Conf-941061--1

Nuclear Science Symposium
Norfolk, Virginia
October 30 - November 5, 1994

BNL-60283

Concepts and Techniques: Active Electronics and Computers in Safety-Critical Accelerator Operation*

R.S. Frankel
Brookhaven National Laboratory
Upton, NY 11973-5000

RHIC (The RELATIVISTIC HEAVY ION COLLIDER), under construction at Brookhaven National Laboratory, requires an extensive Access Control System to protect personnel from Radiation, Oxygen Deficiency and Electrical hazards. In addition, the complicated nature of operation of the Collider as part of a complex of other Accelerators necessitates the use of active electronic measurement circuitry to ensure compliance with established Operational Safety Limits.

Solutions were devised which permit the use of modern computer and interconnection technology for Safety-Critical applications, while preserving and enhancing, tried and proven protection methods. In addition a set of Guidelines, regarding required performance for Accelerator Safety Systems and a Handbook of design criteria and rules were developed to assist future system designers and to provide a framework for internal review and regulation.

Prototypes of fundamental system elements were constructed and program fragments written, which establish that the basic design concept is sound; operation is expected for the first phase of the system in September 1995.

This paper is offered in the hope and belief that many of the techniques developed as part of the Access and Operational Safety systems designed for RHIC (and other recently completed large Research Accelerators) may be directly applicable to smaller research and clinical treatment facilities. Because of the relatively short length of this paper, I will concentrate on four topics: novel features of our system design; potentially useful applications of probability theory empowered by these system concepts: a listing of configuration control and design approaches, which when followed, will enable one to design high reliability hardware and systems (to meet required functionality); software design concepts and approaches.

We start with a basic premise: all radiation facilities must ensure the safety of their operations and maintenance personnel, experimenters and in the case of clinical facilities their patients. Classically at large research machines, this has meant an Access Control System, designed to protect personnel from Radiation hazards. Other safety hazards which are found within accelerator enclosures or support buildings were each mitigated by their own independent administrative controls or engineered safety solution. At RHIC, we are integrating these multiple safety systems into a single RHIC "Personnel Safety System", in lieu of separate design responses to our Oxygen Deficiency Hazards (ODH), Electrical Hazards and Radiation Hazards. It is expected that this approach will result in a superior level of personnel safety and equipment protection while providing greater operational efficiency.

One reason not to employ this approach is that the centralization of multiple safety systems results in an extremely complex assembly of hardware (Relay or Logic implementations) or large and complex software code (PLC and other Computer based implementations). It follows that required validation testing may grow so complex, when one makes even a minor change in a single subsystem aspect, that the revalidation of system functionality becomes a lengthy if not impossible task.

A solution used by others, with some success, is to carefully and rigidly modularize the solution elements. In the case of a Software implementation having modules protected from external changes and possessing clearly defined parameter passing is employed. In Object oriented programming terms we employ *encapsulation*, a binding together of data and code to form a self contained "black box". Other Object oriented concepts which are used with great success in industry such as *polymorphism* and *inheritance*; offer less utility to the Safety System programmer, if only because such techniques are most useful for very large programs and reusable code libraries.

The idea of using Programmable Logic Controllers as Control computers for safety applications was pioneered at CEBAF and the APS at Argonne National Laboratory. We all know from life experience that the average computer program can be expected to contain errors (bugs). Even Application programs and Operating systems with millions of users and tens of man-years of testing and quality control contain "bugs. This state is not likely to change in the foreseeable future as some computer programs are among the most complicated machines ever built by man. Anyone who has tried WINDOWS programming is aware of the problems associated with application and operating system interaction: such systems are not deterministic. The great advantage of PLC computers is that they, with only minor variation, always follow the same sequence. Data is read IN, Data is processed and Data is OUTPUTTED. There is no real multitasking, only minimal use of interrupts and DMA by the PLC operating system itself and one can avoid most use of interrupts in one's own application programs. In additional it is possible to obtain documented high levels of hardware availability, something not readily available for Personnel Computers or Work Stations. Indeed the most irritating aspect of PLC programming, the primitive development environment, acts to reinforce a programmers direct oversight of computer operation.

Therefore we will employ PLCs as the core processors for our project. A difference between RHIC and other PLC based safety installations is our use of more and smaller PLC units in lieu of a few larger PLCs connected to crates of non-intelligent I/O. We intend to interconnect our PLCs in groups (of peers) and use two completely independent divisions each

*Worked performed under the auspices of the U.S. Department of Energy.

made up of sets of peers in order to establish additional redundancy.

We intend to supplement the benefits obtained by using procedure encapsulation with a partitioning of controller tasks by assigning responsibility for the supervision of a limited geographical area to each of a pair of PLC processors, one from each peer division. Thus further limiting hardware unit complexity and software complexity and size.

There is yet another conceptual system advantage to this approach. Numerous PLC units, each being "intelligent" can act as independent intelligences or agents, thus helping to detect complex hardware failure modes while avoiding the complexity of multitasking operating systems. While we avoid the disadvantages of multitasking its advantages are retained via our use of a multiprocessing environment. In more human terms, the "little guys are aware enough to sense not only their own field wiring, but also the state of their peers, if only by lack of response to their queries. Since the amount of safety equipment in a single geographical area is minimal, the complexity of the program in each PLC will also be minimal, with short, easy to modify, and test program segments. This approach is claimed to lower installation costs and to provide more safe operating time for the same cost, when compared to systems using a few larger PLC units connected, to multiple remote I/O chassises.

Thus a network of PLC units compensates for the complex failure mechanisms exhibited by individual processors much as a Op-Amp compensates for component variability with gain and feedback or a bridge is supported by its interconnecting I-beams

The second division of isolated and redundant PLC units will be programmed by a different programmer, constructed using another microprocessor technology and coded using a compiler written by an alternate vendor development group (all to minimize unknowable Common Cause failures).

In physically smaller research or clinical facilities the partitioning might follow another pattern e.g. by function or system. The important concepts are partitioning and multiple intelligence or agents: multiprocessors not multiprocessing.

A system of this type also opens several new Operations options, compared to the classical "detect a failure - stop the facility". Because of higher redundancy levels within the safety system itself one could adopt an operating paradigm of continuing to operate even though parts of the safety system are being maintained or have failed. The notion that one can continue to operate while performing some safety system maintenance or testing is backed by an analysis of what is the chance that the reamaining on-line system elements will fail in the next hour etc. If the mean time to repair is short and repair is allowed total system availability goes up. We have not yet decided whether to actually take advantage of this feature.

All facilities need a Run/Stop feature. Ours will use "pull cord" type switches. There will be essentially continuous coverage on both sides of the enclosure tunnel. These CRASH switches are **not** hard wired into a lockout system, but are connected in a redundant manner to a PLC in each of the redundant division. When a CRASH is called for, a Beam Dump System will be activated, but not depended upon to function as a life-safety element. Instead the Personnel Safety System will remove power from selected critical power supplies or close selected critical vacuum valves. This approach is necessary because the Beam Dump system, while engineered and constructed to high standards, is not considered and tested to be part of the Personnel Safety System.

Our entry system consists of thirty five (35) Gate packages and Nineteen (18) Emergency Entrances and/or Exits Door packages. Redundant interlock switches will be mounted on each of these doors. There will be a set of captive keys interfaced into the PLC logic such that if specific keys are missing from the key bank, then the Collider will not start-up and/or continue to operate. The greatest weakness of a distributed system such as ours is that a local unit might be deceived into thinking that the facility is in a "safe" operating mode when that condition is untrue. The use of captive keys provides supplementary information to the PLC units to establish that they are operating in the correct mode and also act as physical and administrative barriers to improper entry. It is useful to implement more "modes" in a Software controlled system, then is customary in hardware systems because it is harder to validate software patches then test jumpers. The major test modes are built into the software and validated as part of pre-operation tests rather than being written as needed.

The Radiation Monitoring system will employ the "Chipmunk" design used at the AGS and at FERMILAB. Interlock outputs will be connected to system PLC units. Since the data rates from these units is capped at 500, 25 microrem pulses/second, it becomes realistic to feed these pulses directly into dedicated small PLC units and thus avoid the need for outboard physical counters.

Redundant signals are carried via separate cables except when conduit is used; safety system cables are carried either in their own tray or via conduit.

It is extremely important to define and understand what constitutes a "failure" in your context. We have established a requirement that our protective systems will not malfunction or cease to function such that the chance of an individual being exposed to severe and/or serious hazards is only 1/1000 in 10 years.

A very important factor in the design of Accelerator Radiation Safety Systems is that for the most part you **can** turn-off an accelerator. When an accelerator is "off" it can't create a prompt radiation hazard, consequently only that portion of a safety system which can impact our ability to achieve a "fail-safe" result need meet this performance standard. A corollary of this line of thought is that when an accelerator facility is "shut-down" time effectively stops; Mean-Time-To-Repair (MTTR) and related issues become moot.

On the other hand when the hazard being mitigated is, for example, a gas release, then simply shutting off the Accelerator will not in itself create a safe-state, some positive action such as activating an air exchange system must take place therefore a safe-state is achieved; this means that a greater portion of the active circuitry and/or software must be evaluated before it can be accepted as being capable of mitigating its target hazard.

For a major safety system in a complicated facility such as a particle accelerator, early life failures tend to be random because of high quality control requirements, while maintenance and replacement of components help mitigate against wear-out failures. Thus we are usually dealing with the "flat" region of a classical bath-tub shaped hazard model and we need to know the intercepts and slope for this region.

DISCLAIMER

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Unfortunately when devices fail infrequently and are sufficiently complex and costly many tests cannot be performed to characterize patterns of failure. Therefore we are left with only an estimate of $\lambda(t)$, the hazard rate for systems. The usual procedure is to assume that failures occur at a constant rate, the "floor of the bath-tub" is flat, so $\lambda(t) = 1$.

There are two cases of interest when one deals with failure probabilities. The Failure probability per demand for intermittently operated systems , such as a starter motor, is frequently termed Q_d, and the Hazard rate for systems, such as transformers, undergoing continuous use is typically termed $\lambda(t)$, which in the absence of precise data is usually approximated by a constant λ. Sometimes a physical device has both failure modes, for example, the failure of a fuse to open when challenged by an over-current maybe given a Q_d of one out of 10^5 demands, while the chance of a fuse opening at normal current is given as $\lambda = 1 \times 10^{-6}$ hours. Care should be taken in selecting the correct model and data for your application. Fortunately, as a result of military and commercial nuclear power needs, tabulations of basic components are readily available.

In order to achieve a level of performance such that no non-safe failures are expected during the active life of the facility, it is for all practical purposes, necessary to eliminate known early-life-failure effects and to mitigate, by design, calibration or maintenance, end-of-life-failure modes.

When one has substantially eliminated all understood and/or controllable failure mechanisms it may generally be assumed that subsequent failures will be rare and that there is no choice but to assume that remaining failures will tend to occur in a random manner.

When redundant subsystems fail as a result of a single event, the event is called a Common Cause failure. Obvious examples of potential Common Cause failure mechanisms are earth-quakes, fires, loss of primary power or terrorism. More within the control of the system designer is a subset of Common Cause failures known as Common Mode failures; these include design errors, operator errors and environmental effects.

Full credit for reliability improvements based on mathematical calculations derived from independent event formulas cannot be taken unless it can be shown that the subsystem elements are substantially independent for Common Cause events, a difficult point to establish. A reasonable case for the adoption of MTTFs based on formulas derived using assumptions of mathematical independence may be established if all of the following are true: the system consists of circuits or equipment which are of different designs and use different primary components; different application and system software are used; the event of interest is monitored using different technologies; the subsystems are physically and electrically isolated, and powered; redundant wiring is carried via separate cables.

When these factors cannot be established , the suggested practice is to include a multiplier for Common Cause failures which reduces the reliability figure obtained from calculations based solely on mathematical independence. The assumptions which go into an estimate of the Common Cause Hazard rate are almost always based on engineering judgment rather than data.

It is strongly recommended that the selected method or methods of determining the occurrence of a hazard condition relate closely to the nature of the hazard itself. By way of illustration when the hazard is oxygen deficiency, one should monitor for the percentage of available oxygen, not for the presence of an inert gas

One potential advantage of intelligent control systems is that for many of the events being monitored, there is the capacity for active and frequent monitoring of conditions. If for example one adds a time dynamic to a supervised line one can monitor not only whether the line was "open" or "continuous", but whether the line "responded" with a state change between normal and test modes. Multiple intelligent systems are also better at dealing with some types of Common Cause induced failures. Detection of component or signal failure should be an intrinsic part of the design; other approaches require the inclusion of additional circuitry to detect malfunction, increasing the complexity of the design, which in turn reduces reliability.

Perhaps the most useful thing I can provide you with is an index from our internal specification which is intended to ensure that all know or knowable Design, Quality Assurance, Administrative steps are taken to achieve a proper system. We intend to build a system such that only unknowable or random failures remain.

I Safety-Critical Device Design Specification

- 1.0 Scope
- 2.0 Classes & Types
- 3.0 Required Action
  - 3.1 Hazard Classes I and II
  - 3.2 Hazard Class III
  - 3.3 Hazard Class IV
  - 3.4 Hazard Class V
  - 3.5 Hazard Class VI

II Reliability Analysis Guidelines

- 1.0 Overview
  - 1.1 Application of Reliability Engineering to Accelerator Safety
  - 1.2 Quantitative Reliability; Availability, MTTF, MTTR
  - 1.3 Failure Data
- 2.0 FAILURE Analysis
  - 2.1 FAILURE Mechanisms
    - 2.1.1 Fail-Safe
    - 2.1.2 Fail-Soft
    - 2.1.3 Availability of the Facility
  - 2.2 Probability Distributions
  - 2.3 Redundancy
    - 2.3.1 Independent Events and Common Cause Failures
    - 2.3.2 When to use calculations based on Independent Event Formulas

2.3.3 How to correct for Common Cause Events Reliability Calculations
2.4 Monitoring of Events
2.5 Methods of creating redundancy
2.6 Some Operations Scenarios
2.6.1 Shut-down at once
2.6.2 Shut-down after grace period
2.6.3 Operation without Repair

III Design Handbook (Documentation, Design and Construction Practices)

1.0 System Characteristics
1.1 Power-Loss or Shutdown
1.2 Malfunction
1.3 Latching
1.4 Isolation and Grounding
1.5 Remote Locations
1.6 Supervised Lines
1.7 Redundancy
1.8 Hazard Description
1.9 Modular Design

2.0 Power Supply Characteristics
2.1 Source Power
2.1.1 Voltage and Current Levels
2.1.2 Loss of input power
2.1.3 Line surges and spikes
2.1.4 Line noise
2.1.5 UPS
2.1.6 Distribution Panels
2.2 Internal Power
2.2.1 Voltage and Current Levels
2.2.2 Loss or variation of internal power supply
2.2.3 Noise
2.2.4 Indicators
2.3 UPS or Battery Backup
2.3.1 Availability Time
2.3.2 "Switch-over" characteristics
2.3.3 Battery characteristics

3.0 Circuitry Characteristics
3.1 Data Acquisition and Signal Processing Circuits
3.1.1 Acceptable Parameter Range
3.1.2 Frequency Range/Time Domain Range
3.1.3 Asynchronous or Synchronous Signals
3.1.4 Voltage and Current Range
3.1.5 AC or DC Coupling
3.1.6 Limiting and Conditioning
3.1.7 Isolation
3.1.8 Input Signals Lost or Unacceptable
3.1.9 Auxiliary Signals Lost or Unacceptable
3.1.10 Distances between Devices and Sensors
3.2 Actuator Circuits
3.2.1 Acceptable Parameter Range
3.2.2 Steady-State vs Momentary Outputs
3.2.3 Relay Circuitry
3.2.4 Digital Circuits
3.3 Logical Control, Process Control and Status Acquisition Circuits
3.3.1 Analog Circuits
3.3.2 Logic Circuits

4.0 Mechanical and Electromechanical Aspects
4.1 Adjustments, Option Selection and Connections
4.1.1 Subassemblies are Keyed
4.1.2 Operation of Controls and Switches
4.1.3 Connectors
4.2 Boxes, Enclosures and Racks
4.3 Transducers and Sensors

5.0 Cabling
5.1 Interconnection and Power Cables
5.2 DC Cable Characteristics
5.3 Analog and Pulsed Signal Cables
5.4 AC Power Cables
5.5 Labeling

6.0 Location and Environmental Concerns
6.1 Environmental Concerns
6.1.1 Radiation
6.1.2 Temperature
6.1.3 Humidity
6.1.4 Wet Locations
6.1.5 EM Fields
6.1.6 non-ionizing radiation (RF)
6.1.7 Corrosive or explosive atmospheres
6.1.8 Cryogenic Factors
6.2 Location Concerns
6.2.1 Placed at correct site
6.2.2 Placed at protected site

7.0 Validation and Verification
7.1 Testing Concepts
7.1.1 Reviewed Written Specification
7.1.2 Formal Test Records
7.2 Calibration Requirements
7.2.1 Test Equipment
7.2.2 Calibration Intervals
7.3 Self-Test Techniques
7.4 Release for Installation

8.0 Human Factors
8.1 Standards
8.2 Alarms
8.3 Displays
8.4 Controls
8.5 Labeling

9.0 Maintenance
9.1 Preventive Maintenance
9.2 Corrective Maintenance

10.0 Configuration Control
- 10.1 Design modifications
- 10.2 Documentation
- 10.3 Design, Construction, Testing, Commissioning and Operation

11.0 Components
- 11.1 Traceable components
- 11.2 Over-Designed Components
  - 11.2.1 Passive Components
  - 11.2.2 Active Components
  - 11.2.3 Extended Range Components
  - 11.2.4 Printed Circuit Boards
- 11.3 Preferred Components
  - 11.3.1 Passive Cooling
  - 11.3.2 Circuit Integration Level
  - 11.3.3 Relays
  - 11.3.4 Lamps
  - 11.3.5 Contacts
  - 11.3.6 PROM
  - 11.3.7 Capacitors

12.0 Programming Methodology
- 12.1 "Software does Fail"
- 12.2 "Programs and Computers Fail, the Systems Doesn't"
- 12.3 Interaction Failures
- 12.4 Techniques to Achieve System Fail-Safe
  - 12.4.1 Software Enhanced Functionality
  - 12.4.2 Limit the complexity of the Computer Operating System
  - 12.4.3 Avoid Interrupt Driven Operation
  - 12.4.4 Nesting
  - 12.4.5 Logic
  - 12.4.6 Programs should do Parameter Checking
  - 12.4.7 Flow Charts - State Diagrams
  - 12.4.8 Multiprocessor Systems
  - 12.4.9 Operator Input

Some early attempts at computer controlled Safety-Critical systems ended in tragedy. Accidents with some medical accelerator designs have led to a widely held belief among ES & H professionals that the use of computers in applications of this class is unwise. Attempts to blame system failure upon field wiring and input sensors has only compounded this belief. As we noted before, Computer programs are among the most complicated "machines" ever attempted by man; it is only to be expected that at some level of complexity, a developer loses the ability to comprehend the detailed functioning of his own program. With many more people having direct experience with their own PCs, fewer and fewer people are prepared to accept and assume that computer programs don't fail indeed the current bias may be that computers will always fail.

An account of medical electron accelerator accidents by M. Levenson and C. Turner contains the following useful lines " A lesson to be learned from the Therac-25 story is that focusing on particular software bugs is not the way to make a safe system. Virtually all complex software can be made to behave in an unexpected fashion under certain conditions." The authors go on to recommend independent hardware backup.

A technology independent and very useful concept is to assume that anything that can go wrong in a single computer or program will eventually happen.

System malfunctions generated by Software failures are rarely the result of an improperly coded program; problems usually result from the interaction of multiple program modules, the Operating system and/or inputs from the outside world. Classic failures include the following cases: an unexpected interrupt happens and prevents a vital program module from running or completing; a program sits waiting for a start message or interrupt which never occurs; a second interrupt occurs while another interrupt is being processed, information is lost or the system halts.

A PLC type operating system which follows a simple repetitive loop concept is less likely to fail than an interrupt driven or message passing operating system or a system in which tasks run concurrently.

One should limit the number of levels of subroutine nesting to three or at most four; it becomes too hard to trace all paths. Hierarchical logic flow is better than messaging logic in this type of application. Programs should validate their input; if a program receives the wrong or out-of-range values, it then acts to achieve a safe state. Whatever technique is employed, it needs to reflect the time and sequence dependent nature of the system being controlled or monitored. If a program is written which exchange information it needs to be implemented in such a manner that the actual information exchange itself cannot contaminate the interconnected processors. Use multiple processors which function as a very sophisticated "heart beat" or "dead man" circuits and thus achieve collective reliability greater than that of the individual computers.

Operator input routines should be written with the assumption that an Operator will perform the wrong response or will not indicate a response. Default cases must be carefully thought out and critical operator responses questioned and requisitioned. Power on and reset cases need very careful review.

At the present time, the design of the hardware aspects of our safety system is complete and the first generation of software is being written with preliminary code testing to begin during November 1994.

References

DOE 5480.25 "Safety of Accelerator Facilities (Order and Guidance Package)

An Investigation of the Therac-25 Accidents, Levenson and Turner, COMPUTER July 1993, pp. 18-41.

McCormick, N. Reliability and Risk Analysis, Academic Press 1981.