties.

Whereas leakage power is a significant contributor to the total power, the average power consumption and the energy dissipation are dominated by the *dynamic power* in current embedded CMOS integrated circuits. The energy reduction is a quadratic function of the voltage and a linear function of the clock frequency [3]. As a result, Dynamic Voltage Scaling (DVS) can be used to efficiently manage the energy consumption of a device [4]. Supply voltage can be reduced whenever slack is available in the critical path, but one has to take care that scaling the voltage of a microprocessor changes its speed as well. Therefore, adapting the supply voltage is very interesting but implies the use of Dynamic Frequency Scaling (DFS) to keep correct the system behavior. The addition of DFS to DVS is called Dynamic Voltage and Frequency Scaling (DVFS) and results in simultaneously managing both parameters. In many cases, the only performance requirement is that the tasks meet a deadline, where a given task has to be computed before a given time. Such cases create opportunities to run the processor at a lower computing level and achieve the same perceived performance while consuming less energy [5, 6, 7]. As a result, closed-loop control laws are required to manage the energy-performance tradeoff in MPSoCs and new strategies are developed in this sense in this paper.

In addition, embedded integrated systems have two means of implementation. Firstly, the conventional clocked circuits with their global synchronization – in which one faces the huge challenge of generating and distributing a low-skew global clock signal and reducing the clock tree power consumption of the whole chip – makes them difficult for implementation. Secondly, MPSoCs built with predesigned IP-blocks running at different frequencies need to integrate all the IP-blocks into a single chip. Therefore, global synchronization tends to be impractical [2]. By removing the globally distributed clock, Globally Asynchronous Locally Synchronous (GALS) circuits provide a promising solution. A GALS design allows each locally synchronous island to be set independently, the island becoming a Voltage/Frequency Island (VFI), where an Asynchronous Network-on-Chip (ANoC) is used as the mechanism to communicate between the different VFIs. As a consequence, a GALS-ANoC architecture is more convenient for DVFS than the standard synchronous approach, because the power consumption of the whole platform depends on the supply voltage and the clock frequency applied to each VFI [4, 8]. A GALS-ANoC architecture also mitigates the impact of process variability [9], because a globally asynchronous system does not require that the global frequency was dictated by the longest path delay of the whole chip, i.e. the critical path. Each clock frequency is only determined by the slowest path in its VFI.

The present paper is a proof of concept, gathering different techniques together to propose a robust design of an automatic control algorithm dealing with nano-scale process variations. The setup is based on an energy-efficient DVFS technique (previously developed in [10]) applied to a GALS-ANoC MPSoC architecture. A fast predictive control law i) deduces the best two power modes to apply for a given task and then ii) calculates when to switch from one power mode to the other, where a *power mode* is defined by a voltage/frequency pair supplying a VFI. The switching time instant is such that the energy consumption is minimized while the task fits with its deadline, guaranteeing good performance. This is repeated for all the tasks to treat. The control decisions are sent to the voltage and frequency actuators (respectively a Vdd-hopping and a programmable self-timed ring) while speed sensors provide real-time measurements of the processor speed. The control strategy is highly robust to uncertainties since the algorithm does not need any knowledge on the system parameters. It is dynamically (on line) computed, which ensures that it works for any type of tasks (whereas only periodic tasks are generally treated in the literature). Furthermore, the control strategy is simple enough to limit the overhead it may introduces (see [10] for further details).

The rest of the document is organized as follows. In section 2, it is explained why a closed-loop architecture becomes essential in nanometric technologies. The proposed solution is introduced for a practical microprocessor (i.e. a MIPS R2000 processor), detailing also the actuators and sensors. The robust and energy-efficient DVFS control algorithm is depicted in section 3. Stability and robustness are analyzed too. Fine-grained simulation results are then presented in section 4. Note that only simulation results are provided in the present paper. An implementation on a real chip will be possible after the hardware and/or software cost of the proposed approach will be evaluated. Nevertheless, preliminary results are obtained for a realistic island implemented in STMicroelectronics 45nm technology.

## 2. Controlling uncertainty and handling process variability

The main points of interest of the proposal is to handle the uncertainty of a processor (or processing node) over a GALS-ANoC design and reduce its energy consumption. This is possible by means of automatic control methods. Using both an ANoC distributed communication scheme and a GALS approach offer an easy integration of different functional units thanks to a local clock generation [11]. Moreover, this allows better energy savings because each functional unit can easily have its own independent clock frequency and voltage. Therefore, a GALS-ANoC architecture appears as a natural enabler for distributed power management systems as well as for local DVFS. It also mitigates the impact of process variability [9].
An architecture for DVFS and process quality management is presented in Fig. 1 (note that the description in [12] gives more details about this architecture and the processing node circuit.) The operating system (or scheduler) provides a set of information $ref$ (the required computational speed, in terms of number of instructions and deadline, for each task to treat in a given VFI), eventually through the ANoC. This information about real-time requirements of the application enables to create a computational workload profile with respect to time. There are also speed sensors (not represented in Fig. 1) embedded in each processing unit in order to provide real-time measurements of the processor speed $\omega$ (in million of instructions per second for instance). Consequently, combining such a profile with such a monitoring makes possible to apply a power/energy management allowing application deadlines to be met. On the other hand, the DVFS hardware part contains voltage and frequency converters, that are a Vdd-hopping and a Programmable Self-Timed Ring (PSTR) for supplying the voltage $V_{dd}$ and the frequency $f_{clk}$ respectively. A controller then dynamically controls these power actuators in order to satisfy the application computational needs with an appropriate management strategy. It processes the error between the measured speed and the speed setpoint information within a closed-

loop system, and by applying a well-suited compensator sends the desired voltage and frequency code values to the actuators (denoted $V_{level}$ and $f_{level}$). Consequently, the system is able to locally adapt the voltage and clock frequency values clock domain by clock domain. Furthermore, the ANoC is the reliable communication path between the different domains. Data communications between two VFIs can fix the speed to the slowest communicating node in order to have a secure communication without metastability problem and an adaptation to process variability too [13].

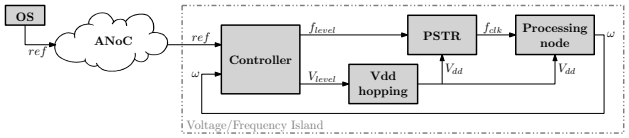



Figure 1: DVFS control of a GALS-ANoC MPSoC architecture: control of energy-performance tradeoff in a VFI.

Without lack of generality, the DVFS technique is supposed to be implemented with two discrete voltage levels $V_{high}$ and $V_{low}$, with $V_{high} > V_{low} > 0$. Also, $\overline{\omega}_{high}$ and $\overline{\omega}_{low}$ denote the maximal computational speeds when the system is running under high and low voltage with its maximal associated clock frequency, with $\overline{\omega}_{high} > \overline{\omega}_{low} > 0$ by construction.

### 2.1 MIPS R2000 as a processing node

The MIPS R2000 is a 32-bit Reduced Instruction Set Computer (RISC). Due to its simplicity in terms of architecture, programming model and instruction set, as well as availability as an open core, it has been used as a processing node (see Fig. 1) in our study case. The whole GALS-ANoC island implementation is done using *Synopsis Design Vision tool* with STMicroelectronics $45\,nm$ technology libraries (CMOS045_SC_9HD_CORE_LS).

### 2.2 Vdd-hopping for voltage scaling

The Vdd-hopping mechanism was described in [14]. Two voltages can supply the chip. The system simply goes to $V_{low}$ or $V_{high}$ with a given transition time and dynamics that depend upon an internal control law (one could refer to the reference above for more details). Considering that this inner-loop is extremely fast with respect to the loop considered in this paper, the dynamics of the Vdd-hopping can be neglected.

### 2.3 Programmable self-timed ring for frequency scaling and variability management

The application of the proposed DVFS to a system requires the use of a process variability robust source for generating adjustable clocks. Today, many studies are oriented to Self-Timed Ring (STR) oscillators which present well-suited characteristics for managing process variability and offering an appropriate structure to limit the phase noise. Therefore, they are considered as a promising solution for generating clocks even in presence of process variability [15]. Moreover, STRs can easily be configured to change their frequency by just controlling their initialization at reset time.

#### 2.3.1 Self-timed rings

A STR is composed of several nested stages which behavior is mainly based on the "tokens" and "bubbles" propagation rule. A given stage $i$ contains a bubble (respectively a token) if its output is equal (resp. not equal) to the output of the stage $i+1$. The number of tokens and bubbles are respectively denoted $N_T$ and $N_B$, with $N_T + N_B = N$, where $N$ is the number of the ring stages. For keeping the ring oscillating, $N_T$ must be an even number. One can think about this as the duality of designing the inverter ring by odd number of stages. Each stage of the STR contains either a token or a bubble. If a token is present in a stage $i$, it will propagate to the stage $i+1$ if and only if this latter contains a bubble, and the bubble of stage $i+1$ will then move backward to stage $i$.

#### 2.3.2 Programmable self-timed rings

The oscillation frequency in STRs depends on the initialization (number of tokens and bubbles and hence the corresponding number of stages) [16]. Programmability can be simply introduced to STRs by controlling the tokens/bubbles ratio and the number of stages. Programmable Self-Timed Ring (PSTR) uses STR stages based on Muller gates which have a set/reset control to dynamically insert tokens and bubbles into each STR stage. Moreover, in order to be able to change the number of stages, a multiplexer is placed after each stage [17]. This idea was also extended to have a fully Programmable/Stoppable Oscillator (PSO) based on the PSTR. Look-up tables loaded with the initialization token control word (i.e. to control $^{N_T}/_{N_B}$), and the stage control word (i.e. to control $N$) was used to program the PSTR with a chosen set of frequencies.

#### 2.3.3 PSTR programmability applied to MIPS R2000

Presently, the variability is captured in the design by using simulation corners, which correspond to the values of process parameters that deviate by a certain ratio from their typical value. In the STMicroelectronics $45\,nm$ CMOS technology, three PVT (Process, Voltage, and Temperature) corners are available, denoted *Best*, *Nominal* and *Worst*. All standard logic cells were characterized at each of these corners. Since, our main goal is to define the optimal oper-

ating clock frequency needed by the processing workload that compensates for the propagation delay variations due to the variability impact. Therefore, the critical path delay of the synthesized MIPS R2000 with respect to the supply voltage is analyzed at the different PVT corners. The resulting optimal frequencies needed by the MIPS R2000 at two specified voltage levels are finally defined in Table 1 for the three process variability corners.

Table 1: Optimal clock frequencies required to compensate for the process variability.

| **Voltage level** | **Clock frequency** for different process variability conditions | | |
|---|---|---|---|
| | *Worst* | *Nominal* | *Best* |
| 0.95 V | 60 MHz | 75 MHz | 85 MHz |
| 1.1 V | 95 MHz | 115 MHz | 145 MHz |

### 2.4 Speed sensors for real-time measurement

Any closed-loop scheme requires measurements to compare the measure with a given setpoint to reach. In the present study case, speed sensors are embedded in each VFI. They provide a real performance measurement of the processor activity, i.e. its computational speed (in number of instructions per second).
Speeds sensors play a critical role and must be carefully selected. A *reference clock* fixes the time window where the number of instructions are counted. Its period is crucial since it determines the accuracy of the calculated average speed and the system speed response. Therefore, according to the set of clock frequencies available for the MIPS R2000 (see Table 1), the reference clock was chosen to be 2 MHz in order to count a considerable amount of instructions with a proper system response. To conclude, the computational speed is now applied in terms of number of instructions executed per $500\,ns$ to the digital controller.

## 3. Energy-efficient DVFS control with strong process variability robustness

The control of the energy-performance tradeoff in a voltage scalable device consists in minimizing the energy consumption (reducing the supply voltage) while ensuring good computational performance (fitting the tasks with their deadline). This is the aim of the controller introduced in Fig. 1, which dynamically calculates a speed setpoint that the system will have to track. This setpoint is based on i) the measurement of the computational speed $\omega$ (obtained with the speed sensors) and ii) some information provided by a higher level device (the operating system or scheduler) for each task $T_i$ to treat. Information are the computational workload, i.e. the number of instructions $\Omega_i$ to execute, and the deadline $\Delta_i$. Furthermore, let $\Lambda_i$ denote the laxity, that is the remaining available time to complete the task $T_i$.

### 3.1 Speed setpoint building

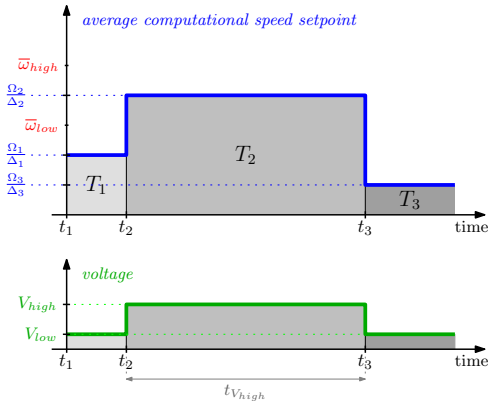
The presence of deadline and time horizon to compute tasks naturally leads to predictive control [18]. The main idea of the predictive strategy is firstly intuitively explained and its formal expression is given in the sequel.

#### 3.1.1 Intuitive DVFS control technique

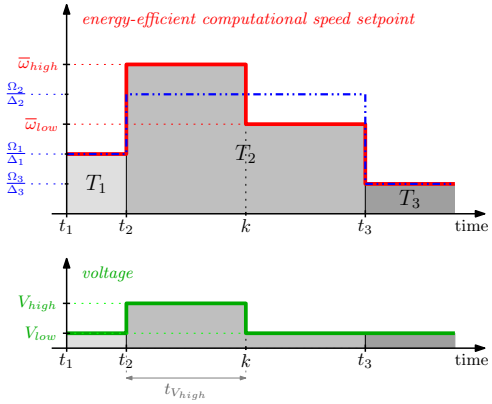
A naive DVFS technique applies a constant power mode for each task $T_i$ to treat, as represented in Fig. 2(a). The average speed setpoint, that is the ratio $\Omega_i/\Delta_i$, is constant for a given task. Therefore, if the computational workload of a given task is higher than the processor capabilities under low voltage, i.e. $\Omega_i/\Delta_i > \overline{\omega}_{low}$, then the VFI executes the whole task with the penalizing high voltage $V_{high}$ and its associated frequency in order not to miss its deadline. This is the case for $T_2$ for instance.
Note that if $\Omega_i/\Delta_i > \overline{\omega}_{high}$ for a given task to treat, the execution of the task is not *feasible* by the desired deadline.

#### 3.1.2 Energy-efficient DVFS control technique

To overcome such intuitive approaches, an energy-efficient control has been proposed in [10]. The idea is to *i) minimize the energy consumption* by reducing as much as possible the penalizing high voltage level *ii) while ensuring the computational performance* required so that the tasks meet their deadline. A task is thus split into two parts, see Fig. 2(b). Firstly, the VFI begins to run under high voltage $V_{high}$ (if needed) with the maximal available speed $\overline{\omega}_{high}$ in order to go faster than required. Then, the task is finished under low voltage $V_{low}$ with a speed lower than or equal to the maximal speed at low voltage $\overline{\omega}_{low}$ which, consequently, highly reduces the energy consumption, where *highly* is task dependent. The task is hence executed with a given ratio between high and low voltage. A key point in this strategy is that the switching time to go from $V_{high}$ to $V_{low}$ has to be suitably calculated in order to meet the task deadline. This is done thanks to a *fast predictive control* law.
In fact, the lower is the supply voltage the better will be the energy savings. For this reason, only one possible frequency $F_{high}$ is possible when running under $V_{high}$ (in order to minimize the penalizing high voltage running time). On the other hand, several frequency levels $F_{low_n}$ are possible under low voltage $V_{low}$ because, as the energy consumption could not be reduced anymore, the degree of freedom on the frequency will allow to fit the task with its deadline (as much as this is possible). In the present case, two frequency levels exist under $V_{low}$, with $F_{low_1} \geq F_{low_2}$. Whereas the maximal levels $F_{high}$ and $F_{low_1}$ are determined from the




(a) Intuitive average speed setpoint.




(b) Energy-efficient speed setpoint.

Figure 2: Different computational speed setpoint buildings can be used to save energy consumption while ensuring good performance.

optimal frequency values in Table 1 (regarding the variability of the chip), the second frequency level at low voltage is chosen equal to $F_{low_1}/2$. This enables to have 3 dB reduction in the power consumption.

### 3.2 Fast predictive control

A predictive issue can be formulated as an optimization problem but an optimal criteria is too complex to be implemented in an integrated circuit. Further, a *fast* predictive control consists in taking advantage of the structure of the dynamical system to make faster the determination of the control profile, see e.g. [18]. In the present case, the closed-loop solution yields an easy algorithm as one simply needs to calculate the so-called *predicted speed* $\delta$, defined as the speed required to fit the task with its deadline regarding what has already been executed

$$\delta(t) = \frac{\Omega_i - \Omega(t)}{\Lambda_i} \tag{1}$$

where $\Omega(t)$ is the number of instructions executed from the beginning of the task $T_i$. The dynamical energy-efficient speed setpoint $\omega_{sp}$ is then directly deduced from the value of the predicted speed

$$\omega_{sp}(t) = \begin{cases} \overline{\omega}_{high} & \text{if} \quad \delta(t) > \overline{\omega}_{low} \\ \overline{\omega}_{low} & \text{elsewhere} \end{cases} \tag{2}$$

and so are the voltage and frequency levels. Indeed, the system has to run under $V_{high}/F_{high}$ when the required setpoint is higher than the capabilities under low voltage, else $V_{low}$ will be enough to finish the task. The method for the frequency control decision when the system is running under low voltage is similar.
The proposed control strategy is dynamically computed. The online measurement of the computational speed $\omega$ ensures that the control law works for any type of tasks, periodic but also non-periodic, because it is not required to a priori know $\omega$. Moreover, the control algorithm will react in case of perturbation, e.g. if $\omega$ does not behave as expected, or if the computing workload ($\Omega_i$ or $\Delta_i$) is adapted/estimated during the execution of the current task. On the other hand, the control strategy is simple enough to limit the overhead it may introduces. Actually, some simplifications are possible for a practical implementation, like removing the division in (1) (see [10] for further details).

### 3.3 Performance and stability

The performance of the proposed control strategy is guaranteed because the execution of a task always starts with the penalizing high voltage level (by construction of the predictive control law) and the low level will not be applied while the remaining computational workload is important (higher than the maximal possible speed at $V_{low}$). As a result, it is not possible to make better. Furthermore, even if the voltage/frequency levels discretely vary, the speed setpoint to track is always higher or equal than required by construction.
On the other hand, the Lyapunov stability is verified. Lyapunov stability is based on an elementary physical constatation: if the total energy of the system tends to continuously decline, then this system is stable since it is going to an equilibrium state. Let

$$V(t) = \Omega_i - \Omega(t) \tag{3}$$

be a candidate Lyapunov function. This latter expression comes from (1) and refers to the remaining workload in the contractive time horizon of the task. As a result, the Lyapunov function intuitively decreases because the speed of the processor can only be positive, and so is ensured the stability of a task. This is verified for all tasks to be executed.

### 3.4 Estimation of the maximal speeds and robustness

The maximal speeds $\overline{\omega}_{high}$ and $\overline{\omega}_{low}$ cannot be directly calculated since they could vary with temperature or location

(variability), and yet, the control law has to be robust to such uncertainties. For these reasons, the maximal speeds are estimated. A solution consists in measuring the system speed for each power mode and using a weighted average of the measured speed in order to filter the (possible) fluctuations of the measurement. One can refer to [10] for further details. The proposed estimation of the computational speeds leads to a control law without any need of knowledge on the system parameters. This means the strategy is robust to technological uncertainties since it self-adapts whatever the performance of the controlled chip. Robustness deals with the conservation of the stability property when things do not behave as expected. Unexpected behaviors can be of two types: a wrong estimation of the number of instructions to process $\Omega_i$ or the deadline $\Delta_i$, or the presence of process variability.

In the first case, if $\Omega_i$ is overestimated (or $\Delta_i$ is underestimated), the task will be completed before its deadline but the Lyapunov function $V$ in (3) remains decreasing. On the other hand, if $\Omega_i$ is underestimated (or $\Delta_i$ is overestimated), the deadline will be missed. In this case, the remaining instructions are added to the next task workload in order to speed up the system. Another solution could be to use a damping buffer as usually done for video decoding, see [19, 20] for instance. Note that $\Omega_i$ and $\Delta_i$ can also be re-estimated during the execution of the task in order to reduce the error of estimation. Nonetheless, $V$ remains decreasing during the whole task execution.

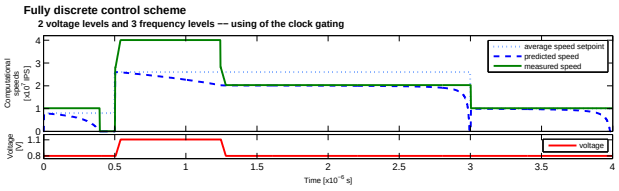
In case of process variability, the real computational speed becomes $\omega_{real}(t) = \alpha\,\omega(t)$, where $\alpha \geq 0$ is the unknown process variability factor, and the system performance is hence affected. $\alpha = 1$ means the real process behaves ideally. Otherwise, the performance of the system is weaker than expected for $\alpha < 1$ and the system is faster than expected for $\alpha > 1$. Nonetheless, $V$ in (3) becomes

$$V(t) = \Omega_i - \alpha\Omega(t) \tag{4}$$

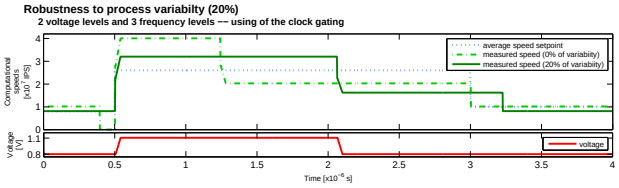
and so the stability is still ensured for all $\alpha > 0$. The convergence rate is reduced for $0 < \alpha < 1$ and increased for $\alpha > 1$. As a result, the system runs a longer (respectively shorter) time under the penalizing high supply voltage (by construction of the control law) to compensate the weaker (resp. better) performance. The only unstable case is for $\alpha = 0$, which means that the processor does not compute at all. In that case, the operating system (or scheduler) must avoid allocating tasks to this part of the chip. Also, note that the tasks can meet their deadline as far as the processor is able to execute the computational workload in the given time (denoted as *feasible* tasks), that is while $\Omega_i/\Delta_i \leq \alpha\,\overline{\omega}_{high}$.

### 3.5 Coarse-grain simulation results

Coarse-grain simulations are performed in *Matlab/Simulink* in order to evaluate the efficiency of the proposed controller. A scenario with three tasks to be executed is run, the number of instructions and deadline for each task being known. The simulation results are presented in Fig. 3. The top plot shows the average speed setpoint of each task (for guideline), the predicted speed (for guideline) and the measured computational speed, while the bottom plot shows the supply voltage. In Fig. 3(a), the system runs during about $80\,\%$ of the simulation time under low voltage. As a result, a reduction of about $30\,\%$ and $65\,\%$ of the energy consumption is achieved (in comparison with a system without DVS and DVFS mechanism respectively).
As the proposed control strategy does not use any knowledge on the system parameters, the controller adapts itself with these uncertainties. Fig. 3(b) shows how the system behaves in case of $20\,\%$ of process variability, that is when the real performance of the circuit are $20\,\%$ less than expected. One could see that the estimation of the maximal computational speed allows the system to still work, even if the processing node does not work as expected. Of course, in order to compensate a lower computational speed induced by the process variability, the system runs a longer time under the penalizing supply voltage.
More simulations and details are presented in [10].




(a) Simulation results with three frequency levels, one for the high voltage level and two for the low voltage level.




(b) Simulation results to test the robustness of the controller with 20% of process variability.

Figure 3: Simulation results of the energy-performance tradeoff control in *Matlab/Simulink*.

## 4. Fine-grain simulation results

In this latter section, fine-grain simulation results are presented for the whole MIPS-R2000 architecture with the

$45\,nm$ CMOS parameter variations. A post layout simulation with *Modelsim* has been performed with a scenario of three tasks, the number of instructions and deadline for each task being known. The simulation results of the system under different process variabilities are shown in Fig. 4. The different available frequencies for each case were summarized in Table 1.

a) The results under *nominal process variability* are depicted in Fig. 4(a). Tasks 1 and 3 are completed successfully using the low voltage and frequency levels. For task 2, the controller speeds up the MIPS R2000 to $V_{high}$ in order to be able to complete the task at the proposed deadline. Once the controller has detected that task 2 can be completed with relaxed conditions (that is after $1.52\,\mu s$), the system switches back to $V_{low}$.

b) The results under *worst process variability* are represented in Fig. 4(b). With reduced performance of the MIPS R2000 (i.e. increased critical path delay), task 2 runs for $3.02\,\mu s$ under $V_{high}$, which is twice longer than processing under nominal process viability.

c) The results when using the *best process variability* configuration are given in Fig. 4(c). The MIPS R2000 is able to successfully complete all the three tasks under $V_{low}$, which adds much more power/energy saving opportunities than under the nominal case. Therefore, our proposal is able to not only compensate for the delay variations with different process variability impacts, but also exploit the enhanced response of the system under best variability conditions to gain more in terms of energy savings.

To evaluate the proposed DVFS control for GALS-ANoC architecture, its average dynamic power, energy consumption, area overhead and robustness to process variability are also analyzed.

### 4.1 Energy and average dynamic power savings

Under nominal process variability, the energy-efficient DVFS control (using dynamic set of clock frequencies) is able to save $21\,\%$ of the energy consumption and $51\,\%$ of the average dynamic power consumed by a system without DVFS, while the intuitive average-based strategy (using fixed set of clock frequencies) only saves $14$ and $37\,\%$ respectively. Therefore, it appears that the energy-efficient control is $1.5$ more power and energy saving efficient than the intuitive control (under nominal process variability).
Our proposal has the ability to adapt the set of clock frequencies with respect to the process variability impact. Thus, the energy-efficient DVFS control was able to exploit the enhanced performance of the system (i.e. reduced critical path delay) to save more energy consumption (i.e. $25\,\%$ under best process variability impact).

Under worst process variability conditions, the used set of clock frequencies for a system without DVFS, and even that for a system with average based DVFS control, violates the MIPS R2000 critical path delay. As a consequence, the MIPS R2000 have erroneous output results. Such a processing node has to be neglected and its allocated tasks have to be reallocated over other high performance processing nodes. However, with the proposed DVFS control architecture, the MIPS R2000 is still able to complete the allocated tasks successfully by using the proper set of maximal clock frequencies. Therefore, this drastically relaxes the fabrication constraints and helps to the yield enhancement.

### 4.2 Robustness to process variability

As already mentioned, the proposed approach is robust to technological uncertainties since the control algorithm does not need any knowledge on the system parameters. Fig. 4 clearly shows how the controller adapts itself with the different (nominal, worst, best) variability corners. The system still works even if the processing node does not work as expected. However, in order to compensate for a weaker (respectively stronger) computational speed induced by the process variability, the system runs a longer (resp. shorter) time under the penalizing supply voltage level. Of course, the energy consumption is impacted in consequence.

### 4.3 Whole energy-efficient control and area overhead

The whole DVFS control system is also evaluated taking into account the consumption of the actuators and the processing element (see Fig. 1). A GALS-ANoC island is compared with a single processing element (i.e. MIPS R2000) and with eight controlled elements.
Under nominal process variability, the average dynamic power and energy saving values of the whole DVFS control system are smaller than but not too far from those presented in previous section ($46$ and $15\,\%$ respectively). Also, the area overhead due to the proposed control approach is about $33\,\%$. On the other hand, a single DVFS control system is needed for all the VFIs in a GALS-ANoC system. Therefore, in a VFI with multiprocessing elements, the efficacy of the DVFS control system is more effective in saving power/energy consumption ($51$ and $20\,\%$ respectively) and, moreover, the area overhead of the extra DVFS hardware is approximately divided by the number of processing elements per a GALS island. For example, the area overhead in a processing island with eight processors is $4.15\,\%$.

## 5. Conclusions

In this paper, a survey of different problems facing designers over the nanometric era was first presented. A GALS-ANoC system was taken as an issue with the application

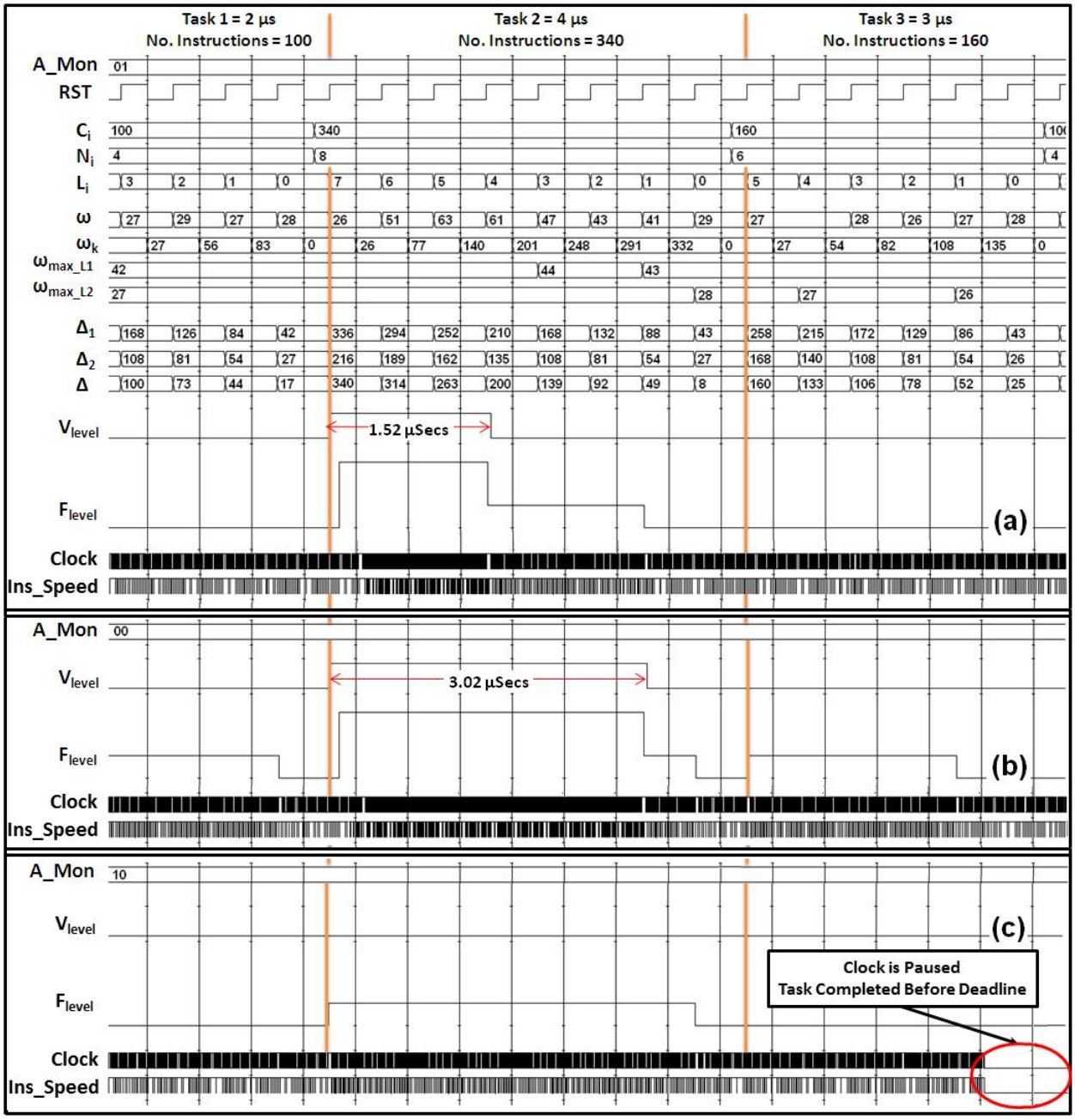



Figure 4: Timing diagram of the digital controller behavior with 3 different MIPS R2000 workloads under different process variability effects.

of DVFS technique. Also, a closed-loop scheme clearly appeared as necessary in such systems and an architecture was proposed in this way. The idea to use integrated sensors embedded in each clock domain, as well as a Programmable Self-Timed Ring (PSTR) oscillator, was presented as one of the promising solutions to reduce the process variability impact. A control algorithm has also been detailed, based on a fast predictive control law. The proposed feedback controller smartly adapts voltages and frequencies (energy-performance tradeoff) with strong technological uncertainties. Stability and robustness were analyzed.

A practical validation was realized in simulation for a MIPS R2000 microprocessor over STMicroelectronics $45\,nm$ technology to obtain results about the power consumption and area overhead of each unit in the power management architecture. Global results for a multicore system were also presented. Through this example, it was hence demonstrated that a dedicated feedback system associated to the

GALS-ANoC paradigm and DVFS techniques, with correct sensors and actuators, is able to achieve better robustness against process variability. This relaxes the fabrication constraints, thanks to an appropriated strategy, and helps to the yields enhancement by applying design techniques.
These preliminary results must now be evaluated on a real test bench. A precise evaluation of the hardware and/or software implementation cost of the proposed control scheme has also to be performed.

**Acknowledgments**

This research has been supported by ARAVIS, a Minalogic project gathering STMicroelectronics with academic partners, namely TIMA and CEA-LETI for micro-electronics, INRIA/LIG for operating system and INRIA/GIPSA-lab for control. The aim of the project is to overcome the barrier of sub-scale technologies (45nm and smaller).

## References

[1] B. F. Romanescu, M. E. Bauer, D. J. Sorin, and S Ozev. Reducing the impact of process variability with prefetching and criticality-based resource allocation. In *Proceedings of the 16th International Conference on Parallel Architecture and Compilation Techniques*, 2007.

[2] L. Fesquet and H. Zakaria. Controlling energy and process variability in system-on-chips: Needs for control theory. In *Proceedings of the 3rd IEEE Multi-conference on Systems and Control - 18th IEEE International Conference on Control Applications*, 2009.

[3] A. P. Chandrakasan and R. W. Brodersen. Minimizing power consumption in digital CMOS circuits. *Proceedings of the IEEE*, 83(4):498–523, 1995.

[4] K. Flautner, D. Flynn, D. Roberts, and D. I. Patel. An energy efficient SoC with dynamic voltage scaling. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, 2004.

[5] A. Varma, B. Ganesh, M. Sen, S. R. Choudhury, L. Srinivasan, and J. Bruce. A control-theoretic approach to dynamic voltage scheduling. In *Proceedings of the International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, 2003.

[6] T. Ishihara and H. Yasuura. Voltage scheduling problem for dynamically variable voltage processors. In *Proceedings of the International Sympsonium on Low Power Electronics and Design*, 1998.

[7] J. Pouwelse, K. Langendoen, and H. Sips. Dynamic voltage scaling on a low-power microprocessor. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, 2001.

[8] P. Choudhary and D. Marculescu. Hardware based frequency/voltage control of voltage frequency island systems. In *Proceedings of the 4th International Conference on Hardware/Software Codesign and System Synthesis*, pages 34 –39, oct. 2006.

[9] D. Marculescu and E. Talpes. Energy awareness and uncertainty in microarchitecture-level design. *IEEE Micro*, 25:64–76, 2005.

[10] S. Durand and N. Marchand. Fully discrete control scheme of the energy-performance tradeoff in embedded electronic devices. In *Proceedings of the 18th World Congress of IFAC*, 2011.

[11] M. Krstic, E. Grass, F. K. Gurkaynak, and P. Vivet. Globally asynchronous, locally synchronous circuits: Overview and outlook. *IEEE Design and Test of Computers*, 24:430–441, 2007.

[12] H. Zakaria and L. Fesquet. Designing a process variability robust energy-efficient control for complex SoCs. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 1:160–172, 2011.

[13] T. Villiger, H. Käslin, F. K. Gürkaynak, S. Oetiker, and W. Fichtner. Self-timed ring for globally-asynchronous locally-synchronous systems. In *Proceedings of the 9th International Symposium on Asynchronous Circuits and Systems*, 2003.

[14] C. Albea Sánchez, C. Canudas de Wit, and F. Gordillo. Control and stability analysis for the vdd-hopping mechanism. In *Proceedings of the IEEE Conference on Control and Applications*, 2009.

[15] J. Hamon, L. Fesquet, B. Miscopein, and M. Renaudin. High-level time-accurate model for the design of self-timed ring oscillators. In *Proceedings of the 14th IEEE International Symposium on Asynchronous Circuits and Systems*, 2008.

[16] O. Elissati, E. Yahya, L. Fesquet, and S. Rieubon. Oscillation period and power consumption in configurable self-timed ring oscillators. In *Joint IEEE North-East Workshop on Circuits and Systems and TAISA Conference*, pages 1 – 4, 2009.

[17] E. Yahya, O. Elissati, H. Zakaria, L. Fesquet, and M. Renaudin. Programmable/stoppable oscillator based on self-timed rings. In *Proceedings of the 15th IEEE International Symposium on Asynchronous Circuits and Systems*, 2009.

[18] M. Alamir. *Stabilization of Nonlinear Systems Using Receding-Horizon Control Schemes: A Parametrized Approach for Fast Systems*, volume 339. Lecture Notes in Control and Information Sciences, Springer-Verlag, 2006.

[19] S. Durand, A.M. Alt, D. Simon, and N. Marchand. Energy-aware feedback control for a H.264 video decoder. *International Journal of Systems Science*, Taylor and Francis online:1–15, 2013.

[20] C. C. Wurst, L. Steffens, W. F. Verhaegh, R. J. Bril, and C. Hentschel. QoS control strategies for high-quality video processing. *Real Time Systems*, 30:7–29, 2005.