# FAULT TOLERANT DESIGN OF A REMOTE PCM SWITCHING SYSTEM*

Cs. Csapodi**, L. T. Kóczy and P. Seres**

Department of Communication Electronics,
Technical University, H-1521 Budapest

Received: September 23, 1983

Presented by Prof. Dr. S. Csibi

## Summary

The paper summarizes the principles of the fault tolerant control of the remote PCM switching system, PRS. After a brief introduction of the system, we deal with the standby strategies in the two main functional units (PRC and PRT) each of them having their own control system, mentioning also the fault detection methods. The last part treats the most important components of the diagnostic software.

## General System Description

The purpose of the microprocessor controlled PCM remote switching system (PRS) is to extend local crossbar telephone exchanges type ARF 102, requiring a small installation space. Applying this system, a subscriber stage of 1000 lines may be remoted in an ARF type office. The PRS is connected to the junction circuit groups SR of the crossbar exchange to handling originate calls and to the output terminals of the second group selector stage GV—II for terminating calls [1].

The whole system is built up of three types of equipment as follows:

— subscriber terminal PRT,
— switching and controlling equipment PRC and
— ARF exchange interface PRA.

The interconnections between these types of equipment and the layout of the PCM remote switching system is shown in Fig. 1. PRTs are installed near the subscribers, but PRAs are operated in the building of the ARF-type crossbar-exchange. PRC may be installed in a remote place and may also be operated in the same building together with PRAs.

* Paper submitted to the μP '83 symposium, preliminary report

** Research-Institute for Telecommunications, Budapest

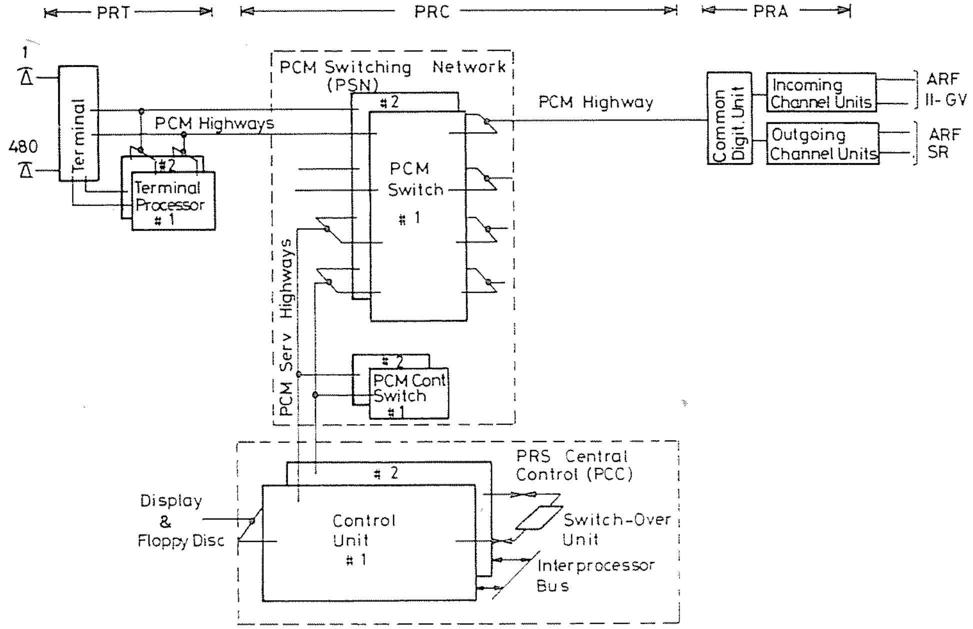

*Fig. 1.* PRS layout

The central control of the system (PCC) with stored program and the PCM switching system (PSN) for switching also conference calls are located in PRC. PRT and PRA are remotely controlled by PCC through signalling channels of the PCM highways [1]. PRTs operate as intelligent terminals containing controller processors, while PRAs are not provided with any intelligence. These latter ones execute the instructions received through their signalling channel without any decision making and directly transmit through the same channel the signals of the crossbar exchange.

All the hardware units of PRS being common for at least 64 subscriber lines are operated with standby, but the standby structure is not the same type in different hardware units. A system equipped with adequate standby units is able to operate correctly with a single error existing for a long time in its hardware, while the traffic handling capacity of the system could be deteriorated if some given types of error occur.

Designing the fault tolerant structure and the reconfiguration strategy of the system, the following principle was taken into consideration: the most important requirement for the operation of a telephone exchange is not to switch through the calls without errors, but to maintain the operability of the exchange every time. This principle involves even an acceptance of faulty interconnections in the case of transient errors occurring and having recorded the faults the system is permitted to continue operating in the same configuration without any intervention of the maintenance system. This is possible, because even in a switching system operating without any hardware errors, 15 percent of the calls are unsuccessful due to the busy line of subscribers that had been dialed and the number of misconnections due to these transient errors of hardware operations will be considerably less, hence practically negligible. An intervention of the maintenance system is required if the error of hardware operations may be considered as a permanent fault and if it is probably impossible to maintain the operability of the system without any reconfiguration.

The procedures and hardware elements required for the fault detection and analysis as well as for reconfiguration of a faulty system are designed to be as simple as possible so as to avoid the procedures and hardware units to be more complicated than those under supervision and reconfiguration. It was supposed that the system would operate correctly for a long time and to operate any standby units would be necessary seldom only.

## Fault detection and recovery strategies in PRG

The fault detection elements indicate any defects occurring in the PCM switch or in any functional subunits of the control unit (CU). Continuous functioning does not necessarily mean a faultless operation all the time. The system should registrate any errors but its reaction depends upon the circumstances whether they can be considered as transient or permanent ones and whether they affect all the connected subscribers (about 1000) or only a small group of them.

In the case of a transient fault affecting, for instance, only a single call-process and between such faults the system operates without malfunction for a longer time, it is not necessary to initiate any error recovery action but to make a record of it for later analysis. Should a permanent fault occur the scale of reaction is to be considered to avoid over-reaction.

There can be faults, such as e.g. power failures, which occur very rarely but may cause serious problems for a longer time. Subunits that can cause such acute failures (e.g. power units, clock generators, CPU's) are duplicated and continuously monitored for immediate actions. Less serious problems, however, require special strategies.

### *Fault detection in the PCM switch*

The possible errors, can be divided into two groups:

— PCM line faults and

— PCM switch faults.

Abnormal functioning in PCM lines are checked by special hardware means resulting in interrupt signalling towards the central controller. The cause of the error is determined by the CPU polling the line *status bytes* sequentially. Once the faulty PCM line has been found it is eliminated and the system resumes normal operation with decreased throughput giving an alarm signal at the same time. Abnormal functioning in the PCM switch is detected by a combined hardware and software technique. The PCM switch sends back the executed switching command to the CPU for validity checking. It is also possible to loop back TEST bytes sent by the CPU accross the PCM switch for testing any possible interconnections.

### *Fault-detection in the Central Controller*

The Central Controller PCC of PRS consists of two independent CUs running in a synchronous (dual) mode of operation executing the same instructions under normal working conditions and a switchover unit based on a single-chip processor.

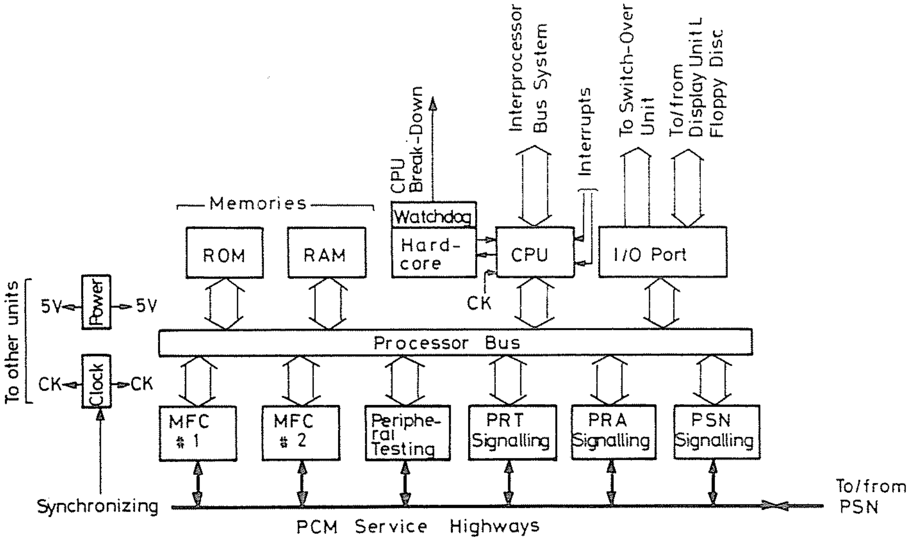

*Fig. 2.* Hardware structure of a Control Unit in PCC

Each CU has 6 peripherals working to and from a PCM service bus and controlled by the central microprocessor (CPU) via the processor bus (Fig. 2). The CPU contains selftesting facilities and it can test the peripherals as well. Each control unit has its own power-unit and a triplicated clock generator synchronized by the master. The CUs are connected to the switch-over unit by means of input/output ports.

During normal operation one of the CUs is the master and the other is the slave. Both CUs receive the same messages arriving at the 1st and 2nd service bus, respectively. The messages arriving from the PRT and PRA are processed independently and synchronously in both CUs working out the required commands for the PCM switch, but only those coming from the master are to be executed. During synchronous operation the CPU's and the peripherals are executing the same operation in every clock cycle but the dual system works as if only the master were working. The synchronous operation is checked only at the level of sending switching commands towards the PCM switch because this is the main task for the CU. If the command sent out by the slave does not comply with that of the master, this indicates an error which is signalized to the switch-over unit but the operation is not interrupted unless the switch-over unit does not initiate it. In case of a difference, the switch-over unit decides whether it was a transient or permanent fault and if it is a permanent one, which of the two CUs should be considered faulty. The necessary additional information is worked out by the switch-over unit (Fig. 3) on the basis of

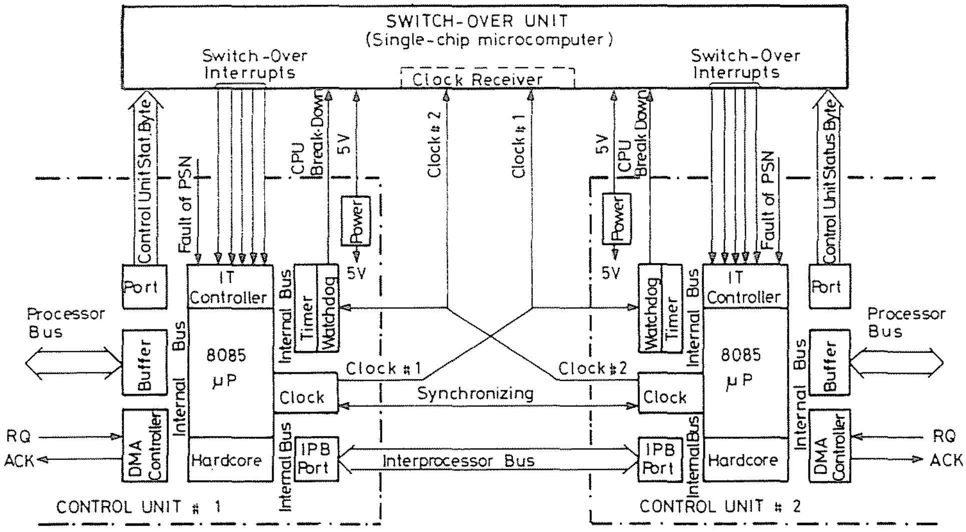

*Fig. 3.* Connecting paths between Control Units and Switch-Over Unit

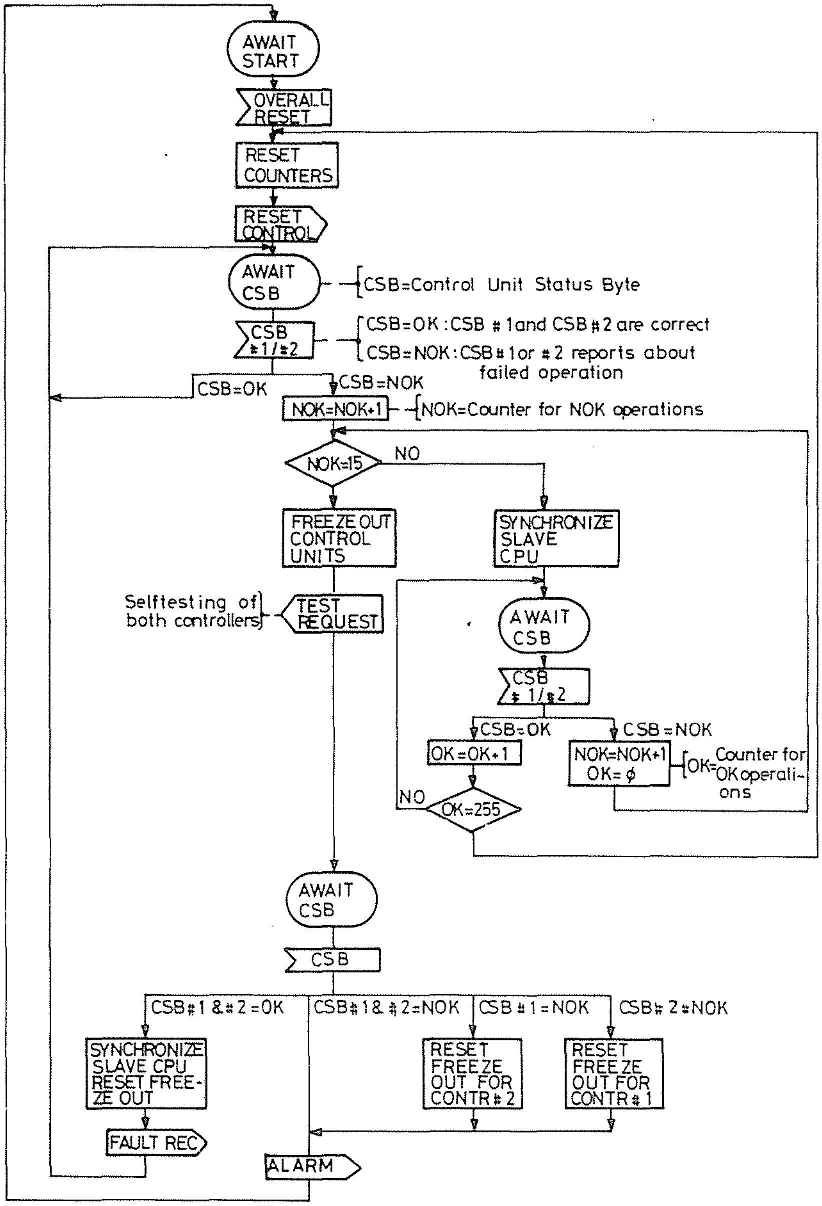
AWAIT START
OVERALL RESET
RESET COUNTERS
RESET CONTROL
AWAIT CSB
CSB=Control Unit Status Byte
CSB #1/#2
CSB=OK : CSB # 1and CSB # 2 are correct
CSB=NOK : CSB # 1or # 2 reports about failed operation
CSB=OK
CSB=NOK
NOK=NOK+1
NOK=Counter for NOK operations
NOK=15
NO
FREEZE OUT CONTROL UNITS
SYNCHRONIZE SLAVE CPU
Selftesting of both controllers
TEST REQUEST
AWAIT CSB
CSB # 1/#2
CSB=OK
CSB=NOK
OK=OK+1
NOK=NOK+1 OK=ϕ
OK=Counter for OK operations
NO
OK=255
AWAIT CSB
CSB
CSB#1 &#2=OK
CSB#1& #2=NOK
CSB #1=NOK
CSB# 2=NOK
SYNCHRONIZE SLAVE CPU RESET FREEZE OUT
RESET FREEZE OUT FOR CONTR # 2
RESET FREEZE OUT FOR CONTR # 1
FAULT REC
ALARM

*Fig. 4.* SDL diagram of simplified switch-over algorithm

accumulated knowledge about previous errors. A difference during comparison does not necessarily mean a fault in one of the functional units. After detecting a number of synchronization errors the first thing is to try to set up again the synchronous mode of operation i.e. the master updates the memory content of the slave and they try to run again synchronously. Should this attempt fail, the switch-over unit initiates a set of test procedures for localizing the fault, starting from the self test of the microprocessors. The results are reported to the switch-over unit which works out the switch-over command, if necessary.

In case of a definite failure such as a watchdog interrupt caused by the master's malfunctioning or by missing clock-signals, an immediate switch-over to the slave is initiated. The above mentioned decision making processes are outlined in the flow-chart (Fig. 4).

Fault-detection is also based on extensive software checks intended to control the normal handshake signals during call-congestions. The synchronous operation is checked by comparing the commands sent by the master and the slave for controlling the PCM switch.

### *Checking the call-processes*

The call-processes contain a number of different *states*, input-output messages, commands and decisions. They can get from one state to another by software subroutines initiated by the input messages. Because of some malfunctioning the necessary subroutine is not called in or it runs in an endless cycle, the call-process sends a time out signal to the maintenance system which can bring the affected call-process to an end making a record of the error at the same time.

### *Checking the synchronous mode of operation*

The structure of the PCM switch is such that after having executed the commands received by the master it sends back the executed commands to the master and the slave for checking purposes. On the other hand it sends back to the master the command worked out by the slave as well. In this way both CUs can cross-check each other, at the same time they can obtain information about the correct operation of the PCM switch. All this data is accessible by the CPUs in their own RAM memory and can be tested regularly. The result of this software comparison is transmitted to the switch-over unit in the form of a status byte from both CUs.

The above comparisons only signal about the presence of an error but they cannot indicate the faulty unit. Therefore whenever the switch-over unit decides on a permanent fault, it suspends the synchronous operation and gives command for the self-tests which can locate the faulty unit with a high probability.

## Fault handling in the terminal controller

For the terminal units (PRT) the controller configuration is in principle similar to that for PRC. There are two controller microprocessors (type I 8085) with separate memories, watchdogs etc. with entirely identical configurations. Each microprocessor (μPA and μPB in Fig. 5) has its own bus system, which are connected by a bidirectional 2×1 byte buffer, in order to provide the processors with the possibility of exchanging information. Filling the buffer belonging to one direction must always be followed by an interrupt request so the other processor turns to the buffer. These IT lines, however, can be masked in order to be protected against faulty interrupt requests from the side of a processor out of normal operation state. To each processor a peripheral driver unit belongs through which the telephone peripherals are controlled. At the same time only one such unit is allowed to forward valid commands. There are two service receivers (transmitter units, in the case of normal operation, both of them transmit the same messages sent by the executing processor), but they forward the messages received from PRC separately to the two processors (1 to A and 2 to B). PRC gets all information concerning the state of the processors through these units.

The cooperation of the two processors is based on a "partial hot stand by" philosophy. This means that both processors are permanently active (except a defection), but one of them is taking the role of the master, the other is working in a slave mode. The slave's memory contains only a part of the information stored in the master. It is expected that switch overs from the master processor to the other will be very seldom (except if intentionally, with no fault event), this is why it does not seem to be important to save calls in a transient state (as e.g. call initiation, dialing) as these will make only a small percentage of the total traffic. So if all calls but those in speach state get lost, no serious disturbance in the function of the exchange can be detected by the subscribers, i.e. only the information concerning calls getting into or out of speach state is transferred to the slave processor in PRT. This information modifies as a matter of course, on one hand the scan table, on the other it generates a process table artificially that is forced directly into the needed state. As the slave has no real contact with any peripherals and all time out processes connected with call handling are masked,

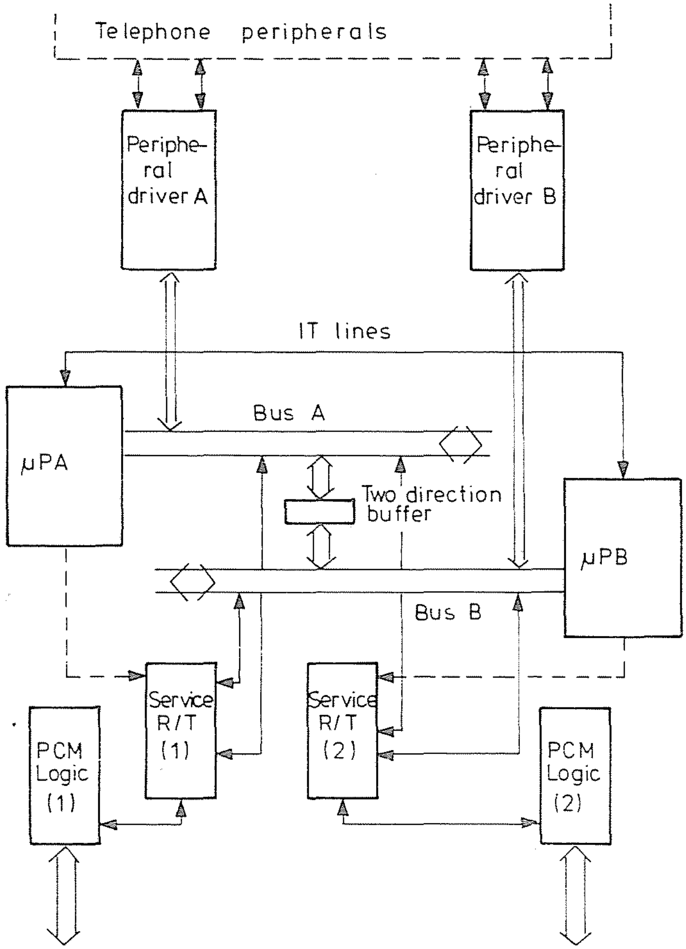

*Fig. 5.* Hardware configuration of PRT Controller

such processes will never be activated (unless the slave takes the master role). This solution provides a very small loss in the case of a defect while the load on the master caused by slave messages is negligible. In the defect situation the lost calls will be cleaned out of the system by the time out processes in PRC.

As has been stated, complete updating of the slave is not necessary. This does not mean, however, that the slave needs not be tested with an even higher intensity than the master. The results of all test processes are periodically transmitted to the master, which takes care of their forwarding to PRC. Also the master is tested (for test aspects see the next section) and the slave evaluates these results. In the case of a faulty or failing test result, the processors send a message to PRC and, parallel, the faulty one takes the role of the slave and starts some special tests. If this did not happen, PRC forces the switch over through a simple circuit attached to the controller board.

Finally, it is to be mentioned that at times with no traffic but not more frequently than once a day a switch over without defect and loss is completed.

## Diagnostic software components

In this section, an overview will be given, concerning the most important software blocks contained in the PRS system without any division according to the hardware units, for diagnostic software has been designed being potentially uniform in both PRC and PRT.

In general, fault detecting and diagnostic software in PRS consists of two areas: tests of the control system (microprocessors and additional circuitry as e.g. watchdogs) and of the peripherals (including also the communication channels). As the latter is much more determined by the hardware design of the specific switching equipment and as it is not typical for a usual microprocessor controlled communication system, only the test of the control system will be dealt with.

The test philosophy is based on the following considerations. In a complex system, there must be dozens or even hundreds of fault types. These fault types, however, never appear in a pure form, they rather cause different secondary effects which often have not got an unambiguous connection with the original fault. By test programs, as a matter of fact, only some effects can be detected, the first time only one or a few of all belonging to the fault situation. So, if the software detects any fault effect, immediately the most thorough possible examination of the concerned system part must be realized. When having a complete image of the state of the system the assumed fault types are ordered into a decreasing probability sequence. This principle is much more difficult to follow if two or more faults are allowed to exist parallel to each other, in the fault detection strategy used in PRS, it is supposed however, that all defects will be diagnosticized in a relatively short time after their occurrence, as the probability of another fault appearing before completing the diagnosis is very small.

In the former section, the watchdogs belonging to each processor have been mentioned. The operational system of each processor is due to send life signs permanently to the watchdog. The allowed time interval between two such signs has a certain tolerance, but it is limited from both directions; too early or too late pulses effect in a hardware alarm from the side of the watchdog (see Fig. 6), i.e. in the case of a sign of life missing the window of the watchdog requests an interrupt. As a direct result of a short self-test procedure begins inside the board containing the controllers. This check involves a check of the important control signals and the PROM-s as well as a simple functional test of the RAM. If no fault has been determined a guess is made concerning the faulty

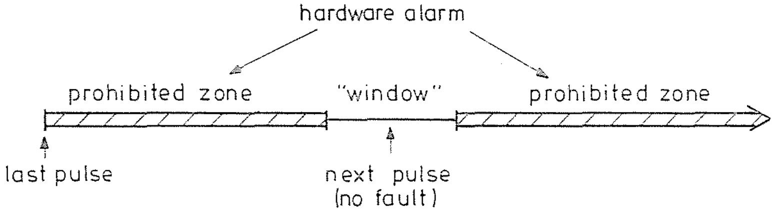

*Fig. 6.* Time diagram of a double watchdog

program or data area to which the register contents are used. According to the software structure of the system, the following characteristics are tested:

— all PROM-s (by control sums),
— all RAM-s,
— the stack pointer (validity check),
— all data connected with the area (validity),
— clear areas or words (contaminating data check),
— pointers (forward and backward) of all chained records referred to in the area.

If no faulty component has been detected, the fault will be handled, if possible at all, as a transient defect. If, however, a positive result has been gained, the contaminated area will be tried to be cleaned by eliminating all connected processes. This means, on one hand, some clearings in the RAM and, on the other hand, the generation of certain messages. If elimination is impossible a direct message to operator will be sent out (as e.g. asking for the exchange of a board).

Transient faults mentioned earlier will be registered in software counters (each belonging to one type of fault and one board). Should a defect occur that cannot be reproduced, the corresponding counter is incremented. In order to eliminate the long time registration of transient faults caused by some occasional outside effects the counters are decremented if a long time has passed without an incrementation (except counters having zero value). If the counter, however, reaches a critical point, the fault will be considered as a permanent one and the fault message will be sent out accordingly to it. In the case of certain fault types a middle value is set to where no outside fault message belongs to but test intensity will be enlarged.

There are also routine test processes which are activated only seldom by the scheduler if there is no special reason but which get a higher priority if e.g. a counter has reached its middle critical value, or there is any hint to some hidden fault. Some examples for such processes are complete pointer checks in chained records, complete validity and contaminating data checks, control byte checks in different tables, stack pointer checks and complete RAM tests.

Among these routine test processes a central part is played by the watchdog test which is carried out with a masked watchdog alarm. This is necessary in order to prevent a double fault occurring when a defective watchdog cannot detect a processor or software fault.

Finally, one of the most important test software blocks must be mentioned: these are "dummy" call processes checking through the whole call-handling system.

As it has been pointed out at the beginning of the section, there is another part of the diagnostic software concerning the peripherals, but there is no possibility here to deal with these problems.

## Reference

1. BLUM, E., HORVÁTH, L., HUTTER, O., NÉMETH, G.: Microcomputers in a Distributed Digital Subscriber Switching System, Submitted to the μP '83 Symposium

Dr. László T. KÓCZY H-1521 Budapest
CSAPODI Csaba } Research Institute for Telecommunication
Dr. SERES Péter } H-1026 Bp. Gábor Áron u. 65