2006 IEEE 24th Convention of Electrical and Electronics Engineers in Israel

Designing Self-checking Circuits with Smooth Power Dissipation

Benjamin Abramov, Vladimir Ostrovsky, Ilya Levin

Tel Aviv University
Ramat Aviv, Tel Aviv 69978, Israel

Abstract-The paper discuses a new approach for designing self-checking sequential circuits with smooth power dissipation The proposed approach enables achievement of circuits with a lower overhead. At the same time it provides for circuits a specific property to have approximately the same power dissipation on all codewords.

A new architecture of the self-checking sequential circuit with smooth power dissipation is proposed. The architecture is investigated on a number of standard benchmarks.

Index Terms - unidirectional error, Finite State Machines, self-checking, switching activity, power stabilization.

I. INTRODUCTION

The progress in microelectronic industry leads to increasing of the complexity of VLSI components. The count of transistors is already in millions and in some cases has risen even higher. The tendencies and necessities for microminiaturization have led to the appearance of new families of compact complex devices that are fed by a small, autonomic power supply sources [1]. Shrinking of device size and reduction of power supply levels as well as the increase in operating speed results in reduction of noise margins. This makes the circuits increasingly sensitive to transient faults caused by Single Event Upsets like atmospheric neutrons and alpha particles [2]. The failures phenomena and the necessity in higher reliability of complex digital systems together with power supply stabilization are of special interest. This should take special consideration when we take into account the influence of the power being supplied on the circuit functionality [3]. All methods of error detection results the following: introducing a redundant hardware, increasing power consumption, and possibly growing a spread of power dissipation. The rise in the spread of power dissipation might induce more noise and as a result increase the probability of errors occurrence (especially transient errors in systems with a weak and unstable power supply source). The spring in the energy consumption of one device on a circuit might cause to a failure of others [4]. In this paper, we present a new approach for designing self-checking digital circuit with smooth power dissipation. This approach is mainly targeting complex autonomic controllers. It is based on introducing specific redundant check bits for error detecting that takes into account stabilization of the power consumption. For this purpose we used the context-oriented m-out-of-n code (CO m-n code) that provides the same number of '1' bits into output codewords [5]. An addition to that CO m-n code minimizes the total hardware overhead of a checker in comparison with known solutions.

The paper is organized as follows: Section 2 describes the fault model, the structure of circuits to be checked and gives necessary fundamentals of power dissipation; So-called context-oriented m-out-of-n code is presented in Section 3; Section 4 presents the architecture of self-checking controller with smooth power dissipation; Section 5 comprises experimental results of the proposed approach in terms of the required overhead and dynamic power dissipation for Finite State Machines (FSMs) with the checker.

II. BASIC NOTIONS

We deal with a problem of error detecting and stabilization of power consumption for controllers defined as Mealy type FSM, which has

$X = \{x_1, \ldots, x_n\}$ - the set of inputs signals,

$Y = \{y_1, \ldots, y_\ell\}$ - the set of output signals,

$A = \{a_1, \ldots, a_k\}$ - the set of output codewords.

Our approach is strictly oriented to controllers in which the number k of possible output vectors is much smaller than 2^ℓ ($k << 2^\ell$) and the set of possible vectors is known in advance. The controller has inverters only on primary inputs, or it may use input signals presented in dual-rail code as shown in Figure 1.

Let us assume that faults are manifested by poles signals of '0' or '1' on input or output contacts of logical elements of the circuit to be checked. The faults can be temporary or permanent. It is traditionally accepted that a time interval between occurrences of two adjacent faults is sufficient for coping with the earliest fault. Therefore, only a single fault can present simultaneously in a circuit under checking. This fact is usually considered when building models of acceptable distortions of output codewords. We used the unidirectional fault model, meaning that a single fault in the controlled device leads to a unidirectional error in the checked vector, i.e. to the corruption of either '0's to '1's only, or '1's to '0's only, but not booth at the same time [6]. This model is based on assumption that the system of Boolean functions, which reflects conversion of information in a scheme under check, is monotonous. In our case proposed scheme (Figure 1) does not include invertors and satisfy the mentioned assumption [2].

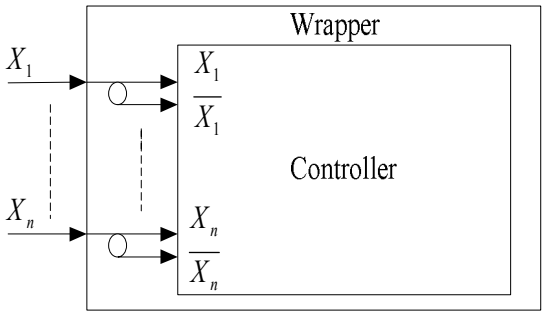

Fig1. Inverters free controller scheme.

Unidirectional errors have been observed as the result of power failures [4], the failures in memory systems [7], in regular VLSI structures such as RAM memory systems and FPGA [8], and PLA, PLD and CPLD [9].The last category has special importance in view of its prevalence as the element base of the contemporary digital systems.

The major work assumption is that we are dealing with the family of control circuits are which most of the output codewords are defined and always smaller then 2^{ℓ} (ℓ is the number of output lines from controller), in addition we implement the combinatorial part function, due to these facts the CO m-n code for output codewords monitoring is the most logical choice. Our interest is error detecting, minimization of hardware overhead and stabilization of power consumption by smoothing of dynamic power dissipation. The CO m-n code will be defined in details in Section 3.

Dynamic power dissipation in CMOS circuits is composed of power consumed in sequential logic and combinational logic. Power dissipated in the combinational logic mainly depends on the complexity of the Boolean logic functions and their gate level implementation. Power dissipation in sequential logic occurs due to capacitance charging and discharging in state registers caused by the state bits switching, which is often described as:

$P = \frac{1}{2} V_{dd}^2 f \sum_i C(i)E(i)$ where V_{dd} is supply voltage, f is clock frequency, C(i) is the capacitance of the register storing the i_{th} state bit, and E(i) is the expected switching activity of the register. C(i) is technology dependent and remains, in general, constant for all the i_{th} state bits. Dynamic power consumption is the major source of power dissipation. Also, dynamic power consumption in a dynamic circuit is solely determined by the signal probabilities of circuit lines. We target the stabilisation of power dissipation in FSM circuits by minimizing the switching activity in the state and output registers.

For state encoding we use the One-hot encoding. The number of bits in the code is equal to the number of states. Each encoded state has just 1 bit in the encoded word set to a '1' (the rest are '0'). During a state transition exactly two flip-flops change state, which often translated into low power consumption. It was proved that for FSM with one-hot code state encoding and monotonous combinatorial logic there is no need to check the state register [10]. In order to identify any error that may occur during FSM execution it will be enough to verify the output codeword. Further code example describes the output codeword, which will be constructed from subgroups, each of which holds one-hot code. It is clear that within the change of codeword, maximum switches in subgroup will be two. The next step to minimize dynamic power dissipation is to implement registers update through '0' values for each bit of the register. As there will be shown further proposed by us architecture of

the checker is optimal from the hardware overhead point of view, which also makes its contribution to the minimization of power consumption (power dissipation in the combinational logic).

III. CO M-OUT-OF-N CODE

Major difficulties in designing of self-checking devices are related to the complexity of encoding of check bits and their decoding (i.e. verification that a given output is a codeword). An unordered code for output vectors is used since for these codes no unidirectional error can transform one codeword into another codeword. The main objective in construction of unordered codes (Berger [11], Smith [12]) is to achieve the required properties by using a minimal number of redundant bits. Nevertheless, decrease of the number of bits in many cases does not lead to an expected reduction in the checker's complexity. Thus, an approach that takes into an account checker's complexity at the earlier stages of the code construction becomes very impotent. Moreover, the basis of implementation has to be taken into consideration. Based on these requirements we deal with CO m-n code. The CO m-n *code* differs from the standard code by the following two features:

- This code is a systematic one
- This code enables dividing the codeword into m fields in such a way that any acceptable codeword has exactly one bit that is equal to one in each of the fields.

The following compatibility relations between pairs of the variables $y_1, \ldots, y_\ell$ can be defined:

Definition A: *Two variables y_i and y_j are compatible if there is a codeword $a \in A$ in which both y_i and y_j are nonzero. Otherwise, they are incompatible.*

Definition B: *A subset $V_i \subseteq Y$ is a subset of (in-) compatible variables if any two variables in that subset are (in-) compatible.*

Clearly, if a particular V_i is a subset of incompatible variables, then each vector $a \in A$ may contain at most one variable from this subset (one or none at all). Based on the above definitions, we design a new code, which allows the detection of any unidirectional error in the circuit.

Definition C: Let V be partitioned into disjoint subsets of incompatible binary variables:
$V = \{V_1, \ldots, V_m\}$; $V_1 \cup \ldots \cup V_m = A$;
$V_i \cap V_j = \varnothing, \forall i, j \in \{1, 2, \ldots, m\}, i \neq j$.

In the code, we assign each subset V_i with one code bit c_i, the value of c_i is '1' in the vectors which contain no variables from V_i, and in these vector only. Clearly, each thus encoded output word will have exactly *m* '1' bits, equal to the number of subsets V_i. This is why we call this code the ***context oriented m-out-of-n code***. Let's illustrate the above with an example. Table I shows the code for the MCNC benchmark, ***tav***. The original output bits are in columns $y_1, \ldots, y_4$, the check bits are in columns $c_1, \ldots, c_3$ (for CO m-*n* code).

TABLE I. CO M-N CODE (TAV BENCHMARK)

i	original codeword a_i				codeword with check bits		
	y_1	y_2	y_3	y_4	$y_1 c_3$;	$y_2 y_3 c_1$;	$y_4 c_2$
0	0	0	0	0	01	001	01
1	0	0	0	1	01	001	10
2	0	0	1	0	01	010	01
3	0	0	1	1	01	010	10
4	0	1	0	0	01	100	01
5	0	1	0	1	01	100	10
6	1	0	0	0	10	001	01
7	1	0	0	1	10	001	10
8	1	0	1	0	10	010	01
9	1	0	1	1	10	010	10
10	1	1	0	0	10	100	01
11	1	1	0	1	10	100	10

The partition upon which the encoding is based is

$$\{y_1, y_2, y_3, y_4\} = \underbrace{\{y_1\}}_{V_1} \cup \underbrace{\{y_2, y_3\}}_{V_2} \cup \underbrace{\{y_4\}}_{V_3}.$$

We call this the coding partition.

After the partitioning, the encoding is simple: for all *i=1, 2, 3*, c_i *='1'* for a codeword that has no '1' bits from the subset *Vi*, and c_i *='0'* otherwise. This way, in each codeword there is a single '1' bit from each of the sets $V_i \cup c_i$. For automation of the partition routine we used developed by us software [13].

IV. THE SELF-CHECKING CONTROLLER

We deal with microcontrollers described by Mealy type FSM, which stores the output codeword into the register. This codeword contains check bits (Fig 2.).

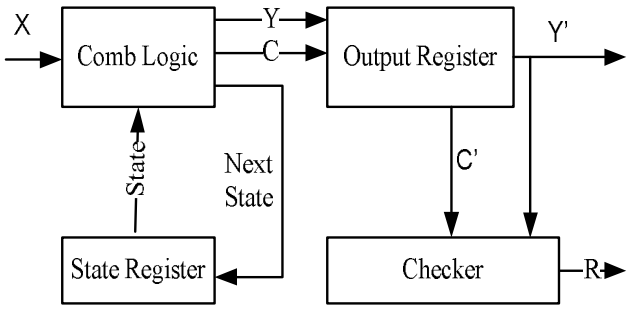

Fig2. Registered Mealy FSM with checker

One way to reduce glitches is to shorten the depth of combinational logic by adding pipeline registers. The output register divides the combinatorial part of the FSM for two logic portions. The first portion includes the next state logic, the check bits logic and the output logic. The second logic portion is the checker. The output (pipeline) register prevents glitches dissemination into checker logic during the first logic portion updates. It is translated into low power consumption [3]. The checker has two layers [13]. The first one consists of *dual-rail-output one-hot analysers* for each of the 1-out-of-*k* codes. The one-hot analyser for 4 arguments y_1, y_2, y_3, y_4 implements the following functions:

$$r_0 = y_1 + y_2 + y_3 y_4 \;\; ; \; r_1 = y_3 + y_4 + y_1 y_2 \, .$$

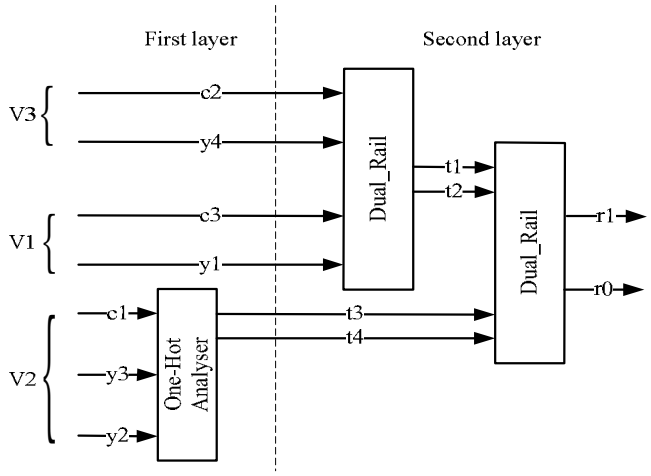

Fig3. A CO m-n code checker (tav benchmark).

The second layer consists of *4-inputs dual-rail checker* [6] for the *m* pairs resulting from the first. The final dual-rail checker implements the following functions:

$$r_0 = t_1 t_3 + t_2 t_4 \;\; ; \; r_1 = t_1 t_2 + t_3 t_4 \, .$$

For example Figure 3 shows the checker for standard MCNC benchmark ***tav***. For this benchmark, the one-hot analyser is very simple. It's functions are the follows:

$$r_0 = c_1 \; ; \; r_1 = y_2 \oplus y_3 \, .$$

As a result we have the three dual-rail pairs for second layer.

V. BENCHMARKS RESULTS

In order to examine the proposed approach for design of self-checking FSM with smooth power dissipation a number of experiments on standard MCNC benchmarks were provided. In order to see the efficiency of CO m-n code checker the results were compared with Berger code checker. All the example circuits were minimized by our software [13] before synthesis with Leonardo Precision tool. Xilinx Spartan-3 FPGA was used as a basis for the implementation of FSM, and used Xilinx XPower tool to calculate the average power consumptions of the circuit. In the power simulation the operating frequency, the supply voltage, and the environment temperature were set to 100 MHz, 1.8 V, 20°C. The results of the experiments are presented in Table II. The first four columns indicate the name of the benchmark, the number of information bits in the output word (O), number of check bits for m-n code and Berger code, C and B correspondingly. The next three columns show the number of Logic Elements (LE) for FSM without/within checker implementation, and the improvement in hardware overhead (in percents) in comparison with other known code for each of the codes: CO m-n code (m-n) and Berger (B). The last three columns show the average dynamic power dissipation for FSM without/within checker. It has to be mentioned that the benchmark results confirm our initial suggestions. Advantages of CO m-n code checkers are sufficiently transparent in most of benchmark results. Hardware overhead of CO m-n code checkers is much less, as well as dynamic power dissipation. Even through for some cases CO m-n code checkers causes bigger hardware overheads, the dynamic power dissipation is not bigger then for Berger code checker.

TABLE II. EXPERIMENTS RESULTS

benchmark	O	C	B	Area(LE/percent)			Dynamic Power (mW)		
				fsm	m-n	b	fsm	m-n	b
bbtas	2	2	2	9	17/189	15/166	1.41	1.92	1.92
ex6	7	6	3	95	162/170	155/163	7.22	11.61	12.95
bbsse	7	3	3	37	66/178	80/216	4.96	6.45	7.87
beecount	4	2	3	27	31/115	44/163	3.21	3.54	3.909
dk512	3	1	2	18	22/122	22/122	2.9	3.15	3.37
tav	4	3	3	9	17/189	27/300	1.51	3.47	5.82
s510	7	4	3	72	90/125	99/132	12.89	15.00	17.05
pma	8	7	4	122	171/140	175/143	13.21	15.33	18.01
dk14	5	3	3	48	73/152	84/175	5.17	7.33	8.01
sse	7	2	3	42	69/164	85/200.3	5.09	7.29	8.34
total	***52***	***31***	***30***	***479***	***718/150***	***786/164***	***57.57***	***75.44/131***	***86.8/150***

VI. CONCLUSION

The present paper proposes universal architecture for FSM based control circuits, which allows stabilizing power consumptions by using a specific unidirectional error detection code. The proposed architecture provides:

- The self checking property of the circuit.
- The smooth power dissipation.

Benchmark analysis shows that the proposed solution is more effective in comparison with widely known Berger code. It reduces the hardware overheat for about 15%; the dynamic power dissipation for about 20%. In our opinion, the presented FSM architecture may be used in autonomic control systems.

REFERENCES:

[1] M. Alidina et al. Pre-computation Based Sequential Logic Optimization for Low Power. In Proc. ICCAD, pages 74.81, Nov. 1994.

[2] L. Znghel, M. Nicolaidis, I. Alzaher-Noufal. Self-Checking circuits versus realistic faults in very deep submicron VLSI Test Symposium, 2000.

[3] S.-J. Ruan et al. A Bipartition-Codec Architecture to Reduce Power in Pipelined Circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems , 20(2):343.349, Feb 2001.

[4] D. K. Pradhan and J. J. Stiffler. Error-correcting codes and self-checking circuits. Computer, vol 13, pp. 27-37 , Mar. 1980.

[5] Levin I., Ostrovsky V., Ostanin S., Karpovsky M. Self-checking Sequential Circuits with Self-healing Ability. Proceedings of the 12-th ACM Great Lake Symposium on VLSI, New York, April 2002.

[6] P. K. Lala. Self-Checking and Fault-Tolerant Digital Design. Academic Press, San Diego, CA, USA, 2001.

[7] W. K. Fuchs, Ch.-Y . Chien, and J. A. Abraham. Concurrent error detection in highly structured logic arrays. IEEE J. Solid-State Circuits, vol SC-22, pp 583-594 Aug. 1987.

[8] M. Goto. Rates of unidirectional 2-column error detectable by arithmetic codes. Kyoto, Japan 1980.

[9] G. P. Mak, J. A. Abraham, and E. S. Davidson. The design of PLA's with concurrent error detection. Santa Monica, CA, June 1982

[10] V. Ostrovsky, I. Levin, S. Ostanin. "Synthesis of Self-checking Controllers Based on Modified (m, n)-code", Automatic Control and Computer Science, 2, pp. 48-55, 2003.

[11] J. Berger. A note on error detection codes for asymmetric channels. Inform. Control, 4:68–73, March 1961.

[12] J. E. Smith. On separable unordered codes. IEEE Transactions on Computers, C-33:741–743, August 1984.

[13] Perelman S, Ostrovsky V, Levin I. (2004) Design Automation of Reliable Checkers for Control Units. Proceedings of the 23-th IEEE Convention Israel.