3 1176 00147 8677

NASA Technical Memorandum 80236

NASA-TM-80236 19800012561

SUMMARY REPORT OF WORKSHOP ON REAL-TIME PROGRAMMING FOR NASA FLIGHT PROJECTS

Cosponsored by the Institute for Computer Applications in Science and Engineering and NASA Langley Research Center

FOR REFERENCE

NOT TO BE TAKEN FROM THIS ROOM

John C. Knight

February 1980

NASA
National Aeronautics and Space Administration

Langley Research Center
Hampton, Virginia 23665

INTRODUCTION

A workshop on real-time programming for NASA flight projects was held in Hampton, Virginia, from October 17 to October 19, 1979. It was jointly sponsored by NASA Langley Research Center and the Institute for Computer Applications in Science and Engineering (ICASE). This report presents a brief summary of the workshop.

The workshop brought together representatives of NASA flight projects, industry, and researchers from universities to discuss the problem of using high-level languages to program the critical real-time parts of flight software systems. The goals were:

1. For the participants from flight projects to give descriptions of various NASA flight systems as examples of the requirements imposed on high-level languages.

2. For the university participants to give descriptions of existing and proposed high-level language designs which might be suitable for programming of real-time systems.

The flight projects represented were SHUTTLE, HiMAT, GALILEO, TCV, SPACE TELESCOPE, VOYAGER, Modular Multi-mission Spacecraft (MMS), SPACE LAB, and the Annular Suspension Pointing System (ASPS). The programming languages discussed were HAL/S, Ada, Concurrent PASCAL, Path PASCAL, GYVE, and MODULA.

Tabular comparisons of the more significant aspects of the projects and of the languages would be desirable but inappropriate. The projects described were very diverse. Some were almost complete, some were under development, and some were still being planned. Similarly, the programming languages varied from those of older design which have been in use for some time to new designs which are incomplete and have not been implemented.

Many topics were discussed during the workshop and this report summarizes those discussions. The workshop program is given in appendix A. A list of attendees and their affiliations is given in appendix B.

CHARACTERISTICS OF NASA FLIGHT SYSTEMS

The organizers of the workshop had hoped that a concise but comprehensive set of requirements could be derived from the descriptions of flight systems. This would have allowed a set of simple guidelines to be drawn up for language designers. The diversity of requirements which were described was surprising, and in many cases, two different projects had requirements which were the opposites of each other.

N80-21045#

A key parameter of real-time systems is the "frame time" or the time that the system takes to perform one cycle. Projects described at the workshop included frame times from a few tens of milliseconds to several hours. In addition, it is often the case that a system will use more than one frame rate where different response rates are required. Most of the projects described employ multiple frame rates.

There are two different approaches to the provision of real-time service. They are referred to as sychronous and asynchronous processing. In a synchronous system, the individual processes which have to be executed during a given frame are executed to completion in a predetermined order. At any given time, only one process is in execution. This contrasts with asynchronous systems where several, perhaps all, of the processes to be executed in a frame are executing concurrently. A priority mechanism may be used, and a processor dispatching algorithm is used for processor management.

Considerable debate about the merits of these two forms of processing arose at the workshop and no conclusions were reached. The advantages of synchronous operation cited by workshop participants were simplicity of organization and reliability. Since processing is essentially sequential, synchronous systems are simpler to test. Asynchronous systems offer flexibility since processes are not active unless they need to be. This leads to a serious problem that was raised by many workshop participants. A set of processes could be initiated for which there is sufficient processor time available in principle, but such that specific deadlines can be missed under extreme or unexpected conditions. This situation may never arise, but the possibility of overcommitting the processor is very undesirable.

As well as the variations in software organization discussed above, there is a great variability in the overall hardware organization being used on flight projects, and this affects the software substantially. Older projects relied on a single processor, but with increasing mission complexity and vastly reduced hardware costs, many recent projects incorporate several processors. The Galileo command and Data Subsystem, for example, uses six microprocessors. The use of many processors which do not share memory is in effect a network, and this raises the need for communications software and the necessary high-level language facilities to handle it.

As in many other areas of computer application, there are differences in software organization even more fundamental than the above. Most systems provide an extremely simple interface to the command uplink using a small number of commands and a simple command structure. This allows relatively simple on-board software but limits flexibility. At the other extreme is project Galileo which will provide a sophisticated command programming language and the on-board software has to be constructed as an interpreter for a coded form of the command language.

PROGRAMMING LANGUAGES

The participants of the workshop were generally agreed on the need for high-level languages, as would be expected. However, it was stressed that flight projects cannot select an unproven programming language for use even if it appears appropriate. Flight projects typically are tied to a set of critical dates, and delays incurred because of unforeseen problems in the use of a new, untried programming language cannot be tolerated. In addition, the lack of readily available compilers for flight computers limits the use of high-level languages. Compilers are expensive programs to write, and the software budget for most flight projects is not large enough to fund the development of a new compiler.

The concurrency of the flight software described was limited, and it seems that the synchronization and exclusion constructs introduced into modern programming languages are probably sufficient. A key element of flight software is the notion of time, and this seems to be very poorly handled from the flight software viewpoint, except in HAL/S. Most programming languages do not provide for explicit scheduling in real time, but require the user to program the required scheduling using lower level facilities. For example, it is often necessary to use priorities to impose an implied schedule since explicit scheduling cannot be done. While this is adequate, explicit provision for the easy use of time in a programming language seems very desirable. Flight software is very closely tied to time, both in small amounts such as frame rates of less than a second, and in large amounts such as mission schedules which may be many hours or days in length.

PRESENT USE OF PROGRAMMING LANGUAGES

The HiMAT project is making some use of FORTRAN with extensions for real-time, and the experience to date is apparently satisfactory. Apart from that, HAL/S is the only high-level language being used by any of the projects represented at the workshop. The way HAL/S is used varies a great deal. SHUTTLE, for example, uses it for almost all of the software in the flight computers. ASPS uses it for applications programs, and the executive system is written in assembly language. Some projects are at the stage of considering HAL/S but have not made decisions on exactly what it will be used for. These decisions may be affected by the availability of compilers.

In a more general sense, the use of software tools varied considerably from project to project. One ongoing flight project is presently writing all software in assembly language and using an assembler which produces absolute code. Thus, any software changes require the reassembly of the entire system. On the other hand, project Galileo is making extensive use of HAL/S and is routinely using a software design language (SDDL).

It is important to realize that there are great benefits to be gained from using well tried and proven software development techniques. Projects which ignore this are wasting their time investigating sophisticated high-level languages.

COMPUTER HARDWARE--PRESENT AND FUTURE

A major source of difficulty in developing flight computer systems is the lack of flight-rated computer hardware. The strict weight, size, temperature, vibration, electrical power, and radiation requirements can be met by very few processors. It is often the case that a processor is selected because it is the only one available rather than because it especially suits the mission requirements.

While this situation is unfortunate, it is understandable. A relatively small market exists for processors which meet all the necessary constraints, and manufacturers are reluctant to pursue this market. From the software viewpoint, there are also important constraints. For example, memory size is usually less than desired, processor speed is often limited, and the software has to be as reliable as possible. Many processor design concepts have been devised which would be valuable under these circumstances, but they are usually missing from the processors which are available for flight projects. A special session was held at the workshop to discuss this situation, and a list of desirable hardware characteristics for flight computers was prepared. This list of requirements is not complete and was prepared informally, but it does indicate the degree of dissatisfaction with computer hardware felt by those involved with flight software. The list of desirable characteristics is:

1. A self test capability built into the chip for large scale integrated circuits.

2. Easy to use, efficient static relocation. There is no need for dynamic relocation or virtual memory.

3. Easy method for external monitoring of the internal operation of processors and memories during testing.

4. Complete and precise description of hardware behaviour under all circumstances. No use of phrases such as "undefined results."

5. Good, flexible memory protection, possibly a tagged memory architecture.

6. Accurate, high resolution clocks and timers yielding information in a useful format. Instructions for READ, MODIFY, WRITE sequences of clocks and timers.

7. Floating point instructions and a single, adequate floating point length and format. Care and attention to the floating point hardware algorithms.

8. A fixed point capability is unnecessary and undesirable.

9. Large, easy to use address space.

10. Compatible range of computers of increasing power with either identical instruction sets or upward compatible instruction sets.

11. Flexibility in the hardware which can be used easily. Advertised flexibility which does not perform adequately when needed is less useful than no flexibility.

12. Comprehensive interrupt structure.

CONCLUSION

Several useful conclusions can be drawn from the presentations and discussions held at this workshop. In summary, the major points are:

1. Existing and planned projects present a very diverse set of flight software requirements.

2. In most cases, existing programming languages do not seem to be well suited to the preparation of real-time software.

3. The presently available computer hardware for flight systems omits many facilities which would be of great value to the software.

This workshop was motivated by the realization that there will be a substantial growth in the need for real-time flight systems in the near future. The number and range of space missions will increase dramatically because of the improved launch capability provided by the Shuttle, and there will be much greater use of digital avionics systems in air transports. The onboard computing that can be used on all of these projects will be substantially greater than in the past because of the reduction in computer hardware costs.

The preparation of all of the required software still presents a difficult problem. High-level languages offer part of the solution, but in the important area of real-time processing, it can be concluded from this workshop that the necessary modern high-level language facilities are not yet available for general use.

APPENDIX A

Program for Workshop on Real-Time Programming for NASA Flight Projects

WEDNESDAY, OCTOBER 17, 1979

8:30 - 8:55	Registration
8:55 - 9:00	Welcome

SESSION 1 - CHAIRPERSON, R. Voigt, ICASE

9:00 - 9:45	SPACE SHUTTLE (J. Garman, Johnson Space Center)
9:45 - 10:30	HiMAT PROJECT (A. Myers, Dryden Flight Research Center)
10:30 - 11:00	BREAK
11:00 - 11:45	HAL/S (M. Ryer, Intermetrics, Inc.)
11:45 - 12:30	ADA (R. Dewar, New York University)
12:30 - 1:30	LUNCH

SESSION 2 - CHAIRPERSON, S. Feyock, College of William and Mary

1:30 - 2:15	GALILEO (R. Loesh, Jet Propulsion Laboratory)
2:15 - 3:00	TERMINAL CONFIGURED VEHICLE (G. Boyles, Langley Research Center)
3:00 - 3:30	BREAK
3:30 - 4:15	CONCURRENT PASCAL (R. Noonan, College of William and Mary)
4:15 - 5:00	PATH PASCAL (R. Campbell, University of Illinois)
6:00	SOCIAL HOUR
7:00	BUFFET DINNER
8:00	OPEN DISCUSSION - Entitled "Why are the Computers We Get Always Turkeys" (World Series permitting)

THURSDAY, OCTOBER 18, 1979

SESSION 3 - CHAIRPERSON, R. Noonan, College of William and Mary

9:00 - 9:45	SPACE TELESCOPE (C. Balentine, Marshall Space Flight Center)
9:45 - 10:30	PROJECT VOYAGER (C. Jones and S. Lingon, Jet Propulsion Laboratory)
10:30 - 11:00	BREAK
11:00 - 11:45	GYVE (E. Schonberg, New York University)
11:45 - 12:30	MODULA (J. Knight, Langley Research Center)
12:30 - 1:30	LUNCH

SESSION 4 - CHAIRPERSON, S. Voigt, Langley Research Center

1:30 - 2:00	AN EXAMPLE FAULT TOLERANT SYSTEM (H. Hecht, SoHar, Inc.)
2:00 - 2:30	FAULT TOLERANCE IN CONCURRENT SYSTEMS (T. Anderson, University of Newcastle-upon-Tyne)
2:30 - 3:00	SPECIFICATION AND ANALYSIS OF FLIGHT SOFTWARE REQUIREMENTS AND DESIGNS (W. Riddle, University of Colorado)
3:00 - 3:30	BREAK
3:30 - 4:15	GALILEO COMMAND AND DATA SUBSYSTEM (T. Clarkson, Jet Propulsion Laboratory)
4:15 - 5:00	MODULAR MULTI-MISSION SPACECRAFT (T. Taylor, Goddard Space Flight Center)

FRIDAY, OCTOBER 19, 1979

SESSION 5 - CHAIRPERSON, J. Knight, Langley Research Center

8:30 - 9:15	SPACE LAB (G. Settle, Marshall Space Flight Center)
9:15 - 10:00	CASE STUDY (G. Danaraj, ICASE)
10:00 - 10:15	BREAK
10:15 - 12:00	OPEN DISCUSSION AND REACTION TO THE WORKSHOP

APPENDIX B

Attendees
at the
Workshop on Real-Time Programming
For NASA Flight Projects

October 17 - 19, 1979

Anderson, Thomas
University of Newcastle-upon-Tyne

Balentine, Chapman
NASA George C. Marshall Space
Flight Center

Barrett, Curtiss C.
NASA Goddard Space Flight Center

Bokhari, Shahid
ICASE

Boyles, George B.
NASA Langley Research Center

Bulle, Richard L.
NASA Langley Research Center

Campbell, Roy H.
University of Illinois

Clarkson, T. B.
Jet Propulsion Laboratory

Clarson, John
College of William and Mary

Collins, Robert
Computer Sciences Corporation

Danaraj, J. Gopal
ICASE

Dewar, Robert
Courant Institute

Dunning, Larry A.
Old Dominion University

Eppley, Stephen M.
Jet Propulsion Laboratory

Feyock, Stefan
College of William and Mary

Fish, Vern R.
TRW, Defense and Space Group

Foudriat, Edwin C.
NASA Langley Research Center

Fulk, Mark
Courant Institute

Garman, John (Jack) R.
NASA Johnson Space Center

Halterman, Karen
OAO Corporation

Hecht, Herbert
SoHaR, Incorporated

Hiraishi, Kenji
University of Illinois

Hoffberg, Susan
The Perkin-Elmer Corporation

Howle, William M.
NASA Langley Research Center

Jones, Christopher P.
Jet Propulsion Laboratory

Kim, Kwang Hae
State University of New York

Knight, John C.
NASA Langley Research Center

LaBaugh, Robert J.
Martin Marietta Aerospace

Lahn, Thomas G.
Honeywell, Inc.

Loesh, Robert E.
Jet Propulsion Laboratory

Lowry, R. David
The Perkin-Elmer Corporation

Madden, M. G.
Lockheed

Margolis, Susan
Computer Sciences Corporation

Martin, Fred H.
Intermetrics, Inc.

Mathis, Robert F.
Old Dominion University

Myers, Al
NASA Dryden Flight Research Center

Ness, W. G.
Lockheed-Georgia

Nikora, Allen P.
Jet Propulsion Laboratory

Noonan, Robert E.
College of William and Mary

Parks, C. Lucille
NASA Langley Research Center

Pratt, Terrence
University of Virginia

Riddle, William E.
University of Colorado

Rose, Milton E.
ICASE

Ryer, Michael
Intermetrics, Inc.

Schonberg, Edmond
Courant Institute

Schwartz, Jacob
Courant Institute

Senn, Edmond H.
NASA Langley Research Center

Settle, Gray L.
NASA Marshall Space Flight Center

Smith, Burton K.
Rockwell International

Taylor, Thomas D.
Goddard Space Flight Center

Thomas, Jack
Jet Propulsion Laboratory

Van Orden, Stuart
OAO Corporation

Voigt, Robert G.
ICASE

Voigt, Susan J.
NASA Langley Research Center

Whittier, W. H.
Lockheed Missiles and Space Co.

Williams, J. Randy
NASA Langley Research Center

1. Report No. NASA TM 80236	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Summary Report of Workshop on Real-Time Programming for NASA Flight Projects		5. Report Date February 1980
		6. Performing Organization Code
7. Author(s) John C. Knight		8. Performing Organization Report No.
		10. Work Unit No. 505-31-83-02
9. Performing Organization Name and Address NASA Langley Research Center Hampton, VA 23665		11. Contract or Grant No.
		13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

This report summarizes a workshop on real-time programming which was held in October 1979, and was cosponsored by the Institute for Computer Applications in Science and Engineering and NASA Langley Research Center. The workshop brought together representatives of NASA flight projects and researchers working on the design and implementation of programming languages. The goal was to discuss the role of high-level languages in the preparation of flight software which is to operate in real time.

17. Key Words (Suggested by Author(s)) Flight computers Programming languages Real-time Workshop		18. Distribution Statement Unclassified--Unlimited Subject Category 61	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 9	22. Price* $4.00

N-305