An Approach for Reducing Dynamic Power Consumption in Synchronous Sequential Digital Designs

Noureddine Chabini
Department of Electrical and Computer Engineering
Royal Military College of Canada
P.B. 17000, Station Forces, Kingston, ON, Canada, K7K 7B4
Email: chabini-n@rmc.ca

Wayne Wolf
Department of Electrical Engineering
Princeton University
Engineering Quadrangle, Olden Street, Princeton, NJ, USA, 08544
Email: wolf@ee.princeton.edu

Abstract— The problem of minimizing dynamic power consumption by scaling down the supply voltage of computational elements off critical paths is widely addressed in the literature for the case of combinational designs. The problem is NP-hard in general. To address this problem in the case of synchronous sequential digital designs, one needs to move some registers while applying voltage scaling. Moving these registers shifts some computational elements from critical paths, and can be done by basic retiming. Integrating basic retiming and voltage scaling to address this NP-hard problem cannot in general be done in polynomial run-time. In this paper, we propose to first apply a guided retiming and then to apply supply voltage scaling on the retimed design. We devise new polynomial time algorithms to realize this guided retiming, and the supply voltage scaling on the retimed design. Experimental results on known benchmarks have shown that the proposed approach can reduce dynamic power consumption by factors as high as 61% for single-phase designs with minimal clock period. Also, they have shown that it can solve optimally the problem, and produce converter-free designs with reduced dynamic power consumption. For large size circuits from ISCAS'89 benchmark suite, the proposed algorithms run in 15s-1h.

I. INTRODUCTION

In CMOS technology, the average power consumption in a digital design is defined as follows [7]:

$$P_{av} = P_{dyn} + P_{short} + P_{leakage} + P_{static}, \quad (1)$$

where P_{dyn}, P_{short}, $P_{leakage}$, and P_{static} stand, respectively, for dynamic power, short-circuit power, leakage power, and static power.

Dynamic power is the dominant component in Equation (1). It is a quadratic function of the supply voltage V_{dd} [7]:

$$P_{dyn} = KCfV_{dd}^2, \quad (2)$$

where *K* is the switching activity factor, C is the loading capacitance, and f is the clock frequency.

The short-circuit power accounts for a small portion of the total power consumed described by Equation (1). It is approximately defined by the following equation [7]:

$$P_{short} = \lambda f(V_{dd} - 2V_{th})^3, \quad (3)$$

where λ is a constant and V_{th} is the threshold voltage.

Basic retiming has been proposed in [3] as an optimization technique for synchronous sequential digital designs. This technique changes the location of registers in the design in order to achieve one of the following goals: (*i*) minimizing the clock period, (*ii*) minimizing the number of registers, or (*iii*) minimizing the number of registers for a target clock period.

Minimizing dynamic power for synchronous sequential digital designs is addressed. The paper [6] presented heuristics to minimize the switching activity. The approach in [6] is based on the fact that registers have to be positioned on the output of computational elements of high switching activity, since the output of a register switches only at the arrival of the clock signal compared to a computational element that may switch many times during the clock period.

In [5], fixed-phase retiming is proposed. The edge-triggered circuit is first transformed to a two-phase level-clocked circuit, by replacing each edge-triggered flip-flop by two latches. Next, the latches of one phase are kept fixed, while the latches belonging to the other phase are moved onto wires with high switching activity and loading capacitance.

Due to the quadratic term in Equation (2), the dynamic power may significantly be reduced by scaling down the supply voltage of some computational elements. The negative effect of scaling down the supply voltage of a computational element is an increase of the execution delay of that element.

For a fixed threshold and based on Equation (3), scaling down the supply voltage of some computational elements may also reduce the short-circuit power consumption.

The problem of Minimizing Dynamic Power Consumption (*MDPC*) by scaling down the supply voltage of computation elements off critical paths has been widely addressed in the case of combinational digital designs. For this kind of designs, interested reader is referred to [2][8][9][12][13] for a literature review on approaches to the problem. Digital systems with multiple supply voltages are designed [10][11], which demonstrates the feasibility of using multiple supply voltages to reduce power consumption.

In this paper, we address the *MDPC* problem for synchronous sequential digital designs. Since critical paths are related to the position of registers in the designs, our aim is not just to scale down the supply voltage of computation elements off critical paths, but also to move registers from their positions in order to maximize the number of computation elements off critical paths, leading to a minimum dynamic power consumption. Registers have to be moved by retiming [3]. Instead of unifying retiming and supply voltages scaling, we propose to apply a *guided retiming* followed by the application of voltage scaling on the retimed design. Unifying retiming and supply voltage scaling to minimize dynamic power consumption cannot be done in general by a polynomial time algorithm, since the problem is NP-hard even for the case of combinational designs [2].

We provide new polynomial time algorithms to realize the *guided retiming* as well as the supply voltage scaling on the retimed design. The algorithms are based on linear programs. Experimental results have shown that this approach can reduce dynamic power consumption by factors as high as 61%. We compare our approach with the approach in [1], which unifies retiming and supply voltage scaling to address the *MDPC* problem using a Mixed Integer Linear Program (*MILP*). Based on the obtained numerical results, our approach can reduce dynamic power consumption by factors equal or very close to the reduction factors obtained by [1]. Also, it is based on polynomial time algorithms compared to [1] that uses an *MILP* whose worst case execution time is exponential for the case of the *MDPC* problem.

II. PRELIMINARIES

A. Design Representation

We represent a synchronous sequential digital design (as in [3]) by a directed cyclic graph $G = (V, E, d, w)$, where V is the set of computational elements in the design, and E is the set of edges that represent interconnections between vertices. Each vertex v in V has a non-negative integer execution delay $d(v) \in N$. Each edge $e_{u, v}$, from node u to node v, in E is weighted with a register count $w(e_{u, v}) \in N$, representing the number of registers on the wire between u and v.

Fig. 1 presents a directed cyclic graph model of a synchronous sequential circuit. The execution delay of each computational element of this circuit is specified as a label on the left of each node. Symbol into each node is a label of that node.

Since a combinational digital design does not have registers on the wires, it can then be modeled as an acyclic graph $G = (V, E, d)$, where V, E and d are defined as above.

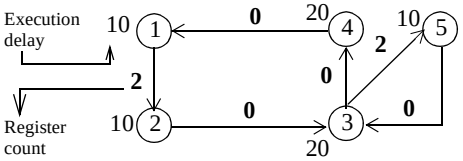

Fig. 1 : Directed cyclic graph model.

B- Basic Retiming

Let $G = (V, E, d, w)$ be a synchronous sequential digital design. Basic retiming (or retiming for short in this paper) r [3] is defined as a function $r : V \rightarrow Z$, which transforms G to a functionally equivalent synchronous sequential digital design $G_r = (V, E, d, w_r)$. The set Z represents natural integers.

The weight of each edge $e_{u, v}$ in G_r is defined as follows:

$$w_r(e_{u, v}) = w(e_{u, v}) + r(v) - r(u), \forall e_{u, v} \in E. \quad (4)$$

Since the weight of each edge in G_r represents the number of registers on that edge, then we must have:

$$w_r(e_{u, v}) \geq 0, \forall e_{u, v} \in E. \quad (5)$$

Any retiming r that satisfies Equation (5) is called a valid retiming. From expressions (4) and (5) one can deduce the following inequality:

$$r(u) - r(v) \leq w(e_{u, v}), \forall e_{u, v} \in E. \quad (6)$$

Let $P(u, v)$ be a path from node u to node v in V. Equation (4) implies that for every two nodes u and v in V, the change in the register count along any path $P(u, v)$ depends only on its two endpoints:

$$w_r(P(u, v)) = w(P(u, v)) + r(v) - r(u), \forall u, v \in V, \quad (7)$$

where:

$$w(P(u, v)) = \sum_{e_{x, y} \in P(u, v)} w(e_{x, y}). \quad (8)$$

Let $d(P(u, v))$ be the delay of a path $P(u, v)$ from node u to node v. $d(P(u, v))$ is the sum of the execution delays of all the computational elements that belong to $P(u, v)$.

A 0-weight path is a path such that $w(P(u, v)) = 0$. The minimal clock period of a synchronous sequential digital design is the longest 0-weight path. It is defined by the following equation:

$$\Pi = Max_{\forall u, v \in V}\{d(P(u, v)) \mid (w(P(u, v)) = 0)\}. \quad (9)$$

Two matrices called *W* and *D* are used in retiming algorithms. They are defined as follows [3]:

$$W(u, v) = Min\{(w(P(u, v)))\}, \forall u, v \in V. \quad (10)$$

$$D(u, v) = Max\{d(P(u, v)) \mid w(P(u, v)) = W(u, v)\}, \forall u, v \in V. \quad (11)$$

The matrices *W* and *D* can be computed as explained in [3].

One application of retiming is to minimize the clock period of synchronous sequential digital designs. For instance, for Fig. 1, the clock period is $\Pi = 60$, which is equal to $d(v_2) + d(v_3) + d(v_4) + d(v_1)$. However, we can obtain $\Pi = 30$ if we apply the following retiming vector $\{1, 0, 0, 1, 0\}$ to the vector of nodes $\{1, 2, 3, 4, 5\}$ in G. The retimed graph G_r is presented by Fig. 2 (a).

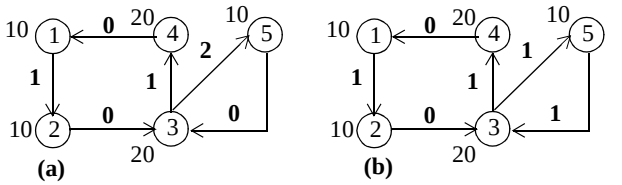

Fig. 2 : Retimed graphs with minimal clock period $\Pi = 30$.

For the purpose of this paper, we extract from [3] the following two theorems **(**Theorems 1 and 2**),** which are also proved in [3].

Theorem 1 : *Let* $G = (V, E, d, w)$ *be a synchronous digital design, and let* Π *be a positive real number. Then there is a retiming r of G such that the clock period of the resulting retimed design* G_r *is less than or equal to* Π *if and only if there exists an assignment of integer value r(v) to each node v in V such that the following conditions are satisfied: (1)* $r(u) - r(v) \leq w(e_{u, v}), \forall e_{u, v} \in E$, *and (2)* $r(u) - r(v) \leq W(u, v) - 1, \forall u, v \in V$ *such that* $D(u, v) > \Pi$. ❑

Theorem 2 : *Let* $G = (V, E, d, w)$ *be a synchronous circuit, and let* Π *be a positive real number. Then the clock period of G is less than or equal to* Π *if and only if there exists a function* $s : V \rightarrow [0, \Pi]$ *such that* $-s(v) \leq -d(v), \forall v \in V$ *and such that* $s(u) - s(v) \leq -d(v), \forall e_{u, v} \in E$ and $w(e_{u, v}) = 0$. ❑

III. IMPORTANCE OF UNIFYING RETIMING AND SUPPLY VOLTAGE SCALING

Let us use the design in Fig. 1. Suppose that we have two supply voltages 5V and 3V. Also, assume that if one uses the supply voltage 5V, the computational elements $\{1, 2, 3, 4, 5\}$ will have, respectively, the execution delays $\{10, 10, 20, 20, 10\}$, and will consume, respectively, the dynamic powers $\{26, 26, 51, 26, 26\}$. And that if the supply voltage 3V is used, the execution delays and the dynamic power consumptions for these computational elements will be, respectively, $\{24, 12, 24, 24, 12\}$ and $\{21, 21, 42, 21, 13\}$.

Our objective is to minimize the dynamic power consumption for designs operating with a target clock period. Assume that we want a single-phase clocked design with a minimal clock period. For the design in Fig. 1, which operates with supply voltage 5V, the minimal clock period is $\Pi = 30$, which is obtained as discussed in Section B. A possible retimed design with clock period $\Pi = 30$ is presented in Fig. 2 (a). An other possible retimed design with the same clock period $\Pi = 30$ is presented in Fig. 2 (b).

We have now two possible designs both operate at $\Pi = 30$. But, which one of them consumes less dynamic power? Let us now compute reduction factors of dynamic power consumption for designs in Fig. 2, using the two supply voltages above and $\Pi = 30$.

In Fig. 2 (a), we have three critical paths $P(2, 3)$, $P(4, 1)$ and $P(5, 3)$. Consequently, no computational element can operate with supply voltage 3V. Hence, we cannot reduce the dynamic power consumption for that design. For Fig. 2 (b), only the paths $P(2, 3)$ and $P(4, 1)$ are critical. Consequently, computational elements 1, 2, 3 and 4 must be powered by 5 V, and computational element 5 can operate with supply voltage 3V. In this case, dynamic power consumption is reduced by 8.38%.

To minimize, by supply voltage scaling, the dynamic power consumption for synchronous sequential designs, it is then clear that one needs to simultaneously apply retiming and supply voltage scaling.

IV. Problem Definition and a Heuristic Approach for its Resolution

We are given a synchronous sequential digital design $G = (V, E, d, w)$ that operates with a given supply voltage, which is called here highest supply voltage. The design is assumed to operate at a clock period, Π, that is given by the designer or determined by applying a retiming for clock period minimization on G. We are also given multiple supply voltages and every delay as well as dynamic power consumption of each computational element for each supply voltage. The number of supply voltages and their values are assumed to be known and can be not the same for all the computational elements (In this paper, we do not deal with the problem of determining the number of supply voltages to be used, and the value for each one of them for each computational element). Our objective is then to assign new supply voltages, from the set of the given supply voltages, to the computational elements of G in order to minimize the total dynamic power consumption. Each computational element will have one and only one supply voltage. Since we want that G keeps operating at the clock period Π, the computational elements on critical paths will be kept operating at their original supply voltages, while the supply voltages of those off critical paths are replaced by low supply voltages.

As seen in Section III, minimizing dynamic power consumption in synchronous sequential digital designs by only changing the supply voltages of computational elements off critical paths is not enough. We have to also apply retiming to shift some computational elements from the critical paths.

As a summary, our main problem in this paper is then to develop an approach that allows to apply, in some manner, a retiming and a supply voltage scaling, to have a functionally equivalent design that operates at a target clock period and consumes the minimum dynamic power. We refer to this problem as the *MDPC* problem.

To optimally solve the *MDPC* problem, one then needs to unify retiming and supply voltage scaling. A Mixed Integer Linear Program (*MILP*) is proposed in [1] to realize this unification. However, since the problem is NP-hard in its general form, then this *MILP* cannot be solved, in general, by a polynomial time algorithm. Thus, heuristic approaches are required to determine approximate solutions to the problem. In the following we propose a heuristic to determine this kind of solutions to the problem. We call this heuristic *No-Way*. To the best of our knowledge, *No* other *Way* is available in the literature at this moment to determine approximate solutions to the *MDPC* problem.

Before presenting the algorithm *No-Way*, we first introduce the basic idea on which it is based. This idea comes from the following remark. Imagine that we have a design where there is one and only one register on each wire that connects each two computational elements. In this case, there is no need to compute a retiming. The supply voltage of each computational element can then be scaled down until the execution delay of that element becomes equal to the given clock period. For this design, the *MDPC* problem transforms to the problem of finding a manner to apply voltage scaling on that design in order to minimize the dynamic power consumption without changing the clock period of the design.

When only some wires have a number of registers greater than or equal to 1 for each one of them, then the idea is to maximize the number of wires having registers on them. Note that the resulting design must still have the target clock period. Maximizing the number of this kind of wires can be done by applying a retiming for a target clock period. Let us double-check if this will help to reduce dynamic power consumption, and if yes, is it possible to obtained the same dynamic power reduction as if one unifies retiming and supply voltage scaling? Assume that we have to solve the *MDPC* problem for the design in Fig. 1 for a target clock period $\Pi = 30$, using the supply voltages presented in Section III. As explained in Section III, there is two possible retimed designs both with $\Pi = 30$. The number of wires having registers is 3 for Fig. 2 (a) and is 4 for Fig. 2 (b). As presented in Section III, dynamic power consumption is reduced only for design in Fig. 2 (b). Note that in this case, Fig. 2 (b) is an optimal solution to the *MDPC* problem, and should be obtained if one unifies retiming and supply voltage scaling. Hence, by first maximizing the number of wires having registers on them while keeping the same clock period, and then applying a supply voltage scaling on the resulting design, we can produce optimal solution to the *MDPC* problem.

Our proposed heuristic to determine approximate solution to this problem is presented by Fig. 3. The algorithm *No-Way* produces a design with reduced dynamic power consumption, assuming that we do not consider dynamic power consumed by registers. Since to reduce the switching activities one needs to place registers on the output of computational elements, *No-Way* may allow to also reduce the switching activities in the design (see the synthesis of [5] and [6] presented in Section I). The design produced by *No-Way* is guaranteed to operate with the target clock period Π.

In the sequel, we devise methods to realize Steps 1 and 3 of the algorithm *No-Way*. Note that, in Step 1 of the algorithm, one may need to also consider register sharing [3] in order to reduce the total number of registers.

Inputs:

In1- Synchronous sequential design $G = (V, E, d, w)$.

In2- Target clock period Π.

In3- Possible supply voltages for each $v \in V$

In3.1: n_v supply voltages for each $v \in V$

In3.2: At supply voltage k $(1 \le k \le n_v)$, each $v \in V$ has an execution delay $d(v) = d_{v, k}$ and consumes the dynamic power $p_{v, k}$.

Output: A functionally equivalent design with clock period Π, and with a reduced dynamic power consumption.

Begin

1-Apply a retiming r on G to maximize the number of arcs $e_{u, v} \in E$ *such that* $w_r(e_{u, v}) \ge 1$, while keeping the clock period equal to Π. Let G_r be the resulting retimed design.

2- Remove all arc $e_{u, v} \in E$ *such that* $w_r(e_{u, v}) \ge 1$ from G_r. Let G_c be the resulting combinational design.

3-Solve the *MDPC* problem for the case of combinational designs using G_c, while keeping the clock period equal to Π. (G_c may not be connected. This may help to reduce the run time of the method used to solve the latter problem). Let G_o be the resulting design.

4-Put all the removed arcs in Step 2 back to G_o.

5-Return G_o.

End

Fig. 3 : Algorithm *No-Way*.

V. MAXIMIZING THE NUMBER OF WIRES WITH REGISTERS IN SYNCHRONOUS SEQUENTIAL DESIGNS

In this section, we provide a method to determine a retiming r that transforms a synchronous sequential design, $G = (V, E, d, w)$, operating with a clock period Π, to another functionally equivalent design, $G_r = (V, E, d, w)$, operating with the same clock period Π and having a maximum number of wires that possess registers.

For every arc $e_{u, v}$ in G_r, let $m(e_{u, v})$ be a 0-1 unknown variable defined as follows. $m(e_{u, v})$ takes the value 1 if $w_r(e_{u, v}) \geq 1$, and 0 otherwise. Based on the definition of $m(e_{u, v})$'s and on Inequality (5), we then have:

$$0 \leq m(e_{u, v}) \leq 1, \forall e_{u, v} \in E, \text{ and} \quad (12)$$

$$m(e_{u, v}) \leq w_r(e_{u, v}), \forall e_{u, v} \in E. \quad (13)$$

Using Equation (4), we can transform (13) to:

$$m(e_{u, v}) + r(u) - r(v) \leq w(e_{u, v}), \forall e_{u, v} \in E. \quad (14)$$

From Theorem 1, we can derive the following theorem. Proofs of all theorems presented in this paper are omitted due to space limitation.

Theorem 3 : *Let* $G = (V, E, d, w)$ *be a synchronous digital design, and let* Π *be a positive real number. Then there is a retiming r of G such that the clock period of the resulting retimed design* G_r *is less than or equal to* Π *if there exists an assignment of integer value r(v) to each node v in V, and 0/1 value* $m(e_{u, v})$ *to each arc* $e_{u, v}$ *in E, such that the following conditions are satisfied:*

(1) $0 \leq m(e_{u, v}) \leq 1, \forall e_{u, v} \in E,$ *(2)*

$m(e_{u, v}) + r(u) - r(v) \leq w(e_{u, v}), \forall e_{u, v} \in E,$ *and* *(3)*

$r(u) - r(v) \leq W(u, v) - 1, \forall u, v \in V$ such that $D(u, v) > \Pi$.

❑

The number of wires that possess registers in G_r is:

$$M = \sum_{e_{u, v} \in E} m(e_{u, v}). \quad (15)$$

The system of Inequalities in Theorem 3 may have many possible solutions. To have a solution that maximizes *M*, one can add the following formula

$$\text{Maximize}\left(\sum_{e_{u, v} \in E} m(e_{u, v})\right) \quad (16)$$

to that system. The resulting system is an Integer Linear Program (*ILP*), and given by Fig. 4.

Theorem 4 : *The ILP in Fig. 4 has always a solution, and can always be solved optimally in polynomial run-time.*

$$\text{Maximize}\left(\sum_{e_{u, v} \in E} m(e_{u, v})\right)$$

Subject to:

$$-m(e_{u, v}) \leq 0, \forall e_{u, v} \in E \quad (17)$$

$$m(e_{u, v}) \leq 1, \forall e_{u, v} \in E \quad (18)$$

$$m(e_{u, v}) + r(u) - r(v) \leq w(e_{u, v}), \forall e_{u, v} \in E \quad (19)$$

$$r(u) - r(v) \leq W(u, v) - 1, \forall u, v \in V \text{ such that } D(u, v) > \Pi \quad (20)$$

$$m(e_{u, v}) \text{ and } r(v) \text{ are non-negative integers, } \forall e_{u, v} \in E, \forall v \in V \quad (21)$$

Fig. 4 : An ILP to maximize *M*.

VI. AN APPROACH FOR SOLVING THE *MDPC* PROBLEM FOR COMBINATIONAL DESIGNS

The latency, denoted here by Π, of a combinational design $G = (V, E, d)$ can be fixed by the designer or assigned to a value equal to its lower bound. This lower bound denoted here by L is defined as:

$$L = \text{Max}_{\forall u, v \in V}\{d(P(u, v))\}. \quad (22)$$

The problem we address in this section is defined as follows. We are given a combinational design G that operates with a given supply voltage, which is called here highest supply voltage. We are also given multiple supply voltages and every delay as well as dynamic power consumption of each computational element for each supply voltage. The number of supply voltages and their values are assumed to be known and can be not the same for all the computational elements (Again, we do not address in this paper the problem of determining the number of supply voltages to be used, and the value for each one of them for each computational element). Our objective is then to assign new supply voltages, from the set of the given supply voltages, to the computational elements of G in order to minimize the total dynamic power consumption without changing the target latency of the design. Each computational element will have one and only one supply voltage. Since we want that G still has a given latency Π, then the computational elements on critical paths will be kept operating at their original supply voltages, while the supply voltages of those off critical paths are replaced by low supply voltages.

Before presenting an approach for solving the problem, we first provide additional notation and definitions. Based on the supply voltages used, we assume that we have n_v different implementations for each computational element v. If supply voltage V_{dd}^k, where $1 \leq k \leq n_v$, is used, then the computational element v has an execution delay $d(v) = d_{v, k}$ and consumes the dynamic power $p_{v, k}$. Assume that supply voltages are sorted from the highest to the smallest. For each $v \in V$, and for each k such that $1 \leq k \leq n_v$, we denote by $x_{v, k}$ a binary variable, which is equal to 1 if supply voltage k is used, and to 0 otherwise.

We announce the following theorem that we will use to develop a Mixed Integer Linear Program (*MILP*) to optimally solve the problem.

Theorem 5 : *Let* $G = (V, E, d)$ *be a combinational design, and let* Π *be a positive real number. Then the latency of G is less than or equal to* Π *iff there exists an assignment of non-negative real values s(v) to each node v in V such that the following conditions are satisfied: (1)* $-s(v) \leq 0, \forall v \in V$, *(2)* $d(v) + s(v) \leq \Pi, \forall v \in V$, *and (3)* $d(u) + s(u) - s(v) \leq 0, \forall e_{u, v} \in E$.

❑

Based on the definition of the problem, the execution delay of each computational element can be expressed as follows:

$$d(v) = \sum_{k = 1}^{n_v} d_{v, k} \cdot x_{v, k}, \forall v \in V, \text{ and} \quad (23)$$

$$\sum_{k = 1}^{n_v} x_{v, k} = 1, \forall v \in V. \quad (24)$$

From Theorem 5 and Equations (23) and (24), one can then derive the following theorem.

Theorem 6 : *Let* $G = (V, E, d)$ *be a combinational design, and let* Π *be a positive real number. Then the latency of G is less than or equal to* Π *if there exists an assignment of non-negative real values s(v) to each node v in V, and 0/1 values to each binary variable* $x_{v, k}$ *such that the following conditions are satisfied:*

(1) $\sum_{k = 1}^{n_v} x_{v, k} = 1, \forall v \in V,$

(2) $-s(v) \leq 0, \forall v \in V,$

(3) $\left(\sum_{k = 1}^{n_v} d_{v, k} \cdot x_{v, k}\right) + s(v) \leq \Pi, \forall v \in V,$ *and*

(4) $\left(\sum_{k = 1}^{n_u} d_{u, k} \cdot x_{u, k}\right) + s(u) - s(v) \leq 0, \forall e_{u, v} \in E.$

❑

We now return to the *MILP* for solving the *MDPC* problem. The objective function to minimize in the *MILP* is the total dynamic power consumption, which is equivalent to:

$$\text{Minimize}\left(\sum_{v \in V}\sum_{k=1}^{n_v} p_{v,k} \cdot x_{v,k}\right). \quad (25)$$

The system of equalities and inequalities in Theorem 6 may have many possible solutions. To have a solution that leads to a minimum dynamic power consumption, one may add Equation (25) to that system as well as 0-1 constraints on variables $x_{v,k}$ s. The resulting system is an *MILP* and presented by Fig. 5. By solving it we obtain the optimal solution to the *MDPC* problem.

$$\text{Minimize}\left(\sum_{v \in V}\sum_{k=1}^{n_v} p_{v,k} \cdot x_{v,k}\right) \quad (26)$$

Subject to

$$\sum_{k=1}^{n_v} x_{v,k} = 1, \forall v \in V \quad (27)$$

$$-s(v) \le 0, \forall v \in V \quad (28)$$

$$\left(\sum_{k=1}^{n_v} d_{v,k} \cdot x_{v,k}\right) + s(v) \le \Pi, \forall v \in V \quad (29)$$

$$\left(\sum_{k=1}^{n_u} d_{u,k} \cdot x_{u,k}\right) + s(u) - s(v) \le 0, \forall e_{u,v} \in E \quad (30)$$

$$x_{v,k} \in \{0, 1\}, \forall v \in V \text{ and } k = 1, ..., n_v \quad (31)$$

Fig. 5 : An MILP to optimally solve the MDPC problem.

Theorem 7 : *(a) The MILP in Fig. 5 has always a solution. (b) The design* G *obtained from the optimal solution to this MILP has a latency less than or equal to* Π .

❑

The size of the combinational designs in Step 3 of the algorithm *No-Way* may be very small. Consequently, the *MILP* in Fig. 5 can in this case be used to solve optimally the problem in reasonable time.

Note that this *MILP* cannot be always solved in polynomial time, since the problem is NP-hard in general. Hence, for general and very large designs, heuristics to determine approximate solutions to the problem are required. In the sequel, we provide a new heuristic to determine this kind of solutions to the problem.

Since the execution delay of any implementation of the computational element *v* is between $d_{v,1}$ and d_{v,n_v}, we have that:

$$\sum_{k=1}^{n_v} x_{v,k} d_{v,k} \ge d_{v,1}, \forall v \in V, \text{ and} \quad (32)$$

$$\sum_{k=1}^{n_v} x_{v,k} d_{v,k} \le d_{v,n_v}, \forall v \in V. \quad (33)$$

Linear programs are solvable in polynomial time. To determine approximate solutions to the *MDPC* problem, a Linear Program (*LP*) can be derived from the *MILP* in Fig. 5. This by dropping the constraints (31), and adding the constraints (32) and (33) to the constraints of the resulting linear program. The dynamic power consumption obtained in the optimal solution of this *LP* is a lower bound on the value of (26).

Solving this *LP* may lead to non-realizable solutions; that is there may exist at least one computational element v such that $d(v)$ found in these solutions does not correspond to any execution delay of the available implementations of v. But a non-realizable solution from a solution to the *MDPC* problem can be transformed to a realizable one. Indeed, once the *LP* is solved, the supply voltage under which computational element v must operate is V_{dd}^m, where m is non-negative integer such that $d_{v,m} \le d(v) < d_{i,m+1}$. Note that, if v is on a critical path, then $m = 1$.

VII. EXPERIMENTAL RESULTS

The objective in this section is to assess the quality, in terms of dynamic power consumption, of the designs produced by the proposed approach. Circuits of small size are used to compare the quality of the designs produced by our approach *vs.* by the exact method proposed in [1]. These circuits are at the system level, where computational elements are for instance *IP Blocks* such as adders and multipliers. To also assess the execution times required by algorithms of the proposed approach, we use large size circuits from *ISCAS'89* benchmark suite [14]. The proposed approach can be used for any number of supply voltages as well as any value for these supply voltages. Since we do not have a library of supply voltages that is available at this moment, we opt to an automatic choice. As supply voltages, we use in this paper the first *x*-supply voltages from the set $\{5V, 4.5V, 4V, 3.5V, 3V, 2.5V, 2V, 1.5V\}$, where $x = 1, 2, ..., 8$; recall that determining the right number of supply voltages and their values for each computational element is not the problem we address in this paper. The difference between two successive supply voltages is fixed to 0.5V, since we also want to test the effectiveness of the proposed approach in producing designs without level converters based on [9]. The approach in [9] assumes that a level-converter between computational elements u and v can be omitted if (34) is satisfied:

$$\forall e_{u,v} \in E, \sum_{k=1}^{n_v} x_{v,k} V_{dd}{}^k - \sum_{k=1}^{n_u} x_{u,k} V_{dd}{}^k \le V_{st}, \quad (34)$$

where V_{st} is a given value, which is fixed arbitrarily here to 0.5V. We assume that supply voltages are greater than $2 \cdot V_{th}$, where V_{th} is the threshold voltage. We use $V_{th} = 0.7V$, which is a typical value used in the literature. To determine the delay $d_{i,k}$ and the dynamic power consumed $p_{i,k}$, for a given computational element v and supply voltage V_{dd}^k, we proceed as follows. First, we use the expression

$$d_{v,k} = (V^k_{dd}/(V^k_{dd} - V_{th})^2) \cdot ((V^1_{dd} - V_{th})^2 / V^1_{dd}) \cdot d_{v,1}$$

described in [8], where V_{dd}^1 is assumed to be known. Second, $p_{v,k}$'s are determined assuming that the fanout of the computational element v has the same loading capacitance. From Equation (2), we then have:

$$p_{v,k} = KC_0 f. (Fanout_v \cdot (V^k_{dd})^2).$$

For each circuit, the clock period Π is fixed to the minimal value determined by applying a *retiming* for clock period minimization on the circuit operating at the highest supply voltages (i.e., 5V). More reductions of dynamic power consumption should be obtained if the clock period is fixed to a value greater than this used value.

We implement the algorithm in Fig. 3. For Steps 1 and 3 of this algorithm, we use respectively the *ILP* in Fig. 4, and the *LP* derived from Fig. 5 as discussed in Section VI. Once the *LP* is solved, the obtained solution is then converted to a realizable design as discussed in Section VI (i.e., if for example we find a supply voltage of value say 4.75V, we change it to 5V instead of 4.5V, since 4.75V does not belong to the set of possible supply voltages and since a slowest component can always be replaced by a fastest one while satisfying timing constraints). The dynamic power consumption of the design is then computed.

All experiments were done using an UltraSparc 10 with 1GB RAM. We use [4] to solve the *ILPs* and *LPs*. Recall that, based on Theorem 4, these *ILPs* are solvable in polynomial time. For each circuit, the *ILPs* and *LPs* are automatically generated by a module we coded in C++.

Before presenting obtained numerical results, we first introduce some notations and definitions. We denote by P_h the dynamic power consumed if one uses only the highest supply voltages. By P_m we denote the dynamic power consumed if one used the first $(x-1)$ supply voltages as defined above. We define the relative saving RS as follows:

$$RS = ((P_h - P_m)/P_h) \times 100 .$$

We denote by *#C* the number of required level-converters in the design. A level-converter is required if (34) is not satisfied in the design (this design is obtained by transforming the solution of the *LP* to a realizable solution as explained above and in Section VI).

Obtained numerical results are presented by Tables I and II. Table I presents numerical results using the exact approach in [1] *vs.* the approach of this paper, using small size circuits. Its first column presents the name of circuits used. The second column presents P_h divided by (KC_0f). The *x*th column, where $x = 3, 4, ..., 9$, gives the relative saving ratios: RS^* that corresponds to the use of the exact method in [1], and RS that corresponds to the use of the proposed approach. The content of each cell of that column is as follows: the first line corresponds to the relative saving ratios without considering (34), and its second line corresponds to relative saving ratios when (34) is considered. We omit column 9 since its content is the same as the one of column 8. As it can be observed from the first line of each cell of columns 2-8 in Table I, the values of RS are from 2.78% to 36.22%, and are sometimes the same as those of RS^*. This indicates that the proposed approach is able to sometimes solve optimally the *MDPC* problem. For the second line of each one of these columns, the performance of the proposed approach is still close to the one of [1].

Since the *MDPC* problem is NP-hard in general, the approach in [1] cannot be used to produce designs with reduced dynamic power consumption in reasonable run-time for circuits of large size such as circuits from *ISCAS'89* benchmark suite. In contrast to [1], the proposed approach have been exercised on circuits from this benchmark suite, and proved that it is able to produce designs with reduced dynamic power consumption in reasonable run-time as can be seen in Table II. The first two columns of this table are as for the case of Table I. Its *x*th column, where $x = 3, 4, ..., 9$, gives the relative saving ratio RS, the number of required level-converters *#C*, and the run-time in seconds *T(s)* required by the proposed approach. The content of each cell of that column is as follows: the first line corresponds to the values of RS, *#C*, and *T(s)* without considering (34); its second line corresponds to the values of RS, *#C*, and *T(s)* when (34) is considered. As it can be observed from the first line of each cell of columns 2-9 in Table II, relative dynamic power savings as high as 61% have been obtained when (34) is not considered, in a run-time around 1h. When this equation is incorporated in constraints of the *LP*, relative dynamic power savings as high as 40% are obtained. We counted the level-converters required for the produced designs. When Equation (34) is not used in the constraints of the *LP*, the number of required level-converters varies from 0 to 182 (see the first line of each cell of columns 2-9 in Table II). But, when this equation is incorporated in constraints of the *LP*, this number became 0 except for circuits S1238 and S1488 (see the second line of each cell of columns 2-9 in Table II). For these two circuits, 2 level-converters were required in the produced design when 7 and 8 supply voltages are used to save an additional very small portion of dynamic power compared to when 2-6 supply voltages are used. This indicates that the proposed approach can produce designs with reduced dynamic power consumption and also without level-converters.

Power consumed by level-converters may be ignored if for instance computational elements are adders and/or multipliers as done in [12]. When computational elements consume power close to the power consumed by a level-converter, then the effectiveness of the proposed approach without using Equation (34) will depend on the ratio of the total power saved and the total power consumed by level-converters. Currently, we do not have benchmarks to experimentally measure this ratio.

VIII. CONCLUSIONS

Digital systems with multiple supply voltages are designed. The objective in using multiple supply voltages is to reduce dynamic power consumption. To this end, one needs to solve the problem of minimizing dynamic power consumption under timing constraints by supply voltage scaling. This problem is widely addressed in the literature in the case of combination designs, but not for the case of sequential designs. It is NP-hard in general.

We have focused on solving the problem in the case of sequential designs. For this kind of designs, applying supply voltages scaling alone to reduce dynamic power consumption is not enough as we have shown with an example. Critical paths are determined by the position of registers in the design. Hence, repositioning the registers while applying supply voltages scaling is required. Since the problem is NP-hard in general even for the case of combinational designs, instead of unifying retiming and supply voltage scaling using an exact algorithm, we proposed a heuristic approach with polynomial time complexity. To the best of our knowledge, this is the first heuristic approach to determine approximate solutions to the *MDPC* problem. For this approach, we proposed to first apply a retiming that maximizes the number of arcs with registers, and then to apply supply voltage scaling on the retimed design. This kind of retiming tries to maximize the number of computational elements off critical paths, and hence to allow them to operate with a lower supply voltage.

We devised polynomial time complexity methods to realize this kind of retiming as well as the supply voltage scaling on the retimed design.

The proposed approach proved effective in producing designs with reduced dynamic power consumption, and also without level converters. Although no optimization is done at this moment to speed up the run-time of the current implementation of the approach, experimental results have shown that around 1h was enough for the proposed algorithm to determine approximate solutions to the problem for designs with large size such as circuits from *ISCAS'89* benchmark suite.

The kind of retiming we proposed puts registers on the outputs of many computational elements. Hence, it may allow to also reduce dynamic power due to the switching activities. Note that, retiming for minimizing the switching activities is proposed in the literature, and all the approaches to realize it try to put registers on the output of computational elements of high switching activities.

We are working on extending this first version of the proposed approach. In the present version, the number of registers sometimes decreases by applying the suggested kind of retiming. Of course, it may increase. Hence, it may increase or decrease the total power consumption. Ongoing research is then to control the power consumption due to registers. Also, to make the proposed retiming more smart by assigning a cost to each wire; at the present version, the cost is constant and is the same for all the wires. Such new costs may allow to put registers on the output of computational elements that switch the most, and/or may allow to shift off critical paths computational elements that will lead to the most dynamic power

saving. Yet an other implementation issue related to how to apply supply voltage scaling on the retimed design: one may need to give preference to first scale down the supply voltage of computational elements that have registers on their outputs; this may allow to also reduce dynamic power due to these registers if we assume that any computational element and its fanout registers have the same supply voltage.

The proposed approach can be used for any number of supply voltages as well as any value for these supply voltages. But, more dynamic power savings could be obtained if one first solve the problem of determining the right number of supply voltages and their values for each computational element. This problem was beyond the scope of this paper and still be open.

REFERENCES

[1] N.Chabini, I.Chabini, E.-M.Aboulhamid, and Y.Savaria, "Unification of Basic Retiming and Supply Voltage Scaling to Minimize Dynamic Power Consumption for Synchronous Digital Designs," *Proceedings of the ACM Great Lakes Symposium on VLSI*, April 28-29, 2003, Washington DC, USA, pp. 221-224.

[2] J.-M.Chang and M.Pedram, "Energy minimization using multiple supply voltages," *IEEE Trans. on VLSI*, Vo.5, No.4, 1997, pp.1-8.

[3] C.E. Leiserson and J.B. Saxe, "Retiming synchronous circuitry," Algorithmica, pp. 5-35, Jan., 1991.

[4] The LP_Solve Tool: *ftp://ftp.ics.ele.tue.nl/pub/lp_solve/*

[5] K.N. Lalgudi and M. Papaefthymiou, "Fixed-phase retiming for low power," *Proc. of the Int. Symposium of Low Power Electronics and Design*, pp. 259-264, 1996.

[6] J. Monteiro, S. Devadas, and A. Ghosh, "Retiming sequential circuits for low power," *Proc. of the IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 398-402, 1993.

[7] A. Raghunathan, N. K. Jha, and S. Dey, *High-Level Power Analysis and Optimization*, Kluwer Academic Pub., MA, 1997.

[8] K.Usami and M.Horowitz, "Clustered voltage scaling technique for low-power design", *Proc. Int. Workshop on Low Power Design*, 1995, pp. 3-8.

[9] Y.-J.Yeh, S.-Y.Kuo, and J.-Y.Jou, "Converter-free multiple-voltage scaling techniques for low-power CMOS digital design," *IEEE Transactions on CAD of ICAS*, V.20, N.1, 2001, pp.172-6.

[10] I.Mutsunori, and *al.*, "A low-power design method using multiple supply voltages," *Proc. of the Int. Symp. on Low Power Electronics and Design*, August 1997, pp. 36-41.

[11] T. Pering, T.D. Burd, and R.W. Brodersen, "Voltage scheduling in the IpARM microprocessor system," *Int. Symp. on Low Power Electronics and Design*, 2000, pp. 96-101.

[12] S.Raje and M.Sarrafzadeh, "Scheduling with multiple voltages," *Integration, the VLSI J.*, Vol. 23, 1997, pp. 37-59.

[13] K. Roy, Wei Liqiong, Chen Zhanping, "Multiple-V_{dd} multiple-V_{th} CMOS (MVCMOS) for low power applications," *Proc. of IEEE Int. Symposium on Circuits and Systems*, Vol.1, 1999, pp. 366-70.

[14] ISCAS'89 Benchmark suite. North Carolina State University, Department of Computer Science. *http://www.cbl.ncsu.edu/benchmarks*

TABLE I : ASSISSMENT OF THE EXACT METHOD IN [1] *VS.* THE APPROACH OF THIS PAPER.

Circuit Name	**1-Vdd** $P^h_{dyn}/(KC_0f)$	**2-Vdd** *RS*(%), RS(%)*	**3-Vdd** *RS*(%), RS(%)*	**4-Vdd** *RS*(%), RS(%)*	**5-Vdd** *RS*(%), RS(%)*	**6-Vdd** *RS*(%), RS(%)*	**7-Vdd** *RS*(%), RS(%)*
AllPoleLatticeFilter	414	4.83, 3.62 4.83, 3.62	6.03, 6.03 6.03, 6.03	6.03, 6.03 6.03, 6.03	6.03, 6.03 6.03, 6.03	6.03, 6.03 6.03, 6.03	6.03, 6.03 6.03, 6.03
BiquadraticFilter	343	7.48, 4.49 7.48, 4.49	14.97, 8.98 11.97, 5.98	17.66, 11.67 13.77, 5.98	18.86, 12.87 13.77, 5.98	18.86, 13.47 13.77, 5.98	18.86, 13.47 13.77, 5.98
SecondAvenhausFilter	412	7.28, 7.28 7.28, 7.28	13.34, 12.13 12.13, 10.92	16.26, 13.59 12.86, 10.92	17.23, 13.59 12.86, 10.92	17.23, 13.59 12.86, 10.92	17.23. 13.59 12.86, 10.92
FifthOrderWaveDigitalFilter	1281	7.49, 6.24 7.49, 6.24	11.55, 10.53 10.30, 7.80	12.95, 12.17 10.53, 8.04	14.44, 14.05 10.53, 8.04	15.06, 14.67 10.53, 8.04	15.53, 14.67 10.53, 8.04
ThirdAvenhausFilter	566	8.83, 8.83 8.83, 8.83	16.78, 16.78 15.90, 15.9	21.02, 21.02 18.02, 18.02	23.14, 23.14 18.02, 18.02	23.85, 23.85 18.02, 18.02	23.85, 23.85 18.02, 18.02
Diffirential Equation Solver	588	10.71, 10.71 10.71, 10.71	22.61, 20.91 12.41, 11.56	27.72, 27.04 12.41, 11.56	34.52, 33.84 12.41, 11.56	37.24, 36.22 12.41, 10.71	37.24, 36.05 12.41, 11.56
FourAvenhausFilter	720	9.72, 9.72 9.72, 9.72	18.75, 18.75 18.05, 18.05	23.75, 23.75 21.38, 21.38	26.52, 26.52 21.94, 21.94	27.63, 27.63 21.94, 21.94	27.63, 27.63 21.94, 21.94
PolynomDivider	309	9.7, 9.7 9.7, 9.7	19.41, 19.41 17.79, 17.79	25.24, 25.24 20.71, 20.71	27.83, 27.83 20.71, 20.71	29.12, 29.12 20.71, 20.71	29.12, 29.12 20.71, 20.71
SecondOrderIIR	359	2.78, 2.78 2.78, 2.78	5.57, 5.57 2.78, 2.78	5.57, 5.57 2.78, 2.78	5.57, 5.57 2.78, 2.78	5.57, 5.57 2.78, 2.78	5.57, 5.57 2.78, 2.78
Correlator	283	8.48, 8.48 8.48, 8.48	15.54, 15.54 13.78, 13.78	18.02, 18.02 13.78, 13.78	19.43, 19.43 13.78, 13.78	19.78, 19.43 13.78, 13.78	19.78, 19.43 13.78, 13.78

TABLE II : ASSISSMENT OF THE PROPOSED APPROACH ON LARGE-SIZE CIRCUITS.

Circuit Name	**1-Vdd** $P^h_{dyn}/(KC_0f)$	**2-Vdd** *RS(%), #C, T(s)*	**3-Vdd** *RS(%), #C, T(s)*	**4-Vdd** *RS(%), #C, T(s)*	**5-Vdd** *RS(%), #C, T(s)*	**6-Vdd** *RS(%), #C, T(s)*	**7-Vdd** *RS(%), #C, T(s)*	**8-Vdd** *RS(%), #C, T(s)*
S344	6225	12.88, 0, 15 12.88, 0, 15	26.21, 36, 15 21.87, 0, 16	32.17, 40, 15 24.27, 0, 16	36.7, 53, 15 23.53, 0, 16	38.02, 57, 15 23.63, 0, 18	38.02, 56, 17 23.63, 0, 20	39.03, 57, 18 23.63, 0, 21
S641	8457	8.73, 0, 6 8.73, 0, 6	18.02, 34, 6 15.71, 0, 7	23.28, 36, 6 17.45, 0, 7	29.92, 47, 7 18.49, 0, 8	32.63, 48, 7 18.65, 0, 10	35.35, 56, 7 18.65, 0, 12	35.62, 58, 9 18.65, 0, 13
S713	9758	9.24, 0, 8 9.24, 0, 8	19.18, 34, 8 17.08, 0, 9	25.08, 34, 8 19.48, 0, 10	32.58, 43, 9 21.94, 0, 11	35.73, 46, 9 21.65, 0, 13	39.49, 53, 10 21.77, 0, 16	39.78, 55, 12 21.77, 0, 20
S1238	25017	10.66, 0, 151 10.66, 0, 159	20.85, 61, 157 19.39, 0, 164	26.36, 61, 157 22.56, 0, 175	32.26, 84, 164 25.6, 0, 189	34.42, 101, 174 26.15, 0, 222	36.01, 144, 192 26.41, 2, 258	36.23, 157, 203 26.42, 2, 306
S1423	24533	13.74, 0, 71 13.74, 0, 75	28.6, 56, 72 25.54, 0, 79	37.18, 58, 75 30.97, 0, 93	48.2, 84, 77 35.91, 0, 97	53.51, 87, 82 38.12, 0, 130	59.98, 121, 92 40.34, 0, 153	61, 127, 114 40.68, 0, 202
S1488	33360	8.99, 0, 3618 8.99, 0, 3623	17.62, 89, 3620 14.72, 0, 3633	21.52, 94, 3631 17.2, 0, 3659	25.34, 147, 3637 19.36, 0, 3683	26.67, 152, 3663 19.77, 0, 3762	27.59, 178, 3752 20.13, 2, 3812	27.62, 182, 3794 20.13, 2, 3898
S1494	33863	8.91, 0, 3628 8.91, 0, 3635	17.16, 85, 3631 15.11, 0, 3645	20.76, 9, 3644 17.45, 0, 3675	24, 128, 3650 19.78, 0, 3715	25.14, 139, 3682 20.19, 0, 3788	25.94, 167, 3711 20.46, 0, 3850	25.98, 170, 3755 20.46, 0, 3925