CMOS Dynamic Power Estimation based on Collapsible Current Source Transistor Modeling [*]

Abelardo Pardo[†] R. Iris Bahar[†] Srilatha Manne[†] Peter Feldmann[‡] Gary D. Hachtel[†] Fabio Somenzi[†]

[†] University of Colorado
Dept. of Electrical and Computer Engineering
Boulder, CO 80309

[‡] AT&T Bell Laboratories
600 Mountain View
Murray Hill, NJ, 07974

Abstract

When estimating the dynamic power dissipated by a circuit different methods ranging from numeric analog simulation to event-driven logic simulation have been proposed. However, as the technology reaches the deep sub-micron range, additional effects as the short circuit current, partial voltage swings, transient behavior between clock cycles and leak current are becoming more relevant to achieve an accurate estimation of the consumption. In this paper we present Meiga, an **event-driven** *simulation-based power estimator that accounts for power dissipation due to short circuit current, partial swings and transient behavior. Experiments have shown that circuits in the order of several thousands of transistors are simulated in seconds of CPU per input vector.*

1 Introduction

In order for designers to effectively reduce power dissipation on a chip, a fast, yet accurate power dissipation model is needed to estimate power consumption of the entire circuit and target specific parts that need to be better optimized for low power. Power consumption of a chip is most often measured in terms of average power dissipation.

Several factors contribute to the power dissipation of a circuit [1]:

$$\begin{aligned} P_{total} &= (I_{ds} + I_{sc} + I_{leakage})V_{dd} \\ &= ((p_t \cdot C_L \cdot V \cdot f_{clk}) + I_{sc} + I_{leakage})V_{dd}. \quad (1) \end{aligned}$$

The first term , I_{ds}, (source-drain current) is usually considered the dominant factor in power dissipation and it depends on several factors: V is the voltage swing across the transistors, f_{clk} is the clock frequency of the chip, C_L is the sum of the collective load capacitances seen at the output of the gates, and p_t is the transition frequency seen at each gate output. This transition frequency results from both functional transitions or transient voltage swings, known as *glitches*, on the output. For a selected set of circuits, it has been found that glitching accounts for anywhere between 20% to 70% of the total power dissipated by a circuit [2].

[*]This work is supported in part by NSF/DARPA grant MIP-9115432 and SRC contract 94-DJ-560.

The other two currents that contribute to power dissipation are I_{sc} (direct-path short circuit current) appears when both NMOS and PMOS devices are on. Finally, $I_{leakage}$ (leakage current), is primarily determined by fabrication technology. It has often been argued that in "well-designed" circuits, short circuit and leakage currents are dominated by the source-drain currents, and can therefore be ignored when estimating power dissipation. However, in the sub-micron range, short circuit and leakage currents make a larger contribution to the total power dissipation. First, leakage currents do not scale down proportionally with device sizes making them more visible for sub-micron devices. In addition, the *slew rate*, or input slope will have a larger influence on the gate delay and power dissipation.

A variety of power estimators exist which constitute different tradeoffs between execution time and accuracy. Pattern independent simulators [3] obtain current estimation based in the structure of the circuit, no input vector is required. Simulation based tools require the analysis of different input vectors to estimate the power. In [4] Ghosh *et al.* proposed a symbolic simulation technique such that given the input transition rates, the transition frequencies of all gates in the circuit are computed. This method has the benefit of accounting for correlation among internal gates and sequential states from one time frame to the next. However, this type of simulation does not take into account the transient behavior of the circuit between clock cycles.

A variety of non-symbolic vector-based simulators exist. On one extreme we can place SPICE, accurate but with computationally expensive numerical methods; on the other extreme are switch level simulators such as MOSSIM [5], where each transistor is modeled as a switch whose state is either on, off, or unknown. In between there are simulators such as MOTIS [6] and RSIM [7]. MOTIS uses stored tables to describe the transistor behavior, avoiding the expensive matrix calculations done by SPICE. It uses interpolation or extrapolation and scaling of the stored values to find the actual currents, and hence loses some accuracy. RSIM's goal is to provide more accurate behavior than other switch level simulators by modeling the pull-up or pull-down structures as a resistor. Other simulators based on RSIM, such as Mom [8], have attempted to generate more accurate results by using a piece-wise linear model to represent the transistor's behavior, but this resulted in larger run times. In [9] Rouatbi *et al.* presented a tool specifically targeting power estimation for sub-micron circuits. However their approach still modeled gates by collapsing their structure into equivalent inverters.

Meiga is a simulation-based power estimation tool which aims to capture most of the non-linear behavior of CMOS timing characteristics while still remaining computationally efficient for large circuits by using event-driven simulation. The tool is based on the *Collapsible Current Source Model* that we consider is the simplest model of an MOS transistor that still preserves enough behavior to account for effects such as the ones described above. This model will allow us to simulate both the pull-up and pull-down blocks of a CMOS device without any kind of collapsing. Given an arbitrary piecewise-linear voltage waveform for each input, Meiga simulates this vector through the circuit accounting for power dissipation due to source-drain as well as short circuit currents.

2 Collapsible Current Source Model

In [10], Shoji proposes a simplified transistor model that emphasizes the non-linearity of the device. Considering this model, in an NMOS transistor model, the drain-source current is equal to $I_{ds} = \alpha(V_{gs} - V_{th})$ when $V_{ds} > 0$, where I_{ds} is the drain to source current, V_{gs} is the gate to source voltage, V_{th} is the threshold voltage, V_{ds} the drain to source voltage, and α is the driving capability of the transistor. If the current through the device is forced to be less than $\alpha(V_{gs} - V_{th})$, then $V_{ds} = 0$. In this region any current $0 \le I_{ds} \le \alpha(V_{gs} - V_{th})$ is allowed to flow. Furthermore, a voltage $V_{gs} - V_{th} < 0$ will produce $I_{ds} = 0$.

This model, referred as the *Collapsible Current Source Model* (CCSM), defines two different regions of operation for the transistor:

- Current Source region: In this region $V_{ds} > 0$ and $I_{ds} = max(\alpha(V_{gs} - V_{th}), 0)$.
- Collapsed Source region: The transistor is in this region whenever I_{ds} is forced to be less than $max(\alpha(V_{gs} - V_{th}), 0)$. Furthermore $V_{ds} = 0$.

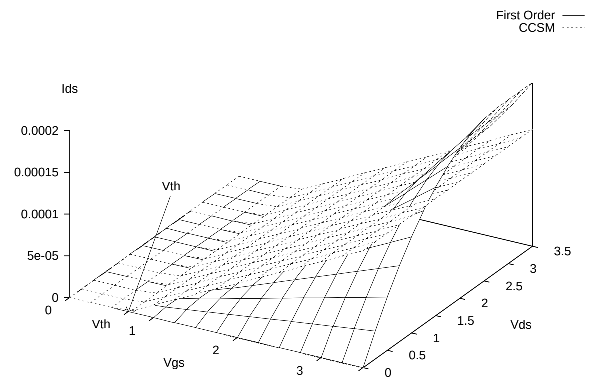

Figure 1: Comparison between first-order model and CCSM.

Figure 1 depicts the difference in characteristics between the first order MOS model and the CCSM.

The main features of the model are the following: It has a non-linear dependency of I_{ds} on V_{ds}, it reflects the driving capability of the transistor through the α parameter, the region of operation depends only on V_{ds} and finally, the current across the transistor is independent of V_{ds}.

Apart from the transistor model, an additional simplification is needed to achieve efficiency. The evolution of currents and voltages in the circuit through time will be represented as piecewise-linear functions. The waveform represents the current or voltage values at certain time points, and a linear approximation is used to obtain intermediate values. The power analysis is done in terms of operations over these current and voltage waveforms.

3 Equivalent Current Computation

The power estimation tool will receive a mapped CMOS circuit with n inputs and m outputs and the voltage waveforms at each input. The input voltage waveforms are assumed to have either $V(0) = V_{dd}$ or $V(0) = 0$, that is, the tool assumes all the nodes in the circuit be in steady state before applying any voltage change to the inputs.

Since only CMOS circuits are considered, the circuit will be regarded as a set of P-N blocks, where each block is a set of PMOS transistors connected in series with a complementary set of NMOS transistors. The analysis of a circuit at a P-N block level is performed as follows. Each P-N block receives a set of voltage waveforms in its inputs and produces a current waveform at its output. At the output of each P-N block there is a capacitor C_L connected to *gnd*. From the initial voltage in the capacitor and the current produced by the P-N Block a voltage waveform is produced. The way C_L is computed will be discussed in Section 4.

The process of analyzing the behavior of a P-N block can be divided in three major steps:

1. Compute the total current supplied by the P block.
2. Compute the total current supplied by the N block.
3. Combine both currents with the load capacitor and obtain the output voltage waveform.

Since in CMOS designs the transistors are connected using either serial or parallel topologies, we will define the waveform transformations needed to compute the equivalent of a set of transistors depending on the type of connection.

3.1 Parallel and Series Equivalent of a Set of Transistors

The following discussion is developed only considering NMOS transistors. The reasoning is completely symmetric for the case of PMOS transistors. Furthermore, it will be assumed that all the nodes in the circuit have non-negative voltages.

Definition 1 Maximum Current Waveform*: For a given transistor with voltage waveform $V(t)$ at the gate, let us denote as $I^{max}(t)$ the current waveform obtained when $V_{gs}(t) =$*

$V(t)$. By the definition of the CCSM in Section 2:

$$I^{max}(t) = max\{\alpha(V(t) - V_{th}), 0\} \tag{2}$$

Following the definitions given above, we can compute the equivalent maximum current source of a set of transistors connected in parallel as depicted in Figure 2(a).

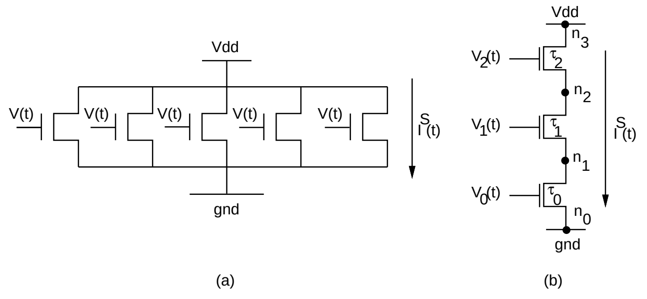

Figure 2: Set of Transistors in Parallel and Series Connection

Proposition 3.1 *The equivalent maximum current waveform across a set S of transistors connected in parallel is equal to:*

$$I^S(t) = \sum_{j \in S} I_j^{max}(t) \tag{3}$$

This proposition derives clearly from the fact that each transistor behaves as a current source with $I_j^{max}(t)$ as the current waveform. $I^S(t)$ represents the current across the set of transistors given that all of them are in the current source region.

To compute the equivalent of a set of transistors connected in series we will label the nodes from gnd to V_{dd} as $n_0 = gnd, n_1, \ldots, n_i = V_{dd}$. Figure2(b) shows an example chain of four transistors.

Proposition 3.2 *If the voltages in the intermediate nodes of a chain of transistors S connected in series are assumed to have non-decreasing voltages $V_{n_0} \leq V_{n_1} \leq \ldots \leq V_{n_i}$ then, the current driven by the chain S is given by the expression:*

$$I^S(t) = min_{j \in S}\{I_j^{max}(t)\} \tag{4}$$

Proof. Let us assume that there exists a time t such that $I^S(t) \neq min_{j \in S}\{I_j^{max}(t)\}$, then there are two possible cases:

Let us suppose $I^S(t) > min_{j \in S}\{I_j^{max}(t)\}$. This case would imply that at time t there is a transistor in the chain such that $I_{ds}(t) > I^{max}(t)$, but this is possible only if there is a node in the chain with negative voltage which violates the non-negative node voltage assumption.

Let us suppose $I^S(t) < min_{j \in S}\{I_j^{max}(t)\}$. The implication in this case is that at time t all the transistors in the chain are driving less current than the maximum current. Therefore by the assumption $V_{n_0} \leq V_{n_1} \leq \ldots \leq V_{n_i}$:

$$\forall j \in S, I_j(t) < \alpha(V_j(t) - V_{th}) \Rightarrow \forall j \in S, V_s > 0. \tag{5}$$

However, there is at least one transistor such that $V_{gs} = 0$ namely the one connected to gnd. □

Example: The result of Proposition 3.2 will become more evident by the following example. Let us consider the chain of three NMOS transistors depicted in Figure 2(b). We will label the transistors, starting from the gnd node τ_0, τ_1, and τ_2. The transistors have voltage waveforms $V_0(t)$, $V_1(t)$, and $V_2(t)$, respectively, at the gate. Suppose that at $t = 0$, $V_0 = V_{dd}$, $V_1 = 0$, and $V_2 = V_{dd}$. In this situation, the model allows a unique solution which corresponds to transistor τ_1 being in the current source region with $V_{ds} = V_{dd}$ and the rest of devices being in the collapsed source region with $V_{ds} = 0$. From this solution, the voltages V_{n_0}, and V_{n_1} are equal to gnd. Suppose that $V_1(t)$ starts increasing slowly from zero volts. Clearly, the current flowing across the chain will be limited by the transistor τ_1. Suppose that the current across τ_1 is increasing until it reaches a value equal to the current that another transistor, for example, τ_2 would allow to flow if its $V_{ds} > 0$. If $V_1(t)$ keeps increasing, then transistor τ_2 now becomes the limiting transistor, therefore it will be in the current source region and all the others in the collapsed source region. □

3.2 Algorithm to Analyze a P-N Block

The parallel and series equivalents defined above will allow us to compute the current waveform of an arbitrary P-N block.

Definition 2 Equivalent Graph of a block of Transistors*: The graph obtained by replacing each circuit node in the block by a node in the graph, and each transistor in the block by an edge connecting two nodes. Each edge will be labeled with the corresponding gate label of the transistor.*

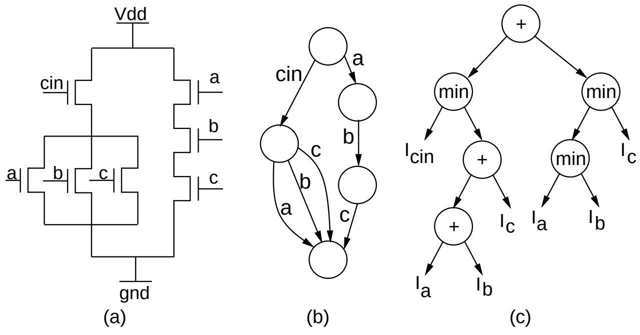

Figure 3: (a) Block, (b) Equivalent Graph, (c) Expression Graph

Figure 3(a)-(b) show a transistor block taken from a full-adder cell and its equivalent graph.

Once the description of a certain P-N block is read by the tool, the equivalent graph is built. The topology of this graph represents the sequence of operations executed over the input waveforms to obtain the equivalent current across the block. In order to perform these operations more efficiently, a second stage of translation is applied. From the equivalent graph an *expression tree is created.* This expression tree represents the operations and operands over waveforms. Every time a particular block is evaluated, this tree is traversed to obtain the equivalent current.

Assuming that I_a, I_b, I_c, and I_{cin} denote the maximum current supplied by each transistor, Figure 3(c) shows the expression tree obtained from the equivalent graph. This tree represents the expression:

$$I_N^{max}(t) = (I_a \perp I_b \perp I_c) + (I_{cin} \perp (I_a + I_b + I_c)), \quad (6)$$

where $\perp$ represents the minimum operation over waveforms. Since CMOS designs are considered, only one traversal operation over the tree is necessary to compute the equivalent current for the PMOS part of the block. Following the example depicted in Figure 3, the corresponding expression for the complementary P Block is:

$$I_P^{max}(t) = (I_A \perp I_B \perp I_C) + (I_{CIN} \perp (I_A + I_B + I_C)) \quad (7)$$

which can be obtained from Equation 6 by just interchanging the minimum and addition operations. Therefore only one pass through the tree computing a *minimum* operation whenever a + node is found and vice versa is needed to obtain the equivalent current waveforms $I_P^{max}(t)$.

4 Power Estimation Algorithm

The tool will account for two sources of power consumption. The short-circuit current and the power dissipated while charging and discharging the capacitance at the output of each P-N block. Once the maximum equivalent currents of the NMOS and PMOS transistors are obtained, the effect of these currents on C_L and the amount of short-circuit current flows across the block still remains to be computed.

So far, the analysis of a P-N block has been done in terms of maximum currents. However, the model proposed in Section 2 considered the situation in which $V_{ds} = 0$ and $I_{ds} < \alpha(V_{gs} - V_{th}$. The magnitude that will force this condition is the voltage across C_L.

The formula used to obtain the output capacitance for each P-N Block is the following:

$$C_L = C^j + \sum_{k \in fanout} C_k^{route} + \sum_{k \in fanout} C_k^{gate} \quad (8)$$

where C^j is the diffusion capacitance at the output of the P-N block (this value is specified in the description of the block), C^{route} is the capacitance associated with the routing, and C^{gate} is the gate capacitance at the input of the next block.

Given the maximum current waveforms of each block $I_P^{max}(t)$ and $I_N^{max}(t)$, we define the following two waveforms:

Definition 3 Charging Current $I_C(t)$: *The current responsible for charging and discharging the collective load capacitance. Its value is computed as:*

$$I_C(t) = I_P^{max}(t) - I_N^{max}(t). \quad (9)$$

For a given time t, a positive value of $I_C(t)$ means that the block is able to supply current to charge collective load capacitance. Analogously, a negative value means that the block is able to discharge the capacitance. Since equivalent currents were obtained for both the PMOS and the NMOS parts and they are connected in series, the short-circuit current can be obtained as another series equivalent.

Definition 4 Short-Circuit Current $I_{sc}(t)$: *This is the current flowing directly from V_{dd} to gnd across the P-N block. Its value is computed as:*

$$I_{sc}(t) = min\{I_P^{max}(t), I_N^{max}(t)\} \quad (10)$$

Given $I_C(t)$ and a certain initial voltage across the capacitor, we need to compute the voltage waveform produced at the output of the P-N Block. Since the input to the block was a piecewise linear voltage waveform, and the operations to compute the equivalent current all preserve this characteristic, $I_C(t)$ will also be a piecewise linear function. Given two points of this function $(t_j, i_j), (t_{j+1}, i_{j+1})$ and the voltage at time t_j in the capacitor $V_c(t_j)$, assuming an ideal capacitor, the voltage in the capacitor at time t_{j+1} is:

$$V_c(t_{j+1}) = V_c(t_j) + \frac{1}{C_L} \int_{t_j}^{t_{j+1}} \frac{i_j - i_{j+1}}{t_j - t_{j+1}}(t - t_{j+1}) + i_{j+1} dt \quad (11)$$

In theory, the points obtained when computing this integral are part of a second order polynomial because we are integrating a linear function. If this waveform in terms of second order polynomials is propagated through a P-N block, it produces a piecewise third order polynomial. In order to keep the computation efficient, some sort of simplification is needed. In the current implementation of the tool, the output waveform is approximated again by a piecewise linear.

Equation 11 is applied only if the resulting capacitance voltage is between the values *gnd* and V_{dd}. When in the process of charging C_L, the output voltage reaches the value V_{dd} there is a zero voltage drop across the PMOS part. Since the nodes inside the block are assumed to be non-negative, that implies that all the devices moved into the collapsed source region. At this point in time, even if the value of $I_C(t)$ is non-zero, the real value is set to zero. The fact that the PMOS part is in the collapsed source region implies that the current allowed by the NMOS part will be flowing across the PMOS part as well. Since the capacitance is fully charged, this current, if any, is precisely I_{sc}. The same reasoning applies when discharging the collective load capacitance and the output voltage reaches the value *gnd*.

```
Procedure CombinePN_Waveforms($I_C(t)$,$I_{sc}(t)$,$V(C_L)$) {
    for every time $t$ {
        if ($I_C(t) > 0$) then
            if ($V(C_L) < Vdd$) then   /* charge $C_L$ with $I_C$ */
                $V(C_L) = V(C_L) +$ ChargeCap($I_C(t)$);
                $Power_C =$ CalculatePower($V(C_L)$);
            else No current nor power consumption;
        else if ($V(C_L) > 0$) then   /* discharge $C_L$ with $I_C$ */
                $V(C_L) = V(C_L) -$ ChargeCap($I_C(t)$);
                $Power_C =$ CalculatePower($V(C_L)$);
        else No current nor power consumption;  }
    $Power_{sc} = Power_{sc} + V_{dd} \cdot Area(I_{sc}(t))$  }
```

Figure 4: Algorithm for Power Dissipation of P-N Block

Figure 4 shows the pseudo-code of the algorithm to combine both $I_C(t)$ and $I_{sc}(t)$ and to account for the dissipated power. The voltage across the capacitor, $V(C_L)$, varies according to the value of the charging current, $I_C(t)$. This updated voltage across the capacitor is then used to calculate the power dissipation due to the charging current. The power dissipated due to short-circuit current is computed as the supply voltage times the area of the piecewise-linear approximation of I_{sc} over time.

The complete power estimation algorithm requires the following operations over waveforms: addition, subtraction, minimum, multiplication by a constant and integration. All these operations can be performed in linear time in the number of points in the waveform.

5 Analog simulation comparison

In this section compare Meiga with SPICE for two simple circuits, shown in Figure 5. The inverter chain, Figure 5(a), shows the accuracy of Meiga in estimating the propagation of glitches through a several levels of logic. Each inverter in the chain is three times the size of the preceding inverter, and is balanced in size to produce similar rise and fall times. The circuit in Figure 5(b) uses a 2-input NAND gate to show the effects of multiple switching inputs and the glitch this produces in the final output. Again, the circuit is sized to produce similar rise and fall times. For both these circuits, routing capacitances have not been included so that we may show the effect of gate capacitances on the waveforms.

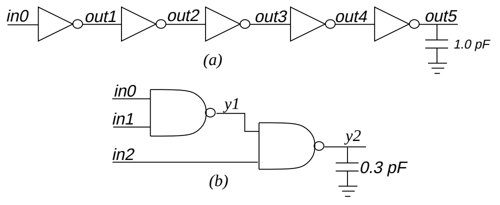

Figure 5: Simple Circuits used for Simulation Comparison.

Figures 6 and 7 show the simulation results for two different waveforms propagated through the circuits in Figures 5(a) and 5(b) respectively. Our results show that Meiga compares favorably with SPICE in both delay estimations and glitch propagation. In addition, power estimates using Meiga were within 3% of SPICE for the circuit in Figure 5(a) and within 9% of SPICE for the circuit in Figure 5(b).

Since Meiga is very sensitive to the value of α (the drive capability factor) chosen for each transistor, we believe that more accurate results can be obtained once α has been tuned to better reflect process characteristics. Meiga currently only handles a fixed value of α for each transistor independent of its input waveform or capacitive load. Therefore, if α is modeled as a function of the peak value in the saturation range, our simulator will produce smaller circuit delays. On the other hand, if α is modeled as a function of the bottom of the saturation range, we will produce larger circuit delays. In the future, a more realistic approach of adjusting α dynamically as a function of input slew rate and output load of each gate in the circuit may be applied.

6 Experimental Results

In the previous section, we have demonstrated the accuracy of Meiga for small circuits. However, the intent of the tool is to provide power estimation results for larger circuits with reasonable execution times. Table 1 presents the results obtained with circuit sizes up to 7700 transistors. These experiments were obtained on a DEC-Station 5240 with 92 MB of memory. Meiga was given as input a mapped circuit from SIS which used a library of inverters, nand and nor gates of varying sizes. Based on the sizes of the gates, the α values were calculated and specified as input to Meiga.

The information given in Table 1 is presented as follows: The columns labeled *Tr.* and G describe the size of the circuit in terms of transistors and gates, respectively; the *Time* column shows the average execution time for ten arbitrarily chosen vectors. These vectors had different slew rates as well as different switching times. The *Tr./S* column shows the number of transistors simulated per second; the columns labeled by P_{Meiga} and P_{SPICE} show the power estimated for one single vector with Meiga and SPICE, respectively. These two last columns show also the accuracy provided by the tool of circuits with up to 266 transistors.

Our simulation times averaged 1782 transistors per second per vector. Simulation time increased linearly with the size of the circuit and the largest circuit simulated,`tbk`, evaluated one vector in an average time of 5.47 seconds. Since the execution time depends linearly on the number of devices in the circuit, it is a safe assumption that this trend will hold for even larger circuits. This dependency derives from the fact that all the operations over waveforms to analyze the P-N blocks have linear complexity.

Circuit	*Tr.*	*G*	*Time*	*Tr./S*	P_{Meiga}	P_{SPICE}
s27	40	12	0.06	694	3.29e-10	2.826e-10
arbiter2	74	25	0.07	1088	3.80e-10	3.323e-10
modulo12	148	45	0.09	1705	7.55e-10	8.274e-10
arbiter4	152	50	0.10	1587	8.01e-10	7.524e-10
lion9	164	54	0.10	1657	7.79e-10	7.316e-10
bbtas	180	59	0.10	1804	1.06e-09	9.892e-10
arbiter6	222	71	0.13	1675	9.82e-10	9.990e-10
beecount	266	85	0.13	2008	1.10e-09	9.812e-10
dk512	392	125	0.19	2048	2.29e-09	
opus	582	180	0.30	1934	2.60e-09	
mm4	828	260	0.51	1609	1.08e-09	
sse	926	299	0.59	1570	4.18e-09	
ex2	1098	344	0.71	1536	4.90e-09	
donfile	1222	369	0.80	1526	6.15e-09	
cse	1762	553	0.55	3233	8.70e-09	
dk16	2098	634	0.66	3181	1.06e-08	
ex1	2230	703	2.35	948	9.12e-09	
s1	3772	1134	1.69	2230	1.50e-08	
styr	4418	1318	2.19	2016	1.17e-08	
planet	4982	1531	2.85	1748	1.19e-04	
tbk	7708	2217	5.47	1408	2.26e-08	

Table 1: Average CPU time for Ten Vector Simulations.

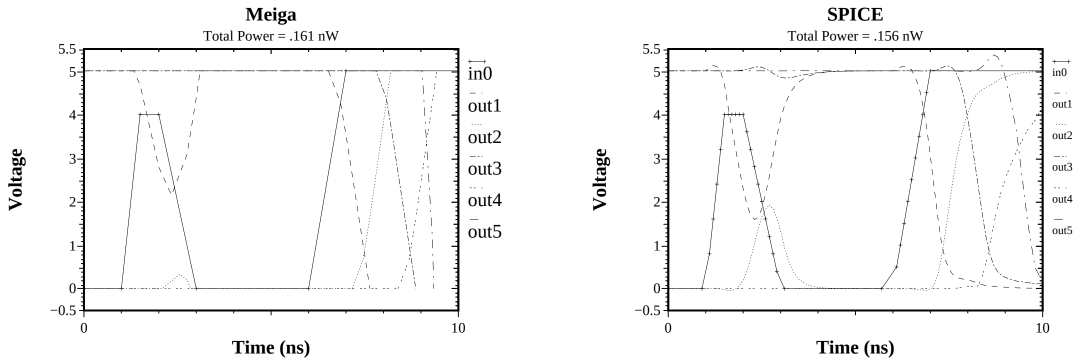

Figure 6: Vector Waveforms for Meiga Compared With SPICE for Circuit in Figure 5(a).

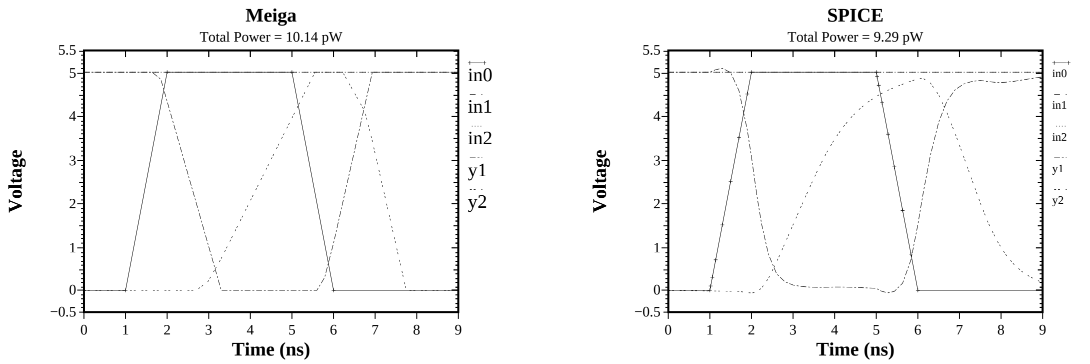

Figure 7: Vector Waveforms For Meiga Compared With SPICE for Circuit in Figure 5(b).

7 Conclusions and Future Work

We have presented Meiga, a tool for estimating the power for large circuits efficiently, having simulation times linear with the size of the circuit. We have also shown that Meiga compares favorably with SPICE in accurately modeling the nonlinear behavior of CMOS circuitry. Compared with SPICE, we trade off a modest amount of accuracy for a major reduction in execution time.

Compared to existing symbolic techniques for power estimation, Meiga will provide more accurate information at the expense of longer execution times. On the other hand, because of the linear complexity of the algorithm, execution times are not formidable.

In the future we hope to expand Meiga's power estimating capability by including leakage current in our calculations. We are also planning to incorporate the effects of slew rate and load capacitance in the calculation of α, the driving capability of the transistor. Finally, we would like to incorporate our power estimation tool into an entire low-power synthesis and verification environment, where accurate initial power estimations are essential for properly targeting synthesis for low-power.

References

[1] A. P. Chandrakasan, S. Sheng, and R. W. Broderson, "Low-power CMOS digital design," *IEEE Jour. Solid State Circ.*, pp. 473–484, Apr. 1992.

[2] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer, "On average power dissipation and random pattern testability of CMOS combinational logic networks," in *Proceedings of the International Conference on Computer-Aided Design*, (Santa Clara, CA), pp. 402–407, Nov. 1992.

[3] R. Burch, F. Najm, P. Yang, and D. Hocevar, "Pattern-independent current estimation for reliability analysis of CMOS circuits," in *25th ACM/IEEE Design Automation Conference*, pp. 294–299, June 1988.

[4] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," in *Proceedings of the Design Automation Conference*, (Anaheim, CA), pp. 253–259, June 1992.

[5] R. E. Bryant, "Mossim: A switch-level simulator for mos lsi," in *Proceedings of the Design Automation Conference*, pp. 280–287, June 1981.

[6] B. R. Chawla, H. K. Gummel, and P. Kozak, "Motis – an mos timing simulator," *IEEE Transactions on Circuits and Systems*, vol. 22, pp. 901–910, Dec. 1975.

[7] C. J. Terman, "Rsim. a logic-level timing simulator," in *Proceedings of the International Conference on Computer Design*, pp. 437–440, Oct. 1983.

[8] R. Kao and M. Horowitz, "Piecewise linear models for RSIM," in *Proceedings of the International Conference on Computer-Aided Design*, pp. 753–758, Nov. 1993.

[9] F. Rouatbi, B. Haroun, and A. J. Al-Khalili, "Power estimation tool for sub-micron CMOS VLSI circuits," in *Proceedings of the International Conference on Computer-Aided Design*, pp. 204–209, 1992.

[10] M. Shoji, *Theory of CMOS Digital Circuits and Circuit Failures.* Princeton University Press, 1992.