

Chapter 6

Conclusions

In this thesis we have shown that methods for improving the reliability of digital systems which are based on N -modular redundancy can be generalized to methods which are based on a distributed implementation of error-correcting codes. These methods have been called “Generalized Masking Redundancy”

Within the class of fault-tolerant systems based on generalized masking two sub-classes can be identified which characterize the systems that are based on generalized masking.

The first class requires, after repair, some state initialization from the environment and is identified by the a distributing function \mathcal{X} , (the encoder function of an error-correcting code), an observing function \mathcal{Y} (the decoder function of an error-correcting code) and the number T of maliciously behaving modules which are tolerated.

The second class re-initializes itself each time instance and is identified by a distributing function \mathcal{X} , an observing function \mathcal{Y} , a state decoder \mathcal{Z} (the decoder function of an error-correcting code) and the number T of maliciously behaving modules which are tolerated.

It has been shown that in practical cases none of the two classes will suffice. Firstly, because re-initialization by external means causes the reliability of the system to depend on the environment. And secondly, because re-initializing the entire state each time instance is too costly. So practical implementations will be a mixture of these two classes, i.e. the system will re-initialize itself within a particular span of time.

An example of an architecture based on generalized masking is called the (N, K) -concept, of which the feasibility is proved by application in a commercial system. Due to the concept of generalized masking on which the (N, K) -concept is based the following is gained

- The ratio between memory and processor redundancy can be optimized with respect to the total amount of hardware needed or with respect to the call-rate of the system.
- Codes can be introduced which are capable of correcting both symbol and bit faults without requiring additional redundancy. In this way the code which is usually applied for memory protection is saved.
- The number of modules (fault isolation areas) required for fault tolerance can be adapted to the number of modules required for consistent communication with the environment.

The existence of codes which are able to correct both symbol and bit errors without requiring additional redundancy is shown with the presentation of such a symbol and bit-error-correcting code for the $(4, 2)$ -concept. Moreover we have shown that a one-chip decoder for this code can be designed with a propagation delay of about 100nsec.

One of the worst problems in fault tolerance is the Input Problem. If special precautions are not taken, a correctly functioning fault-tolerant system might go down due to external faults. We have shown that this Input Problem is similar to the Interactive Consistency problem.

Interactive Consistency Algorithms give rise large numbers of messages which need to be exchanged between the modules. This limits their application to systems in which at most 3 or 4 faulty modules are tolerated.

In order to reduce the number of messages which need to be exchanged, a new class of Interactive Consistency Algorithms based on voting and coding has been defined. In practical applications in which less than 4 maliciously behaving modules are tolerated, these synchronous deterministic Interactive Consistency Algorithms turn out to be superior to their existing counterparts.

The properties of the class of Interactive Consistency Algorithms based on voting and coding have been proven on the basis of a newly defined class of

algorithms called Dispersed Joined Communication algorithms which have more liberal properties.

Finally, four algorithms based on Dispersed Joined Communication algorithms and Interactive Consistency algorithms have been presented which solve the Input Problem.

In this thesis we restricted ourselves to fault-tolerant systems in which the means for reliability improvement are implemented rather close to the hardware level. In general this turns out to be a cost-effective approach, both from the point of view of system cost and from the point of view of separation of concerns during the design process. It enables the development of software independently from the reliability requirements.

Implementation of the means for reliability improvement close to the hardware level naturally leads to a synchronous deterministic approach. This holds for the system fault tolerance as well as for the algorithms which solve the Input Problem.

However, if one wishes to start from off-the-shelf modules, the approach followed in this thesis will often not suffice. Firstly, a less strict definition of synchronism will be needed which is based on only time-outs. This will require adapted fault-tolerant synchronization algorithms. Secondly, the algorithms for solving the Input Problem will be applied on a higher level, in which case algorithms based on authentication will be cheaper. Much work in this “quasi asynchronous” field has still to be done.

The reliability of systems not only depends on the reliability of their constituting physical components but also on the correctness of the design. The field of specification and design description with respect to fault tolerance has hardly been explored. Only on the basis of formal description methods can proof systems be designed. Any result in this field will ease the validation problem of fault-tolerant systems.