

Low-Power Scheduling with Resources Operating at Multiple Voltages

Wen-Tsong Shiue, *Member, IEEE* and Chaitali Chakrabarti, *Member, IEEE*

Abstract—This paper presents a resource-constrained scheduling scheme and a latency-constrained scheduling scheme that minimize power consumption for the case when the resources operate at multiple voltages. The resource-constrained scheduling reduces the power consumption by maximally utilizing resources operating at reduced voltages and, at the same time, reducing the latency. The latency-constrained scheduling scheme reduces the power consumption by assigning as many nodes (of the data flow graph) as possible to the resources operating at reduced voltages. Both schemes consider the effect of switching activity on the power consumption of the functional units. In addition, both schemes use heuristics to reduce the power consumed by the level shifters. Experiments with HLS benchmark examples show that the proposed schemes achieve significant power reduction when the operating voltages are 5 and 3.3 V or 5, 3.3, and 2.4 V.

Index Terms—Behavioral synthesis, latency-constrained scheduling, low-power design, multiple voltage scheduling, resource-constrained scheduling.

I. INTRODUCTION

IN THE PAST, the primary considerations in very large-scale integration (VLSI) design were performance, cost, reliability, and area. In recent years, the demand on portability has compelled the growth of personal computing devices such as portable desktops, audio- and video-based multimedia products, and wireless communication systems. All these devices require not only high-speed computation and complex functionality, but demand low-power consumption. Consequently, power considerations have become an increasingly dominant factor in VLSI design today.

It has been well established that to design a low-power system, power has to be optimized at all levels of the design process: system, algorithm (behavior), architecture, logic, circuit, and processing technology [1]. The work presented in this paper focuses on behavioral level power optimization. Research in power optimization at the behavioral level has not been as intensive as in optimization at the logic synthesis or circuit levels. Algorithmic level transformations have been used to optimize power in [2]. Procedures for minimizing the ac-

tivity of a functional unit have been proposed in [3]–[5]. Other behavioral approaches to reducing power include reducing the number of registers, reducing the switching in registers, and efficient register allocation [4]. The most effective way of reducing power, however, is by reducing the supply voltage. The resulting increase in the circuit delay (and reduction in the throughput) has to be compensated by parallelism and pipelining [1]. Another way of maintaining the throughput is to use resources operating at multiple voltages [6]–[12]. Nodes on the critical paths can be assigned to the high-voltage resources (thus meeting the required timing constraints) while nodes on the non-critical paths can be assigned to low-voltage resources (thus reducing the power consumption). However, a number of practical problems must be overcome before use of multiple voltage becomes prevalent [6]. These include routing of multiple supply voltage lines, area/delay overhead of level shifters, and lack of design tools and methodologies for multiple voltage system design.

In this paper, we address the problem of scheduling a data-flow graph under resource constraints and under latency constraints, for the case when the resources operate at multiple voltages. The proposed resource-constrained algorithm tries to achieve a balance between reducing the number of control cycles and maximally utilizing the resources operating at reduced voltages. The algorithm is list based. In each control cycle, the ready nodes are assigned based on their priorities, where the priority of a node is a function of its: 1) depth; 2) mobility; 3) switched capacitance; 4) interconnection complexity; and 5) need for a level shifter. The algorithm has polynomial time complexity, while the other existing algorithms based on dynamic programming [9] or integer linear programming [8], [12] have pseudo-polynomial or even exponential time complexity. The integer linear programming (ILP)-based algorithms and the heuristic algorithms of [12], as well as the dynamic programming-based algorithm of [9], do not consider the effect of switching activity and the power consumed by the level shifters. In addition, the power models used in [9] are simplistic since they assume that the power consumption is only a function of the supply voltage. The ILP-based algorithm in [8] assumes a user-specified number of supply voltage levels (the actual values are not specified) and considers the power consumed by both the level shifters and the interconnection (MUX's). A comparison of all these algorithms reveals that our list-based algorithm which considers the effect of switching activity, interconnect complexity, and power consumption by level shifters is computationally less complex.

The latency-constrained scheduling algorithm proposed in this paper also has polynomial time complexity and minimizes

Manuscript received December 1998; revised February 2000. This work was supported by the National Science Foundation (NSF) State/Industry/University Cooperative Research Centers' Center for Low Power Electronics under NSF Grant EEC-9523338, by the State of Arizona, and by Burr-Brown, Inc., Conexant, Gain Technology, Intel Corporation, Medtronic Microelectronics Center, Microchip Technology, Motorola, Inc., The Motorola Foundation, Raytheon, Texas Instruments, and Western Design Center. This paper was recommended by Associate Editor M. Bayoumi.

The authors are with the Department of Electrical Engineering and the Center for Low Power Electronics, Arizona State University, Tempe, AZ 85287-5706 USA.

Publisher Item Identifier S 1057-7130(00)04977-6.

power consumption for the case when the resources operate at multiple voltages (5, 3.3, and 2.4 V). Since there is no restriction on the number of available resources, the bias is towards operating resources at low voltages without violating the timing constraint. The minimum cost assignment takes into account the switched capacitance and the power consumed by the level shifters. Experiments with some benchmark examples show that for tight timing constraint (i.e., when the latency is equal to the critical path length), the average reduction is $\approx 39\%$ for a two-voltage system (5 V, 3.3 V) and $\approx 58\%$ for a three-voltage system (5 V, 3.3 V, 2.4 V). The problem of latency-constrained scheduling with multiple voltage units has been addressed in [10]. The algorithm in [10] iteratively computes the voltage assignment of each node and is computationally more complex compared to the one-pass algorithm presented here. In addition, it does not consider the effect of switching activity and the power consumed by level shifters. The ILP-based technique of [12] also does not consider the effect of switching activity and the power consumed by level shifters. The dynamic programming technique proposed in [6] has pseudo-polynomial complexity and produces optimal results for trees (and not for general directed acyclic graphs). This algorithm takes into account both the switched capacitance and the power consumed by the level shifters.

The rest of the paper is organized as follows. Section II deals with the preliminaries. Section III defines the problems formally. Section IV presents the algorithm and illustrative examples for resource-constrained scheduling, while Section V presents the algorithm and examples for latency-constrained scheduling. Section VI concludes the paper.

II. PRELIMINARIES

A. Definitions

A *data flow graph* (DFG) is a directed acyclic graph whose nodes represent operations and edges represent dependencies between the operations. For instance, an edge from node i to node j means that node i has to be executed before node j . Each node has a delay associated with it, where the delay is a function of the operating voltage.

The *depth of a node* is defined as the length of the path from the node to the sink in the data flow graph. We assume that the sink has a depth equal to 1. In Fig. 1, node 5 has depth 1, node 4 has depth 2, node 3 has depth 3, and so on.

The *mobility* of a node F is the difference between its as-late-as-possible (ALAP) schedule time and its as-soon-as-possible (ASAP) schedule time. For resource-constrained scheduling, the ASAP and the ALAP times are computed assuming that the latency is equal to the length of the critical path. For latency-constrained scheduling, they are computed using the specified latency.

In a list-based schedule, there is a *ready set* associated with every control cycle, where a ready set is defined as the set of nodes that could all be assigned at that particular control cycle if there were no resource constraints. The ready sets for control cycle 1 in the DFG of Fig. 2 are M [1, 2, 6, 8] for multipliers and A [10] for adders.

Algorithm_Resource (3 voltage levels)

1. Determine the ASAP, ALAP, depth, and mobility of each node
2. For each cycle, do the following:
 - for nodes in the ready set with mobility ≥ 2 /* ignore this step if considering 2 voltage levels*/
 - if there are insufficient number of 2.4V resources,
 - Compute priority;
 - Assign high priority nodes to the available 2.4V resources;
 - for nodes in the ready set with mobility ≥ 1
 - if there are insufficient number of 3.3V resources,
 - Compute priority;
 - Assign high priority nodes to the available 3.3V resources;
 - for the remaining nodes in the ready set
 - if there are insufficient number of 5V resources,
 - Compute priority;
 - Assign high priority nodes to the available 5V resources;

/* lower voltage resources should be utilized */

if 2.4V or 3.3V resources are under utilized,

- Compute priority;
- Assign high priority nodes to the available 2.4V and 3.3V resources;

- update the ASAP and ALAP value of each node.

Fig. 1. Algorithm for resource-constrained scheduling.

B. Power Model

There are three major sources of power consumption: 1) switching; 2) short-circuit current; and 3) leakage current. Among these, switching is the most dominant factor. The power consumed due to switching in a digital circuit is given by $P_{\text{switching}} = \alpha_{0-1} C_L V_{\text{dd}}^2 f_{\text{clk}}$, where α_{0-1} is the probability of a 0–1 transition, i.e., the average number of times a node makes a power consuming transition (0–1) in one clock period C_L is the load capacitance, V_{dd} is the supply voltage swing, and f_{clk} is the clock frequency.

The power consumption of the different functional units (multipliers, adders, and level shifters) have been borrowed from [6]. In [6], the power consumption values for the multipliers and adders are described in terms of the switched capacitance $\alpha_1 * C1(V) + \alpha_2 * C2(V) + C3(V)$, where α_1 and α_2 are the transition probabilities of the inputs, and $C1(V)$, $C2(V)$ and $C3(V)$ are the regression coefficients that are used to model the power consumption of the functional units operating at V volts.

III. PROBLEM DEFINITION

The input to the resource and latency-constrained algorithms is a data flow graph, input data stream, and resources that operate at multiple voltages. We assume that: 1) the functional unit can be operated at three voltages: 5 V, 3.3 V, and 2.4 V and 2) if the delay of the functional unit at 5 V is Δ , then the delay of the unit at 3.3 V is 2Δ and that at 2.4 V is 3Δ . We choose a maximum of three operating voltages, since increasing the number of operating voltages to more than three does not cause a significant reduction in the power consumption.

We assume that multiple power lines are available. Thus there is no need for on-chip level shifters for the power lines.

On-chip level shifters are, however, required in the data path, since data is transferred between resources operating at different voltages. We assume that level shifters are only required between a low-voltage resource and a high-voltage resource (i.e., step-up level shifter). Our scheme assigns lower voltage to the nodes with larger ASAP values such that the number of step-up level shifters is minimized.

A. Resource-Constrained Scheduling

In resource-constrained scheduling, the problem is to schedule the data flow graph, given the resource constraints, such that the number of control cycles is minimum. The resource constraint is specified as follows.

- 1) The number of computational units for each type is given.
- 2) The delay of each type of computational unit is also given.
The delay is directly related to the operating voltage.

While on the one hand, it is important to minimize the number of control cycles needed to finish a computation because it determines the sample period of the system (and is thus related to power consumption), on the other hand, operating resources with reduced supply voltage reduces the power at the expense of an increase in the number of control cycles. The algorithm presented in this paper tries to balance the conflicting requirements of reducing the number of control cycles and utilizing resources operating at reduced voltage.

B. Latency-Constrained Scheduling

In latency-constrained scheduling, the problem is to assign voltages to the nodes of the DFG, given the timing constraint, such that the power consumption is minimum. Thus, the more nodes that can be assigned to lower voltages, the greater is the power reduction. The statement of assigning “voltage to a node” is equivalent to assigning voltage to the functional unit onto which the node is mapped.

IV. RESOURCE-CONSTRAINED SCHEDULING

A. Algorithm

In this section, we present a list-based algorithm for resource-constrained scheduling. The input to this algorithm in the DFG, resource constraints, input data streams, and the output is the voltage assignment of each node. The algorithm is list based and operates in the following way.

In each control cycle, nodes are first assigned to the 2.4-V resources, then to the 3.3-V resources, and finally to the 5-V resources. The procedure is as follows. Nodes with mobility $> = 2$ are considered first. Priorities are determined for this subset of nodes, and the high-priority nodes are assigned to the 2.4-V resources. Next, priorities are determined for nodes with mobility $> = 1$, and the high-priority nodes are assigned to the 3.3-V resources. Then, priorities are determined for nodes with mobility $> = 0$ and the high-priority nodes are assigned to the 5-V resources. Finally, the remaining nodes (if any) are assigned to the available low-voltage resources. An assignment in this step results in an increase in the latency. The priority of a node is

a function of its depth, mobility, switched capacitance, interconnection complexity, and the possibility of requiring a level shifter.

- 1) Depth is the most important parameter, since it is directly linked to the latency, and reducing latency is an important goal.
- 2) Mobility is the second most important parameter. Nodes with high mobility are given high priority during assignment of low-voltage resources, while nodes with low mobility are given high priority during assignment of high-voltage resources (this is because assigning a node with high mobility to a high-voltage resource may result in an increase in the latency).
- 3) Switched capacitance is the third most important parameter. The switching at the inputs of the functional units are determined and the switched capacitance calculated. Assignments which result in lower switched capacitance are given higher priority. Details of this scheme are described later in this section.
- 4) Interconnection complexity is the fourth most important parameter. The scheme involves identifying templates and trying to assign nodes of different instances of the same template to the same resources. Details of this scheme are described later in this section.
- 5) Level shifter introduction is the fifth most important parameter. By assigning a voltage that is the same or lower than the voltages that have been assigned to its parents, level shifters can be avoided.

1) *Switched Capacitance:* In resource-constrained scheduling, multiple nodes get assigned to the same functional unit, and thus, the data stream that is finally input to a functional unit is obtained by interleaving the input data streams of the individual nodes. However, the switching activity at the inputs of the functional unit (which is related to the power consumption of the functional unit) in cycle i is the average number of bit level switches between the data streams of the nodes that have been assigned in cycles i and $i - 1$. In each control cycle, the transition activity for each possible mapping (of ready node to functional unit) is calculated and this activity is used to calculate the switched capacitance using the model in [6]. Since there are two inputs to each functional unit, there are two ways in which the input data streams can be interleaved. Both ways are investigated, and the one with the lower switched capacitance is considered. If the number of nodes in the ready set is r , and the number of functional units that it could be assigned to is m , then a total of $\max(m, r)C_{\min(m, r)}$ assignments have to be checked, and the assignments with smaller switched capacitance are given higher priority.

2) *Interconnection Complexity:* It has been established that the interconnection complexity can be reduced by identifying repetitive patterns or templates and assigning nodes of different instances of the same template to the same resources [13]. Thus, once one instance of a template is mapped to a set of hardware units, the procedure tries to map other instances of the template to the same set of units. In the list-based scheduling algorithm, if a “ready” node belongs to a template that has already been mapped, a high priority is given to assigning the ready node

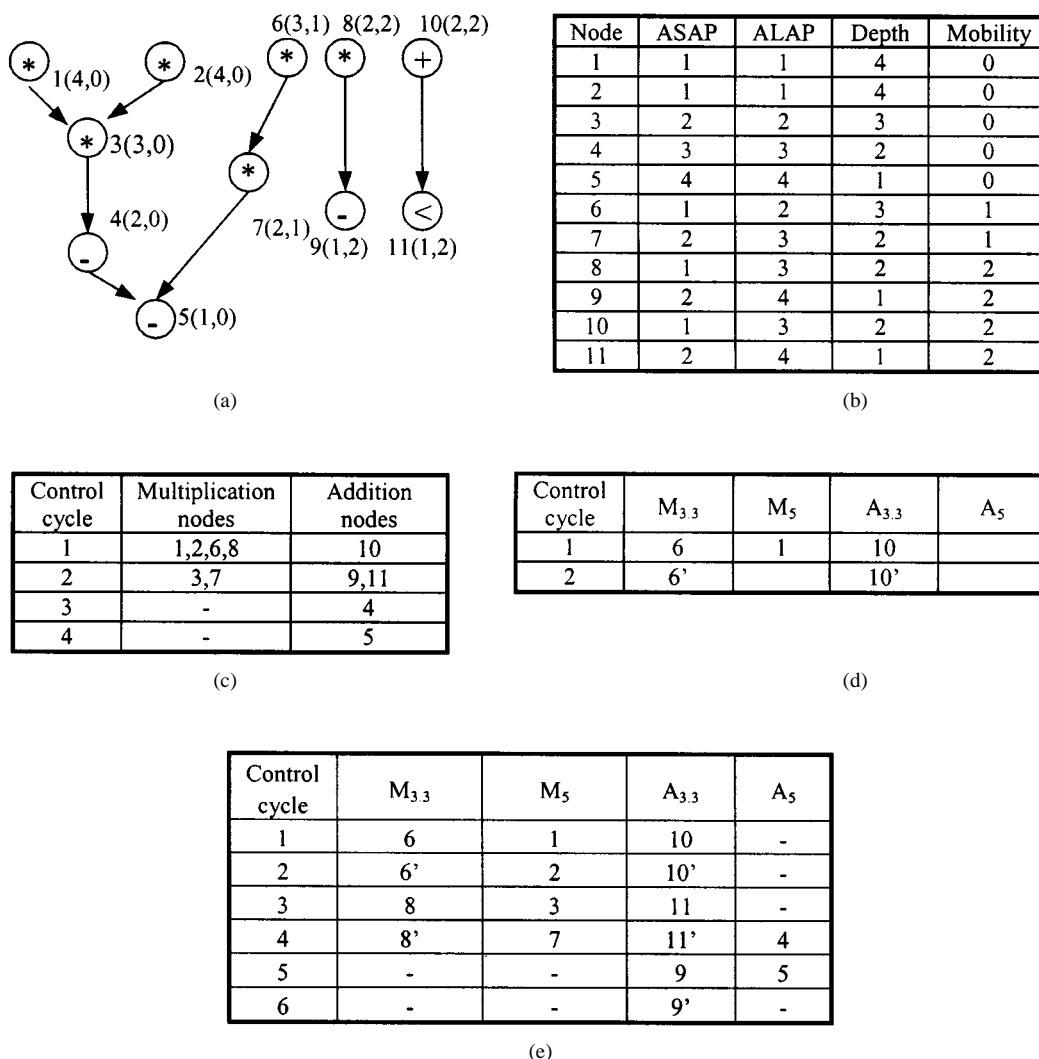


Fig. 2. Resource-constrained scheduling. Example DIFFEQ for the case when there are one 3.3-V multiplier, one 5-V multiplier, one 3.3-V adder, and one 5-V adder. (a) DFG of differential equation example—each node is labeled with the tuple (depth, mobility). (b) ASAP, ALAP, mobility, and depth value of each node. (c) Initial ready set. (d) Scheduling in cycle 1. (e) Final scheduling.

to the same unit as its corresponding node in the template. For instance, if (a, u) and (b, v) are multiple instances of template E, and if (a, u) has already been mapped to (Unit1, Unit2), and if b is a ready node, then a high priority is given to assigning b to Unit1.

The proposed list-based algorithm is shown in Fig. 1.

B. Examples

We explain this algorithm for the case when the number of voltage levels is two with the help of the DIFFEQ example (Fig. 2). The data flow graph consists of both addition and multiplication, as shown in Fig. 2(a). Our objective is to schedule the nodes among two multipliers (one 3.3-V and one 5-V multiplier) and two adders (one 3.3-V and one 5-V adder). Here, the delay of a multiplier is the same as the delay of an adder. Also, the input data streams are uncorrelated and thus the transition activities at the inputs of the functional units are 0.5. The ASAP, ALAP, depth, and mobility values have been listed in Fig. 2(b). The initial ready set is shown as Fig. 2(c). There are four multiplication nodes in the ready set of cycle 1. Nodes 6 and

8 have mobility ≥ 1 and can be assigned to 3.3-V multipliers. Since there is only one 3.3-V multiplier, and node 6 has a higher priority due to its higher depth value, node 6 is assigned to the 3.3-V multiplier. Since there is only one 5-V multiplier, priority has to be determined for the three remaining nodes, i.e., nodes 1, 2, and 8. Of these nodes, nodes 1 and 2 have high priority, and either can be assigned to the 5-V multiplier. If node 1 is assigned to the 5-V multiplier, then the remaining nodes, node 2 and node 8, are moved to the ready set of cycle 2. The ASAP and ALAP values are updated and the procedure repeated for the next control cycle. The scheduling after cycles 1 and 2 are given in Fig. 2(d) and the final scheduling result is given in Fig. 2(e). From the results, we see that since five nodes have been assigned to 3.3-V resources, power consumption has been significantly reduced. One (3.3 V \rightarrow 5 V) level shifter is used since node 6 is assigned to a 3.3-V multiplier and node 7 is assigned to a 5-V multiplier.

We use the same DIFFEQ example to illustrate the algorithm for three voltage levels in Fig. 3. The resource constraint consists of three multipliers (5 V, 3.3 V, 2.4 V) and three adders (5 V, 3.3 V, 2.4 V). The delay of a multiplier is the same as the delay of

Control cycle	Multiplication nodes	Addition nodes
1	1,2,6,8	10
2	3,7	9,11
3	-	4
4	-	5

(a)

Control cycle	M _{2,4}	M _{3,3}	M ₅	A _{2,4}	A _{3,3}	A ₅
1	8	6	1	10		
2	8'	6'		10'		
3	8''			10''		

(b)

Control cycle	M _{2,4}	M _{3,3}	M ₅	A _{2,4}	A _{3,3}	A ₅
1	8	6	1	10	-	-
2	8'	6'	2	10'	-	-
3	8''	7	3	10''	-	-
4	-	7'	-	9	11	4
5	-	-	-	9'	11'	5
6	-	-	-	9''	-	-

(c)

Fig. 3. Resource-constrained scheduling. Example DIFFEQ for the case when there are one 2.4-V multiplier, one 3.3-V multiplier, one 5-V multiplier, one 2.4-V adder, one 3.3-V adder, and one 5-V adder. (a) Initial ready set. (b) Scheduling in cycle 1. (c) Final scheduling.

an adder and the input data streams are uncorrelated. The ASAP, ALAP, depth, and mobility values are the same as before. The initial ready set is given in Fig. 3(a). Of the four multiplication nodes in the ready set of cycle 1, only node 8 has mobility $>=2$, and thus, is assigned to the 2.4-V multiplier. Node 6 has mobility $>=1$ and is assigned to the 3.3-V multiplier. Nodes 1 and 2 have the same depth, mobility, and switched-capacitance values, and either can be assigned to the 5-V multiplier. We assign node 1 to the 5-V multiplier. The ready set of cycle 2 consists of only node 2, which is assigned to the 5-V multiplier. The ready set of cycle 3 consists of nodes 3 and 7. Since node 7 has mobility $>=1$, it is assigned to the 3.3-V multiplier and node 3 is assigned to the 5-V multiplier.

The scheduling after cycle 1 is given in Fig. 3(b) and the final scheduling result is given in Fig. 3(c). In this schedule, two level-shifters are required: a 3.3 \rightarrow 5-V shifter and a 2.4 \rightarrow 3.3-V shifter.

C. Benchmarks

We present the results obtained by running our algorithm on some high-level synthesis benchmarks. The algorithms were implemented in MATLAB. Fig. 4 describes the power savings results when exact values of power consumption of the resources (multipliers, adders and level shifters) are used. The power models of [6] that consider the capacitance, and the switching activity have been used. Note that, while for the DIFFEQ and AR filter example, the percentage reductions with the level shifter are comparable to that without the level shifter, for the EW filter, the difference in the percentage reduction is not significant. Thus, the area/delay/power overhead of level shifters is relatively small.

V. LATENCY-CONSTRAINED SCHEDULING

In this section, we first describe the proposed latency-constrained scheduling algorithm and then present the results for

Bench-mark	Resource Constraint				Timing Constr. (cycles)	% Reduction	
	M _{3,3}	M ₅	A _{3,3}	A ₅		Without Level Shifter	With Level Shifter
DIFF EQ	1	1	1	1	6	18.90	18.79
	2	2	2	2	5	28.24	28.14
AR Filter	1	1	1	1	16	10.57	10.50
	2	2	2	2	11	14.08	14.00
EW Filter	1	1	1	1	20	14.16	13.94
	2	2	2	2	17	14.16	13.94

Fig. 4. Resource-constrained scheduling results for DIFFEQ, AR, and EW filter when exact values of power consumption from [8] are used. Switching activity is 0.5 for all the input nodes.

some HLS benchmark examples. Since the timing constraint is a hard constraint, low voltages can only be assigned to nodes with non-zero mobility. The proposed algorithms try to assign low voltages to as many nodes of the DFG as possible. The average switching activity and the capacitance for each node is included in the power-consumption calculations, and thus, nodes with high capacitance or high switching activity are assigned to lower voltages. In addition, the algorithm tries to reduce the number of step-up level shifters. The algorithm is significantly different from those based on integer linear programming, and also computationally less complex. We present the algorithm when the operating voltages are 5 and 3.3 V, or when the operating voltages are 5, 3.3, and 2.4 V.

A. Algorithm

The input to this algorithm is the DFG, the timing constraint, the number of voltage levels and the input data stream, and the output is the voltage assignment of each node. A voltage assignment of 5 V is represented by $f = 0$, a voltage assignment of 3.3 V is represented by $f = 1$, and a voltage assignment of 2.4 V is represented by $f = 2$. First, we compute the as-soon-as-possible (ASAP) value and the as-late-as-possible (ALAP) value

Algorithm_Latency (2 and 3 voltage levels)	
1. Calculate the switching activity at the inputs of each node.	
2. Determine ASAP, ALAP and mobility F for each node.	
3. Form "groups" with nodes having the same ASAP and ALAP times. A group can have a cardinality of 1.	
4. Form a "set" with groups having the same mobility value.	
5. Assign priorities to each member (or group) of each set.	
• The cost of a member (proportional to priority) is	
	$\sum_i n_i * (C_{1,i} \alpha_{i,1} + C_{2,i} \alpha_{i,2} + C_{3,i})$
	where $\alpha_{i,1}$, $\alpha_{i,2}$ are the average switching activities of the input operands of functional unit (or node), and C_i 's are regression model constants obtained from [6]. Order the groups according to their priority, i.e., priority of group $i \geq$ priority of group $i+1$.
• Assign a higher priority to a group having a higher ASAP value in case of conflict.	
6. The cost of a set is the sum of the costs of its members. Process the sets in the order of their costs.	
For mobility value $F = j$, do the following:	
• if $j = 0$, then all the groups in SET_0 are assigned $f = 0$.	
• if 2 voltage levels and $j \geq SET_j $,	
All the groups in SET_j are assigned $f = 1$	
else	
(i) find the possible f assignments such that	
	$\sum_{i=1}^{SET_j(n)} f_i = j \text{ and } f_i \in \{0,1\}$
(ii) For each assignment, compute the power consumption.	
(iii) Choose the assignment with the minimum power.	
• if 3 voltage levels and $j \geq 2 * SET_j $,	
All the groups in SET_j are assigned $f = 2$.	
else	
(i) Find the possible f assignments such that	
	$\sum_{i=1}^{SET_j(n)} f_i = j \text{ and } f_i \in \{0,1,2\}$
	Since the groups are ordered according to their priorities, $f_{i+1} \geq f_i$.
(ii) For each assignment compute the power consumption.	
(iii) Choose the assignment with the minimum power.	

Fig. 5. Algorithm for latency-constrained scheduling.

for each node. This is done using a modified version of the breadth first search (BFS) algorithm. Then we compute the mobility of each node. We form groups with nodes which have the same ASAP and ALAP values. Thus, the nodes in a group have the same mobility. Then we form sets with groups which have the same mobility. Specifically, SET_0 consists of groups with mobility 0, SET_1 consists of groups with mobility 1, etc. Next, we compute the cost of each member of each set (i.e., group) where the cost is a function of the number of nodes, the capacitance of each node, and the switching activity of each node. The cost value is used to determine the priority of a member (high cost implies high priority). A group with a high priority is assigned to a low voltage. In case of conflict, the group with a higher ASAP value is given higher priority. This feature results in fewer level shifters being required. The proposed algorithm is shown in Fig. 5.

Note that steps 1–5 are the same for the two-voltage level assignment and the three-voltage assignment. For the three-voltage level assignment, finding the f assignments in

Node	ASAP	ALAP	Mobility	Group	Set	Cost	
*1	1	1	0	G1	SET ₀ =4	1,346	
*2	1	1	0				
*3	2	2	0	G2			673
4	3	3	0	G3			5
5	4	4	0	G4		5	
*6	1	2	1	G5	SET ₁ =2	673	
*7	2	3	1	G6			673
*8	1	3	2	G7	SET ₂ =2	678	
10	1	3	2				
9	2	4	2				
11	2	4	2	G8		10	

Fig. 6. Example DIFFEQ. ASAP, ALAP, mobility of each node, group, and set formation, and the priority of each group.

step 6 is clearly move complex. Since $f \in \{0, 1, 2\}$, it is not obvious whether $f = 2$ should be assigned to a small number of groups or whether $f = 1$ should be assigned to a large number of groups. In fact, if j is the mobility, then the number of possible assignments that need to be checked is $O(j)$. However, it is sufficient to consider only the f assignments for which $f_{i+1} \geq f_i$. This is because any other assignment would either result in an increase in the number of level shifters or an increase in the power consumed by the level shifter. For instance, if $j = 4$ and $|SET_j| = 6$, then assignments such as $f_1 = f_3 = f_4 = f_6 = 1$, $f_2 = f_5 = 0$ would result in two 3.3 V→5 V level shifters, and an assignment with $f_1 = f_2 = f_3 = f_4 = 1$, $f_5 = f_6 = 0$, would result in one 3.3-V→5 V level shifter, while an assignment with $f_1 = f_2 = 0$, $f_3 = f_4 = f_5 = f_6 = 1$ or $f_1 = f_2 = f_3 = f_4 = 0$, $f_5 = f_6 = 2$ would result in no level shifters.

B. Examples

We explain Algorithm_Latency for the two-voltage level assignment problem with the help of the DIFFEQ example. The DFG is given in Fig. 2(a). Assume that the computation time of each node is the same and that the latency is constrained to 4 (length of the critical path). In Step 2, the ASAP time, ALAP time, and the value of mobility is computed for each node. The results are listed in Fig. 6. In step 3, groups G_1 through G_8 are formed. Note that groups G_1 , G_7 and G_8 have a cardinality of 2, while the other groups have a cardinality of 1. In step 4, sets are formed and the elements in a set are ordered according to their priority. $SET_0 = \{G_1, G_2, G_3, G_4\}$, $|SET_0| = 4$, $SET_1 = \{G_5, G_6\}$, $|SET_1| = 2$, $SET_2 = \{G_7, G_8\}$, $SET_2(n) = 2$. In step 5, we calculate the cost of each group. For instance, the cost of group 1 is $n * (C_1 * \alpha_1 + C_2 * \alpha_2 + C_3) = 1,346$, assuming that the switching activity for the inputs of each node are 0.5. Since G_4 has a higher ASAP value than G_3 , we assign G_4 a higher priority value. Thus the priority of the groups in SET_0 is $G_1 > G_2 > G_4 > G_3$. Similarly, the priority of the groups in SET_1 is $G_6 > G_5$ and the priority of the groups in SET_2 is $G_7 > G_8$. In step 6, assign $f = 0$ to elements of SET_0 , (G_1, G_2, G_3, G_4). For $j = 1$, since $|SET_j| > j$ and $\sum_{i=1}^2 f_i = 1$, assign $f = 1$ to G_6 since it has a higher priority than G_5 . Finally, for $j = 2$, since $|SET_j| = 2$, assign $f = 1$ to all the groups in SET_2 . Thus nodes 1, 2, 3, 4, 5, 6 are assigned to 5 V and nodes 7, 8, 9, 10, 11 are assigned to 3.3 V. The total

Timing Constraint k	Power (pJ) Using 5V	Without level shifter		With level shifter	
		Power (pJ)	% red.	Power (pJ)	% red.
*12	101,630	82,425	18.90	82,529	18.79
13	101,630	53,929	46.94	54,241	46.63
14	101,630	44,431	56.28	44,639	56.08
15	101,630	44,361	56.35	44,569	56.14
18	101,630	44,292	56.42	44,292	56.42

(a)

Timing Constraint k	Power (pJ) Using 5V	Without level shifter		With level shifter	
		Power (pJ)	% red.	Power (pJ)	% red.
*12	101,630	78,877	22.39	78,981	22.28
13	101,630	50,382	50.43	50,694	50.12
14	101,630	37,430	63.17	37,687	62.92
15	101,630	33,907	66.64	34,099	66.44
18	101,630	23,478	76.90	23,577	76.80

(b)

Fig. 7. Differential equation example. Power consumption results for (a) two voltage levels and (b) three voltage levels.

power due to this assignment (including level shifter power) is 82,425 pJ. This is a 18.8% reduction compared to the case when all the nodes are operated at 5 V.

Next, we explain Algorithm_Latency for the three-voltage level assignment problem with the help of the same DIFFEQ example for the case when the latency is 4. Steps 1–5 are identical to the two-voltage level assignment and we will not repeat it here. In step 6, first assign $f = 0$ to elements of SET_0 , i.e., G_1, G_2, G_3, G_4 . When $j = 1, 2 * |SET_j| > j$, and the f_i 's have to be determined such that $\sum_{i=1}^2 f_i = 1$. Since the two groups in SET_1 have the same cost, we consider the power consumed by a level shifter. We assign $f = 1$ to G_6 and $f = 0$ to G_5 , since this assignment has lower power consumption (24,154 pJ compared to 24,258 pJ). When $j = 2, 2 * |SET_j| > j$, and the f_i 's have to be determined such that $\sum_{i=1}^2 f_i = 2$. There are three possible assignments: 1) assign $f = 1$ to both G_7 and G_8 ; 2) assign $f = 2$ to G_7 and $f = 0$ to G_8 ; and 3) assign $f = 0$ to G_7 and $f = 2$ to G_8 . Since the priority of G_7 is higher than G_8 (due to node h being a multiplier), assignment 3) will always result in higher power compared to assignment 2). Next we compute P for 1) and 2). Since $P = 7,492$ pJ for assignment 1) and $P = 4,155$ pJ for assignment 2), we choose assignment 2). Thus, nodes 1, 2, 3, 4, 5, 6, 9, 11 are assigned to 5 V, node 7 is assigned to 3.3 V, and nodes 8 and 10 are assigned to 2.4 V.

C. Benchmarks

In this section, we present the results obtained by running our algorithms on some high-level synthesis benchmarks. The algorithms have been implemented in MATLAB. In all the examples, we assume that the switching activity at the inputs of each

Timing Constraint k	Power (pJ) Using 5V	5V and 3V resources		5V, 3V, and 2.4V resources	
		Power (pJ)	% red.	Power (pJ)	% red.
*20	138,810	100,060	27.92	85,773	38.21
21	138,810	90,768	34.61	76,484	44.90
22	138,810	72,049	48.10	57,764	58.39
23	138,810	62,759	54.79	48,475	65.08
30	138,810	60,612	56.33	32,886	76.31

Fig. 8. AR lattice filter example. Power consumption results (including the power consumption due to level shifters) for two and three voltage levels.

Timing Constraint k	Power (pJ) Using 5V	5V and 3V resources		5V, 3V, and 2.4V resources	
		Power (pJ)	% red.	Power (pJ)	% red.
*27	138,030	119,070	13.74	119,000	13.78
28	138,030	118,690	14.01	118,620	14.06
29	138,030	99,517	27.90	99,375	28.00
30	138,030	99,379	28.00	99,273	28.08
40	138,030	60,244	56.35	32,515	76.44

Fig. 9. EW filter example. Power consumption results (including the power consumption due to level shifters) for two and three voltage levels.

node is 0.5, and that the delay of a multiplier is five times the delay of an adder. The delay figures are consistent with those described in [6]. We obtain the power consumption values of the multipliers, adders, and level shifters from [6]. The results for the *differential equation* benchmark are listed in Fig. 7, those for *AR lattice filter* are listed in Fig. 8, and those for *elliptic wave (EW) filter* are listed in Fig. 9. In each case, we tabulate the results for different timing constraints. The timing constraint marked with a "*" corresponds to the critical path delay. In all three examples, our procedure yielded a very high reduction in power. For instance, for the differential equation three-voltage assignment example, we achieved a 50.44% average reduction. It is to be noted that for large timing constraints, the power reduction is even greater. This is to be expected since for large constraints all the nodes can be operated on at a reduced supply voltage. Also note that the difference in the percentage power reduction with and without the level shifter power being considered is 0.2%–0.4%. The summary of power reduction results for the differential equation, AR lattice filter, and EW filter are presented in Fig. 10.

VI. CONCLUSION

In this paper, we present polynomial time algorithms for 1) resource-constrained scheduling and 2) latency-constrained scheduling for the case when the resources operate at multiple voltages. Both scheduling schemes try to reduce the overall power consumption of the datapath components (including

Benchmark	Timing Constraint, k	% reduction (5V, 3.3V)	% reduction (5V, 3.3V, 2.4V)
DIFFEQ	*12	18.79	22.28
	18	56.42	76.80
AR filter	*20	27.92	38.21
	30	56.33	76.31
EW filter	*27	13.74	13.78
	40	56.35	76.44

Fig. 10. Summary of power reduction results for two and three voltage levels. Values for the minimum timing constraint (marked with a “*”) and 1.5 times the timing constraint have been listed.

power consumed by the level shifters). The resource-constrained scheduling scheme tries to balance the conflicting requirements of reducing the latency and maximally utilizing resources operating at reduced voltages. The latency-constrained scheduling scheme assigns as many nodes as possible to the resources operating at low voltages without violating the timing constraint. In addition, both schemes consider the effect of switching activity on the power consumption of the functional units, the interconnect complexity, and the power consumed by the level shifters. Experiments with HLS benchmarks show that the proposed schemes achieve significant power reduction when the number of voltage levels is two (5 V, 3.3 V) or three (5 V, 3.3 V, 3.4 V).

Another voltage-scaling method that results in significant power savings in the variable-supply voltage method in which the supply voltage of the entire processors core (and not isolated functional units) is controlled adaptively. We are currently trying to integrate the proposed method with the variable-supply voltage method, as in [7].

ACKNOWLEDGMENT

The authors would like to thank S. Raje, J.-M. Chang, and M. Pedram for sharing their work.

REFERENCES

- [1] A. Chandrakasan and R. Brodersen, “Minimizing power consumption in digital CMOS circuits,” in *Proc. IEEE*, vol. 83, Apr. 1995, pp. 498–523.
- [2] A. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, and Brodersen, “Optimizing power using transformations,” *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 12–31, Jan. 1995.
- [3] A. Dasgupta and R. Karri, “Simultaneous scheduling and binding for low power minimization during microarchitecture synthesis,” in *Proc. Int. Symp. Low-Power Design*, Apr. 1995, pp. 69–74.
- [4] A. Raghunathan and N. K. Jha, “Behavioral synthesis for low power,” in *Proc. IEEE Design Automation Conf.*, 1995.
- [5] E. Musoll and J. Cortadella, “High-level synthesis techniques for reducing the activity of functional units,” in *Proc. Int. Symp. Low Power Design*, 1995, pp. 99–104.

- [6] J.-M. Chang and M. Pedram, “Energy minimization using multiple supply voltages,” *IEEE Trans. VLSI Syst.*, vol. 5, Dec. 1997.
- [7] M. Takahashi, M. Hamada, T. Nishikawa, H. Arakida, T. Fujita, F. Hatori, S. Mita, K. Suzuki, A. Chiba, T. Terazawa, F. Sano, Y. Watanabe, K. Usami, M. Igarashi, T. Ishikawa, M. Kanazawa, T. Kuroda, and T. Furuyama, “A 60-mW MPEG4 video codec using clustered voltage scaling with variable supply-voltage scheme,” *IEEE J. Solid-State Circuits*, vol. 33, pp. 1772–1780, Nov. 1998.
- [8] M. C. Johnson and K. Roy, “Datapath scheduling with multiple supply voltages and level converters,” *ACM Trans. Design Automat. Electron. Syst.*, vol. 2, no. 3, pp. 227–248, July 1997.
- [9] S. Raje and M. Sarrafzadeh, “Scheduling with two voltages under resource constraints,” Dept. Elect. Eng. Comput. Sci., Northwestern Univ., Evanston, IL, Tech. Rep., 1995.
- [10] —, “Scheduling with multiple voltages,” *Integration: The VLSI J.*, vol. 23, pp. 37–59, 1997.
- [11] W.-T. Shiue and C. Chakrabarti, “Low power scheduling with resources operating at multiple voltages,” in *IEEE Int. Symp. Circuits and Systems*, vol. 2, June 1998, pp. 437–440.
- [12] Y.-R. Lin, C.-T. Hwang, and A. C.-H. Wu, “Scheduling techniques for variable voltage low power designs,” *ACM Trans. Design Automat. Electron. Syst.*, vol. 2, no. 2, pp. 81–97, April 1997.
- [13] R. Mehra and J. Rabaey, “Exploiting regularity for low-power design,” in *Proc. Int. Conf. Computer-Aided Design*, 1996, pp. 166–172.



Wen-Tsong Shiue (M’97) received the M.S. degree in electrical engineering from Western Michigan University, Kalamazoo, MI, in 1991. He is currently working toward Ph.D. degree in electrical engineering at Arizona State University, Tempe, AZ, in the area of high-level synthesis and memory management for low-power applications in VLSI systems.

From 1991 to 1996, he was with China Airlines, Taoyuan, Taiwan, where he was engaged in system and electronics design. He was also an Instructor at Tamkang University, Taipei, Taiwan, teaching courses in logic design and electronics. His current research interests include low-power memory design, CAD tools for high-level synthesis low-power design, VLSI architectures and algorithms for signal processing and image processing, and parallel processing. He is a member of the Center for Low Power Electronics.



Chaitali Chakrabarti (S’89–M’90) received the B.Tech. degree in electronics and electrical communication engineering from the Indian Institute of Technology, Kharagpur, India, in 1984. She received the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park, MD, in 1986 and 1990, respectively.

She has been with Arizona State University, Tempe, AZ, since fall 1990, where she is now an Associate Professor. Her research interests are in the areas of VLSI architectures and algorithms for signal processing and image processing, low-power system design, high-level synthesis for low power, CAD tools for VLSI, and VLSI design. She is a member of the Center for Low Power Electronics.

Dr. Chakrabarti is currently an Associate Editor for the *IEEE TRANSACTIONS ON SIGNAL PROCESSING* and the *Journal of VLSI Signal Processing Systems*.