September 13, 1991

G. J. K. Asmis, Ph.D., P. Eng., Manager
Safety Evaluation Division (Engineering)
Atomic Energy Control Board
Ottawa, Canada.
K1P 5S9

Dear Dr. Asmis,

SUBJECT: REVIEW OF PROPOSED STANDARD FOR COMPUTERS IN THE SAFETY SYSTEMS OF NUCLEAR POWER STATIONS (BASED ON IEC STANDARD 880) (TAC M40006)

Your letter dated August 2, 1991, requested comments on the subject proposed standard. Members of the Instrumentation and Control System Branch reviewed the proposed standard. I am enclosing their comments for your information. No particular emphasis has been placed on the order the comments are presented.

Please note that these comments are personal comments based on the individual's experience and do not represent any U. S. Nuclear Regulatory Commission approved position.

Thank you for the opportunity to comment on the proposed standard, and please contact me if you should require any clarification on the comments.

Sincerely,

Original signed by:

Scott Newberry, Chief
Instrumentation and Control Systems Branch
Division of Systems Technology

Enclosure:
Comments On Proposed Standard For Software For Computers In The Safety Systems Of Nuclear Power Stations (Based On IEC Standard 880)

DISTRIBUTION
PDR
Central File
SICB R/F
A. Thadani J. Joyce
M. Waterman J. Stewart
M. Chiramal L. Beltracchi
S. Newberry J. Gallagher

C4R-25
X RD-25
Nu Power

C70017

DF01

OFC	:SICB	:SICB	:SICB:DST	:	:	:	:
NAME	:MWaterman	:lm:MChiramal	:SNewberry	:	:	:	:
DATE	:9/12/91	:9/13/91	:9/13/91	:	:	:	:

OFFICIAL RECORD COPY
Document Name: CHIRAMAL/CANADA STANDARDS

September 13, 1991

G. J. K. Asmis, Ph.D., P. Eng., Manager
Safety Evaluation Division (Engineering)
Atomic Energy Control Board
Ottawa, Canada.
K1P 5S9

Dear Dr. Asmis,

SUBJECT: REVIEW OF PROPOSED STANDARD FOR COMPUTERS IN THE SAFETY SYSTEMS OF NUCLEAR POWER STATIONS (BASED ON IEC STANDARD 880) (TAC M40006)

Your letter dated August 2, 1991, requested comments on the subject proposed standard. Members of the Instrumentation and Control System Branch reviewed the proposed standard. I am enclosing their comments for your information. No particular emphasis has been placed on the order the comments are presented.

Please note that these comments are personal comments based on the individual's experience and do not represent any U. S. Nuclear Regulatory Commission approved position.

Thank you for the opportunity to comment on the proposed standard, and please contact me if you should require any clarification on the comments.

Sincerely,

Original signed by:

Scott Newberry, Chief
Instrumentation and Control Systems Branch
Division of Systems Technology

Enclosure:
Comments On Proposed Standard For Software For Computers In The Safety Systems Of Nuclear Power Stations (Based On IEC Standard 880)

DISTRIBUTION
PDR
Central File
SICB R/F
A. Thadani J. Joyce
M. Waterman J. Stewart
M. Chiramal L. Beltracchi
S. Newberry J. Gallagher

OFC	:SICB	:SICB	:SICB:DST	:	:	:	:
NAME	:MWaterman:lm	:MChiramal	:SNewberry	:	:	:	:
DATE	:9/12/91	:9/13/91	:9/16/91	:	:	:	:

OFFICIAL RECORD COPY
Document Name: CHIRAMAL/CANADA STANDARDS

ENCLOSURE

COMMENTS ON PROPOSED STANDARD FOR SOFTWARE FOR COMPUTERS IN THE SAFETY SYSTEMS OF NUCLEAR POWER STATIONS (BASED ON IEC STANDARD 880).

The proposed standard is entitled "Software for Computers ..," yet page 2 title states "Software Documentation ....". Since this standard discusses several items besides documentation, the title should be "Software for Computers ....".

Even though this proposed standard is based on IEC 880 there is a major departure from the approach taken for IEC 880 on the sections that deal with software requirements and development (design and coding) of safety system software.

The proposed standard places emphasis on the use of mathematical relations to document the relationships between the monitored and controlled plant (environmental) variables in the section that deals with computer system requirements. In a similar manner the document places emphasis on the use of formal methods to document the relationships between the computer system input and output variables in the section that deals with the computer system design. The standard then proceeds to describe how these two documents can be combined to establish the overall software requirements and the resultant software implementation (6.2-(5)) and the requirement for acceptable software (6.1-(6)). In summary the standard, with respect to sections 4, 5, and 6 is a tutorial on the use of formal methods to produce a mathematical relation between monitored and controlled variables that could be used as the bases for acceptance of the software for the computer system. (It is interesting to note that the rewrite of equation 6 to the form in 6a by introducing functional relations and natural language descriptions is an admission that the formal systems development is impractical in the absence of automatic verification tools to perform the proof stated in equation 6 - this latter conclusion was taken from the Software Reliability Handbook, Elsevier Applied Science, London and New York, 1990).

In contrast to the above, IEC 880, in the section on software requirements, presents requirements and guides that form the basis for software development. IEC 880 addresses software/hardware constraints, requirements for continuous supervision of software and hardware, interface requirements between the safety system and other systems, and computer system configuration attributes that yield a more reliable system. In a similar manner, the sections on development (design and coding) of safety system software include requirements and guidelines for the purpose of promoting design and coding practices that, through experience, have resulted in improved reliability of software or by prohibiting practices that have been a known source of either software error or a significant impediment to the verification activity-especially with respect to completeness of verification.

The proposed standard does not address hardware/software integration and, therefore, does not include the very important matter of verification of the integrated system, other than to require an integrated system verification test report. IEC 880 presents requirements for a system integration plan, system configuration control, integrated system verification, and error resolution procedures. In particular, the requirements address the question, should an error found during system integration been detected earlier, e.g. at the module verification level?

The proposed standard does provide better guidance on general requirements as they relate to the different phases of the computer system development; e.g., documentation, verification reviewers, and a summary of verification criteria.

RECOMMENDATIONS:

1. The main body of the standard could be revised to be more consistent with IEC 880 in the areas discussed above. The sections on formal methods could be included as an appendix that provides guidance on the (optional) general use of formal methods, specifically, for the development of the functional specification. For completeness there should also be an appendix that provides guidance on the use of object-oriented design for the development of the functional specification and the realization of the software architecture design. The most effective approach to achieve reliable software may be a combination of the two methods. Object oriented design for achieving the software architecture design and formal methods to ensure a precise definition of the relation between monitored and controlled variables. Formal methods may also be of value where the implementation language does not support a general model of inheritance.

2. A recommendation would be to endorse IEC 880 and use the proposed standard to discuss the differences, or add to this standard the other definitions from IEC 880, such as defense-in-depth, diversity, fault tolerance, etc.

SPECIFIC COMMENTS AND OBSERVATIONS:

1) This standard references ISO Std 2382/1 in its definition of computer systems, "Computers consist of processing units and peripheral equipment which is controlled by internally stored programs written in a general purpose computer language." Does this standard apply to programmable logic controllers, which use a machine-specific language?

2) How do application-specific integrated circuits (ASICs) fit into this standard?

3) The definition of module in this standard appears to rule out the use of libraries generated by another vendor. The "group of programmers requirement is ambiguous since there is no definition of what constitutes a group.

4) Section 2.2 (page 3) states that the computer programs are written in "a general purpose programming language". Is this intended to prohibit assembly language programming?

5) Section 3 - page 4 - The phases identified in this section do not correspond to the software life-cycle phases identified in definition 2.9.

6) The module definition in Section 2.5 should either be expanded to provide a more explicit description of what is expected or a reference should be provided to the information hiding principles for software module design.

7) The last sentence in Section 2.7 concerning simultaneous failures of redundant software is a significant statement that appears to imply that additional steps are needed to be taken (such as diversity) to meet the single failure criterion. This does not appear to be discussed in any further detail in the document. Further discussion on this and acceptable methods to meet single failure criterion should be explicitly discussed in the standard.

8) In Section 2.9, the term "software life cycle" instead of "software life time" could be used to be consistent with IEEE and IEC usage.

9) Section 3.1.3 states that every product shall be systematically checked after each phase. However, the standard does not define what constitutes a systematic method of checking products.

10) Section 3.1.5 states that each phase shall be terminated by critical review. "Critical" is not defined, and is therefore ambiguous.

11) The software quality assurance (SQA) discussion provides general guidance for a SQA plan. The guidance should reference other standards for more detailed guidance.

12) Section 4 requires mathematical statements of environmental variables. Among the variables listed as examples is administrative information, such as the number of personnel assigned to the task. How can this variable be implemented as a mathematical environmental variable? Administrative information is not an environmental variable in this context.

13) The proposed standard does not demonstrate that the mathematical construct approach can be implemented across the industry.

14) The proposed standard emphasizes time-functions as the input and output variables. Can non-time-functions also be important? Time-function is not defined in this standard.

15) This standard imposes rigid mathematically defined constraints on a system. Is this necessary to ensure adequate system development?

16) This standard appears to be too esoteric, and it may be difficult to verify correct implementation. Note that any data-based system or recursive system will result in a very complex mathematical statement. Most systems data-base their data. Feedback (recursion) is also used extensively in controls.

17) With the exception of the mathematical descriptions requirements, there is nothing in this standard that hasn't been stated by at least three other standards bodies.

18) This document appears to require that formal methods and notations be used. Yet, this appears to be contradicted by Section 6.1 which states that natural language is unavoidable.

19) Module structure, Section 8, uses "information hiding" as if it were a new concept. Carefully constructed libraries and subroutines provide the same function.

20) The software module guide requirement is specified in other standards under different names.

21) The purpose of a standard is to clearly and unambiguously define the limits within which the user must operate. The technical complexity of this standard is such that there may be only a limited number of people who can implement or review the validity of the output products.

22) Section 10 states that each module will be implemented by a "private data structure" and one or more programs. There is no definition of what constitutes a "private data structure".

23) Most of the references in the proposed standard are for publications by Mills and Parnas. Other industry experts, such as Fagan and Leveson, have not been included in the bibliography. The scope of information may be too limited by these omissions.

24) Section 12 provides a good requirement regarding testing of compilers and languages.

25) Section 13.2 - Verification Plan - states that the verification plan shall describe the activities to be performed to evaluate whether the reliability requirements specification has been met. The standard does not describe the process for defining quantitative reliability criteria. Should reliability be quantitatively specified?

26) Section 13.3 - Verification Personnel - states that the qualifications of the personnel should overlap in adjacent phases, otherwise the verification cannot be met. This is consistent with the discussion of the verification phase, except for the Program Design phase , which uses experts that are not used in any other phase.

27) Section 13.4, Verification Criteria, provides a table without quantitative values for the guidelines. This is a shortcoming of other standards as well.

28) Sections 13.5, Software Test Specification, Section 13.6, Software Test Report, and Section 13.7, Integrated System Verification Test Report, Section 14, Computer System Validation, and Section 15, Maintenance and Modification, provide well defined guidance.