

Linux based demonstrator of a failure tolerant numerical control system

Hans-Peter Bock

Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW)
Seidenstraße 36, 70174 Stuttgart, Germany
Hans-Peter.Bock@isw.uni-stuttgart.de

Prof. Dr.-Ing. Alexander Verl

Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW)
Seidenstraße 36, 70174 Stuttgart, Germany
Alexander.Verl@isw.uni-stuttgart.de

Abstract

In current research work the Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW) has developed a continuous concept for a redundant, failure tolerant numerical control system. The results of this research project meet the increasing demands on the reliability of machines and plants and also the frequent applications of numerical controls in safety-critical areas. In this case, reliability means the capability of a numerical control to function according to its specification during the entire machining time. This is achieved by a redundant design and will be shown on a demonstrator. This demonstrator based on linux is introduced in this paper.

1 Introduction

In the meantime, numerical control systems are not only used in the conventional applications like milling or drilling. Meanwhile they also control fairground rides or surgery assistants [1]. These new application areas place higher demands on the system reliability than the manufacturing applications. The safety-oriented approach where a machine is stopped in case of a failure is no longer adequate. In the case of fairground rides for example this means that the passengers have to hold out for a longer time in an uncomfortable position. With surgeries a failure would cause a prolongation and possible complications. With certain surgery methods, as for example the endoscopic method with surgery assistants, a small hole is drilled through the skullcap. For a manual intervention the opening would have to be

enlarged considerably [2].

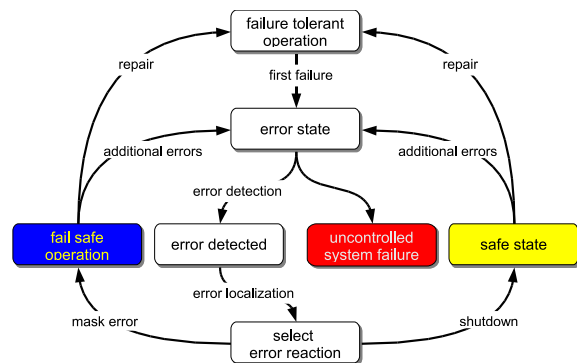


FIGURE 1: states in a failure-tolerant control system [3]

Figure 1 shows the intended states in a failure-tolerant control system [3]. Usually the control system is in a state of failure-tolerant operation. After the first failure the system is in a state of failure. If the failure is not detected, the system will sooner or later fail. This must be avoided. Therefore, it is nec-

essary to identify occurring failures and to localize them. After the localization of the failure, an adequate reaction has to follow. With safety-oriented systems a safe state is aimed for, which is mostly achieved by stopping the machine.

If a safe state can only be reached after the machining process has finished, a shutdown of the system cannot be accepted. In this case the operation has to be continued, which can be achieved without interruption by masking the failure. Masking a failure means that faulty values are replaced by correct values by several and independent repetitions of the same calculations. After the masking the system is only in a failure-safe operational state, but it can still continue with the machining.

The objective of this research work is the increased reliability of the control system. This means that the specified function of the control, the generation of correct setpoints, is still maintained in spite of occurring failures. Here the masking of failures during operation is necessary.

2 State of the art

In control systems there are already many mechanisms for failure detection and the reaction to them. Figure 2 shows the common methods for the various components.

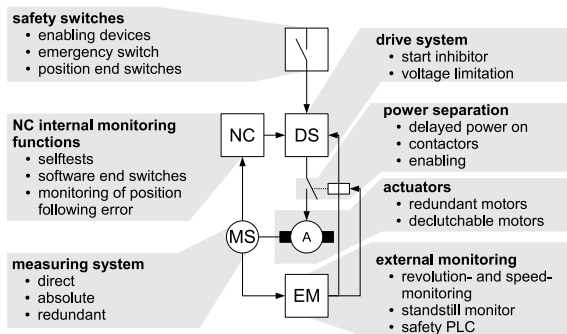


FIGURE 2: *common methods for failure detection for the various components [3]*

For all components there are relating failsafe and redundant parts. With redundant measuring systems the measured values can be checked for plausibility and thus individual sensors can be tested for their error-free function. Monitoring units check whether certain measuring values surpass their permitted value range and take safety measures if needed. Actors can be redundant in order to compensate the failure of a particular actor.

But for the control itself, the central component of the control system, there is no overall failure detection. It is nearly impossible for the rest of the components to detect that the control generates faulty setpoints. The existing measures for error detection in the control like self-tests, software end switches, etc. are suitable to only a limited extent for the detection of all failures. For masking the failures they are not sufficient.

2.1 Failure classification

As shown in figure 3, failures can originate from three different places. Outside the system, in the the system hardware or in the system software. Failures outside the system are failures of the basic conditions, which the system relies on. One example for that is the power supply. Because failures outside the system can independently be detected and handled, this work focuses on failures which originate inside the system boundaries.

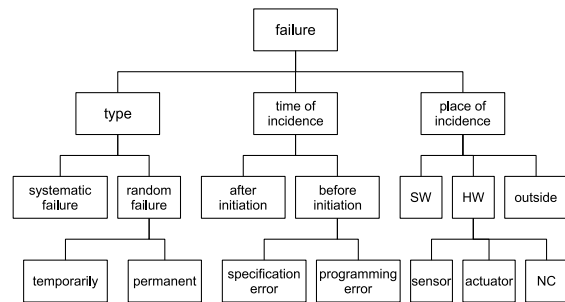


FIGURE 3: *classification of failures [3]*

Failures inside the system can either be present in the software or occur in the hardware. Software failures are generally systematic failures. They consist of specification- or programming errors and get into the system before its initiation. An error free software will never generate wrong values since software does not grow old. But a software containing errors will always generate wrong values.

Software errors can either be avoided before the initiation or be detected during operation. Actions to avoid software errors are software analysis, failure simulation tests or the usage of special tools which analyse source code. Actions to detect software failures is additional monitoring during operation like range or plausibility checks. Another possibility to detect software failures during operation is to cross check the values computed by redundant and divers developed software.

Hardware failures can either be systematic or random. Systematic hardware errors happen before

the initiation. They originate of specification, design or programming errors. These systematic hardware failures can be detected during operation only by divers and redundant hardware.

Random hardware failures occur during operation. They can be detected by several self tests which check e.g. the processor or memory. It is difficult to run these self tests during operation, since the control will need most of the system resources for itself. With redundant hardware it is possible to detect random hardware failures during operation by cross checking the computed values.

The main focus of the work is put on the masking of systematic and random hardware failures. In order to safely identify hardware failures, the control hardware needs to be designed redundantly and diversely. Redundantly, so that an accidental failure is identified by the redundant control system. Diversely, so that systematic specification and production failures do not occur in the redundant systems at the same place and at the same time.

In aviation as well as in mechanical engineering such redundant and divers control systems are employed in safety-oriented programmable logic controls (PLCs) with "Fly by wire". For particularly challenging applications of PLCs there is a failure-tolerant type with triple redundancy, the so called "Triple Modular Redundancy". For numerical controls there is up to now only one failure-safe version. It will be presented in the following section.

2.2 The concept of the failsafe control system

Figure 4 shows the existing concept of a failsafe numerical control system [3]. It consists of two numerical controls which run on divers hardware platforms. The two controls are synchronized by a shared memory by which they also exchange computation results. The machine only receives setpoints by control #2.

Each of the two controls contains a reaction manager. He compares the own computed values with those of the other control. If he notices differences, he can autonomously disable the actors.

If in this control system a failure occurs in one of the hardware platforms, it will lead to falsely computed setpoints. Even if the failure is so extensive that the hardware platform cannot recognize it anymore, the redundant platform will diagnose this and

bring the machine into a safe state by shutdown.

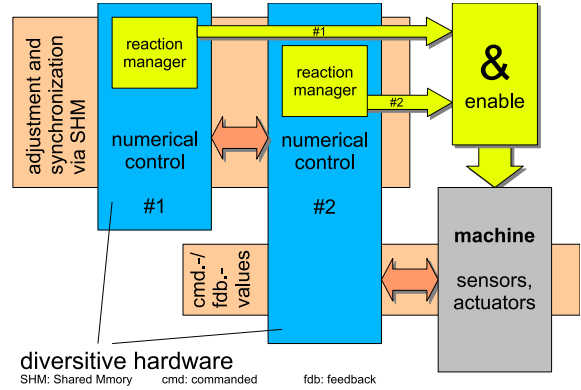


FIGURE 4: *concept of a failsafe numerical control system*

3 Deficits

The design of the failsafe numerical control is satisfactory for applications in which by machine shutdown a safe state can be achieved at any time. In failsafe control systems failures may be detected reliably, but the reaction manager cannot localize them in the case of differing values. Due to the lacking failure localization he is not able to mask a failure. There is only the possibility to bring the machine to a safe state. The machining process is interrupted. For applications in which the operation need to be maintained over a specific period, this design is not adequate.

4 Approach

Failures that are not easy to locate are in other application areas - like in the control of airplanes in aviation or in safety PLCs - localized and masked by multiple redundancy. This approach is very complex because three or more systems have to deal with the same task. The computation results of the systems are compared and differences are masked by majority decision. This high expense is only justified when a high reliability of a system is required. Such requirements are less often encountered with conventional machine tools, but more so with medical surgery assistant systems.

In contrast to the safety PLCs which compute a simple logic between input and output, a numerical control conducts per millisecond complex calculations for the timely output of the correct position data. Compared to the failsafe numerical control, the majority decision for masking faulty setpoints is new.

Contrary to a failsafe numerical control the failure-tolerant system needs at least three divers controls. This design is shown in figure 5 [4].

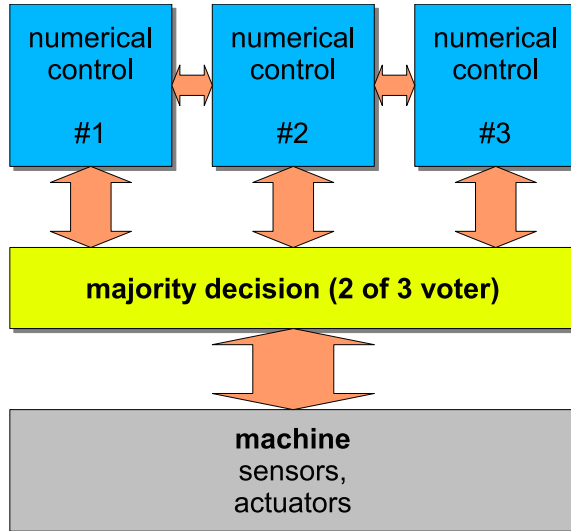


FIGURE 5: *majority decision as approach [4]*

4.1 Voter in every actuator

Figure 6 shows a failure tolerant control layout, in which every single actor contains the necessary voter. All voters receive the setpoints of all controls to pass one setpoint chosen by a majority decision on to the actuator. In case of a failure of a single actuator, the control can mask this failure using redundant actuators.

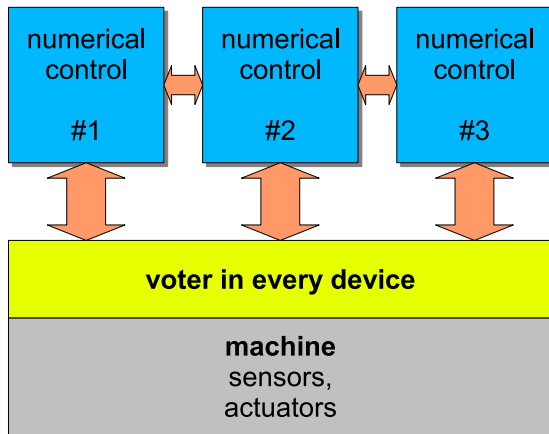


FIGURE 6: *voter in every single actuator*

The advantage of this layout is, that the actuators already contain software. This software can be extended to run the majority decision. A disadvantage is, that it might happen, that different

actors use setpoints of different controls. This can be avoided by synchronization between all actuators which results in a more complex system setup.

4.2 Voter inside communication system

Another possible layout is shown in figure 5. The actuators are not modified. The majority decision takes place inside the communication system. The result is passed on to the actuators.

The advantage of this layout is, that the actuators do not need to be modified. The disadvantage is, that communication systems usually do not possess any computation power. The communication system would have to be extended at an appropriate place to run the majority decision. This extension also needs to be redundant, so that a failure of it does not lead to a breakdown of the whole control system.

4.3 Voter inside numerical controls

Figure 7 shows the possible layout to place the voter inside the numerical controls. The connection from each numerical control to the communication system can be interrupted by the other numerical controls. In normal operation, the first numerical control provides its setpoints to all actuators. The other numerical controls will do the same calculations and synchronize themselves to the first one. If one voter notices incorrect setpoints, it will try to disconnect the faulty control from the communication system. If two controls do this, the faulty control will be disconnected. If the disconnected numerical control has been the one providing setpoints to the actuators, one of the remaining controls will resume this task.

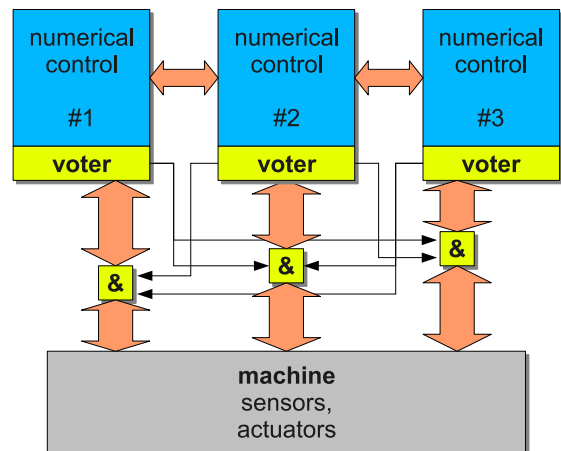


FIGURE 7: *voter inside numerical controls*

The advantage of this layout is, that the actuators do not need to be modified. The voter can easily be integrated into the numerical control systems. The disadvantage is, that the communication system needs to be extended, so that it can disconnect single controls and switch over to another one.

5 System design

For the demonstrator, the failure tolerant layout with the voter in every actuator, figure 6, is chosen. This layout does not need any extensions for the communication system. It also does not insert a single point of failure into the system, as long as redundant actuators are provided.

To work around the difficulty of extending the firmware of actuators, no real machine is applied for the design of the demonstrator. Rather the software package Virtuos by the company ISG is used. This software package allows real hardware in the loop simulation of a complete machine [5, 6] as well as the integration of the required majority decision.

As operating system, linux with rt-preempt has been chosen for its excellent availability and costs. The missing safety certification does not disturb at this point, since this demonstrator is a proof of concept.

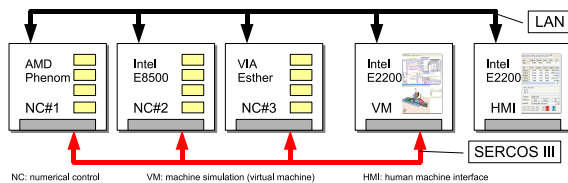


FIGURE 8: *design of the demonstrator*

Figure 8 shows the design of the demonstrator which is based on five conventional PCs with divers hardware as well as network and SERCOS III interface cards for communication. Each of the first three PCs is running linux with a numerical control kernel. One of them has a SERCANS III master pci card from Bosch Rexroth and runs the free linux kernel driver¹. The two others have a SERCOS III slave pci card from Automata with a proprietary slave stack. The fourth PC runs linux and simulates the machine to be controlled with the solver of Virtuos. Connected are these PCs by the ethernet-based industrial communication system SERCOS III. The fifth PC runs

windows for the user interface of the controls and Virtuos as well as the visualization of the simulated machine.

6 Conclusions

The demonstrator setup shows how to redundantly operate numerical controls. The new cross communication concept of SERCOS III is used for the exchange of data as well as for synchronisation. Errors as well as the breakdown of a whole control is tolerated by the concept of synchronised numerical controls and the majority decision based on setpoints. The failure tolerance of the demonstrator will be shown by error simulation tests.

The presented failure tolerant system enables numerical control systems to gain higher reliability. A machine vendor trades of this advantage against a considerably complex system design along with raised costs. Nevertheless, a failure tolerant system is justifiable, if a very high reliability is required. This high reliability is required in fairground rides or in medical applications, e.g. surgery assistants.

References

- [1] G. Pritschow, K.-H. Wurst, A. Haubelt, G. Bauer; *Ergonomisch gesteuertes Chirurgie-Assistenzsystem*; wt Werkstatttechnik online 94; Düsseldorf; 2004
- [2] A. Hein; *Eine interaktive Robotersteuerung für chirurgische Applikationen*; VDI-Verlag; 2000
- [3] U. Laible; *Aufbau numerischer Steuerungssysteme für sicherheitskritische Anwendungen*, Jost-Jetter Verlag, Heimsheim; 2005
- [4] A. Verl, H.-P. Bock; *Fehlertolerante numerische Steuerung*, wt Werkstattstechnik online, H.5, Düsseldorf; 2006
- [5] G. Pritschow, S. Röck; *"Hardware in the Loop" Simulation of Machine Tools*; Annals of the CIRP 53, S. 295...298; 2004
- [6] S. Röck, H. Rüdele, G. Pritschow; *Real-time simulation environment for machine-tools*, Virtual Modelling and Rapid Manufacturing, S. 285...290; 2005

¹SERCANS III linux kernel driver announcement: <http://tinyurl.com/luapuv>