Transforming FSMs for Synthesis by Fault Tolerant Nano-PLAs

Ilya Levin, Osnat Keren
Tel Aviv University, Israel; Bar Ilan University, Israel
ilia1@post.tau.ac.il, kereno@eng.biu.ac.il

Abstract

The paper deals with designing finite state machines (FSM) by Nano PLA structures. Due to the device-missing domination in nano-structures, it is desired to minimize both the area and the number of PLA devices required for the FSM implementation. We propose a solution of the minimization problem. Our solution is based on: a) partition of the FSM for two interacting components; b) transformation of one or two of the component FSMs into a dense form. We show that combining the above techniques provides a significant area reduction in comparison with the straightforward FSM implementation.

1. Introduction

Due to the significant grow of fault occurrence in nanoelectronic circuits, and in particular in nano-PLAs, intensive fault tolerant techniques are required. The well-known way to increase fault-tolerance of a circuit is by introducing redundancy. The tautology is the most popular methods for increasing the fault tolerance (FT) of Nano PLAs. The method is based on doubling rows and/or columns of PLA plans and called tautology. These methods basically double empty crosspoints of the PLA, which results in unreasonable overhead. PLA crosspoints may be with or without devices. PLA including a small percent of devices is considered to have low density. Due to the device missing domination in nano-PLAs, the PLA with a more number of devices (dense PLA) seems less fault-tolerant. Thus, both the number of PLA devices and the PLA area overhead are considered as optimization criterion in synthesis of logic circuits by nano-PLA.

The methods presented in [1][2] are based on tautology and allow obtaining rather efficient implementation, in comparison with the known TMR approach. Methods described in [1], have become the starting point of research works [3][4]. The first of these works suggests "hardening" of such portions of PLA, which are most critical to the potential faults. For example, one of the proposed algorithms comprises gradually adding "hardening" tautologies into PLA, up to reaching a required degree of fault-tolerance, where the degree of fault-tolerance is estimated according to analytical techniques taking into account probability of faults as a function of a specific logic structure. The above approach, being a continuation of [1], allows obtaining PLA-implementations with predetermined fault tolerance features and with the best overhead results, in comparison with the overhead obtained using a conventional approach.

The second paper [4] develops an approach, which is also based on tautologies. It proposes a a compression of the initial system combined with the tautology. The method is intended for implementing of a specific class of logical systems, namely - of combinational portions of sequential circuits. The method proposes a dense transformation (compression) of the initial system of logic functions that leads to reducing the area required for the circuit implementation with negligibly increasing the number of devices. The paper demonstrates that the proposed compromise between the area and the number of devices can be utilized for selecting the desired PLA implementation. For the majority of benchmarks, the method [4] leads to a significant improvement in comparison with known schemes of pure tautology [1][2]. However, there are cases where the dense structure doesn't improve the straightforward implementation of the initial FSM. The aim of the present paper is to provide an area reduction also for such problematic cases.

We propose to divide the initial FSM into a network of two interacting component FSMs. Our approach is motivated by the following reasons.

1. After the partitioning, each of the component FSMs may have smaller number of input variables than the initial FSM, which leads to the total PLA area reduction.
2. There are FSMs whose dense implementation [4] is inefficient. The proposed partitioning can allow to apply the dense transformation to one or to both of the component FSMs and to achieve the total area reduction in comparison with the initial FSM.

The paper is organized as follows. The second section presents a motivation example. The third section describes the proposed partitioning approach. Evaluation of the proposed solution is presented in the forth section. Conclusions are provided in Section 5.

2. Motivation example

A PLA-description of FSM is a matrix comprising both dense and sparse fragments (see, for example, Table I). A sparse PLA may be compressed in order to obtain an equivalent dense structure of smaller area. A dense PLA structure is proposed in [4]. This structure consists of three portions: Input transformation PLA, Core PLA and the Output transformation PLA. In the dense structure, two data compressions are applied: the compression of the input variables, which is a transformation of the initial AND plane, and the compression of the output variables, which is a transformation of the initial OR plane.

Benchmark results reported in [4] demonstrate that the effectiveness of the dense transformation depends of the structure and density of the initial description. However, if the initial FSM includes a specific state with transitions depending of a large number of input variables, the corresponding dense structure may require a large area. Consequently, it means that the FSM includes this "undesirable" state that, being removed from for FSM, would allow reducing the area drastically. It leads to an idea for improvement of the method [4] by partitioning of the initial FSM for two interacting component FSMs. After the partitioning, each of the component FSMs can be transformed and implemented separately allowing utilizing specific characteristics of each of the components. Moreover, the partitioning may bring additional area reduction even without the dense transformation. Indeed, if the component FSMs are defined on small subsets of input variables the partitioning may reduce the total PLA area. The partitioning of FSMs is in the focus of our study.

Let us illustrate this idea by partitioning the FSM example shown in Table I.

Table I. Example of PLA based FSM

	Inputs									Outputs	
Present state	x1	x2	x3	x4	x5	x6	x7	x8	x9	Next state	Y
a1	1									a2	Y2
	0	1								a3	Y4
	0	0								a4	Y4
a2				0						a1	Y3
			0	1						a2	Y1
			1	1						a3	Y3
a3					1					a1	Y4
					0	1				a2	Y3
					0	0	1			a3	Y1
					0	0	0	1		a4	Y0
					0	0	0	0	1	a4	Y1
					0	0	0	0	0	a1	Y2
a4									0	a3	Y2
								0	1	a2	Y1
							0	1	1	a1	Y4
						0	1	1	1	a3	Y3
					0	1	1	1	1	a4	Y3
					1	1	1	1	1	a1	Y1

In this table, the first column corresponds to the present state of the FSM. Columns from 2 to 8 correspond to input variables. The 9-th column corresponds to the next state of the FSM. The 10-th column indicates the output vector. In our paper, for simplicity, we ignore outputs of the FSM and don't indicate outputs in our FSM examples. Rows of the table are in one-to-one correspondence with transitions of the FSM.

The straightforward implementation of the above FSM has an area 468 bits. The corresponding dense structure for the example is shown in Table II.

Table II. Dense implementation of the FSM: Input transformation PLA (left); Core PLA (right)

Inputs										Outputs				
x1	x2	x3	x4	x5	x6	x7	x8	x9	Present state	p1	p2	p3	p4	p5
1									a1	1				
	1								a1		1			
		1							a1	1				
			1						a2		1			
				1					a2	1				
					1				a2		1			
						1			a3			1		
							1		a3				1	
								1	a3					1
				1					a3	1				
					1				a3		1			
						1			a3			1		
							1		a4				1	
								1	a4					1

Inputs						Outputs
Present state	p1	p2	p3	p4	p5	Next state
a1	1					a2
	0	1				a3
	0	0				a4
a2		0				a1
	0	1				a2
	1	1				a3
a3	1					a1
	0	1				a2
	0	0	1			a3
	0	0	0	1		a4
	0	0	0	0	1	a4
	0	0	0	0	0	a1
a4					0	a3
				0	1	a2
			0	1	1	a1
		0	1	1	1	a3
	0	1	1	1	1	a4
	1	1	1	1	1	a1

The total area of the dense structure equals 628, i.e. the straightforward implementation is preferable for this FSM.

The proposed in the paper allows to compensate the growing of the PLA area in such cases. Observing Table I we can see that the FSM comprises two very different portions: the top portion including transitions from states a_1 and a_2 and the bottom portion including transitions form states a_3 and a_4. The top portion is extremely suitable for the dense transformation since its transitions depend of a small number of input variables and, consequently, can be effectively compressed by using the Input transformation. In contrast, the bottom portion is a dense one, and it cannot be improved by the Input transformation. Such class of FSMs is in the focus of our paper.

We study partitioning the initial FSM into two interacting components in such a way that allows handling each of components separately. The partitioning allows to apply the dense transformation just for a suitable component FSM and to implement the rest FSM straightforwardly.

3. FSM partitioning

In this section, we define component FSMs of the proposed partitioning network formally. Let an initial FSM S be defined as $S\left(A,\ X,Y,\delta,\lambda,a_1\right)$, where: $A=\left\{a_1,\ldots,a_M\right\}$ - the set of states of the FSM; $X=\left\{x_1,\ldots,x_L\right\}$ - the set of input variables of the FSM; $Y=\left\{y_1,\ldots,y_N\right\}$ - the set of output variables of the FSM; $\delta: A\times X \Rightarrow A$ - the transition function; $\lambda: A\times X \Rightarrow Y$ - the output function; a_1 - the initial state of the FSM.

Beside the initial FSM S, a two-block partition $\pi=\left\{A_1;A_2\right\}$ on the set of its states A is defined.

The proposed structure consists of two interacting component FSMs S_1 and S_2. The interacting is arranged in such a way that the network functions equivalently to the initial FSM before the partitioning. Define the first component FSMs $S_1\left(B_1,X_1,Y_1,\delta_1,\lambda_1,a_1^1\right)$ formally. It is clear that the component FSM S_2 is defined in the same way.

We use FSM described by Table I to illustrate our approach. Let $\pi=\left\{\overline{a_1,a_2};\overline{a_3,a_4}\right\}$ be the partition on the set A.

Let us define each of elements of the component FSM $\left(B_1,X_1,Y_1,\delta_1,\lambda_1,a_1^1\right)$ as follows:

1. A set of states of the FSM S_1: $B_1 = A_1 \cup b_1$, where b_1 is a single auxiliary state of S_1. The component FSM S_1 resides in state b_1 when FSM S_2 resides in states from the set A_2. In our example: $B_1 = \{a_1, a_2, b_1\}$ $B_2 = \{a_3, a_4, b_2\}$.

2. A set of input variables: $X_1 = \bigcup X(a_m) \cup E_{21}$, where: $X(a_m)$ is a set of input variables of the initial FSM sampled at transitions from $a_m \in A_1$; E_{21} is a set of auxiliary input variables connecting an input of S_1 with an output of component FSM S_2. In our example: $X_1 = \{x_1, x_2, x_3, x_4\} \cup e_1 \cup e_2$, $X_2 = \{x_5, x_6, x_7, x_8, x_9\} \cup e_3 \cup e_4$.

3. A set of output variables: $Y_1 = \bigcup Y(a_m) \cup E_{12}$, where $Y(a_m)$ – is a set of output variables of the initial FSM, generated at transitions from $a_m \in A_1$; E_{12} – is a set of auxiliary output variables arriving from the output of S_1 to the input of the S_2. In this paper, we deal just with input transformation and don't provide FSM outputs. In our example: $E_{12} = \{e_3, e_4\}$; $E_{21} = \{e_1, e_2\}$

4. The transition and the output functions of the component FSM S_1 are defined as follows:

a) In the case of transition within the one and the same component FSM:

$$\left(\delta(a_m, X_h) = a_j\right) \& \left(\lambda(a_1, X_h) = Y_t\right) \& \left(a_m, a_j \in A_1\right) \Rightarrow \left(\delta_1(a_m, X_h) = a_j\right) \& \left(\lambda_1(a_1, X_h) = Y_t\right);$$

b) In the case of transition between the two component FSMs:

$$\left(\delta(a_m, X_h) = a_j\right) \& \left(\lambda(a_m, X_h) = Y_t\right) \& \left(a_m \in A_1, a_j \in A_2\right) \Rightarrow$$

$$\Rightarrow \left(\delta_1(a_m, X_h) = b_1\right) \& \left(\lambda_1(a_m, X_h) = Y_t \cup E_{21}^j\right) \&$$

$$\& \left(\delta_2(b_2, M(E_{21}^j)) = a_j\right) \& \left(\lambda_2(b_2, M(E_{21}^j)) = Y_0\right)$$

where E_{21}^j - the set of auxiliary variables equal to 1 when a component FSM moves to the state a_j; $M(E_{21}^j)$ - a minterm corresponding to the this set. In our example:

$E_{21} = \{e_1, e_2\}$ $M(E_{21}^1) = e_1\bar{e}_2$; $M(E_{21}^2) = \bar{e}_1 e_2$;

$E_{12} = \{e_3, e_4\}$; $M(E_{12}^3) = e_3\bar{e}_4$; $M(E_{12}^4) = \bar{e}_3 e_4$.

Component FSMs S_1 and S_2 constructed according to the above definitions are presented in Tables III and IV respectively.

Table III. PLA of Component FSM S_1

	Inputs						Outputs		
Present State	x1	x2	x3	x4	e1	e2	Next state	e3	e4
a1	1						a2	0	0
	0	1					b1	1	0
	0	0					b1	0	1
a2				0			a1	0	0
			0	1			a2	0	0
			1	1			b1	1	0
b1					1	0	a1	0	0
					0	1	a2	0	0
					0	0	b1	0	0

Table IV. PLA of Component FSM S_2

			Inputs					Outputs		
Present state	c3	c4	x5	x6	x7	x8	x9	Next state	c1	c2
a3			1					a1	0	0
			0	1				a2	0	0
			0	0	1			b2	1	0
			0	0	0	1		b2	0	1
			0	0	0	0	1	e4	0	0
			0	0	0	0	0	a1	0	0
a4							0	a3	0	0
						0	1	a2	0	0
					0	1	1	a1	0	0
				0	1	1	1	b2	1	0
			0	1	1	1	1	b2	0	1
			1	1	1	1	1	a1	0	0
b2	1	0						a3	0	0
	0	1						a4	0	0
	0	0						b2	0	0

The total area of the network comprising two component FSMs is equal to 180+270=450, which is lower than PLA area required for the initial FSM.

Let us show that the dense transformation of the first FSM provides an additional improvement. The corresponding Input transformation PLA and the Core PLA are presented in Table V.

Table V. Dense PLA implementation of the component FSM S_1

				Inputs			Outputs	
x1	x2	x3	x4	e1	e2	Present state	p1	p2
1						a1	1	0
	1					a1	0	1
		1				a2	1	0
			1			a2	0	1
				1		b1	1	0
					1	b1	0	1

	Inputs		Outputs		
Present state	p1	p2	Nexr state	e3	e4
a1	1		a2	0	0
	0	1	b1	1	0
	0	0	b1	0	1
a2		0	a1	0	0
	0	1	a2	0	0
	1	1	b1	1	0
b1	1		a1	0	0
		1	a2	0	0
	0	0	b1	0	0

For this example, the dense implementation of the FSM S_1 leads to 20% total area reduction in comparison with the straightforward implementation of the initial FSM without the partitioning.

4. Experimental Results

Benchmarks results provided in [4] indicate the area overhead required for the dense transformation. We estimate the area overhead analytically by using the approximate expressions.

Obviously, there is no exact expression for the dense PLA area. We suggest to estimate the area approximately, assuming that the number of products in the Input transformation PLA is equal to the number of FSM states. In most cases this estimation is the upper bound of the number of products. Based on the above, the Input transformation PLA area can be calculated as follows:

$W_I = \left(L + 2R + P\right)R$, where: L – the number of FSM inputs, R – the number of FSM states, P – the number of output variables of the Input transformation PLA. The area of the AND plane of the Core PLA equals to $W_C = 2\left(P + R\right)H$.

Now it is easy calculate the expected area overhead. The question is: how the proposed estimation expressions reflect the real area overhead obtained by the algorithm [4]? To answer for this question we compared the benchmarks results with the area values obtained by our formulas. It is easy to check that our approximate formula gives area values that are 30%-60% more than real area of dense PLA [4]. . We use the formula to evaluate results of the proposed method of FSM partitioning.

To evaluate the method of FSM partitioning, we performed experiments with the same set of FSM benchmarks that was used in [4]. The aim of the experiments was to compare the area required for the dense PLA implementation with the area required for PLA implementation of the same FSM after the partitioning.

Obviously, the important concern in the partitioning process is choosing the optimal partition. It is clear that the success of the partitioning strongly depends of the chosen partition. In this paper, we don't provide an algorithm of choosing the optimal partition. We choose the desired partition manually for each benchmark, taking into account a number of heuristics. Due to the limited space, we don't present these heuristics in the paper. The partitions were chosen in such a way that one of the component FSMs would be desirable for the dense transformation while the second is not.

According to this strategy, one FSM was transformed and the other was implemented straightforwardly.

The benchmark results are presented in Table VI. Columns of the table are (left to right): benchmark names; areas required: W_{sf} - for the straightforward PLA implementation, W_d - for the dense structure, W_p - partitioned structure; W_{pd} - partitioned structure with dense transformation of one component; $k_d = \left(W_d / W_{sf}\right) \bullet 100\%$; $k_p = \left(W_p / W_{sf}\right) \bullet 100\%$ and $k_{pd} = \left(W_{pd} / W_{sf}\right) \bullet 100\%$.

Table VI. Benchmark results

Benchmark	Wsf	Wd	Wp	Wpd	kd%	kp%	kpd%
V1120	10640	11931	11267	6090	1.12	1.06	0.572
V110	6800	7230	6858	5464	1.06	1.01	0.804
V16	4674	4957	5750	5463	1.06	1.23	1.169
SOL	19860	18931	14376	7994	0.95	0.72	0.403
SASI	8200	5996	7782	7994	0.73	0.95	0.975
SARA	4950	5274	5067	4798	1.07	1.02	0.969
ROIZ	8970	10016	11017	8980	1.12	1.23	1.001
RAZ	7308	7020	6992	6809	0.96	0.96	0.932
RATM	10660	11070	12038	9898	1.04	1.13	0.929
RAFI	10512	12306	11452	8986	1.17	1.09	0.855
PP	5250	5082	5112	5098	0.97	0.97	0.971
OSHR	10200	10522	10675	9987	1.03	1.05	0.979
MOGI	13650	14412	12374	12019	1.06	0.91	0.881
MICKS	8632	8431	9865	7985	0.98	1.14	0.925
MD	18328	17684	17554	17548	0.96	0.96	0.957
LIOR	13108	12023	13245	13080	0.92	1.01	0.998
LIFT2	1728	1716	1887	1578	0.99	1.09	0.913
LCU	4000	5039	4456	4467	1.26	1.11	1.117
KOBZ	8832	9993	10562	7568	1.13	1.2	0.857

It is clear from the table that, for the majority of the benchmarks, the proposed partitioning allows reduction of the required area in comparison with the straightforward implementation. Moreover, the dense transformation of one of the component FSMs, leads to additional area reduction in most cases. Recall that all the results were obtained by the approximate formula that gives about 50% overhead, which means that real improvement, would be meaningfully larger.

5. Conclusions

We presented an approach for synthesis FSMs by Nano-electronic PLAs. The main concern in such synthesis is the fault-tolerance that can be achieved by introducing significant hardware overhead. The most promising approaches to synthesis NanoPLAs are based on the tautology, which is doubling of PLA rows and columns. On this way, the bottleneck is a huge area overhead. Additionally, due to the devices missing domination, the problem of increasing the number of devices becomes especially important.

In our paper, we presented a solution of the above problems. A newly introduced approach allows minimizing the area overhead by combining two manipulations with FSMs: partitioning and dense transformation.

We introduced the FSM partitioning procedure formally and preformed the partitioning for a set of FSM benchmarks. In each FSM, we carried out the dense transformation of one of the components. Results show that both the partitioning and the dense transformation lead to area reduction in majority of cases.

The main conclusions of the paper may be summarized as follows.

1. The FSM partitioning in majority of cases leads to reduction in the PLA area .
2. The separate transformation of component FSM leads to significant area reduction in comparison with non-partitioned FSM.
3. The combination of the FSM partitioning and the dense transformation is a promising way for synthesis of fault-tolerant FSMs by NanoPLAs.

References

[1] Wenjing Rao, Alex Orailoglu, Ramesh Karri, “Logic Level Fault Tolerance Approaches Targeting Nanoelectronics PLAs”, Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE '07, 2007 pp. 1 – 5.

[2] Wenjing Rao, Alex Orailoglu, Ramesh Karri, “Fault Tolerant Approaches to Nanoelectronic Programmable Logic Arrays”, 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2007. DSN '07, 216 –224.

[3] Ilia Polian, Wenjing Rao: Selective Hardening of NanoPLA Circuits. DFT 2008: 263-271.

[4] Baranov S., Levin I., Keren O., Karpovsky M. (2009) Designing Fault Tolerant FSM by Nano-PLA, 15th IEEE International On-Line Testing Symposium (IOLTS 2009, Sesimbra-Lisbon, Portugal, 229-234.