Modified Force-Directed Scheduling for Peak and Average Power Optimization using Multiple Supply-Voltages

A. K. Allam and J. Ramanujam
Electrical and Computer Engineering Department, Louisiana State University, USA
(atef, jxr) @ece.lsu.edu

Abstract

In this paper, we consider the problem of peak and average power optimization in high-level synthesis. We focus on the scheduling task under timing constraint using supply voltage scaling since it is considered as the most efficient technique for reducing power consumptions in CMOS circuits. We propose a two-phase heuristic for peak and average power minimization using multiple supply voltages scheduling technique. The first phase is the modified power-force-directed scheduling (MPFDS) heuristic based on the well-known force-directed scheduling technique. The second phase is a post-processing procedure (power-area-saving) that is a revisit of the output schedule from the first phase in order to exploit the available rooms to get more power and/or the operating resources minimization. Results show that our proposed heuristic is capable of achieving near-optimal results with polynomial complexity.

I. Introduction

High-level synthesis (HLS) is the process of mapping the behavioral specification of the system into register transfer description. The outcome of the high-level synthesis is a structural view of the data path and a logical view of the control unit. High-level synthesis involves three main tasks: scheduling, allocation, and binding. The central task is scheduling, which is the process of determining at which control step(s) each operation in the data-flow graph (DFG) executes. We define *Scheduling for Low Power and Energy (SLoPE)* in high-level synthesis as the process of determining at which control step(s), and at what voltage level each operation in the DFG executes with the goal of minimizing power and energy. Although conventional design metrics such as performance, size and testability are important, the most critical design metric nowadays is power. The demand for long-life batteries within tolerable size and weight and the reliability of integrated circuits are the main factors that dictate power-aware design of embedded systems. Reliability of integrated circuits is tightly related to the peak and average power consumption. The main sources of power dissipation in CMOS circuits are the capacitive switching power, P_{sw}, the short-circuit power, P_{sc}, and the power consumption due to leakage current, $P_{leakage}$.

The capacitive switching power together with the short-circuit power is called *dynamic power*, and it is due to charging and discharging in CMOS gate. Dynamic power is considered to be a significant part in the total power consumption and is given by the following equation

$$P_{dynamic} = \tfrac{1}{2}\, \alpha\, C_L\, V_{dd}^2\, f_{clock}, \qquad (1)$$

where C_L is the load capacitance at the gate output, f_{clock} is the circuit clock frequency, V_{dd} is the supply voltage, and α is the average number of transitions per clock cycle at the gate output, referred to as the *switching activity*.

Power/energy reduction in embedded system can be achieved by carefully designing each of its constituent components targeting low power/energy design.

B. Related Work

In recent years, a lot of research work has been done to solve the multiple supply voltages scheduling (MVS) problem. Some of these research works addressed the MVS problem using heuristics [1, 2, 3, 11, 12], while others addressed it using *integer linear programming* (ILP) [3, 4, 5, 6, 8]. Mohanty and Raganathan [8] introduced an ILP based optimization technique for simultaneous minimization of peak and average power using a multiple supply voltages scheme. They introduced two datapath scheduling schemes, one using multiple supply voltages and dynamic clocking and the other using multiple supply voltages and multicycling. Shiue [9] has presented an ILP model and a modified force-directed scheduling (MFDS) heuristic that minimizes peak power under latency constraint considering multicycling and pipelining but he did not consider multiple supply voltages.

The scheduling problem in HLS is a well-known NP-complete problem. A fast and effective heuristic is needed to search for the solution especially when the design space is explored to trade off one objective for another such as power and execution time. The algorithm in this work is a potential solution in this direction. Our MPFDS algorithm is an extension of but is different from the original form of force-directed scheduling introduced by Paulin and Knight [10], in which the basic FDS does not deal with multicycle operations or variable cost for the same operation since the delay and power consumption are different for each voltage level. Moreover, our MPFDS algorithm considers multicycles as well as the variable power and delay of each operation when it is tentatively scheduled with different voltage levels which is different than the work presented by [9].

The rest of this paper is organized as follows. Section 2 introduces the MVS problem targeting peak and average power minimization. In section 3, we develop our MPFDS algorithm followed by the power-resources saving algorithm in Section 4. Section 5 shows the results of some benchmarks to illustrate our proposed solutions. Section 6 concludes with a summary.

II. Problem Definition

The input to the problem includes a DFG representation of the design problem, a set of voltage levels for the operating resources, a power/delay table that contains the average power consumption and the delay time needed for each resource operating on each voltage level and the time constraint, λ. The goal is to find a schedule (in which each operation is stamped to a control step, cstep $\in$ (1, 2, … , λ) and a voltage level from the set of input voltage levels) that minimizes the peak power consumption as well as the average power and energy consumption.

Our solution to the multiple supply voltages scheduling (MVS) problem is a two-phase algorithm. The first phase is a modified power-force-directed scheduling (MPFDS) heuristic based on the force-directed scheduling heuristic introduced by Paulin and Knight [10] targeting power (peak and average) and energy consumption minimization. The second phase is the *power-resources saving* procedure that revisits the output schedule from the first phase in which it tries to further reduce the power and/or the number of operating resources by scheduling DFG operations in lower voltage levels if possible and/or by moving the operations within their new timeframes where possible without violating the peak power obtained from the first phase.

III. Modified Power-Force-Directed Scheduling

The goal of our modified power-force-directed scheduling algorithm (MPFDS) is to minimize power (peak and average) consumption by assigning to each operation the smallest voltage level possible from the input voltage levels without violating the time constraint and at the same time to distributes the DFG operations over the total allowed time in such a way as to balance the power consumptions to achieve minimum peak power. This is achieved by taking into account the global effect of power consumptions in the entire DFG when attempting to schedule an operation in a certain tentative cstep within its timeframe (The time-frame of an operation is the set of csteps that start at its earliest time, ASAP, and end at its latest time, ALAP) and at a certain voltage level.

A distribution graph called *the power distribution graph*, pDG, is constructed to represent the probabilistic power consumption at each cstep taking into consideration the different voltage levels that an operation can be assigned. The power distribution graph, pDG, at each cstep j is given by:

$$pDG(j) = \sum_{i} \sum_{v} prob(i, j, v) * p(i, v), \qquad (2)$$

where $p(i, v)$ is the power consumed by the resource executing operation i when operating at voltage level v, and $prob(i, j, v)$ is the probability of an operation i to be scheduled at cstep j with voltage level v. Because we deal with multicycle operations, we refer to the last cycle of the operation when we talk about scheduling the operation at a certain cstep j and at the same time all other cycles of the operation are considered at csteps $j-1, j-2, \ldots, j-d(i,v)+1$. Thus the probability, $prob(i, j, v)$, is computed in such a way as to capture the possibility that an operation i is active in csteps $j, j-1, j-2, \ldots, j-d(i,v)+1$ and the possibility that there is room for operation i to be scheduled with voltage level v. It is computed as the reciprocal of its mobility multiplied by the number of voltage levels with which it can be scheduled; where the mobility of a node i is the length of its time-frame and equals the difference between its ALAP and ASAP time steps plus one. For example, consider a multiplication operation i using the modules library in Table 1 with ASAP/ALAP times are *2/5*. Therefore, its mobility is 4 and its probability equals 1/(4*2) = 1/8 at csteps 2, 3, 4, 5, and 6 for both v_1 and v_2 and zero elsewhere, as shown in Figure 1-(a). If the ASAP/ALAP times of operation i changed to be 2/3, its mobility is 2 and its probability is 1/(2*1) at csteps 2, 3, and 4 for voltage level v_1 and zero for v_2 (there is no room to be scheduled with voltage level v_2) as shown in Figure 1-(b).

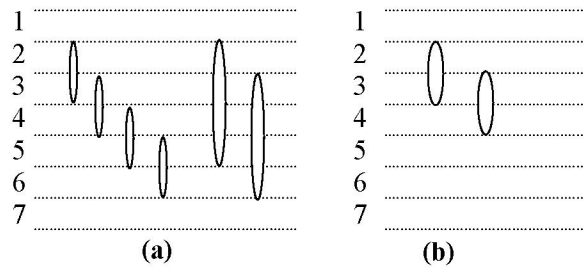

Figure 1: Potential schedule of sample operation: **(a)** ASAP/ALAP is 2/5, **(b)** ASAP/ALAP is 2/3.

Our MPFDS algorithm works as shown in Figure 2. First, the ASAP and ALAP times for each operation are computed using the delay of the corresponding functional unit when it operates at the highest voltage level followed by computing the probability of each operation using the way discussed previously. Then a power distribution graph is constructed using Equation (2). The main loop of the algorithm is the one used to compute the resultant forces when an operation is tentatively scheduled at a cstep within its time-frame and at a possible voltage level. These forces are calculated in such a way as to balance the power

distribution over all csteps and at the same time schedule operations with the smallest power possible taking into consideration the global effect when doing so. The total force is the sum of two forces, self force accounting for the effect of power consumption by an operation *i* when it is tentatively scheduled at cstep *j* and voltage level *v*; and the predecessor/successor force, psforce, accounting for the power consumption of predecessors/successors of operation *i* due to the change of their time-frames. Equations (3) and (4) show how these forces are calculated and how the effect of power consumption is considered. Finally, the operation with the smallest total force is picked and stamped to the corresponding cstep and the corresponding voltage level. Then, the time-frame for each operation is updated and the process is repeated until all operations are scheduled.

$$selfForce(i,j,v) = \sum_{j_2=j-delay(i,v)+1}^{j} pDG(j_2) * p(i,v) - \sum_{j_1}\left(pDG(j_1) \sum_{vv} prob(i,j_1,vv) * p(i,vv) \right) \quad (3)$$

and

$$psForce(i,j,v) = \sum_{k \in ps(i)} \left(\sum_{j_4}\left(pDG(j_4) \sum_{vv} newprob(k,j_4,vv) * p(k,vv) \right) - \sum_{j_3}\left(pDG(j_3) \sum_{vv} prob(k,j_3,vv) * p(k,vv) \right) \right) \quad (4)$$

where $j_1 \in$ [ASAP(i)+d(i,v_1)-1, ALAP(i)], $j_3 \in$ [ASAP(k)+d(k,v_1)-1, ALAP(k)], and $j_4 \in$ [newASAP(k)+d(k,v_1)-1, newALAP(k)]; and *newASAP*, *newALAP*, and *newProb* are the ASAP, ALAP, and probability of predecessors/successors of operation *i,* respectively, due to the change in their time-frames.

Repeat until *(all operations are scheduled)*

1. *Calculate ASAP and ALAP times.*
2. *Update power distribution graph (pDG) using Equation (2).*
3. *For each operation that is not scheduled yet, calculate self force and predecessor/successor forces for each voltage-level v and at each tentative cstep using Equations (3) and (4) respectively.*
4. *Add the computed self forces and predecessor/successor forces together to form the total forces.*
5. *Schedule operation with the lowest total force at the corresponding cstep using the associated voltage-level.*

End Repeat

Figure 2: MPFDS algorithm.

A. Time Complexity of MPFDS

The worst-case time complexity of our modified power force directed scheduling algorithm (MPFDS) is in the order of $O(\lambda V^2 n^3)$, where λ is the total time constraint, V is the number of input voltage-levels, and n is the number of operations in the DFG. In practice, because the usual number of operating voltage-levels in a circuit is two or three, this time complexity is reduced to $O(\lambda n^3)$ same as the basic FDS algorithm. Here is the derivation for this time complexity.

1. There is at least one operation scheduled in each iteration. Since scheduling an operation affects mobility of other operations forcing their time-frames to be ones (same ASAP and ALAP values for each operation), they are scheduled in the same iteration too. Thus, there are at most n iterations.
2. At each iteration there are at most n unscheduled operations that must be considered for force calculations.
3. Forces for each of these unscheduled operations are calculated for each potential voltage-level and for each cstep within its time-frame, which requires $O(\lambda Vn)$ calculations. Because the time-frame for an operation is at most λ and the number of potential voltage-levels is at most V (a voltage-level is excluded for an operation when there is no room for that operation to be scheduled using that voltage-level).
4. For each potential voltage-level and for each tentative cstep of an operation to be scheduled at, there are at most n-1 predecessors/successors affected. This requires $O(Vn)$ calculations because their forces need to be calculated for each voltage-level.

IV. Power- Resources Saving

The goal of the second phase of the algorithm, power-resources saving procedure, is to gain additional power and/or resources saving through exploiting any available flexibility for an operation. It tries to schedule the DFG operation with a lower voltage level (more power saving) and/or to move it up and down within the available room without violating the peak power obtained from the first phase, to get more peak power saving and/or resources saving. The algorithm for power-resources saving is shown in Figure 3. The inputs to the algorithm are the resultant schedule attributes from the first phase (MPFDS) where *scheduleStep* and *vLevel* are the cstep and the voltage-level stamped to each operation, respectively, and peak_power is the resultant peak power consumption from the first phase.

A. Time Complexity of Power-Resources Saving

The power-resources saving algorithm has a worst-case time complexity in the order of $O(n^2)$, where n is the number of operations in the DFG. Following is a derivation for this time complexity.

- The core of the algorithm inside the first "for-loop" is executed exactly n times (once for each operation)

since only unmarked operations are subject to the computations inside that "for-loop". These n iterations are the summation of the iterations through both "while-loop" and "for-loop".

- At each iteration, ASAP and ALAP routines are executed to update the time-frames of the DFG operations. This is done with $O(n)$ time complexity.
- The potential reduction in peak power and /or in resources is examined for each potential voltage-level and for each cstep within the time-frame of unmarked operation at hand. This requires $O(\lambda V)$ calculations where λ is the total time constraint and V is the number of input voltage-levels. This is because the time-frame for an operation is at most λ and the number of potential voltage-levels is at most V (a voltage-level is excluded for an operation when there is no room for that operation to be scheduled using that voltage-level).
- In each iteration, the complexity is determined by the maximum of $O(n)$ and $O(\lambda V)$. In practice, the usual number of operating voltage-levels in a circuit is two or three and the time-fame of an operation is much smaller than λ, therefore, the complexity of steps 2 and 3 is $O(n)$.

Power-resources saving*(scheduleStep, vLevel, peak_power)*
marked = 0 for all operations.
count = 0
while(count < numOperations) do
  for(op = 1: numOperations)
    1. *if (marked(op)=1 or indegree(op) > 0) skip the rest of loop body.*
    2. *update the time frames*
    3. *compute the room of op using scheduleStep of its predecessors and its successors.*
    4. *for(v = numVlevels:1 step -1)*
      4.1 if (there is a room to schedule op with v without violating the peak power and the input resource constraints if any)
        4.1.1 schedule op at cstep within its time frame to get smaller power and/or resources and set:
        4.1.2 marked(op) = 1.
        4.1.3 vLevel(op) = v.
        4.1.4 scheduleStep(op) = cstep.
        4.1.5 count = count + 1.

Figure 3: Power-resources saving algorithm.

V. Experimental Results

Our presented algorithm is tested on standard benchmarks like HAL, ARF, and EWF using the module library in Table 1. The Experiments take place on the *SUN ENTERPRISE 4500* workstation. This workstation has eight 333MHz SPARC CPU's and 2GB RAM, and it works with SOLARIS 8 operating system. Table 2 shows peak power, average power, and the number of resources for each benchmark with time constraint varying from the critical path length to twice the critical path length after each phase of the algorithm. Results are compared to the optimal solution (ILP solution) as tabulated in Table 2 showing that in most cases the results of our heuristic well match those obtained from the optimal solution especially for the time constraints located between the critical path length and around 1.5 times the critical path length. In some cases, the number of resources resulting from the optimal solution is more than that obtained from the MPFDS heuristic because the ILP solution is a time-constrained scheduling and does not take into consideration resource minimization, while phase II of our algorithm exploits any opportunity to get a smaller number of resources. Table 2 also shows the benefits of post processing the output of the MPFDS heuristic with the power-resources saving procedure in both power and resource minimization.

In many cases, the power-resources saving algorithm brings the output of MPFDS back to match the optimal solution or to be within a small error for average and/or peak power and it reduces the number of resources required even when there is no room for more power reduction.

Table 1: Modules library for MVS

Module	5.0 V		3.3 V	
	d	power	d	power
MULT16	2	84	4	13
ADD16	1	26	2	6
SUB16	1	26	2	6

Results for the power-resources saving algorithm in Table 2 are obtained with no constraint on the resources obtained from the first phase. In some cases, the resultant resources after the power-area saving might get higher than that obtained from the MPFDS phase to get more power reduction as in the HAL benchmark under time constraint 9, the ARF benchmark under time constraint 16, and the EWF benchmark under time constraint 18 in Table 2. The power-resources saving algorithm can eliminate this problem by keeping the resources obtained from the MPFDS phase inviolated in the power-resources saving post processing. This is achieved in the cost of less power savings in some cases.

VI. Conclusion

In this paper, we have again considered the problem of peak and average power minimization in the scheduling in HLS with multiple supply voltages under time constraint. We have proposed a two-phase heuristic to get a near-optimal solution in a small amount of time. The first phase is a modified power-force-directed scheduling (MPFDS) heuristic based on the known

basic force-directed scheduling followed by the second phase, power-resources saving that is a post-processing of the output schedule from the first phase to exploit the available room to get more power and/or operating resource minimization. Results show that our proposed heuristic is capable of achieving near-optimal results with polynomial time complexity.

Table 2: Peak and average power results for ILP solution and the two-phase heuristic

benchmark	L	ILP			MPFDS			Power_area saving		
		avg	peak	[*,+]	avg	peak	[*, +]	avg	peak	[*, +]
HAL	6	162.33	265	4,3	166	265	4,3	162.33	265	4,3
	7	122.57	181	3,3	125.71	187	3,3	124.14	181	3,3
	8	78.25	110	4,3	94.12	181	3,3	92.75	181	3,3
	9	55.44	97	4,2	56.67	116	3,3	55.44	97	4,3
	10	39.4	45	3,3	40.5	49	3,2	40.5	46	3,2
	11	34.82	39	3,3	34.82	45	3,3	34.82	45	3,3
	12	31	39	3,3	31.92	39	3,3	31	39	3,3
ARF	11	225.27	362	6,3	227.27	375	7,4	225.27	362	6,3
	15	103.33	194	6,4	103.33	207	7,4	103.33	194	6,4
	16	95.5	194	8,3	96.87	200	8,2	96.19	194	8,3
	18	59.11	65	5,4	72	194	6,3	71.4	194	6,3
	22	45.36	52	4,2	57.91	194	5,4	56.91	194	4,3
EWF	17	111.65	252	3,5	106.12	265	4,5	104.82	265	4,5
	18	99	181	3,5	93.78	194	4,4	91.33	194	4,5
	21	62.24	110	3,5	66.7	116	3,5	65.67	116	3,5
	28	30.7	37	2,5	35.25	97	2,3	34.86	97	2,3
	34	22.7	26	2,4	29	84	2,3	28.38	84	2,3

References

[1] L. Wang, Y. Jiang, and H. Selvaraj, "Synthesis Scheme for Low Power Designs with Multiple Supply Voltages by Heuristic Algorithms," in *Proc. ITCC*, pp. 826-829, 2004.

[2] J. Chang and M. Pedram, "Energy Minimization Using Multiple Supply Voltages," *IEEE Trans. on VLSI Systems*, vol. 5, no. 4, pp. 436–443, Dec. 1997.

[3] Y.-R. Lin, C.-T. Hwang, and A. C.-H. Wu, "Scheduling Techniques for Variable Voltage Low Power Designs," *ACM Trans. on Design Automation and Electronic Systems*, vol. 2, no. 2, pp. 81–97, Apr. 1997.

[4] W.-T Shiue and C. Chakrabarti, "ILP-Based Scheme for Low Power Scheduling and Resource Binding," in *Proc. IEEE Intl. Symp. on Circuits and Systems*, pp. 279--282, May 2000.

[5] A. Manzak and C. Chakrabarti, "A Low Power Scheduling Scheme with Resources Operating at Multiple Voltages." *IEEE Trans. on VLSI Systems*, vol. 10, No 1, pp. 6-14, Feb. 2002.

[6] N. Chabini, I. Chabini, E.-M. Aboulhamid, and Y. Savaria, "Methods for Minimizing Dynamic Power Consumption in Synchronous Designs with Multiple Supply Voltages," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 22, no. 3, pp. 346-351, Mar. 2003.

[7] S. Gupta and S. Katkoori, "Force-Directed Scheduling for Dynamic Power Optimization," in *Proc. IEEE Computer Society Annual Symposium on VLSI*, pp. 68-73, 2002.

[8] S. P. Mohanty, N. Ranganathan, and S. K. Chappidi, "Simultaneous Peak and Average Power Minimization During Datapath Scheduling for DSP Processors," in Proc. *ACM Great Lakes Symposium on VLSI*, pp. 215-220, 2003.

[9] W.-T. Shiue, "High Level Synthesis for Peak Power Minimization Using ILP," in *Proc. IEEE Int'l Conf. on Application Specific Systems, Architectures and Processors*, pp. 103–112, 2000.

[10] P. G. Paulin and J. P. Knight, "Force Directed Scheduling for the Behavior Synthesis of ASICs," *IEEE Transactions on CAD*, vol. 8, pp. 661-679, June 1989.

[11] W.-T. Shiue and C. Chakrabarti, "Low Power Scheduling with Resources Operating at Multiple Voltages", *IEEE Transactions on Circuit and Systems Part II: Analog and Digital Signal Processing*, vol. 47, no. 6, pp. 536-543, June 2000.

[12] B. Radhakrishnan and M. Venkatesan, "Multiple Voltage and Frequency Scheduling for Power Minimization," in *Proc. Euromicro Symp. on Digital System Design*, pp. 279-285, Sep. 2003.