Decomposition of Extended Finite State Machine for Low Power Design

MingHung Lee, TingTing Hwang, Shi-Yu Huang
Department of Computer Science
National Tsing Hua University, HsinChu, Taiwan 30043

Abstract

Power reduction can be achieved by turning off portions of circuits that are idle. Unlike previous work which focused only on either controller or data-path, we propose a decomposition technique that takes both controller and data-path into consideration where, in the former, the state probability of an FSM provides the execution frequency of operations in that state. And, in the latter, operations performed in states provide the information of resource requirements which can be used to determine the resource sharing among states. The goal is to synthesize circuits such that the submachine with small area (data-path and controller) will be turned on most of the time (high state probability) and all other parts are turned off. Our experimental results show that on the average, 10% area reduction and 24% power reduction can be achieved as compared with designs without decomposition.

I. Introduction

In this paper, we will focus on the techniques that synthesize data-path and controller at architectural-level synthesis process. The input abstract model used at architectural-level is a data flow graph. Its result can be viewed as the composition of a number of data-path components and control blocks. Since there are strong two-way interactions between data-path and control components, research efforts[1], [2] have been devoted to stitching together these two types of components and a single abstract model called Extended Finite State Machine (EFSM) model or Finite State Machine with a datapath (FSMD) has been proposed.

We found that the coupling relations of a state of FSM and the datapath operations executed in the state were not explored before. For example, the state probability of an FSM provides the execution frequency of operation in that state. Operations performed in states provide the resource requirement that can be used to determine the resource sharing among states. Therefore, in this paper, we propose a partition method to synthesize EFSM for low power design taking into consideration coupling relations of states and their corresponding operations.

We propose to partition the EFSM into several sub-EFSMs taking state probability and resource sharing into account. Then, each sub-EFSM is synthesized to have its own controller and datapath such that the area of each submachine is smaller than the original EFSM. At any time only one sub-EFSM is active while the others are idle. By turning off idle circuits, the switching activity can be reduced and power consumption minimized.

Supported in part by a grant from the National Science Council of R.O.C. under contract no. NSC-90-2215-E-007-043

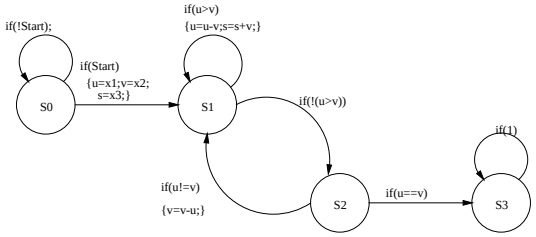

Fig. 1. An EFSM model for a circuit

II. Architecture for Decomposed EFSM

In an EFSM model, the transition can be expressed by an "*if statement*" consisting of a set of *trigger conditions* and a set of *data operations*. If the trigger condition is satisfied, the data operation is executed and the machine is brought from the current state to the next state. Figure 1 shows an example.

Given an Extended Finite State Machine, a synchronous design composed of a number of datapath components and control blocks can be synthesized. The control block is a conventional Finite State Machine that realizes the state transition graphs of the EFSM model. The datapath block evaluates the trigger conditions associated with each transition and performs the data operation. Figure 2 shows one architecture to implement a single EFSM example of Figure 1.

Suppose we have a transition from $S2$ to $S1$. In the present state $S2$, the *subtracter* performs $v - u$ and the *comparator* performs the comparison of data from registers u and v. Then, the comparison-result will feed back to EFSM. From the information of the present state and the feed-back signal, the machine will transit from $S2$ to $S1$ and register v load the result of *subtracter*. Note that in Figure 2, although the result of functional unit *adder* is not needed in this cycle, it will also be active because one of its input is from register v.

We proposed a decomposed architecture as shown in Figure 3 where the original EFSM is decomposed into two sub-EFSMs: $EFSM1$ and $EFSM2$. In this decomposition, $S0$ and $S1$ are in $EFSM1$, and $S2$ and $S3$ in $EFSM2$. Two submachines have their respective datapath components: $datapath1$ and $datapath2$ and clock gating technique is used to turn on/off the two datapaths.

There are two types of transitions we must consider. The first type is a transition occurred within a submachine, called *inner edge transition*. When this type of transition occurs, only the submachine with the active transition is on and all other submachines are turned off. The second type is a transition from the present submachine to the

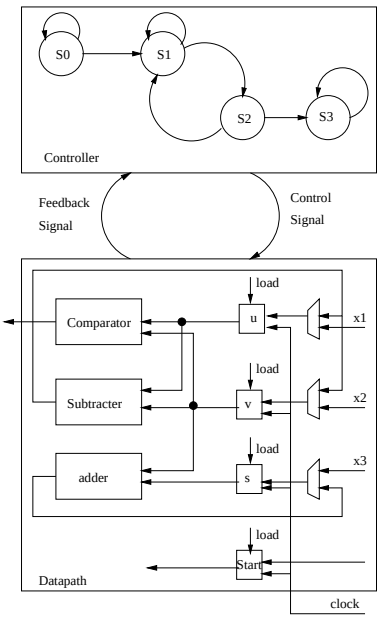

Fig. 2. Implementation of a single EFSM

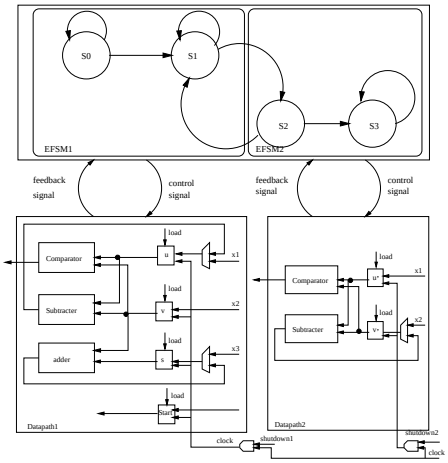

Fig. 3. Implementation of decomposed EFSM

next submachine, called *crossing edge transition*. When this type of transition occurs, both the present submachine and the next submachine are on and all other submachines are turned off.

For an inner edge transition, transition occurs within one sub-EFSM, power consumption is incurred in only one sub-EFSM and only one datapath. For a crossing edge transition, while in the controller component, both submachines involved in the transition will be on [3], in the datapath component, only one datapath will be on. Therefore, to reduce power consumption, the goal is to decompose an EFSM such that the probabilities of state transitions within a sub-EFSM are high and the probabilities of state transitions between two submachines are low. Moreover, the submachine with high state probability will have small datapath (area).

TABLE I

Benchmark Set

Name	#s	characteristics
$Mont$	6	compute the inverse of a number
$5GCD$	10	GCD generator from five numbers
$Control$	10	a small micro-controller
$FMult$	12	evaluates the floating point multiplier
$FDiv$	12	evaluates the floating point divider
$Chinese$	29	evaluates the Chinese Remainder Theorem

TABLE II

Comparisons of Design with/without Decomposition

	Area		Power in mW		Cycle Time	
Name	Orig.	Ours	Orig.	Ours	Orig.	Ours
$Mont$	821	805	32.09	25.29	0.34	0.34
$5GCD$	716	632	30.28	22.83	0.31	0.31
$Control$	963	876	32.82	23.43	0.34	0.34
$FMult$	2369	2115	33.63	24.51	0.51	0.50
$FDiv$	3117	2753	37.98	28.15	0.51	0.50
$Chinese$	5413	4784	160.21	122.70	0.48	0.47

III. The Design Procedure

The input consists of a design described by EFSM model and the state/transition probability of the machine. In the design procedure, the first step is to partition the EFSM model into n submachines, where n is given. These submachines are said to be coupled in the sense that state transitions take place either within a submachine or between two submachines. After the system is partitioned into a number of submachines, the next step is to perform controller and datapath synthesis for each submachine. For datapath synthesis, functional unit and then variable bindings are performed. In each submachines, resource sharing technique are utilized to reduce power and area. Our objective is to minimize the number of functional units/registers in each submachine. For controller synthesis, state assignment is performed. After synthesis of datapath and controller, RTL description in Verilog HDL is generated using muxes and functional units such as adders, multipliers and registers. Generated RTL code is synthesized using Synopsys Design Compiler with a 0.35 um cell library from TSMC.

IV. Experimental Results

Column ratio is the improvement ratio which is defined as $(Original - Ours) / Original$.

Different EFSM examples were tested. Table I shows the description of our benchmark examples. The experiment in Table II is to compare our results of decomposition with those results without decomposition. Our results are the best results selected from 2,3,4-way decompositions in terms of power consumption.

References

[1] Chien-Jyh Liu and Shi-Yu Huang, "Low-Power Synthesis For Extended Finite State Machines", *Proc. of the 12th VLSI/CAD Symposium*, 2001.

[2] D.D. Gajski. "Principles of digital design", Prentice Hall, 1997.

[3] S-H. Chow, Y-C. Ho, T-T. Hwang and C. L. Liu, "Low Power Realization of Finite State Machines-A Decomposition Approach", *ACM Transactions on Design Automation of Electronic Systems*, vol. 1, no. 3, pp. 315-340, 1996.