

Error Detection in Fault Secure Controllers using State Encoding

Eckart Voskamp¹

Robert Bosch GmbH
Tuebinger Str. 123
72762 Reutlingen, Germany
E-mail: evoskamp@dic.k8.rt.bosch.de

Wolfgang Rosenstiel

University of Tuebingen
Sand 13
72076 Tuebingen, Germany
rosenstiel@informatik.uni-tuebingen.de

Abstract

In applications where safety is of utmost importance, defective functions must frequently be detected very early in order to restore the system to a safe operating condition. At the same time the availability may not be restricted by measures implemented to detect errors. The efficiency of error detection state encodings for a control unit is investigated based on the example of the Motorola micro-processor core M68HC11. The terms "dependent encoding" and "independent encoding" are introduced. The designed circuits are compared regarding to error detection of stuck-at errors and chip area. The degree of error detection is shown with the help of a fault simulation.

1 Introduction

The dramatically increasing use of microelectronic systems for the regulation and control of applications in the area of safety has resulted in a strong demand for low-cost safety concepts.

The present safety requirement coupled with a high availability are taken into consideration by using dual- or multi-channel systems. At present, different methods are being investigated in the government-subsidized Jessi-AE11 project. The methods involve the realization of operating safety levels equal to, or higher than, the present level with a greatly reduced amount of hardware.

The focus of this work is the enlargement of the central microprocessor by means of on-chip testing hardware to a fail-stop processor, that can detect an error itself and trigger the appropriate emergency program.

During fail secure realizations for the data path are known [1, 2], so far, there are no economically viable concepts for the control path.

Measures taken involving hardware to improve error detection are redundant in regard to the function of a system and, therefore, create additional costs. These result in enlarged IC-chip areas. This study aims to keep the chip area as small as possible while at the same time increasing reliability.

Four central terms are explained as they will be used in the context of this study. Any other explanations or

mathematical definitions can be found in the literature [3].

Code word refers to all the binary vectors of the length n that represent a permissible bit pattern. The set of all the code words is referred to as "Code". All the other binary vectors of the same length n are called *non-code words*. The combination of the sets of all code words and all non-code words of the same length n represents the binary vector space.

The *Hamming distance* between two binary vectors $\underline{u}_1$ and $\underline{u}_2$ of equal length is written as $d(\underline{u}_1, \underline{u}_2)$ and represents the number of different elements in the same bit position of these vectors. For the binary vectors $\underline{u}_1$ and $\underline{u}_2$ is true that $d(\underline{u}_1, \underline{u}_2) = w(\underline{u}_1 \oplus \underline{u}_2)$. The symbol $\oplus$ stands for the XOR operation and $w(\underline{x})$ stands for the weight of the vector $\underline{x}$.

A system is *fault secure* if it never produces an incorrect output due to a fault, unless it simultaneously indicates a present fault. A *fail safe* system, on the other hand, produces always the expected output function during the normal operations.

2 State of technology

Basic error detection and/or error correction methods for state assignment can be found in literature [3, 4, 5, 6]. While many current theoretical work focuses on error correction or on very high error detection [7, 8], only a few papers to find an acceptable solution for application regarding the costs [9, 2].

In [9], an algorithm is shown that can be employed for error correction of single bit errors (Single Error Correction, SEC), since a code with the Hamming distance $d = 3$ can be produced. This code is equal to a Hamming code, only the generation is considerably more efficient. In addition, the smallest possible Hamming distance is maintained between adjacent states in order to minimize the area. It is mentioned that the algorithm is not suitable for large controllers because it becomes too complex.

Other state assignment procedures are concerned with minimizing the area of FSMs (Finite State Machine). These procedures (NOVA, JEDI, MUSTANG) do not allow for any error detection or error correction. These algorithms allocate code words with the smallest possible Hamming distance to adjacent states. It can be seen that this optimization target for large designs, in which the logic is large in comparison to

¹Since 01/01/96 with Texas Instruments Germany, Mixed Signal Systems, Haggertystr. 1, 85356 Freising, Germany, E-mail: e-voskamp@msg.ti.com

the sequential circuit components, is not valid in every case.

In the relevant literature, it is typical to compare designs with about 50 states from the MCNC91 benchmarks. In this paper an error-coded FSM with 615 states of a standard- μC , the M68HC11, has been realized in a design under industrial conditions.

Error detection with the smallest possible chip area of a comparably large control unit, is the aim of this paper.

3 Fault security of FSMs using state encoding

The degree of error detection depends on the Hamming distance d of the code words of a code space. For a single error detection (SED), d must have a minimum value of $d = 2$.

In the following text the terms *dependent* and *independent* encoding will be introduced. The relevant error detection circuits (EDC) are called *dynamic* and *static*.

3.1 Dependent encoding and dynamic error detection

Two symbolic states S_1 and S_2 of a FSM are adjacent if one of these states succeeds the other.

An encoding is being sought for which a function f_d of two binary vectors $\underline{u}_1$ and $\underline{u}_2$ adjacent states for a defined error set $\mathcal{F}$ exists. The bit errors from $\mathcal{F}$ can be detected with this function. Such encoding is called *dependent* since the current code word must be determined depending on the previous code word.

The function f_d is called *dependent error function*. The hamming distance d between two code words of adjacent states is a suitable dependent error function. If d remains constant, then two successive binary vectors can be checked for their adherence to the Hamming distance.

The answers to the following questions are of considerable importance:

1. What is the minimum length n of the code word (block length as well) if the number of different, and to each other adjacent, states are encoded with a fixed Hamming distance?
2. Which connection exists between the number t of clearly detectable bit errors and the Hamming distance d in the dynamic error detection?

These questions will be answered in the sections 3.1.1 and 3.1.2.

3.1.1 Defining the number of adjacent states that can be encoded

The number q of binary vectors, which all have to have a pre-defined Hamming distance d to each other, is to be determined. Such codes are called *equidistant*. Since q depends strongly on d , the following two dominant groups are presented here separately.

Hamming distance d is odd

Two binary vectors $\underline{u}_1$ and $\underline{u}_2$ have an odd Hamming distance d . A third binary vector $\underline{u}_3$ of the same length with the same odd Hamming distance d to $\underline{u}_1$ and $\underline{u}_2$ is to be determined.

By considering all the possibilities of a $\underline{u}_3$ construction, it can be shown that a third vector with this assumptions does not exist. The consequence is that a dependent encoding with an odd Hamming distance for states that have more than one adjacent state is impossible.

A dependent encoding with code words that must have an odd Hamming distance is thus irrelevant.

Hamming distance d is even

Likewise the number q of different binary vectors which in pairs have the same even Hamming distance d can be defined by means of construction. For the Hamming distance d of two binary vectors $\underline{u}_1$ and $\underline{u}_2$, the following equation is valid:

$$d(\underline{u}_1, \underline{u}_2) = \sum_{i=1}^n (\underline{u}_{1(i)} \oplus \underline{u}_{2(i)}) = (\underline{u}_{1(1)} \oplus \underline{u}_{2(1)}) + (\underline{u}_{1(2)} \oplus \underline{u}_{2(2)}) + \dots + (\underline{u}_{1(n)} \oplus \underline{u}_{2(n)}) \quad (1)$$

The summands are grouped in such a way that first all terms with the XOR value "0" (following called Low) and then all terms with the XOR value "1" (following called HIGH) are recorded. If one writes the index i for the Low-group and the index j for the HIGH-group, then the following is valid:

$$d = \left(\underbrace{\sum_{i=1}^n (\underline{u}_{1(i)} \oplus \underline{u}_{2(i)})}_{d_0=0} + \underbrace{\sum_{j=1}^n (\underline{u}_{1(j)} \oplus \underline{u}_{2(j)})}_{d_1=d} \right) \quad \left| \begin{array}{l} i \neq j \\ i, j \leq n \end{array} \right. \quad (2)$$

Low-group HIGH-group

Vector $\underline{u}_3$ which was to be defined is for the moment identical with $\underline{u}_2$. In order to achieve an *even* hamming distance d to $\underline{u}_1$ and to $\underline{u}_2$, $\frac{d}{2}$ bits need to be inverted both in the Low-group and in the HIGH-group of equation (2).

This is valid for all subsequent vectors which, according to the requirements, should comply with a constant d in pairs to all previously defined vectors.

Vectors with a length n allow this step exactly q -times. With $q \in N$ results in

$$n = q \frac{d}{2} + R \quad \text{respectively} \quad q = \left\lfloor \frac{2n}{d} \right\rfloor, \quad q \in N \quad (3)$$

Equation (3) can be used to answer the first question in this section of the text. An odd Hamming distance is only applicable for two adjacent states.

If the Hamming distance d is even, then the greatest number q of different, paired, adjacent states can be

coded with $d = 2$ and a fixed block length n , while $n \geq q$ must be true. The number of necessary bits for the length of the code word must be at least equal to the number of adjacent states.

3.1.2 Dynamic error detection

Error detection will be called *dynamic* if bit errors of the defined error set $\mathcal{F}$ can be detected with the dependent error function f_d . The error detection takes place by connecting the binary vectors of two adjacent states.

In Figure 1 a circuit for dynamic error detection is shown.

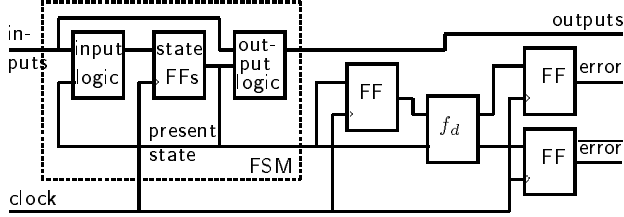


Figure 1: Circuit for dynamic error detection

It can be shown that bit errors t can only be reliably detected dynamically when the Hamming distance d between the code words of the adjacent states is equal. The basic idea behind this proof is that, based on the equation (2), the faulty bit position i in the binary vector is random. The possibilities $d \geq x$ and $d \leq x$ with x being a fixed value can be ruled out.

Other important insights concerning the number of detectable bit errors by means of dynamic error detection are summarized below. Proof for this conclusion has not been included here. The number of falsified bits is shown as t and the Hamming distance between two binary vectors to be checked as d .

1. For a number t of faulty bits within the limits $1 \leq t < 2d$ and a code word length of $n \geq 2d$, bit errors can only be reliably detected by means of dynamic error detection when t is odd.
2. The number t of faulty bits that can be reliably detected dynamically is within the limits $1 \leq t < 2d$ and a code word length of $n \geq 2d$. This is independent of the Hamming distance d .
3. If $t > 2d$ for the number t of the falsified bits of a bit error is true, then these bit errors can *always* be detected by means of dynamic error detection.

It can be concluded that the Hamming distance $d = 2$ is the most suitable for dependent encoding using *equidistant* codes. The largest number of encodable adjacent states is possible with this value. This number is greatly restricted anyway because of the code word length. With $d = 2$, all single bit errors and all other bit errors with $t > 4$ can be detected.

3.2 Independent encoding and static error detection

Independent encoding, in contrast to dependent encoding, does not require that the binary vector of the previous state be known in order to detect bit errors.

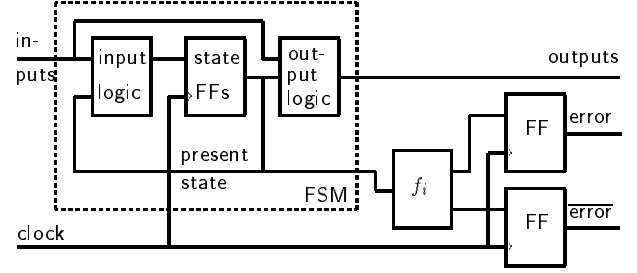


Figure 2: Circuit for static error detection

The inner properties of a code word are used for error detection. Otherwise, all of the code words in the error detection circuit need to be implemented as permissible. In Figure 2, a circuit for static error detection is shown.

3.2.1 Separable Codes

If the bit positions i of the information bits and the check bits of all code words of the code $\mathcal{S}$ are known, then $\mathcal{S}$ is separable.

Separability assumes that check bits exist. Likewise an independent error function f_i between the information bits and the check bits exists that can be checked for error detection.

In this study three standard codes were examined in order to compare their efficiency in regard to chip area and error detection.

The Parity-Code, the Hamming-Code and the Berger-Code were studied extensively in relevant literature [3, 4, 5]. Therefore, only two new results regarding error detection are presented here:

1. With the odd parity encoding, more global errors are detected than with the even parity encoding. Global errors are bit errors in which all the bits fail as a result of common mode failures to either logic-"0" or logic-"1". They should therefore be observed carefully.
2. Studies of the Hamming-Codes in regard to error detection have shown that, by negating the check bits, a greater number of multiple errors can be found than with the original Hamming-Codes. This codes were called hereafter "modified Hamming-Codes".

Theoretical results are validated by means of fault simulation and studied with regard to the chip area.

3.2.2 Non-separable Codes

Code words that cannot be connected by means of an error function among themselves, or within the respective code word, can also be used for error detection.

Such codes, therefore, are called "free" codes hereafter. In this case all the code words must be explicitly recognizable by means of the error detection circuit.

The advantage of "free" codes is that the degree of freedom in allocating codes, that allow an optimal use of chip area, is not limited by a restrictive error function. The resulting chip area for the FSM *without* the error detection circuit can, therefore, be small compared to circuits encoded with separable encodings.

In order to detect all single bit errors, the Hamming distance between all code words must be $d \geq 2$. As described in section 3.1.2, this requirement is not applicable in dynamic error detection since no clear implementation of an error function is possible.

One disadvantage is that the chip area of the required for the error detection circuit is large compared to the error detection circuits with a clearly defined error function. For this reason, the total chip area was calculated and compared to other methods in section 4.2.

4 Results

A control unit with the instruction set of Motorola's standard microprocessor M68HC11 was designed in VHDL. Over 70 different fault secure encoded FSMs were synthesized with the tools from the company Synopsys². The theoretical results have also been taken into account.

The resulting circuits with the smallest chip area for every encoding type were selected for fault simulation. The error detection circuits were designed and included. The complete circuits were validated with a fault simulation on gate level for stuck-at faults using the tool HILO³.

4.1 Characteristic properties of the control unit

A flat FSM hierarchy was purposely chosen for 615 states in order to make use of a large optimization potential created by the broad range of code words. In this way the additional redundancy that is necessary for error detection can be kept small.

After the optimization of the FSM with Synopsys, 161 equivalent states were compiled and the remaining 454 states were extracted in to a state table. The states were assigned [10] to different fault secure codes in the state table and finally the new circuit was synthesized.

Self loops were not allowed. This was achieved by means of adjacent states of the same function being assigned alternately.

The node with the most branching of the state transition graph is assigned the state for the ifetch instruction. This state has 218 adjacent states. Therefore, based on the results in section 3.1.1, a dependent encoding with dynamic error detection for the circuit investigated here was not applicable.

4.2 Synthesized chip areas

The control unit was reduced to equivalent states using Synopsys and synthesized with the option "auto-encoding", which provides the best results in regard to chip area. The result of this effort serves hereafter as a comparative value and is always specified as a reference circuit.

In Table 1 the synthesized chip areas of the complete circuits, the fault secure encoded control unit plus the respectively error detection circuits, are shown in comparison to the reference circuit.

For all of the synthesis runs the timing constraints remained unchanged. However, they had to be kept at least as an optimization target.

An encoding method described in section 3.2.2 is called "free" encoding. A hamming distance $d \geq 2$ between all code words are realized with a "free" encoding. An error function between two code words or between the information bits and check bits are not demanded. Free parameters, therefore, can be used for chip area minimization using heuristics regarding the output function of the encoded FSM [10]. With this encoding an circuit can be synthesized (without the error detection) with a smaller chip area in comparison to the reference.

An area minimization by increasing the number of flip-flops of the state vector can, therefore, be achieved in some cases, especially in large designs. This potential can also be used for error detection.

4.3 Fault simulation

The best solutions for each encoding method regarding the chip-area were selected for the fault simulation as shown in Table 1.

All percentages refer to the reference circuit with no concurrent error detection is possible. The fault coverage refers to the stuck-at faults that were detected with the employed encoding and the respective error detection circuit. The set of the test vectors is equivalent to the complete instruction set of the processor M68HC11. Therefore, the fault simulation using the stuck-at model is representative for the error detection during the normal operation. This is true, because either a fault affects no errors on the outputs or the defect results in a bit error. This considered always real time error detection and is not true, if an off-line test is used.

Two different fault simulation methods are compared: the on-line (concurrent check) error detection as the aim of this paper and additionally the off-line error detection, as shown in Table 1.

In the first case, the on-line error detection is compared to all *observable* stuck-at faults, including the error detection circuit, using only the two error signals, as shown in Figure 1 and 2. All stuck-at faults of the FSM output logic are not observable. Therefore the second column regarding the on-line fault coverage in Table 1 shows the results compared with all possible stuck-at faults of the total circuit, including the error detection circuit.

The codes with a larger state vector length compared to each other are better in error detection re-

²Synopsys is a registered trademark of Synopsys Inc., Mountain View, CA, USA

³HILO is a registered trademark of GenRad Inc., Hants, UK

Encoding Method	State Vector Bits	Gate Area			Fault Coverage		
					on-line		off-line
		FSM	EDC	Total	Observable	All stuck-at	All stuck-at
Reference	9	100 %	0 %	100 %	0 %	0 %	99.4 %
Even Parity	10	104.0 %	1.4 %	105.4 %	98.5 %	60.3 %	99.2 %
Odd Parity	10	105.1 %	1.4 %	106.5 %	98.9 %	60.3 %	99.2 %
Hamming-Code	20	115.4 %	3.4 %	118.8 %	99.3 %	80.5 %	99.4 %
Mod. Hamming-Code	20	113.6 %	3.4 %	117.0 %	99.6 %	80.1 %	99.4 %
Mod. Hamming-Code	24	111.8 %	4.2 %	116.0 %	98.8 %	84.1 %	99.5 %
Berger-Code	21	102.7 %	6.4 %	109.1 %	96.2 %	77.3 %	98.5 %
Berger-Code	28	99.3 %	8.9 %	108.2 %	96.3 %	84.6 %	99.2 %
"Free" Encoding	20	90.0 %	18.1 %	108.1 %	83.5 %	70.3 %	91.1 %

Table 1: Gate areas (FSM, Error Detection Circuit and Total) in comparison with the reference circuit (100 %) and the results for the respective fault coverage

garding all possible stuck-at faults. This is because the additional flip-flops the depth of the logic is reduced. The part of the FSM output logic becomes smaller and the observable stuck-at faults become higher.

In order to compare the proposed error detection method with an off-line test, the results for the complete circuits regarding all possible stuck-at faults are computed. All outputs of the FSM and the two error signals are checked. It should be noted that with this type of test an error detection during the normal operation is not possible. Furthermore the stuck-at model does not completely describe the physical defects, as the test is not in real time mode.

An other aspect is, that through the additional error detection circuit and the fault secure encoding, the off-line testability is not reduced. Therefore there is a usefull possibility to combine these two methods.

5 Summary

In order to detect bit errors in operating control units, the error encoding with an error detection circuit is appropriate. The availability of fail critical systems is not limited by on-line error detection.

The instruction set of the standard processor M68HC11 was realized in a fault secure control unit. The theoretical results regarding the different dependencies between the chip area and the state encoding were proven qualitatively and quantitatively in over 70 synthesis runs.

It was shown that with an increase of the relatively gate area by about 8 %, an on-line error detection with about 84 % can be achieved. During this, the availability is not reduced. The potential of the state encoding benefits is high. Therefore, improving fault security, reducing chip area, improving testability, combined with the efficient handling of timing constraints should be aspects of future research.

Acknowledgments

This work has been part of the European Jessi-AE11 project supported by the german federal ministry "Bundesministerium fuer Bildung, Wissenschaft,

Forschung und Technologie (BMBF)". We wish to thank Dr.-Ing. Ingo Kreuzer, Robert Bosch GmbH, Reutlingen, Germany, for many valuable suggestions and the extensive support of this work.

References

- [1] JESSI AE11, "Ultra-Large Scale Integration of a Control Unit for Safety-Critical Systems", Robert Bosch GmbH, 1994.
- [2] M. Nicolaidis, "Efficient Implementations of Self-Checking Adders and ALUs" *International Symposium on Fault-Tolerant Computing Systems*, pp. 586 – 595, 1993.
- [3] J. Wakerly, *Error Detecting Codes, Self-Checking Circuits and Applications*, North-Holland, 1978.
- [4] W. W. Peterson and E. J. Weldon, *Error-Correcting Codes*, The MIT Press, 1986.
- [5] T. R. N. Rao and E. Fujiwara, *Error-Control Coding for Computer Systems*, Prentice-Hall, 1989.
- [6] E. Fujiwara and D. K. Pradhan, "Error-Control Coding in Computers" *IEEE Trans. Computers*, pp. 63 – 72, 1990.
- [7] M. C. Lin, "Constant Weight Codes for Correcting Symmetric Errors and Detecting Unidirectional Errors" *IEEE Trans. Computers*, pp. 1294 – 1302, 1993.
- [8] K. De, C. Natarajan, D. Nair, and P. Banerjee, "RSYN: A System for Automated Synthesis of Reliable Multilevel Circuits" *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 186 – 195, 1994.
- [9] R. Leveugle, "Optimized State Assignment of Single Fault Tolerant FSMs Based on SEC Codes" *IEEE Design Automation Conference*, pp. 14 – 18, 1993.
- [10] T. Buchholz, "Zustandskodierung fuer fehlersichere Steuerwerke" Master's thesis, University Karlsruhe, 1994.