Channel Coded Processor for Enhanced Safety

A. Steinkirchner, T. Fuhrmann, and M. Niemetz

***Abstract*—Several concepts are known for improving processor safety, all of them having their pros and cons. Some are very resource intensive, others have limited capabilities regarding the provided error safety. In this paper we use research on channel coding of noisy communication channels known from communication theory as an analogy to random bit errors in processors. We incorporate this knowledge into processor design to suggest a new error correction concept by using channel coding in processors. The new concept of a Channel Coded Processor could provide effective implementation of redundancy by the channel coding that enables error correction. The concept could also create a complete chain of redundancy in all areas and components of the processor ranging from the code compiler through the processing hardware to the output of the information.**

***Index Terms*—Channel coded processor concept, communication theory, safety, soft error.**

I. Introduction

From the early beginning of computers the challenge is to obtain reliable calculations. Errors are especially dangerous for the integrity of results when not leading to computer crashes but giving incorrect results. Many errors are caused by temporary state changes which do not lead to permanent damage of computer hardware and are therefore called soft errors. There are two types of soft errors: chip-level soft errors and system-level soft errors.

Chip-level soft errors can be caused by charged particles or ionizing radiation that interact with the semiconductor material and create electron-hole pairs within a pn-junction. This can lead to a charge reversal of circuit nodes and thus to a change of the digital information, a so-called “bit flip” [1].

Tests have identified several specific design factors which influence error rates of chip-level soft errors [2]-[4]:

- Higher-density chips are more likely to have errors.
- Lower-voltage devices are more likely to have errors.
- Higher speeds (lower latencies) contribute to higher error rates.
- Lower cell capacitance causes higher error rates.
- Shorter bit-lines result in fewer errors.

System-level soft errors occur when the data being processed is influenced by an electromagnetic wave which originates from outside or inside the chip. This noise phenomenon leads also to a change of digital information; a “bit flip” takes place.

For the classification of the requirements for safety of processor-based systems, different standards have been introduced with the most important ones being the IEC 61508 “Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems” and its adaptation for the automotive industry, the ISO 26262 “Road vehicles – Functional safety”. The safety integrity levels of those standards are not easily to achieve without error correcting measures.

TABLE I: Comparison of Established Concepts

	EDC	**ECC**	**Lock-Step**	**SoR**	**fR**
Hardware Expenditure	low	low	mid	high	low
Software Expenditure	low	low	mid	high	low
Forward Error Correction	no	yes	no	no	no
Limitations to Certain Elements	RAM	RAM	CPU	no	no
Safety of Address and Control Bus and Lines	no	no	CPU	yes	no
Safety begins at Compilation	no	no	no	no	no
Protection of the additional Hardware	no	no	no	no	no

In the next section established failure tolerant processor concepts are explained. Section III shows the analogy between a processor with soft errors and a noisy communication channel. The possibilities of a new processor concept which includes communication theory is explained and compared to existing safety concepts in Section IV. Conclusion and outlook are presented in Section V.

II. Established Failure Tolerant Processor Concepts

There are several well-known concepts to reduce failure probability within processors. Error detecting codes (EDC) are typically simple parity codes which can detect an odd number of errors but cannot correct errors. Error correction codes (ECC) can detect and correct a specified number of errors [5]. These two concepts are typically used for ensuring the correctness of data stored in memories.

Redundant processors are designed with multiple parts up to the entire processor. State of the art concepts are the Lock-Step [6] and Sphere of Replication (SoR) [7]. Here, the CPU or the main core components are realized several times.

Another concept is to equip error-prone areas of the processor with diagnostics that conduct systematic monitoring and therefore recognize soft errors. Here, state of the art is the faultRobust (fR) technology [8] which is based on a detailed “Failure Modes and Effects Analysis” (FMEA) for a selected part of components of the processor [9].

Each solution has its pros and cons and therefore its preferred applications. In Table I the different error detection and correction systems for processors with their outstanding

Manuscript received February 14, 2017; revised March 30, 2017.

The authors are with the Faculty of Electrical Engineering and Information Technology, OTH Regensburg, Germany (e-mail: a.r.steinkirchner@gmx.de, thomas.fuhrmann@oth-regensburg.de, michael.niemetz@oth-regensburg.de).

doi: 10.18178/ijfcc.2017.6.2.489

and missing features are listed.

We will discuss the new concept of Channel Coded Processors in the following chapters.

III. Processor as a Communication Channel

Communication theory has developed many algorithms and channel coding schemes for a nearly optimal communication via a noisy channel. In the following sections the analogy between a noisy communication channel and a processor suffering from soft errors is used to apply concepts from communication theory to data processing in processors.

A. General Model of a Noisy Communication Channel

The general model of a noisy communication channel is based on the well-known Shannon's communication theory published in 1948 [10]. It uses the term entropy as average information content to define the concentration of information.

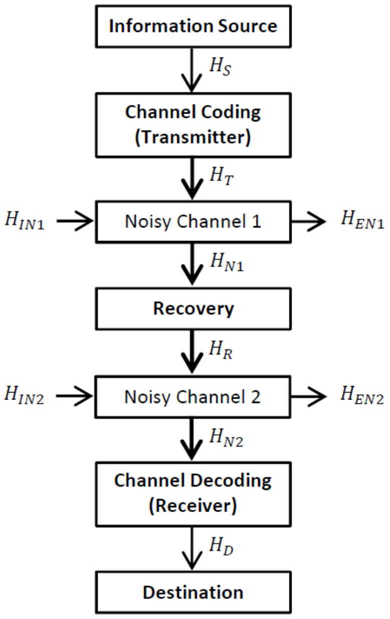

Fig. 1. General model of a noisy communication channel.

Fig. 1 shows a model of a noisy communication channel. Here, the information source has the entropy H_S. This information is forwarded to the transmitter, which performs a channel coding using an alphabet with a specified probability of occurrence for all symbols. The entropy at the transmitter H_T is

$$H_T = H_S + R_T \tag{1}$$

with the entropy of the source H_S and the redundancy of the channel coding R_T. The coded message is sent through the channel and therefore the redundancy which is the stochastic dependence among the symbols is interpreted as information by the channel and therefore as additional entropy. The distortion of the channel can be described by the conditional entropies irrelevance and equivocation as follows:

The irrelevance H_{IN1} describes the uncertainty of characters received at a given transmission symbol. The transfer through a degraded channel is regarded as a random experiment which contributes to uncertainty as the noise of the channel is itself a source of information causing a distortion of the transmitted information.

The equivocation H_{EN1} represents the uncertainty of the transmitted signal at a given receiving symbol.

In case of an error-free channel, the equivocation is zero and the information is propagated from the channel input to the channel output without a change in entropy. If the transfer of information is completely disturbed, $H_{EN1} = H_T$ and no transport of information from input to output is possible at all.

The exchanged information through the first noisy channel (see Fig. 1), which is a discrete, memoryless channel, is represented by the mutual information I_{N1}. I_{N1} is the part of the transmitter's entropy H_T, which can be transmitted to the receiver through the noisy channel 1 [11]:

$$I_{N1} = H_T - H_{EN1} = H_{N1} + H_{IN1}. \tag{2}$$

The mutual information I_{N1} is not negative, with the value zero indicating the independence of the transmitter and receiver, which means all the information is lost in the noisy communication channel and replaced by noise.

During channel decoding at the receiver the irrelevance has to be removed and the equivocation has to be restored. Obviously, for the perfect recovery the mutual information I_{N1} has to be exactly equal to the entropy H_S of the source. To recover the full information of the source using the decoding, the remaining redundancy R_R, i.e. the mutual information I_{N1} minus the entropy H_S of the source, has to be greater than or at least equal to zero as seen by the channel:

$$\begin{aligned} R_R &= I_{N1} - H_S \geq 0 \\ R_R &= H_{N1} - H_{IN1} - H_S \geq 0. \end{aligned} \tag{3}$$

It depends on the efficiency of the code and its decoding algorithm how much from the remaining redundancy has to be used for the decoding and therefore which value of R_R is at least needed for a successful decoding. After a perfect recovery, the entropy H_R at the receiving recovery stage is identical to the entropy at the transmitter $H_R = H_T$.

The transmission through the second noisy channel and the associated recovery included in the receiver are analogous to the processes described above for the first channel. Finally, the received symbols at the receiver have to be decoded using the alphabet of the channel coding before being used at the destination.

It is assumed in this model that the transmitter including channel coding, the recovery as well as the receiver including the channel decoding are error-free and thus are not part of a noisy channel.

B. Adaptation for Processors

To describe a processor using the general model of a noisy communication channel as shown in Fig. 1, the channel characteristics need to be defined. The processor represents a discrete, binary, memoryless and systematic channel. A discrete channel means, the in- and output of the channel are operating with discrete values, i.e. only a finite alphabet is used. For a binary channel, the alphabet only contains the possible values zero and one. The characteristic memoryless means, that a value in the channel is not influenced by the previously transmitted values. For symmetric channels, the probability of a “bit flip” does not depend on the direction of the “bit flip”. This type of channel, called binary symmetric channel (BSC), is a standard channel, frequently treated in detail in literature of communication theory [11].

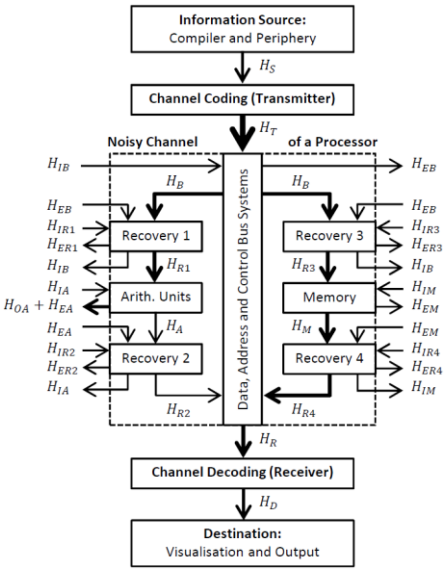

Fig. 2. Application of the general model of a noisy communication channel for the specific case of processors.

Fig. 2 shows the application of the noisy channel model to a processor. The source consists of the compiler which initially generates the program data and writes it into the target computer memory. These data with the entropy H_S are channel coded and through this transferred to H_T before they are passed on to the noisy channel, i.e. the target computer. The entropy is unchanged by the transmitter from the perspective of the source, since only redundancy R_T is added during the channel coding. From the point of view of the channel, the added redundancy is considered as information. So for the channel the entropy at the channel input is analog to equation (1).

As an implementation possibility the noisy channel of a processor consists of three different sub-areas which are each separated by recovery stages. These are structured into arithmetic units, memory and bus systems and hence represent all components of a general processor. Here, the three bus systems, which are the data, address and control buses, connect the arithmetic and logic units to the memory. They also represent the interfaces of the channel to the transmitter and receiver.

All components of these three different sub-areas and their subsequent recovery stages are seen as noisy subchannels of the noisy channel of the processor; for each of the components the general model of a noisy communication is applicable. Therefore, they are all potentially affected by soft errors. The resulting equivocations and irrelevances of the sub-areas are denoted by H_{Ex} and H_{Ix}, where x marked the first letter of each sub-area. Similarly, the irrelevance and equivocation of the four recovery stages are referred as H_{ERx} and H_{IRx}.

Another equivocation H_{OA} can occur in the sub-channel of the arithmetic and logic units. During some of the operations up to half of the entropy may be lost (e.g. when adding up two n-bit values resulting in a new n-bit value). This reduces entropy by H_{OA} which is also indicated by a thinner arrow of H_A in Fig. 2.

The recovery stages can recognize redundancy due to the channel coding, and thus identify and correct errors. Thereby the irrelevance H_{Ix} of the preceding sub-channel is removed and replaced with the restored equivocation H_{Ex} of the preceding sub-channel. This is only possible, if the remaining redundancy of this sub-channel fulfils the condition in equation (3), since it is also a noisy channel. The recovery stages are also subject to soft-errors and therefore the subchannel equivocation H_{ERx} is removed and irrelevance H_{IRx} is added. For Recovery 1, the inflow of entropy is the added redundancy by the recovery stage itself H_{IR1} and its restored equivocation of the preceding bus systems H_{EB}. The outflow is the equivocation caused by possible errors of the recovery stage H_{ER1} and the irrelevance of the preceding bus systems H_{IB}. The flows of entropy for all four of the here depicted recovery stages are identical.

Also the data receiver is analogous to the general model of a noisy communication channel. The received symbols have also to be decoded before the destination, which includes the recovery of information. The mutual information I_P that is transmitted through the noisy channel of a processor, can be estimated as follows:

$$I_P = H_T - H_{EP} - H_{\sum OA} = H_R + H_{IP}. \qquad (4)$$

In the equation, H_{EP} represents the equivocation of the entire channel which could not be corrected by the recoveries. $H_{\sum OA}$ is the sum of the equivocation caused by the arithmetic and logic operations, which depends on the processor operations. H_{IP} is the irrelevance of the entire channel, which is caused by the soft errors. In analogy to equation (3), to recover the full information of the source minus the loss of information during arithmetic and logic operations using the decoding, the remaining redundancy R_R has to be greater than or at least equal to zero:

$$\begin{aligned} R_R &= I_P - (H_S - H_{\sum OA}) \geq 0 \\ R_R &= H_R - H_{IP} - H_S + H_{\sum OA} \geq 0. \end{aligned} \qquad (5)$$

The same statements about the required redundancy by the code as mentioned in equation (3) apply here, too.

IV. Discussion on the Channel Coded Processor Concept and Comparison with Established Safety Concepts

A. Characteristics of a Code for the Channel Coded Processor Concept

For the application of the general model of a noisy communication channel to a processor, one of the most important details is the characteristics of the selected code. It has to fulfil the following requirements:

- The code must be a block code or a finite convolution code, since the information of the data words should have no connection with each other.
- The code must be available in a systematic form or be isomorphic to this, so that the information processing can be performed in the established and well-known way. Each block code can be transferred through line operations in the generator matrix without compromising the properties of the code in a systematic block code.
- Each of the operations that are executed by the processor on data words has a corresponding operation on the correction bits.

The remaining degrees of freedom that are left in the mathematical construction of the code can be used for optimizations. These include, among others, the efficient and effective implementation of error correction with the smallest possible amount of circuits.

B. Key Questions of the Processor Concept

The main challenge is the construction of the channel code. This has to involve the information of all possible operations the processors are capable to perform, as this influences the required corresponding operations on the correction bits. This leads to mathematical construction rules of the code, and to a decrease of the efficiency of the code.

Another question is the combination of the code with data, address and control lines in the processor. An undetected error, for example, in the control lines of the ALU may change the operation on two data words. The digital circuits that are used for these lines must contain redundancy to correct soft errors, which also occur there, by itself. Any information in the processor must be protected at all times in some form by redundancy. This is especially true for the recovery within the noisy communication channel of the processor. Here, an error propagation of individual errors in arithmetic combination of words can occur. An error at one place of an input word can affect adjacent bits of the output word.

V. Conclusion and Outlook

We developed the idea of a new concept for enhancing the safety of processors. The aim is to integrate modern concepts of signal transmission theory trough noisy communication channels with concepts for safe processors.

Currently, we think that this new concept has the potential for an effective error correction scheme within a processor. Compared to the established concepts, this concept provides a consistent and effective implementation of protection against soft errors enabling error detection and correction. Soft errors originated in a recovery stage may be corrected in the following recovery unit as long as the error doesn't generate a valid but incorrect code word. We assume that the additional hardware required for the new concept in relation to the performance of error correction could be very efficient.

The concept can also create a complete chain of redundancy in all areas and components of the processor that ranges from the beginning in the compiler and the periphery until the output of the information.

We know that many questions remain open. The points

- mathematical description of the noisy communication channel model for a processor,
- channel code having the required characteristics mentioned above,
- algorithms for error correction,
- concept for self-correcting digital circuits and transition to data, address and control lines and
- implementation and testing of a simple processor into a Field Programmable Gate Array (FPGA)

are planned for further research.

In the first step, known codes of channel coding can be tested for their usability for the model of the channel coded processor described in Fig. 2. First, only systematic block codes are examined for the following two reasons:

- Block codes guarantee the information-theoretical independence between individual code words.
- By means of the systematic coding, the arithmetic and logic operations of the data words in the processor can be executed in the usual manner. The associated check bits can be computed with a calculation rule which takes the operation on the data words into account and thereby generates a valid code word.

If no suitable calculation rules can be found on the check bits, an attempt can be made to modify the code using the well-known methods: extension, punctuation, expurgation and shortening.

These two postulates concretize and simplify the possible realization of a block code mathematically, but this leads probably to a deterioration of the efficiency of the code if the information of the postulates cannot be used efficiently for the error correction.

We appreciate discussions on open questions and further development possibilities of this proposed concept.

References

[1] T. Juhnke, "Die soft-error-rate von submikrometer," *CMOS Logikschaltungen*.

[2] R. Baumann *et al.*, "Boron compounds as a dominant source of alpha particles in semiconductor devices," *Reliability Physics Symposium, 1995. 33rd Annual Proceedings*, IEEE International.

[3] C. Slayman, *Soft Error Trends and Mitigation Techniques in Memory Devices*, 2010.

[4] T. Semiconductor, *Soft Errors in Electronic Memory — A White Paper*, Naperville, 2004.

[5] M. Y. Hsiao, *A Class of Optimal Minimum Odd-Weight-Column SECDED Codes*, 1970.

[6] W. Wong, *Lock Step Microcontroller Delivers Safe Motor Control*, 2012.

[7] Freescale Semiconductor, *Qorivva MPC5643L Microcontroller Data Sheet*, 06/2013.

[8] R. Mariani *et al.*, "A platform-based technology for fault-robust SoC design," *Design & Reuse*.

[9] R. Mariani and P. Fuhrmann, "Comparing fail-safe microcontroller architectures in light of IEC 61508," presented at 22nd IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, 2007.

[10] C. E. Shannon, "A mathematical theory of communication," Dissertation, Bell System Technical Journal, 1948.

[11] M. Werner, *Information und Codierung. Grundlagen und Anwendung*, Wiesbaden: Vieweg+Teubner; Vieweg, 2008.

Alfons Steinkirchner was born in 1987. He studied from 2008 to 2012 electrical engineering and information technology (B. Eng.) and from 2012 to 2016 electrical and microsystems engineering (M. Eng.), both at OTH Regensburg. Since his B. Eng. he works at REWAG as head of the energy and grid control center. After his M. Eng. he has started a doctorate at Faculty of Electrical Engineering and Information Technology at the OTH Regensburg.

Thomas Fuhrmann was born in 1971. He studies from 1989 to 1993 at the FH Coburg electrical engineering. He made his Dr.-Ing. at the TU Berlin. From 1998 to 2006 he worked at different companies as a research and development engineer. Since 2006 he is a professor for electrical engineering at the OTH Regensburg. From 2010 to 2015 he was the dean of the Faculty. Since 2015 he works as vice president for international contacts.

Michael Niemetz was born in 1971. He studied from 1990 to 2001 mathematics and physics at Regensburg University. After his PhD in low temperature physics he worked at SiemensVDO and continental as software architect for engine management control software. Since 2011 he is professor for electrical engineering at the OTH Regensburg and since 2015 he is the dean of the faculty. Among his main interests are software architecture and programming of embedded real time devices.