An O(N) Supply Voltage Assignment Algorithm for Low-Energy Serially Connected CMOS Modules and a Heuristic Extension to Acyclic Data Flow Graphs

Abdulkadir Ut peanut, Yuwraj Singh Dhillon, Kyu-won Choi, and Abhijit Chatterjee
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332
USA
{utku, yuvrajsd, kwchoi, chat}@ece.gatech.edu

Abstract

In this paper, a novel algorithm is proposed for assigning supply voltages to serially executing functional units (FUs) in a digital system such that the overall dynamic energy consumption is minimized for a given timing constraint. Novel closed form expressions for optimum supply voltage values are presented. The computation time of the algorithm is O(N) for N FUs in series. An extension of the O(N) algorithm is proposed for optimizing the acyclic data flow graph associated with any given task. Given the number of FUs available for the task, the operations required for the task are scheduled on the FUs. Voltages are then assigned to the FUs on each path of the flow graph using the O(N) algorithm. Energy savings of 10-60% are achieved on DSP filter designs using the proposed high-level optimization methodology over single supply voltage designs.

1. Introduction

Energy consumption is well-recognized as one of the most important parameters in designing modern portable electronic systems. Supply voltage is the major parameter affecting energy consumption and judicious use of different supply voltages for different modules in a system can lead to huge savings in energy consumption. Some previous work in this area presumes technology that allows dynamic voltage scaling while the system is operating [8, 9, 10, 11], but this is really not feasible - for present systems, at least - without huge overheads in system complexity and even energy. If proper profiling of the inputs to the system is performed, then using multiple static supply voltages give significant energy consumption reductions [2, 3, 12]. There are some practical problems in using multiple supply voltages such as routing multiple supply voltage lines and area/delay overhead of level shifters. As shown in [12], the area/delay overhead of level shifters is relatively small.

In this paper, we present a novel algorithm for assigning delay and supply voltage values for serially-connected modules that minimizes energy consumption. We also present a heuristic extension of this algorithm for arbitrary data flow graphs (DFGs) under given resource constraints (i.e. number of functional units (FUs)). We assign delay values to the functional units such that the overall energy consumption is minimized. The inputs to our optimization algorithm are the behavioral description of the digital system; number of available functional units; and characteristics for the functional units such as delay and energy consumption values for a fixed supply voltage. The scheduling and allocation of the operations onto the available resources is done using conventional ASAP scheduling [6, 7].

There has been previous work on using multiple supply voltages to reduce energy consumption. In [3], the voltage/delay assignment problem is formulated as an integer linear programming (ILP) problem. For large systems with tight constraints on timing and resources, this method might not be scalable.

In [4], a datapath scheduling algorithm using dynamic frequency clocking is presented. A heuristic scheduling algorithm is used in which more energy consuming operations are scheduled on lower frequency FUs. The heuristic is based on empirical observations about energy dissipation patterns of FUs whereas we use energy and delay models of CMOS modules to mathematically prove a minimum energy condition that helps us find optimum delays for FUs in sequential DFGs. We then use this to generate low-energy schedules for arbitrary DFGs on a path-by-path basis.

In [5], a variable voltage task-scheduling algorithm is proposed for sequentially executing tasks to minimize total energy. They distribute the available slack over all the tasks using an iterative algorithm. In contrast, our approach directly assigns delays to the FUs from a path-by-path analysis of the data flow graph. This approach allows very large problems to be solved giving optimal or near-optimal solutions.
The paper is organized as follows. In Section 2, we propose a delay assignment algorithm for modules in a digital system that gives minimum energy consumption, while meeting the timing constraints. We propose that optimum energy consumption is achieved when the delay of each module is proportional to the energy consumption of that module. In Section 3, we prove this idea mathematically for the general case of N modules connected in series. In Section 4, we show experimental validation of our idea on a simple digital system. Section 5 describes how the idea can be extended to complicated systems. Section 6 discusses the results. Finally, Section 7 concludes.

2. The Energy-Delay Ratio (EDR) Paradigm

2.1. Theoretical Background

In order to formulize the optimization problem, we obtain approximate equations for delay and energy dissipation for a module in terms of supply voltage. For a CMOS circuit, delay can be approximated as being proportional to $\frac{k_0 \cdot V_i}{(V_i - V_{Th})^\alpha}$ [1].

$$d_i = \frac{k_0 \cdot V_i}{(V_i - V_{Th})^\alpha} \quad (1)$$

Here $d_i$ is the delay of the $i^{th}$ module, $k_0$ is a constant for the module, $V_i$ is the power supply voltage applied to the module, $V_{Th}$ is the threshold voltage ($V_{Th}$ is assumed constant for all modules), and $\alpha$ is the velocity saturation coefficient. The delay constant ($k_0$) includes the effects of process, device sizes, load capacitance, and gate depth in that module.

Having formed an estimate equation for the delay in terms of supply voltage, we now form an equation for the energy consumption in terms of supply voltage. In this paper, we consider only the dynamic energy consumption. Then we can write the energy consumed in a module as:

$$E_i = 0.5 \cdot C_i \cdot V_i^2 \quad (2)$$

Here $C_i$ is the term for all the capacitances that are switched during operation of the $i^{th}$ module including possible multiple switching of some nodes. $V_i$ is the supply voltage for the module as before. Then we can write energy as:

$$E_i = k_{1i} \cdot V_i^2 \quad (3)$$

where $k_{1i}$ stands for the circuit, process, and application dependent terms including switching activity. An average value for total switching activity in the module can be found by running several different tasks on the module and averaging the switching activity results.

Given these approximate models for delay and energy in terms of supply voltage, we can state the energy-optimization problem for a digital system consisting of N modules in series as follows:

Minimize $\sum_{i=1}^{N} E_i$ under the constraint $\sum_{i=1}^{N} d_i = T$

where $E_i$, $d_i$ are the energy consumption and the delay of the $i^{th}$ module for the task under consideration, $T$ is the deadline for the task, and the variables are the supply voltages for each module.

We propose an optimum solution to this problem in Section 2.2.

2.2. EDR Paradigm

Given a digital system consisting of N modules in series, the lowest system energy consumption for a particular task is achieved when the following holds:

$$\frac{E_1}{d_1} = \frac{E_2}{d_2} = \ldots = \frac{E_N}{d_N} = k \quad (4)$$

where $E_i$, $d_i$ are the energy consumption and the delay of the $i^{th}$ module, $k$ is a constant and $\sum_{i=1}^{N} d_i = T$, the deadline for the task.

3. Derivation of Optimal Delay/Voltage Values for Minimum Energy

In this section, closed-form expressions for optimum supply voltages are derived as opposed to iterative methods reported in [5]. Consider a system of N modules in series (See Figure 1). The total energy consumed by the system is given by:

$$E_{total} = k_{11} \cdot V_1^2 + k_{12} \cdot V_2^2 + \ldots + k_{1N} \cdot V_N^2 \quad (5)$$

Total delay of the system is given by:

$$\sum_{i=1}^{N} d_i = \frac{k_{01} \cdot V_1}{(V_1 - V_{Th})^\alpha} + \frac{k_{02} \cdot V_2}{(V_2 - V_{Th})^\alpha} + \ldots + \frac{k_{0N} \cdot V_N}{(V_N - V_{Th})^\alpha} \quad (6)$$

We want to minimize $E_{total}$ subject to the constraint $\sum_{i=1}^{N} d_i = T$. For this, we construct the auxiliary function $G(V_1,V_2,,..,V_N,\lambda)$ given below:

$$G(V_1,V_2,,..,V_N,\lambda) = \sum_{i=1}^{N} E_i - \lambda \left( \sum_{i=1}^{N} d_i - T \right) \quad (7)$$

For minimum energy consumption, we have the following:

$$\frac{\partial G(V_1,V_2,,..,V_N,\lambda)}{\partial V_i} = 0 \quad \text{for all } i. \quad (8)$$
Also, 
\[
\frac{\partial G(V_1, V_2, \ldots, V_N, \lambda)}{\partial \lambda} = 0
\]  
(9)

Eq. (8) becomes 
\[
\frac{\partial}{\partial V_i} \left( \lambda \cdot \frac{d_i}{d} \right) = 0
\]  
(10)

Using Eq. (1) and Eq. (3) in Eq. (10), we get 
\[
2k_i \cdot V_i = \lambda \cdot k_{bi} \cdot \frac{(V_i - V_n)^{\alpha} - \alpha \cdot (V_i - V_n)^{\alpha-1} \cdot V_i}{(V_i - V_n)^{2\alpha}}
\]  
(11)

Simplifying Eq. (11), we get 
\[
2k_i \cdot V_i \cdot (V_i - V_n)^{\alpha+1} = -\lambda \cdot k_{bi}
\]  
(12)

Using Eq. (1) and Eq. (3) in Eq. (12), we get, 
\[
2 \cdot \frac{E_i}{d_i} \left( \frac{V_i - V_{th}}{((\alpha-1) \cdot V_i + V_n)} \right) = -\lambda
\]  
(13)

The above equation can be approximated as 
\[
\frac{E_i}{d_i} = k
\]  
for every module with very little error for \( V_i > 4V_{th} \) (keeping \( V_i > 4V_{th} \) is a standard industry practice) because the term \( \frac{V_i - V_{th}}{((\alpha-1) \cdot V_i + V_n)} \) becomes nearly equal to \( \frac{1}{\alpha - 1} \) for all modules. This proves the EDR Paradigm stated in Section 2.2.

Also, when \( V_i > 4V_{th} \), the delays for the modules can be written as:
\[
d_i = k_{bi} \cdot (V_i)^{\alpha-1}
\]  
(14)

The benefit of using Eq. (14) for the delay is that it yields a closed-form expression for the optimum supply voltage values for the modules. This improves the solution time as opposed to the iterative solution proposed in [5].

The constant \( k \) in Eq. (4) can be obtained as follows:
Let \( E_{i(0)}, d_{i(0)} \) be the initial values of energy and delay for each module, respectively. Compute \( C_i = E_{i(0)} \cdot d_{i(0)}^{\alpha-1} \) for all i. Using Eq. (3) and Eq. (14), \( C_i = k_{bi} \cdot d_{i(0)}^{\alpha-1} \cdot E_{i(0)} \).

Hence, \( C_i \) is independent of \( V_i \). Let \( E_i, d_i \) be the optimum values for energy and delay for the \( i \)th module. Then,
\[
E_i = E_{i(0)} \cdot d_{i(0)}^{\alpha-1} = E_{i(0)} \cdot d_i = k
\]  
(15)

For any \( i \), \( E_i = k \cdot d_i \) and \( C_i = E_i \cdot d_i^{\alpha-1} \). Therefore,
\[
d_i = \left( \frac{C_i}{k} \right)^{\alpha-1}
\]  
(16)

From Eq. (16) and since \( \sum_{j} d_j = T \), we get:
\[
\sum_{j} \left( \frac{C_j}{k} \right)^{\alpha-1} \cdot \frac{1}{\alpha-1} = T \Rightarrow \frac{1}{\alpha-1} \cdot \sum_{j} \left( \frac{C_j}{k} \right)^{\alpha-1} \cdot \frac{1}{\alpha-1} = \frac{C_j}{k}
\]  
(17)

From Eq. (16) and Eq. (17), we now get the optimum values for delays as:
\[
d_i = \left( \frac{C_i}{k} \right)^{\alpha-1} = \frac{T \cdot C_i}{\sum_{j} C_j^{\alpha-1}}
\]  
(18)

The optimum supply voltage values are then given by:
\[
V_i = \left( \frac{k_{bi}}{d_i} \right)^{\frac{1}{\alpha-1}} \cdot \left( \frac{1}{\sum_{j} C_j^{\alpha-1}} \right)^{\frac{1}{\alpha-1}}
\]  
(19)

We see that optimal voltages for the \( N \) modules can be found in \( O(N) \) time.

The above proof applies to the case where only the supply voltages of the circuit modules are used to modulate the delays of individual modules while keeping the total delay constant. In general, if in addition to supply voltage, the device sizes and the threshold voltage are also modulated for energy, the equations involved become too complex to allow a simple closed-form solution. In the following section, the EDR paradigm is validated on a test circuit.
4. Experimental Validation of the EDR Paradigm

To verify the idea presented in Section 3, we used the HSPICE circuit simulator (TSMC 0.25\(\mu\)m, Level 49). We considered two modules, a 4-bit adder and a 4-bit multiplexer (Figure 2) in series. For each simulation run, we assigned different delays (using different supply voltages) to each module keeping the total system delay constant at 0.9ns. A level-shifter was used between the modules to guarantee proper operation at the two different supply voltages. The individual module energies and total energy consumed in each case was computed.

To see the effect of delay assignment on the energy, we define the following metrics:

\[
\text{Simple Metric} = \frac{E_{\text{max}}/d_{\text{adder}}}{E_{\text{adder}}/d_{\text{adder}}} = \frac{V_{\text{max}}-V_{\text{th}}}{(\alpha-1)\cdot V_{\text{max}} + V_{\text{th}}} \tag{20}
\]

\[
\text{Complex Metric} = \frac{E_{\text{max}}/E_{\text{adder}}}{d_{\text{max}}/d_{\text{adder}}} = \frac{V_{\text{adder}}-V_{\text{th}}}{(\alpha-1)\cdot V_{\text{adder}} + V_{\text{th}}} \tag{21}
\]

As Eq. (13) shows, the complex metric should be 1 at the minimum energy consumption point. In Figure 3, we plot total energy consumption versus metric values. We see that the minimum energy point is indeed near the metric values of 1 and that the curves corresponding to the two metrics are very close at this point. This is the reason we can use the simple metric in our formulation of the EDR Paradigm (Eq. 4), instead of the complex metric. This also validates our idea that the total energy consumption of the entire circuit is minimized when the energy consumptions and delays of the individual modules are proportional.

5. DSP Synthesis Using the EDR Paradigm: An Application

In this section, we propose a new DSP synthesis algorithm that performs energy optimization of complex resource-shared DSP data flow graphs using the EDR paradigm. It is assumed that the different functional units (FUs) can have different supply voltages. We show how the EDR paradigm can be used to synthesize minimal-energy architectures using supply voltage scaling.

In the following, we walk through the steps of the proposed synthesis algorithm for an example application. We describe how the algorithm can be used to optimize DSP flow graphs for energy.

First we obtain the Data Flow Graph (DFG) for the IIR filter shown in Figure 4 with additions and multiplications as nodes of the graph. The constraints for the optimization are total execution time, number of multipliers, and number of adders. Let us assume that 2 adders (A0 and A1) and 2 multipliers (M0 and M1) are available for the filtering. It is assumed initially that the unoptimized multiplier takes 2 units of time and the adder takes 1 unit of time. Note that those delay values are used to jump start the algorithm and are modified as the algorithm proceeds. The DFG for the example filter is shown in Figure 5. For simplicity, we consider all switching activities among adders and among multipliers to be the same. In practice, they are evaluated using prior activity simulation. In this example, we assume that adders consume 1 unit of energy per execution and the multipliers consume 5 units of energy per execution.

After the DFG is formed, scheduling and allocation are done. The scheduling information is shown incorporated along with the DFG in Figure 5. In our approach, we have used a greedy hardware allocation and scheduling method based on the initially assigned adder and multiplier delay values. We find that the length of the critical path is 7 units. Then, we find all the paths. The dashed arrows are dependences because of resource sharing. The paths are sorted according to their slacks in
increasing order. For our example, path with operations 0, 2, 6, 8, 9 is one of the critical paths (Path A in Figure 6). We apply the EDR Paradigm to this path and get delay values of 2.25 time units for M0 and 0.83 time units for A0 and A1. The optimized delays of adders turn out to be equal since we assumed (for simplicity) that the switching activities are equal. But this is not the case in general. After this path is optimized, we have delay values for A0, A1, and M0. Now, all the operations performed on A0, A1, and M0 have assigned delays (see shaded operations in DFG of Figure 7). To get the delay value of M1, we now pick the next critical path passing through one or more operations with unassigned delay (Path B in Figure 7). In general, when the next critical path is considered, some of the operations in that path will have assigned delay values. The available total delay is

\[ d_{\text{available}} = T - \sum (\text{assigned delay values in the path}) \]  

The EDR Paradigm is then applied to distribute the available total delay \( d_{\text{available}} \) among the operations on the path with unassigned delays. For example, \( d_{\text{available}} \) for Path B in Figure 7 is

\[ d_{\text{available}} = 7 - (d_M + d_{A0} + d_{A1}) = 4.51 \]
Delay of M1 is trivially obtained in this case to be 2.25 time units. For the remaining paths, we verify that this assignment of delays to the FUs does not violate the timing constraints and we can proceed to next step. Now we assign delay values of 2.49 to the multipliers in order to make the delay a multiple of 0.83. The critical path delay with the new delays is 7.47. Finally, we normalize the delays to make critical path delay equal to 7. Resulting adder delays are 0.77 each; resulting multiplier delays are 2.31 each. Using the approximation that E.D^2 is constant, we find the new energy consumptions of the FUs.

For multipliers M0 and M1, the energy consumption per execution changed from 5 to 3.74. For adders A0 and A1, the energy consumption per execution changed from 1 to 1.68. Since there are 5 additions and 5 multiplications, the total energy consumption dropped from 30 to 27.1.

The resulting system has two VDD values, one for adders and one for multipliers. New VDD values are found using Eq. 3 and the starting VDD and delay values. Let us assume the initial VDD for the system was Vinit. Then, the new VDD value for adders A0 and A1 is 1.3xVinit; new VDD value for multipliers M0 and M1 is 0.86xVinit. A generalized overview of this optimization is given in Figure 8.

6. Results

A C++ program (approximately 3000 lines of code) was written to implement the EDR Paradigm. We used low-pass FIR filters of orders 5 and 50 and a band-pass IIR filter of order 5 as benchmark circuits. We used a 16-bit carry Look-Ahead (CLA) Adder (designed with TSMC .25u) as the base adder and a 16-bit Wallace Tree multiplier (designed with TSMC .25u) as the base multiplier for modeling the functional units (FUs) used in the filters. Unoptimized energy consumption values are obtained by simulating the base adder and multiplier with Power Compiler (by Synopsys).

After the EDR Paradigm is applied to the benchmark circuits, the new delay values obtained for the FUs are used to generate the corresponding energy values using the approximations in Eq. (3) and Eq. (14). We report the optimized energy consumption values for the benchmark circuits in Table 1. Each different delay value corresponds to a different supply voltage. The number of supply voltages used is given in the “# Voltages” field in Table 1. The “Delay” field is the time in which an input sample has to be processed by the filter and represents the timing constraint on the system (T = \sum d). The “# Adders” and “# Multipliers” fields in Table 1 give the constraints on number of adders and multipliers. Figure 9 shows the energy savings for delays assigned randomly to FUs for the benchmark FIR8 with 3 adders and 3 multipliers. Most savings occur when the delays are assigned according to EDR Paradigm (upper-most big point in Figure 9).
7. Conclusion

In this paper, we presented an algorithm, EDR Paradigm, to find optimum energy consumption of sequentially executing circuits. This method is proved mathematically. Then we used this method as a basis for generating a heuristic way to optimize data flow graphs with a timing constraint for low energy consumption. When the results are examined, it is seen that more energy consuming modules should run slower (run with lower supply voltage), and less energy consuming modules should run faster (run with higher supply voltage) for obtaining low energy consumption while meeting the timing constraints.

References


