A Decomposition Procedure for Register-Transfer Level Power Management

Elena Fomina, Andres Keevallik, Margus Kruus, and Alexander Sudnitson

Abstract: *Power dissipation has become one of the most important constraints in the design of integrated circuits. This work describes a decomposition based approach for power reduction using dynamic power management. The problem of low power synthesis corresponds to an optimal multiple decomposition of a finite state machine. A decomposition procedure that enables the distribution of primary controller inputs among components is elaborated. The technique for decomposition is based on quantitative modeling through entropic relationships. The presented technique leads to a general low power design methodology targeting selective disabling of a subset of primary inputs.*

Key words: *Finite State Machine, Low Power Design, Decomposition*

INTRODUCTION

With increasing sizes of design and a need for low power applications, in addition to timing and area, power is another optimization constraint that has become critical for very large scale integration circuits. The drive towards a system on a chip has accelerated the significance of a low power design methodology. In the last ten years, research on the techniques for low power at various levels of design has intensified and the area of power consumption estimation and optimization has been covered by many authors as reviewed in [1]. Current research in low-power design focuses on the techniques on the reduction of dynamic power dissipation of the circuit. The work presented in this paper uses a fundamental source of power reduction – shutting down useless parts of a circuit. This idea is known as power management. Power management can be applied at different levels of the design process of digital circuits and systems. The main feature at the register transfer level and at the logic level power management is that the shutdown of hardware is decided on every clock cycle, hence the name dynamic power management.

With regard to the analysis of different techniques for dynamic power management, our work proceeds from the fact that a detection on a per-clock-cycle basis the parts of design which are idle is a substantial problem of design. We consider this problem as a particular task of the decomposition of FSM and apply the guarded evaluation style of implementation [1]. To avoid unnecessary switching activity we partition original FSM into the network of component FSM with a restricted number of binary inputs and place guards at the inputs of those parts of the network that need to be selectively turned off.

Various techniques have been developed to enhance the capability and efficiency of decomposition, and they fall broadly into two categories: those based on the algebraic theory [2] and those based on the factorization or on the identification in the state transition graph of subroutines [3]. In the last ten years, research on decomposition techniques for low power have intensified and a range of techniques has been proposed for the register-transfer level optimization of circuits for low power using the second approach [4]-[6]. As distinct from previous work in this paper conceptually more general algebraic theoretical background for decomposition is presented. Our reasoning proceeds from the premise that the solution of the problem of FSM synthesis for low power can be reduced to the FSM decomposition with distributed primary input/output variables and appropriate synthesis of FSM network.

Our work proceeds from the fact that the principal NP-complete problem of FSM decomposition is searching a set of partitions on the set of states of source FSM. As in [2], only such sets of partitions may be used for FSM decomposition. Because of lack of methodology of searching of these partitions the application of the powerful algebraic decomposition theory is substantially limited in practice. We attempt to surmount this obstacle.

DECOMPOSITION MODEL

Formally, the FSM is defined as a quintuple $< S, X, Y, \delta, \lambda >$, where

$S=\{s_1, s_2, \ldots, s_M\}$ is a set of states,

$X=\{x_1, x_2, \ldots, x_L\}$ is a set of binary input variables,

$Y=\{y_1, y_2, \ldots, y_T\}$ is a set of binary output variables,

δ: $D(\delta)\rightarrow S$ is a multiple valued next state function with a domain $D(\delta)=D_1\times\ldots\times D_L\times S$ and codomain S,

$D_i=\{0, 1\}$ represents a set of values (symbols) each input variable x_i may assume,

λ: $D(\lambda)\rightarrow R(\lambda)$ is an output function with a domain $D(\lambda)=D(\delta)$ and codomain $R(\lambda)=E_1\times\ldots\times E_T$,

$E_i=\{0, 1\}$ represents a set of values each output variable y_i may assume.

The behaviour of a controller can be described by state transition graph or, equivalently, by presentation by the list of transitions.

Table 1. An example FSM

Present state s_p	Input condition α_{pq}	Next state s_q	Output signal β_{pq}
1	x_2	*1*	y_1
	$\hat{}x_2$	*2*	$y_1 y_2 y_7$
2	x_1	1	y_1
	$\hat{}x_1 \& x_3$	3	$y_2\ y_5$
	$\hat{}x_1 \& \hat{}x_3$	5	$y_1 y_2 y_6$
3	$x_2 \vee \hat{}x_2 \& \hat{}x_4$	3	$y_3\ y_7$
	$\hat{}x_2 \& x_4$	4	y_3
4	x_4	1	$y_5\ y_6$
	$\hat{}x_4$	2	$y_3\ y_5$
5	x_2	3	$y_4 y_7$
	$\hat{}x_2$	4	y_4

In this table, input condition is a Boolean function, α_{tq}, which is equal to 1 when the controller makes the transition from the state s_t to state s_q. Output signal is a microinstruction, β_{tq}, the list of output signals which are equal to 1 on the transition of the FSM from s_t to s_q. The search for the next state and the corresponding output means the evaluation of the Boolean functions α on the Boolean space $\{0, 1\}^L$.

Our decomposition procedure is based on the general form of decomposition without the restriction on their interconnection (Figure 1). Informally, the essence of the decomposition task could be described as follows. Given a prototype FSM description of a desired terminal behavior, the decomposition problem is to find sub-machines which, when interconnected in a prescribed way, will display that terminal behavior. Our procedure of decomposition is based on the general form of decomposition without the restriction on their interconnection. Each sub-FSM corresponds to a partition on the set of states (a partition π on the set of states, S, in a machine is a collection of disjoint subsets of states whose set union is S). In general decomposition, each partitioned machine has information about the current state of the others.

The state behavior of the FSM network forms the basis of the decomposition model. The state behavior of the prototype machine is formally described by the network of state machines $A_i=<X_i , S_i , \delta_i>$ where S_i is the set of states which correspond to blocks of partition π_I, $X_i=Z_i\cup E_i$, where Z_i is a set of internal symbolic variables (state variables) and $E_i\subseteq X$ is a set of external inputs. Each of the sub-machines receives, as inputs, not only the primary inputs and their own state variables, but also the state variables of the other sub-machine. $\delta_i: D(\delta_i) \rightarrow S_i$, is a transition function.

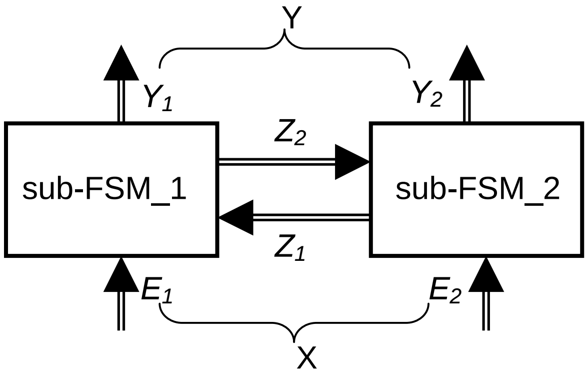

Fig. 1: Structure of decomposed machine

Our approach to the decision of partition choice problem is based on the new notion of partition with Don't Care's and its relation to pair algebra.

To distribute input variables among component FSMs, we introduce the notion of α-partition with Don't Care's.

A partition with Don't Care's (PDC) ρ of a set *S* is a collection of disjoint nonempty subsets of *S*.

The disjoint subsets are called blocks of ρ and their set union is equal to $S_d \subseteq S$. The set difference $S \setminus S_d$ is the Don't Care's area of the PDC and we can consider it as some distinguished (special) block which may be empty.

In our reasoning for partitions search, we proceed from information theoretic concepts, which are rationalized on the basis of algebraic structure theory of sequential machine. In the following, we assume that the state lines of the FSM are modeled as a Markov chain. Entropy is related to switching activity, that is if the signal switching is high, it is likely that entropy is also high. Theoretically confirmed the high correlation proves that partition entropy is suitable for estimating corresponding sub-machines, which makes it a good measure for partition choice for appropriate decomposition.

DECOMPOSITION PROCEDURE

The idea of partition for low power here is that in behavioral descriptions of hardware, a small set of computation (computational kernel) often accounts for most of the computational complexity as well as power dissipation [5]. We extract a computational kernel during the decomposition process. It enables us then to simplify the computational kernel in a stand alone manner to achieve power savings. To find kernel, primary inputs with high probability have to be selected. So, we decompose the example prototype machine into network of two sub-FSMs. The procedure of decomposition is divided into six phases.

1) Finding of primary α-partitions.

We use the representation of Boolean functions with complexes of cubes [7]. Two products (cubes) *C* and *C'* are in the relation of consensus (*C con C'*) if and only if they have opposite values (*0* and *1*) exactly in one bound component. Two covers K_1 and K_2 are in consensus if and only if there are $C \in K_1$ and $C' \in K_2$, which are in consensus.

For every $x \in X$, we define such symmetric binary relation ω on *S* that $s_p\ \omega\ s_q\ (p \neq q)$ if and only if for some s_t exist transitions $\langle s_t,\ s_p,\ \alpha_{tp} \rangle$ and $\langle s_t,\ s_q, \alpha_{tq} \rangle$ such that the correspondent input conditions α_{tp} and α_{tq} are in consensus. As a result of transitive closure operation of relation ω we will receive symmetric and transitive relation on *S* which we represent as PDC and call it *primary α-partition* on *S.*

Let $x \in X$ and $\alpha(x)$ is PDC of S, then if both s_l and s_j are contained in the same nonspecial block of $\alpha(x)$, s_i and s_j are "indistinguishable" by the input variable (channel) x. According to the definition of the primary α-partition for the fixed binary input it is a such partition on the set of states that the work of corresponding component FSM does not depend on the chosen binary input.

In our example, we have four primary α-partitions: $\alpha(x_1)$= {{1, 3, 5}, <2, 4>}, $\alpha(x_2)$ = {{*1, 2*}, {*3, 4*}, <5>}, $\alpha(x_3)$= {{3, 5}, <1, 2, 4>}, and $\alpha(x_4)$ = {{*1, 2*}, <3, 4, 5>}.

2) Constructing of the α-partition.

The PDC ρ in reality defines a set of conventional partitions, denoted by $G(\rho)$, generated by distributing the elements of the distinguished block over the other blocks of the PDC and over new created blocks in all possible ways.
The next affirmation is important for receiving of decomposition with the distribution of the inputs. Let $\alpha(X^*)$ be the sum of all α -partitions $\alpha(x_i)$ such that $x_i \in X^*$ and A_i a component FSM constructed in accordance with some partition π_i from $G(\alpha(X^*))$, then the behavior of A_i does not depend on all binary inputs of from X^*.

To meet the demand that selection logic, depending on the input patterns, selects either the kernel or the rest of the circuit (in a mutually exclusive fashion) the restriction is that for all states $s_t \in S$, $X(s_t) \subseteq X_c$ or $X(s_t) \subseteq X^*$. Here, $X(s_t)$ is the set of inputs essentially determining the transition (next state) from the state s_t. We sum the primary α-partitions with the greatest entropy to receive the computational kernel (the first sub-FSM). In the following, we assume that as result of selection the set X_c={x_1, x_3} is selected.

In this step we generated the first partition. For every input variable $x \in X^*$={x_2, x_4} which must to be excluded generate $\alpha(x)$. By summing of PDCs corresponding to elements of X^*, we construct the new partition: $\alpha(x_2, x_4) = \alpha(x_2) + \alpha(x_4)$ ={{1, 2}, {3,4}, <5>}

3) Determining of the complete set of partitions.

Generate the partitions π_1 and π_2 such that $\pi_1 \in G(\ \alpha(x_2, x_4)\ )$ and their product is zero-partition, $\pi_1 \bullet \pi_2 = 0$.

In our example, π_1 = {{1, 2}, {3, 4}, {5}}; π_2 = {{1, 3, 5}, {2, 4}}.

4) Coding of the network.

The coding of global states of the network gives us a set of internal binary variables of the network Z. Consider a set of states S and an encoding function $e: S \rightarrow \{0, 1\}^c$, for a given c (encoding length), that to each symbol $s \in S$ a code, i.e., a binary vector of length c. A necessary requirement is that different symbols are mapped to different binary vectors. Given a set of symbols S, a face constraint is a block $B \subseteq S$ in the partition specifying that the symbols in B are to be assigned to one face (or sub-cube) of a binary c-dimensional cube, without any other symbol sharing the same face. So, face constraints are generated by step of partition search, c is the number of internal binary variables of the net, $|Z|$. Every variable $z \in Z$ corresponds to some two-block partition on S. Let the binary internal state variable z^i_j be produced by the sub-machine A_i. Then z^i_j is a state variable of sub-machine A_i and corresponds to the two-block partition h^i_j. One of the blocks of h^i_j is coded by 0, the other one by 1. In this step we decide an combinatorial problem called face hypercube embedding [7], to find the minimum c and related $e: S \rightarrow \{0, 1\}^c$ such that face constraints are satisfied i.e., $h_{ij} \leq \pi_i$.

In our example, corresponding two-block partitions and internal binary variables are: $z_1 \sim h^1_1$ = {{1, 2, 5}, {3, 4}}; $z_2 \sim h^1_2$ = {{1, 2}, {3, 4, 5}}; $z_3 \sim h^2_1$= {{1, 3, 5}, {2, 4}}.

5) Determining of the structure of the network.

For a subset B of S, we define $\delta(B,\sigma)=\{s \mid s=\delta(t, \sigma), t \in B\}$ and we say that the state subset B goes into set B' under input σ if and only if $\delta(B, \sigma) \subseteq B'$. Pair (π_1, π_2) is a state-state pair if and only if π_1 and π_2 are partitions of S and for all inputs σ, $s \sim t\ (\pi_1)$ implies $\delta(s, \sigma) \sim \delta(t, \sigma)\ (\pi_2)$. Thus (π_1, π_2) is a partition pair on A if and only if the blocks of π_1 are mapped into the blocks of π_2 by *A*. That is, for every input σ and $B_{\pi 1} \in \pi_1$, there exists a $B_{\pi 2} \in \pi_2$ such that $\delta(B_{\pi 1}, \sigma) \subseteq B_{\pi 2}$. In other words, if we only know the block of π_1 which contains the state of *A*, then we can compute for every input the block of π_2 to which this state is transferred by *A*. For any partition π on S of *A* we define the operator: $M(\pi)=\Sigma\{\pi_i \mid (\pi_i, \pi)$ *is a partition pair on A*$\}$. The operator $M(\pi)$ gives the maximum front partition of partition pair. $M(\pi_i)$ defines the information received from the other components of the net sufficient for the sub-machine A_i to compute its next state and output [5].

$M(\pi_1) = \{\{1, 4\}, \{2\}, \{3, 5\}\}; M(\pi_1) \le h^1_2 \bullet h^2_1.$

$M(\pi_2) = \{\{1, 5\}, \{2\}, \{3\}, \{4\}\}; M(\pi_2) \le h^1_1 \bullet h^2_1.$

It means that state variable of the first machine z_1, corresponding to h^1_1, is the internal input of the second component machine. The variable z_2 is the state variable and z_3 is input variable for the first sub-machine. The sub-FSMs network of our examle decomposition is presented in Figure 2.

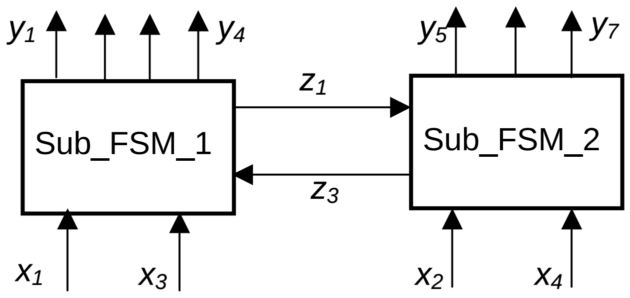

Figure 2: Example FSM

6) *Defining of the basic of the network*.

The set of states of component FSM A^i is equal to the set of blocks of partition π_i: {1, 2} ~ a_1, {3, 4} ~ a_2, {5} ~ a_3 The internal inputs of component machines are defined in the previous step of the procedure.

Synthesize the sub-FSMs corresponding to partitions π_1 and π_2. The transition table of the first sub-FSM is presented in Table 2.

Table 2. The first sub-FSM transition table

Present state	Input condition	Next state	Output signal
a_1	$\hat{}z_3$	a_1	y_1
	$x_1 \,\&\, z_3$	a_1	y_1
	$\hat{}x_1 \,\&\, x_3 \,\&\, z_3$	a_2	y_2
	$\hat{}x_1 \,\&\, \hat{}x_3 \,\&\, z_3$	a_3	$y_1\, y_2$
a_2	$\hat{}z_3$	a_2	y_3
	z_3	a_1	y_3
a_3	*1*	a_2	y_4

EXPERIMENTAL RESULTS AND CONCLUSIONS

Our reasoning proceeds from the premise that the solution of the problem of FSM synthesis for low power can be reduced to the FSM decomposition with distributed primary input variables and appropriate synthesis of FSM network. In the next section we apply the developed concepts to finding corresponding decomposition partitions of prototype FSM.

WWW-based system is developed [8]. Experiments have been carried out on the set of well-known FSM benchmarks [9] to certify the viability of our concepts. Preliminary results confirmed that it is possible to significantly reduce switching activity of implementation and that significant reduction in power consumption could be achieved.

The developed Web-based design system can be considered as a research tool that we use to carry out experiments guided to further development of decomposition synthesis.

Table 3: Results and comparison of implementations

Circuit	State #	Inputs #	Total states # of sub-FSMs	Total states # (alternative approach)	Max # of sub-FSMs inputs	Area noncombin. ratio	Area combinat. ratio
log	17	9	12	19	5	0.75	0.85
dvram	35	8	14	37	5	0.50	0.52
nucpwr	29	13	15	31	8	0.68	0.67
sync	52	19	26	54	13	0.67	0.64
planet	48	7	24	50	5	0.59	0.56
ex6	8	5	9	10	3	1.14	1.09
opus	10	5	7	12	4	0.78	0.75
ex4	14	6	9	16	5	0.75	0.76
rie	29	9	15	31	5	0.66	0.64

Some results of experiments are presented in Table 3. The table contains the results of comparative experiments of our decomposition technique and approach used in [4]-[6]. The area estimation was done using the commercial design frame (SYNOPSIS). This parameter was chosen for complexity criteria for decomposition system.

ACKNOWLEDGEMENT

This work has been supported by the Estonian Science Foundation (under Grant 5643) and by EU V Framework projects IST-2001-37592 and IST-2000-30193.

REFERENCES

[1] Benini, L., De Michelli G., and E Macii. Designing Low-Power Circuits: Practical Recipes. IEEE Circuits and Systems Magazine, pp. 7-25, Vol. 1, No. 1, 2001.

[2] Hartmanis J. and R. E. Stearns. Algebraic Structure Theory of Sequential Machines. N.J.: Prentice-Hall, Englewood Cliffs, 1966.

[3] Ashar P., Devadas S., and A.R. Newton. Sequential Logic Synthesis, Boston: Kluwer Academic Publishers, 1992.

[4] Hwang E., Vahid F., and Y.-C.Hsu. FSMD Functional Partitioning for Low Power. Proc. DATE Conf. , pp.22-28, March 1999.

[5] Lee M.H., Hwang T.T., and S.-Y. Huang. Decomposition of extended finite state machine for low power design. Proc. DATE Conf. , pp. 1152 – 1154, 2003.

[6] Monteiro J.C. and A.L.Oliveira. Implicit FSM decomposition applied to low-power design. IEEE Trans. on VLSI Systems, vol.10, No. 5, pp.560-565, October 2002.

[7] De Micheli G., Synthesis and Optimization of Digital Circuits, N. Y.:McGraw-Hill, 1994.

[8] Decomposition Applets. Available: http://www.pld.ttu.ee/dildis/automata/applets

[9] K. McElvain, LGSynth'93 Benchmark Set: V. 4.0, 1993.
Available: http://www.cbl.ncsu.edu/benchmarks.

ABOUT THE AUTHOR

Ph.D. Student E. Fomina, Prof. A. Keevallik,
Ph. D., Assoc. Prof. M. Kruus, Ph.D., Assoc. Prof. A. Sudnitson, Ph.D.,
Department of Computer Engineering, Tallinn Technical University, Raja 15, 12617 Tallinn, Estonia, Phone: +372 620 2251, E-mail: alsu@cc.ttu.ee