UDK621.3:(53+54+621+66), ISSN0352-9045

LOGIC PERTURBATIONS: A BASIS FOR DIGITAL CIRCUITS OPTIMIZATION

A. Žemva, B. Zajc
Faculty of Electrical Engineering, Ljubljana, Slovenia

Keywords: digital circuits, digital circuits optimization, logical perturbations, logic synthesis, test pattern generation, fault simulation, multi-level digital circuits, wave synthesis method, single-output circuits, multi-output circuits, experimental results, optimization algorithms

Abstract: In this paper, we introduce the concept of permissible logic perturbations as a method for logic optimization of multi-level digital circuits. The presented approach, denoted as a wave synthesis, refers to a sequence of procedures performed in order of logic levels that transform a perturbation region of multi-input, multi-output wires into a multi-input, multi-output logic subcircuit. Primary goal of the wave synthesis, which in contrast to the other methods for logic optimization relies on fault simulation and test pattern generation algorithms, is the area optimization of the initial, technology-independent multi-level circuit. We have verified the wave synthesis concept for several multi-input, multi-output combinational circuits and the experimental results obtained with the prototype software WASP confirmed the presented approach.

Logične perturbacije: Osnova za optimizacijo digitalnih vezij

Ključne besede: vezja digitalna, optimizacija vezij digitalnih, perturbacije logične, generiranje vzorcev testnih, simulacija napak, vezja digitalna večnivojska, metoda valovne sinteze, vezja enoizhodna, vezja večizhodna, rezultati eksperimentalni, algoritmi optimizacijski

Povzetek: V članku je predstavljena metoda za optimizacijo večnivojskih logičnih vezij na osnovi dovoljenih perturbacij. Predlagana metoda, imenovana valovna sinteza, temelji na zaporedju procedur s katerimi večvhodno, večizhodno perturbacijsko področje pretvorimo v večvhodno, večizhodno mutacijsko podvezje. Osnovni cilj valovne sinteze, ki v nasprotju z ostalimi metodami za logično optimizacijo temelji na algoritmih za generacijo testnih vzorcev ter simulacijo napak, je optimizacija vezij s stališča površine. Metodo smo preverili na množici testnih vezij in eksperimentalni rezultati so potrdili kvaliteto predlaganega pristopa.

1 Introduction

The increasing complexity of the modern VLSI circuitry is only feasible through the advanced CAD systems which as one of the important components include logic optimization tools. Automatic logic synthesis and optimization tools transform a high-level logic description into a multi-level network of realizable logic gates /1/.

The concept of logic perturbations has already been proven for optimizing multi-level logic combinational Boolean networks. It was first demonstrated in a transduction method, acronym for transformation and reduction, presented in /2/. Circuit transformations and reductions based on the permissible functions were repeatedly applied until a network of the sufficient lower cost was obtained. In this sense, the transduction method was significantly different from the known design methods. Optimization by logic perturbations has been further investigated in /3/. In particular, the replacement of a gate in a synchronous Boolean network was modeled by a perturbation of the gate functionality.

Perturbations presented in /4,5,6/ are based on redundancy addition and removal which can be efficiently computed using ATPG techniques. The heuristics for adding one redundant wire at a time and removing redundant wires caused by such perturbation was proposed in /4,5/. In /6/, the improved heuristics for identifying gates which are good candidates for a local functionality change, was described.

In this paper, logic optimization based on the concept of undetectable perturbations is discussed. The following discussion begins with the basic notations and definitions, followed by the description of the optimization method. Experimental results on the benchmark circuits confirmed the proposed method.

2 Notations and Definitions

We consider a synchronous multi-level Boolean network as shown in Figure 1; all flip-flops (FFs) are implicitly synchronized with a single clock. All combinational logic nodes of the direct acyclic graph (DAG) between primary/pseudo-primary inputs (PI/PPI) and primary/pseudo-primary outputs (PO/PPO) are assigned into two basic partitions: *a synthesized permissible mutation subcircuit $M_i(1 \leq i < j)$ and a remainder subcircuit R_j.* Both subcircuits are separated by a *target perturbation region P_j*. Initially, and as shown in Figure 1, this region consists of p directed edges or wires. We will refer to any pair of input pins of this region as (a, b), and the output pins as (a*, b*), respectively. Depending on the context, we may also refer to a wire pair (a, b) in P_j.

Perturbations in P_j are based on 2-in-2 perturbations introduced in /7/. These perturbations are special cases of k-in-p perturbations for k=2 and p=2. Examples of such perturbations within P_j are shown in Figure 2b. A pair of perturbations of type {$\underline{0}$ $\underline{1}$} is injected onto the wires sunk by output pins (a*, b*) - after decoding the signals on input pins (a=0, b=1). Similarly, a single of

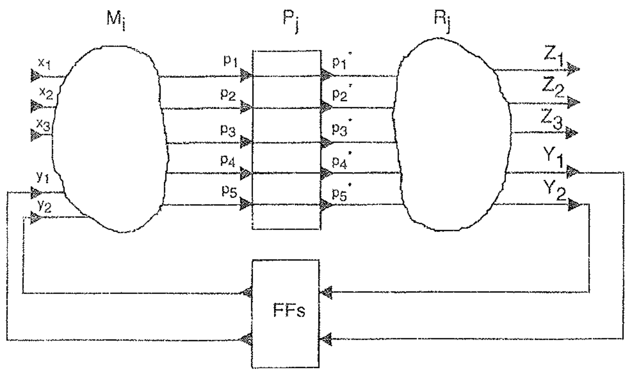


Fig. 1. Perturbation region in the Boolean network

perturbation of type $\{\underline{1}\ 0\}$ is injected onto the wire sunk by output pin (a*) - after decoding the signals on input pins (a=1, b=0). All $2^k \times (2^p-1) = 2^2 \times (2^2-1) = 12$ cases of detectability functions of 2-in-2 perturbations are thus summarized as follows:

$$\begin{aligned}
&\tau^{ab}_{\{\underline{0}\,\underline{0}\}} = \overline{f}_a\overline{f}_bO^{ab}_{\underline{0}\,\underline{0}}\ ; \quad \tau^{ab}_{\{\underline{0}\,0\}} = \overline{f}_a\overline{f}_bO^{ab}_{\underline{0}\,0}\ ; \quad \tau^{ab}_{\{0\,\underline{0}\}} = \overline{f}_a\overline{f}_bO^{ab}_{0\,\underline{0}}\\
&\tau^{ab}_{\{\underline{0}\,\underline{1}\}} = \overline{f}_af_bO^{ab}_{\underline{0}\,\underline{1}}\ ; \quad \tau^{ab}_{\{\underline{0}\,1\}} = \overline{f}_af_bO^{ab}_{\underline{0}\,1}\ ; \quad \tau^{ab}_{\{0\,\underline{1}\}} = \overline{f}_af_bO^{ab}_{0\,\underline{1}}\\
&\tau^{ab}_{\{\underline{1}\,\underline{0}\}} = f_a\overline{f}_bO^{ab}_{\underline{1}\,\underline{0}}\ ; \quad \tau^{ab}_{\{\underline{1}\,0\}} = f_a\overline{f}_bO^{ab}_{\underline{1}\,0}\ ; \quad \tau^{ab}_{\{1\,\underline{0}\}} = f_a\overline{f}_bO^{ab}_{1\,\underline{0}}\\
&\tau^{ab}_{\{\underline{1}\,\underline{1}\}} = f_af_bO^{ab}_{\underline{1}\,\underline{1}}\ ; \quad \tau^{ab}_{\{\underline{1}\,1\}} = f_af_bO^{ab}_{\underline{1}\,1}\ ; \quad \tau^{ab}_{\{1\,\underline{1}\}} = f_af_bO^{ab}_{1\,\underline{1}}
\end{aligned} \tag{1}$$

where for example, perturbation observability function $O^{ab}_{\underline{1}\,\underline{1}}$ is defined as

$$O^{ab}_{\underline{1}\,\underline{1}} = \sum_r Z_r(x, a^* = f_a, b^* = f_b) \oplus Z_r(x, a^* = \bar{f}_a, b^* = \bar{f}_b) \tag{2}$$

where Σ_r designates OR-ing the respective functions for all outputs. Given that P_j consists of m wires, a total of 12 x m x (m-1)/2 perturbations are considered. Permissible perturbation in P_j is defined for any pair of wires in P_j whose detectability, as per equations in (1), is identical to 0.

Basic properties of perturbation region P_j:

1. P_j is perturbable if there is at least one permissible perturbation in P_j;
2. P_j is non-perturbable if there is no permissible perturbation in P_j.

Permissible mutation function set $\{(f_{a^*}, f_{b^*})_i\}$ is synthesized from the pairwise permissible perturbations in P_j. We will also refer to this set as permissible pairwise mutations or simply permissible mutations. Suppose that we have the following number of permissible perturbations for a wire pair (a, b):

- n_{00} permissible perturbations for (a=0, b=0),
- n_{01} permissible perturbations for (a=0, b=1),
- n_{10} permissible perturbations for (a=1, b=0),
- n_{11} permissible perturbations for (a=1, b=1),

the size of the set of all permissible pairwise mutations is

$$|I_{ab}| = (n_{00}+1)(n_{01}+1)(n_{10}+1)(n_{11}+1) - 1 \tag{3}$$

and the set can be denoted as

$$\{(f_{a^*}, f_{b^*})_i \mid i \in I_{ab}\} \tag{4}$$

where the mutation index set I_{ab} is computed from the perturbation indices as follows:

$$I_{ab} = \{i \mid i = j4^3 + k4^2 + l4^1 + m4^0\} \forall (j,k,l,m) \in P_{ab} \tag{5}$$

and P_{ab} is a 4-tuple of permissible perturbation indices. For example, for (a=0, b=0) we have:

j=0: no perturbation is permissible for $\{0\ 0\}_{ab}$;

j=1: perturbation $\{0\ \underline{0}\}_{ab}$ is permissible;

j=2: perturbation $\{\underline{0}\ 0\}_{ab}$ is permissible;

j=3: perturbation $\{\underline{0}\ \underline{0}\}_{ab}$ is permissible;

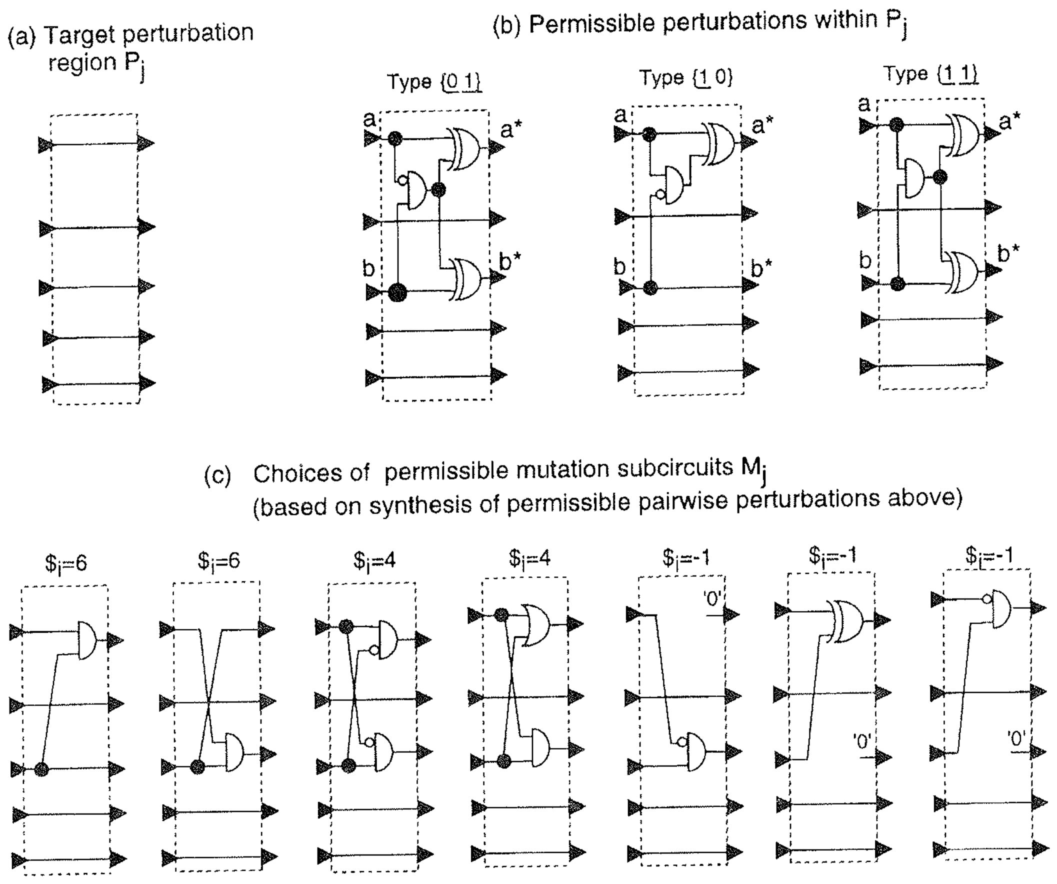
and k, l and m are defined similarly for (a=0, b=1), (a=1, b=0) and (a=1, b=1).

Permissible Mutation Subcircuit M_j is formed with the permissible mutations from $\{(f_{a^*}, f_{b^*})_i\}$ as a maximum cover at the minimum cost. A greedy heuristics for this cover is given in the next section.

Illustrative Example. We have already discussed the target perturbation region such as P_j in Figure 2a. Given that any of the three perturbations illustrated in Figure 2b are permissible (n_{00}=0, n_{01}=1, n_{10}=1 and n_{11}=1), we can synthesize up to 2·2·2-1=7 permissible mutations as shown in Figure 2c. In Figure 2d, we have tabulated permissible mutations in terms of their respective perturbation and mutation indices, along with yet to be defined mutation cost of each function, $\$_i$. For example, the first permissible mutation $f_{a^*}= f_af_b$, $f_{b^*}=f_b$ in Figure 2d is synthesized by considering perturbation $\{\underline{1}\ 0\}$ permissible (l=2). Associated perturbation index P_{ab}=0.0.2.0 and the corresponding mutation index $I_{ab}=0+0+2\cdot4^1+0=8$.

If all 12 perturbations were permissible, we would have a choice of 4·4·4·4 - 1 = 255 permissible mutations for the wire pair (a, b).

Free and Bound Wire. Wire a is referred as free if $f_{a^*}=f_a$, $f_{a^*}=\bar{f}_a$, $f_{a^*}=f_b$ or $f_{a^*}=\bar{f}_b$; otherwise it is referred as bound or covered. The same applies for wire b.


(a) Target perturbation region P_j
(b) Permissible perturbations within P_j
Type {0 1}
Type {1 0}
Type {1 1}
a
a*
b
b*
(c) Choices of permissible mutation subcircuits M_j (based on synthesis of permissible pairwise perturbations above)
$_i=6
$_i=6
$_i=4
$_i=4
$_i=-1
$_i=-1
$_i=-1
'0'

(d) Relating perturbation and mutation indices to permissible mutations and mutation costs

Pert. index	Mut. index	Permissible mutations		Mutation cost
0.0.2.0	8	$f_{a^*} = f_a f_b$	$f_{b^*} = f_b$	$\$_i$=6
0.3.2.0	56	$f_{a^*} = f_b$	$f_{b^*} = f_a f_b$	$\$_i$=6
0.0.0.3	3	$f_{a^*} = f_a \overline{f}_b$	$f_{b^*} = \overline{f}_a f_b$	$\$_i$=4
0.3.0.0	48	$f_{a^*} = f_a + f_b$	$f_{b^*} = f_a f_b$	$\$_i$=4
0.0.2.3	11	$f_{a^*} = 0$	$f_{b^*} = \overline{f}_a f_b$	$\$_i$=-1
0.3.0.3	51	$f_{a^*} = f_a \oplus f_b$	$f_{b^*} = 0$	$\$_i$=-1
0.3.2.3	59	$f_{a^*} = \overline{f}_a f_b$	$f_{b^*} = 0$	$\$_i$=-1

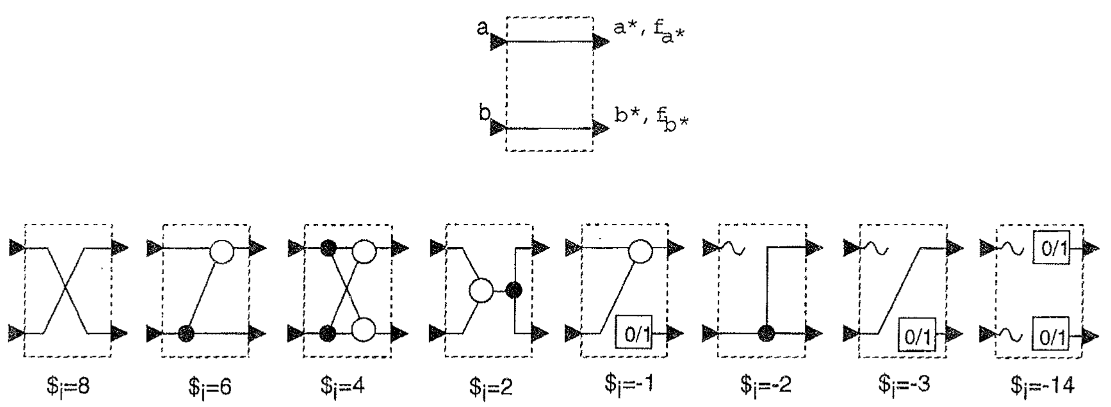
Fig. 2 Permissible perturbation and synthesized mutation functions

Cost Classes of Permissible Mutations. For each wire pair (a, b) and each permissible mutation i, we define a cost $\$_i$ as follows:

$$\$_i = \$(a) + \$(a^*) + \$(b) + \$(b^*) \quad (6)$$

where

$(a)	= -4	if the input pin is floating
	= 0	otherwise
$(a*)	= 4	if the output pin is driven by a free wire
	= 2	if the output pin is driven by a bound wire
	= 1	if the output pin is driven by a bound wire and is logically equivalent to the other output pin
	= 1	if the output pin is driven by a free wire and is logically equivalent to the other output pin
	= -3	if the output pin is driven by a constant.



Class size	Mutation cost $\$_i$	Cost			
		$\$(a)$	$\$(b)$	$\$(a^*)$	$\$(b^*)$
7	$\$_i$=8	0	0	4	4
80	$\$_i$=6	0	0	2	4
80	$\$_i$=4	0	0	2	2
20	$\$_i$=2	0	0	1	1
40	$\$_i$=-1	0	0	2	-3
8	$\$_i$=-2	-4	0	1	1
16	$\$_i$=-3	-4	0	4	-3
4	$\$_i$=-14	-4	-4	-3	-3

Fig. 3 Partitioning of 255 mutation functions into 8 cost equivalence classes

and where \$(b) and \$(b*) are calculated similarly.

For the set of permissible perturbations in Figure 2b, the costs of permissible mutations range from 6 to -1 as shown in Figure 2c-d.

The cost assignment to I/O nodes of the permissible mutations as shown in (6) induces a partition of all 255 permissible mutations into 8 cost equivalence classes (CEC). Generic topology of 2-input, 2-output permissible mutations into the 8 CECs is shown in Figure 3, along with the table that depicts the size of each class, associated with the mutation cost $\$_i$. Shaded nodes represent fanout nodes, unshaded nodes represent logic nodes that can implement all irredundant functions of 2-inputs. The square node with 0/1 represents a constant 0/1 signal that can prune logic nodes in the forward path. The wire terminated with ~ represents a floating wire that can prune logic nodes driving this wire. Wires show no inverters, although inverters may be presented without affecting the cost function. The cost of the permissible mutation in each class ranges from 8 to -14. The lower the cost of the mutation, the more pruning potential we associate with the mutation.

Since we can precompute and store all possible 2^{12}-1 perturbation indices that map into the 255 mutation indices ranked by its cost, we can determine the minimum cost mutation by simple table look-up, given the 4-tuple of permissible perturbation indices.

3 Optimization Algorithm

Consider a sample circuit shown in Figure 4, consisting of 20 2-input gates on 6 levels with a maximum fanout of 4, generated with SIS /8/ for a given minimal two-level specification. Starting at the primary inputs, we introduce a perturbation region P_1 as a set of directed wires that connect primary inputs to the inputs of the original circuit, designated as a remainder R_1, as shown in Figure 4.

Given the perturbation region P_1, we assign to each wire pair a set of 2-in-2 perturbations introduced in /7/, and determine which, if any, of these perturbations are permissible. Permissible mutation subcircuit M_j is formed by inserting permissible pairwise mutations from $\{(f_{a^*}, f_{b^*})_i\}$ for each pair.

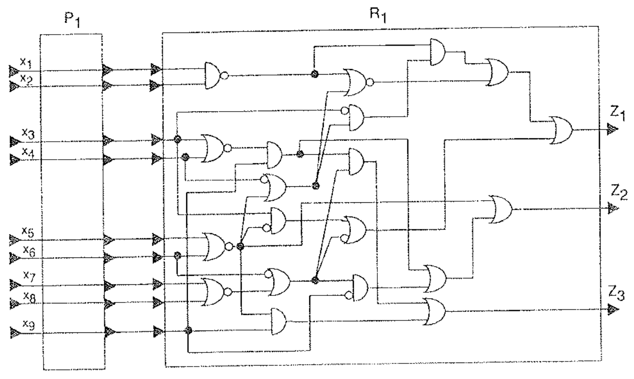


Fig. 4 Sample multi-output circuit

Cost of permissible mutation subcircuit M_j. Cost of permissible mutation subcircuit M_j, denoted as $\$_{M_j}$, is defined as:

$$\$_{M_j} = \sum_{i=1}^{m}(\$(a),\$(a^*)) \tag{7}$$

where \$(a) and \$(a*) are the cost of input and output pins as defined in the previous section.

Heuristics for M_j **synthesis.** We are formulating the problem of M_j synthesis as the problem of a maximal covering at the minimum cost. Assigning initially a tuple of (0,4) to all wires in M_j, the synthesis problem is related to the problem of finding a minimum cost $\$_{M_j}$ for a given set of permissible mutations $\{(f_{a^*}, f_{b^*})_i\}$.

Given a set of $|I_{ab}|$ permissible mutations per pair (a,b), the upper bound of the syntheses on m-output M_j is:

$$|M_j| = |I_{ab}|^{m(m-1)/2} \tag{8}$$

Greedy heuristics is used for the synthesis of permissible mutation subcircuit M_j. Permissible mutations, starting with the lowest cost, are assigned one after another to the free wires until all wires are bound or there are no more permissible mutations. The approach is illustrated in Figure 5 for M_1 of the sample circuit in Figure 4. For M_1, $f_{x1^*}=f_{x1}f_{x2}$, $f_{x2^*}=0$ and $f_{x7^*}=0$, $f_{x8^*}=f_{x7}+f_{x8}$ are selected first from the set of 36 pairs. The number of free lines is reduced by 4 and $f_{x3^*}=f_{x3}$, $f_{x4^*}=f_{x3}+f_{x4}$ and $f_{x5^*}=f_{x5}+f_{x6}$, $f_{x6^*}=f_{x6}$ are selected from the set of the remaining 10 pairs. The number of free lines is reduced to 1 (x_9) and the procedure is terminated. Complexity of the greedy heuristics used is $O(m^2)$, where m is the number of wires in P_j.

Using the proposed greedy approach, one of the cost equivalent mutation subcircuit is constructed. In order to construct M_j with different $\{(f_{a^*}, f_{b^*})_i\}$, a backtracking capability is embedded into the algorithm for M_j synthesis.

As shown in Figure 5, new perturbation region P_2 consists of directed wires driven by the synthesized permissible mutation subcircuit M_1. Remainder R_2 is the remaining circuit driven by the output wires of P_2. Applying the same procedure on P_2, mutation subcircuit M_2 is synthesized as shown in Figure 6.

The steps illustrated in Figures 5 and 6, denoted as the wave synthesis, are repeated until either the size of the perturbation region has been reduced to the two wires and the remainder itself becomes the last permissible mutation subcircuit or the perturbation region is found unperturbable.

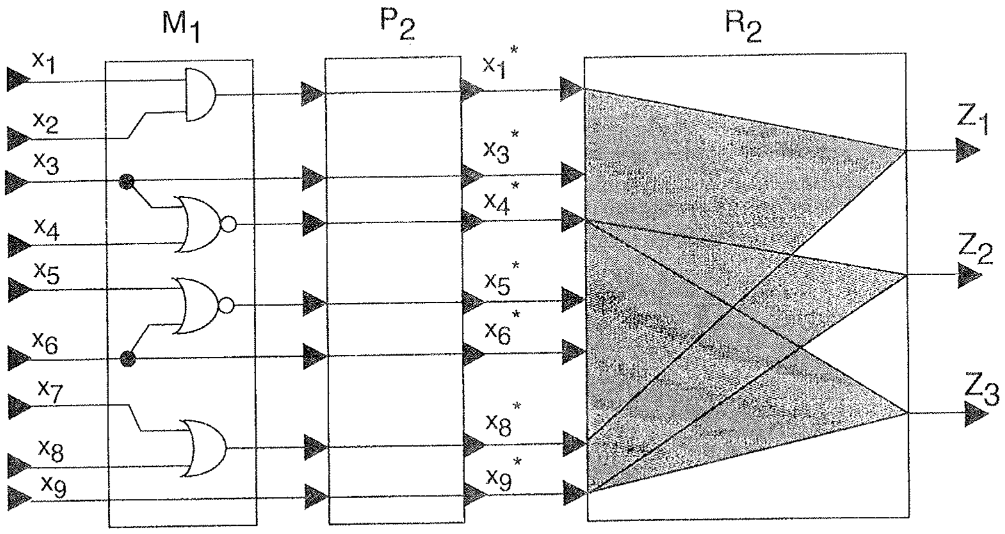


Fig. 5 Synthesis of M_1 for the sample circuit

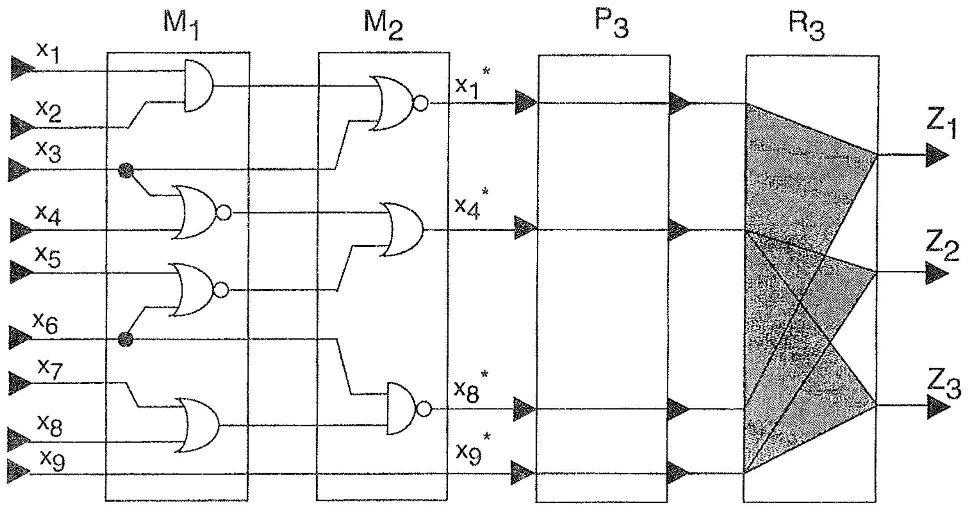


Fig. 6 Case of unpertutbable perturbation region in multi-output circuit

Case of Unperturbable Perturbation Region.

The problem arises when we reach the perturbation region such as P_3 in Figure 6, which is unperturbable. In this case, we have to consider a partitioning strategy for the remainder such as R_3 in Figure 6. Remainder R_j is partitioned into two parts: one with a single output and the other with the remaining outputs. Our decision to select the 'best single output candidate partition' is based on the notion of the extraction potential. Our definition of the extraction potential expands the notation of replication potential introduced in /9/.

Let $A_{Z1}=[1\ 1\ 1\ 0]^T$, $A_{Z2}=A_{Z3}=[0\ 1\ 1\ 1]^T$ denote the adjacency vectors of the remainder R_3. The purpose of the adjacency vector is to show the dependency of the primary outputs of the circuit to the wires in the perturbation region. A value of 1 for the k-th wire in the adjacency vector denotes that there is at least one path from the k-th wire to the PO Z_i, while a value of 0 denotes that there is no path from the k-th wire to the PO Z_i.

The extraction potential ψ_{Zi} for each PO Z_i of the remainder is then

$$\psi_{Z_i} \hat{=} \left\| \left(A_{Z_i} \oplus \prod_{j=1;j\neq i}^{m} A_{Z_j} \right) \right\| \qquad (9)$$

All operations performed in (9) are defined as follows:

- Logical XOR. For example, given $A_{Z1}=[1\ 1\ 1\ 0]^T$ and $A_{Z2}=[0\ 1\ 1\ 1]^T$, $A_{Z1} \oplus A_{Z2} = [1\ 0\ 0\ 1]^T$.
- Logical AND. For example, given $A_{Z1}=[1\ 1\ 1\ 0]^T$ and $A_{Z2}=[0\ 1\ 1\ 1]^T$, $\Pi^2_{j=1} A_{Zj}=A_{Z1} \cdot A_{Z2} = [0\ 1\ 1\ 0]^T$.
- Norm. For example, given $A_{Z1}=[1\ 1\ 1\ 0]^T$, $\|A_{Z1}\| = 3$.

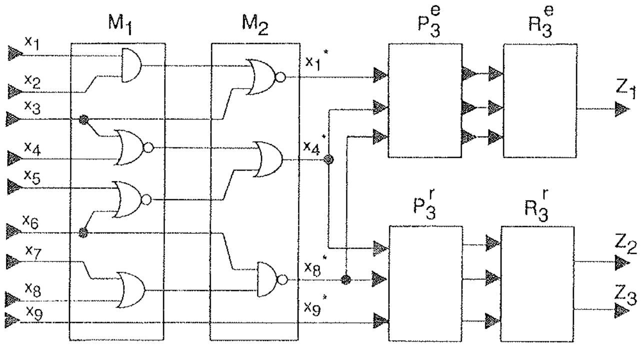


Fig. 7 Partitioning perturbation regions

Single-output partition with the highest extraction potential ψ_{Zi} is selected. The higher is the extraction potential for Z_i, the lower is the number of common wires of P_j driving Z_i and the remaining POs. Similarly, $\psi_{Zi}=0$ denotes that all wires of P_j drive Z_i and the remaining POs. For the adjacency vectors of the remainder R_3 in Figure 6, $A_{Z1}=[1\ 1\ 1\ 0]^T$, $A_{Z2}=A_{Z3}=[0\ 1\ 1\ 1]^T$, we find $\psi_{Z1}= \|\ ([1\ 1\ 1\ 0]^T \oplus [0\ 1\ 1\ 1]^T)\ \| = 2$ and $\psi_{Z2} = \psi_{Z3} = \|\ [0\ 1\ 1\ 1]^T \oplus [0\ 1\ 1\ 0]^T)\ \|=1$. Primary output Z_1 is extracted and the remainder R_3 is partitioned into $R_3^e=\{Z_1\}$ and $R_3^r=\{Z_2,Z_3\}$ as illustrated in Figure 7. Perturbation region P_3^e consists of wires $\{x_1^*,x_4^*,x_8^*\}$ and perturbation region P_3^r of wires $\{x_4^*,x_8^*,x_9^*\}$. Applying the same concept on remainder R_3^r, the final circuit is shown in Figure 8. It is composed of 13 2-input gates on 4 levels with a maximum fanout of 3.

By inserting M_j, redundancy may be introduced in the previously synthesized permissible mutation subcircuits M_i ($1 < i < j$) as well as in the remainder R_j. We postpone redundancy removal until the optimization procedure terminates.

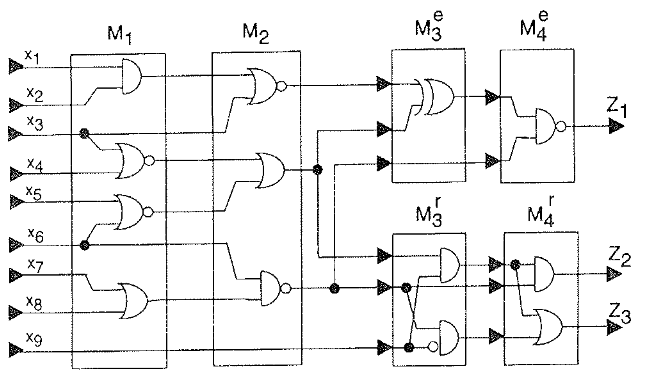


Fig. 8 Final synthesized multi-output circuit

4 Experimental Results

We have implemented the wave synthesis algorithm as a program WASP in Programming Language C and run it on Sun SPARC 10 workstation. In this section, we report our experimental results obtained with the circuits from the Benchmark set from /10/.

In Table 1, we compare WASP results to the results obtained with SIS, using the commands from script algebraic. The initial benchmark circuits exist in the multi-level form or in the minimum two-level representation (circuits market with p). Since the final, technology-independent circuits generated by WASP consist of 2-input nodes, we used command xl_split-n2 for the SIS generated circuits to split any logic nodes with more than 2 inputs into 2-input gates. For all circuits, we then report the number of 2-input nodes, logic levels and the maximum fanout. The results are summarized in Table 1. We have improved the number of 2-input nodes for 15.64%, number of logic levels for 5.08% and the maximum fanouts for 23.01%. This confirms that the presented algorithm for logic optimization optimizes the area and the maximum fanout by in general any additional increase of the delay.

5 Conclusions

In this paper, we have introduced the concept of wave synthesis for optimization of digital circuits. The premise of the proposed concept is exploiting permissible perturbations evaluated for wire pairs within the perturbation regions. Given a set of permissible functions associated to any pair with permissible perturbations, permissible mutation subcircuit of the lowest cost is synthesized and inserted into the nominal circuit. The

Table 1: Comparison of technology-independent circuit

Circuit Name	Inp. Nmb	Out. Nmb	SIS			WASP		
			Gates	Levels	Fanout	Gates	Levels	Fanout
cm82a	5	3	13	5	3	12	5	3
rd53^p	5	3	34	6	8	22	7	4
cm138a	6	8	20	4	8	20	4	8
rd73^p	7	3	59	10	8	44	9	4
z4ml	7	4	18	9	3	18	8	3
inc^p	7	9	100	9	19	94	10	13
5xp1^p	7	10	94	10	15	75	9	12
rd84^p	8	4	61	14	6	49	10	3
misex1^p	8	7	49	7	7	50	7	8
clip^p	9	5	99	10	14	88	11	13
sao2^p	10	4	128	12	17	108	13	14
x2	10	7	42	7	9	37	7	8
cm85a	11	3	36	6	5	26	8	3
t481	16	1	27	9	4	15	4	1
Total			780	118	126	658	112	97
Improvement [%]			Gates: 15.64		Levels: 5.08		Fanout: 23.01	

proposed approach is feasible for single and multi-output circuits. Experimental results on the benchmark circuits demonstrated the proposed method by optimizing circuits in terms of area and wiring.

References

/1/ G. De Micheli. Synthesis and Optimization of Digital Circuits. McGraw Hill, New York, 1994.

/2/ S. Muroga, Y. Kambayashi, H. C. Lai, and J. N. Culliney. The Transduction Method - Design of Logic Networks Based on Permissible Functions. IEEE Transaction on Computer Aided Design, 38(10):1404-1424, October 1989.

/3/ M. Damiani and G. De Micheli. Don't Care Set Specifications in Combinational and Synchronous Logic Circuits. IEEE Trans. on Computer-Aided Design, CAD-12(3):365-388, March 1993.

/4/ K.-T. Cheng and L. A. Entrena. Multi-Level Logic Optimization By Redundancy Addition and Removal. In European Conference on Design Automation, pages 373-377, 1993.

/5/ L. A. Entrena and K.-T. Cheng. Sequential Logic Optimization By Redundancy Addition and Removal. In International Conference on Computer Aided Design, pages 271-276, 1993.

/6/ S.-C. Chang and M. Marek-Sadowska. Perturb and Simplify: Multi-level Boolean Network Optimizer. In International Conference on Computer Aided Design, pages 2-5, 1994.

/7/ A. Žemva and F. Brglez. Detectable Perturbations: A Paradigm for Technology Specific Multi-Fault Test Generation. In VLSI Test Symposium, pages 350-357, April 1995.

/8/ SIS -Release 1.2. UC Berkeley Soft. Distr., July 1994.

/9/ R. Kužnar, F. Brglez, and B. Zajc. Multi-way Netlist Partitioning into Heterogeneous FPGAs and Minimization of Total Device Cost and Interconnect. In ACM/IEEE 31st DAC, pages 238-243, June 1994.

/10/ Logic Synthesis Benchmarks, 1994 Available under Benchmarks at http://www.cbl.ncsu.edu/www/. For autoreply about benchmarks, send e-mail to benchmarks@cbl.ncsu.edu.

Dr. Andrej Žemva, dipl.ing.
Prof. Dr. Baldomir Zajc, dipl.ing.
Faculty of Electrical Engineering
Tržaška 25, 1000 Ljubljana, Slovenia
Tel.: +386 61 176 83 46
Fax: +386 61 126 46 30
E-mail: andrej.zemva@fe.uni-lj.si

Prispelo (Arrived): 24.4.1997 *Sprejeto (Accepted): 6.5.1997*