

SYCLOP: Synthesis of CMOS Logic for Low Power Applications

Kaushik Roy and Sharat Prasad
Texas Instruments Inc.
Semiconductor Process and Design Center
Dallas, Texas 75230

Abstract

In this paper we present a system called SYCLOP that synthesizes Finite State Machines (FSMs) and combinational logic for low power applications. SYCLOP tries to minimize the area and the *transition density* at the internal nodes of a CMOS circuit. The minimization is based on the input signal probabilities and the transition densities. Results have been obtained for a wide range of benchmark examples.

1 Introduction

Power dissipation in CMOS circuits can be minimized by scaling down the supply voltage. However, the circuit performance becomes slower. This can be compensated for by scaling down the device features. In a CMOS logic-gate, the power dissipation is given by $V_{DD}^2 \cdot f \cdot C$, where C is the capacitive load and f is the frequency of ZERO-to-ONE or ONE-to-ZERO transitions. V_{DD} denotes the supply voltage. For non-periodic signals, f is defined as the *transition density* [1] and can be expressed in terms of $\frac{n_x}{T}$, where n_x is the number of transitions at node x in the time interval $[-\frac{T}{2}, \frac{T}{2}]$. In our synthesis system, we try to minimize the transition density *ie. the average switching rate* in a circuit. The rate at which the node transitions occur is also a measure of the *stress* that can cause failures in digital circuits [3, 1]. Hence, the circuits synthesized using transition-density measure are less susceptible to run-time failures.

FSM and combinational-logic synthesis have been conventionally targeted to reducing the area and the critical-path delay [5, 7]. Recently, testability has also been considered during the synthesis process [6, 8]. Power dissipation considerations, however, have received but little attention. Brodersen et. al. [4] have considered the aspect of power-minimization, but they primarily focused on the impact of the *architecture* on power dissipation. Our synthesis system addresses the power-issue in two parts: state assignment to determine the combinational-logic function, and multilevel optimization of the combinational logic. The technique is based on input signal probabilities and transition densities,

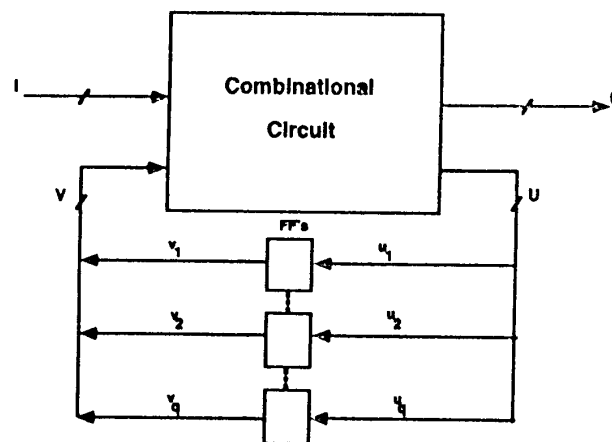


Figure 1: State machine representation

and hence, a particular circuit can be optimally synthesized in different ways for different applications.

2 Preliminaries and Definitions

2.1 Representation of FSMs

We represent FSMs by Probabilistic State Transition Graphs (PSTGs). PSTGs are directed graphs consisting of a set of nodes S and a set of edges E . Each node $S_i \in S$ represents a state of the FSM. There is a directed edge between nodes S_i and S_j with a label L_{ij} if there exists a set of inputs I , which causes the machine at state S_i to go to state S_j with output O . L_{ij} denotes the values of the primary inputs and the corresponding primary outputs for that transition. Each edge is also associated with a number p_{ij} , $0 < p_{ij} \leq 1$, which denotes the conditional probability of a state transition from S_i to S_j . Hence, if there are k outgoing edges from node S_j , each associated with a prob-

ability of p_{jm} , $m \leq k$, then

$$\sum_m p_{jm} = 1$$

We consider completely specified FSMs. If a machine is incompletely specified, then we make it completely specified by introducing a self loops corresponding to the don't care inputs at each state S_j .

The *Hamming distance* between any two states S_i and S_j denotes the total number of bits that the states differ in, and is denoted by $H(S_i, S_j)$.

2.2 Logic signals

This subsection will briefly describe the concepts of signal probability and transition density[1].

Definition: The *signal probability* is the probability that a signal g assumes a logic value of ONE is given by:

$$P(g) = \lim_{T \rightarrow \infty} \frac{1}{2T} \int_{-T}^{+T} g(t) dt$$

Definition: The *transition density* of a logic signal $g(t)$ is given by

$$D(g) = \lim_{T \rightarrow \infty} \frac{n_g(T)}{T}$$

where $n_g(t)$ is the number of transitions of $g(t)$ in the interval $[-\frac{T}{2}, \frac{T}{2}]$.

For a multi-input, multi-output logic module M , we assume that the inputs $g_1, g_2, \dots, g_n$ to the module are mutually independent processes, each having a signal probability $P(g_i)$ and a transition density $D(g_i)$, $i \leq n$. The signal probability at the output can be easily computed using the methods described in [9, 2]. For example, if P_1, P_2 , and P_3 are the input signal probabilities to a three input AND gate, the output signal probability is given by $P_1 P_2 P_3$. The transition density at any output h_j , of M is given by [1]

$$D(h_j) = \sum_{i=1}^n P(B(h_j, g_i)) D(g_i)$$

where $B(h_j, g_i)$ represents the Boolean difference of h_j with respect to g_i . The signal probabilities and the transition densities at the primary inputs are usually available. The P and D values at the internal nodes are symbolically computed if the circuit has reconvergence fanout[2].

2.3 Average power dissipation

The average power dissipation in a CMOS circuit can be written as

$$POWER_{avg} = \frac{1}{2} V_{DD}^2 \sum_i C_i D(g_i)$$

where g_i is the i^{th} circuit node. During the multilevel-logic synthesis process, the capacitive load C_i , at each circuit

node i , is approximated by the fanout factor at that node. We define the *power dissipation measure* Φ as

$$\Phi = \sum_{i \in R} fanout_i D(g_i)$$

where R is the set of all the inputs and the internal nodes, and $fanout_i$ is the number of fanouts at node i .

3 State Assignment for FSMs

This section addresses the encoding of the states of a synchronous sequential machine. This encoding is based on the input signal probabilities and the transition densities. The state encoding scheme uses the *likelihood of state transition* information.

Let p_{ij} and L_{ij} , respectively denote the state transition probability and the label associated with an edge $S_i - S_j$ of a PSTG. Let there be N_I primary inputs each having a signal probability P_x , $x \leq N_I$. The state transition probability p_{ij} is determined by the value that a primary input x in L_{ij} assumes and is given by

$$p_{ij} = \prod_{x \in \text{inputs in } L_{ij}} W_x$$

where

$$W_x = \begin{cases} P_x & \text{if } value(x) = 1 \\ 1 - P_x & \text{if } value(x) = 0 \\ 1 & \text{if } value(x) = \text{don't care} \end{cases}$$

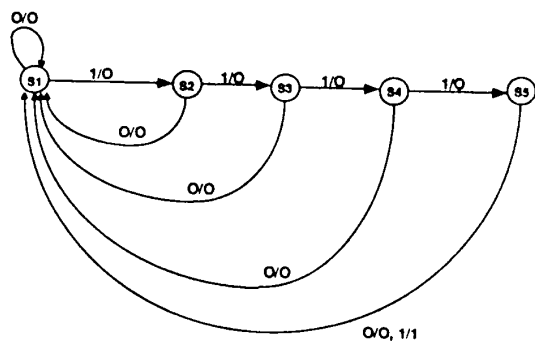
For N_S states, the minimum number of flip-flops required for coding is $\lceil \log_2 N_S \rceil$. It should be noted that if one-hot coding with N_S flip-flops is used, the Hamming distance between any two states is always 2, and hence, an optimum assignment to minimize the number of transitions at the present state inputs might not be obtainable. Besides, one-hot coding also increases the number of present state inputs to the combinational logic of Figure 1. The average number of switching to the present state inputs to a state machine can be minimized if the state assignment scheme is such that the following objective function is minimized:

$$\gamma = \sum_{\text{over all edges}} p_{ij} H(S_i, S_j)$$

The higher the state transition probability between two states, the lower should be the Hamming distance between them.

Due to the complex nature of the objective function γ , simulated annealing was used to solve the problem of state assignment. We start with a random assignment of states having the prescribed number of bits. Two types of moves are allowed during annealing: interchanging the codes of two states, or assigning an unassigned code to a state randomly picked for an exchange.

Figure 2 shows a state machine which produces an output of ONE whenever a sequence of five ONEs appear, else it outputs a ZERO. The machine was implemented by three D



	CODING 1	CODING 2
S1	010	000
S2	101	100
S3	000	111
S4	111	010
S5	100	011

Figure 2: State assignment

type flip-flops. Two assignment schemes were used; these are shown in the Table. The input signal probability is assumed to be 0.5. For Coding-1, γ is 10, whereas, for Coding-2, γ is 5.5. Both machines can be implemented using 34 transistors and 3 flip-flops.

State assignment determines the functionality of the combinational logic which is represented by $\mathcal{F}(I, V)$, where I is the set of primary inputs, and V represents the present state inputs (refer to Figure 1). Given the signal probabilities and the transition densities for each input $i_k \in I$, the signal probabilities and the transition densities for the V inputs have to be determined for multi-level logic optimization. We use simulation to determine the P and D values at the V inputs.

Primary input signals are randomly generated such that the signal probabilities and the transition densities conform to a given distribution. The state machine is then simulated to determine the number of transitions occurring at bit v_j of the machine and the percentage of time the bit has a logical value of ONE. The number of transitions divided by the total number of simulations give the transition density $D(v_j)$, at the input.

4 Logic Synthesis

Conventional logic-synthesis systems try to minimize the silicon area required for the combinational-logic blocks. In SYCLOP, power savings is considered as an additional objective during the synthesis process. As in MIS [5], SYCLOP finds multiple cube common divisors or single cube common divisors among two or more nodes. A set

Table 1: Results on benchmark examples with minimum coding bits

example	states	input	edges	γ_{min}	γ_{max}
ex1	20	9	138	25.5	40.03
ex3	10	2	36	16.25	23.25
ex7	10	2	36	13.75	21.75
keyb	19	7	170	68.1	145.8
sand	32	11	184	37.48	57.65
train11	11	2	25	5.5	10.75
opus	10	5	22	6.59	13.15
bbtas	6	2	24	3.5	8.75
lion9	9	2	25	5.0	12.0
bbara	10	4	60	3.5	5.8
planet	48	7	115	110.81	172.875

of kernels are computed for each logic expression. Then the non-trivial intersection among the kernels from different functions are formed. We consider not only the literal-count reduction but also the reduction in the power dissipation measure Φ , while choosing between various alternative intersections. Both level 0 and higher level kernels are generated. The best N kernel intersections, on the basis of literal count reduction, are chosen. Among them, the one which gives the best trade-off between area and power reduction is selected.

For calculating the reduction in Φ , a naive implementation would involve putting the factor as a node in the network, subsequently dividing all the nodes for which this new node can be a factor, calculating Φ , and finally removing the node, along with all its effect, from the network. Choosing the best factor out of N choices is expensive (from a computational point of view) if the above method is used. So, we compute the power dissipation due to the transitions at the circuit node local to the factor under consideration. This value is used to compute a power savings $\Delta\Phi$ similar to the area savings ΔA . The total savings is given by

$$S = \mu * \frac{\Delta\Phi}{\Phi} + (1 - \mu) * \frac{\Delta A}{A},$$

where $\frac{\Delta A}{A}$ and $\frac{\Delta\Phi}{\Phi}$ represents the fractional change in area and Φ respectively. μ denotes the relative weight factor defined by the user, which reflects the relative importance of the reduction in power dissipation measure with respect to area savings.

5 Implementations and Results

The algorithms for state assignment and logic synthesis have been implemented in LISP on an Explorer workstation. Table 1 shows the results of our state assignment scheme on the MCNC benchmark examples. For all primary inputs, a signal probability of 0.5 and a transition density of 0.5 (transition/clock cycle) were assumed. The

Table 2: Synthesis Results

example	γ_{min}		γ_{max}	
	transistors	Φ	transistors	Φ
ex1	2072	984.4	2380	1465.2
ex3	284	46.4	344	55.3
ex7	304	48.8	404	62.8
keyb	1364	561.4	2424	1170.4
sand	2958	1497.1	3040	1503.4
train11	418	101.3	466	151.3
opus	452	160.1	530	234.2
bbtas	90	20.2	130	55.2
lion9	276	66.9	374	131.1
bbara	290	69.0	342	105.4
planet	2650	2012.0	3226	3854.9

state machines were experimented with the $\lceil \log_2 S \rceil$ state-bits. The minimum value of the objective function γ_{min} is shown in the Table. For comparison, γ_{max} is also shown in Table 1. The signal probabilities and the transition densities at the V inputs to the combinational logic were determined by simulating 10,000 randomly generated primary inputs. Table 2 shows the results of applying our synthesis algorithm to the examples of Table 1. Two types of circuits were synthesized for comparison. The second and the third columns show the transistor count and the power dissipation measure Φ of the circuits synthesized using the state encoding scheme which produced an objective function value of γ_{min} . Similar results for circuits encoded with states which produced γ_{max} as objective function value are shown next for comparison. In both cases, the combinational logic was synthesized to minimize the power dissipation measure. The results show that the difference in power dissipation measure can be large for most of these circuits. For example, the increase in transistor count between the two representations of *ex1* is 13%, however, the increase in power dissipation measure is 33%. With the same inputs, the two machines of Figure 2 were simulated with 1000 inputs using SPICE. Figure 3 shows time-average power for the two machines. Coding-1, for which γ is higher, produced 25% more power dissipation than the one with lower γ . Both machines require the same number of transistors for implementation.

6 Conclusions

A synthesis system called SYCLOP has been developed to synthesize both finite state machines and combinational logic for low power applications. SYCLOP tries to minimize the transition density at the internal nodes of a circuit in order to minimize the power dissipation during its normal operation. Using this system, a particular circuit can be optimally synthesized for different applications.

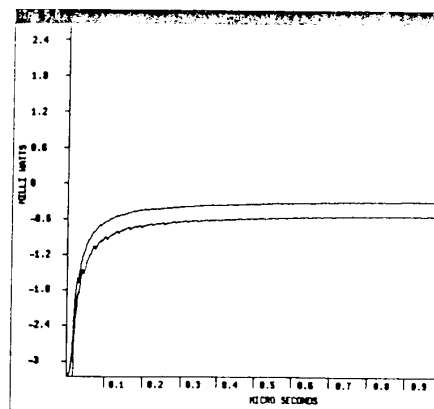


Figure 3: SPICE plots for the machine of Figure 2

References

- [1] F.N. Najm, "Transition Density, A Stochastic Measure of Activity in Digital Circuits", *ACM/IEEE Design Automation Conf.*, 1991.
- [2] K. Parker, E. McCluskey, "Probabilistic Treatment of General Combinational Networks", *IEEE Trans. on Comp.*, June, 1975.
- [3] R. Iyer, D. Rossetti, and M. Hsueh, "Measurement and Modeling of Computer reliability as Affected by System Activity", *ACM Trans. on Computer Systems*, Vol. 4, No. 3, August, 1986, pp. 214-237.
- [4] R.W. Brodersen, A. Chandrakasan, and S. Sheng, "Technologies for Personal Communications", *1991 Symp. on VLSI Circuits*, Tokyo, pp 5-9.
- [5] R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang, "MIS: A Multiple-Level Logic Optimization System", *IEEE Trans. on Computer-Aided Design*, November, 1987, pp. 1062-1081.
- [6] K. Roy, J. Abraham, K. De, and S. Lusky, "Synthesis of Combinational Circuits for Robust Delay Fault Testability", *Intl. Conf. on Computer-Aided Design*, 1989, pp. 418-421.
- [7] S. Devadas, H-K T. Ma, A. Newton, and A. Sangiovanni-Vincentelli, "MUSTANG: State Assignment of Finite State Machines Targeting Multi-Level Logic Implementations", *IEEE Trans. on Computer-Aided Design*, December, 1988, pp. 1290-1300.
- [8] A. Pramanik, S. Reddy, and S. Sengupta, "Synthesis of Combinational Logic Circuits for Path Delay Fault Testability", *Intl. Symp. on Circuits and Systems*, May 1990, pp3105-3108.
- [9] J. Savir, G. Ditlow, and P. Bardell, "Random Pattern Testability", *IEEE Trans. on Computers*, January, 1984, pp 79-90.