Voltage Scheduling Under Unpredictabilities: A Risk Management Paradigm

Azadeh Davoodi and Ankur Srivastava
Department of ECE, University of Maryland, College Park
azade,ankurs@eng.umd.edu

Abstract

This paper addresses the problem of voltage scheduling in unpredictable situations. The voltage scheduling problem assigns voltages to operations such that the power is minimized under a clock cycle constraint. In presence of unpredictabilities meeting the clock constraint cannot be guaranteed. This paper proposes a novel risk management based technique to solve this problem. The risk management paradigm assigns a quantified value to the amount of risk the designer is willing to take on the clock cycle constraint. The algorithm then assigns voltages in order to meet the expected value of clock cycle constraint while keeping the maximum delay within the specified "risk" and minimizing the power.

Categories & **Subject Descriptors** B.5.2 Design Aids Optimization
General Terms: Algorithms, Reliability, Design
Keywords: Predictability, Voltage Scheduling, Low Power, Design Closure

1. INTRODUCTION

Estimation in design automation is marred by inaccuracies. Important design objectives like power are extremely hard to predict especially at high levels of design flow. On the other hand optimization of design objectives at system level has tremendous impact on design quality. Hence critical optimizations need to be performed in unpredictable scenarios. This paper presents a novel methodology for addressing unpredictabilities through a risk management paradigm. This philosophy is demonstrated using the voltage scheduling problem in high level synthesis.

Risk Management essentially tries to control/manage the amount of error/unpredictability associated with any estimation. Under this paradigm the user specifies a risk which signifies the amount of inaccuracy the designer can "risk". Given a constraint C, the user tries to optimize design quality by risking a certain violation likelihood of C. This paper demonstrates this paradigm through the voltage scheduling problem which assigns voltages to individual operations in a data flow graph for power optimization [5] under a clock latency constraint. We present a methodology in which this voltage scheduling can be performed such that the expected value of clock latency is always less than user specified value of C, the maximum clock latency is less than user specified value R and the power dissipation is minimum. The advantage of a risk management paradigm is that it gives control of the unpredictabilities to the designer. Depending on the amount of "risk" the designer is willing to take, the design quality changes. If the acceptable risk is high, the algorithm will be relaxed and will generate a lower power solution. Hence we can expect a "risk" vs power tradeoff.

The rest of this paper is organized as follows. Section 2 contains a brief discussion on the issue of unpredictabilities. Section 3 re-investigates the low power voltage scheduling problem. Section 4 presents our risk management algorithm and Section 5 contains the experimental results.

ISLPED'03, August 25–27, 2003, Seoul, Korea.

2. UNPREDICTABILITIES: AN INTEGRAL ASPECT OF DESIGN AUTOMATION

Accuracy of estimation is severely limited at system level since various design parameters are not known. On the other hand design decisions taken at high level greatly impact the design quality and the time to market. Hence critical design decisions need to be made in the presence of high degrees of estimation inaccuracies.

2.1 Unpredictability: Definition, Sources and Impact

Unpredictability is defined as the quantified form of accuracy [1]. Focusing this discussion primarily on high level stage of design flow, there are many sources of unpredictabilities (for various cost functions such as power). The unawareness of exact logic structure of functional module makes exact estimation of power,area,delay etc impossible. Another important source of unpredictability in power estimation is unawareness of exact switching activity. Furthermore exact values of low level details like wire-delay, wire-capacitance cannot be determined either, forcing estimation to have inaccuracies.

Hence inaccuracies/unpredictabilities creep in due to unawareness of exact implementation details. Purely improving the models (by incorporating low level parameters) will not help since the complete implementation is not available. Therefore a system level strategy is desired which not only optimizes the pertinent design objective but also gives adequate control over the associated predictability. This paper proposes such a technique for voltage scheduling for power optimization which is a popular technique at system level.

2.2 Risk Management: Tradeoff Between Design Quality and Predictability

All optimizations in VLSI CAD have two aspects: Design objective and Design constraints. The design constraints must be satisfied to generate a valid/feasible design. Let us suppose there is a design objective O and a constraint C. The traditional

approach has been to optimize O while maintaining C. The resulting solution does not guarantee the satisfaction of C after the execution of the whole design process due to the presence of estimation unpredictabilities. In order to solve this problem, we present a "risk management approach". Let us assume that we know the kind of unpredictabilities associated with each solution of the optimization problem. Which essentially means that we know the range of values that the constraint C can have for each solution. Hence for each solution to the voltage scheduling problem, we know not only the expected value of the clock cycle C but also its' range. Let us also suppose that the designer specifies a risk factor he/she is willing to take on C. This essentially specifies the maximum violation of C the designer is willing to tolerate. For example C might be 10 clocks but the designer is willing to tolerate C upto 12. In the presence of such a scenario, an **unpredictability/risk management** paradigm would approach the problem in a following style

1. The expected value of the constraint should always be less than C
2. The associated unpredictability (range of values) with C is such that it is always less than the designer specified risk
3. The objective value O is the best possible in these scenarios

Hence the unpredictability management paradigm essentially manages the associated error to keep it within acceptable bounds. Note that the expected value of the constraint must still be less than C. Its just that the worst case likelihood of the constraint can be tuned. This papers presents an unpredictability/risk management approach for the power driven voltage scheduling problem.

3. LOW POWER VOLTAGE SCHEDULING

In this section we will briefly overview the multiple supply voltage scheduling problem. Power is quardrically dependent on the supply voltage as illustrated below [3].

$$Power = K.V^2.\beta \quad (1)$$

Where
K: Proportionality constant, V: Supply voltage, β : Switched Capacitance factor

Numerous techniques have been proposed to optimize the various components of this expression. Voltage scaling which basically reduces the supply voltage quadratically affects the power but also increases the delay which can be represented as follows

$$Power = K1.V/(V - V_t)^{\alpha} \quad (2)$$

Where
K1: Proportionality constant, V: Supply voltage, V_t : Threshhold voltage

System level voltage scheduling techniques take a data flow graph (DFG) as input and assign voltages to each operation such that the sum of overall operation power is minimized and the given clock cycle constraint is satisfied. This problem was tackled in [2] and solved optimally for general "Directed Acyclic Graphs" like DFGs assuming the same voltage-delay/power curves for all nodes. [5] relaxed this assumption and solved the problem optimally for Tree DFGs.

The voltage scheduling problem has the following inputs and outputs

1. INPUT: DFG, Voltage vs Delay,Power curves for all operations. This can include availability of different architectures for each operation type. Which essentially means that each operation can have multiple Voltage vs Delay,Power curves (corresponding to different architectures). A delay constraint in clock cycles C is also known
2. OUTPUT: Minimize the sum of power dissipation for all operations while maintaining the clock constraint. This is achieved by assigning voltages and architectures to each operation

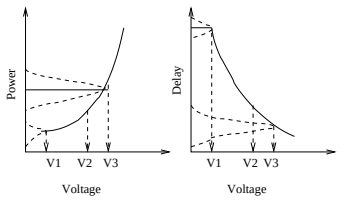

Figure 1: Power/Voltage, Delay/Voltage variation

Figure 1 illustrates a typical variation of voltage, delay and power. The formulation assumes a set of predecided voltages (V1, V2 and V3 in this case) which will be available on the chip. The problem is to assign voltages from this predecided list. The details of the algorithm can be found in [5].

4. RISK MANAGEMENT PARADIGM

Referring back to section 2, we propose a new voltage scheduling methodology to address unpredictabilities. The previous algorithm assumed accurate information about delay/voltage and power/voltage variations. This is definetly not a realistic assumption. In a risk management context, we assume that for each voltage choice we are given the distribution in delay and power (as illustrated by the dotted curves in figure 1). With this information we redefine our objective as follows

1. With each voltage choice we also have the probability distribution of delay and power. Given a delay constraint of C clocks. Also given a risk factor R where $R \geq C$ signifies the worst case delay that the designer is willing to risk
2. Minimize the expected value of power such that the expected value of delay $\leq$ C clocks. The worst case delay of the associated delay distribution of the scheduled DFG is always less than the risk R

Since we add the power dissipation of all nodes in the final solution, raeplacing the power estimate for each operation by its expected value should be enough. This is because the expected value of a sum of n variables is the sum of the expected values. Handling delay constraint and delay risk requires a sophisticated algorithm which will be described later.

4.1 Algorithms for Risk Management

The algorithm in [5] is modified to consider risk management. Once again, we are given delay and power distributions for all voltage choices. In this work we assume that these distributions are provided as input. An important question is "How do we know the distribution in delay and power for each voltage?" This is a very tricky problem. Distributions will strongly depend on the kind of optimizations the sub-design will be subjected to in future. Which in turn will depend on how critical a certain objective function becomes in that sub-design. It also depends on the *sensitivity* between different cost functions. Estimating these distributions requires a sophisticated estimation engine which does not exist in current state of the art estimation methodologies.

Let us suppose we know for each operation and each modular architecture for that operation, the expected value, maximum value and distribution for delay in clocks for each prespecified voltage. The problem is to assign voltages and architectures to operations such that the objective enumerated above can be satisfied. The algorithm contains two passes over the DFG. The forward pass traverses DFG in forward topological order and reverse for the reverse pass.

4.1.1 The Forward Pass

Before the forward pass starts some preprocessing performed. For each node we first generate a double dimensioned array. The row corresponds to the expected delay 1..C. The column signifies the risk 1..R. Let us call this information as $Node-Info(n)$ for each node n. The values stored at ij signifies the minimum power solution with expected node delay as i and max node delay as j. This info is stored in $Node-Info(n)[i][j].power$. The associated probability distribution of delay is also stored in $Node-Info(n)[i][j].probability$. We also remember the voltage and architecture that resulted in this solution. Note that this data is computed independently without any consideration to the inputs of these operations.

Now we proceed with the forward topological traversal of the DFG. At each node we compute another double dimensional data called $Arrival-Info(n)$ with dimensions 1..C,1..R. If node n is primary input then $Node-Info(n)$ and $Arrival-Info(n)$ are the same. The ij-th location of $Arrival-info(n)$ essentially contains the solution with minimum power dissipation (or minimum expected power dissipation) of the sub-graph rooted at node n and expected arrival delay of exactly i and max delay (risk) exactly j. The term arrival delay essentially signifies the number of clock cycles it would take for the data of n to become available. Conceptually it is similar to arrival time in gate level circuits. $Arrival-Info(n)[i][j].power$ stores the power value and $Arrival-Info(n)[i][j].probability$ stores the distribution in arrival delay. If a node n is not a primary input then the computation of $Arrival-Info$ is more involved.

ALGORITHM 1. *MAX-PROB(P1,P2)*

```
INPUTS: P1,P2: Input distributions
OUTPUT: P : Distribution which is Max(P1,P2)
comment: P1 and P2 are bounded by R, the maximum risk
for (i =1; i ≤ R; i++)
        P[i] = 0;
        for (j=1; j ≤ i; j++)
                P[i] += P1[i] * P2[j]
        for (j=1; j ≤ i; j++)
                P[i] += P2[i] * P1[j]
return P
end
```

ALGORITHM 2. *ARRIVAL-MAX(n1,n2)*

```
INPUTS: Operations n1,n2
OUTPUT: Arival-Max(Max(n1,n2))
comment: Arrival-Info(n1)and Arrival-Info(n2) has been computed
Allocate Memory for Temp
for (i =1; i ≤ C; i++)
        for (j=1 j ≤ R; j++)
            for (k = 1; k ≤ C; k++)
                for (l = 1; l ≤ R; l++)
Temp[i,j,k,l].probability =
MAX-PROB(Arrival-Info[n1][i][j].probability,
                Arrival-Info[n2][k][l].probability)
Temp[i,j,k,l].power =
Arrival-Info[n1][i][j].power + Arrival-Info[n2][k][l].power
for (i=1; i ≤ C; i++)
        for (j=1; j ≤ R; j++)
            Find k,l,m,n such that
            EXPECT(Temp[k,l,m,n].probability) = i and
            MAX(Temp[k,l,m,n].probability) = j, and with minimum power.
            Arrival-Max[i][j].probability = Temp[k,l,m,n].probability
            Arrival-Max[i][j].power = Temp[k,l,m,n].power
return Arrival-Max
end
```

ALGORITHM 3. *SUM-PROB(P1,P2)*

```
INPUTS: P1,P2: Input distributions
OUTPUT: P : Distribution which is SUM(P1,P2)
comment: P1 and P2 are bounded by R, the maximum risk
comments: P can be more than R
for (i =1; i ≤ R; i++)
        for (j=1; j ≤ R; j++)
            P[i+j] += P1[i] * P2[j]
return P
end
```

ALGORITHM 4. *Arrival-Info(n)*

```
INPUTS: n
OUTPUT: Arrival-info for n
compute Arrival-Max for n using Algorithm-2
for (i =1; i ≤ C; i++)
        for (j=1 j ≤ R; j++)
            for (k = 1; k ≤ C; k++)
                for (l = 1; l ≤ R; l++)
Temp[i,j,k,l].probability =
SUM-PROB(Arrival-Max[n][i][j].probability,
                Node-Info[n][k][l].probability)
Temp[i,j,k,l].power =
Arrival-Max[n][i][j].power + Node-Info[n][k][l].power
for (i=1; i ≤ C; i++)
        for (j=1; j ≤ R; j++)
            Find k,l,m,n such that
            EXPECT(Temp[k,l,m,n].probability) = i and
            MAX(Temp[k,l,m,n].probability) = j, and with minimum power.
            Arrival-Info[i][j].probability = Temp[k,l,m,n].probability
            Arrival-Info[i][j].power = Temp[k,l,m,n].power
return Arrival-Info
end
```

Let us point out the following, the arrival time for a node is defined as

$$arrival(n) = max(arrival(i)|i \in Fanin(n)) + delay(n) \quad (3)$$

In the current case both the delay and arrival time for fanins are distributions. Hence we need to first compute the max of all fanins probabilistically and add it with the delay of the node. Since we are traversing the DFG topologically, the $Arrival-Info$ for all fanins is known. The computation of max is illustrated in Algorithms 1 and 2. Algorithm 2 merges the $Arrival-$

Module/Architecture	Delay(ns)	Power(uw)
Adder/Ripple	17.07	11.25
Adder/Carry Look-Ahead	12.92	12.12
Mult/Arch1	41.62	191.91
Mult/Arch2	34.14	305.85

Table 1: Module Data at 5 Volts

$Info$ of two fanins using a max function. Essentially for all possible combinations of arrival delay distributions at the output of fanins, the algorithm calls Algorithm 1, which computes the max . Out of all these combinations, the ones with minimum power are chosen. This data is stored in $Arrival - Max$. This procedure is executed between first two fanins to generate an $Arrival - Max$. All subsequent fanins are then merged iteratively with this $Arrival - Max$ using Algorithm 2 (just replace one of the $Arrival - Info$ by $Arrival - Max$). Finally we have one $Arrival - Max$ data structure which contains the first term of equation 3. This needs to be added to the delay of the node n in order to compute $Arrival - info(n)$. This again needs to be done probabilistically. This procedure is illustrated in Algorithm 4. After forward traversal, the Arival-Info data structure for all the primary outputs is investigated to select the solution with minimum power.

4.1.2 The Backward Pass

After reaching the PO of the DFG, we choose a particular solution with the minimum power dissipation. This corresponds to a certain expected clock $\leq C$ and a certain risk $\leq R$. Taking this solution we traverse the DFG in topological order from PO to PI. At each step the fanout of the pertinent node forces a certain expected and max delay (basically ij-th location in Arrival-Info(n)). This corresponds to a certain architecture and voltage for the node n and certain expected and max delay values for the fanins. In this fashion the voltage and architecture for all the operations can be determined. If the DFG is a DAG, then there could be nodes with more than one fanout. Hence more than one solution could exist for such nodes. In this case we pick the solution with minimum arrival delay.

5. RESULTS

The primary objective of this paper is to propose the idea of risk management. The basic philosophy is to have a user controlled parameter which we call "risk" which controls the amount of possible risk in meeting the constraint at the penalty of design quality.

Table 1 illustrates the operation voltage vs delay,power at 5 volts (obtained using Synopsys DC). We assumed the same delay/power curves for all operations of the same type. Four distinct on-chip voltages 5, 3.3, 2.4 and 1.5 were assumed. Equations 1, 2 were used to calculate the power/delay values at these voltages. The underlying distribution at these voltages was assumed to be Gaussian with 20% variance and values predicted by equations 1, 2 as the mean.

Table 2 illustrates the comparison between traditional and risk driven approach. The benchmarks were a mix of MediaBench [4] and traditional High Level Synthesis bench-set. The same expected value and risk of timing was given to the algorithm. This essentially means that the designer is not willing to take any risk on timing constraint. It can be seen that the risk driven approach always results in a valid solution whereas the traditional approach does not. We took the traditional solution and estimated the maximum possible clock latency. It was observed that for all the benchmarks the worst case latency was more than the designer specified risk. The traditional approach which only considered the expected value could not result in meeting the constraint since the associated risk of the final solution was much higher than that specified by the designer. Next, without changing the expected value of timing, we illustrate the variation of power as the risk changes. Figure 2 illustrates this variation for various expected delay constraints.

Bench	T-Clk(ns)	Expect/Risk	Tradi	Risk-Manage
dct	8	25/25	-	301.35
ecbenc-4	8	25/25	-	225.92
ellipt	12	40/40	-	523.5
fft2	8	15/15	-	543
fir	8	35/35	-	197
jdmer-4	8	20/20	-	429
jdmer-3	8	8/8	-	472.78
jdmer-1	8	8/8	-	441.5
motion-2	8	14/14	-	655.38
motion-3	8	14/14	-	655
noise-2	8	8/8	-	837.44

Table 2: Comparison

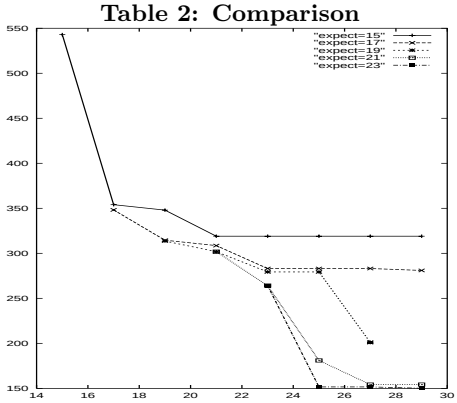

Figure 2: FFT2: X-Axis Risk, Y-Axis Power

6. CONCLUSION AND FUTURE WORK

This research proposes a formal methodology of addressing unpredictabilities at system level. Essentially the idea is to associate a risk factor with each constraint and by controlling the risk factor an appropriate solution that is guaranteed to be within the prescribed limits can be generated. This was demonstrated using the voltage scheduling problem. The future work includes the development of an unpredictability estimation engine.

7. REFERENCES

[1] A. Srivastava and M. Sarrafzadeh . "Predictability: Definition, Analysis and Optimization ". In *International Conference on Computer Aided Design*, Nov. 2002.

[2] S. Raje and M Sarrafzadeh. "Variable Voltage Scheduling". In *Proc. International Workshop on Low Power Design*, 1995.

[3] A.P. Chandrakasan, S. Sheng and R.W. Brodersen. "Low Power CMOS Digital Design". In *IEEE Journal of Solid State Circuits*, pages 472–484, April 1992.

[4] C. Lee, M. Potkonjak and W.H. Mangione-Smith. "MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems". In *International Symposium on Microarchitecture*, 1997.

[5] J.M. Chang and M. Pedram. "Energy Minimization Using Multiple Supply Voltages". In *IEEE Trans. on VLSI Systems*, Dec 1997.