Fully Discrete Control Scheme of the Energy-Performance Tradeoff in Embedded Electronic Devices

Sylvain Durand, Nicolas Marchand

Preprints of the 18th IFAC World Congress
Milano (Italy) August 28 - September 2, 2011

Fully Discrete Control Scheme of the Energy-Performance Tradeoff in Embedded Electronic Devices

Sylvain Durand * **Nicolas Marchand** **

* *NeCS team, INRIA - GIPSA-lab - Univ. of Grenoble, Grenoble, France (sylvain.durand@inrialpes.fr).*
** *NeCS and SysCo teams, GIPSA-lab - CNRS - INRIA - Univ. of Grenoble, Grenoble, France (nicolas.marchand@gipsa-lab.inpg.fr)*

Abstract: A voltage scalable device is known to be interesting for energy saving. It enables to reduce the general speed of the device and, therefore, its consumption. We already proposed a fast predictive control strategy to deal with this power-performance tradeoff in an electronic device supplied by two voltage levels and a continuously varying frequency. In this paper, the approach is extended to a fully discrete scheme with M possible voltage levels and N frequency levels. The proposed approach clearly gives an important reduction of the energy consumption with a very low control computational cost. Moreover, the control strategy is highly robust to tackle variability since it is not based on any parameters of the system.

Keywords: Fast predictive control, energy-performance tradeoff, robustness to variability

INTRODUCTION

An energy-performance tradeoff is nowadays one of the key problem in embedded electronic systems. Actually, three power consumption sources exist in CMOS circuits - as explained in Chandrakasan and Brodersen (1995) - which can be sorted into a dynamic consumption, due to electrical gate switching, and a static consumption, induced by short circuit and leakage currents, such as

$$P = P_{switching} + P_{short\ circuit} + P_{leakage}$$
$$P = K_{dyn} f_{clk} V_{dd}^2 + K_{sc} f_{clk} V_{dd} + K_{leak} V_{dd} \qquad (1)$$

where K_{dyn}, K_{sc} and K_{leak} are some parameters fixed by the design of the chip. It appears that the consumption can be reduced decreasing the supply voltage, i.e. V_{dd}, or the clock frequency, i.e. f_{clk}. The voltage is the main control variable since the dynamic consumption is the most important term in (1). In other words, decreasing the voltage (quasi)-quadratically decreases the energy consumption. However, controlling the voltage is a power-delay tradeoff - the power consumption decreases while the delay increases - since the propagation delay seriously increases as the voltage approaches the threshold voltage of the device. Therefore, the frequency has to be decreased first to ensure the maximum delay over the critical path (the longest electrical path a signal can travel to go from a point to another of the circuit). On the other hand, decreasing only the frequency results in a slower running task and the energy consumption finally remains unchanged. As a result, the supply voltage and the clock frequency have to be controlled together, as suggested in Varma et al. (2003). Clearly, it is required to decrease the frequency before decreasing the voltage and, symmetrically, increase the voltage before increasing the frequency. A common approach in embedded systems is using a dynamic voltage and frequency scaling (*DVFS*) mechanism. This method consists in adapting the variables to the computational load and leads up to an important energy consumption reduction in most of applications. Several behaviors are useful for energy saving, notably those in Ishihara and Yasuura (1998). Classically, each task are considered independently and its execution time has to fit with the deadline using an unique supply voltage to minimize the energy consumption. If the chip can only use a small number of discretely voltage levels, the two voltages with the lowest energy consumption are the immediate neighbors of the optimal one. Selecting some of these levels leads to a drastic energy reduction even if the number of levels is very small. At the end, the voltage has to be reduced as much as possible and the frequency adapted to the computational load, as explained in Pouwelse et al. (2001).

Embedded electronic systems have today to be low-power systems but not only. Indeed, with the upcoming nanometric technologies these systems have to face with process variability too, which refers to the unpredictability in manufacturing: uncertainties about how a chip will perform are introduced. Although a circuit is designed to run at a nominal clock frequency, the fabricated implementation may vary far from this expected performance, and can lead to chip failures in certain cases. One could refer to Zakaria et al. (2010) and the reference therein for more details on technological variability issues. Manufacturing yield success is hence hard to achieve and control loops become essential in order to be able to use all the chips whatever their performances. Note that this point is presented in

⋆ This research has been supported by the ARAVIS project, a Minalogic project gathering ST-Microelectonics with academic partners, namely TIMA and CEA-LETI for micro-electronics and INRIA for operating system and control. The aim of the project is to overcome the barrier of subscale technologies (45nm and smaller).

Fesquet and Zakaria (2009). This is why we proposed in Durand and Marchand (2009) a robust strategy to control the energy-performance tradeoff in a voltage scalable device. In this first work, two discrete voltage levels and a continuous frequency range for each level were available. Considering a continuously varying frequency is however not very realistic and the present paper proposes to extend the control strategy for some limited frequency values, that yields N possible frequency levels. Furthermore, another contribution is to now consider M possible voltage levels. Thus, the system architecture is introduced in the next section. Then, the initial control strategy is shortly presented in section 2 before detailing how to handle to a finite number of frequency levels. We also explain why the control law is stable and robust to process variability. Finally the two strategies with continuously and discretely varying frequencies are compared in section 3 in term of energy consumption and control computational needs.

1. SYSTEM ARCHITECTURE

The system architecture is given in Fig. 1.

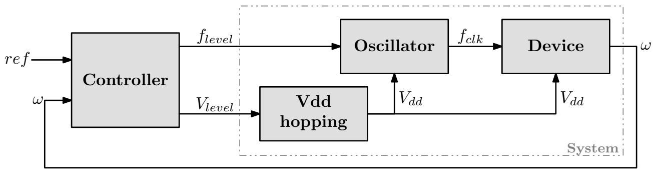


Fig. 1. Controlled system architecture.

The ***Device*** is the electronic system to control (a processor or a system on chip for example). Although it usually runs at nominal supply voltage and constant clock frequency, the proposed architecture allows to dynamically scale these quantities. An energy consumption reduction is then possible introducing a closed-loop controller which monitors the activity of the device - its computational speed ω (in number of instructions by second) - in order to adapt the control variables with respect to a given computational load ref to treat. The model of the device is a linear static function with some unknown parameters

$$\omega = \alpha(V_{dd})f_{clk} + \beta(V_{dd}) \tag{2}$$

where $\alpha(\cdot)$ and $\beta(\cdot)$ can be identified but highly vary with temperature and location on the chip (variability). Nevertheless, the dynamics introduced by the control law will make possible to control the system without any information on these parameters.

The ***Oscillator*** and the ***Vdd-hopping*** are the two actuators used in DVFS in general, which respectively provide the frequency and the voltage to the device:

• A Vdd-hopping mechanism was described in Albea Sánchez et al. (2009), where two voltages V_{low} or V_{high} could supply the chip. In that case, the system simply goes to low voltage when the input signal becomes $V_{level} = V_{level\_low}$, or to high voltage when $V_{level} = V_{level\_high}$ respectively, with a given transition time and dynamics that depend upon an internal control law (one could refer to the reference above for more details). This principle can then be extended to M voltage levels. Thus, the Vdd-hopping provides the voltage V_m when $V_{level} = V_{level\_m}$, with $m \in \{1, 2, \ldots, M\}$ and $V_m > V_{m+1}$. Considering that this inner-loop is extremely fast w.r.t. the loop considered in this paper, one can neglect the dynamics of the Vdd-hopping.

• A ring oscillator is suggested in Yahya et al. (2009). The model is $f_{clk} = \gamma f V_{dd}$, where γ is a constant while the desired frequency f depends on the input signal, i.e. $f = \psi(f_{level})$, using such a look-up table mechanism for instance. Actually, only some limited frequency values are possible, that is $f_{level} = f_{level\_n}$ with $n \in \{1, 2, \ldots, N\}$ and $f_n > f_{n+1}$, and switching from one frequency to another can be considered as instantaneous. Moreover, we choose $N \geq M$ (a discussion on this point follows in section 2).

Afterwards, the device with the two actuators is called the ***System*** whose model can be approximated by an affine function, that is

$$\omega = \sigma f V_{dd} \tag{3}$$

where $\sigma \simeq \alpha\gamma$ since α and β in (2) can be considered as constant (since the voltage range is very small in the present study case) and β can be discarded due to its small impact on the speed.

2. CONTROL STRATEGY

Two aspects have to be taken into account when controlling the energy-performance tradeoff in a voltage scalable device. On a first hand, the controller has to ***i)*** *minimize the energy consumption* in reducing as much as possible the supply voltage and, on another one, ***ii)*** *ensure some good computational performances* fitting the tasks with their deadline. To do that, we propose to dynamically calculate an energy-efficient computational speed setpoint (which minimizes the penalizing high voltage running time) that the system will then have to track. This setpoint is based on some information provided by the operating system for each task T_i to treat, that are the computational load - i.e. the number of instructions C_i - and the deadline N_i. Moreover, let L_i denote the laxity, that is the remaining available time to complete a given task. Note that these parameters can change during the running time of a task (if the operating system decides to update them for instance), this is why they are time-dependant.

The presence of deadline and time horizon to compute tasks naturally leads to predictive control. Predictive control consist in finding an open-loop control profile over some time horizon and in applying it until the next time instant. The control problem is then reconsidered using the new state variables and a new control profile is generated. This finally yields a closed-loop control and the stability relies in the way the open-loop control is chosen. The horizon can be constant, infinite or less classically contractive as in the present paper. The key point is the choice of the open-loop strategy and its computational cost. Indeed, if predictive control is known to be a robust approach, it is also often associated to high computational cost which is not acceptable in the present case. Whereas the classical strategy consists in minimizing some cost functions, the strategy adopted here is called fast predictive control and consists in taking advantage of the structure of the dynamical system to fasten the finding of the open-loop control. This is explained in Alamir (2006). The simplicity of system (3) considered here is therefore very suitable for such strategies. The predictive strategy of the present paper is intuitively explained next and its formal expression is given in subsection 2.1.

In order to simplify the understanding, the behavior is reduced to only two voltage levels, i.e. V_{high} and V_{low}, with a continuously varying frequency range for each level. This case is detailed in Durand and Marchand (2009). Afterwards, let ω^{max} denote the maximum computational speed when the system is running at high voltage, that is $\omega^{max} = \sigma F_{V_{high}max} V_{high}$ from (3), where $F_{V_{high}max}$ is the maximum frequency in the available range at V_{high}. Respectively, let ω_{max} denote the maximum possible speed at low voltage, that is $\omega_{max} = \sigma F_{V_{low}max} V_{low}$, where $F_{V_{low}max}$ is the maximum frequency at V_{low}. The high voltage level will hence be necessary as soon as the average speed setpoint of a task is higher than ω_{max} in order to not miss the deadline. An intuitive method consists in building the average speed setpoint of each task - that is the ratio C_i/N_i - in such a way that the number of instructions to do is performed at the end of the task. This is depicted in Fig. 2 (left). However, this method is not energy-efficient since a whole task can be computed with the penalizing high supply voltage, such as for task T_2. Nevertheless, a suggested solution is to split the tasks into two parts. This is represented in Fig. 2 (right). Firstly, the chip begins to run at high voltage - if required - with the maximum available frequency in order to achieve the maximum possible speed ω^{max} to go faster, such as for T_2 from t_2 to t_{switch}. Then, the task could be finished at low voltage - which, consequently, highly reduces the energy consumption - with a speed lower than ω_{max}. 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 ensure some good computational performances. However, it is not *a priori* known and, therefore, a predictive control law has to be used to dynamically calculate the switching time.

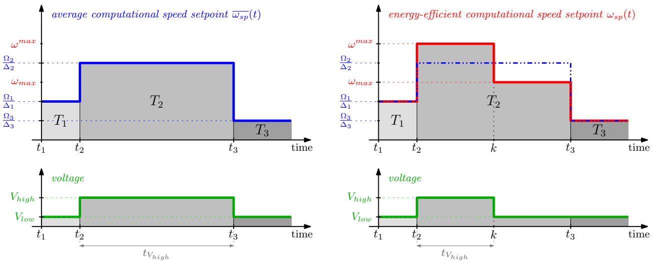


Fig. 2. Different setpoint buildings: the intuitive average speed setpoint vs. a more energy-efficient one.

Whereas the main idea was introduced, in fact considering a continuously varying frequency is not realistic and we then propose to extend the principle to a fully discrete control scheme. Therefore, M voltage and N frequency levels are considered straight afterwards. The lower is the supply voltage the better will be reduced the energy consumption - since the supply voltage is the penalizing parameter - and the system has hence to run at the maximum possible computational speed for all the voltage levels except the lowest one. For this reason, we propose to have only one possible frequency f_m per voltage level, that is the maximum available frequency when the chip is running at V_m by definition. Let ω_m denote the (maximum) possible speed at V_m (this value is implicitly maximum since f_m is the maximum value in the available frequency range), that is $\omega_m = \sigma f_m V_m$ from (3). Thus, for instance, when the system runs with the supply voltage V_2 and the clock frequency f_2, the corresponding computational speed is ω_2. As regards the lowest voltage level V_M, we propose to have several possible frequency levels because, as the energy consumption could not be reduced anymore - since no lower voltage level exists - the degree of freedom on the frequency will allow to fit the task with its deadline (as much as this is possible). This is why we defined $N \geq M$ in section 1. Therefore, the electronic device could run with the different clock frequencies $f = \{f_M, f_{M+1}, \ldots, f_N\}$, which lead to the computational speeds $\omega = \{\omega_M, \omega_{M+1}, \ldots, \omega_N\}$ respectively. Afterwards, we also note $V_x = V_M \quad \forall\, M \leq x \leq N$ in order to simplify the next equations. Eventually, the control strategy principle remains the same as in the two-voltage level case since, for each task, the two computational speeds which are immediate neighbors to the average speed will minimize the energy consumption (this was presented in introduction). For each task to treat, the controller has hence to deduce these two neighbors speeds, denoted ω_j and ω_{j+1} with $\omega_j > \omega_{j+1}$, in order to execute the task running firstly at V_j and then at V_{j+1}. Anyway, the controller still has to dynamically predict the switching time to go from V_j to V_{j+1} in order to minimize the penalizing V_j running time while guaranteeing that the task will not miss its deadline.

2.1 Fast predictive control

Actually, the predictive issue can be formulated as an optimization problem. For each task T_i to treat, what is the computational speed setpoint which minimizes the high voltage running time t_{V_j} - when the two immediate neighbors are V_j and V_{j+1} - while guaranteeing that the executed instruction number is equal to the number of instructions to do at the end of the task. This is

$$\min\ t_{V_j} \qquad \text{s.t.} \quad \int_{N_i(t)} \omega(t)\ dt = C_i(t)$$

where $\int \omega dt$ corresponds to the executed number of instructions for the current task. This optimal criteria allows to solve the predictive problem but is too complex to be implemented in an electronic chip with low resources, as in the present case. Nevertheless, the closed-loop solution yields an easier and faster algorithm since one simply needs to know ***i)*** *the computational load to treat* and ***ii)*** *how much time is available to do it.* The remaining time before the end of the task is hence necessary, this is why the laxity L_i will be used next instead of the deadline N_i. Eventually, the speed required to fit the task with its deadline regarding what it has already been executed - afterwards denoted the *predicted speed* δ - is dynamically calculated at each sampling instant as follows

$$\delta(t_{k+1}) = \frac{C_i(t_k) - \sum_{t_i}^{t_k - t_i} \omega(t_k)}{L_i(t_k)} \tag{4}$$

where t_i is the beginning of the task T_i, t_k and t_{k+1} are the current and next sampling time respectively. The implementation of the previous equation then becomes

$$\begin{aligned}\Omega(t_k) &= \Omega(t_{k-1}) + T_s\omega(t_k)\\ \delta(t_{k+1}) &= \frac{C_i(t_k) - \Omega(t_k)}{L_i(t_k)}\end{aligned} \tag{5}$$

where Ω is the integration of the computational speed ω, T_s is the sampling period and t_{k-1} is the last sampling time. Furthermore, a conditional instruction is added to be coherent with (4). Indeed, as the computational speed is integrated on the running time of each task, the variable Ω has to be reset when a task is executed, which means in the last sampling time before its deadline. More precisely, it is not set to zero to prevent the case when the task is not completely executed at its deadline but, in fact, it is adjusted with the difference between what it has already been done and what it was required to do, such as

$$\Omega(t_k) = \Omega(t_k) - C_i(t_k) \quad \text{if} \quad L_i(t_k) \leq T_s$$

The energy-efficient speed setpoint ω_{sp} is then directly deduced from the value of δ (and so are the voltage and frequency levels). First of all, note that $\omega_j > \delta \geq \omega_{j+1}$ by construction since we assumed the two immediate neighbors are V_j and V_{j+1}. Applying the proposed algorithm, the device firstly runs with the more penalizing speed ω_j. As a result, the value of δ dynamically decreases - if the system correctly tracks the setpoint - until achieving ω_{j+1}, which means that the task can then be finished with this less penalizing speed. In fact the computational speed setpoint is not really required since the control variables are easily deduced, but we notice ω_{sp} anyway (for a well understanding) in the control decisions, that are

$$\left|\begin{array}{ll}\omega_{sp}(t_{k+1}) &= \omega_j\\ V_{level}(t_{k+1}) &= V_{level\_j}\\ f_{level}(t_{k+1}) &= f_{level\_j}\end{array}\right. \quad \text{if} \quad \delta(t_{k+1}) > \omega_{j+1}$$
$$\left|\begin{array}{ll}\omega_{sp}(t_{k+1}) &= \omega_{j+1}\\ V_{level}(t_{k+1}) &= V_{level\_j+1}\\ f_{level}(t_{k+1}) &= f_{level\_j+1}\end{array}\right. \quad \text{otherwise} \tag{6}$$

Furthermore, we previously assumed that $\omega_j > \delta \geq \omega_{j+1}$ but, in practice, δ can go out these bounds (for instance if the running computational speed goes faster/slower than expected). Nevertheless, the algorithm (6) still works with a problem now translated to $\omega_{j+1} > \delta \geq \omega_{j+2}$.

Eventually, the performances are guaranteed because the execution of a task always starts with the more penalizing voltage level - by construction of the predictive control law - and a lower level will not be applied while the remaining computational load is important. As a result, it is not possible to make better. Furthermore, the speed setpoint to track is always higher or equal than required. The 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 $V = x^T Px$ be a candidate Lyapunov function, with P positive definite and $x(t_k) = C_i(t_k) - \sum_{t_i}^{t_k-t_i}\omega(t_k)$. This latter expression comes from (4), where x refers to the remaining load in the contractive time horizon of the task. Therefore, the Lyapunov function intuitively decreases - because the speed of the electronic device can only be positive - and so is ensured the stability of a task.

On top of the proposed strategy, a last control decision is also possible "deactivating" the clock of the device. This is called the clock-gating principle. One could refer to Kuzmicz et al. (2007). In this case, the device runs with the lowest voltage V_M and a null frequency (in fact the clock is only paused but a null frequency is used in simulation to highlight the clock-gating intervals). This behavior is useful - especially when the number of voltage and frequency levels is poor - because a higher voltage level than required is used most of the time by construction, which leads to consuming idle intervals. Consequently, it could be interesting to deactivate the clock until the beginning of the following task. However, in order to minimize the using of the clock-gating principle we propose to pause the clock only if the beginning of the following task is not too close, that is when $L_i(t_k) > L_{min}$, where L_{min} is a tunable parameter.

2.2 Estimation of the computational speeds

The available speeds ω_m might be obtained from the system model (3), i.e. $\omega_m = \sigma f_m V_m$, where σ is inherent to the device but could vary with temperature or location (variability), and yet, the control has to be robust to such an uncertainty. Furthermore, the value of σ, f_m and V_m are not known. For these reasons, we propose to estimate ω_m. Let $\tilde{\omega}_m$ denote the estimated speeds. A solution consists in measuring the system speed for each couple voltage/frequency levels. Therefore, the speeds ω_m are measured when the supply voltage is V_m and the clock frequency f_m. Moreover, we propose to use a weighted average of the measured speed in order to filter the (possible) fluctuations of the measurement, which yields

$$\text{if} \quad \left\{\begin{array}{ll} V_{level}(t_{k-1}) &= V_{level\_m}\\ f_{level}(t_{k-1}) &= f_{level\_m}\end{array}\right.$$
$$\tilde{\omega}_m(t_k) = (1-\rho)\tilde{\omega}_m(t_{k-1}) + \rho\omega(t_k) \tag{7}$$

where $0 \leq \rho \leq 1$ is the weighted value. Note that a problem could appear during the voltage transitions. Indeed, the algorithm (6) allows to dynamically calculate the predicted speed δ and compare its value with the computational speeds ω_m (in fact with the estimation of the speeds $\tilde{\omega}_m$). The controller thus changes the voltage and frequency levels as soon as δ crosses a possible speed. However, during this level transition the estimated speed could vary (due to the fluctuations in the estimation) and, because of this phenomena, the levels could switch and switch again. A solution is hence required. For this reason, we propose to bound the value of ρ in such a way that the variation of the estimation is always lower than the variation of δ. First, let $\Delta\tilde{\omega}_m$ denote the variation of the computational speed estimation, obtained from (7), that is

$$\begin{aligned}\Delta\tilde{\omega}_m(t_k) &= \frac{\tilde{\omega}_m(t_k) - \tilde{\omega}_m(t_{k-1})}{T_s}\\ &= \frac{\rho}{T_s}\left[\omega(t_k) - \tilde{\omega}_m(t_{k-1})\right]\end{aligned}$$

Then, let $\Delta\delta$ denote the variation of the predicted speed, calculated from (5), that is

$$\begin{aligned}\delta(t_{k+1}) &= \frac{C_i(t_k) - \Omega(t_k)}{L_i(t_k)}\\ &= \frac{C_i(t_k) - \Omega(t_{k-1})}{L_i(t_k)} - \frac{T_s\omega(t_k)}{L_i(t_k)}\end{aligned}$$

$$\delta(t_k) = \frac{C_i(t_{k-1}) - \Omega(t_{k-1})}{L_i(t_{k-1})} \simeq \frac{C_i(t_k) - \Omega(t_{k-1})}{L_i(t_k)}$$

$$\Delta\delta(t_k) = \frac{\delta(t_{k+1}) - \delta(t_k)}{T_s} \simeq -\frac{\omega(t_k)}{L_i(t_k)}$$

The approximation comes from the fact that the instruction number usually does not change for a given task and the laxity is only different from one sampling period T_s between two measurements, which can be neglected. Finally, the variation of the estimation has to be lower than the variation of the predicted speed, such as

$$\Delta\tilde{\omega}_m(t_k) \leq \Delta\delta(t_k)$$
$$\Leftrightarrow 0 \leq \rho \leq -\frac{T_s\omega(t_k)}{L_i(t_k)\left[\omega(t_k) - \tilde{\omega}_m(t_{k-1})\right]}$$

This result comes from ***i)*** $\rho \geq 0$ by construction and, moreover, ***ii)*** we consider that a problem could only appear during a decreasing switching, that is when $\omega(t_k) \geq \tilde{\omega}_m(t_{k-1})$. Therefore, the parameter ρ is bounded and this has to be considered next in the implementation.

As explained in subsection 2.1, the stability is ensured and, therefore, a task will meet its deadline. Furthermore, the proposed estimation of the computational speeds ω_m - which leads to a control law without any information on the system parameters - yields a robust strategy which will self-adapt whenever the performance of the controlled chip. This is very important for process variability.

3. PERFORMANCE EVALUATION

A scenario with three tasks to execute is proposed for simulations: the first task starts with 4 instructions to do in $0.5\mu s$, then a 65 instruction task has to be executed in $2.5\mu s$ and the last one has to compute 10 instructions in $1\mu s$. These data are represented in Fig. 3.

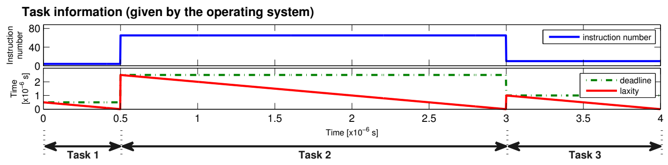
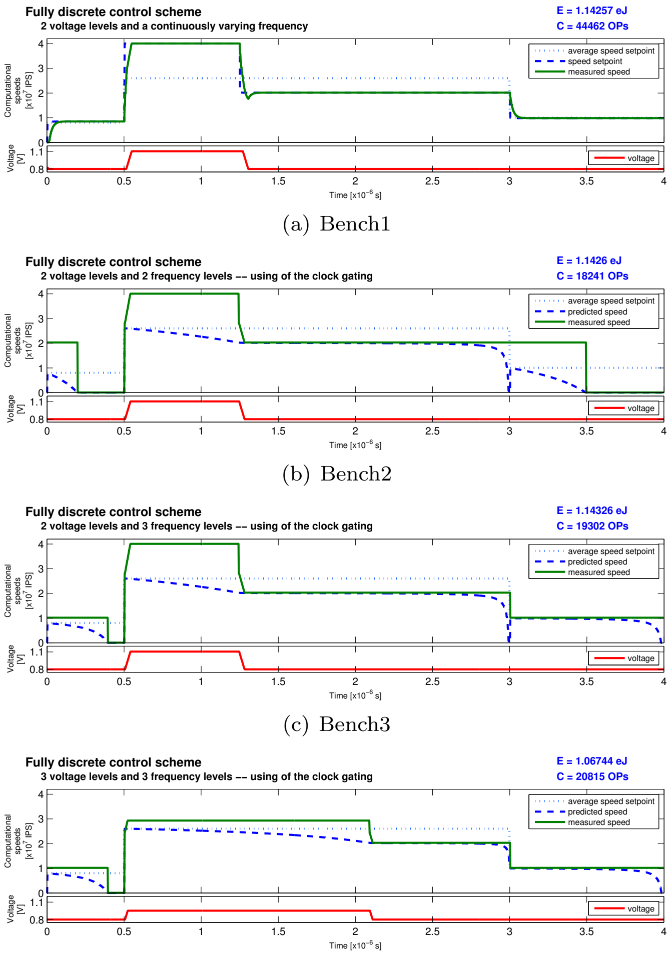


Fig. 3. References used for the simulation: the number of instructions C_i, the deadline N_i and the laxity L_i.

Several benchmark tests are then proposed for simulations, with different values of voltage and frequency levels:

Bench1 : 2 voltage levels and a continuously varying frequency, shown in Fig. 4(a),
Bench2 : 2 voltage levels and 2 frequency levels with the clock-gating principle, Fig. 4(b),
Bench3 : 2 volt. 3 freq. levels and clock-gating, Fig. 4(c),
Bench4 : 3 volt. 3 freq. levels and clock-gating, Fig. 4(d).

The top plot shows the average speed setpoint C_i/N_i (for guideline), the predicted speed δ (for guideline) and the measured speed ω, whereas the bottom one shows the supply voltage V_{dd}. Note that the clock frequency f_{clk}, the frequency level f_{level} and the voltage level V_{level} can be deduced from them. Furthermore, the results are quantified in term of energy consumption and computational cost. The power consumption comes from (1) where a



(a) Bench1

(b) Bench2

(c) Bench3

(d) Bench4

Fig. 4. Simulation results of the fully discrete scheme.

ratio is also added due to the Vdd-hopping, that is 20 % more during the voltage transition and 3 % more during the steady-states intervals, such as suggested in Miermont et al. (2007). An integration during the whole running time finally gives the total energy consumption of the system (afterwards denoted E in equivalent-joules eJ). The control computational cost is obtained with the Lightspeed Matlab toolbox proposed in Minka (2009), which provides a number of operations (afterwards denoted C in OPs). The cost is different for an addition, a multiplication (twice the cost of an addition) or a division (the most consuming which is eight times the cost of an addition) for instance. Finally, the different strategies are compared with a system without DVFS and DVS mechanism, where the supply voltage is fixed to the most penalizing level, i.e. $V_{dd} = V_1$, while the clock frequency is fixed to $f_{clk} = f_1$ in the first case and is continuously varying in the second one.

With two voltage levels - Fig. 4(a) to (c) - the system runs during almost 80 % of the simulation time at low voltage and a reduction of the energy consumption of about 30 % is achieved compared to a system without DVS and 65 % compared to a system without DVFS mechanism. Furthermore, the fully discrete control scheme - Fig. 4(b) to (c) - requires a lower computational cost than in the continuous frequency case because, for almost the same energy consumption reduction, the control computational cost is divided by more than two. This is because the control variables are directly deduced from the predictive control law (without requiring to calculate a speed setpoint and applying a setpoint tracking). Note that the predicted speed is decreasing all the time in the fully discrete control scheme because the levels are always higher (or equal) than required - by construction - due to the limited number of frequency values. On the other hand, with three voltage levels - Fig. 4(d) - the system does not need to go to the highest level to treat the tasks of the present bench, but it runs a larger time (during about 60% of the simulation time) at the middle voltage level to compensate. This leads to a reduction of the energy consumption of about 10% anyway whereas the paying tradeoff is an increase of the control computational cost (about 10 % more) due to the extra voltage level to control. Eventually, comparing Fig. 4(b) and (c) for the number of frequency levels, or Fig. 4(c) and (d) for the number of voltage levels, shows that in both cases the control computational cost increases - due to the extra levels to manage - without reaching a clearly better energy saving. Nevertheless, the number of levels is important but it is important to notice that, in practice, designing a circuit with several voltage levels is more complex and less area-efficient than adding some possible frequency levels in the ring oscillator. For this reason, one would prefer to have a small number of voltage levels - two seem to be enough - and choose the number of frequency levels regarding the expected performances.

Furthermore, the controller is highly robust to process variability. This phenomena can be modeled as an unknown gain in the equation of the chip (2), which becomes $\omega = \lambda(\alpha f_{clk} + \beta)$. Let $0 \leq \lambda \leq 1$ denote this gain. Note that $\lambda = 1$ is a chip without any uncertainties, the measured speed ω is reduced when $\lambda < 1$ and so are the maximum speeds ω_m and, finally, the chip does not work at all if $\lambda = 0$. The simulation results in Fig. 5 show how the system still works with 20 % of variability, i.e. $\lambda = 0.8$. Of course, the system runs a longer time at the penalizing supply voltage in order to compensate a weak performance, but the tasks meet their deadline anyway. The estimation of the maximum computational speeds allows this robustness.

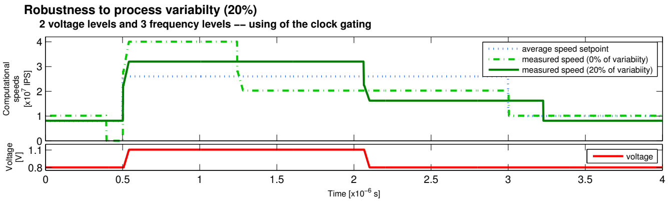


Fig. 5. Simulation results to test the robustness of the controller with 20% of process variability.

CONCLUSIONS AND FUTURE WORKS

This paper proposes a discrete architecture to control the energy-performance tradeoff in an (embedded) electronic device. A fast predictive control technique - based on a previous work in Durand and Marchand (2009) - allows to minimize the energy consumption while guaranteeing some good computational performances. The proposal yields important energy saving with a low control computational cost. Furthermore, it is highly robust in the case of large dispersion phenomena, like the one arising in chip in 45 nm and smaller technologies.

The next step is to test the proposed control strategy in a real nanometric chip.

REFERENCES

Alamir, M. (2006). *Stabilization of Nonlinear Systems Using Receding-Horizon Control Schemes: A Parametrized Approach for Fast Systems.* Lecture Notes in Control and Information Sciences. Springer-Verlag, London.

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

Chandrakasan, A. and Brodersen, R. (1995). Minimizing power consumption in digital cmos circuits. In *Proceedings of the IEEE*, volume 83, 498–523.

Durand, S. and Marchand, N. (2009). Fast predictive control of micro controller's energy-performance tradeoff. In *Proceedings of the 3rd IEEE Multi-conference on Systems and Control - 18th IEEE International Conference on Control Applications.*

Fesquet, L. and Zakaria, H. (2009). 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.*

Ishihara, T. and Yasuura, H. (1998). Voltage scheduling problem for dynamically variable voltage processors. In *Proceedings of the International Sympsonium on Low Power Electronics and Design*, 197–202.

Kuzmicz, W., Piwowarska, E., Pfitzner, A., and Kasprowicz, D. (2007). Static power consumption in nano-cmos circuits: Physics and modelling. In *Proceeding of the 14th International Conference Mixed Design of Integrated Circuits and Systems.*

Miermont, S., Vivet, P., and Renaudin, M. (2007). A power supply selector for energy- and area -efficient local dynamic voltage scaling. In *PATMOS'07: 17th International Workshop on Power and Timing Modeling, Optimization and Simulation*, 556–565.

Minka, T. (2009). The lightspeed matlab toolbox v2.2. Http://research.microsoft.com/en-us/um/people/minka/software/lightspeed/.

Pouwelse, J., Langendoen, K., and Sips, H. (2001). Dynamic voltage scaling on a low-power microprocessor. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, 251–259.

Varma, A., Ganesh, B., Sen, M., Choudhury, S., Srinivasan, L., and Bruce, J. (2003). A control-theoretic approach to dynamic voltage scheduling. In *Proceedings of the International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, 255–266.

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

Zakaria, H., Durand, S., Fesquet, L., and Marchand, N. (2010). Integrated asynchronous regulation for nanometric technologies. In *VARI 2010: first European workshops on CMOS Variability.*