

Fault-Tolerant Communication in Large-Scale Manipulators

H.-D. Kochs¹, W. Geisselhardt², H. Hilmer¹, and M. Lenord

¹ Gerhard-Mercator-University Duisburg, Germany
Faculty of Mechanical Engineering, Department of Information Processing
{kochs, hilmer, lenord}@mti.uni-duisburg.de
<http://www.mti.uni-duisburg.de>

² Gerhard-Mercator-University Duisburg, Germany
Faculty of Electrical Engineering, Department of Dataprocessing
gd@uni-duisburg.de
<http://www.fb9dv.uni-duisburg.de/dve.html>

Abstract. In this paper concepts for fault-tolerant communication systems in large-scale manipulators for heavy weights are introduced. This class of robots makes high demands on safety and real-time behaviour of data transmission, which cannot be fulfilled by conventional communication systems. According to different types of data the communication system has to cope with, two fault-tolerant architectures, based on the CAN-protocol and on the ATM system, are exposed.

1 Introduction

Large-scale manipulators build a special class in the area of big robots. They cover a wide range of applications. Examples are the employment in the building industry, the cleaning and restauration of buildings, the inspection of freeway bridges, the employment in the mining industry and as evacuation equipment in disaster control operations.

As an example, Fig. 1 demonstrates a scenario of the mining industry. It shows a tunnel drilling robot like it is applied nowadays. A specialist handles the robot locally. In the future the operator will work over-meet and remotely observe and control the robot. A head-mounted-display, a data-cage or a data-glove can be employed. All versions of heavy load handling systems have a very complex data processing system, which is clarified on the basis of a further configuration in Fig. 2. It covers a multiplicity of sensors, on the one hand for environment recording, on the other hand for the positioning of the joints with consideration of the elasticity. Further units fulfill those functions: drive, trajectory planning, collision avoidance, man-machine interface (MMI) and monitoring. As a result, various data flow between the components, which require a suitable communication system. Due to the multiplicity of the components, the spatial expansion and the safety and reliability requirements a distributed system structure on the basis of serial bus systems is aimed at. The communication system does not only have to meet high temporal requirements for the fulfilment of the control

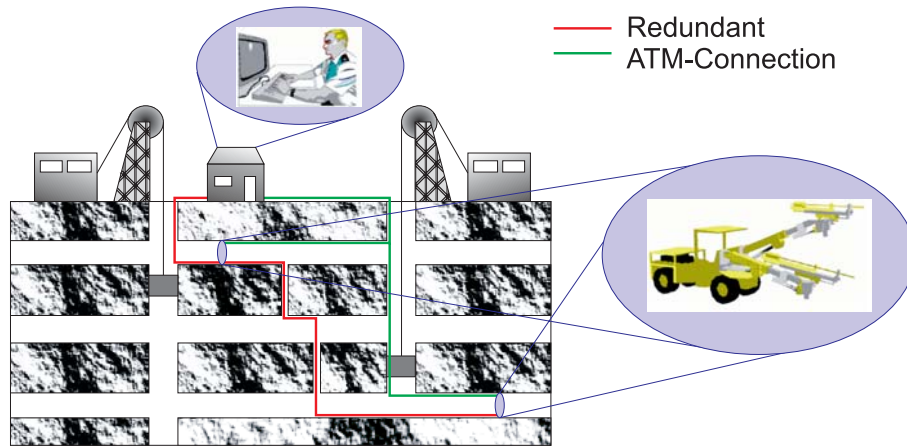


Fig. 1. Application from the mining industry

functions, but is also subject to very high safety and reliability request. These result from the dynamics and size of the device as well as the possible stay of humans in the work area. Data flows cover data with different characteristics. Therefore different communication protocols and architectures are to use. In the following two fault-tolerant architectures are introduced, which can be used for the data communication in different levels of the data processing hierarchy of a large-scale manipulator.

2 Data Processing in Large-Scale Manipulators

In order to specify the requirements in detail a view of the most important processes is necessary, which run on a distributed computer system and are linked together. The operator only moves the endeffector of the robot arm, all further arm segments align themselves independently. Since the manipulator arm has several degrees of freedom in the motion, it can move in such a way around obstacles. This requires a complex calculation of all joint angles [8]. Besides, mechanical components must be held on the debit position by an intelligent controller. On the basis of a layer model the necessary processes are structured from the point of view of the communication system.

2.1 Hierarchical Model of Data Processing

In the automation engineering communication units are roughly classified [3] by a hierarchical model. Fig. 3 shows the model of the data processing in large-scale manipulator systems, which makes a classification according to the control tasks of a layer and the data processing components.

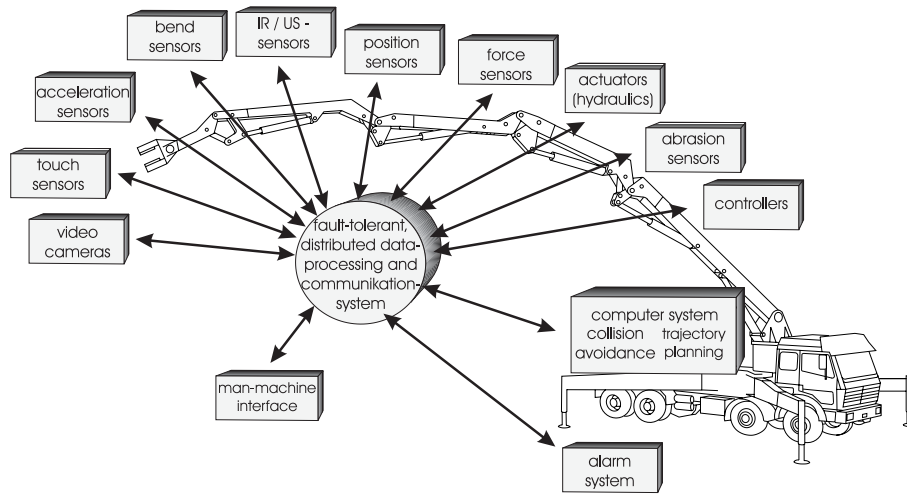


Fig. 2. Prototype of a large-scale manipulator

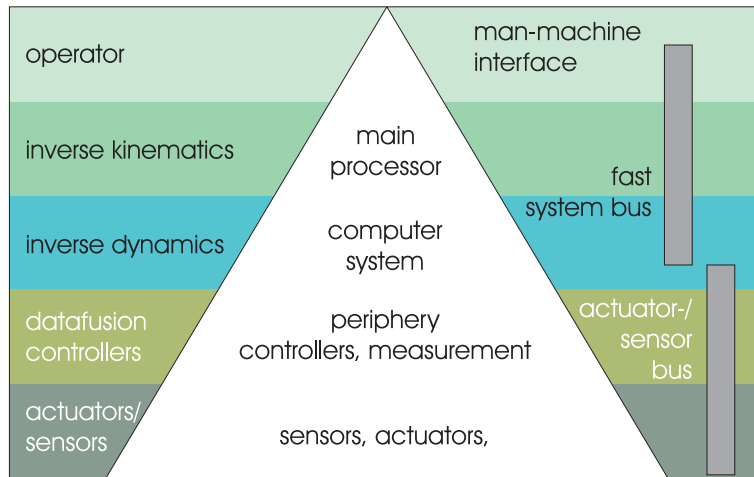


Fig. 3. Hierarchical model of the data processing

- Operator’s layer
The operator controls and observes the working process of the large-scale manipulator. The core function is the movement of the endeffector. If the control station is far away, it is called telerobotics. A possible scenario was already described in chapter one. The user needs a very large quantity of data to control the system (video, sound, sensor and control data).
- Inverse kinematics
Processes of this layer receive the coordinates of the endeffector and determine the appropriate joint angles of all arm segments. Also information of the sensor layer is transferred to this point. Major task is the on-line trajectory planning including the collision avoidance.
- Inverse dynamics
This layer executes processes, which have been invoked by the inverse kinematics. Major tasks are calculations for the inverse dynamics. The torque is calculated in order to transfer the joint into the desired angle position. Therefore data of all layers are necessary.
- Datafusion, controllers
In this layer the controlling of hardware components takes place. Joints are driven into certain positions by force control. The control processes obtain their reference input from the inverse dynamics and are linked over field busses.
- Sensor/actuator layer
Here, the collection and preprocessing of sensor information as well as the controlling of the actuators takes place.

2.2 Communication Requirements

Regarding the data processing requirements, the communication system has to provide:

- Real-time behaviour (sufficient bandwidth, determinism of data transfer, i.e. worst case delays have to be guaranteed);
- High safety (fail-safe behaviour);
- High reliability.

According to the different layers of data processing there is a great number of data types, distinguishing from the following characteristics:

- Data length;
- Frequency of data occurrence;
- Kind of data occurrence: event-driven (spontaneous) time-driven (cyclic, periodic); Performance requirements;
- Determinism requirements (e.g. maximum jitter);
- Safety-related;
- Kind of addressing: uni-, multi- or broadcast.

Regarding data lengths and real-time requirements, there are two major categories of data:

1. Large Quantities of data, comprising large message lengths, to be transferred under medium real-time constraints. These kind of data arise in high layers of data processing (e.g. telerobotics).
2. Short messages occurring with high frequency, to be transferred with high determinism. Data of the sensor/actuator layer belong to this category.

In general, communication systems like Ethernet (TCP/IP), FDDI and ATM are utilized for the transfer of data of the first category. But, regarding the high safety and reliability requirements of large-scale manipulators, these protocols are not suitable without fault tolerance enhancements. Standardized communication systems belonging to the fieldbus or sensor/actuator domain may meet the real-time constraints of the low level data transfer of the second category, but suffer from limited fault tolerance capabilities, too. In the following, fault-tolerant communication architectures, on one hand suitable for sensor/actuator communication and on the other hand adapted for telerobotics, are introduced. As far as possible, standardized protocols and components are used, enhanced by software (protocol) and hardware features in order to improve the safety and reliability of data transfer.

3 Sensor/Actuator Communication

On the lowest level of the control hierarchy, the sensor/actuator level, data is characterized by information occurring predominantly cyclically. The transfer of these periodic values, controlling the process, are subject to high real-time constraints i.e. defaulted time limits must be kept with high accuracy. By the pre-determination of cyclical data occurrence, both the communication protocol and the application software can use fixed processing schedules which guarantee a high determinism. In addition to information occurring periodically, a bus system must provide transmission capacity for acyclic event-oriented data. These messages, such as emergency signals of the process, error messages concerning the communication system components or acyclic control instructions, are subject to high temporal constraints, too. For example, an error message has to be transmitted very rapidly in order to start fault tolerance measures without effecting the process seriously. Overcoming both periodic and event-oriented data transfer under high realtime, safety and reliability constraints, depends on the kind of the chosen communication protocol, in particular the bus access strategy. Cyclical bus access mechanisms like token-, polling- or time slice mechanisms are tailored to the deterministic transfer of cyclically occurring data. The transfer of event-oriented data requires reserving bus capacity which remains unused in the case of not occurring events. As a result, the bus load increases leading to an increase in the medium bus access delay, especially regarding networks, which comprise a high number of nodes. In contrast, event-oriented (multi master, CSMA) strategies provide a high efficiency, i.e. low bus access delays, transferring event-driven data. Using a multi master protocol, each node is allowed to start the transmission of a message at any point of time, unless the bus is occupied by another transmission activated before. Due to collisions, i.e. the concurrent bus access

of two or more nodes, determinism of data transfer is worse compared to time-driven mechanisms. The complex nature of data in a large-scale manipulator, comprising event-triggered as well as time driven data, requires the combination of different protocol strategies.

In order to achieve high safety and reliability, it has to be guaranteed that data transfer continues even in the case of a component fault. Mainly, there are two types of faults:

- Global faults lead to a breakdown of the system-wide communication. This causes the crash of the overall system function. Sample global faults are: short failures of the transmission line, open failures of the transmission line (interruption of a bus wire), short failures at a bus-side output of a network node, bubbling idiot failures, i.e. a deadlock of the bus due to a permanently transmitting node.
- Local faults are limited to malfunctions of node components, leading to a separation of the erroneous node, but do not influence overall system communication seriously.

Moreover, faults can be classified as

- temporary faults, which disturb the system operation for a limited amount of time;
- permanent faults, which are not reversible.

Global faults such as a total breakdown of a transmission line only can be tolerated by the use of hardware redundancy, for example the duplication of bus lines. Redundancy has to be managed in an efficient way, comprising several stages:

1. Fault detection
2. Fault separation (to avoid impairment of the system operation by an erroneous component)
3. Fault notification (to all of the network nodes)
4. Redundancy switch-over (i.e. replacement of an erroneous component by an operational replica)
5. Recovery of a consistent system state

The primary goal of fault-tolerant system design is that in case of a fault redundancy handling takes place without any loss, corruption, and duplication of messages. In other words, data consistency has to be maintained even in the case of component faults without exceeding the timing constraints. With respect to real-time requirements, the delays of the fault handling stages have to be minimized. A very important aspect of fault-tolerant system design is fault detection. Faults have to be detected with a high probability, i.e. with a high error coverage, and within a minimal amount of time, i.e. with a minimal error latency. In the following, a concept for a fault-tolerant system is introduced, comprising enhanced fault tolerance features and redundant system structures. The system is based on the event-triggered standardized communication protocol CAN.

3.1 A Fault-Tolerant System Concept based on CAN

3.2 The Fault-tolerant System Architecture

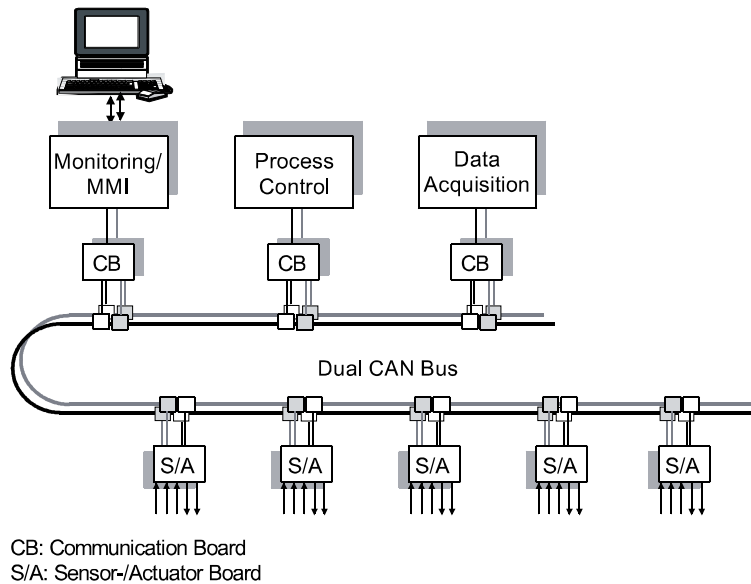


Fig. 4. Sample sensor/actuator communication system

As to be seen in Fig. 4, a distributed communication system of a manipulator using the proposed fault tolerance concept [5] includes several functional units such as process control, MMI (Man Machine Interface), monitoring and data acquisition, as well as sensor and actuator components. Each unit is attached to a dual CAN bus by the use of a communication board or a sensor/actuator board, respectively. A board, also known as node, provides two fully redundant bus links, each link comprising a micro-controller, a CAN communication controller, and a transceiver, as is illustrated in Fig. 5. The micro-controller initiates transmissions of data, selects received messages and passes them on for processing. In addition, the enhanced mechanisms for redundancy management and fault control are executed by the micro-controller. The CAN controller controls all of the mechanisms specified in the CAN protocol [1]. Finally, the transceiver carries out the physical connection of the node to the transmission medium.

Basics of the CAN Protocol. CAN, which was originally developed for use as a sensor/actuator bus in a motor vehicle, is suitable for wide areas of automation technology. Above all, the realization of a highly reliable and effective atomic multicast transmission principle and the mechanisms for fault detection

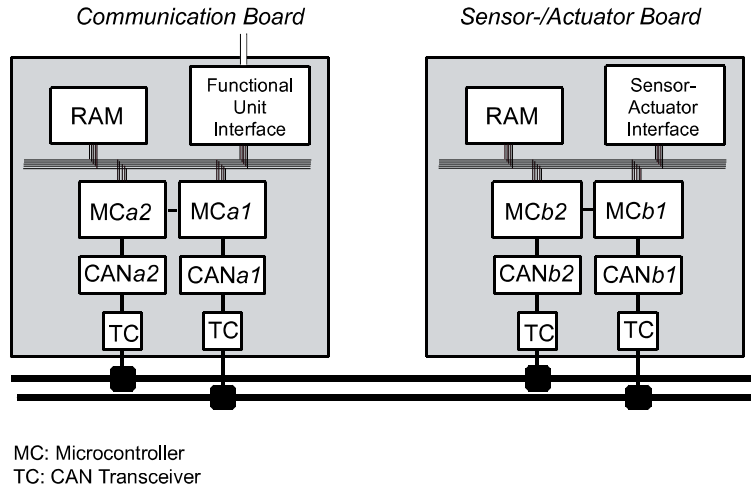


Fig. 5. Redundant communication nodes

make CAN a basis for highly reliable real-time systems. CAN networks are multi-master systems, i.e. bus access for every node is allowed if the bus is idle, also referred to as CSMA/CR or Carrier Sense, Multiple Access, with Collision Resolution. In the case of a collision, the message of highest priority prevails and may continue transmission. Therefore, a priority is assigned to every message.

Thus, worst case access times for all messages can be calculated during system design. The maximum data rate for a CAN system is specified at 1 Mbit/s. The transfer of data in a CAN network takes place as follows. A node broadcasts a message to all other nodes. If any node discovers a transmission fault, it destroys the message while the transmission is still taking place by overwriting the bus level with an error frame. As a result, all of the nodes discard the erroneous message and the transmitter starts a new transmission. This corresponds to the principle of atomic multicast. A negative confirmation mechanism is realized, i.e. in a fault-free case, a transaction requires the transmission of only one message. CAN has built-in mechanisms for error detection and localization. Defective nodes are switched off the bus, implementing a fail-safe behavior. For this purpose, each node maintains an internal error counter, which is incremented after a transmission fault has been detected and decremented after each fault-free transmission. If an error counter reaches the value 127, the node automatically goes from the error active into the error passive state, i.e. it may continue to transmit and receive messages but cannot destroy any faulty message by transmitting an error frame. A CAN controller having an error counter which reaches the value 256 switches into the bus off state, i.e. it no longer participates in the bus traffic in any way. That means, a faulty node disrupts the bus traffic by transmitting error frames until it goes into the passive state.

Enhanced Fault Tolerance Management. Since CAN is designed for use in single bus systems it has a number of disadvantages with respect to the requirements explained relating to fault tolerance. The limitations of CAN are:

- The management of redundancy is not provided for in the CAN protocol.
- Concerning node component faults, high error latencies may occur.

The redundant realization of system components, in particular of the bus line, requires additional mechanisms for the consistent switch-over to the replicated components in the case of a fault without exceeding real-time constraints. In addition, the negative confirmation mechanism of CAN causes high error latencies: the transmitter of a message does not detect the failure of another network node but rather assumes that if error frames do not occur all of the receivers have received its message without faults. Thus, node losses cannot be detected, especially in a sufficient period of time. The proposed protocol enhancements eliminate this through the use of a fault-active mechanism comprising system monitoring and error notification methods. The fault-active mechanism allows each communication link to be monitored by the other link of its node. For this purpose each micro-controller serves as a watchdog processor for the other link. In addition, in case of a component fault a node is able to become active autonomously, i.e. it informs all network nodes about the failure by transmitting an error notification message through the operational link and communication channel. Thus, the second bus system fulfills the function of a watchdog bus. During normal operation the entire process data traffic is executed through one communication channel. A sample fault reaction process started after the occurrence of a failure of CAN controller a1 (Fig. 5), takes place as follows: CAN controller a1 disrupts all network traffic on bus 1 until its error counter reaches 128; reaching the error counter value 96 CAN controller a1 transmits an error interrupt to micro-controller a1; micro-controller a1 informs micro-controller a2 of the loss of the CAN controller; micro-controller a2 starts the transmission of an error notification message through bus 2; all of the network nodes receive the error message through the links 2; all nodes switch off bus 1 and continue the transmission of process data through bus 2. The advantage of this method lies in the fact that the fault tolerance process is executed while the faulty CAN controller is in the active error state (error counter 127). This controller therefore continuously destroys the message which is detected as being faulty up to the switch-over process. As a result, no message is lost and no faulty message is processed until the bus switch-over has finished. Concerning the recovery of a consistent system state after the occurrence of faults, an important aspect is the maximum number of messages, which are lost while the fault tolerance measures take place. Message losses may occur because the faulty node is not able to receive any message until it is replaced by its redundant component. Lost messages have to be retransmitted through the use of time-consuming recovery processes. Regarding the proposed concept, in case of a CAN controller fault no message loss occurs. Regarding specific malfunctions of a micro-controller, it may not be possible to entirely exclude a message loss. This situation can occur, for example, if the fault latency is greater than the duration of the transmission

of a message. Nevertheless, only a few messages are to be retransmitted causing a minimal recovery effort. Thus, the preconditions of maintaining the consistency of data in the case of a fault are fulfilled.

4 Fault-Tolerant Data Transmission in the Telerobotics

For a better understanding the following exposition is based on the mining scenario introduced in chapter two. Main topic is the communication link between the control station over-meet and the large-scale manipulator working down in the tunnel. To fulfil the requirements for safety and reliability a redundant realisation is necessary. The communication links don't only have to be implemented dual, but they must also run on spatial separate paths. Otherwise a collapse in a single part of the tunnel could lead to a complete damage of both links. A possibility to communicate via an ATM network enhanced concerning safety is presented.

4.1 The Model of an ATM Communication

ATM connects the following components:

- the man-machine-interface,
- the video cameras,
- the sensors for the acquisition of environment data e.g. ultra-sonic sensors and
- the distributed computer system. Parallel bus systems like VME or
- PCI are not appropriate because such a tight coupling does not avoid
- error propagation.

4.2 Reasons for the Implementation of ATM

ATM is known to be a high performance communication system for telecommunication and multi-media. The following reasons show why ATM is suggestive in this case.

- In the telerobotics mainly multi-media information is present. Most important sensors are the video cameras, which have to transfer a realistic picture of the robot environment in real-time.
- ATM is not only implemented in wide area networks (WAN). The employment as a computer-computer communication system has been proved. The Washington University in St. Louis has formed the term DAN (desk area network) and has developed a micro chip which supports the communication in the local computer environment (APIC, ATM Port Interconnect Controller)[2].
- ATM is a real-time system. The quality-of-service-protocols guaranty properties of the transmission (maximum cell transfer delay - CTD, cell delay variation - CDV).

Yet ATM cannot fulfil the high safety requirements. Therefore enhancements for redundant communication paths are introduced.

4.3 Safety Enhancements

Figure 6 shows two ATM nodes which are linked via an ATM network. The layer model of ATM is not derived from the ISO/OSI model ([4], [6], [7]). The ATM network consists of many ATM switches. As an example the figure shows the functional structure of one. Which enhancements have to be made to improve ATM concerning safety requirements?

- The signaling protocol has to be extended. On demand two redundant, disjunctive routes have to be found. The maximum difference of the transmission delay has to be transferred to the ATM layer.
- The ATM layer has to evaluate two streams of data. It has to buffer the incoming cells until the redundant cells have been received in order to compare them. The sender also has to handle a buffer in order to enable a backward recovery.

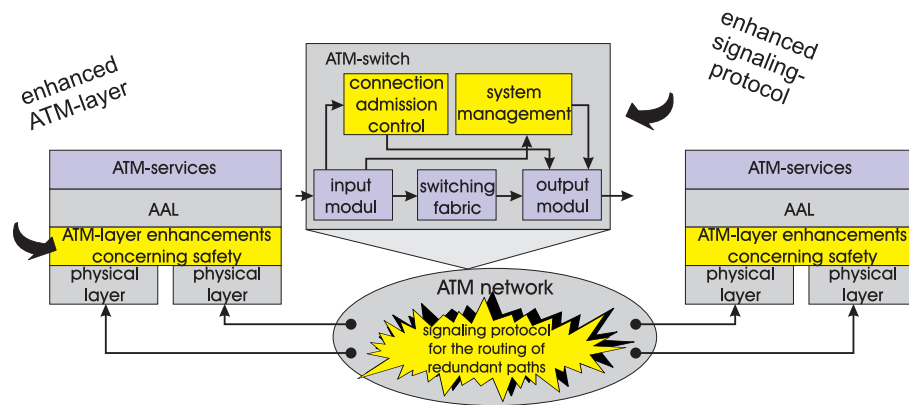


Fig. 6. Enhancements of ATM concerning the safety

4.4 Fault Detection and Handling

The process of error handling is roughly shown in Fig. 7. The upper boxes present four important possibilities of faults. The lower ones show four ways to handle the faulty state.

If two incoming ATM cells are different, the error handling depends on the priority of the information they carry. For this reason the cell loss priority flag (CLP) in the ATM-header is used. If CLP equals 1 the information has low priority e.g. video or audio data. The faulty information can be interpolated by the system (forward recovery). In the case of high priority data a backward recovery has to be made. The receiver rejects all incoming cells beginning from this point of time and demands a repetition. Therefore the sender copies all

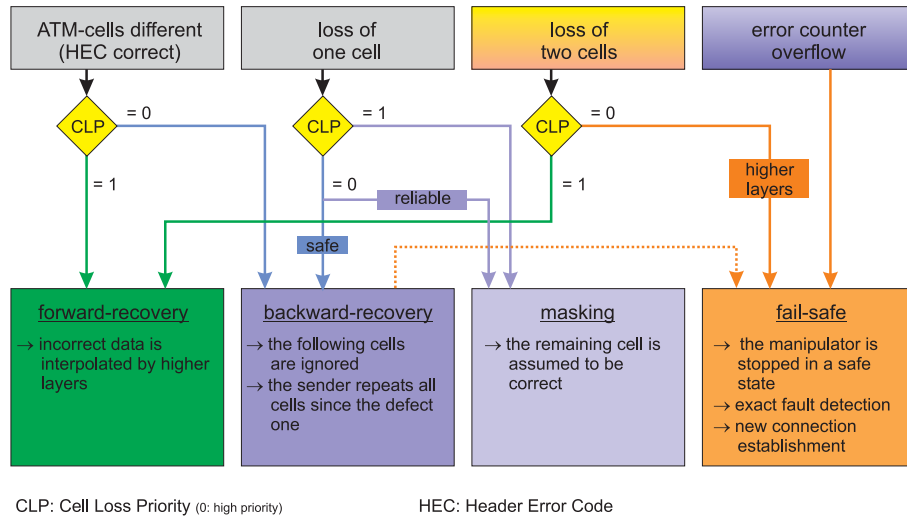


Fig. 7. Fault detection and handling in the ATM layer

outgoing cells into a ringbuffer. If the backward recovery fails, the operation has to be stopped and the system must be transferred into a safe state. If a running cell exceeds the maximum difference of the transmission delay, it is lost. Again the priority determines the way to handle this error. In the case of low priority information the system assumes the remaining cell to be correct (masking). Even if the data is faulty, it has no negative effect on the operation. If the cell has high priority, the way to handle this state depends on the consideration whether safety or reliability dominates. If on the one hand reliability is more important, you can mask the fault because the occurrence of two faulty cells is very improbable. On the other hand a safe system has to invoke the backward recovery to avoid any risk of a faulty state. The case of two lost cells is very unlikely, but has the most dangerous effects. It is very difficult to detect this kind of fault. If the cells have high priority only higher layers can avoid catastrophic consequences. It has to be analysed, if a single-error assumption is maintainable or if employments have to be implemented to detect it. In all error cases a counter is incremented. If this counter exceeds a defined number in a period of time, the system has to be transferred into a safe state. It must run an exact diagnostic program.

5 Conclusion and Future Work

Conventional communication systems do not meet the high requirements concerning safety/reliability and real-time of large-scale redundant manipulators. Therefore standardized protocols have been enhanced by extensive fault tolerance measures. The proposed CAN-based system is suitable for data transfer in the sensor/actuator layer. The requirements of higher communication layers

can be satisfied by the extended ATM system. Within the scope of the research project SFB 291 these concepts will be applied to real prototypes. In addition an ATM simulator is being developed. A virtual ATM network can be built, in which different connections are simulated and errors injected. It is based on a distributed system concept.

References

1. CAN Bus Specification 2.0. (12nc 9398 706 64011). Philips Semiconductors, Hamburg (1994)
2. Dittia Z.-D., Zubin D., Cox, J.-R., Parulkar, G.-M.: Washington University's Gigabit ATM Desk Area Network. (1994) 315-333
3. Eilmes, C.: Feldbus ja, aber welcher? Messtech (1996)
4. Händel, R., Huber, M.-N., Schröder, S.: ATM Networks Concepts, Protocols, Applications. Addison-Wesley, Cambridge (1994)
5. Hilmer, H., Kochs, H.-D., Dittmar, E.: A Fault-Tolerant Architecture for Large-Scale Distributed Control Systems. In: Proceedings of the 14th IFAC Workshop of Distributed Computer Control Systems (DCCS'97), Seoul (1997)
6. Onvural, R.-O.: Asynchronous Transfer Mode Networks. Artech House, Norwood (1995)
7. De Prycker, M.: Asynchronous Transfer Mode. Prentice Hall International, Hemel Hempstead (1995)
8. Schneider, M., Hiller, M.: Nonlinear Motion Control of Hydraulically Driven Large Redundant Manipulators. In: Proceedings of the IFAC-Workshop Motion Control, Munich (1995)