MIXDES 1998

TITLE: CHECKING SEQUENCE DESIGN FOR FSM

AUTHORS: A.KEEVALLIK, M. KRUUS, H. LENSEN
TALLINN TECHNICAL UNIVERSITY, ESTONIA

KEYWORDS: FINITE STATE MACHINE , TESTABILITY, DISTINGUISHING SEQUENCE, CHECKING SEQUENCE

ABSTRACT: In this paper we present a new checking sequence design method for finite state machines (FSMs). Microprogram automaton model is used to describe source FSM. The proposed method enables to compose a universal checking sequence which will be independent of FSM implementation. The fault classification for microprogram automaton (MPA) model is introduced. It is shown that composed sequence checks all the observed faults. A design for testability method is proposed to guarantee the existence of a short distinguishing sequence for MPA and reduce the length of checking sequence. The introduced methods are illustrated by examples. Experimental results on MCNC FSM benchmark examples show that the most of real complexity digital control units have a short distinguishing sequence and checking sequences composed by our method are considerably shorter than the upper bound shows.

Introductions

For a long time the formalized methods for digital control systems logical design, based on finite automata theory, were not widely used in practical design. The main reason was the high combinatorial complexity of optimization tasks. However, with the growth of the integration degree of digital components the situation is changed and highly efficient formalized design methods present today the only way to guarantee design correctness and quality. Most of the modern formalized methods for digital control units design use the Finite State Machine (FSM) model to describe the source unit. But it is also well-known that FSMs are difficult to test. The test methods, based on checking experiments theory [1,2] are not often used because of the high upper bound of checking sequence length. In the same time this approach can be acceptable if FSM is implemented on a single chip (like PLA with internal register) and the present state of FSM is not directly observable.

In this paper, we propose to use so-called microprogram automaton (MPA) model [3] to compose an universal checking sequence which will be independent of FSM implementation. The fault classification for MPA model is introduced. It is shown that composed sequence enables to check all the observed faults in MPA. A testable design method is proposed to guarantee the existence of distinguishing sequence [1,2] for MPA and reduce the length of checking sequence. The proposed methods are illustrated by examples. Results of the experiments using the MCNC FSM benchmark examples are provided and analyzed.

Basic Notations

Our basic research object is regarded as Microprogram Automaton (MPA). This model is quite similar to the ordinary initial FSM model, but includes sets of binary vectors as input and output alphabets.

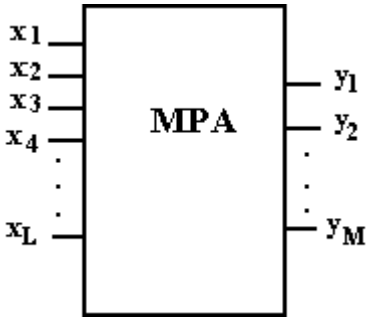

Fig. 1. MPA Model

Microprogram Automaton A is defined [3] as system $A = \left(\{0,1\}^L, S, \{0,1\}^M, \delta, \lambda, s_0 \right)$,

where

$\{0,1\}^L$ is input alphabet, L - the number of binary inputs;

S - set of internal states, $|S| = n$;

$\{0,1\}^M$ - output alphabet, M - the number of binary outputs;

$\delta: S \times \{0,1\}^L \to S$ - transition function;

$\lambda: S \times \{0,1\}^L \to \{0,1\}^M$ - output function;

s_0 - initial state of MPA.

The input and output vectors are denoted by $x_1x_2\ldots x_L$ and $y_1y_2\ldots y_M$,respectively.
MPA *A* can be presented by the list of its transitions (*Tab.1.)*, similar to the state and output tables of FSM.

Any row from this list describes one of generalized transitions (G-transitions) of MPA. The set of MPA transitions is determined by the set of generalized input vectors (G-inputs) $\tilde{x}_1\tilde{x}_2\ldots.\tilde{x}_L$, where $\tilde{x}_i \in \left\{x_i, \bar{x}_i, \times\right\}, 1 \le i \le L$ and $\tilde{x}_i = \times$ denotes that described G-transition is not dependent of input x_i. For any transition of MPA there exist the sets of essential inputs (on which transition and output functions depend) and inessential inputs. For essential inputs $\tilde{x}_i \in \left\{x_i, \bar{x}_i\right\}$, for inessential inputs $\tilde{x}_i = \times$. Any row of transition list describes the set of elementary transitions (E-transitions).

Number of transition	Initial state	Final state	Generalized input vector	Output vector
1	s1	s2	x1	y1y2
2		s2	-x1x2	y1y3
3		s5	-x1-x2	y1y2
4	s2	s1	-x1-x3	y3y4
5		s2	x1-x3	y4
6		s3	x3	y1y2y3
7	s3	s1	-x1-x4	y3y4
8		s2	x1-x4	y3
9		s4	x4	y2y3y4
10	s4	s1	x3	y1
11		s2	-x3	y4
12	s5	s4	1	y3y4

Table 1. Transition list of MPA A

For example, G-transition in row 7 describes four E-transitions from state s_3 to state s_1 under input vectors 0000, 0010, 0100, 0110. For this generalized transition $\left\{x_1, x_4\right\}$ and $\left\{x_2, x_3\right\}$ are the sets of essential and inessential inputs. The number of generalized transitions in the list is denoted as *H*.

Distinguishing sequence design for MPA

In this section we will discuss about distinguishing sequence design method for MPA. Our approach is based on the checking experiments theory [1,2], but some useful properties of MPA model give us possibility to find a short DS in the most of cases. If it is not possible, we will apply the input or output expansion to guarantee the existence of a short DS.

Let $p = X_1X_2\ldots X_k, X_i \in \{0,1\}^L$ be a sequence of input vectors. Length of sequence *p* is denoted by *d(p).* The set of finite length input and output sequences is denoted by $\left\{\{0,1\}^L\right\}^*$ and $\left\{\{0,1\}^M\right\}^*$, respectively.

Function $\bar{\delta}: S \times \left\{\{0,1\}^L\right\}^* \to S$ is said to be a **generalized transition function** of MPA *A*. $\bar{\delta}(s,p)$ defines the final state of MPA if input sequence *p* is applied in the initial state *s*.

Function $\bar{\lambda}: S \times \left\{\{0,1\}^L\right\}^* \to \left\{\{0,1\}^M\right\}^*$ is said to be a **generalized output function** of MPA *A* and $\bar{\lambda}(s,p)$ defines the output sequence if input sequence *p* is applied in the initial state *s*.

Input sequence *p* is said to be **the distinguishing sequence** [1] *(DS)* for MPA *A*, iff for any pair of states

$$s,t \in S \quad \bar{\lambda}(s,p) = \bar{\lambda}(t,p) \Leftrightarrow s = t.$$

In the following, MPA *A* is regarded as **k-testable** iff there exist the *DS* for MPA *A* and *d(DS)=k.*

The main problem of FSM testing is usually regarded as an identification problem: checking sequence α must distinguish a correct (fault-free) machine from all other (faulty) ones. In common case the checking sequence α for FSM *A* can be constructed by the Hennie's method [1] and it must check all states and transitions of FSM *A* (by applying DS after any transition under check). The length of checking sequence essentially depends upon the number of transitions under check and the length of distinguishing sequence.

MPA has usually a considerable redundancy of outputs. This property is very characteristic for real digital control units. Almost any G-transition has a unique output reaction. Therefore, majority of MPAs have a very short *DS.* For our example there are only some G-transitions with similar output reaction. It is enough ordinary in practical design that any input vector can be regarded as *DS* for MPA.

Described property enables us to avoid the high complexity algorithms for *DS* design [1,2], based usually on investigation of FSMs successor tree. Of course, there are possible rare MPAs with long *DS*. In these cases we assume that the methods of testable

design can be applied (for example, by introducing some extra inputs and/or outputs).

Let us denote by *g(i)*, $1 \le i \le H$, the number of E-transitions described by G-transition in row *i* and by *U* the set of all G-transitions of MPA *A*. **Following method enables to find *DS* for 1-testable MPA *A*.**

Algorithm 1

1. *i=0, k=1*
2. *i=i+1;* find the set $W \subset U$ of G-transitions from states $s_{k+1},\dots,s_n$, which have a similar output vector with generalized transition *i.*
3. If $W \neq \varnothing$ **then** include G-transition *i* into *W;* select the transition $w \in W$, which have greatest value *g(w)* in the set *W* ; ban G-inputs of other G-transitions from set *W* as *DS* ; $U = U \setminus W$.
4. If there are no more G-transitions from state *k*, **then** *k=k+1*.
5. If *k<n,* **then** goto **2**.
6. End.

Note that if there are some G-transitions with equal *g(w)* in step 3 then select such G-transition *w* which brings along minimum new restrictions for *DS*.

As a result of proposed algorithm we get the set of input vectors each of which can be used as a *DS* for observed MPA. Let us illustrate proposed method by our example. Karnough map is used to describe the steps of algorithm *(Tab.2)*.

x3x4 x1x2	00	01	11	10
00	r1	r1	***DS***	r1
01	r1	r1	***DS***	r1
11	r2	r2	***DS***	***DS***
10	r2	r2	***DS***	***DS***

Table 2. Karnough map for *DS*

Transitions from state s_1 have not similar output reactions with other states. Consequently, output vectors y_1y_2 and y_1y_3 can be used to distinguish state s_1 with no restrictions on input vector. G-transition 4 from state s_2 has similar output reaction with G-transitions 7 and 12. Since transition 12 is unconditional, output reaction y_3y_4 must be attached to state s_5. Therefore we have restrictions on input vector denoted by r1 on Karnough map. G-transition 5 has a similar reaction with G-transition 11. Hence there are four E-transitions described in row 5 and eight E-transitions in row 11, lets ban G-input of transition 5 (restrictions are denoted by r2). No other similar output reactions occurs in our MPA. Undaged input vectors can be used as *DS* for MPA *A*. These vectors are denoted by *DS* on Karnough map.

The above algorithm fails if the source MPA is not 1-testable. But 1-testability can be achieved by introducing some extra binary inputs or outputs. Lets discuss about these possibilities. MPA *B (Tab.3)* is not 1-testable (moreover, it also has not longer *DS*), but 1-testability results from introducing an extra input x_3. For MPA *B′ (Tab.4)*, where $\delta(s_i, x_3) = s_i$ and $\lambda(s_i, x_3)$ gives a unique reaction for any state s_i, G-input x_3 can be used as *DS*. 1-testability of MPA *B* can also be achieved by introducing one extra output (y_4) as it is done for MPA *B″ (Tab.5)*. An extra output y_4=1 for G-transitions 2 and 3 and y_4=0 for other G-transitions.

As a result, G-input $\bar{x}_1$ can be used as a *DS*. Denote that he number of extra outputs z may be greater than 1 and in common case it can be estimated as follows: $z \le int(\log 2(|W_{\max}|)) + 1$, where $W_{\max}$ is the set *W* with maximal power in **algorithm 1**. We assume in following that source MPA is 1-testable or 1-testability is achieved by above methods.

Number of transition	Initial state	Final state	Generalized input vector	Output vector
1	s1	s2	x1	y1
2		s1	-x1x2	y1y3
3		s3	-x1-x2	y2y3
4	s2	s1	x1	y1
5		s3	-x1	y2y3
6	s3	s2	x2	y1y3
7		s1	-x2	y1

Table 3. MPA B

Number of transition	Initial state	Final state	Generalized input vector	Output vector
1	s1	s2	x1-**x3**	y1
2		s1	-x1x2-**x3**	y1y3
3		s3	-x1-x2-**x3**	y2y3
4		**s1**	**x3**	**y1**
5	s2	s1	x1-**x3**	y1
6		s3	-x1-**x3**	y2y3
7		**s2**	**x3**	**y2**
8	s3	s2	x2-**x3**	y1y3
9		s1	-x2-**x3**	y1
10		**s3**	**x3**	**y1y2**

Table 4. MPA B'

Number of transition	Initial state	Final state	Generalized input vector	Output vector
1	s1	s2	x1	y1
2		s1	-x1x2	y1y3**y4**
3		s3	-x1-x2	y2y3**y4**
4	s2	s1	x1	y1
5		s3	-x1	y2y3
6	s3	s2	x2	y1y3
7		s1	-x2	y1

Table 5. MPA B''

Checking sequence design for MPA

The problem of checking sequence design can be regarded as task of composing the input sequence able to distinguish a fault-free FSM from all faulty FSMs. It is assumed usually that any fault in FSM does not increase the number of FSM states. FSM is considered as "black box" in this approach and the states of FSM is assumed not to be directly observable. According these assumptions, the input sequence α is said to be the checking sequence, if it passes all states and transitions of FSM (from known initial state) and final state of any transition is checked by *DS*. In previous section we reduced the length of *DS*: *d(DS)*=1. However, now we have another problem: we cannot check all E-transitions of MPA, because of the great number of binary inputs in common case. Let us discuss how to decrease the number of checkable transitions without the loss of generality of our approach.

First of all, we will design the checking sequence that enables to pass and check all G-transitions of MPA. Such a sequence is enough easy to compose: it can be build up as some traversal α of state transition graph (STG) of MPA with *DS* after any G-transition under check. The initial part of such traversal α for MPA *A (Tab.1)* can be composed from initial state s1 as follows:

State	s1	s2	s3	s4	s5	s2 ....
E-input	1011	0011	0011	0011	0111	.....

Note that *DS* can be chosen from results of **algorithm 1** *(Tab.2)* and the new G-transition under check can be regarded as DS for previous G-transition if it is possible. Checking sequence α checks actually one of E-transitions from any G-transitions under check. Inessential inputs of used E-transition are fixed by random way.

Lets introduce the fault classification for MPA and determine how these faults can be detected by sequence α. We assumed that MPA is a "black box" and only its inputs are controllable and its outputs are observable. All the faults of MPA can be divided into external and internal faults.

External faults are regarded as permanent faults on MPA inputs and outputs (like stuck-at faults in some digital implementation). All external faults are surely detectable by sequence α, since these faults change the output reaction of some G-transitions or deform the arguments of output and transition functions, which can also be detected by sequence α.

The internal faults are regarded as the faults changing the MPA "internal behaviour" and deform some G-transitions in MPA description. There can be denoted three main types of internal faults: the faults of output function, faults of the transition function and faults of G-inputs.

The faults of output function are extra or missing faults of binary outputs of some G-transition. These faults can be easily detected from MPA outputs immediately if faulty G-transition is applied. Sequence α includes all G-transitions and thus all output faults can be checked.

The faults of transition function can be described as the final state faults: $\delta\left(s_i, x\right)$ gives as result the faulty final state. Fault is detectable by applying *DS* after faulty G-transition and ,thus, can be checked by sequence α.

G-input faults means that G-input of some transition is deformed (like shrinkage and growth fault in AND-array of PLA-implementation). These faults are difficult to describe by MPA model. G-input is faulty if there is an extra binary input or there misses a required input. An extra input in G-transition means that in the faulty G-transition the number of included E-transitions is decreased (shrinkage fault). Such a fault transforms MPA under check into incompletely defined one. Fault can be detected, if two E-transitions from G-transition are included into checking sequence: the unessential inputs of first E-transition are inversed in another. Note that one of these E-transitions gives as result don't care values of transition and output functions, which can be regarded as fault sign. The missing G-input faults (the growth faults) are the most unpleasant for our approach. As the result of missing fault there are two G-transitions that are satisfied by some input vector. This is unacceptable for MPA model. MPA model is unable to decide which will be the final state and output reaction if such a fault occurs. Denote that we have not solved the state encoding task and our MPA model has abstract states. Detection of such faults is possible if we fix the transition and output function values as don't care if two G-transitions are satisfied simultaneously, which will be also regarded as fault sign.

Let us return to composed checking sequence α. Sequence α was composed as the traversal of STG and includes *DS* after any checked transition. Therefore sequence α can detect all external faults

and also all faults in transition and output functions. The G-input faults can be detected if the sequence α' is concatenated to sequence α. Sequence α' includes the second traversal to detect shrinkage G-input faults. Sequence α' is shorter than sequence α, since α' don't include DS after transitions. Note that sequences α and α' may be partially covered.

The length of sequence α has the following upper bound: $d(\alpha) \le (n+1) \times H$, where n is the number of MPA states and H is the number of G-transitions of MPA. Each of H G-transitions must be passed with following *DS (d(DS)=1)* and there is possible that transfer sequence (with length $\le (n-1)$) is necessary to reach next G-transition under check.

The length of α' can be estimated: $d(\alpha') \le n \times H$.

The length of concatenation $\alpha\alpha'$: $d(\alpha\alpha') \le (2n+1) \times H$. For our first example (Tab.1): $d(\alpha\alpha') \le 11 \times 12 = 132$ input vectors. The actual length in our example is 30 input vectors.

The complete checking sequence $\alpha\,\alpha$' can be presented as follows (Table 6). '+' in forth column shows that input vector can be used as DS for MPA A.

Initial state	Input vector	Next state	/DS/
s1	1110	s2	+
s2	1010	s3	+
s3	0011	s4	+
s4	0011	s1	+
s1	0111	s2	+
s2	0111	s3	+
s3	1110	s2	+
s2	1111	s3	+
s3	0110	s1	
s1	0011	s5	+
s5	1111	s4	+
s4	1110	s1	+
s1	1001	s2	
s2	0101	s1	
s1	0111	s2	+
s2	1000	s2	
s2	0011	s3	+
s3	1110	s4	+
s4	0000	s2	
s2	0010	s3	+
s3	0000	s1	
s1	0100	s2	
s2	1101	s2	
s2	0000	s1	
s1	0000	s5	
s5	0000	s4	
s4	1101	s2	
s2	0010	s3	
s3	1000	s2	
s2	0000	s1	

Table 6. CS for MPA A

The checking sequence for MPA *B (Tab.3)* is presented after output expansion (MPA *B'', Tab.5)*:

Initial state	Input vector	Final state	/DS/
s1	01	s1	+
s1	00	s3	+
s3	01	s2	+
s2	00	s3	+
s3	00	s1	+
s1	01	s1	+
s1	10	s2	
s2	01	s3	+
s3	11	s2	
s2	10	s1	
s1	00	s3	+
s3	10	s1	
s1	11	s2	
s2	11	s1	

Table 7. CS for MPA B''

Note that input vectors 00 and 01 can be used as *DS* after expanding.

Experimental results and conclusions

The proposed methods of checking sequences design are implemented in CAD system DILOS - the system of decompositional design of digital control units, created in Department of Computer Engineering (Tallinn Technical University). Subsystem TESTER includes next main parts related with proposed methods: DSSYN - distinguishing sequence design procedure for 1-testable MPAs; EXSYN - expansion procedure to guarantee the 1-testability of MPA; CSSYN - checking sequence design procedure for 1-testable MPA. Experimental results on MCNC benchmarks showed that the most of real complexity digital control units have the short distinguishing sequence, but 65% of observed examples were not 1-testable. The extra output or input introducing is used for these cases.

The experimental researches showed also that real checking sequences are essentially shorter than the upper bound shows and their length is 15...60 % of the estimated upper bound. Some experimental results are illustrated in *Table 8*.

FSM	Initial Inputs	Initial Outputs	States	Extra Inputs or Extra Outputs	CS upper bound	CS actual length
lion	2	3	4	0	55	27
train11	2	1	11	+2 Outputs	300	124
mark1	5	16	15	0	352	149
beecount	3	4	7	+2 Outputs	224	113
beecount	3	4	7	+1 Input	280	127
tav	4	4	4	+1 Input	265	175
tav	4	4	4	+2 Outputs	245	126
dk27	1	2	7	+2 Outputs	112	38
keyb	7	2	19	+3 Outputs	3400	907
ex1	9	19	20	+1 Output	2898	926
ex4	6	9	14	+1 Output	315	160
train4	2	1	4	+2 Outputs	70	40

Table 8. Experimental results

THE AUTHORS

Prof. Andres Keevallik is with Department of Computer Engineering, Tallinn Technical University, Tallinn, Estonia
e-mail: akeev@cc.ttu.ee

Margus Kruus is with Department of Computer Engineering, Tallinn Technical University, Tallinn, Estonia
e-mail: kruus@cc.ttu.ee

Harri Lensen is with Department of Computer Engineering, Tallinn Technical University, Tallinn, Estonia
e-mail: hl@cc.ttu.ee

REFERENCES

1. F.C.Hennie, „Fault detection experiments for sequential circuits," in Proc. 5th Annu. Symp. on Switching Theory and Logical Design, 1964, pp. 95-110.

2. A.Gill, „Introduction to the Theory of Finite State Machines", Mc-Graw Hill Book Co, 1962.

3. S.Baranov, „Synthesis of control automata with large numbers of inputs and outputs", Technical Report FC 330 MCS 038, Dep. of Math. and Comp. Science, Ben-Gurion University, Israel, 1992.